UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

"Beam-selection" Otimizado por Aprendizado de Máquina: Uma Abordagem Multimodal

Jamelly Freitas Ferreira

DM: 37/2023

UFPA / ITEC / PPGEE
Campus Universitário do Guamá
Belém-Pará-Brasil
2023

UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Jamelly Freitas Ferreira

"Beam-selection" Otimizado por Aprendizado de Máquina: Uma Abordagem Multimodal

DM: 37/2023

UFPA / ITEC / PPGEE
Campus Universitário do Guamá
Belém-Pará-Brasil
2023

UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

Jamelly Freitas Ferreira

"Beam-selection" Otimizado por Aprendizado de Máquina: Uma Abordagem Multimodal

Dissertação submetida para o comitê de avaliação do departamento de pós-graduação da Universidade Federal do Pará para obtenção do título de Mestre em Engenharia Elétrica com foco em Telecomunicacões.

UFPA / ITEC / PPGEE
Campus Universitário do Guamá
Belém-Pará-Brasil
2023

UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

"BEAM-SELECTION OTIMIZADO POR APRENDIZADO DE MÁQUINA: UMA ABORDAGEM MULTIMODAL"

AUTOR: **Jamelly Freitas Ferreira**

DISSERTAÇÃO DE MESTRADO SUBMETIDA À BANCA EXAMINADORA APROVADA PELO COLEGIADO DO PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA, SENDO JULGADA ADEQUADA PARA A OBTENÇÃO DO GRAU DE MESTRE EM ENGENHARIA ELÉTRICA NA ÁREA DE TELECOMUNICAÇÕES.

APROVADA EM: 30/10/2023

BANCA EXAMINADORA:

gov.br Documento assinado digitalmente
ALDEBARO BARRETO DA ROCHA KLAUTAU JUNI
Data: 15/12/2023 14:43:02-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. ALDEBARO BARRETO DA ROCHA KLAUTAU JUNIOR
(Orientador – PPGEE/UFPA)

gov.br Documento assinado digitalmente
LEONARDO LIRA RAMALHO
Data: 16/12/2023 12:25:46-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. LEONARDO LIRA RAMALHO
(Avaliador Interno – PPGEE/UFPA)

gov.br Documento assinado digitalmente
ILAN SOUSA CORREA
Data: 21/11/2023 17:38:52-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. ILAN SOUSA CORREA
(Avaliador Externo ao Programa – UFPA)

DIEGO DE AZEVEDO GOMES:83983023215
Assinado de forma digital por DIEGO DE AZEVEDO GOMES:83983023215
Dados: 2023.11.21 14:51:36 -03'00'

Prof. Dr. DIEGO DE AZEVEDO GOMES

VISTO:

gov.br Documento assinado digitalmente
Diego Lisboa Cardoso
Data: 05/12/2023 16:07:40-0300
Verifique em https://validar.iti.gov.br

Prof. Dr. Diego Lisboa Cardoso
(Coordenador do PPGEE/ITEC/UFPA)

Agradecimentos

Agradeço primeiramente aos meus, respectivamente, orientador e co-orientador Prof.Dr Aldebaro Barreto da Rocha Klautau Júnior e Prof.Dr.Diego de Azevedo Gomes, cujo apoio e suporte técnico e humano foram fundamentais para a conclusão deste trabalho. Aproveito também para prestar os meus sinceros agradecimentos à banca examinadora deste trabalho.

Agradeço à minha família, especialmente meus pais Jozinaldo e Alba, que lutaram de maneira incessante para que eu e meus irmãos pudéssemos receber a melhor educação formal possível. À minha irmã Marie Clayre e meu irmão Yego agreço pelo companheirismo e suporte que sempre me foi oferecido. Sem vocês a vida não teria sido tão espetacular quanto tem sido.

Não poderia deixar de agradecer também aos amigos e colegas de trabalho que tive o prazer de cultivar ao longo da vida. Não poderei citar todos aqui, entretanto, deixo meu agradecimento especial para Luciano Gouvêa, Felipe Bastos e Cleverson Nahum, que me inspiram como ser humano e engenheiro.

Por fim, porém não menos importante, agradeço a minha amada namorada Sandy Ianka que sempre torceu por, e comigo celebrou, cada pequena vitória.

Jamelly Ferreira

Outubro, 2023

Não me envergonho de mudar de opinião, porque não me envergonho de pensar.

Blaise Pascal

Lista de Acrônimos

AI Artificial Intelligence

SNR Signal to Noise Ratio

BS Beam-Selection

mmWaves Millimeter Waves

MIMO Multiple-input Multiple-output

CSV Comma-separated values

LIDAR Light Detection and Ranging

ReLU Rectified Linear Unit

CNN Convolutional Neural Network

ML Machine Learning

DL Deep Learning

GPS Global Positioning System

EUA Estados Unidos da América

RT Ray Tracing

Lista de Figuras

Lista de Tabelas

Sumário

Abstract

This dissertation aims to investigate the use of machine learning models using multimodal data as input to optimize the Beam-Selection process in millimeter-wave based networks. The use of Deep Learning has intensified in different areas, and it is possible to obtaing performance equal or superior to human performance, so its use is also promising in wireless communication scenarios. This work used data from different sources, which proved to be convenient since it is possible to adjust the model according to the quality/availability of this data. After executing the experiments and obtaining the results, it was observed that it is possible to obtain significant performance in different metrics even with simpler data such as image and coordinate.

***Keywords* —** Machine Learning, Neural Networks, Deep Learning, Millimeter Waves, Beam Selection

Resumo

Esta dissertação tem como objetivo investigar a utilização de modelos de aprendizado de máquina usando dados multimodais como entrada para otimizar o processo de "Beam-Selection"em redes baseadas em ondas milimétricas. O uso de Deep Learning tem se intensificado em diferentes áreas, sendo possível obter performance igual ou superior à humana, desta forma seu uso mostra-se promissor também em cenários de comunicação sem fio. Neste trabalho foram usados dados de diferentes naturezas o que se mostou conveniente ao passo que é possível ajustar o modelo de acordo com a qualidade/disponibilidade destes dados. Após execução dos experimentos, e obtenção dos resultados, foi observado que é possível obter significativa performance em diferentes métricas, mesmo com dados mais simples como Imagem e Coordenada.

***Palavras-chave* —** Aprendizado de máquina, Redes Neurais, Aprendizagem Profunda, Ondas Milimétricas, Beam Selection

Capítulo 1

Introdução

O uso das redes móveis foi e tem sido um dos principais propulsores da criação e adaptação de serviços aos meios digitais. Na atualidade, é praticamente impossível não ter contato com um *smartphone* e as facilidades que o mesmo possibilita, principalmente aquelas relacionadas à internet. Em vista disso, é uma tendência esperada que as redes móveis precisem evoluir para suportar a demanda e possibilitar conexões cada vez mais rápidas [5].

A quinta geração de redes móveis (5G) é dependente de uma série de tecnologias e técnicas que são necessárias para garantir que características como baixa latência e alta disponibilidade sejam possíveis. O uso de ondas de frequência milimétrica e grande número de antenas , chamado de Massive Multiple-input Multiple-output (MIMO) [6], são alguns exemplos[7]. A aplicação de tais técnicas e tecnologias, entretanto, implica em alguns ônus tais quais a maior tendência ao desvanecimento de sinal em ondas milimétricas, sobretudo em situações de propagação em espaço livre ou em condições de chuva [8]. Desta forma, processos como o "Beam Selection" se fazem necessários para viabilizar a qualidade de serviço.

No tocante ao processo do "Beam Selection" é importante ressaltar que existem diversas formas de realizá-lo, por exemplo, é possível realizar a escolha do par de beams ótimo através de uma busca exaustiva por todas as possíveis combinações em um cenário de alta Signal to Noise Ratio (SNR), entretanto, isso levaria a latência e sobrecarga proibitivas [9]. Dessa forma, uma estratégia comumente proposta é o uso de técnicas de Machine Learning (ML), especialmente o Deep Learning (DL) de modo a aprimorar esse processo.

O DL é um dos tópicos mais discutidos atualmente na academia. Seu extenso leque de aplicações, que vai desde a computação visual até a sintetização de voz, encorajam a sua aplicação nos mais diferentes cenários, o que inclui as telecomunicações. Um dos pontos po-

sitivos do DL é o fato de que quanto maior a disponibilidade de dados de treinamento maior é a sua performance, o que se contrapõe à outras técnicas de ML que tendem a atingir um platô de performance mesmo que a disponibilidade de dados de treinamento se torne alta [10]. Apesar desta vantagem, um dos fatores que acabam por desencorajar o uso de DL em cenários de telecomunicações é que a obtenção de dados de treinamento pode se tornar inviável, uma vez que obter dados de campo pode se tornar proibitivamente caro.

Em vista disso, este trabalho propõe uma arquitetura que utiliza diferentes técnicas de DL em um conjunto de dados obtidos através de simulações, tais dados conseguem ser realistas o suficiente a ponto de podermos simular e gerar resultados em um contexto de V2X usando dados simulados que são compostos por informações de: Imagem, LIDAR, Global Positioning System (GPS) e informações do canal de comunicação. O cenário utilizado foi baseado nas cidade de Rosslyn (Estados Unidos da América (EUA)).

1.1 Trabalhos Relacionados

A popularização de técnicas de DL, aliada à maior disponibilidade de hardware, propiciou o surgimento de diversas propostas que tentam resolver um problema parecido ao que é proposto neste trabalho, onde a principal diferença entre estas reside no tipo de dados utilizados e o pré-processamento aos quais estes são submetidos. Alguns exemplos de tais propostas são listados na Tabela 1.1

Em [9] o autor utiliza apenas dados espaciais, nominalmente as coordenadas do receptor e o ângulo que o este forma em relação à um eixo de referência. Esta abordagem é interessante se analisarmos por uma ótica que privilegia o uso de poucos recursos, entretanto, também pode-se dizer que esta abordagem não leva em conta a oferta cada vez maior de sensores nos dispositivos. Já em [11] a estratégia adotada faz uso de dados de imagem, além de realizar uma predição de bloqueio. O uso de dados de imagem e coordenadas é ignorado em [12] onde o problema de Beam-Selection (BS) é convertido em um problema de reconstrução de imagem através unicamente dos dados do canal de comunicação, o que torna esta abordagem interessante em cenários com baixa variadade de dados sensoriais.

Mesmo apresentando muitas inovações e técnicas rebuscadas, os trabalhos acima não levam em consideração a, anteriormente citada, cada vez maior disponibilidade de sensores nos receptores. Este trabalho não só tenta explorar cada qualidade dos diferentes sensores como

também busca usar a união dos dados de modo a promover melhores resultados.

Tabela 1.1: Trabalhos relacionados a esta dissertação de mestrado.

Trabalho	Dados utilizados
[9]	Apenas dados espaciais: coordenadas do receptor e o ângulo que este forma em relação ao eixo de referência
[11]	Apenas dados de imagem
[12]	Não utiliza dados sensoriais, apenas os dados do canal são utilizados em um problema de reconstrução de imagem

1.2 Contribuições

As contribuições desta dissertação são as seguintes

- Apresenta uma proposta que reduz consideravelmente o custo financeiro para se treinar, utilizando técnicas de ML, o processo de ”Beam Selection”em redes móveis. Uma vez que não se faz necessária a coleta de dados em campo.
- O trabalho apresenta resultados promissores no cenário em que se usam apenas dados de imagem, o que é algo interessante do ponto de vista financeiro visto que sensores de imagem são muito mais baratos que, por exemplo, alternativas caras como o LIDAR.
- O processo de treinamento da rede é extremamente flexível, podendo-se variar entre diferentes tipos de dados de entrada.

1.3 Organização

Essa dissertação está organizada da seguinte forma. O Capítulo 2 apresenta os conceitos fundamentais para a compreensão deste trabalho e os elementos que o compõem. O Capítulo 3 apresenta os resultados obtidos em diferentes cenários utilizando diferentes conjuntos de dados, mostrando a flexibilidade da proposta e sua eficácia nas situações propostas. Por fim, o Capítulo 4 expõe as possíveis melhorias, próximos trabalhos e conclui esta dissertação.

1.4 Produção

Durante o desenvolvimento dos trabalhos que culminaram nesta dissertação o paper a seguir foi escrito e apresentado na *X Conferência Nacional em Comunicações, Redes e Segurança da Informação* (ENCOM 2020)

Papers de conferência:

1. **Jamelly Ferreira**, Diego Gomes and Aldebaro Klautau. *A New Multimodal Approach to Beam-Selection Based on Deep Learning*. X Conferência Nacional em Comunicações, Redes e Segurança da Informação (ENCOM 2020).

Capítulo 2

Metodologia

Neste capítulo serão detalhados o problema e os componentes do cerne deste trabalho. Na Seção 2.1 serão explicados os conceitos de AI que sedimentam este trabalho. Na Seção 2.2 será descrito o que é o processo de Beam Selection e o porquê da necessidade de técnicas que otimizem este processo. Na Seção 2.3 será descrito o conjunto de dados que será utilizado para treinamento e validação, além disso, serão descritos também os pré-processamentos ao qual cada categoria de dados será submetida. Por fim, na Seção 2.4 serão apresentados alguns conceitos básicos e também o modelo de aprendizado de máquina proposto por este trabalho.

2.1 Conceitos Básicos de AI

Artificial Inteligence (AI), em tradução direta Inteligência Artificial, é um termo que deixou de figurar entre soluções de tecnologia que revolucionariam o futuro próximo para estar entre as tecnologias que se tornam indispensáveis no presente. O ChatGPT [13], que tem sido constantemente citado em grandes veículos de mídia (mesmo aqueles sem foco em tecnologia), é um notável exemplo de como tecnologias de AI já estão inseridas na sociedade moderna. É importante, entretanto, pontuar que o termo AI é demasiadamente genérico pois não se trata de uma única tecnologia mas sim um conjunto destas que viabilizam as diversas facilidades que utilizamos cotidianamente como por exemplo: reconhecimento de placas de carro, reconhecimento facial e reconhecimento de voz. Sendo assim, com o desenvolvimento tecnológico, alguns campos de estudo foram se desenvolvendo através da definição de AI, que segundo [1] consiste no ”esforço em automatizar tarefas intelectuais normalmente realizadas por humanos”.

Dos campos de estudo englobados pela premissa de AI, dois se destacam:

- ML: “Machine Learning” do Inglês *Aprendizado de Máquina*
- DL: “Deep Learning” do Inglês *Aprendizagem profunda*

A relação entre estes diferentes campos de estudo estão ilustradas na figura 2.1. ML, de acordo com [2] pode ser definido como: “A ciência (e arte) de programar computadores de modo que estes *aprendam* a partir de dados”. É importante aqui destacar que o conceito de *aprendizado* está relacionado a capacidade de adaptação a novos dados. É importante tal destaque para que seja feita a diferenciação entre o real aprendizado de máquina e os algorítmos pertencentes a classe de *Inteligência Artificial Simbólica*, pois estes nada mais eram que regras pré-definidas para uma determinada tarefa que não eram capazes de adaptar-se em cenários diferentes das quais foram concebidas, a Figura 2.3 ilustra este conceito. Como pode-se notar, o ponto de partida é o estudo do problema, a partir dai, são levantadas as regras que permitem solucionar o problema e parte-se então para avaliação do resultado que, caso seja satisfatório, é colocado para teste em produção, caso contrário, são analizados os erros e o processo reinicia. Esta abordagem possui problemas inerentes, o primeiro é a sua baixa flexibilidade, que pode levar a desnecessária complexidade, por exemplo, caso o problema seja a conversão de voz humana para texto a simples concepção das regras já torna-se um desafio. Outra dificuldade seria conseguir definir regras que fossem genéricas o suficiente para abranger a variedade de tons e timbres da voz Humana.

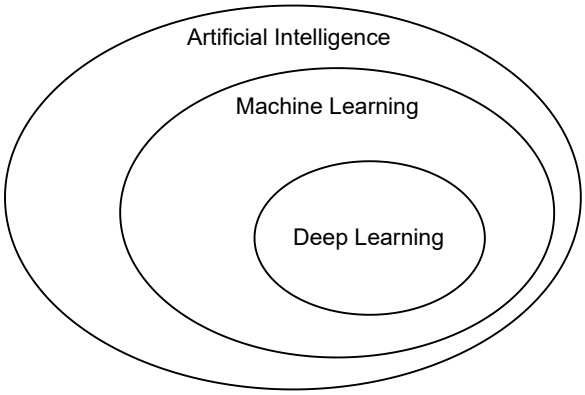


Figura 2.1: Relação entre os termos AI, ML e DL

Fonte: adaptado de [1]

Diferente da *Inteligência artificial simbólica* o ML tem como elemento central os dados. É através deles que o sistema *aprende* a inferir as regras, o que elimina a necessidade da caracterização das mesmas, e ser flexivel a medida que novos dados são injetados conforme

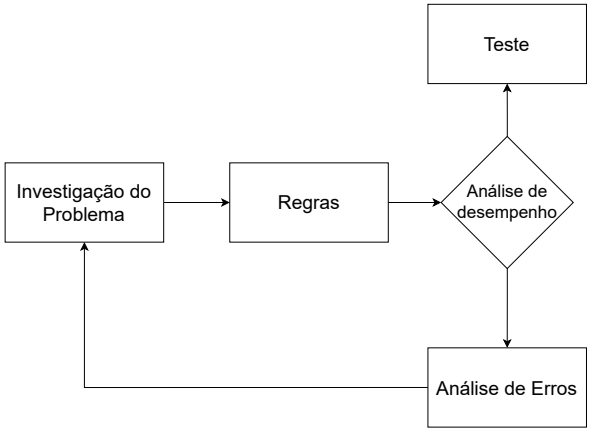


Figura 2.2: AI Simbólico

Fonte: adaptado de [2]

apresentado na Figura 2.3. Podemos categorizar o aprendizado de algorítmos de ML em dois grandes campos:

- Aprendizagem supervisionada
- Aprendizagem não supervisionada

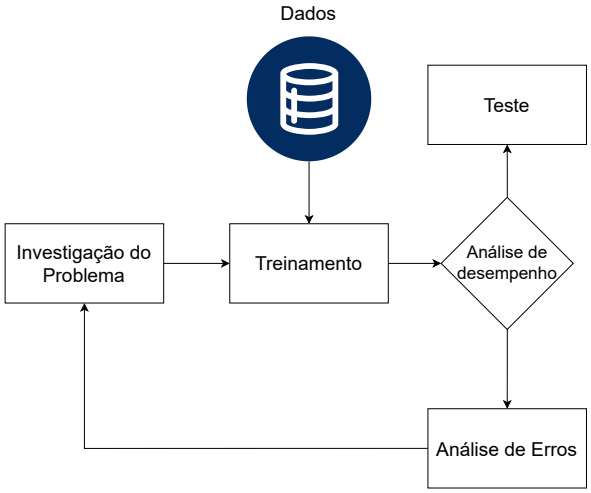


Figura 2.3: Aprendizado de Máquina

Fonte: adaptado de [2]

Na aprendizagem supervisionada, o algorítmo é alimentado não somente com os dados de entrada mas também com a resposta desejada, este trabalho concentra-se neste tipo de aprendizado, já na aprendizagem não supervisionada o algorítmo consegue inferir somente através dos

dados de entrada a resposta desejada. São algorítmos utilizados em aprendizagem supervisionada: *Redes Neurais*, *Regressão Logística* e *Regressão Linear*.

Em uma *Rede Neural*, o processo de aprendizado se dá através da atualização dos pesos de cada neurônio ao longo das camadas, e é justamente o número destas que caracteriza o DL. Tomando como exemplo a rede neural representada na Figura 2.4, nela vemos 2 camadas escondidas tendo 3 neurônios cada, além disso notamos na camada de saída um único neurônio. O objetivo é atualizar os pesos e "bias"destas 3 camadas de modo a minimizarmos a *função de custo* $J(w, b)$. Este processo é feito em duas etapas: propagação e retropropagação. Na etapa de propagação os dados da camada de entrada passam por todas as camadas e é calculada a diferença entre o valor gerado na camada de saída e o valor desejado, chamamos essa diferença de *perda*. Sendo assim, seja a matriz $X_{n_x,m}$ onde n_x representa o número de componentes de cada amostra - no exemplo $n_x = 2$ - e m igual ao número de amostras do *Dataset*, temos que a saída da *primeira camada escondida* calculada por:

$$A^{[1]} = \sigma(W^{[1]T} * X + b^{[1]}) \tag{2.1}$$

Onde $W^{[1]}$, $b^{[1]}$ e σ são, respectivamente a matriz dos pesos dos neurônios da camada 1, a matriz dos valores de *bias* dos neurônios da camada 1 e finalmente a função de ativação da mesma camada. Para a segunda camada escondida são performados cálculos semelhantes. Na camada de saída temos o seguinte :

$$\hat{Y} = \sigma(W^{[2]T} * A^{[2]} + b) \tag{2.2}$$

Esta Matriz representa a saída da rede neural para cada amostra do *Dataset* e a partir destas, podemos calcular a função de perda para a i-ésima amostra através de:

$$L^{(i)} = y^{(i)} - \hat{y}^{(i)} \tag{2.3}$$

Com a função de perda, calcula-se a função de custo:

$$J(w, b) = \frac{1}{m} * \sum_{n=1}^{m} L^{(i)} \tag{2.4}$$

É através da função de custo que os pesos de cada neurônio são atualizados, gerando o *aprendizado*. Esse processo de atualização se dá através da seguinte equação $w = w - \frac{\partial J}{\partial w} * \alpha$.

Onde α é a taxa de aprendizado. A atualização de cada peso é realizada pelo algorítmo de *backpropagation*, cuja explicação está fora do escopo deste trabalho.

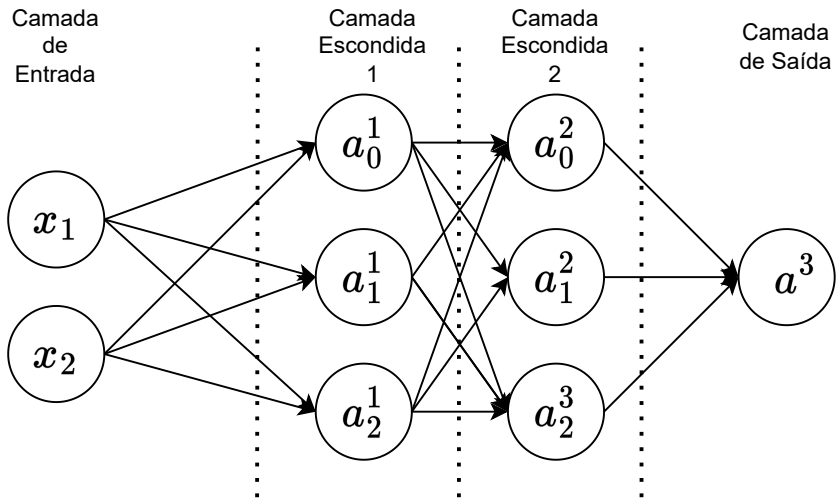


Figura 2.4: Representação de uma Rede Neural com duas camadas escondidas

Fonte: Autor

2.2 Descrição do Problema

Em modelos de comunicação que utilizam Millimeter Waves (mmWaves) os ”beams”do transmissor e do receptor devem estar alinhados de modo a otimizar a razão sinal-ruído SNR, do inglês ”Signal to Noise Ratio” [14]. Seja N_t o número de antenas do transmissor e N_r o número de antenas do receptor, pode-se calcular o número de beams (N_b) como:

$$N_b = N_t \times N_b \tag{2.5}$$

Neste trabalho, tratamos o Beam Selection como um problema de *classifição* de múltiplas classes, onde cada uma destas é representada por beam. No nosso caso, $N_b = 256$ pois $N_t = 32$ e $N_r = 8$

2.3 Dataset

Um dos principais elementos de qualquer trabalho que envolva ML é o *Dataset*, visto que é dele que o modelo extrai conhecimento, portanto, trata-se de uma escolha sensível dado o

grau de influência no resultado final. Em cenários em que se opta por uma arquitetura de DL é fundamental levar-se em conta que a disponibilidade de dados é um fator determinante para o melhor aproveitamento do modelo, uma vez que conforme explicado anteriormente, diferente de outras soluções de ML onde existe uma limitação na influência que a quantidade de dados exerce sobre a performance, o DL performa melhor com dados abundantes.

Dada a importância da disponibilidade de dados, este fator torna-se um ponto chave ao empregar técnicas de DL em problemas relacionados a comunicação baseada em redes *5G*, onde a obtenção de dados reais torna-se muitas vezes inviável em função dos altos custos associados. Em vista disso, o uso de dados sintéticos - provenientes de ambientes simulados - torna-se uma solução pertinente, como pode ser visto em trabalhos tais quais [15]. Sendo assim, o *Dataset* utilizado neste trabalho foi o Raymobtime [4, 16]. A escolha por este *Dataset* se deu por algumas características como:

- Diversidade de dados
- Aplicabilidade em cenários de mobilidade

A arquitetura utilizada na geração de dados pelo Raymobtime combina o software *Wireless Insite* [17] da empresa *Remcom* e o simulador de tráfego *SUMO* [18]. O uso do *Wireless Insite* mostra-se extremamente vantajoso pois simuladores que usam Ray Tracing (RT) tem sido largamente utilizado na modelagem de canais de comunicação sem fio de alta frequência em vista de que torna-se possível, por exemplo, capturar os efeitos das reflexões do sinal em edificações e suas irregularidades [19] o que é particularmente conveniente em estudos que lidam com cenários urbanos. Apesar de vantajoso, é fundamental evidenciar que o RT é extremamente custoso do ponto de vista computacional, sendo assim, a solução adotada pelo Raymobtime foi a de ao invés de capturar continuamente as informações durante a simulação, opta-se por por extrair *episódios* compostos por *cenas*. Entre cada cena é realizada a integração com o simulador *SUMO* o que permite que haja mobilidade não apenas dos receptores mas também de possíveis bloqueadores ao longo dos diferentes episisódios. A figura 2.5 ilustra a composição dos episódios.

Das opções disponibilizadas pelo Raymobtime, neste trabalho foi utilizada a versão S008. Nesta versão são incluídos dados de : canal, coordenadas, imagem e LIDAR. A tabela 2.1 sumariza algumas deste *Dataset*.

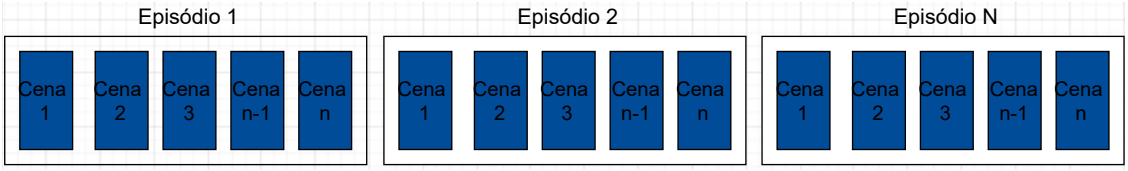


Figura 2.5: Composição dos episódios

Fonte: adaptado de [3]

Tabela 2.1: Características do Dataset S008

Fonte: Adaptado de [4]

Versão do Wireless Insite	3.2
Cenário 3D	Rosslyn
Frequência	60 GHz
Número e tipo de receptores	10 do tipo móvel
Tempo entre cenas	0.1 s
Tempo entre episódios	30 s
Número de episódios	2086
Número de cenas por episódio	1
Número de canais válidos	$11k$

2.3.1 Pré-processamento dos dados de coordenada

Os dados de coordenada do Raymobtime [4] estão disponíveis no arquivo *CoordVehiclesRxPerScene_s008.csv*. Este arquivo do tipo Comma-separated values (CSV), que dispõe os dados de maneira tabular, onde as informações de coordenadas estão disponíveis nas colunas *x*, *y* e *z*. A Tabela 2.2 é uma representação simplificada das primeiras 5 linhas deste arquivo.

De modo a utilizar estes dados na arquitetura de ML proposta, os dados passam pelo seguinte pré-processamento:

- Separar os dados na proporção de 75% para treino e 25% para validação
- Salvar os dados de treino e validação em arquivos sem compressão usando a função Savez [20] da biblioteca Numpy

Os arquivos são separados em duas porções de modo a evitar o processo de overfitting, ou seja, evitar que o modelo performe bem apenas no dataset que é utilizado no treinamento. No

Tabela 2.2: Representação simplificada do arquivo CoordVehiclesRxPerScene_s008.csv

Val	EpisodeID	...	x	y	z	rays	LOS
I	0	...	704.18681848265	670.58971778045	1.59	25	LOS=0
V	0	...	753.3833848633	655.44754663915	4.3	25	LOS=0
I	0	...	783.94351941255	408.6253222917	1.59	25	LOS=1
I	0	...	832.2173789167	371.13676687655	3.2	25	LOS=0
V	0	...	756.6564658932	514.19045865405	1.59	25	LOS=1

final, são obtidos dois "arrays"de dimensões $[9234, 2]$ e $[1960, 2]$ usados para, respectivamente, treinar e validar o modelo.

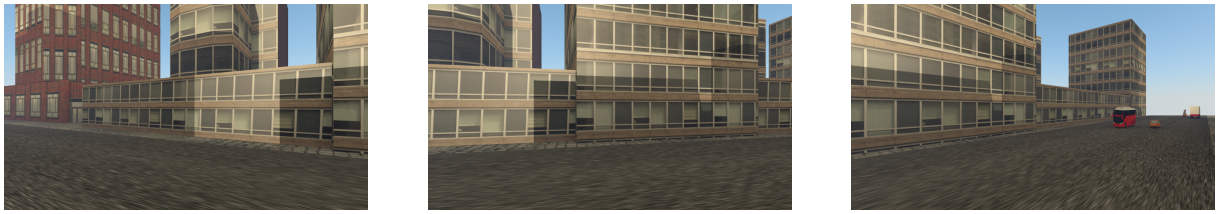
2.3.2 Pré-processamento dos dados de LIDAR

O processamento realizado nos dados de LIDAR é realizado filtrando, primeiramente, a partir do arquivo CSV apenas os dados onde o canal é valido. Feita a verificação é extraido o número total de amostras que será utilizado para fazer a distribuição entre dados de *treinamento* e *validação*. Este dado é também importante pois, de modo a garantir um melhor desempenho ao algorítmo de pré-processamento, usamos este valor para realizar a pré-alocação de um "array"de tamanho $(11194, 20, 200, 10]$ onde o primeiro valor representa o número de amostras utilizadas(que possuem o canal válido). Posteriormente este "array"é dividido em duas partes de tamanho $[9234, 20, 200, 10]$ e $[1960, 20, 200, 10]$ usadas para para treinar e validar o modelo.

2.3.3 Pré-processamento dos dados de imagem

Da mesma forma que os dados de *Coordenadas* e *LIDAR*, os dados de imagem precisam também passar por um pré-processamento antes de serem utilizados na arquitetura de ML proposta. As imagens disponíveis no *dataset S008* do Raymobtime [4] possuem dimensão de *960 X 540* pixels cada, a figura 2.6 mostra uma composição entre as imagens obitadas, respectivamente, pelas cameras 2, 1 e 3.

O processo de seleção das imagens é similar ao que é feito com os dados de coordenada ao passo em que apenas imagens de cenas válidas são levadas em conta, essa validação é feita através do mesmo arquivo CSV utilizado no pré-processamento de dados de coordenada. Uma



(a) Imagem da Camera 2 **(b)** Imagem da Camera 1 **(c)** Imagem da Camera 3

Figura 2.6: composição entre as imagens obtidas pelas cameras na primeira cena válida

Fonte: Autor

Figura 2.7: Exemplo de Imagem obtida após o processamento

Fonte: Autor

vez validadas, as imagens passam pelos seguintes processos:

1. Redução das dimensões por um fator de 10;

2. Filtragem passa-alta para destacar as bordas;

3. Junção das 3 imagens referentes a uma cena de modo a criar uma visão panorâmica.

De modo que, ao fim destes passos tenhamos dois ”arrays”de dimensões $[9234, 96, 162, 1]$ e $[1960, 96, 162, 1]$ utilizados, respectivamente, para a *treinamento* e *validação* do modelo de ML que processa imagens. Após o processamento a imagem 2.6 é transformada na imagem 2.7.

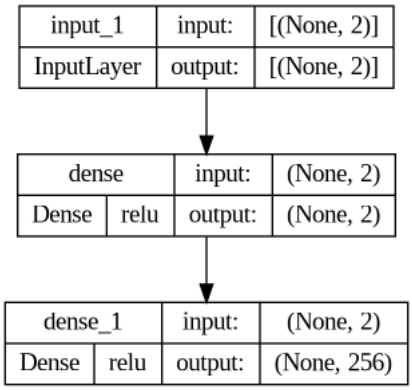


Figura 2.8: Visão geral da arquitetura de coordenadas

Fonte: Autor

2.4 Arquitetura de aprendizado de máquina

2.4.1 Modelo proposto

O modelo proposto neste trabalho é uma composição de três modelos subespecializados para cada tipo de dados de entrada (Coordenadas, Imagens e LIDAR). Nas subseções 2.4.1.1, 2.4.1.2 e 2.4.1.3 cada arquitetura será detalhada. Por fim, na subseção 2.4.1.4 será explicado como o modelo é configurado quando se é utilizado mais de um tipo de dado de entrada.

2.4.1.1 Arquitetura de AI para coordenadas

A arquitetura projetada para processar os dados de coordenada é composta por: camada de entrada, uma camada escondida e a camada de saída. A camada de entrada possui tamanho 2 visto que a mesma recebe o par de coordenadas (uma para o eixo x e outra para o eixo y) enquanto que a camada escondida possui possui dois neurônios com funções de ativação do tipo Rectified Linear Unit (ReLU). Por fim, na camada de saída são utilizados 256 neurônios, que são o número de classes possíveis e também é utilizada a função de ativação ReLU. A figura 2.8 mostra uma visão geral da arquitetura projetada para lidar com os dados de coordenadas.

2.4.1.2 Arquitetura de AI para imagens

Em se tratando da arquitetura utilizada no processamento de imagens, temos um cenário diferente do que acontece com os dados de coordenada, visto que para imagens foi explorado o uso de Redes neurais do tipo Convolutional Neural Network (CNN), dada a notável perfor-

mance que este tipo apresenta quando usada em dados de imagem. A arquitetura é composta da seguinte forma: camada de entrada, duas camadas escondidas do tipo *Conv2D* intercaladas por duas camadas do tipo *MaxPooling2D*, uma camada do tipo *Dropout* e por fim uma camada do tipo *Flatten* e a camada de saída.

A camada de entrada possui o formato de entrada $[96, 162, 1]$. O primeiro valor é referente a altura original da imagem reduzida por um fator de 10, o segundo valor é referente ao valor original de largura também reduzido por um fator de 10, porém conforme explicado anteriormente, durante o pré-processamento as imagens das três câmeras são unidas de modo a dar um efeito panorâmico. O último valor é referente ao número de canais utilizados, como neste trabalho usam-se as imagens em preto e branco temos apenas 1 canal.

As camadas escondidas do tipo *Conv2D* foram ambas configuradas igualmente. São utilizados 4 filtros com dimensão $3X3$ usando a função ReLU como função de ativação. Além disso, o parâmetro ”padding”é setado para ”same”, isso é feito para que a dimensão dos dados não mude da entrada para saída.

As camadas do tipo *MaxPooling2D* servem o propósito de realizar uma subamostragem nos dados, no caso específico deste trabalho ambas as camadas foram configuradas de modo que em sua saída os dados tivessem metade do tamanho de entrada, conforme pode ser notado na Figura 2.9. A subamostragem obtida pelas camadas *MaxPooling2D* é importante pois graças a ela podemos obter uma rede com um menor número de parâmetros a serem treinados.

Por fim, temos a camada de *Dropout* que serve o proposito de regularizar a rede contribuindo de modo a evitar o processo de ”overfitting”. Antes da camada de saída, que também é constituida de 256 neurônios e usando a função de ativação ReLU, temos uma camada do tipo *Flatten* que serve para planificar os dados antes da camada de saída.

2.4.1.3 Arquitetura de AI para Lidar

A arquitetura utilizada no processamento dos dados de LIDAR é bem semelhante aquela usada para os dados de imagens, ao passo que esta funciona também baseada em redes do tipo CNN. Uma visão geral da mesma pode ser vista na Figura 2.10

2.4.1.4 Arquitetura de AI para dados multimodais

Um dos principais objetivos deste trabalho é explorar as possibilidades que utilizar dados de diferentes naturezas podem possibilitar. Esse tópico é tratado com destaque devido ao fato de

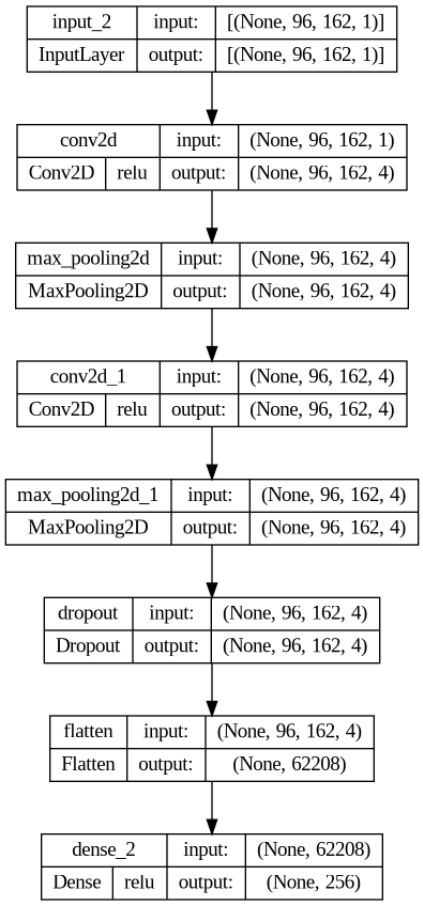


Figura 2.9: Visão geral da arquitetura de imagens

Fonte: Autor

que veículos, sobretudo aqueles com algum grau de autonomia, dependem fundamentalmente de diferentes tipos de sensores para seu correto funcionamento. Alguns exemplos são: Cameras, LIDAR, radar, sonar e GPS [21]. Como cada veículo pode apresentar um conjunto específico de sensores, é fundamental que a arquitetura seja flexível e consiga ser adaptada para as entradas disponíveis.

A estratégia utilizada neste trabalho para lidar com múltiplas entradas foi a de configurar a arquitetura de maneira dinâmica de acordo com os dados disponibilizados como entrada. Em um primeiro momento, as arquiteturas de ML são criadas para tipo de dado disponibilizado. Por exemplo, se são disponibilizados dados de *coordenada* e *imagem* as arquiteturas explicadas nas subseções 2.4.1.1 e 2.4.1.2 são criadas e configuradas normalmente e em seguida as camadas de saída dos modelos são concatenadas através da funçao *concatenate* [22] de modo que, se anteriormente as camadas de saída das arquiteturas de *coordenada* e *imagem* tinham, respectivamente, os formatos $[None, 256]$ a partir da concatenação temos $[None, 512]$. Após esta etapa,

adiciona-se uma camada do tipo *Densa* com 256 neurônios e função de ativação ”softmax”que passará a ser a nova *camada de saída*. Por fim, antes do modelo ser compilado, configuramos a entrada para que esta seja uma lista de ”arrays”composta pela camada de entrada do modelo de *coordenadas* e pela entrada do modelo de *imagem*. A figura 2.11 a arquitetura resultante obtida neste exemplo.

Vale ressaltar que apesar do exemplo dado utilizar os dados de imagem e coordenada, este mesmo processo é realizado em qualquer das possíveis combinações entre dados multimodais.

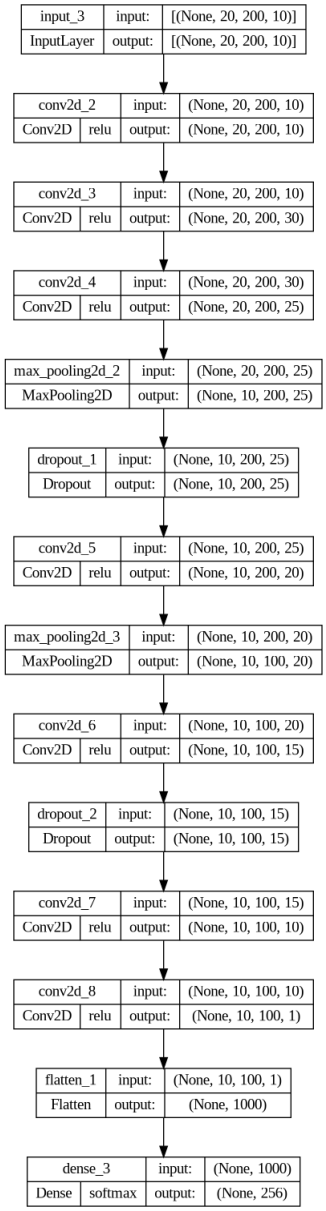


Figura 2.10: Visão geral da arquitetura de Lidar

Fonte: Autor

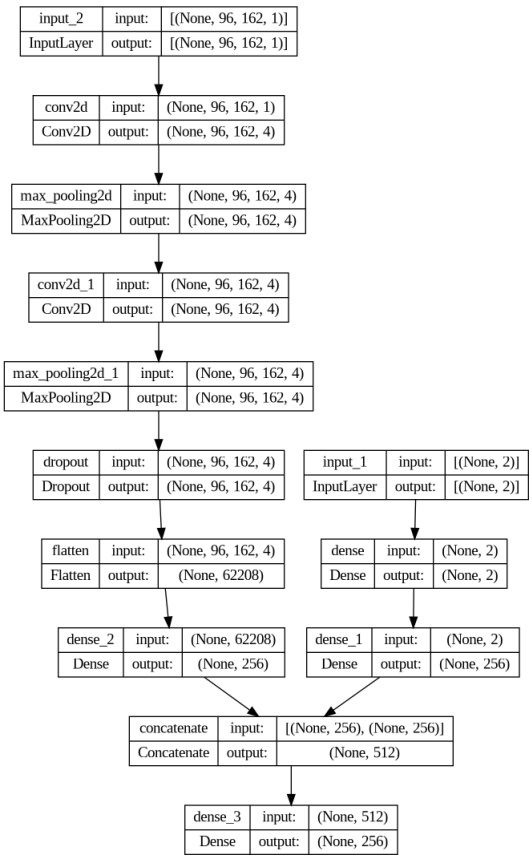


Figura 2.11: Visão geral da arquitetura de Multimodal usando Coordenadas e Imagens

Fonte: Autor

Capítulo 3

Resultados

Nesta Seção serão apresentados e discutidos os resultados obtidos através da proposta apresentada neste trabalho. Na Seção 3.1 serão explicadas as métricas utilizadas para avaliação dos resultados gerados, que por sua vez, serão exibidos na Seção 3.3.

3.1 Métricas de avaliação

3.1.1 top_k_categorical_accuracy

De modo a avaliar o desempenho obtidado pelo modelo proposto, faz-se necessário o uso de métricas que consigam expor de maneira simplificada o quão bem tal modelo performa. Sendo assim, neste trabalho, optou-se por utilizar as seguintes métricas: *top_1_categorical_accuracy*, *top_5_categorical_accuracy* e *top_10_categorical_accuracy*. Estas métricas foram obtidas usando a função *top_k_categorical_accuracy*[23] da biblioteca Keras [24].

A métrica *top_k_categorical_accuracy* funciona da seguinte forma: supondo que se esteja trabalhando com um problema de classificação de múltiplas classes usando codificação do tipo "one-hot" seja Y_{real} o conjunto de respostas esperadas e Y_{pred} o conjunto de predições obtidas

$$\begin{aligned} Y_{real} = [&[0, 0, 0, 0, 1],\\ &[0, 0, 0, 1, 0],\\ &[1, 0, 0, 0, 0]] \end{aligned}$$

$$Y_{pred} = [[0.02, 0.35, 0.55, 0.05, 0.03],$$
$$[0.02, 0.01, 0.01, 0.95, 0.01],$$
$$[0.93, 0.01, 0.02, 0.02, 0.02]]$$

Com esses dados em mãos, o próximo passo é ordenar de maneira decrescente as probabilidades em cada uma das predições, obtendo-se assim:

$$Y_{pred} = [[0.55, 0.35, 0.05, 0.03, 0.02],$$
$$[0.95, 0.02, 0.01, 0.01, 0.01],$$
$$[0.93, 0.02, 0.02, 0.02, 0.01]]$$

Desta forma, temos que as métricas top_1_categorical_accuracy e top_3_categorical_accuracy deste exemplo seriam ambas iguais a 0.66 pois após a ordenação a predição correta apareceu em dois dos três casos seja na primeira posição ou entre as três primeiras. De maneira simplificada, compreende-se que quanto maior o valor de k mais permissiva a métrica se torna, de modo que um modelo razoavelmente bom deve apresentar, pelo menos, um alto valor em top_5_categorical_accuracy.

3.1.2 Precisão e Recall

Serão usadas também outras duas métricas que comumente são aplicadas em problemas de classificação de múltiplas classes: *precisão* e *recall*. De modo a facilitar a sua compreensão, suponhamos que temos o seguinte desafio: medir a performance de um modelo de classificação binária, ou seja, com apenas duas classes cujo objetivo é detectar ou não a presença de um agente patógeno em uma determinada amostra. Neste caso, temos basicamente 4 cenários possíveis:

- Verdadeiro Positivo (VP): A amostra é classificada como contaminada e está, de fato, contaminada
- Falso Positivo (FP): A amostra é classificada como contaminada mas não está contaminada
- Verdadeiro Negativo (VN): A amostra é classificada como não contaminada e, de fato, não está contaminada

- Falso Negativo (FN): A amostra é classificada como não contaminada mas está contaminada

Em um cenário como este, com poucas classes, é comum recorrer a ferramentas de visualização como, por exemplo, uma matriz de confusão[25] que é exemplificada na figura 3.1.

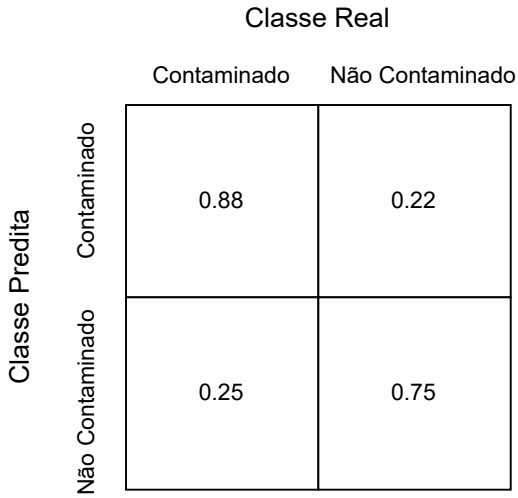


Figura 3.1: Exemplo de Matriz de Confusão

Fonte: Autor

No exemplo utilizado vemos que o classificador prevê corretamente, um *Verdadeiro Positivo*, 88% das vezes enquanto que para o caso de *Falso Positivo* a taxa é de 75% das vezes. A matriz de confusão torna-se portanto em um exemplo de poucas classes uma excelente ferramenta de visualização, entretanto, para um problema de muitas classes, como é o caso deste trabalho, seu tamanho se tornaria inviável e sua capacidade de facilmente representar os dados se perderia. Para fins de curiosidade, neste trabalho temos 256 classes, logo a matriz de confusão teria o inconcebível tamanho de $256X256$.

Apesar da limitação para visualização em problemas com múltiplas classes, a partir da *Matriz de Confusão* podemos extrair métricas muito esclarecedoras em problemas de classificações independentemente do número de classes. São os casos das métricas de *precisão* e *recall* que podem ser calculadas para cada classe. A precisão de uma classe i pode ser definida por:

$$Precisão_i = \frac{VP_i}{VP_i + FP} \tag{3.1}$$

Consideramos como verdadeiro positivo da classe i (VP_i) todas as ocorrências em que a saída predita Y_{pred} e a saída real Y_{real} correspondem a classe i. Analogamente, é considerado

como FP toda saída predita como pertencente a classe i entretanto a saída real é pertencente a qualquer outra classe. De forma semelhante o recall de uma classe i é calculado como por:

$$recall_i = \frac{VP_i}{VP_i + FN} \tag{3.2}$$

Na métrica de *recall* o VP_i é intepretado da mesma forma que na *precisão*, já o FP é constituido de toda saida predita Y_{pred} que aponte para qualquer outra classe quando a classe real é i.

Estas métricas são importantes para avaliar eventuais otimizações que um modelo pode passar. Por exemplo, no problema avaliado neste trabalho se o *recall* de um determinado beam apresenta um valor muito baixo, isso indica que o modelo prevê erroneamente qualquer outro beam como correto ao invés do beam real, o que impacta na performance do modelo. Em [26] algumas das métricas mais utilizadas em problemas de classificação com múltiplas classes são detalhadas e exemplificadas.

3.2 Resultados obtidos pela baseline Dummy

De modo a auxiliar na avaliação da performance dos modelos apresentados, uma baseline foi utilizada. Neste trabalho, optou-se pela utilização de uma baseline do tipo “Dummy”, ou seja, aquela cujos resultados não são extraidos com base nos padrões existentes nos dados de entrada [27]. Sendo assim, o modelo ”DummyClassifier” [28], disponibilizado pela biblioteca ”Scikit-Learn”, que consiste em um classificador do tipo ”Dummy”e seu funcionamento é regido, basicamente, pelo parâmetro ”strategy”.

Para gerar a baseline deste trabalho, o parâmetro ”strategy”foi configurado como ”stratified”. Os dados usados para o treinamento e validação , usando os método ”fit”e ”score”, respectivamente, foram os mesmos usados nos modelos reais apresentados na seção 3.3. A acurácia média obtida pela baseline para cada tipo de dado pode ser vista na Tabela 3.1. É importante destacar que por limitação da implementação, não há suporte para múltiplas entradas, logo só foi possível obter resultados na baseline ”Dummy”para dados singulares.

Tabela 3.1: Resultados obtidos pela baseline Dummy

Fonte: Autor

Tipo de Dados	Resultado
Coordenadas	0.025
Imagens	0.023
Lidar	0.028

3.3 Resultados obtidos utilizando o dataset S008

Os resultados aqui apresentados, utilizando dados multimodais ou não, foram obtidos através da mesma configuração apresentada na Tabela 3.2. A Figura 3.2 mostra os resultados que foram obtidos na métrica *top_k_categorical_accuracy* utilizando apenas um tipo de dado por vez, de modo a que possamos observar se eventualmente a combinação com outros tipos de dados resulta em melhor performance. Percebe-se que, neste cenário, os melhores resultados são obtidos usando os dados de LIDAR, que domina em todas as métricas e destaca-se em *top_10_categorical_accuracy* onde obtêm-se a performance de aproximadamente 87%, entretanto, vale ressaltar também o bom resultado obtido ao utilizar dados de imagens onde nesta mesma métrica a performance foi de aproximadamente 84%. Esse resultado é destacável pois o sensor de imagem é consideravelmente mais comum, além é claro de ser notadamente mais barato que um sensor LIDAR. Nas Figuras 3.3, 3.4 e 3.5 as métricas de *precisão* e *recall* são exibidas para cada tipo de dado de entrada, e podemos ver o porquê da acurácia tão baixa obtida ao usar somente dados de *coordenada*: este tipo de dado não consegue ser suficiente para fazer as predições corretas nos dados de validação, conseguindo apenas ter alguns acertos em 3 “beams”, muito diferente do que acontece quando utilizamos dados de LIDAR e Imagem.

Tabela 3.2: Configurações utilizadas nos modelos

Fonte: Autor

Função de Perda	Categorical cross-entropy
Otimizador	Adam
Número de Épocas	800
Tamanho do ”batch”	64

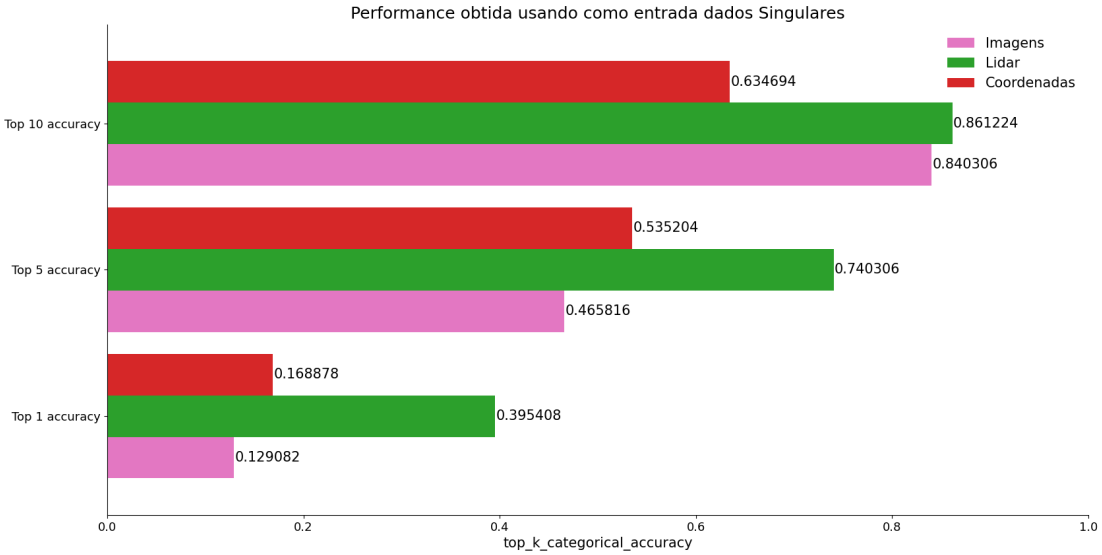


Figura 3.2: Resultados obtidos pela arquitetura proposta usando como dados de entrada apenas dados de um tipo por vez

Fonte: Autor

O uso de dados multimodais é o fundamento deste trabalho, desta forma, os mesmos testes realizados nos dados singulares foram executados para este cenário e os resultados obtidos para a métrica *top_k_categorical_accuracy* podem ser observados na figura 3.6.

Assim como observado no cenário de dados singulares, existe uma clara vantagem quando o LIDAR é utilizado. Por exemplo, ao utilizar dados de *Coordenada* combinados com dados de LIDAR é obtida uma excelente performance de aproximadamente 93%, 87% e 47%, respectivamente, nas métricas :

- top_10_categorical_accuracy
- top_5_categorical_accuracy
- top_1_categorical_accuracy

Esta performance foi superior aquela apresentada ao utilizar somente dados de LIDAR. Quando combinamos os dados *Coordenada* e de *Imagem* observa-se a performance de 83%, 67% e 24% nestas mesmas métricas o que é um indicativo de que pode-se melhorar, através da combinação de dados, a performance abaixo do desejado obtida ao se usar dados que não

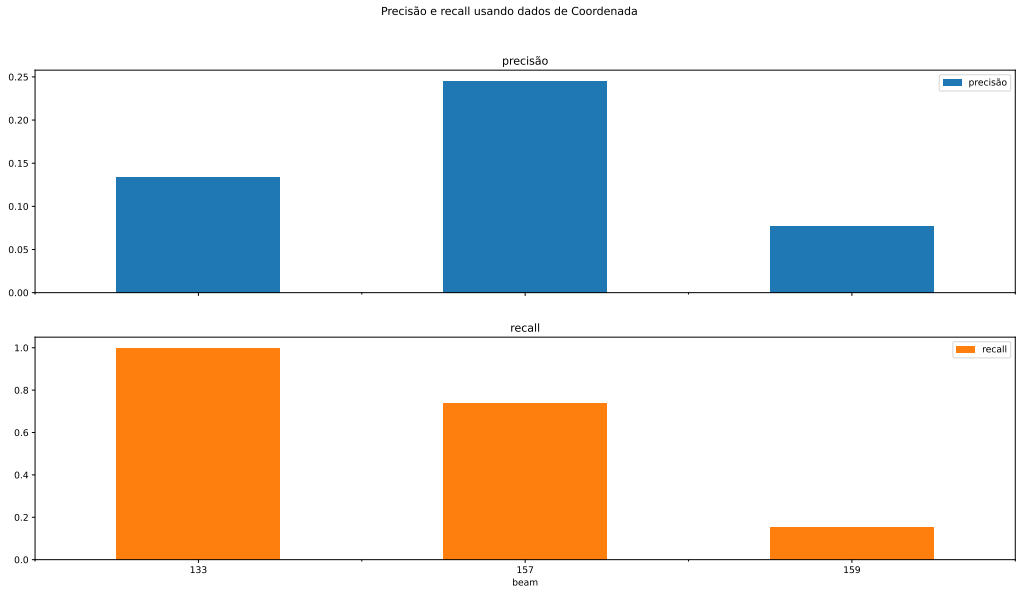


Figura 3.3: Precisão e Recall usando dados de Coordenada

Fonte: Autor

performam bem em determinado cenário. Essa observação é particularmente importante justamente pelo fato de ser possível obter considerável performance ao se utilizar dados que são comuns, como é o caso de *Coordenadas* e *Imagens*. As métricas de *precisão* e *recall* para todas as combinações podem ser visualizadas nas figuras 3.7, 3.8, 3.9, e 3.10. Pode-se perceber que através da combinação de dados, foi possível obter uma melhora significativa nos cenários em que se utilizam dados de *Coordenada* e isso se dá pelo fato de que, em tais cenários, o modelo performa muito melhor nos dados de validação conseguindo acertar um número maior de ”beams”.

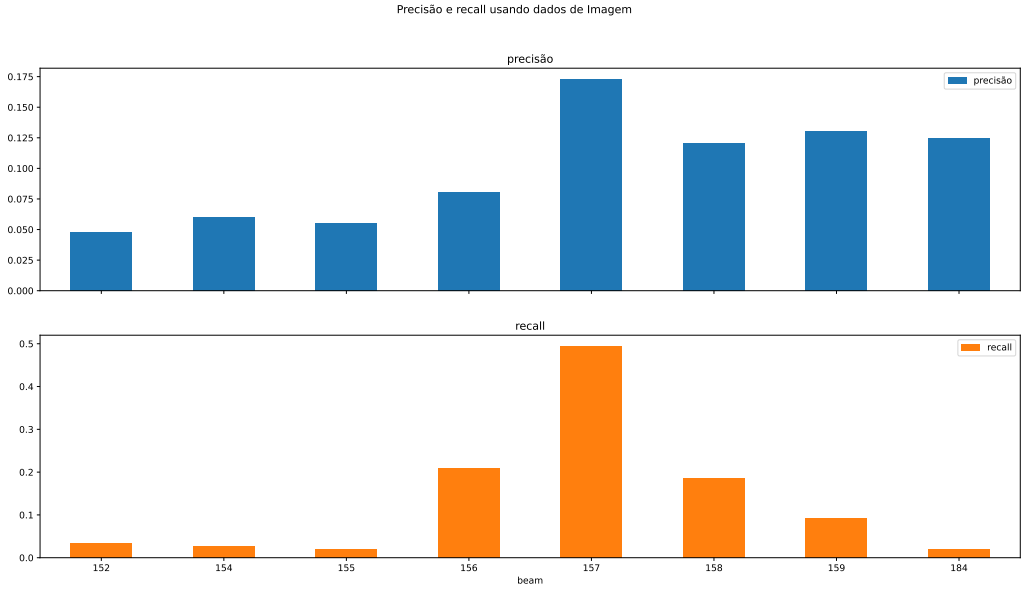


Figura 3.4: Precisão e Recall usando dados de imagem

Fonte: Autor

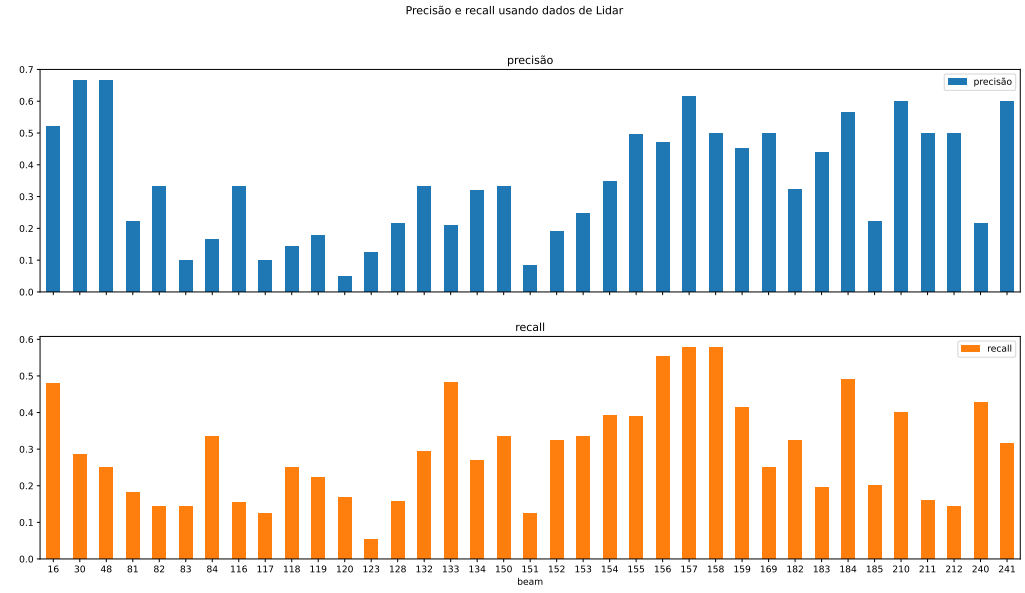


Figura 3.5: Precisão e Recall usando dados de Lidar

Fonte: Autor

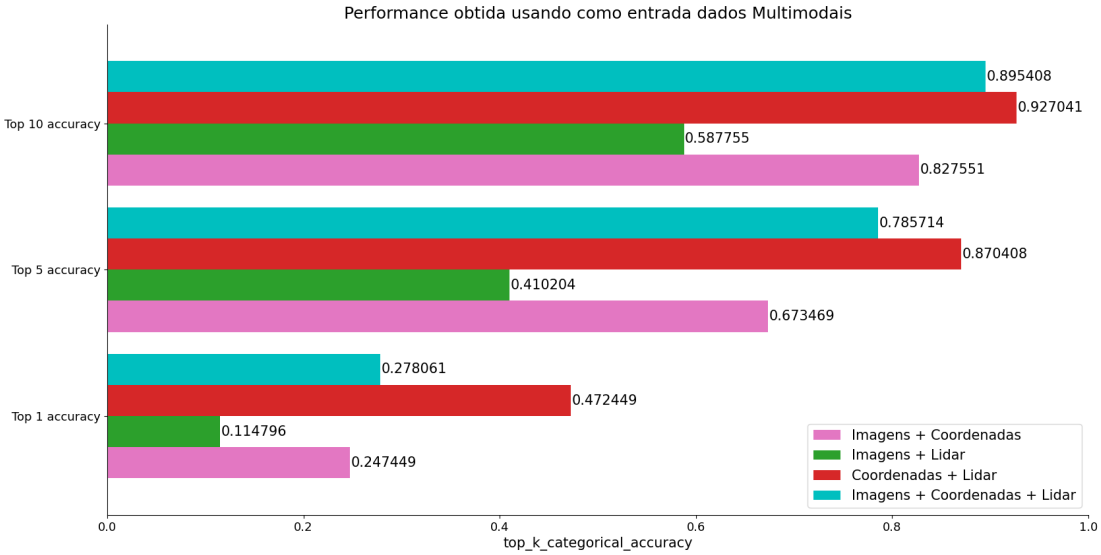


Figura 3.6: Resultados obtidos pela arquitetura proposta usando como dados de entrada a combinação dos diversos tipos de entrada

Fonte: Autor

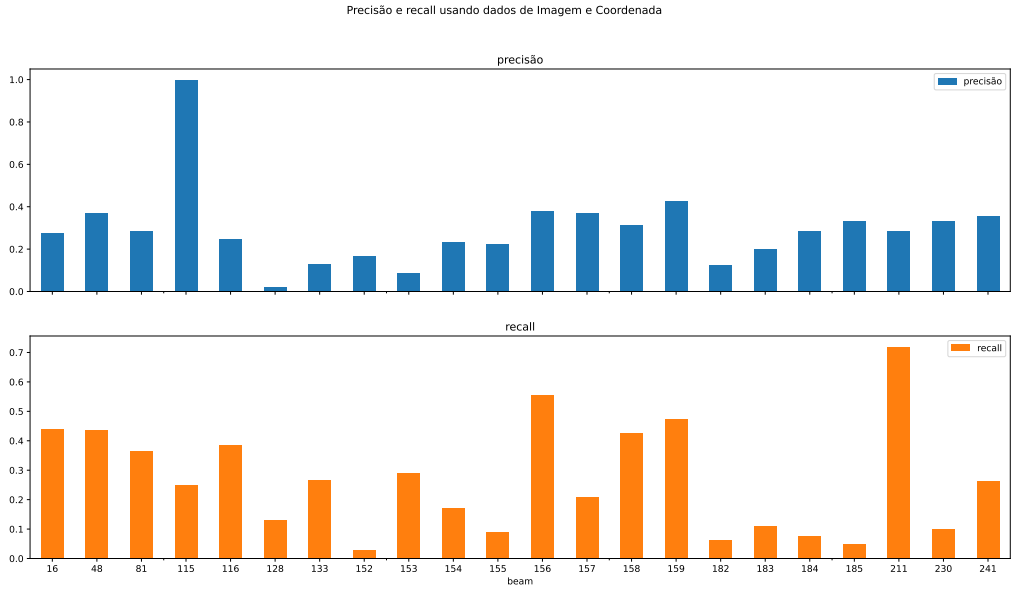


Figura 3.7: Precisão e Recall usando dados de Imagem e Coordenada

Fonte: Autor

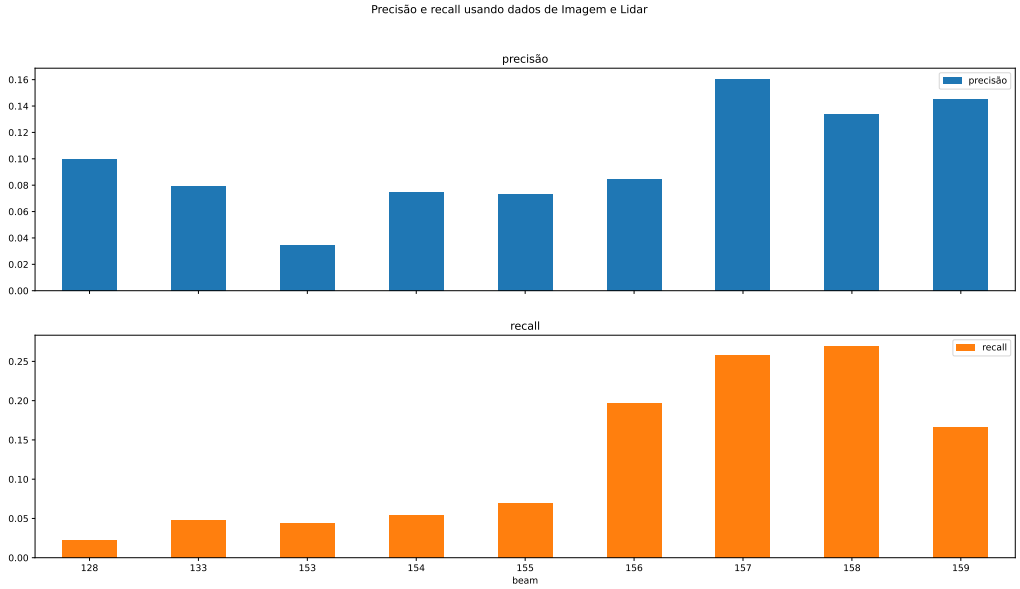


Figura 3.8: Precisão e Recall usando dados de Imagem e LIDAR

Fonte: Autor

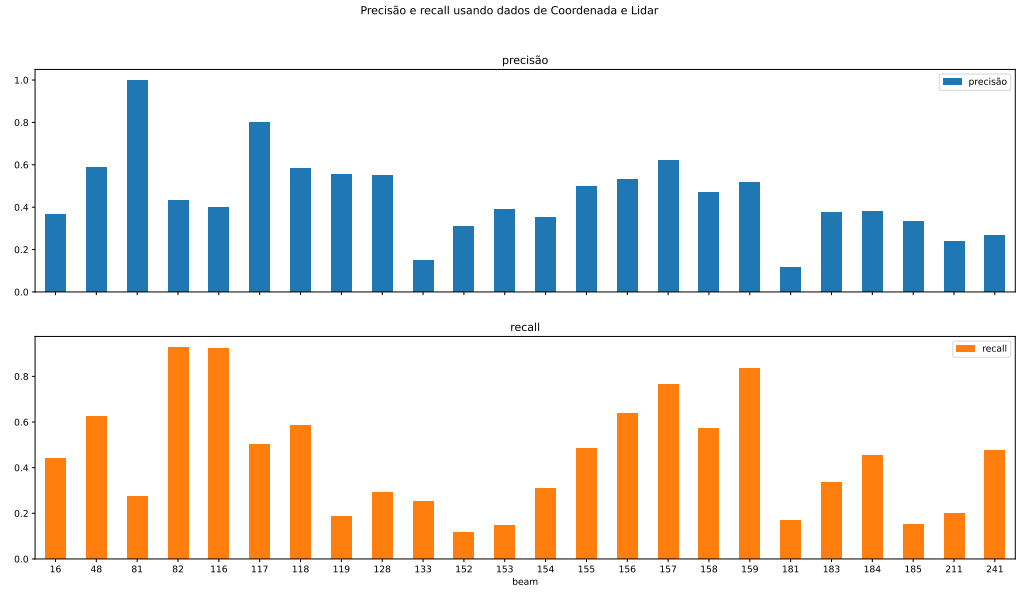


Figura 3.9: Precisão e Recall usando dados de Coordenada e LIDAR

Fonte: Autor

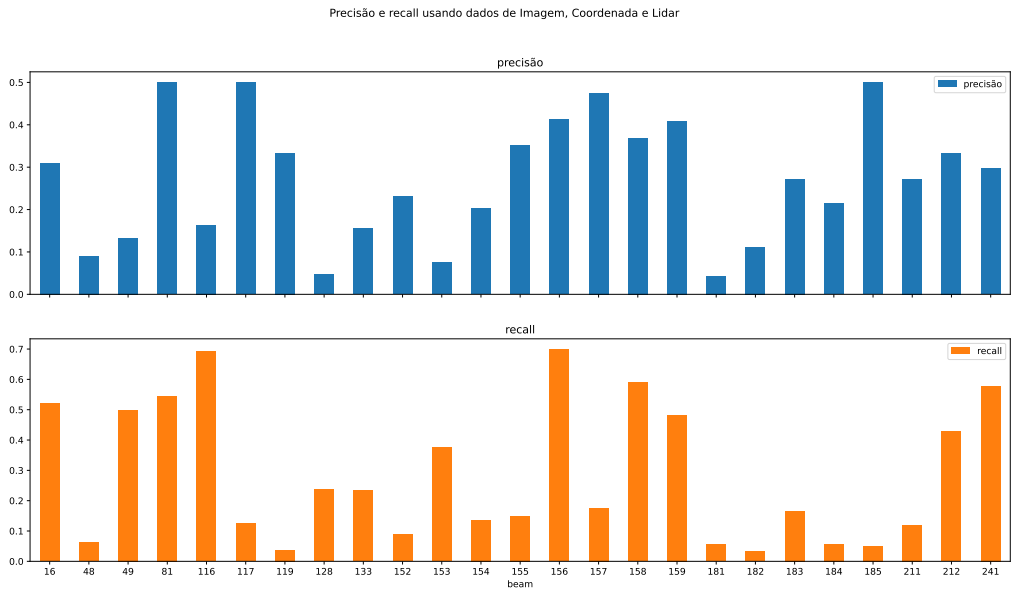


Figura 3.10: Precisão e Recall usando dados de Imagem, Coordenada e LIDAR

Fonte: Autor

Capítulo 4

Conclusão e Trabalhos Futuros

Este trabalho apresentou uma arquitetura de ML que, através da estratégia do uso de dados de diferentes naturezas, apresenta resultados interessantes no contexto de beam selection, tendo como destaque cenários em que se utilizam dados de LIDAR, por exemplo, ao combina-los com dados de coordenada. Destaca-se também os resultados obtidos nos cenários em que são utilizados dados de *coordenadas* e *imagens* visto que esse tipo de informação já pode ser coletada em um cenário real até mesmo nos mais econômicos modelos de carros, logo a arquitetura de ML proposta destaca-se por não ancorar a sua utilidade ao vislumbre de um futuro altamente conectado distante sobretudo para países emergentes.

O uso de dados sintéticos utilizados no treinamento e validação das redes neurais também merece destaque, visto que, além dos altos custos relacionados a uma coleta de dados do mundo real para estudos de tal calibre, usar dados provenientes de simulação torna-se cada vez mais interessante graças a disponibilidade de hardware especializado mais barato e eficiente ao longo dos anos aliado a facilidade com a qual os parâmetros de simulação podem ser alterados.

Dentre os possíveis trabalhos futuros destaca-se a possibilidade de simular os dados de imagens não apenas vindo de câmeras instaladas em estações base mas também nos próprios veículos, o que proporcionaria não só riqueza na qualidade dos dados como também tornariam os mesmos mais próximos da realidade, visto que é cada vez mais comum a presença de sensores de imagens em veículos, destacando-se aqueles utilizados em sistemas de assistência ao condutor.

Referências Bibliográficas

[1] F. Chollet, *Deep learning with Python*. Simon and Schuster, 2021.

[2] A. Géron, *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow*. "O'Reilly Media, Inc.", 2022.

[3] "Raymobtime — github.com," https://github.com/lasseufpa/Raymobtime/wiki/Raymobtime, [Accessed 22-10-2023].

[4] A. Klautau, P. Batista, N. González-Prelcic, Y. Wang, and R. W. Heath, "5g mimo data for machine learning: Application to beam-selection using deep learning," in *2018 Information Theory and Applications Workshop (ITA)*. IEEE, 2018, pp. 1–9.

[5] G. Y. Mihaylov, T. B. Iliev, T. D. Bikov, E. P. Ivanova, I. S. Stoyanov, V. P. Keseev, and A. Dinov, "Test cases and challenges for mobile network evolution from lte to 5g," in *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*. IEEE, 2018, pp. 0449–0452.

[6] S. Kutty and D. Sen, "Beamforming for millimeter wave communications: An inclusive survey," *IEEE communications surveys & tutorials*, vol. 18, no. 2, pp. 949–973, 2015.

[7] R. W. Heath, N. Gonzalez-Prelcic, S. Rangan, W. Roh, and A. M. Sayeed, "An overview of signal processing techniques for millimeter wave mimo systems," *IEEE journal of selected topics in signal processing*, vol. 10, no. 3, pp. 436–453, 2016.

[8] L. Wei, R. Q. Hu, Y. Qian, and G. Wu, "Key elements to enable millimeter wave communications for 5g wireless systems," *IEEE Wireless Communications*, vol. 21, no. 6, pp. 136–143, 2014.

[9] S. Rezaie, C. N. Manchón, and E. De Carvalho, "Location-and orientation-aided millimeter wave beam selection using deep learning," in *ICC 2020-2020 IEEE International Conference on Communications (ICC)*. IEEE, 2020, pp. 1–6.

[10] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[11] M. Alrabeiah, A. Hredzak, and A. Alkhateeb, "Millimeter wave base stations with cameras: Vision-aided beam and blockage prediction," in *2020 IEEE 91st vehicular technology conference (VTC2020-Spring)*. IEEE, 2020, pp. 1–5.

[12] C.-H. Lin, W.-C. Kao, S.-Q. Zhan, and T.-S. Lee, "Bsnet: A deep learning-based beam selection method for mmwave communications," in *2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall)*. IEEE, 2019, pp. 1–6.

[13] OpenAI, "Chatgpt: Optimizing language models for dialogue," Feb 2023. [Online]. Available: https://openai.com/blog/chatgpt/

[14] J. Ferreira, D. Gomes, and A. Klautau, "A new multimodal approach to beam-selection based on deep learning," *Encom 2020*, 2020.

[15] J. Viana, H. Farkhari, P. Sebastiao, S. Lagen, K. Koutlia, B. Bojovic, and R. Dinis, "A synthetic dataset for 5g uav attacks based on observable network parameters," *arXiv preprint arXiv:2211.09706*, 2022.

[16] A. Klautau, N. González-Prelcic, and R. W. Heath, "Lidar data for deep learning-based mmwave beam-selection," *IEEE Wireless Communications Letters*, vol. 8, no. 3, pp. 909–912, 2019.

[17] Remcom, "Wireless em propagation software - wireless insite," Mar 2023. [Online]. Available: https://www.remcom.com/wireless-insite-em-propagation-software

[18] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wießner, "Microscopic traffic simulation using sumo," in *The 21st IEEE International Conference on Intelligent Transportation Systems*. IEEE, 2018. [Online]. Available: https://elib.dlr.de/124092/

[19] B. Colo, A. Fouda, and A. S. Ibrahim, "Ray tracing simulations in millimeter-wave vehicular communications," in *2019 IEEE 30th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. IEEE, 2019, pp. 1–4.

[20] "Numpy.savez." [Online]. Available: https://numpy.org/doc/stable/reference/generated/numpy.savez

[21] J. Kocić, N. Jovičić, and V. Drndarević, "Sensors and sensor fusion in autonomous vehicles," in *2018 26th Telecommunications Forum (TELFOR)*. IEEE, 2018, pp. 420–425.

[22] Keras, "Keras: Concatenate layer," Mar 2023. [Online]. Available: https://keras.io/api/layers/merging_layers/concatenate/

[23] ——, "Keras: Accuracy metrics," Mar 2023. [Online]. Available: https://keras.io/api/metrics/accuracy_metrics/#topkcategoricalaccuracy-class

[24] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[25] S. Narkhede, "Understanding confusion matrix — by sarang narkhede — towards data science," Mar 2023. [Online]. Available: https://towardsdatascience.com/understanding-confusion-matrix-a9ad42dcfd62

[26] B. Tuychiev, "Comprehensive guide to multiclass classification metrics — towards data science," Mar 2023. [Online]. Available: https://towardsdatascience.com/comprehensive-guide-on-multiclass-classification-metrics-af94cfb83fbd

[27] B. Tezcan, "Why using a dummy classifier is a smart move — by berke tezcan — towards data science," Mar 2023. [Online]. Available: https://towardsdatascience.com/why-using-a-dummy-classifier-is-a-smart-move-4a55080e3549

[28] Scikit-learn, "sklearn.dummy.dummyclassifier," Mar 2023. [Online]. Available: https://scikit-learn.org/stable/modules/generated/sklearn.dummy.DummyClassifier.html#sklearn.dummy.DummyClassifier.score