UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA
CURSO DE ENGENHARIA DE COMPUTAÇÃO

RODOLFO VEBBER BISOL

Coleta e Análise de Características de Fluxo para Classificação de Tráfego em Redes Definidas por Software

Monografia apresentada como requisito parcial para a obtenção do grau de Bacharel em Engenharia da Computação

Orientador: Prof. Dr. Alberto Egon Schaeffer Filho

Porto Alegre
2015

“I may not have gone where I intended to go,
but I think I have ended up where I needed to be.”
— Douglas Adams

AGRADECIMENTOS

Agradeço a Universidade Federal do Rio Grande do Sul por ter proporcionado toda a estrutura de ensino necessária para a minha formação acadêmica; aos professores e funcionários que fazem parte desta renomada instituição, especialmente ao professor Alberto E. Schaeffer Filho pela orientação durante a realização deste trabalho. Também agradeço aos professores Cormac Sreenan e Ken Brown da University College Cork pela oportunidade que me encaminhou para a área de redes de computadores.

Agradecimento especial aos meus pais Vilson e Margareth e à minha irmã Rafaela, que apoiaram todas as minhas decisões, desde a escolha do curso e universidade até este momento. Também agradeço à Daiane que esteve sempre ao meu lado durante este último ano de graduação.

Aos meus amigos e colegas da UFRGS, principalmente ao Lucas e Vinícius, e a todos da turma de 2010/2, que ajudaram a manter minha saúde mental e social perante todos os desafios destes cinco anos e meio.

Agradeço também aos bolsistas, mestrandos e doutorandos do Grupo de Redes do Instituto de Informática em especial a todos que comigo dividiram o laboratório 212 que me ajudaram com dicas a respeito de aspectos técnicos.

E a todas pessoas que direta ou indiretamente fizeram parte desta jornada.

RESUMO

Uma classificação acurada de fluxos de tráfego é uma tarefa muito importante no âmbito de gerência de redes de computadores, podendo ser aplicada com diversos objetivos como, por exemplo, detecção de fluxos de tráfego maliciosos. Estes mecanismos, apesar de já serem muito sofisticados, ainda precisam ser aprimorados pois certos ataques podem se camuflar entre os tráfegos benignos que normalmente existem na rede. Neste contexto, este trabalho oferece uma proposta de estender os contadores oferecidos pelo protocolo OpenFlow com o objetivo de criar um conjunto de características avançadas que descrevam os perfis de tráfego existentes na rede. Adicionalmente, por meio da da aplicação de técnicas de seleção de características, determinar o melhor subconjunto a fim de aprimorar os mecanismos de classificação de fluxos de tráfego. Além disso, é proposto um módulo externo que pode ser adicionado aos controladores de rede, o qual possui uma arquitetura modular que permite a implementação de diversos algoritmos de classificação de fluxos de tráfego e seleção de características. Os experimentos comprovam que existem combinações de características que aprimoram a precisão da classificação de diferentes tipos de fluxo de tráfegos.

Palavras-chave: Seleção de características. Classificação de fluxos de tráfego. Redes definidas por software.

Gathering and Analysis of Flow Features for Traffic Classification in Software Defined Networks

ABSTRACT

An accurate flow classification is a very important task in network management and can be applied with many objectives such as malicious traffic detection. These mechanisms although sophisticated still need to be refined because certain attacks can hide behind benign flows normally existing in the network. In this context, this work proposes to extend the native counters offered by the OpenFlow protocol aiming to create an advanced flow feature set that represents the traffic profiles existing in the network. And through the application of feature selection techniques determine the best subset to enhance traffic classification mechanisms. Also, is proposed an external module that can be added to the network controllers and has a modular architecture that allows to implement different algorithms to flow classification and feature selection. The experiments prove that certain combinations of flow features improve flow classification accuracy in different scenarios.

Keywords: Feature selection, Traffic classification, Software defined networking.

LISTA DE ABREVIATURAS E SIGLAS

API	Application Program Interface
DDoS	Distributed Denial of Service
FCBF	Fast Correlation-Based Filter
GA	Genetic Algorithm
IANA	Internet Assigned Numbers Authority
ML	Machine Learning
ONF	Open Networking Foundation
PCA	Principal Component Analysis
QoS	Quality of Service
SBS	Sequential Backward Selection
SDN	Software Defined Networking
SVM	Support Vector Machine
TTL	Time-to-Live

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Redes de computadores e principalmente a Internet se tornaram essenciais da vida moderna, sendo utilizadas por governos, empresas e instituições de ensino. Por isso, é muito importante identificar comportamentos anômalos e ataques de usuários maliciosos, a fim de manter seu adequado funcionamento.

A Classificação de fluxos de tráfego pode ser utilizada para diversos objetivos, como detecção de ataques, realocação de recursos de redes e modelagem de perfil de usuários (NGUYEN; ARMITAGE, 2008). Para se obter uma classificação acurada, é necessário identificar diversas características, tais como duração do fluxo e número de pacotes transmitidos que são capazes de descrever fielmente os diferentes perfis de tráfego que possam existir dentro da rede, como por exemplo, fluxos de tráfego HTTP e comunicação P2P. A operação de coleta de características é dificultada por limitações existentes na organização de redes IP tradicionais, tais como: *(i)* o acesso à informação sobre o estado da rede é heterogêneo para dispositivos de diferentes fabricantes; *(ii)* a grande complexidade da adição de novas funcionalidades aos switches ou roteadores de rede, já que cada equipamento deve ser modificado individualmente; e *(iii)* a lógica distribuída em cada elemento de rede pode alterar um fluxo durante seu trânsito pela rede, por exemplo, fazendo alterações na prioridade dos pacotes ou em cabeçalhos TCP.

A adoção de Redes Definidas por Software (Software Defined Networking - SDN), neste contexto, elimina as limitações mencionadas e facilita o estudo e implementação de novas técnicas que tornam a classificação de fluxos de tráfego mais acurada (ONF, 2012). O fato de SDN ser caracterizada pela separação dos planos de controle e encaminhamento faz com que a lógica esteja centralizada em controladores SDN, como o NOX (GUDE et al., 2008) e HyperFlow (TOOTOONCHIAN; GANJALI, 2010). Assim, alterações na lógica da rede são realizadas apenas nos controladores. Além disso, a coleta de dados sobre os fluxos de tráfego existentes na rede se torna uma tarefa mais simples, sendo realizada através dos controladores que possuem visão de grandes porções da rede, ou até mesmo global, quando utilizados em conjunto.

Dentro deste cenário, o objetivo deste trabalho é criar um mecanismo capaz de coletar o maior número possível de informações sobre os fluxos de tráfego oferecidas pelo protocolo OpenFlow. Com estas informações, derivar um conjunto de informações formado por características de fluxo avançadas que representem fielmente o comportamento de cada perfil de tráfego existente. Adicionalmente, conseguir identificar quais subconjuntos de características são mais eficientes para aprimorar mecanismos de classificação de tráfego, por exemplo, sistemas de detecção e mitigação de ataques. Desta forma, as contribuições deste trabalho são: *(i)*

identificação e análise de um conjunto avançado de características de fluxo de tráfego baseadas em contadores nativos do protocolo OpenFlow; *(ii)* comparação de três técnicas de seleção de características, Sequential Backward Selection (SBS), Principal Component Analysis (PCA) e Genetic Algorithm (GA), para obter o subconjunto de características que torna a classificação de fluxo mais eficaz em algoritmos baseados em aprendizado supervisionado e não supervisionado; e *(iii)* definição de um módulo a ser executado externamente aos controladores de rede para identificação, coleta e análise de características de fluxos de tráfego.

Este trabalho está estruturado na seguinte forma: o capítulo 2 apresenta uma visão geral dos assuntos abordados, tais como Redes Definidas por Software, Seleção de Características e Classificação de Fluxos de Tráfego; o capítulo 3 apresenta como a solução é modelada, descrevendo arquitetura e procedimentos para a coleta a extensão de características; o Capítulo 4 descreve detalhes sobre implementação da solução, incluindo a parametrização dos algoritmos utilizados, ferramentas utilizadas e resultados obtidos; o Capítulo 5 conclui este trabalho apresentando as considerações finais.

2 REVISÃO BIBLIOGRÁFICA

Neste Capítulo, é apresentada uma contextualização dos conceitos de Redes Definidas por Software (Software-Defined Networking - SDN), Classificação de Fluxos de Tráfego e Seleção de Características. A Seção 2.1 apresenta a arquitetura SDN e o Protocolo OpenFlow. A Seção 2.2 apresenta diversos métodos de classificação de fluxos. Por fim, a Seção 2.3 apresenta as técnicas de seleção de características mais utilizadas.

2.1 Redes definidas por Software (Software-Defined Networking)

As Redes Definidas por Software possuem duas características que as definem: a separação do Plano de Controle do Plano de Encaminhamento e a consolidação do plano de controle, de forma que apenas um controlador, em software, controle diversos dispositivos do plano de dados (FEAMSTER; REXFORD; ZEGURA, 2013).

2.1.1 Visão Geral

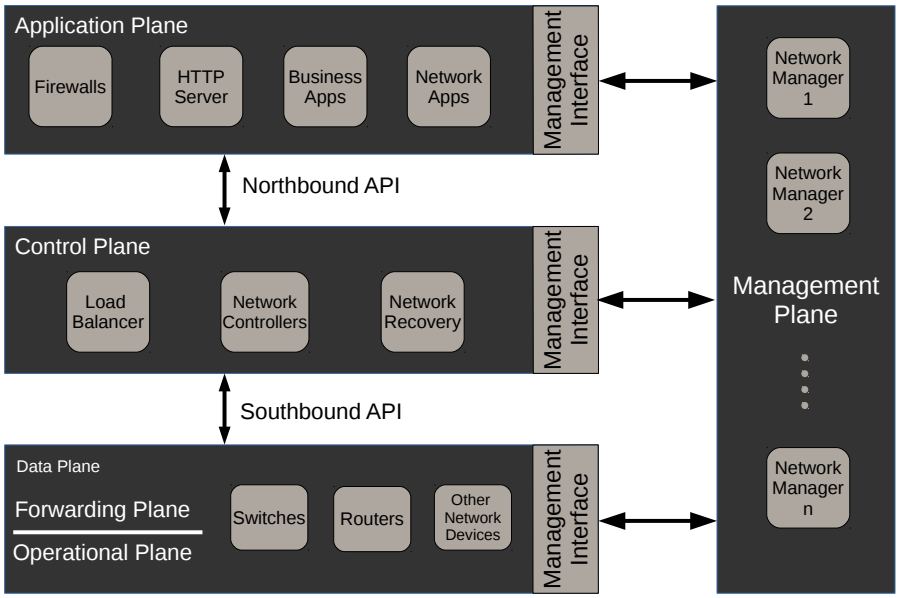
A criação de Redes Definidas por Software foi motivada pela necessidade de simplificar a gerência de redes de computadores e, consequentemente, facilitar o processo de inovação e evolução. Esta necessidade vem das limitações que as redes de computadores tradicionais possuem, entre elas: *(i)* a configuração complexa, pois elas possuem diversos dispositivos, como roteadores e switches, e *midleboxes*, como firewalls, que precisam ser configurados individualmente; *(ii)* o controle descentralizado dos dispositivos de rede, que torna difícil a implementação de novas políticas, protocolos e outras alterações no núcleo da rede; e *(iii)* o atual crescimento da quantidade de dados trafegando pelas redes, o qual desafia a escalabilidade destas (ONF, 2012) (NUNES et al., 2014).

As facilidades e simplificações oferecidas por SDN são frutos da separação dos planos de controle e encaminhamento e da centralização da inteligência em um controlador. Assim, modificações no comportamento ou nas políticas da rede passam a ser feitas apenas no controlador SDN e não em cada dispositivo. Desta forma, administradores de rede e pesquisadores podem implementar e testar novas políticas e protocolos de maneira mais rápida, eficiente e segura em um ambiente próximo à realidade.

A Figura 2.1 apresenta a arquitetura SDN composta por cinco planos (HALEPLIDIS

et al., 2015): Plano de Aplicação (*Application Plane*); Plano de Controle (*Control Plane*); Plano de Gerenciamento (*Management Plane*); Plano de Encaminhamento (*Forwarding Plane*); e Plano Operacional (*Operational Plane*). Estes dois últimos podem ser encontrados em algumas literaturas como apenas um plano chamado de Plano de Dados (*Data Plane*).

Figura 2.1: Arquitetura de Redes definidas por Software.



Fonte: Próprio autor.

Cada um dos cinco planos possui responsabilidades específicas. O Plano de Aplicação é onde as aplicações e serviços que definem o comportamento da rede se encontram. O Plano de Controle é responsável por tomar as decisões de como os pacotes serão encaminhados pelos dispositivos de rede e enviar essas decisões para o Plano de Encaminhamento. As operações de monitoramento, configuração e manutenção dos dispositivos de rede são realizadas pelo Plano de Gerenciamento. O Plano de Encaminhamento é responsável por manipular os pacotes de acordo com as instruções recebidas do Plano de Controle - por exemplo: encaminhamento, descarte e modificações nos pacotes. E finalmente, o Plano Operacional gerencia o estado operacional dos dispositivos de rede, tais como, se o dispositivo está ativo ou não, quais portas estão ativas, entre outros aspectos.

Referente a comunicação entre os planos, são definidas APIs específicas. A comunicação entre os planos de Controle e de Aplicação é feita através da interface *Northbound API*. A *Southbound API* realiza a comunicação entre os planos de Controle e de Encaminhamento. Por último, existem as *Management interfaces*, que permitem a comunicação do Plano de Gerenciamento com os demais planos (WICKBOLDT et al., 2015).

2.1.2 Protocolo OpenFlow

O Protocolo OpenFlow é o protocolo mais utilizado e difundido em redes SDN. Esse protocolo foi criado com o objetivo de padronizar a comunicação entre os planos de Controle e de Encaminhamento, facilitando a implantação de SDN. O OpenFlow foi proposto em 2008 por pesquisadores da Stanford University (MCKEOWN et al., 2008), e neste artigo, McKeown et al. apresentam um mecanismo para ser inserido nos switches comerciais. A principal finalidade deste mecanismo é permitir que pesquisadores na área de redes tenham um ambiente para rodar experimentos dentro das universidades sem interferir com o tráfego normalmente encontrado na infraestrutura da instituição.

Explorando o fato de que a maioria dos switches Ethernet existentes no mercado utilizam tabelas de fluxo para realizar suas funções funções - as quais mesmo diferentes entre variados vendedores, possuem uma série de características em comum - foi definido o OpenFlow Switch. Neste switch, o objetivo é que os fluxos criados para pesquisa possam circular separadamente dos fluxos normais dentro da universidade, sem que ocorram interferências entre eles, por meio da configuração das tabelas de fluxo.

O OpenFlow Switch é composto por três componentes básicos (MCKEOWN et al., 2008): (i) uma tabela de fluxos que diz qual ação a ser tomada para cada fluxo; (ii) o protocolo OpenFlow, que define um padrão de comunicação entre um controlador SDN e os switches; e (iii) um canal seguro de comunicação entre o controlador e o switch.

A primeira especificação do OpenFlow por McKeown *et al.* define as ações que cada switch deve ser capaz executar para os pacotes de fluxos presentes nas tabelas de encaminhamento. Estas ações são: *(i)* encaminhar os pacotes para uma porta específica; *(ii)* enviar o pacote para um controlador pelo canal seguro, para que o controlador possa decidir qual ação deve ser tomada para os todos os pacotes pertencente à aquele fluxo, sendo que esta ação é geralmente realizada apenas para o primeiro pacote de um novo fluxo; e *(iii)* descartar o pacote. Outras ações foram adicionadas em novas versões do protocolo, entre essas: *(i)* criação e atribuição de grupos que determinam o comportamento de pacotes; *(ii)* determinação de filas específicas para cada pacote na porta de saída do switch; e *(iii)* modificação do TTL de pacotes (ONF, 2014).

O OpenFlow é dito como a mais nova e inovadora ideia em redes de computadores e permite diversas simplificações (LIMONCELLI, 2012):

- Permite a programação e otimização da rede por regras de roteamento baseadas em fluxos. Por exemplo, um fluxo com um grande volume pode ser direcionado para o caminho mais barato, enquanto um fluxo sensível à latência é direcionado por um caminho mais

confiável e de maior velocidade.

- Pelo fato de todo o processamento ser realizado no controlador, o crescimento da rede implica em aumentar a capacidade computacional apenas do controlador, podendo deixar os dispositivos de rede sem alterações.
- Pelo fato do controlador designar todas as operações que os dispositivos devem executar para cada pacote, cada dispositivo pode se comportar de maneira diferente para cada fluxo podendo agir como firewall para alguns, balanceadores de carga para outros, ou até mesmo reescrevendo fluxos.

Outro aspecto importante que torna o protocolo OpenFlow recomendado para uma série de aplicações diferentes é a tabela de contadores definida na sua especificação (ONF, 2014). Essa tabela foi criada com o objetivo de auxiliar a implementação de soluções de QoS. Os contadores presentes nas tabelas do OpenFlow são atualizados cada vez que um novo pacote chega ao switch e são divididos em conjuntos, por exemplo: *(i)* contadores sobre tabelas de fluxo, como contador de entradas; *(ii)* contadores sobre portas, como contadores de quantidade de pacotes descartados e colisões; *(iii)* contadores sobre fluxos, como contadores de bytes e pacotes.

2.2 Classificação de Fluxos de Tráfego

Classificação de fluxos de tráfego é o processo de determinar a qual classe de tráfego cada fluxo pertence. Com isso, é possível definir comportamentos diferentes para cada classe existente e, consequentemente, para cada fluxo oferecendo um maior controle aos administradores de redes de computadores.

2.2.1 Visão Geral

Um classificador de fluxos de tráfego é um processo automatizado que captura pacotes de diferentes fluxos e determina a qual classe de tráfego esses fluxos pertencem. Essa ação é realizada através da análise de dados sobre os pacotes, por exemplo, os endereços IP de origem e de destino (BLAKE et al., 1998). A classificação de fluxos de tráfego é utilizada para identificar os perfis de tráfego existentes na rede. Classificação de fluxos de tráfego é importante para auxiliar a administração de redes de computadores, sendo utilizada para monitoramento, filtro de conteúdo, avaliação de qualidade de serviço (QoS), entre outras aplicações.

Os primeiros esforços para classificação de fluxos de tráfego consistiam em identificar as portas TCP/UDP de origem e destino de cada fluxo e, assim, classificar o fluxo de acordo com as portas padrões definidas pelo IANA. Essa técnica é conhecida como *Port Level Analysis* e possui uma série de limitações, por exemplo, algumas aplicações não utilizam as portas padrões ou sequer possuem portas padrões registradas. Para eliminar estas limitações, passou-se a usar todas informações contidas nos pacotes e não apenas os números das portas, técnica conhecida como *Deep Packet Inspection* (NGUYEN; ARMITAGE, 2008).

Porém, a partir de 2004, com o rápido crescimento do tamanho e da complexidade das redes de computadores, técnicas de classificação de fluxos de tráfego como *Port Level Analysis* e *Deep Packet Inspection* caíram em desuso, por apresentarem baixo desempenho e não escalarem juntamente com as redes. Com isso, muitos pesquisadores começaram a empregar algoritmos de aprendizado de máquina (Machine Learning - ML) para a classificação de fluxos de tráfego (HU; SHEN, 2012). Esses algoritmos são caracterizados por extrair características de uma amostra de fluxo de tráfego - como por exemplo tamanho médio de pacotes e duração do fluxo - e, de acordo com essas, determinar a que tipo de fluxo a amostra pertence.

Algoritmos baseados em ML já eram muito utilizados em diversas áreas, como para realização de diagnósticos médicos e previsão do tempo, e passaram a ser utilizados em redes de computadores a partir de 1990. A classificação de fluxos de tráfego por ML pode ser divido em dois grupos: por aprendizado supervisionado e não supervisionado.

O aprendizado supervisionado é caracterizado por possuir duas fases distintas: a primeira fase, onde uma estrutura, por exemplo uma rede neural, é treinada com um conjunto de amostras pré-classificadas criando um modelo de classificação; e a segunda fase, que consiste em testar o modelo criado, quando novas amostras, também já pré-classificadas, são entregues ao modelo para avaliar a acurácia do modelo. Exemplos de algoritmos que utilizam aprendizado supervisionado são os baseados em Support Vector Machine, criados por Vladimir Vapnik (VAPNIK, 1982).

O aprendizado não supervisionado difere por não haver uma pré-classificação das amostras. A criação desse método de classificação foi motivada pelo crescimento da quantidade de dados a serem estudados, o que torna a pré-classificação das amostras um processo extremamente custoso. Métodos que utilizam aprendizado não supervisionado são amplamente aplicados em processamento de imagem e estudo de genomas (JAIN, 2010). Esse processo é caracterizado por agrupar as amostras estatisticamente em clusters, baseando-se nas similaridades e diferenças entre as características de cada amostra. Por este motivo, algoritmos que utilizam aprendizado não supervisionado também são conhecidos como algoritmos de clusterização. Um

exemplo de classificação por clusterização é o K-means (MACQUEEN, 1967).

2.2.2 Métricas

Métricas de comparação são utilizadas para comparar a eficácia ou qualidade de diferentes classificadores de tráfego de maneira imparcial e não ambígua e devem ser diferentes para cada caso específico. Por exemplo, as métricas utilizadas para comparar classificadores baseados em aprendizado supervisionado são diferentes das utilizadas para aqueles baseados em clusterização.

As métricas mais comuns para medir o desempenho de algoritmos baseados em aprendizado supervisionado tomam como premissa que já se sabe a priori à qual classe pertence cada amostra. Desta forma, é possível relacionar diretamente o resultado do classificador com a realidade. As principais métricas são: *recall* e *precision*. *Recall* é definido como a porcentagem de fluxos de uma classe corretamente classificados, e *precision* é a porcentagem de fluxos da classe X dentre os classificados como sendo da classe X. Porém, alguns autores também utilizam *accuracy* (ou acurácia) como métrica. *Accuracy* representa a porcentagem de fluxos corretamente classificados perante o total de fluxos (NGUYEN; ARMITAGE, 2008).

Como algoritmos de clusterização não utilizam amostras pré-classificadas, não é possível usar as métricas citadas acima. As métricas para esse grupo de algoritmos devem avaliar as similaridades entre os elementos pertencentes a um cluster, a densidade dos clusters e outros aspectos relacionados aos clusters. Existem três índices que podem ser utilizados para validação de clusters, são estes: interno, externo e relativo (XU; WUNSCH D., 2005). Um algoritmo muito utilizado de validação de clusters é o método de Silhouettes (ROUSSEEUW, 1987), o qual faz a relação existente de cada elemento com outros elementos pertencentes ao mesmo cluster e também com os elementos externos ao cluster.

2.3 Seleção de Características

A utilização de um grande conjunto de características de fluxo para a classificação de fluxos de tráfego possui uma série de desvantagens, por exemplo: características irrelevantes podem adicionar ruído, dificultando a classificação; e o processamento de um grande conjunto de características, possivelmente correlacionadas, gera desperdício de recursos computacionais. Por esses motivos, existe uma etapa no processo de classificação baseado em características de

fluxo chamada de Seleção de Características (*Feature Selection*). Essa etapa consiste na seleção das características ou atributos mais relevantes para um determinado problema e na eliminação das características menos relevantes. Este processo pode trazer diversas vantagens, tais como: melhor entendimento e visualização dos dados; redução do tempo de treinamento de algoritmos de ML; e maior precisão na classificação de fluxos (GUYON; ELISSEEFF, 2003).

Existem diversas definições para o que significa uma característica ser relevante ou não para uma determinada aplicação. O principal motivo de não existir uma definição única de o que é relevante vem da pergunta: "relevante para o que?". Para tentar resolver este problema, Blum e Langley (BLUM; LANGLEY, 1997) definem cinco noções de relevância:

1. **Relevante para a classe**: uma característica x_i é relevante para uma classe c se existem duas amostras A e B que se diferem apenas pela característica x_i e $c(A) \neq c(B)$.
2. **Fortemente relevante para a amostra**: uma característica x_i é fortemente relevante para um conjunto de amostras S se existem duas amostras A e B pertencentes à S que se diferem apenas pela característica x_i e possuem classes diferentes.
3. **Fracamente relevante para a amostra**: uma característica x_i é fracamente relevante para um conjunto de amostras S se é possível remover um subconjunto de características que torne x_i fortemente relevante.
4. **Relevante como uma medida de complexidade**: dado um conjunto de amostras S e um conjunto de classes C, seja $r(S, C)$ o número de características relevantes de acordo com o Item 1 para uma classe em C tal que, fora de todos subconjuntos cujo desempenho em S é menor, tem o menor número de características. Em outras palavras, é o conjunto com o menor número de características que possui o melhor desempenho em S para uma classe em C.
5. **Relevância incremental**: dado um conjunto de amostras S e um algoritmo de ML L, e um conjunto de características A, uma característica x_i é relevante incrementalmente para L se o desempenho de L usando um subconjunto $x_i \cup A$ é maior que o desempenho usando apenas o conjunto de características A.

Por não ser uma tarefa trivial, existem diversas técnicas de seleção de características que podem ser divididas em três grupos (SAEYS; INZA; LARRAñAGA, 2007):

- **Filtros:** analisam as propriedades intrínsecas dos dados, como o comportamento correlacionado entre características. Essas técnicas ordenam as características de acordo com a importância de cada uma e eliminam as características de menor importância ou correlacionadas. Filtros são independentes de algoritmos de classificação ou indução e podem

ser usados juntamente com outras técnicas de seleção de características. Exemplos de filtros são: Principal Component Analysis (JOLLIFFE, 1986) e Fast Correlation-Based Filter (YU; LIU, 2003).

- ***Wrappers***: possuem por característica o uso de métodos de indução como uma sub-rotina. Essas técnicas realizam uma busca pelo melhor subconjunto dentro do espaço de todos os subconjuntos possíveis usando como heurística de busca o desempenho estimado para cada subconjunto. *Wrappers* são muito utilizados em problemas de *pattern recognition* e comumente adotam algoritmos de ML. Exemplos de wrappers são o OBLIVION (LANGLEY; SAGE, 1994) e Algoritmos Genéticos Híbridos (OH; LEE; MOON, 2004).
- ***Embedded***: são técnicas construídas baseadas em um determinado classificador e específicos para um determinado algoritmo de ML. Em sua maioria, utilizam operações lógicas para adicionar ou eliminar características baseando-se em uma hipótese. Essas técnicas são em geral construídas para aplicações específicas e consomem menos recursos computacionais que técnicas baseadas em *Wrappers*.

3 MODELO DE SISTEMA PARA COLETA E ANÁLISE DE CARACTERÍSTICAS DE TRÁFEGO EM SDN

Este capítulo descreve o funcionamento da solução desenvolvida neste trabalho. Na Seção 3.1 é descrita a arquitetura do sistema; a seção 3.2 apresenta como é realizada a coleta de informações da rede, a criação e extensão de características de fluxo; e a seção 3.3 descreve os algoritmos utilizados para a seleção de características e classificação de fluxos de tráfego.

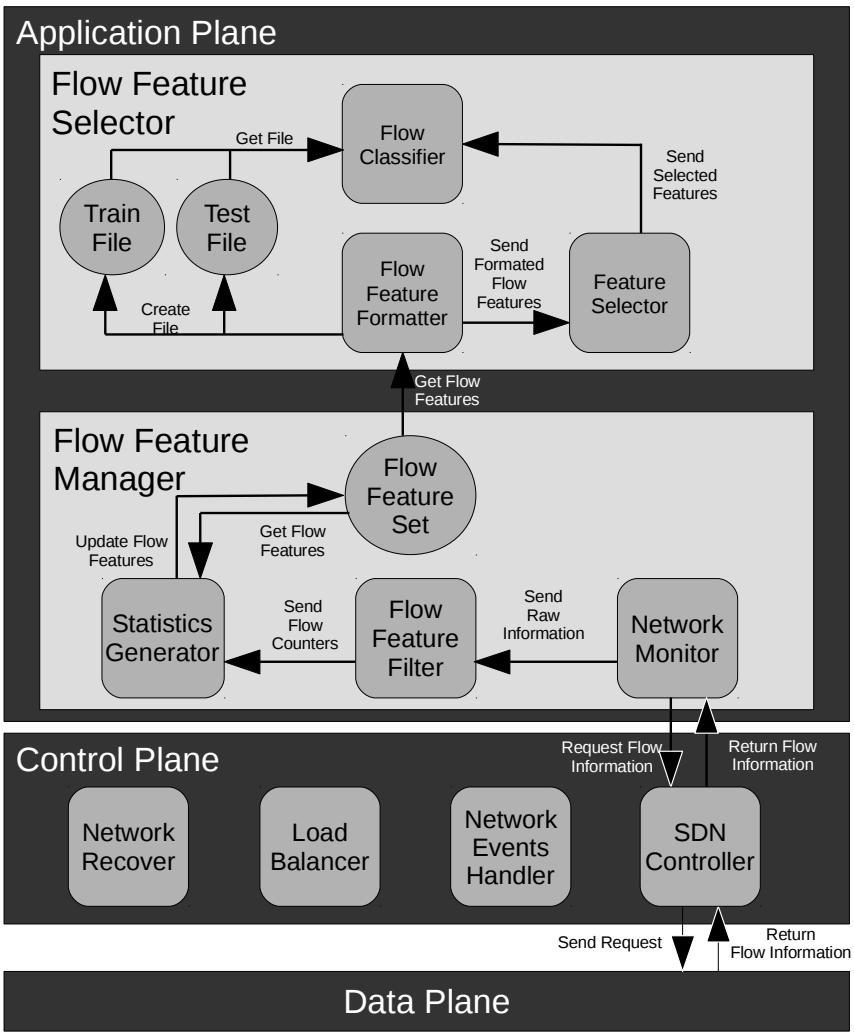
3.1 Arquitetura

A arquitetura da solução está representada na Figura 3.1 e consiste em dois componentes: (i) *Flow Feature Manager*, responsável pela coleta de informações da rede e criação de um conjunto de características avançadas que possam representar fielmente cada fluxo existente; e (ii) *Flow Feature Selector*, responsável por definir o melhor subconjunto de características para classificação de fluxos de tráfego. A seguir, os dois componentes e seus módulos serão descritos em maiores detalhes:

1. **Flow Feature Manager**: este componente é formado pelos módulos: Network Monitor Module, responsável por coletar informações da rede; Flow Feature Filter Module, responsável por filtrar as informações recebidas, selecionando somente as informações relevantes ao sistema; e Statistics Generator Module, responsável por criar o conjunto de características avançadas com as informações recebidas do Flow Feature Filter Module. O conjunto de características criado é representado pela estrutura de dados Flow Feature Set.
2. **Flow Feature Selector**: este componente é formado pelos módulos: Flow Feature Formatter Module, responsável por reorganizar o conjunto de características em um formato padrão e criar os arquivos de treinamento e teste; Feature Selector Module, responsável por selecionar as características de fluxo mais importantes e criar subconjuntos formados por estas características; e Flow Classifier Module, responsável por avaliar a qualidade dos subconjuntos criados através da classificação de fluxos de tráfego.

É importante ressaltar que esta é uma arquitetura modular na qual qualquer um dos módulos descritos pode ser alterado livremente sem que sejam necessárias grandes modificações

Figura 3.1: Arquitetura desenvolvida para coleta e extensão de características de fluxos de tráfego.



Fonte: Próprio autor.

nos demais módulos. Por exemplo, o Feature Selector Module pode ser alterado para suportar outros algoritmos além dos três que são implementados neste trabalho (Seção 3.3) sem que precise alterar os módulos Flow Feature Formatter e Flow Classifier. Também não requer alterações no controlador e pode ser acoplada a qualquer controlador disponível.

3.2 Coleta e Extensão de Característica de Fluxo

O sistema funciona da seguinte forma: primeiro, o Network Monitor envia uma requisição por informações sobre os fluxos existentes na rede ao controlador. O controlador por sua vez reúne as informações presentes nas tabelas de encaminhamento dos switches e as envia para o Network Monitor. Este repassa as informações para o Flow Feature Filter que seleciona

apenas as informações úteis: valor dos contadores de bytes e pacotes que são utilizados para a criação do conjunto de características. Além desses contadores, o Flow Feature Filter também seleciona os endereços IP e portas TCP/UDP de origem e destino e o protocolo da camada de transporte, os quais serão utilizados para identificação dos fluxos. O Statistics Generator recebe os identificadores de fluxo e os dois contadores e estende estes contadores em características de fluxo mais representativas, como tamanho dos pacotes e tempo entre a chegada de dois pacotes consecutivos (*packet inter-arrival time*). Este módulo também é responsável por criar e manter atualizada uma estrutura de dados, o Flow Feature Set, que reúne todas as características de cada fluxo de tráfego identificado na rede.

As informações coletadas da rede, utilizadas para a criação das características de fluxos de tráfego, são dois contadores nativos presentes na tabela de contadores do protocolo OpenFlow, e são: Byte Counter e Packet Counter. Para estender os contadores nativos do OpenFlow, o sistema utiliza outros dois valores criados internamente: contador de amostras, que salva a quantidade de vezes em que um determinado fluxo foi amostrado na rede, e tempo entre requisições enviadas ao controlador pelo Network Monitor.

A definição do tempo entre requisições depende de um trade-off: quanto mais curto for esse tempo mais precisas serão as características criadas, porém requisições muito frequentes geram um aumento do custo computacional necessário para calcular as características. No entanto, se esse tempo for mais longo, perde-se granularidade de informação sobre os fluxos de tráfego. Outros parâmetros que devem ser considerados para a definição deste valor são os perfis de tráfego que se pretende identificar na rede, uma vez que tempo entre requisições muito longo não pode ser usado em ambientes que possuem muitos *bursts* de tráfego.

Com estas quatro informações são criadas quatro características básicas: bytes por segundo ($B_{/s}$); pacotes por segundo ($P_{/s}$); comprimento dos pacotes (P_l); e *packet inter-arrival time* (P_{iat}). Essas quatro características são estimativas dos valores reais, uma vez que para se obter os valores reais de cada uma delas seria necessário coletar informações em um nível de pacote, o que não é oferecido nativamente pelo OpenFlow e demandaria uma maior capacidade computacional por parte da aplicação.

As quatro características básicas são obtidas pelas equações a seguir. Nestas, B_c representa o contador de bytes, P_c representa o contador de pacotes e T representa o tempo entre duas requisições.

$$B_{/s} = \frac{B_c}{T} \qquad (3.1) \qquad\qquad P_{/s} = \frac{P_c}{T} \qquad (3.2)$$

$$P_l = \frac{B_c}{P_c} \quad (3.3) \qquad P_{iat} = \frac{T}{P_c} \quad (3.4)$$

Estas quatro características isoladamente não oferecem informações suficientes sobre o perfil ou o estado dos fluxos de tráfego. Com o objetivo de obter um maior entendimento sobre o comportamento dos fluxos de tráfego, estas quatro características são expandidas em três grupos de características: (i) características escalares, as quais consistem nos máximos e mínimos das características básicas, duração do fluxo, e o tamanho em bytes e pacotes do fluxo; (ii) características estatísticas, que consistem em médias, variâncias, primeiro quartil e terceiro quartil; e (iii) características complexas, que são os componentes da Transformada de Fourier para as amostras do *packet inter-arrival time*. As características escalares descrevem, de maneira mais completa do que simples contadores, o estado dos fluxos de tráfego em um determinado tempo, características estatísticas oferecem um entendimento mais preciso sobre o comportamento dos fluxos de tráfego ao longo do tempo e a Transformada de Fourier descreve o comportamento no domínio de frequências.

3.2.1 Características Escalares

As Características Escalares representam o estado dos fluxos de tráfego em um determinado tempo. O máximo e mínimo de cada uma das quatro características básicas são calculados no momento em que cada nova amostra é capturada na rede, comparando-as com os valores anteriormente armazenados. O tamanho total dos fluxos é representado de duas maneiras: tamanho em bytes, que representa o total de bytes pertencentes ao fluxo; e tamanho em pacotes, que representa quantos pacotes foram enviados durante a duração do fluxo. Ambos são armazenados como contadores e são incrementados a cada nova amostra.

A quinta característica é a duração do fluxo. A duração do fluxo é uma estimativas pois os contadores *Duration* e *DurationNanoSeconds* da Meter Table do OpenFlow representam o tempo em que a regra deve ser mantida na tabela de encaminhamento do switch e não o tempo em que o fluxo está ativo na rede (ONF, 2014). A duração dos fluxos é estimada pelo produto do tempo entre requisições T e o contador de amostras n.

$$flowDuration = T * n \quad (3.5)$$

3.2.2 Características Estatísticas

Para evitar salvar em memória todas as amostras para as quatro características básicas, a média e a variância são calculadas utilizando os seguintes parâmetros: a nova amostra (θ), a média (μ_o) ou variância (σ_o^2) atual e o contador de amostras n. A média (μ_n) e variância (σ_n^2) atualizadas são obtidas por meio das Equações 3.6 e 3.7, respectivamente.

$$\mu_n = \frac{n * \mu_o + \theta}{n+1} \tag{3.6}$$

$$\sigma_n^2 = \frac{n}{n+1} * (\sigma_o^2 + \frac{(\theta - \mu_n) * (\theta - \mu_o)}{n}) \tag{3.7}$$

Os quartis completam o entendimento de como as amostras estão distribuídas, porém necessitam de todas as amostras para que possam ser calculados. Os quartis são calculados apenas para o tamanho e inter-arrival time dos pacotes, devido ao recurso extra de memória necessário e devido ao fato de que os mesmos valores para bytes e pacotes por segundo não oferecem o mesmo grau de entendimento sobre os fluxos de tráfego. Por exemplo, variações no estado da rede podem influenciar nas velocidades de transmissão, mas não nos tamanhos dos pacotes, que são definidos apenas pela aplicação que os gera.

3.2.3 Características Complexas

As características complexas são os dez primeiros componentes da Transformada Discreta de Fourier (equação 3.8). A Transformada de Fourier representa o comportamento da função em um domínio de frequência, onde cada componente é um par complexo, no qual o valor real representa o quanto a frequência (ou amostra) é comum e o valor complexo representa a fase da frequência.

$$X_k = \sum_{n=0}^{N-1} x_n . e^{-2\pi i k n / N},\ k \,\epsilon\, \mathbb{Z} \tag{3.8}$$

Os dez primeiros componentes da Transformada de Fourier também são avaliadas como características relevantes para a classificação de fluxos de tráfego nos trabalhos: (MOORE; ZUEV, 2005) e (AULD; MOORE; GULL, 2007).

A tabela 3.1 apresenta as trinta e três características de fluxo criadas usando os contadores nativos do OpenFlow.

Tabela 3.1: Conjunto de características avançadas criado utilizando contadores nativos do protocolo OpenFlow.

Características Escalares	**Características Estatísticas**	**Características Complexas**
Bytes per Second Maximum Value	Bytes per Second Mean	Fourier Transform 1st Component
Bytes per Second Minimum Value	Bytes per Second Variance	Fourier Transform 2nd Component
Packets per Second Maximum Value	Packets per Second Mean	Fourier Transform 3rd Component
Packets per Second Minimum Value	Packets per Second Variance	Fourier Transform 4th Component
Packet Length Maximum Value	Packet Length Mean	Fourier Transform 5th Component
Packet Length Minimum Value	Packet Length Variance	Fourier Transform 6th Component
Packet Inter-arrival Time Maximum Value	Packet Inter-arrival Time Mean	Fourier Transform 7th Component
Packet Inter-arrival Time Minimum Value	Packet Inter-arrival Time Variance	Fourier Transform 8th Component
Flow Size in Bytes	Packet Length 1st Quartile	Fourier Transform 9th Component
Flow Size in Packets	Packet Length 3rd Quartile	Fourier Transform 10th Component
Flow Duration	Packet Inter-arrival Time 1st Quartile	
	Packet Inter-arrival Time 3rd Quartile	

Fonte: Próprio autor.

3.3 Avaliação de Características de Fluxo

O processo de avaliação de características de fluxo envolve avaliar o conjunto de características criado e determinar quais características formam o subconjunto mais eficaz na classificação de fluxos de tráfego em um determinado cenário. Essa etapa é responsabilidade do Flow Feature Selector Component.

3.3.1 Feature Selector Module

O Feature Selector Module é responsável por executar técnicas de seleção de características a fim de determinar um subconjunto que torne a classificação de fluxos de tráfego mais precisa. A seleção de características pode ser realizada por meio de qualquer algoritmo de seleção de características. Neste trabalho são avaliados três algoritmos diferentes: Sequential Backward Selection (SBS); Principal Component Analysis (PCA); e Genetic Algorithm (GA). Estes algoritmos são utilizados individualmente, mas também podem ser utilizados em conjunto; por exemplo, pode-se usar o conjunto obtido com a utilização do SBS como sugestão para o GA (BLUM; LANGLEY, 1997).

Estes algoritmos foram escolhidos por se tratarem de três algoritmos com propostas diferentes para a criação dos subconjuntos de características: o SBS elimina características por dificultarem a classificação, o PCA utiliza a correlação entre características para determinar o melhor subconjunto dessas e o GA realiza uma busca pelo espaço de possíveis soluções. A seguir, os três algoritmos são descritos em maiores detalhes.

O primeiro algoritmo, SBS, trata-se de um filtro muito simples: começando com o conjunto completo, esse algoritmo avalia, a cada interação, qual característica deve ser eliminada. O processo de escolha da característica a ser eliminada começa com um subconjunto inicial e avalia a qualidade desse com a eliminação de cada característica individualmente, sendo que o subconjunto escolhido para a próxima interação é aquele que apresenta o melhor desempenho e possui uma característica a menos que o subconjunto inicial. Em outras palavras, este algoritmo elimina, uma a uma, as características que inserem ruídos que dificultam a classificação. O SBS possui dois métodos de parada: quando atinge o número de características que se deseja eliminar e quando, em alguma iteração, todos os possíveis novos subconjuntos resultam em um decréscimo na qualidade.

O SBS não garante que o subconjunto final será o melhor subconjunto existente existente, pois não considera a terceira definição de Blum, descrita na Seção 2.3, de modo que uma característica depois de eliminada, nunca é adicionada novamente ao conjunto, mesmo que possa resultar em uma melhor qualidade.

Existem duas maneiras de implementar o SBS: *(i)* avaliando apenas as características e a relação entre elas; ou *(ii)* usando-o juntamente com um algoritmo de classificação, assemelhando-se a um *wrapper*, quando cada subconjunto é enviado ao algoritmo de classificação, obtendo a qualidade exata de cada um deles. Essa segunda opção é a utilizada neste trabalho. O Pseudo Código apresentado na Figura 3.2 descreve o funcionamento do SBS.

O segundo algoritmo implementado é o PCA. Esse algoritmo também é classificado como sendo um filtro. O método matemático PCA foi desenvolvido por Karl Pearson em 1901 (PEARSON, 1901) e avalia a correlação existente entre as diferentes variáveis - neste caso as características dos fluxos de tráfego - de um problema, criando o que são chamados de Componentes Principais.

Figura 3.2: Funcionamento do algoritmo SBS.

Algorithm 1 Simple Backward Selection

```
1: Y ← completeFeatureSet
2: Acc_h ← accuracyOf(Y)
3: n ← numberOfFeaturesToRemove
4: for 0 to n do
5:     f_rem ← null
6:     for each feature f_i ϵ Y do
7:         Y_temp ← Y − f_i
8:         Acc_temp ← accuracyOf(Y_temp)
9:         if Acc_temp ≥ Acc_h then
10:            Acc_h ← Acc_temp
11:            f_rem ← f_i
12:    if f_rem ≠ null then
13:        Y ← Y − f_rem
14:    else
15:        stop
```

O PCA transforma as variáveis iniciais em Componentes Principais linearmente não relacionados por meio de uma transformação ortogonal. Esse procedimento cria um novo espaço N-dimensional, onde N é o número inicial de variáveis e cada eixo é representado por um componente principal.

Os componentes principais são compostos pelas variáveis originais multiplicadas por um peso. Esse peso varia de -1 a 1 e, quanto mais próximo de 1, mais importante a variável é para o componente. Além disso, são ordenados pela variância entre as variáveis, sendo o primeiro componente o com a maior variabilidade, ou seja, aquele que melhor representa o conjunto de dados. O peso que cada característica possui na criação dos componentes é utilizado como critério de seleção para os subconjuntos: as características que possuem o peso maior que um determinado threshold são selecionadas e as demais, descartadas.

O terceiro e último algoritmo avaliado é o GA. Esse algoritmo realiza uma busca no espaço de solução baseando-se na seleção natural por meio da combinação de genes, evoluindo

Figura 3.3: Funcionamento do Algoritmo Genético.

Algorithm 2 Genetic Algorithm

1: $gen = 0$
2: $initPopulation(P(gen))$
3: $evaluate(P(gen))$
4: **while** $stopCondition = false$ **do**
5: $\quad P_p(gen) \leftarrow fittestIndividuals(P(gen))$
6: $\quad P_c(gen) \leftarrow crossover(P_p)$
7: $\quad mutate(P_c(gen))$
8: $\quad evaluate((P_c(gen)))$
9: $\quad P(gen+1) \leftarrow newPoputalion(P(gen), P_c(gen))$
10: $\quad gen \leftarrow gen + 1$

os elementos de uma população a cada geração. Os elementos que compõem esta classe de algoritmos são: uma população na qual cada indivíduo é uma possível solução para o problema; uma função de avaliação, responsável por avaliar a qualidade de cada indivíduo, ou seja, a qualidade da solução; e uma condição de parada, que pode ser um teste de convergência ou o limite de gerações que são rodadas.

GAs normalmente representam os indivíduos de uma população em um formato binário e, sobre esta representação realizam, as operações de *crossover* e *mutation*. Crossover é a operação de pareamento de dois indivíduos para gerar dois novos indivíduos na geração seguinte, começando com os dois indivíduos mais adaptados (com maior qualidade). A mutação tem o objetivo de inserir novas características aos indivíduos, invertendo bits dos indivíduos de acordo com uma probabilidade. Essa mutação não ocorre nos indivíduos mais adaptados para não perder qualidade de soluções (HAUPT; HAUPT, 1998). O Pseudo Código apresentado na Figura 3.3 resume o funcionamento de um algoritmo genético, a função de avaliação da população é determinada pela aplicação, neste trabalho, esta função avalia a qualidade dos subconjuntos.

3.3.2 Flow Classifier Module

A classificação dos fluxos de tráfego é realizada por meio de dois algoritmos, um algoritmo de aprendizado supervisionado - Support Vector Machine (SVM) - e um de aprendizado não supervisionado, o K-means.

Support Vector Machines são algoritmos utilizados em diversas áreas, desde no reconhecimento de imagens até na bioinformática. E já foi confirmada a eficácia desse algoritmo para a detecção de ataques maliciosos, tais como DDoS (MUKKAMALA; SUNG, 2003).

O princípio de funcionamento do SVM é a construção de um hiperplano que separa as classes de um problema em um espaço N-dimensional, onde N é a quantidade de características do problema. Esta separação procura maximizar a distância entre cada uma das classes e é posicionado entre os dois pontos mais próximos pertencentes a duas classes diferentes (BENNETT; CAMPBELL, 2000).

O segundo algoritmo implementado é o K-means. Esse algoritmo é baseado em aprendizado não supervisionado, sendo sua utilização voltada para processos de clusterização. O K-means separa as amostras de um espaço n-dimensional em clusters, cuja partição minimiza o erro quadrático médio entre o centro dos clusters e cada ponto pertencente a este. O procedimento, de acordo com MacQueen (MACQUEEN, 1967), consiste em iniciar o procedimento com k clusters, cada um sendo um ponto randomicamente selecionado no espaço, e a cada iteração adicionar a cada cluster o ponto que minimiza a média da distância entre eles.

O K-means possui três parâmetros que devem ser especificados pelo usuário: (i) a quantidade k de clusters que devem ser criados; (ii) a quantidade de iterações; e (iii) a métrica para o cálculo da distância entre os pontos (JAIN, 2010).

4 PROTÓTIPO E ANÁLISE DE RESULTADOS

Este capítulo descreve o ambiente em que o sistema é avaliado e os parâmetros adotados para os algoritmos utilizados. Também são apresentados os casos de testes rodados e resultados obtidos.

4.1 Ambiente de Execução

O ambiente de execução do sistema deste trabalho consiste em: uma rede organizada em uma topologia de árvore - criada utilizando o Mininet[1]; e um controlador externo, como o Floodlight Controller[2], utilizado neste trabalho. Foi escolhida uma topologia em árvore pois esta ser escalável e muito utilizada na prática, por exemplo em data-centers e universidades. Adicionalmente, sobre esse ambiente, são executados um módulo gerador de tráfego e o mecanismo de coleta, extensão e análise de características de fluxos de tráfego, descrito no Capítulo 3.

Os fluxos de tráfego gerados na rede são de quatro tipos diferentes: (i) um servidor HTTP, que é alvo de ataques DDoS partindo de diversos hosts; (ii) um servidor de stream de vídeo hospedado em um host e os demais hosts assinam este serviço, sendo este tráfego gerado com a utilização do programa VLC[3] tanto no servidor quanto nos clientes; (iii) trocas de arquivos entre os hosts utilizando o protocolo FTP; e (iv) tráfego TCP gerado através do Scapy[4].

Tanto o sistema quanto os programas geradores de tráfego são implementados usando Python v2.7.6. A estrutura de dados que armazena o conjunto de características é um dicionário cuja chave é a 5-tupla identificadora de fluxos, descrita na Seção 3.2, e os arquivos de treinamento e teste são salvos em formato .csv. O algoritmo SBS é implementado usando Python v2.7.6 enquanto os algoritmos SVM, PCA, GA, e K-means são implementados em R v3.0.2 utilizando as bibliotecas e1071, psych, genalg e cluster, respectivamente.

[1] http://mininet.org/
[2] http://www.projectfloodlight.org
[3] http://www.videolan.org/vlc/index.html
[4] http://www.secdev.org/projects/scapy/

4.2 Casos de Teste

Com objetivo de avaliar a qualidade das características geradas e o impacto da etapa de seleção de características na classificação de fluxos de tráfego, foram escolhidos três cenários de teste nos quais os algoritmos de classificação procuram identificar corretamente o tráfego de background, gerado com o Scapy, dos outros três perfis de tráfego descritos na Seção 4.1. A Tabela 4.1 apresenta os três cenários de testes avaliados neste trabalho.

Tabela 4.1: Cenários de teste definidos para avaliar o sistema desenvolvido.

Cenário A	Ataques DDoS juntamente com tráfego de background
Cenário B	Tráfego FTP juntamente com tráfego de background
Cenário C	Streaming de vídeo juntamente com tráfego de background

Fonte: Próprio autor.

Em cada um dos três cenários são utilizadas, para o treinamento do SVM, 1500 amostras de fluxo de cada perfil de tráfego e 250 amostras de cada perfil de tráfego para teste. E, para o processo de clusterização realizado pelo K-means, são utilizadas 1500 amostras de cada perfil de tráfego em todos os três cenários de teste.

A quantidade de amostras utilizadas nesse trabalho é pequena se comparada com outros trabalhos: em média seis mil amostras para Mukkamala e Sung (MUKKAMALA; SUNG, 2003) e noventa e sete mil para Bennet e Campbell (BENNETT; CAMPBELL, 2000). As quantidades de amostras descritas no parágrafo anterior (1500 e 250 amostras de cada perfil de tráfego), são utilizadas nesse trabalho para avaliar com maior precisão a qualidade das características criadas e também o impacto da seleção de características na classificação de fluxos de tráfego. Durante os experimentos, foram testados outros conjuntos de treinamento com cinco mil e dez mil amostras. Estes casos já apresentavam uma qualidade de classificação adequada utilizando as trinta e três características criadas, em torno de 98% de acurácia na classificação do algoritmo SVM, deixando pouco espaço para avaliar o impacto da adoção de seleção de características nos mesmos.

Para permitir a comparação entre a qualidade de cada subconjunto criado serão utilizadas as métricas de acurácia para o algoritmo SVM - ou seja, a porcentagem de amostras corretamente classificadas - e o método de Silhouettes para o algoritmo K-means. Os valores de acurácia variam de 0% a 100% e Silhouettes de -1 a 1, sendo 1 a melhor qualidade. Ambas métricas são descritas em mais detalhes na Seção 2.2.2.

4.3 Parametrizações dos Algoritmos de Seleção de Características

Algoritmos de seleção de características possuem diversos parâmetros a serem definidos pelos usuários; esses parâmetros devem ser refinados de acordo com a aplicação em questão, com objetivo de alcançar melhores resultados. As próximas seções descrevem os parâmetros e seus valores para cada algoritmo de seleção de características empregado neste trabalho. A parametrização dos algoritmos de classificação é descrita na Seção 4.4.

4.3.1 Sequential Backward Selection

O Sequential Backyard Selection, por ser um algoritmo simples, só possui um parâmetro a ser definido: a quantidade de características que serão eliminadas. O valor definido é o tamanho do conjunto de características completo (trinta e três), pois quanto menor o conjunto de características, menor a demanda de recursos computacionais para a classificação dos fluxos de tráfego.

A utilização deste valor só é possível devido à existência do segundo critério de parada descrito na Seção 3.3.1. O algoritmo interrompe a execução quando a qualidade do subconjunto decresce com a eliminação de qualquer uma das características do subconjunto que está sendo avaliado. Dessa forma, este algoritmo cria o menor subconjunto possível que tenha um desempenho máximo localmente.

4.3.2 Principal Component Analysis

O PCA possui como parâmetro o número de componentes principais que se deseja extrair. Nesse trabalho é adotado o valor padrão que é a quantidade de características definidas inicialmente. Nesse caso, o algoritmo cria trinta e três componentes principais PC_n que são compostos pelo somatório do valor de cada característica f_i multiplicado por um peso w_i que varia de -1 a 1, de acordo com a Equação 4.1.

$$PC_n = \sum_{i=1}^{33} f_i * w_i \tag{4.1}$$

Cada componente principal representa uma proporção da variabilidade do conjunto de dados, sendo os primeiros componentes os mais representativos. Os experimentos demonstra-

ram, nos três cenários de teste, que os primeiros onze componentes representam mais de 90% da variabilidade, restando menos de 10% para os demais vinte e dois componentes. Portanto, são considerados apenas os onze primeiros como sendo significativos, visto que os outros vinte e dois individualmente representam muito pouco os dados do problema.

Uma questão importante é como estes componentes são empregados na criação de um subconjunto de características. Uma opção seria transformar todas as características de todas as amostras nesses componentes, e utilizar os componentes principais para a classificação dos fluxos de tráfego e não as características originais. Esta abordagem não é utilizada pois, desta forma, as características dos fluxos não teriam mais um significado individualmente.

Assim, a proposta abordada é utilizar os pesos usados para gerar os componentes principais para selecionar as características, visto que esse peso informa o quanto uma característica é relevante para o componente (quanto mais perto de 1 mais relevante é a característica para o componente e quanto mais próximo a -1 menos relevante). Para isso, foram escolhidos três fatores que servem como limite. As características que possuem o peso maior que o fator de limite são selecionadas para o subconjunto e as demais são descartadas. Os três fatores são: 0.025, 0.05 e 0.1. Estes foram escolhidos por se encontrarem no intervalo entre 0 e 0.1, onde a maior parte dos pesos se encontram, sendo que as características mais importantes possuem um peso bem maior e as irrelevantes, um peso muito próximo ou menor que 0. As demais características variam o seu peso próximo à essa faixa. Assim, a diferença de um subconjunto para o outro está nas características que possuem o seu peso dentro deste intervalo.

4.3.3 Genetic Algorithm

Algoritmos genéticos possuem diversos parâmetros a serem definidos[5], são estes: tamanho da população, quantidade de gerações, probabilidade de mutação e elitismo. O tamanho da população é definido em duzentos indivíduos randomicamente inicializados; ou seja, em cada geração são avaliadas duzentas soluções possíveis.

Devido ao fato de que a biblioteca utilizada não oferece condição de parada baseada em convergência, é preciso definir a quantidade de gerações que serão executadas. Para o aprendizado supervisionado, são rodadas duzentas gerações e para a clusterização, são rodadas cem gerações. Essa diferença é causada pelo tempo de execução do algoritmo em cada caso: no caso do K-means esse tempo é em média 16.26h e no caso do SVM, é apenas 1.006h.

A probabilidade de mutação, como já foi comprovado (SANTOS et al., 2015), não influ-

[5]https://cran.r-project.org/web/packages/genalg/genalg.pdf

encia na qualidade da solução final; para essa foi utilizado o valor 0.01. Para o elitismo, que é a porcentagem da população que não é afetada por mutação, é utilizado o valor padrão de 20%.

4.4 Resultados

Nesta seção são apresentados os resultados obtidos aplicando os algoritmos de seleção de características ao conjunto inicial e computando a melhoria, ou não, na qualidade da classificação dos fluxos de tráfego para cada um dos três cenários definidos.

Primeiramente, é realizada a classificação dos fluxos de tráfego utilizando o conjunto completo de características para se ter uma qualidade de classificação inicial. A Tabela 4.2 apresenta a acurácia de classificação do SVM e a qualidade dos clusters do K-means.

Tabela 4.2: Qualidade da classificação utilizando o conjunto completo de características.

	Cenário A	**Cenário B**	**Cenário C**
SVM	95.4%	93.2%	70%
K-means	0.92742	0.93696	0.88831

Fonte: Próprio autor.

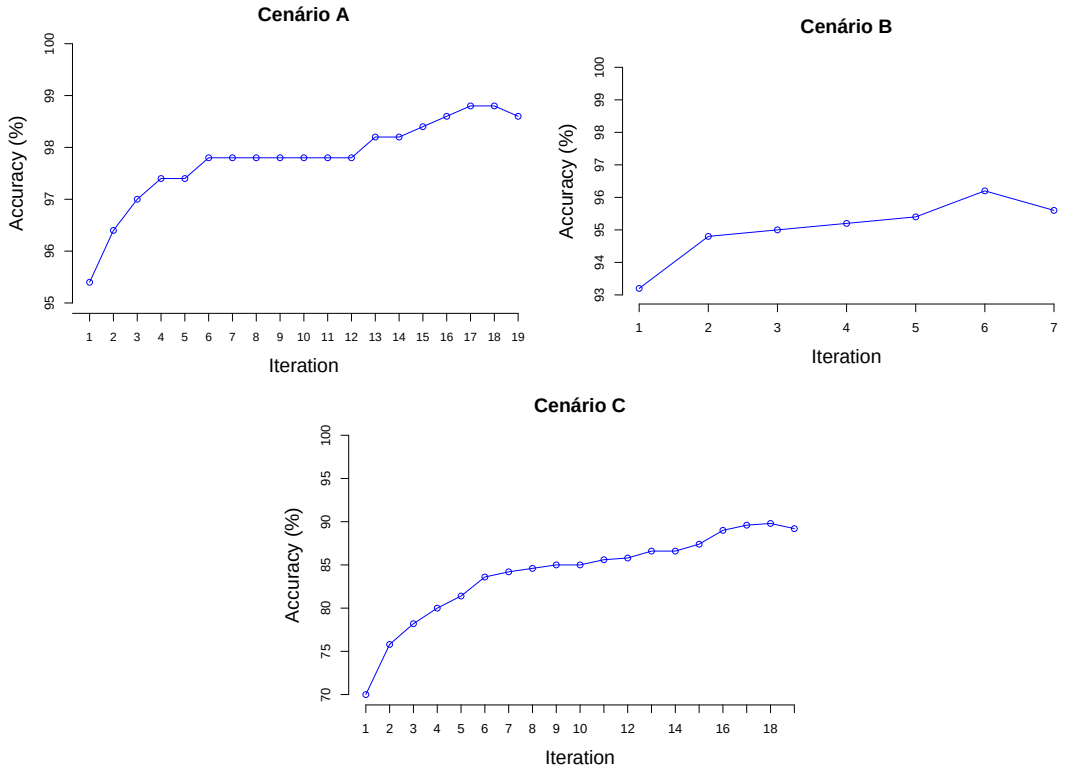
4.4.1 Aprendizado Supervisionado - SVM

O primeiro algoritmo de classificação a ser avaliado é o SVM. Este algoritmo não possui parâmetros específicos a serem definidos e recebe como entrada apenas o arquivo de treinamento. Após isso, o classificador diz a qual classe, ou perfil de tráfego, pertence cada amostra do arquivo de teste, e como já se sabe a priori a qual cada classe pertencem todas as amostras do arquivo, é possível determinar a acurácia do classificador.

Dessa forma, são executados os três algoritmos de seleção de características utilizando os parâmetros descritos na Seção 4.3. O primeiro algoritmo de seleção de características executado é o SBS e os gráficos apresentados na Figura 4.1 apresentam a evolução da acurácia do subconjunto a cada iteração do algoritmo. Duas observações podem ser feitas nestes gráficos: *(i)* a última iteração resulta em um decréscimo da acurácia, indicando o momento em que o algoritmo interrompe a execução e retorna o subconjunto obtido na iteração anterior; e *(ii)* existem dois momentos, no Cenário A, em que a acurácia não sofre alteração. Isso mostra que o algoritmo sempre busca o menor subconjunto possível, mesmo quando a acurácia não se altera com a eliminação de uma característica.

O segundo algoritmo de seleção de características avaliado é o PCA. A Figura 4.2 apresenta três gráficos, um para cada cenário, apresentando a acurácia de cada subconjunto criado por este algoritmo. Nestes gráficos, as linhas horizontais representam a acurácia do conjunto completo de características e a acurácia do melhor subconjunto criado. Estas linhas demonstram a evolução na qualidade da classificação utilizando o melhor subconjunto criado pelo PCA.

Figura 4.1: Evolução da acurácia do subconjunto a cada iteração do algoritmo SBS.

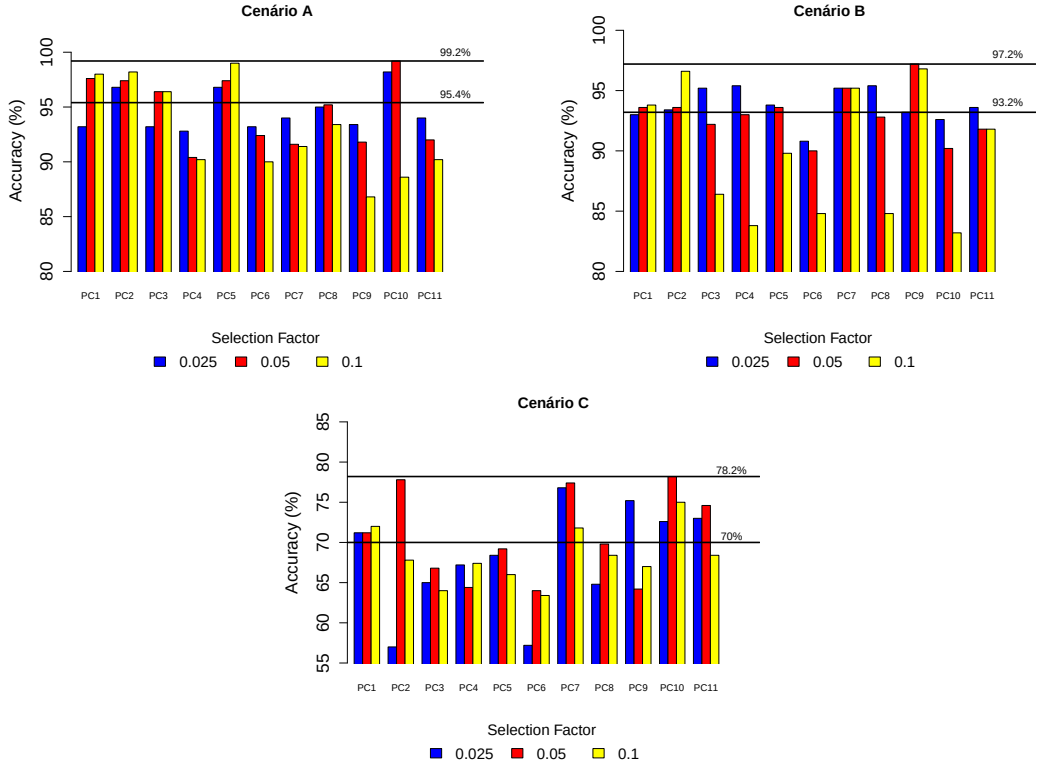


Fonte: Próprio autor.

O que pode ser observado nos gráficos da Figura 4.2 é a importância da escolha do peso utilizado como fator adotado para fazer a seleção. A qualidade dos subconjuntos criados por alguns dos componentes possuem uma grande variação com a mudança do fator de seleção, como o componente PC2 do cenário C e o PC10 do cenário A.

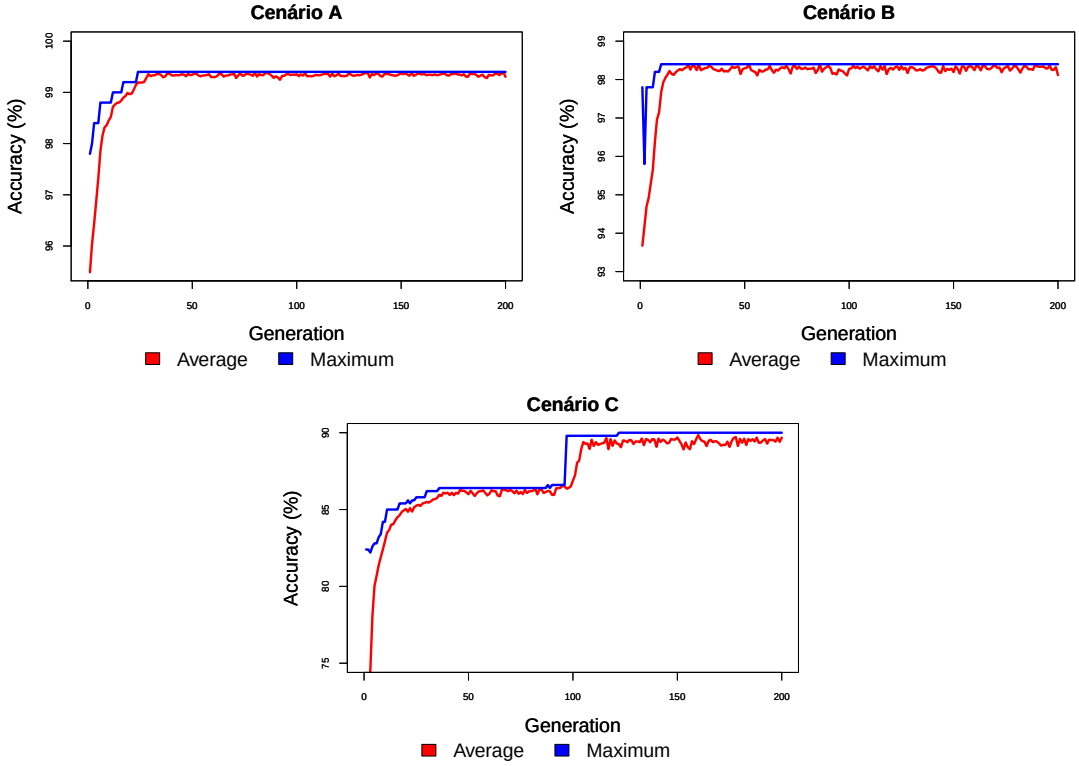
Os três gráficos presentes na Figura 4.3 apresentam a evolução da acurácia da classificação para cada um dos subconjuntos analisados pelo algoritmo genético ao longo das duzentas gerações nas quais o algoritmo é rodado. Em vermelho é apresentada a média de cada geração e azul a maior acurácia encontrada em cada geração. Pode-se observar que a média tende ao máximo encontrado algumas gerações após um novo máximo ser encontrado e, mesmo com a

Figura 4.2: Acurácia dos subconjuntos de características criados pelo algoritmo PCA.



Fonte: Próprio autor.

Figura 4.3: Gráficos que apresentam a evolução da acurácia dos subconjuntos criados pelo Algoritmo Genético ao longo das 200 gerações.



Fonte: Próprio autor.

solução tendo convergido para um máximo, o algoritmo continua avaliando outros subconjuntos, o que pode ser notado pelas variações apresentadas pela média ao longo das gerações.

A Tabela 4.3 compara a acurácia atingida pelo melhor subconjunto de características criado por cada algoritmo de seleção de características. O tamanho de cada subconjunto de características é informado entre parênteses após a acurácia do mesmo. As tabelas 4.5, 4.6 e 4.7 apresentam as características presentes em cada um dos três subconjuntos para os cenários A, B e C, respectivamente.

Tabela 4.3: Tabela comparativa entre a acurácia dos subconjuntos criados para a classificação utilizando algoritmo SVM.

	Todas Características	**SBS**	**PCA**	**Algoritmo Genético**
Cenário A	95.4%	98.8% (15)	99.2% (11)	99.4% (24)
Cenário B	93.2%	96,2% (27)	97.2% (15)	98.4% (28)
Cenário C	70%	89.8% (15)	78.2% (11)	90% (18)

Fonte: Próprio autor.

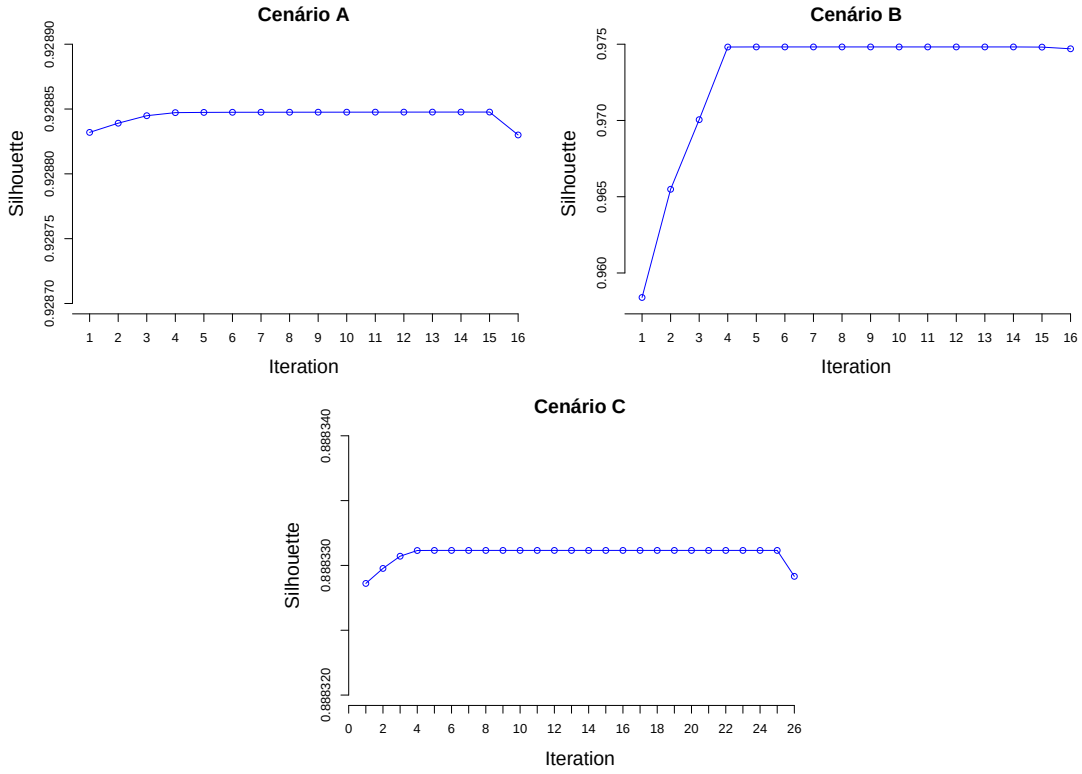
4.4.2 Aprendizado Não supervisionado - K-means

Diferentemente do SVM, o K-means possui alguns parâmetros a serem definidos: a quantidade de clusters a serem criados, a quantidade máxima e mínima de iterações e a fórmula de se calcular as distâncias entre pontos. Devido ao fato de que cada um dos três cenários de teste possui dois perfis de tráfego diferentes e sabendo que a métrica de Silhouettes só é válida para comparar a qualidade de clusterizações com a mesma quantidade de clusters, o k-means é configurado para criar apenas dois clusters, sendo que cada um destes clusters representa um dos perfis de tráfego de cada cenário de teste. Para as quantidades mínimas e máximas de iterações são utilizados os valores padrões de cem e mil iterações, respectivamente; e para o cálculo da distância entre os pontos é utilizada a distância Euclidiana.

Igualmente ao SVM, as figuras 4.4, 4.5 e 4.6 apresentam a qualidade dos subconjuntos de características criados pelos algoritmos SBS, PCA e GA, respectivamente. A Tabela 4.4 apresenta os resultados para cada um dos três algoritmos de seleção de características e o tamanho de cada subconjunto.

Com objetivo de ter uma melhor visualização da melhoria da qualidade da clusterização oferecida por cada algoritmo de seleção de características, são plotadas todas as amostras do arquivo de treinamento em um espaço bidimensional. As figuras 4.7, 4.8 e 4.9 apresentam o

Figura 4.4: Evolução da qualidade da clusterização do subconjunto a cada iteração do algoritmo SBS.



Fonte: Próprio autor.

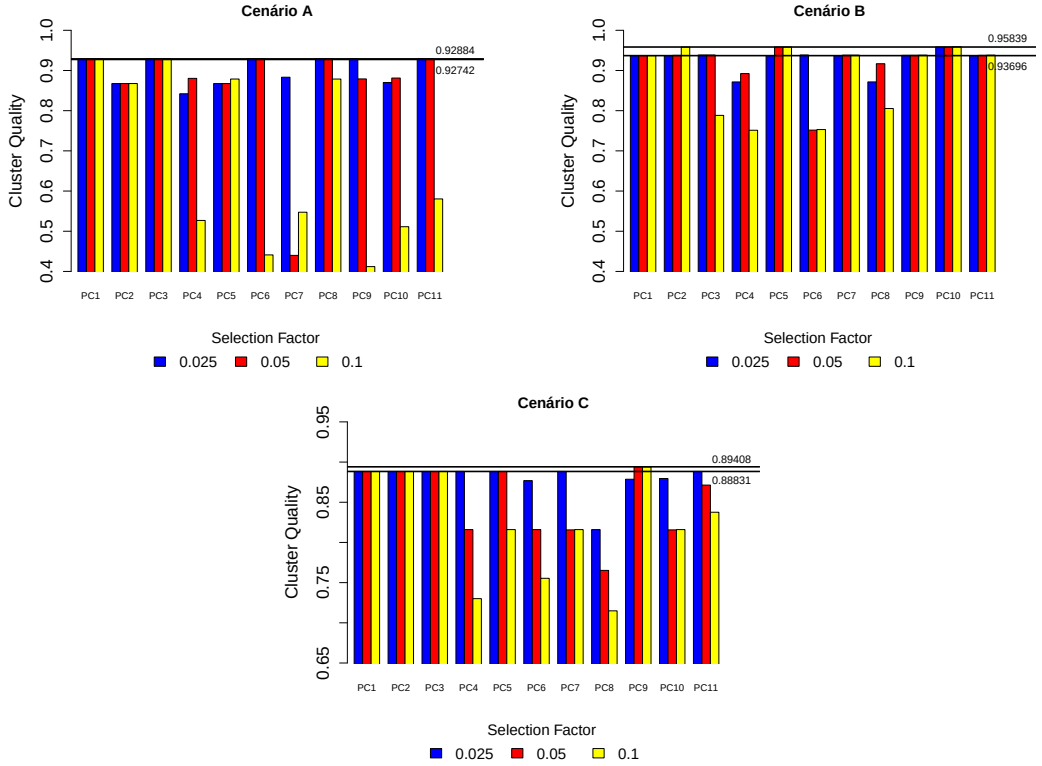
resultado da clusterização para os cenários A, B e C, respectivamente. Nesses gráficos são identificados os dois clusters por meio das elipses azul e vermelha.

Tabela 4.4: Tabela comparativa entre a Silhueta dos subconjuntos criados para clusterização utilizando K-means.

	Todas Características	**SBS**	**PCA**	**Algoritmo Genético**
Cenário A	0.92742	0.92884 (18)	0.92884 (12)	0.92809 (12)
Cenário B	0.93696	0.97482 (18)	0.95839 (13)	0.97482 (5)
Cenário C	0.88831	0.88833 (8)	0.89408 (5)	0.90481 (16)

Pode-se observar, nos gráficos gerados pelo K-means, que as amostras se encontram distribuídas em grupos visivelmente mais compactos quando a etapa de seleção de características é aplicada ao problema. Porém, esta visualização não pode ser utilizada como único parâmetro de avaliação entre algoritmos por ser uma representação em apenas duas dimensões de um espaço multidimensional.

Figura 4.5: Qualidade dos subconjuntos de características criados pelo algoritmo PCA.



Fonte: Próprio autor.

Também é possível observar que os dois componentes representados neste espaço (eixos X e Y) possuem um maior grau de representação nos resultados obtidos utilizando os subconjuntos de características selecionadas, quando comparado ao resultado com o resultado utilizando as trinta e três características. A única exceção é o subconjunto criado pelo algoritmo SBS para o cenário B (Figura 4.8). Isso é um indicador da qualidade da clusterização: quanto mais representativos são cada componente gerados pelo k-means, mais precisa é a clusterização.

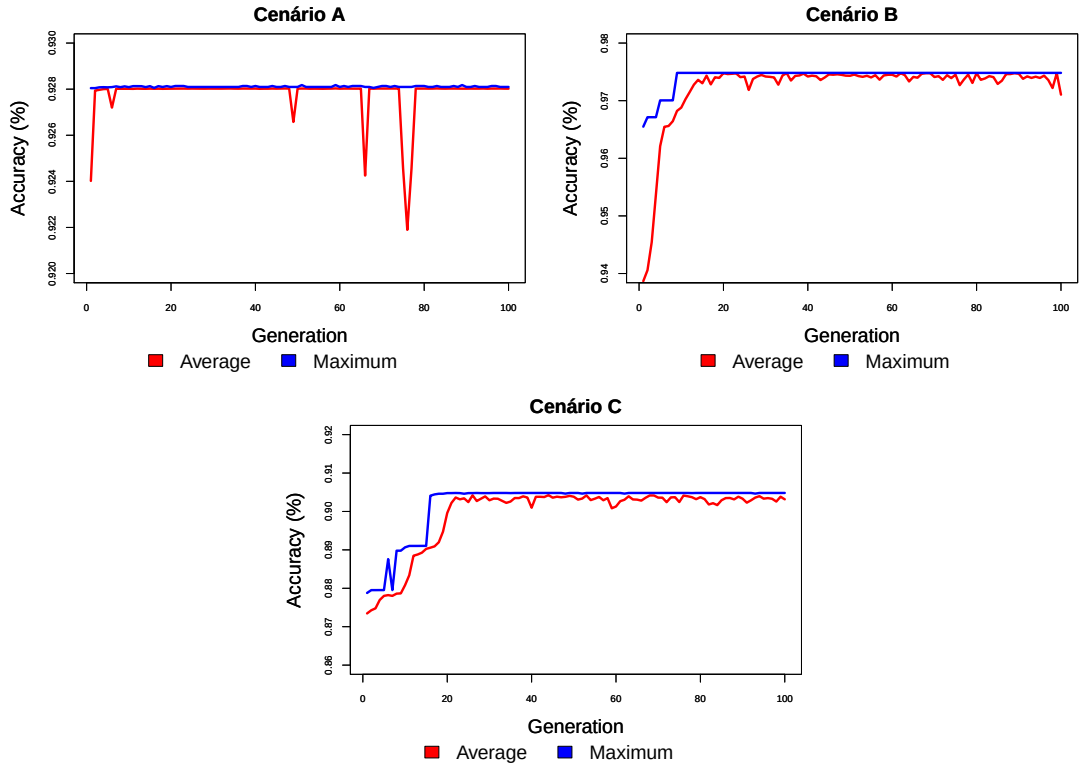
As tabelas 4.8, 4.9 e 4.10 apresentam as características presentes em cada um dos três subconjuntos para os cenários A, B e C, respectivamente.

4.4.3 Comparativo da Classificação dos Fluxos de Trafego Utilizando o Conjunto Completo e os Subconjuntos Criados

Os gráficos apresentados na Figura 4.10 resumem os resultados obtidos com a execução dos algoritmos de classificação utilizando o conjunto completo de características e os subconjuntos compostos pelas características selecionadas por cada um dos algoritmos de seleção de características.

Observando o gráfico à esquerda, que apresenta o comparativo da acurácia do SVM,

Figura 4.6: Gráficos que apresentam a evolução da qualidade dos subconjuntos criados pelo Algoritmo Genético ao longo das 100 gerações.



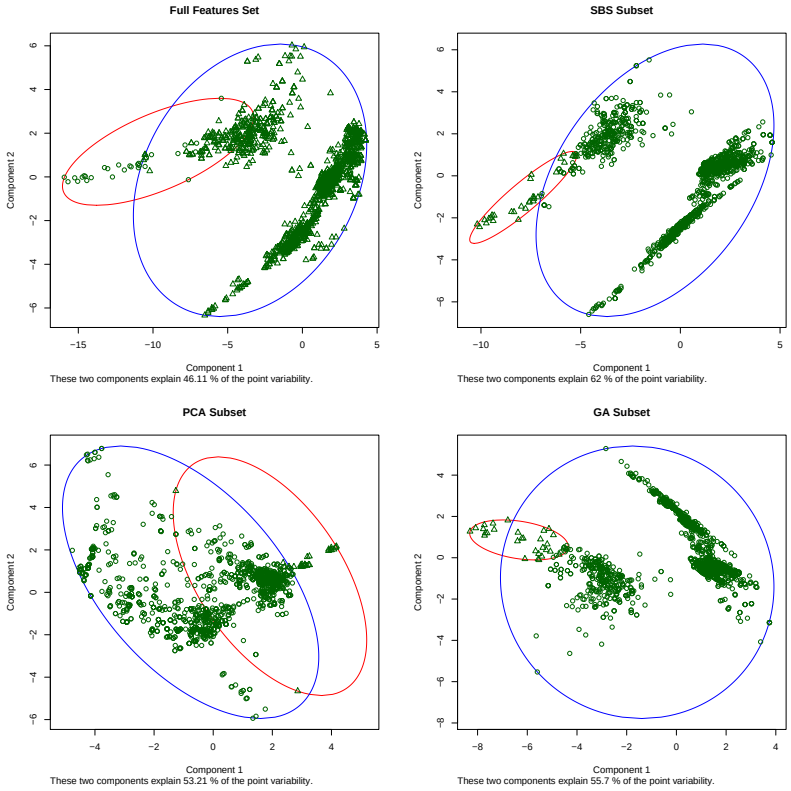
Fonte: Próprio autor.

pode-se tirar diversas conclusões. Entre estas: cada técnica de seleção de características oferece um ganho diferente de acurácia, principalmente no cenário C, onde a acurácia é aprimorada em 20%; o GA é o algoritmo que cria o melhor subconjunto de características em todos os cenários, por se tratar de um método heurístico de busca que abrange um grande espaço de soluções; o SBS oferece um desempenho muito próximo ao dos outros algoritmos mais complexos, ficando até mesmo à frente do PCA no cenário C, provando ser um algoritmo rápido e eficiente.

Em contrapartida, os resultados obtidos com a aplicação dos algoritmos de seleção de características para clusterização por meio do K-means não são tão óbvios. Existe uma contradição entre o que é apresentado no gráfico à direita da Figura 4.10 e os gráficos da Figura 4.7. Isto se deve ao fato de que a silhueta para o Cenário A não apresenta melhoria significativa com a utilização dos algoritmos de seleção, porém a visualização gráfica dos clusters leva a entender o contrário, ou seja, que os algoritmos de seleção de fato melhoram a qualidade da clusterização. Um segundo ponto que deve ser reparado é em relação ao Cenário B: o gráfico comparativo (Figura 4.10) aponta uma melhoria da silhueta dos clusters (aproximadamente 4.04%) e os gráficos da Figura 4.8 mostram que os dois principais componentes (eixos X e Y dos gráficos) passam a representar menos os dados, passando de 50.24% de representatividade para 44.74%.

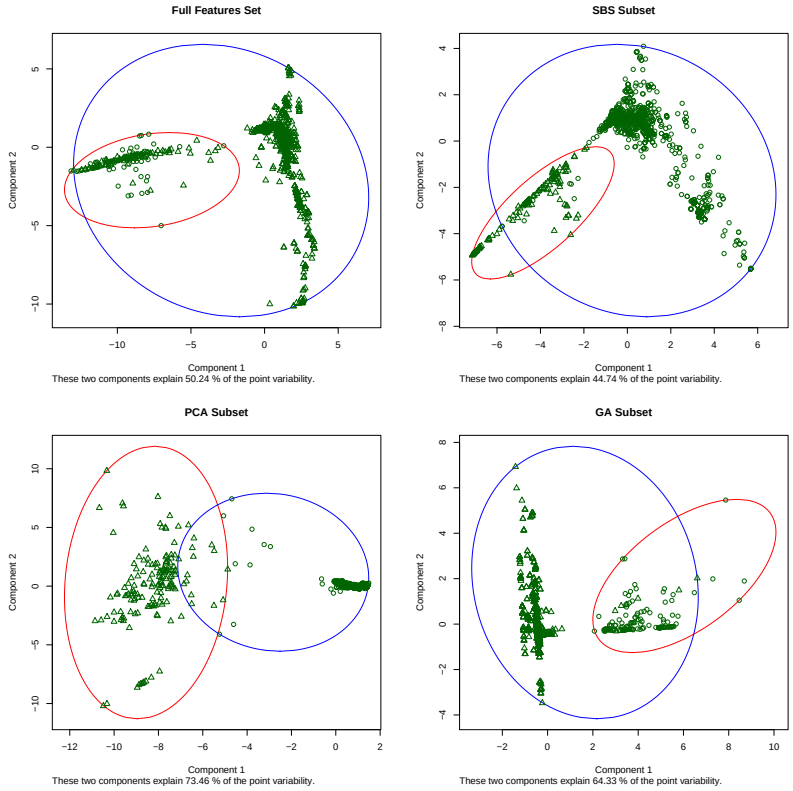
Pode-se concluir observando exclusivamente os valores da silhueta que essas técnicas não são indicadas para a detecção de ataques DDoS e que o método de SBS é a melhor opção para o cenário B, pois apresenta uma melhoria na qualidade com um baixo custo computacional.

Figura 4.7: Resultados da clusterização realizada pelo algoritmo K-means para o cenário A.



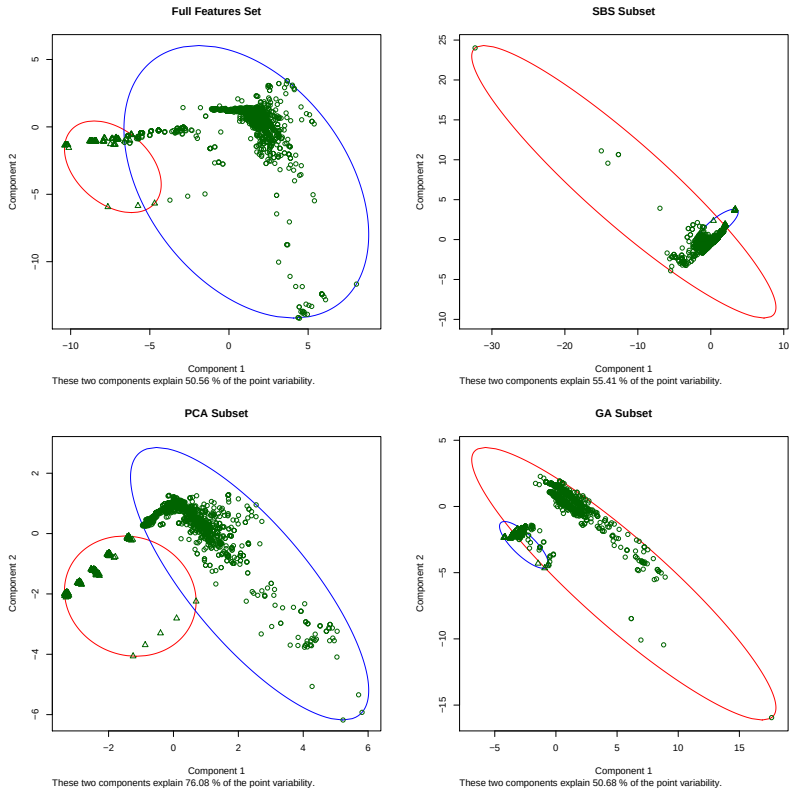
Fonte: Próprio autor.

Figura 4.8: Resultados da clusterização realizada pelo algoritmo K-means para o cenário B.



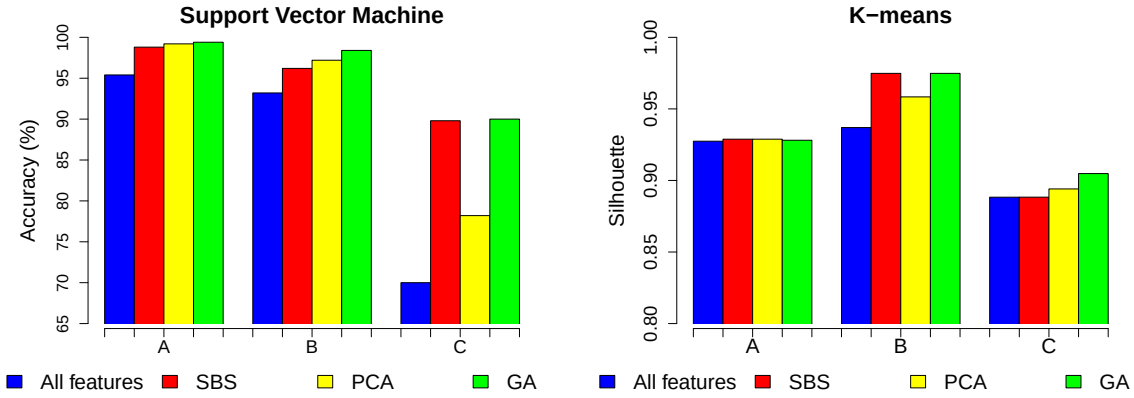
Fonte: Próprio autor.

Figura 4.9: Resultados da clusterização realizada pelo algoritmo K-means para o cenário C.



Fonte: Próprio autor.

Figura 4.10: Gráficos comparativos da qualidade da solução oferecida pela utilização dos três algoritmos de seleção de características.



Fonte: Próprio autor.

Tabela 4.5: Características que constituem os subconjuntos do Cenário A na classificação utilizando SVM.

SBS	**PCA**	**GA**
Bytes per second Minimum value	Packet Length Mean	Bytes per second Mean
Packets per second Maximum value	Packet Length Variance	Bytes per second Variance
Packets per second Minimum value	Packet Length Minimum value	Bytes per second Maximum value
Packet Length Maximum value	Packet inter-arrival time Maximum value	Bytes per second Minimum value
Packet inter-arrival time Variance	Packet inter-arrival time Minimum value	Packets per second Mean
Packet inter-arrival time Minimum value	Flow Size in packets	Packets per second Variance
Flow duration	Packet inter-arrival time 1st quartile	Packets per second Maximum value
Flow Size in Bytes	Packet Length 1st quartile	Packets per second Minimum value
Flow Size in packets	Packet Length 3rd quartile	Packet Length Mean
Packet Length 1st quartile	Fourier Transform 2nd component	Packet Length Variance
Packet Length 3rd quartile	Fourier Transform 10th component	Packet Length Maximum value
Fourier Transform 3rd component		Packet Length Minimum value
Fourier Transform 4th component		Packet inter-arrival time Variance
Fourier Transform 5th component		Packet inter-arrival time Maximum value
Fourier Transform 8th component		Packet inter-arrival time Minimum value
		Flow duration
		Flow Size in Bytes
		Flow Size in packets
		Packet inter-arrival time 1st quartile
		Packet inter-arrival time 3rd quartile
		Packet Length 1st quartile
		Packet Length 3rd quartile
		Fourier Transform 2nd component
		Fourier Transform 10th component

Fonte: Próprio autor.

Tabela 4.6: Características que constituem os subconjuntos do Cenário B na classificação utilizando SVM.

SBS	**PCA**	**GA**
Bytes per second Mean	Bytes per second Mean	Bytes per second Mean
Bytes per second Variance	Bytes per second Minimum value	Bytes per second Variance
Bytes per second Maximum value	Packets per second Mean	Bytes per second Maximum value
Bytes per second Minimum value	Packets per second Variance	Bytes per second Minimum value
Packets per second Variance	Packets per second Maximum value	Packets per second Mean
Packets per second Maximum value	Packet Length Minimum value	Packets per second Minimum value
Packets per second Minimum value	Packet inter-arrival time Mean	Packet Length Mean
Packet Length Mean	Packet inter-arrival time Variance	Packet Length Variance
Packet Length Variance	Packet inter-arrival time Maximum value	Packet Length Maximum value
Packet Length Maximum value	Packet inter-arrival time Minimum value	Packet Length Minimum value
Packet Length Minimum value	Flow Size in Bytes	Packet inter-arrival time Mean
Packet inter-arrival time Mean	Flow Size in packets	Packet inter-arrival time Variance
Packet inter-arrival time Variance	Fourier Transform 3rd component	Packet inter-arrival time Maximum value
Packet inter-arrival time Maximum value	Fourier Transform 7th component	Packet inter-arrival time Minimum value
Packet inter-arrival time Minimum value	Fourier Transform 10th component	Flow Size in Bytes
Flow Size in Bytes		Flow Size in packets
Flow Size in packets		Packet inter-arrival time 1st quartile
Packet inter-arrival time 1st quartile		Packet inter-arrival time 3rd quartile
Packet inter-arrival time 3rd quartile		Packet Length 1st quartile
Packet Length 1st quartile		Packet Length 3rd quartile
Packet Length 3rd quartile		Fourier Transform 1st component
Fourier Transform 1st component		Fourier Transform 2nd component
Fourier Transform 2nd component		Fourier Transform 4th component
Fourier Transform 4th component		Fourier Transform 5th component
Fourier Transform 5th component		Fourier Transform 7th component
Fourier Transform 7th component		Fourier Transform 8th component
Fourier Transform 8th component		Fourier Transform 9th component
		Fourier Transform 10th component

Fonte: Próprio autor.

Tabela 4.7: Características que constituem os subconjuntos do Cenário C na classificação utilizando SVM.

SBS	**PCA**	**GA**
Bytes per second Mean	Bytes per second Minimum value	Bytes per second Mean
Bytes per second Variance	Packets per second Mean	Bytes per second Variance
Bytes per second Maximum value	Packets per second Variance	Bytes per second Minimum value
Packet Length Mean	Packet Length Mean	Packets per second Variance
Packet Length Variance	Packet Length Variance	Packet Length Mean
Packet Length Minimum value	Packet Length Minimum value	Packet Length Variance
Packet inter-arrival time Maximum value	Packet inter-arrival time Maximum value	Packet Length Maximum value
Packet inter-arrival time Minimum value	Flow Size in packets	Packet inter-arrival time Mean
Flow Size in Bytes	Packet Length 1st quartile	Packet inter-arrival time Variance
Packet inter-arrival time 1st quartile	Packet Length 3rd quartile	Packet inter-arrival time Minimum value
Packet inter-arrival time 3rd quartile	Fourier Transform 10th component	Flow Size in Bytes
Packet Length 1st quartile		Packet Length 1st quartile
Fourier Transform 2nd component		Packet Length 3rd quartile
Fourier Transform 6th component		Fourier Transform 2nd component
Fourier Transform 9th component		Fourier Transform 3rd component
		Fourier Transform 6th component
		Fourier Transform 9th component
		Fourier Transform 10th component

Fonte: Próprio autor.

Tabela 4.8: Características que constituem os subconjuntos do Cenário A na classificação utilizando K-means.

SBS	**PCA**	**GA**
Packets per second Mean	Packet inter-arrival time Mean	Bytes per second Minimum value
Packets per second Variance	Packet inter-arrival time Variance	Packets per second Variance
Packets per second Maximum value	Packet inter-arrival time Minimum value	Packet Length Variance
Packets per second Minimum value	Flow Size in Bytes	Packet Length Maximum value
Packet Length Mean	Fourier Transform 1st component	Packet Length Minimum value
Packet Length Variance	Fourier Transform 2nd component	Packet inter-arrival time Mean
Packet Length Maximum value	Fourier Transform 3rd component	Flow duration
Packet Length Minimum value	Fourier Transform 4th component	Flow Size in Bytes
Packet inter-arrival time Mean	Fourier Transform 5th component	Packet inter-arrival time 1st quartile
Packet inter-arrival time Variance	Fourier Transform 6th component	Packet Length 1st quartile
Packet inter-arrival time Maximum value	Fourier Transform 9th component	Fourier Transform 7th component
Packet inter-arrival time Minimum value	Fourier Transform 10th component	Fourier Transform 10th component
Flow duration		
Flow Size in Bytes		
Packet inter-arrival time 1st quartile		
Packet inter-arrival time 3rd quartile		
Packet Length 1st quartile		
Packet Length 3rd quartile		

Fonte: Próprio autor.

Tabela 4.9: Características que constituem os subconjuntos do Cenário B na classificação utilizando K-means.

SBS	**PCA**	**GA**
Bytes per second Maximum value	Bytes per second Mean	Bytes per second Maximum value
Packet Length Mean	Bytes per second Variance	Packet Length Mean
Packet Length Variance	Bytes per second Maximum value	Packet Length Variance
Packet Length Minimum value	Bytes per second Minimum value	Packet inter-arrival time 3rd quartile
Packet inter-arrival time Mean	Packets per second Mean	Fourier Transform 10th component
Packet inter-arrival time Variance	Packets per second Variance	
Packet inter-arrival time Maximum value	Packets per second Maximum value	
Packet inter-arrival time Minimum value	Packet Length Mean	
Flow duration	Packet Length Variance	
Packet inter-arrival time 1st quartile	Packet Length Maximum value	
Packet inter-arrival time 3rd quartile	Packet inter-arrival time 3rd quartile	
Packet Length 3rd quartile	Packet Length 1st quartile	
Fourier Transform 3rd component	Fourier Transform 10th component	
Fourier Transform 5th component		
Fourier Transform 6th component		
Fourier Transform 7th component		
Fourier Transform 9th component		
Fourier Transform 10th component		

Fonte: Próprio autor.

Tabela 4.10: Características que constituem os subconjuntos do Cenário C na classificação utilizando K-means.

SBS	**PCA**	**GA**
Packets per second Minimum value	Bytes per second Maximum value	Bytes per second Mean
Packet Length Variance	Packets per second Maximum value	Packets per second Minimum value
Packet inter-arrival time Mean	Packet inter-arrival time Mean	Packet Length Mean
Packet inter-arrival time Variance	Packet inter-arrival time Variance	Packet Length Variance
Packet inter-arrival time Minimum value	Fourier Transform 10th component	Packet Length Maximum value
Flow duration		Packet inter-arrival time Mean
Flow Size in Bytes		Packet inter-arrival time Variance
Packet inter-arrival time 1st quartile		Packet inter-arrival time Minimum value
		Flow duration
		Packet inter-arrival time 1st quartile
		Packet inter-arrival time 3rd quartile
		Packet Length 1st quartile
		Fourier Transform 1st component
		Fourier Transform 3rd component
		Fourier Transform 5th component
		Fourier Transform 6th component

Fonte: Próprio autor.

5 CONCLUSÃO

Neste trabalho foi desenvolvido um sistema capaz de criar um conjunto de características de fluxo avançadas a partir da extensão de dois contadores oferecidos pelo protocolo OpenFlow. Além disso, foram avaliados três algoritmos para seleção de características, SBS, PCA e Algoritmo Genético, criando subconjuntos de características utilizados para aprimorar a classificação de fluxos de tráfego. Por último, foram analisadas as influências destes subconjuntos para a classificação fluxos de tráfego utilizando dois algoritmos diferentes: SVM e K-means.

A arquitetura do sistema desenvolvido tira proveito da separação dos planos de Controle e de Encaminhamento, que caracteriza as Redes Definidas por Software, com objetivo de simplificar a coleta de informações sobre o tráfego da rede. Também possui sua implementação realizada no Plano de Aplicações, podendo ser adicionada a qualquer controlador sem necessidade de modificações. Além disso, a arquitetura modular permite que esta seja refinada para se adaptar a situações específicas, por meio da inserção de técnicas diferentes para a seleção de características ou para a classificação de fluxos de tráfego.

Tendo em vista que todos os experimentos realizados neste trabalho são baseados em características criadas à partir de contadores oferecidos pelo protocolo OpenFlow, pode-se concluir que os contadores oferecidos pelo protocolo são eficazes na criação e extensão de um conjunto de características que descrevem os fluxos de tráfego.

Outro ponto a ser observado é que subconjuntos com características diferentes apresentam resultados semelhantes. Isso prova que não apenas as características individualmente devem ser consideradas, mas que também devem ser analisadas as relações que elas possuem umas com as outras para a definição de um subconjunto ótimo de características para a classificação de fluxos de tráfego.

Os resultados obtidos com a execução dos experimentos demonstram que a solução desenvolvida pode ser aplicada em contextos que possuem perfis de tráfego semelhante aos utilizados neste trabalho e que necessitam de classificação de fluxos de tráfegos. Isso se deve ao fato de que na grande maioria dos cenários é apresentada uma evolução na qualidade da classificação.

Baseando-se nos resultados obtidos, também é possível concluir que os algoritmos de seleção de características analisados são adequados para a classificação utilizando o SVM, porém nem todos são adequados para a clusterização realizada pelo K-means. Por exemplo, o tempo de execução do Algoritmo Genético faz com que este não seja recomendável para ser utilizado juntamente com o K-means, tornando o SBS e o PCA mais indicados, pois são mais

rápidos e apresentam resultados semelhantes.

Possíveis extensões que podem ser adicionadas a este trabalho incluem: utilização de tráfego real e não apenas tráfego sinteticamente criado; pesquisar novas características por meio do estudo dos demais contadores oferecidos pelo OpenFlow, por exemplo contador de descarte de pacotes; uma segunda estratégia para a criação de novas características pode ser por meio da análise dos fluxos como sendo bidirecionais, e não apenas unidirecionais, como os que fazem parte do escopo deste trabalho. Um outro aspecto que pode ser explorado com esta mesma arquitetura é precisão na identificação de ataques em um cenário que possui diversos perfis de tráfego, e não apenas dois perfis de tráfego como é feito nos experimentos deste trabalho.

REFERÊNCIAS

AULD, T.; MOORE, A.; GULL, S. Bayesian neural networks for internet traffic classification. **Neural Networks, IEEE Transactions on**, v. 18, n. 1, p. 223–239, Jan 2007. ISSN 1045-9227.

BENNETT, K. P.; CAMPBELL, C. Support vector machines: Hype or hallelujah? **SIGKDD Explor. Newsl.**, ACM, New York, NY, USA, v. 2, n. 2, p. 1–13, dec. 2000. ISSN 1931-0145. Available from Internet: <http://doi.acm.org/10.1145/380995.380999>.

BLAKE, S. et al. **RFC2475 - An Architecture for Differentiated Services**. [S.l.], 1998. 35 p. Available from Internet: <https://tools.ietf.org/html/rfc2475>.

BLUM, A. L.; LANGLEY, P. Selection of relevant features and examples in machine learning. **Artif. Intell.**, Elsevier Science Publishers Ltd., Essex, UK, v. 97, n. 1-2, p. 245–271, dec. 1997. ISSN 0004-3702. Available from Internet: <http://dx.doi.org/10.1016/S0004-3702(97)00063-5>.

FEAMSTER, N.; REXFORD, J.; ZEGURA, E. The road to sdn. **Queue**, ACM, New York, NY, USA, v. 11, n. 12, p. 20:20–20:40, dec. 2013. ISSN 1542-7730. Available from Internet: <http://doi.acm.org/10.1145/2559899.2560327>.

GUDE, N. et al. Nox: Towards an operating system for networks. **SIGCOMM Comput. Commun. Rev.**, ACM, New York, NY, USA, v. 38, n. 3, p. 105–110, jul. 2008. ISSN 0146-4833. Available from Internet: <http://doi.acm.org/10.1145/1384609.1384625>.

GUYON, I.; ELISSEEFF, A. An introduction to variable and feature selection. **J. Mach. Learn. Res.**, JMLR.org, v. 3, p. 1157–1182, mar. 2003. ISSN 1532-4435. Available from Internet: <http://dl.acm.org/citation.cfm?id=944919.944968>.

HALEPLIDIS, E. et al. **RFC7426 - Software-Defined Networking (SDN): Layers and Architecture Terminology**. [S.l.], 2015. 35 p. Available from Internet: <https://tools.ietf.org/html/rfc7426>.

HAUPT, R. L.; HAUPT, S. E. **Practical Genetic Algorithms**. New York, NY, USA: John Wiley & Sons, Inc., 1998. ISBN 047-1188735.

HU, B.; SHEN, Y. Machine learning based network traffic classification: A survey. JICS, Hong Kong, v. 9, n. 11, p. 3161–3170, oct. 2012. ISSN 1548–7741. Available from Internet: <http://www.joics.com/publishedpapers/2012_9_11_3161_3170.pdf>.

JAIN, A. K. Data clustering: 50 years beyond k-means. **Pattern Recognition Letters**, v. 31, n. 8, p. 651 – 666, 2010. ISSN 0167-8655. Award winning papers from the 19th International Conference on Pattern Recognition (ICPR)19th International Conference in Pattern Recognition (ICPR). Available from Internet: <http://www.sciencedirect.com/science/article/pii/S0167865509002323>.

JOLLIFFE, I. **Principal Component Analysis**. [S.l.]: Springer Verlag, 1986.

LANGLEY; SAGE, S. Oblivious decision trees and abstract cases. In: **Proc. AAAI-94 Workshop on case-based reasoning**. [s.n.], 1994. p. 113–117. Available from Internet: <http://www.isle.org/~langley/papers/oblivion.cbr94.ps.gz>.

LIMONCELLI, T. A. Openflow: A radical new idea in networking. **Queue**, ACM, New York, NY, USA, v. 10, n. 6, p. 40:40–40:46, jun. 2012. ISSN 1542-7730. Available from Internet: <http://doi.acm.org/10.1145/2246036.2305856>.

MACQUEEN, J. Some methods for classification and analysis of multivariate observations. In: **Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics**. Berkeley, Calif.: University of California Press, 1967. p. 281–297. Available from Internet: <http://projecteuclid.org/euclid.bsmsp/1200512992>.

MCKEOWN, N. et al. Openflow: Enabling innovation in campus networks. **SIGCOMM Comput. Commun. Rev.**, ACM, New York, NY, USA, v. 38, n. 2, p. 69–74, mar. 2008. ISSN 0146-4833. Available from Internet: <http://doi.acm.org/10.1145/1355734.1355746>.

MOORE, A. W.; ZUEV, D. Internet traffic classification using bayesian analysis techniques. In: **Proceedings of the 2005 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems**. New York, NY, USA: ACM, 2005. (SIGMETRICS '05), p. 50–60. ISBN 1-59593-022-1. Available from Internet: <http://doi.acm.org/10.1145/1064212.1064220>.

MUKKAMALA, S.; SUNG, A. Detecting denial of service attacks using support vector machines. In: **Fuzzy Systems, 2003. FUZZ '03. The 12th IEEE International Conference on**. [S.l.: s.n.], 2003. v. 2, p. 1231–1236 vol.2.

NGUYEN, T.; ARMITAGE, G. A survey of techniques for internet traffic classification using machine learning. **Communications Surveys Tutorials, IEEE**, v. 10, n. 4, p. 56–76, Fourth 2008. ISSN 1553-877X.

NUNES, B. et al. A survey of software-defined networking: Past, present, and future of programmable networks. **Communications Surveys Tutorials, IEEE**, v. 16, n. 3, p. 1617–1634, Third 2014. ISSN 1553-877X.

OH, I.-S.; LEE, J.-S.; MOON, B.-R. Hybrid genetic algorithms for feature selection. **Pattern Analysis and Machine Intelligence, IEEE Transactions on**, v. 26, n. 11, p. 1424–1437, Nov 2004. ISSN 0162-8828.

ONF. **Software Defined Netwoking: The New Norm for Networks**. 2012. 12 p. Available from Internet: <https://www.opennetworking.org>.

ONF. **OpenFlow Switch Specification**. 2014. Available from Internet: <https://www.opennetworking.org/images/stories/downloads/sdn-resources/onf-specifications/openflow/openflow-switch-v1.5.0.noipr.pdf>.

PEARSON, K. Liii. on lines and planes of closest fit to systems of points in space. **Philosophical Magazine Series 6**, v. 2, n. 11, p. 559–572, 1901. Available from Internet: <http://dx.doi.org/10.1080/14786440109462720>.

ROUSSEEUW, P. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. **J. Comput. Appl. Math.**, Elsevier Science Publishers B. V., Amsterdam, The Netherlands, The Netherlands, v. 20, n. 1, p. 53–65, nov. 1987. ISSN 0377-0427. Available from Internet: <http://dx.doi.org/10.1016/0377-0427(87)90125-7>.

SAEYS, Y.; INZA, I.; LARRAñAGA, P. A review of feature selection techniques in bioinformatics. **Bioinformatics**, v. 23, n. 19, p. 2507–2517, 2007. Available from Internet: <http://bioinformatics.oxfordjournals.org/content/23/19/2507.abstract>.

SANTOS, A. S. et al. Identification and selection of flow features for accurate traffic classification in sdn. **14th IEEE International Symposium on Network Computing and Applications (NCA 2015)**, p. 137–141, September 2015.

TOOTOONCHIAN, A.; GANJALI, Y. Hyperflow: A distributed control plane for openflow. In: **Proceedings of the 2010 Internet Network Management Conference on Research on Enterprise Networking**. Berkeley, CA, USA: USENIX Association, 2010. (INM/WREN'10), p. 3–3. Available from Internet: <http://dl.acm.org/citation.cfm?id=1863133.1863136>.

VAPNIK, V. **Estimation of Dependences Based on Empirical Data: Springer Series in Statistics (Springer Series in Statistics)**. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 1982. ISBN 0387907335.

WICKBOLDT, J. et al. Software-defined networking: management requirements and challenges. **Communications Magazine, IEEE**, v. 53, n. 1, p. 278–285, January 2015. ISSN 0163-6804.

XU, R.; WUNSCH D., I. Survey of clustering algorithms. **Neural Networks, IEEE Transactions on**, v. 16, n. 3, p. 645–678, May 2005. ISSN 1045-9227.

YU, L.; LIU, H. Feature selection for high-dimensional data: A fast correlation-based filter solution. In: . [S.l.: s.n.], 2003. p. 856–863.