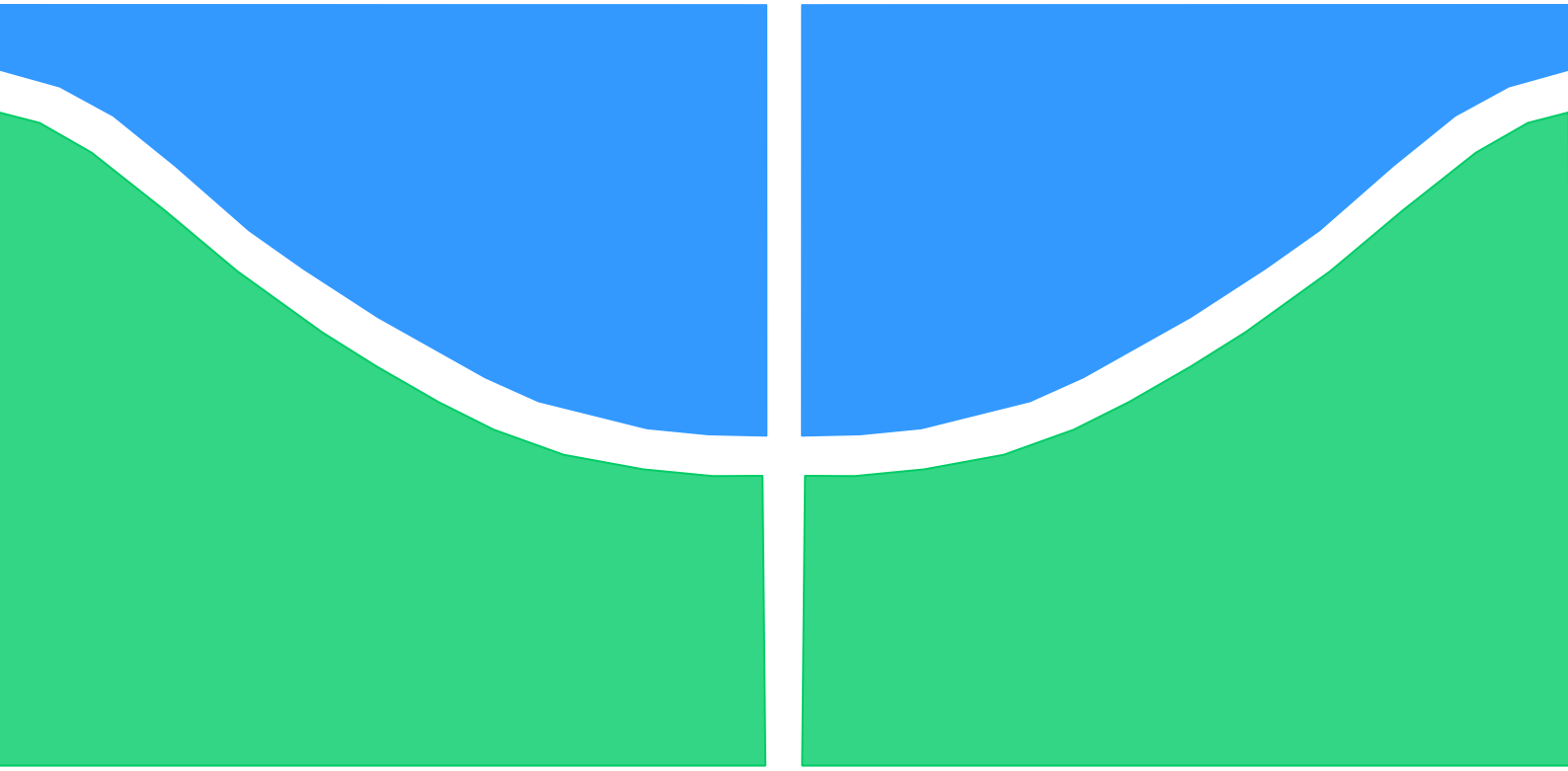

TRABALHO DE GRADUAÇÃO

MODELAGEM DE SISTEMAS DE COMUNICAÇÃO RF COM AUTOENCODER

Gustavo Viana Penido

Brasília, 7 de outubro de 2022

UNIVERSIDADE DE BRASILIA

FACULDADE DE TECNOLOGIA

UNIVERSIDADE DE BRASILIA
Faculdade de Tecnologia

TRABALHO DE GRADUAÇÃO

MODELAGEM DE SISTEMAS DE COMUNICAÇÃO RF COM AUTOENCODER

Gustavo Viana Penido

Relatório submetido como requisito parcial para obtenção
do grau de Engenheiro de Redes de Comunicação

Banca Examinadora

Prof. Paulo Henrique Portela de Carvalho, UnB/ ENE
(Orientador)

Prof. João Paulo Leite, UnB/ ENE

Prof. Leonardo Aguayo, UnB/ ENE

Dedicatórias

Dedico este trabalho à minha família que sempre esteve ao meu lado, me apoiando, motivando e ajudando, pois além de me incentivarem e lutarem para a minha criação, sempre cuidaram para que eu possa ser um profissional, um cidadão, um filho, um ser humano cada vez mais imbuído de seus ensinamentos e valores.

Gustavo Viana Penido

Agradecimentos

Primeiramente, gostaria de agradecer a Deus e minha família, meu pai Advagner, minha mãe Luciana e meu irmão Geovanne, que, dentro de uma formação católica e cristã, pude perceber o quão motivante e inspirador é viver em comunidade, assim compreendi que o sentimento de pertencimento e atitude de sempre fazer o lugar que estamos seja agradável e o melhor possível deixa a vida mais leve e gratificante.

Gostaria também e lembrar de todos que estiveram ao meu lado por todos esses anos, sejam os amigos de escola, irmãos da igreja, e por toda a minha família constituída por avós, tias, tios, primos e os demais agregados que sempre se preocuparam comigo e de alguma forma estiveram presentes na minha formação.

Agradecer também a todas as amizades e experiências em que vivenciei nos anos do curso de graduação. Sair de casa, morar longe da família, ir para uma cidade nova, crescer e ter que ser independente, com certeza foi mais fácil na presença de minhas amigas de longa data Stephanie e Yasmim, que juntos desde o ensino médio estudamos e fomos cursar engenharia em busca da nossa formação profissional e crescimento pessoal.

Como não agradecer à UnB enquanto instituição, pública, federal e gratuita de ensino, que por meio de seus auxílios possibilitou minha permanência na faculdade. Além de sempre falar aos mais novos alunos que 'Universidade é sinônimo de oportunidade', oportunidade de crescimento acadêmico, profissional e pessoal, lugar que fincamos nossos valores e, quem sabe, descobrimos nosso propósito, assim como não destacar a CEU, o Ramo do IEEE da UnB, a EJ, o PIBIC, os projetos de pesquisas pelos quais realizei e tentei sempre sugar o máximo que pude dentro deste universo de oportunidades.

Por fim, quero agradecer a todos os professores (e os técnicos administrativos também!) que tive, sempre atenciosos e todos nunca deixaram de responder uma dúvida sequer minha e sempre acreditaram que eu pudesse chegar mais longe. E em nome de todos quero agradecer imensamente ao meu orientador e professor Dr. Paulo Portela que me mostrou os caminhos da pesquisa acadêmica, do incrível universo das Telecomunicações e que a dedicação vem em primeiro lugar, me tratando sempre com seriedade, cuidado e atenção que sempre desejei em um orientador.

Gustavo Viana Penido

RESUMO

Este trabalho apresenta um estudo da sistematização e da modelagem da técnica de redes neurais chamada de *Autoencoder* aplicada a sistemas de comunicação de radiofrequência fim a fim. Todas as simulações realizadas validam a capacidade de codificação e modulação do *Autoencoder* em comparação com os sistemas convencionais. Assim, são apresentados os resultados obtidos em comparação com sistemas AWGN, Rayleigh, com Codificação, MIMO utilizando arquitetura com *Deep Neural Network* (DNN) e por fim com uma abordagem via *Convolutional Neural Network* (CNN), além de todo o equacionamento para uma justa comparação com os sistemas convencionais de comunicação. Logo, o *Autoencoder* se mostra competitivo em relação aos sistemas convencionais uma vez alcançado o mesmo desempenho.

ABSTRACT

This work presents a modeling systematization of the neural network technique called Autoencoder applied to end-to-end radiofrequency communication systems. The simulations performed validate the encoding and modulation capacity of the Autoencoder in comparison with the traditional communications systems. This is how the results obtained are presented in comparison with AWGN, Rayleigh, systems with Coding, MIMO using architecture with Deep Neural Network (DNN) and finally with an approach via Convolutional Neural Network (CNN), in addition to the whole equation for a fair with the systems conventional communications. Therefore, the Autoencoder is competitive in relation to conventional systems once the same performance is achieved.

SUMÁRIO

1 INTRODUÇÃO 13

1.1 Objetivo do trabalho 14

1.2 Trabalhos relacionados 15

2 FUNDAMENTAÇÃO TEÓRICA 17

2.1 Sistemas de comunicação RF 17

2.1.1 O canal de comunicação 19

2.1.2 Modulação digital e codificação de canal 20

2.1.3 Sistemas MIMO 24

2.2 Redes neurais e o aprendizado de máquina 30

2.3 O *Autoencoder* 37

3 MODELAGEM DOS SISTEMAS DE COMUNICAÇÃO 42

3.1 Da modelagem do sistema 42

3.1.1 Do mapeamento dos sinais mensagens 43

3.1.2 Da energia média de símbolo e as restrições do canal 46

3.2 Da arquitetura da rede neural 50

3.3 Da modelagem para canais com desvanecimento 52

3.4 Da modelagem para sistemas MIMO 53

4 RESULTADOS 55

4.1 Desempenho do *Autoencoder* sob os sistemas tradicionais 57

4.1.1 Análise do *Autoencoder* quanto às restrições e a normalização 57

4.1.2 Considerações quanto à normalização 63

4.1.3 Comparação com sistemas tradicionais 70

4.2 Capacidade de codificação do *Autoencoder* 71

4.2.1 O *Autoencoder* sob canal AWGN 72

4.2.2 O *Autoencoder* sob o canal Rayleigh 78

4.2.3 Comparação com técnicas de codificação de canal 83

4.3 O *Autoencoder* para outras arquiteturas 86

4.3.1 O *Autoencoder* como MIMO 86

4.3.2 O Autoencoder com CNN 93

5 CONCLUSÃO 97

REFERÊNCIAS BIBLIOGRÁFICAS 100

LISTA DE FIGURAS

Figura 2-1 – Diagrama de um sistema de comunicação. 18
Figura 2-2 – Constelação para modulação QPSK. 21
Figura 2-3 – Constelação para modulação 16-QAM. 21
Figura 2-4 – Curvas de BER para as modulações tradicionais 23
Figura 2-5 – Multiplexação espacial o sistema MIMO. [13] 26
Figura 2-6 – Diversidade espacial o sistema MIMO. [13] 27
Figura 2-7 – Configurações e nomenclaturas de um sistema MIMO. [13] 29
Figura 2-8 – Esquema de um neurônio de Perceptron. Adaptado de [1]. 31
Figura 2-9 – Rede neural completamente conectada e com camadas ocultas. Fonte: [1] 34
Figura 2-10 – *Autoencoder* aplicado a imagens. 38
Figura 2-11 – Arquitetura geral de um Autoencoder. [16] 38
Figura 2-12 – Arquitetura de *Autoencoder* na forma de rede neural. 39
Figura 2-13 – Tipos de Autoencoders: (a) Denoising Autoencoder; (b) Contractive Autoencoder e (c) Variational Autoencoder. 40
Figura 3-1 – Sistema de comunicação em um canal AWGN modelado por um *Autoencoder*. [2] 44
Figura 3-2 – Fluxo de camadas da rede neural. 45
Figura 3-3 – Representação do bloco sequencial das L mensagens. 53
Figura 3-4 – Modelo MIMO Open Loop. 54
Figura 3-5 – Modelo MIMO Closed Loop. 54
Figura 4-1 – Constelações para o AE(1,3) - (a) Restrição de Amplitude, (b) Restrição de Energia e (c) Restrição de Potência Media. 64
Figura 4-2 – Constelações para o AE(1,4) - (a) Restrição de Amplitude, (b) Restrição de Energia e (c) Restrição de Potência Media. 64
Figura 4-3 – Constelações utilizando a Restrição de Potência Media - (a) AE(1,6) e (b) AE(1,5). 65
Figura 4-4 – Curva de BLER AE(1,3) em face às normalizações. 66
Figura 4-5 – Curva de BLER AE(1,4) em face às normalizações. 67
Figura 4-6 – Constelações para o AE(1,4) - (a) Combinação de normalização Amplitude e Energia e (b) Combinação de normalização Amplitude e Potência média. 68
Figura 4-7 – Curva de desempenho para o AE(1,4) com combinação de normalizações. 69
Figura 4-8 – Curvas desempenho para diversos *Autoencoders* frente às modulações tradicionais. 71
Figura 4-9 – Curvas de BLER para AE(2,2). 73
Figura 4-10 – Constelação produzida pelo AE(2,2). 73

Figura 4-11 – Curvas de BLER para AE(2,4). 75
Figura 4-12 – Amostra da constelação produzida pelo AE(2,4) distribuída pelos usos do canal. 75
Figura 4-13 – Constelação produzida pelo AE(2,4). 76
Figura 4-14 - Curvas de BLER para AE(8,8). 77
Figura 4-15 – Amostra da constelação produzida pelo AE(8,8) distribuída pelos usos do canal. 78
Figura 4-16 – Desempenho para o AEs com um e dois usos do canal frente às curvas do canal Rayleigh. 79
Figura 4-17 – Constelações para (a) o AE(1,2) e (b) o AE(2,2). 81
Figura 4-18– Constelações para o AE(2,4). 82
Figura 4-19– Constelações para o AE(2,6). 82
Figura 4-20 – Comparação das curvas de desempenho dos AEs com os códigos em Bloco. 85
Figura 4-21 – Comparação das curvas de desempenho dos AEs com os códigos de Hamming. 85
Figura 4-22 – Estrutura do *Autoencoder* para o caso MIMO *Open loop*. 88
Figura 4-23 – Esquema de Codificação Alamouti. 89
Figura 4-24 – Desempenho do sistema MIMO 2x1 com codificação Alamouti em comparação com o AE(2,2). 89
Figura 4-25 – Constelação geradas para M =16 para o AE(2,2). 90
Figura 4-26 – Estrutura do *Autoencoder* para o caso MIMO *Closed loop*. 91
Figura 4-27– Desempenho do sistema MIMO 2x2 com decomposição SVD em comparação com o AE(2,2). 92
Figura 4-28 – Compilação dos resultados para o canal AWGN utilizando-se CNN. 94

LISTA DE TABELAS

Tabela 1-1 – *Benchmarking* e resumo bibliográfico. 15
Tabela 2-1 – Lista de funções de ativação..... 33
Tabela 3-1 – Tabela dos tipos de restrições..... 48
Tabela 3-2 – Arquitetura e configuração do *Autoencoder*. 51
Tabela 3-3 – Parâmetros de Treinamento da rede neural. 52
Tabela 4-1 – Testes para a normalização pela restrição de Energia com M=4. 59
Tabela 4-2 – Testes para a normalização pela restrição de Energia com M=8. 59
Tabela 4-3 – Testes para a normalização pela restrição de Energia com M=16. 60
Tabela 4-4 – Testes para a normalização pela restrição de Potência Média com M=4. 61
Tabela 4-5 – Testes para a normalização pela restrição de Potência Média com M=8. 61
Tabela 4-6 – Testes para a normalização pela restrição de Potência Média com M=16. 62
Tabela 4-7 – Técnicas, códigos e taxas. 84
Tabela 4-8 – Parâmetros do *Autoencoder* para MIMO. [20]..... 86
Tabela 4-9 – Compilação dos resultados obtidos por [22]. 95

LISTA DE SÍMBOLOS

Símbolos Gregos

α	Learning Rate
σ	Desvio padrão do Ruído AWGN
σ^2	Variância do Ruído AWGN
φ	Função de ativação
Ω	Codificação da rede neural (*encoder*)

Siglas

AE	*Autoencoder*
ASK	*Amplitude Shift Keying*
AWGN	*Additive White Gaussian Noise*
BER	*Bit Error Rate*
BLER	*Block Error Rate*
CAE	*Contractive Autoencoder*
CNN	*Convolutional Neural Networks*
CSI	*Channel State Information*
DAE	*Denoising Autoencoder*
DNN	*Deep Neural Networks*
FEC	*Forward Error Correcting*
FSK	*Frequency Shift Keying*
IA	Inteligência Artificial
IEEE	*Institute of Electrical and Electronic Engineers*
LMMSE	*Linear Minimum Mean Square Error*
LSTM	*Long Short-Term Memory*
MIMO	*Multiple-Input Multiple-Output*

MISO	*Multiple-Input Single-Output*
NN	*Neural Network*
OFDM	*Orthogonal Frequency-Division Multiplexing*
PSK	*Phase Shift Keying*
QAM	*Quadrature Amplitude Modulation*
RF	Radiofrequência
RN	Rede Neural
RNN	*Recurrent Neural Network*
RTN	*Radio Transformers Networks*
SIMO	*Single-Input Multiple-Output*
SISO	*Single-Input Single-Output*
SNR	*Signal-Noise Ratio*
STBC	*Space-time Block Coding*
STC	*Space-time Coding*
VAE	*Varational Autoencoder*
ZF	*Zero-Forcing*

1 INTRODUÇÃO

O mundo da Inteligência Artificial (IA) se dá pela busca de máquinas que se baseiem nos processos de tomada de decisão do ser humano para se condicionar à atuação das máquinas. Dentro deste enorme campo de estudo têm-se o aprendizado de máquina, do inglês *machine learning* (ML), que é um conjunto de técnicas que tem como finalidade a automação das ações tomadas pela máquina com base nas experiências apresentadas, ou seja, tais técnicas se firmam no processo de treinamento prévio das situações possíveis para que a IA responda de acordo com as entradas do modelo [1].

Dentre os diversos métodos de ML têm-se a técnica de Redes Neurais (RN) que se inspira no modelo conexionista dos neurônios do cérebro humano para construir modelos e em que se busca definir os pesos destas conexões baseando-se no processo de treinamento [1].

A inteligência artificial e as técnicas de aprendizado de máquina se popularizaram nas mais diversas áreas de pesquisa com o desenvolvimento e a acessibilidade de capacidade computacional. Não demorou muito para que tais temáticas também chegassem ao campo das telecomunicações. Aplicá-las, testá-las e estudá-las em sistemas de comunicação se tornaram um campo de pesquisa forte e vem se consolidando cada vez mais entre estudiosos da área [2].

Para além das técnicas convencionais, baseadas nos modelos estatísticos do canal, técnicas apoiadas em aprendizado de máquina para se determinar a melhor representação dos

símbolos de transmissão se mostram vantajosas ao observar melhores desempenhos em relação os sistemas atuais. Dentre elas, destaca-se a técnica de *Autoencoder*, que visa replicar, a partir de uma nova representação, as entradas na saída [3].

Aponta-se assim o problema da representação junto à estratégia do *Autoencoder*, uma técnica de *machine learning* que se utiliza de redes neurais cascateadas para se montar um único sistema formado por uma rede neural que realiza a codificação (o *encoder*) e outra rede neural para a decodificação (o *decoder*) de uma dada entrada.

Pesquisas vêm mostrando a capacidade do *Autoencoder* de atuar nos sistemas de comunicação [2], [4], [5], [6], [7], [8]. Com o *Autoencoder* têm-se duas redes neurais, uma para atuar como *encoder* e outra como *decoder*. O *encoder* deve encontrar a melhor codificação possível dadas as restrições de potência (inerentes aos sistemas de comunicação) e ao canal de comunicação, este podendo ser modelado como um canal AWGN (Additive White Gaussian Noise) ou Rayleigh, para que o *decoder* possa desfazer a codificação e detectar as mensagens transmitidas com a menor taxa de erro possível.

1.1 Objetivo do trabalho

O objetivo principal deste trabalho é o estudo e a compreensão da sistematização e modelagem de sistemas de comunicação em radiofrequência (RF) sob a perspectiva da técnica de *Autoencoder*, a fim de se mostrar as potencialidades e os principais pontos no que tange as perspectivas de evolução destes sistemas. Assim, como objetivo específico foi examinado o desempenho do *Autoencoder* no canal AWGN, Rayleigh e MIMO sob a modelagem proposta.

1.2 Trabalhos relacionados

Foram selecionados alguns artigos com resultados pertinentes sob o prisma do *Autoencoder* para se analisarem e estudarem as possibilidades de aplicações e dos estudos de casos envolvendo o *Autoencoder* em sistemas de comunicação. Para tanto, a Tabela 1-1 foi produzida a fim de se sumarizar os objetivos, as técnicas utilizadas e os resultados obtidos por cada autor.

Tabela 1-1 – *Benchmarking* e resumo bibliográfico.

Artigo	Objetivo	Simulação e Resultados	Desafios e Conclusões	Rede Neural
[2]	Apresentam novas aplicações de aprendizagem profunda (DL) para a camada física. Utiliza-se do *Autoencoder* para otimizar os componentes de transmissão e recepção.	Simularam *Autoencoders* para sistemas fim a fim de comunicação. *Autoencoders* para múltiplos transmissores e receptores. A técnica de RTN (*Radio Transformer Networks*) para processamento de sinais na detecção e aplicação de CNN (*Convolutional Neural Networks*) para tarefas de classificação da modulação.	Não há um consenso sobre o *dataset* de treino que seja aceito ou comum à toda comunidade. Não há uma representação única dos dados, o tipo de função de custo ou mesmo um valor comum de SNR de treino. A representação de valores complexos na rede neural, a utilização de informações de feedback, CSI e privacidade no modelo de rede neural e a escalabilidade do tamanho das mensagens em função do número de bits e do custo computacional envolvido.	DNN, DNN+RTN e CNN.
[4]	Propuseram um sistema com o *Autoencoder* como	Simularam um sistema com *Autoencoder* fim a fim para sistemas 1×2 SIMO, 2×1 MISO e 2×2 MIMO.	O *Autoencoder* não consegue utilizar a CSI (*Channel State Information*) de forma eficiente porque o receptor recebe a CSI e os símbolos sem distingui-los; e há uma falta de	LSTM+CNN

	uma técnica STC (*Space Time Code*).		compreensão teórica sobre a relação entre a topologia da rede neural e o seu desempenho, fazendo com que a sua estrutura seja inexplicável e imprevisível.	
[5]	Objetivaram estudar em como o conhecimento a priori do canal pode contribuir para melhorar o sistema de comunicação,	Simularam o *Autoencoder* para sistemas Open-loop MIMO; Closed-loop MIMO; MU-MIMO e para *Training Complexity*.	Complexidade do treinamento, em que o AE requer grandes quantidades de amostras de treino e a escalabilidade pela complexidade de se ter mais antenas e usuários. A Taxa de adaptação da arquitetura de rede neural, uma vez que ela é fixa e há a necessidade de adaptação na transmissão.	DNN
[9]	Utilizaram uma abordagem de *Autoencoder* com interferência dinâmica e como ela pode ser aprendida e predita, assim, propuseram um algoritmo para predizer a interferência.	Simularam o *Autoencoder* para um estudo de constelação. Para uma comparar o AE, que é baseado em DL, e as técnicas convencionas de modulação como M-PSK e M-QAM para um único usuário. Comparação destas técnicas convencionais de modulação junto à técnicas de equalização. Simularam o AE também para o caso de dois usuários com interferência de forma simétrica e assimétrica e analisaram o AE para o algoritmo proposta de estimativa de interferência.	Melhorar a eficiência computacional do esquema proposto e a implementação em plataformas reais.	DNN

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo aborda os principais embasamentos teóricos necessários para moldar a proposta de sistematização e modelagem no presente trabalho.

2.1 Sistemas de comunicação RF

O processo de transmissão de mensagens entre dois dispositivos é estruturado pelo chamado sistema de comunicação, representado na Figura 2-1. Este sistema é constituído por módulos, cada um com sua função, como a codificação, a modulação e a conversão Digital-Analógico, entre outros, que atuam efetivamente na composição deste sistema.

Sistemas de comunicação sem fio (também chamados de radiofrequência, RF) utilizam-se das ondas eletromagnéticas no ar como canal de comunicação para transmissão e recepção do sinal mensagem de interesse. Para tanto, esse meio impõe diversos desafios para o sistema, como a atenuação, o desvanecimento e os múltiplos percursos, fazendo com que haja erros na detecção dos símbolos enviados, ou seja, um sinal mensagem no lado do receptor errado.

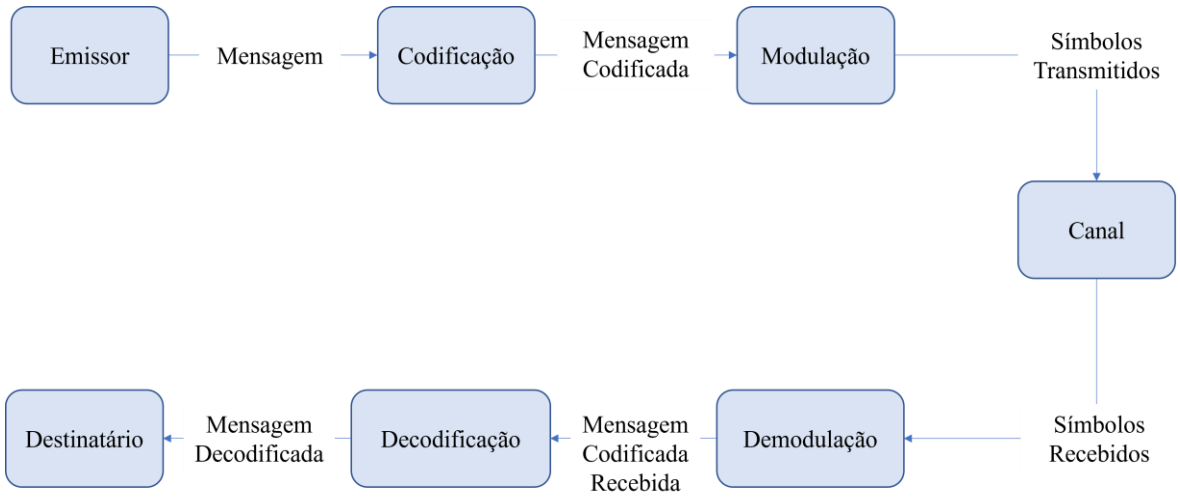

Figura 2-1 – Diagrama de um sistema de comunicação.

Para tratar dessas adversidades intrínsecas ao canal dos módulos, vide Figura 2-1, que compõem o sistema de comunicação, eles podem ser analisados e aperfeiçoados de forma individual. Dentre estes módulos destacam-se os processos de codificação e modulação dos sinais mensagens, uma vez que juntos eles objetivam encontrar a melhor representação dos símbolos a serem transmitidos no canal de comunicação. A partir dessas duas técnicas pode-se traçar curvas de desempenho do sistema, tendo como métrica, por exemplo, a taxa de erro de símbolos (ou bits) em um determinado intervalo de tempo ou para um conjunto de bits transmitidos.

2.1.1 O canal de comunicação

O canal de comunicação sem fio, bem como os componentes de circuitaria, introduzem ruído ao sinal transmitido. Modela-se esse ruído por um processo aleatório que é somado ao sinal de interesse. Em geral, esse ruído pode ser oriundo de diversas fontes, como o ruído térmico produzido pelos amplificadores ou conversores digitais-analógicos presentes nos circuitos de transmissão e recepção [10].

Por ser de natureza aleatória recorre-se ao tratamento probabilístico e estatístico para se modelar o ruído que afeta o sinal transmitido, comumente conhecido como AWGN, do inglês Additive White Gaussian Noise, (Ruído Gaussiano Branco Aditivo), em que é aditivo por se somar ao sinal, branco por possuir potência uniforme em toda a banda de frequência que compõem o sinal e gaussiano por suas amostras obedecerem à uma distribuição Gaussiana de média zero e variância $\sigma^2 = {N_0}/{2}$, uma vez que N_0 é a densidade espectral de potência do ruído para uma banda de frequência em Hz, [10] [11]. Pela Equação 1 têm-se o chamado canal AWGN no domínio do tempo, em que ***y*** é o sinal recebido, ***x*** o sinal transmitido e o ruído ***r***.

$$y = x + r. \tag{1}$$

Outro modelo de canal bastante conhecido é o canal com desvanecimento plano, vide Equação 2. Na situação em que não existe linha de visada entre transmissor e receptor, é modelado por uma distribuição Rayleigh [11]. O desvanecimento, termo ***h*** da Equação 2, é um fator multiplicativo sobre o sinal transmitido que representa a ação dos múltiplos percursos do

ambiente entre transmissor e receptor, sendo de caráter aleatório e obedecendo uma distribuição Gaussiana de média zero e variância unitária [10]. Logo, observa-se que este modelo, o do sinal recebido, é um processo aleatório formado pela variação no tempo do envelope do sinal recebido, resultando em uma variável aleatória com distribuição Rayleigh [11].

$$y = h \cdot x + r. \tag{2}$$

2.1.2 Modulação digital e codificação de canal

Nos sistemas atuais são utilizados esquemas de modulação digital como ASK (*Amplitude Shift Keying*), FSK (*Frequency Shift Keying*), PSK (*Phase Shift Keying*), QAM *(Quadrature Amplitude Modulation*), entre outros. Estes sistemas ditos *M-ários* fazem referência aos $\boldsymbol{M}$ símbolos possíveis que representam um sinal mensagem constituído por k bits, em que $\boldsymbol{M} = 2^k$.

Para um sistema M-QAM qualquer, têm-se uma técnica de modulação que mapeia os sinais mensagens em pontos dentro de um espaço de representações em duas dimensões chamado de Constelação, em que cada ponto é um sinal complexo representado pelos sinais em fase (eixo X) e quadratura (eixo Y), componentes "I" e "Q" nas figuras, respectivamente. Como exemplo, pode-se observar as Figura 2-2 e Figura 2-3, em que temos $\boldsymbol{M}$ = 4 para k = 2 e $\boldsymbol{M}$ = 16 para k = 4, respectivamente.

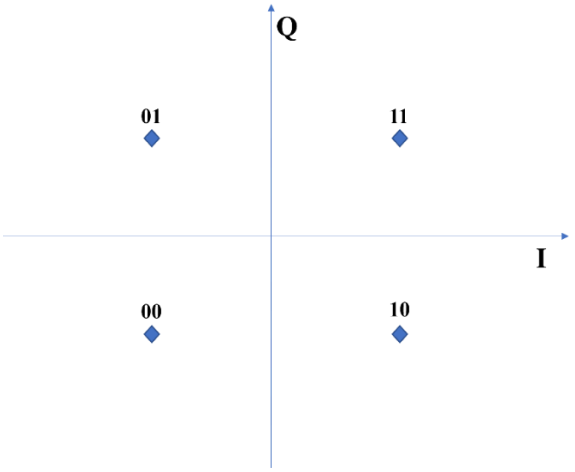

Figura 2-2 – Constelação para modulação QPSK.

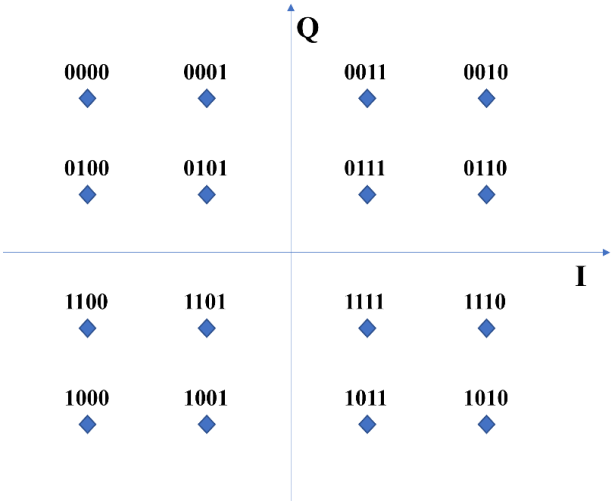

Figura 2-3 – Constelação para modulação 16-QAM.

Da teoria de comunicações digitais [10], quanto maior o $\boldsymbol{M}$, maior é a ordem da modulação. Neste espaço de representação as constelações são formadas por pontos que representam $\boldsymbol{k}$ bits. Mantendo-se inalterada a energia média de símbolos quanto maior o $\boldsymbol{M}$, menor a distância entre os pontos, aumentando-se assim a vulnerabilidade do sinal ao ruído e a interferência. Conhecer o canal de comunicação é extremamente importante para que, por meio da modulação e da codificação, possa-se adaptar o sinal mensagem de tal forma para que se complete o processo de comunicação evitando erros.

A taxa de transmissão, medida em bits por segundo [bps], aumenta de acordo com o k, desde que o tempo para se transmitir uma mensagem fique inalterado com a variação desse parâmetro. Por tanto, quanto maior a ordem da modulação empregada maior é a taxa de transmissão do sistema de comunicação [10].

Dentre os principais objetivos, os sistemas de comunicação são projetados para se obter a menor taxa de erro possível. Assim, quanto menos se errar, mais efetiva será a comunicação. Um indicador bastante utilizado para caracterizar os sistemas de comunicação é a curva de taxa de erro de bit, do inglês *Bit Error Rate* (BER), ou ainda, as curvas de *Block Error Rate* (BLER), que representam a taxa de erro de blocos de bits transmitidos. Tais curvas são apresentadas em um gráfico Mono-Log, em que, no eixo-Y, se tem a taxa de erro para determinada relação sinal-ruído, em decibéis [dB], expressa no eixo-X, como observado na Figura 2-4.

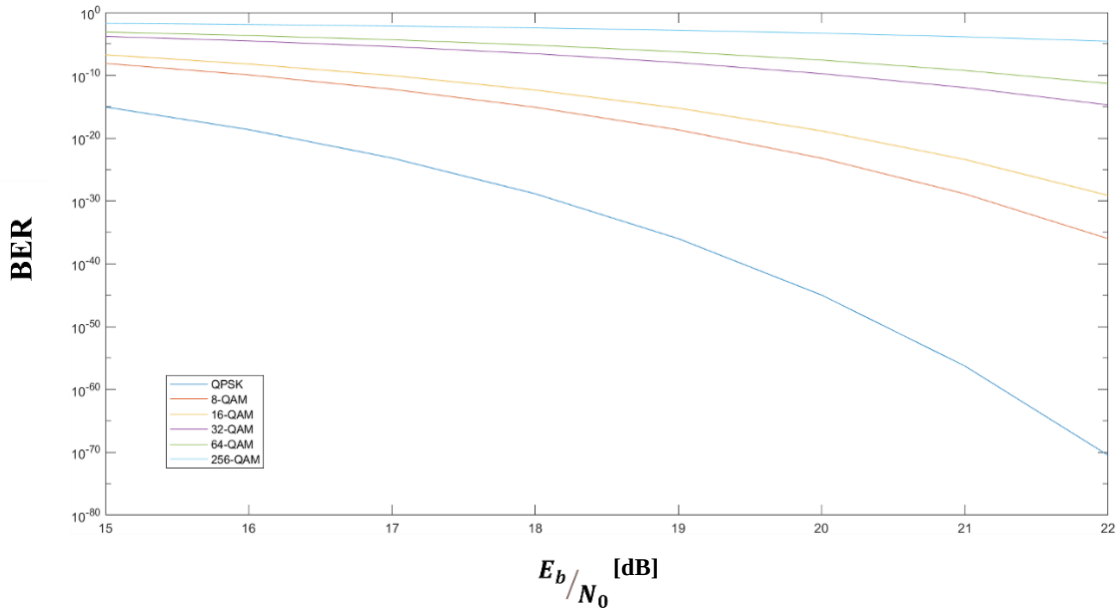

Figura 2-4 – Curvas de BER para as modulações tradicionais.

Das curvas observadas na Figura 2-4 dois pontos chaves merecem ser observados, o primeiro é que, quanto maior é a relação sinal-ruído, menores são as taxas de erros, ou seja, quanto maior a energia do símbolo transmitido e menor a energia do ruído menos se erra no sistema. O segundo ponto é que quanto maior é a ordem de modulação ***M*** maiores são os requisitos de E_b/N_0, ou seja, para uma taxa de erro fixa, maiores são os valores de relação sinal-ruído de acordo com o aumento do ***M***. Levando em consideração requisitos como potência, a relação sinal-ruído do canal, a taxa de transmissão em [bps] desejada e a taxa de erro exigida pode-se determinar o melhor esquema de modulação ***M***-QAM para um determinado sistema de comunicação.

Já a codificação de canal foca em encontrar o melhor código que represente o sinal mensagem, código este que se possa determinar se uma mensagem foi corrompida ou não, e em alguns casos que seja possível até encontrar o erro e poder corrigi-lo. Na prática, quer se determinar a melhor forma de se adicionar redundância ao sinal mensagem [10].

Desde o artigo pioneiro de Shannon [12], há muitas pesquisas na área códigos de correção de erros (*forward error correcting*, FEC), sendo a redundância a chave para alcançar uma comunicação digital livre de erros na presença de distorção, ruído e interferência e a adição apropriada de bits aos dados originais [10].

Além das técnicas envolvidas no processo de construção dos códigos, sejam eles lineares, convolucionais etc. A taxa de codificação, $\boldsymbol{R_c}$, um dos parâmetros que caracteriza tais códigos, sendo esta a taxa entre o número de bits do sinal mensagem original sob o número de bits do sinal mensagem codificado.

Portanto, com a codificação e a modulação o sinal mensagem original é alterado para que ele possa ser transmitido e suporte as adversidades impostas pelo canal. De tal modo, que os blocos de demodulação e decodificação vão desfazer o que foi feito nos blocos anteriores para que se possa completar a comunicação.

2.1.3 Sistemas MIMO

Sistemas de comunicação sem fio exploram uma coleção de técnicas de processamento de sinais oriundas do sistema MIMO (do inglês Multiple-Input Multiple-Output), [13], sistema

esse em que se têm múltiplas antenas de transmissão e recepção nos terminais dos dispositivos de comunicação com o objetivo de melhorar a comunicação, seja na taxa de transmissão ou a qualidade do sinal recebido.

Tais sistemas MIMO também foram desenvolvidos para sinais transmitidos em altas frequências (na ordem de GHz), podendo-se assim ter arranjos com múltiplas antenas, uma vez que o comprimento de onda se encontra na ordem de centímetros ou milímetros, fazendo com que comprimento da antena seja menor e possibilitando que haja múltiplas antenas em dispositivos móveis [14].

Além do comprimento de onda do sinal transmitido, o MIMO explora a multiplexação ao se transmitir dados em múltiplas antenas e da diversidade dos múltiplos percursos na recepção do sinal, ou seja, se há mais de uma antena há mais de um sinal sendo transmitindo (e por consequência maior é a taxa de transmissão), enquanto que ao aumentarmos o número de antenas, transmitindo o mesmo sinal, o sinal recebido no receptor será a soma de um mesmo sinal que percorreu caminhos distintos, que, associado a técnicas de processamento de sinais pode-se aumentar a qualidade do sinal recebido e diminuir a taxa de erro.

De modo que se pode modelar o sistema MIMO pela Equação 3 no domínio do tempo, em que se têm $\boldsymbol{N_t}$ antenas de transmissão no lado do transmissor e $\boldsymbol{N_r}$ antenas de recepção no lado do receptor. Tal que $x \in \mathbb{R}^{N_t}$, o ruído $r \in \mathbb{R}^{N_r}$, o sinal recebido $y \in \mathbb{R}^{N_r}$ e matriz do canal $H \in \mathbb{R}^{N_r \times N_t}$.

$$y = Hx + r.$$

$$H = \begin{bmatrix} h_{11} & h_{12} & \cdots & h_{1N_t} \\ h_{21} & h_{22} & \cdots & h_{2N_t} \\ \vdots & \vdots & \ddots & \vdots \\ h_{N_r 1} & h_{N_r 1} & \cdots & h_{N_r N_t} \end{bmatrix}. \quad (3)$$

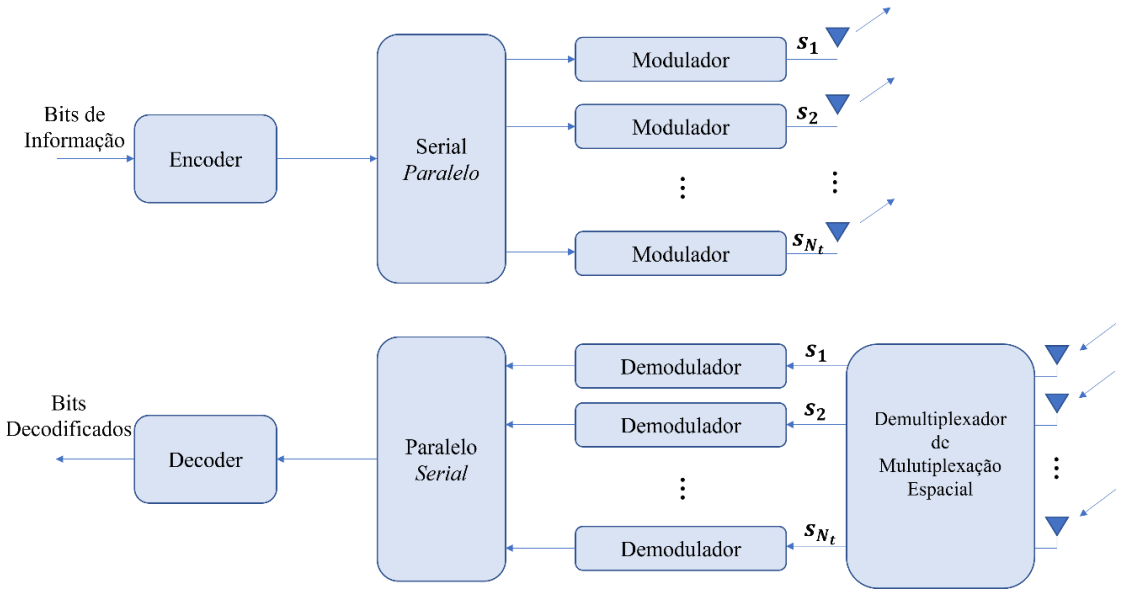

Figura 2-5 – Multiplexação espacial o sistema MIMO [13].

Sistemas MIMO podem trabalhar para **explorar** os múltiplos percursos (*multipath*) e o desvanecimento (*fading*) experienciado nas componentes frequenciais a fim de se melhorar a taxa de transmissão e a eficiência espectral, para isso são utilizadas associadamente técnicas de equalização e demutiplexação espacial no receptor, como o *Zero-Forcing* (ZF) ou *Linear*

Minimum Mean Square Error (LMMSE). Para essa multiplexação espacial o sistema MIMO, Figura 2-5, atribui-se um *ganho de multiplexação* [13].

A depender do canal de comunicação e das alterações sofridas pelo sinal transmitido (o desvanecimento, por exemplo) o sistema MIMO pode combater os múltiplos percursos e o desvanecimento aumentando a disponibilidade do sistema e diminuindo a BER, a taxa de erro de bits na recepção. São utilizadas em conjunto técnicas codificações de tempo e espaço na transmissão, os STC (*Space-Time Code*). Para essa diversidade espacial têm-se o *ganho de diversidade* associado ao sistema, Figura 2-6 [13].

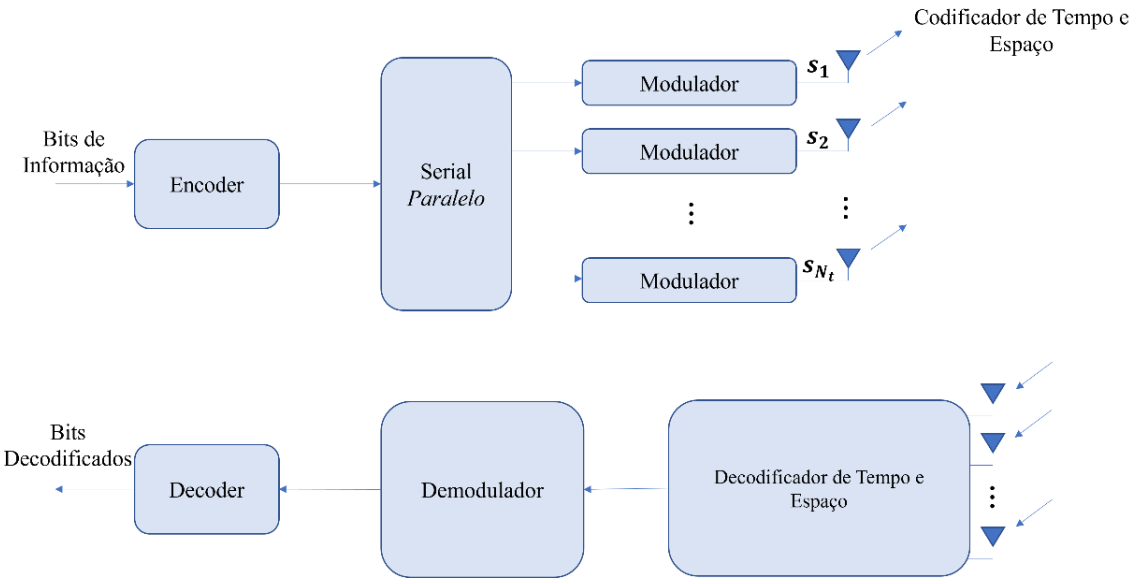

Figura 2-6 – Diversidade espacial o sistema MIMO [13].

Além dos ganhos que o sistema MIMO pode operar, pontua-se a arquitetura dos sistemas na literatura. Razoavelmente, como visto na Figura 2-7, podemos ter sistemas em que o transmissor e receptor possuem apenas uma única antena cada (SISO – *Single Input Single Output*), ou que apenas o transmissor possui múltiplas antenas (MISO – *Multiple Input Single Output*), ou que apenas o receptor possui múltiplas antenas (SIMO – *Single Input Multiple Output*). De maneira geral, as estações rádio base (ERB, ou do inglês *Base Station*, BS), estejam elas em uma única célula de um sistema móvel celular ou mesmo em uma rede Wi-Fi, possuem mais recursos do que os dispositivos móveis, assim, é comum encontrar arquiteturas que o transmissor tenha múltiplas antenas e o receptor tenha apenas uma. No entanto, caso o receptor tenha múltiplas antenas este se torna apto e pode utilizá-las para mitigação da interferência e melhorar a SNR (*signal-noise ratio*). Encontra-se dois casos de usos distintos na literatura [14]:

- **Point-to-point MIMO**: em que a estação transmissora possui múltiplas antenas e se comunica com um único usuário, que possui múltiplas antenas também;

- **Multi-user MIMO**: em que a estação transmissora possui múltiplas antenas e se comunica com múltiplos usuários, em que estes possuem apenas uma ou múltiplas antenas.

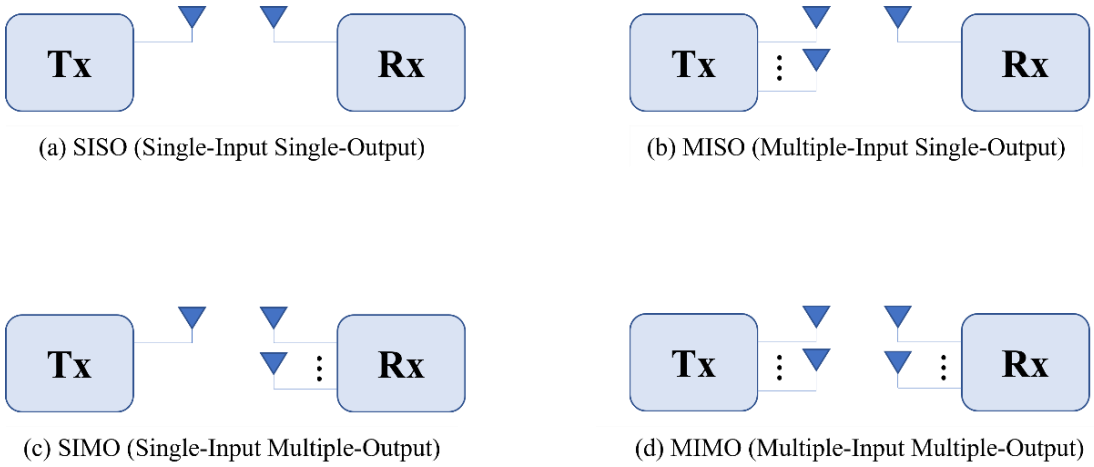

Figura 2-7 – Configurações e nomenclaturas de um sistema MIMO [13].

Pode-se citar a implementação do MIMO-OFDM *(*Multiple-Input, Multiple-Output Orthogonal Frequency-Division Multiplexing*)*, como uma melhoria na transmissão em camada física, em que múltiplas antenas são utilizadas para realizar a comunicação, buscando otimizar a taxa de transmissão e minimizar os erros. Em uma rede Wi-Fi (padrão IEE 802.11n), por exemplo, a técnica MIMO pode ser implementada através de três mecanismos: multiplexação espacial (SM - *Spatial Multiplexing)*, codificação espaço-tempo (STBC - *Space-time Block Coding)* e transmissão *Beamforming.* No mecanismo SM estabelece-se os fluxos espaciais (*spatial streams)* que chegam ao receptor com diferentes intensidades e atrasos, no qual o sinal de saída do transmissor é dividido nestes múltiplos espaços que são transmitidos simultaneamente por diferentes antenas, maximizando as taxas de transmissão. No STBC, utilizando-se dos diferentes fluxos espaciais, o sinal é codificado de maneiras diferentes e transmitido de forma redundante, aumentando a intensidade do sinal e reduzindo a taxa de erros

para a SNR do sistema. Já na transmissão *Beaforming,* é utilizado um arranjo com múltiplas antenas, com objetivo de se transmitir o sinal na direção onde se encontra o receptor, aumentando a taxa de transmissão e melhorando a SNR do receptor [15].

2.2 Redes neurais e o aprendizado de máquina

Dentro do contexto de Inteligência Artificial (IA), encontra-se o chamado aprendizado de máquina (do inglês *machine learning)* ou mesmo o aprendizado profundo (*deep learning)*. Tais técnicas estão sob o domínio da IA, uma vez que se propõem a determinar sistemas computacionais a simularem a inteligência humana na realização de tarefas e na tomada de decisões de forma autônoma.

Estes sistemas são formados a partir de um treinamento com enormes quantidades de exemplos em que vão se aprimorando iterativamente, constituindo-se assim a *aprendizagem* da máquina, e quanto mais exemplos se têm mais *profundo* é esse aprendizado, formando o conhecido *deep learning*. Desse modo, o aprendizado de máquina se tornou cada vez mais útil quanto mais exemplos de treinamentos estão disponíveis, junto com o crescimento da infraestrutura computacional a nível de *hardware* e *software* para tal tarefa, somado ao fato de que o aprendizado de máquina passou a resolver rapidamente aplicações complexas com boa acurácia [16].

Tratando-se especialmente das redes neurais artificiais, estas foram inspiradas e motivadas no cérebro humano, uma vez que o cérebro apresenta um comportamento não linear e paralelo no que diz respeito ao sistema de processamento da informação [1]. De modo que

uma rede neural é um processador distribuído massivamente paralelo constituído por simples unidades de processamento que naturalmente são propensas a armazenarem um conhecimento experimental e torná-lo disponível para uso [1]. Assim, o que chamamos de rede neural é uma das diversas técnicas existentes no mundo do aprendizado de máquina, ela foi inspirada na arquitetura do cérebro humano e nas conexões entre neurônios que ali existem, daí a denotação conexionista [1].

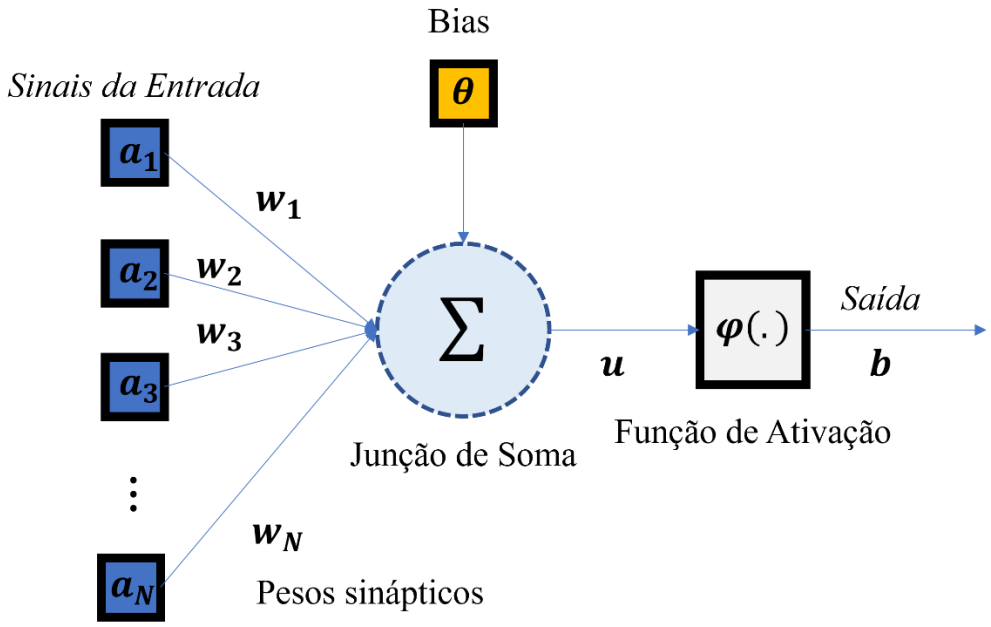

Figura 2-8 – Esquema de um neurônio de Perceptron. Adaptado de [1].

Um neurônio, Figura 2-8, é formado pelo sinal de entrada **a**, em que cada entrada é ponderada por um peso **w**, que são somados para gerar o resultado **u**. A essa soma adiciona-se

um limiar de ativação (da rede Perceptron) ou mais comumente conhecido com *bias*, aqui representado por **θ**, e essa saída passa por uma função de ativação **φ(.)**, para se obter **b**, a saída do neurônio [1].

Assim, chega-se na modelagem matemática de um neurônio:

$$b = \varphi\left(\sum_i w_i a_i + \theta \right). \tag{4}$$

Em que **b,** modelado pela Equação 4, representa a saída do neurônio, **a** a(s) entrada(s) do neurônio que será(ão) somada(s) de acordo com o(s) peso(s) **w** e o *bias* **θ**. A esta soma passa-se por uma função de ativação, que, em geral, na literatura é representada pela função **φ(.)**.

O processo de se somar de forma ponderada as entradas é linear, enquanto o *bias* serve para se ter ajuste fino dessa soma (veja que a depender de seu valor ele leva essa soma a passar ou não pela origem [1]). É a função de ativação que vai adicionar ao sistema a capacidade de se mapear não-linearidades. Em geral, a função de ativação visa limitar entre 0 e 1 ou -1 e 1 a saída do neurônio, uma vez que a soma ponderada das entradas pode tender a mais ou menos infinito, e nisto (nesse processo não-linear) faz com que a saída do neurônio seja não-linear e limitada.

Tabela 2-1 – Lista de funções de ativação.

Função	$\varphi(u_i)$	Intervalo
Linear	u_i	(-∞,∞)
ReLU	$\max(0, u_i)$	[0, ∞)
Tanh	$\tanh(u_i)$	(-1,1)
Sigmoid	$\frac{1}{1+e^{-u_i}}$	(0,1)
Softmax	$\frac{e^{u_i}}{\sum_j e^{u_j}}$	(0,1)

O conjunto de neurônios é chamado de rede neural. Como visto na Figura 2-9, cada círculo representa um neurônio, tal qual visto na Figura 2-8, de modo que neste esquema todas as entradas alimentam todos os neurônios e a cada uma gera-se uma saída. Vale ressaltar que a saída de cada neurônio está na entrada de todos os neurônios da camada seguinte, formando assim uma rede neural completamente conectada (em que todos os nós se conectam, do inglês *fully connected*). De forma análoga, existem estruturas de rede neural parcialmente conectadas. Assim, chega-se no conceito de camadas escondidas (do inglês *hidden layers*). Cada coluna na imagem é chamada de camada e todas as camadas entre os sinais de entrada e saída são ocultas, observa-se que quanto mais camadas internas há, mais complexo se torna a arquitetura da rede neural.

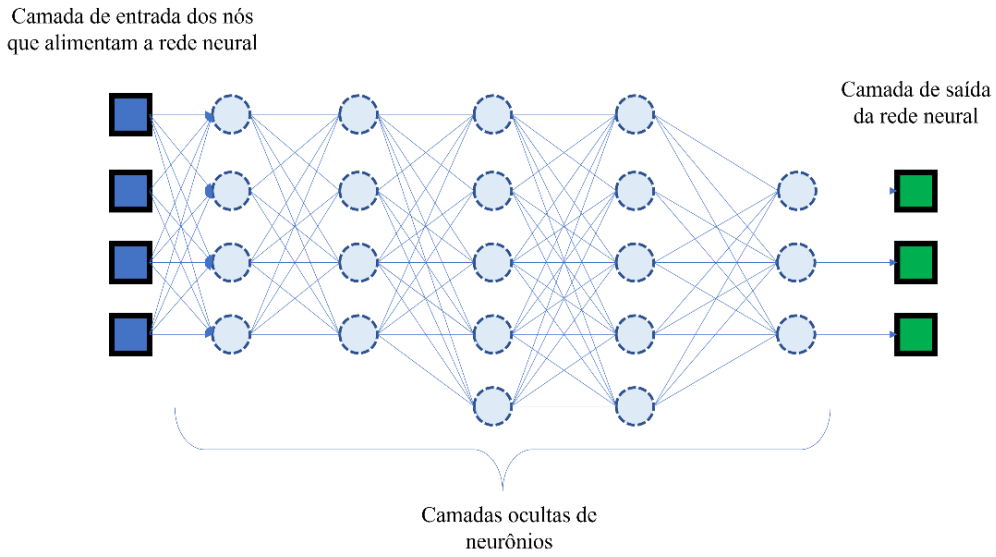

Figura 2-9 – Rede neural completamente conectada e com camadas ocultas. Fonte: [1].

Um ponto importante é que na modelagem de qualquer problema de *machine learning* têm-se, em geral, os dados de entrada **a**, os dados desejados **b** e se quer encontrar (para uma dada função de ativação conhecida) os valores ótimos de **w** e **θ**. Ou seja, temos aí um problema de otimização! A partir de um chute inicial (que é aleatório) quer-se encontrar os melhores valores para os pesos de tal maneira que se possa modelar a saída **b** pelas entradas **a**.

Para problemas em que se conhecem os dados desejados de saída para um conjunto de dados de entrada temos o chamado aprendizado supervisionado, ou seja, pode-se encontrar os valores ótimos dos pesos e do *bias* por meio desses dados. E é neste processo de aprendizado

que chegamos a outro termo chave desta área, o treinamento, processo pelo qual a rede neural irá passar para encontrar tais valores.

O treinamento é feito de forma iterativa em que para cada iteração calcula-se o erro (ou o custo) **J(*w*)** para os valores de **w** e **θ** utilizados. Para isso define-se a função de erro (ou função de custo) em que tal função visa mensurar o quão distante (ou o quão diferente) está a saída estimada $\hat{\boldsymbol{b}}$ da saída deseja **y**. Assim, o objetivo é minimizar a função de erro. Há algoritmos específicos para se fazer a atualização dos pesos **w**, de modo, que se quer o erro (ou custo) zero entre a saída estimada pela rede e a saída desejada em cada iteração do treinamento, esses algoritmos se utilizam bastante da operação de derivação, uma vez dado o problema de minimização, daí encontra-se outra restrição fundamental para a função de ativação, ela precisa ser diferenciável [1].

$$J(w) = \frac{1}{2N}\sum_{i=1}^{N}\left(\hat{b}_i - b_i\right)^2 = \frac{1}{2N}\sum_{i=1}^{N}\left(\varphi\left(\sum_j w_j a_j + \theta\right)_i - b_i\right)^2. \qquad (5)$$

A ideia é minimizar a diferença entre o valor esperado **b** e o valor gerado pela rede $\hat{\boldsymbol{b}}$. Na Equação 5 observamos que **N** é a quantidade de amostras disponíveis para treino e utilizamos como função erro a função MSE (*Mean Square Error*), erro médio quadrático, em que pegamos a média da diferença das saídas ao quadrado. Em geral, este modelo é utilizado para problemas de regressão. Outra função erro bastante utilizada em *machine learning* e na Regressão Logística, para problemas em que se quer classificar os dados de entrada em geral, é a função *Cross-Entropy* (Entropia Cruzada), em que se pega a média do produto da saída pelo

logaritmo da saída menos o produto de um menos a saída e o logaritmo desta diferença, como visto na Equação 6.

$$J(w) = \frac{1}{2N} \sum_{i=1}^{N} \left[-b_i \log(\hat{b}_i) - (1 - b_i) \log(1 - \hat{b}_i) \right]. \quad (6)$$

Uma vez que se quer encontrar os valores ótimos de **w** e **θ** para uma determinada arquitetura de rede neural que modele bem a relação entre os dados de entrada **x** e **y,** o conjunto de dados conhecidos (de **a** e **b**, comumente conhecido de *dataset* de treinamento) no processo de treinamento, que por sua vez, é um processo iterativo que é quantificado por uma função de erro até que a rede convirja, ou seja, chegue a valores aceitáveis de erro para determinados valores de **w** e **θ**. A atualização destes parâmetros se dá pela utilização do algoritmo de retropropagação (do inglês, **backpropagation** [1]) com base no resultado da função de erro escolhida.

Para tanto, utilizamos uma das técnicas de otimização mais conhecida, o *gradient descendent*, em português, a descida do gradiente [17], uma técnica baseada na derivação da função de erro, o gradiente do erro. O gradiente do erro é um vetor com o mesmo tamanho do vetor de pesos em que o j-*ésimo* elemento é definido como a derivada parcial de **J(*w*)** em $\boldsymbol{w_j}$. Com este gradiente, podemos atualizar os pesos ponderados por **α**, uma taxa conhecida como *learning rate*, ou taxa de aprendizagem, parâmetro este que pondera o impacto que o gradiente terá na atualização dos pesos, como visto na Equação 7.

$$\frac{\partial J(w)}{\partial w_j} = \frac{1}{N}\sum_{i=1}^{N}\left(\hat{b}_i - b_i\right)a_i^{(j)}\,.$$
$$w_j = w_j - \alpha\frac{\partial J(w)}{\partial w_j}. \tag{7}$$

2.3 O *Autoencoder*

O *Autoencoder* é uma técnica de *machine learning* que utiliza redes que constitui um sistema formado por uma rede neural que realiza a codificação (o *encoder*) e outra rede neural para a decodificação (o *decoder*) de uma dada entrada [16].

De acordo com [16], o *Autoencoder* é uma rede neural que é treinada para tentar replicar as entradas na saída, que internamente têm-se uma camada escondia denominada ***z*** que codifica os dados de entrada, dando-lhes uma nova representação. Ou seja, o *Autoencoder* (AE) nada mais é do que uma configuração de redes neurais em cascata utilizada para o aprendizado da representação e na identificação de variedades como objetivo de codificar de forma eficiente os dados de entrada em um espaço de representação desejado.

O objetivo do *Autoencoder* é encontrar a melhor representação dos dados de entrada da rede pelo *encoder* e a capacidade de reconstruir dos dados de entrada pelo *decoder* a partir da representação gerada [16]. Um exemplo de utilização do *Autoencoder* é a redução de

dimensionalidade de imagens, ou mesmo na exploração e determinação de *features* (certas características e/ou informações) das imagens.

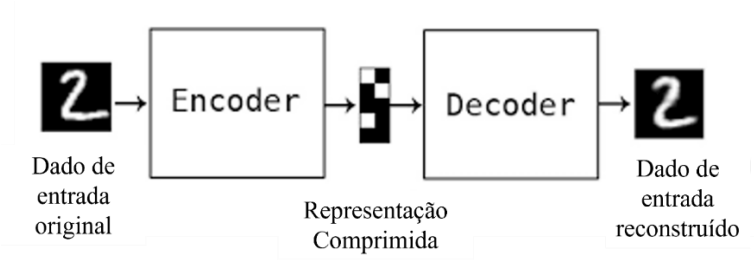

Figura 2-10 – *Autoencoder* aplicado a imagens.

Na Figura 2-10 há um exemplo de utilização do *Autoencoder*. Na entrada têm-se uma imagem que, ao passar pelo *encoder*, gera uma nova representação da imagem. Em seguida, coloca-se essa representação, que foi comprimida, na entrada do *decoder* para que ele reconstrua a imagem original.

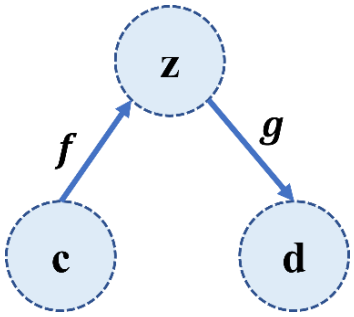

Figura 2-11 – Arquitetura geral de um Autoencoder [16].

A rede neural é constituída de duas partes, o *encoder* que codifica os dados de entrada definido como $\boldsymbol{z = f(c)}$ e o *decoder* que reconstrói os dados originais a partir da representação estabelecida, definido como $\boldsymbol{d = g(z)}$, obtendo-se assim a arquitetura apresentada na Figura 2-11

Assim, para que o *Autoencoder* obtém sucesso, basta aprender $\boldsymbol{g(f(c)) = c}$ para qualquer valor de entrada $\boldsymbol{c}$ treinado. Este modelo de rede neural prioriza as principais características da entrada para gerar a nova representação codificada.

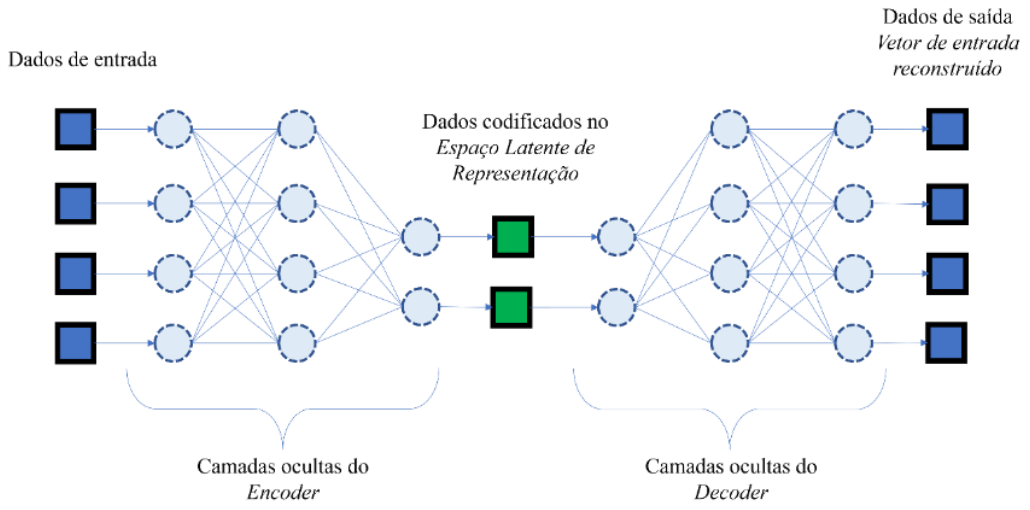

Figura 2-12 – Arquitetura de *Autoencoder* na forma de rede neural.

Chama-se *espaço latente* o espaço de codificação ou espaço de representação gerado pelo *encoder*. Seja $\boldsymbol{M}$ a quantidade de elementos na entrada do *encoder* e $\boldsymbol{n}$ a quantidade de elementos na saída do *encoder*, temos duas situações para os *Autoencoders* [16]:

- Se ***M*** **>** ***n***: têm-se um *Autoencoder subcomplete*, ou seja, a quantidade de elementos do espaço de representação, o espaço latente, é menor do que a quantidade de elementos da entrada.

- Se ***M*** **<** ***n***: têm-se um *Autoencoder overcomplete*, ou seja, a quantidade de elementos do espaço de representação, o espaço latente, é maior do que a quantidade de elementos da entrada.

Na Figura 2-12 pode-se observar uma arquitetura do *Autoencoder*, em que se observa as camadas do *encoder* e *decoder* e os dados codificados (que são resultado da saída do *encoder*) no espaço latente da rede.

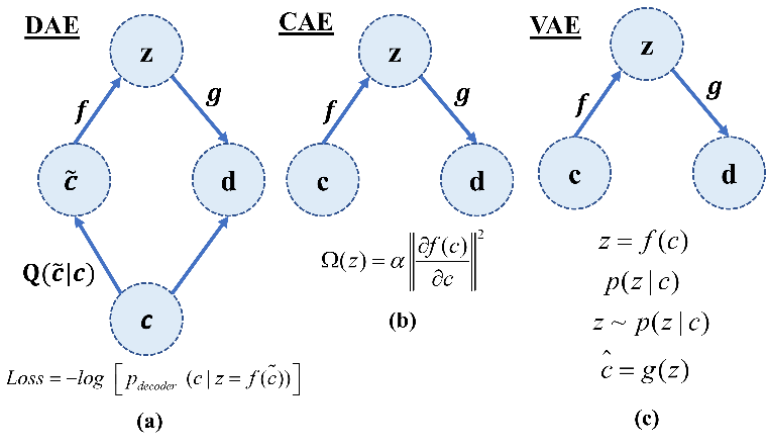

Figura 2-13 – Tipos de Autoencoders: (a) Denoising Autoencoder; (b) Contractive Autoencoder e (c) Variational Autoencoder.

Há diversos tipos de *Autoencoders* em detrimento da sua função ou arquitetura. Pode-se citar o *Denoising Autoencoder* (DAE), [3], em que as entradas $\boldsymbol{c}$, vide Figura 2-13(a), passam por um processo probabilístico, $\boldsymbol{Q(.)}$, antes de se entrar no *encoder*, aplicado a situações em que as entradas já se encontram ruidosas. Já o *Contractive Autoencoder* (CAE), [3], é utilizado para se evitar características indesejadas, Figura 2-13(b), assim adiciona-se uma penalização, $\boldsymbol{\Omega}(\boldsymbol{z})$, à função de custo visando limitar as representações geradas pelo *encoder*. E ainda há, entre outros, o *Variational Autoencoder* (VAE), [18], que com o objetivo de garantir que seu espaço latente tenha boas propriedades, regulariza a sua distribuição de codificações durante o treinamento, além do mais, o termo *variacional* possui uma relação entre a regularização e o método de inferência variacional em estatística, uma vez que um ponto do espaço latente, $\boldsymbol{z}$, após a codificação, é amostrado e é calculado o erro de reconstrução por meio do dado decodificado, Figura 2-13(c).

3 MODELAGEM DOS SISTEMAS DE COMUNICAÇÃO

Este capítulo descreve todas as especificações, funcionalidades e arquiteturas utilizadas para as simulações.

3.1 Da modelagem do sistema

A modelagem consiste na estruturação e adaptação da técnica, o *Autoencoder*, nos moldes dos requisitos referentes aos sistemas de comunicações, ou seja, a parametrização e configuração da rede deve ser compatível para que se possa analisar e comparar os resultados obtidos com os das técnicas atuais. Tal modelo é proposto e exposto pela primeira vez em [2], e deveras adotado, como amplamente utilizado por diversos autores.

Define-se a *quantidade de usos do canal* como $\boldsymbol{n}$. Cada usuário possui um determinado intervalo de tempo em que ele tem para si os recursos do canal disponíveis para que ele possa transmitir. Logo, assumiremos que para cada transmissão o usuário utilizará $\boldsymbol{n}$ vezes o canal para transmitir um dado sinal mensagem.

Logo, para um sistema com $\boldsymbol{k}$ bits há $\boldsymbol{M}$ mensagens possíveis, sendo $\boldsymbol{M} = \boldsymbol{2^k}$, define-se $\boldsymbol{R = k/n}$ como a taxa de bits por usos do canal [2]. Assim, cada mensagem de $\boldsymbol{k}$ bits é mapeada em um vetor complexo $\boldsymbol{x} \in \mathbb{C}^{\boldsymbol{n}}$, em que cada uso do canal é enviado um símbolo. No receptor

aguarda-se a recepção dos $\boldsymbol{n}$ símbolos, que concatenados, serão utilizados para realizar a detecção do sinal mensagem enviado.

O modelo de canal básico é o AWGN (*Additive Whithe Gaussian Noise*), assim, o sinal recebido será modelado da seguinte forma: $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{r}$, em que $\boldsymbol{y}$ é o sinal recebido, $\boldsymbol{x}$ o sinal transmitido e $\boldsymbol{r}$ o ruído que é modelado por uma variável aleatória gaussiana com média zero e variância σ^2.

Para representação da arquitetura do *Autoencoder*, é utilizada a notação **AE(n,k),** em que $\boldsymbol{n}$ é a quantidade de usos do canal para transmissão e $\boldsymbol{k}$ a quantidade de bits do sinal mensagem. Observa-se que para este sistema têm-se $\boldsymbol{M} = \boldsymbol{2^k}$ mensagens possíveis.

3.1.1 Do mapeamento dos sinais mensagens

Para cada mensagem $\boldsymbol{s} \in \boldsymbol{M}$, será codificada a mensagem $\boldsymbol{s}$ em um *one-hot vector*, processo esse também conhecido como *embedding*. A representação *one-hot vector* estabelece que apenas uma posição do vetor (pode ser na coluna ou na linha) seja igual a 1 e o resto igual a 0, em geral faz-se igual a 1 a posição do vetor de $\boldsymbol{M}$ elementos correspondente ao número da mensagem $\boldsymbol{M}$. Assim, para a mensagem de número 6, de representação binária '0110', pode-se mapear em um *one-hot vector* '0000 0000 0010 0000'. Observa-se que para $\boldsymbol{k = 4}$, temos $\boldsymbol{M}$ = **16** possibilidades de mensagens.

A arquitetura da rede neural utilizada segue a Figura 3-1. Tal modelo foi proposto em [2]. Primeiro, mapeia-se em um conjunto de $\boldsymbol{B}$ mensagens, um *batch*, cada uma em sua forma

vetorial ***1xM***, chegando-se em uma matriz ***B***x***M***, em que cada linha represente uma mensagem e as colunas sejam a representação no formato *one-hot* vector.

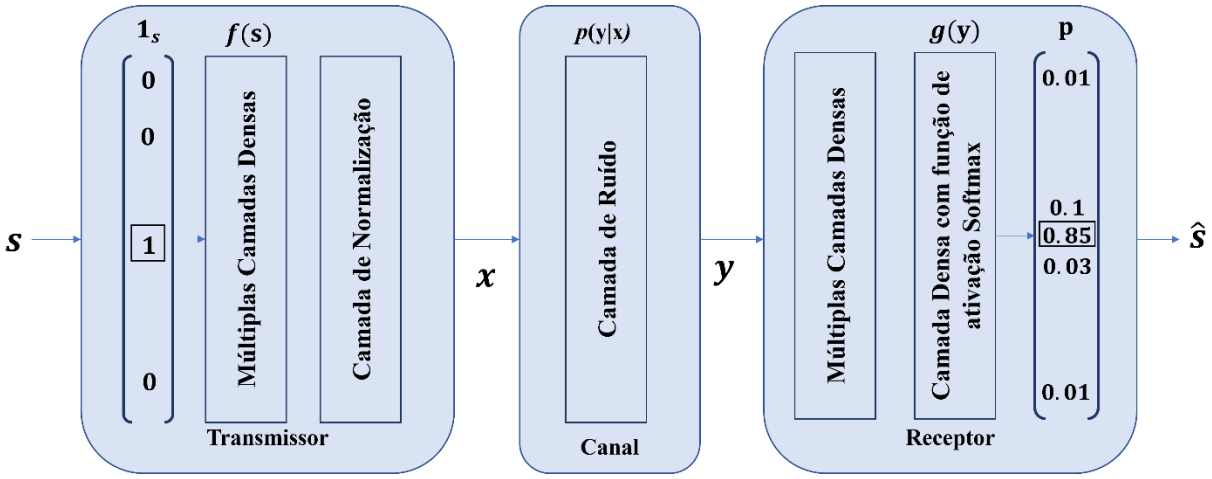

Figura 3-1 – Sistema de comunicação em um canal AWGN modelado por um *Autoencoder* [2].

Cada uma das camadas na arquitetura apresentada na Figura 3-1, e sumarizadas na Figura 3-2, possuem uma funcionalidade. Têm-se como entrada um vetor ***s*** *one-hot* na entrada da rede neural, seguida por uma camada de normalização, de modo que será obtido a representação **x** dos dados de entrada no espaço latente. Assim, é adicionado o ruído com média zero e variância σ^2, veja que esta é uma camada externa ao *Autoencoder*. Logo, há o sinal **y**, que é a soma da saída do *encoder* com o ruído, se tornando a entrada de mais uma camada de rede neural, o *decoder*, para a reconstrução dos dados. Na última camada encontra-se uma função de ativação *softmax*, em que para cada posição do vetor será gerada um valor de probabilidade ***p*** que quantifica a chance de se ser um nesta posição.

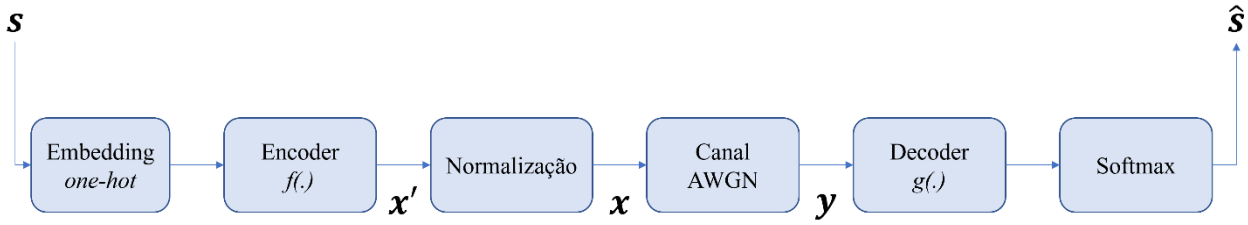

Figura 3-2 – Fluxo de camadas da rede neural.

$$
\begin{aligned}
& s \in M = \{1, 2, \ldots, M\} && M = 2^k \\
& x' = f(s) && f : M \mapsto \mathbb{R}^{2n} \\
& \hat{s} = g(y) && g : \mathbb{R}^{2n} \mapsto M
\end{aligned}
\tag{8}
$$

A camada de entrada é definida pelo *one-hot vector* de tamanho ***M***, passando pelas camadas ocultas do *encoder*, ***f(.)***, e seguida pela normalização em função das restrições de potência e energia do canal de comunicação e do modelo adotado. O canal é representado pelo ruído AWGN que se soma aos símbolos codificados ***x***. Tal resultado passa para o *decoder*, ***g(.)***, para ser decodificado na mensagem de ***M*** símbolos transmitida, em que tais processos são descritos na Equação 9.

Pontua-se que, em um canal físico, os símbolos são transmitidos em quadratura, ou seja, considera-se aqui que para cada uso do canal se transmite um símbolo complexo, por isso temos na saída do *encoder* e na entrada do *decoder* um vetor de dimensão **2*n***, de tal forma que se possa gerar a representação complexa a partir dos símbolos gerados pelo *encoder*.

3.1.2 Da energia média de símbolo e as restrições do canal

Para que possamos fazer uma comparação justa entre os desempenhos dos sistemas de comunicações convencionais com o desempenho observado pelo *Autoencoder* há a necessidade de estabelecer uma relação entre os usos do canal com os requisitos de transmissão, como as restrições oriundas do hardware, do canal e das condições de energia para a transmissão do sinal mensagem.

Dado um sistema equiprovável em que $\boldsymbol{E_s}$ é a energia média de símbolo, $\boldsymbol{E_b}$ é a energia média de bit, com $\boldsymbol{E_s = kE_b}$, os símbolos que representam uma dada mensagem de $\boldsymbol{k}$ bits serão enviados em $\boldsymbol{n}$ usos do canal, logo, a energia total por símbolo, $\boldsymbol{E_T}$, utilizada na transmissão de uma mensagem nos $\boldsymbol{n}$ usos do canal, é definida como a soma de todas as energias dos $\boldsymbol{i}$ símbolos possíveis nos $\boldsymbol{j}$ usos do canal dividido pelos $\boldsymbol{M}$ símbolos possíveis, com $\boldsymbol{i = 1, 2, \dots, M}$ as e $\boldsymbol{j = 1, 2, \dots, n}$, respectivamente, vide Equação 9.

$$E_T = \frac{\sum_{j=1}^{n}\sum_{i=1}^{M} E_{s_{ij}}}{M}. \tag{9}$$

Pela representação geométrica dos sinais mensagens nas constelações e na busca de limitantes para a probabilidade de erro, em especial para os casos em que não há simetria na constelação de um dado mapeamento entre mensagens e símbolos de transmissão de um sistema de comunicação [10], têm-se que a distância ao quadrado de cada símbolo na constelação é igual à energia de uma determinada mensagem $\boldsymbol{s_i}$ de $\boldsymbol{k}$ bits, assim $E_{s_i} = d_{ij}^2$. Portanto, para a i-*ésima* mensagem $\boldsymbol{s_i} \in \boldsymbol{M}$ mapeada no vetor de símbolos $\boldsymbol{x_i} = [\boldsymbol{x_{i1}}, \boldsymbol{x_{i2}}, \ldots, \boldsymbol{x_{ij}}, \ldots, \boldsymbol{x_{in}}]$ nos j-*ésimos* usos do canal, define-se como energia média utilizada na transmissão da a i-*ésima* mensagem $\boldsymbol{s_i}$ como nos $\boldsymbol{n}$ usos do canal como: $E_{s_i} = {\sum_{j=1}^{n} |x_{ij}|^2}/{n} = {\sum_{j=1}^{n} d_{ij}^2}/{n}$, de modo que, pode-se definir a energia média de símbolos nos j-*ésimos* usos do canal como: $E_{s_j} = {\sum_{i=1}^{M} |x_{ij}|^2}/{M} = {\sum_{i=1}^{M} d_{ij}^2}/{M}$. Consequentemente, chega-se na Equação 10, uma expressão que represente a energia média de símbolo.

$$E_s = E_T = \frac{\sum_{j=1}^{n}\sum_{i=1}^{M} E_{s_{ij}}}{M} = \frac{\sum_{j=1}^{n}\sum_{i=1}^{M} d_{ij}^2}{M}. \tag{10}$$

Como $\boldsymbol{\sigma^2 = N_0/2}$, o desvio padrão é igual a raiz de $\boldsymbol{N_0/2}$, $\boldsymbol{E_s}$ a energia média de símbolo com $\boldsymbol{E_s = kE_b}$, e $\boldsymbol{E_b}$ sendo a energia média de bit, têm-se a probabilidade de erro para um valor de SNR em dB. Assim, para se manter justas as comparações entre os sistemas de comunicação precisam-se adequar o ruído em função da SNR desejada e dá na energia média de símbolo, de modo que é necessário fazer uma restrição quanto a energia média de símbolo, logo têm-se que

$\boldsymbol{E_T = E_s \leq n}$ como restrição de energia. Seja *snr* o valor adimensional de SNR, chega-se à variância do ruído AWGN necessária para se fazer comparações justas, vide Equação 11.

$$N_0 = \frac{E_b}{snr} = \frac{E_s/k}{snr} = \frac{E_T/k}{snr} = \frac{n/k}{snr}.$$
$$\sigma^2 = \frac{N_0}{2} = \frac{n}{2 \cdot k \cdot snr} \rightarrow \sigma = \sqrt{\frac{1}{2 \cdot k/n \cdot snr}} = \sqrt{\frac{1}{2 \cdot R \cdot snr}}. \quad (11)$$

Como observado na Tabela 3-1 pode-se aplicar algum tipo de restrição aos dados de saída do *encoder* a fim de se expressar as condições físicas da transmissão. Essas restrições são aplicadas na camada de normalização da rede neural apresentada na Figura 3-1

Tabela 3-1 – Tabela dos tipos de restrições.

Tipos de Restrições	
Restrição de Amplitude	$\lvert x_i \rvert \leq 1 \ \ \forall i$
Restrição de Energia	$\lVert x \rVert_2^2 \leq n$
Restrição de Potência Média	$\mathrm{E}\left[\lvert x_i \rvert^2 \right] \leq 1$

Por exemplo, para uma potência média de transmissão P_T, em que $P_T = \frac{E_S}{n}$, e energia média de símbolo $E_s = E[|x'|^2]$, têm-se a seguinte normalização, Equação 12, em decorrência da restrição de potência média:

$$\frac{\mathrm{E}\left[\left|x'_i\right|^2\right]}{nP_T} \leq 1.$$

$$x = \frac{x'}{\sqrt{\frac{\mathrm{E}\left[\left|x'_i\right|^2\right]}{nP_T}}} = \frac{x'\sqrt{nP_T}}{\sqrt{\mathrm{E}\left[\left|x'_i\right|^2\right]}}. \tag{12}$$

Com relação aos tipos de normalização aplicada à rede neural em função da restrição escolhida, define-se a partir de agora a seguinte nomenclatura utilizada neste trabalho quanto as normalizações e restrições:

a) ***Normalização pela restrição de amplitude***: emprega-se uma normalização sob os símbolos gerados pelo *encoder* a fim de respeitar a restrição de amplitude;

b) ***Normalização pela restrição de energia***: emprega-se uma normalização sob os símbolos gerados pelo *encoder* a fim de respeitar a restrição de energia;

c) ***Normalização pela restrição de potência média***: emprega-se uma normalização sob os símbolos gerados pelo *encoder* a fim de respeitar a restrição de potência média.

3.2 Da arquitetura da rede neural

Assim como qualquer técnica de ML, o *Autoencoder* e as suas redes neurais se caracterizam a partir do tipo de rede neural, da arquitetura utilizada, do número de camadas ocultas, quantidade de neurônios, algoritmo de otimização e função de custo, além de parâmetros como quantidade de épocas de treinamento, quantidade de amostras de treino e taxa de aprendizado (*learning rate*).

Entre os diversos tipos de redes neurais, de arquiteturas e de configurações, optou-se por se trabalhar com redes neurais profundas e completamente conectadas, as chamadas DNN (*Deep Neural Networks*) ou mesmo MLP (*Multilayer Perceptrons*). Toda a arquitetura e configuração foram inspiradas a partir de [2], [5], [6], [7], [9], em que pode-se encontrar aplicações do *Autoencoder* constituído de redes neurais lineares.

A arquitetura da rede neural segue de acordo com a Tabela 3-2, formada por três camadas ocultas, as duas primeiras são compostas pela função de ativação ReLU (*Rectified Linear Unit*), função que retorna o valor da entrada se ele for positivo e retorna zero se a entrada for negativa. Na última camada a função de ativação é a Linear, ou seja, ela replica o resultado da operação de soma ponderada dos neurônios na saída da rede, seguida pela camada normalização.

Uma vez com a representação dos dados pelo *encoder* já estejam normalizados, estes seguem para a operação do canal, neste caso, a de soma com ruído AWGN. Estes dados corrompidos são a entrada da segunda rede neural, o *decoder*, que por sua vez também é

constituído por três camadas, as duas primeiras com função de ativação ReLu e a última com a função Softmax.

Tabela 3-2 – Arquitetura e configuração do *Autoencoder*.

Autoencoder	**Camada + Função de Ativação**	**Dimensão de Saída**
Encoder	Entrada	***M***
	Densa + ReLU	512
	Densa + Linear	30
	Densa + Linear	**2*n***
	Normalização	**2*n***
Canal com ruído	Ruído AWGN	**2*n***
Decoder	Densa + ReLU	512
	Densa + ReLU	30
	Densa + Softmax	***M***

No que tange à construção e estruturação do ambiente de simulação, a rede neural foi construída em Pytorch [19], um *framework* de código aberto para aprendizado de máquina baseada na biblioteca Torch. Quanto aos parâmetros de treinamento, foi utilizado o otimizador Adam [20] para atualização dos pesos e para a geração do ruído AWGN de treinamento foi usado 3 dB como relação sinal-ruído, assim como os demais parâmetros apresentados na Tabela 3-3.

Tabela 3-3 – Parâmetros de Treinamento da rede neural.

Parâmetros de Treinamento

Learning Rate	0.001
Batch Size	$10\boldsymbol{M}$
Épocas	2000

3.3 Da modelagem para canais com desvanecimento

A modelagem para o canal Rayleigh segue as mesmas configurações e parametrizações adotadas anteriormente para o canal AWGN. No que se refere à simulação do canal, a cada época de treinamento será gerado um valor aleatório de distribuição Gaussiana de média zero e variância unitária para representar o fator de desvanecimento.

A estrutura da rede neural se mantém igual a apresentada na Tabela 3-2, exceto pela entrada do *decoder* que passa a incluir informações do canal de comunicação. Assim, os valores gerados para o canal serão concatenados ao vetor de saída do canal. Tal configuração representa os sistemas convencionais de comunicação que se utiliza de técnicas de estimação e equalização do canal, logo, para esta modelagem adota-se uma perfeita estimação do canal.

3.4 Da modelagem para sistemas MIMO

Nos sistemas MIMO, para se transmitir as $\boldsymbol{M}$ mensagens possíveis, faz-se o uso de $\boldsymbol{N_t}$ antenas de transmissão e $\boldsymbol{N_r}$ antenas de recepção para se transmitir um bloco $\boldsymbol{L}$ de mensagens, tal que cada mensagem $\boldsymbol{s} \in \boldsymbol{M^L}$, como visto na Figura 3-3 [21].

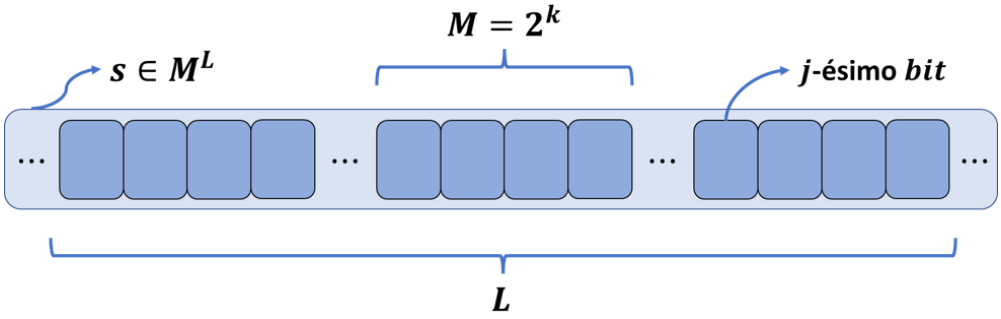

Figura 3-3 – Representação do bloco sequencial das L mensagens.

Para os sistemas MIMO convencionais a estimação do canal é necessária na aplicação das técnicas de processamento de sinais nos processos de detecção e codificação, tal estimação é condensada no chamado *Channel State Information* (CSI), [13]. Logo, a modelagem utilizada irá se concatenar com a matriz do canal utilizada, assim temos dois cenários possíveis:

- **Open Loop:** quando somente o receptor possui informações do canal, vide Figura 3-4;

- **Closed Loop:** quando receptor e transmissor possuem informações do canal, representado na Figura 3-5.

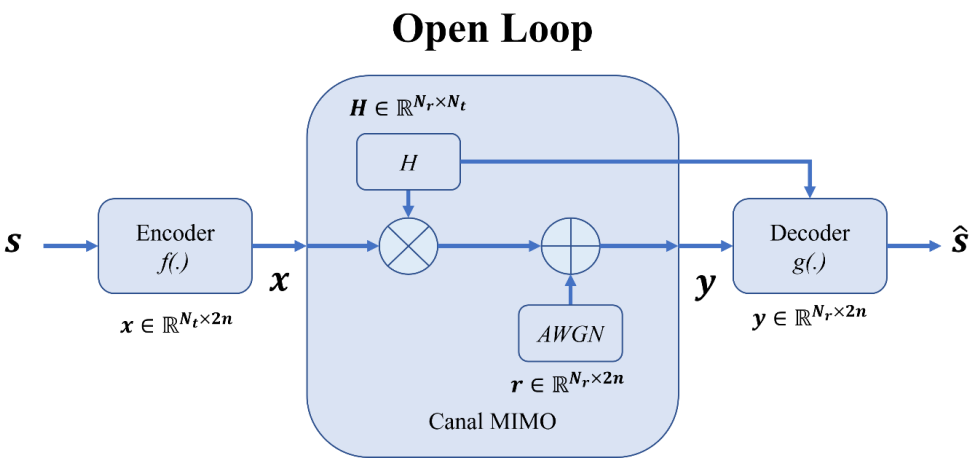

Figura 3-4 – Modelo MIMO Open Loop.

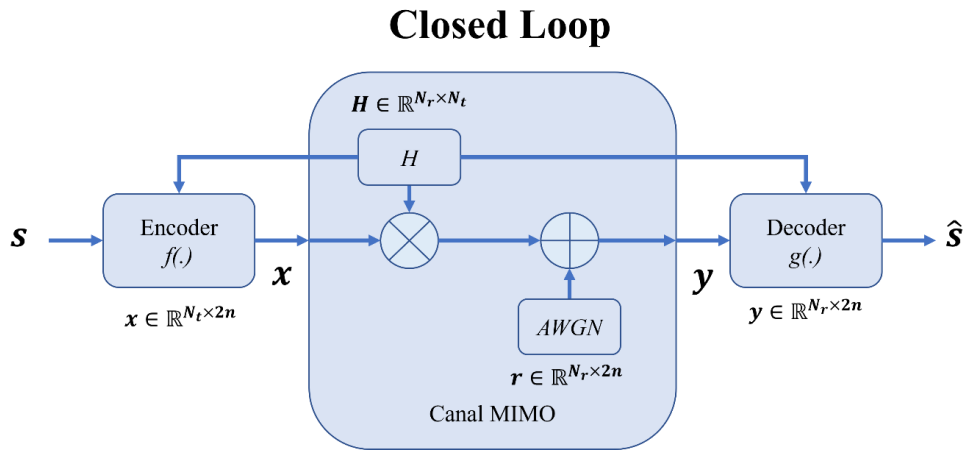

Figura 3-5 – Modelo MIMO Closed Loop.

4 RESULTADOS

Este capítulo apresenta os resultados das simulações, bem como busca traçar o perfil de comportamento do Autoencoder através dos testes e simulações realizados.

Primeiramente, para a modelagem adotada, gerou-se uma saída no espaço de $\mathbb{R}^{2n}$, em contrapartida de [2] que gerou no $\mathbb{R}^{n}$. Adicionalmente optou-se por visualizar as constelações em cada uso do canal, assim, para uma dada mensagem, a representação gerada pelo *encoder* foi levada para o espaço complexo $\mathbb{C}^{n}$ e distribuiu-se os símbolos de acordo com os $\boldsymbol{n}$ usos do canal.

Reforça-se que, de forma a se replicar os resultados presentes na literatura e estudar o comportamento do *Autoencoder*, consolidou-se um ambiente de simulação na linguagem de programação Python, utilizando a *framework* Pytorch para a construção das redes neurais para os seguintes resultados. Observa-se também que foi utilizado a normalização de potência média.

Para analisarmos a capacidade do *Autoencoder* de observar a curva de BLER, em que para uma dada relação sinal-ruído pega-se a média da taxa de erro do *decoder*, ou seja, para um bloco de mensagens enviadas calcula-se a taxa de quantas foram detectadas erradas sobre o total. Similarmente, para apreciação, plota-se a constelação gerada pelo *encoder* a fim de se entender a representação dos sinais mensagens geradas pela rede neural.

A partir destas duas ferramentas de análise gerou-se os resultados para diversos esquemas de *Autoencoder* e para todos os resultados plotou-se as curvas de BER das modulações tradicionais no intuito de se balizar e observar os resultados obtidos frente aos sistemas convencionais.

A organização deste capítulo se dá pela apresentação dos resultados por blocos de simulações realizadas. No primeiro momento foi realizado um conjunto de simulações que fosse possível observar o comportamento do *Autoencoder* em relação às diferentes normalizações possíveis. Em seguida, para a modelagem e a configuração propostas no capítulo anterior, gerou-se as curvas de desempenho para diversos esquemas com o intuito de se comparar com o desempenho dos sistemas tradicionais.

Observando os resultados sob a perspectiva dos modelos tradicionais e das normalizações, focou-se em se mostrar a capacidade de codificação do *Autoencoder* nos ***n*** usos do canal perante os canais AWGN e Rayleigh, e se comparou os resultados obtidos com algumas técnicas tradicionais de codificação sob o olhar das taxas de codificação e de bits por uso do canal.

Finalmente, têm-se os resultados para a modelagem MIMO, para os casos *open loop* e *closed loop*. Finaliza-se com os resultados obtidos para um sistema SISO com uma arquitetura de *Autoencoder* proposta [22], utilizando-se redes neurais convolucionais (CNN).

4.1 Desempenho do *Autoencoder* sob os sistemas tradicionais

O primeiro entendimento, a nível de sistematização e comparação, sobre o *Autoencoder* é analisar se ele é viável e competitivo frente aos modelos tradicionais. Para tanto gerou-se uma série de simulações para se verificar tal desempenho.

Os parâmetros de treinamento e configuração mantiveram-se inalterados, no entanto o principal ponto é que nesta fase do estudo optou-se por sistemas com apenas um uso do canal, ou seja, $\boldsymbol{n = 1}$ para todos os casos a fim de estabelecer a mesma base de comparação com os sistemas atuais.

Contudo, na primeira subseção será feia uma verificação sobre as restrições de energia e potência média, sendo que o objetivo é se testar se de fato a normalização adota está sendo respeitada mediante a modelagem abordada.

4.1.1 Análise do *Autoencoder* quanto às restrições e a normalização

Como abordado no capítulo anterior, neste ponto do trabalho será analisado e testado se os símbolos gerados pelo *Autoencoder*, em função das restrições de energia e potência média, estão de acordo com a normalização aplicada e os resultados observados corroboram com a modelagem aplicada. Ressalta-se a importância desta etapa, uma vez que validado as restrições adotadas garante-se as comparações de desempenho que se seguirão ao longo deste capítulo.

Como já definido, para a i-*ésima* mensagem $\boldsymbol{s_i} \in \boldsymbol{M}$ mapeada no vetor de símbolos $\boldsymbol{x_i} = [\boldsymbol{x_{i1}}, \boldsymbol{x_{i2}}, \ldots, \boldsymbol{x_{ij}}, \ldots, \boldsymbol{x_{in}}]$ nos j-*ésimos* usos do canal, pode-se calcular E_{s_i}, a energia média utilizada na transmissão da a i-*ésima* mensagem $\boldsymbol{s_i}$ e E_{s_j}, sendo a energia média de símbolos nos j-*ésimos* usos do canal.

Para observação dos dados das Tabela 4-1 até Tabela 4-6 são compilados das energias médias de símbolos, seja por uso do canal ou energia média em todos os usos do canal para constelações com $\boldsymbol{M}$ igual a 4, 8 e 16 símbolos para 1, 2 e 3 usos do canal. Salienta-se que, para além da distribuição de energia nos símbolos na constelação gerada pelo *Autoencoder*, os dados analisados estão destacados nas seguintes tabelas.

Para a *normalização pela restrição de energia* espera-se que, pela definição de energia média de símbolo e pela restrição de energia, $\|\boldsymbol{x}\|_2^2 \leq \boldsymbol{n}$, que a energia média dos símbolos, a partir da constelação gerada pela rede neural, seja numericamente igual aos usos do canal. Assim nas Tabela 4-1, Tabela 4-2 e Tabela 4-3 observa-se que os símbolos gerados pelo AE obedecem à restrição imposta por meio da normalização aplicada. Um fato interessante é que para um uso do canal o AE distribui de forma uniforme a energia entre os símbolos, gerando, para este cenário uma constelação semelhante à de uma modulação PSK.

Tabela 4-1 – Testes para a normalização pela restrição de Energia com M=4.

Normalização pela restrição de Energia – M=4

Símbolos	n=1	n=2			n=3			
	1	1	2	ΣE_{s_i}	1	2	3	ΣE_{s_i}
E_{s_1}	1	1,3919	0,2502	1,6421	11.940	0,9944	0,7652	2,9536
E_{s_2}	1	1,0363	0,9623	1,9986	12.128	0,7833	0,9568	2,953
E_{s_3}	1	1,0379	0,9606	1,9985	0.6929	1,3043	0,9048	2,902
E_{s_4}	1	0,5569	1,2999	1,8569	0.8875	0,6447	1,3404	2,8726
ΣE_{s_j}	4	4,0231	3,473	7,961	3,9872	3,7268	3,9672	11,6812
E_{s_j}	1	1,0058	0,8682	-	0.9968	0,9317	0,9918	-
E_s	1	1,0058	-	1,874	1	-	-	2,9203

Tabela 4-2 – Testes para a normalização pela restrição de Energia com M=8.

Normalização pela restrição de Energia – M=8

Símbolos	n=1	n=2			n=3			
	1	1	2	ΣE_{s_i}	1	2	3	ΣE_{s_i}
E_{s_1}	1	0,9015	1,0896	1,9911	0,6404	1,0709	1,2013	2,9126
E_{s_2}	1	1,0127	0,9871	1,9998	1,2824	0,467	1,0664	2,8159
E_{s_3}	1	0,9752	1,0242	1,9994	1,1864	0,9884	0,7845	2,9594
E_{s_4}	1	1,0764	0,9173	1,9937	1,0753	1,1025	0,7926	2,9704
E_{s_5}	1	0,8663	1,1178	1,9841	0,5737	1,1826	1,128	2,8843
E_{s_6}	1	0,9809	1,0187	1,9996	0,9271	1,1544	0,8988	2,9803
E_{s_7}	1	1,0664	0,9289	1,9953	0,8696	1,2359	0,8463	2,9518
E_{s_8}	1	1,0737	0,9204	1,9941	1,1133	0,9228	0,9533	2,9895
ΣE_{s_j}	8	7,9532	8,004	15,9572	7,6683	8,1246	7,6712	23,4641
E_{s_j}	1	0,9942	1,0005	-	0,9585	1,0156	0,9589	-
E_s	1	-	-	1,9946	-	-	-	2,933

Tabela 4-3 – Testes para a normalização pela restrição de Energia com M=16.

Normalização pela restrição de Energia – M=16

Símbolos	n=1	n=2			n=3			
	1	1	2	ΣE_{s_i}	1	2	3	ΣE_{s_i}
E_{s_1}	1	1,3961	0,2258	1,6218	0,294	0,9626	1,4096	2,6661
E_{s_2}	1	0,8341	1,142	1,9762	1,0453	0,8273	1,1059	2,9785
E_{s_3}	1	1,1314	0,8486	1,9799	0,5821	1,2631	1,0324	2,8776
E_{s_4}	1	1,1934	0,7588	1,9522	1,2439	1,1332	0,4107	2,7878
E_{s_5}	1	0,4329	1,3463	1,7792	1	1,0977	0,8916	2,9893
E_{s_6}	1	0,9768	1,0227	1,9995	0,8076	1,1723	0,9866	2,9665
E_{s_7}	1	0,4694	1,3341	1,8034	1,0804	0,6239	1,2015	2,9057
E_{s_8}	1	0,9621	1,0365	1,9986	1,2266	0,3988	1,1561	2,7814
E_{s_9}	1	1,3757	0,3277	1,7034	0,7432	0,9757	1,223	2,9419
$E_{s_{10}}$	1	0,7869	1,175	1,962	1,256	0,6091	1,0254	2,8905
$E_{s_{11}}$	1	0,7004	1,2286	1,929	0,9435	1,3379	0,5656	2,8469
$E_{s_{12}}$	1	1,018	0,9817	1,9997	0,2613	1,2194	1,2019	2,6827
$E_{s_{13}}$	1	1,1108	0,8753	1,9861	1,4586	0,2196	0,9079	2,5861
$E_{s_{14}}$	1	1,1206	0,8627	1,9833	1,1655	0,9574	0,8514	2,9744
$E_{s_{15}}$	1	0,3752	1,3635	1,7388	0,7457	1,0224	1,1826	2,9507
$E_{s_{16}}$	1	1,3084	0,5368	1,8452	1,1665	1,1393	0,5843	2,89
ΣE_{s_j}	16	15,1921	15,0661	30,2582	15,02	14,9597	15,7365	45,7162
E_{s_j}	1	0,9495	0,9416	-	0,9387	0,935	0,9835	-
E_s	1	-	-	1,8911	-	-	-	2,8573

Já nas Tabela 4-4, Tabela 4-5 e Tabela 4-6 temos os dados para a normalização pela restrição de potência média, em que, com a restrição de $\boldsymbol{E[|x|^2] \leq 1,}$ a potência total utilizada na transmissão nos $\boldsymbol{n}$ usos do canal deve respeitar a $\boldsymbol{P \leq nM,}$. Logo, observa-se que a energia

é distribuída pelo AE entre os ***M*** símbolos nos ***n*** usos do canal de forma a não estabelecer um padrão, excetuando-se para o caso de 1 uso do canal, pode-se observar que há diferença na distribuição das energias entre os símbolos, no entanto, pelos valores obtidos percebe-se uma simetria na posição dos símbolos na constelação em função dos valores de suas energias.

Tabela 4-4 – Testes para a normalização pela restrição de Potência Média com M=4.

Normalização pela restrição de Potência Média – M=4

Símbolos	n=1	n=2			n=3			
	1	1	2	ΣE_{s_i}	1	2	3	ΣE_{s_i}
E_{s_1}	1	1,0496	1,0182	2,0678	1,0482	0,8558	1,0151	2,919
E_{s_2}	0,9574	0,9094	0,9979	1,9073	1,0506	1,0817	0,8663	2,9985
E_{s_3}	1	0,9944	1,0387	2,0331	1,0497	1,1196	0,8616	3,031
E_{s_4}	0,9949	0,8572	1,1118	1,969	0,9493	1,0692	0,986	3,0045
ΣE_{s_j}	3,9983	3,8106	4,1667	7,9773	4,0977	4,1263	3,729	11,953
E_{s_j}	0,9996	0,9526	1,0417	-	1,0244	1,0316	0,9322	-
E_s	0,9996	-	-	1,9943	-	-	-	2,9883

Tabela 4-5 – Testes para a normalização pela restrição de Potência Média com M=8.

Normalização pela restrição de Potência Média – M=8

Símbolos	n=1	n=2			n=3			
	1	1	2	ΣE_{s_i}	1	2	3	ΣE_{s_i}
E_{s_1}	0,3213	1,0077	1,0656	2,0734	1,2703	1,1996	0,4121	2,8821
E_{s_2}	1,0058	0,9544	0,9511	1,9054	0,9906	0,9631	0,1913	2,145
E_{s_3}	0,3937	0,9614	0,9724	1,9338	0,5945	0,292	0,5949	1,4813
E_{s_4}	1,1857	1,0505	1,0612	2,1117	1,3712	1,5411	1,8367	4,749

E_{s_5}	1,1769	1,0464	1,0517	2,0981	0,7426	0,3708	0,2639	1,3773
E_{s_6}	1,0202	1,0494	0,9662	2,0156	0,8141	0,5574	1,2695	2,641
E_{s_7}	1,2321	0,8938	0,9466	1,8404	1,1926	0,8438	1,4878	3,5241
E_{s_8}	1,1749	1,0405	0,9596	2,0001	0,7036	1,439	0,3921	2,5346
ΣE_{s_j}	7,5106	8,004	7,9744	15,9784	7,6795	7,2067	6,4483	21,3344
E_{s_j}	0,9388	1,0005	0,9968	-	0,9599	0,9008	0,806	-
E_s	0,9388	-	-	1,9973	-	-	-	2,6668

Tabela 4-6 – Testes para a normalização pela restrição de Potência Média com M=16.

Normalização pela restrição de Potência Média – M=16

Símbolos	n=1	n=2			n=3			
	1	1	2	ΣE_{s_i}	1	2	3	ΣE_{s_i}
E_{s_1}	1,2932	0,0211	0,0409	0,062	0,8843	1,4054	0,1262	2,416
E_{s_2}	0,6036	1,3472	0,5645	1,9118	0,4172	0,6856	1,2033	2,306
E_{s_3}	0,0273	1,2404	0,7966	2,037	1,4115	0,1267	1,1372	2,6754
E_{s_4}	0,5895	1,0045	1,0973	2,1017	1,1672	0,6885	0,5464	2,4021
E_{s_5}	1,3795	0,7099	1,2439	1,9538	1,5174	1,231	1,0616	3,8101
E_{s_6}	1,2825	1,3685	0,3706	1,7391	0,2985	1,021	1,5747	2,8942
E_{s_7}	0,624	1,2324	0,7058	1,9382	0,6711	2,2225	1,2036	4,0973
E_{s_8}	1,3161	0,8248	1,1742	1,999	0,9658	0,7068	0,2207	1,8934
E_{s_9}	0,6331	0,8882	1,1505	2,0387	0,5258	0,3876	1,1125	2,0258
$E_{s_{10}}$	1,3654	0,6054	1,3478	1,9532	1,0772	1,2305	0,7131	3,0207
$E_{s_{11}}$	1,2752	1,2879	0,6916	1,9795	0,8221	0,5517	0,3602	1,734
$E_{s_{12}}$	1,3109	0,9605	1,116	2,0765	1,1836	1,5196	1,1917	3,8949
$E_{s_{13}}$	0,0233	0,2275	1,4601	1,6876	1,3022	0,4421	0,6505	2,3948
$E_{s_{14}}$	0,5837	0,8032	1,2938	2,097	1,4602	0,7871	1,7274	3,9747
$E_{s_{15}}$	1,2735	1,3446	0,533	1,8776	0,0984	0,3077	0,6879	1,094
$E_{s_{16}}$	0,6105	0,7914	1,2325	2,0239	0,1589	0,4552	0,9354	1,5494

ΣE_{s_j}	14,1914	14,6577	14,819	29,4766	13,9614	13,7691	14,4523	42,1828
E_{s_j}	0,887	0,9161	0,9262	-	0,8726	0,8606	0,9033	-
E_s	0,887	-	-	1,8423	-	-	-	2,6364

4.1.2 Considerações quanto à normalização

Um dos principais aspectos que alia as técnicas de *machine learning* às técnicas referentes aos sistemas de comunicação é a normalização, uma vez que ela determina a forma de saída dos dados do *encoder* e ao passo que nela estão intrínsecas às restrições do canal utilizado. Desta forma, analisar e estudar as diversas normalizações possíveis se torna um fator de ganho no que tange a representação gerada ou mesmo do desempenho do *Autoencoder*.

Paras as normalizações apresentadas na parte teórica deste trabalho pode-se observar as constelações geradas pelas normalizações para a restrição de amplitude, energia ou de potência média para os AE(1,3) e AE(1,4) nas Figura 4-1 e Figura 4-10. De forma complementar, pode-se observar a constelação para a restrição de potência média para o AE(1,5) e AE(1,6) na Figura 4-3.

Notadamente, pode-se observar como a normalização pela restrição de amplitude distribuí os pontos sob uma ótica retangular, vale ressaltar como é restritiva essa normalização, uma vez que ela restringe seus valores de -1 a 1 em ambos os eixos

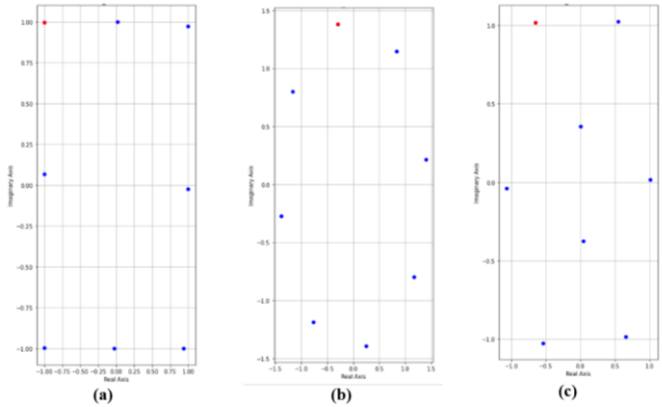

Figura 4-1 – Constelações para o AE(1,3) - (a) Restrição de Amplitude, (b) Restrição de Energia e (c) Restrição de Potência Media.

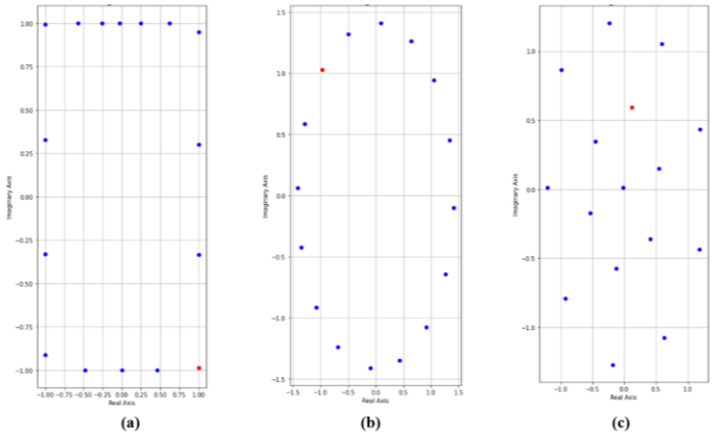

Figura 4-2 – Constelações para o AE(1,4) - (a) Restrição de Amplitude, (b) Restrição de Energia e (c) Restrição de Potência Media.

Já a constelação gerada pela restrição de energia se aproxima das constelações da modulação PSK, de acordo com a distribuição de energia média entre os símbolos vista na seção anterior. Com a restrição de potência média temos algo semelhante à modulação QAM circular, em que se têm circunferências concêntricas, simetria entre os símbolos verificada na seção anterior, fato este ainda mais visível para os casos apresentados na Figura 4-3.

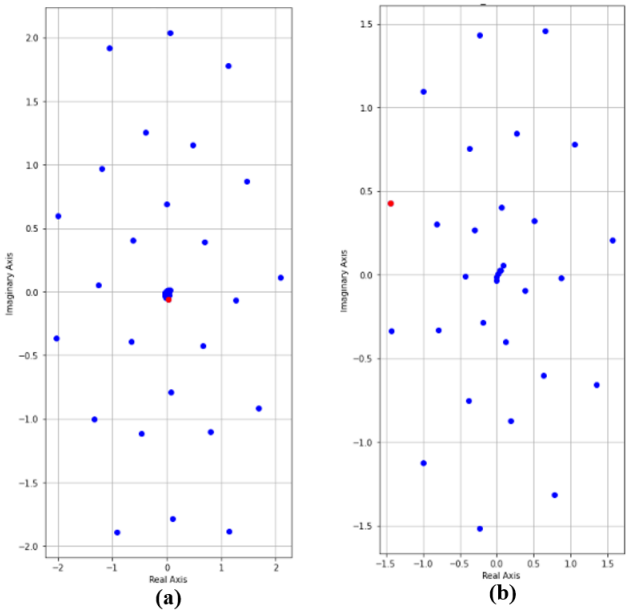

Figura 4-3 – Constelações utilizando a Restrição de Potência Media - (a) AE(1,6) e (b) AE(1,5).

Além do olhar sobre as constelações é interessante observar o desempenho desses *Autoencoders* sob a curva de BLER. Para tal foi utilizada a mesma semente (*seed* = 0) para essas simulações, ou seja, a geração das amostras de treino e a inicialização dos pesos da rede

neural se utilizaram da mesma equação geradora dentro do PyTorch. Assim, têm-se a curva desemprenho para os AE(1,3) e AE(1,4) nas Figura 4-4 e Figura 4-5, respectivamente.

Peculiarmente, para o caso do AE(1,3) a curva de BLER se mostrou com melhor desempenho para as normalizações baseadas nas restrições de energia e amplitude, desempenho este superando inclusive às modulações tradicionais. O que já não acontece com o AE(1,4), em que a normalização pela restrição de potência média se mantém próxima à modulação tradicional. No entanto, podemos observar que para ambos os casos a normalização por restrição de potência é que melhor reproduz os desempenhos das modulações tradicionais.

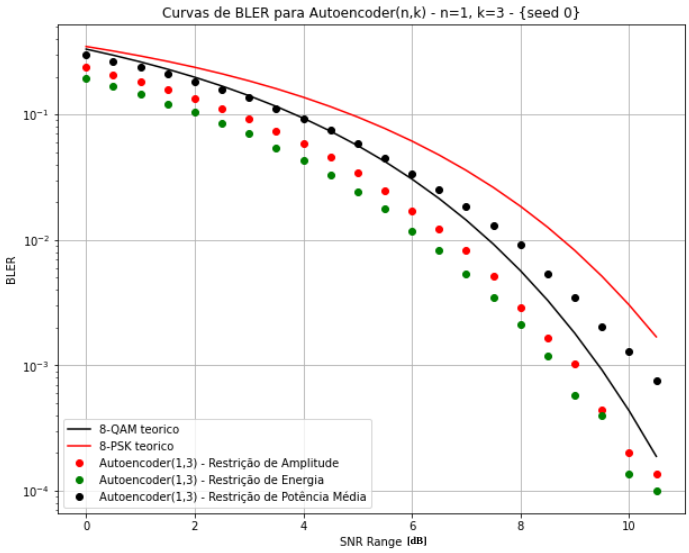

Figura 4-4 – Curva de BLER AE(1,3) em face às normalizações.

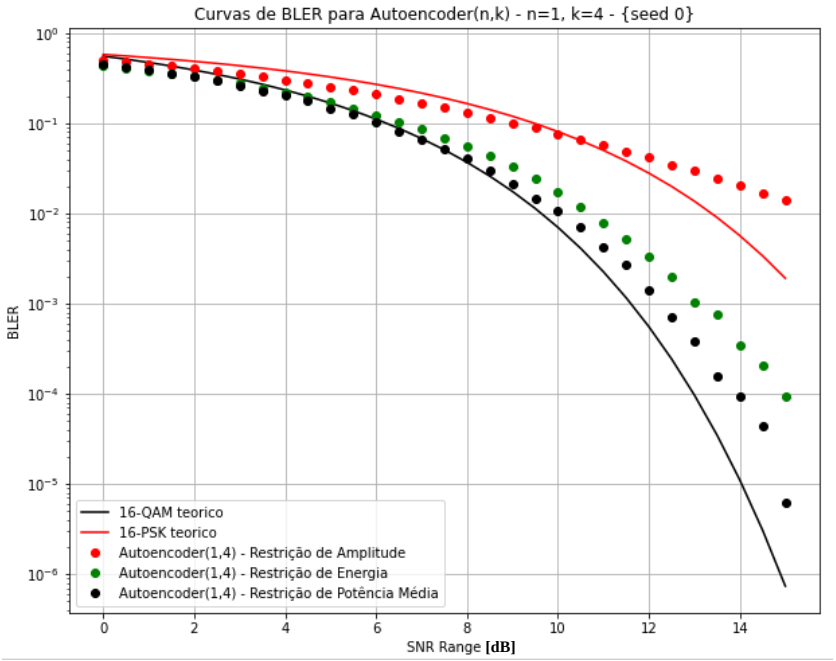

Figura 4-5 – Curva de BLER AE(1,4) em face às normalizações.

Uma outra forma de se analisar é combinando estas normalizações, uma vez que a normalização pela restrição de amplitude se mostra mais restritiva em relação às outras. Assim como em [17], foi observado que o *Autoencoder* alcança melhores desempenhos quando utilizado a normalização por restrição de potência média. No entanto, a nível de investigação foram gerados alguns resultados sob essa ótica de combinação das normalizações. Para tanto observamos, vide Figura 4-6, dois cenários para o AE(1,4), primeiro fixou-se a normalização em amplitude e, em seguida foi feita a normalização de energia ou de potência média sob os dados gerados pelo *encoder*.

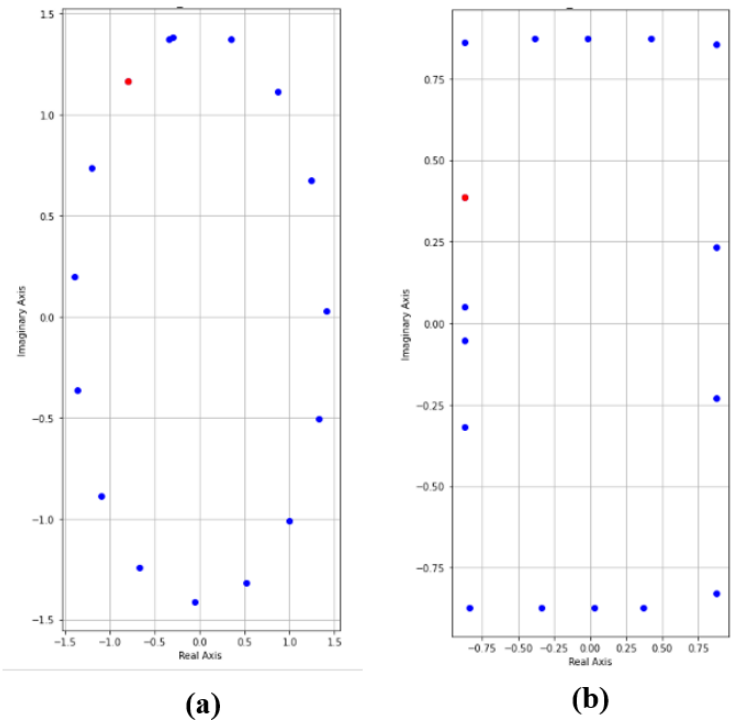

Figura 4-6 – Constelações para o AE(1,4) - (a) Combinação de normalização Amplitude e Energia e (b) Combinação de normalização Amplitude e Potência média.

Assim, pode-se visualizar a influência dessas combinações nas constelações geradas. Ao passo que para a combinação de amplitude mais energia foi mantida a forma circular de distribuição dos pontos, percebe-se que os pontos extrapolam os pontos -1 e 1 nos eixos, ou seja, uma sobreposição da normalização de energia sobre a normalização por amplitude. Já na combinação de amplitude mais potência média manteve-se a forma retangular das constelações geradas a partir da normalização por amplitude, mas com valores entre -1 e 1 nos eixos, ou seja,

como não há nenhum valor acima de 1 a normalização por amplitude não ultrapassou essa marca.

Por fim, a nível de desempenho, como visto na Figura 4-7, não foi observado nenhum tipo de ganho ou melhoria por parte da combinação das normalizações, mas fica claro que os desempenhos se aproximam da modulação PSK, conforme as constelações geradas por ambas as combinações.

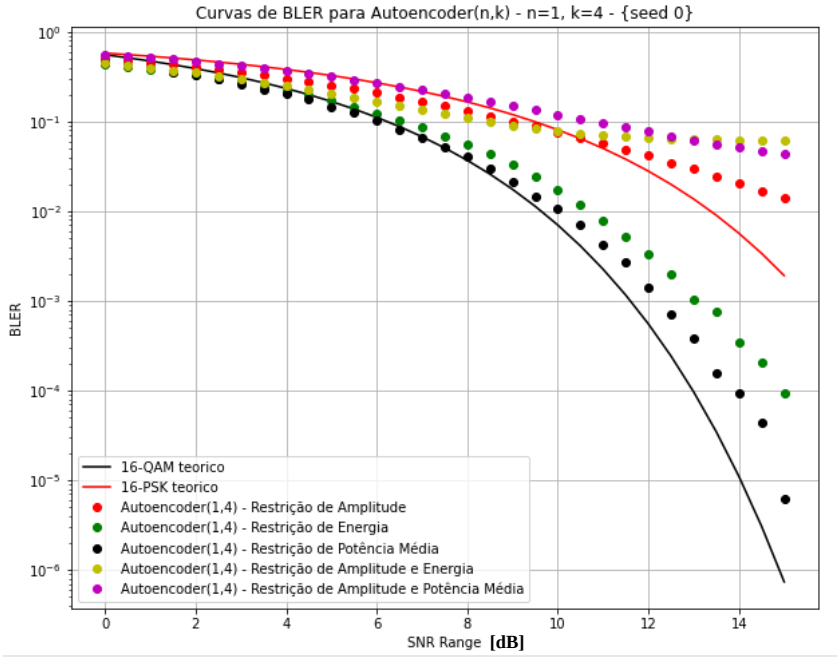

Figura 4-7 – Curva de desempenho para o AE(1,4) com combinação de normalizações.

4.1.3 Comparação com sistemas tradicionais

Para além da constelação gerada por cada *Autoencoder*, precisa-se verificar o seu desempenho em relação aos sistemas tradicionais de comunicação. A constelação é gerada e definida durante a fase de treinamento da rede neural, logo, ela está sob influência das amostras de treino gerada, bem como a SNR de treino adotada, sob os algoritmos de otimização e atualização dos pesos utilizados e até pela função de erro escolhida. Já as curvas de desempenho são geradas a partir da rede já treinada e para várias SNR. Assim, garantir que o *Autoencoder* gerou exatamente a mesma constelação do que uma modulação tradicional não garante que ele terá o mesmo desempenho, ao passo que, caso ele gere algo completamente diferente também não garante um péssimo desempenho. Logo, analisar as constelações é um bom indício, mas não se tem garantia até observar as curvas de desempenho do *Autoencoder*.

Par isso, observar as curvas de desempenho se torna representativo para se analisar o comportamento do *Autoencoder*. Desta maneira, a Figura 4-8 no apresenta um compilado dos desempenhos das redes neurais frente às curvas de BER das modulações tradicionais, logo podemos perceber como a utilização de redes neurais e técnicas de *machine learning* podem ser tão competitivas quanto os modelos tradicionais. Fica evidente o desafio de se parametrizar e encontrar a melhor arquitetura da rede neural para que se alcance o mesmo desempenho, vide resultados do AE(1,5) e AE(1,6).

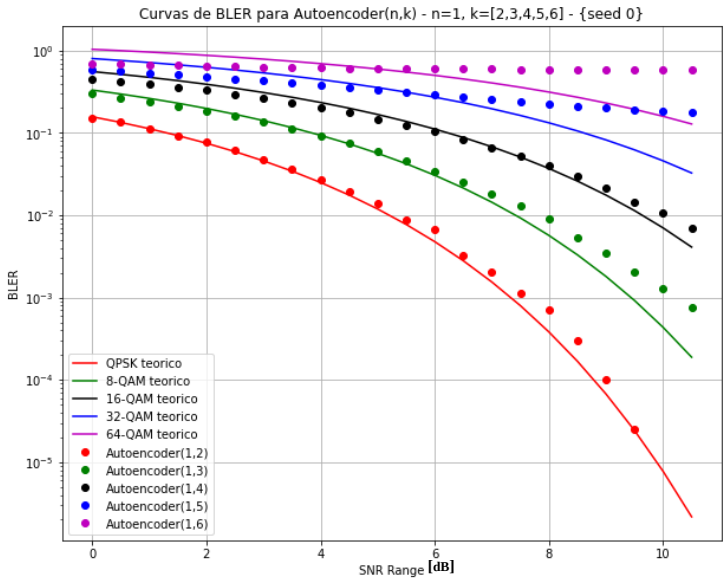

Figura 4-8 – Curvas desempenho para diversos *Autoencoders* frente às modulações tradicionais.

4.2 Capacidade de codificação do *Autoencoder*

Inicialmente, os resultados apresentados serão referentes a um conjunto de esquemas de *Autoencoders*, visando observar a capacidade de codificação de canal conjunta à representação dos símbolos nos usos do canal.

Observar as diversas possibilidades com o *Autoencoder* demanda a analisarmos sob os aspectos de diferentes tipos de canais de comunicação e comparar seus efeitos de forma justa perante os sistemas de comunicações atuais.

4.2.1 O *Autoencoder* sob canal AWGN

Foi realizado, em sequência, as simulações para AE(2,2), AE(2,4) e AE(8,8), e os gráficos de BLER – Figura 4-9, Figura 4-11 e Figura 4-14 – foi utilizado como comparação os esquemas de modulação tradicionais [10], de tal modo que pode-se observar como o *Autoencoder* se torna competitivo frente aos sistemas tradicionais, uma vez que têm-se resultados próximos ou melhores.

Vale ressaltar o custo computacional para simulações de mais altas ordens, ou seja, quanto maior o $\boldsymbol{M}$, maiores são os vetores e mais amostras são necessárias para se realizar a simulação, tornando-se cada vez mais custosa a simulação em termos de memória e processamento dos dados.

Para visualização das constelações geradas optou-se por adotar uma estratégia (a partir da limitação das cores disponíveis), em que se escolheu (até) quatro pontos da constelação (de acordo com o sistema simulado), e plotá-los de acordo com os usos do canal. Assim, as Figura 4-10, Figura 4-12, Figura 4-13 e Figura 4-15 mostram as constelações geradas para o AE(2,2), AE(2,4) e AE(8,8), respectivamente, simulados.

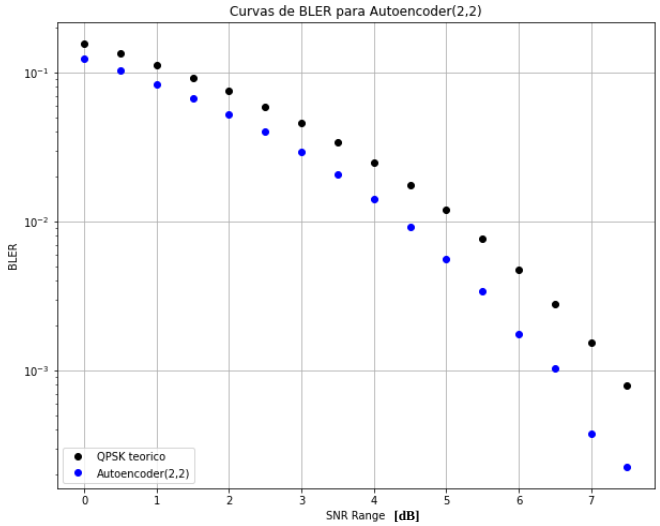

Figura 4-9 – Curvas de BLER para AE(2,2).

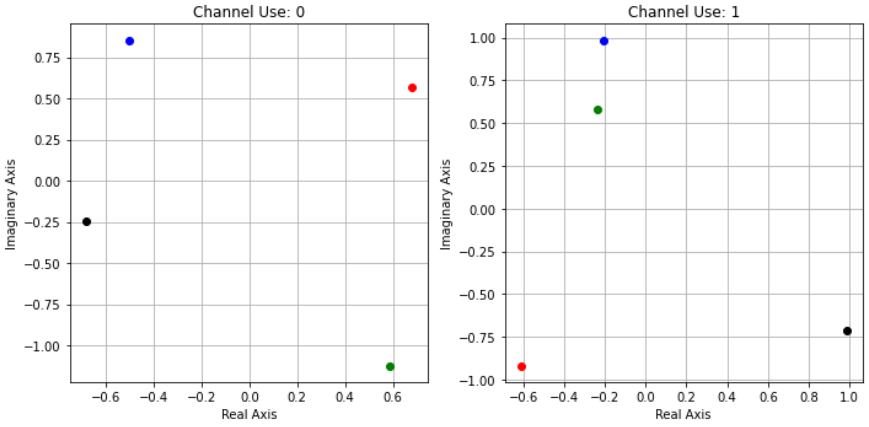

Figura 4-10 – Constelação produzida pelo AE(2,2).

Percebe-se que o *Autoencoder* procura distribuir os pontos dentro do espaço de acordo com o canal simulado, por exemplo, se pegarmos o ponto em azul da Figura 4-10, no primeiro uso do canal temos que, visualmente, ele está mais próximo aos símbolos em vermelho e preto, no entanto, já no segundo uso do canal este mesmo símbolo azul está mais distante destes mesmos símbolos citados, indo ao encontro da teoria de comunicação digital. Para se ter a menor taxa de erro deseja-se distanciar ao máximo possível os pontos relativos aos sinais mensagens na constelação [10].

Observa-se que neste esquema do AE (2,2) utilizamos duas vezes o canal de comunicação para transmitir dois bits, ou seja, no primeiro uso do canal envia-se o ponto representado em azul e no segundo uso do canal o outro ponto em azul. Estes dois pontos juntos são repassados ao *decoder* que interpretará por um sinal mensagem constituído por dois bits. Assim, analisando a curva de BLER e o mapa de constelação produzidos, verifica-se que a rede neural conseguiu produzir um esquema de codificação que melhora ou se aproxima da taxa de erro teórica por meio de uma codificação.

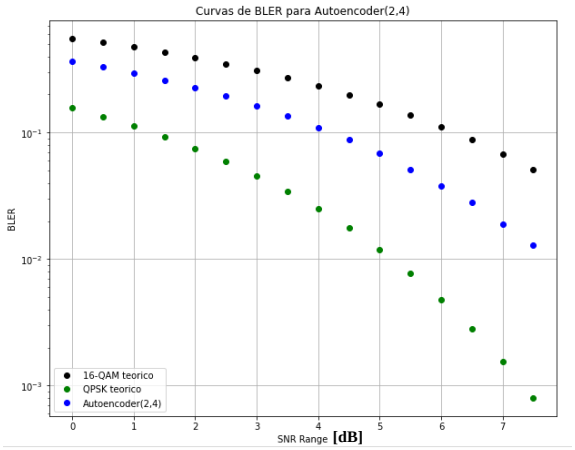

Figura 4-11 – Curvas de BLER para AE(2,4).

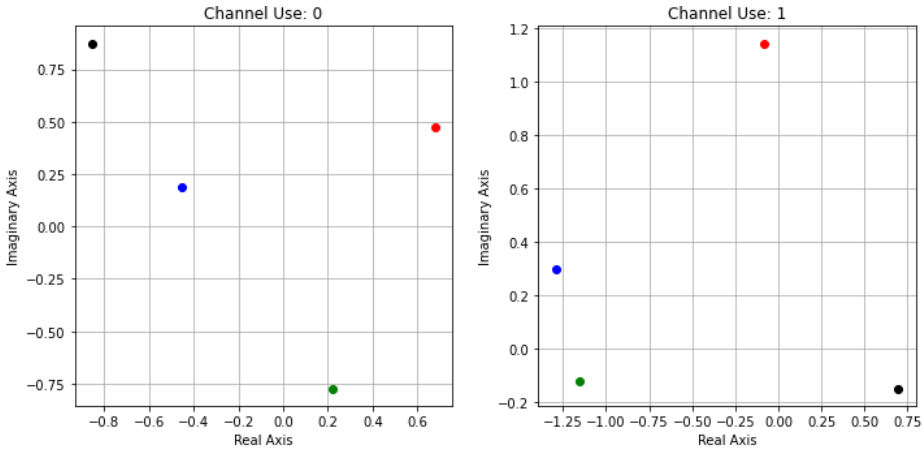

Figura 4-12 – Amostra da constelação produzida pelo AE(2,4) distribuída pelos usos do canal.

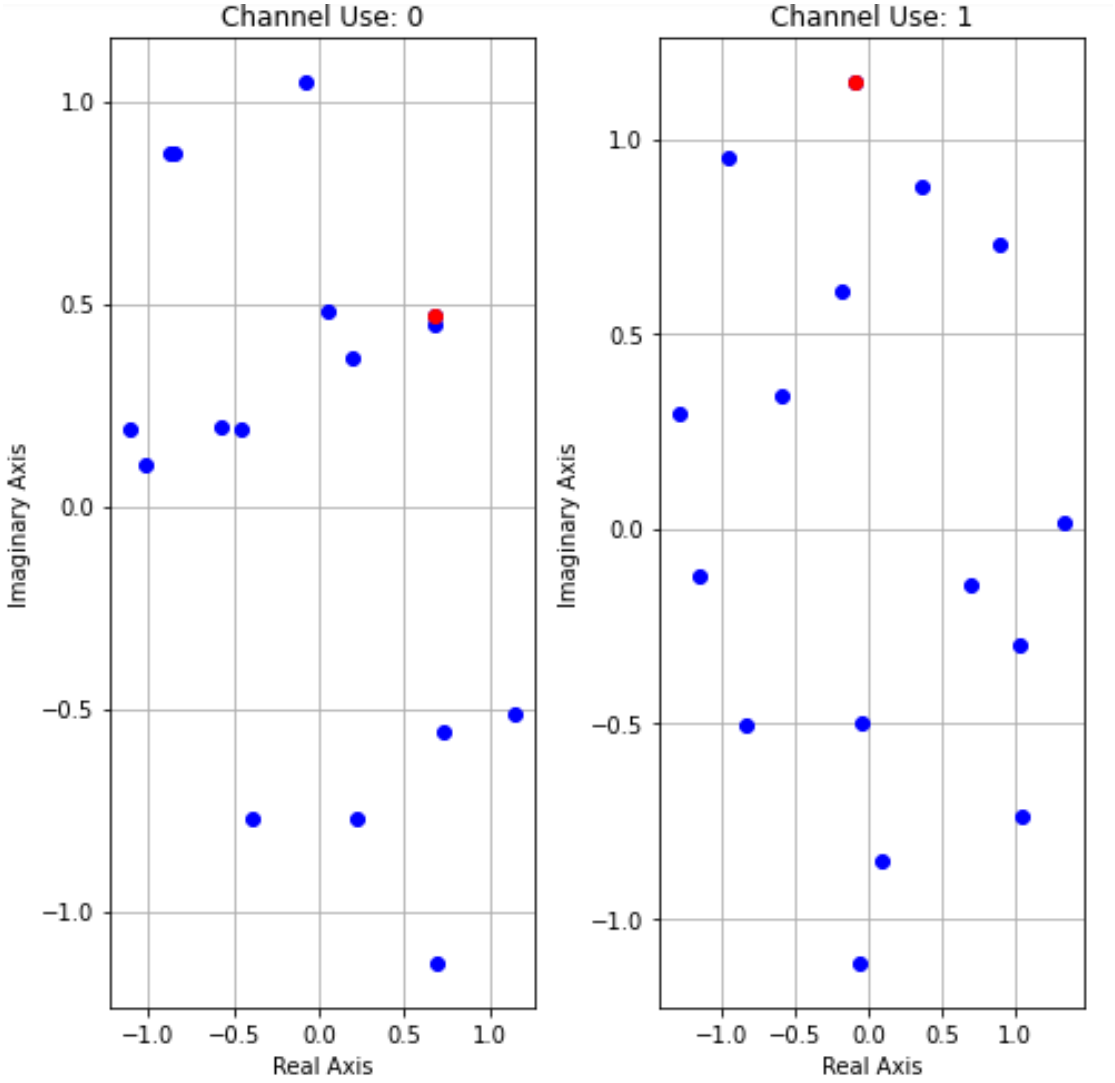

Figura 4-13 – Constelação produzida pelo AE(2,4).

Pontua-se que na Figura 4-13 temos toda a constelação produzida pelo AE(2,4). Dela pode-se notar que em cada novo treinamento a rede neural produzirá uma constelação diferente. Isso se deve ao fato de trabalharmos com um sistema de natureza aleatório, como o canal de comunicação, e que os pesos iniciais da rede neural são aleatórios a cada treinamento. Fica claro também que, não necessariamente a constelação gerada pela rede neural irá tender para os modelos tradicionais de modulação, afinal, o objetivo do *Autoencoder* é encontrar a melhor representação para um dado canal de comunicação.

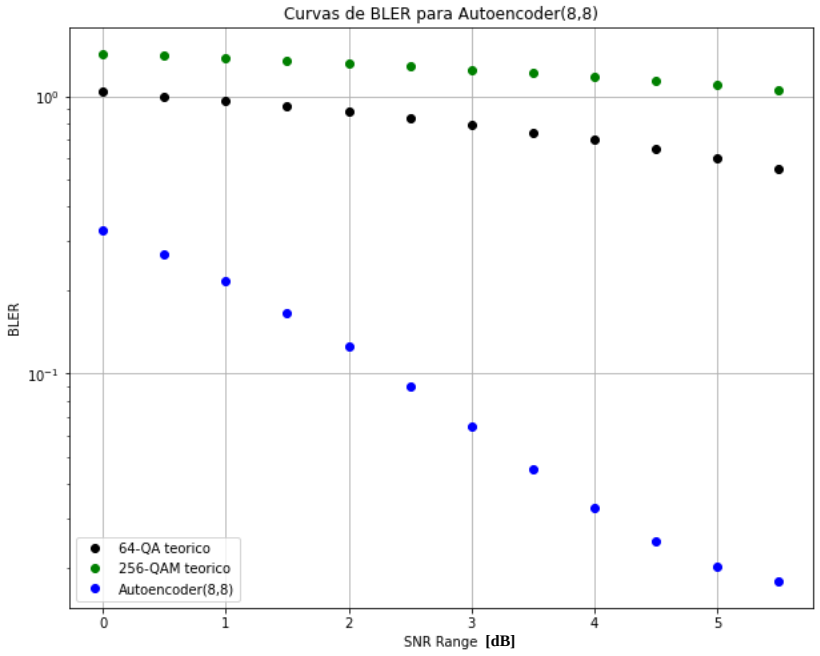

Figura 4-14 - Curvas de BLER para AE(8,8).

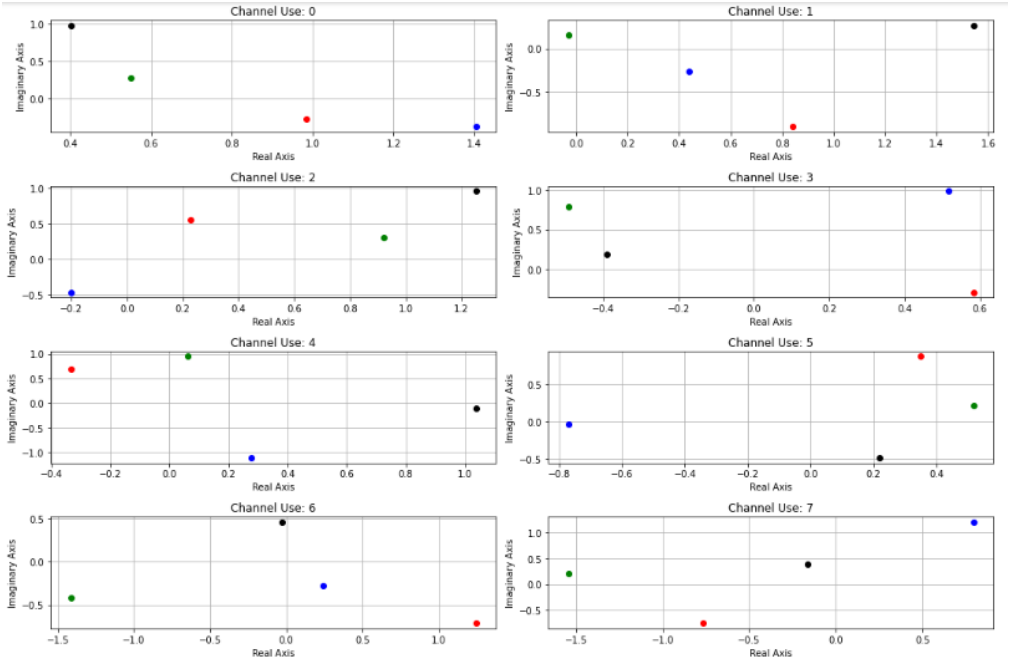

Figura 4-15 – Amostra da constelação produzida pelo AE(8,8) distribuída pelos usos do canal.

4.2.2 O *Autoencoder* sob o canal Rayleigh

O canal AWGN é o modelo de canal mais simples que existe, fazendo-se necessário observamos o desempenho do *Autoencoder* sob a perspectiva dos diferentes tipos de canais existentes, uma vez que assim poderemos verificar de fato a capacidade e robustez em face das técnicas existentes.

Logo, submetemos a rede neural sob um canal Rayleigh, que se comporta com um canal com desvanecimento plano, para avaliarmos os fatores que influem sob o *Autoencoder*. Neste

ponto foi-se utilizado uma relação sinal-ruído de treino de 12 dB e para o *decoder* foi feita a concatenação da saída do canal, ***y***, com o canal utilizado no treino, ***h***, de resto manteve-se os demais parâmetros tais qual as Tabela 3-2 e Tabela 3-3. Obteve-se os desempenhos da Figura 4-16.

Temos que observar que quanto maior a ordem de ***M*** pior são as curvas. Outro ponto é que mesmo se utilizando de mais de uma vez do canal não se conseguiu acompanhar o desempenho dos modelos tradicionais, logo precisamos olhar com atenção para identificar os pontos chaves nestes casos.

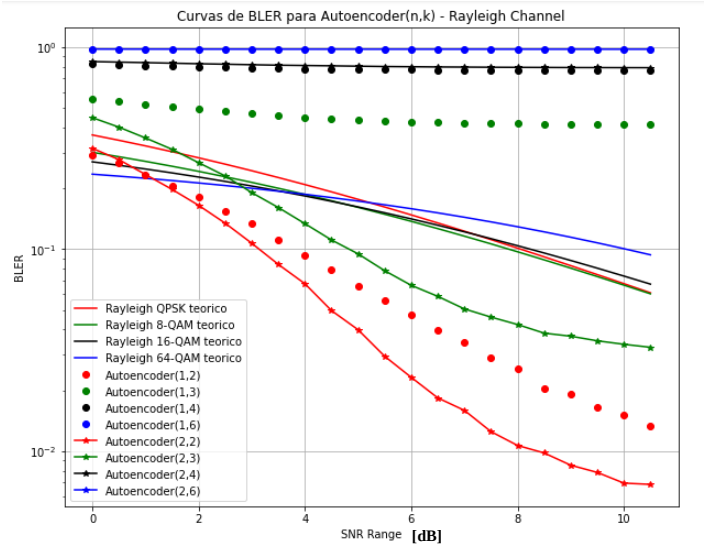

Figura 4-16 – Desempenho para o AEs com um e dois usos do canal frente às curvas do canal Rayleigh.

Um ponto relevante para análise são os gráficos das constelações, assim, vide Figura 4-17, Figura 4-18 e Figura 4-19 podemos compreender o desempenho destes sistemas.

Fica perceptível pelas Figura 4-18 e Figura 4-19 que mesmo fazendo mais de um uso do canal os *Autoencoders* obtiveram um péssimo desempenho, os pontos das constelações geradas se aglutinaram em torno de uma única região ou ponto, fazendo-o errar mais do que esperado no momento da detecção. Já pelas constelações apresentadas pela Figura 4-17 observamos que mesmo com o canal Rayleigh a constelação para o AE(1,2) se aproximou da constelação do QPSK, e para o AE(2,2) a distribuição dos pontos encontradas favoreceu na detecção, dado que ambas as redes tiverem desempenho melhor do que as dos sistemas tradicionais e para mais de um uso do canal conservou-se o comportamento esperado de melhoria do desempenho.

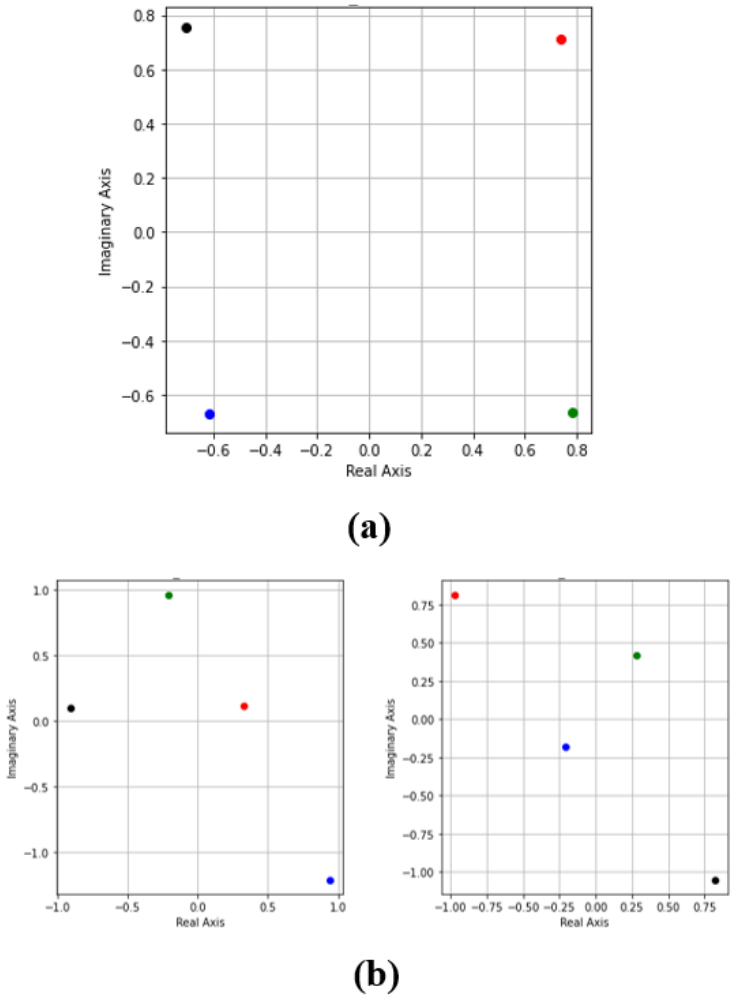

Figura 4-17 – Constelações para (a) o AE(1,2) e (b) o AE(2,2).

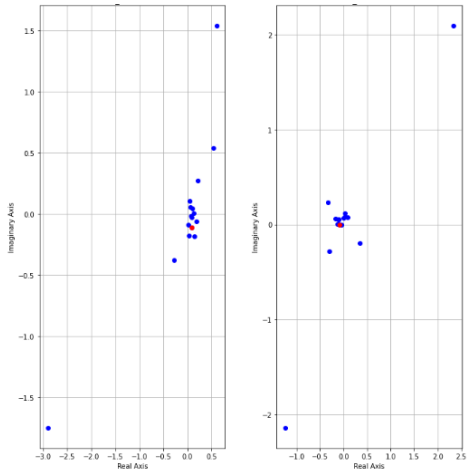

Figura 4-18– Constelações para o AE(2,4).

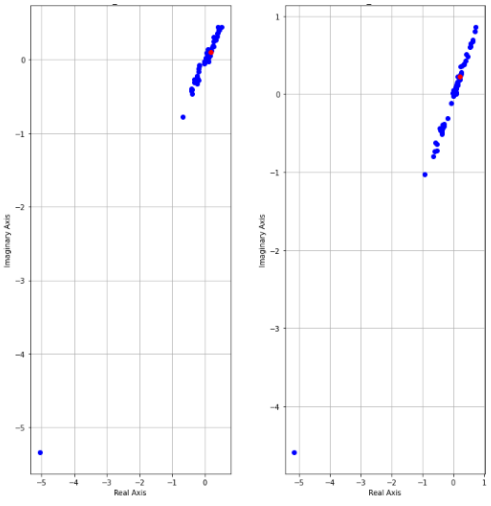

Figura 4-19– Constelações para o AE(2,6).

4.2.3 Comparação com técnicas de codificação de canal

Pelas Figura 4-9, Figura 4-11 e Figura 4-14 em que se têm mais de um uso do canal, observa-se que o *Autoencoder* aplica algum tipo de redundância na sua codificação. E, uma vez que o número de bits enviado permanece inalterado, que tal capacidade de codificação se dá pela utilização de mais recursos para transmissão, ou seja, o *Autoencoder* com vários usos do canal oferece uma maior confiança no processo de comunicação, dado os ganhos observados na taxa de erro.

Em termos de comparação, fica evidente que simplesmente comparar com as modulações tradicionais para usos do canal maior do que um é injusto. Assim, como em [2], foram gerados alguns resultados frente a sistemas que se utilizam de técnicas de codificação de canal, em especial com os códigos lineares (em bloco) e de Hamming.

Para essa comparação foram gerados os resultados via funções já implementadas em Matlab, em especial a função **bercoding** [23]**,** que gera as curvas de BER para canais AWGN codificados.

Como parâmetro de escolha de quais códigos gerar a curva de BLER utilizou-se a taxa de bits por uso do canal **R**, já definida. Uma vez que as técnicas tradicionais visam incluir redundância nas mensagens para se obter melhores taxas de erro o *Autoencoder* se apoia nos vários usos do canal, como mostrado na Tabela 4-7. Apesar de ambos os códigos serem lineares, seja os códigos em Blocos ou de Hamming, escolheu-se eles em função dos valores de suas taxas de codificação $\boldsymbol{R_c}$.

Tabela 4-7 – Técnicas, códigos e taxas.

Técnica	Taxa
AE(2,4)	2
AE(4,4)	1
AE(7,4)	0.571
AE(8,6)	0.75
Block(4,4)	1
Block(7,4)	0.571
Block(8,6)	0.75
Hamming(7,4)	0.571
Hamming(15,11)	0.733

Desta forma, vide Figura 4-20 e Figura 4-21, podemos observar a capacidade de codificação do *Autoencoder* sob a ótica dos usos do canal. Pelas curvas escolhidas e dada a proximidade dos desempenhos percebe-se que há a possibilidade de comparação dos sistemas de comunicação via a taxa de codificação e a taxa de bits por uso do canal, uma vez que a proximidade das curvas de desempenha indicam alguma relação, uma vez que não se têm diferenças significativas entre os desempenhos.

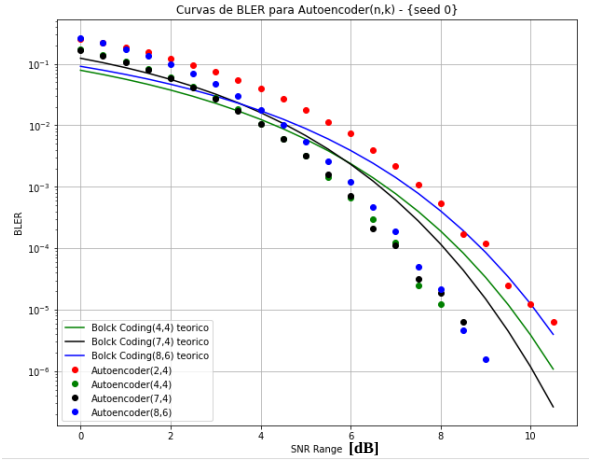

Figura 4-20 – Comparação das curvas de desempenho dos AEs com os códigos em Bloco.

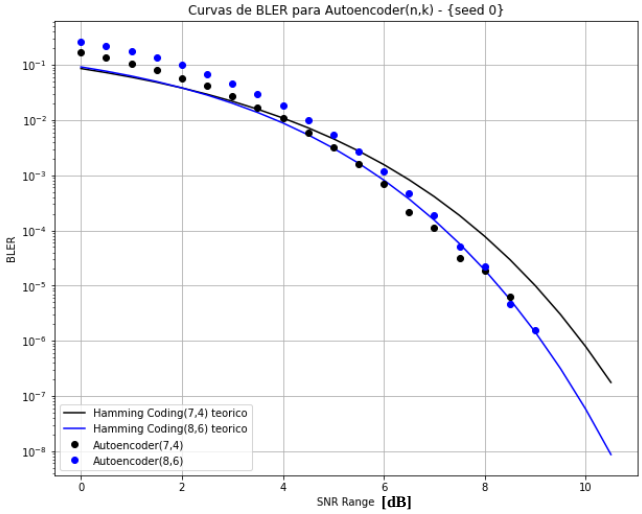

Figura 4-21 – Comparação das curvas de desempenho dos AEs com os códigos de Hamming.

4.3 O *Autoencoder* para outras arquiteturas

Os resultados apresentados a seguir são oriundos da replicação de estudos presentes na literatura. Utilizando-se outras arquiteturas para se explorar o comportamento e visando encontrar novas possibilidades.

4.3.1 O *Autoencoder* como MIMO

A partir da modelagem apresentada e utilizando [5] e [21] como inspiração, conseguiu-se replicar os resultados para o caso *Open Loop* e *Closed Loop*. Foi utilizada a arquitetura da rede neural de acordo com a Tabela 4-8. Sendo que foram possíveis apenas 2000 épocas de treinamento em detrimento das 10000 épocas utilizadas por [21], devido ao recurso computacional disponível, utilizou-se uma SNR de treino de 15 dB para o treinamento do caso *Open Loop* e 12 dB para o *Closed Loop*.

Tabela 4-8 – Parâmetros do *Autoencoder* para MIMO. [20]

Autoencoder	**Camada + Função de Ativação**	**Dimensão de Saída**	
		Open Loop	***Closed Loop***
Encoder	Entrada	M^2	$M+2NrNt$
	Densa + ReLU	64	64
	Densa + Linear	64	64

	Densa + Linear	***2nNt***	***2nNt***
	Normalização	***2nNt***	***2nNt***
Canal sem fio	Canal com Desvanecimento plano e Ruído AWGN	***2nNr+2NrNt***	***2nNr+2NrNt***
Decoder	Densa + ReLU	512	512
	Densa + ReLU	512	512
	Densa + ReLU	512	512
	Densa + Softmax	***M***	***M***

Para o caso Open Loop a Figura 4-22 traduz o esquema utilizado, em que se concatenou o canal de comunicação com ***y***, o resultado obtido pela multiplicação matricial com as representações geradas pelo *encoder* ***x*** com a matriz do canal ***H*** para se entrar no *decoder*.

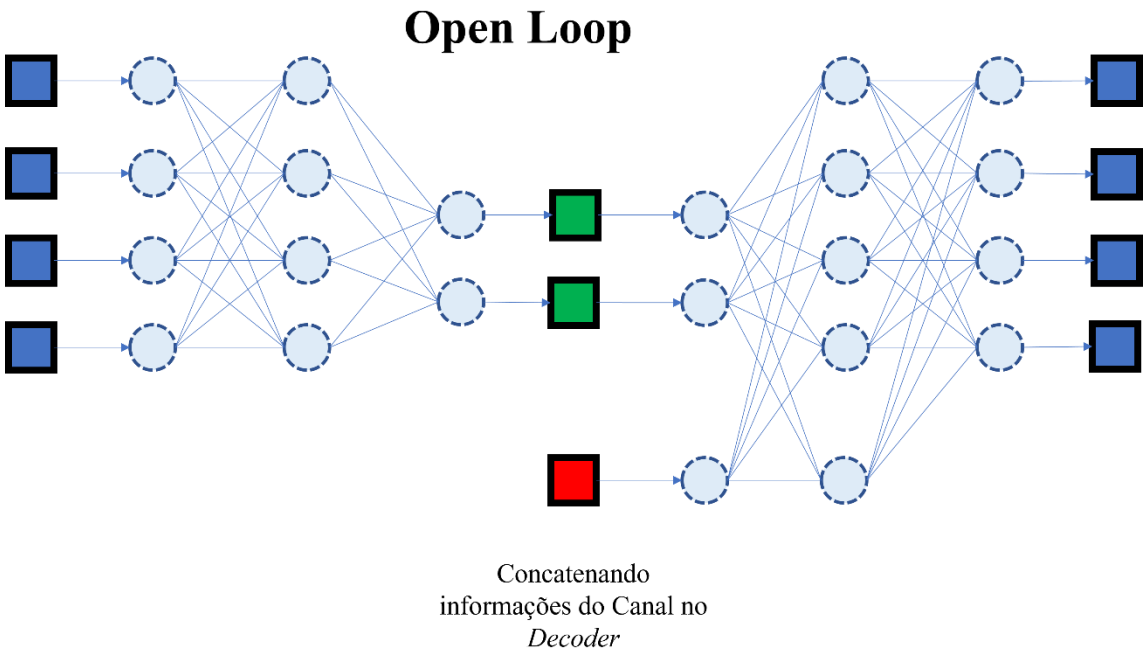

Figura 4-22 – Estrutura do *Autoencoder* para o caso MIMO *Open loop*.

Os parâmetros do sistema para este caso foram ***M*** = 4, ***L*** = 2, ***Nt*** = 2 e ***Nr*** = 1. Logo, por essas definições temos M^L = 16 mensagens possíveis. Para comparação utiliza-se o esquema de codificação no tempo e espaço proposto por Alamouti, [13] e [21], como visto na Figura 4-23.

Logo, os resultados obtidos para este sistema são mostrados nas Figura 4-24 e Figura 4-25, acarretando desempenhos semelhantes aos obtidos por [21], ainda com a redução do número de épocas de treinamento.

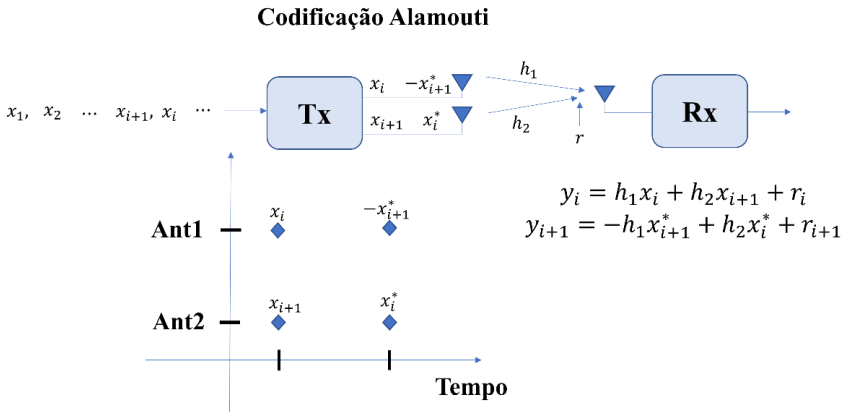

Figura 4-23 – Esquema de Codificação Alamouti.

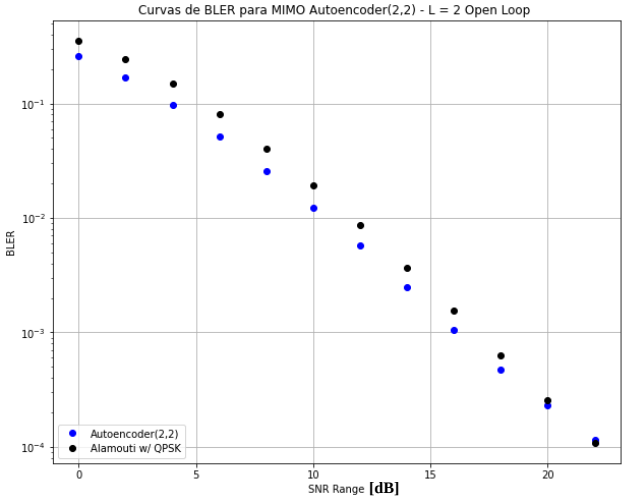

Figura 4-24 – Desempenho do sistema MIMO 2x1 com codificação Alamouti em comparação com o AE(2,2).

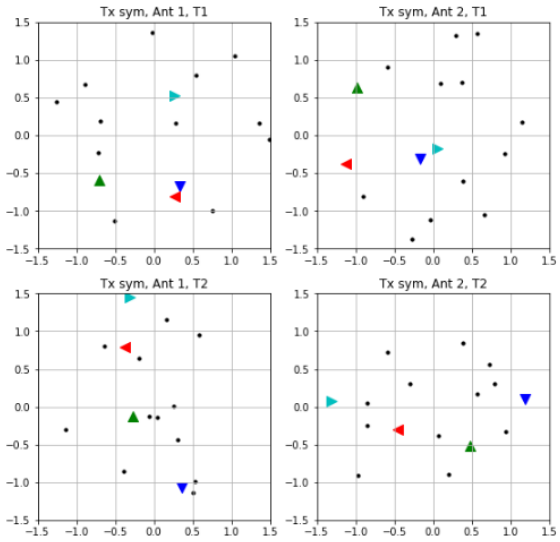

Figura 4-25 – Constelação geradas para *M* =16 para o AE(2,2).

Já para o caso *Closed Loop* a Figura 4-26 traduz o esquema utilizado, em que se concatenou o canal de comunicação com os vetores de entrada ***s*** e ***y***, em que ***y*** é o resultado obtido pela multiplicação matricial com as representações geradas pelo *encoder* ***x*** com a matriz do canal ***H*** para se entrar no *decoder* e ***s*** a representação em *one-hot vector* dos dados de entrada do *encoder*.

Os parâmetros do sistema para este caso foram ***M*** = 16, ***Nt*** = 2 e ***Nr*** = 2. Logo, por essas definições temos M^L = 16 mensagens possíveis. Para comparação utiliza-se o esquema de decomposição em autovalores da matriz do canal, SVD (*singular value decomposition*) [13] e [21].

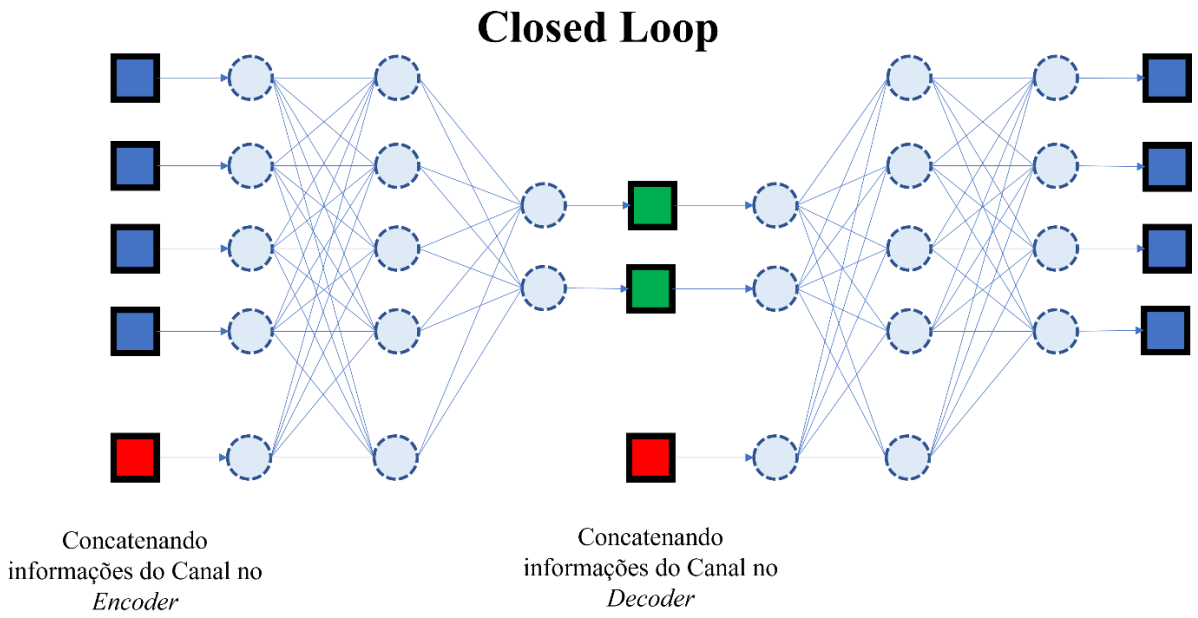

Figura 4-26 – Estrutura do *Autoencoder* para o caso MIMO *Closed loop*.

O esquema de comparação utilizado agora é o SVD, em que é possível decompor a matriz do canal, $\boldsymbol{H}$, em valores singulares de modo que $\boldsymbol{H} = \boldsymbol{U\Sigma V^{H}}$, em que as matrizes U e V são matrizes unitárias do tipo $\boldsymbol{U} \in \mathbb{C}^{\boldsymbol{N_r \times R_H}}$ e $\boldsymbol{V} \in \mathbb{C}^{\boldsymbol{N_t \times R_H}}$, em que R_{H} é o rank da matriz $\boldsymbol{H}$ e $\boldsymbol{\Sigma}$ a matriz diagonal de autovalores [8]. Observa-se que para este esquema utiliza-se da chamada precodificação $\boldsymbol{x = Vs}$, em que se codifica os símbolos enviados de acordo com a matriz $\boldsymbol{V}$ da decomposição do canal.

Com relação ao desempenho obtido pode-se verificar na Figura 4-27, e chegou-se a resultados semelhantes de acordo com [21], pontuando-se a redução do número de épocas.

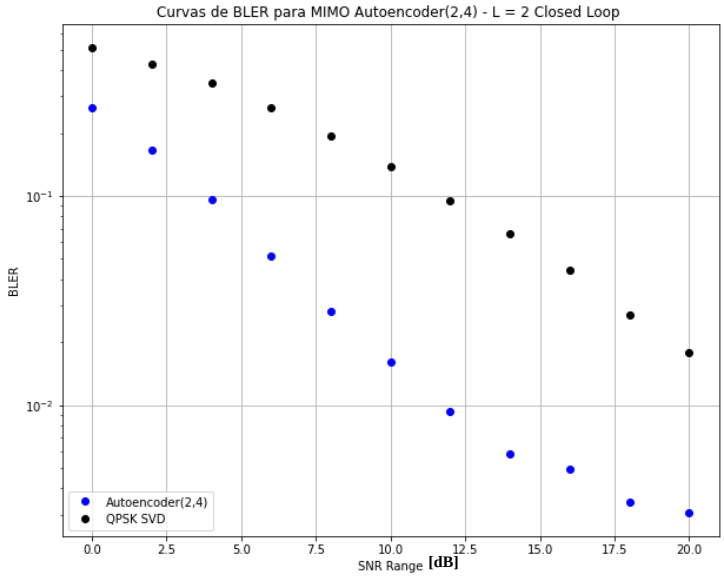

Figura 4-27– Desempenho do sistema MIMO 2x2 com decomposição SVD em comparação com o AE(2,2).

Por fim, observa-se a necessidade de recursos computacionais, sejam eles de memória ou de processamento, para se realizar as simulações para sistemas MIMO de ordens maiores, bem como para sistemas SISO em que se também há ordens maiores de modulações. Encontra-se em [24] uma análise sobre a escalabilidade dos recursos computacionais necessários, em especial de memória, em função do número de antenas no transmissor e da constelação desejada.

4.3.2 O Autoencoder com CNN

A forma de se estruturar os $\boldsymbol{L}$ blocos de dados em sequência, como mostrado na Figura 3-3 não se mostra um tanto quanto interessante devido a sua escalada exponencial em detrimento das limitações físicas do hardware de simulação. Com essa motivação buscou-se na literatura outras estruturas para se construir o *Autoencoder*, e entre tantas variações, cita-se os resultados obtidos em [22] com uma estrutura de redes neurais convolucionais (CNN) em uma dimensão

A razão aqui para se utilizar uma CNN baseia-se exatamente na forma de visualização e formatação dos dados de entrada da rede neural (em suma, a representação em *one-hot vector*) junto à transmissão de blocos de mensagens (seja em um canal SISO ou MIMO). Portanto, em vez de se ter uma forma vetorial, $\boldsymbol{1xM}$, buscou-se formatar os dados de entrada em sua forma matricial do tipo $\boldsymbol{LxM}$.

Em [22] têm-se uma proposta de arquitetura baseada em CNN de uma dimensão, assim, a operação de convolução discreta é realizada apenas em uma dimensão, a saber, na dimensão $\boldsymbol{M}$, em que na representação *one-hot vector* mapeou-se as mensagens possíveis nesta dimensão.

Para tanto, chega-se aos resultados, compilados na Figura 4-28, tal qual encontrado na literatura e descrito neste trabalho, para $\boldsymbol{L}$ = 10 sob o canal AWGN utilizando-se CNN.

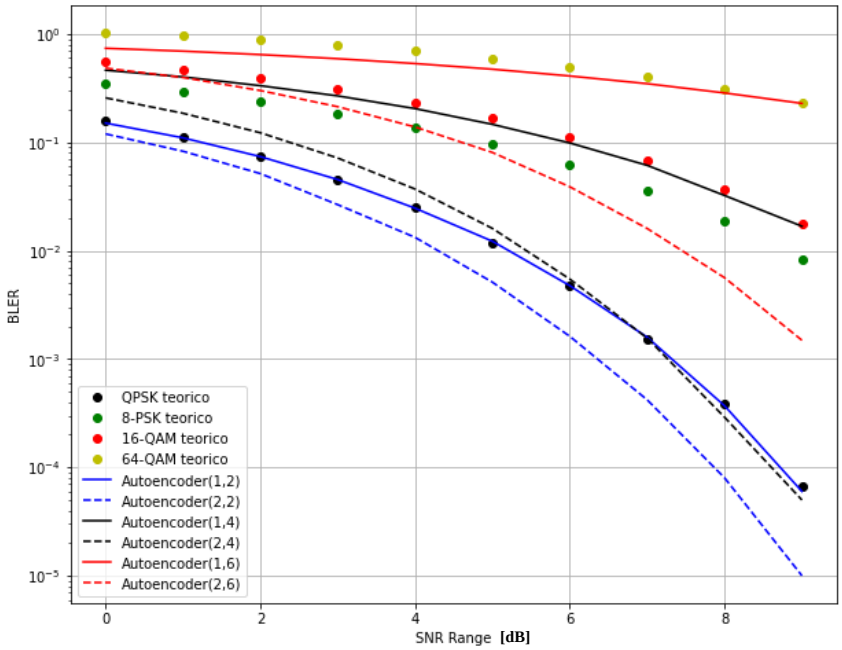

Figura 4-28 – Compilação dos resultados para o canal AWGN utilizando-se CNN.

Busca-se aqui analisar os resultados mostrados na Figura 4-28, em que os pontos representam as modulações tradicionais, as linhas sólidas para $\boldsymbol{n} = 1$**,** ou seja, um uso do canal, e em seguida as linhas tracejadas representam os *Autoencoders* para dois usos do canal, $\boldsymbol{n} = 2$. Diante disso observa-se que o *Autoencoder* apresentou o mesmo desempenho dos sistemas tradicionais quando comparado a um uso do canal e quando se têm mais de um uso do canal o *Autoencoder* apresentou um ganho na BER.

De [22] compilaram-se os resultados obtidos na Tabela 4-9. Deve-se destacar alguns pontos. O primeiro é que através da CNN conseguiu-se obter os mesmos resultados que os

sistemas convencionais para o canal AWGN e Rayleigh. O segundo ponto é referente à relação sinal-ruído de treino, por mais que se tenha obtido o mesmo desempenho, para cada esquema de *Autoencoder*, foi requisitado um valor de E_b/N_0 diferente para se chegar a esse desempenho. Por fim, um terceiro ponto que se quer levantar é o de blocos de mensagens, ou seja, os resultados se mantiveram constantes independentemente do número de mensagens utilizadas, ou seja, o desempenho não se alterou seja para $\boldsymbol{L} = 10$ ou $\boldsymbol{L} = 100$, o que é interessante se notar a nível de escalabilidade e potencialidade sobre o *Autoencoder*.

Dentro deste tema ressalta-se a robustez encontrada nesta estrutura, de tal maneira que se encontrou outras referências, como [25], que se utilizaram desta arquitetura para se obter novos resultados nos mais diferentes cenários possíveis.

Tabela 4-9 – Compilação dos resultados obtidos por [22].

k: Nº de Bits	*n*: Usos do Canal	Eb/N0 de Treino	Tipo de Canal	Resultados de Comparação obtidos
1	1	3 dB	AWGN	Pior BLER
1	1	9 dB	AWGN	Melhor BLER
1	1	20 dB	AWGN	-
1	1	5 dB	AWGN	BPSK Teórico
2	1	6 dB	AWGN	QPSK Teórico
4	1	9 dB	AWGN	16-QAM Teórico

6	1	14 dB	AWGN	64-QAM Teórico
1	1	14 dB	Rayleigh	BPSK Teórico
2	1	16 dB	Rayleigh	QPSK Teórico
4	1	22 dB	Rayleigh	16-QAM Teórico
6	1	27 dB	Rayleigh	64-QAM Teórico
4	1	9 dB	AWGN	16-QAM Teórico com L =100
4	2	6 dB	AWGN	L =100
6	1	14 dB	AWGN	64-QAM Teórico com L =100
6	2	7 dB	AWGN	L =100
4	1	22 dB	Rayleigh	16-QAM Teórico com L =100
4	2	15 dB	Rayleigh	L =100
6	1	27 dB	Rayleigh	64-QAM Teórico com L =100
6	2	19 dB	Rayleigh	L =100

5 CONCLUSÃO

Observa-se aqui a capacidade que o *encoder* possui de se determinar a melhor constelação dos símbolos, visto também em [2], [6] e [9], seja treinado em um simples canal AWGN ou em canais mais complexos, logo, o objetivo de se encontrar a melhor codificação dos sinais mensagens a fim de ser ter uma melhor taxa de erro em função da relação sinal-ruído do canal, mostrou-se bastante interessante e promissora em virtude do desempenho apresentado pelo *Autoencoder*.

Observa-se que as constelações produzidas se espelham nas curvas de BER ou BLER, como indicadores da eficiência da técnica utilizada. Assim, o *Autoencoder* combina as técnicas de codificação e modulação para melhorar a comunicação. Logo, pode-se notar o quanto a utilização das redes neurais se mostra favorável nos sistemas de comunicação. Dados os presentes resultados e a bibliografia estudada, percebe-se que o *Autoencoder* atua como uma espécie de codificador no tempo a fim de se melhorar a BER, propiciando-se assim um ganho de diversidade. No entanto, pode-se explorar diferentes configurações para o *Autoencoder* a fim de se minimizar o custo computacional de treinamento e aumentar a taxa de transmissão, obtendo-se assim um ganho de multiplexação.

Tratando-se do sistema MIMO, que combina diversas técnicas de processamento de sinais para se obter um ganho de multiplexação e diversidade, se pondera que o *Autoencoder* desempenha o papel como um bloco codificador e modulador referente aos tradicionais STC. Enquanto se dispõe de informações do canal pode-se aplicar tais dados na entrada do *encoder*

para que se obtenha um melhor desempenho em relação os sistemas que se utilizam deste artifício.

A partir da revisão bibliográfica e da replicação de resultados realizada fica claro a potencialidade e o leque de possibilidades que se observa envolvendo técnicas de aprendizado de máquinas aplicados em telecomunicações, em especial, as redes neurais em cascata que formam o *Autoencoder* que modelam sistemas de comunicações fim a fim. Resultando em promissoras técnicas, sejam elas envolvendo aprendizado de máquina ou de cunho híbrido com as técnicas convencionas já existentes para se aprimorar ainda mais os sistemas de comunicações.

Com a revisão bibliográfica percebe-se que o *Autoencoder* pode ser construído a partir de diversas arquiteturas de redes neurais (sejam elas DNNs, CNNs ou até mesmo LSTMs). Ainda não se há um consenso sobre uma única configuração utilizada por toda a comunidade, como uma relação sinal-ruído única para treinamento, ou método e modelo de rede neural genérica. No entanto, percebe-se que o *Autoencoder* está sendo aplicado em sistemas com desvanecimento, alta interferência, múltiplos usuários, sistemas MIMO, entre outras áreas e vem se mostrando interessante, principalmente como um bom aliado no que diz respeito à diminuição da taxa de erro e no aumento da taxa de transmissão, tornando os sistemas de comunicações cada vez melhores e mais robustos perante os efeitos deletérios do canal de comunicação.

Como perspectivas de trabalhos futuros elenca-se a utilização de arquiteturas e modelos mais complexos na configuração do *Autoencoder*, seja para sistemas MIMO ou não, como a

utilização de CNN em duas ou três dimensões a fim de se ter um ganho de diversidade e um ganho de multiplexação combinados para esse sistema, uma vez que se pode explorar uma arquitetura para codificação de blocos de mensagens maiores.

Outro campo de estudo é o de aplicação de outros modelos de *Autoencoders* a sistemas fim a fim, como o VAE, o DAE e até o CAE, visando obter-se uma melhor constelação ou mesmo garantir, independentemente das inicializações aleatórias dos pesos, a convergência da rede neural para valores delimitadamente aceitáveis. Bem como se fazer um estudo para se encontrar e estabelecer configurações estáveis para o *Autoencoder.*

Para além disso, busca-se melhoramentos na etapa de treinamento, na modelagem do canal e na detecção dos símbolos junto à bibliografia existente, quanto a algoritmos adaptativos e genéticos que possam auxiliar e estabelecer esses delimitadores que condicionem à robustez do *Autoencoder* ou mesmo técnicas de recompensa ou penalização como já utilizado por técnicas de aprendizado por reforço (as redes neurais recorrentes, RNN).

Fica evidente os potenciais ganhos no desempenho que o a utilização do *Autoencoder* pode trazer ao campo de telecomunicações. Neste trabalho foi feita uma exposição da técnica, bem como a sua sistematização e modelagem no que tange os seus principais parâmetros e arquiteturas, além do mais observa-se como técnicas de *machine learning* estão ganhando cada vez mais espaço das pesquisas e no desenvolvimento de novas aplicações.

REFERÊNCIAS BIBLIOGRÁFICAS

[1] S. Haykin, Neural networks and learning machines, Third Edition ed., New Jersey: Pearson Prentice Hall, 2009.

[2] T. O. a. J. Hoydis, “An Introduction to Machine Learning Communications Systems,” 2017.

[3] I. G. a. Y. B. a. A. Courville, Deep Leraning, The MIT Press, 2016.

[4] Mu, T., Chen, X., Chen, L., Yin, H., & Wei, G., “Spatial-Time MIMO Autoencoder for Physical Layer,” *The 12th International Conference on Wireless Communications and Signal Processing*, pp. 143-148, 2020.

[5] Song, J., Häger, C., Schröder, J., O'Shea, T., & Wymeersch, H., “Benchmarking End-to-end Learning of MIMO Physical-Layer Communication,” 19 May 2020.

[6] O'Shea, T. J., Erpek, T., & Clancy, T. C, “Deep Learning-Based MIMO Communications,” 2017.

[7] Dörner, S., Cammerer, S., Hoydis, J., & Ten Brink, S., “Deep Learning-Based Communication Over the Air,” 2017.

[8] Wang, T., Wen, C. K., Wang, H., Gao, F., Jiang, T., & Jin, S., “Deep Learning for Wireless Physical Layer: Opportunities and Challenges,” *China Communications*, pp. 92-111, November 2017.

[9] M. N. a. Y. W. DEHAO WU, “Deep Learning-Based Autoencoder for m-User Wireless Interference Channel Physical Layer Design,” *IEEE Access*, pp. 174679-174691, 5 October 2020.

[10] B. P. Lathi e Z. Ding, Sistemas de Comunicações Analógicos e Digitais Modernos, 4ª edição ed., LTC; 4ª edição, 2012.

[11] A. A.-Z. Simon R. Saunders, ANTENNAS AND PROPAGATION FOR WIRELESS COMMUNICATION SYSTEMS, JohnWiley & Sons Ltd, 2007.

[12] C. E. Shannon, “Mathematical Theory of Communication,” *Bell System Technical Journal*, 1948.

[13] J. R. Hampton, Introduction to MIMO Communications, New York: Cambridge University Press, 2014.

[14] T. V. C. a. E. Björnson, “Massive MIMO Communications,” em *5G Mobile Communications*, Springer, 2017, pp. 77-116.

[15] Karmakar, R., Chattopadhyay, S., & Chakraborty, S., “Impact of IEEE 802.11n/ac PHY/MAC High Throughput Enhancement over Transport/Application Layer Protocols—A Survey,” *IEEE Communication Surveys and Tutorials*, 2017.

[16] Y. B. a. A. C. Ian Goodfellow, Deep Leraning, The MIT Press, 2016.

[17] S. Ruder, “An overview of gradient descent optimization,” Dublin, 2017.

[18] DSA Data Science Academy, “Deep Learning Book,” [Online]. Available: https://www.deeplearningbook.com.br/variational-autoencoders-vaes-definicao-reducao-de-dimensionalidade-espaco-latente-e-regularizacao/. [Acesso em 2022 10 06].

[19] The PyTorch Foundation, “PyTorch,” The Linux Foundation, [Online]. Available: https://pytorch.org/. [Acesso em 2022 09 30].

[20] J. L. B. Diederik P. Kingma, “ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION,” em *ICLR: International Conference on Learning Representations*, San Diego, CA, USA, 2015.

[21] J. Song, C. Häger, J. Schröder, T. J. O’Shea, E. Agrell e H. Wymeersch, “Benchmarking and Interpreting End-to-end Learning,” *IEEE Transactions on Wireless Communications*, vol. 21, pp. 7287 - 7298, 15 03 2022.

[22] N. Wu, X. Wang, B. Lin e K. Zhang, “A CNN-Based End-to-End Learning Framework Toward Intelligent Communication Systems,” *IEEE Access* , pp. 110197 - 110204, 05 07 2019.

[23] The MathWorks, Inc., “bercoding documentation,” [Online]. Available: https://www.mathworks.com/help/comm/ref/bercoding.html. [Acesso em 30 09 2022].

[24] D. G. Martí, D. Badini, D. D. Donno e J. Widmer, “Scalable Machine Learning Algorithms to Design Massive MIMO,” MSWiM '21: Proceedings of the 24th International ACM Conference on Modeling, Analysis and Simulation of Wireless and Mobile Systems, p. 167–171, 11 2021.

[25] S. B. F. Gomes e M. D. Yacoub, “CNN-Based Learning System in a Generalized Fading Environment,” 2020.

[26] The MathWorks, Inc., “Autoencoders for Wireless Communications,” The MathWorks, Inc, [Online]. Available: https://www.mathworks.com/help/comm/ug/autoencoders-for-wireless-communications.html. [Acesso em 30 09 2022].