CENTRO UNIVERSITÁRIO FEI

ANTONIO PEGORELLI NETO

ANÁLISE DO IMPACTO DA QUANTIDADE DE PONTOS DE WI-FI NA LOCALIZAÇÃO DE ROBÔS SEQUESTRADOS EM AMBIENTES INTERNOS UTILIZANDO TÉCNICAS DE REDES NEURAIS ARTIFICIAIS RECORRENTES

São Bernardo do Campo

2022

ANTONIO PEGORELLI NETO

ANÁLISE DO IMPACTO DA QUANTIDADE DE PONTOS DE WI-FI NA LOCALIZAÇÃO DE ROBÔS SEQUESTRADOS EM AMBIENTES INTERNOS UTILIZANDO TÉCNICAS DE REDES NEURAIS ARTIFICIAIS RECORRENTES

Dissertação de Mestrado, apresentada ao Centro Universitário da FEI como requisito para obtenção do título de Mestre em Engenharia Elétrica. Orientado pelo Prof. Dr. Flavio Tonidandel.

São Bernardo do Campo

2022

Pegorelli Neto, Antonio.
Análise do impacto da quantidade de pontos de Wi-Fi na localização de robôs sequestrados em ambientes internos utilizando técnicas de redes neurais artificiais recorrentes / Antonio Pegorelli Neto. São Bernardo do Campo, 2022.
124 p. : il.

Dissertação - Centro Universitário FEI.
Orientador: Prof. Dr. Flavio Tonidandel.

1. Localização por Wi-Fi. 2. Indicador de Potência de Sinal Recebido. 3. Problema do Robô Sequestrado. 4. Redes Neurais Recorrentes. 5. k-ésimo Vizinho Próximo. I. Tonidandel, Flavio, orient. II. Título.

Elaborada pelo sistema de geração automática de ficha catalográfica da FEI com os dados fornecidos pelo(a) autor(a).

centro universitário FEI

APRESENTAÇÃO DE DISSERTAÇÃO
ATA DA BANCA EXAMINADORA

Programa de Pós-Graduação Stricto Sensu em Engenharia Elétrica

Mestrado

PGE-10

Aluno: Antonio Pegorelli Neto **Matrícula:** 119101-4

Título do Trabalho: ANÁLISE DO IMPACTO DA QUANTIDADE DE PONTOS WI-FI NA LOCALIZAÇÃO DE ROBÔS EM AMBIENTES INTERNOS DE TÉCNICAS DE REDES NEURAIS RECORRENTES PARA LOCALIZAÇÃO DE ROBÔS SEQUESTRADOS.

Área de Concentração: Inteligência Artificial Aplicada à Automação e Robótica

Orientador: Prof. Dr. Flavio Tonidandel

Data da realização da defesa: 04/03/2022

ORIGINAL ASSINADA

Avaliação da Banca Examinadora:

O aluno fez sua apresentação de forma adequada. Respondeu as perguntas da banca satisfatoriamente. A banca considerou de forma unânime, o trabalho aprovado. Sugestões de melhorias foram feitas para a versão final mas que não alteram a essência do trabalho

APROVADO. A banca solicitou alteração do título da dissertação para "Análise do impacto da quantidade de pontos de Wi-Fi na localização de robôs sequestrados em ambientes internos utilizando técnicas de redes neurais artificiais recorrentes".

São Bernardo do Campo, / / .

MEMBROS DA BANCA EXAMINADORA	
Prof. Dr. Flavio Tonidandel	Ass.: _____
Prof. Dr. Reinaldo Augusto da Costa Bianchi	Ass.: _____
Prof. Dr. Rogério Sales Gonçalves	Ass.: _____

A Banca Julgadora acima-assinada atribuiu ao aluno o seguinte resultado:

APROVADO ☒ REPROVADO ☐

VERSÃO FINAL DA DISSERTAÇÃO

APROVO A VERSÃO FINAL DA DISSERTAÇÃO EM QUE FORAM INCLUÍDAS AS RECOMENDAÇÕES DA BANCA EXAMINADORA

_____

Aprovação do Coordenador do Programa de Pós-graduação

_____

Prof. Dr. Carlos Eduardo Thomaz

RESUMO

A localização de robôs em ambientes fechados, onde o *Global Positioning System* (GPS) não pode atuar, é um constante desafio dos sistemas que envolvem robôs autônomos. Soluções baseadas em análise de radiofrequência apresentam alta complexidade devido à natureza dos sinais, passíveis de diversas interferências em ambientes quanto maior a quantidade de equipamentos eletrônicos, além da movimentação de pessoas e objetos, aumentando a dificuldade de processamento das informações tratadas. As limitações de processamento e uso de energia em dispositivos móveis também limitam a implementação de hardware embarcado capaz de realizar cálculos complexos exigidos em diversas soluções para o processamento e análise das informações em tempo hábil para apresentar eficácia na atuação. Com a portabilidade desses sistemas robotizados encontramos um novo desafio, o sequestro de robôs, quando o usuário movimenta o robô manualmente, sem nenhuma notificação no sistema e dificultando o processo de localização. Neste trabalho é proposto um estudo das mais recentes técnicas de localização em ambientes fechados, utilizando redes neurais recorrentes, como a *Gated Recurrent Unit* (GRU) e a *Long-Short Term Memory* (LSTM) para o tratamento de dados de potência de sinal Wi-Fi, ou *Received Signal Strength* (RSS), assim como a aplicação da técnica *k-Nearest Neighbors* (KNN) para casos onde ocorre o sequestro de robôs. O estudo foi feito a partir de um simulador construído dentro do ambiente virtual Webots® a partir do *fingerprinting* de um ambiente real com 6 pontos de acesso Wi-Fi. Funções de distribuição acumulada foram utilizadas para avaliar cada um dos 3 sistemas para diversas combinações de pontos de acesso, níveis de ruído e níveis de atenuação simulados.

Os resultados mostram que os sistemas de redes neurais recorrentes podem alcançar acurácia entre 0.44m$\pm$0.39m para LSTM, e 0.50m$\pm$0.38m para GRU, enquanto para a proposta baseada em KNN, temos 0.68m$\pm$0.73m, demonstrando a capacidade dos sistemas avaliados de se recuperarem após um evento de sequestro de robôs, mantendo resultados semelhantes aos observados sem a ocorrência de tais eventos.

Palavras-chave: Localização por Wi-Fi, Indicador de Potência de Sinal Recebido, Problema do Robô Sequestrado, Redes Neurais Recorrentes, k-ésimo Vizinho Próximo, LSTM, GRU

ABSTRACT

The robot localization in indoor rooms, without GPS assist, is a big challenge for autonomous robotic systems. Due to the complexity of radio frequency nature, susceptible to interference dependent on electronic devices density in the area, people and object movement, raising the data processing complexity. Limited processing and power resources on mobile devices also limits the implementation of hardware capable of processing complex calculations required for data analysis for solutions in usable time, before its action efficiency is reduced. When the portability of those robotic systems is considered, a new challenge called the Kidnapped Robot Problem *(KRP) is presented, when those robots are handled by the user without any notification to its system, interfering in the localization process. This work proposes an analysis of the earliest indoor localization techniques based on recurrent neural networks like GRU and LSTM to process Wi-Fi received signal strength data, including KNN, for the kidnapped robot problem. This analysis will be built on a simulator in Webots® virtual environment and based on the* fingerprinting *of a real indoor room with 6 Wi-Fi access points. The efficiency of each system will be evaluated using cumulative distribution function for several access point combinations, noise and vanishing levels.*

The results show that RNN systems can achieve mean localization accuracy between 0.44m±0.39m for LSTM and 0.50m±0.38m for GRU and the KNN technique proposal reaching 0.68m±0.73m, proving the capability of those systems to recover from a KRP event keeping similar results obtained without any event.

Keywords: Wi-Fi Localization, Received Signal Strength Indicator, Kidnapped Robot Problem, Recurrent Neural Networks, k-Nearest Neighbor, LSTM, GRU

LISTA DE ILUSTRAÇÕES

Figura 1 – Demonstração do processo de escolha de vizinhos na KNN, com $k = 9$ . . 27
Figura 2 – Funções de distância que podem ser utilizadas com o KNN 28
Figura 3 – Funcionamento de um Perceptron 30
Figura 4 – Estrutura básica de uma rede neural. 31
Figura 5 – Comparação entre uma RNR (a) e uma RNA comum (b) 32
Figura 6 – Representação de uma célula de memória LSTM. 34
Figura 7 – Comparação entre uma célula LSTM e uma célula GRU 35
Figura 8 – Exemplo de função de distribuição acumulada 38
Figura 9 – Exemplo de análise gráfica de distribuição do valor F para populações de 1000 amostras, base de cálculo para obter o valor p. 39
Figura 10 – Classificação das técnicas de localização via RFID. 42
Figura 11 – Fases do sistema proposto por Xu e Chou (2017) durante a localização. . . 43
Figura 12 – Arquitetura de tratamento de dados do sistema proposto por Xu e Chou (2017). 44
Figura 13 – Planta representativa da casa de referência para a geração do *fingerprinting* 47
Figura 14 – *Channel monitor* 49
Figura 15 – *Heatmaps* medidos em ambiente real 50
Figura 16 – Interface do sistema Webots® 51
Figura 17 – Exemplo de cálculo do simulador de sinais RSS com 6 APs. O "Vetor de entrada RSS" obtém os sinais RSS medidos através do *fingerprinting* para a localização atual do robô para cada AP do ambiente de simulação. 53
Figura 18 – Barreira virtual do iRobot® dentro do ambiente de simulação, dispositivo utilizado para balizar o movimento do robô em certos pontos do ambiente, seja impedindo de passar por algum vão ou gerando uma área restrita radial, conforme indicado pela área em cinza. 54
Figura 19 – Descrição dos tipos de separação de dados. 59
Figura 20 – Exemplo de fluxo de dados no sistema MISO. 60
Figura 21 – Imagem do simulador em operação. As flechas vermelha, azul e verde definem o eixo de orientação no ambiente de simulação. 62
Figura 22 – Metodologia completa do método deste estudo. O código utilizado neste estudo está disponível na plataforma *GitHub*[7] 64

Figura 23 – Exemplo de gráfico de avaliação. 67
Figura 24 – Comparação dos resultados de predição para ao sistema KNN com diferentes valores k de vizinhos, para a base de dados com *stream* de dados, todos os APs, 1dBm de ruído, e 10% de máscara de atenuação. 69
Figura 25 – Resultado da progressão da acurácia e função de perdas durante o treinamento do sistema GRU para os dados de treinamento e validação em pacotes de 30 segundos de cada teste. 70
Figura 26 – Resultado da progressão da acurácia e função de perdas durante o treinamento do sistema LSTM para os dados de treinamento e validação em pacotes de 30 segundos de cada teste. 70
Figura 27 – Resultado da progressão da acurácia e função de perdas do treinamento do sistema GRU para os dados de treinamento e validação com *stream* de dados. 71
Figura 28 – Resultado da progressão da acurácia do treinamento do sistema LSTM para os dados de treinamento e validação com *stream* de dados. 72
Figura 29 – Resultado comparativo da FDA, utilizando pacotes de 30 segundos. 73
Figura 30 – Comparação estatística dos resultados, utilizando pacotes de 30 segundos. Na análise ANOVA, temos $F = 3539$ e $p = 0{,}0$, indicando que os resultados possuem desempenho significativamente diferentes. 74
Figura 31 – Exemplo de teste da RNR GRU, demonstrando a dificuldade de convergir a predição quando treinado com pacotes de 30 segundos. 74
Figura 32 – Resultado comparativo da FDA, utilizando *stream* de dados de 10 segundos. 75
Figura 33 – Comparação estatística dos resultados, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 1367$ e $p = 0{,}0$, indicando que os resultados possuem desempenho significativamente diferentes. 76
Figura 34 – Resultado do teste com modelo LSTM base. 76
Figura 35 – Resultado comparativo da FDA para os modelos específicos e testados com 3 APs em triangularização. 77
Figura 36 – Comparação estatística dos resultados para os modelos específicos, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 260{,}2$ e $p = 2{,}41e^{-113}$, indicando que os resultados possuem desempenho significativamente diferentes. 78

Figura 37 – Análise do resultado dos testes para o modelo específico LSTM e testados com 3 APs em triangularização, demonstrando o aumento de resposta do sistema após um KRP. 78
Figura 38 – Análise do resultado dos testes para o modelo específico KNN e testados com 3 APs em triangularização, apresentando dificuldade de em manter precisão. 79
Figura 39 – Resultado comparativo da FDA para os modelos base e testados com 3 APs em triangularização. 80
Figura 40 – Comparação estatística dos resultados para os modelos base, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 170{,}9$ e $p = 8{,}64e^{-75}$, indicando que os resultados possuem desempenho significativamente diferentes. 80
Figura 41 – Análise dos resultados para o modelo base LSTM e testado com 3 APs em triangularização, com o sistema sem nenhuma reação. 81
Figura 42 – Resultado comparativo da FDA para os modelos específicos e testados com 1 AP centralizado (AP 3). 82
Figura 43 – Comparação estatística dos resultados para os modelos específicos, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 1931$ e $p = 0.0$, indicando que os resultados possuem desempenho significativamente diferentes. 82
Figura 44 – Análise dos resultados para o modelo específico GRU com o robô próximo ao AP 3 centralizado. 83
Figura 45 – Resultado comparativo da FDA para os modelos base e testados com 1 AP centralizado (AP 3). 84
Figura 46 – Comparação estatística dos resultados para os modelos base, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 711{,}5$ e $p = 7{,}97e^{-307}$, indicando que os resultados possuem desempenho significativamente diferentes. 84
Figura 47 – Análise dos resultados para o modelo base GRU com 1 AP centralizado (AP 3), incapaz de se localizar. 85
Figura 48 – Resultado do teste para o modelo específico LSTM com o robô ficando próximo do AP 4 após o KRP. 86

Figura 49 – Comparação estatística dos resultados para os modelos específicos, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 1281$ e $p = 0{,}0$, indicando que os resultados possuem desempenho significativamente diferentes. 87
Figura 50 – Resultado comparativo da FDA para os modelos específicos e testados com um único AP na extremidade (AP 4). 87
Figura 51 – Resultado comparativo da FDA para os modelos base e testados com um único AP na extremidade (AP 4). 88
Figura 52 – Comparação estatística dos resultados para os modelos base, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 132{,}6$ e $p = 3{,}19e^{-58}$, indicando que os resultados possuem desempenho significativamente diferentes. 89
Figura 53 – Resultado do teste para o modelo base LSTM com um único AP na extremidade (AP 4), demonstrando uma condição limite onde o sistema não consegue prever a localização. 89
Figura 54 – Resultado do teste para o modelo base LSTM com um único AP na extremidade (AP 4), demonstrando um teste em que o sistema não reage ao KRP. 90
Figura 55 – Resultado comparativo da FDA para os modelos específicos e testados com um único AP na extremidade (AP 1). 91
Figura 56 – Comparação estatística dos resultados para os modelos específicos, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 1806$ e $p = 0{,}0$, indicando que os resultados possuem desempenho significativamente diferentes. 91
Figura 57 – Detalhe de movimentos que influenciaram a precisão dos sistemas de localização utilizando modelo específico nos testes com apenas o AP 1. 92
Figura 58 – Resultado do teste para o modelo base LSTM com um único AP na extremidade (AP 1), demonstrando o atraso para relocalização após o KRP. . . . 92
Figura 59 – Resultado do teste para o modelo específico KNN com um único AP na extremidade (AP 1), indicando flutuação na predição. 93
Figura 60 – Resultado comparativo da FDA para os modelos base e testados com um único AP na extremidade (AP 1). 94

Figura 61 – Comparação estatística dos resultados para os modelos base, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 500{,}9$ e $p = 8{,}07e^{-217}$, indicando que os resultados possuem desempenho significativamente diferentes. 94
Figura 62 – Resultado do teste para o modelo específico LSTM e testado com 2 dBm de ruído, com pequena instabilidade na predição. 95
Figura 63 – Comparação estatística dos resultados para os modelos específicos, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 132{,}7$ e $p = 3{,}07e^{-58}$, indicando que os resultados possuem desempenho significativamente diferentes. 96
Figura 64 – Resultado comparativo da FDA para os modelos específicos e testados com 2 dBm de ruído. 96
Figura 65 – Resultado do teste para o modelo base LSTM e testado com 2 dBm de ruído, demonstrando melhores resultados do que o modelo específico. . . . 97
Figura 66 – Comparação estatística dos resultados para os modelos base, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 7899$ e $p = 0{,}0$, indicando que os resultados possuem desempenho significativamente diferentes. 98
Figura 67 – Resultado comparativo da FDA para os modelos base e testados com 2 dBm de ruído. 98
Figura 68 – Exemplo comparativo de resultado do teste com 2 dBm para ambos os modelos KNN. A esquerda está o teste realizado com o modelo base. A direita, o teste realizado com o modelo específico. 99
Figura 69 – Resultado do teste para o modelo base GRU e testado com 2 dBm de ruído, demonstrando o atraso para relocalização após o KRP. 99
Figura 70 – Resultado do teste para o modelo específico LSTM e testado com 10 dBm de ruído, apresentando comportamento semelhante ao KNN, oscilando em torno da localização correta. 101
Figura 71 – Comparação estatística dos resultados para os modelos específicos, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 2604$ e $p = 0{,}0$, indicando que os resultados possuem desempenho significativamente diferentes. 101

Figura 72 – Resultado comparativo da FDA para os modelos específicos e testados com 10 dBm de ruído. 102
Figura 73 – Resultado do teste para o modelo base KNN e testado com 10 dBm de ruído, apresentando alto desvio na predição. 103
Figura 74 – Resultado comparativo da FDA para os modelos base e testados com 10 dBm de ruído. 103
Figura 75 – Comparação estatística dos resultados para os modelos base, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 1799$ e $p = 0{,}0$, indicando que os resultados possuem desempenho significativamente diferentes. 104
Figura 76 – Resultado do teste para o modelo base LSTM e testado com 10 dBm de ruído, demonstrando a dificuldade do sistema em convergir. 104
Figura 77 – Resultado do teste para o modelo base GRU e testado com 10 dBm de ruído, demonstrando a dificuldade do sistema em convergir. 105
Figura 78 – Resultado do teste para o modelo específico KNN e testado com 20% de atenuação, com deslocamentos pontuais na predição. 106
Figura 79 – Resultado do teste para o modelo específico LSTM e testado com 20% de atenuação, com dificuldade de convergir após um KRP. 107
Figura 80 – Comparação estatística dos resultados para os modelos específicos, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 61{,}82$ e $p = 1{,}49e^{-27}$, indicando que os resultados possuem desempenho significativamente diferentes. 107
Figura 81 – Resultado comparativo da FDA para os modelos específicos e testados com 20% de atenuação. 108
Figura 82 – Resultado do teste para o modelo base GRU e testado com 20% de atenuação, apresentando ainda boa precisão e tempo de resposta, sendo pouco prejudicado pela atenuação. 109
Figura 83 – Comparação estatística dos resultados para os modelos base, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 3165$ e $p = 0{,}0$, indicando que os resultados possuem desempenho significativamente diferentes. 109
Figura 84 – Resultado comparativo da FDA para os modelos base e testados com 20% de atenuação. 110

Figura 85 – Resultado do teste para o modelo específico LSTM e testado com 30% de atenuação, apresentando atraso na detecção do KRP. 111
Figura 86 – Resultado do teste para o modelo específico KNN e testado com 30% de atenuação, com flutuação na predição, mas resposta rápida ao KRP. 111
Figura 87 – Comparação estatística dos resultados para os modelos específicos, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 74{,}39$ e $p = 5{,}29e^{-33}$, indicando que os resultados possuem desempenho significativamente diferentes. 112
Figura 88 – Resultado comparativo da FDA para os modelos específicos e testados com 30% de atenuação. 112
Figura 89 – Resultado do teste para o modelo base LSTM e testado com 30% de atenuação, mantendo boa precisão nas predições. 113
Figura 90 – Comparação estatística dos resultados para os modelos base, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 7902$ e $p = 0{,}0$, indicando que os resultados possuem desempenho significativamente diferentes. 114
Figura 91 – Resultado do teste para o modelo base KNN e testado com 30% de atenuação, com alta oscilação nas predições. 114
Figura 92 – Resultado comparativo da FDA para os modelos específicos e testados com 30% de atenuação. 115
Figura 93 – Resultados obtidos por Hoang et al. (2019) para sistemas GRU e LSTM em gráfico de função de distribuição acumulada 118
Figura 94 – Resultados obtidos em função de distribuição acumulada para os diversos experimentos deste estudo. Os resultados com "_O" na legenda indicam que foi utilizado o modelo treinado com os parâmetros base deste estudo. . 119

LISTA DE TABELAS

Tabela 1 – Disposição dos APs no ambiente de referência 48
Tabela 2 – Parâmetros de configuração iniciais para o sistema LSTM 61
Tabela 3 – Exemplos de combinações da APs para teste 63
Tabela 4 – Dados capturados pelo log de avaliação 65
Tabela 5 – Parâmetros de teste avaliados no simulador para treinamento dos modelos base 65

LISTA DE ABREVIATURAS

AMCL	*Adaptative Monte Carlo Localization*
ANN	*Artificial Neural Network*
ANOVA	*Analysis of Variance*
BLE	*Bluetooth Low Energy*
CPU	*Central Processing Unit*
CSV	*Comma Separated Values*
dBm	Decibel-miliwatts
FDA	Função de Distribuição Acumulada
GPS	*Global Positioning System*
GPU	*Graphics Processing Unit*
GRU	*Gated Recurrent Unit*
HDF5	*Hierarchical Data Format 5*
IDF	*IoT Development Framework*
IEEE	*Institute of Electrical and Electronics Engineers*
KNN	*k-Nearest Neighbors*
KRP	*Kidnapped Robot Problem*
LRF	*Laser Range Finder*
LSTM	*Long-Short Term Memory*
Mbps	*Mega bits per second*
MCL	*Monte Carlo Localization*
MIMO	*Multiple Input Multiple Output*
MISO	*Multiple RSSI input, Single Location Output*
MLR	*Multiple Linear Regression*
mW	Miliwatts
NPY	Numpy
PC	*Personal Computer*
PCB	*Printed circuit board*
RAM	*Random Access Memory*
RFID	*Radio frequency identification*
RNA	Redes Neurais Artificiais
RNN	*Recurrent Neural Network*

RNP	Rede Neural Profunda
RNR	Redes Neurais Recorrentes
RSS	*Received Signal Strength*
RSSI	*Received Signal Strength Indicator*
SLAM	*Simultaneous Localization And Mapping*
Wi-Fi	*Wireless Fidelity*

SUMÁRIO

1 Introdução . . . 19
1.1 Objetivo . . . 20
1.2 Motivação . . . 20
2 Conceitos Fundamentais . . . 22
2.1 Problema do Robô Sequestrado . . . 22
2.2 Wi-Fi . . . 22
2.2.1 ***Access Points*** . . . 23
2.2.2 ***Received Signal Stregth*** **(RSS)** . . . 23
2.2.2.1 **Received Signal Stength Indicator** ***(RSSI)*** . . . 24
2.3 Modelo de Perda de Propagação . . . 25
2.4 *Fingerprinting* . . . 25
2.5 *k-Nearest Neighbors* (KNN) . . . 26
2.5.1 Aprendizagem preguiçosa . . . 27
2.5.2 Funções de distância . . . 28
2.5.2.1 Aplicação de pesos na predição . . . 28
2.5.2.2 Distância Euclidiana . . . 29
2.5.3 Seleção do valor de k . . . 29
2.5.4 Função de Predição . . . 29
2.6 Redes Neurais Artificiais . . . 30
2.6.1 Redes Neurais Recorrentes (RNR) . . . 31
2.6.1.1 Long Short-Term Memory (LSTM) . . . 32
2.6.1.2 Gated Recurrent Unit (GRU) . . . 34
2.6.2 Treinamento . . . 36
2.6.2.1 Treinamento Supervisionado . . . 36
2.6.3 Validação . . . 37
2.7 Função de Distribuição Acumulada (FDA) . . . 37
2.8 *Analysis of Variance* (*Analysis of Variance* (ANOVA)) . . . 38
3 Revisão Bibliográfica . . . 40
3.1 *Received Signal Strength* (RSS) . . . 40
3.2 Sequestro de robôs . . . 42
3.3 Combinações . . . 44

4 Metodologia . . . 46
4.1 Ambiente de Referência . . . 46
4.2 *Access Points* e Wi-Fi *fingerprinting* . . . 46
4.3 Ambiente Webots® . . . 49
4.3.1 Supervisório de atuação do sistema Webots® . . . 51
4.4 Simulador de sinais de RSS . . . 52
4.4.1 Sistema de atenuação . . . 52
4.4.2 Sistema de ruído . . . 53
4.5 Gerador de caminhos . . . 53
4.5.1 Controlador do robô *iRobot® Create* . . . 54
4.6 Procedimento de geração de dados de treinamento, validação e teste no ambiente de simulação . . . 55
4.7 Preparação da base de dados para treinamento das RNRs . . . 56
4.7.1 Separação de dados em pacotes de 30 segundos de cada teste . . . 57
4.7.2 Separação de dados em *stream* de dados de 10 segundos . . . 58
4.8 Sistema de localização KNN . . . 59
4.9 Sistema de localização LSTM . . . 60
4.10 Sistema de localização GRU . . . 60
4.11 Rotina de Testes . . . 61
4.12 Parâmetros de teste e variações . . . 65
4.12.1 Gráficos de função de distribuição acumulada . . . 66
4.12.2 Gráficos de avaliação dos testes . . . 66
5 Resultados experimentais . . . 68
5.1 Treinamento do sistema KNN . . . 68
5.2 Resultados de treinamento e progressão dos valores de acurácia e função de perdas das RNRs . . . 68
5.2.1 Separação de dados em pacotes de 30 segundos de cada teste . . . 69
5.2.1.1 Treinamento do sistema GRU . . . 69
5.2.1.2 Treinamento do sistema LSTM . . . 70
5.2.2 Separação de dados em *stream* de dados de 10 segundos . . . 71
5.2.2.1 Treinamento do sistema GRU . . . 71
5.2.2.2 Treinamento do sistema LSTM . . . 71

5.3 Análise da progressão e limite de estabilização do treinamento para o *stream* de dados 72
5.4 Resultados de performance com variação na composição dos dados apresentados 73
5.4.1 Separação de dados em pacotes de 30 segundos de cada teste 73
5.4.2 Separação de dados em *stream* de dados de 10 segundos 75
5.5 Resultados de performance com variação na quantidade de APs disponíveis . . 77
5.5.1 Simulador operando com 3 APs em triangularização 77
5.5.1.1 Utilizando modelo específico treinado com os mesmos parâmetros dos dados de teste 77
5.5.1.2 Utilizando modelo base treinado com os parâmetros* stream *de dados, todos os APs disponíveis, 1dBm de ruído e 10% de atenuação 79
5.5.2 Simulador operando com um único AP centralizado no ambiente (AP 3) . 81
5.5.2.1 Utilizando modelo específico treinado com os mesmos parâmetros dos dados de teste 81
5.5.2.2 Utilizando modelo base treinado com os parâmetros* stream *de dados, todos os APs disponíveis, 1dBm de ruído e 10% de atenuação 83
5.5.3 Simulador operando com um único AP localizado em uma extremidade do ambiente (AP 4) 85
5.5.3.1 Utilizando modelo específico treinado com os mesmos parâmetros dos dados de teste 85
5.5.3.2 Utilizando modelo base treinado com os parâmetros* stream *de dados, todos os APs disponíveis, 1dBm de ruído e 10% de atenuação 88
5.5.4 Simulador operando com um único AP localizado em uma extremidade do ambiente (AP 1) 90
5.5.4.1 Utilizando modelo específico treinado com os mesmos parâmetros dos dados de teste 90
5.5.4.2 Utilizando modelo base treinado com os parâmetros* stream *de dados, todos os APs disponíveis, 1dBm de ruído e 10% de atenuação 93
5.6 Resultados de performance com variação nos níveis de ruído simulado 95
5.6.1 Simulador operando com 2dBm médio de ruído gaussiano 95
5.6.1.1 Utilizando modelo específico treinado com os mesmos parâmetros dos dados de teste 95

5.6.1.2 ***Utilizando modelo base treinado com os parâmetros* stream *de dados, todos os APs disponíveis, 1dBm de ruído e 10% de atenuação*** 97
5.6.2 Simulador operando com 10dBm médio de ruído gaussiano 100
5.6.2.1 ***Utilizando modelo específico treinado com os mesmos parâmetros dos dados de teste*** 100
5.6.2.2 ***Utilizando modelo base treinado com os parâmetros* stream *de dados, todos os APs disponíveis, 1dBm de ruído e 10% de atenuação*** 102
5.7 Resultados de performance com variação nos níveis de atenuação simulado . . 105
5.7.1 Simulador operando com 20% de atenuação independente 105
5.7.1.1 ***Utilizando modelo específico treinado com os mesmos parâmetros dos dados de teste*** 106
5.7.1.2 ***Utilizando modelo base treinado com os parâmetros* stream *de dados, todos os APs disponíveis, 1dBm de ruído e 10% de atenuação*** 108
5.7.2 Simulador operando com 30% de atenuação independente 110
5.7.2.1 ***Utilizando modelo específico treinado com os mesmos parâmetros dos dados de teste*** 110
5.7.2.2 ***Utilizando modelo base treinado com os parâmetros* stream *de dados, todos os APs disponíveis, 1dBm de ruído e 10% de atenuação*** 113
6 Discussões e Conclusões 116
6.1 Variação na densidade de *access points* 116
6.2 Variação no nível de ruído 117
6.3 Variação no nível de atenuação 117
6.4 Observações finais 118
7 Trabalhos futuros 120
REFERÊNCIAS 121

1 Introdução

Com a popularização de sistemas embarcados, devido à redução dos custos de componentes, incluindo microcontroladores com crescente capacidade computacional e boa eficiência energética, baterias de longa duração e sistemas de comunicação e sensoriamento de baixa potência, a cada dia há mais robôs migrando de aplicações estritamente comerciais e industriais para compor soluções de casas inteligentes. Nesse escopo, para obter bons resultados na interação dos robôs com o ambiente e as pessoas presentes, o rastreamento constante e rápido de sua localização durante o funcionamento é um fator importante para seu funcionamento em geral.

Localização de robôs em ambientes internos é um grande desafio para sistemas de robôs autônomos (ZAFARI; GKELIAS; LEUNG, 2019), devido a impossibilidade de uso do GPS em ambientes internos, onde o sinal não opera normalmente, complexidade de estrutural de tais ambientes, incluindo salas e objetos, além das constantes mudanças que ocorrem em tais ambientes. A complexidade das informações, atreladas a capacidades limitadas de potência e hardware para processamento durante a realização de cálculos complexos em tempo hábil são limitações que devem ser consideradas antes que o sistema tenha sua eficácia reduzida antes do robô se localizar corretamente.

Quando considerada a portabilidade desses sistemas robóticos, um novo problema é apresentado, o sequestro de robôs, quando tais robôs são movidos manualmente pelos usuários sem nenhuma notificação ao sistema de localização, interferindo no processo. Existem vários estudos buscando solucionar o problema utilizando sinais de Wi-Fi como base de obtenção de dados, cada um com seus pontos positivos e negativos, entre disponibilidade, custos, complexidade e precisão (ZAFARI; GKELIAS; LEUNG, 2019). Mesmo existindo diversas técnicas que já permitem resolver o problema, utilizando tecnologias como *Laser Range Finder* (LRF) (SEOW et al., 2017) e análise de imagens por câmera (XU; CHOU; DONG, 2019) e *Radio frequency identification* (RFID) (LI; MO; ZHANG, 2019), tais soluções exigem sensoriamento adicional específico para a realizar a localização, complicando arquiteturas de projeto, consumindo mais energia e ocupando portas de comunicação com o sistema de controle ou bandas de comunicação que poderiam ser utilizadas para outras funcionalidades dos robôs a serem localizados.

Apesar do crescimento da área, ainda existem grandes desafios relacionados ao sensoriamento e localização de robôs com precisão e velocidade em locais fechados (ZAFARI; GKELIAS; LEUNG, 2019), principalmente quando considerados os desafios ao projetar siste-

mas embarcados, quando limitações em capacidade de processamento devido ao uso de baterias e dimensões para manter sua aplicabilidade. Um bom recurso para suprimir o problema de sequestro de robôs em ambientes fechados, onde o GPS não consegue atuar é um exemplo desse desafio. Sistemas residenciais também não podem exigir uma ampla gama de hardware específico distribuído pelo ambiente para implementação, pois podem demandar, além de reformas proibitivas para sua instalação, suporte de instalação e manutenção especializada.

Dentre os diversos padrões de soluções de comunicação por radiofrequência existentes, o Wi-Fi foi escolhido como o foco deste estudo, por ser uma tecnologia importante para a aplicação de soluções em internet das coisas e devido a sua disseminação em diversos tipos de ambientes internos, desde industriais, até residenciais.

1.1 Objetivo

O objetivo deste trabalho é avaliar os impactos da quantidade de *Access Points* (APs), níveis de ruído e níveis de atenuação em sistemas de localização de robôs que utilizam sinais RSS em Redes Neurais Recorrentes GRU (CHO et al., 2014) e LSTM (HOCHREITER; SCHMIDHUBER, 1997) e em um sistema de aprendizado de máquina KNN (RUSSELL; NORVIG, 2009) como propostas para solucionar o problema do robô sequestrado.

Realiza-se uma comparação de performance de cada sistema proposto utilizando gráficos de função de distribuição acumulada e análises pontuais de comportamentos observados durante os testes para as soluções avaliadas atuando em diversas situações envolvendo os parâmetros propostos de teste, como agrupamento de dados, combinações de *access points*, níveis de ruído e de atenuação aplicados no simulador.

1.2 Motivação

As técnicas de localização por Wi-Fi e RSS, apesar de desafiadoras, permitem uma larga gama de aplicações por utilizar uma interface de comunicação presente em grande parte dos dispositivos móveis. O maior benefício de utilizar RSS consiste no fato de não exigir nenhuma estrutura adicional de sensores ou atuadores (ZÀRUBA et al., 2007), apresentando-se como uma solução viável para aplicações de simples implementação.

El-Naggar, Wassal e Sharaf (2019) demonstram que uma única referência de Wi-Fi não permite a construção de um sistema de localização via RSS com confiabilidade, exigindo ao me-

nos 3 referências para realizar uma triangularização, mas a relação entre precisão, velocidade de localização e a quantidade de APs pode variar em função da técnica adotada para tratamento dos dados. Em (ABBAS et al., 2019) e (ZÀRUBA et al., 2007) esse aspecto é estudado dentro de suas próprias técnicas de tratamento, obtendo resultados diferentes. Por outro lado, Sadowski e Spachos (2019) chegaram a conclusão após realizarem estudos com *Bluetooth Low Energy* (BLE), que o excesso de pontos de comunicação de radiofrequência podem congestionar os canais de comunicação, piorando os resultados. Devido a essas variações, é importante compreender o volume ideal de APs em cada ambiente para obter o melhor desempenho de cada sistema de localização.

No estudo de Tian et al. (2018), conclui-se que a definição da localização por *fingerprinting* é um problema *NP-Hard* e exige técnicas específicas para obter resultados em tempo hábil, portanto, um estudo prévio do impacto da quantidade de APs na precisão e no tempo de localização será importante para balancear o dimensionamento de sistemas em trabalhos futuros para obter os melhores resultados possíveis. Xue et al. (2019) define uma estratégia de seleção de APs para reduzir a carga computacional que a análise de múltiplos APs pode gerar no sistema, mas ainda depende de medir o RSS de todos os APs no alcance e realizar cálculos de entropia a cada medição para concluir a seleção.

2 Conceitos Fundamentais

Os estudos na área de localização de robôs em ambientes fechados baseiam-se em diversos conceitos essenciais sobre as características do sistema de comunicação sem fio por radio frequência até técnicas que possibilitam o processamento em sistemas embarcados. Nos próximos tópicos serão citados os conceitos considerados neste estudo.

2.1 Problema do Robô Sequestrado

O problema do robô sequestrado, ou *Kidnapped Robot Problem* (KRP) (FOX et al., 2001) acontece quando um robô que já se localizou em certo ponto do ambiente é teleportado para outra posição sem que ele seja avisado. Tal fenômeno pode acontecer por diversos fatores, como falta de energia, manipulação pelo usuário, falha no sensoriamento, ou qualquer outra situação onde o robô fica sem obter dados para localizar-se por um período, e quando retoma o funcionamento normal do sistema de localização, está em uma posição e/ou orientação diferente da última obtida.

É um problema específico pois o robô possui convicção de sua localização anterior no momento do sequestro e é amplamente estudado devido a dificuldade de identificação e realização de uma nova localização em tempo satisfatório. Um bom método para avaliação de sua capacidade de recuperação após uma situação catastrófica, motivo pelo qual este será o principal conceito estudado neste trabalho.

2.2 Wi-Fi

Amplamente utilizado em sistemas de compartilhamento de internet sem fio, o Wi-Fi (definido pela IEEE 802.11) é um padrão de comunicação por rádio frequência padronizado pelo *Wi-Fi Alliance* em 1999 (PERAHIA, 2008) para redes de computadores e escolhido como sistema de radiofrequência de referência para a realização das simulações, devido a sua ampla adoção como sistema de comunicação e compartilhamento de internet para dispositivos móveis residenciais e comerciais, tornando interessantes os estudos de localização utilizando o padrão, pois demanda menores aquisições de hardware dedicado para implementação e a quantidade e diversidade de dispositivos disponíveis mantém os preços dos componentes baixos.

O Wi-Fi já sofreu diversas modificações e otimizações durante seus mais de 20 anos de existência (PERAHIA, 2008), incluindo sistemas de certificação e criptografia de dados, melhorias em velocidade de comunicação e alcance de sinal, gerando um grande número de subpadrões. Este estudo focará no padrão IEEE 802.11n, devido a sua maturidade, adoção e variedade de dispositivos certificados.

O IEEE 802.11n (PERAHIA, 2008) foi o primeiro padrão Wi-Fi a adotar velocidades de comunicação acima dos 100Mbps, alcançando o padrão ethernet, devido à introdução no padrão da tecnologia *Multiple Input Multiple Output* (MIMO) com múltiplas antenas, que facilita a utilização de diversas subfrequências em paralelo.

2.2.1 *Access Points*

Access Points ou "pontos de acesso", são dispositivos agregadores de comunicação, *hosts* que coordenam os parâmetros base da comunicação, como padrão de comunicação, canal de frequência, método de autenticação, criptografia e velocidade de comunicação. Os roteadores *wireless* são um exemplo de equipamentos com funcionalidade de APs do padrão Wi-Fi, coordenando as comunicações entre o mesmo e os clientes, enquanto compartilha o acesso à internet e implementa protocolos de comunicação entre os dispositivos conectados.

2.2.2 *Received Signal Stregth* (RSS)

RSS (*Received Signal Strength*) é a potência do sinal recebido pelo receptor de radio frequência, medido em Decibel-miliwatts (dBm) ou Miliwatts (mW) (ZAFARI; GKELIAS; LEUNG, 2019). É um processo presente em diversos sistemas de comunicação sem fio, como ZigBee® (DING; DONG, 2020), Bluetooth® (SADOWSKI; SPACHOS, 2019), LoRa® (DARAMOUSKAS; KAPOULAS; PARASKEVAS, 2019) e até telefonia celular (YANG; ZHOU; LIU, 2013) e é, desconsiderando interferências e outras perturbações, inversamente proporcional a distância entre o emissor e o receptor (EL-NAGGAR; WASSAL; SHARAF, 2019).

Devido a possibilidade de realizar uma relação entre distância e RSS, o tópico já foi amplamente estudado em diversas aplicações para sistemas de localização em ambientes fechados, mas devido a sua característica de facilmente sofrer variações causadas por fatores não triviais, como efeitos de multi-propagação, pontos cegos, ruídos e interferência (LADD et al., 2005), apresenta até hoje um grande desafio interpretar os dados de forma precisa e eficaz. Chapre

et al. (2013) detalham outros fatores previsíveis que podem causar variações nos sinais de RSS, como:

a) Orientação, tipo e direcionalidade da antena;
b) Distância entre receptor e emissor (AP);
c) Horário e período de medição dos sinais;
d) Interferência de sinais de radiofrequência operando no mesmo ou em outro canal;
e) Presença e movimentação de pessoas;
f) Características do ambiente, como estrutura, objetos e materiais empregados.

O estudo de Yiu et al. (2017) também identificou perturbações no sinal RSS devido ao conflito de pacotes em curtas distâncias, chegando a quedas de sinal entre 20dB e 50dB.

A precisão dos sistemas de localização baseados em RSS pode ser afetada pela quantidade de pontos de referência disponíveis (ABBAS et al., 2019) (SADOWSKI; SPACHOS, 2019), exigindo, com modelos conhecidos, uma triangularização (EL-NAGGAR; WASSAL; SHARAF, 2019) ou até múltiplos pontos de referência para atingir certa precisão. Devido a grande variação do comportamento dos sistemas que utilizam RSS como dados de referência para determinar localização, a análise deste trabalho é focada em seu impacto para alguns sistemas existentes.

Além das potências dos sinais poderem ser utilizadas para estimar a distância aproximada entre o emissor e o receptor do sinal a partir de diversos modelos, também existe a possibilidade das estimativas de RSS para cada localização serem, determinadas a partir do processo de *fingerprinting*, onde das potências estimadas dos sinais são medidas previamente no ambiente e mapeadas (TIAN et al., 2018).

2.2.2.1 Received Signal Stength Indicator *(RSSI)*

Received Signal Strength Indicator (RSSI), ou RSS Indicator (Indicador de Potência de Sinal Recebido em tradução livre), é um indicador arbitrário relativo à medição do RSS, que normalmente é definido pelo fabricante do chip Wi-Fi. Por exemplo, enquanto a Atheros® utiliza valores de RSSI entre 0 e 60, a Cisco® utiliza RSSI entre 0 e 100 (ZAFARI; GKELIAS; LEUNG, 2019).

2.3 Modelo de Perda de Propagação

É um modelo dos sinais do ambiente onde o robô circulará. Utilizando RSS e um Modelo de Perda de Propagação do ambiente (KUMAR; REDDY; VARMA, 2009), a distância d entre emissor e receptor pode ser estimada com a fórmula (1) onde n é o fator de perda de propagação, que pode variar entre 2 para ambientes abertos e 4 para ambientes fechados, e A é o valor da distância de referência do receptor até o emissor. Apesar de simples implementação, o Modelo de Perda de Propagação é difícil de ser implementado com precisão devido ao grande número de variações que os ambientes internos podem sofrer, mas a teoria pode ser aplicada entre pontos de medição (ZÀRUBA et al., 2007), facilitando o processo de *fingerprinting* deste trabalho, se selecionados pontos de medição estratégicos em função do ambiente em questão.

$$RSSI = -10 * n * Log_{10}(d) + A \tag{1}$$

2.4 *Fingerprinting*

Será utilizado neste trabalho uma alternativa ao Modelo de Perda de Propagação, onde ao invés de calcular um modelo que possa representar todo o ambiente, é realizado um mapeamento do mesmo. Chamada de *fingerprinting*, a técnica atraiu grande interesse da comunidade científica após o trabalho que demonstrou seu funcionamento através do sistema RADAR nos anos 2000 (BAHL; PADMANABHAN, 2000 apud LAOUDIAS et al., 2018).

O processo de obtenção de dados para utilizar o *fingerprinting* é mais trabalhoso (XUE et al., 2019), pois consiste em um passo adicional prévio, onde avalia-se os valores de sinal RSS em diversos pontos de referência do ambiente, mas em contrapartida, o resultado final apresenta um mapa dos sinais de todos os APs, já considerando parte de suas atenuações e interferências. A solução apresenta-se como uma melhor opção frente ao modelo calculado de propagação. Esse primeiro passo pode ser chamado de fase de treinamento ou fase offline (TIAN et al., 2018), onde será incluído neste trabalho o processo de treinamento da LSTM e da GRU para geração da matriz de pesos.

Na segunda fase, chamada de fase de localização ou fase online, os dados de referência da fase offline são comparados com os dados medidos em tempo real pelo robô para obtermos a localização do mesmo (TIAN et al., 2018). Esta fase exige técnicas complexas e podem apresentar diferentes resultados dependendo da técnica utilizada, pois mesmo com uma prévia

avaliação do ambiente, o sinal pode ser afetado por outros fatores dinâmicos, como movimentação de pessoas e objetos, além de sofrer interferência de outros sistemas de radiofrequência que não estejam representados durante as medições (CHAPRE et al., 2013).

2.5 *k-Nearest Neighbors* (KNN)

O sistema de aprendizado de máquina k-Nearest Neighbors (KNN), ou k-ésimo vizinho mais próximo, é definida por Al-Dosary, Al-Hamed e Aboukarima (2019) como uma técnica de fácil implementação e baixo esforço computacional que pode realizar classificações e regressões. Um método não paramétrico que apesar de sua simplicidade, pode ser bem efetivo para realização de predições (GUO et al., 2003). Guo et al. (2003) indica também que o resultado das predições são extremamente dependentes da escolha apropriada da quantidade k de vizinhos para cada proposta de predição desenvolvida.

Quando utilizada para realização de regressões, a técnica tem capacidade de atingir resultados melhores que a regressão linear múltipla, ou *Multiple Linear Regression* (MLR) (AL-DOSARY; AL-HAMED; ABOUKARIMA, 2019). Souza et al. (2020) também concluíram em seu estudo de predição de fator de forma artificial de árvores do híbrido *Eucalyptus urophylla* e *Eucalyptus grandis* que a KNN tem melhor capacidade de generalização que o método de Mínimos Quadrados Ordinário.

A predição é realizada utilizando um método de aprendizagem preguiçosa, a partir do cálculo das distâncias entre o dado a ser avaliado e cada dado da base de referência, que é mantida por completo para a realização de cada predição (GUO et al., 2003). A seleção dos k dados de referência que ficaram mais próximos do dado apresentado serão utilizados como base para obter a predição do modelo. Devido à tais características, KNN também é conhecido por ser um método de predição puramente preguiçoso (WETTSCHERECK; AHA; MOHRI, 1997).

A figura 1 demonstra graficamente um processo de escolha de vizinhos utilizando KNN para realizar uma classificação, com a base de dados de treinamento demonstrada na figura 1 (a), enquanto na figura 1 (b) o círculo branco marca a área onde o KNN identificou os vizinhos próximos do ponto d_i para classificação. O procedimento para seleção de vizinhos é igual ao utilizado na KNN para regressão, mudando apenas a função de predição.

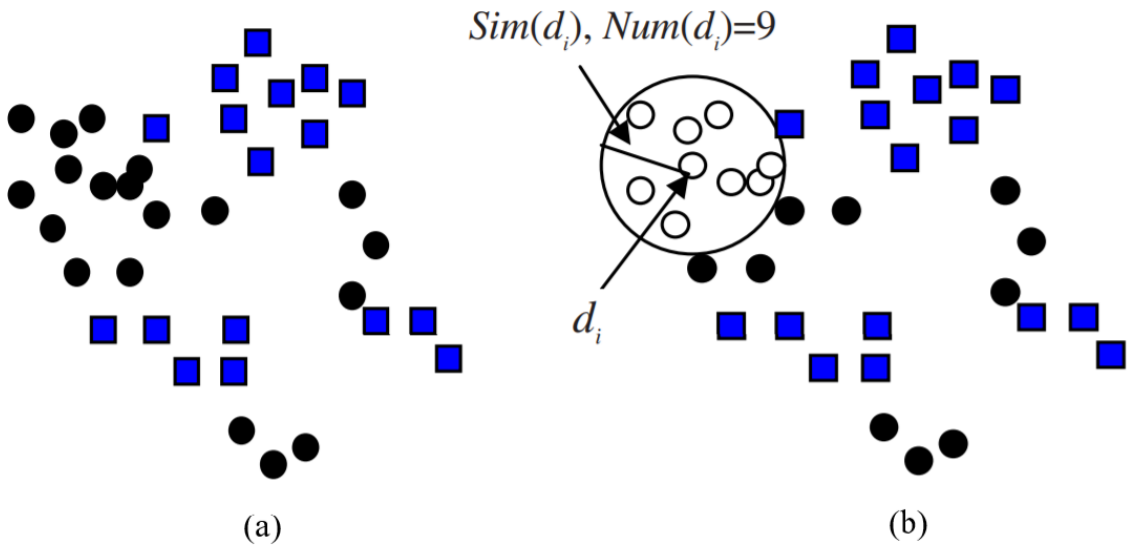

Figura 1 – Demonstração do processo de escolha de vizinhos na KNN, com $k = 9$

Fonte: Guo et al., 2003.

2.5.1 Aprendizagem preguiçosa

O método de aprendizagem preguiçosa, ou *Lazy Learning*, é definido por Wettschereck, Aha e Mohri (1997) como um método que apresenta as seguintes características:

a) Adiamento: Todos os dados de treinamento são armazenados e seu processamento é postergado até o momento da predição;
b) Replicação: Predições são feitas combinando a entrada com os dados de treinamento, onde normalmente as instâncias são definidas como pontos em um espaço definido, com uma função de similaridade e uma função de predição definidas;
c) Descarga: Depois de realizada a predição, os resultados obtidos e quaisquer outros dados processados são descartados.

O método não exige o treinamento prévio de um modelo, mas a apresentação dos dados de treinamento sem nenhum pré-processamento pode enviesar as predições quando a base de treinamento possui dados redundantes, irrelevantes, ruidosos ou dependentes, pois todos os dados possuem a mesma representatividade para realização das predições (WETTSCHERECK; AHA; MOHRI, 1997).

2.5.2 Funções de distância

Para realização da predição no sistema KNN é necessário calcular a distância de cada conjunto de valores na base de dados de treinamento para os valores observados de entrada. Diversas técnicas de cálculo que podem ser utilizadas com o KNN foram demonstradas por Wilson e Martinez (2000) e estão demonstradas na figura 2.

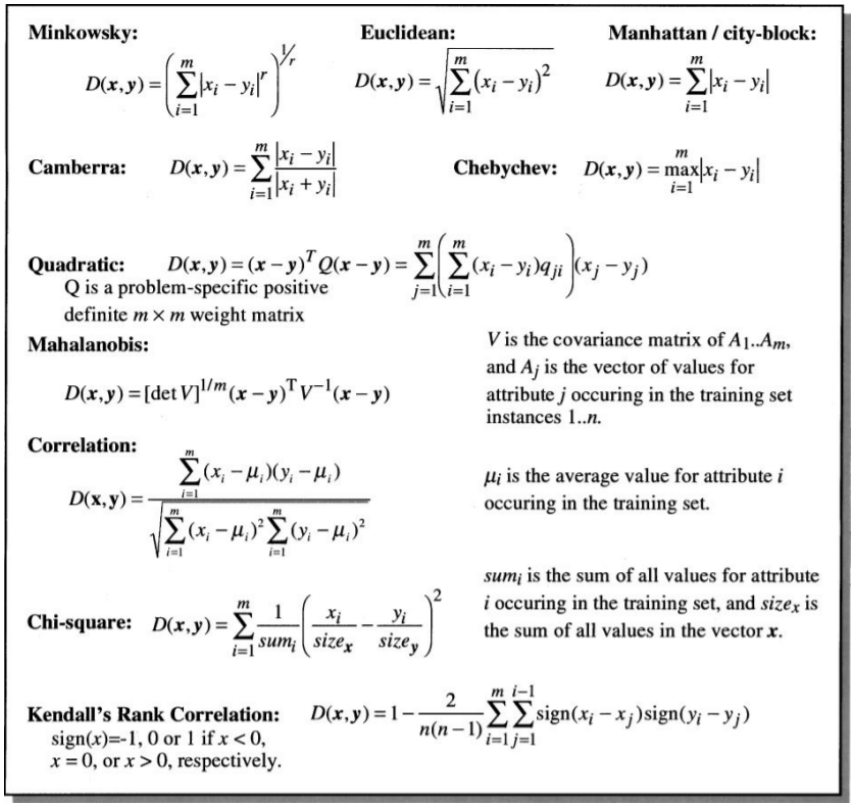

Figura 2 – Funções de distância que podem ser utilizadas com o KNN

Fonte: Wilson e Martinez, 2000.

2.5.2.1 Aplicação de pesos na predição

Algumas soluções desenvolvidas para cálculo de distância na predição KNN envolve determinar pesos para cada vizinho conforme a distância de cada um para o resultado final (WETTSCHERECK; AHA; MOHRI, 1997). O peso calculado para cada vizinho mais próximo da base de treinamento determinará qual sua representatividade no cálculo da regressão para determinação da predição.

2.5.2.2 Distância Euclidiana

Neste trabalho o método de cálculo de distância escolhido será a distância Euclidiana, baseado nos métodos utilizados no trabalho de referência, onde a distância d_i é calculada a partir da fórmula 2 determinada por Xu e Chou (2017):

$$d_i = \sqrt{\sum_{j=1}^{n} (\bar{S}_i^j - \bar{S}^j)^2} \tag{2}$$

Na fórmula 2 adaptada para este estudo, $\bar{S}_i^j$ representa os valores de RSS para cada cada AP j de cada dado de treinamento i utilizado no treinamento dos modelos das RNRs para obter comparabilidade entre as soluções. A fórmula 2 original de Xu e Chou (2017) considera a instância i para $\bar{S}_i^j$ sendo o i-ésimo dado obtido no processo de *fingerprinting*.

2.5.3 Seleção do valor de k

Souza et al. (2020) descrevem a importância da seleção de um número k apropriado para o sistema de predição, pois a escolha pode ter grande efeito sobre os resultados obtidos de predição e sua precisão. Enquanto valores baixos de k podem levar o sistema ao *overfitting* comprometendo a variância das predições, altos números podem enviesar as respostas obtidas, comprometendo também a precisão nos predições.

Neste trabalho os valores de $k >= 2$ determinados por Xu e Chou (2017) serão reavaliados para a condição deste estudo na seção de resultados experimentais.

2.5.4 Função de Predição

Conforme determinado no trabalho de Xu e Chou (2017), a função de predição neste trabalho será baseada na média das coordenadas dos vizinhos mais próximos e sem nenhuma aplicação de pesos entre os vizinhos detectados, conforme determinada pela função 3, onde:

a) $(\bar{x},\bar{y})$: Coordenadas da predição;
b) (x_i,y_i): Coordenadas do i-ésimo vizinho;
c) K: Quantidade de vizinhos mais próximos selecionados.

$$(\bar{x},\bar{y}) = \frac{1}{K} \sum_{i=1}^{K} (x_i,y_i) \tag{3}$$

Neste estudo, para manter comparabilidade dos resultados, serão utilizados os mesmos dados de treinamento gerados na etapa de simulação para cada modelo RNR estudado ao invés dos próprios dados de *fingerprinting* do ambiente.

2.6 Redes Neurais Artificiais

Inspirado na hipótese do funcionamento de neurônios em um cérebro (RUSSELL; NORVIG, 2009), Redes Neurais Artificiais (RNA), ou *Artificial Neural Network* (ANN) implementa uma rede de estruturas chamadas *perceptrons*, que simulam o processo de neurônios biológicos, funcionando como um *gate* onde sua saída é definida a partir da aplicação do resultado de uma combinação ponderada de suas entradas e do bias à função de ativação, conforme a figura 3.

A saída do *perceptron* é definida pela fórmula (4), onde a função de entrada é gerada pela somatória ponderada de cada entrada a_i (onde a_0 é o *Bias*) é ponderada pelo peso $w_{i,j}$ e aplicada à função de ativação $g(in_j)$ para definir a saída a_j.

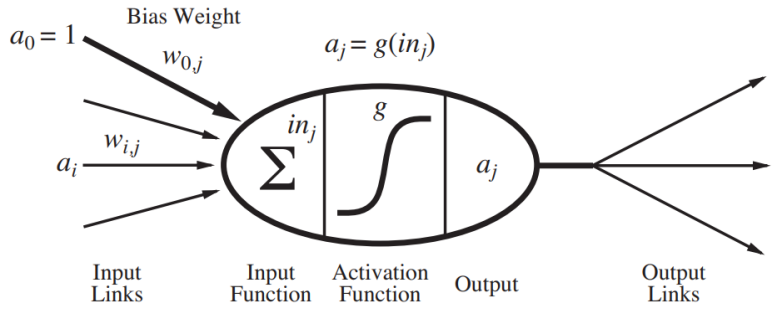

Figura 3 – Funcionamento de um Perceptron

Fonte: Russell e Norvig, 2009.

$$a_j = g(in_j) = g(\sum_{i=0}^{n} w_{i,j} a_i) \tag{4}$$

Apesar de existirem inúmeras possibilidades de construção de estrutura de RNAs, envolvendo a quantidade de perceptrons, quantidade de camadas, parâmetros de aprendizado, conexões entre perceptrons, entre outros, o foco deste trabalho será nas Redes Neurais Recorrentes.

A figura 4 demonstra uma estrutura básica de rede neural, com cada célula representando um *perceptron* e os valores de w indicando os pesos de cada conexão. Na figura 4 (a) há uma rede com duas células de entrada e duas células de saída de dados, na figura 4 (b) a rede neural

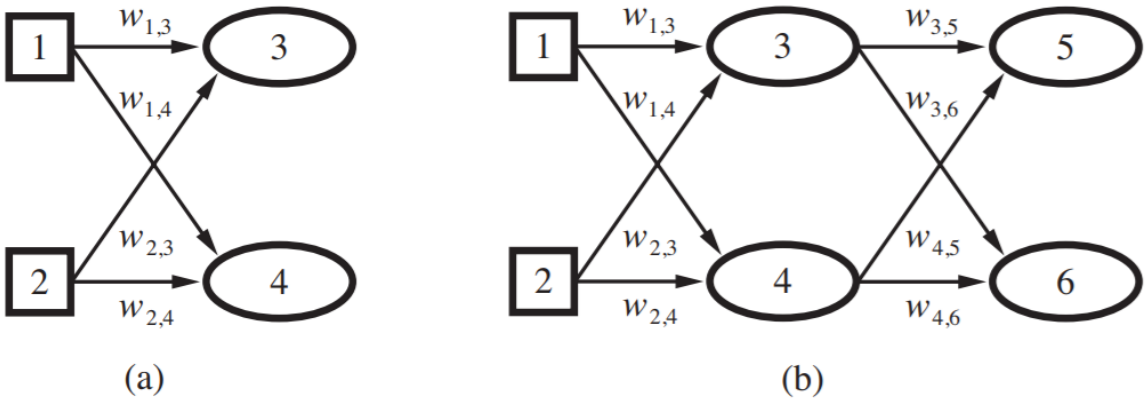

Figura 4 – Estrutura básica de uma rede neural.

Fonte: Russell e Norvig, 2009.

apresenta também uma camada oculta com mais duas células. Os dados de *bias* não estão representados nessa figura.

2.6.1 Redes Neurais Recorrentes (RNR)

Uma classe das Redes Neurais Artificiais onde, ao invés dos dados correrem em apenas um sentido, nas Redes Neurais Recorrentes (RNR), ou *Recurrent Neural Network* (RNN) o resultado obtido em um instante de tempo anterior é realimentado nas entradas da rede neural de forma que o(s) resultado(s) do(s) estado(s) anterior(es) influencia(m) no resultado do instante atual, formando um sistema dinâmico que pode atingir um estado estável, exibir resultados oscilatórios ou até apresentar comportamentos caóticos (RUSSELL; NORVIG, 2009).

Apesar de serem mais complexas que redes neurais com alimentação para frente por apresentar seu resultado dependendo de seu estado inicial, apresentam melhores modelos de comportamento cerebral (RUSSELL; NORVIG, 2009) e são melhores para tratar problemas que possuem correlação sequencial, como no caso da localização de robôs, que pode utilizar os dados de localização dos instantes anteriores, além de considerar dados de movimentação do robô como referência para melhorar a precisão da localização atual (HOANG et al., 2019).

A figura 5 demonstra a diferença entre uma RNA com alimentação para frente e uma RNR, com a camada oculta sendo realimentada diretamente com dados da última instância de processamento, além dos dados da instância atual. A descrição é uma representação básica de uma RNR, que pode ser composta por diversos tipos de conexão entre suas células, tanto nas camadas de entrada e saída, quanto nas camadas ocultas.

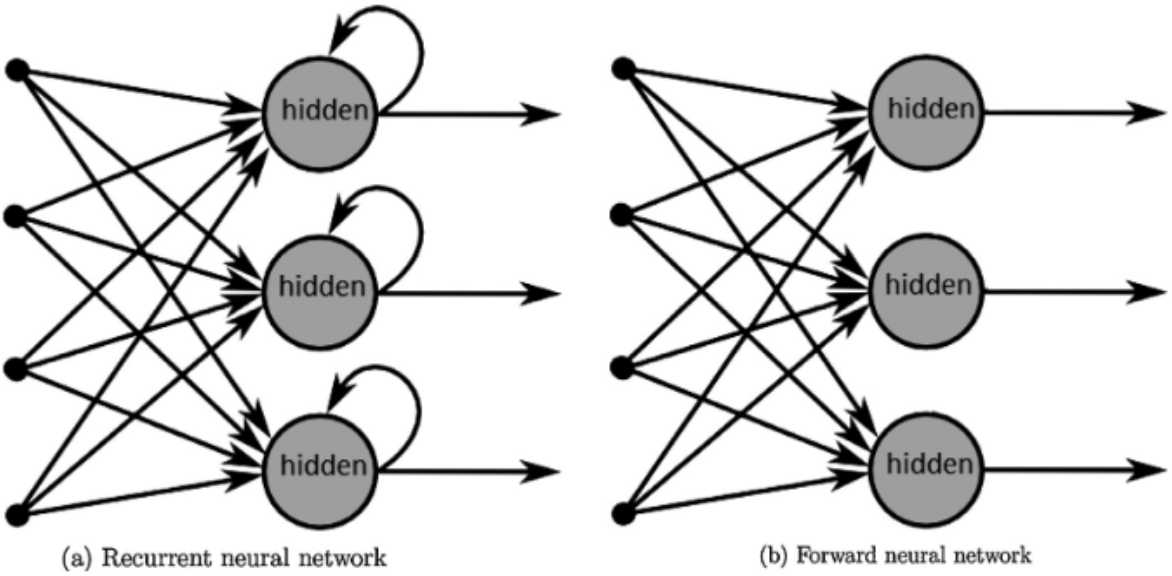

Figura 5 – Comparação entre uma RNR (a) e uma RNA comum (b)

Fonte: De Mulder, Bethard e Moens, 2015.

2.6.1.1 Long Short-Term Memory (LSTM)

Hochreiter e Schmidhuber (1997) definem LSTM como uma arquitetura de Rede Neural Recorrente em conjunto com um algoritmo apropriado de base de gradiente, criando pontes de intervalo temporais acima de 1000 passos caso seja detectada uma sequência ruidosa ou incompreensível. É uma estrutura de rede neural que permite além de analisar processos dependentes de tempo, convergir a respostas corretas em casos de flutuação nos dados devido à perturbações nas informações. O modelo se assemelha a uma RNR com uma camada oculta, mas substituindo cada nó da camada oculta por uma célula de memória (LIPTON; BERKOWITZ; ELKAN, 2015).

Enquanto nas RNRs existe a memória de longo prazo representada pelos pesos dos nós mudando lentamente durante o processo de treinamento e a memória de curto prazo representada pela recorrência dos dados de saída na entrada da RNR, nas LSTMs é introduzida um tipo de memória intermediária com as células de memória, unidades elaboradas a partir de nós comuns com conexões específicas entre os mesmos e com o adicional dos nós multiplicativos, representados na figura 6 pela letra Π (LIPTON; BERKOWITZ; ELKAN, 2015).

A célula de memória é formada pelas seguintes estruturas (LIPTON; BERKOWITZ; ELKAN, 2015):

a) Nó de entrada: É o nó que recebe as entradas da célula de memória do instante atual e as recorrências, aplicando função de ativação $tanh$, representado na figura 6 como g_c;

b) Gate de entrada: Os gates são unidades sigmoides que recebem os dados de entrada e recorrentes, e recebem esse nome pois funcionam como uma porta de controle pois seu valor é usado para multiplicar-se com resultado de um nó, podendo cortar sua saída caso o gate seja 0 ou passar a informação completa caso seja 1. A saída do gate de entrada (representado por i_c) é multiplicado depois com a saída do nó de entrada.

c) Estado Interno: No centro da célula de memória fica o Estado Interno s_c, um nó com função de ativação linear e recorrência própria de peso único, permitindo que o erro flua entre instantes sem sumir ou estourar. As vezes chamado de carrossel de erro constante, sua saída é calculada a partir da função (5), onde $*$ representa multiplicações escalares. É comum o estado interno apresentar função de ativação $tanh$ como um nó oculto de mesma função, mas a aplicação da função de ativação linear apresenta maior facilidade de treinamento, devido ao seu maior alcance dinâmico.

d) Gate de Esquecimento: Os gates de esquecimento f_c providenciam um meio de extinguir os conteúdos do estado interno, útil em aplicações de uso contínuo da rede, e representado pela fórmula da figura 6 (GERS; SCHMIDHUBER; CUMMINS, 1999 apud LIPTON; BERKOWITZ; ELKAN, 2015).

e) Gate de Saída: A saída v_c da célula de memória é calculada a partir da multiplicação escalar entre o estado interno s_c e o gate de saída o_c, fórmula (6), que recebe as entradas e recorrências assim como o gate de entrada.

$$s_c^{(t)} = s_c^{(t)} * i_c^{(t)} + s_c^{(t-1)} * f_c^{(t)} \tag{5}$$

$$v_c^{(t)} = s_c^{(t)} * o_c \tag{6}$$

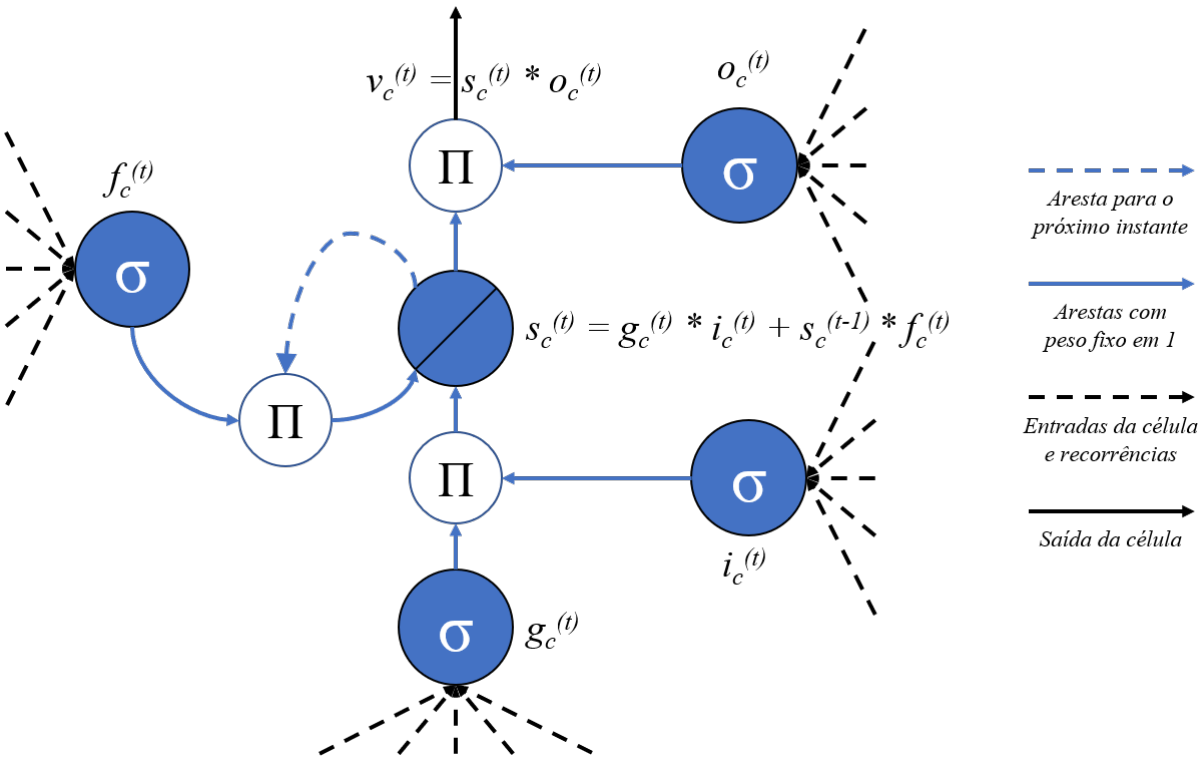

Figura 6 – Representação de uma célula de memória LSTM.

Fonte: Lipton, Berkowitz e Elkan (2015)

2.6.1.2 Gated Recurrent Unit (GRU)

Cho et al. (2014) define a GRU como uma arquitetura de rede neural recorrente inspirada pela LSTM, mas com as unidades da camada oculta mais fáceis de implementar e com menor esforço computacional, devido à GRU consistir em apenas 2 *gates* se comparado aos 3 gates presentes na LSTM, conforme pode ser observado na figura 7. Na célula LSTM i, f e o são respectivamente os *gates* de entrada, esquecimento e saída, enquanto c e $\tilde{c}$ representam a célula de memória e a nova célula de memória. Já na GRU r e z são os *gates* de redefinição e atualização, enquanto h e $\tilde{h}$ são ativação e a candidata a ativação.

A função do gate de redefinição é definir em qual momento as informações de longo prazo devem ser esquecidas, enquanto o *gate* de atualização controla o quanto dessa informação do estado anterior deve passar para o próximo estado (CHO et al., 2014).

Segundo Cho et al. (2014), para cada célula oculta j ,temos o *gate* de redefinição r_j calculado pela fórmula (7), onde σ representa a função logística sigmoid, x e $h_{(t-1)}$ são a entrada e o estado anterior da célula oculta, W_r e U_r são os pesos a serem definidos no processo de aprendizado.

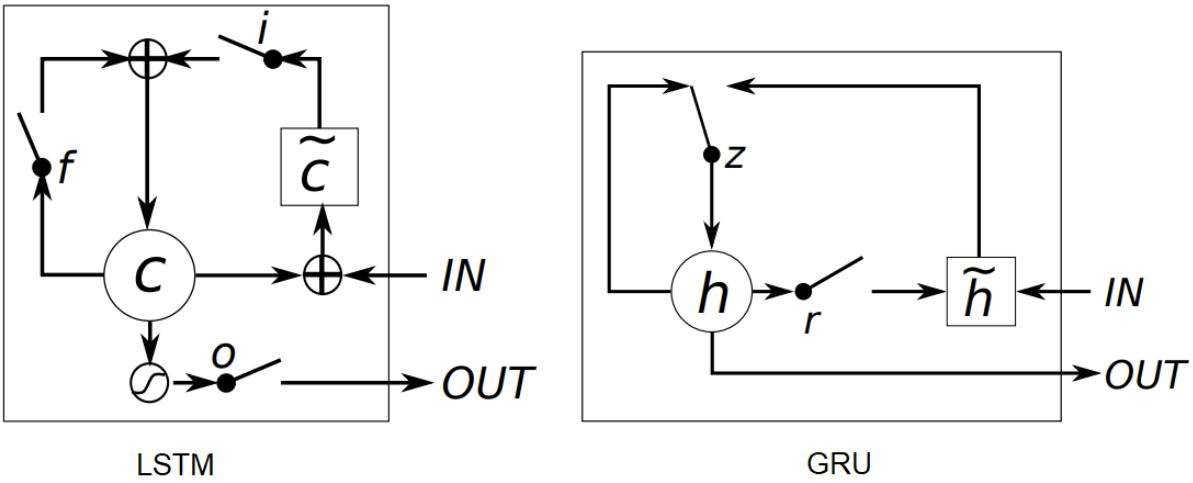

Figura 7 – Comparação entre uma célula LSTM e uma célula GRU

Fonte: Chung et al. (2014)

$$r_j = \sigma([W_r x]_j + [U_r h_{(t-1)}]_j) \tag{7}$$

Para o *gate* de atualização z_j, seguimos com a mesma representação de variáveis exemplificadas no *gate* de redefinição para a fórmula (7), mas com os pesos W_r e U_r substituídos pelos pesos W_z e U_z do *gate* de atualização.

$$z_j = \sigma([W_z x]_j + [U_z h_{(t-1)}]_j) \tag{8}$$

A saída de ativação h_j é calculada com a fórmula (9), onde temos

$$h_j^{(t)} = z_j h_j^{(t-1)} + (1 - z_j)\tilde{h}_j^{(t)} \tag{9}$$

Onde podemos calcular $\tilde{h}^{(t)}$ com a fórmula (10), com ϕ representando a função tangente hiperbólica ($tanh$) aplicada sobre o resultado, e W e U representando os pesos de entrada.

$$\tilde{h}_j^{(t)} = \phi([Wx]_j + [U(r * h_{(t-1)}]_j) \tag{10}$$

Pelo fato de cada célula possuir seus próprios *gates* de redefinição e atualização, cada célula pode aprender instantes independentes entre si para ativação, contendo na mesma rede neural unidades que ativam seu gate de redefinição frequentemente para padrões curtos, enquanto outras podem aprender padrões de maior duração, apresentando maior ativação do *gate* de atualização (CHO et al., 2014).

As fórmulas citadas acima definem um comportamento das células na camada oculta que ignora os estados anteriores assim que o *gate* de *reset* atingir valor próximo de 0, forçando a célula a trabalhar apenas com as entradas atuais da GRU, ao mesmo tempo o *gate* de atualização regula o quanto de informação será passada dos estados anteriores para o estado atual, comportamento semelhante ao das redes LSTM, relembrando informações de longo prazo (CHO et al., 2014).

Chung et al. (2014) complementa observando que, por não possuir nenhum mecanismo de controle de exposição do estado da célula, o mesmo é sempre repassado por completo para o próximo estado.

2.6.2 Treinamento

As RNAs necessitam passar pelo processo de treinamento, onde seus pesos são recalculados para aprenderem a representatividade de cada entrada em sua sinapse. O treinamento consiste em aplicar dados na RNA e avaliar sua resposta iterativamente até o erro convergir para as taxas máximas de projeto. As técnicas de treinamento podem ser classificadas em 3 categorias (RUSSELL; NORVIG, 2009):

a) Treinamento por reforço: Quando a RNA recebe uma pontuação em função da performance de cada atuação;
b) Treinamento não supervisionado: Geralmente utilizado por classificadores, quando o intuito é agrupar dados por semelhanças, sem determinar requisitos arbitrários;
c) Treinamento supervisionado: São apresentados pares de dados de entrada e sua saída esperada para avaliar o erro da RNA.

Este estudo será focado na realização do treinamento supervisionado da RNA.

2.6.2.1 Treinamento Supervisionado

Com dados apresentados em pares de entradas e saídas, no treinamento supervisionado a rede recebe diversos dados de exemplo com respostas corretas conhecidas, e sua saída é comparada com a resposta esperada. A partir do erro da resposta do sistema, os pesos das entradas de cada *perceptron* são remodelados para realizarem a função pretendida posteriormente, passo chamado de *back-propagation*, devido ao seu processo de recalcular os pesos de cada entrada e

do bias de cada perceptron em sentido oposto ao funcionamento normal da rede, partindo do(s) *perceptron(s)* de saída até o(s) *perceptron(s)* de entrada (RUSSELL; NORVIG, 2009). Este método será abordado neste estudo utilizando como entradas os dados de RSS adquiridos no simulador em função do tempo e como dados de saída a localização do robô estipulada para a geração dos dados simulados de RSS.

Especificamente nas RNRs, os dados de treinamento são compostos por um par de conjuntos de vetores reais de entrada $(x^{(1)}, x^{(2)}, x^{(3)}, ..., x^{(T)})$ e de saída $(y^{(1)}, y^{(2)}, y^{(3)}, ..., y^{(T)})$ representando uma sequência de dados que podem ser finitas ou composta por infinitos contáveis e podem ser sequências temporais ou não temporais (LIPTON; BERKOWITZ; ELKAN, 2015). Neste trabalho será abordado sequências de dados finitas temporais igualmente espaçadas no tempo para geração de pares de sequências de dados de treinamento, validação e determinação da localização nos testes, representando o processo contínuo de obtenção de dados do robô.

2.6.3 Validação

Após a aplicação de uma grande quantidade de dados de treinamento em cada época de processamento, a capacidade de interpretação da RNA deve ser avaliada com a base de dados de validação, que devem ser diferentes dos dados utilizados no processo de treinamento e também diferente dos dados de teste (RUSSELL; NORVIG, 2009).

Ao realizar a validação pode-se verificar a eficácia da rede neural, que pode sofrer problemas no processo de aprendizado, como o *overfitting*, um fenômeno genérico que pode acontecer com qualquer sistema de treinamento, mesmo quando o os dados são aleatórios, quando aprende-se com tendência a resolução de um padrão que pode ser observado nos dados de treinamento, mas não representa um fenômeno real (RUSSELL; NORVIG, 2009).

A probabilidade de ocorrer *overfitting* aumenta com a quantidade de classificações que o sistema deve aprender, mas sua probabilidade diminui com o aumento da quantidade de dados de treinamento (RUSSELL; NORVIG, 2009).

2.7 Função de Distribuição Acumulada (FDA)

Deisenroth (2020) define a Função de Distribuição Acumulada como um método estatístico para demonstrar a probabilidade de um evento contínuo (por uma linha real $\mathbf{R}$ por exemplo), e será útil neste trabalho para avaliar a qualidade e velocidade de detecção dos sistemas de

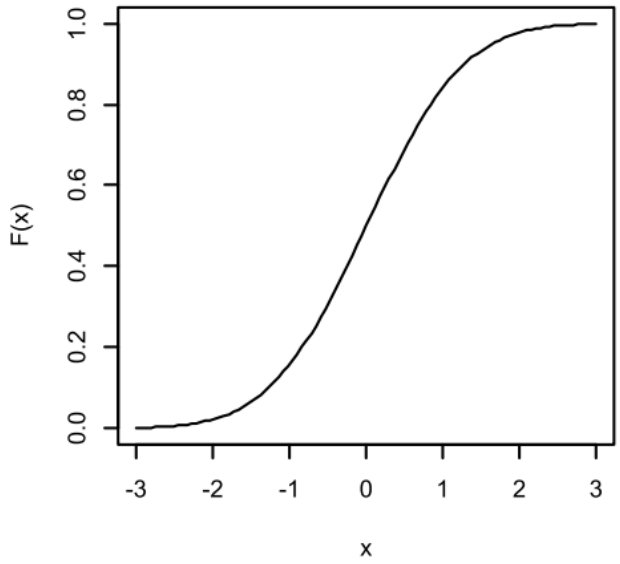

Figura 8 – Exemplo de função de distribuição acumulada

Fonte: Stowell (2014)

localização do robô nos diversos testes que serão executados e comparar o comportamento dos métodos aplicados em cada configuração de APs simulada. O método consiste em especificar a probabilidade de um valor aleatório X ser menor que um valor x definido por $P(X \leq x)$.

Stowell (2014) define a Função de Distribuição Acumulada (FDA) como "uma situação onde você selecionaria aleatoriamente um membro de uma população, qual seria a probabilidade desse membro possuir um valor menor que x, ou um valor entre x e y?". Partindo dessa descrição, podemos definir o FDA para este estudo como a probabilidade $F(x)$ de uma medição em certo instante ter erro menor ou igual a x.

2.8 *Analysis of Variance* (ANOVA)

Definido por Chandrakantha (2014) como um procedimento estatístico de análise de variância, ANOVA é um conceito utilizado para avaliar a hipótese das médias de três ou mais populações independentes serem consideradas semelhantes ou não.

A técnica consiste em calcular duas estimativas de variância σ^2 entre as populações, uma entre condições e a outra entre as amostras. A variância entre condições é baseada na variância das médias das amostras retiradas de cada população e a variância entre amostras é obtida a partir dos valores individuais de cada amostra.

O valor F é calculado a partir da variância entre condições e da variância entre amostras, conforme fórmula 11:

$$F = \frac{\text{Variância entre condições}}{\text{Variância entre amostras}} \tag{11}$$

O valor p indica a probabilidade de haver semelhança entre as hipóteses e é calculado a partir da curva de distribuição de F, que pode ser indicada de modo gráfico, conforme a figura 9 (CHANDRAKANTHA, 2014). Para a avaliação concluir que podemos descartar a possibilidade de que as hipóteses sejam semelhantes, o valor de p deve ficar abaixo de $\alpha = 0{,}05$, indicando que há menos de 5% de probabilidade, sem a necessidade de gerar o gráfico sugerido por Chandrakantha (2014).

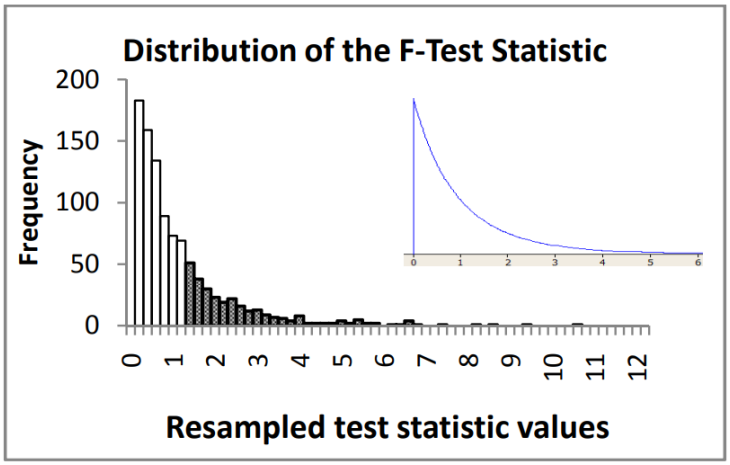

Figura 9 – Exemplo de análise gráfica de distribuição do valor F para populações de 1000 amostras, base de cálculo para obter o valor p.

Fonte: Chandrakantha (2014)

3 Revisão Bibliográfica

Os estudos sobre localização a partir de sinais de Wi-Fi cobrem uma larga escala de aplicações, desde sua utilização para resolver o Problema do Robô Sequestrado (SEOW et al., 2017) até localizar pessoas dentro de um ambiente a partir dos dados em um celular (HOANG et al., 2019). Mais do que aplicações, a grande diversidade de estudos na área se apresentam quando considerados os métodos utilizados por cada um.

3.1 *Received Signal Strength* (RSS)

Estudos baseados em RSS variam no método como os dados obtidos são tratados. Os primeiros estudos na área consideravam modelar a perda de propagação e aplicação de filtros para obter a localização por meio de triangularização (EL-NAGGAR; WASSAL; SHARAF, 2019) com certa precisão em ambientes em vão livre, mas falharam em obter bons resultados em ambientes com paredes ou mais complexos, mesmo aplicando filtros para reduzir o impacto de interferências dinâmicas.

O mapeamento de sinais RSS em diversos pontos, processo chamado de *fingerprinting*, é mais confiável que o modelo de propagação (XUE et al., 2019) e pode ser observado em trabalhos como o do Google® em seu API de geolocalização do Google Maps® que localiza dispositivos móveis comparando dados de RSS medidos pelo dispositivo com dados obtidos pelos carros da companhia durante a obtenção de imagens para o *Google Street View* e atrelados à localização por GPS do veículo (GOOGLE, Feb. 2018 apud LAOUDIAS et al., 2018).

Os desafios para a implementação de soluções de localização por RSS foram levantados por Chapre et al. (2013), e incluem orientação e tipo de antena, distância entre transmissor e receptor, interferência de outros dispositivos de radiofrequência, presença e/ou movimentação de pessoas e toda a estrutura, objetos e seus respectivos materiais empregados no próprio ambiente.

O trabalho de Zàruba et al. (2007) demonstra que a técnica abordada para geração de pontos de referência e tratamento dos sinais apresentam diferentes impactos com a quantidade de APs, realizando a modelagem de reflexos e atenuações dos sinais de uma antena omnidirecional a partir de alguns pontos de medição real e da planta do ambiente, obtendo precisão razoável de localização com apenas um único AP, mas apesar de exigirem menor quantidade de medições, funciona a partir de um complexo sistema de cálculo de *ray-tracing* e é dependente de uma planta precisa do ambiente.

Novos estudos como o WiDeep (ABBAS et al., 2019), aplicam os dados de RSS em umaRede Neural Profunda (RNP) treinada a partir do *fingerprinting* da área somado à ruídos aplicados digitalmente para treinamento do modelo da RNP. O método considera todos os APs disponíveis na área de atuação, que podem sofrer alterações drásticas fora do controle do usuário, como desativação ou mudança de localização dos APs. O estudo também realiza análises de impacto médio sobre a densidade de APs detectados no *fingerprinting* a serem utilizados no modelo de predição, concluindo que sua precisão se mantém acima de 2.8 metros, mesmo diminuindo a quantidade de APs considerados no sistema para os 10 mais significativos em termos de potência.

O uso de RSS vai além das aplicações com Wi-Fi, com trabalhos aplicando Redes Neurais em RSS de LoRa®, atingindo precisões de 300 metros (DARAMOUSKAS; KAPOULAS; PARASKEVAS, 2019), mas devido à característica de longo alcance do LoRa®, atingindo distâncias de 10km a 15km (SAARI et al., 2018), o resultado é útil para localização de pessoas em cidades. Também há aplicações em ZigBee® (DING; DONG, 2020) que atingiram precisão de 2 metros com níveis de ruídos em torno de 2dB a 3dB. Sadowski e Spachos (2019) realizaram análises de densidade de transmissores *Bluetooth® Low Energy* (BLE) para localização a partir de RSS, concluindo que o excesso de transmissores pode gerar interferência entre os sinais e piorar o resultado final. Trabalhos voltados a sistemas de rede celular 5G também apresentam técnicas de aplicação de RSS para localização de usuários de dispositivos móveis (KHAN et al., 2017).

Soluções utilizando RFID também apresentaram resultados de localização em ambientes internos entre 5 cm e 3 metros, dependendo da técnica aplicada (LI; MO; ZHANG, 2019). As técnicas que envolvem RFID podem ser classificadas conforme figura 10. Em soluções dependentes de cálculo de distância, utiliza-se além de RSS, técnicas como tempo de propagação e fase de frequência para obter a distância entre o *tag* RFID e cada antena no ambiente, obtendo a localização a partir de triangularização das distâncias. No caso das técnicas que não envolvem cálculo de distância, soluções que envolve o uso de *fingerprinting* utilizam RSS para obter os dados de um conjunto de diversas antenas e *tags*. Soluções sem cálculo de distância e sem uso de *fingerprinting* utilizaram-se de RSS e análise de fase para obter localização de itens em trajetos pré definidos. Tais soluções apresentam alta precisão, mas dependem de antenas operadas por um sistema central que realiza a análise dos dados e mantém o sincronismo em casos de triangularização. Li, Mo e Zhang (2019) também concluiu que o RFID é mais sensível a mudanças no ambiente do que o Wi-Fi.

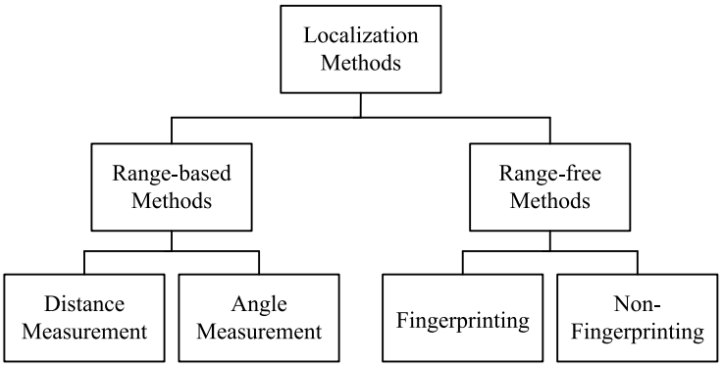

Figura 10 – Classificação das técnicas de localização via RFID.

Fonte: Li, Mo e Zhang (2019)

3.2 Sequestro de robôs

O problema do sequestro de robôs é um desafio que já inspirou diversos trabalhos buscando sua solução. Alguns dependem de sensoriamento especial, como câmeras estéreo para analisar características de objetos em 3 dimensões de referências observadas no ambiente (MAJDIK et al., 2010), identificando a ocorrência de um KRP até quando há caraterísticas dinâmicas, como pessoas se movimentando no ambiente. Outras soluções dependem de uma complexa análise de dados de sensores, como no trabalho de Desrochers, Lacroix e Jaulin (2015), que buscou identificar eventos de KRP mapeando os arredores com sensores laser LIDAR e comparando-os com informações de terreno obtidas por aeronaves não tripuladas em um campo aberto, ou Diamantas (2019) que utilizou imagens obtidas da câmera de vídeo do robô e as comparou com imagens geolocalizadas obtidas em redes sociais para obter a localização aproximada do robô e identificando se houve sequestro.

Xu e Chou (2017) utilizaram dados de RSS como uma forma de obter a localização aproximada do robô e usar como base para distribuir as partículas de Localização de Monte Carlo Adaptativa em torno dessa localização no uso de dados de LRF, evitando que durante um sequestro de robôs haja um colapso devido à degradação de partículas. O trabalho consiste em implementar o tratamento dos dados de RSS Wi-Fi associado a um processo de tratamento de dados sensoriais de LRF para melhorar a precisão do sistema utilizando Localização Adaptativa de Monte Carlo, ou *Adaptative Monte Carlo Localization* (AMCL), uma técnica baseada no sistema Localização de Monte Carlo, ou *Monte Carlo Localization* (MCL), onde a localização é feita a partir da distribuição aleatória de partículas por toda a área de resposta e processamentos iterativos convergem tais partículas até a(s) resposta(s) correta(s). Na figura 11, Xu e Chou

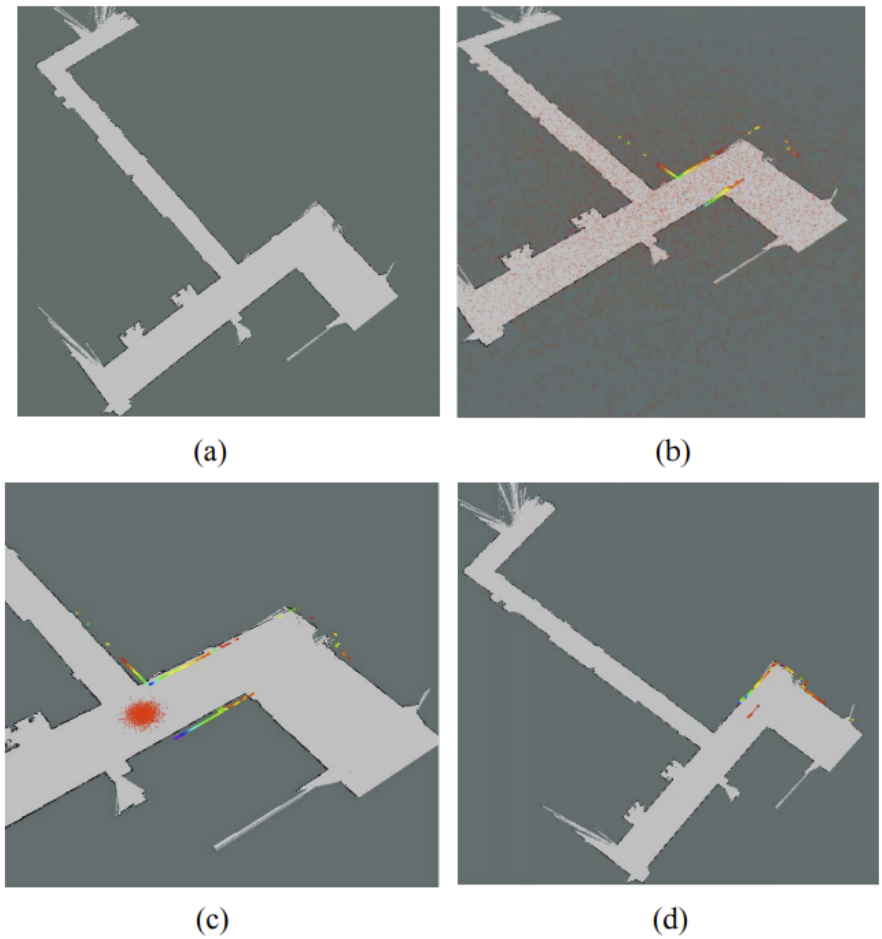

Figura 11 – Fases do sistema proposto por Xu e Chou (2017) durante a localização.

(a): Mapa do ambiente. (b): Distribuição normal das partículas AMCL sem localização com KNN. (c): Distribuição após localização com KNN. (d): Resultado após refino do posicionamento com sensoriamento com Laser Range Finder. Fonte: Xu e Chou (2017).

(2017) detalham graficamente as diferentes fases do processo de localização: distribuição das partículas, redistribuição com o sistema KNN baseado em RSS e refino da localização com sensoriamento laser (LRF), aplicando AMCL.

A arquitetura de sistema proposto por Xu e Chou (2017) é composta por duas etapas de localização, identificadas na arquitetura da figura 12 como “Fingerprint Coarse Localization”, seguido pelo procedimento utilizando Laser Range Finder para refinamento da localização (identificado na figura 12 como “AMCL Fine Localization”).

Enquanto algumas soluções dependem de seu próprio sistema de localização para identificação do sequestro de robô, algumas propostas consistem em utilizar subsistemas dedicados apenas à identificação de tais eventos e notificá-los ao sistema principal de localização. Yi e Choi (2011) propuseram a análise de entropia de objetos reconhecidos por imagens de câmera em um mapa de distâncias semânticas para detecção do sequestro do robô. Redes neurais recor-

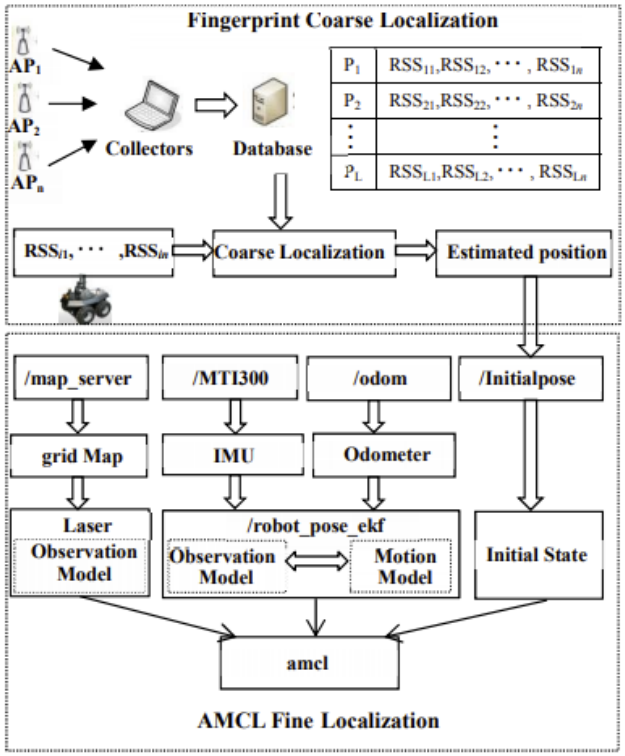

Figura 12 – Arquitetura de tratamento de dados do sistema proposto por Xu e Chou (2017).

Fonte: Xu e Chou (2017).

rentes também foram utilizadas exclusivamente para detecção de eventos de sequestro, como no trabalho de Xu, Chou e Dong (2019), que propôs um sistema LSTM para identificar o KRP a partir de uma sequência de imagens obtidas pela câmera do robô, enquanto um sistema de localização baseado em LRF realiza a localização precisa do dispositivo.

3.3 Combinações

Muitos trabalhos buscam uma combinação de diversas técnicas de localização para complementar os dados, como Carrera Villacrés et al. (2019) definem um sistema onde o Problema do Robô Sequestrado (KRP) pode ser resolvido a partir de análise de dados de sinais RSS em um sistema que alia Aprendizado por Reforço e Modelo Oculto de Markov a um Filtro de Partículas para melhorar a robustez do sistema de localização. Em (SEOW et al., 2017), o autor combinou informações de RSS com *Laser Range Finder* (LRF) para o sistema RSS melhorar a velocidade de se relocalizar caso ocorra o KRP e utilizando o LRF para o ajuste fino de posição e orientação. Aplicações utilizando LSTM (HOANG et al., 2019) mostram grande progresso em conjunto com técnicas de Filtro de Partículas. El-Naggar, Wassal e Sharaf (2019) utilizaram

Filtros de Kalman para fundir os dados de RSS com Sistema de Navegação Inercial e definir a localização por triangularização.

A associação de filtros para a melhora do sinal analisado pelo sistema é fundamental para evitar erros de identificação provenientes de interferências de outros emissores de radio-frequência e situações dinâmicas presentes no ambiente, como o movimento de pessoas pelo local ou mudança de posição de móveis. O trabalho de Sadowski e Spachos (2019) conclui que a associação entre Filtro de Média Móvel e Filtro de Kalman podem melhorar a precisão em casos de múltiplos sinais.

4 Metodologia

O foco deste trabalho é verificar o impacto da quantidade de *Access Points* (APs) nos sistemas de localização baseados em LSTM, GRU e em KNN como soluções para o problema do robô sequestrado, utilizando o ambiente virtual Webots®, simulado com base em dados obtidos de *fingerprinting* de um ambiente real. Serão verificados os requisitos mínimos e robustez de cada sistema para as configurações de APs, além do tempo máximo e a média para atingir uma precisão de pelo menos 2 metros com cada configuração de APs, conforme métrica estabelecida por Hoang et al. (2019). Serão avaliados e comparados os sistemas LSTM, GRU e KNN com relação ao tempo de obter ao menos 2 metros de precisão, a progressão da Função de Distribuição Acumulada, e a precisão contínua final.

4.1 Ambiente de Referência

O ambiente a ser realizado os testes é o piso térreo de uma casa composta por sala, sala de jantar, cozinha, escritório e banheiro, formando um espaço total com 9,77m de de frente por 9,00m de profundidade e área de aproximadamente 63,60m², descrito na figura 13, onde também pode-se observar o posicionamento dos APs utilizados nas medições de *fingerpriting*.

4.2 *Access Points* e Wi-Fi *fingerprinting*

Os APs utilizados como base para a obtenção dos dados de *fingerprinting* da área são construídos a partir de placas de desenvolvimento ESP-WROOM-32 e ESP8266, por serem microcontroladores de baixo custo com hardware Wi-Fi 802.11n incluído e certificado pelo Wi-Fi Alliance[1], programadas na plataforma Arduino com o código WiFiAccessPoint[2], atuando exclusivamente como APs para a realização do *fingerprinting* da área para ser utilizado como referência para simulação dos dados de RSS enviados ao robô pelo simulador e gerar a base de dados para construção do simulador de sinais.

Os APs foram construídos com as antenas PCB inclusas nas placas de desenvolvimento, com a placa exposta e sem nenhuma carcaça, e instalados com suas antenas apontadas verti-

[1]Certificado presente em: `https://www.espressif.com/sites/default/files/ESP32-WROOM-32_Wi-Fi_Certificate.pdf`

[2]Exemplo incluso no IDF 1.0.4 da Espressif em: `https://github.com/espressif/arduino-esp32/releases/`

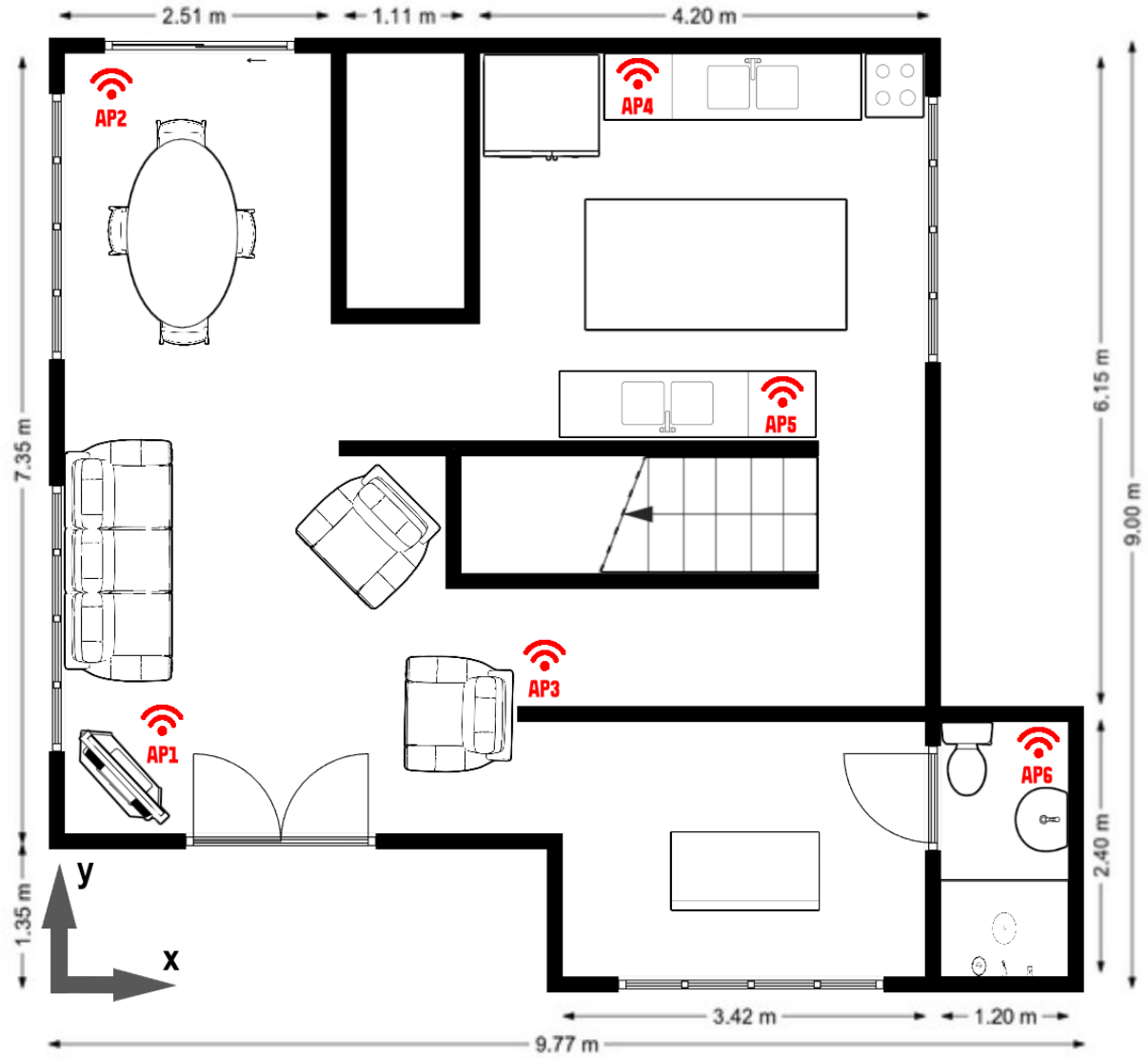

Figura 13 – Planta representativa da casa de referência para a geração do *fingerprinting*

Planta gerada no website Floorplanner: `https://floorplanner.com/`

calmente para cima, mas cada AP foi posicionado em uma altura diferente dependendo das condições do local de instalação, que podem ser observadas na tabela 1. Diferentes alturas foram utilizadas em favor de obter dados mais condizentes com aplicações reais, onde não há padrão para a instalação de APs. Cada dispositivo foi configurado para operar em um canal diferente de Wi-Fi, evitando conflitos de propagação desnecessários no processo de obtenção de dados de *fingerprinting*.

O primeiro passo tomado no desenvolvimento da simulação é realizar o *fingerprinting* de RSS de um ambiente real (conforme descrito na figura 15) e aplicá-los à uma base de dados que comporta tais informações em uma grade de 50cm x 50cm, devido a esperada precisão de aproximadamente 2m, conforme definido por Hoang et al. (2019), que será também a resolução

Tabela 1 – Disposição dos APs no ambiente de referência

Access Point	Disposição
1	Sobre uma mesa de TV com 80cm de altura
2	Sobre uma cadeira a 50cm do chão
3	Sobre uma cadeira a 50cm do chão
4	Sobre uma pia com 80cm de altura
5	Sobre um forno elétrico a 120cm do chão
6	No topo de um armário de 200cm de altura

máxima do sistema de localização, gerando um total de 326 pontos de referência frente aos possíveis 380 pontos numa grade de 9,77m x 9,00m, devido à exclusão de áreas não acessíveis do ambiente de referência, como paredes e locais externos fora dá área estudada. Para o processo de *fingerprinting*, os dados de RSS para cada ponto de medição no ambiente de referência serão organizados em forma de matriz e armazenados em arquivos do tipo *Comma Separated Values* (CSV), com um arquivo para cada AP. O arquivo tipo CSV é um padrão para armazenamento de tabelas ou matrizes em um arquivo de texto corrido. Consiste em interpretar cada caractere de "ponto e vírgula" como a identificação de uma nova coluna e cada caractere de "nova linha" como uma nova linha da tabela/matriz.

A obtenção dos dados de RSS no ambiente de referência será realizada utilizado o aplicativo WiFi Heatmap[3] para celulares com sistema operacional Android em um celular Samsung Galaxy S9+ (SM-G9650). O procedimento para realização das medições consiste nos seguintes passos:

a) Gerar uma imagem da planta do ambiente de referência (figura 13), para utilizar como plano de fundo durante a realização das medições.

b) Carregar no aplicativo a imagem de fundo e acertar a proporção entre imagem e medição real com base em alguma referência preliminar de distância, que neste estudo foi utilizada a frente, com 9,77m.

c) Posicionar e verificar o funcionamento correto de cada *access point* e sua visibilidade em curta distância para certificar-se que todos estão funcionando corretamente. Esse procedimento pode ser realizado direto no aplicativo *WiFi Heatmap*[3], na aba *"Channel Monitor"* (figura 14) ou utilizando a busca de APs do próprio dispositivo;

[3]WiFi Heatmap disponível no Google Play Store em: `https://play.google.com/store/apps/details?id=com.optivelox.wifiheatmap&hl=en`

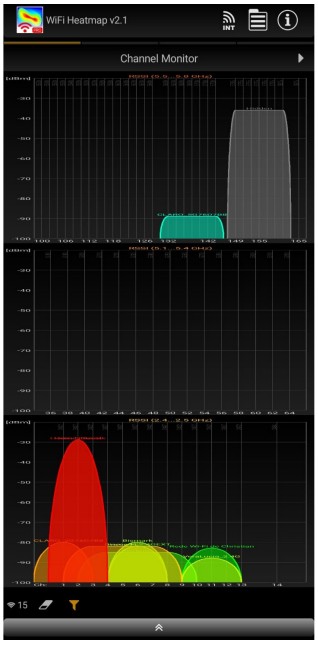

Figura 14 – *Channel monitor*

Monitor de sinais e canais de WiFi do aplicativo *WiFi Heatmap*[3]. Fonte: Autor

d) Uma trena laser foi utilizada para a determinação de cada ponto de medição no ambiente utilizando as distâncias até as paredes como referência;

e) Um vez determinado o ponto de medição no ambiente real, o ponto é selecionado no aplicativo para inicialização da medição;

f) Para tentar evitar distorções provenientes de quem está manuseando o celular durante a medição, ela é realizada sentado no chão com o braço esticado e o celular apontado para cima, na altura de 1,5m;

g) A medição é realizada durante 3 segundos, a partir da média de 3 medições realizadas, uma por segundo, e o aplicativo *WiFi Heatmap* gera uma média das medições realizadas;

Apesar do aplicativo *WiFi Heatmap* realizar uma integração dos dados de RSS entre pontos de medição, tal função foi utilizada apenas para a construção de uma imagem de visualização do *heatmap* mais coesa, todos os pontos utilizados neste estudo são provenientes de medições reais no ambiente de referência.

4.3 Ambiente Webots®

Para a realização da simulação e geração dos dados de treinamento, validação e teste dos sistemas de localização, será utilizado o ambiente virtual Webots® de simulação de robôs, ver-

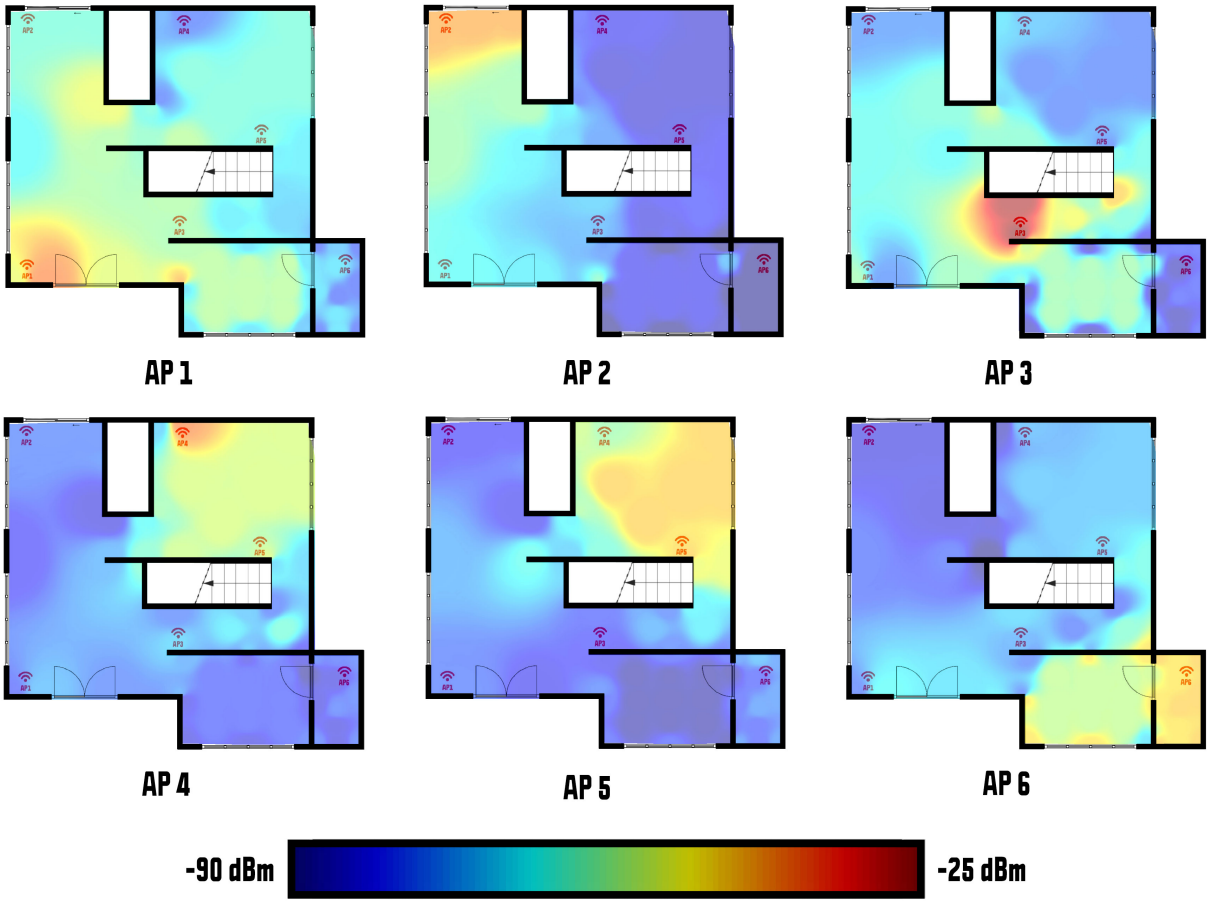

Figura 15 – *Heatmaps* medidos em ambiente real

Heatmap obtido com o uso do software *WiFi Heatmap*[3] a serem utilizados como base para a construção do simulador. Fonte: Autor

são R2020a revisão 1, no sistema operacional Windows® 10, em um PC com CPU AMD Ryzen 7 1700 com 16Gb de RAM DDR4 e GPU NVIDIA GeForce GTX 1070Ti, tomando como base para a construção do simulador um exemplo de ambiente de simulação incluso no software, baseado na plataforma de desenvolvimento *iRobot create*®, que simula o funcionamento de um robô aspirador em uma casa. A planta da casa de referência será reproduzida dentro do simulador, mas sem nenhum móvel nem objeto, pois os dados de *fingerprinting* obtidos no ambiente real que serão utilizados para a geração dos sinais de referência já representam as atenuações básicas que ocorrem na presença dos objetos e pelo fato do sistema de localização não depender de nenhum dado de sensoriamento de colisão.

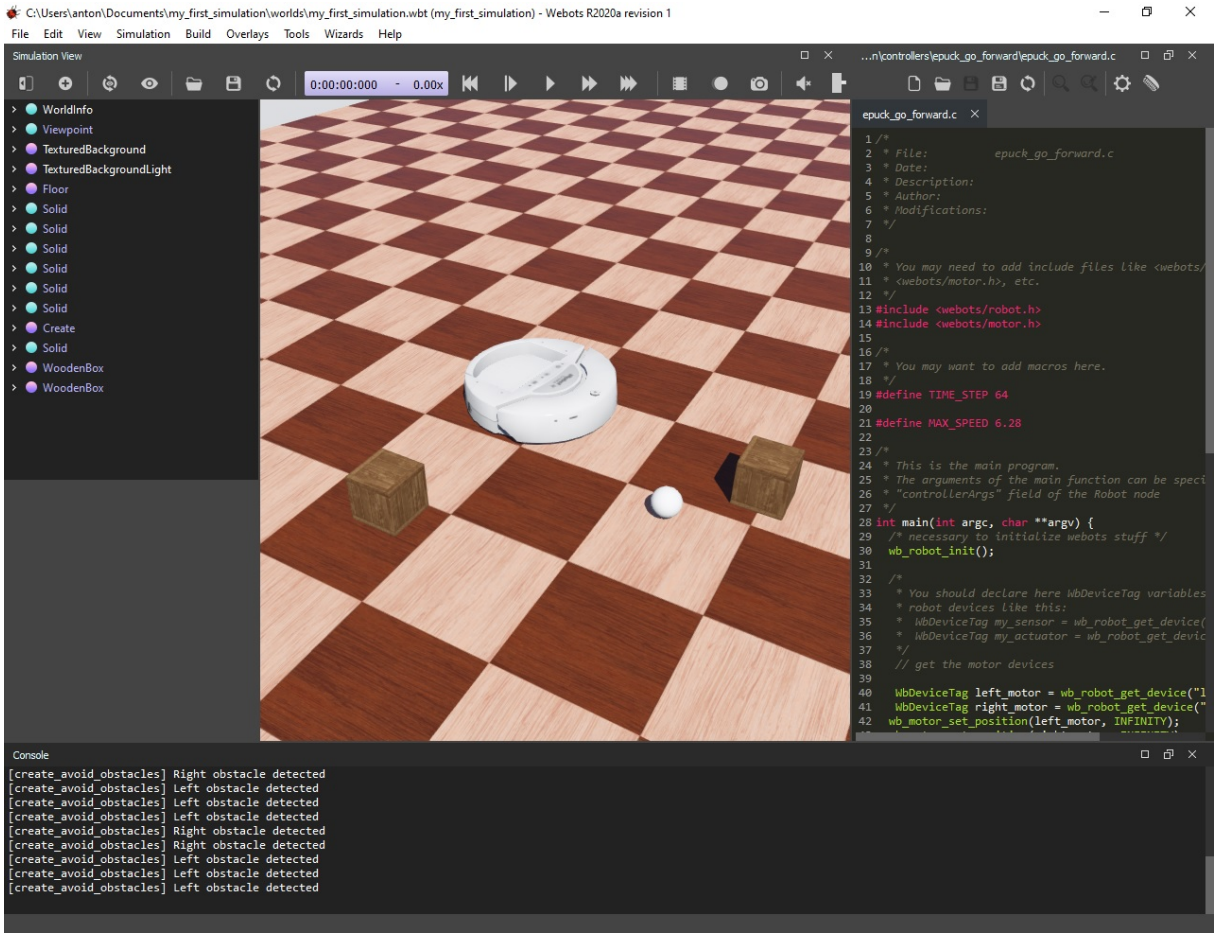

Figura 16 – Interface do sistema Webots®

A esquerda está a lista de objetos simulados, no centro a visualização 3D da simulação e na direita o código utilizado para a movimentação do robô no ambiente. Fonte: Autor

4.3.1 Supervisório de atuação do sistema Webots®

O sistema de Supervisório do simulador Webots® consiste em um robô virtual, sem construção física no ambiente de simulação, e com programação específica para a realização de medições do ambiente de simulação e atuar sobre o mesmo. O sistema será programado na linguagem python e utilizado no ambiente de simulação para realizar diversas tarefas relacionadas ao processo de simulação:

a) Manter controle do tempo de simulação, como o momento para obtenção de dados, instante para realização do KRP e encerramento de cada teste;

b) Obter os dados de posicionamento do robô a cada segundo, como pré-determinado para este estudo;

c) Referenciar o posicionamento com os dados de fingerprinting e calcular o sinal RSSI simulado com os parâmetros de teste estabelecidos;

d) Catalogar os dados de número do teste, posicionamento real do robô, RSSI simulados, tempo de teste, e se já foi realizado o KRP no teste atual;
e) Realizar o reposicionamento do robô no instante escolhido para realização do KRP e no final de cada teste;
f) Converter os dados para inclusão no arquivo CSV;
g) Manter um log de simulação atualizado com o último passo realizado;
h) Encerrar a simulação quando obter a quantidade necessária de dados de teste e salvar os dados obtidos de simulação em arquivo na pasta de dados de treinamento.

4.4 Simulador de sinais de RSS

Os dados RSS de treinamento, validação e teste serão gerados dentro do ambiente Webots® e construídos com base nos dados de *fingerprinting* obtidos no ambiente real. Para a construção dos dados de RSS virtuais, o sinal de Wi-Fi obtido pelo robô receberá um ruído digital, incluindo Mascaramento de Dados e Ruído Gaussiano de 1 dBm proposto por Abbas et al. (2019) para obter uma simulação de efeitos de áreas de baixo sinal e ruídos mais próxima a uma aplicação real, conforme figura 17. Evitando uma padronização do comportamento, cada AP possuirá seu próprio simulador de ruídos, calculados independentemente, e com parâmetros de aleatoriedade únicos, buscando melhor variabilidade nos dados de treinamento, validação e teste gerados.

4.4.1 Sistema de atenuação

Primeiro passo no cálculo do valor simulado de RSS é a inclusão do sistema de atenuação definido por (ABBAS et al., 2019), que cria um vetor de atenuação com valores binários aleatórios e em seguida realiza uma multiplicação escalar com o vetor de RSS gerado pelo *fingerprinting* para a determinada posição. Seu intuito é simular variações com o tempo devido a multi propagações e efeitos de desvanecimento (YOUSSEF; AGRAWALA, 2003 apud ABBAS et al., 2019). A técnica é aplicada para representar a flutuação dos APs, simulando uma situação em que o robô não detecta a presença algum AP.

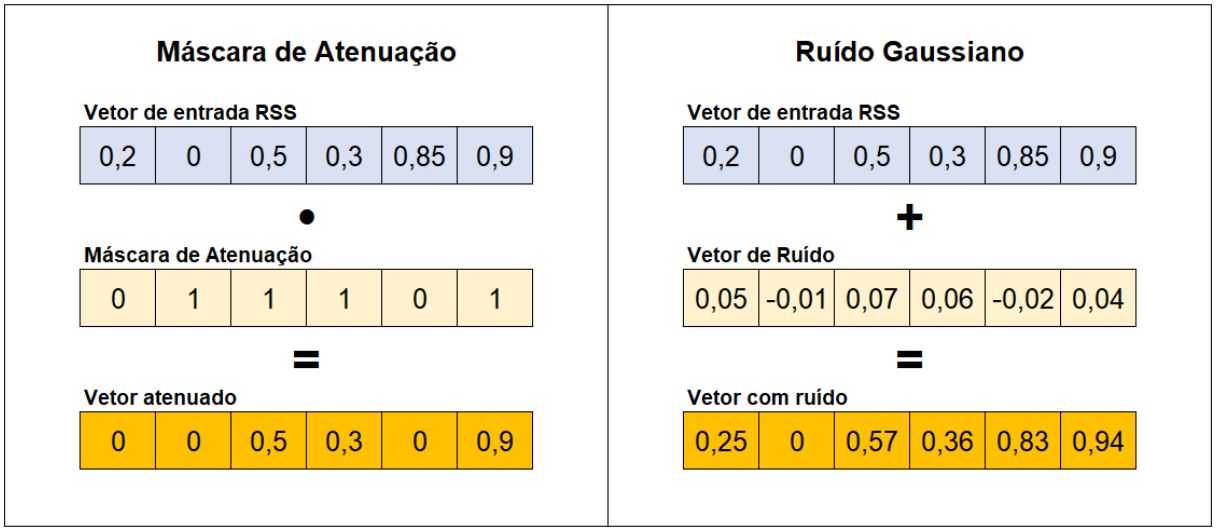

Figura 17 – Exemplo de cálculo do simulador de sinais RSS com 6 APs. O "Vetor de entrada RSS" obtém os sinais RSS medidos através do *fingerprinting* para a localização atual do robô para cada AP do ambiente de simulação.

Fonte: Abbas et al. (2019).

4.4.2 Sistema de ruído

O objetivo do segundo passo é gerar aleatoriamente um vetor de ruído, devido ao comportamento errático dos dados de RSS em um ambiente real, por sofrerem alterações devido a outros dispositivos operando no mesmo canal ou até variações no hardware dos dispositivos (ABBAS et al., 2019). Baseado no mesmo trabalho citado no sistema de atenuação, o sistema de ruído consiste em aplicar um ruído branco gaussiano com desvio padrão s para cada entrada do vetor de RSS, realizando uma soma escalar entre o vetor de RSS do *fingerprinting* e o valor de ruído calculado. Caso algum valor saia da escala de medida do RSS, o dado deve sofrer uma normalização, que, por definição de escala do processo de *fingerprinting*, é definido entre 0dBm e -100dBm.

4.5 Gerador de caminhos

Para realizar a construção da base de dados para treinamento, validação e teste, será construído um gerador de caminhos aleatório, baseado no controlador "create_avoid_obstacles.c", incluso no ambiente de simulação Webots® e considerando as restrições de movimento do robô no ambiente e sua velocidade de movimento em relação ao período entre obtenção de atualização dos dados de RSS.

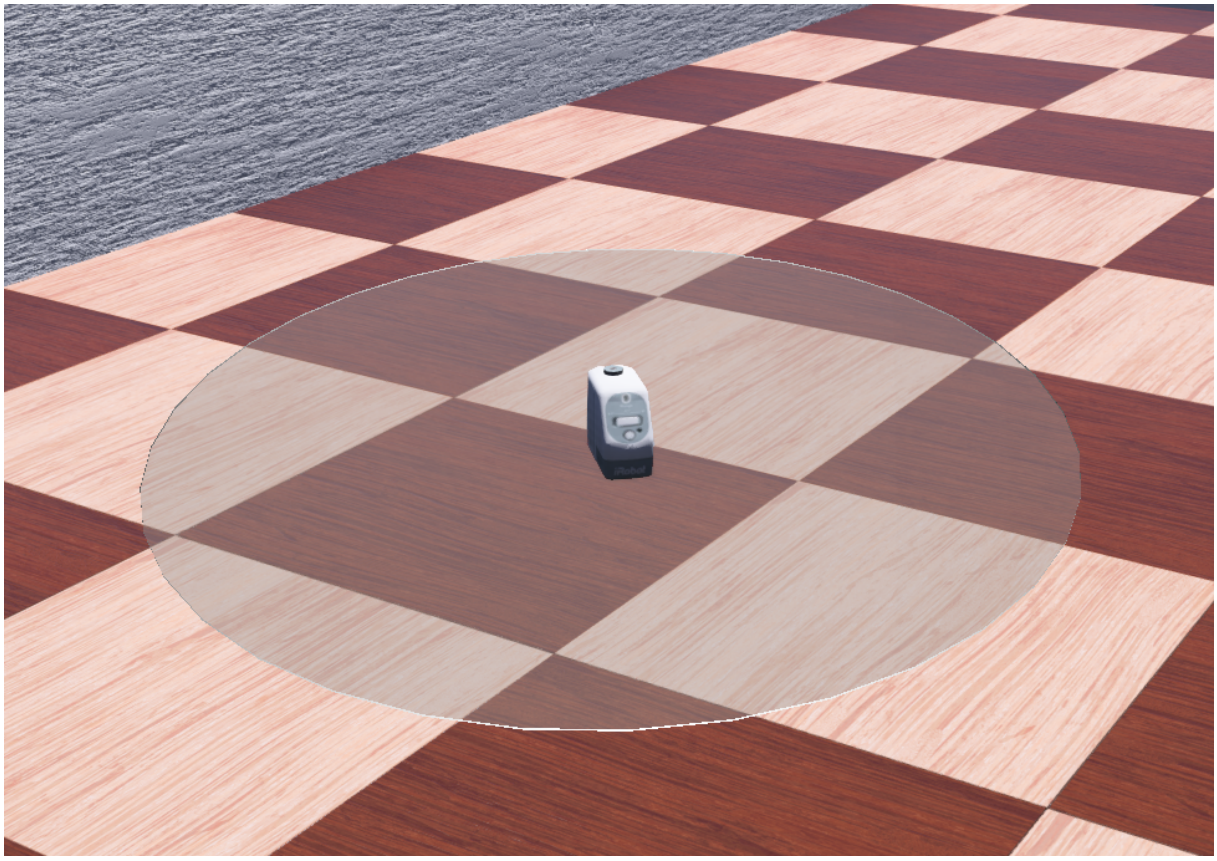

Figura 18 – Barreira virtual do iRobot® dentro do ambiente de simulação, dispositivo utilizado para balizar o movimento do robô em certos pontos do ambiente, seja impedindo de passar por algum vão ou gerando uma área restrita radial, conforme indicado pela área em cinza.

4.5.1 Controlador do robô *iRobot® Create*

O controlador é inspirado na plataforma de desenvolvimento *iRobot® Create* e consiste em realizar primeiramente um movimento a frente com velocidade de 0,5m/s até atingir um obstáculo ou receber o sinal de uma "barreira virtual" (figura 18). Ao identificar com seus sensores que atingiu uma dessas condições, o robô anda 25cm com metade da velocidade total no sentido contrário e aleatoriamente realiza uma rotação em torno de seu próprio eixo para, em seguida, retomar o processo inicial, andando a frente com velocidade máxima.

Outra condição do controlador do robô é reconhecer a beirada de um "penhasco", que pode ser, por exemplo, a beirada de uma varanda ou o primeiro degrau de uma escada para um pavimento inferior. Nessa condição o robô consegue identificar com antecedência e reduzir sua velocidade pela metade até alcançar a beirada, para assim realizar o mesmo movimento feito ao encontrar um obstáculo.

4.6 Procedimento de geração de dados de treinamento, validação e teste no ambiente de simulação

Com base nos parâmetros para o sistema de localização indicado por Hoang et al. (2019), onde o comprimento de memória da LSTM é definido por 10 segundos, separados em 10 períodos de1 segundo cada, mas considerando o desafio apresentado de avaliar a recuperação após a realização do KRP, a base de dados deste estudo será compilada em testes simulados de 30 segundos cada, com 10 segundos para a convergir a localização inicial, 10 segundos para a aplicação aleatória do KRP e 10 segundos finais para o sistema de localização apresentar tempo hábil de se relocalizar após a ação.

O simulador gerará para cada conjunto de dados um trajeto de 30 segundos, no modo acelerado e sem gráficos do software Webots®, obtendo uma medição a cada segundo.

O robô partirá de uma localização e uma orientação aleatória no ambiente de simulação, e a partir desse ponto inicial o robô realizará seu movimento padrão do controlador. Em um instante aleatório entre 10 a 20 segundos, como um modo de simular a realização de um KRP, o robô será instantaneamente reposicionado e reorientado aleatoriamente no ambiente de simulação, com os últimos 10 segundos reservados para avaliar a resposta do sistema de localização após o KRP. No final de cada teste, o simulador realizará outro reposicionamento nos mesmos moldes do KRP para o início de um novo ciclo de simulação de conjunto de dados. Tal processo permite que se faça uma análise contínua dos dados entre testes como um procedimento diferente dos sistemas implementados pelos trabalhos de referência, onde todos os testes podem ser considerados como um ciclo contínuo de funcionamento de longo prazo.

Devido a um erro no modelo de robô *iRobot® Create* ou no software de simulação Webots®, a cada reposicionamento do robô a referência de centro do mesmo sofre um pequeno deslocamento, que pode comprometer os dados durante uma simulação prolongada. O problema de deslocamento do centro do robô pode ser evitado com o supervisório realizando uma reinicialização de todo o ambiente de simulação periodicamente.

Outro erro que ocorre no sistema de simulação e não pôde ser identificado neste trabalho, causa o fechamento inesperado do software Webots®, interrompendo toda a simulação abruptamente e perdendo todos os dados de simulação. Não foi identificado nenhum procedimento para evitar que o problema de fechamento aconteça, mas para evitar que todo o trabalho seja perdido caso aconteça o erro de fechamento inesperado, o supervisório salva os dados de simulação periodicamente.

Frente a tais erros, a cada 200 testes o sistema supervisório realiza os seguintes passos para manter a qualidade dos dados aferidos pelo sistema e não perder o progresso realizado:

a) Interrompe a simulação e o controlador *iRobot® Create*;
b) Abre o arquivo CSV de dados simulação e inclui os dados obtidos de simulação para liberar memória RAM do sistema e evitar que os dados simulados até então sejam perdidos;
c) Abre o arquivo CSV de log de simulação e atualiza com progresso atual;
d) Avalia se o teste finalizou e se deve continuar;
e) Se ainda não finalizado, redefine o ambiente de simulação no sistema Webots®, com todos os parâmetros do ambiente de simulação e posicionamento dos objetos;
f) Reposiciona o robô aleatoriamente para iniciar um novo teste;
g) Reinicializa o controlador e a simulação.

O processo será repetido até obter 10.000 conjuntos de dados de treinamento, 2.500 dados de validação e 2.500 dados de teste para cada configuração de APs a serem avaliadas neste estudo, baseado no trabalho de Hoang et al. (2019). Os dados serão então utilizados para o treinamento tanto do sistema LSTM quanto do sistema GRU, e para montar a base de vizinhos do sistema KNN.

Para economizar tempo de processamento, os dados foram tratados previamente em arrays da biblioteca numpy[4] e salvos em arquivos NPY, biblioteca em python para o tratamento de listas de dados, e que podem ser processados diretamente na biblioteca Keras[5] para construção dos modelos. Os dados são organizados em dados de treinamento, validação e teste e salvos em pastas por tipo de teste, de modo que apenas uma mudança de variável de inicialização muda a referência de dados a ser utilizada.

4.7 Preparação da base de dados para treinamento das RNRs

Como primeiro passo da experimentação, temos o trabalho de preparação dos dados gerados na simulação para parâmetros que possam gerar bons resultados nos sistemas envolvidos neste estudo.

A preparação envolve aplicação dos seguintes passos:

a) Leitura dos dados pré-processados em dataframes do Pandas[6];

[4]Biblioteca Numpy, disponível em: `https://numpy.org/`
[5]Biblioteca Keras, disponível em: `https://keras.io/`
[6]Biblioteca Pandas, disponível em: `https://pandas.pydata.org/`

b) Separação dos dados que serão utilizados em pacotes de treinamento, avaliação e teste. Este passo deve ser realizado antes da normalização para evitar vazamento de informações entre pacotes e enviesar os resultados;
c) Separação dos dados entre dados de entrada e dados de saída esperada;
d) Preparação dos formatos de dados a serem trabalhados nos sistemas de localização, vetorizando corretamente cada teste;
e) Normalização dos dados de RSS;
f) Criação da pasta para inclusão dos dados processados e arquivos com cada conjunto de dados.

Para a vetorização dos dados, eles são agrupados em tensores para inclusão no processamento do sistema de treinamento da RNR.

Durante a normalização, os dados apresentados de RSS, que variam entre 0dBm e -100dBm, são recalculados para ficar entre os valores de 0 a 1, facilitando o treinamento das RNRs por remover pesos de unidades de medida e evitando que o treinamento fique preso em pontos ótimos locais (KAUFMAN; ROUSSEEUW, 2009). Não é necessário utilizar nenhum método de remoção de *outliers* pois os valores de RSS medidos no *fingerprinting* e utilizados como base para a simulação já são baseados em médias de medições.

Neste estudo os dados serão testados em 2 tipos de agrupamento diferentes. O primeiro agrupando os 30 segundos de informação de cada teste realizado e tratando os dados de cada teste como uma instância única. O outro agrupamento considera todos os dados da base como um fluxo contínuo, denominado neste estudo como *stream* de dados, com tempo de memória de 10 segundos, conforme os parâmetros apresentado por Hoang et al. (2019), com a sequência a ser apresentada para os sistemas de localização sendo o próximo instante de processamento da simulação, independente da indicação de cada teste no banco de dados.

4.7.1 Separação de dados em pacotes de 30 segundos de cada teste

A separação dos dados em pacotes de 30 segundos segue o determinado para cada teste simulado, conforme citado na rotina de testes e baseado no trabalho de (HOANG et al., 2019), mas adaptado para a condição deste trabalho, com a ocorrência do KRP. O trabalho de referência considera 10 segundos de teste para convergência da localização, mas sem considerar a possibilidade de ocorrência de um KRP. Devido à maior complexidade aplicada neste estudo, cada teste foi considerado como um período de 30 segundos, composto por:

a) Período de 0 a 10 segundos: Período de convergência da localização inicial, conforme especificado por (HOANG et al., 2019);
b) Período de 10 a 20 segundos: Período de aplicação do KRP, em instante aleatório determinado pelo supervisório na simulação;
c) Período de 20 a 30 segundos: Período para recuperação dos sistemas e determinação da nova localização, seguindo os passos do primeiro item.

Como base de comparação, tanto durante o treinamento quanto durante a validação a testes, os sistemas de localização receberão todo o período de cada teste (30 dados, com cada dado um equivalente a 1 segundo) como entradas do sistema, que deverá determinar a localização final do robô no instante de 30 segundos.

4.7.2 Separação de dados em *stream* de dados de 10 segundos

A separação dos dados por *stream* de dados de 10 segundos funciona como um sistema de funcionamento contínuo, baseado na rotina de simulação para a geração dos dados no simulador Webots®.

Cada instante apresentado para os sistemas de localização é um pacote com o histórico dos últimos 10 segundos, desta vez acompanhando o tamanho da entrada de dados RSS definido por (HOANG et al., 2019). Além da diferença na quantidade de dados apresentada para os sistemas de localização, ao invés de cada conjunto de dados ser apresentado como um pacote independente de teste, sem nenhuma relação com o pacote anterior, os dados apresentados em sequência no *stream* de dados parte do instante seguinte, contendo ainda 9 instantes do teste anterior.

Tal composição permite manter-se o histórico nos mesmos moldes do estudo de Hoang et al. (2019) e mantendo a possibilidade de treinar casos onde ocorreram KRP, mas devido ao tamanho do histórico apresentado, nem sempre um conjunto de de dados apresentará um KRP, incrementando a variabilidade do sistema durante o treinamento, com exemplos de funcionamento do robô sem ocorrências de KRP nas bases de treinamento, validação e testes.

Tal base de dados, apesar de simplificar o processamento por se tratar de 10 segundos de obtenção de RSS, aumenta em quase 30 vezes a quantidade de dados de teste e por consequência aumenta consideravelmente o tempo de treinamento de cada RNR e o tempo de processamento do sistema KNN, conforme detalhado no embasamento teórico da solução.

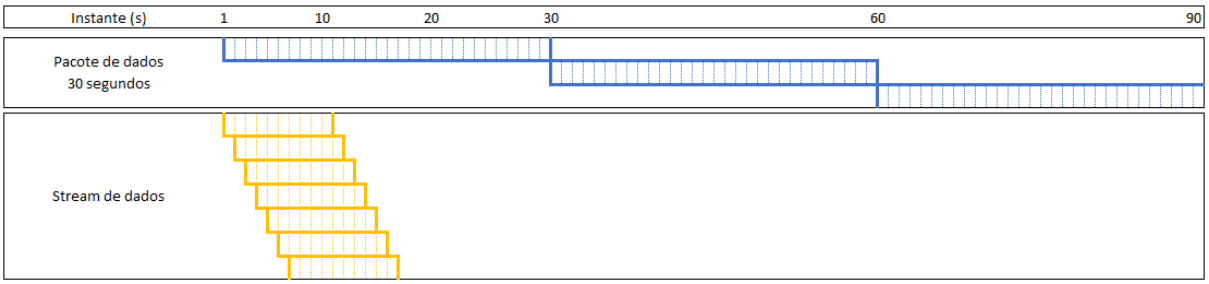

Figura 19 – Descrição dos tipos de separação de dados.

Fonte: Autor.

Como determinado na construção do supervisório de simulação, na transição entre testes é realizado um procedimento nos moldes do KRP que somente será visualizado pelo método de separação de dados com *stream* de dados e será mantido como uma ocorrência de KRP quando presente.

A figura 19 demonstra de modo gráfico a diferença entre os métodos de distribuição dos dados para o mesmo conjunto de dados de simulação em um determinada linha do tempo, com cada linha determinando um conjunto de dados a ser apresentado para os sistemas de localização e cada caixa representando um instante de simulação nos moldes da tabela 4.

4.8 Sistema de localização KNN

A implementação do sistema utilizando KNN será baseado no trabalho de Xu e Chou (2017), que determinaram uma solução utilizando KNN para identificar com baixa precisão a localização de um dispositivo Wi-Fi.

Neste trabalho o processo de *fingerprinting* proposto por Xu e Chou (2017) será substituído pelo processo de *fingerprinting* realizado no ambiente real descrito anteriormente para manter comparabilidade entre as outras soluções deste estudo.

Os dados de sensoriamento do sistema proposto por Xu e Chou (2017) serão substituídos pelas informações fornecidas pelo ambiente Webots® e o controlador *iRobot® Create*, mantendo a funcionalidade da solução apresentada dentro do ambiente virtual até o instante (c) identificado na figura 11 e identificado na arquitetura da figura 12 como “Fingerprint Coarse Localization”, devido ao procedimento utilizando Laser Range Finder para refinamento da localização (identificado na figura 12 como “AMCL Fine Localization”) sair do escopo determinado por este estudo.

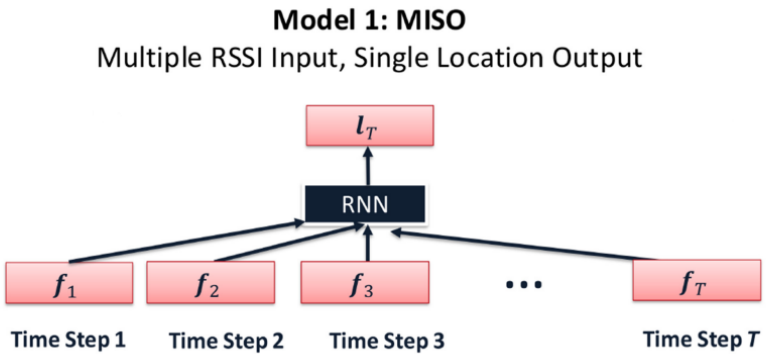

Figura 20 – Exemplo de fluxo de dados no sistema MISO.

Fonte: Hoang et al. (2019).

4.9 Sistema de localização LSTM

Baseado no sistema de localização definido por Hoang et al. (2019) como *Multiple RSSI input, Single Location Output* (MISO), e detalhado na figura 20, será construído um sistema LSTM para identificação da localização do robô, adotando todos os parâmetros estudados pelo trabalho e definidos pela tabela 2. Os dados de treinamento e validação serão gerados a partir do *fingerprinting* obtido no ambiente real e tratados no próprio sistema simulador, citado na seção de *Access Points* e *fingerprinting*.

A RNR será construída na linguagem python, utilizando a biblioteca Keras[5], e o modelo treinado com os dados de treinamento, avaliação e teste previamente gerados no simulador será guardado em um arquivo no formato HDF5 para utilização na fase de avaliação dentro do simulador Webots®.

Caso o modelo LSTM não apresente resultados satisfatórios comparados ao trabalho original (HOANG et al., 2019), será realizado um novo treinamento dos sistemas.

4.10 Sistema de localização GRU

Para efeitos de comparabilidade da eficácia de cada RNR, o sistema de localização desenvolvido com GRU será construído em python com a biblioteca Keras[5], apresentará os mesmos parâmetros da tabela 2 e treinado com os mesmos dados de treinamento, validação e testes apresentados no sistema LSTM.

Tabela 2 – Parâmetros de configuração iniciais para o sistema LSTM

Categoria	Valor
Testes para avaliação	10
Tipo da RNR	LSTM
Comprimento da memória (T)	30 / 10
Modelo	MISO
Função de Perda	RMSE
Camadas ocultas	2
Quantidade de neurônios de cada camada oculta	100
Decaimento	0,2
Otimizador	Adam
Taxa de aprendizado	0,001
Número de trajetos de treinamento para o teste 1-fold	10.000 a 300.000
Número de épocas de treinamento	1000
σ	2 m
Δ_t	1 s
d_{max}	2 m

Apenas durante a avaliação da eficácia final do sistema pronto, cada solução atuará em diferentes situações simuladas, devido à aleatoriedade automatizada do sistema de rotas, KRP e geração de perturbações.

Assim como citado para o sistema LSTM, caso haja problemas em obter dados satisfatórios de localização, um novo ajuste de parâmetros será avaliado para treinamento.

4.11 Rotina de Testes

Para a realização dos testes, o robô será inicializado em um local aleatório diferente do último local onde esteve em operação e sua movimentação começará no instante inicial seguindo a lógica do controlador "create_avoid_obstacles.c", incluso no ambiente de simulação Webots®, com um sistema supervisório dentro do próprio simulador Webots® atuando a cada segundo para obter a localização real durante um período inicial aleatório entre 10 a 20 segundos de operação para atingir a precisão ótima do sistema de localização e avaliar se houve algum *overfitting* sobre o KRP no instante de 15 segundos. Em seguida o robô será instantaneamente reposicionado em outra localização (ao menos 3 metros distante da localização atual) e com outra orientação, simulando o sequestro de robôs, sem nenhuma notificação prévia ou posterior do ocorrido ao sistema.

A partir dos sinais de Wi-Fi obtidos com o simulador de sinais, o localizador deve obter o RSS de cada AP de referência a partir do simulador e avaliar a resposta do sistema. Serão

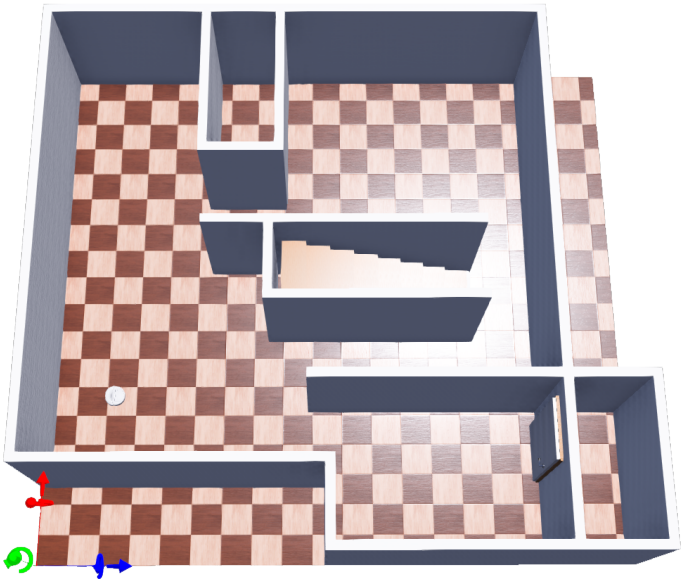

Figura 21 – Imagem do simulador em operação. As flechas vermelha, azul e verde definem o eixo de orientação no ambiente de simulação.

Fonte: Autor.

catalogados os dados de localização correta, predição do sistema localizador e precisão obtida a cada iteração e, no final, o tempo para atingir ao menos 2 metros de convergência, conforme definido por Hoang et al. (2019). O processo deve ser repetido por ao menos 10 vezes com o conjunto de teste para cada configuração de APs para avaliação de diversos casos e geração de dados médios, sempre utilizando os mesmos modelos treinado em cada sistema de localização como base de comparação.

Para a definição de cada configuração de APs de teste, será utilizado para a seleção de configurações de APs um número binário de 6 bits, cada bit representando um AP, gerando 63 combinações diferentes eliminando a possibilidade de nenhum AP, numerados conforme localização definida no mapa de *heatmaps* e exemplificados pela tabela 3 e atuando do mesmo modo que atenuador de sinal proposto por (ABBAS et al., 2019) dentro da simulação do gerador de caminhos e no procedimento de teste. A seleção de configuração de teste será inversamente sequencial, partindo da configuração 63 (com todos os APs acionados) até a configuração 1 (com apenas o AP 1 em funcionamento. O valor obtido será colocado bit a bit em um vetor que será escalarmente multiplicado com o vetor de dados base RSS obtidos no *fingerprinting*. O valor 0 será desconsiderado no sistema aleatório por significar nenhum AP disponível, situação que não permite nenhum tipo de sinal ser gerado para testar o sistema de localização.

Paralelamente ao simulador, um log de dados capturado pelo supervisório memorizará os dados descritos na tabela 4 em formato CSV (*Comma Separated Values*) para a realização dos

Tabela 3 – Exemplos de combinações da APs para teste

Configuração	Quant APs	binário	AP1	AP2	AP3	AP4	AP5	AP6
63	6	111111	X	X	X	X	X	X
62	5	111110	X	X	X	X	X	
61	5	111101	X	X	X	X		X
60	4	111100	X	X	X	X		
59	5	111011	X	X	X		X	X
58	4	111010	X	X	X		X	
.	.	.						
.	.	.						
.	.	.						
06	3	000111				X	X	X
05	2	000101				X		X
04	1	000100				X		
03	2	000011					X	X
02	1	000010					X	
01	1	000001						X

gráficos comparativos e conclusão do estudo. Devido ao comportamento aleatório do momento em que ocorrerá o KRP, a cada instante do log será incluso um valor booleano para indicar se o KRP já ocorreu para facilitar a identificação do instante exato em que ocorreu.

O supervisório também realizará um levantamento do progresso da simulação e o horário de cada evento para armazenar em um outro arquivo CSV, caso haja algum problema na simulação e ela seja interrompida, o simulador poderá retomar de onde parou. Na inicialização dos sistemas, antes de começar a simulação, o supervisório avalia até que ponto está a simulação e reinicializa os sistemas de simulação a partir do último ponto processado. Tal processo resolveu um problema no ambiente de simulação Webots® que apresenta uma falha não identificada e fecha depois de um grande volume de pedidos de redefinição de simulação realizados pelo supervisório. O código do simulador e do sistema de localização está disponível na plataforma *GitHub*[7].

Tais dados serão salvos em arquivos CSV separados para cada configuração de teste descrita na tabela 3 para uso com treinamento, validação ou teste depois de convertidos para dados NPY para utilização nos sistemas RNR e KNN.

[7]Repositório *GitHub* com o código utilizado no trabalho: `https://github.com/antoniopegorelli/RSS-KRP-with-LSTM-GRU-and-KNN`

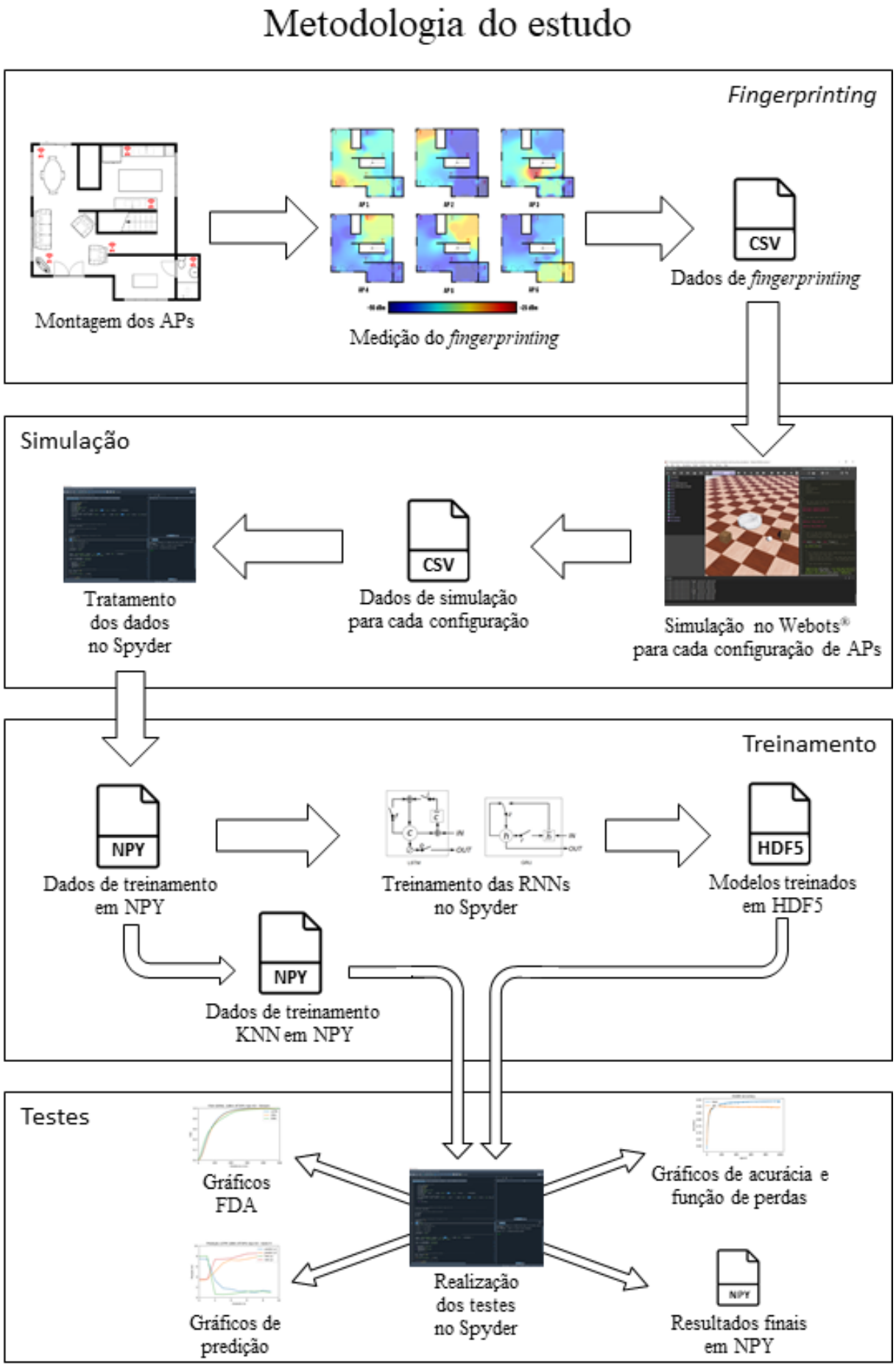

Figura 22 – Metodologia completa do método deste estudo. O código utilizado neste estudo está disponível na plataforma *GitHub*[7]

Fonte: Autor.

Tabela 4 – Dados capturados pelo log de avaliação

Dado	Unidade	Variável
Teste	sequencial	Inteiro
Tempo	segundos	Inteiro
Posição x	metros	Ponto flutuante
Posição y	metros	Ponto flutuante
KRP	Verdadeiro/ Falso	Booleano
RSS medido [6]	dBm	Ponto flutuante[6]

4.12 Parâmetros de teste e variações

Para realização das variações em cada teste e evitando que a combinação de fatores causem erros na avaliação dos efeitos de cada parâmetro no contexto, apenas um parâmetro será mudado por vez, enquanto todos os outros parâmetros serão mantidos nos parâmetros base da tabela 5. Os dados controláveis do simulador serão utilizados como parâmetros de teste para realização de cada variação durante a avaliação de performance dos modelos treinados. O modelo base será a condição inicial testada, baseado nos trabalhos realizados por Hoang et al. (2019), com os parâmetros definidos na tabela 5:

Tabela 5 – Parâmetros de teste avaliados no simulador para treinamento dos modelos base

Estrutura de dados	Quantidade de APs	Ruído	Atenuação
Stream de dados 10 segundos	6 APs	2 dBm	10%

Para a análise de performance foram gerados gráficos comparativos da função de distribuição acumulada entre diversas composições que serão apresentadas nas subseções seguintes, além de apresentar gráficos comparativos entre predição e localização em certos períodos de teste em que pode-se observar fenômenos que justificam os resultados obtidos nos testes. As composições avaliadas neste estudo são:

a) Variação no método de apresentação de dados
   - Pacotes de cada teste de 30 segundos de histórico
   - *stream* de dados com 10 segundos de histórico

b) Variação na quantidade de APs disponíveis durante o teste
   - 3 APs disponíveis em posição de triangularização
   - 1 AP centralizado no ambiente
   - 1 AP no canto do ambiente

c) Variação no nível de ruído gaussiano aplicado no simulador

2dBm de ruído aplicado independente por AP

10dBm de ruído aplicado independente por AP

d) Variação no nível de atenuação aplicado no simulador

20% de atenuação aplicada independente por AP

30% de atenuação aplicada independente por AP

Para facilitar as comparações entre modelos treinados, as situações citadas na lista acima foram testadas adicionalmente às soluções avaliadas utilizando os dados relacionados a cada modo, uma condição base, que é definida pelos parâmetros definidos na tabela 5.

4.12.1 Gráficos de função de distribuição acumulada

Os gráficos de FDA, conforme detalhado nos conceitos fundamentais deste estudo, apresentam a performance de cada sistema na condição de teste para realização de uma análise comparativa com cada série apresentando o resultado geral de cada proposta avaliada.

Adicionalmente aos dados avaliados com o modelo base, um modelo específico treinado com dados simulados a partir dos mesmos parâmetros de teste será testado para cada avaliação realizada, cada um com seu próprio gráfico de FDA.

Nos gráficos finais comparativos de FDA da conclusão, todos os modelos de proposta de cada teste são apresentadas para comparação, com as séries apresentadas com "_O" no final da legenda representam o resultado do teste utilizando o modelo base, treinado com os parâmetros base de teste, conforme a tabela 5.

4.12.2 Gráficos de avaliação dos testes

Para realização da avaliação de fenômenos observados durante os testes, foram criados gráficos comparativos de posicionamento em função do tempo, conforme figura 23, onde temos determinado na legenda como *"Predição"* os dados de predição do sistema avaliado, e as linhas identificadas como *"Real"* identificando a localização do robô no ambiente de simulação para os instantes avaliados. A linha roxa nomeada como *"KRP"*, identifica o momento em que ocorreu o sequestro do robô, facilitando a visualização. Cada gráfico de avaliação detalhada de períodos de teste apresenta um período de 10 segundos de teste para avaliação dos resultados em um certo período dos testes para realização de observações.

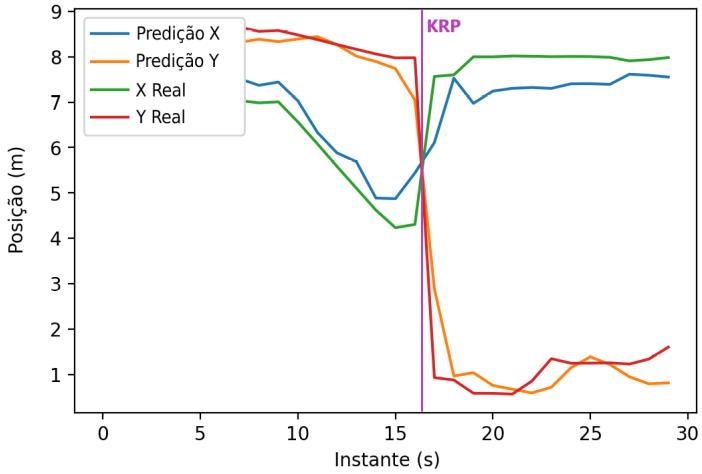

Figura 23 – Exemplo de gráfico de avaliação.

Fonte: Autor.

Nos gráficos de avaliação, os valores de x e y determinam a localização do robô nos eixos de saída, conforme identificado na figura 21, onde temos o eixo de orientação no simulador, com a flecha azul identificando o eixo x e a flecha vermelha y. Os quadrados do piso no simulador são de tamanho 0,5m por 0,5m, facilitando a visualização da localização do robô no simulador durante seu funcionamento.

Para análise dos dados, quanto mais próxima a linha *"Predição X"* (em azul) estiver da linha *"X Real"* (em verde) e a linha *"Predição Y"* (em laranja) estiver da linha *"Y Real"* (em vermelho), mais próxima está a predição do sistema avaliado.

Todas as análises deste estudo apresentam períodos com uma ocorrência de KRP, indicada pela linha roxa nas figuras, mantendo as análises dentro do escopo do trabalho. Fora caso específico em que o KRP não faça parte da demonstração da análise em questão.

5 Resultados experimentais

Neste capítulo serão apresentados os resultados de função de distribuição acumulada para cada simulação realizada neste estudo, com algumas variações determinadas interessantes a serem testadas para avaliação da robustez dos sistemas implementados e análise dos comportamentos de cada solução.

Serão apresentados também gráficos de avaliação de testes, demonstrando comportamentos observados durante o processo.

5.1 Treinamento do sistema KNN

Para a realização do treinamento do sistema KNN foi implementado uma lista contínua com todos os dados de treinamento de cada simulação, onde os dados de entrada de cada teste realizado é comparado com todos os dados de treinamento e determinada a distância. Devido a este processo, não há pré treinamento do sistema, em contrapartida exigindo maior poder de processamento para os testes. Para realização dos testes e com base no trabalho de Xu e Chou (2017), que determinou o valor de $k >= 2$, foram avaliados os valores de vizinhos k entre 2 e 5, com o valor de $k = 2$ apresentando os melhores resultados para a base de treinamento, apesar da baixa variabilidade dentro dos valores avaliados, conforme analisado na figura 24.

Não foram avaliados valores de k maiores de 5 pois o erro apenas aumenta com o aumento da quantidade de vizinhos e a grande base de treinamento já é capaz de apresentar resultados consistentes na regressão com o valor de $k = 2$.

5.2 Resultados de treinamento e progressão dos valores de acurácia e função de perdas das RNRs

Como primeiro passo nas análises do treinamento, avaliou-se a progressão da acurácia e função de perdas para ambas as RNRs GRU e LSTM, tanto para a apresentação dos dados em pacotes de 30 segundos, como no *stream* de dados.

Para manter a comparabilidade entre as RNRs, os dados de progressão de acurácia e perdas foram avaliados na mesma base de dados de treinamento, avaliação e teste, com 1dB de ruído e 10% de atenuação independente, apenas modificando o método de agrupamento e

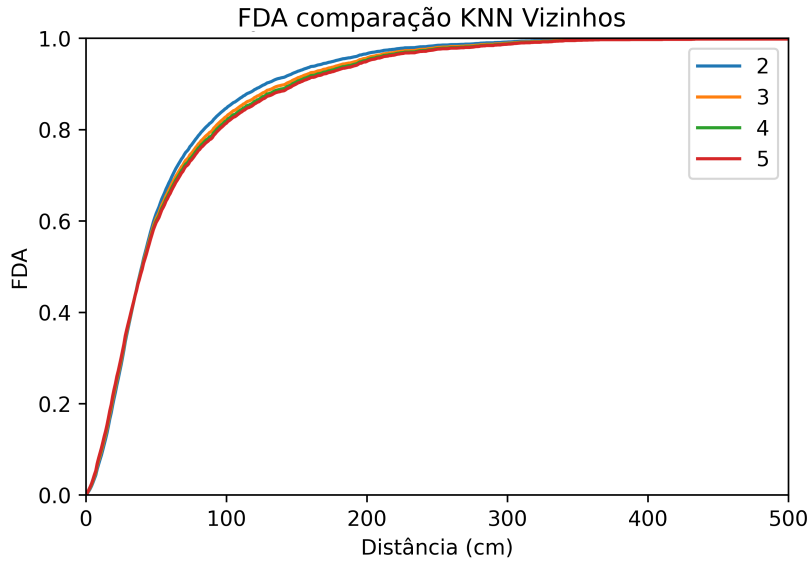

Figura 24 – Comparação dos resultados de predição para ao sistema KNN com diferentes valores k de vizinhos, para a base de dados com *stream* de dados, todos os APs, 1dBm de ruído, e 10% de máscara de atenuação.

Fonte: Autor.

apresentação dos dados para os sistemas de localização propostos, conforme detalhado na figura 19.

5.2.1 Separação de dados em pacotes de 30 segundos de cada teste

Como detalhado na metodologia, a separação dos dados em pacotes de 30 segundos cada consiste em apresentar para os sistemas de localização em cada instante de treinamento uma lista de 30 dados RSS em conjunto com a respectiva localização. As avaliações seguintes foram realizadas treinando os modelos com os parâmetros base de teste.

5.2.1.1 Treinamento do sistema GRU

A partir das observações realizadas sobre os dados de acurácia e função de perdas durante o treinamento do sistema GRU para a apresentação de dados em pacotes de 30 segundos de cada teste, observa-se que, conforme a figura 25, a partir da época de treinamento 50 há um descolamento dos resultados entre os dados de treinamento e validação, indicando que há uma tendência de *overfitting* para esse tipo de dados no sistema GRU. O *overfitting* é identificado

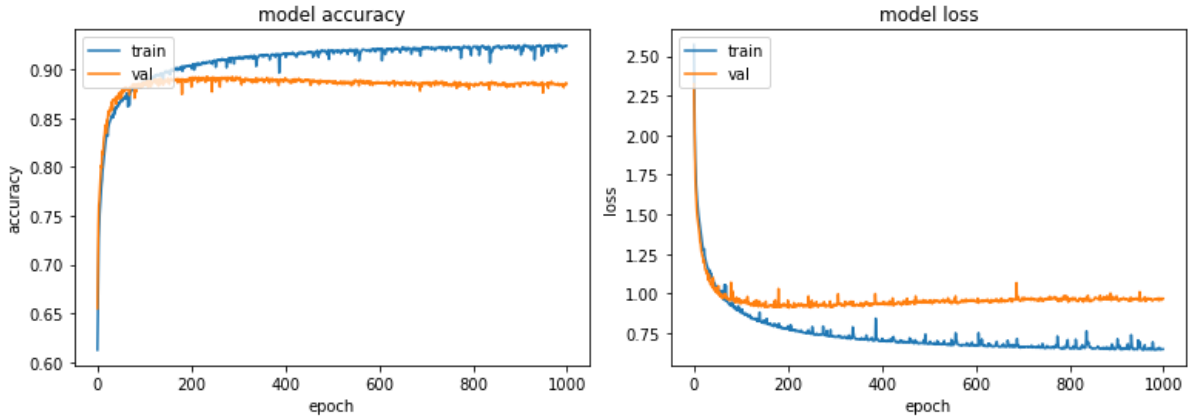

Figura 25 – Resultado da progressão da acurácia e função de perdas durante o treinamento do sistema GRU para os dados de treinamento e validação em pacotes de 30 segundos de cada teste.

Fonte: Autor.

devido à desconexão da acurácia e função de perda entre dados de treinamento e simulação, com resultados muito melhores quando avaliados os dados de treinamento.

5.2.1.2 Treinamento do sistema LSTM

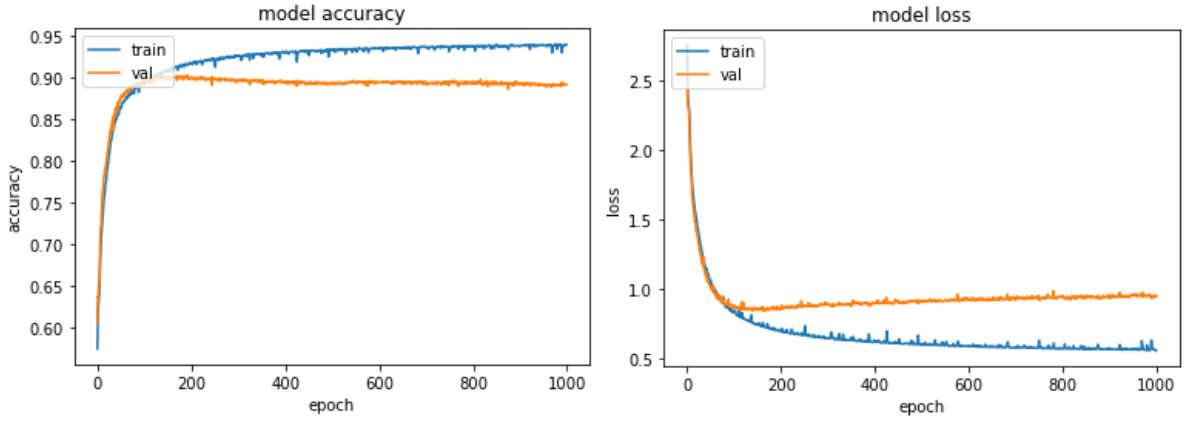

Figura 26 – Resultado da progressão da acurácia e função de perdas durante o treinamento do sistema LSTM para os dados de treinamento e validação em pacotes de 30 segundos de cada teste.

Fonte: Autor.

Quando realizado o treinamento do sistema baseado na RNR LSTM, observa-se o mesmo comportamento na figura 26, quando o comportamento dos dados de acurácia e função de perda também apresentam um deslocamento, mas no caso da LSTM ele ocorre na época de treina-

mento 100, comprovando que o sistema LSTM é um pouco mais resistente ao *overfitting* que o observado durante o treinamento da GRU, quando o fenômeno se iniciou já na época 60, realizando o treinamento com os mesmos dados. Mas de qualquer modo, ainda observa-se um expressivo deslocamento da acurácia e função de perdas entre dados de treinamento e validação, indicando que mesmo assim ainda há *overfitting* no modelo treinado.

5.2.2 Separação de dados em *stream* de dados de 10 segundos

5.2.2.1 Treinamento do sistema GRU

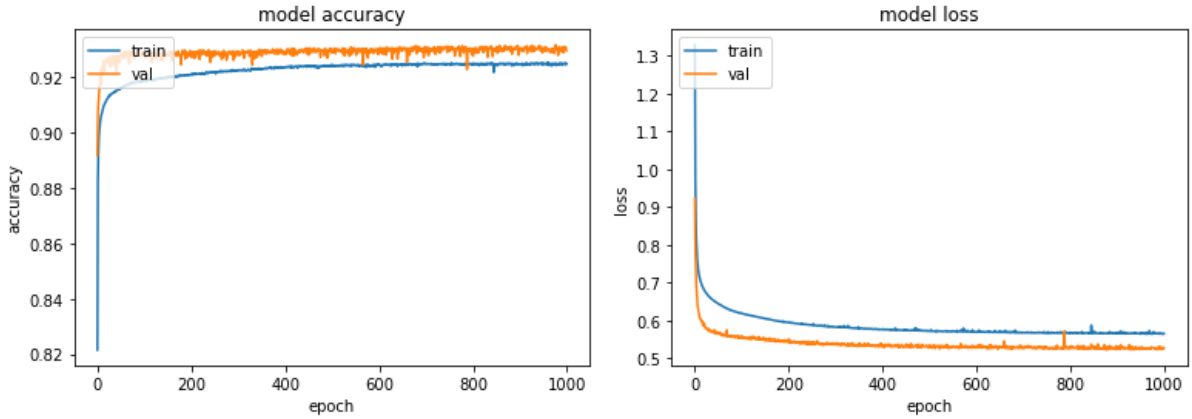

Figura 27 – Resultado da progressão da acurácia e função de perdas do treinamento do sistema GRU para os dados de treinamento e validação com *stream* de dados.

Fonte: Autor.

Quando apresentamos os dados de treinamento com os parâmetros base de teste em forma de *stream* para o sistema GRU, observou-se uma performance inicial melhor para os dados de validação do que com os próprios dados de teste.

5.2.2.2 Treinamento do sistema LSTM

Para o sistema baseado na RNR LSTM, os dados obtidos apresentam comportamento semelhante ao verificado no sistema GRU, mas um resultado um pouco melhor em geral.

Tal comportamento deve-se aos seguintes fatores, conforme observado por Geron (2019):

a) Regularização dos dados, que acontece durante o treinamento, mas não durante a validação;

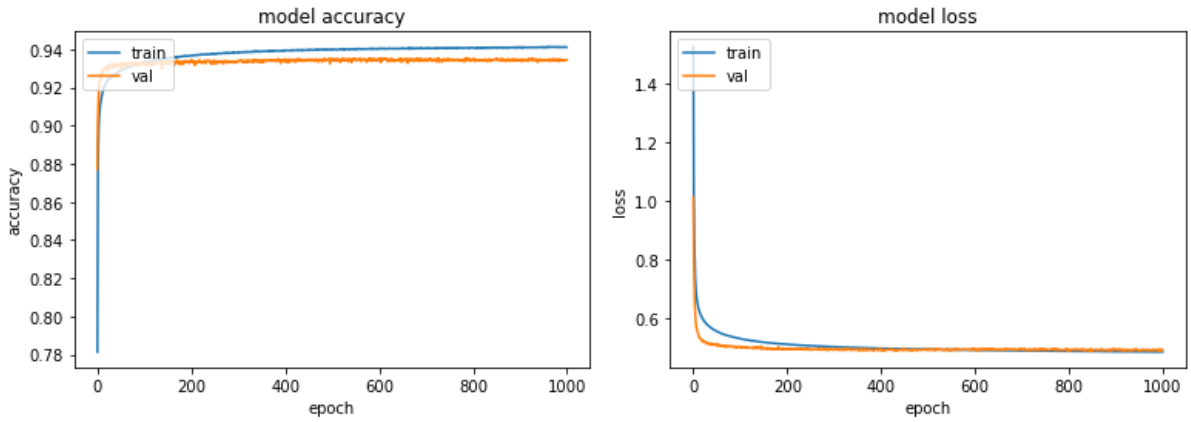

Figura 28 – Resultado da progressão da acurácia do treinamento do sistema LSTM para os dados de treinamento e validação com *stream* de dados.

Fonte: Autor.

b) A obtenção dos dados acurácia e função de perda para os dados de treinamento ocorre durante o processamento da época, ou seja, sua média situa-se no meio de cada época e não na sua conclusão;

c) Os dados de acurácia e função de perda para os dados de validação ocorrem no final de cada época;

d) A base de dados de validação pode ser mais fácil para avaliação que os dados de treinamento;

Mesmo com a acurácia e a função de perdas dos dados de validação e de teste se aproximando, não podemos afirmar que não houve *overfitting* durante o treinamento (GERON, 2019).

5.3 Análise da progressão e limite de estabilização do treinamento para o *stream* de dados

Com a capacidade de gerar grande volume de dados para os conjuntos de *stream* de dados, devido ao processo de seleção de conjuntos de dados reutilizar grande parte das informações do instante anterior no próximo, conforme figura 19, é possível observar o ponto de limite de treinamento, onde não houve mais benefícios em acurácia nem em função de perdas. Como é possível observar na figura 28, a partir da época 600 não houve mais ganhos em acurácia nem na função de perdas, estabilizando até o final do processo de treinamento.

5.4 Resultados de performance com variação na composição dos dados apresentados

5.4.1 Separação de dados em pacotes de 30 segundos de cada teste

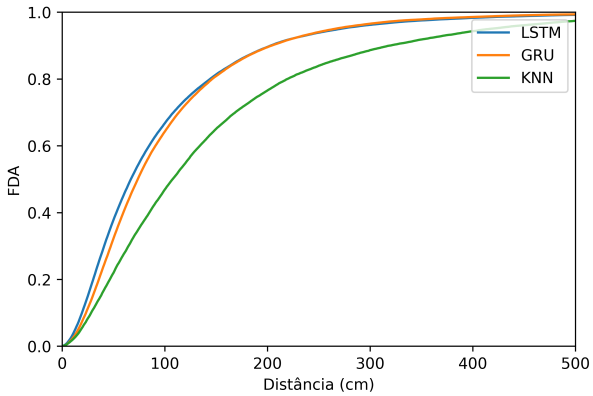

Figura 29 – Resultado comparativo da FDA, utilizando pacotes de 30 segundos.

Fonte: Autor.

Na seção anterior, durante o treinamento com separação de dados em pacotes de 30 segundos, já podíamos observar um erro médio maior que o esperado, conforme resultados da figura 29. Tal comportamento se comprovou durante os testes, confirmando que houve um leve *overfitting* durante o processo, comprometendo os resultados quando organizamos os dados por este método. Para o sistema KNN, por não avaliar histórico de dados para gerar uma predição, ele não sofre influência do método de agregação da base de dados. Abaixo estão os resultados obtidos de média e desvio padrão para cada sistema avaliado.

Resultados obtidos de erro médio e desvio padrão durante os testes de cada proposta para a condição atual, com os dados estatísticos na figura 30:

a) LSTM: 0,97m $\pm$ 0,94m

b) GRU: 1,00m $\pm$ 0,90m

c) KNN: 1,47m $\pm$ 1,31m

Os resultados acima possuem média próxima ao avaliado por (HOANG et al., 2019) em seus estudos para a solução LSTM com arquitetura MISO, mas o desvio padrão subiu em 24,3%, principalmente devido à oscilação durante os períodos de estabilidade, onde as soluções tiveram maior dificuldade de melhorar a precisão, tanto para a LSTM quanto para a GRU, conforme observa-se na figura 31. Mesmo apresentando resultados semelhantes, devemos considerar que o resultado do teste acima foi pior que trabalho de referência, pois foi realizado em

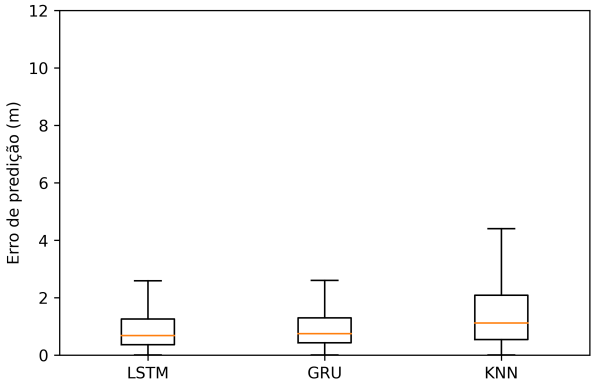

Figura 30 – Comparação estatística dos resultados, utilizando pacotes de 30 segundos. Na análise ANOVA, temos $F = 3539$ e $p = 0{,}0$, indicando que os resultados possuem desempenho significativamente diferentes.

Fonte: Autor.

ambiente controlado por simulador, com dados de perturbação gerados por uma média controlada e sem considerar *outliers* devido a distorções fora do padrão.

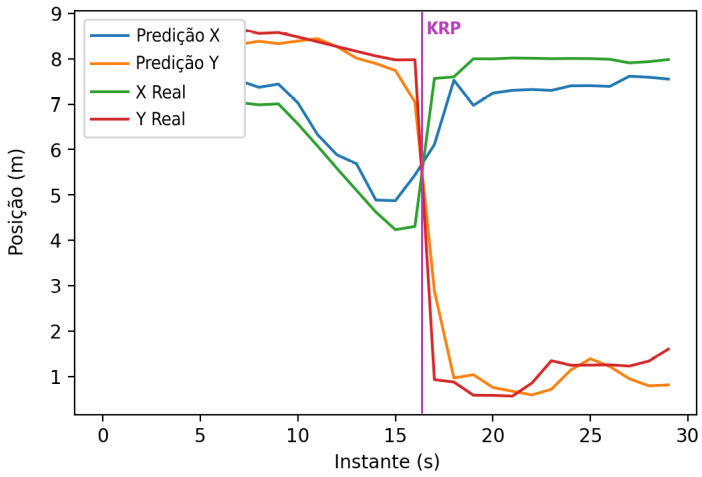

Figura 31 – Exemplo de teste da RNR GRU, demonstrando a dificuldade de convergir a predição quando treinado com pacotes de 30 segundos.

Fonte: Autor.

Na análise do resultado da GRU, observa-se uma predição média pior que avaliado na LSTM, com desvio padrão menor, indicando que a solução manteve-se mais tempo próxima do erro médio observado, devido a menor capacidade dos *gates* interpretarem o KRP para se relocalizar.

Como esperado, por ser uma solução mais simples e apenas utilizando comparações com situações semelhantes, o sistema KNN apresentou os piores resultados, devido às grandes distâncias entre vizinhos diretamente comparáveis.

5.4.2 Separação de dados em *stream* de dados de 10 segundos

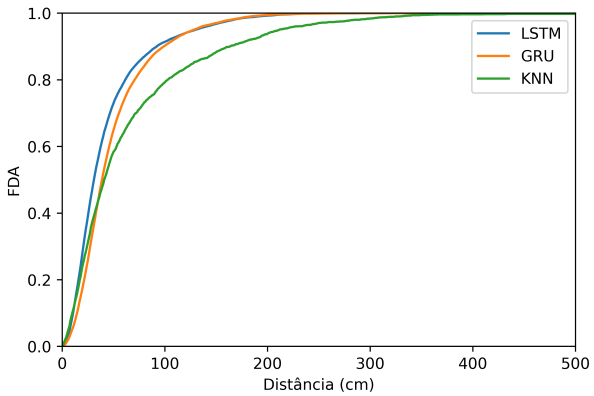

Figura 32 – Resultado comparativo da FDA, utilizando *stream* de dados de 10 segundos.

Fonte: Autor.

Quando os dados foram consolidados em *stream* de dados de 10 segundos (figura 32), o treinamento apresentou melhores resultados de acurácia comparados ao método de agrupamento em pacotes de 30 segundos de cada teste, comprovados quando observados os resultados apresentados.

O sistema KNN apresentou melhores resultados apenas por ser alimentado com maior volume de dados na base de treinamento, aumentando a variabilidade das informações a serem comparadas. e diminuindo a distância entre os 2 vizinhos mais próximos, refinando os resultados médios.

Resultados obtidos de erro médio e desvio padrão durante os testes de cada proposta para a condição atual, com os dados estatísticos na figura 33:

a) LSTM: 0,44m $\pm$ 0,39m

b) GRU: 0,50m $\pm$ 0,38m

c) KNN: 0,68m $\pm$ 0,73m

Os resultados nesse teste foram melhores que o trabalho de referência realizado por (HOANG et al., 2019) devido à avaliação acontecer em um ambiente simulado, onde controlamos o ruído e a atenuação com parâmetros dentro da simulação, limitando as interferências

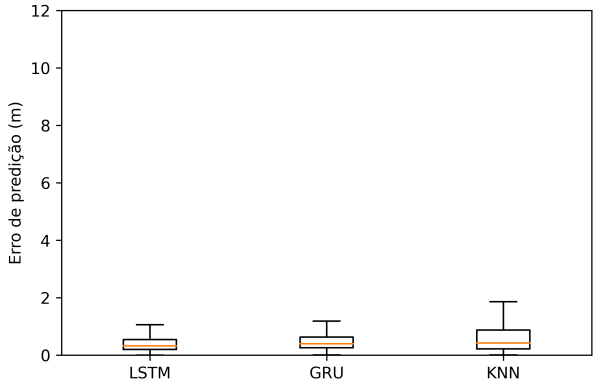

Figura 33 – Comparação estatística dos resultados, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 1367$ e $p = 0{,}0$, indicando que os resultados possuem desempenho significativamente diferentes.

Fonte: Autor.

aplicadas nos sinais interpretados pelos sistemas de localização. Mesmo com maior precisão, o KNN apresenta grande desvio padrão por trabalhar com distâncias em 6 dimensões, uma para cada AP, mantendo uma variação constante em torno dos valores reais de localização.

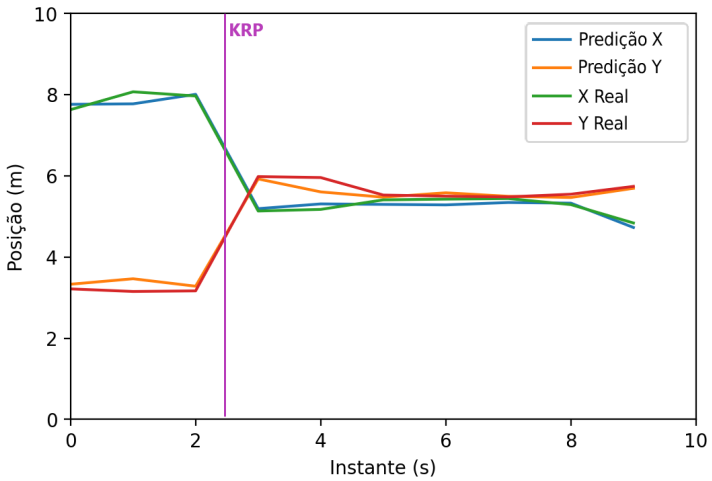

Figura 34 – Resultado do teste com modelo LSTM base.

Fonte: Autor.

Observa-se na figura 34 que o sistema de localização funciona quase sem erro, respondendo rapidamente na próxima iteração assim que ocorre o KRP no instante 3 segundos do gráfico. Tal comportamento se deve ao baixo nível de ruído e atenuação limitado pelo sistema de simulação e pela diversa base de dados de treinamento apresentada ao sistema.

5.5 Resultados de performance com variação na quantidade de APs disponíveis

A primeira condição a ser avaliada neste estudo é relacionado a quantidade de APs disponíveis no ambiente. Tal teste serve para avaliar a capacidade do sistema de continuar prevendo a prosição do robô após um KRP quando parte dos APs deixarem de funcionar.

5.5.1 Simulador operando com 3 APs em triangularização

Para este teste foram escolhidos os APs 1, 4 e 6, localizados respectivamente na sala, na cozinha e no banheiro, conforme figura 13. Tal configuração foi selecionada por apresentar melhor distribuição da variabilidade de sinais, conforme estudo de *fingerprinting* da figura 15.

5.5.1.1 Utilizando modelo específico treinado com os mesmos parâmetros dos dados de teste

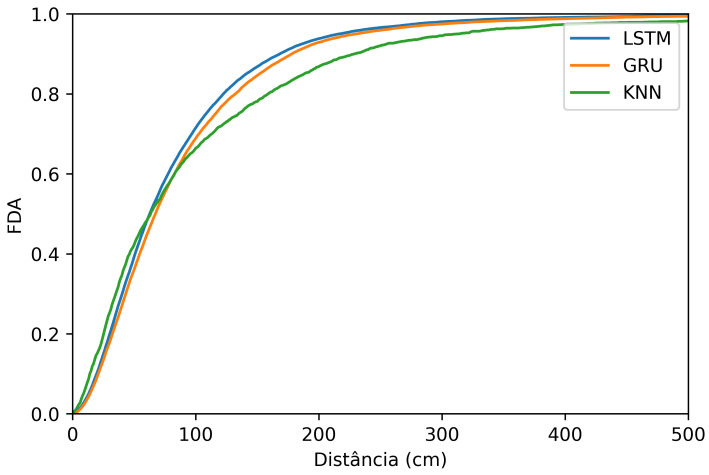

Figura 35 – Resultado comparativo da FDA para os modelos específicos e testados com 3 APs em triangularização.

Fonte: Autor.

Na condição de apenas 3 APs funcionando, gerando uma triangularização no ambiente, ao realizar um novo treinamento das RNRs com a base de dados específica para a condição apresentada, o sistema apresenta uma leve queda na curva do FDA (figura 35), devido ao aumento no tempo de resposta durante o KRP, que subiu de 1 segundo (a unidade mínima entre instantes de medição) para 3 segundos em média, conforme observa-se na figura 37. Tal comportamento

comprometeu o resultado final do teste, piorando parcialmente tanto as médias quanto o desvio padrão em relação ao teste envolvendo todos os APs:

Resultados obtidos de erro médio e desvio padrão durante os testes com 3 APs para os modelos específicos, com os dados estatísticos na figura 36:

a) LSTM: 0,85m $\pm$ 0,77m

b) GRU: 0,91m $\pm$ 0,82m

c) KNN: 1,04m $\pm$ 1,21m

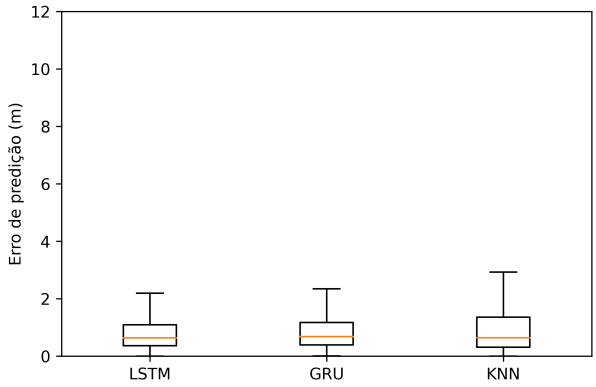

Figura 36 – Comparação estatística dos resultados para os modelos específicos, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 260{,}2$ e $p = 2{,}41e^{-113}$, indicando que os resultados possuem desempenho significativamente diferentes.

Fonte: Autor.

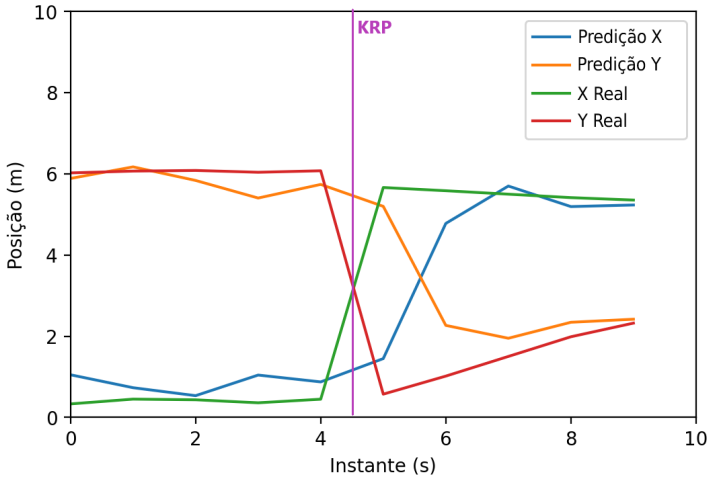

Figura 37 – Análise do resultado dos testes para o modelo específico LSTM e testados com 3 APs em triangularização, demonstrando o aumento de resposta do sistema após um KRP.

Fonte: Autor.

Se comparado com o sistema rodando com todos os APs e utilizando os mesmos parâmetros na simulação, o atraso gera erro, mas após esse período o sistema se mantém com boa precisão até a ocorrência de um novo KRP. No caso do KNN, os resultados apresentaram piora menos significativa em tempo de resposta, entre 1 e 2 segundos, devido ao grande volume de dados apresentados na base de treinamento, mas o KNN tem maior dificuldade para manter precisão durante períodos de estabilidade, sem ocorrência de KRP, comprometendo os resultados médios.

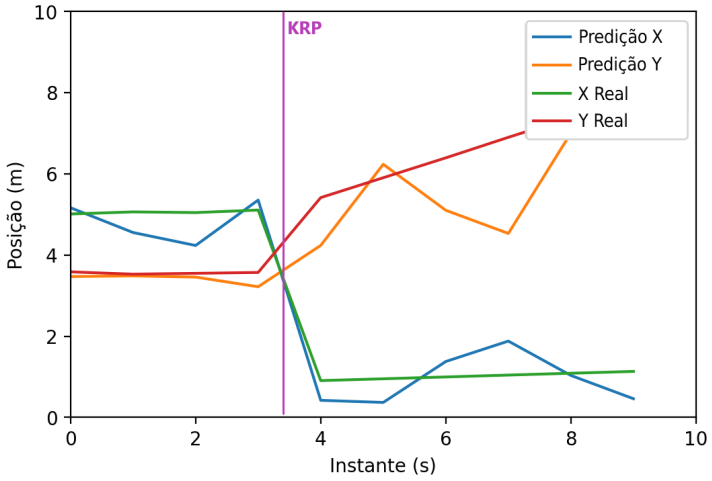

Figura 38 – Análise do resultado dos testes para o modelo específico KNN e testados com 3 APs em triangularização, apresentando dificuldade de em manter precisão.

Fonte: Autor.

5.5.1.2 Utilizando modelo base treinado com os parâmetros **stream** *de dados, todos os APs disponíveis, 1dBm de ruído e 10% de atenuação*

Nos testes com o modelo base (figura 39), já pode-se observar que os sistemas treinados não conseguem converter para uma localização adequada, pois a base de treinamento raramente chega na condição testada, com metade dos APs continuamente desativados e ainda mantendo os mesmos parâmetros para a máscara de atenuação no simulador.

Resultados obtidos de erro médio e desvio padrão durante os testes com 3 APs para os modelos base, com os dados estatísticos na figura 40:

a) LSTM: 6,39m $\pm$ 2,81m

b) GRU: 5,94m $\pm$ 2,73m

c) KNN: 6,05m $\pm$ 2,93m

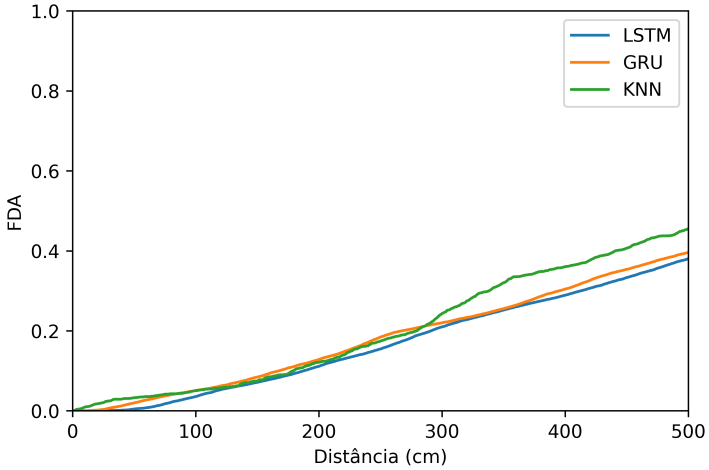

Figura 39 – Resultado comparativo da FDA para os modelos base e testados com 3 APs em triangularização.

Fonte: Autor.

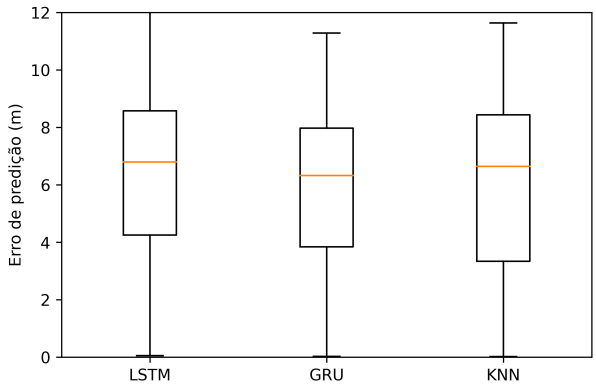

Figura 40 – Comparação estatística dos resultados para os modelos base, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 170{,}9$ e $p = 8{,}64e^{-75}$, indicando que os resultados possuem desempenho significativamente diferentes.

Fonte: Autor.

Nesse momento em que o KNN, apesar de também sofrer com a falta de dados comparáveis na base de treinamento, apresenta performance um pouco melhor que as RNRs, pois leva em consideração apenas o valor atual de sinais de RSS e os compara com os diversos dados da base de treinamento.

Os dados apresentados na figura 41 demonstram a inatividade do sistema de localização, mesmo em momentos de estabilidade de localização, independente da ocorrência do KRP.

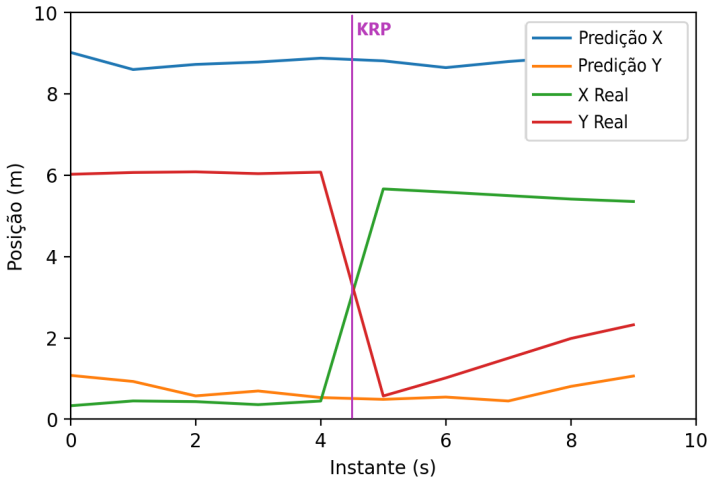

Figura 41 – Análise dos resultados para o modelo base LSTM e testado com 3 APs em triangularização, com o sistema sem nenhuma reação.

Fonte: Autor.

5.5.2 Simulador operando com um único AP centralizado no ambiente (AP 3)

Para o primeiro teste a ser realizado com apenas um único AP foi escolhido o AP 3 (denominado como teste tipo 8, devido ao número binário correspondente ao terceiro bit mais significativo apenas com bit 1), localizado no corredor próximo a escada, devido a sua localização centralizada no ambiente de simulação. Esta configuração apresenta diversos mínimos locais que podem dificultar o processo de treinamento e predição, conforme observado no estudo de *fingerpriting* da figura 15.

5.5.2.1 Utilizando modelo específico treinado com os mesmos parâmetros dos dados de teste

No primeiro teste realizado com apenas 1 AP acionado, é possível observar na figura 42 o impacto no sistema de localização que a falta de APs de referência gera no momento do KRP. O resultado para as RNRs indicou total inoperatividade dos sistemas.

Resultados obtidos de erro médio e desvio padrão durante os testes com um único AP centralizado para os modelos específicos, com os dados estatísticos na figura 43:

a) LSTM: 2,23m $\pm$ 1,44m

b) GRU: 2,26m $\pm$ 1,44m

c) KNN: 3,02m $\pm$ 2,18m

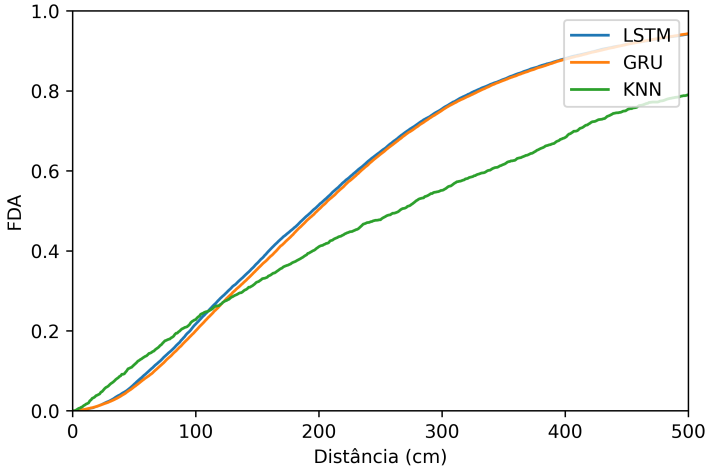

Figura 42 – Resultado comparativo da FDA para os modelos específicos e testados com 1 AP centralizado (AP 3).

Fonte: Autor.

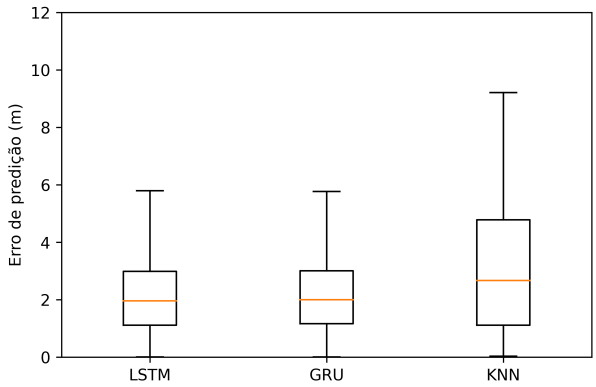

Figura 43 – Comparação estatística dos resultados para os modelos específicos, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 1931$ e $p = 0.0$, indicando que os resultados possuem desempenho significativamente diferentes.

Fonte: Autor.

Isso se deve ao comportamento dos sinais do AP 3 no ambiente de referência, com localizações em extremidades opostas do ambiente apresentando níveis semelhantes de RSS (figura 15), comprometendo por completo quaisquer possibilidade de reação do sistema de localização, fora raros instantes em que o robô está próximo do AP, como demonstrado na figura 44, o único local onde os sistemas conseguiram alguma reação de se localizar.

Tal comportamento, atrelado aos baixos sinais de RSS observados em certas áreas, prejudicou ainda mais os resultados, como pode-se observar na figura 47, mesmo treinando as RNRs com dados simulados com os mesmos parâmetros, não atingindo o máximo de 2 metros

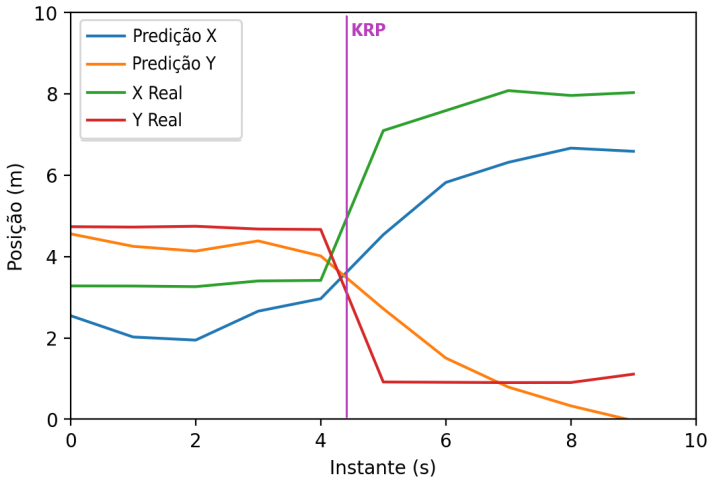

Figura 44 – Análise dos resultados para o modelo específico GRU com o robô próximo ao AP 3 centralizado.

Fonte: Autor.

desejáveis de erro na localização em grande parte do tempo, conforme especificado no capítulo da metodologia.

Os resultados médios da KNN ficaram mais próximos das soluções com RNRs devido ao comportamento de se posicionar no centro entre os dois vizinhos mais próximos, que se distribuíram radialmente em relação ao centro da área de simulação. Com isso, a KNN fixou-se em uma localização mais centralizada, diminuindo em termos probabilísticos os erros, mas sem nenhuma relação com quaisquer convergência de localização.

5.5.2.2 Utilizando modelo base treinado com os parâmetros **stream** *de dados, todos os APs disponíveis, 1dBm de ruído e 10% de atenuação*

Quando realizamos o teste para o AP 3 utilizando os modelos de RNR treinados com os parâmetros base, temos como resultado um sistema inoperante, pois o modelo base não consegue se relocalizar até a realização de outro KRP, fazendo com que o modelo apresente resultados muito ruins durante todo o teste (figura 47). Para o KNN, o resultado foi um pouco melhor, pelo fato de não se basear no histórico para se localizar e consequentemente não sofrer tanta influência pelas ocorrências do KRP.

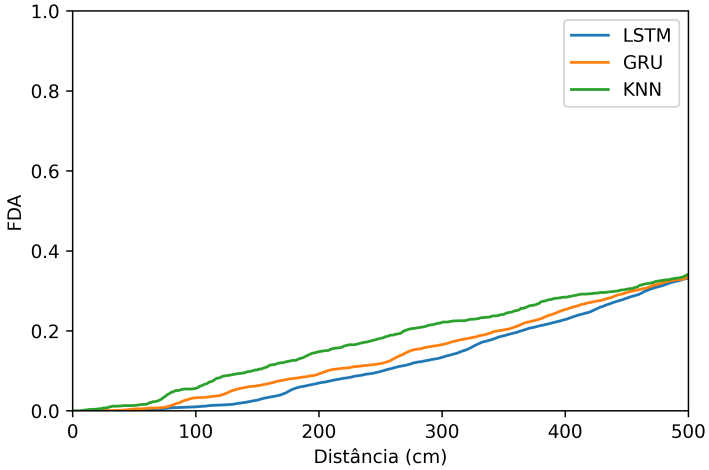

Figura 45 – Resultado comparativo da FDA para os modelos base e testados com 1 AP centralizado (AP 3).

Fonte: Autor.

Resultados obtidos de erro médio e desvio padrão durante os testes com um único AP centralizado para os modelos base, com os dados estatísticos na figura 46:

a) LSTM: 7,07m $\pm$ 2,70m

b) GRU: 6,42m $\pm$ 2,65m

c) KNN: 6,18m $\pm$ 2,82m

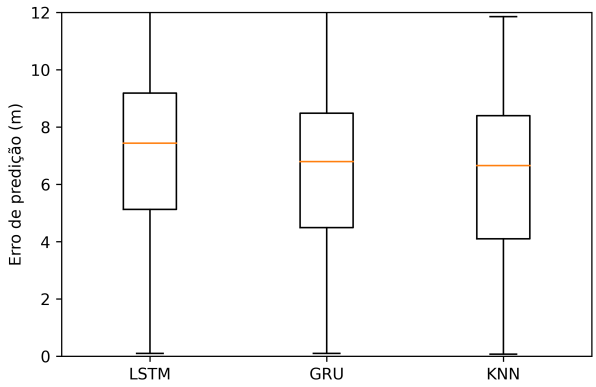

Figura 46 – Comparação estatística dos resultados para os modelos base, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 711{,}5$ e $p = 7{,}97e^{-307}$, indicando que os resultados possuem desempenho significativamente diferentes.

Fonte: Autor.

Para a solução aplicada em KNN, a total falta de vizinhos mesmo que parcialmente semelhantes faz com que a solução opere com respostas aleatórias e sem conexão com quaisquer localização real.

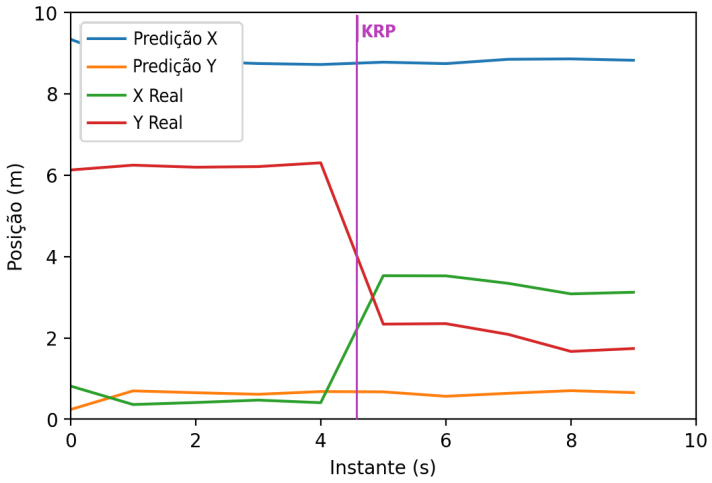

Figura 47 – Análise dos resultados para o modelo base GRU com 1 AP centralizado (AP 3), incapaz de se localizar.

Fonte: Autor.

5.5.3 Simulador operando com um único AP localizado em uma extremidade do ambiente (AP 4)

Adicionalmente ao teste realizado com um único AP centralizado, foram realizados dois testes com um único AP localizado na extremidade da área de simulação (denominado como testes tipo 4 e tipo 1, devido ao número binário correspondente ao quarto bit mais significativo apenas com bit 1 e o bit menos significativo apenas com bit 1, respectivamente). Para essas avaliações, foram escolhidos os AP 1 e 4, localizados respectivamente na sala e na cozinha, pois o AP 1 apresenta melhor possibilidade de apresentar bons resultados e o AP 4 tem expectativa do contrário, conforme avaliação do estudo de *fingerprinting* da figura 15, onde podemos observar boa distribuição dos sinais RSS para o AP 1 e para o AP 4, diversos pontos ocultos e com problemas de multipropagação, gerando sinais de baixo RSS em grande parte do ambiente de simulação.

5.5.3.1 Utilizando modelo específico treinado com os mesmos parâmetros dos dados de teste

O resultado dos testes com o modelo específico para a configuração contendo apenas o AP 4 apresenta dificuldade para se relocalizar após a realização de um KRP. Para a KNN, o resultado fica praticamente aleatório, pois os dados de treinamento se tornam tão próximos

que mesmo com a seleção de apenas 2 vizinhos, eles podem estar em extremidades opostas do ambiente de simulação.

Assim como ocorre na solução com um único AP centralizado, neste estudo de caso a realização do KRP não surtiu diferença pois nenhum sistema de localização foi capaz de chegar próximo a prever a localização aproximada, fora raros instantes onde a própria aleatoriedade do KRP levou o robô a uma posição próxima aos valores previstos ou quando o robô se localizava dentro de um raio de 1 metro do AP, onde o sinal RSS apresentou destaque nos valores, mas assim que o robô se distancia do AP 4 os sistemas voltam a se tornarem inoperantes.

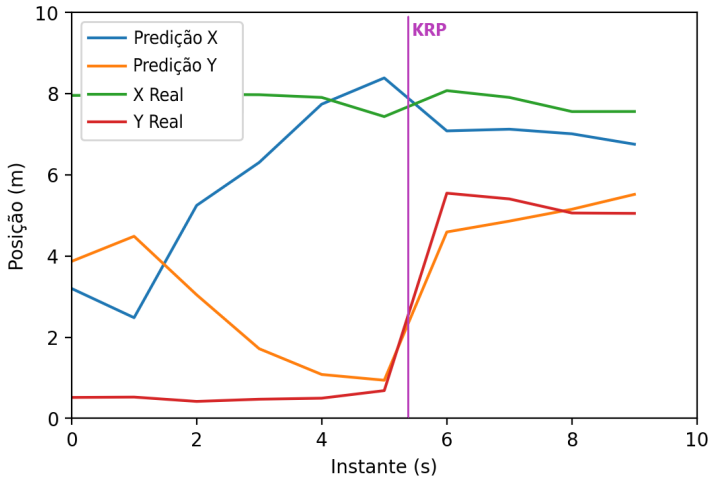

Figura 48 – Resultado do teste para o modelo específico LSTM com o robô ficando próximo do AP 4 após o KRP.

Fonte: Autor.

A figura 48 demonstra um instante onde o sistema de localização estava inoperante, mas conseguiu convergir após a ocorrência de um KRP que levou o robô a uma localização perto do AP 4.

Resultados obtidos de erro médio e desvio padrão durante os testes com um único AP na extremidade (AP 4) para os modelos específicos, com os dados estatísticos na figura 49:

a) LSTM: 2,37m $\pm$ 1,82m

b) GRU: 2,37m $\pm$ 1,76m

c) KNN: 3,16m $\pm$ 2,52m

Conforme esperado para esta simulação, observa-se um grande impacto nos resultados de predição da figura 50, mesmo treinando o modelo com os dados nos mesmos parâmetros. Isso ocorre devido a invariabilidade da base de sinais de RSS no *fingerpriting* em grande parte do ambiente de simulação, onde mesmo com o ruído médio em 1dBm, sobrepõe grande parte

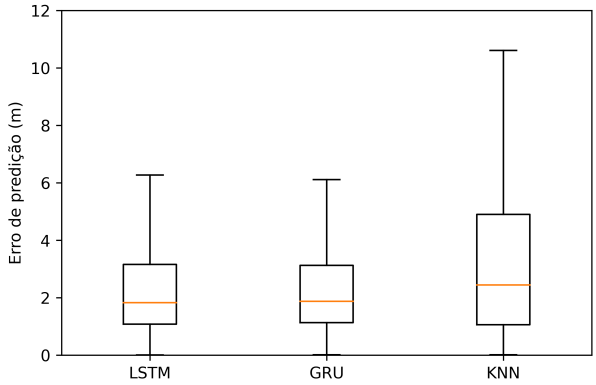

Figura 49 – Comparação estatística dos resultados para os modelos específicos, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 1281$ e $p = 0{,}0$, indicando que os resultados possuem desempenho significativamente diferentes.

Fonte: Autor.

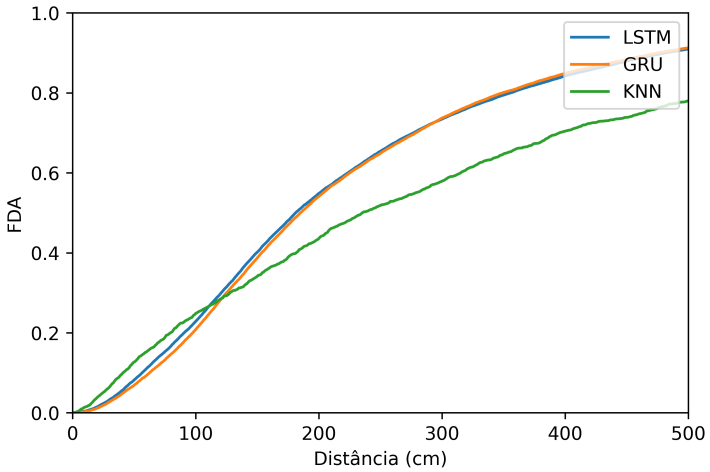

Figura 50 – Resultado comparativo da FDA para os modelos específicos e testados com um único AP na extremidade (AP 4).

Fonte: Autor.

das soluções e dificultando o treinamento adequado do sistema de localização. O resultado é uma dificuldade de se relocalizar após a realização de um KRP.

Para a KNN, o resultado fica variando aleatoriamente em dentro de um raio de distância em relação ao AP, pois os dados de treinamento se tornam tão próximos que, mesmo com a seleção de apenas 2 vizinhos, eles podem indicar respostas em extremidades opostas do ambiente de simulação.

5.5.3.2 *Utilizando modelo base treinado com os parâmetros* **stream** *de dados, todos os APs disponíveis, 1dBm de ruído e 10% de atenuação*

A invariabilidade do RSS do AP 4 fica ainda mais evidente quando utiliza-se os modelos de RNR treinados com os parâmetros base (figura 51), pois o modelo base aprende no treinamento que o AP 4 tem baixa representatividade na predição da localização, fazendo com que o modelo apresente resultados praticamente inválidos durante todo o teste, mesmo quando próximo do AP 4.

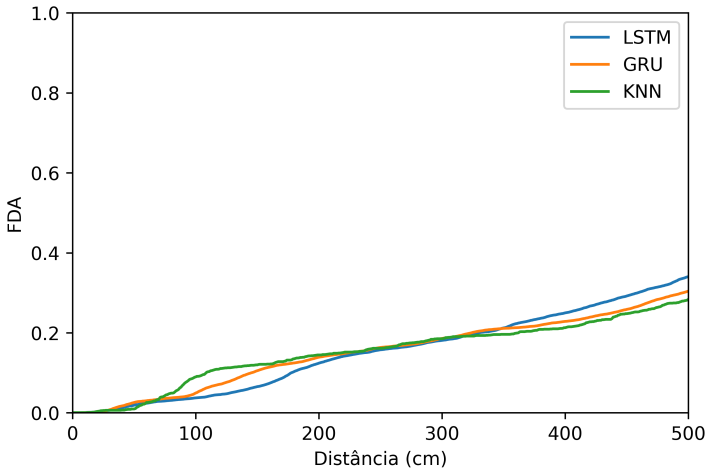

Figura 51 – Resultado comparativo da FDA para os modelos base e testados com um único AP na extremidade (AP 4).

Fonte: Autor.

Resultados obtidos de erro médio e desvio padrão durante os testes com um único AP na extremidade (AP 4) para os modelos base, com os dados estatísticos na figura 52:

a) LSTM: 7,01m $\pm$ 2,91m

b) GRU: 6,87m $\pm$ 2,85m

c) KNN: 6,60m $\pm$ 2,79m

Para a KNN, o resultado também fica praticamente aleatório, pois os dados de treinamento se tornam tão distantes devido à desativação dos outros APs, que mesmo com a seleção de apenas 2 vizinhos, eles podem estar em extremidades opostas do ambiente de simulação, comprometendo os resultados.

Neste estudo a realização do KRP não surtiu diferença pois nenhum sistema de localização foi capaz de chegar próximo a prever a localização aproximada, fora raros instantes onde a própria aleatoriedade do KRP levou o robô a uma posição próxima aos valores previstos.

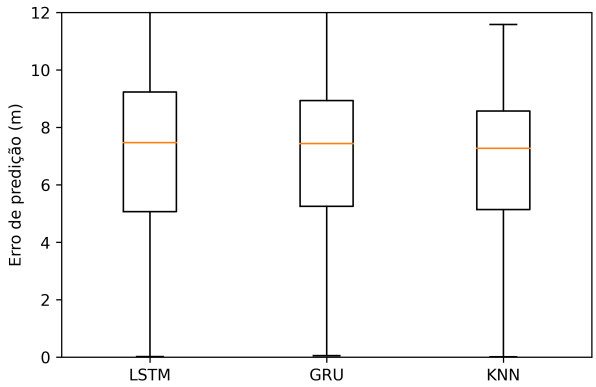

Figura 52 – Comparação estatística dos resultados para os modelos base, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 132{,}6$ e $p = 3{,}19e^{-58}$, indicando que os resultados possuem desempenho significativamente diferentes.

Fonte: Autor.

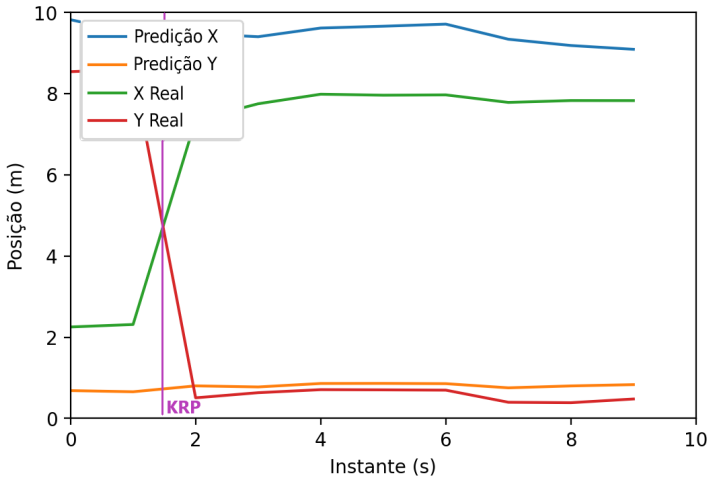

Figura 53 – Resultado do teste para o modelo base LSTM com um único AP na extremidade (AP 4), demonstrando uma condição limite onde o sistema não consegue prever a localização.

Fonte: Autor.

No gráfico da figura 53, observa-se a condição inoperante do sistema de localização LSTM com o modelo base.

O gráfico da figura 54 demonstra também uma condição em que, mesmo com predição próxima da localização correta, o modelo base LSTM não é capaz de identificar a ocorrência do KRP para os parâmetros de teste um único AP (AP 4).

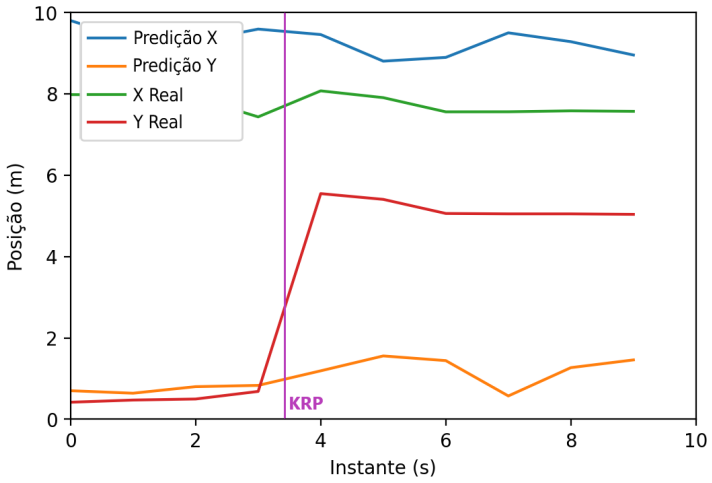

Figura 54 – Resultado do teste para o modelo base LSTM com um único AP na extremidade (AP 4), demonstrando um teste em que o sistema não reage ao KRP.

Fonte: Autor.

5.5.4 Simulador operando com um único AP localizado em uma extremidade do ambiente (AP 1)

Adicionalmente à configuração com apenas o AP 4 atuando, a configuração de AP único com o AP 1 acionado foi escolhida pois apresenta uma distribuição de sinais RSS mais homogênea pelo ambiente de simulação conforme observa-se nos dados de *fingerprinting* (figura 15), enriquecendo os resultados comparativos de qualidade de sinal.

5.5.4.1 Utilizando modelo específico treinado com os mesmos parâmetros dos dados de teste

Para os testes com o AP 1, conforme esperado durante a análise prévia do *heatmap* utilizado como base para geração dos dados de simulação, obteve-se resultados melhores do que os testes com o AP 4, apesar de sofrer dos mesmos problemas indicados no estudo anterior, como a tendência de variação da localização em termos radiais, mas com menor intensidade devido a melhor distribuição dos níveis de sinais no ambiente real.

Resultados obtidos de erro médio e desvio padrão durante os testes com um único AP na extremidade (AP 1) para os modelos específicos, com os dados estatísticos na figura 56:

a) LSTM: 2,09m $\pm$ 1,47m

b) GRU: 2,16m $\pm$ 1,48m

c) KNN: 2,94m $\pm$ 2,23m

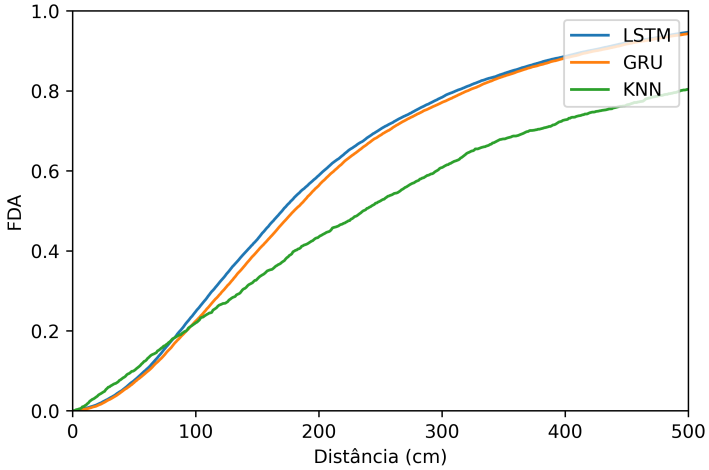

Figura 55 – Resultado comparativo da FDA para os modelos específicos e testados com um único AP na extremidade (AP 1).

Fonte: Autor.

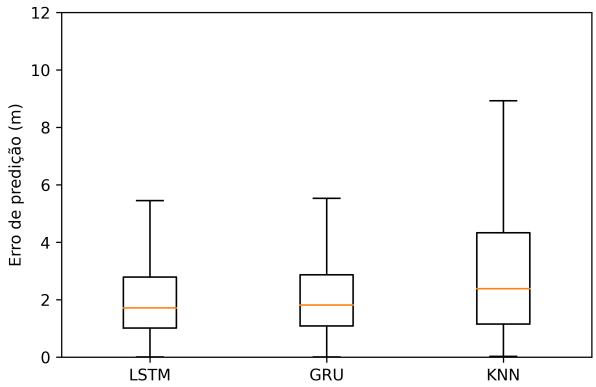

Figura 56 – Comparação estatística dos resultados para os modelos específicos, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 1806$ e $p = 0{,}0$, indicando que os resultados possuem desempenho significativamente diferentes.

Fonte: Autor.

Com a melhoria da distribuição dos sinais no ambiente de simulação, pôde-se observar um fenômeno para os sistemas baseados em RNRs que indica uma localização melhor quando o robô está se movimentando em sentido radial, se distanciando ou se aproximando do AP 1, e gradativamente piorando a precisão da localização quando o movimento se torna tangencial ao raio de sinais RSS, quando a tendência se torna uma piora na precisão.

Na figura 57 detalha-se sobre o *heatmap* do AP 1 com uma linha roxa o que seria um exemplo de movimento radial “(a)”, quando os sistemas apresentam melhor precisão, e um exemplo de movimento tangencial “(b)”, quando as RNRs indicam maior dificuldade de indicar

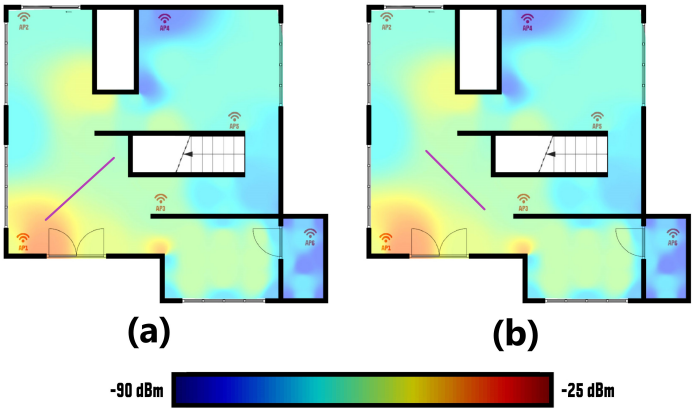

Figura 57 – Detalhe de movimentos que influenciaram a precisão dos sistemas de localização utilizando modelo específico nos testes com apenas o AP 1.

Fonte: Autor.

a localização correta. As linhas não apresentam seta de sentido pois não afeta o resultado, apenas a direção tem representatividade na precisão das pedições.

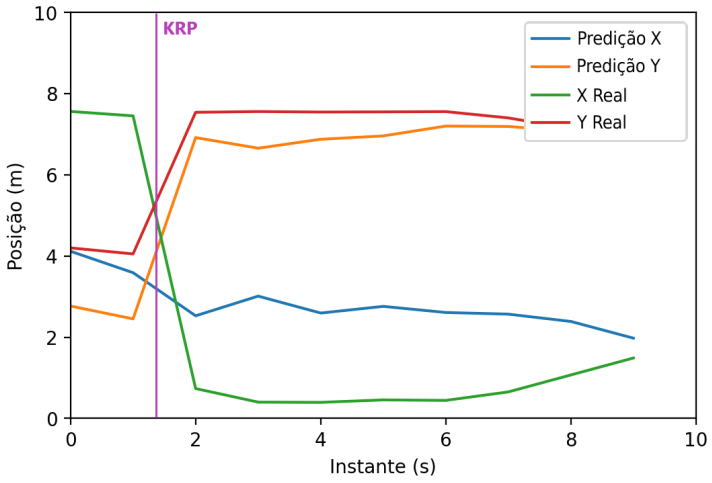

Figura 58 – Resultado do teste para o modelo base LSTM com um único AP na extremidade (AP 1), demonstrando o atraso para relocalização após o KRP.

Fonte: Autor.

Tal comportamento gera um atraso para as RNRs convergirem em localizações mais precisas quando o robô sofre o KRP e inicia um movimento tangencial, conforme dados apresentados na figura 58.

A solução aplicada em KNN foi menos beneficiada nessa análise em comparação aos testes com o AP 4 pois não trabalha com análise histórica dos dados e mantém incertezas semelhantes à movimentação tangencial para as RNRs, pior caso da análise. Os resultados para a

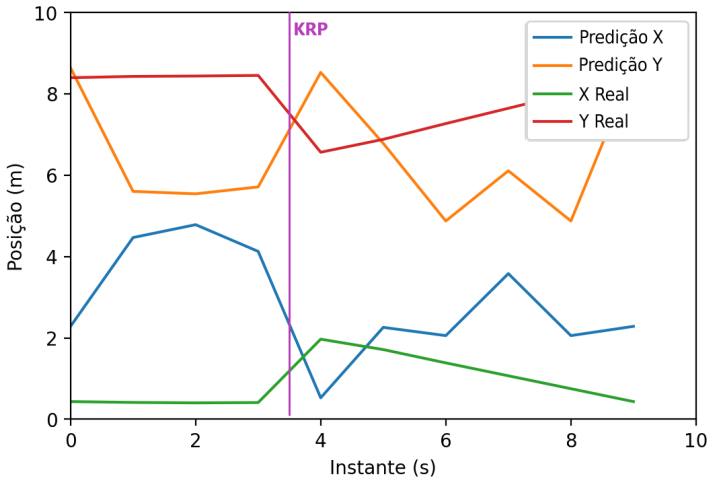

Figura 59 – Resultado do teste para o modelo específico KNN com um único AP na extremidade (AP 1), indicando flutuação na predição.

Fonte: Autor.

KNN e sua dificuldade em encontrar pares de vizinhos próximos que reflitam em um posicionamento mais preciso.

Ao comparar os resultados obtidos entre as soluções de predição, podemos observar que para as RNRs o impacto geral foi em atraso para relocalizar-se após o KRP, dependendo de variações específicas de movimento para convergir em dados mais precisos, conforme figura 58. Para o KNN, os gráficos da figura 59 indicam a grande flutuação da solução enquanto busca o valor correto de predição.

5.5.4.2 Utilizando modelo base treinado com os parâmetros **stream** *de dados, todos os APs disponíveis, 1dBm de ruído e 10% de atenuação*

Quando utiliza-se o modelo treinado com os parâmetros base, a comparação dos resultados indica que apesar da média de erro manter-se em níveis semelhantes entre os APs 4 e 1, o desvio padrão diminuiu para o AP 1, indicando o benefício da melhor distribuição dos sinais pelo ambiente, e comprovado na análise de função de distribuição acumulada (FDA) da figura 60.

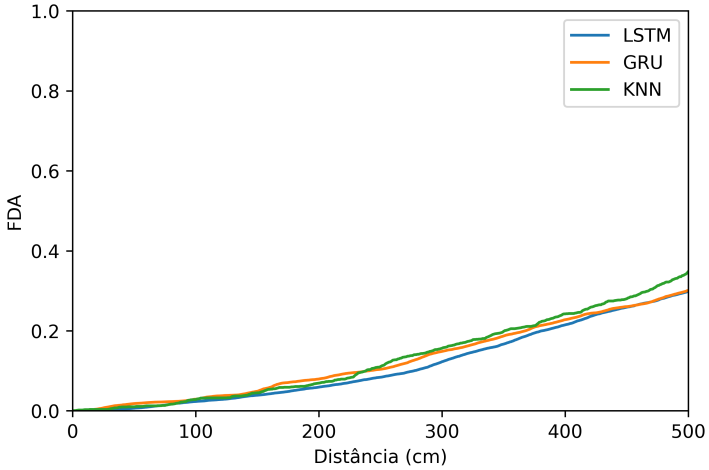

Figura 60 – Resultado comparativo da FDA para os modelos base e testados com um único AP na extremidade (AP 1).

Fonte: Autor.

Resultados obtidos de erro médio e desvio padrão durante os testes com um único AP na extremidade (AP 1) para os modelos base, com os dados estatísticos na figura 61:

a) LSTM: 7,17m $\pm$ 2,69m

b) GRU: 6,78m $\pm$ 2,69m

c) KNN: 6,42m $\pm$ 2,58m

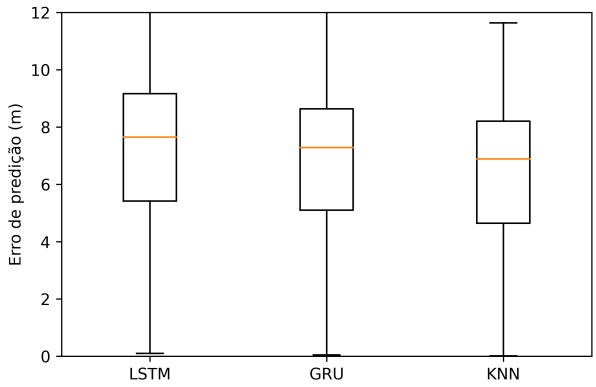

Figura 61 – Comparação estatística dos resultados para os modelos base, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 500{,}9$ e $p = 8{,}07e^{-217}$, indicando que os resultados possuem desempenho significativamente diferentes.

Fonte: Autor.

Mesmo apresentando melhores resultados de desvio padrão quando comparado aos estudos do modelo base com o AP 4, esta configuração ainda mantém níveis de convergência baixíssimos, com todas as propostas de sistema de localização inoperantes.

5.6 Resultados de performance com variação nos níveis de ruído simulado

Outro estudo que busca avaliar a robustez das soluções de localização apresentadas neste estudo consiste em variar o nível de ruído aplicado no simulador. Em adição aos testes realizados com 1dBm de ruído independente por AP, esta seção testará os resultados de testes com os níveis de ruído médio gaussiano em 2dBm e 10dBm respetivamente, mantendo as análises com modelos específicos para cada configuração em questão e utilizando o modelo base determinado nas seções anteriores.

5.6.1 Simulador operando com 2dBm médio de ruído gaussiano

A primeira análise com variação de ruído consiste em dobrar a média aplicada no modelo base proposto por Abbas et al. (2019), com ruído médio gaussiano em 2dBm.

5.6.1.1 Utilizando modelo específico treinado com os mesmos parâmetros dos dados de teste

Utilizando o modelo treinado especificamente para a condição de ruído em 2dBm, os sistemas de localização apresentaram convergência média de localização abaixo de 1 metro com desvio padrão abaixo de 1 metro, indicando bom funcionamento dos sistemas de localização, tanto os sistemas baseados em RNRs quanto o sistema KNN.

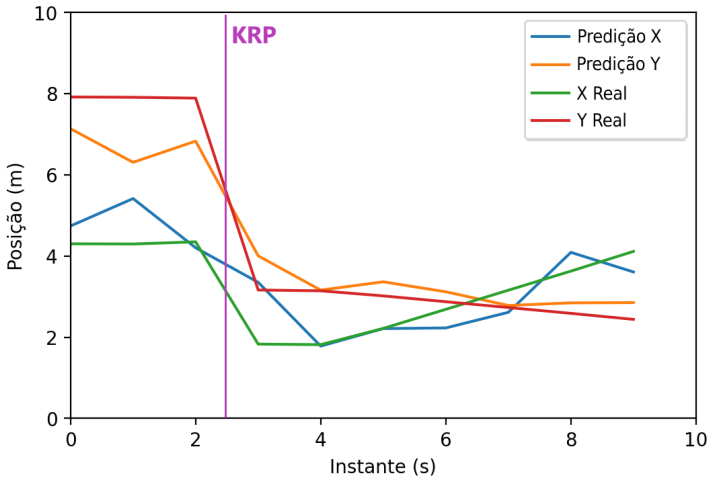

Figura 62 – Resultado do teste para o modelo específico LSTM e testado com 2 dBm de ruído, com pequena instabilidade na predição.

Fonte: Autor.

Resultados obtidos de erro médio e desvio padrão durante os testes com 2 dBm de ruído para os modelos específicos, com os dados estatísticos na figura 63:

a) LSTM: 0,80m $\pm$ 0,62m

b) GRU: 0,89m $\pm$ 0,69m

c) KNN: 0,90m $\pm$ 0,97m

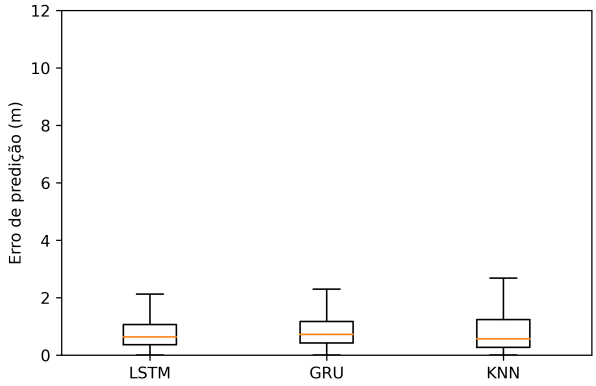

Figura 63 – Comparação estatística dos resultados para os modelos específicos, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 132{,}7$ e $p = 3{,}07e^{-58}$, indicando que os resultados possuem desempenho significativamente diferentes.

Fonte: Autor.

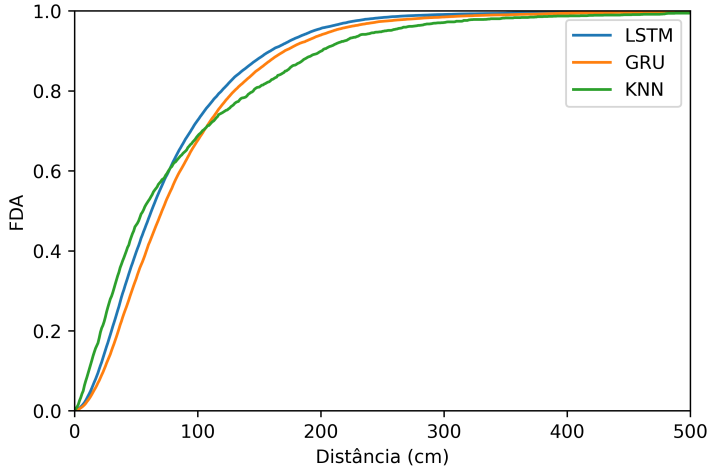

Figura 64 – Resultado comparativo da FDA para os modelos específicos e testados com 2 dBm de ruído.

Fonte: Autor.

Observando com detalhe os resultados obtidos quando aumentamos o ruído para 2dBm, conforme figura 65, as RNRs apresentam mais instabilidade na predição durante períodos estáveis do robô, sem ocorrência do KRP. Tal comportamento afeta o treinamento das RNRs por

apresentar maior amplitude de RSS em locais muito próximos, mas em contrapartida favorece o sistema baseado em KNN, pois aumenta a variabilidade dos dados de treinamento apresentados e consequentemente apesenta vizinhos mais próximos dos valores de predição, diminuindo um pouco a variabilidade da predição.

5.6.1.2 Utilizando modelo base treinado com os parâmetros **stream** *de dados, todos os APs disponíveis, 1dBm de ruído e 10% de atenuação*

Quando o teste com ruído médio gaussiano em 2dBm é realizado com o modelo de treinamento base, a situação se inverte, conforme observa-se na figura 67.

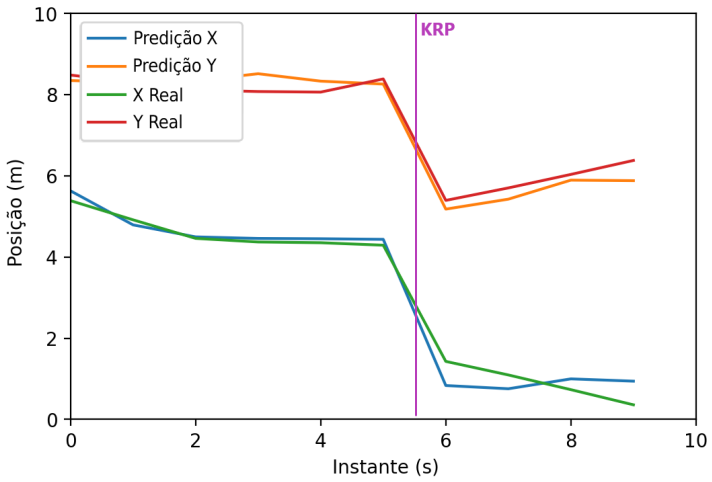

Figura 65 – Resultado do teste para o modelo base LSTM e testado com 2 dBm de ruído, demonstrando melhores resultados do que o modelo específico.

Fonte: Autor.

No gráfico da figura 65 já se observa um princípio de instabilidade na predição das RNRs gerada pelo aumento do ruído.

A estabilidade dos dados de treinamento e o histórico de dados de entrada das RNRs favorecem a predição quando há uma certa variabilidade nas informações de teste, mas para o KNN, temos menor representabilidade dos dados e afastamento dos vizinhos quando o ruído chega a valores acima do identificado nos dados de treinamento, aumentando as distâncias até os vizinhos mais próximos e comprometendo as predições. Por este motivo, na figura 67 fica bem claro a inversão das curvas de FDA entre RNRs e KNN.

Resultados obtidos de erro médio e desvio padrão durante os testes com 2 dBm de ruído para os modelos base, com os dados estatísticos na figura 66:

a) LSTM: 0,42m $\pm$ 0,36m

b) GRU: 0,53m $\pm$ 0,40m

c) KNN: 1,38m $\pm$ 1,52m

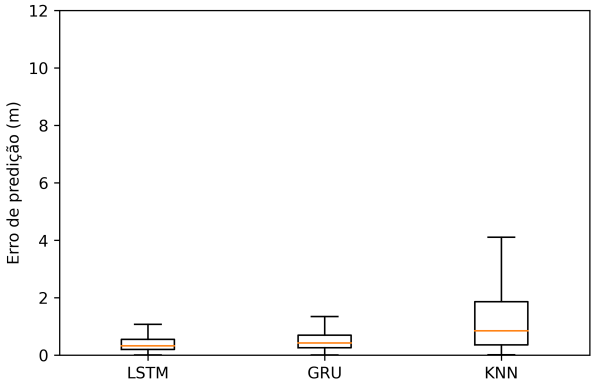

Figura 66 – Comparação estatística dos resultados para os modelos base, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 7899$ e $p = 0{,}0$, indicando que os resultados possuem desempenho significativamente diferentes.

Fonte: Autor.

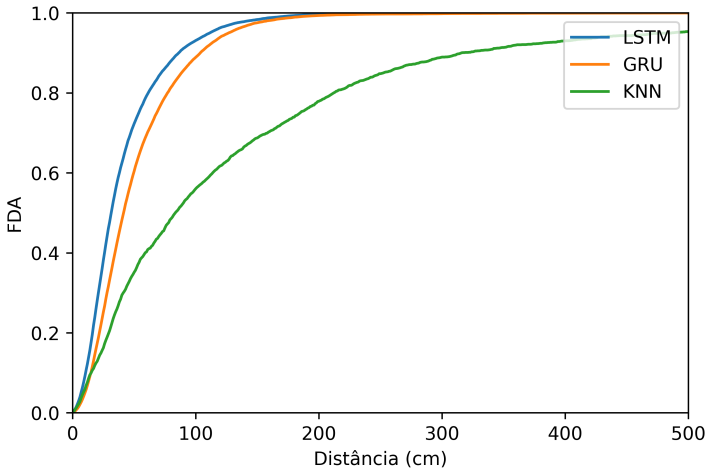

Figura 67 – Resultado comparativo da FDA para os modelos base e testados com 2 dBm de ruído.

Fonte: Autor.

Quando comparamos as soluções de modelos treinados em KNN, observa-se um grande impacto na localização quando utilizado o modelo base, devido a baixa variabilidade de vizinhos para convergir em exemplos mais próximos dos dados de teste, apresentando incapacidade de manter a convergência em momentos de estabilidade, sem a ocorrência do KRP.

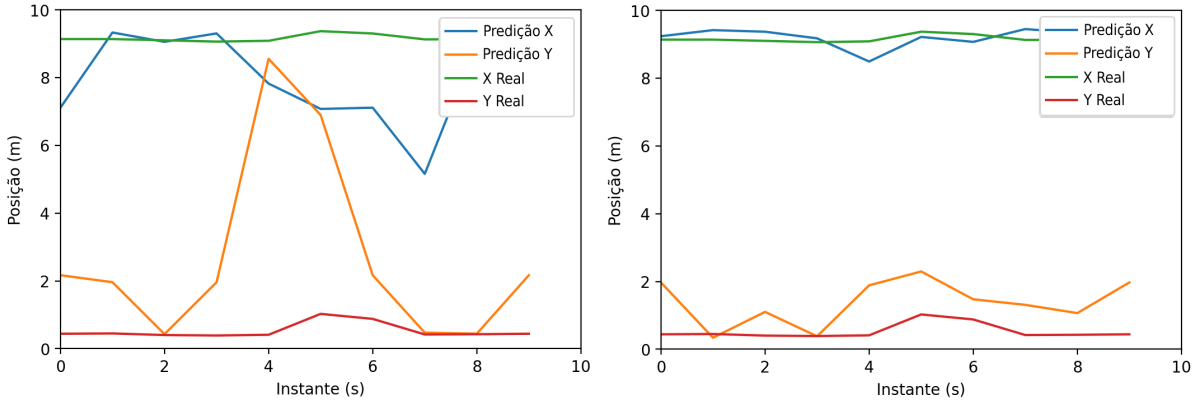

Figura 68 – Exemplo comparativo de resultado do teste com 2 dBm para ambos os modelos KNN. A esquerda está o teste realizado com o modelo base. A direita, o teste realizado com o modelo específico.

Fonte: Autor.

Para a base de treinamento com 2dBm de ruído, o KNN recupera sua capacidade de convergir durante a estabilidade, por manter exemplos de vizinhos na base de treinamento mais próximos dos dados apresentados para teste.

A figura 68 demonstra lado a lado a diferença dos dois modelos para o mesmo teste em momentos de estabilidade, em que não ocorre KRP, onde fica claro o impacto da falta de vizinhos representativos que possam convergir os dados em certos instantes que o ruído dos dados de teste superam os dados da base de treinamento.

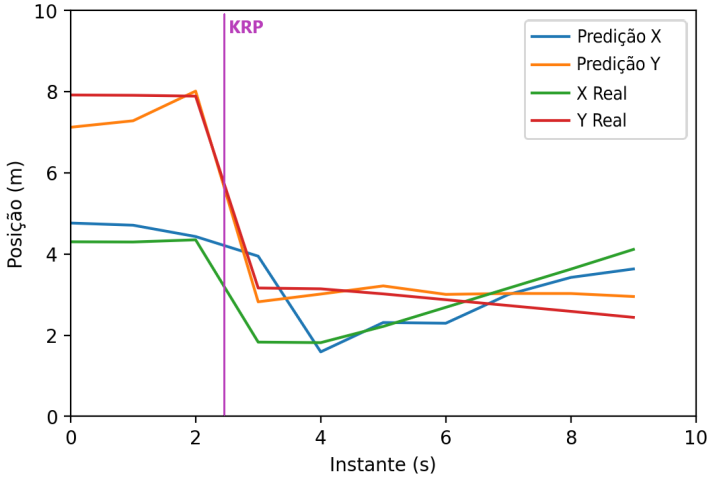

Figura 69 – Resultado do teste para o modelo base GRU e testado com 2 dBm de ruído, demonstrando o atraso para relocalização após o KRP.

Fonte: Autor.

As RNRs não sofrem desse problema por considerarem 10 instantes de histórico de localização, mas o ruído maior aplicado nesta avaliação dificulta em partes a identificação do KRP, gerando um pequeno atraso de 1 segundo em média no tempo de resposta e gerando uma pequena oscilação em momentos de estabilidade, afetando os resultados. Mesmo assim, o modelo treinado com os dados base apresentou melhor precisão na localização por manter menor elasticidade nos sinais RSS dos dados de treinamento e por consequência definindo com maior percentual de acerto os valores de predição.

5.6.2 Simulador operando com 10dBm médio de ruído gaussiano

A condição de ruído de 10dBm foi incluída neste estudo para avaliar o comportamento dos sistemas em casos extremos, onde o ruído poderia superar a variação total de certos APs no *fingerprinting* do ambiente de referência.

5.6.2.1 Utilizando modelo específico treinado com os mesmos parâmetros dos dados de teste

Nos testes com 10dBm de ruído, chegamos em situações mais críticas, onde as RNRs se comportam quase como a KNN, com a predição flutuando sobre os valores corretos e gerando um erro médio alto, mas mesmo em condições adversas, quando o modelos RNRs foram treinados na mesma condição, as soluções apresentaram condições de manter as médias de localização abaixo dos 2 metros de erro.

Na figura 70 observa-se o comportamento comum durante o teste para as soluções RNR sobre a condição com 10 dBm em modelos treinados com os mesmos parâmetros de teste. Como as RNRs foram treinadas com dados que apresentaram grande variabilidade de RSS para cada localização, mesmo as RNRs trabalhando com histórico de 10 segundos, elas apresentaram instabilidade nas respostas de saída, flutuando em torno da localização correta.

Resultados obtidos de erro médio e desvio padrão durante os testes com 10 dBm de ruído para os modelos específicos, com os dados estatísticos na figura 71:

a) LSTM: 1,70m $\pm$ 1,13m

b) GRU: 1,72m $\pm$ 1,11m

c) KNN: 2,45m $\pm$ 1,68m

Observa-se também que a variabilidade pode afetar o tempo de resposta quando ocorre o KRP. Isso acontece devido a grande variabilidade nos dados para a mesma localização que tal

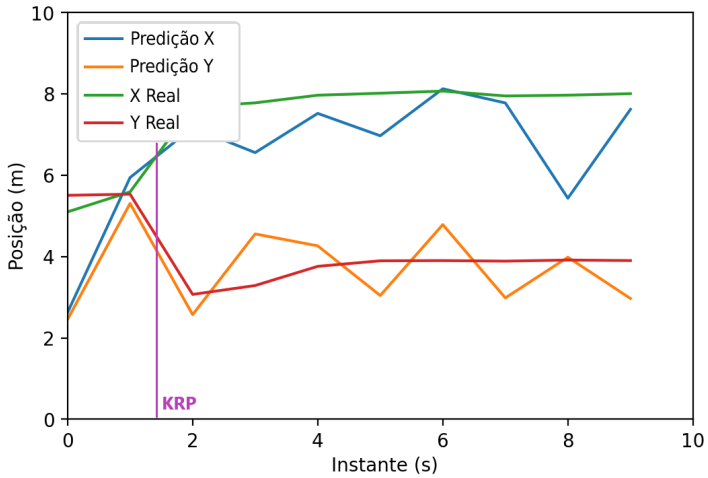

Figura 70 – Resultado do teste para o modelo específico LSTM e testado com 10 dBm de ruído, apresentando comportamento semelhante ao KNN, oscilando em torno da localização correta.

Fonte: Autor.

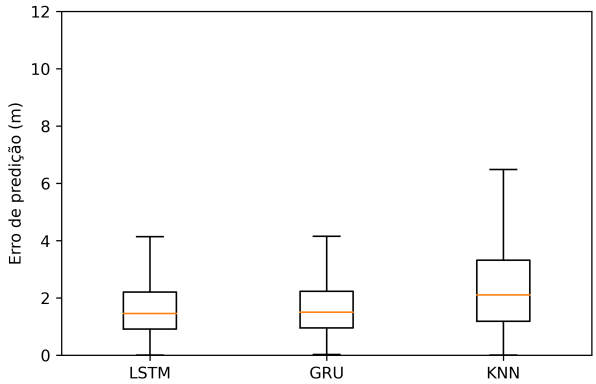

Figura 71 – Comparação estatística dos resultados para os modelos específicos, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 2604$ e $p = 0{,}0$, indicando que os resultados possuem desempenho significativamente diferentes.

Fonte: Autor.

simulação gera, fazendo com que a predição tenha uma probabilidade de acerto mais dispersa e incerta, comprometendo o acionamento dos *gates* de esquecimento. Mesmo chegando próximo, o KNN também começa a sofrer influência negativa da variabilidade, devido a sobreposição de vizinhos com predições muito distantes.

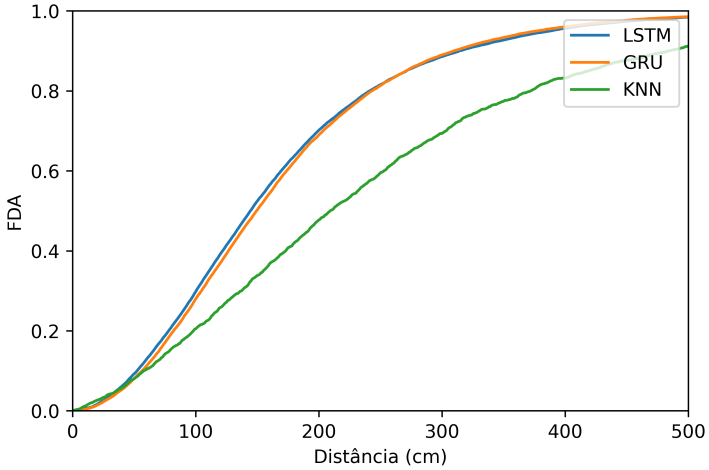

Figura 72 – Resultado comparativo da FDA para os modelos específicos e testados com 10 dBm de ruído.

Fonte: Autor.

5.6.2.2 Utilizando modelo base treinado com os parâmetros **stream** *de dados, todos os APs disponíveis, 1dBm de ruído e 10% de atenuação*

A condição de teste com 10dBm de ruído para os modelos treinados com os parâmetros base foi a única condição em que o KNN apresentou precisão maior que as RNRs, conforme observa-se no gráfico de FDA da figura 74. O caso não significa que o KNN teve bom desempenho, pois tal comportamento deve-se ao fato de que os dados de teste se distanciaram demais dos dados de treinamento com o alto ruído aplicado, impossibilitando as RNRs de realizar quaisquer correlação histórica, acionando constantemente os *gates* de esquecimento.

No caso do KNN, com a variabilidade dos dados, estatisticamente o KNN obteve erro menor nas predições, favorecido pela média dos vizinhos mais próximos (figura 73), mas com grande desvio padrão e erro médio considerável. Mesmo assim, o KNN ainda poderia gerar dados aproximados, alimentando uma base para outro sistema de refino de localização, como realizado no trabalho de Hoang et al. (2019).

Resultados obtidos de erro médio e desvio padrão durante os testes com 10 dBm de ruído para os modelos base, com os dados estatísticos na figura 75:

a) LSTM: 4,29m $\pm$ 1,90m

b) GRU: 3,58m $\pm$ 1,58m

c) KNN: 3,39m $\pm$ 1,82m

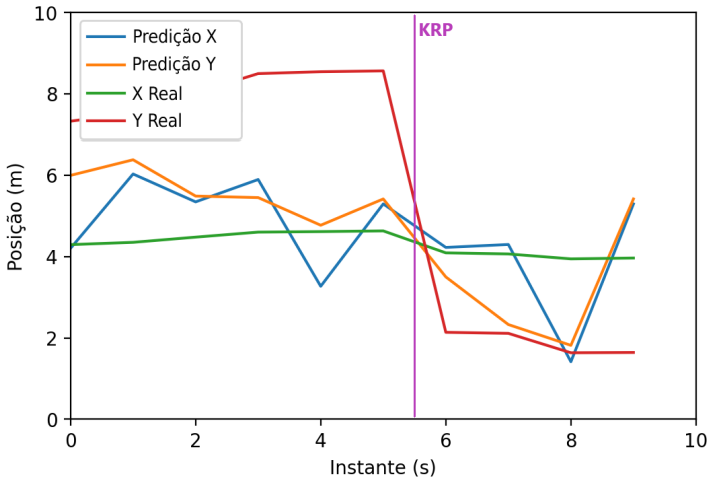

Figura 73 – Resultado do teste para o modelo base KNN e testado com 10 dBm de ruído, apresentando alto desvio na predição.

Fonte: Autor.

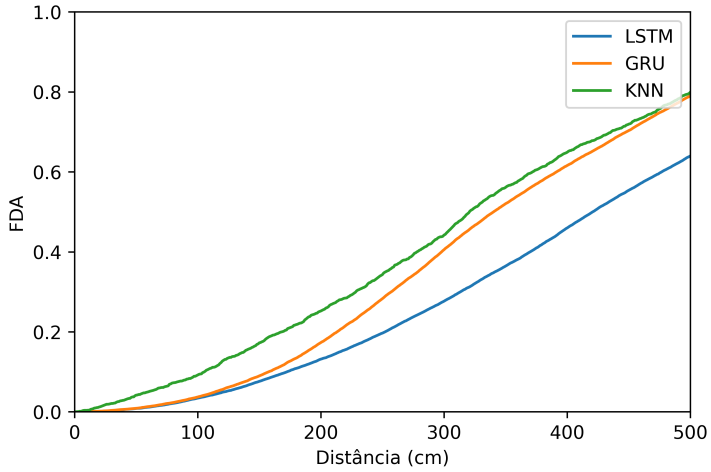

Figura 74 – Resultado comparativo da FDA para os modelos base e testados com 10 dBm de ruído.

Fonte: Autor.

Outra condição que impossibilita as RNRs de se relocalizarem corretamente depois da ocorrência de um KRP, é a questão dos valores de RSS apresentados saírem muito dos padrões apresentados na fase de treinamento, sem apresentar um período de estabilidade entre acionamento dos *gates* de esquecimento por ocorrências de KRP, comprometendo também a identificação do acontecimento.

Na figura 76, o teste demonstra a dificuldade do modelo base do sistema LSTM treinado com os parâmetros originais deste estudo em conseguir convergir para uma predição mais pre-

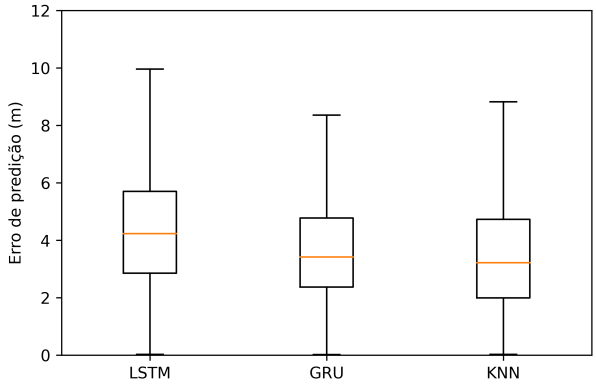

Figura 75 – Comparação estatística dos resultados para os modelos base, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 1799$ e $p = 0{,}0$, indicando que os resultados possuem desempenho significativamente diferentes.

Fonte: Autor.

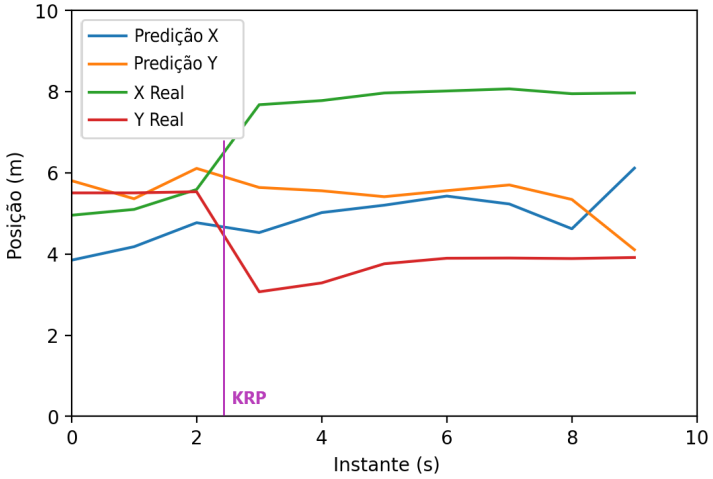

Figura 76 – Resultado do teste para o modelo base LSTM e testado com 10 dBm de ruído, demonstrando a dificuldade do sistema em convergir.

Fonte: Autor.

cisa, além da incapacidade de identificar o instante em que ocorreu o KRP, tornando a solução impraticável.

O mesmo ocorre para a RNR construída com base na GRU, com resultados um pouco melhores, por apresentarem melhor treinamento de acionamento dos *gates* e acionando-os com menor frequência. A figura 77 demonstra a flutuação menor para o sistema, mas ainda com tempo de resposta alto para a ocorrência do KRP.

Os resultados com ruído comprovam o grande impacto que a sobreposição de níveis de sinais de RSS para diferentes localidades aplica sobre os sistemas de localização durante os períodos de estabilidade. Os resultados com altos níveis de ruído demonstraram também

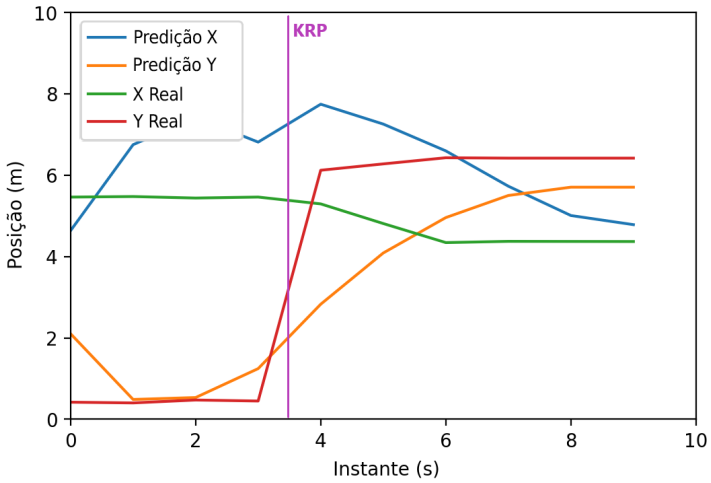

Figura 77 – Resultado do teste para o modelo base GRU e testado com 10 dBm de ruído, demonstrando a dificuldade do sistema em convergir.

Fonte: Autor.

que a alta variação afetou o acionamento dos *gates* de esquecimento das RNRs, eliminando os benefícios do uso de histórico de dados para localização nos modelos treinados com os parâmetros base.

5.7 Resultados de performance com variação nos níveis de atenuação simulado

O último estudo deste trabalho consiste em avaliar os impactos do nível de atenuação para cada proposta estudada. A atenuação é aplicada nas configurações do simulador durante o processo de geração de dados para simular o desvanecimento dos APs, geralmente ocasionados por multipropagações e níveis críticos de interferência. As propostas serão avaliadas nos níveis de 20% e 30%, adicionalmente aos 10% dos parâmetros base.

5.7.1 Simulador operando com 20% de atenuação independente

Os dados com 20% de atenuação apresentam o dobro da probabilidade independente de cada AP ser anulado na apresentação dos sinais de RSS pelo simulador na geração dos dados de treinamento, validação e teste.

5.7.1.1 Utilizando modelo específico treinado com os mesmos parâmetros dos dados de teste

Utilizando um modelo treinado com parâmetros de 20% de atenuação no simulador, tem-se resultados próximos quando comparadas as 3 propostas de sistemas deste estudo, mas a análise dos testes para cada sistema apresentou pequenas diferenças ao aplicar o KRP.

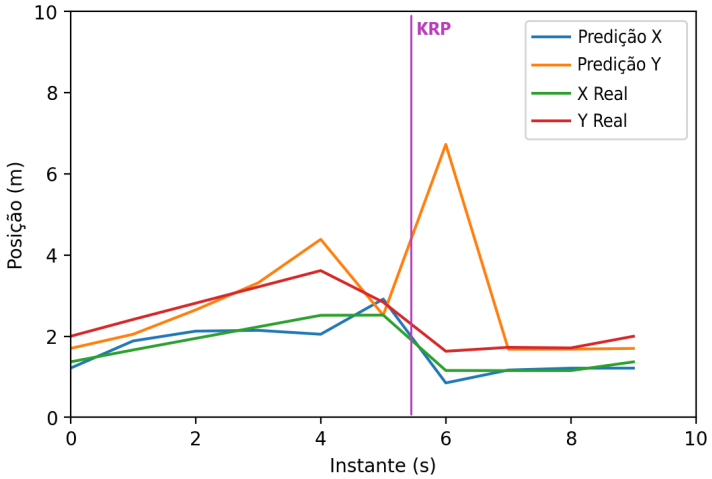

Figura 78 – Resultado do teste para o modelo específico KNN e testado com 20% de atenuação, com deslocamentos pontuais na predição.

Fonte: Autor.

A solução baseada em KNN apresentou menor flutuação durante períodos de estabilidade, com grande parte dos instantes nos testes avaliados conseguindo encontrar vizinhos próximos o suficiente para convergir em valores bem próximos da localização real, mas ocasionalmente a atenuação apresentava uma condição mais distante dos dados de treinamento, gerando um grande deslocamento entre localização real e predição devido a falta de variabilidade nos dados de teste, comprometendo os resultados. Tal comportamento pode ser minimizado para esta condição ampliando os dados de treinamento da KNN.

Na figura 78 observa-se dois eventos de deslocamento por falta de vizinhos mais próximos, um de baixa intensidade no instante 4 segundos e um crítico no instante 6 segundos, logo após o KRP.

Quando avaliados os resultados de teste com as RNRs em tal condição, os resultados apresentam uma tendência de um fenômeno semelhante ao ocorrido na desativação de parte do APs, com dificuldade em convergir após a ocorrência de um KRP para a mesma precisão de períodos em estabilidade, conforme observa-se na figura 79. O comportamento se dá devido

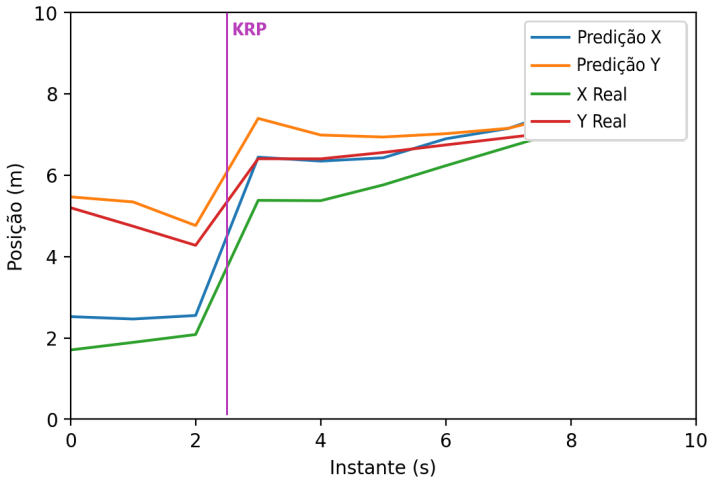

Figura 79 – Resultado do teste para o modelo específico LSTM e testado com 20% de atenuação, com dificuldade de convergir após um KRP.

Fonte: Autor.

aos dados de treinamento com maior desvanecimento oferecerem menor variabilidade de dados RSS para modelagem.

Resultados obtidos de erro médio e desvio padrão durante os testes com 20% de atenuação para os modelos específicos, com os dados estatísticos na figura 80:

a) LSTM: 0,78m $\pm$ 0,67m

b) GRU: 0,80m $\pm$ 0,65m

c) KNN: 0,83m $\pm$ 0,89m

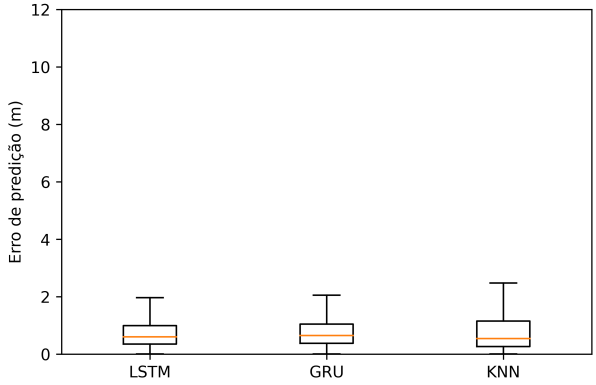

Figura 80 – Comparação estatística dos resultados para os modelos específicos, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 61{,}82$ e $p = 1{,}49e^{-27}$, indicando que os resultados possuem desempenho significativamente diferentes.

Fonte: Autor.

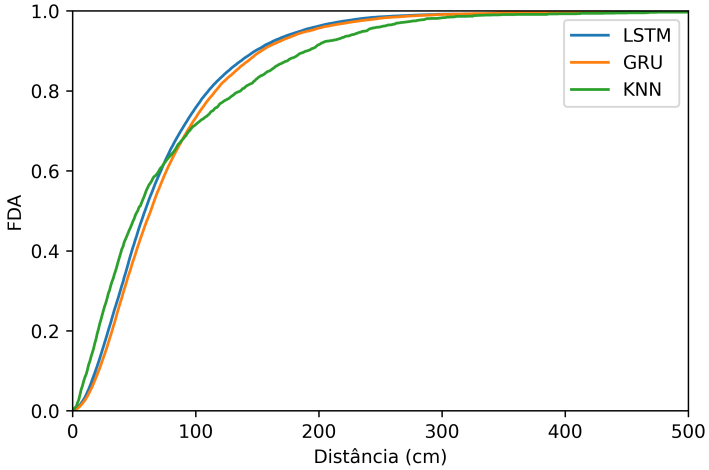

Figura 81 – Resultado comparativo da FDA para os modelos específicos e testados com 20% de atenuação.

Fonte: Autor.

Outra observação é que esta representa umas das condições em que o sistema KNN apresentou médias bem próximas das RNRs e com bons resultados de predição geral, resultado semelhante ao obtido com 2dBm de ruído.

5.7.1.2 Utilizando modelo base treinado com os parâmetros **stream** *de dados, todos os APs disponíveis, 1dBm de ruído e 10% de atenuação*

Durante os testes utilizando o modelo treinado com os parâmetros base deste estudo, as RNRs apresentaram melhores resultados que os obtidos com o modelo treinado com os mesmos parâmetros de teste. Isso ocorre pois os dados de treinamento com os parâmetros base apresentam maior riqueza de dados RSS, por consequência refinando melhor os pesos nas camadas ocultas das RNRs durante o processo. O resultado apresenta melhor identificação dos instantes onde ocorre o KRP por exemplo, e realizando comparações com os resultados obtidos com menor atenuação, observa-se pouco impacto na média de erro e no desvio padrão, como observa-se no comportamento da GRU da figura 82.

Resultados obtidos de erro médio e desvio padrão durante os testes com 20% de atenuação para os modelos base, com os dados estatísticos na figura 83:

a) LSTM: 0,51m $\pm$ 0,46m

b) GRU: 0,56m $\pm$ 0,46m

c) KNN: 1,01m $\pm$ 1,18m

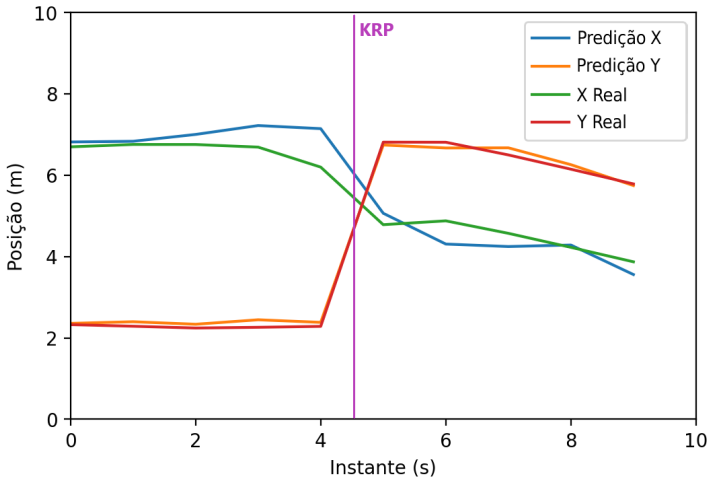

Figura 82 – Resultado do teste para o modelo base GRU e testado com 20% de atenuação, apresentando ainda boa precisão e tempo de resposta, sendo pouco prejudicado pela atenuação.

Fonte: Autor.

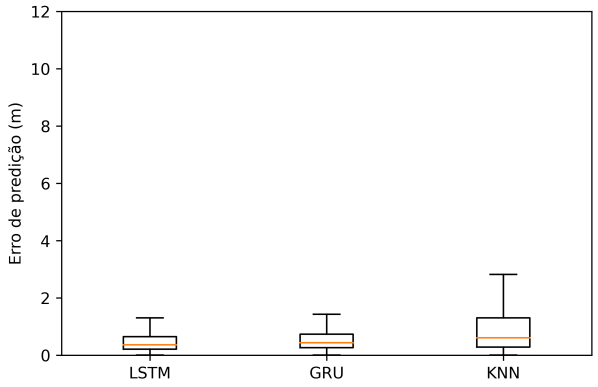

Figura 83 – Comparação estatística dos resultados para os modelos base, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 3165$ e $p = 0{,}0$, indicando que os resultados possuem desempenho significativamente diferentes.

Fonte: Autor.

Quando observados os resultados dos testes realizados com o sistema KNN, o impacto por usar o modelo base foi o oposto do observado nas RNRs, por ampliar as distâncias entre os dados de treinamento e os dados apresentados para predição, com menor quantidade de APs detectados em cada instante de teste e aumentando a incidência dos deslocamentos observados na figura 78, consequentemente piorando os resultados finais, mas ainda apresentando erros aceitáveis de predição.

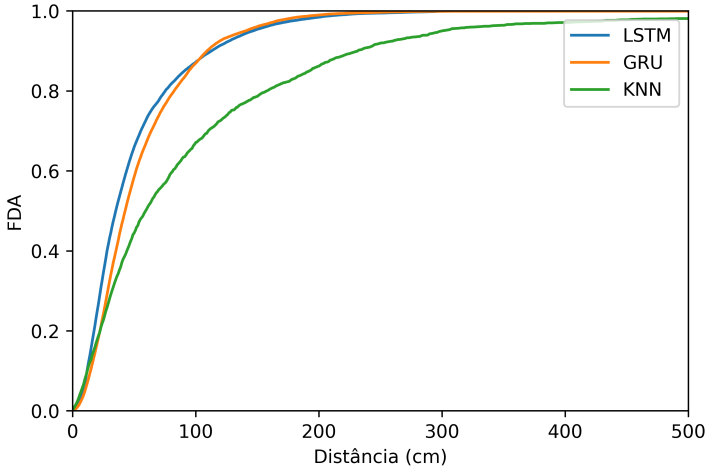

Figura 84 – Resultado comparativo da FDA para os modelos base e testados com 20% de atenuação.

Fonte: Autor.

5.7.2 Simulador operando com 30% de atenuação independente

A condição de 30% de atenuação foi considerada neste estudo para avaliar-se a progressão e adaptabilidade das propostas com esvanecimento mais frequente dos APs.

5.7.2.1 Utilizando modelo específico treinado com os mesmos parâmetros dos dados de teste

Observando-se os resultados dos testes obtidos com a máscara de atenuação em 30% independente para cada AP para as RNRs, a menor incidência de aparecimento dos dados de RSS nos dados de teste teve leve impacto nos resultados para modelos treinados com os mesmos parâmetros de teste, devido a uma menor robustez durante o treinamento da detecção do KRP, e afetando ainda mais os tempos de resposta para relocalizar-se, conforme figura 85.

Observando-se os resultados dos testes para o KNN, temos uma flutuação maior nos dados, principalmente depois da ocorrência de um KRP, e comprometendo o desvio padrão para a proposta, mas a resposta instantânea ao sequestro ajudou no erro médio geral. A figura 86 demonstra a oscilação da predição obtida nos testes.

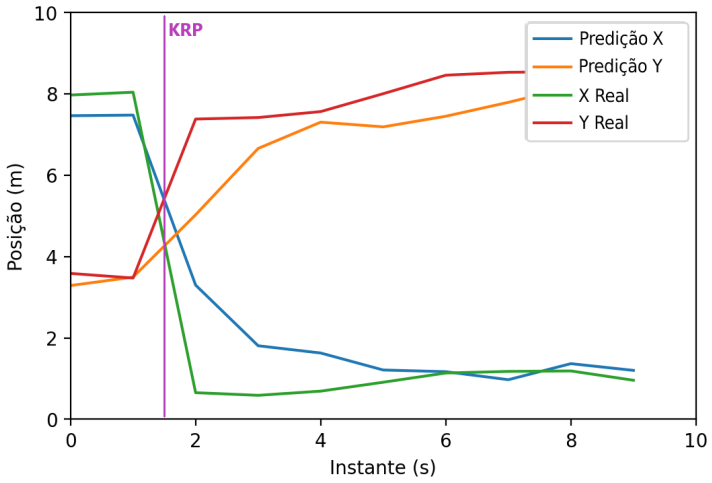

Figura 85 – Resultado do teste para o modelo específico LSTM e testado com 30% de atenuação, apresentando atraso na detecção do KRP.

Fonte: Autor.

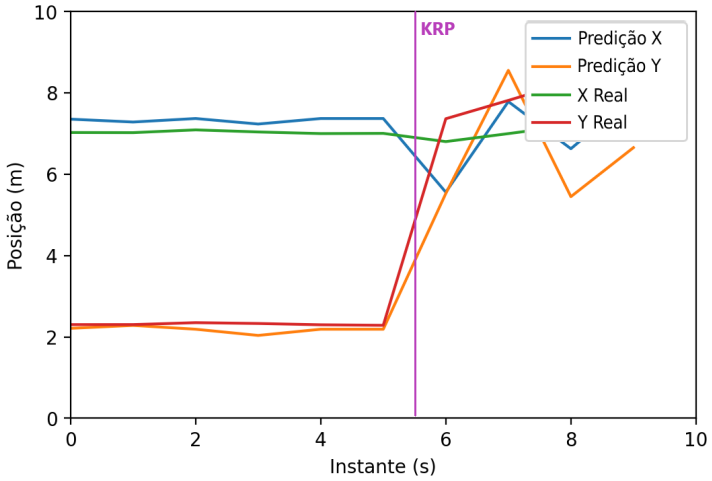

Figura 86 – Resultado do teste para o modelo específico KNN e testado com 30% de atenuação, com flutuação na predição, mas resposta rápida ao KRP.

Fonte: Autor.

Resultados obtidos de erro médio e desvio padrão durante os testes com 30% de atenuação para os modelos específicos, com os dados estatísticos na figura 87:

a) LSTM: 0,81m $\pm$ 0,68m

b) GRU: 0,87m $\pm$ 0,68m

c) KNN: 0,90m $\pm$ 0,97m

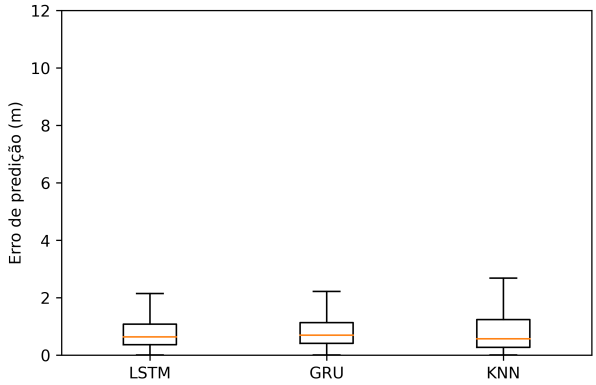

Figura 87 – Comparação estatística dos resultados para os modelos específicos, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 74{,}39$ e $p = 5{,}29e^{-33}$, indicando que os resultados possuem desempenho significativamente diferentes.

Fonte: Autor.

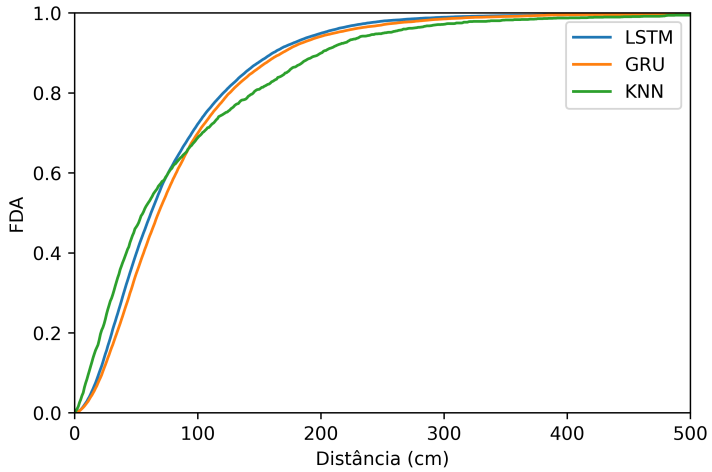

Figura 88 – Resultado comparativo da FDA para os modelos específicos e testados com 30% de atenuação.

Fonte: Autor.

Em geral, os resultados com os modelos treinados a partir dos dados com 30% de atenuação não obtiveram resultados ruins, mas ao contrário de obter maior precisão na predição testando com dados nas mesmas condições de simulação, os resultados apresentaram leve piora quando comparado aos testes com o modelo treinado com os parâmetros de base da próxima seção.

5.7.2.2 *Utilizando modelo base treinado com os parâmetros* **stream** *de dados, todos os APs disponíveis, 1dBm de ruído e 10% de atenuação*

Quando os testes para a condição de simulação com 30% de atenuação foram realizados com os modelos treinados a partir dos parâmetros base deste estudo, observa-se para as RNRs que não houve nenhum impacto nos resultados de predição. Isso ocorre devido aos *gates* de memória trabalharem também nos instantes de atenuação, mantendo os dados obtidos no instante anterior ao testado quando o AP some das medições simuladas do robô. Como a atenuação não gera deslocamento dos dados de RSS em função da predição, apenas desativa o AP, a detecção da ocorrência do KRP não é afetada também. Os resultados estão na figura 89 desta seção.

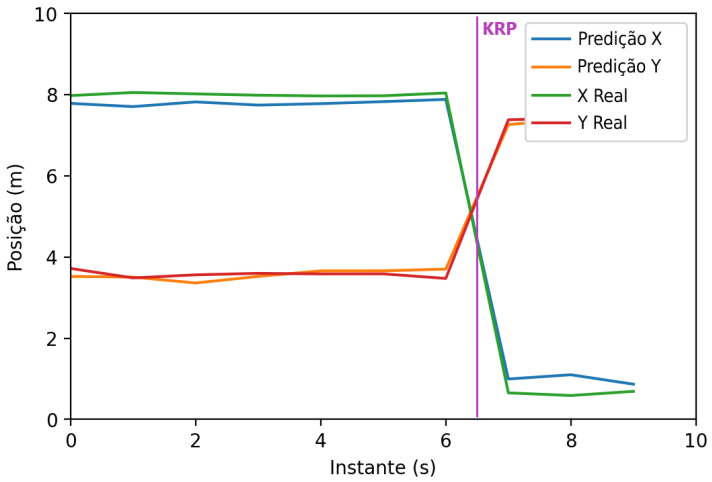

Figura 89 – Resultado do teste para o modelo base LSTM e testado com 30% de atenuação, mantendo boa precisão nas predições.

Fonte: Autor.

Resultados obtidos de erro médio e desvio padrão durante os testes com 30% de atenuação para os modelos base, com os dados estatísticos na figura 90:

a) LSTM: 0,43m $\pm$ 0,36m

b) GRU: 0,53m $\pm$ 0,40m

c) KNN: 1,38m $\pm$ 1,52m

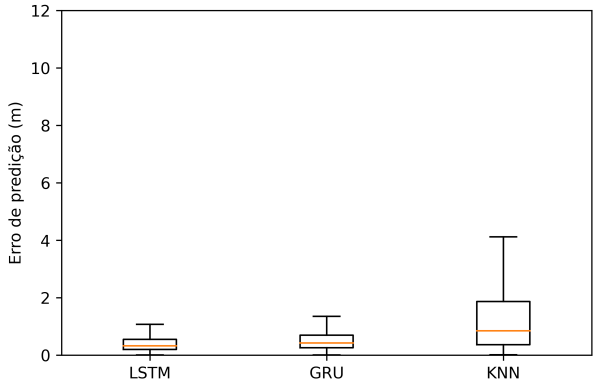

Figura 90 – Comparação estatística dos resultados para os modelos base, utilizando *stream* de dados de 10 segundos. Na análise ANOVA, temos $F = 7902$ e $p = 0{,}0$, indicando que os resultados possuem desempenho significativamente diferentes.

Fonte: Autor.

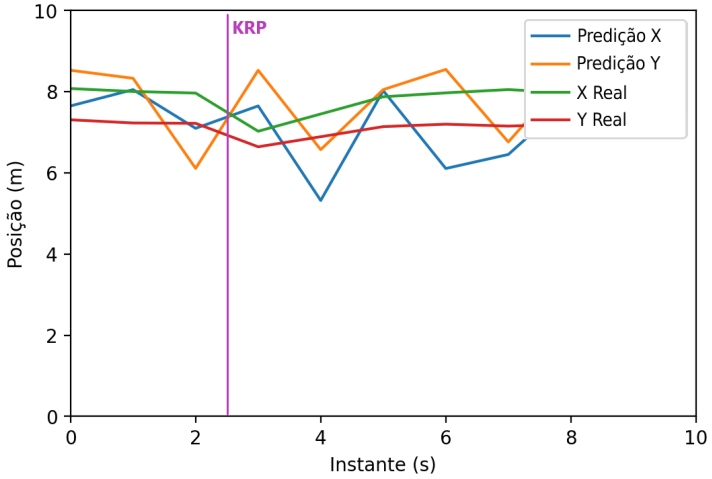

Figura 91 – Resultado do teste para o modelo base KNN e testado com 30% de atenuação, com alta oscilação nas predições.

Fonte: Autor.

Quando analisa-se os resultados para a solução baseada na proposta KNN, ao intensificar a atenuação, as distâncias entre os vizinhos mais próximos oscila com maior frequência, devido as distâncias geradas pela falta de dados representativos com menor quantidade de APs na base de treinamento. O resultado observa-se na figura 91, quando uma oscilação contínua compromete os resultados gerais obtidos de predição para o teste.

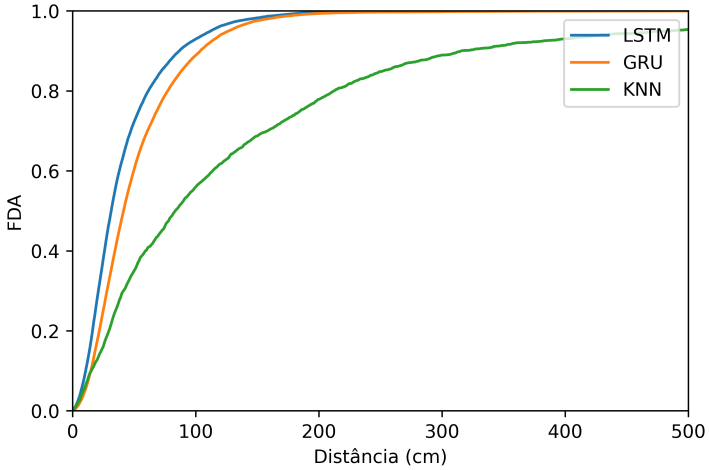

Figura 92 – Resultado comparativo da FDA para os modelos específicos e testados com 30% de atenuação.

Fonte: Autor.

Avaliando os resultados obtidos com 30% de atenuação, podemos observar o baixo impacto que os níveis testados geraram nas predições baseadas em RNR, enquanto a KNN aumentou a oscilação perante dados com baixa semelhança na base de treinamento, ampliando as distâncias entre os vizinhos mais próximos.

6 Discussões e Conclusões

Este trabalho buscou testar diversas condições para avaliar os sistemas sugeridos pelos trabalhos relacionados em termos de robustez e aplicabilidade referente a quantidade de APs, variabilidade dos níveis de ruído e de atenuação dentro de um ambiente virtual controlado no software Webots®.

6.1 Variação na densidade de *access points*

Considerando o resultado geral das RNRs, menos que 6 APs na simulação resultou em flutuações em períodos de estabilidade do robô e apresentando atraso na detecção dos sequestros de robô nos modelos base, devido ao dados de treinamento não serem totalmente comparáveis aos dados testados em geral.

O treinamento de modelos específicos resolveu parcialmente o problema, mas com menos de 3 APs, nem mesmo a seleção do AP com melhor distribuição de sinais RSS no ambiente pôde trazer o erro médio abaixo de 2 metros, com atraso na detecção de um evento de sequestro em até 6 segundos.

Para os modelos base KNN, os resultados foram semelhantes ao das RNRs, com dificuldade de localização com menos de 3 APs, e utilizando modelo específico para a condição, a falta de capacidade do sistema em interpretar dados históricos causou um alto desvio padrão nas predições, piorando o resultado final de acurácia. A utilização de dados de mais APs poderia melhorar a precisão de todas as propostas avaliadas neste estudo.

Nos testes com baixa quantidade de APs e/ou alto nível de atenuação, a direção de movimentação do robô pode influenciar os resultados do modelo treinado com os mesmos parâmetros, devido a variabilidade dos dados apresentados no histórico de dados de entrada nas RNRs.

Após análise de diversas variações com apenas um único AP, esgotamos as possibilidades de utilizar um modelo único contendo apenas dados de teste com vários APs e que consiga obter precisão razoável ou minimamente convergível para realizar a localização aproximada do robô, independente da realização do KRP.

6.2 Variação no nível de ruído

Variações nos níveis de ruído revelaram que, para os modelos base de RNRs e testados com ruído até 2dBm, o impacto nas predições foi baixo devido à estabilidade dos dados de treinamento e a avaliação de dados históricos dentro das RNRs. Quando avaliados instantes de ocorrência de KRP, um pequeno atraso de 1 segundo foi observado nos testes devido à variação maior dos dados de RSS, afetando o acionamento dos *gates* de esquecimento.

Utilizando modelos específicos para cada caso gerou sobreposição nos dados de treinamento e comprometeu os resultados gerais até 2dBm. Somente acima de 2dBm, em situações com alto nível de ruído, que houve benefício para os modelos específicos, quando a alta sobreposição de resultados foi melhor interpretada pelos modelos treinados, mas mesmo com melhor desempenho, a precisão geral não foi satisfatória, com alto desvio padrão.

O nível de ruído até 2dBm gerou grande impacto na performance dos modelos base KNN durante períodos de estabilidade devido à falta de vizinhos próximos, mas não houve impacto na detecção de um KRP.

Quando testados os modelos específicos de KNN, o resultado foi o oposto do ocorrido nas RNRs, com melhoria na precisão da localização em geral.

Com 10dBm nenhum sistema conseguiu atingir precisão aceitável na localização do robô, mas foi a condição em que o KNN obteve melhor desempenho, devido às RNRs se comportarem de modo semelhante, acionando os *gates* de esquecimento a todo instante.

6.3 Variação no nível de atenuação

Variando os níveis de atenuação até 30% não apresentou nenhum grande impacto nas RNRs, tanto para modelos base quanto modelos específicos, com apenas uma pequena oscilação na predição e um leve atraso de 1 segundo na detecção de eventos de sequestro.

O modelo base KNN apresentou certa dificuldade em manter estabilidade nas predições já em 20% de atenuação, piorando o desvio padrão devido a alta oscilação provocada pela distância entre os dados de teste e os dados de treinamento. Os resultados com 30% só intensificaram os mesmos pontos, mas ainda manteve resultados satisfatórios, com erro médio abaixo dos 2 metros.

Modelos KNN específicos para as condições de teste melhoraram os resultados de predição, com maior representatividade de vizinhos mais próximos.

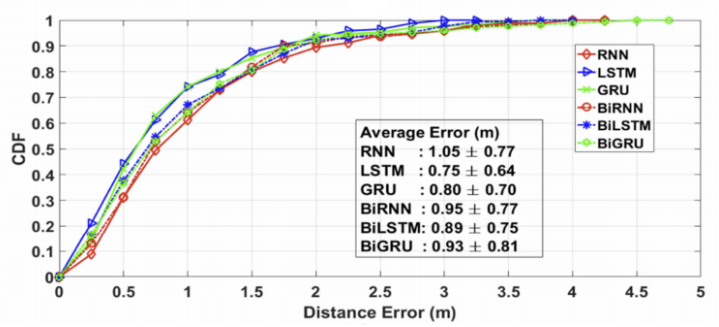

Figura 93 – Resultados obtidos por Hoang et al. (2019) para sistemas GRU e LSTM em gráfico de função de distribuição acumulada

6.4 Observações finais

Entre as RNRs, os modelos utilizando LSTM foram mais precisos que modelos GRU, como esperado, com o KNN apenas atuando melhor que as RNRs em situações extremas, quando os *gates* de esquecimento foram acionados o tempo todo, ignorando os dados históricos apresentados.

Durante os testes foram obtidos resultados até melhores que os trabalhos de Hoang et al. (2019) e Abbas et al. (2019) como observa-se na figura 93, mas tal solução foi influenciada pela limitação no ruído e na atenuação gerada pelo simulador durante a geração dos dados de treinamento em ambiente controlado.

Quando observamos os resultados do sistema KNN, o aumento da quantidade de pontos de avaliação referente ao trabalho de Xu e Chou (2017) (de 20 para 326) não afetou os resultados, que apresentaram precisões próximas ao trabalho original, mas apresenta instabilidade na predição dos valores quando o padrão de dados de teste apresentados foge muito dos dados de treinamento apresentados, seja por falta de APs, ruído ou desvanecimento, gerando erros contantes de predição devido à falta de vizinhos significativos para convergir os dados.

O KNN por si não é uma solução eficaz para localização quando apresentado sozinho, mas é uma poderosa ferramenta quando associada a outra solução de refino de localização (conforme o trabalho de Xu e Chou (2017)) para relocalizar-se durante o KRP, pois não sofre influência do histórico de dados.

A avaliação da solução KNN com diferentes valores k de vizinhos mais próximos em condições com maiores níveis de ruído e atenuação pode apresentar melhores resultados, mas estudos passados indicam que a modificação reduzirá a precisão em instantes com maior qualidade de informação, enviesando os resultados gerais.

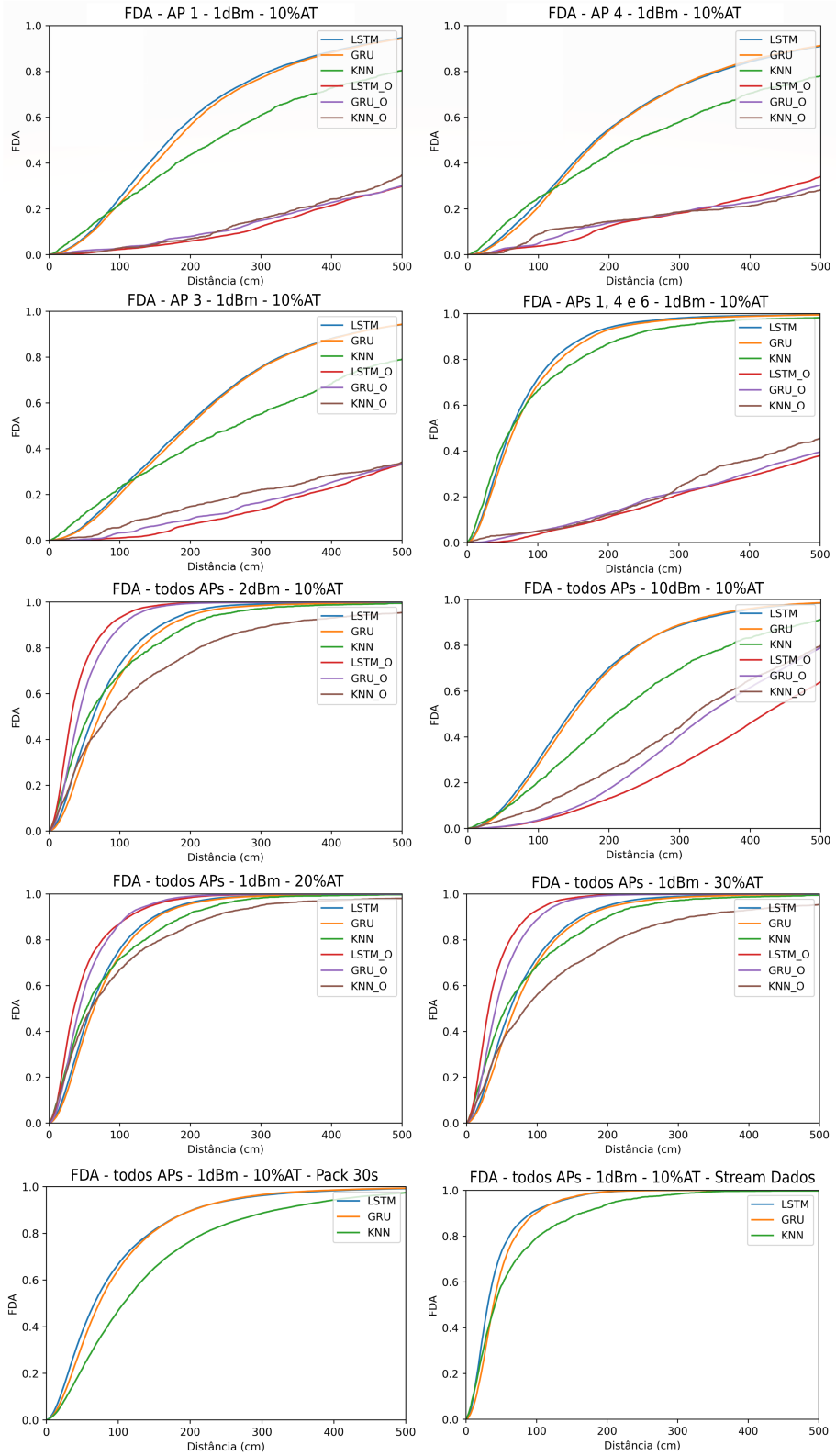

Figura 94 – Resultados obtidos em função de distribuição acumulada para os diversos experimentos deste estudo. Os resultados com "_O" na legenda indicam que foi utilizado o modelo treinado com os parâmetros base deste estudo.

Fonte: Autor.

7 Trabalhos futuros

Não foi possível reproduzir a conclusão de Zafari, Gkelias e Leung (2019), quando observou que uma grande quantidade de APs pode gerar mais interferência que melhoria na localização devido ao comportamento simulado e congelado no momento do *fingerprinting*, mas também os testes realizados não atingiram o platô de localização, indicando que novos testes com maior quantidade de APs ainda podem melhorar os resultados obtidos neste trabalho.

Entre outros trabalhos futuros, a avaliação da influência de conflitos de canal entre APs e comparação entre a variação dos *fingerprints* nessas condições e outras previstas por Chapre et al. (2013) poderiam incrementar a representatividade do gerador de dados RSS no ambiente virtual com outras variabilidades dos sinais de RSS, aprimorando a geração da base de dados com novas condições de funcionamento para futuros estudos na área.

Uma nova proposta de estrutura de rede para as RNRs, variando a quantidade de nós e camadas ocultas e recorrências entre camadas, pode melhorar a predição especificamente para ocorrências de KRP.

Outro trabalho relacionado que poderia ser implementado nos próximos estudos na área consiste em um sistema de atualização contínua do *figerpriting*, técnica chamada de *Simultaneous Localization And Mapping* (SLAM), como proposto por Mirowski, Palaniappan e Ho (2012), mas utilizada em diversos trabalhos. A técnica não foi utilizada neste estudo por fugir do escopo da análise de performance dos sistemas de localização, sendo o SLAM uma técnica para atualização do *fingerprinting* no longo prazo, ao ocorrer mudanças mais significativas no ambiente em níveis de RSS.

Outro estudo não coberto na análise deste trabalho seria realizar testes de modelos treinados com dados de treinamento compostos por combinações de diversos exemplos propostos nesta análise, buscando melhor flexibilidade dos sistemas de localização em condições adversas de base de informação, principalmente para o KNN.

REFERÊNCIAS

ABBAS, M. et al. WiDeep: WiFi-based Accurate and Robust Indoor Localization System using Deep Learning. In: 2019 IEEE International Conference on Pervasive Computing and Communications (PerCom. [S.l.: s.n.], 2019. P. 1–10.

BAHL, P.; PADMANABHAN, V. N. RADAR: an in-building RF-based user location and tracking system. In: PROCEEDINGS IEEE INFOCOM 2000. Conference on Computer Communications. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies (Cat. No.00CH37064). [S.l.: s.n.], 2000. v. 2, 775–784 vol.2.

CARRERA VILLACRÉS, J. L. et al. A Particle Filter-Based Reinforcement Learning Approach for Reliable Wireless Indoor Positioning. **IEEE Journal on Selected Areas in Communications**, v. 37, n. 11, p. 2457–2473, 2019.

CHANDRAKANTHA, Leslie. Learning ANOVA concepts using simulation. In: IEEE. PROCEEDINGS of the 2014 Zone 1 Conference of the American Society for Engineering Education. [S.l.: s.n.], 2014. P. 1–5.

CHAPRE, Y. et al. Received signal strength indicator and its analysis in a typical WLAN system (short paper). In: 38TH Annual IEEE Conference on Local Computer Networks. [S.l.: s.n.], 2013. P. 304–307.

CHO, Kyunghyun et al. **Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation**. [S.l.: s.n.], 2014. arXiv: 1406.1078 [cs.CL].

CHUNG, Junyoung et al. Empirical evaluation of gated recurrent neural networks on sequence modeling. **arXiv preprint arXiv:1412.3555**, 2014.

DARAMOUSKAS, I.; KAPOULAS, V.; PARASKEVAS, M. Using Neural Networks for RSSI Location Estimation in LoRa Networks. In: 2019 10th International Conference on Information, Intelligence, Systems and Applications (IISA). [S.l.: s.n.], 2019. P. 1–7.

DE MULDER, Wim; BETHARD, Steven; MOENS, Marie-Francine. A survey on the application of recurrent neural networks to statistical language modeling. **Computer Speech & Language**, Elsevier, v. 30, n. 1, p. 61–98, 2015.

DEISENROTH, Marc. **Mathematics for machine learning**. Cambridge New York, NY: Cambridge University Press, 2020. ISBN 9781108455145.

DESROCHERS, Benoît; LACROIX, Simon; JAULIN, Luc. Set-membership approach to the kidnapped robot problem. In: IEEE. 2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). [S.l.: s.n.], 2015. P. 3715–3720.

DIAMANTAS, Sotirios. Towards resolving the kidnapped robot problem: Topological localization from crowdsourcing and georeferenced images. In: SPRINGER. SCIENCE and Information Conference. [S.l.: s.n.], 2019. P. 551–563.

DING, Xinhui; DONG, Shi. Improving Positioning Algorithm Based on RSSI. **Wireless Personal Communications**, v. 110, n. 4, p. 1947–1961, fev. 2020. ISSN 1572-834X. DOI: 10.1007/s11277-019-06821-0. Disponível em: <https://doi.org/10.1007/s11277-019-06821-0>.

AL-DOSARY, Naji Mordi Naji; AL-HAMED, Saad Abdulrahman; ABOUKARIMA, Abdulwahed Mohamed. K-Nearest Neighbors Method For Prediction Of Fuel Consumption In Tractor-Chisel Plow Systems. **Engenharia Agrícola**, SciELO Brasil, v. 39, p. 729–736, 2019.

FOX, Dieter et al. Particle filters for mobile robot localization. In: SEQUENTIAL Monte Carlo methods in practice. [S.l.]: Springer, 2001. P. 401–428.

GERON, A. **Hands-on machine learning with scikit-learn, keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems**. O'Reilly Media: [s.n.], 2019.

GERS, F. A.; SCHMIDHUBER, J.; CUMMINS, F. Learning to forget: continual prediction with LSTM. In: 1999 Ninth International Conference on Artificial Neural Networks ICANN 99. (Conf. Publ. No. 470). [S.l.: s.n.], 1999. v. 2, 850–855 vol.2.

GOOGLE. **Google Maps Geolocation API**. [S.l.], Feb. 2018. Disponível em: <https://developers.google.com/maps/documentation/geolocation/intro>.

GUO, Gongde et al. KNN model-based approach in classification. In: SPRINGER. OTM Confederated International Conferences"On the Move to Meaningful Internet Systems". [S.l.: s.n.], 2003. P. 986–996.

HOANG, M. T. et al. Recurrent Neural Networks for Accurate RSSI Indoor Localization. **IEEE Internet of Things Journal**, v. 6, n. 6, p. 10639–10651, 2019.

HOCHREITER, Sepp; SCHMIDHUBER, Jürgen. Long Short-Term Memory. **Neural Comput.**, MIT Press, Cambridge, MA, USA, v. 9, n. 8, p. 1735–1780, nov. 1997. ISSN 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. Disponível em: <https://doi.org/10.1162/neco.1997.9.8.1735>.

KAUFMAN, Leonard; ROUSSEEUW, Peter J. **Finding groups in data: an introduction to cluster analysis**. [S.l.]: John Wiley & Sons, 2009. v. 344.

KHAN, M. A. et al. Location Awareness in 5G Networks Using RSS Measurements for Public Safety Applications. **IEEE Access**, v. 5, p. 21753–21762, 2017.

KUMAR, P.; REDDY, L.; VARMA, S. Distance measurement and error estimation scheme for RSSI based localization in Wireless Sensor Networks. In: 2009 Fifth International Conference on Wireless Communication and Sensor Networks (WCSN). [S.l.: s.n.], 2009. P. 1–4.

LADD, Andrew M et al. Robotics-based location sensing using wireless ethernet. **Wireless networks**, Springer, v. 11, n. 1-2, p. 189–204, 2005.

LAOUDIAS, C. et al. A Survey of Enabling Technologies for Network Localization, Tracking, and Navigation. **IEEE Communications Surveys Tutorials**, v. 20, n. 4, p. 3607–3644, 2018.

LI, Chenyang; MO, Lingfei; ZHANG, Dongkai. Review on UHF RFID localization methods. **IEEE Journal of Radio Frequency Identification**, IEEE, v. 3, n. 4, p. 205–215, 2019.

LIPTON, Zachary C.; BERKOWITZ, John; ELKAN, Charles. **A Critical Review of Recurrent Neural Networks for Sequence Learning**. [S.l.: s.n.], 2015. arXiv: 1506.00019 [cs.LG].

MAJDIK, Andras et al. New approach in solving the kidnapped robot problem. In: VDE. ISR 2010 (41st International Symposium on Robotics) and ROBOTIK 2010 (6th German Conference on Robotics). [S.l.: s.n.], 2010. P. 1–6.

MIROWSKI, Piotr; PALANIAPPAN, Ravishankar; HO, Tin Kam. Depth camera SLAM on a low-cost WiFi mapping robot. In: IEEE. 2012 IEEE International Conference on Technologies for Practical Robot Applications (TePRA). [S.l.: s.n.], 2012. P. 1–6.

EL-NAGGAR, Abdelrahman; WASSAL, Amr; SHARAF, Khaled. Indoor Positioning Using WiFi RSSI Trilateration and INS Sensor Fusion System Simulation. In: PROCEEDINGS of the 2019 2nd International Conference on Sensors, Signal and Image Processing. Prague, Czech Republic: Association for Computing Machinery, 2019. (SSIP 2019), p. 21–26. ISBN 9781450372435. DOI: 10.1145/3365245.3365261. Disponível em: <https://doi.org/10.1145/3365245.3365261>.

PERAHIA, Eldad. IEEE 802.11n Development: History, Process, and Technology. **Communications Magazine, IEEE**, v. 46, p. 48–55, ago. 2008. DOI: 10.1109/MCOM.2008.4557042.

RUSSELL, Stuart; NORVIG, Peter. **Artificial Intelligence: A Modern Approach**. 3rd. USA: Prentice Hall Press, 2009. ISBN 0136042597.

SAARI, M. et al. LoRa — A survey of recent research trends. In: 2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO). [S.l.: s.n.], 2018. P. 0872–0877.

SADOWSKI, S.; SPACHOS, P. Optimization of BLE Beacon Density for RSSI-Based Indoor Localization. In: 2019 IEEE International Conference on Communications Workshops (ICC Workshops). [S.l.: s.n.], 2019. P. 1–6.

SEOW, Y. et al. Detecting and solving the kidnapped robot problem using laser range finder and wifi signal. In: 2017 IEEE International Conference on Real-time Computing and Robotics (RCAR). [S.l.: s.n.], 2017. P. 303–308.

SOUZA, Deivison Venicio et al. k-NEAREST NEIGHBOR AND LINEAR REGRESSION IN THE PREDICTION OF THE ARTIFICIAL FORM FACTOR. **FLORESTA**, v. 50, n. 3, p. 1669–1678, 2020.

STOWELL, Sarah. Probability Distributions. In: USING R for Statistics. Berkeley, CA: Apress, 2014. P. 87–98. ISBN 978-1-4842-0139-8. DOI: 10.1007/978-1-4842-0139-8_7. Disponível em: <https://doi.org/10.1007/978-1-4842-0139-8_7>.

TIAN, X. et al. Optimization of Fingerprints Reporting Strategy for WLAN Indoor Localization. **IEEE Transactions on Mobile Computing**, v. 17, n. 2, p. 390–403, 2018.

WETTSCHERECK, Dietrich; AHA, David W; MOHRI, Takao. A review and empirical evaluation of feature weighting methods for a class of lazy learning algorithms. **Artificial Intelligence Review**, Springer, v. 11, n. 1, p. 273–314, 1997.

WILSON, D Randall; MARTINEZ, Tony R. Reduction techniques for instance-based learning algorithms. **Machine learning**, Springer, v. 38, n. 3, p. 257–286, 2000.

XU, Song; CHOU, Wusheng. An improved indoor localization method for mobile robot based on WiFi fingerprint and AMCL. In: IEEE. 2017 10th International Symposium on Computational Intelligence and Design (ISCID). [S.l.: s.n.], 2017. v. 1, p. 324–329.

XU, Song; CHOU, Wusheng; DONG, Hongyi. A Robust Indoor Localization System Integrating Visual Localization Aided by CNN-Based Image Retrieval with Monte Carlo Localization. **Sensors (Basel, Switzerland)**, v. 19, 2019.

XUE, M. et al. Locate the Mobile Device by Enhancing the WiFi-Based Indoor Localization Model. **IEEE Internet of Things Journal**, v. 6, n. 5, p. 8792–8803, 2019.

YANG, Zheng; ZHOU, Zimu; LIU, Yunhao. From RSSI to CSI: Indoor Localization via Channel Response. **ACM Comput. Surv.**, Association for Computing Machinery, New York, NY, USA, v. 46, n. 2, dez. 2013. ISSN 0360-0300. DOI: 10.1145/2543581.2543592. Disponível em: <https://doi.org/10.1145/2543581.2543592>.

YI, Chuho; CHOI, Byung-Uk. Detection and recovery for kidnapped-robot problem using measurement entropy. In: SPRINGER. INTERNATIONAL Conference on Grid and Distributed Computing. [S.l.: s.n.], 2011. P. 293–299.

YIU, Simon et al. Wireless RSSI fingerprinting localization. **Signal Processing**, Elsevier, v. 131, p. 235–244, 2017.

YOUSSEF, M.; AGRAWALA, A. Small-scale compensation for WLAN location determination systems. In: 2003 IEEE Wireless Communications and Networking, 2003. WCNC 2003. [S.l.: s.n.], 2003. v. 3, 1974–1978 vol.3.

ZAFARI, F.; GKELIAS, A.; LEUNG, K. K. A Survey of Indoor Localization Systems and Technologies. **IEEE Communications Surveys Tutorials**, v. 21, n. 3, p. 2568–2599, 2019.

ZÀRUBA, Gergely V et al. Indoor location tracking using RSSI readings from a single Wi-Fi access point. **Wireless networks**, Springer, v. 13, n. 2, p. 221–235, 2007.