Universidade Estadual de Campinas
Faculdade de Engenharia Elétrica e de Computação

Samuel Borges Ferreira Gomes

Sistema de Comunicação Baseado em Deep Learning sob Ambiente de Desvanecimento Geral

CAMPINAS
2021

Samuel Borges Ferreira Gomes

Sistema de Comunicação Baseado em Deep Learning sob Ambiente de Desvanecimento Geral

Dissertação apresentada à Faculdade de Engenharia Elétrica e de Computação da Universidade Estadual de Campinas como parte dos requisitos exigidos para a obtenção do título de Mestre em Engenharia Elétrica, na Área de Telecomunicações.

Orientador: Prof. Dr. Michel Daoud Yacoub

Este trabalho corresponde à versão final da Dissertação defendida por Samuel Borges Ferreira Gomes e orientada pelo Prof. Dr. Michel Daoud Yacoub.

CAMPINAS
2021

Ficha catalográfica
Universidade Estadual de Campinas
Biblioteca da Área de Engenharia e Arquitetura
Rose Meire da Silva - CRB 8/5974

Gomes, Samuel Borges Ferreira, 1993-
G585s Sistema de comunicação baseado em deep learning sob ambiente de desvanecimento geral / Samuel Borges Ferreira Gomes. – Campinas, SP : [s.n.], 2021.

Orientador: Michel Daoud Yacoub.
Dissertação (mestrado) – Universidade Estadual de Campinas, Faculdade de Engenharia Elétrica e de Computação.

1. Sistemas de comunicação. 2. Rádio - Transmissões e transmissão - Desvanecimento. 3. Aprendizado profundo. 4. Redes neurais covolucionais. I. Yacoub, Michel Daoud, 1955-. II. Universidade Estadual de Campinas. Faculdade de Engenharia Elétrica e de Computação. III. Título.

Informações para Biblioteca Digital

Título em outro idioma: Deep learning based communication system under general fading environment
Palavras-chave em inglês:
Communication systems
Radio - Fading
Deep learning
Convolutional neural networks
Área de concentração: Telecomunicações e Telemática
Titulação: Mestre em Engenharia Elétrica
Banca examinadora:
Michel Daoud Yacoub [Orientador]
Carlos Rafael Nogueira da Silva
Rausley Adriano Amaral de Souza
Data de defesa: 16-07-2021
Programa de Pós-Graduação: Engenharia Elétrica

Identificação e informações acadêmicas do(a) aluno(a)
- ORCID do autor: https://orcid.org/0000-0003-3170-1347
- Currículo Lattes do autor: http://lattes.cnpq.br/0334996194736624

COMISSÃO JULGADORA - DISSERTAÇÃO DE MESTRADO

Candidato: Samuel Borges Ferreira Gomes **RA:** 261663

Data da defesa: 16/07/2021

Título da Dissertação: "Sistema de Comunicação Baseado em Deep Learning sob Ambiente de Desvanecimento Geral"

Membros da Comissão Julgadora:

Prof. Dr. Michel Daoud Yacoub (Presidente)
UNICAMP

Prof. Dr. Carlos Rafael Nogueira da Silva (Titular Interno)
UNICAMP

Prof. Dr. Rausley Adriano Amaral de Souza (Titular Externo)
INATEL

A ata da defesa, assinada pelos membros da Comissão Examinadora, consta no SIGA/Sistema de Fluxo de Dissertação/Tese e na Secretaria do Programa da Unidade.

Mathematics requires a small dose,
not of genius,
but of an imaginative freedom which,
in a larger dose,
would be insanity.

(Angus K. Rodgers)

Agradecimentos

Agradeço,

A Deus, sustentáculo vital em minha vida, por prover muito mais do que preciso, e por me abençoar muito mais do que mereço.

Aos meus pais, Miriam e José Nilson, nos quais tenho máxima admiração e respeito. Obrigado pela compreensão, pela paciência, pelo amor, e por todos os valores que me ensinaram. Acima de tudo, obrigado pelos sacrifícios que vocês fizeram para que eu pudesse ser quem eu sou hoje. Amo vocês.

Aos meus irmãos, Camilla, Victor, Larissa, André e Luana, por proporcionarem momentos inesquecíveis na minha vida, onde cada um contribuiu de maneira especial para minha formação humana.

À minha noiva, companheira, e melhor amiga Carla, que esteve comigo nos momentos tristes e felizes, e com paciência e amor sempre me ajudou. Mais uma conquista nossa.

Ao meu orientador, Prof. Dr. Michel Daoud Yacoub, pela acolhida e oportunidade concedida, que de forma maestral conduziu-me durante esta jornada. Meus sinceros agradecimentos.

À FEEC/UNICAMP, pelo seu ensino de excelência e bom ambiente de pesquisa.

A todos os colegas que conheci, guardo lembranças especiais de cada um de vocês.

À CAPES, agência financiadora, pelo programa de excelência acadêmica PROEX processo 88887.373496/2019-00, responsável pelo suporte financeiro durante todo o programa de mestrado, sendo fundamental para conclusão deste trabalho.

Resumo

Esta dissertação ilustra o potencial de *deep learning* aplicado a sistemas de comunicação. Para isso, é feita uma releitura de um sistema de comunicação tradicional completo como uma técnica de aprendizado profundo em termos de um *Autoencoder*, sem nenhum conhecimento a priori de engenharia de comunicações, exceto pela modelagem estatística do canal. Para a análise, consideramos que o sistema inteligente se comunica por um canal de desvanecimento geral seguindo as estatísticas α-μ, escolhida tanto por sua flexibilidade, quanto por sua aplicabilidade em cenários reais. O sistema inteligente consiste em duas redes neurais convolucionais emulando o codificador e o decodificador do sistema de comunicação, otimizadas conjuntamente por uma única função objetivo. O desempenho do sistema é analisado em termos da taxa de erro na mensagem que, para comparação e propósitos de consistência, também é apresentado analiticamente para esquemas tradicionais de modulação. Os resultados de simulação mostram que a arquitetura estudada corresponde a soluções ótimas de engenharia humana, mesmo sem o dóminio de conhecimento específico. Além disso, a arquitetura profunda é capaz de convergir rapidamente mantendo sua capacidade de generalização sob diferentes comprimentos de bloco da mensagem enviada, relações sinal-ruído, taxas de códigos e ambientes de desvanecimento.

Palavras-chave: Sistemas de comunicação; modelo de desvanecimento α-μ, *deep learning*; aprendizado fim-a-fim; redes neurais convolucionais.

Abstract

This dissertation provides an illustration of the deep learning potential applied to communication systems. For this, a complete traditional communication system is re-defined as a deep learning technique in terms of an Autoencoder, with no prior engineering knowledge of communications, except for the statistical modeling of the channel. For analysis, we consider that the intelligent system communicates through a general fading channel, following the statistics α-μ, chosen both for its flexibility and for its applicability in real scenarios. The intelligent system consists of two convolutional neural networks simulating the encoder and decoder of the communication scheme, optimized together by a single objective function. The performance of the system is analyzed in terms of the block error rate which, for comparison and consistency purposes, is also presented analytically for traditional modulation schemes. The simulation results show that the studied architecture corresponds to optimal human engineering solutions, even without the knowledge of specific wireless domain. In addition, the deep architecture is able to converge quickly while maintaining its ability to generalize under different block lengths, signal-to-noise ratios, code rates and fading environments.

Keywords: Communication systems; α-μ fading model; deep learning; end-to-end learning; convolutional neural network.

Lista de Figuras

Lista de Tabelas

Lista de Abreviações e Siglas

MIMO	*Multiple Input Multiple Output* - sistema com múltiplas entradas e múltiplas saídas
ADALINE	*Adaptative Linear* - neurônio linear adaptativo
MADALINE	*Multiple Adaptative Linear* - neurônio linear adaptativo com múltiplas camadas
XOR	*Exclusive Or* - ou-exclusivo
RNN	*Recurrent Neural Network* - rede neural recorrente
RBMM	*Restricted Boltzman Machine* - máquinas de Boltzman restritas
CNN	*Convolutional Neural Networks* - redes neurais convolucionais
GPU	*Graphic Processing Unit* - unidade de processamento gráfico
ILSVRC	*ImageNet Large Scale Visual Recognition Challenge*
ReLU	*Rectified Linear Unit* - unidade linear retificada
ELU	*Exponential Linear Unit* - unidade linear exponencial
MLP	*Multilayer Perceptron* - Perceptron com múltiplas camadas
MSE	*Mean Square Error* - erro quadrático médio
BCE	*Binary Cross Entropy* - entropia cruzada binária
SGD	*Stochastic Gradient Descent* - gradiente descendente estocástico
ADAM	*Adaptative Moment Estimation* - estimação adaptativa de momentos
PCA	*Principal Component Analysis* - análise de componentes principais
SNR	*Signal-to-Noise Ratio* - relação sinal ruído
AWGN	*Additive White Gaussian Noise* - ruído aditivo Gaussiano branco
BLER	*Block Error Rate* - taxa de erro de bloco, ou taxa de erro na mensagem
PSK	*Phase-Shift Keying* - deslocamento por mudança de fase
BPSK	*Binary Phase-Shift Keying* - chaveamento binário de mudança de fase

QPSK *Quadrature Phase-Shift Keying* - chaveamento de mudança de fase em quadratura

QAM *Quadrature Amplitude Modulation* - modulação de amplitude em quadratura

Lista de Símbolos

α	Taxa de aprendizado (passo) do método iterativo de otimização
α_n	Amplitude do sinal recebido $r(t)$ em função do n-ésimo caminho
$\bar{\gamma_b}$	SNR média
$\bar{P_b}$	Probabilidade de erro de bit média
$\Gamma(\cdot)$	Função Gama
γ_b	SNR instantânea
$\hat{\mathbf{m}}_k$	Estimativa de primeiro momento dos gradientes
$\hat{\mathbf{v}}_k$	Estimativa de segundo momento dos gradientes
$\hat{r}$	α-ésima raiz do valor médio referente a envoltória α-μ
$\mathbb{E}[\cdot]$	Operador Esperança
$\mathbb{E}[R^2]$	Potência do sinal desvanecido (Rayleigh)
$\mathbb{E}[R_N^2]$	Potência do sinal desvanecido (Nakagami-m)
$\mathbf{p}$	Vetor probabilístico resultante da saída *Softmax*
$\mathbf{w}^{[n]}$	Todos os pesos da n-ésima camada
$\mathbf{y}$	Todas as saídas da rede neural
$\mathbf{Y}^f$	Mapas de características produzidos pelas convoluções
$\mathbf{z}^{[n]}$	Todos as ativações internas da n-ésima camada
$\mathbb{1}_s$	Vetor *one-hot*
$\nabla J(\mathbf{w})$	Vetor gradiente da função custo
$\nabla J^2(\mathbf{w})$	Vetor contendo o quadrado dos elementos do vetor gradiente
$\nabla^2 J(\mathbf{w})$	Matriz Hessiana da função custo
Ω	Potência do sinal desvanecido (Rayleigh)
Ω_N	Potência do sinal desvanecido (Nakagami-m)
Φ	Fase Rayleigh e α-μ

ϕ_{D_n}	Deslocamento de fase Doppler em função do n-ésimo caminho
Φ_N	Fase do sinal desvanecido Nakagami-m
ϕ_n	Fase do sinal resultante em função do n-ésimo caminho
σ	Desvio padrão das componentes em fase e em quadratura do sinal desvanecido
τ	Atraso de propagação
$\triangleq$	Operação de definição
$A^{[n]}$	Todas as saídas após ativação interna da n-ésima camada
$a_k^{[i]}$	Representação após ativação do k-ésimo neurônio na i-ésima camada
B	Tamanho do *mini-batch* de treinamento
b	Termo de polarização, ou *bias*
D	Número de amostras disponíveis para treinamento
E_b	Energia média por bit
F	Número de filtros convolucionais
$f_{\Phi_N}(\phi_N)$	Função densidade de probabilidade marginal da fase (Nakagami-m)
$f_{\Phi}(\phi)$	Função densidade de probabilidade marginal da fase (Rayleigh e α-μ)
$f_{R,\Phi}(r,\phi)$	Função densidade de probabilidade conjunta da envoltória e fase (Rayleigh e α-μ)
$f_{R_N,\Phi_N}(r,\phi)$	Função densidade de probabilidade conjunta da envoltória e fase (Nakagami-m)
$f_{R_N}(r_N)$	Função densidade de probabilidade marginal da envoltória (Nakagami-m)
$f_R(r)$	Função densidade de probabilidade marginal da envoltória (Rayleigh e α-μ)
$f_W(w)$	Função densidade de probabilidade marginal da potência (Rayleigh, Nakagami-m, e α-μ)
$f_{X,Y}(x,y)$	Função densidade de probabilidade conjunta das componentes em fase e quadratura (Rayleigh)
$f_{X_\alpha,Y_\alpha}(x,y)$	Função densidade de probabilidade conjunta das componentes em fase e quadratura (α-μ)
$f_{X_N,Y_N}(x,y)$	Função densidade de probabilidade conjunta das componentes em fase e quadratura (Nakagami-m)
$g'(\cdot)$	Derivada de primeira ordem de $g(\cdot)$

$g(\cdot)$	Função de ativação
J	Operador Jacobiano
$J(\mathbf{w})$	Função custo, ou função objetivo
k	Número de bits por símbolo transmitidos
L	Tamanho do bloco de mensagem
M	Número total de estímulos de entrada; Ordem da modulação
n	Número de estímulos de entrada (neurônio); Número de usos discretos do canal
$N(t)$	Número de componentes multipercurso
N_i	Número de neurônios da i-ésima camada
N_0	Densidade espectral de potência do ruído Gaussiano branco
P	Potência normalizada do sinal desvanecido (Rayleigh e α-μ)
$P_b(\gamma_b)$	Probabilidade de erro de bit no canal AWGN
P_{eM}	Probabilidade de erro na mensagem
$Q(\cdot)$	Função Q
R	Envoltória (Rayleigh e α-μ); Taxa de código
$r(t)$	Sinal recebido no receptor, devido ao efeito de multipercurso
R_N	Envoltória (Nakagami-m)
$S(\mathbf{z})$	Função *Softmax*
t	Tempo
$u(t)$	Sinal transmitido em banda base complexa
w_j	Peso associado ao j-ésimo estímulo de entrada
$w_{kj}^{[i]}$	j-ésimo peso associado ao k-ésimo neurônio da i-ésima camada
X, Y	Variáveis aleatórias Gaussianas de média zero e variância σ^2; Componentes em fase e em quadratura (Rayleigh)
X_α, Y_α	Componentes em fase e em quadratura (α-μ)
x_j	j-ésimo estímulo de entrada
X_N, Y_N	Componentes em fase e em quadratura (Nakagami-m)
z	Ativação interna do neurônio
$z_k^{[i]}$	Ativação interna do k-ésimo neurônio na i-ésima camada

Lista de Publicações

- S. B. F. Gomes e M. D. Yacoub, "CNN-Based Learning System in a Generalized Fading Environment," *Anais do XXXVIII Simpósio Brasileiro de Telecomunicações e Processamento de Sinais (SBrT'20)*, Florianópolis, Brasil, Set. 2020.

Sumário

Capítulo 1

Introdução

1.1 Contextualização do Tema

Sistemas sem fio convencionalmente construídos sob premissas e modelos analíticos e probabilísticos atingiram um alto grau de desempenho, atendendo a requisitos exigentes em inúmeras aplicações devido à capacidade de lidar com os efeitos do mundo real. Inspirados por C. Shannon [1], a camada física de um sistema de comunicação é geralmente representada como uma cadeia de blocos fundamentais de processamento. Cada bloco é modelado de forma que os resultados do sistema como um todo sejam matematicamente tratáveis.

Nos dias atuais, é de fundamental importância transmitir informações em meio sem fio de um ponto a outro de forma rápida, confiável e segura. Além do desenvolvimento de cadeias de processamento, comunicações sem fio é um campo de conhecimento especializado que envolve o projeto de formas de onda, tratamento de interferência e efeitos de tráfego, modelagem de canais, recuperação de símbolos e *bits* distorcidos, suporte à segurança sem fio, compensação de imperfeições de hardware de rádio, dentre outros.

Em comunicações sem fio, o desvanecimento de curto prazo ocorre sempre que os sinais alcançam um receptor por meio de vários caminhos. Como é amplamente conhecido, em tal caso, o comportamento do sinal pode ser modelado usando modelos estocásticos. Um grande número de distribuições descreve as estatísticas de canais de comunicação sem fio, incluindo principalmente Hoyt, Rayleigh, Weibull, Nakagami-m e Rice. Cada modelo tenta capturar os efeitos do mundo real em termos de diferentes suposições, como condição de onda de linha de visada e a linearidade do meio de propagação. A distribuição α-μ é um modelo de desvanecimento geral usado para representar a variação de pequena escala do sinal de desvanecimento em um ambiente não-linear no qual *clusters* de múltiplos caminhos estão presentes. Além disso, α-μ é um dos diversos modelos generalizados que inclui distribuições importantes como Nakagami-m, Weibull e Rayleigh.

Apesar de serem responsáveis por nos conduzir aos complexos e bem-sucedidos sistemas de comunicação adotados atualmente, as teorias de comunicação existentes exibem fortes limitações em aproveitar recursos do espectro limitado, além de dificuldade em lidar com a complexidade da otimização de sistemas sem fio emergentes. Para muitos sistemas, é difícil ter precisão ao capturar todos os efeitos em uma representação analítica de forma fechada. Dessa forma, eles são frequentemente representados por modelos simplificados

(lineares, estacionários, e Gaussianos) que podem não expressar totalmente as complexidades do mundo real. Por este motivo, é altamente desejável construir por meio de algum método de aprendizagem um esquema de comunicação de rádio diretamente de amostras da resposta de dispositivos (*hardwares*) e canais físicos reais, em vez de tentar modelar e/ou simplificar as respostas matematicamente.

De forma geral, a inteligência artificial é uma área da computação cujo objetivo é conceber máquinas com capacidade de resolução de problemas e tomadas de decisões. Suas principais vertentes de estudo se ramificam de acordo com a diferença de aquisição de conhecimento pela máquina. Dentre elas, *machine learning* vem sendo responsável por avanços recentes de interesse prático em diferentes áreas, por envolver a extração de conhecimento diretamente de amostras disponíveis a fim de realizar uma classificação, predição, ou dar suporte à tomada de decisão, sem ser explicitamente programado para tal. Fundamentados em teoria de probabilidade, estatística e otimização, os algoritmos de aprendizado de máquina englobam redes neurais com arquiteturas profundas, termo conhecido como *deep learning*, estando entre as suas mais bem-sucedidas vertentes, especialmente desde a última década.

Deep learning é um paradigma capaz de aprender as intrincadas inter-relações de variáveis, especialmente aquelas que são difíceis de descrever com precisão usando modelos matemáticos [2]. Isso nos permite projetar sistemas de comunicação sem fio sem o conhecimento pleno desses modelos, o que é impossível no contexto de princípios de projetos de sistemas sem fio atuais [3].

Embora muitas partes dos sistemas de comunicação possam ser resolvidas de forma otimizada, existem casos importantes em que o aprendizado profundo pode trazer grandes melhorias. Em particular, pode ser usado para reduzir a complexidade de algoritmos conhecidos ou para lidar com *hardware* ou canais não-lineares de forma eficiente.

Desse modo, há alguns papéis importantes que o aprendizado profundo pode desempenhar nas comunicações. Existem muitos problemas em que o algoritmo conhecido encontra o desempenho ótimo, mas tem uma complexidade proibitivamente alta para implementação em tempo real. Além disso, existem casos em que os modelos de sistema padrão são inadequados ou incompletos.

1.2 Trabalhos Anteriores e Contribuições

Em geral, há duas abordagens para aplicação de *deep learning* na camada física de sistemas de comunicação. A primeira envolve a substituição de um ou mais blocos da cadeia de processamento por algoritmos e técnicas de aprendizagem profunda, incluindo detecção [4, 5, 6], estimação de canal [7, 8], decodificação [9, 10], classificação de modulação para uma rede distribuída de detecção de espectro sem fio [11], codificação conjunta fonte-canal [12, 13], dentre outros. A outra abordagem interpreta todo o sistema de comunicação como uma técnica de aprendizagem profunda otimizada fim-a-fim.

A abordagem de interpretar um sistema de comunicação completo como um paradigma de aprendizagem profunda em termos de um *Autoencoder* foi originalmente dado proposta em [14]. No que diz respeito às comunicações, o objetivo de um *Autoencoder*

é encontrar representações das entradas (sinais transmitidos) em alguma camada intermediária que sejam robustas com respeito às restrições do mapeamento do canal (por exemplo, ruído, desvanecimento e distorção), permitindo a reconstrução na saída (sinal recebido) com pequena probabilidade de erro. O sistema de autoaprendizagem pode ser visto como pesos aprendidos de um sistema de redes neurais otimizadas fim-a-fim. Essa idéia foi posteriormente aprimorada em [15] com o intuito de combinar conhecimento especializado de domínio em comunicações sem fio em um modelo de *deep learning* por meio das redes transformadoras de rádio [16]. A viabilidade da implementação em *hardwares* desse esquema de autoaprendizagem fim-a-fim foi investigada em [17, 18]. Além disso, o conceito original foi estendido em [19] para sistemas com múltiplas entradas e saídas (MIMO, do inglês *Multiple Input Multiple Output*).

A adoção de camadas convolucionais na estrutura do *Autoencoder* foi primeiramente investigada em [20]. Porém, o desempenho analisado sofreu de um erro irredutível em regimes de alta relação sinal-ruído. Inspirado por essa deficiência, idéias integradas de engenharia de comunicações para propor uma estrutura inteligente com uma melhor capacidade de generalização foi investigada em [21]. No entanto, nenhum desses trabalhos aqui mencionados consideraram o meio de propagação em termos de sua não-linearidade.

O objetivo desta dissertação é ilustrar o emergente potencial de *deep learning* aplicado à comunicação sem fio, especialmente na camada física. Para isso, inspirados nos trabalhos mencionados, é feita uma releitura de um sistema de comunicação tradicional em que todo o seu processamento é substituído por apenas uma arquitetura de aprendizado profundo, sem nenhum conhecimento específico de engenharia de comunicações, exceto da modelagem do canal. Para a análise, consideramos um canal de comunicação com desvanecimento geral sujeito às estatísticas do modelo α-μ, as quais são apresentadas ao longo desta dissertação.

A análise é dada por meio da capacidade de generalização do *Autoencoder*, sob o cenário de desvanecimento geral, para arbritrário comprimento de bloco, SNR (do inglês *Signal-to-Noise Ratio*) de treinamento, e taxa de código. Para fins de comparação e validação, o desempenho analítico da taxa de erro na mensagem para esquemas tradicionais de modulação é considerado. É mostrado que o modelo dirigido por dados pode corresponder ao desempenho de soluções de engenharia humana ideais existentes. Pelo que consta ao autor, a configuração do sistema proposta ainda não foi investigada na literatura técnica e esta dissertação visa preencher esta lacuna.

1.3 Descrição dos Capítulos

Esta dissertação possui a seguinte estrutura:

- Capítulo 2: Este capítulo abrange os conceitos fundamentais de *deep learning* para o desenvolvimento da análise do modelo estudado. Duas arquiteturas de redes neurais artificiais são apresentadas, bem como os principais elementos necessários para o entendimento de suas respectivas otimizações. Além disso, o capítulo apresenta *Autoencoders*, importante técnica de aprendizado profundo adotado nesta dissertação.

- Capítulo 3: Neste capítulo, revisitamos os modelos físicos de canal, bem como suas respectivas relações com as distribuições de Rayleigh, Nakagami-m, e α-μ. Tais distribuições são responsáveis por descrever o comportamento do canal adotado no sistema inteligente abordado nesta dissertação.

- Capítulo 4: Este capítulo apresenta o modelo sistêmico dirigido por dados e como ele se relaciona com um sistema de comunicação tradicional. As comparações são feitas em termos do desempenho do sistema em termos da taxa de erro de bloco, onde sua derivação analítica é apresentada.

- Capítulo 5: Neste capítulo, conclusões finais e perspectivas de investigações futuras são apresentadas.

Capítulo 2

Deep Learning

2.1 Introdução

A história de Inteligência Artificial e, consequentemente, *deep learning*, é marcada por períodos de pessimismo e dificuldade de expansão, e períodos de entusiasmo e grande expansão literária. Por ser uma extensão de redes neurais, é natural que a literatura atribua o surgimento de aprendizado profundo em 1943, com o desenvolvimento do primeiro artifício de modelagem matemática de um neurônio artificial, o modelo de McCulloch & Pitts [22]. Esse modelo tinha a premissa de que um certo número de sinapses deve ser excitado em um determinado período para que o neurônio "dispare" em um processo binário, do tipo "tudo ou nada". A importância desse modelo se dá por ter possibilitado uma relação com a computação digital ao estabelecer uma conexão entre o funcionamento de um neurônio artificial e a lógica proposicional.

No fim da década de 50, mais precisamente em 1957, aproveitando as idéias propostas por [22], Frank Rosenblatt propôs o modelo de neurônio que ficaria conhecido e utilizado até os dias atuais, sendo o principal bloco de construção de redes neurais modernas. Esse modelo, chamado de Perceptron [23], se diferencia do [22] fundamentalmente por ter abandonado o domínio numérico exclusivamente booleano (binário). A idéia é que a ativação desse neurônio seja dada em termos da combinação linear entre os estímulos de entrada e os pesos sinápticos. Se essa ativação exceder certo limiar, ocorrerá o "disparo" do neurônio, expresso por meio de uma função de ativação.

Outros modelos de neurônios, criados por Widrow *et. al* ainda em 1959, foram denominados de neurônio linear adaptativo (ADALINE, do inglês *Adaptative Linear*) e neurônio linear adaptativo com múltiplas camadas (MADALINE, do inglês *Multiple Adaptative Linear* [24]. Desses dois modelos, o primeiro foi desenvolvido para reconhecer padrões binários a fim de prever o próximo bit na transmissão de uma linha telefônica, enquanto o último ficou conhecido por ser a primeira rede neural aplicada a problemas reais, usando um filtro adaptativo que elimina ecos nas linhas telefônicas, ainda em uso comercial atualmente.

Em 1969, Marvin Minsky e Seymour Papert [25] publicaram um livro que mostra diferentes limitações para o Perceptron. Dentre elas, a mais importante é relacionada com a computação de funções que não são linearmente separáveis, como a função ou-exclusivo (XOR, do inglês *Exclusive Or*). Papert & Minsky mostraram que um único

neurônio do tipo Perceptron não é capaz de desempenhar um mapeamento produzido pela porta lógica XOR. Além disso, os autores também acreditavam que não havia razão para supor que redes com múltiplas camadas de neurônios Perceptron pudessem conduzir a uma solução para o problema proposto. A partir dessa conjectura, somado com fatores relacionados com o exagero do potencial das redes neurais defendido no mundo acadêmico e técnico, e severas críticas de escritores sobre os possíveis efeitos negativos de "máquinas pensantes" na sociedade, o computador digital se tornou cada vez mais popular em relação ao neurocomputador, e a área de Inteligência Artificial passou pelo seu primeiro grande momento de descrença. Esse período foi marcado por redução drástica de grande parte do financiamento em pesquisas e pouco progresso acadêmico na área, com raras exceções.

O desânimo acadêmico só daria espaço para a volta do crescimento de debates construtivos na área de Inteligência Artificial apenas a partir de 1982, quando John Hopfield apresentou uma rede com memória, conhecida como rede de Hopfield [26], a qual posteriormente foi categorizada como uma forma de rede neural recorrente (RNN, do inglês *Recurrent Neural Network).* A abordagem de Hopfield não era simplesmente modelar cérebros, mas criar dispositivos úteis. Além disso, máquinas de Boltzman restritas (RBM, do inglês *Restricted Boltzman Machine*) também surgiram nessa época [27], caracterizadas por uma rede neural artificial generativa estocástica capaz de aprender uma distribuição de probabilidade a partir de uma série de entradas. No entanto, apenas em 1986 com o advento do algoritmo de retropropagação (*backpropagation*) [28], o qual permite o ajuste automático de pesos para redes neurais com múltiplas camadas, houve a quebra da conjectura do problema da função XOR, mostrando que com apenas três neurônios em cascata a rede ja era capaz de mapear tais funções [29]. Como um computador digital pode ser concebido apenas de portas lógicas XOR, então um neurocomputador que reproduz todas as portas lógicas XOR usando três neurônios é tão poderoso quanto um computador digital, pelo menos. Com a conjectura derrubada, a comunidade acadêmica conseguiu renovar o interesse pela área, bem como o retorno de financiamento em pesquisas.

A década seguinte foi marcada por desenvolvimento de diferentes arquiteturas de redes neurais artificiais. Em 1989, foi realizada a primeira demonstração prática do *backpropagation*, em que foi adotada uma rede neural convolucional (CNN, do inglês *Convolutional Neural Network*), rede inspirada ainda em 1979 com o Neocognitron [30], para reconhecer dígitos manuscritos de cheques. O aprendizado foi inteiramente automático, e essa abordagem se tornou a fundação da área de visão computacional moderna. Apesar do sucesso à época, os tempos de treinamento para os sistemas foram medidos em dias, tornando o aprendizado dessas redes neurais artificais impraticáveis para o uso no mundo real. Esse fator somado com abordagens de autores excessivamente otimistas, exagerando no potencial imeditado de redes neurais, foram responsáveis por uma quebra de expectativas de investidores, contribuindo para um segundo momento de descrença social e literária de Inteligência Artificial.

O próximo passo evolutivo significativo em *deep learning* foi dado no início dos anos 2000, com o desenvolvimento de GPUs (do inglês, *Graphics Processing Units*), onde os computadores começaram a se tornar mais rápidos no processamento de dados. Em 2006, o termo aprendizado profundo começou a ganhar mais popularidade, com a demonstração de como uma rede neural de várias camadas poderia ser pré-treinada uma camada por

vez [31]. Para muitos autores, essa é uma fase conhecida pelo "renascimento" de redes neurais.

Em 2009 foi lançada o *ImageNet*, uma base dados de mais de 14 milhões de imagens rotuladas. Foi também nesse ano a constatação de que, com um conjunto de dados suficientemente grande, as redes neurais não precisam de pré-treinamento e as taxas de erros caem significativamente. Desde 2010, o projeto *ImageNet* realiza um concurso anual de *software*, o ILSVRC (do inglês, *ImageNet Large Scale Visual Recognition Challenge*). Concursos dessa natureza se tornaram responsáveis por acelerar a busca de algoritmos competitivos para uma tarefa de interesse.

Em 2012, algoritmos de reconhecimento de padrões alcançaram desempenho em nível humano em determinadas tarefas. Desde então, importantes avanços têm sido registrados nos mais variados ramos da sociedade, de aplicações que variam de visão computacional, robótica, e assistentes digitais a mercado financeiro e medicina.

Este capítulo visa abordar de maneira simples a formação de diferentes composições de redes neurais, em especial redes plenamente conectadas e redes convolucionais, explorando os principais elementos que as compõem, bem como uma descrição detalhada dos respectivos processos de treinamento. Todo o processo de treinamento de redes neurais se consolida em um problema de otimização não-linear irrestrita, a fim de sintetizar um mapeamento regido por uma função objetivo. Além disso, *Autoencoders* são explorados como uma técnica auto-supervisionada de aprendizagem profunda a fim de produzir uma boa representação da entrada, em termos do entendimento do modelo, para sua reconstrução na saída.

2.2 Neurônio: Inspiração Biológica

O Perceptron [23] é o modelo matemático de neurônio mais adotado como unidade básica de processamento de uma rede neural artificial, ilustrado na Figura 2.1. Ao se considerar n estímulos de entrada, sua saída pode ser descrita por:

$$y = g(z) = g\left(\sum_{j=1}^{n} w_j x_j + b\right), \tag{2.1}$$

onde x_j e w_j representam o j-ésimo estímulo de entrada e o peso associado a ela, respectivamente. A função $g(\cdot)$ é a função de ativação, z a ativação interna do neurônio, e b é o termo de polarização ou *bias*, responsável pela translação da função de ativação.

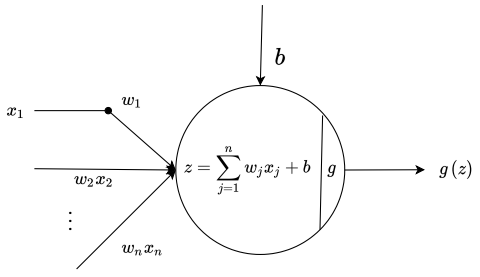

Figura 2.1: Modelo matemático de um neurônio artificial.

A topologia de uma rede neural artificial com Perceptrons é conexionista, sendo formada por camadas de neurônios empilhados espacialmente, ilustradas na Figura 2.2. A saída do k-ésimo neurônio de uma camada é dada por:

$$y_k = g(z_k) = g\left(\sum_{j=1}^{n} w_{kj}x_j + b_k\right). \tag{2.2}$$

Incluindo o *bias* no vetor de pesos com valor fixo $x_0 = 1$ e $w_{k0} = b_k$, a saída do neurônio é matricialmente formulada:

$$y_k = g(z_k) = g\left(\sum_{j=0}^{n} w_{kj}x_j + b_k\right) = g(\mathbf{w}^T\mathbf{x}). \tag{2.3}$$

O argumento da função de ativação e, portanto, a ativação interna do neurônio z_k é um produto interno. Isso tem o efeito de comparação, na forma de projeção, do vetor de estímulos de entrada $\mathbf{x}$ na direção do vetor de pesos sinápticos $\mathbf{w}$. Esses pesos possuem a capacidade de deslocar a função de ativação no eixo das abscissas, bem como alterar sua inclinação e amplitude.

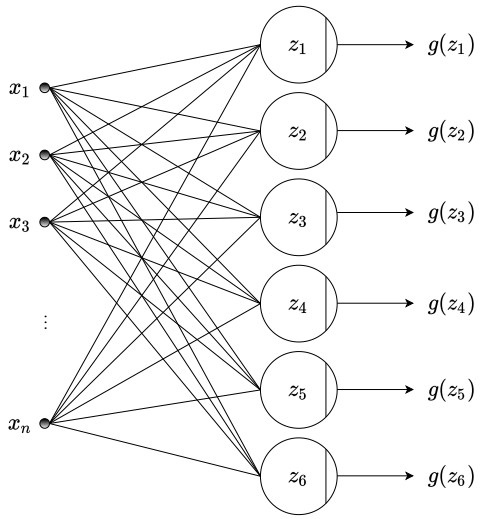

Figura 2.2: Camada de neurônios plenamente conectados.

2.3 Funções de ativação

Funções de ativação são peças fundamentais na composição de redes neurais artificiais, por introduzirem um mapeamento não-linear na ativação do neurônio. Além disso, por terem em seu argumento um produto interno, são chamadas de funções de expansão ortogonal. O mapeamento resultante reproduz o seu comportamento não-linear em uma direção, e se expande constantemente nas direções ortogonais. A Figura 2.3 exemplifica o mapeamento produzido por um neurônio cuja ativação é uma função logística.

Há uma grande variedade de funções de ativações utilizadas para diferentes objetivos. A seguir, destacamos as comumente adotadas e suas principais características.

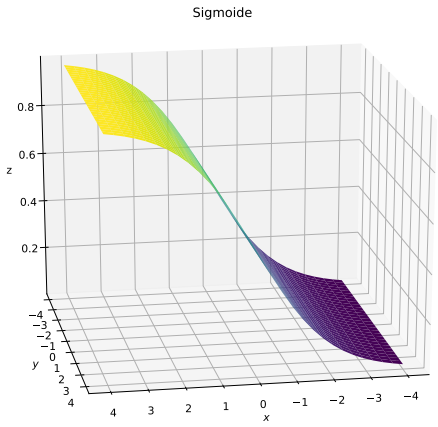

Figura 2.3: Sigmoide com expansão ortogonal.

2.3.1 Função Logística

Funções logísticas fazem parte das funções sigmoidais (com formato em S), e foram classicamente usadas em composições de redes neurais, por serem fáceis de trabalhar, e possuírem todas as propriedades de funções de ativação: ser não-linear, contínua, diferenciável, monotônica, e com alcance de saída fixo. As Equações (2.4) e (2.5) representam a função e sua derivada, respectivamente, ilustradas na Figura 2.4.

$$g(z) = \frac{1}{1 + e^{-z}} \tag{2.4}$$

$$g'(z) = g(z)(1 - g(z)) \tag{2.5}$$

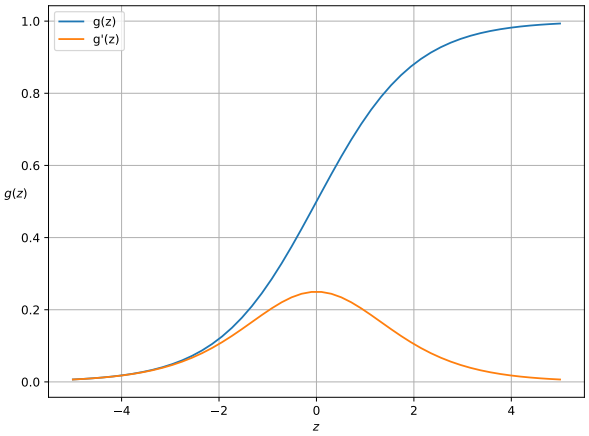

Figura 2.4: Função logística e sua derivada.

A função logística excursiona entre [0,1]. Apesar de ter seu uso bem reduzido com o passar dos anos, ainda é adotada como saídas de redes neurais cujo intuito é modelar uma variável binária, tarefa comumente encontrada em classificação binária.

2.3.2 Tangente Hiperbólica

Funções do tipo tangente hiperbólica também pertencem a sigmoidais. Diferente da função logística, é centrada em zero, possuindo seu excursionamento entre [-1,1]. As Equações (2.6) e (2.7) representam a função e sua derivada, respectivamente, ilustradas na Figura 2.5.

$$g(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \tag{2.6}$$

$$g'(z) = 1 - g(z)^2 \tag{2.7}$$

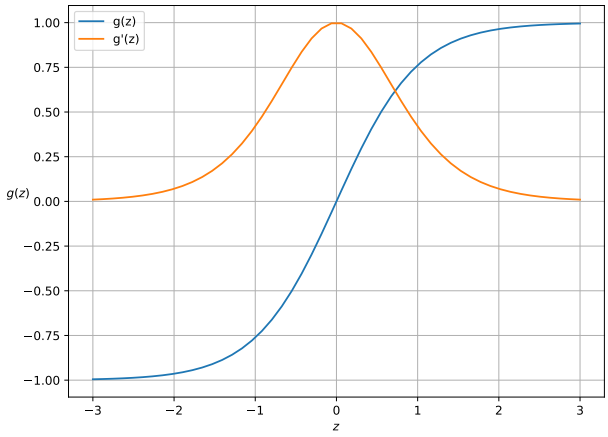

Figura 2.5: Função tangente hiperbólica e sua derivada.

Como veremos na Seção 2.5, o cômputo das derivadas (gradientes) das funções de ativação dos neurônios que compõem uma rede neural é fundamental para um bom desempenho da mesma. Embora possuam derivadas de fácil cômputo, funções sigmoidais são transcendentaiss, não possuindo cálculo exato em um computador digital. Além disso, magnitudes muito altas ou muito baixas dos gradientes dos pesos podem levar à essas funções a atuarem em regiões de saturação, dificultando seu treinamento, conduzindo ao problema conhecido como explosão e desvanecimento dos gradientes [32].

2.3.3 ReLU

Funções do tipo Unidade Linear Retificada (RELU, do inglês, *Rectified Linear Unit*), atuam como função identidade caso a ativação interna seja positiva, e nula caso contrário, operando como uma porta lógica [33]. Atualmente, ReLU é a função de ativação

mais adotada nas camadas intermerdiárias de uma rede neural artificial. Por não atuar em região de saturação, mitigando o efeito de desvanecimento de gradientes, e também por possuir fácil cômputo de sua derivada, funções dessa natureza são bem mais eficientes computacionalmente comparadas a funções sigmoidais. As Equações (2.8) e (2.9) representam a função e sua derivada, respectivamente, ilustradas na Figura 2.6.

$$g(z) = \max\{0, z\} \tag{2.8}$$

$$g'(z) = \begin{cases} 1, & z \geq 0 \\ 0, & \text{caso contrário} \end{cases} \tag{2.9}$$

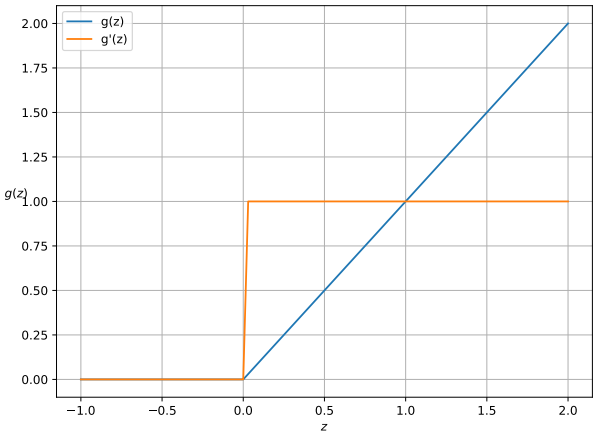

Figura 2.6: Função ReLU e sua derivada.

Uma desvantagem que precisa ser levada em conta ao adotar funções ReLU é que caso muitos neurônios ativem com valores negativos para muitos estímulos, ou seja, quando a soma ponderada antes da aplicação de uma ReLU se torna negativa, os mesmos deixam de participar do treinamento e de serem ajustados, possuindo derivada nula. Esse fenômeno faz com que a rede neural produza só zeros, e é conhecido na literatura como *dying gradients*. Além disso, excursiona entre [0,+∞], sendo suscetível a sofrer de explosões dos gradientes.

2.3.4 ELU

Proposta em 2015, funções do tipo Unidade Linear Exponencial (ELU, do inglês *Exponential Linear Unit*) também mitiga o problema de desvanecimento de gradientes, sendo uma função identidade para valores positivos [34]. As Equações (2.10) e (2.11) representam a função e sua derivada, respectivamente, ilustradas na Figura 2.7.

$$g(z, \alpha) = \begin{cases} z, & z \geq 0 \\ \alpha(e^z - 1), & \text{caso contrário} \end{cases} \tag{2.10}$$

$$g'(z,\alpha) = \begin{cases} 1, & z \geq 0 \\ g(z,\alpha) + \alpha, & \text{caso contrário} \end{cases} \tag{2.11}$$

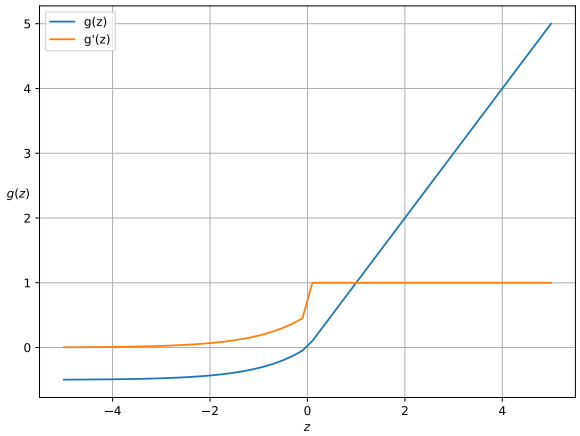

Figura 2.7: Função ELU com $\alpha = 0.5$ e sua derivada.

Diferentemente da ReLU, funções ELU permitem a entrada de um certo grau de valores negativos, dito por um parâmetro interno α. Isso mantém a média dos sinais de entrada próxima a zero com baixa complexidade computacional, aumentando a velocidade de treinamento. Funções ELU também possuem a desvantagem de excursionarem entre $[0,+\infty]$, sendo possível obter gradientes explodindo para um valor muito alto. Apesar de sua não-linearidade também saturar na parte negativa, estudos de redes com esse tipo de ativação têm mostrado desempenho superior a redes com ativação ReLU [34], bem como suas variantes com vazamento como a *Leaky* ReLU [35].

2.3.5 *Softmax*

A função *softmax* não possui nenhum apelo biológico em sua construção, sendo um artifício matemático largamente utilizado como função de ativação para tarefas de classificação. Dado um vetor $\mathbf{z}$ de K elementos reais, a função *softmax* transforma $\mathbf{z}$ de forma que seus elementos possuam soma unitária, dando uma noção de distribuição de probabilidade. Como resultado, produz um mapeamento $S\colon \mathbb{R}^K \mapsto [0,1]^K$ na forma

$$S(z)_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}}, \forall i \in \{1,2,\ldots,K\}. \tag{2.12}$$

Softmax aplica a função exponencial para cada elemento z_i do vetor de entrada $\mathbf{z}$ e normaliza esses valores pela soma de todas as exponenciais. Esta normalização certifica que a soma de todos elementos do vetor de saída $S(\mathbf{z})$ seja 1.

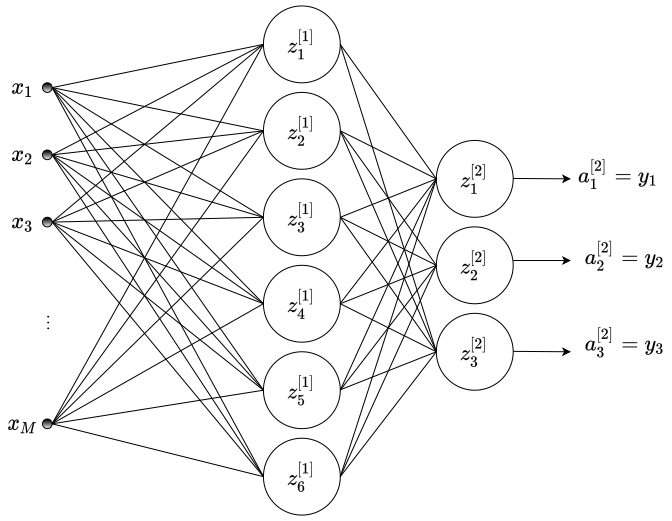

Figura 2.8: MLP com uma camada intermediária de 6 neurônios, e 3 neurônios na camada de saída.

2.4 Redes MLP

A utilização de neurônios Perceptron em uma única camada possui um potencial de atuação limitado pela capacidade de separação linear. Isso pode ser evitado empregando-se múltiplas camadas. A conexão entre neurônios e suas camadas, como ilustrado na Figura 2.2, forma uma rede neural. Uma rede de múltiplas camadas Perceptron (MLP, do inglês *Multilayer Perceptron*), também conhecida como rede *feedforward*, é uma estrutura com camadas plenamente conectadas, ou seja, a saída de cada neurônio da camada precedente é entrada para todos os outros neurônios da camada seguinte.

Considerando-se apenas uma camada intermediária, a k-ésima saída da rede é dada pela soma ponderada das funções de ativações da camada intermediária, possuindo mapeamento entrada-saída na forma:

$$\hat{y}_k = a_k^{[2]} = g(z_k^{[2]}) = g\left(\sum_{j=0}^{N_1} w_{kj}^{[2]}\ a_j^{[1]}\right) = g\left(\sum_{j=0}^{N_1} w_{kj}^{[2]}\ g\left(\sum_{i=0}^{M} w_{ji}^{[1]}\ x_i\right)\right). \tag{2.13}$$

O *bias* está sendo considerado na soma como $w_{k0} = b_k$ e $a_0^{[i]} = 1, i = \{1, 2\}$. A ativação interna e a sua representação após passar pela função de ativação $g(\cdot)$ do k-ésimo neurônio na i-ésima camada é dada por $z_k^{[i]}$ e $a_k^{[i]}$, respectivamente. M é o número de estímulos de entrada x, e N_i representa o número de neurônios na i-ésima camada. O j-ésimo peso associado ao k-ésimo neurônio da i-ésima camada é representado por $w_{kj}^{[i]}$. A segunda camada é a camada de saída, portanto $k = \{0, 1, 2, \ldots, N_2\}$.

Todos os N_1 neurônios recebem os mesmos (todos os) estímulos de entrada, mas cada neurônio faz uma ponderação possivelmente distinta dos mesmos. A camada de saída recebe como estímulo todas as saídas dos neurônios da camada intermediária, produzindo N_2 saídas. A Figura 2.8 exemplifica uma rede neural artificial com 6 neurônios na camada intermediária e 3 saídas, representando um mapeamento $\hat{f}\colon \mathbb{R}^M \mapsto \mathbb{R}^3$.

O mapeamento não-linear realizado por uma rede neural é, portanto, uma combinação linear de funções de expansão ortogonal. Variando-se os pesos da camada intermediária e os pesos da camada de saída, formam-se diferentes funções com diferentes escalas e orientações. Portanto, o principal objetivo de redes neurais artificiais é realizar mapeamentos multidimensionais a partir de amostras, em que cada mapeamento produzido depende dos pesos sinápticos da rede.

De fato, é sabido que redes neurais com uma única camada intermediária e um número finito de neurônios fazem parte do conjunto de modelos que possuem *capacidade de aproximação universal* [36]. Esse teorema existencial afirma que tais modelos são capazes de aproximar qualquer mapeamento contínuo com um grau arbitrário de erro, desde que restrito a uma região compacta (limitada e fechada) do espaço de entrada.

Uma formulação mais geral para uma rede MLP de múltiplas camadas é ilustrada na Figura 2.9, e dada como segue. A k-ésima saída da rede é dada por:

$$\hat{y}_k = a_k^{[L]} = g(z_k^{[L]}) = g\left(\sum_{j=0}^{N_L} w_{kj}^{[L]}\ a_j^{[L-1]}\right) = g\left(\sum_{j=0}^{N_L} w_{kj}^{[L]}\ g\left(\sum_{i=0}^{N_{L-1}} w_{ji}^{[L-1]}\ a_i^{[L-2]}\right)\right). \quad (2.14)$$

Isso é equivalente à seguinte forma matricial recursiva:

$$\begin{aligned} \hat{\mathbf{y}} &= A^{[L]} = g^{[L]}(\mathbf{z}^{[L]}) \\ \mathbf{z}^{[L]} &= \mathbf{w}^{[L]} A^{[L-1]} \end{aligned} \quad (2.15)$$

Os vetores $\mathbf{w}^{[n]}$, $\mathbf{z}^{[n]} = \{z_1^{[n]}, z_2^{[n]}, \ldots, z_{N_n}^{[n]}\}$, e $A^{[n]} = \{a_1^{[n]}, a_2^{[n]}, \ldots, a_{N_n}^{[n]}\}$ representam todos os pesos sinápticos (e *bias*), ativações internas, e saídas após função de ativação $g^{[n]}(\cdot)$ dos neurônios da n-ésima camada da rede, respectivamente. N_n representa o número de neurônios na n-ésima camada. O vetor $\hat{\mathbf{y}} = \{y_1, y_2, \ldots, y_{N_L}\}$ contém todas as saídas, em que L representa a última camada (de saída) da rede. Os índices seguem da mesma forma da Equação (2.13). Nessa formulação, $A^{[0]}$ representa os padrões de entrada da rede.

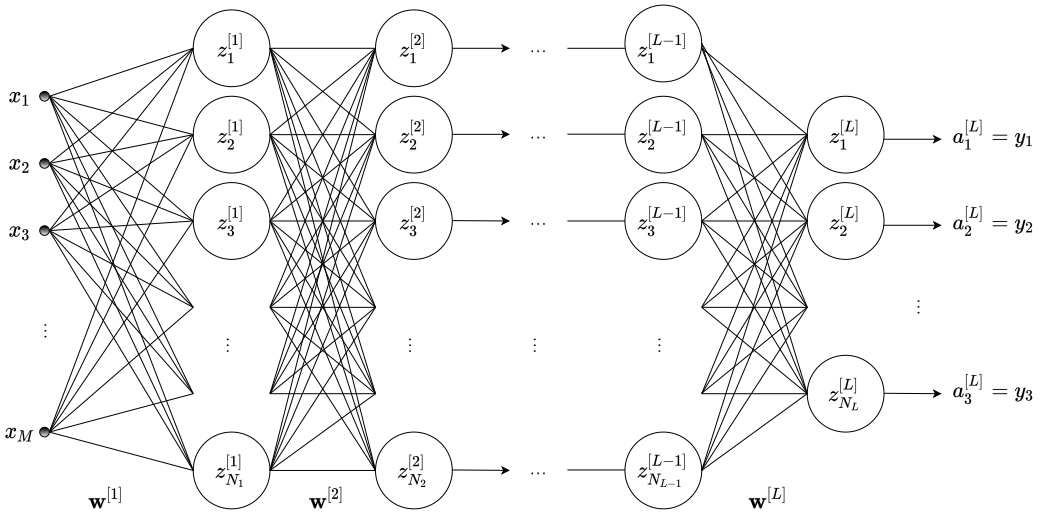

Figura 2.9: Modelo de rede MLP com L camadas.

Em *machine learning*, aprendizagem supervisionada corresponde à tarefa de aprender (sintetizar) funções que mapeiam entrada-saída, a partir de pares de amostras entrada-saída. Com redes neurais artificiais, o processo de treinamento supervisionado é equivalente a um problema de otimização não-linear irrestrita, em que a superfície de erro é minimizada a partir do ajuste dos pesos sinápticos [37].

Além de minimizar o erro de treinamento junto à superfície de erro, é fundamental controlar adequadamente o grau de flexibilidade de modelos com capacidade de aproximação universal. Modelos com boa capacidade de generalização conseguem bom desempenho frente a amostras nunca vistas antes, enquanto que modelos com baixa capacidade de generalização tendem a "decorar" os dados de treinamento, possivelmente produzindo estimações erradas para novas amostras.

Embora não abordadas neste trabalho, há diversas formas exploradas na literatura para o ajuste de flexibilização de modelos, conhecido como técnicas de regularização[1]. As estratégias de regularização são projetadas para reduzir o erro de teste de um algoritmo de aprendizado de máquina, possivelmente às custas do erro de treinamento. Existem muitas formas diferentes de regularização em *deep learning*. Dentre elas, as que mais se destacam são as que controlam a norma do vetor de pesos sinápticos, penalizando-o com o intuito do mesmo não crescer muito durante o treinamento [37].

2.5 Aprendizado de Redes Neurais

O problema de otimização em redes neurais artificais, sem perda de generalidade, corresponde a uma tarefa de minimização de uma função objetivo $J(\mathbf{w})$, com respeito a um vetor de parâmetros ajustáveis $\mathbf{w} = \{\mathbf{w}^{[1]}, \mathbf{w}^{[2]}, \ldots, \mathbf{w}^{[L]}\}$. Essa função, também conhecida como função custo, representa o critério de desempenho a ser otimizado. A idéia de aprendizado, nesse contexto, é a busca de um $\mathbf{w}^*$, sendo visto como um problema de otimização:

$$\mathbf{w}^* = \min_{\mathbf{w}} J(\mathbf{w}) \tag{2.16}$$

Em aprendizado supervisionado, geralmente há três mapeamentos envolvidos no processo de treinamento:

- O mapeamento a ser aproximado (sintetizado) $f\colon X \subset \mathbb{R}^M \mapsto \mathbb{R}^{N_L}$, do qual se conhece D amostras do conjunto $\{(\mathbf{x}^{(i)}, \mathbf{y}^{(i)}) \in \mathbb{R}^M \times \mathbb{R}^{N_L}\}_{i=1}^{D}$. Considera-se que X é uma região compacta, portanto fechada e limitada, do espaço de entradas $\mathbb{R}^M$. Além disso, é admitido que os vetores de entrada $\mathbf{x}^{(i)}$ estão distribuídos conforme uma função densidade de probabilidade $d_p\colon \mathbb{R}^M \mapsto [0, 1]$. Os vetores de saída são produzidos na forma $y_i = f(\mathbf{x}^{(i)}) + \epsilon_i$, em que $\epsilon_i \in \mathbb{R}$ é uma variável aleatória de média zero e variância fixa.

- O mapeamento aproximado $\hat{f}\colon \mathbb{R}^M \mapsto \mathbb{R}^{N_L}$, fornecido pela rede neural, na forma $\hat{\mathbf{y}}^{(i)} = \hat{f}(\mathbf{x}^{(i)}, \mathbf{w})$.

[1] Para uma melhor descrição dos diferentes métodos de regularização, referenciamos [38].

- O mapeamento entre $\mathbf{w}$ e $J(\mathbf{w})$, denominado superfície de erro.

A função objetivo $J(\mathbf{w})$ mede o custo incorrido de predições incorretas, e depende da modelagem do problema. Para problemas de regressão, cuja saída desejada é um número real, é comum adotar o erro quadrático médio (MSE, do inglês *Mean Square Error)*, na forma:

$$\begin{aligned} J(\mathbf{w}) &= \frac{1}{D}\sum_{i=1}^{D}\left(\mathbf{y}^{(i)} - \hat{f}(\mathbf{x}^{(i)}, \mathbf{w})\right)^2 \\ &= \frac{1}{D}\sum_{i=1}^{D}\left(\mathbf{y}^{(i)} - \hat{\mathbf{y}}^{(i)}\right)^2 \end{aligned} \tag{2.17}$$

em que $\mathbf{y}^{(i)}$ representa o valor verdadeiro (rótulo) da i-ésima amostra, e $\hat{\mathbf{y}}^{(i)}$ denota o valor aproximado pela rede para mesma amostra, com $i = 1, 2, \ldots, D$, sendo D o número total de amostras disponíveis para treinamento.

Para problemas de classificação, é comum adotar a entropia cruzada binária (BCE, do inglês *Binary Cross Entropy*), sendo frequentemente associada à diferença entre duas distribuições de probabilidade (verdadeira e estimada) sobre o mesmo conjunto de eventos [2]. É largamente adotada em modelos cuja saída é uma distribuição de probabilidade entre 0 e 1. Sua representação binária é dada como segue:

$$\begin{aligned} J(\mathbf{w}) &= -\frac{1}{D}\sum_{i=1}^{D}\mathbf{y}^{(i)}\log(\hat{f}(\mathbf{x}^{(i)}, \mathbf{w}) + (1 - \mathbf{y}^{(i)})\log(1 - \hat{f}(\mathbf{x}^{(i)}, \mathbf{w})) \\ &= -\frac{1}{D}\sum_{i=1}^{D}\mathbf{y}^{(i)}\log(\hat{\mathbf{y}}^{(i)}) + (1 - \mathbf{y}^{(i)})\log(1 - \hat{\mathbf{y}}^{(i)}) \end{aligned} \tag{2.18}$$

O processo de otimização não-linear envolvido no ajuste de pesos de redes neurais realiza aproximações locais (de primeira ou segunda ordem) junto à superfície de erro, com ajustes iterativos, incrementais e recursivos. Métodos de primeira ordem são baseados nas derivadas de primeira ordem da função custo em relação a todos os pesos que compõem a rede neural, agrupados em um vetor gradiente $\nabla J(\mathbf{w})$, enquanto que métodos de segunda ordem baseiam-se na informação trazida pela segunda derivada da função custo, contida na matriz hessiana $\nabla^2 J(\mathbf{w})$.

Métodos de segunda ordem, embora tenham tendências de convergência mais rápida comparado a métodos de primeira ordem, possuem a desvantagem de serem mais custosos por iteração de busca, devido ao cálculo da matriz hessiana. Existem iniciativas de métodos de segunda ordem que buscam evitar o emprego direto da matriz hessiana [39, 40], mas são técnicas ainda pouco exploradas.

Por ter seu processo de ajuste baseado apenas em informações locais, métodos de otimização de redes neurais geralmente convergem para o mínimo local mais próximo de onde se inicia a busca, ditados por suas bacias de atração. Mínimos locais representam uma solução ótima em relação a seus vizinhos, enquanto que mínimo global representa a solução ótima em todo o domínio considerado. A Figura 2.10 ilustra esses conceitos, bem como bacias de atração dos mínimos, considerando $\mathbf{w}$ com uma única dimensão.

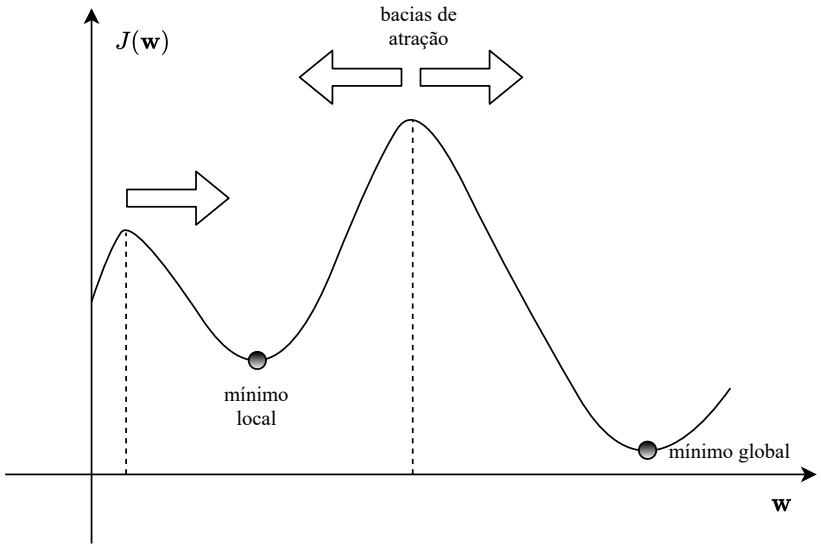

Figura 2.10: Exemplo de mínimo local, global e bacias de atração.

2.5.1 Gradiente Descendente

Intuitivamente, esse método usa o gradiente da função objetivo $\nabla J(\mathbf{w_0})$, calculado em um ponto $\mathbf{w_0}$ da superfície, para indicar o crescimento da função, e toma um passo na direção contrária ao gradiente, rumo ao mínimo da função objetivo. A atualização (ajuste) dos pesos da k-ésima iteração é dada por:

$$\mathbf{w_{k+1}} = \mathbf{w_k} - \frac{\alpha}{B} \sum_{i=1}^{B} \nabla J_i(\mathbf{w_k}) \tag{2.19}$$

O cálculo do vetor gradiente $\nabla J(\mathbf{w})$ é feito fundamentalmente adotando a regra da cadeia para retropropagação do erro [28], de modo que primeiro é calculada a derivada da função custo em relação aos pesos da última camada, para posteriormente ser retropropagado para o cálculo das derivadas das camadas anteriores. Esse algoritmo é conhecido na literatura como *backpropagation*.

O tamanho do passo é definido pela taxa de aprendizado α. Esse algoritmo de otimização iterativo é conhecido como *mini-batch gradient descent*, em que divide os dados disponíveis para treinamento em frações (lotes) de tamanho $B \subset \{1, 2, \ldots, D\}$.

Se $B = 1$, o erro $J(\mathbf{w})$ é calculado para cada amostra de treinamento, bem como a atualização do modelo. Na literatura essa versão é conhecida como gradiente descendente estocástico (SGD, do inglês *Stochastic Gradient Descent*). Se a atualização do modelo for dada a cada passagem completa pelos dados de treinamento, ou $B = D$, o algoritmo é conhecido como *batch gradient descent*.

Uma taxa de aprendizado muito pequena pode levar a uma convergência muito lenta, podendo nem convergir, enquanto que um passo de ajuste iterativo muito alto pode levar à instabilidade, seja por apresentar oscilações em torno de um mínimo local, seja por divergir da solução ótima [2]. Desse modo, a adoção de uma taxa fixa geralmente não é uma boa opção, tendendo a ser mais produtivo a promoção de ajustes mais intensos para pesos

com ajustes menos frequentes. Além disso, em *deep learning*, é comum mínimos locais de baixa qualidade e existência de pontos de selas e *plateaus* [41], locais que, em algumas direções são atratores, em outras não, prejudicando o desempenho do treinamento.

Embora métodos de segunda ordem tendam a levar bem menos iterações para convergir, são mais custosos computacionalmente, com difícil cômputo da hessiana. Desse modo, algoritmos adaptativos vêm sendo propostos como uma alternativa, promovendo, em média, progressos expressivos na exploração da superfície de erro a cada iteração [37]. Tais algoritmos foram propostos a fim de se obter uma mudança adaptativa da taxa de aprendizado α conforme a superfície de erro é explorada.

2.5.2 ADAM

Sendo um dos algoritmos adaptativos mais adotados atualmente, estimação adaptativa de momentos (ADAM, do inglês *Adaptative Moment estimation*) recorre a um passo de aprendizado específico para cada peso da rede neural, diferente das variantes do gradiente estocástico, cuja taxa de aprendizado é fixa para todos os elementos. Esse algoritmo recorre a estimativas de primeiro momento (média) e segundo momento (variância) dos gradientes, tendo sua atualização dos pesos na seguinte forma [42]:

$$\mathbf{w_{k+1}} = \mathbf{w_k} - \frac{\alpha}{\sqrt{\hat{\mathbf{v}}_\mathbf{k}} + \epsilon}\hat{\mathbf{m}}_\mathbf{k}, \tag{2.20}$$

com:

$$\begin{aligned} \hat{\mathbf{m}}_\mathbf{k} &= \frac{\gamma_1 \mathbf{m_{k-1}} + (1-\gamma_1)\nabla J(\mathbf{w_k})}{1-\gamma_1^k} \\ \hat{\mathbf{v}}_\mathbf{k} &= \frac{\gamma_2 \mathbf{v_{k-1}} + (1-\gamma_2)\nabla J^2(\mathbf{w_k})}{1-\gamma_2^k} \end{aligned} \tag{2.21}$$

sendo as estimativas de primeiro e segundo momento com correção de viés, respectivamente. Adota-se $\mathbf{m_0}$ e $\mathbf{v_0}$ como vetores nulos, $\nabla J^2(\mathbf{w_k})$ é o vetor contendo o quadrado dos elementos do vetor gradiente, e γ_1, γ_2, e ϵ são hiperparâmetros (parâmetros não ajustados pelo treinamento, e sim pelo projetista) com valores recomendados: $\gamma_1 = 0.9$, $\gamma_2 = 0.999$ e $\epsilon = 10^{-8}$.

2.6 Inicialização dos pesos

Como o treinamento de redes neurais artificiais é dado em termos de métodos iterativos, o mesmo depende de uma inicialização adequada. Além disso, pode haver variações expressivas em relação à velocidade de convergência.

Embora existam heurísticas e técnicas mais elaboradas de inicialização [43], é comum inicializar os pesos da rede neural com valores pequenos e aleatoriamente distribuídos, centradas em zero. São tipicamente adotadas distribuições Gaussianas ou uniformes.

Adotando esse tipo de inicialização, o mapeamento produzido pela rede neural tende a se aproximar de um hiperplano, sem nenhuma tendência definida, em termos de comportamento não-linear. Além disso, a ativação dos neurônios se encontrará fora da região

de saturação (em caso de ativações sigmoidais), facilitando o processo de ajuste dos pesos sinápticos.

Intuitivamente, adotando-se esse tipo de inicialização, o mapeamento produzido pela rede neural se inicia sem nenhuma contorção expressiva, mas com máxima capacidade de se contorcer de acordo com a demanda da aplicação [37].

2.7 Redes Convolucionais

Esse tipo de arquitetura de redes neurais abandona o paradigma completamente conexionista que emprega a operação de multiplicação matricial, típico de redes MLP. Em redes neurais convolucionais (CNN), as conexões são locais e emprega-se o uso da convolução[2] da entrada com um conjunto de filtros (também chamados de *kernels*) como operação principal em pelo menos uma de suas camadas [2].

A convolução é uma operação linear que explora informações em estruturas organizadas, sejam elas 1D, 2D, ou superior. Dentre as vantagens de redes convolucionais quando comparado a redes MLP, se destacam a invariância a translações e rotações da entrada, e a economia de pesos sinápticos. Além disso, em redes convolucionais há compartilhamento de parâmetros, de forma que o mesmo conjunto de pesos aprendidos é usado em diferentes partes da entrada, aumentando significativamente sua eficiência de armazenamento de informação.

Redes convolucionais foram propostas como um método de treinamento eficiente para estrutras de dados 2D (imagens) [44] e, assim como redes MLP, possuem inspirações biológicas em sua construção. A adoção dessa arquitetura levou a importantes progressos em visão computacional [45, 46, 47], sendo uma das principais responsáveis pelo recente sucesso de *deep learning* em diferentes ramos da sociedade.

A idéia de redes CNN é adotar uma arquitetura profunda de forma que cada camada aprenda um nível de representação do problema que permita sua solução. Por exemplo, em classificação de imagens, a primeira camada é responsável por detectar características simples (como traços e bordas), enquanto que os neurônios das últimas camadas já disparam para padrões mais complexos (como faces e objetos).

Geralmente, redes CNN são compostas por três diferentes tipos de camadas: camadas convolucionais, camadas de *pooling*, e camadas plenamente conectadas (*fully-connected*). Uma camada convolucional consiste em um conjunto de F filtros $\mathbf{W}^f \in \mathbb{R}^{a\times b}$, com $f = 1, 2, \ldots, F$, que produz de uma matriz de entrada $\mathbf{X} \in \mathbb{R}^{n\times m}$, mapas de características $\mathbf{Y}^f \in \mathbb{R}^{n'\times m'}$, de acordo com a convolução:

$$Y^f_{i,j} = \sum_{k=0}^{a-1}\sum_{\ell=0}^{b-1} W^f_{a-k,b-\ell} X_{1+s(i-1)-k,1+s(j-1)-\ell} \tag{2.22}$$

Os filtros (*kernels*) $\mathbf{W}^f$ percorrem a entrada $\mathbf{X}^{n\times m}$ realizando produtos internos entre localidades da matriz de entrada e os mesmos, por isso são chamados de filtros locais. O percurso é dado de cima para baixo, da esquerda para direita, como ilustrado na

[2]Matematicamente, camadas convolucionais realizam operações de correlação cruzada, sem o *flip* do filtro, mas a literatura convencionou por chamar de convolução.

Figura 2.11. O passo de deslocamento do percurso é dito por $s \geq 1$, chamado de *stride*. Os mapas de características resultantes possuem dimensões $n' = 1 + \lfloor \frac{n+a-2}{s} \rfloor$ e $m' = 1 + \lfloor \frac{m+b-2}{s} \rfloor$. Após a convolução, é introduzido não-linearidade na rede por meio de funções de ativação $g(\cdot)$.

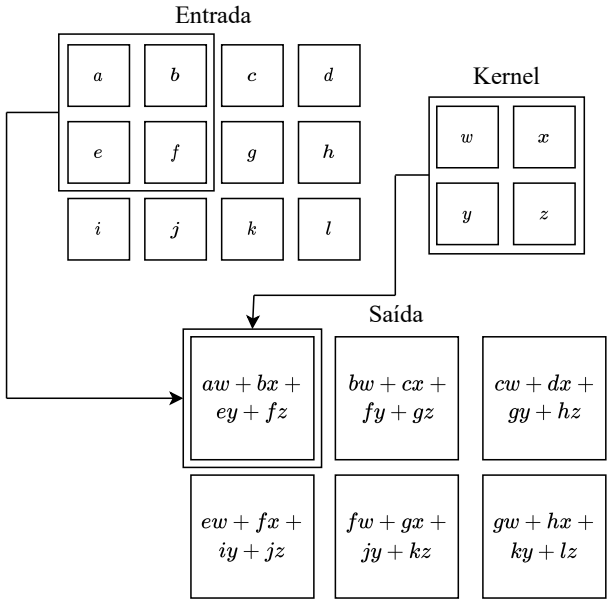

Figura 2.11: Exemplo de uma convolução 2D. [2]

As dimensões da saída podem ser reduzidas aumentando-se o *stride* ou adicionando uma camada de *pooling* em sequência. Essas camadas são responsáveis por particionar o mapa de características $\mathbf{Y}$ em regiões $p \times p$ menores, tomando um valor singular em toda a região, geralmente sendo o valor máximo (*max pooling*) ou valor médio (*average pooling*) da janela observada. Embora haja pequena perda de informação, essas camadas conduzem a ganhos significativos na construção da arquitetura [48].

Camadas plenamente conectadas são adotadas em redes convolucionais geralmente após uma série de camadas convolucionais (realizando convoluções seguidas de uma não-linearidade) com camadas *pooling*, constituindo, portanto, o fim da rede. Intuitivamente, as camadas convolucionais e *pooling* realizam a extração de características da entrada de maneira hierárquica, sendo as mais complexas reconhecidas nas camadas mais profundas. Em seguida, são adotadas algumas camadas plenamente conectadas com o intuito de modular esses mapas de características para diferentes objetivos, dependendo da demanda da aplicação. A Figura 2.12 ilustra um exemplo de CNN para tarefa de classificação de 10 classes. O bloco convolucional (contendo camadas convolucionais e *pooling*) é responsável por extrair características relevantes à classificação, enquanto que o bloco de camadas plenamente conectadas sintetizam o mapeamento não-linear produzido no fim da cascata de filtros para um vetor de probabilidades (por intermédio da ativação *softmax*), indicando a probabilidade de a imagem de entrada pertencer a cada uma das classes do problema.

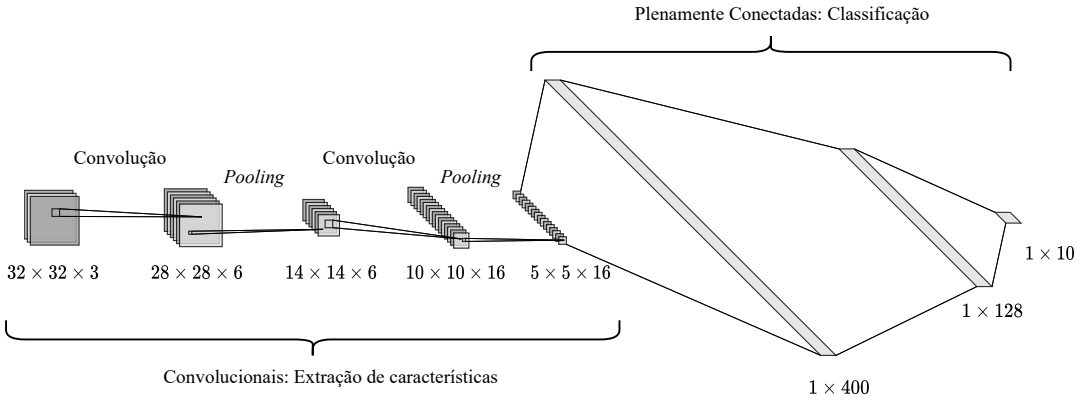

Figura 2.12: Exemplo de uma CNN para classificação de imagens.

A aprendizagem em redes CNN é realizada da mesma maneira que em redes MLP, em que se busca minimizar os parâmetros ajustáveis junto a uma superficie de erro, expresso na Equação (2.16). Todas as camadas realizam operações não-lineares e, portanto, o ajuste dos pesos deve se dar por técnicas de otimização não-linear, com *backpropagation*. A diferença é que o ajuste é feito tanto nos pesos que constituem os *kernels* das camadas convolucionais, quanto nos pesos das camadas densas (plenamente conectadas).

Para estruturas unidimensionais (1D), como séries temporais, redes convolucionais também se mostram eficientes para processamento de informações presente nos dados. Além de serem vastamente utilizadas como redutor de dimensionalidade, convoluções unidimensionais constituem elementos que formam blocos de construção fundamentais em estruturas de redes neurais mais complexas, como o bloco *Inception* da *GoogleNet* [47].

O benefício de se usar redes convolucionais para classificação de sequências é que elas podem aprender diretamente com os dados brutos da série temporal e, por sua vez, não requerem experiência de domínio para projetar manualmente as características de entrada. O modelo pode aprender uma representação interna dos dados da série temporal e, de maneira ideal, obter desempenho comparável aos modelos que se encaixam em uma versão do conjunto de dados com recursos de engenharia.

A saída de um sinal discreto $x(n)$ convoluído com um *kernel* $w(n)$ é dado por [2]:

$$y(n) = (x * w)(n) = \sum_{k=-\infty}^{\infty} x(k)w(n-k) \tag{2.23}$$

2.8 *Autoencoders*

Autoencoders são técnicas de aprendizagem profunda constituídas por redes neurais treinadas cujo objetivo é reconstruir sua entrada na saída. O modelo é projetado de forma a minimizar o erro de reconstrução entre a entrada e a saída, forçando-o a priorizar quais aspectos da entrada devem ser copiados.

Geralmente, *Autoencoders* podem ser interpretados como duas redes neurais em cascata. Dada uma entrada $\mathbf{x}$, a primeira rede neural realiza a função codificadora $\mathbf{h} = f(\mathbf{x})$, enquanto que a segunda rede é responsável pela decodificação, produzindo a reconstrução

$\mathbf{r} = g(\mathbf{h})$. A Figura 2.13 ilustra o mapeamento produzido pelas redes neurais em sequência, enquanto que a Figura 2.14 exemplifica uma possível composição do modelo, bem como seus componentes principais.

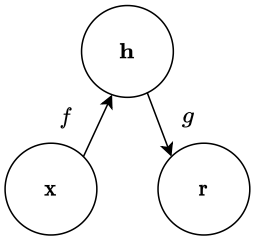

Figura 2.13: Sequência de mapeamentos produzidos pelo *Autoencoder*. [2]

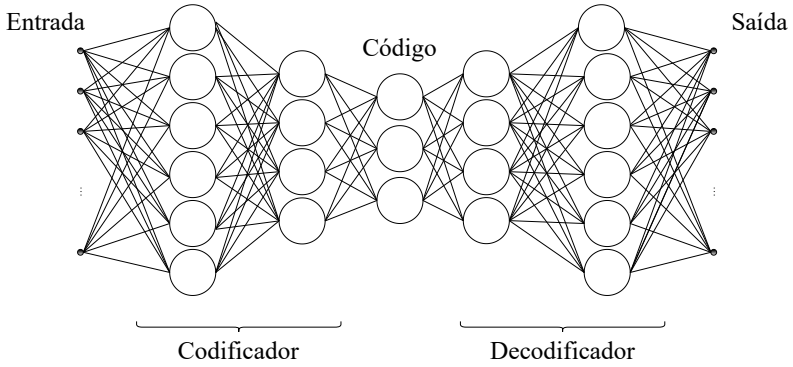

Figura 2.14: Exemplo de um *Autoencoder* composto por duas redes MLP.

O principal objetivo de *Autoencoders* é codificar eficientemente os dados de entrada em um espaço de representação, conhecido como espaço de variáveis latentes. O código obtido possui uma dimensão reduzida comparado com a entrada, de tal forma que o uso desse código permita a reconstrução da entrada com erro arbitrariamente pequeno. Essa abordagem tem sido empregada em diversas aplicações, como em modelos geracionais e eliminação de ruído [37].

O codificador e decodificador são treinados simultaneamente para reduzir o erro de reconstrução na saída da rede neural. Para isso, uma única função custo é adotada englobando as duas redes, tendo sua otimização sendo realizada através de métodos iterativos com *backpropagation* para cômputo dos gradientes. O processo de treinamento é dado na forma de minimização da função custo [2]:

$$L\left(\mathbf{x}, g(f(\mathbf{x}))\right), \tag{2.24}$$

em que a função custo L é uma penalização da transformação $g(f(\mathbf{x}))$ ser diferente de $\mathbf{x}$, como por exemplo o erro quadrático médio (MSE) dado por (2.17), e a entropia cruzada (CE) dada por (2.18).

Autoencoders com funções codificadoras e decodificadoras não-lineares podem aprender um método geral de análise de componentes principais [49] (PCA, do inglês *Principal Component Analysis*) mais competente. Adotando-se um *Autoencoder* com decodificador

linear e função custo L sendo MSE, o mesmo é capaz de projetar o mesmo subespaço produzido por PCA [2].

2.9 Conclusões

Neste capítulo, a formulação de redes neurais como um problema de otimização não-linear irrestrito foi apresentada. O entendimento de redes neurais como potenciais mapeadores universais é de fundamental importância para a asbtração em problemas reais de engenharia. Além da importância de inicialização dos pesos, as principais funções de ativação adotadas atualmente na literatura também foram exploradas, bem como um algoritmo adaptativo de mudança de passo de aprendizado, responsável por facilitar o processo de treinamento. Por fim, discutimos como redes convolucionais particionam o problema de representação em subtarefas, e como *Autoencoders* interpretam o sistema a fim de reconstruir sua entrada na saída.

Capítulo 3

O Canal em Sistemas sem Fio

3.1 Introdução

Canais de comunicações sem fio operam por meio da radiação de ondas eletromagnéticas do transmissor para o receptor. Na prática, a intensidade do sinal rádio-móvel não está apenas suscetível a ruído e inteferência, mas também sofre frequentes flutuações, ora proporcionando um sinal de boa qualidade, ora degradando o mesmo, resultando no fenômeno conhecido como desvanecimento.

Os principais fatores que afetam consideravelmente a intensidade do sinal rádio-móvel envolvem sua reflexão, difração, e espalhamento. A reflexão ocorre quando a onda incide sobre uma superfície de dimensões muito maiores que o seu respectivo comprimento de onda (obstruções topológicas e morfológicas de larga escala). A difração ocorre quando o sinal atravessa a fronteira de um objeto com dimensões de mesma ordem de grandeza do seu comprimento de onda. O espalhamento decorre de uma reflexão do sinal em múltiplas réplicas quando obstruído por corpos de dimensões muito menores em relação ao seu respectivo comprimento de onda.

A caracterização da intensidade do sinal rádio-móvel é baseada em três fenômenos: perda de percurso, desvanecimento de longo prazo (ou lento), e desvanecimento de curto prazo (ou rápido). Os dois primeiros se referem a efeitos de propagação de larga escala, em que a variação do sinal recebido ocorre em distâncias relativamente longas ao respectivo comprimento de onda. O desvanecimento ocorre devido a rápidas variações na intensidade do sinal causadas pela adição construtiva (ou destrutiva) de componentes multipercurso, caracterizando efeitos de propagação de pequena escala.

Perda de percurso se refere à atenuação do sinal ao longo do percurso de transmissão. Além da distância entre transmissor e receptor, a perda de percurso leva em consideração diversos fatores como altura da antena de transmissão, frequência de operação, e ambiente de propagação. Em sua versão simplificada, perda de percurso é uma função determinística da distância.

O desvanecimento de longo prazo provoca uma variação na média global do sinal recebido, causado pelo efeito do sombreamento provocado por obstruções de larga escala. Essa degradação aleatória é geralmente caracterizada por modelos estocásticos admitindo um processo aleatório log-normal. A aleatoriedade presente no ambiente é capturada modelando-se a densidade dos obstáculos e sua capacidade de absorção.

O desvanecimento de curto prazo afeta a média local do sinal e ocorre em intervalos de aproximadamente frações de comprimento de onda, considerando variações instantâneas dos parâmetros relacionados ao sinal recebido. Dentre os principais fatores que influenciam o desvanecimento rápido, destaca-se a propagação de multipercurso por meio da presença de objetos espalhadores e refletores, alterando constantemente o ambiente e levando a uma dissipação de energia em amplitude, fase e tempo. Isso resulta em múltiplas versões deslocadas no tempo e no espaço do sinal transmitido, provavelmente distorcidas. Além do ambiente ser altamente dinâmico, a velocidade relativa dos receptores com a estação rádio-base, ou de objetos no seu entorno, resultam em modulação aleatórias em frequência devido ao deslocamento Doppler de cada um dos componentes multipercursos [50].

Uma sólida compreensão das estatísticas do canal a ser modelado é de fundamental importância para o dimensionamento de redes sem fio, bem como seus respectivos cálculos de métricas de desempenho. Este capítulo visa revisitar modelos utilizados para caracterização das flutuações rápidas do sinal rádio-móvel. Em particular, relacionados a distribuições Rayleigh e Nakagami-m. O entendimento das estatísticas dessas distribuições servem de aporte teórico para o entendimento do modelo α-μ, adotado na modelagem do sistema sem fio abordada nesta dissertação.

3.2 Propagação Multipercurso

A propagação por múltiplos percursos é caracterizada pela ação conjunta de múltiplas reflexões, espalhamento e difrações do sinal propagado ao longo do percurso de transmissão. Se um único pulso for transmitido por um canal multipercurso, o sinal recebido no receptor é na forma de um trem de pulsos, com um pulso representando a linha de visada, e os outros pulsos sendo as componentes de multipercurso associadas a espalhadores e refletores distintos.

O tempo de atraso entre a primeira e a última componente (linha de visada ou multipercurso) que chega no receptor é conhecido na literatura como atraso de propagação, sendo tipicamente medidos em relação à componente com o qual o demodulador é sincronizado. Esse parâmetro é importante em sistemas sem fio pois indiretamente define diferentes premissas do canal.

O canal multipercurso pode ser modelado como uma resposta ao impulso aleatória de um sistema linear variante no tempo. Desconsiderando-se o ruído, o sinal recebido é dado pela soma de todas as componentes de multipercurso com a componente de linha de visada, na forma [51]:

$$r(t) = \mathrm{Re}\left\{ \sum_{n=0}^{N(t)} \alpha_n(t) u(t - \tau_n(t)) e^{j(2\pi f_c(t-\tau_n(t)) + \phi_{D_n}} \right\} \tag{3.1}$$

em que $N(t)$ componentes multipercursos são somadas. O modelo considera $n = 0$ a componente linha de visada, e $\tau_n(t)$, ϕ_{D_n}, e $\alpha_n(t)$ respectivamente seu atraso, deslocamento Doppler, e ganho da n-ésima componente de multipercurso correspondente. O sinal transmitido é representado em banda base complexa $u(t)$, e $\mathrm{Re}(z)$ a parte real do número complexo z.

Sem perda de generalidade, o atraso de propagação é muito menor que o inverso da largura de banda do canal propagado, ou seja, admitindo a premissa de modelos de canal de banda estreita. Considerando $\phi_n(t) = 2\pi f_c \tau_n(t) - \phi_{D_n}$ toda a influência do efeito Doppler, a Equação (3.1) pode ser reescrita na forma:

$$r(t) = \mathrm{Re}\left\{ u(t) e^{j2\pi f_c t} \left(\sum_{n=0}^{N(t)} \alpha_n(t) e^{-j\phi_n(t)} \right) \right\} \tag{3.2}$$

Como $\alpha_n(t)$ é uma função dependente da perda de percurso e sombreamento, e $\phi_n(t)$ do atraso e do efeito Doppler, geralmente é admitido que esses dois processos aleatórios são independentes, ambos estacionários e ergódicos. Além disso, o fator de escala destacado nos parênteses da Equação (3.2) é independente da envoltória complexa $u(t)$, de forma que o sinal recebido se torna

$$\begin{aligned} r(t) &= \mathrm{Re}\left\{ \left[\sum_{n=0}^{N(t)} \alpha_n(t) e^{-j\phi_n(t)} \right] e^{j2\pi f_c t} \right\} \\ &= r_I(t)\cos(2\pi f_c t) - r_Q(t)\sin(2\pi f_c t), \end{aligned} \tag{3.3}$$

sendo $r_I(t)$ e $r_Q(t)$ as componentes em fase e quadratura do sinal, respectivamente.

3.3 Canal Rayleigh

Nesse modelo físico de canal, é admitido que não há propagação via linha de visada, e os ângulos de chegada no receptor são uniformemente distribuídos. Além disso, é considerado um número alto de caminhos multipercusos $N(t)$.

Da Equação (3.3), podemos representar respectivamente as componentes em fase e quadratura do sinal recebido via múltiplos caminhos como

$$X \triangleq r_I(t) = \sum_{n=1}^{N(t)} \alpha_n(t)\cos\phi_n(t), \tag{3.4}$$

$$Y \triangleq r_Q(t) = \sum_{n=1}^{N(t)} \alpha_n(t)\sin\phi_n(t). \tag{3.5}$$

Como $\alpha_n(t)$ e $\phi_n(t)$ são independentes e $N(t)$ é considerado grande, é possível aplicar o teorema central do limite para aproximar a soma dos fasores aleatórios resultante que compõe cada componente em processos aleatórios conjuntamente Gaussianos. Desse modo, as componentes $r_I(t)$ e $r_Q(t)$ são vistas como variáveis aleatórias Gaussianas [52] tal que $X, Y \sim \mathcal{N}(0, \sigma^2)$.

A envoltória R e fase Φ do sinal desvanecido são respectivamente dadas por

$$R = |r(t)| = |X + jY| \tag{3.6}$$

$$\Phi = \arg(X + jY). \tag{3.7}$$

Como X e Y são Gaussianas com média nula e variância σ^2, possuem suas funções densidade de probabilidade dadas, respectivamente, por

$$f_X(x) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{x^2}{2\sigma^2}\right), -\infty < x < \infty, \tag{3.8}$$

$$f_Y(y) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{y^2}{2\sigma^2}\right), -\infty < y < \infty. \tag{3.9}$$

A função densidade de probabilidade conjunta das componentes em fase e em quadratura é dada por

$$f_{X,Y}(x,y) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2+y^2}{2\sigma^2}\right), -\infty < x, y < \infty, \tag{3.10}$$

Podemos obter $X = R\cos\Phi$ e $Y = R\sin\Phi$ por meio do processo padrão de transformação de variáveis aleatórias [53], de forma que $f_{R,\Phi}(r,\phi) = |J| f_{X,Y}(x,y)$, em que $|J| = R$ corresponde ao módulo do operador Jacobiano J.

A equação resultante do processo de transformação é obtida

$$f_{R,\Phi}(r,\phi) = \frac{r}{2\pi\sigma^2} \exp\left(-\frac{r^2}{2\sigma^2}\right), 0 \leq r < \infty, -\pi \leq \phi < \pi. \tag{3.11}$$

A densidade de probabilidade da envoltória e da fase podem ser obtidas extraindo as marginais, sendo dadas, respectivamente, por

$$f_R(r) = \int_0^\infty f_{R,\Phi}(r,\phi) d\phi \tag{3.12}$$

$$f_\Phi(\phi) = \int_0^{2\pi} f_{R,\Phi}(r,\phi) dr. \tag{3.13}$$

Com isto, é possível obter as funções densidade de probabilidade da envoltória R e da fase Φ de um sinal com desvanecimento de Rayleigh, dadas, respectivamente, por

$$f_R(r) = \frac{r}{\sigma^2} \exp\left(-\frac{r^2}{2\sigma^2}\right), 0 \leq r < \infty \tag{3.14}$$

$$f_\Phi(\phi) = \frac{1}{2\pi}, -\pi \leq \phi < \pi \tag{3.15}$$

Considerando a média quadrática da envoltória $\Omega = \mathbb{E}[R^2] = 2\sigma^2$, com $\mathbb{E}[\cdot]$ o operador esperança [53], é possível obter a versão normalizada das funções densidade de probabilidade de envoltória e fase, respectivamente

$$f_R(r) = \frac{2r}{\Omega} \exp\left(-\frac{r^2}{\Omega}\right), 0 \leq r < \infty \tag{3.16}$$

$$f_\Phi(\phi) = \frac{1}{2\pi}, -\pi \leq \phi < \pi \tag{3.17}$$

Adotando esse modelo de canal, o sinal recebido $r(t)$ possui fase uniforme, como mostra a Equação (3.17) e amplitude com distribuição Rayleigh de acordo com a Equação (3.16),

e ambos são mutualmente independentes.

É possível encontrar a distribuição de probabilidade na potência através da mudança de variáveis $W = R^2 = |r(t)|^2$ em (3.16), resultando

$$p_W(w) = \frac{1}{\Omega}\exp\left(-\frac{w}{\Omega}\right) = \frac{1}{2\sigma^2}\exp\left(-\frac{w}{2\sigma^2}\right), 0 \leq w < \infty \tag{3.18}$$

Esse modelo é chamado de desvanecimento de Rayleigh, e é comumente adotado na literatura por sua fácil tratabilidade e simplicidade em ambientes com um número relativamente baixo de refletores e espalhadores. A potência do sinal recebido $r(t)$ é exponencialmente distribuída com média $\Omega = \mathbb{E}[W] = 2\sigma^2$. O sinal em banda base equivalente é dado por $r_I(t) + jr_Q(t)$, com fase $\Phi = \arctan(r_Q(t)/r_I(t))$.

3.4 Canal Nakagami-m

O modelo Nakagami-m ilustra um ambiente de desvanecimento em que é considerado que os sinais chegam ao receptor através de agrupamentos, conhecidos como *clusters* de multipercurso, decorrentes dos diversos fenômenos de propagação. Este modelo foi inicialmente estudado em [54], com propostas mais recentes considerando o comportamento de fase não-uniforme [55], e o desbalanceamento de potência das componentes em fase e quadratura [56].

De fato, o quadrado da envoltória de uma variável aleatória Nakagami-m pode ser vista como uma soma dos quadrados de m variáveis aleatórias Rayleigh, representando a influência de m *clusters* de multipercurso no sinal recebido. Considerando uma variável aleatória Rayleigh R, o quadrado de uma variável aleatória Nakagami-m R_N pode ser vista na forma

$$R_N^2 = \sum_{i=1}^{m} R_i^2 \tag{3.19}$$

O quadrado da envoltória Rayleigh pode ser decomposto como uma soma dos quadrados de Gaussianas como ilustra a Equação (3.6), produzindo

$$R_N^2 = \sum_{i=1}^{m}(X_i^2 + Y_i^2), \tag{3.20}$$

em que $X_i, Y_i \sim \mathcal{N}(0, \sigma^2)$ correspondem à i-ésima Gaussiana de média nula e variância $\frac{\mathbb{E}[R^2]}{2} = \frac{\Omega}{2} = \sigma^2$. Como as Gaussianas são mutualmente independentes, é possivel obter

$$\begin{aligned} R_N^2 &= \sum_{i=1}^{m} X_i^2 + \sum_{i=1}^{m} Y_i^2 \\ &= X_N^2 + Y_N^2 \end{aligned} \tag{3.21}$$

em que X_N e Y_N representa respectivamente as componentes em fase e quadratura de Nakagami-m com variância $\frac{\mathbb{E}[R_N^2]}{2} = \frac{\Omega_N}{2}$, com $\Omega_N = \sum_{i=1}^{m}\Omega_i = \sum_{i=1}^{m}\mathbb{E}[R_i^2]$. Como R_i são variáveis aleatórias independentes e identicamente distribuídas, temos que $\Omega_N = m\Omega$.

As respectivas funções densidade de probabilidade das componentes de fase e quadratura de uma variável Nakagami são dadas por [55]

$$f_{X_N}(x_N) = \left(\frac{m}{\Omega_N}\right)^{\frac{m}{2}} \frac{|x_N|^{m-1}}{\Gamma(\frac{m}{2})} \exp\left(-\frac{mx_N^2}{\Omega_N}\right), -\infty < x_N < \infty, \tag{3.22}$$

$$f_{Y_N}(y_N) = \left(\frac{m}{\Omega_N}\right)^{\frac{m}{2}} \frac{|y_N|^{m-1}}{\Gamma(\frac{m}{2})} \exp\left(-\frac{my_N^2}{\Omega_N}\right), -\infty < y_N < \infty, \tag{3.23}$$

em que $\Gamma(\cdot)$ corresponde a função Gama [57, eq. (6.1.1)].

Como as comoponentes X_N e Y_N são independentes, a função densidade de probabilidade conjunta pode ser obtida por meio do produto de suas respectivas densidades de probabildade:

$$f_{X_N,Y_N}(x_N, y_N) = \left(\frac{m}{\Omega_N}\right)^{m} \frac{|x_N y_N|^{m-1}}{\Gamma(\frac{m}{2})^2} \exp\left(-\frac{m(x_N^2 + y_N^2)}{\Omega_N}\right), -\infty < x_N, y_N < \infty. \tag{3.24}$$

Por meio da transformação de variáveis aleatórias, encontra-se a função densidade de probabilidade conjunta da fase Φ_N e envoltória R_N, dada por

$$f_{R_N,\Phi_N}(r_N, \phi_N) = \left(\frac{m}{\Omega_N}\right)^{m} \frac{r_N^{2m-1}|\sin\phi_N \cos\phi_N|^{m-1}}{\Gamma(\frac{m}{2})^2} \exp\left(-\frac{mr_N^2}{\Omega_N}\right), \quad 0 \le r_N < \infty, -\pi \le \phi < \pi. \tag{3.25}$$

Após o processo de extração das marginais das densidades similar a Equações (3.12) e (3.13), obtém-se as funções densidade de probabilidade de envoltória e de fase, respectivamente, como

$$f_{R_N}(r_N) = \left(\frac{m}{\Omega_N}\right)^{m} \frac{2r_N^{2m-1}}{\Gamma(m)} \exp\left(-\frac{mr_N^2}{\Omega_N}\right), 0 < r_N < \infty, \tag{3.26}$$

$$f_{\Phi_N}(\phi_N) = \frac{\Gamma(m)|\sin(2\phi_N)|^{m-1}}{2^m\Gamma(\frac{m}{2})^2}, -\pi < \phi_N < \pi. \tag{3.27}$$

A fim de garantir valores fixos para variância e desvio padrão, bem como a independência com o parâmetro Ω_N, pode-se utilizar o processo de transformação de variáveis aleatórias com o objetivo de normalizar a envoltória pela raiz do valor médio quadrático $\sqrt{\Omega_N}$. Desse modo, a densidade de probabilidade conjunta da envoltória e fase Nakagami-m normalizada $P_N = \frac{R_N}{\sqrt{\Omega_N}}$ pode ser obtida substituindo (3.25) em $f_{P_N,\Phi_N}(\rho_N, \phi_N) = f_{R_N,\Phi_N}(r_N, \phi_N)|J|$, com $|J| = \sqrt{\Omega_N}$, produzindo

$$f_{P_N,\Phi_N}(\rho_N, \phi_N) = \frac{m^m \rho_N^{2m-1}|\sin\phi_N \cos\phi_N|^{m-1}}{\Gamma(\frac{m}{2})^2} \exp(-m\rho_N^2), \quad 0 \le \rho_N < \infty, -\pi < \phi_N < \pi. \tag{3.28}$$

Por meio da extração das marginais similar a (3.12) e (3.13), são obtidas as densidades

marginais da envoltória e fase normalizada, respectivamente, dadas por

$$f_{P_N}(\rho_N) = \frac{m^m \rho_N^{2m-1}}{\Gamma(m)} \exp(-m\rho_N^2), 0 \leq \rho_N < \infty, \tag{3.29}$$

$$f_{\Phi_N}(\phi_N) = \frac{\Gamma(m)|\sin(2\phi_N)|^{m-1}}{2^m \Gamma(\frac{m}{2})^2}, -\pi < \phi_N < \pi. \tag{3.30}$$

Também é possível encontrar a distribuição de probabilidade na potência por meio da transformação de variáveis $W = R_N^2$ em (3.26), resultando em

$$f_W(w) = \left(\frac{m}{\Omega_N}\right)^m \frac{w^{m-1}}{\Gamma(m)} \exp\left(-\frac{mw}{\Omega_N}\right). \tag{3.31}$$

Os modelos Rayleigh e Nakagami-m foram derivados admitindo um campo de espalhamento homogêneo, produzido por pontos espalhadores aleatoriamente distribuídos. Essa hipótese é uma aproximação, devido às superfícies serem espacialmente correlacionadas, caracterizando um ambiente de propagação não-linear [58].

3.5 Canal α-μ

Com o intuito de modelar o canal sem fio admitindo-se um ambiente de desvanecimento não-homogêneo, a distribuição α-μ representa o modelo de desvanecimento rápido e descreve as características não-lineares do ambiente de propagação [59]. Este modelo possui dois principais parâmetros que dão origem ao seu nome e, da mesma forma que Nakagami, admite que o sinal recebido é composto por um número de *clusters* de multipercurso, parametrizado por μ. A não-linearidade do meio é modelada em termos do parâmetro de potência $\alpha > 0$.

Por meio da adequada configuração dos parâmetros α e μ, diferentes distribuições podem ser obtidas, como por exemplo Gama, Nakagami-m, Exponencial, Weibull e Rayleigh. Desse modo, α-μ é dito ser um modelo geral de desvanecimento.

O modelo α-μ incrementa Nakagami-m por considerar que a intensidade do sinal recebido não é somente dada pelo módulo da soma dos componentes de multipercurso, mas sim pelo módulo de um exponente de potência α. Desse modo, a envoltória resultante R é obtida como uma função não-linear do módulo da soma dos componentes de multipercurso, dada por

$$R^\alpha = X_\alpha^2 + Y_\alpha^2, \tag{3.32}$$

$$R^\alpha = \sum_{i=1}^{\mu} X_i^2 + \sum_{i=1}^{\mu} Y_i^2, \tag{3.33}$$

em que X_α e Y_α são as componentes em fase e quadratura do modelo α-μ compostas, respectivamente, por uma soma de μ variáveis Gaussianas de média nula e variância $\frac{\mathbb{E}[R^\alpha]}{2\mu} = \frac{\hat{r}^\alpha}{2\mu}$, com $\hat{r}$ correspondendo à α-ésima raiz do valor médio, tal que $\hat{r} = \sqrt[\alpha]{\mathbb{E}[R^\alpha]}$.

Se $\alpha = 2$, a Equação (3.33) se degenera para o caso Nakagami-m, com $\mu = m$. Desse

modo, as seguintes relações são obtidas:

$$R^{\frac{\alpha}{2}} = R_N, \tag{3.34}$$

$$\hat{r}^{\alpha} = \Omega_N. \tag{3.35}$$

Substituindo a Equação (3.35) em (3.22) e (3.23), obtém-se as componentes em fase e quadratura para o modelo α-μ dadas, respectivamente, na forma:

$$f_{X_\alpha}(x) = \left(\frac{\mu}{\hat{r}^{\alpha}}\right)^{\frac{\mu}{2}} \frac{|x|^{\mu-1}}{\Gamma(\frac{\mu}{2})} \exp\left(-\frac{\mu x^2}{\hat{r}^{\alpha}}\right), -\infty < x < \infty, \tag{3.36}$$

$$f_{Y_\alpha}(y) = \left(\frac{\mu}{\hat{r}^{\alpha}}\right)^{\frac{\mu}{2}} \frac{|y|^{\mu-1}}{\Gamma(\frac{\mu}{2})} \exp\left(-\frac{\mu y^2}{\hat{r}^{\alpha}}\right), -\infty < y < \infty. \tag{3.37}$$

Como as componentes X_α e Y_α são variáveis aleatórias independentes, a relação $f_{X_\alpha,Y_\alpha}(x,y) = f_X(x)f_Y(y)$ é válida. Com isto, obtém-se

$$f_{X_\alpha,Y_\alpha}(x,y) = \left(\frac{\mu}{\hat{r}^{\alpha}}\right)^{\mu} \frac{|xy|^{\mu-1}}{\Gamma(\frac{\mu}{2})^2} \exp\left(-\frac{\mu(x^2+y^2)}{\hat{r}^{\alpha}}\right), -\infty < x, y < \infty. \tag{3.38}$$

Com objetivo de obter a função densidade de probabilidade conjunta de fase e envoltória de uma variável α-μ, transformações de variáveis aleatórias são feitas na forma $X = R^{\frac{\alpha}{2}} \cos\Phi$ e $Y = R^{\frac{\alpha}{2}} \sin\Phi$. A densidade conjunta de envoltória e fase obtida é dada por

$$f_{R,\Phi}(r,\phi) = \left(\frac{\mu}{\hat{r}^{\alpha}}\right)^{\mu} \frac{\alpha r^{\alpha\mu-1} |\sin\phi\cos\phi|^{\mu-1}}{\Gamma(\frac{\mu}{2})^2} \exp\left(-\left(\frac{\mu r}{\hat{r}}\right)^{\alpha}\right), 0 \le r < \infty, -\pi \le \phi < \pi. \tag{3.39}$$

Extraindo suas marginais da mesma forma que (3.12) e (3.13), as respectivas funções probabilidade de densidade de envoltória e fase α-μ são obtidas na forma:

$$f_R(r) = \left(\frac{\mu}{\hat{r}^{\alpha}}\right)^{\mu} \frac{\alpha r^{\alpha\mu-1}}{\Gamma(\mu)} \exp\left(-\mu\left(\frac{r}{\hat{r}}\right)^{\alpha}\right), 0 \le r < \infty, \tag{3.40}$$

$$f_\Phi(\phi) = \frac{\Gamma(\mu)|\sin(2\phi)^{\mu-1}|}{2^{\mu}\Gamma(\frac{\mu}{2})^2}, -\pi \le \phi < \pi. \tag{3.41}$$

Novamente, pelo processo de tranformação de variáveis aleatórias, obtém-se a função densidade de probabilidade conjunta de fase e envoltória normalizada $P = R/\hat{r}$, na forma:

$$f_{P,\Phi}(\rho,\phi) = \frac{\alpha\mu^{\mu}\rho^{\alpha\mu-1}|\sin\phi\cos\phi|^{\mu-1}}{\Gamma(\frac{\mu}{2})^2} \exp\left(-\mu\rho^{\alpha}\right), 0 \le \rho < \infty, -\pi \le \phi < \pi. \tag{3.42}$$

Finalmente, extraindo as marginais da densidade conjunta da Equação (3.42), obtém-se as densidades marginais de envoltória normalizada e fase, respectivamente, como:

$$f_R(\rho) = \frac{\alpha\mu^{\mu}\rho^{\alpha\mu-1}}{\Gamma(\mu)} \exp\left(-\mu\rho^{\alpha}\right), 0 \leq \rho < \infty, \tag{3.43}$$

$$f_{\Phi}(\phi) = \frac{\Gamma(\mu)|\sin(2\phi)|^{\mu-1}}{2^{\mu}\Gamma(\frac{\mu}{2})^2}, -\pi \leq \phi < \pi. \tag{3.44}$$

Com intuito de encontrar a distribuição de probabilidade na potência por meio da transformação de variáveis $W = R^2$ da densidade da Equação (3.40), obtém-se a função densidade de probabilidade na potência, na forma:

$$f_W(w) = \frac{\alpha\mu^{\mu}w^{\frac{\alpha\mu}{2}-1}}{2\bar{w}^{\frac{\alpha\mu}{2}}\Gamma(\mu)} \exp\left(-\mu\left(\frac{w}{\bar{w}}\right)^{\frac{\alpha}{2}}\right), \tag{3.45}$$

com $\bar{w} = \mathbb{E}[W] = \hat{r}^2$ definido como a potência normalizada.

3.6 Conclusões

Neste capítulo, descrevemos as estatísticas dos modelos de desvanecimento de Rayleigh, Nakagami-m, e α-μ, em termos das distribuições de probabilidade conjunta e marginais das respectivas componentes em fase e quadratura; e da envoltória e fase do sinal recebido. Essas estatísticas são fundamentais para cálculos de desempenho de redes sem fio, em termos como probabilidade de sucesso/falha de comunicação, probabilidade de erro de *bit*, dentre outros. Em especial, tais expressões servirão para o cálculo da probabilidade de erro na mensagem para canais α-μ, desempenho analisada no modelo de comunicação inteligente abordado nesta dissertação, descrito em detalhes no próximo capítulo.

Capítulo 4

Sistema de Comunicação Dirigido por Dados

4.1 Introdução

Na camada física, sistemas tradicionais de comunicação sem fio são projetados de forma a reduzir sua complexidade focando nas funções específicas de cada bloco de processamento. Transmissor e receptor são divididos em subtarefas como codificação de fonte, codificação de canal, modulação, equalização, demodulação, dentre outros. Essa arquitetura, cuja vantagem é permitir a otimização individual de cada bloco de processamento, é utilizada como arcabouço teórico em sistemas complexos bem sucedidos usados hoje em dia.

Embora a cadeia de processamento independente de blocos como paradigma de comunicação levou ao sucesso de sistemas eficientes de hoje [60], a otimização individual desses blocos de processamento é subótima [1], dado que não otimiza o sistema como um todo.

A modelagem tradicional de sistemas sem fio muitas vezes fica aquém ao tentar capturar a delicada relação entre dados de espectro altamente complexos. Nesse sentido, técnicas de *deep learning* possuem capacidade robusta o suficiente para atender a diferentes requisitos exigidos por sistemas sem fio de próximas gerações, como taxa de dados, mobilidade, latência, densidade de conexão, eficiência energética, capacidade de tráfego, dentre outros [61].

Este capítulo investiga o desempenho do sistema em termo da taxa de erro de bloco em um sistema baseado em aprendizado profundo, por meio de um *Autoencoder*, se comunicando através de um canal de ambiente de desvanecimento geral caracterizado pela distribuição α-μ. A análise do desempenho é dada em termos da capacidade de generalização do sistema inteligente para diferentes ambientes de desvanecimento, comprimentos da mensagem, SNR de treinamento, e taxas de código.

Para comparação, um sistema de comunicação sem fio tradicional é apresentado, onde seu desempenho é analisado também em termos da taxa de erro de bloco. O sistema investigado é treinado com mensagens aleatórias uniformemente geradas, sem nenhum conhecimento prévio no domínio de comunicações sem fio. Ainda sim, seu desempenho corresponde ao desempenho de soluções de engenharia humana adotadas atualmente, como será demonstrado a seguir.

4.2 Sistema de Comunicação Tradicional

Considere um sistema de comunicação ponto a ponto em sua forma mais simples, constituído por três principais blocos de processamento: transmissor, canal, e receptor [1]. O transmissor envia uma mensagem ao receptor através do canal, e a comunicação é dita bem sucedida se o receptor e o transmissor concordarem no que foi enviado.

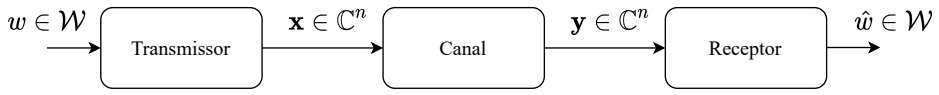

Figura 4.1: Ilustração de um sistema de comunicação simples.

A tarefa do transmissor é selecionar e mapear símbolos de uma fonte de informação $w \in \mathcal{W} = \{1, 2, \ldots, M\}$ em uma sequência de n símbolos do canal $\mathbf{x} \in \mathbb{C}^n$ que compõem o sinal transmitido, realizando n usos discretos do canal. Para isto, aplica a transformação $f\colon \mathcal{W} \mapsto \mathbb{C}^n$ na mensagem w a fim de gerar o sinal transmitido $\mathbf{x} = f(w) \in \mathbb{C}^n$.[1] O vetor transmitido é sujeito a uma restrição de potência no *hardware* tal que $||\mathbf{x}||_2^2 \leq n$. Cada mensagem é representada por uma sequência de *bits* de comprimento $k = \log_2(M)$. Desse modo, a taxa de comunicação deste sistema de comunicação é $R = k/n$ [*bits*/transmissão].

O canal age como um sistema estocástico descrito por sua matriz de transição definida pela função densidade de probabilidade condicional $p(\mathbf{y}|\mathbf{x})$ entre sua entrada e saída. Na recepção, uma versão possivelmente distorcida $\mathbf{y} \in \mathbb{C}^n$ de $\mathbf{x}$ pode ser observada. Desse modo, o objetivo do receptor é aplicar a transformação determinística $g\colon \mathbb{C}^n \mapsto \mathcal{W}$ a fim de produzir uma estimativa $\hat{w} = g(\mathbf{y})$ da mensagem original w com erro arbitrariamente pequeno.

A relação entrada-saída do i-ésimo uso de canal pode ser escrito na forma

$$y_i = x_i h_i + n_i, \tag{4.1}$$

em que n_i captura o efeito do ruído aditivo branco (AWGN, do inglês *Additive White Gaussian Noise*) tal que $n_i \sim \mathcal{N}(0, N_0/2)$. A densidade espectral de potência unilateral do ruído é dada por N_0, o coeficiente de desvanecimento h_i segue as estatísticas do modelo de desvanecimento α-μ, enquanto x_i e y_i representam a entrada e a saída do canal, respectivamente.

Em ambientes de desvanecimento, a potência do sinal recebido varia aleatoriamente com a distância e com o tempo, resultado de fenômenos como sombreamento e desvanecimento rápido. Analisar sistemas cuja potência recebida é aleatória depende da taxa de mudança do desvanecimento.

Para esta dissertação, consideramos um sistema de comunicação sem fio ideal com temporização e sincronização perfeitas de frequência e fase da portadora. Além disso, é considerado que o nível de desvanecimento do sinal é constante durante o período de n símbolos. Dessa forma, a razão sinal-ruído por *bit* $\gamma_b = E_b/N_0$ é constante durante esse

[1] Para implementação, consideramos o mapeamento complexo como um mapeamento para $\mathbb{R}^{2n}$, interpretado como uma concatenação da parte real e imaginária de $\mathbf{x}$.

período, onde E_b é a energia por *bit*. Para especificações de desempenho, tipicamente estamos interessados na probabilidade de erro de *bit* média $\bar{P}_b$ em função de γ_b.

A probabilidade de erro média é computada integrando-se a probabildade de erro no canal AWGN sobre a distribuição de desvanecimento, dada na forma [51]:

$$\bar{P}_b = \int_0^\infty P_b(\gamma_b) f_{\gamma_b}(\gamma) d\gamma, \tag{4.2}$$

em que $f_{\gamma_b}(\gamma)$ pode ser obtida por meio da transformação de variáveis para uma dada distribuição de amplitude de desvanecimento, ou envoltória R, como Rayleigh, Nakagami-m, α-μ, dentre outros. Para o canal α-μ, a Equação (3.45) descreve sua distribuição na potência com a SNR instantânea $\gamma_b = W = R^2$, e SNR média $\bar{\gamma_b} = \mathbb{E}[\gamma_b]$. A probabilidade de erro de *bit* determinística sobre o canal AWGN $P_b(\gamma_b)$ é calculada para diferentes esquemas de modulação digital.

A probabilidade de erro na mensagem (BLER, do inglês *Block Error Rate*) é dada por:

$$P_{eM} = \text{BLER} = 1 - (1 - \bar{P}_b)^{\log_2 M}, \tag{4.3}$$

em que M é a ordem da modulação.

O processo de modulação digital em sistemas de comunicação convencional possui como princípio básico a codificação da informação na onda portadora a ser transmitida pelo canal de comunicação. Diferentes modulações possuem diferentes diagramas de constelação fixas e preestabelecidas. A taxa de dados desejada determina o esquema de constelação e agrupamento de *bits* de entrada para construção do símbolo. Desse modo, o objetivo da modulação é enviar informação com alta taxa de dados, enquanto minimiza a probabilidade de erro dos dados. Regiões de decisão linear simplifica a decisão das informações no receptor para demodulação.

Em sistemas com modulação por chaveamento de mudança de fase (PSK, do inglês *Phase-Shift Keying*), toda a informação é codificada na fase dos sinais transmitidos. Neste trabalho, consideramos dois casos denominados chaveamento binário de mudança de fase (BPSK, do inglês *Binary Phase-Shift Keying*) e chaveamento de mudança de fase em quadratura (QPSK, do inglês *Quadrature Phase-Shift Keying*), com ordens de modulação $M = 2$ e $M = 4$, respectivamente. Para ambos os casos, a probabilidade de erro de *bit* sobre o canal AWGN $P_b(\gamma)$ é aproximada na forma [51]:

$$P_b \approx Q(\sqrt{2\bar{\gamma}_b}), \tag{4.4}$$

em que $Q(\cdot)$ representa a área sob a cauda da função densidade de probabilidade Gaussiana normalizada tal que $Q(x) \triangleq \frac{1}{\sqrt{2\pi}} \int_x^\infty \exp(-\frac{u^2}{2}) du$.

Além da modulação PSK, consideramos a modulação M-ária por amplitude em quadratura (QAM, do inglês *Quadrature Amplitude Modulation*) retangular, em que a informação é codificada tanto na amplitude, quanto na fase da onda portadora. Para esse esquema, a probabilidade de erro de *bit* sobre o canal AWGN é dado, aproximadamente, na forma [62]:

$$P_b \approx \frac{4}{\log_2 M}\left(1 - 2^{-\frac{\log_2 M}{2}}\right) Q\left(\sqrt{\frac{3\bar{\gamma}_b \log_2 M}{M-1}}\right). \tag{4.5}$$

4.3 *Autoencoder* como Sistema de Comunicação

O sistema de comunicação ilustrado na Figura 4.1 pode ser visto como um problema de reconstrução fim-a-fim em que as mensagens transmitidas são reconstruídas no receptor em um canal físico [15]. Nessa abordagem, transmissor e receptor, bem como suas funções codificadora e decodificadora, são implementados como redes neurais.

Um *Autoencoder* com redes convolucionais é considerado nesta dissertação, cuja estrutura é ilustrada na Figura 4.2. As camadas convolucionais 1D (Conv1D), cuja transformação é descrita na Equação (2.23), permite que o transmissor processe uma sequência de símbolos S, onde um número total de $k \times L$ *bits* são manuseados simultaneamente, em que $k = \log_2 M$ é o número de *bits* por símbolo e L o número de símbolos (comprimento de bloco). A entrada converte cada símbolo recebido da sequência S para um vetor *one-hot* $\mathbb{1}_s \in \{0,1\}^M$, isto é, um vetor M-dimensional cujo s-ésimo elemento é igual a um, e os demais nulos.

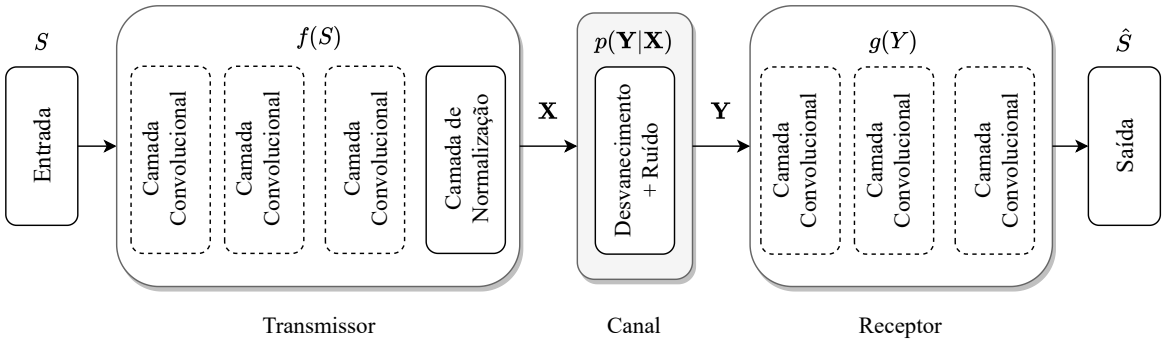

Figura 4.2: Representação do *autoencoder* de comunicação. As camadas em tracejado representam as camadas cujos pesos são modificados no aprendizado.

O transmissor consiste em três camadas convolucionais seguidas de uma camada de normalização. Em uma perspectiva de codificação de fonte, as camadas convolucionais do transmissor permitem uma codificação de bloco linear/não-linear. As operações de convolução facilitam a codificação linear, enquanto que as funções de ativações permitem a não-linearidade na codificação. Em uma perspectiva de modulação, o transmissor realiza a transformação $\mathbf{X} = f(S)$ a fim de expandir a entrada em um espaço de 256 dimensões e, por meio do mapeamento de cada vetor *one-hot* nesse espaço, permitir buscar a representação mais adequada do símbolo de entrada [21]. A camada de normalização é usada para satisfazer a restrição de potência do transmissor, comprimindo a representação dos símbolos de dimensão 256 para um espaço de dimensão $2n$, considerando que cada *slot* de tempo possui componentes em fase e quadratura, de forma que os pontos da constelação são projetados em um espaço de dimensão $2n$.

O canal é representado como uma probabilidade condicional $p(\mathbf{Y}|\mathbf{X})$ seguindo as estatísticas α-μ, composto por uma camada de rúido AWGN com variância fixa $\sigma = (2R\bar{\gamma}_b)^{-1}$.

O receptor é responsável por classificar cada sinal de chegada $\mathbf{Y}$ de M possibilidades com base nas características aprendidas do símbolo. As três camadas convolucionais são responsáveis por expandir novamente o sinal recebido $\mathbf{Y}$ em um vetor de 256 dimensões, buscando extrair informação latente adequada, capazes de capturar representações intrínsecas de $\mathbf{Y}$. A última camada do receptor mapeia essa informação latente em um vetor *one-hot* de comprimento M, modulados em vetores probabilísticos com saída $\mathbf{p} \in (0,1)^M$, por meio da ativação *softmax*. Todo este processo é mapeado pela transformação $\hat{S} = g(\mathbf{Y})$, em que a mensagem decodificada corresponde ao índice cujo elemento em $\mathbf{p}$ possui maior probabilidade. Além disso, a Tabela 4.1 ilustra as demais funções de ativações adotadas nas camadas do modelo, bem como toda a estrutura do *Autoencoder*.

As duas redes que compõem o *Autoencoder* de comunicação podem ser otimizadas fim-a-fim, caracterizando um problema de otimização irrestrita na forma da Equação (2.16), minimizando a função custo de entropia cruzada binária, dada na Equação (2.18), entre $\mathbb{1}_s$ e $\mathbf{p}$.

Tabela 4.1: Parâmetros das redes que compõem o *autoencoder*.

Bloco	Camada	Ativação	Dimensão de saída
Transmissor	Entrada	-	L × M
	Conv1D	ELU	L × 256
	Conv1D	ELU	L × 256
	Conv1D	Linear	L × 2n
	Normalização	-	L × 2n
Canal	Desvanecimento + Ruído	-	L × 2n
Receptor	Conv1D	ELU	L × 256
	Conv1D	ELU	L × 256
	Conv1D	*Softmax*	L × M

4.4 Análise de Desempenho

Nesta seção, numerosas simulações computacionais são conduzidas com o intuito de investigar a taxa de erro na mensagem (BLER), ou $\Pr(s \neq \hat{s})$, do *Autoencoder* se comunincando em um canal com desvanecimento α-μ. O modelo ilustrado na Figura 4.2 foi empregado com todos os parâmetros de rede dados na Tabela 4.1, adotando um tamanho unitário para os *kernels* convolucionais e *stride*, lidando com cada símbolo individualmente. Os dados de treino e teste são gerados aleatoriamente usando sequências de *bits* extraídas independente e identicamente (iid) de uma distribuição uniforme. Além disso, camadas de normalização em lote [63] são empregadas após as camadas convolucionais para melhorar a estabilidade da rede. Além dos parâmetros do canal (α e μ) e da taxa desejada $R = k/n$, precisamos fornecer a SNR média para o treinamento da rede.

O sistema de auto-aprendizagem dirigido por dados foi implementado em Keras [64], treinado usando 16000 mensagens, onde cada mensagem contém um bloco de L símbolos, contendo k *bits* de informação cada. A rede foi testada através de 80000 mensagens, e otimizada fim-a-fim com o método adaptativo Adam [42] com passo de aprendizado igual

a 10^{-3} e decaimento de um fator de 10 quando saturado, a fim de garantir convergência rápida. Para o treinamento, 50 épocas se mostraram suficientes.

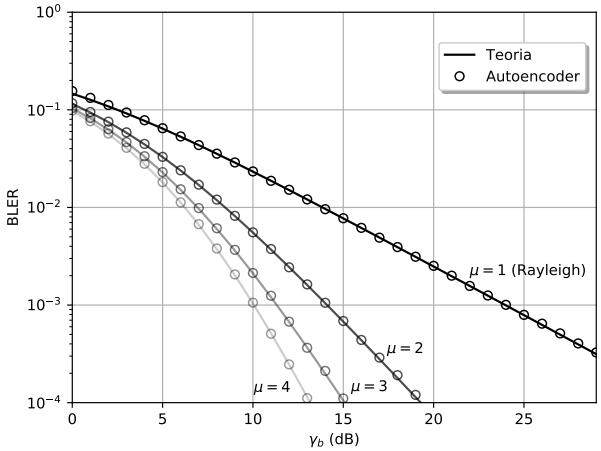

Figura 4.3: Desempenho BLER do sistema de auto-aprendizagem sob o canal de desvanecimento Nakagami-m. Resultado analítico para modulação BPSK é representado como curvas sólidas.

Fixando $\alpha = 2$, o desvanecimento em questão se torna o de Nakagami-m, com $\mu = m$. A Figura 4.3 ilustra esse cenário de desvanecimento para $R = 1$, com $k = 1$ e $n = 1$, comparando o desempenho de BLER do sistema de auto-aprendizagem treinado a uma SNR fixa de $\bar{\gamma}_b = 14$dB. Fixando $\mu = 1$, a distribuição α-μ se degenera para a distribuição Weibull [65] para diferentes α. Esse cenário é ilustrado na Figura 4.4, com $R = 2$ e *Autoencoder* treinado a SNR fixa $\bar{\gamma}_b = 16$dB. Em ambos esquemas, o transmissor aprendeu representações significativas da entrada e o receptor aprendeu a equalizar os severos efeitos de desvanecimento antes da decodificação, de forma que o desempenho do *Autoencoder* é compatível com o desempenho de modulações ótimas clássicas BPSK (para $R = 1$) e QPSK ($R = 2$) em todo o alcance de SNR, ilustradas em curvas sólidas.

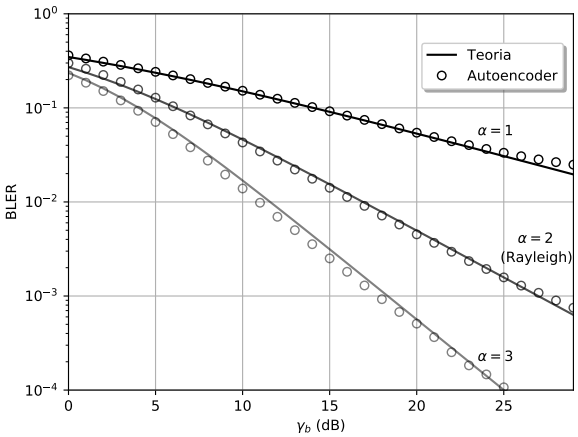

Figura 4.4: Desempenho BLER do sistema inteligente sob o canal de desvanecimento Weibull. Resultado analítico para modulação QPSK representado como curvas sólidas.

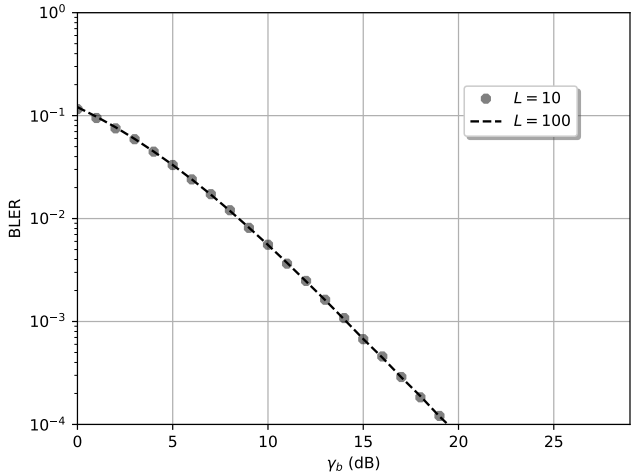

Figura 4.5: Comparação do desempenho BLER para transmissão de diferentes comprimento de bloco.

A generalização para qualquer comprimento de bloco L é ilustrada na Figura 4.5. Uma transmissão de $R = 1$ é adotada com desvanecimento Nakagami-m ($\alpha = 2$) com $\mu = 2$. É observado que o desempenho é exatamente a mesma para blocos de tamanhos substancialmente diferentes, com $L = 10$ e $L = 100$. O sistema dirigido por dados é apto a se comunicar com máximo desempenho não importando o tamanho do bloco transmitido. Desse modo, por simplicidade, todas as curvas apresentadas nesta dissertação é com um comprimento fixo de bloco tal que $L = 10$.

Uma realização do treinamento é ilustrada na Figura 4.6, por meio da progressão do erro a ser minimizado na otimização iterativa. É observado que o sistema rapidamente converge. Isso foi facilitado pelo uso do passo adaptativo Adam com fator de decaimento, juntamente com a ação das camadas de normalização em lote [63].

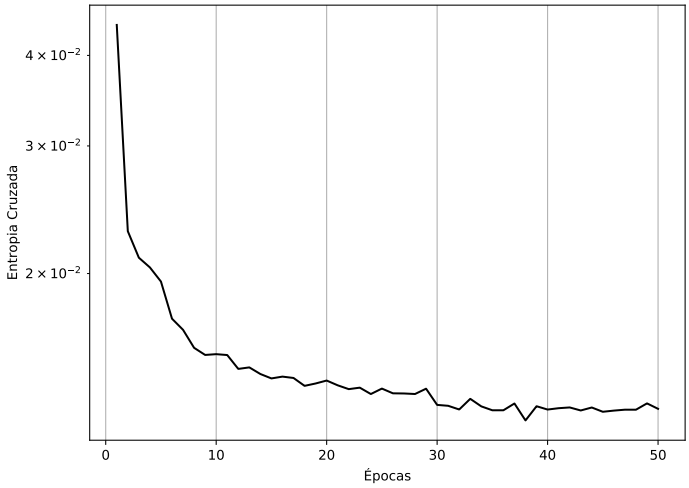

Figura 4.6: Progressão do erro de treinamento.

Em se tratando de modulação, o *Autoencoder* é capaz de projetar constelações no espaço $2n$-dimensional, com potencial capacidade de maximizar a distância Euclidiana mínima entre pontos da constelação, em contraste ao espaço bidimensional referente às componentes em fase e quadratura. No que diz respeito à codificação de canal, o *Autoencoder* aprendeu uma codificação através do espaço $2n$-dimensional capaz de aumentar a distância mínima de *hamming* entre palavras-código [21].

Ao aumentar o número de usos do canal, é esperado um melhor desempenho BLER. De fato, a Figura 4.7 ilustra um ganho substancial de desempenho ao aumentar para $n = 2$. Além disso, ilustra o desempenho do sistema inteligente para diferentes taxas de comunicação $R = 4$ e $R = 6$, treinados sob SNR fixa $\gamma_b = 27$dB. É observado que o *Autoencoder* possui um desempenho levemente superior comparado aos esquemas clássicos de modulação 16-QAM (com $R = 4$) e 64-QAM (com $R = 6$), quando transmitidos sob o canal com parâmetros $\alpha = 3$ e $\mu = 2$, possivelmente devido às curvas terem sidos geradas através de aproximações.

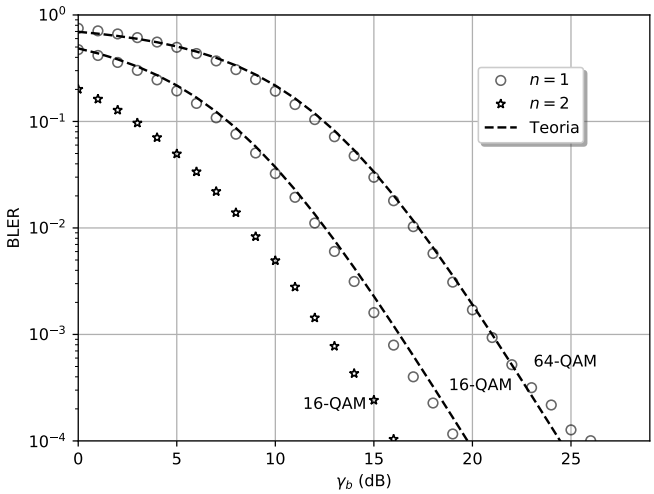

Figura 4.7: Desempenho do *Autoencoder* para diferentes usos de canal $n = 1, 2$ e taxas de comunicação $R = 4$ e $R = 6$. As curvas analíticas são representadas como linhas tracejadas.

O sistema foi capaz de aprender transformações adequadas dos símbolos, mesmo para elevadas taxas de comunicação. A arquitetura profunda com camadas convolucionais facilitou a busca por tais transformações através dos 256 *kernels*. Embora cada sequência de entrada possua M^L possibilidades, o *Autoencoder* para comunicação foi treinado usando apenas 16000 mensagens. Isso significa que a rede é capaz de generalizar a decodificação de palavras-códigos não vistas antes.

4.5 Conclusões

Neste capítulo, traçamos um paralelo entre um sistema de comunicação tradicional e um sistema inteligente dirigido por dados, com base em *deep learning*, otimizado fim-a-fim

para modelagem do sistema completo. As transformações de codificação e decodificação, bem como a influência do ruído e desvanecimento do canal, são apresentadas para ambos paradigmas. Além disso, o desempenho dos sistemas (tradicional e inteligente) é analisada em termos da probabilidade de erro na mensagem enviada.

Os resultados obtidos mostram que o *Autoencoder* consegue modelar todo o sistema de comunicação sem fio fim-a-fim, bem como seus principais elementos influentes, sem nenhum conhecimento prévio em termos de parâmetros para caracterização do mesmo, exceto pelo conhecimento pleno das estatísticas do canal no receptor. A arquitetura analisada se mostra potencialmente proveitosa para sistemas sem fio emergentes, com alta capacidade de generalização em termos de diferentes ambientes de desvanecimento, comprimentos da mensagem, SNR de treinamento, e taxas de código.

Capítulo 5

Considerações Finais

O projeto e a implementação de sistemas de comunicações sem fio são construídos sob fortes modelos analíticos probabilísticos e premissas como o meio de propagação e a presença ou não de linha de visada. Nesse contexto, tais modelos podem não capturar precisamente efeitos de tecnologias de aplicações sem fio emergentes, tais como realidade aumentada, realidade virtual, IoT, e compartilhamento espectral. Apenas recentemente *deep learning* passou a se mostrar potencialmente proveitoso para aplicações em comunicações sem fio. Portanto, ainda existem poucos estudos na literatura analisando o impacto de aprendizagem profunda em sistemas de comunicações, especialmente na camada física.

Nesta dissertação, um sistema de comunicação baseado em aprendizado profundo é investigado, o *Autoencoder* para comunicações. A ideia principal é modelar um sistema que substitua toda a cadeia de comunicação, sem necessariamente precisar passar por processos fixos como codificação/decodificação e modulação/demodulação. A arquitetura do *Autoencoder* consiste em duas redes neurais convolucionais, caracterizando o codificador e decodificador, com uma camada extra fixa entre elas, representando o canal restrito às estatísticas α-μ. As redes são conjuntamentes treinadas de maneira fim-a-fim, otimizando uma única função custo. Essa perspectiva se difere completamente da abordagem tradicional, em que blocos fixos de processamento são individualmente otimizados, sendo, portanto, uma solução global subótima.

O desempenho do sistema analisado nesse trabalho foi avaliado em termos da taxa de erro na mensagem, em que foi validado que o sistema capaz de autoaprender consegue atingir desempenho semelhante, comparado com modulações tradicionais, desenvolvidas por meio da engenharia humana. Para comparação, a taxa de erro de bloco analítica é explorada sob o mesmo ambiente de desvanecimento generalizado. Além disso, mostramos que a capacidade de generalização é mantida não importando tamanho do alfabeto da fonte, comprimento de bloco, alcance de SNR, taxas de código e usos discreto do canal.

A categorização de técnicas de *deep learning* aplicadas na camada física de sistemas de comunicações sem fio estão em fase de consolidação. Nesse contexto, aplicações de *Autoencoders* estão, em sua grande maioria, em transição de fase entre simulação e implementação no mundo real. Desde sua proposição, diferentes trabalhos explorando *Autoencoders* em diferentes sistemas e configurações foram analisados, mas nenhum deles considerou um cenário de desvanecimento geral.

5.1 Investigações Futuras

Esta dissertação visou ilustrar o poder promissor de abordagens de *deep learning* aplicadas em sistemas de comunicação. Desse modo, é fruto de um trabalho em andamento e possui vasta área no ramo da pesquisa que pode ser investigada, voltada a aplicações de Inteligência Artificial em sistemas de comunicações sem fio, baseada nos resultados aqui ilustrados. Listamos algumas propostas de trabalhos futuros a serem feitos relacionados a esta dissertação:

- Relacionado à interpretabilidade do sistema. Métodos de otimização baseados em aprendizagem profunda ainda não possuem sólidas interpretações físicas e matemáticas. Sem explicações físicas claras, não conseguimos facilmente identificar o motivo de certa estrutura alcançar determinado desempenho, nem a escolha precisa da estratégia de treinamento. Como o treinamento em si é um problema de otimização, teorias avançadas de otimização podem ajudar a encontrar funções custos e estratégias de treino mais adequadas.

- Relacionado aos dados. Adquirir uma quantidade suficiente de dados de alta qualidade para treinar sistemas sem fio reais é desafiador, como a transmissão de grande quantidade de pacotes como dados rotulados, devido à alta eficiência espectral requerida em ambientes sem fio. Além disso, dados em comunicações sem fio são geralmente de altas dimensões com alta heterogeneidade. Desse modo, métodos mais eficientes de tratamento desses dados podem ser estudados, permitindo um fluxo de processamento mais específico e criterioso.

- Relacionado a algoritmos. Abordagens de aprendizado profundo não foram diretamente adequadas para lidar com ambientes variantes no tempo. Nesse sentido, se um canal sem fio mudar rapidamente, sistemas sem fio baseados em *deep learning* devem ser robustos e versáteis o suficiente, podendo ser frequente e completamente re-treinados do começo a fim de manter o desempenho durante o tempo, possivelmente tendo um alto custo computacional. Uma possível solução para isso é considerar o uso de aprendizagem por transferência [2], em que redes treinadas em determinados ambientes são aplicadas em novos cenários similares, atingindo, portanto, melhor robustez e adaptatividade ambiental. Nesse contexto, faz-se necessário investigar algoritmos capazes de lidar com essa adaptatividade no tempo, promovendo o desenvolvimento tanto teórico quanto algorítmico.

- Relacionado à implementação. Sistema de comunicação sem fio clássico tem sua infraestrutura equipada principalmente com funcionalidades para processamento de sinais e radiofrequência. Para implementar tais sistemas com métodos baseados em aprendizagem profunda, servidores em nuvem devem ser desenvolvidos levando em consideração a infraestrutura existente. Por último, novas arquiteturas capazes de tomarem proveito, coexistindo em um mesmo sistema, tanto de métodos tradicionais quanto métodos dirigidos por dados, são diretrizes interessantes para análises e formam um assunto interessante de estudo.

As propostas de trabalhos futuros aqui apresentadas pretendem complementar a emergente abordagem de aplicação de *deep learning* em sistemas de comunicações sem fio. Esse recente campo de estudo tem despertado interesse em diversos pesquisadores da área que, motivados pelos poderosos resultados obtidos em diferentes domínios, recentemente têm publicado com o objetivo de melhor fundamentar essa intersecção. Acreditamos que tais propostas poderão contribuir de forma significativa neste sentido.

Referências Bibliográficas

[1] C. E. Shannon. A mathematical theory of communication. *The Bell System Technical Journal*, 27(3):379–423, 1948.

[2] Y. Bengio I. Goodfellow and A. Courville. *Deep Learning*. MIT Press, 2016.

[3] Linglong Dai, Ruicheng Jiao, Fumiyuki Adachi, H. Vincent Poor, and Lajos Hanzo. Deep learning for wireless communications: An emerging interdisciplinary paradigm. *IEEE Wireless Communications*, 27(4):133–139, 2020.

[4] Nariman Farsad and Andrea Goldsmith. Neural network detection of data sequences in communication systems. *IEEE Transactions on Signal Processing*, 66(21):5663–5678, 2018.

[5] Neev Samuel, Tzvi Diskin, and Ami Wiesel. Deep mimo detection. In *2017 IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pages 1–5, 2017.

[6] Neev Samuel, Tzvi Diskin, and Ami Wiesel. Learning to detect. *IEEE Transactions on Signal Processing*, 67(10):2554–2564, 2019.

[7] David Neumann, Thomas Wiese, and Wolfgang Utschick. Learning the mmse channel estimator. *IEEE Transactions on Signal Processing*, 66(11):2905–2917, 2018.

[8] Mehran Soltani, Vahid Pourahmadi, Ali Mirzaei, and Hamid Sheikhzadeh. Deep learning-based channel estimation. *IEEE Communications Letters*, 23(4):652–655, 2019.

[9] Tobias Gruber, Sebastian Cammerer, Jakob Hoydis, and Stephan ten Brink. On deep learning-based channel decoding. In *2017 51st Annual Conference on Information Sciences and Systems (CISS)*, pages 1–6, 2017.

[10] Yihan Jiang, Sreeram Kannan, Hyeji Kim, Sewoong Oh, Himanshu Asnani, and Pramod Viswanath. Deepturbo: Deep turbo decoder. In *2019 IEEE 20th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pages 1–5, 2019.

[11] Sreeraj Rajendran, Wannes Meert, Domenico Giustiniano, Vincent Lenders, and Sofie Pollin. Deep learning models for wireless signal classification with distributed low-cost spectrum sensors. *IEEE Transactions on Cognitive Communications and Networking*, 4(3):433–445, 2018.

[12] Nariman Farsad, Milind Rao, and Andrea Goldsmith. Deep learning for joint source-channel coding of text. In *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 2326–2330, 2018.

[13] Hao Ye, Geoffrey Ye Li, and Biing-Hwang Juang. Power of deep learning for channel estimation and signal detection in ofdm systems. *IEEE Wireless Communications Letters*, 7(1):114–117, 2018.

[14] Timothy J. O'Shea, Kiran Karra, and T. Charles Clancy. Learning to communicate: Channel auto-encoders, domain specific regularizers, and attention. In *2016 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, pages 223–228, 2016.

[15] Timothy O'Shea and Jakob Hoydis. An introduction to deep learning for the physical layer. *IEEE Transactions on Cognitive Communications and Networking*, 3(4):563–575, 2017.

[16] Timothy J. O'Shea, Latha Pemula, Dhruv Batra, and T. Charles Clancy. Radio transformer networks: Attention models for learning to synchronize in wireless systems. In *2016 50th Asilomar Conference on Signals, Systems and Computers*, pages 662–666, 2016.

[17] Sebastian Dörner, Sebastian Cammerer, Jakob Hoydis, and Stephan ten Brink. Deep learning based communication over the air. *IEEE Journal of Selected Topics in Signal Processing*, 12(1):132–143, 2018.

[18] Fayçal Ait Aoudia and Jakob Hoydis. End-to-end learning of communications systems without a channel model. In *2018 52nd Asilomar Conference on Signals, Systems, and Computers*, pages 298–303, 2018.

[19] Timothy J. O'Shea, Tugba Erpek, and T. Charles Clancy. Physical layer deep learning of encodings for the mimo fading channel. In *2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 76–80, 2017.

[20] Banghua Zhu, Jintao Wang, Longzhuang He, and Jian Song. Joint transceiver optimization for wireless communication phy using neural network. *IEEE Journal on Selected Areas in Communications*, 37(6):1364–1373, 2019.

[21] Nan Wu, Xudong Wang, Bin Lin, and Kaiyao Zhang. A cnn-based end-to-end learning framework toward intelligent communication systems. *IEEE Access*, 7:110197–110204, 2019.

[22] W. Mcculloch and W. Pitts. A logical calculus of ideas immanent in nervous activity. *Bulletin of Mathematical Biophysics*, 5:127–147, 1943.

[23] F. Rosenblatt. *The Perceptron, a Perceiving and Recognizing Automaton Project Para*. Report: Cornell Aeronautical Laboratory. Cornell Aeronautical Laboratory, 1957.

[24] B. Widrow and M. E. Hoff. Adaptive switching circuits. In *1960 IRE WESCON Convention Record, Part 4*, pages 96–104, New York, 1960. IRE.

[25] M. Minsky and S. Papert. *Perceptrons: An Introduction to Computational Geometry.* MIT Press, Cambridge, MA, USA, 1969.

[26] J. J. Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences of the United States of America*, 79(8):2554–2558, April 1982.

[27] P. Smolensky. Information processing in dynamical systems: Foundations of harmony theory. In *Parallel distributed processing: Explorations in the microstructure of cognition*, pages 194–281–. MIT Press, Cambridge, MA, 1986.

[28] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. *Learning Internal Representations by Error Propagation*, page 318–362. MIT Press, Cambridge, MA, USA, 1986.

[29] D. E. Rumelhart, G. E. Hinton, and J. L. McClelland. *A General Framework for Parallel Distributed Processing*, page 45–76. MIT Press, Cambridge, MA, USA, 1986.

[30] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36:193–202, 1980.

[31] G E Hinton and R R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, July 2006.

[32] Y. Bengio, P. Simard, and P. Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

[33] Vinod Nair and Geoffrey E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, page 807–814, Madison, WI, USA, 2010. Omnipress.

[34] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). In Yoshua Bengio and Yann LeCun, editors, *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, 2016.

[35] Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *in ICML Workshop on Deep Learning for Audio, Speech and Language Processing*, 2013.

[36] G. Cybenko. Approximation by superpositions of a sigmoidal function. *Mathematics of Control, Signals, and Systems (MCSS)*, 2(4):303–314, December 1989.

[37] F.J. Von Zuben. *Notas de Aulas do Curso "Redes Neurais" (IA353)*. FEEC-UNICAMP, 2020.

[38] Jan Kukacka, Vladimir Golkov, and Daniel Cremers. Regularization for deep learning: A taxonomy. *CoRR*, abs/1710.10686, 2017.

[39] James Martens. Deep learning via hessian-free optimization. In Johannes Fürnkranz and Thorsten Joachims, editors, *ICML*, pages 735–742. Omnipress, 2010.

[40] James Martens and Ilya Sutskever. Training deep and recurrent networks with hessian-free optimization. In *Neural Networks: Tricks of the Trade*, 2012.

[41] Yann N Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 27. Curran Associates, Inc., 2014.

[42] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[43] Xavier Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. *Journal of Machine Learning Research - Proceedings Track*, 9:249–256, 01 2010.

[44] Y. LeCun. Generalization and network design strategies. In R. Pfeifer, Z. Schreter, F. Fogelman, and L. Steels, editors, *Connectionism in Perspective*, Zurich, Switzerland, 1989. Elsevier. an extended version was published as a technical report of the University of Toronto.

[45] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[46] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[47] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Computer Vision and Pattern Recognition (CVPR)*, 2015.

[48] Dominik Scherer, Andreas Müller, and Sven Behnke. Evaluation of pooling operations in convolutional architectures for object recognition. In Konstantinos Diamantaras,

Wlodek Duch, and Lazaros S. Iliadis, editors, *Artificial Neural Networks – ICANN 2010*, pages 92–101, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[49] Jonathon Shlens. A tutorial on principal component analysis. *CoRR*, abs/1404.1100, 2014.

[50] Simon Haykin. *Communication Systems*. Wiley Publishing, 5th edition, 2009.

[51] Andrea Goldsmith. *Wireless Communications*. Cambridge University Press, USA, 2005.

[52] Lord Rayleigh F.R.S. Xii. on the resultant of a large number of vibrations of the same pitch and of arbitrary phase. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, 10(60):73–78, 1880.

[53] A. Papoulis. *Probability, random variables, and stochastic processes*. McGraw-Hill, 1991.

[54] M. NAKAGAMI. The m-distribution, a general formula of intensity distribution of rapid fading. *Statistical Methods in Radio Wave Propagation*, 1960.

[55] Michel Daoud Yacoub. Nakagami-m phase-envelope joint distribution: An improved model. In *2009 SBMO/IEEE MTT-S International Microwave and Optoelectronics Conference (IMOC)*, pages 335–339, 2009.

[56] Michel Daoud Yacoub. Nakagami-m phase–envelope joint distribution: A new model. *IEEE Transactions on Vehicular Technology*, 59(3):1552–1557, 2010.

[57] Milton Abramowitz and Irene A. Stegun. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Dover, New York, ninth dover printing, tenth gpo printing edition, 1964.

[58] W.R. Braun and U. Dersch. A physical mobile radio channel model. *IEEE Transactions on Vehicular Technology*, 40(2):472–482, 1991.

[59] Michel Daoud Yacoub. The α-μ distribution: A physical fading model for the stacy distribution. *IEEE Transactions on Vehicular Technology*, 56(1):27–34, 2007.

[60] E. Zehavi. 8-psk trellis codes for a rayleigh channel. *IEEE Transactions on Communications*, 40(5):873–884, 1992.

[61] Manuel Eugenio Morocho-Cayamcela, Haeyoung Lee, and Wansu Lim. Machine learning for 5g/b5g mobile and wireless communications: Potential, limitations, and future directions. *IEEE Access*, 7:137184–137206, 2019.

[62] John R. Barry, David G. Messerschmitt, and Edward A. Lee. *Digital Communication: Third Edition*. Kluwer Academic Publishers, USA, 2003.

[63] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Francis Bach and David Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, Lille, France, 07–09 Jul 2015. PMLR.

[64] François Chollet et al. Keras. `https://keras.io`, 2015.

[65] W. Weibull. A statistical distribution function of wide applicability. *Journal of Applied Mechanics*, 18:293–297, 1951.