UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Aprendizagem Profunda Aplicada a Telecomunicações: Classificação de Modulação e Controle de Congestionamento

Ingrid Ariel Silva Nascimento

DM: 18/2019

UFPA / ITEC / PPGEE
Campus Universitário do Guamá
Belém-Pará-Brasil
2019

UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Ingrid Ariel Silva Nascimento

Aprendizagem Profunda Aplicada a Telecomunicações: Classificação de Modulação e Controle de Congestionamento

DM: 18/2019

UFPA / ITEC / PPGEE
Campus Universitário do Guamá
Belém-Pará-Brasil
2019

UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Ingrid Ariel Silva Nascimento

Aprendizagem Profunda Aplicada a Telecomunicações: Classificação de Modulação e Controle de Congestionamento

Dissertação submetida a Banca Examinadora do Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Pará para a obtenção do Grau de Mestre em Engenharia Elétrica na área de Telecomunicações.

UFPA / ITEC / PPGEE
Campus Universitário do Guamá
Belém-Pará-Brasil
2019

Dados Internacionais de Catalogação - na – Publicação (CIP) Sistema de Bibliotecas da UFPA

N244a Nascimento, Ingrid Ariel Silva, 1991-

Aprendizagem profunda aplicada a telecomunicações: Classificação de modulação e controle de congestionamento / Ingrid Ariel Silva Nascimento.-2019.

Orientador: Aldebaro Barreto da Rocha Klautau Júnior
Dissertação (Mestrado) - Universidade Federal do Pará, Instituto de Tecnologia, Programa de Pós-Graduação em Engenharia Elétrica, Belém, 2019.

1. Redes de radio cognitivo. 2. Modulação digital. 3. Redes de computação – Protocolos. 4. Telecomunicações – Tráfego. I. Título.

CDD 23. ed. 621.384

Elaborada por Lucicléa S. de Oliveira – CRB -2/648

Aprendizagem Profunda Aplicada a Telecomunicações:
Classificação de Modulação e Controle de Congestionamento

Dissertação de mestrado submetida a avaliação da banca examinadora aprovada pelo colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal do Pará e julgada adequada para obtenção do Grau de Mestre em Engenharia Elétrica na área de Telecomunicações.

Aprovada em _____ / _____ / _____

_____
Prof. Dr. Aldebaro Barreto da Rocha Klautau Junior
ORIENTADOR

_____
Prof. Dr. Antônio Jorge Gomes Abelém
MEMBRO DA BANCA EXAMINADORA

_____
Profa. Dra. Jasmine Priscyla Leite de Araújo
MEMBRO DA BANCA EXAMINADORA

_____
Profa. Dra. Maria Emília de Lima Tostes
DIRETOR DA PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Agradecimentos

Primeiramente gostaria de agradecer a Deus, por ser aquele que me concedeu a vida, a força nos momentos difíceis da minha jornada acadêmica, a capacitação necessária para vencer os novos desafios e todas as novas oportunidades.

Em segundo lugar, ao Prof.Aldebaro Klautau, por ser meu orientador neste trabalho, me concedendo todo o apoio necessário para a execução desta pesquisa, compartilhando seus preciosos conhecimentos comigo, além da oportunidade de compor o time do LASSE, um dos melhores núcleos de pesquisa científica da UFPA.

Gostaria também de demonstrar minha gratidão a todos os amigos que me apoiaram no período de constituição deste trabalho, em especial ao meu amigo Andrey Silva, por sua co-orientação e todo auxílio prestado no desenvolvimento deste trabalho, e ao amigo Flávio Mendes, por todo o suporte e assistência oferecida.

E por fim, gostaria de expressar todo o meu amor e gratidão aos meus pais, José Antonio Nascimento, por ser o meu maior exemplo de força, fé e perseverança, e à minha mãe, Rosemary Nascimento, por todo o carinho, apoio e cuidado oferecidos por toda a minha vida. Graças aos sacrifícios deles, sempre posso planejar vôos mais altos. Ao meu amado irmão Yuri, por todas as palavras de apoio e motivação concedidas, e ao meu querido noivo, Thiago Cavalcante, por seu amor, auxílio e amparo providos.

Ingrid Ariel Silva Nascimento

Maio 2019

Lista de Símbolos

A3C Asynchronous Advantage Actor-Critic

BBU Baseband Unit

C-RAN Centralized Radio Access Network

CAM Classificação Automática de Modulação

CNN Convolutional Neural Network

CPRI Common Public Radio Interface

CTAR Classificação de Tecnologia de Acesso de Rádio

DL Deep Learning

DQN Deep Q-Network

DRL Deep Reinforcement Learning

DT Decision Tree

FH Fronthaul

GSM Global System for Mobile Communications

LTE Long Term Evolution

NB Naive Bayes

NR New Reno

RE Radio Equipment Control

RE Radio Equipment

RF Random Forest

RRH Remote Radio Head

RRU Remote Radio Unit

SNR Signal to Noise Ratio

SVM Support Vector Machines

UE User Equipment

UL Uplink

Lista de Figuras

Lista de Tabelas

Conteúdo

Abstract

The goal of this dissertation is to explore Deep Learning (DL) techniques applied to Telecommunications. DL has achieved success in areas such as computer vision and object detection and it is timely to investigate DL in communication problems. Then, two distinct problems are investigated. First, DL is applied to Automatic Modulation Classification (AMC) to detect the adopted modulation scheme automatically. AMC is important, for instance, in Cognitive Radios and military applications. In this dissertation, we discuss the benefits and drawbacks of using DL as an alternative for AMC and show its efficiency in comparison to other machine learning methods applied to AMC. Other DL application in communication is Congestion Control. The context is related to Fronthaul in C-RAN architecture using congestion control in order to attend the strict requirements of the 5G system. Specifically, DL is investigated in conjuction with Deep Reinforcement Learning (DRL) techniques. In this topic, this dissertation presents a framework for investigations in congestion control for Fronthaul, and the implementation of a model and environment using NS-3 and Gym API of the OpenAI group for simulation. The developed framework is validated with preliminary experiments that compare Deep Reinforcement Learning methods with traditional congestion control techniques, using as figures of merit throughput and latency.

Resumo

O objetivo deste trabalho é explorar técnicas de *Deep Learning* (DL) aplicada a Telecomunicações. DL obtém sucesso comprovado em áreas como Visão Computacional e Detecção de Objetos. Torna-se portanto importante investigar a sua aplicação em problemas na área da Comunicação. Sendo assim, dois problemas distintos são examinados. No primeiro deles, DL é aplicada à Classificação de Modulação (CAM), onde o problema é detectar de forma automática o esquema de modulação adotado. CAM é importante, por exemplo, em Rádios Cognitivos e aplicações militares. Neste trabalho são discutidos benefícios e desvantagens de usar DL como alternativa para CAM, em especial comparando-se a eficiência de DL em relação a outros métodos de Aprendizado de Máquina. A outra aplicação de DL em comunicações é no Controle de Congestionamento. O contexto é o de *Fronthaul* em arquiteturas C-RAN, com o controle de congestionamento usado em prol do atendimento de requisitos estritos impostos à rede 5G. Mais especificamente, investiga-se a aplicação de DL juntamente com técnicas de *Reinforcement Learning* (RL). Neste tópico, a dissertação apresenta um arcabouço para investigações de controle de congestionamento em *fronthaul*, e a implementação de um modelo e respectivo ambiente utilizando NS-3 e a API Gym do grupo OpenAI para simulações. O arcabouço desenvolvido é validado com experimentos preliminares comparando métodos de *Deep Reinforcement Learning* (DRL) e métodos tradicionais de Controle de Congestionamento, utilizando-se índices como vazão e latência.

Capítulo 1

Introdução

Esta dissertação versa sobre aprendizagem profunda aplicada a comunicações. Além da Classificação Automática de Modulação (CAM), a outra aplicação de aprendizagem profunda investigada nesta dissertação é o controle de congestionamento do *fronthaul* no contexto dos sistemas 5G. Sendo o controle de congestionamento uma aplicação totalmente distinta de CAM, desejou-se investigar os possíveis benefícios da aprendizagem profunda neste contexto. Outra motivação é investigar nesta dissertação, ferramentas de software para estudo do controle de congestionamento usando simuladores como o NS-3. Assim, a dissertação apresenta, além de CAM, um arcabouço para uso de aprendizagem profunda em controle de congestionamento. Este capítulo apresenta inicialmente conceitos de CAM, e posteriormente o mesmo aborda controle de congestionamento.

1.1 Classificação Automática de Modulação (CAM)

A classificação automática de modulação (CAM) pode ser definida como o processo de identificação da modulação utilizada em ondas de rádio desconhecidas. Esta é comumente empregada em equipamentos de vigilância de espectro, no caso de aplicações militares e em sistemas de rádio cognitivos, para evitar interferência entre usuários [1]. Além disso, esta também é adotada em rádio cognitivo para otimização de dados de transmissão e melhoras na alocação de espectros [2]. A seguir será visto a importância de CAM e sua aplicabilidade.

1.1.1 Importância de Classificação Automática de Modulação (CAM)

Em função principalmente das necessidades militares após a Segunda Guerra Mundial, houve um grande interesse no desenvolvimento de equipamentos capazes de identificar a informação contida em sinais gerados pelo inimigo, independentemente da modulação usada. Para isso, o primeiro passo seria o desenvolvimento de um decodificador universal para distinguir entre os tipos de modulação com o mínimo de conhecimento prévio disponível [3]. Neste contexto, os estudos acerca da distinção entre os sinais foram intensificados.

A técnica de reconhecimento relacionada à CAM é a que distingue entre dois ou mais padrões de comunicação conhecida como Tecnologia de Acesso de Rádio (TAR). Uma das aplicações de TAR está relacionada à detecção de espectro, que visa identificar os usuários primários licenciados de usuários secundários a fim de evitar interferência [4]. Classificação de TAR (CTAR) é também utilizada para detectar diferentes sistemas *wireless* com esquema baseado em estimativa de máxima verossimilhança [5].

Outros trabalhos de classificação de TAR distinguiram entre sinais LTE e tecnologia Wi-Max através da análise de fatores cicloestacionários [6], além do que foi feito em [7] onde aplicou-se testes estatísticos baseados na distribuição de probabilidade das formas de onda para diferenciar sinais LTE e GSM.

Geralmente o processo de CAM ocorre de duas maneiras: (1) Através da construção de um modelo estatístico que descreve os sistemas de comunicação, sendo empregada a detecção de modulação pelo critério de máxima verossimilhança ou (2) através da extração de característica dos dados usando métodos de aprendizado de máquina. A dificuldade relacionada à primeira forma de aplicar CAM está relacionada a discrepância existente entre o modelo estabelecido e o verdadeiro sistema, além da complexidade computacional encontrada [8].

Sendo assim, métodos de aprendizado de máquina começaram a ser empregados, como em [9] onde árvores de decisão que utilizam momentos cíclicos no domínio do tempo das formas de onda dos sinais são usados para classificação. Além disso, Redes Neurais Profundas também são aplicadas devido a sua melhor desempenho em relação a outros métodos [8]. Outros algoritmos tradicionais são empregados, como *Support Vector Machines* (SVM) em que histogramas da frequência instantânea são usadas como dados de entrada para a classificação de modulações analógicas. Além de classificadores como *K-Nearest Network* (KNN) utilizado conjuntamente com SVM para a classificação de sinais com diferentes SNRs [10].

Devido ao surgimento e sucesso das técnicas de *Deep Learning* aplicadas às mais diversas

áreas, também iniciou-se experimentos acerca de sua efetividade em CAM, já que a classificação de modulação continua sendo um tópico importante atualmente. Sendo assim, as aplicações mais recentes de DL para CAM serão exploradas de forma mais detalhada na seção seguinte.

1.1.2 *Deep Learning* e a Classificação Automática de Modulação

Em CAM, técnicas de DL se destacam em relação aos demais métodos de aprendizado de máquina por sua aplicabilidade em tipos de dados diversos, como por exemplo em [11], onde uma CNN (do inglês, *Convolutional Neural Network*) foi empregada para classificar oito modulações digitais (BPSK, QPSK, 8PSK, 16QAM, 64QAM, BFSK, CPFSK, e 4PAM) e duas modulações analógicas (WB-FM e AM-DSB). Em trabalho posterior [12], os autores verificaram que uma rede CLDNN (do inglês, *Convolutional Long Short-term Deep Neural Network*) alcança maior desempenho em relação à rede CNN adotada em [11].

Há também trabalhos mais recentes que utilizam imagens como entrada para a CNN a fim de classificar os diferentes tipos de modulação [13]. Além de aplicações em sinais de rádio que estão utilizando DL para o reconhecimento dos tipos de modulações intra-pulso [14]. Outros trabalhos utilizam CNN para a detecção de modulação de sinais de co-canais devido a sua habilidade em extrair informações de dados brutos, diferentemente do que é utilizado nos métodos convencionais de CAM, onde estatísticas de alta ordem e cíclicas são utilizadas [15].

Métodos propostos utilizam DL para a extração de características de longas sequências de taxa de símbolos ao longo de diferentes valores de SNR, melhorando a velocidade de convergência com a aplicação de CNN para CAM [16]. Além de CNN, há também a aplicação de *Autoencoders* para a classificação de modulação com a introdução de dados de entrada, sem valor negativo no processo de treinamento, o que gerou resultados melhores em comparação a métodos que utilizam *sparse autoencoder* [17].

Também foram propostas alternativas para realizar CAM quando há desvanecimento de canal (*channel fading*), através da conversão de símbolos IQ para a forma polar a serem utilizadas como dados de entrada de arquiteturas profundas. Este estudo demonstrou uma melhora na acurácia de 5 por cento, e redução de *overhead* de treinamento de 48 por cento [18]. CAM também é aplicada com DL em pesquisas que envolvam sinais acústicos submarinos, como é feito em [19], em que a CNN é empregada em dados no domínio do tempo, enquanto que uma rede LSTM (do inglês, *Long Short-term Memory network*) aprende em fase e quadratura, demonstrando alta taxa de reconhecimento em sinais de baixa SNR.

Métodos mais recentes como *Constrative Fully Convolutional Network* (CFCN) utilizam como entrada uma matriz de constelação extraída de sinais recebidos usando um pré-processamento de baixa complexidade. Uma função-perda é utilizada para o treinamento, o que contrasta a diferença entre as modulações [20]. Em [21] é proposto um novo modelo para sistemas de detecção *wireless*, em que o aprendizado é feito a partir dos sinais no domínio do tempo e fase, sem a necessidade de momentos cíclicos de alta ordem. Uma rede LSTM compõe o modelo para a classificação de modulação, o qual alcançou uma acurácia de mais de 75 por cento em dados de teste variando a SNR de 0 a 20 dB.

Como pode ser visto, CAM continua sendo um tópico de pesquisas com bastante relevância, sendo agora melhor explorado com a aplicação de técnicas de DL para os mais diversos cenários. Sua eficácia foi comprovada através dos trabalhos recentes em comparação aos métodos tradicionais, e esta dissertação buscou investigar aspectos que não estavam claros a partir da literatura.

A seguir, comuta-se da aplicação de aprendizagem profunda em CAM, para controle de congestionamento em fronthaul.

1.2 Controle de Congestionamento para Fronthaul 5G

1.2.1 *Centralized-based* RAN e desafios para o fronthaul

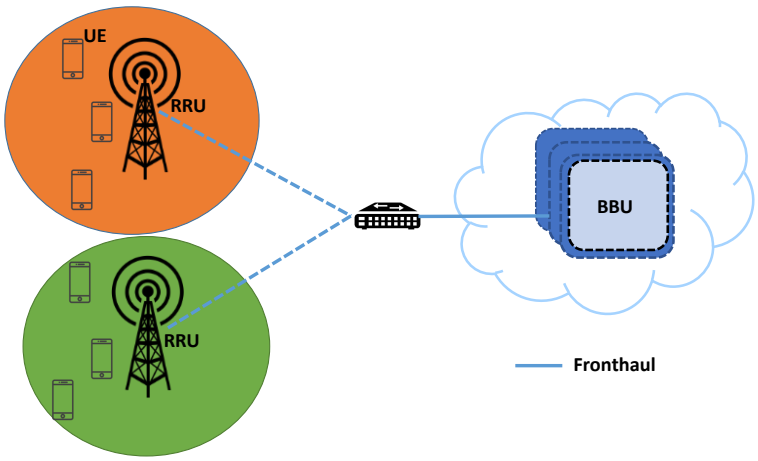
O crescente número de usuários de redes móveis com aplicações que geram cada vez mais tráfego de dados, tem exigido um aumento significativo em largura de banda de sistemas 5G. E diversas aplicações (em especial com dispositivos IoT) exigem mínima latência, sendo este um grande desafio para os sistemas 5G. De forma a atender estes requisitos, uma nova arquitetura de rede foi proposta a qual concentra a *Remote Radio Head* (RRH), em células menores, diminuindo assim a distância entre as unidades e aumentando a taxa de dados.

Na arquitetura C-RAN [22], o processamento básico dos sinais é realizado no *Radio Equipment* (RE) através das RRHs, e o processamento de sinais banda base na seção *Radio Equipment Control* (REC), por meio da BBU. A interligação entre RRH e BBU é realizada através da rede *Fronthaul* que é regida por protocolos de transporte como CPRI [23], eCPRI [24] e outros.

O tráfego de dados do usuário entre RRHs e a BBU é realizada através de pacotes, em uma topologia de rede mais complexa, que agrega todos os *enlaces* em um único canal sendo

esta a rede *Fronthaul* (FH) conforme Figura 1.1, a qual demonstra a topologia *Dumbbell*.

Figura 1.1: Topologia Dumbbell.



Fonte: Autor.

Esta topologia, assim como outras, suscita a disputa de dados das diferentes estações-base pelo mesmo canal e exige o controle de congestionamento, para evitar a perda de pacotes e assegurar a transmissão de cada um dos fluxos gerados pelas RRHs.

Os algoritmos tradicionais de controle de congestionamento encontrados na literatura apresentam regras fixas de operação que, geralmente, não são aplicados a cenários realísticos ou facilmente adaptáveis a mudanças na rede [25]. Por esta razão, novas metodologias estão sendo propostas, as quais utilizam as funcionalidades das técnicas de aprendizado por reforço para cenários de rede, em especial aquelas cujo comportamento estatístico não varia bruscamente com o tempo (podem ser consideradas como estacionárias ao longo de um certo intervalo).

Com o advento do *Deep Learning* (DL) aliado aos métodos de *Reinforcement Learning* (RL) torna-se possível o controle de congestionamento através de respostas imediatas à mudanças do ambiente de rede, sem a necessidade de longo processo de treinamento. Este fato motiva um dos objetivos desta dissertação: gerar um arcabouço que viabilize futuras pesquisas sobre a aplicação de técnicas de *Deep Reinforcement Learning* (DRL) ao controle de congestionamento em *fronthaul* (FH) de arquitetura C-RAN.

A arquitetura C- RAN é diferenciada em relação às gerações anteriores de sistemas móveis não apenas pela maior distância implementada entre RRH e BBU, mas também devido a atribuição de alguns processamentos de banda base à RRH, o que faz com que estas também sejam chamadas de *Radio Units* (RU) ou *Remote Radio Units*(RRU). A interligação entre as unidades de banda base e as RUs são feitas através de fibras óticas de alta velocidade o que caracteriza a rede FH.

Os requerimentos de altíssima velocidade e baixa latência de FH, deve-se ao enorme crescimento de aparelhos móveis, o que impulsionou um aumento exponencial no tráfego de dados sendo estes estimados na escala dos *zetabytes* [26]. Sendo assim, protocolos de transporte como o CPRI (*Common Public Radio Interface*) devem atender requerimentos que exigem ACK e NACK em menos de 3 milisegundos dentre RE e REC, o que garante um *delay* entre 100 e 200 micro segundos para a rede FH [27].

Além disso, aplicações de vídeo em alta definição adicionados à demanda crescente de usuários implicam que redes 5G necessitam suportar um aumento de até 1000 vezes no tráfego total de banda larga até 2020 [28] [29]. Com isso, é esperado um acréscimo de tráfego de até 100 vezes por usuário, sendo o número de usuários até 10 vezes maior [29].

De forma a diminuir os requerimentos sobre as redes FH, técnicas de compressão da taxa de dados foram propostas [30] [31], além da divisão de funcionalidades entre as estações-base e a BBU, flexibilizando assim o fluxo de dados a serem transmitidos pelo FH, assim como gerando a diminuição dos custos de implantação geral da rede [31].

A maneira como as informações serão transmitidas na rede FH também tem sido tema de grande pesquisa. Em semelhança as redes Ethernet, os dados serão transportados através de pacotes entre RRH e BBU, gerando novos cuidados em relação à latência e *overhead* adicionados pelo *frame* Ethernet [32]. Além disso, em uma topologia de rede como mostrada na Figura 1.1, a forma como os pacotes serão roteados e o índice de *fairness* na transmissão entre todas as RRH's, em um dado período de tempo, também são fatores estudados com a implantação do conceito de pacotes como unidades de propagação de dados.

De modo a garantir o escalonamento de pacotes e a utilização equitativa dos recursos da rede entre todas as RRH's, propostas que utilizam sistemas inteligentes tem ganhado espaço através da implementação de algoritmos de controle, os quais irão orquestrar a transmissão de pacotes e realizar o controle de congestionamento através do aprendizado ao longo do tempo. Para isso, o DL aliado aos métodos de RL estão sendo apresentados, o que será discutido na

Seção 1.2.2 a seguir.

1.2.2 *Deep Reinforcement Learning* e o Controle de Congestionamento

Os algoritmos tradicionais de controle de congestionamento em redes TCP geralmente utilizam o conceito de janela de congestionamento (*Cwnd*) para gerenciar o fluxo de pacotes nos enlaces. Estes métodos utilizam regras específicas que ditam como inicializar a *Cwnd* e ajustá-las ao sinal de congestionamento. Um exemplo, é um dos algoritmos mais utilizados ainda hoje, o TCP Reno, o qual aplica a idéia de AIMD (*Additive Increase Multiplicative Decrease*), em que a *Cwnd* é inicializada de forma lenta, sendo acrescida a cada ACK recebido pelo transmissor, na fase posterior, tem o seu crescimento desacelerado como uma forma de evitar o congestionamento.

Outras propostas utilizam medidas da rede para indicar o congestionamento, como o algoritmo TCP Vegas, que utiliza o *Round Trip Time* (RTT) para apontar o condicionamento da rede. Neste caso, a diferença entre o *throughput* esperado e o *throughput* atual da rede são medidos a cada RTT para que se saiba a quantidade de pacotes que estão sendo transmitidos. Caso essa diferença ultrapasse um valor pré-especificado, o tamanho da janela é reduzido antes que ocorra a perda de pacotes indicando o congestionamento. Devido a esta característica, trabalhos anteriores mostraram que o TCP Vegas alcançou maior performance e fluxo de dados que algoritmos tradicionais como o TCP Reno [33].

Novas versões de algoritmos surgiram a fim de solucionar problemas específicos, como o TCP SACK (*Selective Acknowledgement*), que informa ao transmissor os segmentos que chegaram e estão em fila no receptor. Dessa forma, o transmissor pode reenviar os pacotes que ainda não foram recebidos, reduzindo assim a ocorrência de *timeouts*.

Modificações dos algoritmos mais famosos foram lançadas, como o TCP New Reno que implementa algoritmos de retransmissão e recuperação rápida (*fast-retransmit fast-recovery*), podendo assim recuperar segmentos perdidos em uma única janela [34]. Assim como, TCP New Vegas surgiu para solucionar problemas de performance quando o nível de latência é maior que 50 milissegundos e a janela inicial é maior que 2 segmentos [33].

Apesar da eficácia comprovada em diversas aplicações, estes algoritmos apresentam algumas limitações em relação ao ajuste de suas métricas à dinamicidade da rede [25]. Além disso, estes algoritmos não possuem "memória", sendo mais suscetíveis a erros, diante de condições da rede já experimentadas.

Devido a estas características, trabalhos recentes tem aliado as vantagens do RL ao controle de congestionamento devido a sua capacidade em tomar decisões sem conhecimento prévio dos atributos da rede. Além disso, diferentemente dos demais algoritmos de aprendizado de máquina, algoritmos de RL não necessitam de longo processo de treinamento *offline* para aprender de experiências passadas. No contexto do controle de congestionamento, estudos acerca da concepção de protocolos que atendam as características inconstantes da rede, através do aprendizado, tem sido tema de grande atenção [35].

Em [25] um novo algoritmo é proposto, o qual utiliza um método clássico de RL juntamente com algoritmo de otimização para transpor a barreira dos múltiplos estados possíveis encontrados no contexto do controle de congestionamento. Além deste, o *software Remy* [36] foi proposto a fim de gerar algoritmos de controle de congestionamento aplicando conceitos inerentes ao RL. As comparações com algoritmos clássicos revelaram resultados muito satisfatórios em relação a altas taxas de *throughput* e baixo atraso na fila do receptor.

De forma a obter características da rede mais realistas, em [37] o agente reúne dados sobre o estado de congestionamento e aplica *Deep Convolutional Neural Networks* (DCNN) juntamente com uma rede *Long Short Term Memory* (LSTM) para a escolha da ação mais adequada. Este trabalho mostrou resultados competitivos quanto ao índice de *fairness* entre os usuários e o nível de vazão alcançado. Adicionalmente, o método Monte Carlo está sendo aplicado a problemas de roteamento para os enlaces da rede FH em arquitetura C-RAN de forma a atender os requisitos de baixa latência [38].

Este tipo de abordagem utilizando DRL para gerar o controle de congestionamento, ainda é muito recente em relação às aplicações tanto para controle de congestionamento em redes como a Internet, como para atender as exigências impostas para as redes FH. Neste sentido, estas técnicas que já são aplicadas em áreas como visão computacional e detecção de objetos [39], sistema de jogos[40], dentre outros; possuem um grande potencial devido ao ótimo desempenho alcançado, por isso devem ser exploradas em novos contextos, tendo o potencial de trazer soluções eficientes aos rígidos requerimentos de latência e vazão impostas sobre as redes FH.

1.3 Síntese da Dissertação

Como descrito, trabalhos recentes têm comprovado a eficácia das técnicas de DL com o aumento de acurácia na classificação de modulação, além de motivar a diminuição da complexidade dos dados de entrada para o classificador, já que as técnicas de DL mantém sua eficiência mesmo com dados de entradas sem pré-processamento.

Apesar de haver vários trabalhos de DL em CAM, como descrito, há uma fragilidade dos mesmos no tocante às bases de dados utilizadas devido a baixa variabilidade dos canais de propagação que as constituem. Daí esta dissertação se preocupou com este aspecto. O desenvolvimento de dados públicos para experimentos na área de telecomunicações é de suma importância, dada a necessidade de bases de dados para algoritmos que fazem a identificação de diferentes padrões de comunicação.

Após a constituição da base de dados pública, algoritmos tradicionais de aprendizado de máquina e DL foram submetidos a diferentes configurações de treino e teste para avaliar fatores importantes como: performance, custo computacional e acurácia das técnicas no processo de CAM.

Não de menos importância do que CAM na academia, o problema de controle de congestionamento é também discutido nesta dissertação. Não faz parte do escopo da mesma atingir uma comparação sistemática e conclusiva entre métodos baseados em DL e os de controle de congestionamento convencional, mas sim o desenvolvimento de um arcabouço que viabilizará pesquisas neste sentido.

Os tópicos a seguir sumarizam a organização deste trabalho e a suas principais contribuições. Optou-se por apresentar o RL no âmbito de controle de congestionamento, ao invés de no âmbito de CAM, devido à maior popularidade da primeira aplicação, enquanto CAM pode ser vista como uma aplicação bem específica de processamento de sinais em telecomunicações.

1. O Capítulo 2 aborda CAM, introduz a base de dados *UFPATelecom*, sua importância e a metodologia utilizada para a sua constituição. Ademais, são apresentados os métodos tradicionais de classificação e a técnica de DL que foram aplicados ao experimentos de CAM e CTAR utilizando a base de dados desenvolvida. Também são demonstradas as características extraídas dos dados de entrada para os algoritmos de aprendizado de máquina.

2. O Capítulo 3 apresenta os principais resultados obtidos nas experimentações em CAM

utilizando amostras no domínio do tempo extraídas de base de dados proposta na literatura. Além disso, também é apresentado os resultados alcançado em CTAR utilizando amostras baseadas em conhecimento e domínio do tempo extraídas da base UFPATelecom. É realizada uma análise do desempenho dos algoritmos tradicionais em relação ao algoritmo de DL.

3. O Capítulo 4 introduz os principais elementos acerca do *Reinforcement Learning* e apresenta as técnicas escolhidas para serem aplicadas à problemática do controle de congestionamento. Além disso, é apresentado o algoritmo New Reno, o qual adota métricas tradicionais e provê o "baseline" a fim de gerar comparações de performance com os agentes DRL.

4. O Capítulo 5 descreve o *framework* (arcabouço) utilizado para a constituição da rede FH simulada e suas ferramentas. Além disso, é apresentada a modelagem utilizada pelos agentes DRL e as principais configurações que regem o ambiente de simulação.

5. O Capítulo 6 apresenta os principais resultados obtidos nas experimentações em controle de congestionamento. São apresentadas as características dos cenários de simulação analisados, sendo comparado o desempenho dos agentes DRL em relação ao algoritmo TCP New Reno. Além disso, é realizado uma breve discussão acerca dos resultados encontrados na literatura com os que foram alcançados nestas experimentações.

6. O Capítulo 7 apresenta as principais conclusões e as possíveis futuras extensões deste trabalho.

1.4 Contribuições da pesquisa

As principais contribuições deste trabalho consistem na criação da base de dados pública **UFPATelecom**, constituída de sinais LTE e GSM, e o processo de captura destes sinais. Além disto, foi realizada uma investigação do desempenho dos classificadores comparando algoritmos de aprendizado de máquina com técnica de DL em condições incompatíveis ("mismatch"), no processo de treino e teste. Através dos resultados obtidos, foi possível realizar a comparação com outros trabalhos da literatura para tirar conclusões acerca das possíveis discrepâncias e dificuldades em reproduzir os experimentos de forma perfeita quando dados e os softwares não estão totalmente disponíveis.

Além disso, também foi implementado um arcabouço para pesquisas em controle de congestionamento utilizando técnicas de DRL, através da configuração de um ambiente de simulação integrado pela ferramenta NS3Gym. A fim de validar o ambiente configurado, foram realizados testes preliminares com agentes implementados pelos algoritmos A3C e DQN, juntamente com um método que é tradicionalmente aplicado ao controle de congestionamento na literatura, o TCP New Reno. Os testes realizados através deste ambiente visaram averiguar as funcionalidades das ferramentas que compõem o arcabouço proposto, e se mostraram adequadas para futuras pesquisas e análises dos requerimentos de redes FH.

Capítulo 2

Sistemática para classificação automática de modulação (CAM)

No campo das aplicações de aprendizado de máquina, a disponibilidade de base de dados para o treino e teste de algoritmos é de suma importância. Devido às tendências de reprodutibilidade nas pesquisas, os autores em [41] disponibilizaram uma base de dados importante para CAM. Assim como em [42] foi disponibilizado uma base de dados com o emprego de imperfeições reais de canais a fim de gerar cenários realistas.

De acordo com o crescimento do aprendizado de máquina, dados são muito significativos para modelos de canais, sendo estes essenciais para o desenvolvimento de TAR's. Sendo assim, a base de dados pública *UFPATelecom* é constituída de sinais LTE e GSM na forma digitalizada e simulada, sendo utilizada na comparação de métodos de DL com outros tradicionalmente empregados em problemas de classificação.

A seguir será apresentada a base de dados *UFPATelecom* e os métodos que tradicionalmente são aplicados em problemas de aprendizado não-supervisionado, e o algoritmo de DL escolhido para CAM. Serão também apresentadas as configurações de simulações adotadas para o processo de treinamento e teste utilizando a base de dados UFPATelecom, assim como outra disponível em [11], para fins de comparações.

Segue a organização deste capítulo: A Seção 2.1 apresenta a base de dados UFPATelecom, a sua constituição e o processo de sua produção. A Seção 2.2 ilustra os métodos de aprendizado de máquina e a técnica de DL, suas características e a motivação para a escolha destes para a realização de CAM e CTAR. Por fim, a Seção 2.3 demonstra as características utilizadas como entrada para os classificadores.

2.1 Base de dados UFPATelecom

Dados estão disponíveis em grande quantidade ou são produzidos com baixo custo em muitas aplicações de aprendizado de máquina. Por exemplo, o sistema que converte texto para voz apresentado em [43], que representa o estado-da-arte, alcança qualidade próxima a linguagem humana natural após treinamento de 24.6 horas com fala digitalizada. Entretanto, para aplicações de aprendizado de máquina em telecomunicações, a quantidade de dados públicos ainda é limitada, para o uso em pesquisa e desenvolvimento. Sendo assim, esta é uma das principais motivações para a criação desta base de dados.

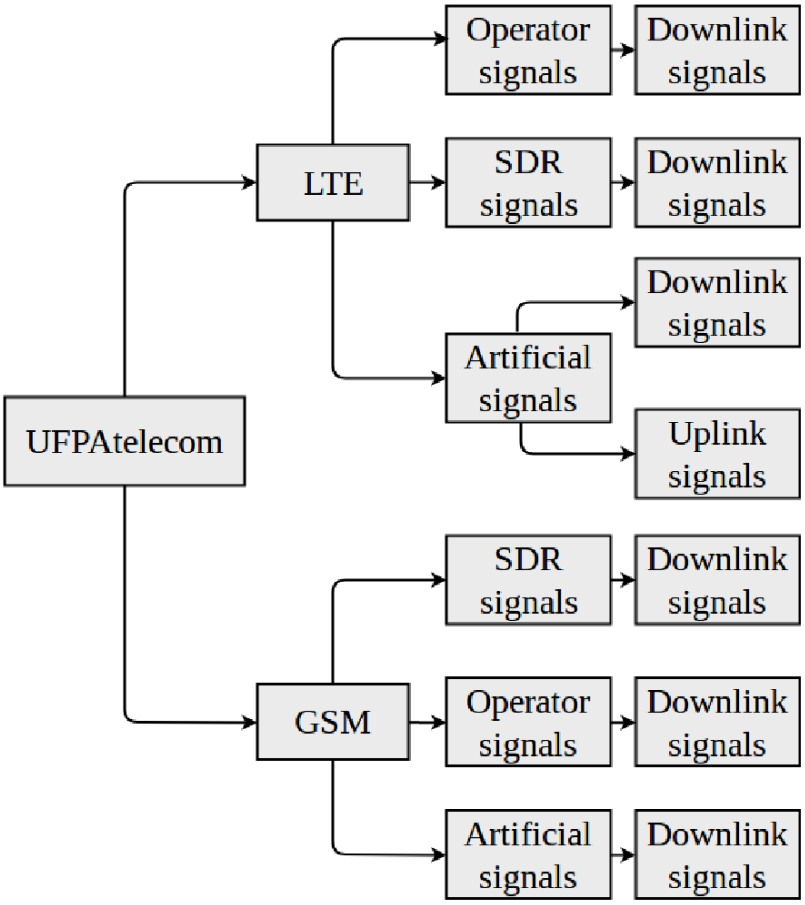
Outra característica importante considerada para geração da base UFPATelecom, é a variação dos canais de propagação. Existe uma quantidade significativa de bases de dados que usam diferentes tipos de modulação, porém com pequena alternância quanto aos tipos de canais utilizados. Por exemplo, em [11], um único canal é utilizado para gerar todos os sinais através do software GNU Radio, para a base de treino e teste. A relevância dos dados é diretamente dependente do canal, pois em um sistema de comunicação, o canal de propagação é a única parte que não é controlado por humanos. Sendo assim, neste trabalho este quesito é altamente priorizado, visando uma base de dados que auxilie na capacidade de generalização dos algoritmos dependentes de dados em telecomunicações.

Com estas motivações, a base de dados UFPATelecom foi criada para auxiliar na educação, pesquisa e desenvolvimento de algoritmos aplicáveis à classificação de TAR. Atualmente está disponível em [44] e é constituída de sinais GSM e LTE, dividida em três categorias: sinais **artificiais**, sinais **digitalizados** de operadoras de redes móveis e sinais de rádio definido por software **(SDR)** que foram gerados nas instalações do Núcleo de P&D em Telecomunicações, Automação e Eletrônica (LASSE) na UFPA. A organização geral da base pode ser vista na Figura 2.1 a seguir.

Os sinais artificiais demonstrados na Figura 2.1 foram gerados a partir do software MATLAB, variando o índice de razão sinal-ruído (do inglês, *Signal-to-Noise Ratio*) (SNR) de um sinal base sem ruído. De forma mais específica, a variação de SNR deu-se entre -20 a 20 dB a cada dois valores, criando assim 20 versões do sinal base original.

Os sinais de operadoras foram coletados utilizando o aparelho Keysight EXA Signal Analyser N9010A (VSA) com o software Keysight VSA 89600 (VSA software) [45]. Este dispositivo juntamente com seu software permite demodulação *online* e *offline*, e gera análises dos sinais recebidos que expressam dados no domínio do tempo e da frequência. A terceira ca-

Figura 2.1: Diagrama da base de dados UFPATelecom.



Fonte: Autor.

tegoria é a de sinais SDR que foram gerados utilizando configurações baseadas no *GNURadio* com o *Universal Software Radio Peripheral* (USRP). As informações gerais acerca do tamanho em Gigabytes (GB) e a duração total em segundos da base de dados UFPATelecom pode ser vista na Tabela 2.1.

Na Tabela 2.1 está relacionado apenas dados de *downlink*, estando em desenvolvimento as configurações para sinais de *uplink*. Nas seções a seguir serão descritas de forma mais detalhada as características de cada uma das categorias que constituem a base UFPATelecom.

Tabela 2.1: Duração em segundos (sec) e tamanho (GB) para as categorias de sinais da base UFPATelecom.

	LTE				GSM			
	Downlink		Uplink		Downlink		Uplink	
	Sec.	GB	Sec.	GB	Sec.	GB	Sec.	GB
Artificial	52.8	66	8.8	6.6	340	2.98	-	-
Operator	1.28	0.05	-	-	11.2	0.13	-	-
SDR	4800	199	800	20	17	0.33	-	-

Fonte: Autor.

2.1.1 Sinais Artificiais

De forma a criar os sinais GSM simulados, o software Osmocom [46] foi utilizado, pois possui uma extensa biblioteca dedicada à tecnologia GSM. Com o uso deste software, a base de sinais GSM de *downlink* foi criada, com 4 amostras por símbolo (do inglês, *samples-per-simbol* (sps)), obtendo um tamanho de *burst* de 625 bits em vez de 156 bits. Além disto, esta possui duração de 17 segundos (29.544 *bursts*), dado que cada *burst* de GSM dura cerca de 0.577 milisegundos. Após o processo de alternância do SNR, esta base possui um total de duração de 340 segundos e um tamanho total de 2.98 GB.

Para a base LTE, sinais de *downlink* e *uplink* foram criados com larguras de banda de 1.4, 3, 5, 10, 15 e 20 MHz utilizando o *Matlab LTE Toolbox* [47]. Após esta etapa, os sinais são transmitidos por um canal de propagação multi-caminho. Os parâmetros do canal de propagação utilizados são os definidos pela 3GPP [48], além do modelo selecionado ser o *Extended Pedestrian A* (EPA) com frequência Doppler de 5 Hz. Cada sinal base de *downlink* de cada uma das larguras de banda citadas possuem duração total de 2.64 segundos, enquanto que os sinais base de *uplink* tem uma duração total de 0.44 segundos. Levando em consideração o processo de variação de SNR e cada uma das larguras de banda utilizadas, a base LTE possui um total de duração de 52.8 segundos e um tamanho de 66 GB para *downlink*, e para *uplink* de 8.8 segundos com 66 GB de tamanho total.

2.1.2 Sinais de operadoras móveis

Os sinais de operadora foram digitalizados com o auxílio do *hardware* e *software* do VSA, como ressaltado anteriormente. As gravações foram realizadas no prédio do "Espaço Inovação"localizado no Parque de Ciência e Tecnologia do Guamá (PCT-Guamá), na UFPA. A busca pelos sinais LTE e GSM foram realizadas dentro das faixas de frequência distribuídas pela ANATEL para as operadoras móveis.

Para os sinais GSM, os valores dos canais absolutos de rádio frequência (do inglês, *Absolute Frequency-Radio Channel*) (ARFCN) foram selecionados manualmente com a ajuda dos dispositivos de *hardware* e *software* do VSA utilizando o módulo GSM/EDGE disponível no aparelho. Neste cenário, apenas 7 canais de *downlink* foram encontrados na faixa de frequência de 900 MHz.O módulo usado revela a constelação GMSK, tipo de *burst*, *slots* demodulados, assim como calcula medidas da qualidade de serviço dos sinais recebidos.Estes parâmetros foram utilizados para validar os sinais, já que apenas sinais GSM deveriam ser gravados. A base de sinais digitalizados GSM estão na faixa de frequência de 200 KHz, com frequência de amostragem de 640 KHz e foram salvos com *span* de frequência de 500 KHz, indicando a largura de banda analisada.

Foram utilizados os mesmos procedimentos para os sinais digitalizados LTE, sendo encontrados apenas 5 sinais de *downlink* que foram recebidos e analisados. O módulo avançado de LTE do *software* do VSA foi utilizado para verificar informações como, constelação da modulação, os valores dos canais físicos demodulados, os blocos de recursos ativos, as medidas de qualidade do sinal recebido, e outros parâmetros. As bandas de frequências disponíveis na base LTE digitalizada são de 10 MHz.

2.1.3 Sinais SDR *indoor*

Sinais GSM e LTE SDR foram gerados utilizando configurações do GNURadio com 2 USRP's. A primeira é utilizada para enviar o sinal artificial sem ruído pelo ar para que a outra USRP possa capturá-lo. Foram utilizados 560 sinais artificiais LTE, 480 para *downlink* e 80 para *uplink*, e apenas 1 sinal artificial GSM de *downlink*.

Na fase de transmissão e captura dos sinais LTE, estes são repetidos até que cada sinal tenha alcançado 10 segundos de duração total, enquanto que os sinais GSM são transmitidos uma única vez. Sendo assim, o tempo total dos sinais LTE de *downlink* são de 4800 segundos, sendo 800 segundos o tempo de duração para os sinais LTE de *uplink*, e 17 segundos para sinais

GSM de *downlink*.

Os sinais GSM e LTE foram salvos como arquivos binários, sendo a parte real da amostra (IQ) o primeiro elemento do arquivo, e a parte imaginária da primeira amostra IQ compõe a segunda parcela do arquivo binário, e assim por diante. Para utilizar em diferentes plataformas, foram usados números do tipo *float* em formato *little-endian*.

2.2 Classificadores para CAM e CTAR

Conforme descrito anteriormente, CAM e CTAR consistem na identificação dos esquemas de modulação ou a detecção da tecnologia de rádio utilizada por um dado sistema de comunicação com alta probabilidade de acerto em um período curto de tempo. Os exatos valores de probabilidade e o período de detecção dependem da aplicação, especialmente se o processamento pode ser feito *online* ou *offline*.

Outro fator importante está relacionado ao conhecimento prévio do classificador como, por exemplo, taxa de símbolo, largura de banda do sinal, frequência de portadora, dentre outros. Na literatura que retrata trabalhos em CAM e CTAR, os algoritmos são testados assumindo modelos de canais simplificados e/ou parâmetros estimados de forma perfeita, o que pode ser considerado um fator não-realista para diversos cenários.

Além de bases de dados variadas, também foi considerado neste trabalho algoritmos com características diferentes em relação ao custo computacional e acurácia. A seguir serão apresentados três métodos tradicionalmente aplicados em tarefas de aprendizado não-supervisionado e a técnica de DL selecionada para este experimento.

2.2.1 Árvore de Decisão

A árvore de decisão (do inglês, *Decision Tree*) (DT) é uma técnica considerada um conjunto de regras *if/else* baseadas em limiares, que são geralmente aplicadas em tarefas de classificação e regressão. As árvores de decisão possuem nós que representam atributos, sendo cada ramo de um nó um valor possível deste atributo. As folhas provenientes de um conjunto de ramos estão associadas a uma classe, e as regras de classificação são mapeadas da raiz dos nós até as folhas.

No processo de classificação, a DT estima a probabilidade de que uma instância pertença a uma classe k específica. O primeiro passo é uma busca realizada pela árvore a fim de en-

contrar a folha que representa a instância, e então é calculada a probabilidade desta pertencer a uma dada classe. A DT é uma técnica considerada um modelo *white box* devido sua fácil interpretabilidade [49].

2.2.2 Floresta Aleatória

A Floresta aleatória (do inglês,*Random Forest*) (RF) pode ser descrita como um conjunto de árvores de decisão combinadas a fim de alcançar maior acurácia e maior estabilidade. De forma geral, o processo de treinamento de uma RF é realizado pelo método *bagging* que, de forma simplificada, representa a combinação de modelos de aprendizado. Este algoritmo é aplicado em tarefas de classificação e regressão.

O algoritmo RF busca pela melhor característica quando ocorre a partição de um nó em um sub-grupo de características, adicionando maior aleatoriedade ao processo de crescimento da árvore [49]. Essa ação melhora a diversidade das árvores, além de gerar um melhor modelo geral. Além disso, a RF apresenta facilidade na medição da importância de uma característica, o que é uma das vantagens deste método.

2.2.3 *Naive Bayes*

Outro método comumente aplicado à tarefas de classificação é o algoritmo Naive Bayes (NB), que aplica o Teorema de Bayes considerando a indepêndencia entre os pares de características, caracterizando assim uma suposição ingênua. Este algoritmo pode ser bem rápido em comparação com outros métodos, além de aliviar problemas relacionados a dimensionalidade [49].

Esse método é ainda muito aplicado em problemas de categorização de textos, utilizando a frequência de palavras como características. Além disso, o algoritmo NB pode ainda ser bem competitivo, com uma etapa de pré-processamento apropriada, quando comparado com métodos mais avançados como *Support Vector Machines* (SVM) [50].

2.2.4 Rede Neural Convolucional

A Rede Neural Convolucional (do inglês, *Convolutional Neural Network*) (CNN) é a técnica de DL selecionada devido a sua alta empregabilidade em aplicações de CAM, garantindo performance ótima em relação a níveis de acurácia, conforme demonstrado na seção 1.1.

A CNN é composta de múltiplas camadas escondidas, diferenciando-se da rede neural tradicional pela camada convolucional que ela possui. Esta camada combina características de baixo nível com características de alto nível nas camadas consecutivas alcançando assim alta performance [49].

Dentre as vantagens de utilizar a CNN, é possível destacar a capacidade de extração de características de dados sem pré-processamento, não havendo portanto, a necessidade de fornecimento de informações a priori, minimizando assim o esforço humano no desenvolvimento de suas funções principais.

2.3 Extração de características

Todos os classificadores apresentados na Seção 2.2 são treinados e testados com $N_{(tr)}$ e $N_{(te)}$ amostras, respectivamente. A entrada para os classificadores é um vetor com K elementos de valores reais. Neste trabalho foram adotados dois conjuntos distintos de características para representar os sinais, sendo o primeiro um conjunto de amostras de valores complexos dos sinais no domínio do tempo, em similaridade ao que foi feito em [11]. O segundo conjunto são características estatísticas conforme adotado em [51].

2.3.1 Amostras no domínio do tempo

Os algoritmos de CAM foram aplicados também em uma base de dados que é variação da base *RML2016.10a*, apresentada em [11], como por exemplo, a base *RML2018.03* que é uma versão modificada da primeira.

Dada a capacidade das CNN em extrair características de dados brutos, os autores em [11] utilizaram uma janela de 128 amostras consecutivas no domínio do tempo, de valores complexos, como entrada para as redes profundas. Sendo assim, o vetor de características de entrada possui dimensão $K = 256$ para os componentes reais e imaginários. O sinal gerado para cada modulação passa através de um canal simulado (modelo GNU Radio) que inclui *random walk drifting* do oscilador de frequência de portadora, desvanecimento multi-canal da resposta ao impulso do canal e ruído branco gaussiano aditivo, sendo este utilizado como entrada para os métodos de classificação.

2.3.2 Características baseadas em conhecimento

De maneira a gerar entradas para os classificadores, foram adotados os conjuntos de características propostos em [51]. Para a extração de características, são calculados os valores de $A[k]$ e $X[k]$, que são o componente de frequência de magnitude máxima quadrática e o valor máximo da Transformada Discreta de Fourier (TDF), respectivamente.

Estes valores são calculados a fim de obter características que considerem a razão entre dois máximos nos valores absolutos de $X[k]$. Além disso, os valores da estimativa da frequência central de acordo com o tamanho da janela de observação das amostras, e a razão entre as potências dos sinais dentro de larguras de banda diferentes em torno de uma frequência central, também são utilizados como características para os classificadores.

Capítulo 3

Resultados das experimentações em CAM e CTAR

Neste capítulo serão apresentados os resultados alcançados pelos métodos de DL aplicados aos experimentos em CAM e CTAR.

Segue a organização deste capítulo: A Seção 3.1 apresenta os resultados de classificação de modulação aplicada à amostras no domínio do tempo. A Seção 3.2 demonstra os resultados obtidos nos experimentos em classificação de tecnologias de acesso de rádio comparando o desempenho de algoritmos tradicionais com a rede neural convolucional.

3.1 Resultados de CAM

Os resultados de CAM são apresentados utilizando as base de dados *RML2016.10a* e *RML2018.03*. Conforme descrito na Seção 2.3, foram adotadas $K = 256$ características para experimentos com estas duas bases de dados, totalizando 128 amostras de valores complexos no domínio do tempo. Para CAM, não foram utilizadas amostras baseadas em conhecimento, conforme apresentado na Seção 2.3.2.

O objetivo dos experimentos em CAM é para a comparação com trabalhos referência na literatura, no contexto da aplicação de DL, sendo assim, os resultados apresentados são contrastados com o que foi feito em [11]. Nos experimentos foram utilizados a mesma CNN utilizada em [11], porém com diferentes probabilidades de *dropout*, que é a técnica que suspende alguns neurônios durante a etapa de treinamento. Esta técnica ajuda na capacidade de generalização da rede e é largamente aplicada em DL.

Os parâmetros adotados para os classificadores são apresentados a seguir. Para DT, foi atribuído a profundidade máxima da árvore de 80, sendo para RF o número de árvores igual a 80 e profundidade máxima de 200. Para CNN, o número de épocas adotados foi de 100.

A Tabela 3.1 mostra a acurácia média obtida para todos os valores de SNR para todos os classificadores apresentados anteriormente. É possível verificar que, em todos os casos, exceto para o algoritmo RF, os testes com a base de dados *RML2018.03* alcançaram maior performance, sendo a maioria dos resultados comparáveis aos obtidos em [11], porém não sendo exatamente iguais.

Tabela 3.1: Média geral de acurácia dos classificadores para as base de dados *RML2016.10a* e *RML2018.03* utilizando amostras no domínio do tempo.

Algorithms	**RML2016.10a**	**RML2018.03**
CNN2 Dropout (60%)	62.3%	63.6%
CNN2 Dropout (50%)	62.7%	67.0%
CNN2 Dropout (0%)	68.8%	69.1%
Árvore de Decisão	24.4%	25.4%
Naive Bayes	18.5%	20.2%
Floresta Aleatória	13.6%	13.3%

Fonte: Autor.

As conclusões que podem ser feitas através dos resultados apresentados pela Tabela 3.1 são que, embora as redes neurais convolucionais sejam capazes de extrair informações de dados não-processados, os classificadores tradicionais precisam de características mais apropriadas, sendo as amostras no domínio do tempo não adequadas para métodos que utilizam regras *if/else*, como é o caso da árvore de decisão.

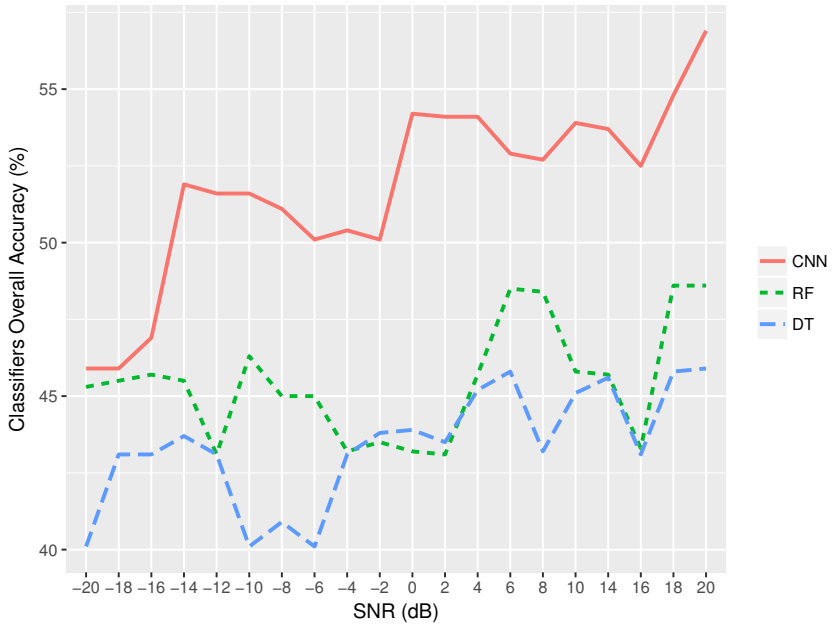
3.2 Resultados de CTAR

Para experimentos em CTAR foi usada a base de dados UFPATelecom com amostras no domínio do tempo, assim como as características baseadas em conhecimento. O número total de exemplos N disponíveis depende das características adotadas. Para amostras no domínio

do tempo, N pode ser facilmente calculado dado o número total de amostras e a duração da janela, assim como para as características baseadas em conhecimento, cada 512 amostras de dados brutos gera exemplos de 136 amostras (sendo que cada número de amostras representa as características apresentadas na Seção 2.3.2).

Os valores de N para cada categoria dos sinais quando utilizado características baseadas no conhecimento são 560040, 25768 e 295090 para a base artificial (ar), de operadoras (op) e SDR (sdr), respectivamente. A Figura 3.1 mostra os valores de acurácia para a classificação de tecnologias de rádio sobre todos os valores de SNR quando utilizados amostras no domínio do tempo como características para os classificadores. A Tabela 3.2 mostra os valores de acurácia para CTAR quando usadas as amostras baseadas em conhecimento.

Figura 3.1: Acurácia em CTAR para as bases de teste com amostras no domínio do tempo para valores diferentes de SNR.



Fonte: Autor.

Os resultados em CTAR revelaram que, utilizando amostras no domínio do tempo, não proveram bons resultados mesmo com o uso de CNN em comparação aos resultados alcançados em CAM, neste trabalho e em [11]. Entretanto, quando as amostras utilizadas em [51] foram aplicadas, houve um aumento de acurácia para o conjunto de teste dos sinais artificiais, quando os sinais de operadora compõe o conjunto de treino em 50 por cento. A acurácia de todos os

Tabela 3.2: Acurácia geral em CTAR utilizando amostras baseadas em conhecimento como características.

Train dataset	Test dataset	CNN	DT	RF
50% *ar*	50% *ar*	90.00%	83.36%	89.82%
50% *ar*	50% *op*	47.69%	52.65%	41.706%
50% *ar*	50% *sdr*	52.39%	53.23%	45.124%
50% *ar* + 50% *op*	50% *ar*	88.56%	83.86%	88.98%
50% *ar* + 50% *op*	50% *op*	98.932%	99.976%	99.98%
50% *ar* + 50% *op*	50% *sdr*	96.32%	98.32%	97.9%

Fonte: Autor.

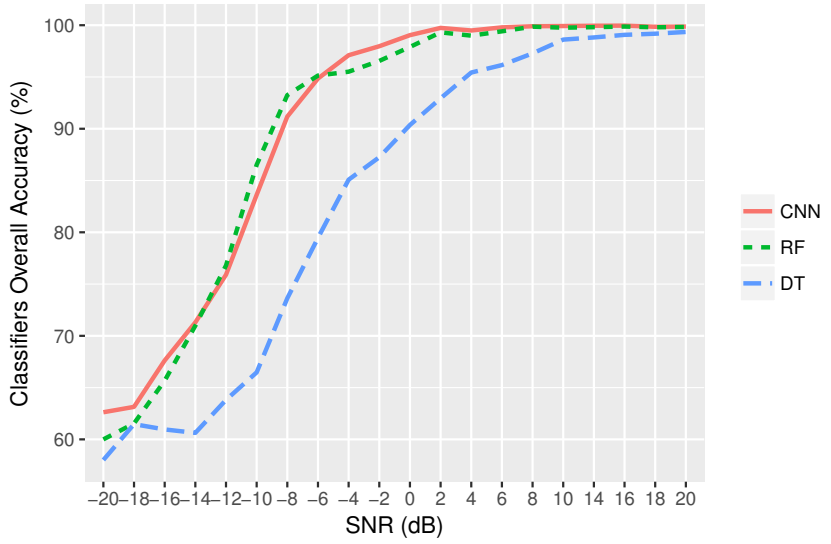
conjuntos de teste apresentaram bons resultados.

A Tabela 3.2 ilustra que os classificadores RF e DT atingiram performance similar a CNN na maioria dos cenários constituídos de dados de sinais *indoor* e *outdoor* utilizados no conjunto de treino, o que sugere que outros fatores além de performance devem ser considerados neste caso.

A Figura 3.2 mostra a performance dos classificadores em CTAR para diferentes valores de SNR no conjunto de teste. Neste caso, apenas amostras de sinais artificiais foram utilizados no conjunto de treino. A combinação de dados de diferentes origens na fase de treino ajuda na avaliação de robustez dos classificadores, sendo que neste trabalho buscou-se revelar os resultados em condições incompatíveis.

O algoritmo DT é o método com menor custo computacional, e obteve o processamento mais rápido em todos os cenários, com um valor de profundidade máxima $D = 80$. Além disso, o algoritmo RF tem menor complexidade computacional em comparação a CNN que é constituída de 3 camadas de *dropout*, 2 camadas convolucionais e 2 camadas densas que demandam um maior tempo de processamento, assim como um maior nível de computação entre as camadas.

Figura 3.2: Acurácia em CTAR para o conjunto de teste com valores distintos de SNR utilizando características baseadas em conhecimento.



Fonte: Autor.

Capítulo 4

Fundamentos do *Reinforcement Learning* e técnicas para o Controle de Congestionamento

No capítulo anterior foram abordadas as principais características acerca da arquitetura C-RAN e os desafios impostos para a rede *fronthaul* baseada em pacotes. Como discutido anteriormente, além de requerimentos restritos de velocidade e latência, o controle de congestionamento é uma parte extremamente importante para evitar a perda de pacotes e garantir o atendimento justo de todas as RRHs.

Neste capítulo serão complementados em relação à introdução, os conceitos centrais de RL que fundamentam os métodos utilizados durante as simulações de rede e a motivação para a escolha destes. Para fins de comparação de resultados, será apresentado de maneira mais aprofundada o algoritmo de controle de congestionamento já ressaltado anteriormente, o TCP New Reno.

Este capítulo tem a seguinte divisão: A Seção 4.1 introduz a importância e os conceitos básicos do *Reinforcement Learning*. A Seção 4.2 apresenta de forma mais aprofundada os elementos fundamentais do *Reinforcement Learning* que constituem o aprendizado de agentes DRL. A Seção 4.3 apresenta os métodos de *Deep Reinforcement Learning*, suas aplicações e suas particularidades. Por fim, a Seção 4.4 descreve, de forma geral, o os conceitos centrais acerca do controle de congestionamento realizado pelo TCP e as características do algoritmo TCP New Reno.

4.1 Introdução ao *Reinforcement Learning*

O *Reinforcement Learning* é uma área de aprendizado que considera um agente que deverá mapear ações a situações diversas a fim de maximizar a recompensa recebida por cada ação tomada ao longo do tempo. As principais características desta área é concernente a situações em que as ações tomadas não apenas trarão recompensas imediatas mas também afetarão as ações e circunstâncias futuras [52].

De forma geral, os métodos de RL tentam solucionar os problemas através da definição de três aspectos: os **estados** encontrados em dado cenário, as **ações** possíveis e o **objetivo** principal. Sendo assim, um agente que interage em um ambiente deve ser capaz de capturar o estado atual e tomar decisões que afetem o estado encontrado, além de definir objetivos em relação a este estado.

Um dos aspectos que diferenciam o RL das demais formas de aprendizado, como o aprendizado supervisionado e o não-supervisionado, consiste no agente que aprende através de suas próprias experiências. Diferentemente do aprendizado supervisionado em que um supervisor externo recebe amostras que caracterizam uma situação com um rótulo indicando uma ação que agente deve tomar, os métodos de RL tomam decisões dependendo exclusivamente de sua interação com o ambiente, sendo portanto, mais adequado a este tipo de problema.

Os métodos não-supervisionados tem por característica procurar estruturas escondidas nos dados, sendo que em RL o objetivo principal é maximizar a recompensa das ações, ainda que seja útil para o agente descobrir estas estruturas. Sendo assim, o RL é considerado um terceiro paradigma de aprendizado de máquina, além do aprendizado supervisionado e não-supervisionado [52].

Um dos desafios do aprendizado em RL é o dilema entre **exploração** e **utilização** de ações já conhecidas. O agente deve ser capaz de manter o equilíbrio entre explorar as ações bem sucedidas, e experimentar novas ações que possivelmente potencialize as recompensas retornadas pelo ambiente. Essa não é uma tarefa fácil, pois o agente deve garantir a utilização e exploração sem falhar no objetivo central do problema.

As vantagens do RL moderno consiste na sua aplicabilidade na engenharia e nas diversas áreas científicas, principalmente na sua integração com a inteligência artificial e aprendizado de máquina para solucionar problemas de otimização, estatística e matemática [52]. Devido ao potencial inerente as técnicas de RL para lidar com a incerteza de estados dos mais diversos problemas, este será estudado e aplicado ao controle de congestionamento neste trabalho.

4.2 Principais elementos do *Reinforcement Learning*

Outros conceitos fundamentais a serem ressaltados são a **política** usada pelo agente, a definição do sinal de **recompensa**, a **função-valor** utilizada, e possivelmente a declaração do **modelo do ambiente** a ser estudado.

A política utilizada pelo agente define como serão mapeadas as ações quando encontrado determinados estados. De forma geral, a política adotada pode ser estocástica, o que atribui probabilidades às ações disponíveis ao agente. Essa característica é uma das partes centrais para determinar o comportamento do agente.

O sinal de recompensa diz ao agente o quanto as ações tomadas foram boas ou ruins. Essas atribuições de valor informam de forma imediata ao agente como mudar a política, pois se uma ação escolhida retornou um baixo valor de recompensa, então a política deve ser mudada para outra decisão no futuro. Deste modo, as recompensas devem ser funções estocásticas dos estados do ambiente e das ações tomadas, sendo que o objetivo principal do agente é potencializar a soma das recompensas ao longo do tempo [52].

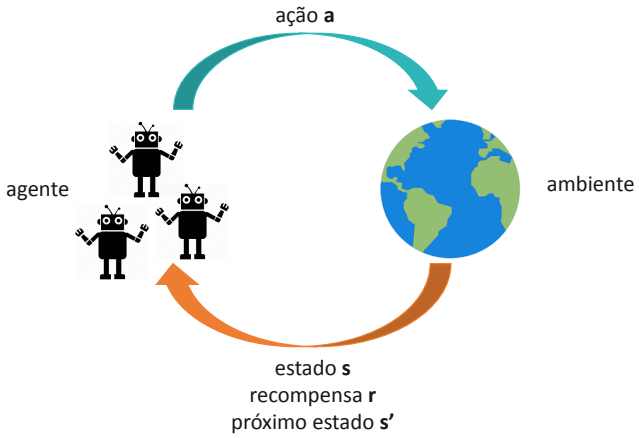
Enquanto que os sinais de recompensa atribuem valores imediatos às ações, o valor de um estado é definido pela soma de recompensas que o agente espera receber a longo prazo [52]. Sendo assim, a função-valor especifica o quanto o comportamento do agente está sendo útil ao longo do tempo, sendo um indicativo do quanto os estados são desejáveis através das recompensas futuras que sucedem o primeiro passo. Por exemplo, ainda que em um estado específico o agente tenha recebido uma baixa recompensa, a função-valor poderá ter um valor alto, pois é possível que os estados subsequentes garantam boas recompensas.

Um dos aspectos mais importantes do RL é a estimativa do valor dos estados, pois a escolha pelas ações que conduzem a estados com maior valor é mais importante do que as que conduzem a recompensas superiores, porque a longo prazo estas ações conduzem a maior retribuição.

Outro conceito relevante usado para fins de planejamento é o modelo. Através deste, é possível verificar os próximos estados e recompensas a partir das ações adotadas, podendo assim analisar o comportamento do ambiente. Há os métodos que usam modelos, os quais experimentam ações e verificam estados antes que eles sejam vivenciados, e os métodos livres de modelos que tem o aprendizado por tentativa e erro. Na figura 4.1 é possível verificar a representação da dinâmica de um modelo RL.

Com base nas definições aqui introduzidas, na próxima seção é possível apresentar os

Figura 4.1: Modelo de *Reinforcement Learning*.



Fonte: Autor.

métodos RL utilizados nos experimentos deste trabalho.

4.3 *Deep Reinforcement Learning* e metodologias

Dentre as várias definições de DL é possível destacar que esta é um sub-campo do aprendizado de máquina em que o aprendizado se baseia em muitos níveis de representação, de forma a modelar a relação entre os dados [53]. Além disso, esta representação em níveis (ou camadas) maiores são definidas por características ou fatores que provém de níveis inferiores, sendo esta hierarquia o que especifica a arquitetura profunda [53].

Desde o surgimento das primeiras técnicas de DL, esta vem ganhando espaço em diversas aplicações, como visão computacional, detecção de objetos, reconhecimento de fala e outras, devido a sua capacidade em extrair informações de níveis mais altos de dados sensoriais brutos. As redes neurais artificiais foram usadas como base para a criação de técnicas de DL, como redes convolucionais, redes neurais recorrentes, máquinas de Boltzman, sendo aplicadas em tarefas de aprendizado supervisionado e não-supervisionado.

Conforme já ressaltado na seção anterior, as técnicas de RL não necessitam de amostras rotuladas para o treinamento, isso também representa um desafio devido ao atraso existente entre a ação e a recompensa, se comparada a associação direta entre as entradas e os alvos no aprendizado supervisionado [54].

De forma a solucionar outras questões relacionadas com a interdependência das amostras e a mudança de distribuição dos dados, em [54] uma nova técnica foi proposta a qual utiliza uma rede neural convolucional treinada com uma versão modificada do algoritmo *Q-learning* juntamente com a técnica de gradiente descendente pra atualizar os pesos da rede. Esta foi uma das primeiras técnicas de DRL a serem aplicadas com desempenho ótimo aplicado aos jogos Atari 2600 da *Arcade Learning Environment*.

A seguir serão vistos os métodos de DRL, seus fundamentos e principais aplicações que foram levados em consideração para a utilização neste trabalho.

4.3.1 *Deep Q-Network* (DQN)

O algoritmo *Deep Q-Network*(DQN) foi proposto pela *DeepMind Technologies* [54] em 2012, com a proposta de aliar um algoritmo RL com uma *Deep Neural Network* a fim de receber dados de imagens RGB que possam ser usadas durante o treinamento, sendo feita a atualizações dos pesos da rede com o gradiente descendente estocástico.

Este algoritmo utiliza a técnica de *replay* de experiência para guardar o conhecimento adquirido pelo agente em um espaço de tempo, representado por $e_t = (s_t, a_t, r_t, s_{t+1})$, sendo s_t o estado no tempo t, a ação a_t no tempo t, r_t a recompensa recebida do ambiente no tempo t e o s_{t+1} o próximo estado. As experiências adquiridas são armazenadas em um banco de dados $D = e_1, ..., e_N$ reunidos por muitos episódios e salvos na memória *replay*.

Uma das etapas do algoritmo consiste na atualização de amostras da experiência através do algoritmo *Q-learning*. Este algoritmo faz parte de uma das formas de aprendizado em RL mais importantes denominada **Diferença-Temporal** (DT). Métodos DT tem por característica a atualização da estimativa da função-valor v_π, seguindo uma política π, a cada período de tempo.

Além disso, o algoritmo *Q-Learning* faz uso de uma importante identidade chamada **equação Bellman**, que estabelece a função estado-ação ótima $Q^*(s, a)$, a qual define que recompensas futuras terão os seus valores descontados por uma variável γ, a cada tempo t, sendo portanto, o máximo retorno recebido, $R_t = \Sigma_{t+1=t}^{T} \gamma^{t+1-t} r_{t+1}$, em que T é o tempo em que o estado terminal é encontrado.

De forma a definir a função estado-ação ótima Q $^*(s, a)$, a equação Bellman estabelece que se o valor ótimo da sequência de estado s_{t+1} é conhecido por todas as ações a_{t+1}, então a melhor decisão é a escolha da ação que maximiza o valor esperado de r + $\gamma^*(s^{t+1}, a^{t+1})$, sendo r a recompensa recebida. Sendo assim, a função estado-ação ótima pode ser definida a seguir como:

$$Q^*(s,a) = \sum_{s^{t+1},r} p(s^{t+1}, r|s, a)[r + \gamma max_a^{t+1} Q^*(s^{t+1}, a^{t+1})] \tag{4.1}$$

No algoritmo DQN, a função Q definida pelo algoritmo *Q-learning* é utilizada como função de aproximação para o caso ótimo, sendo definido na Equação 4.2 a seguir.

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma max_a Q(S_{t+1}, a) - Q(S_t, A_t)] \tag{4.2}$$

A seleção de ações é realizada de forma aleatória com probabilidade ϵ, caso contrário é realizada pela política *e-greedy* que determina que a ação que possui maior valor Q será selecionada. As entradas na rede neural são produzidas através da função ϕ para ser processado pela função Q que pode ser vista no Algoritmo 1 abaixo.

Algorithm 1: Deep Q-Network (DQN)

Inicializar memória *replay* D com capacidade N

Inicializar função Q com pesos aleatórios

for *episódio*$= 1, M$ **do**

  Inicializar sequência $s_1 = x_1$ e sequência pré-processada $\phi_1 = \phi(S_1)$

  for $t = 1, T$ **do**

    Com probabilidade ϵ selecione a ação a_t

    Caso contrário selecione $a_t = max_a Q * (\phi(s_t), a; \theta)$

    Executar ação a_t e observar a recompensa r_t e imagem x_{t+1}

    Definir $s_{t+1} = s_t, a_t, x_{t+1}$ e pré-processar $\phi_{t+1} = \phi(s_{t+1})$

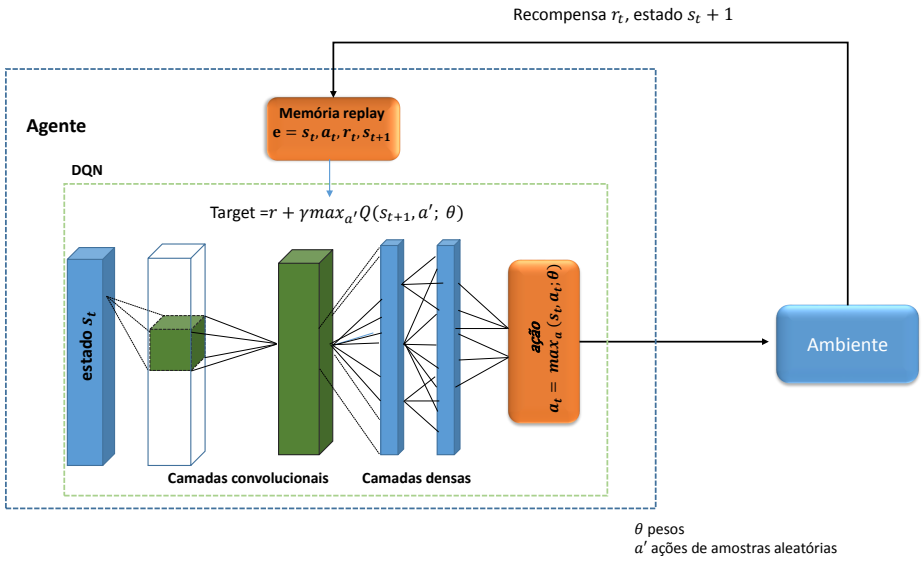
    Guardar transições $(\phi_t, a_t, r_t, \phi_{t+1})$ em D

    Selecionar amostras das transições aleatoriamente $(\phi_t, a_t, r_t, \phi_{t+1})$ de D

    Definir $y_j = \begin{cases} r_j \\ r_j + \gamma max'_a Q(\phi_{j+1}, a'; \theta) \end{cases}$

As vantagens de *Deep Q-Network* em relação ao método Q-learning tradicional consiste na potencialização da eficiência de dados através da atualização dos pesos da rede realizada múltiplas vezes, além da quebra da correlação entre os dados devido a seleção de amostras aleatórias, reduzindo assim a variância das atualizações [54]. Os esquema e o conjunto de passos tomados pelo algoritmo pode ser visto na Figura 4.2 a seguir.

Figura 4.2: Diagrama do algoritmo DQN.



Fonte: Autor.

Além de ser uma técnica pioneira no ramo de DRL alcançando muito sucesso no ramo dos jogos, e até o presente momento, não haver na literatura pesquisas relacionadas ao uso deste método em protocolos de controle de congestionamento, neste trabalho é experimentada a aplicação de DQN de forma inovadora nesta problemática.

4.3.2 *Asynchronous Advantage Actor-Critic* (A3C)

O algoritmo A3C surgiu para acrescentar soluções para problemas que surgiram com o uso da técnica de *replay* de experiência, sendo estes o uso excessivo de memória e poder computacional por interação com o ambiente, além da necessidade de algoritmos *off-policy* para a atualização dos dados de políticas antigas [55].

As vantagens advindas com o algoritmo A3C são o uso de paralelismo para múltiplas cópias do ambiente, não necessitando o uso de *replay* de experiência, além de não exigir a utilização de Unidade de Processamento Gráfico (UPG), sendo executado em uma única máquina com múltiplos núcleos em uma CPU.

Este algoritmo faz parte de uma classe de métodos chamada *actor-critic* pois realiza aproximações tanto da política como da função valor. O "ator"é uma referência a política aprendida

e o "crítico"se refere à função-valor conhecida, sendo neste caso, a função estado-valor [52]. O crítico indica através da função-valor o quanto a ação tomada foi boa ou ruim, sendo a política atualizada pelo ator.

Além disso, algoritmos *actor-critic* são também classificados como métodos *policy-gradients*, pois os parâmetros da política são estimados pelo cálculo de gradientes de uma medida escalar de performance [52]. Neste método temos: $\pi(a|s,\theta) = Pr(A_t = a|S_t = s, \theta_t = \theta)$, sendo a probabilidade da ação a, tomada no tempo t, dado o ambiente no estado s no tempo t, com parâmetro θ.

O algoritmo do A3C emprega a estimativa da função-valor $V(s_t;\theta_v)$ e a política $\pi(a|s,\theta)$, sendo atualizadas a cada número de ações(t_{max}) ou quando um estado terminal é encontrado. Assim, a atualização aplicada pelo algoritmo é feita por $\nabla_{\theta'} \log \pi(a_t|s_t;\theta')A(s_t,a_t;\theta,\theta))$, em que $A(s_t,a_t;\theta,\theta)$ é a **função vantagem** dada por:

$$A(s_t,a_t;\theta,\theta) = \sum_{i=0}^{k-1} \gamma^i r_{t+i} + \gamma^k V(s_{t+k};\theta_v) - V(s_t;\theta_v) \tag{4.3}$$

A variável k representa a mudança de estado e é limitada ao número máximo de ações executadas em um período. Uma saída *softmax* de uma rede neural convolucional é utilizada para a política $\pi(a|s,\theta)$ e outra saída linear para a função-valor $V(s_t;\theta_v)$ com as demais camadas compartilhadas [55].

A entropia (H) da política π é utilizada na função objetivo para evitar a convergência prematura em políticas determinísticas [55]. O cálculo do gradiente da função objetivo incluindo o termo de regularização da entropia (θ) pode ser visto a seguir.

$$\nabla_{\theta'} \log \pi(a_t|s_t;\theta')(R_t - V(s_t;\theta_v)) + \beta \nabla_{\theta'} H(\pi(s_t;\theta')) \tag{4.4}$$

O Algoritmo 2 que detalha o método A3C pode ser visto a seguir.

Algorithm 2: Asynchronous Advantage Actor-Critic (A3C)

//Vetor de parâmetros θ e θ_v compartilhados e contador global $T = 0$

Inicializar contador de passos t$\leftarrow$1

repeat

  Reinicializar gradientes: d$\theta \leftarrow 0$ e d$_\theta \leftarrow 0$

  Sincronizar parâmetros específicos $\theta' = \theta$ e $\theta'_v = \theta_v$

  $t_{start} = t$

  Tomar t_{start}

  repeat

    Faça a$_t$ de acordo com a política $\pi(a|s, \theta)$

    Receber recompensa r$_t$ e novo estado s$_{t+1}$

    t $\leftarrow t + 1$

    T $\leftarrow T + 1$

  until *Estado terminal* s_t *ou t-*$t_{start} == t_{max}$

  $R = \begin{cases} 0 \\ V(s_t, \theta'_v) \end{cases}$

  for $i \in t - 1, ..., t_{start}$ **do**

    $R \leftarrow r_i + \gamma R$

    Acumular gradientes $d\theta' : d\theta' \leftarrow d\theta + \nabla_{\theta'} \log \pi(a_i|s_i; \theta')(R_i - V(s_i; \theta'_v))$

    Acumular gradientes $d\theta'_v : d\theta_v \leftarrow d\theta_v + \partial(R - V(s_i; \theta'_v))^2 / \partial\theta'_v$

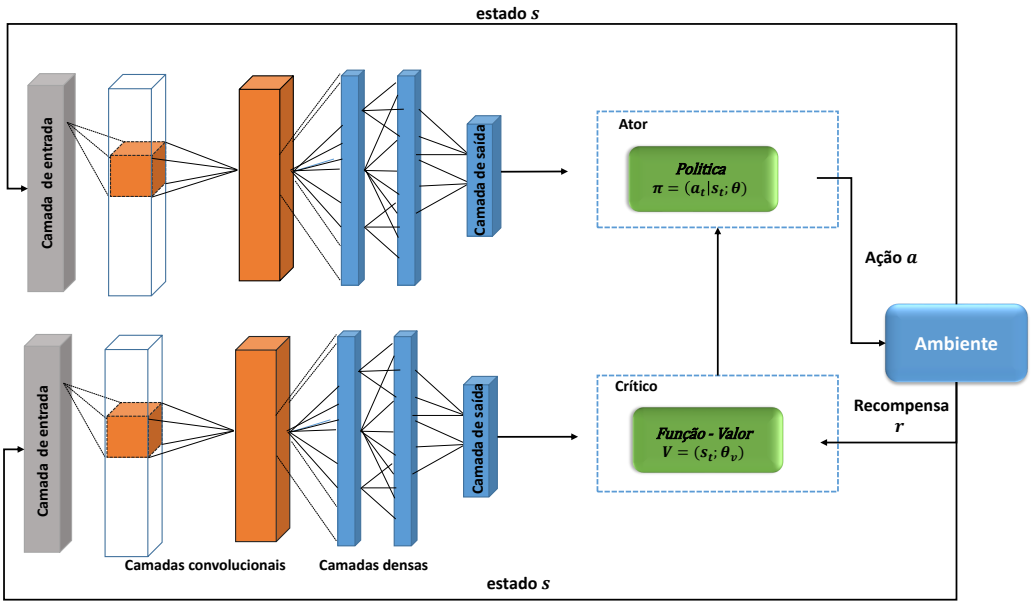
  Atualização assíncrona de θ usando $d\theta$ e de θ_v usando $d\theta_v$

until $T \geq T_{max}$

Uma das vantagens dos métodos de políticas parametrizadas é que a aproximação da política tende a ser determinística, diminuindo assim a chance de selecionar ações aleatoriamente e portanto mais ineficazes como em políticas *e-greedy* [52]. A Figura 4.3 a seguir pode ser visto um fluxograma explicitando o processo de métodos *actor-critic* no geral.

Até o presente momento, não são encontrados métodos *actor-critic* implementados no âmbito do controle de congestionamento, sendo portanto uma nova abordagem com grande potencial para problemas com alto nível de não-estacionariedade.

Figura 4.3: Diagrama de algoritmos *actor-critic*.



Fonte: Autor.

4.4 Controle de congestionamento e o TCP NewReno

O TCP (*Transmission Control Protocol*) foi desenvolvido a fim de garantir a entrega de informações de forma confiável e em sequência. Com isso, seu objetivo é oferecer um fluxo de bytes fim a fim confiável em inter-redes não confiáveis [56]. Isto torna-se necessário devido as inter-redes possuírem propriedades diversas, sendo o TCP projetado para ser robusto diante de diferentes tipos de falhas.

Para o controle de congestionamento, a medida tomada pelo TCP é a observação da ocorrência de *timeouts* de transmissão, pois estes usualmente indicam que há congestionamento na rede [56]. De modo a evitar o congestionamento, o transmissor possui o conhecimento de duas janelas de transmissão: a janela do receptor e a janela de congestionamento. Cada uma delas indica quanto o transmissor pode enviar, sendo a quantidade de bytes a ser enviada o valor mínimo entre as duas janelas.

Durante o estabelecimento de uma conexão, o transmissor ajusta a janela de congestionamento para o tamanho máximo de um segmento, caso este segmento seja confirmado antes da ocorrência de um *timeout*, o transmissor enviará dois segmentos. A medida que a quantidade de n segmentos forem enviados e confirmados, a janela de congestionamento será aumentada

em número de bytes correspondentes a n segmentos. De forma geral, cada segmento máximo confirmado duplica a janela de congestionamento. A janela, então mantém seu crescimento exponencial até que um *timeout* ocorra ou o tamanho da janela do receptor seja alcançado. Esta fase é conhecida como *slow-start*.

Os protocolos da Internet em geral, além dos conceitos de janela do receptor e de congestionamento, aplicam outro parâmetro conhecido como limiar. O limiar tem seu tamanho inicial definido, e em caso de *timeout* tem seu tamanho reduzido pela metade do valor da janela de congestionamento, interrompendo assim o crescimento exponencial da janela que caracteriza a fase de *slow-start*. A partir do momento em que a transmissão atinge o limiar, o crescimento da janela passa a ser linear. Esta fase é conhecida como *congestion-avoidance*.

O algoritmo TCP New Reno utiliza estes conceitos aplicados pelo TCP além dos métodos de *fast-recovery* e *fast-retransmit* [57]. No processo de *fast-retransmit* três ou mais ACK's duplicados são recebidos, sendo este um forte indício de perda de segmentos, então o TCP realiza a retransmissão do segmento perdido sem aguardar um *timeout*. Neste caso, a fase de *congestion-avoidance* é iniciada em vez da fase de *slow-start*. Sendo este processo denominado de *fast-recovery*. Em caso de perdas de muitos pacotes e apenas alguns dos segmentos são reconhecidos, o TCP New Reno se mantém no processo de *fast-retransmit* para evitar uma redução muito grande na *cwnd*, solucionando assim alguns dos problemas do algoritmo TCP Reno.

Este algoritmo também melhorou o processo de *fast-recovery* implementado pelo TCP Reno, em que para cada ACK duplicado, este envia um pacote não-enviado do final da *cwnd* a fim de manter a janela totalmente utilizada [33]. De forma geral, o TCP New Reno foi desenvolvido para solucionar problemas de múltiplas perdas de pacotes que ocorriam no algoritmo TCP Reno.

Este método foi escolhido dentre outros algoritmos tradicionais de controle de congestionamento, por ser muito conhecido e largamente aplicado, sendo utilizado para fins de comparação da eficiência dos métodos DRL para o objetivo deste trabalho.

Capítulo 5

Sistemática para implementação de *deep learning* em rede *fronthaul* simulada

De forma a gerar o ambiente para uma investigação preliminar dos requerimentos de uma rede Fronthaul de arquitetura C-RAN, com todos os entraves inerentes à uma rede real, foram utilizados neste trabalho ferramentas que auxiliaram na constituição do ambiente de simulação, assim como na implementação dos métodos de controle de congestionamento que foram empregados para validar este ambiente. Portanto, neste capítulo serão apresentados os instrumentos e as metodologias adotadas para gerar análises iniciais acerca do controle de congestionamento neste ambiente simulado.

Segue a organização deste capítulo: A Seção 5.1 introduz o esquema para a constituição do ambiente de simulação para o controle de congestionamento e as ferramentas utilizadas em cada parte do *arcabouço*. A Seção 5.2 demonstra o modelo RL utilizado pelos agentes DRL para a definição de seus estados, ações e recompensas recebidas do ambiente simulado que contribuem para o aprendizado dos algoritmos. A Seção 5.3 descreve as arquiteturas e configurações adotadas para o aprendizado profundo dos métodos DRL.

5.1 Arcabouço e ferramentas para o estudo do controle de congestionamento

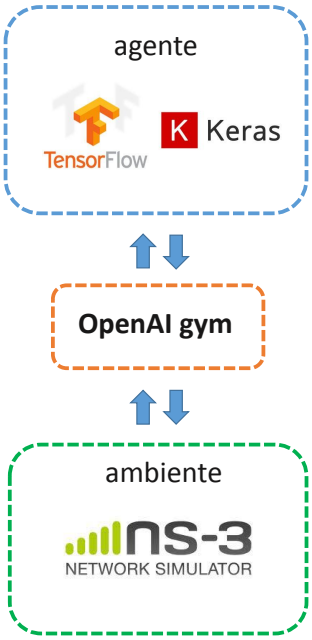
Nesta seção serão apresentadas as partes que compõe o arcabouço deste estudo e as ferramentas utilizadas para a integração de todas as funcionalidades.

5.1.1 Keras e Tensorflow

Keras [58] é uma biblioteca *open-source* escrita em Python que foi concebida com o objetivo de gerar a implementação de redes neurais profundas de forma que tenha interface amigável ao usuário, estrutura modular e extensível. Ela pode ser usada em conjunto com ferramentas computacionais matemáticas como o Tensorflow [59], que apresenta uma arquitetura baseada em grafos de fluxos de dados, os quais são denominados *tensors*. Através da flexibilidade proposta por essa ferramenta, é possível gerar a computação com múltiplas CPU's e GPU's, além também de incluir a integração com a ferramenta de visualização *Tensorboard*. Assim como o Keras, a biblioteca Tensorflow é escrita em linguagem Python.

Essas ferramentas dão suporte à implementação dos algoritmos DQN e A3C apresentados no capítulo anterior. Estas são utilizadas na primeira parcela do arcabouço que constitui o ambiente de simulação utilizado neste trabalho conforme é mostrado na Figura 5.1 a seguir.

Figura 5.1: Arcabouço para controle de congestionamento em rede FH.



Fonte: Autor.

Conforme apresentado na Figura 5.1, o arcabouço do estudo em questão é constituído de três partes distintas: a implementação do agente RL na forma dos algoritmos DQN e A3C, a ferramenta que interliga os agentes ao ambiente, e a rede Fronthaul que é o ambiente de rede

onde será realizado o controle de congestionamento. A seguir será apresentada a descrição e as características da ferramenta NS3-gym que constitui a segunda parte deste *arcabouço*.

5.1.2 NS3-gym

Essa ferramenta foi concebida com a proposta de realizar a integração entre o meio que fornece agentes RL, a biblioteca OpenAIgym [60], com o *software* de simulação de rede bem conhecido, o NS-3.

Esta ferramenta foi constituída com base nos seguintes princípios: escalabilidade, provendo a possibilidade de rodar múltiplas instâncias do NS-3; baixo *overhead* de entrada, possibilitando a conversão de *scripts* de simulação do NS-3 para serem usados em ambientes implementados pelo OpenAIgym; rápida prototipagem, permitindo a fácil depuração dos *scripts* de agentes locais; e fácil manutenção, pois a ferramenta é implementada como um módulo comum do NS-3 [61].

De forma geral, a interface permite a gerência do ciclo de vida da simulação no NS-3, assim como a comunicação e entrega dos estados e ações entre os agentes e o ambiente de simulação [61].

Os agentes dos algoritmos A3C e DQN conectam-se ao NS3-gym através de diferentes portas, enviando ações que irão regular o comportamento da janela de congestionamento, assim como recebem as observações e estados advindas do ambiente, que serão usadas para atualizações do processo de treinamento dos métodos.

5.1.3 NS-3 *Network Simulator*

O NS-3 é um simulador de redes de propósito geral que viabiliza a integração com *testbeds* e aplicações reais através de funcionalidades que dão suporte a diferentes tecnologias como LTE, Wifi e WiMAX [61]. Este *software* possui ampla aceitabilidade na comunidade acadêmica e pesquisas devido aos resultados relevantes comprovadamente obtidos em diversos problemas.

Neste trabalho, essa ferramenta é responsável por prover o ambiente de interação característicos de problemas RL. As configurações de rede utilizada para caracterizar uma rede FH e realizar os estudos de latência e vazão de pacotes será apresentada na seção seguinte.

De forma geral, a principal contribuição deste trabalho é a integração de ferramentas com diferentes propósitos de forma a criar o meio de simulação para a pesquisa em controle de congestionamento. Enviar as informações provindas do ambiente de forma instantânea para o

agente DRL utilizando a ferramenta NS3-gym representou um dos principais desafios encontrados, pois foi necessário estabelecer a conexão e mantê-la ao longo do tempo.

5.2 Modelo RL para o controle de congestionamento

Nesta seção serão apresentadas a formulação do modelo RL para o controle de congestionamento na rede FH. Serão apresentados os espaços de estados, as ações e a política de recompensa adotadas para esta aplicação.

5.2.1 Estados

As variáveis que constituem o espaço de estados de um dado problema, devem ser escolhidas a fim de expressar as condições do ambiente estudado, além de estarem diretamente relacionadas ao objetivo do agente. Um espaço de estado muito grande constitui um desafio maior ao algoritmo, aumentando de forma exponencial o espaço a ser explorado, assim como desacelera a convergência [25].

Sendo assim, o espaço de estados definido para o problema em questão possui apenas três observações, definidas de acordo com o proposto em [36] e [25]. A média móvel ponderada exponencialmente (do inglês,*Exponentially Weighted Moving Average - EWMA*) é considerada para três casos: **(1)** o intervalo de tempo entre dois pacotes enviados, **(2)** o intervalo de tempo entre o recebimento de dois *ACK's* consecutivos, **(3)** do RTT (do inglês, *Round Trip Time*), que mede o tempo decorrido entre o envio do pacote e a confirmação de recebimento por parte do receptor. Assumindo que estas variáveis podem assumir diversos valores, considera-se o espaço de estados muito grande e não-estacionário para este problema.

De forma geral, as variáveis que resumem as observações do ambiente podem ser vistas na Equação 5.1 a seguir:

$$< send_e wma, rec_e wma, rtt_e wma > \tag{5.1}$$

Os valores de $send_e wma$ caracterizam a taxa de envio pelo transmissor, enquanto que o valor de $rec_e wma$ indica o tráfego real pelo receptor, sendo estas duas variáveis importantes indicativos de congestionamento na rede, já que naturalmente $send_e wma$ seria igual a $rec_e wma$ em condições normais. Em caso de $send_e wma < rec_e wma$, há um grande indício da rede estar congestionada, sendo necessária a diminuição do fluxo de dados pelo transmissor.

Outras variáveis foram descartadas em [36] por não apresentarem melhora significativa de performance do algoritmo. Além disto, as variáveis da Equação 5.1 conseguem carregar informações relevantes que traduzem o histórico de observações do ambiente, sendo portanto, consideradas neste trabalho.

5.2.2 Ações

A política de ações considerada prevê três decisões distintas em relação ao tamanho da janela de congestionamento do TCP *(cwnd)*. São elas: **(1)**incrementar a *cwnd*, **(2)** decrementar a *cwnd*,e **(3)** manter a *cwnd*, conforme pode ser visto na Tabela 5.1 abaixo.

Tabela 5.1: Ações para a janela de congestionamento.

Mudança na cwnd	**Valor (bytes)**
Incrementar	50
Decrementar	-10
Manter	0

Fonte: Autor.

O incremento na *cwnd* adiciona 50 bytes ao tamanho atual da janela, assim como em caso de congestionamento a janela é diminuída em 10 bytes. As motivações para mudanças significativas no fluxo de dados através da janela dá-se pelo mesmo princípio proposto em [25], o qual visa estimular o agente a mudanças rápidas na *cwnd* utilizando de forma completa a largura de banda, mas dando a opção de redução da taxa de dados quando necessário.

Assim como em [25], as ações executadas pelo agente são tomadas a cada intervalo de tempo ($t_{interval}$) para que as ações do estado anterior possam agir em cada fluxo, diferentemente de algoritmos tradicionais, como o TCPNewReno, em que a *cwnd* é ajustada a cada ACK recebido.

5.2.3 Métrica de Recompensa

A métrica considerada neste trabalho leva em consideração o cálculo da função utilidade, que pondera a vazão dos pacotes em relação ao RTT, conforme é indicado em [25]. A equação

da Função Utilidade(U) pode ser vista a seguir.

$$U = \alpha \log(\text{throughput}) - \delta \log(\text{RTT}) \tag{5.2}$$

As variáveis α e δ agregam peso à vazão e RTT, respectivamente. A função acima sugere que todos os fluxos devem maximizar a vazão e minimizar o RTT. Quando são assumidos os valores de $\alpha = \delta = 1$, a função U é maximizada e a vazão proporcional(do inglês, *proportional throughput*) e o atraso justo (do inglês, *delay fairness*) são considerados [36], sendo portanto, a função objetivo escolhida como base para a atribuição de recompensas aos agentes.

A métrica testada leva em consideração o cálculo da vazão dos pacotes para um dado fluxo, sendo esta calculada como: $vazao = (bits/latencia(s))$. Este valor é tomado como parâmetro para a definição de pontuações para os fluxos que se mantiverem dentro de uma faixa de vazão proporcional à capacidade da rede FH dividida por todos os *links* de acesso. A Tabela 5.2 exemplifica as pontuações adotadas, de acordo com a faixa de vazão alcançada pelos fluxos da rede.

Tabela 5.2: Valores de recompensa para a segunda métrica de recompensas.

Condições	**Recompensa**
$x - 0.3 \leq v < x + 0.3$	50
$x - 0.6 < v < x - 0.3$	30
$v < x - 0.6$ ou $v > x + 0.3$	5

Fonte: Autor.

A variável v da Tabela 5.2 indica o valor de vazão alcançado pelo *link*, medida em megabits por segundo (Mbps), sendo o valor x a referência de vazão ótima, calculado da seguinte maneira: $x = (FHcapacity \div nlinks)$, em que $FHcapacity$ é a capacidade total da rede FH (em Mbps), e $nlinks$ a quantidade total de fluxos (RRH's) competindo para transmitir através da rede FH.

As recompensas e os limites que definem as faixas de vazão adotadas visam estimular o agente a manter constante o fluxo de dados ótimo, de modo que a largura de banda disponível seja bem distribuída entre os fluxos. Sendo assim, de acordo com que essa distribuição torna-se mais desproporcional, o valor da recompensa diminui. Em caso de perda de pacotes, o agente é penalizado com pontuação negativa (-50).

Assim como ocorre com o TCP, o controle da transmissão de dados ocorre nas bordas, neste caso, o agente será responsável por tomar as ações previstas sobre os enlaces para o ajuste de vazão necessário. Para aplicar a penalidade, o agente irá observar o enlace principal (fronthaul) para verificar se há perda de pacotes.

5.3 Arquitetura e configurações para o treinamento de métodos DRL

Nesta seção serão demonstradas as características mais intrínsecas das partes do esquema apresentado na Seção 5.1. Quais as arquiteturas e configurações adotadas e as motivações para tais escolhas.

5.3.1 Arquitetura das redes DQN e A3C

As configurações da rede DQN adotadas para o processo de treinamento são 3 camadas densas interconectadas, com 24 neurônios cada, com a função de ativação dos neurônios sendo a ReLU (*Rectified Linear Units*) [62]. Essa função é comumente aplicada em problemas de DL e retorna o valor zero para entrada negativa.

A última camada possui 3 neurônios, os quais representam o espaço de ações do problema. A função de ativação utilizada é a função linear. O algoritmo de otimização aplicado ao modelo da rede DQN é o Adam (do inglês, *Adaptative Moment Estimation*), o qual foi escolhido devido ao sua fácil implementação, eficiência computacional, baixo requerimento de memória e adequado para problemas de caráter não-estacionário [63]. Além disso, a taxa de aprendizado utilizada é $\alpha = 0.001$, tendo sido realizado testes que comprovaram que valores maiores afetaram diretamente a performance do algoritmo.

O modelo da rede global do método A3C possui 2 camadas densas interligadas a outras duas camadas de saída. A camada *Dense1* se conecta a camada de saída, a qual possui 3 neurônios, pois retorna o valor das probabilidades das ações para um dado estado de entrada. A camada *Dense2* interliga-se a rede que possui um único neurônio na camada de saída, pois esta gera a medida da função-valor ($V(s)$) para o estado de entrada. Cada uma das camadas densas possuem 100 neurônios, os quais são ativados pela função de ativação ReLU. Para o A3C, o algoritmo de otimização dos pesos da rede é computado por funções que calculam as perdas

da função-valor e da política, conforme demonstrado na Seção 4.3, os quais são usados para determinar o valor do gradiente que será utilizado para atualizar os pesos dos neurônios da rede global. A atualização da rede global ocorre a uma frequência de 10 interações com o ambiente ou quando o episódio termina. A Tabela 5.3 resume as configurações de entrada dos modelos das redes DQN e A3C.

Tabela 5.3: Arquitetura e configurações das redes DQN e A3C.

Estrutura	**DQN**	**A3C**
Camadas Densas	2	2
Número de neurônios por camada	24	100
Função de ativação	ReLU	ReLU
Camada de saída	1	2
Algoritmo de otimização	Adam	A3C gradiente

Fonte: Autor.

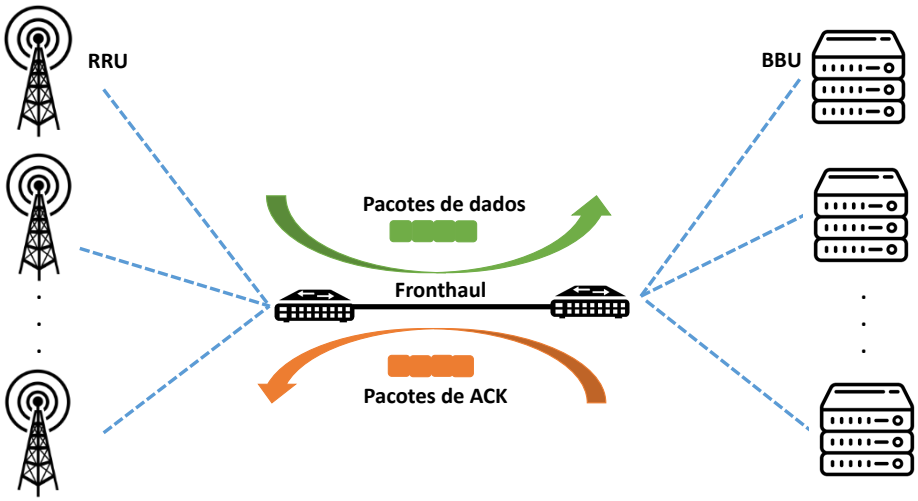
A escolha pelas configurações adotadas para as redes dos algoritmos foram baseadas em experimentações prévias com ambientes providos por bibliotecas como o OpenAIGym, devido ao bom desempenho alcançado, estas foram aceitas para serem testadas no ambiente de simulação da rede FH.

5.3.2 Configurações do ambiente de rede FH simulada

Conforme já informado anteriormente, o NS-3 é o meio pelo qual o ambiente de interação dos agentes é implementado. Através deste, a topologia de rede *Dumbbell* é empregada, onde é assumido que o link principal representa a rede FH, e os fluxos transmissores representam o

conjunto das RRH's, sendo os pares receptores, o conjunto de BBU's, separados por apenas 1 salto de distância, conforme mostra a Figura 5.2.

Figura 5.2: Visão geral da topologia de rede.



Fonte: Autor.

Para esta topologia, são considerados diferentes números de fluxos competindo para transmitir através da rede FH. As simulações foram geradas com 4, 6, 8 e 10 fluxos sendo controlados pelos agentes DQN e A3C conforme as regras apresentadas na Seção 5.2. Além disso, as configurações do cenário as quais os fluxos estão submetidos estão resumidos na Tabela 5.4 abaixo.

As interações dos agentes DQN e A3C com o ambiente que caracteriza a rede FH são baseadas no tempo, por isso cada episódio terá um tempo limite, sendo este 100 segundos conforme a tabela acima. Além disso, o agente interage com a *cwnd* a cada 300 milissegundos, recebendo as observações do ambiente para atualização de sua rede global. As configurações para a rede de topologia Dumbbell, quanto aos valores referentes aos enlaces de acesso (RRH) e rede FH, foram escolhidas em similaridade ao que é proposto em [37] para possível comparação de resultados.

É relevante ressaltar que a geração de tráfego para cada RRH é ordenada através de intervalos de tempos definidos por variáveis aleatórias, onde uma variável define o tempo de

Tabela 5.4: Configurações do cenário de simulação para a topologia Dumbbell.

Configurações do cenário	**Valor**
largura de banda (FH)	8 Mbps
latência (FH)	80 ms
Taxa de dados	16 Mbps
largura de banda dos enlaces	8 Mbps
latência dos enlaces	20 ms
time step de simulação	300 ms
duração total de simulação	100 s

Fonte: Autor.

transmissão (período *on*), sendo o tráfego definido pela taxa de dados e o tamanho dos pacotes, e o período em que a RRH irá aguardar para transmitir (período *off*). Em uma das experimentações, cada uma das RRH's terá valores de tempo de transmissão variáveis *on*/*off*, durante o tempo de simulação.

Capítulo 6

Resultados das experimentações em Controle de Congestionamento

Neste capítulo serão apresentados os resultados alcançados pelos agentes DRL comparados ao algoritmo de controle de congestionamento TCP New Reno em dois experimentos.

Segue a organização deste capítulo: A Seção 6.1 apresenta os resultados para dois experimentos em controle de congestionamento. O primeiro experimento foi realizado em 3 cenários diferentes para verificar o comportamento dos agentes DRL em comparação ao algoritmo TCP New Reno. Além disso, o segundo experimento realizado leva em consideração um cenário com maior dinamicidade e com maior atraso de propagação imposto a rede FH a fim de comparar o desempenho dos métodos. A Seção 6.2 aborda uma discussão acerca dos resultados apresentados na literatura recente em comparação aos resultados encontrados nos experimentos realizados com o arcabouço proposto.

6.1 Resultados para DL aplicado ao controle de congestionamento

Foram realizados dois experimentos distintos para verificar o desempenho dos agentes DRL com o algoritmo New Reno na tarefa de controle de congestionamento. A principal diferença entre os dois consiste no intervalo de tempo de transmissão entre os fluxos, sendo utilizada uma variável aleatória no início de cada simulação para estipular por quanto tempo cada fluxo ficará ativo transmitindo dados (período *on*), e quanto tempo irá aguardar para enviar novamente (período *off*).

No experimento I, cada fluxo irá transmitir por 1 segundo e entrar no período *off* por mais 1 segundo, permanecendo nesta dinâmica até o fim do episódio. No experimento II, o intervalo de tempo do período *on* e *off* varia ao longo da simulação e estes valores são definidos de forma aleatória para cada um dos fluxos.

Além disso, o experimento I é constituído de 3 cenários diferentes, os quais representam variações no índice de atraso de propagação para ambientes onde a quantidade de usuários varia. Os atrasos de propagação impostos sobre a rede são de 20 ms, 60 ms e 80 ms testados na topologia Dumbbell, com variação de fluxos em 4, 6, 8 e 10 RRHs, de acordo com as configurações de rede apresentadas na Seção 5.3. Para todos os cenários, foram realizadas simulações com 10 *seeds* para cada número diferente de fluxos, obtendo duração total de 1000 s em cada simulação.

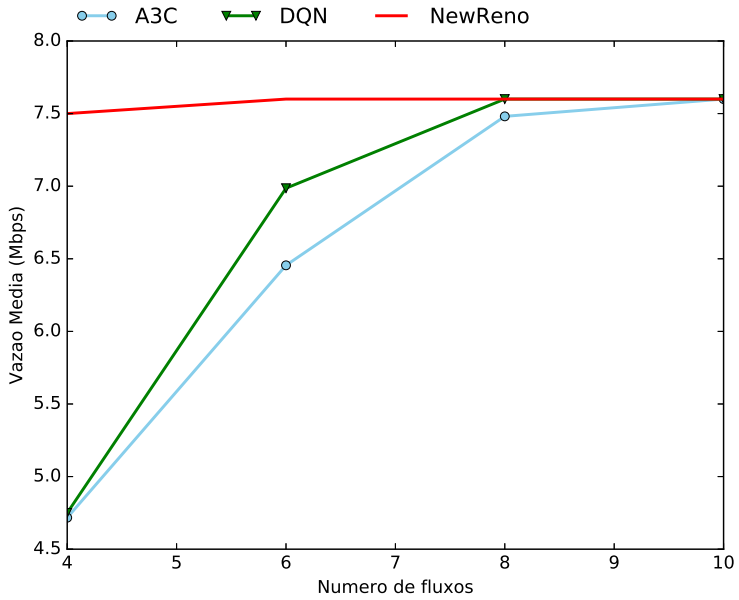
Para testar o comportamento dos algoritmos em ambiente distinto, um pouco mais "dinâmico", no experimento II será avaliado o desempenho dos métodos em um cenário com maior atraso de propagação aplicado a rede FH (100 ms), gerando simulações com 10 seeds para 4, 8 e 10 fluxos, com duração total de 1000 s para cada simulação.

6.1.1 Experimento I

No primeiro cenário, com menor atraso de propagação imposto sobre o link FH, o algoritmo TCP New Reno (NR) apresentou melhor desempenho quanto à média de vazão obtida, independente da quantidade de fluxos competindo para a transmissão, conforme mostra a Figura 6.1. É possível verificar que os agentes A3C e DQN aumentaram a utilização do link FH gradativamente com o aumento de usuários (RRH's) utilizando os 8 Mbps de largura de banda disponível na rede.

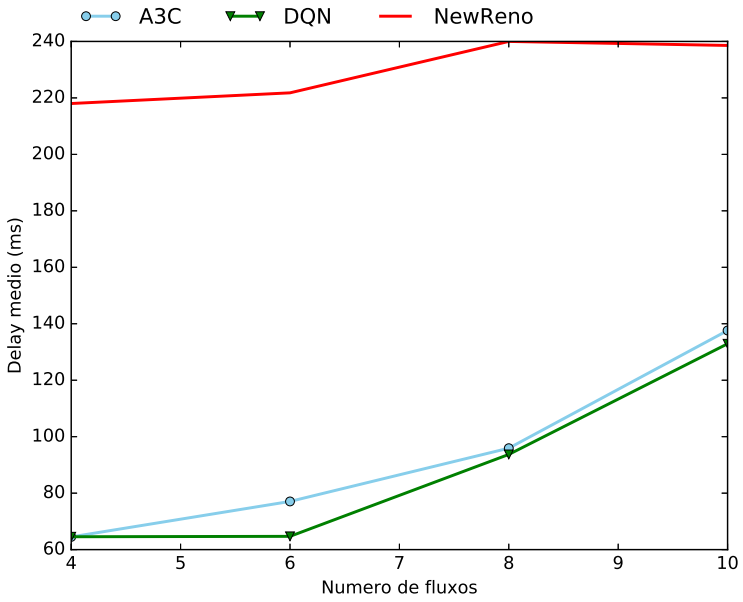
Apesar de uma maior média de utilização da largura de banda da rede, o algoritmo NR apresentou uma média de perda de pacotes significativa em relação aos demais agentes, conforme ilustra a Figura 6.3. O atraso médio dos pacotes é calculado através da soma dos atrasos dividido pela quantidade de pacotes enviados. Este atraso possui relação direta com a quantidade de pacotes perdidos como é demonstrado na Figura 6.1.1, pois com alta latência é natural que pacotes sejam descartados pela rede, sendo um indício de inabilidade do algoritmo NR no controle de congestionamento.

Figura 6.1: Vazão média variando o número de usuários para o cenário 1.



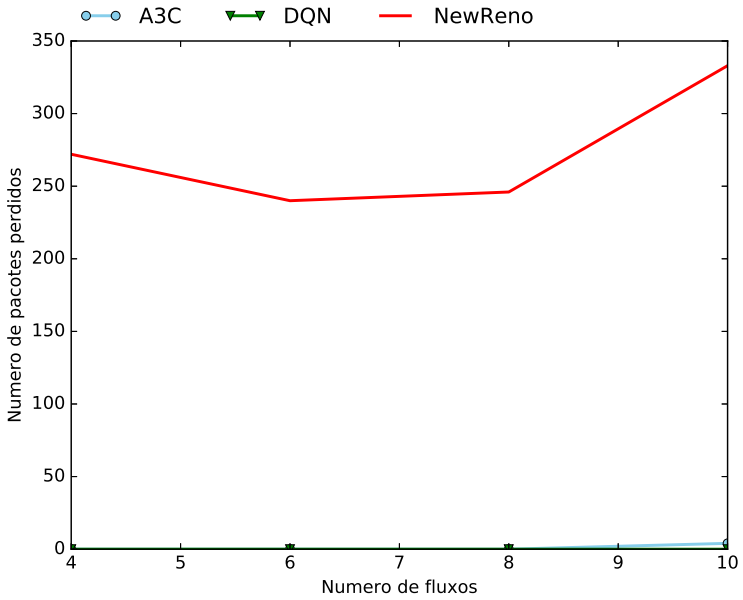
Fonte: Autor.

Figura 6.2: Atraso médio dos pacotes variando o número de fluxos para o cenário 1.



Fonte: Autor.

Figura 6.3: Média da perda de pacotes variando o número de fluxos para o cenário 1.



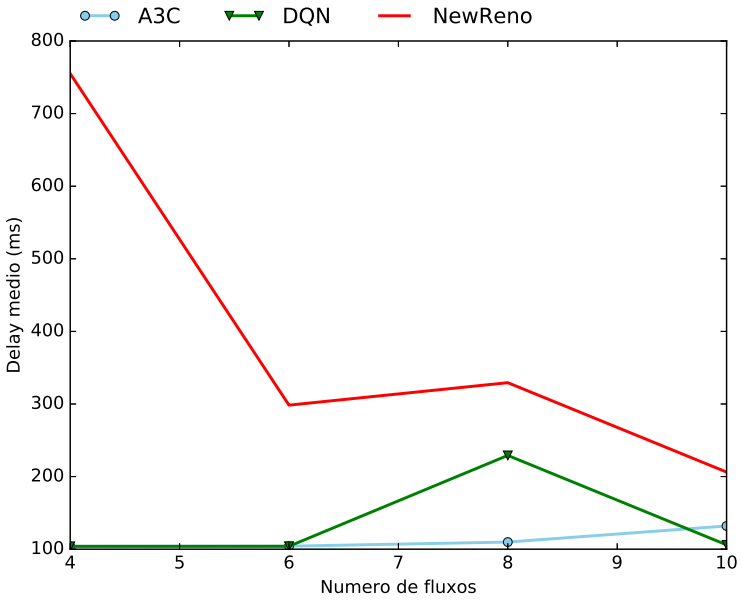
Fonte: Autor.

Conforme ilustrado na Figura 6.3, a média de pacotes perdidos pelos agentes A3C e DQN foram próximos a zero, mesmo com o aumento da quantidade de fluxos. Esta característica é reflexo da política adotada pelos agentes, a qual prevê alta punição (-50) para ações que geram altos índices de perdas de pacotes na rede. Sendo assim, é possível considerar que em termos de tráfego útil, os agentes A3C e DQN tornam-se mais efetivos levando em consideração boa taxa de vazão dos fluxos e baixos níveis de latência.

No cenário 2, onde é aplicado sobre a rede FH um atraso de propagação de 60 ms, é possível verificar que o atraso médio dos pacotes para o algoritmo NR é aproximadamente 8 vezes maior que o esperado para uma quantidade pequena de fluxos, levando em consideração o atraso dos links de acesso somado ao atraso de propagação da rede FH (RRHlink:20 ms + FH:60 ms + BBUlink:20 ms). Os agentes A3C e DQN mantiveram a latência da rede em níveis satisfatórios, mantendo a mesma taxa de perda de pacotes do cenário 1, com baixa variação para diferentes fluxos conforme demonstrado pela Figura 6.4.

Conforme discutido na Seção 5.2, a métrica de recompensas adotada para os agentes A3C e DQN busca priorizar a justiça na transmissão de dados entre os fluxos, gratificando as ações que mantiverem a proporcionalidade da vazão distribuída entre a quantidade de RRH's

Figura 6.4: Atraso médio dos pacotes variando o número de fluxos para o cenário 2.



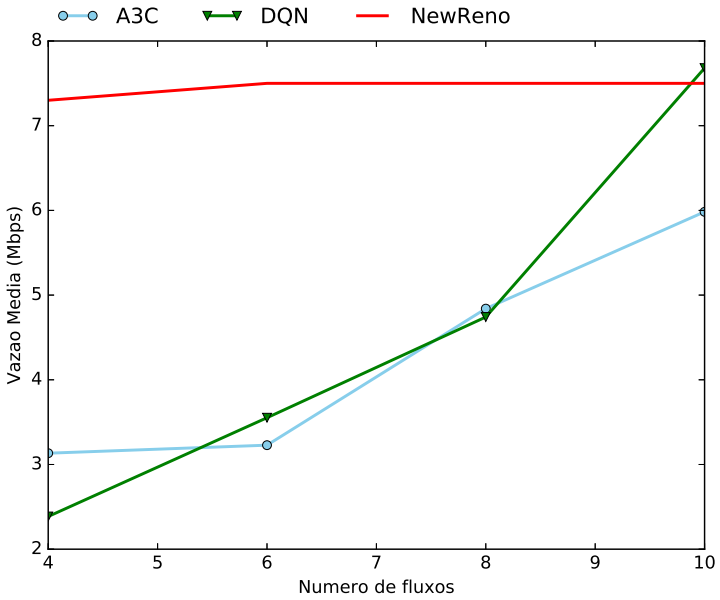
Fonte: Autor.

disponíveis.

No terceiro cenário, no qual é aplicado o atraso de propagação de 80 ms, é possível verificar que o algoritmo NR possui média constante de vazão para diferentes quantidades de fluxo, enquanto que os agentes A3C e DQN apresentam um aumento gradativo de vazão de acordo com o crescimento no número de fluxos. Para 10 fluxos, o agente DQN alcançou valor médio de vazão ainda maior que o algoritmo NR, conforme pode ser visto na Figura 6.5.

A política implementada prevê um crescimento gradual da janela de congestionamento afim de evitar grandes filas nos roteadores e, consequentemente o descarte de pacotes. Segundo as regras do TCP que regem o algoritmo New Reno, antes que este alcance o limiar da primeira fase (*slow start*), a janela *cwnd* duplica a cada confirmação de segmento recebido, provocando um aumento significativo no *buffer* do roteador, refletido na quantidade de pacotes perdidos em todos os cenários.

Figura 6.5: Vazão média para diferentes fluxos do cenário 3.



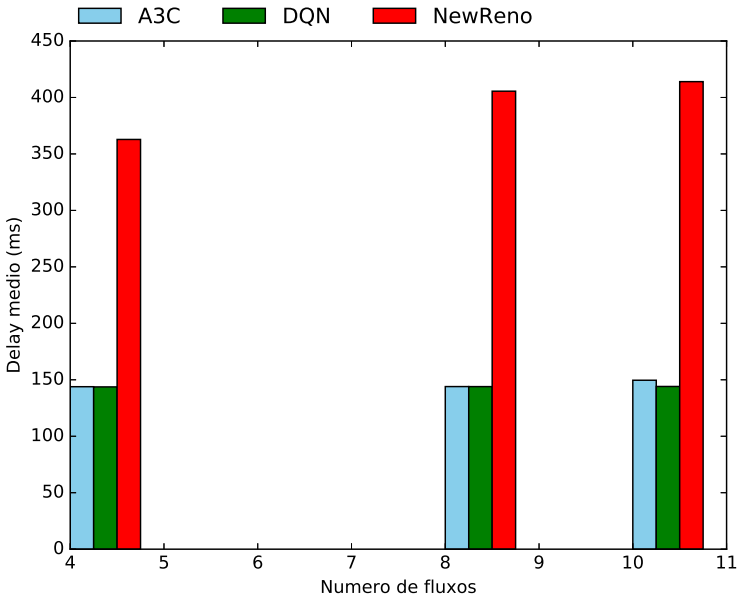
Fonte: Autor.

6.1.2 Experimento II

Neste cenário, com maior atraso de propagação aplicado ao link principal, acrescido de variabilidade dos períodos *on/off* de cada fluxo, foi assumido que há diferentes aplicações e, portanto, diferentes demandas das RRH's representadas pelos fluxos.

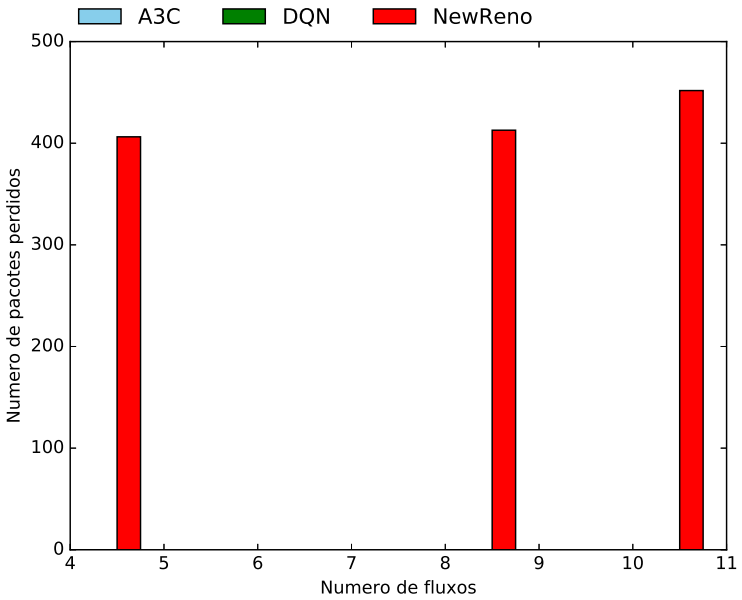
Devido à necessidade do frequente ajuste da janela de congestionamento, a Figura 6.6 e a Figura 6.7 demonstram a inabilidade do algoritmo NR em ajustar a vazão nos links de acesso, mantendo constante a alta taxa de transmissão dos pacotes, sendo refletidos no aumento da latência e, consequentemente, em uma substancial perda de pacotes. É possível também verificar que o valor de latência e descarte de dados aumenta com acréscimo de usuários no cenário.

Figura 6.6: Atraso médio dos fluxos para o experimento II.



Fonte: Autor.

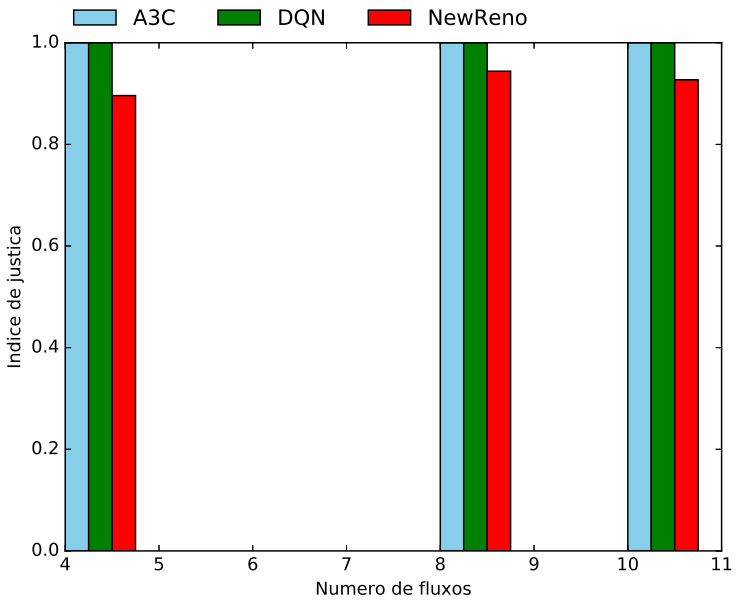
Figura 6.7: Média de perda de pacotes para o experimento II.



Fonte: Autor.

O algoritmo NR apresenta dificuldade de adaptação quando parâmetros da rede variam em curtos intervalos de tempo, assim como também foi demonstrado em [25]. Isso se dá devido a regras fixas impostas ao algoritmo, diferentemente do que ocorre com os agentes DRL, os quais possuem a habilidade de controlar as mudanças na janela *cwnd* através da escolha de ações efetivas, trazendo respostas rápidas a mudanças no ambiente. Além disso, os algoritmos DRL conseguem manter uma alta taxa de justiça entre as diferentes aplicações, mesmo com alta variabilidade dos tempos de transmissão entre os fluxos, conforme é indicado na Figura 6.8.

Figura 6.8: Índice de justiça Jain aplicado aos fluxos para o experimento II.



Fonte: Autor.

Os experimentos realizados são preliminares. Mas assumindo o escopo restrito dos mesmos, os resultados do aprendizado por reforço são promissores. A memória *replay* do agente DQN e o cálculo de gradientes utilizando informações advindas do ambiente de simulação a cada iteração, para a otimização da rede neural profunda do agente A3C, revelam a capacidade de adaptação destes métodos a mudanças repentinas no ambiente, obtendo potencial relevante para atender aos requerimentos impostos pelos sistemas 5G, onde baixos níveis de latência e distribuição justa de recursos entre as diferentes aplicações são esperados.

6.2 Discussões de resultados acerca da literatura

Trabalhos recentes encontrados na literatura, que utilizam métodos de DL para o controle de congestionamento usando topologia Dumbbell, realizam análises de performance em cenários que consideram apenas vazão e latência a fim de realizar comparações com métodos tradicionais, como por exemplo é realizado em [37] e [25]. Análises mais profundas são necessárias para indicar de forma mais precisa índices relevantes como tráfego útil da rede, a taxa de atrasos de pacotes e média de pacotes perdidos, a fim de avaliar o desempenho dos métodos propostos.

Além disso, em [37], simulações sem variabilidade dos aspectos do ambiente são realizadas a fim de testar a eficácia do método DRL proposto. Mesmo utilizando configurações bastante similares de cenários, os agentes DQN e A3C apresentaram resultados superiores para os índices de vazão média dos fluxos e a taxa de atrasos em relação ao apresentado em [37]. Além do mais, análises que constituem cenários com diferentes atrasos de propagação não são consideradas para quantidade de fluxos distintas.

Em relação ao que foi realizado em [25], a metodologia adotada utiliza uma função de aproximação para o mapeamento do espaço de estados do problema (protótipos), juntamente com métodos RL tabulares para a definição de sua política de ações, o que afeta diretamente a velocidade de convergência do algoritmo. Nos experimentos realizados neste trabalho, os agentes DRL testados alcançaram rápida convergência, mesmo sendo realizado o cálculo de entropia da política para evitar convergência prematura, conforme indicado em [55].

Trabalhos iniciais com a aplicação de princípios RL, como o que foi proposto em [36], necessitam de longo processo de treinamento e alto poder computacional, para mapear todas as possíveis condições da rede. Neste trabalho apresentou-se resultados que demonstraram desempenho satisfatório com relativo baixo custo computacional dos agentes DRL, sem a necessidade de extenso processo de treino *offline* de [36].

De forma geral, os resultados aqui apresentados, realizados sob condições similares ao que foi proposto na literatura recente, demonstraram bom desempenho e grande potencial de exploração para soluções com DRL para combater limitações apresentadas pelos métodos tradicionais de controle de congestionamento.

Capítulo 7

Conclusão

Esta dissertação abordou a aplicação de métodos de *Deep Learning* em duas áreas relevantes da comunicação: classificação de modulações usadas em redes móveis e outras, e o estabelecimento de um arcabouço para pesquisas em controle de congestionamento usando DRL, a partir da configuração de ambiente de simulação baseado no simulador NS-3.

Uma das principais contribuições deste trabalho é a produção e disponibilização da base de dados UFPATelecom, composta de dados gerados com diferentes canais de propagação, aplicáveis em experimentos de aprendizado de máquina em telecomunicações, relevantes não somente para a pesquisa, mas também para o ensino.

Além disto, foi investigada a performance de métodos de classificação tradicionais juntamente com uma rede neural convolucional em condições de treino e teste incompatíveis, afim de realizar comparações quanto aos índices de acurácia, velocidade de convergência e custo computacional. Através dos experimentos realizados foi possível concluir que o desempenho de muitos classificadores são altamente dependentes da escolha adequada de características de entrada.

Ademais, as saídas dos classificadores revelaram que estes podem alcançar performance similar a alcançada por CNN quando treinados com dados divergentes. Por fim, os resultados revelaram que dados de diferentes canais são muito importantes para a avaliação do aprendizado de máquina aplicado à Telecomunicações.

No âmbito do controle de congestionamento foi considerada a aplicação de métodos de *Deep Reinforcement Learning* para validar o ambiente de simulação proposto, sendo este composto pelos agentes DRL e a configuração de rede, integrada pela ferramenta NS3Gym. Nos experimentos de validação, os resultados comprovaram a habilidade dos agentes DRL em man-

ter uma distribuição justa na transmissão de dados entre os links de acesso, além de alcançarem índices médios de vazão satisfatórios quando comparados com os trabalhos recentemente publicados na área.

Além disso, foi verificado que o algoritmo NR consegue manter alta utilização da largura de banda do link principal, mas em contrapartida, apresentou considerável índice de atrasos e perdas de pacotes demonstrando ineficiência no controle de congestionamento em comparação aos agentes DRL testados nos cenários propostos.

Sendo assim, é possível concluir que os métodos DRL são merecedores de maior investigação, pois possuem grande potencial para atender aos requerimentos de redes FH que exigem baixa latência e a rápida adaptação à mudanças, devido ao já citado alto tráfego e a dinamicidade característica de sistemas 5G.

7.1 Trabalhos Futuros

As extensões consideradas para este trabalho incluem a incorporação de dados de canais de propagação diferentes a base de dados UFPATelecom, além da incorporação de outras modulações. Com a metodologia desenvolvida, pode-se também incluir sinais GSM e LTE, sempre no propósito de tornar a base a mais rica possível. Com bases mais diversas, pode-se então expandir as análise realizadas, encontrar limitações, e eventualmente motivação para investigação de novos métodos de DL para a classificação de modulação e tecnologias de acesso.

É também planejada a realização de simulações com métodos DRL e outros algoritmos de controle de congestionamento, em ambientes de simulação com cenários mais dinâmicos e topologias de rede mais complexas a fim de expandir a investigação preliminar realizada e trazer novos dados e análises quanto aos requerimentos da rede FH em sistemas 5G.

7.2 Publicações

Os resultados referentes à CAM foram submetidos ao Simpósio Brasileiro de Telecomunicações (SBrT) no ano de 2018, através do artigo “Deep Learning in RAT and Modulation Classification with a New Radio Signals Dataset”, o qual foi aprovado. O ambiente de simulação para o controle de congestionamento e os resultados dos experimentos de validação do mesmo estão sendo preparados para a submissão em conferência relevante para a área ainda

este ano.

Bibliografia

[1] O. A. Dobre, "Signal identification for emerging intelligent radios: Classical problems and new challenges," *IEEE Instrumentation & Measurement Magazine*, vol. 18, no. 2, pp. 11–18, 2015.

[2] B. Tang, Y. Tu, Z. Zhang, and Y. Lin, "Digital signal modulation classification with data augmentation using generative adversarial nets in cognitive radio networks," *IEEE Access*, vol. 6, pp. 15 713–15 722, 2018.

[3] D. Grimaldi, S. Rapuano, and L. De Vito, "An automatic digital modulation classifier for measurement on telecommunication networks," *IEEE Transactions on Instrumentation and Measurement*, vol. 56, no. 5, pp. 1711–1720, Oct 2007.

[4] S. Baban, D. Denkoviski, O. Holland, L. Gavrilovska, and H. Aghvami, "Radio access technology classification for cognitive radio networks," in *2013 IEEE 24th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*. IEEE, 2013, pp. 2718–2722.

[5] H. Cao, W. Jiang, M. Wiemeler, T. Kaiser, and J. Peissig, "A robust radio access technology classification scheme with practical considerations," in *2013 IEEE 24th International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC Workshops)*. IEEE, 2013, pp. 36–40.

[6] O. A. Dobre, R. Venkatesan, D. C. Popescu *et al.*, "Second-order cyclostationarity of mobile wimax and lte ofdm signals and application to spectrum awareness in cognitive radio systems," *IEEE Journal of Selected Topics in Signal Processing*, vol. 6, no. 1, pp. 26–42, 2012.

[7] Y. A. Eldemerdash, O. A. Dobre, O. Üreten, and T. Yensen, "Identification of cellular networks for intelligent radio measurements," *IEEE Transactions on Instrumentation and Measurement*, vol. 66, no. 8, pp. 2204–2211, 2017.

[8] B. Kim, J. Kim, H. Chae, D. Yoon, and J. W. Choi, "Deep neural network-based automatic modulation classification technique," in *2016 International Conference on Information and Communication Technology Convergence (ICTC)*, Oct 2016, pp. 579–582.

[9] A. Swami and B. M. Sadler, "Hierarchical digital modulation classification using cumulants," *IEEE Transactions on communications*, vol. 48, no. 3, pp. 416–429, 2000.

[10] A. Ali and F. Yangyu, "Automatic modulation classification using principle composition analysis based features selection," in *2017 Computing Conference*, July 2017, pp. 294–296.

[11] T. J. O'Shea, J. Corgan, and T. C. Clancy, "Convolutional radio modulation recognition networks," in *International conference on engineering applications of neural networks*. Springer, 2016, pp. 213–226.

[12] X. Liu, D. Yang, and A. El Gamal, "Deep neural network architectures for modulation classification," in *2017 51st Asilomar Conference on Signals, Systems, and Computers*. IEEE, 2017, pp. 915–919.

[13] J. H. Lee, K. Kim, and Y. Shin, "Feature image-based automatic modulation classification method using cnn algorithm," in *2019 International Conference on Artificial Intelligence in Information and Communication (ICAIIC)*, Feb 2019, pp. 1–4.

[14] F. Akyön, Y. K. Alp, G. Gök, and O. Arıkan, "Deep learning in electronic warfare systems: Automatic intra-pulse modulation recognition," in *2018 26th Signal Processing and Communications Applications Conference (SIU)*, May 2018, pp. 1–4.

[15] J. Sun, G. Wang, Z. Lin, S. G. Razul, and X. Lai, "Automatic modulation classification of cochannel signals using deep learning," in *2018 IEEE 23rd International Conference on Digital Signal Processing (DSP)*, Nov 2018, pp. 1–5.

[16] F. Meng, P. Chen, L. Wu, and X. Wang, "Automatic modulation classification: A deep learning enabled approach," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 11, pp. 10 760–10 772, Nov 2018.

[17] A. Ali and F. Yangyu, "Automatic modulation classification using deep learning based on sparse autoencoders with nonnegativity constraints," *IEEE Signal Processing Letters*, vol. 24, no. 11, pp. 1626–1630, Nov 2017.

[18] C. Teng, C. Liao, C. Chen, and A. A. Wu, "Polar feature based deep architectures for automatic modulation classification considering channel fading," in *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, Nov 2018, pp. 554–558.

[19] D. Li-Da, W. Shi-Lian, and Z. Wei, "Modulation classification of underwater acoustic communication signals based on deep learning," in *2018 OCEANS - MTS/IEEE Kobe Techno-Oceans (OTO)*, May 2018, pp. 1–4.

[20] S. Huang, Y. Jiang, Y. Gao, Z. Feng, and P. Zhang, "Automatic modulation classification using contrastive fully convolutional network," *IEEE Wireless Communications Letters*, pp. 1–1, 2019.

[21] S. Rajendran, W. Meert, D. Giustiniano, V. Lenders, and S. Pollin, "Deep learning models for wireless signal classification with distributed low-cost spectrum sensors," *IEEE Transactions on Cognitive Communications and Networking*, vol. 4, no. 3, pp. 433–445, Sep. 2018.

[22] C. Kuilin and D. Ran, "C-ran the road towards green ran," *China Mobile Research Institute, White Paper*, 2011.

[23] C. P. R. I. CPRI, "Interface specification (v7. 0)," Technical report, Tech. Rep., 2015.

[24] A. Ericsson and N. Huawei, "Nokia, ecpri presentation."

[25] W. Li, F. Zhou, K. R. Chowdhury, and W. M. Meleis, "Qtcp: Adaptive congestion control with reinforcement learning," *IEEE Transactions on Network Science and Engineering*, 2018.

[26] W. Obile, "Ericsson mobility report," *Nov*, 2016.

[27] C. Mobile, "C-ran: The road towards green ran, white paper, ver. 2.5," *China Mobile Research Institute*, 2011.

[28] H. Jung, "Cisco visual networking index: global mobile data traffic forecast update 2010–2015," Technical report, Cisco Systems Inc, Tech. Rep., 2011.

[29] Y. Qi, M. Z. Shakir, M. A. Imran, A. Quddus, and R. Tafazolli, "How to solve the fronthaul traffic congestion problem in h-cran?" in *2016 IEEE international conference on communications workshops (ICC)*. IEEE, 2016, pp. 240–245.

[30] B. Guo, W. Cao, A. Tao, and D. Samardzija, "Lte/lte-a signal compression on the cpri interface," *Bell Labs Technical Journal*, vol. 18, no. 2, pp. 117–133, 2013.

[31] J. Bartelt, P. Rost, D. Wubben, J. Lessmann, B. Melis, and G. Fettweis, "Fronthaul and backhaul requirements of flexibly centralized radio access networks," *IEEE Wireless Communications*, vol. 22, no. 5, pp. 105–111, 2015.

[32] C.-Y. Chang, N. Nikaein, and T. Spyropoulos, "Impact of packetization and scheduling on c-ran fronthaul performance," in *2016 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 2016, pp. 1–7.

[33] J. Sing and B. Soh, "Tcp new vegas: improving the performance of tcp vegas over high latency links," in *Fourth IEEE International Symposium on Network Computing and Applications*. IEEE, 2005, pp. 73–82.

[34] S. Floyd, T. Henderson, and A. Gurtov, "The newreno modification to tcp's fast recovery algorithm," Tech. Rep., 2004.

[35] A. Sivaraman, K. Winstein, P. Thaker, and H. Balakrishnan, "An experimental study of the learnability of congestion control," in *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 4. ACM, 2014, pp. 479–490.

[36] K. Winstein and H. Balakrishnan, "Tcp ex machina: computer-generated congestion control," 2013.

[37] K. Xiao, S. Mao, and J. K. Tugnait, "Tcp-drinc: Smart congestion control based on deep reinforcement learning," *IEEE Access*, 2019.

[38] Y. Nakayama, D. Hisano, T. Kubo, T. Shimizu, H. Nakamura, J. Terada, and A. Otaka, "Low-latency routing for fronthaul network: A monte carlo machine learning approach," in *2017 IEEE International Conference on Communications (ICC)*. IEEE, 2017, pp. 1–6.

[39] Z. Wu, N. M. Khan, L. Gao, and L. Guan, "Deep reinforcement learning with parameterized action space for object detection," in *2018 IEEE International Symposium on Multimedia (ISM)*. IEEE, 2018, pp. 101–104.

[40] M. Lu and X. Li, “Deep reinforcement learning policy in hex game system,” in *2018 Chinese Control And Decision Conference (CCDC)*, June 2018, pp. 6623–6626.

[41] T. J. O’shea and N. West, “Radio machine learning dataset generation with gnu radio,” in *Proceedings of the GNU Radio Conference*, vol. 1, no. 1, 2016.

[42] T. J. O’Shea, T. Roy, and T. C. Clancy, “Over-the-air deep learning based radio signal classification,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 1, pp. 168–179, 2018.

[43] J. Shen, R. Pang, R. J. Weiss, M. Schuster, N. Jaitly, Z. Yang, Z. Chen, Y. Zhang, Y. Wang, R. Skerrv-Ryan *et al.*, “Natural tts synthesis by conditioning wavenet on mel spectrogram predictions,” in *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2018, pp. 4779–4783.

[44] “UFPAtelecom Dataset.” [Online]. Available: https://www.lasse.ufpa.br/UFPAtelecom/

[45] “VSA 89600.” [Online]. Available: https://www.keysight.com/br/pt/software/application-sw/89600-vsa-software.html

[46] “Open Source Mobile Communications.” [Online]. Available: https://osmocom.org/

[47] “LTE System Toolbox.” [Online]. Available: https://www.mathworks.com/products/lte-system.html

[48] E. U. T. R. A. Network, “3rd generation partnership project; technical specification group services and system aspects; general packet radio service (gprs) enhancements for evolved universal terrestrial radio access network (e-utran) access,” 2011.

[49] A. Géron, *Hands-on machine learning with Scikit-Learn and TensorFlow: concepts, tools, and techniques to build intelligent systems*. "O’Reilly Media, Inc.", 2017.

[50] J. D. Rennie, L. Shih, J. Teevan, and D. R. Karger, “Tackling the poor assumptions of naive bayes text classifiers,” in *Proceedings of the 20th international conference on machine learning (ICML-03)*, 2003, pp. 616–623.

[51] K. Lau, M. Salibian-Barrera, and L. Lampe, “Modulation recognition in the 868 mhz band using classification trees and random forests,” *AEU-International Journal of Electronics and Communications*, vol. 70, no. 9, pp. 1321–1328, 2016.

[52] R. S. Sutton and A. G. Barto, "Reinforcement learning: An introduction," 2011.

[53] L. Deng, D. Yu *et al.*, "Deep learning: methods and applications," *Foundations and Trends® in Signal Processing*, vol. 7, no. 3–4, pp. 197–387, 2014.

[54] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[55] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *International conference on machine learning*, 2016, pp. 1928–1937.

[56] A. S. Tanenbaum *et al.*, "Computer networks, 4-th edition," *ed: Prentice Hall*, 2003.

[57] W. R. Stevens, "Tcp slow start, congestion avoidance, fast retransmit, and fast recovery algorithms," 1997.

[58] F. Chollet *et al.*, "Keras," 2015.

[59] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: http://tensorflow.org/

[60] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," *arXiv preprint arXiv:1606.01540*, 2016.

[61] P. Gawłowicz and A. Zubow, "ns3-gym: Extending openai gym for networking research," *arXiv preprint arXiv:1810.03943*, 2018.

[62] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.

[63] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *arXiv preprint arXiv:1412.6980*, 2014.