Ip-Tab-Learning: Neural Network-Oriented Firewall Learning System

Ip-Tab-Learning: Sistema de Aprendizagem de Firewall Orientado por Redes Neurais

Franciscarlos de Ávila Nascimento
Centro Universitário do Sul de Minas – UNISMG – Varginha – MG – Brasil.
Centro Federal de Educação Tecnológica de Minas Gerais – CEFETMG –
Nepomuceno – MG – Brasil.
+55 35 3861-4500
franciscarlospereira@gmail.com

Rodrigo Franklin Frogeri
Departamento de Pesquisa - Centro Universitário do Sul de Minas – UNISMG – Varginha – MG – Brasil.
Programa de Pós-Graduação em Sistemas de Informação e Gestão do Conhecimento - Universidade FUMEC – Belo Horizonte – MG – Brasil.
+55 35 3219-5045
rodrigo.frogeri@professor.unis.edu.br

Liz Áurea Prado
Departamento de Pesquisa - Centro Universitário do Sul de Minas – UNISMG – Varginha – MG – Brasil.
Centro Federal de Educação Tecnológica de Minas Gerais – CEFETMG – Varginha – MG – Brasil
+55 35 3219-5045
liz.prado@professor.unis.edu.br

Afonso de Paula Feliciano
Departamento de Pesquisa - Centro
Centro Universitário do Sul de Minas – UNISMG – Varginha – MG – Brasil.
+55 35 3219-5045
afonsofeliciano@live.com

ABSTRACT

This study presents a neural network oriented Iptables firewall learning system. Such an approach is justified by the current scenario of information security in organizations and by the demand for professionals trained in mitigating threats to computer infrastructures. The objective of the study was to describe and present a neural network-oriented Iptables firewall learning software that is able to assist students and professionals in Information Security to understand Iptables firewall rules applicable to attacks on computer networks of the brute-force and DoS type. This purpose was achieved through the use of the concepts of natural language processing, text mining, and the application of the naive bayes algorithm in a neural network. Methodologically, the research was characterized as Design Science (DS), exploratory, and bibliographical. The study demonstrated that it is possible to apply text mining techniques and machine learning algorithms in neural networks to develop a firewall learning system.It was observed that questions asked in the Portuguese language were converted into internal rules of the *Iptables* firewall, meeting the established objective. The software developed made it possible, through simple questions, for a student or professional in information security to have access to the *Iptables* firewall rules and their respective technical explanations. The study allows new functionalities to be implemented or adaptations to related issues, increasing the interdisciplinarity between the fields of artificial intelligence, information security, and teaching / learning process.

Keywords

Neural network; Iptables; Naive bayes algorithm; Natural language processing; Text mining.

RESUMO

Este estudo apresenta um sistema de aprendizagem de firewall *Iptables* orientado por uma rede neural. Tal abordagem se justifica mediante o atual cenário da segurança da informação nas organizações e pela demanda por profissionais capacitados na mitigação de ameaças às infraestruturas computacionais. O objetivo do estudo foi descrever e apresentar um software de aprendizagem de firewall *Iptables* orientado por redes neurais, capaz de auxiliar estudantes e profissionais em Segurança da Informação a compreender regras de firewall *Iptables* aplicáveis aos ataques a redes de computadores do tipo força bruta e negação de serviço. Este propósito foi conseguido por meio da utilização dos conceitos de processamento de linguagem natural, mineração de textos e da aplicação do algoritmo naive bayes em uma rede neural. Metodologicamente, a pesquisa caracterizou-se como Design Science (DS), exploratória e bibliográfica. O estudo demonstrou que é possível aplicar técnicas de mineração de textos e algoritmos de aprendizagem de máquina em redes neurais para desenvolver um sistema de aprendizagem de firewall. Observou-se que perguntas realizadas na língua portuguesa foram convertidas em regras internas do firewall *Iptables*, atendendo ao objetivo estabelecido. O software desenvolvido possibilitou que, mediante questionamentos simples, um estudante ou profissional

em segurança da informação tivesse acesso às regras de firewall *Iptables* e suas respectivas explicações técnicas. O estudo possibilita que novas funcionalidades sejam implementadas ou adequações a assuntos correlatos, ampliando a interdisciplinaridade entre os campos da inteligência artificial, segurança da informação e processo de ensino/aprendizado.

Palavras-chave

Rede neural. Iptables. Algoritmo naive bayes. Processamento de linguagem natural. Mineração de texto.

1. INTRODUÇÃO

A exposição das infraestruturas de comunicação das organizações à rede mundial de computadores é uma realidade impossível de ser revertida. Especialistas em Segurança da Informação (SI) são unânimes em afirmar que um sistema computacional só pode ser considerado seguro se estiver desconectado de qualquer rede e isolado por bloqueios físicos. Diante desse cenário, a utilização de ferramentas físicas e lógicas que protejam os ativos de tecnologia da informação e comunicação das organizações contra acessos internos ou externos indevidos, vêm sendo desenvolvidos e aplicados por profissionais com diferentes níveis de conhecimento. Dentre os mecanismos de proteção mais utilizados, destaca-se o firewall; compreendido como um software ou hardware que segundo configurações prévias bloqueia ou permite conexões aos ativos de uma rede de computadores organizacional.

Segundo o Centro de Estudos, Respostas e Tratamento de Incidentes de Segurança da Informação Brasileiro [2], entre janeiro e dezembro de 2017, mais de 50% dos incidentes reportados ao órgão eram do tipo *scan*[1]. Esse tipo de ação é considerado pela literatura de SI como o primeiro passo em um processo de invasão a uma rede de computadores, sendo combatido, principalmente, com o uso de regras aplicadas ao firewall da empresa. No mercado há diferentes tipos de fornecedores e ferramentas que aplicam funcionalidades de firewall, destacando-se o *Iptables*, recurso nativo das distribuições Linux e amplamente utilizado por organizações de diferentes segmentos e tamanhos. Por se tratar de um recurso destinado a administradores do sistema operacional Linux, os comandos *Iptables* se baseiam em regras complexas, inseridas, geralmente, em scripts ou via *prompt* de comando, dificultando a sua utilização, compreensão e disseminação entre profissionais. Há esforços relevantes nas comunidades online para a propagação das regras *Iptables* que são aplicáveis aos inúmeros tipos de ataques às redes de computadores, mas esses esforços se baseiam em uma interação entre usuários de fóruns de discussão, levando a recorrentes assincronias na comunicação.

Agrega-se aos fatores expostos a baixa disponibilidade de profissionais capacitados [2]na área de SI, levando as arquiteturas tecnológicas corporativas à exposição de vulnerabilidades, exploração por parte de pessoas com baixo conhecimento ou por ferramentas automatizadas.

Nesse contexto, este estudo objetiva descrever e apresentar um sistema de aprendizagem de firewall orientado por redes neurais que seja capaz de auxiliar estudantes e profissionais em SI a compreender regras aplicáveis a determinados tipos de ataque a redes de computadores. Para atingir o objetivo proposto, definiu-se quatro objetivos específicos, a saber: O1: realizar levantamento bibliográfico dos temas redes neurais, segurança de redes de computadores e firewall *Iptables*; O2: identificar questionamentos em linguagem natural sobre regras de firewall; O3: analisar e refinar os dados de entrada da rede neural; O4: desenvolver um sistema computacional em linguagem gráfica baseado em uma rede neural treinada.

Metodologicamente a pesquisa se caracteriza como do tipo Design Science (DS), exploratória e bibliográfica. A utilização da metodologia Design Science se justifica pela proposta de desenvolvimento de um artefato software como produto da pesquisa. Este estudo está dividido em cinco seções, além desta introdução. A seção dois apresenta o referencial teórico do estudo, sendo este composto pela apresentação conceitual dos princípios do firewall *Iptables*, introdução ao conceito de redes neurais, apresentação das técnicas de análise e identificação de textos e estudos correlatos ao tema proposto. A terceira seção traz os aspectos metodológicos. Finaliza-se o estudo com a apresentação do artefato de software desenvolvido. Informamos que este estudo é uma versão revisada do trabalho apresentado no IV Simpósio Mineiro de Gestão, Educação, Comunicação e Tecnologia da Informação (SIMGETI) [11].

2. REFERENCIAL TEÓRICO

O referencial teórico deste estudo apresenta uma compreensão do conceito de firewall e suas características, para que, nos tópicos seguintes, sejam discutidos aspectos de redes neurais e algoritmos para mineração de textos. Estudos correlatos ao tema proposta são apresentados ao final do referencial teórico.

2.1 Conceito de *Firewall* e suas Características

Pode-se compreender o conceito de firewall como uma solução de segurança baseada em hardware ou software (mais comum) que separa duas ou mais redes, cuja função é a filtragem dos pacotes de dados, concedendo ou negando acesso, de acordo com um conjunto de regras ou instruções. Classificam-se os firewalls em três tipos: pacotes, aplicação e de inspeção de estado, variando de acordo com a tecnologia de filtragem empregada. Os mais comumente utilizados são os firewalls baseados em pacotes e estado [12].

Os firewalls de pacotes atuam na camada de rede e de transporte da pilha de protocolos TCP/IP, a filtragem dos pacotes TCP são feitas com base no endereço de origem, endereço de destino, porta de origem e porta de destino, contidos no cabeçalho do pacote. Já a análise dos pacotes UDP são analisados pela porta de origem e destino e, por fim, os pacotes ICMP são verificados por meio das informações do código e tipo de mensagem de controle ou erro. As vantagens do firewall de pacotes incluem baixo *overhead*[3], alto

[1] *Scan*: notificações de varreduras em redes de computadores, com o intuito de identificar quais computadores estão ativos e quais serviços estão sendo disponibilizados por eles. É amplamente utilizado por atacantes para identificar potenciais alvos, pois permite associar possíveis vulnerabilidades aos serviços habilitados em um computador.

[2] Canaltech: A falta de ritmo entre a Era Digital e o profissional de Segurança da Informação. 06. Mar. 2018. Disponível em: <https://canaltech.com.br/mercado/a-falta-de-ritmo-entre-a-era-digital-e-o-profissional-de-seguranca-da-informacao-109344/>. Acesso em: 28. jul 2018.

[3] *Overhead*, em ciência da computação, é geralmente considerado qualquer processamento ou armazenamento em excesso, seja de tempo de computação, de memória, de largura de banda ou qualquer outro recurso que seja requerido para ser utilizado ou gasto para executar uma determinada tarefa. Como consequência pode piorar o desempenho do aparelho que sofreu o *overhead*.

desempenho da rede, baixo custo, simplicidade, flexibilidade na implementação e transparência ao usuário. Em relação às desvantagens, pode-se citar o difícil gerenciamento em ambientes complexos, não oferece autenticação de usuários, dificuldade de filtragem para serviços que utilizam portas dinâmicas, e vulnerabilidade contra-ataques do tipo IP *spoofing* [4][12].

2.2 Firewall *Iptables*

O firewall *Iptables* é um filtro de rede (*netfilter*) que veio em substituição à ferramenta *Ipchains*. O *Iptables* é um *firewall* em nível de pacotes cujo funcionamento se baseia na análise das informações de marcações e datagramas do pacote (endereço/porta de origem/destino do pacote, prioridade, entre outros). Funciona por meio da comparação de regras para saber se um pacote tem ou não permissão para passar.

O *netfilter* é um conjunto de mecanismos para manipulação de pacotes implementados diretamente no *kernel* do sistema operacional *Linux*. Os mecanismos do *netfilter* são selecionados por meio de três tabelas: (i) *filter*: utilizada para filtrar pacotes; (ii) *mangle*: utilizada para ações de marcação e manipulação de pacotes; (iii) *nat*: utilizada para as ações de tradução de endereços (NAT - *Network Address Translation* - Tradução de endereços de Rede e NAPT - *Network Address and Port Translation* - Tradução de endereços e portas de rede) [13].

O *Iptables* corresponde aos mecanismos utilizados para organizar as regras utilizadas pelo *netfilter*, utiliza o conceito de *chains* para indicar quais pacotes terão uma determinada regra aplicada. As *chains* utilizadas pelo mecanismo de filtragem de pacotes são: (i) *input*: as regras se referem aos pacotes que entram por uma interface; (ii) *output*: as regras se referem ao que sai por uma interface; *forward*: as regras se aplicam aos pacotes que serão roteados, isto é, aos pacotes que não são destinados ao próprio computador. A sintaxe para uma regra *Iptables* poder ser: *Iptables* <tabela><*chain*><opções><regra/ação> [13].

Uma regra de firewall *Iptables* é formada pela combinação de *flag* de inserção ou remoção, uma tabela, uma *chain*, *flags* para especificar portas e IPs de origem e destino, *flag* de protocolo, protocolo e opções de ação e ação.

Na seção seguinte são apresentados os tipos de ataques e serviços analisados neste estudo.

2.2.1 Regras de Firewall Iptables

Para este estudo foram selecionados dois tipos de ataques a redes de computadores, a saber: *syn flooding* e *brute force*. A escolha se deu devido à variabilidade de regras de firewall que esses tipos de ataque possuem e a sua capacidade de comprometer um sistema computacional. As respectivas explicações técnicas das regras de *firewall* estão disponíveis na página do projeto na comunidade GitHub[5]. As regras do Quadro 1 bloqueiam ataques de *syn flooding* em um servidor web.

Quadro 1 - Regras do *firewall Iptables* syn flooding

```
echo 1 > /proc/sys/net/ipv4/tcp_syncookies
echo 2048 > / proc / sys / net / ipv4 / tcp_max_syn_backlog
echo 3 > / proc / sys / net / ipv4 / tcp_synack_retries
iptables -N syn-flood
iptables -A INPUT -i $WAN -p tcp --syn -j syn-flood
iptables -A syn-flood -m limit --limit 10/s --limit-burst 4 -j RETURN
iptables -A syn-flood -j DROP
```

Fonte: [23].

As regras do Quadro 2 bloqueiam ataques de *brute force* em um servidor ssh (*secure shell*).

Quadro 2 - Regras do *firewall Iptables* ssh

```
iptables -A INPUT -p tcp –dport 22 -i eth0 -m state --state NEW -m recent --name CONSSH --set -j LOG --log-prefix ”CONEXÕES SSH “
iptables -A INPUT -p tcp –dport 22 -i eth0 -m state --state NEW -m recent --name CONSSH --rcheck --seconds 180 --hitcount 2 -j DROP
```

Fonte: [24].

A regra do Quadro 3 bloqueia ataque de *brute force* em um formulário de *login*.

Quadro 3 - Regras do *firewall Iptables* formulário de login

```
iptables -A INPUT -p tcp --dport 80 -m limit --limit 25/minute --limit-burst 100 -j ACCEPT
```

Fonte: [25].

As regras do Quadro 4 bloqueiam ataques de *brute force* no servidor ftp (*file transfer protocol*).

Quadro 4 - Regras do *firewall Iptables* FTP

```
iptables -A INPUT -i eth0 -p tcp -m multiport --dports 20,21 -m state --state NEW -m recent --set --name FTP
sudo iptables -A INPUT -i eth0 -p tcp -m multiport --dports 20,21 -m state --state NEW -m recent --rcheck --seconds 60 --hitcount 4 --rttl --name FTP -j DROP
```

Fonte: [26].

As regras do Quadro 5 bloqueiam ataques de *brute force* no servidor *telnet*.

Quadro 5 - Regras do *firewall Iptables* TELNET

```
iptables -A INPUT -p tcp –dport 23 -j DROP
```

Fonte: [27].

As regras do Quadro 6 bloqueiam ataques de *brute force* no servidor RDP (*Remote Desktop Protocol*).

Quadro 6 - Regras do *firewall Iptables* RDP

```
iptables -A INPUT -m state --state ESTABLISHED -j ACCEPT
iptables -A INPUT -p tcp --dport 3389 -m recent --update --seconds 180 -j DROP
iptables -A INPUT -p tcp --dport 3389 --tcp-flags syn,ack,rst syn -m recent --set -j ACCEPT
```

[4] *IP spoofing* é um ataque que consiste em mascarar (*spoof*) pacotes IP utilizando endereços de remetentes falsificados.

[5] Página de desenvolvimento do projeto: <https://github.com/franciscarlospereira/IpTabLearing.git> .

Fonte: [28]

As regras do Quadro 7 bloqueiam ataques de *brute force* no servidor MYSQL.

Quadro 7 - Regras do *firewall Iptables* MYSQL

```
iptables -A INPUT -p tcp -m state --syn --state NEW --dport 3306 -m limit --limit 1/minute --limit-burst 1 -j ACCEPT
iptables -A INPUT -p tcp -m state --syn --state NEW --dport 3306 -j DROP
```

Fonte: [29].

As regras do Quadro 8 bloqueiam ataques de *brute force* no servidor FIREBIRD.

Quadro 8 - Regras do *firewall Iptables* FIREBIRD

```
iptables -A INPUT -p tcp -m state --syn --state NEW --dport 3050 -m limit --limit 1/minute --limit-burst 1 -j ACCEPT
iptables -A INPUT -p tcp -m state --syn --state NEW --dport 3050 -j DROP
```

Fonte: [30]

As regras do Quadro 9 bloqueiam ataques de *brute force* no servidor SAMBA.

Quadro 9 - Regras do *firewall Iptables* SAMBA

```
iptables -A INPUT -i eth0 -p udp --dport 137 -m state --state NEW -m recent --set --name netbios-ns
iptables -A INPUT -i eth0 -p udp --dport 137 -m state --state NEW -m recent --update --seconds 20 --hitcount 2 --rttl --name netbios-ns -j DROP
iptables -A INPUT -i eth0 -p udp --dport 138 -m state --state NEW -m recent --set --name netbios-dgm
iptables -A INPUT -i eth0 -p udp --dport 138 -m state --state NEW -m recent --update --seconds 20 --hitcount 2 --rttl --name netbios-dgm -j DROP
iptables -A INPUT -i eth0 -p tcp --dport 139 -m state --state NEW -m recent --set --name netbios-ssn
iptables -A INPUT -i eth0 -p tcp --dport 139 -m state --state NEW -m recent --update --seconds 20 --hitcount 2 --rttl --name netbios-ssn -j DROP
iptables -A INPUT -i eth0 -p tcp --dport 445 -m state --state NEW -m recent --set --name microsoft-ds
iptables -A INPUT -i eth0 -p tcp --dport 445 -m state --state NEW -m recent --update --seconds 20 --hitcount 2 --rttl --name microsoft-ds -j DROP
```

Fonte: [31].

As regras do Quadro 10 bloqueiam ataques de *brute force* no servidor web com suporte de segurança SSL (*Secure Socket Layer*).

Quadro 10 - Regras do *firewall Iptables* HTTPS

```
iptables -N SSL-Brute-Force
iptables -A INPUT -m tcp -p tcp --dport 443 -j SSL-Brute-Force
iptables -A SSL-Brute-Force -m state --state NEW -m recent --set --name SSL --rsource
iptables -A SSL-Brute-Force -m state --state NEW -m recent --rcheck --seconds 600 --hitcount 5 --name SSL --rsource -j LOG --log-prefix "[SSL Brute Force] "
iptables -A SSL-Brute-Force -m state --state NEW -m recent --rcheck --seconds 600 --hitcount 5 --name SSL --rsource -j DROP
iptables -A SSL-Brute-Force -j ACCEPT
```

Fonte: [32].

As regras do Quadro 11 bloqueiam ataques de *brute force* em um servidor web executando em seu porta padrão (80).

Quadro 11 - Regras do *firewall Iptables* HTTP

```
$IPTABLES -N HTTP-Brute-Force
$IPTABLES -A INPUT -m tcp -p tcp --dport 80 -j HTTP-Brute-Force
$IPTABLES -A HTTP-Brute-Force -m state --state NEW -m recent --set --name HTTP --rsource
$IPTABLES -A HTTP-Brute-Force -m state --state NEW -m recent --rcheck --seconds 600 --hitcount 5 --name SSL --rsource -j LOG --log-prefix "[HTTP Brute Force] "
$IPTABLES -A HTTP-Brute-Force -m state --state NEW -m recent --rcheck --seconds 600 --hitcount 5 --name SSL --rsource -j DROP
$IPTABLES -A HTTP-Brute-Force -j ACCEPT
```

Fonte: [31].

As regras dos quadros acima foram armazenadas em uma estrutura de dados de forma que pudessem ser consultadas de acordo com a pergunta realizada pelo usuário do *software*. A base técnica/teórica que fundamenta a correta resposta ao usuário, é apresentada na próxima seção.

2.3 Redes Neurais

Desenvolvidas na década 60, as redes neurais tiveram pouca aplicação comercial, reservando-se aos laboratórios das universidades. A sua eficácia e eficiência eram seriamente comprometidas devido à falta de informações digitais. Com a atual quantidade de dados difundido pela rede mundial de computadores, as redes neurais passaram a figurar na vanguarda da grande área da Inteligência Artificial (IA) [5].

Uma rede neural artificial é um modelo constituído de uma estrutura de neurônios artificiais interligados entre si por meio de sinapses e dendritos. Uma sinapse é o nome dado à conexão existente entre neurônios. Nessas conexões são atribuídos valores, que são chamados de pesos sinápticos. As primeiras modelagens das redes neurais artificiais foram feitas por meio da observação e estudo do funcionamento dos neurônios do cérebro humano [5].

O neurônio do cérebro humano recebe as informações com origem nos impulsos elétricos de outros neurônios por meio de suas sinapses e os transmite para o neurônio seguinte por meio de seus dendritos. Estima-se que haja aproximadamente 10 bilhões de neurônios no córtex humano e 60 trilhões de sinapses ou conexões no cérebro humano. A ideia básica das redes neurais artificiais é realizar o processamento das informações tendo como princípio a organização de neurônios do cérebro humano. Um dos benefícios é sua habilidade de aprender e generalizar [5]. A Figura 1, a seguir, exemplifica o funcionamento de um neurônio.

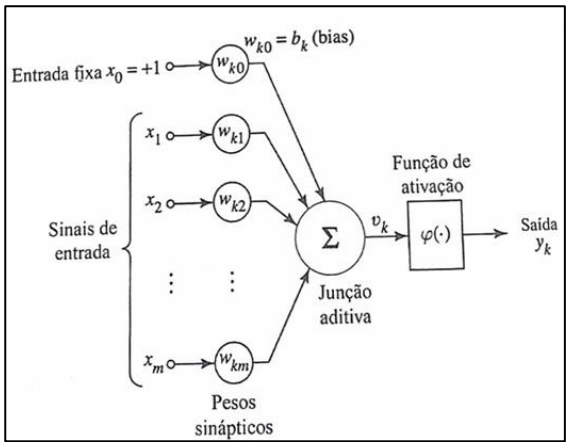

Figura 1 - Modelagem de um neurônio artificial

Fonte: [5, p. 36].

Basicamente, cada neurônio recebe um valor de entrada (sinal de entrada) que é multiplicado pelo seu respectivo peso. Esse procedimento acontece para todos os neurônios de entrada e se propaga para os demais neurônios, de acordo com a interconexão estabelecida entre eles. O resultado do somatório é adicionado um valor de bias (polarização), o qual tem um efeito de aumentar ou diminuir a entrada líquida da função de ativação. Finalmente, o valor obtido yk é submetido a uma função de ativação, que define o valor de saída do neurônio, com base no valor de uk. Se o valor de uk >= 0, resulta em yk = 1. Caso contrário se uk < 0, resulta yk = 0. Caso o valor seja 1 significa que o valor do peso sináptico foi aceito (ativado) para as variáveis de entrada (sinal de entrada) que conecta ao neurônio em questão [5].

Segundo [5, p.75] "uma rede neural aprende acerca do seu ambiente por meio de um processo iterativo de ajustes aplicados a seus pesos sinápticos e níveis de bias. Idealmente, a rede se torna mais instruída sobre o seu ambiente após cada interação do processo de aprendizagem"

O algoritmo utilizado para realizar o treinamento da rede neural deste estudo segue o paradigma de aprendizado supervisionado. No aprendizado supervisionado a máquina é treinada por meio de um conjunto de dados onde para cada entrada a saída é conhecida. Os dados para este tipo de método devem possuir rótulos. É possível comparar a técnica a um aprendizado acompanhado, onde o professor supervisiona todo o processo de aprendizado, dizendo o que é certo, o que é errado e aonde se quer chegar [10]. As figuras 2, 3 e 4, a seguir, apresentam as estruturas matemáticas para o cálculo dos pesos nos neurônios.

$$y_k = \begin{cases} 1 & \text{se } v_k \geq 0 \\ 0 & \text{se } v_k < 0 \end{cases}$$

Figura 2 - Sistema de equação

$$v_k = \sum_{j=1}^{m} w_{kj} x_j + b_k$$

Figura 3 - Formula vk = uk + bk

As variáveis x1, x2, …. xm são sinais de entrada para os neurônios. As variáveis wk1, wk2 ….. wkm são os pesos sinápticos do neurônio k. Uk é a saída resultante do somatório dos sinais de entrada, ponderados pelas respectivas sinapses do neurônio. Bk é o bias e $\sigma(.)$ é a função de ativação, yk é o sinal de saída do neurônio. [5, p.39]. Uk =∑_(j=1)^m wkj xj é o somatório da multiplicação dos sinais de entrada pelos respectivos pesos de cada neurônio, produzindo uma saída uk [5, p.37].

Existem muitas funções de ativação, mas a função sigmóide é a mais utilizada na construção de redes neurais artificiais [5].

$$\varphi(v) = \frac{1}{1 + \exp(-av)}$$

Figura 4 - Função sigmoide

Fonte: [5, p. 40].

O aprendizado de uma rede neural acontece por meio da aplicação de um algoritmo de aprendizagem. O algoritmo realiza uma série de cálculos em toda a cadeia de neurônios para o devido treinamento da rede neural.

2.3.1 Conceito de processamento de linguagem natural

O processamento de linguagem natural busca criar mecanismos computacionais e algoritmos capazes de fazer um computador entender textos escritos em linguagem humana [22].

O Processamento de Linguagem Natural (PLN) é a subárea da IA que estuda a capacidade e as limitações de uma máquina em entender a linguagem dos seres humanos. O objetivo do PLN é fornecer aos computadores a capacidade de entender e compor textos. "Entender" um texto significa reconhecer o contexto, fazer análise sintática, semântica, léxica e morfológica, criar resumos, extrair informação, interpretar os sentidos, analisar sentimentos e até aprender conceitos com os textos processados [22].

A aplicação das técnicas de mineração de texto permite que os dados textuais possam ser processados (estruturados) a ponto de serem transformados em dados para análise. Assim, a mineração de texto é um recurso computacional que torna viável o processamento de linguagem natural.

2.3.2 Mineração de texto

A mineração de texto pode ser entendida como um processo que utiliza algoritmos capazes de analisar documentos de texto. Os dados textuais dos documentos não possuem um esquema para descrever sua estrutura, ou seja, os dados textuais não estão organizados em campos e tabelas relacionais, conforme acontece com os dados estruturados em um sistema gerenciador de banco de dados [4].

As informações em texto plano ou semiestruturada não estão representadas de uma forma que facilite o processo de análise automática, já que não se dispõe de um esquema rígido para descrever os dados e guiar o processo de mineração. Além disso, quando se trabalha com texto, é preciso conviver com uma série de problemas inerentes ao processo de interpretação do texto, como: existência de sinônimos, erros ortográficos, diversidade de idiomas, recursos estilísticos (metáfora, metonímia, ironia, etc.), entre outros.

Este conjunto de dificuldades faz com que a mineração de textos precise utilizar técnicas específicas e diferentes das utilizadas na mineração de dados tradicional [4].

Tem-se duas abordagens para trabalhar com o processo de mineração de textos: abordagem estatística, que se baseia na frequência dos termos dentro de um texto, ignorando qualquer informação semântica; e a abordagem de processamento de linguagem natural, onde o algoritmo tenta processar os textos da mesma forma que os seres humanos fazem ao ler um texto, ou seja, interpretando as estruturas sintáticas e semânticas das frases. As etapas envolvidas no processo de mineração de texto são: (i) coleta de documentos; (ii) pré-processamento; (iii) extração de conhecimento; e (iv) avaliação e interpretação dos resultados [4]. Essas etapas são destacadas na metodologia do estudo.

A mineração de texto permite que os dados textuais sejam estruturados, ficando prontos para o processo de análise, classificação e aprendizagem da rede neural. Esse processo é feito por meio do algoritmo *naive bayes*.

2.3.3 Algoritmo Naive Bayes

O algoritmo *naive bayes* pode ser entendido como uma técnica de classificação baseado no teorema de *Bayes* com uma suposição de independência entre os preditores. Em outras palavras, um classificador *naive bayes* assume que a presença de uma característica particular em uma classe não está relacionada com a presença de qualquer outro recurso. Por exemplo, um fruto pode ser considerado como uma maçã se é vermelho, redondo, e tiver cerca de 3 polegadas de diâmetro. Mesmo que esses recursos dependam uns dos outros ou da existência de outras características, todas estas propriedades contribuem de forma independente para a probabilidade de que este fruto é uma maçã [21]. O algoritmo recebe o termo "*naive*" (ingênuo) no seu nome, pois desconsidera completamente a correlação entre as variáveis (*features*). Ou seja, se determinada fruta é considerada uma "Maçã", se ela é "Vermelha", "Redonda" e possui aproximadamente "10cm de diâmetro", o algoritmo não vai levar em consideração a correlação entre esses fatores, tratando cada um de forma independente [1].

O teorema de *Bayes* fornece uma forma de calcular a probabilidade posterior P (C | X) a partir de P (C), P (x) e P (X | c), conforme a Figura 5.

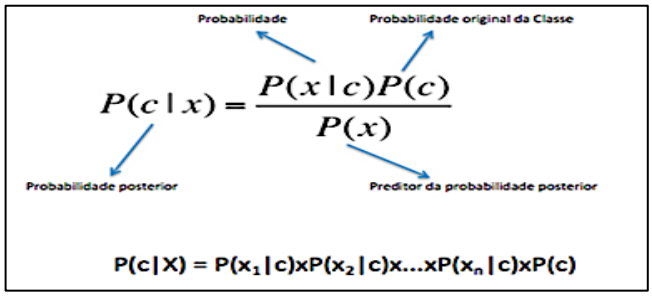

Figura 5 - Fórmula da probabilidade condicional
Fonte: [21].[6]

Conforme a fórmula, acima, tem-se que P (c | x) é a probabilidade posterior da classe (c, alvo) dada preditor (x, atributos), P (c) é a probabilidade original da classe, P (x | c) é a probabilidade que representa a probabilidade de preditor dada a classe e P (x) é a probabilidade original do preditor [21].

Dado um conjunto de dados de entrada é criado uma tabela de frequência, onde são computadas as ocorrências (repetições) de cada palavra em função das classes que deseja alcançar. Em seguida, com base na tabela de frequência, é criada a tabela de probabilidade, dividindo-se o número de repetições de cada palavra pelo número total de palavras. Na sequência, divide-se o número total de ocorrências das palavras pertencentes a cada classe pelo total de palavras. Finalmente, obtém-se o valor com a maior probabilidade posterior [21].

O algoritmo *naive bayes* tem bom desempenho quando trabalha com variáveis categóricas, quando comparado com variáveis numéricas. Para variáveis numéricas pode-se aplicar a distribuição normal (curva de sino) [21]. *Naive Bayes* usa um método similar para prever a probabilidade de classe diferente com base em vários atributos. Este algoritmo é usado principalmente em classificação de texto e com problemas que possuem múltiplas classes [21]. O algoritmo *naive bayes* é aplicado nas áreas de previsões em tempo real, previsões multi-classes, classificação de textos, filtragem de spam, análise de sentimento e em sistemas de recomendação [21].

O resultado do processo de classificação do algoritmo *naive bayes* gera uma matriz de resultados, chamada matriz de confusão, que se assemelha a matriz de probabilidades criada entre os dados de testes e a classificação feita pelo algoritmo *naive bayes*.

2.3.4 Matriz de confusão

A matriz de confusão é uma tabela que permite visualizar o desempenho do classificador [15]. Acurácia e eficiência são métricas destinadas a avaliar o nível de aprendizagem de uma rede neural. A acurácia é uma métrica básica para medir o aprendizado, leva em consideração a proporção de predições corretas, sem levar em consideração a quantidade de erros. A acurácia é dada pelo número de acertos dividido pelo total de eventos [10]. A sensibilidade é uma métrica que fornece a proporção de verdadeiros positivos, ou seja, a capacidade do algoritmo em prever casos corretamente para os casos que são realmente verdadeiros [10].

A especificidade é uma métrica que mostra a proporção de verdadeiros negativos, o algoritmo prediz situações erradas que realmente são falsas [10]. Finalmente, a eficiência é uma métrica que retorna a média aritmética da sensibilidade e especificidade. Essa métrica indica se o algoritmo está mais suscetível a ter verdadeiros positivos ou verdadeiros negativos [10].

2.4 Estudos correlatos

[14] conduziram um estudo utilizando duas Redes Neurais Artificiais (RNAs) para automatizar a análise de um grande volume de dados em redes de computadores que utilizam os protocolos TCP e UDP. Para atingir o objetivo do estudo, as RNAs Adaline e Perceptron foram implementadas por meio dos seguintes recursos técnicos: i) o uso do software Tcpdump para capturar e minerar os dados; ii) o desenvolvimento da interface gráfica por meio da linguagem C#, *framework* .NET e um sistema de gerenciamento de banco de dados PostgreSQL para padronização dos dados. Posteriormente, após importar os dados coletados na interface gráfica, foi possível obter, classificar padrões e identificar anomalias e falhas considerando os

[6] Naive-Bayes Python: Disponível em: <https://www.vooo.pro/insights/6-passos-faceis-para-aprender-o-algoritmo-naive-bayes-com-o-codigo-em-python/> Acesso em: 30 set. 2018.

protocolos de rede utilizados. Dessa forma, o estudo demonstrou uma maneira produtiva e aplicável ao problema de classificação de dados trafegados em redes de computadores. O artefato de *software* desenvolvido se mostrou com potencial para ser inserido no cenário das ferramentas já existentes para classificação do tráfego de redes de computadores.

[8] também utilizaram a RNA Perceptron com o objetivo de categorizar o desempenho da qualidade de serviço de uma rede virtual privada (VPN – Virtual Private Network). Para realização dos treinamentos, foram coletados os dados de tráfego sem e com multimídia, e como referências foram aplicados os elementos que medem o atraso na entrega de dados na rede, a taxa de transferência, perda de pacotes e os dados que foram transferidos. Utilizando uma classificação de três níveis (baixo, médio e alto) em relação à distribuição de conteúdo após o processo de análise e classificação, foi possível concluir que a utilização da RNA Perceptron apresentou resultados acima do esperado para o problema proposto.

No estudo de [16], um sistema auto adaptativo denominado STARK foi capaz de demonstrar a capacidade de predição de dados em relação a ataques distribuídos de negação de serviço (DDoS - *Distributed Denial of Service*). Utilizando a teoria da metaestabilidade, o artefato de software desenvolvido utiliza a aprendizagem estatística por meio do aprendizado de máquina não supervisionado. Para avaliação de desempenho do algoritmo desenvolvido, os autores utilizaram um conjunto de dados disponibilizados pelas instituições CAIDA, CTU e DARPA de ataques DDoS reais que foram registrados. O algoritmo permitiu identificar ataques às redes de maneira antecipada. Os indícios de sobrecarga foram observados com uma antecedência de, aproximadamente, vinte e três minutos na rede CIDA, uma hora na CTU e duas horas na DARPA. Assim, o estudo demonstrou a eficiência de um sistema de predição de ataques do tipo DDoS nas redes estudadas, fornecendo aos administradores da rede uma rápida capacidade de reação.

No estudo de [6], observaram-se novas propostas para detecção de anomalias na internet empregando técnicas de aprendizagem de máquina. Entende-se por detecção de anomalias na internet a construção de perfis de comportamento para padrões considerados como atividade normal. Na abordagem dos autores, foi utilizado um conjunto de classificadores (k-NNs - k-Nearest Neighbor) gerado por meio de um método aleatório (RSM - Random Subspace Method) com o intuito de aumentar a taxa de detecção de anomalias em redes de computadores. O método utilizado pelos autores pode ser comparado à técnica denominada TANN (Triangle Area based Nearest Neighbor). Foi possível observar que a combinação dos dois algoritmos (k-NNs e RSM) atingiu um desempenho superior, se utilizados de maneira isolada. O estudo evidenciou que esses classificadores, quando combinados, tornam-se ferramentas fundamentais no desenvolvimento de soluções precisas para a detecção de anomalias na internet.

3. MATERIAIS E MÉTODOS

Esta pesquisa caracteriza-se como do tipo *Design Science* (DS), exploratória e bibliográfica. A pesquisa *Design Science* é centrada em problemas e orientada a prática do cotidiano, do fazer [19]. A DS objetiva identificar o que é eficaz, e produzir um artefato relevante e útil ao problema proposto. [7] estabelece sete diretrizes para a DS: (i) o objeto de estudo deve ser um artefato; (ii) O problema abordado pelo artefato deve ser relevante aos praticantes; (iii) a avaliação da utilidade do artefato deve ser rigorosa; (iv) deve haver contribuição efetiva para a área de conhecimento do artefato; (v) pesquisa rigorosa; (vi) uso eficiente de recurso; (vii) comunicação dos resultados aos praticantes.

A DS se enquadra à pesquisa em desenvolvimento por propor um artefato novo, relevante para os estudos da área pesquisada e com aplicabilidade prática. O Quadro 12, a seguir, relaciona as diretrizes da DS ao estudo.

Quadro 12 - Diretrizes da pesquisa Design Science aplicada a pesquisa

Diretriz	**Descrição da diretriz**	**Aplicação a pesquisa**
Relevância do Problema	A pesquisa aplicada na ciência do DS deve proporcionar novas contribuições e conhecimentos para elaboração de um projeto.	O estudo de sistemas inteligentes na área de segurança de informação busca atender uma demanda crescente por ferramentas automatizadas de ataque e defesa de redes de computadores.
Artefato	Consiste em aplicar o estudo a um objeto denominado artefato. Considera-se um artefato aquilo que é construído pelo homem, ou seja, tudo que não é natural. Em DS, denomina-se artefato o objeto de estudo que se propõe a resolver os problemas levantados em uma pesquisa.	Neste estudo, propõe-se o desenvolvimento de um sistema computacional dotado de inteligência artificial capaz de auxiliar no processo de ensino/aprendizagem de regras de *firewall Iptables*.
Processo de Busca da Solução	O problema levantado na pesquisa precisa ser motivador, interessante e precisa oferecer uma solução benéfica para o usuário final.	O problema em questão é o ensino e a aprendizagem de regras de *firewall Iptables* por parte de alunos e profissionais da área de Tecnologia da Informação.
Rigor da Pesquisa	Necessita de uma captação intermitente de dados precisos e realização de técnicas analíticas dessas informações para construção do projeto.	Buscou-se por meio de contato com especialistas em SI, fóruns de discussão e docentes de cursos relevantes na área definir os dados de aprendizagem da rede neural proposta.
Avaliação	É necessário executar uma avaliação rigorosa testando os atributos	Os testes do sistema foram realizados por meio das práticas

	dos artefatos por meio de métodos precisos. Esses artefatos podem ser avaliados em termos de funcionalidade, completude, consistência, precisão, performance, confiabilidade e usabilidade.	preconizadas por [17].
Contribuições da Pesquisa	Detalhar a contribuição do estudo para a comunidade científica e área de aplicação do artefato.	O estudo teórico realizado, bem como o código fonte do sistema desenvolvido foi disponibilizado em plataforma livre e gratuita. O estudo contribuiu para o aprofundamento na relação inteligência artificial e SI.
Comunicação da Pesquisa	Comunicação detalhada dos resultados e pesquisas aos contribuintes, sendo eles, profissionais, pesquisadores, estudantes.	A comunicação da pesquisa foi realizada por meio de congresso científico e divulgação entre estudantes e profissionais das áreas de tecnologia da informação e SI.

Fonte: Desenvolvido pelos autores (2019).

Cada uma das diretrizes possui ações a serem realizadas pelos pesquisadores com objetivo de atender as necessidades da DS. A pesquisa exploratória, para [18, p. 132], é um "levantamento de informações sobre determinado objeto, delimitando um campo de trabalho". O objeto em exploração pode ser entendido como a "compreensão por parte do algoritmo inteligente dos questionamentos realizados pelo usuário do sistema, e a respectiva apresentação da (s) regra(s) de *firewall Iptables*". Utilizou-se ainda como recurso metodológico a pesquisa bibliográfica. A pesquisa bibliográfica fundamenta-se em documentos, livros, artigos, teses, etc. para alicerçar os argumentos do pesquisador [18].

A coleta de dados para viabilização da base de conhecimento da rede neural se deu por meio da participação de especialistas em SI e consulta a fontes primárias, como fóruns de discussão disponíveis na rede mundial de computadores (dados disponíveis no sítio oficial do projeto[7]). Para [20], os dados primários podem ser obtidos por meio do levantamento de informações já existentes.

A estruturação dos dados coletados para o treinamento da rede neural aconteceu em três etapas: (i) coleta de questionamentos sobre os tipos de ataques pré-definidos pelos pesquisadores (*brute force* e *Denial of Service* (DoS); (ii) tabulação dos dados em planilha Excel; e (iii) agrupamento pelo tipo de ataque e tipo de serviço, conforme Quadro 13, a seguir.

Quadro 13 - Descritivo que relaciona tipo de ataque com o serviço

Tipo de ataque	**Serviço**
Brute force	Servidor WEB, Formulário de *login*, HTTP, FTP, HTTPS, SSL, SSH, SAMBA, RDP, Telnet, MySQL, Firebird.
DOS	Syn Flood

Fonte: Desenvolvido pelos autores (2018).

Na coleta de informações desta pesquisa, buscou-se perguntas em fóruns especializados na área de SI; em seguida, as perguntas em linguagem natural passaram por um pré-processamento, a fim de remover as *stop words* (artigos, advérbios e pronomes) - termos com pouca significação semântica. Além disso, foi utilizado a técnica de *stemming* (radicalização das palavras), a fim de diminuir a dimensionalidade dos dados a serem processados. Esta redução permitiu a redução do custo computacional das etapas seguintes do processo [3]. Na terceira etapa, ocorreu a extração do conhecimento por meio de algoritmos de extração automática de conhecimento. Nessa etapa, as informações são agrupadas em grupos com características dissimilares entre si [3]. Na última etapa ocorreu a avaliação e interpretação dos resultados com a ajuda de um especialista para descobrir se os resultados obtidos são satisfatórios, e em caso negativo, que etapas poderiam ser refeitas para melhorias [9].

A base de dados final para treinamento da rede neural totalizou 1089 perguntas, divididas em nove classes (*ssh brute force, ftp brute force, samba brute force, mysql brute force, rdp brute force, https brute force, http brute force, telnet brute force, Dos syn-flooding*), compondo 121 perguntas para cada classe.

4. APRESENTAÇÃO DO ARTEFATO DE SOFTWARE

Foi utilizado a IDE Pycharm (*Integrated Development Environment* – Ambiente de Desenvolvimento Integrado), juntamente com a distribuição anaconda *python* versão 3.6, para desenvolver o sistema de aprendizagem de *firewall* orientado por rede neural. O primeiro aspecto a ser mencionado foi a inclusão do módulo *ntlk (natural linguage toolkit)*. Esse módulo permitiu que a linguagem *python* pudesse trabalhar com os algoritmos de redes neurais. Outra inclusão foi o módulo *tkinter*, para permitir o desenvolvimento do sistema de aprendizagem de *firewall* em ambiente gráfico.

O sistema possui os seguintes elementos gráficos: uma caixa de texto, dois rótulos para os textos ("Digite a sua dúvida" e "Pergunta"), dois botões ("Pesquisar" e "Sair") e dois rótulos para exibir as regras *Iptables* e as suas respectivas explicações, conforme Figura 6.

[7]Sítio oficial do projeto: <https://github.com/franciscarlospereira/IpTabLearing.git>.

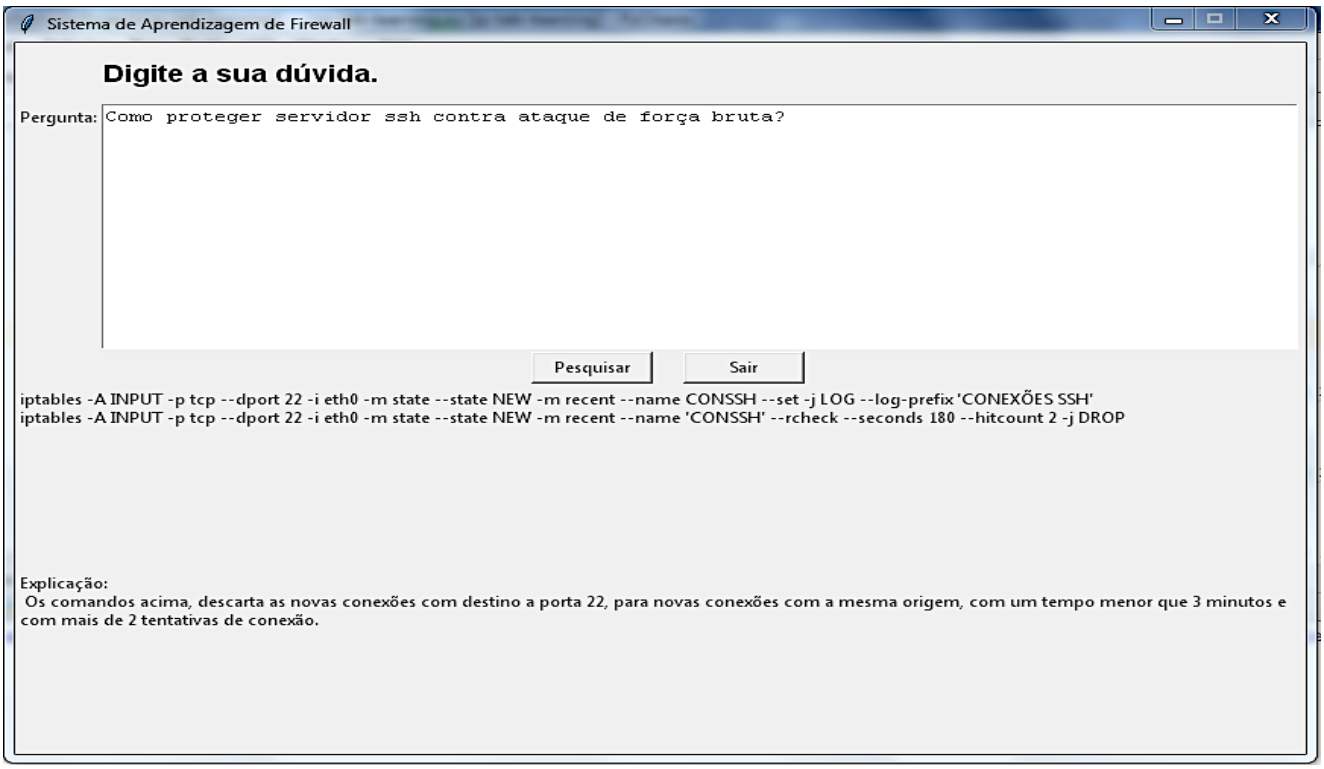

Figura 6 – Interface do *software* desenvolvido

Fonte: Desenvolvida pelos autores (2019).

O usuário digita o seu questionamento na caixa de texto e o sistema retorna a (s) regra (s) *Iptables* correspondentes com a respectiva explicação técnica abaixo. A base de dados de treinamento e teste da rede neural passa por uma fase de pré-processamento. Dentro das várias etapas do pré-processamento, destacam-se os métodos abaixo (Quadros 14, 15 e 16).

Quadro 14 - Cria e remove uma lista de *stopwords* com o uso da biblioteca nltk

```
self.stopwordsnltk = nltk.corpus.stopwords.words('portuguese')
# remove as stopwords e tira os radicais das palavras para reduzir a dimensionalidade.
def aplicastemment(self, classe):
        stemmer = nltk.stem.RSLPStemmer()
    frasesstemming = []
for (palavras, classe) in texto:
        comstemming = [str(stemmer.stem(p)) for p in palavras.split() if p not in self.stopwordsnltk]
frasesstemming.append((comstemming, classe))
return frasesstemming
```

Fonte: Desenvolvido pelos autores (2019).

Após o pré-processamento das bases, os dados previsores são transformados em valores binários (0 ou 1 - *True* ou *False*). O código a seguir apresenta essa conversão (Quadro 15).

Quadro 15 - A função apply_features faz o preenchimento dos registros dos dados previsores com valores *True/False*

```
self.basecompletatreinamento = nltk.classify.apply_features(self.extrator_palavras,
self.frasescomstemmingtreinamento)
self.basecompletateste = nltk.classify.apply_features(self.extrator_palavras,
self.frasecomstemmingteste)
```

Fonte: Desenvolvido pelos autores (2019).

Finalmente, os valores dos pesos são atualizados por meio do método *train* (Quadro 16).

Quadro 16 - Recupera os dados e cria a tabela de probabilidade

```
self.classificador = nltk.NaiveBayesClassifier.train(self.basecompletatreinamento)
```

Fonte: Desenvolvido pelos autores (2019).

Os pesos estão relacionados à probabilidade de que determinados resultados estejam associados à entrada apresentada, ou seja, de acordo com o questionamento apresentado pelo usuário são dados pesos em relação a base de conhecimento da rede neural. O nível de aprendizagem alcançado por uma rede neural é medido por meio de métricas de desempenho. Essas métricas utilizam dados gerados pela matriz de confusão para calcular o número de acertos e erros de uma rede neural. O Quadro 17 apresenta os dados da matriz de confusão para o projeto proposto.

Quadro 17 - Matriz de confusão

- Acurácia: 96,25%;
- Precisão (taxa de falsos positivos): 96,20%;
- Sensibilidade (taxa de falsos negativos): 95,74%;
- F-*score* (média entre a precisão e sensibilidade): 95,83%.

Fonte: Desenvolvido pelos autores (2019).

Inicialmente, tem-se a métrica de acurácia. Definida pelo total de acertos dividido pelo total de amostras. O sistema retornou uma acurácia de 96,25%, o que indica uma alta taxa de acertos do classificador. A métrica precisão fornece a taxa de falsos positivos retornado pelo sistema e obteve um valor de 96,20%, o que indica uma baixa taxa de falsos positivos, aproximadamente 5%. A métrica sensibilidade ou *recall* fornece a taxa de falsos negativos e totalizou 95,74%, o que indica uma baixa taxa de falsos negativos, também próximo de 5%. A métrica F-*score* corresponde a duas vezes a média harmônica entre precisão e sensibilidade. O resultado encontrado para a métrica F-*score* foi de 95,83%. Por fim, argumenta-se que o resultado geral médio em 95% garante ao sistema confiabilidade nas decisões tomadas e consequente correto retorno ao usuário.

O artefato de *software* desenvolvido permite que os principais questionamentos de profissionais ou estudantes de *firewall Iptables* possam obter respostas com alta confiabilidade e em um único ambiente. Diferentemente das práticas atuais em que as informações afetas ao tema estão dispersas em fóruns de discussão ou sítios na internet, o artefato de *software* apresentado centraliza esses dados e processa as solicitações realizadas em linguagem natural (português brasileiro). A dinâmica do artefato de *software* se assemelha a uma interação em um fórum de discussão em que um usuário realiza um questionamento e outro profissional, com mais experiência, envia a resposta. Acredita-se que a ampliação da base de treinamento da rede neural pode possibilitar a centralização dos dados de um determinado assunto e auxiliar no processo de ensino e aprendizado, assim como otimizar a busca por informações com maior assertividade.

As práticas e técnicas utilizadas neste estudo foram observadas em relação a estudos semelhantes da literatura [6; 8; 14; 16]. Percebeu-se que os temas segurança da informação e inteligência artificial têm sido aplicados em conjunto nos estudos científicos, essencialmente, com base no tráfego das redes de computadores e na busca por identificação de padrões no volume de dados gerados pelas comunicações digitais. Essas observações permitem inferir que o produto deste estudo, um artefato de *software* que utiliza fundamentos de inteligência artificial e tem como objetivo facilitar o processo de ensino e aprendizagem de regras de *firewall Iptables*, figura-se como uma abordagem que difere da literatura observada.

Para que o projeto deste estudo possa ser melhorado e/ou aplicado em outros contextos, disponibilizou-se o código fonte e a base de perguntas utilizadas no treinamento da rede neural na comunidade de desenvolvedores *online* GitHub (https://github.com/franciscarlospereira/IpTabLearing.git).

5. CONSIDERAÇÕES FINAIS

O estudo objetivou desenvolver um sistema de aprendizagem de *firewall Iptables* orientado por redes neurais. Utilizou-se o algoritmo de aprendizagem supervisionado *naive bayes* para que fosse possível ao usuário digitar uma pergunta na língua portuguesa e, internamente, o sistema buscasse as regras de *firewall Iptables* pertinentes à pergunta realizada, bem como a sua respectiva explicação técnica. A alta acurácia apresentada pela rede neural denotou consistência na modelagem computacional do problema proposto.

A interdisciplinaridade aplicada na construção do problema de pesquisa possibilitou que o estudo apresentasse uma abordagem destoante do restante da literatura que utilizou técnicas de inteligência artificial semelhantes às deste trabalho aplicadas ao campo da SI. Buscamos ampliar a aplicabilidade dos algoritmos de inteligência artificial a um artefato de *software* que auxiliasse no ensino e no aprendizado de práticas do cotidiano de profissionais e alunos em SI. A possibilidade de evolução do estudo, aplicação em outros contextos da segurança da informação e pelo seu propósito acadêmico, situa o trabalho na fronteira interdisciplinar entre a segurança da informação, a inteligência artificial e o processo de ensino/aprendizagem. Acreditamos que as técnicas de *machine learning* utilizadas neste estudo podem ser adaptadas a problemas de diferentes naturezas.

Este estudo limitou-se a compreender apenas dois tipos de ataques a redes de computadores (DoS *e brute-force*), sendo estes os mais comuns. Entretanto, há uma ampla gama de ataques na literatura, possibilitando que o sistema possa ser ampliado e novos estudos realizados.

6. AGRADECIMENTOS

Agradecemos ao Departamento de Pesquisa do Centro Universitário do Sul de Minas – UNISMG e ao Programa de Pós-Graduação em Sistemas de Informação e Gestão do Conhecimento da Universidade FUMEC.

7. REFERÊNCIAS

[1] Candiago, L. *Algoritmo de Classificação Naive Bayes*. 2017. Disponível em:<https://goo.gl/2q4FN5>. Acesso em: 01 ago. 2018.

[2] CERT.br**.** *Estatísticas dos Incidentes Reportados ao CERT.br*. 2018. Disponível em: <https://www.cert.br/stats/incidentes/2017-jan-dec/scan-portas.html>. Acesso em: 26 set. 2018.

[3] Corrêa, G. N et al. Uso da mineração de textos na análise exploratória de artigos científicos. *Relatório técnico n. xxx. Instituto de Ciências Matemáticas e de Computação de São Carlos*. 2012. Disponível em: <https://bit.ly/33jCwfi>. Acesso em: 01 jun. 2018.

[4] Gonçalves, E. C. *SQL Magazine.* Rio de Janeiro, n. 105, 2012. Disponível em: <https://goo.gl/AmsXp9>. Acesso em: 30 jul. 2018.

[5] Haykin, S. *Redes Neurais: princípios e prática*. Simon Haylin; Trad. Paulo Martins Engel. 2.ed. - Porto Alegre: Bookman, 2001.

[6] Henke, M. et al. Detecção de Intrusos usando Conjunto de k-NN gerado por Subespaços Aleatórios. *XI Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais*, 2011. Disponível em: <https://bit.ly/2YSSRnA>. Acesso em: 8 jul. 2019.

[7] Hevner, A. R. et al. Design Science in Information Systems Research. *MIS Quarterly*, v. 28, n. 1, p. 75–105, 2004. Disponível em: <http://dblp.uni-trier.de/rec/bibtex/journals/misq/HevnerMPR04>. Acesso em: 8 jul. 2019.

[8] Lima, P. L. S. et al. Classificação de QoS em Conteúdo Multimídia para Rede VPN utilizando Rede Neural Multilayer Perceptron. In: *XIV SEMINF (Semana de Informática) e EIN (Escola de Informática Norte)*. Belém, 2011. Disponível em: <https://bit.ly/2M5jcNr>. Acesso em: 8 jul. 2019.

[9] Martins, C. A. et al *Uma Experiência em Mineração de Textos Utilizando Clustering Probabilísticos Clustering Hierárquico.* Relatório técnico n. 205. Instituto de Ciências Matemáticas e de Computação de São Carlos. 2003. Disponível em: <http://conteudo.icmc.usp.br/CMS/Arquivos/arquivos_enviados/BIBLIOTECA_113_RT_205.pdf>. Acesso em: 8 jul. 2019.

[10] Moura, C. *Aprendizado de Máquina: conceitos e práticas da área que está movendo o mundo*. 2017. Disponível em: <https://goo.gl/znopFk>. Acesso em: 31 set. 2018.

[11] Nascimento, F. Á. et al. *IP-TAB-LEARNING: sistema de aprendizagem de firewall orientado por redes neurais*. 2018, Varginha, MG: Even3, 2018. p. 1–29. Disponível em: <https://www.even3.com.br/Anais/simgeti/111717-IP-TAB-LEARNING--SISTEMA-DE-APRENDIZAGEM-DE-FIREWALL-ORIENTADO-POR-REDES-NEURAIS>.

[12] Nakamura, E. T; Geus, E. *Segurança de redes em ambientes cooperativos*. São Paulo: Novatec, 2007.

[13] Urubatan Neto. *Dominando Linux Firewall Iptables.* Rio de Janeiro: Editora Ciência Moderna Ltda, 2004.

[14] Oliveira, D. J.; Sá, A. A. R.. Redes Neurais Aplicadas à Classificação de Tráfego de Redes de Computadores Utilizando os Protocolos TCP e UDP. *Revista de Sistemas e Computação-RSC*, v. 8, n. 1, 2018. Disponível em: <https://revistas.unifacs.br/index.php/rsc/article/download/5302/3493>. Acesso em: 8 jul. 2019.

[15] Pacheco, A. *Medida de desempenho de classificadores*. 2017. Disponível em:<https://goo.gl/GM8fQF>. Acesso em: 31 set. 2018.

[16] Pelloso, M. et al. Um Sistema Autoadaptável para Predição de Ataques DDoS Fundado na Teoria da Metaestabilidade. In: *Anais.. XXXVI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*. SBC, 2018. Disponível em: <https://portaldeconteudo.sbc.org.br/index.php/sbrc/article/view/2454>. Acesso em: 8 jul. 2019.

[17] Pressman, R., Maxim, B. *Engenharia de Software – Uma Abordagem Profissional*. AMGH; Edição: 8. 2016.

[18] Severino, A. J. *Metodologia do Trabalho Científico*. 24. ed. São Paulo: Cortez, 2016.

[19] Sordi, J. O.; Azevedo, M. C.; Meireles, M. A Pesquisa Design Science no Brasil segundo as Publicações em Administração da Informação. *Revista de Gestão da Tecnologia e Sistemas de Informação*, v. 12, n. 1, p. 165–186, 2015. Disponível em: <http://www.jistem.fea.usp.br/index.php/jistem/article/view/10.4301/S1807-17752015000100009>. Acesso em: 8 jul. 2019.

[20] Souza, E. M. *Metodologias e analíticas qualitativas em pesquisa organizacional: uma abordagem teórico-conceitual*. Vitória: EDUFES, 2014.

[21] Sunil, Ray. *6 passos fáceis para aprender o algoritmo Naive Bayes (com o código em Python)*. 2016. Disponível em: <https://goo.gl/XMjByC>. Acesso em: 30 set. 2018.

[22] Rodrigues, J. *O que é o Processamento de Linguagem Natural?* 2017. Disponível em:<https://goo.gl/GL6Fjf>. Acesso em: 01 ago. 2018.

[23] Borba, Gabriel. *Firewall – Iptables*. Disponível em: <http://gabrielborba.com.br/?page_id=521>. Acesso em: 01 ago. 2018.

[24] Stato, A**.** *Firewall Dinâmico*. 2016. Disponível em: < https://goo.gl/7r692N>. Acesso em: 01 ago. 2018.

[25] BLOG PORTA 80. Restringindo tráfego via *Iptables* para servidores Web. 2018. Disponível em: <https://goo.gl/dc45v4>. Acesso em: 02 ago. 2018.

[26] UBUNTU FORUMS. FTP brute force blocking via *Iptables*. Disponível em: <https://ubuntuforums.org/showthread.php?t=791596>. Acesso em: 01 ago. 2018.

[27] Pearce, Peavita. How to configure *IPTABLES* to block Telnet and FTP – The Visual Guide. 2015. Disponível em: <https://goo.gl/SUYQby>. Acesso em: 01 ago. 2018.

[28] Vasconcellos, P. Restringindo conexão brute force com *Iptables*. 2010. Disponível em: <https://goo.gl/J6CKAL>. Acesso em: 01 ago. 2018.

[29] Queiroz, Rubens de Almeida. Dicas avançadas de segurança para SSH. 2018. Disponível em: <https://goo.gl/4n12od>. Acesso em: 01. ago. 2018.

[30] MNX SOLUTIONS. Rate limiting connections with *Iptables*. 2018. Disponível em: <https://goo.gl/fsGVgD>. Acesso em: 01 ago. 2018.

[31] STACK EXCHANGE. preventing brute force attack on samba server [closed]. 2018. Disponível em: <https://goo.gl/eaMuRF>. Acesso em: 01 ago. 2018.

[32] SCHALLER, Jeff - Using *Iptables* to stop brute force attacks on a Apache https site. 2016. Disponível em: <https://goo.gl/MTEUwy>. Acesso em: 01 ago. 2018.