João Pedro Melquiades Gomes

IDENTIFICAÇÃO DE DISPOSITIVOS EM UMA REDE 5G-IOT COM BASE NAS CARACTERÍSTICAS FÍSICAS DAS ANTENAS TRANSMISSORAS

Campina Grande, Paraíba

Novembro de 2023

João Pedro Melquiades Gomes

IDENTIFICAÇÃO DE DISPOSITIVOS EM UMA REDE 5G-IOT COM BASE NAS CARACTERÍSTICAS FÍSICAS DAS ANTENAS TRANSMISSORAS

Trabalho de Conclusão de Curso de Bacharelado submetido à Coordenadoria de Graduação em Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciências no Domínio da Engenharia Elétrica

Universidade Federal de Campina Grande – UFCG

Centro de Engenharia Elétrica e Informática – CEEI

Departamento de Engenharia Elétrica – DEE

Orientador: Edmar Candeia Gurjão, Dr.

Campina Grande, Paraíba

Novembro de 2023

João Pedro Melquiades Gomes

IDENTIFICAÇÃO DE DISPOSITIVOS EM UMA REDE 5G-IOT COM BASE NAS CARACTERÍSTICAS FÍSICAS DAS ANTENAS TRANSMISSORAS

Trabalho de Conclusão de Curso de Bacharelado submetido à Coordenadoria de Graduação em Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para a obtenção do grau de Bacharel em Ciências no Domínio da Engenharia Elétrica

Trabalho aprovado. Campina Grande, Paraíba, 13 de novembro de 2023:

Edmar Candeia Gurjão, Dr.
Orientador

Leocarlos Bezerra da Silva Lima, Dr.
Avaliador

Campina Grande, Paraíba
Novembro de 2023

Dedico este trabalho a minha mãe, eterna guerreira que me fez quem eu sou hoje.

Agradecimentos

Agradeço a meus pais, João Carlos e Iranês, por terem me incentivado e garantido minha educação desde a alfabetização, sempre lutando para que eu estivesse nas melhores escolas através de bolsas. Ao meu padrinho, Edgley, que agiu como intermédio durante o ensino médio para que fosse possível que eu estudasse em uma das melhores escolas de Campina Grande através de uma bolsa integral.

Agradeço a minha noiva, Karine, por estar sempre comigo, pela compreensão, pelas brigas quando eu ultrapassava os limites e dedicava muito tempo ao trabalho, pelas doses de motivação quando achei que tudo ia dar errado. Se termino esse curso, devo muito aos incentivos dela.

Agradeço aos meus amigos, a todos que tive a oportunidade de conhecer durante a jornada da graduação. Aos amigos companheiros de RPG, Kilian, Vinícius, Filipe, Renato e Gabriel, que propiciaram várias doses de alegria nos vários mundos que criamos juntos nas seções de RPG. Aos amigos que me auxiliaram durante o curso, sendo estes os mesmos citados anteriormente, com adição de Pedro Henrique, amigo que fiz no primeiro período e me ajuda até hoje, Lara Sobral, minha "madrinha" de curso, e Ronildo, grande amigo desde o ensino médio.

Agradeço aos meus irmãos escoteiros, movimento que me auxiliou a evoluir como ser humano durante a graduação, e que espero nunca deixar de fazer parte. A todos os chefes do grupo escoteiro Dom Luís Gonzaga Fernandes, na pessoa do chefe Fábio Góes, diretor presidente atual. Ao distrito da Borborema, na pessoa do chefe Cláudio Carvalho, e à Região dos Escoteiros da Paraíba, na pessoa do atual presidente chefe Peron. Todos essas pessoas contribuiram de alguma forma para meu crescimento como escoteiro e como cidadão, e a maturidade que tenho hoje ao final do curso devo também a eles. Além disso, agradeço também a todos os jovens os quais fui responsável durante minha trajetória no movimento, com os quais aprendi muita coisa enquanto também os ensinava.

Agradeço a todos os engenheiros que me apoiaram durante minha jornada como aluno $PD\&I$ no Virtus, por terem formado todas as competências que tenho hoje e que irei exercer no mercado de trabalho. Aos engenheiros Pedro, Matheus, Kelvin, Kaline, Hugo e Gabriel. O futuro engenheiro que serei é também graças a essas pessoas.

Agradeço aos vários professores que pude conhecer durante o curso. Aqui, cito meu orientador de TCC, Edmar Candeia Gurjão, que sempre foi um exemplo de professor por sua forma de ministrar as aulas e sua preocupação com o aprendizado dos alunos, o professor Marcos Morais, o qual admiro pela carga de conhecimento acumulada em uma só pessoa, o que me motiva a adquirir mais conhecimento, e ao meu orientador de estágio

Gutemberg Júnior, responsável por me fazer querer seguir na área de microeletrônica. Que todos continuem sendo essa fonte de motivação para mais jovens como eu.

Por fim, agradeço também a minha pequena cadela Lilica, que entrou em minha vida durante a graduação e também auxiliou a manter minha cabeça no lugar com os momentos de descontração que ela propiciou, seja com seus beijos inesperados ou com os surtos de energia que sempre me faziam rir.

"A filosofia não visa a assegurar qualquer coisa externa ao homem. Isso seria admitir algo que está além de seu próprio objeto. Pois assim como o material do carpinteiro é a madeira, e o do estatuário é o bronze, a matéria prima da arte de viver é a própria vida de cada pessoa."

(Epicteto)

Resumo

A segurança da camada física é um cenário ainda pouco explorado quando o assunto é redes 5G. Dispositivos IoT são passíveis de serem falsificados com os invasores adquirindo informações sensíveis enquanto se passam por dispositivos autenticados. Neste trabalho é descrita a implementação de sistemas de identificação de dispositivos com base nas características de fabricação dos arranjos de antenas que transmitem os sinais na rede, utilizando de diferentes métodos de aprendizagem profunda para solucionar tal problema. Com isso, observou-se resultados de especificidade 100% após o treinamento dos modelos, sendo assim viável a detecção de todos os dispositivos invasores. Isso mostra que é possível utilizar-se de algoritmos de inteligência artificial para proteger as redes de alta taxa de transmissão que surgem com o 5G, possibilitando uma proteção que independe dos dados transmitidos e que se baseia em características difíceis de serem duplicadas.

Palavras-chave: 5G. Aprendizado Profundo. Cibersegurança. Arranjo de fase de antenas.

Abstract

Physical layer security is a relatively unexplored area in the context of 5G networks. IoT devices are susceptible to counterfeiting with the intention of infiltrating a network, acquiring sensitive information while posing as authenticated devices. This work involves the implementation of device identification systems based on the manufacturing characteristics of the antenna arrays that transmit signals in the network, utilizing various deep learning methods to address this issue. As a result, the models achieved a specificity of 100% after training, effectively detecting all invading devices. This demonstrates the feasibility of leveraging artificial intelligence algorithms to safeguard high-speed transmission networks emerging with 5G, providing protection that is independent of transmitted data and relies on difficult-to-duplicate characteristics.

Keywords: 5G. Deep Learning. Cybersecurity. Phased array antennas.

Lista de ilustrações

Figura 1 – Elementos que compõe o padrão de feixe de um arranjo 20
Figura 2 – Arquitetura de um *Perceptron* 24
Figura 3 – Varredura de uma imagem durante uma convolução 27
Figura 4 – Estruturação de um *autoencoder* 29
Figura 5 – Exemplo de classificação binária 30
Figura 6 – Exemplo de matriz de confusão 31
Figura 7 – Posição dos elementos no arranjo de antenas - Arranjo $6x6$ 33
Figura 8 – Arquitetura da rede neural convolucional 36
Figura 9 – Arquitetura do DAE 37
Figura 10 – Arquitetura do DAE convolucional 38
Figura 11 – Padrões de feixe - *StaticPosAllUEs_Dataset* 40
Figura 12 – Padrões de feixe em três dimensões - *StaticPosAllUEs_Dataset* 40
Figura 13 – Padrões de feixe - *StaticPosSingleUE_Dataset* 41
Figura 14 – Padrões de feixe em três dimensões - *StaticPosSingleUE_Dataset* . . . 41
Figura 15 – Padrões de feixe - *DynamicPos_Dataset* 41
Figura 16 – Padrões de feixe em três dimensões - *DynamicPos_Dataset* 42
Figura 17 – Reconstrução de um padrão de feixe autenticado - *StaticPosAllUEs_Dataset* 43
Figura 18 – Reconstrução de um padrão de feixe invasor - *StaticPosAllUEs_Dataset* 43
Figura 19 – Comparação entre os limiares de reconhecimento - *StaticPosAllUEs_Dataset* 44
Figura 20 – Métricas para os três modelos - *StaticPosAllUEs_Dataset* 45
Figura 21 – Matriz de confusão para os três modelos - *StaticPosAllUEs_Dataset* . 45
Figura 22 – Reconstrução de um padrão de feixe autenticado - *StaticPosSingleUE_Dataset* 46
Figura 23 – Reconstrução de um padrão de feixe invasor - *StaticPosSingleUE_Dataset* 46
Figura 24 – Comparação entre os limiares de reconhecimento - *StaticPosSingleUE_Dataset* 47
Figura 25 – Métricas para os três modelos - *StaticPosSingleUE_Dataset* 48
Figura 26 – Matriz de confusão para os três modelos - *StaticPosSingleUE_Dataset* 48
Figura 27 – Reconstrução de um padrão de feixe autenticado - *DynamicPos_Dataset* 49
Figura 28 – Reconstrução de um padrão de feixe invasor - *DynamicPos_Dataset* . . 49
Figura 29 – Métricas para os três modelos - *DynamicPos_Dataset* 50
Figura 30 – Comparação entre os limiares de reconhecimento - *Dynamic_Dataset* . 51
Figura 31 – Matriz de confusão para os três modelos - *StaticPosSingleUE_Dataset* 51
Figura 32 – Reconstrução de um padrão de feixe autenticado - *DynamicPos_Dataset* - Retreinamento 53
Figura 33 – Reconstrução de um padrão de feixe invasor - *DynamicPos_Dataset* -Retreinamento 53
Figura 34 – Métricas para os três modelos - *DynamicPos_Dataset* - Retreinamento 54

Figura 35 – Comparação entre os limiares de reconhecimento - *Dynamic_Dataset* - Retreinamento 55
Figura 36 – Matriz de confusão para os três modelos - *StaticPosSingleUE_Dataset* - Retreinamento 55

Lista de tabelas

Tabela 1 – Diferenças entre os procedimentos de gerenciamento de feixe 22
Tabela 2 – Parâmetros para geração dos bancos de dados 35
Tabela 3 – Parâmetros e hiperparâmetros utilizados na CNN 36
Tabela 4 – Parâmetros e hiperparâmetros utilizados no DAE 37
Tabela 5 – Parâmetros e hiperparâmetros utilizados no ConvDAE 38

Lista de abreviaturas e siglas

mmWave	*milimetrical Waves*
IoT	*Internet of Things*
IoE	*Internet of Everything*
DAE	*Deep autoencoder*
UE	*User equipment*
gNodeB	*Next Generation Node B*
CNN	*Convolutional Neural Network*
DAE	*Deep Autoencoder*
ConvDAE	*Convolutional Deep Autoencoder*
MSE	*Mean Square Error*
ReLU	*Rectified Linear Unit*

Sumário

1 INTRODUÇÃO 15
1.1 Objetivos 15
1.1.1 Objetivo Geral 15
1.1.2 Objetivos Específicos 16
1.2 Metodologia 16
1.3 Organização do Trabalho 17

2 REVISÃO DA LITERATURA 18

3 FUNDAMENTAÇÃO TEÓRICA 19
3.1 Arranjo de fase de antenas 19
3.2 5G - Gerenciamento de feixes 21
3.2.1 Varredura de feixe 21
3.2.2 Medição de feixe 22
3.3 Aprendizagem Profunda 23
3.3.1 Aprendizado com retropropagação 24
3.3.2 Funções de ativação 24
3.3.2.1 *ReLU - Rectified Linear Unit* 24
3.3.2.2 *Sigmoid* 25
3.3.2.3 *Tanh* 25
3.3.3 CNN - Rede neural convolucional 25
3.3.4 DAE - Deep Autoenconder 28
3.3.5 Métricas de desempenho 29

4 IMPLEMENTAÇÃO 32
4.1 Modelagem dos arranjos de antenas 32
4.2 Geração do banco de dados 33
4.3 Implementação dos modelos 35

5 RESULTADOS 39
5.1 Padrões de feixe dos dispositivos 39
5.2 Comparação de métricas 42
5.2.1 *StaticPosAllUEs_Dataset* 42
5.2.2 *StaticPosSingleUs_Dataset* 45
5.2.3 *DynamicPos_Dataset* 48
5.2.3.1 Melhorias no treinamento 52

6 CONCLUSÃO 56

REFERÊNCIAS 58

1 Introdução

O espectro das ondas milimétricas (*mmWaves*) abrange as frequências acima de 24GHz, desempenhando um papel essencial na tecnologia 5G. Ao possibilitar altas taxas de transferência em redes celulares, as *mmWaves* têm se destacado como parte integral dessa evolução. Consequentemente, à medida que as redes 5G avançam, novas aplicações no âmbito da Internet das Coisas (do inglês: *Internet of Things - IoT*) emergem, inaugurando um conceito ainda mais abrangente: a Internet de Tudo (do inglês - *Internet of Everything - IoE*). A tecnologia 5G facilitará a conexão simultânea de milhões de dispositivos, ampliando significativamente a escala das redes.

No entanto, esse progresso traz consigo uma preocupação fundamental: a segurança das redes. Com o surgimento recente de dispositivos que operam nas *mmWaves*, esse campo se torna uma arena potencialmente vulnerável a ataques. Um exemplo notório é o ataque de falsificação (*Spoofing Attack*), em que dispositivos maliciosos tentam se infiltrar na rede, fingindo ser dispositivos autênticos. Nas redes IoT, por exemplo, esse tipo de ataque é amplamente observado devido a práticas inseguras, como a manutenção de senhas padrão em dispositivos.

Sendo assim, diante de todos os fatores apresentados, propõe-se neste trabalho a comparação de diferentes métodos de aprendizagem profunda para a autenticação de dispositivos em uma rede IoT com base nas características físicas de seus arranjos de antenas. Com isso, pretende-se verificar a eficácia da detecção de invasores em dois cenários: Equipamentos de usuário (do inglês: *User equipment - UE*) estáticos e UEs que podem modificar sua posição no ambiente. As principais métricas a serem analisadas serão a acurácia, a precisão, a revocação e a especificidade, sendo as quatro definidas posteriormente neste trabalho.

1.1 Objetivos

1.1.1 Objetivo Geral

- Desenvolver um ambiente de simulação de rede 5G no MATLAB, que inclua a construção detalhada do modelo de antenas de cada dispositivo. Posteriormente, realizar o treinamento e avaliação de diferentes modelos de inteligência artificial, orientando-se pelas características físicas das antenas e com o propósito de identificar a autenticidade dos dispositivos pertencentes à rede.

1.1.2 Objetivos Específicos

- Investigar a fundamentação teórica que viabiliza a abordagem a ser utilizada;
- Construir o ambiente virtual no *MATLAB* para as simulações;
- Construir e treinar diferentes modelos de inteligência artificial para autenticar os dispositivos;
- Comparar todos os modelos treinados durante o processo.

1.2 Metodologia

Considerando uma rede com elementos sem fio, cada um com sua antena, a princípio foram implementados *scripts* em MATLAB para a implementação de diferentes arranjos de antenas, variando alguns parâmetros a fim de gerar diferentes padrões de feixe. Em posse dos modelos, os padrões foram analisados e uma verificação inicial foi feita sobre quais parâmetros das antenas seriam utilizados durante o restante do desenvolvimento, levando em conta a influência no padrão e o *trade-off* entre a demanda computacional necessária para simular tais arranjos e o tempo disponível para a realização do trabalho.

Depois, foi implementado um microambiente dentro do MATLAB contendo: geração de formas de onda, arranjo de antenas transmissor, canal de transmissão, arranjo de antenas receptor e mecanismo de medição de potência. Em seguida, esse sistema foi utilizado de forma iterativa em um *script* que percorria todas as antenas geradas, realizando diversas medições para cada uma, construindo assim os bancos de dados que seria utilizado no restante do trabalho.

Sendo assim, foram gerados três bancos de dados, todos contendo padrões de feixe de quatro dispositivos que possuem um arranjo de antenas $6x6$. O primeiro banco de dados, que será referido como *StaticPosAllUEs_Dataset*, possui medições feitas para uma única posição de UE, modificando apenas a UE para cada conjunto de medidas. Esse banco de dados tem como objetivo comprovar se a detecção se baseia unicamente nos padrões de feixe e não nas variações causadas pela distância ou ângulo de transmissão das antenas. O segundo banco de dados, referido aqui como *StaticPosSingleUE_Dataset* possui medições feitas para os quatro dispositivos em locais diferentes, porém sem a variação das posições durante o processo. Por fim, o último banco de dados, *DynamicPos_Dataset* possui medições feitas com os quatro dispositivos variando suas posições durante o processo de medição, permitindo assim a construção de um modelo um pouco mais robusto. Cada um dos bancos de dados foi dividido em duas partes: uma para o treinamento e uma para o teste dos modelos.

Após a criação dos bancos de dados, três modelos de inteligência artificial foram criados utilizando *Python*, mais especificamente a estrutura de desenvolvimento *Tensorflow*. Os modelos treinados foram:

- DAE - *Deep Auto Encoder*;
- DAE convolucional;
- CNN (Rede neural convolucional, do inglês *Convolutional Neural Network*).

Os três modelos foram então utilizados para realizar predições nas partes de teste dos bancos de dados, comparando assim as métricas de cada um deles.

1.3 Organização do Trabalho

O trabalho está estruturado em 6 capítulos, incluindo este introdutório, conforme a seguir.

O **Capítulo 2** apresenta uma Revisão da Literatura relacionada ao objetivo deste trabalho.

No **Capítulo 3** está descrita toda a fundamentação teórica de todos os tópicos abordados durante o desenvolvimento do trabalho.

O **Capítulo 4** descreve o processo de implementação dos *scripts* já mencionados.

No **Capítulo 5** são apresentados os resultados obtidos, assim como as comparações entre os diferentes cenários.

Por fim, no **Capítulo 6**, têm-se as considerações finais e os pontos de melhoria do trabalho.

2 Revisão da Literatura

Neste capítulo, será apresentada uma revisão da literatura no que se refere ao uso de inteligência artificial para resolução de problemas de cibersegurança em redes 5G, além de como a temática deste trabalho foi escolhida com base em uma revisão sistemática feita com base nesse cenário.

Sendo assim, após a revisão sistemática, a qual envolveu a análise de 267 artigos, sendo apenas 17 considerados para a revisão final, notou-se que poucos trabalhos investigaram técnicas de proteção da camada física das redes 5G, focando sempre em cenários de camada de aplicação, como é o caso de (RODRÍGUEZ et al., 2022), em que os autores utilizam de redes neurais convolucionais e da técnica de transferência de aprendizado para treinar modelos de detecção de ataques de negação de serviço (do inglês: *Denial of Service* - DoS) em redes IoT. Além disso, outro cenário bastante focado nos artigos investigados é o de problemas de autenticação, com técnicas que variam desde identificação de biometria utilizando dados criptografados para o treinamento ao invés das impressões digitais dos usuários (AHMED; TAUFIQ; ARAFATUR, 2021), até a utilização de electromiogramas para a identificação de pacientes (KIM et al., 2022). Entretanto, dos 17 artigos, somente 2 abordaram técnicas de inteligência artificial para a proteção da camada física de redes 5G, sendo um deles relacionado a investigação de métodos de proteção contra ataques adversariais durante o processo de estimação de canal (CATAK et al., 2022). Neste, os autores utilizam da técnica de destilação de conhecimento para treinar redes neurais que sejam fortes contra ataques de envenenamento,

Com isso, o segundo artigo que aborda a segurança da camada física foi o que motivou a escrita deste trabalho, com a implementação de um *autoencoder* profundo (do inglês: *Deep Autoencoder* - DAE) para identificar dispositivos pertencentes a uma rede 5G-IoT (NOSOUHI et al., 2022). Neste, os autores levaram em consideração aspectos de construção tanto dos elementos individuais de um arranjo de antenas quanto parâmetros do arranjo como um todo. Além disso, em (BALAKRISHNAN et al., 2019), é provada a possibilidade de se utilizar do padrão de feixe para a identificação de um dispositivo em uma rede, independente dos dados transmitidos pelo arranjo. Este último utilizou de medições de dispositivos reais para construção do banco de dados, enquanto que o primeiro criou um ambiente de simulação para criar padrões de feixe simulados.

3 Fundamentação teórica

Neste capítulo, serão apresentados e detalhados todos os conceitos teóricos necessários para o completo entendimento do sistema implementado, assim como a técnica de inteligência artificial empregada para a identificação dos dispositivos.

3.1 Arranjo de fase de antenas

A potência do sinal recebido por uma antena normalmente é definida como:

$$P_{rx} = \frac{P_{tx}}{4\pi r^2} \tag{3.1}$$

Entretanto, em 3.1, não estão presentes os efeitos na potência causados pela frequência do sinal transmitido, nem tão pouco são levados em consideração os ganhos da antena transmissora G_{tx} e da antena receptora G_{rx}. Sendo assim, uma equação que descreve melhor o valor dessa potência recebida é:

$$P_{rx} = \frac{P_{tx}}{4\pi r^2}\frac{\lambda^2}{4\pi}G_{rx}G_{tx} \tag{3.2}$$

Assim, em 3.2, nota-se que a potência recebida é diretamente proporcional ao comprimento de onda λ, ou seja, é inversamente proporcional à frequência do sinal (, Wireless Networking and Communications Group, 2014). Em uma rede 5G na qual os dispositivos podem operar em faixas de frequências altas ($> 24GHz$), a potência do sinal cai consideravelmente. Desse modo, uma alternativa para compensar essa queda na intensidade do sinal é aumentar o ganho do transmissor e/ou receptor. Para isso, existem duas opções:

- Aumentar as dimensões das antenas: Essa abordagem, além de aumentar o ganho, também fará com que o sinal irradiado seja mais direcional. Porém, se for necessário irradiar em mais de uma direção, é necessário um esforço mecânico para mover toda a estrutura da antena, tornando essa opção inviável;
- Aumentar o número de antenas e formar um arranjo: Desse modo, o ganho G_{rx} ou G_{tx} irá ser o ganho do arranjo. Além disso, o sinal irradiado também será direcionado, em forma de feixe, e é possível mudar a direção desse feixe a nível de processamento digital de sinais, modificando o defasamento entre cada antena do arranjo.

Dessa forma, arranjos de antenas tornaram-se a solução para operação em altas frequências, tanto por resolver o problema da queda de potência devido ao aumento

de frequência, quanto por reduzir o tamanho necessário do arranjo e eliminar o esforço mecânico para irradiação em múltiplas direções. Com isso, em um arranjo com elementos idênticos, existem cinco unidades de controle que podem ser modificadas a fim de moldar o padrão de feixe do arranjo (BALANIS, 2016). São elas:

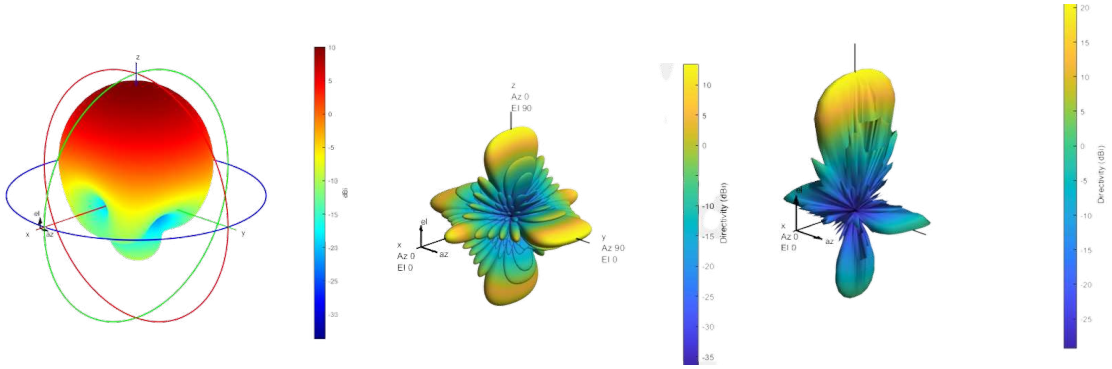
- Configuração geométrica do arranjo (Neste trabalho, utilizou-se um arranjo retangular);
- Distância relativa entre os elementos;
- Defasamento entre os elementos. Esse defasamento permite que a conformação do arranjo aconteça, direcionando o feixe para uma direção específica;
- O padrão individual de cada elemento.

Sendo assim, pode-se definir que o padrão de feixe é uma função que depende de dois fatores: O padrão individual de um elemento $E.P(\theta, \phi)$ e as contribuições do arranjo, chamadas de Fator de arranjo ($A.F(\theta, \phi)$). Ou seja:

$$A.P(\theta, \phi) = E.P(\theta, \phi) \times A.F(\theta, \phi) \tag{3.3}$$

Na qual θ e ϕ são os ângulos de azimute e elevação, respectivamente. Neste trabalho, apenas mudanças no fator de arranjo foram feitas, modificando o espaçamento entre cada elemento a fim de obter uma assinatura única para cada arranjo. Na figura 1 é possível observar o padrão individual, em 1(a), o fator de arranjo em 1(b) e o padrão de feixe da combinação dos dois fatores em 1(c). Essas imagens foram geradas para um arranjo $6x6$ projetado para operar em $28GHz$.

Figura 1 – Elementos que compõe o padrão de feixe de um arranjo



((a)) Padrão individual de cada elemento

((b)) Fator de arranjo

((c)) Padrão de feixe

3.2 5G - Gerenciamento de feixes

Uma característica chave do *design* do 5G-NR é a capacidade de operar em duas faixas de frequência diferentes: abaixo de 6 GHz (*Sub-6 GHz*) e as denominadas *mmWaves*. À medida que a disponibilidade de faixas abaixo de 6 GHz se torna mais limitada, as faixas de frequência *mmWave* com largura de banda mais ampla se tornarão predominantes. Estas, por sua vez, operam acima de $24GHz$ e são de curto alcance, com alta frequência, o que oferece maior capacidade.

A alta frequência dessas ondas faz com que sejam necessários arranjos de antenas para transmissão/recepção, como mencionado na seção anterior. Isso faz com que a irradiação do sinal transmitido seja direcionada, em um feixe, e se faz necessário que haja um protocolo de gerenciamento para que a comunicação entre o UE e a *gNodeB* aconteça.

As características como perda de propagação, bloqueio de sinal e efeitos de desvanecimento diferem das ondas *mmWave* para a faixa *sub-6 GHz*, o que introduz novos desafios no design do sistema e afeta a taxa de transferência e a qualidade da experiência do usuário. Essa taxa é um dos principais fatores para o sucesso do 5G. Para enfrentar essas limitações, os padrões 3GPP 5G NR definem novas características de camada física (PHY) e controle de acesso de meio (MAC) para suportar comunicações direcionais (3GPP, 2018). Entre as características importantes, destaca-se o gerenciamento de feixes, que é usado para estabelecer e manter os feixes de transmissão e recepção.

O gerenciamento de feixes é composto por seis partes (LI et al., 2020), porém, apenas as duas primeiras são relevantes para este trabalho. São elas: Varredura de feixe e medição de feixe.

3.2.1 Varredura de feixe

Essa fase do gerenciamento se caracteriza pela transmissão de vários feixes em diferentes direções. Por exemplo, durante o acesso inicial pelo UE para escolher o melhor feixe. A *gNodeB* transmite em todas as direções em sequência em intervalos de tempo regulares, e caso um UE esteja tentando estabelecer uma conexão, ele irá ler o SSB (do inglês, *Synchronization signal block* - Bloco de sincronização de sinal). Com isso, o UE extrairá o bloco transmitido pela *gNodeB*:

- Sinal de sincronização primária (PSS);
- Sinal de sincronização secundária (SSS);
- *Physical broadcast channel* (PBCH) e sinal de referência para desmodulação (DMRS).

Cada SSB transmitido pela *gNodeB* corresponde a um feixe específico conformado em uma direção específica. A cada $5ms$, a *gNodeB* transmite um certo número de SSBs

em sequência, e o número de blocos por grupo varia de acordo com a faixa de frequência, sendo 64 para as *mmWave*. A *gNodeB* continuamente transmite sequências de SSB a fim de estabelecer conexão com novos UEs. Com isso, nota-se que em uma rede 5G, a conexão nunca é iniciada pelo usuário, e sim pela estação base.

Em síntese, a varredura de feixe é caracterizada pelos múltiplos feixes de transmissão ou recepção com o objetivo de estabelecer ou manter uma conexão.

3.2.2 Medição de feixe

Nessa fase, a UE ou a *gNodeB* realiza medições de potência para avaliar a intensidade do feixe irradiado. Essas medições podem ser com base nos SSBs no caso do estabelecimento do acesso inicial, mas também pode ser baseadas nos sinais de referência sobre a informação do estado do canal (CSI-RS) em transmissões *Downlink*, ou em SRS (do inglês - *Sounding reference signal*), em transmissões *Uplink*. A principal medição realizada é a de potência do sinal de referência recebido (RSRP) (3GPP, 2020).

Além disso, o 3GPP também define três procedimentos de gerenciamento de feixes, cada um com o objetivo de estabelecer/manter a comunicação em um cenário específico., esses procedimentos são chamados de P-1, P-2 e P-3. Na tabela **??** é possível ver as diferenças entre cada um.

Tabela 1 – Diferenças entre os procedimentos de gerenciamento de feixe

Procedimento	**Varredura**	**Medição**	**Propósito**
P-1	UE e *gNodeB*	UE - SSB	Aquisição inicial
P-2	UE ou *gNodeB*	UE - CSI-RS *gNodeB* - SRS	Refinamento do feixe de transmissão
P-3	UE ou *gNodeB*	UE - CSI-RS *gNodeB* - SRS	Refinamento do feixe de recepção

O procedimento P-1 é utilizado para a aquisição inicial e se baseia na transmissão de SSBs pela *gNodeB*. Existe uma varredura tanto durante a transmissão quanto durante o recebimento, com o objetivo de estabelecer um feixe de conexão inicial.

O procedimento P-2 tem como objetivo o refinamento do feixe de transmissão. Para isso, o transmissor executa a varredura, irradiando sinais CSI-RS no caso de um refinamento *Downlink* ou sinais SRS em caso de *Uplink*, e o receptor realiza as medições com o feixe escolhido durante o estabelecimento da conexão inicial.

Por fim, o procedimento P-3 tem a função de refinar o feixe de recepção. Sendo assim, o transmissor irradia um único feixe, enquanto que o receptor varre o espaço com vários feixes a fim de estabelecer a melhor direção. Esse processo também se baseia na medição de CSI-RS ou SRS.

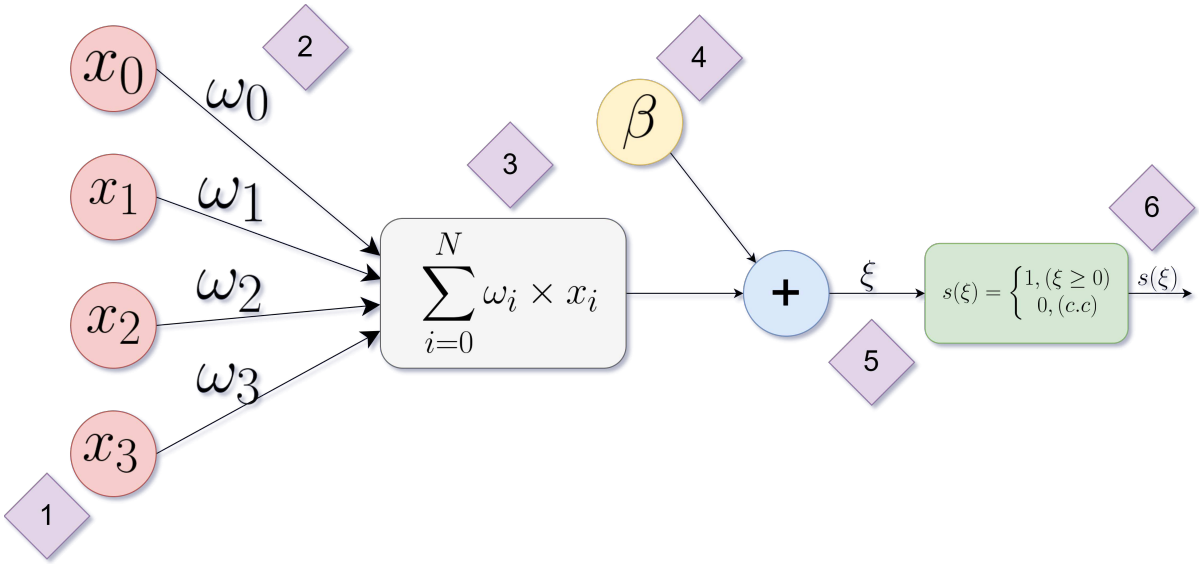
É importante notar que para que os procedimentos P-2 ocorra com transmissões *Uplink*, o UE deve suportar a conformação de feixes durante a irradiação. O mesmo raciocíno é válido para o procedimento P-3 com transmissão *Downlink*. Nesse caso, o UE também deve suportar a conformação para realizar a varredura durante o recebimento.

3.3 Aprendizagem Profunda

Aprendizagem profunda é uma área particular dentro do campo de aprendizado de máquina. Esta área é composta pelos modelos de redes neurais. A primeira menção atais estruturas na história foi feita em 1943 pelo neurofisiologista Warren McCulloch e o matemático Walter Pitts. Os dois cientistas escreveram um artigo explicando como modelar o funcionamento de um neurônio utilizando circuitos elétricos (MCCULLOCH; PITTS, 1943). Entretanto, somente em 1958, Frank Rosenblatt, um especialista em psicologia cognitiva propôs um aprimoramento ao modelo criado em 1943. Esse novo modelo, com pesos atribuídos às entradas, era composto por apenas duas camadas e foi chamado de *Perceptron* (ROSENBLATT, 1958). Tal estrutura veio posteriormente a se tornar a base para a construção de redes neurais mais complexas. Na figura 2, é possível observar os principais componentes dessa estrutura. Em (1), tem-se os neurônios de entrada x_i. Um perceptron possui N neurônios desse tipo. Cada um deles contém uma dos valores de entrada para o problema a ser solucionado. Em (2), é possível ver os pesos associados a cada uma das entradas (ω_i). O objetivo de uma rede neural é aprender durante a fase de treinamento os valores para esses pesos de modo que o problema proposto seja resolvido. Em (3), (4), (5) e (6) é possível ver as estruturas que formam o neurônio principal do *Perceptron*. Primeiramente, os pesos são multiplicados por cada uma das suas respectivas entradas, e em seguida esses valores são somados (3). Posteriormente, em (4), o valor β, chamado de limiar por Rosenblatt, porém mais tarde seria conhecido como viés, é adicionado ao resultado da somatória anterior (5). Em seguida, é aplicada uma função de ativação ao resultado. Nesse caso, a função de etapa binária foi utilizada (6). Em modelos mais recentes, várias outras funções de ativação podem ser utilizadas, cada uma tendo suas particularidades e sendo útil em situações específicas.

Essa estrutura era limitada a resolução de problemas lineares, e por muito tempo o avanço dessa tecnologia esteve bloqueado por limitações computacionais da época. Somente em 1986, Geoffrey Hinton propôs um método de aprendizado que possibilitaria que redes neurais com múltiplas camadas e vários *Perceptrons* aprendessem funções mais complexas. Esse método ficou conhecido como retropropagação.

Figura 2 – Arquitetura de um *Perceptron*



Fonte: Autoria própria

3.3.1 Aprendizado com retropropagação

Durante o aprendizado de uma rede neural profunda, a arquitetura executa de maneira iterativa dois passos repetidamente: Propagação direta e retropropagação. Inicialmente, todos os pesos são iniciados com valores aleatórios e a rede executa uma inferência, ou seja, uma predição da saída $\hat{\mathbf{y}}$ com base nos valores de entrada $\mathbf{x}$. Essa etapa gera uma função de custo, $J(\Theta)$. O objetivo final de qualquer rede neural é minimizar $J(\Theta)$. Para isso, a etapa de retropopagação é executada. Durante essa fase, o gradiente da função de custo é calculado para cada camada da rede. Uma vez que o gradiente mostra o sentido em que a função de custo cresce, os pesos são atualizados de modo que a função de custo decresça localmente. Esse processo é repetido até que o modelo encontre um possível mínimo global da função de custo, ponto esse em que não é possível mais aprender nenhuma informação.

3.3.2 Funções de ativação

Durante a implementação das redes neurais utilizadas neste trabalho, três funções de ativação foram utilizadas: *ReLU*, *Sigmoid* e *Tanh*. Desse modo, todas essas funções serão definidas para que haja um melhor entendimento dos modelos implementados.

3.3.2.1 *ReLU - Rectified Linear Unit*

A função de ativação ReLU é amplamente adotada na construção de redes neurais e sua equação pode ser definida como:

$$f(x) = max(0, x) \tag{3.4}$$

Em primeiro lugar, a função ReLU é não linear, o que facilita a propagação de erros retroativamente e permite o uso de múltiplas camadas de neurônios ativados por ReLU. Uma das principais vantagens dessa função é que ela não ativa todos os neurônios simultaneamente. Isso significa que, quando a entrada é negativa, a saída do neurônio é convertida em zero, o que é semelhante a desativá-lo. Isso resulta em redes esparsas e eficientes, tornando a computação mais eficaz. No entanto, é importante notar que a função ReLU pode enfrentar problemas com gradientes que se aproximam de zero, sendo nesse caso preferíveis outras funções de ativação.

3.3.2.2 *Sigmoid*

A função sigmoide pode ser definida como:

$$f(x) = \frac{1}{1 + e^{-x}} \tag{3.5}$$

Essa função possui um formato de "S" e possui imagem no intervalo de $(0, 1)$. Ela é extremamente útil em problemas de classificação, e tem o comportamento de fazer a entrada atingir seus extremos (0 ou 1). A vantagem em relação À função de etapa binária é que a *Sigmoid* é diferenciável em todos os pontos, o que possibilita o uso da técnica de retropropagação. Entretanto, a função sigmoide torna-se inativa quando seu valor de entrada é extremamente positivo ou negativo, resultando em uma função que se torna quase plana e não reage significativamente a pequenas variações em sua entrada (GOODFELLOW; BENGIO; COURVILLE, 2016).

3.3.2.3 *Tanh*

A função *tanh* pode ser definida como:

$$f(x) = \frac{2}{1 + e^{-2x}} - 1 \tag{3.6}$$

Esta é apenas uma função escalonada *Sigmoid*, e funciona de maneira semelhante, sendo muito utilizada para problemas de classificação. Entretanto, a *tanh* é simétrica em relação à origem, com uma imagem no intervalo $(-1, 1)$. Isso resolve alguns problemas existentes na *Sigmoid*, como a limitação da saída a valores unicamente positivos.

3.3.3 CNN - Rede neural convolucional

As redes neurais convolucionais são um tipo de rede neural artificial que se baseia no conceito de convolução. A convolução é uma operação matemática que permite extrair características relevantes de uma imagem ou sinal, como bordas, texturas e formas. As CNNs são amplamente utilizadas em tarefas de visão computacional, como reconhecimento

de objetos, classificação de imagens e detecção de padrões. (GOODFELLOW; BENGIO; COURVILLE, 2016)

Desse modo, elas foram uma evolução no cenário de modelos com imagens em relação às camadas totalmente conectadas, que são camadas de redes neurais tradicionais em que cada neurônio está conectado a todos os neurônios da camada anterior. As CNNs, por sua vez, utilizam camadas convolucionais que aplicam filtros em regiões específicas da imagem, permitindo que a rede aprenda características locais e invariantes a deslocamentos. Essa abordagem reduz significativamente o número de parâmetros da rede, tornando-a mais eficiente e capaz de lidar com imagens de alta resolução. Para entender a vantagem das CNNs, é importante definir dois princípios que esses modelos se baseiam: Invariância na tradução e Localidade (ZHANG et al., 2021).

- **Invariância na tradução**: Princípio pelo qual as primeiras camadas de uma rede neural convolucional responda da mesma forma à mesma janela de convolução, independente da localização dela na imagem;
- **Localidade**: Ainda sobre as primeiras camadas, estas devem focar em extrair características locais da imagem, e não em características gerais. Em camadas mais internas ao modelo, essas características são agregadas e informações mais gerais podem ser extraídas levando em consideração localidades distantes da imagem.

Á principal diferença em relação a uma rede neural densa é que os pesos em uma camada convolucional são convertidos em núcleos. Cada núcleo é uma janela de varredura que vai multiplicar cada um dos seus valores pela janela atual da imagem que está sendo varrida. A figura 3 apresenta como esse processo de varredura é feito. Além disso, na equação 3.7, esse mesmo processo é descrito matematicamente. Na equação, uma entrada de apenas um canal é utilizada como referência para fins de simplificação, mas a equação pode ser expandida para uma imagem de múltiplos canais.

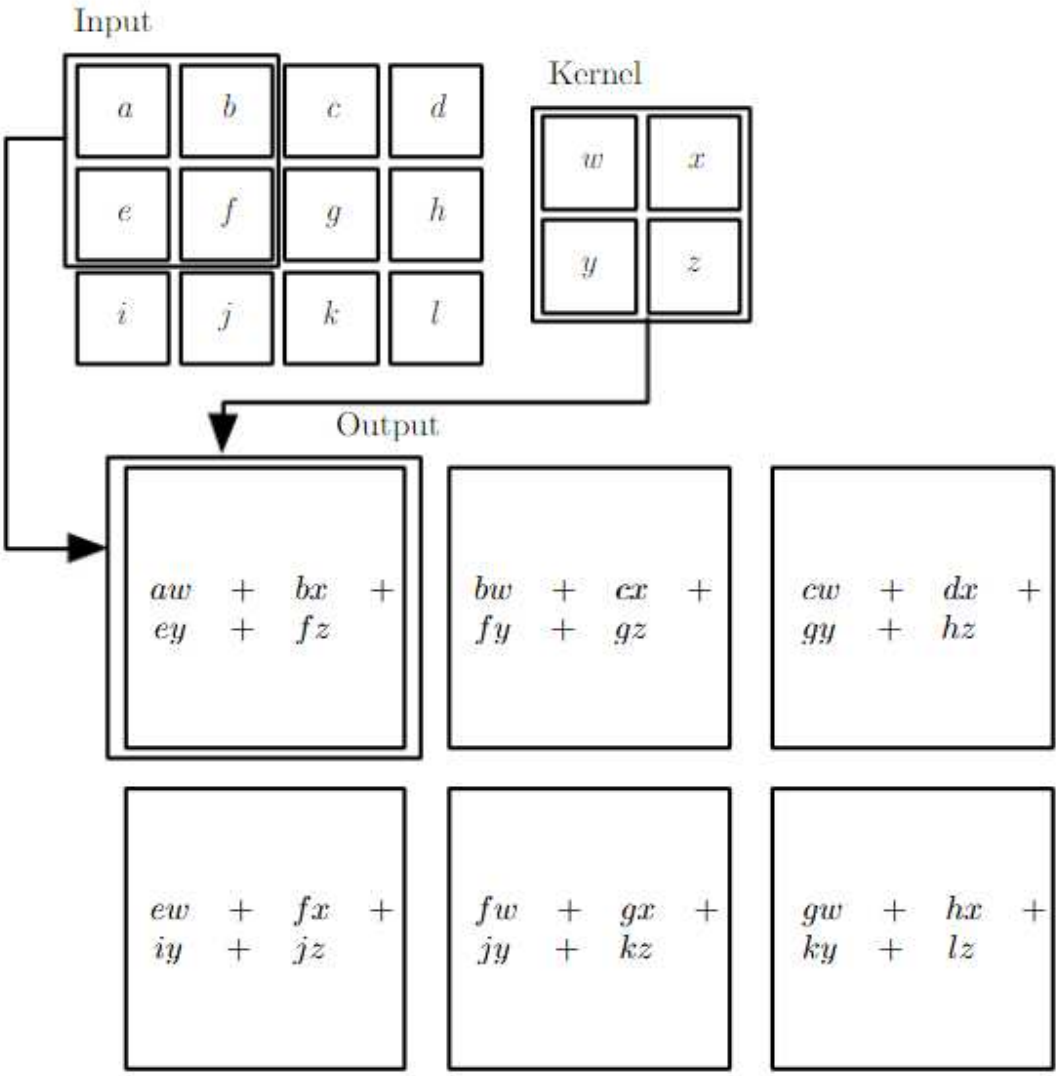
$$[\mathbf{H}_{i,j}] = \beta + \sum_{a=-\Delta}^{\Delta} \sum_{b=-\Delta}^{\Delta} [\mathbf{W}]_{a,b} [\mathbf{X}]_{i+a,j+b} \tag{3.7}$$

Onde:

- β é o viés aplicado à camada;
- Δ representa o tamanho da janela do *kernel*;
- a e b representam os elementos do *kernel*;
- $\mathbf{W}$ é o tensor *Kernel*;
- $\mathbf{X}$ é o tensor de entrada da camada;

- **H** é o tensor de saída da camada, antes de ser aplicada uma possível função de ativação.

Figura 3 – Varredura de uma imagem durante uma convolução



Fonte: (GOODFELLOW; BENGIO; COURVILLE, 2016)

É possível verificar que após a convolução, o tamanho da imagem é reduzida. Esse novo conjunto de dados agrega as características locais extraídas da imagem original. Novas convoluções podem ser feitas para que sejam extraídas mais características de locais mais distantes. Porém, normalmente antes de outra camada de convolução é inserida uma camada de *Pooling*, que tem como objetivo principal a redução da resolução espacial das representações ocultas, agregando informações para que quanto mais internas as camadas, maiores sejam os campos receptivos os quais as características serão extraídas.

Analogamente à convolução, existe também uma janela de varredura nas camadas de *Pooling*. Essa janela, porém, não irá multiplicar os valores como é feito no processo de varredura do *kernel*. Por outro lado, o *Pooling* aplica uma determinada função que tem

como objetivo reduzir o tamanho da representação. Os tipos de *Pooling* mais utilizados são o *Pooling* máximo e o *Pooling* médio. O primeiro converte toda a janela atual no valor máximo dentre todos os valores, enquanto o segundo converte a janela em um elemento cujo valor é igual à media de todos os elementos da janela.

Desse modo, uma CNN normalmente é composta por camadas convolucionais e de *Pooling* de forma intercaladas até um certo ponto de redução, no qual uma camada de achatamento é aplicada e posteriormente camadas totalmente conectadas são utilizadas para a parte final da rede neural. Essa combinação extrai de maneira muito mais eficiente características dos bancos de dados quando estes possuem alguma correlação espacial entre seus valores.

3.3.4 DAE - Deep Autoenconder

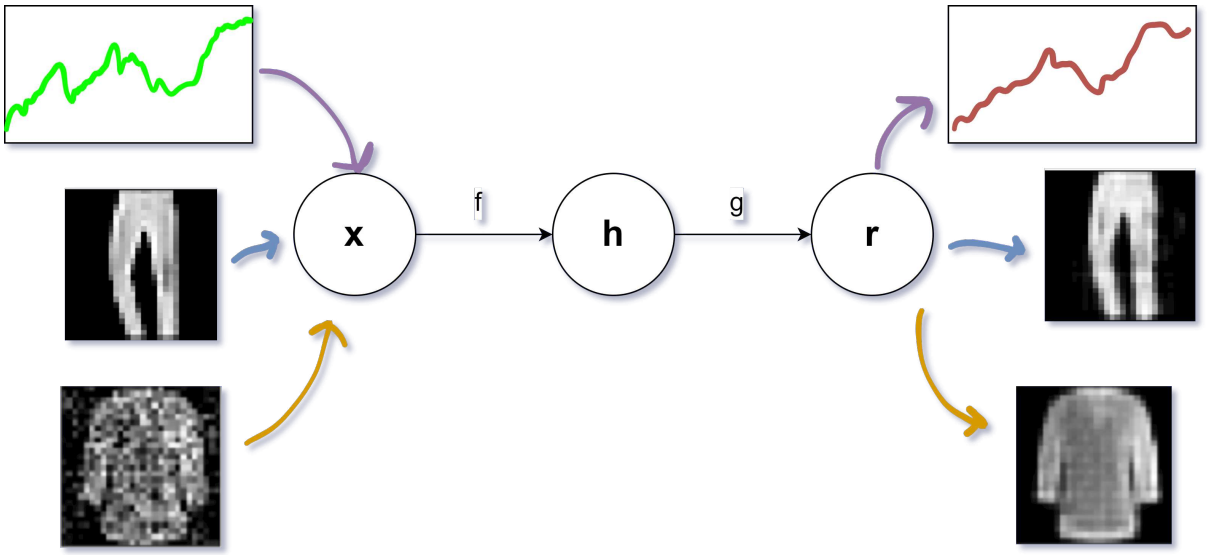
Um *autoencoder* é uma rede neural que visa aprender a copiar sua entrada para a saída, utilizando uma camada oculta para representar o conteúdo da entrada (GOODFELLOW; BENGIO; COURVILLE, 2016). Sua estrutura pode ser dividida em duas partes: a função de codificação, que mapeia a entrada para a camada oculta ($h = f(x)$), normalmente comprimindo a entrada à medida que codifica, e o decodificador, que cria uma reconstrução a partir da camada oculta ($r = g(h)$), descomprimindo durante o processo. Desse modo, a reconstrução não tem como foco uma cópia exata da entrada, visto que durante todo processo de codificação existem perdas inerentes. Assim, o modelo seleciona quais aspectos da entrada são mais importantes, aprendendo a reconstruir de forma aproximada dados semelhantes aos que foram utilizados no seu treinamento.

Os *autoencoders* são utilizados tradicionalmente para redução de dimensionalidade e extração de características (HINTON; SALAKHUTDINOV, 2006). No entanto, conexões teóricas mais recentes entre *autoencoders* e modelos de variáveis latentes os tornaram essenciais para a modelagem generativa. Os *autoencoders* podem ser considerados como uma variação de redes *feedforward* e são treinados com técnicas de descida de gradiente convencionais, com o método de retropropagação para calcular os gradientes. Além disso, suas aplicações vão além de simplesmente reconstruir a entrada na saída. Uma das possibilidades com esse tipo de modelo, utilizada neste trabalho, é a detecção de anomalias. Nesse caso, a reconstrução é comparada com a entrada utilizando a função de custo do erro médio quadrático, definida na equação 3.8. Assim, o erro é calculado para todas as amostras de treinamento, e um limiar é atribuído com base no erro médio das amostras. Desse modo, reconstruções que possuem um erro médio maior que o limiar encontrado são consideradas anomalias, uma vez que o DAE não aprendeu a reproduzir tais entradas.

$$MSE = \frac{1}{N}\sum_{i=1}^{n}(Y_i - \hat{Y}_i)^2 \tag{3.8}$$

Na figura 4, é possível verificar possíveis arquiteturas para um DAE, desde modelos com etradas unidimensionais, até modelos que utilizam de convolução, sendo estes referidos neste trabalho como ConvDAEs.

Figura 4 – Estruturação de um *autoencoder*



Fonte: Autoria própria

3.3.5 Métricas de desempenho

Definidas as redes neurais, é importante definir as métricas de desempenho que foram utilizadas para a comparação entre os diferentes modelos. Foram escolhidas quatro métricas dentre as várias existentes no âmbito de aprendizagem profunda: acurácia, precisão, revocação e especificidade. Todas essas métricas são definidas aqui tomando como base um modelo de classificação binária, na qual os rótulos são: **Positivo** ou **Negativo**. Supõe-se um cenário semelhante ao visualizado na figura 5. Essa figura servirá como base para definir cada uma das métricas supracitadas.

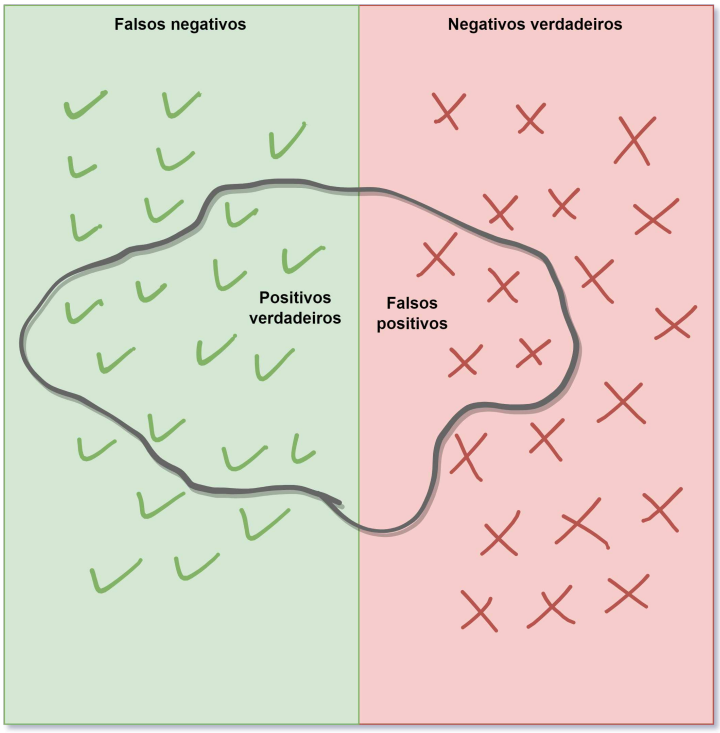
- **Acurácia** - Essa medida pode ser definida como:

$$Acc = \frac{TP + TN}{TP + FP + TN + FN}$$

  Na qual:

  - $TP \rightarrow$ Positivos verdadeiros, ou seja, valores verdadeiros preditos corretamente;
  - $FP \rightarrow$ Falsos positivos, ou seja, valores falsos preditos como verdadeiros;
  - $TN \rightarrow$ Negativos verdadeiros, ou seja, valores falsos preditos corretamente;
  - $FN \rightarrow$ Falsos negativos, ou seja, valores verdadeiros preditos como falsos;

Figura 5 – Exemplo de classificação binária



Fonte: Autoria própria

A acurácia tem como principal objetivo fornecer uma visão geral dos acertos do modelo treinado. Porém, ela não diz nenhuma informação sobre se o modelo acerta mais previsões de valores falsos ou verdadeiros;

- **Precisão** - Pode ser definida como:

$$Precisão = \frac{TP}{TP + FP}$$

Essa medida indica a porcentagem de valores preditos verdadeiros que de fato são verdadeiros. Para um modelo sem erros, o número de falsos positivos será nulo, e o valor de precisão será unitário;

- ***Revocação*** - Este pode ser definido como:

$$Revocação = \frac{TP}{TP + FN}$$

Essa medida indica a porcentagem de valores verdadeiros preditos em relação a todos os valores verdadeiros.

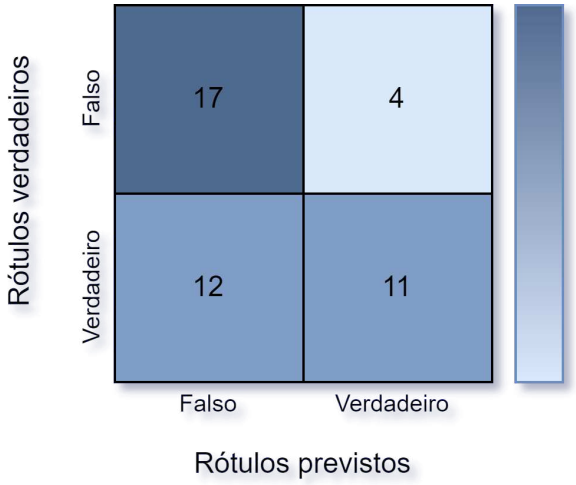
- **Especificidade** - Pode ser definida como:

$$Especificidade = \frac{TN}{TN + FP}$$

Essa medida indica o oposto da precisão. Ou seja, a porcentagem de valores preditos como falsos que de fato são falsos.

Por fim, também é possível agregar esses indicadores em uma visualização chamada matriz de confusão. Essa matriz quadrada, sendo nesse caso de classificação binária de tamanho 2×2, tem como objetivo analisar de maneira mais rápida a eficiência de um determinado modelo. Para isso, basta observar a diagonal principal dessa matriz, composta por positivos verdadeiros e negativos verdadeiros. É esperado que essa diagonal apresente os maiores valores, indicando que o modelo identificou corretamente a maior parte dos dados. Além disso, normalmente um gradiente de cores é aplicado a essa representação, associando valores maiores à cores mais fortes dentro do mesmo gradiente. Um exemplo de matriz de confusão pode ser visto na figura 6, feita a partir do cenário hipotético da figura 5.

Figura 6 – Exemplo de matriz de confusão



Fonte: Autoria própria

Nesse caso, têm-se um modelo com uma alta especificidade $(80,95\%)$, baixa acurácia $(63,64\%)$, precisão mediana $(73,33\%)$ e revocação extremamente baixa $(47,83\%)$.

4 Implementação

Neste capítulo, dividido em três partes, serão detalhados primeiramente os processos de modelagem dos arranjos de antenas, citando também as dificuldades encontradas durante o processo e as tomadas de decisões para superá-las. Depois, a construção do sistema de medições para a geração do banco de dados. Posteriormente, será detalhado como esse banco de dados foi utilizado para o treinamento dos modelos de aprendizagem profunda, descrevendo os três tipos de redes neurais utilizados como comparação.

4.1 Modelagem dos arranjos de antenas

Os dispositivos que operam na faixa de frequência das *mmWaves* normalmente utilizam de arranjos de antenas retangulares. Dessa forma, todos os modelos gerados para esse trabalho foram criados com o objeto *phased.ConformalArray*, que permite uma configuração individual de cada elemento. Dessa forma, durante a construção dos arranjos, a distância entre cada elemento foi variada dentro de uma tolerância de 5% do valor ideal.

Desse modo, durante a modelagem, primeiramente são geradas as posições que cada elemento do arranjo irá ocupar, e isso foi feito tomando o centro do arranjo como origem, posicionando cada elemento com base nesse ponto. As equações 4.1 e 4.2 mostram como foi possível posicionar os elementos para um caso no qual o arranjo possui N par. As equações 4.3 e 4.4 realizam a mesma operação, porém para arranjo com N ímpar (ou seja, a origem do plano cartesiano YZ contém um elemento). Nas equações, d é a distância entre cada elemento do arranjo, sendo a distância referencial escolhida sendo $d = 4mm$.

$$y_{ji} = -\left\lfloor \left| \frac{N-1}{d} \right| \right\rfloor d - \frac{d}{2} + (i-1)d \pm 0.05d \tag{4.1}$$

$$z_{ji} = \left\lfloor \left| \frac{N-1}{d} \right| \right\rfloor d + \frac{d}{2} + (j-1)d \pm 0.05d \tag{4.2}$$

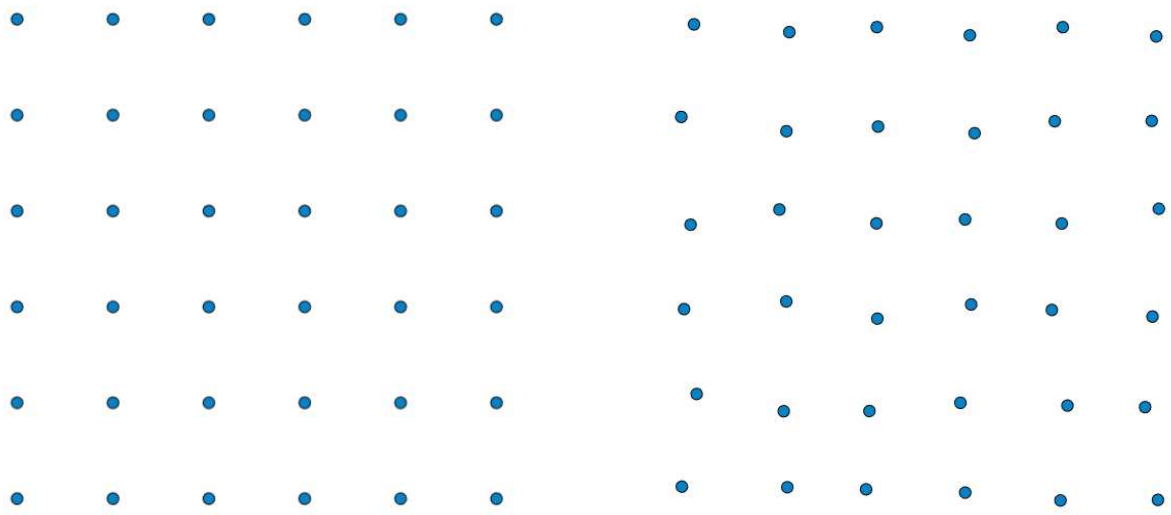
$$y_{ji} = -\left\lfloor \left| \frac{N-1}{d} \right| \right\rfloor d + (i-1)d \pm 0.05d \tag{4.3}$$

$$z_{ji} = \left\lfloor \left| \frac{N-1}{d} \right| \right\rfloor d + (j-1)d \pm 0.05d \tag{4.4}$$

Em todos os casos, essas equações foram implementadas em um laço no qual a variável i foi iterada de 1 à N, com a variável j aninhada sendo iterada também de 1 à N, percorrendo assim toda a matriz $N \times N$ e formando uma nova matriz na qual cada posição

contém um ponto (y_{ji}, z_{ji}). Esse conjunto de valores foi então rearranjado em um vetor unidimensional que é um dos argumentos necessários para a criação de uma *Conformal Array* no MATLAB. Na figura 7 é possível ver a diferença entre um arranjo de antenas ideal (figura 7(a)) e um construído com as imperfeições citadas (figura 7(b)).

Figura 7 – Posição dos elementos no arranjo de antenas - Arranjo $6x6$



((a)) Arranjo ideal

((b)) Arranjo com randomização das posições

Fonte: Autoria própria

O MATLAB possui uma *Toolbox* voltada para a criação e simulação de arranjos de antenas, a *Phased Array System Toolbox*. Inicialmente, o aplicativo *Antenna Array Designer* foi utilizado para a criação dos arranjos, porém o nível de detalhes dos modelos requeriam um poder computacional que não se tinha acesso. Desse modo, outros parâmetros de construção, como a permissividade e espessura do substrato, além da largura e comprimento de cada elemento não foram levados em consideração, porém também são fatores que modificam o padrão de feixe, contribuindo para a assinatura de cada arranjo.

Posteriormente, alguns *scripts* auxiliares foram implementados para iterar L vezes por esse algoritmo, variando os parâmetros e salvando os arranjos em arquivos ".mat", que foram utilizados depois na geração do banco de dados. Desse modo, foram obtidos todos os arranjos que simulariam os UEs durante a geração do banco de dados.

4.2 Geração do banco de dados

Em posse dos dispositivos, um sistema de transmissão e recepção foi implementado com base no exemplo do MATLAB(NR. . . ,) no qual uma transmissão *Downlink* é feita pela gNodeB e as medições de *CSI-RS* são feitas no UE. O modelo de aquisição de padrões de feixe aproveita o fato de que durante o processo de gerenciamento de feixes, as medições

de potência do dispositivo transmissor são coletadas (KOTTKAMP et al., 2019). Desse modo, não há nenhuma carga computacional adicional na gNodeB, sendo essas medições apenas enviadas para o servidor no qual o modelo de reconhecimento está localizado.

Nesse ponto, duas abstrações foram feitas sem que houvesse alterações na qualidade de dados coletados para o treinamento. Primeiramente, em um cenário real, os padrões seriam coletados em uma transmissão *Uplink* na qual o UE transmite os SRS para a gNodeB a fim de refinar ainda mais o ângulo de transmissão. Entretanto, nesse trabalho foi feita a simplificação de considerar que o mesmo processo de medição feito no *Downlink* seria feito no *Uplink*, e os sinais utilizados para a construção do banco de dados foram os CSI-RS do exemplo citado acima.

Além disso, o tempo necessário para transmitir um sinal em um determinado ângulo é muito maior do que o necessário para realizar uma medição, por conta dos procedimentos de conformação de feixe e modulação OFDM necessários na transmissão. Com isso, o tempo de execução do algoritmo de geração de banco de dados tornou-se muito alto, sendo inviável a obtenção de dados suficientes para o treinamento dentro do tempo disponível para a realização deste trabalho. Sendo assim, para a obtenção dos padrões, outra abstração foi feita. Ao invés de transmitir em vários ângulos diferentes no UE, medindo em um mesmo ângulo na gNodeB (o que resultaria no padrão de feixe da UE), o que foi feito foi transmitir uma única vez na gNodeB e medir em vários ângulos no UE. Desse modo, o artefato final é o padrão de feixe daquele UE, que é a entrada necessária para o treinamento dos modelos de inteligência artificial.

Em síntese, essas abstrações tornam a geração dos padrões distantes do que realmente seria feito em um cenário real. Porém, isso não traz nenhum malefício ao trabalho, pois esse sistema só foi implementado devido à impossibilidade de se realizarem medições reais, e o objetivo era unicamente ter um vetor de medições que possuísse os padrões de feixe de cada UE, e esse objetivo foi alcançado com tal abordagem.

Assim, os bancos de dados usados no treinamento contém várias medições de potência para diferentes ângulos θ e ϕ de recepção na UE. Sendo assim, cada medição consiste em um vetor de K posições, sendo esse vetor $\{A.P(\theta_0, \phi_0), A.P(\theta_1, \phi_1), A.P(\theta_2, \phi_2), ..., A.P(\theta_{K-1}, \phi_{K-1})\}$, onde $A.P(\theta, \phi)$ representa o padrão de feixe de um determinado UE para os ângulos θ e ϕ. Com isso, foram gerados três conjunto de dados diferentes, cada um contendo padrões de feixe de um em quatro dispositivos que possuem um arranjo de antenas $6x6$. O primeiro banco de dados, que será referido como *StaticPosAllUEs_Dataset*, possui medições feitas para uma única posição de UE, modificando apenas a UE para cada conjunto de medidas. Esse banco de dados tem como objetivo comprovar se a detecção se baseia unicamente nos padrões de feixe e não nas variações causadas pela distância ou ângulo de transmissão das antenas. O segundo banco de dados, referido aqui como *StaticPosSingleUE_Dataset* possui medições feitas para os quatro dispositivos em locais

diferentes, porém sem a variação das posições durante o processo. Por fim, o último banco de dados, *DynamicPos_Dataset* possui medições feitas com os quatro dispositivos variando suas posições durante o processo de medição, permitindo assim a construção de um modelo um pouco mais robusto. Cada um dos conjuntos foi dividido em duas partes: uma para o treinamento e uma para o teste dos modelos. Na tabela 2, é possível observar todos os parâmetros utilizados na geração de cada um dos conjuntos de dados.

Tabela 2 – Parâmetros para geração dos bancos de dados

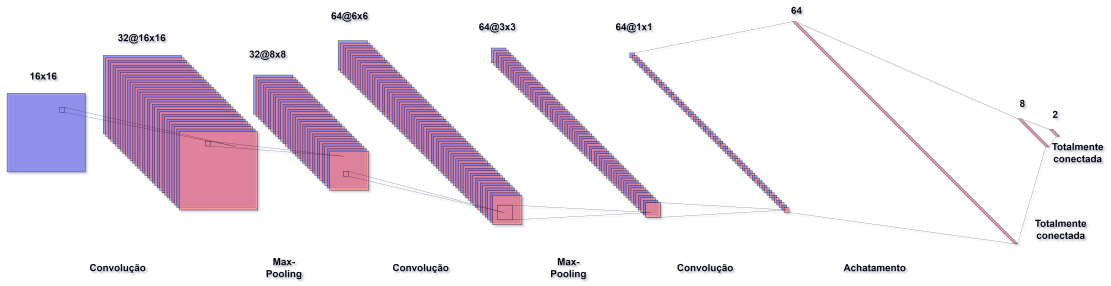
	StaticPosAll	***StaticPosSingle***	***DynamicPos***
Tamanho do arranjo transmissor (gNodeB)	$16x16$		
Tamanho do arranjo receptor (UE)	$6x6$		
Posição do arranjo transmissor	Fixa		
Posição do arranjo receptor	Fixa para todos os UEs	Fixa para cada UE individualmente	Variadas para cada UE
Variação do ângulo de medição no UE	$\pm 15^o$ em torno do ângulo de incidência direta, para θ e ϕ		
SNR do canal simulado	$30dB$		
Faixa de frequência	FR2		
Frequência central	$28GHz$		

4.3 Implementação dos modelos

O primeiro modelo utilizado para o treinamento foi o de redes neurais convolucionais (CNN), utilizado para modelos que lidam com imagens. Convertendo os vetores do banco de dados para sua representação em função do azimute e elevação, obtém-se uma forma análoga a uma imagem, e esse modelo pode ser utilizado. Este foi construído com três camadas convolucionais e uma camada densa, tendo uma entrada de $(16x16)$ e uma saída binária, classificando o dispositivo como *Verdadeiro* ou *Falso*. Esse modelo tem uma desvantagem prática, pois para o treinamento dele, foi necessário o uso de padrões de antenas que não pertenciam a rede. Essas medições, em uma aplicação real, teriam que ser criadas e não medidas, pois um sistema real teria apenas as medições dos dispositivos

autenticados. Na figura 8 é possível ver a estruturação dessa rede neural. Além disso, na tabela 3 são sumarizados todos os parâmetros utilizados para a implementação deste modelo. Na tabela existem três valores para o número de épocas, sendo o primeiro para o *StaticPosAllUEs_Dataset*, o segundo para o *StaticPosSingleUE_Dataset* e o terceiro para o *DynamicPos_Dataset*.

Figura 8 – Arquitetura da rede neural convolucional



Fonte: Autoria própria

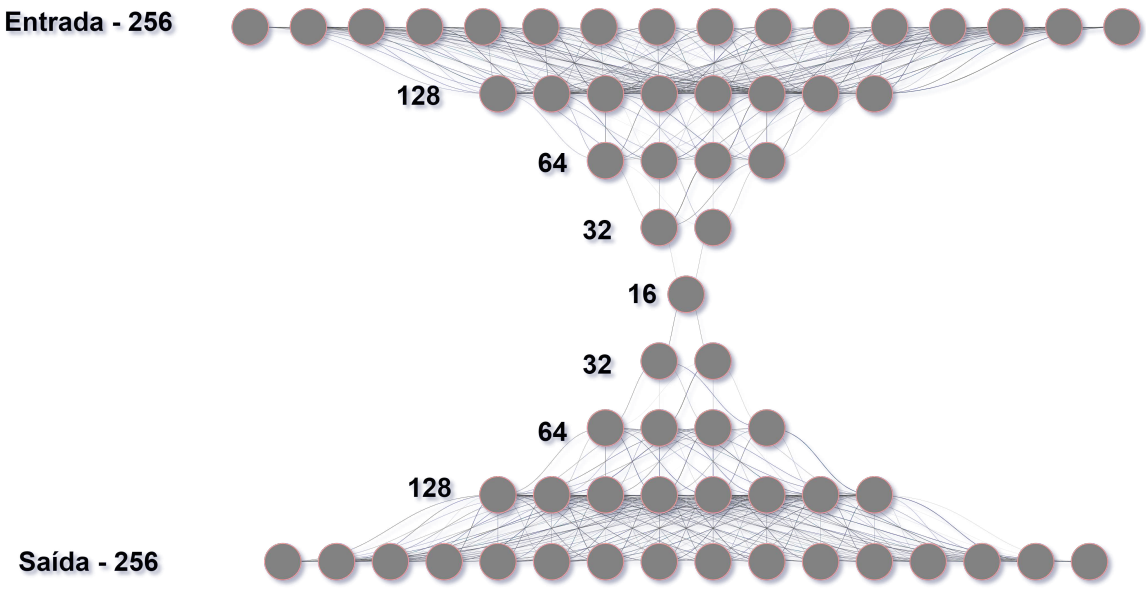
Tabela 3 – Parâmetros e hiperparâmetros utilizados na CNN

Parâmetros	**Valor/Descrição**
Tamanho da entrada	16x16
N° de camadas	3 convolucionais + 2 totalmente conectadas
Otimizador	Adam
Função de custo	*Sparse Categorical Cross Entropy*
N° de épocas	5/5/10
Tamanho do batch	32
Ativação em camadas intermediárias	ReLU
Ativação na camada de saída	Sigmoid

O segundo modelo utilizado neste trabalho foi um *Deep autoencoder - DAE*. O principal objetivo desse tipo de modelo é tentar reconstruir a entrada na saída. Para isso, ele comprime o dado de entrada e posteriormente descomprime na saída, tentando reconstruir as informações perdidas durante a compressão. No cenário de detecção de anomalia, um modelo treinado para reconstruir o padrão de feixes de um dispositivo pertencente a rede não conseguirá reconstruir os padrões oriundos de dispositivos invasores. Sendo assim, com base no erro de reconstrução é possível identificar se o UE é um invasor ou pertence à rede em questão. Na figura 9, é possível observar a arquitetura do DAE implementada para o treinamento dos modelos. O número de neurônios no diagrama foi

reduzido por um fator de 16 para uma melhor visualização. De forma análoga à rede neural convolucional, os parâmetros de treinamento para o DAE pode ser visto na tabela 4.

Figura 9 – Arquitetura do DAE



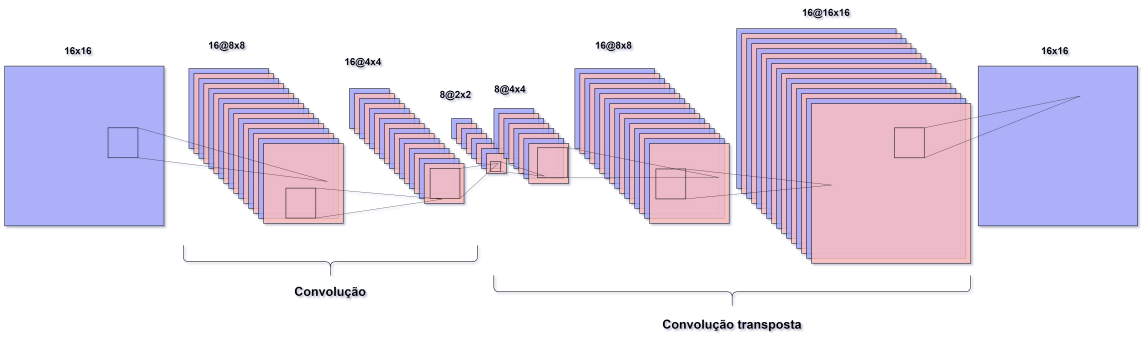
Fonte: Autoria própria

Tabela 4 – Parâmetros e hiperparâmetros utilizados no DAE

Parâmetros	**Valor/Descrição**
Tamanho da entrada	256
N° de camadas	5 de codificação + 4 de decodificação
Otimizador	Adam
Função de custo	Erro médio quadrático
N° de épocas	25/25/50
Tamanho do batch	32
Ativação em camadas intermediárias	ReLU
Ativação na camada de saída	tanh

Por fim, o último modelo implementado foi um DAE convolucional. O objetivo dele é o mesmo do DAE convencional. Porém, sua implementação consiste na criação de camadas convolucionais ao invés de camadas totalmente conectadas. Sendo assim, ele é útil para o treinamento de modelos de reconstrução de imagens. Convertendo novamente os vetores para um tamanho $16x16$, é possível utilizar essa abordagem e tentar extrair mais características dos padrões. Na figura 10 é apresentada a arquitetura dessa rede, e na tabela 5 os parâmetros de treinamento são listados.

Figura 10 – Arquitetura do DAE convolucional



Fonte: Autoria própria

Tabela 5 – Parâmetros e hiperparâmetros utilizados no ConvDAE

Parâmetros	**Valor/Descrição**
Tamanho da entrada	256
N° de camadas	5 de codificação + 4 de decodificação
Otimizador	Adam
Função de custo	Erro médio quadrático
N° de épocas	25/25/50
Tamanho do batch	32
Ativação em camadas intermediárias	ReLU
Ativação na camada de saída	tanh

Todos os parâmetros e hiperparâmetros foram obtidos utilizando heurísticas. Desse modo, não necessariamente as soluções obtidas são ótimas, mas a partir de uma abordagem experimental, obteve-se essas arquiteturas.

Sendo assim, definidas as redes neurais, no próximo capítulo serão comparados os resultados do treinamento tomando como base as métricas definidas durante a fundamentação teórica: acurácia, precisão, revocação e especifidade.

5 Resultados

Este capítulo irá detalhar os resultados obtidos a partir dos bancos de dados criados no *MATLAB*. Todos os gráficos foram gerados em *Python* com as bibliotecas *Matplotlib* e *Seaborn*, e os modelos foram treinados e testados utilizando a infraestrutura de desenvolvimento *Tensorflow*. A estruturação deste capítulo será a seguinte:

1 Visualização dos padrões de feixes de cada UE;

2 Definição das redes neurais implementadas;

3 Comparação das métricas para o banco de dados *StaticPosAllUEs_Dataset*;

4 Comparação das métricas para o banco de dados *StaticPosSingleUE_Dataset*;

5 Comparação das métricas para o banco de dados *DynamicPos_Dataset*.

As métricas utilizadas para comparação foram as definidas previamente: acurácia, precisão, revocação e especificidade. Além disso, para todos os bancos de dados, a matriz de confusão gerada a partir da previsão da parcela do banco de dados reservada para testes.

Como mencionado na metodologia, foram gerados três bancos de dados distintos, cada um com um objetivo específico. Cada elemento desses bancos de dados possui um vetor de medições com 256 amostras. Essas amostras representam a variação do ângulo de transmissão tanto em azimute quanto em elevação. Além disso, para o treinamento, os bancos de dados foram divididos em duas parcelas, uma voltada para o treinamento (com 80% dos vetores) e uma voltada para a inferência (com 20% dos valores). Essa divisão foi feita de forma aleatória utilizando as funções internas do *Tensorflow*.

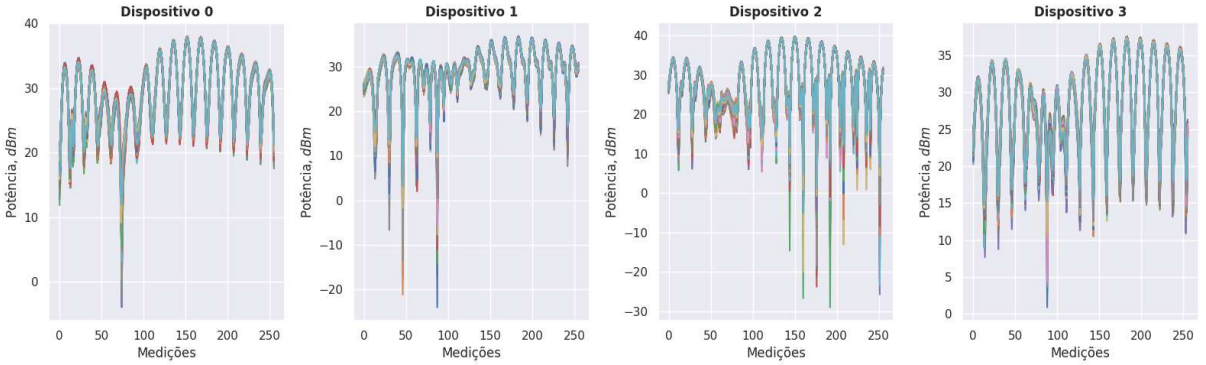
5.1 Padrões de feixe dos dispositivos

Essa seção irá apresentar visualmente os padrões de feixes medidos para cada um dos bancos de dados, tanto em uma vista bidimensional quanto em uma vista tridimensional.

Na figura 11 é possível observar os padrões para o banco de dados *StaticPosAllUEs_Dataset*. Cada janela na imagem contém 1000 padrões sobrepostos de um mesmo UE. Nessa visão, o eixo X indica apenas o índice do vetor de medições, sendo cada um uma combinação (θ_m, ϕ_m) diferente. Pode-se visualizar uma semelhança entre os padrões de cada dispositivo, assim como a diferença entre os padrões de UEs diferentes. Nesse caso, isso se deve unicamente às imperfeições de fabricação de cada antena, pois os dispositivos

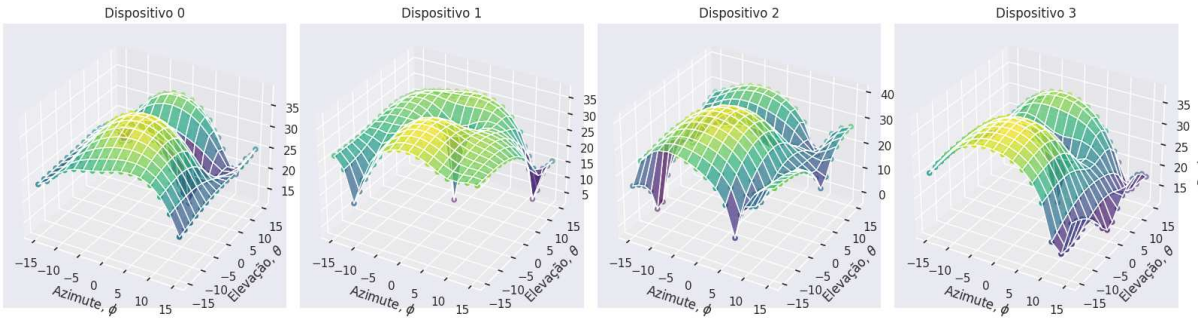
foram simulados exatamente com a mesma posição em relação à *gNodeB* para as medições de potência. Na figura 12, nota-se o padrão de feixe para cada UE em uma visão tridimensional, na qual as coordenadas representam de fato os ângulos de transmissão referenciados com base no ângulo de incidência direta (ponto $(0, 0, P)$, no qual P é a potência medida). Uma vez que o canal simulado não é ideal, não necessariamente esse ângulo irá apresentar a maior potência medida, e isso também pode ser visualizado na vista tridimensional.

Figura 11 – Padrões de feixe - *StaticPosAllUEs_Dataset*



Fonte: Autoria própria

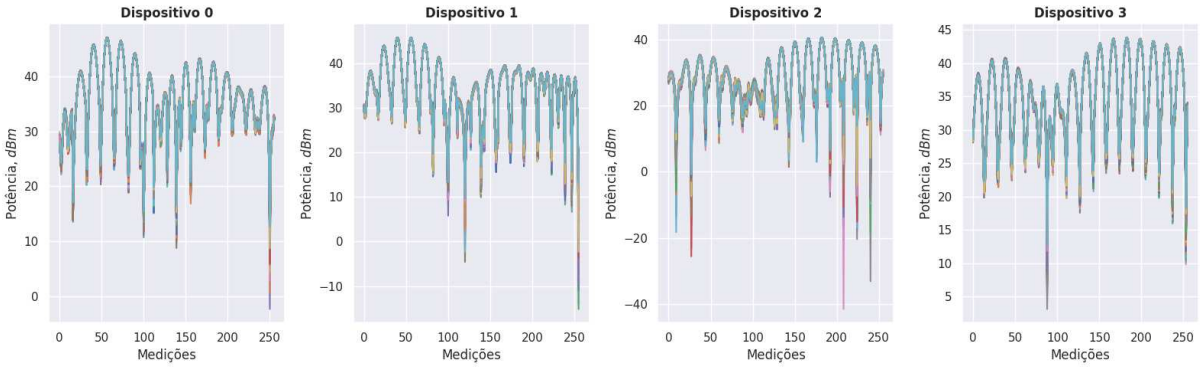
Figura 12 – Padrões de feixe em três dimensões - *StaticPosAllUEs_Dataset*



Fonte: Autoria própria

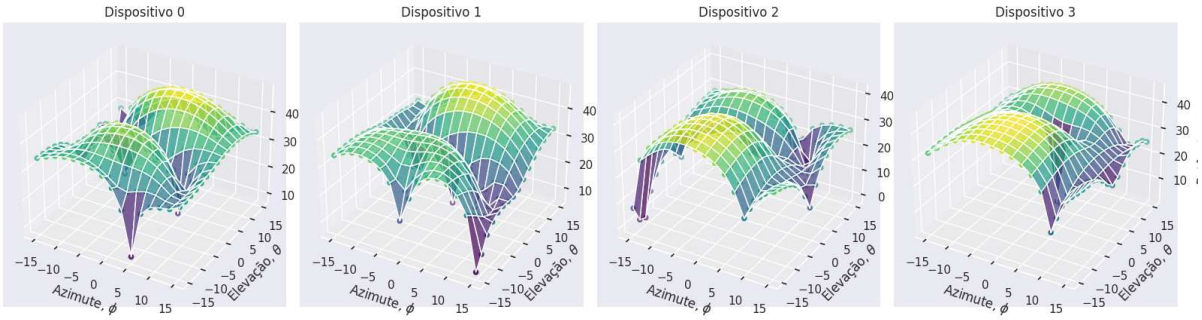
O segundo banco de dados, obtido através das medições de potência com os UEs estáticos, porém em posições distintas entre si, possui um conjunto de padrões de feixe que possuem mais diferenças entre os dispositivos, mas ainda semelhantes para padrões gerados pelo mesmo arranjo de antenas. É possível observar essa semelhança analisando a sobreposição das curvas do mesmo UE na figura 13. Essa sobreposição causa a impressão de que existe apenas uma curva com uma espessura de linha maior, porém ainda são 1000 curvas sobrepostas. Na figura 14, a visão tridimensional dos padrões pode ser vista, assim como para o *StaticPosAllUEs_Dataset*.

Figura 13 – Padrões de feixe - *StaticPosSingleUE_Dataset*



Fonte: Autoria própria

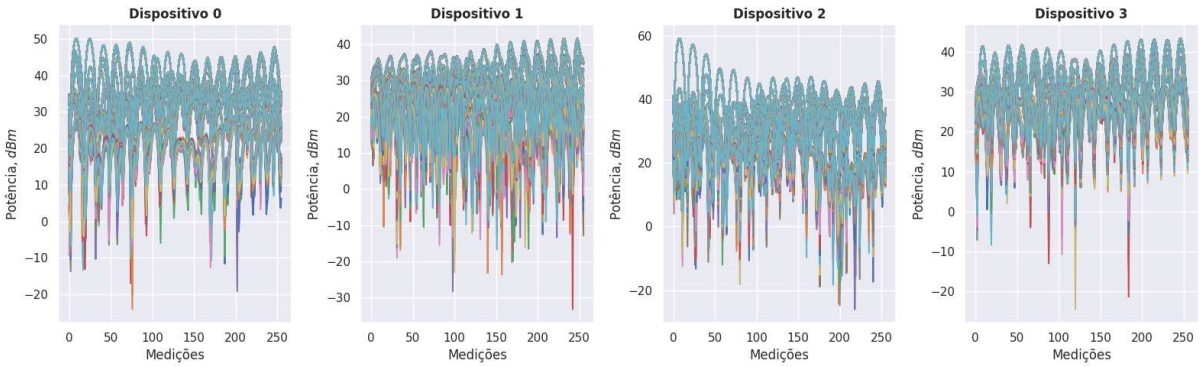
Figura 14 – Padrões de feixe em três dimensões - *StaticPosSingleUE_Dataset*



Fonte: Autoria própria

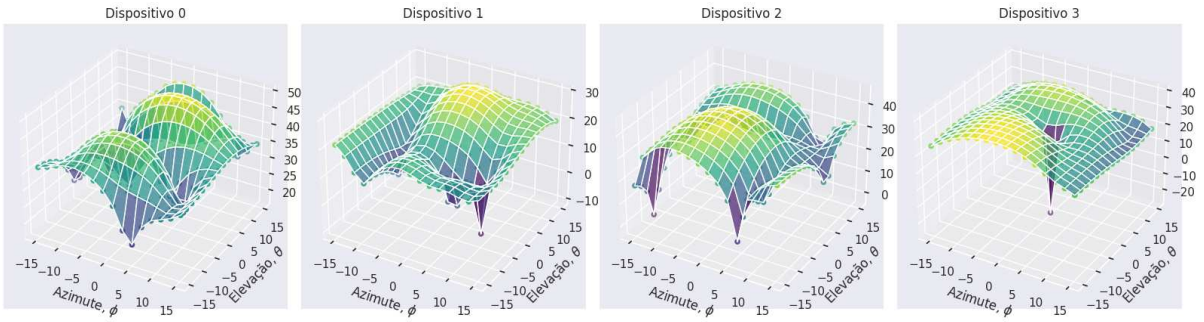
Por fim, as medições que resultaram no terceiro banco de dados, o *DynamicPos_Dataset* são visualizadas através dos padrões de feixe nas figuras 15 e 16. Neste cenário, os UEs tiveram suas posições variadas ao longo das medições, registrando assim os padrões transmitidos em diferentes posições e com angulações distintas. Isso faz com que a visualização de um padrão entre as curvas de um mesmo dispositivo seja menos visível para um visualizador comum, e também com que o treinamento seja um pouco mais difícil, mas também é responsável por uma maior robustez dos modelos de reconhecimento.

Figura 15 – Padrões de feixe - *DynamicPos_Dataset*



Fonte: Autoria própria

Figura 16 – Padrões de feixe em três dimensões - *DynamicPos_Dataset*



Fonte: Autoria própria

5.2 Comparação de métricas

Esta seção irá mostrar os resultados de treinamento e inferência de cada uma das redes neurais utilizadas neste trabalho de conclusão de curso, assim como a comparação entre cada modelo.

5.2.1 *StaticPosAllUEs_Dataset*

Para a CNN, foram necessárias apenas cinco épocas para o treinamento da rede. Isso se deve ao fato que os padrões de feixes de cada UE possuem diferenças nítidas, de modo que o modelo rapidamente identificou as semelhanças nas imagens geradas. Os DAEs, entretanto, precisaram de 25 épocas para um treinamento eficaz. Além disso, de modo geral, nesse caso a rede neural convolucional apresentou resultados melhores em todas as métricas possíveis.

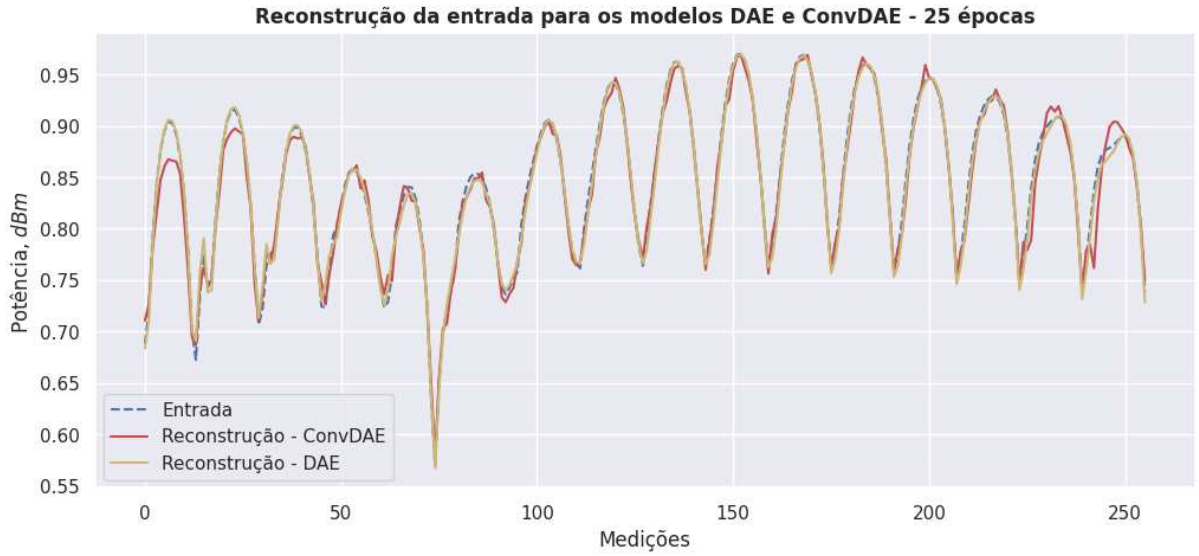
A comparação entre os *autoencoders* foi feita considerando três aspectos:

- Reconstrução de um padrão de feixe de um dispositivo autenticado;
- Reconstrução de um padrão de feixe de um dispositivo invasor;
- Valor do limiar de erro de reconstrução utilizado na identificação dos UEs.

Este último foi calculado com base no valor médio do erro das amostras de treinamento, adicionado ao desvio padrão desses erros. Qualquer valor acima desse limiar é suficiente para considerar um dispositivo como sendo invasor.

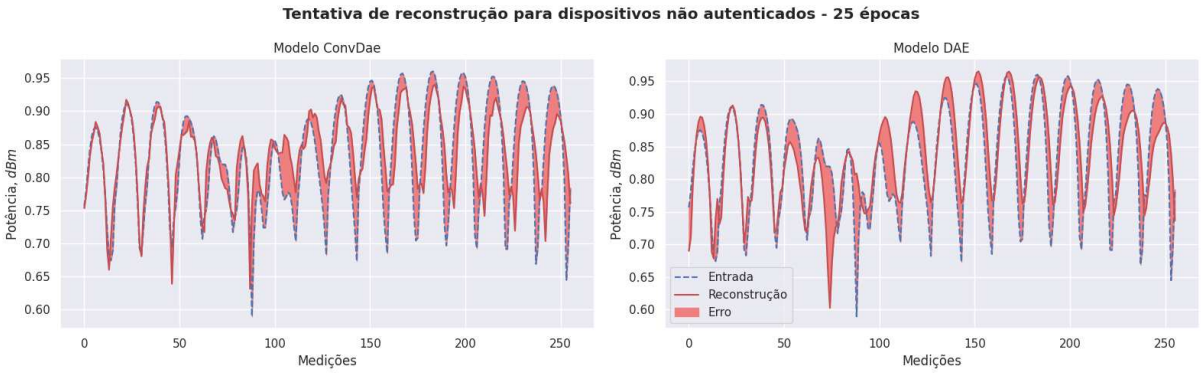
Desse modo, sobre a reconstrução dos padrões de feixe, na figura 17, é possível observar que ambos os DAEs conseguiram reconstruir de forma suficiente o dado de entrada, com o DAE convencional apresentando uma melhor reconstrução em comparação com o DAE convolucional. Além disso, na figura 18, é possível ver a tentativa de reconstrução de um padrão de feixe obtido de um invasor para os dois modelos, juntamente com o erro de reconstrução hachurado em vermelho.

Figura 17 – Reconstrução de um padrão de feixe autenticado - *StaticPosAllUEs_Dataset*



Fonte: Autoria própria

Figura 18 – Reconstrução de um padrão de feixe invasor - *StaticPosAllUEs_Dataset*

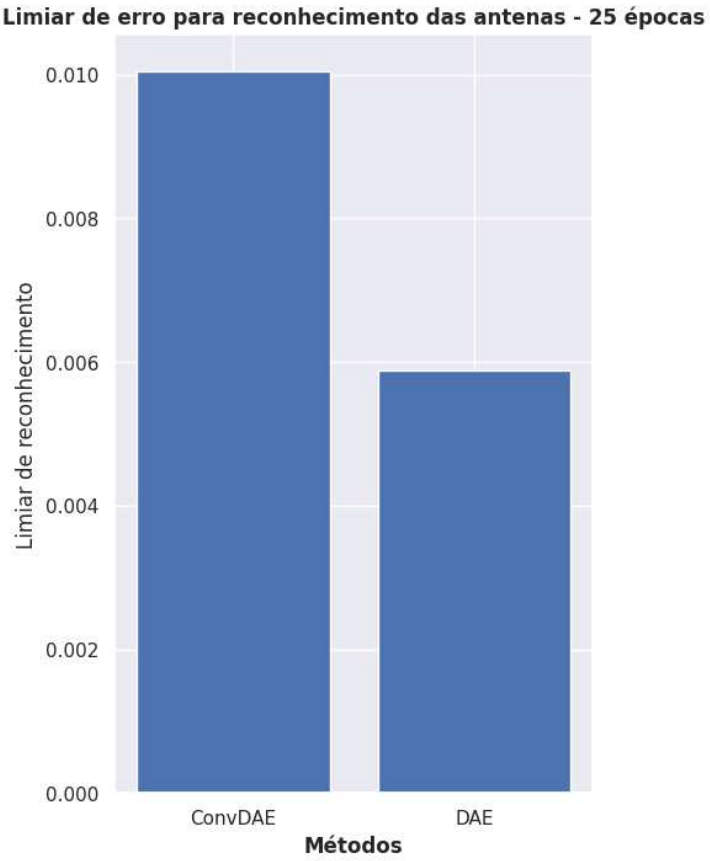


Fonte: Autoria própria

Além disso, considerando todos os dados de entrada utilizados durante o treinamento, o limiar de erro para cada DAE foi calculado. Na figura 19 são observados os dois valores para cada um dos DAEs. Nota-se que o limiar obtido para o ConvDAE foi maior, o que indica que o erro de reconstrução médio também foi maior, corroborando assim para a conclusão que nesse cenário, este modelo apresentou resultados inferiores ao convencional.

Essas métricas não podem ser utilizadas para avaliar o treinamento da CNN pois esta não tem como objetivo reconstruir o dado da entrada na saída da rede. Uma comparação entre todos os modelos treinados foi feita realizando a inferência com a parcela de testes do banco de dados. A figura 20 mostra esses valores. Nota-se aqui que a CNN apresentou valores unitários para todas as medidas, e isso se deve à facilidade de dinstinção entre os padrões gerados. É importante observar que para todos os modelos treinados, a especifidade foi unitária, ou seja, todos os dispositivos invasores foram detectados e não houve nenhum falso positivo.

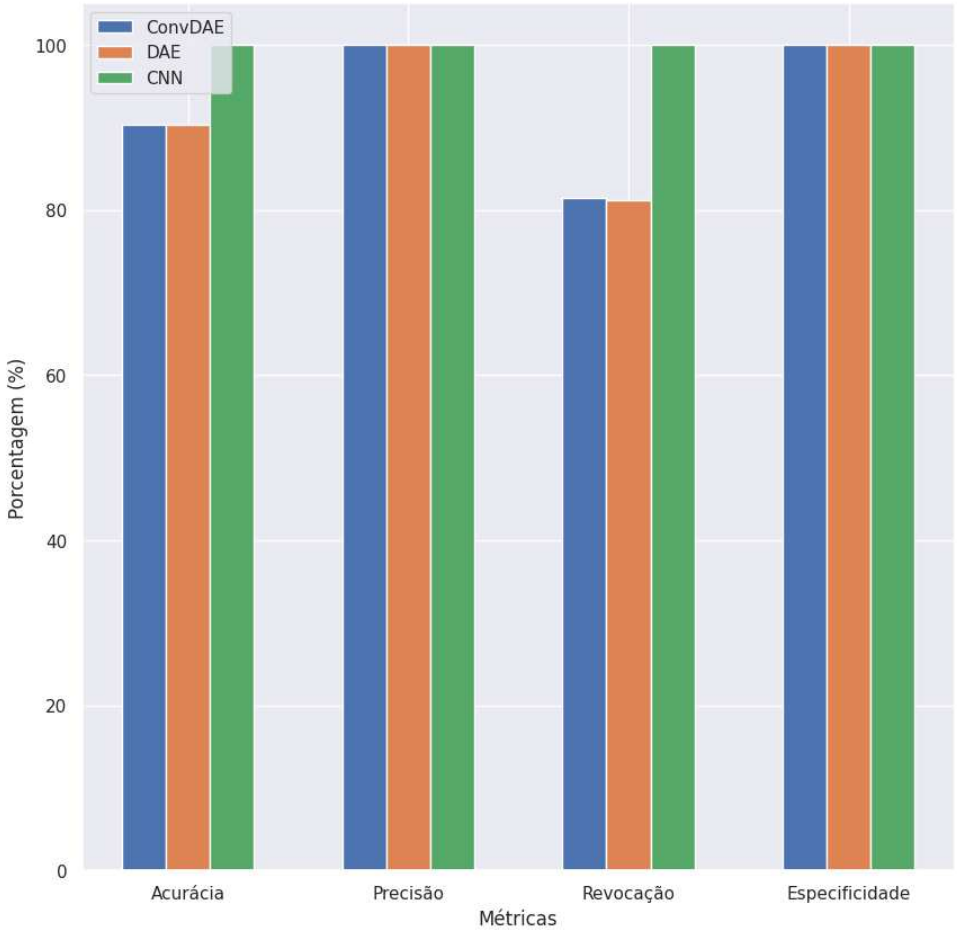
Figura 19 – Comparação entre os limiares de reconhecimento - *StaticPosAllUEs_Dataset*



Fonte: Autoria própria

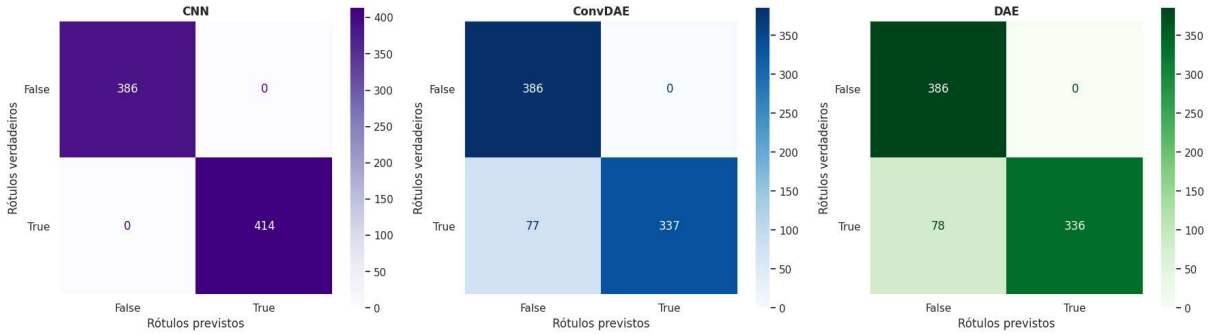
Por fim, a matriz de confusão para os três modelos pode ser vista na figura 21. No caso dos DAEs, foram previstos 77 e 78 falsos negativos para o ConvDAE e o DAE, respectivamente. Isso reflete uma revocação de cerca de 80%, o que para esse tipo de modelo é aceitável, visto que o mais importante para a segurança é bloquear o maior número de invasores.

Figura 20 – Métricas para os três modelos - *StaticPosAllUEs_Dataset*



Fonte: Autoria própria

Figura 21 – Matriz de confusão para os três modelos - *StaticPosAllUEs_Dataset*



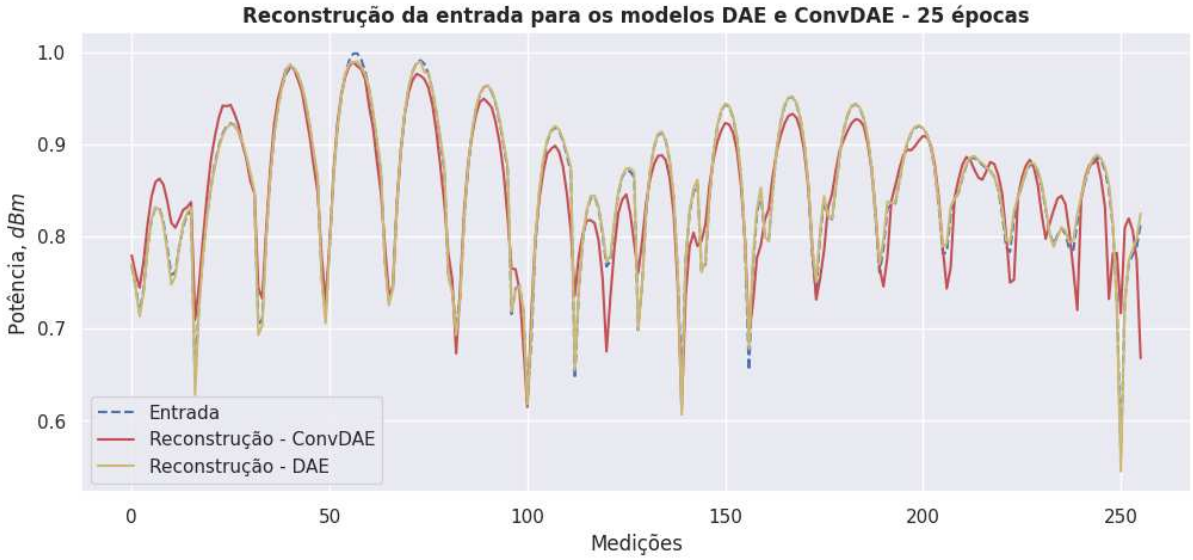
Fonte: Autoria própria

5.2.2 *StaticPosSingleUs_Dataset*

Para esse banco de dados, a CNN também apresentou melhores resultados para todas as métricas. A comparação da reconstrução dos dois DAEs pode ser vista na figura 22 e 23. Nesse caso, o DAE convencional novamente apresentou uma melhor reconstrução dos padrões de feixes de UEs autenticados. Isso mostra que para um conjunto de dados praticamente estático, a vantagem que o DAE convolucional traz em ser capaz de detectar

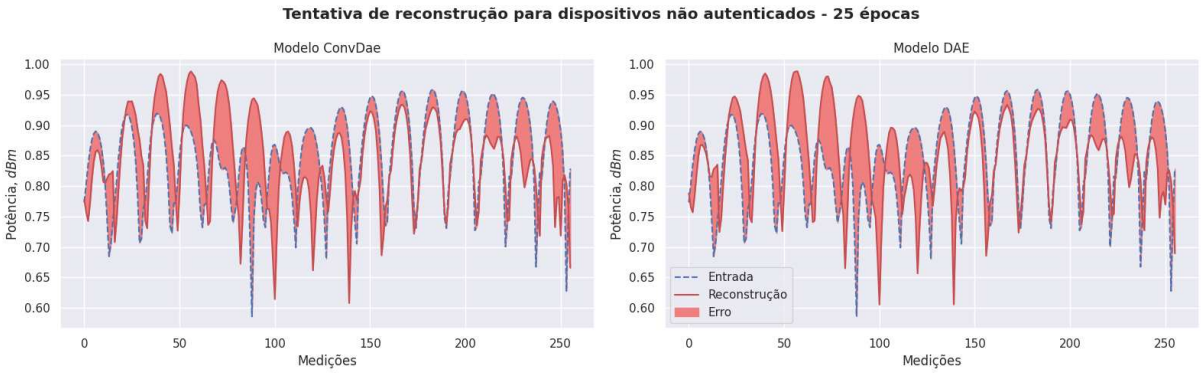
padrões na imagem deixa de ser necessária, pois para o mesmo número de épocas, o modelo convencional de *autoencoder* se mostrou ligeiramente melhor, ou quase semelhante, ao convolucional. Isso também pode ser visto na comparação entre os limiares, ilustrada na figura 24, na qual o DAE apresentou um valor menor em relação ao ConvDAE, assim como para o banco de dados anterior.

Figura 22 – Reconstrução de um padrão de feixe autenticado - *StaticPosSingleUE_Dataset*



Fonte: Autoria própria

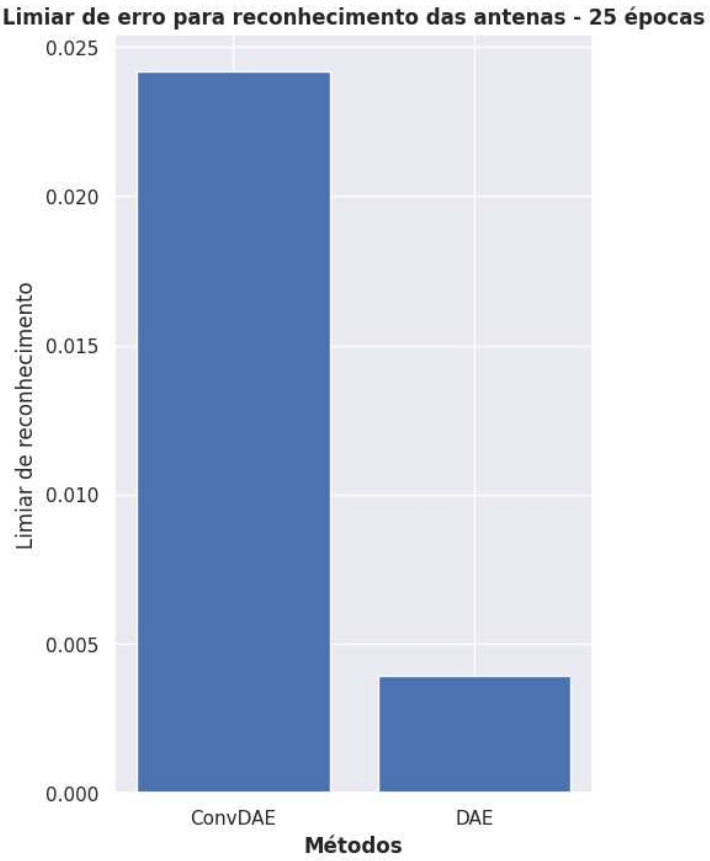
Figura 23 – Reconstrução de um padrão de feixe invasor - *StaticPosSingleUE_Dataset*



Fonte: Autoria própria

Ademais, sobre as quatro métricas relacionadas à inferência, a CNN ainda apresentou resultados melhores do que os demais modelos. É importante lembrar que em nenhum momento até então o número de épocas foi alterado, sendo todos os modelos treinados com as mesmas condições. Na figura 25 é possível ver que os resultados foram bem semelhantes ao obtido com o *StaticPosAllUEs_Dateset*.

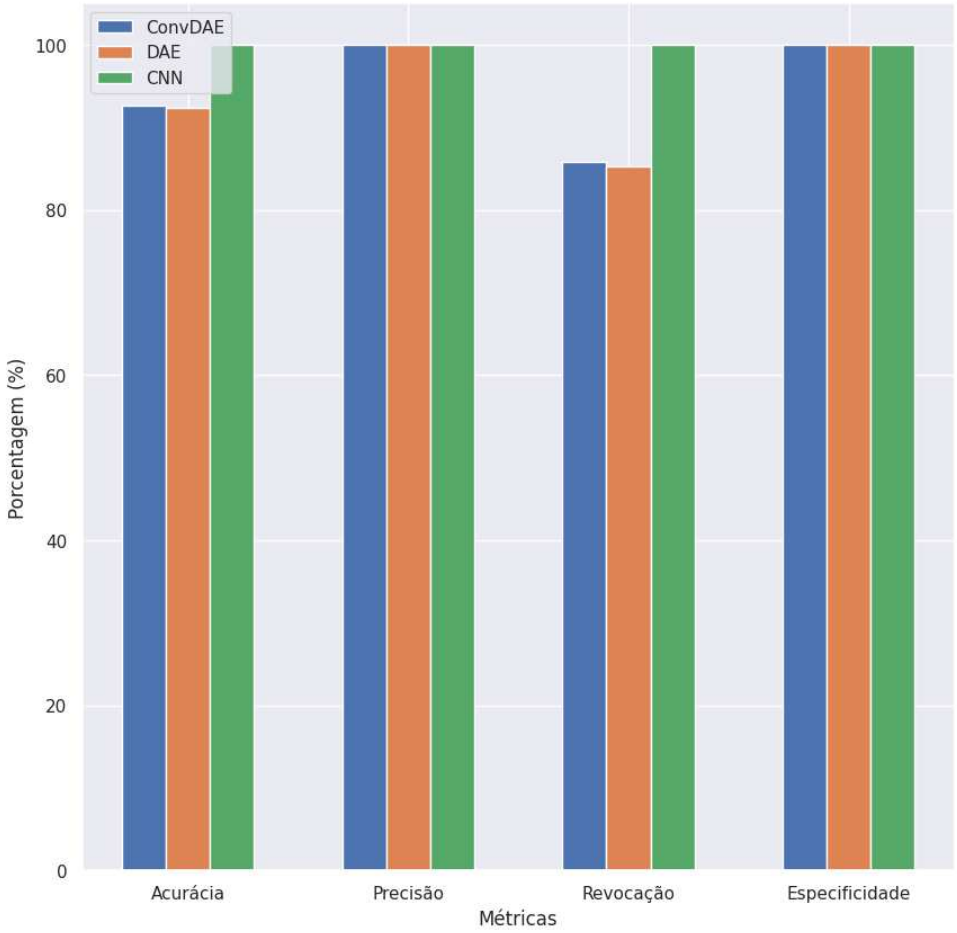
Figura 24 – Comparação entre os limiares de reconhecimento - *StaticPosSingleUE_Dataset*



Fonte: Autoria própria

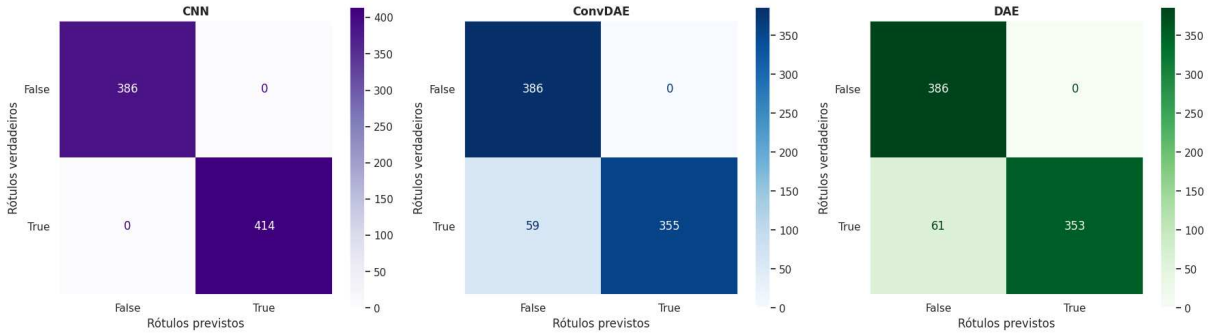
Por fim, a matriz de confusão para os três modelos pode ser vista na figura 26. Aqui os DAEs apresentaram valores ligeiramente melhores, sendo previstos 59 e 61 falsos negativos para o ConvDAE e o DAE, respectivamente, refletindo em uma revocação de cerca de 84%.

Figura 25 – Métricas para os três modelos - *StaticPosSingleUE_Dataset*



Fonte: Autoria própria

Figura 26 – Matriz de confusão para os três modelos - *StaticPosSingleUE_Dataset*



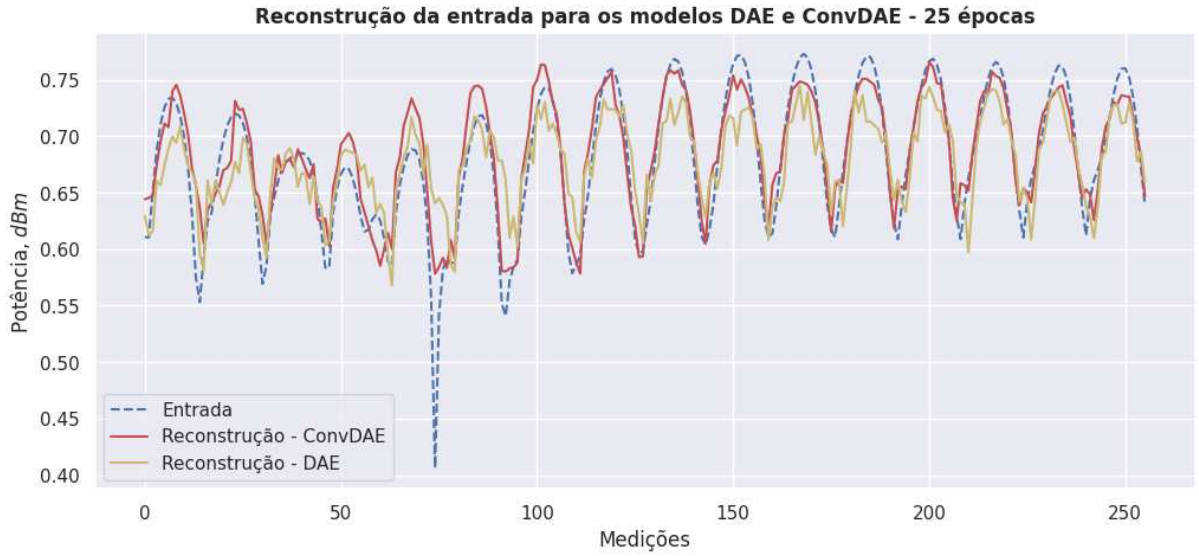
Fonte: Autoria própria

5.2.3 *DynamicPos_Dataset*

Para esse banco de dados, o treinamento se mostrou um pouco mais difícil. A CNN não conseguiu um resultado perfeito dentro das cinco épocas, e os resultados dos Deep Autoencoders não foram satisfatórios com 25 épocas. Nas figuras 27 e 28, é possível observar a tentativa de reconstrução dos padrões de feixe por parte dos dispositivos. O limiar também foi maior nos dois casos, porém nesse cenário o ConvDAE teve um resultado

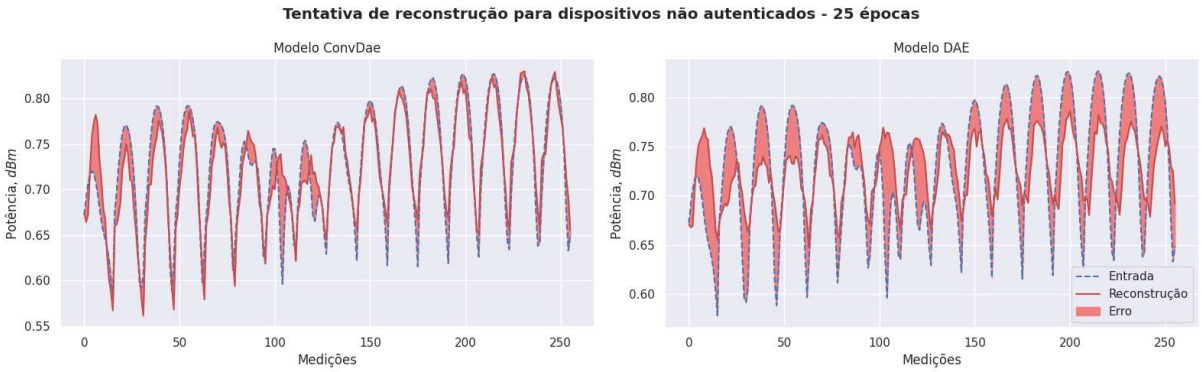
melhor do que o DAE, como mostrado na figura 30. Além disso, na figura 29 as métricas são mostradas com valores inferiores aos testes feitos com os bancos de dados anteriores. Por fim, a matriz de confusão da figura 31 mostra como o número de falsos positivos foi alto, tornando ineficaz a detecção dos invasores para as mesmas configurações de treinamento utilizadas até então.

Figura 27 – Reconstrução de um padrão de feixe autenticado - *DynamicPos_Dataset*



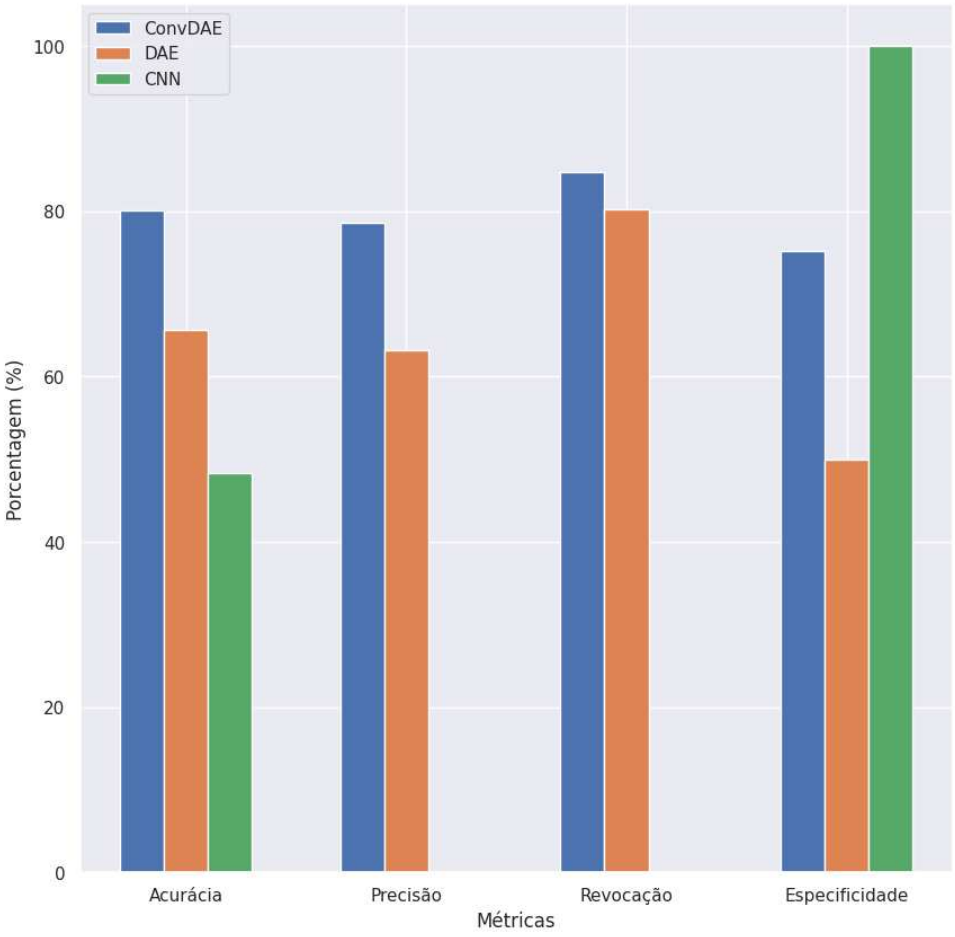
Fonte: Autoria própria

Figura 28 – Reconstrução de um padrão de feixe invasor - *DynamicPos_Dataset*



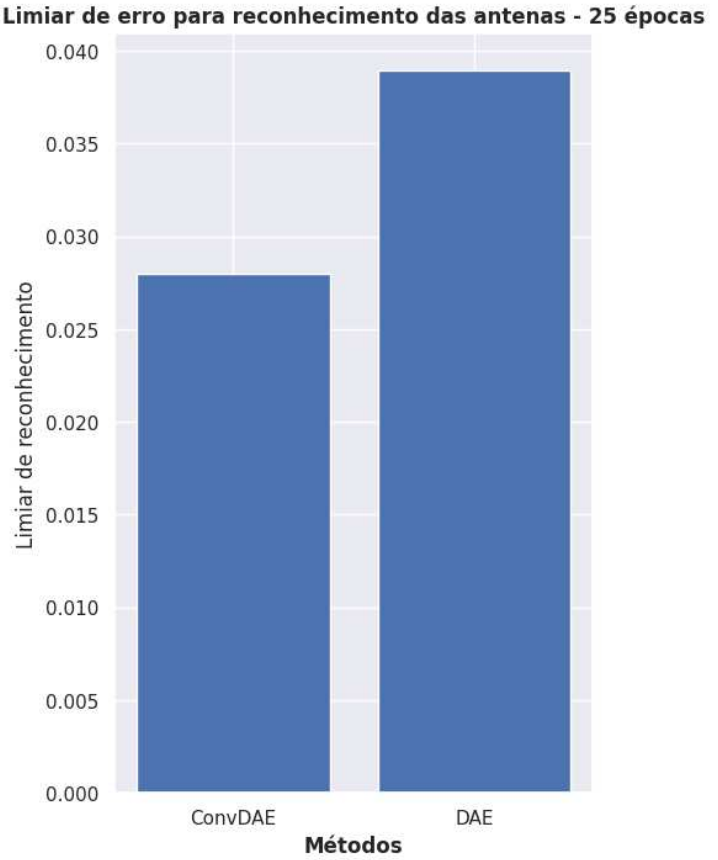
Fonte: Autoria própria

Figura 29 – Métricas para os três modelos - *DynamicPos_Dataset*



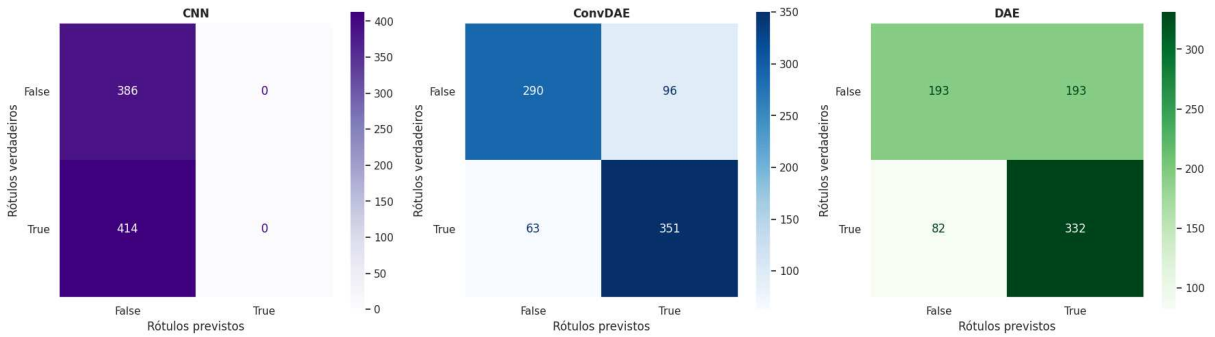
Fonte: Autoria própria

Figura 30 – Comparação entre os limiares de reconhecimento - *Dynamic_Dataset*



Fonte: Autoria própria

Figura 31 – Matriz de confusão para os três modelos - *StaticPosSingleUE_Dataset*



Fonte: Autoria própria

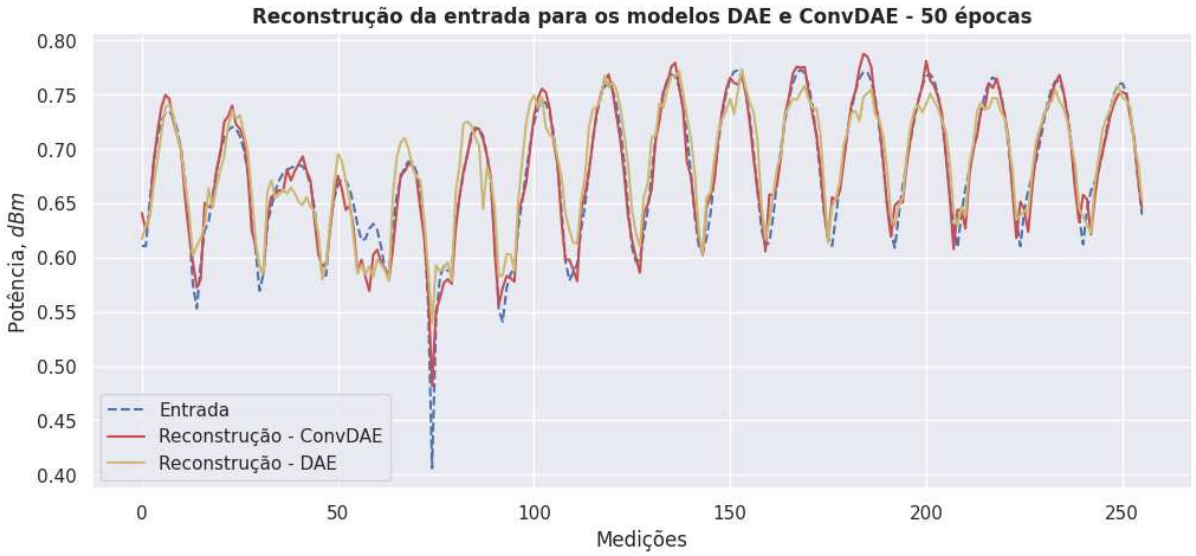
5.2.3.1 Melhorias no treinamento

Desse modo, a fim de melhorar os resultados, a CNN foi treinada novamente com 10 épocas, e os modelos DAE foram treinados com 50 épocas. Isso provocou uma melhoria considerável no reconhecimento por parte de todos os métodos, apresentando resultados semelhantes aos obtidos com os dois primeiros bancos de dados. A reconstrução de padrões de dispositivos autenticados para esse novo treinamento pode ser vista na figura 32, enquanto a tentativa de reconstrução de padrões invasores pode ser vista na figura 33. Além disso, a comparação dos limiares de reconhecimento é mostrada na figura 35. Por fim, as métricas podem ser observadas na figura 34, e a matriz de confusão na figura 36.

É importante notar que nesse cenário o ConvDAE apresentou resultados melhores em relação ao DAE, mostrando que com mais variações nas medições para um mesmo dispositivo, a convolução do ConvDAE se mostra mais eficaz, apresentando um resultado oposto ao obtido para os dois bancos de dados anteriores. Sendo assim, o número de falsos negativos foi de 68 e 75 para o DAE convolucional e para o convencional, respectivamente. Outro detalhe é que o DAE não apresentou uma especificidade de 100%, com um total de 22 falsos positivos.

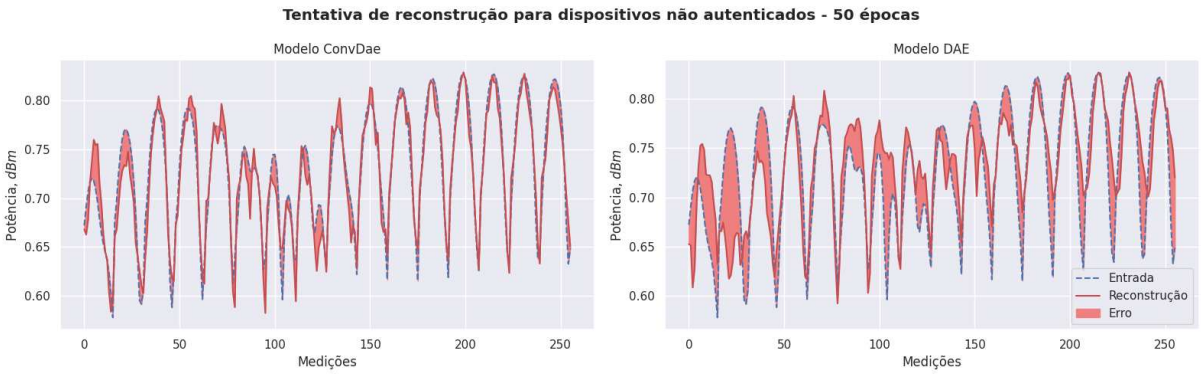
Diante dos resultados obtidos, verifica-se que a rede neural convolucional foi a abordagem mais eficiente em todos os cenários, necessitando de menos épocas para o treinamento e apresentando resultados melhores em comparação com todos os outros modelos. Porém, visto a necessidade de inserir no banco de dados padrões de feixe de arranjos não autenticados, essa abordagem deixa de ser tão vantajosa pois seria necessário um esforço computacional extra para a geração desses padrões “invasores” durante o treinamento. Além disso, caso novos UEs sejam conectados à rede, a CNN receberá novos dados com um único rótulo para o treinamento, o que pode causar um enviesamento do modelo devido ao desbalanceamento entre dados rotulados como autenticados e não autenticados durante o treinamento. Os DAEs, por sua vez, apresentam como rótulos os próprios padrões de feixe, não correndo o risco de viés por desbalanceamento de rótulos. Uma vez que um novo UE é conectado, a rede poderia ser treinada com esses novos dados de foram on-line e sem prejuízo ao conhecimento prévio. Entretanto, a visão desse problema como um sistema de autenticação não é o foco deste trabalho, e sim apenas a demonstração da possibilidade de identificação de invasores. Logo, as abordagens e decisões para caso algum UE novo seja conectado, ou algum antigo seja desconectado, não foram levadas em consideração durante o desenvolvimento.

Figura 32 – Reconstrução de um padrão de feixe autenticado - *DynamicPos_Dataset* - Retreinamento



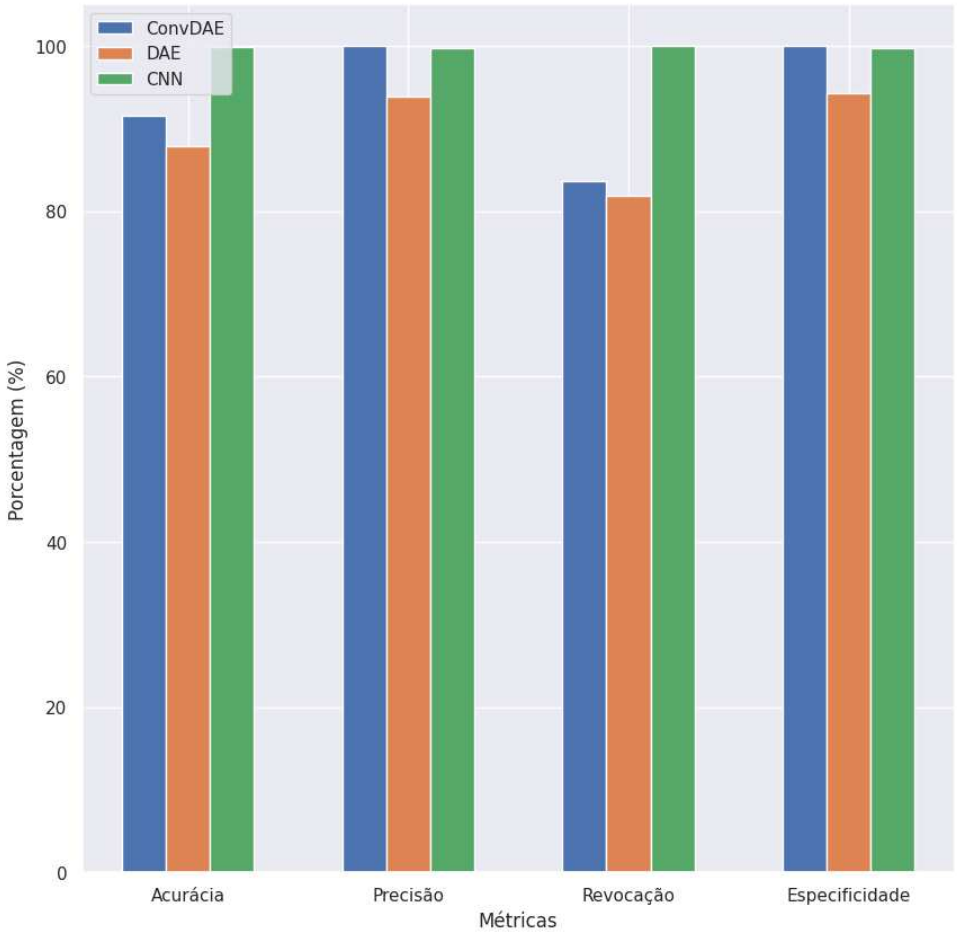
Fonte: Autoria própria

Figura 33 – Reconstrução de um padrão de feixe invasor - *DynamicPos_Dataset* - Retreinamento



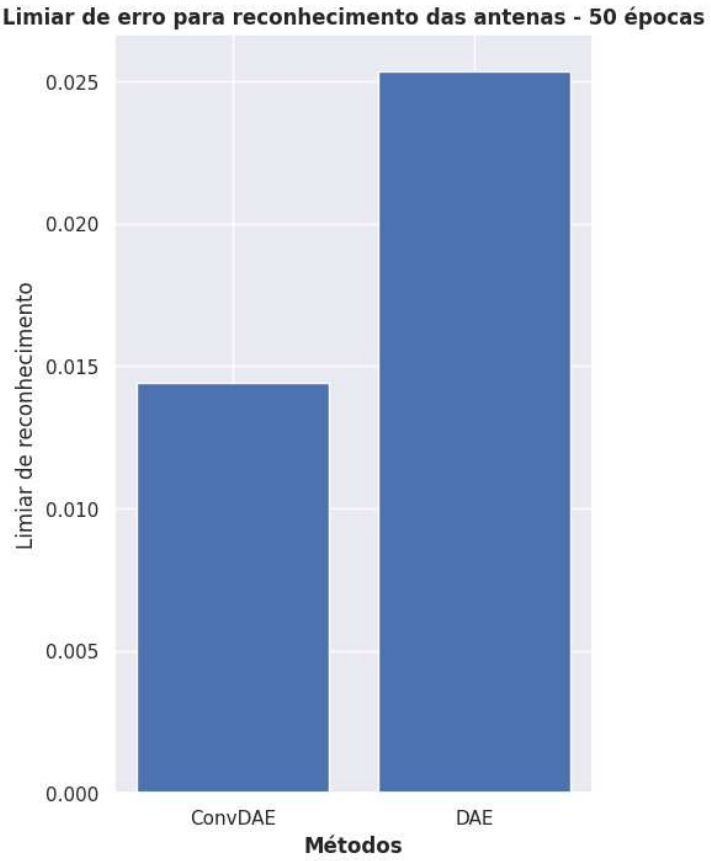
Fonte: Autoria própria

Figura 34 – Métricas para os três modelos - *DynamicPos_Dataset* - Retreinamento



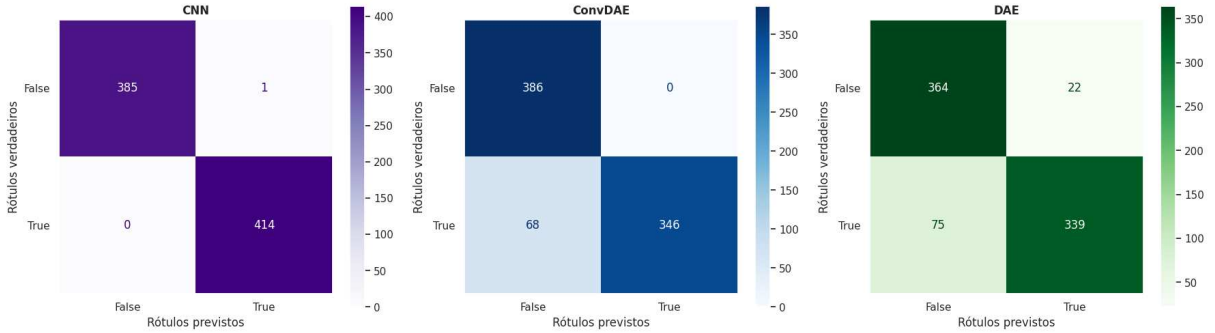
Fonte: Autoria própria

Figura 35 – Comparação entre os limiares de reconhecimento - *Dynamic_Dataset* - Retreinamento



Fonte: Autoria própria

Figura 36 – Matriz de confusão para os três modelos - *StaticPosSingleUE_Dataset* - Retreinamento



Fonte: Autoria própria

6 Conclusão

Diante de tudo que foi exposto, este trabalho surgiu a partir de uma oportunidade oferecida pelo CNPq, a Chamada CNPq/MCTI/SEMPI Nº 56/2022 - Apoio para Estudando Elaborando TCC em Inteligência Artificial. Assim, primeiramente foi feita uma revisão sistemática considerando um cenário mais geral do uso de inteligência artificial para a proteção de redes 5G. Em seguida, finalizada a revisão, surgiu uma motivação para que um trabalho fosse feito em um cenário mais específico, com foco na camada física, explorando características de rádio como o padrão de feixe de arranjos de antenas.

Com isso, após todas as simulações, foi possível notar que a inteligência artificial é uma ferramenta poderosa para o auxílio da cibersegurança em redes 5G. Todos os modelos construídos apresentaram uma arquitetura pequena, sem um alto nível de complexidade, além de serem treinados utilizando poucas épocas. Ainda assim, apresentaram resultados satisfatórios, com os três modelos identificando com sucesso todos os dispositivos invasores. Entretanto, alguns padrões gerados por UEs autenticadas foram identificadas como invasores, mas em uma rede neural que tem como objetivo detectar anomalias, é mais importante que estas sejam devidamente identificadas, e não que as entradas consideradas válidas sejam identificadas como tais.

Além disso, conclui-se que esse cenário ainda é novo e há muito a ser explorado, uma vez que redes 5G ainda estão sendo implementadas mundialmente, e o campo de inteligência artificial nunca cresceu tanto como está crescendo atualmente. Desse modo, métodos cada vez mais eficientes podem ser investigados e aplicados para a proteção de uma rede 5G, e cada vez mais será possível utilizar de modelos mais densos e robustos aproveitando-se de poder de processamento em nuvem e da baixa latência oferecida pela tecnologia 5G.

Sendo assim, é possível afirmar após o desenvolvimento deste trabalho que a proteção de camada física precisa ser mais investigada e pesquisada pela comunidade acadêmica, a fim de que melhores soluções surjam e sejam aplicadas na indústria. Existem muitas pesquisas voltadas a proteção da rede pela camada de aplicação, mas ataques pela camada física, principalmente oriundos de ondas de rádio, ainda são pouco explorados na literatura.

Por fim, diante dos resultados alcançados, é possível elencar os seguintes pontos de melhorias no processo de identificação:

- Geração do banco de dados utilizando um cenário mais próximo do real, com o UE transmitindo sinais SRS e a gNodeB executando as medições;

- Geração de bancos de dados maiores e mais diversificados para um cenário mais próximo da realidade, no qual os dispositivos podem ocupar diversas posições;
- Obtenção das potências recebidas a partir de medições físicas, com o objetivo de obter os padrões de feixe de dispositivos reais e não simulados;
- Melhoria no processo de treinamento utilizando CNN, deixando-o independente de padrões de feixe invasores para o treinamento;
- Mecanismo de inserção de dispositivos novos na rede, de modo que o modelo aprenda os novos padrões sem esquecer os antigos e sem necessariamente ter que aprender novamente todos os padrões que já reconhecia.

Referências

, Wireless Networking and Communications Group. *WNCG Prof. Robert Heath on Millimeter Wave MIMO Communication*. 2014. Https://www.youtube.com/watch?v=BQ45FuGpFQ0. Citado na página 19.

3GPP. *Study on New Radio (NR) access technology* . [S.l.], 2018. Version 15.0.0. Citado na página 21.

3GPP. *NR Physical Layer Measurements*. [S.l.], 2020. Version 16.2.0. Citado na página 22.

AHMED, Y.; TAUFIQ, A.; ARAFATUR, R. M. A cyber kill chain approach for detecting advanced persistent threats. *Computers, Materials and Continua*, Tech Science Press, v. 67, n. 2, p. 2497–2513, 2021. Citado na página 18.

BALAKRISHNAN, S. et al. Physical layer identification based on spatial–temporal beam features for millimeter-wave wireless networks. *IEEE transactions on information forensics and security*, IEEE, v. 15, p. 1831–1845, 2019. Citado na página 18.

BALANIS, C. A. *Antenna theory: analysis and design*. [S.l.]: John wiley & sons, 2016. Citado na página 20.

CATAK, F. O. et al. Defensive distillation-based adversarial attack mitigation method for channel estimation using deep learning models in next-generation wireless networks. *IEEE Access*, IEEE, v. 10, p. 98191–98203, 2022. Citado na página 18.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. <http://www.deeplearningbook.org>. Citado 4 vezes nas páginas 25, 26, 27 e 28.

HINTON, G. E.; SALAKHUTDINOV, R. R. Reducing the dimensionality of data with neural networks. *science*, American Association for the Advancement of Science, v. 313, n. 5786, p. 504–507, 2006. Citado na página 28.

KIM, J. S. et al. Multi-stream cnn-based personal recognition method using surface electromyogram for 5g security. *CMC-COMPUTERS MATERIALS & CONTINUA*, TECH SCIENCE PRESS 871 CORONADO CENTER DR, SUTE 200, HENDERSON, NV 89052 USA, v. 72, n. 2, p. 2997–3007, 2022. Citado na página 18.

KOTTKAMP, M. et al. *5G New Radio: Fundamentals, procedures, testing aspects*. [S.l.]: Rohde & Schwarz GmbH & Company KG, 2019. Citado na página 34.

LI, Y.-N. R. et al. Beam management in millimeter-wave communications for 5g and beyond. *IEEE Access*, IEEE, v. 8, p. 13282–13293, 2020. Citado na página 21.

MCCULLOCH, W. S.; PITTS, W. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, Springer, v. 5, p. 115–133, 1943. Citado na página 23.

NOSOUHI, M. R. et al. Towards spoofing resistant next generation iot networks. *IEEE Transactions on Information Forensics and Security*, IEEE, v. 17, p. 1669–1683, 2022. Citado na página 18.

NR Downlink Transmit-End Beam Refinement Using CSI-RS - MATLAB Simulink. Https://www.mathworks.com/help/5g/ug/nr-downlink-transmit-end-beam-refinement-using-csi-rs.html. Citado na página 33.

RODRÍGUEZ, E. et al. Transfer-learning-based intrusion detection framework in iot networks. *Sensors*, MDPI, v. 22, n. 15, p. 5621, 2022. Citado na página 18.

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 23.

ZHANG, A. et al. Dive into deep learning. *arXiv preprint arXiv:2106.11342*, 2021. Citado na página 26.