Pós-Graduação em Ciência da Computação

PETRÔNIO GOMES LOPES JÚNIOR

Classificação de Tráfego Distribuída: Construindo uma Arquitetura baseada em Redes Virtualizadas

Universidade Federal de Pernambuco
posgraduacao@cin.ufpe.br
www.cin.ufpe.br/~posgraduacao

RECIFE
2017

Petrônio Gomes Lopes Júnior

Classificação de Tráfego Distribuída: Construindo uma Arquitetura baseada em Redes Virtualizadas

Este trabalho foi apresentado à Pós-Graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito parcial para obtenção do grau de Doutor em Ciência da Computação.

ORIENTADOR: **Prof. Dr. Stênio Flávio de Lacerda Fernandes**

RECIFE
2017

Catalogação na fonte

Bibliotecária Monick Raquel Silvestre da S. Portes, CRB4-1217

L864c Lopes Júnior, Petrônio Gomes

Classificação de tráfego distribuída: construindo uma arquitetura baseada em redes virtualizadas / Petrônio Gomes Lopes Júnior. – 2017.

168 f.:il., fig., tab.

Orientador: Stênio Flávio de Lacerda Fernandes.

Tese (Doutorado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2017.

Inclui referências.

1. Redes de computadores. 2. Classificação de tráfego. I. Fernandes, Stênio Flávio de Lacerda (orientador). II. Título.

004.6 CDD (23. ed.) UFPE- MEI 2017-235

Petrônio Gomes Lopes Júnior

Classificação de Tráfego Distribuída: Construindo uma Arquitetura baseada em Redes Virtualizadas

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Doutor em Ciência da Computação.

Aprovado em: 08/03/2017.

_____

Orientador: Prof. Dr. Stênio Flávio de Lacerda Fernandes

BANCA EXAMINADORA

_____

Prof. Dr. José Augusto Suruagy Monteiro
Centro de Informática / UFPE

_____

Prof. Dr. Nelson Souto Rosa
Centro de Informática / UFPE

_____

Prof. Dr. Ricardo Martins de Abreu Silva
Centro de Informática / UFPE

_____

Prof. Dr. Arthur de Castro Callado
Universidade Federal do Ceará – Campus Quixadá

_____

Prof. Dr. Antônio Marinho Pilla Barcellos
Instituto de Informática/ UFRGS

À minha família.

Agradecimentos

Primeiramente, agradeço a Deus por todas as oportunidades que tive. Sem Ele, nenhuma das realizações em minha vida seriam possíveis.

Agradeço à minha esposa, Mirella, que me deu todo suporte que precisava para conduzir este trabalho, sendo paciente e compreensiva diante de todas renúncias que foram necessárias.

Agradeço aos meus pais, Petrônio e Ivana, que são as pessoas responsáveis por quem eu sou e por tudo que pude conquistar ao longo de minha vida.

Agradeço à minha tia e madrinha, Isley, que cuidou de mim da melhor forma possível ao longo da minha vida.

Gostaria de agradecer aos meus irmãos, Priscilla (in memoriam) e Guto, que me ensinaram bastante, cada um a sua maneira, durante esta jornada.

Agradeço aos meus demais familiares, em especial, a meu avô Estácio, que sempre foi um exemplo de superação para mim.

Agradeço a Stênio, meu orientador, por todo o auxílio prestado e orientação para que este trabalho fosse concluído.

Por fim, agradeço a todos aqueles que, de alguma forma, contribuíram para que esta tese fosse finalizada.

"Combati o bom combate, terminei a minha carreira, guardei a fé."

2 Timóteo 4:7

Resumo

Prover um gerenciamento eficiente de uma rede é uma tarefa intimamente ligada ao conhecimento sobre as informações que compõem o tráfego da rede. Tradicionalmente, um ponto único na rede é utilizado para realizar a classificação de tráfego. Além disso, o método de classificação escolhido, bem como o *hardware* de suporte são os focos das melhorias nessa área de pesquisa. Entretanto, existem outras questões de pesquisa em aberto nesse campo. Esta tese propõe e avalia uma arquitetura distribuída para classificação de tráfego da rede, sendo capaz de prover classificação *online,* resiliência a falhas e mecanismos adaptativos para lidar com mudanças no perfil de tráfego. A arquitetura proposta é baseada nas características de redes SDN (*Software-Defined Networking*) e NFV (*Network Function Virtualization*), possibilitando uma abordagem de classificação completamente distribuída. Os classificadores são tratados como funções virtuais da rede que colaboram entre si. Dessa forma, através da proposição de um algoritmo de posicionamento de classificadores (denominado Posicionamento Ideal de Classificadores Otimizado - PICO), a tarefa de indicar onde classificadores virtuais devem ser instanciados em uma determinada topologia é um problema abordado neste trabalho, além da tarefa da classificação propriamente dita. Os resultados demonstram que o módulo de classificação da arquitetura proposta alcança melhorias em sua acurácia quando comparado à classificação tradicional. Além disso, o mecanismo de adaptação a mudanças no perfil de tráfego proposto provê um ganho de aproximadamente 14,5% de acurácia em comparação à abordagem tradicional. Adicionalmente, quando avaliando o PICO, é possível obter um posicionamento até 21% superior ao método aleatório do ponto de vista da cobertura de caminhos da topologia. Por fim, o tempo necessário para instanciar um classificador virtual também é avaliado, demonstrando a capacidade de prover uma rápida instanciação.

Palavras-chave: Classificação de tráfego. Classificação baseada em fluxos. Software-Defined Network. Análise de tráfego. Redes de computadores. Network Function Virtualization.

Abstract

An efficient network management is highly related to the knowledge about the data transported in network packets. Traditionally, a single point in the network is used to performing the network traffic classification. Further, the classification method and the supporting hardware are the focus of the improvements in this field of research. However, there are open research challenges in this area. This thesis proposes and evaluates a distributed architecture for network traffic classification, which is capable of providing online classification, fault resilience, and adaptive mechanisms to deal with traffic profile changes. The proposed architecture is based on several features provided by the Software-Defined Networking (SDN) and Network Function Virtualization (NFV) paradigms, enabling the deployment of a complete distributed approach in an effective way. The traffic classifiers are virtual network functions, which cooperate with each other. In this context, a proposed algorithm (called PICO) provides the placement of the virtual classifiers at a network topology. Results show that the classification module of the proposed architecture enhances the accuracy of the classification considering cooperative approaches, overcoming the traditional classification with an isolated classifier. Furthermore, the adaptive mechanism provides an increase of the accuracy compared to the traditional training method in about 14.5%. Additionally, the placement of the virtual network traffic classifiers provides a high rate of path coverage, overcoming the random approach. The PICO algorithm enhances the path coverage up to 21%. Finally, we study the deployment time to achieve a quick start of the classification.

Keywords: Traffic classification. Flow-based traffic classification. Software-Defined Network. Traffic analysis. Computer network. Network Function Virtualization.

Lista de Figuras

Lista de Tabelas

Abreviações e Acrônimos

CapEx	Capital Expenditures
DCM	Distributed Colaborative Monitoring
DSC	Dempster-Shafer Combination
DPI	Deep Packet Inspection
EDSC	Enhanced Dempster-Shafer Combination
FPTAS	Fully Polynomial Time Approximation Scheme
HTTP	The Hypertext Transfer Protocol
IDS	Intrusion Detection System
IP	Internet Protocol
GC	Grau de Confiança
GPU	Graphic Processing Unit
GSDT	GPU-based Streaming Decision Tree
ISP	Internet Service Provider
MILP	Mixed Integer Linear Programming
ML	Machine Learning
MLC	Maximum Likelihood Combination
MLP	MultiLayer Perceptron
NFV	Network Function Virtualization
NFVI	Network Function Virtualization Infraestructure
nmeta	Network Metadata
NOS	Network Operating System
OpEx	Operating Expenditures
OE	Objetivos Específicos
OSPF	Open Shortest Path First
PIC	Posicionamento Ideal de Classificadores
P2P	Peer-to-Peer
QoE	Quality of Experience
QoS	Quality of Service
QP	Questões de Partida
RC	Random Combination
SCADA	Supervisory Control and Data Acquisition
SCID	SCADA Intrusion Detection

SDN	Software-Defined Network
SDNFV	Software-Defined Network Function Virtualization
SIMD	Single Instruction Multiple Data
SQL	Structured Query Language
TCAM	Ternary Content Addressable Memory
TLS	Transport Layer Security
VS	Votação Simples

Sumário

1 Introdução

Este capítulo fornece uma visão geral do problema identificado e estudado nesta tese, expondo os relacionamentos entre os diferentes conceitos e tecnologias utilizados para concepção deste trabalho. Além disso, são apresentados hipótese, questões de partida, objetivo geral e objetivos específicos. Por fim, um resumo dos resultados obtidos é apresentado.

1.1 Motivação e Contextualização

Desde o advento da Internet, sua importância é crescente em diversas áreas de conhecimento, suportando a utilização de ferramentas ou funcionalidades como *email*, comunicação sobre IP ou mesmo distribuição de dados (MAIER et al., 2009). Do ponto de vista do gerenciamento (FINAMORE et al., 2011), é essencial conhecer o tipo de informação que trafega na rede a fim de prover QoS, segurança e escalabilidade, por exemplo. Algumas atividades dependentes dessa classificação são interessantes como, por exemplo, posicionamento de servidores de *cache* de conteúdo, bloqueio de aplicações indesejadas (segurança) ou mesmo limitação da banda disponível para determinado tipo de tráfego. A classificação de tráfego surgiu nesse contexto como opção para que os provedores de serviço pudessem identificar o tráfego em sua rede (DAINOTTI et al., 2012). A acurácia com que essa classificação é realizada é extremamente importante para que o gerencimanento seja efetivo de fato (SOYSAL et al., 2010). No entanto, esse não é o único aspecto a ser avaliado no cenário de classificação de tráfego. Além da acurácia, o desempenho computacional da abordagem utilizada, bem como a sobrecarga de processamento imposta à rede em que está aplicada, são essenciais para que a classificação de tráfego seja aplicada em um cenário real (MELO et al., 2014) (LOPES et al., 2014) (SUTHAHARAN, 2014). É necessário um balanceamento dessas características para que a classificação atenda ao seu propósito sem que comprometa o desempenho do tráfego de pacotes na rede. Em outras palavras, o ajuste entre a capacidade de processamento e seu custo precisa ser realizado para que a classificação alcance um nível ótimo de utilização de recursos e operação. Em geral, a classificação de tráfego é realizada em um ponto único de coleta de tráfego - geralmente a borda da rede (JIANG et al., 2007), exemplificado na

Figura 1.1, e as melhorias que podem ser aplicadas à classificação são realizadas em relação ao método de classificação (dentro do classificador) e/ou ao *hardware* utilizado (FERNANDES et al., 2009) (LOPES et al., 2014) (MELO et al., 2014) (SANTOS et al., 2013) (SMITH et al., 2009).

Um ponto único e central de classificação pode gerar alguns problemas pouco investigados pela literatura atual, como a existência de um único ponto de falha, o monitoramento do tráfego que sai da rede por um nó específico apenas e a imposição de um gargalo ao tráfego (RANTALA et al., 2008) (SCHUSTER et al., 2012).

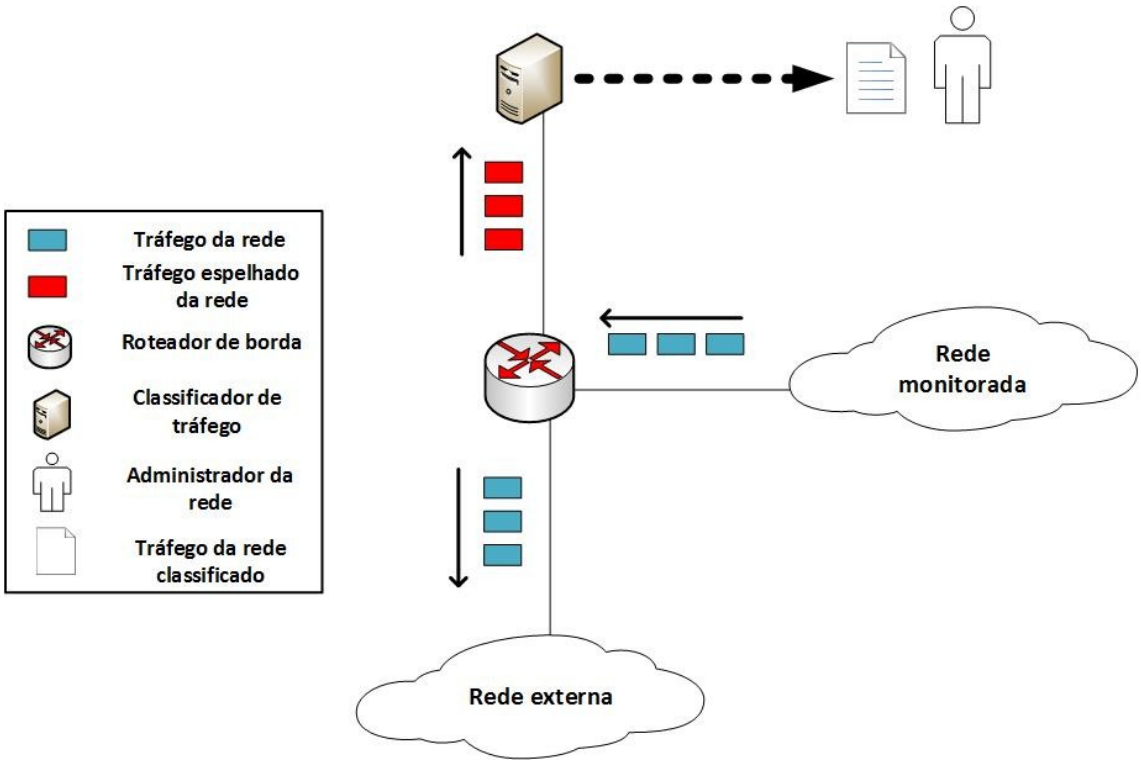


Figura 1.1 – Classificação realizada da maneira tradicional

A partir do momento em que a forma de interação entre a classificação e a rede é analisada e definida (arquitetura da classificação, ou seja, se há um ponto central de classificação e onde a coleta do tráfego é realizada, por exemplo), a próxima etapa é a definição da interação do classificador propriamente dito com o tráfego da rede. Esse tráfego pode ser observado com diferentes níveis de granularidade. De maneira geral, na literatura, são descritos três métodos para realização da classificação de tráfego (CALLADO et al., 2009) (RICHTER et al., 2015) (ZHANG et al., 2015):

1. Classificação baseada em portas (BERNAILLE et al., 2006);

2. Classificação baseada em pacotes (MELO et al., 2014) (CASCARANO et al., 2009) (FINSTERBUSCH et al., 2014), e;
3. Classificação baseada em fluxos (LOPES et al., 2014) (NGUYEN et al., 2008) (SANTOS et al., 2013);

Alguns estudos apontam um quarto método para realizar a classificação de tráfego. Essa abordagem se baseia na interação do usuário com a rede, utilizando seu comportamento para tornar a classificação efetiva (RICHTER et al., 2015). Apenas os métodos baseados no tráfego propriamente dito são relevantes para este trabalho, já que o quarto método depende de informações adicionais ao tráfego da rede.

A primeira abordagem descreve a classificação do tráfego através da utilização de portas bem conhecidas por aplicações específicas. Embora o desempenho computacional dessa técnica seja muito bom, a precisão não costuma ser alta, uma vez que as aplicações podem, intencionalmente, utilizar portas destinadas a outras para que o tráfego não seja identificado. Por exemplo, quando uma aplicação utiliza a porta 80, em teoria, o tráfego deveria ser classificado como *Web*[1]. No entanto, outras aplicações, como o BitTorrent[2] (ou outra aplicação que pretenda passar despercebida pela administração da rede), podem fazer uso dessa mesma porta para que seu tráfego não seja identificado, fazendo com que o administrador da rede possa identificar o tráfego de maneira errônea (KARAGIANNIS et al., 2005). Além disso, cerca de 10% do tráfego *Web* não é direcionado à porta 80 (RICHTER et al., 2015). Por fim, é importante destacar que a manutenção de regras e configurações por portas é uma atividade insustentável para o administrador da rede (NG et al., 2015).

Já a classificação baseada em pacotes pode ser bastante precisa, mas, em geral, é uma técnica custosa do ponto de vista computacional (ALIZADEH et al., 2015). O principal método baseado em pacotes é a inspeção profunda de pacotes (em inglês *Deep Packet Inspection* – DPI). A inspeção de pacotes se baseia em uma lista pré-definida de assinaturas. A classificação consiste na busca de uma ou mais assinaturas no conteúdo do pacote. Além disso, é essencial que a lista de assinaturas seja mantida atualizada. Outra questão surge quando o classificador é um DPI e o tráfego a ser classificado está encriptado, nesse caso, a inspeção de pacotes não é capaz de identificá-lo. Uma aplicação

[1] The Internet Assigned Numbers Authority, IANA. Disponível em: http://www.iana.org/. Acessado em Março de 2017.
[2] Disponível em http://www.bittorrent.com/. Acessado em Março de 2017.

bastante difundida que tem seu tráfego encriptado é o Skype[3]. Nesse caso, mesmo um DPI que apresente uma base de assinaturas robusta e atualizada não seria capaz de detectar o tráfego gerado pelo Skype (BONFIGLIO et al., 2007). Outra questão bastante relevante na inspeção de pacotes é a legislação em alguns países, que restringe a análise dos pacotes (a inspeção de seu conteúdo) e, consequentemente, sua classificação (ALIZADEH et al., 2015) (RIZZI et al., 2015) (TURNBULL et al., 2009).

Por fim, a classificação baseada em fluxos é geralmente realizada através de técnicas de Aprendizagem de Máquina (em inglês*, Machine Learning* – ML), que apresentam um bom desempenho e boa precisão, mas dependem da qualidade do conjunto de treinamento que será utilizado. Adicionalmente, cada fluxo apresenta um conjunto de características que são coletadas na rede a critério do seu administrador (LOPES et al., 2014). A partir desse momento, o algoritmo de classificação será aplicado ao tráfego coletado. Nele, as características avaliadas serão as do fluxo, que podem ser, por exemplo, o número de *bytes* acumulados, o número de pacotes coletados, o tempo médio de intervalo entre pacotes e o número da porta de origem da aplicação. Esses conjuntos de características são utilizados para que a classificação mais apropriada seja atribuída ao fluxo. Além disso, eles podem variar as características dos fluxos que são armazenadas de acordo com a coleta realizada na rede, com a definição do administrador da rede ou com a capacidade de armazenamento do ambiente que está sendo monitorado (GRINGOLI et al., 2009) (SOYSAL et al., 2010) (ZHANG et al., 2015).

Analisando o cenário atual da classificação de tráfego, é possível observar que existem técnicas capazes de prover resultados satisfatórios para diferentes situações isoladamente, o que ficará mais evidente no Capítulo 3 deste trabalho. No entanto, uma mudança na arquitetura tradicionalmente utilizada para aplicação de técnicas voltadas à classificação de tráfego de redes pode proporcionar ganhos de acurácia aliados à minimização ou resolução de problemas inerentes às propostas mencionadas, como a imposição de um único gargalo na rede. Nesse contexto, técnicas, metodologias ou paradigmas utilizados e aplicados em outros campos da área de redes de computadores podem fornecer o suporte necessário para concepção de uma nova arquitetura voltada para a classificação de tráfego. Essa linha de pesquisa que propõe uma quebra entre a classificação de tráfego e o paradigma utilizado atualmente não é explorada em trabalhos voltados para essa área de pesquisa.

[3] Disponível em http://www.skype.com. Acessado em Março de 2017.

Quando se descreve o contexto em que ocorre a classificação de tráfego, fica claro que as redes IP tradicionais são, de uma maneira geral, bastante complexas e difíceis de lidar, sendo, consequentemente, de difícil gerenciamento. Nessas redes, reconfigurações e mecanismos de respostas automáticos são virtualmente inexistentes no cenário atual (KREUTZ et al., 2015). Além disso, estudos recentes destacam duas características essenciais ao monitoramento de tráfego, o balanceamento de carga (para lidar com problemas de escalabilidade) e o monitoramento por fluxo (em que diferentes ações são tomadas para diferentes fluxos) (YU et al., 2014). *Software-Defined Network* (SDN) (MCKEOWN, 2011) e *Network Function Virtualization* (NFV) (HAN et al., 2015) têm conceitos bastante difundidos atualmente e, diante do cenário atual da utilização da classificação de tráfego, podem suportar o gerenciamento de redes sob uma nova perspectiva.

De acordo com (KREUTZ et al., 2015), SDN é um paradigma que pretende modificar o modelo vigente de integração vertical, separando o controle lógico da rede dos seus ativos (roteadores e *switches*), centralizando o seu controle e permitindo a sua programação. Em outras palavras, um importante princípio do paradigma *Software-Defined Networking* é a separação de tarefas. As políticas da rede, a sua implementação em *hardware* e o encaminhamento do tráfego representam camadas distintas. Nesse contexto, o paradigma SDN levanta a possibilidade de superar as limitações da infraestrutura atual. A existência de um plano de controle para gerenciar operações de classificadores e a possibilidade de utilizar múltiplos classificadores é bastante interessante para o cenário atual de redes de computadores. É muito relevante destacar que esse modelo lógico centralizado não implica em um sistema centralizado (KOPONEN et al., 2010) e, portanto, não impede que o conceito de SDN esteja associado a uma arquitetura distribuída (significa dizer que a própria estrutura da SDN, em especial o plano de controle, pode ser distribuída ou não). Embora venham sendo cada vez mais difundidos, a maior parte dos conceitos pertencentes ao paradigma SDN não são propriamente novos (FEAMSTER et al., 2013). O paradigma SDN permite que soluções baseadas em virtualização sejam utilizadas para suprir funções específicas que são desejáveis em uma rede. A separação do plano de controle e plano de encaminhamento simplifica a integração entre diferentes aplicações (CASADO et al., 2014).

Adicionalmente, ao observar a infraestrutura atual de redes IP, percebe-se que o custo de implantação e manutenção de novas funcionalidades é significativo. Isso limita a inovação e adição de novas características e novos serviços. Diante desse cenário, é comum a utilização de *middleboxes* (componentes especializados que são inseridos na rede) para

suprir tais limitações (KREUTZ et al., 2015). Essa abordagem minimiza os custos associados à implementação de novas funcionalidades em uma rede, embora não sejam facilmente manipuláveis ou de simples instalação. *Firewalls,* sistemas de detecção de intrusão e classificadores baseados na inspeção de pacotes, por exemplo, podem operar como componentes especializados em uma rede específica. No entanto, o uso de *middleboxes* implica no crescimento considerável da complexidade de uma rede, tanto do ponto de vista operacional como de *design* (KREUTZ et al., 2015). Diante desse cenário, NFV implementa funções virtuais através de técnicas de virtualização de *software*, além de ser executada no *hardware* de elementos da rede. Nesse caso, as ferramentas e funções virtuais podem ser instanciadas sob demanda sem a necessidade de instalar novos equipamentos.

Embora apresentem características interessantes, é importante destacar que, do ponto de vista da classificação de tráfego, SDN e NFV são muito pouco utilizados, aparecendo em poucos trabalhos e, aparentemente, de forma inicial (AGARWAL et al., 2013) (BREMLER-BARR et al., 2014) (PEREZ et al., 2014) (YU et al., 2014). De acordo com (CUI et al., 2012), NFV e SDN são complementares embora não sejam dependentes um do outro. Como dito anteriormente, os conceitos de NFV propõem a virtualização de aplicações ou funções individuais enquanto os conceitos de SDN tratam da virtualização do controle lógico da rede. Fica evidente que os conceitos são compatíveis, resultando em um cenário em que controladores SDN podem gerenciar a configuração e a instanciação de funções NFV na rede. Em síntese, o controlador SDN gerencia a operação da rede (encaminhamento de pacotes e o momento de utilizar uma determinada *middlebox*), enquanto as funções executadas na rede (como um *firewall,* por exemplo) são virtualizadas através de NFV. Por exemplo, a orquestração e a flexibilidade providas pelos paradigmas SDN e NFV possibilitam a instanciação de funções virtuais na rede e a colaboração entre elas. Juntos, esses paradigmas são chamados de *Software-Defined Network Function Virtualization* (SDNFV) (LI et al., 2015). Nesse caso, NFV é responsável pelo provisionamento de funções virtuais da rede, enquanto SDN fornece uma visão global da rede e controle do tráfego da rede (CUI et al., 2012).

É importante notar que existem diversas questões de pesquisa em aberto quando se analisa o cenário atual de classificação de tráfego. Várias abordagens foram propostas com o objetivo de solucionar diferentes questões, como a melhoria da capacidade de processamento da classificação através de *hardware* (SZABÓ et al., 2010), proposição de algoritmos mais eficazes e eficientes (MELO et al., 2014) (LOPES et al., 2014), combinação de diferentes algoritmos para melhorar a precisão da classificação (CALLADO et al., 2010)

e uso de DPI associado a NFV e SDN (BREMLER-BARR et al., 2014), por exemplo. No entanto, diante do levantamento bibliográfico realizado (e apresentado no Capítulo 3), não há nenhuma arquitetura que se proponha a reunir e conciliar essas diferentes abordagens em um único arcabouço, seja de maneira centralizada ou distribuída. Alguns exemplos podem ser apresentados, como em (CARELA-ESPAÑOL et al., 2013), em que os autores propõem uma arquitetura em que o classificador é tratado como um *middlebox* ou, em (CALLADO et al., 2010), onde os autores combinam técnicas de classificação diferentes para obter o melhor resultado possível. Em uma arquitetura como a proposta neste trabalho não se trata apenas de utilizar as diferentes técnicas em um mesmo cenário, mas de combinar de forma harmônica algumas características específicas de cada uma das abordagens originais a fim de que se complementem. Nesse caso, o maior desafio é conseguir aumentar a acurácia alcançada – percentual de tráfego corretamente identificado – e aumentar a capacidade de processamento sem que, para isso, seja necessário o crescimento indiscriminado dos custos do arcabouço utilizado. Além disso, todas as soluções propostas pelos diferentes artigos estudados durante a concepção deste trabalho propõem uma classificação centralizada. Em outras palavras, a classificação ocorre em um ponto central, em geral sem adaptação às condições da rede, gerando possivelmente um gargalo e normalmente de maneira estática.

Uma arquitetura baseada em uma classificação verdadeiramente distribuída seria capaz de prover adaptação às condições da rede e a falhas, bem como diminuição da existência de gargalos na rede. Adicionalmente, uma vez que a classificação distribuída implica na utilização de vários classificadores espalhados na rede, caso esses classificadores sejam diferentes, é possível otimizar o processo para que a identificação do tráfego seja complementar e, portanto, a acurácia seja aumentada. Por outro lado, a distribuição da classificação implica na necessidade de lidar com alguns problemas recorrentes em sistemas distribuídos. Tradicionalmente, algumas desvantagens dos sistemas distribuídos são listadas como (i) a necessidade de *software* específico, (ii) a saturação da rede, (iii) mais componentes suscetíveis a falhas e (iv) segurança (COULOURIS et al., 2011). A distribuição de classificadores com base em uma infraestrutura SDNFV permite que essas desvantagens sejam minimizadas. O próprio gerenciamento SDN com pequenas alterações é capaz de coordenar esse processo, sem exigir grandes mudanças no tráfego da rede (poucas mensagens adicionais na rede são necessárias). Além disso, os múltiplos classificadores utilizados são independentes e parte da infraestrutura NFV, permitindo que, em caso de falha de um classificador, os demais continuem a operação sem que seja necessária a

substituição imediata do classificador que falhou. Por fim, como não são capazes de atuar ativamente na rede, questões de segurança nos classificadores são minimizadas.

Finalmente, observando o cenário da classificação de tráfego em redes IP tradicionais e as características de redes SDNFV, o grande desafio é conciliar diferentes abordagens de melhoria de desempenho (tanto do ponto de vista de acurácia como de performance computacional) para que todos sejam obtidos conjuntamente. Com base na infraestrutura SDNFV, a classificação de tráfego pode ser uma função virtual gerenciada pelo controlador da rede. Isso permite a utilização de múltiplos classificadores na rede, embora a quantidade de recursos disponíveis, normalmente, seja limitada para o administrador da rede. Adicionalmente, existem importantes questões em aberto no projeto de classificação de tráfego tradicional: (i) a existência de um único ponto de falha, (ii) o monitoramento do tráfego que sai da rede por um nó específico apenas, (iii) a imposição de um gargalo ao tráfego, (iv) as mudanças no perfil do tráfego e (v) a necessidade de manutenção de uma alta acurácia ao longo do tempo (RANTALA et al., 2008) (SCHUSTER et al., 2012) (SIMMROSS-WATTENBERG et al., 2011). Quando um único ponto de classificação é utilizado, a classificação está mais sujeita a uma interrupção, já que a falha de apenas um nó pode interromper a classificação. Em outras palavras, quando mais classificadores são posicionados em uma rede, a chance de todos os classificadores falharem é minimizada. Destaca-se também que, em redes empresariais, existem inúmeros caminhos possíveis para o tráfego, dificultando a coleta do tráfego em apenas um ponto (NG et al., 2015). Quando são considerados vários classificadores virtuais, questões relacionadas ao posicionamento deles na rede passam a ter uma maior relevância, bem como a decisão de quando instanciá-los e o tempo necessário para isso. Vários esforços foram realizados para indicar a viabilidade dessas tarefas para funções virtuais genéricas (MIJUMBI et al., 2016).

Diante da discussão sobre a classificação de tráfego e do contexto atual de redes de computadores, é possível afirmar que o paradigma atual para classificação de tráfego da rede (que apresenta um ponto único de classificação) é uma abordagem limitada, apresentando uma série de questões em aberto (citadas anteriormente, como um único ponto de falha e incapacidade de lidar com mudanças no perfil de tráfego, por exemplo). Diante desse problema, uma nova arquitetura distribuída suportada pelos conceitos de redes virtualizadas (através do paradigma SDNFV), utilizando vários classificadores de maneira integrada, será proposta, descrita e avaliada detalhadamente neste trabalho.

1.2 Hipótese e questões de partida

No contexto em que a necessidade de monitorar redes com alta precisão cresce juntamente com o crescimento da importância da Internet, é necessário que novas alternativas surjam para que o tráfego seja classificado de maneira satisfatória, levando em consideração a necessidade de restrição de recursos e adaptação às mudanças no tráfego. Poucos trabalhos se propõem a alterar a maneira tradicional em que a classificação de tráfego é realizada para atender essas novas demandas. Dessa forma, este trabalho será desenvolvido com base na seguinte hipótese:

- Uma arquitetura de classificação de tráfego distribuída, suportada pela infraestrutura de uma rede SDNFV, é capaz de melhorar a acurácia da classificação de tráfego, de se adaptar a mudanças no perfil do tráfego e de prover resiliência a falhas. A arquitetura distribuída permite que múltiplos classificadores virtuais sejam posicionados na rede em vez de utilizar um único ponto de monitoramento. A classificação combina a classificação de múltiplos classificadores.

A hipótese inicial, em linhas gerais, pode ser representada pela Figura 1.2. Diante dessa hipótese, surgem algumas questões de partida (QP) para o desenvolvimento desta tese:

QP01. Quais as principais características que precisam ser atendidas pela classificação de tráfego do ponto de vista da entidade responsável por gerenciar a rede?

QP02. É possível construir um ambiente de classificação distribuído que maximize a acurácia da classificação e forneça resiliência a falhas?

QP03. Como os classificadores devem interagir entre si?

QP04. Considerando a limitação de recursos, como os classificadores devem ser posicionados em uma rede com base em sua topologia, alcançando o máximo de cobertura possível da rede?

QP05. O posicionamento dos classificadores fornece uma solução ótima?

QP06. Qual o custo associado ao posicionamento de um classificador?

QP07. Como uma rede SDNFV pode organizar e gerenciar toda a arquitetura de classificação distribuída?

QP08. Como a classificação realizada na arquitetura proposta pode se adaptar a eventuais mudanças no perfil de tráfego monitorado?

QP09. É possível elaborar um método de avaliação que indique a eficiência e eficácia da arquitetura proposta?

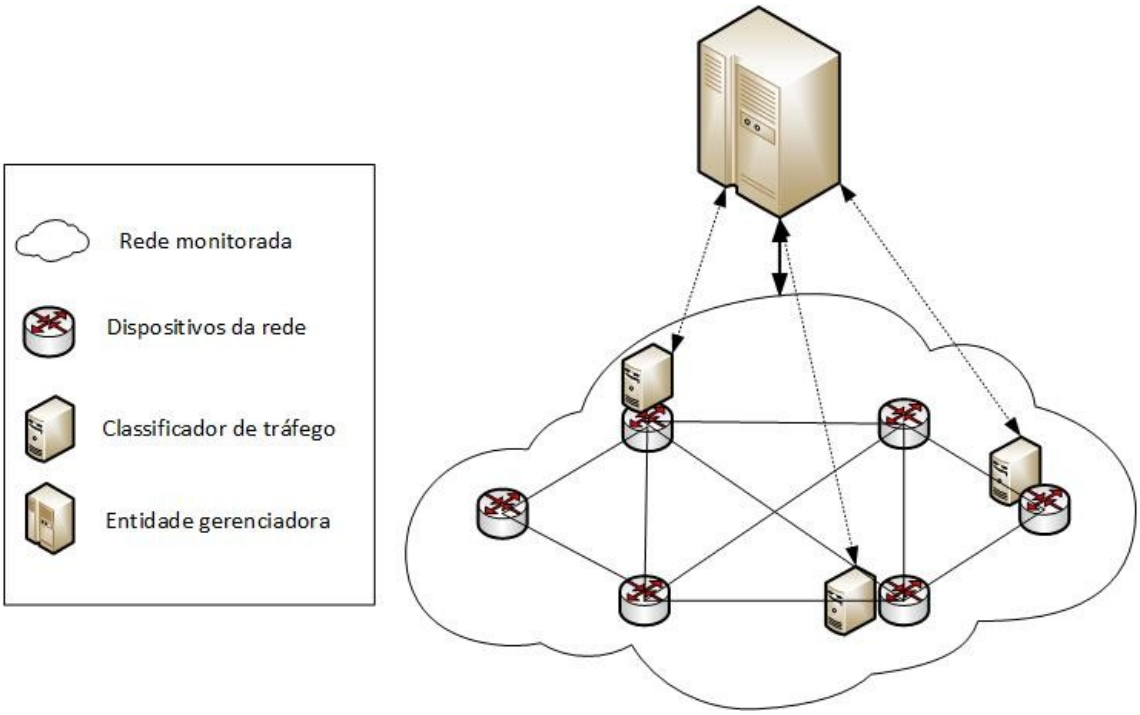


Figura 1.2 – Representação genérica da hipótese inicial

1.3 Objetivos

Diante do que foi exposto, este trabalho apresenta o seguinte objetivo geral:

- Propor uma arquitetura para classificação distribuída de tráfego de redes, capaz de fornecer, em tempo real, resiliência a falhas e adaptabilidade a mudanças do perfil de tráfego. Além disso, a arquitetura visa melhorar a acurácia da classificação em comparação ao obtido pela abordagem tradicional sem que o consumo de recursos (processamento, memória e armazenamento, por exemplo) aumente significativamente.

Adicionalmente, além do objetivo geral apresentado, alguns objetivos específicos (OE) devem ser destacados:

OE01. Propor e avaliar a interação entre classificadores de tráfego para que a acurácia seja melhorada (QP01, QP02, QP03, QP07);

OE02. Desenvolver e avaliar um algoritmo de posicionamento de classificadores em uma rede de topologia genérica, visando maximizar o número de fluxos coletados levando em conta a limitação de recursos (QP04, QP05);

OE03. Avaliar os custos operacionais envolvidos no posicionamento de um classificador virtual (QP06);

OE04. Desenvolver o método de operação e gerenciamento semi-autônomo da arquitetura com base em uma rede SDNFV (QP07);

OE05. Possibilitar e avaliar a adaptabilidade da arquitetura a mudanças do perfil de tráfego, baseado em retreinamento de técnicas de aprendizagem de máquina, e (QP08, QP09);

OE06. Analisar o comportamento da arquitetura proposta em relação às características desejadas na concepção da classificação distribuída (QP09);

1.4 Resumo dos resultados

A tese se desenvolve na direção dos objetivos citados anteriormente, em especial o objetivo geral descrito. Dessa forma, os resultados alcançados neste trabalho atestam a efetividade da arquitetura proposta.

Em relação à acurácia, a classificação distribuída chega a alcançar um ganho superior a 5% quando comparada a uma técnica de classificação tradicional. Os ganhos de efetividade e eficácia obtidos demonstram consistência em uma análise mais detalhada das classes do tráfego. Além disso, quando é considerado um tráfego com mudanças em seu perfil, a capacidade de adapatação da classificação proposta a mudanças permite um ganho de cerca de 14,5% de acurácia em comparação ao cenário tradicional. Ainda considerando a adaptação a mudanças no tráfego, a infraestrutura NFV não apresenta custos operacionais relevantes para prover essa adaptação. Adicionalmente, observando o posicionamento de classificadores, o algoritmo de posicionamento proposto supera o posicionamento aleatório em mais de 20% em relação à cobertura de caminhos da topologia. Por fim, a distribuição dos classificadores fornece resiliência a falhas, mantendo o percentual de cobertura de caminhos da topologia estável apesar da ocorrência de falhas nos nós da rede.

1.5 Estrutura da Tese

Diante da motivação e da contextualização apresentadas, considerando a hipótese, questões de partida, objetivo geral e objetivos específicos, o restante desta tese se organiza da seguinte maneira:

- No Capítulo 2, será apresentado o referencial teórico sobre as principais tecnologias e paradigmas utilizados ao longo do desenvolvimento da arquitetura. Nesse caso, os conceitos essenciais ao entendimento do trabalho serão apresentados;
- No Capítulo 3, os trabalhos relacionados à classificação de tráfego, a algoritmos de posicionamento em diferentes áreas, SDN, NFV e SDNFV serão apresentados;
- Em seguida, no Capítulo 4, a arquitetura de classificação distribuída suportada pelos conceitos apresentados no Capítulo 2 é descrita. Esse capítulo se divide em descrição geral da arquitetura e descrição detalhada de cada um dos módulos que compõem essa arquitetura;
- No Capítulo 5, o algoritmo para posicionamento de classificadores nos nós da topologia da rede é apresentado. Os passos realizados pelo algoritmo são discutidos detalhadamente para sua melhor compreensão;
- No Capítulo 6, a metodologia de avaliação deste trabalho é descrita. Nela, os cenários, métricas e método de avaliação da arquitetura serão apresentados;
- Na sequência, o Capítulo 7 apresenta os resultados dos experimentos descritos no capítulo anterior, analisando o comportamento dos módulos da arquitetura do ponto de vista da classificação de tráfego, retreinamento, posicionamento e instanciação de classificadores. Nesse capítulo, também são discutidas as possíveis ameaças à validade deste trabalho, e;
- O último capítulo desta tese se dedica às considerações finais deste trabalho. Aqui, é realizada uma discussão que correlaciona os resultados obtidos e as questões de partida elencadas anteriormente. Além disso, as contribuições desta tese também são listadas e descritas. Por fim, os trabalhos futuros são apresentados.

2 Referencial Teórico

Este capítulo apresenta inicialmente os principais conceitos envolvidos com SDN, NFV, posicionamento de servidores em uma rede e classificação de tráfego. Dessa forma, os fundamentos necessários para compreensão da proposta são apresentados. Para um leitor com prévio conhecimento sobre os assuntos abordados neste capítulo, a leitura torna-se dispensável já que se trata da apresentação de conceitos básicos (mas relevantes ao entendimento deste trabalho). Em síntese, o referencial aborda os conceitos sobre SDNFV, mineração de fluxos de dados e teoria dos grafos (em especial, componentes biconectados e triconectados), bem como a aplicabilidade desses conceitos no contexto de redes de computadores, em especial na classificação de tráfego. Também será apresentada uma breve discussão sobre o problema de *facility location* (além da apresentação de alguns conceitos) e sua relação com o contexto deste trabalho. Além disso, alguns conceitos sobre otimização de desempenho e, especificamente, sobre o método de Subida na Encosta são apresentados. Por fim, uma visão geral sobre a ferramenta gt[4] (utilizada para obtenção de um conjunto de tráfego 100% conhecido) também é fornecida.

2.1 *Software-Defined Network*

Em face da crescente complexidade observada nas redes IP tradicionais, como dito anteriormente, uma nova alternativa que possibilite a evolução dessas redes se faz necessária. Atualmente, a infraestrutura das redes enfrenta, de certa forma, o problema de "ossificação" (LI et al., 2015) (MCKEOWN et al., 2008). Em outras palavras, a infraestrutura das redes demonstra dificuldade para prover adaptação de sua operação diante de diferentes circunstâncias enfrentadas.

Em (NUNES et al., 2014), SDN é definida como um novo paradigma de redes de computadores que dissocia o *hardware* de encaminhamento (roteadores e *swicthes*) do processo de tomada de decisão. Em outras palavras, a inteligência da rede é logicamente centralizada nos controladores SDN (plano de controle), enquanto os ativos da rede servem apenas como dispositivos de encaminhamento (plano de dados), que podem ser

[4] http://netweb.ing.unibs.it/~ntw/tools/gt/. Acessado em Janeiro de 2017.

programados de maneira relativamente simples, como pode ser observado na Figura 1.3. As interfaces de comunicação são abstraídas nessa figura.

O cenário mais comumente encontrado atualmente se traduz em um sistema com controle distribuído e regulação própria conforme exposta na Figura 1.3(a) (NUNES et al., 2014). Mudanças de política, por exemplo, implicam que cada um dos ativos que atuam na rede precisa ser alterado individualmente.

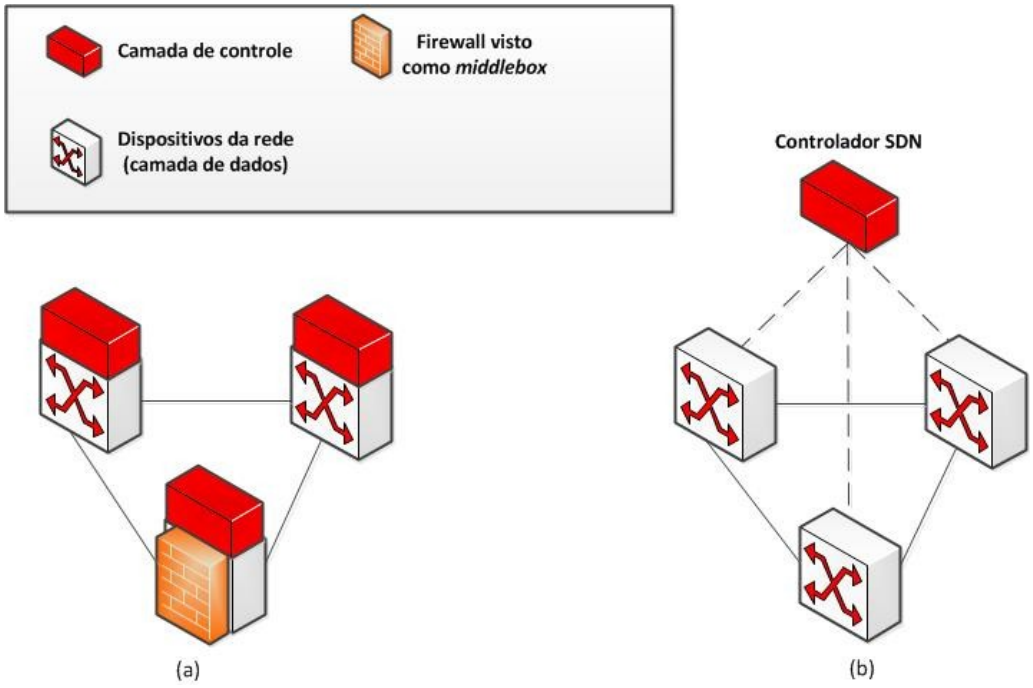


Figura 1.3 – Comparação entre duas infraestruturas de rede quando o paradigma SDN é aplicado ou não. (a) Cenário atual das redes IP. (b) Cenário gerenciado por SDN (NUNES et al., 2014).

No cenário da Figura 1.3(a), caso outro *firewall* fosse necessário em outro ponto da rede, seria preciso que um novo *middlebox* fosse implantado, sendo necessária até mesmo uma intervenção física na rede. Isso dificulta, conforme já descrito, qualquer tipo de modificação, tornando a administração da rede extremamente custosa (podendo ser custosa tanto do ponto de vista operacional quanto de recursos financeiros). É importante destacar também que as redes atuais não são, normalmente, preparadas para se adaptar a mudanças no perfil de tráfego ou mesmo para lidar com falhas. Em outras palavras, é possível afirmar que o cenário das redes atual é de integração vertical entre o plano de controle e o plano de dados implementado nos próprios dispositivos (KREUTZ et al., 2015). O paradigma SDN muda esse modelo de integração. Em (LIN et al., 2014), por exemplo, os autores proveem uma infraestrutura que permite a integração entre os recursos de aplicações heterogêneas com base no controlador SDN e um controlador de nuvem.

O ambiente que passa a ser vislumbrado a partir do momento da implementação do paradigma SDN está relacionado ao apresentado na Figura 1.3(b). Nesse caso, o plano de controle representado é desacoplado dos dispositivos da rede, operando fora da topologia da rede propriamente dita. Embora esse plano de controle apareça na Figura 1.3(b) como um ponto único central, nada impede que esse controle seja realizado de forma distribuída (LANGE et al., 2015). Outra possibilidade é que cada dispositivo de encaminhamento apresente um módulo referente ao plano de controle associado. Já em relação ao *firewall,* percebe-se que não existe o papel de *midllebox* no cenário SDN. Isso acontece porque a função do *firewall* pode estar implementada apenas logicamente, independente do dispositivo que está operando no plano de dados. Esse cenário permite que a rede torne-se programável, inserindo capacidade de decisão diante de uma visão global do ambiente através do plano de controle (KREUTZ et al., 2015). Esse plano mantém o controle sobre o plano de dados, onde os pacotes da rede trafegam (CANINI et al., 2015). O plano de controle é comumente referenciado como o controlador SDN, que é a principal entidade do plano de controle e desempenha um papel essencial para este paradigma (centraliza logicamente a inteligência da rede) (JAIN et al., 2013) (NUNES et al., 2014).

Em síntese, de acordo com (KREUTZ et al., 2015), a arquitetura SDN pode ser definida com quatro principais características que a suportam:

- As camadas de controle e dados estão dissociadas. Nesse caso, o controle da rede é retirado dos ativos da rede, que apenas redirecionam pacotes;
- Decisões de encaminhamento passam a ser baseadas em fluxos em vez de considerar apenas os destinos, como ocorre tradicionalmente (SHU et al., 2016). Dessa forma, o encaminhamento de pacotes na rede é orientado a serviço, sendo capaz de prover diferentes níveis de serviço. Em outras palavras, é possível fornecer QoS com base na compreensão do fluxo que trafega em determinado momento na rede;
- O controle lógico da rede é realizado em uma entidade externa denominada controlador SDN ou sistema operacional de rede. Essa entidade é basicamente quem permite que a rede seja programável, funcionando como um sistema operacional de fato, e;
- Por fim, a rede é programável através de aplicações que são executadas sobre o controlador SDN. Essa é considerada a principal característica proposta por uma arquitetura SDN.

Existem algumas implementações que seguem o paradigma SDN como ForCES (DORIA et al., 2010) e OpenFlow (MCKEOWN et al., 2008). A primeira implementação separa as camadas de controle e de dados, mas mantém o dispositivo de encaminhamento e controle lógico como uma única entidade, ou seja, mesmo dissociadas as camadas estão próximas. Já no caso do OpenFlow, o dispositivo de encaminhamento se comunica de maneira segura com o controlador, que não faz parte da mesma entidade, através do protocolo OpenFlow. Embora tratem do mesmo modelo (Figura 1.4), essas arquiteturas apresentam algumas diferenças fundamentais, como modelo de encaminhamento e protocolo de interface, por exemplo (WANG et al., 2012).

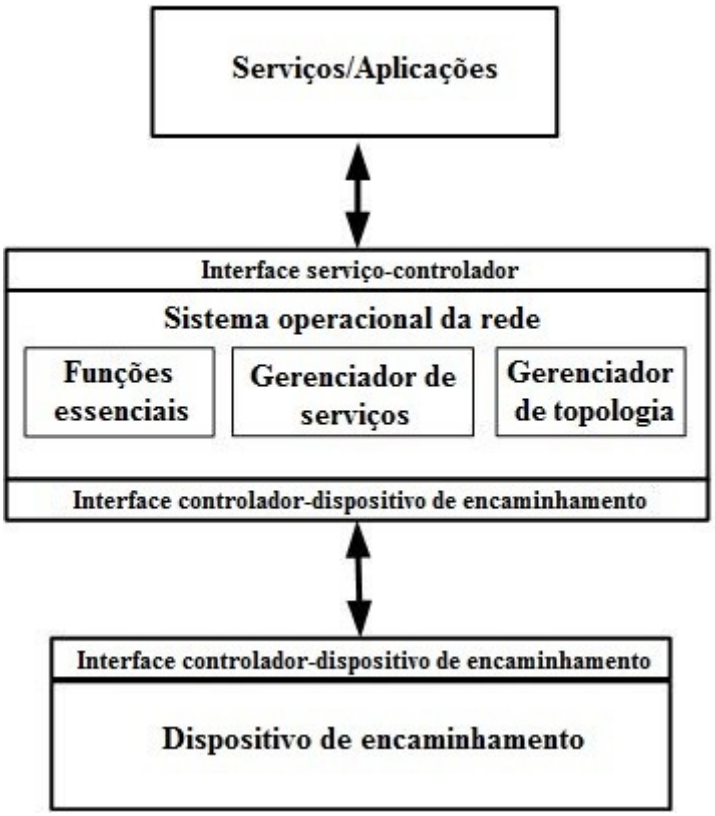


Figura 1.4 – Arquitetura com os componentes gerais SDN (NUNES et al., 2014).

Essas diferenças devem ser consideradas em uma análise e são determinantes para que seja definido o cenário de aplicação de cada uma das arquiteturas descritas. Dessa forma, percebe-se que existe certa flexibilidade da aplicação dos conceitos propagados pelo paradigma SDN, sendo possível, portanto, adaptá-los para diferentes objetivos. Atualmente, de acordo com (NUNES et al., 2014), a literatura considera o OpenFlow como o padrão de fato do paradigma SDN. Seguindo essa mesma linha de pensamento, é interessante notar que existem alguns "módulos" da arquitetura SDN que devem ser destacados e serão brevemente apresentados neste trabalho (Figura 1.4).

Na Figura 1.4, é possível observar os relacionamentos, em linhas gerais, dos componentes que fazem parte da arquitetura SDN (KREUTZ et al., 2015) (NUNES et al., 2014): controlador SDN (ou NOS), dispositivos de encaminhamento e interfaces de comunicação.

2.1.1 Controlador SDN

Os controladores SDN desempenham um papel fundamental em suas respectivas arquiteturas. Existem diversas implementações de controladores como, por exemplo, Beacon[5], Floodligth[6], Helios[7], Maestro (CAI et al., 2010) e Nox[8]. Segundo (NUNES et al., 2014), a abstração fornecida pelo controlador permite que a rede seja tratada como um único sistema e "centralize" as aplicações fora da rede propriamente dita. Isso possibilita que um controlador opere em ambientes heterogêneos.

Na concepção de um controlador SDN, surgem duas questões de projeto: (i) escalabilidade e (ii) modelo de controle.

Quando se discute o paradigma SDN, um dos primeiros questionamentos que surge está relacionado à escalabilidade de um sistema que centraliza seu controle. Nesse cenário, de acordo com (NUNES et al., 2014), um único controlador é capaz de lidar com um grande número de fluxos e está habilitado a operar mesmo em grandes redes. Ainda assim, novos controladores estão sendo desenvolvidos para ampliar sua capacidade de processamento (VOELLMY et al., 2012) e vários controladores podem ser utilizados em conjunto, reduzindo a latência e aumentando a tolerância a falhas (NUNES et al., 2014).

Já em relação aos modelos de controle, a discussão se desenvolve em torno de algumas questões como (NUNES et al., 2014): (i) centralização ou distribuição, (ii) grau de granularidade e (iii) controle reativo ou proativo. Normalmente, a arquitetura de uma rede SDN define a existência de um controlador, que, na forma usual de descrição, parece ser centralizado. No entanto, um controlador centralizado fisicamente implica em um único ponto de falha no controle da rede. Caso o plano de controle seja definido através de múltiplos controladores, uma eventual falha poderia ser contornada com os controladores redundantes. Existem algumas implementações que fornecem um controlador logicamente centralizado, mas fisicamente distribuído (KOPONEN et al., 2010) (TOOTOONCHIAN

[5] https://openflow.stanford.edu/display/Beacon/Home. Acessado em Abril de 2015.
[6] http://floodlight.openflowhub.org/. Acessado em Abril de 2015.
[7] http://www.nec.com/. Acessado em Abril de 2015.
[8] http://www.noxrepo.org/. Acessado em Maio de 2015.

et al., 2010). Nesse contexto, a centralização física ou não do plano de controle se trata mais de uma decisão de projeto do que uma definição do paradigma. Adicionalmente, dentro do paradigma SDN, a granularidade é uma importante decisão de projeto. Tradicionalmente, a unidade básica manipulada em redes de computadores é o pacote. Mas nem sempre é necessário se analisar cada pacote para que se tome uma determinada decisão. O encaminhamento de um pacote na rede, por exemplo, pode ser definido através da análise do fluxo. Em outras palavras, é necessário analisar apenas o primeiro pacote (ou alguns dos pacotes iniciais) enquanto os demais, que pertencem ao mesmo fluxo, são apenas redirecionados da mesma maneira – recebendo o mesmo tratamento (FERNANDES et al., 2009). Dessa forma, agregação de pacotes em fluxos permite uma redução da sobrecarga de processamento gerada pela grande quantidade de pacotes. Quando essa granularidade é discutida em um cenário SDN, nota-se que existe uma sobrecarga de comunicação entre o plano de controle e o plano de dados (comunicam-se remotamente). Essa sobrecarga é minimizada quando apenas as informações de fluxos são trocadas em vez da informação de cada pacote. A troca de informações através de fluxos representa um volume menor de dados, implicando que a arquitetura SDN seja mais hábil em lidar com fluxos que com pacotes. Por fim, em relação à política de controle, pode se ter uma abordagem reativa ou proativa. No primeiro caso, a controladora SDN é consultada pelo dispositivo de encaminhamento para que a sua decisão em relação ao tráfego seja informada. Do ponto de vista da abordagem proativa, as decisões são tomadas e informadas previamente ao plano de dados, que apenas segue o procedimento de acordo com o informado pelo plano de controle.

Percebe-se que o plano de controle da SDN (representado pelo controlador) exerce um papel fundamental para o funcionamento de uma rede desse tipo. Em especial, considerando o cenário de aplicação deste trabalho, o sistema operacional da rede é capaz de suportar a função de coordenação de todo o ambiente de classificação (que será descrito em detalhes adiante). Isto é, o estado da arte atual em relação a controladores SDN provê todas as ferramentas e instrumentos necessários para que a arquitetura proposta neste trabalho opere sem que seja preciso qualquer tipo de nova estrutura. Nesse caso, é necessário apenas que as estruturas atualmente existentes sejam estendidas e adaptadas para execução das atividades propostas neste trabalho. Dessa forma, a contribuição do ponto de vista de SDN está relacionada ao novo cenário de aplicação, enquanto para a classificação de tráfego a inovação está na forma de orquestrar a classificação distribuída.

2.1.2 Dispositivos de encaminhamento

Os dispositivos de encaminhamento, genericamente denominados *switches* na terminologia SDN, são definidos como um *hardware* específico capaz de encaminhar pacotes para uma determinada interface de rede. De acordo com (KREUTZ et al., 2015), esses dispositivos são "encaminhadores" que têm uma interface de comunicação aberta e que, de acordo com algum protocolo de comunicação (estabelecido previamente), recebe as informações do plano de controle e as executam. Esse cenário também está representado e pode ser observado na Figura 1.4.

Nesse contexto, fica evidente que os dispositivos de rede deixam de tomar decisões ou implementar localmente qualquer tipo de serviço de rede e, embora sejam essenciais para a operação da rede, exercem menos atividades do que acontece nas redes tradicionais. A partir da implementação do paradigma SDN, os serviços da rede são logicamente executados fora dos dispositivos (no plano de controle) e seus resultados são repassados ao plano de dados para que os pacotes sejam devidamente tratados (KREUTZ et al., 2015) (NUNES et al., 2014).

2.1.3 Interfaces de comunicação

No paradigma SDN, existem basicamente duas interfaces de comunicação essenciais ao funcionamento da arquitetura (KREUTZ et al., 2015) (NUNES et al., 2014). A interface entre o plano de controle e o plano de dados (em inglês *southbound interface*) permite que os dispositivos de encaminhamento sejam controlados pelo controlador SDN. O OpenFlow (MCKEOWN et al., 2008) é um protocolo bastante utilizado para realizar essa comunicação e algumas versões fornecem suporte ao TLS (*Transport Layer Security*). Isso permite que essa comunicação seja segura, característica bastante interessante para a rede. No caso da segunda interface, não há um padrão definido. Ela trata das interações entre os serviços e aplicações externos e o plano de controle, bem como entre diferentes controladores que compõem a arquitetura (quando em um cenário distribuído). Essa comunicação é realizada de maneira *ad hoc*.

2.2 *Network Function Virtualization* (NFV)

Em redes de computadores, a política de encadeamento de serviços (do inglês *service-chaining policy*) é a execução de múltiplos serviços como funções da rede (relacionadas geralmente a

desempenho e segurança) em uma sequência específica (DING et al., 2015). Tradicionalmente, o processamento dessas funções é realizado através de *middleboxes* (JOSEPH et al., 2008), que nada mais são que funções da rede (LI et al., 2016). Entretanto, isso origina algumas questões em aberto no projeto de redes de computadores para redução de custos operacionais e custos financeiros (DING et al., 2015). Para lidar com tais questões de projeto de redes e reduzir custos de capital (CapEx – *Capital expenditure*) e custos operacionais (OpEx – *Operational expenditure*), a virtualização surgiu como uma abordagem para separar o *software* de processamento da rede e aplicações do *hardware* de suporte, permitindo que serviços da rede sejam implementados como *software* (LI et al., 2015).

A virtualização de funções da rede foi proposta para minimizar os problemas de custos através de um *hardware* genérico e tecnologias de virtualização (DING et al., 2015). Em outras palavras, NFV oferece, através de tecnologias de virtualização, uma alternativa de projeto, instanciação e gerenciamento dos serviços da rede – substituindo os equipamentos dedicados (MIJUMBI et al., 2016) (YANG et al., 2016). Adicionalmente, NFV pode ser associado a outras arquiteturas ou tecnologias, como SDN ou computação na nuvem (CUI et al., 2012). NFV implementa funções virtuais através de técnicas de virtualização de *software*. Essas funções são executadas pelo *hardware* de dispositivos da rede em plataformas dedicadas. Nesse cenário, a função virtual da rede pode ser instanciada de acordo com a demanda, sem a necessidade de um novo equipamento, mitigando a utilização de *middleboxes,* por exemplo. A possibilidade de instanciação de funções de rede sob demanda origina novos problemas de pesquisa (GEMBER-JACOBSON et al., 2015) (HAN et al., 2015) (HERRERA et al., 2016) (HWANG et al., 2015) (LI et al., 2016):

- Como posicionar funções virtuais de maneira efetiva e eficiente. Em outras palavras, a decisão de onde a função virtual estará hospedada é extremamente importante;
- Realizar a alocação de recursos necessários para suprir a demanda de diferentes funções virtuais na infraestrutura NFV;
- Como definir o momento de instanciação de uma nova função virtual. Em outras palavras, perceber a necessidade de um serviço na rede de acordo com a demanda tornou-se um desafio, e;

- Como lidar com o impacto do tempo necessário para instanciação de uma função virtual e o tempo de movimentação dela. O tempo necessário para inicializar uma função em condições específicas deve ser considerado.

Como dito anteriormente, NFV se constitui, basicamente, da transferência de funções da rede hospedadas por *hardwares* dedicados para máquinas virtuais. Essas funções são executadas em plataformas padrão de redes como *switches* de alta performance, por exemplo. A análise de tráfego (DPI e QoE, dentre outros) e dispositivos de segurança (*firewalls,* por exemplo) são comumente tratados como funções de rede para NFV (CHIOSI et al., 2012). Geralmente, as vantagens de utilização de NFV estão relacionadas à redução de custos e à flexibilidade da operação. Formalmente, as funções virtuais são instanciadas e executadas na infraestrutura NFV (denominada NFVI – Figura 1.5).

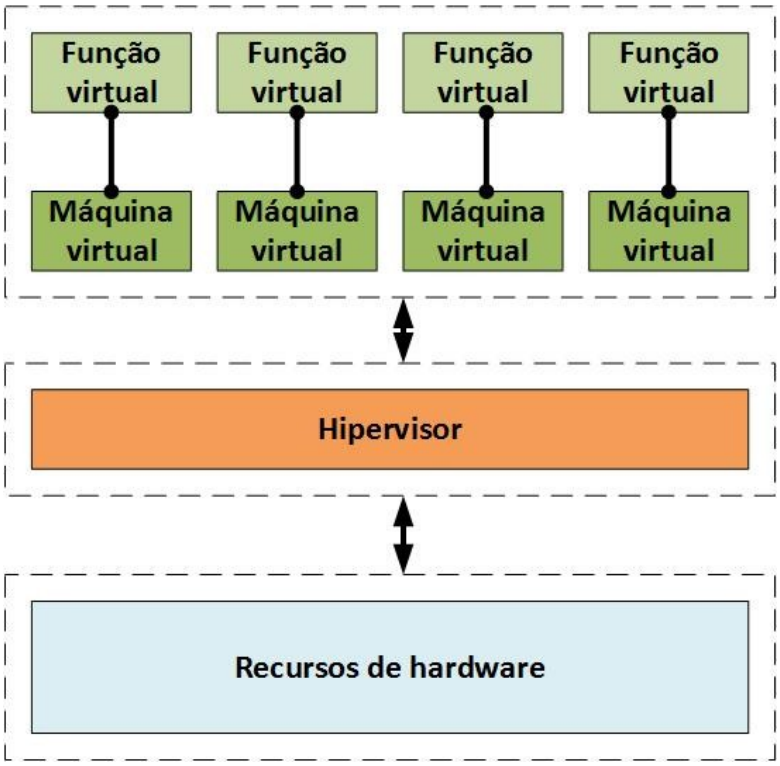


Figura 1.5 – Infraestrutura NFV (LI et al., 2015).

Essa infraestrutura consiste em servidores que proveem uma camada de software que permite a emulação de máquinas virtuais, denominada hipervisor (do inglês, *hypervisor*). O cenário mais comum é que, para cada máquina virtual, uma função da rede é mapeada (LI et al., 2015). Na Figura 1.5, é possível observar, resumidamente, a infraestrutura NFV. Embora o conceito de NFV tenha sido apresentado nesta seção de maneira isolada, é interessante destacar o relacionamento entre NFV e o paradigma SDN.

2.3 *Software-Defined Network Function Virtualization* (SDNFV)

Em (LI et al., 2015), os autores tratam a integração entre NFV e SDN como uma arquitetura denominada *Software-Defined Network Function Virtualization* (SDNFV). Essa arquitetura, que tem se tornado a forma dominante de NFV, beneficia diretamente o encadeamento de serviços (*service chaining*) (LI et al., 2015).

Quando o paradigma SDN é aplicado juntamente ao paradigma NFV, a resolução de desafios no gerenciamento dinâmico de recursos e na orquestração de serviços inteligentes é facilitada. Em outras palavras, a arquitetura SDNFV permite que o controle lógico centralizado e a virtualização minimizem os custos de provisionamento de serviços e maximizem a utilização dos recursos da rede.

O encadeamento de serviços desempenha um papel fundamental no cenário atual de redes para os provedores de serviço. Trata-se de um modelo que prevê a utilização de um conjunto de *hardwares* dedicados para serviços específicos da rede como *firewalls* e balanceadores de carga (JOSEPH et al. 2008). Ou seja, vários serviços são sobrepostos em um mesmo ponto da rede (ou em pontos diferentes) através, tradicionalmente, de *middleboxes*. O SDNFV permite que o encadeamento de serviços seja simplificado, tornando mais barato e fácil prover serviços em redes locais, redes empresariais, *data center* ou redes ISP através da utilização da virtualização (LI et al., 2015). Isso acontece porque, considerando a arquitetura SDNFV, são combinados os benefícios do provisionamento dinâmico de funções da rede de NFV com o controle centralizado do paradigma SDN.

De acordo com (LI et al., 2015), a arquitetura SDNFV é sintetizada na Figura 1.6. Ela consiste de três componentes principais: (i) módulo de controle, (ii) dispositivos de encaminhamento e (iii) plataforma NFV. Na Figura 1.6, os relacionamentos entre os componentes são expostos. O módulo de controle contém o controlador SDN e o orquestrador NFV. O primeiro é responsável pela decisão sobre o encaminhamento de pacotes e pela visão geral da rede, enquanto o segundo trata das funções virtuais da rede. É interessante salientar que o orquestrador é comandado pelo controlador SDN. Ao longo da descrição da arquitetura deste trabalho, sempre que for feita menção ao plano de controle SDN, está se referindo ao módulo de controle composto por controlador SDN e orquestrador NFV. O módulo de controle é, portanto, quem gerencia a operação dos módulos da arquitetura. A plataforma NFV, por sua vez, é a NFVI inserida nesse novo

contexto. Em outras palavras, é o conjunto formado por *hardware,* hipervisor, máquinas virtuais e funções virtuais.

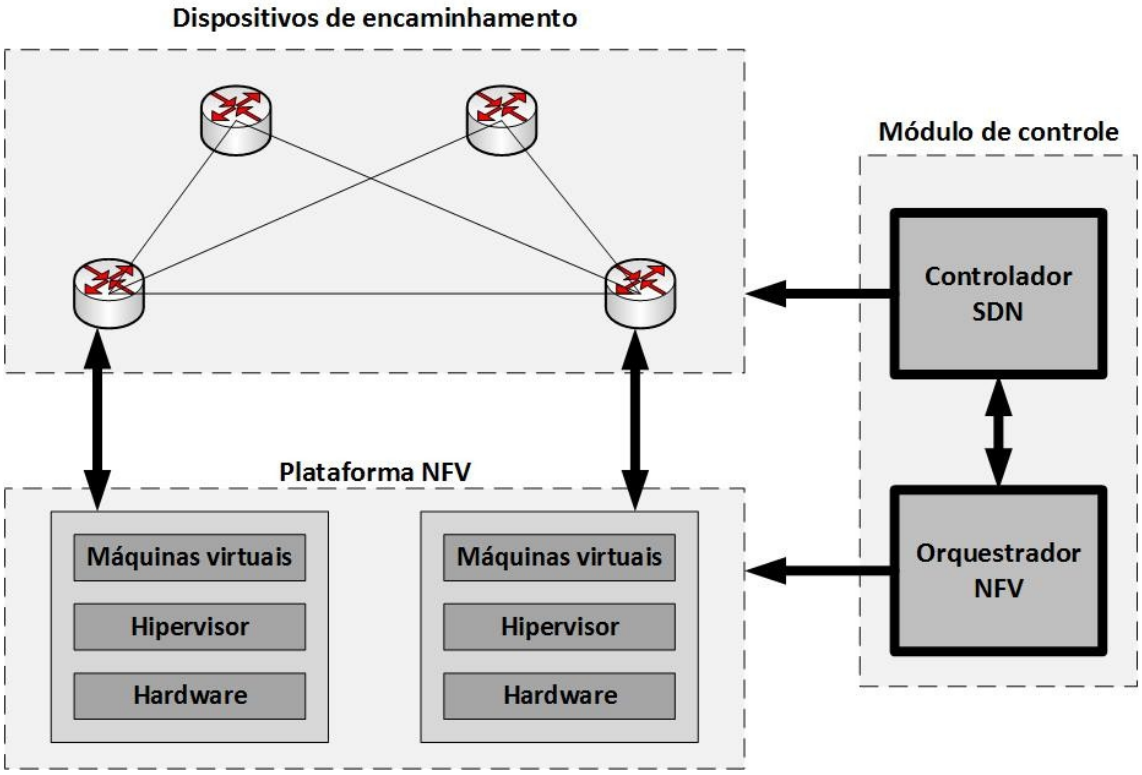


Figura 1.6 – Arquitetura SDNFV (LI et al., 2015)

Finalmente, observa-se que a arquitetura SDNFV se apresenta como um instrumento bastante poderoso no cenário de redes de computadores, uma vez que combina dois paradigmas com diferentes funcionalidades, mas extremamente compatíveis e complementares entre si. Dessa forma, a visão geral dessa arquitetura permite um melhor entendimento de como SDN e NFV podem ser úteis para o cenário atual de redes.

2.4 Aprendizagem de máquina no contexto de classificação de tráfego

O conceito que define um fluxo relacionado ao tráfego em uma rede de computadores é bastante relevante para a compreensão das técnicas de aprendizagem de máquina. A 5-tupla que define um fluxo bem como outras características do tráfego que podem ser agregadas permite que a classificação seja realizada. Especificamente, os pacotes são agregados de acordo com uma 5-tupla que define um fluxo (CALLADO et al., 2009): (i) endereço IP de origem do pacote, (ii) porta de origem do pacote, (iii) endereço IP de destino do pacote, (iv) porta de destino do pacote e (v) protocolo da camada de transporte. Nesse caso, não há

a necessidade de inspecionar cada pacote individualmente, todos que apresentarem a correspondente 5-tupla serão agregados compondo um único fluxo. Em outras palavras, todos os pacotes que tiverem os mesmos endereços de IP de origem e destino, utilizarem as mesmas portas de origem e destino e o mesmo protocolo da camada de transporte serão reunidos através de uma estrutura lógica em um mesmo fluxo.

Além da definição dos métodos de classificação, a literatura especifica que as medições de tráfego podem ser ativas ou passivas (CALLADO et al., 2009). O primeiro tipo de medição não interfere na operação normal da rede, funciona geralmente espelhando o tráfego para outro ponto em que ele será analisado. Nesse caso, o comportamento da rede analisada se mantém inalterado, embora, a depender da abordagem utilizada, alguma sobrecarga de processamento pode ser inserida no ambiente monitorado. Por outro lado, uma medição ativa insere mais pacotes na rede e interfere, mesmo que de maneira controlada, na operação normal da rede. Quando se faz a opção por uma medição passiva, é importante perceber que existem algumas restrições de desempenho. Em um ambiente com *links* de alta velocidade, por exemplo, essa técnica pode ter dificuldade de lidar com o grande volume de tráfego gerado, tanto do ponto de vista de processamento como do armazenamento dos dados (CALLADO et al., 2009). A necessidade de espaço de armazenamento acompanha de maneira linear o crescimento do tráfego que passa em um determinado roteador (APPENZELLER et al., 2004).

Considerando a área de aprendizagem de máquina, é interessante destacar os diferentes tipos de aprendizado. Existem quatro tipos básicos de aprendizagem de acordo com (WITTEN et al., 2005): (i) supervisionado (classificação), (ii) não supervisionado (clusterização), (iii) associação e (iv) predição numérica. Partindo-se dessa definição, técnicas supervisionadas são as mais indicadas para tratamento do problema abordado nesta pesquisa (já que o problema discutido corresponde à classificação de tráfego). Nesse caso, genericamente, a classificação consiste em submeter um conjunto pré-classificado de exemplos a um classificador (técnica de aprendizagem de máquina), a fim de possibilitar a construção de um modelo para classificar amostras ainda não classificadas.

A classificação de tráfego baseada em técnicas de aprendizagem de máquina, em geral, é composta por duas etapas: (i) uma fase de treinamento e (ii) uma fase de classificação propriamente dita. Na primeira etapa, um conjunto pré-classificado de fluxos (conjunto de treinamento) será entregue ao classificador para treiná-lo de maneira adequada. Com base nesse conjunto, um modelo de classificação será construído para realização da segunda etapa, a classificação de novos fluxos (NGUYEN et al., 2008). A

obtenção de um conjunto de treinamento é um dos desafios impostos por essa área (GRINGOLI et al., 2009). Nesse contexto de classificação de tráfego baseada em fluxos, ferramentas como o gt são muito importantes para que uma boa precisão seja alcançada (GRINGOLI et al., 2009). Essa ferramenta provê um conjunto de fluxos associados ao nome da aplicação que os gerou, possibilitando assim a identificação precisa do tráfego e, portanto, do conjunto de treinamento e de teste. Existem outras formas de conseguir uma base de conhecimento para realizar o treinamento de técnicas de ML, que podem ser utilizadas em conjunto para obtenção de um conjunto de treinamento bastante completo (CARELA-ESPAÑOL et al., 2013).

Existem diversas técnicas de aprendizagem de máquina que, de acordo com o objetivo da classificação, podem ser utilizadas para alcançar melhores desempenhos. Para classificação de tráfego, alguns trabalhos apontam a técnica C4.5 como o melhor método de classificação em relação à precisão (DAINOTTI et al., 2012) (QUINLAN, 1993) (LOPES et al., 2014) (WILLIAMS et al., 2006).

A aprendizagem de máquina traz, ainda, alguns desafios que influenciam em sua acurácia e desempenho computacional, como o tamanho do conjunto de treinamento e sua proporção em relação ao número de fluxos a serem classificados, por exemplo. Alguns desses aspectos serão tratados na metodologia e nos experimentos deste trabalho.

2.5 Mineração de fluxos de dados (*Streaming data mining)*

Além das características apresentadas anteriormente, a classificação de tráfego durante a operação de uma rede enfrenta alguns desafios específicos. O tipo de dados que trafega e será posteriormente classificado em uma rede consiste de um fluxo contínuo de tráfego com múltiplas origens, que respeita uma distribuição estatística não estacionária e que apresenta altas taxas de geração. Essas características implicam que o tráfego de uma rede se encaixa perfeitamente na definição de fluxos de dados (*data streams*) (GAMA et al., 2007). Existem métodos que tratam da extração de informação de fontes de dados com essas características. Esses métodos são denominados mineração de fluxos de dados. Para realizar a extração de informação desses fluxos de dados, algumas restrições devem ser consideradas (JIANG et al., 2006). Em outras palavras, a mineração de fluxos de dados deve respeitar uma série de restrições, que não costumam ser enfrentadas em um modelo de mineração tradicional. De acordo com (ABDULSALAM et al., 2007), são restrições para métodos de mineração de fluxos de dados:

1. Ser rápido o suficiente diante da chegada de novos dados;
2. Adaptar-se a mudanças na distribuição estatística que define os dados;
3. Operar em tempo real, e;
4. Extrair as informações necessárias em uma ou poucas passagens pelo conjunto de dados.

Do ponto de vista deste trabalho, essas restrições devem ser levadas em consideração já que o tráfego a ser classificado, como dito anteriormente, se ajusta à definição de fluxos de dados. Dessa forma, é importante a manutenção de todas as abordagens adotadas nesta pesquisa prezando pela simplicidade e pela efetividade da solução com o mínimo de recursos possível.

Serão descritas adiante algumas características da arquitetura proposta e dos seus módulos que visam respeitar as restrições de um cenário de mineração de fluxos de dados, como a preparação prévia do classificador, o módulo de retreinamento e a utilização de conjuntos de treinamentos pequenos, por exemplo. No entanto, embora os limites impostos pela mineração de fluxos de dados sejam respeitados, é objetivo deste trabalho alcançar resultados satisfatórios apesar das referidas restrições.

2.6 Grafos

A área de teoria dos grafos é bastante extensa e, dentro da computação, apresenta diversas vertentes aplicadas em diferentes áreas de pesquisa. Um grafo é uma estrutura de dados capaz de representar muitos problemas da ciência da computação e em outras áreas (CORMEN et al., 2001). Observando um mapa geográfico, por exemplo, os nós podem representar cidades, enquanto as arestas são estradas (SIPSER, 2007). Nesse caso, algoritmos em grafos podem encontrar o menor caminho entre duas cidades ou o melhor caminho com base em um critério de valoração pré-definido, por exemplo.

No contexto deste trabalho, a topologia da rede é representada como um grafo e a teoria dos grafos é utilizada para suportar a interação da arquitetura distribuída proposta com a topologia da rede. Para isso, alguns conceitos são essenciais para o desenvolvimento deste trabalho, como a definição de grafos *K*-conectados. No entanto, antes de apresentar tais definições, é necessário que uma visão geral sobre grafos seja fornecida. Neste trabalho, para simplificação do problema a ser tratado, quando se faz referência a um grafo, embora não seja dito explicitamente, necessariamente corresponde a um grafo conectado e não direcionado.

Existem várias definições de terminologia, mas não há um único padrão. De acordo com (READ, 2014), um grafo *G* consiste em um conjunto de nós *V,* que são conectados entre si através de arestas não direcionadas pertencentes a um conjunto de arestas *E*. Dessa forma, um grafo pode ser definido formalmente como na Equação 1.1.

Equação 1.1 – Definição formal de um grafo.

$$G = (V, E)$$

Dados dois nós quaisquer *i* e *j,* uma aresta entre eles é definida pelo par (*i,j*). No contexto apresentado, os pares (*i,j*) e (*j,i*) tratam da mesma aresta (SIPSER, 2007). De acordo com a definição formal apresentada, na Figura 1.7, é exposto um grafo rotulado *G* para ilustrar o seu conceito. Nesse caso, o grafo é definido pelo conjunto de nós *V* = {*V1,V2,V3,V4,V5,V6*} e pelo o conjunto de arestas *E* = {*E1,E2,E3,E4,E5,E6,E7,E8,E9*}, em que cada aresta é um par de nós (*E1 = (V1,V2)*, por exemplo).

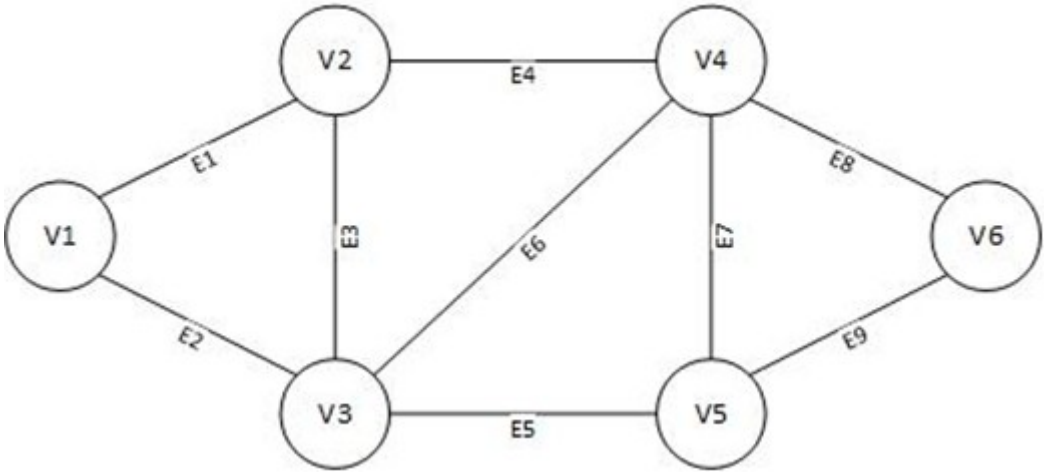


Figura 1.7 – Exemplo de grafo

Diante das definições apresentadas até este ponto, o grafo rotulado *G* então pode ser representado formalmente como descrito a seguir na Equação 1.2. Nele, as arestas estão representadas por pares de nós em vez de utilizar os rótulos expostos na Figura 1.7. O número de arestas de cada nó determina o seu grau (SIPSER, 2007). Observando a Figura 1.7, é possível determinar o grau dos nós do grafo: o nó *V1* tem grau 2, enquanto o nó *V4* tem grau 4, por exemplo.

Equação 1.2 – Representação formal do grafo *G*.

$$G = \begin{pmatrix} \{V1, V2, V3, V4, V5, V6\}, \\ \{(V1, V2), (V1, V3), (V2, V3), (V2, V4), (V3, V4), (V3, V5), (V4, V5), (V4, V6), (V5, V6)\} \end{pmatrix}$$

Em (SIPSER, 2007), os autores definem um subgrafo de um grafo *G* como um grafo que o seu conjunto de nós é subconjunto dos nós de *G* e suas arestas são um subconjunto das arestas de *G*. Além desse conceito, é relevante a definição do conceito de caminho. Em um determinado grafo, um caminho é uma sequência de nós conectados por arestas. Diferentemente da definição formal, todas as referências à cobertura de um nó (denominado *A*) de um grafo estão relacionadas ao percentual de caminhos entre dois nós quaisquer que contêm o nó *A*. Um ciclo, por sua vez, é um caminho que começa e termina em um mesmo nó. Adicionalmente, um grafo é considerado desconectado quando pode ser subdividido em dois ou mais subconjuntos de nós que não sejam interligados por nenhuma aresta. Em outras palavras, trata-se de um grafo formado por dois ou mais subgrafos que não têm ao menos um caminho em comum.

Alguns outros conceitos relacionados a grafos são importantes além dos que foram apresentados até este ponto do trabalho. Vários desses conceitos apresentados a seguir são definidos em (CORMEN et al., 2001), (READ, 2014) e (SIPSER, 2007). Um grafo simples não apresenta arestas que ligam um nó a ele mesmo e dois nós são conectados por, no máximo, uma aresta. Quando a retirada de um nó e de todas as suas arestas de um determinado grafo gera outros grafos desconectados, esse nó é denominado ponto de articulação ou vértice de corte e o grafo é separável. Isso é, quando um nó pertence a todos os caminhos entre dois subgrafos, ele é um ponto de articulação. Por outro lado, quando um grafo não apresenta um ponto de articulação, ele é não separável ou multiplamente conectado.

Um importante conceito utilizado no posicionamento de classificadores neste trabalho está relacionado à definição de grafos multiplamente conectados. Generalizando o conceito em (READ, 2014), um determinado grafo é *K*-conectado quando é possível retirar *K*-1 vértices (que funcionam como pontos de articulação) sem que o grafo se apresente desconectado. Nesse caso, um grafo é 3-conectado (triconectado) quando é possível remover de sua estrutura quaisquer dois nós e suas arestas sem que o grafo passe a ser desconectado. Com base nesses conceitos, este trabalho utiliza o conceito de grafos multiplamente conectados para identificar, em uma determinada topologia, componentes biconectados e triconectados. Quando esses componentes são identificados, os pontos de

articulação desses componentes são indicados e utilizados no posicionamento de classificadores na rede como será descrito com mais detalhes adiante.

2.7 Problema de locação de recursos (*Facility Location problem)*

Facility location corresponde a um problema de locação de um conjunto de recursos para minimizar o custo de satisfazer determinadas demandas (possivelmente, clientes) considerando algumas restrições. Essa classe de problemas se aplica em diversos cenários que envolvem operação e logística (WOLF, 2009). Em outras palavras, o problema de locação se refere à modelagem, formulação e solução da classe de problemas que lida com a alocação de recursos em um determinado espaço. Os termos alocação, instanciação e posicionamento são utilizados para referenciar esse problema (WOLF, 2009).

Quatro componentes descrevem problemas de locação de acordo com (WOLF, 2009):

- Clientes – que estão previamente localizados em determinados pontos do espaço do problema;
- Recursos – que serão alocados;
- Espaço do problema – que remete ao ambiente em que clientes e recursos são posicionados, e;
- Métrica – que representa a distância ou o tempo entre os clientes e recursos.

Com base nesses componentes, é possível realizar um mapeamento direto para um dos problemas abordados nesta tese, o posicionamento de classificadores. Nesse caso, os clientes são os fluxos do tráfego da rede. Os recursos são os classificadores virtuais a serem distribuídos na rede. O espaço do problema é a topologia da rede associada à infraestrutura SDNFV. Por fim, a métrica pode ser representada pela cobertura de caminhos entre dois nós da topologia obtida pelos classificadores, o que maximiza a disponibilidade de recursos (classificadores) para os clientes (fluxos).

Especificamente, o problema discutido neste trabalho se aproxima do problema de alocação de local (*location-allocation problem*) (COOPER, 1963). O problema de alocação de local apresenta as seguintes variáveis (WOLF, 2009): (i) número de recursos, (ii) localização dos recursos, (iii) demanda dos clientes e (iv) capacidade de cada um dos recursos. Conforme será descrito ao longo desta tese, o problema de posicionamento de

classificadores pode ser mapeado para as variáveis da alocação de local. O número de classificadores é definido pelo administrador da rede, a localização dos classificadores será a saída do algoritmo de posicionamento, a demanda dos clientes está associada à cobertura dos fluxos e a capacidade dos classificadores é homogênea. Outro problema similar ao estudado neste trabalho é apresentado em (CHURCH et al., 1974). Trata-se do problema de maximização de cobertura em uma rede. A ideia é cobrir o máximo da demanda dos clientes possível, reduzindo a distância entre recursos e clientes. Alguns parâmetros são importantes para o problema de cobertura da rede (WOLF, 2009): (i) pontos de demanda, (ii) o número de pontos de demanda, (iii) pontos candidatos de alocação, (iv) o número de pontos candidatos à alocação e (v) o custo de posicionar um recurso. Nesse caso, os pontos de demanda e os pontos candidatos de alocação são todos os nós da topologia, além do custo ser o mesmo para todos os recursos.

No entanto, é importante notar que, diferentemente do que é descrito genericamente para os problemas discutidos (problemas de alocação de local e cobertura), os clientes não estão fixos na topologia da rede, o que torna a definição da métrica relacionada à cobertura de caminhos ainda mais relevante. A peculiaridade de apresentar clientes que se movimentam na topologia gerenciada dificulta o cálculo da "distância" entre recursos e clientes. Dessa forma, o algoritmo proposto no Capítulo 5 desta tese considera a "distância" entre classificadores e caminhos entre dois nós, sem considerar os fluxos para sua concepção.

2.8 Otimização de Desempenho

Otimizações podem ser úteis em diversas áreas da computação, incluindo a área de redes de computadores. Nas subseções a seguir, serão apresentados alguns conceitos de otimização importantes para este trabalho e o contexto de aplicação, bem como a descrição de um método específico que será utilizado.

2.8.1 Conceitos de otimização

Conforme apresentado ao longo deste trabalho, a necessidade de promover a utilização de recursos computacionais de maneira otimizada é crescente no contexto de redes de computadores (LI et al., 2015) (OTOKURA et al., 2016). Nesse caso, várias questões relacionadas ao contexto de redes de computadores podem ser discutidas como problemas de otimização, como apresentado em (OTOKURA et al., 2016). Este trabalho apresenta

um algoritmo para apontar o posicionamento ideal de classificadores em uma determinada topologia, maximizando a cobertura da topologia e minimizando o número de classificadores necessários. O posicionamento desses classificadores pode ser tratado como um problema de otimização. Em geral, existem diversos métodos ou algoritmos que buscam uma solução ótima para um determinado problema, evitando uma busca exaustiva dentre as soluções existentes. No entanto, o resultado obtido nem sempre é a solução ótima.

Nesse contexto, surgem os algoritmos probabilísticos. Alguns desses algoritmos pretendem alcançar bons resultados com custo computacional baixo (WEISE, 2009). Tais resultados estão diretamente relacionados às heurísticas utilizadas. Em (WEISE, 2009), uma heurística é definida como a parte de um algoritmo de otimização que utiliza informações coletadas pelo algoritmo para ajudar na decisão de qual o próximo candidato a solução a ser examinado ou como gerar o próximo indivíduo. É importante destacar que os métodos heurísticos são capazes de fornecer uma solução com uma qualidade considerável, mas não necessariamente ótima. Como dito anteriormente, esses métodos surgem como alternativa para solucionar problemas que são computacionalmente intratáveis por métodos exatos ou inviáveis em determinadas circunstâncias (GOMES, 2006). Normalmente, os problemas analisados apresentam uma grande complexidade computacional, muitas vezes podendo ser tratados como problemas de otimização e busca. Nessa classe de problemas, nem sempre é possível apontar a existência de uma solução ótima que seja obtida com tempo e custo computacionais aceitáveis (MONTEIRO et al., 2011). Meta-heurísticas, por sua vez, são métodos para solucionar classes de problemas genéricas. Elas combinam funções objetivo ou heurísticas de maneira abstrata e eficiente, normalmente sem modificar suas estruturas (WEISE, 2009).

Existem vários métodos de otimização como, por exemplo, o Subida na Encosta (RUSSEL et al., 2003), Algoritmos evolucionários (BÄCK, 1996) e Colônia de formigas (DORIGO et al., 2004). No entanto, não é possível apontar o melhor dentre eles. O melhor método de busca depende do problema estudado e suas restrições. De acordo com (WEISE, 2009), uma possível interpretação para o Teorema "*No Free Lunch*" é que é impossível, para qualquer algoritmo de otimização, superar buscas aleatórias ou exaustivas para todos os possíveis problemas. Para cada problema que um método apresente bons resultados, é possível construir um problema que o mesmo método tenha resultados opostos.

Diante do cenário de otimização apresentado até este ponto, é necessário encontrar e adaptar o algoritmo de otimização mais adequado para cada tipo de problema. Existem algoritmos de diferentes famílias, como métodos de busca local ou métodos baseados na população. Métodos de busca local se baseiam na exploração do espaço de soluções por um caminho composto por possíveis soluções visitadas. A segunda família, por sua vez, explora o espaço de solução através de uma população de indivíduos (BAROLLI et al., 2011). Neste trabalho, a solução proposta faz uso do algoritmo Subida na Encosta.

2.8.2 Subida na Encosta

O algoritmo Subida na Encosta (do inglês, *Hill Climbing*), assim como outros algoritmos de otimização, foi criado com base em um processo físico (MONTEIRO et al., 2011) (WEISE, 2009). Nesse caso, o processo é baseado na subida de um monte até seu ponto mais alto por um indivíduo que não conhece o caminho e tem visão limitada dele. Para realizar a subida, o indivíduo examina os pontos mais próximos para descobrir se algum deles é um ponto mais alto. A subida é encerrada quando um pico é alcançado, ou seja, nenhum ponto próximo é mais alto que a atual localização do indivíduo.

Considerando o mapeamento entre o processo de subida em um monte e o algoritmo, podemos estabelecer algumas correlações (MONTEIRO et al., 2011):

- O ponto de partida da subida é identificado como um estado inicial (nó corrente). Esse estado inicial é, normalmente, definido aleatoriamente (OLIVEIRA, 2008);
- Os pontos mais próximos que serão examinados são os estados vizinhos (ou apenas vizinhos), e;
- O ponto final da subida é o estado-objetivo (máximo global) ou um estado intermediário que não tem vizinhos mais altos (máximo local).

O algoritmo Subida na Encosta consiste em um laço repetitivo que movimenta o nó corrente através de seus vizinhos no sentido de um resultado ótimo (estado-objetivo). Em outras palavras, trata-se de uma busca heurística que gera e testa soluções (vizinhos) a partir de uma possível solução (nó corrente) para obter um resultado final satisfatório (RUSSEL et al., 2003). O fato de estabelecer o nó corrente inicial de maneira aleatória implica, em alguns casos, na limitação dos resultados do algoritmo. O método de Subida na Encosta, nesse caso, é apontado como um método de busca local (OLIVEIRA, 2008). No entanto, existem diferentes abordagens para minimizar o impacto desse problema, como o

algoritmo Subida na Encosta com Reinícios Aleatórios (SPALL, 2003) e a criação direcionada de um estado inicial (MONTEIRO et al., 2011).

A avaliação feita nos vizinhos para realizar a tomada de decisão é representada por uma função objetivo (ou função custo) (OLIVEIRA, 2008). Nesse caso, o valor retornado por essa função deve ser maximizado ou minimizado de acordo com o problema a ser solucionado. Considerando o problema original (subida em um monte), a função objetivo retorna a altura dos estados vizinhos ao nó corrente e a busca é realizada até que o ponto mais alto seja alcançado.

Outro aspecto importante relacionado ao algoritmo é a criação de uma condição de parada da busca (a descoberta de um ótimo local). Geralmente, quando a função objetivo não pode ser maximizada, o algoritmo será encerrado. No entanto, é possível que o espaço de busca do algoritmo seja grande o suficiente para tornar o algoritmo custoso demais. Nesse caso, um critério de parada da busca pode ser estabelecido, como a quantidade de interações do algoritmo (MONTEIRO et al., 2011).

De acordo com (WEISE, 2009), o principal problema enfrentado por esse algoritmo é a convergência prematura. Esse método de busca frequentemente fica preso em um máximo local, limitando os benefícios da solução encontrada. Uma maneira de minimizar esse risco é através de métodos que determinam o estado inicial (ponto de partida) do algoritmo (MONTEIRO, 2011). É importante destacar que o método Subida na Encosta pode operar de maneira exaustiva desde que o espaço de busca seja limitado e o poder computacional seja suficiente para realizar uma busca exaustiva (WEISE, 2009).

Dessa forma, o problema de posicionamento de classificadores apresentado neste trabalho pode ser tratado como um problema de otimização e ser modelado com base no método de Subida na Encosta. A definição do estado inicial será direcionada e o algoritmo será determinístico, conforme será apresentado na seção a seguir.

2.9 Ferramenta gt

Diversos trabalhos científicos na área de classificação de tráfego se deparam com o problema de avaliação em um ambiente controlado. Uma das principais dificuldades encontradas é a reprodução do tráfego real em um ambiente de emulação/simulação. Geradores de tráfego são ferramentas bastante úteis, mas apresentam uma série de limitações e dificilmente reproduzem com fidelidade o tráfego real (BOTTA et al., 2010). Além disso, um conjunto de dados (*trace*) que reproduza o comportamento de um ambiente

real nem sempre está disponível e, quando existe, muitas vezes não é descrito por metadados. Em outras palavras, não é um conjunto de dados totalmente conhecido, sem a devida descrição dos dados, impossibilitando que o ambiente de avaliação seja totalmente controlado. É possível agrupar grande parte dos mecanismos de obtenção de *traces* utilizados atualmente em uma das seguintes descrições (GRINGOLI et al., 2009):

i. Criação manual dos *traces* através da execução isolada de aplicações em uma ou mais máquinas. Nesse caso, cada máquina gera um tipo de tráfego. Essa técnica, no entanto, não reproduz o comportamento das aplicações em um cenário real;

ii. Registrar o tráfego em uma rede real e aplicar algum método ou ferramenta de DPI, complementada pela análise de portas. Nesse caso, a identificação do tráfego será tão eficiente quanto o DPI utilizado e o tráfego criptografado não será devidamente identificado, e;

iii. Utilizar geradores de tráfego que, de acordo com (BOTTA et al., 2010), apresentam algumas limitações para reproduzir fidedignamente o tráfego de uma rede real.

Em (GRINGOLI et al., 2009), o gt é apresentado e descrito. Essa ferramenta consiste em uma arquitetura para obter conjuntos de dados reais e seus respectivos metadados. Como descrito em (LOPES et al., 2014) e exposto na Figura 1.8, o gt é segmentado em três partes: um cliente, um banco de dados e um classificador (IPClass). A primeira parte, o cliente, executa em uma ou mais máquinas distribuídas em uma rede, como um *daemon* em cada uma delas, enviando para o banco de dados (SQL *Server*) as informações relevantes de cada fluxo gerado pela máquina monitorada, adicionalmente também é coletado o nome do processo que deu origem ao fluxo. Ao mesmo tempo em que o cliente e o banco de dados realizam suas atividades, é realizada uma captura de todos os pacotes das máquinas monitoradas, através de uma ferramenta de coleta de tráfego como o TCPDUMP[9], por exemplo, no roteador de borda da rede. No final desse processo, a terceira parte da arquitetura, denominada IPClass é executada, relacionando os pacotes coletados às informações que existem no banco de dados. Por fim, um DPI simples, que compõe essa terceira parte da arquitetura, é executado para classificar todos os pacotes que não tiverem as respectivas informações no banco de dados. Os resultados obtidos nesse trabalho demonstram que a ferramenta é capaz de associar a classificação a mais de 99% dos *bytes* capturados e a cerca de 95% dos fluxos que são produzidos e monitorados.

[9] Disponível em: http://www.tcpdump.org/. Acessado em Maio de 2017.

A Figura 1.8 representa uma implementação comum da ferramenta para obtenção de conjuntos de dados em um ambiente real e contém todos os elementos desenvolvidos e descritos em (GRINGOLI et al., 2009). No entanto, a implementação não está restrita a este formato, podendo ser instanciados mais coletores de tráfego em mais de um roteador de borda, por exemplo. Esse conhecimento da arquitetura e seus módulos possibilita que sejam realizadas modificações necessárias para atender diferentes necessidades. Assim como em (LOPES et al., 2014), a fim de eliminar imprecisões encontradas no conjunto de dados coletado com o gt, para o desenvolvimento deste trabalho, o resultado da execução do IPClass não foi utilizado, já que ele insere imprecisão ao executar um DPI. Nesse caso, neste trabalho, foram utilizados *traces* disponibilizados pelos desenvolvedores do gt mediante solicitação em sua página *WEB* (mesma página da ferramenta gt). Apenas os nomes dos processos associados aos fluxos são utilizados, fornecendo total confiança do tipo de tráfego que foi gerado. Da mesma forma, os fluxos que não apresentam o processo devidamente identificado por algum motivo ou processos que não geram um tipo de tráfego conhecido são retirados do conjunto de dados utilizado na avaliação deste trabalho. Esse pré-processamento é essencial para a confiabilidade do conjunto de dados utilizado.

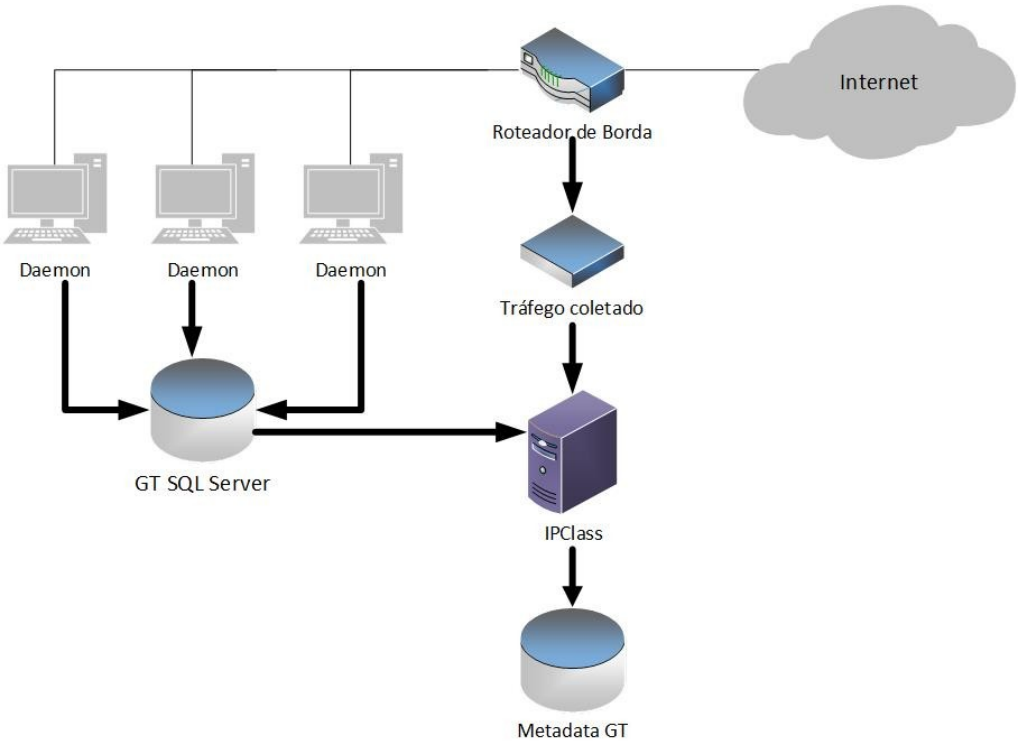


Figura 1.8 – Arquitetura GT (GRINGOLI et al., 2009).

3 Trabalhos Relacionados

Após uma extensa pesquisa nas áreas que circundam e permeiam a arquitetura proposta neste trabalho, nas próximas subseções, serão apresentados os trabalhos mais relevantes que, de alguma forma, estejam relacionados à classificação de tráfego e à hipótese inicial deste trabalho. Em especial, pesquisas sobre classificação e suas arquiteturas, bem como SDN e NFV no contexto da classificação de tráfego e posicionamento de servidores no cenário de gerenciamento e monitoramento de tráfego de redes de computadores.

3.1 Classificação de tráfego

Nas subseções a seguir, serão apresentados trabalhos relacionados a este trabalho que pertencem à área de classificação de tráfego, além de um resumo com as principais características desses trabalhos que se relacionam com esta tese.

3.1.1 Trabalhos relacionados na área de classificação de tráfego

Diante da crescente necessidade de identificar o tráfego da rede (discutida no Capítulo 0), muitas pesquisas foram realizadas nessa área nos últimos anos. Apesar de existirem inúmeros objetivos que motivam o desenvolvimento das diferentes abordagens para classificação, é possível observá-las em um mesmo contexto. Em outras palavras, é possível agrupá-las e analisá-las sob a perspectiva arquitetural, apesar das diferenças que apresentam. Este trabalho apresentará pesquisas realizadas para aumento da acurácia e redução do custo computacional, por exemplo. De maneira geral, os trabalhos nessa área buscam melhorar um ou mais dos seguintes aspectos: acurácia, tempo de classificação, quantidade de processamento e quantidade de memória. Para alcançar esses objetivos, várias arquiteturas foram propostas ao longo dos anos, permitindo que o tráfego fosse manipulado de maneira mais eficiente.

Em (LOPES et al., 2014), os autores propuseram uma arquitetura baseada na distribuição da classificação de tráfego em dois dispositivos que se comunicam. Nesse caso, os dois dispositivos em questão são um computador tradicional e uma GPU (*Graphic Processing Unit*). O trabalho se baseia principalmente na capacidade da GPU seguir o paradigma SIMD (*Single intruction multiple data*), em que a mesma instrução é executada para

múltiplos dados, e no tratamento do fluxo contínuo de dados na rede. A classificação em (LOPES et al., 2014) faz uso de um algoritmo próprio, o GSDT (GPU-*based Streaming Decision Tree*) – uma árvore de decisão modificada para atender necessidades específicas do fluxo contínuo e do cenário de redes de computadores, que implementa uma técnica de aprendizagem de máquina que lida com fluxos. Essa técnica é executada em duas etapas: treinamento e classificação propriamente dita. No treinamento, os fluxos previamente conhecidos são submetidos ao algoritmo de maneira sequencial para que a árvore de decisão seja construída. Passada essa etapa, o GSDT pode iniciar sua operação de classificação de tráfego propriamente dita. Nesse caso, um *buffer* de armazenamento é preenchido com fluxos que são coletados na rede monitorada. Quando esse *buffer* está cheio, esses dados são enviados para a GPU, que no tempo de uma execução do algoritmo, classifica todos os dados recebidos. Os resultados demonstram um considerável crescimento da vazão do classificador quando a GPU é utilizada, melhorando o tempo de processamento e minimizando a necessidade de armazenamento do classificador. Além disso, por se tratar de uma técnica projetada para o cenário de redes de computadores, ela obtém um ganho de acurácia quando comparada ao C4.5 (QUINLAN, 1993), apontada na literatura como a melhor técnica de aprendizagem de máquina para esse tipo de classificação. Em (LOPES et al., 2014), uma extensão da abordagem da classificação de tráfego através de uma GPU demonstra ainda que, enquanto o GSDT está em operação, outra árvore de decisão pode ser gerada com um novo conjunto de treinamento. Isso faz com que a arquitetura se adapte às mudanças de perfil do tráfego sem que para isso a classificação seja interrompida, apenas a nova árvore gerada deve substituir a antiga na GPU. O GSDT, portanto, propõe uma nova arquitetura baseada em uma GPU, que obtém ganhos de acurácia e performance. No entanto, ainda apresenta um único ponto de falha e trata do classificador isoladamente, sem demonstrar como essa arquitetura seria viabilizada. Além dessa, outras pesquisas baseiam suas arquiteturas em unidades de processamento gráfico, como em (BARRIONUEVO et al., 2015), por exemplo.

Assim como em vários outros trabalhos, em (MELO et al., 2014), um aspecto específico de um tipo de classificação é estudado. Nesse caso, o estudo é realizado considerando um DPI que faz uso de uma lista de assinaturas para realizar a classificação e que a velocidade de operação do DPI pode aumentar através da manipulação dessa lista. A ideia é bastante simples: as assinaturas mais relevantes são postas no início da lista para que, caso ocorra um *match* no início, o resto das assinaturas não precise ser inspecionado. Para pôr em prática essa ideia, os autores desenvolveram uma função para valorar os *matches* de

uma determinada assinatura de maneira similar ao que acontece com algoritmos de *cache*. Dessa forma, as assinaturas que apresentarem os "melhores" valores de acordo com a função serão posicionadas no início da lista. Em outras palavras, é proposto um método de reordenamento dinâmico da lista de assinaturas. Os resultados mostram que a configuração mais reativa da função de valoração das assinaturas permite a economia de mais de 50% do tempo de processamento. Por fim, uma das principais contribuições desse trabalho é a possibilidade de ser integrado e/ou adaptado com várias outras técnicas ou arquiteturas de classificação desde que um classificador com uma lista de assinaturas seja utilizado. Esse classificador poderia ser implementado em um *hardware* específico ou ser combinado com uma técnica de aprendizagem de máquina, por exemplo, sem prejuízo para a reordenação das assinaturas. Outras pesquisas apresentam alterações no método de classificação para obter um resultado mais efetivo. Em (WANG et al., 2014), os autores propuseram a utilização de informações de *background* e estatísticas do tráfego para aumentar a acurácia da clusterização, obtendo ganhos significativos na precisão e *recall* da classificação da técnica proposta.

Em (YUAN et al., 2016), os autores propõem uma melhoria na acurácia e na velocidade de execução da técnica de aprendizagem C4.5 para classificação de tráfego. Para isso, foi utilizada a plataforma Hadoop, um arcabouço de nuvem aberto capaz de permitir que a classificação seja paralelizada. Embora os ganhos de acurácia tenham sido muito pequenos, os ganhos obtidos no tempo de execução foram bastante interessantes. É importante notar que, assim como outros trabalhos, esse trabalho foca em melhorar o método utilizado tradicionalmente, sem propor nenhuma alteração no modelo de classificação amplamente utilizado atualmente.

Em (CARELA-ESPAÑOL et al., 2013), uma arquitetura é proposta e descrita. Essa arquitetura trata o classificador como um *middlebox* que será implantado na rede. Basicamente, são 3 características destacadas ao longo do trabalho. A primeira é a utilização de *Sampled* Netflow para exportar fluxos IP dos roteadores e *switches* da rede. Já a segunda característica trata do método de classificação que combina vantagens de múltiplas técnicas, minimizando algumas limitações inerentes ao processo de classificação. Por fim, o terceiro aspecto proposto é o sistema de retreinamento autônomo. No módulo de classificação propriamente dito (segunda característica listada), são utilizadas técnicas de aprendizagem de máquina e a classificação final reflete a combinação de três abordagens distintas. A primeira é uma técnica baseada na identificação do endereço IP contido no fluxo. Quando algum desses IPs pertence a algum *site* bastante popular, o fluxo será identificado. Além

desse, o segundo método também considera o IP, mas junto com a porta e o protocolo, compondo a tripla que define serviços (<IP, porta, protocolo>). Caso o serviço seja previamente conhecido, o fluxo será classificado. Ambas as abordagens descritas até aqui para classificação podem ocorrer de maneira *offline*. Finalmente, a última etapa da classificação é uma técnica de aprendizagem de máquina chamada C5.0 (QUINLAN, 2004), uma evolução do C4.5. Essas três técnicas fornecem a sua classificação associada a um valor de confiança. Aquela que tiver o maior valor de confiança fornecerá a classificação final do fluxo. Para obtenção do conjunto de treinamento, utilizado tanto para treinar a técnica de aprendizagem de máquina da primeira vez, como para realizar o retreinamento, um conjunto de classificadores DPI é utilizado para fornecer uma base confiável de dados. Dentre esses classificadores é possível destacar o PACE[10], o OpenDPI[11] e o L7-filter[12]. O módulo de retreinamento dessa arquitetura é bastante interessante. Nele, existem dois classificadores que operam a fim de possibilitar a avaliação da classificação que está em curso. Enquanto o tráfego é classificado pelo C5.0, algumas amostras de fluxos são enviadas ao módulo de retreinamento e são classificadas por um DPI e por uma árvore C5.0, idêntica à do módulo de classificação. Caso eles mantenham um alto percentual de classificações iguais, acima de um determinado limiar, não há alteração na operação. Caso contrário, um novo treinamento da técnica de aprendizagem de máquina ocorre, gerando uma nova árvore de decisão que operará no módulo de classificação. Na avaliação dessa pesquisa, os autores sugerem que apenas um DPI limitaria a classificação em seu cenário, sendo menos eficiente. Em resumo, a arquitetura proposta, embora limitada a um ponto específico da rede, permite uma adaptação dinâmica às mudanças do tráfego, além de demonstrar que a combinação de técnicas permite o ganho de precisão com alto grau de completude. Essa arquitetura foi posta em produção em uma rede de pesquisa da Catalunha (CESCA).

Seguindo na linha de combinação de técnicas de classificação, os autores em (CALLADO et al., 2010) propuseram quatro métodos capazes de combinar algoritmos de classificação, partindo do pressuposto de que não há uma técnica que seja melhor em todos os cenários. Destaca-se ainda que, de acordo com os autores, a complexidade associada à realização da combinação é muito pequena quando comparada à complexidade dos

[10] PACE, ipoque's Protocol and Application Classification Engine: http://www.ipoque.com/. Acessado em Janeiro de 2017.
[11] OpenDPI, The Open Source Deep Packet Inspection Engine: http://www.opendpi.org/. Acessado em Janeiro de 2017.
[12] L7-filter, Application Layer Packet Classifier for Linux: http://l7-filter.clearfoundation.com/. Acessado em Janeiro de 2017.

algoritmos de classificação. Nesse trabalho, são apresentados os seguintes métodos de combinação: (i) *Random combination (RC)*, (ii) *Maximum likelihood combination (MLC),* (iii) *Dempster-Shafer combination (DSC)* e (iv) *Enhanced Dempster-Shafer combination (EDSC)*. O primeiro dos métodos, o RC, consiste na escolha aleatória do classificador que será aplicado a um determinado fluxo. Dessa forma, os resultados variam bastante até que o número de fluxos aumente consideravelmente. Nesse ponto, a acurácia tanto em relação a *bytes* como em relação a fluxos passa a se aproximar da média da acurácia dos classificadores envolvidos. O RC serve como uma base de comparação para as demais técnicas utilizadas. Já o MLC é o primeiro dos métodos a tentar maximizar a combinação dos resultados dos diferentes classificadores. Esse método promove uma "votação" entre os classificadores e o resultado mais votado é a classificação final do fluxo. Embora trate de diversas abordagens, o MLC parte do pressuposto que os algoritmos de classificação erram isoladamente, permitindo que a maioria consiga superar esse erro. Outro método utilizado nesse trabalho foi o DSC. Ele se baseia na Teoria da Evidência (GUAN et al., 1991), que permite quantificar a evidência de determinado evento com base no cálculo de funções de confiança. Para a operação desse método, é necessário que o grau de corretude da identificação de tráfego de cada algoritmo em relação a um fluxo seja armazenado previamente. Em outras palavras, é preciso que para cada fluxo exista a informação de qual técnica é mais precisa. Dessa forma, para cada fluxo a ser classificado, o classificador mais apropriado será escolhido (dentre aqueles que não apontarem o fluxo como desconhecido). Por fim, o EDSC é bastante similar ao DSC, apresentando uma pequena variação. Nele, a informação previamente armazenada é relacionada ao resultado da classificação em vez do fluxo. Isto é, o algoritmo que apresentar o maior nível de acerto para uma determinada classe (resultado) será escolhido como classificador do fluxo analisado. Além dos métodos de combinação utilizados, os autores discutem também o conceito de combinação perfeita – cada fluxo será classificado da melhor forma possível, recebendo a classificação correta sempre que ao menos um classificador o identifique corretamente. Essa combinação define o limite superior que a eficácia de uma combinação pode alcançar, ou seja, nenhuma combinação apresentará resultados melhores que ela. Ainda segundo os autores, esse trabalho demonstra a capacidade de alcançar resultados tão bons ou melhores que o obtido com o melhor algoritmo de classificação para cada cenário. Embora consiga mostrar melhorias, em geral, os resultados apresentados nesse artigo não superam 90% de acurácia da classificação ou corretude.

Em (TUMER et al., 1996), os autores discutem a combinação de classificadores sob uma diferente perspectiva. Para eles, escolher apenas o melhor classificador pode implicar na perda de informações valiosas para a classificação. Combinar os resultados dos classificadores se apresenta, portanto, como uma interessante alternativa. Nesse trabalho foram elencados vários métodos como, por exemplo, esquema de votação, média ponderada, escolha baseada em *ranking*, método baseado em grau de confiança, dentre outros. Esses esquemas, em geral, visam otimizar o desempenho computacional e a precisão da classificação. No entanto, a principal motivação para o desenvolvimento do trabalho é que, independente do método de combinação utilizado, caso os classificadores sejam muito correlacionados (apresentando os mesmos erros e/ou acertos), o ganho agregado no resultado da combinação será pequeno. Uma alternativa para minimizar esse problema é analisar a seleção e treinamento dos classificadores que estarão envolvidos na classificação. Nesse trabalho foram avaliados quatro métodos para reduzir a correlação entre os classificadores individuais. O primeiro método corresponde ao particionamento do conjunto de treinamento para que cada classificador utilize um subconjunto distinto em seu treinamento. O segundo método se baseia na poda das entradas para construção de novos conjuntos de treinamento. O terceiro remete à geração de novos conjuntos de treinamento a partir de métodos de amostragem. O último método descrito pelos autores considera o particionamento do conjunto de treinamento, mas isso não é feito aleatoriamente como anteriormente, a divisão é feita de maneira espacial. Nesse caso, existe uma redução da complexidade de cada classificador. Diante das técnicas de redução da correlação propostas, fica evidente que, para reduzir a correlação entre classificadores, é necessária também a redução da performance individual de um classificador, uma vez que ele é treinado com menos dados. Após a avaliação realizada no trabalho, os autores ratificaram a dificuldade de realização de melhorias de performance através da combinação de técnicas. Em especial, as técnicas propostas nesse trabalho dependem também do tamanho do conjunto de treinamento que será utilizado. Quando o conjunto de treinamento é pequeno, subdividi-lo torna-o ainda menor, deteriorando de maneira substancial a precisão dos classificadores individualmente. Caso o conjunto de treinamento seja suficientemente grande, tanto a técnica de subdivisão do conjunto de treinamento, quanto a amostragem do conjunto apresentam resultados interessantes, sendo desejáveis nesse cenário. Dessa forma, percebe-se que a eficiência da combinação de diferentes técnicas de classificação é uma abordagem interessante, mas depende de algumas características específicas, considerando

desde o método de combinação de resultados ao método de treinamento dos classificadores individuais.

Do mesmo modo que resultados da classificação de diferentes técnicas podem ser combinados, um mesmo pacote pode receber várias classificações diferentes e todas estarem corretas. Em (XU et al., 2014), os autores defendem a classificação com múltiplos resultados em detrimento da classificação que utiliza apenas o melhor resultado. Em geral, trabalhos relacionados à classificação múltipla são avaliados em relação à classificação, mas não ao desempenho computacional. O trabalho utiliza TCAM (*Ternary Content Addressable Memory*) para paralelizar o processo de obtenção de vários resultados para um mesmo pacote. Nesse trabalho, a classificação múltipla é tratada como a concatenação de classificações individuais e utiliza uma árvore de assinaturas para fazê-la. Em cada nível da árvore, podem haver *matches* com uma assinatura ou não. Devido ao alto grau de paralelismo da técnica proposta, a velocidade de classificação é muito alta, mesmo com os baixos requisitos de armazenamento. Os resultados das simulações demonstraram uma velocidade superior a outras abordagens avaliadas, mas utilização de recursos de armazenamento similares.

Já no contexto de detecção de intrusão, os autores em (SCHUSTER et al., 2012) propuseram um IDS distribuído para redes de automação industrial. Na área de automação industrial, existem ferramentas de supervisão do controle e aquisição de dados (SCADA). A administração e monitoramento de ambientes de automação com SCADA é realizada normalmente através da associação a ferramentas específicas de monitoramento de redes. No entanto, essa associação expõe as ferramentas de automação a todas as vulnerabilidades inerentes às funcionalidades de tecnologia da informação que forem utilizadas. A proposta dos autores consiste na inclusão de SCIDs (sistemas de detecção de intrusão SCADA) na infraestrutura SCADA. Os múltiplos SCIDs são postos tanto na camada de controle quanto nas máquinas de operação de fato. Cada um desses detectores de intrusão é responsável pelo tráfego que observam. Quando o tráfego é analisado e uma anomalia é detectada, a informação de segurança é propagada para os outros detectores de intrusão. Nesse caso, mesmo ataques difusos, que pretendem passar despercebidos por utilizarem dispositivos aparentemente sem relação, são descobertos com base na propagação de eventos entre os SCIDs. Além disso, a completa distribuição sem um ponto central de operação permite que não haja um único ponto de falha. Finalmente, essa arquitetura permite que outras operações sejam executadas onde estiver implantado o SCID, como um DPI, por exemplo. Embora necessite de maior aprofundamento na avaliação da arquitetura

proposta, os autores possibilitaram a visualização de potenciais benefícios da detecção distribuída de anomalias em redes de automação, o que pode ser extrapolado para a classificação de tráfego de redes tradicionais de computadores.

Em (ORIOLA et al., 2012), são discutidos alguns desafios importantes do projeto de sistemas de detecção de intrusão distribuídos, dessa vez aplicados em um cenário de redes de computadores tradicional. Os desafios abordados são divididos em algumas áreas: (i) compartilhamento de dados, (ii) natureza da análise dos dados e (iii) características de segurança e confiança. No compartilhamento de dados, é possível observar uma abordagem centralizada e outra totalmente descentralizada. Na primeira abordagem, vários agentes enviam informações para um único ponto em que essas informações são consolidadas e analisadas. Nesse caso, os dois problemas que se destacam são a existência de um ponto único de falha e a escalabilidade da solução. Uma alternativa ao modelo totalmente centralizado está no controle hierárquico (PORRAS et al., 1997). Já no caso da detecção distribuída, segundo os autores, as soluções são mais raras e pouco desenvolvidas. Existem várias técnicas de compartilhamento de informações, mas nenhuma delas costuma ser aplicada no contexto de IDS distribuído. Quanto à natureza da análise dos dados, ela pode acontecer tanto através de conjuntos de regras específicos ou através de esquemas de limiar. No caso dos conjuntos de regras, as políticas de segurança, comportamentos normais e anômalos, além de alertas, são relacionados à existência de alguns eventos pré-definidos nas regras. Já no cenário da utilização de esquemas de limiar, cada ocorrência de um determinado evento de segurança aumenta o nível de alerta do sistema, bem como a ausência de eventos durante algum tempo deve diminuir esse nível. As características de segurança e confiança de sistemas distribuídos de detecção de intrusão são muito importantes embora não sejam priorizadas em face das questões de projeto descritas anteriormente. Como haverá comunicação entre os detectores ou entre os detectores e o controle, é essencial que se possa confiar nessas informações que estão sendo trocadas. Adicionalmente, os autores destacam que a interação entre agentes de detecção não é muito realizada, sendo uma das contribuições desse trabalho. Por fim, o conceito proposto e avaliado é a integração de agentes de mineração P2P.

Recentemente, os autores em (CANZIAN et al., 2015) propuseram o estudo de uma rede de classificadores para extrair conhecimento em tempo real de um conjunto de dados considerando um cenário de mineração de fluxo de dados e suas restrições. As contribuições do artigo são, basicamente, o levantamento bibliográfico de trabalhos que lidam com redes de classificadores, a apresentação de desafios e limitações da área de

mineração de fluxos de dados e, por fim, a discussão das possíveis direções da pesquisa nessa área e das questões de pesquisa em aberto. Quando se discute a possibilidade de implementação de uma rede de classificadores reais, alguns problemas são apresentados e são bastante relevantes para este trabalho. O primeiro problema está relacionado à ocorrência de um erro em um classificador específico. Isso será refletido no restante da rede, aumentando a necessidade de processamento ou na propagação do erro para o resultado final. Nesse caso, existe a necessidade de otimização de acurácia e performance em conjunto. Além disso, diferentes configurações ou tipos de classificadores podem coexistir na rede de classificadores. Isso possibilita que classificadores com diferentes tipos de erros sejam complementares. Dessa forma, a ordenação ideal dos classificadores depende das características do classificador. É importante destacar que, de acordo com os autores, abordagens de redes de classificadores não suportam adaptação dinâmica às características do ambiente de classificação. Adicionalmente, essas abordagens são baseadas em uma entidade centralizada, que limita a escalabilidade e adaptividade, estão sujeitas a um único ponto de falha e dependem de trocas de informações entre os nós de classificação. Por fim, os autores sugerem ainda a otimização do tempo para classificação de fluxos de dados, devido às restrições de tempo, através da paralelização do aprendizado dos classificadores.

Ainda extrapolando o cenário de redes de computadores, é possível observar que a classificação distribuída já existe em outros campos de conhecimento, com algumas características que podem ser adaptadas. Em (THAJCHAYAPONG et al., 2013), é proposto um novo algoritmo que, de maneira distribuída e utilizando variáveis microscópicas, classifica anomalias no tráfego de veículos. Anomalia nesse ambiente é tudo aquilo que não é comportamento esperado. Variáveis microscópicas consideradas nesse trabalho são aquelas que remetem ao comportamento ou interação de veículos individuais, como velocidade relativa e espaço entre veículos, por exemplo. Nesse trabalho, o monitoramento distribuído se refere ao processo em que anomalias são detectadas e classificadas através dos veículos e estradas, sem enviar informações para um centro de gerenciamento. O algoritmo apresenta resultados satisfatórios para os autores, sendo interessante sua implementação. Embora não se trate de um trabalho na área de redes de computadores, é bastante simples o mapeamento desse problema para o cenário de redes de computadores. Nesse caso, o algoritmo fornece um novo ponto de vista em relação à classificação distribuída.

3.1.2 Sumário da classificação de tráfego

Mediante o levantamento bibliográfico realizado, pôde ser visto que a área de classificação de tráfego não apresenta soluções distribuídas, perdendo os benefícios desse tipo de abordagem. No entanto, ferramentas para detecção de intrusão já demonstraram a eficiência e eficácia de abordagens distribuídas tanto na área de redes de computadores como em outros ambientes. A literatura ainda discute as formas de interação entre classificadores. Especificamente, destaca importantes questões sobre a combinação de técnicas de classificação, característica desejável para a classificação distribuída. Alguns aspectos merecem destaque, como a necessidade de baixa correlação entre classificadores que colaboram entre si e a simplicidade no método de colaboração. Observando as abordagens de redes de classificadores, mesmo não sendo aplicadas especificamente no cenário de redes de computadores, é possível perceber que classificadores distribuídos apresentam uma série de benefícios, mas possuem também várias questões de pesquisa em aberto. Por fim, diferentes arquiteturas foram propostas para otimizar a operação de um classificador e alguns conceitos podem ser reproduzidos e utilizados no cenário deste trabalho. Na Tabela 1.1, é apresentado o quadro resumo dos trabalhos relacionados com as principais pesquisas que se relacionam com esta tese (destacando-se, em especial, a abordagem da arquitetura e a interação entre os classificadores).

Tabela 1.1 – Quadro resumo da classificação de tráfego atual

Trabalho	Finalidade	Abordagem da arquitetura	Interação entre os classificadores	Tecnologias utilizadas
(CARELA-ESPAÑOL et al., 2013)	Classificação de tráfego	Distribuída	Não apresenta	Aprendizagem de máquina; DPI
(CALLADO et al., 2010)	Classificação de tráfego	Centralizada	RC, MLC, DSC e EDSC	Aprendizagem de máquina
(TUMER et al., 1996)	Classificação genérica	Centralizada	Votação, média ponderada, *ranking,* grau de confiança, dentre outros	Aprendizagem de máquina

(XU et al., 2014)	Classificação de tráfego	Centralizada	Múltiplas classificações	DPI
(ORIOLA et al., 2012)	Detecção de intrusão	Distribuída	Não apresenta	IDS

3.2 SDN e NFV no contexto da classificação de tráfego

Nas subseções a seguir, serão apresentados trabalhos relacionados a este trabalho que tratam da classificação de tráfego com suporte dos paradigmas SDN e NFV, além de um resumo com as principais características desses trabalhos que se relacionam com esta tese.

3.2.1 Trabalhos relacionados na área de classificação de tráfego suportados por SDN e NFV

Nesta subseção, o foco está nas interações entre SDN e NFV e de como funções da rede podem se beneficiar disso. Especificamente no contexto de classificação de tráfego, o paradigma SDN é aplicado para gerenciar e melhorar a classificação em poucos trabalhos, com ideias interessantes, mas, em geral, pouco desenvolvidas. Por outro lado, NFV apresenta isoladamente mais aplicações na área de classificação de tráfego. Diante do levantamento bibliográfico realizado, poucos trabalhos aplicam em conjunto esses paradigmas para efetuar a classificação de tráfego e, até onde a pesquisa foi realizada, nenhum que adote uma abordagem distribuída para isso.

Inicialmente, em (NG et al., 2015), é descrita uma abordagem que pretende fazer uso de uma rede SDN para construir um classificador de tráfego. Os autores apresentaram seu trabalho como um passo inicial nessa área. Nele, reportam experiências práticas e lições aprendidas com a implementação da classificação suportada por uma rede SDN. Em síntese, os autores perceberam no controlador SDN uma possibilidade de centralização lógica da classificação através de uma ampla visão do estado do fluxo na rede. Para isso, é prevista a possibilidade de adição de *software* ao plano de controle para execução da classificação, bem como adição de *hardware* para otimização caso necessário. A proposta desse trabalho é descrever como é possível realizar a classificação utilizando SDN, sem avaliar algoritmos de classificação ou mesmo performance. Uma interessante contribuição desse trabalho está no surgimento da ideia de customização do protocolo OpenFlow (cada vez mais popular e estabelecido na área) para atender aos requisitos da classificação. Além

da proposta conceitual, os autores propõem a criação de um sistema para realização da classificação denominado nmeta (*Network Metadata*). Em resumo, esse sistema é construído em cima do controlador SDN, recebendo suas informações sobre o fluxo e obtendo uma classificação. Por se tratar de um ponto centralizado de classificação, os autores reportam uma degradação de desempenho importante do classificador e do controlador devido à necessidade de constante troca de informações, o que limita a atuação desse sistema. Essa pesquisa apresentou os primeiros passos em direção à realização da classificação de tráfego com suporte do paradigma SDN, sendo importante por trazer a possibilidade de serem utilizados em conjunto. No entanto, ainda existem várias questões de pesquisa em aberto que não foram abordadas nesse trabalho, como, por exemplo, a imposição de um único gargalo para a classificação e a sobrecarga gerada pela comunicação adicional necessária.

Em (AMARAL et al., 2016), é realizado um estudo sobre a aplicabilidade de técnicas de aprendizagem de máquina para classificar tráfego em uma rede SDN. Os autores propuseram a coleta de estatísticas sobre os fluxos com o protocolo OpenFlow. Com base em informações coletadas por padrão em uma rede SDN, várias técnicas de aprendizagem supervisionada foram aplicadas aos dados coletados para realizar a classificação. Os resultados obtidos apresentam uma alta acurácia das técnicas aplicadas, alcançando o objetivo de identificação de tráfego da rede monitorada.

Os trabalhos apresentados em (NG et al., 2015) e (AMARAL et al., 2016), apresentam esforços iniciais na condução de uma classificação de tráfego através do paradigma SDN. O primeiro trabalho trata da viabilidade da realização da classificação propriamente dita baseando-se no controle SDN (do ponto de vista arquitetural), enquanto a segunda pesquisa discute se o tipo e o formato de informação coletada em uma rede SDN são suficientes para realizar uma classificação de tráfego com aprendizagem de máquina supervisionada. Os trabalhos atacam questões distintas e complementares, essenciais à aplicação das características de uma rede SDN ao problema de classificação de tráfego.

Alguns trabalhos apresentam estudos que fazem uso do potencial de controladores SDN para coordenar a classificação de tráfego na camada de aplicação. Em (LI et al., 2014), os autores combinam duas das principais técnicas de classificação de tráfego, DPI e aprendizagem de máquina. Nesse caso, uma arquitetura SDN é proposta para que sejam possíveis as trocas de informações entre classificadores, controlador e plano de dados. O objetivo dos autores é manter uma alta precisão enquanto a velocidade de classificação é aumentada. Basicamente, a estrutura da multiclassificação pode ser descrita da seguinte

forma: a técnica de aprendizagem de máquina classifica os pacotes e, caso o grau de confiabilidade seja maior que um limiar previamente definido, o pacote receberá a classificação dessa técnica. Caso o grau de confiabilidade não seja superior ao limiar estabelecido, o pacote é repassado ao DPI e receberá a sua classificação (desde que o DPI seja capaz de atribuir uma classificação ao pacote). Dessa forma, essa abordagem pretende ser capaz de aproximar sua precisão do melhor caso (classificação do DPI), reduzindo o tempo necessário para processamento dos pacotes (o DPI é bastante custoso computacionalmente). Os resultados apresentados evidenciam que a proposta consegue uma alta acurácia e vazão, aproximando-se do limite superior nos dois casos. Embora essa proposta combine mais de uma técnica de classificação, ela não se trata de uma classificação distribuída e sim multinível. A classificação se dá da mesma forma que acontece tradicionalmente. No entanto, quando o primeiro nível de classificação (aprendizagem de máquina) não atende a um determinado nível de confiabilidade, a classificação é realizada por um segundo nível (DPI). Outros trabalhos apresentam soluções similares, como em (JARSCHEL et al., 2013) e (QAZI et al., 2013).

Por outro lado, alguns trabalhos já foram propostos ligando SDN a outras atividades relacionadas ao monitoramento e gerenciamento de redes. Os autores de (YU et al., 2014), por exemplo, propuseram um sistema com uso eficiente de memória para monitoramento distribuído e colaborativo de fluxos, denominado DCM (*Distributed Colaborative Monitoring*). Esse sistema faz uso de *Bloom filters* (BLOOM, 1970) para representar regras de monitoramento que utilizam pouca memória. O DCM se sustenta na conveniência proporcionada pelo paradigma SDN para implantar uma ferramenta de monitoramento dinâmica e customizável no plano de dados da rede. Nesse caso, a SDN permite que além da operação, o DCM seja atualizado e reconstruído logicamente no plano de encaminhamento de dados. Além disso, devido à visão geral da rede fornecida pelo plano de controle, na figura do controlador SDN, é possível detectar e minimizar a influência de falsos positivos no monitoramento. O dispositivo de encaminhamento, por sua vez, processa pacotes e, quando solicitado pelo controlador, armazena informações de monitoramento localmente. Periodicamente, as informações armazenadas são enviadas ao controlador que realiza as tomadas de decisão de acordo com o fluxo monitorado. Um dos principais benefícios para o DCM de ter um plano de controle que conhece toda a rede é a possibilidade do controlador de escolher em que dispositivo cada fluxo será monitorado. Dessa forma, é possível prover uma distribuição de carga conveniente ao gerenciamento. Essa abordagem permite que a sobrecarga de processamento e memória seja minimizada

nos dispositivos de encaminhamento, bem como a precisão do que é monitorado seja maior, já que um menor volume será observado. Os autores, através da realização de experimentos com tráfego e topologias reais, demonstraram que o DCM apresenta resultados do monitoramento de fluxos precisos e com uso eficiente de memória para duas atividades em especial, contagem do tamanho do fluxo e amostragem de pacotes. Quando comparado a outras soluções o DCM obtém uma ampla cobertura dos fluxos da rede e uma alta precisão das medições utilizando a mesma quantidade de memória. Nesse trabalho, ficam evidentes os benefícios proporcionados por uma abordagem de monitoramento distribuída, como o balanceamento de carga, por exemplo. Além disso, é possível perceber que a construção do DCM só é viável mediante a implantação de um controlador SDN que o suporte. Isso acontece porque a visão geral da rede se faz necessária para conciliar diversos pontos de monitoramento, além de realizar atualizações e melhorias de maneira dinâmica.

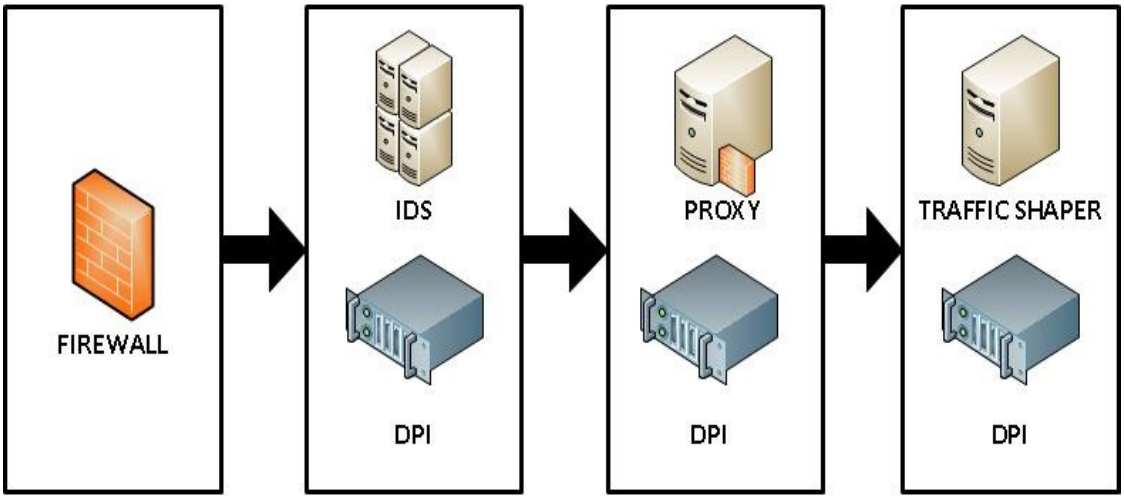
Já em (AGARWAL et al., 2013), é estudada a efetiva implantação de SDN em uma rede tradicional para implementação de engenharia de tráfego. A proposta dessa pesquisa é demonstrar como um controle central da rede pode minimizar perda de pacotes e atrasos, bem como melhorar a utilização da rede. Para isso, os autores formularam o problema de otimização do controlador SDN para engenharia de tráfego com implantação parcial e desenvolveram o FPTAS (*Fully Polynomial Time Approximation Scheme*) para resolver o problema. Decidir quando é possível realizar alguma engenharia de tráfego de maneira efetiva, uma vez que nem toda a rede pode ser controlada centralmente por um controlador SDN, é o ponto central abordado. Esse trabalho é desenvolvido em um cenário em que apenas alguns nós da rede (dispositivos de encaminhamento) são controlados por um controlador SDN, os demais ativos da rede funcionam com algum protocolo de encaminhamento como o OSPF, por exemplo. Esse cenário é bastante relevante quando se considera a implantação de SDN em redes tradicionais, nem todos os nós da rede costumam ser modificados e, portanto, não seguem o novo paradigma. Isso acontece principalmente em redes de grandes dimensões. Em síntese, o objetivo dos autores desse trabalho é o desenvolvimento de uma arquitetura SDN que pode ser dinâmica e adaptativamente gerenciar o tráfego da rede, adaptando-se a diferentes padrões de tráfego. A interação entre os controladores SDN (referenciados como SDN-C pelos autores) e os elementos de encaminhamento (referenciados como SDN-FE) permite que o sistema proposto tenha uma visão geral do tráfego da rede e, com isso, possa tomar decisões sobre o tráfego. Essas decisões definem a operação dos elementos de

encaminhamento e podem afetá-los isoladamente ou em conjunto. Os experimentos demonstraram que mesmo posicionando estrategicamente poucos elementos de encaminhamento gerenciáveis, a performance da rede é melhorada. É importante salientar que o esquema proposto não prevê nenhuma alteração de protocolo nos nós que se mantêm da forma tradicional. Tanto os experimentos quanto as simulações realizadas expõem um ganho significativo na vazão, bem como minimizam o atraso e as perdas de pacotes na rede.

Considerando a classificação de tráfego suportada pelo paradigma NFV, é possível destacar o arcabouço vTC (HE et al., 2016). Esse arcabouço defende a ideia de que diferentes modelos de classificadores são mais efetivos de acordo com os protocolos ou características coletadas dos fluxos. Nesse cenário, o paradigma NFV provê flexibilidade e escalabilidade para a seleção de classificadores em tempo de execução. Os resultados consideraram diversas técnicas tradicionais de aprendizagem de máquina, demonstrando um ganho na acurácia de até 13%. Esse ganho ratifica a possibilidade de ganho de acurácia com classificadores colaborativos (o que não foi proposto em (HE et al., 2016)). Outros trabalhos apresentam a possibilidade de uso de um classificador de tráfego como função virtual, como observado em (BOUET et al., 2015).

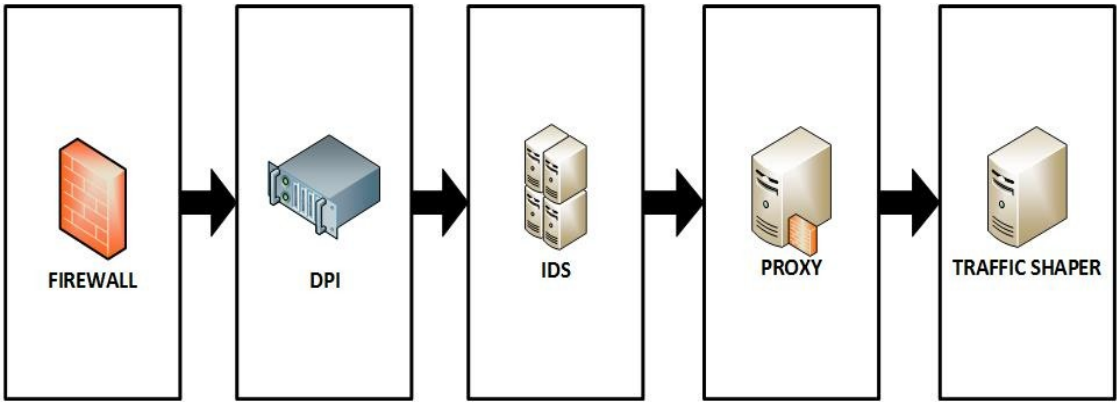
Algumas pesquisas apresentam a classificação de tráfego através da integração entre SDN e NFV. Em (BREMLER-BARR et al., 2014), os autores do trabalho discutem a possibilidade de tratar a classificação de tráfego através de um DPI como um serviço. Para lidar com o DPI virtual também são utilizados os conceitos de NFV. Nesse trabalho, é possível perceber que foram feitos muitos investimentos para que DPIs em geral fossem otimizados. Apesar disso, decidir quando o classificador deve ser executado, além de minimizar o impacto de seguidas classificações, é uma vantagem teórica considerável quando se tem a figura do controlador SDN. A vantagem da abordagem proposta nesse trabalho fica evidente observando a Figura 1.9. A Figura 1.9(a) exibe o cenário atual, enquanto a Figura 1.9(b) expõe a proposta. No primeiro caso, seguidas classificações são necessárias para atender as demandas dos *middleboxes* em cadeia, como o IDS e o modelador de tráfego, por exemplo. Já no segundo cenário, a virtualização do serviço do DPI permite que uma única classificação seja realizada e sua saída 'alimente' os serviços subsequentes que necessitam conhecimento sobre o tráfego. Essa abordagem nitidamente proporciona um aumento de performance da rede, uma vez que reduz o número de classificações significativamente, além de reduzir o consumo de memória decorrente da utilização de classificadores. Além disso, isolar o classificador de outros serviços possibilita

uma maior especialização do DPI. Isso permite que otimizações próprias para o serviço de classificação sejam realizadas. Nesse caso, é possível inclusive fornecer um serviço mais robusto sem oferecer um único ponto de falha (em alguns casos, serviços de segurança com *engines* de DPI embutidas foram alvo de DoS[13]). Os autores em (BREMLER-BARR et al., 2014) propuseram um *framework* para implantação de um DPI como serviço, juntamente com um algoritmo que combina diferentes origens de padrões de entrada. Adicionalmente, também realizaram a implementação de uma SDN para que fosse viável a instanciação do DPI como serviço. O papel do controlador SDN é realizar a comunicação com as *middleboxes* que serão instanciadas para que seja identificada a necessidade de um DPI e essa necessidade seja atendida, além de controlar os recursos utilizados pelo inspetor de pacotes. Essa tarefa é realizada sob demanda, de maneira individual para cada um dos nós da rede que necessitam de determinados serviços. Isso viabiliza a implantação de classificadores distintos pelo controlador de acordo com o tipo de classificação que é necessária. Os autores demonstraram ainda melhorias de desempenho através de seus experimentos e uma arquitetura flexível e escalável. Esse trabalho apresentou uma nova perspectiva para a classificação de tráfego baseada em NFV e SDN. Mesmo tendo direcionado seus esforços para virtualização de um DPI como serviço, as análises e propostas apresentadas podem facilmente ser extrapoladas para outros tipos de classificadores.



(a)

[13] Sony Ericsson Latest Victim of SQL Injection Attack. http://www.eweek.com/c/a/Security/Sony-Data-Breach-Was-Camouflaged-by-Anonymous-DDoS-Attack-807651, 2011. Acessado em Maio de 2017.



(b)

Figura 1.9 – Cadeias de *middleboxes*. (a) Cenário com vários classificadores. (b) Cenário com um único classificador (BREMLER-BARR et al., 2014).

No entanto, ao contrário do que é proposto neste trabalho, a abordagem dos autores de (BREMLER-BARR et al., 2014) considera apenas os DPIs isoladamente, avaliando apenas a necessidade de cada cadeia de *middleboxes* individualmente. Não há nenhum tipo de integração entre os diferentes classificadores, seja para realizar um melhor gerenciamento do tráfego ou mesmo para alcançar uma maior precisão das ferramentas. Essa característica poderia ser alcançada através da operação do controlador SDN conforme hipótese e proposta deste trabalho que será detalhada a seguir.

Adicionalmente, a arquitetura SDNFV pode ser bastante útil para outras funções da rede além da classificação de tráfego. Em (DING et al., 2015), os autores apresentaram uma plataforma denominada OpenSCaaS para prover o encadeamento de serviços como um serviço, discutindo a integração entre SDN e NFV. Nessa pesquisa, o encaminhamento do tráfego está relacionado à cadeia de serviços desejada. O conhecimento sobre a rede é pertinente à SDN, enquanto a flexibilidade é provida pelo NFV, que em conjunto permitem a otimização da aplicação dos serviços encadeados.

3.2.2 Sumário de SDN e NFV no contexto de classificação de tráfego

É possível perceber que vários esforços envolvendo o paradigma SDN e o paradigma NFV têm surgido para melhorar o gerenciamento e o monitoramento do tráfego de redes de computadores.

Em geral, o controlador SDN surgiu como potencial gerente desse tipo de operação, principalmente para tomadas de decisão relacionadas ao encaminhamento de pacotes. Adicionalmente, alguns trabalhos tratam da classificação orquestrada pelo plano de

controle SDN, seja do ponto de vista de virtualização da solução ou combinação de técnicas. Em geral, os trabalhos que relacionam SDN e a classificação de tráfego propriamente dita ainda são esforços iniciais, que trazem mais possibilidades em aberto que soluções de fato. Por outro lado, a aplicação do paradigma NFV para classificação de tráfego surge como uma área de pesquisa mais consolidada. O classificador virtualizado, no entanto, não difere do modelo tradicional de classificação, senão pela composição do classificador (físico X virtual). Dessa forma, diante do estudo bibliográfico realizado, não há pesquisa reportada que utilize o poder dos paradigmas SDN e NFV para, além de melhorar o desempenho da classificação, implementar uma classificação distribuída. Isso é uma lacuna em aberto, que será explorada neste trabalho.

Na Tabela 1.2, é apresentado o quadro resumo dos principais trabalhos relacionados levando-se em consideração a interação entre os paradigmas SDN e NFV e a classificação de tráfego. Foram listadas as pesquisas que se apresentam mais correlacionadas com esta tese. É importante destacar que todos os trabalhos apresentados neste quadro resumo tratam do problema de classificação de tráfego. Finalmente, percebe-se na Tabela 1.2 que não foram apresentados trabalhos que apresentem uma abordagem distribuída, com interação entre SDN e NFV e interação entre classificadores.

Tabela 1.2 – Quadro resumo dos trabalhos relacionados considerando a interação entre SDN, NFV e classificação de tráfego

Trabalho	Interação entre SDN e NFV	Abordagem da arquitetura	Interação entre classificadores	Paradigmas utilizados
(NG et al., 2015)	Não apresenta	Centralizada	Não apresenta	SDN
(AMARAL et al., 2016)	Não apresenta	Centralizada	Não apresenta	SDN
(LI et al., 2014)	Não apresenta	Centralizada	Classificação multinível	SDN
(HE et al., 2016)	Não apresenta	Centralizada	Não apresenta	NFV

(BREMLER-BARR et al., 2014)	SDN decide detalhes da instanciação do classificador virtual	Distribuída	Não apresenta	SDN e NFV

3.3 Posicionamento de servidores para classificação de tráfego

Nas subseções a seguir, serão apresentados trabalhos relacionados a este trabalho que tratam do posicionamento de servidores para a classificação de tráfego suportada, além de um resumo com as principais características desses trabalhos que se relacionam com esta tese.

3.3.1 Trabalhos relacionados na área de posicionamento de servidores para classificação de tráfego

O posicionamento de servidores (do inglês *Server placement*) é realizado em diversos cenários de redes de computadores e desempenha um papel muito importante para alcançar diferentes objetivos, desde a redução da latência da rede até otimizações de algoritmos de cache. No entanto, considerando a classificação de tráfego e a pesquisa bibliográfica realizada, não há registro de posicionamento de servidores para classificação de tráfego, até mesmo pela ausência de uma arquitetura de classificação distribuída. Nesse caso, decidir onde posicionar os classificadores para obter a maior cobertura do tráfego possível é um desafio que surge junto à classificação distribuída. Embora a arquitetura proposta nesse trabalho não preveja o posicionamento físico de servidores na rede que é monitorada, alguns trabalhos que lidam com esse tipo de posicionamento podem ser importantes na construção do conhecimento. Uma vez que o posicionamento dos servidores é definido, o processo de instanciação também desempenha um papel relevante para a operação de funções virtuais na rede.

Vários trabalhos consideram o problema de posicionamento de um servidor virtual ou de uma função virtual independente do encaminhamento do tráfego da rede. Em (MOHAMMADKHAN et al., 2015), os autores lidam com o problema de posicionamento juntamente com o problema de encaminhamento dos fluxos como um

único problema de programação linear inteira mista (MILP – *mixed integer linear programming*). A ideia é obter um posicionamento ótimo dos serviços e um roteamento ótimo dos fluxos. Eles propuseram uma solução incremental, variando à medida que fluxos são inseridos na rede. A formulação do problema dessa forma permite a obtenção de um posicionamento que minimiza a utilização de *links* e nós.

A pesquisa em (BOUET et al., 2015) trata do posicionamento de DPIs virtuais em uma rede. O problema leva em consideração a existência de uma infraestrutura NFV, bem como a demanda de tráfego. Os autores propuseram um algoritmo guloso baseado na centralidade para indicar os nós mais importantes considerando a sua conectividade. A proposta visa encontrar um nó que minimize o custo geral da instanciação de um vDPI. Esse custo é definido a partir de três restrições: (i) o custo do DPI propriamente dito, (ii) o custo do encaminhamento de tráfego associado ao DPI e (iii) restrições operacionais. Em outras palavras, trata-se de um problema que visa balancear a minimização dessas três restrições. Os resultados demonstraram que a infraestrutura e as restrições influenciam fortemente o tempo necessário para o posicionamento. Além disso, alguns problemas com escalabilidade foram descritos, com o aumento exponencial do tempo quando mais de 80 nós são considerados.

É perceptível que as abordagens de posicionamento de funções virtuais necessitam de um alto grau de orquestração. Os autores em (CLAYMAN et al., 2014) propuseram uma arquitetura capaz de prover a alocação de nós virtuais, bem como das respectivas funções virtuais desejadas. Para gerenciar esses recursos virtuais, é apresentado um orquestrador que lida com recursos virtuais para obter um posicionamento automático. Nessa arquitetura, existe um módulo que detém os algoritmos de posicionamento de nós ou serviços, podendo ser baseado na infraestrutura ou em métricas virtuais da rede. Em resumo, trata-se de uma arquitetura que encapsula métodos de posicionamento para recursos virtuais em uma rede.

O posicionamento de funções virtuais da rede apresenta uma grande variedade de abordagens para solucionar o problema. Em geral, existem métodos para posicionamento de funções virtuais genéricas (CLAYMAN et al., 2014) (MOHAMMADKHAN et al., 2015) ou para funções virtuais específicas (BOUET et al., 2015). Embora forneçam direções de pesquisa interessantes, cada cenário apresenta restrições e características bastante específicas que precisam ser consideradas.

Os autores de (MA et al., 2014a) e (MA et al., 2014b) discutem e propõem um método de posicionamento de servidores de monitoramento na rede a fim de possibilitar, através de uma técnica de monitoramento, a máxima identificação da topologia monitorada com o mínimo de monitores. A motivação dessa abordagem está relacionada à descoberta das métricas da rede a cada nó, em vez do modelo tradicional que monitora o início e o final do trajeto dos pacotes da rede. Nesse caso, o monitoramento se baseia na tomografia da rede (do inglês, *Network tomography*) (COATES et al., 2002), que permite a identificação de características internas da rede. Para a identificação das métricas da topologia monitorada, os autores apontaram algumas restrições, dentre as quais a necessidade de que as métricas identificadas sejam aditivas e o tráfego não realize ciclos (um mesmo nó não aparece duas vezes em um mesmo caminho). Em relação ao posicionamento de monitores de rede, a principal contribuição desse trabalho reside na proposição de um algoritmo de posicionamento em topologias arbitrárias em $O(|L| + |V|)$, onde L é o conjunto de *links* e V é o conjunto de nós. Os resultados demonstraram que o algoritmo proposto necessita de menos monitores que a técnica utilizada como comparação. Em síntese, o algoritmo se baseia nas seguintes regras:

1. Todo nó que estiver em uma extremidade da topologia, conectado ao grafo por apenas uma aresta, deverá ser um monitor. Caso isso não seja feito, esse *link*, especificamente, não será monitorado;
2. Todo nó que faça parte apenas de um único caminho entre dois nós deverá ser um monitor. Caso contrário, os *links* adjacentes a esse nó, só poderão ser medidos em conjunto, mas não individualmente;
3. Para cada subgrafo com dois vértices de corte (pontos de articulação), ao menos um nó desse subgrafo que não seja vértice de corte deverá ser um monitor, e;
4. Para cada subgrafo com um vértice de corte, ao menos dois nós desse subgrafo devem ser monitores da rede.

Nesse caso, os autores provaram que, atendendo essas quatro regras, é possível alcançar uma cobertura máxima da topologia. Provas formais bem como o pseudocódigo do algoritmo foram apresentados no trabalho. Embora trate especificamente do problema de identificação de métricas aditivas, observando a topologia do ponto de vista dos *links*, o algoritmo proposto fornece um bom suporte para o desenvolvimento de um algoritmo de posicionamento para a classificação de tráfego, que, por sua vez, necessita que a topologia seja observada do ponto de vista dos nós da rede.

Por fim, é necessária a discussão sobre a instanciação de funções virtuais em uma rede, já que trata-se de uma etapa imprescindível para qualquer proposta envolvendo o paradigma NFV. Especificamente, é uma etapa essencial ao posicionamento de funções virtuais. Em (BONDAN et al., 2016), é feita uma comparação entre diferentes provedores de NFV. Os autores avaliaram o tempo necessário para que cada um dos provedores estivesse inicializado e apto para operar normalmente. O tempo de instanciação de uma função virtual também é avaliado em (HWANG et al., 2015). Outra métrica interessante envolvendo a instanciação de funções virtuais é o tempo de movimentação de uma função (o tempo necessário para alterar o posicionamento de uma função específica). Algumas pesquisas consideram o cenário de falhas em funções virtuais para discutir a movimentação de funções em tempo real, sendo essas funções de propósito genérico (GEMBER-JACOBSON et al., 2015).

3.3.2 Sumário do posicionamento de servidores para classificação de tráfego

Considerando que no estudo bibliográfico realizado neste trabalho não houve registro de um trabalho que propusesse a classificação de tráfego distribuída da maneira proposta nesta tese (introduzindo a colaboração entre classificadores), dificilmente seria identificado um trabalho que posicionasse vários servidores no contexto da classificação de tráfego. Entretanto, é possível perceber que várias propostas foram realizadas para posicionar servidores com diferentes funcionalidades, especialmente para monitoramento da rede. Esses esforços, mesmo não sendo específicos para o cenário de aplicação deste trabalho, permitem que exista um direcionamento no desenvolvimento da técnica de posicionamento de servidores que é necessária para esta pesquisa. É importante destacar que o posicionamento de uma maneira geral é realizado de acordo com o cenário considerado pelo administrador da rede. Em outras palavras, cada rede fornece condições e restrições específicas para a realização do posicionamento de funções virtuais. O cenário de aplicação considerado neste trabalho será descrito adiante. Finalmente, é necessário destacar também o papel fundamental da análise dos custos oriundos da instanciação de uma função virtual. Sem isso, qualquer política de posicionamento de funções virtuais torna-se ineficaz. Na Tabela 1.3, é apresentado um quadro resumo contendo os principais trabalhos relacionados considerando o posicionamento de classificadores.

Tabela 1.3 – Quadro resumo dos trabalhos relacionados considerando o posicionamento de classificadores

Trabalho	Finalidade	Colaboração entre funções	Dados de entrada para o posicionamento
(MOHAMMADKHAN et al., 2015)	Posicionamento de serviços e roteamento de fluxos	Não	Topologia e a demanda de tráfego
(BOUET et al., 2015)	Posicionamento de DPIs virtuais	Não	Topologia e a demanda de tráfego
(CLAYMAN et al., 2014)	Alocação de nós e das funções virtuais	Não	Topologia ou métricas virtuais da rede
(MA et al., 2014a) (MA et al., 2014b)	Monitorar *links* da rede	Sim. Os monitores, em conjunto, aferem métricas aditivas da rede.	Topologia
(BONDAN et al., 2016)	Estudar provedores de NFV	Não se aplica	Não se aplica

4 Uma Arquitetura Distribuída para Classificação de Tráfego baseada em SDNFV

Neste capítulo, serão apresentados alguns conceitos sobre arquitetura de sistemas e como esses conceitos são utilizados para definir a arquitetura distribuída proposta neste trabalho. Em seguida, será apresentada a forma como os paradigmas utilizados nesta tese se relacionam com a arquitetura proposta.

4.1 A arquitetura distribuída

De acordo com (PRESSMAN, 2011), uma arquitetura fornece uma visão geral, garantindo um correto entendimento do projeto. Especificamente, trata-se da maneira como um conjunto de componentes é integrado para formar um todo coeso. A fim de facilitar a compreensão da forma que a arquitetura foi concebida, é apresentada a Figura 1.10, inspirado no conceito de *Design Rationale* (DILLON, 1997).

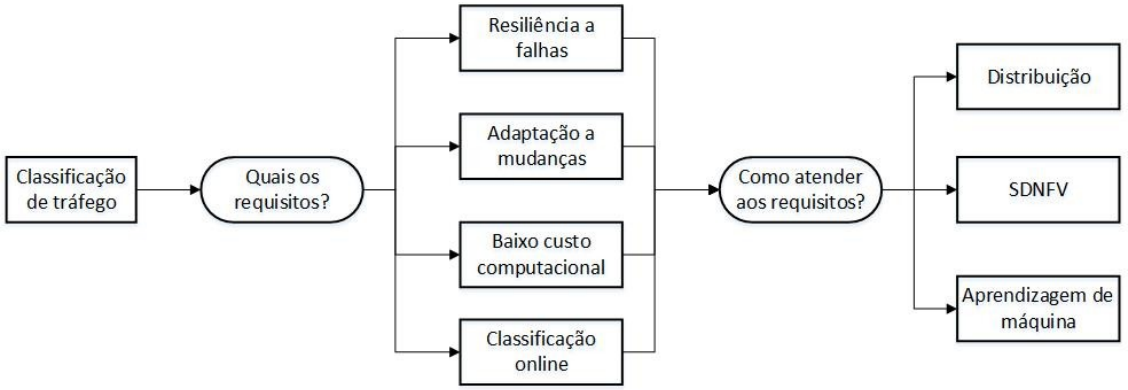


Figura 1.10 – *Design rationale* da arquitetura

Existem três diferentes tipos de requisitos ao longo de um processo de engenharia (PRESSMAN, 2011): (i) requisitos normais, (ii) requisitos esperados e (iii) requisitos fascinantes. Este trabalho é pautado pelos dois primeiros tipos de requisitos. Os requisitos normais se referem a necessidades explicitamente identificadas, enquanto os requisitos esperados são inerentes ao que está sendo produzido (mesmo que implícitos). Dessa forma, na Figura 1.10, é possível identificar o conjunto de requisitos normais relacionados à

classificação de tráfego que guiaram a concepção da arquitetura: (i) resiliência a falhas, (ii) adaptação a mudanças no perfil de tráfego, (iii) baixo custo computacional e (iv) operação *online*. Além disso, alguns requisitos esperados não são apresentados no *Design Rationale*, mas são importantíssimos e considerados no desenvolvimento da arquitetura, como: (i) alta efetividade da classificação (percentual de classificações corretas), (ii) classificação contínua e (iii) maximização do volume de tráfego classificado. Para atender esses requisitos, foi concebida uma arquitetura distribuída suportada por uma infraestrutura SDNFV e com classificadores baseados em técnicas de aprendizagem de máquina.

Os módulos que compõem o processo de classificação são:

i. Definição dos classificadores – O administrador da rede define quantos classificadores serão distribuídos na rede (de acordo com recursos e restrições da rede) e a técnica de aprendizagem de máquina que será utilizada pelos classificadores da rede.
ii. Posicionamento dos classificadores – Os nós em que os classificadores serão instanciados são definidos. O algoritmo de posicionamento proposto neste trabalho é executado.
iii. Treinamento dos classificadores – Cada um dos classificadores instanciado na rede é treinado de acordo com um conjunto de treinamento previamente obtido. Nesse momento, um provedor NFV disponibiliza os recursos necessários para o classificador ser construído.
iv. Classificação – Sempre que o tráfego passa por um classificador, ele será classificado conforme a técnica instanciada.
v. Amostragem de fluxos – Alguns fluxos são coletados e submetidos ao teste de acurácia. Esse módulo opera em conjunto ao módulo Teste de acurácia e, em alguns casos, eles serão tratados como apenas um módulo.
vi. Teste de acurácia – a amostra de fluxos coletados será classificada por algum método que forneça um *baseline* confiável. Podem ser utilizados um ou mais DPIs, por exemplo. Se a acurácia for inferior a um limiar pré-definido, o módulo de controle é informado para que um novo classificador seja instanciado.

É oportuno destacar que o módulo Amostragem de fluxos é completamente integrado ao Teste de acurácia. Dessa forma, em alguns momentos, quando se faz referência ao módulo Teste de acurácia, o módulo Amostragem de fluxos estará inserido no mesmo contexto. Esses módulos serão discutidos de maneira isolada apenas quando as

atividades de cada módulo forem apresentadas. Em ambos os casos (quando apresentados em conjunto ou não), o objetivo é facilitar o entendimento da arquitetura e sua operação.

Diante da variedade de elementos que compõem a arquitetura apresentada neste trabalho, é possível descrevê-la sob duas diferentes perspectivas. A primeira trata das relações estabelecidas entre os diferentes paradigmas utilizados, enquanto a segunda perspectiva remete à interação operacional dos diferentes módulos da arquitetura proposta.

4.2 Arquitetura distribuída sob a perspectiva dos paradigmas utilizados

A arquitetura proposta é dependente da infraestrutura SDNFV. Na Figura 1.11, é possível observar a forma como SDN, NFV e dispositivos de encaminhamento interagem para permitir que a classificação seja realizada. Nesse caso, os módulos apresentados anteriormente são apenas posicionados visualmente na infraestrutura SDNFV. Não há pretensão de apresentar especificamente, nesse momento, a relação entre os módulos.

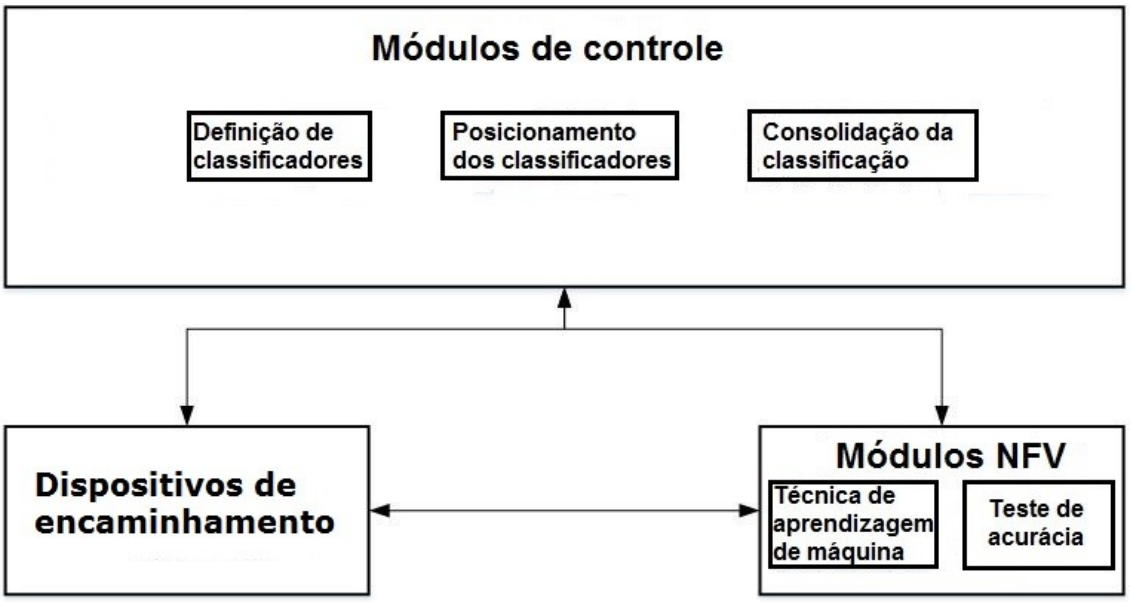


Figura 1.11 - Visão macro da arquitetura proposta.

Adicionalmente, é apresentada também a distribuição lógica dos módulos da arquitetura. O posicionamento dos módulos apresentados remete à arquitetura SDNFV (descrita na Figura 1.6). O plano de controle SDN possibilita que a arquitetura tenha uma visão geral do estado da rede (LIN et al., 2015). Nesse caso, é possível delegar a esse plano a responsabilidade de executar as atividades que necessitem informações sobre o estado dos nós e classificadores gerenciados. O plano de controle SDN abriga os módulos de

controle da arquitetura. Dessa forma, a definição dos classificadores (que também pode ser realizada pelo gerente da rede), o posicionamento dos classificadores e a consolidação da classificação são realizados no módulo de controle. Considerando a arquitetura SDNFV, o Módulo de Controle é responsável por esses submódulos através do controlador SDN e orquestrador NFV. Conforme será descrito a seguir, o módulo de Posicionamento dos classificadores trata da definição dos nós que hospedarão os classificadores, que serão instanciados através da infraestrutura NFV, estando, portanto, relacionado ao orquestrador NFV. O Teste de acurácia, por sua vez, pode operar como uma função virtual e, na Figura 1.6, engloba o módulo Amostragem de fluxos. Se um DPI for utilizado neste teste, por exemplo, ele poderá ser instanciado como uma função virtual junto ao classificador. Entretanto, todas as decisões tomadas a partir do resultado do Teste de Acurácia são realizadas no Módulo de Controle. Adicionalmente, a classificação propriamente dita é realizada através da instanciação de classificadores com recursos virtuais, se baseando na infraestrutura NFV (controlada pelo orquestrador NFV no módulo de controle). Nesse caso, cada classificador será instanciado em um nó específico da rede, selecionado através do algoritmo de posicionamento de classificadores executado no plano de controle.

Em síntese, os módulos que compõem a arquitetura estão dispostos em diferentes partes da infraestrutura da rede de acordo com as informações necessárias ou funcionalidades providas. O plano de controle SDN se comunica com os dispositivos de encaminhamento para que as diretrizes de encaminhamento sejam repassadas, bem como para receber informações sobre o tráfego. Os classificadores virtuais coletam os fluxos a serem classificados dos dispositivos de encaminhamento e exportam seus resultados para o plano de controle, que consolidará todas as classificações obtidas. Nesse caso, o objetivo da classificação é a maximização da acurácia, que é o percentual de fluxos corretamente classificados em relação ao número de fluxos total (essa e outras métricas serão mais bem definidas e discutidas adiante).

4.3 Arquitetura distribuída sob a perspectiva dos módulos operacionais

Considerando apenas as relações entre os módulos operacionais da arquitetura, o fluxo de operação pode ser resumido conforme a Figura 1.12. As atividades realizadas são controladas pelo módulo de controle, especificamente pelo controlador SDN associado ao orquestrador NFV. Cada um dos módulos apresentados anteriormente representa uma

atividade específica na arquitetura (existe um mapeamento direto entre as atividades realizadas e os módulos apresentados).

Algumas dessas atividades precedem a operação do controlador SDN e orquestrador NFV, realizando a preparação da infraestrutura dos classificadores. Em outras palavras, algumas atividades podem ser iniciadas de maneira *offline,* sem relação com o início da classificação propriamente dita ou execução do controle.

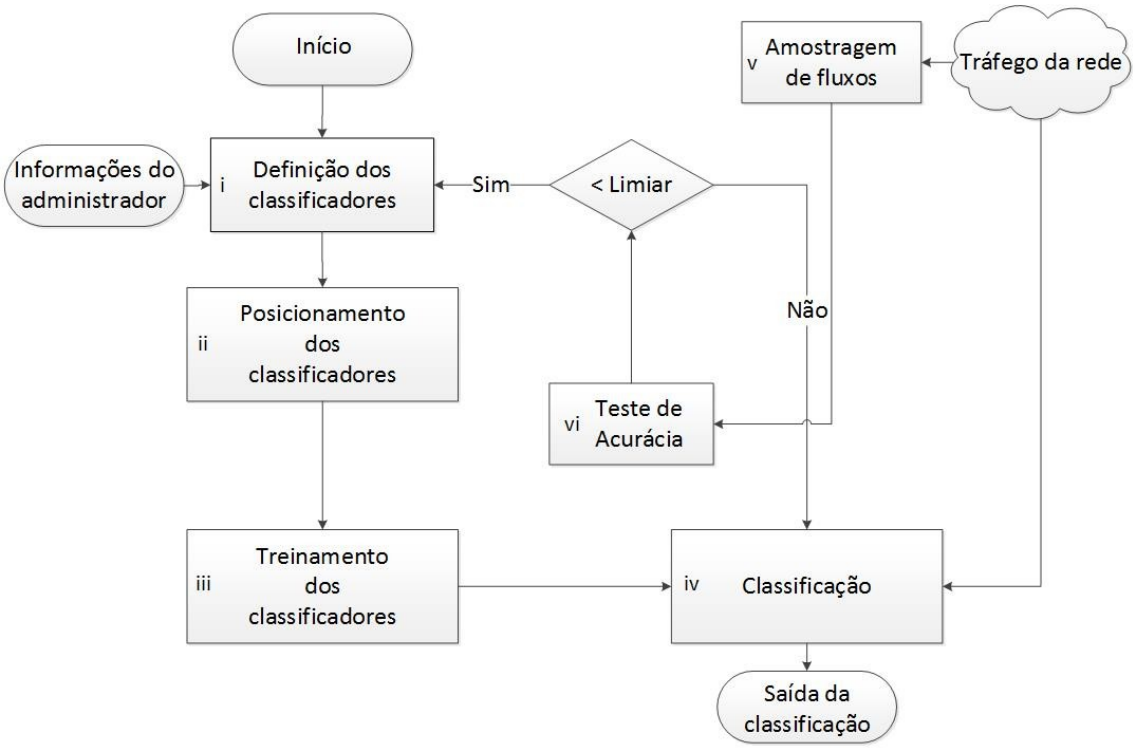


Figura 1.12 – Processo de classificação

Na Figura 1.12, o processo é iniciado com o módulo de definição dos classificadores. Esse módulo recebe como informações do administrador da rede dois aspectos bastante relevantes: (i) quais classificadores serão utilizados e (ii) quantos classificadores serão instanciados. O primeiro ponto suscita o primeiro momento de decisão do administrador da rede. É possível utilizar vários classificadores distintos com um mesmo conjunto de treinamento (ou seja, um conjunto de treinamento único é utilizado para várias técnicas de aprendizagem de máquina) para que suas classificações se complementem e a acurácia do sistema como um todo aumente. Outra possibilidade é utilizar um mesmo classificador instanciado várias vezes em diferentes nós (nesse caso, apenas uma técnica de classificação será utilizada), sendo cada uma dessas instâncias treinadas com um conjunto de treinamento específico. É importante que os classificadores não apresentem uma grande correlação entre si para que seja possível melhorar a acurácia

como um todo. Neste trabalho, as avaliações serão realizadas utilizando um único tipo de classificador, variando os conjuntos de treinamento. Outro aspecto a ser considerado é o número de classificadores. O administrador da rede deve tomar essa decisão de acordo com os recursos que estão disponíveis em sua infraestrutura.

Por exemplo, o gerente da rede pode decidir pela utilização de cinco classificadores na rede sendo duas Redes Bayesianas (PEARL et al., 2000) e três C4.5 ou mesmo cinco classificadores com técnicas distintas (uma para cada classificador). Como dito anteriormente, é importante destacar que, quando dois ou mais classificadores utilizam a mesma técnica, os conjuntos de treinamento utilizados devem ser diferentes. Nesse caso, é possível distribuir classificadores idênticos, otimizando a acurácia e cobrindo uma maior parte do tráfego da rede (IGBE et al., 2016).

Em seguida, uma vez que a quantidade de classificadores foi definida, eles devem ser posicionados na rede através do módulo Posicionamento dos classificadores. Na realidade, cabe ao controlador SDN modificado decidir quais nós deverão hospedar os classificadores e, nesse momento, o orquestrador NFV define a instanciação da função virtual de classificação no nó escolhido. Para decisão de onde posicionar os classificadores, é utilizado um algoritmo específico, que será apresentado na sequência deste trabalho. A operação desse módulo está fortemente relacionada à tolerância a falhas da arquitetura proposta. O fato de se tratar de uma arquitetura distribuída já permite, por si só, que, quando um classificador falhe (por qualquer motivo, como um problema no nó que o hospeda, por exemplo), os outros classificadores permaneçam em operação. No entanto, esse módulo permite que, diante de uma falha que altere a topologia logicamente, o algoritmo de posicionamento seja executado novamente e os pontos de instanciação dos classificadores sejam redefinidos. Além do algoritmo proposto, o gerente da rede pode fornecer como entrada para esse módulo, de acordo com seu conhecimento sobre a rede, os nós nos quais os classificadores serão posicionados. O tempo necessário para instanciação de um classificador na rede tem papel fundamental na viabilidade desse módulo e será estudado nesta tese.

O próximo passo no fluxo de execução da ferramenta é a atividade relacionada ao módulo de treinamento dos classificadores. Esse módulo está intimamente ligado ao primeiro, já que a escolha dos classificadores influencia na forma como eles serão treinados. O treinamento das técnicas de aprendizagem de máquina se dá através de conjuntos de fluxos bem conhecidos. Esses fluxos podem ser oriundos de um DPI (que será assumido como 100% confiável), de uma ferramenta voltada para obtenção de

conjuntos de dados desse tipo, como o gt, ou da intervenção do administrador da rede. Assim como em vários trabalhos já realizados anteriormente, a qualidade do conjunto de treinamento impactará diretamente na acurácia da classificação (OLIVEIRA et al., 2015). No entanto, é esperado que os classificadores demonstrem robustez em relação a erros no conjunto de treinamento, já que diferentes técnicas são influenciadas de diferentes maneiras e diferentes conjuntos de treinamentos apresentem erros distintos.

A Figura 1.13 ilustra o resultado da operação dos três primeiros módulos. Nesse cenário especificamente, os nós escolhidos para hospedar os classificadores foram R2, R3 e R6, sem que o dispositivo de encaminhamento existente precise ser modificado, sendo separados fisicamente do provedor de virtualização. Isto é, o processamento da classificação bem como sua consolidação são realizados pela função de rede virtualizada e pelo módulo de controle, respectivamente.

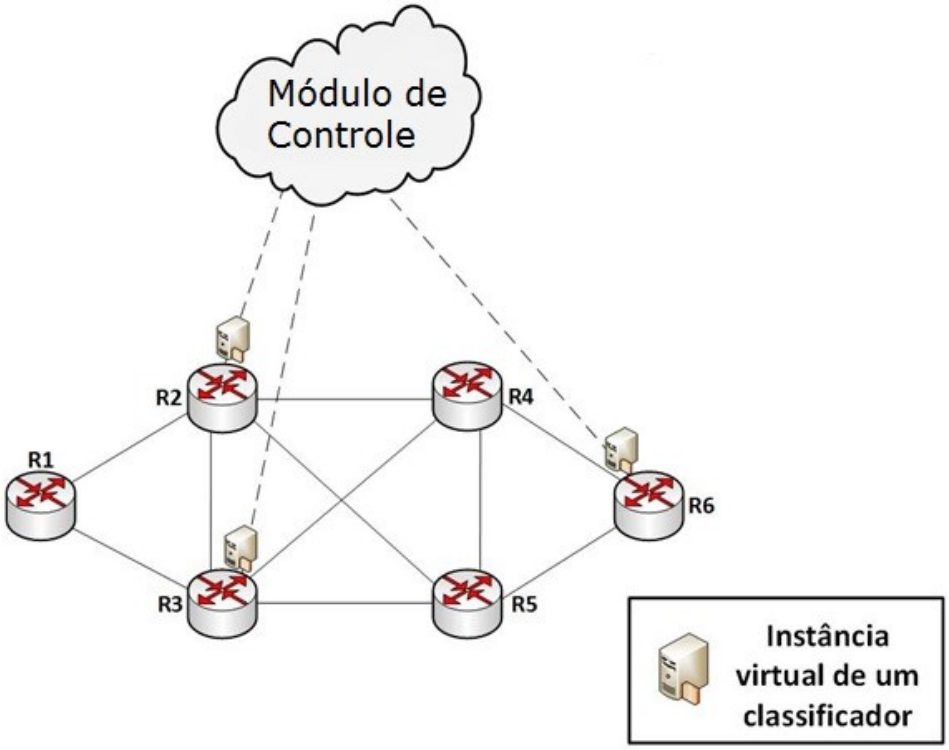


Figura 1.13 – Instâncias virtuais dos classificadores posicionadas logicamente na rede.

Uma vez que os primeiros três módulos foram executados, a arquitetura está apta a realizar a classificação de tráfego da rede (que acontece no módulo de Classificação). Em outras palavras, a três primeiras etapas que foram descritas consistem da preparação do ambiente para que a classificação ocorra de fato. A classificação é o núcleo dessa arquitetura e, portanto, precisa ser detalhada. Considerando que existam dois ou mais

classificadores na execução desse módulo, a forma que eles interagem entre si precisa ser definida. Cada classificador fornece uma única classificação para um fluxo, que pode ser utilizada localmente pelo nó. O módulo de controle da arquitetura consolida as classificações obtidas em uma única. Os métodos de consolidação da classificação são apresentados como importantes contribuições desta pesquisa, uma vez que não existem trabalhos voltados para classificação distribuída na literatura como o proposto nesta tese e, portanto, não existem métodos de colaboração que se apliquem nesse contexto. Nesse caso, foram propostos dois métodos para consolidação da classificação: (i) a classificação baseada em votação (VS) e (ii) a classificação baseada no grau de confiança (GC) – em inglês, *confidence value*. Embora existam diversos estudos para combinação de técnicas genéricas em sistemas distribuídos (HO et al., 1994) (KUMAR et al. 1991), esses dois métodos foram propostos visando apresentar uma baixa complexidade, uma baixa sobrecarga na rede e não tornar os classificadores dependentes uns dos outros. Em outras palavras, um classificador não interfere na operação de outro (classificadores não são alterados por conta de outro classificador), a sobrecarga de troca de mensagens é pequena (apenas a classificação é adicionada à informação do fluxo já repassada ao controlador) e a consolidação no controlador é bastante simples. O primeiro método é um esquema de votação simples. Nele, cada classificador distribuído na rede fornece uma classificação, funcionando como um voto para definição do resultado final e sendo armazenado pelo controlador. A classe mais votada é definida como a classificação do fluxo. Esse método será referenciado neste trabalho como VS (Votação Simples). Já o segundo método se baseia no grau de confiança que cada uma das técnicas de classificação fornece para seu resultado. Os classificadores exportam a classificação do fluxo juntamente com o grau de confiança dessa classificação. Nesse caso, no momento da consolidação da classificação, a técnica que tiver apresentado o maior grau de confiança em sua classificação será escolhida como o resultado final da classificação. Esse esquema de decisão será referenciado no restante deste trabalho como GC (Grau de Confiança). A comunicação entre o plano de controle e a infraestrutura NFV é pequena e será abstraída neste trabalho, assim como a sobrecarga gerada.

Por fim, como dito anteriormente, os dois últimos módulos operam em conjunto: a Amostragem dos fluxos e o Teste de acurácia. Ambos os módulos operam na NFVI como funções virtuais. Enquanto todos os fluxos são classificados pelo módulo de classificação, alguns são escolhidos aleatoriamente (amostragem) para serem enviados para o teste de acurácia. O objetivo do módulo Teste de acurácia é testar a efetividade da classificação à

medida que o tempo passa (na área de aprendizagem de máquina, esse problema é chamado de *concept-drift*). Por exemplo, esse módulo pode ser composto por um DPI (público ou privado, mas que deverá ser assumido como 100% confiável) que proverá um conjunto de fluxos totalmente conhecido. Periodicamente, os fluxos da rede são selecionados amostralmente e classificados pelo DPI virtual e comparados com a respectiva classificação proporcionada pelo módulo de classificação. Adicionalmente, um limiar de acurácia é definido pelo gerente da rede como o índice mínimo aceitável por seu classificador. Nesse cenário, a acurácia obtida pelo módulo de classificação em relação aos fluxos selecionados (utilizando como *ground-truth* a classificação do DPI) será comparada ao limiar pré-definido. Quando o limiar for maior que a acurácia obtida, os primeiros três módulos da arquitetura são executados para que novos classificadores substituam os que estão degradados e sejam capazes de fornecer uma maior acurácia na classificação. Caso a acurácia se mantenha acima do limiar, a classificação continua ocorrendo normalmente. O teste de acurácia deve ser realizado considerando sempre uma janela bem definida para que fluxos antigos não impactem no comportamento atual da classificação. Essa janela pode ser definida por tempo, número de fluxos ou quantidade de *bytes* analisados pelo classificador. Se a janela for maior que o necessário, o teste de acurácia não perceberá mudanças no perfil de tráfego. O módulo de classificação pode ser baseado em qualquer método confiável que forneça um conjunto de fluxos classificados corretamente, como um tráfego sintético ou coletado (OLIVEIRA et al., 2015). Para efeitos de avaliação, simplificando a validação da arquitetura proposta, foi utilizada a ferramenta gt como fornecedora do conjunto de fluxos classificados. Esse conjunto de dados foi assumido como 100% confiável.

O módulo Teste de acurácia implica, em alguns casos, no retreinamento dos classificadores. No entanto, se o treinamento for realizado de imediato, de maneira sequencial, a classificação que está ocorrendo será interrompida e vários fluxos deixarão de ser classificados. A partir desse cenário, um modo de retreinamento que permita a operação contínua do módulo de classificação se faz necessário. A Figura 1.14 sintetiza a operação de retreinamento proposta. Para tornar o estudo do retreinamento mais compreensível, duas abordagens são analisadas em conjunto ao proposto neste trabalho. A abordagem sem retreinamento realiza todo o treinamento no início do período de classificação. Nesse caso, mesmo ocorrendo uma mudança no perfil de tráfego (do perfil 1 para o perfil 2 na Figura 1.14), o classificador continua sendo o mesmo. O retreinamento ideal, por sua vez, prevê a mudança no perfil do tráfego antes mesmo que ela ocorra. Dessa forma, o retreinamento é realizado antes que o tráfego mude, realizando a mudança do classificador sem desperdício

de tempo. Essa abordagem funciona como um limite superior de desempenho para uma avaliação. Por fim, no método com retreinamento, apenas um pequeno conjunto de treinamento é utilizado no início da classificação. Depois disso, o classificador é capaz de se adaptar a mudanças no perfil do tráfego de acordo com a operação do teste de acurácia. Quando um retreinamento é necessário, ele ocorre em paralelo à classificação que já está acontecendo.

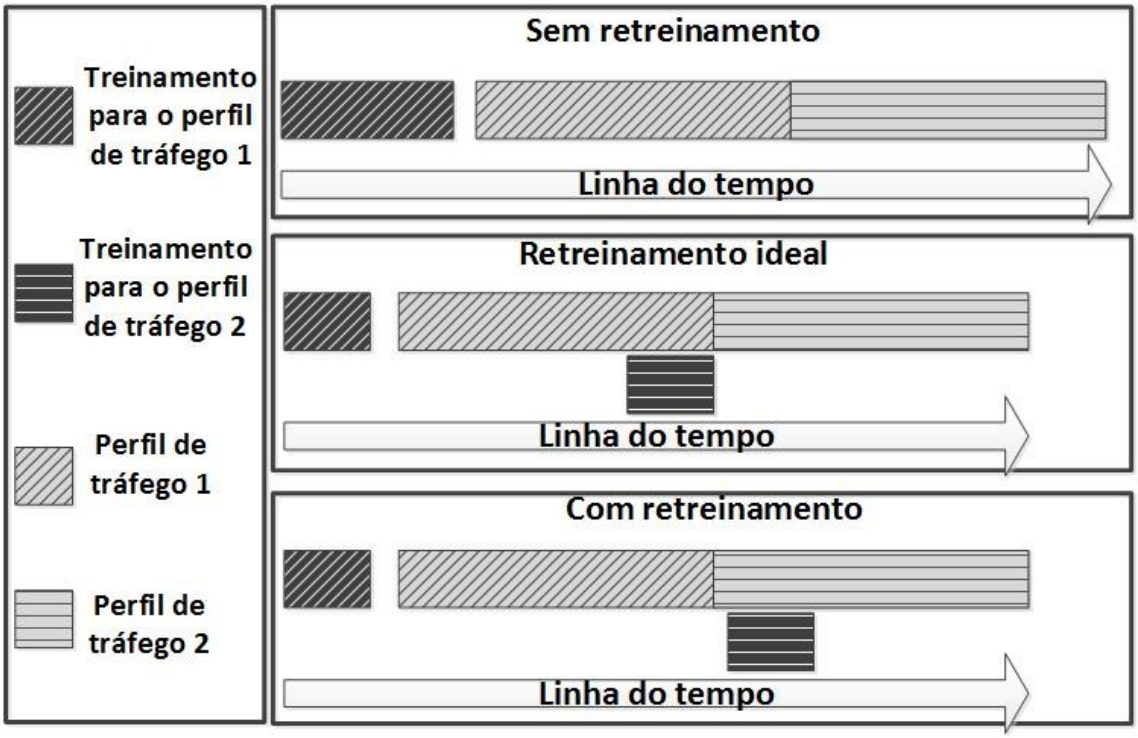


Figura 1.14 – Estrutura dos modos de (re)treinamento.

Essa abordagem é interessante porque permite uma rápida resposta a mudanças no perfil do tráfego, mesmo diante da ocorrência de um evento repentino como um *flash-crowd*. Além disso, essa abordagem permite que o treinamento inicial seja reduzido, já que o classificador pode mudar ao longo do tempo. Realizar o retreinamento em paralelo à classificação que já está ocorrendo permite que a classificação seja contínua (mesmo que a precisão esteja degradada), sem deixar fluxos sem classificação. Além disso, destaca-se que a forma de executar o retreinamento influencia diretamente na acurácia do sistema como um todo ao longo do tempo, bem como no tempo que a classificação ficará comprometida ou não. Por fim, vale salientar que os tempos relacionados ao retreinamento, bem como o tempo de detecção da mudança do perfil de tráfego, são variáveis e dependentes do tamanho de conjunto de treinamento e do limiar do Teste de acurácia.

4.4 Considerações finais

Em resumo, a arquitetura proposta neste trabalho apresenta módulos para efetuar a classificação de tráfego de maneira distribuída combinando os benefícios de diferentes técnicas de aprendizado de máquina, minimizando a possibilidade de falhas com redundância (e provendo a capacidade de recuperação de falhas na rede), permitindo que o teste de acurácia seja realizado, que o retreinamento não interrompa a classificação e, por fim, sendo suportada pela infraestrutura de uma rede SDNFV. Os módulos e atividades apresentados neste capítulo interagem entre si para que os objetivos supracitados sejam alcançados. A seguir, os detalhes relacionados ao algoritmo de posicionamento serão apresentados e as avaliações relacionadas às atividades dos módulos serão discutidas.

É importante destacar que este trabalho não se propõe a realizar a implementação destes módulos em um controlador SDN real, embora seja possível estender um controlador existente para realizar tais operações. Isso ocorre também porque as avaliações, como será descrito no Capítulo 6, serão realizadas isolando as atividades dos módulos (para que cada módulo analisado não sofra interferência dos demais na avaliação). Para realizar as avaliações propostas neste trabalho, o módulo de controle foi implementado para simular seu comportamento. Além disso, os protocolos de comunicação entre as camadas serão abstraídos. A ideia proposta é a validação dos conceitos desta arquitetura através da avaliação de alguns de seus módulos, demonstrando sua viabilidade e os benefícios da realização da classificação de tráfego de maneira distribuída.

5 Algoritmo para posicionamento de classificadores

Este capítulo apresenta, em detalhes, os passos do algoritmo proposto, estabelecendo a relação entre suas etapas e o método de otimização utilizado, bem como a relação com as possíveis soluções. Por fim, o método Subida na Encosta (já apresentado no referencial teórico) é detalhado no contexto do algoritmo proposto.

5.1 Posicionamento para a classificação distribuída

A revisão bibliográfica realizada anteriormente indicou que não existiam trabalhos voltados para o posicionamento de classificadores em uma rede (principalmente que apresentem uma operação interligada), até mesmo pela ausência de abordagens distribuídas. Problemas genéricos apresentados no contexto do problema de locação de recursos consideram clientes estáticos, enquanto o posicionamento de classificadores tem como clientes os fluxos (que mudam de posição) (WOLF, 2009). Dessa forma, teve origem a ideia de posicionar recursos sem considerar a distribuição dos fluxos, apenas a topologia da rede. Em (MA et al., 2014a), uma abordagem para medição de *links* é apresentada que, juntamente com alguns conceitos de teoria dos grafos, forneceu inspiração para a concepção do algoritmo de posicionamento deste trabalho. O algoritmo proposto é denominado PICO (Posicionamento Ideal de Classificadores Otimizado) e tem como objetivo indicar quais nós de uma determinada rede são os melhores hospedeiros para uma função virtual classificar o tráfego de acordo com a topologia.

Com a finalidade de fornecer uma melhor compreensão sobre o funcionamento do PICO, sua estrutura pode ser subdividida em duas partes. A primeira parte consiste da identificação dos nós mais indicados para hospedar um classificador através da decomposição da topologia. Essa identificação ocorre através da valoração da utilidade de cada um dos nós da topologia para hospedar um classificador. De acordo com algumas restrições (descritas adiante), cada nó recebe "indicações" para hospedar um nó. Ao final da primeira parte, os nós com mais indicações são utilizados como entrada para a segunda parte do PICO. A segunda parte trata da aplicação do método Subida na Encosta, tendo como ponto de partida o estado inicial fornecido pela primeira parte do algoritmo. A

escolha do método Subida na Encosta se dá devido à sua simplicidade e efetividade para resolução de problemas com espaço de busca limitado. Além disso, o fato da primeira parte do PICO prover um ponto de partida para o método Subida na Encosta potencializa os benefícios desse método (fornecendo uma direção para a busca). Antes de apresentar o algoritmo propriamente dito (através do seu pseudocódigo), faz-se necessário descrever em detalhes as características do método Subida na Encosta devidamente mapeadas para o contexto de aplicação desta tese.

5.2 Subida na Encosta

Como dito anteriormente, a fim de compreender como o método Subida na Encosta pode ser mapeado para o problema de posicionamento de classificadores discutido neste trabalho, é necessário que algumas características e restrições sejam apresentadas. As subseções a seguir apresentarão detalhadamente cada uma das partes do método Subida da Encosta no contexto deste trabalho.

5.2.1 Estado

Considerando uma topologia qualquer G = (V, A), onde V é o conjunto de nós da topologia e A é o conjunto de *links,* um conjunto de N classificadores e uma função f, representada por $f(v_i)$, que retorna o número de indicações atribuídas ao nó v_i pela primeira parte do algoritmo, um estado *e* pode ser definido como uma N-tupla composta por N nós indicados para hospedar classificadores. Um estado não pode apresentar elementos repetidos. De acordo com a primeira parte do algoritmo, seriam escolhidos os nós que provavelmente possuem os respectivos maiores valores retornados pela função f como solução, já que, quanto mais indicações, mais um nó é indicado para hospedar um classificador. Nesse caso, um estado *e* com N elementos será definido pela primeira parte do PICO como na Equação 1.3. É importante destacar que os elementos da N-tupla na Equação 1.3 foram numerados apenas para diferenciá-los nessa equação. Os elementos do estado são definidos exclusivamente pelo resultado obtido através da função f.

Equação 1.3 – Estado definido pela primeira parte do PICO.

$$e = (v_1, v_2, \dots, v_n), onde\ f(v_1) \geq f(v_2) \geq \cdots \geq f(v_n).$$

Para simplificar o entendimento do que é um estado nesse contexto, alguns valores serão atribuídos, como exemplo, às variáveis envolvidas, para que seja possível uma melhor visualização do conceito:

- V = (v_a, v_b, v_c, v_d, v_e);
- $f(v_a) = 2$, $f(v_b) = 3$, $f(v_c) = 0$, $f(v_d) = 3$ e $f(v_e) = 0$, e;
- N = 3;

Nesse caso, um estado *e* qualquer pode ser definido pela 3-tupla composta por quaisquer 3 dos 5 nós de V (sem que haja repetição de nós). No entanto, o estado *e* que seria escolhido pela primeira parte do PICO como solução nesse cenário é a 3-tupla formada pelos nós com maior valor de indicações na seguinte ordem: (v_b, v_d, v_a).

Diante da definição de estado apresentada, o número de soluções no espaço de busca *E* pode ser definido na Equação 1.4 como:

Equação 1.4 – Número total de soluções no espaço de busca

$$|E| = \binom{|V|}{N}$$

Dessa forma, conclui-se que o espaço de busca *E* está relacionado ao tamanho da topologia, crescendo rapidamente à medida que a topologia apresenta um maior número de nós. Partindo de um nó corrente inicial escolhido de maneira aleatória, o algoritmo Subida na Encosta estaria bastante sujeito a uma convergência prematura (a depender da natureza do problema), uma vez que o espaço de busca tende a apresentar um número considerável de soluções (a solução poderia ser limitada rapidamente a um máximo local) (WEISE, 2009). Para minimizar o impacto desse problema, o estado inicial neste trabalho não será definido aleatoriamente como é preconizado pelo método de Subida na Encosta original. O ponto de partida para a implementação realizada neste trabalho será a N-tupla composta pelos N nós da topologia mais indicados pela primeira parte do algoritmo.

5.2.2 Espaço de busca e a interação do método proposto

Na aplicação do método Subida na Encosta, o espaço de busca não será o mesmo especificado pela Equação 1.4. Isso acontece devido a duas definições que suportam o método proposto: (i) o estado de partida da busca será a N-tupla extraída da primeira parte do algoritmo e (ii) existe confiança nessa N-tupla inicial do algoritmo para a formulação da Subida na Encosta. Quando se faz referência à confiança na solução fornecida pela

primeira parte do algoritmo, trata-se de considerar relevantes as indicações recebidas pelos nós. Nesse caso, quanto mais indicações, mais relevante é o nó para a cobertura. Dessa forma, para a N-tupla inicial, independente do valor de N, o último elemento sempre será o nó que recebeu menos indicações e, portanto, o menos indicado para hospedar um classificador (dentre os nós da N-tupla). No exemplo apresentado na Subseção 5.2.1, o estado inicial obtido através da primeira parte do algoritmo é a tupla ($\mathrm{v_b}$, $\mathrm{v_d}$, $\mathrm{v_a}$). Nesse caso, como é possível observar, $\mathrm{v_a}$ (o último elemento do estado) é o que obteve menos indicações quando comparado aos outros nós da solução ($\mathrm{v_b}$ e $\mathrm{v_d}$).

Baseado nas indicações da primeira parte do algoritmo, o último elemento da N-tupla inicial será o objeto da busca realizada com o método Subida na Encosta. Uma vez que os primeiros elementos foram os mais indicados, considera-se que a dúvida da solução obtida reside no último elemento. Em outras palavras, para a implementação proposta da otimização, os N-1 nós mais relevantes serão mantidos fixos na solução, enquanto o N-ésimo nó da solução (menos indicado) estará sujeito à variação para que os seus vizinhos sejam examinados. Mais uma vez, assumindo-se o cenário de exemplo descrito na Subseção 5.2.1, o estado inicial seria descrito pela 3-tupla (v_b, v_d, v_a). Nesse caso, apenas os vizinhos de v_a seriam examinados para maximizar a função objetivo, enquanto os dois primeiros elementos (N-1) da 3-tupla seriam mantidos fixos na solução.

Diante dessa abordagem, o número de elementos do espaço de busca *E* será obtido de acordo com Equação 1.5. O tamanho do espaço de busca, em geral, será muito inferior ao obtido na Equação 1.4.

Equação 1.5 – Número total de soluções no espaço de busca modificado

$$|E| = |V| - N$$

O tamanho reduzido do espaço de busca permite que a pesquisa seja realizada de maneira exaustiva, assim como discutido em (WEISE, 2009). É importante destacar que, quanto maior o número de nós da topologia e o número de classificadores, maior será a diferença entre o espaço de busca original e o proposto neste trabalho.

Outro aspecto importante em relação à abordagem proposta no PICO é a diferença de relevância entre diferentes nós com um número de indicações próximas. Para manutenção da simplicidade do algoritmo e limitação do espaço de busca, apenas o último elemento da tupla sofrerá variações de acordo com o proposto pelo PICO. No entanto, é possível que o último elemento e o penúltimo elemento da tupla, por exemplo, apresentem

um número similar de indicações, não sendo possível determinar que, de fato, o último elemento seja o "pior" elemento da tupla. Nesse caso, poderia ser sugerida a variação de ambos os elementos. Aumentar o número de elementos que serão afetados pelo método Subida na Encosta implica no aumento do espaço de busca, tornando o algoritmo proposto mais custoso (fazendo com que o espaço de busca cresça desde o espaço da Equação 1.5 até o espaço da Equação 1.4). Por outro lado, como será evidenciado a seguir, quando se considera apenas o último elemento da tupla, o método Subida na Encosta busca pelo melhor nó para complementar a N-tupla, que já é composta por N-1 nós fixos. O algoritmo, nesse caso, se mantém com um custo baixo, já que o espaço de busca é relativamente pequeno. Os resultados obtidos corroboram a decisão de projeto do PICO de variar apenas o último elemento da N-tupla de solução do problema. Entretanto, analisar a variante do PICO considerando menos nós fixos na solução é um interessante estudo futuro.

5.2.3 Função objetivo

Funções objetivo desempenham papéis essenciais para a efetividade de um método heurístico. O principal objetivo do PICO é alcançar a maior cobertura possível dos caminhos com apenas um subconjunto dos nós da topologia analisada. Quanto maior a cobertura de caminhos obtida por uma determinada N-tupla, mais interessante será a solução dentro do espaço de busca.

A função objetivo é, portanto, uma função que recebe como parâmetro o estado atual escolhido como solução e retorna o número de caminhos cobertos pelos classificadores escolhidos. Em síntese, a função objetivo é representada pela Equação 1.6.

Equação 1.6 – Função objetivo do algoritmo Subida na Encosta proposto

$$f(v_1, v_2, \dots, v_n) = \mathrm{C}$$

Nesse caso, C representa o número total de caminhos cobertos pela N-tupla $(v_1, v_2, \dots, v_n)$. Como descrito anteriormente, o método proposto foca na maximização da função objetivo através da "variação" do último nó a ser escolhido (que é o nó com menos indicações recebidas dentre os escolhidos pela primeira parte do algoritmo). Considerando que uma função $c(v_i)$ retorne o conjunto de caminhos cobertos pelo nó v_i, C pode ser desmembrado da seguinte maneira (Equação 1.7) para a N-tupla $(v_1, v_2, \dots, v_n)$:

Equação 1.7 – Desmembramento do número total de caminhos cobertos (valor de C)

$$C = c(v_1) \cup c(v_2) \cup \ldots \cup c(v_n)$$

Alguns aspectos dessa função objetivo são bastante relevantes e devem ser destacados para uma melhor compreensão deste trabalho. O valor de C representa caminhos únicos cobertos pelo estado escolhido e não o somatório dos caminhos cobertos por cada nó. Como representado na Equação 1.7, trata-se da união dos conjuntos de caminhos cobertos por cada um dos nós escolhidos. Em outras palavras, o valor da cobertura obtido por $f(v_1, v_2)$ pode ser diferente de $f(v_1) + f(v_2)$. O valor obtido por $f(v_1, v_2)$ será necessariamente menor ou igual ao valor de $f(v_1) + f(v_2)$. Isso acontece porque é possível que existam caminhos que sejam cobertos por v_1 e também por v_2. Essa característica é essencial para o entendimento sobre a forma de aplicação da Subida na Encosta proposta.

Consequentemente, além do nó v_n ser o menos indicado da solução inicial (e provavelmente cobrir menos caminhos), sua contribuição à cobertura será limitada aos caminhos que não foram cobertos por nenhum dos nós escolhidos anteriormente. Dessa forma, o método proposto através do algoritmo Subida na Encosta pretende encontrar um nó que tenha uma cobertura razoável, mas também tenha a menor interseção possível com os nós já escolhidos. Isso permite que a cobertura total obtida pela função objetivo seja maximizada com caminhos ainda não cobertos.

5.2.4 Função multiobjetivo

Atender uma função multiobjetivo consiste em encontrar uma solução possível que satisfaça algumas restrições e aperfeiçoe uma função composta por duas ou mais funções objetivo (HASHIMOTO, 2004). No cenário específico deste trabalho, é possível que o administrador da rede informe restrições ou forneça outros objetivos a serem alcançados pelo algoritmo de posicionamento.

Nesse caso, a função objetivo apresentada anteriormente poderia ser parte de uma função multiobjetivo. Informações sobre o tráfego (como principal nó de origem e/ou destino em uma rede) podem ser outro objetivo associado, por exemplo. No entanto, funções multiobjetivo tendem a aumentar a complexidade da busca, já que aumentam a complexidade das tomadas de decisão (HASHIMOTO 2004). Além disso, nem sempre é possível otimizar todos os objetivos com uma única solução.

Como o trabalho apresentado visa à otimização do posicionamento através de um método efetivo e pouco custoso, funções multiobjetivo não serão aplicadas ao longo do desenvolvimento deste trabalho, embora sejam uma possibilidade para um administrador que possua objetivos diferentes dos apresentados nesta tese.

5.2.5 Vizinhos

Uma vez que a forma de interação do método de busca e o funcionamento da função objetivo foram definidos, é necessário que seja apresentada a definição de estados vizinhos. Considerando que apenas o último nó da N-tupla será modificado na busca por uma solução otimizada, um estado será vizinho de outro quando os últimos nós de cada um deles forem vizinhos.

Nesta tese, os vizinhos de um nó serão definidos como todos aqueles que tiverem obtido um número igual ou menor de indicações através do algoritmo. A fim de compreender melhor a formulação do método da Subida na Encosta neste cenário, o estado inicial pode ser visualizado como o centro da superfície a ser avaliada, enquanto as demais soluções possíveis são as bordas da superfície, circundando o centro. Nesse caso, todas as soluções possíveis seriam vizinhas da solução inicial. Para fins de implementação da busca, os vizinhos serão examinados desde aquele que foi indicado o maior número de vezes até o menos indicado. Em outras palavras, os vizinhos serão examinados de acordo com o número de indicações que seu último elemento recebeu da primeira parte do PICO. Na sequência, como será descrito na operação do PICO, quando um vizinho é uma solução melhor que o estado atual, ele passa a ser o estado atual, os vizinhos são atualizados e todos os demais vizinhos continuam sendo avaliados.

De maneira geral, o número de vizinhos será igual ao número de elementos no espaço de busca. Isso implica que o Subida na Encosta será exaustivo no cenário do método proposto neste trabalho. A possibilidade de uma solução exaustiva já havia sido apresentada em (WEISE, 2009). Diante da definição de vizinhos e do espaço de busca limitado, todos os vizinhos necessariamente serão examinados. A solução fornecida pelo PICO, dado que os N-1 nós da solução são fixos, é ótima.

O único cenário em que o número de vizinhos de um nó será superior ao espaço de busca surge quando um ou mais nós têm o mesmo número de indicações e mais de um deles são escolhidos como possíveis hospedeiros para os classificadores. Por exemplo, os dois últimos elementos de uma determinada N-tupla apresentam o mesmo número de

indicações (serão representados por v_1, penúltimo elemento, e v_2, último). Embora, v_1 seja vizinho de v_2, a substituição de v_2 por v_1 não cria um estado possível, já que os dois últimos elementos seriam iguais. Consequentemente, o número de soluções no espaço de busca seria menor que o número de vizinhos do nó v_2.

O conceito de vizinhos apresentado se baseia no tamanho reduzido do espaço de busca e na possibilidade de ter uma busca exaustiva. Uma vez que os primeiros N-1 elementos da N-tupla são mantidos fixos, é possível afirmar com certeza que o último elemento da solução será a melhor escolha possível.

5.3 PICO (Posicionamento Ideal de Classificadores Otimizado)

O pseudocódigo do PICO é apresentado no Algoritmo 1.1. O algoritmo em (MA et al., 2014a) forneceu inspiração para a concepção das etapas iniciais do PICO. A principal diferença entre a primeira parte do PICO e o algoritmo proposto em (MA et al., 2014a) é o seu objetivo. O primeiro pretende alcançar um alto grau de cobertura do tráfego da rede enquanto o segundo pretende observar *links* e inferir algumas métricas. Dessa forma, as decisões tomadas por cada um dos algoritmos são diferentes, embora a maneira como a topologia é decomposta seja similar. Por exemplo, em um cenário onde determinado nó A (com vários nós adjacentes) é o único nó adjacente de um nó B, o algoritmo proposto em (MA et al., 2014a) posiciona o monitor em B, para ser capaz de monitorar o *link* A-B. Por outro lado, a primeira parte do PICO indicaria o nó A para hospedar um classificador, a fim de que todo o tráfego com origem em B ou que passe por A seja monitorado.

Para compreender como o PICO pretende alcançar seu objetivo, faz-se necessária a descrição em detalhes das etapas do pseudocódigo. A implementação do algoritmo foi realizada em C/C++ utilizando a biblioteca OGDF[14] para manipulação dos grafos utilizados pelo PICO. Especificamente, foram utilizados métodos de decomposição de grafos em componentes biconectados e triconectados. No primeiro caso, é utilizado um algoritmo básico para descoberta de componentes biconectados (TARJAN, 1971). Para decomposição em componentes triconectados, algumas estruturas podem ser construídas em tempo linear e aplicam a correta versão do algoritmo de Hopcroft e Tarjan (HOPCROFT et al., 1973).

[14] Open Graph Drawing Framework. http://www.ogdf.net/. Acessado em maio de 2017.

Algoritmo 1.1 – Pseudocódigo do algoritmo PICO.

Posicionamento Ideal de Classificadores Otimizado (PICO)

entrada: Topologia da rede G; Número de classificadores H
saída: Tupla com H elementos com os nós mais indicados para hospedarem um classificador
foreach nó N_i da topologia **do**
    if N_i é um vértice de corte de um componente biconectado de G **then**
        indique N_i para hospedar um classificador
    end if
    if N_i tiver apenas um nó adjacente N_j **then**
        indique N_j para hospedar um classificador
    end if
end foreach
decompor a topologia G em componentes triconectados
foreach componente C_i de um componente triconectado **do**
    foreach nó N_i de C_i **do**
        if N_i for vértice de corte de C_i **then**
            indique N_i para hospedar um classificador
        end if
    end foreach
end foreach
solucao_corrente = tupla contendo os H nós mais indicados até o momento
define lista de vizinhos de solucao_corrente
foreach vizinho de solucao_corrente **do**
    if *f*(solucao_corrente) < *f*(vizinho) **then**
        solucao_corrente = vizinho
        atualize lista de vizinhos de solucao_corrente
    end if
end foreach
return solucao_corrente

A classificação de tráfego tem alguns requisitos específicos considerando a topologia. O posicionamento dos classificadores deve obter a máxima cobertura dos caminhos da rede e, consequentemente, dos fluxos (considerando uma distribuição homogênea dos fluxos na topologia). Conforme descrição anterior, a cobertura de um nó referida neste trabalho é o percentual de caminhos da topologia que contém o nó avaliado. A classificação proposta neste trabalho é uma classificação baseada em fluxos (através de uma técnica de

aprendizagem de máquina). Nesse caso, é importante cobrir o máximo de fluxos possíveis. Na realidade, o PICO pode ser aplicado para posicionamento de diferentes funções virtuais que tenham esse objetivo. Em outras palavras, o PICO pode ser descrito como um algoritmo de propósito geral para posicionamento de funções virtuais na rede, que busquem maximizar sua cobertura dos caminhos da rede embora tenha sido concebido considerando o cenário de classificação de tráfego.

Inicialmente, o algoritmo recebe como entrada a topologia da rede que é gerenciada e o número de classificadores que serão instanciados (representado por H). A topologia será descrita como um grafo não direcionado G. É importante destacar que essa topologia remete aos nós gerenciados e capazes de lidar com camadas de virtualização. Já a saída do algoritmo é uma tupla dos H nós pertencentes a G, mais indicados para hospedar um classificador. Nesse caso, o primeiro nó da tupla será o mais indicado para receber um classificador para compor a arquitetura de classificação distribuída. Em outras palavras, o PICO é baseado na quantidade de indicações recebidas pelos nós. Como será descrito a seguir, em determinadas circunstâncias, um nó recebe uma indicação para hospedar um classificador. Aquele nó com mais indicações é apontado como mais propício a receber a função virtual. Para cada nó que compõe a topologia G, duas verificações são realizadas: (i) checa se um nó N_i é um vértice de corte de um componente biconectado de G e (ii) checa se um nó N_i tem apenas um nó adjacente N_j. Em ambos os casos, se a verificação retornar um valor verdadeiro, um nó receberá uma indicação para hospedar um classificador. No primeiro caso, a indicação será destinada ao nó N_i, que representa a borda de um subgrafo de G, sendo, portanto, um nó que possivelmente receberá mais tráfego da rede. Por outro lado, para a segunda verificação, a indicação será destinada ao nó N_j. Nesse segundo caso, isso acontece porque todo o tráfego originado no nó N_i passará necessariamente pelo nó N_j para continuar na rede. Esses passos são apresentados no pseudocódigo da linha 1 à linha 8 do Algoritmo 1.1. Na linha 9, seguindo um procedimento similar ao apresentado em (MA et al., 2014a), o grafo da topologia recebido como entrada é decomposto em componentes triconectados. Em seguida, o algoritmo caminha pelo conjunto de componentes triconectados, inspecionando todos os nós de cada um deles para verificar quais são os vértices de corte em cada caso. Se um nó for identificado como vértice de corte, ele receberá mais uma indicação para hospedar um classificador. Da mesma forma que na etapa realizada nas linhas 2-4, a identificação dos vértices de corte indica, de certa forma, os nós de borda de cada subgrafo analisado. Os passos descritos neste parágrafo são apresentados nas linhas 10 a 16 do pseudocódigo.

A operação do método Subida na Encosta (que representa a segunda parte do PICO) é iniciada quando ocorre a definição da solução corrente inicial (linha 17) a partir da primeira parte do algoritmo e da definição do número de classificadores. Ou seja, o nó corrente definido para a execução do método Subida na Encosta é o estado inicial. Em seguida, o elemento menos indicado para hospedar um classificador do estado inicial é identificado e os possíveis estados vizinhos são elencados (linha 18). Na sequência da operação, a função objetivo é calculada para o nó corrente e comparada com o valor obtido pela função objetivo para um vizinho. Essa operação é repetida para todos os vizinhos identificados (linhas 19-24). Os vizinhos são examinados do que possui mais indicações para o menos indicado. Caso o estado vizinho maximize o valor obtido pela função objetivo, a solução corrente passará a ser igual ao vizinho em questão e os estados vizinhos serão atualizados. Caso contrário, as comparações continuam sem modificação no nó corrente e na lista de vizinhos. Essa avaliação dos estados vizinhos continua até que todos os vizinhos tenham sido avaliados. Por fim, a solução corrente atual será apontada como a solução otimizada fornecida pelo PICO (linha 25).

5.4 A aplicabilidade do PICO

Com base na tupla retornada pelo algoritmo PICO, o controle da rede passa a saber onde os classificadores devem ser instanciados. Isto é, a etapa de posicionamento dos classificadores consiste da execução do algoritmo e da instanciação dos classificadores. O gerente da rede participa de processo de maneira assíncrona, ou seja, o gerente pré-define o número de classificadores N segundo qualquer critério (recursos disponíveis ou características da rede, por exemplo) e os N nós retornados serão os escolhidos como hospedeiros. É interessante perceber que, quanto maior o entendimento do administrador sobre o comportamento do tráfego, melhor será a definição do número de classificadores. O administrador da rede pode, ainda, avaliar e descobrir a melhor relação entre recursos utilizados e desempenho da classificação.

Adicionalmente, o PICO não está necessariamente ligado ao posicionamento estático dos classificadores. Sempre que um determinado evento ocorrer, seja uma queda da acurácia ou uma falha de um nó/*link,* o algoritmo pode ser executado novamente e o posicionamento das funções virtuais refeito. No caso da queda de acurácia, por exemplo, o administrador da rede poderá promover o aumento do número de classificadores caso existam recursos disponíveis. Dessa forma, faz-se necessária uma nova execução do PICO

para posicionar um ou mais novos classificadores. Considerando o caso de falha de um nó ou *link,* a execução do algoritmo é essencial já que a topologia de entrada G será, necessariamente, alterada.

Existe ainda um cenário em que o administrador deve apontar o posicionamento de um novo classificador de acordo com sua percepção do comportamento do tráfego independentemente do resultado do algoritmo. Isso acontece quando ocorre um *flash crowd,* por exemplo. Um evento específico que gere um grande volume de tráfego em determinada parte da topologia não será percebida pelo algoritmo proposto, uma vez que o PICO considera a estrutura da rede e não o comportamento do tráfego. Nesse cenário, em resposta a esse tipo de evento, o administrador pode decidir inserir um novo classificador ou movimentar um classificador antigo de acordo com a região que o evento ocorreu. Adicionalmente, esse tipo de medida pode ser automatizada. A partir do momento que as estatísticas coletadas em um determinado nó indicam que houve um aumento relevante e repentino na sua taxa de fluxos, a construção da arquitetura proposta não impede que um novo classificador seja instanciado no referido nó independentemente do posicionamento indicado pelo algoritmo.

6 Metodologia de avaliação

Este capítulo está subdividido em duas partes: (i) a metodologia geral desta tese e (ii) a metodologia específica para as avaliações realizadas. Na primeira parte, é apresentada uma descrição geral da metodologia a ser seguida para desenvolvimento desta pesquisa e obtenção dos resultados definitivos. Na segunda parte são descritos os passos seguidos para obtenção dos resultados deste trabalho.

6.1 Metodologia geral

Nesta subseção, são apresentados os direcionamentos que serão seguidos na sequência desta pesquisa para obtenção dos resultados, que estão relacionados às questões de pesquisa apresentadas neste trabalho. Especificamente, serão apresentados os possíveis métodos de obtenção do tráfego utilizado (a carga de trabalho utilizada), suas características e quais as principais características das topologias consideradas nos experimentos, além da descrição da infraestrutura NFV utilizada e do cenário considerado para a avaliação.

6.1.1 Geração da carga de trabalho

A geração da carga de trabalho é extremamente importante para a realização de experimentos confiáveis e reproduzíveis. Existem diversas formas de se obter registros de fluxos em *traces* de tráfego da rede. No entanto, cada método se apresenta mais apropriado para um cenário de aplicação específico. Para a classificação baseada em fluxos algumas características específicas são desejadas, como a existência de metadados por fluxo, por exemplo.

Em geral, *traces* podem ser obtidos através da geração sintética de tráfego na rede ou da coleta de tráfego em um ponto específico da rede (BOTTA et al., 2010) (GRINGOLI et al., 2009). Ambos os métodos aparecem em vários contextos diferentes com resultados satisfatórios.

De acordo com (BOTTA et al., 2010), a geração sintética trata da criação de um tráfego que apresenta características que correspondem a um modelo estatístico teórico ou derivado de estudos experimentais. Em síntese, essa geração pode ser dividida em três

categorias de acordo com o nível de granularidade do tráfego gerado: (i) aplicação, (ii) fluxos e (iii) pacotes. Existem ainda os geradores de tráfego multinível, que apresentam um grau de complexidade bastante elevado e não costumam ser utilizados pela comunidade científica. De acordo com a pesquisa bibliográfica realizada neste trabalho, não há na literatura uma ferramenta capaz de fornecer os resumos de fluxos sintéticos com metadados associados fluxo a fluxo.

Vários trabalhos, por sua vez, utilizam a coleta de tráfego para obtenção de *traces* para realização de experimentos controlados (BUJLOW et al. 2012) (CALLADO et al., 2010) (GRINGOLI et al., 2009). Uma vez coletado, o tráfego deve ser identificado para ser útil como conjunto de treinamento. Nesse ponto, é observada uma variação significativa da forma que esse *trace* é identificado. Essa identificação pode ser realizada manualmente. No entanto, devido à quantidade de pacotes que costumam ser coletados, essa operação manual torna-se, em geral, inviável. Em alguns casos, como em (CALLADO et al., 2010), é utilizado um classificador baseado em pacotes para identificar esse tráfego. O problema observado nessa abordagem está na precisão do classificador utilizado. O *trace* identificado, que será o conjunto de treinamento, tem sua eficácia limitada à eficácia do classificador de pacotes. Em outras palavras, dificilmente o conjunto de treinamento obtido será 100% conhecido. Como alternativa para conduzir a coleta do tráfego, é possível destacar as abordagens utilizadas em (BUJLOW et al. 2012) (GRINGOLI et al., 2009). Nesses casos, alguns clientes na rede são monitorados através de um processo na máquina observada e todo o tráfego gerado por eles é identificado. Dessa forma, apenas o tráfego gerado nesses pontos específicos monitorados é contabilizado para a composição do *trace*. Realizar a coleta de acordo com essa abordagem é um método bastante custoso. No entanto, a partir do momento que a coleta é realizada, o *trace* obtido é identificado de maneira 100% confiável.

Nesse contexto, para utilização de um tráfego totalmente confiável, em um primeiro momento, foi escolhida a utilização de um *trace* proveniente da coleta de tráfego identificado através dos clientes que geraram esse tráfego para os experimentos. Utilizar alguma ferramenta para coleta desse tráfego em uma rede de grandes dimensões implicaria na necessidade de controlar um número considerável de clientes nessa rede, o que é uma tarefa bastante complicada. Dessa forma, para atingir os objetivos dos experimentos desta pesquisa, foi utilizado como base para o desenvolvimento dos experimentos deste trabalho o *trace* disponibilizado na página da ferramenta gt. O tráfego foi coletado no roteador de borda da rede do campus da Universidade de Brescia em 3 dias consecutivos de 2009

(30/09, 01/10 e 02/10). As informações detalhadas da coleta realizada nesse período estão disponíveis na página oficial da ferramenta (fornecida anteriormente). O tráfego coletado é apropriado para cumprir o papel de facilitar a validação e avaliação dos módulos propostos neste trabalho, já que apresenta os fluxos com as características desejadas e os respectivos metadados.

Adicionalmente, o tráfego coletado é submetido a um pré-processamento a fim de manter apenas informações relevantes para a classificação proposta nesta tese. Especificamente, o tráfego destinado à interface de *loopback* ou gerado por aplicações desconhecidas foi filtrado. Como o tráfego da rede monitorada é coletado na sua borda e nem todas as estações que geram tráfego na rede executam o *daemon* de monitoramento, nem todo o tráfego coletado na borda apresentará uma classificação associada. Dessa forma, o pré-processamento (filtragem) consiste, basicamente, da retirada dos fluxos não classificados do *trace* final. Os fluxos retirados representam um volume inferior a 1% do total coletado. Desse ponto em diante, inclusive na apresentação dos cenários de experimentos, esse tráfego pré-processado será tratado como *trace* original. É apresentado um resumo geral das características do *trace* utilizado após a filtragem na Tabela 1.4. Nele, são apresentadas as características dos fluxos que foram utilizadas na classificação, bem como as classes possíveis e o número total de fluxos.

Tabela 1.4 – Resumo de características do *trace* original

Características do *trace*	
Características utilizadas dos fluxos	Duração, porta de origem, porta de destino, número de *bytes*, número de pacotes e protocolo de transporte
Classes	HTTP, P2P, Skype, Mail, Transmission, SSH e Outros
Número total de registros de fluxos	13311

Nesse caso, serão consideradas apenas 6 características de cada fluxo: duração, porta de origem, porta de destino, número de *bytes*, número de pacotes e protocolo da camada de transporte. É importante que se destaque que as características dos fluxos que foram consideradas não são as únicas que podem ser utilizadas para realização da classificação pela arquitetura. No entanto, essas características são comumente coletadas, não sendo necessário nenhum processamento adicional. Como diferentes redes podem gerar diferentes padrões de características dos fluxos (como intervalo entre pacotes e tamanho dos pacotes), as características foram escolhidas visando a manutenção de sua simplicidade

(IBRAHIM et al., 2016). Além disso, considerando uma rede SDN, é possível obter uma classificação efetiva utilizando apenas características que são coletadas por padrão pelo OpenFlow, não exigindo nenhuma alteração nas informações já obtidas na SDN (AMARAL et al., 2016). No entanto, uma seleção de características robusta pode possibilitar o aumento da acurácia, como observado em outros trabalhos através da execução de um algoritmo de seleção de características (WILLIAMS et al., 2006).

Os *traces* utilizados neste trabalho serão divididos em duas partes: um conjunto de treinamento e um conjunto de teste. O conjunto de treinamento será utilizado para que cada classificador crie seu modelo de classificação. Em outras palavras, uma instância do classificador receberá o conjunto de treinamento e aprenderá como classificar o tráfego da rede. Já o segundo conjunto funciona como um meio de testar o treinamento realizado anteriormente. Os fluxos presentes no conjunto de teste serão classificados e seu resultado será comparado à classificação real contida no *trace*. A definição de como segmentar os *traces,* para gerar os conjuntos de treinamento, foi realizada com base em experimentos preliminares e testes práticos, com o objetivo de utilizar o menor conjunto de treinamento possível e manter a acurácia em níveis elevados. Embora algumas considerações sobre o resultado sejam realizadas com base no tamanho do conjunto de treinamento, a melhor relação entre conjunto de treinamento e tráfego a ser classificado pode variar em um cenário real.

Cada um dos 13311 fluxos apresentará as informações referentes às suas características para o classificador. Adicionalmente, também são apresentadas 7 classes que definem os fluxos: HTTP, P2P, Skype, Mail, Transmission, SSH e Outros. A classe Transmission, especificamente, remete a todo tipo de fluxos de dados, sem diferenciá-los. Após o pré-processamento realizado no *trace* original, a distribuição de classes obtida é apresentada na Tabela 1.5. Assim como na etapa anterior ao pré-processamento, o tráfego mantém aproximadamente a proporção de fluxos por classe. Assim como na maior parte das redes observadas por diversos trabalhos, a maior parcela do tráfego se refere a tráfego WEB. Os números das classes SSH e Outros foram unidos na Tabela 1.5 na classe Outros, já que se trata de uma parcela muito pequena do volume total do tráfego.

Por outro lado, de acordo com (LI et al., 2013), pesquisadores têm dificuldade de obter os fluxos de um tráfego, seja por acordos de não divulgação ou por outras questões (como a ausência de informações por fluxo, por exemplo). Além disso, considerando a classificação de tráfego baseada em técnicas de aprendizagem de máquina, existe uma crescente dificuldade em obter conjuntos de treinamento que sejam recentes e apresentem

as características desejadas para avaliação. Essa situação também é observada neste trabalho. Nesse contexto, para a concepção de alguns experimentos deste trabalho, será utilizada a geração sintética de fluxos (WETTE et al., 2015), simulando diferentes circunstâncias às quais determinados módulos da arquitetura proposta serão submetidos. Dessa forma, é possível obter um conjunto de treinamento 100% confiável e com características pré-definidas para avaliação do comportamento dos módulos da arquitetura proposta através da geração de tráfego sintético no nível de fluxos. Isso será possível mediante a extensão de uma ferramenta existente para geração de *traces* no nível de fluxos.

Tabela 1.5 – Distribuição das classes.

Classe	Número de Fluxos	% representada no *trace*
HTTP	7384	~ 55,5 %
Transmission	3391	~ 25,5%
Mail	1113	~ 8,4%
P2P	721	~ 5,4 %
Skype	652	~ 4,9%
Outros	50	~ 0,3%

O gerador de tráfego sintético utilizado neste trabalho foi o D-ITG[15] (BOTTA et al., 2010). O gerador foi modificado para que fossem gerados fluxos com a respectiva classificação e não apenas tráfego classificado (informação sobre a classificação foi fornecida fluxo a fluxo). Essa modificação consiste, basicamente, da agregação dos pacotes gerados e, diante da obtenção desses fluxos, da associação dos respectivos metadados aos fluxos. Para a avaliação do retreinamento, foram gerados três *traces* de tráfego sintético. Cada um desses *traces* possui 20000 fluxos e 5 classes em cada um deles. Os *traces* apresentam as mesmas classes: A, B, C, D e E. Essas classes são apenas rótulos que são mantidos para os três *traces* gerados. A ideia é concatenar os três *traces* formando um grande conjunto de tráfego de 60000, com duas mudanças do perfil de tráfego (do primeiro para o segundo *trace* e do segundo para o terceiro). Para isso, foi proposta uma variação na distribuição estatística que define as quatro primeiras classes (A, B, C e D). Em outras palavras, a distribuição estatística que define uma mesma classe é diferente para cada *trace* gerado. Por outro lado, a classe E teve sua distribuição estatística mantida para as três gerações, servindo como uma linha de base para futuras avaliações. Adicionalmente, as classes são muito distintas entre si, facilitando a classificação, que obtém uma alta acurácia (o objetivo desses conjuntos de dados é a avaliação do retreinamento). O *trace* final impõe

[15] http://www.grid.unina.it/software/ITG/. Acessado em Janeiro de 2017.

mudanças de tráfego, permitindo a análise do impacto do *concept-drift*. Por questões de simplicidade, os detalhes desses *traces* não são relevantes para a avaliação. Além disso, considerando um ambiente de mineração de fluxos de dados, o conjunto de treinamento para esse *trace* sintético foi definido como 5% do total de fluxos (3 mil fluxos), embora a utilização desse conjunto de treinamento varie de acordo com o método de retreinamento utilizado.

Na Tabela 1.6, o resumo da forma que os *traces* obtidos são utilizados nos experimentos é apresentado (tanto os *traces* reais, como o sintético). Em síntese, são 4 avaliações que fazem uso desses conjuntos de dados. Para as avaliações da classificação, assume-se que todos os classificadores recebem todos os fluxos, sem diferenças entre os fluxos avaliados.

Tabela 1.6 – Resumo da utilização dos *traces*

Traces	*Descrição*
Trace 1	*Trace* originalmente coletado somado ao conjunto de treinamento, sendo 10% do tráfego destinado ao conjunto de treinamento.
Trace 2	*Trace* 1 modificado. Metade dos fluxos da classe Transmission são postos na porta 80, sendo 10% do tráfego destinado ao conjunto de treinamento.
Trace 3	*Trace* 1 modificado. Algumas características das 5 classes mais frequentes são parcialmente alteradas. O conjunto de treinamento é 10% do total de fluxos.
Trace 4	Três diferentes *traces* sintéticos são concatenados para criar um grande *trace* sintético com 60000 fluxos, 5 classes e conjunto de treinamento de 5%.

O *trace 1* é utilizado na primeira avaliação da acurácia da classificação. Ele consiste de 14790 fluxos, sendo 13311 a coleta original feita pelo gt e 1479 o conjunto de treinamento (escolhidos aleatoriamente do *trace* original). Esse conjunto de treinamento representa 10% do total de fluxos considerados, tendo seu tamanho (pequeno em relação ao total) definido

de maneira similar ao observado em (LI et al., 2007) (WILLIAMS et al., 2006) (ZHANG et al., 2015). O *trace* original é classificado com uma alta acurácia por todos os classificadores utilizados. Isso acontece porque, nesse caso, as portas utilizadas pelas aplicações são bem conhecidas e determinantes para a classificação final. Por exemplo, a classe predominante no conjunto de dados é o tráfego HTTP. Esse tráfego utiliza a porta 80 (tráfego *web*). Dessa forma, apenas a porta utilizada já fornece informações suficientes para a correta classificação. Para que a avaliação da acurácia fosse mais robusta, algumas modificações no conjunto de dados original foram propostas, criando novos *traces*.

O *trace* 2, por sua vez, é a modificação da porta de metade dos fluxos Transmission do *trace* 1 para a porta 80. Essa modificação é a alteração no arquivo de resumo de fluxos do valor atribuído à porta da camada de transporte do fluxo da classe Transmission. Nesse caso, a alteração proposta torna a porta menos decisiva para o resultado da classificação.

O *trace* 3 é o que apresenta mais modificações. Em resumo, algumas características do *trace* 1 são alteradas para gerar um maior grau de dificuldade na classificação:

- 1/3 dos fluxos HTTP passam da porta 80 para a porta 8080;
- 50% dos fluxos P2P tem seu protocolo de transporte alterado de TCP para UDP;
- 1/3 dos fluxos Skype são postos na porta 80, enquanto outro terço tem o número de *bytes* dobrado;
- 1/3 dos fluxos Transmission são postos na porta 80, enquanto outro terço passou a utilizar UDP na camada de transporte.

Por fim, o *trace* 4 é a concatenação de 3 conjuntos de dados sintéticos de 20000 fluxos cada um. Cada um dos sub-*traces* apresenta um perfil de tráfego distinto, possibilitando a análise de mudança no comportamento do tráfego da rede.

6.1.2 Topologias da rede

A instanciação de funções virtuais na rede é muito importante na infraestrutura NFV. Mais especificamente, posicionar os classificadores é uma atividade crítica dos módulos da arquitetura proposta. Esse posicionamento precisa ser efetivo em diferentes topologias para ser uma solução útil. Em outras palavras, o PICO deve ser uma solução genérica para

diferentes topologias. A fim de verificar esse requisito, são utilizadas 3 topologias reais e 1 sintética. As topologias reais estão disponíveis na Internet[16].

O resumo dos detalhes das topologias é apresentado na Tabela 1.7. Na sequência desta tese, cada uma das topologias será referenciada pelo nome do proprietário apresentado na tabela. As topologias reais têm diferentes características, visando uma avaliação que englobe variados cenários. Além disso, objetivando o estudo da escalabilidade, a topologia sintética foi criada com uma quantidade muito superior de nós e *links* quando comparada a uma topologia real. Para isso, uma topologia da China Telecom foi utilizada como base (disponível no mesmo *site* e classificada como *Backbone*). Ela foi replicada 4 vezes e foram adicionados alguns *links* e nós aleatoriamente para fornecer uma topologia conectada.

Observando a topologia sintética, é possível destacar que todas as características são mais significativas que nas demais topologias. Especificamente, o número de caminhos é mais de 70 vezes o número de caminhos da topologia AT&T. Nesse caso, a questão da escalabilidade pode ser tratada para a solução proposta. Por fim, outras características das topologias reais podem ser obtidas na Internet, inclusive o posicionamento geográfico dos nós da topologia.

Tabela 1.7 – Detalhes das topologias

Proprietário	**Classificação**	**Número de nós**	**Número de *links***	**Número de caminhos**	**Data**
IBM	*Backbone; Costumer*	18	24	306	2011
MCI	*Backbone*	19	45	342	2011
AT&T	*Backbone; Costumer; Transit*	25	57	600	2008
Topologia sintética	-	210	345	43890	-

[16] http://www.topology-zoo.org/dataset.html. Acessado em Janeiro de 2017.

6.1.3 Infraestrutura NFV

Para avaliar a desempenho da instanciação de classificadores, são considerados o provedor de virtualização, a instalação da função virtual como serviço e o número de fluxos no conjunto de treinamento.

Primeiro, com base na análise da pesquisa feita em (BONDAN et al., 2016), o provedor de virtualização ClickOS[17] foi escolhido como a melhor solução para nosso cenário, já que apresenta um tempo de inicialização reduzido. Trata-se de uma máquina virtual baseada em Linux, que recebe a instalação do classificador como serviço. O tempo considerado necessário para a inicialização do provedor (VM) é o tempo necessário para responder uma mensagem ICMP. Esse tempo será referenciado como tempo do provedor. Para cada provedor, a técnica de aprendizagem de máquina do classificador já está instalada como um serviço do sistema operacional. Isso permite uma rápida inicialização da classificação. Nesta pesquisa, o WEKA[18] (ferramenta de código aberto para mineração de dados implementada em Java) foi instalado como serviço do sistema operacional na máquina virtual. Por fim, é definido o conjunto de treinamento do classificador para sua construção. O número de registros no conjunto de treinamento pode afetar a duração da construção do classificador. Essa duração é chamada de tempo de construção do classificador. Um classificador estará apto a realizar a classificação depois de decorrido o tempo do provedor e o tempo de construção do classificador.

Uma vez que as métricas a serem aferidas na infraestrutura NFV também considerem o desempenho computacional, é importante a descrição do ambiente computacional. Nesse caso, foi utilizada para instanciar um classificador virtual, uma máquina virtual com 1 processador AMD, 1,5 GB de memória RAM e 20 GB de armazenamento, além de uma placa de rede de 1 Gbps.

6.1.4 Cenário geral da avaliação

Existem diversas variáveis que podem ser analisadas na avaliação dos módulos de uma arquitetura como a apresentada nesta tese. Embora as possíveis avaliações não se limitem ao apresentado neste trabalho, é preciso definir um cenário geral da avaliação para realização de experimentos, já que se trata de uma pesquisa com o escopo bem definido.

[17] http://cnp.neclab.eu/clickos/. Acessado em Maio de 2017.
[18] http://www.cs.waikato.ac.nz/ml/weka. Acessado em Janeiro de 2017.

Dessa forma, o cenário de avaliação deste trabalho será definido da maneira que segue:

- Uma aplicação é capaz de executar os módulos de definição dos classificadores, de posicionamento dos classificadores, de consolidação da classificação e de teste de acurácia. Além disso, ela é suportada pela infraestrutura de uma rede SDNFV, tendo acesso a suas informações;
- Os dispositivos de encaminhamento que compõem a rede são simples e estão associados (cada um) a um provedor de virtualização. Nesse caso, é possível instanciar funções virtuais em todos os nós da rede;
- Os dispositivos de encaminhamento e os provedores de NFV compõem o plano de dados;
- A comunicação entre as camadas é conduzida por um protocolo de comunicação geralmente utilizado em SDN, como o OpenFlow, por exemplo. Algumas extensões seriam necessárias. A comunicação em si não será alvo deste trabalho e, portanto, será abstraída;
- São basicamente três informações que são trocadas entre o módulo de controle e o plano de dados: (i) em quais nós os classificadores devem ser instanciados, (ii) o resultado da classificação e (iii) o resultado do teste de acurácia. No primeiro caso, a informação tem origem no controlador e é enviada para o provedor NFV. Já no segundo e terceiro casos, o sentido da informação é invertido, ou seja, o classificador instanciado (ou o módulo da arquitetura Teste de acurácia) informa ao controlador os resultados de suas classificações;
- O Teste de acurácia opera como função virtual da rede, trocando informações com o classificador através do próprio provedor de virtualização, e;
- Por fim, o controlador SDN tem conhecimento e gerencia toda a topologia utilizada, bem como todos os nós da rede podem ter, em algum momento, funções de rede associadas.

Diante da descrição geral do cenário de aplicação, a interação entre as camadas da arquitetura e sua operação fica mais clara. O tráfego será classificado através da arquitetura com base nas informações que são trocadas entre as camadas. É evidente que apenas a visão macroscópica do cenário de aplicação foi apresentada até este ponto do presente

trabalho. No entanto, para realização de experimentos, é necessário que mais detalhes sejam apresentados, independente do módulo a ser avaliado.

Genericamente, as avaliações podem ser descritas da seguinte maneira:

- Avaliar a acurácia e a precisão da classificação mediante a utilização de diferentes conjuntos de treinamentos para várias instâncias de um mesmo classificador. Nesse caso, o conjunto de treinamento deverá variar para que diferentes classificadores sejam construídos;
- Avaliar o comportamento de um classificador diante de uma grande mudança no perfil do tráfego, avaliando a sensibilidade da classificação à mudança e a efetividade da classificação após a alteração do comportamento do tráfego;
- Avaliar custos computacionais relacionados ao retreinamento do ponto de vista do virtualizador na infraestrutura NFV;
- Avaliar o posicionamento de classificadores de acordo com o algoritmo proposto diante de uma topologia qualquer;
- Avaliar a resiliência do algoritmo de posicionamento dos classificadores quando ocorrem falhas de elementos da topologia, e;
- Avaliar o tempo necessário para que um classificador esteja apto a realizar a classificação, desde a inicialização do provedor até a construção do classificador propriamente dito.

A lista de possíveis experimentos realizados não define todos os experimentos possíveis e, portanto, outros experimentos podem ser propostos como trabalhos futuros. Além disso, maiores detalhes serão fornecidos na descrição de cada um dos experimentos realizados. Nesse caso, os experimentos são detalhados a seguir.

6.2 Metodologia específica para as avaliações

Seis tipos de experimentos foram realizados para atestar os ganhos decorrentes da implementação dos módulos propostos, como a efetividade da classificação dos diferentes esquemas de votação, a eficácia do retreinamento proposto, a medição dos custos computacionais para o retreinamento na infraestrutura NFV, a cobertura de caminhos obtida pelo posicionamento dos classificadores, a resiliência a falhas desse posicionamento

e a análise do tempo de instanciação de um classificador. Nesse caso, os experimentos realizados são o meio para comprovar a efetividade da arquitetura distribuída concebida.

Utilizando o conjunto de fluxos bem conhecidos, a efetividade da classificação distribuída é avaliada. Para essa avaliação, algumas abstrações foram necessárias já que, do ponto de vista da classificação, alguns módulos da arquitetura não precisam ser envolvidos. Nesse caso, o foco reside principalmente no módulo de classificação da arquitetura.

Inicialmente, foram utilizados 5 classificadores (com a mesma técnica de classificação) implementados na ferramenta WEKA. A técnica escolhida foi o J48 (implementação em Java do C4.5), devido à sua simplicidade e eficácia comprovada para problemas de classificação. Em resumo, cada um dos classificadores será instanciado virtualmente em um nó da rede, irá operar localmente no nó e a classificação será consolidada em um plano de controle modificado. Nesse caso, no plano de controle, as classificações individuais serão combinadas para definir a melhor classe para um fluxo. Como descrito anteriormente, são considerados dois esquemas de votação. O primeiro (VS) define a classificação final como a mais frequente entre os classificadores. Isto é, cada classificador representa um voto e a classe mais votada é escolhida. Já o segundo esquema (GC) remete ao grau de confiança das técnicas de aprendizagem de máquina. Ou seja, para cada fluxo, o classificador com maior grau de certeza de sua classificação será escolhido como o melhor classificador.

O método de consolidação dos resultados (esquema de votação) é determinante para a acurácia da classificação da arquitetura. Dessa forma, o resultado da classificação realizada sempre será segmentado de acordo com o esquema de votação utilizado. Além disso, os diferentes conjuntos de dados utilizados, bem como o tamanho do conjunto de treinamento impactarão diretamente no resultado. Portanto, as subseções a seguir descrevem com bastantes detalhes os fatores e níveis que serão considerados na concepção e apresentação das avaliações.

6.2.1 Métricas

Para realizar as avaliações para o cenário de avaliação definido, como descrito anteriormente, foram realizados experimentos para avaliar 4 funcionalidades implementadas: (i) classificação, (ii) retreinamento, (iii) posicionamento de classificadores e (iv) tempo de instanciação.

Para a classificação, são verificadas algumas métricas tradicionais da área de aprendizagem de máquina. Especificamente, são observados a acurácia (mesma definição de *recall*), cobertura de *bytes,* cobertura de pacotes, a precisão e o *f-measure* dos 5 classificadores (todos utilizam a técnica C4.5 e possuem diferentes conjuntos de treinamento). Nesse caso, a acurácia é a relação entre o número de verdadeiros positivos da classificação e o número total de fluxos (o *recall* é também conhecido como sensibilidade). A cobertura de *bytes* é o resultado da divisão entre o número de *bytes* corretamente classificado sobre o número total de *bytes* do *trace*. A cobertura de pacotes é calculada de maneira análoga: número de pacotes corretamente classificados dividido pelo número total de pacotes. A precisão (também conhecida como valor de predição positiva) é a razão entre o número de verdadeiros positivos e o total de ocorrências (verdadeiros positivos + falsos positivos). Por fim, *f-measure* é a combinação das métricas *recall* e precisão, expondo o mapeamento entre elas. Os resultados de cada classificador são comparados e apresentados em alguns gráficos. É importante destacar que, quando as métricas foram observadas por classe (e não para todo o *trace*), apenas as 5 classes mais relevantes foram consideradas. Isso acontece porque, juntas, as duas classes menos frequentes representam aproximadamente 0,3% do *trace*.

Ainda tratando das métricas apresentadas, é importante destacar, além da definição já fornecida, o significado de algumas das métricas. Em especial, a discussão envolvendo *recall* e precisão tem grande relevância. O *recall* remete à completude da técnica, fornecendo a informação do quão completos são os resultados. A precisão, por sua vez, se refere à utilidade da classificação, indicando o quão úteis são os resultados (indicando a razão entre instâncias classificadas corretamente em uma determinada classe sobre o número total de instâncias posicionadas nessa classe).

Como dito anteriormente, a técnica C4.5 está disponível na ferramenta WEKA. Uma vez que o foco dos experimentos não são os ajustes da técnica de classificação, a configuração padrão do algoritmo foi utilizada nos experimentos. Nesse caso, com o resultado de cada classificador é obtido através do WEKA, o módulo de controle teve sua operação simulada, consolidando a classificação. Para isso, foi desenvolvida uma ferramenta para simular o gerenciamento do módulo de controle em C/C++.

Para realizar a avaliação do retreinamento proposto, a duração do treinamento inicial varia quando o retreinamento é utilizado. O retreinamento pretende minimizar os efeitos do *concept-drift*. Nesse caso, são avaliados a acurácia, a precisão e o *f-measure* (calculados de maneira análoga ao realizado nos experimentos da classificação) por classe, comparando

com as abordagens de retreinamento ideal e sem retreinamento. Além disso, o retreinamento implica na coexistência de dois classificadores no mesmo virtualizador. Nesse caso, as métricas são extraídas da máquina virtual a fim de verificar o impacto do retreinamento. Dessa forma, são medidos o consumo de memória RAM, o número de interrupções e o número de trocas de contexto durante o período em que os classificadores coexistem.

Quando o objetivo é avaliar o posicionamento dos classificadores na rede, a métrica utilizada é a cobertura de caminhos (referenciada diversas vezes apenas como cobertura neste trabalho). Para obter essa métrica, todos os possíveis pares de nós da topologia são considerados. O caminho entre dois nós é calculado com o algoritmo Dijkstra (todos os *links* foram assumidos com o mesmo custo). A cobertura de caminhos é, dessa forma, definida como a porcentagem de caminhos que apresentam ao menos um classificador em seus nós. Além disso, a cobertura de caminhos também será observada quando ocorrem falhas na topologia (diferentemente do primeiro caso descrito em que a topologia se apresenta sem falhas). Essas falhas estão relacionadas a nós e são definidas aleatoriamente.

Por fim, o tempo necessário para instanciar um classificador de tráfego como uma função virtual da rede é medido. Esse tempo é composto pelo tempo necessário para inicializar o provedor de virtualização (tempo do provedor) e pelo tempo necessário para o treinamento (tempo de construção do classificador). Esses tempos representam o tempo total necessário para que o classificador virtual esteja operacional. Especificamente, o classificador está apto quando a árvore de decisão está construída.

A Tabela 1.8 resume as métricas que serão utilizadas nos resultados deste trabalho.

Tabela 1.8 – Resumo das métricas.

Métrica	Definição
Acurácia (R*ecall*)	$\frac{(Verdadeiros\ positivos)}{(Verdadeiros\ positivos + Falso\ negativos)}$
Cobertura de *bytes*	Número de *bytes* corretamente classificados dividido pelo número total de *bytes* classificados
Cobertura de pacotes	Número de *pacotes* corretamente classificados dividido pelo total de pacotes classificados
Precisão	$\frac{(Verdadeiros\ positivos)}{(Verdadeiros\ positivos + Falso\ positivos)}$
F-measure	$\frac{2 * (Precisão * Recall)}{(Precisão + Recall)}$

Cobertura de caminhos	Porcentagem de caminhos da topologia que incluem ao menos um classificador
Memória	Quantidade de memória RAM consumida na máquina virtual em dado momento
Interrupção	Número total de interrupções do sistema operacional na máquina virtual em dado momento
Trocas de contexto	Número total de trocas de contexto na máquina virtual em dado momento
Tempo de instanciação	Tempo necessário para que um classificador virtual esteja disponível

6.2.2 Cenários de avaliação

As possíveis configurações do ambiente para realização dos experimentos será identificada como um cenário específico. Nesse caso, cada configuração será denominada "Cenário #", onde # representa o seu número identificador.

Os cenários descritos a seguir para realização dos experimentos são compostos por: características do ambiente (*trace,* topologia ou NFVI), módulo da arquitetura a ser executado, métricas analisadas e fatores avaliados (método de colaboração, número de classificadores ou tamanho do conjunto de treinamento). Esses componentes juntos compõem um cenário de avaliação específico e permitem discussões acerca dos módulos da arquitetura avaliados.

A Tabela 1.9 resume os cenários de avaliação considerados. Nesse caso, cada gráfico apresentado no próximo capítulo está relacionado à apresentação de uma métrica específica em um dos cenários descritos a seguir. É importante destacar que os três primeiros cenários são diferenciados pelos *traces* utilizados, sendo relevantes à medida que os *traces* apresentam características distintas. A descrição detalhada dos *traces* utilizados nos cenários e experimentos listados foi apresentada na subseção 6.1.1.

Por fim, é importante destacar que, para o Cenário 6 e o Cenário 7, as topologias também variam. Nesse caso, isso implica que cada uma das topologias apresentará a análise dos fatores e suas respectivas métricas. Adicionalmente, no Cenário 7, apenas as topologias reais serão consideradas, já que o comportamento do posicionamento diante das falhas depende da lógica existente na composição da topologia. No demais cenários, apenas os fatores variam e as métricas são coletadas.

Tabela 1.9 – Cenários de avaliação

Cenários	Características do ambiente	Módulo da arquitetura	Métricas	Fatores
Cenário 1	*Trace 1*	Módulo de classificação	Acurácia, coberturas, precisão, *f-measure*	Esquema de votação (VS ou GC)
Cenário 2	*Trace 2*	Módulo de classificação	Acurácia, coberturas, precisão, *f-measure*	Esquema de votação (VS ou GC)
Cenário 3	*Trace 3*	Módulo de classificação	Acurácia, coberturas, precisão, *f-measure*	Esquema de votação (VS ou GC)
Cenário 4	*Trace 4*	Módulo de retreinamento	Acurácia, precisão, *f-measure*	Esquema de retreinamento
Cenário 5	NFVI	Módulo de retreinamento	Memória, interrupções e trocas de contexto por segundo	Número de classificadores
Cenário 6	Topologias (IBM, MCI, AT&T e sintética)	Módulo de posicionamento de classificadores	Cobertura de caminhos	Número de classificadores
Cenário 7	Topologias (IBM, MCI e AT&T)	Módulo de posicionamento de classificadores	Cobertura de caminhos	Número de classificadores e de falhas
Cenário 8	NFVI	Módulo de posicionamento de classificadores	Tempo de instanciação	Número de fluxos no conjunto de treinamento

7 Resultados

Os resultados obtidos foram divididos de acordo com o módulo avaliado e o objetivo da avaliação. Nesse caso, serão avaliados a classificação do tráfego (com algumas métricas voltadas para medir a eficácia da colaboração dos classificadores), o retreinamento, o posicionamento dos classificadores e o custo de instanciação de um classificador virtual. Os resultados serão apresentados na mesma sequência em que foram apresentados na metodologia. Ao final deste capítulo, serão apresentadas algumas das ameaças à validação deste trabalho que foram identificadas.

7.1 Avaliação do módulo de Classificação

Antes da discussão detalhada sobre cada um dos experimentos propostos, é interessante especificar para cada uma das avaliações, os respectivos fatores e níveis. Embora essa informação esteja presente no Capítulo 6, apresentar esse detalhamento próximo à discussão dos experimentos facilita o entendimento das conclusões alcançadas. Na Tabela 1.10, os fatores e níveis para a avaliação do módulo de Classificação são apresentados.

Tabela 1.10 – Fatores e níveis para a avaliação do módulo de Classificação

Fatores	Níveis
Classificadores	C1, C2, C3, C4, C5, VS e GC
Conjunto de dados	*Trace 1* (Cenário 1)*, Trace 2* (Cenário 2) e *Trace 3* (Cenário 3)

Como dito anteriormente, para cada experimento, o conjunto de dados foi fixado e algumas métricas são analisadas para cada um dos classificadores. Entende-se como classificadores, nesse caso, tanto os classificadores isolados como os métodos de colaboração propostos. Para cada conjunto de dados, um cenário é definido e as respectivas análises apresentadas.

Na Figura 1.15, a acurácia, a cobertura de *bytes* e a cobertura de pacotes para o Cenário 1 são apresentadas para cada um dos classificadores utilizado na classificação e para os esquemas de colaboração propostos. O *Trace* 1 utilizado nesse caso é o produto da

coleta de tráfego original realizada na Universidade de Brescia. Para simplificar a apresentação dos resultados, as instâncias de classificadores utilizados foram denominadas C1, C2, C3, C4 e C5.

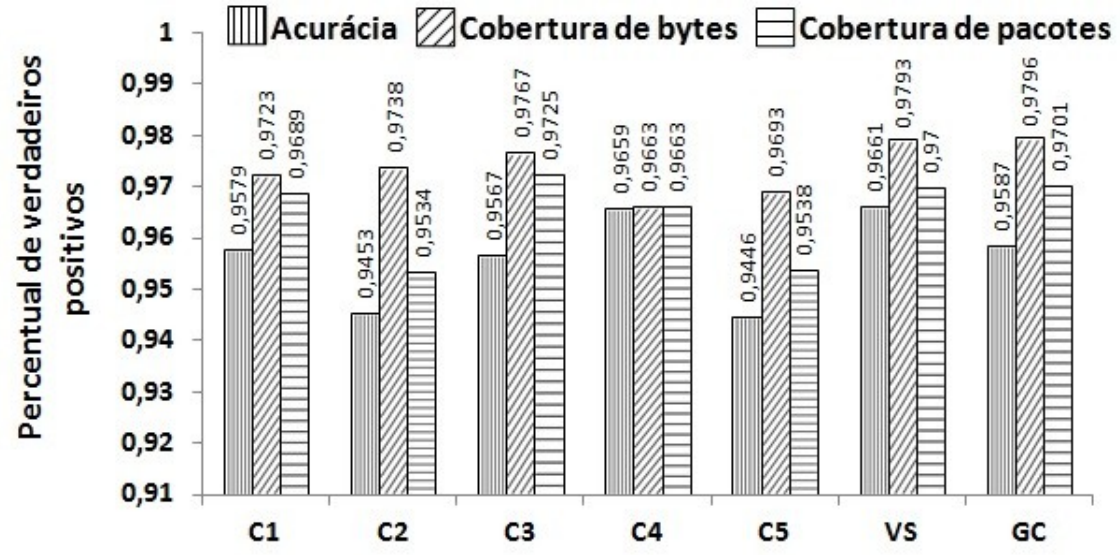


Figura 1.15 – Acurácia, cobertura de *bytes* e cobertura de pacotes para o cenário 1

Como descrito anteriormente, os esquemas propostos serão referenciados como VS (Votação Simples) e GC (Grau de Confiança). É possível observar que a acurácia obtida pelo método VS é superior ao obtido por todas as instâncias de classificadores. O VS é um pouco melhor que o classificador C4, superando os demais classificadores em até 2,08%. O esquema de votação simples (VS) alcança aproximadamente 96,61% enquanto o C2 alcança aproximadamente 94,53%. Um comportamento similar pode ser observado quando são consideradas a cobertura de *bytes* e a cobertura de pacotes, embora a melhor das instâncias de classificação isoladas seja o C3. Percebe-se que o esquema VS supera quase todas as outras abordagens de classificação tanto para a cobertura de *bytes* quanto para a cobertura de pacotes. O VS obtém um percentual de 97,93% de *bytes* classificados corretamente e 97% de pacotes que foram atribuídos para a classe correta após a classificação. Comparativamente, as coberturas de *bytes* e pacotes obtidas pelo C5, por exemplo, foram de 96,93% e 95,38%, respectivamente. O melhor classificador isolado quanto à cobertura de *bytes* e pacotes é o C3 (97,67% e 97,25%, respectivamente). Nesse caso, a abordagem VS supera a acurácia do melhor classificador e a cobertura de *bytes,* enquanto se aproxima da melhor cobertura de pacotes. Por se tratarem de percentuais próximos, o intervalo de confiança foi calculado para os percentuais de acertos obtidos pelos classificadores isolados em relação à classificação de fluxos de *bytes* e de pacotes. Na Tabela 1.11, são apresentados os intervalos de confiança dos percentuais com confiança de 95%. O método de

colaboração VS apresenta um percentual de acurácia e de cobertura de *bytes* estatisticamente superior ao obtido pelos demais classificadores. Quando considerada a cobertura de pacotes, o VS obtém aproximadamente o mesmo índice do limite superior do intervalo de confiança obtido.

Tabela 1.11 – Intervalos de confiança para o Cenário 1

Métrica	**Intervalo**		**Métodos de colaboração**	
	Limite inferior	**Limite superior**	**VS**	**GC**
Acurácia	0,9463	0,9624	0,9661	0,9586
Cobertura de *bytes*	0,9682	0,9752	0,9792	0,9796
Cobertura de pacotes	0,9552	0,9707	0,97	0,9701

Para o Cenário 1, é importante destacar que a acurácia apresentada é muito alta para quase todos os classificadores considerados. Isso ocorre porque existe uma relação direta entre porta e aplicação (como descrito anteriormente), sendo uma característica determinante para o resultado final. Em outras palavras, a porta é determinante para a classificação final. Uma vez que o percentual de verdadeiros positivos é muito alto, os benefícios das abordagens propostas são bastante limitados. Nesse caso, para atestar que as melhorias na precisão são consistentes (não eventos isolados), outras métricas foram coletadas para as principais classes do *trace* (HTTP, P2P, Skype, Mail e Transmission). Apenas as classes mais representativas foram apresentadas para simplificar a análise.

Primeiramente, é apresentada a precisão para cada uma das principais classes na Figura 1.16 para o Cenário 1. Embora a acurácia e as coberturas de *bytes* e pacotes sejam próximas entre os classificadores e os esquemas de votação, é possível observar uma interessante característica na precisão: a precisão obtida pelo método VS alcança uma precisão maior ou igual ao obtido pelos classificadores para todas as classes. Isso significa dizer que ocorreu um ganho consistente da precisão com essa abordagem, sendo apresentada uma classificação mais útil. O VS obteve os seguintes percentuais de acurácia por classe: 99,4% para HTTP, 88,6% para P2P, 83,5% para Skype, 98,9% para Mail e 95,7% para Transmission. Para efeitos de comparação, o classificador C5 obteve 99%, 77,4%, 73,9%, 98,1% e 91,5%, respectivamente. A precisão próxima a 100% para a classe HTTP, por exemplo, é decorrência direta das portas utilizadas por seus fluxos. O GC, por

sua vez, não mantém uma precisão superior para todas as classes quando comparado aos classificadores isolados. Além disso, quando a comparação ocorre entre os esquemas de votação, o VS apresenta um desempenho superior para todos os casos.

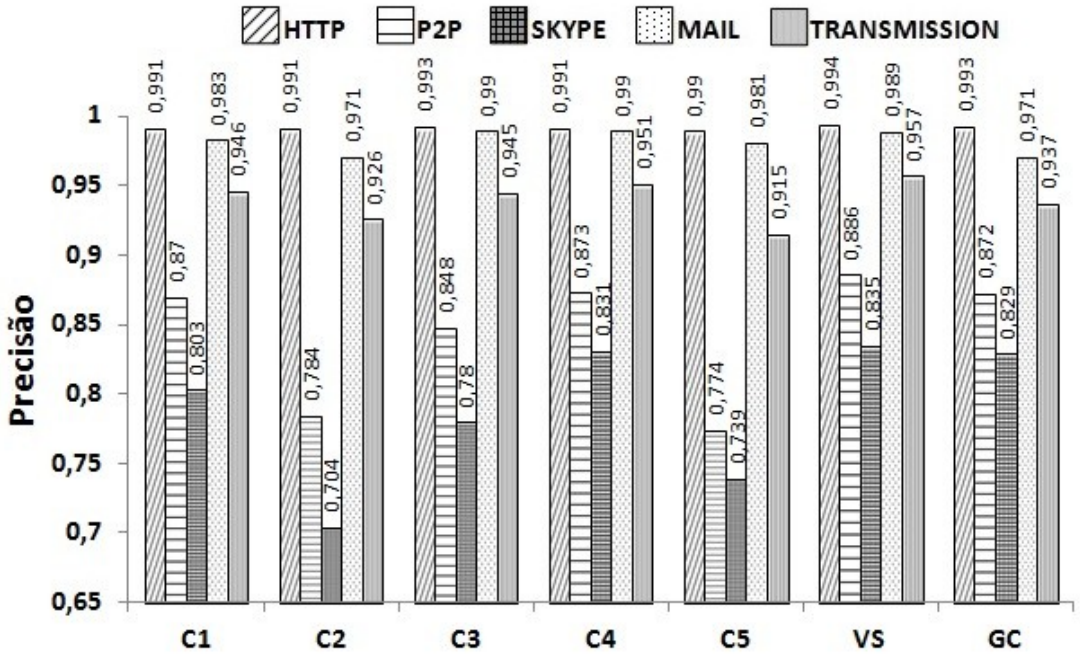


Figura 1.16 – Precisão por classe para o Cenário 1

Para ratificar a consistência dos resultados obtidos, será apresentado o *recall* para as principais classes do conjunto de dados. Na Figura 1.17, observa-se que o comportamento do *recall* é muito similar ao da precisão. Mais uma vez, o VS obtém índices superiores ao obtido pela maior parte dos classificadores avaliados, apresentando um resultado completo. É interessante notar que, se todos os fluxos fossem postos em uma classe majoritária (HTTP, por exemplo), seria possível alcançar uma precisão razoável. No entanto, operando dessa forma, não seria possível obter um alto *recall*. Dessa forma, a combinação da precisão e do *recall* fornece um importante indicador da eficácia do método de classificação. Consequentemente, o *f-measure* por classe para o Cenário 1 é apresentado na Figura 1.18. O método VS obteve um *f-measure* que supera quase todos os outros índices obtidos pelos classificadores por classe. Especificamente, apenas o classificador C4 para a classe Transmission obteve um maior *f-measure,* embora bastante próximo.

Essas métricas apresentadas até este ponto para o Cenário 1 confirmam a hipótese de ganhos consistentes de acertos (acurácia e precisão) nas classificações atribuídas aos fluxos.

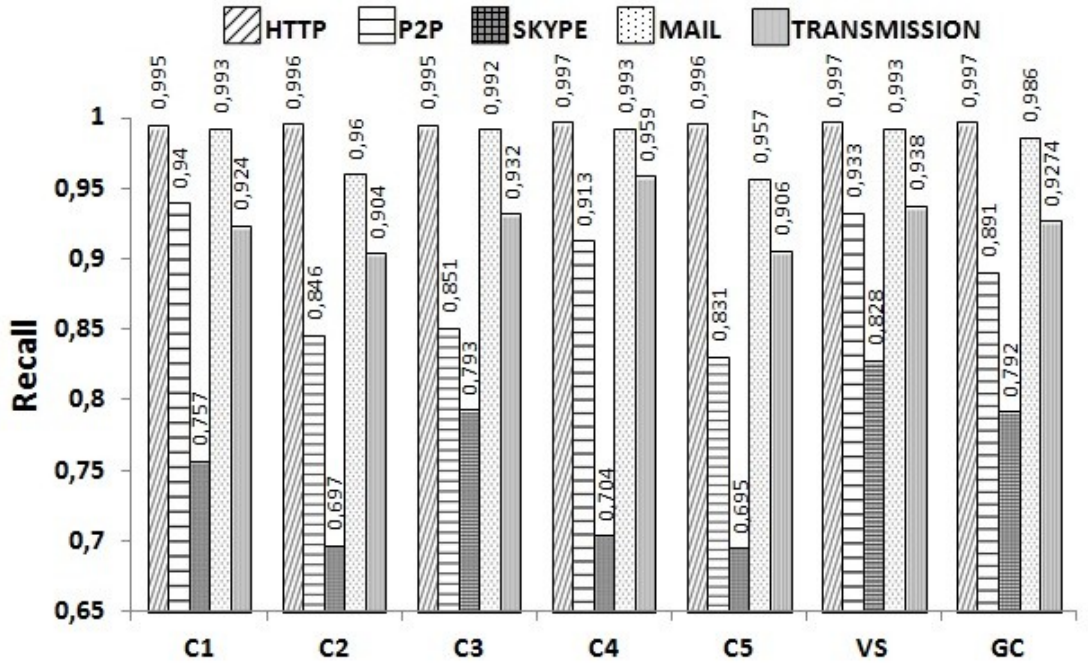


Figura 1.17 – Recall por classe para o Cenário 1.

Por outro lado, como discutido anteriormente, a porta dos fluxos é determinante para a classificação, fazendo que o nível de acertos seja muito alto, limitando os benefícios reais das abordagens propostas.

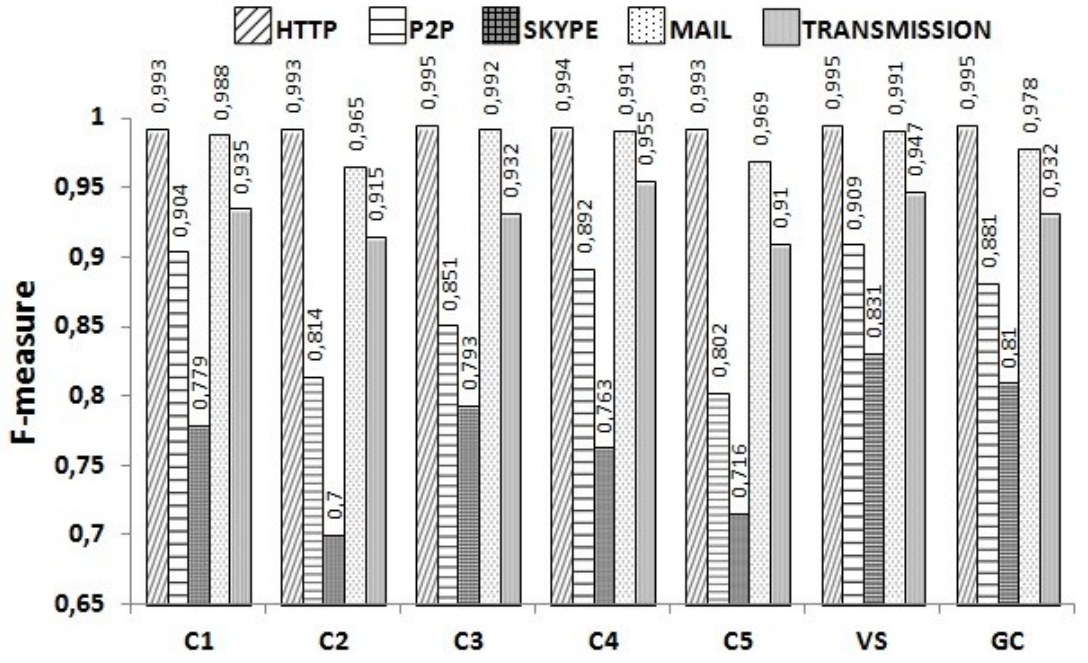


Figura 1.18 – *F-measure* por classe para o Cenário 1

O Cenário 2 foi proposto no Capítulo 6 com o objetivo de verificar o comportamento da classificação diante de um tráfego modificado. Na Figura 1.19, um

gráfico similar ao apresentado para o Cenário 1 é exposto relacionando os resultados da acurácia, da cobertura de *bytes* e da cobertura de pacotes para os classificadores utilizados. O *Trace* 2 utilizado nesse cenário de avaliação apresenta fluxos de classes distintas na mesma porta do protocolo de transporte. Uma vez que a porta tornou-se menos determinante para definição da classe final de um fluxo, os percentuais obtidos foram mais baixos para todas as técnicas avaliadas em comparação ao Cenário 1. Por outro lado, os ganhos obtidos com a utilização dos esquemas propostos foram mais significativos, mesmo as diferenças entre o Cenário 1 e o Cenário 2 não sendo extensas. Nenhum dos classificadores isolados alcançou o desempenho do método VS, que obteve 93,85% de acurácia, superando o melhor classificador tradicional (C3) em aproximadamente 0,73%. A diferença na acurácia para os outros classificadores chega a até 1,5%. Embora a diferença seja relativamente pequena em porcentagem, em um cenário de uma rede real, 1,5% pode representar milhares ou mesmo milhões de fluxos (APPENZELLER et al., 2004) (LOPES et al., 2014). Para uma aplicação de detecção de intrusão, por exemplo, esse percentual pode ser bastante significativo para descoberta de um eventual ataque. A acurácia do método GC é levemente inferior ao observado no método VS, mas supera todos os classificadores tradicionais. As coberturas de *bytes* e pacotes apresentam comportamento similar ao da acurácia, fornecendo ganhos bastante parecidos.

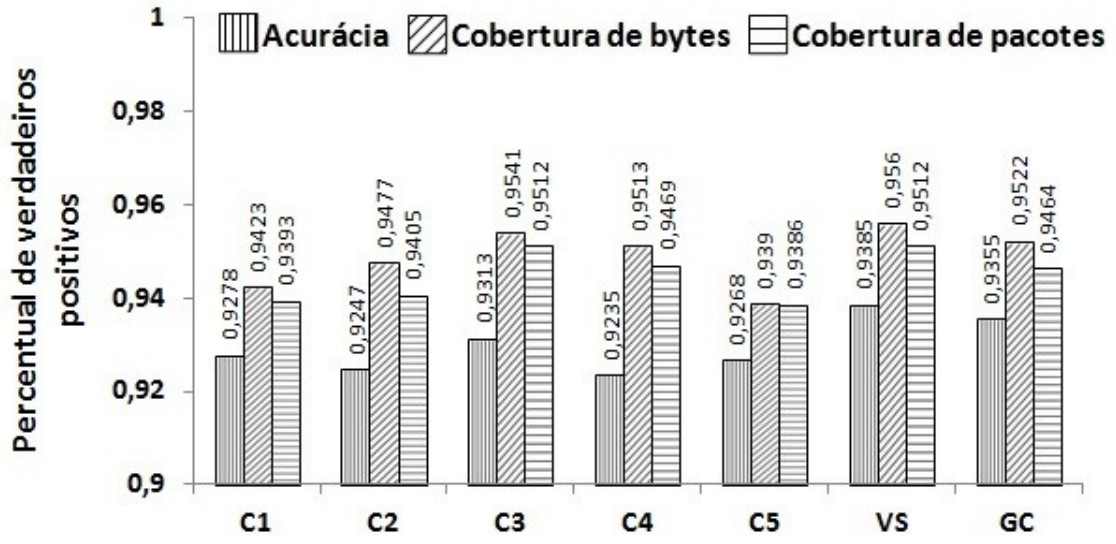


Figura 1.19 – Acurácia, cobertura de *bytes* e cobertura de pacotes para o Cenário 2

Embora os percentuais obtidos sejam relevantes no contexto de classificação de tráfego, os valores são relativamente próximos. Dessa forma, na Tabela 1.12, o intervalo de confiança para cada uma das métricas é apresentado. A confiança é de 95% considerando os classificadores isolados. Nesse cenário, em que a imprecisão dos classificadores isolados

é maior que o observado no primeiro cenário, o método de colaboração VS alcança percentuais estatisticamente superiores aos intervalos obtidos através dos demais classificadores. Em outras palavras, é possível afirmar com 95% de certeza, que o valor médio do percentual obtido por um classificador isolado será menor que o obtido para o método VS para acurácia, cobertura de *bytes* e cobertura de pacotes.

Tabela 1.12 - Intervalos de confiança para o Cenário 2

Métrica	Intervalo		Métodos de colaboração	
	Limite inferior	Limite superior	VS	GC
Acurácia	0,9242	0,9294	0,9385	0,9355
Cobertura de *bytes*	0,9414	0,9523	0,956	0,9522
Cobertura de pacotes	0,9385	0,9481	0,9512	0,9464

Como discutido para o Cenário 1, outras métricas são relevantes para demonstrar a consistência dos benefícios para o Cenário 2. Inicialmente, na Figura 1.20, será apresentada a precisão dos cinco classificadores tradicionais e dos métodos de colaboração propostos para cada uma das 5 classes mais frequentes no conjunto de dados.

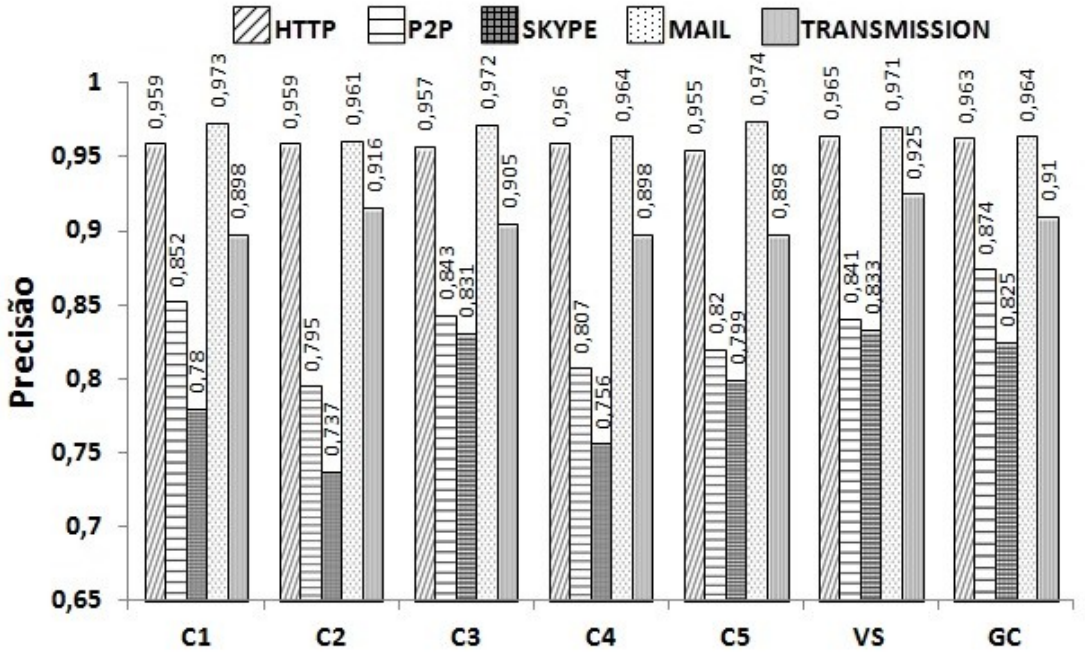


Figura 1.20 – Precisão por classe para o Cenário 2

É evidente que as mudanças de portas propostas no Cenário 2 afetarão diretamente os índices de precisão e *recall* obtidos. Uma vez que a alteração foi realizada na classe Transmission, a análise do desempenho das técnicas propostas será focada nela. O ganho de precisão obtida com o método VS para a classe Transmission foi de até 2,7%. Os classificadores tradicionais (C1, C2, C3, C4 e C5) apresentam um crescimento do percentual de falsos positivos para essa classe. No entanto, quando combinados através do VS, um ganho de precisão considerável é alcançado. De maneira complementar, observa-se o comportamento dos classificadores e métodos do ponto de vista do *recall* na Figura 1.21.

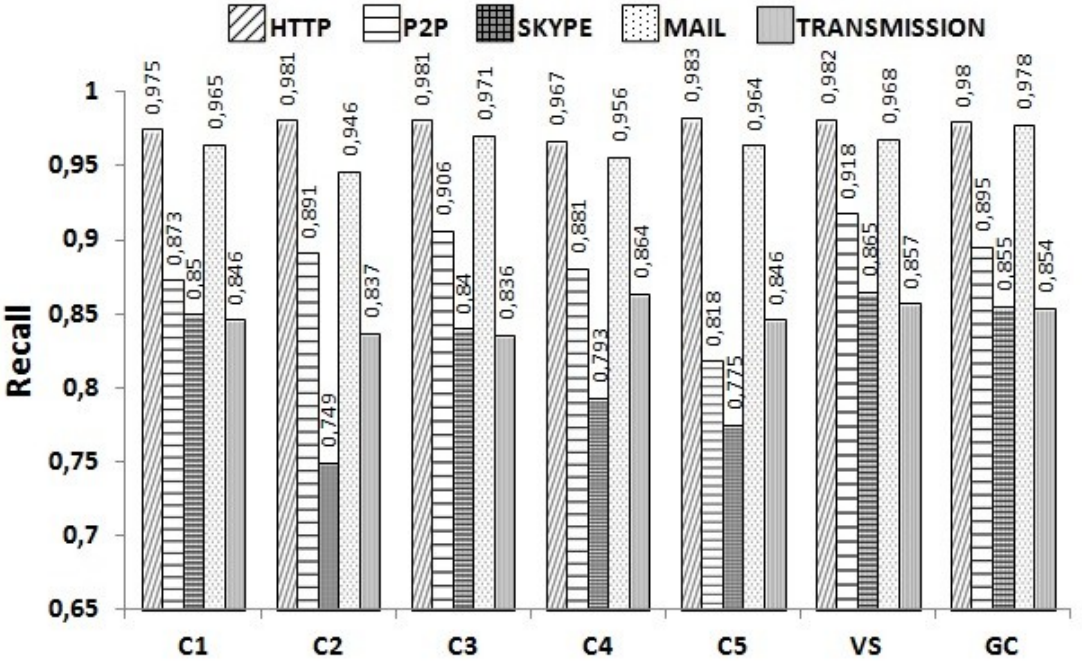


Figura 1.21 – *Recall* por classe para o Cenário 2.

Com o crescimento do número de falsos positivos, existe uma diminuição do *recall* obtido para todos os casos. Entretanto, o *recall* da abordagem VS para a classe Transmission supera quase todos os classificadores isolados. Quando o tráfego da rede apresenta algumas variações, o método proposto de votação é mais robusto, considerando o percentual de verdadeiros positivos. Em outras palavras, quando é mais difícil diferenciar os fluxos do tráfego, a colaboração de diversas instâncias de classificadores surge como uma alternativa interessante para melhorar a acurácia sem incrementar o tamanho do conjunto de treinamento (e, consequentemente, a duração do treinamento). Especificamente no caso do *recall,* o VS chega a alcançar um ganho de 2% comparado a um dos classificadores (C2) para a classe Transmission. Para as demais classes, o comportamento observado é análogo ao que acontece no Cenário 1, com o método VS

obtendo ganhos no percentual de *recall* (chegando a 11,6% de ganho do VS comparado ao C2 para a classe Skype).

A fim de fomentar a discussão sobre a consistência dos resultados obtidos para o Cenário 2, na Figura 1.22, é apresentado o *f-measure* para as classes HTTP, P2P, Skype, Mail e Transmission.

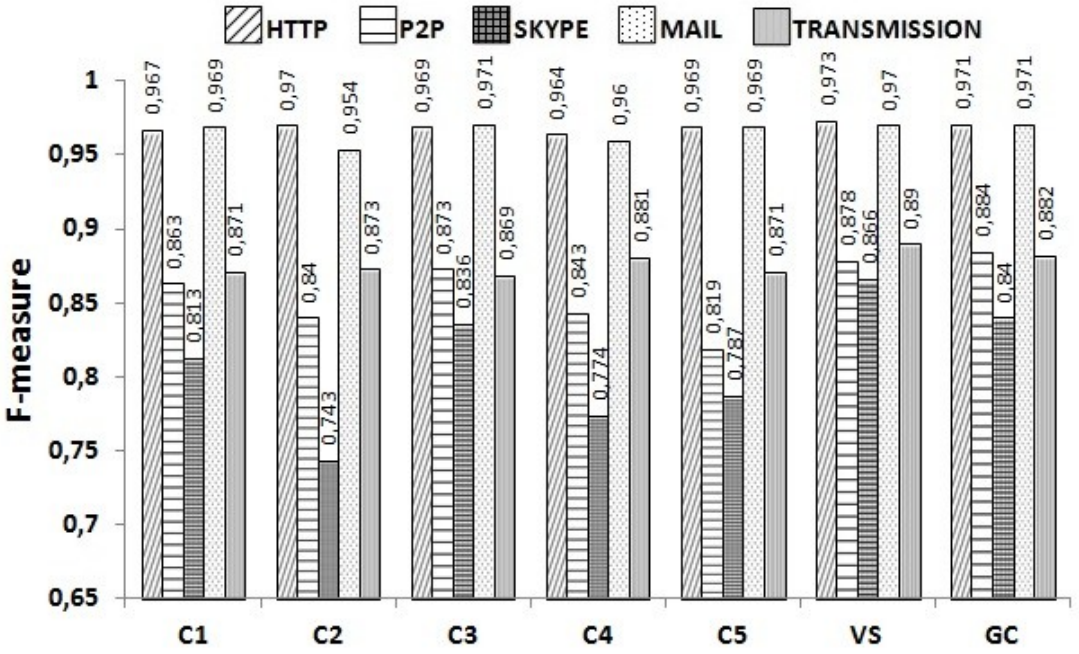


Figura 1.22 – *F-measure* por classe para o Cenário 2

Mais uma vez, a combinação de precisão e *recall* confirma a consistência dos resultados obtidos pelos métodos propostos. O *f-measure* alcançado pelo método VS para a classe Transmission no Cenário 2 forneceu um ganho de até 2,1% quando comparado aos demais classificadores, além de superar todos eles.

Outra avaliação foi realizada, impondo modificações em mais classes e podendo ser observada no Cenário 3. O *Trace* 3 utilizado neste cenário altera as características de alguns fluxos para dificultar a classificação. Nesse caso, são apresentados a acurácia, cobertura de *bytes* e cobertura de pacotes na Figura 1.23. O ganho de acurácia obtido com a abordagem VS foi de até 5%. Além disso, foi observado um ganho de até 2,5% na cobertura de *bytes* e um ganho de 3% na cobertura de pacotes do método VS em comparação aos classificadores isolados. Uma vez que o percentual de verdadeiros positivos é menor para esse cenário em comparação ao apresentado pelos demais, ganhos percentuais de fluxos corretamente classificados são mais significativos. Os ganhos obtidos são relevantes devido ao volume de tráfego que circula nas redes atuais. Por exemplo, algumas redes apresentam

um pico de taxa de tráfego de mais de 2 Tbps (RICHTER et al., 2015). Nesse caso, 5% dos fluxos representa um volume enorme de tráfego da rede.

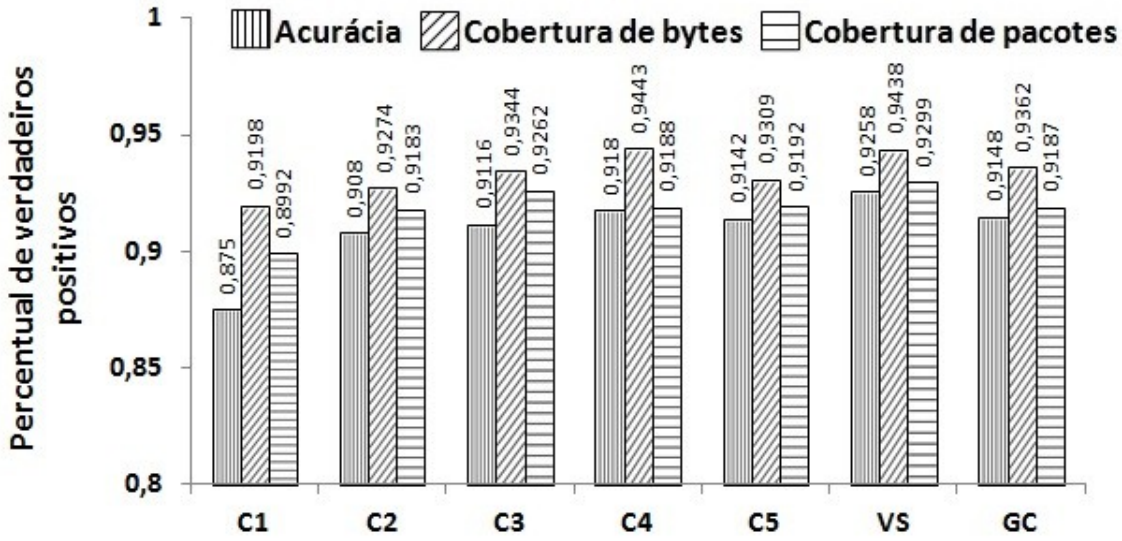


Figura 1.23 – Acurácia, cobertura de bytes e cobertura de pacotes para o Cenário 3.

Do mesmo modo que foi calculado para os cenários anteriores, o intervalo de confiança para o Cenário 3 é apresentado na Tabela 1.13. A confiança mais uma vez é de 95%. Mais uma vez, é possível afirmar com 95% de certeza que o método VS será superior que o percentual médio obtido pelos classificadores tradicionais isolados considerando-se acurácia, cobertura de *bytes* e cobertura de pacotes. Esse comportamento, repetido ao longo da avaliação dos experimentos do módulo de Classificação, confirma a efetividade do método de colaboração VS em comparação aos classificadores isolados.

Tabela 1.13 - Intervalos de confiança para o Cenário 3

Métrica	**Intervalo**		**Métodos de colaboração**	
	Limite inferior	**Limite superior**	**VS**	**GC**
Acurácia	0,8901	0,9206	0,9258	0,9148
Cobertura de *bytes*	0,9234	0,9393	0,9438	0,9462
Cobertura de pacotes	0,9075	0,9252	0,9299	0,9187

Na sequência, para compreender o quão consistentes são os resultados obtidos pelos métodos propostos, serão apresentados a precisão, o *recall* e o *f-measure*. Na Figura 1.24, a precisão é a primeira dessas métricas a ser apresentada. É interessante destacar que,

para as classes P2P, Skype e Transmission, a precisão dos classificadores tradicionais é muito baixa quando comparada aos cenários anteriores. Comparando com o método VS, o ganho de precisão da abordagem proposta é de 10,9% para o P2P, 15,9% para o Skype e 15,8% para o Transmission.

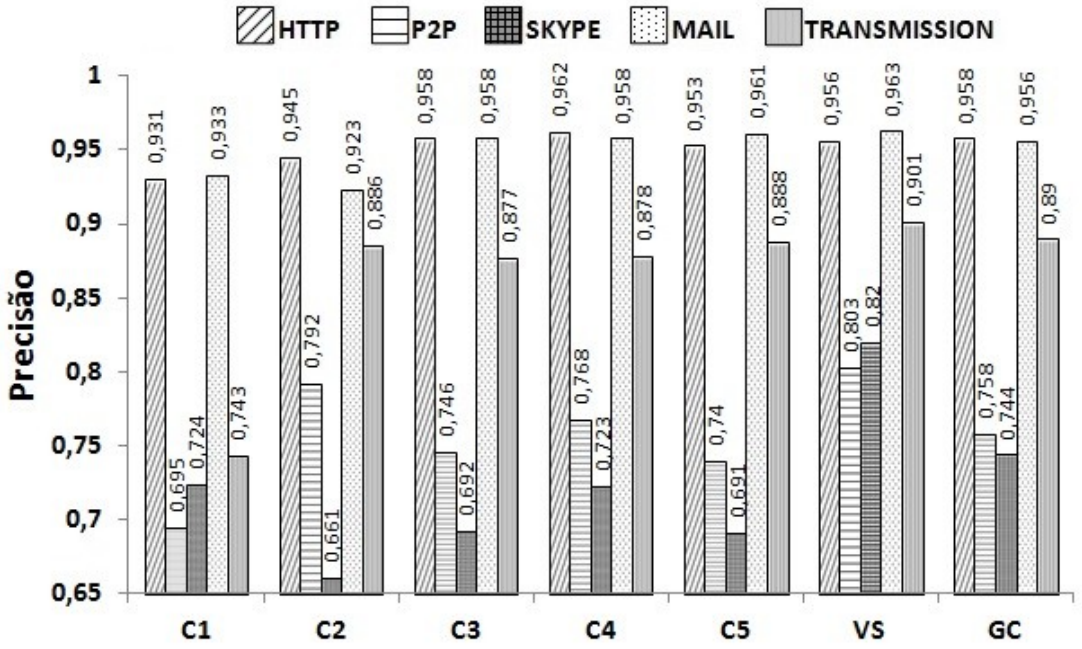


Figura 1.24 – Precisão por classe para o Cenário 3

Adicionalmente, o *recall* é apresentado na Figura 1.25.

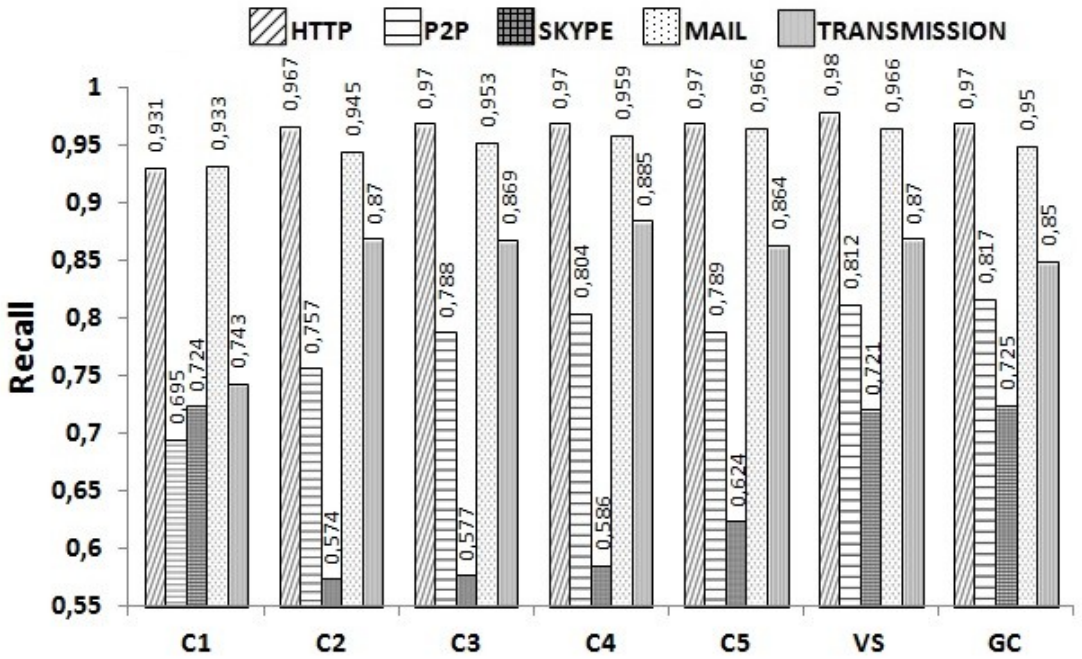


Figura 1.25 - *Recall* por classe para o Cenário 3.

Conforme observado para a precisão do Cenário 3, o *recall* também é bastante melhorado para as classes P2P, Skype e Transmission utilizando o VS. Tanto para a precisão quanto para o *recall,* o método GC apresenta percentuais próximos ao VS (embora inferiores). Apesar do decréscimo no percentual de todas as métricas apresentadas até este ponto para o Cenário 3, tanto precisão quanto *recall* alcançam percentuais superiores a 90% para algumas das classes observadas.

Por fim, o *f-measure* é exposto na Figura 1.26 para o Cenário 3. Já que precisão e *recall* apresentam comportamentos similares, o mesmo padrão pode ser visto na métrica *f-measure*. Por apresentarem percentuais relativamente próximos, a comparação entre os métodos VS e GC é dificultada nos gráficos apresentados anteriormente.

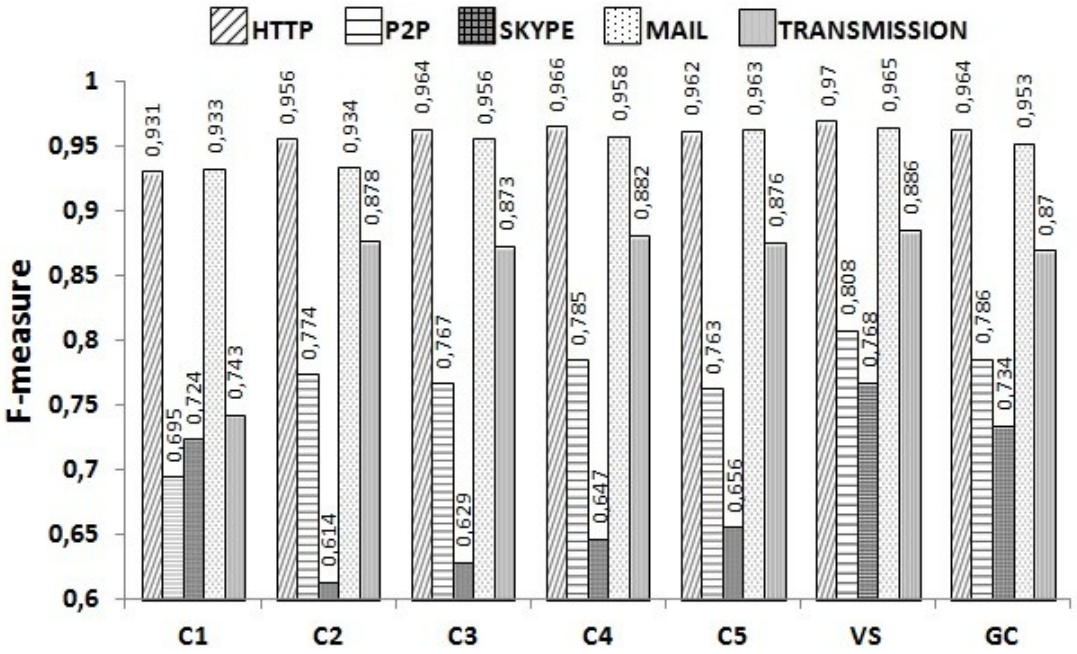


Figura 1.26 – *F-measure* por classe para o Cenário 3.

Essa comparação é apresentada na Tabela 1.14. Em destaque na tabela estão os campos que apresentam o melhor percentual para acurácia, cobertura de *bytes* e cobertura de pacotes. A tabela evidencia que, para a maior parte dos casos, o VS atinge percentuais maiores para representar os fluxos, os *bytes* e os pacotes corretamente classificados. A diferença entre as duas abordagens cresce da mesma forma que ocorreu nas avaliações apresentadas anteriormente. Ou seja, quando existe um maior grau de dificuldade na classificação dos fluxos, o método VS apresenta resultados mais consistentes em comparação aos classificadores isolados e também em comparação ao método GC. Além disso, o VS alcançou percentuais estatisticamente significativos para todos os cenários

avaliados. Por fim, após a realização dessas avaliações, bem como das discussões associadas, é possível concluir a efetividade dos métodos de colaboração propostos, em especial o VS. Mais importante que os ganhos percentuais observados é o comportamento da classificação diante da colaboração entre classificadores menos efetivos. A abordagem proposta provê ao administrador da rede uma classificação mais efetiva e confiável.

Tabela 1.14 – Comparação entre os cenários 1, 2 e 3

	Cenário 1		**Cenário 2**		**Cenário 3**	
	VS	GC	VS	GC	VS	GC
Acurácia	96,61%	95,87%	93,85%	93,55%	92,58%	91,48%
Cobertura de *bytes*	97,93%	97,96%	95,6%	95,22%	94,38%	93,62%
Cobertura de pacotes	97%	97,1%	95,12%	94,64%	92,99%	91,87%

7.2 Avaliação do retreinamento

Alguns fatores e níveis são apresentados na Tabela 1.15 a fim de facilitar a compreensão dos experimentos realizados nesta subseção. Nesse caso, além dos fatores e níveis, também foi especificado o cenário que identifica o fator e os níveis considerados na avaliação e as respectivas métricas.

Tabela 1.15 – Fatores e níveis para a avaliação do retreinamento

Cenário	Fatores	Níveis	Métricas
Cenário 4	Método de retreinamento	Retreinamento ideal, sem retreinamento e com retreinamento.	Acurácia, precisão, *recall* e *f-measure*.
Cenário 5	Quantidade de classificadores no mesmo virtualizador	1 e 2.	Memória RAM, interrupções e trocas de contexto por segundo.

Em geral, a maior limitação das técnicas de aprendizado de máquina é a fase de treinamento, seja pela necessidade de um conjunto de treinamento confiável ou pelo tempo de duração dessa fase. Neste trabalho, o problema do tempo de duração da etapa de treinamento das técnicas foi abordado através do módulo de retreinamento da arquitetura proposta, que pretende alcançar a redução do tempo inicial de treinamento, bem como a minimização do impacto de posteriores retreinamentos.

A primeira consideração a ser feita está relacionada à área de mineração de fluxos de dados: o conjunto de treinamento deve ser o menor possível devido a restrições de tempo (LOPES et al., 2014). Essa restrição existe, também, pelo fato do aumento do tamanho do conjunto de treinamento geralmente implicar no crescimento linear da complexidade da árvore do classificador (OATES et al, 1997) e no aumento do tempo de instanciação do classificador como será apresentado a seguir. Do ponto de vista da velocidade e complexidade da árvore, o retreinamento permite que seja utilizado um pequeno conjunto de treinamento no início da classificação, o que possibilita ganhos no desempenho do classificador. Além disso, o retreinamento realizado ocorre em paralelo ao processo de classificação, permitindo que não seja "desperdiçado" nenhum período de tempo.

Além disso, o retreinamento é proposto nesta tese com base na premissa de que é possível adaptar a classificação a mudanças no perfil do tráfego. Três métodos de treinamento são considerados no Cenário 4 (conforme Tabela 1.15). Nesse caso, cada um dos métodos de retreinamento faz uso do conjunto de treinamento de uma maneira diferente. O método denominado "Sem retreinamento" realiza o treinamento apenas no início do processo de classificação. Nenhum mecanismo adaptativo é utilizado para lidar com as mudanças no perfil do tráfego. O "Com retreinamento" representa o método proposto neste trabalho. Nesse caso, quando ocorre uma degradação da acurácia (que significa uma mudança no perfil do tráfego), um novo (re)treinamento é realizado em paralelo à classificação que já ocorre. O último método considerado é o "Retreinamento ideal". A diferença entre ele e o método "Com retreinamento" é que ele sabe que a acurácia será degradada antes que isso ocorra, treinando o classificador para o perfil de tráfego subsequente. É evidente que esse último método é hipotético, mas extremamente útil para a análise a ser realizada. O "Retreinamento ideal" funciona como um limite superior da acurácia para o método proposto nesta pesquisa.

Para validar a premissa da possibilidade de adaptação a mudanças no perfil de tráfego, o Cenário 4 é avaliado. A Figura 1.27 apresenta a acurácia para o Cenário 4,

considerando os métodos "Sem retreinamento", "Com retreinamento" e "Retreinamento ideal". Nesta avaliação, o conjunto de treinamento utilizado foi de 5% do número total de fluxos.

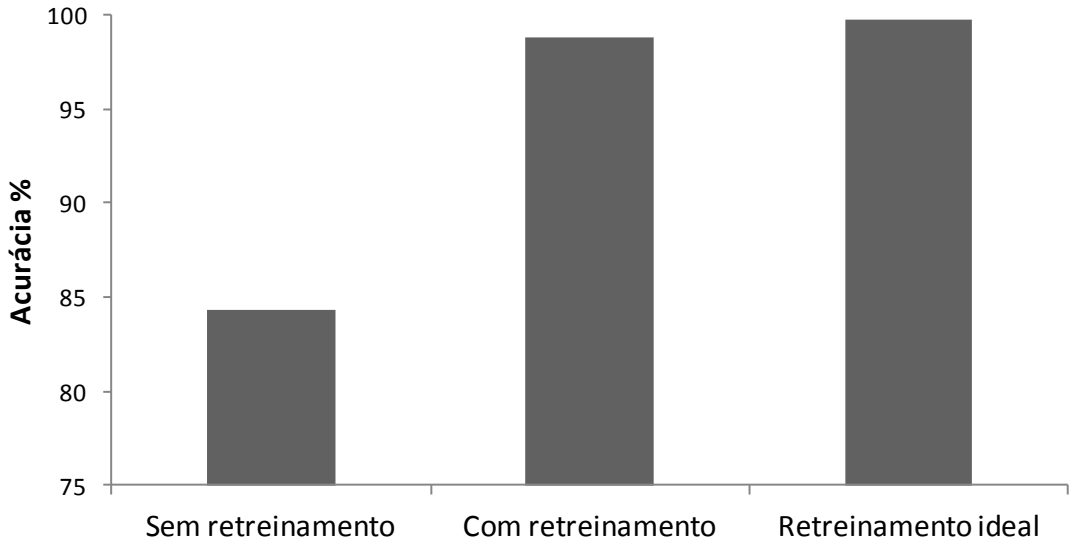


Figura 1.27 – Acurácia para o Cenário 4

A ideia é obter um grande percentual de acurácia mesmo utilizando um pequeno conjunto de treinamento e com variações no perfil de tráfego. Uma vez que o tráfego do Cenário 4 apresenta duas grandes mudanças no seu comportamento (ou seja, 3 perfis de tráfego), o conjunto de treinamento é dividido em 3 subconjuntos, cada um para um perfil específico. Os resultados na Figura 1.27 demonstram uma melhoria na acurácia comparando os métodos com e sem retreinamento. O ganho percentual é de aproximadamente 14,5% de fluxos corretamente classificados, o que é significativo para qualquer cenário de classificação de tráfego de redes, em especial o gerenciamento de redes. Além disso, é importante destacar que o método "Com retreinamento" se aproxima bastante do "Retreinamento ideal", sendo superado em menos de 1%.

Para ampliar a discussão, serão apresentados a precisão, o *recall* e o *f-measure* para cada uma das classes do Cenário 4. Na Figura 1.28, a precisão pode ser observada. A precisão alcançada com o retreinamento é próxima do obtida pelo método ideal. As classes B, C e D apresentam consideráveis variações em sua distribuição estatística ao longo do tempo no *trace* gerado, aumentando o número de falsos positivos. Consequentemente, a precisão sem retreinamento sofre degradação considerável. Por outro lado, a classe E,

utilizada como *baseline* (já que não apresenta variações de perfil ao longo do tempo), tem uma precisão muito similar para os três métodos.

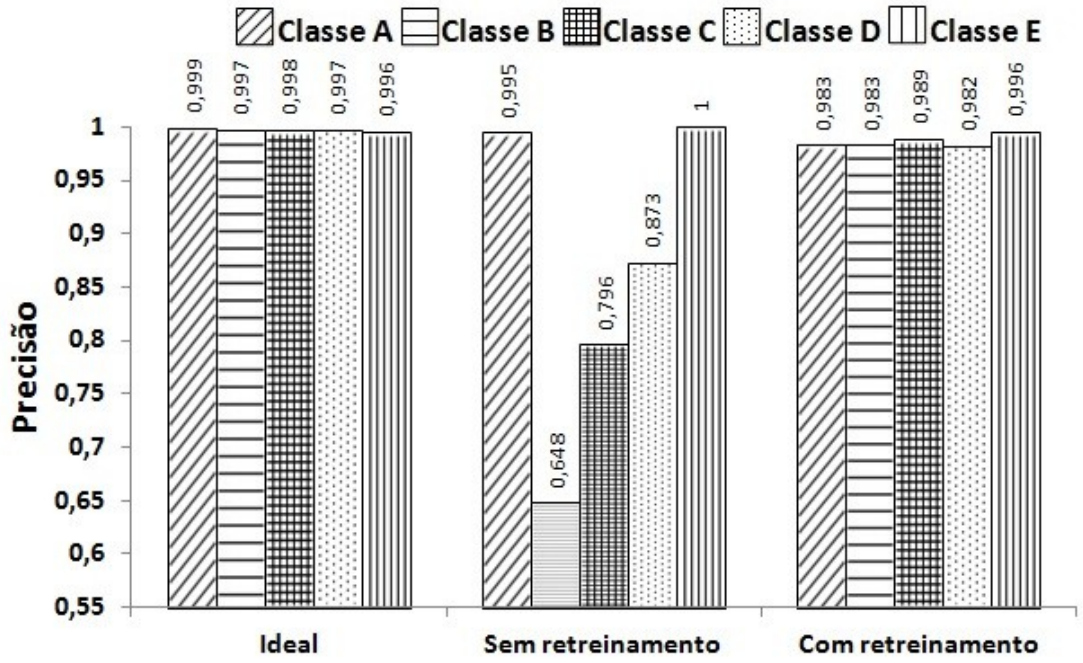


Figura 1.28 – Precisão dos métodos de retreinamento por classe para o Cenário 4.

Na Figura 1.29, o *recall* para cada um dos métodos de (re)treinamento é apresentado para cada classe.

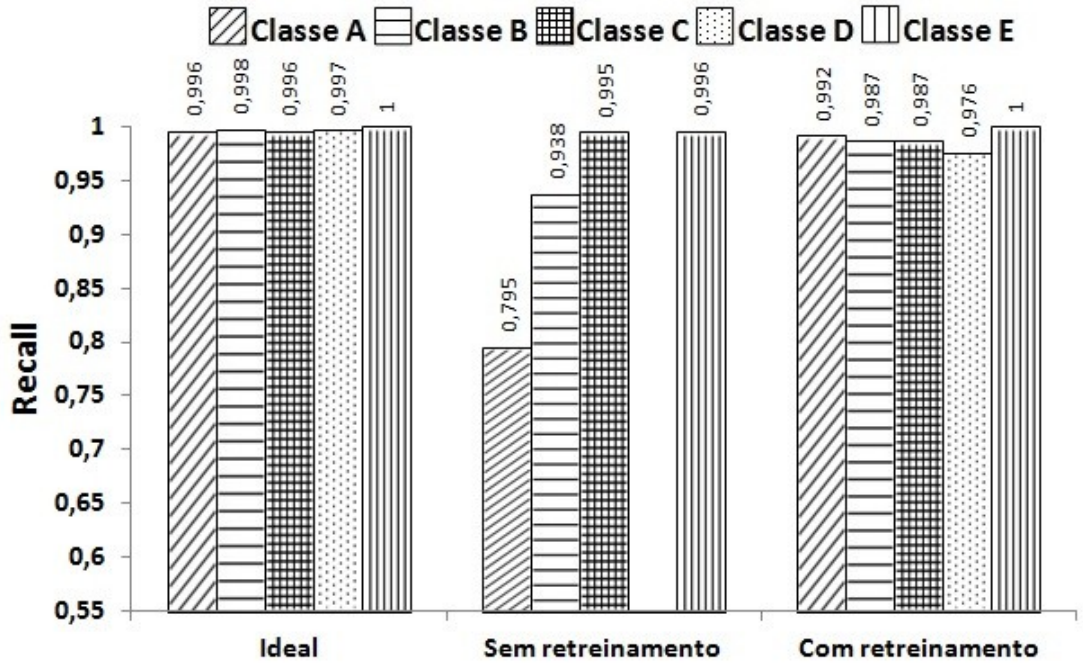


Figura 1.29 – Recall dos métodos de retreinamento por classe para o Cenário 4.

Nessa avaliação, as classes A, B e D têm um aumento no número de falsos negativos ao longo do tempo quando não é realizado retreinamento. Dessa forma, essas classes têm um menor índice de *recall* quando não realizam retreinamento. Observando tanto a precisão quanto o *recall,* constata-se que o método "Com retreinamento" alcança um desempenho próximo ao método ideal. A combinação dessas métricas pode ser vista na Figura 1.30. Os índices obtidos de *f-measure* indicam exatamente o comportamento esperado quando da proposta desse experimento. As classes com variações das respectivas distribuições estatísticas ao longo do tempo (A, B, C e D) apresentam um alto grau de degradação da classificação quando não é realizado nenhum retreinamento. Além disso, com o retreinamento proposto, a degradação é minimizada para todas as classes. Observando a classe E, destaca-se que não existem diferenças significativas independente da abordagem utilizada, já que não há mudanças no perfil de tráfego. Dessa forma, o experimento evidencia a eficácia do retreinamento na mitigação dos efeitos do *concept-drift*.

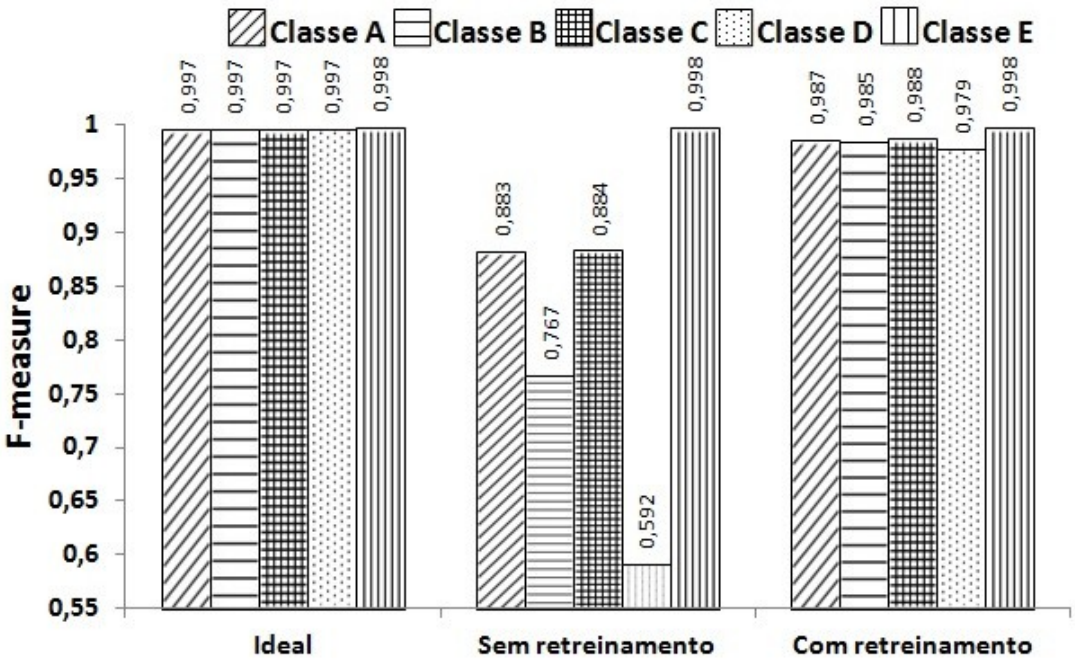


Figura 1.30 – *F-measure* dos métodos de retreinamento por classe para o Cenário 4

Embora o método "Com retreinamento" proposto seja dependente do componente Teste de Acurácia da arquitetura, foi possível obter índices de precisão, *recall* e *f-measure* muito próximos ao "Retreinamento ideal", que representa um limite superior para qualquer técnica.

Considerando a efetividade do retreinamento, é importante destacar o impacto que a forma que ocorre a mudança do perfil de tráfego, bem como as características que

definem o módulo Teste de Acurácia. Em outras palavras, a efetividade do retreinamento passa a estar dependente do limiar definido pelo administrador da rede para o módulo Teste de Acurácia e dependente da maneira como ocorre a mudança do perfil de tráfego. Na avaliação do Cenário 4, os pontos de mudança de perfil de tráfego são previamente conhecidos. Dessa forma, o retreinamento do Cenário 4 foi avaliado com as melhores configurações possíveis, fazendo com que os resultados demonstrem a máxima efetividade do retreinamento. No entanto, em um cenário real, a percepção da mudança do comportamento do tráfego deve ser mais lenta, a depender do limiar pré-definido. Isso implica que os ganhos de acurácia e precisão obtidos com o retreinamento possam ser menos significativos. É necessário, nesse caso, que exista um ajuste fino das características que definem a operação do módulo Teste de acurácia. A avaliação dessas características do módulo Teste de Acurácia faz parte dos trabalhos futuros propostos nesta tese.

Além do impacto na efetividade da classificação (considerando as métricas analisadas até aqui), o retreinamento proposto nesta tese implica na coexistência de dois classificadores na infraestrutura NFV durante um breve período de tempo. Eles coexistem enquanto o novo classificador está sendo preparado para substituir o primeiro. Nesse caso, existe um impacto computacional na máquina virtual que hospeda os classificadores, que corresponde ao Cenário 5 da avaliação proposta. Na Figura 1.31, é possível observar o consumo de memória RAM quando se ocorre ou não o retreinamento.

São identificados três momentos de medição. O primeiro momento de medição remete ao instante anterior ao início da classificação, o segundo trata do momento em que um novo perfil de tráfego foi detectado e, por fim, o terceiro momento remete ao período em que o perfil de tráfego já está estabilizado. Quando é analisada a classificação sem o retreinamento, a máquina virtual apresenta utilização de 26,36% de memória RAM, enquanto se mantém constante nos pontos 2 e 3 de medição (em aproximadamente 42,26%). Quando observamos a classificação que faz uso do retreinamento, o comportamento é modificado apenas no segundo ponto de medição. Nesse momento, o consumo de memória sobe 13,38%, saindo de 42,26% (sem retreinamento) para 55,64% (com retreinamento).

Mesmo com o crescimento no consumo de memória, é evidente que a quantidade de recursos necessários para realizar o retreinamento não é demasiada. Esses percentuais de consumo são relativos a uma máquina virtual com apenas 1,5 GB de memora RAM. Nesse caso, em relação ao consumo de memória, é possível concluir a viabilidade do método de retreinamento proposto.

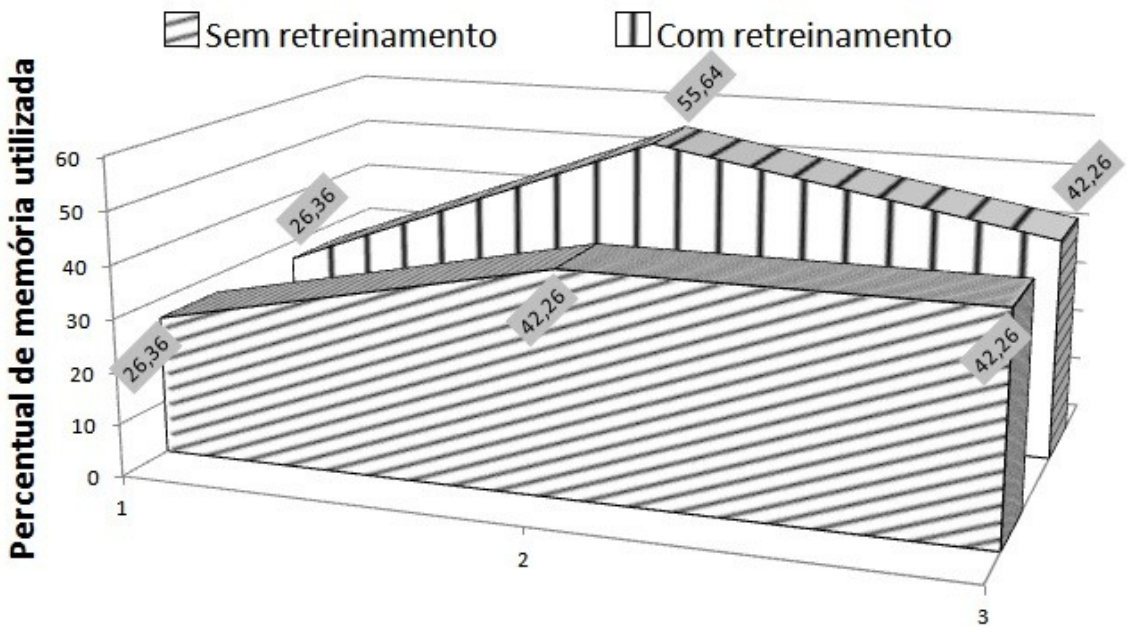


Figura 1.31 – Consumo de memória RAM na NFVI para o Cenário 5.

Adicionalmente, outras métricas foram coletadas considerando 3 momentos a serem discutidos: (i) servidor de virtualização inativo, (ii) classificação sem retreinamento e (iii) classificação com retreinamento. Na Figura 1.32, são apresentados os valores absolutos para interrupções do sistema operacional e trocas de contexto por segundo.

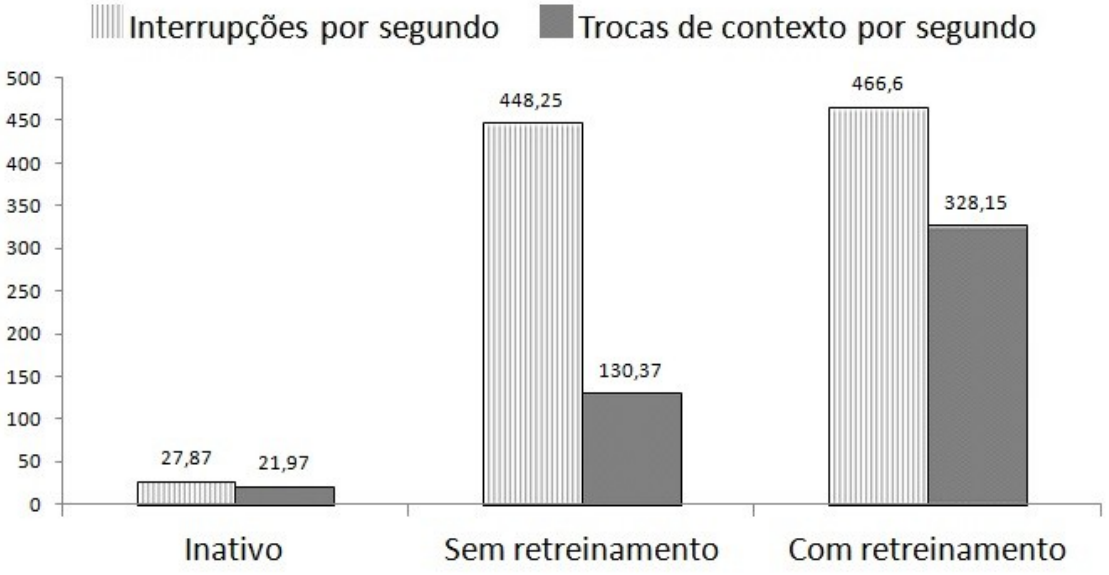


Figura 1.32 – Interrupções e trocas de contexto por segundo na NFVI para o Cenário 5.

O número de interrupções por segundo não apresenta grandes alterações quando se compara a abordagem tradicional e o método de retreinamento proposto. A máquina

virtual se mantém com o número de interrupções por segundo muito próximo nas duas abordagens. Por outro lado, o número de trocas de contexto cresce consideravelmente. Embora o número de trocas de contexto por segundo não seja excessivo, o crescimento se justifica porque o ambiente utilizado apresenta um único processador com um núcleo de processamento. Uma vez que dois processos passam a executar de maneira intensa (o classificador antigo e o novo classificador), passam a ser necessárias várias trocas de contexto. Apesar da limitação imposta pela infraestrutura considerada neste trabalho, é natural que, em um cenário real, o número de núcleos de processamento disponíveis seja maior.

7.3 Avaliação do posicionamento dos classificadores

As avaliações realizadas nesta subseção têm como objetivo verificar a efetividade do algoritmo proposto (PICO). O processo de definição do número de instâncias de classificadores envolve vários aspectos, como a disponibilidade de recursos e o conhecimento do administrador da rede, por exemplo. Os experimentos realizados nesta subseção consideram um número pré-estabelecido de classificadores. Em todos os casos, o posicionamento foi realizado até que algum dos métodos de posicionamento alcançasse um índice igual ou superior a 80% de cobertura de caminhos da referida topologia. Esse índice foi definido considerando-se o Princípio de Pareto (KIREMIRE, 2011). Em todas as topologias reais, a avaliação foi iniciada com 3 instâncias de classificadores. Para a topologia sintética, foram utilizados 10 classificadores inicialmente devido ao seu grande número de nós. Em todos os casos, são comparados 2 métodos de posicionamento: (i) posicionamento aleatório e (ii) PICO. No caso do posicionamento aleatório, para todos os cenários, foram executadas 30 rodadas em que é escolhido, ao acaso, um conjunto de nós para receber as instâncias dos classificadores e o valor médio é apresentado. Em todos esses casos, o intervalo de confiança foi calculado.

Para a avaliação do PICO, foram propostos dois cenários de avaliação, com fatores, níveis e objetivos distintos. A Tabela 1.16 apresenta um resumo dessas informações. Para o Cenário 6, são consideradas as 3 topologias reais e a topologia sintética. O objetivo dessa avaliação é identificar a efetividade do PICO em comparação ao posicionamento aleatório em um ambiente sem falhas. Já para o Cenário 7, o posicionamento do PICO é analisado quando ocorrem falhas na topologia. A ideia é verificar a resiliência a falhas do posicionamento proposto. Apenas as topologias reais são submetidas a falhas, que são de

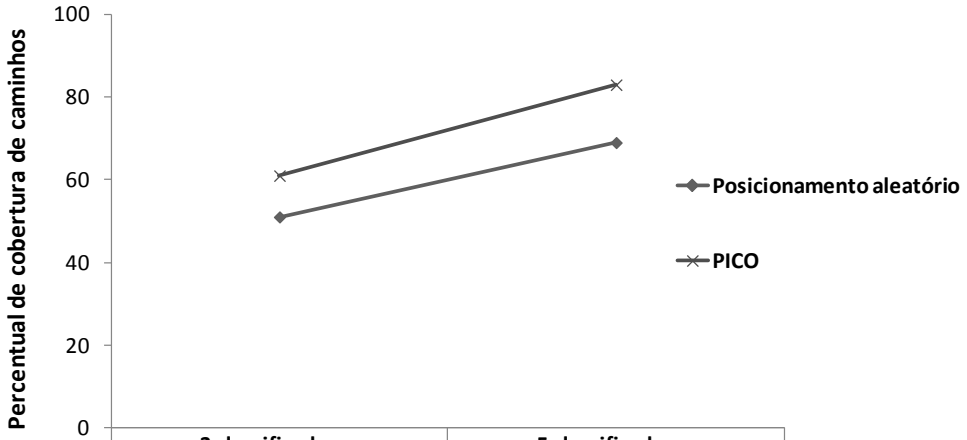
10% ou 20% dos nós da topologia (para os dois índices, caso o número de nós que falham não seja exato, o valor considerado é o inteiro superior mais próximo). Em todos os experimentos do Cenário 7, os nós que falham são escolhidos aleatoriamente e, para cada índice de falhas, são realizadas 30 execuções. Dessa forma, quando os resultados obtidos para esse cenário são discutidos, os respectivos intervalos de confiança serão apresentados.

Tabela 1.16 – Fatores e níveis para a avaliação do PICO.

Cenário	Fatores	Níveis	Métricas
Cenário 6	Topologias	IBM, MCI, AT&T e a topologia sintética.	Cobertura de caminhos da topologia.
	Número de classificadores	3, 5, 10, 20 e 30.	
Cenário 7	Topologias	IBM, MCI e AT&T.	Cobertura de caminhos da topologia.
	Número de classificadores	3, 5 e 10.	
	Percentual de falhas na rede	10% dos nós e 20% dos nós.	

Para prosseguir com a verificação da efetividade do PICO, cada uma das topologias detalhadas anteriormente são submetidas ao Cenário 6. Inicialmente, na Figura 1.33, a evolução do posicionamento é exposta para 3 e 5 classificadores para a topologia IBM. Nesse gráfico, observa-se que, desde o primeiro momento, a cobertura de caminhos obtida pelo PICO é bastante superior ao posicionamento aleatório. Quando são utilizados apenas 3 classificadores, o posicionamento realizado através do PICO provê um índice de cobertura de caminhos bastante interessante, superando o posicionamento aleatório em 10%. O comportamento dos algoritmos passa a ser mais relevante à medida que o número de classificadores cresce. Dessa forma, o PICO supera o desempenho médio da abordagem aleatória em 14% quando são utilizados 5 classificadores. Nesse caso, o intervalo de confiança calculado para o posicionamento aleatório indica que é possível afirmar com 95% de certeza que a cobertura alcançada estará entre 211 e 237 caminhos. Em números absolutos, o PICO obtém 255. Para a topologia IBM, o número mínimo necessário para

alcançar 100% de cobertura dos caminhos é 13 nós. Em comparação, o PICO alcança 83% de cobertura dos caminhos com apenas 5 nós da topologia, ou seja, menos que 40% do necessário para obter cobertura total e apenas 27,78% do número total de nós da topologia. Quantos menos nós são escolhidos para a topologia IBM, os benefícios da utilização do PICO ficam mais evidentes. Para confirmar os benefícios do PICO, mais topologias serão submetidas aos métodos de posicionamento.



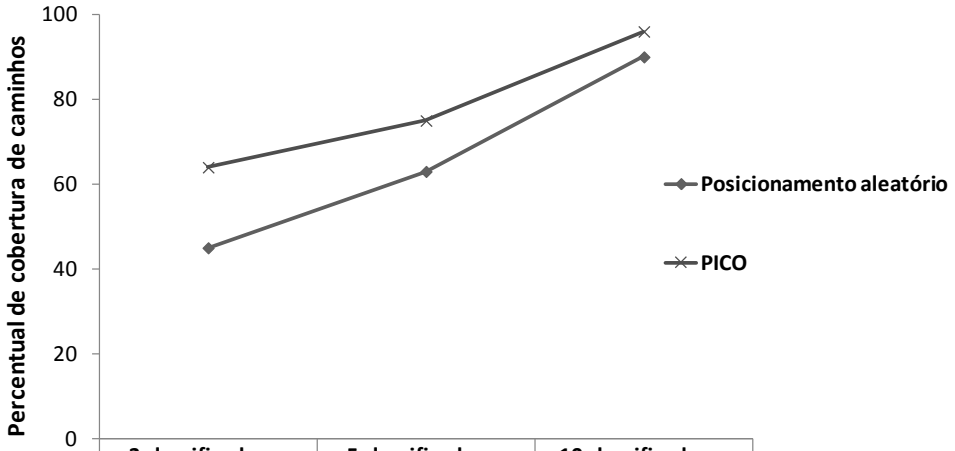
	3 classificadores	5 classificadores
Posicionamento aleatório	51	69
PICO	61	83

Figura 1.33 – Cenário 6 para a topologia IBM.

Na sequência, na Figura 1.34, os resultados dos experimentos para o Cenário 6 com a topologia MCI são apresentados. O comportamento observado é semelhante ao exposto na Figura 1.33. A cobertura de caminhos cresce com o crescimento do número de classificadores posicionados na topologia.

Mais uma vez, o PICO apresenta um desempenho muito superior ao posicionamento aleatório quando o número de classificadores (recursos disponíveis) é pequeno. Nos experimentos com 3 e 5 classificadores, o PICO supera o posicionamento aleatório em 19% e 12%, respectivamente. Considerando o número final de classificadores instanciados, o PICO apresenta a maior cobertura de caminhos. Como para o posicionamento aleatório os valores apresentados são médias, o intervalo de confiança é muito importante. Com 10 classificadores, é possível afirmar com 95% de certeza que a média da cobertura de caminhos do posicionamento aleatório estará, em valores absolutos, entre 300 e 315 caminhos. Com o PICO e 10 classificadores, são cobertos 326 caminhos. De acordo com o objetivo do administrador e com as características da rede, o volume de tráfego em cada caminho da rede pode ser muito grande e, portanto, ser essencial para operações de gerenciamento ou segurança, por exemplo. Adicionalmente, para a topologia MCI, o

número mínimo de nós para prover uma cobertura de caminhos total é de 11 classificadores. É interessante notar que o PICO obteve 96% de cobertura de caminhos posicionando 10 classificadores. É um índice de cobertura de caminhos muito alto e próximo de 100%, indicando que o algoritmo alcança um resultado próximo da solução ótima global (convergindo para o ótimo mais rapidamente que o posicionamento aleatório). Vale destacar que com apenas 5 classificadores, o PICO alcança uma cobertura de 75%.

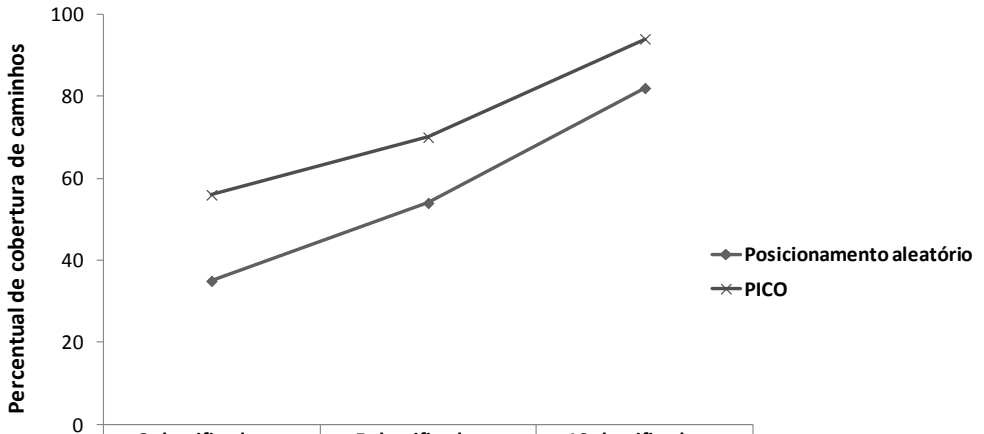


	3 classificadores	5 classificadores	10 classificadores
Posicionamento aleatório	45	63	90
PICO	64	75	96

Figura 1.34 – Cenário 6 para a topologia MCI.

Na Figura 1.35, o comportamento observado repete o que foi visto para as topologias anteriores. Quando o número de nós cresce, é natural que o número de caminhos apresente um aumento ainda maior. Na topologia AT&T, existem 600 caminhos, contra 306 da IBM e 342 da MCI.

Considerando o intervalo de confiança de 95% para o posicionamento aleatório com 10 classificadores na topologia AT&T, a cobertura de caminhos média está entre 470 e 514 caminhos. Quando se considera o PICO com 10 classificadores nessa topologia, são 566 caminhos cobertos, o que representa aproximadamente 94% de cobertura de caminhos. Os ganhos obtidos com a utilização do PICO são muito representativos, superando o posicionamento aleatório em 21% utilizando apenas 3 classificadores. Na topologia AT&T, o número mínimo de classificadores necessário para cobrir 100% dos caminhos é 18. Com 10 classificadores, o PICO alcança 94% de cobertura de caminhos. Isso representa uma economia significativa de recursos para o administrador da rede, uma vez que, com aproximadamente 55% do número de classificadores necessários para a cobertura de caminhos total, os caminhos da topologia são quase completamente cobertos.



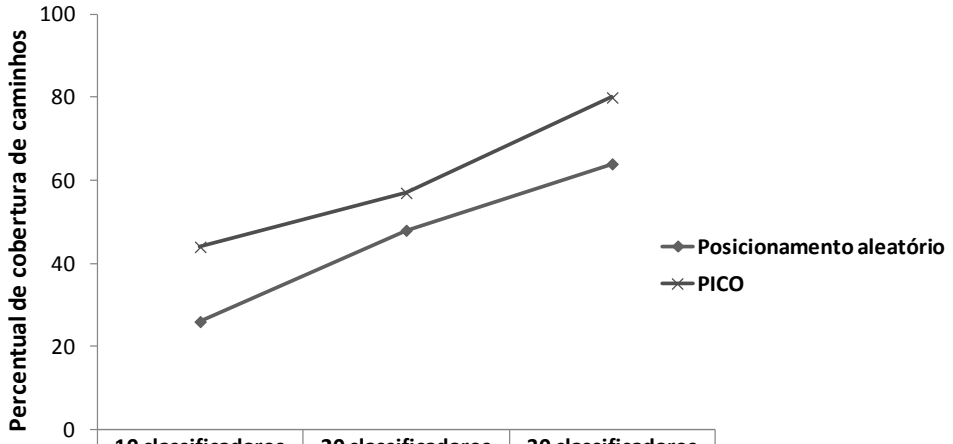
	3 classificadores	5 classificadores	10 classificadores
Posicionamento aleatório	35	54	82
PICO	56	70	94

Figura 1.35 – Cenário 6 para a topologia AT&T.

Para todas as topologias avaliadas até este ponto, o PICO surge como a melhor escolha do ponto de vista da cobertura de caminhos, principalmente quando o número de classificadores é pequeno em relação ao número total de nós da topologia. Os ganhos com poucos classificadores são muito grandes, demonstrando uma importante eficácia da otimização. Quando o número de classificadores a ser posicionado cresce, o PICO permanece apresentando os melhores resultados, embora os ganhos em relação ao posicionamento aleatório sejam minimizados. Dessa forma, é possível afirmar que o PICO apresentou resultados consistentes e um alto índice de cobertura dos caminhos da topologia.

Por fim, para reforçar a aplicabilidade do PICO no cenário de posicionamento de classificadores, uma avaliação com uma topologia sintética foi realizada visando a análise da escalabilidade da solução. Para isso, a topologia sintética foi submetida ao Cenário 6 e as coberturas de caminhos obtidas apresentadas na Figura 1.36. Assim como observado para as topologias reais, o PICO alcança maiores índices de cobertura de caminhos que o posicionamento aleatório. É notório que, quanto maior a topologia, mais recursos serão necessários para que o administrador alcance resultados interessantes. Nesse caso, utilizando 30 classificadores, calculando o intervalo de confiança, conclui-se com 95% de certeza que o posicionamento aleatório alcançará, em média, entre 26456 e 29664 caminhos cobertos. Com o PICO e o mesmo número de classificadores, são cobertos 35329 caminhos. Da mesma forma que acontece para as demais topologias, o PICO apresenta resultados consistentes, sendo superior que o posicionamento aleatório em até 18%, dependendo do número de classificadores. Adicionalmente, com pelo menos 64

classificadores, é possível cobrir completamente os caminhos da topologia sintética. Na Figura 1.36, o PICO apresenta uma cobertura de 80% com apenas 30 classificadores (aproximadamente 47% do necessário para a cobertura de caminhos total e aproximadamente 14% do número total de nós).



	10 classificadores	20 classificadores	30 classificadores
Posicionamento aleatório	26	48	64
PICO	44	57	80

Figura 1.36 – Cenário 6 para a topologia sintética

Diante disso, uma importante discussão está relacionada à utilização de informações sobre o tráfego no algoritmo de posicionamento (em outras palavras, considerar a distribuição dos fluxos na rede). Além das questões envolvendo a complexidade algorítmica decorrente da utilização de uma função multiobjetivo discutidas na Seção 5.2.4, a ideia é manter o PICO alcançando o máximo percentual de cobertura de caminhos da rede sem que sejam necessárias informações adicionais (além da topologia). Adicionalmente, quando apresentada a proposta de classificação distribuída no capítulo inicial desta tese, a resiliência a falhas foi uma das características interessantes da arquitetura proposta. Embora o fato de ter vários classificadores já apresente algum grau de resiliência a falhas (visto que, em caso de falha de um determinado classificador da rede, os demais classificadores continuam operando), o PICO provê um posicionamento observando pontos "estratégicos" da topologia. Dessa forma, mesmo que a topologia seja alterada por conta de falhas em alguns nós, o posicionamento fornecido pelo algoritmo proposto deve fornecer um alto percentual de cobertura de caminhos com os classificadores que ainda se mantiverem ativos. Caso os fluxos fossem levados em consideração para que o posicionamento fosse definido, é provável que uma nova execução do algoritmo de posicionamento fosse necessária já que uma nova distribuição de fluxos passaria a existir (influenciando, portanto, na nova distribuição dos classificadores). Essa nova execução do algoritmo geraria novas

trocas de mensagens entre o módulo de controle e a rede, já que, possivelmente, novas instanciações de classificadores seriam necessárias, gerando um custo adicional no momento de falha.

A fim de atestar a resiliência a falhas do PICO, uma vez que o desempenho do PICO foi discutido quando a rede apresenta condições normais, é importante verificar o quão efetivo é o algoritmo proposto quando ocorrem falhas na topologia da rede. Nesse caso, serão apresentados os experimentos relacionados ao Cenário 7. A Figura 1.37 expõe a cobertura de caminhos do PICO quando ocorrem 10% (2 nós) e 20% (4 nós) de falhas para a topologia IBM.

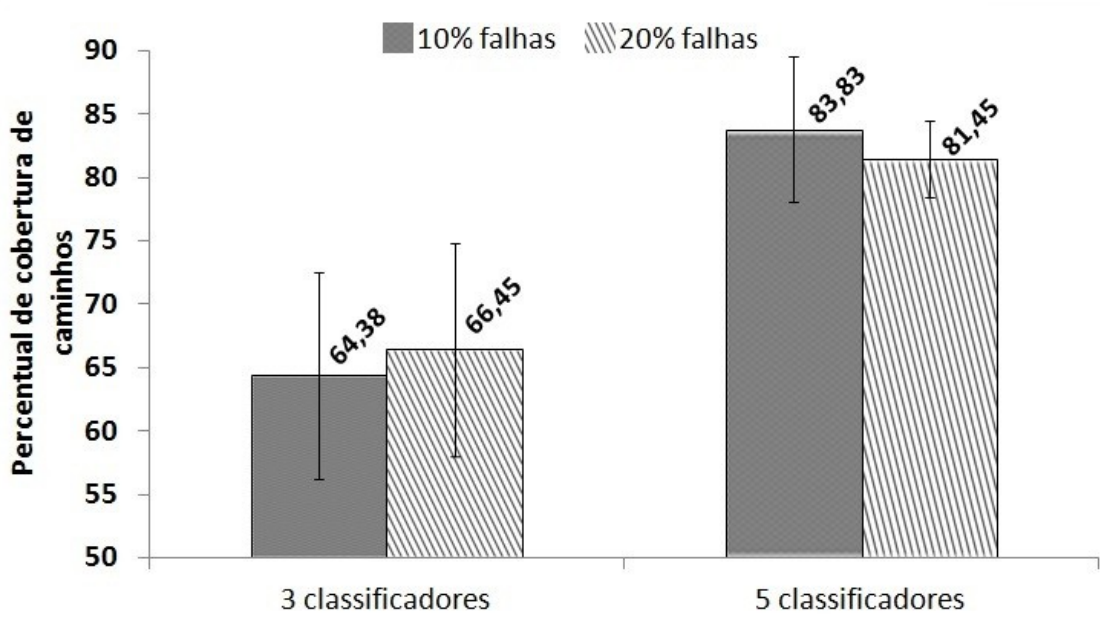


Figura 1.37 – Experimento do Cenário 7 para a topologia IBM.

Quanto menor o número de classificadores, maior o impacto das falhas (o desvio padrão é maior). Isso acontece porque quando o nó que falha hospeda um classificador, essa falha é maior proporcionalmente. Por exemplo, caso estejam presentes 3 classificadores na rede e 1 deles falhe, 33% dos classificadores deixará de ser operacional. Por outro lado, se o número total de classificadores for 5, apenas 20% dos classificadores deixará de ser operacional.

Entretanto, analisando a cobertura de caminhos obtida diante das falhas, o índice obtido se mantém na média. Na Tabela 1.17, os intervalos de confiança calculados são apresentados para cada um dos casos. A confiança nos intervalos obtidos é de 95%. Percebe-se, como dito anteriormente, que quanto maior o número de classificadores disponíveis, menor o impacto das falhas. Adicionalmente, é importante destacar que o

posicionamento obtido pelo PICO mantém em média um percentual alto de cobertura dos caminhos da topologia, próximo ao obtido sem falhas.

Tabela 1.17 – Intervalos de confiança para o Cenário 7 com a topologia IBM.

Número de classificadores	Índice de falhas da topologia	Limite inferior do intervalo	Limite superior do intervalo	Valor médio obtido de cobertura de caminhos
3	10%	57,19%	71,57%	64,38%
3	20%	59,1%	73,8%	66,45%
5	10%	78,8%	88,86%	83,83%
5	20%	78,45%	84,46%	81,45%

Um comportamento bastante similar pode ser observado na Figura 1.38, em que o experimento ocorre no Cenário 7 com a topologia MCI.

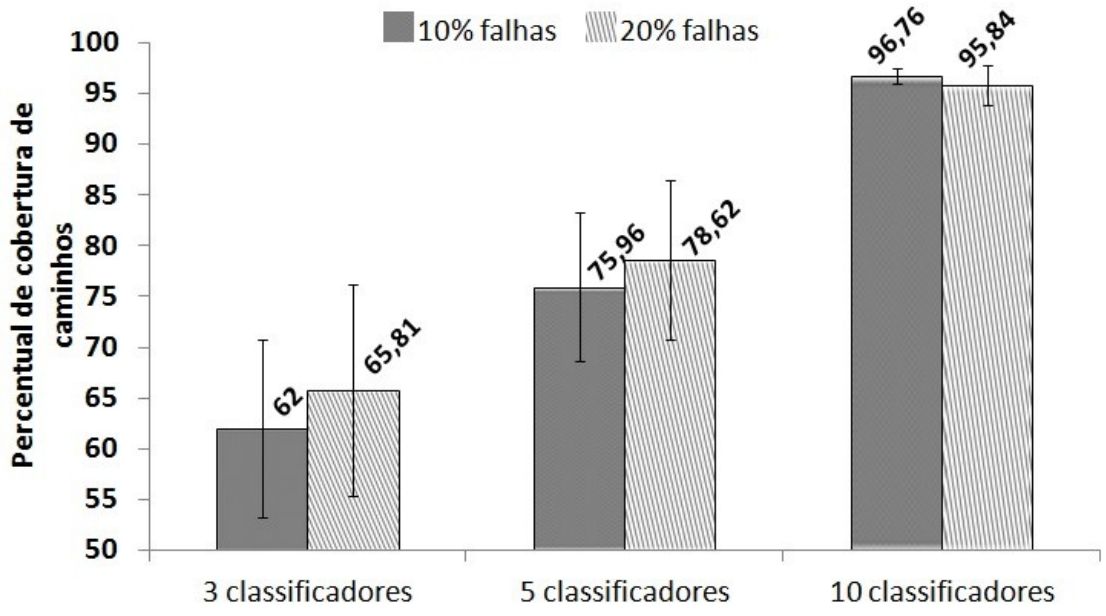


Figura 1.38 – Experimento do Cenário 7 com a topologia MCI.

Assim como no experimento anterior, o desvio padrão da cobertura de caminhos diminui à medida que o número inicial de classificadores aumenta. A cobertura média para cada uma das configurações (3, 5 e 10 classificadores) é bastante próxima ao observado nos

experimentos sem falhas. Com 10 classificadores, por exemplo, o PICO alcança 96% de cobertura de caminhos sem falhas, 96,76% com 10% de falhas e 95,84% com 20% de falhas.

Na Tabela 1.18 são apresentados os intervalos de confiança para cada um dos fatores e respectivos níveis considerados no Cenário 7. A confiança nos intervalos obtidos é de 95%. Quando o número de classificadores aumenta, o percentual de cobertura de caminhos será mais próximo à média obtida. Com 10% de falhas, são 2 nós que falham na topologia. Já com 20% de falhas, serão 4 nós que apresentam problemas. Dessa forma, mesmo que os 4 nós falhos hospedem classificadores, no experimento com 10 classificadores, ainda restam 6 classificadores em funcionamento.

É importante destacar que, quando existem falhas na topologia, o número de caminhos total da topologia diminui. Isso acontece porque todos os caminhos que continham os nós que falharam, deixam de existir e seus *links* deixam de ser considerados. Nesse caso, mesmo perdendo 4 classificadores, por exemplo, ainda é possível obter um grande percentual de cobertura, já que o total de caminhos foi reduzido.

Tabela 1.18 – Intervalos de confiança para o Cenário 7 com a topologia MCI.

Número de classificadores	Índice de falhas da topologia	Limite inferior do intervalo	Limite superior do intervalo	Valor médio obtido de cobertura de caminhos
3	10%	56,55%	67,45%	62%
3	20%	57,72%	72,16%	65,81%
5	10%	69,58%	82,33%	75,96%
5	20%	70,95%	81,82%	78,62%
10	10%	96,03%	97,5%	96,76%
10	20%	94,1%	97,57%	95,84%

Em seguida, os experimentos são realizados no Cenário 7 considerando a topologia AT&T a fim de ratificar o comportamento observado para as topologias reais analisadas anteriormente do ponto de vista de resiliência a falhas.

Na Figura 1.39, são apresentados os resultados dos experimentos realizados considerando os diferentes fatores e níveis para o Cenário 7. Mais uma vez, o comportamento percebido no gráfico é bastante similar ao observado anteriormente. Adicionalmente, assim como observado nas topologias anteriores, o desvio padrão observado no gráfico diminui à medida que o número de classificadores aumenta. Especificamente com 10 classificadores, o desvio padrão é pequeno em relação à média. Além disso, os valores médios de cobertura de caminhos obtidos com falhas (93,14% e 92,98%) se aproximam bastante ao obtido no experimento sem falhas (94%). Nesse caso, o PICO apresenta um posicionamento bastante resiliente a falhas, fornecendo um alto percentual de cobertura de caminhos das topologias mesmo quando ocorrem falhas de nós. Especificamente para a topologia AT&T, 10% de falhas correspondem a 3 nós, enquanto 20% de falhas correspondem a 5 nós.

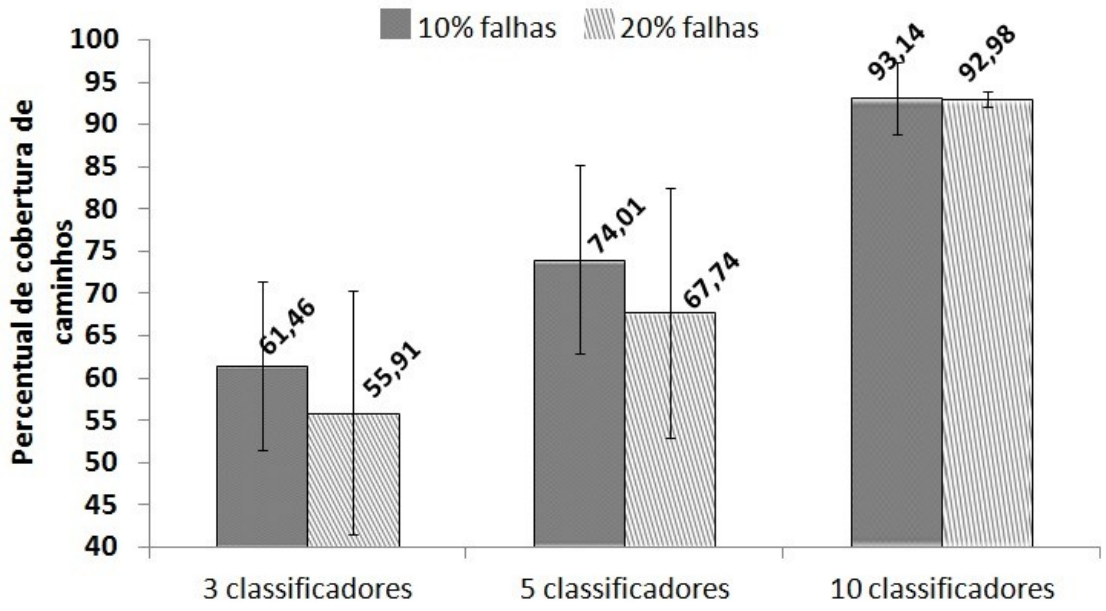


Figura 1.39 – Experimento do Cenário 7 com a topologia AT&T.

Na Tabela 1.19, os intervalos de confiança obtidos para o Cenário 7 com a topologia AT&T são apresentados. A confiança, mais uma vez, é de 95% nos intervalos obtidos. Os resultados dos experimentos do Cenário 7 mantêm o mesmo comportamento mesmo diante da variação da topologia, atestando a capacidade do PICO fornecer um

posicionamento de classificadores com alto percentual de cobertura de caminhos, considerando que podem existir falhas na topologia ou não.

Por fim, analisando todas as avaliações realizadas, é possível afirmar que o PICO é uma abordagem efetiva para posicionamento de classificadores virtuais em uma determinada topologia. Esse método maximiza a cobertura de caminhos, superando a abordagem aleatória, possibilitando o monitoramento e classificação de um grande número de fluxos. Os resultados demonstraram uma relevante economia de recursos, já que grandes índices de cobertura de caminhos são alcançados com poucas instâncias virtuais de classificadores. Além disso, os experimentos demonstraram que o posicionamento provido pelo PICO é resiliente a falhas na topologia. Nesse caso, fica claro que o PICO é uma alternativa interessante para posicionamento de funções virtuais quando o administrador da rede possui uma quantidade limitada de recursos.

Tabela 1.19 – Intervalos de confiança para o Cenário 7 com a topologia AT&T.

Número de classificadores	Índice de falhas da topologia	Limite inferior do intervalo	Limite superior do intervalo	Valor médio obtido de cobertura de caminhos
3	10%	55,24%	67,68%	61,46%
3	20%	44,37%	67,45%	55,91%
5	10%	66,27%	81,76%	74,01%
5	20%	57,52%	77,96%	67,74%
10	10%	89,43%	96,85%	93,14%
10	20%	92,17%	93,79%	92,98%

7.4 Avaliação da instanciação de classificadores

A instanciação de um classificador de tráfego virtual é diretamente relacionada à instanciação de uma função virtual da rede. Como descrito anteriormente, o tempo necessário para inicializar o provedor de virtualização e o tempo necessário para construir o

classificador compõem o atraso imposto à rede para iniciar a operação do classificador. O tempo total de atraso será medido como a soma do tempo de inicialização da máquina virtual (tempo do provedor) e o tempo necessário para o treinamento do classificador (tempo de construção). Depois desse atraso, a árvore de classificação estará pronta no servidor virtual para realizar suas atividades.

Para realização dos experimentos do Cenário 8, foram considerados os fatores e níveis explicitados na Tabela 1.20. Cada um dos experimentos foi executado 30 vezes, o desvio padrão obtido não é significativo, mas é apresentado no gráfico a seguir. Dessa forma, todas as análises realizadas, consideram apenas a média de tempo obtida.

Nessa avaliação, o provedor NFV escolhido foi o ClickOS conforme apresentado em (BONDAN et al., 2016). Trata-se de uma máquina virtual baseada em Linux capaz de isolar funções virtuais previamente instaladas. A avaliação realizada nesse ponto é limitada à instanciação de um classificador isolado em um provedor. A ideia é medir o atraso de um classificador e medir o impacto do número de instâncias do conjunto de treinamento.

Tabela 1.20 – Fatores e níveis para a avaliação da instanciação de classificadores.

Fatores	**Níveis**
Número de instâncias de fluxos no conjunto de treinamento	1479 fluxos, 2958 fluxos, 4437 fluxos, 5916 fluxos, 7395 fluxos, 8874 fluxos, 10353 fluxos, 11832 fluxos e 13311 fluxos.

Como observado em (BONDAN et al., 2016), o tempo de início de um provedor para receber uma função virtual é de 0,025 segundos (isso é uma métrica constante, realizada anteriormente ao treinamento). Na Figura 1.40, é observado o tempo total do processo de instanciação para diferentes tamanhos de conjuntos de treinamento. Nesse caso, o *trace* do gt foi utilizado no experimento. Considerando o número total de fluxos de 14790 (conjunto de treinamento + conjunto de teste), o conjunto de treinamento varia 10% a cada execução, iniciando em 10% (1479) e indo até 90% (13311).

É possível observar que, para todas as avaliações realizadas, a instanciação dura menos de 1 segundo. Isso permite uma rápida provisão do classificador como um serviço virtual na rede, além de habilitar uma rápida reconfiguração da disposição dos serviços (reposicionamentos). Com o tempo para início do classificador virtual sendo baixo, o volume de tráfego que não será avaliado enquanto a operação não inicia também será

baixo. Por outro lado, é importante destacar que existe um crescimento do tempo de instanciação de acordo com o aumento do tamanho do conjunto de treinamento. Por exemplo, um classificador estará apto a operar em aproximadamente 0,614 segundos quando o conjunto de treinamento tem 5916 fluxos, mas demorará 0,822 segundos quando o conjunto de treinamento tiver 8874 fluxos. Embora para todos os conjuntos de treinamento o tempo de instanciação seja inferior a 1 segundo, do menor para o maior conjunto de treinamento é apresentado um crescimento de 189% do tempo de instanciação. Dessa forma, o tamanho do conjunto de treinamento deve ser tão pequeno quanto possível, já que impacta no atraso da instanciação do classificador virtual, além de ser uma restrição da área de mineração de fluxos de dados.

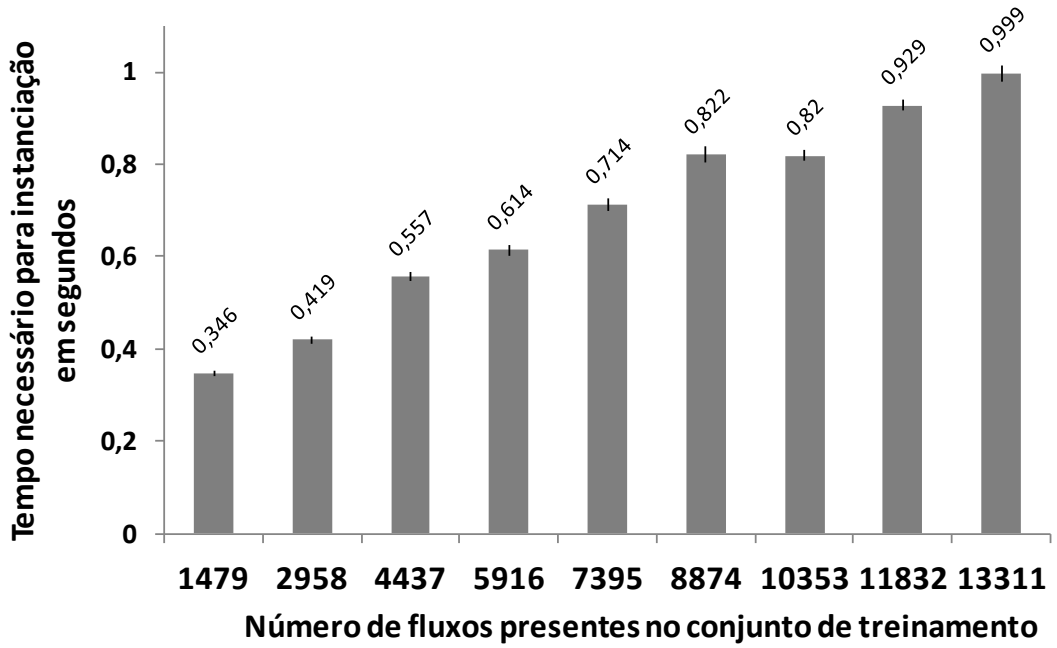


Figura 1.40 – Avaliação do tempo de instanciação (Cenário 8).

Finalmente, é interessante notar que rápidas instanciações permitem reconfiguração do posicionamento dos classificadores (devido à alteração proposta para aumentar a cobertura ou à redução de recursos do administrador da rede), bem como a expansão do número de classificadores virtuais caso mais recursos sejam disponibilizados.

7.5 Ameaças à validação

A investigação da validade da pesquisa está na relação entre as conclusões obtidas e a realidade. Em outras palavras, a validação preocupa-se com a possibilidade das conclusões estarem erradas (MAXWELL, 2013). Em (COOK et al., 1979), quatro tipos principais de

ameaças à validação foram apresentadas: (i) ameaças à validade da conclusão, (ii) ameaças à validade da construção, (iii) ameaças à validade interna e (iv) ameaças à validade externa.

Neste trabalho, algumas ameaças à validação foram identificadas, sendo uma parte delas diretamente tratadas e outras sendo relacionadas como trabalhos futuros. Nas subseções a seguir, algumas dessas ameaças são categorizadas e discutidas.

7.5.1 Ameaças à validade da conclusão

A validade da conclusão reside na confirmação de que a saída de um determinado experimento está relacionada às medidas adotadas na experimentação. Isso significa dizer que a validade da conclusão visa atestar que os resultados não seriam aleatórios.

Considerando a validade da conclusão, a avaliação da acurácia se ajusta a esse tipo de ameaça. Isso porque a avaliação foi apresentada para os *traces* como um todo. Nesse caso, a colaboração entre os classificadores poderia categorizar os fluxos em uma única classe (a mais frequente), incrementando a acurácia geral, mas degradando a efetividade para algumas classes. Para contornar essa ameaça, foram calculadas algumas métricas (*recall,* precisão e *f-measure*) para cada uma das classes mais frequentes. Essa análise visa demonstrar a solidez dos resultados apresentados. Os ganhos são observados para todas as classes, demonstrando que se trata de um crescimento consistente da efetividade da classificação.

7.5.2 Ameaças à validade da construção

A validade da construção está focada na relação entre a teoria que suporta o experimento e o que é observado do experimento propriamente dito. Mesmo que as conclusões obtidas sejam válidas, os fatores observados e modificados podem não corresponder à realidade.

Neste trabalho, o escopo de implantação da arquitetura proposta foi definido considerando uma infraestrutura totalmente virtualizada. Caso o modelo de classificação proposto seja aplicado em um cenário apenas parcialmente virtualizado, os resultados podem ser afetados. Nesse caso, a implantação em uma rede parcialmente virtualizada se configura em uma ameaça à validade da construção (embora a restrição do escopo da tese previna tal ameaça).

Além disso, todos os classificadores classificam todos os fluxos nas avaliações realizadas nesta pesquisa. Nesse caso, com o encaminhamento real ocorrendo nos

dispositivos da rede, nem todos os fluxos passarão por todos os classificadores, podendo afetar a avaliação do modelo proposto de classificação. Dessa forma, avaliar o comportamento da classificação diante dessa condição do encaminhamento de fluxos na rede é um dos trabalhos futuros listados nesta tese.

7.5.3 Ameaças à validade interna

A validade interna remete à relação de causa e efeito da configuração de um experimento e do respectivo resultado. Podem existir outros fatores que não são controlados ou não mensurados que potencialmente afetariam o experimento.

Considerando a validade interna, algumas ameaças foram identificadas: (i) a troca de mensagens entre os módulos e (ii) o comportamento do módulo de teste de acurácia. A primeira ameaça não foi mensurada, as comunicações entre os módulos da arquitetura podem inserir uma sobrecarga na rede. Nesse caso, a arquitetura proposta poderia ser avaliada sob esse ponto de vista mediante a sua implementação em um ambiente real, permitindo a análise (dentre outros aspectos) da troca de mensagens entre os módulos. A segunda ameaça destacada (o comportamento do teste de acurácia) pode interferir diretamente no comportamento e na efetividade do retreinamento. Se o teste for muito sensível a mudanças de perfil de tráfego, muitos retreinamentos serão necessários. Por outro lado, se o teste de acurácia for pouco sensível a mudanças no tráfego, os classificadores não se adaptarão. Além disso, o tempo necessário para perceber uma determinada mudança no perfil de tráfego pode limitar a efetividade da classificação. A avaliação da configuração e do comportamento do módulo de teste de acurácia está entre os passos futuros deste trabalho.

7.5.4 Ameaças à validade externa

A validade externa se refere à capacidade de generalização dos resultados obtidos para cenários fora do escopo avaliado. Mesmo que os experimentos e resultados apresentem uma relação de causa e efeito e que os resultados possuam significância estatística, é necessário que as conclusões obtidas se mantenham em situações diferentes das avaliadas.

Nesse caso, é possível destacar duas ameaças à validade externa desta tese: (i) a representatividade dos *traces* utilizados e (ii) a diversidade das topologias avaliadas. No primeiro caso, os *traces* utilizados configuram um cenário específico de avaliação, o que

dificultaria a extensão das conclusões para outros conjuntos de dados sem avaliá-los. No entanto, é interessante destacar a independência das técnicas de classificação (neste trabalho, o C4.5 especificamente é utilizado) em relação ao número de classes e fluxos presentes nos conjuntos de dados. Uma vez que um conjunto de dados de treinamento seja obtido contendo as classes que serão classificadas, o classificador será capaz de identificar os novos fluxos de acordo com o treinamento. Vale salientar que o objetivo do trabalho não trata da melhoria específica da técnica de aprendizagem utilizada, mas da melhoria da classificação geral através da colaboração entre classificadores, e que outros trabalhos apresentam diferentes técnicas aplicadas com sucesso em outros *traces*. Dessa forma, se conclui que a pequena diversidade de *traces* não compromete a validade externa da pesquisa. Entretanto, a partir da implantação da arquitetura proposta em um cenário real, a avaliação com maior diversidade de classes será, consequentemente, realizada. De maneira similar, o conjunto restrito de topologias utilizadas para avaliação do posicionamento dos classificadores pode ser tratada como uma ameaça à validade externa dos experimentos. Nesse caso, foram escolhidas topologias com características diferentes (número de nós, número de *links,* relação entre o número de nós e de caminhos, entre outras) para minimizar a possibilidade dessa ameaça. Em outras palavras, o posicionamento dos classificadores foi realizado em cenários com diferentes configurações para validar sua efetividade. No entanto, a demonstração da efetividade do algoritmo proposto para uma topologia qualquer é uma etapa subsequente para a evolução do algoritmo.

8 Conclusões e trabalhos futuros

O gerenciamento das redes é essencial para que elas operem de maneira satisfatória. Para isso, a classificação de tráfego se constitui como uma ferramenta extremamente valiosa. A partir da identificação do tipo de dados que trafegam na rede, é possível tomar medidas convenientes ao administrador. Fornecimento de uma qualidade de serviço diferenciada para uma determinada aplicação, por exemplo, é uma das possíveis aplicações da classificação de tráfego. Outras aplicações possíveis a partir da classificação de tráfego são interessantes como, por exemplo, posicionamento de servidores de *cache* de conteúdo, bloqueio de aplicações indesejadas (segurança) ou mesmo limitação da banda disponível para determinado tipo de tráfego. Nesse cenário, a aplicação de melhorias no processo de classificação sob diferentes aspectos desempenha um papel fundamental para a área de redes de computadores. Em geral, pesquisas nessa área focam nas melhorias aplicadas à técnica de classificação ou ao *hardware* utilizado. Diante desse cenário, este trabalho manteve o foco no desenvolvimento e avaliação de alguns módulos de uma nova arquitetura distribuída proposta, capaz de lidar com diferentes questões existentes na classificação de tráfego.

8.1 Contribuições

As principais contribuições desta pesquisa têm origem na concepção da nova arquitetura de classificação distribuída baseada nos conceitos de redes virtualizadas. A arquitetura combina características dos paradigmas SDN e NFV para sustentar a classificação de tráfego distribuída, apresentando algumas características inerentes a aplicações distribuídas, como tolerância a falhas e confiabilidade, além de prover adaptação a mudanças na rede. Essa arquitetura está fortemente correlacionada ao objetivo geral proposto de acordo com a hipótese inicial desta tese, ambos descritos no Capítulo 0 deste trabalho. A classificação distribuída, conforme discutido ao longo do trabalho, permite a diversificação dos classificadores e o consequente aumento da efetividade da classificação através da colaboração. Além disso, a classificação passa a apresentar tolerância a falhas inerente à distribuição dos classificadores e um menor custo para instanciação de classificadores devido à virtualização dos serviços da rede. Além do objetivo geral, foram propostos alguns

objetivos específicos, relacionados às questões de partida identificadas para o desenvolvimento desta tese (Tabela 1.21).

Tabela 1.21 – Relação entre objetivos específicos e atividades implementadas

Objetivos específicos	Questões de partida	Atividades implementadas
OE01 – Propor e avaliar a interação entre classificadores de tráfego para que a acurácia seja melhorada	QP01, QP02, QP03 e QP07	Arquitetura proposta e métodos de colaboração (VS e GC).
OE02 – Desenvolver e avaliar um algoritmo de posicionamento de classificadores em uma rede de topologia genérica, visando maximizar o número de fluxos coletados levando em conta a limitação de recursos	QP04 e QP05	Proposição do PICO e avaliação de seu desempenho considerando o percentual de cobertura de caminhos da topologia.
OE03 – Avaliar os custos operacionais envolvidos no posicionamento de um classificador virtual	QP06	Aferição do tempo de instanciação, além da memória, das interrupções e trocas de contexto.
OE04 – Desenvolver o método de operação e gerenciamento semi-autônomo da arquitetura com base em uma rede SDNFV	QP07	Descrição do papel da SDN, além da operação do posicionamento e do teste de acurácia.
OE05 – Possibilitar e avaliar a adaptabilidade da arquitetura a mudanças do perfil de tráfego, baseado em retreinamento de técnicas de aprendizagem de máquina	QP08 e QP09	Discussão sobre o teste de acurácia e retreinamento, avaliando um possível cenário de retreinamento.
OE06 – Analisar o comportamento da arquitetura proposta em relação às características desejadas na concepção da classificação distribuída	QP09	Avaliação da acurácia, do retreinamento, do PICO, da instanciação e resiliência a falhas.

Na Tabela 1.21, são expostos todos os objetivos específicos, as questões de partida relacionadas e o que foi feito para alcançá-los. Observando a tabela, é possível perceber que todos os objetivos específicos foram alcançados ao longo do desenvolvimento desta tese. Especificamente, a proposição da arquitetura distribuída, a proposição do PICO e as avaliações dos módulos individualmente permitiram que esses objetivos fossem atingidos. Consequentemente, as principais contribuições deste trabalho foram apresentadas a partir do momento em que objetivos específicos foram alcançados para composição da arquitetura. Nesse caso, 5 contribuições merecem destaque:

1) A proposição do algoritmo de posicionamento de classificadores otimizado (PICO), que aponta os nós mais indicados para hospedar um classificador, visando a maximização dos fluxos coletados e permitindo um desempenho bastante superior ao posicionamento aleatório;

2) A apresentação da análise da resiliência a falhas a partir do posicionamento dos classificadores quando ocorrem falhas aleatórias na topologia;

3) A apresentação da análise do impacto do tamanho do conjunto de treinamento no tempo de instanciação de um classificador virtual;

4) A apresentação de dois esquemas de consolidação da classificação de tráfego distribuída, o VS e o GC, e;

5) A proposição do mecanismo de retreinamento das técnicas dentro da arquitetura proposta, que possibilita a otimização do tempo de treinamento e a adaptação a mudanças no perfil do tráfego.

Os resultados apresentados neste trabalho atestam o conceito da arquitetura proposta. Mais importante, quantificam os benefícios obtidos pela utilização dos módulos desta arquitetura, demonstrando sua viabilidade, além de seu potencial de auxiliar o administrador da rede de acordo com seus objetivos.

A avaliação da efetividade da colaboração dos classificadores apresentou um ganho de até 5% na acurácia. No cenário de uma grande rede corporativa ou *backbone,* a classificação da arquitetura proposta pode prover um ganho de milhares de fluxos classificados corretamente. É importante destacar que a colaboração permite que uma única abordagem seja sempre utilizada e apresente acurácia superior à obtida com uma técnica tradicional. Observando a cobertura de *bytes* e pacotes, um comportamento similar ao da acurácia é

obtido. Dentre os esquemas de colaboração propostos, o VS apresentou um desempenho superior em praticamente todas as avaliações realizadas.

Em relação à efetividade da classificação quando o retreinamento acontece, o esquema proposto obtém um ganho de aproximadamente 14,5% na acurácia comparado ao esquema sem retreinamento. Vale destacar que, na avaliação realizada nesta tese, o retreinamento proposto apresentou uma efetividade da classificação muito próxima à obtida com a abordagem de retreinamento ideal, que representa um limite superior para qualquer técnica a ser proposta.

Em seguida, o algoritmo de posicionamento de classificadores foi apresentado e discutido de maneira teórica com base em conceitos de teoria dos grafos. Uma vez que o PICO foi proposto através de um método de otimização, visando a obtenção de um posicionamento ótimo. O PICO alcançou ganhos superiores a 20% da cobertura em relação ao posicionamento aleatório em alguns casos. Em todos os cenários, a informação utilizada pelos métodos de posicionamento é exclusivamente referente à topologia da rede.

Por fim, foi avaliado e discutido o tempo necessário para instanciar um classificador virtual como uma função. No cenário proposto nesta tese, para todos os conjuntos de treinamento considerados, a instanciação durou menos que 1 segundo. Esse tempo contabiliza tanto a inicialização do provedor, bem como o tempo de construção do classificador. Nesse caso, a rápida instanciação ratifica a possibilidade de operação em tempo real dos módulos propostos da arquitetura.

8.2 Trabalhos futuros

Devido à quantidade de módulos da arquitetura proposta, não seria possível avaliar todos os aspectos dos módulos propostos em um único trabalho. O escopo definido para esta tese é limitado. Dessa forma, algumas outras avaliações interessantes serão tratadas em trabalhos futuros.

A capacidade de uma rede SDN de redirecionar fluxos de acordo com alguma condição não foi explorada nesta pesquisa. Nesse caso, avaliar o balanceamento de carga entre os classificadores, bem como avaliar a possibilidade de encaminhar um determinado fluxo para um classificador específico são experimentos bastante interessantes, que podem impactar positivamente no consumo de recursos ou na efetividade da classificação. Além disso, essas avaliações considerando um tráfego real (relacionado à sua topologia e comportamento) são a continuação natural deste trabalho.

Uma possibilidade elencada durante a proposição da arquitetura é da utilização de vários tipos de classificadores diferentes. As avaliações foram realizadas utilizando um conjunto de classificadores iguais (mesma técnica) com conjuntos de treinamento diferentes. No entanto, como cada classificador funciona independentemente dos outros, é possível utilizar várias técnicas diferentes (efetuando os ajustes necessários) para alcançar uma maior efetividade da colaboração para classificação.

Adicionalmente, a avaliação do módulo de teste de acurácia será realizada. A análise do impacto da variação do limiar utilizado por esse módulo e o tempo de reação são aspectos relevantes que podem variar de acordo com a configuração escolhida.

Em relação ao posicionamento de classificadores, outros algoritmos de otimização podem ser utilizados para encontrar uma solução ideal já que o PICO faz uso do método Subida na Encosta. Algoritmos de otimização mais sofisticados como o *Simulated Annealing,* por exemplo, serão avaliados em trabalhos futuros.

Por fim, serão avaliados os impactos da implementação dos módulos dessa arquitetura em um ambiente real. Nesse caso, o foco estará nas trocas de mensagens entre os diferentes módulos, bem como nas consequências da virtualização e da visão global da rede provida pela SDN.

8.3 Publicações científicas

Diante da importância de contribuições científicas para construção da tese de doutorado, na Tabela 1.22, estão listadas as publicações científicas referentes ao período do doutorado (2012 a 2017).

Os trabalhos foram realizados nesse período e estão indiretamente relacionados ao conteúdo central da tese.

Tabela 1.22 – Lista de publicações científicas no período do doutorado

Nº	Título	Autores	Evento	Ano
1	GPU-Oriented Stream Data Mining Traffic Classification	Lopes Júnior, P. G.; Fernandes, S.; Melo, W. D. B.; Hadj Sadok, D. F.	THE 19th IEEE SYMPOSIUM ON COMPUTERS AND COMMUNICATIONS	2014
2	On the Performance of DPI Signature Matching with	Melo, W. D. B.; Lopes Júnior, P. G.; Fernandes, S.;	THE 19th IEEE SYMPOSIUM ON COMPUTERS AND	2014

	Dynamic Priority	Hadj Sadok, D. F.; Szabo, G.	COMMUNICATIONS	
3	Reordenando Assinaturas em Mecanismos de Inspeção de Pacotes Baseado em Prioridade Dinâmica	Lopes Júnior, P. G.; Melo, W. D. B.; Antonello, R.; Fernandes, S.; Hadj Sadok, D. F.	WPerformance - XIII Workshop em Desempenho de Sistemas Computacionais e de Comunicação	2014
4	Explorando o processamento paralelo na classificação de tráfego em redes de alta velocidade	Santos, A. F.; Lopes Júnior, P. G.; Fernandes, S.; Hadj Sadok, D. F.	31º Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos	2013
5	Multi-gigabit Traffic Identification on GPU	Santos, A. F.; Lopes Júnior, P. G.; Fernandes, S.; Hadj Sadok, D. F.; Szabo, G.	1st Workshop on High Performance and Programmable Networking - The 22nd International ACM Symposium on High-Performance Parallel and Distributed Computing	2013
6	GSDT: Um Algoritmo Para Classificação de Tráfego Baseado em Mineração de Fluxos de Dados	Lopes Júnior, P. G.; Fernandes, S.; Hadj Sadok, D. F.	XVII Workshop de Gerência e Operação de Redes e Serviços	2012

Referências

ABDULSALAM, H.; SKILLICORN, D. B.; MARTIN, P. *Streaming Random Forests*. In: International Database Engineering and Applications Symposium (IDEAS 2007), 11., 2007, Banff, Alta., 2007, p. 225-232.

AGARWAL, S.; KODIALAM, M.; LAKSHMAN, T.V. *Traffic engineering in software defined networks*. In: INFOCOM, 2013, Turin, Proceedings IEEE, 2013, p. 2211-2219.

ALIZADEH, H.; KHOSHROU, A.; ZUQUETE, A. *Traffic classification and verification using unsupervised learning of Gaussian Mixture Models*. In: IEEE International Workshop on Measurements & Networking (M&N), 2015, Coimbra, p. 1-6.

AMARAL, P.; DINIS, J.; PINTO, P.; BERNARDO, L.; TAVARES, J.; MAMEDE, H.. *Machine Learning in Software Defined Networks: Data collection and traffic classification*. In: International Conference on Network Protocols (ICNP), 24., 2016, Singapore, 2016, p. 1-5.

APPENZELLER, G.; KESLASSY, I.; MCKEOWN, N. *Sizing router buffers*. In: SIGCOMM, 2004, Portland, Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications, 2004, vol. 34, n. 4, p. 281-292.

BÄCK, Thomas. *Evolutionary Algorithms in Theory and Practice*: Evolution Strategies, Evolutionary Programming, Genetic Algorithms. Oxford: University Press US, January 1996.

BAROLLI, A.; XHAFA, F.; S´NCHEZ, C.; TAKIZAWA, M. *A Study on the Performance of Search Methods for Mesh Router Nodes Placement Problem*. In: IEEE International Conference on Advanced Information Networking and Applications, 2011, Singapore, 2011, p. 756-763.

BARRIONUEVO, M.; LOPRESTI, M.; MIRANDA, N.; PICCOLI, M. F. Solving a big-data problem with GPU: the network traffic analysis. *Journal of Computer Science & Technology*, vol. 15, n. 1, p. 30-39, Abril 2015.

BERNAILLE, L.; TEIXEIRA, R.; AKODKENOU, I.; SOULE, A.; SALAMATIAN, K. *Traffic classification on the fly*. In: SIGCOMM, 2006, Computer Communication Review, 2006, vol. 36, n. 2, p. 23-26.

BLOOM, Burton H. Space/time trade-offs in hash coding with allowable errors. *Communications of the ACM*, vol. 13, n. 7, p. 422–426, 1970.

BONDAN, L.; DOS SANTOS, C. R. P.; GRANVILLE, L. Z. *Comparing virtualization solutions for NFV deployment: A network management perspective*. In: IEEE Symposium on Computers and Communication (ISCC), Messina, 2016, p. 669-674.

BONFIGLIO, D.; MELLIA, M.; MEO, M.; ROSSI, D.; TOFANELLI, P. *Revealing skype traffic: when randomness plays with you*. In: SIGCOMM, 2007, Conference on Applications, technologies, architectures, and protocols for computer communications, 2007, Nova Iorque, NY, USA, 2007, p. 37-48.

BOTTA, A.; DAINOTTI, A.; PESCAPE, A. Do you trust your software-based traffic generator?. *IEEE Communications Magazine*, vol.48, n. 9, p.158-165, Setembro 2010.

BOUET, M.; LEGUAY, J.; CONAN, V. Cost-based placement of vDPI functions in NFV infrastructures. *International Journal of Network Management*, vol. 25, n. 6, p. 490–506, 2015.

BREMLER-BARR, A.; HARCHOL, Y.; HAY, D.; KORAL, Y. *Deep Packet Inspection as a Service*. In: ACM International on Conference on emerging Networking Experiments and Technologies (CoNEXT '14), 10., 2014, Nova Iorque, NY, USA, 2014, p. 271-282.

BUJLOW, T.; RIAZ, T.; PEDERSEN, J.M. *A method for classification of network traffic based on C5.0 Machine Learning Algorithm*. In: International Conference on Computing, Networking and Communications (ICNC), 2012, Maui, HI, 2012, p. 237-241.

CAI, Z.; COX, A. L.; NG, T. S. E. *Maestro: A system for scalable openflow control.* TSEN Maestro-Technical Report TR10-08, Houston: Rice University, 2010.

CALLADO, A.; KAMIENSKI, C.; SZABO, G.; GERO, B. P.; KELNER, J.; FERNANDES, S.; SADOK, D. A Survey on Internet Traffic Identification. *IEEE Communications Surveys & Tutorials*, vol. 11, n. 3, p. 37-52, 2009.

CALLADO, A.; KELNER, J.; SADOK, D.; KAMIENSKI, C.; FERNANDES, S. Better network traffic identification through the independent combination of techniques. *Journal of Network and Computer Applications*, Londres, vol. 33, n. 4, p. 433-446, 2010.

CANINI, M.; KUZNETSOV, P.; LEVIN, D.; SCHMID, S. *A distributed and robust SDN control plane for transactional network updates*. In: IEEE Conference on Computer Communications (INFOCOM), 2015, Kowloon, 2015, p. 190-198.

CANZIAN, L.; VAN DER SCHAAR, M. Real-time stream mining: online knowledge extraction using classifier networks. *IEEE Network,* vol. 29, n. 5, p. 10-16, 2015.

CARELA-ESPAÑOL, V. An Autonomic Traffic Classification System for Network Operation and Management. *Journal of Network and Systems Management*, Nova Iorque, vol. 23, n. 3, p. 1-19, 2013.

CASADO, M.; FOSTER, N.; GUHA, A. Abstractions for software-defined networks. *Communications of the ACM*, Nova Iorque, vol. 57, n. 10, p. 86 -95, 2014.

CASCARANO, N.; ESTE, A.; GRINGOLI, F.; RISSO, F.; SALGARELLI, L. *An Experimental Evaluation of the Computational Cost of a DPI Traffic Classifier*. In: GLOBECOM, 2009, Honolulu, HI, Global Telecommunications Conference, 2009, p. 1-8.

CHIOSI, M.; CLARKE, D.; WILLIS, P.; REID, A.; FEGER, J.; BUGENHAGEN, M.; KHAN, W.; FARGANO, M.; CUI, C.; DENG, H.; BENITEZ, J. *Network functions virtualisation: An introduction, benefits, enablers, challenges and call for action*. In: SDN and OpenFlow World Congress, 2012, p. 22-24.

CHURCH, R.; VELLE, C. R. The maximal covering location problem. *Papers in regional science*, vol. 32, n. 1, p. 101-118, 1974.

CLAYMAN, S.; MAINI, E.; GALIS, A.; MANZALINI, A.; MAZZOCCA, N. *The dynamic placement of virtual network functions*. IEEE Network Operations and Management Symposium (NOMS), 2014, Cracóvia, Polônia, 2014, p. 1-9.

COATES, M.; HERO III, A. O.; NOWAK, R.; YU, B. Internet tomography. *IEEE Signal Processing Magazine*, vol. 19, n. 3, p. 47–65, Maio 2002.

COOK, Thomas; CAMPBELL, Donald. *Quasi experimentation: Design & analysis issues for field settings*. Boston: Houghton Mifflin, 1979.

COOPER, L. Location-allocation problems. *Operations research*, vol. 11, n. 3, p. 331-343, 1963.

CORMEN, Thomas H.; LEISERSON, Charles E.; RIVEST, Ronald L.; STEIN, Clifford. *Introduction to algorithms*. Cambridge: MIT press, 2001.

COULOURIS, George; DOLLIMORE, Jean; KINDBERG, Tim; BLAIR, Gordon. *Distributed Systems: Concepts and Design*. USA: Addison-Wesley Publishing Company, 2011, 5 ed.

CUI, C.; DENG, H.; TELEKOM, D.; MICHEL, U.; DAMKER, H. *Network Functions Virtualisation*. SDN and OpenFlow World Congress. Darmstadt, Alemanha. White Paper, 2012.

DAINOTTI, A.; PESCAPÉ, A.; CLAFFY, K. Issues and Future directions in Traffic Classification. *IEEE Network*, vol. 26, n. 1, p. 35-40, Janeiro-Fevereiro 2012.

DILLON, A. Design Rationale. *Journal of the American Society for Information Science*, vol. 48, n. 8, p. 762-763, 1997.

DING, W.; QI, W.; WANG, J.; CHEN, B. OpenSCaaS: an open service chain as a service platform toward the integration of SDN and NFV. *IEEE Network*, vol. 29, n. 3, p. 30-35, Maio-Junho 2015.

DORIA, A.; SALIM, J. H.; HAAS, R.; KHOSRAVI, H.; WANG, W.; DONG, L.; GOPAL, R.; HALPERN, J. *Forwarding and Control Element Separation (ForCES) Protocol Specification*. RFC 5810, IETF. Disponível em <http://tools.ietf.org/html/rfc5810>. Acessado em 28 de Janeiro de 2017.

DORIGO, M.; BIRATTARI, M.; STUTZLE, T. Ant colony optimization. *IEEE Computational Intelligence Magazine*, vol. 1, n. 4, p. 28-39, Novembro 2006.

FEAMSTER, N.; REXFORD, J.; ZEGURA, E. The road to SDN. *Queue*, vol. 11, n. 12, p.20:20 -20:40, 2013.

FERNANDES, S.; ANTONELLO, R.; LACERDA, T.; SANTOS, A.; SADOK, D.; WESTHOLM, T. *Slimming Down Deep Packet Inspection Systems*. IEEE INFOCOM Workshops, 2009, Rio de Janeiro, 2009, p. 1-6.

FINAMORE, A.; MELLIA, M.; MEO, M.; MUNAFO, M. M.; DI TORINO, P.; ROSSI, D. Experiences of Internet traffic monitoring with tstat. *IEEE Network*, vol. 25, n. 3, p. 8-14, Maio-Junho 2011.

FINSTERBUSCH, M.; RICHTER, C.; ROCHA, E.; MULLER, J. A.; HANSSGEN, K. A Survey of Payload-Based Traffic Classification Approaches. *IEEE Communications Surveys & Tutorials*, vol. 16, n. 2, p. 1135-1156, 2014.

GAMA, João; RODRIGUES, Pedro. *Data Stream Processing*. In: GAMA, João; Gaber, Mohamed M. *Learning from Data Streams*: Processing Techniques in Sensor Networks. Berlim: Springer Berlin Heidelberg, 2007, cap. 3.

GEMBER-JACOBSON, A.; VISWANATHAN, R.; PRAKASH, C.; GRANDL, R.; KHALID, J.; DAS, S.; AKELLA, A. OpenNF: Enabling in-novation in network function control. *ACM SIGCOMM Computer Communication Review*, vol. 44, n. 4, p. 163-174, 2015.

GOMES, Lalinka de Campos Teixeira. *Inteligência computacional na síntese de meta-heurísticas para otimização combinatória e multimodal.* 2006. Tese (Doutorado em Engenharia Elétrica) Universidade Estadual de Campinas, Campinas, SP.

GRINGOLI, F.; SALGARELLI, L.; DUSI, M.; CASCARANO, N.; RISSO, F. GT: picking up the truth from the ground for internet traffic. *SIGCOMM Computer Communication Review*, vol. 39, n. 5, p. 12-18, Outubro de 2009.

GUAN, Jiwen W.; BELL, D. A. *Evidence theory and its applications*. Nova Iorque: Elsevier Science Inc., 1991.

HAN, B.; GOPALAKRISHNAN, V., JI, L., & LEE, S. Network function virtualization: Challenges and opportunities for innovations. *IEEE Communications Magazine*, vol. 53, n. 2, p. 90-97, Fevereiro de 2015.

HASHIMOTO, Kleber. Técnicas de otimização combinatória multiobjetivo aplicadas na estimação do desempenho elétrico de redes de distribuição. 2004. Tese (Doutorado em Sistemas de Potência) - Escola Politécnica, Universidade de São Paulo, São Paulo, 2004.

HE, L.; XU, C.; LUO, Y. *vTC: Machine Learning Based Traffic Classification as a Virtual Network Function*. In: SDN-NFV Security, 2016, New Orleans, Louisiana, USA, Proceedings of the 2016 ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization, 2016, p. 53-56.

HERRERA, J. G.; Botero, J. F. Resource Allocation in NFV: A Comprehensive Survey. *IEEE Transactions on Network and Service Management*, vol. 13, n. 3, p. 518-532, Sept. 2016.

HO, T. K.; HULL, J. J.; SRIHARI, S. N. Decision combination in multiple classifier systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, n. 1, p. 66-75, Janeiro de 1994.

HOPCROFT, J.; TARJAN, R. Dividing a graph into triconnected components. *SIAM Journal on Computing*, vol. 2, n. 3, p. 135–158, 1973.

HWANG, J.; RAMAKRISHNAN, K. K.; WOOD, T. NetVM: High Performance and Flexible Networking Using Virtualization on Commodity Platforms. *IEEE Transactions on Network and Service Management*, vol. 12, n. 1, p. 34-47, Março de 2015.

IBRAHIM, H. A. H.; AL ZUOBI, O. R. A.; AL-NAMARI, M. A.; MOHAMEDALI, G.; ABDALLA, A. A. A. *Internet traffic classification using machine learning approach: Datasets validation issues*. In: Conference of Basic Sciences and Engineering Studies (SGCAC), 2016, Cartum, Sudão, 2016, p. 158-166.

IGBE, O.; DARWISH, I.; SAADAWI, T. *Distributed Network Intrusion Detection Systems: An Artificial Immune System Approach*. IEEE First International Conference on Connected Health: Applications, Systems and Engineering Technologies (CHASE), 2016, Washington, DC, 2016, p. 101-106.

JAIN, R.; PAUL, S. Network virtualization and software defined networking for cloud computing: a survey. *IEEE Communications Magazine*, vol. 51, n. 11, p. 24-31, Novembro de 2013.

JARSCHEL, M.; WAMSER, F.; HOHN, T.; ZINNER, T.; TRAN-GIA, P. *SDN-Based Application-Aware Networking on the Example of YouTube Video Streaming*. In: European Workshop on Software Defined Networks (EWSDN), 2., 2013, p. 87-92.

JIANG, H.; MOORE, A. W.; GE, Z.; JIN, S.; WANG, J. *Lightweight application classification for network management*. In: Proceedings of the 2007 SIGCOMM workshop on Internet network management, 2007, p. 299-304.

JIANG, N.; GRUENWALD, L. Research issues in data stream association rule mining. *ACM SIGMOD Record*, vol. 35, n. 1, p. 14-19, Março de 2006.

JOSEPH, D. A.; TAVAKOLI, A.; STOICA, I. A policy-aware switching layer for data centers. *ACM SIGCOMM Computer Communication Review*, vol. 38, n. 4, p. 51-62, 2008.

KARAGIANNIS, T.; PAPAGIANNAKI, K.; FALOUTSOS, M. BLINC: multilevel traffic classification in the dark. *SIGCOMM Computer Communication Review*, vol. 35, n. 4, p. 229-240, Agosto de 2005.

KIREMIRE, A. R. The application of the Pareto principle in software engineering. *Consulted January*, vol. 13, 2011.

KOPONEN, T.; CASADO, M.; GUDE, N.; STRIBLING, J.; POUTIEVSKI, L.; ZHU, M.; RAMANATHAN, R.; IWATA, Y.; INOUE, H.; HAMA, T.; SHENKER, S. *Onix*: A distributed control platform for large-scale production networks. In: USENIX, 9., 2010, Vancouver, conference on Operating systems design and implementation, Vancouver: USENIX Association Berkeley, 2010, p.1-6.

KREUTZ, D.; RAMOS, F. M.; VERISSIMO, P. E.; ROTHENBERG, C. E.; AZODOLMOLKY, S.; UHLIG, S. Software-Defined Networking: A Comprehensive Survey. *Proceedings of the IEEE*, vol. 103, n.1, p.14-76, Janeiro de 2015.

KUMAR, A; MALIK, K. Voting Mechanisms in Distributed Systems. *IEEE Transactions on Reliability*, vol 40, n. 5, p. 593-600, 1991.

LANGE, S.; GEBERT, S.; ZINNER, T.; TRAN-GIA, P.; HOCK, D.; JARSCHEL, M.; HOFFMANN, M. Heuristic Approaches to the Controller Placement Problem in Large Scale SDN Networks. *IEEE Transactions on Network and Service Management*, vol. 12, n. 1, p. 4-17, Março de 2015.

LI, B.; SPRINGER, J.; BEBIS, G.; GUNES, M. H. A survey of network flow applications. *Journal of Network and Computer Applications*, vol. 36, n. 2, p. 567-581, 2013.

LI, W.; MOORE, A. W. *A Machine Learning Approach for Efficient Traffic Classification*. 15th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems, Istanbul, 2007, p. 310-317.

LI, Y.; CHEN, M. Software-Defined Network Function Virtualization: A Survey. *IEEE Access*, vol. 3, p. 2542-2553, 2015.

LI, Y.; LI, J. *MultiClassifier: A combination of DPI and ML for application-layer classification in SDN*. IN: International Conference on Systems and Informatics (ICSAI), 2., 2014, p. 682-686.

LI, X.; QIAN, C. *A survey of network function placement*. In: IEEE Annual Consumer Communications & Networking Conference (CCNC), 13., 2016, Las Vegas, NV, 2016, p. 948-953.

LIN, T.; KANG, J. M.; BANNAZADEH, H.; LEON-GARCIA, A. *Enabling SDN applications on Software-Defined Infrastructure*. In: IEEE Network Operations and Management Symposium (NOMS), 2014, Cracóvia, 2014, p. 1-7.

LIN, Y.; LIN, P. C.; YEH, C. H.; WANG, Y. C.; LAI, Y. C. An extended SDN architecture for network function virtualization with a case study on intrusion prevention. *IEEE Network*, vol. 29, n. 3, p. 48-53, Maio-Junho de 2015.

LOPES, P.; FERNANDES, S.; MELO, W.; SADOK, D. H. *GPU-oriented stream data mining traffic classification*. IEEE Symposium on Computers and Communication (ISCC), 2014, p. 1-7.

LOPES, P. *Classificação de tráfego baseado em mineração de fluxos de dados*. 2012. Dissertação (Mestrado em Cência da Computação) – Centro de Informática, Universidade Federal de Pernambuco, Pernambuco.

MA, L.; HE, T.; LEUNG, K. K.; SWAMI, A.; TOWSLEY, D. Inferring Link Metrics From End-To-End Path Measurements: Identifiability and Monitor Placement. *IEEE/ACM Transactions on Networking*, vol. 22, n. 4, p. 1351-1368, Agosto de 2014 (2014a).

MA, L.; HE, T.; LEUNG, K. K.; SWAMI, A.; TOWSLEY, D. *Monitor placement for maximal identifiability in network tomography*. In: Proceedings IEEE INFOCOM, 2014b, p. 1447-1455.

MAIER, G.; FELDMANN, A.; PAXSON, V.; ALLMAN, M. *On dominant characteristics of residential broadband internet traffic*. In: ACM SIGCOMM conference on Internet measurement conference (IMC '09), 9., 2009, New York, NY, USA, 2009, p. 90-102.

MAXWELL, Joseph A. *Qualitative research design:* An interactive approach. 3 ed. Vancouver: Sage Publications Inc., 2013.

MCKEOWN, N. *How SDN will shape networking*. 2011. Disponível em: <https://www.youtube.com/watch?v=c9-K5O_qYgA>. Acessado em 29 de Janeiro de 2017.

MCKEOWN, N.; ANDERSON, T.; BALAKRISHNAN, H.; PARULKAR, G.; PETERSON, L.; REXFORD, J.; SHENKER, S.; TURNER, J. Openflow: enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review*, vol. 38, n. 2, p. 69–74, 2008.

MELO, W.; LOPES, P.; ANTONELLO, R.; FERNANDES, S.; SADOK, D. *On the performance of DPI signature matching with dynamic priority*. In: IEEE Symposium on Computers and Communication (ISCC), 2014, p. 1-6.

MIJUMBI, R.; SERRAT, J.; GORRICHO, J. L.; BOUTEN, N.; DE TURCK, F.; BOUTABA, R. Network Function Virtualization: State-of-the-Art and Research Challenges. *IEEE Communications Surveys & Tutorials*, vol. 18, n. 1, p. 236-262, 2016.

MOHAMMADKHAN, A.; GHAPANI, S.; LIU, G.; ZHANG, W.; RAMAKRISHNAN, K. K.; WOOD, T. *Virtual function placement and traffic steering in flexible and dynamic software defined networks*. IEEE International Workshop on Local and Metropolitan Area Networks, 21., 2015, Pequim, 2015, p. 1-6.

MONTEIRO, F. C.; PRUDENCIO, R. B. C. *Algoritmo Subida da Encosta para Otimização de Parâmetros de Máquinas de Vetores Suporte*. In: Congresso Brasileiro de Inteligência Computacional, 2011, Fortaleza, Anais do Congresso Brasileiro de Inteligência Computacional, 2011.

NG, B.; HAYES, M.; SEAH, W. K. G. *Developing a traffic classification platform for enterprise networks with SDN*: Experiences & lessons learned. In: IFIP Networking Conference (IFIP Networking), 2015, p.1-9.

NGUYEN, T. T. T.; ARMITAGE, G. A survey of techniques for internet traffic classification using machine learning. *IEEE Communications Surveys & Tutorials*, vol. 10, n. 4, p. 56-76, 2008.

NUNES, B.A.A.; MENDONCA, M.; NGUYEN, X. N.; OBRACZKA, K.; TURLETTI, T. A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks. *IEEE Communications Surveys & Tutorials*, vol. 16, n. 3, p. 1617-1634, 2014.

OATES, T.; JENSEN, D. *The Effects of Training Set Size on Decision Tree Complexity*. In: International Conference on Machine Learning, 14., 1997.

OLIVEIRA, A. *Otimização da sequência de montagem de componentes para programação de insersoras automáticas usando métodos heurísticos locais*. 2008. Dissertação (Mestrado em Engenharia Elétrica) - Universidade Federal do Amazonas, Amazonas.

OLIVEIRA, M. R.; NEVES, J.; VALADAS, R.; SALVADOR, P. *Do we need a perfect ground-truth for benchmarking Internet traffic classifiers*. In: IEEE INFOCOM, 2015, p. 2452-2460.

ORIOLA, O.; ADEYEMO, A. B.; ROBERT, A. B. C. Distributed Intrusion Detection System Using P2P Agent Mining Scheme. *African Journal of Computing & ICT*, vol. 5, n. 2, p. 3-10, 2012.

OTOKURA, M.; LEIBNITZ, K.; KOIZUMI, Y.; KOMINAMI, D.; SHIMOKAWA, T.; MURATA, M. *Application of evolutionary mechanism to dynamic Virtual Network Function Placement*. IEEE International Conference on Network Protocols (ICNP), 24., 2016, Singapore, 2016, p. 1-6.

PEARL, J.; RUSSELL, S. Bayesian networks. Report (R-277), 2000, *Handbook of Brain Theory and Neural Networks*, M. Arbib, Cambridge: MIT Press, p. 157–160, 2000.

PEREZ, K. G.; YANG, X.; SCOTT-HAYWARD, S.; SEZER, S. *Optimized Packet Classification for Software-Defined Networking*. In: IEEE International Conference on Communications (ICC), 2014, p. 859-864.

PORRAS, P.A.; NEUMAN, P.G. *EMERALD*: Event monitoring enabling responses to anomalous live disturbances. In: National Information Systems Security Conference (NISTNCSC), 20., 1997, p. 353–365.

PRESSMAN, Roger. *Engenharia de Software*: Uma Abordagem Profissional. 7ª ed., McGraw Hill Companies, NY, EUA, 2011.

QAZI, Z. A.; LEE, J.; JIN, T.; BELLALA, G.; ARNDT, M.; NOUBIR, G. Application-awareness in SDN. *ACM SIGCOMM Computer Communication Review*, vol. 43, n. 4, p. 487-488, 2013.

QUINLAN, J. Ross. *C4.5*: Programs for Machine Learning. São Francisco: Morgan Kaufmann Publishers, 1993.

QUINLAN, J. Ross. *Data mining tools See5 and C5.0*. 2004. Disponível em: <http://www.rulequest.com/see5-info.html>. Acessado em: 29 de Janeiro de 2017.

RANTALA, V.; LEHTONEN, T.; LILJEBERG, P.; PLOSILA, J. *Distributed Traffic Monitoring Methods for Adaptive Network-on-Chip*. NORCHIP, 2008, p. 233-236.

READ, Ronald C. *Graph theory and computing*. Academic Press, 2014.

RICHTER, P.; CHATZIS, N.; SMARAGDAKIS, G.; FELDMANN, A.; WILLINGER, W. *Distilling the Internet's Application Mix from Packet-Sampled Traffic*. In: International Conference on Passive and Active Network Measurement, 2015, p. 179-192.

RIZZI, A.; IACOVAZZI, A.; BAIOCCHI, A.; COLABRESE, S. A low complexity real-time Internet traffic flows neuro-fuzzy classifier. *Computer Networks*, vol. 91, p. 752-771, 2015.

RUSSELL, Stuart J.; NORVIG, Peter. *Artificial intelligence: a modern approach*. 2 ed., Upper Saddle River: Prentice hall, 2003.

SANTOS, A.; FERNANDES, S.; LOPES, P.; SADOK, D.; SZABO, G. *Multi-gigabit traffic identification on GPU*. In: High performance and programmable networking (HPPN '13), 2013, New York, NY, USA, 2013, p. 39-44.

SCHUSTER, F.; PAUL, A. *A distributed intrusion detection system for industrial automation networks*. In: IEEE Conference on Emerging Technologies & Factory Automation (ETFA), 17., 2012, p.1-4.

SIMMROSS-WATTENBERG, F.; ASENSIO-PEREZ, J. I.; CASASECA-DE-LA-HIGUERA, P.; MARTIN-FERNANDEZ, M.; DIMITRIADIS, I. A.; ALBEROLA-LOPEZ, C. Anomaly Detection in Network Traffic Based on Statistical Inference and alpha-Stable Modeling. *IEEE Transactions on Dependable and Secure Computing*, vol. 8, n. 4, p. 494-509, Julho-Agosto 2011.

SIPSER, Michael. *Introdução à Teoria da Computação*. 2007. Boston, MA: Thomson Course Technology.

SMITH, R.; GOYAL, N.; ORMONT, J.; SANKARALINGAM, K.; ESTAN, C. *Evaluating GPUs for Network Packet Signature Matching*. In: International Symposium on Performance Analysis of Systems and Software (ISPASS'09), 2009, Boston, MA, 2009, p. 175-184.

SOYSAL, M.; SCHMIDT, E. G. Machine learning algorithms for accurate flow-based network traffic classification: Evaluation and comparison. *Performance Evaluation*, vol. 67, n. 6, p. 451-467, Junho de 2010.

SPALL, James C. *Introduction to Stochastic Search and Optimization*. Estimation, Simulation, and Control. 1 ed., Wiley-Interscience Series in Discrete Mathematics and Optimization. John Wiley & Sons, 2003.

SHU, Z.; WAN, J.; LIN, J.; WANG, S.; LI, D.; RHO, S.; YANG, C. Traffic engineering in software-defined networking: Measurement and management. In: *IEEE Access*, vol. 4, p. 3246-3256, 2016.

SUTHAHARAN, S. Big data classification: Problems and challenges in network intrusion prediction with machine learning. *ACM SIGMETRICS Performance Evaluation Review*, vol. 41, n. 4, p. 70-73, Abril de 2014.

SZABÓ, G.; GÓDOR, I.; VERES, A.; MALOMSOKY, S.; MOLNÁR, S. Traffic classification over Gbit speed with commodity hardware. *IEEE Journal Communications Software and Systems*, vol. 5, n. 3, 2010.

TARJAN, R. *Depth-first search and linear graph algorithms*. In: 12th Annual Symposium on Switching and Automata Theory, 1971, East Lansing, MI, USA, 1971, p. 114-121.

THAJCHAYAPONG, S.; GARCIA-TREVINO, E. S.; BARRIA, J. A. Distributed Classification of Traffic Anomalies Using Microscopic Traffic Variables. *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, n. 1, p. 448-458, Março de 2013.

TOOTOONCHIAN, A.; GANJALI, Y. *Hyperflow*: A distributed control plane for openflow. In: Internet network management conference on Research on enterprise networks, 2010, USENIX Association, 2010, p. 3–3.

TUMER, K.; GHOSH, J. Error correlation and error reduction in ensemble classifiers. *Connection science*, vol. 8, n. 3-4, p. 385-404, 1996.

TURNBULL, B.; OSBORNE, G.; SIMON, M. *Legal and Technical Implications of Collecting Wireless Data as an Evidence Source Forensics*. In: International Conference on Forensics in Telecommunications, Information and Multimedia, 2., 2009, Adelaide, Austrália, 2009, p. 36-41.

VOELLMY, A.; WANG, J. *Scalable software defined network controllers*. ACM SIGCOMM, 2012, New York, NY, USA, Conference on Applications, technologies, architectures, and protocols for computer communication, 2012, p. 289–290.

WANG, Z.; TSOU, T.; HUANG, J.; SHI, X.; YIN, X. *Analysis of Comparisons between OpenFlow and ForCES*. IETF, Março de 2012.

WANG, Y.; XIANG, Y.; ZHANG, J.; ZHOU, W.; WEI, G.; YANG, L. T. Internet Traffic Classification Using Constrained Clustering. *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, n. 11, p. 2932-2943, Novembro de 2014.

WEISE, Thomas. *Global optimization algorithms-theory and application*. 2 ed., Publicação própria, 2009.

WETTE, P.; KARL, H. DCT2Gen: A traffic generator for data centers. Computer Communications, vol. 80, p. 45-58, Abril de 2016.

WILLIAMS, N.; ZANDER, S.; ARMITAGE, G. A preliminary performance comparison of five machine learning algorithms for practical IP traffic flow classification. *ACM SIGCOMM Computer Communication Review*, vol. 36, n. 5, p. 5-16, Outubro de 2006.

WITTEN, Ian; FRANK, Eibe; HALL, Mark A.; PAL, Christopher J. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. 2 ed., Morgan Kaufmann Publishers, 2005.

WOLF, Gert W. *Facility location: concepts, models, algorithms and case studies. Series: Contributions to Management Science.* Editores: ZANJIRANI FARAHANI, REZA AND HEKMATFAR, MASOUD, Alemanha, Physica-Verlag, 2009.

XU, Y.; LIU, Z.; ZHANG, Z.; CHAO, H. J. High-throughput and memory-efficient multimatch packet classification based on distributed and pipelined hash tables. *IEEE/ACM Transactions on Networking* (TON), vol. 22, n. 3, p. 982-995, 2014.

YANG, W.; FUNG, C. *A survey on security in network functions virtualization*. In: IEEE NetSoft Conference and Workshops (NetSoft), 2016, Seoul, 2016, p. 15-19.

YU, Y.; QIAN, C.; LI, X. *Distributed and collaborative traffic monitoring in software defined networks*. In: Workshop Hot Topics Software Defined Networking, 3., 2014, Chicago, Illinois, USA, 2014, p. 85 -90.

YUAN, Z.; WANG, C. *An improved network traffic classification algorithm based on Hadoop decision tree*. IEEE International Conference of Online Analysis and Computing Science (ICOACS), 2016, Chongqing, 2016, p. 53-56.

ZHANG, J.; CHEN, X.; XIANG, Y.; ZHOU, W.; WU, J. Robust Network Traffic Classification. *IEEE/ACM Transactions on Networking*, vol. 23, n. 4, p. 1257-1270, Agosto de 2015.