Um Algoritmo Não Supervisionado e Rápido para Seleção de Características em Classificação de Tráfego *

Martin Andreoni Lopez[1,2], Antonio G. P. Lobato[1],
Diogo Menezes F. Mattos[1,2], Igor D. Alvarenga[1],
Otto Carlos M. B. Duarte[1], Guy Pujolle[2]

[1]GTA / PEE-COPPE / UFRJ – Brasil
[2]LIP6/CNRS (UPMC Sorbonne Universités) – França

{martin, antonio,diogo,alvarenga,otto}@gta.ufrj.br
{guy.pujolle}@lip6.fr

Abstract. *Security applications such as anomaly detection and attack mitigation need real-time monitoring to reduce risk. Information processing time should be as small as possible to enable an effective defense against attacks. In this paper, we present a fast and efficient feature-selection algorithm for network traffic classification based on the correlation between features. For the evaluation of the algorithm, we run the algorithm against a dataset containing more than 16 types of threats, in addition to normal traffic. The presented algorithm chooses an optimized subset of features that improves accuracy by more than 11% with a 100-fold reduction in processing time when compared to traditional feature selection and reduction algorithms.*

Resumo. *Aplicações de segurança como a detecção de anomalias e a mitigação de ataques precisam de monitoramento em tempo real para a diminuição dos riscos. Os tempos para o processamento das informações devem ser os menores possíveis para habilitar elementos de defesa. Este artigo apresenta um algoritmo rápido e eficiente de seleção de características para a classificação de tráfego baseado na correlação entre características. Para a avaliação do algoritmo, é utilizado um conjunto de dados contendo mais de 16 tipos de ameaças, além de tráfego normal. O algoritmo desenvolvido escolhe um subconjunto otimizado de características que melhora a acurácia em mais do 11% com redução de até 100 vezes do tempo de processamento, quando comparado com algoritmos tradicionais de seleção e redução de características.*

1. Introdução

É fundamental entender o tipo, o volume e as características intrínsecas de cada fluxo, a fim de manter a estabilidade, a confiabilidade e a segurança de uma rede. Monitores de redes eficientes permitem ao administrador um melhor entendimento do comportamento da rede [Hu et al. 2015]. O monitoramento da rede possui diversos níveis de complexidade. Desde uma simples coleção de estatísticas de utilização de link até complexas análises de camadas superiores para a realização de sistemas de detecção de intrusão (*Intrusion Detetection System* – IDS), otimização de desempenho ou depuração de protocolos. Os sistemas atuais de monitoramento de rede como `tcpdump`, NetFlow, SNMP, Bro [Paxson 1999], Snort [Roesch 1999], entre outros, são executados em um único servidor e não são capazes de atingir as velocidades de tráfego atuais de forma escalável [Vallentin et al. 2007]. O IDS Bro, por exemplo, evoluiu

*Este trabalho foi realizado com recursos do CNPq, CAPES, FAPERJ e FINEP.

para a utilização em agrupamentos (*clusters*), a fim de atingir 10 Gb/s. O IDS Suricata foi proposto como uma melhoria sobre o IDS Snort, pois utiliza GPU (*Graphical Processing Unit*) para paralelizar a inspeção de pacotes.

No monitoramento de tráfego, dados chegam em tempo real, originados de fontes heterogêneas de informação [Bar et al. 2014], como a captura de pacotes de redes ou os *logs* de sistemas e aplicações. Esse tipo de monitoramento em tempo real por fluxo é caracterizado por uma sequência de eventos infinita, abstraídos em uma estrutura de dados chamada de tuplas, que chegam continuamente [Stonebraker et al. 2005]. Um dos problemas deste tipo de monitoramento é o volume de dados gerados. Mesmo redes de velocidades moderadas podem gerar uma grande quantidade de dados. Monitorar um link *ethernet gigabit*, por exemplo, funcionando a uma taxa de 50% de utilização, gera um *terabyte* de dados em questão de horas.

Uma forma de otimizar os métodos de processamento é através da utilização de algoritmos de aprendizado de máquina. As técnicas de aprendizado de máquina são adequadas para o processamento de grandes massas de dados (*big data*), já que, com maiores amostras para o treinamento, as técnicas tendem a melhorar a sua efetividade [Mayhew et al. 2015]. No entanto, com grandes volumes de dados, esses métodos apresentam maiores atrasos devido ao alto consumo de recursos computacionais. Esse atraso é uma desvantagem para esses métodos, já que com as grandes massas de dados, o processamento deve ser tão rápido quanto possível, a fim de se ter respostas em tempo real. A seleção de características é uma forma de resolver esse problema, já que reduz o número de características analisadas pelos algoritmos de aprendizado de máquina a um subconjunto menor do que o original [Mladenić 2006].

Este artigo apresenta um algoritmo eficiente e rápido de seleção de características para a classificação de tráfego. O algoritmo realiza a seleção de características de modo não supervisionado, ou seja, sem conhecer a classe de cada fluxo *a priori*. Além disso, o algoritmo determina as principais características para fazer uma classificação acurada do tráfego entre as classes normal, negação de serviço e ameaças de varredura de portas. Para a avaliação do algoritmo é utilizado um conjunto de dados[1] criado a partir do monitoramento de tráfego real [Pastana Lobato et al. 2016]. O algoritmo apresentado é comparado com métodos tradicionais, tais como ReliefF [Robnik-Šikonja e Kononenko 2003], Seleção de Características Sequencial (*Sequential Feature Selection* - SFS), a Análise de Componentes Principais (*Principal Component Analysis* - PCA) [Schölkopf et al. 1999] e a Seleção Recursiva de Eliminação por Máquinas de Vetores de Suporte (*Support Vector Machine Recursive Feature Elimination* SVM-RFE) [Guyon et al. 2002]. O algoritmo apresenta melhor acurácia nos métodos de aprendizado de máquinas utilizados, assim como uma melhora no tempo de processamento total.

O restante do artigo está organizado da seguinte forma. A Seção 2 discute os trabalhos relacionados. O conjunto de dados de segurança é apresentado na Seção3. O algoritmo de Seleção de Características é introduzido na Seção 4. Na Seção 5, o algoritmo apresentado é comparado com métodos tradicionais de seleção de características. Por fim, a Seção 6 conclui o trabalho.

2. Trabalhos Relacionados

Existem dois modos de reduzir o número de variáveis a serem analisadas e melhorar o tempo de processamento dos algoritmos de aprendizado de máquinas: redução e projeção de características. O primeiro deles é a seleção de características, no qual um subconjunto das características é selecionado de acordo com a quantidade de informação que cada carac-

[1]O conjunto de dados é disponibilizado através do contato com os autores por *e-mail*.

terística possui. Existem diversas métricas para se medir a quantidade de informação de cada característica, como por exemplo, o ganho de informação ou a correlação entre elas, como no algoritmo apresentado. O segundo modo é a redução de características através da projeção dos dados em um novo espaço vetorial, no qual uma transformação é realizada sobre o conjunto de características original, o que leva o conjunto original para um espaço de dimensão menor, em que as novas características são combinações das originais. No novo espaço, o número de novas características a serem tratadas pelos algoritmos de aprendizado é reduzido através do descarte de componentes menos significativas. Cada característica nesse novo espaço é uma combinação das características originais [Van Der Maaten et al. 2009].

A Análise de Componentes Principais (*Principal Component Analysis* - PCA) é um dos métodos mais populares para a redução não supervisionada de características. A ideia do PCA é projetar as características originais de maneira que as novas características sejam ortogonais, ou seja, independentes. A partir dessa transformação, as características que representam as maiores variâncias dos dados originais são selecionadas. A direção que representa a maior variância projetada é chamada da Componente Principal. Assim, as características geradas são uma combinação linear das componentes principais. Como os componentes são ordenados seguindo uma ordem decrescente de variância, a dimensionalidade do dado original pode ser reduzida, eliminando as componentes com a menor variância. A maior desvantagem das técnicas de projeção de características é a perda de expressividade, já que as características geradas são combinações de outras características e perdem seu significado original.

Uma solução para isso é a utilização de métodos de seleção de características em que um subconjunto das características originais é escolhido. A seleção de características pode ser dividida em três grupos [Mladenić 2006]: *wrapper*, filtragem e embarcados. Nos métodos *wrapper* são usados diferentes classificadores como as Máquinas de Vetores de Suporte (*Support Vector Machine*- SVM), redes neurais, entre outros, para dar uma pontuação a um subconjunto de características. A partir da pontuação, escolhem-se as características que obtiveram a maior pontuação ao serem usadas para classificar os dados. Os métodos de filtragem analisam as propriedades intrínsecas dos dados, também chamadas de metadados, e realizam algum tipo de função para determinar a sua importância. Os métodos embarcados realizam a seleção mediante o uso de classificadores no processo de aprendizado.

Os métodos de filtragem são computacionalmente mais leves e evitam a superespecialização ou *over-fitting*. Esses métodos utilizam alguma heurística para avaliar a relevância da característica dentro do conjunto de dados [Chandrashekar e Sahin 2014]. Um dos métodos mais populares de filtragem é o Relief. Nele, a pontuação é calculada como a diferença de distância da amostra mais próxima de uma mesma classe e a amostra mais próxima de uma classe diferente. Um aspecto negativo desse método é ter de saber as classes do conjunto de dados. O Relief é limitado a duas classes, mas o ReliefF [Robnik-Šikonja e Kononenko 2003] permite a análise de múltiplas classes utilizando a técnica dos k-vizinhos mais próximos.

Os métodos de *Wrapper* utilizam o erro dos classificadores como critério de avaliação da característica. A seleção é considerada como um problema de busca, criando um problema de tipo NP-complexo. Os métodos de *wrapper* apresentam melhores resultados que os métodos de filtragem, porém com alto custo computacional [Ang et al. 2016]. Uma das implementações mais populares do *Wrapper* é o (*Sequential Forward Selection* - SFS). O algoritmo começa com um conjunto vazio S e o conjunto completo de todas as variáveis X. O SFS faz uma pesquisa e gradualmente adiciona variáveis selecionadas por uma função de avaliação que minimiza o erro quadrático médio. Em cada iteração, a variável a ser incluída em S é selecionada entre as restantes em X. A principal desvantagem do SFS é que, ao adicionar uma característica ao

conjunto S, o método não é capaz removê-la se ela tiver menor erro, após a adição de outras.

Os métodos embarcados (*embedded methods*) apresentam um comportamento semelhante aos métodos *Wrapper*, usando um classificador para avaliar a relevância da variável. No entanto, os métodos embarcados realizam a seleção da variável no processo de aprendizado, reduzindo o tempo computacional dos *Wrappers*. Na Seleção Recursiva de Eliminação usando Máquinas de Vetores de Suporte (*Support Vector Machine Recursive Feature Elimination* SVM-RFE) [Guyon et al. 2002], o algoritmo obtém uma pontuação das variáveis baseado no treinamento do SVM com *kernel* linear. A característica com a menor pontuação é removida de acordo com um critério w em modo sequencial de eliminação para trás. O critério w é o valor da decisão do hiperplano no SVM.

O algoritmo apresentado neste artigo é inspirado na Seleção de Características baseada em correlação [Hall 1999] (*Correlation based Feature Selection* - CFS). Hall utiliza a ideia de pontuar as variáveis através da correlação entre as variáveis e a classe de saída. O algoritmo CFS calcula a correlação entre a variável e a classe para obter a importância de cada característica. Assim, o CFS é dependente da informação da classe de saída *a priori*, logo é um algoritmo supervisionado. A dependência da informação da classe de saída a que pertence cada amostra é uma limitação em aplicações como a classificação de trafego de redes, já que não existe um conhecimento *a priori* entre o fluxo e a classe correspondente.

O algoritmo apresentado realiza a seleção de características de modo não supervisionado. O cálculo da correlação entre as variáveis é utilizado para medir a quantidade de informação que cada variável representa em relação às demais variáveis. Dessa forma, o algoritmo apresentado executa com menor tempo de processamento e não necessita conhecer *a priori* a classe de saída. A diferença do CFS, que considera a correlação individual entre variáveis, o algoritmo proposto é independente da classe e considera a soma das correlações.

3. Criação de um Conjunto de Dados de Segurança

Existem poucos conjuntos de dados disponíveis para a avaliação de pesquisas de mecanismos de defesa contra ataques. A pouca disponibilidade desses conjuntos de dados é devido à preocupação com a privacidade e o receio de vazamento de informações confidenciais contidas no corpo dos pacotes [Heidemann e Papdopoulos 2009]. Um dos principais conjuntos de dados disponíveis, o DARPA [Lippmann et al. 2000], é composto por tráfego TCP/IP e dados coletados de sistemas operacionais de uma rede simulada. Já o KDD 99 [Lee et al. 1999] é gerado a partir dos arquivos `tcpdump` do DARPA processados e abstraídos em características. Esses conjuntos de dados contêm ataques simulados e marcados como ataques. A principal crítica, tanto para os dados do DARPA como do KDD, é que o tráfego não corresponde a um cenário de uma rede de computadores real[Tavallaee et al. 2009], já que são resultado de simulações. Além disso, existem dados redundantes que afetam o cálculo dos métodos de classificação dos dados. Ademais, esses conjuntos de dados estão defasados, pois foram simulados há mais de 15 anos [Sommer e Paxson 2010], e muitas aplicações, assim como ataques, sugiram e mudaram durante esse período.

O conjunto de dados utilizado foi criado pelos autores previamente [Pastana Lobato et al. 2016]. Esse conjunto de dados é obtido através da captura de pacotes de tráfego real do laboratório do Grupo de Teleinformática e Automação (GTA) da UFRJ. O tráfego capturado contém comportamento normal e ameaças reais de redes. Depois da captura dos pacotes, os dados são agrupados em uma janela de tempo. O tráfego é armazenado em forma de fluxo, uma sequência de pacotes com o mesmo IP de origem e IP de destino.

Cada fluxo possui 24 características obtidas dos cabeçalhos TCP/IP. A lista completa

Tabela 1: As 24 características obtidas do cabeçalho TCP/IP para cada fluxo.

Número	Característica	Descrição
1	qtd_pkt_tcp	Quantidade de Pacotes TCP
2	qtd_src_port	Quantidade de Pacotes Portas Origem
3	qtd_dst_port	Quantidade de Pacotes Portas Destino
4	qtd_fin_flag	Quantidade de Flags FIN
5	qtd_syn_flag	Quantidade de Flags SYN
6	qtd_psh_flag	Quantidade de Flags PSH
7	qtd_ack_flag	Quantidade de Flags ACK
8	qtd_urg_flag	Quantidade de Flags URG
9	qtd_pkt_udp	Quantidade de Pacotes UDP
10	qtd_pkt_icmp	Quantidade de Pacotes ICMP
11	qtd_pkt_ip	Quantidade de Pacotes IP
12	qtd_tos	Quantidade de Tipos de Serviço IP
13	ttl_m	TTL Médio
14	header_len_m	Tamanho Médio dos cabeçalhos
15	packet_len_m	Tamanho Médio dos Pacotes
16	qtd_do_not_frag	Quantidade de Flags “Do Not Frag”
17	qtd_more_frag	Quantidade de Flags “More Frag”
18	fragment_offset_m	Offset Médio do Fragmento
19	qtd_rst_flag	Quantidade de Flags RST
20	qtd_ece_flag	Quantidade de Flags ECE
21	qtd_cwr_flag	Quantidade de Flags CWR
22	offset_m	Offset Médio
23	qtd_t_icmp	Quantidade de Tipos ICMP
24	qtd_cdg_icmp	Quantidade de Códigos ICMP

das características é mostrada na Tabela 1. Ataques de Negação de Serviço (*Denial of Service* - DoS) e ameaças de varredura de porta (*Probe*) compõem as duas classes de comportamento anormal. O conjunto possui sete tipos de DoS e nove tipos de ameaças de varredura.

3.1. Descrição das Ameaças no Conjunto de Dados

No **ataque LAND** (*Local Area Network DoS*) o atacante manda um pacote TCP com etiqueta SYN com o endereço IP falsificado, contendo o endereço e porta da vítima como o endereço e porta de origem e destino. Assim, o sistema responde com um pacote SYN-ACK para ele mesmo, criando uma conexão vazia que permanece aberta até o temporizador estourar. Logo, o sistema entra em um laço enviando respostas para ele mesmo até eventualmente colapsar. ***Nestea*** é um ataque que afeta aos servidores Linux e o ataque ***teardrop*** é o semelhante para servidores Windows. O ataque consiste em explorar uma vulnerabilidade no protocolo TCP/IP ao enviar pacotes fragmentados para a vítima. Quando a soma do deslocamento e tamanho de um pacote fragmentado diferem daquele do próximo pacote fragmentado, os pacotes se sobrepõem e a vítima tenta montar o pacote entrando em uma negação de serviço.

Os ataques por inundação (*flooding*) enviam uma quantidade anormal de pacotes sem esperar respostas. Logo, a vítima, ao tentar processar todos os pacotes, não consegue atender a todas as requisições a tempo e provoca a negação de serviço. No conjunto de dados usado, existem a **inundação por SYN**, a **inundação por ICMP**, e a **inundação por UDP**. Na inundação por SYN são enviados múltiplos pacotes com endereços de origem falsificados para manter co-

nexões abertas e assim estourar os recursos da vítima. Na inundação por ICMP, também são enviados múltiplos pacotes ICMP sem esperar a resposta da vítima. A inundação por UDP tem uma filosofia diferente das anteriores, pois o UDP não é orientado à conexão. Nesse ataque, são enviados pacotes com o mesmo endereço, mas com diferentes portas, com o objetivo de saturar a largura de banda.

O ataque de ***Smurf***, de forma semelhante a inundação por ICMP, utiliza um pacote *Echo Request* (ICMP) com o endereço de difusão (*broadcast*) da rede como endereço de destino, mas o endereço de origem é falsificado com o endereço da vítima. As respostas a esse pacote são para o endereço da vítima, assim todas as estações da rede respondem para o endereço da vítima, tornando-a inutilizável e até podendo causar falhas no funcionamento da rede. O *Smurf* é considerado um ataque de negação de serviço distribuído.

A varredura de portas consiste em verificar quais são os serviços ativos em um servidor, isto é, se há uma porta TCP escutando requisições e um processo executando para tratar as requisições que chegam a essa porta. No entanto, a varredura de portas é considerada uma ameaça já que metade dos ataques é precedida por uma varredura de portas para identificar as vulnerabilidades do sistema que podem ser exploradas. Na varredura de portas, o atacante gera pacotes e monitora as respostas para determinar se um serviço está vulnerável ou não. **A varredura SYN** é a mais conhecida pela simplicidade e velocidade, e também é conhecida como *half-open scanning*, já que nunca abre uma conexão completa. Um pacote SYN é enviado e, se o sistema está escutando conexões TCP na porta para qual o pacote é destinado, a vítima responderá com um pacote SYN-ACK e, logo, o atacante enviará um RST, fechando a conexão antes que o *handshake* seja estabelecido. Se o serviço não está aberto, a vítima responderá com um pacote RST. Se não há resposta, após várias tentativas, a porta é marcada como filtrada, isto é, há um *firewall* que impede a comunicação entre o atacante e a vítima por tal porta.

A **varredura de conexão TCP**, também conhecido como *full-open scanning*, tem um comportamento semelhante ao anterior, mas neste caso a comunicação é feita totalmente se a porta estiver aberta, ou seja, o atacante envia o SYN, espera pelo SYN-ACK e, por fim, envia o ACK para estabelecer a conexão. Se a porta estiver fechada, o atacante recebe um pacote RST/ACK.

Os ataques de TCP **FIN, XMAS, ACK, NULL** e **Maimon** são semelhantes, já que a ausência de resposta significa que a porta da vítima está aberta ou que existe um *firewall* no caminho entre o atacante e a vítima. No **FIN** é enviado um pacote TCP com o bit FIN ativo em um. Dependendo do sistema operacional da vitima, se a porta da vítima está aberta não terá resposta e se estiver fechada a vítima responderá com um pacote RST/ACK. O **TCP XMAS** simula o comportamento normal de um cliente, já que envia pacotes TCP com os bits URG, PUSH e FIN ativos em um mesmo pacote. De novo, se a porta estiver aberta não haverá resposta e se estiver fechada terá como resposta um pacote RST/ACK. No **NULL**, ao contrário do FIN, é enviado um pacote com todos os bits em zero, sem resposta quando a porta está aberta e com RST/ACK quanto está fechada. A varredura de **TCP Maimon**, nome recebido devido a seu criador, envia um pacote TCP com bit FIN ativo e também com ACK ativo.

A varredura **TCP por Janela** utiliza diferentes tamanhos de janela a uma resposta RST. Dependendo do sistema operacional, as portas abertas usam um tamanho de janela positivo enquanto as fechadas têm uma janela de tamanho zero. Alternativamente aos protocolos de transporte convencionais TCP, UDP e ICMP, é criada a varredura de **SCTP INIT**. Essa técnica tem um comportamento semelhante à varredura TCP SYN, já que cria uma "meia" conexão. Um *chunk* INIT é enviado pelo atacante. Um *chunk* INIT-ACK indica que a porta está aberta,

enquanto um *chunk* ABORT é indicativo de fechada. Se nenhuma resposta for recebida após várias retransmissões, a porta é marcada como filtrada.

3.2. Análise do Conjunto de Dados

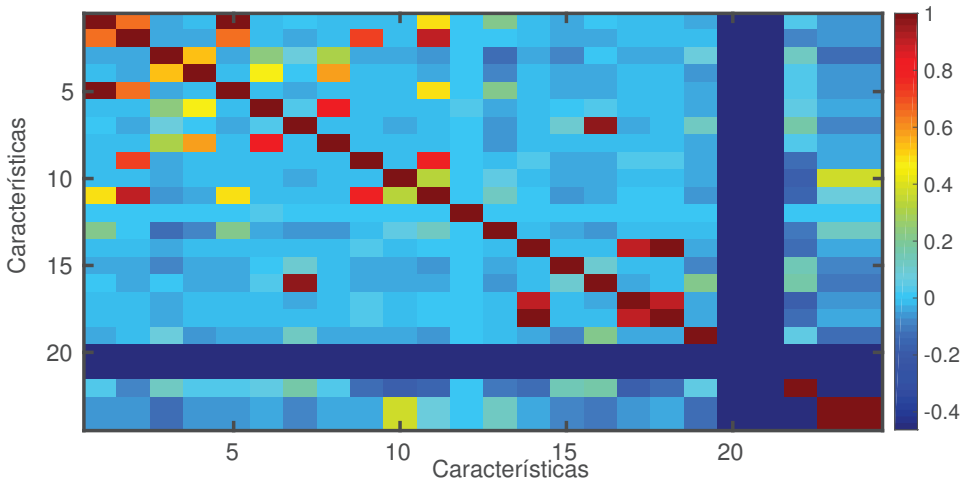

Figura 1: Matriz de correlação das 24 características disponíveis no conjunto de dados. Os pontos escuros em vermelho indicam a máxima correlação e os pontos em azul indicam a correlação mínima.

A Figura 1 mostra a correlação entre as 24 características que compõem o conjunto de dados usado. A matriz calcula a correlação a partir da distância de Pearson (*Pearson Product-Moment Correlation Coefficient* - PPMCC). O produto de Pearson é uma medida da dependência linear entre duas variáveis X e Y, sendo +1 quando a correlação entre as variáveis é linear, 0 quando não tem correlação e -1 quando a correlação entre as variáveis é inversamente linear. Na figura, a correlação linear é representada com vermelho escuro, a correlação nula é representada pela cor branca e o azul representa a correlação inversa. As características 21 e 22 indicam "Quantidade de ECE Flags" e "Quantidade de CWR Flags", respectivamente. As etiquetas de *Explicit Congestion Notification* (ECN) e de *Congestion Window Reduced* (CWR) são usadas para alertar ao remitente de possíveis situações de congestionamento na rede, a fim de evitar a perda de pacotes e retransmissões. Na matriz de correlação, essas duas características estão representadas em azul escuro indicando um baixo nível de correlação. No caso do conjunto de dados criados, as variáveis 21 e 22 não agregam nenhuma informação aos dados, pois são sempre nulas. Isso se deve ao fato que na rede em que os dados foram obtidos não havia congestão ou sobrecarga. Por outro lado, as características 23, "Quantidade de Tipos ICMP", e 24, "Quantidade de Códigos ICMP", são representadas em vermelho. Assim, essas duas variáveis estão altamente correlacionadas entre si, pois o tipo do ICMP está relacionado como o código ICMP.

O conjunto de dados conta com mais de 95 GB em pacotes capturados, resultando em 214.200 fluxos compostos de tráfego malicioso e normal. A Figura 2 mostra a relação entre as classes no conjunto de dados. A classe normal possui aproximadamente 70% dos dados do conjunto, com 106.955 amostras etiquetadas como tráfego normal, a classe de Negação de Serviço representa 10% do conjunto de dados, com 16.741 amostras, e a classe de varredura de portas possui 20% das amostras, ou seja, 30.491 fluxos.

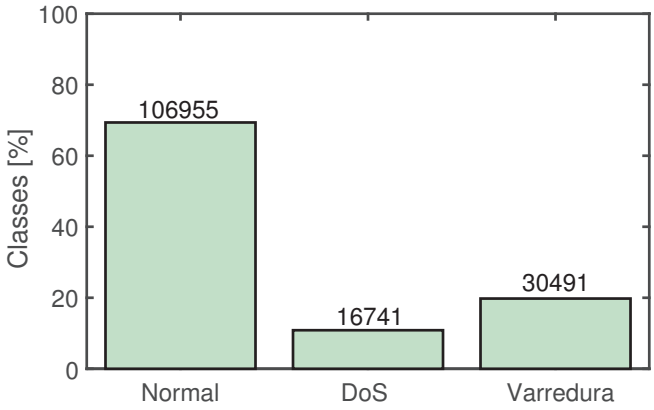

Figura 2: Distribuição das classes no conjunto de dados. A classe principal é a Normal com quase o 70% dos dados, a classe de Negação de Serviço (DoS) tem o 10% e a classe de varredura corresponde ao 20% do conjunto de dados.

4. A Seleção Baseada na Correlação de Características

O método de seleção de características estudado nesse artigo utiliza a correlação entre as variáveis através do coeficiente de Pearson. O valor do coeficiente de Pearson é $-1 \leq \rho \leq 1$, em que 1 significa que duas variáveis estão diretamente correlacionadas através de uma relação linear e -1, o caso inverso, correlação inversa ou anticorrelação.

O coeficiente de Pearson ρ é calculado em termos de média μ e o desvio padrão σ como:

$$\rho(A,B) = \frac{1}{N-1}\sum_{i=1}^{N}\left(\frac{A_i - \mu_A}{\sigma_A}\right)\left(\frac{B_i - \mu_B}{\sigma_B}\right), \tag{1}$$

ou como função da co-variância σ como:

$$\rho(A,B) = \frac{cov(A,B)}{\sigma_A \sigma_B}. \tag{2}$$

Primeiramente, é preciso obter a matriz de correlação baseada na Equação 2. A matriz de correlação apresenta o cálculo da covariância entre cada par de variáveis. O método de seleção de características estudado neste trabalho atribui um peso w a cada variável. O peso w é uma medida da importância de cada variável que representa uma característica em relação às demais variáveis aleatórias. O peso w para cada característica é dado pela soma do valor absoluto da correlação entre a característica considerada e as demais características do conjunto de dados. A soma é em valor absoluto, pois o coeficiente de Pearson ρ pode apresentar valores negativos, porém, de maneira qualitativa, uma variável que apresenta uma forte correlação inversa com outra aporta tanta informação para a representação dos dados quanto uma variável que tenha o mesmo valor absolto de correlação direta. O valor absoluto da correlação indica o quanto uma variável é linearmente dependente de outra. Correlação próxima a 1 representa uma forte dependência linear. Correlação próxima a 0 representa independência linear entre as duas variáveis. Além disso, o peso w é uma indicação da quantidade de informação que uma variável tem independentemente das outras. O peso w tem valores entre $0 \leq w \leq N$, onde N é a quantidade de características no conjunto de dados, e 0 representa uma variável totalmente independente das outras. Quanto mais alto é o valor do peso w, maior é a dependência linear com outras variáveis e menor é a quantidade de informação que a variável aporta à representação do conjunto de dados.

Algoritmo 1: Seleção de Características Baseada na Correlação.

Input : X: Matriz de Dados
Output: **r**: Vetor de Caraterísticas pontoadas, **w**: Vetor de pesos

```
1 A = Corr(X) /* Calculo da Matriz de Correlação          */
2 for 0 ≤ i < len(A) do
3 |   w_i = 0
4 |   for 0 ≤ j < len(A_i) do
5 |   |   k_i = abs(A_ij) /* Calculo do Valor Absoluto    */
6 |   |   w_i += k_i /* Calculo do Peso                   */
7 |   end
8 end
9 r = sort(w, reverse = True) /* Ordenamento descendente de w */
```

5. Caso de Uso de Seleção de Características: Classificação de Tráfego

Para a classificação automática de tráfego são utilizados os algoritmos mais utilizados de aprendizado de máquina [Buczak e Guven 2015]. Em todos os métodos, o treinamento é realizado com 70% do conjunto de dados e a avaliação com os 30% restante. Durante a fase de treino foi realizada a validação cruzada para evitar a superespecialização ou *overfitting*. Na validação cruzada, o conjunto de dados de treino é dividido em diversas partes e algumas delas não são utilizadas para estimação dos parâmetros. Elas são usadas posteriormente para verificar se o modelo é geral o suficiente para se adaptar a novos dados, evitando a superespecialização.

Na árvore de decisão, as folhas representam a classe final, e os ramos representam as condições baseadas no valor de uma das características de entrada. Durante o processo de treinamento o algoritmo de C4.5 determina a estrutura do tipo da árvore para a classificação. A implementação em tempo real da árvore de decisão consiste em regras "*if-then-else*" que são geradas na forma da árvore previamente calculadas.

As redes neurais são baseadas no cérebro humano no qual cada neurônio realiza uma pequena parte do processamento, transferindo o resultado para o próximo neurônio. Nas redes neurais artificiais, a saída representa o grau de pertinência para cada classe. Os vetores de pesos Θ são calculados durante o treinamento e determinam o peso de cada conexão dos neurônios. No treinamento, há erros causados por cada parâmetro, que são minimizados através do algoritmo de propagação posterior.

Para determinar a qual classe pertence uma amostra, cada camada de rede neural calcula as seguintes equações:

$$z_{(i+1)} = \Theta_{(i)} a_{(i)} \qquad (3) \qquad a_{(i+1)} = g(z_{(i+1)}) \qquad (4) \qquad g(z) = \frac{1}{1+e^{-z}} \qquad (5)$$

onde $a_{(i)}$ é um vetor que determina a saída das camadas i, para a camada $i+1$, e $a_{(i+1)}$ é a saída da camada $i+1$. A função $g(z)$ é a função *Sigmoid* que é a função de ativação de cada neurônio, importante para a classificação. Para altos valores de z, $g(z)$ retorna um e para valores baixos de z, $g(z)$ retorna zero. Logo, a camada de saída retorna o grau de pertinência para cada classe entre zero e um, classificando a amostra como a de maior valor.

As Máquinas de Vetores de Suporte (*Support Vector Machine* - SVM) são um classificador binário baseado no conceito de hiperplanos de separação entre classes que define os limiares da decisão. O algoritmo de SVM classifica mediante a construção de um hiperplano

em um espaço multidimensional que divide as diferentes classes. Um algoritmo iterativo minimiza a função de erro, encontrando o melhor hiperplano de separação. Uma função de *kernel* define o hiperplano, sendo linear ou não linear. Assim, o SVM encontra a margem máxima de separação entre duas classes. A classificação em tempo real é realizada para uma das classes: normal ou não normal; DoS ou não DoS; e varredura ou não varredura. Uma vez que o SVM calcula a saída, a classe é escolhida segundo a maior pontuação. A pontuação de uma amostra x é a distância de x para os limites de decisão que vão desde $-\infty$ a $+\infty$. A pontuação do classificador:

$$f(x) = \sum_{j=1}^{n} \alpha_j y_j G(x_j, x) + b, \tag{6}$$

onde $(\alpha_1, ..., \alpha_n.b)$ são os parâmetros estimado do SVM, e $G(x_j, x)$ é o *kernel* utilizado. Neste trabalho o *kernel* usado foi um *kernel* linear, $G(x_j, x) = x_j^{'}x$, o qual apresenta um bom desempenho com uma mínima quantidade de parâmetros de entrada.

5.1. Resultados

O algoritmo de Seleção de Características desenvolvido foi comparado com a Análise de Componentes Principais (PCA), o algoritmo de ReliefF, a eliminação por Seleção Sequencial para a Frente (*Sequential Forward Selection* - SFS), e a Seleção de Características Recursiva usando SVM (*Support Vector Machine Recursive Feature Elimination* SVM-RFE). Todos os métodos são comparados considerando a seleção de quatro e seis características. A mérito de justiça na comparação, todos os métodos de seleção e redução de características foram executados com os classificadores apresentados anteriormente: árvore de decisão com o algoritmo de CART com 4096 folhas; redes neurais com uma camada escondida e dez neurônios; e máquinas de vetores de suporte (SVM) com *kernel* linear. Todos os experimentos são realizados com validação cruzada 10-*fold*.

A Figura 3 mostra os autovalores associados ao conjunto de dados proposto. É possível ver que com quatro e seis variáveis se obtém entre o 80% e 90% do total da variância. Dessa forma, todas as demais componentes representam entre 10% e 20% da variância e, portanto, podem ser eliminadas sem prejuízos para a representação dos dados.

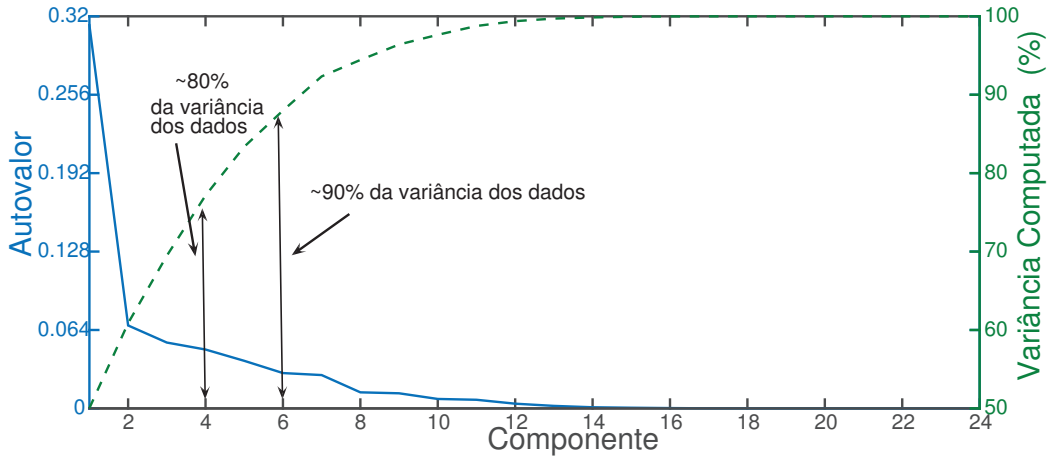

Figura 3: Autovalor para cada uma das 24 características do fluxo. O autovalor associado a cada uma das características transformadas é proporcional à variância dos dados. Entre a quarta e a sexta componente mais importantes, representam de 80% a 90% do total da variância dos dados.

A Figura 4 mostra o desempenho comparando a acurácia dos três classificadores. No primeiro grupo, são mostrados os resultados da árvore de decisão, na qual o algoritmo desenvolvido possui o melhor desempenho com seis características com 97.4% de acurácia. Na sequência, há resultado do PCA com quatro e seis características em 96% e 97.2% respectivamente. O *Sequential Forward Selection* (SFS) apresenta o mesmo resultado com quatro e com seis características com acurácia a 95.5%. O algoritmo de ReliefF também apresenta o mesmo resultado para quatro e seis características, 91.2%. Finalmente o pior resultado é apresentado pelo SVM-RFE com quatro e seis características, 80.2% e 85.8%.

No segundo classificador, a rede neural, o melhor resultado é apresentado pelo PCA com seis características com 97.6% de acurácia. ReliefF apresenta o mesmo resultado com ambas em 90.2%. O algoritmo de seleção por correlação proposto mostra um resultado de 83.9% e 85.0% com quatro e seis características. Por outro lado, o SFS apresenta o pior resultado de todos os classificadores com 78.4% com quatro características e 79.2% com seis. Um resultado inesperado foi o do SVM-RFE, já que com quatro características apresenta um desempenho baixo de 73.6% de acurácia, sendo um dos piores em todos os classificadores, no entanto, com seis características é o segundo melhor para as redes neurais com 90.1%. Vale ressaltar que as características consideradas pelo PCA são variáveis transformadas que podem ser combinações lineares de um maior número características originais, enquanto nos demais métodos são escolhidos subconjuntos de características que limitam o conjunto de dados tratado pelos classificadores.

Nas Máquinas de Vetores de Suporte (SVM) o PCA apresenta um comportamento semelhante ao apresentado nas redes neurais. Com seis características tem a melhor acurácia de todos os classificadores com 98.3%, mas só 87.8% para quatro características. ReliefF novamente apresenta um resultado semelhante para ambas características em 91.4%. O algoritmo proposto tem 84% de acurácia com quatro e 85% com seis características. SFS apresenta a mesma acurácia para as duas amostras 79.5%. Finalmente, o resultado mais baixo para esse classificador é o SVM-RFE com 73.6% em ambos os casos.

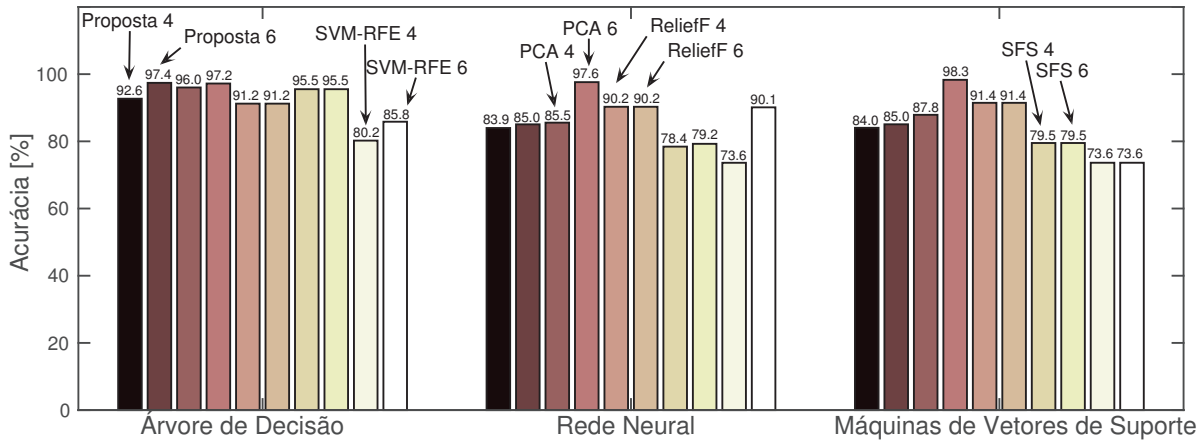

Figura 4: Comparação da acurácia em três classificadores dos métodos de Seleção e redução de características. Nosso algoritmo proposto, PCA Lineal, ReliefF, Sequence Forward Selection (SFS) e SVM-RFE na árvore de decisão, na rede neural e na máquina de vetores de suporte.

Na Tabela 2 é apresentada a seleção feita por cada um dos métodos. Foi excluído o método do PCA já que a saída apresentada é composta por novas características sintéticas e

Tabela 2: Comparação da seleção de características de cada um dos métodos.

	Proposta	ReliefF	SFS	SVM-REF
Número de Característica	12,15,19, 13,10,7	2,11,9, 19,22,3	15,11,8, 5,20,21	14,12,13, 24,23,22
Nome da Característica	qtd_tos, packet_len_m, qtd_rst_flag, ttl_m, qtd_pkt_icmp, qtd_ack_flag	qtd_src_port, qtd_pkt_ip, qtd_pkt_udp, qtd_rst_flag offset_m, qtd_dst_port	packet_len_m, qtd_pkt_ip, qtd_urg_flags, qtd_syn_flag, qtd_ece_flag, qtd_cwr_flag	header_len_m qtd_tos ttl_m qtd_cdg_icmp qtd_t_icmp offset_m

não corresponde com uma comparação dos outros métodos. Todos os métodos mostram as seis características mais importantes. É possível ver que nenhum método escolhe o mesmo grupo de características. No entanto, ReliefF e SFS selecionaram como a segunda melhor característica a número 11, "*qtd_pkt_ip*". Um resultado bem interessante é que o SFS selecionou as características 20 e 21, "*qtd_ece_flag* e *qtd_cwr_flag*". Na matriz de correlação apresentada na Seção3 foi discutida que essas duas características não adicionam nenhuma informação por que são variáveis vazias. Contudo, depois de uma análise foi reparado que a característica mais importante é a número 15 "*pkt_len_m*". Nesse conjunto de dados, a medida do tamanho de pacotes é fundamental para a classificação dos ataques.

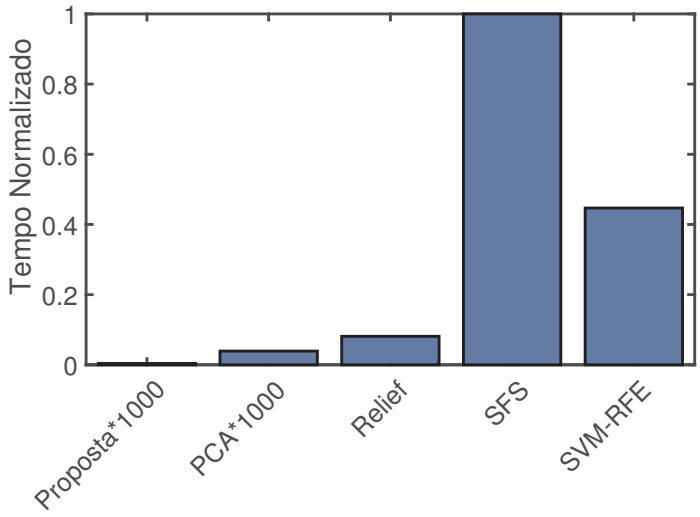

Figura 5: Avaliação de desempenho em relação ao tempo de processamento dos algoritmos de seleção de características. O algoritmo proposto é até 3 vezes mais rápido que o PCA.

Na Figura 5 é apresentado o tempo de processamento de cada um dos algoritmos durante a seleção de características. Os valores são normalizados dividindo o tempo de processamento de cada método pelo valor máximo, obtido pelo SFS. Tanto o algoritmo desenvolvido como o PCA foram multiplicados por 1000 por motivos de escala. O SFS mostra o pior desempenho, porque apresenta a maior complexidade, com tempo médio de processamento de 83.480 segundos. O SFS é um algoritmo que realiza múltiplas iterações a fim de minimizar o erro quadrático médio. Consequentemente, todas as interações incrementam o tempo de processamento. O algoritmo de seleção de características desenvolvido nesse trabalho mostra o menor

tempo de processamento, aproximadamente 0.72 segundos em média, seguido do PCA, com 2.04 segundos em média. Os dois métodos apresentam um bom despenho no cálculo de tempo de processamento, já que ambos utilizam a multiplicação de matrizes para obter o conjunto de características. A multiplicação de matrizes é uma função computacional simples já que pode ser paralelizada. As medidas foram obtidas e um computador com processador Intel Xeon com frequência de relógio de 2.6 GHz e 256 GB de memória RAM.

6. Conclusão

Esse artigo apresentou um algoritmo de seleção de características para a classificação de tráfego de redes. O algoritmo calcula a correlação das variáveis de modo não supervisionado. A fim de realizar uma avaliação, os resultados obtidos a partir do algoritmo apresentado de seleção de características foram comparados com os obtidos através métodos populares de seleção e redução de características. Os métodos de seleção comparados são o ReliefF, como método de filtragem, a Seleção Sequencial de Características (*Sequential Feature Selection* - SFS), como método *wrapper*, a Seleção Recursiva de Eliminação usando Máquinas de Vetores de Suporte (*Support Vector Machine Recursive Feature Elimination* - SVM-RFE), como método embarcado, e finalmente a Análise de Componentes Principais (*Principal Component Analises* - PCA), como método de projeção de características. Para a avaliação e comparação dos métodos de seleção, foi utilizado um conjunto de dados de segurança que apresenta mais 16 tipos diferentes de ataques assim como tráfego normal. Nesse conjunto de dados foram aplicados três algoritmos de classificação como a árvore de decisão, redes neurais e as máquinas de vetores de suporte. O algoritmo desenvolvido de filtragem apresentou uma acurácia de 97.4% com 6 características selecionadas no algoritmo de árvore de decisão. Além disso, o algoritmo apresentado reduz o tempo de processamento em mais de 100 vezes quando comparado com o método de pior desempenho.

Referências

Ang, J. C., Mirzal, A., Haron, H. e Hamed, H. N. A. (2016). Supervised, Unsupervised, and Semi-Supervised Feature Selection: A Review on Gene Selection. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 13(5):971–989.

Bar, A., Finamore, A., Casas, P., Golab, L. e Mellia, M. (2014). Large-scale network traffic monitoring with DBStream, a system for rolling big data analysis. Em *2014 IEEE International Conference on Big Data (Big Data)*, páginas 165–170. IEEE.

Buczak, A. e Guven, E. (2015). A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. *IEEE Communications Surveys Tutorials*, (99):1–26.

Chandrashekar, G. e Sahin, F. (2014). A survey on feature selection methods. *Computers & Electrical Engineering*, 40(1):16–28.

Guyon, I., Weston, J., Barnhill, S. e Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine learning*, 46(1-3):389–422.

Hall, M. A. (1999). *Correlation-based Feature Selection for Machine Learning*. PhD thesis, The University of Waikato.

Heidemann, J. e Papdopoulos, C. (2009). Uses and challenges for network datasets. Em *Conference For Homeland Security, 2009. CATCH'09. Cybersecurity Applications & Technology*, páginas 73–82. IEEE.

Hu, P., Li, H., Fu, H., Cansever, D. e Mohapatra, P. (2015). Dynamic defense strategy against advanced persistent threat with insiders. Em *IEEE Conference on Computer Communications (INFOCOM)*, páginas 747–755.

Lee, W., Stolfo, S. J. e Mok, K. W. (1999). Mining in a data-flow environment: Experience in network intrusion detection. Em *Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining*, páginas 114–124. ACM.

Lippmann, R. P., Fried, D. J., Graf, I., Haines, J. W., Kendall, K. R., McClung, D., Weber, D., Webster, S. E., Wyschogrod, D., Cunningham, R. K. e others (2000). Evaluating intrusion detection systems: The 1998 DARPA off-line intrusion detection evaluation. Em *Proceedings of DARPA Information Survivability Conference and Exposition. DISCEX'00.*, volume 2, páginas 12–26. IEEE.

Mayhew, M., Atighetchi, M., Adler, A. e Greenstadt, R. (2015). Use of machine learning in big data analytics for insider threat detection. Em *IEEE Military Communications Conference, MILCOM*, páginas 915–922.

Mladenić, D. (2006). Feature Selection for Dimensionality Reduction. Em Saunders, C., Grobelnik, M., Gunn, S. e Shawe-Taylor, J., editors, *Subspace, Latent Structure and Feature Selection (SLSFS): Statistical and Optimization Perspectives Workshop.*, páginas 84–102. Springer Berlin Heidelberg, Bohinj, Slovenia.

Pastana Lobato, A. G., Andreoni Lopez, M. e Duarte, O. C. M. B. (2016). Um Sistema Acurado de Detecção de Ameaças em Tempo Real por Processamento de Fluxos. Em *SBRC'2016*, páginas 572–585, Salvador, Bahia.

Paxson, V. (1999). Bro: a System for Detecting Network Intruders in Real-Time. *Computer Networks*, 31(23-24):2435–2463.

Robnik-Šikonja, M. e Kononenko, I. (2003). Theoretical and Empirical Analysis of ReliefF and RReliefF. *Machine Learning*, 53(1/2):23–69.

Roesch, M. (1999). Snort-Lightweight Intrusion Detection for Networks. Em *Proceedings of the 13th USENIX conference on System administration*, páginas 229–238. USENIX Association.

Schölkopf, B., Smola, A. J. e Müller, K.-R. (1999). Kernel principal component analysis. Em *Advances in kernel methods*, páginas 327–352. MIT Press.

Sommer, R. e Paxson, V. (2010). Outside the closed world: On using machine learning for network intrusion detection. Em *IEEE Symposium on Security and Privacy (SP)*, páginas 305–316. IEEE.

Stonebraker, M., Çetintemel, U. e Zdonik, S. (2005). The 8 requirements of real-time stream processing. *ACM SIGMOD Record*, 34(4):42–47.

Tavallaee, M., Bagheri, E., Lu, W. e Ghorbani, A.-A. (2009). A detailed analysis of the KDD CUP 99 data set. Em *Proceedings of the Second IEEE Symposium on Computational Intelligence for Security and Defence Applications*. IEEE.

Vallentin, M., Sommer, R., Lee, J., Leres, C., Paxson, V. e Tierney, B. (2007). The NIDS Cluster: Scalable, Stateful Network Intrusion Detection on Commodity Hardware. Em *Recent Advances in Intrusion Detection*, páginas 107–126. Springer Berlin Heidelberg, Berlin, Heidelberg.

Van Der Maaten, L., Postma, E. e den Herik, J. (2009). Dimensionality reduction: a comparative. *Journal of Machine Learning Research*, 10:66–71.