UNIVERSIDADE FEDERAL DE JUIZ DE FORA
FACULDADE DE ENGENHARIA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

João Paulo Sales Henriques Lima

Aplicações de Handover Inteligente em Redes Celulares Baseadas em Aprendizado de Máquina

Juiz de Fora
2022

João Paulo Sales Henriques Lima

Aplicações de Handover Inteligente em Redes Celulares Baseadas em Aprendizado de Máquina

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Juiz de Fora como requisito parcial à obtenção do título de Mestre em Engenharia Elétrica. Área de concentração: Sistemas Eletrônicos

Orientador: Prof. Dr. Álvaro A. M. de Medeiros

Coorientador: Prof. Dr. Eduardo Pestana de Aguiar

Juiz de Fora

2022

Ficha catalográfica elaborada através do Modelo Latex do CDC da UFJF
com os dados fornecidos pelo(a) autor(a)

Lima, João Paulo S. H..
Aplicações de Handover Inteligente em Redes Celulares Baseadas em Aprendizado de Máquina / João Paulo Sales Henriques Lima. – 2022.
58 f. : il.

Orientador: Álvaro A. M. de Medeiros
Coorientador: Eduardo Pestana de Aguiar
Dissertação (Mestrado) – Universidade Federal de Juiz de Fora, Faculdade de Engenharia. Programa de Pós-Graduação em Engenharia Elétrica, 2022.

1. Handover. 2. Inteligência Artificial. 3. Redes Móveis. I. de Medeiros, Álvaro A. M., orient. II. de Aguiar, Eduardo P., coorient. III. Título.

João Paulo Sales Henriques Lima

Aplicações de Handover Inteligente em Redes Celulares Baseadas em Aprendizado de Máquina

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Juiz de Fora como requisito parcial à obtenção do título de Mestre em Engenharia Elétrica. Área de concentração: Sistemas Eletrônicos

Aprovada em 12 de julho de 2022.

BANCA EXAMINADORA

Prof. Dr. Álvaro Augusto Machado de Medeiros - Orientador
Universidade Federal de Juiz de Fora

Prof. Dr. Eduardo Pestana de Aguiar - Coorientador
Universidade Federal de Juiz de Fora

Prof. Dr. Edelberto Franco Silva
Universidade Federal de Juiz de Fora

Profa. Dra. Michelle Soares Pereira Facina
Centro de Pesquisa e Desenvolvimento em Telecomunicações

Juiz de Fora, 23/06/2022.

Documento assinado eletronicamente por **Eduardo Pestana de Aguiar**, **Professor(a)**, em 12/07/2022, às 12:08, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Alvaro Augusto Machado de Medeiros**, **Professor(a)**, em 12/07/2022, às 14:04, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Michelle Soares Pereira Facina**, **Usuário Externo**, em 12/07/2022, às 14:25, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Edelberto Franco Silva**, **Professor(a)**, em 12/07/2022, às 16:32, conforme horário oficial de Brasília, com fundamento no § 3º do art. 4º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no Portal do SEI-Ufjf (www2.ufjf.br/SEI) através do ícone Conferência de Documentos, informando o código verificador **0840442** e o código CRC **312A30F1**.

AGRADECIMENTOS

Agradeço aos meus pais pelo apoio incondicional durante toda a jornada deste trabalho. Pelo estímulo e por tudo que não consigo descrever em palavras. Foi um período mais longo que planejei, com muitas mudanças e desafios no trajeto.

Aos meus familiares, pela boa convivência, pelos momentos de risadas e carinhos nesses anos.

Aos meus grandes amigos, que estiveram comigo em todo o tempo e proporcionaram (e proporcionam) a vida com mais felicidades e música.

À Fernanda, pelo amor, convivência, paciência e apoio em todos esses anos, além de boas edições de imagens (algumas ao longo desse texto, inclusive).

Ao Álvaro, pela insistência, pelo conhecimento, pelos ótimos papos e por estimular com entusiasmo vários projetos ao longo do curso.

Ao Eduardo, Edelberto e Vicente, que proporcionaram grande aprendizado em todo esse período.

À UFJF, por todos os anos excelentes vividos com grandes alegrias, crescimento e desenvolvimento.

Ao PPEE e à CAPES, pelo apoio e suporte durante o trabalho.

Aos engenheiros Marcelo Nunes e Alysson Reis e a equipe de medição de campo da Anatel-Belo Horizonte, pelos dados cedidos para avaliação em redes reais.

“Bom mesmo é ser um realista esperançoso.” (Ariano Suassuna).

RESUMO

Este trabalho se dedica a estudar, implementar e avaliar diferentes estratégias para a integração de modelos de Inteligência Artificial e Aprendizado de Máquina em procedimentos de *handover* em redes celulares. As propostas de arquitetura para implementação são desenvolvidas baseadas em dados obtidos através de simulações de redes LTE (*Long Term Evolution*) utilizando o simulador *ns-3*, bem como campanhas de coletas de dados em rede real de uma empresa operadora celular em Belo Horizonte. Os resultados observados demonstram grande capacidade dos modelos em gerar respostas com alto grau de acurácia e baixos valores de desvio padrão, tanto em tarefas de classificação, quanto naquelas de regressão propostas. São contemplados diversos modelos presentes na literatura ao longo das avaliações, desde alguns muito consolidados, como máquinas de vetores de suporte, aqueles com arquitetura baseada em árvores, redes neurais artificiais, bem como sistemas *fuzzy*, redes neurais LSTM, além de avançadas máquinas *boosting*.

Palavras-chave: Handover. Inteligência Artificial. Redes Móveis. Aprendizado de Máquina.

ABSTRACT

This work aims to study, implement and evaluate different strategies to integrate Artificial Intelligence and Machine Learning models to handover procedures in cellular networks. The architecture proposals for the implementation are developed based on data from simulations of LTE networks with *ns-3* simulator, and data collection campaigns on live network from a mobile network operator in Belo Horizonte. The observed results demonstrate great capacity of the models to produce outputs with high level of accuracy and diminished values of standard deviation, in both classification and regression proposed tasks. A wide range of models from the literature are encompassed in the evaluation, since more traditional ones, such as support vector machines, those based on tree architecture, artificial neural networks, as well as fuzzy systems, LSTM neural networks and latter boosting machines.

Keywords: Handover. Artificial Intelligence. Mobile Networks. Machine Learning

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

QoE	*Quality of Experience*
LTE	*Long Term Evolution*
NR	*New Radio*
UE	*User Equipment*
RSRP	*Reference Signal Received Power*
RSRQ	*Reference Signal Received Quality*
3GPP	*3rd Generation Partnership Project*
TTT	*Time to Trigger*
IA	Inteligência Artificial
ML	*Machine Learning*
RNA	Rede Neural Artificial
SVM	*Support Vector Machine*
RF	*Random Forest*
GBM	*Gradient Boosting Machine*
LightGBM	*Light Gradient Boosting Machine*
XGBoost	*Extreme Gradient Boosting Machine*
KNN	*K-Nearest Neighbor*
AP	*Access Point*
RNN	*Recurrent Neural Network*
MCS	*Modulation Coding Scheme*
LSTM	*Long Short-Term Memory*
RLF	*Radio Link Failure*
BS	*Base Station*
ALMMo	*Autonomous Learning Multimodel System*
SOFL	*Self-Organizing Fuzzy Logic Classifier*
T2	Sistema Fuzzy do Tipo-2
MLP	*Multilayer Perceptron*
REM	*Radio Environment Map*
eNB	*Evolved NodeB*
SINR	*Signal to Interference-Plus-Noise Ratio*
TCP	*Transmission Control Protocol*
EMA	Erro Médio Absoluto

SUMÁRIO

1 INTRODUÇÃO

A expansão do uso de tecnologias sem fio para comunicações se dá de maneira veloz ano após ano e cada vez mais ferramentas instigam um uso ainda maior de dados em redes de comunicação. Numa breve perspectiva histórica, no fim do século XX, tais redes já apresentavam serviços sofisticados, como os e-mails, as redes celulares iniciais, a Internet, transmissões via satélite, entre outras tecnologias. No entanto, a evolução acelerada das tecnologias de informação e comunicação no início dos anos 2000 propiciou um terreno fértil para o enorme salto na quantidade de usuários e na escala de serviços por eles demandados.

Atualmente, com os *smartphones* dramaticamente popularizados, redes de banda larga sem fio são imperativas para sustentar tráfegos como *streamings* de vídeos em alta qualidade, levando em conta também o movimento dos usuários. Nos cenários atuais, o desafio de promover, com eficácia e alta qualidade de experiência para os usuários, uma rede celular em centros densamente urbanizados é enorme por muitos fatores: a demanda de dados é altíssima; o número de usuários em tais locais é grande, além da distribuição deles no espaço não ser homogênea; os tipos de tráfegos exigidos são diversos (como *streamings*, chamadas tradicionais, *downloads* de arquivos, navegação web, etc); a movimentação dos usuários enquanto utilizam recursos sem fio também aumenta com o passar dos anos, entre outros aspectos.

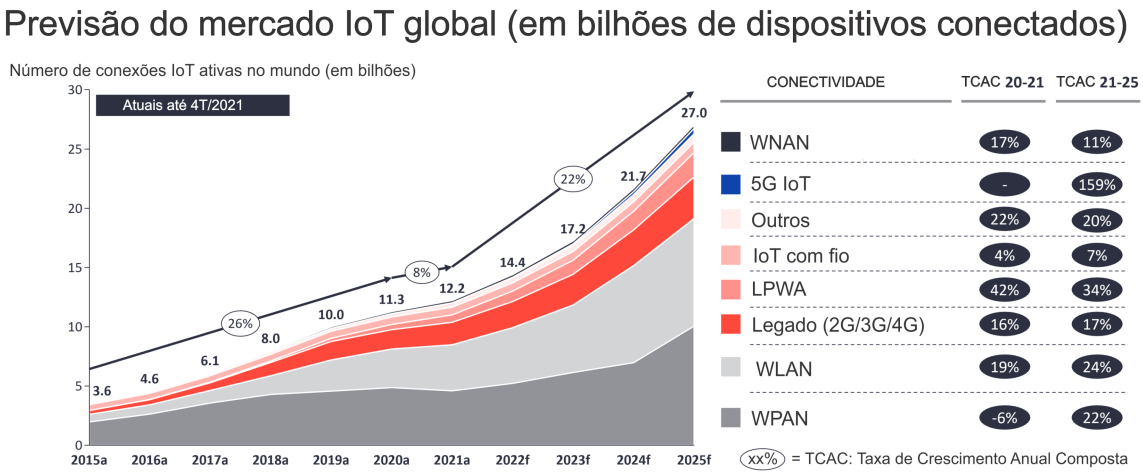
Nesse sentido, a sucessão dos padrões internacionais de comunicação celular se justifica pela busca da otimização dos recursos utilizados, visando promover, para mais usuários, a disponibilidade de tráfegos de maior demanda com maior estabilidade. Através do aperfeiçoamento dos processos, é possível extrair grandes ganhos de eficiência em tais redes.

Na quinta geração das especificações do padrão para comunicações celulares, destacam-se as *small cells*, um tipo de célula que visa proporcionar altas taxas de velocidade através de melhor qualidade do sinal, justamente por serem projetadas para a cobertura de pequenas áreas (da ordem de centenas de metros). Elas desempenham um papel fundamental, visto que o 5G terá operações em diversas faixas de frequências, desde bandas abaixo de 1 GHz até as ondas milimétricas, acima dos 28 GHz. Sabendo-se que, com frequências mais altas, as dificuldades de propagação se acentuam, será cada vez mais necessária a instalação de um maior número de antenas para cobrir uma mesma região, visando o oferecimento de maior banda para os usuários. Dessa forma, a implementação de tais redes será em cenários cada vez mais densos.

Nesse contexto, a implementação de *small cells* também traz grandes desafios, com destaque para o procedimento de *handover*. Esse procedimento é responsável pela transferência de uma sessão de um usuário da rede de uma célula para outra, sem que

haja perdas de conexão. Com o maior número de células coexistindo em uma mesma área, dada a mobilidade de um usuário, o número de *handovers* tende a crescer drasticamente. E além disso, o número de disparos de *handover* crescerá ainda mais, caso os algoritmos permaneçam ineficientes na seleção adequada do momento da transição.

Outra tendência das redes de comunicação que também intensificará o desafio para o procedimento de *handover* é a crescente presença da Internet das Coisas observada nos últimos anos. Projeta-se que o número de dispositivos móveis conectados num ambiente urbano aumentará significativamente, conforme visto na Figura 1 [1]. Não mais veremos apenas os aparelhos celulares conectados a consumir recursos das redes celulares. Teremos também a conectividade ampliada para os objetos portáteis pessoais, os veículos, os itens urbanos das cidades inteligentes, as grandes redes de sensores integrados, as ferramentas de segurança e monitoramento, entre outros inúmeros elementos. E com a mobilidade de cada vez mais elementos conectados, mais *handovers* serão efetuados, impondo um desafio ainda maior para a utilização da banda com carga de sinalização.



– Fig. 1: Previsão do crescimento de dispositivos conectados. Adaptado de [1].

Esse talvez seja o ponto que chama a maior atenção para a questão do *handover*. A eficiência espectral degradada por processos de *handover* ineficazes prejudica não somente a qualidade de experiência (QoE) dos usuários, como também a eficiência espectral da rede e, portanto, a sua lucratividade. A eficiência espectral deriva do Teorema de Shannon sobre a capacidade de um canal de comunicação e é avaliada em *bits*/segundo/Hz. Ou seja, é uma métrica que indica a capacidade de produzir transmissões de dados (*bits*/s) para cada Hz disponível na banda daquele serviço. Se a rede utiliza muitos bits com mensagens de coordenação da própria rede, isto é, com mensagens de sinalização, menor a banda restante disponível para alocação dos serviços prestados por uma operadora.

1.1 *Handover* nos padrões LTE e NR atualmente

O *handover* é o procedimento de rede que executa a transferência da sessão de um usuário de uma célula para outra quando este se movimenta. É necessário que exista uma ação bem coordenada entre os elementos da rede para que, ao se movimentar e se afastar da célula que promove sua conexão, o usuário seja realocado em outra célula, sem que haja uma interrupção no seu serviço, como perda de conexão ou queda em sua chamada, por exemplo. No padrão LTE (*Long Term Evolution*), tratado popularmente como 4G, por ser a quarta geração de tais padrões, e mesmo no padrão NR (*New Radio*), ou 5G, esse processo é estabelecido a partir de comparações muito simplórias entre os níveis de potência dos sinais dos enlaces entre o usuário e as células vizinhas. Dessa forma, muitas decisões podem ser tomadas de forma precipitada ou ineficiente, de maneira geral. Tal processo leva a *handovers* ineficientes, necessidades de reestabelecimento de conexões e aos *handovers* ping-pongs, os quais se identificam como *handovers* sucessivos entre as mesmas células em pequenos intervalos de tempo. Quando um usuário se movimenta próximo à borda da célula, o sinal está sujeito a diversas flutuações, de modo que, o mecanismo de *handover* pode ser desencadeado diversas vezes. Com isso, não será promovida ao usuário melhor qualidade do serviço, devido às repetidas reconexões. Por outro lado, apenas a rede será inundada por um grande volume de tráfego de sinalização, o que degrada a eficiência espectral almejada.

Uma mensagem do tipo *Measurement Report* é trocada periodicamente entre os UEs (*User Equipment*) e a rede, a fim de inspecionar as condições do canal. Tais sinais incluem *Received Signal Reference Power* (RSRP) e *Received Signal Reference Quality* (RSRQ), as principais métricas para decisões de *handover* e re-seleção de células. A primeira fornece uma medida de potência de sinal específica por cada célula baseada em uma média linear e inclui a célula que serve o UE naquele momento, bem como células vizinhas que são percebidas por ele. A segunda métrica tem como objetivo trazer uma perspectiva da qualidade do sinal, visto que ela inclui cálculos com a potência do sinal desejado e também daqueles interferentes.

Nos padrões atuais, o procedimento de *handover* é executado após um monitoramento recorrente de tais sinais de referência. O momento de disparo de *handover* é determinado por eventos que relacionam tais medidas, especificados por normas [2]. Alguns deles são descritos na Tabela 1.

Como visto na Tabela 1, o procedimento de *handover* utiliza-se de eventos estabelecidos em norma para sua coordenação e, por isso, esse modelo é denominado *handover* determinístico. As medidas das condições do canal, traduzidas por sinais de referência, RSRP ou RSRQ, fornecem as métricas para a atuação dos algoritmos que executam a decisão do momento e do alvo de *handover*. Por exemplo, um dos algoritmos mais estabelecidos é o *A2A4RSRQ*. Ele se fundamenta nos eventos A2 e A4, bem como na medida de

Tabela 1 – Eventos característicos para *handovers* 3GPP [2].

Evento	**Descrição**
A1	Célula primária (PC) se torna melhor que um limiar
A2	Célula primária (PC) se torna pior que um limiar
A3	Célula secundária se torna melhor que PC por uma margem
A4	Célula secundária se torna melhor que um limiar

RSRQ, como sugere sua denominação. Dessa forma, quando o sinal RSRQ da célula de origem daquele usuário se torna pior que um certo limiar (evento A2), busca-se entre as células vizinhas uma que atenda ao evento A4, ou seja, uma que proporcione uma medida de RSRQ maior que tal limiar. Quando essa condição é atingida, inicia-se uma rápida espera para que essas condições permaneçam atendidas, denominada *Time-To-Trigger* (TTT), da ordem de milissegundos, com valores definidos em norma [3]. Após o TTT, se as condições de A2 e A4 continuarem atendidas, o *handover* é iniciado.

É importante notar que, nesse tipo de algoritmo, não há nenhum mecanismo de inteligência agregado, que considere, por exemplo, as rotas dos usuários através de suas coordenadas, ou mesmo os recentes *handovers* disparados, sejam eles bem ou mal sucedidos. Isto é, se certa condição dos sinais de referência atingida levou a inúmeras perdas de conexão há pouco tempo, isso nada implica numa melhoria que mude a relação de tal condição com o disparo dos próximos. Os *handovers* continuarão a ser disparados, acarretando em novas perdas de conexão.

Isto posto, observa-se que há grande espaço para melhorias nesse processo. Ao produzir *handovers* ineficientes e aumentar a carga de sinalização na rede, reduz-se a eficiência espectral. Em outras palavras, o recurso que é utilizado pelos sistemas de comunicação celular para operarem serviços aos seus clientes é mal empregado para comunicação interna da própria rede. Além de bastante regulada, a utilização do espectro é muito custosa para as operadoras. O direito de fornecer serviços nas faixas de frequências determinadas é obtido em leilões entre órgãos reguladores e operadoras concorrentes. Em novembro de 2021, por exemplo, ocorreu o leilão de faixas de frequência para o 5G no Brasil, e foram arrecadados mais de 40 bilhões de reais [4]. Tal montante representa o quanto as operadoras participantes se dispuseram a pagar para terem simplesmente o direito de prover serviços em tais frequências, além de outras contrapartidas em cobertura, infraestrutura, entre outras áreas.

1.2 Inteligência artificial e aprendizado de máquina

Por sua vez, as técnicas de Inteligência Artificial (IA) apresentam uma nova fronteira do conhecimento a ser incorporada de forma intensa nas ferramentas utilizadas pelas empresas de telecomunicações. Apesar de já estudadas desde a metade do século

passado aproximadamente, tais técnicas frequentemente tinham suas atuações limitadas pelas capacidades de processamento das máquinas da época. No entanto, com os enormes ganhos de poder computacional introduzidos nos processadores acessíveis no mercado, o uso da Inteligência Artificial tem recebido um destaque cada vez maior, visto que podem ser implementadas para variadas tarefas em muitos campos do conhecimento.

A IA é tida como um termo amplo (e até vago) que descreve a ciência que estuda as maneiras de fazer computadores obedecerem a certos comportamentos. Como ramificações, temos os dispositivos interativos do tipo homem-máquina (como, por exemplo, os robôs Alexa e Siri que podemos ter em casa), o áudio e a visão computacionais e a aprendizagem de máquina (ML, sigla para *machine learning*). Esse último termo, por sua vez, muitas vezes é empregado como equivalente à IA. Porém, a aprendizagem de máquina se concentra no estudo dos algoritmos capazes de permitir que sistemas computacionais melhorem seu desempenho através da experiência. Ou seja, durante uma fase inicial em que esses algoritmos são treinados com o uso de uma base de dados, eles tendem a modificar parâmetros e mecanismos de modo a alcançar um melhor resultado. Sob esse conceito poderoso, é possível categorizar as principais atividades do aprendizado de máquina em três amplos grupos:

- Aprendizado supervisionado;
- Aprendizado não supervisionado;
- Aprendizado por reforço.

O aprendizado supervisionado agrupa as tarefas que almejam aprender a partir das relações e dependências estabelecidas entre os alvos de saída previstos e as características de entrada fornecidas. Dessa forma, ao ser apresentado a uma entrada ainda desconhecida, o algoritmo será capaz de prever uma saída, baseado no que foi aprendido com a base de dados de treino utilizada. As técnicas desse grupo de ML são a classificação e a regressão. A primeira tem como objetivo indicar a qual classe pertence certa entrada da base de dados (por exemplo, se a fruta apresentada é uma maçã ou uma banana, se o cliente irá pagar ou não o empréstimo, se o tumor do laudo médico é benigno ou maligno). Já a segunda tem a meta de fornecer uma saída numérica a partir das características de entrada fornecidas (como exemplo, quanto é o salário de uma pessoa, qual o valor de mercado de uma casa, quantos litros de combustível certo automóvel consome).

Já o aprendizado não supervisionado tem como objetivo modelar a estrutura ou a distribuição do dados apresentados na entrada, de forma a aprender melhor com tal modelo. Se diferencia do aprendizado supervisionado pois não possui uma saída correta associada a cada entrada. Sendo assim, as técnicas comuns desse tipo de aprendizado são a associação e o agrupamento. A primeira busca estabelecer regras de associação entre

as diversas entradas (por exemplo, num supermercado, um cliente que compra o produto X, geralmente também compra o produto Y). Por sua vez, a segunda pretende agrupar, quando possível, as entradas de forma a organizá-las em grupos com características comuns (entradas, sobremesas e pratos principais; produtos de higiene e produtos alimentícios; aves, répteis e mamíferos).

Por sua vez, o aprendizado por reforço visa o aprendizado de uma sequência de ações, através da interação do algoritmo com o ambiente em que ele é executado. Ao ser configurado num estado inicial, o algoritmo interage com o ambiente, buscando atingir uma meta pré-estabelecida. A depender do resultado de sua operação com o ambiente, o algoritmo recebe recompensas ou penalidades, as quais são retroalimentadas para modificar as próximas ações do algoritmo. Seu objetivo, portanto, é maximizar a recompensa total. Tal campo de ML é muito utilizado, por exemplo, na programação de jogos, em que um usuário se move no cenário, com diversas interações e objetivos definidos.

Acima, foram mencionadas os principais grupos de técnicas de IA/ML. Mas deve-se notar que, com a sua grande disseminação, muitas delas são aplicadas na área de comunicações sem fio e, destacadamente, no processo de *handover*. A próxima Seção traz uma vasta revisão bibliográfica de diversos trabalhos relevantes relacionados a esses temas.

1.3 Revisão bibliográfica

Recentemente, vários esforços foram produzidos no sentido de implementar algoritmos de inteligência computacional para otimizar diversos parâmetros e solucionar desafios futuros para os sistemas de comunicação sem fio, como discutido em [5]. Prever a *Quality of Experience* (QoE) é uma tarefa relevante e há uma vasta comparação de métodos em [6]. Nesse contexto, a QoE para *streaming* de vídeos em rede LTE pode ser prevista empregando-se Redes Neurais Artificiais (RNAs) em [7], ao passo que o algoritmo de Árvore de Decisão é utilizado em [8] em se tratando de aplicativos populares de *smartphones*. Os autores de [9] abordam a invasão de rede com o uso de Máquina de Vetores de Suporte (SVM, sigla do inglês) e o algoritmo *Random Forest* (RF), enquanto os de [10] implementam métodos de *Deep Learning*. Já em [11] utiliza-se RF para predição de potência de transmissão *uplink* em redes celulares, enquanto em [12] emprega-se *LightGBM* para a predição de tráfego. Um algoritmo para mudança de pistas para veículos autônomos é desenvolvido em [13] baseado em Extreme Gradient Boosting (XGBoost). Por fim, [14] traz uma ampla pesquisa do uso de lógica *fuzzy* em redes LTE.

Também de maneira vasta, um grande número de trabalhos têm buscado soluções baseadas em inteligência computacional a fim de auxiliar e aperfeiçoar os processos de *handover*, evitando consequências danosas ao sistema de comunicação, como visto em [15]. Já os autores de [16] oferecem uma vasta pesquisa sobre gestão autônoma de *handovers* em redes heterogêneas. Certos trabalhos também utilizam algoritmos clássicos, tais como o K-

Nearest Neighbors (KNN) [17] e redes neurais, para as decisões de *handover* em cenários de redes veiculares. Os autores de [18] implementam uma regressão Bayesiana para melhorias de *handover* em trens de alta velocidade na Coreia do Sul, ao passo que em [19] aplica-se KNN para possíveis decisões em tempo real de *handover* em redes veiculares. Em [20], regressão através de SVM é empregada para estimar o melhor momento de *handover* entre *Access Points* (APs) de uma rede WiFi em ambiente interno. Já em [21], um novo algoritmo é proposto para identificar padrões de mobilidade observando séries temporais de RSRP, visando uma melhor tomada de decisão de *handovers*. Os autores de [22, 23] implementam lógica *fuzzy* e aprendizado por reforço para otimização dos parâmetros tradicionais do *handover*, como TTT e *Hysteresis*. Por sua vez, em [24], novas soluções são propostas para atenuar os efeitos causados por *handover* ping-pong. No contexto do uso de lógica *fuzzy*, os trabalhos [25] e [26] oferecem novas estratégias de otimização para um método tradicional de *handover*. Em [27] e [28] o desenvolvimento de redes neurais para mecanismos de *handover* foi implementado em diferentes estágios. Tais trabalhos demonstram que a integração *handover-redes neurais*, sob diferentes configurações, é capaz de superar os métodos clássicos de *handover* do LTE e do NR.

Os autores de [29] propõem uma Rede Neural Recorrente (RNN, sigla do inglês) para monitorar uma extensa lista de parâmetros de diferentes camadas do protocolo LTE a fim de predizer o tempo de *download* de um arquivo. Em [30], os autores expandiram o estudo para englobar dados disponíveis na rede e melhorar decisões em esquemas de *Modulation Coding Schemes* (MCS) após um *handover*, empregando redes neurais do tipo *Long Short-Term Memory* (LSTM) e *Autoencoders*, estratégias de Aprendizado Profundo (*Deep Learning*), que serão tratadas mais adiante no trabalho. É importante destacar que o processamento e a complexidade de tais abordagens aumentam consideravelmente, visto que essas estratégias demandam o monitoramento de métricas de todas as camadas de comunicação. Já em [31], uma rede LSTM é desenvolvida para monitorar as células próximas para predição de falhas de enlace de rádio (RLF, do inglê *Radio Link Failures*) após processos de *handover*. No entanto, o cenário utilizado nas simulações é muito simplório, contendo apenas duas estações rádio-base (BS, do inglês *Base Station*), além de contar com movimento retilíneo do UE, o que facilita a predição tanto do movimento quanto das condições de canal.

Outra abordagem é discutida em [32], em que se usa LSTM novamente no contexto de gestão de mobilidade. É proposta uma solução para predizer a trajetória dos UEs de modo a otimizar o desempenho dos *handovers*. Um modelo similar para predição de trajetórias é desenvolvido em [33] num cenário de redes veiculares, também baseado em LSTM. Nota-se que tais trabalhos requerem localização de UEs como entradas para seus modelos, o que não é padronizado globalmente para algoritmos de *handover* e podem esbarrar em questões legais, além do aumento da complexidade do processamento dos dados, visto que as amostras não são capturadas em curtos intervalos de tempo. Em [34],

uma classificação é realizada com algoritmos baseados em árvores de decisão e LSTM para indicar a próxima célula a atender certo UE, de acordo com medidas de canal anteriores.

Diante do exposto acima, está claro que o uso de modelos de IA/ML para otimizar decisões de *handover* está sendo amplamente investigado. Entretanto, algumas abordagens tendem a aumentar dramaticamente a complexidade de tal procedimento e demandar maiores custos computacionais. Diferente delas, as soluções apresentadas ao longo deste trabalho se preocupa em não requerer dados extras de outras camadas dos protocolos em uso (LTE, NR), ou de outras origens (como, por exemplo, geolocalização). Nas propostas realizadas aqui, são usados somente os tradicionais sinais de referência para procedimentos de *handover* e re-seleção de células, bem estabelecidos nos padrões vigentes. Com isso, os modelos fazem previsões acerca de características importantes do *handover* em questão, tratadas de maneiras diferentes em cada um dos casos estudados.

1.4 Objetivos

A partir dos experimentos propostos nesta dissertação, tem-se como objetivo a avaliação de técnicas de inteligência artificial que possam fornecer respostas úteis para o aprimoramento do processo de *handover*, sem aumentar drasticamente a complexidade computacional exigida. Além disso, a não utilização de outros tipos de dados como entradas para tais técnicas também é considerada, visando a facilitação do emprego de tais técnicas em redes reais, não tocando em outros tipos de problemas que possam ocorrer ao utilizar outras fontes de dados. Dessa forma, apenas as medidas já conhecidas e amplamente estabelecidas nos padrões 4G e 5G são empregadas.

1.5 Publicações decorrentes da pesquisa

A pesquisa desenvolvida ao longo de todo o curso de Mestrado propiciou a publicação de trabalhos em eventos técnico-científicos e em periódico relacionado ao tema, os quais estão listados a seguir:

- Lima, J., de Medeiros, A., de Aguiar, E., de Sousa Jr., V., Guerra, T. “User-Level Handover Decision Making Based on Machine Learning Approaches”. Journal of Communication and Information Systems, vol. 37, n. 1, 2022. DOI: 10.14209/jcis.2022.11
- Lima, J., de Medeiros, A., de Aguiar, E., Discini, D., Fonseca, L., Guerra, T., Dantas, Y., de Sousa Jr., V. “Sistemas Fuzzy aplicados ao Handover em Redes LTE com Falhas de Cobertura”. Em: Simpósio Brasileiro de Telecomunicações e Processamento de Sinais, v. 1, n. 1, 2019. DOI: 10.14209/sbrt.2019.1570557660
- Miranda, M., Lima, J., Cândido, P., de Medeiros , A. “Predição de Perda de Percurso em Veículos Aéreos Não Tripulados Utilizando Rede Neural Artificial”; Em: Simpósio

Brasileiro de Telecomunicações e Processamento de Sinais, v. 1, n. 1, 2019. DOI: 10.14209/sbrt.2019.1570557378

- Macêdo, R., Miranda, M., Lima, J., de Medeiros, A. "Modelagem inteligente de perda de percurso utilizando transceptores LoRa". Em: Simpósio Brasileiro de Telecomunicações e Processamento de Sinais, v. 1, n. 1, 2019. DOI: 10.14209/sbrt.2019.1570557263

1.6 Organização da dissertação

Esta dissertação está organizada da seguinte maneira: O Capítulo 2 trata de todas as técnicas de Inteligência Artificial, *Machine Learning* e análise de dados empregadas ao longo de todo o trabalho. O Capítulo 3 discute a primeira arquitetura proposta para o emprego dos algoritmos visando a otimização do procedimento de *handover*. Nele, são abordados dois cenários simulados, bem como os resultados obtidos das tarefas de classificação e regressão propostas. Já o Capítulo 4 traz a segunda arquitetura elaborada, baseada em *Deep Learning*, a qual foi empregada em dois novos cenários para testes e validações, um simulado e outro produzido a partir de uma campanha de medições em rede real. Finalmente, as conclusões são expostas no Capítulo 5.

2 TÉCNICAS DE IA UTILIZADAS

Neste Capítulo, são descritas as técnicas de IA/ML empregadas em todo o trabalho, tanto para as tarefas de classificação, bem como para regressão, e também para a previsão de série temporal (*Deep Learning*). Além disso, outras ferramentas para manipulação dos dados, processamento, tratamento e treinamento também são discutidas. Por fim, são tratadas algumas considerações para implementação de tais modelos em redes reais.

2.1 Classificação

A classificação é o processo de predizer classes a partir de certos dados fornecidos como entradas. Tais classes são normalmente chamadas de rótulos ou categorias. Um modelo preditivo de classificação é aquele que aproxima uma função de mapeamento f a partir de variáveis de entrada x para saídas discretas y. Por exemplo, a detecção de um spam em um serviço de e-mail pode ser entendido como um problema de classificação binária visto que apenas duas classes são identificadas (spam ou não-spam). O classificador utiliza uma série de dados de treinamento para entender como as variáveis de entrada se relacionam às classes. Nesse caso, conhecidos e-mails de spam e não-spam devem ser usados como dados de treinamento. E quando o classificador é treinado de forma adequada, ele pode ser utilizado para detectar um e-mail ainda não visto.

Muitos são os algoritmos disponíveis para tarefas de classificação. Dentre eles, podem-se citar as redes neurais artificiais, as árvores de decisão, o sistemas *fuzzy* de classificação, as máquinas de vetores de suporte, entre outros.

Por exemplo, os sistemas *fuzzy* se destacam como ferramentas robustas para modelagem de processos complexos ao passo que preservam uma forma interpretável para a representação do conhecimento. O fato de recorrer a termos linguísticos para expressar os termos envolvidos nas regras *fuzzy* é um fator-chave para associar o formalismo matemático e a inferência lógica à interpretabilidade humana [35]. Adicionalmente, sistemas reais são por natureza ruidosos e qualquer elemento externo pode contribuir com incertezas. Lidar com incertezas é condição *sine qua non* para o emprego de sistemas *fuzzy* em aplicações reais [36].

A lógica *fuzzy* foi inicialmente proposta na teoria dos conjuntos *fuzzy* pelo Prof. Zadeh [37, 38]. Implementada para tratar o conceito de verdade parcial, na qual um valor verdadeiro pode alcançar uma gama de valores entre completamente verdadeiro e completamente falso, com diferentes graus de pertencimento. Tal lógica contrasta com a tradicional lógica *booleana*, em que as variáveis apenas assumem os valores inteiros de zero ou um. Em outras palavras, na teoria de conjuntos *fuzzy*, um elemento pode pertencer a um conjunto com um certo grau de pertencimento, ao invés de pertencer por completo ou não.

Dessa maneira, aproxima-se uma tarefa classificatória, por exemplo, da maneira humana de pensar. Suponha a indicação de como o tempo está em um determinado dia. Não necessariamente, a caracterização tem de ser feita apenas por dados numéricos, como a temperatura em graus Celsius, a umidade relativa do ar, a velocidade do vento, entre outros dados físicos. Pode-se, por exemplo, afirmar que o dia está ensolarado, nublado ou chuvoso, soando assim mais próximo a uma classificação empírica humana, que naturalmente tende a ser confusa, ruidosa ou imprecisa (*fuzzy*).

Os autores de [39] e [40] apontam que estratégias *fuzzy* são capazes de fornecer desempenho equivalente (ou até melhor), consumindo menos recursos computacionais, quando confrontados com ferramentas clássicas de inteligência artificial. Por essa razão, muitas técnicas baseadas em regras *fuzzy* são encontradas em diversas aplicações, como em [41], [26] e [25].

Sendo assim, os métodos de classificação aplicados nesse trabalho são:

- *Autonomous Learning Multimodel System* (ALMMo) [40];
- *Self-Organizing Fuzzy Logic Classifier* (SOFL) [42];
- *Type-2 Fuzzy Logic Classifier* (T2) [41];
- Máquinas de Vetores de Suporte (SVM) [43]
- *Multilayer Perceptron Network* (MLP) [28]

O modelo ALMMo é um algoritmo que pode ser aplicado em diversos tipos de problemas computacionais, como classificação, regressão, identificação, entre outros. O modelo em questão é capaz de extrair as informações necessárias dos dados para constituir seus conjuntos *fuzzy* e suas regras de inferência. Para isso, ele utiliza o conceito de densidade unimodal, o qual proporciona a relação de distâncias entre os dados para a formação de nuvens de dados e suas funções de pertinência. A partir de uma sequência lógica, analisa-se cada amostra para a constituição de pontos focais, ao redor do qual se formam as nuvens de pontos e, então, faz-se um julgamento da relevância daquela formulação. Caso um conjunto *fuzzy* formado seja considerado pouco relevante (segundo métricas internas do método), ele será descartado. O mesmo pode ocorrer com regras de inferência produzidas pelo método. Assim, o mesmo é capaz de classificar as amostras e manter um padrão alto de eficiência. Mais detalhes sobre esse modelo podem ser vistos em [40].

Por sua vez, o modelo SOFL utiliza o conceito de densidade multimodal para organizar sua nuvem de pontos a partir da análise feita em cada amostra. Produz-se uma lista ordenada de amostras, baseada em suas distâncias e suas densidades. Com tal lista, são elaborados protótipos que reúnem os pontos com características comuns e, então,

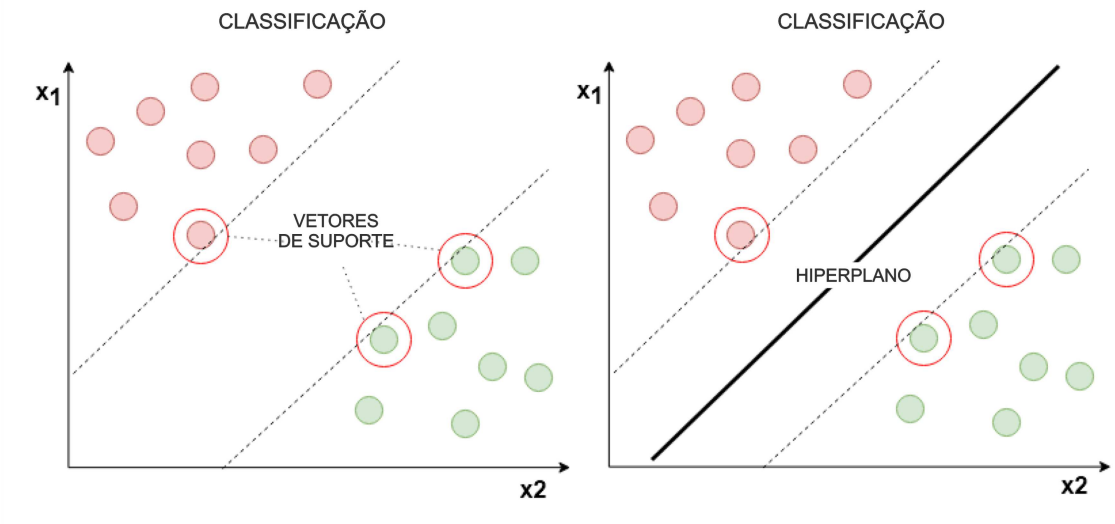
as regras *fuzzy* são depreendidas. A influência de cada protótipo nas nuvens de pontos é definida através do conceito de granularidade, uma métrica que relaciona distâncias entre pontos e é definida pelo usuário. Com isso, o sistema é facilmente adaptável para testes pelo usuário, o qual tem a liberdade de definir o nível de granularidade para seu experimento. Em geral, quanto maior esse parâmetro, maior o refinamento de detalhes, mais protótipos são produzidos e melhor costuma ser o desempenho desse classificador. O modelo pode ser visto em [42].

Mendel em [44] torna o uso de sistemas de inferência *fuzzy* do tipo-2 (T2) muito mais acessível, fornecendo fórmulas matemáticas e fluxogramas para a computação das derivadas necessárias na implementação de métodos de máximo declive para treinamento. O mesmo explica os motivos do cálculo de tais derivadas serem muito mais complexas do que são para um sistema de inferência *fuzzy* do tipo-1. Classificadores baseados em T2 estão sendo cada vez mais usados em diversas aplicações devido a sua capacidade de modelar incertezas do conjunto de dados [36]. O T2 demonstrou melhores habilidades para o trato das incertezas do que sistemas de inferência *fuzzy* do tipo-1 em diferentes aplicações. A visão geral e as comparações discutidas em [45] e [46] ajudaram os pesquisadores e desenvolvedores de T2 a escolherem a estrutura e os algoritmos de redução de tipo mais adequados, do ponto de vista do custo computacional para implementações.

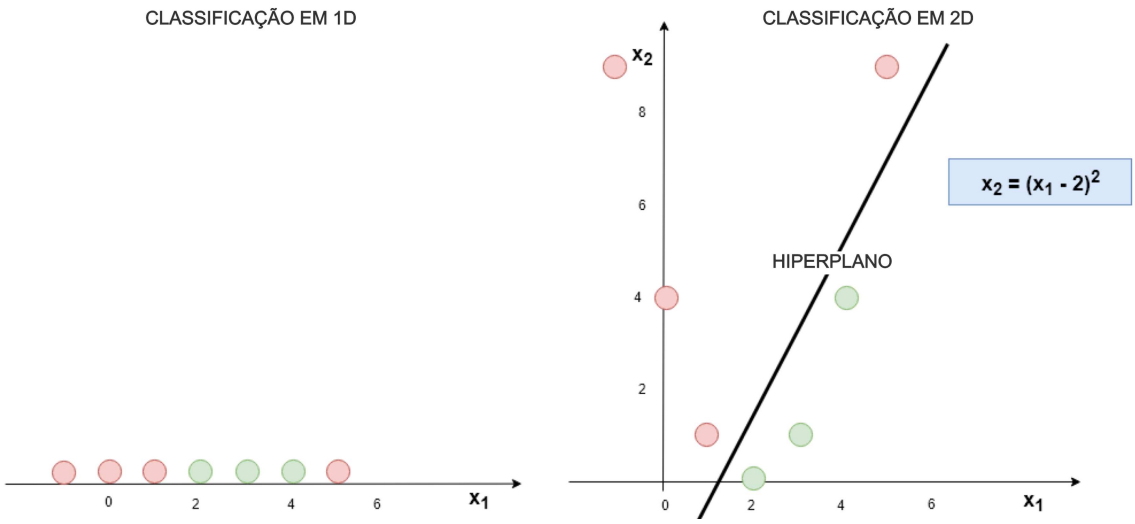
As Máquinas de Vetores de Suporte (*SVM*, sigla do inglês) funcionam como um mecanismo robusto para classificação em inteligência computacional. Essa técnica se baseia no entendimento de que certas amostras chamadas de vetores de suporte são mais importantes que outras. Suponha que temos um problema de classificação binária linearmente separável como mostrado na Figura 2. As amostras mais relevantes para este método são aquelas mais difíceis de classificar, ou seja, aquelas que se encontram o mais perto possível da nuvem de pontos da outra classe, como indicado na imagem. Com base nessas amostras, o método traça tais vetores de suporte e, com isso, um hiperplano que separa as categorias, compondo uma superfície de decisão. O hiperplano será posicionado de modo a maximizar a margem de separação entre tais exemplos relevantes das diferentes classes.

Para problemas não linearmente separáveis, o SVM usa o artifício das funções de Kernel. Essa é uma função que modifica o espaço amostral para mapear os dados em um novo espaço com um número maior de dimensões, usando produtos internos. Muito frequentemente, tais problemas podem se tornar facilmente separáveis em um espaço com dimensões mais elevadas, como mostrado na Figura 3. Nesse exemplo, é possível verificar que um problema de classificação em uma dimensão é resolvido após as amostras serem mapeadas num espaço de duas dimensões, através de um Kernel polinomial de grau 2.

Por outro lado, *MLP* se baseia na teoria de Redes Neurais Artificiais (RNA), uma outra poderosa estratégia consolidada na literatura. As RNAs são modelos computacionais



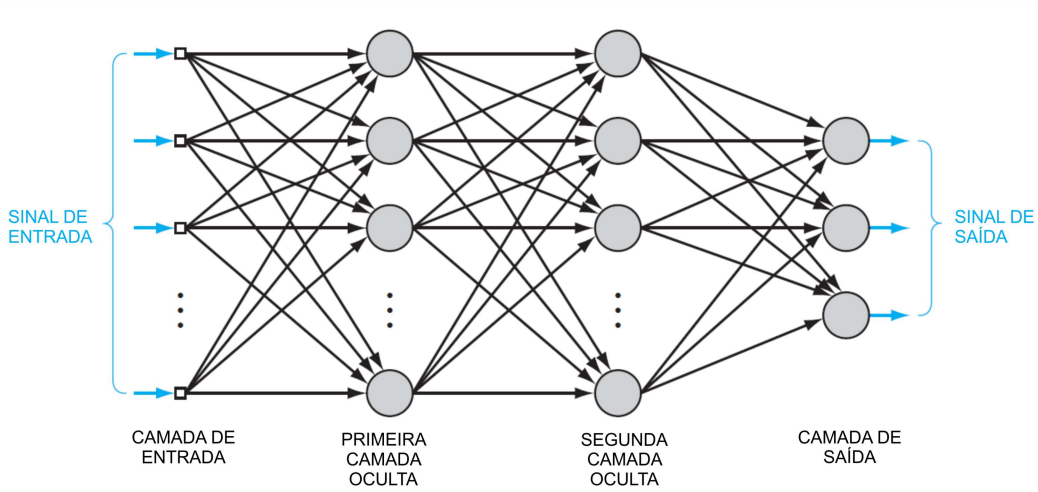
– Fig. 2: Traçando o hiperplano com vetores de suporte [27].



– Fig. 3: Usando o kernel para facilitar a classificação [27].

que tentam reproduzir a forma com a qual o cérebro realiza uma tarefa. Em relação às suas estruturas, essas redes são constituídas por unidades básicas de processamento conhecidas como neurônio computacional. Esses neurônios estão organizados em camadas, sendo elas: a camada de entrada, as camadas ocultas e a camada de saída. Como visto na Figura 4, um neurônio não tem conexões com os outros neurônios de sua própria camada, mas eles são totalmente conectados a todos os outros das camadas anterior e posterior a sua. Essas conexões são conhecidas como sinapses, que possuem pesos para especificar o grau de proximidade entre um neurônio e o outro [47].

Na fase de treinamento da rede, o objetivo é encontrar o melhor conjunto de pesos sinápticos que minimizam a função de perda, melhorando assim a capacidade de predição do modelo. Neste trabalho, utilizou-se o modelo *MLP*, um tipo de rede neural *feed-forward*



– Fig. 4: Esquemático de uma RNA [47].

multicamadas, que emprega a estratégia de *backpropagation* para a fase de treinamento. Esse método clássico tem inúmeras aplicações em muitas áreas, exemplificadas por [48, 49].

2.2 Regressão

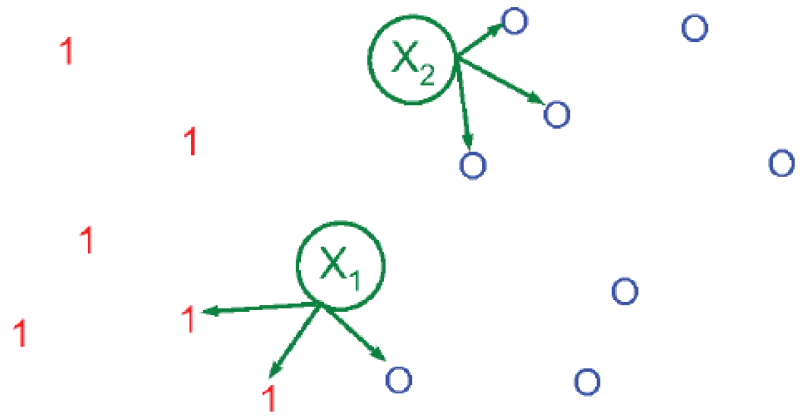
Nesta Seção, são apresentados os seis métodos de regressão empregados ao longo deste trabalho. Os métodos avaliados são:

- *K-Nearest Neighbors (KNN)*
- *Random Forest (RF)*
- *Gradient Boosting Machine* (GBM)
- *Extreme Grandient Boosting* (XGBoost)
- *LightGBM*
- *MLP*

Visto que o modelo MLP já foi abordado na Seção anterior, apenas os demais modelos serão aqui tratados.

O *KNN* é um modelo de aprendizado de máquina popular em virtude de seu funcionamento ser descomplicado. Sua implementação é relativamente simples devido ao fato de conter poucos parâmetros, se comparado a outras arquiteturas mais complexas [17].

Em relação ao seu funcionamento, primeiramente é necessário que o algoritmo detenha o conjunto de dados de treino $X_i Y_i$. Dessa forma, quando se insere uma amostra



– Fig: 5: Uma classificação KNN com $K = 3$ [50].

inédita de entrada X_j com valor de saída Y_j desconhecido, realiza-se um cálculo da distância entre a entrada X_j e todas as outras entradas X_i pertencentes ao conjunto de treino. Após isso, seleciona-se o subconjunto X_kY_k pertencente a X_iY_i que contém K amostras com a menor distância em relação à entrada X_j inserida. Dessa maneira, num problema de regressão, o valor da saída Y_j será a média das K amostras mais próximas determinadas pelo método. Já num problema de classificação, a saída será a classe predominante entre essas K amostras, conforme exemplificado na Figura 5.

Já a técnica de *Random Forest* constrói várias árvores de decisão de forma independente, adicionando recursos que geram aleatoriedade no momento de treiná-las [11]. Primeiro, é preciso saber que a árvore de decisão é um regressor que contêm três elementos: os nós, os ramos e as folhas. Sua construção se inicia pelo nó, um objeto que contém um teste sobre algum dos atributos de entrada X_i do conjunto de treino. O teste é aplicado e o nó raiz se divide, dando origem a um nó filho para cada resultado possível do teste, sendo que os ramos são responsáveis pela interligação entre os nós pai e filho. Esse processo é efetuado de forma recursiva até que a profundidade máxima da árvore seja atingida. Por fim, as folhas são os terminais da árvore. Elas agrupam os valores de Y_i do conjunto de treino que satisfazem a sequência de testes que a produziram. A saída produzida pelo RF será, num problema de classificação, a classe mais prevista pelas árvores de decisão, enquanto num problema de regressão, a média de suas predições.

Mas apesar de poderosas, é importante destacar que as árvores de decisão são bastante sensíveis aos dados de treinamento, de forma que qualquer alteração no conjunto de treino pode fazer com que uma árvore seja totalmente diferente da outra. Com isso, o algoritmo as treina com certas alterações nos dados de treinamento. Para tanto, o conjunto de treinamento X_iY_i dá origem a vários subconjuntos, que são produzidos via reamostragem com substituição [11]. Além disso, as árvores não terão acesso a todos os atributos do conjunto de treino, pois caso tivessem, os mesmos atributos mais homogêneos

sempre seriam escolhidos.

Por outro lado, o conceito geral do algoritmo de *Gradient Boosting Machine* é fazer uso de vários modelos simples de predição, conhecidos como aprendizagem fraca, os quais são combinados sequencialmente para formar um modelo composto, conhecido como aprendizagem forte que, por sua vez, tem uma poderosa capacidade de predição. A aprendizagem fraca utilizada é a árvore de decisão e a aprendizagem forte será o resultado da combinação de todas as árvores usadas sequencialmente. Para tal tarefa, é empregado o modelo aditivo, que tem como objetivo realizar a descida de gradiente. Assim ele adiciona, em cada iteração, uma nova árvore de decisão que contribui para minimizar o erro [51].

De forma mais detalhada, o algoritmo funciona da seguinte forma: inicia-se com uma previsão F_0, que é a mediana dos valores Y_i de treinamento. Após isso, é calculado o residual R_0 entre os valores previstos F_0 e os valores medidos Y_i. O conjunto de treinamento X_iY_i é reamostrado, dando origem a vários subconjuntos X_jY_j, sendo que os resíduos referentes a Y_j são R_{0j}. Assim começa o modelo aditivo, que acrescenta uma árvore de decisão treinada com um subconjunto X_jR_{0j} e o valor previsto pela arvore é multiplicado por uma espécie de peso, chamado de taxa de aprendizagem. Então, o valor resultante é somado ao valor previsto inicial F_0, resultando no novo valor de previsão F_1. Em seguida, o valor residual R_1 é calculado pela diferença entre F_1 e Y_i, e todo esse processo se repete iterativamente até que alguma condição de parada seja atingida [51].

Conhecido popularmente como *XGBoost*, o modelo *eXtreme Gradient Boosting* é uma melhoria do *GBM* pois, além de conter recursos adicionais na parametrização, também tem melhor desempenho e velocidade de execução, o que é muito vantajoso para treinamento de grandes volumes de dados. Em virtude de seu bom desempenho, ele se estalebeceu na área de ML e vem sendo usado tanto por grandes corporações quanto em estudos recentes e relevantes que integram comunicações e aprendizado de máquina [13]. Dentre as melhorias que ele oferece, se destacam a variedade de recursos, visto que os usuários têm à disposição diversas opções de parametrização e critérios de avaliação do algoritmo; e o processamento paralelo, em que, durante a construção do modelo na fase de treinamento, a árvore de decisão é dividida em galhos. Cada um deles é construído e treinado separadamente em um núcleo do processador, diminuindo assim o tempo para treinar grandes volumes de dados.

O *LightGBM* é um modelo criado pela empresa *Microsoft* e tem como principal objetivo treinar grandes volumes de dados de forma rápida e precisa. Apesar de ser baseado no *GBM*, esse algoritmo tem diferenças na fase de construção das árvores. Ele busca se expandir na folha que teve melhor ajuste, ou seja, a que apresentou a maior redução da função de perda. Sendo assim, no *LightGBM* as árvores crescem de forma vertical, enquanto nos outros modelos elas crescem de forma horizontal [12].

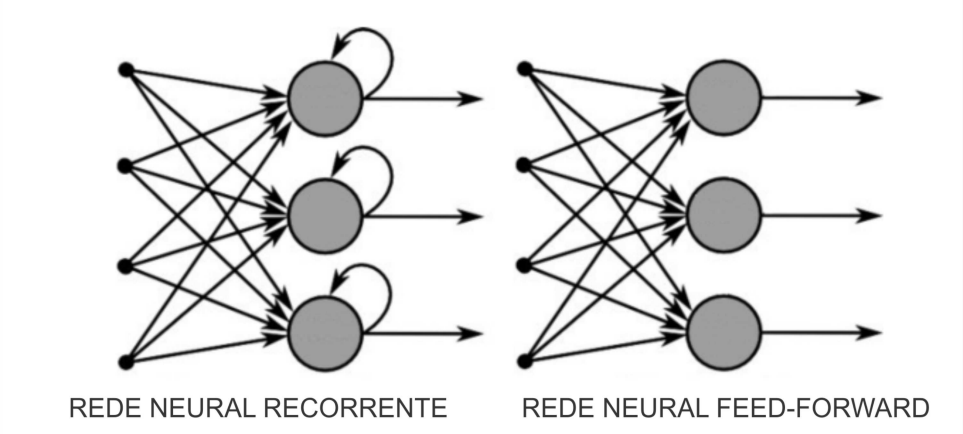
Apesar do algoritmo apresentar certas melhorias, uma desvantagem é que, para

pequenas quantidades de dados, o modelo pode se sobreajustar demais ao conjunto de treino, processo conhecido como *overfitting*. Por isso, para esses casos é recomendado limitar a profundidade máxima da árvore.

2.3 Rede neural recorrente

Redes neurais recorrentes (RNN, sigla do inglês) são uma classe de redes neurais que permitem que saídas anteriores possam ser usadas como entradas, ao mesmo tempo que possuem camadas ocultas. Algumas vantagens de uma arquitetura típica de RNN são ter seus pesos compartilhados ao longo do tempo, além do fato de suas computações considerarem informações históricas, isto é, informações que já foram apresentadas anteriormente em alguma outra etapa do seu desenvolvimento. Modelos RNN são amplamente utilizados em áreas como o reconhecimento de fala e o processamento de linguagem natural, além de uma grande empregabilidade na análise de séries temporais, seja no campo das finanças ou nas engenharias.

De maneira simplificada, as redes neurais recorrentes se diferem das tradicionais pela característica da persistência da informação. Em outras palavras, na sua arquitetura há a presença de *loops* que fazem com que uma parte da informação apresentada na unidade neuronal de processamento seja retida. Ou seja, é como se o seu neurônio possuísse diversas cópias de si mesmo, cada um encaminhando a mensagem ao seu sucessor, como exemplificado pela Figura 6. Naturalmente, esse arranjo em cadeia a predispõe a se associar com problemas de listas, sequências ou eventos [52].



– Fig. 6: Representação típica de RNN, em contraste com uma rede neural convencional [52].

2.4 Outras ferramentas utilizadas e considerações para uso das técnicas em redes reais

2.4.1 Validação cruzada para treino e teste dos algoritmos

Em busca de maior robustez estatística, utilizou-se a validação cruzada com *Stratified k-Fold* (com $k = 5$) [47]. Nesse método, os dados são divididos em k porções, de modo a separar aproximadamente de forma balanceada o número de amostras de cada classe em cada uma das k porções. São realizadas, então, k rodadas de treinamento e validação, de forma que, em cada rodada, $k - 1$ porções são utilizadas para treino, e a porção restante usada para a validação. O resultado final será, portanto, a média dos k resultados obtidos nesse processo, que está ilustrado na Figura 7. Uma notável vantagem de se utilizar a validação cruzada k-Fold é o fato de se utilizar todos os dados disponíveis tanto para a treinamento quanto para teste.

– Fig. 7: Validação cruzada k-Fold com $k = 4$ [47].

Ademais, os testes foram realizados 33 vezes, sendo novamente o resultado final a média obtida nesses experimentos.

2.4.2 Otimização via pesquisa aleatória

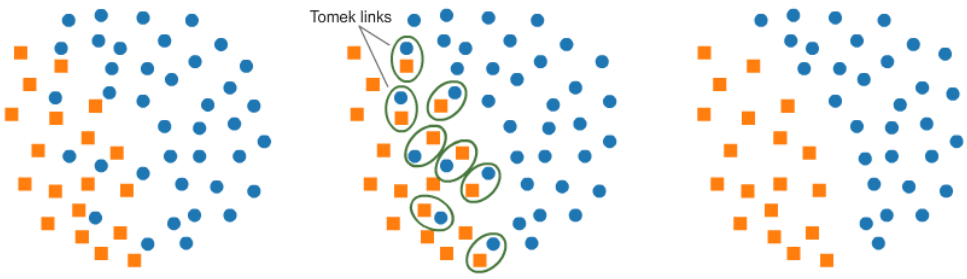
Os hiperparâmetros são aqueles cujos valores são empregados para aprimorar o processo de aprendizagem dos modelos. Para encontrá-los em cada aplicação desenvolvida, utilizou-se a técnica conhecida como pesquisa aleatória. Ela é um método de otimização que consiste em realizar numerosas combinações aleatórias dos hiperparâmetros a fim de encontrar uma solução ótima [53]. Para isso, é necessário definir o intervalo de valores possíveis para cada um dos hiperparâmetros, o número máximo de iterações que a pesquisa deve realizar e também a função de perda que será minimizada.

2.4.3 Técnicas para super e sobreamostragem

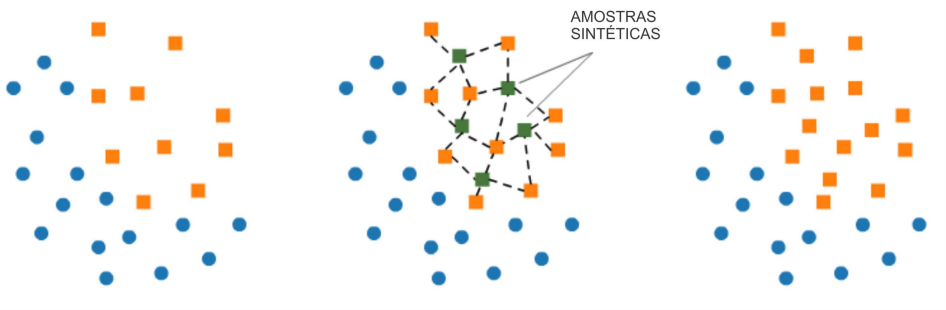
É válido destacar que o disparo de *handover* pode ser considerado um evento raro entre todas as amostras de potência medidas, isto é, a exata amostra de tempo em que um *handover* é disparado é apenas um ponto em uma longa série temporal. Por exemplo, mais de 1 milhão de amostras de sinais de referência foram coletados para avaliação do modelo neste trabalho, o que será abordado *a posteriori*. No entanto, apenas algumas centenas acarretaram em disparos de *handover*, levando a uma base de dados desbalanceada. Esse é um problema comum em aplicações de IA/ML, em que há uma disparidade muito grande entre o número de amostras anotados em cada classe. Isso pode afetar a performance do classificador, como representado na Figura 8.

– Fig. 8: Técnicas para tratamento de bases de dados desbalanceadas [54].

Nesse contexto, foram utilizadas duas técnicas clássicas para lidar com essa situação: primeiramente, aplicou-se uma técnica de subamostragem (*Tomek Links*) e, então, uma outra de superamostragem (*SMOTE*). A primeira é responsável por remover aquelas amostras consideradas desnecessárias para a tarefa de classificação, as quais possuem pequenas distâncias para pontos da classe de menor quantidade. Isso tem por objetivo balancear a base de dados, enquanto colabora com a tarefa de classificação, facilitando a separação das categorias, como visto na Figura 9. Já a segunda técnica é utilizada para popular a base de dados com novas amostras produzidas, possuindo grande semelhança com aquelas da classe em defasagem, balanceando ainda mais a base de dados, o que pode ser observado na Figura 10.



– Fig. 9: Tomek Links para subamostragem [54].



– Fig. 10: SMOTE para superamostragem [54].

2.4.4 Implementação de modelos em rede real

Há algumas considerações sobre como os modelos de ML poderem ser implementados em rede real. Primeiramente, é necessária uma etapa inicial, em que a rede atua sem a ação dos modelos para coletar e armazenar dados. Nas propostas apresentadas nesse trabalho, as medidas de RSRP e RSRQ são coletadas nessa etapa, junto com algumas informações sobre os *downloads*, suas finalizações, os tempos requeridos, bem como os *handovers* realizados. Esses dados, por sua vez, servirão de entradas para os modelos a serem treinados. Isso também indica a importância de se estimar o tempo de processamento demandado, visto que os modelos devem ser atualizados o mais rápido possível, sem danos à rede. Após essa etapa, os modelos já treinados estão disponíveis para atuar em tomadas de decisão de *handover* junto às BSs (*Base Stations*). Ressalta-se que, se novas mudanças forem feitas na rede, como por exemplo a introdução de uma nova BS, seria necessário o retreinamento dos modelos para levarem em conta a nova topologia do cenário em questão [28].

2.5 Conclusões

Este Capítulo apresentou as técnicas de IA/ML que serão utilizadas ao longo de todo o trabalho. Além disso, outras estratégias para manipulação dos dados, treinamento e teste dos modelos, validação cruzada, bem como métodos para lidar com o desbalanceamento entre classes dos dados em questão foram abordadas. Finalmente, o Capítulo trouxe breves considerações sobre a aplicabilidade das arquiteturas propostas em uma rede real.

Na sequência, a primeira arquitetura proposta para a integração de ML em procedimentos de *handover* é apresentada, com os cenários simulados e os resultados obtidos pelos modelos avaliados.

3 ESTUDOS COM A PRIMEIRA ARQUITETURA PROPOSTA

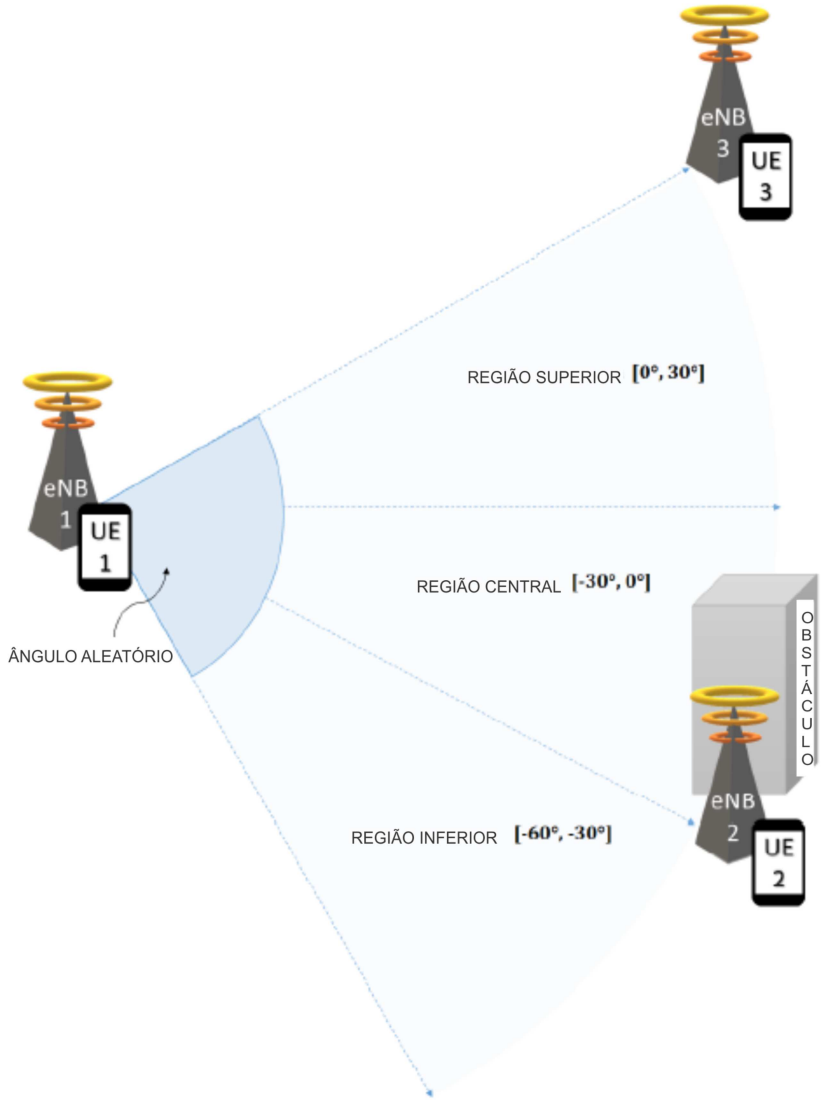
Neste Capítulo, descreve-se a primeira arquitetura proposta para possíveis aplicações de IA/ML no contexto da otimização de *handovers* em redes móveis. Essa arquitetura é testada em dados obtidos por uma primeira campanha de simulações, na qual dois cenários são contemplados. Todos os seus parâmetros, bem como os resultados e suas discussões são abordados aqui. Além disso, todos os testes foram realizados com códigos em Python e Matlab em um computador com processador i7-6700HQ (2.6 GHz).

Destaca-se que nenhuma medida de localização ou mobilidade foi utilizada como entrada para esses modelos, o que aumentaria a complexidade dos algoritmos. Dessa maneira, nenhum dado de direção, trajetória, ângulos ou distâncias foi considerado, uma observação importante, visto que os métodos atuais de *handover* empregam apenas comparações básicas dos sinais de referência já tratados *a priori*.

3.1 Cenários com falha de cobertura

Nessa Seção, é apresentado o processo de simulação e aquisição de dados para a primeira campanha de simulações. O ambiente de simulação é baseado em [55], utilizando-se a versão 3.22 do simulador de redes *ns-3* [56]. A configuração geral dessa campanha pode ser observada na Figura 11 e dois cenários foram desenvolvidos a partir dela. O primeiro é assentado no modelo de propagação Okumura-Hata, considerando apenas a perda de percurso como efeito do canal, o que torna essa configuração determinística. Por outro lado, o segundo considera adicionalmente o efeito *shadowing*, isto é, acrescenta-se um desvanecimento de larga escala aleatório, que tem a intenção de modelar perturbações no sinal causadas por colinas, vegetações, construções, entre outros. Além disso, são incorporados obstáculos, acrescentando falhas de cobertura para avaliação em ambos os cenários. Comparando os *Radio Environment Maps* (REMs) da torre eNB (*Evolved NodeB*) 2 de ambos cenários (Figuras 12 e 13), é possível verificar que as medidas de *Signal to Interference-Plus-Noise Ratio* (SINR) indicam uma aleatoriedade causada pelo *shadowing* ao modelo de propagação.

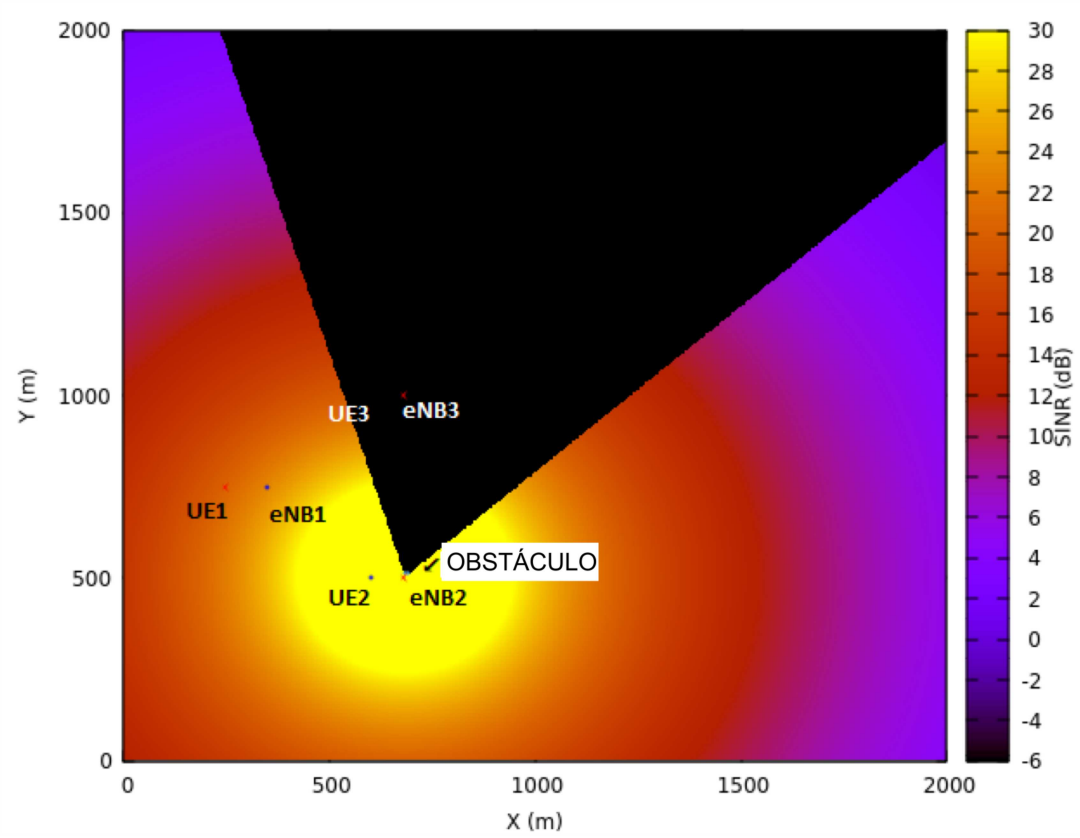
Ambos os cenários contam com três estações base (3 eNBs), três usuários da rede (3 UEs) e um obstáculo próximo à torre eNB 2. O UE 1 inicia simultaneamente um *download* e o movimento em linha reta com velocidade constante de 60 km/h, com um ângulo aleatório entre -60° e 30°. À medida que se locomove, esse usuário abandona a área de cobertura da eNB 1 e entra na área das torres eNBs 2 e 3, requisitando o *handover*. É importante destacar que a região central (-30° a 0°) é o local das maiores divergências, considerando que as regiões superior e inferior são fortemente influenciadas pelas torres próximas. Cada nova semente de simulação representa um novo ângulo aleatório para simulação. Para cada cenário, foram produzidas cerca de 1200 execuções. Os níveis



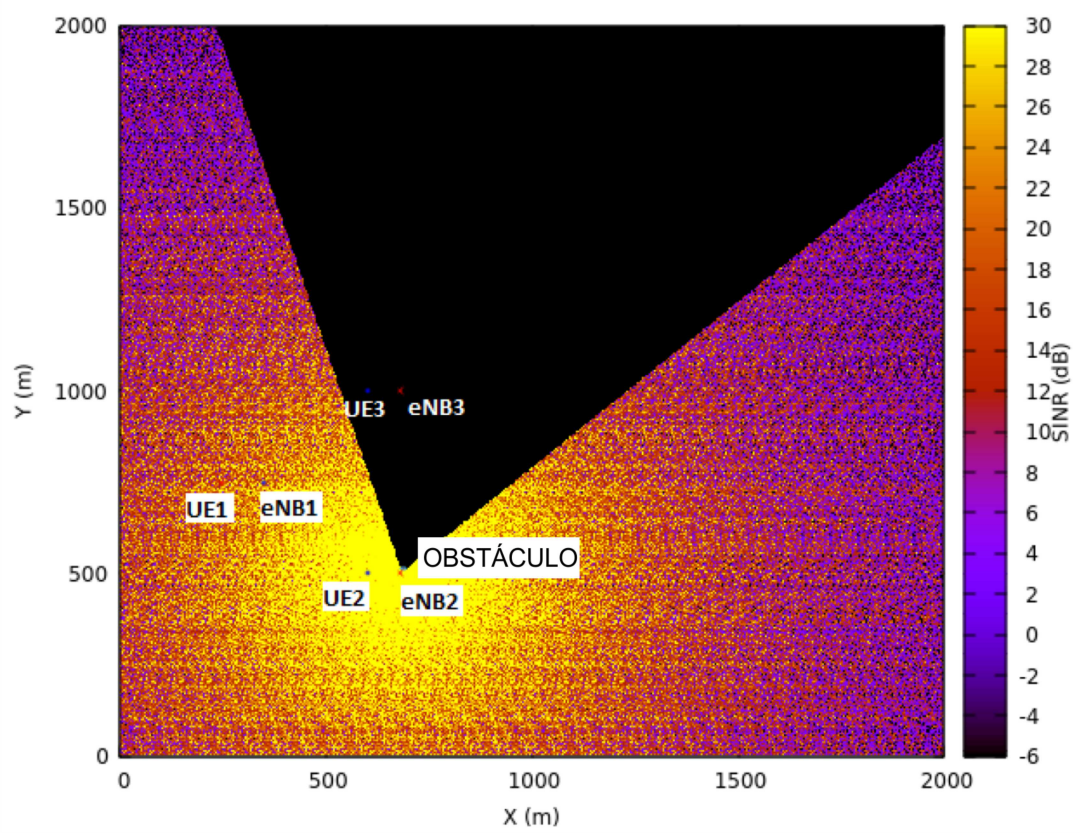
– Fig. 11: A configuração da simulação [27].

de RSRP e RSRQ foram capturados a cada 200 ms, sendo essas as medidas obtidas para alimentar a base de dados de entrada dos modelos. O *download* utiliza o protocolo *Transmission Control Protocol* (TCP), escolhido pela popularidade desse tipo de tráfego, e possui o tamanho de 15 MB. O arquivo TCP é segmentado em partes de 1448 bytes e o tamanho máximo do *buffer* de transmissão é 60 KB. Após o *download*, não há mais troca de dados entre as partes. Finalmente, a simulação tem tempo de execução de 100 segundos.

Para uma visualização mais ampla, os parâmetros de simulação são apresentados na Tabela 2.



– Fig. 12: REM para cenário 1 (sem *shadowing*).



– Fig. 13: REM para cenário 2 (com *shadowing*).

Tabela 2 – Parâmetros de simulação baseados em [48].

Parâmetro	**Valor**
Banda do sistema	5 MHz
Distância interna	500 m
Adapt. Enlace, Model. erros	MiErrorModel
Área de simulação	2000m x 2000m
Número de eNBs	3
Pot. de trans. das eNBs	46 dBm
Número de UEs	3
Velocidade do UE 1	60 km/h
Perda de percurso	Okumura-Hata
Shadowing cenário 1	Sem shadowing
Shadowing cenário 2	LogNormal ($\sigma = 8dB$)
Altura da antena eNB	30 m
Altura do obstáculo	35 m
Tráfego	Bulk file transfer
Tamanho do arquivo	15 MB
Tempo de simulação	100 seconds

– Fig. 14: Primeira arquitetura a ser implementada. Elaborada pelo autor.

3.2 Primeira implementação proposta

Embora simples, a implementação dos modelos de IA/ML que utilizam dados dessa campanha de simulações pretende ser bastante poderosa, visto que ela poderá substituir por completo o emprego dos algoritmos tradicionais de *handover*. A proposta aqui não é apenas ser um mecanismo auxiliar, mas sim fornecer saídas que levem a tomadas de decisão mais precisas, baseada nos dados da própria rede, como podemos ver no diagrama da Figura 14.

A partir dos dados simulados, duas tarefas de ML serão contempladas nesse estudo. A classificação é proposta para a determinação de qual o melhor alvo de *handover* para o UE que se movimenta, afastando-se da eNB que o serve. A outra tarefa será uma regressão sobre o tempo necessário para o download que o UE realiza enquanto se desloca

no ambiente.

Com a finalidade de desenvolver uma estratégia para determinar o melhor alvo de *handover*, três regras foram estabelecidas. A primeira indica que o melhor alvo será aquele que permite o *download* ser completado. A segunda regra determina que o melhor alvo será aquele que proporciona o menor tempo de *download*, caso as duas opções sejam capazes de completar o *download*. Finalmente, a terceira regra diz que a melhor opção para *handover* será aquela que permite o maior número de bytes recebidos, caso em ambas opções o *download* não seja completado. Baseado nesse entendimento, certamente é possível inferir que o objetivo da simulação vai de encontro a um problema do tipo classificação, assentado no contexto do aprendizado de máquina.

Os algoritmos usados nessa tarefa, descritos no Capítulo 2, foram avaliados com os seguintes parâmetros:

- Como ALMMo extrai de maneira autônoma todas os parâmetros diretamente dos dados apresentados, esse modelo não possui parâmetros para refinamento;
- O modelo SOFL apresentou melhor desempenho com uso da distância *Mahalanobis*, além de *Granularity level* em 2.9;
- Para o modelo T2FLS, o parâmetro de aprendizagem foi $\alpha = 0.01$, o de tolerância $\epsilon = 10^{-8}$, além de $\beta_1 = 0.9$ e $\beta_2 = 0.999$;
- No SVM, foi empregado Kernel linear e o parâmetro de penalidade de 10 e 100, nos cenários 1 e 2, respectivamente;
- A rede MLP foi utilizada com *solver LBFGS*, contendo 6 neurônios na camada oculta.

Para a regressão, cada cenário necessitou de uma variável diferente. No primeiro, a previsão é feita para a duração do tempo de *download*, visto que a grande maioria dos *downloads* foram completados dentro do tempo de simulação. No entanto, para o segundo cenário, tido como mais desafiador e com propagação mais adversa, a previsão foi feita para o percentual de *download* completado. A maioria dos *downloads* simulados não se completaram, produzindo uma concentração de pontos anotados como 100 s.

Nesse sentido, os algoritmos empregados para essa tarefa foram testados sob as seguintes condições:

- Agora, o modelo MLP conta com 22 e 4 neurônios nas camadas ocultas, funções de ativação tangente hiperbólica e logística, além de *solver LBFGS*, nos cenários 1 e 2, respectivamente;

- KNN foi utilizado considerando 4 e 6 vizinhos mais próximos nos cenários 1 e 2, respectivamente;
- RF foi desenvolvido com 94 e 106 árvores na floresta nos cenários 1 e 2, respectivamente;
- Já o modelo GBM utilizou 84 e 120 árvores no seu conjunto, em cada um dos casos;
- XGBoost aplicou 174 e 120 estimadores em cada cenário;
- Finalmente, LightGBM empregou 148 e 139 estimadores no seu conjunto nos cenários 1 e 2, respectivamente.

Todos os detalhes e hiperparâmetros relacionados aos modelos podem ser acessados livremente em um repositório desenvolvido justamente para esse fim, em [57]. Nele, há uma coleção de *Jupyter Notebooks*, e todos os códigos desenvolvidos para treinamento, teste e avaliações estão disponíveis.

3.3 Resultados da classificação

Para a classificação do melhor alvo de *handover*, três métricas foram utilizadas na comparação: a *Acurácia* média das predições, ou seja, o percentual das predições corretas do classificador, de acordo com Eq. (3.1); o *Desvio Padrão* associado, expresso em Eq. (3.2) e o tempo de processamento. O Desvio Padrão é avaliado por expressar a estabilidade de cada modelo, visto que cada um deles foi avaliado 33 vezes, com diferentes sementes, sendo x_i o resultado obtido no i-ésimo experimento, $\overline{x}$ a média dos resultados e n o número de experimentos.

$$Acurácia = \frac{Predições\ corretas}{Total\ de\ predições} \tag{3.1}$$

$$Desv.\ Padrão = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \overline{x})}{n}} \tag{3.2}$$

Os resultados iniciais são exibidos nas Tabelas 3 e 4, para os cenários 1 e 2, respectivamente.

Em seguida, aplica-se o Teste-t de duas amostras [58] para verificar a validade estatística dos dados obtidos com a comparação. Esse teste é representado pela Eq. (3.3) [58]:

$$t = \frac{\bar{G}_1 - \bar{G}_2}{\sqrt{\frac{{s_{G_1}}^2}{l} + \frac{{s_{G_2}}^2}{h}}}, \tag{3.3}$$

onde $\bar{G}_1$ e $\bar{G}_2$ são as médias, s_{G_1} e s_{G_2} os desvios padrão e h e l são o tamanho das amostras G_1 e G_2, respectivamente.

Tabela 3 – Resultados iniciais para o cenário 1.

Modelo	**Acurácia (± Desv. Padrão) (%)**	**Tempo (s)**
MLP	99.72 (± 0.10)	5.25
SVM	99.74 (± 0.08)	0.54
SOFL	99.11 (± 0.22)	12.51
T2FLS	98.94 (± 0.68)	5845.78
ALMMo	99.61 (± 0.11)	257.47

Tabela 4 – Resultados iniciais para o cenário 2.

Modelo	**Acurácia (± Desv. Padrão) (%)**	**Tempo (s)**
MLP	86.22 (± 0.96)	15.94
SVM	86.34 (± 0.33)	21.26
SOFL	85.32 (± 0.71)	14.60
T2FLS	72.40 (± 0.98)	7672.62
ALMMo	68.06 (± 1.16)	357.74

Também define-se o Grau de Liberdade (DF) como:

$$DF = l + h - 2 \tag{3.4}$$

e, como DF é alto ($DF = 328$), não é necessário verificar a normalidade da distribuição do erro [58].

Além da avaliação de t, é importante também inferir as hipóteses $H_0 : \bar{G}_1 = \bar{G}_2$ e $H_1 = \bar{G}_1 \neq \bar{G}_2$, em que H_0 é a hipótese nula e indica que ambos os métodos $\bar{G}_1$ e $\bar{G}_2$ apresentam a mesma acurácia; e H_1 é a hipótese alternativa, a qual indica que os níveis de acurácia são distintos.

Dado um nível de significância α_t, o p-valor, que é calculado a partir de t, representa o menor valor possível para rejeitar H_0. Valores menores que α_t indicam que H_0 não é verdadeira [58]. Aqui, é assumido $\alpha_t = 0.05$. Além disso, os limites inferior e superior de cada comparação são calculados para indicar o intervalo de confiança obtido na execução do teste.

Nessa fase, o Teste-t tem a intenção de verificar a comparação de desempenhos dos modelos avaliados para a obtenção da maior acurácia na classificação do melhor alvo de *handover* para o UE 1. Dessa forma, a Tabela 5 apresenta os resultados do teste para o cenário 1 (sem o efeito de sombreamento) e a Tabela 6, para o cenário 2 (com sombreamento).

Baseado nas Tabelas 3 e 5, os resultados indicam que todos os métodos demonstram ótimo desempenho quando não há a presença do efeito de sombreamento atrapalhando as predições. No entanto, *SVM* e *MLP* possuem os melhores resultados e um tempo

Tabela 5 – Resultados do Teste-t para o cenário 1.

G_1	G_2	p-**valor**	**Limite inf.**	**Limite sup.**	H_0
SVM	ALM	3.56E-07	0.0009	0.0018	1
SVM	MLP	0.425	-2.70E-4	6.34E-4	0
SVM	SOFL	4.66E-19	0.0055	0.0071	1
SVM	T2FL	1.17E-07	0.0056	0.0105	1
MLP	ALM	2.13E-05	0.0007	0.0017	1
MLP	SOFL	5.12E-19	0.0053	0.007	1
MLP	T2FL	1.83E-07	0.0054	0.0103	1
SOFL	ALM	9.99E-16	-0.0058	-0.0041	1
SOFL	T2FL	0.1803	-0.0008	0.0042	0
ALM	T2FL	3.39E-06	0.0042	0.0091	1

Tabela 6 – Resultados do Teste-t para o cenário 2.

G_1	G_2	p-**valor**	**Limite inf.**	**Limite sup.**	H_0
SVM	ALM	1.26E-44	0.1786	0.1871	1
SVM	MLP	0.5019	-0.0024	0.0048	0
SVM	SOFL	1.66E-9	0.0075	0.013	1
SVM	T2FL	2.47E-44	0.1358	0.1431	1
MLP	ALM	1.86E-60	0.1764	0.1869	1
MLP	SOFL	5.76E-5	0.0049	0.0132	1
MLP	T2FL	7.31E-57	0.1334	0.143	1
SOFL	ALM	6.14E-55	0.1679	0.1773	1
SOFL	T2FL	1.78E-54	0.125	0.1334	1
ALM	T2FL	2.47E-24	-0.0487	-0.0381	1

de processamento reduzido. Observa-se que o p-valor é maior que α_t apenas quando são comparados *SVM* e *MLP*, e *SOFL* e *T2FL*, o que indica a validade da hipótese nula ($H_0 = 0$). Assim, o Teste-t demonstra que não há diferença estatística significativa entre tais algoritmos nesses casos, confirmando que os melhores modelos para a tarefa de classificação nesse cenário são *SVM* e *MLP*. Além disso, o modelo baseado em lógica *fuzzy ALMMo* também apresenta uma excelente acurácia, mas não fornece um tempo reduzido para processamento.

Ademais, ao observar os resultados para o cenário 2 nas Tabelas 4 e 6, no qual o há o sombreamento, alguns algoritmos conseguem ainda atingir uma acurácia razoável, especialmente *SVM* e *MLP*, embora haja uma queda considerável (cerca de 13%) em relação ao cenário anterior. Novamente, esses dois modelos superam os outros na comparação, e não apresentam diferença estatística entre si, o que é confirmado pelo Teste-t. No entanto, vale destacar nesse caso que o modelo baseado em lógica *fuzzy SOFL* foi capaz de atingir uma acurácia competitiva, quando comparado com os demais, demandando o menor tempo

Tabela 7 – Resultados da regressão para o cenário 1.

Modelo	**EMA (± Desv. Padrão) (%)**	**Tempo (s)**
MLP	0.27163 (± 0.00800)	21.85
KNN	0.12791 (± 0.00490)	1.24
RF	0.11987 (± 0.00524	49.34
GBM	0.12452 (± 0.00727)	32.98
XGBoost	0.12343 (± 0.00714)	25.67
LightGBM	0.11553 (± 0.00317)	14.81

Tabela 8 – Resultados da regressão para o cenário 2.

Modelo	**EMA (± Desv. Padrão) (%)**	**Tempo (s)**
MLP	6.32415 (± 0.07108)	8.98
KNN	6.00046 (± 0.07525)	1.68
RF	5.99726 (± 0.05373)	53.05
GBM	6.19593 (± 0.07841)	34.15
XGBoost	5.88485 (± 0.10150)	17.53
LightGBM	5.08845 (± 0.04832)	13.76

de processamento. Tal feito é bastante significativo para o contexto de lógica *fuzzy*.

3.4 Resultados da regressão

Já no estudo da regressão proposto, as métricas analisadas são o *Erro Médio Absoluto* (EMA) entre os tempos de *download* previstos e reais (ou os percentuais de *download* completado, no caso do cenário 2), expresso pela Eq. (3.5); o *Desvio Padrão* associado, conforme a Eq. (3.2); e o tempo de processamento.

$$EMA = \sqrt{\frac{1}{T} \cdot \sum_{t=0}^{T-1} |real_t - previsto_t|}. \tag{3.5}$$

Os resultados são exibidos nas Tabelas 7 e 8 a seguir, para os cenários 1 e 2, respectivamente.

Com base na Tabela 7, é possível observar que todos os modelos apresentaram EMA relativamente baixo, sendo *LightGBM* aquele com melhor desempenho, com $EMA = 0.11553$; e *MLP* aquele com o pior desempenho, com $EMA = 0.27163$. Já na Tabela 8, devido à presença de sombreamento, os valores de EMA são maiores em geral, conforme esperado. Novamente, o mais preciso foi *LightGBM* com $EMA = 5.08845$, e o menos preciso foi *MLP* com $EMA = 6.32415$.

Sob a perspectiva do tempo de processamento, *KNN* obteve a maior velocidade de execução, possivelmente explicado pela base de dados não tão numerosa, o hiperparâmetro

Tabela 9 – Resultados do Teste-t para a regressão no cenário 1.

G_1	G_2	p-valor	**Limite inf.**	**Limite sup.**	H_0
XGB	Light	6.85E-7	0.0052	0.0106	1
XGB	GBM	0.5785	-0.0047	0.0027	0
XGB	MLP	8.11E-40	-0.1530	-0.1406	1
XGB	RF	0.0493	-0.0014	0.0063	1
XGB	KNN	0.0049	-0.0076	-0.0014	1
Light	GBM	1.88E-7	-0.0118	-0.0061	1
Light	MLP	2.16E-34	0.1605	0.1489	1
Light	RF	9.43E-6	0.0068	-0.0029	1
Light	KNN	5.89E-17	-0.0143	-0.0104	1
GBM	MLP	2.16E-40	-0.1520	-0.1395	1
GBM	RF	0.0128	0.0009	0.0073	1
GBM	KNN	0.0347	-0.0067	-0.0002	1
MLP	RF	1.30E-36	0.1439	0.1558	1
MLP	KNN	8.43E-36	0.1364	0.1486	1
RF	KNN	4.92E-8	0.0051	0.0100	1

K consideravelmente pequeno e a natural simplicidade do modelo, reduzindo o custo computacional. O segundo menor tempo de execução foi obtido pelo *LightGBM*, que possui a velocidade de processamento como uma vantagem de sua arquitetura. Em contraste, *RF* foi aquele com maior tempo demandado, possivelmente devido ao número de árvores criadas em sua fase de treinamento.

O Teste-t de duas amostras [58] é novamente aplicado, agora buscando diferenças estatísticas entre os modelos de regressão. As Tabelas 9 e 10 são obtidas, portanto, para os cenários 1 e 2, respectivamente.

Analisando ambas as Tabelas, no cenário 1 observa-se a ocorrência da hipótese nula uma vez. *XGBoost* e *GBM* podem ser considerados, nesse caso, equivalentes. O mesmo se aplica a *RF* e *KNN* no cenário 2. Dessa forma, fica claro que *LightGBM* é aquele que melhor se adequa à base de dados levantada pela primeira campanha de simulações, também apresentando um tempo de execução pequeno, visto que ele possui o menor valor para EMA. O Teste-t confirma que não há outro modelo com desempenho equivalente, ou seja, a hipótese nula $H_0 = 0$ não está associada a nenhuma comparação com o mesmo. Observa-se que o *KNN* oferece uma performance razoável, requerendo um tempo extremamente baixo para execução.

3.5 Conclusões

Este Capítulo apresentou os cenários elaborados, bem como os detalhes das campanhas de simulação produzidas para a obtenção de dados, visando a implementação da

Tabela 10 – Resultados do Teste-t para a regressão no cenário 2.

G_1	G_2	p-**valor**	**Limite inf.**	**Limite sup.**	H_0
XGB	Light	1.99E-37	0.7565	0.8363	1
XGB	GBM	3.00E-20	-0.3563	-0.2658	1
XGB	MLP	1.13E-27	-0.4830	-0.3956	1
XGB	RF	1.19E-6	-0.1576	-0.0731	1
XGB	KNN	2.71E-6	-0.1602	-0.0710	1
Light	GBM	1.95E-53	-1.1401	-1.0748	1
Light	MLP	3.94E-60	-1.2661	-1.2053	1
Light	RF	3.51E-57	-0.9399	-0.8836	1
Light	KNN	1.42E-50	-0.9437	-0.8803	1
GBM	MLP	3.50E-9	-0.1656	-0.0908	1
GBM	RF	3.95E-16	0.1602	0.2314	1
GBM	KNN	5.43E-15	0.1571	0.2338	1
MLP	RF	3.89E-28	0.2904	0.3576	1
MLP	KNN	2.75E-26	0.2871	0.3602	1
RF	KNN	0.9867	-0.0344	-0.0350	0

primeira arquitetura proposta para a integração de modelos de IA/ML ao procedimento de *handover* em redes celulares. Além disso, os resultados obtidos também foram discutidos, observando-se as métricas avaliadas para o desempenho dos modelos testados.

Em seguida, será apresentada a segunda arquitetura de integração de modelos computacionais ao *handover*, dessa vez utilizando-se uma rede LSTM, além dos classificadores.

4 ESTUDOS COM A SEGUNDA ARQUITETURA PROPOSTA

Este Capítulo apresenta com detalhes a segunda arquitetura proposta para o uso de IA visando a otimização de processos de *handover* em redes celulares. Serão abordados detalhes da campanha de simulação realizada, os resultados obtidos com a aplicação de uma nova proposta para a predição do disparo de *handover* que emprega *Deep Learning* e classificadores. Além disso, são apresentados os resultados da aplicação de tais modelos em dados reais obtidos a partir de uma rodada de medições em Belo Horizonte, na rede de uma operadora de telefonia celular operando na faixa de 700 MHz. Novamente, todos os códigos foram testados utilizando Python em um computador com processador i7-6700HQ de 2.6 GHz.

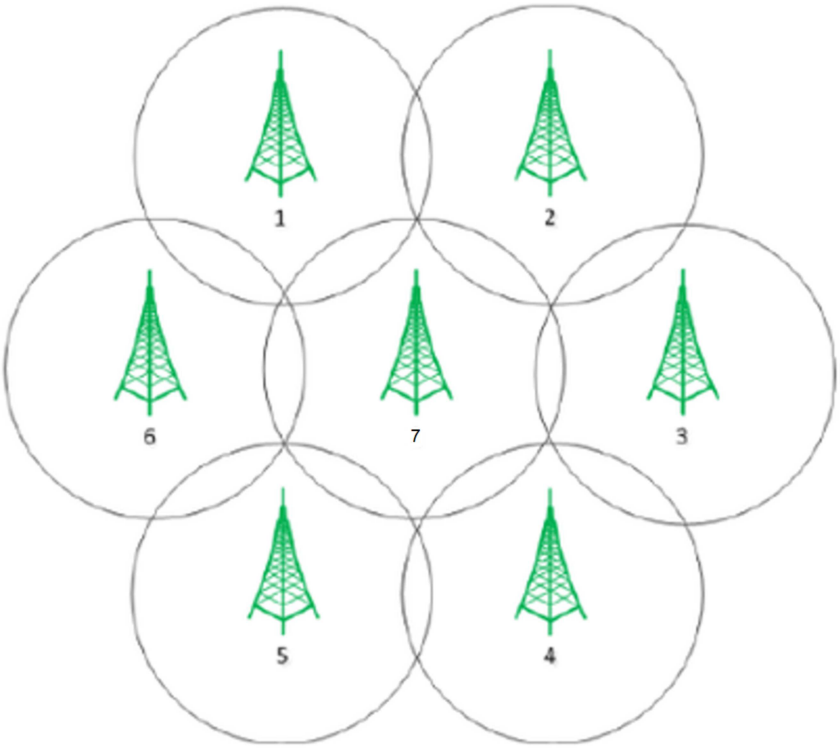
4.1 Cenário em grid hexagonal

Foi utilizado o simulador *ns-3* [56], muito tradicional para simulações de redes baseadas em eventos [29, 30]. O ambiente, dessa vez, conta com 21 células setorizadas, alocadas em 7 BSs, como ilustrado na Figura 15, além de 20 UEs se deslocando com velocidade de 60 km/h. O modelo Okumura-Hata para propagação é usado para modelagem da perda de percurso. Além disso, o sombreamento de larga escala é empregado ($\sigma = 8$ dB) e o desvanecimento rápido, seguindo uma distribuição *Rayleigh*, também é considerado. Os níveis de RSRP são capturados a cada 1 ms para todos os UEs. Novamente, empregou-se o protocolo LTE. Cada simulação, dessa vez, dura 10 segundos. Todo disparo de *handover* também é anotado, a fim de rotular a base de dados para a classificação posterior.

4.2 Cenário de rede real

Nesse cenário, o quarto contemplado por este trabalho, agora a arquitetura proposta é submetida a testes com os dados levantados a partir de uma coleta realizada pela Anatel em Belo Horizonte (Minas Gerais-MG) em uma rede celular LTE. Os dados foram coletados para eNBs operando na faixa de 700 MHz (4.5G) utilizando uma banda de 10 MHz. A campanha de medições se deu por diversos bairros da cidade, ao longo dos dias 14, 18 e 19 do mês de dezembro de 2018. Portanto, podemos nos basear nas séries temporais das medidas dos sinais de referência capturados por um dispositivo móvel que, ao se deslocar pelo ambiente, realizará *handovers* conforme a norma já discutida.

As Figuras 16 e 17 mostram alguns dos trajetos e condições de sinal medidas pelo *drive test* realizado.



– Fig. 15: Grid hexagonal com 7 células do cenário 3. Elaborada pelo autor.

– Fig. 16: Um dos trajetos realizado no bairro Braúnas, Belo Horizonte.

– Fig. 17: Outro trajeto realizado no bairro Castelo, Belo Horizonte.

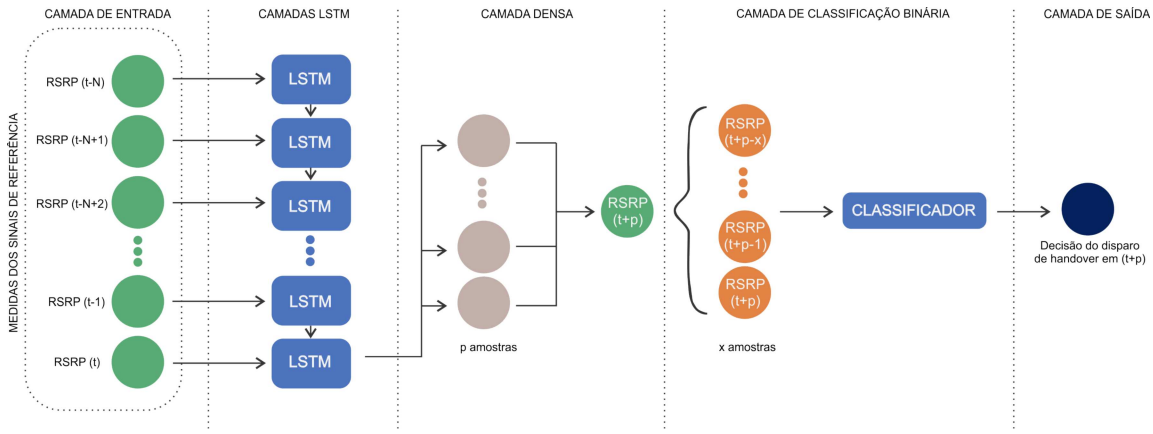
4.3 Segunda implementação proposta

O objetivo desta implementação é prever uma decisão de *handover* ao estimar as condições de canal que levariam aos eventos de disparo. Primeiramente, são analisadas as medidas de canal obtidas, seja nas campanhas de simulação, ou naquelas em rede real. Em seguida, tais medidas são usadas no modelo de *Deep Learning* desenvolvido para estimar futuras amostras de sinais de potência utilizando redes do tipo LSTM. Finalmente, a série temporal composta por amostras previstas é empregada como entrada para o estágio final da proposta elaborada, a qual classifica se tal série acarretará ou não em um disparo de *handover*.

Uma vez que a decisão de disparo de *handover* é prevista, outros elementos da arquitetura de redes móveis pode fazer uso de tal informação relevante para melhorar o desempenho tanto da rede quanto do UE com outras aplicações, como, por exemplo:

- Alocar recursos de rádio para o UE que entrará na célula com boa previsibilidade;
- Evitar *handovers* desnecessários que podem ser disparados por flutuações das condições de canal inesperadas;
- Alertar veículos conectados sobre um *handover* próximo em cidades inteligentes;
- Adaptar planos de voo de UAVs (sigla em inglês para veículos aéreos não-tripulados, comumente tratados por drones) em cenários urbanos;
- Alimentar outras ferramentas da rede para evitar falhas de enlace de radio (conhecidas como *Radio Link Failures*) e reestabelecimentos de conexão.

Em seguida, a segunda arquitetura para a integração de algoritmos de IA/ML para otimização de *handovers* é apresentada. O modelo, baseado em uma RNN, é composto por cinco estágios consecutivos, como visto na Figura 18.



– Fig. 18: Segunda arquitetura aplicada à otimização de *handover*. Elaborada pelo autor.

- O primeiro estágio é a camada de entrada, em formato de janelamento, para moldar as últimas N amostras obtidas pelas medidas do canal (através dos *Measurement Reports*);

- O segundo estágio é composto por camadas LSTM. Uma camada LSTM é um tipo específico de camada para RNNs. Com a adição de mais conexões internas comumente tratadas como *gates* (do inglês, portões), a unidade LSTM consegue potencializar muito sua memória longa de curto prazo, característica desse design. Seu mecanismo se baseia na capacidade de preservar por mais etapas de treinamento o erro que foi retornado pelo algoritmo de *backpropagation*. Dessa forma, ao possuir um erro menos volátil, o neurônio (e consequentemente a rede como um todo) consegue aprender por mais tempo, extraindo informações remotas no tempo.

- O terceiro estágio é a camada densa, na qual as saídas oriundas das camadas LSTM são combinadas em um último neurônio, produzindo a estimativa do sinal de referência no tempo $t + p$. Tal elemento possui uma função de ativação diferente, visando produzir a saída numérica desejada (a estimativa de uma medida de potência). A função selecionada para esse propósito foi a função *Sigmoid*, enquanto a função *ReLu* se mostrou mais adequada para as camadas ocultas. O resultado desse estágio são amostras previstas de potência;

- Em seguida, o quarto estágio é modelado através de uma vetorização das saídas do estágio anterior, com x amostras servindo como entrada para a etapa de classificação binária. Essa etapa é composta por outro algoritmo, um classificador construído para melhor indicar se a sequência de amostras de potência previstas, submetida em

sua entrada, levará ou não a um disparo de *handover*. Se trata, por tanto, de uma tarefa de classificação;

- Finalmente, o quinto estágio é a camada de saída, a qual determina um resultado binário de uma estimativa de disparo de *handover* no tempo $t + p$.

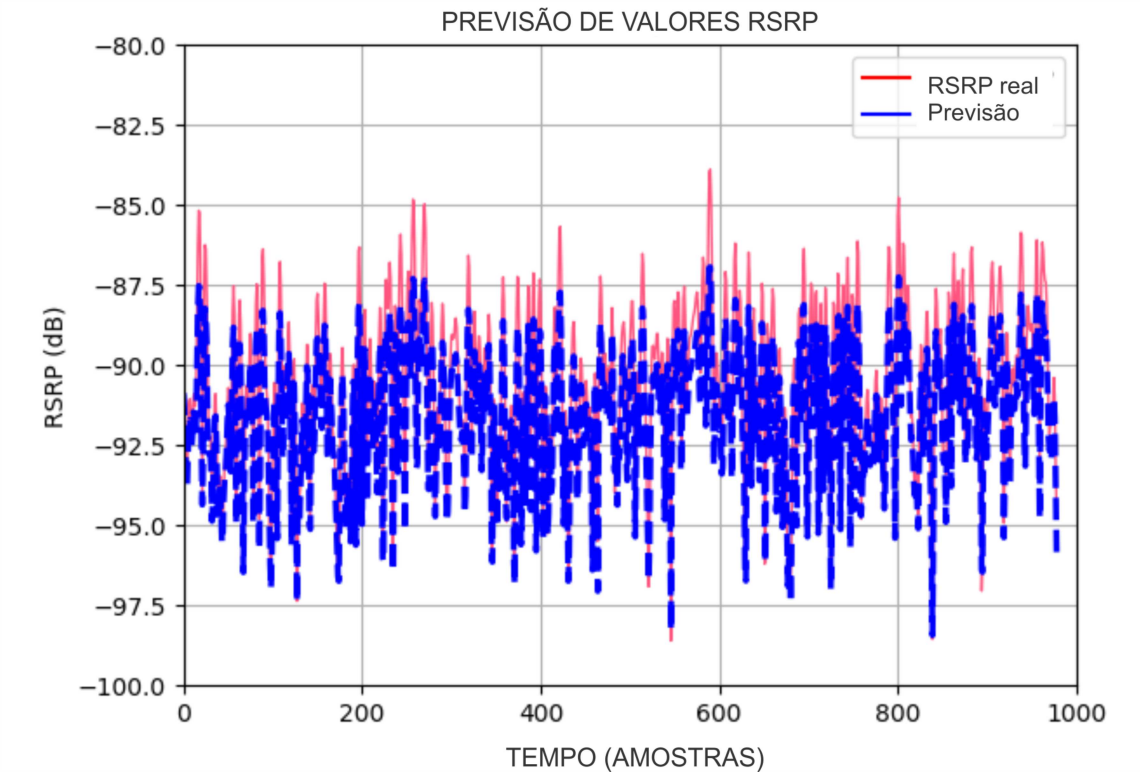
Após a coleta de dados obtidos com a campanha de simulação, há a necessidade de pré-processamento dos mesmos, para configurar as dimensões das entradas para os modelos utilizados. A rede LSTM requer um vetor 3D com amostras, registro temporal e as características, ao passo que a classificação necessita de um vetor 2D com amostras e características, apenas. Além disso, nota-se que a entrada do quarto estágio é composta de x amostras, visando oferecer flexibilidade à proposta. Ao prever p amostras além do tempo t no estágio LSTM, não necessariamente precisam-se utilizar as p amostras como entradas no classificador. Neste trabalho, utilizaram-se 50 amostras como x.

O modelo LSTM foi desenvolvido com o uso das bibliotecas Keras [59], empregando o otimizador RMSProp e a função de perda do Erro Médio Quadrático, para que o menor Erro Médio Absoluto fosse verificado nas amostras previstas de sinais de potência. O treinamento do estágio de LSTM foi feito com base em muitas séries temporais de RSRP dos UEs simulados. Foram adicionadas, também, camadas *Dropout* no modelo LSTM para evitar o efeito de *overfitting*. Tais camadas ignoram parte dos resultados dos neurônios aleatoriamente, não permitindo que as camadas posteriores compensem possíveis erros produzidos por camadas anteriores. Nesse sentido, a técnica de validação cruzada *Stratified k-Fold* (com $k = 5$) [52], abordada no Capítulo 2, foi novamente utilizada no estágio de classificação para partição dos dados em treino e teste.

Na sequência, foram comparados quatro algoritmos clássicos de classificação, sendo eles SVM, MLP, KNN and RF, descritos no Capítulo 2. Os melhores parâmetros encontrados para cada um deles foram:

- Para o SVM, o Kernel empregado foi o *rbf*, com fator de penalidade 500;
- O MLP contou com *solver LBFGS*, além de 2 camadas ocultas de 120 neurônios cada;
- Por sua vez, o KNN tem $K = 2$, além do fator de potência 1, o qual indica o uso da distância *Manhattan*;
- E por fim, o modelo RF usou 200 estimadores e empregou o critério *Gini* para a qualidade das suas subdivisões.

Novamente, todos os detalhes de treino, teste, parâmetros dos modelos, bem como códigos utilizados para as simulações em *ns-3* são disponibilizados em [60].



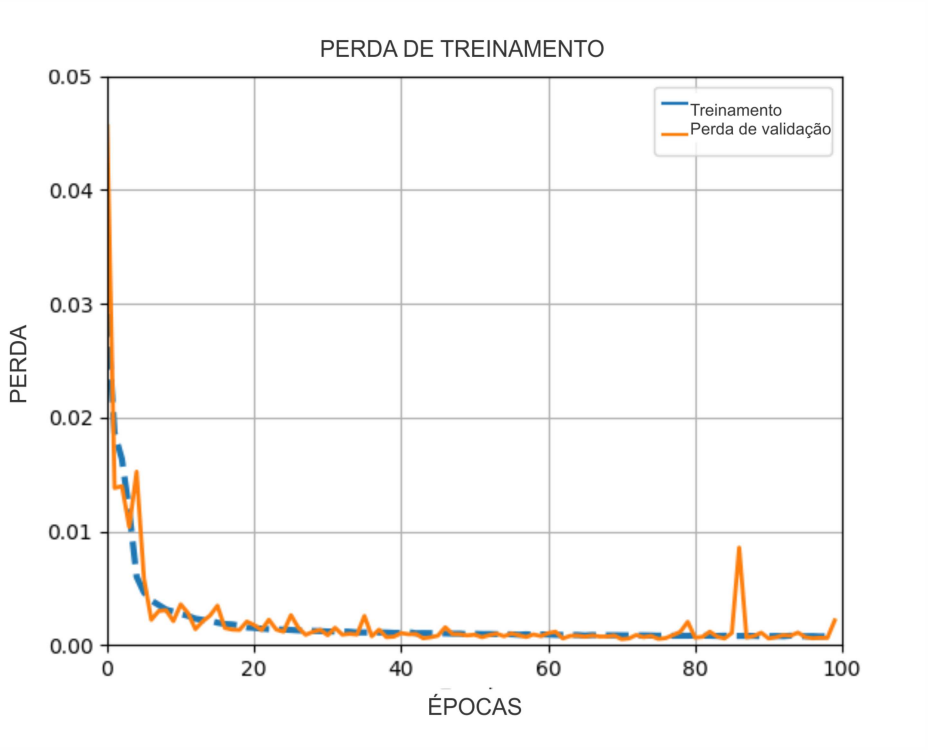
– Fig. 19: Previsões de RSRP feitas pela rede LSTM.

4.4 Resultados com cenário em grid hexagonal

A avaliação do modelo LSTM que prevê as amostras de RSRP no tempo $t + p$ também é novamente feita segundo a métrica de Erro Médio Absoluto (EMA), dada pela Eq. (3.5).

Após a fase de treinamento, foi obtido $EMA = 0.6$ dB, valor considerado muito baixo, dado que os valores típicos de RSRP estão no intervalo [-60, -110 dB]. A Figura 19 ilustra a ótima adequação do modelo ao problema, considerando severas condições de propagação aplicadas na simulação. É relevante notar que o desempenho obtido é dado para a predição numa janela de 100 amostras à frente do tempo t, o que nesse caso significa 100 ms de antecipação, uma quantidade de tempo significativa para tomadas de decisão para modelos de IA/ML (como os empregados na segunda etapa) ou para a rede, em geral. Ao aumentar muito esse parâmetro, visando prever amostras para tempos cada vez mais distantes, o erro da rede LSTM naturalmente aumenta. O processo de treinamento pode ser visualizado na Figura 20, na qual a função de perda decresce ao passo que o modelo evolui em desempenho.

Para a etapa de Classificação, a avaliação consiste nas métricas de Acurácia, Desvio Padrão e *F1 Score*, expressas por (3.1), (3.2) e (4.3), respectivamente. Essa última compõe um balanço entre Precisão e *Recall*, representados por (4.1) e (4.2), respectivamente. Nesse caso, TP é o número de verdadeiros positivos, TN é o número de verdadeiros negativos,



– Fig. 20: Perdas no treinamento e validação da rede LSTM.

FP é o número de falsos positivos e FN, o de falsos negativos. Por usa vez, a métrica *F1 Score* indica melhor se o modelo demonstra um desempenho satisfatório, pois ela considera não só a precisão do modelo, mas também o quão sensível ele é ao evento de interesse. Ou seja, se ele é capaz de ter alta precisão nas duas classes presentes na base de dados. É válido salientar que optou-se por utilizar a métrica *F1 Score* nesse estudo como uma nova forma de avaliação dos modelos, visando justamente a exploração de mais metodologias relacionadas às ferramentas matemáticas e estatísticas aplicadas à manipulação de dados, num caráter didático. Assim como nas avaliações feitas no Capítulo 3, empregou-se o Teste-t, como forma de consolidar um estudo de testes de hipóteses ao longo do curso.

$$Precisão = \frac{TP}{TP + FP} \tag{4.1}$$

$$Recall = \frac{TP}{TP + FN} \tag{4.2}$$

$$F1 = \frac{2}{\frac{1}{Precisão} + \frac{1}{Recall}} = \frac{TP}{TP + 0.5 \cdot (FP + FN)} \tag{4.3}$$

Os resultados dessa etapa são mostrados na Tabela 11.

Embora MLP e SVM sejam algoritmos robustos para classificação binária, vê-se na Tabela 11 que RF tem desempenho levemente superior considerando as métricas avaliadas.

Tabela 11 – Comparação de algoritmos de classificação no cenário 3 (grid hexagonal).

Modelo	**Acurácia (± Desv. Padrão) (%)**	**F1 Score (%)**
SVM	96.51 (± 0.18)	97.28
MLP	96.11 (± 0.29)	96.23
KNN	88.72 (± 0.28)	89.77
RF	97.63 (± 0.13)	97.64

É possível que nesse caso sua melhor performance se dê pelo fato de se tratar de um algoritmo baseado em árvores de decisão, o qual tende a se adequar bem a problemas que envolvem avaliação de decisões em múltiplos passos. Alcançar mais de 97% de acurácia é considerado um excelente resultado. Isso significa dizer que quase todas as séries temporais de sinais de referência são corretamente classificadas para predizer futuros disparos de *handover*. Visto que *F1 Score* atinge sua melhor avaliação quando próximo de 1 (0 seria o pior caso), os altos valores obtidos determinam que os modelos são sensíveis o suficiente para fornecer alta acurácia e alta precisão. Ou seja, eles são capazes de classificar adequadamente ambas as categorias (será ou não um disparo de *handover*). Por outro lado, KNN não pôde entregar um desempenho competitivo nesse caso, provavelmente devido à simplicidade do seu algoritmo baseado em cálculos simples de distâncias.

Outro aspecto importante para menção é que o estágio de classificação requer muito pouco tempo de processamento nesse caso. As fases de treinamento e avaliação de modelos demandaram apenas alguns segundos. Dessa forma, com um modelo já treinado, o tempo consumido para a classificação de uma amostra é desprezível, em conformidade para aplicações em redes reais.

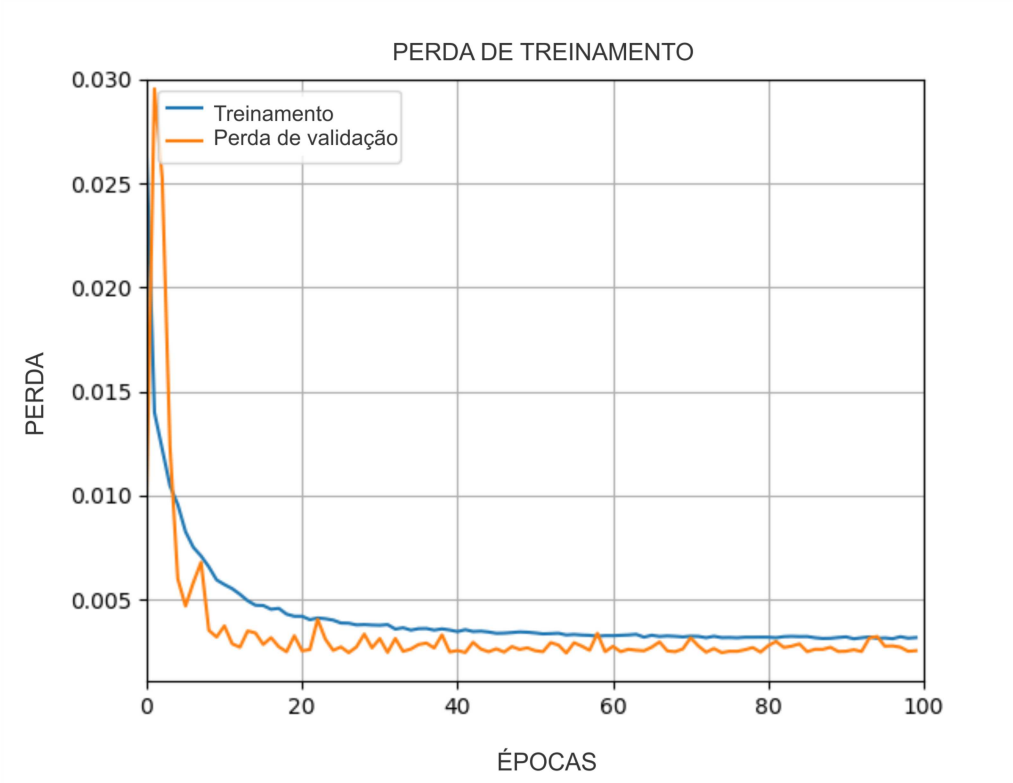
4.5 Resultados com dados da rede real

Inicialmente, foi testado o modelo LSTM que melhor se adequou aos dados simulados tratado na Seção anterior. Com isso, visa-se avaliar a possibilidade de gerar modelos eficientes treinados a partir de dados simulados, o que poderia conferir um grande horizonte de possibilidades para experimentação, validação e implementação de modelos de IA em redes celulares.

No entanto, ao carregar o modelo previamente salvo e fazer previsões do sinal de RSRP a partir dos dados da nova base, o resultado obtido não se mostrou satisfatório. Mais uma vez com base na métrica do EMA, já descrito *a priori* pela Eq. (3.5), obteve-se $EMA = 3.41\ dB$, contra apenas $0.6\ dB$ quando utilizado sobre os dados simulados.

Portanto, faz-se necessário o treinamento do modelo com os dados reais das medições feitas em Belo Horizonte para refinar o desempenho da arquitetura proposta. Após essa etapa, a qual pode ser visualizada em Figura 21, o modelo se tornou mais aderente

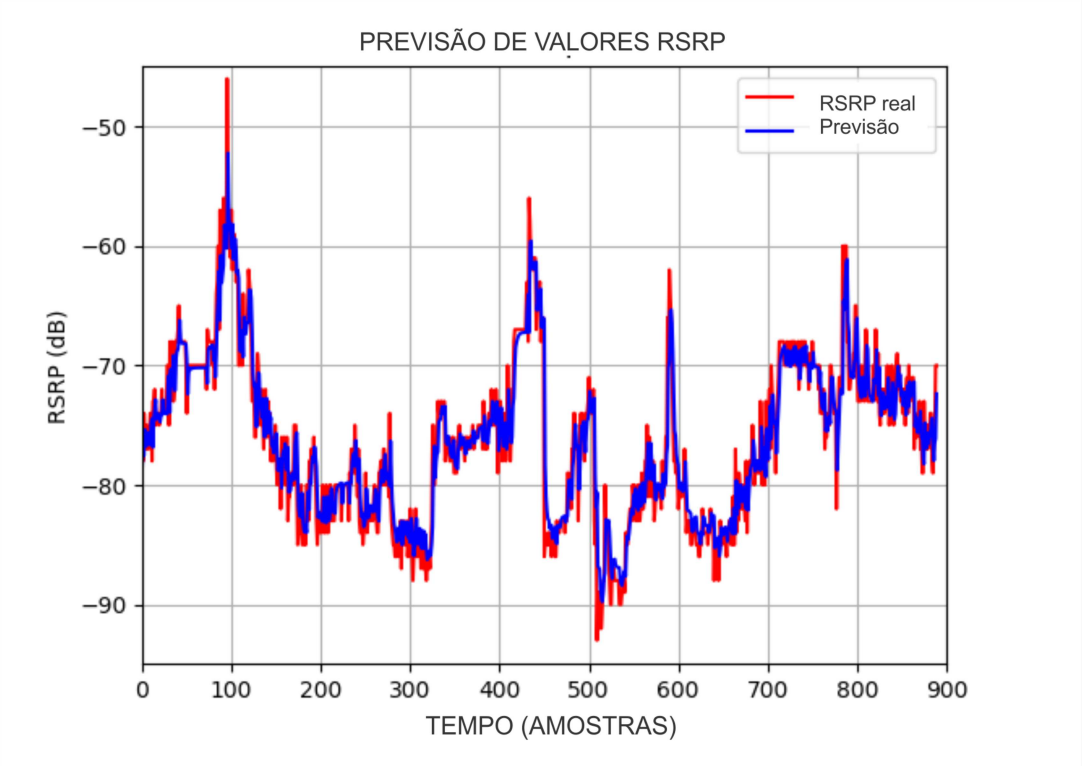
aos dados, o que pode ser comprovado pelo novo erro obtido: $EMA = 1.74\ dB$, ainda significativamente maior do que aquele com dados simulados. Como confirmado pela Fig. 21, ao longo das épocas de treinamento, a perda resultante decresce, devido ao correto ajuste dos parâmetros e pesos do modelo. Tem-se, dessa maneira, a comparação entre o sinal RSRP real e suas amostras previstas exibida na Figura 22. É possível observar uma ótima conformidade entre os dados previstos e reais, com pequenos desvios ocorrendo principalmente nos pontos mais agudos ou extremados (nos grandes picos), fenômeno natural e esperado. É importante reafirmar que os dados utilizados nesse experimento são dados reais de uma operadora celular, o que valida com mais solidez a proposta desenvolvida.



– Fig. 21: Perdas no treinamento e validação da rede LSTM com dados reais.

Agora, para a nova etapa de classificação, novamente serão utilizadas as mesmas métricas para avaliação dos modelos: *Acurácia* (dada pela Eq. (3.1)), *Desvio Padrão* e *F1 Score* (dado pela Eq. (4.3)). Foram comparados os mesmos algoritmos do estudo anterior (KNN, SVM, MLP e RF). Dessa maneira, os resultados obtidos são exibidos na Tabela 12.

Neste estudo, é possível observar na Tabela 12 uma ligeira vantagem em acurácia do algoritmo SVM em relação aos demais. Já em termos da métrica *F1 Score*, vê-se que RF e KNN apresentam valores levemente superiores e, por consequência, erram menos no balanço entre ambas as classes avaliadas (será ou não um *handover*). Isso permite dizer que esses modelos se mostram mais calibrados e aderentes aos dados apresentados. Novamente,



– Fig. 22: Previsões de RSRP feitas pela rede LSTM com dados reais.

Tabela 12 – Comparação de algoritmos de classificação no cenário 4 (com dados reais).

Modelo	**Acurácia (± Desv. Padrão) (%)**	**F1 Score (%)**
SVM	95.39 (± 0.26)	95.01
MLP	93.69 (± 0.30)	93.97
KNN	95.08 (± 0.29)	95.24
RF	95.10 (± 0.30)	95.23

MLP teve o pior resultado, se mostrando menos eficaz para lidar com o problema proposto.

4.6 Conclusões

Este Capítulo apresentou os cenários utilizados para a avaliação da segunda arquitetura proposta para otimização de *handover* empregando modelos de IA/ML. Os detalhes das campanhas de simulação produzidas para a obtenção de dados, bem como informações da rede real utilizada para coleta de dados de uma operadora celular foram abordados. Além disso, os resultados obtidos também foram discutidos, observando-se as métricas avaliadas para o desempenho dos modelos testados.

Para finalizar, a seguir serão discutidas as conclusões oriundas de todo o trabalho, além de possibilidades de desenvolvimentos futuros, que venham a enriquecer este trabalho.

5 CONCLUSÃO E TRABALHOS FUTUROS

A seguir, são apresentadas as conclusões e indicações de trabalhos futuros advindos dessa dissertação.

5.1 Considerações finais

Este trabalho se propôs a investigar e comparar o desempenho de diversas técnicas de inteligência artificial e aprendizado de máquina aplicadas ao aprimoramento do *handover* em redes celulares, a partir da integração de tais técnicas aos procedimentos existentes atualmente nos padrões 4G e 5G vigentes. Foram elaboradas duas arquiteturas para aplicações, cada uma avaliada em dois estudos distintos. Primeiramente, cenários simplificados com diferentes condições de propagação foram simulados para tarefas de classificação e regressão de acordo com a primeira arquitetura, para predizer o melhor alvo de *handover* disponível, além do tempo previsto para o término do download que é realizado na simulação. Na sequência, um cenário mais complexo, composto por um grid hexagonal de sete células, foi proposto para o desenvolvimento da segunda arquitetura, a qual incluiu o uso de *Deep Learning* para previsões de séries temporais (redes LSTM), com a intenção de prever sinais de referência típicos utilizados para o disparo de *handover*, possibilitando a classificação de tal momento de forma antecipada. Adicionalmente, tal arquitetura também foi submetida a um conjunto de dados reais obtidos a partir de uma campanha de medições realizadas em Belo Horizonte na rede de uma operadora celular.

Uma gama de algoritmos foi estudada e avaliada, visando consolidar uma ampla comparação de técnicas no contexto de *handovers*, desde técnicas mais simples, como KNN e RF, passando por mais robustas e consolidadas, como redes neurais artificiais e SVM. O estudo também desenvolveu comparações englobando sistemas *fuzzy* do estado da arte, bem como técnicas avançadas da família *boosting machines*.

Além disso, diversas outras estratégias foram aplicadas nas etapas que envolvem o treinamento e teste dos modelos, bem como a adequação e processamento dos dados. Destacam-se, nesse sentido, a validação cruzada, as técnicas de super e sobre-amostragem, a busca por hiperparâmetros dos modelos, bem como o teste estatístico para verificação do desempenho dos modelos.

No estudo da primeira arquitetura, pôde-se comprovar que SVM e MLP obtiveram os melhores resultados na tarefa de classificação do melhor alvo de *handover*, sem diferença significativa entre ambos os modelos. Já na tarefa de regressão para o tempo ou percentual de download, o modelo *LightGBM* foi aquele capaz de entregar o menor erro médio absoluto, além de um tempo de processamento reduzido.

Por sua vez, para a segunda arquitetura proposta, a rede LSTM apresentou excelente aderência aos sinais RSRP avaliados, com erro médio absoluto de 0.6 dB e 1.74 dB quando

aplicada sobre os dados simulados e reais, respectivamente. Tal desempenho é expressivo quando posto em perspectiva o fato de que os valores típicos desse sinal flutuam ao redor de -60 a -110 dB, possibilitando o uso dos dados previstos em outras tarefas sequentes com alta eficácia. Dessa maneira, a arquitetura se propôs a determinar se a série temporal prevista iria ou não resultar em um disparo de *handover*. Essa é uma aplicação prática com grande valor no contexto de redes celulares, pois tal informação, quando precisa, pode antecipar e servir de base para a tomada de decisão em diversos contextos e estratégias. Essa avaliação obteve cerca de 97% e 95% de acurácia para os dados simulados e reais respectivamente, um notável desempenho capaz de validar a proposta.

5.2 Trabalhos futuros

Como decorrência desse trabalho, são indicadas algumas possibilidades para futuras contribuições e aprimoramentos, listadas abaixo:

- Busca por novos algoritmos para uma comparação ainda mais ampla nos métodos;
- Desenvolvimento de um modelo inteiramente *fuzzy* para a predição de amostras futuras de sinais de referência (RSRP, por exemplo), além da etapa de classificação;
- Testar as arquiteturas propostas com dados coletados em sistemas que usam frequências mais altas, como por exemplo 3.5 GHz e até mesmo 26 GHz, que naturalmente terão ainda mais dificuldades na propagação;
- Implementar a primeira arquitetura em um cenário em grid hexagonal (como o cenário 3), baseada num algoritmo de Aprendizagem por Reforço atuando sobre os parâmetros típicos de disparo de *handover* (*Time-To-Trigger*, por exemplo), observando métricas do sistema de comunicação (por exemplo, *throughput* total das células, número médio de *handovers* por usuário por segundo, latência média resultante, entre outras);
- Implementação das arquiteturas propostas embarcadas em microsserviços numa infraestrutura de rede aberta (Open RAN).

REFERÊNCIAS

[1] Mohammad Hasan. *State of IoT 2022: Number of connected IoT devices growing 18% to 14.4 billion globally*. 2022. URL: https://iot-analytics.com/number-connected-iot-devices/.

[2] "3GPP TS 36.133". Em: Evolved Universal Terrestrial Radio Access (E-UTRA); Requirements for Support of Radio Resource Management. 3GPP, 2009.

[3] "3GPP TS 36.331". Em: Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Resource Control (RRC); Protocol specification. 3GPP, 2009.

[4] Ministério das Comunicações. *Leilão do 5G confirma expectativas e arrecada 47,2 bilhões de reais*. 2021. URL: https://www.gov.br/pt-br/noticias/transito-e-transportes/2021/11/leilao-do-5g-confirma-expectativas-e-arrecada-r-47-2-bilhoes.

[5] Chunxiao Jiang et al. "Machine learning paradigms for next-generation wireless networks". Em: *IEEE Wireless Communications* 24.2 (2017), pp. 98–105.

[6] Tasnim Abar, Asma Ben Letaifa e Sadok El Asmi. "Machine learning based QoE prediction in SDN networks". Em: *International Wireless Communications and Mobile Computing Conference*. IEEE. 2017, pp. 1395–1400.

[7] Tarik Begluk, Jasmina Baraković Husić e Sabina Baraković. "Machine learning-based qoe prediction for video streaming over lte network". Em: *International Symposium Infoteh-Jahorina*. IEEE. 2018, pp. 1–5.

[8] Pedro Casas et al. "Predicting QoE in cellular networks using machine learning and in-smartphone measurements". Em: *International Conference on Quality of Multimedia Experience*. IEEE. 2017, pp. 1–6.

[9] Yaping Chang, Wei Li e Zhongming Yang. "Network intrusion detection based on random forest and support vector machine". Em: *International Conference on Computational Science and Engineering and International Conference on Embedded and Ubiquitous Computing*. IEEE. 2017, pp. 635–638.

[10] Chuanlong Yin et al. "A deep learning approach for intrusion detection using recurrent neural networks". Em: *IEEE Access* 5 (2017), pp. 21954–21961.

[11] Robert Falkenberg et al. "Machine Learning Based Uplink Transmission Power Prediction for LTE and Upcoming 5G Networks Using Passive Downlink Indicators". Em: *IEEE Vehicular Technology Conference*. IEEE. 2018, pp. 1–7.

[12] Huiwei Xia et al. "Traffic Prediction Based on Ensemble Machine Learning Strategies with Bagging and LightGBM". Em: *IEEE International Conference on Communications Workshops*. IEEE. 2019, pp. 1–6.

[13] Xinping Gu, Yunpeng Han e Junfu Yu. “A Novel Lane-Changing Decision Model for Autonomous Vehicles Based on Deep Autoencoder Network and XGBoost”. Em: *IEEE Access* 8 (2020), pp. 9846–9863.

[14] Ahmad Mudassir et al. “A survey on fuzzy logic applications in wireless and mobile communication for LTE networks”. Em: *International Conference on Complex, Intelligent, and Software Intensive Systems*. IEEE. 2016, pp. 76–82.

[15] Muhammad Tayyab, Xavier Gelabert e Riku Jäntti. “A survey on handover management: From LTE to NR”. Em: *IEEE Access* 7 (2019), pp. 118907–118930.

[16] Adamantia Stamou et al. “Autonomic Handover Management for Heterogeneous Networks in a Future Internet Context: A Survey”. Em: *IEEE Communications Surveys & Tutorials* 21.4 (2019), pp. 3274–3297.

[17] L. Yan et al. “Machine Learning-Based Handovers for Sub-6 GHz and mmWave Integrated Vehicular Networks”. Em: *IEEE Transactions on Wireless Communications* 18.10 (2019), pp. 4873–4885.

[18] June-ho Bang et al. “A Bayesian Regression Based LTE-R Handover Decision Algorithm for High-Speed Railway Systems”. Em: *IEEE Transactions on Vehicular Technology* 68.10 (2019), pp. 10160–10173.

[19] Li Yan et al. “Machine Learning-Based Handovers for Sub-6 GHz and mmWave Integrated Vehicular Networks”. Em: *IEEE Transactions on Wireless Communications* 18.10 (2019), pp. 4873–4885.

[20] Sadegh Aghabozorgi et al. “Toward Predictive Handover Mechanism in Software-Defined Enterprise Wi-Fi Networks”. Em: *IEEE Sustainability through ICT Summit*. IEEE. 2019, pp. 1–6.

[21] Diego Castro-Hernandez e Raman Paranjape. “Optimization of Handover Parameters for LTE/LTE-A in-Building Systems”. Em: *IEEE Transactions on Vehicular Technology* 67.6 (2018), pp. 5260–5273.

[22] Mohamed Saeed, Hanan Kamal e Mona El-Ghoneimy. “Novel type-2 fuzzy logic technique for handover problems in a heterogeneous network”. Em: *Engineering Optimization* 50.9 (2018), pp. 1533–1543.

[23] Assem Abdelmohsen et al. “LTE Handover Parameters Optimization Using Q-Learning Technique”. Em: *IEEE International Midwest Symposium on Circuits and Systems*. IEEE. 2018, pp. 194–197.

[24] Malhar Kirtivadan Thakkar et al. “Reducing ping-pong handovers in LTE by using A1-based measurements”. Em: *National Conference on Communications*. IEEE. 2017, pp. 1–6.

[25] Evelin Cardoso, Ketyllen Silva e Renato Francês. “Intelligent handover procedure for heterogeneous LTE networks using fuzzy logic”. Em: *International Wireless Communications and Mobile Computing Conference*. IEEE. 2017, pp. 2163–2168.

[26] Mohamed Saeed, Mona El-Ghoneimy e Hanan Kamal. “An enhanced fuzzy logic optimization technique based on user mobility for LTE handover”. Em: *National Radio Science Conference*. IEEE. 2017, pp. 230–237.

[27] Tarciana Guerra. “Machine Learning Based Handover Management for LTE Networks with Coverage Holes”. Diss. de mestr. Natal, Brazil: Universidade Federal do Rio Grande do Norte, 2018.

[28] Tarciana Guerra, Ycaro Dantas e Vicente Sousa Jr. “A Machine Learning Approach for Handover in LTE Networks with Signal Obstructions”. Em: *Journal of Communication and Information Systems* 35.1 (2020), pp. 271–289.

[29] Zoraze Ali et al. “Recurrent neural networks for handover management in next-generation self-organized networks”. Em: *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications*. IEEE. 2020, pp. 1–6.

[30] Zoraze Ali et al. “Multi-Task Learning for Efficient Management of Beyond 5G Radio Access Network Architectures”. Em: *IEEE Access* 9 (2021), pp. 158892–158907.

[31] Shubham Khunteta e Ashok Kumar Reddy Chavva. “Deep learning based link failure mitigation”. Em: *IEEE International Conference on Machine Learning and Applications*. IEEE. 2017, pp. 806–811.

[32] Qian Liu et al. “Proactive mobility management with trajectory prediction based on virtual cells in ultra-dense networks”. Em: *IEEE Transactions on Vehicular Technology* 69.8 (2020), pp. 8832–8842.

[33] Weijing Qi et al. “Social Prediction-Based Handover in Collaborative-Edge-Computing-Enabled Vehicular Networks”. Em: *IEEE Transactions on Computational Social Systems* 9.1 (2022), pp. 207–217.

[34] Navdeep Uniyal et al. “Intelligent Mobile Handover Prediction for Zero Downtime Edge Application Mobility”. Em: *IEEE Global Communications Conference*. IEEE. 2021, pp. 1–6.

[35] Jose M Alonso, Ciro Castiello e Corrado Mencar. “Interpretability of fuzzy systems: Current research trends and prospects”. Em: *Springer Handbook of Computational Intelligence*. Springer, 2015, pp. 219–237.

[36] Jerry M Mendel. *Uncertain rule-based fuzzy systems*. Springer, 2017.

[37] Lotfi A Zadeh. “Fuzzy sets”. Em: *Information and control* 8.3 (1965), pp. 338–353.

[38] Lotfi A Zadeh. “The concept of a linguistic variable and its application to approximate reasoning—I”. Em: *Information sciences* 8.3 (1975), pp. 199–249.

[39] Plamen P Angelov e Xiaowei Gu. “Empirical fuzzy sets”. Em: *International Journal of Intelligent Systems* 33.2 (2018), pp. 362–395.

[40] Plamen P Angelov, Xiaowei Gu e José C Príncipe. “Autonomous Learning Multimodel Systems from Data Streams”. Em: *IEEE Transactions on Fuzzy Systems* 26.4 (2018), pp. 2213–2224.

[41] Pedro Calderano et al. “An enhanced aircraft engine gas path diagnostic method based on upper and lower singleton type-2 fuzzy logic system”. Em: *Journal of the Brazilian Society of Mechanical Sciences and Engineering* 41.2 (2019), pp. 1–14.

[42] Xiaowei Gu e Plamen P Angelov. “Self-organising fuzzy logic classifier”. Em: *Information Sciences* 447 (2018), pp. 36–51.

[43] Yao Wang et al. “SVM-based spectrum handoff scheme for mobile cognitive radio networks”. Em: *Journal of Information & Computational Science* 12.4 (2015), pp. 1301–1309.

[44] Jerry M Mendel. “Computing derivatives in interval type-2 fuzzy logic systems”. Em: *IEEE Transactions on Fuzzy Systems* 12.1 (2004), pp. 84–98.

[45] Dongrui Wu. “Approaches for reducing the computational cost of interval type-2 fuzzy logic systems: overview and comparisons”. Em: *IEEE Transactions on Fuzzy Systems* 21.1 (2013), pp. 80–99.

[46] Dongrui Wu. “An overview of alternative type-reduction approaches for reducing the computational cost of interval type-2 fuzzy logic controllers”. Em: *International Conference on Fuzzy Systems*. IEEE. 2012, pp. 1–8.

[47] Simon S Haykin. *Neural networks and learning machines*. Pearson Education, 2009.

[48] Zoraze Ali et al. “Machine learning based handover management for improved QoE in LTE”. Em: *Network Operations and Management Symposium*. IEEE. 2016, pp. 794–798.

[49] Rafaela A Campos et al. “A new model to distinguish welds performed by short-circuit GMAW based on FRESH algorithm and MLP ANN”. Em: *Journal of the Brazilian Society of Mechanical Sciences and Engineering* 41.3 (2019), pp. 1–10.

[50] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT Press, 2012.

[51] Nashreen Nesa, Tania Ghosh e Indrajit Banerjee. “Outlier detection in sensed data using statistical learning models for IoT”. Em: *IEEE Wireless Communications and Networking Conference*. 2018, pp. 1–6.

[52] Ian Goodfellow, Yoshua Bengio e Aaron Courville. *Deep Learning*. MIT Press, 2016.

[53] A. Javeed et al. “An Intelligent Learning System Based on Random Search Algorithm and Optimized Random Forest Model for Improved Heart Disease Detection”. Em: *IEEE Access* 7 (2019), pp. 180235–180243.

[54] R. Alencar. *Resampling strategies for imbalanced datasets*. URL: https://www.kaggle.com/code/rafjaa/resampling-strategies-for-imbalanced-datasets/notebook.

[55] Zoraze Ali et al. "Simulating LTE mobility management in presence of coverage holes with ns-3." Em: *International Conference on Simulation Tools and Techniques*. 2015, pp. 279–283.

[56] nsnam. *Network Simulator-3*. URL: https://www.nsnam.org/.

[57] J. P. S. H. Lima. *Notebooks GitHub Repository*. URL: https://github.com/jpshlima/systems_notebooks.

[58] Peter G Moore. "The two-sample t-test based on range". Em: *Biometrika* 44.3/4 (1957), pp. 482–489.

[59] François Chollet. *Keras, the Python Deep Learning API*. 2022. URL: https://keras.io.

[60] J. P. S. H. Lima. *Data GitHub Repository*. URL: https://github.com/jpshlima/ML-based-handover.