UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO DE JOINVILLE
ENGENHARIA MECATRÔNICA

VITOR UCHIKAWA

ANÁLISE DO TRÁFEGO PUBLICITÁRIO COM ENFOQUE NO PROTOCOLO QUIC

Joinville
2024

VITOR UCHIKAWA

ANÁLISE DO TRÁFEGO PUBLICITÁRIO COM ENFOQUE NO PROTOCOLO QUIC

Trabalho apresentado como requisito parcial para obtenção do título de bacharel em Engenharia Mecatrônica, no Curso de Engenharia Mecatrônica, do Centro Tecnológico de Joinville, da Universidade Federal de Santa Catarina.

Orientador: Dr. Ricardo Jose Pfitscher

Joinville
2024

Este trabalho é dedicado aos meus colegas de classe e aos meus queridos pais.

AGRADECIMENTOS

Gostaria de expressar minha gratidão ao Prof. Dr. Ricardo José Pfitscher por suas orientações e pelo constante suporte ao longo deste trabalho. Agradeço também à minha família pelo apoio e pela motivação que me deram durante todo este processo.

“ *Todos nós estamos agora conectados pela Internet, como neurônios em um cérebro gigante.* ”

Stephen William Hawking

RESUMO

Diante do avanço das tecnologias comunicacionais, expande-se o montante de conteúdo disponível online sob circunstâncias reconhecidas como gratuitas, não obstante, são expostas publicidades como meio de monetização. O tráfego gerado por esse fluxo adicional pode propiciar sobrecarga nos enlaces e custos adicionais de franquias, estabelecendo interesse na determinação da parcela que esse compõe do todo, entretanto, em virtude da adoção de protocolos cifrados, à exemplo do YouTube com o QUIC, essa inspeção é dificultada. Sob esse pretexto desenvolveu-se neste trabalho meios de discernir as informações publicitárias difundidas dos conteúdos efetivamente requisitados, através de algoritmos de classificação e uso de inteligência artificial.
Como resultado das capturas simuladas, foram obtidas médias de 9,60% de propaganda, com um desvio padrão de 7,41%. Além disso, o modelo Random Forest apresentou uma acurácia de 99% no conjunto de dados gerado, embora seja necessário validar esses resultados em datasets mais extensos.

Palavra-chave: Propaganda; redes; classificação; inteligência artificial.

ABSTRACT

Given the advancement of communication technologies, the amount of content available online is expanding under circumstances recognized as free, however, advertisements are displayed as a means of monetization. The traffic generated by this additional flow can cause overload on links and additional franchise costs, establishing an interest in determining the portion that this makes up of the whole, however, due to the adoption of encrypted protocols, such as YouTube with QUIC, this inspection is difficult. Under this pretext, this work developed means of discerning the advertising information disseminated from the content actually requested, through classification algorithms and the use of artificial intelligence.
As a result of the simulated captures, averages of 9,60% advertising were obtained, with a standard deviation of 7,41%. Furthermore, the Random Forest model presented an accuracy of 99% on the generated dataset, although it is necessary to validate these results on more extensive datasets.

Keywords: Advertisement; networks; classification; artificial intelligence.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

ASCII	American Standard Code for Information Interchange
AUC	Area Under Curve
CSV	Comma Separated Values
CSS	Cascading Style Sheets
DNS	Domain Name Service
HTML	Hyper Text Markup Language
IETF	Internet Engineering Task Force
IP	Internet Protocol
ISP	Internet Service Provider
LSTM	Long Short-Term Memory
RFC	Request for Comment
TLS	Transport Layer Security
UDP	User Datagram Protocol

SUMÁRIO

1 INTRODUÇÃO

A internet pode ser descrita como uma infraestrutura de redes voltada a fornecer conexão entre aplicações distribuídas (Kurose; Ross, 2021). Com o aumento da globalização, sua conexão chegou a 90% das residências brasileiras (Nery; Britto, 2022), possibilitando acesso a variados conteúdos de maneira frequentemente gratuita.

Em contrapartida à disponibilidade pública, propagandas são incorporadas no corpo de websites como meio de monetização. No quarto trimestre de 2022, a plataforma YouTube obteve a receita líquida de US$ 13,62 bilhões (Spangler, 2023), mostrando rentabilidade do sistema, todavia, o download desses comerciais também pode resultar em sobrecarga indesejada nas redes do ponto de vista do consumidor.

Para o carregamento de páginas transferidas pela internet, são necessários os downloads dos objetos que a compõem, o que inclui publicidades. Zietz e Redel (2021) filtraram essa parcela no tráfego de um Internet Service Provider (ISP), constatando 18% do volume voltado a esse fim, entretanto, nessa análise foi omitida a parcela cuja as requisições DNS são cifradas, abrangendo o trafego codificado em QUIC.

Técnicas de criptografia estão presentes nos protocolos que mediam a troca de mensagens entre o servidor e cliente devido ao encaminhamento de informações estar sujeito a enlaces compartilhados por diferentes grupos. Em decorrência do uso desses mecanismos, a confiabilidade e integridade são garantidas ao mesmo tempo que há a devida interpretação dos dados nos receptores, entretanto, a análise de tráfego por gerenciadores intermediários é dificultada, uma vez que não possuem as informações necessárias para decifrar o conteúdo (Kurose; Ross, 2021).

Tendo em vista a expressividade dos anúncios online e sua disposição criptografada, propõem-se determinar a contribuição desses elementos para o fluxo de dados capturado, utilizando tanto arquivos de captura de um computador pessoal quanto dados de um IPS anônimo disponibilizados na comunidade Kaggle, tendo enfoque nas promoções cifradas presentes no YouTube, a fim de discutir seu impacto em volume.

Para filtrar os datagramas provenientes de endereços resolvidos por consultas não protegidas, foi realizado um processo de comparação entre os endereços de origem com os de hosts conhecidos no mercado publicitário. Com relação aos anúncios do YouTube transmitidos em quadros de resoluções cifradas, desenvolveu-se uma rotina capaz de identificar a reprodução desses comerciais e a transmissão de seus pacotes através da inspeção dos elementos da página, culminando na concepção de uma inteligência artificial classificatória treinada com os resultados desse programa. Tomou-se como base os trabalhos de Zietz e Redel (2021), procedendo com a observação dos dados transportados por um ISP, junto às produções de Akbari *et al.* (2021),

Tong *et al.* (2018) e Almuhammadi, Alnajim e Ayub (2023) para a classificação de informações criptografadas transmitidas pelo protocolo QUIC utilizado pelo YouTube.

1.1 OBJETIVOS

Para resolver a problemática da determinação do tráfego publicitário, propõem-se os seguintes objetivos.

1.1.1 Objetivo geral

Determinar a proporção de pacotes publicitários presentes em arquivos de captura obtidos de computadores pessoais e de um IPS anônimo, incluindo a fração criptografada a utilizar o protocolo QUIC adotado pelo sítio eletrônico YouTube, por meio de uma rotina de acesso controlada e uma inteligência artificial de aprendizado supervisionado.

1.1.2 Objetivos Específicos

- Estabelecer um programa de navegação na web que simule o histórico de um computador pessoal e seja capaz de identificar os momentos de reprodução de publicidade no YouTube;
- Elaborar um algoritmo de seleção de pacotes com base nos IPs presentes em resoluções DNS de nomes publicitários conhecidos e nos momentos de captura dos pacotes presentes nos intervalos previamente definidos de exibição comercial;
- Desenvolver um modelo eficaz de classificação artificial;
- Avaliar o impacto do tráfego estabelecido relacionado a anúncios.

2 FUNDAMENTAÇÃO TEÓRICA

De modo a amparar o entendimento da inteligência artificial, algoritmos de categorização e automação a serem concebidos, serão descritos neste capítulo os conceitos relacionados a esses. A seguir explica-se como se procede o acesso à internet, junto com os protocolos nela executados, as ideias de inteligência artificial, propaganda online, além da exposição sobre as ferramentas Wireshark e DPKT a serem empregadas.

2.1 NAVEGAÇÃO NA WEB

Para o acesso à internet, dispositivos são conectados à ISPs por diferentes meios, sejam por linhas telefônicas ou até fibra óptica. Esses enlaces são, então, multiplexados para o uso de múltiplos usuários por meio de técnicas como divisão de bandas de frequências (Tanenbaum; Wetherall, 2011).

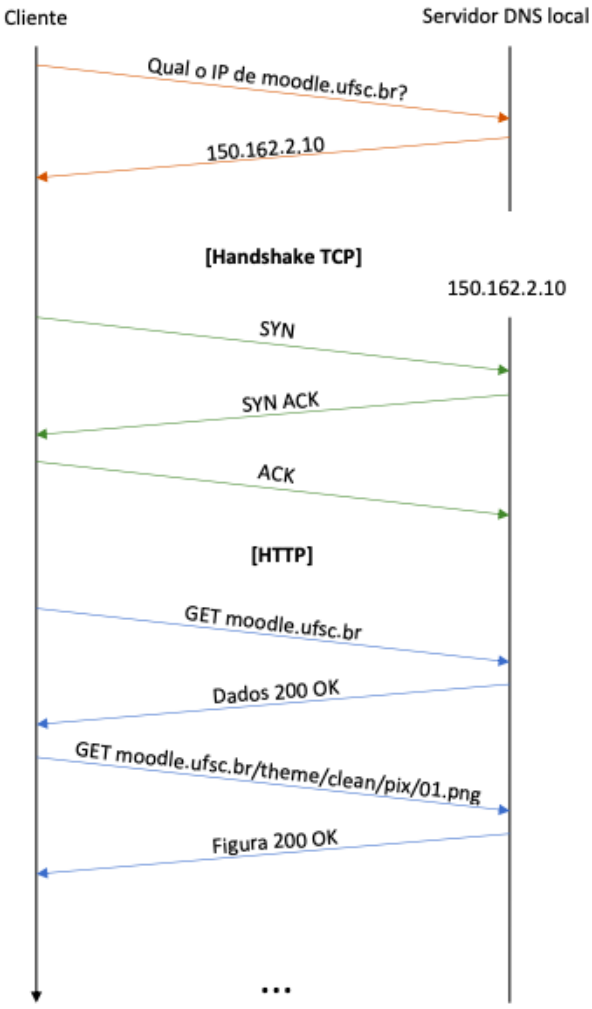
Uma vez conectados, é possibilitado o acesso a Web, qualificada como uma aplicação voltada a permitir o download de documentos remotos. Para essa transferência, um dos protocolos utilizados é o HTTP concomitante à adoção de dinâmica cliente-servidor, reconhecida pela existência de uma entidade servidor de número Internet Protocol (IP) fixo e permanentemente ativa, designada a atender as requisições dos clientes (Kurose; Ross, 2021).

Para a obtenção do IP desses data centers são empregadas requisições Domain Name Service (DNS) (Figura 1), que tem por função a tradução dos nomes de hospedeiros, como www.youtube.com, digitados no browser para endereços IP. Também são proporcionados por esse serviço, a interpretação de apelidos em nomes canônicos e distribuição de carga, uma vez que uma URL pode ser disponibilizada em formato simplificado para uso público e sua página estar presente em mais de um servidor, viabilizando listar resoluções com IPs de centros menos congestionados (Kurose; Ross, 2021).

Assim como os demais protocolos online, o DNS é detalhado em Request for Comments (RFCs) catalogadas na Internet Engeneering Task Force (IETF), junto com suas definições. Nesse documento é estabelecido uso em sua camada adjacente do User Datagram Protocol (UDP), encarregada a entregar os pacotes recebidos na máquina aos seus respectivos processos (Mockapetris, 1987).

Os processos que queiram realizar consultas iniciam conexão com os servidores na porta 53, sem um handshake inicial com a apresentação da entidade, já enviando os segmentos com campos reservados ao hostame e IP a ser traduzido pelo servidor, ambos estabelecidos no protocolo DNS. Após a chegada das respostas dos

Figura 1 – Procedimento de download de páginas online



Fonte: Elaborado pelo autor (2024).

data centers, os dados devem ser entregues aos devidos processos, para tanto, utilizando a dupla porta de origem e destino armazenadas na extensão do regulamento UDP do frame (Kurose; Ross, 2021).

Uma vez obtidos o endereço IP da página almejada, inicia-se o requerimento dos dados em si utilizando o protocolo HTTP, procedendo com a avaliação da necessidade de aquisição dos demais arquivos de origem oriunda, por exemplo, as propagandas de interesse deste trabalho. Para efetivar-se o download desses objetos remanescentes é plausível que ocorram demais requisições DNS prévias ao descarregar de seus efetivos conteúdos, capaz de desenrolar-se na mesma conexão ou de maneira sequencial, a depender da versão HTTP empregada (Pollard, 2019).

2.1.1 Versões do HTTP

O protocolo HTTP, além de empregar o TCP, esteve sujeito a modificações nos estágios de sua evolução. Em sua documentação inicial, referenciado como versão 0.9, teve estipulada sua conexão na porta 80, contendo uma linha em formato American Standard Code for Information Interchange (ASCII) com método GET criado para requisitar o documento em seguida referenciado, enfim, encerrando comunicação com seu recebimento (Pollard, 2019).

Com o transcorrer do tempo, foram publicadas as diretrizes das seguintes versões do HTTP para acomodar as demandas concebidas com o avanço da internet. Na versão 1.0 foram adicionados cabeçalhos de requisição para informações adicionais, como linguagens de marcação solicitada, códigos de erros, tal como *página não encontrada – 404*, métodos HEAD para obtenção de características sobre o arquivo procurado e POST para viabilizar envio de dados do cliente ao servidor. Sua versão subsequente 1.1, adicionou o campo de host a fim de viabilizar URLs relativas, devido à ascensão de sites com múltiplas telas, e incorporou conexões permanentes a possibilitar o download dos múltiplos objetos que passaram a compor páginas em uma única conexão ao invés de ter que restabelecê-la a cada item, como se procedia anteriormente (Pollard, 2019).

O HTTP/1, embora tenha obtido conexões persistentes, deparou-se com atribulações devido a intensificação de objetos a compor websites, necessitando baixá-los em seus downloads sequenciais, alavancando o desenvolvimento do HTTP/2 com enfoque em assincronismo. Nessa versão o protocolo passa a usar código binário em sua integridade, com mensagens enviadas em frames com identificadores de fluxo e de prioridade ao invés de serem constituídos por linhas ASCII e implementam push de servidor, de modo que o data center pode transmitir mais de uma resposta para uma requisição, eliminando a necessidade de demandá-las em demais mensagens GET (Pollard, 2019).

Apesar dos aperfeiçoamentos apresentados no HTTP/2, seu emprego sobre o TCP ainda trouxe limitações em respeito a performances, para tanto o protocolo QUIC, baseado em UDP, foi desenvolvido pela Google. Em 2021 foi publicado a RFC900 junto aos seus padrões alinhados às propostas da IETF tendo denominação HTTP/3 quando utilizado sobre o HTTP (Iyengar; Thomson, 2021).

Além da modificação dos sockets advinda do embaço do UDP, efetivaram-se melhorias descritas em sua documentação. A respeito do estabelecimento de conexões, tal passa a ocorrer com uma única troca de mensagens, apresentado entidades e trocando certificados de criptografia simultaneamente, também a incrementar o sistema de migração de conexão, descartando a necessidade de repetição do mecanismo, caso haja alteração do IP do cliente (Pollard, 2019).

Acerca da transmissão de dados, o QUIC obteve reestruturação no seu mo-

delo de frame, adotando os novos QPACKS. São criados múltiplos fluxos com identificadores distintos com o intuito de multiplexação, e embora baseie-se no UDP, continua a prover entrega confiável ordenada mediante reconhecimento de entrega por ACKs e execução em fluxos paralelos, de forma a permitir continuidade das demais correntes, mesmo que haja perda em determinada linha (Kurose; Ross, 2021).

Ao passo que foram realizadas melhorias em performance no HTTP também deve-se considerar a segurança fornecida pelo mesmo, assim configura-se o uso do HTTPS integrando o protocolo com o Transport Layer Security (TLS). São aplicadas técnicas de criptografia como meio de integridade, autenticação e sigilo (Kurose; Ross, 2021).

O TLSv1.2 aplicado no HTTP/2 segue a dinâmica de chaves públicas e privadas, por utilizar diferentes segredos nos processos de cifragem e decifragem. Uma vez que o cliente mande ao servidor sua capacidade de criptografia, o data center escolhe o TLSv1.2 e envia a chave do servidor a ser usada durante a conexão em conjunto com seu certificado. Por fim, são enviados por parte do cliente sua chave já criptografada, empregando o segredo anterior, passando a dialogar em instância integralmente codificada (Pollard, 2019; Kurose; Ross, 2021).

O HTTP/3 por sua vez emprega a versão TLS mais recente 1.3 para camada de segurança. Nessa nova edição são aperfeiçoadas o sistema de conexão para exigência de uma única excursão de dados no estabelecimento das configurações criptográficas, a trazer maior leveza ao procedimento, uma vez que a própria dinâmica de tradução já é laboriosa (Pollard, 2019).

2.2 INTELIGÊNCIA ARTIFICIAL

Com o advento do HTTP/3, foram viabilizadas consultas DNS sobre o protocolo QUIC, tornando inacessível o conteúdo das respostas. Adotou-se, assim, os conceitos de Norvig e Russell (2014) de inteligência artificial.

Dentre as definições de IA, é possível descrevê-la como um agente computacional que age de maneira autônoma no ambiente que está inserido. Para tanto, são usadas suas capacidades sensoriais para alcançar o melhor resultado na operação designada, que, neste caso, envolve a classificação de pacotes publicitários em uma captura passiva (Norvig; Russell, 2014).

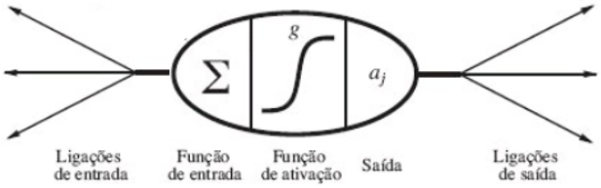
Para que uma IA atinja os objetivos propostos, ela é exposta a técnicas de aprendizado: no aprendizado não supervisionado, identifica padrões sem receber feedback direto; na instrução por reforço, ajusta parâmetros com base em recompensas e penalidades; no treinamento supervisionado, aprende observando exemplos e ajustando parâmetros do modelo aplicado conforme os casos apresentados (Norvig; Russell, 2014).

No contexto do aprendizado de máquina, o one-hot encoding é uma técnica fundamental para a representação de inputs categóricos em formato numérico necessário ao processo de treinamento. A título de exemplo, os protocolos TCP, UDP e TLS podem ser expressos como vetores binários [1, 0, 0], [0, 1, 0] e [0, 0, 1], respectivamente, evitando que os diferentes modelos de IA interpretem erroneamente a relação numérica entre as categorias (Ali, 2020).

2.2.1 Redes neurais

Redes neurais são modelos que possuem uma estrutura análoga ao cérebro humano e têm sido discutidos desde os primórdios da área. Nesses modelos, unidades chamadas de neurônios (Figura 2) recebem entradas com pesos definidos durante o treinamento. A saída de um neurônio é determinada pela soma ponderada das entradas, e esse valor resultante é então processado por uma função de ativação, como a função sigmoide, que determina o resultado (Norvig; Russell, 2014).

Figura 2 – Célula de uma rede neural



Fonte: adaptado de Norvig e Russell (2014, p. 843).

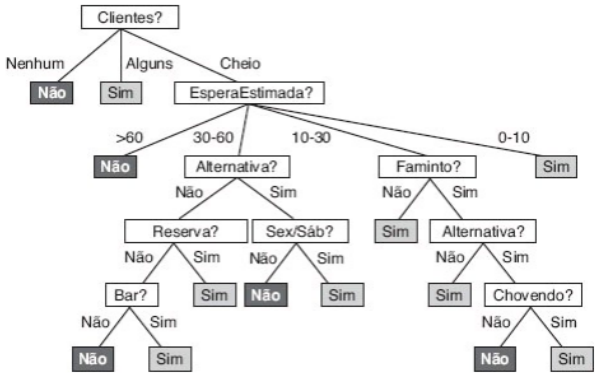
As células nervosas podem apresentar diferentes disposições de conexões entre si, de acordo com o intuito da aplicação. As redes organizadas em camadas possuem instância de entrada, saída e parcela intermediárias de células ocultas em configuração de avanço progressivo de sinais, visando estabelecer uma função da entrada, ou recorrentes, havendo outputs conectados a entradas, a fim de simular memória de curto prazo (Ceccon, 2020; Norvig; Russell, 2014).

2.2.2 Árvores de decisão

Além dos modelos baseados em redes neurais, também existem modelos baseados em árvores de decisão. Uma árvore de decisão é composta por nós e ramificações, conforme ilustrado na Figura 3 (Norvig; Russell, 2014).

Os nós representam pontos de decisão, onde são avaliadas as características dos atributos, enquanto os ramos determinam o fluxo de execução com base nas respostas obtidas, culminando na classificação final (Norvig; Russell, 2014).

Figura 3 – Árvore de decisão

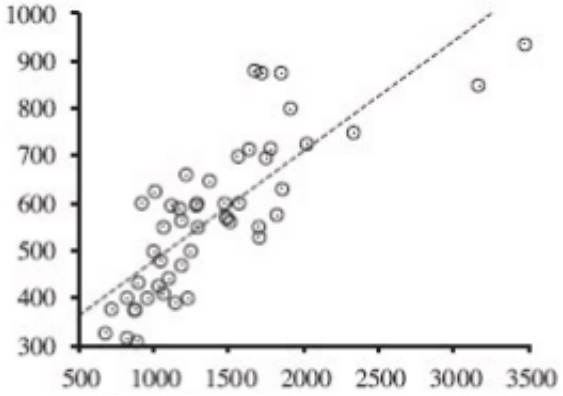


Fonte: adaptado de Norvig e Russell (2014, p. 812).

2.2.3 Classificadores lineares

O modelo baseado em classificações lineares assume a forma de uma função polinomial, onde as variáveis independentes representam as entradas e os coeficientes correspondem aos parâmetros calibrados na fase de capacitação (Norvig; Russell, 2014).

Figura 4 – Classificação linear univariada



Fonte: adaptado de Norvig e Russell (2014, p. 833).

A Figura 4 ilustra um exemplo de resultado desse tipo de modelo, mostrando como as classes são determinadas pela posição relativa dos dados em relação à reta ajustada (Norvig; Russell, 2014).

2.2.4 Avaliação

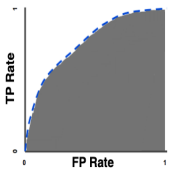
Com a intenção de averiguar a eficácia da estrutura criada, segue-se o método do traçado da evolução temporal do erro ao realizar seu teste com o banco de dados empregado no treinamento e outro distinto. Ao traçar a evolução do erro, é possível identificar a superadaptação do modelo quando o erro no conjunto de dados excluído do treinamento começa a aumentar, enquanto as predições no conjunto de

treinamento tornam-se progressivamente mais precisas, indicando que a capacidade ótima da rede foi ultrapassada (Norvig; Russell, 2014).

A avaliação do estado de treinamento também pode ser realizada por meio da validação cruzada, uma técnica na qual o conjunto de dados é fragmentado em k subconjuntos distintos. O modelo é então treinado k vezes, empregando k-1 subconjuntos como dados de treinamento em cada iteração, enquanto o subconjunto restante é reservado para teste (Ali, 2020).

Após a etapa de treinamento, o sistema também pode ter seu desempenho avaliado por diferentes métricas. Podem-se definir as métricas de acurácia, precisão, recall e área sob a curva (AUC) (Quadro 1).

Quadro 1 – Métricas de Avaliação de Modelos

Métrica	**Definição**	**Representação**
acurácia	Proporção das predições corretas em relação ao número total de estimativas realizadas.	$\frac{TP+TN}{TP+TN+FP+FN}$
Precisão	Proporção entre identificações positivas corretas em relação ao número total de predições positivas.	$\frac{TP}{TP+FP}$
Recall	Relação entre de positivos corretamente identificados pelo modelo em relação a todos os exemplos positivos presentes nos dados de teste.	$\frac{TP}{TP+FN}$
AUC	Área sob a curva traçada pela relação entre a taxa de verdadeiros positivos e a taxa de falsos positivos, sendo que a melhor performance ocorre quando se aproxima do valor unitário	
TP: verdadeiro positivo TN: verdadeiro negativo FP: falso positivo FN: falso negativo		

Fonte: Adaptado de Machine... (2024).

As métricas descritas no Quadro 1 permitem a comparação dos resultados de treinamento de diferentes arquiteturas, possibilitando a escolha da estrutura que melhor se adéque às necessidades da aplicação.

2.3 PROPAGANDAS NA INTERNET

No contexto dessas novas tecnologias, emerge o fenômeno contemporâneo do marketing digital, cujo impacto é estudado neste trabalho. Conforme definido por Chaffey e Ellis-Chadwick (2016), o marketing digital envolve a aplicação de tecnologias digitais e mídias para fins publicitários. Através do meio computacional, exploram-se

diversas possibilidades, como a identificação das necessidades dos compradores, a antecipação de suas demandas e a satisfação dos clientes por meio dos variados canais de divulgação online (Chaffey; Ellis-Chadwick, 2016).

Como vias de disseminação de sua marca, empresas podem recorrer a divulgação própria, através da palavra de influenciadores e por meio de mídias remuneradas. Essa terceira classe funciona de modo a posicionar anúncios no corpo de websites mediante a remuneração baseada no apuramento de clicks obtidos com tal (Chaffey; Ellis-Chadwick, 2016).

Seguindo o sistema de compensação respaldado em clicks, tem-se o Google Ads, bastando a criação de uma conta na interface seguida da abertura de uma campanha para instaurar um grupo de publicidade. A fim de efetivamente publicar os anúncios, são selecionadas as opções de palavras-chave e região geográfica de exibição para determinar os locais de exibição junto ao método de investimento, em consequência da disputa dos sítios, e montante disposto a pagar (Marshall; Rhodes; Todd, 2020).

Por intermédio de seu sistema de publicidade, a Google obtém parte considerável de sua renda, tendo destaque nos clipes em formato de vídeos inseridos antes de vídeos do YouTube. Nesse paradigma, entretanto, manifesta-se um volume de comerciais que pode tornar-se uma inconveniência para a experiência do usuário (Chaffey; Ellis-Chadwick, 2016).

Em resposta ao incremento de propagandas emergiram tecnologias bloqueadoras de anúncios em formato de extensões de navegadores e softwares. Esses algoritmos baseiam-se no confronto de domínios requisitados com listas de hosts vinculados ao marketing e regras de bloqueio que, para título de esclarecimento, cita-se a identificação de contêineres a serem obstruídos em definidos websites (Tomkevičiūtė, 2023). A exemplo dessas plug-ins, pode-se citar Nord e uBlock Origin disponíveis online (Gazvoda, 2023; Grigutytė, 2023).

2.4 TRABALHOS RELACIONADOS

Após a elaboração dos conceitos que definem o escopo deste trabalho, apresentam-se a seguir as pesquisas relacionadas de temática semelhante, as quais contribuíram com ideias para esta monografia.

Tong *et al.* (2018) propôs um método de classificação baseado em redes neurais convolucionais, alcançando uma precisão de aproximadamente 99% nas categorias estabelecidas. A abordagem incluiu uma etapa de categorização a partir das características do fluxo para identificar serviços de Chat e Chamada de Voz, seguida de uma análise baseada nas características dos pacotes para classificar os fluxos de rede em transferência de arquivos, streaming de vídeo e música do Google Play.

Almuhammadi, Alnajim e Ayub (2023) investigaram a classificação dos serviços do Google utilizando o tráfego QUIC, alcançando uma precisão superior a 96% através de modelos de ensemble baseados em árvores de decisão treinados com parâmetros de entrada extraídos de pacotes capturados em sua totalidade. Como sugestão para estudos subsequentes, recomendou-se a exploração de modelos alternativos e a utilização de ferramentas para ajuste automatizado de hiperparâmetros.

Akbari *et al.* (2021) exploraram os tópicos de redes neurais e tráfego criptografado, articulando uma arquitetura neural que utilizou filtros convolucionais aplicados aos dados de entrada junto a unidades Long Short-Term Memory (LSTM), capazes de reter informações de cálculos anteriores. Para validar essa abordagem, foram utilizados conjuntos de dados de uma operadora móvel e de um ISP, alcançando uma precisão de 95% na classificação dos serviços.

Uma monografia focada em propagandas online foi realizada por Zietz e Redel (2021), utilizando um algoritmo de classificação baseado nas requisições DNS presentes em uma captura também fornecida por um ISP. A abordagem revelou que 18,2% do conteúdo acessado na internet estava relacionado a comerciais, no entanto, a análise se limitou a pacotes cujos hosts foram traduzidos por consultas DNS convencionais com conteúdo em aberto.

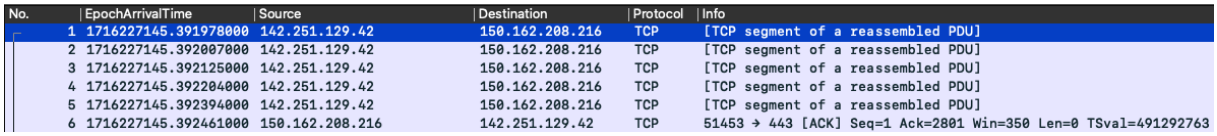
2.5 FERRAMENTAS

Após estabelecer os fundamentos deste trabalho, serão destacadas as ferramentas utilizadas em conjunto a esses conceitos para a captura e classificação dos dados, essenciais para o avanço de cada uma dessas etapas.

2.5.1 Captura passiva

Softwares conhecidos como sniffers possuem a habilidade de capturar tráfego de rede e oferecem opções de filtragem para assistir profissionais no diagnóstico de redes. Destaca-se o programa Wireshark (Figura 5) como líder nesse campo de aplicação, disponibilizado sob licença pública e equipado com uma interface gráfica a assegurar acessibilidade, ao passo que também estão disponíveis no setor o TCPDump, utilizando comandos de linha, e o SolarWinds mediante concessão privada (Chappell, 2012; Sikos, 2020).

Figura 5 – Interface do Wireshark



Fonte: Elaborado pelo autor (2024).

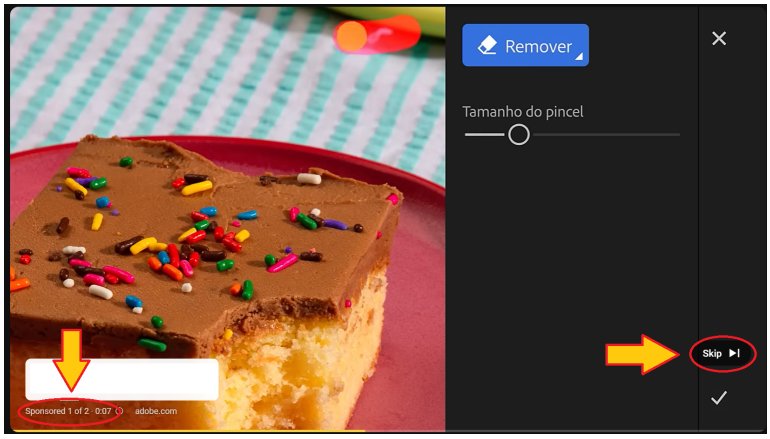
Após a captura dos pacotes, por intermédio dessa aplicação, é possível a visualização dos atributos atrelados a cada quadro propagado na rede. Pode-se observar na Figura 5 os endereços IP de origem e destino, as portas reservadas aos sockets, subordinando-os à processos como navegadores WEB, e campos pertencentes ao protocolo da camada de aplicação, estabelecendo a quantidade de bytes transmitidos, assim como o fluxo do qual o segmento faz parte (Chappell, 2012).

2.5.2 Captura ativa

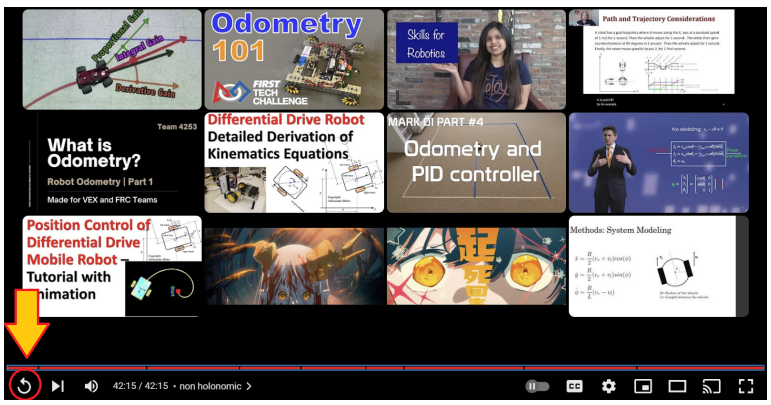
Em razão da possibilidade de ocorrerem solicitações DNS cifradas, nem todos os pacotes interceptados pelo Wireshark poderão ter seus IPs relacionados a hosts. Para lidar com esse problema, utiliza-se a biblioteca Selenium (Molina; Harsha; Fortner, 2024).

O controle da captura está baseado na lista dos sites mais visitados da internet, juntamente com a inspeção dos elementos Hyper Text Markup Language (HTML) (Figura 6) ao acessar vídeos hospedados no YouTube, a fim de verificar a presença de anúncios antes do conteúdo principal.

Figura 6 – Elementos HTML utilizados



(a) Elementos durante propaganda.



(b) Elemento após propaganda.

Fonte: Elaborado pelo autor (2024).

HTML é uma linguagem de marcação padronizada que define a estrutura das páginas da web, complementada por recursos como Cascading Style Sheets (CSS) e JavaScript. A presença de seções relacionadas a anúncios pode ser identificada pela localização dos elementos correspondentes no documento HTML, utilizando a biblioteca Selenium e seus identificadores relativos, conforme mostrado na Figura 6 (Molina; Harsha; Fortner, 2024; Pollard, 2019; Tanenbaum; Bos, 2016).

2.5.3 Classificação de Tráfego

Após a aquisição das capturas de tráfego, procede-se à etapa de classificação dos dados realizadas por intermédio das bibliotecas facilitadoras DPKT, Selenium e PyCaret.

A captura passiva incluiu os frames de resposta DNS contendo relações IP/host, que são posteriormente utilizadas pelos computadores que as requisitaram. Comparando essas relações com uma lista de servidores comerciais e utilizando a biblioteca DPKT para dissecção de pacotes em campos passíveis de análise em sintaxe Python, é possível catalogar os IPs pertencentes a esses servidores para uma avaliação subsequente considerando os tamanhos inscritos nos cabeçalhos (Zietz; Redel, 2021).

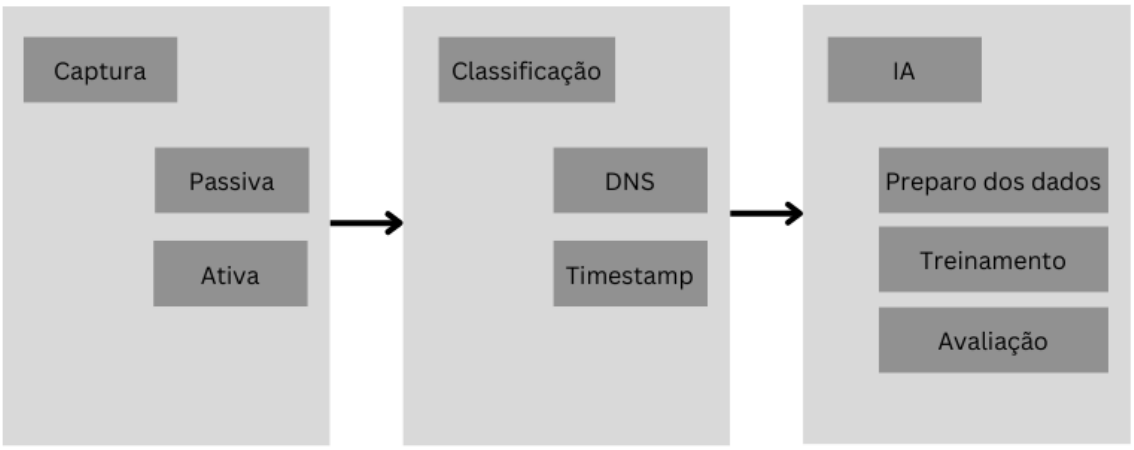
Para a captura ativa, projetada para mitigar a questão do DNS cifrado, são obtidos os intervalos de reprodução comercial através do método de inspeção de elementos pelo pacote Selenium, registrando a transmissão de seus pacotes. Juntamente com a classificação por DNS, são considerados na contabilização do tráfego comercial os pacotes interceptados entre os timestamps armazenados.

Os pacotes sinalizados por seus timestamps também colaboraram para o desenvolvimento da inteligência classificadora (Akbari *et al.*, 2021). Por intermédio do kit de desenvolvimento aberto PyCaret (Ali, 2020), dedicado a possibilitar o aprendizado em máquina em poucas linhas de script, é factível criar diversos classificadores baseados em diferentes arquiteruras e classifica-los de acordo com seus desempenhos em quesitos como precisão como desenvolvido neste trabalho e apresentado no capítulo quatro.

3 METODOLOGIA

Para abordar a problemática, adotou-se a sistemática apresentada na Figura 7, garantindo que cada etapa anterior suprisse as necessidades da seguinte, culminando no resultado final do consumo publicitário das capturas.

Figura 7 – Fluxograma da metodologia



Fonte: Elaborado pelo autor (2024).

Para uma melhor compreensão desse processo, este capítulo detalha os procedimentos para a concepção do banco de capturas, o raciocínio subjacente ao algoritmo e a metodologia empregada para desenvolver e treinar o modelo inteligente.

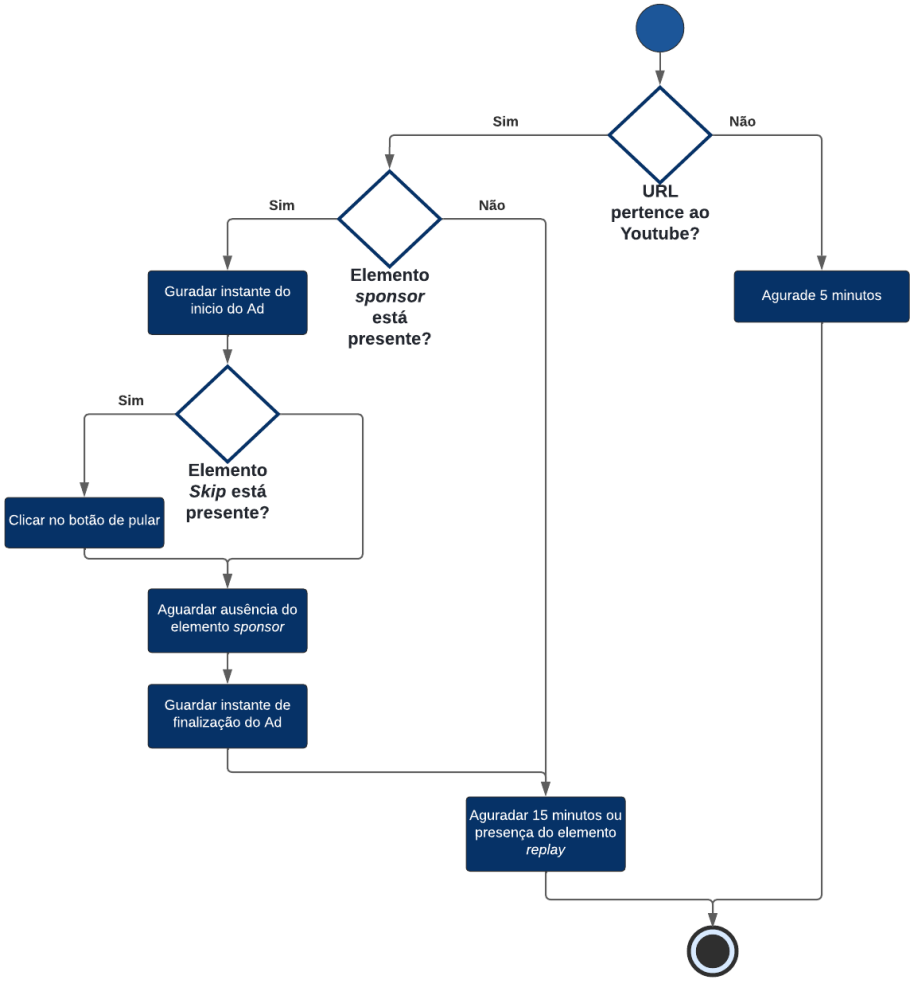
3.1 VISITA AUTOMATIZADA À PÁGINAS

Em razão da captura ser realizada a partir de uma máquina privada, por questões de reprodutibilidade e intenção de representar um fluxo residencial, foram acessados os portais mais procurados mundialmente disponíveis no repositório da companhia SimilarWeb[1]. Nos acessos direcionados ao YouTube, foram exibidos os vídeos categorizados em destaque pela plataforma.

Uma vez estabelecido o repertório de hiperlinks, definiu-se o script de acesso assistido pela biblioteca Selenium, designada para automatizar a operação do navegador web (Molina; Harsha; Fortner, 2024). A rotina (Figura 8) consistiu no início da captura da interface de rede, abertura do navegador web e visita dos websites mais visitados e vídeos em destaque, armazenando os intervalos de tempo em que anúncios são exibidos antes dos clipes propriamente requisitados.

[1] Disponível em: https://www.similarweb.com/top-websites/brazil/ Acesso em 20 maio 2024.

Figura 8 – Diagrama de atividades da rotina de visita



Fonte: Elaborado pelo autor (2024).

A automatização começou com a inicialização dos recursos a serem utilizados, sendo estes o sniffer, navegador, lista de links para acesso e lista de intervalos registrados com transmissão de propaganda, como mostra a Figura 9.

Na Figura 9, observa-se a abertura do arquivo de texto contendo os endereços dos websites e a criação do registro que será preenchido com os intervalos de tempo em que foram identificadas transmissões de publicidades. Em seguida, configurou-se a ferramenta de navegação Firefox com o perfil padrão para acessar os sites logados. A opção de reprodução automática de mídias, que não é ativada por padrão, foi passada como argumento. Por fim, criou-se o subprocesso de captura através da execução de um script bash.

Figura 9 – Código de inicialização dos recursos para visita

```
    init_prog_epoch = time.time()
    init_prog_str = time.strftime("%d_%m_%y__%H_%M", time.localtime())
    time_stamps_file = open("%s.txt" %init_prog_str, "w")

    websites_file = open("websites_list/websites.txt", "r")
    websites = websites_file.readlines()
    websites = [item.rstrip("\n") for item in websites]
    websites_file.close()

    yt_trending_file = open("websites_list/yt_trending.txt", "r")
    yt_trendings = yt_trending_file.readlines()
    yt_trendings = [item.rstrip("\n") for item in yt_trendings]
    yt_trending_file.close()

    option = Options()
    firefox_profile = FirefoxProfile('/home/vitor/.mozilla/firefox/ler32c1y.default')
    firefox_profile.set_preference('media.autoplay.default', 0)
    option.profile = firefox_profile
    driver = webdriver.Firefox(options=option)

    subprocess.Popen(["bash", "tshark_script.sh", "%s.pcap"%init_prog_str])
```

Fonte: Elaborado pelo autor (2024).

Prosseguindo à sessão de visitas, a URL extraída da lista deve ser testada para determinar-se o tratamento que deverá ser seguido (Figura 10).

Figura 10 – Código para visita de sites

```
    url = random.choices(websites, weights=normalized_weights, k=1)[0]
    if url.startswith("https://youtube"):
        youtubeUrlTreatment(driver, random.choice(yt_trendings), time_stamps_file)
    else:
        driver.get(url)
        time.sleep(website_wait_time)
```

Fonte: Elaborado pelo autor (2024).

Na Figura 10 é apresentado um procedimento onde um endereço selecionado aleatoriamente a partir de uma lista de websites é comparado com o endereço do YouTube. Se o endereço corresponder a uma página comum, ela é acessada e o sistema aguarda por um período de 5 minutos, intervalo determinado com base na média estabelecida em SimilarWeb. Nos casos em que o endereço visitado pertencia ao YouTube, foi aplicado um tratamento durante a visualização de um vídeo, levando em consideração os períodos de exibição de anúncios, reprodução do conteúdo e espera para acessar o próximo vídeo, juntamente com a opção de pular a propaganda caso fosse longa. Para identificar esses diferentes estados, foram inspecionados os elementos da página, conforme ilustrado na Figura 11.

Figura 11 – Tratamento para acesso ao YouTube

```
    try:
        WebDriverWait(driver, timeout=10, poll_frequency=.1).until(
        EC.presence_of_element_located((By.XPATH, sponsor_xpath)))

        AdPresence = True
        if verbose: print("Ad found at: ", start)

        element = WebDriverWait(driver, timeout=30, poll_frequency=.1).until
        (EC.any_of(EC.invisibility_of_element_located((By.XPATH, sponsor_xpath)),
                   EC.element_to_be_clickable((By.XPATH, skip_xpath)),
                   EC.element_to_be_clickable((By.XPATH, skip_xpath_alternate))))

        if not isinstance(element, bool): element.click()

    except TimeoutException as ex:
        if verbose: print("There was no Ad.")

    if AdPresence == False:
        if verbose: print("Actual video starting at", start)
    else:
        time_stamps_file.write("{} {}\n".format(start, time.time()))
        if verbose: print("Actual video strating at:", time.time())
```

Fonte: Elaborado pelo autor (2024).

Esclarecendo o programa da Figura 11, inicialmente o elemento *sponsor* é procurado por dez segundos. Quando for identificado no documento HTML, é estabelecido que um anúncio esteja em exibição. Em seguida, é averiguada a existência do botão de pular para que, se possível, seja pressionado, como um usuário comum procederia. Enfim, é aguardado o final da mídia, apontado pelo botão de recomeçar. Nesse algoritmo, a toda identificação de propaganda, o instante em formato Unix é salvo, assim como quando é terminada, facultado a posterior separação dos pacotes publicitários.

3.2 FILTRO DOS PACOTES PUBLICITÁRIOS

Para inspeção da parcela não cifrada, desenvolveu-se um script na linguagem de programação Python assistido pela biblioteca DPKT escrita por Song, Alperovich e Bellamy (2023) para interpretação do arquivo PCAP, procedendo com a correlação dos hosts presentes nas requisições DNS com uma lista de domínios renomados pela difusão de propagandas, a exemplo de Zietz e Redel (2021). Uma vez extraídos dos pacotes DNS os IPs dos domínios publicitários, esses endereços demarcaram os frames de interesse à pesquisa dentre os demais, permitindo estabelecer as estatísticas de volume almejadas.

O algoritmo consiste na inspeção das respostas DNS, classificação dos pacotes de acordo com a tradução de seu IP de origem para host, comparação com a

lista de domínios promocionais conhecidos, fornecida por Black (2024), e avaliação de correspondência entre o instante de chegada dos quadros com os intervalos de transmissão de marketing previamente estabelecidos.

Inicialmente, são carregados os intervalos publicitários salvos no arquivo texto *timestamps* em uma lista de tuplas a representar o instante de início e fim (Figura 12).

Figura 12 – Lista para intervalos publicitários

```
    ad_intervals: List[Tuple[float, float]]= []
    with open(time_stamps_file_name, "r") as time_stamps_file:
        for line in time_stamps_file:
            start, end = line.strip().split(" ")
            ad_intervals.append((float(start), float(end)))

    def isInAdInteval(time_stamp: float)->bool:
        index = bisect.bisect_left(ad_intervals, (time_stamp, time_stamp))
        if index > 0 and ad_intervals[index - 1][0] <= time_stamp and
        time_stamp <= ad_intervals[index - 1][1]:
            return True

        if index < len(ad_intervals) and ad_intervals[index][0] <= time_stamp and
        time_stamp <= ad_intervals[index][1]:
            return True
        return False
```

Fonte: Elaborado pelo autor (2024).

Para verificar se um determinado instante está contido em algum outro intervalo, a Figura 12 apresenta a implementação de uma função que retorna um valor booleano, examinando os componentes da lista em busca de algum que englobe o instante fornecido.

Após a classificação baseada nos instantes, estabeleceu-se a relação entre os endereços IP dos pacotes com os hosts de servidores publicitários resolvidos por meio de consultas DNS anteriores à transmissão desses pacotes. Tornou-se necessário implementar uma classe para armazenar as relações entre hosts promocionais e IPs, conforme ilustra a Figura 13.

Na Figura 13 encontram-se os dois atributos de *relation*: um para armazenar strings de IP e outro para strings de host. Acompanhando suas variáveis internas, existe o método *addRelation*, que recebe um registro de recursos proveniente de uma resposta DNS, criado com o intuito de catalogar as relações caso contenham endereços coincidentes com alguma entrada da lista de Black (2024).

Após finalizar a estrutura da classe destinada a armazenar as respostas relevantes (Figura 14), foram desenvolvidas diretrizes para selecionar todos os retornos DNS no arquivo PCAP, a fim de passá-los para a função de adição apropriada.

Figura 13 – Classe para relacionar hosts e IPs

```
    class Relations:
        def __init__(self) -> None:
            self.hosts  : List[str] = []
            self.ips    : List[str] = []

        def addRelation(self, rr) -> None:
            question_index = bisectIsIn(self.hosts, rr.name)
            if question_index[0] == False:
                if bisectIsIn(sorted_flagged_domains, rr.name)[0] == False:
                    return
                else:
                    self.hosts.insert(question_index[1], rr.name)

            if rr.type == dpkt.dns.DNS_A:
                ip_str = inet_to_str(rr.ip)
                ans_index = bisectIsIn(self.ips,ip_str)
                if ans_index[0] == False: self.ips.insert(ans_index[1], ip_str)

            elif rr.type == dpkt.dns.DNS_AAAA:
                ip_str = inet_to_str(rr.ip6)
                ans_index = bisectIsIn(self.ips,ip_str)
                if ans_index[0] == False: self.ips.insert(ans_index[1], ip_str)

            elif rr.type == dpkt.dns.DNS_CNAME:
                ans_index = bisectIsIn(self.hosts, rr.cname)
                if ans_index[0] == False: self.hosts.insert(ans_index[1], rr.cname)
```

Fonte: Elaborado pelo autor (2024).

Figura 14 – Filtragem e interpretação dos pacotes DNS

```
    for ts, buf in pcap:
        eth = dpkt.ethernet.Ethernet(buf)
        if (not isinstance(eth.data, dpkt.ip.IP) and not
        isinstance(eth.data,dpkt.ip6.IP6)): continue
        ip = eth.data
        if not isinstance(ip.data, dpkt.udp.UDP): continue
        udp = ip.data
        if udp.sport != 53: continue
        try:
            dns = dpkt.dns.DNS(udp.data)
        except (dpkt.dpkt.NeedData, dpkt.dpkt.UnpackError):
            continue
        for rr in dns.an: relations_obj.addRelation(rr=rr)
```

Fonte: Elaborado pelo autor (2024).

Primeiramente, o pacote em iteração é averiguado para que tenha o cabeçalho de IP, pertença ao protocolo UDP e tenha porta de origem 53, o caracterizando uma resposta DNS. Cada resposta pode conter um endereço IPv4, IPv6 ou nome canônico. Ao constatar se a pergunta já foi inserida na lista de relações com a função *hostAtIndex*, o programa prossegue com a inserção dos IP e nomes canônicos em seu

devido objeto da lista. Uma vez preenchida a lista de relações, cada entrada tem seus itens do atributo hosts comparado com a lista de hosts conhecidos. Os IPs atrelados que consigam ser correlacionados serão adicionados à lista de IPs sinalizados para a análise final.

Enfim, a captura é uma segunda vez percorrida (Figura 15), utilizando os IPs sinalizados e intervalos publicitários agregados previamente.

Figura 15 – Determinação do consumo publicitário

```
    for ts, buf in pcap:
        eth = dpkt.ethernet.Ethernet(buf)

        if (isinstance(eth.data, dpkt.ip.IP)):
            ip = eth.data
            total_usage += ip.len
            if bisectIsIn(sorted_flagged_ips, inet_to_str(ip.src))[0] == True or
            isInAdInteval(time_stamp=ts): total_ad_usage += ip.len

        elif (isinstance(eth.data, dpkt.ip6.IP6)):
            ip = eth.data
            total_usage += ip.plen + 40
            if bisectIsIn(sorted_flagged_ips, inet_to_str(ip.src))[0] == True or
            isInAdInteval(time_stamp=ts): total_ad_usage += ip.plen + 40
```

Fonte: Elaborado pelo autor (2024).

O código da Figura 15 funciona de maneira em que todos os pacotes com cabeçalho IP serão acessados. No caso de um pacote ter o IP de origem pertencente a um endereço sinalizado como publicitário, ou tenha sido extraído durante o período de exibição de propaganda de algum vídeo do YouTube, terá seu tamanho adicionado à variável de consumo publicitário, objetivo deste trabalho.

3.3 INTELIGÊNCIA ARTIFICIAL

Como meio de enfrentar a problemática dos pacotes que teriam suas pesquisas DNS criptografadas sob o protocolo QUIC, foi proposto o treinamento de um modelo de inteligência artificial com os dados da captura realizada no computador pessoal. Para seu concebimento, foi necessário a preparação dos dados da captura para criar um quadro de dados com atributos pertinentes ao treinamento supervisionado (Tabela 1).

A Tabela 1 apresenta uma amostra do arquivo Comma Separated Values (CSV) gerado a partir de um exemplo de PCAP capturado pelo Wireshark. Nesta tabela, são fornecidos dados numéricos, como endereço de origem, destino e tamanho do quadro em bytes, enquanto informações como protocolo, versão IP, portas de origem e destino, e número de conexão QUIC são tratadas como atributos categóricos.

Tabela 1 – Exemplo de entrada do arquivo CSV utilizado no treinamento

Campo	Valor	Variável
Endereço de origem	2800:3f0:4001:831::2003	Source1 - Source8
Endereço destinatário	2804:14c:5fe6:88c0:496a:1368:d213:8415	Destination1 - Destination8
Protocolo	QUIC	Protocol
Tamanho do quadro (bytes)	1010	Length
Versão do IP	6	IPv
Porta de origem	443	SourcePort
Porta destinatária	59749	DestinationPort
Número de conexão QUIC	2	QuicConnectionNumber
Presença de anúncios pré-roll	Não	Ad

Fonte: Elaborado pelo autor (2024).

A amostra de treinamento contida no CSV foi posteriormente processada pelo método de treinamento do PyCaret (Figura 16).

Figura 16 – Código de treinamento da IA

```
    pycaret_setup = setup(data=data_frame, target='Ad',
    categorical_features=["Protocol", "IPv"],
    numeric_features=["Source1", "Source2", "Source3", "Source4",
                      "Source5", "Source6", "Source7", "Source8",
                      "Destination1","Destination2","Destination3","Destination4",
                      "Destination5","Destination6","Destination7","Destination8",
                      "Length", "SourcePort", "DestinationPort",
                      "QuicConnectionNumber"],
    session_id=123,
    log_experiment=True)
```

Fonte: Elaborado pelo autor (2024).

Esse método é responsável por preparar os dados, aplicando técnicas como one hot encoding, e treinar várias arquiteturas usando os dados fornecidos para prever se o conteúdo relacionado ao input é um vídeo de propaganda do YouTube ou não. As configurações adotadas pela biblioteca envolvem a separação de 70% dos dados fornecidos para treinamento, a designação explícita dos dados a serem aplicados ao tratamento categórico, definição de dez sub sets para validação cruzada e a seleção dos 20% dos features mais expressivos do modelo durante sua consolidação. São passados como parâmetros o dataframe com os dados de treinamento, a coluna a ser predita, os inputs de característica categórica e os inputs numéricos, como os segmentos que compõem o endereço IPV6 de origem representados pelas colunas *source*.

4 RESULTADOS

Após detalhar os procedimentos adotados neste trabalho, este capítulo apresentará os resultados obtidos por meio das abordagens algorítmicas e de inteligência artificial implementadas.

4.1 CLASSIFICAÇÃO PUBLICITÁRIA POR ALGORITMO

Após 24 horas de execução da rotina de visitas, foram gerados quatro arquivos de capturas controladas, além dos três arquivos PCAP obtidos de um servidor ISP anônimo (Tabela 2).

Tabela 2 – Arquivos de captura originados

Data	Tamanho	Espera em vídeo	Espera em website	Propaganda
		Dataset gerado		
19/05/24 16:27 19/05/24 22:27	0,83 GB	15 min	05 min	3,81%
19/05/24 10:22 19/05/24 16:22	0,70 GB	15 min	05 min	17,95%
19/05/24 02:04 19/05/24 08:04	0,45 GB	15 min	05 min	7,03%
01/06/24 23:28 02/06/24 23:28	2,19 GB	03 min	02 min	49,08%
		Dataset Kaggle		
06/05/23 09:29 06/05/23 09:30	0,78 GB	-	-	3,31%
06/05/23 09:23 06/05/23 09:24	0,78 GB	-	-	0,20%
06/05/23 09:20 06/05/23 09:21	0,08 GB	-	-	0,26%

Fonte: Elaborado pelo autor (2024).

As quatro primeiras entradas foram geradas a partir de capturas de tráfego simuladas, enquanto as demais foram obtidas a partir do conjunto de dados de Yar (2023) no qual as configurações de espera não se aplicavam, sendo representadas como vazias na tabela. A última captura simulada teve os tempos de espera reduzidos para tornar a simulação mais dinâmica, uma vez que não havia um usuário real interagindo com a página.

A base de dados composta pelas três primeiras capturas geradas apresentou um tamanho médio de 0,66 GB, com um desvio padrão de 0,19 GB, e uma média de

9,60% de conteúdo publicitário, com um desvio padrão de 7,41%. A quarta captura utilizou configurações de acesso mais dinâmicas, registrando um tamanho de 2,19 GB, dos quais 49,08% eram compostos por publicidade. Finalmente, os três arquivos provenientes do ISP apresentaram uma média de 1,26% de conteúdo publicitário, com um desvio padrão de 1,78%.

Para distinguir as origens do conteúdo sinalizado, o algoritmo de classificação fundamentado nas requisições DNS foi aplicado separadamente aos registros de rede listados anteriormente, de modo a revelar os consumos publicitários de IPs consultados por meio de protocolos não criptografados (Tabela 3).

Tabela 3 – Propaganda identificada por DNS.

Captura	Número de hosts	Número de IPs	Consumo relativo ao total publicitário
Dataset gerado			
19/05/24 16:27 19/05/24 22:27	247	665	85,70%
19/05/24 10:22 19/05/24 16:22	181	476	93,34%
19/05/24 02:04 19/05/24 08:04	47	161	75,23%
01/06/24 23:28 02/06/24 23:28	311	996	98,44%
Dataset Kaggle			
06/05/23 09:29 06/05/23 09:30	38	104	100%
06/05/23 09:23 06/05/23 09:24	36	57	100%
06/05/23 09:20 06/05/23 09:20	11	25	100%

Fonte: Elaborado pelo autor (2024).

A Tabela 3 sintetiza os resultados obtidos pela abordagem que utiliza o serviço de resolução de nomes, exibindo o número de anunciantes, os IPs relacionados a esses hosts e a porcentagem relativa entre o total de bytes de propaganda originados da análise de DNS em relação ao total de propaganda.

Os três primeiros conjuntos de dados gerados resultaram em médias de 158,33 hosts, 434,00 IPs e 84,76% de consumo relativo ao total de publicidade, com desvios padrões de 101,91, 254,61 e 9,09%, respectivamente. Em contraste, o conjunto com configurações distintas de espera apresentou números maiores: 311 hosts publicitários identificados, 996 IPs relacionados a esses hosts e 98,44% do total de propaganda estabelecido pela análise DNS. Quanto ao dataset do Kaggle, após filtragem

baseada no estudo das respostas de tradução de domínios, foram observadas médias de 28,33 hosts e 62,00 IPs, com desvios de 15,04 e 39,74%, respectivamente.

Para mitigar a questão da inacessibilidade às consultas DNS sobre o protocolo cifrado QUIC, desenvolveu-se na metodologia um método de categorização por instante de transmissão. Os dados numéricos resultantes desse tratamento estão disponíveis na Tabela 4.

Tabela 4 – Propaganda devido vídeos.

Captura	Número de intervalos	Consumo relativo ao total publicitário
19/05/24 16:27 19/05/24 22:27	3	15,09%
19/05/24 10:22 19/05/24 16:22	5	8,14%
19/05/24 02:04 19/05/24 08:04	6	25,70%
01/06/24 23:28 02/06/24 23:28	19	1,88%

Fonte: Elaborado pelo autor (2024).

A Tabela 4 apresenta o número de vezes que vídeos de marketing foram identificados ao longo da captura, juntamente com a fração total de bytes correspondentes em relação ao total de propaganda.

O processamento por meio da comparação de timestamps permitiu identificar, nas três primeiras gravações de tráfego, médias de 4,67 intervalos comerciais e 16,31% do marketing relacionado a esse tipo de mídia. Os valores de desvio padrão para essas estatísticas foram de 1,53 intervalos e 8,84%. Na última captura, entretanto, observou-se um aumento no número de intervalos publicitários em relação à média anterior, totalizando 19 intervalos, e uma menor porcentagem em relação ao total, possivelmente devido à presença menos frequente de filmes publicitários e a uma maior proporção de acessos aos demais portais descritos no SimilarWeb.

Ao analisar simultaneamente a última coluna das Tabelas 3 e 4, nota-se que a soma ultrapassa 100%, sugerindo a presença de propagandas provenientes de IPs consultados em aberto. Essa observação implica a reproduções de vídeos do YouTube fora do protocolo QUIC em algumas instâncias, o que pode acarretar em resultados significativos apenas com a filtragem por DNS.

Foi constatado que a média do tráfego publicitário das quatro capturas controladas foi de 19,97%, em comparação com apenas 1,26% nas capturas disponíveis na plataforma Kaggle. Essa disparidade pode ser atribuída ao fato de que as capturas controladas são simulações do tráfego real, focando em visitas às páginas mais popu-

lares globalmente, enquanto as capturas do ISP são menos abrangentes e originam-se de usuários que possivelmente utilizam bloqueadores de anúncios ativos.

A média das capturas ativas se aproximou da cota de 18,2% identificada por Zietz e Redel (2021), contudo, a alta variância de 426,35 nessas novas capturas também evidenciou a dispersão dos resultados, devido à aleatoriedade dos conjuntos produzidos. Considerando o resultado da captura ativa com menos anúncios e assumindo uma relação linear aproximada, em 30 dias seriam transferidos 3,80 GB de publicidade. Por outro lado, aplicando a mesma suposição ao conjunto com mais propagandas de 19/05/24, seriam baixados 15,09 GB.

4.2 CLASSIFICAÇÃO PUBLICITÁRIA POR IA

Com o intuito de poder classificar capturas não controladas, foi proposta a elaboração de uma IA. Após executar o método de treinamento de PyCaret, foram retornados os dez modelos de melhor performance (Tabela 5).

Tabela 5 – Métricas dos modelos gerados

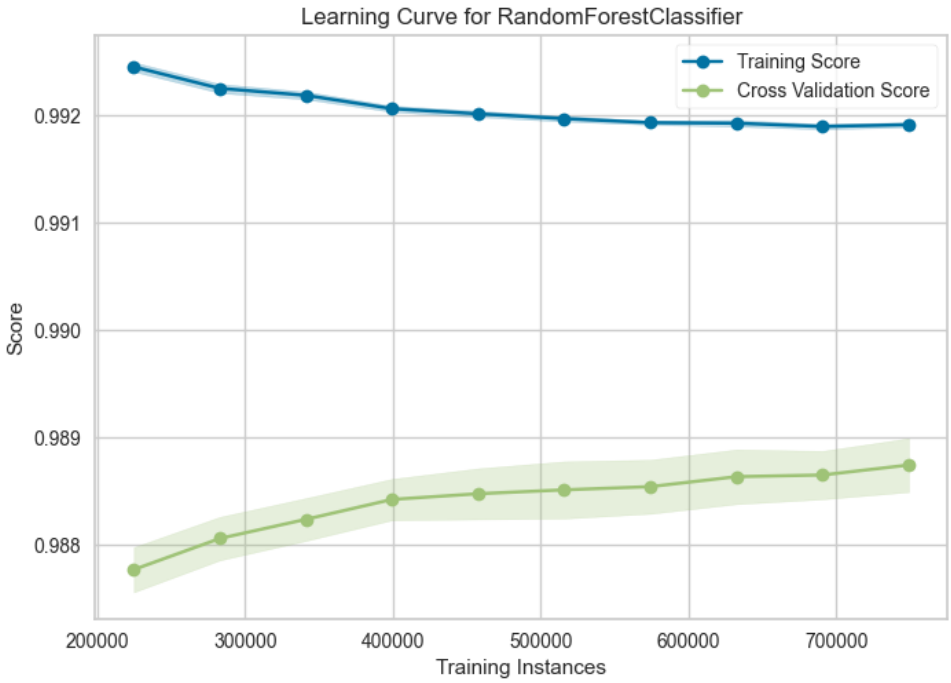
Model	**Accuracy**	**AUC**	**Recall**	**Precision**
Random Forest Classifier	0.9887	0.9794	0.9887	0.9879
Extra Trees Classifier	0.9887	0.9673	0.9887	0.9878
Decision Tree Classifier	0.9886	0.9502	0.9886	0.9878
K Neighbors Classifier	0.9871	0.9274	0.9871	0.9858
Gradient Boosting Classifier	0.9854	0.9734	0.9854	0.9835
Ada Boost Classifier	0.9796	0.9588	0.9796	0.9724
Logistic Regression	0.9792	0.6003	0.9792	0.9588
Ridge Classifier	0.9792	0.7267	0.9792	0.9588
Linear Discriminant Analysis	0.9792	0.7267	0.9792	0.9588
Dummy Classifier	0.9792	0.5000	0.9792	0.9588
SVM - Linear Kernel	0.8955	0.5942	0.8955	0.9596
Quadratic Discriminant Analysis	0.4068	0.5783	0.4068	0.9682
Naive Bayes	0.1214	0.7035	0.1214	0.9791

Fonte: Elaborado pelo autor (2024).

O treinamento levou cerca de 40 minutos em uma máquina com processador Intel i7-12700H e 36 GB de memória, sem a utilização de recursos como placa gráfica dedicada. Revelou-se que o classificador por floresta aleatória se destacou em todas as quatro métricas em comparação com os outros modelos avaliados. A floresta aleatória é uma técnica que combina várias árvores de decisão, buscando reduzir a variância ao diversificar esse conjunto. Em vez de variar os exemplos de treinamento a cada ponto de divisão, uma amostra aleatória de atributos é selecionada (Norvig; Russell, 2014).

Para explorar as características do modelo concebido, foram gerados gráficos avaliativos. Inicialmente, apresenta-se o gráfico de treinamento, ilustrado na Figura 17.

Figura 17 – Curva de aprendizado do modelo



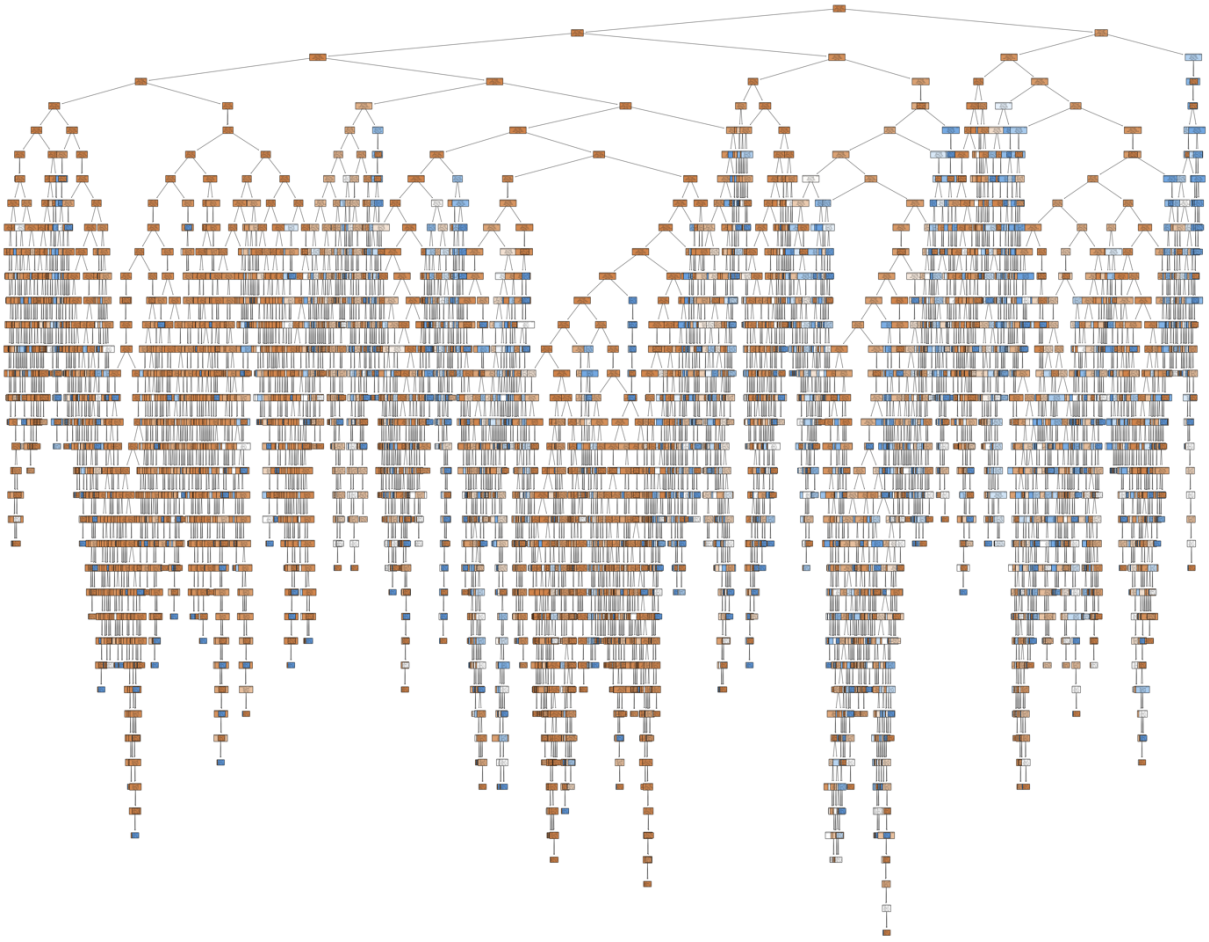
Fonte: Elaborado pelo autor (2024).

No gráfico de aprendizado, é evidente que foram necessárias mais de 700000 instâncias de treinamento para alcançar pontuações de acurácia superiores a 0,991, porém, também se nota na curva de validação cruzada uma performance abaixo de 0,989. Tal observação sugere uma diminuição de desempenho do framework em conjuntos de dados não utilizados durante a etapa de treinamento.

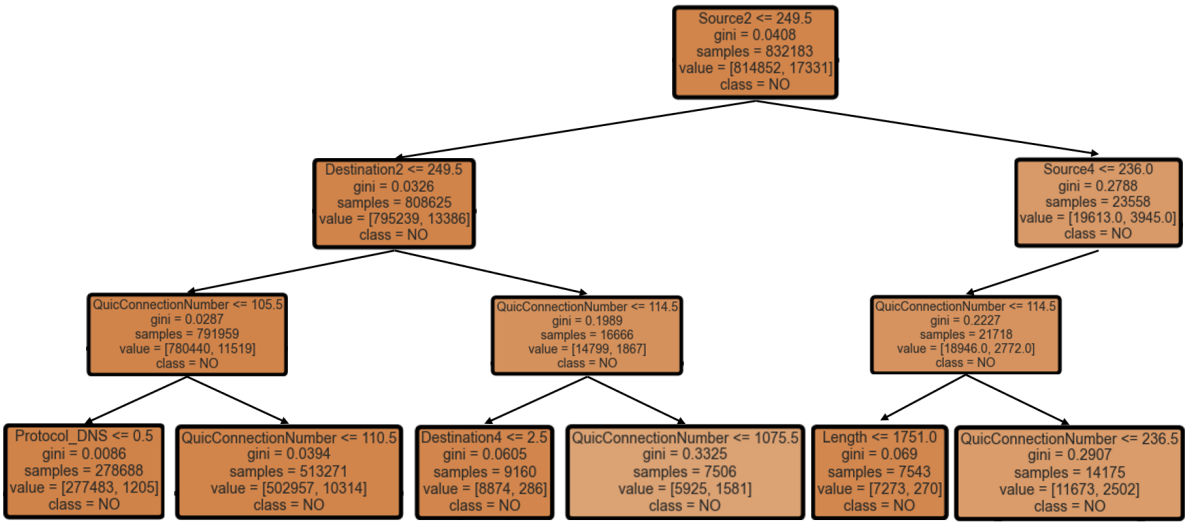
Para compreender melhor o processo de classificação adotado pelo modelo de melhor desempenho, foi gerada uma árvore utilizando a arquitetura Decision Tree (Figura 18), que apresenta um comportamento semelhante ao do modelo Random Forest.

Na Figura 18b, o nó raiz direciona o fluxo com base na avaliação do segundo segmento do endereço IP de origem, encaminhando-o para a esquerda se o identificador de rede for 249,5 ou menor, e para a direita caso contrário. Em seguida, são considerados outros atributos como o endereço destinatário e o tipo de protocolo, alcançando uma profundidade máxima de 39 níveis (Figura 18a). A floresta aleatória elaborada, composta por várias árvores que tomam decisões por consenso da maioria, apresentou uma profundidade máxima de 53 níveis em uma única árvore, em contraste com o número anterior.

Figura 18 – Árvore do modelo Decision Tree



(a) Visão geral

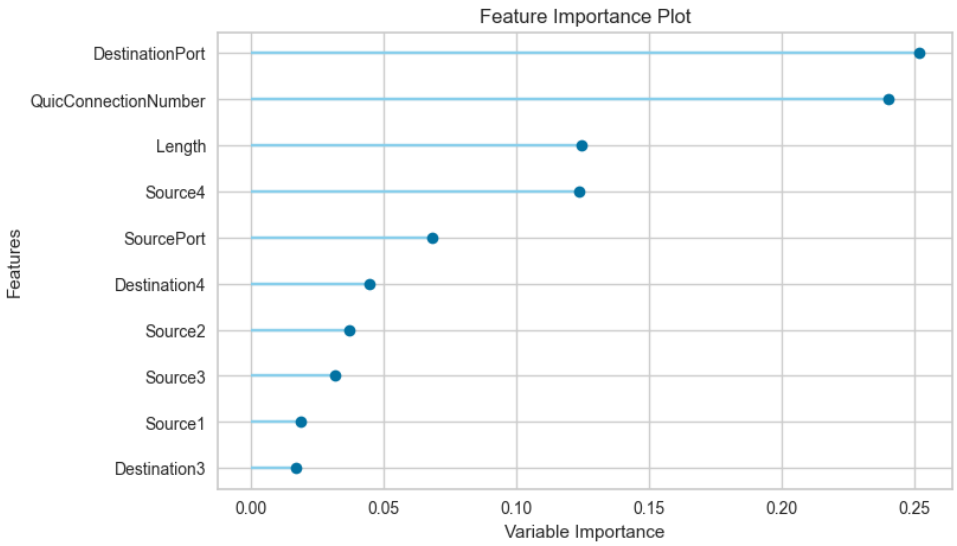


(b) Visão em detalhe

Fonte: Elaborado pelo autor (2024).

No processo de treinamento, foram determinados os inputs de maior importância para a decisão binária, conforme exposto na Figura 19.

Figura 19 – Features mais expressivos



Fonte: Elaborado pelo autor (2024).

As características mais influentes na categorização foram a porta de destino, o número da conexão QUIC, o tamanho do quadro e o quarto componente do endereço IP. Com essas informações, seria razoável inferir o processo de tomada de decisão com base na exibição dessas características do pacote.

Figura 20 – Pacotes publicitários filtrados

EpochArrivalTime	Source	Destination	Length	SourcePort	DestinationPort	QuicConnectionNumber
1716101369.580422000	142.251.129.162	192.168.37.128	1399	443	34931	135
1716101369.609432000	142.251.129.162	192.168.37.128	1399	443	34931	135
1716101369.609436000	142.251.129.162	192.168.37.128	657	443	34931	135
1716101369.609437000	142.251.129.162	192.168.37.128	69	443	34931	135
1716101369.641998000	142.251.129.162	192.168.37.128	165	443	34931	135
1716101369.656385000	172.217.162.142	192.168.37.128	76	443	43850	106
1716101369.664464000	172.217.162.142	192.168.37.128	76	443	43850	106
1716101369.722624000	177.193.95.77	192.168.37.128	73	443	57906	132
1716101369.723296000	177.193.95.77	192.168.37.128	165	443	57906	132
1716101369.726881000	177.193.95.77	192.168.37.128	168	443	57906	132

(a) Pacotes identificados como publicitários

EpochArrivalTime	Source	Destination	Length	SourcePort	DestinationPort	QuicConnectionNumber
1716101376.384604000	172.217.30.86	192.168.37.128	1399	443	44844	129
1716101376.384605000	172.217.30.86	192.168.37.128	1399	443	44844	129
1716101376.391142000	172.217.30.86	192.168.37.128	1399	443	44844	129
1716101376.391144000	172.217.30.86	192.168.37.128	1399	443	44844	129
1716101376.391145000	172.217.30.86	192.168.37.128	1399	443	44844	129
1716101376.391146000	172.217.30.86	192.168.37.128	1399	443	44844	129
1716101376.391146000	172.217.30.33	192.168.37.128	1395	443	33380	136
1716101376.391147000	172.217.30.33	192.168.37.128	1399	443	33380	136
1716101376.391147000	172.217.30.33	192.168.37.128	1399	443	33380	136
1716101376.391148000	172.217.30.33	192.168.37.128	1399	443	33380	136

(b) Pacotes identificados como não publicitários

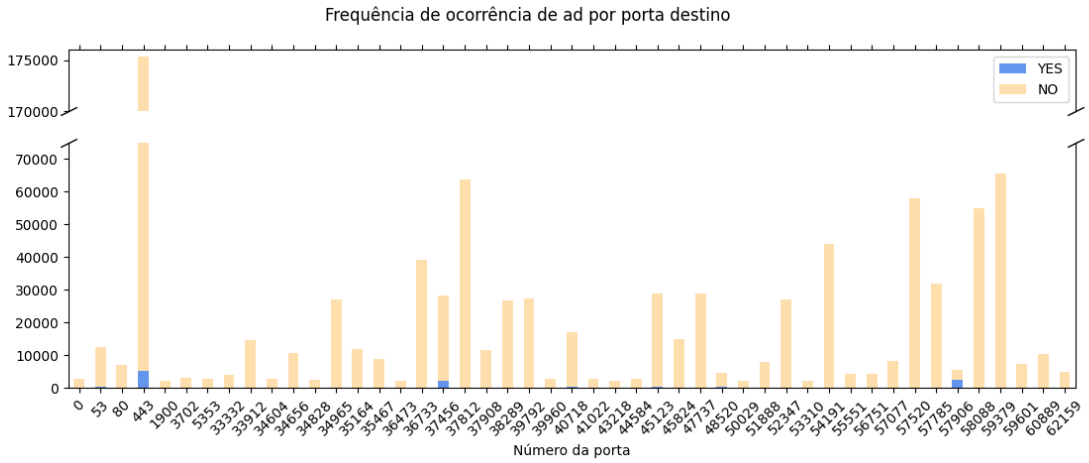
Fonte: Elaborado pelo autor (2024).

A Figura 20 ilustra uma captura de dados realizada em 19/05/2024, onde foi observada a exibição de anúncios antes do vídeo requisitado no YouTube. Utilizando

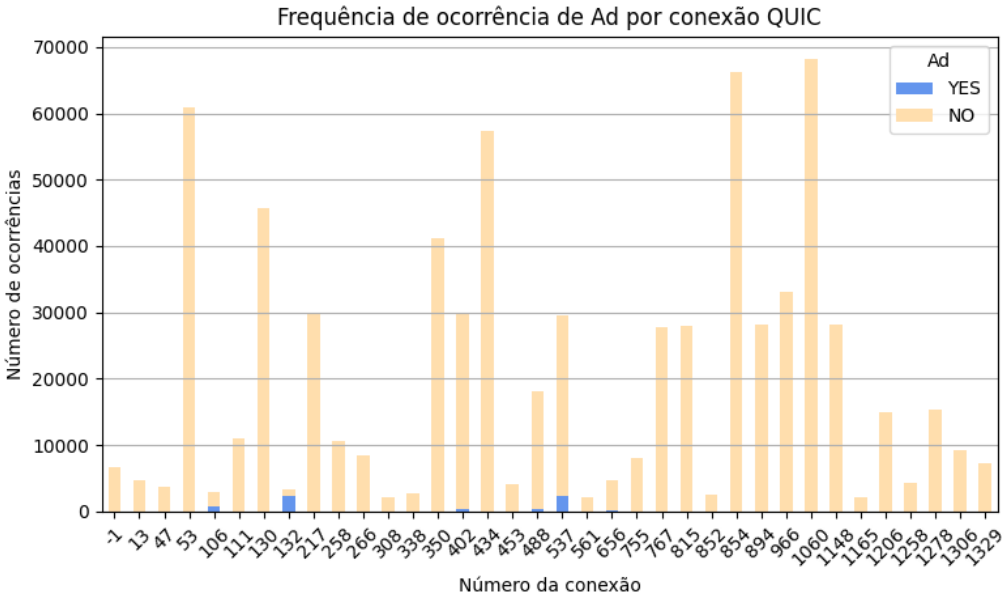
um filtro para o protocolo QUIC e configurando a janela para exibir os recursos mais relevantes, não foi possível identificar uma semelhança explícita entre os conteúdos dos campos destacados nos pacotes, como a repetição de valores em determinados campos.

Como uma abordagem alternativa, foi gerada a Figura 21 para estabelecer uma relação entre as categorias mais importantes e sua influência no processo de decisão.

Figura 21 – Gráficos de ocorrências de Ad por features mais relevantes



(a) Ocorrências por porta destino



(b) Ocorrências por número de conexão QUIC

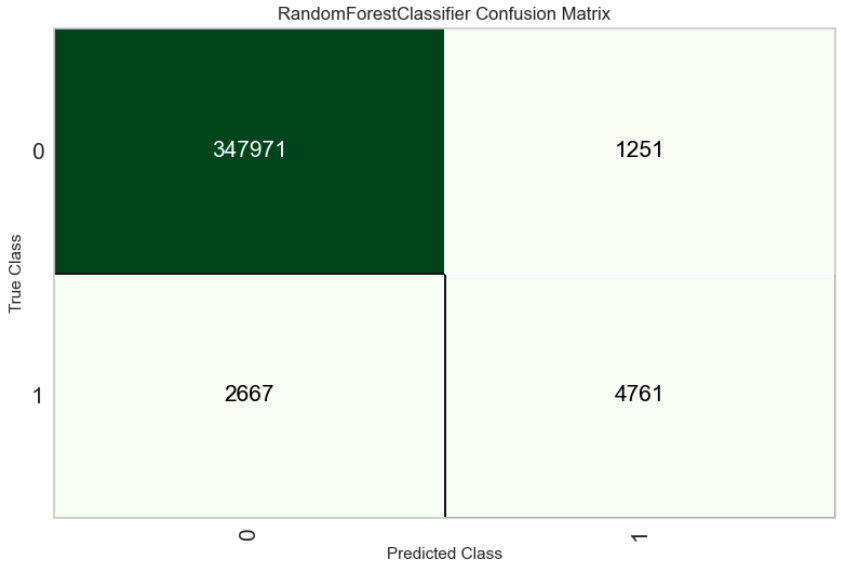
Fonte: Elaborado pelo autor (2024).

Os gráficos gerados visualizam a proporção de pacotes publicitários e não publicitários por porta de destino e número de conexão QUIC que tiveram mais de dois

mil pacotes transmitidos. Vale ressaltar que os casos em que o número de conexão QUIC não foi identificado tiveram esse campo preenchido com menos um, um número de porta inválido. Apesar de a porta 443 e a conexão 537 terem registrado os maiores números de pacotes publicitários pré-vídeos, com 5175 e 2347, respectivamente, esses mesmos parâmetros também identificaram 170191 e 27210 pacotes classificados como não comerciais, levantando questões sobre a eficácia da IA em correlacionar esses dados de forma adequada.

Com objetivo de proporcionar melhor visualização das classificações, os resultados são representados na forma de uma matriz de confusão, apresentada na Figura 22.

Figura 22 – Matriz de confusão



Fonte: Elaborado pelo autor (2024).

Com essa representação, é possível distinguir as classificações corretas dos positivos e negativos, além das classificações incorretas de ambos os grupos. A diagonal principal da matriz representa as previsões corretas, enquanto a diagonal secundária representa as previsões incorretas.

As métricas do PyCaret foram calculadas usando um mapeamento de *1* para *YES* e *0* para *NO*, com base nas médias das validações realizadas. Ao calculá-las manualmente através através da matriz de confusão e definindo as previsões positivas de anúncios como 1, as estatísticas resultantes são: acurácia de 98,90%, precisão de 79,19% e recall de 64,10%.

Embora as predições verdadeiras constituam uma grande proporção do total de avaliações (98,87% conforme a médias das validações cruzadas), o número absoluto de classificações errôneas permanece elevado. Com base nessa constatação, conclui-se que o modelo carece de validação.

O melhor modelo desenvolvido foi aplicado nos conjuntos de dados do Kaggle, que não possuíam timestamps para identificar a origem publicitária das possíveis visitas ao YouTube pelos computadores conectados ao ISP. As análises não detectaram pacotes publicitários provenientes de pré-rolls do YouTube durante os três minutos de captura, e embora esse seja um intervalo curto, a ausência desses pacotes aumenta ainda mais as dúvidas sobre a eficácia da IA. Esse comportamento indica um possível overfitting dos dados no treinamento.

Para fins de reprodução, os parâmetros internos da floresta são apresentados na Tabela 6.

Tabela 6 – Hiperparâmetros do Modelo de Random Forest

Parâmetro	**Valor**
`bootstrap`	True
`ccp_alpha`	0.0
`class_weight`	None
`criterion`	'gini'
`max_depth`	None
`max_features`	'sqrt'
`max_leaf_nodes`	None
`max_samples`	None
`min_impurity_decrease`	0.0
`min_samples_leaf`	1
`min_samples_split`	2
`min_weight_fraction_leaf`	0.0
`monotonic_cst`	None
`n_estimators`	100
`n_jobs`	-1
`oob_score`	False
`random_state`	123
`verbose`	0
`warm_start`	False

Fonte: Elaborado pelo autor (2024).

A Tabela 6 apresenta os hiperparâmetros retornados pelo método *get_params* do modelo de Random Forest criado, especificando, por exemplo, 100 estimadores a serem gerados e comparados, sem um limite pré-estabelecido para a profundidade das árvores. Além disso, foi aplicado o critério de Gini para a avaliação dos nós de decisão, que quantifica a probabilidade de um item ser classificado incorretamente se for escolhido aleatoriamente.

4.3 LIMITAÇÕES

O trabalho, por mais que tenha encontrado os percentuais de marketing almejados, encontrou limitações em seu desenvolver, as quais serão discutidas a seguir.

Os percentuais publicitários determinados pela análise de timestamps e DNS apresentaram uma alta variabilidade, o que diminuiu sua significância. Essa dispersão foi influenciada pelo tamanho e número dos datasets utilizados, sugerindo que uma amostra de dados mais ampla poderia proporcionar uma representação mais precisa.

Para criar os datasets controlados, a navegação de um usuário foi simulada utilizando um algoritmo de visitas. Por mais que tenham sido considerados os sites mais visitados e seus respectivos tempos médios de permanência, a interação com a página foi mais limitada em comparação com a interação de um usuário real.

Durante a validação da IA, foram identificados bons percentuais nas métricas avaliativas mais comuns; no entanto, o número absoluto de erros e o risco de overfitting ainda foram observados. Novamente, um conjunto de dados mais amplo seria necessário para dar maior convicção as classificações realizadas.

5 CONCLUSÃO

A internet, frequentemente descrita como uma rede de redes, teve seu acesso amplamente expandido nos últimos anos, facilitando o acesso gratuito a uma vasta gama de conteúdos online. No entanto, como contrapartida a essa liberdade de acesso, surgiu o fenômeno das propagandas online, que podem resultar em um consumo indesejado de banda.

Com a elaboração de um algoritmo automatizado para navegação online e filtragem baseada em requisições DNS capturadas e nos momentos de transmissão dos pacotes, foi possível determinar o percentual de conteúdo publicitário presente nos conjuntos de dados adquiridos. Para ampliar a aplicação a gravações de tráfego não controlado, propôs-se também o desenvolvimento de um modelo de inteligência artificial para classificação.

Após fundamentar os conceitos de redes, inteligência artificial e propaganda, a metodologia detalhou os códigos desenvolvidos para as aplicações. Esses algoritmos criaram e analisaram bases de dados, resultando em uma média de 9,60% de conteúdo publicitário nas capturas controladas baseadas nos valores de permanência do SimilarWeb, enquanto os arquivos de monitoramento do ISP apresentaram um percentual de 1,26%.

A inteligência artificial foi treinada utilizando dados capturados durante a navegação em sites populares. O melhor modelo, utilizando a arquitetura de random forest, alcançou uma acurácia de 98% na validação dos pacotes. Apesar dessa boa métrica relativa, o modelo classificou erroneamente 3918 pacotes no conjunto de teste e não identificou pacotes comerciais nas capturas do ISP, sugerindo a possibilidade de overfitting.

Retomando os objetivos listados no início deste trabalho, foi estabelecido um programa de navegação na web que simula o comportamento de um usuário comum e elaborado um algoritmo de filtragem baseado nas respostas DNS. O objetivo de criar uma IA classificatória foi parcialmente alcançado, pois ela ainda necessita de validação, mas ainda foi possível avaliar o impacto do tráfego publicitário nesta pesquisa.

Para alcançar estatísticas mais precisas, considera-se em futuros estudos aumentar a precisão dos resultados com aquisições adicionais de dados controlados. Estratégias para tornar a simulação mais próxima do comportamento humano incluem manipulação da scroll bar e login em contas em diversos sites, além do uso de payloads criptografados por meio de técnicas avançadas de processamento de linguagem, como aquelas implementadas pela biblioteca Spacy em Python.

REFERÊNCIAS

AKBARI, I. *et al.* A look behind the curtain: Traffic classification in an increasingly encrypted web. **Proceedings of the ACM on Measurement and Analysis of Computing Systems**, v. 5, p. 1–26, 02 2021.

ALI, M. Pycaret: An open source, low-code machine learning library in python. 2020. Disponível em: https://www.pycaret.org. Acesso em: 5 maio 2024.

ALMUHAMMADI, S.; ALNAJIM, A.; AYUB, M. Quic network traffic classification using ensemble machine learning techniques. **Applied sciences**, v. 13, n. 8, p. 17, 2023.

BLACK, S. Unified hosts file with base extensions. **GitHub**, 2024. Disponível em: https://github.com/StevenBlack/hosts. Acesso em: 5 maio 2024.

CECCON, D. **Os tipos de redes neurais**. 2020. Disponível em: https://iaexpert.academy/2020/06/08/os-tipos-de-redes-neurais/?doing_wp_cron=1700143846.045502901077270507812 5#comments. Acesso em: 15 out. 2023.

CHAFFEY, D.; ELLIS-CHADWICK, F. **Digital marketing**: strategy, implementation and practice. 6. ed. Upper Saddle River: Pearson Education, 2016.

CHAPPELL, L. A. **Wireshark network analysis**: the official wireshark certified network analyst study guide. 2. ed. San Jose: Protocol Analysis Institute, Chappell University, 2012.

GAZVODA, U. **Free, open-source ad content blocker**. 2023. Disponível em: https://ublockorigin.com/. Acesso em: 15 out. 2023.

GRIGUTYTė, M. What is ad blocking and how does it work. **NordVPN**, 2023. Disponível em: https://nordvpn.com/pt-br/blog/what-is-ad-blocking/. Acesso em: 15 out. 2023.

IYENGAR, J.; THOMSON, M. **QUIC**: A UDP-Based multiplexed and secure transport. RFC Editor, 2021. RFC 9000. (Request for Comments, 9000). Disponível em: https://www.rfc-editor.org/info/rfc9000. Acesso em: 27 out. 2023.

KUROSE, J.; ROSS, K. **Redes de computadores e a Internet**: uma abordagem top-down. 8. ed. Porto Alegre: Bookman, 2021.

MACHINE Learning Crash Course. 2024. Disponível em: https://developers.google.com/machine-learning/crash-course. Acesso em: 5 maio 2024.

MARSHALL, P.; RHODES, M.; TODD, B. **Ultimate guide to Google Ads**. 6. ed. Irvine: Entrepreneur Press, 2020. (Ultimate Series).

MOCKAPETRIS, P. **Domain names**: implementation and specification. RFC Editor, 1987. RFC 1035. (Request for Comments, 1035). Disponível em: https://www.rfc-editor.org/info/rfc1035. Acesso em: 27 out. 2023.

MOLINA, D.; HARSHA, S.; FORTNER, T. The selenium browser automation project. **GitHub**, 2024. Disponível em: https://www.selenium.dev/documentation/. Acesso em: 5 maio 2024.

NERY, C.; BRITTO, V. Internet já é acessível em 90,0% dos domicílios do país em 2021. **IBGE**, 2022. Disponível em: https://agenciadenoticias.ibge.gov.br/agencia-noticias/2012-agencia-de-noticias/noticias/34954-internet-ja-e-acessivel-em-90-0-dos-domicilios-do-pais-em-2021#:~:text=Internet%20chega%20a%2090%2C0,%25%20para%2092%2C3%25. Acesso em: 15 out. 2023.

NORVIG, P.; RUSSELL, S. **Inteligência artificial**. 3. ed. Rio de Janeiro: Elsevier Brasil, 2014.

POLLARD, B. **HTTP/2 in action**. New York, NY: Manning, 2019.

SIKOS, L. F. Packet analysis for network forensics: a comprehensive survey. **Forensic Science International**, Elsevier Ltd, v. 32, 2020.

SONG, D.; ALPEROVICH, T.; BELLAMY, N. Unified hosts file with base extensions. **GitHub**, 2023. Disponível em: https://github.com/StevenBlack/hosts. Acesso em: 15 out. 2023.

SPANGLER, T. Alphabet misses q4 earnings estimates, youtube ad revenue drops by nearly 8%. **Variety**, 2023. Disponível em: https://variety.com/2023/digital/news/alphabet-google-q4-2022-earnings-youtube-revenue-falls-1235510514/. Acesso em: 15 out. 2023.

TANENBAUM, A. S.; BOS, H. **Sistemas operacionais modernos**. 4. ed. São Paulo: Pearson, 2016.

TANENBAUM, A. S.; WETHERALL, D. **Computer networks**. 5. ed. Boston: Prentice Hall, 2011.

TOMKEVIčIūTė, A. How do ad blockers work? all you need to know - cybernews. **Cybernews**, 2023. Disponível em: https://cybernews.com/best-ad-blockers/how-ad-blocking-works/. Acesso em: 15 out. 2023.

TONG, V. *et al.* A novel quic traffic classifier based on convolutional neural networks. *In*: **Proceedings** of the IEEE GLOBAL COMMUNICATIONS CONFERENCE (GLOBECOM), 09-13 dez. 2018, Abu Dhabi, United Arab Emirates. 2018. p. 1–6. Disponível em: https://ieeexplore.ieee.org/document/8647128/figures#figures. Acesso em: 27 out. 2023.

YAR, M. A. Internet traffic data set: more than 2000 internet users real time traffic data with raw files. 2023. Disponível em: https://www.kaggle.com/datasets/asfandyar250/network. Acesso em: 5 maio 2024.

ZIETZ, D.; REDEL, R. A. **Análise do tráfego proveniente de publicidades em um ISP**. 2021. Trabalho de Conclusão de Curso (Graduação em Engenharia de Computação) — Centro Universitário Sociesc, UniSociesc, Joinville, 2021.