

UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

Fatiamento inteligente em redes B5G para suporte a aplicações ITS

Tiago do Vale Saraiva

Orientador

Dr. Carlos Alberto Vieira Campos

RIO DE JANEIRO, RJ - BRASIL
MARÇO DE 2026

Fatiamento inteligente em redes B5G para suporte a aplicações ITS

TIAGO DO VALE SARAIVA

TESE APRESENTADA AO PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA DA UNIVERSIDADE FEDERAL DO ESTADO DO RIO DE JANEIRO (UNIRIO) COMO REQUISITO PARCIAL PARA A OBTENÇÃO DO TÍTULO DE DOUTOR EM INFORMÁTICA. APROVADA PELA COMISSÃO EXAMINADORA ABAIXO ASSINADA.

Aprovada por:

Documento assinado digitalmente
gov.br
CARLOS ALBERTO VIEIRA CAMPOS
Data: 22/06/2026 15:45:47-0300
Verifique em https://validar.iti.gov.br

Dr. CARLOS ALBERTO VIEIRA CAMPOS — UNIRIO (Presidente / Orientador)

Documento assinado digitalmente
gov.br
SIDNEY CUNHA DE LUCENA
Data: 20/06/2026 22:40:47-0300
Verifique em https://validar.iti.gov.br

Dr. SIDNEY CUNHA DE LUCENA — UNIRIO

Documento assinado digitalmente
gov.br
JEFFERSON ELBERT SIMOES
Data: 01/07/2026 18:58:40-0300
Verifique em https://validar.iti.gov.br

Dr. JEFFERSON ELBERT SIMÕES — UNIRIO

Documento assinado digitalmente
gov.br
CHRISTIAN RODOLFO ESTEVE ROTHENBERG
Data: 23/06/2026 08:03:31-0300
Verifique em https://validar.iti.gov.br

Dr. CHRISTIAN ESTEVE ROTHENBERG — UNICAMP

Documento assinado digitalmente
gov.br
JOSE FERREIRA DE REZENDE
Data: 24/06/2026 08:35:43-0300
Verifique em https://validar.iti.gov.br

Dr. JOSÉ FERREIRA DE REZENDE — UFRJ

RIO DE JANEIRO, RJ - BRASIL
MARÇO DE 2026.

Catalogação informatizada pelo(a) autor(a)

S243 Saraiva, Tiago do Vale
Fatiamento inteligente em redes B5G para suporte a aplicações ITS / Tiago do Vale Saraiva. -- Rio de Janeiro : UNIRIO, 2026.
157 f.

Orientador: Carlos Alberto Vieira Campos.
Tese (Doutorado) - Universidade Federal do Estado do Rio de Janeiro, Programa de Pós-Graduação em Informática, 2026.

1. Fatiamento de Rede. 2. Sistemas Inteligentes de Transporte. 3. Aprendizado Federado. I. Campos, Carlos Alberto Vieira, orient. II. Título.

Dedico à minha família, pelo apoio incondicional e compreensão ao longo de toda a caminhada.

AGRADECIMENTOS

Ao Prof. Dr. Carlos Alberto Vieira Campos, meu orientador, pela confiança e orientação dedicada ao longo de todo o doutorado. Sua disposição em acompanhar este trabalho, inclusive nos momentos mais difíceis do percurso, foi fundamental.

Aos professores Dr. Sidney Cunha de Lucena, Dr. Christian Esteve Rothenberg e Dr. José Ferreira de Rezende, que acompanham minha trajetória acadêmica desde o mestrado, pelas contribuições ao longo desses anos.

Ao Prof. Dr. Jefferson Elbert Simões, pelas observações criteriosas que contribuíram para o aprimoramento da versão final da tese.

Ao Programa de Pós-Graduação em Informática da UNIRIO, pela estrutura e pela oportunidade de realização deste doutorado.

Aos colegas que acompanharam esta caminhada, pelas discussões, pela troca de ideias e pelo apoio nos diversos momentos ao longo desses anos.

Aos colegas e gestores da Petróleo Brasileiro S.A. (Petrobras), pela compreensão nos momentos em que foi necessário conciliar as atividades profissionais com as exigências da pesquisa.

À minha família, pela compreensão das ausências e pelo apoio constante. Em especial aos meus pais, a quem dedico este trabalho.

SARAIVA, TIAGO DO VALE. **Fatiamento inteligente em redes B5G para suporte a aplicações ITS**. UNIRIO, 2026. 157 páginas. Tese de Doutorado. Departamento de Informática Aplicada, UNIRIO.

RESUMO

O crescimento das aplicações de Sistemas Inteligentes de Transporte (*Intelligent Transportation Systems* — ITS) impõe requisitos heterogêneos de latência, vazão e confiabilidade às redes de comunicação móvel. O fatiamento de rede (*network slicing*), consolidado com o 5G e mantido como elemento central nas redes Beyond-5G (B5G), permite isolar logicamente recursos para diferentes serviços, porém o gerenciamento do ciclo de vida das fatias em cenários veiculares permanece um problema em aberto, dada a variação de carga, as mudanças de topologia e a heterogeneidade das condições de canal. Esta tese propõe e avalia abordagens para o gerenciamento inteligente de fatias em redes B5G voltadas a aplicações ITS, organizadas em uma narrativa progressiva com sete contribuições. Duas abordagens de aprendizado supervisionado identificam o contexto veicular a partir de dados públicos de mobilidade urbana e de estatísticas da rede de comunicação. Um *framework* de fatiamento baseado em SDN orquestra recursos de forma dinâmica segundo os requisitos de quatro classes de aplicação ITS, alcançando taxa de entrega de pacotes (PDR) de 85,1% para a aplicação prioritária, contra 50,1% da abordagem de referência. O *framework* AIMS (*Adaptive and Intelligent Management of Slicing*) introduz uma arquitetura híbrida *cloud-edge* com classificação de impacto de políticas de fatiamento em quatro níveis semânticos; modelos *ensemble* leves atingem 94,8% de acurácia na tradução de telemetria veicular em sinais acionáveis. Uma extensão com aprendizado federado (*Federated Learning* — FL) distribui o treinamento entre controladores locais associados a *Road-Side Units*, com desempenho federado próximo ao centralizado (Macro F1-*score* $> 0{,}93$ em cenário IID e 93,9% de acurácia em cenário Non-IID). Uma investigação de segurança revela que, sem defesa, ataques de envenenamento direcionado alcançam 100% de misclassificação de fatias críticas, enquanto estratégias de agregação Bizantino-robusta (Krum e Trimmed Mean) mantêm a integridade da classificação (C2A $= 0\%$ com Krum). Os resultados demonstram a viabilidade de integrar inteligência na borda ao ciclo de vida das fatias, desde a identificação de contexto até a proteção contra participantes maliciosos.

Palavras-chave: Fatiamento de Rede; Sistemas Inteligentes de Transporte; Aprendizado Federado; Segurança da Informação; B5G/6G; Inteligência na Borda.

SARAIVA, TIAGO DO VALE. **Intelligent slicing in B5G networks to support ITS applications**. UNIRIO, 2026. 157 pages. Doctoral Thesis. Department of Applied Informatics, UNIRIO.

ABSTRACT

The growth of Intelligent Transportation Systems (ITS) applications imposes heterogeneous latency, throughput, and reliability requirements on mobile communication networks. Network slicing, consolidated in 5G and retained as a core element in Beyond-5G (B5G) networks, enables logical resource isolation for different services. However, managing the slice lifecycle in vehicular scenarios remains an open problem due to load variations, topology changes, and heterogeneous channel conditions. This thesis proposes and evaluates approaches for intelligent slice management in B5G networks supporting ITS applications, organized as a progressive narrative comprising seven contributions. Two supervised learning approaches identify vehicular context from public urban mobility data and from communication network statistics. An SDN-based slicing framework dynamically orchestrates resources according to the requirements of four ITS application classes, achieving a Packet Delivery Ratio (PDR) of 85.1% for the highest-priority application, compared with 50.1% for the baseline approach. The AIMS framework (Adaptive and Intelligent Management of Slicing) introduces a hybrid cloud-edge architecture with four-level semantic impact classification of slicing policies; lightweight ensemble models reach 94.8% accuracy in translating vehicular telemetry into actionable signals. A Federated Learning (FL) extension distributes training across local controllers associated with Road-Side Units, with federated performance close to centralized training (Macro F1-score $>$ 0.93 under IID conditions and 93.9% accuracy under Non-IID conditions). A security investigation reveals that, without defenses, directed poisoning attacks achieve 100% misclassification of critical slices, whereas Byzantine-robust aggregation strategies (Krum and Trimmed Mean) preserve classification integrity (C2A Rate = 0% with Krum). The results demonstrate the feasibility of integrating edge intelligence into the slice lifecycle, from context identification to protection against malicious participants.

Keywords: Network Slicing; Intelligent Transportation Systems; Federated Learning; Information Security; B5G/6G; Edge Intelligence.

LISTA DE ILUSTRAÇÕES

Figura 1 – Serviços-chave, aplicações e roteiro de evolução das redes móveis até o 6G, com destaque para o posicionamento desta tese na transição B5G. 29
Figura 2 – Visão de ecossistema das redes 6G, com arquitetura integrada espaço-ar-solo e principais casos de uso previstos 30
Figura 3 – Soluções de IA em ITS divididas em categorias. 32
Figura 4 – Arquitetura proposta para identificação de contexto veicular utilizando dados públicos 43
Figura 5 – Fluxograma da abordagem proposta para identificação de contexto veicular utilizando dados públicos 43
Figura 6 – Duas regiões na cidade do Rio de Janeiro de onde os dados foram analisados - extraídos do OpenStreetMap. 45
Figura 7 – Velocidade média por hora 46
Figura 8 – Número de registros de GPS por hora 46
Figura 9 – Número de linhas (rotas de ônibus) 47
Figura 10 – Número de ônibus por hora 47
Figura 11 – Resultados consolidados obtidos com os modelos de ML avaliados (LR, KNN-3, KNN-5, DT, RF e SVM) para acurácia, precisão, revocação e F1-*score*. 49
Figura 12 – Abordagem proposta com o uso de dados estatísticos de operação da rede veicular. 55
Figura 13 – Cenário de avaliação representando um congestionamento de tráfego com diferentes níveis de consumo de recursos em cada RSU ao longo do tempo. 56
Figura 14 – Medições de *throughput* e RTT coletadas na infraestrutura MEC durante a implementação do cenário de congestionamento de tráfego em 3 rodadas de emulação. 57
Figura 15 – Resultados consolidados obtidos com os modelos LR, KNN-3, KNN-5, DT, RF e SVM para as métricas de acurácia, precisão, *recall* e F1-*score*. 59
Figura 16 – Taxa de erro em função de k com os dados de estatísticas coletadas na rede de comunicação no cenário veicular avaliado. 59
Figura 17 – Arquitetura do framework proposto com fatiamento de rede para atender requisitos de aplicações ITS. 66
Figura 18 – Diagrama de fluxo do framework em função de mudanças na topologia, aplicações ou recursos de infraestrutura. 68

Figura 19 – Cenário de avaliação representando um congestionamento em uma via urbana densa. 75
Figura 20 – Resultados de *throughput* e RTT durante o tempo de avaliação para as aplicações E, $E2$, G e S, utilizando as diferentes abordagens avaliadas. 80
Figura 21 – Taxa de entrega de pacotes para as diferentes aplicações nas abordagens avaliadas. 82
Figura 22 – ECDFs dos resultados de RTT para as diferentes aplicações utilizando as abordagens avaliadas. 83
Figura 23 – Arquitetura híbrida do AIMS ao longo das quatro fases do ciclo de vida: Preparação (Painel 1), Comissionamento (Painel 2), Operação (Painel 3) e Descomissionamento (Painel 4). As funções F1–F7 e os fluxos de comunicação entre o Controlador Central (CC) e os Controladores Locais (LCs) são indicados em cada painel. 89
Figura 24 – F1-score por nível de impacto para os três classificadores. O TabNet apresenta degradação acentuada no nível *Warning* (F1 = 0,550), enquanto Random Forest e CatBoost mantêm desempenho superior a 0,88 em todos os níveis. 96
Figura 25 – Matriz de confusão normalizada — Random Forest. Confusão *Warning*→*Severe* de aproximadamente 14%. 97
Figura 26 – Matriz de confusão normalizada — CatBoost. Confusão *Warning*→*Severe* de aproximadamente 14%. 97
Figura 27 – Matriz de confusão normalizada — TabNet. Confusão *Warning*→*Severe* de aproximadamente 40%, evidenciando o colapso na fronteira entre os níveis intermediários. 98
Figura 28 – Acurácia de teste versus tempo total de treinamento (escala logarítmica). O CatBoost alcança a maior acurácia (94,8%) em 169 s. O Random Forest é o mais rápido (68 s) com acurácia competitiva (94,4%). O TabNet apresenta a pior relação custo-efetividade: 3.947 s para 82,9% de acurácia (58,1× mais lento que o RF). 99
Figura 29 – Comparação entre F1-Macro em validação cruzada (CV) e no conjunto de teste retido (*holdout*). O TabNet apresenta o maior gap de generalização (0,102), evidenciando sensibilidade do modelo à otimização de hiperparâmetros. 100
Figura 30 – Estabilidade da acurácia de teste em função do número de avaliações de HPO. O Random Forest mantém 94,4% em todos os orçamentos. O CatBoost varia entre 93,2% e 94,8%, atingindo o melhor resultado em $T = 40$. O TabNet estabiliza em 82,9% a partir de $T = 40$. 101

Figura 31 – Gerenciamento fim a fim do ciclo de vida de fatias ITS com suporte de aprendizado federado centralizado. Os controladores locais nas RSUs treinam o classificador de impacto com dados de monitoramento locais e enviam atualizações de modelo ao controlador central na nuvem. O modelo global é redistribuído para decisões orientadas a impacto, sem compartilhamento de dados brutos. 105
Figura 32 – Comparação entre cenários IID e Non-IID. Linhas tracejadas representam as linhas de base centralizadas. A degradação Non-IID é inferior a 1 ponto percentual para todas as arquiteturas. 113
Figura 33 – Curvas de convergência para DNN, GRU e LSTM ao longo de 30 rodadas de comunicação. As quatro configurações (IID/Non-IID × FedAvg/FedProx) convergem de forma similar em cada arquitetura, com todas atingindo o limiar de 85% entre as rodadas 2 e 3, e 90% entre as rodadas 5 e 10. O DNN apresenta a convergência mais rápida e estável. 114
Figura 34 – Compromisso entre eficiência e acurácia. O DNN oferece o melhor equilíbrio entre desempenho e tempo de treinamento no cenário federado. 115
Figura 35 – Compromisso entre *accuracy* e segurança (C2A Rate) na Fase 1b. Cada ponto representa uma configuração modelo/estratégia/escala. As defesas Bizantino-robustas (Krum e Trimmed Mean, em verde e vermelho) mantêm-se no quadrante inferior direito (*accuracy* $> 91\%$, C2A ≈ 0), enquanto FedAvg e FedProx sem defesa (azul e laranja) migram para o quadrante superior esquerdo com o aumento do fator de escala, atingindo C2A próximo de 100% com $\lambda = 10\times$. 131
Figura 36 – Evolução da C2A Rate ao longo das 30 rodadas de comunicação na Fase 1b, por modelo e estratégia de agregação. Com fatores $5\times$ e $10\times$, FedAvg e FedProx (azul e laranja) apresentam oscilação persistente, indicando que o modelo global alterna entre estados envenenados e parcialmente recuperados sem convergir. Krum e Trimmed Mean (verde e vermelho) mantêm C2A ≈ 0 em todas as rodadas e escalas. 132
Figura 37 – Sensibilidade ao número de épocas locais ($e \in \{1, 3, 5, 10\}$) sob *label-flipping* sem escalamento. Painéis superiores: *accuracy* final por modelo e estratégia — valores crescem monotonicamente com o número de épocas. Painéis inferiores: C2A Rate correspondente — permanece essencialmente nula em todas as configurações, confirmando que o *label-flipping* isolado não compromete a classificação da classe *Critical* independentemente da intensidade do treinamento local. 133

LISTA DE TABELAS

Tabela 1 – Comparativo entre aprendizado supervisionado e não supervisionado. . 34
Tabela 2 – Comparativo entre aprendizado centralizado e federado. 37
Tabela 3 – Parâmetros utilizados no cenário de avaliação de desempenho com estatísticas de rede. 57
Tabela 4 – Comparação da proposta com a literatura. 65
Tabela 5 – Resumo dos parâmetros utilizados no cenário de avaliação de desempenho. 76
Tabela 6 – Características das aplicações. 76
Tabela 7 – Visão geral das funções centrais do framework AIMS. 88
Tabela 8 – Limiares de classificação de impacto e pesos das métricas por classe de aplicação. 93
Tabela 9 – Especificações das arquiteturas de redes neurais avaliadas. 108
Tabela 10 – Matriz de alocação condicional por classe no cenário Non-IID. 110
Tabela 11 – Configuração experimental completa. 111
Tabela 12 – Resultados experimentais completos: métricas de desempenho, convergência e eficiência. 116
Tabela 13 – Capacidades assumidas para o adversário. 124
Tabela 14 – Peso efetivo do cliente malicioso na agregação FedAvg em função do fator de escala λ, com $n = 3$ clientes de volumes iguais. 125
Tabela 15 – Ambiente computacional. 127
Tabela 16 – Parâmetros da topologia federada. 127
Tabela 17 – Distribuição Non-IID de classes por cliente (RSU). 127
Tabela 18 – Matriz completa de experimentos (144 configurações). 129
Tabela 19 – Fase 1a — *Label-flipping* puro, escala 1,0× (todos os 18 experimentos com C2A = 0,0000). 130
Tabela 20 – Fase 1b — Fator de escala 5× (*label-flip* 100%). 130
Tabela 21 – Fase 1b — Fator de escala 10× (*label-flip* 100%). 131
Tabela 22 – Sensibilidade ao número de épocas locais — DNN FedAvg (*label-flip*, $\lambda = 1{,}0$). 132
Tabela 23 – Síntese das contribuições, questões de pesquisa e resultados quantitativos. 141

LISTA DE ABREVIATURAS E SIGLAS

3GPP	3rd Generation Partnership Project
5G	5G Networks
6G	6G Networks
AI	Artificial Intelligence
B5G	Beyond-5G Networks
BS	Base Station
CN	Core Network
DT	Decision Tree
eMBB	Enhanced Mobile Broadband
FL	Federated Learning
H2M	Human to Machine Communications
H2H	Human to Human Communications
IMT	International Mobile Telecommunications
IOT	Internet of Things
ITS	Intelligent Transportation Systems
ITU	International Telecommunication Union
LEO	Low Earth Orbit Satellite
LR	Logistic Regression
M2M	Machine to Machine Communications
ML	Machine Learning
mMTCs	Massive Machine Type Communication
NGMN	Next Generation Mobile Networks Alliance
RAN	Radio Access Network

RF	Random Forest
RTT	Round-trip time
SDN	Software Defined Networks
SVM	Support Vector Machine
TN	Transport Network
UAV	Unmanned Aerial Vehicle
URLLCs	Ultra-Reliable Low Latency Communication
VIoTs	Vehicular IoTs
VNFs	Virtual Network Functions

LISTA DE SÍMBOLOS

Γ	Letra grega Gama
Λ	Lambda
ζ	Letra grega minúscula zeta
$\in$	Pertence

SUMÁRIO

1 INTRODUÇÃO 20
1.1 Motivação 20
1.2 Problema 22
1.3 Objetivos 22
1.3.1 Objetivo principal 22
1.3.2 Objetivos parciais 22
1.3.3 Hipótese de pesquisa 23
1.4 Questões de pesquisa 23
1.5 Contribuições 24
1.6 Métodos de pesquisa 25
1.7 Produção científica associada ao doutorado 26
1.8 Organização da tese 27

2 REFERENCIAL TEÓRICO 28
2.1 Redes 5G, B5G e fatiamento de rede 28
2.2 Gerenciamento do ciclo de vida das fatias 30
2.3 Sistemas Inteligentes de Transporte e redes veiculares 31
2.4 Aprendizado de máquina 32
2.4.1 Aprendizado supervisionado 32
2.4.2 Aprendizado não supervisionado 33
2.5 Redes neurais artificiais 33
2.5.1 Redes neurais profundas (DNN) 34
2.5.2 Redes LSTM 35
2.5.3 Redes GRU 35
2.6 Aprendizado centralizado versus federado 36
2.6.1 FedAvg 36
2.6.2 FedProx 37
2.7 Considerações sobre o referencial teórico 38

3 DETECÇÃO DE CONTEXTO VEICULAR COM DADOS PÚBLICOS DE TRANSPORTE 39
3.1 Contextualização e motivação 39
3.2 Trabalhos relacionados à detecção de congestionamentos urbanos . 40
3.3 Abordagem proposta para identificação de contexto veicular utilizando dados públicos 42

3.4 Experimentos de validação . . . 44
3.4.1 Pré-processamento e configuração dos modelos . . . 47
3.5 Resultados . . . 48
3.6 Considerações finais . . . 50

4 PREDIÇÃO DE CONTEXTO VEICULAR POR ESTATÍSTICAS DE REDE DE COMUNICAÇÃO . . . 52
4.1 Trabalhos relacionados . . . 53
4.2 Abordagem proposta para identificação de contexto veicular utilizando estatísticas de rede . . . 54
4.3 Avaliação de desempenho . . . 55
4.3.1 Implementação do cenário de congestionamento de tráfego . . . 56
4.3.2 Definição das características e classes de ML . . . 57
4.3.3 Treinamento e teste dos modelos de ML . . . 58
4.4 Resultados . . . 58
4.5 Considerações finais . . . 60

5 FRAMEWORK DE FATIAMENTO DE REDE ORIENTADO A APLICAÇÕES ITS . . . 61
5.1 Trabalhos relacionados . . . 62
5.2 Framework proposto para redes veiculares orientadas a aplicações . . . 64
5.2.1 Plano de aplicação . . . 65
5.2.2 Plano de dados . . . 67
5.2.3 Plano de controle . . . 67
5.2.3.1 MANO (*Management and Orchestration*) . . . 68
5.2.3.2 Controlador de rede central . . . 70
5.2.3.3 Controlador de rede local . . . 71
5.3 Avaliação experimental . . . 73
5.3.1 Cenário de avaliação . . . 75
5.3.2 Abordagens comparadas . . . 77
5.3.3 Métricas de avaliação . . . 78
5.4 Resultados . . . 79
5.4.1 Vazão e RTT ao longo do tempo . . . 79
5.4.2 Taxa de entrega de pacotes . . . 81
5.4.3 Tempo de ida e volta (RTT) . . . 83
5.4.4 Discussão . . . 84
5.5 Considerações finais do capítulo . . . 84

6 AIMS: GERENCIAMENTO ADAPTATIVO E INTELIGENTE DE FATIAS PARA REDES ITS DE PRÓXIMA GERAÇÃO . . . 86

6.1 Posicionamento em relação aos trabalhos correlatos . . . 87
6.2 Arquitetura do framework AIMS . . . 88
6.2.1 Fase 1: Preparação das fatias . . . 89
6.2.2 Fase 2: Comissionamento das fatias . . . 90
6.2.3 Fase 3: Operação das fatias . . . 90
6.2.4 Fase 4: Descomissionamento das fatias . . . 91
6.3 Taxonomia de classificação de impacto e formulação do problema . 91
6.3.1 Engenharia de atributos . . . 92
6.3.2 Esquema de rotulagem de impacto . . . 92
6.3.3 Justificativa para o uso de aprendizado de máquina . . . 92
6.4 Configuração experimental . . . 93
6.4.1 Dataset e cenário . . . 93
6.4.2 Protocolo de avaliação *leakage-free* . . . 94
6.4.3 Modelos avaliados . . . 94
6.4.4 Reprodutibilidade . . . 95
6.5 Resultados e discussão . . . 95
6.5.1 Desempenho por nível de impacto . . . 95
6.5.2 *Trade-off* entre tempo de treinamento e acurácia . . . 96
6.5.3 Gap de generalização . . . 98
6.5.4 Estabilidade e eficiência de otimização . . . 98
6.6 Considerações finais do capítulo . . . 99

7 CLASSIFICAÇÃO FEDERADA DE IMPACTO DE FATIAS EM REDES ITS . . . 102
7.1 Motivação e contexto . . . 103
7.2 Arquitetura FL para gerenciamento de fatias . . . 104
7.2.1 FL centralizado para inteligência de borda . . . 104
7.2.2 Gerenciamento do ciclo de vida orientado a impacto . . . 105
7.2.3 Papel do classificador de impacto . . . 106
7.3 Estratégias de treinamento e arquiteturas neurais . . . 106
7.3.1 Treinamento centralizado (linha de base) . . . 107
7.3.2 Média federada (FedAvg) . . . 107
7.3.3 Média federada com regularização proximal (FedProx) . . . 107
7.3.4 Arquiteturas de redes neurais . . . 108
7.4 Configuração experimental . . . 109
7.4.1 Dataset e cenário . . . 109
7.4.2 Particionamento de dados: cenários IID e Non-IID . . . 110
7.4.3 Métricas e protocolo de avaliação . . . 110
7.4.4 Configuração completa dos experimentos . . . 111
7.5 Resultados e discussão . . . 112

7.5.1 Desempenho em cenários IID . . . 112
7.5.2 Efeito da heterogeneidade estatística (Non-IID) . . . 112
7.5.3 FedAvg versus FedProx em cenários Non-IID . . . 113
7.5.4 Convergência e seleção de modelo . . . 114
7.5.5 Resultados completos . . . 115
7.6 Considerações finais . . . 116

8 SEGURANÇA DA INFORMAÇÃO EM REDES ITS COM FATIAMENTO INTELIGENTE . . . 118
8.1 Estado da Arte em Segurança para Fatiamento Inteligente em ITS 118
8.1.1 Segurança de FL em redes veiculares . . . 119
8.1.2 Ataques de envenenamento em aprendizado federado . . . 120
8.1.3 Agregação Bizantino-robusta . . . 120
8.1.4 Lacuna: segurança do FL no contexto de fatiamento de rede para ITS . . . 121
8.1.5 Pergunta de pesquisa e contribuições . . . 122
8.2 Modelo de Ameaça e Formulação do Problema . . . 122
8.2.1 Cenário adversarial . . . 123
8.2.2 Capacidades do atacante . . . 123
8.2.3 Ataque de *label-flipping* (envenenamento de dados) . . . 123
8.2.4 Ataque de *gradient scaling* (envenenamento de modelo) . . . 124
8.2.5 Classificação na taxonomia de ataques . . . 125
8.2.6 Métrica operacional de segurança: C2A Rate . . . 125
8.2.7 Formulação do problema . . . 126
8.3 Prova de Conceito: Metodologia Experimental . . . 126
8.3.1 Dataset e pré-processamento . . . 126
8.3.2 Arquitetura de aprendizado federado . . . 126
8.3.3 Distribuição de dados . . . 126
8.3.4 Modelos de classificação . . . 128
8.3.5 Estratégias de agregação . . . 128
8.3.6 Desenho experimental . . . 128
8.4 Resultados e Discussão . . . 129
8.4.1 Resultados experimentais . . . 129
8.4.1.1 Fase 1a — *Label-flipping* puro . . . 129
8.4.1.2 Fase 1b — *Gradient scaling* . . . 130
8.4.1.3 Análise de sensibilidade — épocas locais . . . 132
8.4.2 Discussão . . . 132
8.4.2.1 Vulnerabilidade estrutural ao *gradient scaling* . . . 132
8.4.2.2 FedProx não constitui defesa . . . 133
8.4.2.3 Eficácia das defesas Bizantino-robustas . . . 134
8.5 Limitações e Direções Futuras . . . 134

8.5.1 Limitações experimentais 134
8.5.2 Direções futuras 136
8.6 Considerações Finais do Capítulo 136

9 CONCLUSÕES E TRABALHOS FUTUROS 139
9.1 Síntese das contribuições 139
9.2 Limitações 142
9.3 Trabalhos futuros 142

REFERÊNCIAS 144

1 INTRODUÇÃO

À medida que a sociedade avança em direção à digitalização completa e a Internet das Coisas (IoT) se expande, os Sistemas Inteligentes de Transporte (*Intelligent Transportation Systems* — ITS) desempenham um papel central na transformação da mobilidade urbana. As aplicações ITS dependem cada vez mais de redes de comunicação capazes de atender simultaneamente a requisitos heterogêneos, que vão de mensagens de segurança com latência crítica a fluxos de vídeo de alta demanda, sobre uma infraestrutura compartilhada.

As redes de quinta geração (5G) endereçaram parte dos desafios com o fatiamento de rede (*network slicing*), que particiona logicamente a rede do operador em fatias isoladas e personalizadas para diferentes serviços fim a fim [1]. Apesar disso, a alta variabilidade de carga, topologia e condições de canal impõe um desafio que o fatiamento estático não resolve: manter o atendimento aos requisitos das aplicações exige gerenciar o ciclo de vida das fatias de forma adaptativa e orientada ao contexto [2].

A evolução para as redes de sexta geração (6G) aponta a inteligência de borda como elemento central para lidar com essa complexidade, ao aproximar o processamento e a tomada de decisão dos dados gerados na rede [3, 4]. Esse cenário motiva a adoção de técnicas de aprendizado de máquina para tornar o gerenciamento das fatias mais autônomo e orientado a dados, abordagem que esta tese investiga no contexto de aplicações ITS.

1.1 Motivação

A identificação de contexto constitui requisito para soluções em cenários ITS, para que na fase de operação do ciclo de vida das fatias a infraestrutura possa se readequar em função de mudanças no contexto dos veículos. A partir do contexto identificado, técnicas de aprendizado de máquina permitem classificar o impacto dessas mudanças sobre as políticas de fatiamento vigentes, gerando sinais acionáveis para adaptação dos recursos. Quando o treinamento desses classificadores precisa ocorrer de forma distribuída entre múltiplos pontos da infraestrutura — como unidades de borda de estrada (*Road-Side Units* — RSUs) — o aprendizado federado (*Federated Learning* — FL) se apresenta como abordagem adequada, pois permite o treinamento colaborativo sem centralização dos dados brutos [3, 5]. Contudo, a distribuição do treinamento introduz superfícies de ataque que demandam investigação sobre a segurança do sistema.

Embora o fatiamento de rede tenha surgido para uso em escala a partir das redes 5G, esta abordagem permanece fundamental para as redes B5G. O ciclo de vida das

fatias compreende desde a preparação, com a identificação correta dos requisitos, passando pela fase de definição e planejamento de alocação dos recursos, até a fase de operação, onde ocorre a efetiva gestão dos recursos em uso na infraestrutura [6]. São muitos os desafios associados ao gerenciamento do ciclo de vida, envolvendo desde a tradução de requisitos dos serviços em configurações e políticas para alocação de recursos, passando pela orquestração adequada desses recursos, além de desafios para manutenção de fatias adaptáveis ao contexto veicular [2], que deve ser identificado de maneira eficiente para uma orquestração orientada a dados [7].

Com base em levantamento da literatura, considerando principalmente referências como [2], [1] e [5], o problema do gerenciamento inteligente do ciclo de vida das fatias em cenários ITS pode ser decomposto nos seguintes desafios técnicos, que estruturam as questões de pesquisa desta tese:

D1. Ausência de abordagens que extraiam o contexto veicular a partir de fontes acessíveis — como dados públicos de transporte ou estatísticas da própria rede de comunicação — para alimentar decisões de gerenciamento de fatias na fase de operação;

D2. Soluções de fatiamento existentes não especificam algoritmos de controle reprodutíveis e não consideram os requisitos heterogêneos de diferentes aplicações ITS na orquestração dinâmica de recursos;

D3. Ausência de mecanismos leves e interpretáveis para classificação de impacto de políticas de fatiamento a partir de telemetria veicular ruidosa, produzindo sinais acionáveis para adaptação proativa das fatias ao longo do ciclo de vida;

D4. A heterogeneidade estatística (Non-IID) entre RSUs em corredores veiculares distribuídos desafia o treinamento federado de modelos de classificação, comprometendo a escalabilidade de soluções baseadas em aprendizado de máquina para o gerenciamento de fatias;

D5. A introdução de aprendizado distribuído no gerenciamento de fatias cria superfícies de ataque — como envenenamento de modelo e manipulação de classificações de impacto — não endereçadas na literatura de fatiamento inteligente para ITS.

Esses cinco desafios, embora frequentemente tratados de forma isolada na literatura, manifestam-se de maneira interdependente em cenários ITS reais. A ausência de uma abordagem que os enderece de forma integrada delimita o problema central desta tese, formulado a seguir.

1.2 Problema

O problema central endereçado nesta tese é o gerenciamento inteligente do ciclo de vida de fatias de rede em cenários B5G que suportam aplicações ITS. Em tais cenários, a dinâmica veicular, marcada por variações de carga, mudanças de topologia e condições de canal heterogêneas, exige mecanismos capazes de: (i) identificar o contexto veicular a partir de fontes acessíveis na infraestrutura; (ii) orquestrar recursos de forma dinâmica com algoritmos de controle orientados aos requisitos heterogêneos das aplicações ITS; (iii) classificar o impacto das condições observadas sobre as políticas de fatiamento vigentes, produzindo sinais acionáveis para adaptação; (iv) escalar o treinamento dos modelos de classificação de forma distribuída sob heterogeneidade estatística; e (v) garantir a integridade dessas classificações diante de participantes potencialmente comprometidos. A ausência de soluções que contemplem essas dimensões de forma integrada constitui a lacuna central que esta pesquisa busca endereçar.

1.3 Objetivos

1.3.1 Objetivo principal

O objetivo principal desta tese é propor e avaliar mecanismos baseados em aprendizado de máquina para apoiar o gerenciamento do ciclo de vida de fatias de rede em ambientes B5G voltados a aplicações ITS. A tese parte da premissa de que dados de contexto veicular e de telemetria de comunicação, obtidos a partir de fontes acessíveis, podem ser transformados em informação acionável para apoiar decisões ao longo do ciclo de vida das fatias.

Para alcançar esse objetivo, a tese desenvolve uma sequência progressiva de abordagens, na qual cada contribuição endereça uma limitação observada na etapa anterior: parte da identificação do contexto veicular, evolui para a orquestração de recursos orientada aos requisitos das aplicações, avança para o gerenciamento adaptativo baseado na classificação do impacto de políticas de fatiamento, estende esse mecanismo para um cenário distribuído e escalável, e conclui com a análise de sua robustez diante de ameaças à integridade da inteligência distribuída.

1.3.2 Objetivos parciais

Para atingir o objetivo principal, foram definidos os seguintes objetivos parciais:

1. Investigar a identificação do contexto veicular por meio de aprendizado de máquina supervisionado, a partir de duas fontes de dados acessíveis e complementares: dados públicos de sistemas de transporte, para a detecção de congestionamentos sem

infraestrutura dedicada de sensoriamento; e estatísticas de tráfego da própria rede de comunicação, obtidas no núcleo via SDN, tratando a rede como sensor implícito do contexto;

2. Propor e avaliar um framework de fatiamento de rede orientado aos requisitos de aplicações ITS, baseado em redes definidas por software, capaz de priorizar e adaptar recursos dinamicamente considerando a mobilidade dos veículos e a heterogeneidade das aplicações;

3. Propor o AIMS, uma abordagem de gerenciamento adaptativo que utiliza a classificação do impacto de políticas de fatiamento, a partir de telemetria de monitoramento, como sinal interpretável para orientar decisões ao longo do ciclo de vida das fatias;

4. Estender a classificação de impacto para um cenário distribuído e escalável por meio de aprendizado federado entre controladores locais, avaliando sua viabilidade sob heterogeneidade estatística dos dados;

5. Investigar a robustez da classificação federada de impacto diante de ataques de envenenamento, quantificando o risco operacional e avaliando a eficácia de estratégias de agregação robusta como mecanismo de mitigação.

1.3.3 Hipótese de pesquisa

A hipótese central desta tese é que o gerenciamento do ciclo de vida de fatias em redes B5G para cenários ITS pode ser apoiado por mecanismos baseados em aprendizado de máquina que convertem o contexto veicular e a telemetria de comunicação, obtidos de fontes acessíveis, em sinais acionáveis para decisões de fatiamento. Especificamente, postula-se que a classificação do impacto de políticas de fatiamento pode fornecer um sinal interpretável para orientar essas decisões; que esse mecanismo pode ser estendido para treinamento distribuído sob heterogeneidade estatística, mantendo desempenho competitivo em relação ao treinamento centralizado; e que os riscos à integridade da classificação, diante de ataques de envenenamento, podem ser mitigados por estratégias de agregação robusta. As questões de pesquisa formuladas na Seção 1.4 operacionalizam essa hipótese, decompondo-a em proposições verificáveis avaliadas experimentalmente ao longo dos capítulos de contribuição.

1.4 Questões de pesquisa

A partir dos desafios identificados, formulam-se as seguintes questões de pesquisa (QPs):

QP1. É possível identificar o contexto veicular (por exemplo, congestionamentos urbanos) a partir de dados públicos de transporte ou de estatísticas de rede de comunicação, utilizando classificadores de aprendizado supervisionado, com acurácia suficiente para uso em decisões de gerenciamento de fatias?

QP2. Um framework de fatiamento de rede 5G orientado a aplicações ITS, com algoritmos de controle explícitos, apresenta ganhos mensuráveis no atendimento aos requisitos das aplicações em comparação com abordagens do estado da arte?

QP3. Modelos de aprendizado de máquina leves são capazes de classificar o impacto de políticas de fatiamento a partir de telemetria veicular ruidosa, produzindo sinais acionáveis para o ciclo de vida das fatias?

QP4. O treinamento federado de classificadores de impacto é viável em cenários com a heterogeneidade estatística (Non-IID) inerente a corredores veiculares distribuídos, preservando desempenho próximo ao do treinamento centralizado? Nesse contexto, a regularização proximal (FedProx) oferece vantagem em relação à agregação por média padrão (FedAvg)?

QP5. Quais são os principais desafios de segurança da informação que emergem quando técnicas de aprendizado distribuído são empregadas no contexto de fatiamento inteligente para ITS, e quais estratégias de agregação robusta preservam a integridade da classificação de impacto?

1.5 Contribuições

As contribuições desta tese se organizam em uma narrativa progressiva que parte da identificação de contexto veicular, avança para o fatiamento orientado a aplicações, evolui para redes B5G com classificação de impacto de políticas, incorpora escalabilidade via aprendizado federado, e trata de aspectos de segurança da informação. As contribuições são sumarizadas a seguir, com indicação da questão de pesquisa endereçada e do capítulo correspondente:

C1. Uma análise de simulações realísticas de redes veiculares ad hoc (VANETs), investigando o impacto de parâmetros de simulação sobre os resultados em cenários com aplicações de disseminação de mensagens [8]. Esta contribuição não possui capítulo dedicado, mas fornece a base metodológica de simulação que orienta os experimentos dos capítulos subsequentes.

C2. Uma abordagem para detecção de congestionamentos urbanos com classificadores de aprendizado supervisionado treinados sobre dados históricos e públicos de mobilidade urbana de ônibus [9]. Endereça QP1 e é apresentada no Capítulo 3.

C3. Uma abordagem de predição de contexto veicular a partir de estatísticas de rede de comunicação, utilizando métricas como RTT e throughput como atributos (*features*) para classificadores [10]. Endereça QP1 e é apresentada no Capítulo 4.

C4. Um framework de fatiamento de rede 5G orientado a aplicações ITS, com arquitetura baseada em SDN, algoritmos de controle e avaliação em emulação realística [11]. Endereça QP2 e é apresentado no Capítulo 5.

C5. O framework AIMS (*Adaptive and Intelligent Management of Slicing*), uma arquitetura híbrida *cloud-edge* para gerenciamento do ciclo de vida de fatias em redes B5G, com classificação de impacto de políticas em taxonomia de quatro níveis [12]. Endereça QP3 e é apresentado no Capítulo 6.

C6. Uma extensão do AIMS com aprendizado federado para classificação distribuída e escalável de impacto de fatias, comparando treinamento centralizado com FedAvg e FedProx em cenários IID e Non-IID [13]. Endereça QP4 e é apresentada no Capítulo 7.

C7. Uma investigação da vulnerabilidade do classificador federado de impacto frente a ataques de envenenamento direcionado, com avaliação de 144 configurações experimentais combinando três arquiteturas neurais, seis estratégias de agregação e múltiplos fatores de amplificação adversarial, indicando que defesas Bizantino-robustas (Krum e Trimmed Mean) mitigam ataques não adaptativos, com limitações documentadas no regime de poucos participantes. Endereça QP5 e é apresentada no Capítulo 8.

1.6 Métodos de pesquisa

Os métodos de pesquisa empregados nesta tese são de natureza quantitativa e experimental. Cada contribuição segue um ciclo que compreende: (i) revisão da literatura para identificação de lacunas; (ii) formulação da proposta (arquitetura, algoritmo ou framework); (iii) implementação de protótipo ou prova de conceito; (iv) avaliação experimental com conjuntos de dados realísticos ou em ambiente de emulação; e (v) análise de resultados com métricas quantitativas (acurácia, F1-score, taxa de entrega de pacotes, latência, entre outras).

Os experimentos dos Capítulos 3 e 4 utilizam dados reais (dados públicos de transporte do Rio de Janeiro e estatísticas de rede em cenário emulado, respectivamente). O Capítulo 5 emprega emulação realística com o Mininet-WiFi. Os Capítulos 6 e 7 utilizam um dataset derivado de cenário ITS com 5.400 amostras e 25 features, seguindo protocolo de avaliação *leakage-free*. O Capítulo 8 reutiliza o mesmo dataset e a infraestrutura federada do Capítulo 7, introduzindo ataques de envenenamento direcionado (*label-flipping* com *gradient scaling*) e avaliando estratégias de agregação robusta em 144 configurações

experimentais que combinam arquiteturas neurais, estratégias de agregação e fatores de amplificação adversarial.

1.7 Produção científica associada ao doutorado

A produção científica associada a este doutorado pode ser organizada em três grupos: trabalhos publicados, trabalho aceito e manuscritos em preparação.

Trabalhos publicados

- [8]: Saraiva, T.V.; Campos, C.A.V. "Realistic Vehicular Networks Simulations". Springer, 2019. (Capítulo de livro)
- [9]: Saraiva, T.V.; Campos, C.A.V. "ML applied to public transport data for identification of vehicular context". IEEE IWCMC 2021. (Conferência)
- [10]: Saraiva, T.V.; Girardi, R.; Campos, C.A.V. "Machine learning applied to network traffic for vehicular applications to predict the vehicle context". IEEE VTC2021-Fall. (Conferência)
- [11]: Saraiva, T.V.; Campos, C.A.V.; Fontes, R.R.; Rothenberg, C.E.; Sorour, S.; Valaee, S. "An Application-Driven Framework for Intelligent Transportation Systems Using 5G Network Slicing". IEEE Transactions on Intelligent Transportation Systems, 2021. (Periódico)

Trabalho aceito

- [12]: Saraiva, T.V.; Campos, C.A.V. "AIMS: Adaptive and Intelligent Management of Slicing for Next-Generation ITS Networks". Aceito para apresentação no IEEE ISCC 2026. (Conferência)

Manuscritos em preparação

- Saraiva, T.V.; Campos, C.A.V. "Poisoning Attacks and Robust Defenses for Federated Slice Impact Classification in Intelligent Transportation Systems". Manuscrito em preparação, 2026.
- [13]: Saraiva, T.V.; Campos, C.A.V. "Federated Learning-Based Scalable Network Slice Impact Classification for Intelligent Transportation Systems". Manuscrito em preparação, 2026.

1.8 Organização da tese

O restante desta tese está organizado da seguinte forma:

- **Capítulo 2 — Referencial Teórico:** apresenta os fundamentos de redes 5G e B5G, fatiamento de rede, gerenciamento de ciclo de vida das fatias, aprendizado de máquina supervisionado e não supervisionado, redes neurais artificiais (DNN, LSTM, GRU) e aprendizado centralizado versus federado (FedAvg, FedProx).
- **Capítulo 3 — Detecção de Contexto Veicular com Dados Públicos:** apresenta uma abordagem de detecção de congestionamentos urbanos baseada em classificadores treinados com dados públicos de mobilidade, com avaliação em cenário real do Rio de Janeiro. Endereça QP1 e apresenta a contribuição C2.
- **Capítulo 4 — Predição de Contexto por Estatísticas de Rede:** descreve uma abordagem de predição de contexto veicular a partir de métricas de rede como atributos para classificação de cenários de congestionamento. Endereça QP1 e apresenta C3.
- **Capítulo 5 — Framework de Fatiamento de Rede 5G:** apresenta o framework orientado a aplicações ITS, com arquitetura SDN, algoritmos de controle e avaliação em emulação realística. Endereça QP2 e apresenta C4.
- **Capítulo 6 — AIMS:** introduz o framework para gerenciamento adaptativo e inteligente de fatias em redes B5G, com taxonomia de classificação de impacto e resultados experimentais. Endereça QP3 e apresenta C5.
- **Capítulo 7 — Classificação Federada de Impacto:** estende o AIMS com aprendizado federado, avaliando a viabilidade da classificação distribuída sob heterogeneidade estatística em corredores veiculares. Endereça QP4 e apresenta C6.
- **Capítulo 8 — Segurança da Informação:** investiga a vulnerabilidade do classificador federado de impacto frente a ataques de envenenamento direcionado (*label-flipping* com *gradient scaling*) e avalia estratégias de agregação Bizantino-robusta (Krum e Trimmed Mean) em 144 configurações experimentais, indicando que tais defesas mitigam ataques não adaptativos, com limitações documentadas no regime de poucos participantes. Endereça QP5 e apresenta C7.
- **Capítulo 9 — Conclusões e Trabalhos Futuros:** sintetiza as contribuições, discute limitações e apresenta direções para trabalhos futuros.

2 REFERENCIAL TEÓRICO

Este capítulo apresenta os fundamentos teóricos necessários para a compreensão das contribuições desta tese. A Seção 2.1 discute as redes 5G, B5G e o conceito de fatiamento de rede. A Seção 2.2 detalha o gerenciamento do ciclo de vida das fatias. A Seção 2.3 contextualiza os Sistemas Inteligentes de Transporte e os desafios das redes veiculares. A Seção 2.4 introduz os conceitos de aprendizado de máquina relevantes para o trabalho. A Seção 2.5 apresenta as arquiteturas de redes neurais empregadas nos experimentos. A Seção 2.6 discute o aprendizado federado e suas motivações no contexto de redes móveis.

2.1 Redes 5G, B5G e fatiamento de rede

As tecnologias de comunicação previstas no 5G foram propostas para suportar diferentes requisitos de diferentes serviços em redes com um grande número de usuários, taxas de dados significativas, baixa latência fim a fim, além de integração com diversas tecnologias de rede sem fio [14]. Os cenários em 5G podem ser classificados de acordo com o padrão envolvido (NGMN, 3GPP, IMT, ITU). O ITU-R, por exemplo, propôs banda larga móvel aprimorada com baixa latência e comunicação massiva para suportar três classes de tráfego com requisitos e características heterogêneos sobre uma infraestrutura física comum: banda larga móvel aprimorada (*enhanced Mobile Broadband* — eMBB), comunicações massivas de tipo de máquina (*massive Machine-Type Communications* — mMTCs) e comunicações ultra-confiáveis de baixa latência (*Ultra-Reliable Low-Latency Communications* — URLLCs) [14].

Com o conceito de fatiamento de rede (*network slicing*), o 5G visa atender aos requisitos de serviços diversificados utilizando as tecnologias disponíveis [15]. De forma geral, uma fatia de rede pode ser entendida como uma rede lógica fim a fim, logicamente isolada e dedicada a um conjunto de serviços com requisitos comuns, provisionada sobre uma infraestrutura física compartilhada. Essa abstração é independente de uma tecnologia de implementação específica, podendo ser realizada por diferentes mecanismos de virtualização e orquestração. Em sua formalização pelo 3GPP, uma fatia de rede é definida como uma rede lógica que compreende uma ou mais cadeias de serviços formadas por funções de rede virtualizadas ou físicas e os enlaces que as conectam, sendo criada com isolamento apropriado, recursos e topologia otimizada para atender a um ou mais serviços de comunicação [2].

Devido à alta dinamicidade de serviços em redes B5G, as infraestruturas modernas precisam de fatiamento fim a fim para lidar com esses diferentes tipos de serviços [16].

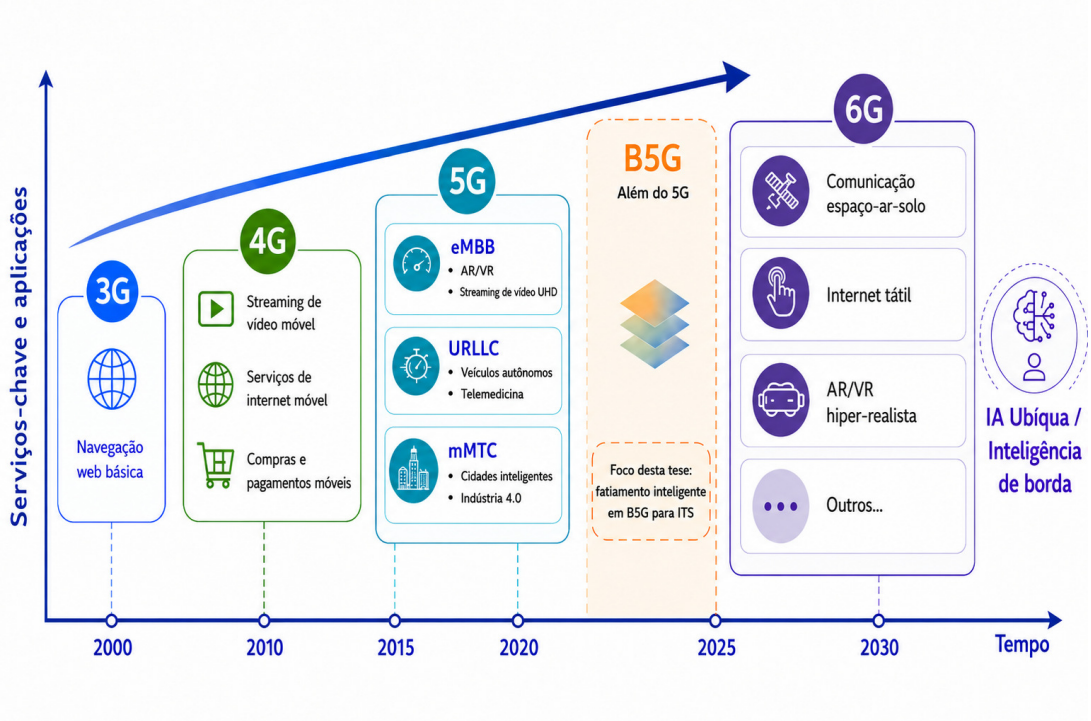


Figura 1 – Serviços-chave, aplicações e roteiro de evolução das redes móveis até o 6G, com destaque para o posicionamento desta tese na transição B5G. Fonte: adaptado de [5].

As atuais redes definidas por software em 5G precisarão adaptar seu fatiamento de rede para atender às rápidas transições de requisitos de serviços, uma vez que a autonomia em massa no 6G exigirá o fatiamento para demandas de serviços diversas e dinâmicas, incorporando algoritmos de IA robustos, precisos e de alta velocidade [17]. O fatiamento inteligente fim a fim traz um nível adicional de complexidade, uma vez que envolve a necessidade de reaprendizagem altamente eficiente e coordenação entre várias entidades [18].

A Figura 1 apresenta o roteiro de evolução das redes móveis até o 6G, adaptado de [5], situando o escopo desta tese na transição para as redes B5G, com ênfase no fatiamento inteligente.

Comparado às redes sem fio existentes, o 6G é mais do que uma melhoria dos requisitos com maiores taxas e capacidades. Espera-se que as redes 6G criem um ecossistema que possibilite redes espaciais (satélites LEO), redes aéreas (UAVs) e redes terrestres (estações base), integradas em uma arquitetura espaço-ar-solo com cobertura global, utilizando técnicas de virtualização e fatiamento de rede para gerenciamento flexível, com inteligência onipresente desde os dispositivos finais, passando pela borda até nuvens remotas [1]. Na Figura 2 consta uma visão de futuro das redes 6G com alguns dos principais casos de uso previstos.

Os objetivos do 6G incluem não apenas uma rede de transporte de dados aper-

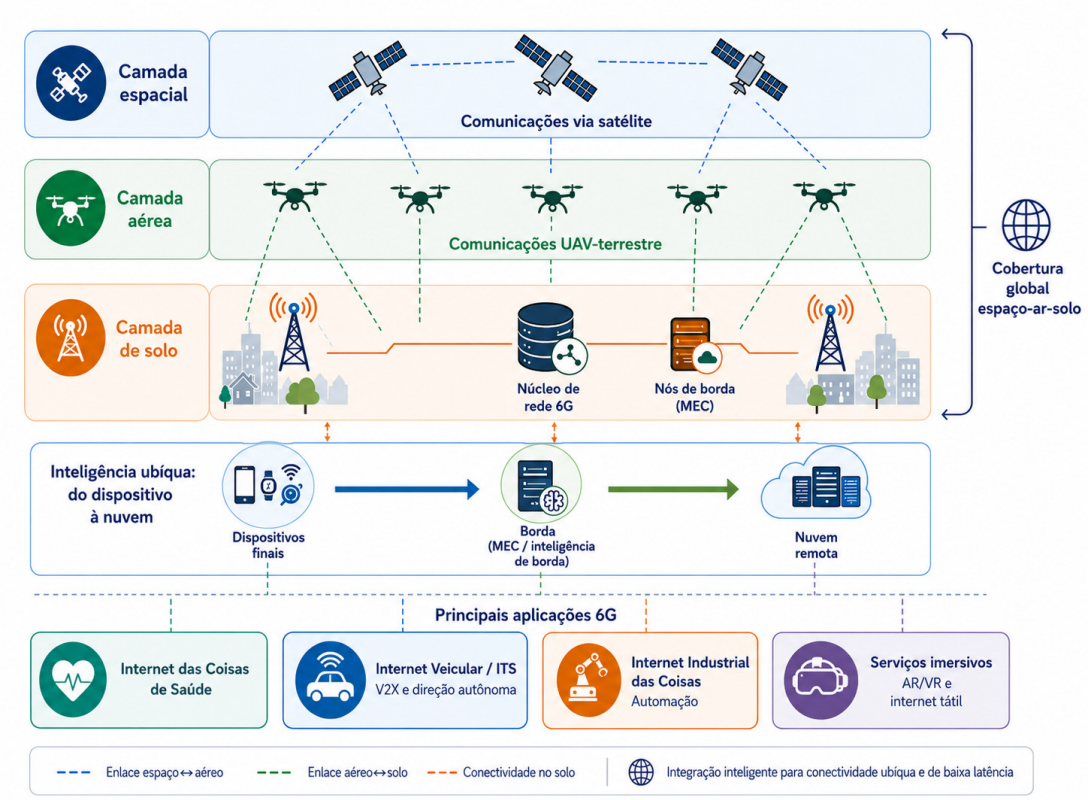


Figura 2 – Visão de ecossistema das redes 6G, com arquitetura integrada espaço-ar-solo e principais casos de uso previstos. Fonte: adaptado de [4].

feiçoada, mas também um sistema altamente inteligente e totalmente autônomo, com uma arquitetura hiper-flexível que traga inteligência em todos os aspectos dos sistemas de rede [19]. Da mesma forma que o aspecto "não rádio" da comunicação 5G (*core* de comunicações de dados) se tornou crítico para o desempenho das redes, em 6G os aspectos de aprendizagem e inteligência se tornam cruciais e estão integrados a diferentes redes de comunicação [20]. Nos sistemas 6G inteligentes, as funções de IA serão estendidas até a borda da rede, graças às capacidades computacionais dos nós de borda, levando a um paradigma chamado inteligência de borda [4]. A inteligência de borda é uma estrutura tecnológica focada na integração de IA, redes de comunicação e computação de borda móvel (*Multiple-access Edge Computing* — MEC) [19].

2.2 Gerenciamento do ciclo de vida das fatias

Para se explorar plenamente as vantagens do fatiamento de rede, os operadores precisam trabalhar com alocação dinâmica de recursos, garantia de serviço, isolamento, proteção e particionamento otimizados em todos os domínios envolvidos no gerenciamento das fatias: Rede de Acesso Rádio (RAN), Rede de Transporte (TN) e Rede Principal (CN), considerando desde a preparação até a desativação das fatias [2].

O ciclo de vida das fatias de rede, conforme definido pelo 3GPP, compreende

quatro fases [6]: (i) **Preparação**, que envolve o projeto da fatia, avaliação de viabilidade e capacidade, bem como a definição de templates; (ii) **Comissionamento/Planejamento**, que inclui a criação da instância da fatia, a alocação e configuração dos recursos necessários; (iii) **Operação**, que cobre a ativação, supervisão, monitoramento de desempenho e modificação da instância da fatia em tempo de execução; e (iv) **Descomissionamento**, que corresponde à desativação e liberação dos recursos alocados.

A integração entre essas fases é um desafio em aberto. A fase de operação frequentemente demanda ajustes que dependem de informações de monitoramento e predição, as quais devem realimentar a fase de planejamento para reconfiguração dos recursos. Essa necessidade de integração entre fases que operam em escalas de tempo diferentes é um dos problemas centrais endereçados nesta tese, especificamente nos Capítulos 6 e 7, que propõem mecanismos de classificação de impacto como gatilhos para ações no ciclo de vida.

O MEC oferece vantagens para redes B5G e 6G, reduzindo a latência e o uso de largura de banda ao aumentar a capacidade da borda para oferecer serviços aos usuários finais [21]. Nesse contexto, abordagens de aprendizado distribuído como o aprendizado federado podem explorar a capacidade oferecida pela computação de borda, reduzindo a quantidade de troca de informações [2] e possibilitando o processamento local de dados veiculares para agregação de predições associadas ao contexto veicular.

2.3 Sistemas Inteligentes de Transporte e redes veiculares

Um dos cenários onde se espera ganho significativo com melhor gerenciamento das redes diz respeito às comunicações que suportam aplicações relacionadas a Sistemas Inteligentes de Transporte (ITS). Nesse tipo de cenário, estão presentes os desafios típicos das redes de comunicação sem fio, com elevado impacto devido à mobilidade de veículos, além dos diferentes serviços relacionados a segurança no trânsito, entretenimento, informações e direção autônoma [17]. As tecnologias chave previstas em 6G, como comunicações THz e IA, são previstas para suportar serviços críticos como o de veículos autônomos conectados (CAV) [22].

As soluções de IA para ITS podem ser divididas em duas categorias: “comunicação em ITS” e “rede em ITS” [23], conforme ilustrado na Figura 3. A primeira categoria abrange técnicas para otimização dos protocolos e mecanismos de comunicação veicular, enquanto a segunda foca no gerenciamento inteligente dos recursos e da infraestrutura de rede.

Dado que cada vez mais dispositivos se interconectam, a metodologia atual de configuração e otimização disponível em 5G pode não ser adequada para as redes sem fio em cenários B5G, sendo que se espera que as próximas redes 6G possam fornecer funciona-

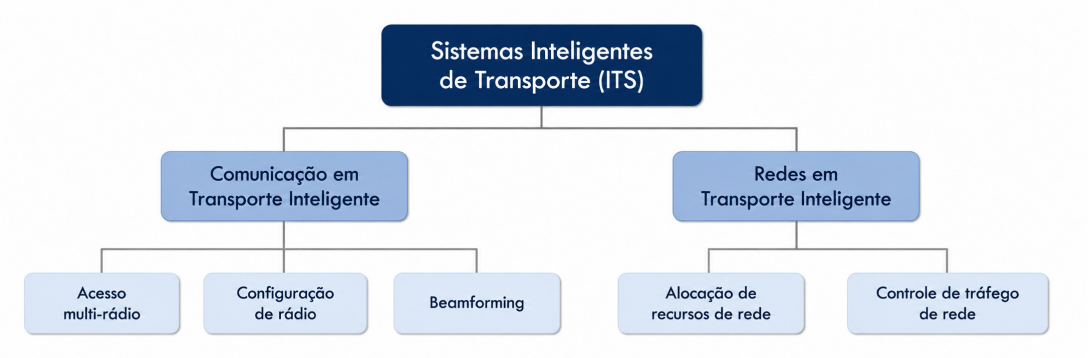


Figura 3 – Soluções de IA em ITS divididas em categorias. Fonte: adaptado de [23].

lidades integradas com técnicas de IA [23]. Os avanços nas tecnologias 6G transformarão de maneira significativa as redes de internet das coisas veiculares (VIoTs) e os sistemas de transporte inteligente [4].

2.4 Aprendizado de máquina

O aprendizado de máquina (*Machine Learning* — ML) é uma subárea da inteligência artificial que se ocupa do desenvolvimento de algoritmos capazes de aprender padrões a partir de dados, sem que sejam explicitamente programados para cada tarefa [24]. No contexto desta tese, técnicas de ML são empregadas em diversas contribuições: nos Capítulos 3 e 4 para classificação de contexto veicular, no Capítulo 6 para classificação de impacto de políticas de fatiamento, e no Capítulo 7 para treinamento distribuído de classificadores. As duas categorias fundamentais de aprendizado — supervisionado e não supervisionado — são descritas a seguir.

2.4.1 Aprendizado supervisionado

No aprendizado supervisionado, o modelo é treinado a partir de um conjunto de dados rotulados, ou seja, pares (x_i, y_i) onde x_i é o vetor de atributos (*features*) de entrada e y_i é o rótulo ou valor alvo correspondente. O objetivo é aprender uma função $f : X \to Y$ que generalize para dados não observados durante o treinamento [24].

As tarefas de aprendizado supervisionado se dividem em dois tipos principais. Na **classificação**, o objetivo é atribuir cada amostra a uma entre um conjunto finito de classes discretas. Na **regressão**, o objetivo é prever um valor contínuo. Nesta tese, todas as tarefas de aprendizado supervisionado são de classificação: classificação binária de congestionamento (Capítulos 3 e 4) e classificação multiclasse de impacto de políticas em quatro níveis (Capítulos 6 e 7).

Os algoritmos de aprendizado supervisionado empregados nesta tese incluem:

- **Random Forest:** um método de *ensemble* que constrói múltiplas árvores de decisão durante o treinamento e combina suas predições por votação majoritária (classificação) ou média (regressão) [25]. É empregado no Capítulo 6 como um dos modelos avaliados para classificação de impacto.
- **CatBoost:** um algoritmo de *gradient boosting* que constrói árvores sequencialmente, onde cada nova árvore corrige os erros residuais das anteriores. É projetado para lidar de forma eficiente com features categóricas [26]. Também é empregado no Capítulo 6.
- **Redes neurais artificiais:** descritas em detalhe na Seção 2.5, são empregadas nos Capítulos 6 (TabNet) e 7 (DNN, LSTM, GRU).

A escolha entre algoritmos depende de fatores como a dimensionalidade e natureza dos dados, requisitos de interpretabilidade, custo computacional de treinamento e inferência, e a quantidade de dados disponíveis. Modelos baseados em árvores, como Random Forest e CatBoost, tendem a apresentar bom desempenho com dados tabulares de média dimensionalidade e oferecem maior interpretabilidade. Redes neurais podem capturar relações não lineares mais complexas, mas geralmente exigem mais dados para treinamento e são menos interpretáveis [2].

2.4.2 Aprendizado não supervisionado

No aprendizado não supervisionado, o modelo é treinado sobre dados sem rótulos, buscando identificar estruturas ou padrões intrínsecos nos dados. As tarefas mais comuns incluem agrupamento (*clustering*), onde o objetivo é particionar os dados em grupos com características similares, e redução de dimensionalidade, que busca representações mais compactas dos dados preservando a informação relevante [24].

Embora os experimentos desta tese utilizem predominantemente aprendizado supervisionado, o aprendizado não supervisionado é relevante no contexto mais amplo de gerenciamento de redes B5G. Técnicas de agrupamento podem ser empregadas para segmentação de tráfego e identificação de padrões de mobilidade sem necessidade de dados rotulados, enquanto técnicas de redução de dimensionalidade podem ser aplicadas à compressão de sinais de monitoramento em cenários com grande volume de features [27]. A distinção entre as abordagens supervisionada e não supervisionada é sintetizada na Tabela 1.

2.5 Redes neurais artificiais

Redes neurais artificiais (*Artificial Neural Networks* — ANNs) são modelos computacionais inspirados no funcionamento do sistema nervoso biológico, compostos por

Tabela 1 – Comparativo entre aprendizado supervisionado e não supervisionado.

Aspecto	**Supervisionado**	**Não supervisionado**
Dados de treinamento	Rotulados (x_i, y_i)	Não rotulados (x_i)
Objetivo	Aprender mapeamento $f : X \to Y$	Identificar estrutura nos dados
Tarefas típicas	Classificação, regressão	Agrupamento, redução de dimensionalidade
Uso nesta tese	Caps. 3, 4, 6, 7	Mencionado como abordagem complementar

unidades de processamento (neurônios) organizadas em camadas e conectadas por pesos ajustáveis [28]. Cada neurônio recebe um conjunto de entradas, aplica uma combinação linear seguida de uma função de ativação não linear, e produz uma saída. O treinamento consiste em ajustar os pesos da rede por meio de um algoritmo de otimização — tipicamente variantes do gradiente descendente — de forma a minimizar uma função de perda que mede a discrepância entre as predições do modelo e os valores alvo.

Nesta tese, três arquiteturas de redes neurais são empregadas nos experimentos dos Capítulos 6 e 7: redes neurais profundas (DNN), redes com memória de longo prazo (LSTM) e unidades recorrentes com porta (GRU). Cada uma é descrita a seguir.

2.5.1 Redes neurais profundas (DNN)

Uma rede neural profunda (*Deep Neural Network* — DNN) é uma rede *feedforward* com múltiplas camadas ocultas entre a camada de entrada e a camada de saída. Cada camada oculta aplica uma transformação não linear nos dados, permitindo que a rede aprenda representações hierárquicas de crescente abstração. Formalmente, para uma rede com L camadas, a saída da camada l é dada por:

$$h^{(l)} = \sigma(W^{(l)} h^{(l-1)} + b^{(l)}) \tag{2.1}$$

onde $W^{(l)}$ é a matriz de pesos, $b^{(l)}$ é o vetor de viés e $\sigma(\cdot)$ é a função de ativação (por exemplo, ReLU). A camada de saída utiliza tipicamente a função *softmax* para tarefas de classificação multiclasse.

No Capítulo 7, DNNs são empregadas como uma das arquiteturas base para a classificação de impacto de fatias, servindo como referência de comparação com as arquiteturas recorrentes.

2.5.2 Redes LSTM

A rede *Long Short-Term Memory* (LSTM) é uma variante de rede neural recorrente (*Recurrent Neural Network* — RNN) projetada para capturar dependências de longo prazo em sequências de dados [29]. O problema fundamental das RNNs tradicionais é o desvanecimento do gradiente durante o treinamento com retropropagação no tempo, o que dificulta o aprendizado de dependências temporais distantes.

A LSTM resolve esse problema por meio de uma célula de memória controlada por três portas: a porta de esquecimento (*forget gate* f_t), que decide quais informações descartar; a porta de entrada (*input gate* i_t), que controla quais novas informações armazenar; e a porta de saída (*output gate* o_t), que determina quais informações da célula produzir como saída. As equações que governam a LSTM são:

$$f_t = \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \tag{2.2}$$
$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{2.3}$$
$$\tilde{C}_t = \tanh(W_C \cdot [h_{t-1}, x_t] + b_C) \tag{2.4}$$
$$C_t = f_t \odot C_{t-1} + i_t \odot \tilde{C}_t \tag{2.5}$$
$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{2.6}$$
$$h_t = o_t \odot \tanh(C_t) \tag{2.7}$$

onde $\odot$ denota o produto elemento a elemento (Hadamard), σ é a função sigmoide e $[h_{t-1}, x_t]$ representa a concatenação do estado oculto anterior com a entrada atual.

No contexto desta tese, LSTMs são avaliadas no Capítulo 7 para a classificação federada de impacto de fatias, dada sua capacidade de modelar dependências temporais presentes em séries de monitoramento de rede.

2.5.3 Redes GRU

A unidade recorrente com porta (*Gated Recurrent Unit* — GRU) é uma simplificação da LSTM que combina as portas de esquecimento e entrada em uma única porta de atualização (*update gate* z_t) e utiliza uma porta de *reset* (r_t) para controlar a influência do estado anterior [30]. As equações da GRU são:

$$z_t = \sigma(W_z \cdot [h_{t-1}, x_t]) \tag{2.8}$$
$$r_t = \sigma(W_r \cdot [h_{t-1}, x_t]) \tag{2.9}$$
$$\tilde{h}_t = \tanh(W \cdot [r_t \odot h_{t-1}, x_t]) \tag{2.10}$$
$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t \tag{2.11}$$

A GRU possui menos parâmetros que a LSTM (duas portas em vez de três, sem célula de memória separada), o que resulta em menor custo computacional de treinamento e inferência. Na prática, GRU e LSTM tendem a apresentar desempenho comparável em diversas tarefas, sendo que a GRU pode ser preferível quando o volume de dados é limitado ou quando o custo computacional é uma restrição [31].

No Capítulo 7, a GRU é empregada como uma das três arquiteturas avaliadas para a classificação federada de impacto, permitindo a comparação entre os *trade-offs* de complexidade do modelo e desempenho em cenários com dados homogêneos e heterogêneos.

2.6 Aprendizado centralizado versus federado

Na abordagem tradicional de aprendizado de máquina centralizado, todos os dados de treinamento são coletados e armazenados em um único local (por exemplo, um servidor na nuvem), onde o modelo é treinado. Embora essa abordagem simplifique o processo de treinamento e permita acesso irrestrito a todos os dados, ela apresenta limitações em cenários de redes móveis: a transferência de grandes volumes de dados brutos para um ponto central consome largura de banda no *backhaul*, introduz latência e pode comprometer a privacidade dos dados dos dispositivos [5].

O Aprendizado Federado (*Federated Learning* — FL) é um paradigma de aprendizado distribuído que permite treinar modelos de aprendizado de máquina de forma colaborativa entre múltiplos dispositivos (clientes), sem a necessidade de centralizar os dados brutos [32]. Cada cliente treina uma cópia local do modelo sobre seus próprios dados e envia apenas as atualizações dos parâmetros (por exemplo, gradientes ou pesos) para um servidor central, que agrega essas atualizações para produzir um modelo global aprimorado. O processo se repete por múltiplas rodadas de comunicação até a convergência.

Essa abordagem traz vantagens em cenários de redes veiculares, onde os dados são gerados de forma distribuída em RSUs e veículos espalhados geograficamente. Ao manter os dados localmente, o FL reduz a carga de comunicação e preserva a localidade dos dados, além de oferecer ganhos de privacidade ao evitar a exposição dos dados brutos [3, 5]. O FL foi reconhecido como uma das soluções mais promissoras para viabilizar a IA onipresente prevista em redes 6G [5].

A Tabela 2 sintetiza as principais diferenças entre as abordagens centralizada e federada.

2.6.1 FedAvg

O *Federated Averaging* (FedAvg), proposto por McMahan et al. [32], é o algoritmo de referência para treinamento federado. Em cada rodada de comunicação t, o servidor

Tabela 2 – Comparativo entre aprendizado centralizado e federado.

Aspecto	**Centralizado**	**Federado**
Localização dos dados	Servidor central	Distribuídos nos clientes
Comunicação	Dados brutos enviados ao servidor	Apenas parâmetros do modelo
Privacidade	Dados centralizados e expostos	Dados permanecem locais
Escalabilidade	Limitada pela largura de banda	Melhor escalabilidade
Heterogeneidade de dados	Não se aplica	Desafio (Non-IID)
Uso nesta tese	Linha de base (Cap. 6 e 7)	Abordagem proposta (Cap. 7)

seleciona um subconjunto de clientes S_t, que recebem o modelo global w_t. Cada cliente $k \in S_t$ executa E épocas de gradiente descendente estocástico (SGD) sobre seu conjunto de dados local $\mathcal{D}_k$, produzindo um modelo local w_t^k. O servidor então agrega os modelos locais por média ponderada pelo tamanho dos conjuntos de dados:

$$w_{t+1} = \sum_{k \in S_t} \frac{|\mathcal{D}_k|}{\sum_{j \in S_t} |\mathcal{D}_j|} w_t^k \tag{2.12}$$

A principal vantagem do FedAvg é sua simplicidade e eficácia quando os dados dos clientes são distribuídos de forma independente e identicamente distribuída (IID). Quando essa premissa é violada — ou seja, em cenários Non-IID, onde cada cliente possui uma distribuição de dados diferente — o FedAvg pode apresentar convergência lenta ou instável [32].

Em cenários de redes veiculares, a distribuição Non-IID é comum, uma vez que RSUs em diferentes localizações geográficas tendem a observar padrões de tráfego distintos. No Capítulo 7, o FedAvg é avaliado em ambos os cenários (IID e Non-IID) com dados de monitoramento de fatias em um corredor veicular com três RSUs.

2.6.2 FedProx

O *Federated Proximal* (FedProx), proposto por Li et al. [33], é uma extensão do FedAvg projetada para lidar com a heterogeneidade de dados (Non-IID) e de sistemas (dispositivos com capacidades computacionais distintas). A ideia central é adicionar um termo de regularização proximal à função objetivo local de cada cliente, que penaliza desvios excessivos em relação ao modelo global:

$$\min_w F_k(w) + \frac{\mu}{2} \|w - w_t\|^2 \tag{2.13}$$

onde $F_k(w)$ é a função de perda local do cliente k, w_t é o modelo global da rodada atual e $\mu \geq 0$ é o coeficiente de regularização proximal. Quando $\mu = 0$, o FedProx se reduz ao FedAvg.

O termo proximal impede que os modelos locais divirjam excessivamente do modelo global, o que pode estabilizar a convergência em cenários Non-IID. A escolha do valor de μ envolve um *trade-off*: valores muito baixos podem não ser suficientes para corrigir a divergência, enquanto valores muito altos podem restringir excessivamente o aprendizado local.

No Capítulo 7, o FedProx é avaliado com $\mu = 0{,}1$ em cenários Non-IID, e os resultados são comparados com o FedAvg para quantificar o ganho efetivo do termo de regularização proximal em dados de monitoramento veicular.

2.7 Considerações sobre o referencial teórico

Este capítulo apresentou os fundamentos teóricos que sustentam as contribuições desta tese. As redes 5G e B5G, com o mecanismo de fatiamento de rede, constituem a infraestrutura base para o suporte a aplicações ITS com requisitos heterogêneos. O gerenciamento do ciclo de vida das fatias demanda abordagens inteligentes que integrem as fases de preparação, planejamento, operação e descomissionamento. As técnicas de aprendizado de máquina supervisionado — em particular modelos baseados em árvores e redes neurais — fornecem as ferramentas para classificação de contexto veicular e impacto de políticas. O aprendizado federado, com as estratégias FedAvg e FedProx, possibilita o treinamento distribuído desses modelos em cenários com dados gerados na borda da rede.

Os capítulos seguintes apresentam as contribuições desta tese, iniciando pelas abordagens de identificação do contexto veicular (Capítulos 3 e 4), avançando para o fatiamento orientado a aplicações (Capítulo 5), a classificação de impacto com o AIMS (Capítulo 6), sua extensão com aprendizado federado (Capítulo 7) e os aspectos de segurança da informação decorrentes da adoção de aprendizado distribuído no gerenciamento de fatias (Capítulo 8).

3 DETECÇÃO DE CONTEXTO VEICULAR COM DADOS PÚBLICOS DE TRANSPORTE

No Capítulo 2, foram apresentados os fundamentos teóricos que sustentam as contribuições desta tese, incluindo conceitos de aprendizado de máquina supervisionado e fatiamento de rede. Este capítulo inicia a sequência de contribuições com a primeira das abordagens de identificação do contexto veicular desenvolvidas nesta tese. A correta identificação do contexto veicular é pré-requisito para que soluções de gerenciamento de fatias possam readequar dinamicamente a infraestrutura alocada, conforme discutido nos desafios D1 (ausência de abordagens que extraiam o contexto veicular a partir de fontes acessíveis para alimentar decisões de gerenciamento de fatias) e D3 (ausência de mecanismos para classificação de impacto de políticas de fatiamento a partir de telemetria veicular), apresentados na Seção 1.1.

O conteúdo deste capítulo é baseado na contribuição publicada em [9][1], que propõe uma abordagem para detecção de congestionamentos urbanos utilizando aprendizado de máquina supervisionado aplicado a dados públicos de mobilidade urbana. A abordagem se diferencia por utilizar dados já disponíveis nos sistemas de transporte público, evitando a necessidade de infraestrutura dedicada de sensoriamento.

O capítulo está organizado da seguinte forma. A Seção 3.1 contextualiza o problema de detecção de congestionamentos e sua relação com o gerenciamento de recursos de comunicação veicular. A Seção 3.2 discute os trabalhos relacionados à detecção de congestionamentos em cenários urbanos. A Seção 3.3 apresenta a abordagem proposta. A Seção 3.4 descreve os experimentos de validação realizados com dados do sistema de transporte do Rio de Janeiro. A Seção 3.5 analisa os resultados obtidos. Por fim, a Seção 3.6 apresenta as considerações finais e a conexão com o capítulo seguinte.

3.1 Contextualização e motivação

A identificação de características dos distritos urbanos em termos de padrões de uso pela população constitui um desafio relevante para o planejamento urbano e o gerenciamento de redes de comunicação [34]. Compreender a dinâmica do tráfego veicular nas cidades é fundamental para otimizar a gestão de recursos em diversas áreas, desde o

[1] Publicado originalmente com o título: “An approach using machine learning and public data to detect traffic jams”.

planejamento urbano até a alocação de recursos de rede [35]. Os veículos modernos estão equipados com uma variedade crescente de sensores capazes de monitorar o ambiente ao redor e o estado operacional do veículo em tempo real [35].

Conforme discutido no Capítulo 2, o crescimento acelerado do tráfego de dados em redes móveis e a diversidade de requisitos das aplicações ITS demandam mecanismos capazes de identificar o contexto veicular em tempo real para subsidiar decisões de alocação de recursos [36, 37].

A eficiência do tráfego urbano é uma preocupação relevante em cidades de grande porte [38]. Quando a capacidade de uma via é excedida, congestionamentos podem ocorrer e, em situações extremas, tornar-se eventos frequentes [39]. Quando usuários de dispositivos móveis se encontram em áreas congestionadas, observa-se uma tendência de aumento no consumo de dados nessas localidades, resultando em degradação da experiência de comunicação. Além dos impactos na comunicação, congestionamentos geram perda de produtividade, poluição atmosférica e aumento do consumo de combustível [40].

A identificação precisa de congestionamentos em tempo real é necessária para aprimorar a eficiência do setor de transportes [41]. A predição de congestionamentos é considerada um dos problemas mais relevantes tanto em países desenvolvidos quanto em desenvolvimento [42]. Aprendizado de Máquina (*Machine Learning* — ML) é um ramo da Inteligência Artificial que permite construir sistemas capazes de operar em ambientes complexos, fornecendo um conjunto de ferramentas para exploração e mineração de múltiplas fontes de dados, incluindo aqueles gerados em redes veiculares [43, 44].

No contexto de gerenciamento de recursos de comunicação veicular, a correta identificação de congestionamentos pode auxiliar as camadas de gerenciamento na tomada de decisão quanto à alocação de recursos, em função das mudanças de topologia e dos níveis de concentração de veículos por localidade. Dado que a maioria dos trabalhos relacionados depende de dados provenientes de sensores instalados na infraestrutura viária, esta contribuição propõe uma abordagem baseada em algoritmos de ML supervisionado aplicados a dados públicos dos sistemas de transporte para detecção de congestionamentos urbanos. O diferencial da solução consiste na avaliação de algoritmos de ML supervisionado aplicados a dados que podem ser obtidos de fontes oficiais e públicas para detecção dinâmica de congestionamentos.

3.2 Trabalhos relacionados à detecção de congestionamentos urbanos

A detecção de congestionamentos veiculares é usualmente realizada pela observação da velocidade média do fluxo de tráfego e do número de veículos por unidade de

comprimento da via [41]. Alguns trabalhos implementam essa detecção considerando o percentual de tempo em que um detector (sensor) é ocupado por veículos e a velocidade média dos veículos que passaram durante um intervalo de tempo específico [42].

Diversas abordagens utilizam dados de táxis como fonte de informação para análise de mobilidade urbana. Em [34], os autores buscam capturar padrões de movimentação de multidões urbanas utilizando táxis como sensores capazes de monitorar o comportamento do tráfego em escala metropolitana. A proposta de [45] consiste em modelar e predizer a evolução de congestionamentos com base em dados de GPS de táxis, empregando técnicas de aprendizado profundo. O trabalho de [46] também utiliza exclusivamente dados de táxis para analisar padrões de mobilidade. Entretanto, táxis são veículos com comportamento dinâmico e sem rotas fixas, de modo que seus padrões de mobilidade podem variar consideravelmente em relação aos demais veículos, comprometendo a representatividade das predições de congestionamento baseadas exclusivamente nessa fonte de dados.

A proposta de [47] emprega *mobile crowdsensing* para detectar anomalias de tráfego a partir de dados de trajetória de ônibus, inferindo condições urbanas de tráfego. Contudo, esse trabalho não trata especificamente da detecção de congestionamentos, e a proposta não é validada com métricas padronizadas de avaliação.

Outra linha de pesquisa baseia-se em redes de sensores sem fio (*Wireless Sensor Networks* — WSN). Em redes de sensores, a densidade de sensores deve estar acima de um limiar específico para manter a cobertura e evitar falhas [48]. Embora o erro de localização dos nós de referência possa ser significativo, eles podem ser utilizados como pontos de referência para a movimentação dos nós de sinalização [49]. O uso de redes de sensores sub-GHz para estimativa de densidade de multidões é proposto por [50]. Em [51], é apresentado um algoritmo para descoberta de padrões frequentes de congestionamento em áreas urbanas, avaliado com dados de sensores de tráfego de Okinawa, Japão. De forma similar, [52] propõe mineração de padrões frequentes para predição de locais congestionados, utilizando registros de tempo de viagem gerados por sensores de via durante quatro semanas. Embora essas abordagens apresentem resultados relevantes, a maioria das pesquisas sobre cobertura de sensoriamento assume que os nós de referência estão em posições corretas, sem considerar componentes de erro na localização dos nós de sensoriamento [53], o que pode comprometer a confiabilidade dos resultados.

Abordagens baseadas em IoT e dados de redes sociais também são exploradas na literatura. A proposta de [42] utiliza um protótipo para modelagem preditiva de congestionamentos em IoT, porém considera apenas o número de veículos como métrica. O trabalho de [54] apresenta uma proposta para mineração de correlações de congestionamento urbano utilizando informações de tráfego extraídas de redes sociais, avaliada com dados da cidade de Chicago. Entretanto, essa abordagem não foca na detecção de congestionamentos a partir de dados dos próprios veículos em movimento.

Em síntese, existem trabalhos relevantes sobre detecção de congestionamentos em cenários urbanos, utilizando abordagens que variam entre sensores urbanos fixos, dados de táxis, redes sociais e redes de sensores sem fio. Há também diversos trabalhos relacionados a ML no ambiente de transportes para variados problemas, incluindo a detecção de congestionamentos, conforme apresentado em [35, 40, 41, 43, 55].

Cabe registrar que, desde a publicação original desta contribuição em 2021, o campo de predição de tráfego evoluiu de forma acelerada com a adoção de arquiteturas de aprendizado profundo. Redes Neurais de Grafos (*Graph Neural Networks* — GNN) passaram a dominar a predição de fluxo em escala de rede viária, modelando explicitamente a topologia das vias [56]. Arquiteturas baseadas em *Transformers* espaciotemporais também alcançaram resultados de estado da arte em *benchmarks* de referência [57]. Levantamentos recentes indicam que abordagens clássicas de ML supervisionado — como as utilizadas nesta contribuição — alcançam tipicamente acurácia na faixa de 75–85% em tarefas de classificação de congestionamentos, enquanto modelos de aprendizado profundo atingem 85–95% [58]. Entretanto, os modelos clássicos mantêm vantagens em cenários com volumes reduzidos de dados e quando a interpretabilidade é requisito [57].

Além disso, o ecossistema de dados de redes sociais utilizado para rotulagem de eventos sofreu mudanças significativas. A partir de 2023, a plataforma Twitter (renomeada para X) restringiu severamente o acesso programático à sua API, impactando pesquisas que dependiam dessa fonte de dados em áreas como transporte e monitoramento urbano [59]. Plataformas alternativas de *crowdsensing*, como Waze e Google Maps, têm sido exploradas como substitutas para validação de eventos de tráfego [58].

Contudo, mesmo diante desses avanços metodológicos, observa-se uma lacuna persistente no que se refere a soluções que utilizem dados públicos e confiáveis dos sistemas de transporte como fonte principal para detecção de congestionamentos com técnicas de ML supervisionado, sem dependência de infraestrutura dedicada de sensoriamento. É essa lacuna que a abordagem apresentada na seção seguinte busca preencher.

3.3 Abordagem proposta para identificação de contexto veicular utilizando dados públicos

Em soluções que buscam manter o atendimento aos requisitos de diferentes aplicações em um ambiente veicular, a identificação adequada do contexto veicular é fundamental para a tomada de decisões de reconfiguração de recursos. Com base nessa premissa, esta contribuição apresenta uma abordagem para identificação do contexto veicular por meio da detecção de congestionamentos urbanos, utilizando uma arquitetura baseada em classificadores de ML alimentados por dados históricos e públicos de mobilidade urbana de ônibus [9].

A proposta utiliza dados de transporte disponibilizados publicamente em tempo real para alimentar algoritmos de ML na detecção de congestionamentos. A Figura 4 apresenta os principais componentes da arquitetura proposta, e a Figura 5 detalha o fluxo de processamento na forma de um fluxograma.

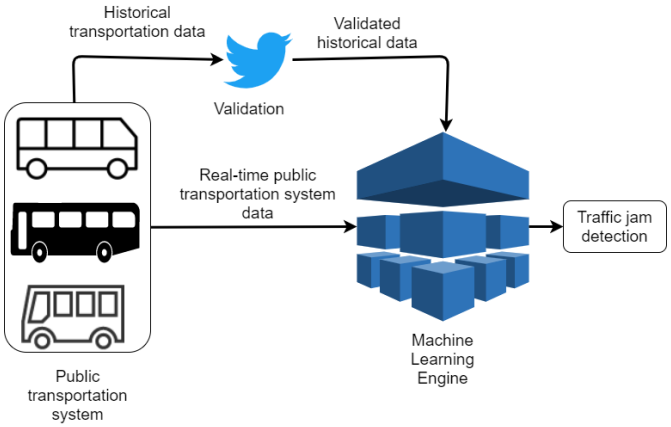


Figura 4 – Arquitetura proposta para identificação de contexto veicular utilizando dados públicos

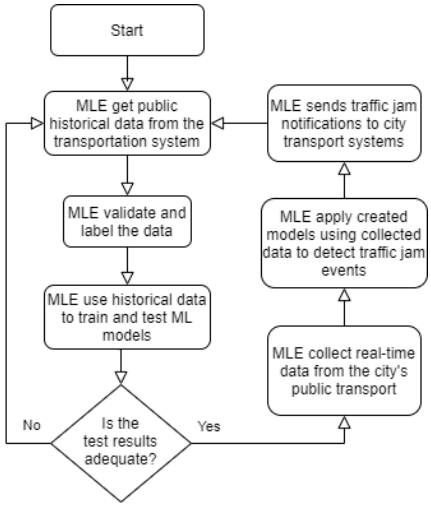


Figura 5 – Fluxograma da abordagem proposta para identificação de contexto veicular utilizando dados públicos

Uma vez que a abordagem emprega algoritmos de aprendizado de máquina supervisionados, é necessária uma fonte de informação para o treinamento desses algoritmos. O *Motor de Aprendizado de Máquina* (*Machine Learning Engine* — MLE) obtém, em primeiro lugar, dados históricos públicos do sistema de transporte. Como esses dados são relacionados a diversos eventos na cidade, é necessário validá-los e rotulá-los de acordo com os eventos sob análise (congestionamentos, no caso desta contribuição). Essa validação pode ser realizada utilizando uma fonte externa de informação, como dados disponíveis nas redes sociais da Prefeitura.

Após a validação e rotulagem, o MLE processa os dados para identificar *features* e classes de ML relevantes para a detecção de congestionamentos. Com as *features* e classes extraídas dos dados históricos, o MLE treina e testa os modelos de ML para criar os classificadores utilizados na predição de eventos de congestionamento. Se os resultados dos testes não forem satisfatórios, o MLE obtém mais dados históricos para criar modelos mais eficientes. Se os modelos forem aprovados, com base em métricas como Acurácia, Precisão, Revocação (*Recall*) e F1-*score*, o MLE passa a coletar dados em tempo real do sistema de transporte público e a utilizar os modelos criados para detectar congestionamentos. Quando um congestionamento é detectado, o MLE envia notificações para os sistemas de gerenciamento de transporte, possibilitando uma gestão de tráfego mais eficiente.

No contexto de integração com uma solução de gerenciamento de recursos de comunicação, como no caso de um *framework* para gerenciamento do ciclo de vida de fatias de rede, essa notificação seria direcionada ao controlador responsável pela operação das fatias, possibilitando a readequação dinâmica dos recursos alocados em resposta às mudanças no contexto veicular. Essa integração entre detecção de contexto e gerenciamento de fatias é consistente com a evolução recente do fatiamento de rede em cenários de cidades inteligentes, onde levantamentos abrangentes identificam a percepção do contexto de tráfego como habilitador para a alocação dinâmica de recursos em redes B5G [60]. Os capítulos subsequentes desta tese exploram essa integração de forma progressiva.

3.4 Experimentos de validação

Esta seção descreve a avaliação de desempenho da abordagem proposta. A Prefeitura do Rio de Janeiro disponibiliza publicamente os dados de GPS dos ônibus do sistema de transporte[2]. Para esta avaliação, foi utilizado um conjunto de dados coletados por outros pesquisadores em outubro de 2014[3]. Os dados compreendem registros enviados pelos ônibus a cada minuto, contendo coordenada GPS, velocidade, número da linha (identificação da rota) e código de identificação do veículo. Foram utilizados dados referentes a sete dias, totalizando 31.189.730 registros. Para a análise de congestionamentos, foram selecionadas duas áreas com movimento significativo de veículos: Lagoa Rodrigo de Freitas e Av. Ayrton Senna, ilustradas na Figura 6 com imagens extraídas do OpenStreetMap[4].

A seleção dessas localidades justifica-se por se tratarem de trechos percorridos por parcela significativa da população que transita entre as zonas sul e oeste da cidade do Rio de Janeiro, tanto de carro quanto de ônibus. No período de análise, essas vias não possuíam o Sistema Rápido de Ônibus (BRS), que consiste em uma faixa dedicada para ônibus e táxis. A ausência do BRS implica que os ônibus enfrentam condições de tráfego

[2] http://www.data.rio/datasets/gps-dos-%C3%B4nibus
[3] https://crawdad.org/coppe-ufrj/RioBuses/20180319
[4] https://www.openstreetmap.org/



(a) Lagoa Rodrigo de Freitas (b) Av. Ayrton Senna

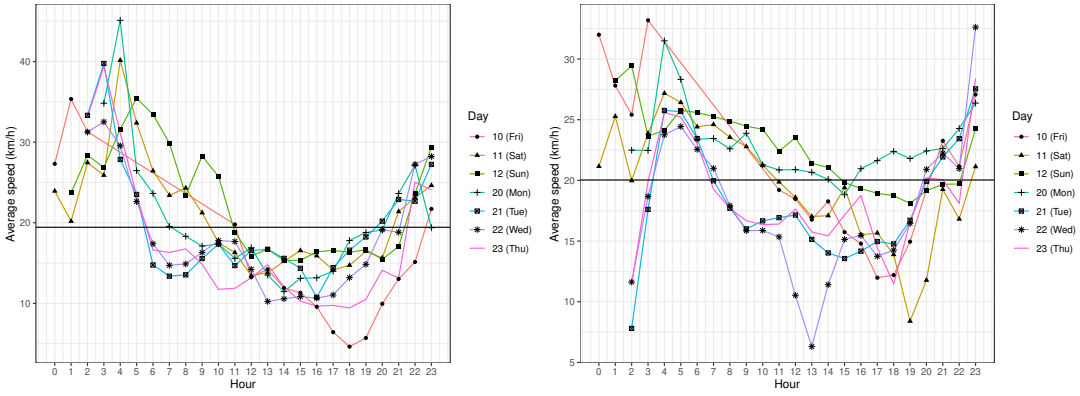
Figura 6 – Duas regiões na cidade do Rio de Janeiro de onde os dados foram analisados - extraídos do OpenStreetMap.

mais próximas às dos veículos comuns, embora tenham que realizar paradas para embarque e desembarque de passageiros. A partir dos dados de GPS, foram extraídos os registros referentes às áreas selecionadas para compor o conjunto de dados utilizado no treinamento dos modelos de ML, sendo 213.011 registros relativos à Lagoa Rodrigo de Freitas e 644.506 relativos à Av. Ayrton Senna.

As *features* definidas para a construção dos modelos de ML foram: velocidade média, número de registros de GPS, número de rotas associadas aos ônibus e número de ônibus por hora do dia, para cada uma das duas regiões analisadas. As Figuras 7, 8, 9 e 10 apresentam, respectivamente, a distribuição dessas *features* ao longo dos dias avaliados. A rotulagem dos dados extraídos dos registros de GPS foi realizada com base em registros históricos de congestionamentos nas áreas analisadas, publicados no Twitter pela Prefeitura[5]. O perfil oficial da Prefeitura (*@OperacoesRio*) divulga, com marcação de data, hora e localização, a ocorrência de congestionamentos na cidade. Esses registros foram utilizados como *ground truth*: para cada janela horária e região analisada, atribuiu-se o rótulo "congestionado" quando havia registro correspondente de congestionamento divulgado pelo perfil para aquela localidade e período, e "não congestionado" caso contrário. Dessa forma, os dados públicos de mobilidade dos ônibus (atributos de entrada) foram associados a rótulos derivados de uma fonte oficial independente.

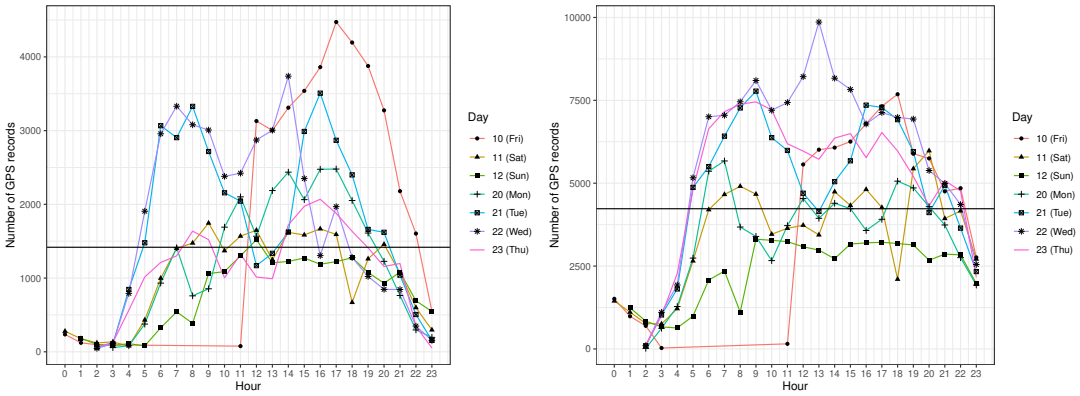
Em relação à velocidade média (Figura 7), observa-se que, em ambas as regiões,

[5] https://twitter.com/OperacoesRio



(a) Lagoa Rodrigo de Freitas (b) Av. Ayrton Senna

Figura 7 – Velocidade média por hora

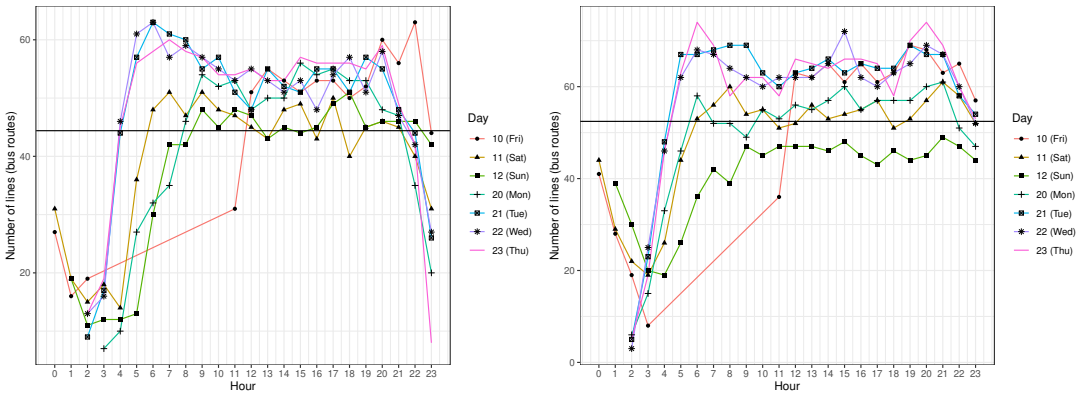


(a) Lagoa Rodrigo de Freitas (b) Av. Ayrton Senna

Figura 8 – Número de registros de GPS por hora

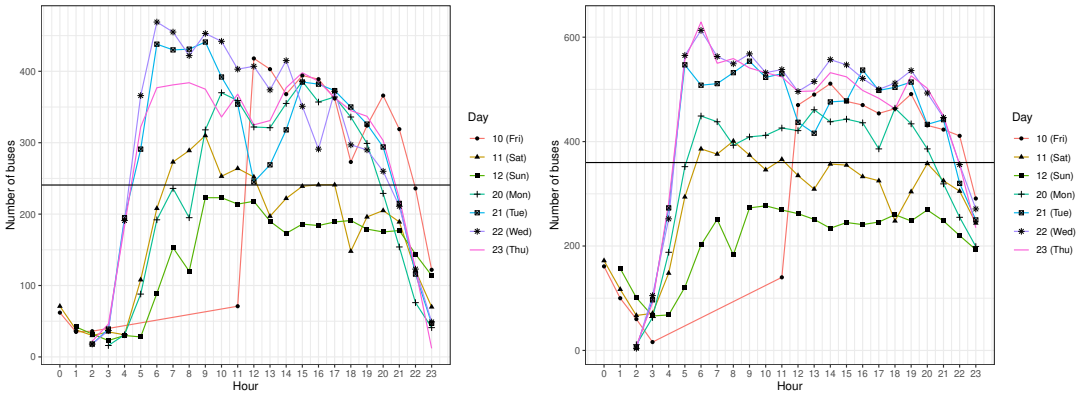
a velocidade diminui a partir das 06:00, horário em que se inicia o período de pico. A média permanece abaixo da média diária, representada como uma linha horizontal preta nos gráficos, até aproximadamente as 20:00. Nos fins de semana, a velocidade média é notavelmente superior. Como os ônibus enviam um registro de GPS a cada minuto, quanto mais intenso o congestionamento, maior o número de registros recebidos por hora em cada área. A Figura 8 confirma que o número de registros de GPS por hora nas áreas avaliadas cresce ao longo do dia, reduzindo-se apenas por volta das 21:00.

Cada ônibus possui um número de linha associado à rota que percorre. O atributo "número de rotas" corresponde, portanto, à quantidade de linhas de ônibus distintas observadas em cada região durante uma dada janela horária, ou seja, quantas rotas diferentes tinham ao menos um ônibus presente na área naquele período. Os ônibus de cada linha circulam periodicamente para atender à demanda de passageiros. Em função da dinâmica do tráfego, eventos de congestionamento podem atrasar os horários, e ônibus de uma mesma rota acabam retidos simultaneamente na mesma região. A Figura 9 apresenta o número de linhas associadas aos ônibus em cada hora nos dias avaliados. Observa-se um



(a) Lagoa Rodrigo de Freitas (b) Av. Ayrton Senna

Figura 9 – Número de linhas (rotas de ônibus)



(a) Lagoa Rodrigo de Freitas (b) Av. Ayrton Senna

Figura 10 – Número de ônibus por hora

número menor de linhas fora do horário comercial, o que é esperado dado que a demanda por transporte público é menor nesses períodos.

É possível identificar distorções nas Figuras 8, 9 e 10 relativas ao dia 10, no intervalo entre 03:00 e 11:00, o que configura um problema nos dados utilizados, provavelmente associado a falhas na coleta. Como os demais períodos desse dia e todos os períodos dos outros dias não apresentam esse tipo de anomalia, o impacto nos resultados é limitado. Dado que o Rio de Janeiro é uma cidade de grande porte, o número de ônibus em circulação durante o dia é elevado. Conforme ilustrado na Figura 10, há uma média de aproximadamente 250 ônibus por dia na região da Lagoa e 380 na Av. Ayrton Senna.

3.4.1 Pré-processamento e configuração dos modelos

Os dados foram normalizados antes de sua utilização com os algoritmos de ML, dado que as *features* estavam em escalas distintas. O processo de normalização consiste em redimensionar os valores para que possuam as propriedades de uma distribuição gaussiana com $\mu = 0$ e $\sigma = 1$, onde μ é a média e σ o desvio padrão. Esse processo foi aplicado

pela fórmula $z = \frac{x-\mu}{\sigma}$, onde z é o valor normalizado e x é a observação original. Para o treinamento dos modelos, foi utilizada validação cruzada *k-fold* com $k = 10$, a fim de mitigar problemas de *overfitting*.

Os seguintes modelos de ML supervisionado foram implementados em Python utilizando a biblioteca Scikit-learn: Regressão Logística (*Logistic Regression* — LR), K-Nearest Neighbor (KNN), Árvore de Decisão (*Decision Tree* — DT), Random Forest (RF) e Máquina de Vetor de Suporte (*Support Vector Machine* — SVM). Além da acurácia, que corresponde à proporção de registros corretamente classificados, foram utilizadas como métricas a Precisão, a Revocação (*Recall*) e o F1-*score*, definidas conforme as Equações (3.1), (3.2), (3.3) e (3.4), onde TP = Verdadeiros Positivos, TN = Verdadeiros Negativos, FP = Falsos Positivos e FN = Falsos Negativos.

$$Acurácia = \frac{TP + TN}{TP + TN + FN + FP} \tag{3.1}$$

$$Precisão = \frac{TP}{TP + FP} \tag{3.2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3.3}$$

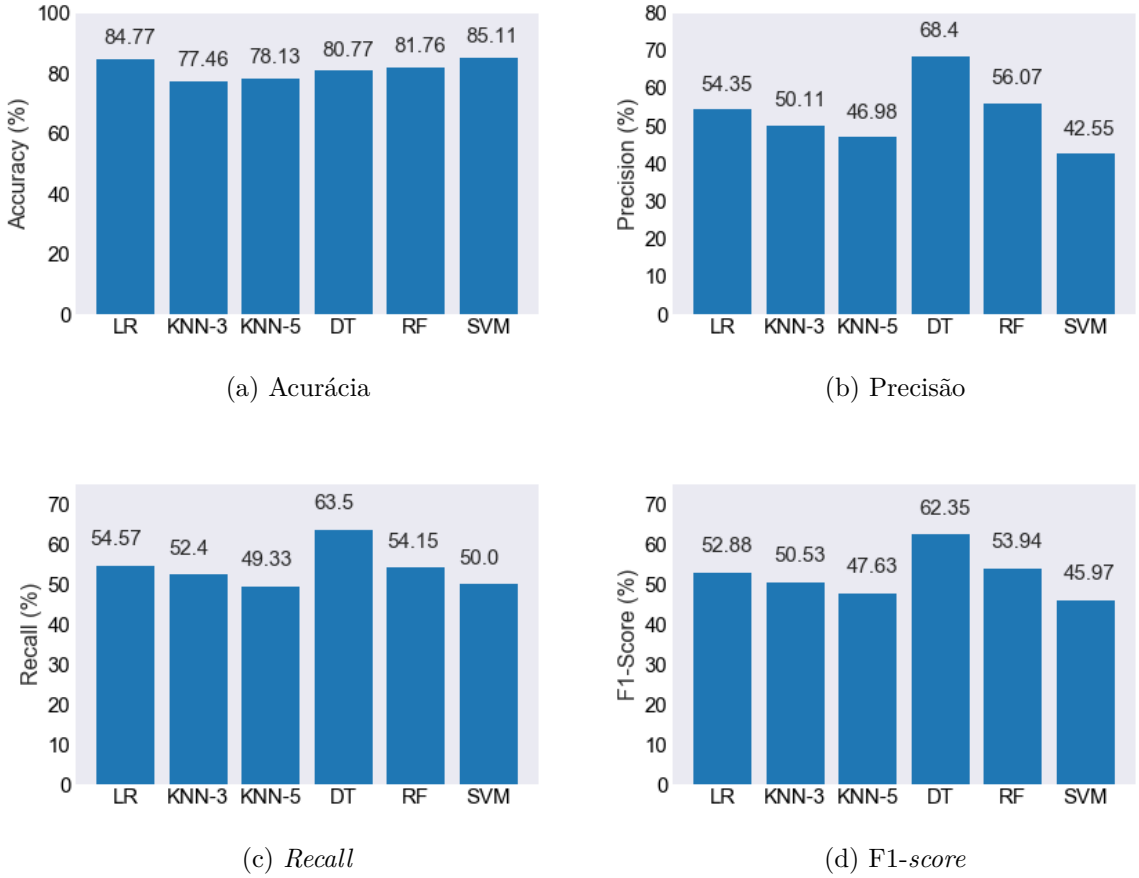
$$F1\text{-}score = \frac{2 * Precisão * Recall}{Precisão + Recall} \tag{3.4}$$

3.5 Resultados

Os resultados de acurácia, precisão, revocação (*recall*) e F1-*score* obtidos com os modelos de ML avaliados (LR, KNN-3, KNN-5, DT, RF e SVM) são apresentados na Figura 11.

A acurácia alcançada foi superior a 80% com quase todos os algoritmos, com exceção do KNN, que constitui o método de aprendizado baseado em instâncias mais simples para regressão e classificação [61]. Os hiperparâmetros do classificador KNN foram ajustados utilizando configurações com $k = 3$, $k = 5$ e $metric = `minkowski'$.

DT é uma abordagem de ML supervisionado para classificação baseada em um conjunto de regras condicionais que favorece a interpretabilidade do modelo [61]. RF, por sua vez, é um modelo de aprendizado de conjunto (*ensemble*) que utiliza a árvore de decisão como classificador base. Nesta avaliação, RF foi configurado com 600 estimadores e DT com o critério *gini*. Cabe observar que os resultados obtidos com RF podem ser contraditórios quando aplicados em cenários distintos [25]. Nos resultados obtidos, DT apresentou o melhor desempenho nas métricas de precisão, revocação e F1-*score*.



(a) Acurácia

(b) Precisão

(c) *Recall*

(d) F1-*score*

Figura 11 – Resultados consolidados obtidos com os modelos de ML avaliados (LR, KNN-3, KNN-5, DT, RF e SVM) para acurácia, precisão, revocação e F1-*score*.

Esse resultado pode ser atribuído ao fato de o conjunto de dados utilizado ser pequeno (302 instâncias) e desbalanceado (45 instâncias representando congestionamentos). Esse comportamento é consistente com os achados de [62], que demonstram que DT (J48) tende a apresentar melhor desempenho com conjuntos de dados pequenos, com resultados de F1-*score* superiores aos de RF em alguns desses cenários.

SVM foi originalmente desenvolvido para tarefas de classificação binária, tendo sido posteriormente estendido para regressão e classificação multiclasse [27]. Nesta avaliação, o modelo foi configurado com o *kernel* RBF (*Radial Basis Function*), alcançando os melhores resultados de acurácia. LR obteve o segundo melhor desempenho nessa métrica. LR é uma abordagem clássica para adaptar a regressão linear a tarefas de classificação e permanece em uso em trabalhos recentes envolvendo aprendizado em redes sem fio [63, 64].

Embora a acurácia seja amplamente utilizada como métrica de avaliação para problemas de classificação, ela possui limitações conhecidas quando a distribuição das classes é desbalanceada [65]. Assim, o fato de SVM apresentar o melhor resultado em acurácia e o pior nas demais métricas não é contraditório. Esses resultados são explicados pelo comportamento documentado na literatura: quando a distribuição de classes é assimétrica,

classificadores como SVM tendem a desconsiderar dados da classe minoritária, tratando-os como ruído [66]. Nesta avaliação, a proporção entre classes é de aproximadamente 15% (45 instâncias de congestionamento em 302 registros). Técnicas de reamostragem como SMOTE, embora amplamente utilizadas em conjuntos de dados maiores, apresentam limitações quando aplicadas a conjuntos pequenos, pois a classe minoritária pode não conter amostras suficientes para a geração de padrões sintéticos representativos [67]. Abordagens alternativas, como aprendizado sensível ao custo (*cost-sensitive learning*), poderiam ser exploradas em trabalhos futuros para mitigar o impacto do desbalanceamento sem exigir aumento no volume de dados.

Em síntese, considerando todas as métricas de avaliação em conjunto, o melhor classificador foi obtido com o algoritmo DT. A aplicação de DT aos dados de transporte enviados pelos ônibus, combinados com eventos de congestionamento publicados pela Prefeitura em redes sociais, possibilitou a criação de uma solução com acurácia elevada e desempenho superior em precisão, revocação e F1-*score* em comparação com os demais algoritmos avaliados.

3.6 Considerações finais

Os resultados obtidos demonstram a viabilidade do uso de técnicas de ML para detecção de congestionamentos a partir de dados públicos do sistema de transporte. A acurácia de aproximadamente 80% obtida com a maioria dos algoritmos é consistente com a faixa de desempenho documentada na literatura para abordagens de ML clássico aplicadas à classificação de congestionamentos [58]. Trabalhos recentes que adotam abordagem similar — classificação de congestionamentos a partir de dados de GPS de ônibus com algoritmos supervisionados — confirmam a viabilidade do conceito, alcançando acurácia superior a 96% com conjuntos de dados maiores (superiores a 20 mil instâncias) e a inclusão de métodos de *ensemble* como XGBoost [68]. Essa comparação evidencia que o principal fator limitante desta avaliação reside no volume reduzido do conjunto de dados (302 instâncias, das quais 45 representam congestionamentos), e não na inadequação da abordagem proposta.

Apesar dos benefícios demonstrados, é necessário reconhecer as limitações intrínsecas a esta fase da pesquisa. Além do volume reduzido de dados, a interpretação eficiente dos dados públicos constitui um desafio contínuo, pois requer a definição clara dos eventos de congestionamento e uma fonte confiável que forneça dados consistentes do sistema de transporte. A utilização de conjuntos de dados mais amplos, de abordagens complementares para rotulagem de eventos ou ainda de outras fontes de dados possibilitará avaliar a criação de classificadores mais robustos para diferentes eventos no ambiente veicular, contribuindo para a readequação da infraestrutura de comunicação em resposta à dinâmica do contexto

de mobilidade.

Nesse sentido, o capítulo seguinte apresenta uma abordagem complementar que investiga o uso de dados de outra natureza — especificamente, estatísticas de tráfego de rede — para predição do contexto veicular, buscando superar a dependência de fontes externas de dados que caracteriza a abordagem apresentada neste capítulo.

4 PREDIÇÃO DE CONTEXTO VEICULAR POR ESTATÍSTICAS DE REDE DE COMUNICAÇÃO

No capítulo anterior, demonstrou-se a viabilidade de utilizar dados públicos de sistemas de transporte para identificar o contexto veicular, como a ocorrência de congestionamentos, sem depender de sensores embarcados nos veículos [9]. Apesar dos resultados promissores, aquela abordagem possui uma limitação inerente: a dependência de fontes externas de dados que podem não estar disponíveis ou não apresentar a frequência de atualização necessária para decisões em tempo real na infraestrutura de rede.

Redes veiculares modernas, com suporte a comunicações V2X (*Vehicle-to-Everything*), geram continuamente dados de tráfego de rede como subproduto natural da operação das aplicações [69, 70]. Medições como taxa de transferência (*throughput*) e tempo de ida e volta (RTT, do inglês *Round-Trip Time*) são inerentes à comunicação entre veículos e servidores de aplicação, podendo ser obtidas diretamente no núcleo da rede por meio de controladores SDN (*Software-Defined Networking*) sem necessidade de infraestrutura adicional de sensoriamento [71, 72].

A premissa que sustenta este capítulo é que mudanças no contexto veicular — como o surgimento de congestionamentos — produzem impactos mensuráveis no padrão de tráfego gerado pelas aplicações. À medida que a densidade de veículos aumenta, a contenção nos recursos de comunicação sem fio altera o perfil de métricas de rede observadas nos servidores. Se esses padrões puderem ser capturados por classificadores de aprendizado de máquina (ML, do inglês *Machine Learning*), a rede de comunicação pode funcionar como um sensor implícito do contexto veicular [43, 73]. Essa premissa foi corroborada por estudos recentes que demonstram a viabilidade de predizer QoS veicular a partir de medições de rede em camada de transporte, inclusive com dados de campo [74].

Este capítulo apresenta a segunda abordagem de identificação de contexto desenvolvida nesta tese, publicada em [10][1]. A investigação utiliza dados intrínsecos à rede de comunicação — especificamente medições de RTT e taxa de transferência em servidores MEC (*Multi-access Edge Computing*) — para predição de níveis de congestionamento veicular. Uma prova de conceito foi conduzida com um cenário urbano emulado contendo 158 veículos, e os dados gerados foram utilizados para treinar e avaliar cinco algoritmos

[1] Publicação original: “Machine Learning Applied to Network Traffic for Vehicular Applications to Predict the Vehicle Context”, IEEE VTC-Fall 2021.

de ML supervisionados. Os resultados indicam que classificadores como Random Forest, Decision Tree e KNN são capazes de predizer o nível de congestionamento com acurácia superior a 85%, utilizando apenas dados de comunicação de rede.

Enquanto a abordagem do Capítulo 3 utiliza dados públicos como fonte principal, a abordagem deste capítulo é complementar, empregando dados que já estão disponíveis na própria infraestrutura de rede. A integração dessas duas perspectivas — dados externos e dados intrínsecos — fornece a base para o módulo de identificação de contexto que será utilizado no framework de fatiamento apresentado no Capítulo 5.

4.1 Trabalhos relacionados

A utilização de redes de sensores sem fio para estimar a densidade de multidões foi proposta por Denis et al. [50], com experimentos em um festival de música de grande porte demonstrando a viabilidade de estimar a densidade com alta acurácia. Entretanto, não há evidências de que a solução possa ser aplicada a cenários veiculares, nos quais a mobilidade dos nós introduz desafios distintos. Lee et al. [34] buscaram capturar padrões de movimentação urbana utilizando táxis como sensores distribuídos pela cidade, mas a abordagem não analisa o impacto no desempenho de redes veiculares.

No domínio de QoS (*Quality of Service*) em redes sem fio, Moura et al. [75] empregaram redes neurais recorrentes para estimar a qualidade de serviço em estações Wi-Fi. Apesar de utilizarem ML, os cenários avaliados não consideram mobilidade, que é um fator determinante em redes veiculares. Yazar e Arslan [44] propuseram um mecanismo baseado em ML para seleção de parâmetros de forma de onda em redes 5G, porém sem considerar ambientes veiculares. Trinh et al. [76] aplicaram redes LSTM à predição de tráfego de rede móvel a partir de dados coletados em eNodeBs na cidade de Barcelona, mas consideram apenas o canal físico sem incluir métricas relevantes para o desempenho das aplicações, como RTT e taxa de transferência.

Abordagens voltadas especificamente ao contexto veicular incluem o trabalho de Vahdat-Nejad et al. [77], que apresenta um levantamento sobre aplicações de redes veiculares cientes de contexto, e Chavhan et al. [78], que propõem um sistema de transporte público inteligente baseado em IoT com decisões descentralizadas por regiões metropolitanas. Kulkarni et al. [79] propuseram um algoritmo de predição de congestionamento baseado em médias para IoT, mas não apresentam resultados consistentes na avaliação. Em trabalho anterior desta tese, foi proposta uma abordagem usando ML aplicado a dados públicos de sistemas de transporte para detectar congestionamentos sem depender de sensores embarcados [9], porém com a limitação de depender de fontes externas de dados.

No contexto de controle de congestionamento em redes veiculares, Taherkhani e Pierre [80] propuseram uma estratégia centralizada e localizada utilizando o algoritmo K-

means, mas não consideram a detecção do contexto veicular e sua relação com o desempenho da rede como um todo. Trabalhos como os de Popoola et al. [81] e Zhao et al. [82] aplicam abordagens avançadas de ML (aprendizado profundo híbrido e autoencoders dinâmicos, respectivamente) para detecção de eventos de segurança em redes IoT, mas não abordam ambientes móveis veiculares.

Mais recentemente, Palaios et al. [74] validaram o uso de ML aplicado a medições de rede em camada de transporte (RTT, taxa de transferência) para predição de QoS em comunicação veicular celular, utilizando dados reais coletados em rodovias na Alemanha. Seus resultados confirmam a viabilidade da abordagem investigada neste capítulo, ao mesmo tempo em que identificam que a estratégia de particionamento dos dados de avaliação tem impacto significativo nas métricas reportadas — uma consideração metodológica relevante para trabalhos futuros. Em um levantamento abrangente, Hamdi et al. [83] mapearam o estado da arte de técnicas de ML aplicadas a fatiamento de rede para IoV em redes 5G e além, confirmando a predição de contexto veicular a partir de dados de rede como uma direção de pesquisa ativa e em expansão.

Em síntese, embora a literatura apresente contribuições relevantes em aspectos individuais do contexto veicular, persiste a necessidade de soluções que utilizem dados de tráfego de rede — já disponíveis na infraestrutura — para inferir o contexto veicular e subsidiar decisões de gerenciamento de recursos de forma integrada. A contribuição deste capítulo consiste na avaliação de ML aplicado às medições de comunicação de rede de aplicações veiculares para predição do contexto veicular, constituindo uma abordagem que fornece informações úteis para o atendimento dinâmico dos requisitos de aplicações ITS.

4.2 Abordagem proposta para identificação de contexto veicular utilizando estatísticas de rede

A abordagem proposta emprega ML aplicado a medições realizadas sobre dados brutos de comunicação de aplicações veiculares para predizer eventos no contexto veicular. Essas medições podem ser realizadas por um controlador de rede (NC, do inglês *Network Controller*), considerando uma rede veicular habilitada para SDN [70]. A viabilidade dessa arquitetura é corroborada por levantamentos recentes que documentam a consolidação de plataformas integradas SDN/NFV/MEC para redes veiculares, cobrindo tanto IEEE 802.11p quanto C-V2X [84]. A Figura 12 apresenta os principais componentes da solução proposta, detalhados a seguir.

A solução utiliza algoritmos de ML supervisionados, que requerem uma fonte de informação para treinamento. Essa fonte consiste em um banco de dados com registros históricos de medições de rede associadas a eventos conhecidos no contexto veicular. O Motor de Aprendizado de Máquina (MLE, do inglês *Machine Learning Engine*) é a entidade

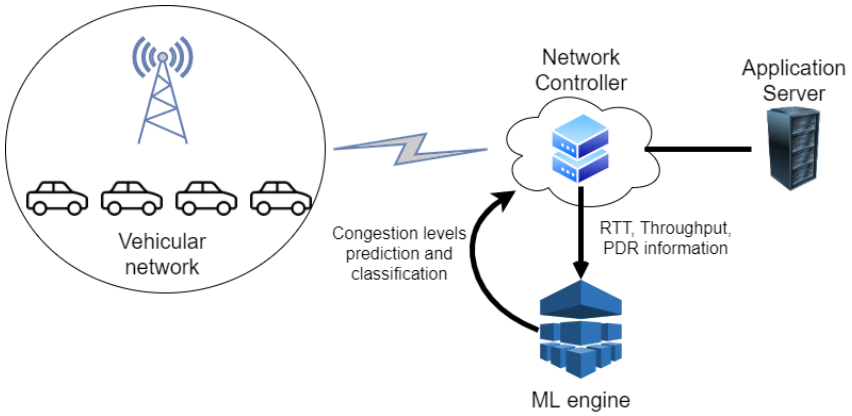


Figura 12 – Abordagem proposta com o uso de dados estatísticos de operação da rede veicular.

responsável por coletar os dados históricos do banco de dados e processá-los para identificar características (*features*) e classes de eventos relevantes. Com as características e classes extraídas, o MLE treina os modelos de ML para criar os classificadores que serão utilizados para predizer eventos futuros do contexto veicular.

Antes da utilização dos classificadores, o MLE os avalia por meio de métricas como acurácia, precisão (*precision*), revocação (*recall*) e F1-*score*, selecionando os melhores classificadores para cada classe de evento. Em operação, o controlador de rede coleta medições em tempo real dos pacotes de comunicação das aplicações veiculares e as encaminha ao MLE, que as utiliza como entrada nos classificadores treinados para identificar novas ocorrências de eventos no contexto veicular.

Após cada classificação, o MLE notifica o controlador de rede sobre o evento predito. O controlador avalia essa informação e, com base no contexto veicular atualizado, executa as ações necessárias para aplicar as políticas de rede adequadas ao atendimento dos requisitos de comunicação das aplicações veiculares. Periodicamente, a solução pode verificar se há necessidade de incorporar novos tipos de evento ao modelo.

4.3 Avaliação de desempenho

A avaliação de desempenho da abordagem proposta foi organizada em três fases: (1) a emulação de um cenário de congestionamento de tráfego, quantificado pela métrica de densidade de tráfego, definida como o número de veículos por unidade de área [85]; (2) a extração de características e definição de classes de congestionamento a partir do conjunto de dados gerado na fase 1; e (3) o treinamento e teste dos modelos de ML para criação dos classificadores responsáveis pela predição de eventos no contexto veicular.

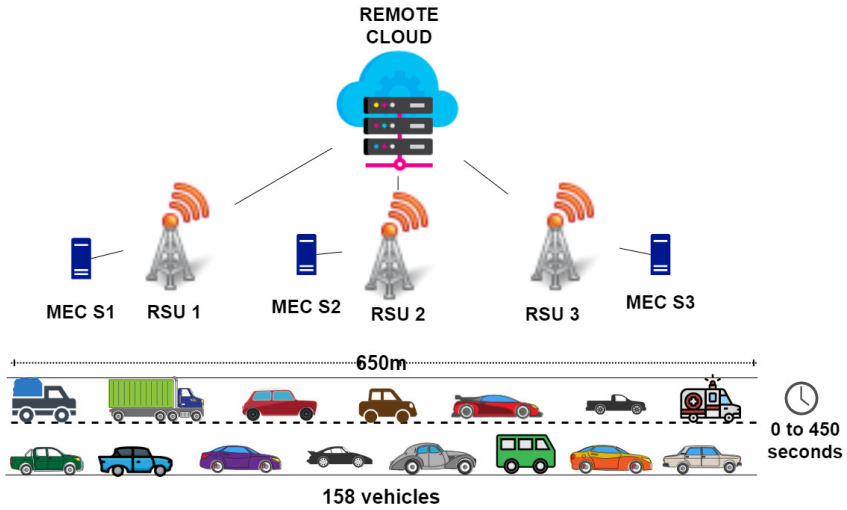


Figura 13 – Cenário de avaliação representando um congestionamento de tráfego com diferentes níveis de consumo de recursos em cada RSU ao longo do tempo.

4.3.1 Implementação do cenário de congestionamento de tráfego

A fase 1 da experimentação utilizou o emulador de rede sem fio Mininet-WiFi [86], um *fork* do emulador Mininet [87]. O cenário de avaliação, ilustrado na Figura 13, representa um congestionamento de tráfego com densidade crescente de veículos por m^2 ao longo do tempo — uma situação frequente no tráfego urbano.

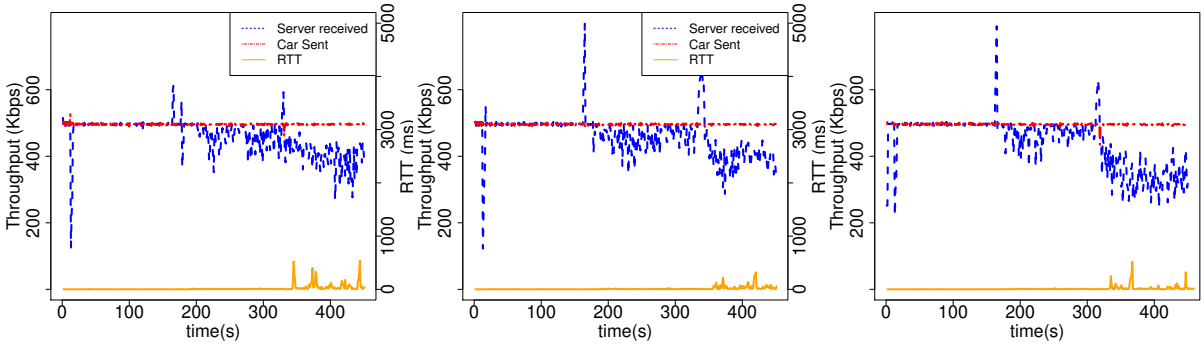
A implementação segue trabalhos como [11], [38] e [88], que avaliaram o desempenho de suas propostas com diferentes níveis de densidade veicular. Os dados foram coletados durante 450 segundos de avaliação com 158 veículos seguindo um padrão de mobilidade de congestionamento configurado no simulador de mobilidade urbana SUMO [89], em uma estrada de 650 m em Manhattan (Nova Iorque), resultando em diferentes níveis de congestionamento ao longo do tempo. O SUMO foi integrado ao Mininet-WiFi para construir a rede de comunicação veicular por meio de emulação SDN sem fio realista. A distância entre os RSUs (*Road-Side Units*) é de 500 m, e a área de cobertura do sinal corresponde a 250 m, conforme [90]. Os parâmetros utilizados nesta fase estão resumidos na Tabela 3.

Todos os 158 veículos geraram fluxos de dados em diferentes taxas de transmissão, com até 50 veículos comunicando-se simultaneamente. Desses, 15 veículos geraram dados para servidores MEC com fluxos de comunicação utilizando 500 Kbps cada. As métricas do conjunto de dados final foram derivadas a partir dos dados coletados nesses servidores MEC, cuja comunicação é menos suscetível a atrasos em comparação com servidores em nuvem remota. No entanto, problemas relacionados à interferência sem fio persistem: à medida que o número de veículos no congestionamento aumenta, espera-se que o impacto na comunicação também aumente.

Utilizou-se uma configuração sem QoS na rede para os dados das aplicações, de

Tabela 3 – Parâmetros utilizados no cenário de avaliação de desempenho com estatísticas de rede.

Parâmetro	**Valor**
Número de veículos	158
Número de RSUs	3
Alcance dos RSUs	250 m
Servidores de aplicação	3
Modelo de Propagação	Log Distance
Camada MAC da RAN	IEEE 802.11g
Taxa de dados por veículo	2 Mbps
Número de aplicações	3
Largura de banda de *upload* dos RSUs	5 Mbps
Duração da emulação	450 segundos



(a) Rodada 1 (b) Rodada 2 (c) Rodada 3

Figura 14 – Medições de *throughput* e RTT coletadas na infraestrutura MEC durante a implementação do cenário de congestionamento de tráfego em 3 rodadas de emulação.

modo que todos os pacotes foram tratados com a mesma prioridade [91]. A intenção com essa configuração foi garantir que o congestionamento de tráfego veicular impactasse diretamente a comunicação de rede, permitindo que as medições coletadas refletissem as condições reais do contexto veicular para uso nos algoritmos de ML.

4.3.2 Definição das características e classes de ML

Com a implementação do cenário de congestionamento, gerou-se um conjunto de dados com medições de rede durante os fluxos de comunicação das aplicações que utilizaram servidores MEC. As medições coletadas foram o tempo de ida e volta (RTT) e o *throughput* dos dados enviados pelos veículos e recebidos pelos servidores de aplicação. O *throughput* é uma métrica utilizada em trabalhos de avaliação de redes veiculares modernas em cenários 5G [70]. O RTT é definido como o tempo gasto por uma aplicação, partindo do nó iniciador (veículo de origem), para enviar uma mensagem e receber uma resposta da rede central [92].

Definiu-se, portanto, um conjunto de duas características (RTT e *throughput* recebido nos servidores) e coletaram-se 1.350 observações de cada característica no experimento de congestionamento, distribuídas em três rodadas de emulação durante os 450 segundos de avaliação, conforme ilustrado nas Figuras 14 (a), (b) e (c).

Formalmente, sendo P o espaço amostral de um evento de congestionamento, P consiste em n amostras, e Y_i representa as características da i-ésima amostra. Assim, $P = \{Y_1, Y_2, \ldots, Y_n\}$. Para cada amostra, há f características, registradas como $Y_i = \{F_{i1}, F_{i2}, \ldots, F_{if}\}$. O conjunto de dados resultante contém 1.350 observações de congestionamento, cada uma com duas características. Como o congestionamento aumenta progressivamente durante o tempo de avaliação, definiram-se três classes: nos primeiros 175 s, o nível de congestionamento foi rotulado como classe 1; nos 150 s seguintes (de 175 s a 325 s), como classe 2; e no intervalo final (de 325 s a 450 s), como classe 3. Esses rótulos foram inseridos no conjunto de dados para treinar e testar os algoritmos de ML na predição dos diferentes níveis de congestionamento.

4.3.3 Treinamento e teste dos modelos de ML

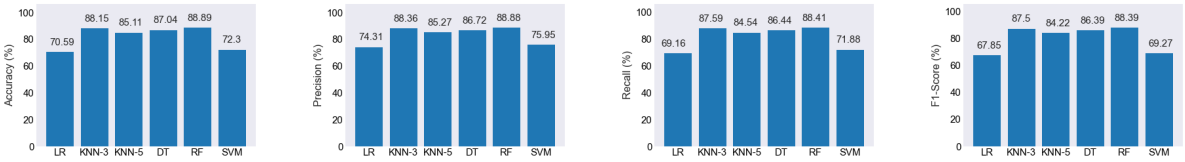
Os dados foram normalizados antes do treinamento pelo mesmo procedimento de padronização *z-score* descrito na Seção 3.4.1, uma vez que as características possuem escalas distintas. Para evitar problemas de *overfitting*, aplicou-se validação cruzada *k-fold* com $k = 10$.

Utilizaram-se os mesmos cinco algoritmos de ML supervisionados empregados no Capítulo 3 — LR, KNN, DT, RF e SVM —, implementados em Python com a biblioteca scikit-learn, com os hiperparâmetros do RF (600 estimadores) e do DT (critério *gini*) mantidos idênticos. As métricas de avaliação — acurácia, precisão, revocação (*recall*) e F1-*score* — seguem as definições apresentadas na Seção 3.4.1. Embora abordagens de aprendizado profundo — como LSTM, GNN e Transformers — tenham se consolidado como estado da arte para predição de tráfego de rede em cenários com grandes volumes de dados e padrões espaço-temporais complexos [93], a escolha de algoritmos supervisionados tradicionais neste trabalho é justificada pela dimensionalidade reduzida do espaço de características (duas *features*) e pelo objetivo de estabelecer uma linha de base interpretável para a tarefa de classificação proposta.

4.4 Resultados

Os resultados de acurácia, precisão, revocação e F1-*score* obtidos com os modelos avaliados (LR, KNN-3, KNN-5, DT, RF e SVM) são apresentados na Figura 15.

Os resultados obtidos com LR apresentaram acurácia de 70,59% e F1-*score* de 67,85%, indicando desempenho limitado na tarefa de classificação multiclasse proposta.



(a) Acurácia (b) Precisão (c) *Recall* (d) F1-*score*

Figura 15 – Resultados consolidados obtidos com os modelos LR, KNN-3, KNN-5, DT, RF e SVM para as métricas de acurácia, precisão, *recall* e F1-*score*.

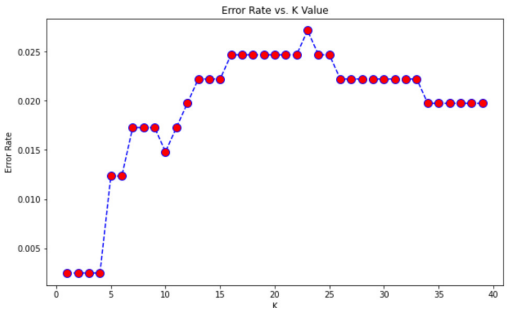


Figura 16 – Taxa de erro em função de k com os dados de estatísticas coletadas na rede de comunicação no cenário veicular avaliado.

Os hiperparâmetros do classificador KNN foram ajustados para otimizar a precisão [94], utilizando-se duas configurações para o parâmetro k que, no contexto do KNN, define o número de vizinhos mais próximos considerados na classificação de cada amostra, após análise da taxa de erro em função de k. Os modelos foram configurados com $k = 3$ e $k = 5$, ambos com metric = 'minkowski'. A Figura 16 apresenta a taxa de erro em função do parâmetro k. O KNN-3 obteve acurácia de 88,15%, superior à da LR, mas sua precisão (74,31%) e revocação (69,16%) ficaram abaixo dos valores obtidos pela LR nessas métricas. O KNN-5 apresentou comportamento intermediário, com acurácia de 85,11% e revocação de 94,54%.

Os classificadores baseados em árvores apresentaram os melhores resultados. O RF obteve os melhores valores em todas as métricas avaliadas: acurácia de 88,89%, precisão de 88,88%, revocação de 88,41% e F1-*score* de 88,39%. O DT apresentou resultados próximos, com valores acima de 86% em todas as métricas.

O SVM obteve desempenho inferior ao dos classificadores baseados em árvores (DT e RF), com acurácia de 72,30% e F1-*score* de 69,27%.

Em termos consolidados, os classificadores RF, DT, KNN-3 e KNN-5 obtiveram acurácia superior a 85%, demonstrando que os modelos de ML treinados com estatísticas de rede são capazes de distinguir os diferentes níveis de congestionamento veicular. Por outro lado, LR e SVM apresentaram acurácia inferior a 73%, indicando limitações dessas abordagens para o problema de classificação multiclasse com as características empregadas.

4.5 Considerações finais

Os resultados obtidos neste capítulo demonstram que é possível analisar dados derivados da comunicação de rede de aplicações veiculares para detectar eventos no contexto de redes veiculares por meio de ML. Dados de aplicações utilizando infraestrutura MEC em um cenário de congestionamento com 158 veículos foram gerados para treinar e avaliar os modelos. Os classificadores RF, DT e KNN apresentaram desempenho consistente, com acurácia superior a 85%, indicando que esses modelos são adequados para detectar a dinâmica no contexto veicular a partir de estatísticas de rede. Os modelos LR e SVM, por outro lado, apresentaram acurácia em torno de 70–72%, o que limita sua aplicabilidade neste cenário.

A abordagem proposta tem como vantagem utilizar dados que podem ser obtidos diretamente no núcleo da rede, sem a necessidade de sensores adicionais ou fontes externas, fornecendo informações que subsidiam o atendimento dinâmico dos requisitos de aplicações ITS. Como direção de evolução, a integração desta abordagem a uma rede definida por software para adaptação dinâmica dos parâmetros de comunicação constitui um passo natural. Adicionalmente, a aplicação de Aprendizado Federado (*Federated Learning*) a cenários com MEC [95] — direção que, desde a publicação original deste trabalho, tem sido objeto de implementações concretas para classificação de tráfego de rede veicular e predição de contexto em ambientes distribuídos [96, 97] — é explorada nos Capítulos 7 e 8 desta tese.

No entanto, embora a abordagem permita a identificação do contexto veicular — e possa ser utilizada para reorganizar políticas de comunicação na rede —, ela não endereça o gerenciamento de fatias de rede (*network slicing*) como um todo. A capacidade de identificar o contexto veicular é um componente necessário, mas não suficiente, para um sistema completo de fatiamento inteligente. Dessa forma, o capítulo seguinte avança nessa direção apresentando um framework para fatiamento de rede que permite a reorganização dinâmica de fatias com base nas aplicações suportadas pela infraestrutura ITS.

5 FRAMEWORK DE FATIAMENTO DE REDE ORIENTADO A APLICAÇÕES ITS

Os Capítulos 3 e 4 demonstraram que técnicas de aprendizado de máquina são capazes de identificar o contexto veicular a partir de fontes de dados distintas: dados públicos de sistemas de transporte e estatísticas de comunicação da própria rede, respectivamente. Entretanto, a capacidade de identificar o contexto não garante, por si só, o atendimento aos requisitos das aplicações ITS em uma infraestrutura de rede compartilhada. Para que a informação de contexto se traduza em ações concretas sobre os recursos, é necessário um mecanismo de orquestração que considere os requisitos heterogêneos das aplicações e reconfigure a rede de forma dinâmica.

Este capítulo apresenta um framework de fatiamento de rede 5G orientado a aplicações ITS, publicado em [11][1]. O framework proposto integra conceitos de fatiamento de rede (*network slicing*) e redes definidas por software (SDN, do inglês *Software Defined Networking*) com algoritmos que consideram os requisitos das aplicações, a infraestrutura disponível e a mobilidade dos veículos para alocar e reconfigurar recursos dinamicamente. Seguindo os princípios do fatiamento de rede 5G, foram desenvolvidos métodos para integração de controladores SDN que utilizam os requisitos de comunicação das aplicações e informações de mobilidade dos veículos em cada rede de acesso via rádio (RAN, do inglês *Radio Access Network*) para atender às necessidades das aplicações ITS de forma dinâmica. A avaliação experimental, conduzida em cenário veicular realístico com emulação, demonstrou melhorias significativas em relação a abordagens do estado da arte.

As principais contribuições apresentadas neste capítulo são: (i) um algoritmo para controladores SDN que utiliza informações dos requisitos de diferentes aplicações ITS — como prioridade, largura de banda, latência e perda de pacotes — para aplicar políticas globais de priorização dos recursos de comunicação da infraestrutura, incluindo integração com uma entidade de Gerenciamento e Orquestração (MANO, do inglês *Management and Orchestration*); (ii) um algoritmo que utiliza informações sobre mudanças de topologia no contexto veicular, obtidas pelos controladores locais em cada RAN, para gerenciar os fluxos de comunicação localmente; e (iii) um protótipo de prova de conceito com controladores SDN de código aberto, *switches* e funcionalidades OpenFlow em ambiente emulado, utilizado para avaliar o framework proposto em comparação com uma abordagem baseada apenas em QoS e outra abordagem da literatura [98].

[1] Originalmente publicado com o título: “An Application-Driven Framework for Intelligent Transportation Systems Using 5G Network Slicing”, no periódico IEEE Transactions on Intelligent Transportation Systems (Qualis A1).

O restante deste capítulo está organizado da seguinte forma. A Seção 5.1 apresenta os trabalhos relacionados. A Seção 5.2 descreve o framework proposto. A Seção 5.3 detalha a configuração da avaliação experimental. A Seção 5.4 apresenta os resultados. A Seção 5.4.4 discute os resultados obtidos. Por fim, a Seção 5.5 apresenta as considerações finais do capítulo.

5.1 Trabalhos relacionados

A ideia de direcionar o comportamento da rede para atender às demandas das aplicações, com utilização eficiente de recursos, não é recente e já havia sido considerada na literatura de provisionamento de QoS [99]. Entretanto, as soluções existentes possuem limitações para adoção em ambientes modernos, principalmente em cenários com redes veiculares para ITS. A proposta em [100] utiliza o conceito de redes orientadas a aplicações (ADN, do inglês *Application-Driven Networks*), que são redes capazes de fornecer caminhos de dados para atender aos requisitos de comunicação das aplicações de forma dinâmica. Embora o trabalho apresente uma arquitetura com algoritmos de controle e uma prova de conceito relacionada ao treinamento de motoristas, a solução não considera os componentes específicos das redes veiculares, como mobilidade e RSUs.

Alguns trabalhos, como [101], [38] e [102], propõem soluções para tratar a QoS das aplicações considerando os requisitos de comunicação de diferentes aplicações no ambiente dinâmico das redes veiculares, utilizando tecnologia SDN. Por outro lado, esses trabalhos não utilizam fatiamento de rede 5G e não especificam os algoritmos necessários para o funcionamento correto dos componentes das soluções propostas. Além disso, carecem de um experimento de prova de conceito realístico que considere tanto os componentes veiculares quanto os do núcleo da rede.

Em [98], os autores analisaram o potencial das redes veiculares definidas por software, enfatizando a necessidade de repensar a abordagem tradicional de SDN, tanto do ponto de vista teórico quanto prático, para gerenciar os recursos de comunicação e de rede no ambiente veicular. De forma semelhante, [103] considera tanto o fatiamento de rede 5G quanto a dinâmica do ambiente veicular, utilizando um experimento de prova de conceito realístico com SDN no núcleo da rede. Um framework que também utiliza SDN para ajustar as condições de rede visando atender dinamicamente aos requisitos das aplicações ITS é proposto em [104]. O problema com esses trabalhos é que não especificam os algoritmos dos componentes da solução proposta e não apresentam uma forma clara de atender dinamicamente aos requisitos de comunicação de diferentes aplicações ITS.

Em [105], os autores propõem uma solução com computação em névoa (*fog computing*) e simulam um experimento de prova de conceito para produzir resultados baseados em condições de tráfego geradas pelo SUMO [89] e OpenStreetMap [106], visando dar

suporte a aplicações veiculares. Entretanto, a solução proposta não trata do atendimento a requisitos específicos das aplicações veiculares. Os autores em [107] também utilizam computação em névoa, mas consideram as necessidades das aplicações. De forma semelhante, [108] propõe uma plataforma para fornecer serviços de nuvem veicular. Todavia, esses trabalhos não fornecem os algoritmos da solução nem um experimento de prova de conceito realístico.

O fatiamento de rede 5G em um núcleo de rede móvel SDN foi utilizado em [109] para atender aos requisitos do caso de uso de aplicações de veículos autônomos. Os autores consideram a dinâmica do ambiente veicular e apresentam um algoritmo de gerenciamento para fatiamento de recursos de direção autônoma. O experimento de prova de conceito utiliza componentes realísticos de SDN e mobilidade. Como o foco é em veículos autônomos, a proposta não considera o atendimento dinâmico aos requisitos de comunicação de diferentes aplicações.

A proposta em [110] utiliza a tecnologia de nuvem de serviços veiculares (VSC, do inglês *Vehicular Service Cloud*), formando VSCs de baixa latência dinamicamente conforme as necessidades dos usuários veiculares. Da mesma forma que o aspecto "não-rádio" da comunicação 5G (por exemplo, SDN e NFV) se tornou mais importante, em 6G os aspectos de aprendizado e inteligência serão fundamentais e devem ser integrados às redes de comunicação [20]. Dessa forma, soluções emergentes incorporam abordagens de Inteligência Artificial (IA) para lidar com a complexidade inerente do ambiente ITS [20]. No contexto da evolução para 6G, o framework IMT-2030 da ITU-R [111] formalizou capacidades inéditas como sensoriamento integrado à comunicação (ISAC, do inglês *Integrated Sensing and Communication*), inteligência artificial nativa e sustentabilidade. Conforme apontado em [112], ISAC desponta como capacidade definidora das redes veiculares 6G, permitindo que a infraestrutura de comunicação também realize funções de percepção do ambiente viário. Essas evoluções reforçam a necessidade de frameworks de fatiamento flexíveis e orientados a aplicações, como o proposto neste capítulo, capazes de acomodar novos tipos de fatias com requisitos conjuntos de comunicação e sensoriamento.

Com base nos estudos revisados, propõe-se que uma solução completa que satisfaça os requisitos das aplicações ITS deve considerar os seguintes itens: (i) cenários com diferentes aplicações e requisitos, uma vez que existe uma variedade de aplicações projetadas com diferentes necessidades de QoS para diversos casos de uso — como sistema cooperativo de prevenção de colisões, realidade aumentada, navegação inteligente baseada em condições de tráfego em tempo real, direção autônoma, pelotão (*platooning*) e vídeo ao vivo em 4K [38, 69]; (ii) dinâmica do ambiente veicular, dado que a entrega de conteúdo em redes veiculares é desafiadora devido à movimentação dos nós [113], sendo necessário configurar a rede dinamicamente para manter o atendimento adequado aos requisitos das aplicações; (iii) algoritmos, pois o projeto de algoritmos de alocação dinâmica de recursos é

essencial para aumentar a eficiência de futuras redes fatiadas, sendo necessários algoritmos inteligentes capazes de prever demandas de serviço móvel e antecipar a reconfiguração de recursos [114]; (iv) fatiamento de rede 5G e SDN, visto que a tecnologia 5G emergente é considerada a solução promissora para melhorar o desempenho e o gerenciamento da rede, garantindo alta taxa de dados e QoS aprimorada [109]; e (v) experimento de prova de conceito realístico, dado que questões críticas emergem ao tentar colocar redes veiculares definidas por software em prática, incluindo pilhas de protocolo OpenFlow completas e avaliação experimental realística considerando código implantável, condições de rede, mobilidade e reprodutibilidade [98].

A partir da revisão da literatura, diversos problemas em aberto demandam investigação adicional. A Tabela 4 apresenta uma comparação da proposta deste capítulo com os trabalhos relacionados, considerando os cinco critérios identificados. Ao conhecimento dos autores, este trabalho é o primeiro a considerar simultaneamente os requisitos das aplicações e a dinâmica das redes veiculares, com validação adequada utilizando emulação SDVN realística, fornecendo uma comparação experimental com uma abordagem que utiliza QoS e outra da literatura que utiliza SDN sem QoS.

Cabe registrar que, após a publicação original deste trabalho em 2021, a literatura convergiu para abordagens alinhadas à arquitetura aqui proposta. Em [115], um sistema multi-agente de aprendizado por reforço para alocação de recursos em fatias veiculares adota classificação de serviços em segurança, eficiência e entretenimento — categorização equivalente às classes A, B e C do framework proposto — alcançando 96,5% de satisfação de QoS. De forma semelhante, [116] propõe um framework SDN+NFV da borda ao núcleo para fatiamento veicular com classificação idêntica de aplicações, incorporando aprendizado por reforço profundo para operação dinâmica do controlador de fatias. Levantamentos recentes dedicados ao fatiamento de rede para IoV [83] e ao gerenciamento de recursos de fatias do 5G ao 6G [117] confirmam que a combinação de controladores SDN hierárquicos com orquestração orientada a requisitos de aplicação permanece como paradigma central, embora soluções baseadas em IA estejam progressivamente substituindo políticas estáticas de alocação de recursos.

5.2 Framework proposto para redes veiculares orientadas a aplicações

As redes de comunicação veicular consistem tipicamente em veículos com unidades embarcadas (OBUs, do inglês *On-Board Units*) e unidades de beira de estrada (RSUs, do inglês *Roadside Units*). As RSUs fornecem comunicação entre os veículos e a infraestrutura da cidade por meio de estações base celulares. Neste capítulo, os termos RSU e estação base são utilizados com o mesmo significado. A Figura 17 apresenta os componentes da

Tabela 4 – Comparação da proposta com a literatura.

	[105]	[108]	[101]	[38]	[102]	[104]	[100]	[98]	[103]	[107]	[109]	[110]	[20]	Esta proposta
Experimento de prova de conceito realístico						✓		✓	✓		✓	✓		✓
Diferentes aplicações e requisitos		✓	✓	✓	✓		✓			✓		✓	✓	✓
Dinâmica do ambiente veicular	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓	✓	✓
Fatiamento de rede 5G								✓	✓	✓	✓			✓
Algoritmos							✓				✓			✓
Redes definidas por software			✓	✓	✓	✓	✓	✓	✓	✓	✓		✓	✓

arquitetura considerada e como estão interconectados. Os elementos dessa arquitetura trabalham em conjunto com tecnologias-chave do 5G, em particular o fatiamento de rede.

O fatiamento de rede e o SDN são tecnologias que trouxeram melhorias significativas tanto para as RANs quanto para as redes centrais das comunicações móveis. Com o conceito de fatiamento de rede, o 5G visa atender a requisitos diversificados de serviço utilizando as tecnologias existentes [15]. Embora a tecnologia necessária para suportar os diferentes tipos de fatias seja bem compreendida, as implicações do fatiamento de rede em termos de eficiência na utilização dos recursos de rede ainda não são completamente entendidas [114]. Apesar da dificuldade em determinar quando os procedimentos para atender aos requisitos específicos de comunicação veicular serão padronizados no 5G, a falta de alternativas tem motivado o interesse na utilização do 5G para comunicações veiculares [118].

Conforme definido pela Open Networking Foundation [119], uma arquitetura SDN é composta pelo plano de aplicação, plano de controle e plano de dados. Os dispositivos do plano de dados, que tratam da comutação de pacotes e do fluxo de tráfego, são separados dos dispositivos do plano de controle, que definem como o plano de dados deve operar. O plano de aplicação é populado por instâncias de aplicação que representam entidades clientes capazes de solicitar algum serviço a um servidor controlador SDN [38], tornando o gerenciamento da rede mais flexível, uma vez que é possível reconfigurar a rede dinamicamente para atender aos requisitos das aplicações. Seguindo uma organização semelhante ao SDN, a arquitetura do framework proposto é detalhada nas subseções a seguir.

5.2.1 Plano de aplicação

Em uma SDN existem dois tipos de aplicações: as aplicações finais, suportadas pela infraestrutura, e as aplicações de controle, que definem o comportamento da rede. Esses dois tipos de aplicações, referidos respectivamente como aplicações “veiculares” e

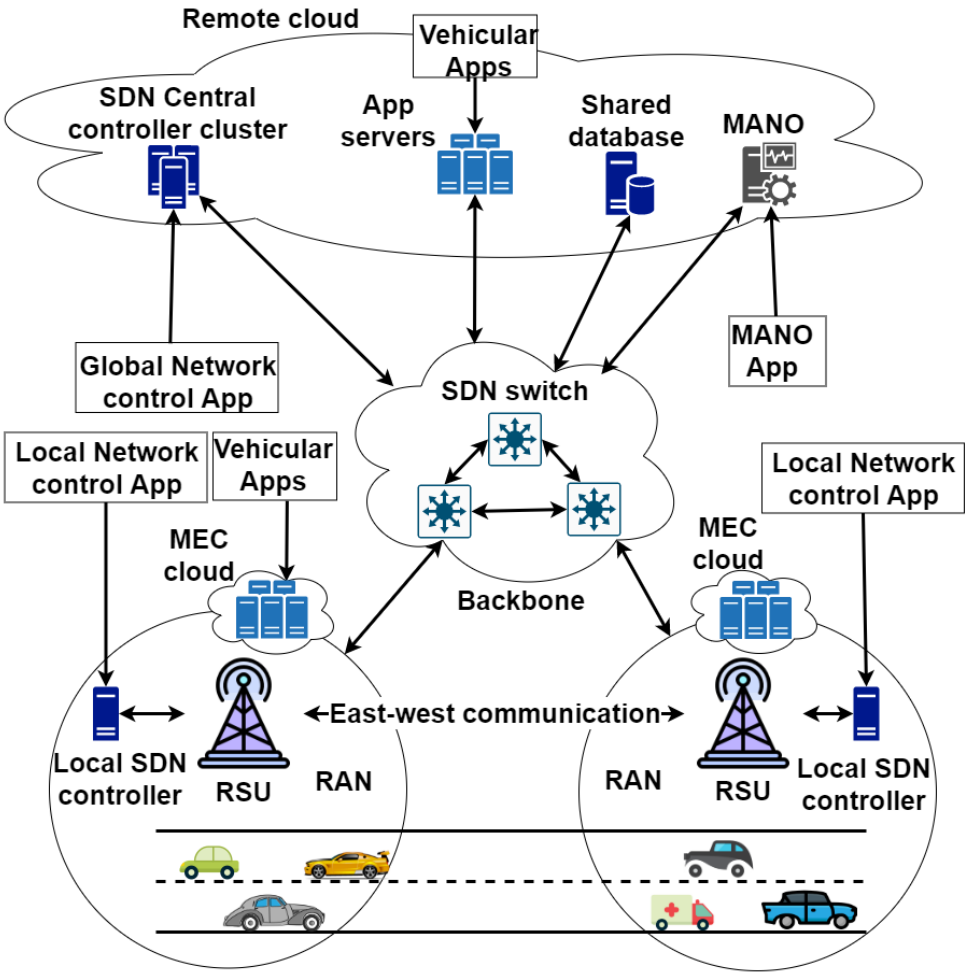


Figura 17 – Arquitetura do framework proposto com fatiamento de rede para atender requisitos de aplicações ITS. Fonte: o autor, publicado em [11].

aplicações de "controle", constituem o plano de aplicação na solução proposta.

Em alguns trabalhos relacionados, as aplicações veiculares interagem diretamente com os controladores de rede para ter suas demandas de comunicação atendidas. Essa abordagem não é a mais adequada em uma rede grande e complexa, suportada por múltiplos operadores e possivelmente com diferentes controladores e suas APIs. Na abordagem proposta, os dispositivos controladores e as aplicações veiculares compartilham as informações relevantes por meio de um banco de dados compartilhado, utilizando uma linguagem de manipulação de dados (DML) padrão como SQL (*Structured Query Language*). Dessa forma, o nível de controle torna-se transparente para as aplicações veiculares, que podem ser indiferentes às funções de controle e focadas em fornecer seus serviços.

Dado esse modelo, cada aplicação veicular em uma cidade inteligente possui seus algoritmos separados para cumprir seu papel, e outro conjunto de algoritmos comuns para atualizar informações relevantes a todas as aplicações — como seus assinantes de serviço e requisitos de aplicação — no banco de dados compartilhado. No framework proposto, os

Algorithm 1 Algoritmo Genérico de Aplicação

Entrada: *vehicle_subscriptions*, *vehicle_unsubscriptions*, KPI
Saída: Atualização do banco de dados, notificações ao MANO
1: Início
2: Atualizar no banco de dados compartilhado as novas *vehicle_subscriptions*, *vehicle_unsubscriptions* e alterações de KPI
3: Definir $MF \leftarrow 1$
4: Notificar o MANO sobre as alterações, enviando o flag de modificação (MF) e o identificador da aplicação (*app_id*)
5: Fim

assinantes de serviço são os veículos que utilizam as aplicações veiculares do framework.

Conforme ilustrado no Algoritmo 1, o algoritmo genérico de aplicação recebe novas subscrições de veículos, cancelamentos de subscrição ou atualizações de indicadores-chave de desempenho (KPIs, do inglês *Key Performance Indicators*). Em seguida, o estado relevante é atualizado no banco de dados compartilhado e o MANO é notificado sobre o identificador da aplicação (*app_id*) e as alterações no flag de modificação (MF).

As aplicações de controle são de três tipos: Global Network Control App, Local Network Control App e MANO App. Como são aplicações relacionadas ao plano de controle, seus algoritmos são detalhados na Seção 5.2.3.

5.2.2 Plano de dados

No framework proposto, o plano de dados consiste nos *switches* SDN que existem tanto no núcleo da rede quanto nos veículos. Esses *switches* recebem regras de comutação dos controladores do plano de controle para encaminhar pacotes e priorizar os dados das aplicações veiculares adequadamente.

5.2.3 Plano de controle

Uma vez que o objetivo do framework proposto é atender aos requisitos das aplicações em um ambiente dinâmico, é necessário definir quando a rede deixa um estado estável — ou seja, quando a infraestrutura não está atendendo aos requisitos das aplicações (por meio de KPIs). Também é fundamental determinar os componentes que devem atuar para restabelecer a estabilidade. Conforme ilustrado na Figura 18, a rede pode deixar o estado estável por uma ou mais de cinco razões, todas resultando em um grupo transitório de ações executadas pelos componentes de controle para levar a rede de volta ao estado estável.

Quando há a implantação de novas estações base na cidade, a implementação de uma nova aplicação veicular ou atualização de aplicações, é necessário que o MANO verifique a infraestrutura atual disponível para analisar se uma reotimização de recursos é necessária. O mesmo se aplica quando há modificações de topologia no nível de regiões, como veículos se movendo para outra cidade. Um cenário geralmente mais frequente é o de

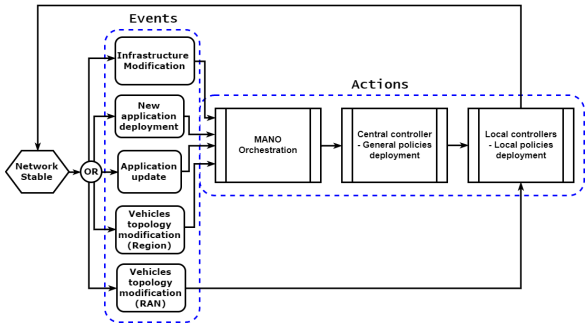


Figura 18 – Diagrama de fluxo do framework em função de mudanças na topologia, aplicações ou recursos de infraestrutura. Fonte: o autor, publicado em [11].

Algorithm 2 Algoritmo de Gerenciamento e Orquestração (MANO)

Entrada: app_id, AIR, ICI, VSI, KPI, MF
Saída: Notificações ao controlador central, orquestração de infraestrutura, consultas e atualizações no banco de dados

Início
if $MF = 0$ **then**
    Consultar o banco de dados compartilhado e atualizar a variável DS
    Computar os recursos para a nova aplicação, usando AIR, DS, ICI, VSI, KPI, e armazenar o resultado em RTI
    Implantar a infraestrutura definida em RTI
    Atualizar no banco de dados as informações da infraestrutura associada à nova aplicação
    Notificar o controlador central, enviando MF e o identificador da nova aplicação (app_id)
    Atualizar a variável DS com informações atualizadas do banco de dados
else
    Consultar o banco de dados compartilhado e atualizar a variável NDS
    Computar as mudanças, com base em NDS e DS, e armazenar o resultado em DC
    Computar os recursos para o novo cenário, com base em DC, e armazenar o resultado em RTI
    Implantar a infraestrutura definida em RTI
    Atualizar no banco de dados as informações da infraestrutura associada à aplicação
    Atualizar a variável DS com informações atualizadas do banco de dados
    Definir $MF = 1$
    Notificar o controlador central enviando MF e app_id
end if
Fim

veículos se movendo entre as áreas de cobertura de diferentes estações base, modificando a topologia da rede no nível de RANs e, em alguns casos, resultando em alta utilização de recursos. Os algoritmos de controle da rede devem tomar as ações necessárias em todos esses cenários para que a rede retorne à estabilidade. As subseções a seguir detalham os algoritmos propostos para esse fim.

5.2.3.1 MANO (*Management and Orchestration*)

O MANO provê gerenciamento e orquestração de recursos físicos e virtuais tanto em nuvens MEC (*Multi-access Edge Computing*) quanto em nuvens remotas. Como resultado, instancia ou migra dinamicamente recursos de infraestrutura, incluindo servidores de aplicação e funções de rede virtualizadas (VNFs, do inglês *Virtual Network Functions*), para cada aplicação veicular, conforme descrito no Algoritmo 2.

O algoritmo do MANO utiliza a variável de controle MF para diferenciar se os parâmetros recebidos como entrada estão relacionados a uma nova aplicação ($MF = 0$)

ou não ($MF \neq 0$). Ao implantar uma nova aplicação, os *stakeholders* da cidade que oferecem serviços precisam submeter ao MANO algumas informações fundamentais, além de $MF = 0$: (i) Recursos de Infraestrutura da Aplicação (AIR), que contém informações sobre demandas de recursos de rede (por exemplo, interfaces de rede, registros DNS, definição NTP) e servidores (por exemplo, contêiner, HTTP, FTP); (ii) Informações de Configuração de Infraestrutura (ICI), que contém informações como URL de imagem de contêiner e alocação de endereço IP; (iii) Informações de Subscrição de Veículos (VSI), composta pelos códigos de identificação de cada veículo, as aplicações associadas a eles e as regiões geográficas às quais pertencem; (iv) Código Identificador da Aplicação (app_id), único para cada aplicação; e (v) Indicadores-Chave de Desempenho (KPI) para os requisitos das aplicações, como latência fim-a-fim, confiabilidade e taxa de dados.

O MANO também precisa verificar o status atual dos elementos da infraestrutura sob seu domínio. Para isso, consulta o banco de dados compartilhado, armazena o resultado na variável DS (*Database Status*) e utiliza-a em conjunto com AIR, ICI, VSI e KPI para computar os recursos necessários para atender aos KPIs das aplicações. Soluções MEC (por exemplo, instanciar um contêiner Docker) podem ser utilizadas para atender aos requisitos de aplicações sensíveis a atraso, se o KPI de latência for inferior a um limiar específico.

Após o cálculo dos recursos, o MANO procede à implantação das ações necessárias na infraestrutura. Em seguida, atualiza as informações relacionadas à nova aplicação, KPIs, veículos e infraestrutura no banco de dados compartilhado. O MANO então notifica o controlador central enviando o valor de MF e o identificador da aplicação implantada (app_id), para que o controlador prossiga conforme descrito na Seção 5.2.3.2 para implementar as políticas de rede globais necessárias. Uma vez que a implantação é concluída, o MANO consulta os dados mais atualizados em todas as tabelas do banco de dados compartilhado e os armazena localmente na variável DS para uso em casos futuros onde $MF \neq 0$.

Se o MANO receber $MF \neq 0$, trata-se de uma atualização que pode ser disparada por uma aplicação que (1) atualiza suas informações no banco de dados compartilhado ou (2) detecta modificações no nível de infraestrutura. Nesses casos, o MANO define uma variável NDS (*New Database Status*) para armazenar os novos valores de dados, e outra variável DC para salvar as mudanças. Com os dados anteriores e as diferenças, o MANO computa os recursos para o novo cenário e os implanta. Uma vez implantados ou otimizados, o MANO atualiza essas informações no banco de dados compartilhado, atualiza a variável DS com os dados mais recentes e notifica o controlador central com $MF = 1$ e o app_id da aplicação relacionada às atualizações.

Em resumo, o processo de gerenciamento e orquestração conduzido pelo MANO consiste na instanciação de recursos virtuais e no posicionamento de VNFs considerando

os requisitos das aplicações (KPIs). Esse processo pode ser realizado de diversas formas, conforme definido pelo MANO Framework [120]. Em [121] são apresentadas algumas tecnologias atualmente utilizadas. Uma solução robusta que pode ser utilizada para orquestrar infraestruturas virtuais heterogêneas (com componentes de rede e computação) é o OpenBaton [122], que fornece um motor que pode ser utilizado para garantir QoS para fatias de rede e gerenciar computação de borda usando ferramentas de implantação de servidores em contêiner.

O framework proposto integra esse processo de alocação de recursos executado pelo MANO com os controladores de rede central e locais, responsáveis por definir as ações a serem executadas pelos componentes SDN ao lidar com os fluxos de comunicação das aplicações ITS. O ganho dessa integração é que os controladores podem configurar os componentes de rede considerando a infraestrutura implantada pelo MANO, os requisitos das aplicações e a mobilidade dos veículos. Cabe observar que, desde a publicação original deste trabalho, o paradigma de orquestração tem evoluído em direção a abordagens nativas em nuvem (*cloud-native*) baseadas em Kubernetes e à arquitetura O-RAN (*Open Radio Access Network*), que introduz controladores inteligentes de RAN (RIC, do inglês *RAN Intelligent Controller*) para adaptação quase em tempo real das configurações de fatiamento [123]. Essas abordagens são complementares à arquitetura proposta, na medida em que preservam a separação entre orquestração de recursos e controle de políticas de rede — princípio central do framework aqui descrito — enquanto oferecem maior granularidade e automação na instanciação e migração de fatias.

5.2.3.2 Controlador de rede central

O controlador de rede central é a entidade que determina as políticas gerais de rede para o plano de dados. Seu algoritmo (Algoritmo 3) recebe como entrada MF e app_id. As saídas do algoritmo são consultas ao banco de dados, notificações aos controladores locais e a implantação de políticas gerais de rede, que resultam em regras para configurações como redirecionamento de fluxos de comunicação e QoS. A última saída é a notificação aos controladores locais em cada RAN, enviando as políticas globais configuradas na rede.

Quando $MF = 0$, ou seja, o MANO finalizou a implantação de uma nova aplicação, o controlador central precisa otimizar os recursos de rede aplicando as políticas necessárias para atender aos requisitos das aplicações. O controlador central consulta o banco de dados compartilhado, armazena o resultado em DS e o utiliza para criar as listas de variáveis: (1) KPIs das aplicações (ARL), (2) veículos associados a cada aplicação (AVL), (3) endereços IP ou URL e a região associada aos controladores locais em cada RAN (BSI), (4) infraestrutura implantada pelo MANO por aplicação (IAL).

Para cada aplicação em ARL, o controlador central define uma nova fatia de rede considerando todos os dados de AVL, BSI e IAL. Dessa forma, cada fatia inclui os

Algorithm 3 Algoritmo de Políticas Gerais

Entrada: MF, app_id
Saída: Consultas ao banco de dados, implantação de políticas gerais de rede, notificação aos controladores locais

Início
if $MF = 0$ **then**
    Consultar o banco de dados compartilhado e atualizar a variável DS
    Criar as listas ARL, AVL, BSI, IAL, com base na variável DS
    for each aplicação **em** ARL **do**
        Criar uma fatia para a aplicação e armazenar em $SDL(app_id)$, com base em app_id, AVL, IAL e BSI
        Computar as políticas globais de rede para atender aos requisitos da aplicação e armazenar em GP, com base em $SDL(app_id)$ e ARL
        Configurar as políticas globais nos elementos da fatia da aplicação, com base em GP e $SDL(app_id)$
        Notificar os controladores locais em cada RAN, enviando GP
    end for
else
    Consultar o banco de dados compartilhado e atualizar a variável NDS
    Computar as mudanças e armazenar em DC, com base em NDS e DS
    Atualizar ARL, AVL, BSI e IAL, com base em DC
    for each aplicação **em** ARL **do**
        Atualizar $SDL(app_id)$ com base em ARL, AVL, BSI e IAL
        Computar as políticas globais para atender aos requisitos da aplicação e armazenar em GP, com base em $SDL(app_id)$ e ARL
        Configurar as políticas globais nos elementos da fatia da aplicação, com base em GP e $SDL(app_id)$
        Notificar os controladores locais em cada RAN, enviando GP
    end for
end if
Fim

elementos — como veículos, servidores e RSUs — relacionados a cada aplicação. O mesmo dispositivo pode estar associado a mais de uma fatia (por exemplo, um veículo assinante de mais de uma aplicação). Após a criação da fatia, o controlador central computa as políticas gerais para cada fatia, de modo a atender aos KPIs das aplicações, e as aplica à rede (por exemplo, utilizando o protocolo OpenFlow). Exemplos de políticas de rede são regras de redirecionamento e filas de QoS definidas pelos algoritmos de aplicação nos controladores. As ações resultantes podem ser implementadas com o protocolo OpenFlow para definir as entradas de fluxo entre veículos e servidores de aplicação visando atender aos requisitos das aplicações ITS. Na implementação experimental, foram utilizadas as informações dos servidores de aplicação (endereço IP e conexões de rede), a informação de qual veículo estava associado a qual aplicação e os requisitos das aplicações (largura de banda e prioridade) para definir e configurar, utilizando o controlador SDN Ryu com OpenFlow, as filas de QoS para atender aos requisitos das aplicações.

Se o controlador central receber $MF \neq 0$, ele precisa consultar o banco de dados para identificar as mudanças utilizando a função $CC(NDS, DS)$. Com essas mudanças, atualiza as listas ARL, AVL, BSI e IAL e as fatias, computa as novas políticas necessárias e notifica os controladores locais.

5.2.3.3 Controlador de rede local

Os controladores locais atuam para atender dinamicamente aos requisitos das aplicações em cada RAN, seguindo as políticas globais do controlador central e considerando

Algorithm 4 Algoritmo dos Controladores Locais

Entrada: GP, $NAKPI$, TKC, IV
Saída: Configurações de políticas de controle de rede local, notificações de balanço disponível aos vizinhos, consultas e atualizações no banco de dados

Início
$TKNA = 0$
Identificar todos os veículos na RAN e salvar as informações em RV
Consultar o banco de dados compartilhado e atualizar AV
for each mensagem $NAKPI$ recebida de vizinhos **do**
    Armazenar no arquivo local $NAKPI.file$ os balanços de recursos recebidos da RSU vizinha
end for
for each veículo Vi em RV **do**
    Consultar no banco de dados a soma total de KPIs das aplicações das quais Vi é assinante e armazenar o resultado em $D(Vi)$ e adicionar a $TKNA$
end for
Calcular o balanço local da RSU ($TKC - TKNA + AV$) e armazenar o resultado em $AKPI$
if $AKPI < 0$ **then**
    Consultar informações de prioridade das aplicações no banco de dados e armazenar em AI
    Definir o nível máximo de classe de prioridade da API e armazenar em $MAPI$, com base em AI
    Definir $API = 1$ para iniciar com as aplicações de menor prioridade
    while $API < MAPI$ **do**
        Definir $BS_Id = NULL$
        Verificar em $NAKPI.file$ se há alguma vizinha capaz de atender o KPI da aplicação de classe API e armazenar o resultado em BS_Id
        if $BS_id \neq null$ **then**
            Computar as políticas para redirecionar fluxos da aplicação de classe API para a vizinha com ID igual a BS_Id e armazenar em LPR
            Configurar LPR na rede
            Atualizar o banco de dados compartilhado, com base em LPR
            Definir $API = API + 1$
        else
            Computar as políticas para limitar o tráfego da aplicação de classe API conforme GP e salvar em LQP
            Configurar LQP na rede
            Atualizar o banco de dados compartilhado, com base em LQP
        end if
    end while
else
    Se houver recursos excedentes, armazenar o valor em $AKPI$ e enviá-lo aos vizinhos
end if
Aguardar IV unidades de tempo para uma nova verificação
Retornar ao Início

os requisitos das aplicações nos veículos que pertencem à sua própria RAN. O algoritmo que executa em cada controlador local (Algoritmo 4) recebe como entrada as políticas globais implementadas pelo controlador central (GP), a informação sobre KPIs que os controladores vizinhos podem atender ($NAKPI$), o KPI total máximo que a RSU atual do controlador pode atender por padrão (TKC) e o intervalo de verificação (IV) para os controladores locais identificarem os veículos em suas RANs. O valor de IV pode ser determinado utilizando heurísticas que consideram diversas variáveis do sistema de transporte da cidade (por exemplo, feriados, acidentes), inclusive utilizando abordagens de IA. A variável TKC pode ser definida quando a RSU é implantada e atualizada posteriormente, se necessário.

Cada controlador local utiliza a variável $TKNA$ ("*Total KPI Necessary*") para armazenar os requisitos de recursos em sua RAN em um dado intervalo de tempo, enquanto AV ("*Availability*") é utilizada para armazenar os recursos disponíveis na RAN.

Inicialmente, o controlador limpa a variável $TKNA$, identifica os veículos em

sua RAN e salva o resultado em RV ("*RAN Vehicles*"). Para cada veículo na RAN, o controlador consulta o banco de dados compartilhado e calcula a soma total dos KPIs das aplicações das quais o veículo é assinante, armazenando o resultado em $D(Vi)$. A soma dos requisitos de todos os veículos na RAN é salva em $TKNA$. O controlador também consulta o banco de dados para atualizar AV, que representa o balanço de recursos redirecionados de ou para a RSU local por suas vizinhas. Para obter seu balanço total de recursos, o controlador da RSU calcula $AKPI = TKC - TKNA + AV$.

Quando $AKPI > 0$, o controlador local envia $AKPI$ para informar suas vizinhas, via uma API de comunicação *east-west*, que possui ociosidade de recursos que poderia ser utilizada (por exemplo, largura de banda disponível nos enlaces que a conectam ao *backbone* da rede), e aguarda IV segundos para rechecar o status da RAN. Cada mensagem com $AKPI$ que o controlador recebe armazena a informação reportada localmente em um arquivo ($NAKPI.file$).

Se $AKPI < 0$, significa que a RSU local não pode cumprir os KPIs e, dessa forma, utiliza as informações de prioridade das aplicações no banco de dados para criar índices de priorização (API e $MPAI$) e manipular os fluxos adequadamente. Em seguida, utiliza o arquivo $NAKPI.file$ para verificar se as RSUs vizinhas podem lidar com parte da demanda. Se o resultado for positivo, o identificador da unidade vizinha que pode atender é salvo em BS_id. O controlador local configura as políticas para redirecionar o tráfego das aplicações (LPR) e atualiza a informação de redirecionamento no banco de dados compartilhado. Se nenhuma unidade vizinha puder atender à demanda adicional, o controlador local implanta as políticas para limitar o tráfego da aplicação (LQP) com base na QoS determinada pelas Políticas Globais. Nesse caso, o banco de dados é atualizado com a informação de limitação de tráfego, pois será necessário calcular corretamente o balanço de recursos.

5.3 Avaliação experimental

A realização prática[2] e a avaliação de desempenho do framework proposto foram baseadas no emulador de redes sem fio Mininet-WiFi [86], um *fork* do conhecido emulador Mininet [87]. A implementação considera quatro classes de prioridade de aplicação: classes A, B, C e G. A classe A é reservada a aplicações que possuem requisitos de latência ultra-baixa e devem utilizar uma solução MEC, enquanto as classes B e C são mais tolerantes a atraso. No caso de recursos limitados na comunicação com servidores remotos, a aplicação de classe B tem mais prioridade que a C. A classe G representa aplicações comuns que não possuem prioridade nem requisitos específicos.

[2] Código-fonte, dados, informações de configuração e documentação de suporte que permitem a reprodutibilidade do experimento estão disponíveis no repositório público do projeto: https://github.com/saraivacode/framework_its_sdn

O papel de controlador central na rede foi implementado utilizando o controlador SDN Ryu[3], um software amplamente utilizado que suporta as versões mais recentes do protocolo OpenFlow e possui uma comunidade ativa [124]. O controlador SDN utiliza as informações sobre veículos, RSUs e requisitos das aplicações armazenados no banco de dados compartilhado, desenvolvido com MySQL, para configurar políticas globais de QoS nos *switches* SDN da rede.

Periodicamente, o controlador local identifica os veículos na RAN de cada RSU, consulta o banco de dados compartilhado para determinar as aplicações associadas a esses veículos e obtém as informações sobre os requisitos dessas aplicações. Em seguida, esses requisitos são somados para obter os requisitos acumulados para cada veículo na RAN. Somando as demandas calculadas para cada veículo na RAN, o controlador local pode computar os requisitos totais que a RSU precisa atender em um dado instante.

A implementação considera a necessidade de priorizar adequadamente o KPI de taxa de dados da aplicação. O controlador local consulta o banco de dados para verificar informações sobre tráfego redirecionado pela RSU ou para ela. Se a RSU redirecionou ou limitou algum tráfego de dados no passado, o valor da taxa de dados relacionada precisa ser adicionado ao balanço final. Por outro lado, se ela recebe algum tráfego redirecionado, o valor deve ser subtraído de seu balanço.

Quando a RSU está congestionada, o controlador local dispara ações para priorizar as aplicações de classe B em detrimento das de classe C, que possuem menor prioridade. Os controladores locais verificam se sua lista local de veículos com aplicações de classe C não está vazia e se alguma unidade vizinha possui balanço para lidar adequadamente com a taxa de dados relacionada. Se ambas as verificações retornarem verdadeiro, os fluxos desses veículos são redirecionados (um por vez, conforme necessário) para a RSU vizinha. A informação sobre o redirecionamento é atualizada pelo controlador local no banco de dados compartilhado. Esses registros são eliminados quando os veículos não estão mais conectados à RAN, evitando distorções nos resultados de balanço.

Se a RSU está congestionada e não pode redirecionar fluxos de classe C, o controlador tenta redirecionar fluxos de aplicações de classe B. Se isso não for possível, limita os fluxos da aplicação de classe C localmente, associando-os a uma fila padrão de tráfego sem prioridade. Se essa fila já contiver tráfego de outras aplicações sem prioridade (classe G), o controlador local também bloqueia esse tráfego para não competir com o tráfego de classe C limitado.

No ambiente de avaliação, quando um veículo deixa a área de cobertura de uma RSU, o *timeout* de associação é de aproximadamente 20 segundos. Dessa forma, o controlador local precisa aguardar pelo menos 20 segundos para confirmar um possível congestionamento de

[3] https://osrg.github.io/ryu/

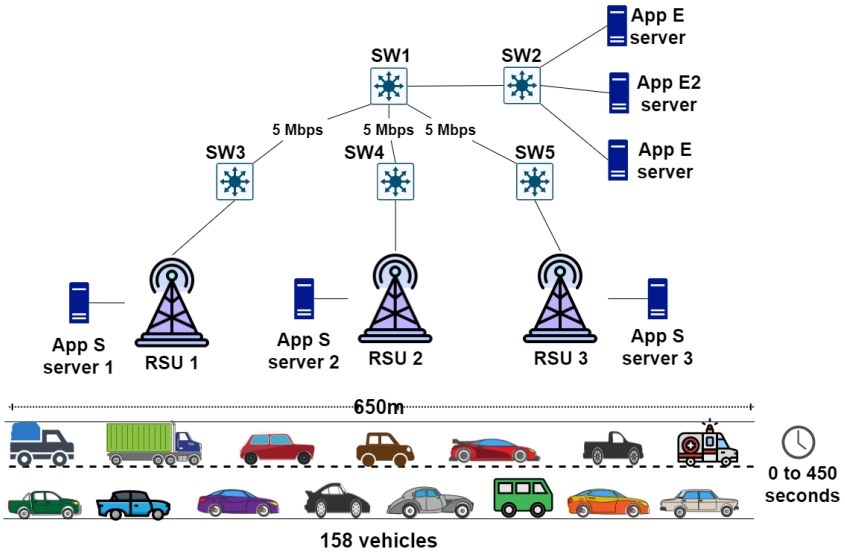


Figura 19 – Cenário de avaliação representando um congestionamento em uma via urbana densa. Fonte: o autor, publicado em [11].

tráfego antes de atuar para priorizar as aplicações. Conforme apresentado na análise dos resultados, esse tempo de espera possui impacto nos resultados obtidos.

5.3.1 Cenário de avaliação

Representando uma situação ordinária que ocorre no trânsito da maioria das cidades ao redor do mundo, o cenário de avaliação consiste em um congestionamento de tráfego, com densidade crescente ao longo do tempo (ou seja, veículos por m^2). Conforme ilustrado na Figura 19, durante os 450 segundos de tempo total de avaliação, 158 veículos se movem nos 650 m de uma via urbana em Manhattan (Nova York), conforme definido pelo simulador de mobilidade urbana SUMO [89], resultando em diferentes níveis de congestionamento ao longo do tempo. Essa configuração foi baseada em trabalhos como [38] e [88], que simularam o desempenho de suas propostas utilizando diferentes níveis de densidade veicular.

A mobilidade foi implementada utilizando a integração entre o emulador Mininet-WiFi e o simulador SUMO. O SUMO foi configurado com o mapa de vias de Manhattan para gerar a mobilidade de 158 veículos, quantidade suficiente para preencher o segmento de via na faixa de comunicação das RSUs. Foram configuradas três RSUs em um segmento de 650 m, com a área de cobertura de sinal correspondente a 250 m em cada RSU, com base em [90].

Para observar o impacto da interferência causada pelas transmissões dos veículos nas métricas avaliadas, dos 158 veículos, no máximo 50 geram tráfego relacionado a diferentes aplicações simultaneamente, sendo 15 veículos gerando dados de aplicações prioritárias. Com esse cenário, é possível avaliar como as diferentes abordagens lidam com mudanças

Tabela 5 – Resumo dos parâmetros utilizados no cenário de avaliação de desempenho.

Parâmetro	**Valor**
Número de veículos	158
Número de RSUs	3
Alcance das RSUs	250 m
Switches SDN de *backbone*	5
Servidores de aplicação	6
Modelo de propagação	Log Distance
Camada MAC da RAN	IEEE 802.11g
Taxa de dados por veículo com aplicações prioritárias	2 Mbps
Número de aplicações	4
Largura de banda do *uplink* das RSUs	5 Mbps
Tempo de emulação	450 segundos

Tabela 6 – Características das aplicações.

Aplicação	**Uso**	**KPI de taxa de dados**	**Protocolo**	**Porta**	**Classe de prioridade**
S	Segurança	500 Kbps	UDP	5002	A
E	Eficiência	500 Kbps	UDP	5003	B
E2	Entretenimento	1 Mbps	UDP	5004	C
G	Genérica	500 Kbps	UDP	5005	G

de topologia, resultando em diferentes níveis de consumo de recursos de rede em um ambiente densamente congestionado, impactando o KPI de taxa de dados das aplicações ITS prioritárias. Os parâmetros utilizados no cenário de avaliação de desempenho estão resumidos na Tabela 5.

Os papéis dos veículos foram divididos de forma que 15 veículos são assinantes de quatro aplicações veiculares hipotéticas, configuradas conforme descrito na Tabela 6. A aplicação denominada S representa uma aplicação de classe A voltada à segurança veicular. Esse tipo de aplicação é considerado um passo fundamental para aumentar a segurança viária, prevenindo a ocorrência de acidentes [125], sendo portanto a mais crítica quanto a atrasos e perdas. A aplicação S possui um KPI de taxa de dados de 500 Kbps, e seus servidores (S1, S2 e S3) estão conectados diretamente à RAN de cada RSU, seguindo uma abordagem MEC.

A aplicação E representa uma aplicação de classe B voltada à eficiência, com KPI de taxa de dados de 500 Kbps. Aplicações ITS de eficiência podem tratar de diversos aspectos do ambiente veicular, como a eficiência energética de veículos totalmente elétricos [126]. A aplicação $E2$ representa uma aplicação de classe C de entretenimento (por exemplo, uma aplicação específica de *streaming* de vídeo) e consome maior largura de banda, com KPI de taxa de dados de 1 Mbps. No cenário de avaliação, essas aplicações não são sensíveis a atraso, e seu tráfego segue para servidores remotos através do enlace de *backbone* das RSUs. A aplicação G representa tráfego genérico de 500 Kbps sem prioridade, como o tráfego de um navegador de internet comum.

Cada um dos 15 veículos assinantes de todas as aplicações gera uma taxa de dados total de 2 Mbps. Uma vez que os enlaces de *backbone* das RSUs são configurados com limite de 5 Mbps, esses enlaces ficam congestionados em alguns períodos durante o tempo de avaliação, sendo necessário priorizar as aplicações adequadamente. Os cinco *switches* SDN de *backbone* ($SW1$ a $SW5$) interconectam as RSUs com os servidores em nuvem das aplicações E, $E2$ e G. Há também um *switch* interno em cada RSU que interconecta as RSUs com suas unidades vizinhas, os respectivos servidores MEC e os *switches* de *backbone*.

O Mininet-WiFi não implementa redes sem fio 5G. Entretanto, o 5G não se resume à tecnologia de acesso via rádio e abrange diversas tecnologias em seu núcleo de rede [127]. Dessa forma, o emulador foi configurado com IEEE 802.11g para a conectividade de acesso via rádio, devido às suas capacidades de fornecer redes SDN sem fio realísticas, suficientes para implantar (em conjunto com outras ferramentas) cenários de avaliação com fatiamento de rede 5G e aplicações ITS. Essa mesma abordagem foi adotada em outros trabalhos com foco em fatiamento SDN para transporte, como [128], que utiliza Mininet-WiFi com SUMO e reconhece explicitamente a mesma limitação. Embora simuladores com suporte nativo ao 5G NR tenham amadurecido desde a realização dos experimentos — como o Simu5G [129], que integra a pilha de protocolos 5G NR com suporte a C-V2X e MEC sobre OMNeT++ — os resultados aqui apresentados se concentram na validação dos algoritmos de controle SDN no núcleo da rede (plano de controle e orquestração), cujo comportamento independe da tecnologia de acesso via rádio empregada na RAN. Fenômenos específicos do 5G NR, como efeitos de numerologia, processos HARQ e seleção de recursos baseada em sensoriamento, não são capturados pela emulação utilizada e constituem oportunidade para avaliações complementares.

5.3.2 Abordagens comparadas

Com base em [91], que considera como esquemas de avaliação a linha de base (sem QoS) e QoS estática, o framework proposto foi comparado a duas outras abordagens, denominadas “QoS only” e “RMSDVN”: (i) **“QoS only”**: a rede define políticas de QoS para atender aos KPIs de taxa de dados das aplicações, configurando regras e filas de QoS como o controlador SDN central da solução proposta, porém sem funcionalidades de controlador local, como redirecionamento de tráfego; portanto, essa abordagem não atualiza a QoS de acordo com mudanças no ambiente veicular; (ii) **“RMSDVN”**: a abordagem implementada em [98] utiliza SDN para gerenciar os recursos na rede veicular considerando a posição dos nós, porém não implementa priorização de tráfego.

5.3.3 Métricas de avaliação

Três métricas são empregadas para avaliar o impacto das abordagens acima. Como em [70], que utiliza o *throughput* para avaliar sua proposta em uma 5G-VANET habilitada por SDN, também é utilizado o *throughput* de dados entre veículos e servidores de aplicação. Como no cenário avaliado o KPI das aplicações priorizadas é a taxa de dados, o *throughput* foi coletado ao longo do tempo para verificar se os servidores de aplicação receberam a taxa de dados esperada de acordo com a transmissão dos veículos. As outras métricas são o tempo de ida e volta (RTT, do inglês *Round Trip Time*) e a taxa de entrega de pacotes (PDR, do inglês *Packet Delivery Ratio*). O RTT pode ser definido como o tempo decorrido desde o envio de uma mensagem pelo nó iniciador (veículo de origem) até o recebimento da resposta do núcleo da rede [130]. Conhecer o valor do RTT é importante porque ele é proporcional ao congestionamento [92], sendo também possível verificar o impacto nos fluxos de comunicação da aplicação S. Com o PDR é possível avaliar o nível de perda de dados em função de cada abordagem utilizada, uma vez que representa a razão média entre o número de pacotes de dados recebidos com sucesso no destino e o número de pacotes enviados. Essa métrica é amplamente utilizada para avaliar o desempenho de redes veiculares, incluindo cenários com múltiplas restrições de QoS [131].

A Equação 5.1 define o RTT médio por segundo, $RTT(s)$, a partir das medições individuais obtidas pelas mensagens ICMP *ping* registradas nos arquivos de log. Nessa equação, $RTTV_i$ é o i-ésimo valor de RTT medido no segundo s, dentre os n valores recebidos nesse intervalo, e NC é o número de veículos transmitindo no período de congestionamento correspondente.

$$RTT(s) = \frac{1}{NC} \sum_{i=1}^{n} RTTV_i \tag{5.1}$$

O PDR foi calculado pela razão entre a soma de pacotes recebidos pelos servidores e o número de pacotes enviados pelos veículos, conforme mostrado na Equação 5.2, onde PS_r é a soma dos pacotes recebidos pelos servidores e PS_t é a soma dos pacotes enviados pelos veículos.

$$PDR = \frac{PS_r}{PS_t} \tag{5.2}$$

Para computar o *throughput*, uma vez que os dados com o tamanho dos pacotes enviados pelos veículos e recebidos pelos servidores foram coletados em bytes, eles foram somados a cada segundo e multiplicados por 8 para obter o resultado em bits por segundo (bps). Os valores obtidos foram divididos pelo número de veículos que estavam efetivamente transmitindo a cada segundo, conforme mostrado nas Equações 5.3 e 5.4. $TTC(s)$ e $TRS(s)$ são, respectivamente, o *throughput* de pacotes enviados pelos veículos e recebidos pelos

servidores de aplicação no segundo s, enquanto PS_i e PR_i são, respectivamente, os n pacotes enviados pelos NC veículos e recebidos pelos servidores no segundo s. NC é crescente ao longo do tempo conforme mais veículos ingressam no congestionamento.

$$TTC(s) = \frac{1}{NC}\left(\sum_{i=0}^{n} PS_i\right) \times 8 \tag{5.3}$$

$$TRS(s) = \frac{1}{NC}\left(\sum_{i=0}^{n} PR_i\right) \times 8 \tag{5.4}$$

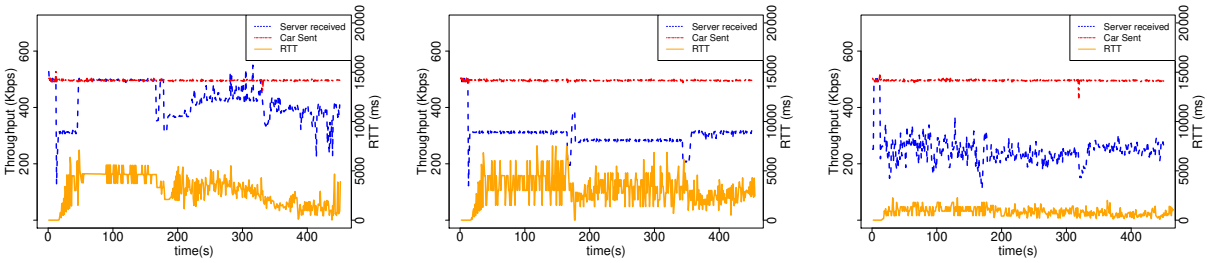
5.4 Resultados

5.4.1 Vazão e RTT ao longo do tempo

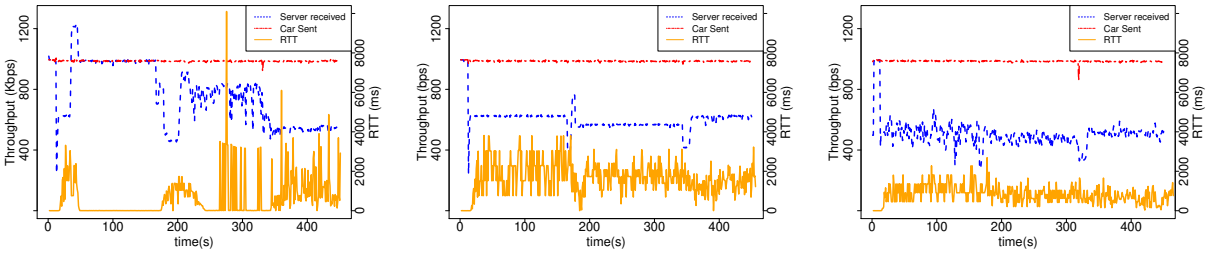
As Figuras 20(a), (b) e (c) ilustram os resultados de *throughput* e RTT relacionados à aplicação E, utilizando as abordagens "Framework", "QoS only" e "RMSDVN", respectivamente. Essa aplicação é a mais crítica entre as priorizadas nos enlaces de *backbone*.

A Figura 20 apresenta os resultados de *throughput* e RTT durante o tempo de avaliação para as aplicações E, $E2$, G e S, utilizando as diferentes abordagens avaliadas. Com todas as abordagens, o *throughput* que alcança o servidor da aplicação E é comprometido no início do tempo de avaliação: enquanto cada veículo gera um *throughput* de aproximadamente 500 Kbps para essa aplicação, no servidor chegam apenas cerca de 300 Kbps. Isso ocorre porque até 48 veículos alcançam a RSU3, sendo 17 gerando diversas taxas de dados, com 5 assinantes das aplicações E e $E2$ entre eles. Como cada um desses 5 veículos gera 2 Mbps de *throughput* para o *backbone* — sendo 1 Mbps para o servidor $E2$, 500 Kbps para o servidor E e 500 Kbps para o servidor G — totalizam-se 10 Mbps de requisito para o *uplink* de 5 Mbps da RSU. Há também 500 Kbps para a aplicação S no servidor MEC local, mas esses fluxos não utilizam o *uplink* da RAN e não são considerados pelo controlador. Com o framework proposto, o controlador local redirecionou 3 Mbps de fluxos de comunicação da aplicação $E2$ de três veículos, e o mecanismo de QoS limitou a aplicação G. Dessa forma, é possível observar que, após um tempo de convergência, a diferença entre o *throughput* gerado pelo veículo e o recebido pelos servidores é praticamente nulo até aproximadamente 200 s. Nesse momento, há 34 veículos gerando dados nas RSUs 1 e 2, com 138 veículos no congestionamento. A densidade de veículos aumenta de modo que, durante os últimos 100 s, ocorre o nível máximo de congestionamento nos experimentos, com 158 veículos totais, 50 gerando dados nas RANs, sendo 15 destes assinantes de aplicações prioritárias.

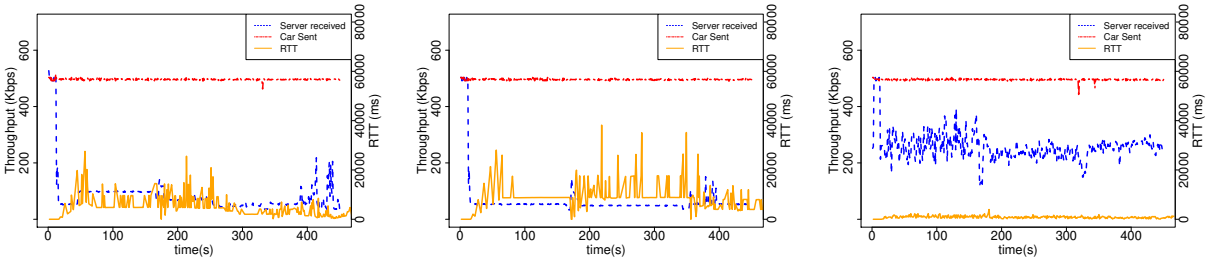
Em síntese, com o framework proposto foram obtidos os melhores resultados de taxa de dados ao longo do tempo para a aplicação E. Mesmo com o impacto observado nos resultados de RTT decorrente da QoS, a abordagem proposta apresentou melhores



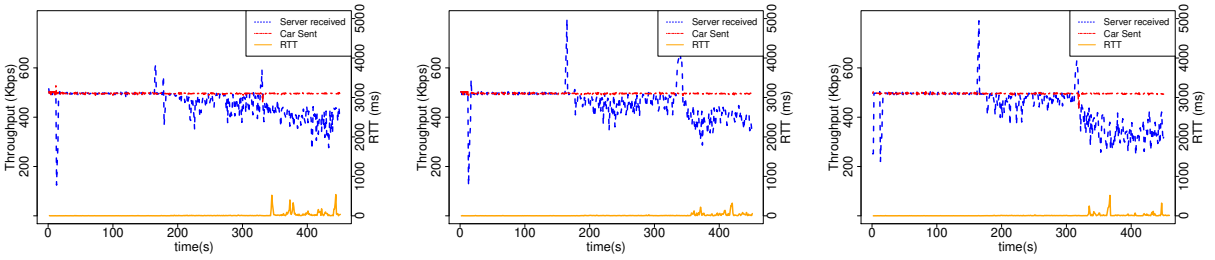
(a) App E - “Framework” (b) App E - “QoS only” (c) App E - “RMSDVN”



(d) App $E2$ - “Framework” (e) App $E2$ - “QoS only” (f) App $E2$ - “RMSDVN”



(g) App G - “Framework” (h) App G - “QoS only” (i) App G - “RMSDVN”



(j) App S - “Framework” (k) App S - “QoS only” (l) App S - “RMSDVN”

Figura 20 – Resultados de *throughput* e RTT durante o tempo de avaliação para as aplicações E, $E2$, G e S, utilizando as diferentes abordagens avaliadas. Fonte: o autor, publicado em [11].

resultados nessa métrica do que a abordagem “QoS only”. Em relação à aplicação $E2$, os resultados obtidos com o framework proposto foram semelhantes aos da abordagem “QoS only”, com os piores resultados nos últimos 100 s, conforme esperado, uma vez que a solução priorizou a aplicação E. Os valores de RTT foram proporcionais ao nível de congestionamento e naturalmente piores quando o tráfego foi limitado. Como a abordagem “RMSDVN” não inspeciona o tráfego para fornecer alguma priorização, seus valores de RTT são menores. Em todo caso, valores altos de RTT não constituem um problema nos cenários avaliados, visto que o KPI utilizado como referência foi a taxa de dados. Em relação à

aplicação G, é possível observar os melhores resultados com a abordagem "RMSDVN", o que era esperado, uma vez que tanto o framework proposto quanto a abordagem "QoS only" limitam a comunicação dessa aplicação para priorizar as demais.

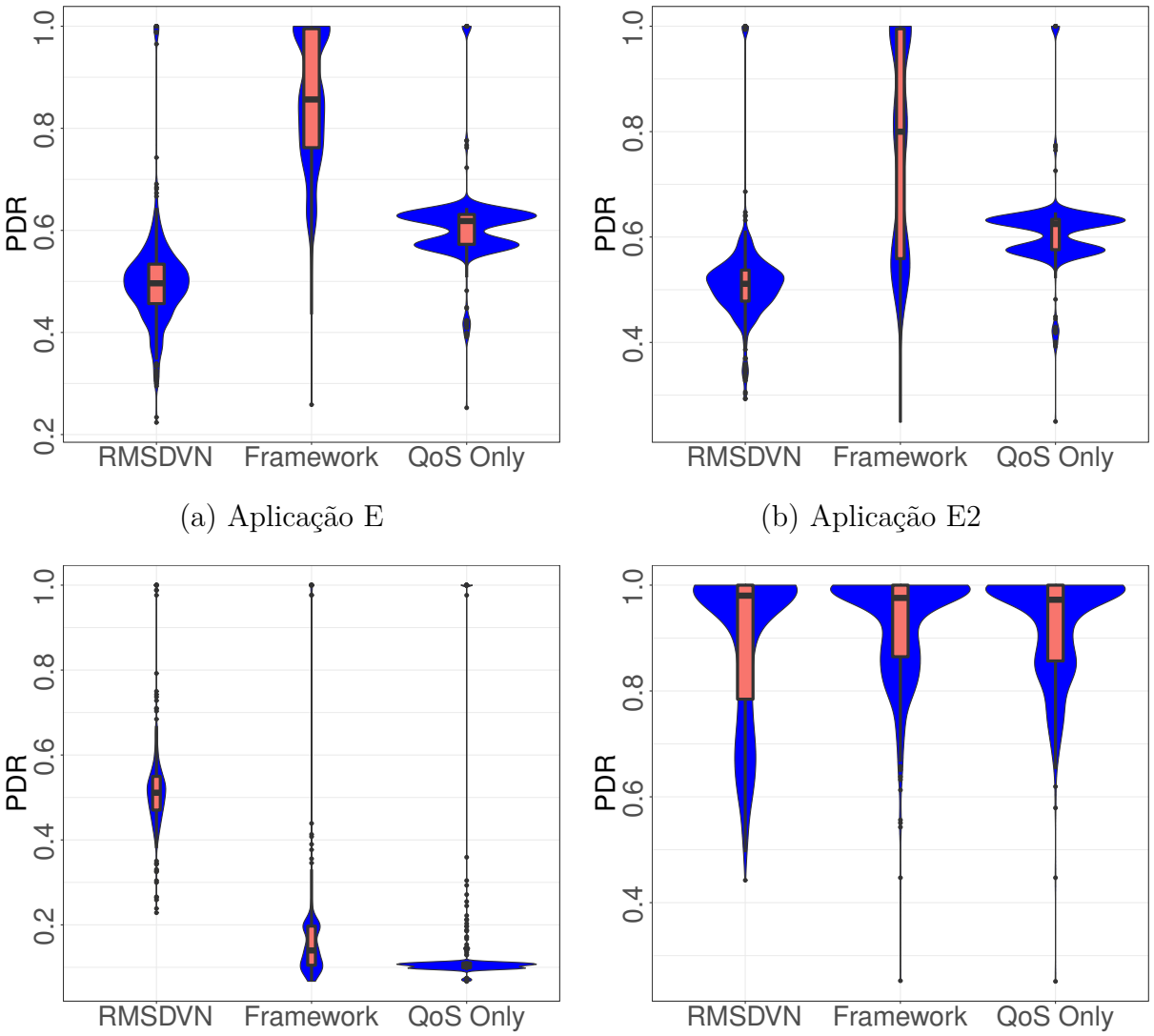
As Figuras 20(j), (k) e (l) mostram os resultados de RTT, taxas de *throughput* de transmissão e recepção coletados ao longo do tempo para a aplicação S. Uma vez que os dados dessas aplicações não passam pelo *backbone* da rede, o resultado esperado é que sejam independentes da abordagem utilizada. Os resultados são bastante semelhantes e consistentes com a teoria, exceto por alguns *outliers*. Observa-se também que, embora o congestionamento no *backbone* não impacte o tráfego dessas aplicações, esse tráfego não é imune à alta interferência dos 50 veículos gerando diferentes taxas de dados nas RANs. Portanto, é possível observar uma redução na taxa de recepção durante os últimos 100 s do tempo de avaliação em todas as figuras, em função da alta densidade de veículos transmitindo. Os valores de RTT foram baixos, conforme esperado em função da utilização de servidores MEC.

5.4.2 Taxa de entrega de pacotes

Nesta seção são analisados os valores de PDR obtidos com as abordagens avaliadas. As Figuras 21(a), (b), (c) e (d) apresentam os gráficos *violin plot* com o PDR calculado durante os experimentos utilizando as abordagens avaliadas para as aplicações E, $E2$, G e S, respectivamente.

É possível observar que os valores com a abordagem "RMSDVN" concentram-se principalmente entre 40% e 60% para as aplicações E, $E2$ e G, uma vez que não há priorização no *backbone* com essa abordagem. Com a abordagem "QoS only", os resultados são semelhantes para as aplicações prioritárias E e $E2$, pois com as regras e filas de QoS é possível alocar alguma largura de banda da rede. Entretanto, não há redirecionamento de tráfego nessa abordagem. Com as perdas de pacotes, os valores de PDR com "QoS only" concentraram-se em torno de 60%, exceto para a aplicação G, que foi penalizada por não ter prioridade. A solução proposta obteve os melhores valores de PDR para as aplicações E e $E2$, pois o framework considera os requisitos das aplicações e a mobilidade em cada RAN. Mesmo para a aplicação G, que é penalizada, a proposta apresentou melhores resultados do que a abordagem "QoS only".

Em resumo, considerando todo o tempo de avaliação, o PDR médio com a proposta foi de aproximadamente 85,1% para a aplicação E, contra respectivamente 50,1% e 60,8% com "RMSDVN" e "QoS only". Para a aplicação $E2$, o PDR médio foi de 76,9% com a proposta, contra respectivamente 51,1% e 61,1% com "RMSDVN" e "QoS only". Esses resultados evidenciam o desempenho superior da proposta quando se considera a métrica PDR para avaliar a operação relacionada às aplicações prioritárias.



(a) Aplicação E

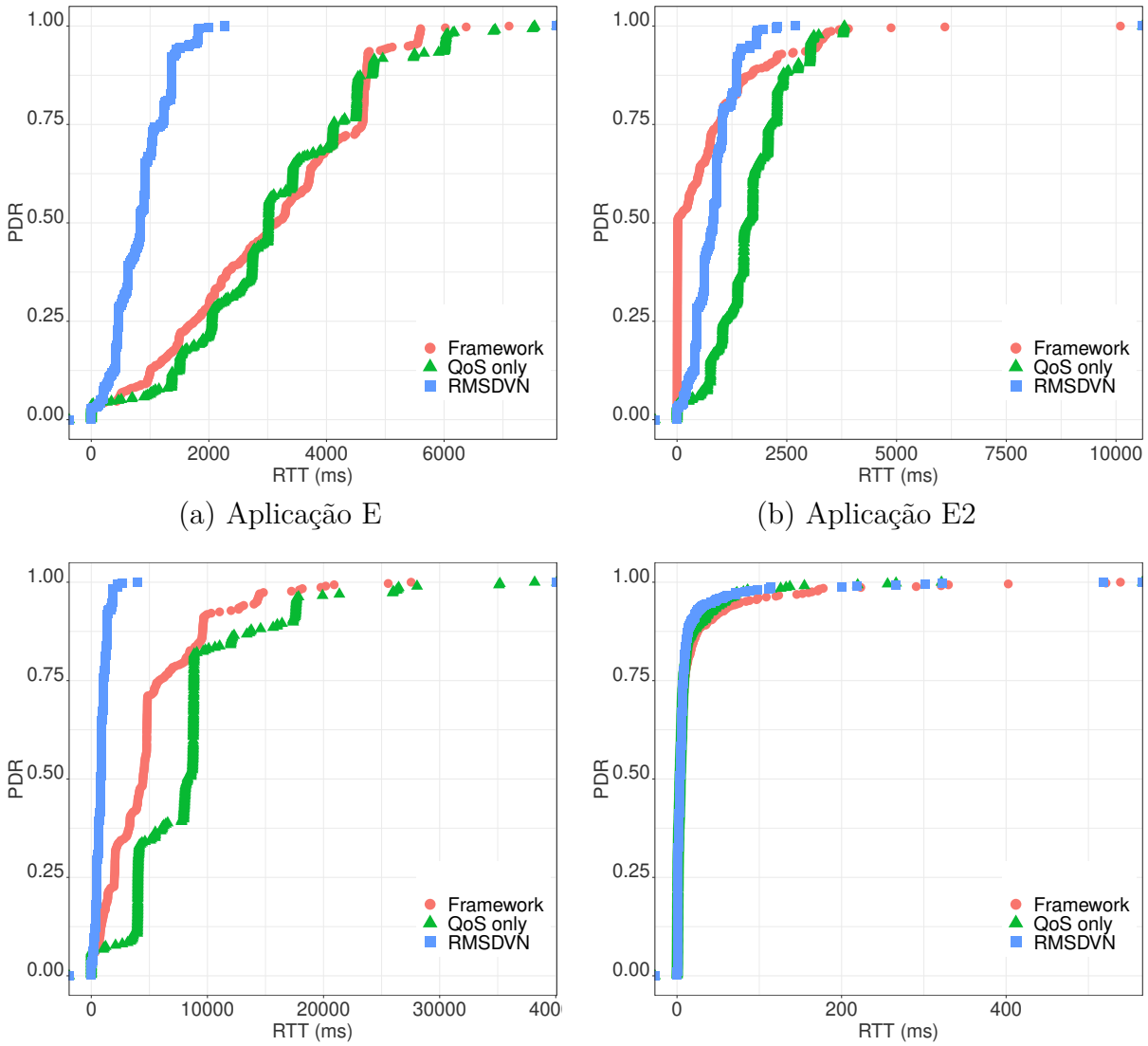
(b) Aplicação E2

(c) Aplicação G

(d) Aplicação S

Figura 21 – Taxa de entrega de pacotes para as diferentes aplicações nas abordagens avaliadas. Fonte: o autor, publicado em [11].

Considerando os valores de todo o tempo de emulação, o PDR médio com todas as abordagens para a aplicação S é de aproximadamente 90%, o que confirma os benefícios da utilização de solução MEC e a independência dessa métrica em relação à abordagem utilizada. Esses resultados são praticamente iguais, uma vez que os dados da aplicação S não passam pelo núcleo da rede e, portanto, não são manipulados pelas abordagens implementadas. Por outro lado, as observações de PDR inferior a 90% são evidências adicionais do impacto da interferência gerada pelos veículos em cada RAN durante os períodos com maior densidade veicular.



(a) Aplicação E (b) Aplicação E2

(c) Aplicação G (d) Aplicação S

Figura 22 – ECDFs dos resultados de RTT para as diferentes aplicações utilizando as abordagens avaliadas. Fonte: o autor, publicado em [11].

5.4.3 Tempo de ida e volta (RTT)

Os resultados de RTT por meio de funções de distribuição cumulativa empírica (ECDFs, do inglês *Empirical Cumulative Distribution Functions*) para as aplicações E, $E2$, G e S são ilustrados nas Figuras 22(a), (b), (c) e (d), respectivamente.

A ECDF da aplicação E mostra que mais de 90% dos valores são inferiores a 1750 ms com a abordagem "RMSDVN", enquanto com a solução proposta e "QoS only" são inferiores a aproximadamente 5000 ms. Como a utilização de QoS afeta o RTT, os melhores resultados são obtidos com a abordagem "RMSDVN" quando se consideram as aplicações que utilizam os *uplinks* nas RANs.

Observou-se que, dado que o framework proposto utiliza QoS aplicada pelo controlador central, os valores de RTT foram superiores aos da abordagem "RMSDVN".

Entretanto, isso não constitui um problema nesse cenário, uma vez que, conforme mencionado anteriormente, essa não era a métrica KPI considerada pelos algoritmos de controle. A aplicação S é a que apresenta menores valores de RTT.

Para a aplicação S, as medições de RTT nas comunicações de dados entre veículos e servidores MEC apresentam resultados de baixo RTT com as abordagens avaliadas, com média de aproximadamente 20 ms, independentemente da abordagem utilizada. Isso demonstra os benefícios dos servidores MEC em relação a essa métrica. Outra conclusão baseada nesses resultados é que a alta interferência não impacta o RTT no mesmo nível que impacta a métrica PDR.

5.4.4 Discussão

Os resultados demonstraram que, com a utilização do framework proposto, é possível orquestrar os recursos de rede para responder à dinâmica do ambiente veicular e atender às prioridades das aplicações ITS e seus KPIs. Em comparação com as outras abordagens, foram obtidos resultados promissores de taxa de dados recebida nos servidores de aplicação e PDR, em condições extremas de congestionamento de dados e interferência em um cenário veicular com topologia de rede variante no tempo.

Observou-se nos resultados que a utilização de políticas de QoS pode aumentar o RTT. Como as aplicações sensíveis a restrições temporais podem utilizar uma abordagem MEC, e no cenário da prova de conceito o KPI utilizado pelos controladores para gerenciar as aplicações em conjunto com sua prioridade foi a taxa de dados, os valores mais elevados de RTT não representam um problema.

A solução proposta apresentou para a aplicação E um resultado de PDR 39,96% superior ao obtido com a abordagem “QoS only” e 69,86% superior ao obtido com a abordagem “RMSDVN”. Cabe registrar que todos os resultados poderiam ser ainda melhores se não houvesse a limitação na implementação do controlador local, na qual é necessário aguardar pelo menos 20 segundos para atuar após detectar a necessidade de intervir para atender aos KPIs das aplicações, o que representa um período significativo considerando o tempo total de avaliação de 450 segundos.

5.5 Considerações finais do capítulo

Este capítulo tratou do problema de como implantar uma infraestrutura móvel com rede veicular capaz de atender dinamicamente aos requisitos de comunicação de diferentes aplicações ITS. A abordagem proposta consiste em um framework que utiliza conceitos de fatiamento de rede 5G em uma rede veicular definida por software, com algoritmos que consideram os requisitos das diferentes aplicações, sua infraestrutura e a mobilidade dos veículos para atender aos requisitos das aplicações [11].

Por meio da avaliação experimental realística, foi possível validar a abordagem e demonstrar como a taxa de dados recebida pelos servidores de aplicação ao longo do tempo, o RTT e o PDR dos fluxos de comunicação foram utilizados como métricas de desempenho. Os resultados obtidos superaram as abordagens alternativas "QoS only" e "RMSDVN" em cenários de congestionamento dinâmico com veículos associados a quatro aplicações com diferentes requisitos de taxa de dados e prioridades. Trabalhos publicados após este artigo, como [115] e [116], adotaram independentemente arquiteturas e taxonomias de serviço alinhadas à proposta aqui apresentada, o que reforça sua contribuição ao estado da arte.

Duas limitações merecem registro. Primeira, a avaliação experimental utilizou IEEE 802.11g como proxy para o acesso via rádio 5G; embora os algoritmos de controle SDN no núcleo da rede operem de forma independente da tecnologia de acesso, uma avaliação futura com simuladores 5G NR nativos permitiria quantificar efeitos específicos da camada de rádio sobre as métricas de desempenho das fatias. Segunda, o framework baseia-se em informações cadastrais de subscrição dos veículos para estimar o consumo de recursos de cada fatia. Em operação, as condições reais de tráfego, mobilidade e canal divergem dessas estimativas iniciais, e o framework não dispõe de um mecanismo para avaliar, em tempo de execução, o impacto dessas divergências sobre o desempenho das fatias ativas. Essa segunda limitação motiva a extensão apresentada no Capítulo 6, que introduz classificação de impacto de políticas de fatiamento com aprendizado de máquina na borda da rede.

6 AIMS: GERENCIAMENTO ADAPTATIVO E INTELIGENTE DE FATIAS PARA REDES ITS DE PRÓXIMA GERAÇÃO

O framework apresentado no Capítulo 5 demonstrou a viabilidade de utilizar fatiamento de rede 5G com orquestração dinâmica de recursos para atender aplicações ITS com requisitos heterogêneos. Aquela abordagem, porém, depende de informações cadastrais de subscrição dos veículos para estimar o consumo de recursos de cada fatia, sem possibilitar o ajuste dinâmico da configuração com base no consumo real e nas condições operacionais observadas em tempo de execução. Em ambientes veiculares com alta variabilidade de mobilidade, carga e condições de canal, essa limitação compromete a capacidade de manter os níveis de serviço exigidos por aplicações ITS.

Este capítulo apresenta o framework AIMS (*Adaptive and Intelligent Management of Slicing*), que endereça essa limitação ao introduzir inteligência leve na borda da rede para análise de impacto de políticas de fatiamento [12]. No contexto da evolução para redes 6G, cujo roteiro prevê integração nativa de IA nas funções de rede [132], o AIMS propõe uma arquitetura híbrida *cloud-edge* que cobre as quatro fases do ciclo de vida das fatias conforme definido pelo 3GPP [133], distribuindo funções especializadas entre controladores centrais e locais. A contribuição experimental central é a validação da função de classificação de impacto, que mapeia sinais multidimensionais de monitoramento em uma taxonomia semântica de quatro níveis — *Adequate*, *Warning*, *Severe* e *Critical* — projetada como gatilho estável para ações de adaptação em *closed-loop*.

A validação experimental apresentada neste capítulo restringe-se ao componente de classificação de impacto, que atua como gatilho interpretável na fase de operação das fatias (F4/F5). Cabe ressaltar que o modelo conceitual do AIMS propõe um ciclo de vida completo com sete funções (F1–F7), conforme detalhado na Seção 6.2. Como as demais etapas operam em nível estritamente arquitetural neste trabalho, a avaliação empírica de toda a malha *closed-loop* em operação conjunta extrapola o escopo atual, configurando-se como uma direção para pesquisas futuras.

O restante deste capítulo está organizado da seguinte forma. A Seção 6.1 posiciona o AIMS em relação aos trabalhos correlatos em gerenciamento de fatiamento com inteligência artificial. A Seção 6.2 descreve a arquitetura do framework e suas funções ao longo do ciclo de vida. A Seção 6.3 detalha a taxonomia de classificação de impacto e a formulação do problema. A Seção 6.4 apresenta a configuração experimental, e a Seção 6.5 discute os

resultados obtidos. Por fim, a Seção 6.6 sintetiza as contribuições e conecta este capítulo ao Capítulo 7, que estende a abordagem com aprendizado federado.

6.1 Posicionamento em relação aos trabalhos correlatos

A pesquisa sobre gerenciamento de fatiamento para redes veiculares segue duas vertentes principais: controle de recursos com IA e arquiteturas de fatiamento AI-native. Levantamentos recentes indicam que muitas abordagens baseadas em aprendizado de máquina (ML, do inglês *Machine Learning*) ainda tratam funções isoladas do ciclo de vida das fatias — como admissão, escalonamento e posicionamento — com integração limitada entre as fases de gerenciamento [2]. No domínio específico de redes veiculares, Hamdi et al. [83] apresentam um levantamento abrangente de técnicas de aprendizado de máquina aplicadas ao fatiamento para IoV em redes 5G e B5G, confirmando que a maioria das propostas se concentra em funções individuais. Uma dimensão complementar, a segurança no fatiamento de rede, é objeto de levantamentos específicos [134] e será abordada no Capítulo 8.

Na vertente algorítmica, o aprendizado por reforço profundo (*Deep Reinforcement Learning* — DRL) é amplamente empregado para alocação dinâmica de recursos, com propostas de projetos hierárquicos para controle de admissão [135], esquemas multiagente para compartilhamento de capacidade na RAN [136] e métodos ator-crítico para provisionamento V2V sob observabilidade parcial [137]. Essas abordagens focam na otimização de recursos em nível operacional. Trabalhos recentes começam a integrar técnicas de IA explicável (XAI) a modelos DRL para aumentar a transparência das decisões de alocação em cenários veiculares [138], porém sem produzir sinais de impacto semanticamente categorizados que sirvam como gatilhos diretos para ações no ciclo de vida das fatias.

De forma complementar, modelos supervisionados e híbridos de aprendizado profundo foram explorados para classificação de tráfego e seleção de tipo de fatia, tipicamente alinhados à fase de *Preparação*. Dangi e Lalwani [139] propõem um modelo CNN+BiLSTM para classificar tráfego e atribuir tipos de fatia, porém sem tratar a interpretação contínua de telemetria em tempo de execução ou a geração de gatilhos para o ciclo de vida durante a fase de *Operação*. Na vertente arquitetural, propostas de frameworks de fatiamento AI-native e inteligência de borda motivam uma integração mais estreita de IA entre os domínios da rede [140, 141, 3]. Wu et al. [142] propõem um framework de *digital twin* de rede AI-native com modelos hierárquicos de DT para predição de estado e tomada de decisão adaptativa em redes 6G. Embora complementares, esses trabalhos e discussões mais amplas sobre garantia de serviço assistida por IA [143, 144] não oferecem um mecanismo concreto e leve que traduza telemetria ruidosa de tempo de execução em *níveis de impacto semanticamente interpretáveis*, projetados especificamente como gatilhos estáveis para

ações de *closed-loop* do ciclo de vida em cenários ITS realísticos.

O AIMS se diferencia ao operacionalizar a **análise de impacto de políticas** como função de primeira classe no plano de controle. Ao mapear sinais multidimensionais de monitoramento em uma taxonomia semântica de quatro níveis (*Adequate*, *Warning*, *Severe*, *Critical*), o framework produz gatilhos proativos e interpretáveis para adaptação de fatias, evitando o comportamento de oscilação (*flapping*) associado a regras estáticas baseadas em limiares.

6.2 Arquitetura do framework AIMS

O framework AIMS introduz uma arquitetura híbrida orientada por IA para o gerenciamento fim-a-fim de fatias de rede. Sua operação é estruturada em torno das quatro fases padronizadas do ciclo de vida das fatias, orquestradas por um conjunto de funções especializadas e um plano de controle híbrido. A Tabela 7 resume as sete funções centrais do framework, e a Figura 23 ilustra a arquitetura completa ao longo das quatro fases.

O plano de controle do AIMS compreende dois controladores inteligentes. O **Controlador Central** (CC, do inglês *Central Controller*) opera no núcleo/nuvem da rede, possui visão global dos recursos e é responsável por decisões estratégicas como planejamento global de recursos e otimização em escala de rede. O **Controlador Local** (LC, do inglês *Local Controller*) é implantado na borda, co-localizado com nós MEC/RSU, e é responsável por tarefas em tempo real como coleta de dados, tradução de perfis e execução adaptativa de políticas.

Tabela 7 – Visão geral das funções centrais do framework AIMS.

ID	Função	Controlador	Fase
F1	Tradução de Perfil (*Profile Translation*)	LC	Preparação
F2	Planejamento Global de Recursos (*Global Resource Planning*)	CC	Comissionamento
F3	Provisionamento de Borda (*Edge Provisioning*)	LC	Comissionamento
F4	Supervisão e Otimização Global (*Global Supervision & Optimization*)	CC	Operação
F5	Adaptação e Reporte Local (*Local Adaptation & Reporting*)	LC	Operação
F6	Descomissionamento Global (*Global Decommissioning*)	CC	Descomissionamento
F7	Descomissionamento Local (*Local Decommissioning*)	LC	Descomissionamento

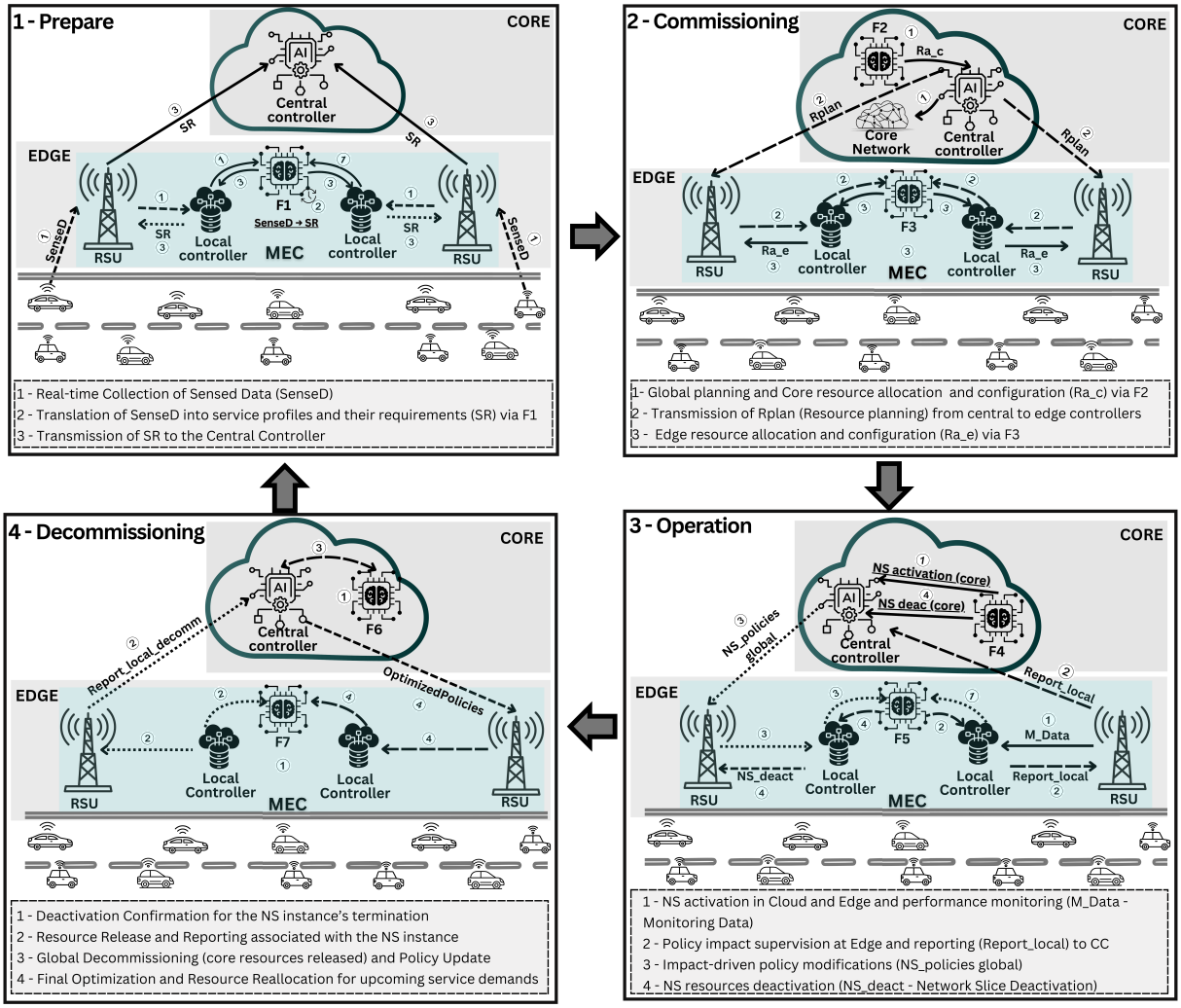


Figura 23 – Arquitetura híbrida do AIMS ao longo das quatro fases do ciclo de vida: Preparação (Painel 1), Comissionamento (Painel 2), Operação (Painel 3) e Descomissionamento (Painel 4). As funções F1–F7 e os fluxos de comunicação entre o Controlador Central (CC) e os Controladores Locais (LCs) são indicados em cada painel. Fonte: o autor, publicado em [12].

As subseções a seguir descrevem o funcionamento do AIMS em cada fase do ciclo de vida.

6.2.1 Fase 1: Preparação das fatias

O objetivo da fase de preparação é traduzir dados ambientais coletados em tempo real em requisitos técnicos para alocação de recursos (Painel 1 da Figura 23). Os LCs coletam continuamente dados sensoriais (*SenseD*) a partir de veículos e do ambiente de rede local por meio de suas RSUs associadas, capturando informações de mobilidade veicular, condições de canal e uso de aplicações.

Cada LC executa sua função de Tradução de Perfil (F1) para converter os dados brutos em perfis de serviço padronizados (ex.: eMBB, mMTC, URLLC) que definem as

necessidades de alto nível das aplicações [83]. A partir desses perfis, F1 deriva **Requisitos de Fatia** (*Slice Requirements* — SR), expressos como metas e restrições de QoS acionáveis. Por exemplo, para URLLC: latência ≤ 5 ms, taxa de entrega de pacotes (PDR) de 99,999% e *throughput* ≥ 5 Mbps. Os LCs transmitem os SRs ao CC, que os agrega para construir uma visão global da rede para planejamento otimizado. Esse processo opera periodicamente, permitindo que o sistema se adapte continuamente a mudanças nas condições operacionais.

6.2.2 Fase 2: Comissionamento das fatias

Na fase de comissionamento, o AIMS cria e configura novas instâncias de fatia com base nos SRs coletados na fase de preparação (Painel 2 da Figura 23). O CC executa a função de Planejamento Global de Recursos (F2) para gerar um plano global de alocação de recursos (**Rplan**), avaliando os SRs agregados contra a disponibilidade atual de recursos e configurando recursos do núcleo da rede — por exemplo, encadeamento de VNFs e parâmetros de QoS. O Rplan resultante especifica tanto as alocações no nível do núcleo (RaL_c) quanto os orçamentos de recursos delegados a cada domínio de borda.

O CC então dissemina o Rplan a todos os LCs relevantes. Guiado por esse plano e informado pelos dados locais em tempo real ($SenseD$), cada LC executa sua função de Provisionamento de Borda (F3), instanciando VNFs em nós MEC e aplicando parâmetros específicos de borda (RaL_e). Esse mecanismo garante que as políticas globais sejam aplicadas com adaptação sensível ao contexto local.

6.2.3 Fase 3: Operação das fatias

A fase de operação constitui um ciclo contínuo de monitoramento, análise e adaptação para manter o desempenho das fatias. Conforme ilustrado no Painel 3 da Figura 23, essa fase equilibra autonomia local e otimização global por meio de F4 (Supervisão e Otimização Global) no CC e F5 (Adaptação e Reporte Local) nos LCs. O ciclo opera em quatro passos.

No **Passo 1 (Ativação e Monitoramento)**, o AIMS ativa as instâncias de fatia no núcleo e na borda, e coleta dados de monitoramento em tempo real (M_Data) por meio dos LCs e do CC. Esses dados incluem métricas de rede (RTT, PDR) e informações de contexto (SLAs, mobilidade).

No **Passo 2 (Supervisão de Impacto e Reporte)**, os LCs classificam o impacto das políticas a partir de M_Data utilizando modelos de ML na função F5 e geram um relatório local ($Report_{\text{local}}$) para o CC. Esse relatório contém um resumo compacto com níveis de impacto (*Adequate*, *Warning*, *Severe*, *Critical*) e agregados de KPIs.

No **Passo 3 (Modificações de Políticas Orientadas por Impacto)**, modificações são acionadas quando violações são previstas. F4 agrega os relatórios locais e avalia

impactos globais por meio do modelo de ML do CC, emitindo políticas globais otimizadas (*NS_policies_global*) para fatias com alto impacto. Os LCs aplicam ajustes locais via F5 e confirmam as alterações, alcançando eficiência de duplo ciclo (*dual-loop*).

No **Passo 4 (Desativação de Instâncias)**, instâncias de fatia que não são mais necessárias são desativadas: F4 agenda desativações no núcleo (*NS_deact*) enquanto F5 libera recursos de borda. O ciclo se repete a cada intervalo Δt para resposta proativa a mudanças na rede.

6.2.4 Fase 4: Descomissionamento das fatias

A fase de descomissionamento assegura a reclamação completa e coordenada de recursos (Painel 4 da Figura 23). Os LCs executam a função F7 para encerrar componentes de borda e reportar o status de descomissionamento ao CC. O CC então executa F6 para liberar recursos do núcleo e acionar reotimização global (*OptimizedPolicies*). Esse processo sincronizado previne vazamento de recursos e reintegra a capacidade recuperada para uso por fatias futuras.

Essa arquitetura fim-a-fim e híbrida possibilita o gerenciamento de fatias ao integrar inteligência distribuída na borda com controle centralizado. Ao estabilizar os gatilhos de decisão sob telemetria ruidosa, o AIMS visa reduzir oscilações de reconfiguração (*flapping*) e sobrecarga desnecessária de orquestração em ambientes ITS dinâmicos.

Embora uma implementação completa de *closed-loop* esteja fora do escopo da avaliação apresentada neste capítulo, os níveis de impacto produzidos pela função de classificação constituem os sinais de decisão que um orquestrador SDN/NFV necessita para acionar atualizações de política — como escalonamento, migração ou controle de admissão — antes que violações de SLA se tornem críticas. A seção a seguir detalha a função inteligente central avaliada neste trabalho: a análise de impacto baseada em ML.

6.3 Taxonomia de classificação de impacto e formulação do problema

A função central de inteligência do AIMS é a classificação de impacto de políticas, que traduz sinais multidimensionais de monitoramento em níveis semânticos interpretáveis. Esta seção descreve a taxonomia de impacto, o esquema de rotulagem e a justificativa para o uso de modelos de ML.

6.3.1 Engenharia de atributos

Para produzir sinais estáveis e significativos a partir de medições voláteis de rede (latência, perda de pacotes, *throughput*), o AIMS aplica médias móveis de janela curta (3 amostras) e realiza recorte de valores nos percentis 1 e 99 para mitigação de *outliers*. Variáveis contextuais são codificadas conforme os requisitos de cada modelo: codificação *one-hot*/ordinal para Random Forest e TabNet, e tratamento nativo de variáveis categóricas para CatBoost. Para prevenir vazamento de dados (*data leakage*), todas as etapas de pré-processamento — como, por exemplo, o `StandardScaler` — são ajustadas exclusivamente nos dados de treinamento antes de serem aplicadas ao conjunto de teste.

6.3.2 Esquema de rotulagem de impacto

O AIMS define quatro níveis de impacto para classificação do estado operacional de cada fatia:

- **Nível 0 (Adequate):** a aplicação opera conforme o esperado, sem degradação observável.
- **Nível 1 (Warning):** degradação menor, frequentemente imperceptível aos usuários, mas que sinaliza tendência de piora.
- **Nível 2 (Severe):** degradação perceptível que requer ação corretiva para evitar violação de SLA.
- **Nível 3 (Critical):** aplicação comprometida, com falha iminente.

Cada métrica de rede recebe uma pontuação individual com base em limiares específicos por classe de aplicação, calibrados a partir de especificações 3GPP/ETSI/5GAA e ajustados ao dataset utilizado. A Tabela 8 apresenta os limiares e pesos por classe de aplicação. O rótulo final de impacto I é uma média ponderada normalizada das pontuações individuais, refletindo a importância relativa de cada métrica para a aplicação em questão — por exemplo, a latência tem maior peso para aplicações de segurança (*Safety*), enquanto o *throughput* é mais crítico para entretenimento (*Entertainment*):

$$I = \text{round}(w_{\text{lat}} \cdot S_{\text{lat}} + w_{\text{loss}} \cdot S_{\text{loss}} + w_{\text{thr}} \cdot S_{\text{thr}}) \,, \tag{6.1}$$

onde S representa as pontuações individuais de cada métrica e w os pesos normalizados.

6.3.3 Justificativa para o uso de aprendizado de máquina

A regra de rotulagem definida na Equação 6.1 é utilizada *offline* para atribuir rótulos de impacto consistentes, ancorados em limiares derivados de padrões técnicos,

Tabela 8 – Limiares de classificação de impacto e pesos das métricas por classe de aplicação.

Classe	**Pesos**			**Latência (ms)**			**Perda de Pacotes (%)**			**Vazão (kbps)**		
	w_L	w_P	w_T	Adeq.	Warn.	Sev.	Adeq.	Warn.	Sev.	Adeq.	Warn.	Sev.
Safety (S)	0,5	0,3	0,2	≤100	≤200	≤500	≤1	≤5	≤10	≥450	≥400	≥300
Efficiency (E)	0,4	0,3	0,3	≤300	≤700	≤1500	≤5	≤10	≤20	≥800	≥600	≥400
Entertainment (E2)	0,4	0,2	0,4	≤100	≤250	≤1000	≤2	≤5	≤10	≥8000	≥3000	≥1000
Generic (G)	0,3	0,3	0,4	≤1000	≤3000	≤5000	≤10	≤30	≤50	≥5000	≥2000	≥500

w_L: peso da latência; w_P: peso da perda de pacotes; w_T: peso do *throughput*.

sobre dados pós-processados. Aplicar a mesma regra determinística *online*, contudo, apresenta fragilidades em implantações ITS. A telemetria em ambientes veiculares é ruidosa, parcialmente observável e sujeita a interações não lineares entre KPIs. Essas características podem induzir oscilações frequentes de estado (*flapping*) e reconfigurações desnecessárias quando se empregam limiares rígidos.

Um classificador treinado por ML, por outro lado, opera sobre o vetor completo de atributos (incluindo agregados temporais) e pode aprender fronteiras de decisão robustas que toleram melhor o ruído e capturam interações entre KPIs. O resultado é a produção de gatilhos de impacto mais estáveis e generalizáveis para ações de *closed-loop*, em comparação com a abordagem puramente baseada em regras.

6.4 Configuração experimental

A avaliação experimental foca na validação da viabilidade da função de classificação de impacto do AIMS. Esta seção descreve o dataset, o protocolo de avaliação e os modelos comparados.

6.4.1 Dataset e cenário

A avaliação utiliza um dataset realístico de ITS composto por **5.400 amostras** e **25 *features***, derivados de fluxos de monitoramento de um corredor veicular urbano com congestionamento, originalmente gerado pelo cenário de simulação descrito no Capítulo 5 [11]. O cenário envolve 158 veículos, com até 15 disputando recursos de rede simultaneamente em uma mesma área de cobertura. O dataset inclui quatro classes de aplicação ITS: *Safety* (S), *Efficiency* (E), *Entertainment* (E2) e *Generic* (G). Os *features* incluem métricas de rede (latência, taxa de perda de pacotes, *throughput*) e variáveis contextuais (tipo de aplicação, condições de mobilidade).

Conforme descrito na Seção 6.3.2, cada amostra recebe um rótulo de impacto entre os quatro níveis definidos. A distribuição natural das classes é desbalanceada: *Adequate* (14%), *Warning* (19%), *Severe* (50%) e *Critical* (17%).

6.4.2 Protocolo de avaliação *leakage-free*

Para garantir uma avaliação robusta e livre de viés, o protocolo experimental foi estruturado em cinco estágios: pré-processamento, rotulagem de impacto, balanceamento de classes com particionamento temporal, otimização de hiperparâmetros (HPO) específica por modelo, e avaliação final em conjunto de teste retido (*hold-out*).

Balanceamento de classes e particionamento temporal. Para evitar que os modelos sejam enviesados pela classe majoritária (*Severe*, com 50% das amostras), aplica-se ponderação de classes durante o treinamento, atribuindo maior importância a classes menos frequentes. Para respeitar a dependência temporal e simular um cenário realístico de implantação, os dados são particionados em blocos de 60 segundos. Essa partição garante que amostras do mesmo bloco temporal não apareçam simultaneamente nos conjuntos de treinamento e teste, prevenindo vazamento temporal de dados. Vinte por cento dos blocos são reservados como conjunto de teste retido (*blind hold-out*) para avaliação final.

Otimização de hiperparâmetros. A HPO é conduzida com otimização Bayesiana por meio do algoritmo TPE (*Tree-structured Parzen Estimator*), implementado no framework Optuna. O espaço de busca inclui parâmetros-chave para cada modelo: número e profundidade de árvores para Random Forest; taxa de aprendizado e estratégias de *bootstrap* para CatBoost; e complexidade arquitetural e regularização para TabNet. O desempenho é avaliado por validação cruzada com 5 *folds* com consciência de grupo (*group-aware*) sobre os blocos temporais, maximizando o macro F1-score. Para modelos iterativos, CatBoost utiliza *early stopping* nativo, enquanto TabNet emprega o *MedianPruner* do Optuna para interromper avaliações improdutivas precocemente.

Treinamento e avaliação finais. Após identificar os hiperparâmetros ótimos, cada modelo é retreinado no conjunto completo de treinamento. Random Forest, como modelo não iterativo, é treinado com todos os dados disponíveis. Para os modelos iterativos (CatBoost e TabNet), 10% dos dados de treinamento são reservados como conjunto interno de validação para *early stopping*. A avaliação final definitiva é realizada uma única vez no conjunto de teste retido, temporalmente disjunto, assegurando uma estimativa livre de vazamento e sem viés das capacidades de generalização dos modelos.

6.4.3 Modelos avaliados

Três modelos de aprendizado de máquina são comparados para a função de classificação de impacto:

- **Random Forest (RF):** modelo *ensemble* baseado em agregação de múltiplas árvores de decisão. Robusto a ruído e com boa capacidade de generalização, é um dos modelos

mais utilizados em problemas de classificação tabulares.

- **CatBoost:** modelo de *gradient boosting* com tratamento nativo de variáveis categóricas e mecanismo de *ordered boosting* que reduz o viés de previsão. Projetado para eficiência computacional, é um candidato para implantação em ambientes de borda com recursos limitados.
- **TabNet:** arquitetura de rede neural baseada em mecanismos de atenção (*attention-based*), projetada para dados tabulares. Incluída na avaliação para verificar se abordagens de aprendizado profundo baseadas em atenção oferecem vantagens sobre modelos *ensemble* clássicos no cenário investigado.

6.4.4 Reprodutibilidade

Todos os materiais suplementares — incluindo código-fonte, pipeline completo, dataset, modelos treinados e resultados de avaliação — estão disponíveis publicamente na cápsula Code Ocean 8789691 (DOI: 10.24433/CO.5216953.v2), que suporta execução interativa e citável em nuvem, e no repositório GitHub do projeto (saraivacode/AIMS). Todas as sementes aleatórias foram fixadas para garantir comportamento determinístico.

6.5 Resultados e discussão

Esta seção apresenta os resultados experimentais organizados em três dimensões: desempenho por nível de impacto, *trade-off* entre custo de treinamento e acurácia, e estabilidade dos modelos com diferentes orçamentos de otimização.

6.5.1 Desempenho por nível de impacto

A Figura 24 apresenta o detalhamento de desempenho por nível de impacto para os modelos em seus orçamentos de referência (15 avaliações para RF e CatBoost; 40 para TabNet). Os resultados revelam que, embora todos os modelos apresentem bom desempenho nas fronteiras de classificação (níveis extremos), a diferenciação principal ocorre nos níveis intermediários.

Para o nível **Adequate**, o desempenho é elevado em todos os modelos, com Random Forest e CatBoost atingindo F1-scores de 0,993. De forma similar, no nível **Critical**, as pontuações são próximas entre os modelos, com CatBoost apresentando leve vantagem (0,964 contra 0,949 do RF). O nível **Warning**, contudo, constitui o principal ponto de inflexão da avaliação: TabNet obtém F1-score de apenas 0,550, contra 0,888 do Random Forest e 0,881 do CatBoost. Essa fragilidade se estende ao nível **Severe**, onde o desempenho do TabNet (0,840) é inferior ao dos demais modelos (CatBoost: 0,953; RF: 0,950).

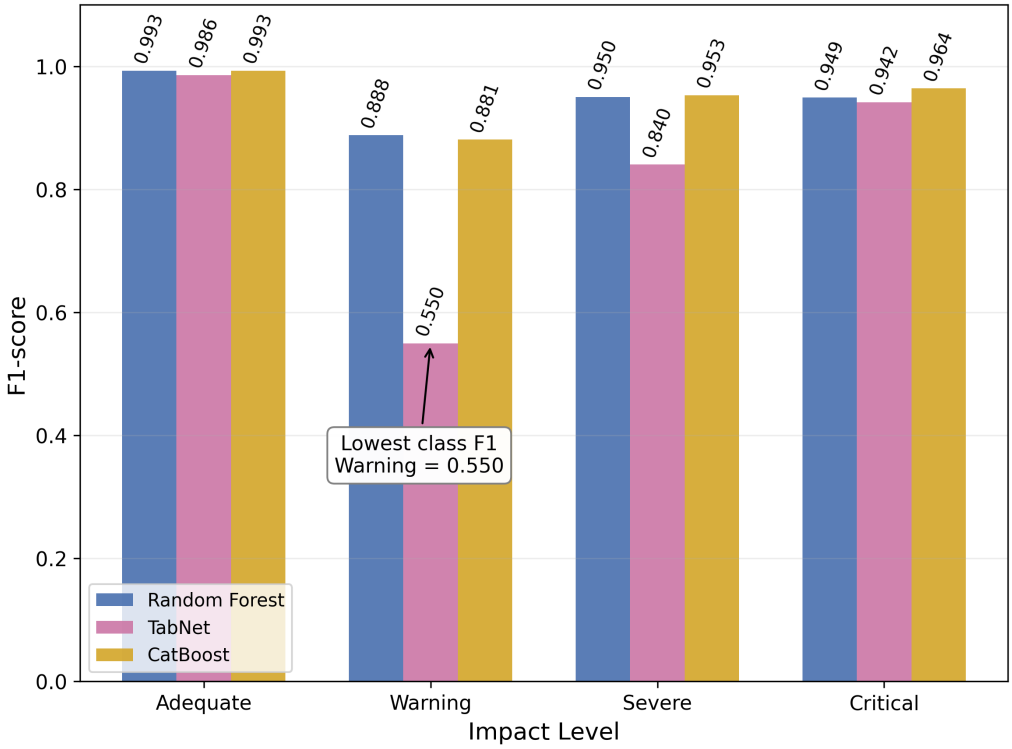


Figura 24 – F1-score por nível de impacto para os três classificadores. O TabNet apresenta degradação acentuada no nível *Warning* (F1 = 0,550), enquanto Random Forest e CatBoost mantêm desempenho superior a 0,88 em todos os níveis. Fonte: o autor, publicado em [12].
]

A incapacidade do TabNet de classificar de forma confiável os níveis intermediários — justamente os que sinalizam degradação incipiente e demandam ação preventiva — limita sua aplicabilidade em cenários de gerenciamento proativo de ITS. Modelos *ensemble* como Random Forest e CatBoost demonstram superioridade nessa tarefa. As Figuras 25, 26 e 27 detalham o padrão de erros por meio das matrizes de confusão normalizadas dos três modelos. O TabNet classifica aproximadamente 40% das amostras *Warning* como *Severe*, comprometendo a detecção precoce de degradação. RF e CatBoost mantêm confusão *Warning*→*Severe* de aproximadamente 14%.

6.5.2 *Trade-off* entre tempo de treinamento e acurácia

A Figura 28 apresenta a relação entre tempo total de treinamento (eixo x, escala logarítmica; inclui HPO com Optuna e ajuste final) e acurácia de teste (eixo y), nos orçamentos de referência de cada modelo. O tamanho dos pontos é proporcional ao macro F1-score.

A análise revela um *trade-off* claro entre os modelos. O CatBoost alcança a maior acurácia (94,8%) e o melhor macro F1-score (0,948) em 169 s, com 15 avaliações de HPO executadas em GPU. O Random Forest atinge acurácia competitiva de 94,4% (F1-Macro

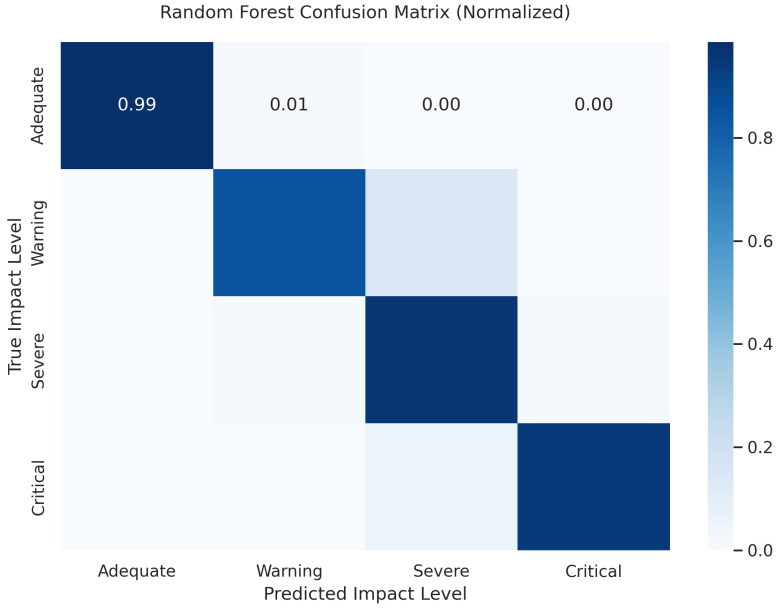


Figura 25 – Matriz de confusão normalizada — Random Forest. Confusão *Warning*→*Severe* de aproximadamente 14%. Fonte: o autor, publicado em [12]

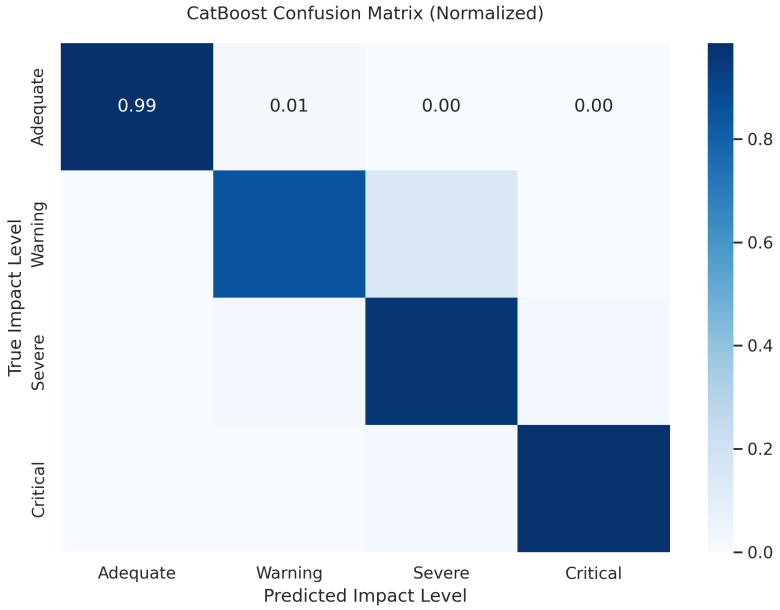


Figura 26 – Matriz de confusão normalizada — Random Forest. Confusão *Warning*→*Severe* de aproximadamente 14%. Fonte: o autor, publicado em [12]

0,945) em apenas 68 s — 2,5× mais rápido que o CatBoost —, posicionando-se como alternativa eficiente para implantação em nós de borda com recursos limitados. O TabNet, que requer 40 avaliações para convergência do TPE em seu espaço de busca de maior dimensionalidade ($d = 9$), apresenta a pior relação custo-efetividade: 3.947 s (23,3× mais lento que o CatBoost) para atingir apenas 82,9% de acurácia.

Esses resultados têm implicações diretas para a implantação do AIMS. O CatBoost constitui a escolha preferencial quando a acurácia de detecção é prioritária, com o melhor

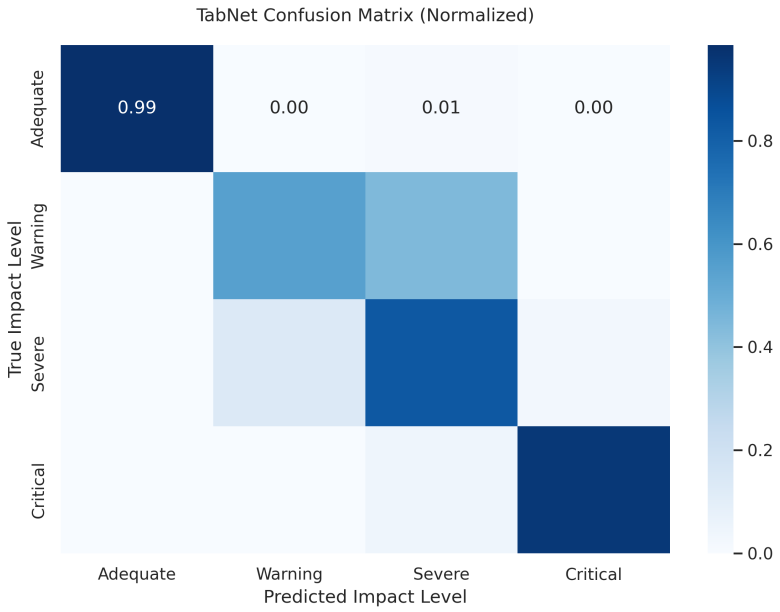


Figura 27 – Matriz de confusão normalizada — TabNet. Confusão *Warning*→*Severe* de aproximadamente 40%, evidenciando o colapso na fronteira entre os níveis intermediários. Fonte: o autor, publicado em [12]

desempenho geral entre os modelos avaliados. Em cenários onde os LCs na borda dispõem de recursos computacionais limitados, o Random Forest oferece desempenho competitivo com tempo de treinamento 2,5× menor, posicionando-se como alternativa custo-efetiva.

6.5.3 Gap de generalização

A Figura 29 compara o F1-Macro obtido em validação cruzada (CV) com o F1-Macro no conjunto de teste retido (*holdout*), quantificando o gap de generalização de cada modelo. Random Forest e CatBoost apresentam gaps moderados de 0,032 e 0,027, respectivamente, indicando que o desempenho observado em CV se transfere de forma consistente para dados não vistos. O TabNet, em contraste, apresenta o maior gap de generalização (0,102): o modelo atinge F1-Macro de 0,932 em CV, porém apenas 0,829 no *holdout*. Essa discrepância sugere que os hiperparâmetros selecionados pelo TPE favorecem os *folds* de validação cruzada em detrimento da capacidade de generalização, um comportamento coerente com a sensibilidade do TabNet ao espaço de busca de alta dimensionalidade ($d = 9$) discutida na seção seguinte.

6.5.4 Estabilidade e eficiência de otimização

Os modelos apresentam diferenças significativas em estabilidade e eficiência de otimização quando avaliados com 5, 15, 40 e 50 avaliações, conforme detalhado na Figura 30.

O Random Forest apresenta estabilidade excepcional, mantendo 94,4% de acurácia em todos os orçamentos de avaliação, com desvio padrão igual a zero — reflexo de

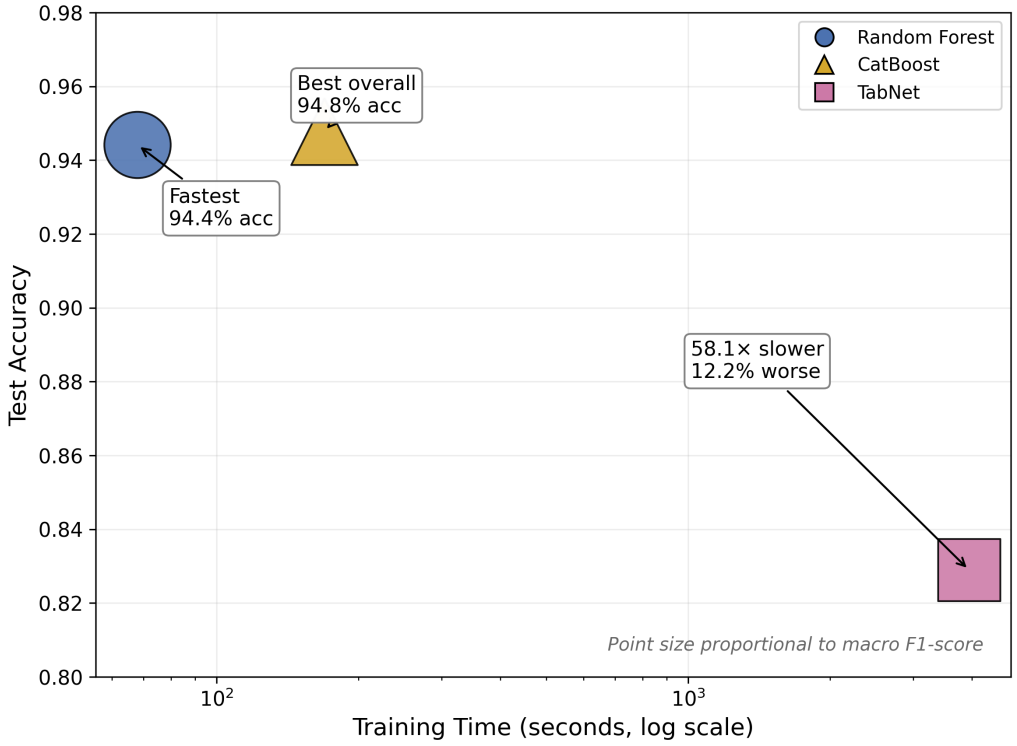


Figura 28 – Acurácia de teste versus tempo total de treinamento (escala logarítmica). O CatBoost alcança a maior acurácia (94,8%) em 169 s. O Random Forest é o mais rápido (68 s) com acurácia competitiva (94,4%). O TabNet apresenta a pior relação custo-efetividade: 3.947 s para 82,9% de acurácia (58,1× mais lento que o RF). Fonte: o autor, publicado em [12]

convergência do TPE já na primeira trial aleatória. O CatBoost apresenta variação de até 1,6 p.p. entre configurações (93,2%–94,8%), consistente com a natureza estocástica da busca em um espaço que inclui variáveis categóricas (*bootstrap_type*) e contínuas (*learning_rate*). O melhor resultado é obtido com 15 trials e 40 trials de TabNet (94,8%), confirmando que o orçamento de referência é adequado.

O TabNet é o modelo mais sensível ao orçamento de otimização. Sua acurácia parte de 84,3% ($T = 5$, exploração aleatória pura), cai para 81,8% ($T = 15$, com apenas 5 iterações guiadas sobre espaço de $d = 9$), e estabiliza em 82,9% a partir de $T = 40$, permanecendo inalterada em $T = 50$. A queda inicial ($T = 5 \rightarrow 15$) reflete a fase de exploração do TPE em um espaço de alta dimensionalidade, enquanto a estabilização em $T \geq 40$ confirma a convergência. A ausência de degradação em $T = 50$ indica que, com o parâmetro n_d fixado em 8, o risco de *meta-overfitting* é mitigado.

6.6 Considerações finais do capítulo

Este capítulo apresentou o AIMS (*Adaptive and Intelligent Management of Slicing*), um framework híbrido que integra orquestração centralizada na nuvem com inteligência

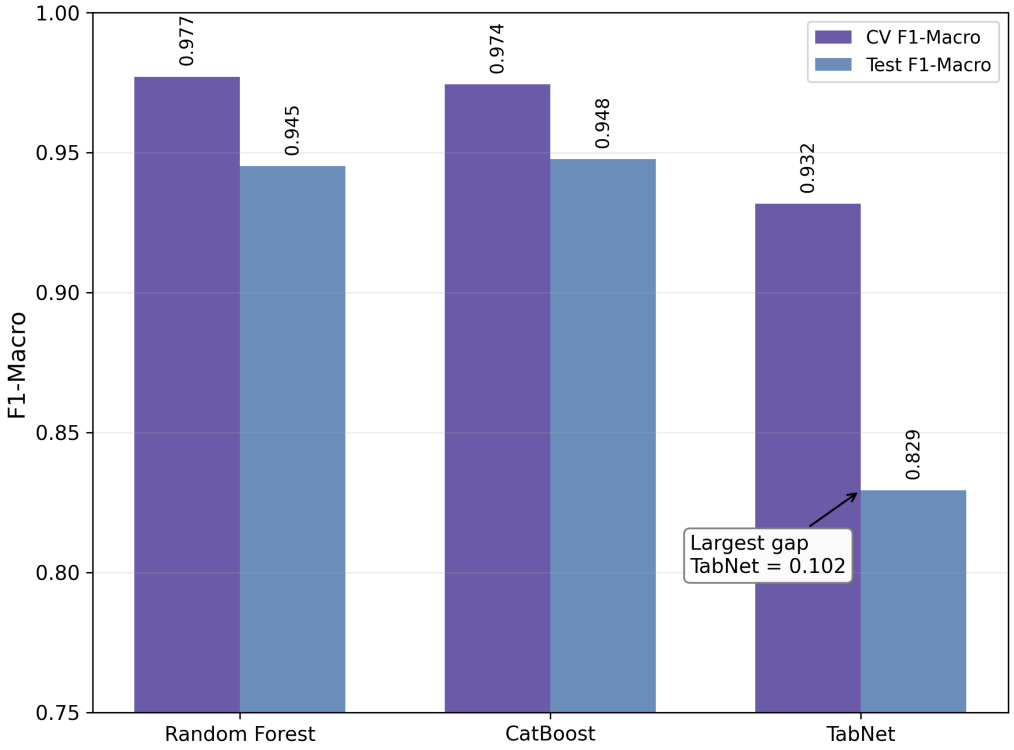


Figura 29 – Comparação entre F1-Macro em validação cruzada (CV) e no conjunto de teste retido (*holdout*). O TabNet apresenta o maior gap de generalização (0,102), evidenciando sensibilidade do modelo à otimização de hiperparâmetros. Fonte: o autor, publicado em [12]

distribuída na borda para governar o ciclo de vida completo de fatias de rede em ambientes veiculares, conforme as quatro fases definidas pelo 3GPP. A contribuição central é a operacionalização da análise de impacto de políticas: um mecanismo que mapeia telemetria multidimensional de tempo de execução em uma taxonomia semântica de quatro níveis (*Adequate*, *Warning*, *Severe*, *Critical*), fornecendo gatilhos estáveis e interpretáveis para adaptação proativa em *closed-loop* e evitando o comportamento de oscilação (*flapping*) associado a regras estáticas baseadas em limiares.

A viabilidade dessa função foi avaliada por meio de um protocolo experimental *leakage-free* sobre um dataset realístico de ITS. O CatBoost alcançou o melhor desempenho geral, com 94,8% de acurácia e 94,8% de macro F1-score, demonstrando capacidade superior na detecção de níveis críticos (F1 Critical = 0,964). O Random Forest obteve acurácia competitiva de 94,4% (F1-Macro 0,945) com estabilidade excepcional em todos os orçamentos de otimização e tempo de treinamento 2,5× menor, posicionando-se como alternativa eficiente para implantação em nós de borda com recursos limitados. O TabNet apresentou fragilidades nos níveis intermediários de impacto (*Warning*: F1 = 0,550; confusão *Warning*→*Severe* de ∼40%) e custo computacional desproporcional (23,3× mais lento que o CatBoost), limitando sua aplicabilidade prática.

Esses resultados validam os princípios de projeto do AIMS e confirmam que modelos

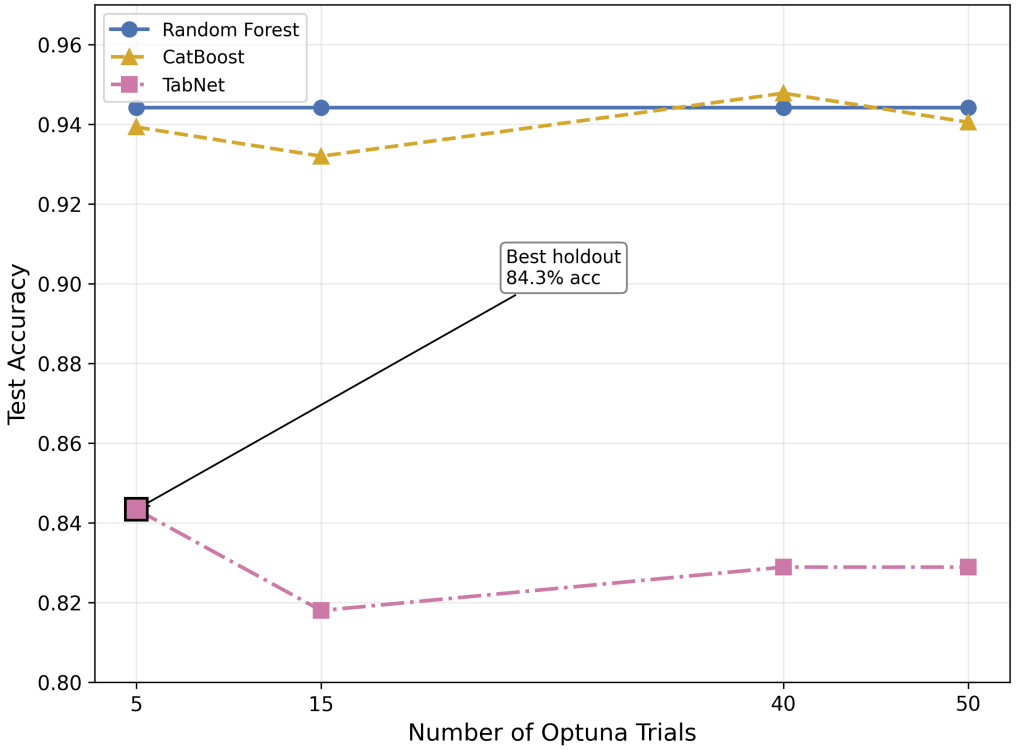


Figura 30 – Estabilidade da acurácia de teste em função do número de avaliações de HPO. O Random Forest mantém 94,4% em todos os orçamentos. O CatBoost varia entre 93,2% e 94,8%, atingindo o melhor resultado em $T = 40$. O TabNet estabiliza em 82,9% a partir de $T = 40$. Fonte: o autor, publicado em [12]

ensemble leves podem traduzir telemetria veicular ruidosa em sinais de impacto acionáveis. Cabe observar que a validação apresentada neste capítulo foca na função de classificação de impacto, e não em uma implementação *closed-loop* completa de ponta a ponta. A orquestração fim-a-fim do ciclo de vida — incluindo a execução automática de ações de adaptação a partir dos gatilhos de impacto — constitui um objetivo para trabalhos futuros, conforme discutido no Capítulo 9.

Uma limitação adicional do modelo centralizado avaliado neste capítulo é a necessidade de concentrar os dados de monitoramento em um único ponto para treinamento. Em implantações reais com múltiplas RSUs distribuídas ao longo de corredores veiculares, essa centralização pode gerar preocupações de privacidade, custo de comunicação e escalabilidade. O aprendizado federado (FL) é uma estratégia promissora para mitigar esse gargalo, embora introduza desafios próprios como heterogeneidade estatística (não-IID) entre clientes, sobrecarga de comunicação de gradientes e custo computacional de treinamento em nós de borda com recursos limitados [145]. O Capítulo 7 investiga essas questões ao propor uma extensão do classificador de impacto com FL, permitindo que os LCs treinem colaborativamente sem compartilhar dados brutos de telemetria.

7 CLASSIFICAÇÃO FEDERADA DE IMPACTO DE FATIAS EM REDES ITS

O framework AIMS, apresentado no Capítulo 6, demonstrou a viabilidade de classificar o impacto de políticas de fatiamento a partir de sinais multidimensionais de monitoramento veicular. Naquela abordagem, modelos como Random Forest e CatBoost alcançaram acurácia superior a 93% na distinção entre quatro níveis de impacto, confirmando que telemetria ruidosa pode ser transformada em sinais acionáveis para o gerenciamento do ciclo de vida das fatias. Entretanto, o treinamento centralizado adotado no AIMS pressupõe que todos os dados de monitoramento sejam transferidos para um único ponto de processamento, uma premissa que se torna impraticável à medida que o número de RSUs e o volume de dados crescem em implantações reais de ITS.

Este capítulo apresenta uma extensão do AIMS que emprega Aprendizado Federado (*Federated Learning* — FL) para tornar escalável a classificação distribuída de impacto de fatias em cenários ITS [13]. Os resultados apresentados neste capítulo utilizam o pipeline unificado do AIMS com a taxonomia de quatro níveis de impacto (Capítulo 6), constituindo uma versão revisada e estendida do manuscrito desenvolvido no contexto desta tese. O classificador de impacto é tratado como uma camada de inteligência transversal ao ciclo de vida das fatias, sendo treinado de forma federada entre controladores locais associados a RSUs e um controlador central na nuvem. Dessa forma, a inteligência coletiva é construída sem a necessidade de centralizar dados brutos, o que reduz o consumo de largura de banda no *backhaul* e preserva a localidade dos dados.

A avaliação experimental compara o treinamento centralizado (linha de base) com duas estratégias federadas, FedAvg [32] e FedProx [33], em cenários com distribuição homogênea (IID) e heterogênea (Non-IID) dos dados, utilizando três arquiteturas de redes neurais: DNN, LSTM e GRU. Os resultados quantificam tanto a viabilidade da abordagem federada quanto os desafios impostos pela heterogeneidade estatística inerente a corredores veiculares distribuídos.

O restante deste capítulo está organizado da seguinte forma. A Seção 7.1 contextualiza a motivação e os trabalhos relacionados à aplicação de FL em fatiamento de rede e redes veiculares. A Seção 7.2 descreve a arquitetura proposta, integrando FL ao gerenciamento do ciclo de vida das fatias. A Seção 7.3 detalha as estratégias de treinamento e as arquiteturas neurais avaliadas. A Seção 7.4 apresenta a configuração experimental, incluindo o particionamento de dados. A Seção 7.5 discute os resultados obtidos. Por fim, a Seção 7.6 sintetiza as contribuições e estabelece a conexão com o capítulo seguinte.

7.1 Motivação e contexto

A natureza dinâmica dos ambientes veiculares impõe desafios ao gerenciamento centralizado de redes [146]. Em sistemas ITS de próxima geração, observações de rede são geradas em localizações de borda distribuídas geograficamente. Centralizar esses dados para treinamento de modelos de aprendizado de máquina cria gargalos de comunicação e compromete a escalabilidade. O fatiamento de rede (*network slicing*) permite a criação de redes lógicas sobre infraestrutura compartilhada [134], gerenciadas ao longo de fases bem definidas: Preparação, Comissionamento, Operação e Descomissionamento [133].Embora a literatura recente evidencie uma franca transição para arquiteturas nativas de Inteligência Artificial integradas fim a fim [147, 148], o gerenciamento que cubra o ciclo de vida completo das fatias sob a alta mobilidade e restrição de recursos de cenários veiculares continua apresentando lacunas. Dessa forma, a orquestração distribuída do ciclo de vida sob as dinâmicas extremas de implantações reais permanece um problema em aberto [147]. Levantamentos recentes confirmam essa lacuna: Lee et al. [149] mapearam aplicações de FL ao gerenciamento de redes móveis 5G e além, constatando que a maioria dos trabalhos se concentra na fase de operação das fatias; Hamdi et al. [150] identificaram que técnicas de aprendizado para fatiamento em redes veiculares priorizam alocação de recursos em detrimento de classificação proativa de impacto. A solução proposta neste capítulo atua diretamente nesta fronteira, contribuindo para a maturação do campo ao fornecer um classificador de impacto que guia o ciclo de vida sem exigir a centralização de dados sensíveis.

Uma lacuna específica reside na ausência de mecanismos escaláveis para gerenciamento proativo de QoS que avaliem o impacto de políticas *antes* que a degradação ocorra. O Aprendizado Federado oferece um paradigma para treinar modelos em dispositivos de borda sem centralizar dados brutos, endereçando simultaneamente questões de privacidade e escalabilidade [151].

Trabalhos recentes exploraram FL para fatiamento e redes veiculares, porém frequentemente abordam aspectos específicos de forma isolada. Roy et al. [152] e Abouaomar et al. [153] propuseram FL para fatiamento na camada de acesso rádio (RAN), com foco em equidade e otimização de recursos, respectivamente, mas sem tratar o gerenciamento fim a fim do ciclo de vida. No domínio veicular, Gupta et al. [154] e Qiang et al. [155] aplicaram FL para inteligência colaborativa V2X e aprendizado dividido (*split learning*), direcionados a tarefas na camada de aplicação e não ao gerenciamento de infraestrutura.

No que se refere à qualidade de serviço, Baganal-Krishna et al. [156] focaram na predição de métricas brutas de QoS, enquanto a abordagem proposta neste capítulo foca na *classificação de impacto* como insumo para tomada de decisão. Tham et al. [157] aplicaram FL para detecção de anomalias em redes veiculares com classificação binária, sem a granularidade necessária para gerenciamento graduado de QoS.

Diferentemente dessas abordagens, o trabalho apresentado neste capítulo integra FL como camada transversal de inteligência para classificação multinível de impacto de políticas, validando-a explicitamente sob as restrições de heterogeneidade de dados típicas de cenários ITS.

7.2 Arquitetura FL para gerenciamento de fatias

Esta seção apresenta a arquitetura proposta para gerenciamento proativo do ciclo de vida das fatias de rede (*network slices*) em cenários ITS, habilitada por Aprendizado Federado. A ideia central consiste em incorporar a *classificação distribuída de impacto* como camada de inteligência transversal: controladores locais em RSUs aprendem a partir de dados de monitoramento locais, enquanto um controlador central na nuvem agrega as atualizações para manter um modelo global.

A Figura 31 sintetiza a arquitetura em dois planos acoplados: (i) a orquestração do ciclo de vida das fatias, cobrindo as fases de Preparação, Comissionamento, Operação e Descomissionamento; e (ii) a infraestrutura de FL, que atualiza continuamente o classificador de impacto sem transferir dados brutos de monitoramento veicular para o núcleo da rede. O modelo resultante fornece suporte consistente à tomada de decisão em todas as fases do ciclo de vida, mantendo-se escalável para implantações distribuídas geograficamente.

7.2.1 FL centralizado para inteligência de borda

O projeto de FL adotado segue o paradigma servidor-cliente, adaptado ao contexto de ITS. Um Controlador Central (CC) na nuvem atua como servidor FL e agrega atualizações de modelo provenientes de múltiplos Controladores Locais (LCs) nas RSUs. Cada LC retém seus dados de monitoramento localmente e treina uma instância do classificador de impacto utilizando observações de nível de fatia e de rádio, como RTT (*Round-Trip Time*), PDR (*Packet Delivery Ratio*) e derivadas temporais que capturam dinâmicas de curto prazo.

Em intervalos predefinidos, os clientes transmitem apenas atualizações de modelo (pesos e gradientes) ao CC, que computa um novo modelo global e o redistribui às RSUs. Esse ciclo constrói inteligência coletiva ao mesmo tempo em que reduz a carga no *backhaul* e preserva a localidade dos dados. Adicionalmente, a topologia de FL centralizado adotada fornece um mecanismo de base para lidar com a heterogeneidade Non-IID naturalmente induzida por corredores veiculares distribuídos espacialmente.

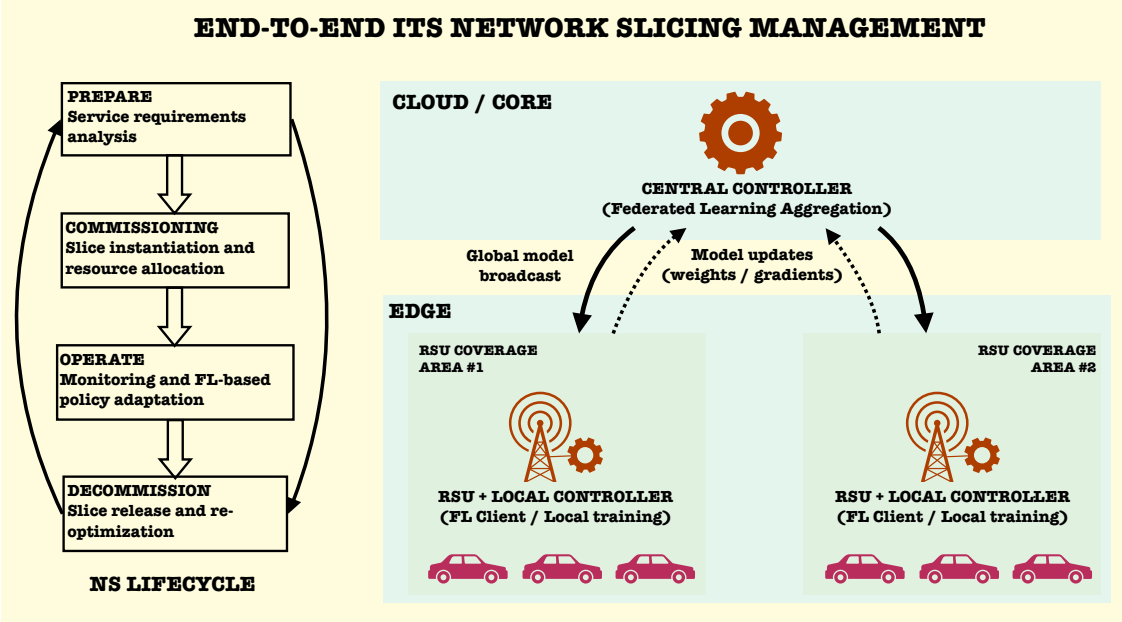


Figura 31 – Gerenciamento fim a fim do ciclo de vida de fatias ITS com suporte de aprendizado federado centralizado. Os controladores locais nas RSUs treinam o classificador de impacto com dados de monitoramento locais e enviam atualizações de modelo ao controlador central na nuvem. O modelo global é redistribuído para decisões orientadas a impacto, sem compartilhamento de dados brutos.

7.2.2 Gerenciamento do ciclo de vida orientado a impacto

A camada de FL é integrada como suporte à decisão ao longo de todo o ciclo de vida das fatias. A orquestração é executada no CC, enquanto o monitoramento e as reações rápidas são realizados nas RSUs.

As quatro fases do ciclo de vida das fatias seguem a estrutura já apresentada no Capítulo 6; esta seção as revisita brevemente sob a perspectiva do treinamento federado, destacando apenas o papel do classificador de impacto distribuído em cada fase, sem reapresentar os detalhes já discutidos.

Na fase de **Preparação**, o CC deriva modelos de fatia (*slice templates*) a partir dos requisitos de serviço; padrões históricos de impacto informam o dimensionamento inicial dos recursos. Na fase de **Comissionamento**, o CC instancia a fatia e aloca recursos; o conhecimento orientado a impacto guia o provisionamento de forma a reduzir configurações inadequadas sob demanda dinâmica. A fase de **Operação** constitui o principal ciclo fechado de controle: as RSUs aplicam o classificador treinado via FL para mapear condições de rede em níveis de impacto; predições de alto impacto disparam adaptações locais ou escalonamento ao CC. Atualizações federadas contínuas mantêm o modelo alinhado com a dinâmica veicular em evolução. Na fase de **Descomissionamento**, o CC coordena a

liberação de recursos; estatísticas agregadas de impacto aprimoram decisões de instanciação subsequentes.

7.2.3 Papel do classificador de impacto

O classificador de impacto é o componente central de inteligência da arquitetura. Em vez de prever métricas brutas de QoS, ele transforma sinais multidimensionais de monitoramento em *níveis de impacto acionáveis* que direcionam respostas de gerenciamento de forma direta. O treinamento é realizado de modo federado entre as RSUs, o que permite ao modelo capturar a variabilidade espaço-temporal sem centralizar dados sensíveis ou de alto volume.

Na prova de conceito apresentada neste capítulo, o classificador produz múltiplos níveis de impacto para suportar ações graduadas: níveis moderados podem disparar ajustes locais de política ou alertas, enquanto níveis críticos demandam intervenção imediata e potencial reconfiguração da fatia. A validação experimental das estratégias de treinamento e das escolhas de modelo é apresentada nas seções a seguir.

Ao transformar métricas brutas em quatro níveis de impacto bem definidos, o framework já fornece uma interface de classificação inteligível e diretamente acionável para a gerência da rede. Contudo, dado que as arquiteturas de redes neurais (como DNN e LSTM) operam fundamentalmente como caixas-pretas, a futura tradução direta das classificações *Severe* ou *Critical* para ações de orquestração totalmente autônomas demandará maior rastreabilidade [158]. Nesses casos, a integração de Inteligência Artificial Explicável (XAI) permitirá extrair justificativas causais sobre quais *features* de telemetria acionaram a degradação, complementando a inteligibilidade de saída do framework com auditoria algorítmica [159].

7.3 Estratégias de treinamento e arquiteturas neurais

Esta seção descreve as três estratégias de treinamento comparadas: centralizado, FedAvg e FedProx, bem como as arquiteturas de redes neurais empregadas como modelos de classificação.

A escolha do FedAvg e do FedProx como algoritmos de agregação justifica-se por sua consolidação como *baselines* industriais canônicos. Embora a literatura recente apresente algoritmos de agregação e seleção hierárquica especificamente otimizados para a dinâmica de Sistemas Inteligentes de Transporte (ITS) [160, 161], os métodos clássicos fornecem uma base de comparação padronizada e incontestável. Sua utilização permite isolar e quantificar de maneira neutra os impactos fundamentais da heterogeneidade de dados (Non-IID) sobre as diferentes arquiteturas neurais, sem introduzir os vieses computacionais de agregadores hiper-especializados.

Embora o Capítulo 6 tenha demonstrado a eficácia de modelos baseados em árvores de decisão (como Random Forest e CatBoost) para o treinamento centralizado, a adaptação desses métodos para o Aprendizado Federado impõe desafios arquiteturais e de comunicação. Diferentemente de redes neurais, os parâmetros estruturais das árvores de decisão (como limiares de particionamento e índices de features) são não-diferenciáveis, o que inviabiliza a agregação matemática direta de modelos locais por meio de algoritmos canônicos como o FedAvg [32, 162]. Além disso, estratégias que suportam fatiamento federado de árvores de decisão frequentemente exigem a troca iterativa de estruturas completas de nós ou de histogramas de features brutas, resultando em elevada sobrecarga de comunicação [163]. Por essa razão, a arquitetura distribuída proposta neste capítulo adota Redes Neurais Artificiais (DNN, LSTM e GRU). A natureza tensorial baseada em gradientes dessas arquiteturas permite sua integração nativa e eficiente com protocolos de agregação padrão, viabilizando a construção do modelo global sob as restrições estritas de largura de banda e recursos computacionais características da borda veicular [151].

7.3.1 Treinamento centralizado (linha de base)

No treinamento centralizado, todos os dados são agregados em uma localização única para aprendizado supervisionado convencional. Essa estratégia serve como limite superior de desempenho alcançável sem restrições de privacidade, além de referência para quantificar o compromisso entre privacidade e desempenho imposto pelo treinamento federado. O modelo é treinado por até 50 épocas com parada antecipada (*early stopping*) baseada na perda de validação, com paciência de 10 épocas, e ponderação de classes (*class weighting*) para compensar o desbalanceamento do *dataset*.

7.3.2 Média federada (FedAvg)

O algoritmo FedAvg [32] constitui a estratégia federada de referência. Cada cliente treina localmente por E épocas e transmite os pesos do modelo ao servidor, que computa uma média ponderada com base no tamanho do conjunto de dados local de cada cliente:

$$w_{t+1} = \sum_{k=1}^{K} \frac{n_k}{n} w_{t+1}^k, \tag{7.1}$$

onde n_k é o número de amostras no cliente k e $n = \sum_k n_k$ é o total de amostras na federação.

7.3.3 Média federada com regularização proximal (FedProx)

O FedProx [33] estende o FedAvg com a adição de um termo proximal que limita a divergência entre os modelos local e global:

Tabela 9 – Especificações das arquiteturas de redes neurais avaliadas.

Especificação	**DNN**	**LSTM**	**GRU**
Dimensão de entrada	25	(1, 25)	(1, 25)
Camadas ocultas	3 Dense	2 LSTM	2 GRU
Tamanho das camadas	128-64-32	64-32	64-32
Função de ativação	ReLU	tanh	tanh
Taxas de *dropout*	0,3 / 0,2	0,3 / 0,2	0,3 / 0,2
Regularização L2	0,001 (1^a/2^a camadas)	–	–
Camada de saída	Softmax (4)	Softmax (4)	Softmax (4)
Parâmetros treináveis	∼13K	∼22K	∼17K
Otimizador	Adam (taxa de aprendizado padrão)		
Função de perda	*Sparse categorical cross-entropy*		
Tamanho do lote	32		

$$h_k(w; w^t) = F_k(w) + \frac{\mu}{2}\|w - w^t\|^2, \tag{7.2}$$

onde μ controla a intensidade da regularização e mitiga o fenômeno de *client drift*, a tendência de modelos locais divergirem excessivamente quando treinados com dados heterogêneos.

Na implementação adotada, o termo de regularização proximal é aplicado como um passo de correção de pesos após cada época local. Essa abordagem constitui uma aproximação prática da função objetivo do FedProx, restringindo efetivamente as atualizações locais em direção ao modelo global. A implementação federada utiliza o *framework* Flower [164], selecionado com base em análise comparativa recente de 15 *frameworks* de FL de código aberto que o posiciona como líder em diversidade de algoritmos, suporte a dispositivos de borda e qualidade de documentação [165]. Cada cliente RSU mantém uma instância local do modelo e treina exclusivamente com sua partição de dados, sem compartilhar amostras brutas.

7.3.4 Arquiteturas de redes neurais

Três arquiteturas de complexidade crescente são avaliadas, conforme sumarizado na Tabela 9. A seleção visa comparar o comportamento do treinamento federado sob diferentes capacidades de modelo — desde uma rede *feedforward* compacta (DNN, ∼13K parâmetros) até arquiteturas com mecanismos de portas (GRU, ∼17K; LSTM, ∼22K) —, avaliando o compromisso entre acurácia, tempo de treinamento e estabilidade de convergência em cenários IID e Non-IID.

A DNN é um modelo *feedforward* com largura decrescente (128→64→32), enquanto LSTM (*Long Short-Term Memory*) e GRU (*Gated Recurrent Unit*) são modelos recorrentes com mecanismos de portas (*gating*). Para as variantes recorrentes, a entrada é reformatada

para o formato tensorial $(1, 25)$. Essa configuração com um único *timestep* implica que os mecanismos de portas do LSTM e do GRU não operam sobre dependências temporais entre amostras consecutivas; a captura de dinâmicas de curto prazo é realizada pelas *features* derivadas presentes no vetor de entrada (taxa de variação do RTT, médias móveis, momentos estatísticos). Nesse contexto, as três arquiteturas funcionam como redes neurais de complexidade crescente, diferindo em capacidade de representação e número de parâmetros treináveis, o que permite avaliar o efeito da complexidade do modelo local sobre o desempenho do treinamento federado. Todos os modelos produzem 4 classes de saída e são treinados com o mesmo otimizador e configuração de lote, assegurando comparabilidade.

7.4 Configuração experimental

7.4.1 Dataset e cenário

Os experimentos utilizam dados derivados do cenário de emulação ITS realístico descrito em [11], que modela um corredor urbano de 650 metros atendido por três RSUs. O cenário captura a dinâmica de comunicação veicular sob variações de congestionamento, com até 17 veículos competindo simultaneamente por recursos de rede.

Os experimentos utilizam o mesmo *dataset* e o mesmo pipeline de pré-processamento e rotulação de impacto empregados no Capítulo 6, assegurando consistência entre as avaliações centralizada e federada. O *dataset* compreende 5.400 amostras de observações de desempenho de fatias em quatro tipos de aplicação ITS: Segurança (*Safety* — S), Eficiência (*Efficiency* — E), Entretenimento (*Entertainment* — E2) e Genérica (*Generic* — G). Cada amostra é representada por 25 *features* derivadas de fluxos de monitoramento, incluindo RTT, PDR e derivadas temporais que capturam dinâmicas de curto prazo (taxa de variação do RTT, médias móveis e momentos estatísticos). As variáveis categóricas (tipo de aplicação, estratégia de fatiamento, perfil de serviço) são codificadas via *one-hot encoding* e as *features* numéricas são normalizadas com *StandardScaler*. A ponderação de classes (*class weighting*) é aplicada durante o treinamento local de cada cliente para compensar o desbalanceamento natural do *dataset*.

Os traços originais foram gerados sob três estratégias de fatiamento: (i) FN (*Flat Network*), sem diferenciação de QoS; (ii) FQ (*Queue-based*), com priorização de tráfego; e (iii) FS (*Full Slicing*), com alocação dedicada de recursos. O alvo de predição é o *nível de impacto* das condições de rede sobre os requisitos de QoS das aplicações, categorizado em quatro classes conforme a taxonomia definida no Capítulo 6: *Adequate* (nível 0 — 768 amostras, 14,2%), *Warning* (nível 1 — 1.013 amostras, 18,8%), *Severe* (nível 2 — 2.715 amostras, 50,3%) e *Critical* (nível 3 — 904 amostras, 16,7%).

Tabela 10 – Matriz de alocação condicional por classe no cenário Non-IID.

Nível de impacto	**RSU 0**	**RSU 1**	**RSU 2**
Adequate (nível 0)	55%	15%	30%
Warning (nível 1)	30%	40%	30%
Severe (nível 2)	10%	40%	50%
Critical (nível 3)	5%	15%	80%

7.4.2 Particionamento de dados: cenários IID e Non-IID

Dois regimes de particionamento são implementados para avaliar a robustez das estratégias federadas.

No cenário **IID** (*Independent and Identically Distributed*), o conjunto de treinamento global (80% do *dataset*, 4.320 amostras) é embaralhado aleatoriamente e dividido em três subconjuntos disjuntos de tamanho aproximadamente igual ($\sim$1.440 amostras por cliente), resultando em proporções de classe similares entre os clientes.

No cenário **Non-IID** (heterogêneo), adota-se uma alocação condicional por classe que induz assimetria de rótulos (*label skew*) entre as RSUs, conforme a Tabela 10. Essa alocação emula a heterogeneidade espaço-temporal observada em implantações reais de ITS, refletindo o acúmulo progressivo de congestionamento ao longo do corredor veicular.

Esse particionamento preserva as proporções globais de classe, ao mesmo tempo em que induz distribuições locais heterogêneas. A RSU 0, em região de entrada do corredor (pré-congestionamento), observa predominantemente condições adequadas (*Adequate* e *Warning*). A RSU 1, em posição intermediária, concentra condições mistas (*Warning* e *Severe*). A RSU 2, na região de maior congestionamento, observa predominantemente degradação significativa (*Severe* e *Critical*, este último com 80% das amostras da classe). Essa configuração captura o efeito de propagação progressiva de congestionamento ao longo do corredor veicular.

7.4.3 Métricas e protocolo de avaliação

O desempenho é avaliado por meio das seguintes métricas: Acurácia, Macro F1-*score*, Rodada de Convergência (primeira rodada a atingir 85% ou 90% de acurácia), Estabilidade (σ — desvio padrão da acurácia nas últimas cinco rodadas) e Tempo Total de Treinamento.

Para o treinamento centralizado, o *dataset* é dividido em 80% para treinamento e 20% para teste, com amostragem estratificada. Para os experimentos federados, o *dataset* é dividido globalmente em 80% para treinamento e 20% para teste, com amostragem estratificada. O conjunto de treinamento global é então particionado entre os clientes conforme o regime adotado (IID ou Non-IID), e cada cliente mantém adicionalmente uma partição local de validação (20% de seus dados) para avaliação distribuída. Após

Tabela 11 – Configuração experimental completa.

Dimensão	**Valores**
Modelos	DNN, LSTM, GRU
Distribuições de dados	IID, Non-IID
Estratégias FL	FedAvg, FedProx
Linha de base	Centralizado (por modelo)
Parâmetros Federados	
Clientes (RSUs)	3
Rodadas de comunicação	30
Épocas locais por rodada	3
Tamanho do lote	32
FedProx μ	0,1
Ponderação de classes	Sim (frequência inversa)
Parâmetros Centralizados	
Épocas de treinamento	50
Divisão de validação	20%
Parada antecipada	Paciência = 10
Configuração de Dados	
Total de amostras	5.400
Features	25
Classes	4 (*Adequate*, *Warning*, *Severe*, *Critical*)
Divisão treino/teste	80%/20% (estratificada)
Infraestrutura	
GPU	NVIDIA L40S
Framework	TensorFlow 2.18 + Flower 1.27
Tempo total de execução	∼40 minutos

cada rodada de comunicação, o modelo global agregado é avaliado no conjunto de teste centralizado, assegurando uma medição imparcial da capacidade de generalização do modelo, independente do viés de distribuição de cada cliente.

7.4.4 Configuração completa dos experimentos

A Tabela 11 sumariza a configuração completa, totalizando **15 experimentos**: 12 federados (3 modelos × 2 distribuições × 2 estratégias) mais 3 linhas de base centralizadas. Todos os geradores de números aleatórios foram inicializados deterministicamente (*seed* = 42, com sementes específicas por cliente: 42 + *client_id*) para garantir reprodutibilidade. O código-fonte e os artefatos experimentais estão disponíveis publicamente[1].

Diferentemente da avaliação centralizada do Capítulo 6, que emprega otimização Bayesiana de hiperparâmetros (HPO) com Optuna, os experimentos federados utilizam uma configuração fixa de hiperparâmetros, consistente com a prática predominante na literatura de FL [32, 33]. Essa escolha é motivada pelo custo computacional proibitivo da

[1] https://github.com/saraivacode/its-fl-slice-impact

HPO em cenário federado — cada avaliação de hiperparâmetros requer uma simulação completa de R rodadas de comunicação entre K clientes — e pela representatividade operacional: em implantações reais, os hiperparâmetros são definidos *a priori* pelo operador da rede. A adequação da configuração adotada é validada pelos *baselines* centralizados, cujo desempenho (GRU: F1-Macro = 0,9542) é comparável ao do Random Forest otimizado com Optuna no Capítulo 6 (F1-Macro = 0,9451), indicando que a arquitetura e os hiperparâmetros escolhidos são competitivos mesmo sem busca automatizada.

7.5 Resultados e discussão

7.5.1 Desempenho em cenários IID

Em cenários IID, todos os modelos federados atingiram acurácia superior a 92%, com Macro F1-*score* entre 0,9272 e 0,9376. O DNN com FedProx obteve o melhor F1-*score* IID (0,9376), empatado com o DNN-FedAvg em cenário Non-IID (Tabela 12). O LSTM com FedProx alcançou F1 de 0,9348, enquanto o GRU com FedAvg obteve 0,9284. Esses resultados representam uma degradação de 0,57 a 2,58 pontos percentuais em relação às linhas de base centralizadas (GRU centralizado: F1 = 0,9542), confirmando que o treinamento distribuído preserva a capacidade de classificação com quatro classes desbalanceadas.

A convergência foi rápida: todas as configurações IID atingiram o limiar de 85% de acurácia entre a segunda e a terceira rodada de comunicação. O limiar de 90% foi alcançado entre as rodadas 5 e 10, dependendo da arquitetura e estratégia. Essa eficiência minimiza a sobrecarga de comunicação, aspecto relevante para implantações ITS com largura de banda limitada no *backhaul*.

7.5.2 Efeito da heterogeneidade estatística (Non-IID)

A transição de cenários IID para Non-IID não introduziu degradação significativa de desempenho. O melhor resultado Non-IID (DNN-FedAvg: F1 = 0,9376) é comparável ao melhor resultado IID (DNN-FedProx: F1 = 0,9376), com diferença média inferior a 0,5 ponto percentual no F1-*score* entre os dois regimes de distribuição ao longo de todas as configurações (Figura 32).

Esse resultado indica que, com a aplicação de ponderação de classes durante o treinamento local e avaliação centralizada do modelo global, a heterogeneidade induzida pela distribuição condicional por classe entre as RSUs não compromete a capacidade de generalização do modelo agregado. A ponderação de classes compensa o desbalanceamento natural do *dataset* (*Severe* ∼50%), mitigando o viés que cada cliente teria ao treinar com sua distribuição local enviesada. O caso notável do DNN-FedAvg, que obtém seu melhor

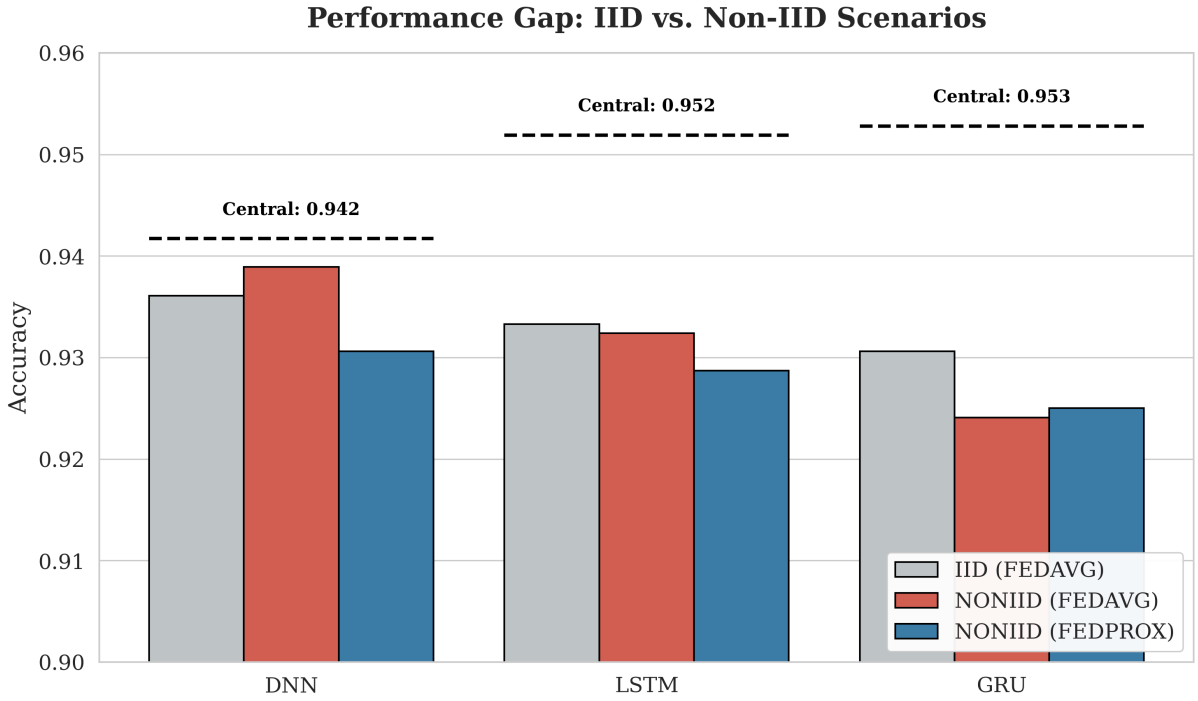


Figura 32 – Comparação entre cenários IID e Non-IID. Linhas tracejadas representam as linhas de base centralizadas. A degradação Non-IID é inferior a 1 ponto percentual para todas as arquiteturas.

resultado justamente em cenário Non-IID (F1 = 0,9376, acurácia 93,89%), sugere que a diversidade de padrões presentes em dados heterogêneos pode beneficiar a agregação para arquiteturas mais simples.

7.5.3 FedAvg versus FedProx em cenários Non-IID

A Tabela 12 mostra que nem FedAvg nem FedProx apresentam vantagem consistente, com diferenças dentro da faixa de ±1,14 ponto percentual ao longo de todas as configurações. O FedProx supera marginalmente o FedAvg em 4 das 6 comparações, porém as margens são insuficientes para constituir vantagem estatisticamente relevante.

No cenário Non-IID, o FedProx demonstra comportamento variável por arquitetura. Para o DNN, o FedAvg supera o FedProx em 1,14 p.p. (F1 0,9376 contra 0,9262). Para LSTM e GRU, as diferenças são desprezíveis (≤0,07 p.p.). Esses resultados indicam que, com apenas 3 clientes e heterogeneidade moderada, a regularização proximal com $\mu = 0,1$ não provê benefício consistente, embora tampouco degrade o desempenho, confirmando-se como uma alternativa metodologicamente válida, embora sem vantagem empírica clara neste cenário. Essa observação é corroborada por estudos comparativos recentes: Baumgart et al. [166] avaliaram seis estratégias de agregação e reportaram que o FedProx apresenta sobrecarga de tempo significativa em relação ao FedAvg sem ganho proporcional de acurácia; Li et al. [167] demonstraram, em estudo experimental com múltiplos cenários

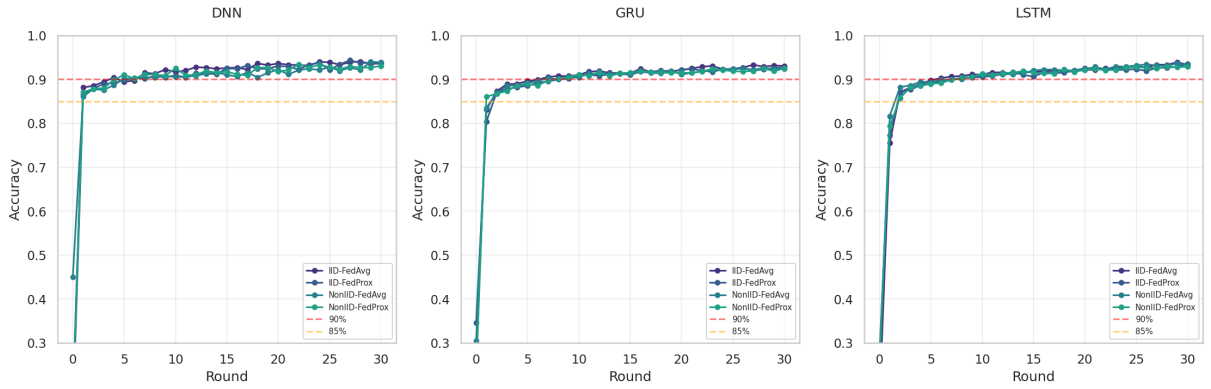


Figura 33 – Curvas de convergência para DNN, GRU e LSTM ao longo de 30 rodadas de comunicação. As quatro configurações (IID/Non-IID × FedAvg/FedProx) convergem de forma similar em cada arquitetura, com todas atingindo o limiar de 85% entre as rodadas 2 e 3, e 90% entre as rodadas 5 e 10. O DNN apresenta a convergência mais rápida e estável.

de heterogeneidade, que a melhoria do FedProx sobre o FedAvg pode ser inferior a 1 ponto percentual sob *label skew* moderado — consistente com os resultados obtidos neste trabalho.

7.5.4 Convergência e seleção de modelo

A Figura 33 ilustra o comportamento de convergência. Todos os modelos atingem o limiar de 85% de acurácia entre as rodadas 2 e 3, tanto em cenários IID quanto Non-IID. O patamar de 90% é alcançado entre as rodadas 5 e 10, com o DNN-FedAvg apresentando a convergência mais rápida (90% na rodada 5 em cenário IID).

Em relação à seleção de arquitetura, o DNN apresenta o melhor compromisso entre desempenho e eficiência no contexto federado: maior F1-*score* (0,9376) com tempos de treinamento de 180–217s, inferiores aos do LSTM e GRU que não superam o DNN em desempenho federado apesar de possuírem mais parâmetros treináveis. Essa inversão em relação ao cenário centralizado — onde GRU (F1 = 0,9542) e LSTM (F1 = 0,9521) superam o DNN (F1 = 0,9433) — é um achado relevante: arquiteturas mais simples, sem conexões recorrentes, se agregam melhor no FedAvg porque a média ponderada de pesos de *gates* recorrentes de clientes com dados heterogêneos pode cancelar informação aprendida localmente.

A estabilidade, medida pelo desvio padrão da acurácia nas últimas cinco rodadas, é satisfatória em todas as configurações ($\sigma < 0,009$). O GRU-FedAvg-NonIID apresenta a maior estabilidade ($\sigma = 0,0012$), enquanto o DNN-FedAvg-NonIID exibe a maior variação ($\sigma = 0,0086$), ainda assim dentro de limites aceitáveis para aplicações práticas.

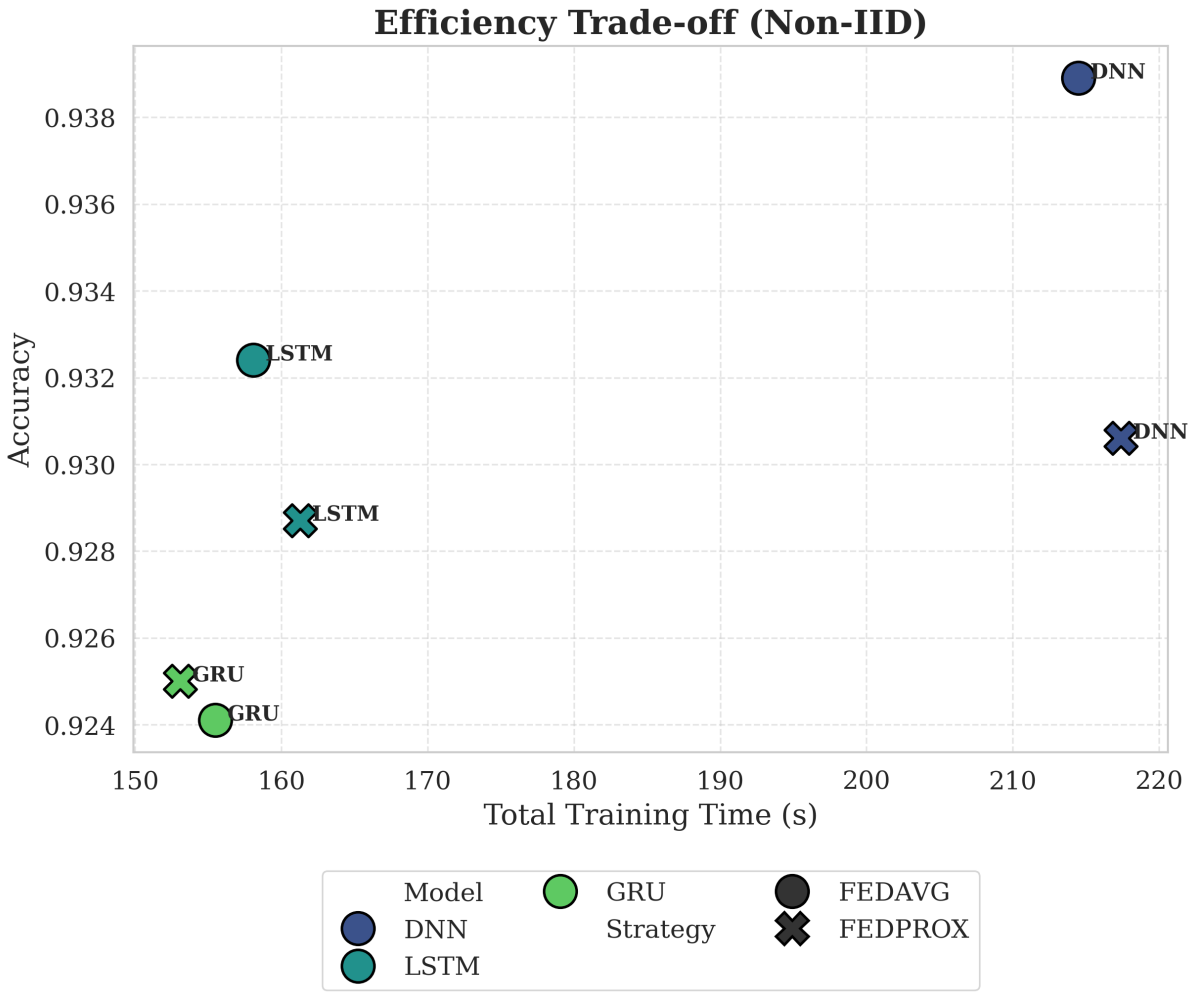


Figura 34 – Compromisso entre eficiência e acurácia. O DNN oferece o melhor equilíbrio entre desempenho e tempo de treinamento no cenário federado.

7.5.5 Resultados completos

A Tabela 12 consolida os resultados de todos os 15 experimentos realizados, abrangendo desempenho, convergência e métricas de eficiência.

Os resultados apresentados na Tabela 12 permitem identificar três constatações principais. Em primeiro lugar, o treinamento federado com quatro classes desbalanceadas preserva desempenho comparável ao centralizado, com degradação máxima de 2,58 pontos percentuais no F1-*score* (GRU), confirmando a viabilidade da abordagem distribuída mesmo com distribuição de classes acentuadamente desigual. Em segundo lugar, a transição de cenários IID para Non-IID não impõe degradação significativa (diferença média inferior a 0,5 ponto percentual em F1-*score*), indicando que a combinação de ponderação de classes e avaliação centralizada do modelo global confere robustez à heterogeneidade induzida. Em terceiro lugar, o DNN apresenta o melhor compromisso entre desempenho e eficiência no contexto federado, superando LSTM e GRU tanto em F1-*score* quanto em robustez à distribuição de dados — uma inversão em relação ao cenário centralizado, onde as

Tabela 12 – Resultados experimentais completos: métricas de desempenho, convergência e eficiência.

Modelo	Distribuição	Estratégia	Acurácia	F1-*Score*	@85%	@90%	Estab. (σ)	Tempo
DNN	IID	FedAvg	0,9361	0,9338	R2	R5	0,0026	179,9s
	IID	FedProx	0,9370	0,9376	R2	R7	0,0030	180,4s
	Non-IID	FedAvg	**0,9389**	**0,9376**	R2	R6	0,0086	214,5s
	Non-IID	FedProx	0,9306	0,9262	R2	R6	0,0026	217,4s
	Centralizado		0,9417	0,9433	–	–	–	25,5s
LSTM	IID	FedAvg	0,9333	0,9330	R3	R7	0,0038	145,0s
	IID	FedProx	**0,9352**	**0,9348**	R3	R9	0,0051	152,5s
	Non-IID	FedAvg	0,9324	0,9277	R3	R9	0,0018	158,1s
	Non-IID	FedProx	0,9287	0,9281	R3	R9	0,0016	161,3s
	Centralizado		0,9519	0,9521	–	–	–	50,7s
GRU	IID	FedAvg	**0,9306**	**0,9284**	R3	R8	0,0022	142,3s
	IID	FedProx	0,9269	0,9272	R3	R10	0,0028	141,9s
	Non-IID	FedAvg	0,9241	0,9238	R3	R8	0,0012	155,5s
	Non-IID	FedProx	0,9250	0,9245	R2	R9	0,0032	153,1s
	Centralizado		0,9528	0,9542	–	–	–	53,9s

@85%/@90%: Primeira rodada a atingir o limiar. Rn = Rodada n. Negrito: melhor resultado federado por modelo. Estabilidade (σ): desvio padrão da acurácia nas últimas 5 rodadas. Menor é melhor.

arquiteturas recorrentes dominam.

A classe *Warning* permanece como principal desafio, com F1-*score* entre 0,83 e 0,86 nos cenários federados (contra 0,87–0,89 nos centralizados e 0,88–0,89 nos modelos *ensemble* do Capítulo 6). A classe *Critical* é consistentemente bem classificada (F1 > 0,98 em todas as configurações federadas), o que é relevante para aplicações de segurança veicular onde a identificação de condições críticas é prioritária.

7.6 Considerações finais

Este capítulo apresentou uma arquitetura habilitada por Aprendizado Federado que integra classificação distribuída de impacto como suporte à tomada de decisão no ciclo de vida de fatias de rede em cenários ITS. O classificador — componente central proposto no framework AIMS (Capítulo 6) — foi validado em sua versão federada utilizando o pipeline unificado do AIMS com a taxonomia de quatro níveis de impacto, comparando treinamento centralizado com FedAvg e FedProx em cenários IID e Non-IID com três arquiteturas neurais (DNN, LSTM e GRU).

Os resultados demonstram que o treinamento federado mantém desempenho comparável ao centralizado, com degradação máxima de 2,58 pontos percentuais no F1-*score* (GRU centralizado: 0,9542; melhor GRU federado: 0,9284). O DNN apresentou o menor *gap* (−0,57 p.p.), alcançando F1 de 0,9376 tanto com FedAvg em cenário Non-IID quanto com FedProx em cenário IID. A transição de cenários IID para Non-IID não impôs degradação significativa (diferença média inferior a 0,5 p.p.), indicando que a ponderação de classes e a avaliação centralizada do modelo global conferem robustez à heterogeneidade induzida pela distribuição espacial das RSUs. A convergência rápida (85% de acurácia entre as rodadas 2 e 3 em todas as configurações) confirma a viabilidade para implantações com restrição de comunicação no *backhaul*.

O DNN apresentou o melhor compromisso entre desempenho e eficiência no contexto federado — uma inversão em relação ao cenário centralizado, onde GRU e LSTM dominam. Essa observação sugere que, no cenário estudado, arquiteturas mais simples podem se agregar de forma mais eficaz no FedAvg do que arquiteturas recorrentes mais complexas, hipótese que merece investigação específica em estudos futuros. O FedProx ($\mu = 0,1$) não demonstrou vantagem consistente sobre o FedAvg ao longo das 12 configurações avaliadas, indicando que a regularização proximal padrão não provê benefício significativo com apenas 3 clientes e heterogeneidade moderada. A classe *Warning* permanece como fronteira mais difícil para a classificação federada, com *gap* de 2–3 p.p. em relação ao centralizado — motivando investigações futuras em estratégias conscientes de heterogeneidade — como aprendizado federado personalizado [168] e agregação hierárquica [169] — avaliadas em implantações de maior escala. A emergência de arquiteturas baseadas em *transformers* para FL em redes veiculares [170] constitui direção complementar, potencialmente capaz de capturar dependências temporais mais complexas do que as arquiteturas recorrentes avaliadas neste trabalho.

Adicionalmente, planeja-se a integração do classificador com sistemas de orquestração de fatias para validação fim a fim do ciclo de vida. Embora a taxonomia de impacto do modelo já produza uma saída de alta inteligibilidade operacional, a incorporação de métodos *post-hoc* de Inteligência Artificial Explicável (XAI) constitui um vetor promissor de evolução. A aplicação de XAI no ambiente federado poderá fornecer transparência causal às inferências das redes neurais, sendo um requisito fundamental para a adoção confiável (*Trustworthy AI*) de orquestração autônoma na era do 6G [158].

A adoção de FL como camada transversal de inteligência, embora resolva questões de escalabilidade e privacidade, introduz novas superfícies de ataque que o modelo centralizado não apresentava. A distribuição do treinamento entre múltiplos clientes federados abre possibilidades para ataques de envenenamento de modelo, inferência de dados a partir de atualizações de gradientes e comprometimento de clientes individuais. Essas vulnerabilidades, específicas do contexto de fatiamento inteligente para ITS, motivam a investigação apresentada no Capítulo 8, que aborda a segurança da informação neste cenário.

8 SEGURANÇA DA INFORMAÇÃO EM REDES ITS COM FATIAMENTO INTELIGENTE

Os capítulos anteriores apresentaram a construção progressiva de um ecossistema de fatiamento inteligente para redes ITS, desde a identificação do contexto veicular (Capítulos 3 e 4), passando pelo fatiamento orientado a aplicações (Capítulo 5), até o gerenciamento inteligente com classificação de impacto (Capítulo 6) e sua extensão distribuída via Aprendizado Federado (Capítulo 7). A introdução de componentes de aprendizado de máquina distribuídos no gerenciamento das fatias amplia as capacidades do sistema, mas também introduz superfícies de ataque que precisam ser investigadas.

Este capítulo aborda a segurança da classificação federada de impacto no contexto de fatiamento inteligente em redes ITS. Arquiteturas como o AIMS, com componentes de FL distribuídos entre controladores locais e central, estão sujeitas a ameaças que vão além das tradicionais em redes de comunicação, incluindo ataques de envenenamento de modelo (*model poisoning*) e manipulação de dados de monitoramento que podem comprometer a integridade das classificações de impacto que orientam decisões no ciclo de vida das fatias.

A investigação é enquadrada como diagnóstico de vulnerabilidade operacional: o objetivo é determinar em que condições ataques de envenenamento comprometem o classificador federado de impacto, quantificar o risco operacional resultante, e avaliar a eficácia de estratégias de agregação robusta como mecanismo de mitigação.

8.1 Estado da Arte em Segurança para Fatiamento Inteligente em ITS

O Capítulo 7 demonstrou que a classificação federada de impacto de fatias é viável em cenários ITS: o treinamento distribuído entre RSUs preserva desempenho comparável ao centralizado, com degradação inferior a 2,6 pontos percentuais em F1-*score* e convergência alcançada em poucas rodadas de comunicação. Essa constatação, no entanto, se sustenta sob o pressuposto de que todos os participantes do sistema são honestos. A distribuição do treinamento entre clientes federados introduz superfícies de ataque ausentes no modelo centralizado do AIMS (Capítulo 6): um controlador local comprometido pode enviar atualizações de modelo adulteradas ao servidor de agregação, manipulando a classificação de impacto que orienta decisões no ciclo de vida das fatias.

Esta seção revisa o estado da arte na interseção entre segurança de aprendizado federado, redes veiculares e fatiamento de rede, identificando a lacuna que motiva a investigação deste capítulo.

8.1.1 Segurança de FL em redes veiculares

A pesquisa em segurança de FL aplicado a redes veiculares convergiu, entre 2024 e 2026, para quatro eixos: robustez contra ataques de envenenamento, preservação de privacidade, gestão descentralizada de confiança e detecção de intrusão federada [151].

No eixo de robustez, a tendência recente é a substituição de regras estatísticas estáticas por mecanismos adaptativos baseados em aprendizado por reforço. Pan et al. (IEEE TVT, 2025) propuseram o DRL-PBFL, em que agentes DDPG aprendem políticas de confiança para filtrar participantes bizantinos sem acesso aos dados em texto claro [171]. He et al. (IEEE TMC, 2026) combinaram privacidade diferencial local condensada com criptografia homomórfica empacotada e filtragem por Trimmed k-means, operando sob heterogeneidade severa (Dirichlet $\alpha = 0{,}1$, até 40% de atacantes) [172]. No domínio veicular especificamente, Sameera et al. [173] propuseram o LFGuard, um mecanismo de defesa contra *label-flipping* baseado em SVMs multi-classe sobre ativações da última camada, com ganho superior a 5% em acurácia global.

No eixo de privacidade, os protocolos evoluíram de técnicas isoladas para composições híbridas. Nguyen et al. (VTC-Fall, 2024) integraram criptografia homomórfica via esquema CKKS diretamente no pipeline FL, demonstrando viabilidade com degradação inferior a 1% [174]. Uma contribuição sistematizadora relevante é a de Du et al. (USENIX Security, 2025), que analisaram os fatores que afetam ataques de inversão de gradiente e concluíram que tais ataques são mais frágeis em condições práticas do que a literatura anterior indicava — particularmente com múltiplas épocas locais e *batch sizes* realistas [175].

No eixo de confiança descentralizada, a integração blockchain-FL amadureceu em direção a mecanismos de consenso específicos para IoV. Abou El Houda et al. (IEEE T-ITS, 2024) — o trabalho com maior convergência no levantamento, confirmado por quatro fontes independentes — combinam FL com blockchain e DRL para detecção de intrusão colaborativa em *vehicular edge computing* [176].

Na detecção de intrusão federada, que constitui a aplicação mais madura de FL seguro em redes veiculares, Althunayyan et al. (Vehicular Communications, 2024) propuseram um IDS em dois estágios com FL hierárquico, alcançando F1 superior a 0,99 para ataques conhecidos em barramento CAN [177]. O framework FedFT (2025) demonstrou que pré-treinamento centralizado seguido de *fine-tuning* leve nos clientes é viável em hardware restrito (Raspberry Pi 4), com redução de 42% no uso de memória

[178].

Apesar dos avanços, três problemas persistem no campo. A crise de reprodutibilidade é severa: menos de 15% dos trabalhos identificados no levantamento disponibilizam código-fonte, e a maioria utiliza *datasets* genéricos (MNIST, CIFAR-10) em vez de *benchmarks* veiculares. A heterogeneidade estatística natural e a alta mobilidade em ambientes veiculares produzem comportamentos que mimetizam ataques, dificultando a separação entre *drift* legítimo e envenenamento. E o *overhead* computacional das técnicas de segurança (criptografia homomórfica, blockchain, MPC) introduz latência frequentemente incompatível com os requisitos de fatias URLLC.

8.1.2 Ataques de envenenamento em aprendizado federado

Os ataques de envenenamento (*poisoning attacks*) exploram a natureza distribuída do FL para comprometer o modelo global a partir de um ou mais participantes maliciosos. A literatura distingue duas superfícies: envenenamento de dados (*data poisoning*), em que o atacante manipula o conjunto de treino local, e envenenamento de modelo (*model poisoning*), em que o atacante manipula os pesos antes de enviá-los ao servidor [179].

No envenenamento de dados, o *label-flipping* é o ataque mais estudado: o cliente malicioso altera os rótulos de amostras selecionadas, fazendo com que o modelo local aprenda associações invertidas entre features e classes. Tolpegin et al. [180] demonstraram a eficácia desse ataque em cenários federados, e Sameera et al. [173] estenderam a análise para redes veiculares com o mecanismo de defesa LFGuard. No contexto de predição de trajetória em condução autônoma, Garg et al. (SAIS, 2024) demonstraram que ataques direcionados contornam defesas clássicas como Krum, Multi-Krum e Trimmed Mean quando aplicados a tarefas de regressão [181], ampliando a superfície de ameaça para além da classificação.

No envenenamento de modelo, o atacante amplifica a influência de sua atualização por meio de técnicas como *gradient scaling* [182] ou otimização adversarial direcionada [179]. Fang et al. [179] demonstraram que ataques adaptativos, que calibram a atualização maliciosa para contornar a defesa em operação, podem comprometer mesmo estratégias Bizantino-robustas. Shejwalkar e Houmansadr [182] estenderam essa análise mostrando que ataques ótimos requerem calibração do fator de amplificação — amplificação excessiva pode ser menos eficaz por produzir atualizações facilmente detectáveis como *outliers*.

8.1.3 Agregação Bizantino-robusta

A defesa no lado do servidor (*server-side*) contra atualizações maliciosas se dá por meio de estratégias de agregação que substituem a média ponderada simples do FedAvg [32] por mecanismos de filtragem ou aparamento.

O Krum [183] seleciona a atualização mais próxima da maioria dos participantes, rejeitando *outliers* no espaço de parâmetros. Sua eficácia depende de satisfazer a condição $n \geq 2f+1$, onde n é o número de clientes e f o número de maliciosos. A Trimmed Mean [184] opera coordenada a coordenada, aparando os valores extremos antes de calcular a média, o que garante que em cada dimensão do modelo o valor selecionado pertence a um cliente honesto.

Essas defesas clássicas, no entanto, apresentam limitações conhecidas. Fang et al. [179] demonstraram que ataques adaptativos — que otimizam a atualização maliciosa para minimizar a distância aos clientes honestos — podem comprometer tanto Krum quanto Trimmed Mean. Além disso, em regimes com poucos participantes ($n = 3$), a margem de operação dessas defesas é mínima: o Krum descarta dois dos três modelos por rodada, e a Trimmed Mean opera como mediana coordenada. O FedProx [185], embora projetado para mitigar o *client drift* em cenários heterogêneos, não incorpora mecanismo de filtragem de atualizações adversariais e não pode ser considerado defesa contra envenenamento.

8.1.4 Lacuna: segurança do FL no contexto de fatiamento de rede para ITS

O levantamento bibliográfico conduzido para esta pesquisa, abrangendo 27 trabalhos em *venues* de alto impacto publicados entre 2024 e 2026, identifica uma lacuna na interseção entre três domínios: segurança de FL, fatiamento de rede e sistemas inteligentes de transporte.

De Alwis et al. [134] apresentam um levantamento abrangente sobre segurança em fatiamento de rede 5G/B5G, cobrindo ataques a cada fase do ciclo de vida, mas não abordam ameaças específicas ao aprendizado federado como camada de inteligência no gerenciamento das fatias. Chellapandi et al. [186] mapeiam aplicações de FL em veículos conectados e autônomos, incluindo aspectos de segurança e privacidade, porém sem tratar o fatiamento de rede como contexto operacional. Trabalhos como os de Pan et al. (DRL-PBFL), He et al. (PPBR) e Sameera et al. [173] avaliam defesas contra envenenamento em FL veicular, mas utilizam tarefas genéricas de classificação sem conexão com o gerenciamento de fatias.

O único trabalho identificado que combina FL, segurança e fatiamento de rede para V2X é o de Boualouache et al. [187], que propõem uma arquitetura FL multi-processo com *differential privacy* para proteger fatias 6G-V2X em cenários transfronteiriços, avaliada com o dataset 5G-NIDD. Esse trabalho, contudo, foca em detecção de intrusão e não aborda a vulnerabilidade do classificador de impacto que orienta as decisões de ciclo de vida — a função central do AIMS.

Nenhum trabalho existente avalia como ataques de envenenamento podem comprometer especificamente a classificação de impacto de políticas de fatiamento em cenários ITS. Essa lacuna é particularmente relevante porque, no contexto do AIMS, a classificação

errônea de uma condição *Critical* como *Adequate* resulta na omissão de ações de ciclo de vida para fatias que atendem aplicações *safety-critical* — um risco operacional que não é capturado por avaliações genéricas de segurança em FL.

8.1.5 Pergunta de pesquisa e contribuições

Com base na lacuna identificada, este capítulo investiga a seguinte pergunta: *Qual é a vulnerabilidade do classificador federado de impacto de fatias ITS frente a ataques de envenenamento direcionado, e em que medida estratégias de agregação Bizantino-robusta podem mitigar esses ataques em cenários com poucos participantes?*

A investigação é enquadrada como diagnóstico de vulnerabilidade operacional — não como proposta de solução completa de segurança. As contribuições específicas deste capítulo são:

1. A avaliação da vulnerabilidade do classificador federado de impacto do AIMS sob ataques de *label-flipping* direcionado (*Critical*→*Adequate*) com *gradient scaling*, quantificando o impacto operacional por meio da métrica C2A Rate (proporção de decisões de ciclo de vida omitidas) em 144 configurações experimentais que combinam três arquiteturas neurais (DNN, LSTM, GRU), seis estratégias de agregação e múltiplos fatores de amplificação.
2. A demonstração de que a vulnerabilidade é estrutural — manifesta-se em todas as arquiteturas avaliadas — e de que o FedProx não constitui mecanismo de defesa, podendo amplificar a vulnerabilidade em cenários com heterogeneidade estatística.
3. A avaliação exploratória de defesas Bizantino-robustas (Krum e Trimmed Mean) no regime $n = 3$, $f = 1$, documentando sua eficácia contra ataques não adaptativos e identificando os limites inerentes ao regime de poucos participantes.

As seções seguintes formalizam o modelo de ameaça (Seção 8.2), descrevem a metodologia experimental (Seção 8.3), e apresentam os resultados e a discussão (Seção 8.4).

8.2 Modelo de Ameaça e Formulação do Problema

Os capítulos anteriores demonstraram a viabilidade da classificação de impacto de políticas de fatiamento tanto em modo centralizado (Capítulo 6, framework AIMS [12] quanto em modo federado (Capítulo 7 [13]). Em ambos os casos, o pressuposto implícito é de que todos os participantes do sistema são honestos — isto é, que os controladores locais nas RSUs executam o treinamento sobre dados legítimos e reportam atualizações de modelo fidedignas ao servidor de agregação. Em implantações ITS reais, esse pressuposto é insustentável. Uma RSU comprometida pode enviar atualizações de modelo adulteradas ao servidor central com o objetivo de manipular o classificador global de impacto de fatias.

Esta seção formaliza o modelo de ameaça, define os ataques avaliados e apresenta a métrica operacional utilizada para quantificar o impacto adversarial sobre as decisões de ciclo de vida das fatias.

8.2.1 Cenário adversarial

O cenário adversarial situa-se na fase de *Operation* do ciclo de vida de fatias 3GPP [133]. Nessa fase, os controladores locais do AIMS coletam telemetria de rede, treinam modelos locais de classificação de impacto e enviam as atualizações de pesos ao controlador central para agregação federada. O modelo global resultante é redistribuído aos controladores locais, que o utilizam para classificar o impacto corrente das políticas de fatiamento e acionar ações de ciclo de vida.

Um participante adversarial busca manipular o modelo global para suprimir a detecção de condições críticas em fatias de rede. O objetivo específico do atacante é induzir o classificador global a rotular amostras de impacto *Critical* (violação de SLA iminente ou em curso) como *Adequate* (operação dentro dos parâmetros). Essa direção de ataque — *Critical*→*Adequate* (C→A) — é a mais perigosa operacionalmente: implica omissão de ações de ciclo de vida em fatias que atendem aplicações *safety-critical* em cenários URLLC.

Na topologia experimental, o sistema federado opera com 3 clientes correspondentes às 3 RSUs do corredor urbano descrito no Capítulo 5 [11]. O atacante controla o *Client* 0 (RSU 0), cuja composição no particionamento Non-IID é predominantemente de amostras *Adequate* (55%), com apenas 5% de amostras *Critical*. Essa configuração representa um cenário relevante do ponto de vista prático: o atacante opera a partir de uma RSU com acesso limitado à classe-alvo, demonstrando que mesmo um participante com poucos exemplos da classe crítica pode comprometer o modelo global quando combina *label-flipping* com amplificação de gradientes.

8.2.2 Capacidades do atacante

A Tabela 13 resume as capacidades assumidas para o adversário, seguindo a formulação de Fang et al. [179] para ataques não adaptativos.

O atacante opera de forma individual e não adaptativa: não otimiza sua estratégia contra uma defesa específica e não possui informação sobre as atualizações dos demais clientes.

8.2.3 Ataque de *label-flipping* (envenenamento de dados)

O *label-flipping* é um ataque de envenenamento de dados [180] em que o cliente malicioso altera os rótulos de seu conjunto de treino local antes do treinamento. No contexto deste trabalho, o ataque é direcionado: todas as amostras com rótulo *Critical*

Tabela 13 – Capacidades assumidas para o adversário.

Capacidade	**Assumida**
Controla 1 dos 3 clientes (*Client* 0, RSU 0)	Sim
Modifica rótulos dos dados de treino locais (*label-flipping*)	Sim
Modifica pesos do modelo antes de enviar ao servidor (*gradient scaling*)	Sim
Conhece o algoritmo de agregação do servidor	Não
Recebe o modelo global a cada rodada de comunicação	Sim
Possui acesso aos dados de outros clientes	Não
Coordena ataques com outros participantes maliciosos	Não

(classe 3) são reclassificadas como *Adequate* (classe 0). Formalmente, dado o conjunto de treino local $D_k = \{(x_i, y_i)\}$ do cliente comprometido k:

$$y_i' = \begin{cases} 0, & \text{se } y_i = 3 \quad (\textit{Critical} \rightarrow \textit{Adequate}) \\ y_i, & \text{caso contrário} \end{cases} \tag{8.1}$$

O modelo local é então treinado sobre $D_k' = \{(x_i, y_i')\}$, aprendendo a associar padrões de telemetria correspondentes a condições críticas ao rótulo *Adequate*, e propagando essa associação ao modelo global durante a agregação.

A justificativa para a direção C→A decorre do significado operacional das classes na taxonomia do AIMS. A classe *Critical* corresponde a condição que requer ação urgente de ciclo de vida. Sua classificação errônea como *Adequate* faz com que o sistema de gerenciamento não tome nenhuma ação enquanto a fatia opera em violação de SLA.

8.2.4 Ataque de *gradient scaling* (envenenamento de modelo)

O *gradient scaling* [179, 182] amplifica a influência da atualização local do cliente malicioso sobre a agregação global. Após o treinamento local, o atacante calcula a diferença entre os pesos treinados w_{local} e os pesos do modelo global w_{global}, e amplifica essa diferença por um fator λ:

$$w_{\text{envio}} = w_{\text{global}} + \lambda \cdot (w_{\text{local}} - w_{\text{global}}) \tag{8.2}$$

Com $\lambda = 1$, a atualização equivale ao treino normal. Os fatores avaliados foram $\lambda \in \{1{,}0;\ 2{,}0;\ 5{,}0;\ 10{,}0\}$. A Tabela 14 apresenta o peso efetivo do cliente malicioso na agregação FedAvg para cada fator.

Tabela 14 – Peso efetivo do cliente malicioso na agregação FedAvg em função do fator de escala λ, com $n = 3$ clientes de volumes iguais.

Fator λ	**Peso efetivo $\frac{\lambda}{\lambda+2}$**	**Interpretação**
1,0× (controle)	33,3%	Contribuição normal
2,0×	50,0%	Paridade com os dois honestos combinados
5,0×	71,4%	Dominância sobre os dois honestos
10,0×	83,3%	Controle quase total da agregação

O *gradient scaling* é sempre combinado com 100% de *label-flipping* nas fases em que é aplicado. O ponto de inserção no código ocorre após o treinamento local, mas antes do retorno dos pesos ao servidor, de modo que as métricas de avaliação refletem o modelo global e não o estado amplificado.

8.2.5 Classificação na taxonomia de ataques

Na taxonomia de Fang et al. [179], os ataques avaliados classificam-se como: envenenamento de modelo não direcionado (*untargeted*), com superfície combinada de *data poisoning* (*label-flipping*) e *model poisoning* (*gradient scaling*), não adaptativo (sem otimização contra defesa específica), e persistente (executado em todas as rodadas de comunicação).

A escolha por ataques não adaptativos é deliberada. O objetivo é estabelecer um diagnóstico de vulnerabilidade de referência: se ataques simples já comprometem o sistema, a necessidade de defesas robustas fica demonstrada sem depender de capacidades adversariais sofisticadas.

8.2.6 Métrica operacional de segurança: C2A Rate

As métricas convencionais — *accuracy* global e F1-score macro — podem mascarar a degradação seletiva de uma classe minoritária. Para capturar o impacto adversarial sobre as decisões de ciclo de vida, este trabalho adota como métrica operacional a **taxa de misclassificação C→A** (*C2A Rate*):

$$\text{C2A Rate} = \frac{\text{CM}[3][0]}{\sum_j \text{CM}[3][j]} \tag{8.3}$$

onde CM é a matriz de confusão na rodada final, CM[3][0] é o número de amostras verdadeiramente *Critical* (classe 3) classificadas como *Adequate* (classe 0), e o denominador é o total de amostras *Critical*. Cada ponto percentual de C2A corresponde a decisões de ciclo de vida omitidas em fatias que atendem aplicações *safety-critical*.

8.2.7 Formulação do problema

O problema investigado é: dado um sistema de classificação federada de impacto de fatias ITS operando com $n = 3$ clientes, em que $f = 1$ cliente é comprometido por um adversário capaz de executar *label-flipping* e *gradient scaling*, determinar (i) a vulnerabilidade do classificador global sob diferentes intensidades de ataque e arquiteturas de rede neural, quantificada pela C2A Rate; e (ii) a eficácia de estratégias de agregação Bizantino-robusta (Krum [183] e Trimmed Mean [184]) em mitigar os ataques, em comparação com agregações padrão (FedAvg [32] e FedProx [185]).

8.3 Prova de Conceito: Metodologia Experimental

Esta seção descreve a metodologia experimental adotada para avaliar a vulnerabilidade do classificador federado frente aos ataques definidos na Seção 8.2.

8.3.1 Dataset e pré-processamento

O dataset utilizado é idêntico ao do Capítulo 7: 5.400 amostras geradas a partir da emulação do corredor urbano descrita no Capítulo 5 [11], com 4 classes de impacto (*Adequate*, *Warning*, *Severe*, *Critical*) correspondentes à taxonomia do AIMS [12]. O pipeline de pré-processamento reutiliza os módulos do AIMS centralizado (`preprocess_dataset.py` e `impact_labeling.py`), acrescidos de seleção de features numéricas, *one-hot encoding* de colunas categóricas, remoção de colunas não preditivas, e normalização *StandardScaler* ajustada no conjunto de treino global antes do particionamento entre clientes.

Cada classe de impacto mapeia para uma ação de ciclo de vida: *Adequate* indica operação dentro dos parâmetros de SLA; *Warning* indica degradação inicial com monitoramento intensificado; *Severe* indica degradação que demanda ação proativa (escalonamento, readmissão); *Critical* indica violação de SLA iminente ou em curso, demandando ação urgente.

8.3.2 Arquitetura de aprendizado federado

A infraestrutura experimental é compartilhada com o Capítulo 7. A Tabela 15 resume os componentes e a Tabela 16 apresenta os parâmetros da topologia federada.

8.3.3 Distribuição de dados

Os experimentos de segurança utilizam exclusivamente particionamento Non-IID, que reflete a dinâmica do corredor urbano. A Tabela 17 apresenta a alocação de classes por cliente.

Tabela 15 – Ambiente computacional.

Componente	**Especificação**
GPU	NVIDIA L40S (46 068 MiB VRAM)
Driver / CUDA	570.124.06 / 12.8
Python	3.11.10
TensorFlow	2.18.0 (backend Keras)
Flower (flwr)	1.27.0 com `fl.simulation.start_simulation()` e Ray
scikit-learn	1.6.1
NumPy	2.2.4
Reprodutibilidade	Seed global 42; seeds por cliente 42 + `client_id`
Repositório	https://github.com/saraivacode/its-fl-slice-impact

Tabela 16 – Parâmetros da topologia federada.

Parâmetro	**Valor**	**Justificativa**
Clientes	3 (RSUs)	Mapeamento direto do cenário físico
Rodadas	30	Suficiente para convergência observada no Cap. 7
Épocas locais	3	Equilíbrio entre treino local e comunicação
Tamanho do *batch*	32	Padrão para *datasets* desta escala
FedProx μ	0,1	Regularização proximal *client-side*

Tabela 17 – Distribuição Non-IID de classes por cliente (RSU).

Cliente	***Adequate***	***Warning***	***Severe***	***Critical***
RSU 0 (*Client* 0 — atacante)	55%	30%	10%	5%
RSU 1 (*Client* 1)	15%	40%	40%	15%
RSU 2 (*Client* 2)	30%	30%	50%	80%

O *Client* 0 (atacante) possui composição predominantemente *Adequate* e apenas 5% de amostras *Critical* — a classe-alvo do *label-flipping*. Essa distribuição decorre da topologia do corredor: a RSU 0, localizada na extremidade com maior concentração de tráfego pré-congestionamento, observa predominantemente condições de operação adequada. O *Client* 2 concentra 80% das amostras *Critical*, correspondendo à RSU que observa as condições mais degradadas.

A escolha do *Client* 0 como atacante configura um cenário operacionalmente relevante: o adversário opera a partir de um ponto de acesso com representação mínima da classe crítica. A eficácia do ataque nessas condições — demonstrada na Seção 8.4.1 — indica que o *gradient scaling* pode compensar a escassez de amostras da classe-alvo no cliente malicioso.

8.3.4 Modelos de classificação

Três arquiteturas de redes neurais são avaliadas, as mesmas utilizadas no Capítulo 7: DNN (*feedforward* com camadas 128-64-32, regularização L2 e *dropout*), LSTM (64-32 unidades com *dropout*) e GRU (64-32 unidades com *dropout*). Todas utilizam camada de saída com 4 neurônios e ativação *softmax*, otimizador Adam e *Sparse Categorical Cross-Entropy*. Para LSTM e GRU, as features são formatadas como sequências de comprimento 1 ($1 \times d$, onde d é a dimensão de features). A avaliação de três arquiteturas com mecanismos de processamento distintos (*feedforward* puro, células de memória com *gates*, unidades recorrentes simplificadas) permite verificar se a vulnerabilidade ao envenenamento é estrutural ou dependente da arquitetura.

8.3.5 Estratégias de agregação

Quatro estratégias de agregação são avaliadas. As duas primeiras — FedAvg [32] e FedProx [185], descritas no Capítulo 2 — não possuem mecanismo de defesa contra atualizações maliciosas e servem como *baselines* de vulnerabilidade. As duas defesas Bizantino-robustas são descritas a seguir.

O **Krum** [183] seleciona, a cada rodada, o modelo local mais "central" no espaço de parâmetros: para cada cliente i, calcula o *score* s_i como a soma das distâncias Euclidianas aos $k = \max(n - f - 2,\ 1)$ vizinhos mais próximos, e adota o modelo com menor *score* como modelo global. A configuração utiliza `num_byzantine` $= 1$, resultando em $k = \max(3 - 1 - 2,\ 1) = 1$ vizinho. A implementação estende a classe `FedAvg` do Flower, sobrescrevendo o método `aggregate_fit`.

O **Trimmed Mean** [184] realiza agregação coordenada a coordenada: para cada parâmetro do modelo, ordena os valores dos n clientes, descarta uma fração `trim_fraction` $=$ 0,1 de cada extremo, e calcula a média dos valores restantes. Com $n = 3$ clientes e `trim_fraction` $=$ 0,1, o efeito prático é equivalente à seleção do valor mediano por coordenada.

Cada defesa (Krum e Trimmed Mean) é combinada com ambas as estratégias base (FedAvg e FedProx), totalizando 6 configurações de agregação por modelo: FedAvg, FedProx, Krum+FedAvg, Krum+FedProx, TrimmedMean+FedAvg e TrimmedMean+FedProx.

8.3.6 Desenho experimental

A avaliação é organizada em três fases progressivas, totalizando 144 experimentos. A Tabela 18 apresenta a matriz completa.

A **Fase 1a** avalia o *label-flipping* isolado (100% de flip, escala 1,0×) para estabelecer se o envenenamento de dados sem amplificação é suficiente para degradar o classificador.

Tabela 18 – Matriz completa de experimentos (144 configurações).

Fase	**Modelos**	**Estratégias**	**Variável**	**Valores**	**Total**
1a — *Label-flip* puro	3	6	—	—	18
1b — *Gradient scaling*	3	6	Fator λ	2, 5, 10	54
Sensibilidade	3	6	Épocas locais	1, 3, 5, 10	72
Total					**144**

A **Fase 1b** combina *label-flipping* a 100% com fatores de escala crescentes (2×, 5×, 10×), identificando o limiar a partir do qual o ataque compromete cada combinação modelo/estratégia. A **análise de sensibilidade** varia o número de épocas locais ($e \in \{1, 3, 5, 10\}$) com *label-flipping* sem escalamento, investigando a interação entre intensidade do treinamento local e resiliência ao ataque.

Todos os experimentos utilizam distribuição Non-IID, 30 rodadas de comunicação e seed determinístico (42 global, 42 + `client_id` por cliente).

8.4 Resultados e Discussão

8.4.1 Resultados experimentais

8.4.1.1 Fase 1a — *Label-flipping* puro

A Tabela 19 apresenta os resultados dos 18 experimentos da Fase 1a. Todos registraram C2A Rate igual a zero.

O resultado é inequívoco: o *label-flipping* sem amplificação de gradientes não degrada a classificação da classe *Critical* em nenhuma das 18 combinações de modelo e estratégia avaliadas. A *accuracy* mantém-se entre 91,76% e 93,52%, compatível com os *baselines* sem ataque do Capítulo 7.

A explicação reside na composição do cliente atacante. O *Client* 0 possui apenas 5% de amostras *Critical*, o que resulta em uma contaminação global mínima: o *label-flipping* afeta poucas dezenas de amostras em um universo de treino de ∼4.320. Além disso, a correlação forte entre features de telemetria e níveis de impacto — estabelecida por construção na emulação — permite que os modelos treinados nos clientes honestos preservem a associação correta, diluindo a contribuição envenenada pela agregação federada.

Essa resiliência não significa que o sistema é seguro. A Fase 1b demonstra que o *gradient scaling* supera essa atenuação natural.

Tabela 19 – Fase 1a — *Label-flipping* puro, escala 1,0× (todos os 18 experimentos com C2A = 0,0000).

Modelo	Agregação	Acc	F1-Macro	@85%	@90%
DNN	FedAvg	0,9352	0,9348	R2	R5
DNN	FedProx	0,9259	0,9249	R2	R7
DNN	Krum (FedAvg)	0,9250	0,9235	R2	R6
DNN	Krum (FedProx)	0,9176	0,9114	R2	R6
DNN	TM (FedAvg)	0,9204	0,9250	R3	R4
DNN	TM (FedProx)	0,9296	0,9285	R2	R10
LSTM	FedAvg	0,9333	0,9333	R3	R7
LSTM	FedProx	0,9278	0,9268	R3	R9
LSTM	Krum (FedAvg)	0,9204	0,9168	R3	R9
LSTM	Krum (FedProx)	0,9343	0,9328	R4	R10
LSTM	TM (FedAvg)	0,9250	0,9254	R3	R9
LSTM	TM (FedProx)	0,9287	0,9278	R3	R9
GRU	FedAvg	0,9287	0,9284	R3	R9
GRU	FedProx	0,9231	0,9223	R3	R12
GRU	Krum (FedAvg)	0,9213	0,9189	R3	R6
GRU	Krum (FedProx)	0,9204	0,9149	R3	R12
GRU	TM (FedAvg)	0,9296	0,9311	R3	R10
GRU	TM (FedProx)	0,9231	0,9219	R3	R9

8.4.1.2 Fase 1b — *Gradient scaling*

A Tabela 20 apresenta os resultados com fator de escala 5× e a Tabela 21 com fator 10×, para as combinações representativas de modelo e estratégia. Os resultados com fator 2× registraram C2A próximo de zero em quase todas as configurações (exceção: DNN FedAvg com C2A = 0,0055) e são omitidos por brevidade.

Tabela 20 – Fase 1b — Fator de escala 5× (*label-flip* 100%).

Modelo	Agregação	Acc	F1-Macro	C2A Rate
DNN	FedAvg	0,9056	0,8873	0,1657
DNN	FedProx	0,8796	0,8468	0,3481
DNN	Krum (FedAvg)	0,9278	0,9202	0,0000
DNN	TM (FedAvg)	0,9407	0,9350	0,0166
LSTM	FedAvg	0,8250	0,7492	0,6354
LSTM	FedProx	0,8593	0,8167	0,4254
LSTM	Krum (FedAvg)	0,9231	0,9217	0,0000
LSTM	TM (FedAvg)	0,9343	0,9347	0,0000
GRU	FedAvg	0,8741	0,8369	0,3757
GRU	FedProx	0,8241	0,7555	0,6133
GRU	Krum (FedAvg)	0,9222	0,9209	0,0000
GRU	TM (FedAvg)	0,9306	0,9335	0,0000

Tabela 21 – Fase 1b — Fator de escala 10× (*label-flip* 100%).

Modelo	Agregação	Acc	F1-Macro	C2A Rate
DNN	FedAvg	0,2231	0,1766	**1,0000**
DNN	FedProx	0,1870	0,0788	0,0000†
DNN	Krum (FedAvg)	0,9222	0,9144	0,0000
DNN	TM (FedAvg)	0,9306	0,9289	0,0055
LSTM	FedAvg	0,7574	0,6161	**0,8785**
LSTM	FedProx	0,7454	0,5961	**0,9006**
LSTM	Krum (FedAvg)	0,9222	0,9198	0,0000
LSTM	TM (FedAvg)	0,9333	0,9327	0,0000
GRU	FedAvg	0,7463	0,5762	**0,9613**
GRU	FedProx	0,7269	0,5422	**0,9945**
GRU	Krum (FedAvg)	0,9287	0,9245	0,0000
GRU	TM (FedAvg)	0,9167	0,9202	0,0000

† DNN FedProx 10×: *accuracy* de 18,7% e C2A = 0,0 indica colapso total do modelo — o classificador atribui todas as amostras a uma única classe, não necessariamente *Adequate*. A métrica C2A é zero por artefato, não por robustez.

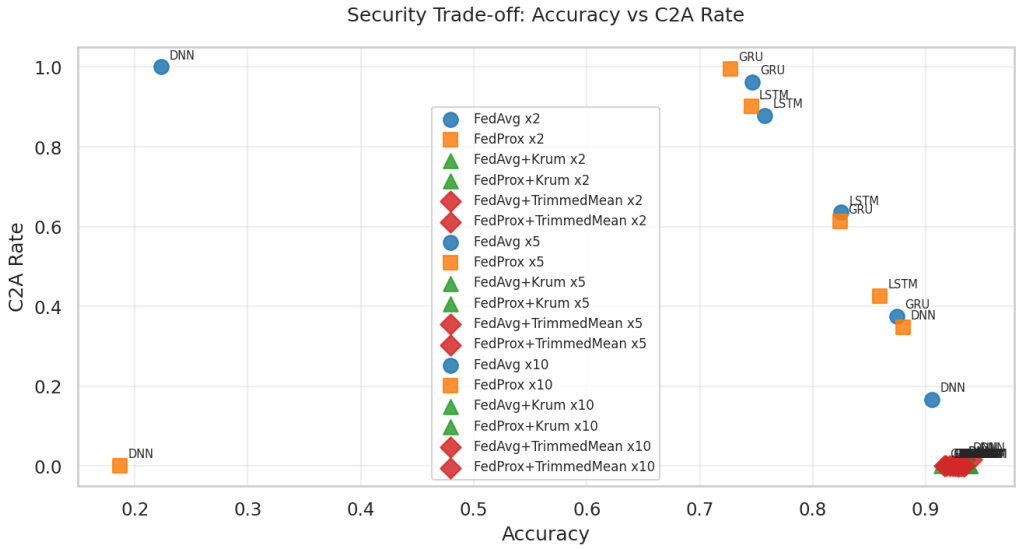


Figura 35 – Compromisso entre *accuracy* e segurança (C2A Rate) na Fase 1b. Cada ponto representa uma configuração modelo/estratégia/escala. As defesas Bizantino-robustas (Krum e Trimmed Mean, em verde e vermelho) mantêm-se no quadrante inferior direito (*accuracy* $> 91\%$, C2A ≈ 0), enquanto FedAvg e FedProx sem defesa (azul e laranja) migram para o quadrante superior esquerdo com o aumento do fator de escala, atingindo C2A próximo de 100% com $\lambda = 10\times$.



Figura 36 – Evolução da C2A Rate ao longo das 30 rodadas de comunicação na Fase 1b, por modelo e estratégia de agregação. Com fatores 5× e 10×, FedAvg e FedProx (azul e laranja) apresentam oscilação persistente, indicando que o modelo global alterna entre estados envenenados e parcialmente recuperados sem convergir. Krum e Trimmed Mean (verde e vermelho) mantêm C2A ≈ 0 em todas as rodadas e escalas.

8.4.1.3 Análise de sensibilidade — épocas locais

A Tabela 22 apresenta os resultados para DNN FedAvg (*label-flip*, escala 1,0×) em função do número de épocas locais, como exemplo representativo.

Tabela 22 – Sensibilidade ao número de épocas locais — DNN FedAvg (*label-flip*, $\lambda = 1{,}0$).

Épocas	**Acc**	**F1-Macro**	**C2A Rate**	**@90%**
1	0,9065	0,9046	0,0000	R17
3	0,9352	0,9348	0,0000	R5
5	0,9333	0,9305	0,0055	R4
10	0,9509	0,9492	0,0000	R3

A C2A Rate permanece essencialmente nula para *label-flipping* sem escalamento em todas as variações de épocas e modelos (72 experimentos). O número de épocas afeta a velocidade de convergência (R17 com $e = 1$ contra R3 com $e = 10$) e a *accuracy* final, mas não a vulnerabilidade à manipulação de rótulos isoladamente. A Figura 37 ilustra esse padrão: os painéis superiores mostram *accuracy* crescente com o número de épocas, enquanto os painéis inferiores confirmam C2A Rate essencialmente nula em todas as configurações. Os dados completos dos 72 experimentos estão disponíveis no repositório.

8.4.2 Discussão

8.4.2.1 Vulnerabilidade estrutural ao *gradient scaling*

Os resultados da Fase 1b demonstram que o *gradient scaling* combinado com *label-flipping* é capaz de comprometer o classificador federado de impacto em todas as três arquiteturas avaliadas. Com fator 10×, a C2A Rate atinge 100% para DNN FedAvg,

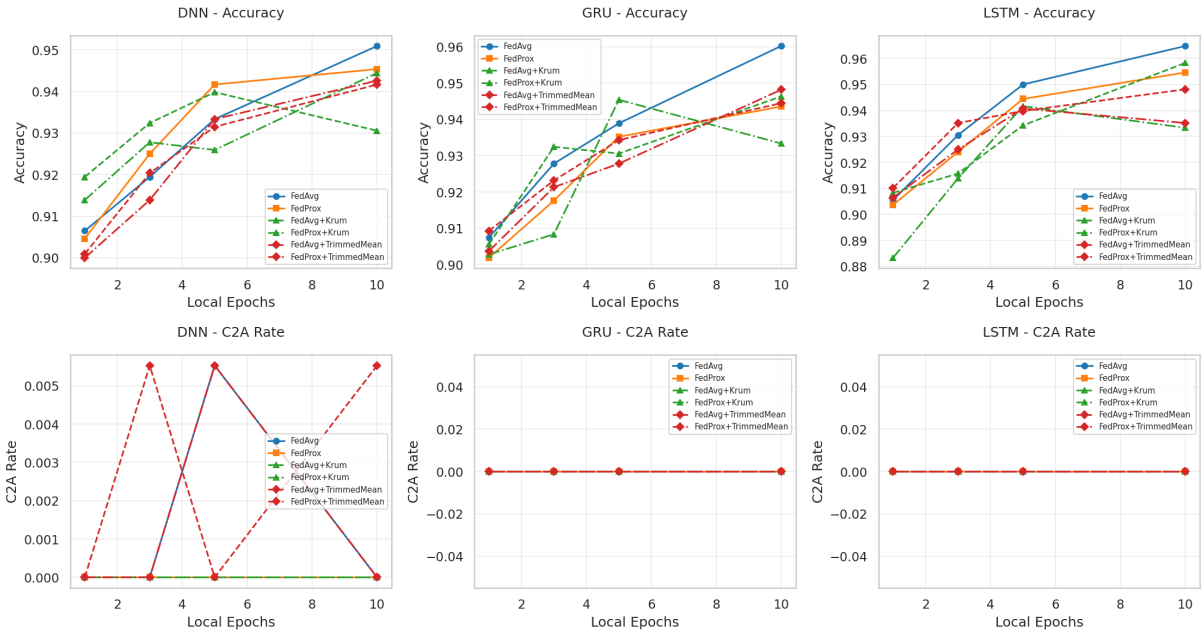


Figura 37 – Sensibilidade ao número de épocas locais ($e \in \{1, 3, 5, 10\}$) sob *label-flipping* sem escalamento. Painéis superiores: *accuracy* final por modelo e estratégia — valores crescem monotonicamente com o número de épocas. Painéis inferiores: C2A Rate correspondente — permanece essencialmente nula em todas as configurações, confirmando que o *label-flipping* isolado não compromete a classificação da classe *Critical* independentemente da intensidade do treinamento local.

96,13% para GRU FedAvg e 87,85% para LSTM FedAvg. O GRU FedProx registra o pior cenário entre as arquiteturas recorrentes, com C2A = 99,45%. A Figura 35 sintetiza essa separação: as configurações sem defesa dispersam-se pelo quadrante de alta C2A e baixa *accuracy*, enquanto Krum e Trimmed Mean permanecem agrupados no quadrante de operação segura.

Essa vulnerabilidade é estrutural: manifesta-se em arquiteturas com mecanismos de processamento distintos (*feedforward*, células de memória, unidades recorrentes). O denominador comum é a ausência de verificação de integridade na camada de agregação — a agregação por média ponderada (FedAvg/FedProx) aceita todas as atualizações sem discriminação.

No contexto operacional do AIMS, uma C2A Rate próxima de 100% significa que fatias em violação de SLA são sistematicamente classificadas como em condição adequada, suprimindo todas as ações de ciclo de vida para aplicações *safety-critical*.

8.4.2.2 FedProx não constitui defesa

O FedProx apresenta C2A Rate igual ou superior ao FedAvg na maioria das configurações sob ataque. Com fator 5×: DNN FedProx registra C2A = 0,3481 contra 0,1657 do FedAvg; GRU FedProx registra 0,6133 contra 0,3757 do FedAvg. Com fator

10×: GRU FedProx atinge 0,9945 contra 0,9613 do FedAvg; LSTM FedProx atinge 0,9006 contra 0,8785 do FedAvg.

O mecanismo que explica esse comportamento é o termo proximal $\frac{\mu}{2}\|w - w^t\|^2$, que mantém o modelo local próximo do modelo global. Quando o modelo global já está parcialmente contaminado por rodadas anteriores, essa proximidade ancora os modelos dos clientes honestos no espaço envenenado em vez de permitir que eles corrijam a direção de treinamento. O FedProx, projetado para mitigar *client drift* em cenários heterogêneos [185], não oferece proteção contra adversários e pode agravar a vulnerabilidade.

8.4.2.3 Eficácia das defesas Bizantino-robustas

O Krum mantém C2A = 0,0000 em todas as configurações com fatores 5× e 10×, para os três modelos e ambas as bases de agregação (FedAvg e FedProx). A *accuracy* permanece acima de 92%, compatível com os *baselines* sem ataque. A atualização do cliente malicioso amplificado constitui o *outlier* no espaço de parâmetros e é rejeitada em todas as rodadas. A Figura 36 confirma essa estabilidade: a C2A Rate do Krum e do Trimmed Mean permanece em zero ao longo de todas as 30 rodadas, contrastando com a oscilação persistente do FedAvg e FedProx sob fatores 5× e 10×.

O Trimmed Mean apresenta C2A $\leq$ 0,0166 em todas as configurações avaliadas, com *accuracy* entre 91,67% e 94,07%. A única exceção marginal é DNN TrimmedMean com 5× (C2A = 0,0166) e 10× (C2A = 0,0055), valores operacionalmente desprezíveis.

Ambas as defesas operam no regime $n = 3$, $f = 1$, que satisfaz a condição mínima $n \geq 2f + 1$. Os resultados demonstram que defesas Bizantino-robustas clássicas são eficazes neste regime contra ataques não adaptativos, mesmo quando o atacante domina mais de 80% do peso efetivo na agregação padrão.

8.5 Limitações e Direções Futuras

Os resultados apresentados neste capítulo devem ser interpretados à luz das limitações inerentes ao escopo experimental adotado. Esta seção documenta essas limitações e identifica direções de investigação que delas decorrem.

8.5.1 Limitações experimentais

A primeira limitação refere-se ao **número de participantes federados**. O sistema opera com $n = 3$ clientes, mapeando diretamente as 3 RSUs do corredor urbano emulado. Esse é o regime mínimo para a operação das defesas Bizantino-robustas avaliadas: o Krum com `num_byzantine` = 1 seleciona o modelo mais próximo de apenas $k = 1$ vizinho, e o Trimmed Mean com `trim_fraction` = 0,1 opera como mediana coordenada. Em

implantações com dezenas de RSUs, essas defesas dispõem de margem significativamente maior, e o próprio FedAvg tende a ser mais resiliente pela diluição da contribuição maliciosa em um pool maior de participantes honestos. A eficácia observada das defesas no regime mínimo constitui um resultado conservador: espera-se desempenho igual ou superior com mais participantes.

A segunda limitação diz respeito à **natureza dos ataques avaliados**. Tanto o *label-flipping* quanto o *gradient scaling* são ataques não adaptativos — o adversário não otimiza sua estratégia contra a defesa em operação. Ataques adaptativos, como os propostos por Fang et al. [179] e Shejwalkar e Houmansadr [182], calibram a atualização maliciosa para minimizar a distância aos clientes honestos no espaço de parâmetros, podendo contornar defesas baseadas em distância (Krum) ou aparamento (Trimmed Mean). A eficácia demonstrada das defesas neste capítulo aplica-se ao cenário de ataques não adaptativos; ataques adaptativos podem exigir mecanismos de detecção mais sofisticados.

A terceira limitação concerne o **escopo da distribuição de dados**. Os experimentos de segurança utilizam exclusivamente o cenário Non-IID, que é o operacionalmente relevante para corredores veiculares com heterogeneidade espacial. A avaliação em cenários IID — nos quais a heterogeneidade entre clientes é menor e a contribuição envenenada pode ser mais difícil de distinguir por defesas baseadas em distância — completaria o diagnóstico e permitiria isolar o efeito da heterogeneidade sobre a eficácia das defesas.

A quarta limitação refere-se à **composição do cliente atacante**. O *Client* 0, selecionado como atacante, possui apenas 5% de amostras *Critical* na distribuição Non-IID. Essa escolha decorre do mapeamento com a topologia do corredor, mas representa um cenário em que o atacante tem acesso mínimo à classe-alvo do *label-flipping*. A avaliação com o *Client* 2 como atacante — que concentra 80% das amostras *Critical* — permitiria quantificar o efeito da composição do cliente malicioso sobre a eficácia do envenenamento. É plausível que um atacante com acesso majoritário à classe *Critical* produza degradação mais severa, mesmo sem *gradient scaling*.

A quinta limitação refere-se à **escala e diversidade do dataset**. O dataset AIMS contém 5.400 amostras provenientes de um cenário de corredor urbano específico (650 m, 3 RSUs, 4 tipos de aplicação ITS). Embora representativo do cenário de fatiamento orientado a aplicações avaliado nos capítulos anteriores, esse volume é modesto em comparação com *benchmarks* de segurança de rede. A validação em *datasets* de maior escala e diversidade fortaleceria a generalização das conclusões.

A sexta limitação refere-se ao **ataque de envenenamento exclusivamente por *label-flipping***. Apenas a direção *Critical*→*Adequate* foi avaliada. Outras direções de manipulação (por exemplo, *Severe*→*Adequate*, ou ataques de *backdoor* que inserem padrões específicos) podem produzir efeitos qualitativamente distintos sobre o classificador de impacto.

8.5.2 Direções futuras

As limitações identificadas motivam quatro direções de investigação futura que estendem o diagnóstico de vulnerabilidade apresentado neste capítulo.

A primeira direção consiste na **progressão do número de participantes** ($n \in \{3, 5, 7\}$) com variação simultânea do número de atacantes. Essa progressão permitiria mapear a curva de resiliência das defesas em função da proporção atacantes/participantes e determinar o limiar a partir do qual defesas clássicas como Krum e Trimmed Mean operam com margem confortável. A expectativa é que o regime $n = 5$ já ofereça margem operacional suficiente para cenários com $f = 1$ atacante.

A segunda direção envolve a **avaliação com ataques adaptativos** (*AGR-tailored*) [179], que otimizam a atualização maliciosa para minimizar a distância aos clientes honestos. Ataques adaptativos constituem o teste mais rigoroso para defesas Bizantino-robustas e representam a extensão natural do diagnóstico não adaptativo apresentado neste capítulo.

A terceira direção consiste na **validação com *datasets* multi-fatia**. O dataset 5G-NIDD [188], que contém tráfego real segmentado por tipo de fatia 5G (eMBB, mMTC, URLLC), permitiria avaliar o impacto diferenciado dos ataques sobre cada classe de serviço e testar defesas orientadas pela criticidade da fatia — aplicando, por exemplo, Trimmed Mean para fatias URLLC e FedAvg para fatias eMBB com menor sensibilidade a classificações errôneas.

A quarta direção envolve a adoção de **aprendizado federado personalizado**. Estratégias como FedBN (*Federated Batch Normalization*) e FedPer (*personalized layers*) permitem que cada cliente mantenha camadas locais adaptadas à sua distribuição de dados, compartilhando apenas as camadas de representação com o servidor [186]. No contexto ITS, a personalização pode simultaneamente melhorar a *accuracy* em cenários Non-IID e reduzir a superfície de ataque, uma vez que as camadas personalizadas não são expostas à agregação global. A combinação de personalização com agregação robusta constitui uma abordagem promissora que endereça simultaneamente heterogeneidade e segurança.

8.6 Considerações Finais do Capítulo

Este capítulo investigou a segurança do classificador federado de impacto de fatias ITS do AIMS frente a ataques de envenenamento, por meio de 144 configurações experimentais que combinam três arquiteturas de rede neural (DNN, LSTM, GRU), seis estratégias de agregação (FedAvg, FedProx, e cada uma combinada com Krum e Trimmed Mean), três fatores de amplificação adversarial (2×, 5×, 10×), e quatro configurações de treinamento local (1, 3, 5 e 10 épocas).

Os resultados constroem uma narrativa em três estágios progressivos. O primeiro estágio (Fase 1a, 18 configurações) estabeleceu que o classificador federado apresenta resiliência natural ao *label-flipping* direcionado (*Critical*→*Adequate*) sem amplificação de gradientes: a C2A Rate permaneceu nula em todas as configurações. Essa resiliência decorre da contaminação limitada do cliente atacante (5% de amostras *Critical*), da correlação forte entre features de telemetria e níveis de impacto, e da diluição natural pela agregação federada.

O segundo estágio (Fase 1b, 54 configurações) demonstrou que o *gradient scaling* supera essa resiliência. Com fator 10×, a C2A Rate atinge 100% para DNN FedAvg, 96,13% para GRU FedAvg e 87,85% para LSTM FedAvg. O GRU FedProx registra C2A de 99,45%. Essa vulnerabilidade é estrutural — manifesta-se nas três arquiteturas avaliadas — e não depende do mecanismo de processamento do modelo. No contexto operacional do AIMS, C2A Rate próxima de 100% corresponde à supressão de todas as ações de ciclo de vida para fatias *safety-critical* em violação de SLA. Um achado adicional com implicação prática é que o FedProx apresenta C2A Rate igual ou superior ao FedAvg na maioria das configurações sob ataque: a regularização proximal, ao ancorar modelos honestos em um modelo global já parcialmente contaminado, amplifica a vulnerabilidade em vez de mitigá-la.

O terceiro estágio demonstrou a eficácia das defesas Bizantino-robustas no regime avaliado. O Krum manteve C2A = 0,0000 em todas as configurações com fatores 5× e 10×, para os três modelos. O Trimmed Mean manteve C2A $\leq$ 0,0166 em todos os 54 cenários da Fase 1b, com *accuracy* compatível com os *baselines* sem ataque. A análise de sensibilidade (72 configurações) confirmou que o número de épocas locais afeta a velocidade de convergência e a *accuracy* final, mas não introduz vulnerabilidade ao *label-flipping* isolado.

A contribuição central deste capítulo é o diagnóstico de vulnerabilidade operacional do classificador federado de impacto de fatias ITS: a demonstração, com evidência experimental sistemática, de que ataques de envenenamento simples podem suprimir completamente a detecção de condições críticas quando a agregação federada não incorpora mecanismos de robustez. A métrica C2A Rate, ao quantificar diretamente decisões de ciclo de vida omitidas, traduz resultados de segurança em aprendizado de máquina para o domínio de gerenciamento de fatias de rede, permitindo que operadores avaliem o risco concreto de cada configuração. A demonstração de que defesas clássicas como Krum e Trimmed Mean são eficazes no regime $n = 3$, $f = 1$ contra ataques não adaptativos complementa o diagnóstico com uma recomendação operacional concreta: a adoção de agregação robusta é requisito para implantações de FL em cenários ITS com fatiamento inteligente.

As limitações identificadas — particularmente o regime de poucos participantes,

a natureza não adaptativa dos ataques e a composição específica do cliente atacante — definem a fronteira do diagnóstico e motivam as direções futuras de progressão de clientes, ataques adaptativos e validação em *datasets* multi-fatia, que compõem a agenda de trabalhos futuros apresentada no Capítulo 9.

9 CONCLUSÕES E TRABALHOS FUTUROS

Esta tese endereçou o problema do gerenciamento inteligente do ciclo de vida de fatias de rede em cenários B5G que suportam aplicações de Sistemas Inteligentes de Transporte (ITS). Conforme formulado no Capítulo 1, a dinâmica veicular, marcada por variações de carga, mudanças de topologia e condições de canal heterogêneas, exige mecanismos capazes de: (i) identificar o contexto veicular a partir de fontes acessíveis na infraestrutura; (ii) orquestrar recursos de forma dinâmica com algoritmos de controle orientados aos requisitos heterogêneos das aplicações ITS; (iii) classificar o impacto das condições observadas sobre as políticas de fatiamento vigentes, produzindo sinais acionáveis para adaptação; (iv) escalar o treinamento dos modelos de classificação de forma distribuída sob heterogeneidade estatística; e (v) garantir a integridade dessas classificações diante de participantes potencialmente comprometidos. As cinco questões de pesquisa (QP1–QP5) e os cinco desafios em aberto (D1–D5) identificados na introdução orientaram as contribuições apresentadas ao longo dos capítulos precedentes, que são sintetizadas a seguir.

9.1 Síntese das contribuições

As contribuições desta tese seguem uma narrativa progressiva que parte dos fundamentos (Capítulo 2), avança pela identificação de contexto veicular (Capítulos 3 e 4), consolida-se no fatiamento orientado a aplicações (Capítulo 5), evolui para a classificação de impacto centralizada (Capítulo 6) e distribuída (Capítulo 7), e aborda a segurança da inteligência distribuída (Capítulo 8).

Detecção de contexto veicular com dados públicos (Capítulo 3 — QP1, D1): O Capítulo 3 apresentou uma abordagem de detecção de congestionamentos urbanos baseada em classificadores de aprendizado supervisionado treinados com dados históricos e públicos de mobilidade urbana de ônibus da cidade do Rio de Janeiro. A abordagem demonstrou a viabilidade de utilizar dados abertos para identificação do contexto veicular em cenários reais, estruturando o processo de decisão com uma *Machine Learning Engine* que treina, testa e aplica os modelos para detecção em dados correntes. Essa contribuição endereça o desafio D1, ao demonstrar que fontes acessíveis — dados públicos de transporte — podem alimentar decisões de gerenciamento de fatias sem depender de sensores dedicados.

Predição de contexto veicular por estatísticas de rede (Capítulo 4 — QP1, D1):

O Capítulo 4 avançou na identificação do contexto veicular ao propor uma abordagem que utiliza estatísticas de rede de comunicação, como RTT e *throughput*, como *features* para classificadores de aprendizado supervisionado. A prova de conceito em cenário típico de congestionamento urbano demonstrou que é possível inferir o contexto veicular a partir de dados obtidos no núcleo da rede, ampliando as possibilidades de detecção sem depender de dados externos à infraestrutura de comunicação. Combinada com a contribuição anterior, esta abordagem completa a resposta à QP1, oferecendo duas vias complementares de identificação de contexto.

Framework de fatiamento de rede 5G orientado a aplicações ITS (Capítulo 5 — QP2, D2): O Capítulo 5 consolidou as contribuições de contexto ao propor um *framework* de fatiamento de rede 5G orientado a aplicações ITS. A arquitetura baseada em SDN, com algoritmos de controle explícitos e avaliação de desempenho em emulação realística com o Mininet-WiFi, demonstrou a viabilidade de atender dinamicamente aos requisitos de diferentes aplicações ITS em uma infraestrutura móvel compartilhada. As quatro classes de prioridade de aplicação (A, B, C e G) e os mecanismos de redirecionamento e limitação de tráfego constituem a base arquitetural sobre a qual as contribuições subsequentes são construídas, endereçando o desafio D2.

AIMS — Gerenciamento Adaptativo e Inteligente de Fatias (Capítulo 6 — QP3, D3): O Capítulo 6 introduziu o *framework* AIMS (*Adaptive and Intelligent Management of Slicing*), que endereça a necessidade de ajuste dinâmico ao introduzir inteligência leve na borda da rede para análise de impacto de políticas de fatiamento. A arquitetura híbrida *cloud-edge*, com sete funções distribuídas entre controladores centrais e locais (F1–F7), cobre as quatro fases do ciclo de vida das fatias conforme definido pelo 3GPP. A taxonomia de classificação de impacto em quatro níveis (*Adequate*, *Warning*, *Severe*, *Critical*) fornece gatilhos estáveis e interpretáveis para ações de adaptação em *closed-loop*. Os resultados experimentais com um *dataset* de 5.400 amostras e 25 *features* demonstraram que modelos *ensemble* leves, como CatBoost (94,8% de acurácia) e Random Forest (94,4% de acurácia), conseguem traduzir telemetria veicular ruidosa em sinais acionáveis de impacto, o que responde à QP3.

Classificação federada de impacto de fatias (Capítulo 7 — QP4, D4): O Capítulo 7 estendeu o AIMS ao empregar aprendizado federado para classificação distribuída e escalável de impacto de fatias. O classificador de impacto foi tratado como camada de inteligência transversal ao ciclo de vida, treinado de forma federada entre controladores locais associados a RSUs e um controlador central. Os resultados experimentais com três arquiteturas neurais (DNN, LSTM e GRU), duas estratégias federadas (FedAvg e FedProx) e cenários IID e Non-IID demonstraram que o treinamento federado é viável e competitivo. Em cenário IID, o desempenho federado aproxima-se do centralizado (Macro F1-*score* $>$ 0,93), com convergência rápida (85% de acurácia já na segunda rodada). O

principal resultado em cenário Non-IID foi a degradação inferior a 0,5 ponto percentual no F1-*score* médio em relação ao cenário IID, com o melhor resultado federado Non-IID alcançando 93,9% de acurácia (DNN-FedAvg). O FedProx com $\mu = 0{,}1$ não apresentou vantagem consistente sobre o FedAvg, com diferenças dentro de $\pm 1{,}14$ pp. Esses resultados respondem à QP4: o treinamento federado é viável mesmo sob a heterogeneidade estatística de corredores veiculares distribuídos.

Segurança da informação em redes ITS com fatiamento inteligente (Capítulo 8 — QP5, D5): O Capítulo 8 investigou a segurança do classificador federado de impacto frente a ataques de envenenamento, por meio de 144 configurações experimentais que combinam três arquiteturas (DNN, LSTM, GRU), seis estratégias de agregação (FedAvg, FedProx, e cada uma combinada com Krum e Trimmed Mean), três fatores de amplificação adversarial (2×, 5×, 10×) e quatro configurações de épocas locais. Os resultados revelaram uma vulnerabilidade estrutural: com *gradient scaling* de fator 10×, a taxa de misclassificação *Critical*→*Adequate* (C2A Rate) atinge 100% para DNN FedAvg, significando que todas as fatias em violação de SLA são classificadas como adequadas. O FedProx não constitui defesa — ao contrário, amplifica a vulnerabilidade ao ancorar modelos honestos em um modelo global já contaminado. As defesas Bizantino-robustas mostraram-se eficazes: o Krum manteve C2A = 0% em todos os cenários com 5× e 10×, e o Trimmed Mean manteve C2A $\leq$ 1,66% em todas as 54 configurações da fase principal. Esses resultados respondem à QP5, demonstrando que a adoção de agregação robusta é requisito para implantações seguras de FL em cenários ITS com fatiamento inteligente.

Visão integrada: A Tabela 23 sintetiza a correspondência entre os desafios identificados, as questões de pesquisa, as contribuições e os principais resultados quantitativos.

Tabela 23 – Síntese das contribuições, questões de pesquisa e resultados quantitativos.

Desafio	QP	Contribuição (Capítulo)	Resultado principal
D1	QP1	Detecção de contexto — dados públicos (Cap. 3)	Viabilidade com dados abertos do Rio de Janeiro
D1	QP1	Predição de contexto — estatísticas de rede (Cap. 4)	Inferência de contexto via RTT/*throughput*
D2	QP2	Framework de fatiamento SDN (Cap. 5)	Atendimento dinâmico a 4 classes de aplicação ITS
D3	QP3	AIMS — classificação centralizada (Cap. 6)	CatBoost 94,8%; RF 94,4% de acurácia
D4	QP4	AIMS + FL — classificação federada (Cap. 7)	DNN-FedAvg Non-IID: 93,9% acc; gap $<$ 0,5 pp
D5	QP5	Segurança do FL (Cap. 8)	C2A=100% sem defesa; Krum C2A=0%; TM $\leq$ 1,66%

Os *datasets* e o código-fonte dos experimentos desta tese foram disponibilizados publicamente. A implementação do AIMS e seu *pipeline* de classificação de impacto estão acessíveis na cápsula Code Ocean (DOI: 10.24433/CO.5216953.v2) e no repositório https://github.com/saraivacode/AIMS; a infraestrutura federada e os experimentos de segurança estão em https://github.com/saraivacode/its-fl-slice-impact; e o framework de fatiamento SDN do Capítulo 5 está em https://github.com/saraivacode/framework_its_sdn. A disponibilização permite a reprodução dos resultados e facilita comparações por trabalhos futuros da comunidade.

9.2 Limitações

As contribuições desta tese apresentam limitações que devem ser reconhecidas.

Em primeiro lugar, os experimentos validam componentes específicos do *framework* — a classificação centralizada de impacto (Capítulo 6), a abordagem federada (Capítulo 7) e a resiliência a ataques (Capítulo 8) — mas não uma implementação *closed-loop* completa de ponta a ponta que cubra todas as fases do ciclo de vida em um *pipeline* integrado. A validação das funções F1–F7 do AIMS ocorreu de forma modular, e a cadeia completa desde a classificação de impacto até a reconfiguração efetiva dos recursos permanece não demonstrada experimentalmente.

Em segundo lugar, os experimentos dos Capítulos 6, 7 e 8 utilizam um único *dataset* de 5.400 amostras com 25 *features*, derivado de cenário ITS específico. Embora o protocolo de avaliação *leakage-free* e a validação cruzada garantam rigor interno, a generalização dos resultados para outros corredores veiculares, densidades de tráfego distintas ou configurações de rede diferentes não foi avaliada.

Em terceiro lugar, as abordagens de identificação de contexto veicular dos Capítulos 3 e 4 foram avaliadas em cenários específicos e não foram integradas diretamente aos experimentos dos capítulos subsequentes. As predições de contexto não alimentaram o *pipeline* de classificação de impacto do AIMS, o que limita a demonstração da narrativa de ponta a ponta.

Em quarto lugar, os experimentos de segurança do Capítulo 8 operam no regime de poucos participantes ($n = 3$, $f = 1$), utilizam exclusivamente ataques não adaptativos e avaliam apenas a direção de envenenamento *Critical*→*Adequate*. Cenários com maior número de clientes, ataques adaptativos (*AGR-tailored*) e outras direções de manipulação podem produzir resultados qualitativamente distintos.

Em quinto lugar, a eficiência computacional e energética dos modelos propostos não foi avaliada em dispositivos de borda reais. Os experimentos foram conduzidos em ambiente controlado, e restrições de memória, latência de inferência e consumo de energia em RSUs físicas constituem fatores que podem limitar a aplicabilidade prática das abordagens propostas.

9.3 Trabalhos futuros

Com base nas contribuições e limitações identificadas, as seguintes direções de trabalho futuro se apresentam:

1. **Validação *closed-loop* do *framework* AIMS:** implementar e avaliar o *pipeline* completo de gerenciamento do ciclo de vida das fatias, integrando as funções F1–F7

em um ambiente de emulação com cenários dinâmicos de ITS. A validação deve cobrir a cadeia completa desde a classificação de impacto até a reconfiguração efetiva dos recursos, incluindo a retroalimentação das decisões de adaptação sobre as métricas de telemetria monitoradas.

2. **Validação com *datasets* multi-fatia:** avaliar as abordagens propostas com *datasets* que contenham tráfego real segmentado por tipo de fatia 5G. O *dataset* 5G-NIDD, que contém tráfego eMBB, mMTC e URLLC, permitiria avaliar o impacto diferenciado dos ataques sobre cada classe de serviço e testar defesas orientadas pela criticidade da fatia.

3. **Avaliação com ataques adaptativos:** estender o diagnóstico de segurança com ataques *AGR-tailored* que otimizam a atualização maliciosa para minimizar a distância aos clientes honestos. Ataques adaptativos constituem o teste mais rigoroso para defesas Bizantino-robustas e representam a extensão natural do diagnóstico não adaptativo apresentado no Capítulo 8.

4. **Estratégias avançadas de FL para cenários Non-IID:** investigar técnicas de personalização federada, como agrupamento de clientes por similaridade de distribuição, estratégias de amostragem adaptativa e variantes do FedProx com ajuste dinâmico do parâmetro μ, visando melhorar o desempenho em cenários com maior heterogeneidade estatística entre RSUs e com maior número de participantes.

5. **Integração das abordagens de contexto veicular:** conectar as abordagens de detecção de contexto dos Capítulos 3 e 4 ao *pipeline* de classificação de impacto do AIMS, utilizando as predições de contexto como *features* adicionais para alimentar a fase de preparação das fatias.

6. **Eficiência computacional e implantação em borda:** avaliar o custo computacional e energético dos modelos propostos em dispositivos de borda reais, considerando restrições de memória, latência de inferência e consumo de energia que são críticos para implantações em RSUs.

REFERÊNCIAS

1 WU, W. et al. AI-Native Network Slicing for 6G Networks. *IEEE Wireless Communications*, IEEE, v. 29, n. 1, p. 96–103, 2022. ISSN 15580687. Citado 3 vezes nas páginas 20, 21 e 29.

2 DONATTI, A. et al. Survey on Machine Learning-Enabled Network Slicing: Covering the Entire Life Cycle. *IEEE Transactions on Network and Service Management*, v. 21, n. 1, p. 994–1011, 2024. ISSN 19324537. Citado 7 vezes nas páginas 20, 21, 28, 30, 31, 33 e 87.

3 YANG, T. et al. Liquid Software-Based Edge Intelligence for Future 6G Networks. *IEEE Network*, IEEE, n. August, p. 1–7, 2022. ISSN 0890-8044. Disponível em: https://ieeexplore.ieee.org/document/9696255/. Citado 3 vezes nas páginas 20, 36 e 87.

4 NGUYEN, D. C. et al. 6G Internet of Things: A Comprehensive Survey. *IEEE Internet of Things Journal*, IEEE, v. 9, n. 1, p. 359–383, 2022. ISSN 23274662. Citado 3 vezes nas páginas 20, 30 e 32.

5 XIAO, Y.; SHI, G.; KRUNZ, M. Towards Ubiquitous AI in 6G with Federated Learning. *arXiv preprint:2004.131563*, p. 1–7, apr 2020. ISSN 2331-8422. Disponível em: http://arxiv.org/abs/2004.13563. Citado 4 vezes nas páginas 20, 21, 29 e 36.

6 ABDENNOUR, R.; AHMED, T.; MOSBAH, M. End-to-End Network Slicing for ITS-G5 Vehicular Communications. In: *2021 International Wireless Communications and Mobile Computing (IWCMC)*. IEEE, 2021. p. 266–271. ISBN 978-1-7281-8616-0. Disponível em: https://ieeexplore.ieee.org/document/9498827/. Citado 2 vezes nas páginas 21 e 31.

7 DALGKITSIS, A. et al. Data Driven Service Orchestration for Vehicular Networks. *IEEE Transactions on Intelligent Transportation Systems*, IEEE, v. 22, n. 7, p. 4100–4109, 2021. ISSN 15580016. Citado na página 21.

8 do Vale Saraiva, T.; CAMPOS, C. A. V. Realistic Vehicular Networks Simulations. In: *Vehicular Ad-hoc Networks for Smart Cities*. [S.l.]: Springer, 2019, (Advances in Intelligent Systems and Computing). Citado 2 vezes nas páginas 24 e 26.

9 do Vale Saraiva, T.; CAMPOS, C. A. V. An approach using machine learning and public data to detect traffic jams. In: *2021 International Wireless Communications and Mobile Computing (IWCMC)*. IEEE, 2021. p. 675–680. ISBN 978-1-7281-8616-0. Disponível em: https://ieeexplore.ieee.org/document/9498622/. Citado 6 vezes nas páginas 24, 26, 39, 42, 52 e 53.

10 do Vale Saraiva, T.; GIRARDI, R.; CAMPOS, C. A. V. Machine learning applied to network traffic for vehicular applications to predict the vehicle context. In: *2021 Vehicular Technology Conference (VTC2021-Fall)*. [S.l.]: IEEE, 2021. ISBN 9781665413688. Citado 3 vezes nas páginas 25, 26 e 52.

11 SARAIVA, T. do V. et al. An application-driven framework for intelligent transportation systems using 5G network slicing. *IEEE Trans. Intell. Transp. Syst.*, v. 22, n. 8, p. 5247–5260, 2021. Citado 15 vezes nas páginas 25, 26, 56, 61, 66, 68, 75, 80, 82, 83, 84, 93, 109, 123 e 126.

12 do Vale Saraiva, T.; CAMPOS, C. A. V. AIMS: Adaptive and Intelligent Management of Slicing for Next-Generation ITS Networks. In: *31st IEEE Symposium on Computers and Communications (ISCC 2026)*. [S.l.: s.n.], 2026. Accepted for presentation. Citado 12 vezes nas páginas 25, 26, 86, 89, 96, 97, 98, 99, 100, 101, 122 e 126.

13 SARAIVA, T. do V.; CAMPOS, C. A. V. Federated learning-based scalable network slice impact classification for intelligent transportation systems. Manuscrito em preparação. 2026. Citado 4 vezes nas páginas 25, 26, 102 e 122.

14 SALAHDINE, F.; HAN, T.; ZHANG, N. 5G, 6G, and Beyond: Recent advances and future challenges. *Annales des Telecommunications/Annals of Telecommunications*, Annals of Telecommunications, 2023. ISSN 19589395. Citado na página 28.

15 LI, X. et al. Network Slicing for 5G: Challenges and Opportunities. *IEEE Internet Computing*, v. 21, n. 5, p. 20–27, 2017. ISSN 1089-7801. Disponível em: http://ieeexplore.ieee.org/document/8039298/. Citado 2 vezes nas páginas 28 e 65.

16 YANG, H. et al. Automatic guarantee scheme for intent-driven network slicing and reconfiguration. *Journal of Network and Computer Applications*, Elsevier Ltd, v. 190, n. July, p. 103163, 2021. ISSN 10958592. Disponível em: https://doi.org/10.1016/j.jnca.2021.103163. Citado na página 28.

17 LI, C. et al. Trustworthy Deep Learning in 6G-Enabled Mass Autonomy: From Concept to Quality-of-Trust Key Performance Indicators. *IEEE Vehicular Technology Magazine*, v. 15, n. 4, p. 112–121, 2020. ISSN 15566080. Citado 2 vezes nas páginas 29 e 31.

18 PANAYIOTOU, T. et al. Centralized and distributed machine learning-based QoT estimation for sliceable optical networks. *2019 IEEE Global Communications Conference, GLOBECOM 2019 - Proceedings*, 2019. Citado na página 29.

19 XIAO, Y. et al. Toward Self-Learning Edge Intelligence in 6G. *IEEE Communications Magazine*, v. 58, n. 12, p. 34–40, 2020. ISSN 15581896. Citado na página 30.

20 RIDHAWI, I. A. et al. Generalizing AI: Challenges and Opportunities for Plug and Play AI Solutions. *IEEE Network*, v. 35, n. 1, p. 372–379, jan 2021. ISSN 0890-8044. Disponível em: https://ieeexplore.ieee.org/document/9210129/. Citado 3 vezes nas páginas 30, 63 e 65.

21 SAMIR, R.; EL-HENNAWY, H.; ELBADAWY, H. Cluster-Based Multi-User Multi-Server Caching Mechanism in Beyond 5G/6G MEC. *Sensors*, v. 23, n. 2, p. 1–27, 2023. ISSN 14248220. Citado na página 31.

22 HE, J.; YANG, K.; CHEN, H. H. 6G Cellular Networks and Connected Autonomous Vehicles. *IEEE Network*, IEEE, v. 35, n. 4, p. 255–261, 2021. ISSN 1558156X. Citado na página 31.

23 SHETH, K. et al. A taxonomy of AI techniques for 6G communication networks. *Computer Communications*, Elsevier B.V., v. 161, n. May, p. 279–303, 2020. ISSN 1873703X. Disponível em: https://doi.org/10.1016/j.comcom.2020.07.035. Citado 2 vezes nas páginas 31 e 32.

24 BISHOP, C. M. *Pattern Recognition and Machine Learning*. New York, NY: Springer, 2006. ISBN 978-0-387-31073-2. Citado 2 vezes nas páginas 32 e 33.

25 Thanh Noi, P.; KAPPAS, M. Comparison of Random Forest, k-Nearest Neighbor, and Support Vector Machine Classifiers for Land Cover Classification Using Sentinel-2 Imagery. *Sensors (Basel, Switzerland)*, v. 18, n. 1, 2017. ISSN 14248220. Citado 2 vezes nas páginas 33 e 48.

26 PROKHORENKOVA, L. et al. Catboost: Unbiased boosting with categorical features. In: *Advances in Neural Information Processing Systems 31 (NeurIPS 2018)*. [S.l.: s.n.], 2018. Citado na página 33.

27 JAGANNATH, J. et al. Machine learning for wireless communications in the Internet of Things: A comprehensive survey. *Ad Hoc Networks*, Elsevier B.V., v. 93, p. 101913, 2019. ISSN 15708705. Disponível em: https://doi.org/10.1016/j.adhoc.2019.101913. Citado 2 vezes nas páginas 33 e 49.

28 GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep Learning*. [S.l.]: MIT Press, 2016. ISBN 978-0-262-03561-3. Citado na página 34.

29 HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural Computation*, v. 9, n. 8, p. 1735–1780, 1997. Citado na página 35.

30 CHO, K. et al. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv:1406.1078*, 2014. Citado na página 35.

31 CHUNG, J. et al. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014. Disponível em: http://arxiv.org/abs/1412.3555. Citado na página 36.

32 MCMAHAN, H. B. et al. Communication-Efficient Learning of Deep Networks from Decentralized Data. In: *20th International Conference on Artificial Intelligence and Statistics (AISTATS)*. [S.l.: s.n.], 2017. v. 54. Citado 8 vezes nas páginas 36, 37, 102, 107, 111, 120, 126 e 128.

33 LI, T. et al. Federated Optimization in Heterogeneous Networks. *Proceedings of Machine Learning and Systems*, v. 2, p. 429–450, 2020. Citado 4 vezes nas páginas 37, 102, 107 e 111.

34 LEE, R. et al. Classification of Urban Districts based on Road-centric Crowd Movements. In: *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. [S.l.: s.n.], 2018. v. 2018-September, p. 1782–1787. Citado 3 vezes nas páginas 39, 41 e 53.

35 LIANG, L.; YE, H.; LI, G. Y. Toward Intelligent Vehicular Networks: A Machine Learning Framework. *IEEE Internet of Things Journal*, v. 6, n. 1, p. 124–135, 2019. Citado 2 vezes nas páginas 40 e 42.

36 MTAWA, Y. A.; HAQUE, A.; BITAR, B. The Mammoth Internet: Are We Ready? *IEEE Access*, v. 7, p. 132894–132908, 2019. Citado na página 40.

37 Ericsson. *Ericsson Mobility Report, November 2025*. [S.l.], 2025. Disponível em: https://www.ericsson.com/en/reports-and-papers/mobility-report/reports/november-2025. Citado na página 40.

38 LIU, J. et al. A Scalable and Quick-Response Software Defined Vehicular Network Assisted by Mobile Edge Computing. *IEEE Communications Magazine*, v. 55, n. 7, p. 94–100, 2017. ISSN 0163-6804. Disponível em: http://ieeexplore.ieee.org/document/7981532/. Citado 6 vezes nas páginas 40, 56, 62, 63, 65 e 75.

39 RASHID, B.; REHMANI, M. H. Applications of wireless sensor networks for urban areas: A survey. *Journal of Network and Computer Applications*, v. 60, p. 192–219, 2016. Citado na página 40.

40 JABBARPOUR, M. R. et al. Applications of computational intelligence in vehicle traffic congestion problem: a survey. *Soft Computing*, v. 22, n. 7, p. 2299–2320, 2018. Citado 2 vezes nas páginas 40 e 42.

41 THAKUR, A.; MALEKIAN, R. Fog Computing for Detecting Vehicular Congestion, an Internet of Vehicles Based Approach: A Review. *IEEE Intelligent Transportation Systems Magazine*, v. 11, n. 2, p. 8–16, 2019. Citado 3 vezes nas páginas 40, 41 e 42.

42 KULKARNI, N. et al. Averaging Based Predictive Modelling for Traffic Congestion in IoT. In: *Proceedings — 2018 IEEE Global Conference on Wireless Computing and Networking (GCWCN)*. [S.l.: s.n.], 2019. p. 49–53. Citado 2 vezes nas páginas 40 e 41.

43 YE, H. et al. Machine Learning for Vehicular Networks: Recent Advances and Application Examples. *IEEE Vehicular Technology Magazine*, v. 13, n. 2, p. 94–101, jun. 2018. Citado 3 vezes nas páginas 40, 42 e 52.

44 YAZAR, A.; ARSLAN, H. Selection of Waveform Parameters Using Machine Learning for 5G and beyond. In: *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. [S.l.: s.n.], 2019. v. 2019-September. Citado 2 vezes nas páginas 40 e 53.

45 MA, X. et al. Large-scale transportation network congestion evolution prediction using deep learning theory. *PLoS ONE*, v. 10, n. 3, p. 1–17, 2015. Citado na página 41.

46 PYRGELIS, A.; CRISTOFARO, E. D.; ROSS, G. J. Privacy-friendly mobility analytics using aggregate location data. In: *GIS: Proceedings of the ACM International Symposium on Advances in Geographic Information Systems*. [S.l.: s.n.], 2016. Citado na página 41.

47 NING, Z. et al. Vehicular Social Networks: Enabling Smart Mobility. *IEEE Communications Magazine*, v. 55, n. 5, p. 49–55, 2017. Citado na página 41.

48 MTAWA, Y. A.; HASSANEIN, H. S.; NASSER, N. The impact of anchor misplacement on sensing coverage. In: *IEEE Wireless Communications and Networking Conference (WCNC)*. [S.l.: s.n.], 2016. v. 2016-September. Citado na página 41.

49 CHEN, Y. et al. Node localization algorithm of wireless sensor networks with mobile beacon node. *Peer-to-Peer Networking and Applications*, v. 10, n. 3, p. 795–807, 2017. Citado na página 41.

50 DENIS, S. et al. Large Scale Crowd Density Estimation Using a sub-GHz Wireless Sensor Network. In: *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. [S.l.: s.n.], 2018. v. 2018-September, p. 849–855. Citado 2 vezes nas páginas 41 e 53.

51 INOUE, R.; MIYASHITA, A.; SUGITA, M. Mining spatio-Temporal patterns of congested traffic in urban areas from traffic sensor data. In: *IEEE Conference on Intelligent Transportation Systems, Proceedings (ITSC)*. [S.l.: s.n.], 2016. p. 731–736. Citado na página 41.

52 NGUYEN, H.; LIU, W.; CHEN, F. Discovering Congestion Propagation Patterns in Spatio-Temporal Traffic Data. *IEEE Transactions on Big Data*, v. 3, n. 2, p. 169–180, 2016. Citado na página 41.

53 MTAWA, Y. A.; HASSANEIN, H. S.; NASSER, N. Measuring the validity of sensing coverage in the presence of anchor misplacement. In: *2017 IEEE International Conference on Communications (ICC)*. [S.l.]: IEEE, 2017. p. 1–6. Citado na página 41.

54 SHEN, D. et al. Forecasting Citywide Traffic Congestion Based on Social Media. *Wireless Personal Communications*, v. 103, n. 1, p. 1037–1057, 2018. Citado na página 41.

55 DJENOURI, Y. et al. A Survey on Urban Traffic Anomalies Detection Algorithms. *IEEE Access*, v. 7, p. 12192–12205, 2019. Citado na página 42.

56 FAN, G. et al. Emerging Trends in Graph Neural Networks for Traffic Flow Prediction: A Survey. *Archives of Computational Methods in Engineering*, v. 32, p. 4811–4855, 2025. Citado na página 42.

57 AFANDIZADEH, S. et al. Deep Learning Algorithms for Traffic Forecasting: A Comprehensive Review and Comparison with Classical Ones. *Journal of Advanced Transportation*, v. 2024, p. 9981657, 2024. Citado na página 42.

58 BRIMOS, P. et al. Advances in Traffic Congestion Prediction: An Overview of Emerging Techniques and Methods. *Smart Cities*, v. 8, n. 1, p. 25, 2025. Citado 2 vezes nas páginas 42 e 50.

59 MURTFELDT, R. et al. RIP Twitter API: A Eulogy to Its Vast Research Contributions. *arXiv preprint*, 2024. ArXiv:2404.07340. Citado na página 42.

60 TAWFIK, A.; IBRAHIM, M.; BARAKAT, T. M. A Survey on Beyond 5G Network Slicing for Smart Cities Applications. *IEEE Communications Surveys & Tutorials*, 2024. Citado na página 44.

61 Praveen Kumar, D.; AMGOTH, T.; ANNAVARAPU, C. S. R. Machine learning algorithms for wireless sensor networks: A survey. *Information Fusion*, Elsevier B.V., v. 49, n. April 2018, p. 1–25, 2019. ISSN 15662535. Disponível em: https://doi.org/10.1016/j.inffus.2018.09.013. Citado na página 48.

62 ALI, J. et al. Random Forests and Decision Trees. *International Journal of Computer Science Issues*, v. 9, n. 5, p. 272–278, 2012. ISSN 1694-0784. Citado na página 49.

63 POORNIMA, I. G. A.; PARAMASIVAN, B. Anomaly detection in wireless sensor network using machine learning algorithm. *Computer Communications*, Elsevier B.V., v. 151, n. August 2019, p. 331–337, 2020. ISSN 1873703X. Disponível em: https://doi.org/10.1016/j.comcom.2020.01.005. Citado na página 49.

64 WANG, Z. et al. A Survey on Human Behavior Recognition Using Channel State Information. *IEEE Access*, IEEE, v. 7, n. October, p. 155986–156024, 2019. ISSN 21693536. Citado na página 49.

65 M, H.; M.N, S. A Review on Evaluation Metrics for Data Classification Evaluations. *International Journal of Data Mining & Knowledge Management Process*, v. 5, n. 2, p. 01–11, mar 2015. ISSN 2231007X. Disponível em: http://www.aircconline.com/ijdkp/V5N2/5215ijdkp01.pdf. Citado na página 49.

66 CHEN, W. et al. A survey on imbalanced learning: latest research, applications and future directions. *Artificial Intelligence Review*, v. 57, p. 119, 2024. Citado na página 50.

67 KARA, M. M.; TURKMEN, H. I.; GUVENSAN, M. A. Smart Organization of Imbalanced Traffic Datasets for Long-Term Traffic Forecasting. *Sensors*, v. 25, n. 4, p. 1225, 2025. Citado na página 50.

68 NUHA, H. H. et al. Ensemble Machine Learning Approach for Traffic Congestion and Travel Time Prediction in Urban Bus Rapid Transit Systems: A Case Study of Trans Metro Bandung. *IoT*, v. 7, n. 1, p. 22, 2026. Citado na página 50.

69 MOLINA-MASEGOSA, R.; GOZALVEZ, J. LTE-V for sidelink 5G V2X vehicular communications: A new 5G technology for short-range vehicle-to-everything communications. *IEEE Vehicular Technology Magazine*, v. 12, n. 4, p. 30–39, dez. 2017. Citado 2 vezes nas páginas 52 e 63.

70 DUAN, X.; LIU, Y.; WANG, X. SDN Enabled 5G-VANET: Adaptive Vehicle Clustering and Beamformed Transmission for Aggregated Traffic. *IEEE Communications Magazine*, v. 55, n. 7, p. 120–127, 2017. ISSN 0163-6804. Disponível em: http://ieeexplore.ieee.org/document/7981536/. Citado 4 vezes nas páginas 52, 54, 57 e 78.

71 YAQOOB, I. et al. Overcoming the key challenges to establishing vehicular communication: Is SDN the answer? *IEEE Communications Magazine*, v. 55, n. 7, p. 128–134, 2017. Citado na página 52.

72 SEçINTI, G. et al. Software defined architecture for VANET: A testbed implementation with wireless access management. *IEEE Communications Magazine*, v. 55, n. 7, p. 135–141, 2017. Citado na página 52.

73 LI, R. et al. Intelligent 5G: When Cellular Networks Meet Artificial Intelligence. *IEEE Wireless Communications*, v. 24, n. 5, p. 175–183, oct 2017. ISSN 1536-1284. Disponível em: http://ieeexplore.ieee.org/document/7886994/. Citado na página 52.

74 PALAIOS, A. et al. Machine learning for QoS prediction in vehicular communication: Challenges and solution approaches. *IEEE Access*, v. 11, p. 92459–92477, 2023. Citado 2 vezes nas páginas 52 e 54.

75 MOURA, H. et al. Estimating quality of service on Wi-Fi stations using recurrent neural networks. In: *Proc. IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. [S.l.: s.n.], 2019. Citado na página 53.

76 TRINH, H. D.; GIUPPONI, L.; DINI, P. Mobile traffic prediction from raw data using LSTM networks. In: *Proc. IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*. [S.l.: s.n.], 2018. p. 1827–1832. Citado na página 53.

77 VAHDAT-NEJAD, H. et al. A survey on context-aware vehicular network applications. *Vehicular Communications*, v. 3, p. 43–57, 2016. Citado na página 53.

78 CHAVHAN, S. et al. IoT-based context-aware intelligent public transport system in a metropolitan area. *IEEE Internet of Things Journal*, v. 7, n. 7, p. 6023–6034, 2020. Citado na página 53.

79 KULKARNI, N. et al. Averaging based predictive modelling for traffic congestion in IoT. In: *Proc. IEEE Global Conference on Wireless Computing and Networking (GCWCN)*. [S.l.: s.n.], 2018. p. 49–53. Citado na página 53.

80 TAHERKHANI, N.; PIERRE, S. Centralized and localized data congestion control strategy for vehicular ad hoc networks using a machine learning clustering algorithm. *IEEE Transactions on Intelligent Transportation Systems*, v. 17, n. 11, p. 3275–3285, 2016. Citado na página 53.

81 POPOOLA, S. I. et al. Hybrid deep learning for botnet attack detection in the Internet-of-Things networks. *IEEE Internet of Things Journal*, v. 8, n. 6, p. 4944–4956, 2021. Citado na página 54.

82 ZHAO, R. et al. An efficient intrusion detection method based on dynamic autoencoder. *IEEE Wireless Communications Letters*, v. 10, n. 8, p. 1707–1711, 2021. Citado na página 54.

83 HAMDI, W. et al. Network Slicing-Based Learning Techniques for IoV in 5G and Beyond Networks. *IEEE Communications Surveys & Tutorials*, v. 26, n. 3, p. 1989–2047, 2024. Citado 4 vezes nas páginas 54, 64, 87 e 90.

84 NURKAHFI, G. et al. On SDN to support the IEEE 802.11 and C-V2X-based vehicular communications use-cases and performance: A comprehensive survey. *IEEE Access*, v. 12, p. 95926–95958, 2024. Citado na página 54.

85 DARWISH, T.; Abu Bakar, K. Traffic density estimation in vehicular ad hoc networks: A review. *Ad Hoc Networks*, Elsevier B.V., v. 24, n. PA, p. 337–351, 2015. ISSN 15708705. Disponível em: http://dx.doi.org/10.1016/j.adhoc.2014.09.007. Citado na página 55.

86 FONTES, R. R. et al. Mininet-WiFi: Emulating software-defined wireless networks. In: *2015 11th International Conference on Network and Service Management (CNSM)*. IEEE, 2015. p. 384–389. ISBN 978-3-9018-8277-7. Disponível em: http://ieeexplore.ieee.org/document/7367387/. Citado 2 vezes nas páginas 56 e 73.

87 LANTZ, B.; HELLER, B.; MCKEOWN, N. A network in a laptop. In: *Proceedings of the Ninth ACM SIGCOMM Workshop on Hot Topics in Networks - Hotnets '10*. New York, New York, USA: ACM Press, 2010. p. 1–6. ISBN 9781450304092. ISSN 1450304095.

Disponível em: http://portal.acm.org/citation.cfm?doid=1868447.1868466. Citado 2 vezes nas páginas 56 e 73.

88 KUHLMORGEN, S. et al. Evaluation of Congestion-Enabled Forwarding with Mixed Data Traffic in Vehicular Communications. *IEEE Transactions on Intelligent Transportation Systems*, v. 21, n. 1, p. 233–247, 2020. ISSN 15580016. Citado 2 vezes nas páginas 56 e 75.

89 KRAJZEWICZ, D. et al. Recent Development and Applications of SUMO - Simulation of Urban MObility. *International Journal On Advances in Systems and Measurements*, v. 5, n. 3-4, p. 128–138, 2012. Disponível em: http://sumo.sourceforge.net/. Citado 3 vezes nas páginas 56, 62 e 75.

90 LI, H.; DONG, M.; OTA, K. Control Plane Optimization in Software-Defined Vehicular Ad Hoc Networks. *IEEE Transactions on Vehicular Technology*, v. 65, n. 10, p. 7895–7904, oct 2016. ISSN 0018-9545. Disponível em: http://ieeexplore.ieee.org/document/7464864/. Citado 2 vezes nas páginas 56 e 75.

91 AHMAD, E. et al. Location-Aware, Context-Driven QoS for IoT Applications. *IEEE Systems Journal*, p. 1–12, 2019. ISSN 1932-8184. Disponível em: https://ieeexplore.ieee.org/document/8640087/. Citado 2 vezes nas páginas 57 e 77.

92 SAHOO, P. K.; YUNHASNAWA, Y. Ferrying vehicular data in cloud through software defined networking. In: *2016 IEEE 12th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob)*. IEEE, 2016. p. 1–8. ISBN 978-1-5090-0724-0. ISSN 21619654. Disponível em: http://ieeexplore.ieee.org/document/7763223/. Citado 2 vezes nas páginas 57 e 78.

93 BENCHAJI, I. et al. Deep learning on network traffic prediction: Recent advances, analysis, and future directions. *ACM Computing Surveys*, v. 57, n. 4, p. 1–40, 2024. Citado na página 58.

94 SYRIGOS, I. et al. On the Employment of Machine Learning Techniques for Troubleshooting WiFi Networks. *2019 16th IEEE Annual Consumer Communications and Networking Conference, CCNC 2019*, IEEE, p. 1–6, 2019. Citado na página 59.

95 HE, Z. et al. Edge device identification based on federated learning and network traffic feature engineering. *IEEE Transactions on Cognitive Communications and Networking*, 2021. Citado na página 60.

96 ZHANG, J. et al. A survey on federated learning in intelligent transportation systems. *IEEE Internet of Things Journal*, v. 11, n. 21, p. 34553–34576, 2024. Citado na página 60.

97 ZHANG, Y.; XU, G.; YANG, X. Federated learning in intelligent transportation systems: Recent applications and open problems. *IEEE Transactions on Intelligent Transportation Systems*, v. 25, n. 4, p. 3259–3280, 2024. Citado na página 60.

98 Dos Reis Fontes, R. et al. From Theory to Experimental Evaluation: Resource Management in Software-Defined Vehicular Networks. *IEEE Access*, v. 5, p. 3069–3076, 2017. ISSN 2169-3536. Disponível em: http://ieeexplore.ieee.org/document/7859348/. Citado 5 vezes nas páginas 61, 62, 64, 65 e 77.

99 ZHANG, L.; VALAEE, S. Congestion Control for Vehicular Networks With Safety-Awareness. *IEEE/ACM Transactions on Networking*, v. 24, n. 6, p. 3290–3299, dec 2016. ISSN 1063-6692. Disponível em: http://ieeexplore.ieee.org/document/7399410/. Citado na página 62.

100 TEGUEU, F. S. et al. Towards application driven networking. In: *2016 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*. IEEE, 2016. v. 2016-Augus, p. 1–6. ISBN 978-1-4673-9882-4. ISSN 19440375. Disponível em: http://ieeexplore.ieee.org/document/7548865/. Citado 2 vezes nas páginas 62 e 65.

101 CORREIA, S.; BOUKERCHE, A.; MENEGUETTE, R. I. An Architecture for Hierarchical Software-Defined Vehicular Networks. *IEEE Communications Magazine*, v. 55, n. 7, p. 80–86, 2017. ISSN 0163-6804. Disponível em: http://ieeexplore.ieee.org/document/7981530/. Citado 2 vezes nas páginas 62 e 65.

102 ZHENG, K. et al. Soft-defined heterogeneous vehicular network: architecture and challenges. *IEEE Network*, v. 30, n. 4, p. 72–80, jul 2016. ISSN 0890-8044. Disponível em: http://ieeexplore.ieee.org/document/7513867/. Citado 2 vezes nas páginas 62 e 65.

103 PETROV, V. et al. Achieving End-to-End Reliability of Mission-Critical Traffic in Softwarized 5G Networks. *IEEE Journal on Selected Areas in Communications*, v. 36, n. 3, p. 485–501, 2018. ISSN 07338716. Citado 2 vezes nas páginas 62 e 65.

104 KAUL, A. et al. Dynamically distributed network control for message dissemination in ITS. In: *2017 IEEE/ACM 21st International Symposium on Distributed Simulation and Real Time Applications (DS-RT)*. IEEE, 2017. p. 1–9. ISBN 978-1-5386-4028-9. Disponível em: http://ieeexplore.ieee.org/document/8167677/. Citado 2 vezes nas páginas 62 e 65.

105 HUANG, C.; LU, R.; CHOO, K.-K. R. Vehicular Fog Computing: Architecture, Use Case, and Security and Forensic Challenges. *IEEE Communications Magazine*, v. 55, n. 11, p. 105–111, nov 2017. ISSN 0163-6804. Disponível em: http://ieeexplore.ieee.org/document/8114558/. Citado 2 vezes nas páginas 62 e 65.

106 HAKLAY, M.; WEBER, P. OpenStreetMap: User-Generated Street Maps. *IEEE Pervasive Computing*, v. 7, n. 4, p. 12–18, oct 2008. ISSN 1536-1268. Disponível em: http://ieeexplore.ieee.org/document/4653466/. Citado na página 62.

107 KHAN, A. A.; ABOLHASAN, M.; NI, W. 5G next generation VANETs using SDN and fog computing framework. In: *2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2018. p. 1–6. ISBN 978-1-5386-4790-5. Disponível em: http://ieeexplore.ieee.org/document/8319192/. Citado 2 vezes nas páginas 63 e 65.

108 TAO, M. et al. Vehicular Data Cloud Platform with 5G Support: Architecture, Services, and Challenges. In: *22017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC)*. IEEE, 2017. v. 1, p. 32–37. ISBN 978-1-5386-3220-8. Disponível em: http://ieeexplore.ieee.org/document/8005769/. Citado 2 vezes nas páginas 63 e 65.

109 CHEKIRED, D. A. et al. 5G-Slicing-Enabled Scalable SDN Core Network: Toward an Ultra-Low Latency of Autonomous Driving Service. *IEEE Journal on Selected Areas in Communications*, v. 37, n. 8, p. 1769–1782, 2019. ISSN 0733-8716. Citado 3 vezes nas páginas 63, 64 e 65.

110 BALASUBRAMANIAN, V. et al. Low-latency vehicular edge: A vehicular infrastructure model for 5G. *Simulation Modelling Practice and Theory*, Elsevier, v. 98, n. August 2019, p. 101968, jan 2020. ISSN 1569190X. Disponível em: https://doi.org/10.1016/j.simpat.2019.101968https://linkinghub.elsevier.com/retrieve/pii/S1569190X19301017. Citado 2 vezes nas páginas 63 e 65.

111 ITU-R. *Framework and Overall Objectives of the Future Development of IMT for 2030 and Beyond*. 2023. Recommendation ITU-R M.2160. Disponível em: https://www.itu.int/rec/R-REC-M.2160-0-202311-I/en. Citado na página 63.

112 LANG, P. et al. Towards 6G vehicular networks: Vision, technologies, and open challenges. *Computer Networks*, v. 257, p. 110916, 2025. Citado na página 63.

113 ZHAO, Z. et al. Mobility prediction-assisted over-the-top edge prefetching for hierarchical VANETs. *IEEE Journal on Selected Areas in Communications*, v. 36, n. 8, p. 1786–1801, 2018. ISSN 15580008. Citado na página 63.

114 MARQUEZ, C. et al. How Should I Slice My Network ? A Multi-Service Empirical Evaluation of Resource Sharing Efficiency. In: *MobiCom '18 Proceedings of the 24th Annual International Conference on Mobile Computing and Networking*. New Delhi, India: [s.n.], 2018. p. 191–206. Citado 2 vezes nas páginas 64 e 65.

115 CUI, Y. et al. Multi-agent reinforcement learning for slicing resource allocation in vehicular networks. *IEEE Transactions on Intelligent Transportation Systems*, v. 25, n. 2, p. 2005–2016, 2024. Citado 2 vezes nas páginas 64 e 85.

116 TRAKADAS, P. et al. QoS-Driven Slicing Management for Vehicular Communications. *Electronics*, MDPI, v. 13, n. 2, p. 314, 2024. Citado 2 vezes nas páginas 64 e 85.

117 EBRAHIMI, S.; BOUALI, F.; HAAS, O. C. L. Resource management from single-domain 5G to end-to-end 6G network slicing: A survey. *IEEE Communications Surveys & Tutorials*, v. 26, n. 4, p. 2836–2866, 2024. Citado na página 64.

118 SHAH, S. A. A. et al. 5G for Vehicular Communications. *IEEE Communications Magazine*, v. 56, n. 1, p. 111–117, jan 2018. ISSN 0163-6804. Disponível em: http://ieeexplore.ieee.org/document/8255748/. Citado na página 65.

119 Open Networking Foundation. *SDN Architecture 1.1*. 2016. ONF TR-521. Disponível em: https://www.opennetworking.org/wp-content/uploads/2014/10/TR-521_SDN_Architecture_issue_1.1.pdf. Citado na página 65.

120 ETSI. *Network Functions Virtualisation (NFV); Management and Orchestration*. 2017. ETSI GS NFV-MAN 001 V1.1.1. Disponível em: http://www.etsi.org/deliver/etsi_gs/NFV-MAN/001_099/001/01.01.01_60/gs_NFV-MAN001v010101p.pdf. Citado na página 70.

121 TALEB, T. et al. On Multi-Access Edge Computing: A Survey of the Emerging 5G Network Edge Cloud Architecture and Orchestration. *IEEE Communications Surveys & Tutorials*, v. 19, n. 3, p. 1657–1681, 2017. ISSN 1553-877X. Disponível em: http://ieeexplore.ieee.org/document/7931566/. Citado na página 70.

122 OpenBaton. *OpenBaton: An open source reference implementation of the ETSI NFV MANO specification*. 2020. Online. Disponível em: https://openbaton-docs.readthedocs.io/en/latest/. Citado na página 70.

123 DALGITSIS, M. et al. Cloud-native orchestration framework for network slice federation across administrative domains in 5G/6G mobile networks. *IEEE Transactions on Vehicular Technology*, v. 73, n. 7, p. 9306–9319, 2024. Citado na página 70.

124 CZIVA, R. et al. SDN-Based Virtual Machine Management for Cloud Data Centers. *IEEE Transactions on Network and Service Management*, v. 13, n. 2, p. 212–225, jun 2016. ISSN 1932-4537. Disponível em: http://ieeexplore.ieee.org/document/7416246/. Citado na página 74.

125 AL-SULTAN, S.; AL-BAYATTI, A. H.; ZEDAN, H. Context-Aware Driver Behavior Detection System in Intelligent Transportation Systems. *IEEE Transactions on Vehicular Technology*, v. 62, n. 9, p. 4264–4275, nov 2013. ISSN 0018-9545. Disponível em: http://ijsetr.com/uploads/325164IJSETR4774-488.pdfhttp://ieeexplore.ieee.org/document/6516983/. Citado na página 76.

126 WANG, Y.; JIANG, J.; MU, T. Context-Aware and Energy-Driven Route Optimization for Fully Electric Vehicles via Crowdsourcing. *IEEE Transactions on Intelligent Transportation Systems*, v. 14, n. 3, p. 1331–1345, sep 2013. ISSN 1524-9050. Disponível em: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=6516898. Citado na página 76.

127 SHAFI, M. et al. 5G: A tutorial overview of standards, trials, challenges, deployment, and practice. *IEEE Journal on Selected Areas in Communications*, v. 35, n. 6, p. 1201–1221, 2017. ISSN 07338716. Citado na página 77.

128 SINGH, R. et al. Coexistence of railway and road services by sharing telecommunication infrastructure using SDN-based slicing: A tutorial. *Network*, MDPI, v. 2, n. 4, p. 670–706, 2022. Citado na página 77.

129 NARDINI, G. et al. Simu5G – an OMNeT++ library for end-to-end performance evaluation of 5G networks. *IEEE Access*, v. 8, p. 181176–181191, 2020. Citado na página 77.

130 SAID, A. M. et al. Modeling interactive real-time applications in VANETs with performance evaluation. *Computer Networks*, Elsevier B.V., v. 104, p. 66–78, jul 2016. ISSN 13891286. Disponível em: http://dx.doi.org/10.1016/j.comnet.2016.04.021https://linkinghub.elsevier.com/retrieve/pii/S1389128616301207. Citado na página 78.

131 Hashem Eiza, M. et al. Situation-Aware QoS Routing Algorithm for Vehicular Ad Hoc Networks. *IEEE Transactions on Vehicular Technology*, v. 64, n. 12, p. 5520–5535, dec 2015. ISSN 0018-9545. Disponível em: http://ieeexplore.ieee.org/document/7286854/. Citado na página 78.

132 LIU, R. et al. The ITU Vision and Framework for 6G: Scenarios, Capabilities, and Enablers. *IEEE Vehicular Technology Magazine*, v. 20, n. 2, p. 114–122, jun. 2025. Citado na página 86.

133 3GPP. *Management and orchestration; Concepts, use cases and requirements (Release 19)*. 2025. Technical Specification (TS) 28.530, Version 19.0.0. Citado 3 vezes nas páginas 86, 103 e 123.

134 De Alwis, C. et al. A Survey on Network Slicing Security: Attacks, Challenges, Solutions, and Research Directions. *IEEE Communications Surveys & Tutorials*, v. 26, n. 1, p. 534–570, 2024. Citado 3 vezes nas páginas 87, 103 e 121.

135 GUAN, W.; ZHANG, H.; LEUNG, V. C. M. Customized Slicing for 6G: Enforcing Artificial Intelligence on Resource Management. *IEEE Network*, v. 35, n. 5, p. 264–271, 2021. Citado na página 87.

136 VILA, I. et al. A Multi-Agent Reinforcement Learning Approach for Capacity Sharing in Multi-Tenant Scenarios. *IEEE Transactions on Vehicular Technology*, v. 70, n. 9, p. 9450–9465, 2021. Citado na página 87.

137 MEI, J.; WANG, X.; ZHENG, K. Semi-Decentralized Network Slicing for Reliable V2V Service Provisioning: A Model-Free Deep Reinforcement Learning Approach. *IEEE Transactions on Intelligent Transportation Systems*, v. 23, n. 8, p. 12108–12120, 2022. Citado na página 87.

138 SUN, H. et al. Advancing 6G: Survey for explainable AI on communications and network slicing. *IEEE Open Journal of the Communications Society*, v. 6, p. 1372–1412, 2025. Citado na página 87.

139 DANGI, R.; LALWANI, P. Optimizing network slicing in 6G networks through a hybrid deep learning strategy. *Journal of Supercomputing*, v. 80, n. 14, p. 20400–20420, 2024. Citado na página 87.

140 WU, W. et al. AI-Native Network Slicing for 6G Networks. *IEEE Wireless Communications*, v. 29, n. 1, p. 96–103, fev. 2022. Citado na página 87.

141 HABIBI, M. A. et al. Toward an Open, Intelligent, and End-to-End Architectural Framework for Network Slicing in 6G Communication Systems. *IEEE Open Journal of the Communications Society*, v. 4, p. 1615–1658, 2023. Citado na página 87.

142 WU, W. et al. AI-Native Network Digital Twin for Intelligent Network Management in 6G. *IEEE Network*, 2025. Early access. Citado na página 87.

143 WANG, J. et al. Artificial Intelligence-Assisted Network Slicing, Network Assurance, and Service Provisioning in 6G Communication Systems. *IEEE Vehicular Technology Magazine*, v. 18, n. 1, p. 49–58, 2023. Citado na página 87.

144 WANG, J. et al. Intelligent Network Slicing for B5G and 6G: Resource Allocation, Service Provisioning, and Security. *IEEE Wireless Communications*, v. 31, n. 2, p. 271–277, 2024. Citado na página 87.

145 HAFI, H. et al. Split Federated Learning for 6G Enabled-Networks : Requirements , Challenges , and Future Directions. *IEEE Access*, IEEE, v. 12, n. January, p. 9890–9930, 2024. Citado na página 101.

146 ZHANG, X. et al. Federated Learning-Assisted Vehicular Edge Computing: Architecture and Research Directions. *IEEE Vehicular Technology Magazine*, v. 18, n. 4, p. 75–84, dez. 2023. Citado na página 103.

147 DONATTI, A. W. et al. Energy efficiency in network slicing: Survey and taxonomy. *IEEE Access*, IEEE, v. 13, p. 134570–134589, 2025. Citado na página 103.

148 KRISHNAMOORTHY, R. et al. Federated learning for dynamic resource allocation in 6g network slicing. *Wireless Personal Communications*, Springer, 2026. Citado na página 103.

149 LEE, J. et al. Federated learning-empowered mobile network management for 5g and beyond networks: From access to core. *IEEE Communications Surveys & Tutorials*, v. 26, n. 3, p. 2176–2212, 2024. Citado na página 103.

150 HAMDI, W. et al. Network slicing-based learning techniques for IoV in 5G and beyond networks. *IEEE Communications Surveys & Tutorials*, v. 26, n. 3, p. 1989–2047, 2024. Citado na página 103.

151 CHELLAPANDI, V. P. et al. Federated Learning for Connected and Automated Vehicles: A Survey of Existing Approaches and Challenges. *IEEE Transactions on Intelligent Vehicles*, v. 9, n. 1, p. 119–135, jan. 2024. Citado 3 vezes nas páginas 103, 107 e 119.

152 ROY, S.; CHERGUI, H.; VERIKOUKIS, C. Explanation-Guided Fair Federated Learning for Transparent 6G RAN Slicing. *IEEE Transactions on Mobile Computing*, 2024. Citado na página 103.

153 ABOUAOMAR, A. et al. Federated Deep Reinforcement Learning for Open RAN Slicing in 6G Networks. *IEEE Transactions on Network and Service Management*, v. 20, n. 4, p. 4080–4094, dez. 2023. Citado na página 103.

154 GUPTA, A. et al. Federated Reinforcement Learning for Collaborative Intelligence in UAV and V2X-Assisted Networks: Architecture, Applications, Opportunities, and Open Challenges. *IEEE Network*, v. 37, n. 6, p. 274–282, 2023. Citado na página 103.

155 QIANG, X. et al. Split Federated Learning Empowered Vehicular Edge Intelligence: Concept, Adaptive Design and Future Directions. *arXiv preprint arXiv:2406.15804*, 2024. Citado na página 103.

156 BAGANAL-KRISHNA, N. V.; KIM, S.; CHOI, J. Y. Federated Learning-Based QoS Prediction for IoT Services. In: *Proc. IEEE Int. Conf. Consumer Electronics (ICCE)*. [S.l.: s.n.], 2024. p. 1–6. Citado na página 103.

157 THAM, C. K.; LUO, T.; LUI, J. C. S. Federated Learning for Cybersecurity in Vehicular Networks. In: *Proc. IEEE Vehicular Technology Conf. (VTC-Fall)*. [S.l.: s.n.], 2023. p. 1–5. Citado na página 103.

158 GUO, S. et al. EXPLORA: Aiml explainability for the open ran. *IEEE Transactions on Network and Service Management*, IEEE, 2025. Citado 2 vezes nas páginas 106 e 117.

159 GUPTA, S. et al. FL and explainable AI framework for privacy-preserving diagnosis. *IEEE Access*, IEEE, v. 14, p. 22638–22650, 2026. Citado na página 106.

160 CHEN, X.; ZHOU, M. et al. MoReCS: Mobility-aware reputation-based vehicle client selection scheme for hierarchical federated learning. *IEEE Transactions on Vehicular Technology*, IEEE, 2025. Citado na página 106.

161 WANG, Z. et al. FedRAV: Hierarchically federated region-learning framework of autonomous vehicles. *IEEE Transactions on Intelligent Vehicles*, IEEE, 2025. Citado na página 106.

162 LI, Q. et al. FedTree: A federated learning system for trees. In: *Proceedings of Machine Learning and Systems*. [S.l.: s.n.], 2023. Citado na página 107.

163 ZHAO, Y. et al. Communication-efficient federated learning for decision trees. *IEEE Transactions on Artificial Intelligence*, 2024. Citado na página 107.

164 BEUTEL, D. J. et al. *Flower: A Friendly Federated Learning Research Framework*. 2022. Citado na página 108.

165 RIEDEL, P. et al. Comparative analysis of open-source federated learning frameworks — a literature-based survey and review. *International Journal of Machine Learning and Cybernetics*, v. 15, p. 5257–5278, 2024. Citado na página 108.

166 BAUMGART, G. A.; LEE, M. et al. Not all federated learning algorithms are created equal: A performance evaluation study. *arXiv preprint*, 2024. ArXiv:2403.17287. Citado na página 113.

167 LI, Q. et al. Federated learning on non-IID data silos: An experimental study. In: *IEEE International Conference on Data Engineering (ICDE)*. [S.l.: s.n.], 2022. p. 965–978. Citado na página 113.

168 ALMANSOUR, N. et al. Adaptive personalized federated learning with lightweight depthwise convolutional bottleneck network for novel intrusion detection system in internet of vehicles. *Scientific Reports*, v. 15, p. 17699, 2025. Citado na página 117.

169 RIZWAN, A. et al. BHFVAL: Blockchain-enabled hierarchical federated variational autoencoder framework for secure intrusion detection in vehicular networks. *Scientific Reports*, v. 15, p. 28560, 2025. Citado na página 117.

170 ZHANG, Y. et al. Federated two-stage transformer-based network for intrusion detection in non-IID data of controller area networks. *Cybersecurity*, v. 8, p. 29, 2025. Citado na página 117.

171 PAN, Z. et al. Privacy-preserving byzantine-robust federated learning via DRL. *IEEE Trans. Veh. Technol.*, v. 74, n. 6, 2025. Citado na página 119.

172 HE, Y. et al. PPBR: Privacy-preserving and byzantine-robust hierarchical federated learning. *IEEE Trans. Mobile Comput.*, v. 25, 2026. Citado na página 119.

173 SAMEERA, K. M.; VINOD, P.; REHIMAN, K. A. R. LFGuard: A defense against label flipping attack in federated learning for vehicular network. *Computer Networks*, v. 252, p. 110768, 2024. Citado 3 vezes nas páginas 119, 120 e 121.

174 NGUYEN, M. et al. HE-enabled federated learning for IDS in resource-constrained IoV. In: *IEEE VTC2024-Fall*. [S.l.: s.n.], 2024. Citado na página 119.

175 DU, J. et al. SoK: On gradient leakage in federated learning. In: *34th USENIX Security Symposium*. [S.l.: s.n.], 2025. ArXiv:2404.05403. Citado na página 119.

176 HOUDA, Z. A. E. et al. Blockchain-enabled federated learning for collaborative intrusion detection in vehicular edge computing. *IEEE Trans. Intell. Transp. Syst.*, v. 25, n. 7, 2024. Citado na página 119.

177 ALTHUNAYYAN, M. et al. A robust multi-stage intrusion detection system for in-vehicle network security using hierarchical federated learning. *Vehicular Communications*, v. 49, p. 100837, 2024. Citado na página 119.

178 FedFT Authors. *FedFT: Fine-Tuning Federated Learning-Based IDS for Transportation IoT*. 2025. ArXiv preprint. ArXiv:2502.06099. Citado na página 120.

179 FANG, M. et al. Local model poisoning attacks to Byzantine-robust federated learning. In: *29th USENIX Security Symposium*. [S.l.: s.n.], 2020. Citado 7 vezes nas páginas 120, 121, 123, 124, 125, 135 e 136.

180 TOLPEGIN, V. et al. Data poisoning attacks against federated learning systems. *arXiv:2007.08432*, 2020. Citado 2 vezes nas páginas 120 e 123.

181 GARG, S. et al. Poisoning attacks on federated learning for autonomous driving. In: *Swedish AI Society Workshop (SAIS)*. [S.l.: s.n.], 2024. ArXiv:2405.01073. Citado na página 120.

182 SHEJWALKAR, V.; HOUMANSADR, A. Manipulating the Byzantine: Optimizing model poisoning attacks and defenses for federated learning. In: *NDSS Symposium*. [S.l.: s.n.], 2021. Citado 3 vezes nas páginas 120, 124 e 135.

183 BLANCHARD, P. et al. Machine learning with adversaries: Byzantine tolerant gradient descent. In: *NeurIPS*. [S.l.: s.n.], 2017. Citado 3 vezes nas páginas 121, 126 e 128.

184 YIN, D. et al. Byzantine-robust distributed learning: Towards optimal statistical rates. In: *ICML*. [S.l.: s.n.], 2018. Citado 3 vezes nas páginas 121, 126 e 128.

185 LI, T. et al. Federated optimization in heterogeneous networks. *Proc. Machine Learning Syst.*, v. 2, p. 429–450, 2020. Citado 4 vezes nas páginas 121, 126, 128 e 134.

186 CHELLAPANDI, V. P. et al. Federated learning for connected and automated vehicles: A survey. *IEEE Trans. Intelligent Vehicles*, v. 9, n. 1, p. 119–135, 2024. Citado 2 vezes nas páginas 121 e 136.

187 BOUALOUACHE, A. et al. Federated learning for V2X misbehavior detection system in 5G edge networks. *IEEE Trans. Intell. Transp. Syst.*, 2024. Citado na página 121.

188 SAMARAKOON, S. et al. *5G-NIDD: A Comprehensive Network Intrusion Detection Dataset Generated over 5G Wireless Network*. IEEE Dataport, 2022. Disponível em: https://dx.doi.org/10.21227/xtep-hv36. Citado na página 136.