Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Classificador baseado em Redes Neurais Convolucionais para algoritmos RMLSA em EONs

Luiz Filipe de Andrade Santos

Monografia apresentada como requisito parcial
para conclusão do Curso de Engenharia da Computação

Orientador
Prof. Dr. André Costa Drummond

Brasília
2020

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Classificador baseado em Redes Neurais Convolucionais para algoritmos RMLSA em EONs

Luiz Filipe de Andrade Santos

Monografia apresentada como requisito parcial
para conclusão do Curso de Engenharia da Computação

Prof. Dr. André Costa Drummond (Orientador)
CIC/UnB

Prof. Dra. Roberta Barbosa Oliveira
Universidade de Brasília

Prof. Dr. Luís Paulo Faina Garcia
Universidade de Brasília

Prof. Dr. João José Costa Gondim
Coordenador do Curso de Engenharia da Computação

Brasília, 07 de Novembro de 2020

Dedicatória

Dedico a minha mãe, por todo suporte que recebi durante a elaboração desse trabalho e da minha formação acadêmica. Também aos amigos e minhas irmãs, que me incentivaram todos os dias e ofereceram apoio nos momentos difíceis".

Agradecimentos

Agradeço a Universidade de Brasília pela oportunidade de ter a formação no curso de engenharia de computação, fornecendo estrutura e professores capacitados. Agradeço também o professor Doutor André Costa Drummond e o professor Guilherme Eneas Silva pela oportunidade de poder participar desse projeto incrível que resultou nesse trabalho. E por último, e não menos importante, a minha mãe Maria Rosa Rios de Andrade por todo apoio nessa jornada, principalmente em tempos difíceis de pandemia.

Resumo

O advento das redes ópticas elásticas trouxeram uma flexibilidade e escalabilidade na alocação de espectro superior às tradicionais redes baseadas em multiplexação por divisão de comprimento de onda, podendo assim suportar a crescente demanda do tráfego da internet. Atualmente, os algoritmos de roteamento e alocação de espectro dessas redes utilizam estratégias fixas. A tendência natural de evolução desses algoritmos é a utilização de estratégias adaptativas, de acordo com o comportamento da rede. O primeiro passo para isso é checar se é possível extrair informações da rede óptica a partir do estado de alocação de espectro. Com esse objetivo em mente, foi criado um classificador baseado em redes neurais convolucionais profundas capaz de reconhecer em determinado momento qual heurística de roteamento e alocação de recursos RMLSA (*Routing, Modulation Level and Spectrum Assigment*) está sendo utilizada. O classificador proposto foi treinado com 11 algoritmos diferentes, e em 2 topologias diferentes de rede óptica elástica, alcançando uma acurácia superior a 90% em todos os casos. Os bons resultados obtidos mostram a viabilidade de generalizar o classificador para quaisquer algoritmos, sendo o passo inicial necessário para adoção de estratégias adaptativas.

Palavras-chave: Redes Ópticas Elásticas, Roteamento e Alocação de Espectro, Aprendizado Profundo, Redes Neurais Convolucionais

Abstract

The advent of elastic optical networks brought superior flexibility and scalability into the spectrum allocation compared to the traditional wavelength division multiplexing based networks, thus being able to support the growing demand of internet traffic. Currently, the routing and spectrum allocation algorithms of these networks use fixed strategies. The natural tendency of evolution of these algorithms is the use of adaptive strategies, according to the behavior of the network. The first step towards this is to check if it is possible to extract information from the optical network through spectrum allocation state. With this objective in mind, we created a deep convolutional neural network based classifier capable of recognizing at a given moment which routing and allocation RMLSA heuristic is being used. The proposed classifier was trained with 11 different algorithms, and with 2 different elastic optical network topologies, reaching an accuracy higher than 90 % in all cases. The good results obtained show the feasibility of generalizing the classifier for any algorithms, being the necessary initial step for the adoption of adaptive strategies.

Keywords: Elastic Óptical Networks, Convolutional Neural Networks, Deep Learning, Routing and Spectrum Allocation, RMLSA

Sumário

Lista de Figuras

Lista de Tabelas

Capítulo 1

Introdução

Nos últimos anos, o tráfego de Internet nas redes de núcleo tem dobrado a cada dois anos, e estudos fazem previsões que indicam que continuará a apresentar um crescimento exponencial devido à aplicações emergentes, como por exemplo a popularização dos serviços de *streaming* e mídias de alta definição [1]. Devido a esse rápido aumento nas demandas de tráfego por diferentes aplicações, nasceu a necessidade, para o futuro das redes ópticas, de sistemas de transmissão de fibra óptica de grande capacidade e custo-benefício. Além disso, está previsto que as redes ópticas serão necessárias para suportar a classe de Terabytes de transmissão em um futuro próximo [2]. Juntamente com esse ambiente das aplicações da Internet emergentes, existem também aplicações que demonstram mudanças imprevisíveis na largura de banda e nos padrões de tráfego geográfico, como a Televisão de Protocolo da Internet (IPTV - *Internet Protocol television*), vídeo sobre demanda e armazenamento em nuvem [3]. Para alcançar essas necessidades do futuro da Internet, a transmissão óptica e as tecnologias de rede estão caminhando para uma direção mais eficiente, flexível e escalável.

As redes ópticas atuais baseadas em multiplexação por divisão de comprimento de onda (*Wavelenght Division Multiplexing* - WDM) não suportam o aumento do tráfego da internet. Isso ocorre devido a característica de grade estática das redes WDM, diminuindo o aproveitamento do espectro e podendo gerar problemas de fragmentação na rede. Redes ópticas elásticas (*Elastic Optical Network* - EONs) surgiram com uma arquitetura mais flexível, fazendo uso eficiente dos recursos de rede. Isso se dá devido à diversas características, como uma grade mais flexível de granularidade mais fina que gera uma utilização melhor do espectro. Devido a essa mudança de heurística para uma rede mais elástica, o problema de Roteamento e Alocação de Comprimento de Onda (RWA - *Routing and Wavelength Allocation*) das WDM foi substituída pelo problema de Roteamento e Alocação de Espectro (RSA - *Routing and Spectrum allocation*) das EONs. Mesmo com o desenvolvimento dessas redes, a exaustão da Internet é uma preocupação crescente na

área. O termo crise de exaustão foi utilizado pela primeira vez no início deste milênio [4], e atualmente está muito presente na discussão sobre o futuro da Internet [5]. Isso ocorre quando os recursos para fornecer a confiabilidade, velocidade desejada e outros requisitos de rede se tornam insuficientes. De acordo com Waldman [6], isso não significa uma catástrofe para a Internet, mas um "novo normal".

As informações de enorme quantidade de dados heterogêneos, transportadas nas redes de telecomunicações, podem ser recuperadas através dos rastros das redes, alarmes de falhas em equipamentos, indicadores de qualidade de sinal (como a taxa de erro de bit - BER), dados comportamentais de usuários etc. Algumas ferramentas matemáticas estão sendo usadas para extrair e interpretar informações importantes, na qual serão utilizadas para tomada de decisões, visando o melhor funcionamento da rede. Uma dessas ferramentas que tem-se se destacado é o aprendizado de máquina (*Machine Learning* - ML). No contexto das redes ópticas, a adoção de técnicas de ML tem sido motivada pelo enorme crescimento de sua complexidade, uma consequência da grande quantidade de parâmetros utilizados atualmente, tais como: configurações de roteamento, formato de modulação, taxa de símbolo, esquemas de codificação adaptativa, espaçamento flexível de canais, dentre outros [7, 8].

ML é um ramo da Inteligência Artificial (IA) com a ideia de que a partir de dados relevantes, as máquinas podem aprender a resolver um problema específico [9]. No entanto, um tipo particular tem sido destacado na literatura, as redes neurais artificiais (RNA). Embora propostas na década de 1950, as redes neurais ganharam destaque recentemente devido principalmente à disponibilidade de grandes quantidades de dados e ao aumento do poder de processamento dos computadores. Assim, com o aumento da complexidade das RNAs, uma nova subclasse de algoritmos surgiu, denominada aprendizado profundo (DL). Um dos algoritmos notáveis de DL, são as redes neurais convolucionais (*Convolutional Neural Network* - CNN), normalmente utilizadas para análise de imagens, devido a sua capacidade de reconhecer padrões.

Com base nessas características, o DL começou a ser usado no planejamento da infraestrutura óptica que irá suportar a crescente demanda de tráfego de internet. Atualmente, grandes esforços têm sido aplicados para desenvolver novas tecnologias que viabilizam maior capacidade de transmissão nas grandes redes de transporte e, nesse contexto, as EONs têm ganho destaque [10]. Essas redes fazem uso do provisionamento de enlaces ópticos de maneira dinâmica e adaptativa para utilizar os recursos da rede de forma eficiente e flexível.

Esse trabalho apresenta um classificador baseado em aprendizado profundo com redes neurais capaz de identificar 11 estratégias diferentes de alocação de espectro em uma rede óptica elástica com altos níveis de acurácia. Foram comparados 3 diferentes algoritmos de

aprendizado profundo, a Rede Neural Profunda (DNN - *Deep Neural Network*), a Rede Convolucional (CNN) de 1 dimensão, e a tradicional Rede Convolucional de 2 dimensões. A escolha desses algoritmos foi devido a sua capacidade de reconhecer padrões e características dos dados analisados, podendo assim, realizar tarefas como a de classificação desses dados. Além disso, a análise foi feita em 2 topologias diferentes, na USANet e na PANEuro, utilizando o simulador de eventos discretos para redes ópticas ONS [11]. Para essa análise utilizamos uma representação da rede em forma de matriz, onde utilizamos diversos parâmetros variáveis, como a carga, o momento e a estratégia de alocação utilizada na rede. Até onde é conhecido, este é o primeiro uso de classificadores de aprendizado profundo para identificar estratégias de alocação de recursos em EON proposta na literatura.

Objetivo Geral

Os objetivo geral desse trabalho é a identificação de estratégias de algoritmos de roteamento e alocação de espectro em redes ópticas elásticas utilizando técnicas de aprendizado profundo.

Objetivos Específicos

- Estudar as estratégias dos algoritmos de roteamento e alocação de espectro nas redes ópticas elásticas.
- Investigar algoritmos de aprendizado profundo de redes convolucionais capazes de identificar e classificar os dados dos estados da rede.
- Desenvolver uma forma de representação de rede que viabilize a classificação das estratégias de roteamento e alocação de espectro pela rede convolucional desenvolvida.
- Investigar uma linguagem que viabilize a implementação de um classificador capaz de aprender através da representação de rede desenvolvida.
- Atestar a viabilidade dos métodos utilizados para a criação do classificador através dos resultados do treinamento e da validação do algoritmo.

Estrutura do Documento

Esse trabalho foi estruturado da seguinte forma:

No Capítulo 2 foi explorado conceitos básicos relacionados à pesquisa e apresentado uma revisão da literatura de trabalhos relacionados. Nos conceitos básicos será exposto as características de uma EON e suas restrições no problema RMLSA(*Routing, Modulation*

Level and Spectrum Assigment). Também será discutido a base de redes neurais, aprendizado profundo e redes convolucionais, onde nessa última será explicado todo o processo devido ao fato de não ser uma rede neural comum e ser usado de uma forma bem ampla nesse trabalho. Nos trabalhos relacionados será analisado o estado da arte, analisando de como os algoritmos de aprendizado profundo têm sido utilizados nos últimos anos. Além disso, também será analisado as fomas de representação da rede em alguns trabalhos, tendo em vista que essa representação é uma das maiores questões ao implementar uma rede neural à uma rede óptica. Será apresentado também as estratégias de alocação de espectro utilizadas para categorização, explicando como cada uma funciona.

No Capítulo 3 é apresentado a composição e os procedimentos realizados para a construção do trabalho. Nele será exposto como o classificador foi construído com cada algoritmo de aprendizado profundo, mencionando a separação de camadas, parametrizações e os motivos de escolha por trás de cada passo dessa construção. Além disso, será mostrado como foi feito os treinos e o tratamento dos dados de entrada para cada algoritmo.

No Capítulo 4 é apresentado os resultados dos treinamentos e testes. Esses resultados serão mostrados através de gráficos de curvas de aprendizado e matrizes de confusão, sendo estes um formato padrão para análise de performance de uma rede neural classificadora. Também serão realizadas várias análises e dissertações sobre essas representações e seus significados, explicando o porque dos resultados e verificando a viabilidade da ferramenta. E para finalizar, o Capítulo 5 contém as conclusões que foram tiradas do trabalho e são apresentadas sugestões para trabalhos futuros.

Capítulo 2

Conceitos Básicos e Revisão de Literatura

2.1 Redes Ópticas Elásticas

Redes ópticas baseadas em multiplexação por divisão de comprimento de onda (*Wavelength Division Multiplexing* - WDM) têm sido a solução principal de transporte de dados de alta velocidade. Elas utilizam para transmissão grades de frequência de tamanho fixo, geralmente com granularidade de $50GHz$ ou $100GHz$ por comprimento de onda, podendo gerar uma subutilização do espectro devido ao fato que as demandas feitas na rede possuem requisições de diferentes tamanhos de banda, gerando problemas como a fragmentação do espectro [12].

Para cada demanda de tráfego, uma grade inteira precisa ser alocada mesmo que ela seja inferior à capacidade de acomodação de um comprimento de onda. Para resolver o problema das limitações das redes tradicionais WDM, foi proposto o uso de uma rede óptica de alocação de espectro elástico baseada na tecnologia OFDM (*Orthogonal Frequency Division Multiplexing*) [13]. A OFDM, ou Modulação por Divisão Ortogonal de Frequência, é uma tecnologia de transmissão de múltiplas portadoras que transmite um fluxo de dados de alta velocidade dividindo-o em vários canais de dados paralelos de baixa velocidade [14].

Como o espectro de canais de subportadora adjacentes são modulados ortogonalmente, eles podem se sobrepor, aumentando a eficiência de transmissão espectral. Além disso, a OFDM óptica pode fornecer capacidade de granularidade fina às conexões pela alocação elástica de subportadoras de baixa taxa e com a possibilidade de alocação de um número de *slots* diferentes para cada circuito. A diferença proporcionada por essa flexibilidade em relação à grade fixa da WDM é exemplificada na Figura 2.1. Esse contexto de alocação mais eficiente e flexível do espectro em relação as redes WDM, gerou o desenvolvimento

de uma nova rede de transporte óptico chamada de rede óptica elástica (EON - *Elastic Optical Networks*). Uma EON tem o potencial de alocar um espectro para circuitos ópticos de acordo com os requisitos de banda dos clientes. O espectro é dividido em *slots* estreitos, geralmente com granularidade de 12,5 GHz e separados por um bit de guarda, e conexões ópticas são alocadas à diferentes números de *slots* [12].

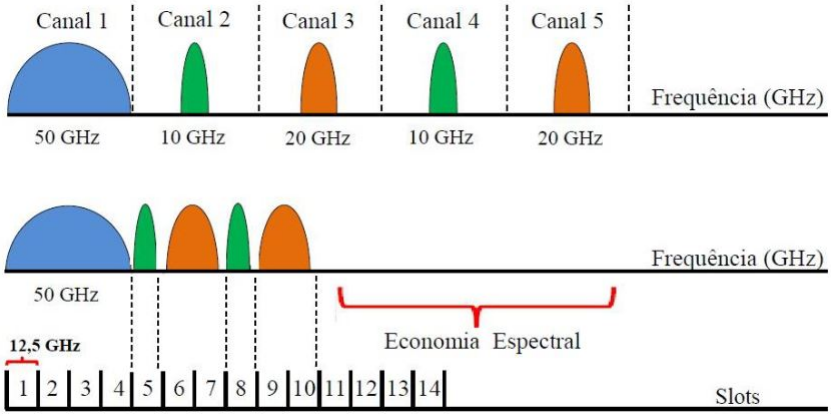


Figura 2.1: Comparação entre grade fixa e grade flexível [15]

Em EONs, um dos principais problemas é o de roteamento e alocação de espectro (RSA - *Routing and Spectrum Allocation*), que é equivalente ao problema de roteamento e alocação de comprimento de onda (RWA - *Routing and Wavelength Assignment*) das redes baseadas em WDM. O problema de estabelecimento de circuitos ópticos para cada requisição de conexão selecionando uma rota apropriada e alocando um comprimento de onda necessário é conhecido como o problema de roteamento e alocação de comprimento de onda (RWA) [16]. Nas redes baseadas em WDM sem conversores de comprimento de onda, o mesmo comprimento de onda deve ser usado durante todo o transporte da requisição. Essa propriedade é conhecida como restrição de continuidade de comprimento de onda.

Porém no problema RSA, devido a capacidade da arquitetura da EON de oferecer uma alocação de espectro flexível, um conjunto de *slots* de espectro contíguos é alocado para uma conexão ao invés do comprimento de onda na RWA em redes baseadas em WDM de grades fixas. Esses *slots* espectrais devem ser colocados próximos um do outro para satisfazer a restrição de contiguidade. Caso não existam *slots* vagos o suficiente no caminho desejado, a requisição de conexão pode ser quebrada em várias pequenas demandas, fazendo com que essas demandas necessitem de um número menor de *slots* de sub portadoras contíguas. Além disso, a continuidade desses *slots* devem ser garantidos de forma similar à restrição de continuidade de comprimento de onda na RWA [17].

Se uma requisição de conexão demandar x *slots* de espectro, então x *slots* contíguos devem estar vagos para serem alocados para a demanda, devido a restrição de contigui-

dade. Da mesma forma, os mesmos x *slots* contíguos devem ser alocados em cada *link* durante todo o trajeto de transporte da demanda, devido a restrição de continuidade. Utilizando a Figura 2.2 como exemplo, ao tentar atender uma requisição do nó 1 ao nó 4, vemos que através da rota link 1 > link 4 apenas a restrição da continuidade é atendida nos *links* 1 e 4, com apenas o *slot* 5 vago continuamente. Já através da rota link 1 > link 2 > link 3, vemos que a restrição da contiguidade é atendida com os *links* 5 e 6 e contínuo durante todo o transporte nos *links* 1, 2 e 3.

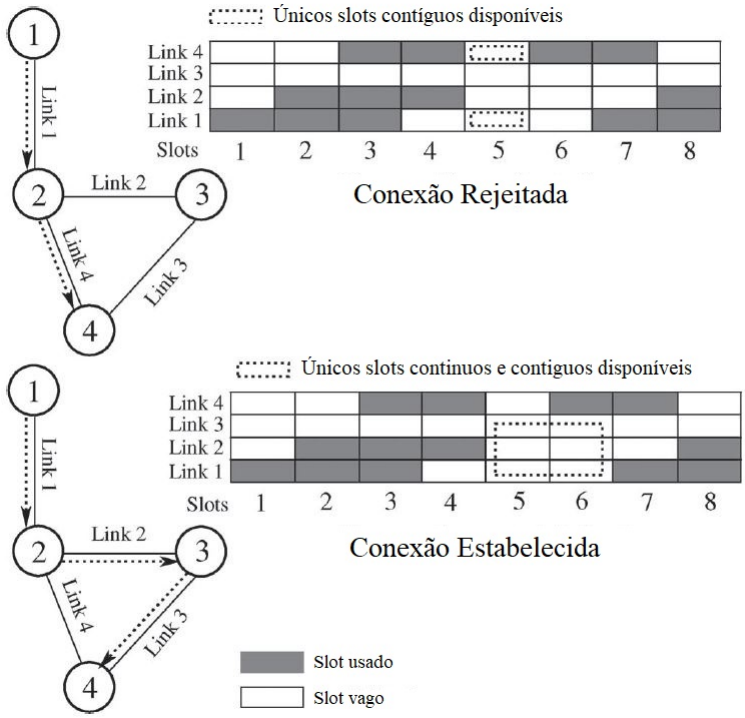


Figura 2.2: Exemplo das restrições de continuidade e contiguidade do espectro [12]

Além de todas essas características, a arquitetura EON possui a possibilidade de escolha do formato de modulação e largura de banda do canal para se adequar melhor à distância de transmissão e à qualidade de transmissão desejada. Essa característica importante de uma rede OFDM é feita através da escolha do número de bits por símbolo modulados para cada sub portadora que aumenta a flexibilidade e eficiência levando a distância em consideração. Com essa habilidade das redes EONs, foi incluído a atribuição do formato de modulação ao espectro da fibra no problema de roteamento e alocação de espectro RSA. Dessa forma, o problema RSA ganhou um novo componente: a modulação adaptativa, passando a ser referenciado na literatura como RMLSA - *Routing, Modulation Level and Spectrum Allocation* [18].

2.2 Algoritmos RMLSA

Nesse trabalho, foi utilizado diversos algoritmos seguindo as principais abordagens encontradas na literatura. Os algoritmos utilizados para a classificação são: SP *Random*, SP DP, SP *First Fit*, SP *Exact Fit*, todos com a estratégia de menor caminho; KSP *Best fit*, KSP PP, KSP *Last Fit* e KSP AP, todos com K = 3; KSP CS com K = 5; e MAdapSPV.

K-Shortest Paths

Os algoritmos de menor caminho (*Shortest Path* - SP) encontram a rota de menor custo entre um nó de origem e todos os demais nós da rede e o caminho preferencial entre esses nós. Quando o menor caminho entre dois nós não está disponível, o algoritmo determina o próximo menor caminho até encontrar um disponível. Essa série de caminhos derivados é conhecida como k-menores caminhos (KSP), representandos por um número k de menores caminhos entre dois nós. Para encontrar os KSPs de uma rede, geralmente é necessário determinar a quantidade de menores caminhos a serem encontrados, como por exemplo: K = 3 (K3SP), que encontra os 3 menores caminhos de uma rede, e K = 5 (K5SP), que encontra os 5 menores caminhos em uma rede [19].

Em EONs, esse algoritmo de roteamento possui dois passos, na qual primeiramente ele processa o roteamento e depois agregação de espectro passo a passo [20]. Esse algoritmo geralmente é combinado com outro algoritmo de alocação de espectro, visando o problema RSA, tanto nas versões SP simples e na KSP com diferentes valores de K. Veremos a seguir alguns desses algoritmos.

K5SP Random Fit

Esse algoritmo utiliza o algoritmo KSP, com K = 5, para o roteamento e a estratégia *Random Fit* para alocação de espectro. Essa estratégia mantém uma lista de *slots* vagos, e quando recebe uma requisição de conexão, o *RF* seleciona um *slot* aleatório da lista que atenda a demanda e aloca no circuito óptico. Após a alocação, o *slot* é deletado da lista até que a chamada seja completada. Utilizando uma seleção de espectro de forma aleatória pode reduzir a possibilidade de múltiplas conexões escolhendo o mesmo espectro, que é possível se essa seleção estiver sendo realizada de forma distribuída [21].

K7SP First Fit

Esse algoritmo utiliza o algoritmo KSP, com K = 7, para o roteamento e utiliza a estratégia *First Fit* para alocação de espectro. Nessa estratégia, os *slots* do espectro são indexados e uma lista de índices dos *slots* vagos e usados é mantida. A cada requisição de conexão, essa estratégia sempre tenta escolher o *slot* de menor índice da lista para

alocar no circuito óptico. A estratégia *First Fit* de alocação de espectro é considerada uma das melhores estratégias de alocação de espectro devido a sua baixa probabilidade de bloqueio de conexão e complexidade de computação, por não ter como requerimento ter informação de toda a rede para ser implementada [21].

K2SP Last Fit

Esse algoritmo utiliza o algoritmo KSP, com K = 2, para o roteamento e a estratégia *Last Fit* para alocação de espectro. Assim como realizado na estratégia *First Fit*, é criado uma lista de *slots* do espectro indexados na qual é mantida e atualizada de acordo com as alocações. Mas diferentemente da *First Fit*, a estratégia *Last Fit* prioriza o *slot* de maior índice [21].

K3SP First-Last Exact Fit

Esse algoritmo utiliza o algoritmo KSP, com K = 3, para o roteamento e a estratégia *First-Last Exact Fit* para alocação de espectro. Nessa estratégia, todos os *slots* do espectro de cada *link* pode ser dividido em um número de partições. A estratégia de alocação *First-Last Exact Fit* tenta escolher a partição com o menor índice e com o tamanho contíguo exato da demanda. Caso não encontre *slots* contíguos de tamanho exato disponíveis, essa estratégia tenta escolher a partição com o maior índice e com o tamanho contíguo exato da demanda. Esse esquema tem como objetivo aumentar o número de *slots* disponíveis alinhados, na qual evita o problema de fragmentação de largura de banda na rede [22].

K1SP Exact Fit

Esse algoritmo utiliza o algoritmo KSP, com K = 1, para o roteamento e a estratégia *Exact Fit* para alocação de espectro. A estratégia *Exact Fit* procura um número exato de *slots* para alocar a requisição de conexão. Caso não haja um bloco de tamanho exato, o espectro é alocado de acordo com a estratégia *First Fit*. Selecionando os *slots* de espectro desse jeito, nós podemos reduzir o problema de fragmentação nas redes ópticas [21].

K2SP Best Fit

Esse algoritmo utiliza o algoritmo dijkstra KSP, com K = 2, para o roteamento e a estratégia *Best Fit* para alocação de espectro. A estratégia *Best Fit* funciona de forma parecida com a estratégia *Exact Fit*, ela primeiramente procura um número exato de *slots* para alocar a requisição de conexão. Caso não haja um bloco de tamanho exato, diferentemente do que é feito na estratégia *Exact Fit*, procura-se um bloco com o número mais próximo do número de *slots* da requisição. A estratégia *Best Fit* é mais eficiente na

tentativa de reduzir o problema de fragmentação, porém é mais lenta do que a estratégia *Exact Fit* por tentar alocar sempre no espaço mais próximo do requisitado no circuito óptico [23].

K4SP Pseudo Particionamento (PP)

Esse algoritmo utiliza o algoritmo KSP, com K = 4, para o roteamento e a estratégia Pseudo Particionamento (PP - *Pseudo Partition*) para alocação de espectro. A estratégia PP consiste em dividir as solicitações de largura de banda alta e largura de banda baixa em ambas as extremidades do espectro, usando *First-Fit* para algumas demandas, e *Last-Fit* para outras demandas. Esse método evita as contenções diretas de taxas de dados alta com taxas de dados baixas, enquanto mantém o compartilhamento de todos os recursos do espectro. Efetivamente, uma "demarcação" estatística é formada para criar duas partições do espectro. Porém, esse é um pseudo particionamento, onde conexões de largura de bandas diferentes ainda são misturados na sua ocupação do espectro [24].

SP DP Partição Dedicada

Esse algoritmo utiliza o algoritmo SP para o roteamento e a estratégia Partição Dedicada (DP - *Dedicated Partition*) para alocação de espectro. A estratégia de Partição Dedicada divide o espectro e cada demanda de largura de banda tem uma parte dedicada do espectro. Cada partição do espectro carrega o mesmos circuitos ópticos de largura de banda, reduzindo cada um à uma rede de comprimento de onda roteada. Essa estratégia faz utilização de umas das estratégias mencionadas acima em conjunto com o particionamento [24].

K4SP AP Propenso a Aceitação

Esse algoritmo utiliza o algoritmo KSP, com K = 4, para o roteamento e a estratégia Propenso a Aceitação (AP - *Acceptance Prone*) para alocação de espectro. A estratégia de Propenso a Aceitação seleciona os *slots* com base em uma função de custo que minimiza a fragmentação do espectro. Essa função calcula a fração média de demandas que cada conjunto de *slots* vagos contíguos pode acomodar [25].

K3SP CS Compartilhamento Completo

Esse algoritmo utiliza o algoritmo KSP, com K = 3, para o roteamento e a estratégia Compartilhamento Completo (CS - *Complete Sharing*) para alocação de espectro. A estratégia de Compartilhamento Completo tenta equilibrar a distribuição link-carga na rede sempre alocando, entre os caminhos disponíveis, o menor espectro possível para uma

solicitação de conexão. Os *slots* do espectro são indexados, e assim que recebe uma requisição de conexão, independentemente da largura de banda, é escolhido o caminho de espectro contíguo de menor índice para atender a demanda. Essa estratégia é uma extensão da estratégia *First Fit* [24].

MAdapSPV

O algoritmo SPV (*Spectrum-Constraint Path Vector Searching*) é um algoritmo heurístico que leva em consideração ambas as etapas de roteamento e agregação de espectro. Nesse algoritmo, todos os caminhos possíveis são encontrados e armazenados em uma árvore de busca de vetor de caminho, onde cada vetor de caminho com o espectro agregado é examinado em cada nível. O MAdapSPV é uma versão do SPV que foi adaptado para suportar os níveis de modulação para solucionar o problema RMLSA [20].

2.3 Redes Neurais Artificiais

As Redes Neurais Artificiais ou RNAs são um paradigma que foi inspirado pela forma na qual o sistema nervoso biológico como o cérebro processa informação. Ele é composto por um grande número de elementos de processamento altamente conectados (neurônios) trabalhando em união para resolver um problema específico.

Uma RNA padrão consiste em vários neurônios simples interconectados, com cada um produzindo uma sequência de ativações com valores reais. Neurônios de entrada são ativados através de sensores percebendo o ambiente, e outros neurônios são ativados através de conexões ponderadas de neurônios ativados anteriormente. Alguns neurônios podem influenciar o ambiente desencadeando ações. O processo de aprendizagem diz respeito a encontrar os pesos ponderados no qual faz a RNA exibir o comportamento desejado, como realizar uma tarefa ou ponderar sobre uma decisão [26].

2.3.1 Neurônio Biológico

O neurônio é uma célula biológica especial do cérebro humano capaz de processar informações. Ela é composta principalmente pelo corpo celular ou soma, na qual possuem dois tipos de ramificações: os dendritos, pela qual é realizada a recepção de informações pelo núcleo, e o axônio, no qual descende da soma e possui várias ramificações terminais na sua extremidade pelos quais é realizada a transmissão de informações para outros neurônios. Na Figura 2.3 podemos observar uma representação do neurônio biológico e sua estrutura.

O cérebro é formado por bilhões de neurônios, contendo inúmeras conexões entre eles formando a rede neural. O neurônio recebe sinais (impulsos) de outros neurônios através

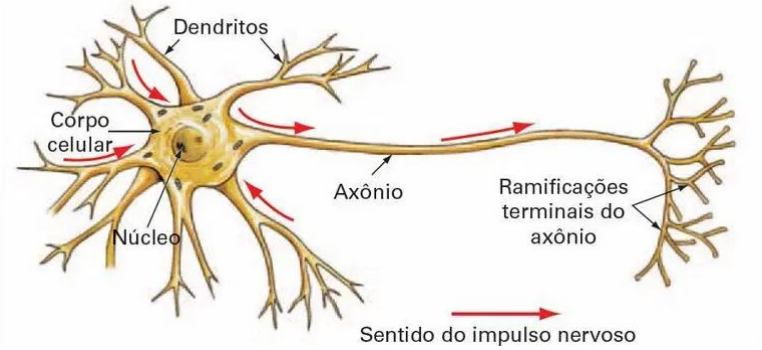


Figura 2.3: Neurônio Biológico simplificado [27]

de seus dendritos e transmite sinais gerados pelo se corpo celular através do axônio, no qual eventualmente ramifica-se em fios e substratos, onde encontrará as sinapses no terminal desses fios. A sinapse é uma estrutura elementar e unidade funcional entre dois neurônios. Quando o impulso atinge o terminal da sinapse, os neurotransmissores são liberados, onde se difundem através do espaço sináptico, para aumentar ou inibir, dependendo do tipo de sinapse, a própria tendência do neurônio receptor de emitir impulsos elétricos. A eficácia da sinapse pode ser ajustada pelos sinais que passam por ela, para que as sinapses possam aprender com as atividades na qual atuam [26].

2.3.2 Neurônio computacional

A partir da estrutura e funcionamento do neurônio biológico, os pesquisadores *McCulloch* e *Pitts* propuseram em 1943 uma unidade de limiar binário, chamada de Perceptron, como modelo para um neurônio artificial.

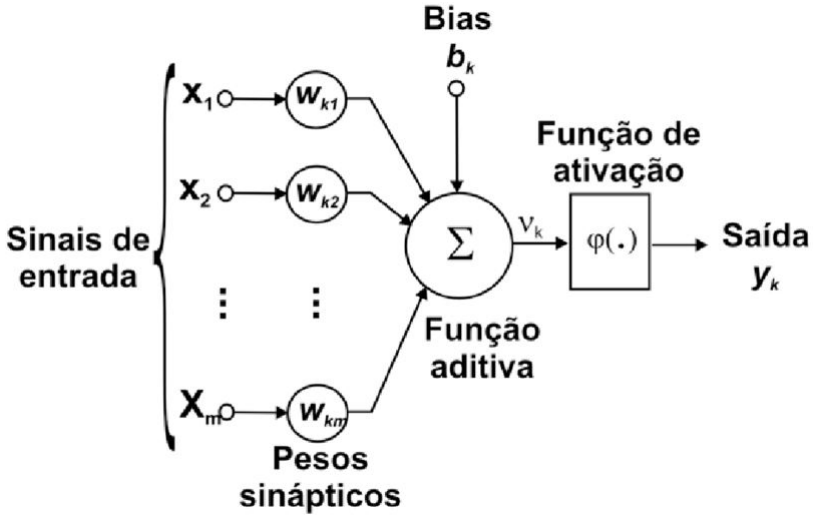


Figura 2.4: Representação do Neurônio Matemático [28]

Esse neurônio matemático k recebe n sinais de entrada, denominados x_n conforme pode se observado na Figura 2.4. Cada um dessas entradas são multiplicados por um peso de conexão ou sinapse, representadas como w_{kn}. O peso mostra a força de um nó específico. Feito isso, é calculado o somatório desses sinais juntamente com o valor de bias b_k, no qual permite alterar a função de ativação para cima ou para baixo, resultando em v_k, que é alimentado a uma função de ativação $\varphi()$ para gerar um resultado, e esse resultado é enviado como sinal de saída y_k. Matematicamente:

$$v = b + x_1 \cdot w_1 + x_2 \cdot w_2 + x_3 \cdot w_3 + ... + x_n \cdot w_n \tag{2.1}$$

$$y = \varphi(v) \tag{2.2}$$

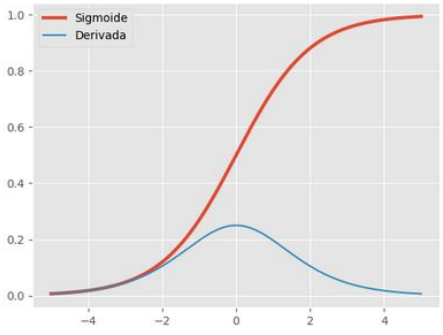
Os sinais de entrada $(x_1, x_2, ..., x_n)$ são sinais externos que o neurônio recebe, que podem ser normalizados para aumentar a eficiência computacional dos algoritmos de aprendizagem. Os pesos sinápticos $(w_1, w_2, ..., w_n)$ são valores que ponderam a força de excitação do nó no neurônio, onde esses valores são aprendidos durante o treinamento. O somador $(\sum)$ calcula todos os valores de entrada, ponderados pelos seus respectivos valores de peso, para resultar em um potencial de ativação. O potencial de ativação (v) é o resultado obtido pela diferença do valor entre o somador e o limiar de ativação, sendo excitatório se positivo, e inibitório caso negativo. A função de ativação $(\varphi())$ é um limiar da saída do neurônio em um intervalo de valores. E o sinal de saída (y) é o valor final de saída do neurônio, podendo ser usado como entrada de outros neurônios ligados em sequência, ou ter o seu valor interpretado para a solução de um determinado problema [29].

A função de ativação, também conhecida como função de transferência, é uma função utilizada para determinar a saída de um nó como um "sim" ou "não". Ela mapeia o valor do potencial de ativação entre 0 e 1, ou -1 e 1, e etc, dependendo do função utilizada. As funções de ativação são separadas primariamente entre linear e não-lineares [9].

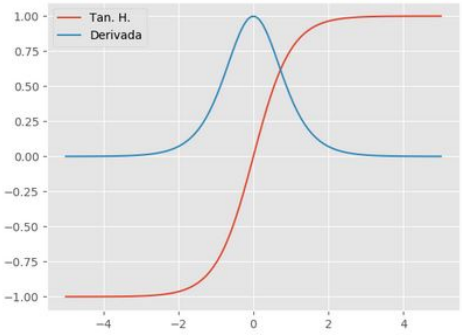
A função linear não restringe a saída entre valores, sendo uma linha reta onde a ativação é proporcional a entrada. Essa característica torna essa função útil geralmente apenas em problemas de baixa complexidade, onde os dados podem ser facilmente separados.

$$f(x) = Cx \tag{2.3}$$

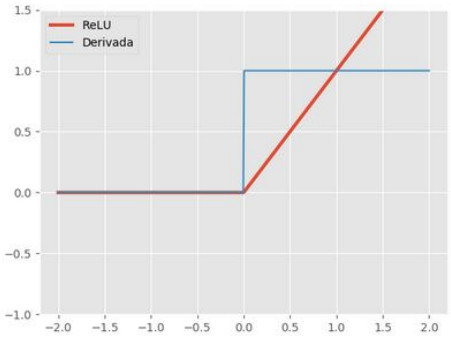
A função não-linear torna fácil para o modelo a generalização ou a adaptação com grandes variedades de dados, por tornar possível uma divisão mais complexa entre os dados, tornando possível solucionar melhor problemas mais complexos. As funções de



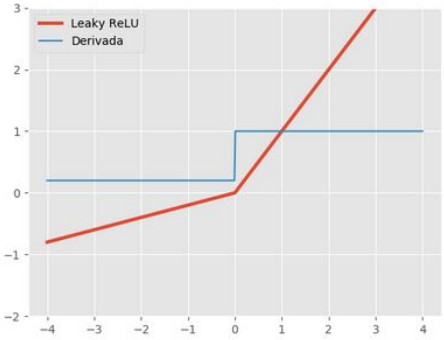
(a) Sigmoide



(b) TanH



(c) ReLU



(d) *Leaky* ReLU

Figura 2.5: Funções de ativação

ativação não-lineares são divididas principalmente baseadas em suas faixa de valores ou curva. As funções mais utilizadas para resolver problemas utilizando redes neurais são a função degrau, sigmoide, tangente hiperbólica, RELU (*Rectified Linear Unit*, ou unidade linear rectificada), *leaky* RELU e *Softmax* [30]. Algumas dessas funções estão representadas na Figura 2.5

A função degrau é uma função binária extremamente simples, onde ela é basicamente um limiar de ativação que decide se o neurônio deve ser ativado ou não. Essa função pode assumir apenas os valores 1 ou 0, como mostrado na Equação 2.4.

$$\begin{cases} f(x) = 1 & \text{se } x >= 0 \\ f(x) = 0 & \text{se } x < 0 \end{cases} \tag{2.4}$$

A função sigmoide é uma função não linear suave e continuamente diferenciável com um formato de 'S'. Ela produz valores entre 0 e 1, e tende a "empurrar" seu resultado para as extremidades desse intervalo. Essa função normalmente é utilizada apenas na camada de saída devido ao fato que nas suas extremidades onde ela satura, as suas derivadas tendem a zero, reduzindo bastante a propagação do gradiente e dificultando o treinamento. Essa

função é representada pela Equação 2.5.

$$\sigma = \frac{1}{1 + e^{-x}} \tag{2.5}$$

A função tangente hiperbólica (TanH) é similar a função sigmoide, porém a sua variação é de -1 a 1. As saturações ainda estão presentes, mas o valor da derivada é maior, chegando ao máximo de 1 quando x=0. Essa função aproxima mais da identidade, sendo assim uma alternativa mais viável do que a sigmoide para servir de ativação às camadas ocultas das RNAs. Essa função é representada pela Equação 2.6.

$$tanh(x) = 2\sigma(2x) - 1 \tag{2.6}$$

A função ReLU produz resultados entre os valores 0 e ∞. Essa função retorna 0 para todos os valores negativos, e retorna o próprio valor para valores positivos. Como o seu resultado é zero para valores negativos, ela pode fazer com que alguns neurônios "morram" e não aprendam nada se eles só receberem valores negativos. Mesmo com suas limitações, essa função é uma das funções de ativação mais utilizadas no treinamento de RNAs, e não costuma ser utilizada na camada de saída. Essa função é representada pela Equação 2.7.

$$\text{ReLU}(x) = max0, x \tag{2.7}$$

A função *Leaky* ReLU é uma modificação da função ReLU. A diferença nessa função é que ao invés de zerar os valores negativos, aplica a eles um fator de divisão (vazamentos ou *leakies*) muito pequeno, fazendo com que a derivada na região negativa ainda seja positiva. Esse comportamento tende a resolver alguns dos problemas da função ReLU relacionados aos valores zerados. Essa função é representada pela Equação 2.8.

$$\text{LeakyReLU}(x, \alpha) = max\alpha x, x \tag{2.8}$$

A função *Softmax* é uma generalização da função sigmoide para casos não-binários. Ela produz valores entre 0 e 1, divididos pelas somas das saídas, dando a probabilidade da entrada estar em uma determinada classe. Devido essa característica, essa função normalmente é aplicada às camadas de saída de problemas de classificação multi-classe. Essa função é representada pela Equação 2.9.

$$\sigma(z)_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \quad \text{para } j = 1, \ldots, K. \tag{2.9}$$

2.3.3 Aprendizado Profundo

Denomina-se de Aprendizado de Máquina (ML - *Machine Learning*) a utilização de algoritmos para extrair informações de dados brutos e representá-los através de algum tipo de modelo matemático. Usamos então este modelo para fazer inferências a partir de outros conjuntos de dados. Existem muitos algoritmos que permitem fazer isso, tais como a Máquina de Vetores de Suporte (*Support Vector Machine* - SVM), o K-Vizinhos mais Próximos (*K-Nearest Neighbor* - KNN) e a Floresta Aleatória (*Random Forest*). Porém um tipo em especial vem se destacando: as redes neurais artificiais (RNAs). Apesar de ter sido proposta na década de 1950, as redes neurais têm ganho grande destaque recentemente devido principalmente à grande quantidade de dados disponíveis e ao aumento no poder de processamento das máquinas. Assim, com o aumento na complexidade das RNAs, surgiu uma nova subclasse de técnicas denominadas de aprendizado profundo (DL - *Deep Learning*).

O aprendizado profundo permite que modelos computacionais de aprendizado de máquina que são compostos de várias camadas de processamento aprendam representações de dados com vários níveis de abstração. Esses métodos melhoram drasticamente o estado da arte em diferentes áreas, como o reconhecimento de fala, reconhecimento visual, detecção de objetos, e muitos outros domínios [31]. Aprendizado profundo possui várias definições ou descrições de alto nível, sendo que os aspectos comuns entre elas é que essa classe de algoritmos de ML usa uma cascata de diversas camadas de unidade de processamento não-linear para a extração e transformação de características, com aprendizagem supervisionada ou não supervisionada em cada camada, formando uma hierarquia das características de baixo nível para as de alto nível. A motivação que trouxe a popularização das ferramentas de aprendizado profundo foram primariamente três acontecimentos: o grande salto do poder de processamento das placas de vídeo (GPU - *Graphic Processing Unit*), a quantidade gigantesca de dados para treinamento, e os avanços recentes em aprendizado de máquina e no processamento de informação [32].

O desempenho dos métodos de aprendizado de máquina são altamente dependentes da escolha de representação dos dados. A aprendizagem de representação (*Representation Learning*) é um conjunto de métodos que permite uma máquina ser alimentada com dados brutos, sem interferência de engenheiros humanos, e descobrir automaticamente as representações necessárias para detecção ou classificação. Os métodos de aprendizado profundo são métodos de aprendizado de representação com vários níveis, obtidos da composição de módulos simples não lineares em que cada uma transforma a representação em um nível (começando com a entrada bruta) em uma representação com um nível ligeiramente mais alto de abstração. Com a composição de transformações suficientes, funções complexas podem ser aprendidas [33].

No caso de modelos de aprendizado profundo probabilísticos, uma boa representação normalmente é aquela que captura a distribuição posterior dos fatores explicativos subjacentes para a entrada observada. Uma boa representação é também aquela que é útil como entrada para um preditor supervisionado. A forma mais comum de aprendizado de máquina, sendo profunda ou não, é o aprendizado supervisionado. Essa forma consiste em uma tarefa de aprendizado de máquina de aprender uma função que mapeia uma entrada para uma saída baseada em um exemplo de pares de entrada-saída [34]. Esses pares consistem de um dado de entrada e uma saída desejada para esse dado. Durante o treinamento, a máquina recebe um objeto de entrada, produzindo uma saída na forma de um vetor de porcentagens para cada categoria possível. Com isso, é calculado uma função objeto que mede o erro entre as porcentagens de saída e o padrão de porcentagem desejada proveniente do par entrada-saída desse objeto. A máquina então modifica os seus parâmetros internos ajustáveis, normalmente chamados de pesos, para reduzir esse erro.

Para ajustar propriamente os vetores de pesos, que são os conjuntos de pesos da máquina, o algoritmo de aprendizado calcula um vetor de gradiente. Esse vetor indica, para cada peso, a quantidade de erro que deve aumentar ou diminuir se o valor do peso for aumentado. O procedimento mais utilizado na prática em aprendizado de máquina é a Descida de Gradiente Estocástico (*Stochastic gradient Descent*) [31]. Ela consiste em utilizar algumas entradas como exemplos na máquina, computando as saídas e os erros, calculando o gradiente médio para esses exemplos e ajustando os pesos adequadamente. Esse processo é repetido para vários conjuntos pequenos de entradas dos dados de treinamento até que a média da função objetiva pare de diminuir. Esse procedimento simples geralmente encontra um bom conjunto de pesos surpreendentemente rápido quando comparado com técnicas de otimização muito mais elaboradas [35].

Para treinar os algoritmos de aprendizado profundo, que possuem uma arquitetura multicamada, é utilizado geralmente o algoritmo de *backpropagation*. Esse algoritmo consiste no procedimento de cálculo do gradiente de uma função objeto em relação aos pesos de uma pilha das camadas do algoritmo treinado. Mas esse cálculo é realizado de trás para frente e camada por camada, começando da saída da rede de aprendizagem (topo da pilha), até a entrada (base da pilha). A eficiência desse algoritmo torna viável o uso de métodos de gradientes para treinar redes profundas, atualizando pesos para minimizar os erros.

2.3.4 Rede Neural Convolucional

A rede neural convolucional (*Convolutional Neural Network* - CNN) é uma classe de algoritmo de aprendizagem profunda caracterizada pela utilização, em pelo menos uma de

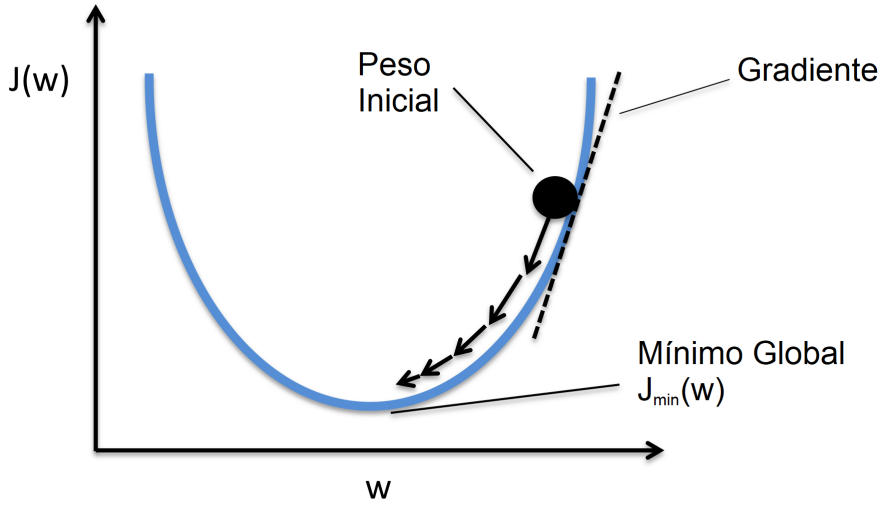


Figura 2.6: Ilustração de descida do gradiente [36]

suas camadas, da operação matemática convolução [37]. A CNN pode processar principalmente: i) dados em forma de grade, como textos e ondas sonoras, sendo conhecidas como redes convolucionais de 1 dimensão (1D); ou ii) imagens, nos quais os dados são organizados em *pixels* com duas ou três dimensões, sendo conhecidas como redes convolucionais 2D [38]. Esse processo é possível devido a capacidade da CNN em detectar bordas, características relevantes e interações esparsas. O processo de convolução também pode ser usado para eliminar variabilidades irrelevantes utilizando subamostragem, parametrização e pesos compartilhados entre as convoluções sendo um dos principais algoritmos de aprendizagem profunda para reconhecimento e categorização de imagens [9].

Uma CNN pode ser dividida em duas partes: a extração de características e a classificação. A extração de características consiste em uma série de camadas convolucionais com filtros (*Kernels*) e camadas de *Pooling*, na qual é realizado a extração de características da imagem de entrada, reduzindo-a para uma forma mais fácil de processar sem perder características importantes para ter uma boa previsão. Ela recebe uma entrada constituída geralmente por imagens de tamanho normalizado e centralizadas, na qual são processadas nessas camadas iniciais. Após esse processo, essa imagem é achatada (*flattening*) e realimentada à um classificador. Este classificador é constituído por uma RNA interconectada que será responsável pela classificação da imagem através das características extraídas.

Essas camadas são organizadas como mostrado na Figura 2.7, mostrando de forma esquemática uma CNN com 2 convoluções e um classificador, que nada mais é que uma rede neural interconectada. Cada convolução é constituída de uma camada convolucional seguida de uma camada de *pooling*. Os conceitos e funcionalidade de cada uma dessas camadas da rede convolucional será explicada e exemplificada nas sub-seções seguintes.

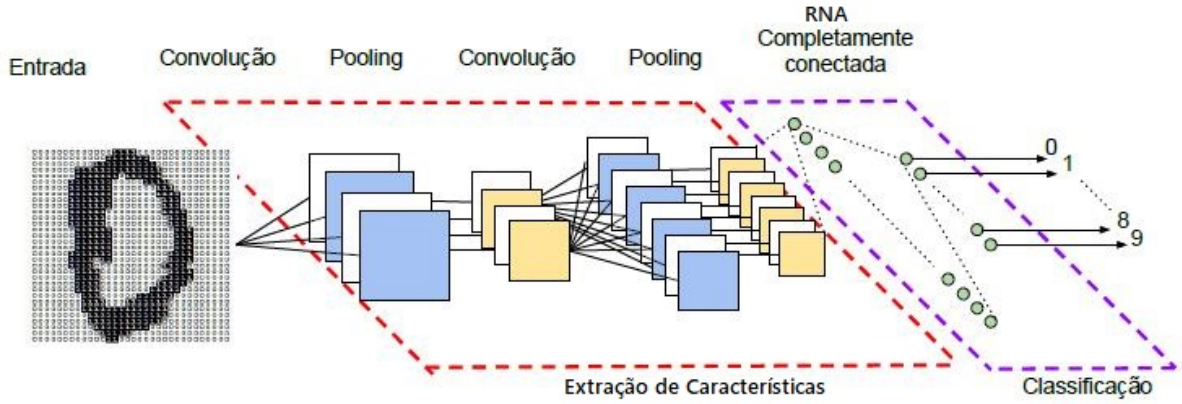


Figura 2.7: Fluxo de uma Rede Neural Convolucional [39]

Camada Convolucional

A convolução é a primeira camada para extrair características de uma imagem de entrada. Ela é composta por uma série de *kernels* (filtros) convolucionais, no qual cada neurônio atua como um *kernel*, contendo uma série de pesos. A entrada é uma matriz de *pixels* que representa uma imagem de formato $altura(a) \times largura(l) \times dimensão(d)$, onde a dimensão consiste no número de canais (escala de cores), onde pode ser processada por operações de normalização ou de preenchimento (*padding*). A camada convolucional divide a imagem em pedaços menores, chamados de campos receptivos, os quais serão utilizados na extração de alguma característica. Essa extração é realizada através da convolução do *kernel* com a imagem, multiplicando os elementos do *kernel* com os elementos correspondentes do campo receptivo. Cada convolução de um campo receptivo resulta em um elemento da matriz do mapa de características (*Feature Map*) [40]. Essa convolução pode ser expressada pela seguinte equação:

$$f_l^k(p,q) = \sum_c \sum_{x,y} i_c(x,y).e_l^k(u,v), \tag{2.10}$$

Onde $i_c(x,y)$ representa um elemento da matriz de imagem I_C e $e_l^k(u,v)$ representa o k-ésimo *kernel* convolucional da l-ésima camada. Cada mapa de características de saída é expressado como $F_l^k = [f_l^k(1,1), ..., f_l^k(P,Q)]$, onde P e Q representam o número de linhas e colunas, respectivamente. Devido a habilidade de compartilhar pesos da operação de convolução, diferentes conjuntos de características podem ser extraídos de uma imagem com a passagem do *kernel* por toda a imagem utilizando o mesmo conjunto de pesos, fazendo com que a CNN tenha mais eficiência com os parâmetros quando comparada com uma RNA interconectada. A convolução pode ser dividida em diferentes categorias baseados no: i) tipo e tamanho dos filtros; ii) tipos de *padding* e; iii) direção da convolução [31]. A passagem do *kernel* é feita da seguinte forma: ele move para a direita com um certo

valor de passada (*stride value*) até analisar a largura completa. Seguindo em frente, ele pula para o início (esquerda) de baixo da imagem com o mesmo valor de passada e repete o processo até que toda a imagem seja percorrida. Esse processo tem como resultado um mapa de características.

Figura 2.8: Exemplo de matriz da imagem e do filtro [41]

Utilizando essas duas matrizes como exemplo, será mostrado como a captura de características é realizada na camada convolucional na prática. A matriz em verde representa a matriz de imagem, enquanto a matriz em laranja representa a matriz do filtro (*kernel*), será realizada várias multiplicações de matrizes que serão ilustradas a seguir através da sobreposição da matriz do filtro sobre a matriz da imagem, tendo como resultado um valor parcial do mapa de características na matriz em vermelho.

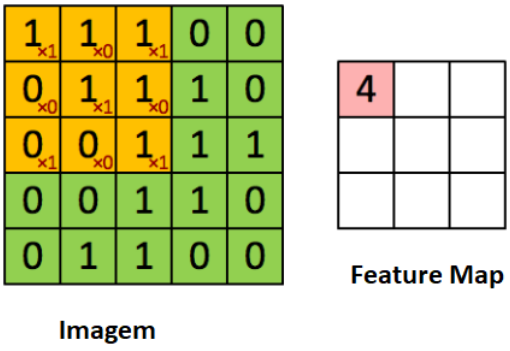


Figura 2.9: Exemplo de uma iteração de convolução parte 1 [41]

Mostrando os cálculos:

$$(1 \times 1) + (1 \times 0) + (1 \times 1) + (0 \times 0) + (1 \times 1) + (1 \times 0) + (0 \times 1) + (0 \times 0) + (1 \times 1) = 4 \tag{2.11}$$

Após essa iteração, o filtro é movido de acordo com um valor de passagem definido. Esse valor é um número inteiro completamente arbitrário, podendo ser "setado" no desenvolvimento da rede. Será usado valor de passagem igual a 1 para esse exemplo, fazendo com que o filtro desloque apenas uma posição por iteração. Feito o deslocamento, será realizada novamente outra multiplicação de matrizes resultado em um novo valor do mapa de características como mostrado nas Figuras 2.10 e 2.11.

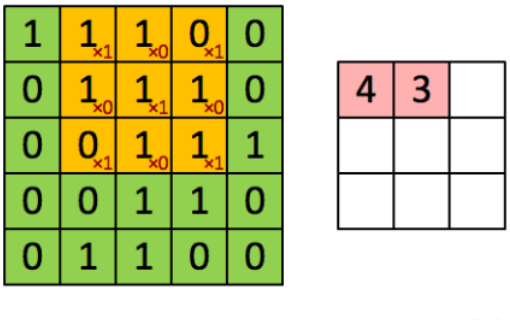


Figura 2.10: Exemplo de uma iteração de convolução parte 2 [41]

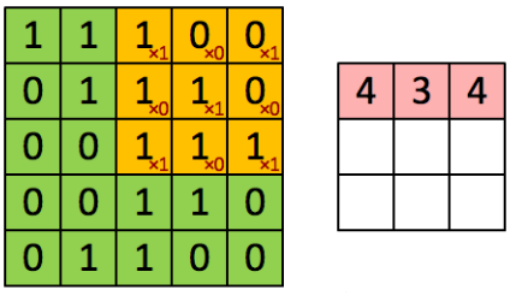


Figura 2.11: Exemplo de uma iteração de convolução parte 3 [41]

Ao chegar na extremidade direita da imagem, o filtro é deslocado para a extremidade da esquerda e depois deslocado para baixo de acordo com o valor de *stride* exemplificado na Figura 2.12. É realizado a multiplicação de matrizes, e o resultado é colocado no *feature map* de acordo com a posição do filtro na matriz de imagem.

Essas multiplicações de matrizes e deslocamentos são realizados até que o filtro percorra toda a imagem, resultando na matriz do *feature map* da Figura 2.13, que poderá ser enviado para uma camada de *pooling*.

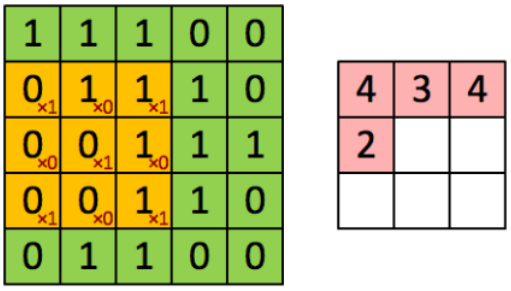


Figura 2.12: Exemplo de uma iteração de convolução parte 4 [41]

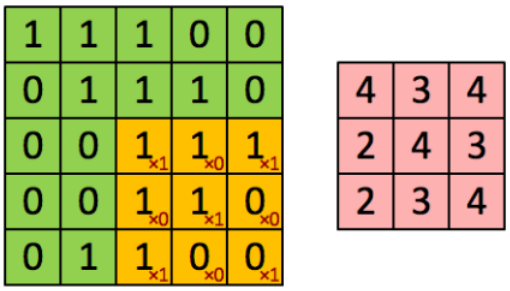


Figura 2.13: Mapa de características resultante do exemplo de convolução [41]

A primeira camada convolucional convencionalmente é responsável por capturar características de baixo nível, como bordas, cores, orientação de gradiente, entre outros [42]. Porém, a rede não precisa se limitar à apenas uma camada, podendo assim, adicionar várias camadas fazendo com que a arquitetura se adapte para características de alto nível, como padrões em uma imagem, resultando em uma rede que possua um entendimento completo das imagens do *dataset* utilizado.

Como o tamanho do *kernel* é sempre menor que a entrada, a matriz de saída sempre é menor que a matriz de entrada devido a operação de convolução. Para evitar que ocorra essa redução, é usado o *padding* (contorno ou preenchimento). Ele consiste no preenchimento em volta da matriz de entrada com *pixels* de valor zero, como mostrado na Figura 2.14, para que o mapa de características não reduza na saída. Além de manter o tamanho espacial constante após a convolução, o preenchimento também melhora o desempenho e garante que o tamanho do *kernel* e do *stride* caiba na entrada.

Camada de Pooling

Ao finalizar a extração de características, a sua exata localização se torna pouco importante, desde que sua posição aproximada relativa às outras características seja preservada.

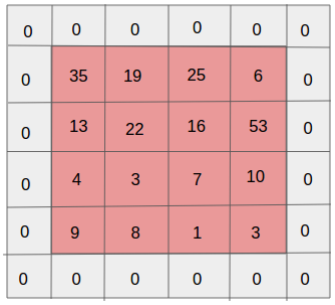


Figura 2.14: Exemplo de padding [43]

A função de *pooling* substitui a saída da camada convolucional em uma determinada localização com o valor estatístico resumido dominante dos valores de saída próximos [44]. Os tipos de *pooling* mais utilizados são o *Max Pooling* e *Average Pooling*. O *Max Pooling* [45] utiliza o maior valor dentro de uma vizinhança retangular coberta por um *kernel* para realizar a substituição. Já o *Average Pooling* retorna a média de todos os valores cobertos por um *kernel*. Essa operação pode ser representada pela seguinte equação:

$$Z_l^k = g_p(F_l^k) \tag{2.12}$$

Onde Z_l^k representa o mapa de características após o *pooling* do l-ésima camada para o k-ésimo mapa de características de entrada F_l^k, e a função $g_p()$ determina o tipo de operação de *pooling*.

Essa operação é realizada da mesma forma que a convolução, mas ao invés de uma multiplicação entre a matriz da imagem e o filtro, é extraído o valor máximo (para o *Max Pooling*) da matriz da imagem para a matriz resultante. Cada iteração é seguida da movimentação do filtro através do valor de *stride*, assim como é feito na camada de convolução. Podemos ver na Figura 2.15 o resultado dessa operação, onde cada iteração da operação de *max pooling* é representada por uma cor diferente.

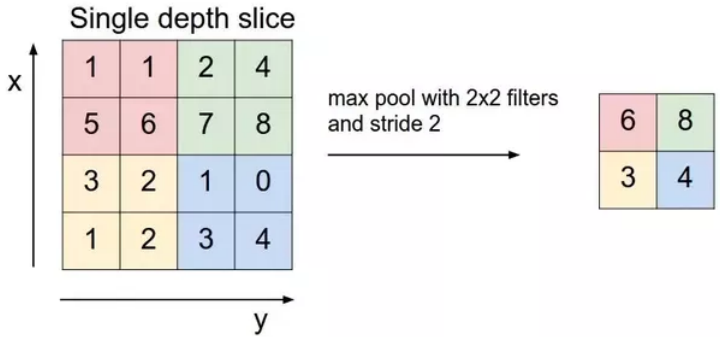


Figura 2.15: Exemplo de *max pooling* com um filtro 2×2 e *stride* 2 [41]

Realizado essa operação, a matriz resultante pode ser realimentada em outra camada convolucional, ou alimentada à camada inter-conectada. A matriz é achatada (*flattening*), transformando-a em um vetor como mostrado na Figura 2.16. Essa saída será alimentada à uma rede neural interconectada que será responsável pela classificação da imagem.

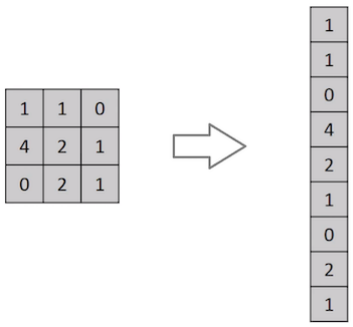


Figura 2.16: *Flattening* de uma matriz 3×3 para um vetor 9×1 [43]

A operação de *pooling* trás como principais vantagens: i) estabilidade na classificação de uma imagem mesmo com mudanças translacionais e pequenas distorções [46]; ii) regulação da complexidade da rede; iii) aumento da capacidade de generalização e consequente redução do *overfitting*, podendo eliminar ruídos [47]. Podemos mencionar ainda outros tipos de *pooling* usados em CNN, como por exemplo o *average*, *overlapping* e o L2 [48].

Foram escolhidas as rede convolucionais 1D e 2D para esse trabalho devido a utilização da operação de convolução de matrizes e redução do tamanho dos dados nessa rede. Essa operação permanece eficaz tanto com o padrão de análise de matrizes 2D, quanto com a utilização de dados unidimensionais 1D. Essa análise unidimensional foi feita de cada enlace separadamente, tendo como entrada uma matriz unitária. Essa ferramenta será essencial para encontrar os padrões e características nos estados da rede para categorizar o algoritmo RMLSA utilizado.

2.4 Trabalhos Relacionados

2.4.1 Representação da rede óptica em aprendizado profundo

A seleção de um conjunto de variáveis de dados de entrada é uma questão importante na utilização de ferramentas de ML. Essa escolha deve levar em conta as características a serem aprendidas e generalizadas, podendo assim gerar resultados satisfatórios para a solução de um determinado problema. Tendo isso em mente, para viabilizar a utilização de técnicas de aprendizado profundo em redes ópticas nasceu a necessidade de uma representação da rede de forma que a característica a ser analisada seja evidenciada nas amostras.

Diversas pesquisas foram feitas utilizando ferramentas de ML, como as RNAs por exemplo, na qual utilizaram diferentes representações de rede para solucionar problemas em diferentes áreas.

Na área de roteamento e previsão de tráfego, os autores de [49] avaliaram diversos modelos de aprendizado de máquina, principalmente de aprendizado supervisionado, com o objetivo de minimizar a superutilização (congestionamento) do *link*. Nessa abordagem eles utilizam um roteamento baseado em dados na qual eles tentam aprender os parâmetros através de uma matriz de demanda, proveniente de fluxos de tráfegos induzidos, com a finalidade de prever à próxima matriz de demanda para otimizar a estratégia de roteamento. A matriz de demanda consiste em uma representação onde o valor (i, j) especifica a demanda de tráfego entre a fonte i e o destino j, em que qualquer matriz de demanda e estratégia de roteamento induz um fluxo de tráfego na rede. Os resultados preliminares indicaram que o aprendizado supervisionado pode ser inefetivo se as condições de tráfego não exibem uma regularidade muito alta. Na mesma área, no trabalho de Stampa [50] é utilizado uma abordagem de aprendizado por reforço (*reinforcement learning*), na qual ele se adapta automaticamente às condições de tráfego atuais e propõe configurações sob medida que tentam minimizar o atraso (*delay*) da rede. O modelo sugerido utiliza uma matriz de tráfego para representar o estado da rede, na qual eles tentam aprender diretamente um comportamento ótimo para as estratégias de roteamento. Os experimentos mostraram um desempenho promissor, mostrando que essa abordagem oferece vantagens operacionais importantes no que diz respeito aos algoritmos de otimização tradicionais.

Em [51] os autores propuseram um agente RMLSA de autoaprendizagem baseado em aprendizado por reforço, para realizar RMLSA de forma autônoma e cognitiva para EONs. Nesse trabalho preliminar foi estruturado uma CNN, na qual consiste múltiplas convoluções e múltiplas camadas interconectadas para aprender as melhores estratégias RMLSA em relação a diferentes estados da EON e requisições de circuito óptico. Posteriormente em [52], foi proposto um agente no qual utiliza DNNs para aprender as parametrizações das estratégias RMLSA com experiências através de provisionamento de circuitos ópticos. Nesse último artigo, a representação do estado de alocação de espectro é simplificada devido à problemas de escalabilidade. Essa representação é um vetor contendo a informação da requisição de circuito óptico e da utilização do espectro nos k-menores caminhos candidatos para a requisição. Inspirados por essa abordagem, nesse trabalho foi utilizado o espectro por completo na análise para evitar qualquer perda de informação. Esses trabalhos também utilizam aprendizado por reforço profundo, e de acordo com [53], essa técnica de aprendizado profundo tende a ter resultados promissores no roteamento de tráfego.

Uma importante limitação das representações existentes é que eles não alcançam uma

boa generalização. Uma boa representação é crucial para simplificar o processo de aprendizagem através da redução no nível de abstração requerido pelo modelo de aprendizado de máquina. No trabalho [54] é proposto uma nova representação do estado da rede que captura tanto a utilização geral da rede quanto as interdependências críticas entre os caminhos resultantes da topologia de rede, de uma forma que é simples e pode ser mais facilmente explorada pelo agente para aprender melhor e mais rápido. Já em [55], os mesmos autores abordam a aplicação de aprendizagem por reforço profundo para realizar roteamento online nas redes de transporte ópticas (OTNs - *Optical Transport Networks*). Os autores desses trabalhos acreditam que a representação direta do estado da rede pode fazer com que o processo de aprendizado do roteamento fique mais complexo. Eles tentam reduzir o nível de abstração do estado da rede através de um processo de engenharia de recursos, no qual eles alavancam relacionamentos entre os *links* do qual formam um caminho ponta a ponta. Basicamente eles usam a ideia de que, de acordo com a requisição de largura de banda de uma demanda de tráfego de entrada, é fácil estimar o uso resultante dos *links* depois de alocar aquela demanda para uma caminho específico de ponta a ponta. Essa abordagem foi avaliada em uma OTN utilizando tráfego incremental, que é um cenário mais simples do que a da EON. Os resultados mostram que as representações desses trabalhos superam propostas anteriores, mas é evidente que ainda há o desafio de escolher a melhor política de roteamento considerando a incerteza das futuras demandas de tráfego.

2.4.2 Trabalhos Relacionados da Literatura com CNN em redes

Os pesquisadores tem investigado diferentes problemas na modelagem e na transmissão em EONs por quase uma década. Isso se dá devido a certas características, como a arquitetura de grade elástica e transmissores de largura de banda variável que geram desafios nas pesquisas relacionadas ao paradigma da EON. Com isso, várias tecnologias e ferramentas avançadas tem sido utilizadas para analisar esse paradigma. Uma dessas ferramentas é a aplicação de algoritmos de aprendizado de máquina, que tem ajudado as pesquisas realizando funções na EONs como por exemplo a análise de desempenho, o gerenciamento de tráfego e a identificação erros [56].

Com a viabilização e popularidade dos algoritmos de aprendizado de máquina, muitos trabalhos utilizando esses algoritmos em redes ópticas elásticas têm sido propostos na literatura nos últimos anos. Dentre os principais algoritmos utilizados, nós temos a RNA e a CNN. As técnicas existentes de aprendizado de máquina aplicadas à redes ópticas podem ser amplamente classificadas em: i) avaliação ou previsão da qualidade de transmissão (*Quality of Transmission* - QoT); ii) previsões de falhas; iii) previsão de tráfego e outros.

Identificação de formato de modulação baseado em CNN

Na área de identificação, o trabalho [57] propõe um modelo de Identificação de Formato de Modulação (*Modulation Format Identification* - MFI) baseado em CNN de baixo custo que garante uma acurácia alta através da extração e aprendizagem de características dos histogramas de amplitude assíncronos (*asynchronous amplitude histograms* - AAH). Esse modelo utiliza um fotodetector de larguras de banda baixas e detecção direta de baixa taxa de amostragem nos nodos intermediários da rede. Esse módulo em CNN garante um baixo custo enquanto mostra melhor acurácia e estabilidade de identificação do que as obtidas através de redes neurais artificiais e de aprendizado profundo utilizando a mesma complexidade computacional. O objetivo desse modulo MFI é a identificação de três formatos de modulação comuns de polarizações multiplexadas (PM-QPSK, PM-16QAM e PM-64QAM) em cenários de transmissão de longa distância.

Em cada link da fibra, um acoplador óptico 1:9 é utilizado para que apenas 10% do sinal óptico de cada link seja utilizado para realizar o reconhecimento de formato de modulação. Dentro do modulo MFI, como mostrado na Figura 2.17, o canal de comprimento de onda alvo é filtrado primeiramente por um filtro ajustável de largura de banda variável (B-VTF). Depois o sinal óptico monitorado é convertido em um sinal elétrico por um fotodetector de detecção direta de baixa largura de banda. Esse sinal elétrico é convertido para um sinal digital por um conversor de baixa velocidade (adc). Esse sinal então é recebido por um processador de sinais digitais (DSP), gerando um histograma AAH por normalização e por estatísticas de densidade de probabilidade. Após esse pré-processamento dos dados, o AAH gerado é enviado à CNN, na qual será identificado o formato de modulação.

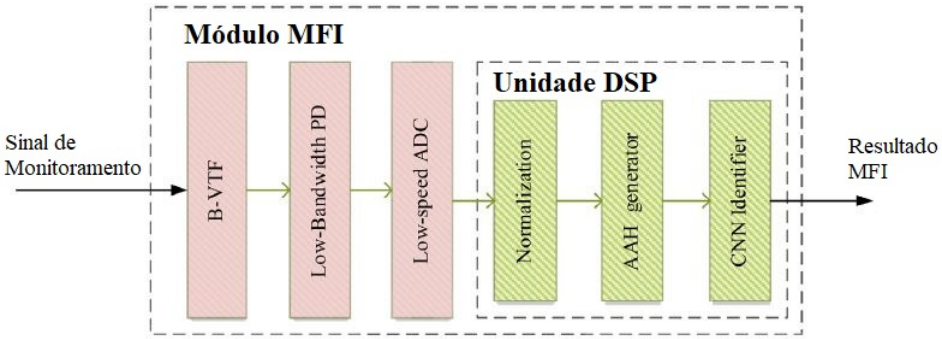


Figura 2.17: Diagrama ilustrativo do esquema MFI proposto em um nó intermediário [57]

O esquema de baixo custo proposto demonstrou uma acurácia de identificação alta nas simulações e experimentos. Os resultados do esquema MFI proposto com os três sistemas QAM mostraram que diferentes formatos de modulação podem ser identificados com 100% de acurácia com Dispersão Cromática (CD - *Chromatic Dispersion*) de 0 a

16,000 ps/nm, sobre varias condições de link como a taxa baud múltipla, relação sinal ruído óptica (*Optical Signal to Noise Ratio* - OSNR), etc. A superioridade do esquema proposto nesse trabalho mostra o potencial da ferramenta em nós intermediários sobre redes ópticas elásticas, abrindo oportunidades para um monitoramento inteligente de toda a rede óptica no futuro.

Monitor de Performance óptico baseado em CNN

Na área de monitoração, nós temos um monitor de performance óptico (*Optical Performance Monitor* - OPM) parcialmente treinável utilizando técnicas de aprendizado de máquina. Na pesquisa realizada em [58] apresenta uma OPM que utiliza CNN com um receptor digital coerente para lidar com o grande número de dados de treinamento e um pré-processamento de dados de entrada necessário para extração das características. Porém, este ainda requer uma seleção de características antes do processo de treinamento. Esse trabalho demonstra experimentalmente como a CNN aprende a estimar a OSNR, extraindo informações úteis de dados brutos medidos logo após uma detecção coerente.

Os resultados demonstram que o OPM baseado em CNN oferece um esquema de monitoramento tratável a partir dos dados brutos, na qual a estimativa de OSNR foi alcançada utilizando os sinais 14- e 16GBd DP-QPSK, 16-QAM, e 64-QAM. No trabalho [59] esse estudo é expandido para investigar os coeficientes dos filtros intermediários na CNN treinada, visando a compreensão das mecânicas dessa ferramenta para estimar a OSNR.

Identificação de erros baseada em CNN

Na área de identificação de falhas, o trabalho [60] propõe um identificador de falha leve baseado em CNN, capaz de identificar 4 falhas leves, sendo elas a mudança de filtro, aperto de filtro, ruído de emissão espontânea amplificada (*amplified spontaneous emission* - ASE) e interferência não linear de fibra (*fiber nonlinear interference* - NLI). A rede recebe como entrada a densidade do espectro de potência do sinal extraído de um compensador de dispersão cromática digital em um receptor coerente. Para gerar esses dados para o treinamento e o teste da CNN, é simulado uma falha de cada vez enquanto mantém as outras falhas nas suas condições normais. O link da fibra simulada consiste em 11 extensões com cada contendo 80 quilômetros de fibra monomodo padrão (*Standard single mode fiber* - SSMF) e um amplificador de fibra com erbium.

Os resultados expostos mostram a convergência durante o treinamento e uma acurácia de 100%. Depois é feito avaliação do identificador proposto quando existe dois tipos de deteriorações de impedimento através da parametrização dos valores de OSNR e taxa de

erro por bit (*Bit Error ratio* - BER). Os resultados mostram que essa CNN também é efetiva em distinguir a importância das causas de falha para quantas múltiplas deteriorações existem.

Capítulo 3

Classificação de Algoritmos RMLSA

Para a criação de um classificador baseado em aprendizado de máquina é preciso realizar três passos: coleta dos dados, tratamento desses dados, e a modelagem da máquina de aprendizagem. A coleta de dados consiste apenas na obtenção dos dados que serão utilizados. Já o tratamento consiste no pré-processamento em que esses dados foram submetidos para garantir a uniformidade e a clareza das informações a serem extraídas. E a modelagem é a escolha dos hiper-parâmetros que vão definir a funcionalidade do algoritmo de aprendizado de máquina. Nesse capítulo é mostrado a forma em que esses passos foram realizados para esse trabalho, explicando e justificando as escolhas tomadas para a criação do classificador.

3.1 Coleta de Dados e Pré-processamento

Para gerar os estados de rede que serão utilizados como dados de entrada, para o treinamento e para os testes, foi utilizado o simulador ONS [11]. Ele consiste em um simulador de redes ópticas de código aberto desenvolvido em Java, na qual ele utiliza eventos discretos para simular requisições de tráfego de uma rede óptica WDM ou EON através de seus problemas de roteamento e alocação (RWA e RMLSA respectivamente). Na Figura 3.1 é mostrado o diagrama de funcionamento dos 4 módulos lógicos na qual o simulador é separado. Na literatura pode ser encontrado uma variedade de ferramentas de simulação para redes ópticas de código aberto, onde muitas delas derivam ou são módulos de simuladores de rede IP (*Internet Protocol*). Mas devido as complexidades da camada IP, esses simuladores trazem complexidades desnecessárias que podem ser abstraídas por simuladores exclusivos para rede óptica. Quando trata-se de uma ferramenta com foco no paradigma de redes ópticas elásticas, foi fechado ainda mais o leque de opções. O simulador ONS teve suas funcionalidades projetadas de forma que elas permitem uma implementação fácil

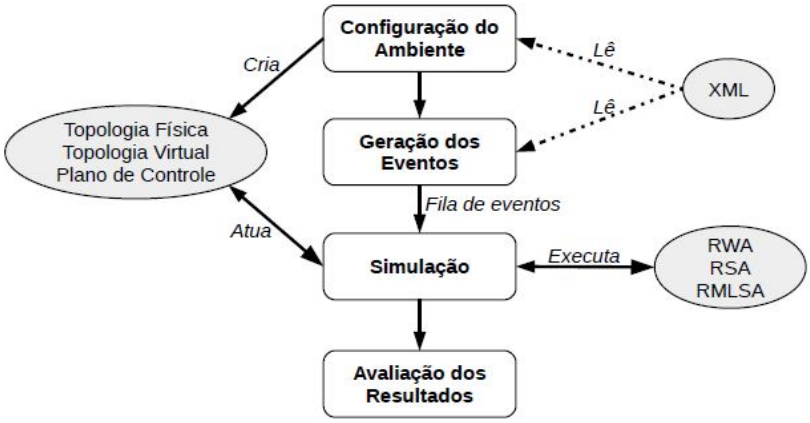
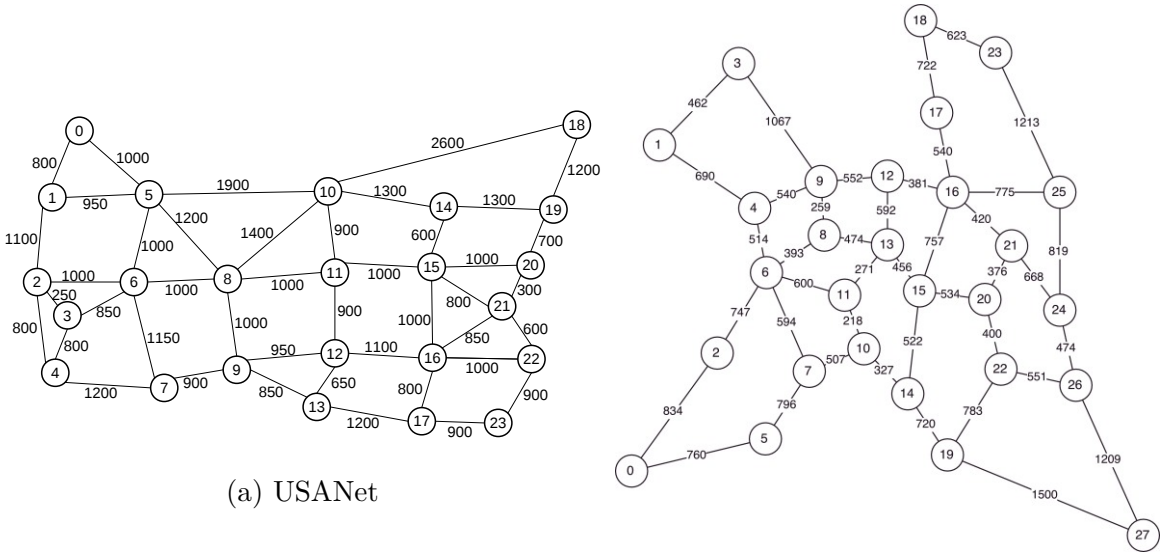


Figura 3.1: Diagrama de funcionamento do ONS [11]

e ágil de novos algoritmos, podendo fornecer assim uma grande quantidade de dados de tráfego de rede dinâmica, mesmo quando grandes topologias de redes são empregadas.

Cada simulação envolveu 100.000 requisições de conexão, onde cada simulação foi realizada 5 vezes para gerar um intervalo de confiança de 95%. Foram geradas requisições com diferentes níveis de granularidade: $12,5$Gbps, 25Gbps, 50Gbps, 100Gbps, 200Gbps e 400Gbps, distribuídos uniformemente. As requisições de conexão seguem um processo de Poisson com o tempo médio de retenção de 600 segundos, de acordo com uma distribuição exponencial negativa e uniformemente distribuída entre todos os pares de nós. As topologias consideradas nas simulações foram a USANet com 24 nós e 86 enlaces, e a PanEURO com 28 nós e 82 enlaces, como mostrado na Figura 3.2. Cada enlace possui uma banda de 4THz, divididos em intervalos de frequência de 12,5 GHz, na qual resulta em 320 *slots* em cada fibra, baseada nas definições atualmente utilizadas [61]. A banda de guarda entre dois circuitos ópticos adjacentes é assumida ser de 2 *slots* para evitar interferência. Cada nó da topologia é equipado com transmissores e receptores suficientes onde cada um possui uma capacidade de transmissão de até 32 *slots*. Foram considerados 3 modulações de sinal para cada algoritmo: BPSK (*Binary Phase Shift Keying*), QPSK (*Quadrature Phase Shift Keying*) e 8QAM (*Quadrature Amplitude Modulation*).

Como mencionado anteriormente, o desempenho do método de aprendizado de máquina é altamente dependente da escolha de representação dos dados. Tendo isso em mente, o primeiro passo para o pré processamento dos dados seria desenvolver uma representação da rede que pudesse ser aproveitada ao máximo pelas redes neurais profundas. Inspirado pelas pesquisas de estado da arte analisadas nos trabalhos relacionados sobre representação da rede óptica em aprendizado profundo, utilizamos uma matriz como forma de representação. De acordo com o número de nós e enlaces mencionados anteriormente de cada topologia, nós podemos representar as redes USANet e PanEURO como matrizes



(a) USANet

(b) PanEURO

Figura 3.2: Topologias USANet e PanEURO

de dimensão 86×320 e 82×320, para cada topologia respectivamente. Nós adaptamos o simulador ONS para que sua saída contenha uma matriz, na qual o valor de cada célula possua o valor de ocupação de cada *slot*.

O valor de cada ocupação é dada pelo número do circuito óptico da requisição que utiliza aquele *slot*. Tendo em vista que a informação que desejamos obter de cada estado de rede é quais *slots* estão sendo alocados para identificar a estratégia usada, a matriz de estado de rede foi convertida para uma matriz binária onde o valor de cada célula identifica a ocupação daquele *slot*. Mais especificamente: Células com o valor "1" determina que aquele *slot* está alocado para algum circuito óptico, incluindo a banda de guarda; e as células com valor "0" indica que aquele *slot* está vazio.

Para a criação do *dataset*, foram salvos diversos estados da rede, em diferentes momentos, durante cada simulação. Para cada requisição de conexão que chega na rede, se aceita, ela muda o estado da rede. Cada momento x foi determinado de acordo com a chegada de uma requisição x na rede. Para esse trabalho, foram considerados 95 estados de rede, igualmente distribuídos entre a chegada da requisição 6.000 e a requisição final 100.000. A análise começa com a solicitação 6.000 para mitigar a fase transitória da simulação.

Os 11 algoritmos RMLSA utilizados nas simulações (Seção 2.2) foram: K5SP *Random*, K3SP DP-*Dedicated Partition* K2SP BF-*Best Fit*, K1SP EF-*Exact Fit*, K4SP PP-*Pseudo Partition*, K4SP AP-*Acceptance Prone*, K2SP LF-*Last Fit*, K3SP FLEF-*First-Last Exact Fit*, K7SP FF-*First Fit*, K3SP CS-*Complete Sharing* e MAdapSPV. Cada algoritmo foi utilizado em 18 cargas de tráfego diferentes, que variam de 75 a 500 erlangs, com pas-

sos de 25 erlangs. Para garantir a variabilidade estatística, foram utilizados 5 sementes aleatórias e os dados foram separados na proporção de 80/20, para treinamento e teste respectivamente. Essa separação dos dados de treinamento e teste é para garantir que a rede neural fez uma generalização dos dados através do aprendizado, ao invés de aprender apenas os dados utilizados no treinamento. Desse modo, a quantidade de matrizes de estados de rede utilizados é mostrada na Equação 3.1. Esse cálculo resulta em um total de 94.050 amostras, com 75.240 para treinamento, e 18.810 para testes.

$$\begin{aligned} \text{dataset} &= momentos \times algoritmos \times cargas \times sementes \\ &= 95 \times 11 \times 18 \times 5 = 94050 \end{aligned} \tag{3.1}$$

Como será visto a seguir, foi usado 3 tipos de algoritmos de aprendizado profundo diferentes: Rede Neural Profunda (DNN - *Deep Neural Network*), Rede Neural Convolucional convencional (CNN - Convolutional Neural Network) de 2 dimensões e uma Rede Neural Convolucional unidimensional (CNN 1d). Como a entrada da DNN e da CNN 1d são vetores, foi preciso redimensionar as matrizes de entrada para uma dimensão quando essas redes fossem utilizadas. As representações obtidas da rede para as redes USANet e PanEURO foram um vetor com 27.520 posições, e um vetor com 26.240 posições para cada rede respectivamente. Logo, apenas a rede neural CNN convencional recebeu o estado de rede como uma matriz explicada anteriormente.

3.2 Hiper-parâmetros das Redes Neurais Profundas

Projetos de aprendizado de máquina diferem dos outros tipos de projetos de *software* devido a quantidade de habilidades, tecnologias e pesquisa requiridas para sua implementação. Por esses motivos, foi necessário o uso de uma linguagem de programação que fosse estável, flexível e que tivesse ferramentas disponíveis. A linguagem Python oferece todas essas características com uma grande variedade de ferramentas e tecnologias, conseguindo ainda ser simples e consistente. Utilizamos a biblioteca Keras [62] para implementar as RNAs. Essa API torna a tarefa de criar e manipular algoritmos de aprendizado de máquina mais intuitiva e fácil de entender.

Para a implementação do classificador, nós utilizamos 2 redes convolucionais diferentes: uma bidimensional (CNN2d); e uma unidimensional (CNN1d). As CNNs são usadas amplamente na área de classificação de imagens e tem avançado continuamente em sua acurácia. Mas essas redes também atuam como extratores de recursos genéricos para vários reconhecimentos de tarefas como detecção de objetos, segmentação semântica, e recuperação de imagem. Essa característica, juntamente com a representação matricial de

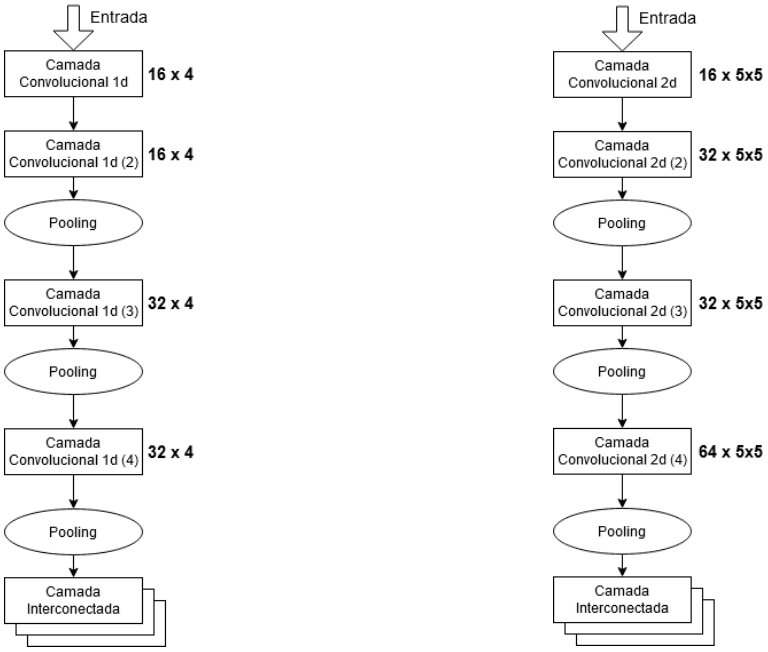
rede desenvolvida, tornou a CNN profunda em uma ferramenta interessante para pesquisa visando a classificação de algoritmos RMLSA.

Inspirados no trabalho [63], a metodologia utilizada foi desenvolver uma rede neural artificial simples com apenas algumas classes para entender melhor o problema. Após essa pesquisa inicial e com auxílio de alguns trabalhos como [64], foi ajustado os hiper-parâmetros e arquitetura da RNA para ajustar ao cenário mais complexo. Esse ajuste veio através da análise de acurácia e desempenho de inúmeros testes com diferentes arquiteturas e parametrizações da RNA. Esses hiper-parâmetros e arquiteturas da rede incluem quantidade de neurônios, taxa de aprendizado, tamanho do *batch*, número de camadas, regularização e inicialização de pesos.

Após essa optimização, a DNN resultante possui: 4 camadas ocultas com 100 neurônios cada, 26.240 neurônios na topologia USANet ou 26.240 neurônios na topologia PanEURO na camada de entrada, 11 neurônios na camada de saída (1 para cada classe de algoritmo) e 2.783.511 parâmetros treináveis. Para criar uma rede neural convolucional equivalente, adotamos como base a quantidade de parâmetros treináveis da DNN. Com isso, temos uma CNN1d , como mostrado na Figura 3.3, com: 4 camadas convolucionais com *padding*; a quantidade de filtros por camada são 16, 16, 32 e 32, necessariamente nessa ordem; com kernel de tamanho 4; 3 camadas de *maxpooling* de 1 dimensão entre as camadas de convolução; e na saída uma DNN com 3 camadas ocultas, com 100 neurônios cada, e uma camada de saída com 11 neurônios para classificação. Para a CNN2d: 4 camadas convolucionais com *padding*; a quantidade de filtros por camada são 16, 32, 64, e 64, necessariamente nessa ordem; com um kernel de tamanho 5×5; 3 camadas de *maxpooling* de 2 dimensões entre as camadas de convolução; e na saída temos a mesma DNN da CNN1d.

Na CNN1d temos 2.780.739 parâmetros treináveis, e na CNN2d temos 2.728.187 parâmetros treináveis. Esses valores são reduzidos para os 3 algoritmos quando calculados na topologia PanEURO. Isso se dá devido a diferença na dimensão da entrada gerada por cada topologia. A função de ativação da camada oculta dos três algoritmos foi a função de ativação linear retificada (ReLU) e nas camadas de saída a função de ativação *softmax*, nas quais são matematicamente representadas respectivamente nas equações 2.7 e 2.9.

Cada camada convolucional e interconectada possui uma quantidade de parâmetros. Esses parâmetros são as variáveis que interferem na tomada de decisões das redes neurais para classificação, ou seja, os pesos e os vieses (*bias*). O cálculo é realizado camada por camada e somado no final, totalizando a quantidade de parâmetros daquela rede. As equações 3.2 e 3.3 representam respectivamente o cálculo para camada convolucional (CONV) e para camada interconectada (FC). Na equação da camada convolucional, multiplica-se a dimensão $m \times n$ (largura $\times$ altura) com o número d de filtros da camada



(a) CNN1d (b) CNN2d

Figura 3.3: Arquitetura dos algoritmos CNN 1d e 2d

anterior. Soma-se com 1, que se refere ao *bias* de cada filtro, e multiplica por k, que é o número de filtros da camada atual. Na equação da camada interconectada, multiplica o número de neurônios da camada atual c com o número de neurônios da camada anterior p. Soma-se com 1, que se refere ao *bias* de cada neurônio, e multiplica por c novamente.

$$CONV = (m \times n \times d) + 1) \times k \tag{3.2}$$

$$FC = (c \times p) + 1 \times c \tag{3.3}$$

O treinamento dos 3 algoritmos foi realizado com os mesmos parâmetros, com a única diferença no formato da entrada, onde a DNN e a CNN1d recebem os estados de rede na forma de vetor, enquanto a CNN2d recebe na forma de matriz inicial. Esses parâmetros são: Tamanho de *batch* de 10 amostras, optimizador de Descida de Gradiente Estocástico com a taxa de aprendizado igual a $0,0001$, *dropout* com probabilidade de 20% e sem regularização. Nós usamos a função de erro de entropia-cruzada (*cross-entropy loss function*) para calcular o erro entre a classe atual e as previstas, de acordo com a equação 3.4, onde y_i é o $i$nésimo valor escalar da saída do modelo, t_i é o valor alvo correspondente, e k é o número de classes.

$$Erro = -\sum_{i=1}^{k} t_i log(y_i) \tag{3.4}$$

A função de ativação utilizada nos 3 algoritmos pelas camadas de convolução e densa foi a ReLU, enquanto na camada de saída foi a função softmax, devido a sua função de retornar a probabilidade de cada classe, possibilitando a categorização. Foi usado um *dropout* com 20%, na qual desconsidera essa porcentagem de neurônios em cada camada por época, visando a redução de *overfitting*. Utilizamos o algoritmo de descida do gradiente com taxa de aprendizado adaptativa Adam para a taxa de aprendizado, com uma taxa de $0,0001$. Para escolher esses parâmetros foram realizados diversos testes baseado em análise empírica, onde buscamos alcançar o melhor valor de acurácia possível.

A diferença entre as redes convolucionais 1d e 2d é basicamente o formato unidimensional da entrada e do *kernel*. Mas devido a forma em que os dados são organizados na nossa representação da rede óptica, com as linhas representando os enlaces, as colunas representando os *slots*, e a natureza do trabalho que é identificar algoritmos RMLSA, foi visto que uma análise de linha por linha seria interessante visando alcançar uma acurácia maior. Foi acreditado que a busca por padrões e parâmetros que levariam a categorização de um algoritmo seria mais fácil de obter se analisarmos um enlace por vez, evidenciando mais a estratégia de alocação de espectro.

As CNN mais avançadas utilizadas recentemente, como as CNNs profundas, demandam bastante tempo para serem treinadas e testadas. Esse custo computacional se dá devido a largura (número de filtros), profundidade (número de camadas), *strides pequenos*, e suas combinações [65]. Com a limitação de tempo para realizar o trabalho, foi escolhido realizar o treinamento com 25 épocas para cada algoritmo.

Para provar a eficácia e a viabilidade de um classificador baseado em CNN, realizamos o treinamento e a validação com cada algoritmo, ambos com diferentes grupos de amostra para garantir a generalização da ferramenta. Além disso, iremos comparar as duas CNNs e a DNN, que são equivalentes em questão de número de parâmetros aprendíveis. Tendo como objetivo verificar o impacto em utilizar uma ferramenta mais complexa, as redes convolucionais, ao invés de uma rede mais simples em comparação, as redes neurais artificiais. Utilizaremos a acurácia como métrica principal para a avaliação do desempenho dos algoritmos.

Capítulo 4

Resultados Numéricos

Nesse capítulo será exposto os resultados do treinamento e da validação dos três algoritmos de aprendizado profundo (DNN, CNN1d e CNN2d) para cada topologia (USANet e PanEURO). Esses resultados serão mostrados de duas formas: um gráfico da curva aprendizado com acurácia e erro; e uma matriz de confusão.

Uma curva de aprendizagem é um gráfico do desempenho de aprendizado ao longo do tempo (épocas). Esse gráfico é amplamente usado para avaliar e diagnosticar problemas em modelos de aprendizado de máquina. A matriz de confusão é uma tabela que permite a visualização da performance de um algoritmo, tipicamente de aprendizado supervisionado. Cada coluna da matriz representa as instâncias em uma classe prevista enquanto cada linha representa as instâncias em uma classe real. A diagonal principal representa as instâncias que foram classificados corretamente, ou seja, os Verdadeiros Positivos (VP) e os Verdadeiros Negativos (TN). Por exemplo, vamos observar os resultados para K5SP *Random* na Tabela 4.1 da matriz de confusão para o algoritmo DNN na topologia USANet:

- O classificador classificou 1.674 amostras de forma correta como K5SP *Random*;
- Em 25 amostras do K5SP *Random* foram classificadas erradas, onde 11 amostras foram classificadas como K2SP BF e 14 amostras foram classificadas como K1SP EF;
- Para finalizar, 7 amostras foram classificadas como K5SP *Random* de forma errada, sendo 2 amostras são K2SP BF e 5 amostras são K1SP EF.

Analisando primeiramente as Tabelas 4.1, 4.2 e 4.3 da topologia USANet, é possível perceber nas 3 tabelas uma concentração nos erros entre as estratégias K7SP *First Fit* e MAdapSPV. Isso é devido ao fato que o MAdapSPV, é uma versão adaptada do SPV que possui uma estratégia parecida com o *First Fit*. O classificador classificou corretamente 18.394 amostras utilizando DNN, 18.543 utilizando CNN1d, e 18.447 utilizando CNN2d.

Analisando as Tabelas 4.4, 4.5 e 4.6 da topologia PanEURO, é possível reparar em uma concentração de erros, entre as mesmas estratégias, que encontramos na topologia USANet. A reprodução desse erro fortalece a hipótese que realizamos anteriormente. Podemos reparar também que esse erro é bem mais acentuado na PanEURO. Para responder esse problemas, foram formuladas duas hipóteses:

1. Como pode ser observado na Figura 3.2, a topologia PanEURO possui menos enlaces quando comparado a topologia USANet, e alguns nós são mais isolados. Essas características fazem a topologia PanEURO ser mais restrita, tornando mais difícil para que o classificador consiga reconhecer o comportamento dos algoritmos RMLSA.

2. Ao observar que os erros continuaram majoritariamente entre os mesmos algoritmos, sendo acentuado proporcionalmente, é possível levantar a hipótese que os algoritmos na topologia PanEURO tiveram apenas mais dificuldade em diferenciar o comportamento dos algoritmos mais parecidos, ou seja, as classificações mais complexas. Tendo isso em mente, a provável fonte do problema deve ser a diferença entre parâmetros aprendíveis entre as duas topologias, devido a diferença na dimensão da entrada gerada por cada topologia.

Para avaliação de modelos de aprendizado de máquina, existem 3 métricas fundamentais: acurácia, precisão e revocação. A precisão é a mais utilizada em problemas onde os Falsos Positivos (FP) são considerados mais prejudiciais que os Falsos Negativos (FN). Revocação é mais utilizado em problemas onde os Falsos Negativos são considerados mais prejudiciais que o Falso Positivo. O problema não prioriza FN ou FP, então foi escolhido utilizar a acurácia como métrica já que esta é um bom indicativo do comportamento geral do modelo. Ele é definido como:

$$Acurácia = \frac{VP + VN}{TP + TN + FP + FN} \tag{4.1}$$

Para analisar o desenvolvimento da aprendizagem dos algoritmos de aprendizado profundo, utilizamos gráficos de curva de aprendizado. Foi feito um gráfico para a fase de treinamento e um para fase de teste, realizando esse processo para as 2 topologias, totalizando 4 gráficos. Ao observar os gráficos, é notável que o desenvolvimento da acurácia entre os 3 algoritmos de classificação são muito parecidos, tornando difícil distinguir cada um. Já na representação de perda nos gráficos de aprendizado da fase de testes nas Figuras 4.3 e 4.4, é possível reparar em uma discrepância entre a linha da DNN e as linhas da CNN1d e 2d. Essa diferença consiste não apenas na altura das curvas, mas também na inclinação e na estabilidade. Com essa observação, é possível inferir que a diferença na acurácia entre os algoritmos tende a aumentar a medida que utilizamos mais épocas. Isso

é justificável devido a maior robustez das redes convolucionais para identificar e classificar objetos quando comparado com a DNN. Já observando os gráficos das Figuras 4.1 e 4.2, é possível reparar que os três algoritmos são quase idênticos, podendo assim ser uma provável confirmação da equivalência dos três algoritmos utilizados, quando implementados com a mesma quantidade de parâmetros.

Para finalizar, o gráfico da Figura 4.5 mostra as acurácias entre os algoritmos nas duas topologias. Eles apresentam uma acurácia na topologia USANet de: 0,9779 para DNN, 0,9858 para CNN1d, e 0,9807 para CNN2d. Na topologia PanEURO: 0,9112 para DNN, 0,9191 para CNN1d, e 0,9196 para CNN2d. É notável apenas uma pequena diferença entre eles, onde os algoritmos de convolução foram ligeiramente melhores. A justificativa desse acontecimento é que devido a pequena quantidade de épocas, os algoritmos de convolução tiveram pouco espaço para se destacar em relação a DNN, na qual seria evidenciada com um número maior de épocas.

É notável com os gráficos de curva de aprendizado das duas topologias, que a acurácia pode ser melhorada para todos algoritmos realizando o treinamento com mais épocas. Esses resultados evidenciam a viabilidade da ferramenta, baseada em CNN, para extrair informação significante de uma amostra de alocação de espectro e utilizá-la para identificar qual estratégia foi utilizada com uma boa acurácia.

		Classe Prevista										
		K5SP Random	K3SP DP	K2SP BF	K1SP EF	K4SP PP	K4SP AP	K2SP LF	K3SP FLEF	K7SP FF	K3SP CS	MAdapSPV
Classe Real	K5SP Random	1674	0	11	14	0	0	0	0	0	0	0
	K3SP DP	0	1658	0	0	0	0	0	0	0	0	0
	K2SP BF	2	0	1700	0	0	0	31	1	0	0	0
	K1SP EF	5	0	0	1707	0	1	0	0	24	0	10
	K4SP PP	0	0	0	0	1745	0	0	0	0	0	0
	K4SP AP	0	0	0	1	0	1771	0	0	2	12	1
	K2SP LF	0	0	2	0	0	0	1751	0	0	0	0
	K3SP FLEF	0	0	0	0	0	0	20	1641	0	0	0
	K7SP FF	0	0	0	0	0	1	0	0	1525	7	138
	K3SP_CS	0	0	0	0	0	1	0	0	7	1705	6
	MAdapMSP	0	0	0	0	1	0	0	0	118	0	1517

Tabela 4.1: Matriz de confusão DNN USANet

		Classe Prevista										
		K5SP Random	K3SP DP	K2SP BF	K1SP EF	K4SP PP	K4SP AP	K2SP LF	K3SP FLEF	K7SP FF	K3SP CS	MAdapSPV
Classe Real	K5SP Random	1699	0	1	5	0	0	0	0	0	0	0
	K3SP DP	0	1701	0	0	0	0	0	0	0	0	0
	K2SP BF	0	0	1794	0	0	0	10	0	0	0	0
	K1SP EF	0	0	0	1682	0	1	0	0	2	0	1
	K4SP PP	0	0	0	0	1768	0	0	0	0	0	0
	K4SP AP	0	0	0	1	0	1697	0	0	1	5	1
	K2SP LF	0	0	1	0	0	0	1652	0	0	0	0
	K3SP FLEF	0	0	0	0	0	0	25	1658	0	0	0
	K7SP FF	0	0	0	0	0	0	0	0	1600	5	90
	K3SP_CS	0	0	0	0	0	1	0	0	3	1706	3
	MAdapMSP	0	0	0	0	0	0	0	0	111	1	1586

Tabela 4.2: Matriz de confusão CNN1d USANet

		Classe Prevista										
		K5SP Random	K3SP DP	K2SP BF	K1SP EF	K4SP PP	K4SP AP	K2SP LF	K3SP FLEF	K7SP FF	K3SP CS	MAdapSPV
Classe Real	K5SP Random	1760	0	3	1	0	0	0	0	0	0	0
	K3SP DP	0	1723	0	0	0	0	0	0	0	0	0
	K2SP BF	3	0	1693	0	0	0	18	6	0	0	1
	K1SP EF	0	0	0	1739	0	1	0	0	2	0	0
	K4SP PP	0	0	0	0	1690	0	0	0	0	0	0
	K4SP AP	0	0	0	0	0	1666	0	0	7	17	2
	K2SP LF	0	0	0	0	0	0	1720	0	0	0	0
	K3SP FLEF	0	0	0	0	0	0	2	1689	0	0	0
	K7SP FF	0	0	0	0	0	5	0	0	1476	74	125
	K3SP_CS	0	0	0	0	0	0	0	0	1	1690	0
	MAdapMSP	0	0	0	0	0	0	0	0	83	13	1601

Tabela 4.3: Matriz de confusão CNN2d USANet

		Classe Prevista										
		K5SP Random	K3SP DP	K2SP BF	K1SP EF	K4SP PP	K4SP AP	K2SP LF	K3SP FLEF	K7SP FF	K3SP CS	MAdapSPV
Classe Real	K5SP Random	1629	0	9	20	0	0	0	2	0	0	0
	K3SP DP	0	1768	0	0	0	0	0	0	0	0	0
	K2SP BF	8	0	1658	0	0	0	6	0	0	0	1
	K1SP EF	16	0	0	1639	0	0	0	0	5	0	9
	K4SP PP	0	0	0	0	1764	0	0	0	0	0	1
	K4SP AP	0	0	0	0	0	1732	0	0	0	3	0
	K2SP LF	0	0	1	0	0	0	1713	5	0	0	0
	K3SP FLEF	0	0	3	0	0	0	64	1657	0	0	0
	K7SP FF	0	0	0	0	0	0	0	0	1552	22	161
	K3SP_CS	0	0	0	0	0	1	0	0	0	1659	0
	MAdapMSP	0	1	0	1	0	0	0	0	1316	15	369

Tabela 4.4: Matriz de confusão DNN PanEURO

		Classe Prevista										
		K5SP Random	K3SP DP	K2SP BF	K1SP EF	K4SP PP	K4SP AP	K2SP LF	K3SP FLEF	K7SP FF	K3SP CS	MAdapSPV
Classe Real	K5SP Random	1753	0	1	2	0	0	0	0	0	0	0
	K3SP DP	0	1710	0	0	0	0	0	0	0	3	0
	K2SP BF	0	0	1764	0	0	0	2	0	0	0	0
	K1SP EF	1	1	0	1690	0	0	0	0	0	0	1
	K4SP PP	0	0	0	0	1664	0	0	0	0	0	0
	K4SP AP	0	0	0	0	0	1692	0	0	0	3	0
	K2SP LF	0	0	0	0	0	0	1687	2	0	0	0
	K3SP FLEF	0	0	0	0	0	0	28	1644	0	0	0
	K7SP FF	0	0	0	0	0	0	0	0	1516	1	179
	K3SP_CS	0	0	0	0	0	1	0	0	0	1707	1
	MAdapMSP	0	0	0	0	0	0	0	0	1292	3	462

Tabela 4.5: Matriz de confusão CNN1d PanEURO

		Classe Prevista										
		K5SP Random	K3SP DP	K2SP BF	K1SP EF	K4SP PP	K4SP AP	K2SP LF	K3SP FLEF	K7SP FF	K3SP CS	MAdapSPV
Classe Real	K5SP Random	1742	0	7	3	0	0	0	0	0	0	0
	K3SP DP	0	1695	0	0	0	0	0	0	0	0	0
	K2SP BF	3	0	1679	0	1	0	1	0	0	0	0
	K1SP EF	0	0	1	1695	0	0	0	0	0	0	0
	K4SP PP	0	0	0	0	1727	0	0	0	0	0	0
	K4SP AP	0	0	0	1	0	1720	0	0	0	0	0
	K2SP LF	0	0	0	0	0	0	1682	6	0	0	0
	K3SP FLEF	0	0	0	0	0	0	6	1690	0	0	0
	K7SP FF	0	0	0	0	0	0	0	0	1543	0	170
	K3SP_CS	0	0	0	0	0	5	0	0	0	1702	4
	MAdapMSP	0	0	0	0	0	0	0	0	1304	0	462

Tabela 4.6: Matriz de confusão CNN2d PanEURO

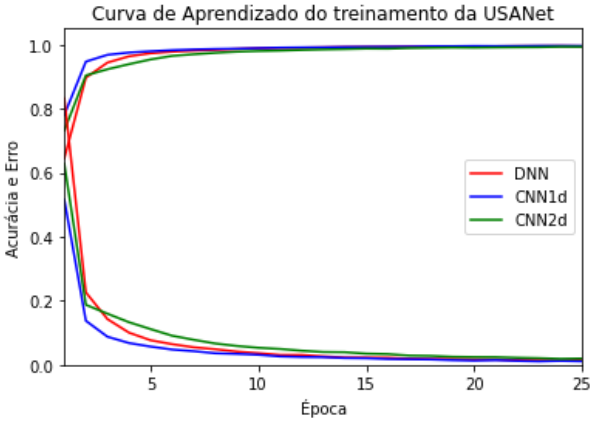


Figura 4.1: Curva de aprendizado USANet - Treinamento

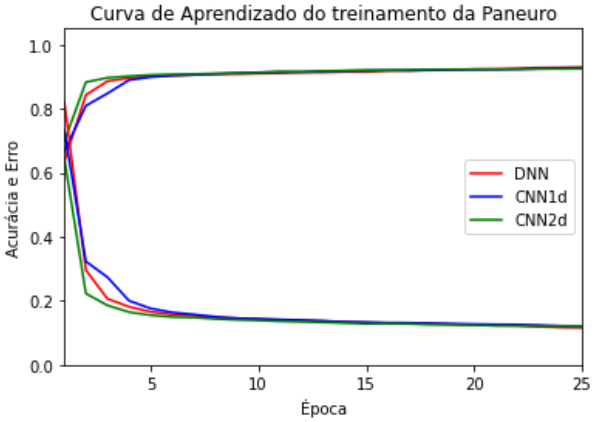


Figura 4.2: Curva de aprendizado PanEURO - Treinamento

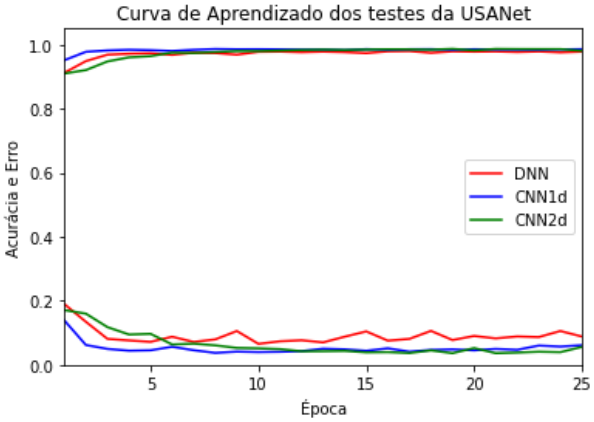


Figura 4.3: Curva de aprendizado USANet - Teste

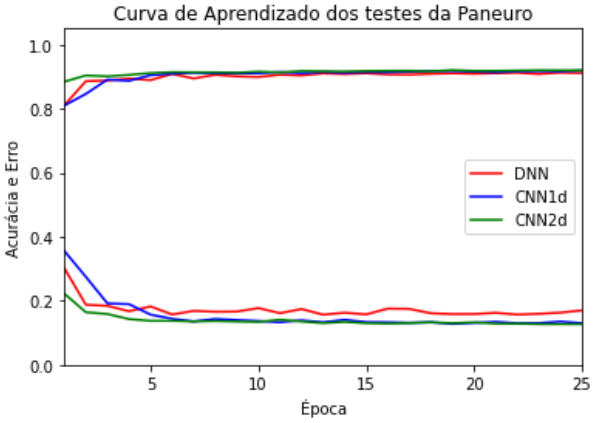


Figura 4.4: Curva de aprendizado PanEURO - Teste

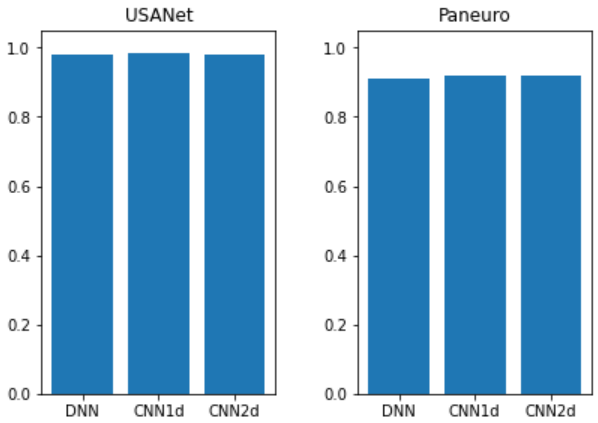


Figura 4.5: Gráfico de Acurácia

Capítulo 5

Conclusões e Sugestões para Trabalhos Futuros

Esse trabalho propõe um Classificador de algoritmos RMLSA baseado em Redes Neurais Convolucionais em Redes Ópticas Elásticas capaz de diferenciar 11 estratégias de alocação de recursos diferentes, avaliada em duas topologias diferentes. Esse classificador alcançou mais de 90% de acurácia em todos os casos, com indícios de espaço para aperfeiçoamentos, e 98,5% de acurácia no melhor caso, utilizando uma rede convolucional de uma dimensão na topologia USANet. Evidenciamos uma melhora mínima de se utilizar uma rede convolucional ao invés de um rede neural artificial profunda para realizar a tarefa de classificação através de uma comparação entre algoritmos equivalentes. Devido a complexidade da CNN, constatamos que a utilização dessa rede ao invés de uma DNN é desnecessária, devido ao baixo ganho de acurácia. Porém ainda há espaço para mais testes e otimizações a serem feitas com a CNN, que não puderam ser otimizadas devido ao limite de tempo e de recursos impostos a esse trabalho. Com isso, atendemos ao primeiro passo para criação de técnicas adaptativas para redes ópticas elásticas.

A contribuição desse trabalho é relacionada a engenharia de tráfego, considerando que entre os desafios de implantação de redes ópticas elásticas, a escolha dos algoritmos de roteamento, nível de modulação e alocação de espectro (RMLSA) certamente desempenham um importante papel. Atualmente, os algoritmos propostos na literatura usam estratégias fixas, e esse mecanismo não combina com o cenário dinâmico de uma rede óptica. Isso é devido a chegada e saída de requisições de conexões, e à incerteza do tráfego futuro. Nesse sentido, extrair informação de um vislumbre do estado de alocação de espectro da rede abre o caminho para o desenvolvimento de algoritmos RMLSA que pode se adaptar ao estado da rede, fazendo melhor uso dos recursos da rede.

Sugestões para Trabalhos Futuros

Com a verificação da viabilidade da extração de informações de uma rede óptica através de um classificador baseado em redes neurais convolucionais, o próximo passo a ser dado seria implementar um modelo baseado em aprendizado profundo que consiga classificar a eficiência da estratégia de alocação de recursos aplicada à rede óptica elástica baseando-se apenas em um vislumbre da alocação de espectro da rede. Além disso, é sugerido como trabalho futuro uma expansão da pesquisa realizada nesse trabalho através da utilização de algoritmos de rede convolucionais profundas desenvolvidas e mais reconhecidas na literatura, como a ResNet e a Inception V4 [66]. Com isso, poderia ser feita uma análise profunda com a utilização de uma rede mais robusta e complexa em relação à generalização da identificação de objetos proporcionada por algoritmos de redes convolucionais.

Referências

[1] Saleh, Adel AM e Jane M Simmons: *Technology and architecture to enable the explosive growth of the internet*. IEEE Communications Magazine, 49(1):126–132, 2011. 1

[2] Jinno, M, H Takara e B Kozicki: *Dynamic optical mesh networks: drivers, challenges and solutions for the future*. Em *2009 35th European Conference on Optical Communication*, páginas 1–4. IEEE, 2009. 1

[3] Sato, Ken ichi e Hiroshi Hasegawa: *Optical networking technologies that will create future bandwidth-abundant networks*. Journal of Optical Communications and Networking, 1(2):A81–A93, 2009. 1

[4] Chralyvy, Andrew: *Plenary paper: The coming capacity crunch*. Em *2009 35th European Conference on Optical Communication*, páginas 1–1. IEEE, 2009. 2

[5] Ellis, AD, N Mac Suibhne, D Saad e DN Payne: *Communication networks beyond the capacity crunch*, 2016. 2

[6] Waldman, Helio: *The impending optical network capacity crunch*. Em *2018 SBFoton International Optics and Photonics Conference (SBFoton IOPC)*, páginas 1–4. IEEE, 2018. 2

[7] Musumeci, Francesco, Cristina Rottondi, Avishek Nag, Irene Macaluso, Darko Zibar, Marco Ruffini e Massimo Tornatore: *An overview on application of machine learning techniques in optical networks*. IEEE Communications Surveys & Tutorials, 21(2):1383–1408, 2018. 2

[8] *Artificial intelligence (ai) methods in optical networks: A comprehensive survey*. Optical Switching and Networking, 28:43 – 57, 2018, ISSN 1573-4277. 2

[9] Goodfellow, Ian, Yoshua Bengio e Aaron Courville: *Deep learning*. MIT press, 2016. 2, 13, 18

[10] Jinno, M., H. Takara, B. Kozicki, Y. Tsukishima, Y. Sone e S. Matsuoka: *Spectrum-efficient and scalable elastic optical path network: architecture, benefits, and enabling technologies*. IEEE Communications Magazine, 47(11):66–73, November 2009, ISSN 0163-6804. 2

[11] Costa, LR, LS de Sousa, FR de Oliveira, KA da Silva, PJS Júnior e AC Drummond: *Ons: Optical network simulator-wdm/eon*, 2016. viii, 3, 30, 31

[12] Chatterjee, Bijoy Chand, Nityananda Sarma e Eiji Oki: *Routing and spectrum allocation in elastic optical networks: A tutorial*. IEEE Communications Surveys & Tutorials, 17(3):1776–1800, 2015. viii, 5, 6, 7

[13] Jinno, Masahiko, Hidehiko Takara, Bartlomiej Kozicki, Yukio Tsukishima, Yoshiaki Sone e Shinji Matsuoka: *Spectrum-efficient and scalable elastic optical path network: architecture, benefits, and enabling technologies*. IEEE communications magazine, 47(11):66–73, 2009. 5

[14] Zhang, Guoying, Marc De Leenheer, Annalisa Morea e Biswanath Mukherjee: *A survey on ofdm-based elastic core optical networking*. IEEE Communications Surveys & Tutorials, 15(1):65–87, 2012. 5

[15] LIRA, Clayton José Natal de: *Utilização de meta-heurística e de divisão espectral para alocação eficiente de espectro em redes ópticas elásticas*. Tese de Mestrado, Universidade Federal de Pernambuco, 2016. viii, 6

[16] Mukherjee, Biswanath: *Optical WDM networks*. Springer Science & Business Media, 2006. 6

[17] Tomkos, Ioannis, Siamak Azodolmolky, Josep Sole-Pareta, Davide Careglio e Eleni Palkopoulou: *A tutorial on the flexible optical networking paradigm: State of the art, trends, and research challenges*. Proceedings of the IEEE, 102(9):1317–1337, 2014. 6

[18] Christodoulopoulos, Konstantinos, Ioannis Tomkos e Emmanuel A Varvarigos: *Elastic bandwidth allocation in flexible ofdm-based optical networks*. Journal of Lightwave Technology, 29(9):1354–1366, 2011. 7

[19] Brander, Andrew William e Mark C Sinclair: *A comparative study of k-shortest path algorithms*. Em *Performance Engineering of Computer and Telecommunications Systems*, páginas 370–379. Springer, 1996. 8

[20] Wan, X., N. Hua e X. Zheng: *Dynamic routing and spectrum assignment in spectrum-flexible transparent optical networks*. IEEE/OSA Journal of Optical Communications and Networking, 4(8):603–613, Aug 2012. 8, 11

[21] Chatterjee, B.C., N. Sarma e E. Oki: *Routing and spectrum allocation in elastic optical networks: A tutorial*. IEEE Communications Surveys Tutorials, 17(3):1776–1800, thirdquarter 2015, ISSN 1553-877X. 8, 9

[22] Chatterjee, Bijoy Chand, Waya Fadini e Eiji Oki: *A spectrum allocation scheme based on first–last-exact fit policy for elastic optical networks*. Journal of Network and Computer Applications, 68:164–172, 2016. 9

[23] Ahumada Cortes, R., A. Leiva Lopez, F. Alonso Villalobos, S. Fingerhuth Massmann e G. Farias Castro: *Spectrum allocation algorithms for elastic dwdm networks on dynamic operation*. IEEE Latin America Transactions, 12(6):1012–1018, Sep. 2014, ISSN 1548-0992. 10

[24] Wang, R. e B. Mukherjee: *Spectrum management in heterogeneous bandwidth networks*. Em *2012 IEEE Global Communications Conference (GLOBECOM)*, páginas 2907–2911, Dec 2012. 10, 11

[25] Shen, J., J. Chen e Y. Sun: *Fragmentation aware routing and spectrum assignment algorithm for elastic optical networks*. Em *TENCON 2015 - 2015 IEEE Region 10 Conference*, páginas 1–4, 2015. 10

[26] Jain, Anil K, Jianchang Mao e K Moidin Mohiuddin: *Artificial neural networks: A tutorial*. Computer, 29(3):31–44, 1996. 11, 12

[27] Academy, Data Science: *Início*, Jul 2019. `https://deeplearningbook.com.br/`. viii, 12

[28] Mol, Adriano A, Luiz S Martins-Filho, José Demisio S da Silva e Ronilson Rocha: *Efficiency parameters estimation in gemstones cut design using artificial neural networks*. Computational materials science, 38(4):727–736, 2007. viii, 12

[29] McCulloch, Warren S e Walter Pitts: *A logical calculus of the ideas immanent in nervous activity*. The bulletin of mathematical biophysics, 5(4):115–133, 1943. 13

[30] Duch, Wlodzislaw e Norbert Jankowski: *Survey of neural transfer functions*. Neural Computing Surveys, 2(1):163–212, 1999. 14

[31] LeCun, Yann, Yoshua Bengio e Geoffrey Hinton: *Deep learning*. nature, 521(7553):436–444, 2015. 16, 17, 19

[32] Deng, Li e Dong Yu: *Deep learning: methods and applications*. Foundations and trends in signal processing, 7(3–4):197–387, 2014. 16

[33] Bengio, Yoshua, Aaron Courville e Pascal Vincent: *Representation learning: A review and new perspectives*. IEEE transactions on pattern analysis and machine intelligence, 35(8):1798–1828, 2013. 16

[34] Russell, Stuart e Peter Norvig: *Artificial intelligence: a modern approach*. 2002. 17

[35] Bottou, Léon e Olivier Bousquet: *The tradeoffs of large scale learning*. Advances in neural information processing systems, 20:161–168, 2007. 17

[36] Bottou, Léon: *Stochastic gradient descent tricks*. Em *Neural networks: Tricks of the trade*, páginas 421–436. Springer, 2012. viii, 18

[37] LeCun, Yann *et al.*: *Generalization and network design strategies*. Connectionism in perspective, 19:143–155, 1989. 18

[38] LeCun, Yann, Yoshua Bengio *et al.*: *Convolutional networks for images, speech, and time series*. The handbook of brain theory and neural networks, 3361(10):1995, 1995. 18

[39] Alom, Md Zahangir, Tarek M Taha, Chris Yakopcic, Stefan Westberg, Paheding Sidike, Mst Shamima Nasrin, Mahmudul Hasan, Brian C Van Essen, Abdul AS Awwal e Vijayan K Asari: *A state-of-the-art survey on deep learning theory and architectures*. Electronics, 8(3):292, 2019. viii, 19

[40] Bouvrie, Jake: *Notes on convolutional neural networks*. 2006. 19

[41] RaghavPrabhu: *Understanding of convolutional neural network (cnn) — deep learning*, Mar 2018. `https://medium.com/@RaghavPrabhu/understanding-of-convolutional-neural-network-cnn-deep-learning-99760835f148`. viii, 20, 21, 22, 23

[42] Krizhevsky, Alex, Ilya Sutskever e Geoffrey E Hinton: *Imagenet classification with deep convolutional neural networks*. Em *Advances in neural information processing systems*, páginas 1097–1105, 2012. 22

[43] gudikandula purnasai: *A beginner intro to convolutional neural networks*, Mar 2019. `https://medium.com/@purnasaigudikandula/a-beginner-intro-to-convolutional-neural-networks-684c5620c2ce`. viii, 23, 24

[44] Lee, Chen Yu, Patrick W Gallagher e Zhuowen Tu: *Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree*. Em *Artificial intelligence and statistics*, páginas 464–472, 2016. 23

[45] Zhou, Yi Tong e Rama Chellappa: *Computation of optical flow using a neural network*. Em *ICNN*, páginas 71–78, 1988. 23

[46] Ranzato, Marc'Aurelio, Fu Jie Huang, Y Lan Boureau e Yann LeCun: *Unsupervised learning of invariant feature hierarchies with applications to object recognition*. Em *2007 IEEE conference on computer vision and pattern recognition*, páginas 1–8. IEEE, 2007. 24

[47] Murray, Naila e Florent Perronnin: *Generalized max pooling*. Em *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, páginas 2473–2480, 2014. 24

[48] Khan, Asifullah, Anabia Sohail, Umme Zahoora e Aqsa Saeed Qureshi: *A survey of the recent architectures of deep convolutional neural networks*. Artificial Intelligence Review, páginas 1–62, 2020. 24

[49] Valadarsky, Asaf, Michael Schapira, Dafna Shahaf e Aviv Tamar: *Learning to route*. Em *Proceedings of the 16th ACM workshop on hot topics in networks*, páginas 185–191, 2017. 25

[50] Stampa, Giorgio, Marta Arias, David Sánchez-Charles, Victor Muntés-Mulero e Albert Cabellos: *A deep-reinforcement learning approach for software-defined networking routing optimization*. arXiv preprint arXiv:1709.07080, 2017. 25

[51] Chen, Xiaoliang, Jiannan Guo, Zuqing Zhu, Roberto Proietti, Alberto Castro e SJ Ben Yoo: *Deep-rmsa: A deep-reinforcement-learning routing, modulation and spectrum assignment agent for elastic optical networks.* Em *2018 Optical Fiber Communications Conference and Exposition (OFC)*, páginas 1–3. IEEE, 2018. 25

[52] Chen, Xiaoliang, Baojia Li, Roberto Proietti, Hongbo Lu, Zuqing Zhu e SJ Ben Yoo: *Deeprmsa: A deep reinforcement learning framework for routing, modulation and spectrum assignment in elastic optical networks.* Journal of Lightwave Technology, 37(16):4155–4163, 2019. 25

[53] Boutaba, Raouf, Mohammad A Salahuddin, Noura Limam, Sara Ayoubi, Nashid Shahriar, Felipe Estrada-Solano e Oscar M Caicedo: *A comprehensive survey on machine learning for networking: evolution, applications and research opportunities.* Journal of Internet Services and Applications, 9(1):16, 2018. 25

[54] Suarez-Varela, Jose, Albert Mestres, Junlin Yu, Li Kuang, Haoyu Feng, Pere Barlet-Ros e Albert Cabellos-Aparicio: *Feature engineering for deep reinforcement learning based routing.* Em *ICC 2019-2019 IEEE International Conference on Communications (ICC)*, páginas 1–6. IEEE, 2019. 26

[55] Suárez-Varela, José, Albert Mestres, Junlin Yu, Li Kuang, Haoyu Feng, Albert Cabellos-Aparicio e Pere Barlet-Ros: *Routing in optical transport networks with deep reinforcement learning.* IEEE/OSA Journal of Optical Communications and Networking, 11(11):547–558, 2019. 26

[56] Choudhury, Panchali Datta e Tanmay De: *Recent developments in elastic optical networks using machine learning.* Em *2019 21st International Conference on Transparent Optical Networks (ICTON)*, páginas 1–3. IEEE, 2019. 26

[57] Du, Jinhao, Tao Yang, Xue Chen, Jia Chai, Yan Zhao e Sheping Shi: *A cnn-based cost-effective modulation format identification scheme by low-bandwidth direct detecting and low rate sampling for elastic optical networks.* Optics Communications, página 126007, 2020. viii, 27

[58] Tanimura, Takahito, Takeshi Hoshida, Tomoyuki Kato, Shigeki Watanabe e Hiroyuki Morikawa: *Data-analytics-based optical performance monitoring technique for optical transport networks.* Em *Optical Fiber Communication Conference*, páginas Tu3E–3. Optical Society of America, 2018. 28

[59] Tanimura, Takahito, Takeshi Hoshida, Tomoyuki Kato, Shigeki Watanabe e Hiroyuki Morikawa: *Convolutional neural network-based optical performance monitoring for optical transport networks.* Journal of Optical Communications and Networking, 11(1):A52–A59, 2019. 28

[60] Lun, Huazhi, Qunbi Zhuge, Mengfan Fu, Yiwen Wu, Qiaoya Liu, Meng Cai, Xiaobo Zeng e Weisheng Hu: *Soft failure identification in optical networks based on convolutional neural network.* Em *45th European Conference on Optical Communication (ECOC 2019)*, páginas 1–3. IET, 2019. 28

[61] Walkowiak, K., M. Klinkowski e P. Lechowicz: *Dynamic routing in spectrally spatially flexible optical networks with back-to-back regeneration.* IEEE/OSA Journal of Optical Communications and Networking, 10(5):523–534, May 2018, ISSN 1943-0620. 31

[62] Team, Keras: *Simple. flexible. powerful.* `https://keras.io/`. 33

[63] Smith, Leslie N: *A disciplined approach to neural network hyper-parameters: Part 1–learning rate, batch size, momentum, and weight decay.* arXiv preprint arXiv:1803.09820, 2018. 34

[64] Glorot, Xavier e Yoshua Bengio: *Understanding the difficulty of training deep feedforward neural networks.* Em *Proceedings of the thirteenth international conference on artificial intelligence and statistics*, páginas 249–256, 2010. 34

[65] He, Kaiming e Jian Sun: *Convolutional neural networks at constrained time cost.* Em *Proceedings of the IEEE conference on computer vision and pattern recognition*, páginas 5353–5360, 2015. 36

[66] Szegedy, Christian, Sergey Ioffe, Vincent Vanhoucke e Alex Alemi: *Inception-v4, inception-resnet and the impact of residual connections on learning.* arXiv preprint arXiv:1602.07261, 2016. 44