Universidade Federal do ABC
Centro de Engenharia, Modelagem e Ciências Sociais Aplicadas
Programa de Graduação em Engenharia da Informação

Uso de aprendizado de máquina na identificação e classificação de sinais de redes de IoT sem fio

Daniel Vitor Beraldo Di Gênova

Santo André - SP, Agosto de 2025

Daniel Vitor Beraldo Di Gênova

Uso de aprendizado de máquina na identificação e classificação de sinais de redes de IoT sem fio

Trabalho de Graduação apresentado ao Programa de Graduação em Engenharia da Informação, como parte dos requisitos necessários para a obtenção do Título de Bacharel em Engenharia da Informação.

Orientador: Prof. Dr. Ivan Roberto de Santana Casella

Santo André - SP

Agosto de 2025

Resumo

A utilização de diferentes tecnologias de comunicação sem fio tem desempenhado um papel fundamental na viabilização da conectividade entre dispositivos, impulsionando o crescimento das redes de Internet das Coisas ou *Internet Of Things* (IoT). O uso de redes sem fio tornaram o uso da IoT mais factível, conectando os dispositivos e proporcionando a troca de mensagens. Neste contexto, este projeto tem como principal objetivo estudar o uso de técnicas de aprendizado de máquina para o reconhecimento de alguns esquemas de modulação empregados em redes de IoT sem fio. Primeiramente, o projeto apresenta uma breve fundamentação teórica de assuntos que são abordados durante sua execução. Em seguida, é desenvolvido um classificador utilizando redes neurais convolucionais para o reconhecimento de sinais de modulação digital utilizando ou não a técnica OFDM (*Orthogonal Frequency Division Multiplexing*). As análises de desempenho realizadas mostraram que o classificador desenvolvido atingiu um desempenho satisfatório com acurácia de 98,83% em sinais obtidos através de simulação em um canal AWGN, 96,50% em sinais obtidos com o uso de um Rádio definido por software (SDR) em um canal real e 77,54% de acurácia com sinais através da simulação de canais Rician. Por fim, são apresentadas as principais conclusões do estudo realizado e possíveis temas futuros que podem ser abordados para a evolução do trabalho.

Palavras-chaves: aprendizado de máquina, redes neurais convolucionais, Internet das coisas, redes sem fio, modulação digital, multiplexação por divisão de frequência ortogonal.

Sumário

Introdução

A utilização de redes sem fio tem desempenhado um papel fundamental na consolidação da Internet das Coisas, ao possibilitar uma comunicação contínua, eficiente e escalável entre dispositivos heterogêneos, como sensores, atuadores, dispositivos móveis e sistemas embarcados. Esses dispositivos são capazes de coletar, processar e compartilhar dados de forma autônoma, permitindo a troca de informações em tempo real. Essa conectividade promove a integração de sistemas, viabiliza a automação de processos e ajuda na tomada de decisões baseada em dados obtidos de maneira distribuída, contribuindo para o desenvolvimento de soluções inteligentes em diferentes setores.

As redes sem fio são sistemas de comunicação que permitem a transmissão e o recebimento de dados sem a necessidade de conexões físicas por cabos, utilizando a propagação de ondas eletromagnéticas pelo ar como meio de transporte da informação. Essas redes oferecem mobilidade, flexibilidade e facilidade de implantação, sendo fundamentais em ambientes onde a instalação de infraestrutura cabeada é inviável, custosa ou pouco prática.

Nas redes sem fio, a informação é modulada sobre ondas portadoras, isto é, o sinal a ser transmitido é modificado conforme a técnica de modulação empregada e de acordo com a informação a ser enviada. Cada tipo de modulação associa símbolos específicos a diferentes sequências de bits. Dessa forma, uma mensagem digital é convertida em uma sequência de símbolos, que correspondem às sequências de bits da informação original, sendo então modulada sobre uma onda eletromagnética. No receptor, o sinal recebido passa por um processo de demodulação, que tem como objetivo recuperar a sequência de bits correspondente à mensagem enviada pelo transmissor. Em cenários em que o receptor recebe sinais provenientes de diferentes tipos de modulação, torna-se essencial identificar corretamente o tipo de modulação empregada no sinal, a fim de realizar a demodulação de forma precisa e eficiente.

Diante da heterogeneidade presente nas redes IoT e do uso de redes sem fio na conexão entre dispositivos, torna-se importante identificar padrões dos sinais transmitidos, como o tipo de modulação empregado, a fim de garantir a eficiência e a qualidade da comunicação. Para auxiliar nessa identificação, existem técnicas como o uso de algoritmos de aprendizado supervisionado do aprendizado de máquina.

No contexto de reconhecimento e classificação de sinais, os algoritmos de aprendizado supervisionado têm se mostrado uma abordagem eficiente para a identificação de características de sinais como o tipo de modulação. Esses algoritmos operam a partir de um conjunto de dados rotulados, composto, por exemplo, de sinais modulados, cada um

associado a um respectivo tipo de modulação. Durante a fase de treinamento, o modelo aprende a extrair e associar características relevantes dos sinais a suas classes correspondentes. Modelos baseados em redes neurais convolucionais são amplamente empregados nesse processo, devido à sua capacidade de capturar padrões espaciais e temporais presentes nos sinais. Uma vez treinado, o modelo é capaz de, a partir de um novo sinal de entrada, classificar de forma precisa qual técnica de modulação foi empregada na sua geração, contribuindo significativamente para a eficiência de sistemas de comunicação, especialmente em cenários heterogêneos, como em redes IoT.

Este trabalho tem como objetivo desenvolver um classificador baseado em redes neurais convolucionais capaz de identificar o tipo de modulação digital presente em sinais modulados semelhantes aos utilizados no padrão Wi-Fi (*Wireless Fidelity*) IEEE 802.11g. Esse padrão, além de empregar modulação digital, faz uso da técnica OFDM, que consiste na divisão da banda de transmissão em múltiplas subportadoras ortogonais, aumentando a eficiência espectral e a robustez contra interferências e atenuações no canal. O classificador proposto será treinado e validado utilizando tanto sinais gerados por simulação quanto sinais reais adquiridos por meio de equipamentos de rádio definido por software. Dessa maneira, uma solução como essa pode ser aplicada em redes IoT, especialmente naquelas que utilizam tecnologias baseadas em redes sem fio.

1 Fundamentação Teórica

Nesta seção, são apresentados os fundamentos teóricos relacionados à aplicação de técnicas de aprendizado de máquina na identificação da modulação em sinais de redes sem fio. Para isso, são abordados conceitos fundamentais sobre sistemas de comunicação e aprendizado de máquina, que sustentam o desenvolvimento da metodologia proposta.

1.1 Sistemas de Comunicação

Os sistemas de comunicação são formados por um conjunto de componentes responsáveis pela troca de informações entre dois ou mais dispositivos. De forma geral, esses sistemas são compostos pelos seguintes elementos: fonte, transmissor, canal, receptor e destino, os quais atuam de maneira coordenada para viabilizar o envio e o recebimento dos dados [1].

A fonte é o componente de onde se origina a mensagem. Alguns exemplos de fontes são a voz humana ou um texto de SMS (*Short Message Service*). Em alguns casos, quando a mensagem transmitida não é elétrica, é necessário o uso de um transdutor para converter essa mensagem para um sinal. Um exemplo de transdutor é o microfone que pode captar uma fala de voz humana. Analogamente, o destino é o componente em que a mensagem enviada será utilizada [1].

O transmissor é o componente responsável por modificar o sinal obtido com o transdutor para um formato eficiente para transmissão. Alguns exemplos de transmissores são o Conversor Digital-Analógico (DAC) e o modulador [1].

O canal é o meio pelo qual o sinal é propagado do transmissor até o receptor. Dentre os meios de propagação comumente utilizados como canais, destacam-se os fios elétricos e o ar [1].

O receptor é o componente responsável por receber o sinal transmitido e processá-lo para recuperar a mensagem enviada. Desta forma, utiliza-se um transdutor para transformar o sinal do meio elétrico para o digital. Como exemplo de receptores tem-se o demodulador e o conversor Analógico-Digital (ADC) [1].

Durante a transmissão de uma mensagem por um canal, podem ocorrer perturbações que afetam a integridade da informação original. Essas perturbações podem ser causadas tanto por ruído, que se origina de fenômenos aleatórios e inevitáveis do próprio sistema, quanto por interferências externas, provenientes de outros sinais ou dispositivos que compartilham o meio de transmissão. O ruído, especificamente, é considerado um dos fatores limitantes da taxa de transmissão, pois pode ocasionar erros na transmissão dos

dados e comprometer o desempenho do sistema [1].

Conforme mencionado anteriormente, o canal de transmissão dos sistemas de comunicação pode ser o ar. Nesse caso, o sistema é classificado como uma comunicação sem fio. As comunicações sem fio são fundamentais para proporcionar mobilidade, flexibilidade e escalabilidade, além de reduzir a necessidade da implementação de cabeamento entre os dispositivos Essas características tornam esse tipo de comunicação essencial em diversos cenários, especialmente nas redes IoT, onde há a necessidade de interconectar dispositivos distribuídos. Diante disso, o foco deste trabalho será direcionado às comunicações sem fio.

1.1.1 Comunicação sem fio

A comunicação sem fio foi uma das tecnologias que mais revolucionou a vida cotidiana nas últimas décadas, ao possibilitar a comunicação "em qualquer lugar e a qualquer momento". Além de seu papel primordial na telefonia móvel, essa tecnologia tem viabilizado avanços significativos em diversos setores, como automação industrial, redes de sensores, sistemas de monitoramento, transporte inteligente e, especialmente, nas redes de IoT, que depende da conectividade entre dispositivos distribuídos [2].

Para viabilizar a transmissão de informações nas comunicações sem fio, são utilizados os princípios da comunicação digital, que permitem representar dados em formato binário. A comunicação digital possibilita não apenas a codificação dos dados, mas também a aplicação de técnicas de modulação, detecção e correção de erros, essenciais para garantir a integridade das informações transmitidas. Dessa forma, entender os fundamentos da comunicação digital é imprescindível para compreender o funcionamento dos sistemas de comunicação sem fio e, consequentemente, das redes IoT e de outras aplicações modernas baseadas em conectividade [1], [3].

Na comunicação digital, as mensagens são representadas por combinações ordenadas de símbolos pertencentes a um alfabeto finito. Cada informação, seja texto, áudio ou vídeo, é convertida em sequências de bits, nas quais cada sequência de bits representa um símbolo do alfabeto. Por exemplo, um texto digitado em um editor utiliza um alfabeto composto por 128 símbolos, conforme a codificação ASCII, que associa uma sequência específica de bits a cada caractere. Assim, uma mensagem digital pode ser caracterizada pela quantidade M de símbolos distintos em seu alfabeto, sendo, portanto, classificada como uma mensagem M-ária. Cada símbolo desse alfabeto possui uma correspondência única com uma sequência de bits, estabelecendo uma relação direta entre os símbolos e as sequências binárias. Dessa forma, uma mensagem pode ser mapeada para uma sequência de símbolos M-ários, o que é fundamental para os processos de modulação digital e transmissão dos dados nos sistemas de comunicação [1], [3].

Para que seja possível a transmissão da mensagem por meio do canal, os símbolos da

mensagem digital são convertidos em sinais de ondas eletromagnéticas, cujas características variam de acordo com cada símbolo, conforme a técnica de modulação empregada. Esses sinais são então transmitidos pelo canal de comunicação até o receptor, onde ocorre o processo inverso, no qual as ondas recebidas são demoduladas e convertidas novamente em símbolos digitais, possibilitando a recuperação da mensagem original [3].

No uso de símbolos digitais em redes sem fio, diversos fundamentos e processos são essenciais para garantir a correta transmissão, recepção e integridade dos dados. Esses conceitos serão aprofundados nas próximas seções.

1.1.1.1 Taxas de Transmissão de Símbolo e de Bit

Nos sistemas de comunicação digital, as mensagens podem ser representadas como uma sequência temporal de símbolos, em que cada símbolo possui uma duração de T_s segundos. Desta forma, um sinal digital que transmite um conjunto de sinais que possui M símbolos, precisará de um número de bits n_b para sua representação dado por

$$n_b = log_2(M), \tag{1.1}$$

onde cada bit terá uma duração de T_b segundos. Portanto, o tempo de duração de um símbolo pode ser obtido por [4]:

$$T_s = log_2(M) \cdot T_b. \tag{1.2}$$

Ao saber o tempo de duração de cada símbolo durante a transmissão do sinal digital, é possível calcular a taxa de transmissão de símbolos da seguinte forma:

$$R_s = \frac{1}{T_s}. \tag{1.3}$$

Além disso, a taxa de transmissão de símbolos pode ser obtida relacionando as expressões (1.2) e (1.3).

1.1.1.2 Características do Sinal

Em um sistema de comunicação digital, existem algumas características importantes relacionadas aos sinais digitais utilizados nas redes sem fio. A seguir serão apresentadas algumas dessas características que são importantes para a compreensão do funcionamento dos sistemas de comunicação digitais.

Um sinal que é transmitido por um sistema digital através de um canal irá apresentar um medida de energia e potência do sinal. Essas duas medidas são importantes para o desempenho da comunicação [4].

A energia do sinal de um sinal r pode ser calculada por [4]:

$$E_r = \int_{-T_r/2}^{T_r/2} r^2(t)\, dt \tag{1.4}$$

onde E_r é a energia do sinal, T_r é a duração do sinal e $r(t)$ é a amplitude do sinal no instante de tempo t.

A potência do sinal de um sinal r pode ser calculada por [4]:

$$P_r = \frac{1}{T_r} \int_{-T_r/2}^{T_r/2} r^2(t)\, dt \tag{1.5}$$

onde P_r é a potência do sinal, T_r é a duração do sinal e $r(t)$ é a amplitude do sinal no instante de tempo t.

Ao aumentar a potência do sinal, os efeitos de sinais ruidosos de menor potência resultam em uma menor interferência no sinal. Essa característica é medida através da Relação Sinal-Ruído (*SNR*). Assim, a *SNR* pode ser expressa por [1]:

$$SNR = \frac{P_S}{P_N}, \tag{1.6}$$

onde P_S é a potência do sinal e P_N é a potência do ruído.

A *SNR* é uma característica que limita a transmissão da informação em um canal. A *SNR* mede a qualidade do sinal transmitido. Desta forma, conforme a *SNR* é maior, maior é a potência do sinal com relação a potência do ruído [1].

Outra métrica amplamente utilizada para avaliar e comparar o desempenho de sistemas de comunicação é a relação E_b/N_0, que pode ser interpretada como uma forma normalizada da razão sinal-ruído (SNR). Essa relação expressa a quantidade de energia média por bit (E_b) em relação à densidade espectral de potência do ruído (N_0), sendo particularmente útil para comparar sistemas com diferentes taxas de transmissão ou diferentes formatos de modulação. A relação E_b/N_0 é definida como [4]:

$$\frac{E_b}{N_0} = SNR \cdot \frac{B_w}{R_b}, \tag{1.7}$$

onde E_b é a energia de bit, N_0 é densidade espectral do ruído, B_w é a largura de banda e R_b é a taxa de transmissão de bits.

1.1.1.3 Características do Canal

Em um sistema de comunicação digital, existem algumas características importantes relacionadas aos canais. Entre elas tem-se a largura de banda e a capacidade máxima [1].

Um sinal ao passar por um canal pode sofrer atenuação ou distorção. No caso de sofrer atenuação, o sinal recebido possui uma energia menor que possuía no transmissor. Isso ocorre devido às perdas de energia durante o percurso, que normalmente é maior em canais com distância cada vez maiores. Agora, no caso de sofrer distorção, o sinal recebido apresenta uma forma diferente que o transmitido, na qual uma das causas desse efeito é a presença de ruído [4].

A largura de banda é o intervalo de frequência em que pode-se transmitir pelo canal com uma boa qualidade, isto é, sem apresentar uma grande perda de potência [1]. Dessa forma, os canais podem distorcer as formas de onda conforme se propaga na direção do receptor. Isso ocorre pois os canais não conseguem passar todas faixas de frequência, fazendo que sinais sejam atenuados em certas frequências, isto é, distorcendo-os [4].

Com a definição das faixas de frequência em que o canal realiza a atenuação, é possível definir os canais como banda-base ou banda-passante. Os canais banda-base tem como característica atenuar as frequências mais altas e preservar na maneira do possível as mais baixas. Logo, podem ser modelados com filtros passa-baixa. Em contrapartida, os canais modelados como banda-passante, isto é, filtros passa-faixa, preservam o sinal em uma faixa de frequência e atenuam nas demais [4].

Em consequência ao efeito de filtragem, as formas ondas dos símbolos transmitidos sofrem um alargamento, se sobrepondo em alguns instantes de tempo. Essa sobreposição é denominada Interferência Intersimbólica (ISI) [4].

Outro fator que pode levar a presença de ISI é o canal apresentar multipercursos. Nesses casos, o sinal transmitido é recebido na antena receptora após passar por diferentes percursos causados pela dispersão e refração em objetos presentes no caminho como prédios e árvores nas comunicações sem fio. Ao chegar no receptor, esses sinais irão se adicionar construtivamente ou destrutivamente, causando a ISI [4].

Essas características de perturbações presentes no canal como a ISI e o ruído interferem na capacidade máxima de informação que pode ser transmitida. O ruído mais comum encontrado em canais é o AWGN (*Additive white Gaussian noise*). Esse ruído é modelado através de um sinal aleatório com densidade espectral de potência constante em todas as frequências. Através desse modelo, foi criada a equação de Shannon para estabelecer a capacidade máxima do canal. A equação de Shannon é definida como [1]:

$$C = B \cdot log_2(1 + SNR), \tag{1.8}$$

onde C é a capacidade máxima do canal em bits/s, B é a largura de banda e SNR é a relação sinal-ruído.

Com o uso da equação de Shannon consegue-se encontrar um limite superior da taxa de transmissão de um canal. Se a taxa de transmissão superar esse limite, a iminência de erros durante a transmissão irá aumentar [1].

1.1.1.4 Largura de Banda

Conforme mencionado anteriormente, a largura de banda é um fator fundamental nos sistemas de comunicação digital, pois define o intervalo de frequências no qual a transmissão pode ocorrer com qualidade satisfatória. Assim, a largura de banda está

diretamente relacionada às características do sistema de comunicação digital que está sendo utilizado [3].

Inicialmente, é importante destacar que os símbolos de informação digital são representados no domínio do tempo discreto. No entanto, em muitos casos, busca-se transmitir sinais originalmente analógicos, como a gravação da voz. Para viabilizar essa transmissão por sistemas digitais, são empregados conversores ADC e DAC, responsáveis pela conversão entre os domínios analógico e digital. Esse processo envolve etapas fundamentais, como amostragem, quantização e codificação, que garantem a representação precisa do sinal analógico no formato digital e, posteriormente, sua reconstrução no receptor [4].

O processo de amostragem consiste em coletar os valores da amplitude de um sinal contínuo no tempo, $s(t)$, em intervalos regulares de T_{sa} segundos, armazenando esses valores em uma sequência discreta [4]. Matematicamente, a amostragem é representada por:

$$s_n = s(nT_{sa}) \tag{1.9}$$

onde n é o índice da amostra, T_{sa} é o período de amostragem, que resulta em uma frequência de amostragem $f_{sa} = 1/T_{sa}$. Esse processo permite representar o sinal contínuo por uma sequência discreta de valores no tempo.

Para que a amostragem represente corretamente um sinal $s(t)$ que possui frequência máxima f_m, é necessário que a frequência de amostragem f_{sa} seja

$$f_{sa} > 2 \cdot f_m. \tag{1.10}$$

Esse resultado é conhecido como teorema de *Nyquist*. Essa inequação estabelece um limite inferior da frequência de amostragem para que a amostragem represente corretamente o sinal e que seja possível reconstruir novamente o sinal [4].

Tendo em vista o teorema de *Nyquist*, pode-se definir a largura de banda para sistemas digitais banda-base por:

$$B = \frac{R_s}{2}. \tag{1.11}$$

Além disso, a largura de banda para sistemas digitais banda-passante é definido por [4]: base por:

$$W = R_s. \tag{1.12}$$

1.1.2 Modelo de Canal de Rayleigh

Durante a transmissão de um sinal, pode ocorrer no canal de propagação, a reflexão do sinal em obstáculos contidos no trajeto. Esses objetos, como no caso do ar, podem ser prédios, árvores etc. Com isso, no receptor são recebidos diferentes sinais em diferentes tempos [5]. A Figura 1 exibe um exemplo de um canal com dois percursos distintos, um

obtido com a transmissão direta do sinal do transmissor até o receptor e o outro através da reflexão do sinal em um obstáculo.

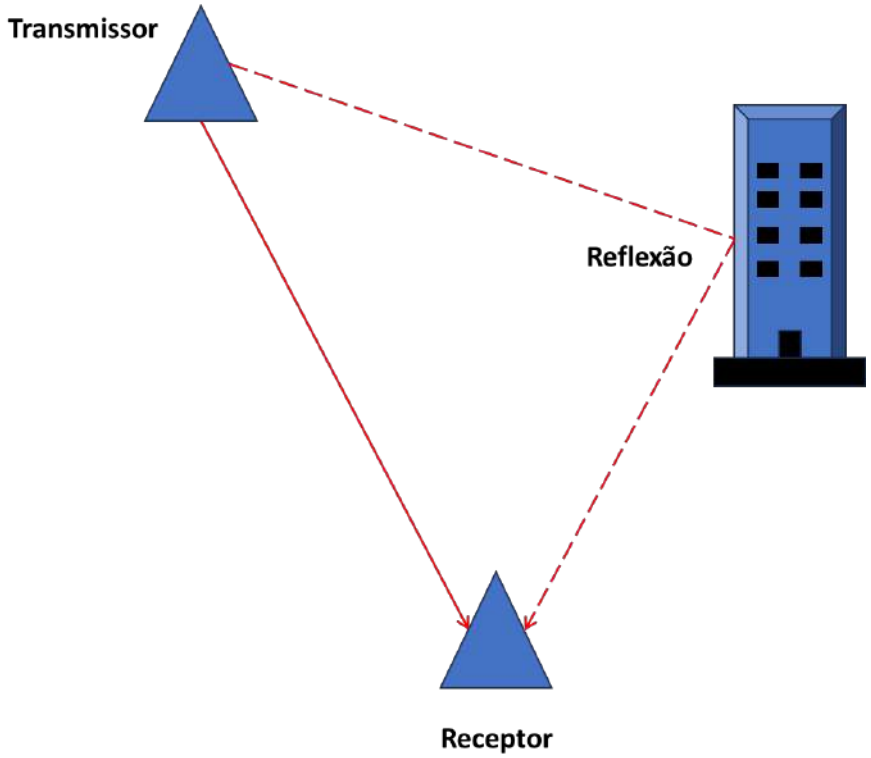

Figura 1 – Exemplo de canal de multipercursos.

Uma forma de representar esses canais é utilizando o modelo *Tapped Delay Line* (TDL) com distribuição de Rayleigh. Esse modelo representa a resposta ao impulso do canal da seguinte forma [6]:

$$h(t) = \frac{1}{\sqrt{n}} \left[h_1(t - t_1) + \cdots + h_n(t - t_n) \right], \tag{1.13}$$

onde n representa o número de multipercurso, $h_n(t - t_n)$ representa o coeficiente do canal do caminho n obtido no tempo t_n e $1/\sqrt{n}$ realiza uma normalização da energia do canal. Cada coeficiente segue uma distribuição estatística Rayleigh, na qual suas partes real e imaginária são variáveis aleatórias gaussianas independentes com média zero.

1.1.3 Modelo de Canal de Rice

O modelo de canal de Rice é semelhante ao modelo de Rayleigh, porém no modelo de Rice existe uma componente de percurso dominante entre todas componentes. Tal componente pode ser, por exemplo, a componente que representa a linha de visada entre os dispositivos presentes na comunicação [7].

Desta forma, a resposta a impulso do canal de Rice pode ser representada pela seguinte forma:

$$h(t) = \sqrt{\frac{K}{K+1}} \cdot h_0(t - t_0) + \sqrt{\frac{1}{K+1}} \cdot [h_1(t - t_1) + \cdots + h_n(t - t_n)], \tag{1.14}$$

onde n representa o número de multipercurso, $h_n(t - t_n)$ representa o coeficiente complexo do canal do caminho n obtido no tempo t_n, h_0 representa a componente dominante e K é o fator do modelo de Rice. Cada coeficiente associado às trajetórias difusas h_i é modelado como uma variável aleatória complexa, cujas partes real e imaginária seguem distribuições gaussianas independentes com média zero. Por outro lado, a componente de linha de visada h_0 é modelada, dependendo do cenário considerado, como uma constante determinística ou como uma exponencial complexa com fase variável [7].

O fator K do modelo de Rice é a razão entre a potência no caminho da componente predominante e a potência nos outros caminhos. Esse fator é importante para mensurar o quanto a componente principal é predominante no modelo [7].

1.1.4 Representação de Sinais Digitais

Na transmissão de mensagens digitais em sistemas que utilizam um alfabeto com M símbolos distintos, a sequência de bits da mensagem é mapeada para uma sequência correspondente de símbolos do alfabeto. Cada símbolo, por sua vez, é convertido em uma forma de onda específica, cuja característica depende tanto do valor do símbolo quanto do esquema de modulação adotado. Esses símbolos normalmente são representados por pontos em um espaço vetorial. Esta representação ajuda na visualização de análises de desempenho do sistema [4].

Para a representação do espaço vetorial, é utilizada a geometria euclidiana, em que um símbolo digital é expresso como uma combinação linear de N formas de ondas ortonormais. Esse conjunto representa um espaço vetorial de N dimensões [4].

Considere uma base ortonormal de N dimensões composta por ϕ de N formas de ondas diferentes, isto é, $\Phi = \{\phi_1(t), ..., \phi_N(t)\}$. Um sinal $s_m(t)$ pode ser representado pela combinação linear [4], ou seja,

$$s_m(t) = \sum_{i=1}^{N} s_{m,i} \cdot \phi_i(t), \tag{1.15}$$

onde $s_{m,1}$,..., $s_{m,N}$ são os escalares utilizados na combinação.

Algumas propriedades de bases ortogonais são importantes para a representação de sinais. A primeira propriedade é que o produto interno entre dois componentes de uma base ortogonal sempre resulta em zero. Portanto,

$$\langle \phi_i(t), \phi_j(t) \rangle = \int_{t_0}^{t_0+T_s} \phi_i(t) \cdot \phi_j^*(t) dt = 0, \tag{1.16}$$

onde $\phi_j^*(t)$ é o complexo conjugado de $\phi_j(t)$. A segunda propriedade é a normalização da energia unitária dos elementos da base, ou seja

$$E_i = \langle \phi_i(t), \phi_i(t) \rangle = 1. \tag{1.17}$$

Essas propriedades são importantes para a criação de uma base ortonormal para ser utilizada na representação dos sinais [4].

Os coeficientes de pesos da equação (1.27) são encontrados através da projeção da forma de onda $s_m(t)$ em cada um dos componentes da base Φ [4]. Logo,

$$s_{m,i} = \langle s_m(t), \phi_i(t) \rangle = \int_{t_0}^{t_0+T_s} s_m(t) \cdot \phi_i^*(t) dt = 0. \tag{1.18}$$

Durante a análise de desempenhos de sistema, normalmente, utiliza-se a correlação e distância entre símbolos no espaço vetorial. A correlação é dada por [4]:

$$E_{s_i,s_j} = \langle s_i, s_j \rangle = \sum_{n=1}^{N} s_{i,n} \cdot s_{j,n}^*. \tag{1.19}$$

Já a distância entre sinais é dada por [4]:

$$d_{s_i,s_j} = |s_i - s_j| = \sqrt{\sum_{n=1}^{N} |s_{i,n} - s_{j,n}|^2}. \tag{1.20}$$

Assim, como visto anteriormente, as formas de ondas podem ser representadas por coordenadas em um espaço vetorial. Para obter esse espaço vetorial e as coordenadas para os sinais pode-se utilizar o processo de Gram–Schmidt.

1.1.5 Processo de Gram–Schmidt

O processo de Gram-Schmidt é utilizado para definir um espaço vetorial para um conjunto de M formas de onda $s_m(t)$ que possuem energias E_{s_m}. Essas energias podem ser medidas por:

$$E_{s_m} = |\boldsymbol{s_m}|^2 = \sum_{n=1}^{N} |s_m[n]|^2. \tag{1.21}$$

Com isso, o processo de Gram-Schmidt encontra um espaço vetorial ortonormal de N dimensões, composto por $N \leq M$ formas de ondas $\phi_n(t)$ que podem representar o conjunto de sinais [4].

Para encontrar o espaço vetorial ortonormal de N dimensões pelo processo de Gram-Schmidt deve-se seguir os seguintes passos [4]:

1. Para encontrar $\phi_1(t)$, considere que $\phi_1(t) = s_1(t)$ normalizado e com energia unitária:

$$\phi_1(t) = \frac{s_1(t)}{\sqrt{E_{s_1}}}. \tag{1.22}$$

2. Para encontrar $\phi_2(t)$, considere que

$$g_2(t) = s_2(t) - s_{2,1} \cdot \phi_1(t), \tag{1.23}$$

$$\phi_2(t) = \frac{g_2(t)}{\sqrt{E_{g_2}}}, \tag{1.24}$$

onde $s_{2,1} = \langle s_2(t), \phi_1(t) \rangle$.

3. Para os demais $\phi_n(t)$, considere que

$$g_n(t) = s_n(t) - \sum_{j=1}^{n-1} s_{n,j} \cdot \phi_j(t), \tag{1.25}$$

$$\phi_n(t) = \frac{g_n(t)}{\sqrt{E_{g_n}}}, \tag{1.26}$$

onde $s_{n,j} = \langle s_n(t), \phi_j(t) \rangle$.

Por fim, quando a base utilizada na representação dos sinais é conhecida, é comum empregar a notação vetorial para representar o sinal de forma mais compacta:

$$\boldsymbol{s_m} = [s_{m,1}, ..., s_{m,N}]^T. \tag{1.27}$$

1.1.6 Representação por Codificação de Linha

A codificação de linha foi desenvolvida para realizar a transmissão de dados digitais através da transformação da sequência para pulsos elétricos. Desta forma, qualquer forma de onda de codificação de linha pode ser representada pelo seguinte sinal de modulação de amplitude de pulso M-ário (M-PAM) [4]:

$$s_{M-PAM}(t) = \sum_i d'_i \cdot h_p(t - i \cdot T_s), \tag{1.28}$$

onde d'_i é o valor real do i-ésimo símbolo de informação de uma codificação e h_p é a forma do pulso.

Considerando que os símbolos de informação também possuem valores complexos, a representação pode ser feita como

$$s_{LPE}(t) = \sum_i d_i \cdot h_p(t - i \cdot T_s), \tag{1.29}$$

onde d_i é o valor complexo do i-ésimo símbolo de informação de uma codificação. Essa representação é conhecida como LPE (*Lowpass Equivalent*), que é simples e útil para representar sinais em ambientes computacionais [4].

1.1.7 Modulação de Sinais

Durante o processo de transmissão de mensagens digitais, é comum a utilização de técnicas de modulação. Nesse contexto, o sinal de banda base, que representa a mensagem digital, é utilizado para modificar algum parâmetro de uma onda portadora de radiofrequência, como a amplitude, a frequência ou a fase. Esse tipo de abordagem é denominado modulação em banda passante [1].

A onda portadora é uma senóide de alta frequência que é modificada de acordo com os símbolos que serão transmitidos. Entre os parâmetros que podem ser modificados estão a amplitude, a frequência e a fase do sinal [1]. Dessa forma, pode-se definir um sinal modulado da seguinte forma:

$$s(t) = A(t) \cdot cos[W(t)\omega_o t + \theta(t)], \tag{1.30}$$

onde $A(t)$ é uma função para se modular a amplitude da onda, $W(t)$ é uma função que varia a frequência angular, $\omega_o = 2\pi f_0$ é a frequência angular da onda e $\phi(t)$ é a variação de fase da onda.

Em contrapartida, na modulação em banda-base, o sinal digital é transmitido diretamente, sem a necessidade de uma portadora de alta frequência. Nessa abordagem, os símbolos digitais são mapeados em formas de onda de baixa frequência, com espectro concentrado em torno da frequência zero. Além de simplificar a implementação e reduzir a complexidade do sistema, a modulação em banda-base é amplamente utilizada em etapas internas do processamento de sinais digitais, como na geração de símbolos. Assim, os sinais de banda base podem ser representados por meio da forma LPE, conforme apresentado na Equação 1.29, na qual os símbolos d_i são números complexos que codificam simultaneamente a amplitude e a fase de cada símbolo transmitido. Neste trabalho, opta-se pela utilização da modulação banda-base justamente por sua maior facilidade de manipulação e análise em ambientes simulados.

Existem diferentes tipos de modulação. Na Figura 2 pode ser visto um exemplo de modulação em amplitude e outro de modulação em frequência.

Além do processo de modulação, existe o processo inverso conhecido como demodulação. Nesse processo um sinal modulado é transformado para um sinal de banda base [1].

1.1.7.1 Modulação ASK

A modulação ASK (*Amplitude Shift Keying*) consiste na modificação da amplitude do sinal modulado de acordo com os símbolos da mensagem. Assim, ao ter M símbolos, a amplitude da portadora pode assumir M amplitudes diferentes [1]. Um exemplo de modulação ASK pode ser visto na Figura 3, que exibe o diagrama de constelação para o

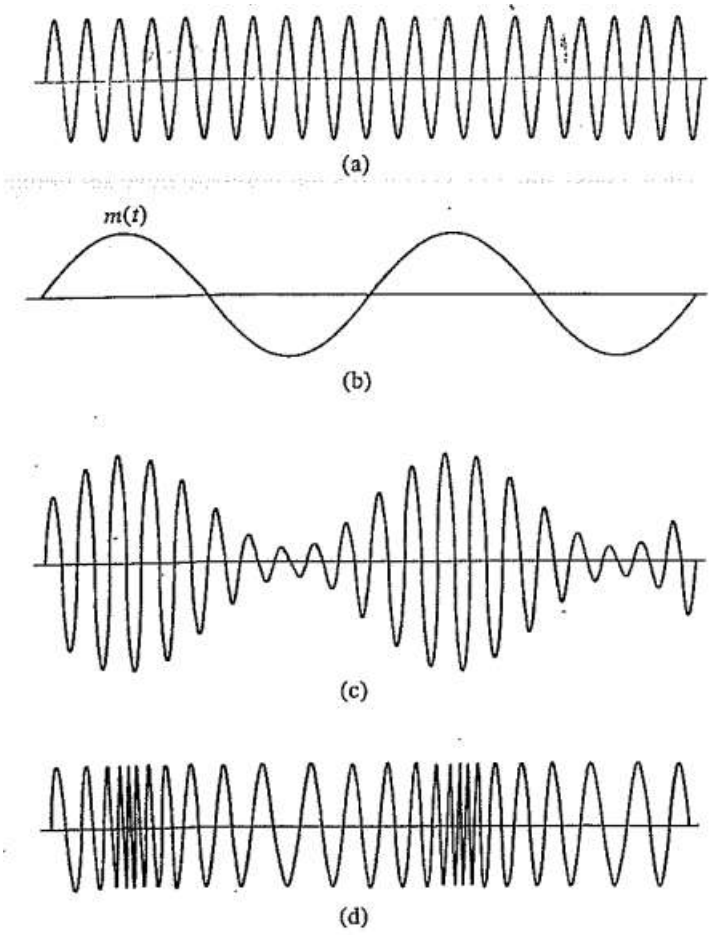

Figura 2 – Exemplos de aplicação de modulações: (a) onda portadora; (b) sinal banda base; (c) sinal modulado em amplitude; (d) sinal modulado em frequência. Fonte: [1].

tipo modulação OOK (*On–Off Keying*). O diagrama de constelação exibe os símbolos da modulação em um espaço vetorial ortonormal.

Um sinal com modulação ASK pode ser representado pela seguinte expressão:

$$s_{ASK}(t) = A(t)cos[\omega_0 t], \tag{1.31}$$

onde $s(t)$ é o sinal modulado, $A(t)$ é uma função que altera o valor da amplitude de acordo com a mensagem $m(t)$ e ω_0 é a frequência angular da portadora. Por exemplo, para o caso do BASK (*Binary amplitude shift keying*) conhecido como OOK, tem-se a seguinte expressão para a forma de onda [4]:

$$s_{OOK}(t) = \sum_i s^i_{OOK}(t), \tag{1.32}$$

onde i-ésima forma de onda é dada pelo símbolo d_i de duração T_s representado como

$$s^i_{OOK}(t) = \begin{cases} 0 & \text{para } d_i = 0 \\ \sqrt{\dfrac{2 \cdot E_s}{Ts}} \cdot cos(\omega_0 t), i \cdot T_s \leq t \leq (i+1) \cdot T_s & \text{para } d_i = 1. \end{cases} \tag{1.33}$$

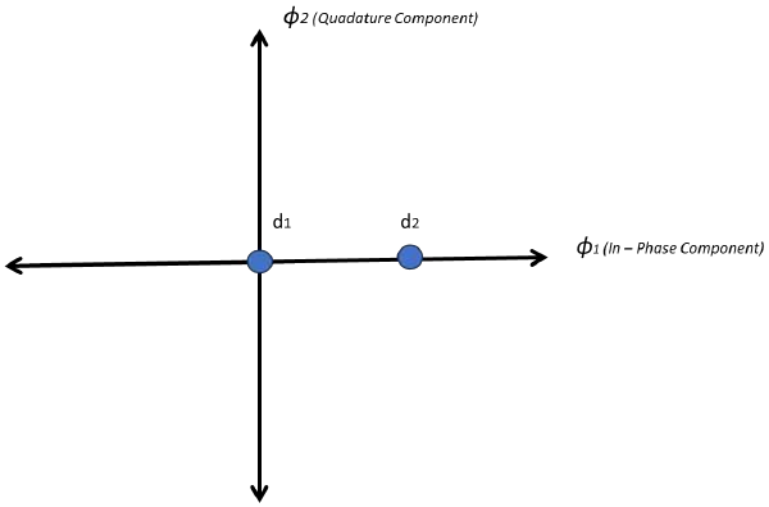

Figura 3 – Diagrama de constelação da modulação OOK.

Com o processo de Gram-Schmidt obtém-se a seguinte base ortonormal [4]:

$$\phi^i(t) = \sqrt{\frac{2}{T_s}} \cdot cos(\omega_o t), i \cdot T_s \leq t \leq (i+1) \cdot Ts. \tag{1.34}$$

Além disso, nesse caso, tem-se na representação LPE que [4]

$$s_{LPE}(t) = \sum_i d_i \cdot \Pi\left(\frac{t - i \cdot T_s}{T_s}\right), \tag{1.35}$$

onde Π é o pulso retangular e $d_i \in \{0, \sqrt{E_s}\}$.

1.1.7.2 Modulação FSK

A modulação FSK (*Frequency Shift Keying*) consiste na modificação da frequência da portadora de acordo com os símbolos da mensagem a ser modulada. Assim, ao ter M símbolos, a frequência da portadora pode assumir M frequências diferentes [1]. Um exemplo de modulação FSK pode ser visto na Figura 4 que exibe o diagrama de constelação para o tipo modulação 2-FSK.

Um sinal com modulação FSK pode ser representado pela seguinte expressão:

$$s(t) = Acos[W(t)(\omega_0)t], \tag{1.36}$$

onde $s(t)$ é o sinal modulado, A é a amplitude da onda portadora, $W(t)$ é uma função que varia a frequência angular de acordo com a mensagem a ser transmitida $m(t)$, e ω_0 é a frequência angular da portadora.

Assim, dado os M símbolos d_i de uma modulação M-FSK, a i-ésima forma de onda é dada pelo símbolo d_i de duração T_s representado como [4]:

$$s^i_{M-FSK}(t) = \sqrt{\frac{2 \cdot E_s}{Ts}} \cdot cos(2\pi f_i t), i \cdot T_s \leq t \leq (i+1) \cdot T_s, \tag{1.37}$$

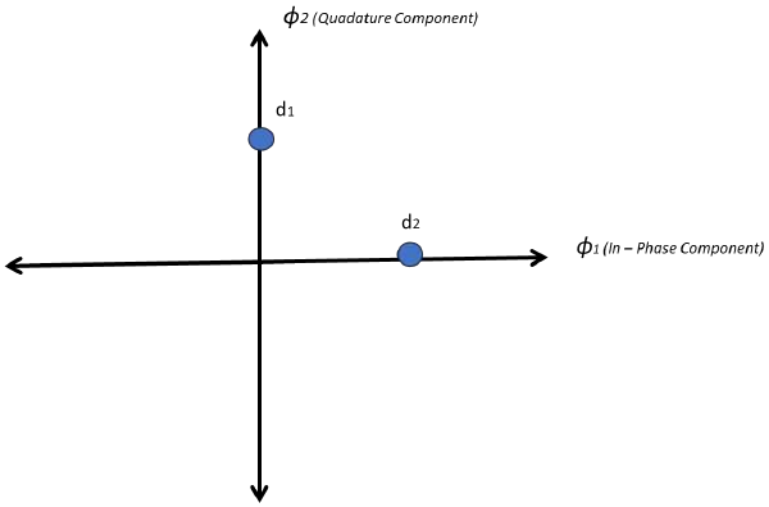

Figura 4 – Diagrama de constelação da modulação 2-FSK.

onde $f_i \in \{f_0 + m/(2 \cdot T_s)\}$, $f_0 = N_{f_0}/(2 \cdot T_s)$, $N_{f_0} \in \mathbb{N}$ e $m = 1, .., M$.

Com o processo de Gram-Schmidt obtém-se a seguinte base ortonormal [4]:

$$\phi_m^i(t) = \sqrt{\frac{2}{T_s}} \cdot cos(2\pi f_i t), 0 \leq t \leq Ts. \tag{1.38}$$

Na representação LPE, os sinais são representados por [4]:

$$s_{LPE}(t) = \sum_i d_i \cdot \Pi\left(\frac{t - i \cdot T_s}{T_s}\right), \tag{1.39}$$

onde

$$d_i \in \left\{\sqrt{E_s} \cdot cos\left(\frac{\pi \cdot m \cdot t}{2 \cdot T_s}\right) + j\sqrt{E_s} \cdot sen\left(\frac{\pi \cdot m \cdot t}{2 \cdot T_s}\right)\right\}, m = 1, .., M. \tag{1.40}$$

1.1.7.3 Modulação PSK

A modulação PSK (*Phase Shift Keying*) consiste na modificação da fase da portadora de acordo com os símbolos da mensagem a ser modulada. Assim, ao ter M símbolos, existirão M modificações diferentes de fase na onda portadora [1]. Um exemplo de modulação PSK pode ser visto na Figura 5 que exibe o diagrama de constelação para o tipo modulação QPSK (*Quaternary Phase Shift Keying* - 4-PSK).

Um sinal com modulação PSK pode ser representado pela seguinte expressão:

$$s(t) = Acos[\omega_0 t + \theta(t)],$$

onde $s(t)$ é o sinal modulado, A é a amplitude da onda portadora, ω_0 é a frequência angular da portadora e $\theta(t)$ é uma função que varia a fase de acordo com a mensagem a ser transmitida $m(t)$.

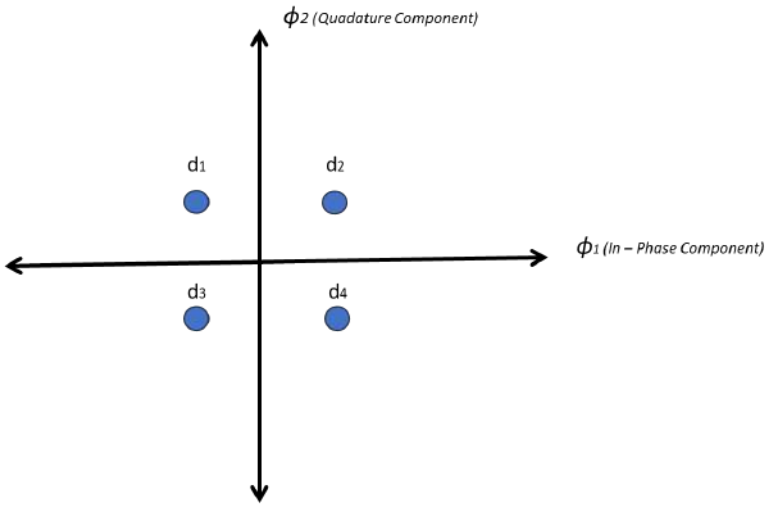

Figura 5 – Diagrama de constelação da modulação QPSK.

Assim, dado os M símbolos d_i de uma modulação M-PSK, a i-ésima forma de onda é dada pelo símbolo d_i de duração T_s representado como [4]:

$$s^i_{M-PSK}(t) = \sqrt{\frac{E_s}{Ts}} \cdot cos(\omega_0 t - \theta_i), i \cdot T_s \leq t \leq (i+1) \cdot T_s, \tag{1.41}$$

onde $\theta_i \in \left\{ \frac{2 \cdot \pi \cdot (m-1)}{M} \right\}$, $m = 1, ..., M$.

Nessa modulação, estão presentes dois componentes para a base ortonormal [4]:

$$\phi^i_1(t) = \sqrt{\frac{2}{T_s}} \cdot cos(2\pi f_0 t), i \cdot Ts \leq t \leq \cdot (i+1) \cdot Ts \tag{1.42}$$

$$\phi^i_2(t) = \sqrt{\frac{2}{T_s}} \cdot sin(2\pi f_0 t), i \cdot Ts \leq t \leq \cdot (i+1) \cdot Ts. \tag{1.43}$$

Na representação LPE, os sinais são representados por [4]:

$$s_{LPE}(t) = \sum_i d_i \cdot \Pi \left(\frac{t - i \cdot T_s}{T_s} \right), \tag{1.44}$$

onde $d_i \in \left\{ a_m \sqrt{E_s} + j b_m \cdot \sqrt{E_s} \right\}$, $\sqrt{a_m^2 + b_m^2} = 1$, $a_m = cos(2\pi \cdot (m-1)/M)$, $b_m = sen(2\pi \cdot (m-1)/M)$ e $m = 1, .., M$.

1.1.7.4 Modulação QAM

A modulação QAM (*Quadrature Amplitude Modulation*) consiste em uma combinação das modulações ASK e PSK. Nessa modulação, inicialmente, a portadora recebe uma amplitude Q de acordo com o símbolo a ser transmitido. Logo após, a portadora é defasada em 90^o e recebe uma outra amplitude I de acordo com o símbolo a ser transmitido. Por

fim, as duas ondas são somadas para formar o sinal modulado QAM [1]. Portanto, um sinal modulado QAM pode ser representado da seguinte maneira:

$$s(t) = Q(t)cos[\omega_c t] + I(t)sen[\omega_c t], \tag{1.45}$$

onde $s(t)$ é o sinal modulado, $Q(t)$ e $I(t)$ são as amplitudes definidas de acordo com o símbolo a ser transmitido na mensagem m(t) e ω_c é a frequência angular da portadora. Um exemplo de modulação QAM pode ser visto na Figura 6 que exibe o diagrama de constelação para o tipo modulação 16-QAM.

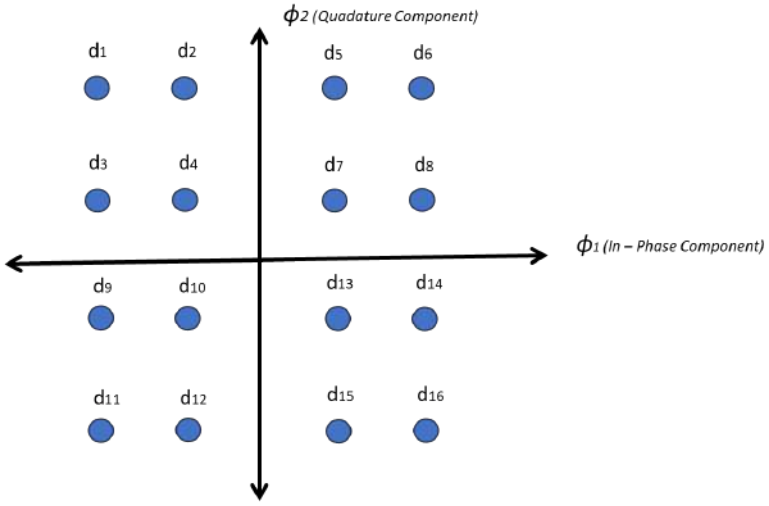

Figura 6 – Diagrama de constelação da modulação 16-QAM.

Assim, dado os M símbolos d_i de uma modulação M-QAM, a i-ésima forma de onda é dada pelo símbolo d_i de duração T_s representado como [4]:

$$s^i_{M-QAM}(t) = \sqrt{\frac{2 \cdot E_i}{Ts}} \cdot cos(\omega_0 t - \theta_i), i \leq t \leq i \cdot T_s, \tag{1.46}$$

onde $\theta_i \in \{\theta_m\}$, $E_i \in \{E_m\}$ e $m = 1, ..., M$.

Nessa modulação, estão presentes dois componentes para a base ortonormal iguais aos da modulação M-PSK [4]:

$$\phi^i_1(t) = \sqrt{\frac{2}{T_s}} \cdot cos(2\pi f_0 t), i \cdot Ts \leq t \leq \cdot(i+1) \cdot Ts \tag{1.47}$$

$$\phi^i_2(t) = \sqrt{\frac{2}{T_s}} \cdot sin(2\pi f_0 t), i \cdot Ts \leq t \leq \cdot(i+1) \cdot Ts. \tag{1.48}$$

Na representação LPE, os sinais são representados por [4]:

$$s_{LPE}(t) = \sum_i d_i \cdot \Pi\left(\frac{t - i \cdot T_s}{T_s}\right), \tag{1.49}$$

onde $d_i \in \left\{a_m\sqrt{E_{min}} + jb_m \cdot \sqrt{E_{min}}\right\}$, a_m e $b_m \in \left\{-\sqrt{M}+1, -\sqrt{M}+3, ..., \sqrt{M}+1\right\}$, E_{min} é a energia do símbolo com menor energia e $m = 1, .., M$.

1.1.8 Critério de Nyquist e o Pulso Cosseno Levantado

Na seção anterior foi visto que os símbolos modulados na representação LPE podem ser escritos como [4]:

$$s_{LPE}(t) = \sum_i d_i \cdot h_p\left(t - i \cdot T_s\right), \tag{1.50}$$

onde d_i é o valor do i-ésimo símbolo digital, h_p é a forma do pulso e T_s é o tempo de duração de cada símbolo.

Um receptor recebe esse sinal, amostrando-o em intervalos de T_s tem-se que a n-ésima amostra é obtida por

$$s_{LPE}(nT_s) = \sum_i d_i \cdot h_p\left(nT_s - i \cdot T_s\right). \tag{1.51}$$

Essa equação pode ser decomposta da seguinte maneira [4]:

$$s_{LPE}(nT_s) = d_n \cdot h_p(0) + \sum_{i \neq n} d_i \cdot h_p\left(nT_s - i \cdot T_s\right), \tag{1.52}$$

onde a primeira parte da equação representa o símbolo desejado e a segunda parte a interferência intersimbólica. Dessa forma, para não ter interferência intersimbólica é necessário que a segunda parte da equação resulte em zero [4]. Assim, a amostragem de h_p deve resultar no Delta de Kronecker, isto é,

$$h_p(nT_s) = \delta_{n,i} = \begin{cases} 1, & \text{se } n = i \\ 0, & \text{se } n \neq i. \end{cases} \tag{1.53}$$

Essa representação equivalente no domínio da frequência pode ser expressa por

$$\frac{1}{T_s} \cdot \sum_i H_p\left(f - \frac{i}{T_s}\right) = 1, \tag{1.54}$$

em que $H_p(f)$ é a transformada de Fourier de $h_p(t)$. Essa equação é denominada de critério de Nyquist.

O critério de Nyquist exibe a condição necessária para se obter uma ISI zero, em que um pulso h_p que satisfaz essa equação é denominado pulso de Nyquist. Portanto, o pulso de Nyquist utilizado garante que o sinal recuperado se amostrado em intervalos de T_s, mesmo existindo sobreposição entre pulsos vizinhos em intervalos diferentes do instante amostrado [4].

Note que nas representações LPE dos símbolos das modulações foi utilizado o pulso retangular. Esse pulso retangular tem como vantagem não possuir interferência entre as informações dos símbolos, por causa que não existe sobreposição dos espectros dos pulsos de cada símbolo. Em contrapartida, esse pulso não pode ser implementado em um sistema real devido a possuir um espectro que se estende ao infinito como pode ser visto na figura 7. Portanto, precisaria de uma largura de banda infinita [8].

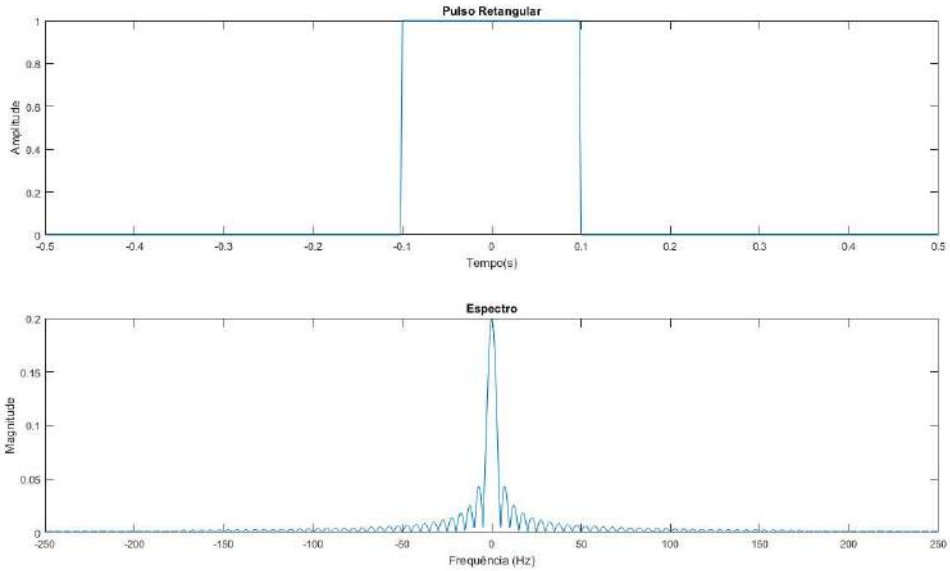

Figura 7 – Pulso retangular no domínio do tempo e da frequência.

Uma alternativa para uso de pulso de Nyquist em transmissões banda base é o pulso cosseno levantado. O pulso cosseno levantado pode ser definido como

$$h_p(t) = \left(\frac{\sin(\pi t/T_s)}{\pi t/T_s}\right)\left(\frac{\cos(\alpha\pi t/T_s)}{1-(2\alpha t/T_s)^2}\right), \tag{1.55}$$

onde α é o fator de decaimento (*roll-off*) que indica o excesso de largura de banda que excede a banda do canal ideal. Esse pulso tem a seguinte transformada de Fourier [4]:

$$H_p(f) = \begin{cases} T_s, & \text{se } |f| \leq \dfrac{1-\alpha}{2T_s} \\ T_s\cos^2\left[\dfrac{\pi T_s}{2\alpha}\left(|f| - \dfrac{1-\alpha}{2T_s}\right)\right], & \text{se } \dfrac{1-\alpha}{2T_s} \leq |f| \leq \dfrac{1+\alpha}{2T_s} \\ 0, & \text{se } |f| > \dfrac{1+\alpha}{2T_s}. \end{cases} \tag{1.56}$$

Com o uso do pulso cosseno levantado, a largura de banda em sistemas banda base pode ser estimada por:

$$B = (1+\alpha)\cdot\frac{R_s}{2}. \tag{1.57}$$

Já para sistemas passa-faixa pode ser estimada por [4]:

$$W = (1+\alpha)\cdot R_s. \tag{1.58}$$

A figura 8 exibe o comportamento do pulso cosseno levantado no domínio do tempo e da frequência para diferentes valores para o fator de decaimento. Além disso, nota-se que esse pulso apresenta uma largura de banda finita, diferentemente do pulso retangular.

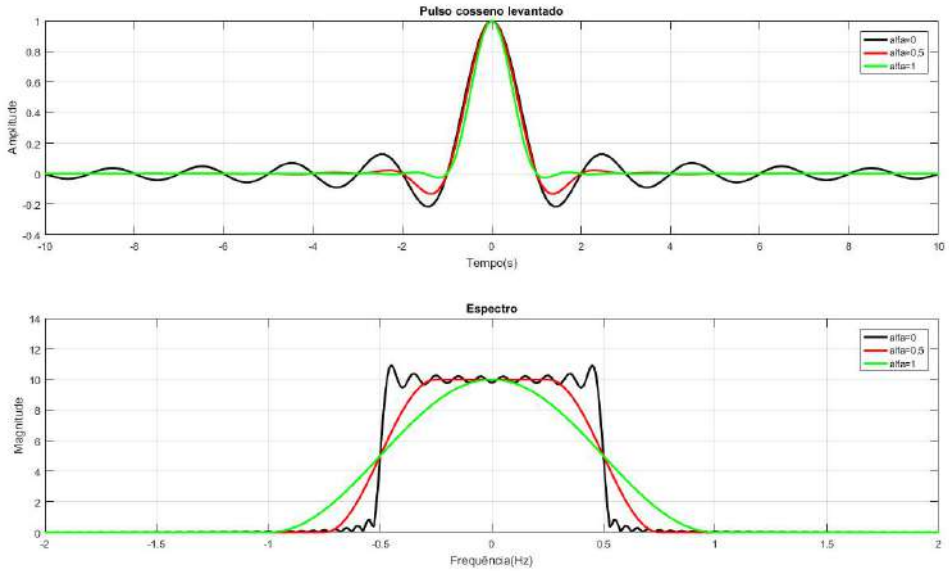

Figura 8 – Pulso cosseno levantado no domínio do tempo e da frequência.

1.1.9 Sincronização

No processo de transmissão e recepção de sinais é comum o envio das informações através de quadros (*frames*) contendo uma quantidade N_S de símbolos. Na recepção do sinal é importante o conhecimento do início e fim dos frames para que os processos de amostragem e demodulação dos símbolos sejam feitos corretamente [4].

Desta forma, uma maneira de realizar a sincronização é no domínio do tempo. A sincronização no domínio do tempo tem como principal objetivo conseguir sincronizar as amostras pertencentes a cada um dos símbolos do sinal recebido. Para isso, é necessário encontrar onde as amostras começam e terminam em cada um dos *frames* [4].

Uma maneira de executar a sincronização no domínio do tempo é utilizando um *frame* de sincronização que contém um código previamente conhecido pelo transmissor e pelo receptor que é utilizado para identificar o início dos *frames* de informação [9].

Uma das codificações mais utilizadas para sincronização é a sequência de Barker [10]. Esse código consiste em uma sequência binária formada por valores +1 e -1, projetada para apresentar uma autocorrelação tão próxima do ideal quanto possível. Isto é, a sequência é composta por N_{bk} elementos que [9]:

$$\left| \sum_{i=1}^{N_{bk}-k} a_i a_{i+k} \right| \leq 1, \tag{1.59}$$

onde $a_i \in [-1, 1]$, $i = 1, \ldots N_{bk}$ para todo $1 \leq i \leq N_{bk}$. Desta forma, os coeficientes de autocorrelação com um atraso diferente de 0 possuem valores bem baixos e com o atraso igual a zero possui um valor elevado de magnitude.

A figura 9 mostra um sinal $s(t)$ que contém a sequência de Barker de comprimento 11 e o resultado da sua autocorrelação. Note que a magnitude da autocorrelação é alta quando não foi aplicado um atraso e baixa com a aplicação de atrasos.

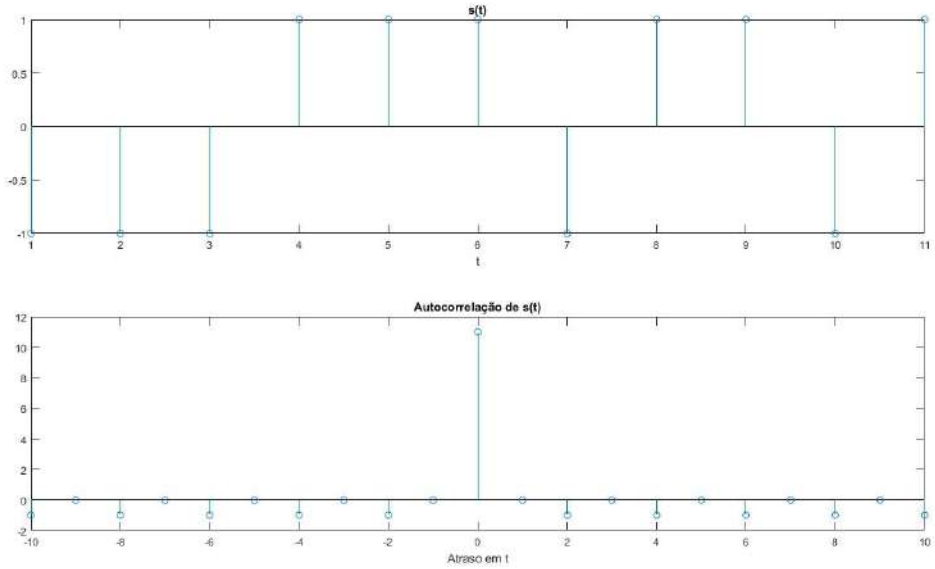

Figura 9 – Sequência de Barker e a sua autocorrelação.

Normalmente, a sequência de Barker em transmissões de sinais é modulada com o uso de BPSK. Assim, pode-se enviar essa sequência como um preâmbulo dos *frames* de informação que serão recebidos. Desta forma, o receptor do sinal identifica a posição que se encontra o preâmbulo com o uso da autocorrelação, descarta a fatia do sinal que contém o preâmbulo e realiza a leitura dos símbolos a partir da amostra seguinte do preâmbulo.

1.1.10 Sistemas Multiportadora

Os sistemas convencionais que possuem uma única portadora possuem limitações causadas principalmente pela ISI durante uma transmissão. Desta forma, alguns sistemas exploram o uso de múltiplas portadoras em um mesmo canal de transmissão. Com o uso de mais de uma portadora, a largura de banda disponível é dividida em subcanais que serão utilizados para efetuar uma transmissão em paralelo. Além disso, devido a propriedade de paralelismo, a taxa de transmissão de símbolos pode ser reduzida, acarretando em um tempo de duração de símbolo maior, reduzindo assim a ISI [11].

Na Figura 10 é mostrado a comparação entre a transmissão de 3 símbolos em um sistema com portadora única e outro com múltiplas portadoras. Note que no sistema com uma única portadora, os símbolos são transmitidos serialmente com um tempo de duração de símbolo T_s. Em contrapartida, no sistema multiportadora, os três símbolos são transmitidos em paralelo com um tempo de duração de símbolo três vezes maior. Dessa

forma, a ISI se torna menor, por causa que o período de transmissão dos símbolos se torna muito maior que a dispersão de tempo do canal.

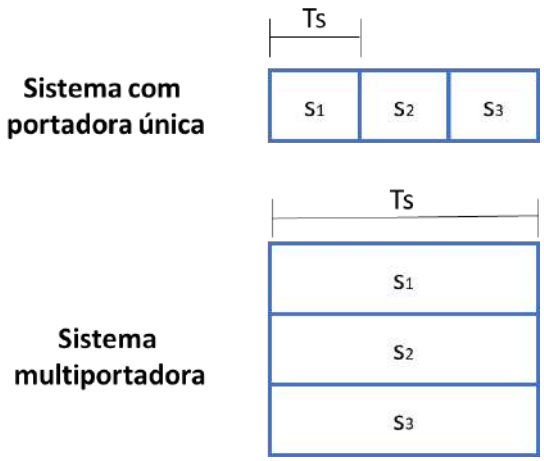

Figura 10 – Comparação de transmissão de dados em sistemas com portadora única e múltiplas portadoras.

1.1.11 Técnica de Transmissão OFDM

O OFDM é uma técnica introduzida por Chang em 1966 [12], que é utilizada em sistemas multiportadora para a transmissão paralela de símbolos com modulações digitais usando multiplexação por divisão de frequência. O OFDM estabelece que as portadoras utilizadas pelos canais sejam ortogonais, isto é, não existe sobreposição de frequência entre elas [4].

Durante o desenvolvimento do OFDM introduzido por Chang, alguns outros autores propuseram algumas modificações como a utilização do algoritmo de FFT (*Fast-Fourier Transform*) e de Prefixo Cíclico (CP) [4].

1.1.11.1 Conceitos Fundamentais para o OFDM

Antes de definir a técnica OFDM, é necessário introduzir alguns conceitos fundamentais que serão utilizados.

Em princípio, no OFDM, utiliza-se o algoritmo da Transformada Rápida de Fourier (FFT). Esse algoritmo converte um sinal representado no domínio do tempo para ser representado no domínio da frequência. A matriz da FFT de ordem N_{FFT} pode ser calculada como [4]:

$$\boldsymbol{F}_{k+1,n+1} = \frac{1}{\sqrt{N_{FFT}}} \cdot e^{-j2\cdot\pi k\cdot n/N_{FFT}}, \tag{1.60}$$

onde N_{FFT} é o número de pontos utilizados para o cálculo e $n, k = 0, ..., N_{FFT} - 1$.

Além disso, é possível converter um sinal do domínio da frequência para o domínio do tempo por meio do algoritmo da Transformada Rápida de Fourier Inversa (IFFT).

Quando a FFT é definida com normalização unitária, a matriz inversa $\boldsymbol{F}^{-1}$ é igual ao transposto conjugado da matriz FFT, ou seja, $\boldsymbol{F}^{-1} = \boldsymbol{F}^{H}$ [4]. A matriz da IFFT pode ser expressa como:

$$\boldsymbol{F^{-1}}_{k+1,n+1} = \frac{1}{\sqrt{N_{FFT}}} \cdot e^{j2\cdot\pi k\cdot n/N_{FFT}}. \tag{1.61}$$

Outro fundamento importante para o OFDM é que pode-se considerar que os canais podem ser modelados por um filtro de Resposta ao Impulso Finita (FIR), ou seja, a resposta ao impulso do canal se torna nula após um tempo finito. Assim, a resposta ao impulso de um canal pode ser definida pelo vetor de N_h coeficientes, $\boldsymbol{h} = [h_0 \quad ... \quad h_{N_h-1}]^T$. Com isso, dado um vetor de entrada $\boldsymbol{x} = [x_0 \quad ... \quad x_{N_x-1}]^T$ de comprimento N_x, o vetor de saída do canal é obtido por [4]:

$$\boldsymbol{y} = \boldsymbol{h} * \boldsymbol{x}, \tag{1.62}$$

onde $*$ representa o operador de convolução linear, aplicado no domínio do tempo discreto, que é dado por [4]:

$$y_n = \sum_{k=0}^{N_h+N_x-1} x_k \cdot h_{n-k}. \tag{1.63}$$

1.1.11.2 OFDM em Canais Ideais

A técnica OFDM, inicialmente, sintetiza eficientemente as subportadoras usando o IFFT no processo de modulação e FFT no processo de demodulação. Assim, ignorando a presença de ISI, um símbolo OFDM $\boldsymbol{s}$ pode ser obtido no domínio de tempo discreto por [4]:

$$\boldsymbol{s} = \boldsymbol{F}^{H}\boldsymbol{d}, \tag{1.64}$$

onde $\boldsymbol{d} = [d_0 \quad ... \quad d_{N_d-1}]$ é um vetor com N_d símbolos de modulações digitais no domínio da frequência. Em outras palavras, cada símbolo d_k é modulado por uma senóide complexa de frequência k/N_d e a soma ao longo do tempo n é representada na forma [4]:

$$s_n = \frac{1}{\sqrt{N_d}} \sum_{k=0}^{N_d-1} d_k e^{j2\cdot\pi k\cdot n/N_d}. \tag{1.65}$$

Na recepção, ao receber um sinal r, o processo de demodulação OFDM pode ser feito por [4]:

$$\hat{\boldsymbol{d}} = \boldsymbol{F}\boldsymbol{r}. \tag{1.66}$$

No processo de demodulação, primeiramente, converte-se o componente do sinal na frequência m/N_d. Para isso, é preciso multiplicar o sinal recebido r_n por $e^{-j2\cdot\pi\cdot m\cdot n/N_d}$. Desta forma, é necessário utilizar um filtro que realize essa multiplicação. Perceba que cada símbolo é modulado pela portadora multiplicada por um pulso retangular discretizado. Para isso, utiliza-se um filtro compatível com o pulso retangular, o qual é representado

pela soma de N_d amostras consecutivas [4], isto é,

$$\hat{d}_m = \frac{1}{\sqrt{N_d}} \sum_{n=0}^{N_d-1} r_n e^{-j2\cdot\pi m\cdot n/N_d}, \tag{1.67}$$

onde $\hat{\boldsymbol{d}}$ são os símbolos digitais demodulados que são iguais aos que foram transmitidos.

1.1.11.3 OFDM em Canais Dispersivos

Diferentemente dos canais ideais, os canais dispersivos provocam ISI no sistema de comunicação digital. Desta forma, a adição de um prefixo cíclico a técnica do OFDM é utilizada para compensar eficientemente essas distorções [4].

Primeiramente, para representar a ISI presente nos sinais digitais, utiliza-se em vez da convolução linear apresentada na equação (1.63), a convolução circular obtida por [4]:

$$r_n = \sum_{k=0}^{N_s-1} s_k \cdot h_{n-k \ mod \ N_s}, \tag{1.68}$$

onde *mod* é o operador de resto de divisão, N_s é o tamanho do vetor do sinal $\boldsymbol{s}$ e $\boldsymbol{r}$ é a convolução circular do sinal $\boldsymbol{s}$ com os coeficientes do canal $\boldsymbol{h}$. Essa operação de convolução circular também pode ser representada matricialmente por [4]:

$$\boldsymbol{r} = \boldsymbol{H_c s} = \begin{bmatrix} h_0 & 0 & \cdots & h_{N_h-1} & \cdots & h_1 \\ h_1 & h_0 & & & \ddots & \vdots \\ \vdots & & \ddots & & & h_{N_h-1} \\ h_{N_h-1} & & & h_0 & & 0 \\ 0 & \ddots & & \vdots & \ddots & \vdots \\ \vdots & & h_{N_h-1} & h_{N_h-2} & \cdots & h_0 \end{bmatrix} \begin{bmatrix} s_0 \\ s_1 \\ s_2 \\ \vdots \\ s_{N_s-1} \end{bmatrix}, \tag{1.69}$$

onde $\boldsymbol{H_c}$ é uma matriz de convolução circular $N_s \times N_s$.

A matriz de convolução $\boldsymbol{H_c}$ pode ser decomposta utilizando duas outras matrizes, ou seja,

$$\boldsymbol{H_c} = h_0 \boldsymbol{I_{N_s}} + h_1 \boldsymbol{Q} + h_2 \boldsymbol{Q}^2 + ... + h_{N_h-1} \boldsymbol{Q}^{L-1}, \tag{1.70}$$

onde I_N é uma matriz identidade de dimensão N x N e a matriz

$$\boldsymbol{Q} = \begin{bmatrix} 0 & 0 & 0 & \cdots & 1 \\ 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ \vdots & & \ddots & & \vdots \\ 0 & 0 & \cdots & 1 & 0 \end{bmatrix} \tag{1.71}$$

representa uma matriz de atraso unitário [4].

Dessa forma, pode-se representar o processo de demodulação por:

$$\hat{\boldsymbol{d}} = \boldsymbol{F}\boldsymbol{r} = \boldsymbol{F}\boldsymbol{H}_c\boldsymbol{F}^H\boldsymbol{r}, \tag{1.72}$$

onde $\boldsymbol{\Lambda} = \boldsymbol{F}\boldsymbol{H}_c\boldsymbol{F}^H$ é uma matriz diagonal com os valores correspondentes a DFT de $\boldsymbol{h}$ em sua diagonal. Logo,

$$\boldsymbol{\Lambda} = h_0\boldsymbol{I}_{\boldsymbol{N_s}} + h_1\boldsymbol{F}\boldsymbol{Q}\boldsymbol{F}^H + h_2\boldsymbol{F}\boldsymbol{Q}^2\boldsymbol{F}^H + ... + h_{N_h-1}\boldsymbol{F}\boldsymbol{Q}^{N_h-1}\boldsymbol{F}^H. \tag{1.73}$$

Portanto,

$$\boldsymbol{D} = \boldsymbol{F}\boldsymbol{Q}\boldsymbol{F}^H = diag\left\{\left[1 \; e^{-j2\pi/N_s} \; \cdots \; e^{-j2\pi k/N_s} \; \cdots \; e^{-j2\pi(N_s-1)/N_s}\right]\right\}, \tag{1.74}$$

onde $diag\{.\}$ representa os elementos de uma matriz diagonal. Assim, a matriz Λ pode simplesmente ser escrita como [4]

$$\Lambda_{k+1,k+1} = \sum_{n=0}^{N_s-1} h_n\boldsymbol{D}^n = \sum_{n=0}^{N_s-1} h_n e^{-j2\pi kn/N_s}, \tag{1.75}$$

onde $k = 0, ..., N_s - 1$ e as demais posições da matriz é igual a zero.

Por fim, a equação (1.72) com o uso de convolução circular pode ser escrita como [4]

$$\hat{\boldsymbol{d}} = \Lambda\boldsymbol{r} = \boldsymbol{h} \circ \boldsymbol{r} = \left[h_0 r_0 \; \cdots \; h_{N_s-1} r_{N_s-1}\right]^T, \tag{1.76}$$

onde $\circ$ representa a operação de produto de matriz Hadamard.

O CP é uma técnica utilizada para transformar a convolução linear realizada em um canal em uma convolução circular [4]. O CP consiste em durante a transmissão de símbolos OFDM estabelecer um intervalo de guarda entre os símbolos para que se evite ISI entre os símbolos. Dessa forma, nesse intervalo de guarda é adicionado G elementos que são iguais aos G últimos símbolos do bloco do OFDM. O tamanho desse intervalo deve ser maior que a dispersão de tempo do canal e não ser muito alto para não afetar fortemente o desempenho do sistema [13].

Na Figura 11 é exibido um diagrama de blocos contendo as etapas de um sistema que aplica a técnica OFDM em canais dispersivos. Note que se o canal for ideal, as etapas de adição e remoção do CP são removidas.

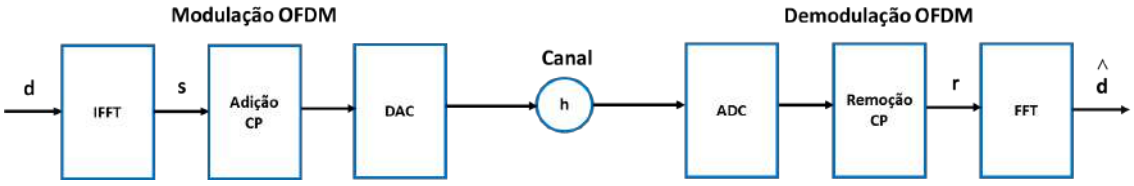

Figura 11 – Diagrama de blocos de um sistema OFDM.

1.1.11.4 Equalização em Sistemas OFDM

Um sinal transmitido, ao passar por um canal, terá suas componentes de amplitude e fase modificadas devido aos efeitos impostos pelo canal. Logo, para que se recupere ou se aproxime do sinal original transmitido, é necessário utilizar um processo de equalização [14].

A equalização *Zero Forcing* (ZF) consiste, no domínio da frequência, em dividir o sinal obtido pela resposta à frequência do canal. Portanto, o sinal pode ser equalizado da seguinte forma:

$$\boldsymbol{x} = \frac{\boldsymbol{y}}{\boldsymbol{h}}, \tag{1.77}$$

onde $\boldsymbol{x}$ é um vetor com os sinais transmitidos no domínio da frequência, y é um vetor com os sinais recebidos no domínio da frequência e h é a resposta a frequência do canal [14].

Considere um sistema OFDM como o da Figura 11. Além disso, considere a adição de ruído AWGN no canal com média zero e variância σ_v^2. Desta forma, o sinal recebido é

$$\hat{\boldsymbol{d}} = \boldsymbol{F}\boldsymbol{r} = \boldsymbol{\Lambda}\boldsymbol{d} + \boldsymbol{v} = [h_0 d_0 \ ... \ h_{N_d-1} d_{N_d-1}]^T + [v_0 \ ... \ v_{N_d-1}]^T , \tag{1.78}$$

onde $\boldsymbol{v}$ é um vetor com o ruído AWGN.

Uma outra forma de realizar a equalização do sinal é através da técnica *one-tap* [4]:

$$\check{\boldsymbol{d}} = \boldsymbol{w} \circ \hat{\boldsymbol{d}} = [w_0 h_0 d_0 \ ... \ w_{N_d-1} h_{N_d-1} d_{N_d-1}] + [w_0 v_0 \ ... \ w_{N_d-1} v_{N_d-1}] , \tag{1.79}$$

onde o vetor $\check{\boldsymbol{d}}$ contém os valores estimados para d e os coeficientes de equalização são obtidos por

$$w_k = \frac{1}{h_k}. \tag{1.80}$$

Além disso, o desempenho do sistema OFDM pode ser avaliado com o uso da *SNR* para cada subportadora de um sinal de um sistema OFDM é dada por [4]:

$$SNR_{OFDM\,k} = |h_k|^2 \frac{\sigma_{d_k}^2}{\sigma_v^2}, \tag{1.81}$$

onde $\sigma_{d_k}^2$ representa a variância dos símbolos transmitidos na subportadora k, σ_v^2 é a variância do ruído aditivo, e h_k é a resposta do canal nessa subportadora.

1.1.12 Rádio Definido por *Software*

Com o constante avanço de tecnologias de comunicação e processamento digital de sinais, sistemas modernos de comunicação se adaptaram para entregar plataformas flexíveis implementadas através de software a fim de acompanharem o ritmo desses novos avanços. Esses sistemas são conhecidos como Rádio Definido por *Software* (SDR), que consistem em sistemas de transmissão e recepção de sinais de radiofrequência com algumas funções como modulação, demodulação, codificação e decodificação implementadas por meio do software [15].

A figura 12 mostra um diagrama de blocos dos componentes principais que estão presentes em um SDR.

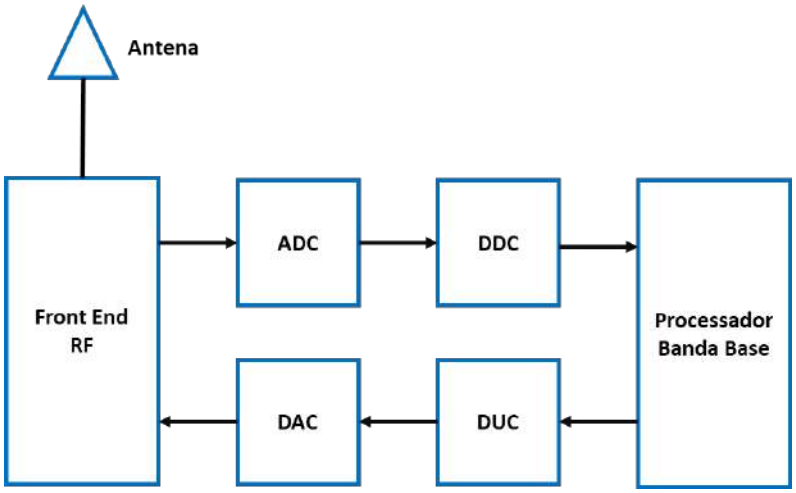

Figura 12 – Diagrama de blocos de um SDR

O *Front End RF* é o componente responsável pela transmissão e recepção dos sinais de radiofrequência. Ele também controla o ganho, filtra ruídos e converte o sinal recebido para uma frequência intermediária, que facilita o processamento pelo conversor analógico-digital (ADC). A conversão para frequência intermediária simplifica operações como amplificação, filtragem e amostragem. Os conversores ADC e DAC realizam a transformação entre os domínios analógico e digital nessa frequência[16].

Os conversores ADC e DAC são utilizados para realizar a conversão do sinal analógico para o digital de frequência intermediária e de digital para analógico de frequência intermediária [16].

O módulo *Digital Down-Conversion* (DDC) é responsável por efetuar uma reamostragem do sinal digital de frequência intermediária para um sinal digital de banda base. Já o módulo *Digital Up-Conversion* (DUC) realiza o processo inverso, transformando o sinal banda base apto a ser utilizado no conversor DAC [16].

Por último, no SDR existe um componente designado a realizar o processamento do sinal banda base. Esse componente realiza operações como equalização, codificação/decodificação e modulação/demodulação de sinais banda base. Esse componente é importante para que se possibilite utilizar o processamento digital de sinais de banda base durante uma transmissão digital [16].

1.2 Aprendizado de Máquina

O aprendizado de máquina é um campo da inteligência artificial voltado ao desenvolvimento de métodos computacionais que extraem padrões e conhecimento a partir de dados. Esses métodos podem ser aplicados a diversas tarefas, como previsão, classificação,

agrupamento e detecção de anomalias. O principal objetivo é permitir que os sistemas melhorem seu desempenho com base na experiência, sem necessidade de reprogramação de seus códigos [17].

Normalmente, os algoritmos de aprendizado de máquina utilizam de um banco de dados que é dividido entre dados de treinamento, validação e teste. Os dados de treinamentos são utilizados durante o treinamento do modelo computacional que irá realizar a classificação ou predição. Os dados de validação são utilizados durante o treinamento para verificar o desempenho do modelo. Por fim, os dados de teste são utilizados após o treinamento do modelo para verificar o desempenho final do modelo [17].

Na Figura 13 é exibido o esquema de funcionamento dos algoritmos de aprendizado de máquina utilizados para realizar classificação ou previsão de dados. Note que durante o treinamento do modelo são utilizados dados para treinamentos que normalmente são a base de dados de treinamento e a de validação. Em seguida, com o modelo já treinado, pode-se utilizar dados para realizar a classificação como o caso da base de dados de testes.

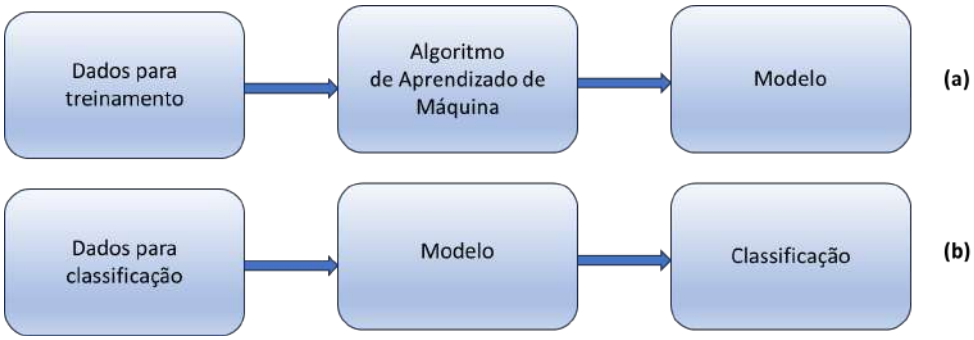

Figura 13 – Esquema de funcionamento de algoritmos de aprendizado de máquina; (a) Treinamento do modelo; (b) Classificação de dados utilizando o modelo treinado.

Os algoritmos de aprendizado de máquina podem ser classificados em três tipos: aprendizado supervisionado, aprendizado não supervisionado e aprendizado por reforço [18].

Os algoritmos de aprendizado supervisionado utilizam de um conjunto de dados de treinamentos que possuem uma rotulação da classe que o algoritmo deve classificar com os dados que são apresentados em sua entrada [18].

No aprendizado não supervisionado, os algoritmos não precisam da classificação dos dados. Os modelos desses algoritmos se ajustam de acordo com os dados de entrada de treinamento [18].

No aprendizado por reforço, um agente interage com um ambiente. O agente possui um objetivo a ser alcançado e pode desempenhar ações. Esse agente recebe recompensas de acordo com suas ações. Assim, se a ação desempenhada for boa para alcançar o objetivo, essa recompensa é positiva. Caso contrário, a recompensa é negativa. Portanto, nesse aprendizado o agente aprende de acordo com as recompensas que ganha [18].

Além dessa classificação, os algoritmos também podem ser divididos em outras categorias como regressão, classificação, aprendizado em conjunto, explicativos, agrupamento e redução de dimensionalidade [19].Cada uma das categorias possui vantagens em serem utilizadas para diferentes ocasiões. Para esse projeto, o algoritmo escolhido foi a Rede Neural Convolucional (CNN) que pertence ao conjunto de algoritmos de classificação. Essa escolha se deve ao bom desempenho desse algoritmo em entradas cuja ordem dos dados é importante, como no caso de imagens e sinais de comunicação.

1.2.1 Classificação de Algoritmos de Aprendizado de Máquina

Os algoritmos de aprendizado de máquina podem ser classificados de acordo com seu objetivo ou funcionalidade. A seguir são descritos alguns dos tipos de algoritmos que podem ser encontrados e quais são suas principais vantagens de uso [19].

1.2.1.1 Algoritmos de Regressão

Os algoritmos de regressão tem como objetivo utilizar um conjunto de variáveis independentes para realizar uma previsão de uma variável de saída. Desta forma, esses algoritmos buscam encontrar uma função que relaciona as variáveis de entrada com a de saída. Alguns exemplos utilizados são a regressão linear e a regressão logística. Esses algoritmos são utilizados principalmente para encontrar funções que representam o sistema de previsão dos dados [19].

Na Figura 14 pode ser visto um resultado de previsão obtida com o uso de uma regressão linear em dados com duas variáveis de entrada.

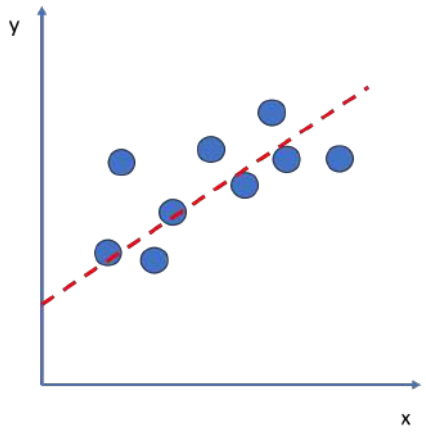

Figura 14 – Exemplo de previsão feita pelo algoritmo de regressão linear em dados com duas variáveis de entrada.

1.2.1.2 Algoritmos de Classificação

Os algoritmos de classificação possuem como objetivo classificar os dados de acordo com um conjunto de classes definidas. Esses algoritmos podem utilizar de decisões ou de

probabilidades para realizar a classificação. Alguns exemplos são as árvores de decisão, o Naive Bayes e as redes neurais artificiais. Essa categoria de algoritmos é amplamente utilizada em problemas de classificação de dados em diferentes classes predefinidas [19].

Um exemplo de classificação em duas diferentes classes e para dados com duas variáveis de entrada pode ser visualizado na Figura 15.

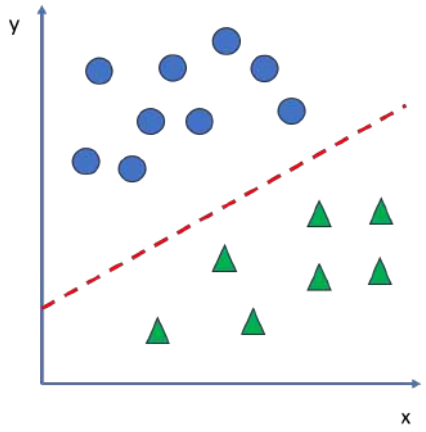

Figura 15 – Exemplo de classificação feita por um algoritmo classificador em dados com duas variáveis de entrada.

1.2.1.3 Algoritmos de Aprendizado em Conjunto

Os Algoritmos de Aprendizado em Conjunto (*Ensemble Learning*) tem como objetivo combinar diferentes previsões ou classificações para realizar uma votação de qual foi a mais frequente para ser utilizada como a previsão ou classificação final. Alguns exemplos de algoritmos de aprendizado em conjunto são florestas aleatórias, *XGBoostName*, *Light GBM* e *CatBoost*. Esses algoritmos possuem um bom desempenho em problemas de classificação e regressão, atingindo muitas vezes uma acurácia maior que com a utilização de somente um classificador ou regressor [19].

Na Figura 16 pode ser visto um exemplo de resultado de uso de um algoritmo de aprendizado em conjunto. Nesse caso foram utilizados 3 classificadores e realizada uma votação para decidir a classificação final.

1.2.1.4 Algoritmos Explicativos

Os algoritmos explicativos possuem como objetivo entender as características das variáveis envolvidas com o problema em vez de realizar uma classificação ou previsão. Alguns algoritmos explicativos são SHAP (*SHapley Additive exPlanations*) e LIME (*Local Interpretable Model-Agnostic Explanations*) [20].

O SHAP e o LIME são ferramentas que auxiliam modelos preditivos a encontrarem a importância de cada variável de entrada na predição final. Desta forma, é possível

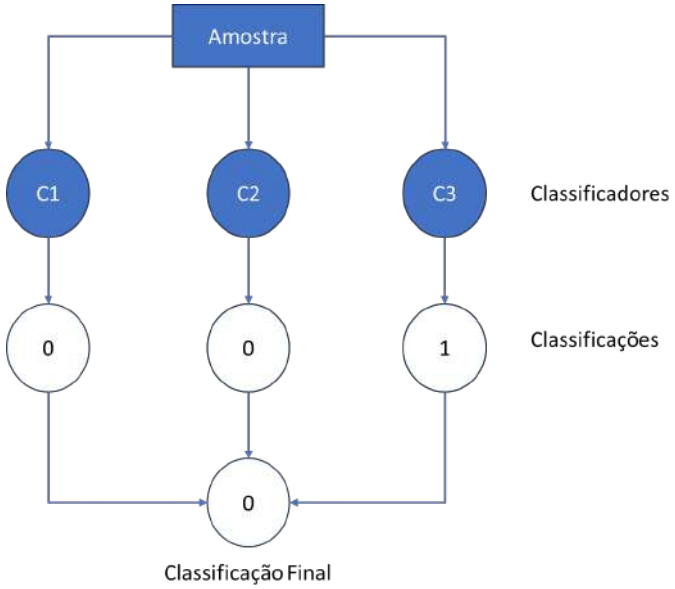

Figura 16 – Esquema de classificação feita por um algoritmo de aprendizado em conjunto.

encontrar faixas de valores que estão mais presentes em cada um dos tipos de classes [20].

Tais algoritmos são utilizados quando principalmente deseja-se entender o motivo que uma determinada classificação foi feita de uma maneira. Logo, esses algoritmos podem ser utilizados em conjuntos com outros tipos de algoritmos para conseguir realizar uma boa análise dos dados [20].

Uma funcionalidade do algoritmo explicativo SHAP é exibida na Figura 17. Note que com o uso desse algoritmo é possível visualizar a contribuição de cada variável de entrada no resultado da saída.

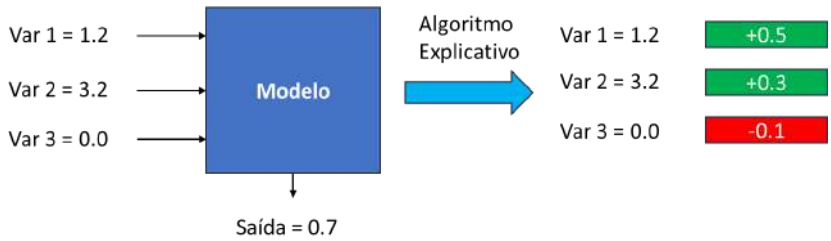

Figura 17 – Exemplo de resultado obtido com o uso do algoritmo explicativo SHAP em dados com 3 variáveis de entrada.

1.2.1.5 Algoritmos de Agrupamento

Os algoritmos de agrupamento tem como objetivo agrupar o conjunto de entradas em grupos (*clusters*). Esses agrupamentos são feitos de forma que a similaridade das características entre os dados do mesmo grupo seja a mais alta possível e entre os dados de diferentes grupos sejam a mais baixa possível. Alguns exemplos de algoritmos de agrupamento são *K-Means* e o agrupamento hierárquico. Esses algoritmos são utilizados

em casos que se deseja encontrar diferentes grupos em um conjunto de dados e não se tem a rotulação dos grupos já previamente definidos [19].

Na Figura 18 é exibido um exemplo de uso de um algoritmo de agrupamento em dados com 2 variáveis de entrada.

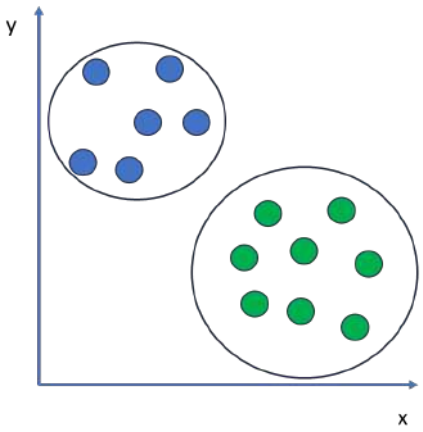

Figura 18 – Exemplo de agrupamento obtido com o uso de um algoritmo de agrupamento para dados de duas entradas.

1.2.1.6 Algoritmos de Redução de Dimensionalidade

Os algoritmos de redução de dimensionalidade possuem como objetivo reduzir a quantidade de variáveis presentes nos dados de entradas. Alguns exemplos desses algoritmos são o PCA (*Principal Component Analysis*) e o LDA (*Linear Discriminant Analysis*) [19].

A maior vantagem do uso desses algoritmos é selecionar as melhores variáveis presentes na entrada que definem os dados e dessa forma economizar recursos computacionais.

Na Figura 19 é exibido um exemplo de uso de um algoritmo de redução de dimensionalidade. Nesse caso a variável z foi descartada, permanecendo assim somente duas variáveis.

1.2.2 Redes Neurais Artificiais

As redes neurais artificiais são modelos do aprendizado supervisionado que possuem uma estrutura inspirada no sistema neural humano. O sistema neural humano é composto por neurônios que se comunicam entre si através de sinapses. Assim, uma informação recebida por um neurônio é processada por ele e transmitida para os outros neurônios através da sinapse [21].

As redes neurais artificiais possuem uma estrutura semelhante ao funcionamento do sistema neural humano. Nas redes neurais existem unidades de processamento que também são chamadas de neurônios. Cada neurônio recebe conexões de dados da entrada do modelo ou da saída de outros neurônios. Em seguida, esse neurônio irá realizar o seu

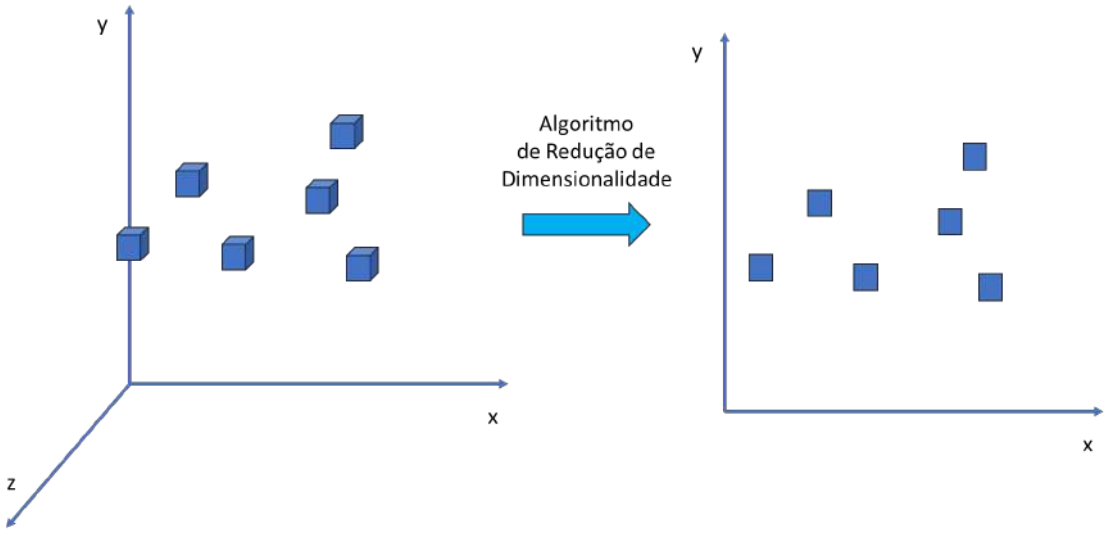

Figura 19 – Exemplo de algoritmo de redução de dimensionalidade, na qual a variável z foi descartada.

processamento que consiste em realizar um somatório ponderado por pesos e a saída é calculada por uma função de ativação [21].

O modelo de rede neural mais elementar é o perceptron criado por Frank Rosenblatt [22], [23]. Esse modelo pode ser visualizado na Figura 20. O perceptron consiste em somente uma unidade de processamento que recebe um vetor de entradas $\boldsymbol{a}$ de tamanho U. Cada uma das conexões da entrada com a unidade de processamento recebe um vetor de pesos $\boldsymbol{p}$ que é utilizado na unidade de processamento para realizar uma soma ponderada das entradas. Por fim, a saída b é calculada utilizando uma função de ativação g que utiliza como entrada a soma ponderada feita anteriormente [22]. Portanto, a saída de um perceptron é dada por:

$$b = \sum_i g(p_i \cdot a_i). \tag{1.82}$$

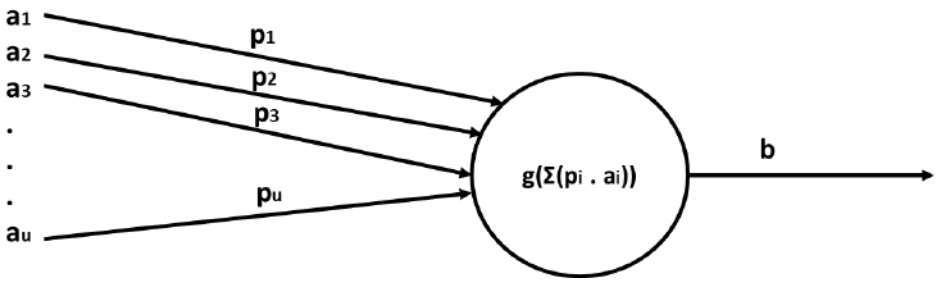

Figura 20 – Exemplo de estrutura de um perceptron.

Normalmente, as redes neurais utilizam mais de um neurônio e esses neurônios são organizados em camadas [23], como pode ser visto na Figura 21. Nessa arquitetura de rede neural tem-se uma camada de entrada, uma ou várias camadas ocultas e uma camada de

saída. A camada de entrada é responsável por receber os dados de entrada. As camadas ocultas são responsáveis por realizar os processamentos dos dados recebidos na entrada ou da saída de camadas anteriores. Finalmente, a camada de saída é responsável por realizar o processamento final e apresentar o resultado final [21].

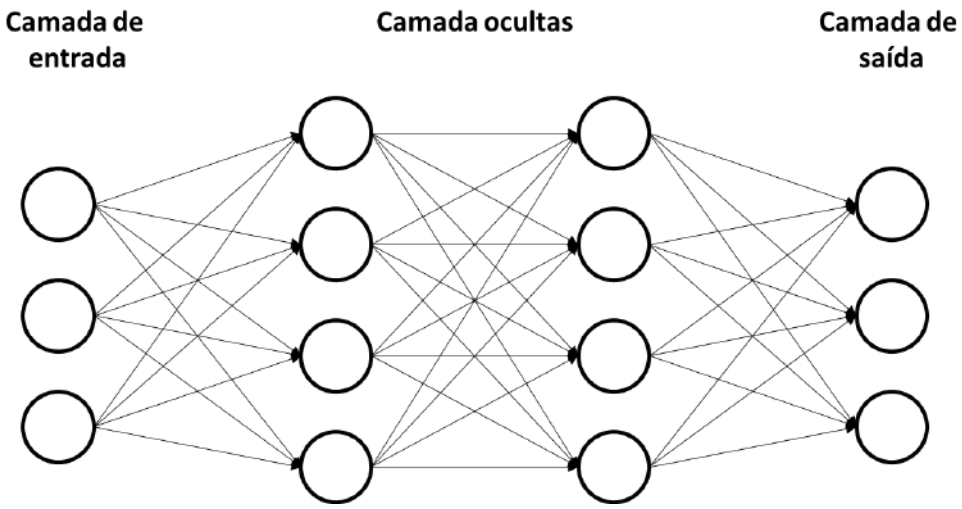

Figura 21 – Exemplo de arquitetura de uma rede neural artificial.

A rede neural exibida na Figura 21, é denominada de rede neural totalmente conectada. Isso ocorre por causa que entre as camadas existem todas as conexões entre cada par de neurônios. Porém, esse fato não ocorre em todas as redes neurais, onde algumas possuem só algumas dessas conexões entre as camadas [21].

1.2.2.1 Treinamento de Redes Neurais Artificiais

Como visto na equação (1.82), os pesos das conexões entre os neurônios contribuem para o resultados das saídas dos mesmos. Logo, para uma rede neural conseguir um bom desempenho, é necessário realizar um processo de treinamento dos valores dos pesos das conexões. Existem diferentes métodos para realizar o treinamento de redes neurais. O método mais utilizado é o de retropropagação do erro (*backpropagation*) [23], [24].

No *backpropagation*, o resultado obtido com os pesos iniciais escolhidos nas redes, é comparado com o resultado desejado. Nessa comparação é calculado um erro que é propagado desde a camada de saída até a camada de entrada, para que os pesos das conexões entre as camadas sejam atualizados de acordo com o erro. Desta forma, é calculado o gradiente da função de erro para encontrar a direção que o valor do peso deve ser modificado para que o erro seja minimizado. Assim, o peso é atualizado somando o valor atual do peso mais uma fração do gradiente calculado. Essa fração é definida por um valor relativamente baixo que é denominado taxa de aprendizado [23], [24].

1.2.2.1.1 ADAM

A taxa de aprendizagem é considerada um dos parâmetros mais difíceis de se definir em um treinamento de rede neural e pode afetar significativamente o desempenho do modelo. Assim, a criação dos algoritmos adaptativos ajudou a contornar esse problema. Tais algoritmos definem uma taxa fixa de aprendizagem ou uma política de ajuste da taxa [24].

O algoritmo ADAM estabelece uma taxa de aprendizagem para cada parâmetro [24]. Logo, os parâmetros podem receber uma taxa de aprendizagem adequada para seu aprendizado sem que isso afete os demais parâmetros. Para isso, é feito o cálculo da estimativa do primeiro momento estatístico (média) e do segundo momento estatístico (variância) dos gradientes consecutivos. Esses cálculos são denotados, respectivamente, por M_i e V_i, como abaixo:

$$M_{i+1} = \gamma_1 M_i + (1 - \gamma_1)\nabla L(\mathbf{p}_i), \tag{1.83}$$

$$V_{i+1} = \gamma_2 V_i + (1 - \gamma_2)\nabla L(\mathbf{p}_i). \tag{1.84}$$

Os cálculos de momentos estatísticos utilizam uma parcela do cálculo da iteração anterior, sendo γ_1 e γ_2 as variáveis que controlam essa relação de aproveitamento da informação anterior e a do novo valor corrente. Com isso, iterações repetidas que tendem a uma mesma direção produzem um impulso para essa direção. Portanto, neste caso, esse método se torna mais rápido que a descida do gradiente convencional para convergir para o ponto mínimo.

Veja que os hiperparâmetros γ_1 e γ_2 são, respectivamente, os pesos estabelecidos para a média e a variância dos gradientes. Porém, esses cálculos de M_i e V_i são enviesados e tendem a mover a solução em direção ao zero. Para evitar isso, é feita a seguinte correção:

$$\hat{M}_i = \frac{M_i}{1 - \gamma_1} \tag{1.85}$$

$$\hat{V}_i = \frac{V_i}{1 - \gamma_2} \tag{1.86}$$

Por fim, o algoritmo realiza o aprendizado interativo dos parâmetros através da equação

$$\mathbf{p}_{i+1} = \mathbf{p}_i - \frac{\eta}{\sqrt{\hat{V}_i} + \varepsilon}\hat{M}_i \tag{1.87}$$

onde η é uma taxa de aprendizado fixa e ε é um hiperparâmetro que evita divisão por zero.

Desta forma, o algoritmo ADAM é uma boa alternativa para ser utilizada no treinamento de redes neurais. Esse algoritmo será utilizado nesse projeto no treinamento das redes.

1.2.2.2 Funções de Ativação

As funções de ativação são transformações não lineares que representam a saída de cada neurônio. Essas funções recebem as saídas de neurônios uma camada anterior e os pesos ajustados pelo treinamento e realizam alguma transformação não linear [24]. Entre as funções de ativação utilizadas temos a *ReLU* e a *Softmax*.

Em grande parte das redes neurais utiliza-se a função de ativação *ReLU* (*Rectified Linear Unit*). Essa função é representada pela função:

$$g(b) = max\{0, b\}, \tag{1.88}$$

onde b é a saída obtida em algum neurônio da rede. A aplicação dessa função de ativação produz uma transformação não linear, porém, a função ainda permanece próxima de uma transformação linear devido aos seus dois intervalos lineares. Desta forma, preserva ainda a facilidade de se otimizar modelos lineares através dos métodos de treinamento baseados no gradiente [24].

Outra função utilizada principalmente em modelos de classificação é a *softmax*. Essa função é útil para problemas de classificação pois retorna as saídas como valores entre 0 e 1 e divide pela soma das saídas. Com isso, a saída da função pode ser vista como um vetor de probabilidades entre 0 e 1. Ela pode ser definida como [24]:

$$f(b)_l = \frac{e^{b_l}}{\sum_{k=1}^{L} e^{b_k}}, \tag{1.89}$$

onde $l = 1, ..L$ e L é o tamanho do vetor de saída b de alguma camada da rede.

1.2.2.3 Redes Neurais Convolucionais

As redes neurais convolucionais são um tipo de arquitetura de redes neurais artificiais que são muito utilizadas em entradas de dados que possuem formato matricial como imagens e sinais modulados [23], [24].

Como visto anteriormente, as redes neurais totalmente conectadas tratam as entradas com posições distantes da mesma forma que entradas com posições próximas, atribuindo pesos a elas. Nas CNNs, a estrutura espacial da entrada é levada em consideração [24].

As CNNs utilizam em sua arquitetura pelo menos uma camada que realiza a operação de convolução. A operação de convolução funciona como um filtro que analisa pequenos blocos de posições adjacentes da entrada para capturar características importantes. Através

dessa operação, pode-se encontrar características como um traço de uma imagem ou uma mudança de fase em um sinal modulado [24].

A Figura 22 mostra uma arquitetura de uma das mais famosas CNN denominada LeNet-5. Veja que nessa rede neural temos duas camadas que executam a operação de convolução entre as diferentes operações executadas nas camadas. Logo, essa arquitetura pode ser classificada como CNN.

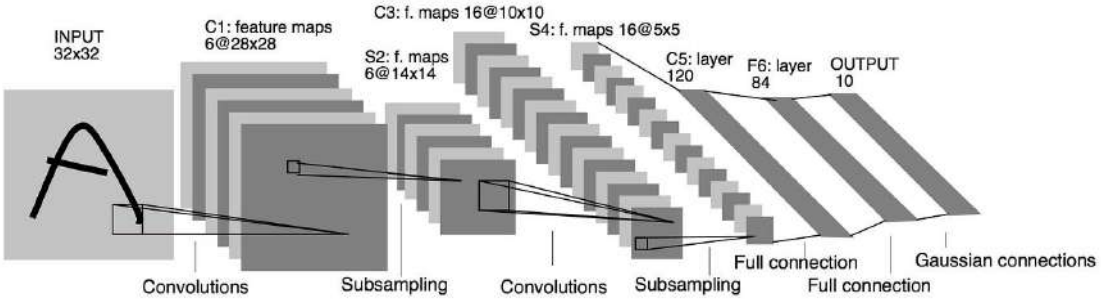

Figura 22 – Exemplo de arquitetura da CNN LeNet-5 [25].

As CNNs em comparação com as redes neurais totalmente conectadas possuem uma quantidade menor de conexões entre neurônios. Isso se deve por causa da operação de convolução que considera somente as saídas de conjunto de neurônios como pode ser visto na Figura 22 em vez de ter conexão com todos os outros neurônios da camada anterior. Essa propriedade faz com que as redes neurais possuam um menor número de pesos de conexões que devem ser treinados. Assim, evita-se conexões que não possuem grande importância para o cálculo da saída de um determinado neurônio [24]. Essa propriedade faz com que as CNNs sejam utilizadas em problemas que possuem como entrada imagens ou sinais de comunicação, pois esses dados possuem características importantes em posições que estão normalmente adjacentes.

1.2.2.3.1 A Operação de Convolução

A convolução é uma operação linear que, dada duas funções, realiza a soma do produto entre as funções ao longo da região em que elas se sobrepõem com o deslocamento entre elas. De forma geral, a convolução é representada por [21]:

$$\boldsymbol{b} = \boldsymbol{a} * \boldsymbol{c}, \tag{1.90}$$

onde $*$ é o operador de convolução, $\boldsymbol{a}$ é um vetor com os dados de entrada da camada convolucional, $\boldsymbol{c}$ é denominado *kernel*. O *kernel* funciona como um filtro de extração de características do vetor de dados. Desta forma, o cálculo da convolução para entradas discretas é [21]:

$$b_u = \sum_{k=0}^{U-1} a_k \cdot c_{u-k}, \tag{1.91}$$

onde U é o tamanho da entrada de dados.

Porém, em diversos casos, a entrada é uma matriz multidimensional de dados. No caso de entradas como sinais representados em uma matriz de duas dimensões, uma para o valor real e outra para o valor complexo do sinal, por serem bidimensionais, normalmente utiliza-se um *kernel* bidimensional e assim é necessário o uso de dois somatórios no cálculo da convolução [21]:

$$b_{v,u} = \sum_k \sum_t a_{u,t} \cdot c_{v-k,u-t}. \tag{1.92}$$

Para melhor compreensão da operação de convolução da CNN, considere a entrada $\boldsymbol{a} = \begin{bmatrix} 1 & 2 & 1 & 2 \\ 1 & 2 & 2 & 1 \end{bmatrix}$ e o *kernel* $\boldsymbol{c} = \begin{bmatrix} 2 & 1 \end{bmatrix}^T$. Ao utilizar a equação (1.91), resulta na seguinte matriz $\boldsymbol{b}$:

$$\boldsymbol{b} = \begin{bmatrix} 1 \times 2 + 1 \times 1 & 2 \times 2 + 2 \times 1 & 1 \times 2 + 2 \times 1 & 2 \times 2 + 1 \times 1 \end{bmatrix} = \begin{bmatrix} 3 & 6 & 4 & 5 \end{bmatrix}$$

e a operação é representada visualmente na Figura 23.

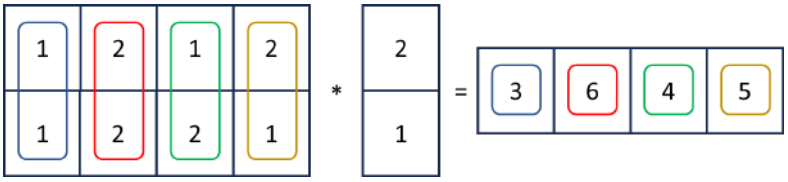

Figura 23 – Exemplo da operação de convolução.

1.2.2.3.2 *Pooling* e *Stride*

Normalmente, em CNNs, temos três estágios que são utilizados em seguida. O primeiro estágio é a utilização da operação de convolução. Em seguida, no segundo estágio, usa-se uma função de ativação. Por fim, são utilizadas as camadas com as funções de agrupamento, também denominadas como *pooling* [24]. Nessas camadas são utilizadas funções de agrupamento que recebem as saídas das unidades da camada anterior para gerar um estatística resumida. Entre as funções de agrupamento mais utilizadas tem-se o valor máximo, o valor mínimo, a média e a mediana.

Por exemplo, considerando as saídas das unidades de uma camada anterior, ao passar pela função de *pooling* de valor máximo, conhecida como *max-pooling*, a maior saída entre as unidades será devolvida pela função de *pooling*.

Para a utilização do *pooling*, também é preciso estabelecer o tamanho do *kernel* que determina a vizinhança que será a entrada para a função de agrupamento. Por exemplo, na Figura 24 foi utilizado a função de agrupamento *max-pooling* com um *kernel* de tamanho 1×2.

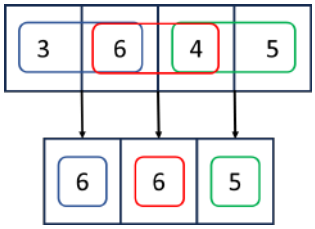

Figura 24 – Exemplo da operação de *max-pooling* com *stride* igual a 1.

O uso da função de agrupamento ajuda no custo de memória e processamento durante o treinamento das redes convolucionais, por causa da redução de dados imposta durante essa sumarização promovida pela função. Além disso, permite encontrar padrões presentes em vizinhança que resultam em uma mesma sumarização [24].

Outro fator importante nas CNNs é o *stride*. O *stride* é a quantidade de posições em que o *kernel* irá se deslocar no vetor de entrada a cada passo da convolução [24]. Note que na Figura 24 foi utilizado um *stride* igual a 1. Desta forma, se o stride for aumentado a 2, tem-se o resultado visualizado na Figura 25.

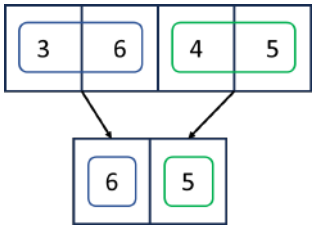

Figura 25 – Exemplo da operação de *max-pooling* com *stride* igual a 2.

Portanto, o *stride* define o nível de sumarização, que conforme maior o seu valor, mais reduzido será a quantidade de informações obtidas na saída da função de agrupamento e melhor será o desempenho computacional.

1.2.3 Métricas de Avaliação

Durante o treinamento do modelo de classificação ou predição, é importante realizar a avaliação do desempenho desse modelo em um conjunto de validação. Para isso, são aplicadas métricas de avaliação. Dentre essas métricas, pode-se destacar a acurácia, a precisão, a revocação e o *F1-score* [26].

Para a definição das métricas, é preciso ressaltar que durante a classificação de um dado pertencente a um classe X pelo modelo, pode-se ocorrer 4 casos possíveis:

1. **Verdadeiro Positivo (VP)**: O modelo classificou o dado como da classe X, e efetivamente o dado é da classe X.
2. **Verdadeiro Negativo (VN)**: O modelo classificou o dado como da classe $Y \neq X$, e o dado é da classe Y.

3. **Falso Positivo (FP)**: O modelo classificou um dado como da classe X, e o dado é da classe $Y \neq X$.

4. **Falso Negativo (FN)**: O modelo classificou dado como da classe $Y \neq X$, e o dado é da classe $Z \neq X$, onde $Z \neq Y$.

Desta forma, somando todas as ocorrências de cada um dos casos acima, em um conjunto de validação, pode-se utilizar esse número para calcular o valores das métricas de avaliação.

A acurácia apresenta a porcentagem de dados classificados corretamente sobre o total de amostras de dados. A acurácia pode ser obtida por [26]:

$$Acurácia = \frac{VP + VN}{VP + VN + FP + FN}, \tag{1.93}$$

onde VP, VN, FP e FN são respectivamente o número total de ocorrências de Verdadeiro Positivo, Verdadeiro Negativo, Falso Positivo e Falso Negativo.

A precisão apresenta a porcentagem de dados classificados corretamente como positivos sobre o total de dados classificados como positivos. Logo, a precisão pode ser obtida por [26]:

$$Precisão = \frac{VP}{VP + FP}. \tag{1.94}$$

A revocação apresenta a porcentagem de dados classificados corretamente como positivo sobre o total de dados que realmente são positivos. Logo, a revocação pode ser obtida por [26]:

$$Revocação = \frac{VP}{VP + FN}. \tag{1.95}$$

Por fim, o *F1-score* é uma métrica de avaliação que consiste em uma média harmônica das métricas de precisão e revocação. Essa métrica é obtida por [26]:

$$F1\text{-}score = 2 \cdot \frac{Precisão \cdot Revocação}{Precisão + Revocação}. \tag{1.96}$$

Além das métricas de avaliação, uma forma de visualizar o desempenho do sistema é através do uso da matriz de confusão. A matriz de confusão permite visualizar a quantidade de classificações que foram preditas para cada uma das classes presentes no conjunto.

Por exemplo, dado um conjunto com três classes: X, Y e Z. Um modelo classifica um conjunto de amostras e na Figura 26 é mostrado como a matriz de confusão exibe os resultados obtidos. Note que $C(i, j)$ indica a quantidade de amostras da classe i que foram classificadas como da classe j. Assim, as quantidades de classificações realizadas corretamente são apresentadas na diagonal principal da matriz, indicada com a coloração azul.

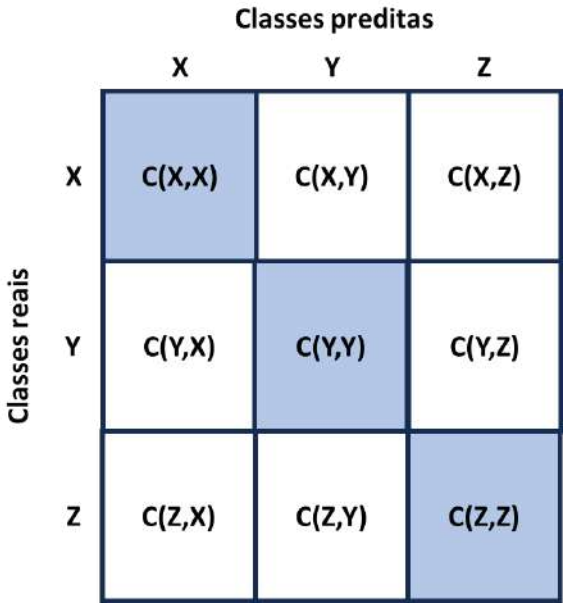

Figura 26 – Exemplo de matriz de confusão para dados que podem ser classificados nas classes X, Y e Z, onde $C(i, j)$ indica a quantidade de amostras da classe i que foram classificadas como da classe j.

2 Metodologia

Este trabalho propõe a criação de um modelo computacional utilizando algoritmos de aprendizado de máquina para a detecção da modulação empregada em sinais digitais. Para o desenvolvimento desse modelo, foi utilizado o *software* MATLAB na geração dos sinais, na simulação do sistema de comunicação e na construção do modelo de aprendizado de máquina.

A metodologia foi dividida em cinco etapas. A primeira responsável pela criação de bases de dados de sinais modulados. Na segunda etapa, foram criadas CNNs para resolverem, em partes, o problema de classificação. Na terceira etapa, as CNNs foram combinadas para a construção de um classificador final. Na quarta etapa, as CNNs do classificador foram treinadas com a base de dados de sinais modulados. Por fim, na última etapa, o desempenho dos modelos são aferidos com o uso de sinais simulados criados através do *software* e com sinais reais recebidos através do uso de um SDR.

A figura 27 exibe um fluxograma das etapas que foram executadas na realização do projeto. A seguir serão discorridos com mais detalhes cada uma das etapas de desenvolvimento.

2.1 *Softwares* e Equipamentos

Para a elaboração do projeto foram utilizados *softwares* e equipamentos que desempenharam um papel fundamental no desenvolvimento e na validação do modelo computacional proposto. Esses recursos permitiram tanto a simulação de cenários quanto a aplicação prática em sistemas reais, melhorando a abrangência dos resultados obtidos.

A seguir, serão discutidos em detalhes os principais *softwares* e equipamentos empregados, bem como seus respectivos usos e contribuições durante as etapas do projeto. Essa análise visa evidenciar a importância dessas ferramentas no alcance dos objetivos estabelecidos e na obtenção dos resultados.

2.1.1 *Software* MATLAB

O principal *software* utilizado durante o projeto foi a versão R2023b do MATLAB. O MATLAB é um programa da empresa MathWorks de cálculo numérico que possui linguagem de programação integrada. Esse programa tem como principal diferencial a presença de pacotes com funções prontas que ajudam na confecção de trabalhos envolvendo tarefas como processamento de sinais e aprendizado de máquina. Durante a produção

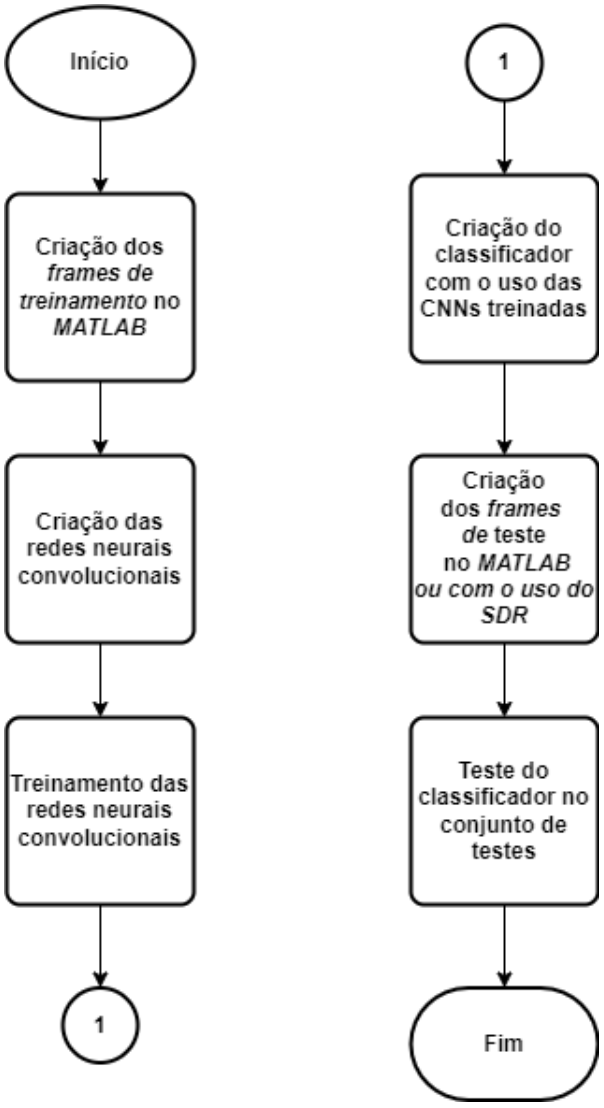

Figura 27 – Fluxograma do projeto.

do trabalho, foram utilizados os pacotes *Deep Learning* e *Communications* presentes no software.

O pacote *Deep Learning* facilita a criação e o treinamento de modelos de aprendizado de máquina. Desta forma, o pacote foi usado para criar as redes neurais convolucionais e para realizar o treinamento das mesmas.

O pacote *Communications* possui funções que facilitam a simulação e a manipulação de sinais digitais. Assim, esse pacote possibilitou a modelagem dos sinais digitais modulados e da simulação da transmissão e recepção destes sinais.

Além dos usos citados anteriormente, o MATLAB funcionou como o intermediário entre o SDR utilizado e os modelos de aprendizado de máquina. Logo, no *software* criaram-se os sinais transmitidos pelo SDR e exibidos os sinais que foram recebidos no SDR.

2.1.2 SDR bladeRF

O equipamento utilizado para teste do modelo computacional com o uso de sinais reais foi o SDR bladeRF xA9. A bladeRF xA9 é um SDR desenvolvido pela empresa Nuand para a elaboração de projetos envolvendo comunicação de radiofrequência. Além disso, a Nuand fornece fóruns, blogs, bibliotecas e tutoriais para ajudar na elaboração dos projetos [27].

A figura 28 exibe um diagrama de blocos dos componentes presentes nesse SDR. Veja que o SDR possui os componentes de acordo com os padrões discutidos anteriormente, possuindo assim um módulo para transmissão e recepção RF, DAC e PLL para conversão dos sinais e um FPGA para o processamentos dos sinais banda base [28].

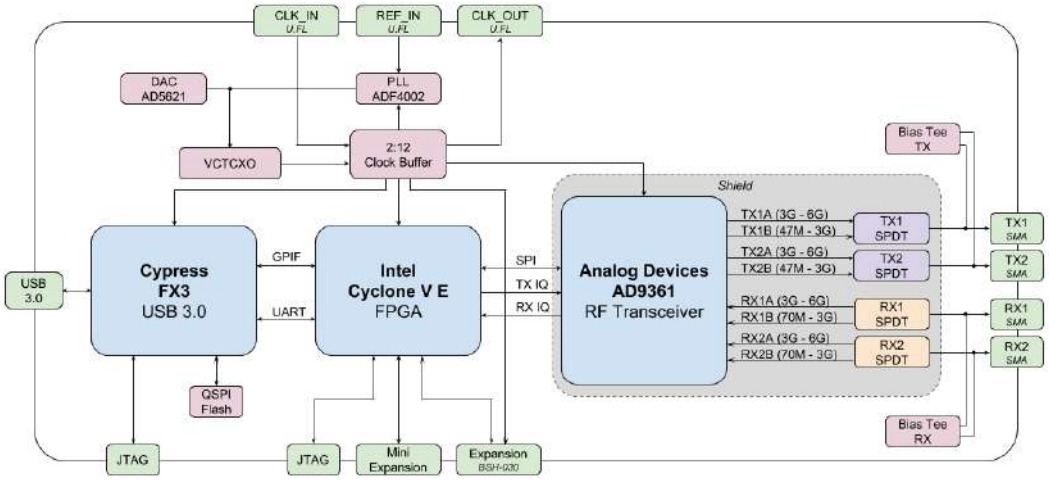

Figura 28 – Diagrama de blocos do SDR bladeRF xA9 [28]

Nesse projeto foi utilizado o pacote *Communications Toolbox Support Package for BladeRF 2.0* presente no MATLAB para realizar a conexão entre o *software* e o SDR. Com isso, foi possível a criação dos sinais digitais no MATLAB que foram enviados de uma antena de transmissão para outra antena de recepção no SDR. Por fim, os sinais captados no SDR são exibidos no MATLAB que podem ser utilizados para futuros processamentos de sinais, classificações e análises.

2.2 Geração de Sinais Modulados

A primeira etapa do trabalho tem como objetivo construir uma base de dados de sinais modulados para que possa ser usada para treinar e testar o classificador do tipo de modulação empregada. Resumidamente, esse processo pode ser visualizado através da Figura 29. O processo envolve o uso de diferentes esquemas de modulação, a transmissão, a recepção através de um canal AWGN ou Rician com ruído AWGN, a normalização do sinal para potência unitária e o armazenamento dos sinais.

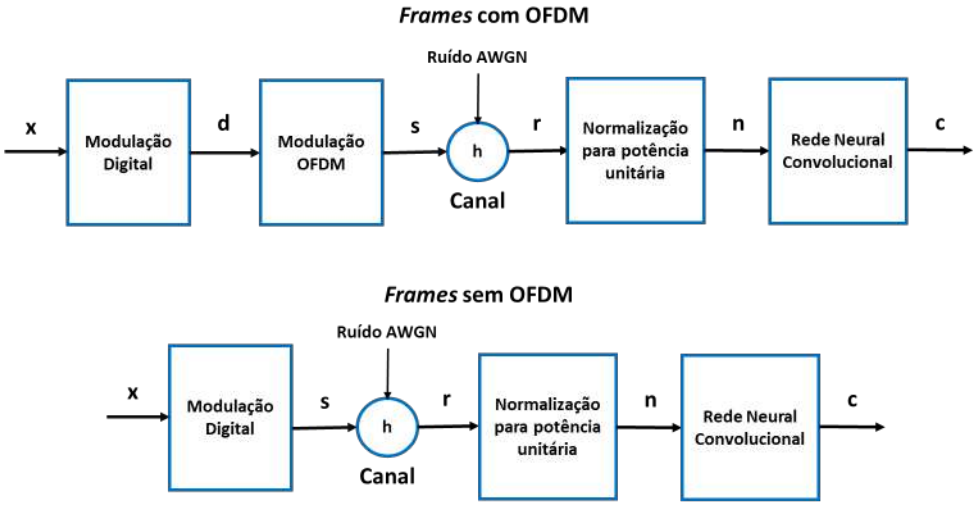

Figura 29 – Processo de criação dos sinais modulados para a base de dados utilizadas nas CNNs.

Tabela 1 – Parâmetros utilizados na criação do canal Rician.

Parâmetros	**Valor**
Taxa de amostragem	4 MHz
Atrasos dos multipercursos	[0 0,1 1,3] $\backslash 4 \cdot 10^6 s$
Ganho médio dos percursos	[0 -3 -10] dB
K-Fator	4
Máxima variação de frequência (Doppler)	2 Hz

O canal Rician utilizado neste projeto possui as características descritas na Tabela 1. Ele é composto por três percursos, com um fator K igual a 4, indicando a presença de uma componente dominante significativa entre os três percursos. Os atrasos dos multipercursos foram normalizados pela taxa de amostragem, o que significa que estão expressos em número de amostras em vez de segundos, facilitando sua aplicação em ambientes de simulação digital. Além disso, o canal apresenta uma variação máxima de frequência de 2 Hz, caracterizando-o como quase estático. Esse canal foi implementado por meio do objeto *comm.RicianChannel* do MATLAB. Durante a geração dos sinais modulados, o canal foi recriado a cada 400 amostras, com uma nova semente aleatória, de forma a evitar o sobreajuste do modelo aos dados.

O sistema de comunicação desenvolvido neste trabalho foi projetado para operar com *frames* compostos por 80 símbolos digitais modulados. Essa escolha se baseia no tamanho dos frames adotados pelo padrão IEEE 802.11g, um dos padrões de redes sem fio que utilizam OFDM. A Tabela 2 apresenta os principais parâmetros do símbolo OFDM conforme especificado no padrão IEEE 802.11g.

Os *frames* gerados neste projeto utilizam quatro esquemas de modulação distintos: BPSK, QPSK, 16QAM e 64QAM. Para cada um desses tipos de modulação, foram criados conjuntos de *frames* tanto com a aplicação da técnica OFDM quanto sem ela. No caso

Tabela 2 – Alguns parâmetros do padrão IEEE 802.11g para os símbolos OFDM [29].

Parâmetros	**Valor**
Pontos da FFT	64
Tamanho do prefixo cíclico	$16 \cdot T_s$
Modulações	BPSK, QPSK, 16-QAM e 64-QAM

dos *frames* que empregam OFDM, os parâmetros adotados seguem as especificações do padrão IEEE 802.11g, conforme descrito na Tabela 2.

Os sinais digitais foram representados no MATLAB através do uso de vetores com os valores dos símbolos na forma LPE. A Figura 30 exibe as constelações das 4 diferentes modulações presentes no padrão IEEE 802.11g com seus respectivos valores na forma LPE em que os símbolos podem assumir.

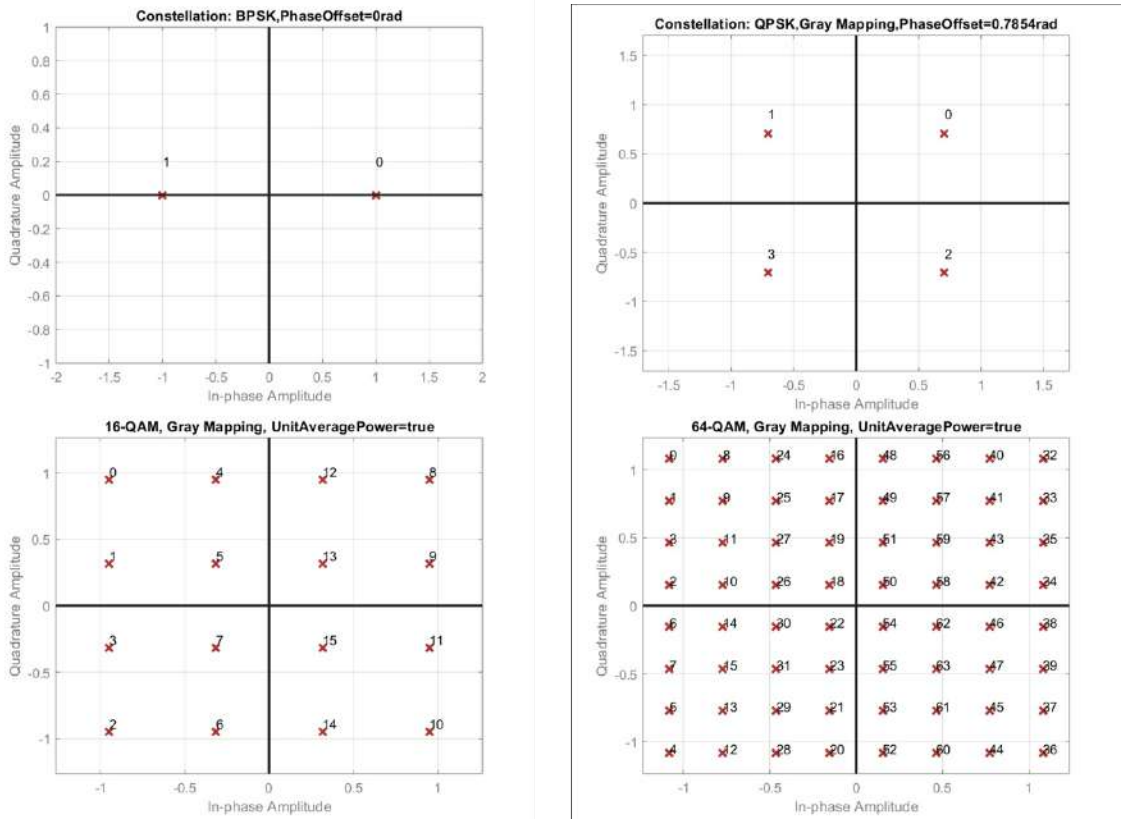

Figura 30 – Constelações das modulações presentes no padrão IEEE 802.11g.

Primeiramente, nos *frames* que utilizam OFDM, os símbolos gerados terão 64 símbolos digitais que foram modulados com uma das quatro possíveis modulações, que com a aplicação da IFFT de 64 pontos e com o acréscimo do CP, resultam em *frames* com 80 sinais modulados.

Em seguida, os símbolos modulados gerados foram representados por vetores com 64 posições. Esse vetor foi utilizado para gerar o símbolo OFDM executando a aplicação da IFFT com 64 pontos e a adição do prefixo cíclico de tamanho 16. Desta forma, repetiu-se

esse processo para se gerar 32.000 símbolos OFDM para cada uma das quatro tipos de modulação, totalizando 128.000 *frames*.

Primeiramente, nos *frames* que utilizam OFDM, os símbolos gerados são organizados em blocos de 64 símbolos digitais, modulados por uma das quatro modulações escolhidas (BPSK, QPSK, 16QAM ou 64QAM). Considerando que há uma amostra por símbolo digital, cada bloco é representado por um vetor de 64 amostras. Esse vetor é transformado em um símbolo OFDM por meio da aplicação da IFFT de 64 pontos. Em seguida, adiciona-se um CP de 16 amostras, resultando em *frames* compostos por 80 amostras. Esse procedimento foi repetido para gerar 32.000 *frames* OFDM para cada uma das quatro modulações, totalizando 128.000 *frames* ao todo.

Em contrapartida, os *frames* sem a utilização de OFDM foram construídos diretamente a partir de 80 símbolos digitais modulados. Essa quantidade foi escolhida de forma a manter o mesmo tamanho dos *frames* utilizados com OFDM, permitindo uma comparação justa entre os dois cenários. Assim, também foram gerados 128.000 *frames* para esse caso.

Para a representação dos símbolos na forma LPE utilizou-se um pulso de cosseno levantado obtido com o uso da função *rcosdesign* do *MATLAB*. Os parâmetros desse pulso foram estabelecidos com o valor de fator de rolloff de 0.5, o número de símbolos considerados em cada amostra (*span*) de 8 e o número de amostras por símbolo (*sps*) de 4. O pulso cosseno levantado utilizado no projeto pode ser visto na figura 31.

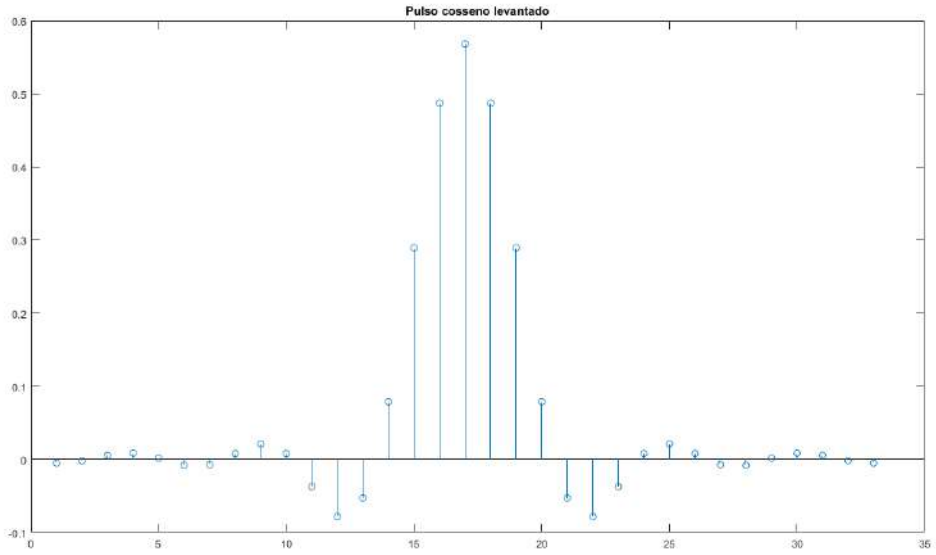

Figura 31 – Pulso cosseno levantado utilizado no projeto.

Assim, os *frames* foram submetidos primeiramente a um processo de *upsample* para possuir 4 amostras por símbolo, ou seja, adicionou-se 3 zeros a cada símbolo. Após isso, passou esse sinal com *upsample* pelo filtro do pulso cosseno levantado, obtendo os sinais

que serão transmitidos pelo canal. Esse processo de transmissão pode ser visto na figura 32.

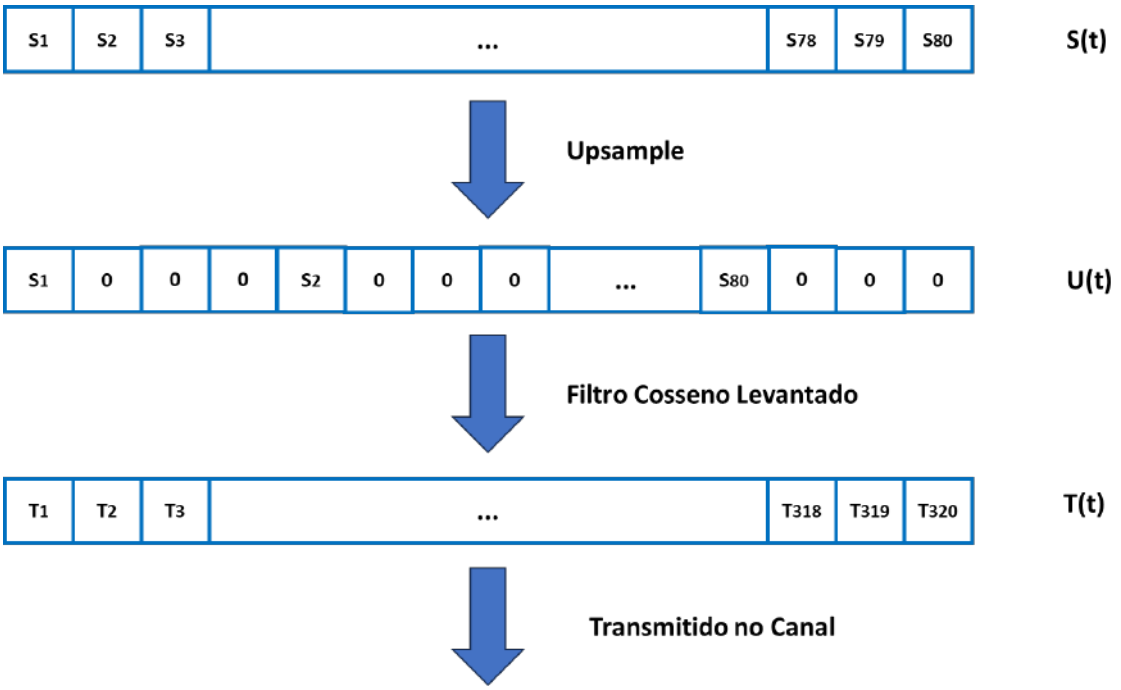

Figura 32 – Processo de transmissão dos sinais modulados.

Os *frames* gerados foram transmitidos por meio de um canal AWGN ou por um canal Rician com ruído AWGN, sob diferentes condições de SNR. Após a transmissão, o sinal recebido foi passado por um filtro com formato de pulso cosseno levantado. Em seguida, todos os atrasos introduzidos pelos filtros foram desconsiderados, e realizou-se a superamostragem do sinal, obtendo-se 4 amostras por símbolo, o que resultou em *frames* com 320 amostras. Por fim, o sinal foi normalizado para que apresentasse potência unitária. Essa normalização foi realizada da seguinte forma:

$$s_n = \frac{s_r}{E(P_r)}, \tag{2.1}$$

onde s_n é o sinal normalizado, s_r é o sinal recebido e $E(P_r)$ é a esperança da potência do sinal recebido.

Esses *frames* finais s_n foram salvos para compor o banco de dados do projeto. O processo de recepção pode também ser visto pela Figura 33.

Esse mesmo processo foi repetido com a adição de diferentes valores de ruído, ou seja, para os valores de $SNR = [5, 10, 15, 20, 25, 30]\ dB$, criando uma base de dados para cada valor. Além disso, foi repetido trocando o tipo do canal entre AWGN e Rician. Tais bases de dados foram utilizadas para o treinamento e avaliação do classificador de aprendizado de máquina projetado.

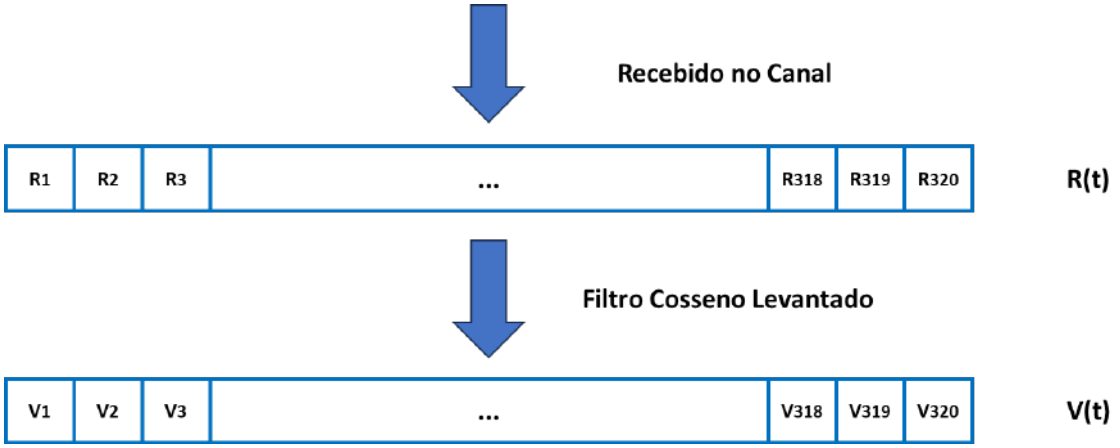

Figura 33 – Processo de recepção dos sinais modulados.

Tabela 3 – Redes neurais convolucionais treinadas no projeto e suas respectivas descrições de funcionamento

CNN	**Descrição**
$CNN_{Unificada,AWGN}$	Identifica qual a modulação Convencional ou OFDM foi utilizada em um *frame* no canal AWGN
CNN_{CxO}	Classifica o frame entre modulação Convencional ou OFDM.
CNN_{Canal}	Classifica o canal que o *frame* passou entre AWGN ou RICIAN.
$CNN_{Conv,AWGN}$	Identifica qual modulação Convencional foi utilizada em um *frame* no canal AWGN.
$CNN_{Conv,Rician}$	Identifica qual modulação Convencional foi utilizada em um *frame* no canal RICIAN.
$CNN_{OFDM,AWGN}$	Identifica qual modulação OFDM foi utilizada em um *frame* no canal AWGN.
$CNN_{OFDM,Rician}$	Identifica qual modulação OFDM foi utilizada em um *frame* no canal RICIAN.

2.3 Criação das Redes Neurais Convolucionais

O modelo computacional de aprendizado de máquina escolhido para a realização da classificação dos sinais modulados foi um conjunto de seis CNNs. Essa escolha é feita devido ao bom desempenho das CNNs em problemas que possuem entrada como sinais de comunicação, que possuem características importantes em posições adjacentes no vetor de entrada. Na tabela 3 são descritas as CNNs que foram construídas e seus respectivos funcionamentos.

Como visto na seção anterior, as bases de dados criadas possuem *frames* com a dimensão $1 \times 320 \times 2$. Esses *frames* serviram como entrada das CNNs para a realização do treinamento de classificação.

Em princípio, a $CNN_{Unificada,AWGN}$ foi utilizada para identificar a modulação

empregada em um *frame* transmitido em um canal AWGN, entre BPSK, QPSK, 16QAM, 64QAM, OFDM-BPSK, OFDM-QPSK, OFDM-16QAM e OFDM-64QAM. A estrutura usada nesse modelo foi a OMCNet desenvolvida por Thien Huynh-The [30] com pequenas adaptações. Essa estrutura pode ser vista na Figura 34.

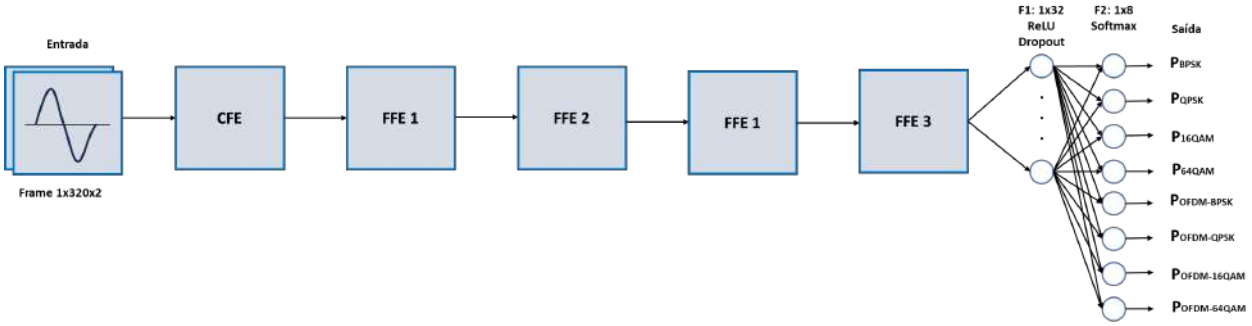

Figura 34 – Arquitetura da CNN OMCNet [31] para a classificação entre as modulações BPSK, QPSK, 16QAM, 64QAM, OFDM-BPSK, OFDM-QPSK, OFDM-16QAM e OFDM-64QAM). Os quadrados CFE (*Coarse Feature Extraction*) indicam blocos de processamento de extração de características grosseiras do sinal, os quadrados FFE (*Fine Feature Extraction*) indicam blocos de processamento de características finas do sinal e as camadas F são totalmente conectadas.

O bloco CFE (*Coarse Feature Extraction*) é utilizado para realizar a extração de características grosseiras do sinal [30]. Dessa forma, utiliza tamanhos maiores para o *kernel* das camadas convolucionais com o objetivo de capturar características presente em um conjunto de amostras consecutivas do sinal. As camadas presentes no bloco CFE pode ser visualizadas na Figura 35.

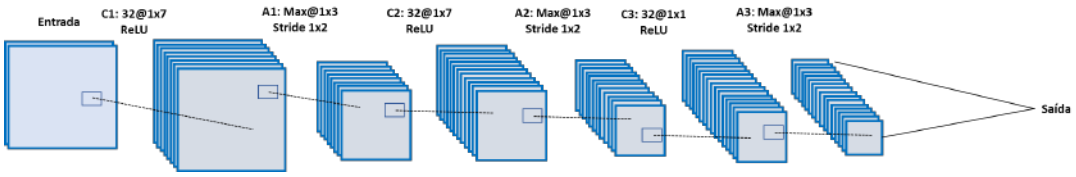

Figura 35 – Camadas presentes dentro do bloco CFE da CNN OMCNet. As camadas C indicam a utilização da operação de convolução e as camadas A indicam a utilização da operação de agrupamento.

O blocos FFE (*Fine Feature Extraction*) são utilizados para a realização de extração de características finas presentes no sinal [30]. Dessa forma, utiliza tamanhos menores para *kernel* das camadas convolucionais com a finalidade de extrair características de amostras bem próxima. Esse bloco possui 3 tipos diferentes: FFE 1, FFE 2 e FFE 3. Os blocos FFE 1 e FFE 2 possuem *kernels* assimétricos com o objetivo de reduzir a quantidade de parâmetros de treinamento. Finalmente, o bloco FF3 possui como característica a realização de função de agrupamento final de média que é utilizada como entrada para as camadas finais de classificação. Os blocos FFE podem ser vistos nas Figuras 36, 37 e 38.

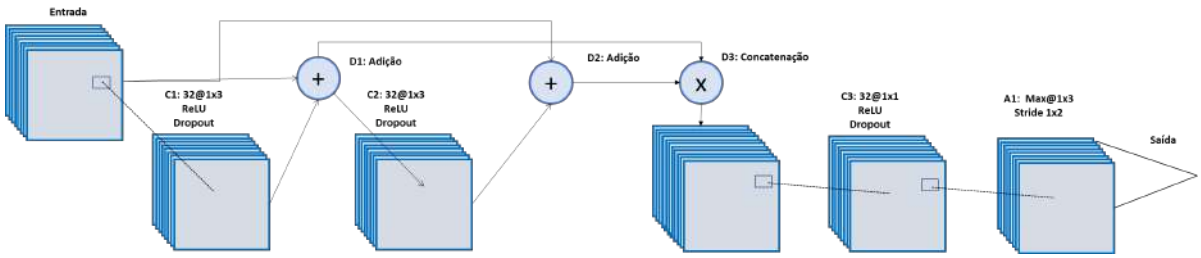

Figura 36 – Camadas presentes dentro do bloco FFE 1 da CNN OMCNet. As camadas C indicam a utilização da operação de convolução, as camadas A indicam a utilização da operação de agrupamento e as camadas D indicam operações computacionais como adição e concatenação.

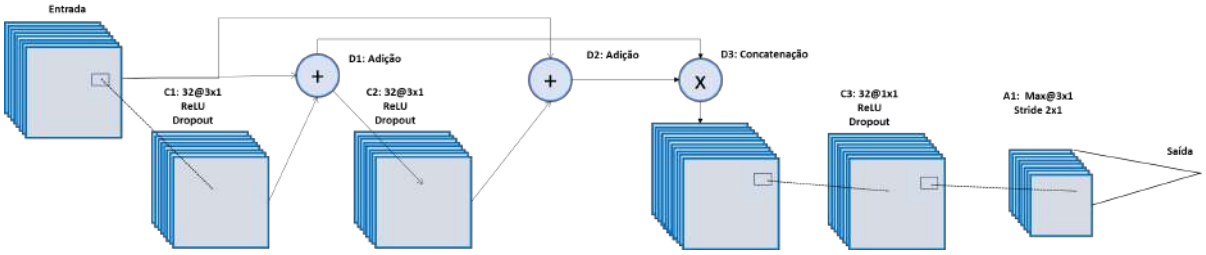

Figura 37 – Camadas presentes dentro do bloco FFE 2 da CNN OMCNet. As camadas C indicam a utilização da operação de convolução, as camadas A indicam a utilização da operação de agrupamento e as camadas D indicam operações computacionais como adição e concatenação.

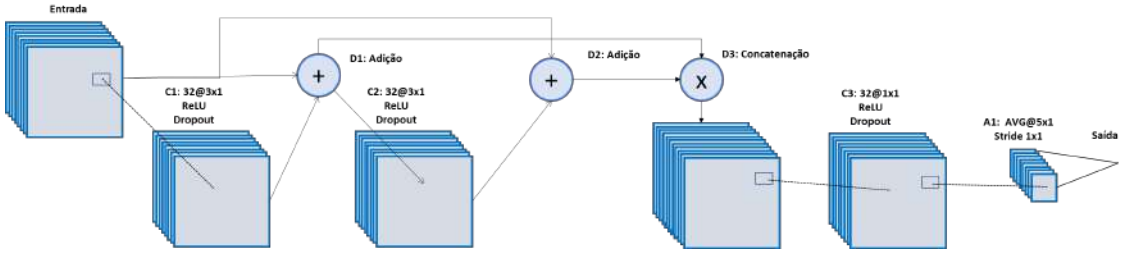

Figura 38 – Camadas presentes dentro do bloco FFE 3 da CNN OMCNet. As camadas C indicam a utilização da operação de convolução, as camadas A indicam a utilização da operação de agrupamento e as camadas D indicam operações computacionais como adição e concatenação.

Em seguida, foram criadas CNNs seguindo a estratégia de divisão e conquista. Nessa abordagem, o problema de classificação foi segmentado em subtarefas, cada uma atribuída a uma CNN específica, com o objetivo de realizar um tipo particular de classificação.

A CNN_{CxO} tem como objetivo classificar a modulação empregada no *frame* entre convencional (BPSK, QPSK, 16QAM e 64QAM) ou OFDM (OFDM-BPSK, OFDM-QPSK, OFDM-16QAM e OFDM-64QAM). Já a CNN_{Canal} tem como objetivo identificar qual o canal (AWGN ou RICIAN) que o frame foi transmitido.

A estrutura das CNNs CNN_{CxO} e CNN_{Canal} pode ser vista na Figura 39, na qual é uma simplificação da estrutura da segunda CNN que é exibida na Figura 40. A única mudança entre as estrutura dessas CNNs são as classes de saída que para CNN_{CxO} são CONV ou OFDM e para CNN_{Canal} são AWGN ou RICIAN. Tal estrutura foi escolhida por ter obtido bom desempenho em classificações de sinais digitais em trabalhos presentes nos exemplos da biblioteca *Deep Learning Toolbox* do MATLAB. A simplificação na CNN ocorre devido à menor complexidade do problema que ela resolve em comparação com aos modelos $CNN_{Conv,AWGN}$ e $CNN_{Conv,Rician}$. Essa estrutura é composta por uma sequência de camada convolucional, função de ativação *ReLU* e função de agrupamento *max-pooling* repetida duas vezes. Na terceira repetição utiliza-se a função de agrupamento da média em vez do máximo. Em seguida, encontra-se uma camada totalmente conectada e uma função de ativação *softmax* responsável por calcular a probabilidade do *frame* pertencer a cada uma das duas classes de classificação. Finalmente, na camada de saída escolhe-se a classe que possui a maior probabilidade calculada na função de ativação *softmax*.

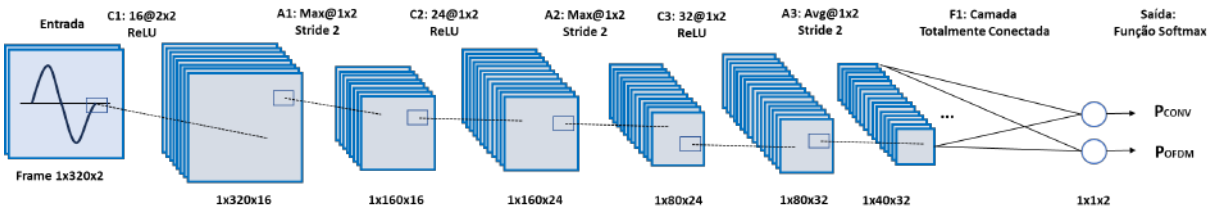

Figura 39 – Camadas presentes na CNN desenvolvida para a classificação entre modulação Convencional ou OFDM. As camadas C indicam a utilização da operação de convolução, as camadas A indicam a utilização da operação de agrupamento e as camadas F indicam as totalmente conectadas.

As CNNs $CNN_{Conv,AWGN}$ e $CNN_{Conv,Rician}$ possuem como objetivo classificar a modulação empregada no *frame* entre os diferentes tipos de modulações convencionais (BPSK, QPSK, 16QAM e 64QAM) respectivamente em um canal AWGN e em um canal Rician com ruído AWGN. A estrutura dessas CNNs é semelhante as CNNs anteriores. As únicas mudanças foram a adição de mais três repetições da sequência de camada convolucional, função de ativação *ReLU* e função de agrupamento *max-pooling* e o aumento do número de neurônios presentes na última camada para o valor da quantidade de tipos de modulações convencionais que serão preditas. Essa estrutura pode ser visualizada na

Figura 40.

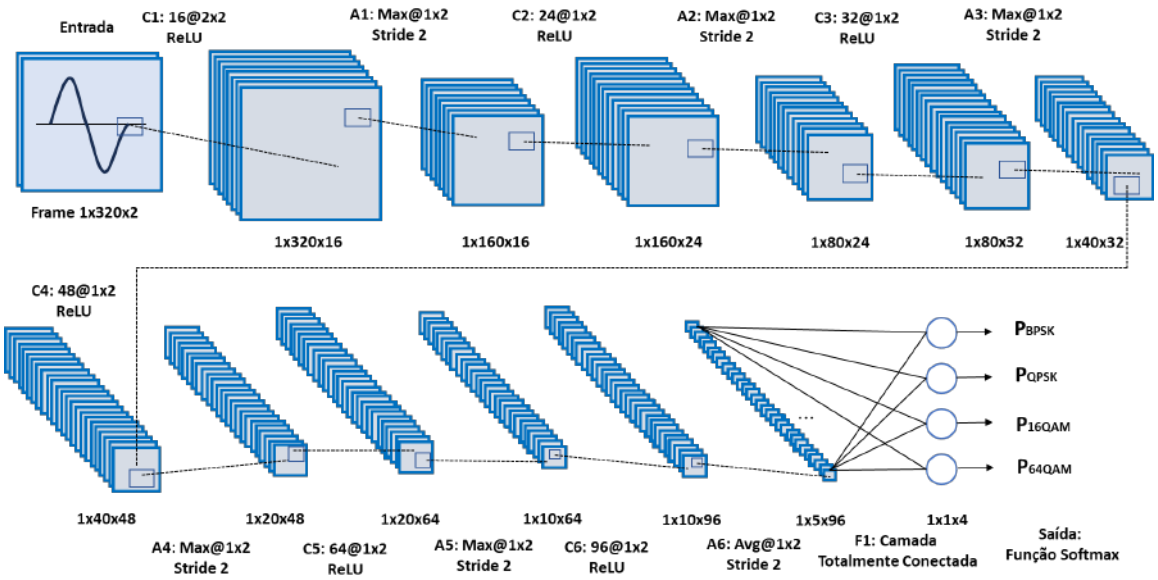

Figura 40 – Camadas presentes na CNN desenvolvida para a classificação entre as modulação convencionais (BPSK, QPSK, 16QAM e 64QAM). As camadas C indicam a utilização da operação de convolução, as camadas A indicam a utilização da operação de agrupamento e as camadas F indicam as totalmente conectadas.

As CNNs $CNN_{OFDM,AWGN}$ e $CNN_{OFDM,Rician}$ possuem como objetivo classificar a modulação empregada no frame entre os diferentes tipos de modulações OFDM (OFDM-BPSK, OFDM-QPSK, OFDM-16QAM e OFDM-64QAM) respectivamente em um canal AWGN e em um canal Rician com ruído AWGN. A estrutura usada nos modelos $CNN_{OFDM,AWGN}$ e $CNN_{OFDM,Rician}$ também foi a OMCNet com pequenas adaptações. A estrutura para esses modelos pode ser vista na Figura 41.

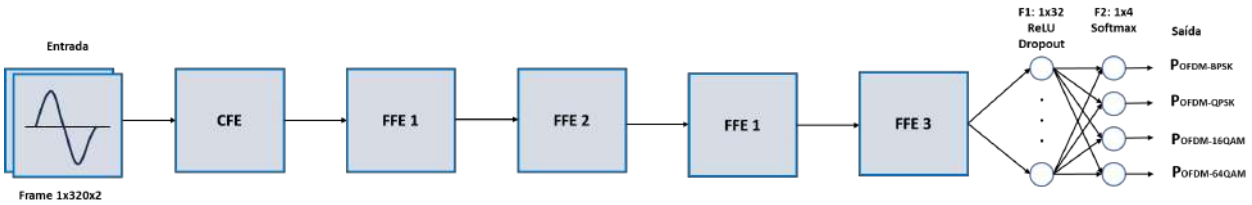

Figura 41 – Arquitetura da CNN OMCNet [31] para a classificação entre as modulação OFDM (OFDM-BPSK, OFDM-QPSK, OFDM-16QAM e OFDM-64QAM). Os quadrados CFE (*Coarse Feature Extraction*) indicam blocos de processamento de extração de características grosseiras do sinal, os quadrados FFE (*Fine Feature Extraction*) indicam blocos de processamento de características finas do sinal e as camadas F são totalmente conectadas.

Os modelos de CNN descritos nessa seção foram criados através da biblioteca de *deep learning* presente no MATLAB. Com o uso dessa biblioteca é possível construir o modelo através da descrição das camadas que serão utilizadas e de suas respectivas dimensões.

Para cada uma das CNNs desenvolvidas, foi necessário construir uma base de dados específica para seu treinamento e teste. Para o modelo CNN_{CxO}, os *frames* foram separados em duas categorias: modulações convencionais e modulações com OFDM. Já para o modelo CNN_{Canal}, os dados foram organizados conforme o tipo de canal de propagação, distinguindo entre *frames* transmitidos em canal AWGN e *frames* transmitidos em canal Rician. Nos modelos $CNN_{Conv,AWGN}$ e $CNN_{Conv,Rician}$, foram utilizadas apenas amostras provenientes de modulações convencionais transmitidas, respectivamente, pelos canais AWGN e Rician. De forma análoga, para os modelos $CNN_{OFDM,AWGN}$ e $CNN_{OFDM,Rician}$, a base de dados foi composta exclusivamente por *frames* das modulações OFDM transmitidas pelos respectivos canais.

2.4 Treinamento das Redes Neurais Convolucionais

Todas as CNNs seguiram um mesmo processo de treinamento, na qual a única diferença está nas classes utilizadas durante a separação das bases de dados de cada CNN. A seguir será descrito o processo de treinamento incluindo o algoritmo e os parâmetros utilizados.

Em princípio, as bases de dados foram separadas em 80% dos sinais para serem utilizadas no treinamento, 10% para a validação e 10% para testes. Entretanto, para realizar o treinamento da CNN utilizou-se somente os sinais com $SNR = 30\ dB$ no conjunto de treinamento e de validação, dado que sinais com uma *SNR* muito baixa acarreta em que o efeito do ruído seja muito significante para a estrutura do sinal e dificulta a classificação dos sinais com menor ruído.

Antes da realização do treinamento, o modelo encontra-se com pesos aleatórios nas conexões entre os neurônios. Assim, o modelo foi treinado com uso do algoritmo ADAM para conseguir realizar a classificação das modulações. Esse treinamento foi configurado de acordo com os parâmetros exibidos na Tabela 4 para os modelos CNN_{CxO}, CNN_{Canal} e $CNN_{Conv,Rician}$, na Tabela 5 para o modelo $CNN_{Conv,AWGN}$ e na Tabela 6 para os modelos $CNN_{OFDM,AWGN}$, $CNN_{OFDM,Rician}$ e $CNN_{Unificada,AWGN}$.

Note que os treinamentos dos dois primeiros conjunto de CNNs utilizam uma quantidade menor de *frames* para treinamento por apresentarem problemas com menor complexidade. Já o terceiro conjunto de CNNs utilizaram uma quantidade maior de *frames* devida a complexidade dos sinais OFDM.

O treinamento possui um número máximo de épocas que ocorrem iterações a cada época. Assim, a cada iteração é feita uma atualização dos pesos em direção a descida do gradiente para uma amostra dos conjuntos de treinamento. Nessa atualização utilizam um momento (contribuição do passo anterior) de 90%. No fim de uma época, os resultados das iterações são reunidos para uma avaliação da acurácia no conjunto de validação

Tabela 4 – Parâmetros utilizados no treinamento da CNN_{CxO}, da CNN_{Canal} e da $CNN_{Conv,Rician}$.

Parâmetros	**Valor**
Algoritmo de treinamento	ADAM
Momento	90%
Taxa de aprendizado inicial	0,01
Método de ajuste da taxa	*piecewise*
Número máximo de épocas	60
Número de *frames*	16.000
Tamanho do minilote (*MiniBatchSize*)	256

Tabela 5 – Parâmetros utilizados no treinamento da $CNN_{Conv,AWGN}$.

Parâmetros	**Valor**
Algoritmo de treinamento	ADAM
Momento	90%
Taxa de aprendizado inicial	0,01
Método de ajuste da taxa	*piecewise*
Número máximo de épocas	60
Número de *frames*	8.000
Tamanho do minilote (*MiniBatchSize*)	256

Tabela 6 – Parâmetros utilizados no treinamento da $CNN_{OFDM,AWGN}$, da $CNN_{OFDM,Rician}$ e $CNN_{Unificada,AWGN}$.

Parâmetros	**Valor**
Algoritmo de treinamento	ADAM
Momento	90%
Taxa de aprendizado inicial	0,1
Método de ajuste da taxa	*piecewise*
Número máximo de épocas	60
Número de *frames*	128.000
Tamanho do minilote (*MiniBatchSize*)	256

através do minilote. Após todas as épocas do treinamento, a rede neural da época que obteve maior desempenho no conjunto de validação, isto é, menor valor na função perda, é retornada como a rede de saída do treinamento. Assim, o parâmetro de tamanho do minilote (*MiniBatchSize*) indica a quantidade de sinais do conjunto de treinamento que serão utilizadas para avaliar o gradiente da função de perda e atualizar os pesos.

Outro parâmetro importante utilizado no treinamento foi o método *piecewise* de atualização da taxa de aprendizado. Nesse método, a cada 15 épocas a taxa de aprendizado é multiplicada por 0,1. Desta forma, a taxa de aprendizado inicial de 0,1 é reduzida a cada 15 épocas para auxiliar o treinamento a atingir ajustes mais finos e alcançar mínimos na função de perda.

Após o treinamento da rede neural, pode-se observar sua acurácia na classificação

da modulação dos sinais presentes no conjunto de testes e também observar quais foram os erros de classificação através da matriz de confusão para cada base de dados de uma SNR específica.

2.5 Criação do Classificador de Modulação

Após a elaboração das CNNs e seus respectivos treinamentos, essas redes foram utilizadas em conjunto para a criação de um modelo de classificação de modulação digital.

O classificador de modulação proposto neste projeto pode ser visto no diagrama de blocos da Figura 42.

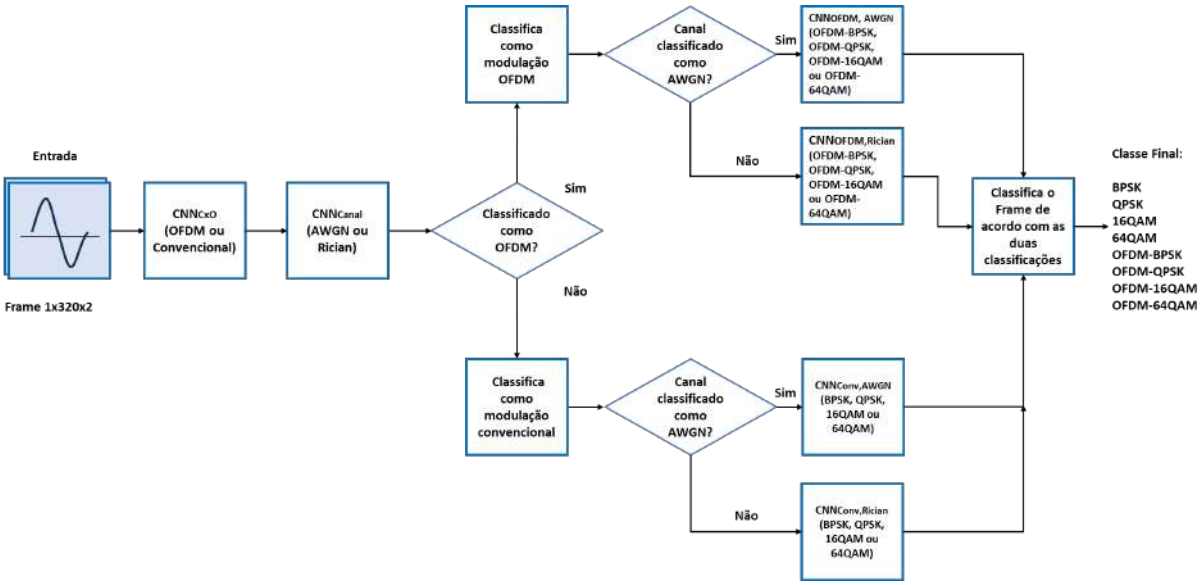

Figura 42 – Diagrama de blocos do classificador de modulação proposto no projeto.

Um *frame* ao ser apresentado na entrada desse classificador, primeiramente, irá passar pela classificação da CNN_{CxO}. Essa CNN tem como objetivo classificar o frame entre modulação OFDM e convencional. Sendo assim, a primeira classificação concedida ao *frame*.

Logo após, a CNN_{Canal} vai identificar qual o canal que foi utilizado na transmissão do *frame*. Essa classificação irá auxiliar no uso da CNN correta para classificação da modulação.

Em seguida, deseja-se encontrar a modulação que foi empregada nos símbolos digitais do *frame*. Os *frames* classificados como modulação convencionais irão ser classificados pela $CNN_{Conv,AWGN}$ quando o canal for classificado pela CNN_{Canal} como AWGN e pela $CNN_{Conv,Rician}$ caso contrário. Em contrapartida, nos *frames* classificados como modulação OFDM irão ser classificados pela $CNN_{OFDM,AWGN}$ quando o canal for classificado pela CNN_{Canal} como AWGN e pela $CNN_{OFDM,Rician}$ caso contrário. Portanto, a classificação final é obtida com a classe obtida na classificação nas CNN finais de classificação do tipo

de modulação. O desempenho dessa classificação também será avaliado através de conjunto de testes com *frame* simulados e reais.

2.6 Teste de Desempenho do Classificador de Modulação

O último passo no desenvolvimento do classificador de modulação do projeto foi o teste de desempenho. O teste de desempenho foi realizado com o cálculo da acurácia e a elaboração da matriz de confusão para conjuntos de testes de *frames* obtidos pela simulação ou por um SDR.

Para os *frames* simulados, a base de dados de teste foi obtida com a separação de 10% feita durante o treinamento das CNNs. Nesse caso, não foi necessária a execução de nenhuma outra etapa adicional. Assim, as métricas foram calculadas a partir da comparação entre a classe predita no classificador obtida e a classe real.

Agora, para os *frames* reais foi necessário o uso do SDR bladeRF xA9. A bladeRF xA9 foi conectada ao *software* MATLAB com o uso da biblioteca *Communications Toolbox Support Package for BladeRF 2.0* e a duas antenas, uma para transmissão e outra para a recepção do *frames*. Esse arranjo experimental pode ser visualizado na Figura 43. Dado o arranjo experimental apresentado, os passos apresentados na Figura 44 guiaram a criação dos *frames* reais.

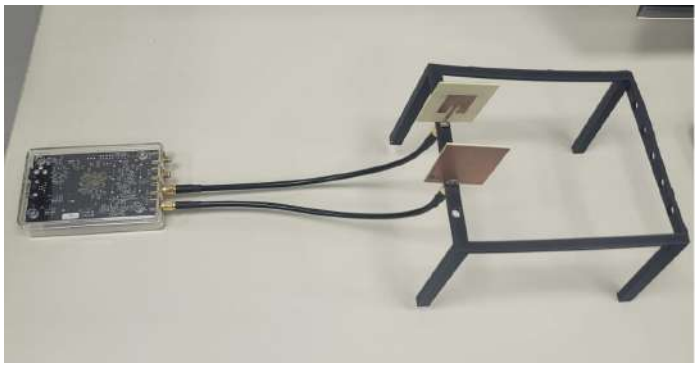

Figura 43 – Arranjo experimental utilizado para o processo de transmissão e recepção dos *frames* reais com o SDR.

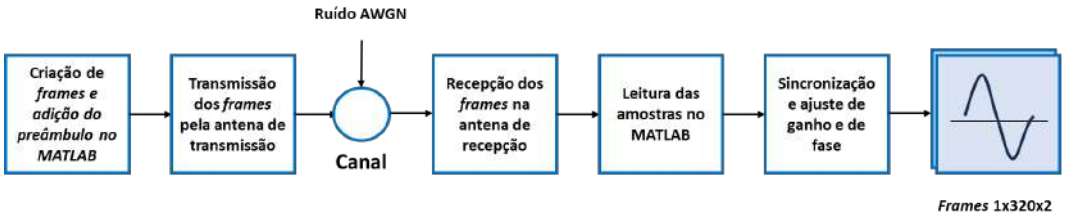

Figura 44 – Diagrama de blocos das etapas do processo de criação dos *frames* reais com o SDR.

Primeiramente, os *frames* reais foram criados de uma forma semelhante que os simulados, na qual aconteceram somente algumas diferenças nos processos de transmissão e de recepção. Os *frames* transmitidos no canal utilizaram os mesmos parâmetros e processos nas modulações e o mesmo filtro de cosseno levantado. A primeira grande diferença ocorre no canal, trocando assim o canal simulado por uma transmissão real pelo ar. Outra diferença é a adição de um preâmbulo contendo o código de Barker de comprimento 11 antes da transmissão dos *frames*, que possui o objetivo de ajudar na sincronização dos *frames*.

No processo de recepção dos *frames* ocorreram algumas mudanças em relação aos simulados. Após a passagem do sinal recebido por um filtro de recepção de cosseno levantado, acontece a primeira grande mudança. A primeira grande mudança é a execução de um processo de sincronização através do uso da operação de autocorrelação para encontrar o início em que os *frames* estão presentes no sinal recebido. Desta forma, após encontrar o preâmbulo, é descartado a parte do sinal em que ele se encontra e a leitura é feita nas amostras seguintes. Na Figura 45 são apresentadas as amplitudes das componentes de um frame real recebido pelo SDR, no qual é possível observar o código de Barker seguido pelo sinal modulado.

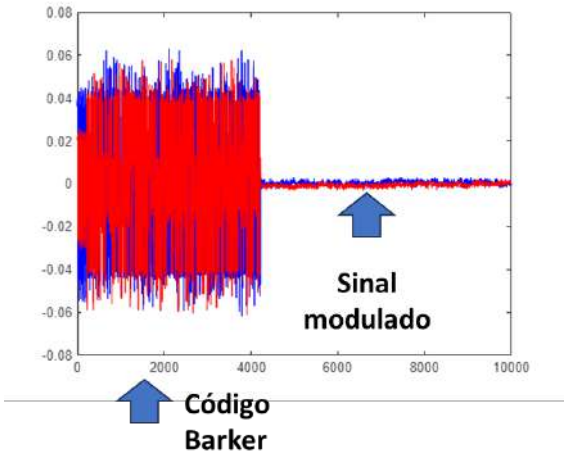

Figura 45 – Exibição da amplitude das componentes de um *frame* real recebido pelo SDR.

Além do sincronismo, com o uso do preâmbulo é executada uma correção de fase, na qual a fase do sinal é modificada de acordo com a rotação que o preâmbulo sofreu durante a recepção. No sinal recebido também acontece uma correção de ganho, normalizando a potência do sinal para um valor unitário. Após esse processo, os *frames* são normalizados da mesma maneira que os simulados e são salvos para que possam ser usados no teste do classificador. As constelações dos sinais durante a execução desse processo pode ser vista na Figura 46.

Com a obtenção dos *frames* reais, o desempenho do classificador de modulação é testado com os frames através do uso da métrica de acurácia e da obtenção da matriz de

Figura 46 – Constelações de um *frame* em modulação QPSK recebido pelo SDR, após a aplicação dos processos de correção de fase e normalização de potência.

confusão.

2.7 Verificação do Uso de Sincronização

Conforme mencionado anteriormente, uma sequência de preâmbulo foi inserida na transmissão dos *frames* reais com o objetivo de auxiliar no processo de sincronização na recepção. Durante esse processo, foi realizada uma correção da fase do sinal recebido, de modo que ele apresentasse a mesma fase do sinal transmitido. Vale destacar que as CNNs foram treinadas com *frames* que não possuem rotação de fase. Por esse motivo, foi conduzido um experimento adicional no qual os *frames* transmitidos por canal AWGN foram multiplicados por um fator complexo $e^{j\phi}$, simulando uma rotação de fase arbitrária no intervalo de 0 a 2π radianos, a fim de avaliar a robustez dos modelos diante de variações de fase.

As CNNs treinadas com os novos *frames* contendo rotação de fase foram avaliadas em dois conjuntos distintos de testes. O primeiro conjunto é composto por *frames* simulados, gerados da mesma forma que os utilizados no treinamento, ou seja, com rotação de fase aplicada. O segundo conjunto é formado por *frames* reais adquiridos com o equipamento SDR, mas sem a aplicação de correção de fase na recepção, de modo a verificar a capacidade dos modelos em lidar com sinais reais afetados por variações de fase.

O segundo teste relacionado ao uso de sincronização consistiu na aplicação de um atraso no início da amostragem do sinal recebido. Considerando que cada *frame* é composto por 320 amostras, com 4 amostras por símbolo digital, foi criado um novo conjunto de teste aplicando um atraso aleatório de 0 a 3 amostras. Esse deslocamento simula uma condição de erro de sincronização temporal na recepção do sinal.

As CNNs treinadas com os novos *frames* contendo atraso de amostragem também foram avaliadas em dois conjuntos de testes. O primeiro conjunto é composto por *frames* simulados, gerados da mesma forma que os utilizados no treinamento, ou seja, com o

atraso de amostragem aplicado. O segundo conjunto é formado por *frames* reais adquiridos com o equipamento SDR e com a aplicação de um atraso de amostragem em relação a posição considerada sincronizada com o uso do preâmbulo.

As CNNs treinadas com os novos *frames* contendo atraso de amostragem também foram avaliadas em dois conjuntos de testes. O primeiro conjunto é composto por *frames* simulados, gerados da mesma forma que os utilizados no treinamento, ou seja, com o atraso de amostragem incorporado. O segundo conjunto é composto por *frames* reais adquiridos com o equipamento SDR, nos quais foi aplicada uma defasagem no início da amostragem em relação à posição considerada sincronizada com base no código de preâmbulo.

3 Resultados e Discussão

Com a realização da metodologia separada nas etapas descritas anteriormente neste trabalho, foi possível elaborar um classificador para realizar a classificação da modulação utilizada em *frames* de sinais.

Nessa seção serão exibidos e discutidos os resultados obtidos com a aplicação da metodologia do projeto.

3.1 Treinamento das Redes Neurais Convolucionais

Inicialmente, as CNNs foram configuradas com pesos de conexão entre os neurônios definidos aleatoriamente. Ao aplicar o procedimento de treinamento das redes neurais, os pesos foram ajustados para obtenção de um melhor desempenho de classificação. Todos os treinamentos foram feitos utilizando o conjunto de *frames* específico para respectivo modelo, conforme foi descrito na seção anterior, contendo *frames* com SNR de 30 dB.

O treinamento do modelo $CNN_{Unificado,AWGN}$, que é uma rede convolucional única que tem como objetivo classificar a modulação de um *frame* entre as modulações BPSK, QPSK, 16QAM, 64QAM, OFDM-BPSK, OFDM-QPSK, OFDM-16QAM e OFDM-64QAM, foi realizado utilizando um conjunto de dados específico para essa tarefa, composto por *frames* transmitidos em um canal AWGN. O desempenho durante o treinamento é ilustrado na Figura 47. Observa-se que o modelo atingiu uma acurácia de 74,98% no conjunto de validação na 48ª época. Veja também que que o treinamento atingiu uma convergência, pois o desempenho da rede se manteve constante desde a quinta época, dado que a função de perda apresentou uma redução mínima nesse período.

O treinamento do modelo CNN_{CxO}, responsável por classificar se um *frame* utiliza modulação convencional ou OFDM, foi realizado utilizando um conjunto de dados específico para essa tarefa, composto por *frames* transmitidos por dois tipos de canais: parte dos *frames* passou por um canal AWGN, enquanto outra parte foi transmitida por um canal Rician com AWGN. O desempenho durante o treinamento é ilustrado na Figura 48. O modelo atingiu uma acurácia de 98,86% no conjunto de validação na 18ª época. Além disso, é notável que o treinamento atingiu a convergência, pois o desempenho da rede se manteve constante desde a terceira época.

Analogamente, o treinamento do modelo CNN_{Canal}, responsável por identificar o tipo de canal que o *frame* foi transmitido, pode ser visualizado na Figura 49. Esse treinamento foi realizado com uma base composta por *frames* de diferentes modulações convencionais e OFDM transmitidos em canais AWGN e Rician com AWGN, em proporções

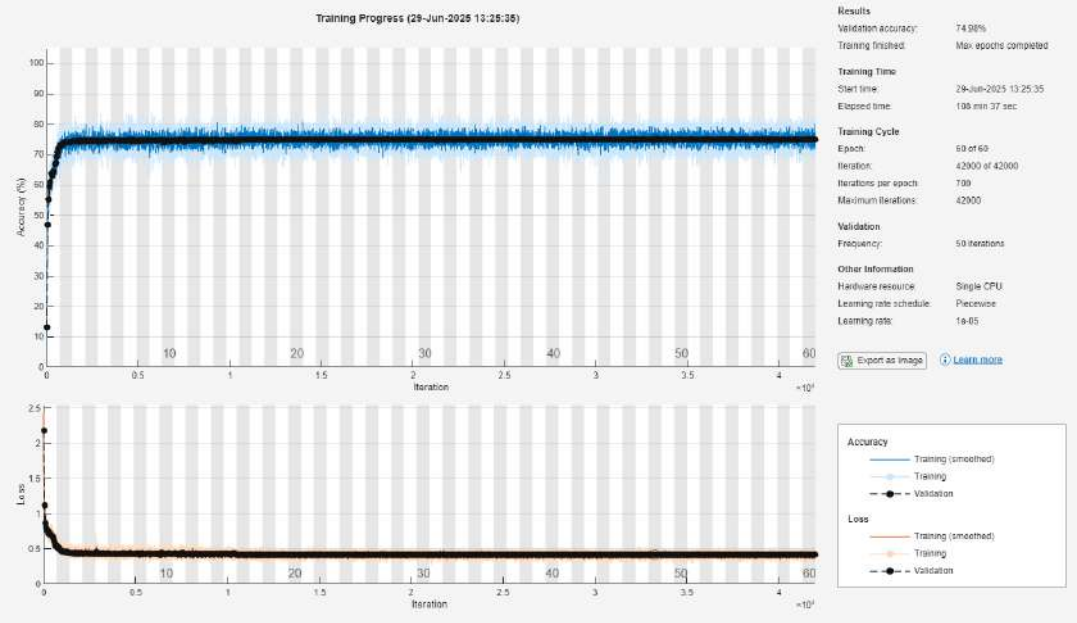

Figura 47 – Treinamento de classificação da $CNN_{Unificada,AWGN}$.

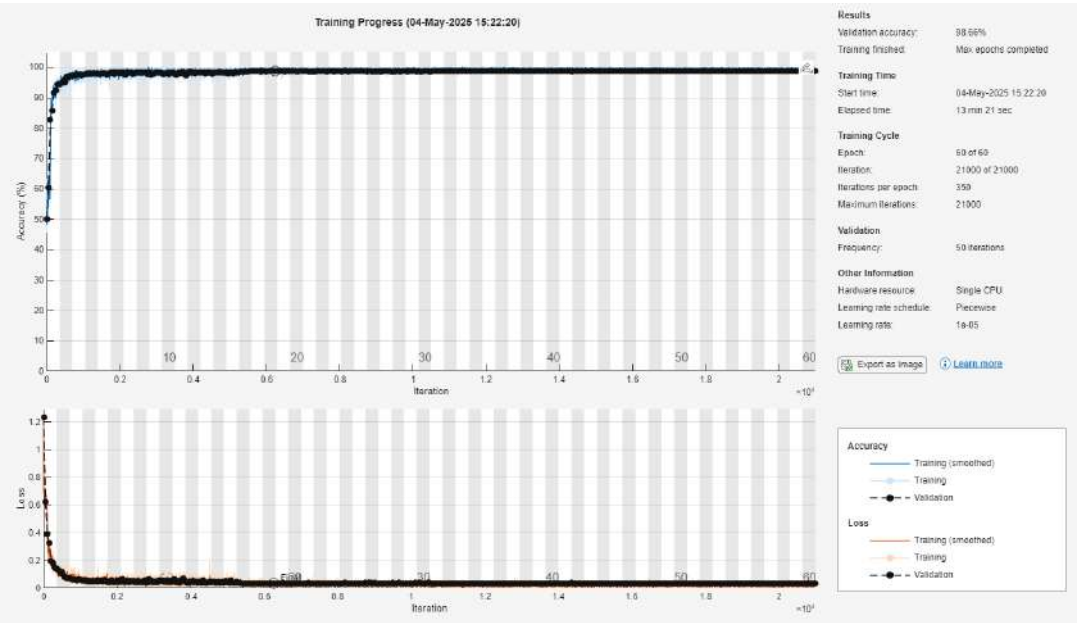

Figura 48 – Treinamento de classificação da CNN_{CxO}.

iguais. Tal treinamento atingiu uma acurácia de 99,49% no conjunto de validação na época 33, correspondente a uma condição de SNR de 30 dB. Além disso, é notável que o treinamento atingiu a convergência, pois o desempenho da rede se manteve constante desde a 18ª época.

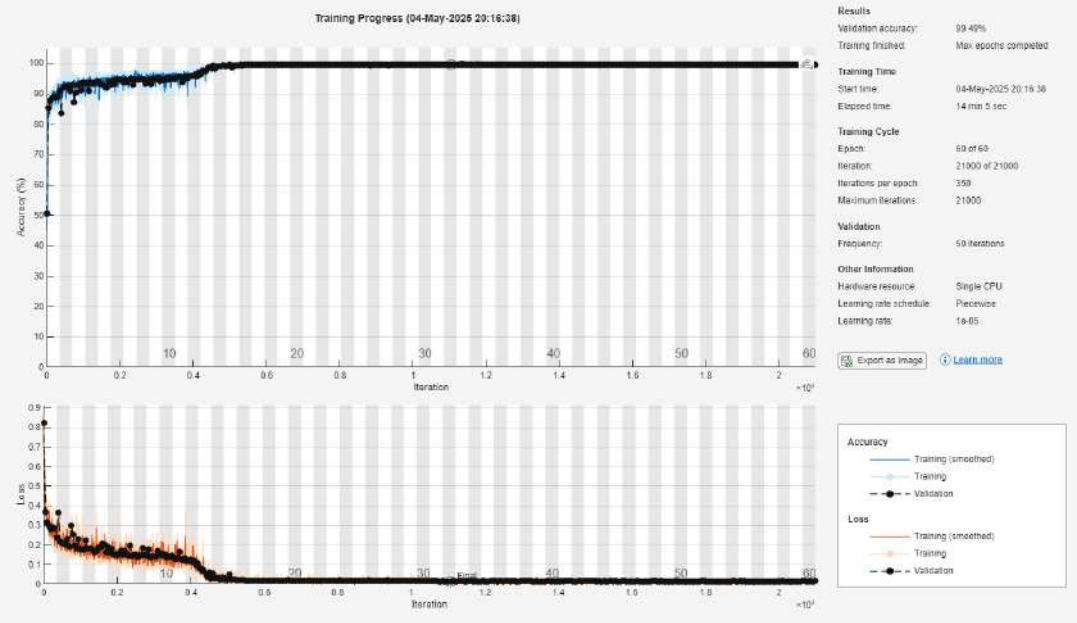

Figura 49 – Treinamento de classificação da CNN_{Canal}.

O treinamento da $CNN_{Conv,AWGN}$, responsável por classificar o tipo de modulação digital convencional empregada nos símbolos do *frame* em um canal AWGN, pode ser visto na Figura 50. Esse treinamento foi realizado com uma base composta por *frames* de diferentes modulações convencionais transmitidos em um canal AWGN. Tal treinamento atingiu uma acurácia de 100% no conjunto de validação na época 11. Além disso, analogamente ao que foi dito na rede anterior, é notável que o treinamento também atingiu a convergência. Desta vez, a convergência ocorreu em torno da época 3, representando que esse problema foi de fácil treinamento para a CNN.

Agora, com o canal Rician, o treinamento da $CNN_{Conv,Rician}$ pode ser visto na Figura 51. Esse treinamento foi realizado com uma base composta por *frames* de diferentes modulações convencionais transmitidos em canais Rician. Tal treinamento atingiu uma acurácia de 99,58% no conjunto de validação na época 25.

O treinamento da $CNN_{OFDM,AWGN}$, responsável por classificar o tipo de modulação OFDM empregada nos símbolos do *frame* em um canal AWGN, pode ser visto na Figura 52. Esse treinamento foi realizado com uma base composta por *frames* de diferentes modulações OFDM transmitidos em um canal AWGN. Tal treinamento atingiu uma acurácia de 99% no conjunto de validação na época 43. Esse treinamento atingiu a convergência em torno da época 20.

Por fim, com o canal Rician, o treinamento da $CNN_{OFDM,Rician}$ pode ser visto na

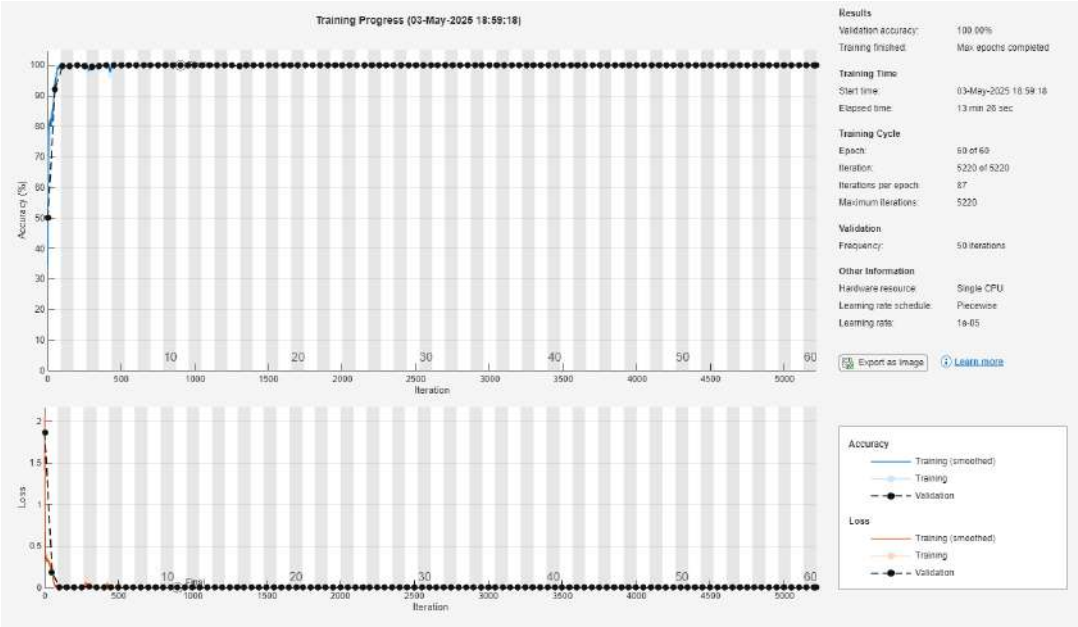

Figura 50 – Treinamento de classificação da $CNN_{Conv,AWGN}$.

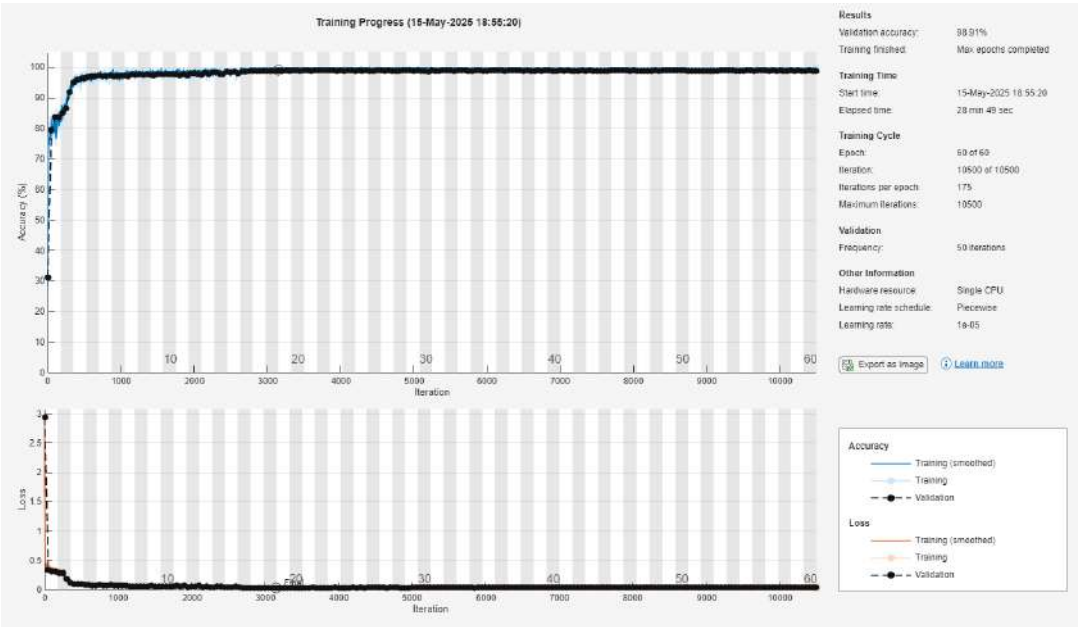

Figura 51 – Treinamento de classificação da $CNN_{Conv,Rician}$.

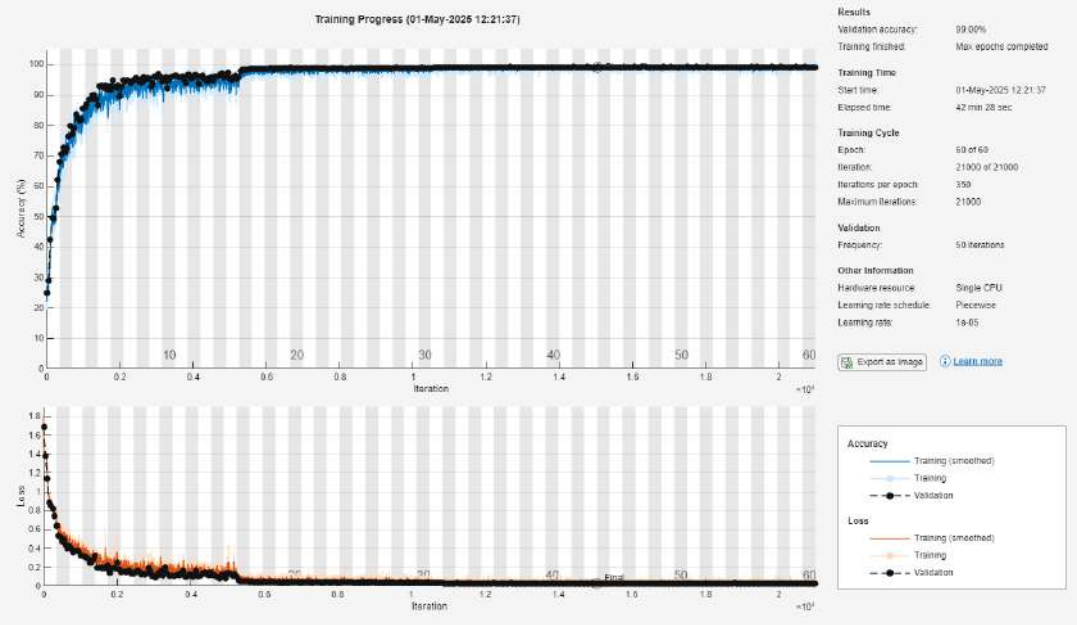

Figura 52 – Treinamento de classificação da $CNN_{OFDM,AWGN}$.

Figura 53. Esse treinamento foi realizado com uma base composta por *frames* de diferentes modulações OFDM transmitidos em canais Rician. Tal treinamento atingiu uma acurácia de 77,47% no conjunto de validação na época 60.

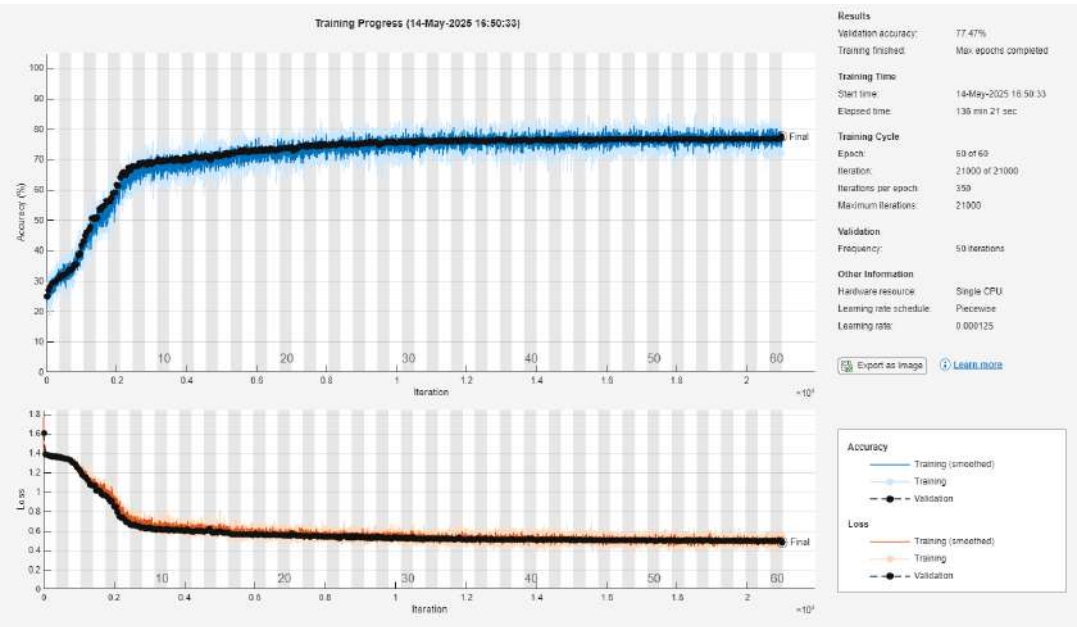

Figura 53 – Treinamento de classificação da $CNN_{OFDM,Rician}$.

3.2 Teste de Desempenho das Redes Neurais Convolucionais

As CNNs treinadas foram submetidas a aferição de acurácia de classificação no conjunto de testes de *frames* simulados. A aferição foi feita separando os *frames* de acordo com a sua SNR.

O desempenho da CNN_{CxO} é exibido na Figura 54. Veja que essa CNN conseguiu uma boa acurácia de 94,16% mesmo com uma SNR baixa de 5 dB. Além disso, para uma SNR de 30 dB, a acurácia atingiu 97,29%, o que pode ser considerado um bom desempenho para a identificação se o *frame* utiliza ou não a técnica OFDM.

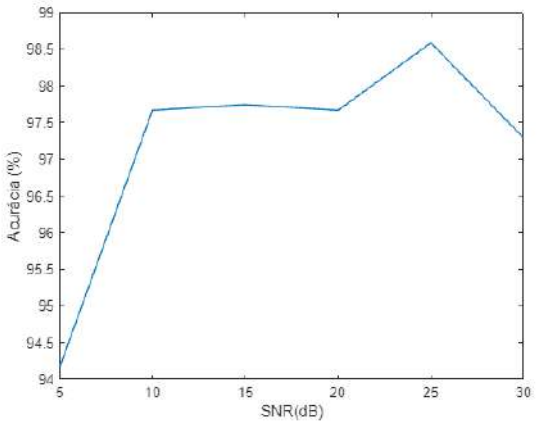

Figura 54 – Acurácia da CNN_{CxO} no conjunto de teste de *frames* simulados.

Ao testar o desempenho da CNN_{Canal}, foi obtido o resultado presente na Figura 55. Note que essa CNN obteve uma acurácia menor em comparação com a CNN anterior em SNRs mais baixas como 5 dB com 89,45% de acurácia. Em contrapartida, esse modelo atingiu um excelente desempenho na identificação do tipo do canal em SNRs maiores ou iguais a 15 dB. Para uma SNR de 30 dB, a acurácia atingiu 99,59%.

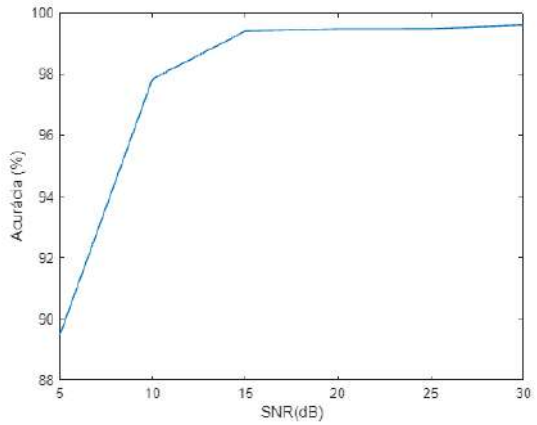

Figura 55 – Acurácia da CNN_{Canal} no conjunto de teste de *frames* simulados.

Agora, o teste de desempenho da $CNN_{Conv,AWGN}$ é mostrado na Figura 56. Diferentemente que as primeiras CNNs, esta obteve um mau desempenho em *frames* com SNR baixa, tendo uma acurácia de 50,67% quando a SNR é de 5 dB. Em contrapartida, em SNRs maiores ou iguais a 15 dB, a acurácia atingiu o valor de 100%.

Para a $CNN_{Conv,Rician}$, temos o resultado exibido na Figura 57. Para a SNR de 30 dB, a acurácia atingiu 99,02%.

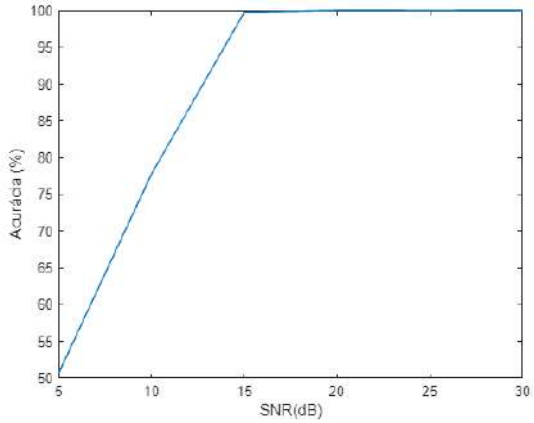

Figura 56 – Acurácia da $CNN_{Conv,AWGN}$ no conjunto de teste de *frames* simulados.

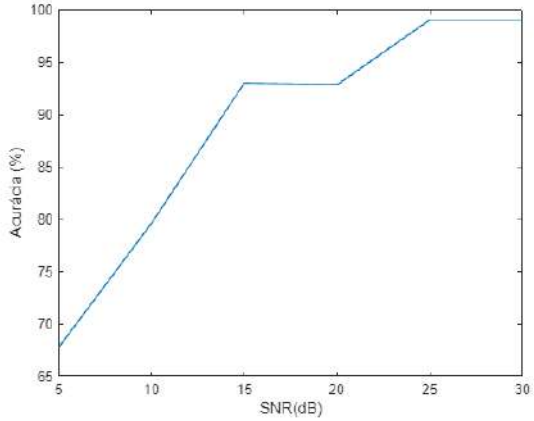

Figura 57 – Acurácia da $CNN_{Conv,Rician}$ no conjunto de teste de *frames* simulados.

Agora, o teste de desempenho da $CNN_{OFDM,AWGN}$ é mostrado na Figura 58. Essa CNN foi uma das que obteve menor desempenho em sinais com SNR mais baixas, sendo a acurácia de 38,3% para SNR de 5 dB. Porém, em SNRs mais altas a acurácia apresentou bons valores. Para os *frames* com SNR de 30 dB a CNN alcançou uma acurácia de 99,06%.

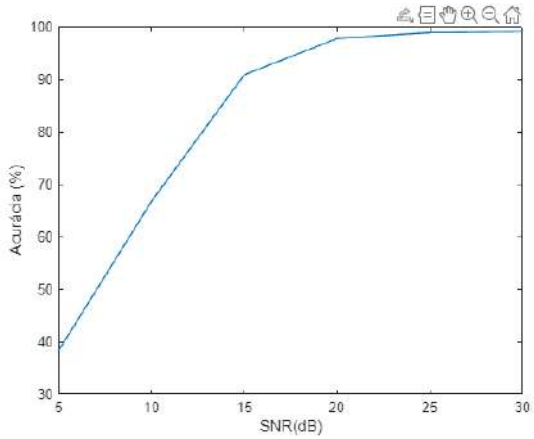

Figura 58 – Acurácia da $CNN_{OFDM,AWGN}$ no conjunto de teste de *frames* simulados.

Finalmente, com o teste de desempenho da $CNN_{OFDM,Rician}$ obteve-se o resultado

exibido na Figura 59. Tal CNN apresentou o pior desempenho em relação às demais CNNs. A acurácia da CNN foi 34,38% para SNR de 5 dB e 72,13% para SNR de 30 dB.

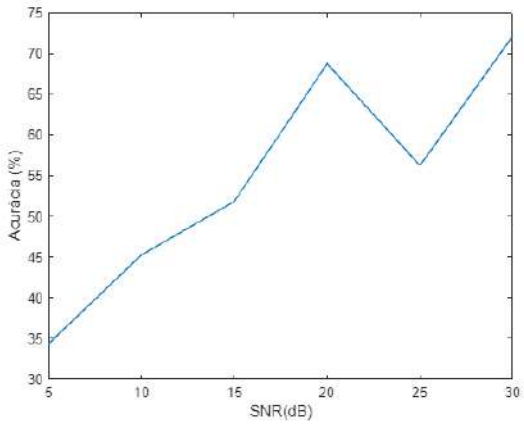

Figura 59 – Acurácia da $CNN_{OFDM,Rician}$ no conjunto de teste de *frames* simulados.

Outra maneira de comparar o desempenho dos modelos foi visualizando suas respectivas matrizes de confusão exibidas nas Figuras 60, 61, 62, 63, 64 e 65. Na matriz de confusão são exibidas as quantidades de amostras que possuem uma classificação real (*True Class*) e em qual das classes que elas foram preditas (*Predicted Class*) no conjunto de teste com os *frames* simulados.

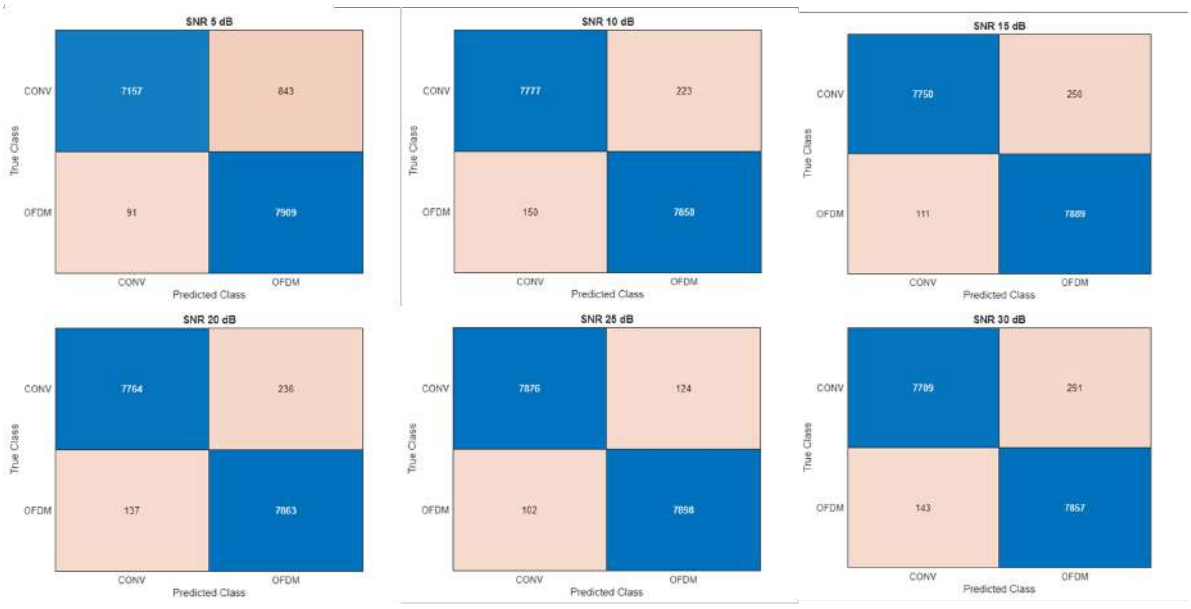

Figura 60 – Matriz de confusão da classificação do conjunto de testes de *frames* simulados feita na CNN_{CxO}.

Observando as matrizes de confusão da CNN_{CxO} e da CNN_{Canal}, nota-se que houveram somente alguns erros de classificação independemente do valor da SNR. O aumento da SNR melhorou levemente o desempenho da classificação.

Em contraste, na matriz de confusão da $CNN_{Conv,AWGN}$ pode-se perceber que o aumento da SNR afeta fortemente o desempenho dessa rede. Em SNRs baixas, a única

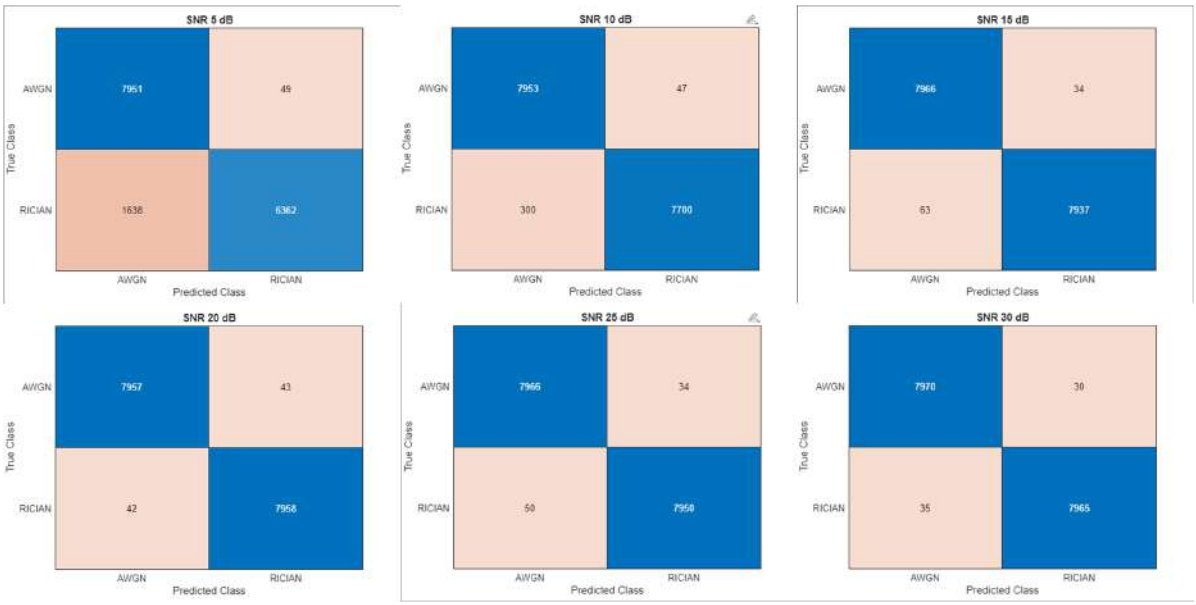

Figura 61 – Matriz de confusão da classificação do conjunto de testes de *frames* simulados feita na CNN_{Canal}.

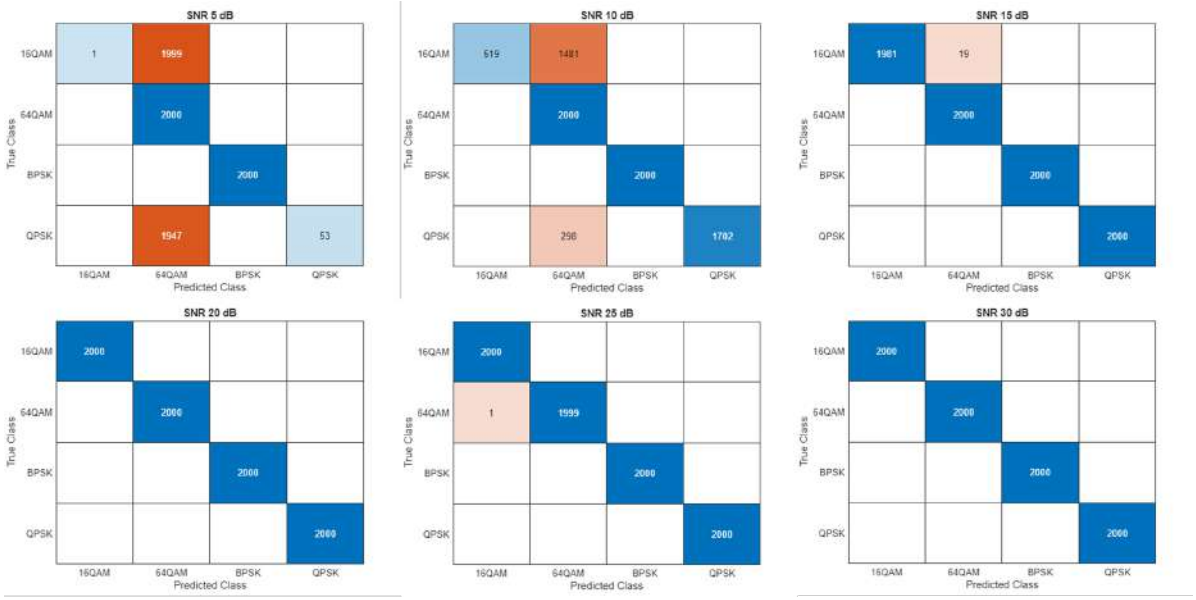

Figura 62 – Matriz de confusão da classificação do conjunto de testes de *frames* simulados feita na $CNN_{Conv,AWGN}$.

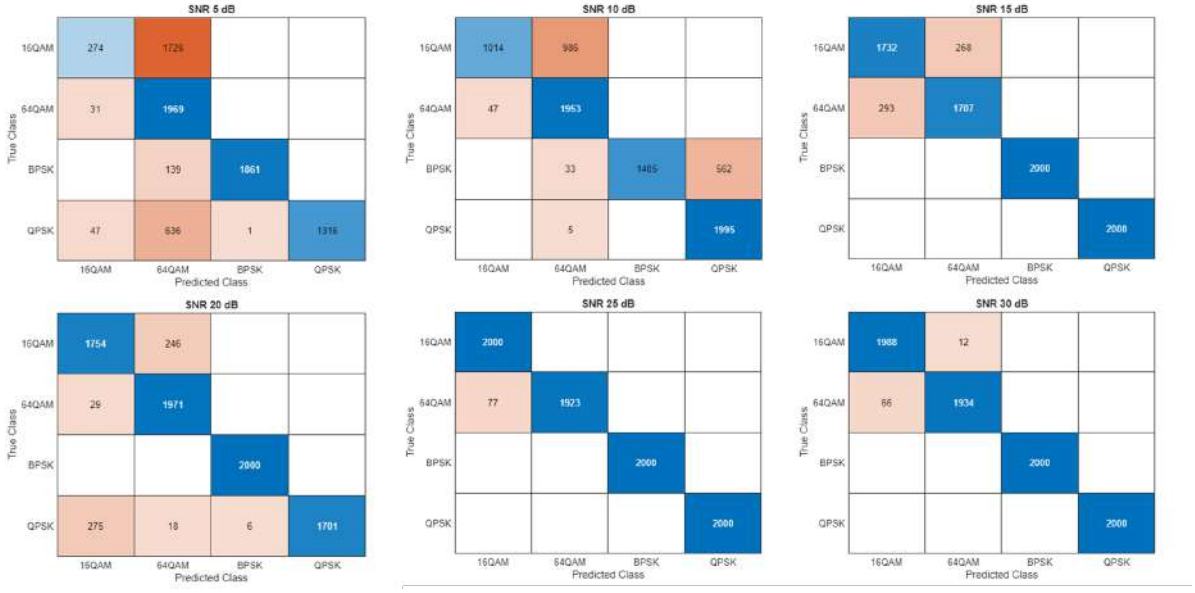

Figura 63 – Matriz de confusão da classificação do conjunto de testes de *frames* simulados feita na $CNN_{Conv,Rician}$.

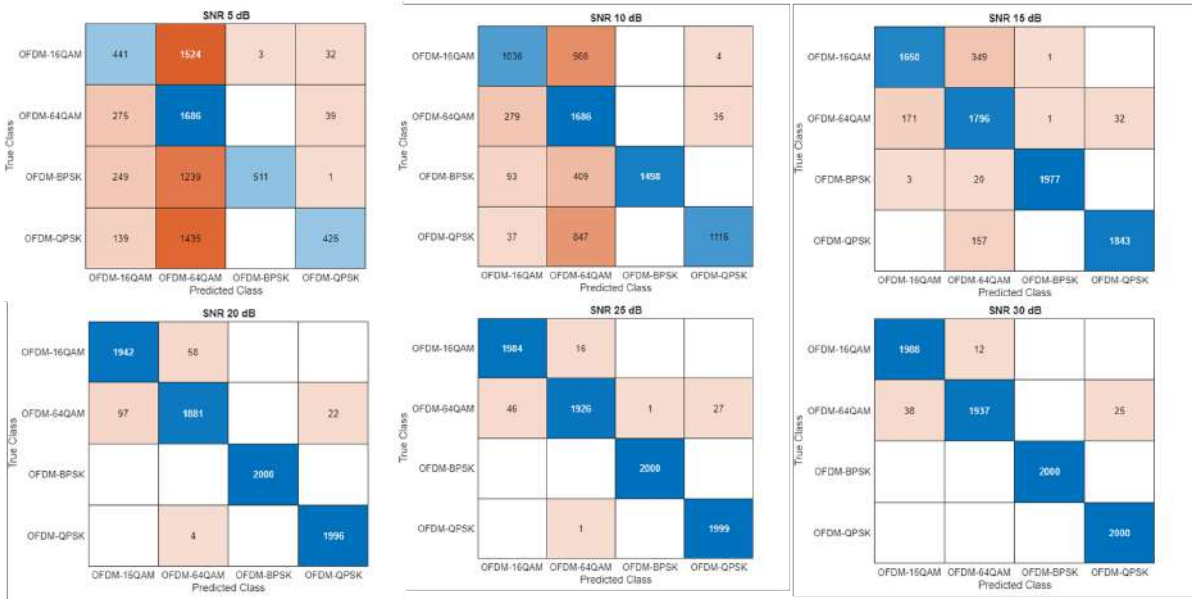

Figura 64 – Matriz de confusão da classificação do conjunto de testes de *frames* simulados feita na $CNN_{OFDM,AWGN}$.

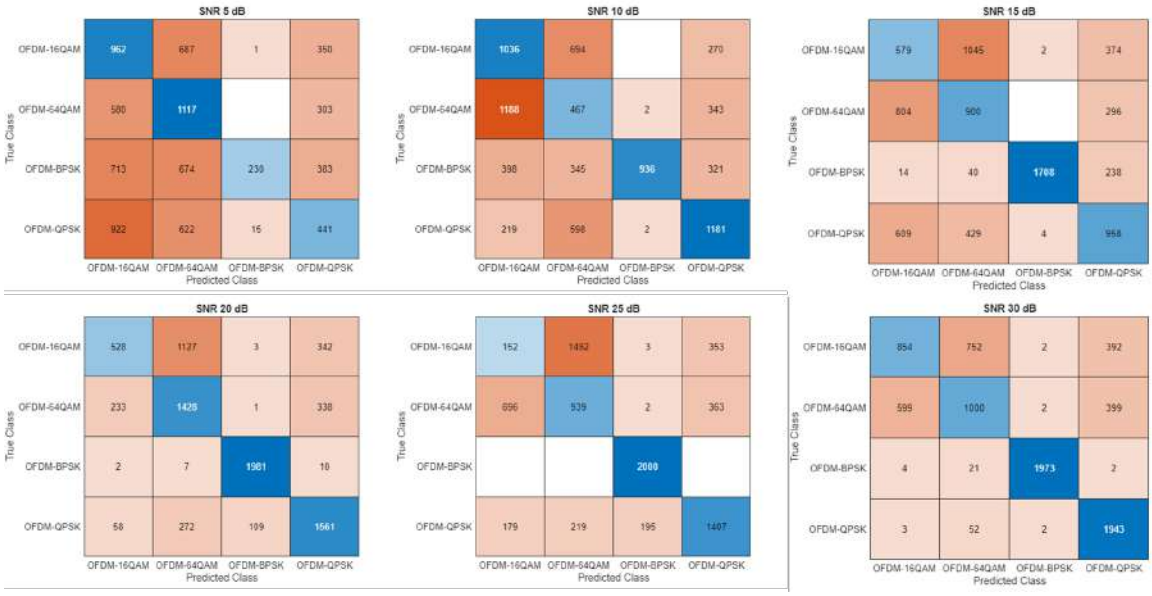

Figura 65 – Matriz de confusão da classificação do conjunto de testes de *frames* simulados feita na $CNN_{OFDM,Rician}$.

classe que a rede acertou sem grandes dificuldades foi a modulação BPSK. Isso deve-se a sua simplicidade em relação às demais modulações. Na medida que a SNR foi aumentando, a dificuldade se concentrou entre a classificação de 16QAM e 64QAM. A partir da SNR de 15 dB, o modelo conseguiu classificar todas as classes com uma acurácia alta.

Agora, na matriz de confusão da $CNN_{Conv,Rician}$ pode-se perceber que as confusões de classificação ficaram principalmente entre as modulações QAM.

Na matriz de confusão da $CNN_{OFDM,AWGN}$ é notável ver que seu desempenho em SNRs baixas foi o menor entre as CNNs. Porém, em *frames* com SNR de 30 dB acontecem equivocações, principalmente, entre as classes OFDM-16QAM e OFDM-64QAM.

Por fim, na matriz de confusão da $CNN_{OFDM,Rician}$ nota-se que o desempenho de classificação foi bem pior em relação às demais CNNs. Com a SNR de 30 dB, o desempenho da CNN foi bom para as classes OFDM-BPSK e OFDM-QPSK, mas teve um mau desempenho entre as classes OFDM-16QAM e OFDM-64QAM. Ou seja, não conseguiu diferenciar as modulações QAM.

Diante dos resultados, conclui-se que a melhoria da qualidade do sinal transmitido contribui significativamente para o desempenho das redes. Esse aprimoramento ocorre, principalmente, devido à diferenciação entre as modulações 16QAM e 64QAM, que exigem uma SNR mais elevada para garantir a correta identificação de seus símbolos. Além disso, nota-se que os modelos projetados para canal Rician apresentaram dificuldade na classificação de modulações QAM. Desta forma, recomenda-se utilizar as modulações PSK quando utilizar canal Rician e OFDM.

3.3 Teste do Classificador de Modulação

Em princípio, foi utilizado o modelo $CNN_{Unificada,AWGN}$ para classificar *frames* transmitidos em um canal AWGN, adotando-se, assim, uma única CNN para realizar todo o processo de classificação. O resultado do teste de classificação pode ser observado no gráfico da Figura 66, no qual a acurácia obtida para uma SNR de 30 dB foi de 74,94%.

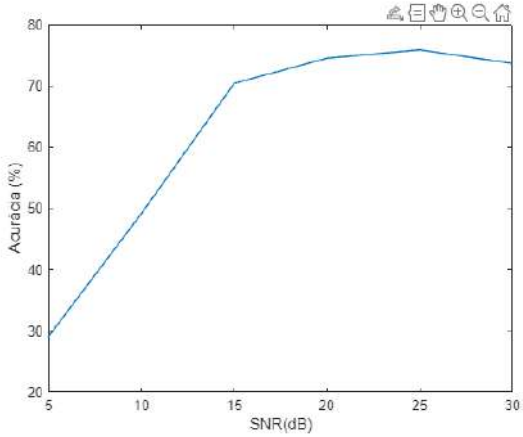

Figura 66 – Acurácia da $CNN_{Unificada,AWGN}$ no conjunto de teste de *frames* simulados.

Note que a acurácia do modelo $CNN_{Unificada,AWGN}$ não foi tão elevada, mesmo com os *frames* sendo transmitidos em um canal AWGN com alta SNR. A matriz de confusão, apresentada na Figura 67, revela que o classificador teve dificuldades especialmente na identificação dos *frames* com modulação OFDM. Esse resultado sugere que o problema de classificação apresenta um grau de complexidade elevado, tornando desafiador o uso de uma única CNN para realizar essa tarefa com alto desempenho.

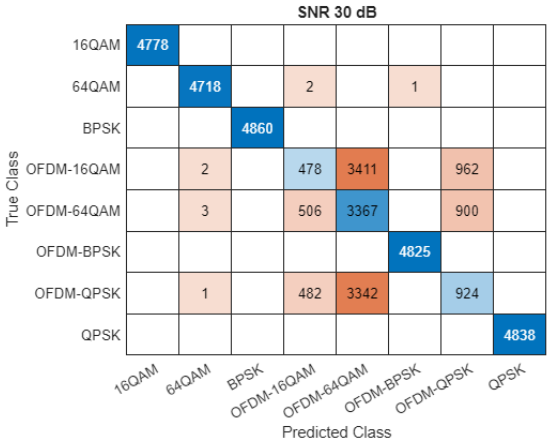

Figura 67 – Matriz de confusão da classificação do conjunto de testes de *frames* simulados com SNR de 30 dB feita na $CNN_{Unificada,AWGN}$.

Assim, a estratégia de divisão e conquista foi implementada por meio do treinamento de CNNs treinadas com subproblemas menores e da construção do classificador final

utilizando essas CNNs. Essa abordagem foi testada tanto com *frames* simulados no MATLAB quanto com *frames* reais adquiridos por meio do equipamento SDR.

Com o classificador construído obteve-se o gráfico da Figura 68 com a acurácia obtida com os *frames* simulados transmitidos por um canal AWGN. Perceba que a acurácia obtida melhora de acordo com o aumento da SNR, obtendo uma acurácia de 98,83% com *frames* de SNR 30 dB. Além disso, é possível notar a partir da SNR 20 dB há uma convergência para uma acurácia em torno de 98%, indicando uma boa faixa de valor para utilizar o classificador.

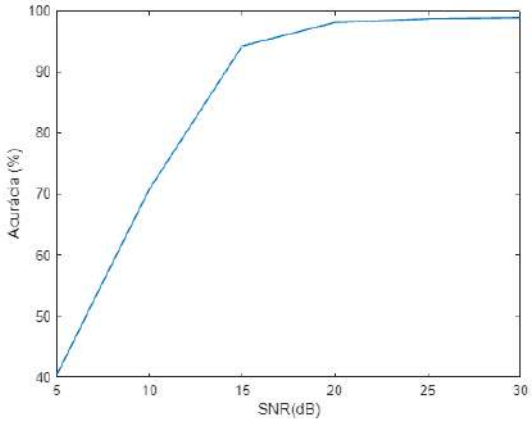

Figura 68 – Acurácia do classificador no conjunto de teste de *frames* simulados em um canal AWGN.

As matrizes de confusão desse teste na simulação pode ser visualizada na Figura 69. Nas matrizes, pode-se observar que, em SNRs mais baixas, ocorrem dificuldades no classificador, principalmente, na classificação entre 16QAM e 64QAM e entre as modulações OFDM. Em SNRs mais altas, o classificador apresentou alta acurácia, com apenas alguns erros na classificação.

O segundo teste com dados simulados foi com um conjunto de *frames* que foram transmitidos por canais Rician. O desempenho do classificador nesse conjunto é exibido na Figura 70. Veja que a acurácia do classificador ficou em torno de 77% para os *frames* com SNR maiores ou iguais a 15 dB.

Pela matriz de confusão desse teste presente na Figura 71, nota-se que o desempenho do classificador no canal multipercurso Rician é inferior para o caso que o canal é AWGN. A maior dificuldade de classificação ficou entre as classes OFDM-16QAM e OFDM-64QAM. Dessa forma, o desempenho do classificador para canais Rician pode ser melhorado utilizando *frames* somente OFDM-16QAM ou OFDM-64QAM.

Em seguida, em um terceiro teste com *frames* simulados, montou-se um conjunto misturando *frames* que foram transmitidos em canal AWGN com *frames* transmitidos em canais Rician. Para esse teste, assim como é mostrado na Figura 72, foi obtida uma

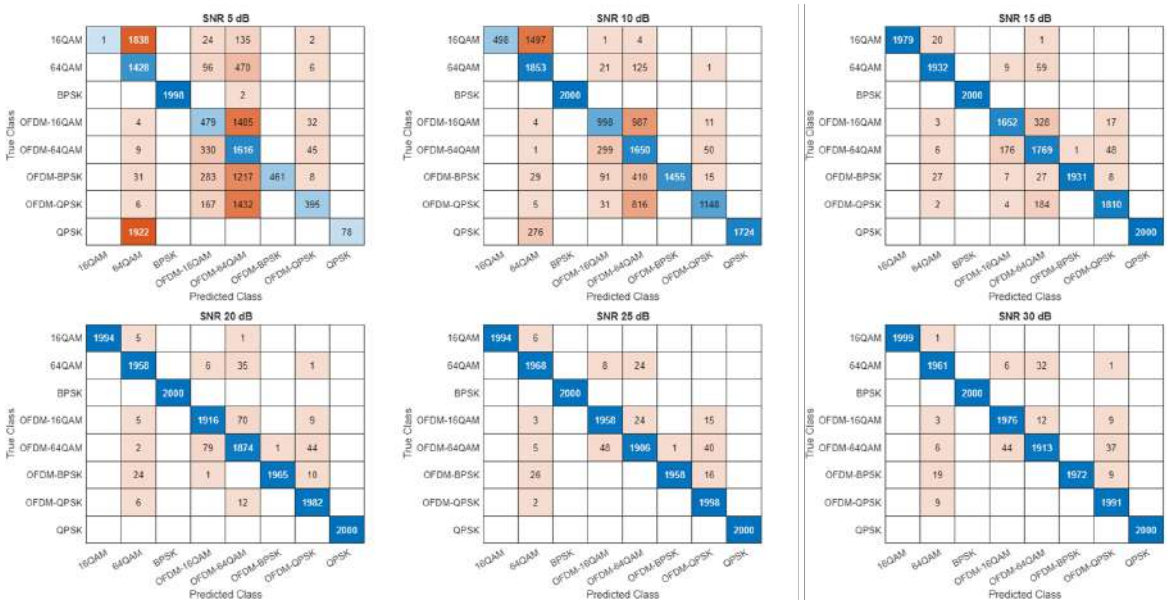

Figura 69 – Matriz de confusão do classificador no conjunto de testes de *frames* simulados em um canal AWGN.

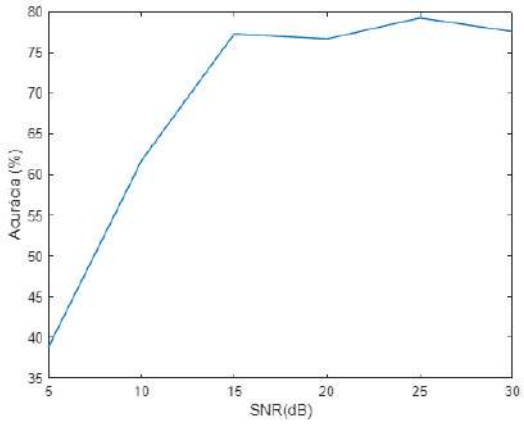

Figura 70 – Acurácia do classificador no conjunto de teste de *frames* simulados em canais Rician.

acurácia em torno de 87% para *frames* com $SNR \geq 15$ dB.

Finalmente, nas matrizes de confusão desse teste presentes na Figura 73, observa-se que a acurácia melhorou por causa da inserção dos *frames* transmitidos em canal AWGN, porém ainda estão presentes as mesma dificuldades de classificação encontradas durante a classificação dos demais conjuntos.

Da mesma maneira, com a realização do teste com o conjunto de *frames* reais, em princípio, utilizou-se um conjunto de *frames* reais sem a aplicação da correção da rotação utilizando o código Barker. As Figuras 74 e 75 mostram o mau desempenho que o classificador obteve em classificar esses *frames* reais obtidos pelo SDR. Esse mau desempenho deve-se principalmente aos canais Rician utilizados durante o treinamento não representaram um canal parecido com o canal real. Desta forma, foi utilizada a correção

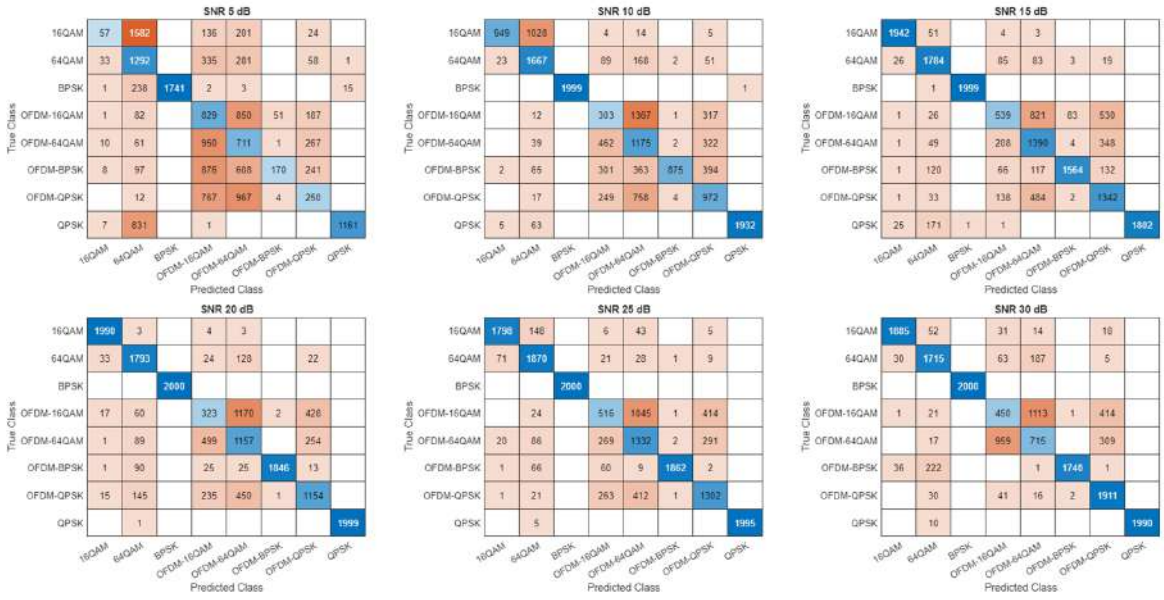

Figura 71 – Matriz de confusão do classificador no conjunto de testes de *frames* simulados em canais Rician.

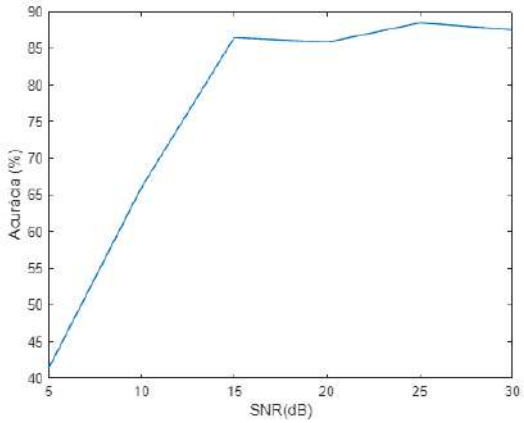

Figura 72 – Acurácia do classificador no conjunto de teste de *frames* simulados em canais Rician e AWGN.

da rotação utilizando o código Barker para que o efeito do canal se assemelhe com um canal AWGN. Assim, o classificador conseguirá melhorar seu desempenho.

Logo, com a aplicação da correção de rotação nos *frames*, obteve-se a acurácia exibida no gráfico da Figura 76. Perceba que com essa correção, o classificador consegue realizar as classificações com uma melhor acurácia, obtendo uma acurácia de 96,5% com *frames* de SNR 30 dB.

As matrizes de confusão desse teste com o uso do SDR pode ser visualizada na Figura 77. Nas matrizes, pode-se observar que, em SNRs mais baixas, também ocorrem dificuldades na classificação entre 16QAM e 64QAM e a classificação errada de frames com modulações convencionais em modulações OFDM. Assim como na simulação, em SNRs mais altas, o classificador apresentou boa acurácia, com somente alguns erros,

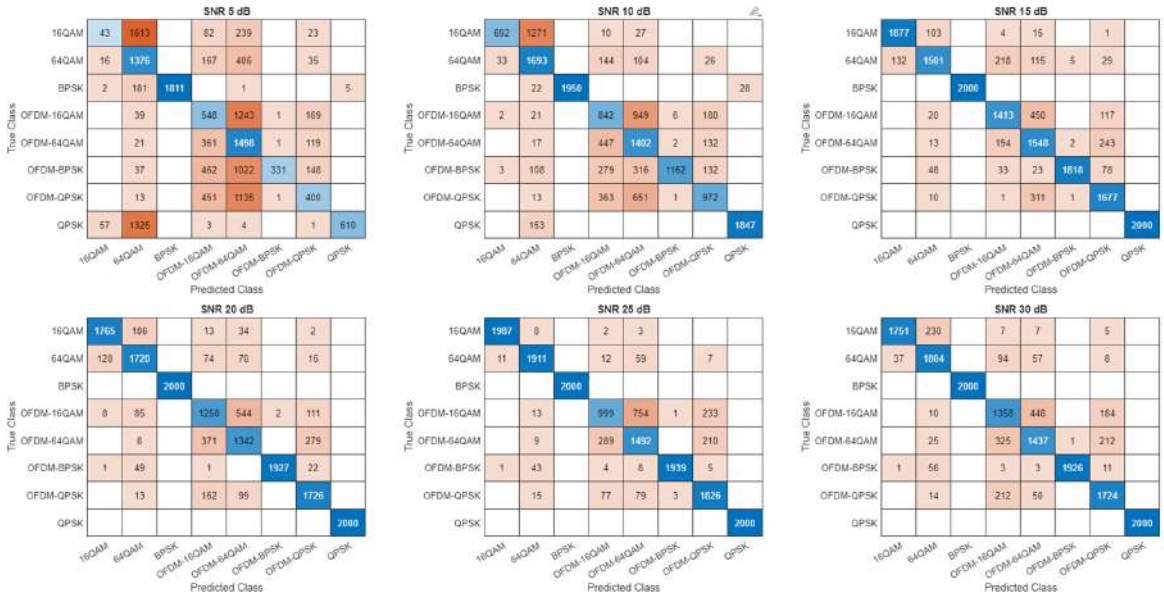

Figura 73 – Matriz de confusão do classificador no conjunto de testes de *frames* simulados em canais Rician e AWGN.

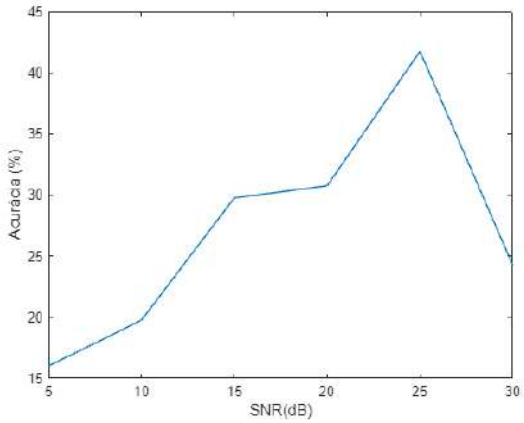

Figura 74 – Acurácia do classificador no conjunto de teste de *frames* reais sem utilização de correção da rotação.

principalmente, na distinção entre as modulações OFDM-16QAM e OFDM-64QAM.

3.4 Resultados do Desempenho do Classificador de Modulação

Na seção anterior foram expostos os resultados do desempenho do classificador construído nesse projeto. Na Tabela 7 estão reunidos os resultados de acurácia para os diferentes conjuntos de *frames* de 30 dB simulados ou reais que foram submetidos nos testes de desempenho do classificador.

Note que o desempenho do classificador nos *frames* reais com correção de rotação e SNR de 30 dB alcançou uma boa acurácia somente 2,33% menor que os *frames* simulados em um canal AWGN. Porém, em SNRs menores a acurácia da classificação em *frames*

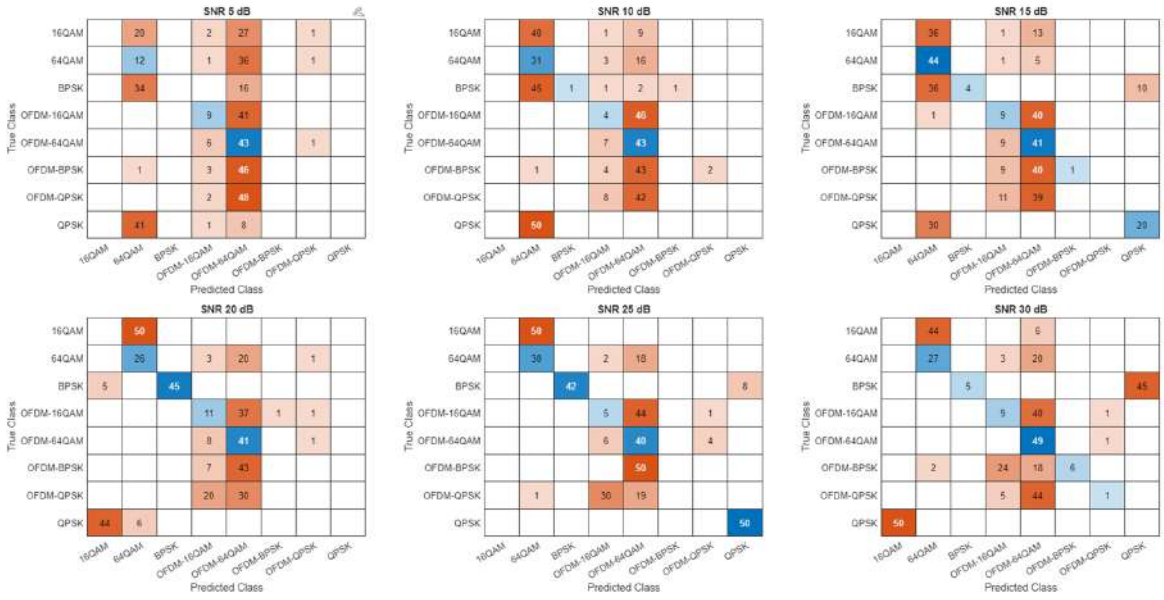

Figura 75 – Matriz de confusão do classificador no conjunto de testes de *frames* reais sem utilização de correção da rotação.

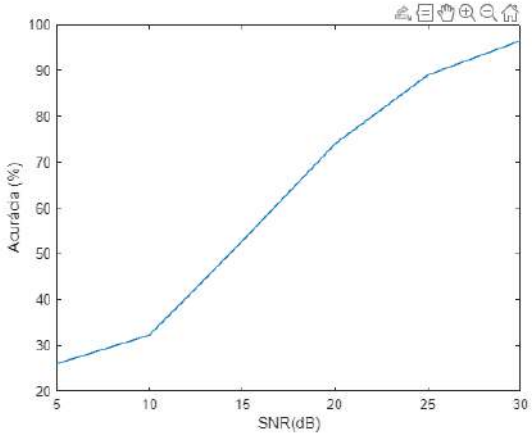

Figura 76 – Acurácia do classificador no conjunto de teste de *frames* reais.

reais baixou consideravelmente, sendo para SNR de 25 dB de 89% nos *frames* reais e de 98,64% nos *frames* simulados. Assim, para a utilização desse classificador em *frames* reais recomenda-se utilizar sinais que tenham uma alta SNR de pelo menos 30 dB.

Além disso, observa-se que o desempenho do classificador foi inferior nos cenários com o canal Rician. Esse fato está diretamente relacionado à dificuldade do modelo em distinguir entre as modulações OFDM-16QAM e OFDM-64QAM. Diferentemente das modulações BPSK e QPSK, que apresentam constelações mais espaçadas e, portanto, características mais distintas e fáceis de aprender, as constelações de 16QAM e 64QAM possuem pontos mais densos e próximos entre si, o que torna a diferenciação mais sutil e desafiadora, especialmente em ambientes com desvanecimento seletivo em frequência, como é o caso dos canais Rician. Assim, em canais Rician, caso seja utilizada a técnica OFDM, é recomendável o uso das modulações PSK ou que o modelo seja treinado com

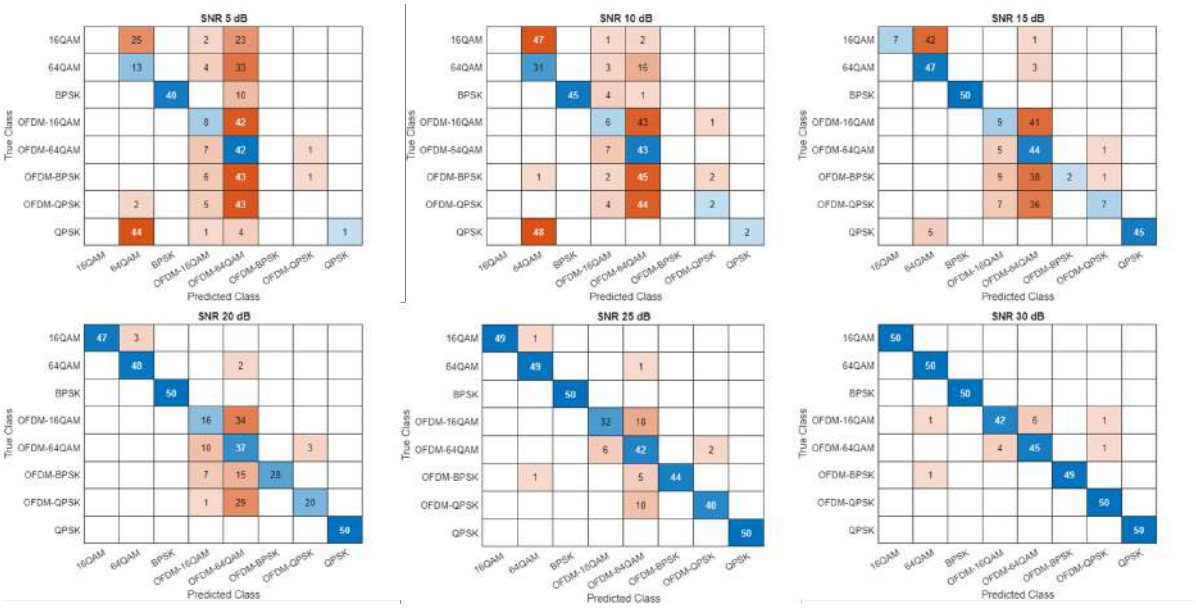

Figura 77 – Matriz de confusão do classificador no conjunto de testes de *frames* reais.

somente uma das modulações QAM para facilitar a diferenciação.

As dificuldades de classificação podem ser observadas tanto nos diagramas de constelação quanto nas componentes I e Q, mostradas nas Figuras 78, 79, 80 e 81, para um *frame* de cada tipo de modulação. Nota-se que o canal Rician provoca espalhamento e rotação dos símbolos, alterando significativamente os diagramas de constelação em relação aos padrões ideais de cada modulação. Nas modulações convencionais, ainda é possível perceber um padrão mais bem definido, com concentrações em determinados pontos da constelação, o que facilita o reconhecimento pelo classificador. Em contraste, nas modulações OFDM é possível identificar facilmente a presença do CP ao observar o início e o final das componentes I e Q do *frame*; porém, o espalhamento mais acentuado dos símbolos torna a diferenciação entre as classes consideravelmente mais difícil.

Por último, vale ressaltar que o canal real utilizado na transmissão pela SDR se assemelha muito ao de um canal AWGN com uma rotação. Desta forma, ao aplicar a correção de rotação, o classificador conseguiu classificar os *frames* reais tão bem quanto os simulados. Portanto, essa estratégia possibilitou que o classificador do projeto fosse utilizado de uma forma semelhante que em uma aplicação real de transmissão de dados entre dispositivos IoT.

3.5 Resultados da Verificação do Uso de Sincronização

Na realização dos testes com o conjunto de *frames* contendo rotação de fase, foram obtidas as acurácias apresentadas na Tabela 8 para cada uma das CNNs avaliadas. Veja que o único modelo que não conseguiu um desempenho parecido com os resultados utilizando correção de rotação foi a $CNN_{OFDM,AWGN}$.

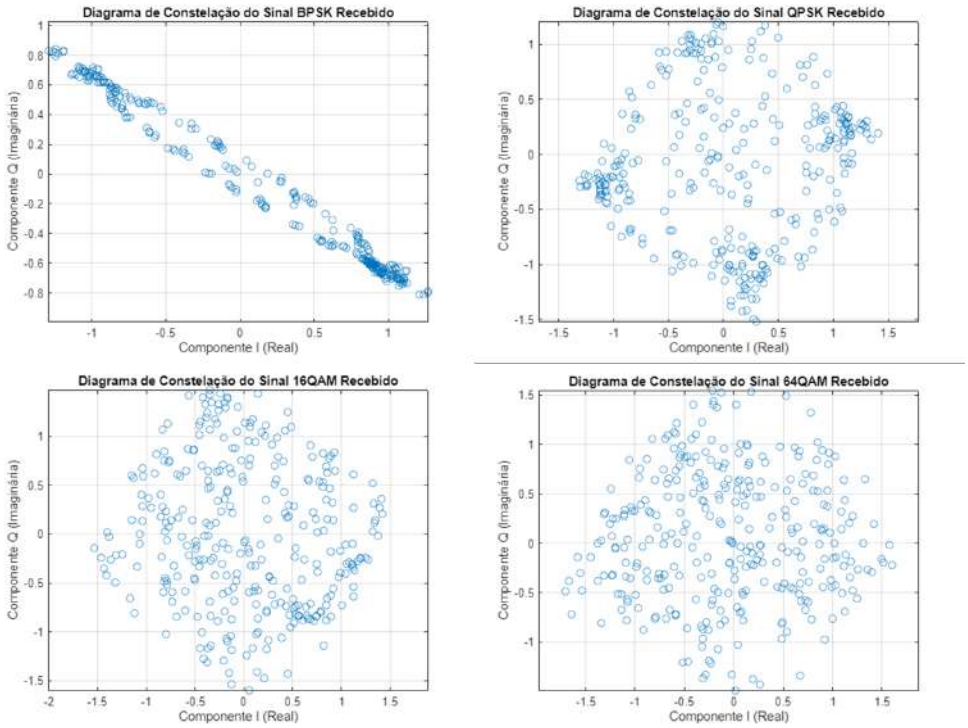

Figura 78 – Diagrama de constelação de um *frame* das modulações convencionais em um canal Rician com SNR de 30 dB.

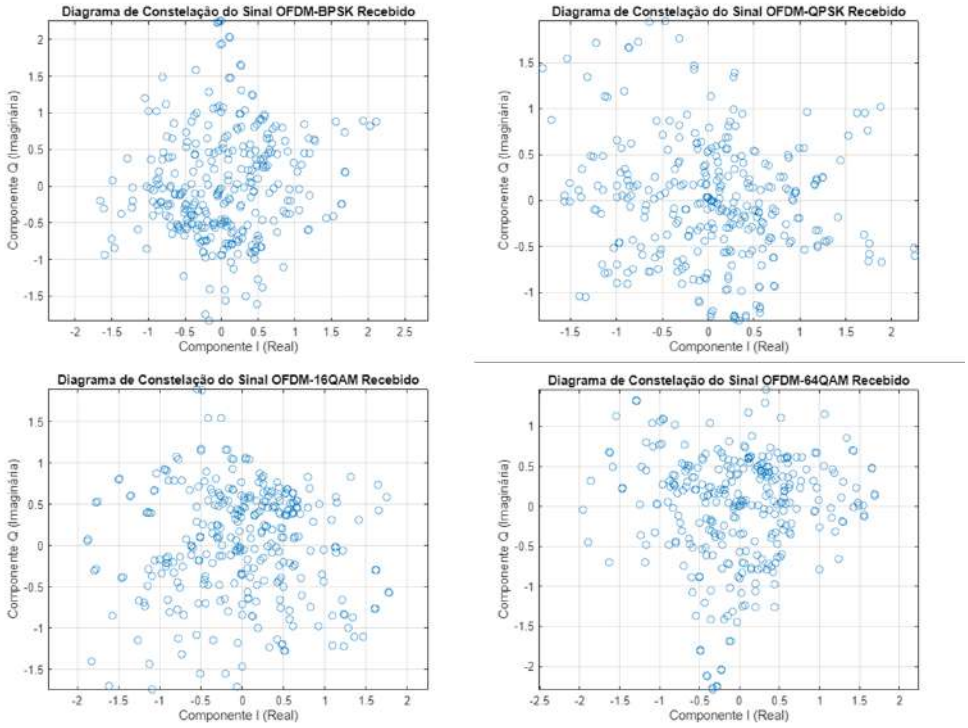

Figura 79 – Diagrama de constelação de um *frame* das modulações OFDM em um canal Rician com SNR de 30 dB.

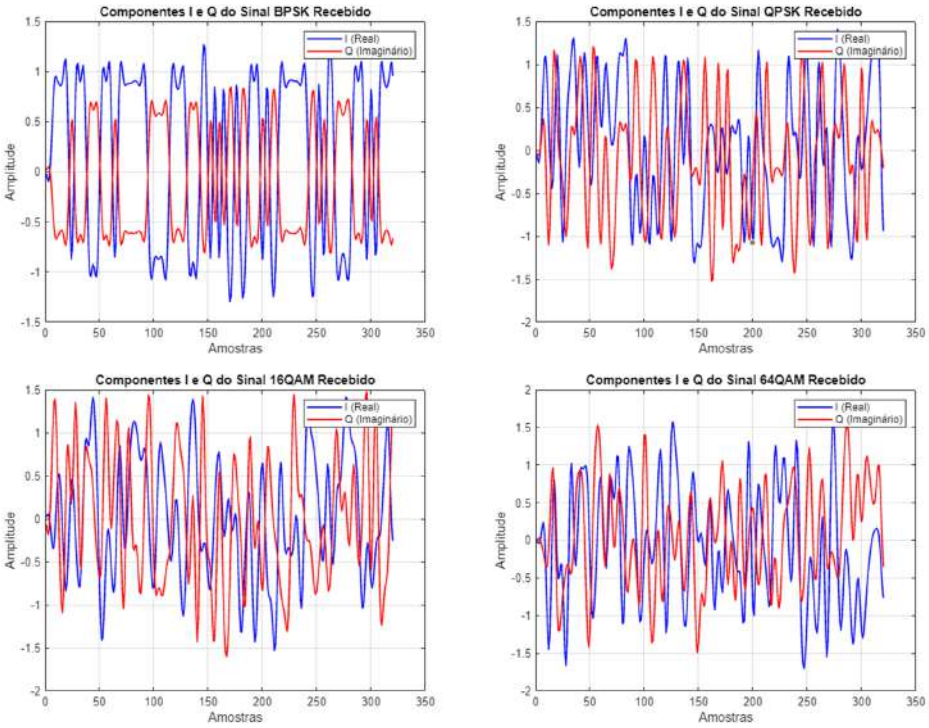

Figura 80 – Componentes I e Q de um *frame* das modulações convencionais em um canal Rician com SNR de 30 dB.

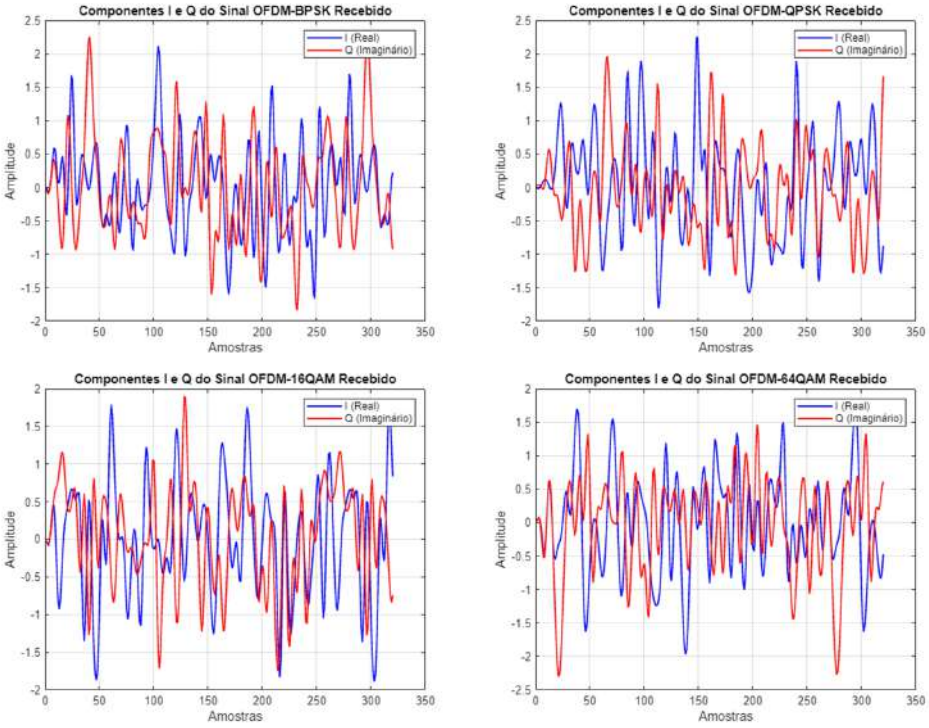

Figura 81 – Componentes I e Q de um *frame* das modulações OFDM em um canal Rician com SNR de 30 dB.

Tabela 7 – Resultados do desempenho do classificador para SNR de 30 dB em diferentes conjuntos de teste

***Frames* de teste**	**Acurácia**
Simulados em canal AWGN	98,83%
Simulados em canais Rician	77,54%
Simulados em canais Rician e AWGN	87,50%
Reais sem correção de rotação	24,25%
Reais com correção de rotação	96,50%

Tabela 8 – Acurácia das CNNs treinadas com as base de dados com *frames* com rotação

CNN	**Acurácia nos *frames* simulados**	**Acurácia nos *frames* reais**
CNN_{CxO}	99,23%	99,75%
$CNN_{Conv,AWGN}$	99,99%	100,00%
$CNN_{OFDM,AWGN}$	70,84%	70,00%

Tabela 9 – Acurácia das CNNs treinadas com as base de dados com *frames* com atraso

CNN	**Acurácia nos *frames* simulados**	**Acurácia nos *frames* reais**
CNN_{CxO}	98,58%	97,50%
$CNN_{Conv,AWGN}$	99,98%	100,00%
$CNN_{OFDM,AWGN}$	98,57%	96,00%

Para compreender o motivo pelo qual o modelo $CNN_{OFDM,AWGN}$ não apresentou um ótimo desempenho, é possível analisar as matrizes de confusão da classificação exibidas na Figura 82. Observando essas matrizes, nota-se que o modelo teve dificuldades principalmente na distinção entre as classes OFDM-16QAM e OFDM-64QAM. Isso se deve ao maior número de símbolos presentes nas constelações dessas modulações, o que torna os sinais mais sensíveis a variações de fase. Com a rotação de fase aplicada nos *frames*, os pontos da constelação se deslocam, dificultando a discriminação entre as modulações com constelações mais densas.

Na realização do segundo teste, isto é, com o conjunto de *frames* contendo atraso na amostragem, foram obtidas as acurácias apresentadas na Tabela 9 para cada uma das CNNs avaliadas. Observa-se que, mesmo com a introdução do atraso, os resultados permaneceram semelhantes aos obtidos com sincronização perfeita, indicando que as redes são relativamente robustas a pequenos desvios na amostragem.

A Figura 83 apresenta as matrizes de confusão obtidas com a aplicação do teste de atraso de amostragem. A partir dessas matrizes, é possível constatar que, mesmo com pequenos atrasos de sincronização, as CNNs mantiveram um bom desempenho na tarefa de classificação.

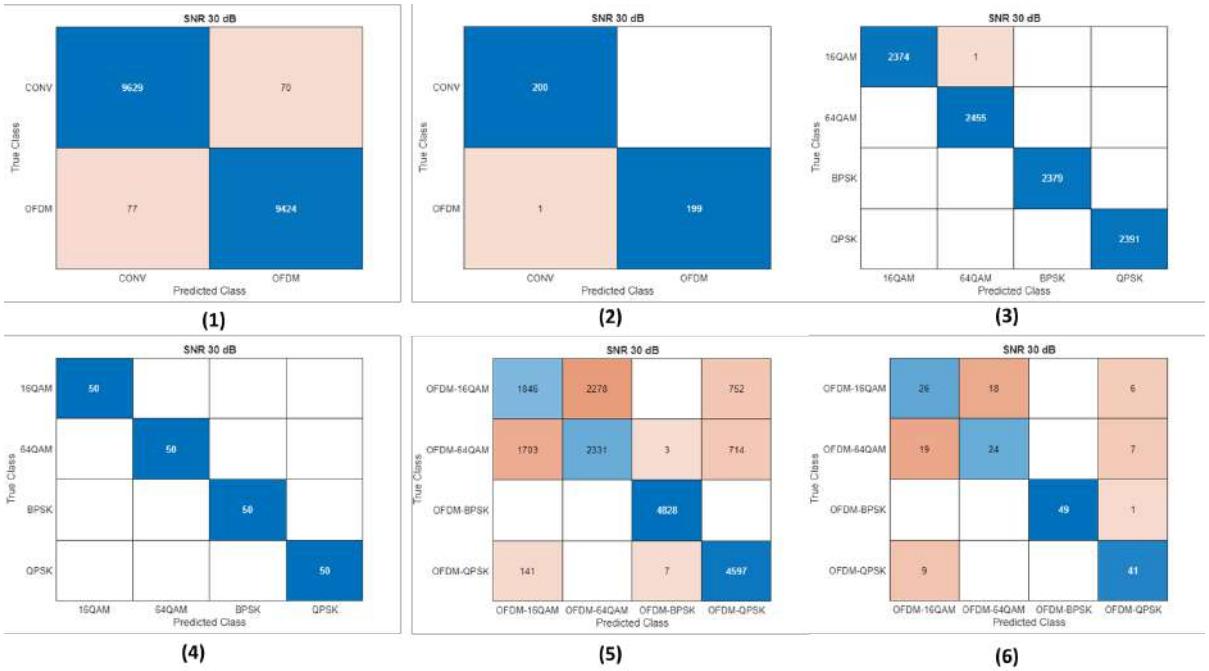

Figura 82 – Matrizes de confusão das CNNs treinadas com rotação. (1) CNN_{CxO} classificando *frames* simulados, (2) CNN_{CxO} classificando *frames* reais, (3) $CNN_{Conv,AWGN}$ classificando *frames* simulados, (4) $CNN_{Conv,AWGN}$ classificando *frames* reais, (5) $CNN_{OFDM,AWGN}$ classificando *frames* simulados e (6) $CNN_{OFDM,AWGN}$ classificando *frames* reais.

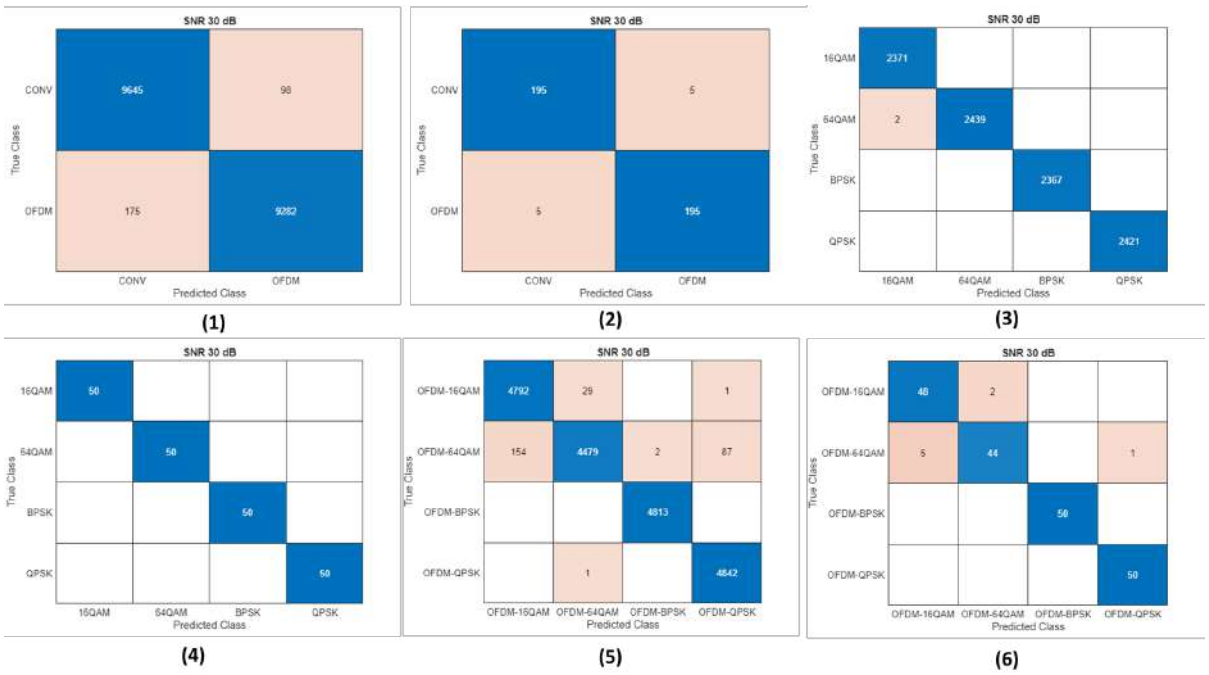

Figura 83 – Matrizes de confusão das CNNs treinadas com atraso. (1) CNN_{CxO} classificando *frames* simulados, (2) CNN_{CxO} classificando *frames* reais, (3) $CNN_{Conv,AWGN}$ classificando *frames* simulados, (4) $CNN_{Conv,AWGN}$ classificando *frames* reais, (5) $CNN_{OFDM,AWGN}$ classificando *frames* simulados e (6) $CNN_{OFDM,AWGN}$ classificando *frames* reais.

Conclusão

Com a execução desse projeto pode-se principalmente realizar o estudo da fundamentação teórica e da aplicação do aprendizado de máquina em sistemas de comunicação digitais. Assim, o desenvolvimento desse projeto possibilitou a aplicação do algoritmo de aprendizado de máquina de CNN para a criação de um classificador de detecção do tipo da modulação de *frames* de sinais digitais para aplicações de IoT.

O modelo de classificação foi testado em canais AWGN e Rician simulados e em um canal de transmissão real. Apresentando um resultado satisfatório na classificação do conjunto de modulações (BPSK, QPSK, 16QAM, 64QAM, OFDM-BPSK, OFDM-QPSK, OFDM-16QAM e OFDM-64QAM) principalmente nos canais AWGN e real. A acurácia no conjunto de testes de *frames* reais captados pelo SDR atingiu 96,50% em uma SNR de 30 dB e 98,83% nos *frames* simulados em um canal AWGN. Já no canal Rician apresentou maior dificuldade de classificação, chegando a 77,54% de acurácia nos *frames* com SNR de 30 dB.

O classificador sofreu dificuldades para classificar entre as modulações OFDM-16QAM e OFDM-64QAM em um canal Rician. Tal dificuldade é demonstrada pelas constelações das modulações 16QAM e 64QAM que possuem pontos mais densos e próximos entre si. Apesar dessa dificuldades, o classificador conseguiu atingir uma boa acurácia para as demais modulações nessas condições de canal.

Ademais, percebe-se que o desempenho do classificador é influenciado diretamente pela qualidade do sinal, na qual em SNR baixas a acurácia dos classificador cai drasticamente, chegando em valores próximos de 40%.

A principal limitação dos classificadores é seu funcionamento restrito a *frames* que seguem os mesmos parâmetros do projeto. Ou seja, os *frames* precisam ter um tamanho fixo de 80 símbolos digitais, utilizar uma das modulações presentes no conjunto de treinamento do classificador e adotar os mesmos parâmetros para a técnica de OFDM e do pulso de cosseno levantado. As alterações nos parâmetros não garante mais que o classificador consiga manter o mesmo desempenho que o exibido no projeto.

Além disso, os testes realizados evidenciaram que a sincronização de fase desempenha um papel fundamental na melhoria da classificação das CNNs nos *frames* reais, especialmente para as redes responsáveis por identificar modulações OFDM. Por outro lado, no teste com variações no ponto de início da amostragem, observou-se que pequenos atrasos não impactaram significativamente o desempenho das CNNs, indicando certa robustez a esse tipo de desvio de sincronização.

A criação e o aperfeiçoamento desses modelos em trabalhos futuros pode possibilitar a sua utilização como suporte a aplicações de IoT que utilizam transmissão de símbolos OFDM como o padrão IEEE 802.11g e necessitam identificar o tipo de modulação que o sinal recebido possui. Além disso, o modelo pode ser aprimorado para melhorar o seu desempenho e realizar um auto ajuste para ser utilizado em *frames* com parâmetros diferentes dos utilizados no projeto. Conseguindo assim gravar esse classificador aparelhos como FPGAs (*Field Programmable Gate Array*) e realizar a classificação de sinais transmitidos entre dispositivos de IoT.

Referências

1 LATHI, B. P. *Modern digital and analog communication systems*. [S.l.]: Oxford University Press, Inc., 1990. Citado 9 vezes nas páginas 4, 5, 7, 8, 14, 15, 16, 17 e 19.

2 MOLISCH, A. F. *Wireless communications*. [S.l.]: John Wiley & Sons, 2012. v. 34. Citado na página 5.

3 COUCH, L. W.; KULKARNI, M.; ACHARYA, U. S. *Digital and analog communication systems*. [S.l.]: Citeseer, 2013. v. 8. Citado 3 vezes nas páginas 5, 6 e 9.

4 CASELLA, I. R.; ANPALAGAN, A. *Power line communication systems for smart grids*. [S.l.]: Institution of Engineering and Technology, 2018. Citado 20 vezes nas páginas 6, 7, 8, 9, 11, 12, 13, 15, 16, 17, 18, 19, 20, 21, 22, 24, 25, 26, 27 e 28.

5 TSE, D.; VISWANATH, P. *Fundamentals of wireless communication*. [S.l.]: Cambridge university press, 2005. Citado na página 9.

6 VISWANATHAN, M. *Wireless Communication Systems in Matlab*. [S.l.: s.n.], 2020. Citado na página 10.

7 ABDI, A. et al. On the estimation of the k parameter for the rice fading distribution. *IEEE Communications letters*, IEEE, v. 5, n. 3, p. 92–94, 2002. Citado 2 vezes nas páginas 10 e 11.

8 STREMLER, F. G. *Introduction to communication systems*. [S.l.: s.n.], 1990. Citado na página 20.

9 MOSSINGHOFF, M. J. Wieferich pairs and barker sequences. *Designs, Codes and Cryptography*, Springer, v. 53, n. 3, p. 149–163, 2009. Citado na página 22.

10 EYADEH, A. A. Frame synchronization symbols for an ofdm system. *International Journal of Communications*, v. 2, n. 1, 2008. Citado na página 22.

11 PINTO, E. L.; ALBUQUERQUE, C. P. de. A técnica de transmissão ofdm. *Revista Científica*, v. 1516, p. 2338, 2002. Citado na página 23.

12 CHANG, R. W. Synthesis of band-limited orthogonal signals for multichannel data transmission. *Bell system technical journal*, Wiley Online Library, v. 45, n. 10, p. 1775–1796, 1966. Citado na página 24.

13 LOPES, E. M.; CARDOSO, F. A.; ARANTES, D. S. Análise de desempenho de equalizadores pré-fft em sistemas ofdm. Citado na página 27.

14 SANTOS, T. T. et al. Efeito da equalização na recepção de sinais ofdm. *Engevista*, v. 20, n. 5, p. 701–711, 2018. Citado na página 28.

15 WYGLINSKI, A. M. et al. *Software-defined radio for engineers*. [S.l.]: Artech House, 2018. Citado na página 28.

16 WYGLINSKI, A. M.; NEKOVEE, M.; HOU, T. *Cognitive radio communications and networks: principles and practice*. [S.l.]: Academic Press, 2009. Citado na página 29.

17 MOHRI, M.; ROSTAMIZADEH, A.; TALWALKAR, A. *Foundations of machine learning*. [S.l.]: MIT press, 2018. Citado na página 30.

18 AYODELE, T. O. Types of machine learning algorithms. *New advances in machine learning*, IntechOpen Rijeka, v. 3, p. 19–48, 2010. Citado na página 30.

19 SAH, S. Machine learning: a review of learning types. Preprints, 2020. Citado 3 vezes nas páginas 31, 32 e 34.

20 LINARDATOS, P.; PAPASTEFANOPOULOS, V.; KOTSIANTIS, S. Explainable ai: A review of machine learning interpretability methods. *Entropy*, MDPI, v. 23, n. 1, p. 18, 2020. Citado 2 vezes nas páginas 32 e 33.

21 BISHOP, C. M.; NASRABADI, N. M. *Pattern recognition and machine learning*. [S.l.]: Springer, 2006. v. 4. Citado 5 vezes nas páginas 34, 35, 36, 39 e 40.

22 ROSENBLATT, F. The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological review*, American Psychological Association, v. 65, n. 6, p. 386, 1958. Citado na página 35.

23 NIELSEN, M. A. *Neural networks and deep learning*. [S.l.]: Determination press San Francisco, CA, USA, 2015. v. 25. Citado 3 vezes nas páginas 35, 36 e 38.

24 GOODFELLOW, I. et al. *Deep learning*. [S.l.]: MIT press Cambridge, 2016. v. 1. Citado 6 vezes nas páginas 36, 37, 38, 39, 40 e 41.

25 LECUN, Y. et al. Lenet-5, convolutional neural networks. *URL: http://yann. lecun. com/exdb/lenet*, v. 20, n. 5, p. 14, 2015. Citado na página 39.

26 DALIANIS, H. Evaluation metrics and evaluation. In: _____. *Clinical Text Mining: Secondary Use of Electronic Patient Records*. Cham: Springer International Publishing, 2018. p. 45–53. ISBN 978-3-319-78503-5. Disponível em: <https://doi.org/10.1007/978-3-319-78503-5_6>. Citado 2 vezes nas páginas 41 e 42.

27 NUAND. *bladeRF*. 2025. <https://www.nuand.com/bladerf-1/> [Acessado: 29/05/2025]. Citado na página 46.

28 NUAND. *bladeRF 2.0 micro xA9*. 2025. <https://www.nuand.com/product/bladerf-xa9/> [Acessado: 29/05/2025]. Citado na página 46.

29 NEE, R. v.; PRASAD, R. *OFDM for wireless multimedia communications*. [S.l.]: Artech House, Inc., 2000. Citado na página 48.

30 HUYNH-THE, T. et al. Learning deep convolutional network for automatic modulation classification in ofdm systems under channel impairments. *IEEE Internet of Thing Journa*, 2020. Citado na página 52.

31 HUYNH-THE, T. et al. An efficient deep network for modulation classification in impaired mimo-ofdm systems. In: IEEE. *2023 IEEE Statistical Signal Processing Workshop (SSP)*. [S.l.], 2023. p. 130–134. Citado 2 vezes nas páginas 52 e 55.