Automatizando a Alocação de Usuários em Slices 5G em Arquiteturas Open RAN

Murilo Silva[12], Lucas B. Oliveira[12], Victor Dias[1], Matheus Gomes[1], Fernando Farias[2], André Riker[1], Antônio Abelém[1]

[1] Programa de Pós-Graduação em Ciência da Computação (PPGCC)
Universidade Federal do Pará (UFPA)
Belém – PA – Brasil

[2]Rede Nacional de Ensino e Pesquisa (RNP)
Rio de Janeiro – RJ – Brasil

{murilo.silva, lucas.oliveira, fernando.farias}@rnp.br
victor.leite@ig.ufpa.br, {ariker, abelem}@ufpa.br
{matheus.gomes}@itec.ufpa.br

Abstract. *5G mobile networks have enabled new applications with different traffic characteristics. In this scenario, the adaptive allocation of a User Equipment (UE) to a network slice can be complex due to the various types of traffic generated by user applications. To address this challenge, this work proposes a solution called* USAP-5G *(UE Smart Allocation Platform in 5G Mobile Networks). The solution was developed considering the use of an LSTM (Long Short-Term Memory) machine learning model integrated with an Open RAN and 5G architecture. In tests conducted on the OpenRAN@Brasil testbed, the solution demonstrated effectiveness in reducing decision-making latency and improving the performance of the UE allocation scheme in 5G slices. As a result, the allocation of users in 5G slices was automated, contributing to the reduction of OPEX (Operational Expenditure) in network service management.*

Resumo. *As redes móveis 5G viabilizaram novas aplicações com diferentes características de tráfego. Nesse cenário, a alocação adaptativa de um Equipamento do Usuário (UE) em um slice de rede pode ser complexa, devido aos diferentes tipos de tráfego gerado pelas aplicações do usuário. Para abordar este desafio, este trabalho propõe uma solução denominada* USAP-5G *(UE Smart Allocation Platform in 5G Mobile Networks). A solução foi desenvolvida levando em conta a utilização de um modelo de aprendizado de máquina LSTM (Long Short-Term Memory) integrado a uma arquitetura Open RAN e 5G. Em testes realizados no testbed OpenRAN@Brasil, a solução demonstrou eficácia na redução da latência na tomada de decisão e no desempenho do esquema de alocação de UEs em slices 5G. Com isso, a alocação de usuários em slices 5G foi automatizada, ajudando a reduzir o OPEX (Custo Operacional) no gerenciamento de serviços da rede.*

1. Introdução

A quinta geração de redes móveis (5G) trouxe avanços significativos às características essenciais da rede, como maior capacidade, menor latência e maior velocidade de conexão, possibilitados pelo suporte a diferentes tipos de serviço. Essa flexibilidade permite alocar recursos conforme as demandas específicas da aplicação, como baixa latência para carros autônomos, alta vazão para *streaming* de vídeo e redes de cidades inteligentes com inúmeros dispositivos conectados. Nas redes móveis 5G, isso é possível por meio do

Network Slicing [Afolabi et al. 2018], que divide os recursos físicos em *slices* virtuais, garantindo isolamento, customização e alocação eficiente para diferentes aplicações.

Tecnologias como Virtualização de Funções de Rede (NFV) e Redes Definidas por *Software* (SDN) também têm papel fundamental na nova geração, promovendo uma maior "softwarização" da rede sem fio e, por consequência, habilitando a implantação de tecnologias abertas e interoperáveis, como é o caso da arquitetura *Open Radio Access Network* (Open RAN/O-RAN). Estas redes são um novo paradigma no campo de telecomunicações, consistindo em um modelo em que as funções de rede são desagregadas do hardware, interconectadas por meio de interfaces abertas e aprimoradas por Controladores Inteligentes de RAN (RICs), fomentando maior flexibilidade e eficiência no gerenciamento e operação das redes móveis [Marinova and Leon-Garcia 2024].

No entanto, a academia e a indústria ainda enfrentam desafios referentes ao gerenciamento inteligente dos serviços. As redes 5G utilizam *slices* para atender diferentes serviços, como Banda Larga Móvel Aprimorada (eMBB), Internet das Coisas Massiva (mIoT) e Comunicações Ultra Confiáveis e de Baixa Latência (URLLC), mas identificar o tipo de serviço e alocar o UE à *slice* mais adequada se mostra uma tarefa complexa em ambientes reais. Este processo demanda uma análise detalhada do perfil de fluxo de dados do usuário e, muitas vezes, não é feito de forma dinâmica, resultando em atrasos e imprecisões que têm impacto direto na Qualidade de Serviço (QoS). Portanto, quando este processo não é realizado de maneira automatizada, seu custo operacional tende a aumentar significativamente, pois depende de intervenções manuais por parte dos operadores da rede, limitando a dinamicidade dos serviços e escolhas que podem não ser ideais para as demandas em tempo real de tais aplicações [Donatti et al. 2023].

Nesse sentido, este trabalho apresenta a solução denominada USAP-5G (*UE Smart Allocation Platform in 5G Mobile Networks*). A solução proposta faz alocação dinâmica de UEs em *slices* 5G na arquitetura O-RAN, utilizando técnicas de aprendizado de máquina para analisar as KPMs (Medidas-chave de Desempenho) fornecidas pelas funções de RAN CU (*Central Unit*) e DU (*Distributed Unit*). Essa abordagem possibilita a identificação do tipo de serviço consumido pelo usuário e alocando-o à *slice* conforme às necessidades de consumo do serviço. A solução possui os seguintes módulos: i) coleta de métricas, ii) codificador/decodificador de mensagens entre a xApp e o controlador, iii) o módulo para classificação do UE e a tomada de ação. A solução foi desenvolvida, testada e avaliada no ambiente de experimentação O-RAN do testbed OpenRAN@Brasil, utilizando projetos de código aberto.

As seções deste documento estão organizadas da seguinte forma: A Seção 2 apresenta o embasamento teórico necessário para o entendimento da arquitetura geral do projeto, na Seção 3 são apresentados os trabalhos alinhados no escopo da pesquisa e suas diferenças, dessa forma, destacando as contribuições do trabalho em relação ao estado da arte. A Seção 4 elucida o processo de modelagem da solução e funcionamento geral de suas estruturas principais e a Seção 5 apresenta o cenário utilizado para validação da solução com suas características específicas. Por fim, a Seção 6 traz os resultados obtidos e realiza uma análise de suas implicações, já a Seção 7 faz uma discussão dos resultados e metas futuras em seu contexto.

2. Principais Conceitos em Open RAN

A arquitetura O-RAN traz consigo uma série de componentes e interfaces, os quais são padronizados pelos grupos de trabalho da organização ORAN Alliance [Alliance 2023]. Um elemento importante é o Near-RT RIC, uma Função de Rede (NF) que permite a otimização e controle de recursos dos elementos presentes na RAN, atuando em um tempo próximo ao real. Esse controle só é possível graças a interface E2, outra característica

principal do RIC está na inclusão de fluxos de trabalho de AM (Aprendizado de Máquina), incluindo o treinamento, inferência e atualizações do modelo.

A O-CU-CP (*O-RAN CU Control Plane*) é uma NF que hospeda o protocolo RRC (*Radio Resource Control*) e a parte do plano de controle do protocolo PDCP (*Packet Data Convergence Protocol*). Diferentemente de uma NF tradicional, ela integra o protocolo E2AP para conectividade com o Near-RT RIC. Já a O-CU-UP (*O-RAN CU User Plane*) é responsável pela parte do plano de usuário do protocolo PDCP e pelo protocolo SDAP (*Service Data Adaptation Protocol*), incluindo também o protocolo E2AP para conectividade com o Near-RT RIC. Além disso, a O-DU (O-RAN DU) implementa as camadas RLC, MAC e High-PHY, com base em uma divisão funcional de camadas inferiores, e também inclui o protocolo E2AP, assim como a O-CU-CP e a O-CU-UP.

A interface E2 conecta o Near-RT RIC a um ou mais O-CU-CPs, O-CU-UPs e O-DUs, enquanto o E2 Node representa um nó lógico que termina a interface E2, considerando, elementos como O-CU-CP, O-CU-UP e O-DU para acesso NR (New Radio). Por fim, temos as xApps que são aplicações executadas junto ao Near-RT RIC, compostas por ao menos um microsserviço que, durante sua implementação, identifica os dados que pode consumir ou fornecer. Essas aplicações são independentes do Near-RT RIC e podem ser desenvolvidas por terceiros, com a interface E2 permitindo sua associação direta com as funcionalidades da RAN. A Figura 1 ilustra a visão geral da arquitetura O-RAN e seus componentes.

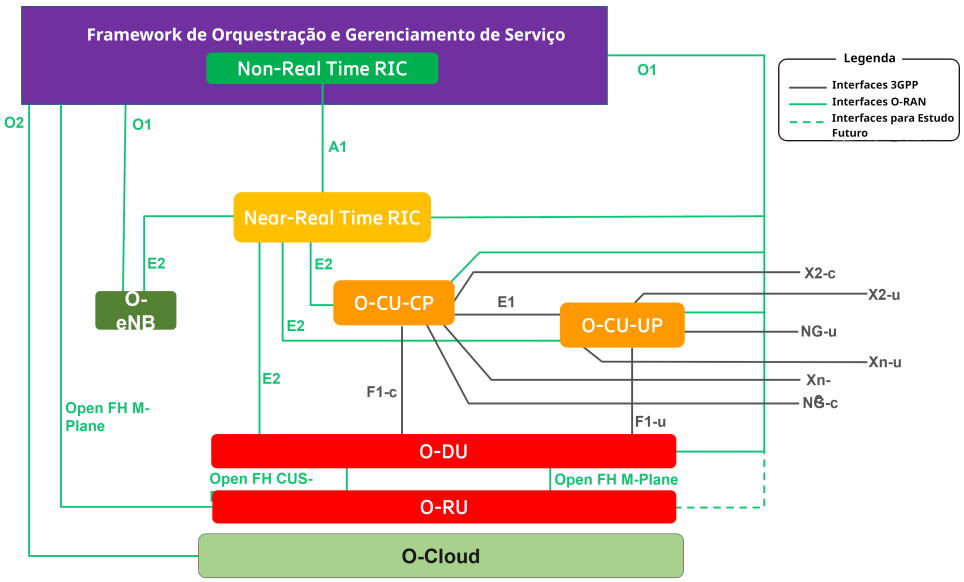


Figura 1. Visão Geral da Arquitetura O-RAN (Fonte: O-RAN SC)

Outro ponto importante na arquitetura O-RAN está na utilização dos Modelos de Serviços E2 (E2SMs), a fim de provisionar as funcionalidades da interface E2, que operam sobre o protocolo E2AP (*E2 Application Protocol*). As aplicações que utilizam a interface E2 dependem dos E2SMs, enquanto a comunicação é gerenciada pelo E2AP. Este protocolo regula o gerenciamento da interface, configuração e conexão dos nós E2 ao Near-RT RIC. A conexão é estabelecida por meio do *Stream Control Transmission Protocol* (SCTP) e o E2AP oferece serviços RIC que podem ser combinados de diferentes formas para implementar os E2SMs [Polese et al. 2022b].

Entre os modelos de serviços padronizados pela O-RAN Alliance no contexto da solução apresentada, o principal utilizado no desenvolvimento da solução é o E2SM

Key Performance Measurement (KPM) [Alliance 2025c], responsável por reportar as métricas de desempenho da RAN via o Serviço de *Report* E2. Durante os procedimentos de configuração da interface E2, a O-CU/O-DU anuncia as métricas disponíveis para exposição. Uma xApp no Near-RT RIC pode então operar enviando uma mensagem de subscrição que pode especificar quais KPMs são de interesse e se o relatório deve ser periódico ou baseado em eventos específicos. Após esta etapa, a O-CU/O-DU usa mensagens de indicação E2 do tipo reporte para transmissão das KPMs selecionadas.

3. Trabalhos Relacionados

Segundo 3GPP [3GPP 2023] os *slices* de rede são considerados uma parte vital de sua arquitetura. Pois, cada *slice* é uma rede lógica de ponto a ponto que pode ser criada, dependendo das demandas da rede, em estruturas dedicadas à diferentes tipos de serviços. Existe uma vasta literatura que aborda *network slicing* para alcançar alto desempenho. Uma solução relevante nessa direção é apresentado por [Filali et al. 2023], a qual propõe uma abordagem de dois níveis para o *slicing* de RAN baseada na arquitetura O-RAN para alocar recursos de comunicação e computação RAN entre dispositivos finais para o tipo de serviço denominado URLLC. Este serviço tem como característica uma alta confiabilidade e baixa latência de 99,99% e 50 ms respectivamente, fornecidos principalmente por meio de recursos de computação de borda. Para cada nível de *slicing* da RAN, Filali et al. modelam o problema de gerenciamento de recursos como um processo de decisão de Markov de agente único e utiliza um algoritmo de aprendizado profundo por reforço para resolvê-lo.

Além disso, a solução apresentada por [Ajibare and Falowo 2019] apresenta a alocação de UEs com diferentes tipos de *slices* baseado na taxa de transferência de dados da UE. Porém, a solução foca em realizar a alocação baseada em prioridade de UE e não por tipo de serviço consumido no momento, priorizando serviços com maior taxa de dados. Por outro lado, o trabalho de [Gorla et al. 2021] apresenta a alocação considerando diferentes classes de *slices*, baseado no tipo de serviço 5G e requisitos de QoS, somado ao uso de blockchain para analisar a admissão. No entanto, a utilização de blockchain tem um impacto significativo quanto a latência da tomada de decisão, exigindo mais recursos computacionais e limitando a alocação de UEs devido aos contratos inteligentes e suas regras pré-definidas.

Outro estudo de [Polese et al. 2022a] apresenta uma solução baseada no ambiente O-RAN e utilizado aprendizado por reforço para otimizar os recursos de *slices* eMMB, URLLC e mIoT. Porém, esta solução apenas explora a otimização das características da *slice* do que a alocação, além do aprendizado por reforço requerer um maior custo temporal durante o processo de convergência. De forma parecida, o trabalho de [Labs 2022] também utiliza este ambiente O-RAN para implementar uma xApp para definição de políticas de distribuição de PRBs (Blocos de Recursos Físicos) em diferentes *slices*. Mas, esta solução apresenta alto custo operacional na definição e modificação de *slices* por meio de políticas, além de não realizar a alocação dinâmica de UEs em *slices*.

Com isso, grande parte dessas soluções dependem de políticas ou ações que requerem intervenção humana, com alto custo OPEX. Somado a isso, questões como latência e interoperabilidade precisam ser consideradas, visto que muitos estudos utilizam plataformas de hardware e software proprietárias e de difícil acesso. Dessa forma, este artigo propõe uma solução para alocação de UEs em *slices* 5G, utilizando projetos *open source* e analisando as métricas específicas dos tipos de serviços e *slices* em um ambiente real por meio do *testbed* OpenRAN@Brasil [Mauricio et al. 2025].

4. Arquitetura da Solução

Para projetar a arquitetura da solução foram definidos 3 módulos necessários integrados ao Near-RT RIC e ao 5GC: o Coletor de Métricas, o Codificador/Decodificador e o Classificador/Gerenciador de *Slices*. Como apresentado na Figura 2, estes módulos visam coletar as métricas de fluxos mIoT, URLLC e eMBB de UEs conectadas à infraestrutura da RAN, cujas respectivas funções de rede conectam-se ao Near-RT RIC. A partir disso, os módulos agem para classificar o tráfego dos UEs e alocá-las no 5GC ao *slice* que corresponde ao tipo de tráfego gerado. Estes módulos serão explicados nas subseções posteriores. A Seção 5 apresenta com mais detalhes os projetos e equipamentos que formam a infraestrutura O-RAN utilizada pela solução no *testbed* OpenRAN@Brasil.

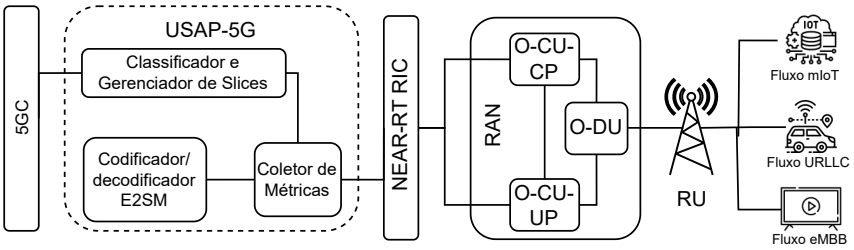


Figura 2. Visão Geral da Solução.

4.1. Coletor de Métricas

Este módulo consiste em uma aplicação O-RAN (xApp) acoplada ao Near-RT RIC. Como apresentado pela Figura 2, o Near-RT RIC apresenta duas interfaces SCTP, uma para comunicação com os nós da RAN, conectando-se à CU/DU, conhecido como E2 e uma conexão com o módulo coletor de métricas. A xApp realiza o envio de uma subscrição ao Near-RT RIC usando o modelo de serviço E2SM-KPM. O E2SM-KPM possui diversos estilos de reporte que são definidos pela O-RAN Alliance e se referem a como é estruturado a mensagem de subscrição e indicação entre a xApp e a O-CU/O-DU.

Para coleta de todas as métricas de UEs conectadas aos nós O-CU e O-DU, o estilo de reporte 4 foi utilizado, pois pode retornar as medições reportadas por uma O-CU ou O-DU especificada, baseado em uma condição comum ao nível de UE. Para subscrição do estilo 4 é necessário definir alguns parâmetros do processo de subscrição da xApp ao Near-RT RIC. Seguindo as especificações da ORAN Alliance. Esses parâmetros são: *Endpoint* da xApp, para retorno das informações de subscrição, O-CU/O-DU Node ID, o identificador do nó de destino da subscrição, o identificador da Função de RAN KPM e os detalhes da subscrição, uma estrutura de dados contendo detalhes da subscrição.

Na lógica de subscrição, é considerado que os nós O-CU e O-DU estejam com a conexão ativa com o Near-RT RIC. Em seguida, é utilizada a API do próprio RIC para obter os dados sobre os nós conectados. Caso o nó esteja conectado, é realizada a verificação se o mesmo possui suporte à função de RAN KPM. Caso o nó possua o suporte, é feita uma requisição para obtenção das métricas suportadas e os estilos de reposte disponíveis. Caso o estilo de reporte 4 esteja disponível, é feito o envio da subscrição para o recebimentos das métricas previamente reportadas pelo nó. A Figura 3, ilustra a lógica de subscrição feita pelo coletor de métricas para cada O-CU e O-DU conectada.

Desse modo, após a subscrição, a xApp está preparada para recebimento de métricas de todas as UEs reportadas por cada O-CU e O-DU. Para cada subscrição, foi definida uma condição de correspondência onde o valor de S-NSSAI (*Single – Network*

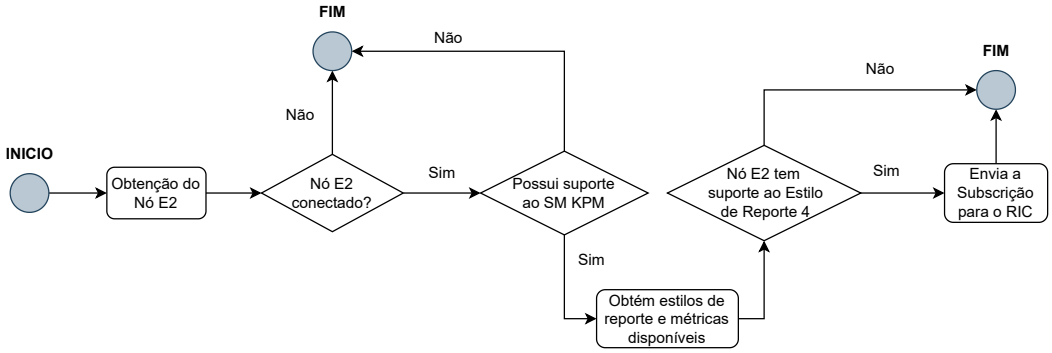


Figura 3. Lógica de Subscrição do Coletor de Métricas.

Slice Selection Assistance Information) corresponde ao SST (*Slice/Service Type*) de interesse, permitindo o isolamento e o monitoramento de UEs em cada *slice* da rede. Além disso, o formato de indicação das métricas coletadas é estruturado e obedece ao padrão definido pela O-RAN Alliance, sendo composto por um cabeçalho, contendo o *timestamp* do início do tempo de coleta, e a mensagem, composta por uma lista de medições por UE.

4.2. Codificador e Decodificador

Outra parte fundamental da solução é o codificador/decodificador, que funciona como um módulo para decodificar as mensagens enviadas pelo Near-RT RIC à xApp e codificar as mensagens de subscrição da xApp para enviar ao Near-RT RIC. Isto é possível por intermédio de um padrão de troca de mensagens entre RIC e os nós O-CU/O-DU. Esta troca funciona sobre o protocolo E2AP [Alliance 2025a], o qual precisa ser implementado tanto pelo Near-RT RIC quanto pelas Funções de Rede da RAN (CU e DU). Por meio deste protocolo, as NFs expõem os modelos de serviço disponíveis, bem como, utilizam para enviar mensagens de indicação para as xApps vinculadas.

Com relação à troca de mensagens entre a xApp e o Near-RT RIC, é utilizado a definição E2SM [Alliance 2025b] que consiste em um documento contém uma série de estruturas de dados para troca de mensagens entre as aplicações formatadas sob o formato ASN.1 (*Abstract Syntax Notation One*). Em cima dessa especificação é utilizado a codificação do tipo PER que é um padrão de codificação de dados binários utilizado para serializar e deserializar as estruturas de dados do formato ASN.1. Para o desenvolvimento deste módulo foram considerados os modelos ASN E2SM.asn e E2SM-KPM.asn contendo as estruturas de mensagens definidas pela O-RAN Alliance. A Figura 4 demonstra a lógica de funcionamento deste módulo.

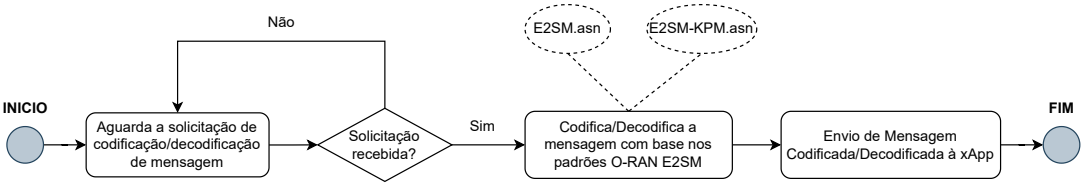


Figura 4. Lógica de Operação do Módulo Codificador/Decodificador.

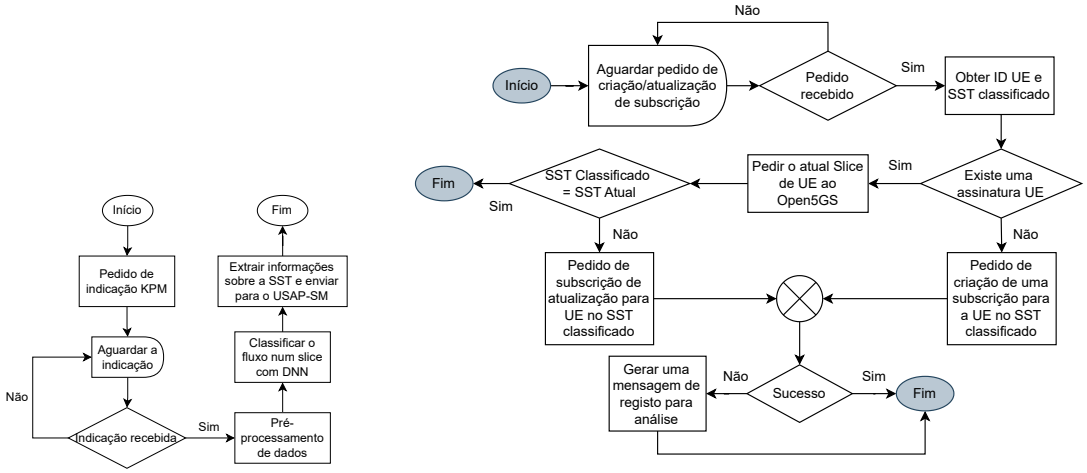
4.3. Classificador e Gerenciador de *Slices*

O Classificador e Gerenciador de *Slices* é um módulo que recebe as mensagens de reporte de métricas da xApp, que seguirão para o pré-processamento das indicações,

consistindo em processos como a seleção de métricas de entrada para o classificador, normalização/padronização das métricas e mapeamento do ID da UE no 5GC. Além disso, o módulo também cria uma conexão permanente com o banco de dados do 5GC para criação, atualização, remoção de sessões PDU (Unidade de Dados de Protocolo) de cada usuário conectado.

A saída do classificador contará com um rótulo do ID do *slice* de destino em que aquele tráfego foi classificado. O ID se refere ao SST no qual a UE deverá ser alocada no 5GC. Esta classificação passará por uma verificação junto ao 5GC para analisar se o SST obtido pela classificação se difere da *slice* atual. Caso esteja distinto, é realizada uma requisição de atualização da sessão PDU do usuário junto ao 5GC para alterar o SST da UE. Se tiverem iguais, o módulo ignora o SST classificado e mantém o SST atual cujo o UE está alocado.

Somado a isso, o módulo também é responsável por verificar continuamente junto ao Near-RT RIC quais as UEs que estão conectadas no banco de dados. Caso não haja nenhuma sessão PDU vinculada a uma UE específica no momento em que seu tráfego é classificado, é enviado uma solicitação de criação de sessão PDU para aquela UE que está entrando na rede. Isto possibilita que não apenas o processo de alocação seja automatizado, mas também, a criação de novas sessões PDU para UEs que estão se conectando na rede. As Figuras 5b e 5a ilustram, respectivamente, a lógica de ação do processo de classificação e de alocação da UE no *slice*.



(a) Lógica do Processo de Classificação.

(b) Diagrama de Ação do Módulo Gerenciador de *Slices*.

Figura 5. Lógicas de Classificação e de Alocação da UE no *slice*

4.3.1. Metodologia de Treinamento do Modelo de AM

Para o treinamento do modelo de classificação, foi utilizado o conjunto de dados ColOran [Polese et al. 2022a], obtido a partir de uma série de experimentos conduzidos no emulador de redes sem fio Colosseum. Este conjunto de dados possui aproximadamente 3,4 GB de tamanho e abrange mais de 73 horas de coleta de métricas relacionadas ao desempenho de RAN. Os experimentos foram configurados com 42 UEs em um cenário de *slicing* de redes, onde cada estação base foi configurada com três *slices* distintas: eMBB, mIoT e URLLC. As UEs foram alocadas dinamicamente entre essas *slices* com base no tipo

de tráfego que geravam, além disso, foi considerado uma *slice* adicional padrão, para os casos em que as taxas de transferências estivessem próximas a 0.

Para treinar o modelo foram consideradas quatro características, sendo elas a taxa de transferência de *downlink*, a taxa de transferência de *uplink* e a quantidade de *Physical Resource Blocks* (PRBs) de *uplink* e *downlink* alocados para cada UE. Essas foram as características possíveis de se utilizar devido a correspondência das métricas coletadas utilizando o SRSRAN com as métricas do dataset ColOran e são cruciais para o modelo poder identificar a *slice* mais adequada para a alocação das UEs.

A arquitetura escolhida para o classificador foi uma rede neural do tipo *Long Short-Term Memory* (LSTM), a qual é especialmente eficaz no tratamento de dados sequenciais e que possuem dependências temporais. Esse tipo de modelo pode capturar relações nas quais a saída depende não apenas da entrada atual, mas também de entradas anteriores — uma característica que reflete o comportamento dinâmico das conexões das UEs ao longo do tempo. Após a camada LSTM, foi adicionada uma camada densa para aprimorar a capacidade de aprendizado do modelo, permitindo a captura de relações mais complexas entre as saídas da LSTM e as previsões finais.

Além disso, visando maximizar o desempenho da rede neural, foi realizado um processo de ajuste de hiperparâmetros, onde foram testadas diversas combinações de valores para parâmetros críticos, buscando identificar as configurações ideais para o treinamento. Para obter as configurações de hiperparâmetros ideais foi adotada a ferramenta *Hyperband Tuner* [Li et al. 2016], onde os valores demonstrados na Tabela 1 foram utilizados para encontrar a melhor combinação de hiperparâmetros para o treinamento da rede neural.

Tabela 1. Configurações de Hiperparâmetros.

Hiperparâmetro	**Valores**
Unidades LSTM	32, 48, 64, 80, 96, 112, 128
Função de Ativação	`relu`, `tanh`
Taxa de Aprendizado	0.01, 0.001, 0.0001

Para avaliar o desempenho do classificador, foram utilizadas as métricas Precisão, *Recall* e Acurácia. Como o problema envolvia quatro classes (default=128, eMBB=1, mIoT=2, URLLC=3), cada uma representando um tipo de *slice*, essas métricas foram calculadas individualmente para cada classe. Além disso, foi considerada a média das somas das métricas para obter uma visão geral do desempenho do modelo em todas as classes. Essa abordagem permite balancear o impacto de classes com diferentes distribuições e garantir que o modelo seja avaliado de forma justa em relação à capacidade de distinguir corretamente entre as diferentes *slices*.

5. Cenário de Validação

O cenário de validação da solução foi montado utilizando a infraestrutura do *testbed* OpenRAN@Brasil [RNP et al. 2025], no site da Rede Nacional de Ensino e Pesquisa (RNP). Neste, foram utilizados dois servidores denominados O-RAN Cloud 2 e O-RAN Cloud 5. Em ambos os servidores foram instalados um kernel *realtime* recomendados para a execução das funções de RAN CU/DU e do núcleo 5G (5GC). Além disso, foram instanciados dois *clusters* Kubernetes para cada um dos servidores, sendo este requisito para a instalação dos *containers* que envolvem a RAN, 5GC e Near-RT RIC. As especificações detalhadas dos servidores O-RAN Cloud 2 e O-RAN Cloud 5 são descritas na Tabela 2.

Para a implementação das funções de RAN CU/DU, foi utilizado o SRSRAN [SRS 2025] da SRS (*Software Radio Systems*). O SRSRAN é um projeto de código

Tabela 2. Especificações Servidores O-RAN.

Servidores	O-RAN Cloud 2	O-RAN Cloud 5
CPU Intel(R) Xeon(R)	Silver 4314 CPU @ 2.40GHz	Gold 6348 CPU @ 2.60GHz
Memória	128 GB	256G
Armazenamento	500 GB	500 GB
S.O	Ubuntu 22.04.5 LTS	Ubuntu 24.04.1 LTS
Kernel	5.15.0-1073-realtime	6.8.1-1014-realtime

aberto que implementa as funções de RAN CU (Unidade Centralizada) e DU (Unidade Distribuída), como também, suporta a arquitetura Open RAN por meio da compatibilidade com conexões E2, suportando os modelos de serviço E2SM-KPM e E2SM-RC no protocolo E2AP. Para a integração junto ao SRSRAN, foi implementado uma xApp na linguagem Golang que realiza a função do coletor de métricas descrito na Subseção 4.1.

O Near-RT RIC utilizado para a integração do módulo de coleta de métricas foi o desenvolvido pela O-RAN Software Community (OSC). A OSC é uma colaboração entre a O-RAN Alliance e a Linux Foundation com a missão de apoiar a criação de software para a RAN [OSC 2025]. Nesse sentido, a OSC disponibiliza uma implementação do Near-RT RIC denominada RICPLT, cujo *framework* em Golang foi utilizado para a integração entre o Near-RT RIC e o coletor de métricas, bem como, para o processo de subscrição no modelo de serviço E2SM-KPM, disponibilizado pelo SRSRAN.

Para o 5GC foi utilizado o Open5GS [Open5GS 2025], o qual é um projeto de código aberto que implementa as principais funções responsáveis pelo acesso, autenticação e troca de dados do Equipamento do Usuário (UE) com a rede. O Open5GS oferece suporte a implementações em redes privadas e Funções de Rede (NFs) baseadas na *Release 17* do 3GPP, com capacidade para *slicing*. Ademais, para a integração junto ao Open5GS, foi implementado um cliente HTTP na linguagem Python que se conecta ao seu banco de dados, seguindo o funcionamento descrito na Subseção 4.3.

No que diz respeito a configuração e ao provisionamento dos *slices*, estes foram pré-configurados e pré-alocados no Open5GS e no SRSRAN. O cenário contou com três tipos de *slices* padrão definidos pelo 3GPP, sendo eles eMBB, URLLC e mIoT, e um *slice* denominado *default* para classes de tráfego que não se enquadram em nenhum dos outros três. A Tabela 3 mostra a configuração utilizada para cada um dos slices que foram provisionados no 5GC e na RAN.

Tabela 3. Configurações dos *Slices* na Infraestrutura

Tipo	SST	SD	DNN	Subnet	MCC	MNC
eMBB	1	0xFFFFFF	embb	10.45.0.0/24	001	01
URLLC	2	0xFFFFFF	urllc	10.45.1.0/24	001	01
mIoT	3	0xFFFFFF	miot	10.45.2.0/24	001	01
default	128	0xFFFFFF	default	10.45.3.0/24	001	01

Por fim, no que diz respeito aos equipamentos de borda do *testbed*, foram utilizados uma O-RU RPQN-7801E da Foxconn, juntamente com dois celulares Samsumg Galaxy S23 Ultra. A O-RU foi configurada para o modo de operação conforme os parâmetros mostrados na Tabela 4. Ademais, estes equipamentos foram cedidos e pré-configurados com suporte da RNP e foram alocados em um ambiente *indoor* dentro do *testbed* OpenRAN@Brasil. A Figura 6 ilustra o ambiente montado para a validação da solução.

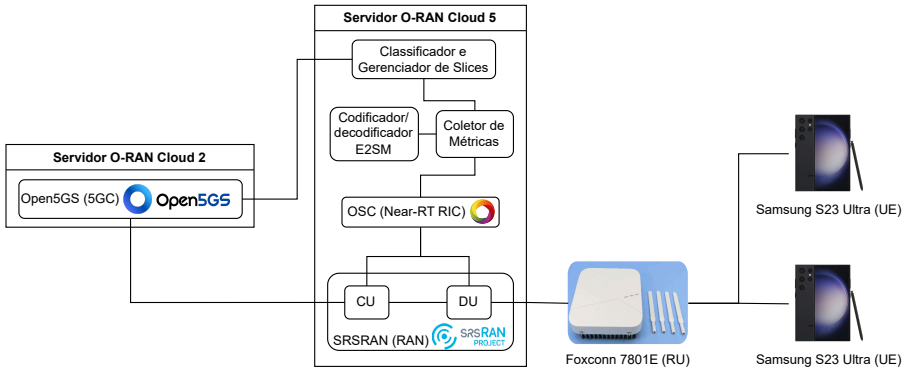


Figura 6. Visão Geral da Solução.

Tabela 4. Modo de Operação O-RU Foxconn.

O-RU Foxconn RPQN-7801E	
Frequência de Operação	3.74 GHz
Largura de Banda	100 MHz
Nº PRBs	273
Modo de Transmissão	4T4R MIMO
Espaçamento entre Subportadora	30 KHz

6. Resultados

Os resultados colhidos durante o experimento feito no *testbed* OpenRAN@Brasil, consideraram duas principais análises da solução. A primeira análise considera a capacidade do modelo de rede neural LSTM de acertar as classes de serviço/slice corretas para cada tipo de entrada de tráfego, considerando os parâmetros utilizados durante o processo de treinamento descrito em 4.3.1. A segunda análise considera as métricas de latência durante cada processo da solução utilizando o cenário de validação ilustrado na Figura 6.

6.1. Análise da Rede Neural LSTM

Para a análise da Rede Neural LSTM foi utilizado um conjunto de dados de testes retirados do dataset do ColORAN que não foi repassada para o modelo durante o processo de treino e validação. Foram analisados um total de 6725 instâncias de tráfego, considerando as 4 métricas utilizadas durante o processo de treinamento. As métricas analisadas foram: a precisão que mede a proporção de previsões de uma determinada classe pelo modelo que estão corretas. O *recall* que mede a proporção dos verdadeiros positivos identificados pelo modelo de uma determinada classe em relação ao total. O F1 Score, o qual é a média harmônica entre precisão e *recall*, para verificar o balanceamento do modelo considerando ambas as métricas. Por fim, a acurácia, que mede a proporção de previsões corretas em relação ao total de previsões feitas.

Tabela 5. Avaliação da Rede Neural LSTM

Classe de *Slice*	**Acurácia**	**Precisão**	***Recall***	***F1 Score***
1 (eMBB)	98.98%	99.71%	98.98%	99.34%
2 (URLLC)	97.93%	96.71%	97.93%	97.32%
3 (mIoT)	97.93%	98.54%	97.93%	98.23%
128 (*default*)	100%	99.96%	100%	99.98%

Os resultados da Tabela 5 mostram que o modelo teve um bom desempenho considerando as 4 classes de serviço/*slice* analisadas. Isso se deve devido a uma série de fatores

de otimizações feitas durante o treinamento do modelo. O treinamento em blocos, devido às características da rede neural LSTM, acabou beneficiando muito a análise do modelo com relação às novas instâncias de dados. Por outro lado, o processo de otimização dos hiperparâmetros feito durante o processo de construção da rede neural com a ferramenta *Hyperband Tuner*, possibilitou um melhor desempenho durante a etapa de aprendizado.

Além disso, pôde-se observar que as classes *default* e eMBB foram as que obtiveram as melhores avaliações de precisão de classificação. Isso se deve também ao comportamento mais determinístico desses tipos de classes de tráfego, sendo que os serviços *default*, considera-se a ociosidade do tráfego, com as taxas de *uplink*, *dowlink* e utilização de PRBs tendendo a 0. Por outro lado, os serviços eMBB considera-se a alta taxa de *uplink* e *downlink*, juntamente com o alto consumo de PRBs. Esse comportamento auxiliou o modelo a criar padrões de tráfego mais bem definidos durante o processo de treinamento, bem como, ajudou no processo de classificação do modelo durante o seu funcionamento.

Em resumo, o modelo aprendizado de máquina utilizando a rede neural LSTM mostrou-se bem eficaz durante o processo de classificação das classes de *slice*/serviço. Além disso, o processo envolvendo a diferenciação de cada classe de tráfego se mostrou bem eficaz, devido à análise em blocos feita pelo modelo. Com isso, o método envolvendo a classificação utilizando a rede neural LSTM pode ser validado e mostrou-se eficaz para o problema de classificação abordado nesta pesquisa.

6.2. Análise da Latência de Operação da Solução

Outro resultado analisado foi a latência obtida em cada processo da solução durante o processo de execução no *testbed* OpenRAN@Brasil. Para este processo foram coletados 5000 mensagens de indicação de fluxo de tráfego dos dispositivos conectados durante o período de aproximadamente 1 hora e 38 minutos. O período de reporte e de granularidade das métricas foram definidos para 1 segundo, sendo estes definidos durante o processo de subscrição da xApp no modelo de serviço E2SM-KPM.

A primeira medida de latência analisada diz respeito a latência de reporte do coletor de métricas. Esta latência corresponde ao tempo decorrido para o recebimento de uma indicação da CU/DU até a xApp. Ela é calculada na xApp, utilizando o *timestamp* contido no cabeçalho da mensagem de indicação E2AP. Após o recebimento da indicação, é feito o cálculo da diferença do *timestamp* gerado no momento do recebimento da mensagem de reporte pelo *timestamp* enviado pela CU/DU. O resultado disso é a latência de reporte das métricas da RAN para a xApp.

A segunda medida de latência analisada foi a latência do servidor de métricas. Essa latência considera o tempo decorrido de envio das métricas da xApp para módulo Classificador e Gerenciador de *Slices* (CGS), descrito na Subseção 4.3. Para o processo de cálculo dessa latência, foi gerado um *timestamp* de envio na xApp e um *timestamp* de recebimento no CGS. O resultado considera a latência de envio das mensagens de reporte, por meio de um servidor gRPC implementado no coletor de métricas (xApp), para o CGS.

Outras medidas de latência analisadas foram a latência de classificação e a latência de alocação do UE no 5GC. A latência de classificação considera o tempo que o modelo leva para classificar o tráfego a partir de 1 instância de dados recebida do servidor de métricas. Por outro lado, a latência de inferência ou alocação considera o tempo que leva para alocar uma UE em uma *slice* no 5GC. A Figura 7 mostra os resultados obtidos durante cada processo de funcionamento da solução.

Conforme mostrado na Figura 7, os resultados de latência de funcionamento no cenário emulado se mostraram de acordo com os requisitos de desempenho definidos pela O-RAN Alliance (10ms até 1s), ficando estes bem abaixo do que é permitido. A maior

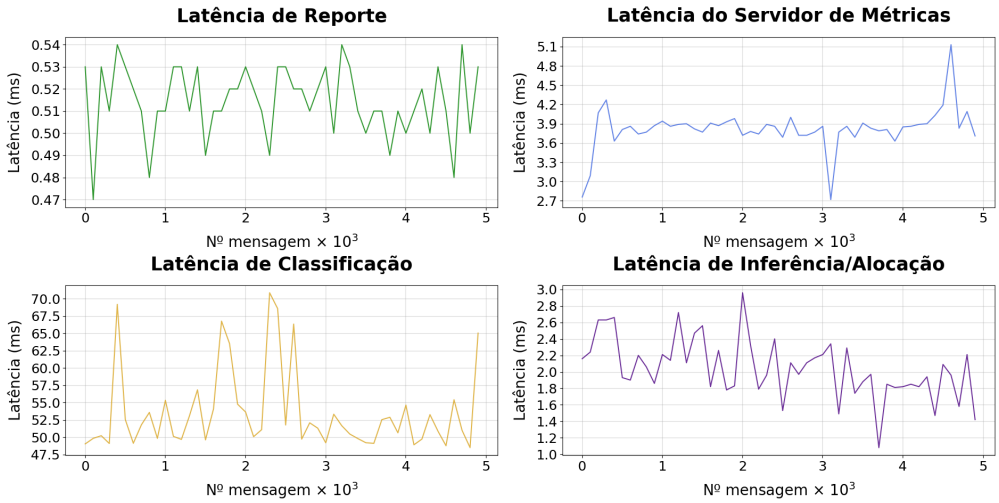


Figura 7. Medição da Latência em Cada Etapa da Solução.

latência considerada foi a do processo de classificação que dado a robustez do modelo de aprendizado de máquina, envolvendo a rede neural LSTM, já era considerado esperado durante a modelagem da solução. Todavia, em geral, a solução se mostrou bem eficaz no que diz respeito ao tempo de resposta total que envolve os processos de recebimento, processamento, classificação e alocação da UE na *slice*. A Figura 8 demostra o gráfico considerando a latência total da solução, considerando a soma de todas as latências descritas anteriormente.

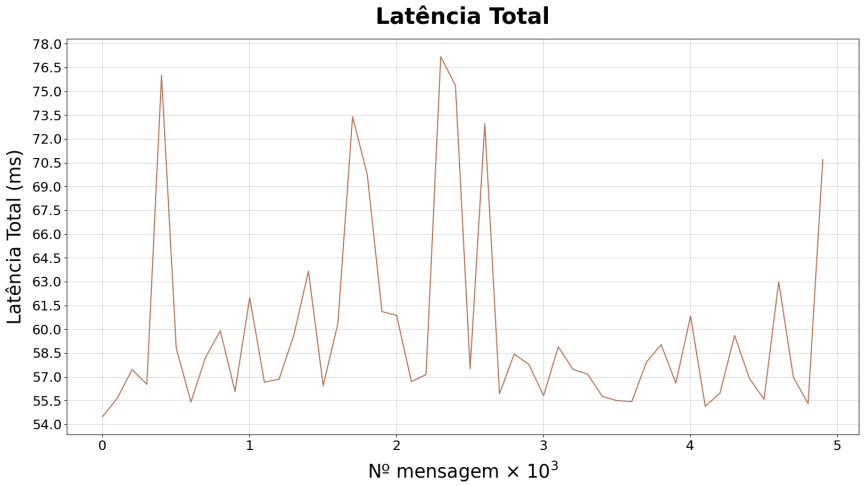


Figura 8. Medição da Latência Total.

7. Conclusão e Trabalhos Futuros

Este trabalho apresentou uma solução para a Alocação Dinâmica de Equipamentos de Usuários em *Slices* 5G Utilizando Arquiteturas O-RAN. A solução desenvolvida considerou uma modelagem modular implementada utilizando o conceito de microsserviços para funcionamento em um ambiente *cloud-native*. Além disso, os resultados obtidos mostraram a eficácia da solução na integração com uma rede O-RAN real, fornecida pela RNP, e também contribuiu para os resultados do Grupo de Trabalho IQoS (GT-IQoS) no *testbed*

OpenRAN@Brasil. Além disso, a solução abordou e validou a utilização do aprendizado de máquina para o problema que envolve a alocação de um usuário em uma *slice* de rede 5G, considerando para isso as diferentes classes de tráfego padronizadas pelo 3GPP. Por outro lado, a solução desenvolvida conseguiu integrar com êxito o modelo de aprendizado de máquina no contexto da arquitetura O-RAN, utilizando uma rede neural LSTM e demonstrando a viabilidade do emprego de soluções envolvendo inteligência artificial no contexto de redes 5G e 6G abertas. Além disso, a solução se mostrou viável e eficiente no que diz respeito ao proposito de melhorar a eficiência de alocação das UEs nas *slices* corretas, como também, com a utilização do modelo de AM, possibilitou uma análise e tomada de decisão mais assertiva durante o processo de alocação no 5GC. Por fim, as métricas de latência analisadas, demostraram que mesmo com a utilização da rede neural LSTM, a solução se mostrou eficiente e com um tempo de resposta nos padrões definidos pela O-RAN Alliance. O repositório do projeto encontra-se disponível em USAP-5G - GitHub, onde podem ser encontrados os códigos implementados para cada módulo da solução, bem como, o *dataset* e os modelos de aprendizado de máquina utilizados.

Como trabalhos futuros, estima-se a melhoria dos componentes da solução para suportar mais de um estilo de reporte KPM. Além disso, o desenvolvimento de um codificador/decodificador na linguagem C que possar ser mais integrada à xApp de coleta de métricas, essa é uma melhoria que eliminaria a necessidade de se ter uma interface gRPC e um módulo específico para esta tarefa. Por outro lado, um estudo de caso da escalabilidade da plataforma poderia ser feito, a partir da adição de uma cenário com mais UEs. Por fim, a utilização do Modelo de Serviço E2SM-RC para realizar a alocação de PRBs individuais para cada UE, conforme o resultado do modelo de classificação de serviço.

Agradecimentos

Este trabalho foi parcialmente financiado pelo Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) projeto 444978/2024-0, pela Fundação Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), e pela Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP) projeto 2023/00811-0, projeto 2023/00673-7, projeto 2021/00199-8 (CPE SMARTNESS), projeto 2020/04031-1, e projeto 2018/23097-3. Ao OpenRAN@Brasil em parceria com a Rede Nacional de Ensino e Pesquisa (RNP) e Ministério de Ciência e Tecnologia (MCTI) projeto 01245.014203/2021-14.

Referências

3GPP (2023). 5G Network slice management. `https://www.3gpp.org/technologies/slice-management`. [acessado: 30 de janeiro de 2025].

Afolabi, I., Taleb, T., Samdanis, K., Ksentini, A., and Flinck, H. (2018). Network slicing and softwarization: A survey on principles, enabling technologies, and solutions. *IEEE Communications Surveys Tutorials*, 20(3):2429–2453.

Ajibare, A. T. and Falowo, O. E. (2019). Resource allocation and admission control strategy for 5g networks using slices and users priorities. In *2019 IEEE AFRICON*, pages 1–6.

Alliance, O. (2023). O-RAN Specifications. `https://www.o-ran.org/specifications`. [acessado: 30 de outubro de 2023].

Alliance, O.-R. (2025a). O-ran e2 application protocol (e2ap) 6.0. Relatório Técnico O-RAN.WG3.E2AP-R004-v06.00, O-RAN Alliance. Acessado em 13 de janeiro de 2025.

Alliance, O.-R. (2025b). O-ran e2 service model (e2sm) 6.0. Relatório Técnico O-RAN.WG3.E2SM-R004-v06.00, O-RAN Alliance. Acessado em 13 de janeiro de 2025.

Alliance, O.-R. (2025c). O-ran e2 service model (e2sm) kpm 5.0. Relatório Técnico O-RAN.WG3.E2SM-KPM-R003-v05.00, O-RAN Alliance. Acessado em 13 de janeiro de 2025.

Donatti, A., Correa, S. L., Martins, J. S. B., Abelem, A., Both, C. B., Silva, F., Suruagy, J. A., Pasquini, R., Moreira, R., Cardoso, K. V., and Carvalho, T. C. (2023). Survey on machine learning-enabled network slicing: Covering the entire life cycle. *IEEE Transactions on Network and Service Management*, pages 1–1.

Filali, A., Nour, B., Cherkaoui, S., and Kobbane, A. (2023). Communication and computation o-ran resource slicing for urllc services using deep reinforcement learning. *IEEE Communications Standards Magazine*, 7(1):66–73.

Gorla, P., Chamola, V., Hassija, V., and Niyato, D. (2021). Network slicing for 5g with ue state based allocation and blockchain approach. *IEEE Network*, 35(3):184–190.

Labs, R. (2022). Quality of Service-based Resource Allocator xApp (QRA-xApp). `https://rimedolabs.com/wp-content/uploads/2022/06/ORAN-QRA-xApp-TechSpec.pdf`. [acessado: 03 de novembro de 2023].

Li, L., Jamieson, K. G., DeSalvo, G., Rostamizadeh, A., and Talwalkar, A. (2016). Efficient hyperparameter optimization and infinitely many armed bandits. *CoRR*, abs/1603.06560.

Marinova, S. and Leon-Garcia, A. (2024). Intelligent o-ran beyond 5g: Architecture, use cases, challenges, and opportunities. *IEEE Access*, 12:27088–27114.

Mauricio, W., Costa Neto, F. H., Silva, M., Aoki, R. K. Y., Melão, E., Barros, S., Farias, F., Bondan, L., and Abinader, F. M. (2025). Assessing nationwide distributed open ran deployments with the openran@brasil testbed. *IEEE Communications Magazine*, 63(2):46–52.

Open5GS (2025). Open Source implementation for 5G Core and EPC, i.e. the core network of LTE/NR network (Release-17).

OSC (2025). *About O-RAN Software Community (SC)*. O-RAN Alliance. Disponível no site oficial da O-RAN Alliance.

Polese, M., Bonati, L., D'Oro, S., Basagni, S., and Melodia, T. (2022a). Colo-ran: Developing machine learning-based xapps for open ran closed-loop control on programmable experimental platforms.

Polese, M., Bonati, L., D'Oro, S., Basagni, S., and Melodia, T. (2022b). Understanding o-ran: Architecture, interfaces, algorithms, security, and research challenges.

RNP et al. (2025). OpenRAN@Brasil. `https://openranbrasil.org.br/`.

SRS (2025). Open-source and ORAN-native 5G CU/DU with a complete stack from I/Q to IP from SRS.