Universidade Federal de Juiz de Fora
Instituto de Ciências Exatas
Bacharelado em Sistemas de Informação

# Aprendizado Federado Hierárquico na Camada Fog

## Uma abordagem Edge–Fog–Cloud para eficiência de comunicação

**Rodrigo Campos Figueiredo**

JUIZ DE FORA
JULHO, 2026

# Aprendizado Federado Hierárquico na Camada Fog

## Uma abordagem Edge–Fog–Cloud para eficiência de comunicação

Rodrigo Campos Figueiredo

Universidade Federal de Juiz de Fora

Instituto de Ciências Exatas

Departamento de Ciência da Computação

Bacharelado em Sistemas de Informação

Orientador: Victor Stroële de Andrade Menezes

JUIZ DE FORA

JULHO, 2026

# APRENDIZADO FEDERADO HIERÁRQUICO NA CAMADA FOG

Uma abordagem Edge–Fog–Cloud para eficiência de comunicação

Rodrigo Campos Figueiredo

MONOGRAFIA SUBMETIDA AO CORPO DOCENTE DO INSTITUTO DE CIÊNCIAS EXATAS DA UNIVERSIDADE FEDERAL DE JUIZ DE FORA, COMO PARTE INTEGRANTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE BACHAREL EM SISTEMAS DE INFORMAÇÃO.

Aprovada por:

Victor Stroële de Andrade Menezes
Doutor em Engenharia de Sistemas e Computação (COPPE/UFRJ)

Luciana Brugiolo Gonçalves
Doutora em Ciência da Computação (UFF)

Wagner Antônio Arbex
Doutor em Engenharia de Sistemas e Computação (COPPE/UFRJ)

JUIZ DE FORA

10 DE JULHO, 2026

*Aos meus amigos e irmãos.*

*Aos pais, pelo apoio e sustento.*

# Resumo

O crescimento de dispositivos conectados à Internet das Coisas tem ampliado a coleta e o processamento de dados sensíveis, especialmente em aplicações de saúde digital e monitoramento contínuo. Modelos tradicionais de aprendizado de máquina, baseados em processamento centralizado na nuvem, apresentam limitações relacionadas à privacidade, à latência e ao custo de comunicação. Nesse contexto, o Aprendizado Federado surge como uma abordagem capaz de treinar modelos de forma distribuída, mantendo os dados brutos nos dispositivos de origem. Entretanto, a comunicação direta entre múltiplos clientes e um servidor central ainda pode gerar gargalos e manter elevada dependência da camada Cloud. Este trabalho propõe, implementa e avalia uma arquitetura de Aprendizado Federado Hierárquico baseada no paradigma Edge–Fog–Cloud, na qual a camada Fog realiza agregação parcial de modelos, mantém modelos regionais e reduz a comunicação direta entre os dispositivos Edge e a Cloud. O protótipo foi desenvolvido com Python, PyTorch, FastAPI e Docker, considerando cenários centralizado, federado plano e federado hierárquico. Os experimentos avaliaram acurácia, perda, tempo de execução, rounds até convergência e volume de comunicação entre as camadas. Os resultados indicaram que a arquitetura hierárquica preservou desempenho preditivo comparável ao do Aprendizado Federado plano, com acurácia final de 0,9000 no cenário hierárquico, 0,8800 no plano e 0,8800 no centralizado, valores próximos entre si em um conjunto de teste temporalmente disjunto do treino, ao mesmo tempo em que reduziu em 75% o tráfego direto com a Cloud. Como o rótulo de estresse foi derivado de uma das variáveis de entrada e a avaliação não contou com anotação clínica, a acurácia reflete a recuperação dessa regra pelo modelo e não a detecção clínica de estresse, de modo que a comparação entre cenários sustenta a viabilidade arquitetural, não uma medida de capacidade diagnóstica. Assim, os resultados demonstram que a camada Fog pode atuar como componente ativo no treinamento federado, contribuindo para maior eficiência de comunicação e para a preservação dos dados brutos nos dispositivos de origem.

**Palavras-chave:** Aprendizado Federado, Aprendizado Federado Hierárquico, Fog Computing, Edge–Fog–Cloud, Eficiência de Comunicação.

# Abstract

The growth of Internet of Things devices has increased the collection and processing of sensitive data, especially in digital health and continuous monitoring applications. Traditional machine learning models based on centralized cloud processing present limitations related to privacy, latency, and communication cost. In this context, Federated Learning emerges as an approach capable of training models in a distributed manner while keeping raw data on the source devices. However, direct communication between multiple clients and a central server may still create bottlenecks and maintain a high dependency on the Cloud layer. This work proposes, implements, and evaluates a Hierarchical Federated Learning architecture based on the Edge–Fog–Cloud paradigm, in which the Fog layer performs partial model aggregation, maintains regional models, and reduces direct communication between Edge devices and the Cloud. The prototype was developed using Python, PyTorch, FastAPI, and Docker, considering centralized, flat federated, and hierarchical federated scenarios. The experiments evaluated accuracy, loss, execution time, rounds until convergence, and communication volume between layers. The results indicated that the hierarchical architecture preserved predictive performance comparable to flat Federated Learning, with a final accuracy of 0.9000 in the hierarchical scenario, 0.8800 in the flat scenario, and 0.8800 in the centralized one, values close to one another, on a test set temporally disjoint from training, while reducing direct Cloud traffic by 75%. Since the stress label was derived from one of the input variables and the evaluation did not rely on clinical annotation, accuracy reflects the model's recovery of that rule rather than clinical stress detection, so the comparison across scenarios supports architectural feasibility rather than diagnostic capability. Therefore, the results demonstrate that the Fog layer can act as an active component in federated training, contributing to communication efficiency and to the preservation of raw data on source devices.



**Keywords:** Federated Learning, Hierarchical Federated Learning, Fog Computing, Edge–Fog–Cloud, Communication Efficiency.

# Agradecimentos

Aos meus pais, Marli e Ronaldo, obrigado por todo o encorajamento, sustento e apoio ao longo dessa jornada. Sem vocês, nada disso seria possível, e o sonho da graduação jamais teria se concretizado. Obrigado por sempre estarem aqui quando precisei e por acreditarem em mim quando nem eu mesmo acreditava. Essa conquista é nossa.

Aos meus irmãos, Rafaella e Davi, obrigado por tornarem a caminhada mais leve, por sempre me apoiarem, me atualizarem com as notícias de casa e me fazerem sentir perto, mesmo com quilômetros de distância. Vocês sempre demonstraram afeto nas chamadas de vídeo, nas conversas aconchegantes nos momentos difíceis e, principalmente, na nossa parceiria de irmãos.

À minha família, de modo geral, gostaria de agradecer por todo o carinho, amor e preocupação que sempre tiveram comigo. Vocês são essenciais para esta conquista e sempre me deram forças e exemplos a seguir. Sou grato e orgulhoso por ter vocês comigo.

Aos meus amigos, obrigado pelo companheirismo e por fazerem a trajetória da faculdade ser tão divertida e mais suave, até mesmo os que estão longe. Quero agradecer em especial à Lavínia, Lara, Duda e Sofia, por todo cuidado, incentivo e encorajamento, e por sempre me lembrarem dos meus sonhos e do quanto sou capza. Agradeço também aos amigos que trilharam o caminho da graduação ao meu lado, em especial ao Diogo, Rodrigo e à Taíssa.

Agradeço a todo o corpo docente do Departamento de Ciência da Computação, que teve paciência e muito cuidado ao transmitir todos os ensinamentos para a minha capacitação e desenvolvimento como profissional. Em especial, ao meu orientador, Victor Ströele, que me guiou e me capacitou para a realização deste trabalho.

Por fim, agradeço a todos que, de algum modo, me ajudaram a chegar a este momento tão esperado da minha vida: a realização de um sonho e o encerramento de um capítulo muito feliz, cheio de aprendizados, evolução profissional, novas amizades e muito bem vivido.

*"Lembra que o sono é sagrado e alimenta de horizontes o tempo acordado de viver".*

*Beto Guedes (Amor de Índio)*

# Conteúdo

**Lista de Tabelas** **9**

**Lista de Abreviações** **10**

**1 Introdução** **11**
1.1 Contextualização . . . 11
1.2 Problema de Pesquisa . . . 14
1.3 Questões de Pesquisa . . . 15
1.4 Objetivos . . . 15
1.5 Justificativa . . . 16
1.6 Contribuições do Trabalho . . . 17
1.7 Organização da Monografia . . . 18

**2 Fundamentação Teórica** **19**
2.1 Internet das Coisas e Dados Sensíveis . . . 19
2.2 Computação em Nuvem, Borda e Névoa . . . 20
2.3 Aprendizado Federado e o Algoritmo FedAvg . . . 22
2.4 Aprendizado Federado Hierárquico . . . 22
2.5 Estratégias de Agregação: FedProx e FedAvgM . . . 23
2.6 Modelo BiLSTM com Atenção . . . 23
2.7 Privacidade e Eficiência de Comunicação . . . 24

**3 Trabalhos Relacionados** **26**
3.1 Do FedAvg às Arquiteturas Hierárquicas e Fog . . . 26
3.2 Estratégias de Agregação e Privacidade . . . 27
3.3 Aprendizado Federado em Saúde e IoMT . . . 28
3.4 Comparação dos Trabalhos Relacionados . . . 28
3.5 Lacunas e Posicionamento da Pesquisa . . . 28

**4 Proposta de Arquitetura** **30**
4.1 Visão Geral, Camadas e Comunicação . . . 30
4.2 Protocolo Hierárquico e Estratégias de Agregação . . . 31
4.3 Modelo, Dados e Métricas de Avaliação . . . 32
4.4 Privacidade na Arquitetura . . . 33

**5 Metodologia Experimental** **34**
5.1 Delineamento do Estudo . . . 34
5.2 Ambiente Experimental . . . 34
5.3 Dados Utilizados . . . 35
5.4 Modelo e Cenários Comparativos . . . 36
5.5 Experimentos de Agregação e de Privacidade . . . 38
5.6 Métricas Coletadas . . . 38
5.7 Síntese da Metodologia . . . 39

**6 Implementação do Protótipo** **41**
6.1 Visão Geral e Tecnologias . . . . . 41
6.2 Implementação das Camadas Edge, Fog e Cloud . . . . . 42
6.3 Modelo BiLSTM com Atenção . . . . . 43
6.4 Conjuntos de Dados . . . . . 43
6.5 Implementação dos Cenários e da Privacidade . . . . . 45
6.6 Coleta de Métricas e Reprodutibilidade . . . . . 45
6.7 Síntese da Implementação . . . . . 46

**7 Resultados e Discussão** **47**
7.1 Resultados Consolidados . . . . . 47
7.2 Análise de Comunicação . . . . . 50
7.3 Estratégias de Agregação na Fog . . . . . 51
7.4 Privacidade e Utilidade . . . . . 53
7.5 Resposta às Questões de Pesquisa . . . . . 56
7.6 Ameaças à Validade . . . . . 57
7.7 Síntese dos Resultados . . . . . 60

**8 Considerações Finais** **62**
8.1 Limitações . . . . . 63
8.2 Trabalhos Futuros . . . . . 65
8.3 Encerramento . . . . . 65

**Bibliografia** **67**

# Lista de Tabelas


3.1 Comparação entre trabalhos relacionados e a proposta deste TCC. . . . . 29

4.1 Responsabilidades das camadas na arquitetura proposta. . . . . 31

5.1 Cenários avaliados nos experimentos. . . . . 36
5.2 Parâmetros-base dos experimentos consolidados. . . . . 37

7.1 Resultados finais dos cenários avaliados. . . . . 48
7.2 Comunicação por rodada nos cenários federados. . . . . 50
7.3 Comparação de estratégias de agregação no cenário hierárquico sob partição não-IID. . . . . 51
7.4 Estratégias de agregação sob partição não-IID, com cinco sementes por estratégia. . . . . 53
7.5 Privacidade diferencial aproximada no cenário hierárquico (FedAvg, *clip* 1,0), com cinco sementes por nível de ruído. . . . . 54

# Lista de Abreviações

| | |
|---|---|
| DCC | Departamento de Ciência da Computação |
| UFJF | Universidade Federal de Juiz de Fora |
| IoT | Internet of Things |
| IoMT | Internet of Medical Things |
| IA | Inteligência Artificial |
| ML | Machine Learning |
| FL | Federated Learning |
| FedAvg | Federated Averaging |
| HFL | Hierarchical Federated Learning |
| DP | Differential Privacy |
| Edge | Camada de borda da arquitetura distribuída |
| Fog | Camada intermediária de processamento e agregação |
| Cloud | Camada de consolidação e armazenamento global |

# 1 Introdução

## 1.1 Contextualização

A popularização de dispositivos conectados à Internet das Coisas, do inglês *Internet of Things* (IoT), ampliou a geração contínua de dados em ambientes distribuídos. Sensores, smartphones, dispositivos vestíveis e aplicações móveis passaram a coletar sinais associados ao comportamento, ao contexto e ao estado fisiológico dos usuários. Em aplicações de saúde digital e monitoramento contínuo, tais dados podem apoiar modelos de aprendizado de máquina voltados à identificação de padrões, classificação de risco, detecção de eventos e personalização de serviços.

Apesar do potencial analítico, a centralização desses dados em servidores remotos cria desafios relevantes. Em uma arquitetura tradicional de aprendizado de máquina, os dados coletados nos dispositivos são enviados para uma infraestrutura central, normalmente na nuvem, onde são armazenados e utilizados no treinamento do modelo. Essa abordagem aproveita a capacidade computacional da Cloud, mas aumenta o volume de tráfego, eleva a dependência de conectividade e amplia a exposição de dados sensíveis. Em cenários de saúde, bem-estar e monitoramento fisiológico, essa exposição é especialmente crítica, pois os dados podem revelar informações privadas sobre rotina, condição física e comportamento individual (ALI et al., 2023; RAZA et al., 2022).

Nesse contexto, paradigmas distribuídos como *Edge Computing*, *Fog Computing* e *Cloud Computing* permitem organizar o processamento em camadas. A camada Edge representa os dispositivos próximos à origem dos dados; a camada Fog atua como elemento intermediário, com capacidade de coordenação, agregação e processamento regional; e a Cloud mantém maior capacidade de consolidação, persistência e coordenação global (AL-QAMASH et al., 2018). Essa organização é adequada para sistemas nos quais a transferência contínua de dados brutos para a nuvem é indesejável.

Além da distribuição computacional, o Aprendizado Federado, do inglês *Federated Learning* (FL), propõe uma forma de treinamento colaborativo na qual os dados

permanecem nos clientes e apenas atualizações de modelo são compartilhadas. O algoritmo *Federated Averaging* (FedAvg), proposto por McMahan et al. (MCMAHAN et al., 2017), consolidou essa abordagem ao permitir que múltiplos clientes treinem modelos locais e enviem seus pesos ou atualizações a um servidor agregador. Embora o FL reduza a necessidade de centralizar dados brutos, sua forma plana, baseada em comunicação direta entre clientes Edge e servidor Cloud, ainda pode gerar gargalos de comunicação em cenários com muitos dispositivos.

O Aprendizado Federado Hierárquico surge como alternativa para reduzir essa dependência direta da Cloud. Em vez de todos os clientes se comunicarem com um único servidor central, nós intermediários agregam atualizações de grupos de clientes e enviam apenas modelos regionais para a camada superior. Trabalhos como o modelo Client–Edge–Cloud e propostas Fog–Cloud indicam que a agregação intermediária pode reduzir tráfego na rede de longa distância e favorecer escalabilidade (LIU et al., 2019; NGUYEN et al., 2021).

Este trabalho investiga essa direção por meio de uma arquitetura Edge–Fog–Cloud para treinamento federado aplicado à predição de estresse a partir de métricas de variabilidade da frequência cardíaca, do inglês *Heart Rate Variability* (HRV). O protótipo desenvolvido, denominado Olivia, contém componentes Edge, Fog e Cloud, utiliza um modelo BiLSTM com mecanismo de atenção temporal e implementa cenários experimentais centralizado, federado plano e federado hierárquico. A contribuição central não está na criação de um novo modelo neural, mas na modelagem, implementação e avaliação da Fog como componente ativo do treinamento federado.

Quanto à abordagem metodológica, este é um estudo de natureza aplicada, conduzido por meio do desenvolvimento de um protótipo funcional e de uma avaliação experimental comparativa. A estratégia adotada compara três cenários, centralizado, federado plano e federado hierárquico, mantendo o mesmo modelo, os mesmos dados e os mesmos hiperparâmetros em todas as execuções, de modo que as diferenças observadas de desempenho preditivo e de comunicação possam ser atribuídas à organização arquitetural, e não a fatores externos. As métricas coletadas, descritas em detalhe no Capítulo 5, abrangem acurácia, perda, rodadas até a convergência, volume de comunicação por enlace

e tempo de execução. O detalhamento do delineamento, do ambiente, dos dados e dos procedimentos experimentais é apresentado no Capítulo 5.

O trabalho aqui apresentado se desenvolve no contexto do projeto Olivia, conduzido no Departamento de Ciência da Computação da UFJF sob orientação do Prof. Victor Ströele, que serve como aplicação concreta para a investigação da Fog como núcleo do treinamento federado. Em sua concepção inicial, descrita por Fernandes et al. (FERNANDES et al., 2025), o Olivia foi proposto como um sistema *offline-first* de detecção de estresse em tempo real, integrando dispositivos vestíveis (smartbands) e um aplicativo móvel para monitorar frequência cardíaca e localização do usuário segundo o paradigma Edge–Fog–Cloud. Nessa versão, contudo, a camada Fog cumpria essencialmente um papel de intermediação: armazenava temporariamente os dados coletados, executava a inferência de um modelo já treinado (uma rede LSTM) e encaminhava os dados brutos à Cloud, onde o treinamento efetivamente ocorria de forma centralizada. Ou seja, apesar de a arquitetura já distribuir o processamento em três camadas, o aprendizado de máquina em si permanecia centralizado, com dados fisiológicos brutos sendo transferidos para fora do dispositivo do usuário.

Em uma etapa posterior do projeto (FERNANDES et al., 2025), o Olivia evoluiu para incorporar Aprendizado Federado plano (FedAvg) diretamente na camada Fog: o treinamento de um modelo preditivo BiLSTM com mecanismo de atenção passou a ocorrer localmente no smartphone de cada usuário, com apenas os parâmetros do modelo sendo enviados à Cloud para agregação, validado com cinco usuários reais. Essa mudança eliminou a exposição de dados brutos identificada na versão anterior, mas manteve uma comunicação direta entre cada cliente Fog e o servidor central, sem qualquer agregação intermediária entre dispositivos. É nesse ponto que se insere minha contribuição específica como autor deste TCC dentro do projeto Olivia: enquanto os trabalhos anteriores trataram a Fog primeiro como camada de inferência e, em seguida, como cliente federado individual, este trabalho propõe, implementa e avalia uma reformulação da Fog como agregadora regional ativa, um nó intermediário que agrega os modelos de múltiplos clientes Edge antes de repassar um modelo já consolidado à Cloud. Minha atuação, desenvolvida integralmente no âmbito deste TCC, consistiu em projetar essa arquitetura hierárquica

Edge–Fog–Cloud, implementar os três cenários experimentais comparativos (centralizado, federado plano e federado hierárquico), avaliar estratégias de agregação na Fog (FedAvg, FedProx e FedAvgM) e um mecanismo aproximado de privacidade diferencial, aplicando-os a uma tarefa de classificação binária de estresse a partir de métricas de variabilidade da frequência cardíaca (HRV), tarefa distinta da regressão de frequência cardíaca explorada nas etapas anteriores do projeto.

## 1.2 Problema de Pesquisa

Embora o Aprendizado Federado preserve os dados brutos nos dispositivos de origem, a arquitetura federada plana mantém uma dependência direta entre cada cliente Edge e o servidor central. Em aplicações com múltiplos dispositivos, essa comunicação direta pode elevar o custo de transmissão e concentrar a coordenação em uma única camada Cloud. Além disso, em cenários de dados heterogêneos, a agregação central pode deixar de explorar estruturas regionais ou agrupamentos naturais de clientes.

A camada Fog pode assumir um papel mais ativo nesse processo. Em vez de funcionar apenas como intermediária de rede, ela pode coordenar clientes de uma mesma região lógica, receber atualizações locais, executar agregações parciais e manter modelos regionais antes da consolidação global na Cloud. Essa decisão arquitetural torna necessário definir operacionalmente o que significa usar a Fog como núcleo intermediário do treinamento federado.

A partir dessas considerações, o problema de pesquisa deste trabalho é formulado da seguinte maneira:

> Como modelar, implementar e avaliar uma arquitetura de Aprendizado Federado Hierárquico baseada em Edge–Fog–Cloud, na qual a camada Fog atue como agregadora regional de modelos, priorizando privacidade operacional e eficiência de comunicação em relação ao aprendizado centralizado e ao Aprendizado Federado plano?

Neste trabalho, privacidade operacional é entendida como a não centralização dos dados brutos dos usuários na Cloud durante o treinamento federado. Mecanismos formais

adicionais, como *Differential Privacy* e *Secure Aggregation*, são discutidos e parcialmente simulados, mas não constituem garantia criptográfica completa no protótipo final.

## 1.3 Questões de Pesquisa

A partir do problema definido, o trabalho é orientado pelas seguintes questões de pesquisa:

- **QP1**: Como o desempenho de um modelo treinado com Aprendizado Federado Hierárquico Edge–Fog–Cloud se compara ao treinamento centralizado e ao Aprendizado Federado plano em termos de acurácia, perda, convergência e custo de comunicação?
- **QP2**: Como diferentes estratégias de agregação na camada Fog, especificamente FedAvg, FedProx e FedAvgM, se comportam no cenário hierárquico do protótipo diante de clientes com distribuições de dados heterogêneas (partição não-IID), em termos de acurácia, perda e número de rodadas até a convergência?
- **QP3**: Qual é o impacto da aplicação de um mecanismo aproximado de privacidade diferencial, baseado em clipping e ruído nos updates locais, sobre a utilidade do modelo hierárquico?

As questões foram delimitadas para manter o foco nas dimensões priorizadas pelo escopo do trabalho: privacidade por não centralização dos dados brutos e eficiência de comunicação. Latência e custo computacional são tratados como métricas secundárias, observadas por meio do tempo de execução *wall-clock* dos experimentos.

## 1.4 Objetivos

O objetivo geral deste trabalho é propor, implementar e avaliar uma arquitetura de Aprendizado Federado Hierárquico baseada no paradigma Edge–Fog–Cloud, na qual a camada Fog realiza agregação regional de modelos e reduz a comunicação direta entre clientes Edge e a Cloud, preservando desempenho preditivo competitivo para a tarefa de classificação de estresse a partir de métricas de HRV. Para alcançá-lo, são definidos os seguintes objetivos específicos:

- revisar os fundamentos de Cloud Computing, Edge Computing, Fog Computing, Aprendizado Federado e Aprendizado Federado Hierárquico;
- definir formalmente o papel das camadas Edge, Fog e Cloud na arquitetura proposta;
- implementar um protótipo experimental com componentes de inferência, orquestração e simulação federada;
- modelar o treinamento federado plano e o treinamento federado hierárquico com agregação regional;
- implementar estratégias de agregação FedAvg, FedProx e FedAvgM no cenário hierárquico;
- instrumentar o protótipo para coletar métricas de acurácia, perda, convergência, bytes trafegados e tempo de execução;
- comparar os cenários centralizado, federado plano e federado hierárquico;
- discutir as limitações do mecanismo aproximado de privacidade diferencial usado no protótipo e sua relação com mecanismos formais como DP e agregação segura.

## 1.5 Justificativa

A relevância deste trabalho está associada à necessidade de arquiteturas capazes de explorar dados distribuídos sem exigir sua centralização integral. Em aplicações de saúde digital, dados fisiológicos e comportamentais podem ser sensíveis, e a transferência desses dados para servidores remotos pode aumentar riscos de exposição. O Aprendizado Federado oferece uma alternativa ao permitir que o treinamento ocorra localmente, mas sua arquitetura plana ainda pode demandar comunicação frequente entre todos os clientes e a Cloud.

A Fog Computing é adequada para esse contexto porque posiciona recursos intermediários entre os dispositivos e a nuvem. Ao realizar agregação regional, a Fog pode reduzir o tráfego direto com a Cloud e organizar grupos de clientes em clusters, preservando o princípio de que os dados brutos permanecem próximos da origem. Essa

abordagem também está alinhada à literatura de Aprendizado Federado Hierárquico, que busca reduzir gargalos de comunicação por meio de agregações intermediárias (LIU et al., 2019; NGUYEN et al., 2021).

Do ponto de vista prático, o protótipo Olivia demonstra como uma aplicação de HRV e predição de estresse pode ser estruturada em camadas. A implementação inclui app móvel, serviços Edge e Fog em FastAPI, API Cloud, simulações federadas em PyTorch, modelo BiLSTM com atenção e persistência de métricas em arquivos JSON/CSV. Com isso, o trabalho transforma a proposta arquitetural em um ambiente executável e avaliável.

## 1.6 Contribuições do Trabalho

As principais contribuições deste trabalho são:

- definição de uma arquitetura Edge–Fog–Cloud para Aprendizado Federado Hierárquico aplicada à classificação de estresse com métricas de HRV;
- especificação operacional da Fog como camada de agregação regional, responsável por coordenar clientes Edge e produzir modelos intermediários;
- implementação de um protótipo com componentes Edge, Fog e Cloud, APIs FastAPI, simulações federadas em PyTorch e modelo BiLSTM com atenção;
- comparação experimental entre treinamento centralizado, FL plano e FL hierárquico;
- análise de comunicação separando tráfego Edge–Fog, Fog–Cloud e Edge–Cloud;
- avaliação exploratória de estratégias de agregação FedAvg, FedProx e FedAvgM;
- discussão do impacto de um mecanismo aproximado de privacidade diferencial no treinamento hierárquico;
- disponibilização pública do código-fonte e dos dados de avaliação em repositório aberto[1], permitindo a reprodução dos experimentos.

[1]Disponível em: ⟨https://github.com/rodriggocampos/TCC.git⟩.

## 1.7 Organização da Monografia

Esta monografia está organizada da seguinte forma. O Capítulo 2 apresenta os conceitos fundamentais de IoT, Cloud, Edge, Fog, Aprendizado Federado, Aprendizado Federado Hierárquico, privacidade e eficiência de comunicação. O Capítulo 3 discute os principais trabalhos relacionados e posiciona a proposta em relação à literatura. O Capítulo 4 descreve a arquitetura Edge–Fog–Cloud proposta. O Capítulo 5 apresenta a metodologia experimental. O Capítulo 6 detalha a implementação do protótipo Olivia. O Capítulo 7 apresenta e discute os resultados. Por fim, o Capítulo 8 apresenta as considerações finais e as limitações do trabalho, encerrando com as direções de aprimoramento identificadas.

# 2 Fundamentação Teórica

Este capítulo apresenta os conceitos necessários para compreender a arquitetura proposta. Primeiro são discutidos a Internet das Coisas e a natureza sensível dos dados de saúde digital. Em seguida, são apresentados os paradigmas de computação em nuvem, borda e névoa e a forma como se combinam em uma arquitetura de três camadas. Por fim, são revisados o Aprendizado Federado e o algoritmo FedAvg, sua variante hierárquica, as estratégias de agregação FedProx e FedAvgM, o modelo BiLSTM com atenção e as questões de privacidade e de eficiência de comunicação que orientam a avaliação.

## 2.1 Internet das Coisas e Dados Sensíveis

A Internet das Coisas é composta por dispositivos físicos capazes de coletar, transmitir e processar dados. Em aplicações de saúde digital, esses dispositivos incluem sensores, *smartphones*, *smartwatches* e outros equipamentos vestíveis, que registram sinais fisiológicos, métricas de atividade, localização, padrões de sono e indicadores de bem-estar. Esses dados fisiológicos são úteis para o desenvolvimento de modelos inteligentes, mas exigem tratamento cuidadoso. Métricas de HRV, por exemplo, podem ser usadas para inferir estado de estresse, recuperação física e resposta autonômica (XU et al., 2021). Nos experimentos, o protótipo utiliza um conjunto de HRV no formato real de aquisição do sistema, sem anotação clínica de estresse, e a aplicação-alvo representa um contexto de dados sensíveis, pois simula um sistema de monitoramento fisiológico distribuído.

A centralização desses dados em servidores remotos pode facilitar o treinamento de modelos, porém aumenta o volume de comunicação e a exposição dos usuários. Arquiteturas distribuídas e técnicas de aprendizado que preservam os dados na origem são, por isso, relevantes para aplicações de IoT e saúde (XU et al., 2021; ALI et al., 2023).

## 2.2 Computação em Nuvem, Borda e Névoa

*Cloud Computing* é um modelo de acesso sob demanda a recursos computacionais compartilhados, como servidores, armazenamento, redes e serviços. A definição do NIST destaca características como elasticidade, autoatendimento e amplo acesso via rede (MELL; GRANCE, 2011). Em sistemas de aprendizado de máquina, a *Cloud* oferece capacidade para consolidar modelos, armazenar artefatos, executar avaliações globais e disponibilizar APIs. Neste trabalho, ela é mantida como camada de consolidação global: recebe os modelos regionais agregados pelas *Fogs* e produz o modelo global, expõe uma API FastAPI, mantém o modelo global BiLSTM com atenção e registra o estado do treinamento federado. No cenário federado estrito, a *Cloud* não recebe dados brutos dos usuários nem atualizações individuais de clientes *Edge*.

*Edge Computing* aproxima o processamento da origem dos dados. Em vez de enviar todas as informações para a *Cloud*, dispositivos de borda executam operações locais, como filtragem, extração de características, inferência e treinamento parcial, o que reduz a latência, diminui o tráfego externo e aumenta a autonomia em cenários de conectividade limitada (YOUSEFPOUR et al., 2019). No protótipo Olivia, a camada *Edge* corresponde ao aplicativo móvel e ao serviço de borda: o aplicativo realiza a captura guiada do sinal PPG pela câmera e o serviço processa as métricas derivadas de HRV, que são RMSSD, SDNN, pNN50 e frequência cardíaca média. Na arquitetura federada, cada cliente *Edge* mantém seus dados locais e executa treinamento local, compartilhando apenas pesos do modelo com a camada superior.

Entre a borda e a nuvem situa-se a *Fog Computing*, que posiciona recursos de processamento, armazenamento e comunicação próximos à borda (IORGA et al., 2018). Diferentemente da *Edge*, diretamente associada ao dispositivo ou ao ponto de coleta, a *Fog* pode coordenar múltiplos dispositivos, consolidar informações regionais e atuar como intermediária inteligente (YOUSEFPOUR et al., 2019). Aqui ela é tratada como componente ativo do Aprendizado Federado Hierárquico, e não como simples encaminhadora de mensagens: coordena grupos de clientes *Edge*, distribui modelos regionais, recebe atualizações locais, executa agregação parcial e envia à *Cloud* apenas o modelo regional agregado. Essa definição operacional delimita o que significa tratar a *Fog* como núcleo da

arquitetura.

A combinação dos três paradigmas forma a arquitetura *Edge–Fog–Cloud*, que organiza as responsabilidades em níveis complementares: a *Edge* coleta dados e executa processamento local, a *Fog* agrega e coordena grupos regionais e a *Cloud* consolida o estado global. No fluxo adotado neste trabalho, os dados brutos de HRV permanecem na *Edge*, a *Fog* recebe apenas atualizações de modelo ou métricas agregadas de seus clientes e a *Cloud* recebe apenas modelos regionais já agregados pelas *Fogs*. Essa separação reduz a exposição dos dados brutos e diminui a quantidade de comunicações diretas entre *Edge* e *Cloud*.

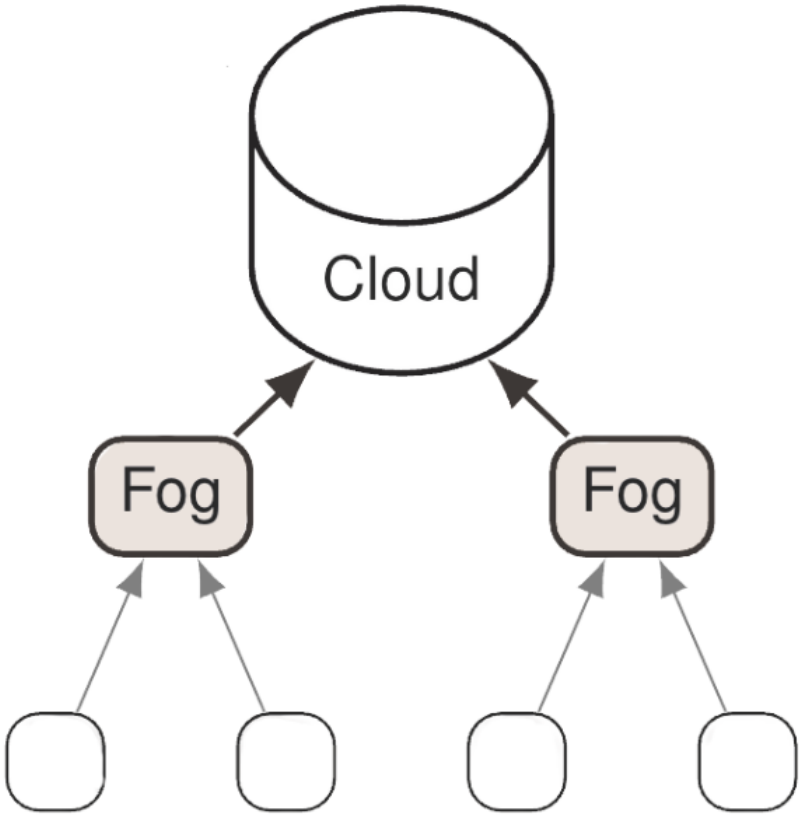


Figura 2.1: Arquitetura conceitual Edge–Fog–Cloud proposta.
Fonte: elaborado pelo autor com apoio de ferramenta de geração de imagens por Inteligência Artificial (2026).

## 2.3 Aprendizado Federado e o Algoritmo FedAvg

Aprendizado Federado é uma abordagem de treinamento distribuído em que múltiplos clientes colaboram para construir um modelo global sem compartilhar seus dados brutos. Cada cliente recebe uma versão do modelo, treina localmente com seus dados e envia uma atualização ao servidor, que as agrega e produz uma nova versão global (MCMAHAN et al., 2017; KAIROUZ et al., 2021). A abordagem é relevante para dados sensíveis porque reduz a necessidade de centralização, mas não garante privacidade absoluta: atualizações de modelo podem vazar informações sob certos ataques, como a reconstrução de gradientes (GEIPING et al., 2020). Mecanismos adicionais, como privacidade diferencial e agregação segura, são frequentemente discutidos como complementos ao FL.

O algoritmo mais difundido nessa área é o FedAvg (MCMAHAN et al., 2017). Em cada rodada, o servidor seleciona clientes, envia o modelo global, recebe os modelos locais treinados e calcula uma média ponderada pelos tamanhos dos conjuntos locais, de modo que clientes com mais amostras contribuam proporcionalmente mais para o modelo agregado. No protótipo Olivia, essa regra é usada tanto no cenário federado plano quanto no hierárquico, variando apenas o nível em que a agregação ocorre: no FL plano todos os clientes enviam atualizações diretamente à *Cloud*, enquanto no FL hierárquico as atualizações são primeiro agregadas pelas *Fogs*, e apenas os modelos regionais resultantes seguem para a *Cloud*.

## 2.4 Aprendizado Federado Hierárquico

O Aprendizado Federado Hierárquico adiciona uma ou mais camadas intermediárias entre clientes e servidor global. Em uma arquitetura *Client–Edge–Cloud*, servidores de borda agregam atualizações de clientes próximos antes da consolidação na *Cloud* (LIU et al., 2019); em arquiteturas *Fog–Cloud*, nós *Fog* assumem papel semelhante, realizando agregação regional (NGUYEN et al., 2021). A principal motivação é reduzir o tráfego de longa distância e melhorar a escalabilidade: como a *Cloud* recebe menos mensagens individuais, o gargalo de comunicação tende a diminuir. Modelos regionais também podem capturar padrões de grupos de clientes, embora isso dependa do particionamento dos

dados e da estratégia de agregação.

No HFL com duas camadas de agregação, o processo ocorre em dois estágios sucessivos. No primeiro, cada *Fog* agrega os modelos locais de seus clientes de forma ponderada pelo número de amostras de cada um. No segundo, a *Cloud* agrega os modelos regionais produzidos pelas *Fogs*, ponderando pela quantidade total de amostras representadas por cada *Fog*. Esse mecanismo reduz o número de mensagens recebidas pela *Cloud*, de todos os clientes individuais para apenas um modelo por *Fog* por rodada (LIU et al., 2019; YUAN et al., 2020).

## 2.5 Estratégias de Agregação: FedProx e FedAvgM

Além do FedAvg, este trabalho avalia FedProx e FedAvgM de forma exploratória. O FedProx adiciona um termo proximal à função de perda local, penalizando desvios excessivos em relação ao modelo global recebido (LI et al., 2020). Controlado por um hiperparâmetro, esse termo atua como uma força de regularização que mantém o modelo local próximo ao ponto de partida global durante o treinamento. A estratégia foi proposta para lidar com heterogeneidade estatística e sistêmica entre clientes, já que em cenários heterogêneos os modelos locais tendem a divergir e prejudicar a convergência.

O FedAvgM incorpora a ideia de momento na atualização global. Em vez de aplicar diretamente a média dos modelos locais ou regionais, acumula-se uma direção de atualização ao longo das rodadas com um fator de momento, que determina quanto da direção anterior é preservado na rodada corrente, de modo análogo ao momento em otimizadores como o SGD. Estratégias adaptativas e com momento são discutidas na literatura como formas de melhorar a estabilidade e acelerar a convergência em FL (REDDI et al., 2021).

## 2.6 Modelo BiLSTM com Atenção

Para a classificação de estresse a partir de métricas de HRV, este trabalho utiliza uma rede neural BiLSTM (*Bidirectional Long Short-Term Memory*) com mecanismo de atenção temporal. Redes LSTM são adequadas a séries temporais porque possuem portas de

memória que permitem capturar dependências de longo prazo, e a versão bidirecional processa a sequência nas duas direções, o que enriquece a representação ao considerar contexto passado e futuro simultaneamente. O mecanismo de atenção temporal atribui pesos diferentes aos passos da sequência, permitindo ao modelo focar nos instantes mais informativos: após a BiLSTM processar a sequência, a atenção calcula uma pontuação de relevância para cada passo e produz um vetor de contexto como média ponderada dos estados ocultos, que é então usado por uma camada classificadora para estimar a probabilidade de estresse.

No protótipo Olivia, o modelo recebe janelas de 30 passos temporais com quatro métricas de HRV por passo, RMSSD, SDNN, pNN50 e frequência cardíaca média, e produz uma probabilidade entre zero e um, classificada como presença de estresse quando supera o limiar de 0,5. A arquitetura foi escolhida por ser adequada a séries temporais curtas e por representar bem as dependências temporais das métricas de HRV (ALI et al., 2023).

## 2.7 Privacidade e Eficiência de Comunicação

O FL reduz a exposição de dados brutos, mas não elimina todos os riscos de privacidade, pois as atualizações de modelo podem carregar informações sobre os dados locais. Duas linhas de defesa frequentemente estudadas são a privacidade diferencial e a agregação segura. A privacidade diferencial fornece uma definição formal para limitar a influência de um indivíduo no resultado de uma análise (DWORK et al., 2006): um mecanismo é diferencialmente privado quando a probabilidade de um resultado não muda de forma significativa caso qualquer indivíduo seja incluído ou excluído da entrada. Em aprendizado profundo, essa ideia é operacionalizada por *clipping* dos gradientes e adição de ruído gaussiano antes da comunicação das atualizações (ABADI et al., 2016), em que o *clipping* limita a contribuição máxima de cada exemplo e o ruído obscurece informações individuais. Ferramentas como Opacus operacionalizam essa abordagem em PyTorch, com contabilidade que rastreia o orçamento de privacidade consumido ao longo do treinamento (YOUSEFPOUR et al., 2021). No protótipo Olivia há suporte ao cliente com Opacus, mas os experimentos consolidados utilizam uma simulação aproximada de privacidade diferencial nos *updates*, baseada em *clipping* L2 e ruído gaussiano, sem contabilidade formal

de epsilon. A agregação segura, por sua vez, impede que o servidor observe atualizações individuais e permite visualizar apenas o agregado (BONAWITZ et al., 2017); esse mecanismo não foi implementado no protótipo final, mas é discutido como evolução natural da arquitetura.

A eficiência de comunicação é outra dimensão central do FL, pois o treinamento pode envolver muitas rodadas e grande volume de parâmetros. No FL plano, cada cliente recebe e envia modelos diretamente ao servidor central a cada rodada. No FL hierárquico, a *Cloud* comunica-se com as *Fogs* e estas com seus clientes, de modo que parte do tráfego é deslocada para enlaces regionais e o tráfego direto com a *Cloud* diminui. Essa vantagem é especialmente relevante quando há muitos clientes e a *Cloud* está em uma rede de longa distância: em vez de receber modelos de todos os clientes a cada rodada, a *Cloud* recebe apenas um modelo por *Fog*, independentemente de quantos clientes essa *Fog* coordena, o que reduz o tráfego na fronteira global de forma proporcional ao número de clientes por *Fog* (LIU et al., 2019; NGUYEN et al., 2021). Neste trabalho, a eficiência de comunicação é medida por bytes trafegados por enlace: o protótipo registra o tráfego *Edge–Cloud*, *Edge–Fog* e *Fog–Cloud*, além de um valor de referência equivalente ao FL plano, que representa o tráfego que ocorreria se todos os clientes comunicassem diretamente com a *Cloud*, permitindo comparar o custo de comunicação entre os cenários.

# 3 Trabalhos Relacionados

Neste capítulo posicionamos nossa proposta em relação à literatura de Aprendizado Federado, Aprendizado Federado Hierárquico, arquiteturas *Fog–Cloud*, privacidade e aplicações em saúde. Não pretendemos uma revisão sistemática exaustiva. O recorte é mais cirúrgico: queremos identificar as contribuições que motivam ou contestam diretamente nossas decisões de projeto e ser honestos sobre onde a proposta se aproxima e onde diverge dos trabalhos existentes. O argumento que percorre o capítulo é que a diferença central entre este trabalho e a maioria da literatura hierárquica está no foco em uma implementação executável, instrumentada por enlace de comunicação e aplicada a um domínio fisiológico concreto.

## 3.1 Do FedAvg às Arquiteturas Hierárquicas e Fog

O trabalho de McMahan et al. (MCMAHAN et al., 2017) é uma referência central para o Aprendizado Federado. Os autores propõem o FedAvg como estratégia de treinamento descentralizado eficiente em comunicação, na qual os clientes realizam múltiplas épocas locais antes de enviar atualizações ao servidor, e essa abordagem é a base do cenário federado plano usado neste TCC. O FedAvg clássico, porém, assume uma organização cliente–servidor em que todos os clientes se comunicam diretamente com um servidor central; a configuração funciona bem em pequena escala, mas tende a gerar gargalo de comunicação quando há muitos dispositivos, e é justamente essa limitação que motiva a introdução de camadas intermediárias de agregação.

Liu et al. (LIU et al., 2019) respondem a essa limitação com uma arquitetura Client–Edge–Cloud para Aprendizado Federado Hierárquico, em que servidores Edge agregam atualizações de clientes próximos e enviam resultados intermediários à Cloud. A principal contribuição está em reconhecer que a comunicação direta cliente–Cloud pode ser substituída por uma estrutura em níveis, o que reduz o tráfego de longa distância. Yuan et al. (YUAN et al., 2020) reforçam essa direção ao discutir a orquestração hierárquica

em redes locais e de longa distância, separando a comunicação local da global. Esses trabalhos inspiram a organização adotada aqui, com uma diferença de ênfase: usamos a terminologia Edge–Fog–Cloud e tratamos a Fog como camada de agregação regional, e não apenas como ponto de passagem.

Na mesma linha, Nguyen et al. (NGUYEN et al., 2021) apresentam o FedFog, que integra Aprendizado Federado a sistemas Fog–Cloud por meio de otimização consciente de rede, aproximando o FL da infraestrutura de Fog Computing e sustentando a hipótese de que a camada intermediária reduz a dependência direta da Cloud. Nossa contribuição difere por seu foco em uma implementação acadêmica executável aplicada a HRV e à classificação de estresse, tornando explícito o papel da Fog no protocolo: coordenar os clientes Edge, executar a agregação parcial e manter modelos regionais antes da consolidação global.

## 3.2 Estratégias de Agregação e Privacidade

A heterogeneidade é uma característica recorrente em FL, pois os clientes podem ter distribuições de dados diferentes, volumes distintos de amostras e capacidades computacionais variadas. O FedProx, proposto por Li et al. (LI et al., 2020), busca mitigar parte desse problema com um termo proximal que restringe o afastamento do modelo local em relação ao modelo global, e estratégias com momento e métodos adaptativos, como os analisados por Reddi et al. (REDDI et al., 2021), também ajudam a melhorar a estabilidade e a convergência. No protótipo Olivia, FedAvg, FedProx e FedAvgM foram implementados no cenário hierárquico para permitir uma comparação exploratória entre estratégias de agregação na Fog.

A privacidade é a outra dimensão complementar ao FL. Embora a abordagem reduza a transferência de dados brutos, a literatura mostra que as atualizações de modelo ainda podem vazar informação, e por isso mecanismos de privacidade diferencial (DWORK et al., 2006; ABADI et al., 2016) e de agregação segura (BONAWITZ et al., 2017) costumam ser combinados com FL para fortalecer as garantias. Neste TCC, a privacidade é tratada em dois níveis: o arquitetural, em que os dados brutos permanecem nos clientes Edge e não são enviados à Cloud no modo federado; e o experimental, em

que uma simulação aproximada de privacidade diferencial é aplicada aos updates locais no cenário hierárquico, por meio de clipping e ruído. Essa segunda camada é exploratória e não substitui uma análise formal de epsilon.

## 3.3 Aprendizado Federado em Saúde e IoMT

Aplicações de FL em saúde vêm sendo estudadas como alternativa para treinar modelos com dados distribuídos de hospitais, dispositivos médicos e sistemas de monitoramento, sem centralizar informações sensíveis (XU et al., 2021; ALI et al., 2023). Raza et al. (RAZA et al., 2022) investigam aprendizado federado em sistemas de monitoramento de ECG, o que evidencia o interesse da área de saúde por modelos distribuídos com preservação de privacidade. O protótipo Olivia se aproxima desse domínio ao usar métricas de HRV para estimar risco de estresse: os experimentos consolidados empregam um conjunto no formato real de aquisição do sistema, sem anotação clínica de estresse, e a arquitetura foi construída para representar uma aplicação de monitoramento fisiológico organizada em camadas.

## 3.4 Comparação dos Trabalhos Relacionados

A Tabela 3.1 sintetiza os trabalhos discutidos nas seções anteriores, destacando o foco de cada um, a arquitetura adotada e sua relação com a proposta deste TCC.

## 3.5 Lacunas e Posicionamento da Pesquisa

A literatura cobre com solidez os fundamentos de FL, HFL, *Fog Computing* e privacidade. A lacuna que identificamos não é teórica, e sim operacional. Poucos trabalhos disponibilizam um protótipo acadêmico executável que conecte as três camadas *Edge–Fog–Cloud* de ponta a ponta, instrumente o tráfego de cada enlace separadamente e aplique tudo isso a um domínio fisiológico. Liu et al. (LIU et al., 2019) e Nguyen et al. (NGUYEN et al., 2021) chegam perto no plano conceitual, mas concentram seu foco na análise de convergência e na otimização de recursos de rede, sem o compromisso de tornar a arquitetura

Tabela 3.1: Comparação entre trabalhos relacionados e a proposta deste TCC.

| **Trabalho** | **Foco** | **Arquitetura** | **Relação com este TCC** |
|---|---|---|---|
| McMahan et al. (MCMAHAN et al., 2017) | FedAvg e FL eficiente em comunicação | Cliente–Servidor | Base do cenário FL plano e da agregação ponderada |
| Liu et al. (LIU et al., 2019) | HFL Client–Edge–Cloud | Clientes, Edge e Cloud | Inspira a agregação intermediária |
| Nguyen et al. (NGUYEN et al., 2021) | FL em Fog–Cloud com consciência de rede | Fog–Cloud | Relacionado ao uso da Fog como agregadora |
| Li et al. (LI et al., 2020) | Heterogeneidade em FL | Cliente–Servidor | Base para FedProx no experimento de agregação |
| Bonawitz et al. (BONAWITZ et al., 2017) | Agregação segura | FL com proteção criptográfica | Trabalho futuro para reforçar privacidade |
| Este TCC | HFL aplicado a HRV e estresse | Edge–Fog–Cloud | Implementa e avalia Fog como agregadora regional |

reproduzível e comparável em um ambiente de contêineres.

# 4 Proposta de Arquitetura

Neste capítulo descrevemos a arquitetura proposta para o Aprendizado Federado Hierárquico em três camadas, *Edge*, *Fog* e *Cloud*. O elemento central da proposta é a definição operacional da *Fog*. Argumentamos que ela não deve ser tratada como um *proxy* de rede, mas como um nó com estado próprio, que coordena clientes regionais, executa agregação parcial e decide o que encaminhar à *Cloud*. A distinção pode parecer sutil, mas é o que separa uma arquitetura hierárquica genuína de um FL plano com um salto a mais na rede.

## 4.1 Visão Geral, Camadas e Comunicação

A proposta consiste em uma arquitetura *Edge–Fog–Cloud* para classificação de estresse a partir de métricas de HRV. A *Edge* coleta ou mantém os dados locais, executa o treinamento local e realiza a inferência próxima ao usuário; a *Fog* coordena grupos de clientes, agrega modelos locais e mantém modelos regionais; e a *Cloud* consolida os modelos regionais e disponibiliza o modelo global. O princípio central é que os dados brutos não precisam ser enviados à *Cloud* no cenário federado: os clientes *Edge* treinam localmente e enviam atualizações de modelo à *Fog*, que executa FedAvg, FedProx ou FedAvgM conforme o experimento e envia apenas o modelo regional agregado à *Cloud*, a qual, por sua vez, agrega os modelos regionais ponderando-os pelo número de amostras representadas por cada *Fog*.

A Tabela 4.1 explicita a separação de responsabilidades, que evita tratar a *Fog* como simples canal de rede: no protocolo proposto, ela possui estado próprio e executa processamento federado. Essa separação se reflete nas fronteiras de comunicação. O tráfego ocorre em duas fronteiras principais: na *Edge–Fog*, a *Fog* distribui modelos regionais aos clientes e recebe atualizações locais; na *Fog–Cloud*, a *Cloud* distribui o modelo global às *Fogs* e recebe os modelos regionais agregados. A fronteira *Edge–Cloud* é evitada no cenário hierárquico, e o protótipo registra, apenas para comparação, um valor de referência que representa o tráfego que ocorreria se todos os clientes se comunicassem

Tabela 4.1: Responsabilidades das camadas na arquitetura proposta.

| **Camada** | **Responsabilidades** | **Estado armazenado** |
|---|---|---|
| *Edge* | Coleta de dados, extração de métricas HRV, inferência local e treinamento local | Dados brutos ou métricas locais, modelo recebido, *update* local |
| *Fog* | Coordenação de clientes, agregação regional, manutenção de modelo regional e encaminhamento à *Cloud* | Modelo regional, métricas agregadas, estado local de orquestração |
| *Cloud* | Consolidação global, avaliação, persistência de artefatos e disponibilização do modelo global | Modelo global, métricas consolidadas, histórico de rodadas |

diretamente com a *Cloud* em um cenário plano equivalente.

## 4.2 Protocolo Hierárquico e Estratégias de Agregação

O protocolo de treinamento hierárquico pode ser descrito em quatro etapas:

1. **Inicialização global**: a *Cloud* inicializa o modelo global e o envia para cada nó *Fog*.

2. **Distribuição regional**: cada *Fog* distribui o modelo regional aos clientes *Edge* de seu grupo.

3. **Treinamento local**: cada cliente *Edge* treina o modelo com seus dados locais por um número fixo de épocas e devolve uma atualização à *Fog*.

4. **Agregação regional e global**: a *Fog* agrega os modelos locais de seus clientes e produz um modelo regional; em seguida, a *Cloud* agrega os modelos regionais de todas as *Fogs* para formar o novo modelo global.

Na agregação regional, a *Fog* calcula a média ponderada dos modelos de seus clientes, com peso proporcional ao número de amostras locais em relação ao total do grupo. Na agregação global, a *Cloud* aplica o mesmo princípio, de modo que cada *Fog* contribui na proporção do total de amostras que representa. Com isso, regiões com mais dados exercem maior influência sobre o modelo global, respeitando a distribuição real dos dados sem exigir sua centralização (LIU et al., 2019; MCMAHAN et al., 2017).

A etapa de agregação foi implementada com três estratégias para o cenário hierárquico. O FedAvg calcula a média ponderada dos modelos locais ou regionais pelo número de amostras. O FedProx mantém essa média, mas adiciona ao treinamento local uma penalização proximal, controlada por hiperparâmetro, que reduz desvios em relação ao modelo recebido. O FedAvgM aplica momento à atualização global, por meio de um fator de *momentum*. No protótipo, essas estratégias são selecionadas pelo parâmetro `--aggregation` do script `simulate_hierarchical.py`, e o objetivo é avaliar se a camada *Fog* acomoda diferentes políticas de agregação sem alterar a estrutura geral da arquitetura.

## 4.3 Modelo, Dados e Métricas de Avaliação

O modelo utilizado é uma rede BiLSTM com mecanismo de atenção temporal. A entrada consiste em janelas de 30 passos temporais com quatro métricas por passo, RMSSD, SDNN, pNN50 e frequência cardíaca média. A BiLSTM possui duas camadas bidirecionais com dimensão oculta de 64 unidades por direção, o que resulta em representações de dimensão 128 para cada passo. O mecanismo de atenção temporal calcula pesos de relevância sobre os 30 passos e gera um vetor de contexto por soma ponderada, usado por uma cabeça classificadora com ativação sigmoide para produzir a probabilidade de estresse. O modelo foi escolhido por ser adequado a séries temporais curtas e por representar dependências temporais nas métricas de HRV; o foco do trabalho, porém, não é comparar arquiteturas neurais, mas avaliar a organização federada hierárquica.

Os experimentos consolidados utilizam um conjunto de medições de HRV no formato real de aquisição do sistema, processado pelo módulo `real_hrv`. A partir de 263 medições ordenadas no tempo, a série é recortada em janelas de 30 passos com quatro *features* por passo, o que produz 234 janelas. A frequência cardíaca média é derivada do número de picos e da duração da captura, e o rótulo binário de estresse é um *proxy* obtido a partir do RMSSD, pois o conjunto não traz anotação de estado. A avaliação usa separação temporal entre treino e teste, com 155 janelas de treino e 50 de teste disjuntas, e as 155 janelas de treino são particionadas entre os clientes de forma aproximadamente uniforme, deixando cerca de 19 janelas por cliente. Durante o desenvolvimento, foi usado também um gerador sintético de HRV, com deslocamentos de perfil fisiológico controlados

por um parâmetro de `profile_shift`, empregado na validação inicial do pipeline; o uso futuro de uma base real anotada continua sendo uma extensão importante para validar a proposta em cenários mais próximos da prática clínica.

A arquitetura é avaliada por métricas de utilidade, comunicação e execução. A utilidade preditiva é medida por acurácia e perda de avaliação; a convergência, pela primeira rodada em que a acurácia atinge 0,90 e a perda de avaliação fica em 0,50 ou menos; a comunicação, pelos bytes trafegados em *Edge–Cloud*, *Edge–Fog* e *Fog–Cloud*; e o tempo, pela duração *wall-clock* de cada rodada. Essas métricas foram escolhidas por estarem diretamente associadas às questões de pesquisa: a comunicação é a mais importante para verificar a contribuição da hierarquia, enquanto a acurácia e a perda indicam se a redução de tráfego compromete o desempenho do modelo.

## 4.4 Privacidade na Arquitetura

A arquitetura adota como princípio que os dados brutos ou amostras individuais de HRV permanecem na *Edge* ou, no máximo, em componentes locais durante a operação, de modo que a *Cloud* recebe modelos regionais agregados e métricas consolidadas, não dados fisiológicos individuais. O protótipo contém duas linhas de suporte à privacidade. A primeira é um cliente compatível com Flower e Opacus, preparado para treinamento local com privacidade diferencial formal (YOUSEFPOUR et al., 2021). A segunda é uma simulação aproximada de privacidade diferencial no cenário hierárquico, que aplica *clipping* L2 sobre o delta dos pesos e adiciona ruído gaussiano antes da agregação (ABADI et al., 2016). A segunda abordagem é a usada nos resultados, por ser simples e executável no ambiente experimental, mas é tratada como aproximação, e não como garantia formal de privacidade diferencial.

# 5 Metodologia Experimental

Neste capítulo descrevemos como desenhamos os experimentos para avaliar a arquitetura proposta. A principal escolha metodológica foi trabalhar com três cenários comparativos, centralizado, FL plano e FL hierárquico, mantendo o mesmo modelo, os mesmos dados e os mesmos hiperparâmetros em todos eles. A decisão foi deliberada, porque só faz sentido atribuir diferenças de desempenho e de comunicação à organização arquitetural quando todo o resto está controlado. O centralizado funciona como referência de desempenho com acesso integral à amostra, o FL plano como *baseline* federado e o hierárquico como a proposta a ser avaliada.

## 5.1 Delineamento do Estudo

O estudo foi delineado para responder às três questões de pesquisa. A QP1 é avaliada pela comparação entre os cenários centralizado, FL plano e FL hierárquico; a QP2, por execuções com FedAvg, FedProx e FedAvgM no cenário hierárquico; e a QP3, por execuções com o mecanismo aproximado de privacidade diferencial baseado em *clipping* e ruído. O foco principal recai sobre a eficiência de comunicação e a preservação arquitetural dos dados brutos, enquanto o desempenho preditivo é tratado como restrição de viabilidade, já que a arquitetura hierárquica só é interessante se reduzir o tráfego com a *Cloud* sem degradar de forma substancial a acurácia.

## 5.2 Ambiente Experimental

O protótipo foi desenvolvido em Python 3.11, PyTorch e FastAPI, com suporte a Docker Compose para a execução dos serviços. A aplicação possui três componentes principais: serviço *Edge*, orquestrador *Fog* e API *Cloud* federada. As simulações foram executadas a partir do contêiner da *Cloud*, com comandos do tipo:

```
docker compose exec cloud-federated-api python -m src.federated.simulate \
```

```
  --scenario flat --rounds 10 --clients 8 --local-epochs 3 --batch-size 8
```

Para o cenário hierárquico, o script utilizado foi:

```
docker compose exec cloud-federated-api python -m src.federated.simulate_hierarchic
  --rounds 10 --clients 8 --fogs 2 --fog-rounds 1 --local-epochs 3 --batch-size 8
```

Embora o projeto contenha um cliente compatível com Flower, os resultados consolidados foram obtidos pelos simuladores próprios `simulate.py` e `simulate_hierarchical.py`, implementados diretamente em PyTorch. Essa escolha reduz dependências externas e torna a instrumentação das métricas de comunicação mais transparente.

## 5.3 Dados Utilizados

Os experimentos consolidados utilizam um conjunto de medições de HRV no formato real de aquisição do sistema, processado pelo módulo `real_hrv`. O conjunto contém 263 medições ordenadas no tempo, das quais derivamos as quatro métricas usadas pelo modelo: RMSSD, SDNN, pNN50 e frequência cardíaca média. A frequência cardíaca média é estimada a partir do número de picos detectados e da duração de cada captura, pois não consta de forma explícita no arquivo.

A série é recortada em janelas deslizantes de 30 passos com passo 1, o que gera 234 janelas. Como o conjunto não possui anotação de estresse, o rótulo binário é um *proxy* derivado do RMSSD, segundo a convenção de que menor variabilidade acompanha maior carga simpática. Como as janelas têm passo 1, janelas vizinhas compartilham 29 dos 30 passos e são quase idênticas, o que torna inadequado embaralhar e dividir aleatoriamente. Por isso, a separação entre treino e avaliação é feita por blocos temporais contíguos: os primeiros 70% da série formam o conjunto de treino e o restante forma o conjunto de teste, descartando na fronteira uma faixa de guarda igual ao comprimento da janela menos um, de modo que nenhuma janela de teste compartilhe passos com janelas de treino. Esse procedimento produz 155 janelas de treino e 50 de teste, temporalmente disjuntas. O limiar do rótulo e a normalização por *feature* são calculados uma única vez sobre o treino e reaplicados globalmente a todos os clientes e ao teste, sem recálculo por cliente,

para não vazar estatística do conjunto de avaliação. Nos cenários federados consolidados (QP1), as janelas de treino são distribuídas entre os clientes de modo aproximadamente uniforme, configurando uma partição aproximadamente IID; a partição não-IID usada na investigação da QP2 é descrita na Seção 5.5. Durante o desenvolvimento, também foi usado um gerador sintético de HRV, descrito no Capítulo 6, empregado na validação inicial do pipeline.

## 5.4 Modelo e Cenários Comparativos

O modelo utilizado em todos os cenários é o BiLSTM com atenção temporal. A entrada possui forma $B \times 30 \times 4$, a BiLSTM é bidirecional com duas camadas e dimensão oculta 64, e a saída é uma probabilidade de estresse no intervalo $[0, 1]$. O treinamento usa função de perda *Binary Cross-Entropy* e otimizador Adam com taxa de aprendizado $10^{-3}$. Manter o mesmo modelo nos três cenários é fundamental para isolar o efeito da arquitetura de treinamento, de modo que as diferenças de desempenho e de comunicação possam ser atribuídas à forma de organização, e não à troca de modelo. Os três cenários avaliados estão resumidos na Tabela 5.1, e os parâmetros comuns a todos eles, na Tabela 5.2.

Tabela 5.1: Cenários avaliados nos experimentos.

| **Cenário** | **Topologia** | **Descrição** |
|---|---|---|
| C1 | Centralizado | Os dados dos clientes são concatenados e treinados em um único modelo central. O tráfego inicial aproxima o envio dos dados brutos à *Cloud*. |
| C2 | FL plano | Oito clientes *Edge* treinam localmente e comunicam-se diretamente com a *Cloud*, que executa FedAvg. |
| C3 | FL hierárquico | Oito clientes são distribuídos entre duas *Fogs*. Cada *Fog* agrega regionalmente e a *Cloud* agrega os modelos regionais. |

A escolha do tamanho de lote merece um comentário. Como cada cliente dispõe de poucas janelas, um lote grande faria com que cada época local executasse um único passo de gradiente, insuficiente para o treinamento federado; um lote de tamanho oito garante vários passos por época em cada cliente e foi o que permitiu a convergência

Tabela 5.2: Parâmetros-base dos experimentos consolidados.

| **Parâmetro** | **Valor** |
|---|---|
| Rodadas globais | 10 |
| Clientes *Edge* | 8 |
| Nós *Fog* | 2 no cenário hierárquico |
| Épocas locais | 3 |
| Janelas de treino por cliente | aproximadamente 19 |
| *Batch size* | 8 |
| Taxa de aprendizado | 0,001 |
| Limiar de convergência | acurácia $\geq$ 0,90 e perda $\leq$ 0,50 |
| Estratégia-base | FedAvg |

observada no Capítulo 7.

No cenário centralizado, os dados de todos os clientes são concatenados em um único conjunto de treinamento. Esse conjunto é exatamente o mesmo que, nos cenários federados, seria particionado entre os clientes, isto é, as 155 janelas reais de treino, sem qualquer adição de dados sintéticos. Garantir que os três cenários compartilhem precisamente a mesma amostra é condição necessária para que a comparação seja atribuível à organização arquitetural. O cenário representa uma referência de desempenho com acesso integral à amostra em um único nó e é, ao mesmo tempo, a abordagem menos favorável à privacidade, por pressupor a centralização dos dados. Seu tráfego é estimado como o custo inicial de envio dos dados brutos para a *Cloud*, sem novo tráfego *Edge–Cloud* após a primeira rodada, o que serve apenas como aproximação para evidenciar o custo da centralização.

No cenário federado plano, cada cliente treina uma cópia local do modelo e envia o modelo atualizado diretamente à *Cloud*, que aplica FedAvg ponderado pelo número de amostras locais. Em cada rodada, o tráfego *Edge–Cloud* inclui o envio do modelo global para todos os clientes e o recebimento dos modelos locais treinados. Esse cenário preserva os dados brutos nos clientes, mas mantém todos eles diretamente conectados à *Cloud* durante o treinamento.

No cenário hierárquico, os clientes são divididos entre duas *Fogs* pela distribuição modular do identificador do cliente. A *Cloud* envia o modelo global para cada *Fog*, que o distribui aos seus clientes, recebe as atualizações locais, aplica a agregação regional e envia apenas o modelo regional agregado à *Cloud*, a qual agrega os modelos regionais

ponderando pelo número de amostras representadas por cada *Fog*. A principal métrica de comunicação desse cenário é `fog_cloud_bytes`; o tráfego *Edge–Fog* continua existindo, porém é considerado regional, e a redução de comunicação com a *Cloud* é calculada comparando o tráfego *Fog–Cloud* com o tráfego *Edge–Cloud* equivalente do cenário plano.

## 5.5 Experimentos de Agregação e de Privacidade

Para investigar a QP2, o cenário hierárquico foi executado com três estratégias, FedAvg, FedProx e FedAvgM, em que o FedProx usa $\mu = 0{,}01$ e o FedAvgM usa *momentum* 0,9. Diferentemente da configuração consolidada da QP1, este experimento usa uma partição não-IID: o conjunto de treino é dividido em blocos temporais contíguos, um por cliente, de modo que cada cliente recebe uma faixa distinta da série e os clientes apresentam distribuições de rótulo diferentes entre si. Essa é a condição adequada para avaliar o mérito do termo proximal do FedProx, concebido para conter a divergência entre clientes heterogêneos. As três estratégias foram executadas uma vez sob essa partição e, em seguida, repetidas com cinco sementes distintas cada, o que permite comparar suas médias de acurácia e perda e avaliar a estabilidade de cada uma. Mesmo com a repetição, a comparação de acurácia entre estratégias permanece sensível ao tamanho reduzido do conjunto de teste, de modo que diferenças de poucos pontos devem ser lidas com cautela e à luz da perda de avaliação, que revela diferenças de calibração não visíveis na acurácia.

Para investigar a QP3, o mesmo cenário hierárquico foi executado com *clipping* L2 do delta de pesos e ruído gaussiano. O parâmetro `dp_clip` foi fixado em 1,0 e o ruído foi variado em sete níveis, de 0,01 a 1,50, ampliando deliberadamente o intervalo até a faixa em que a utilidade colapsa, de modo a mapear o compromisso entre privacidade e utilidade. Esse experimento não calcula epsilon, e por isso os resultados não devem ser interpretados como garantia formal de privacidade diferencial: avaliam apenas o impacto prático de perturbar os *updates* locais antes da agregação.

## 5.6 Métricas Coletadas

As métricas coletadas são:

- `train_loss_mean`: média da perda de treinamento local na rodada;
- `eval_loss`: perda no conjunto global de avaliação;
- `eval_acc`: acurácia no conjunto global de avaliação;
- `edge_cloud_bytes`: bytes trafegados diretamente entre *Edge* e *Cloud*;
- `edge_fog_bytes`: bytes trafegados entre *Edge* e *Fog*;
- `fog_cloud_bytes`: bytes trafegados entre *Fog* e *Cloud*;
- `rounds_until_convergence`: primeira rodada que satisfaz simultaneamente acurácia maior ou igual a 0,90 e perda de avaliação menor ou igual a 0,50;
- `wall_clock_sec`: tempo de execução da rodada.

A convergência é definida por esse critério duplo porque a acurácia isolada pode ser enganosa: uma estratégia pode acertar a maioria das classificações binárias e ainda assim manter perda elevada, sinal de probabilidades mal calibradas. Quando uma estratégia atinge a acurácia-alvo, mas mantém a perda acima do limiar, ela não é considerada convergida segundo este critério, e sua rodada de convergência por acurácia é tratada como não comparável às demais.

O valor de $0,50$ para a perda é adotado como referência ilustrativa, e não como limiar de significado teórico absoluto. Ele situa-se aproximadamente no patamar de uma entropia cruzada binária associada a previsões corretas. Trata-se, portanto, de um corte prático que, neste experimento, separa modelos com probabilidades razoavelmente calibradas dos que acertam o rótulo, mas mantêm previsões mal calibradas.

## 5.7 Síntese da Metodologia

O delineamento adotado tem limitações que reconhecemos. Usamos um conjunto pequeno de medições reais de HRV em formato de aquisição do sistema, com rótulo *proxy*, e parte das configurações foi executada uma única vez, o que torna várias conclusões indicativas e não estatisticamente robustas. Além disso, o conjunto de teste é formado por janelas

deslizantes de passo 1, de modo que janelas vizinhas são fortemente autocorrelacionadas e o número de observações efetivamente independentes é menor que as 50 janelas nominais. Optamos por ser transparentes sobre essas limitações ao longo de toda a análise. O que o experimento pode demonstrar, e demonstra, é que a redução de comunicação com a *Cloud* é estrutural, pois decorre da topologia hierárquica e não de uma configuração específica de dados ou hiperparâmetros. Esse resultado, mesmo obtido em escala controlada, é suficiente para responder às questões de pesquisa dentro do escopo proposto.

# 6 Implementação do Protótipo

Neste capítulo descrevemos como implementamos o protótipo. A Olivia nasceu como uma plataforma funcional de monitoramento de HRV, com aplicativo móvel, captura de PPG e inferência de estresse, e evoluiu para acomodar as simulações federadas necessárias a este trabalho. Por isso, nem toda a estrutura do sistema é estritamente necessária aos experimentos, mas ela reflete a opção por construir algo que funcione como um sistema real, e não apenas como um script de simulação. Descrevemos os dois lados: a plataforma operacional e os componentes experimentais.

## 6.1 Visão Geral e Tecnologias

O projeto está organizado como uma plataforma para captura de sinal PPG, extração de métricas HRV, inferência de estresse e treinamento federado. A estrutura principal contém os módulos `mobile_app`, uma aplicação Flutter para interação com o usuário e captura guiada; `edge_service`, uma API FastAPI responsável por processamento de borda e inferência local; `fog_orchestrator`, uma API FastAPI responsável por orquestração intermediária, fusão de probabilidades e armazenamento local; `cloud_federated`, o módulo de treinamento, simulação federada, modelo global e API *Cloud*; e `metrics`, o diretório de persistência dos resultados experimentais em JSON e CSV. A implementação atende a dois usos: uma demonstração funcional *Edge–Fog–Cloud* de inferência e um ambiente experimental para simular treinamento centralizado, FL plano e FL hierárquico.

As principais tecnologias empregadas são Python 3.11 como linguagem dos serviços e simuladores, PyTorch para o modelo neural, o treinamento local e as agregações, FastAPI na construção das APIs *Edge*, *Fog* e *Cloud*, Docker Compose para a orquestração dos serviços em ambiente local, Flutter no aplicativo móvel de captura, SQLite na persistência local de coletas, usuários, sessões e histórico federado, e ONNX na exportação do modelo global para interoperabilidade. O projeto também possui código compatível com Flower e Opacus no cliente federado, embora os experimentos finais tenham sido

executados pelos simuladores próprios em PyTorch.

## 6.2 Implementação das Camadas Edge, Fog e Cloud

A camada *Edge* é composta pelo aplicativo móvel e pelo serviço `edge_service`. Seu papel funcional é capturar ou receber dados do usuário, calcular métricas de HRV e produzir uma estimativa local de probabilidade de estresse, com o aplicativo orientando a captura do sinal PPG pela câmera. Na avaliação federada, a *Edge* é representada por clientes virtuais: cada cliente possui um conjunto local de dados, fornecido pelo módulo `real_hrv` a partir do conjunto descrito na Seção 6.4, ou pelo gerador sintético `synthetic_hrv` usado durante o desenvolvimento. Em ambos os casos, o cliente treina uma cópia do modelo BiLSTM com atenção e devolve à camada agregadora apenas o `state_dict` do modelo treinado, o que reproduz o princípio do FL de manter os dados locais e compartilhar somente pesos.

A camada *Fog* é implementada pelo serviço `fog_orchestrator` e, nos experimentos, pela lógica de grupos e agregação regional presente em `simulate_hierarchical.py`. Na API funcional, a *Fog* recebe probabilidades locais do *Edge*, consulta a *Cloud* quando necessário e combina respostas, com uma fusão operacional do tipo $0{,}6 \cdot p_{cloud} + 0{,}4 \cdot p_{edge}$ para a inferência consolidada. No treinamento hierárquico, seu papel é mais específico: a função `_client_groups` divide os clientes entre as *Fogs* e a função `_fedavg` calcula a média ponderada dos modelos, de modo que cada *Fog* envia à *Cloud* apenas um modelo regional por rodada, reduzindo o número de mensagens globais.

A camada *Cloud* é implementada no módulo `cloud_federated`, que mantém a API global, o modelo BiLSTM com atenção, as rotinas de inferência e os scripts de simulação. O arquivo `simulate.py` implementa os cenários centralizado e federado plano, e o arquivo `simulate_hierarchical.py` implementa o cenário hierárquico com FedAvg, FedProx, FedAvgM e privacidade diferencial aproximada. A *Cloud* também persiste os modelos treinados em `models/`; no cenário federado plano, o script exporta o modelo global em `global_model.pt` e gera um arquivo ONNX, o que permite reaproveitar o modelo em outros componentes da plataforma.

## 6.3 Modelo BiLSTM com Atenção

O modelo `BiLSTMAttention` recebe uma sequência temporal de HRV com quatro atributos por passo. Sua arquitetura é composta por uma BiLSTM bidirecional de duas camadas, com dimensão oculta 64 por direção, seguida de um mecanismo de atenção temporal com camadas lineares e função `Tanh`, que produz um vetor de contexto por soma ponderada dos estados temporais. Esse vetor alimenta um classificador com camada linear, ReLU, *dropout* e saída sigmoide. A saída final é uma probabilidade de estresse: para a avaliação binária, probabilidades acima de 0,5 são classificadas como classe positiva, enquanto na lógica operacional do aplicativo um limiar mais conservador, como 0,70, pode ser usado para disparar o alerta ao usuário.

## 6.4 Conjuntos de Dados

Durante o desenvolvimento, o arquivo `synthetic_hrv.py` gerou dados sintéticos por cliente para a validação inicial do pipeline. O gerador cria sequências temporais de RMSSD, SDNN, pNN50 e frequência cardíaca média, em que a probabilidade de estresse aumenta quando há menor variabilidade cardíaca e maior frequência cardíaca média, e cada cliente recebe um deslocamento fisiológico controlado por `profile_shift`, o que cria heterogeneidade entre clientes sem exigir uma base real grande. A normalização é aplicada por *feature* ao final da geração. Como os dados sintéticos têm uma separação entre classes relativamente fácil, optou-se por um conjunto mais próximo da prática para os experimentos consolidados.

Esses experimentos consolidados utilizam um conjunto de medições de HRV no formato que o próprio Olivia produz a partir da captura de PPG. O arquivo, com 263 medições ordenadas no tempo, traz por linha as métricas RMSSD, SDNN, pNN50 e SDSD, as bandas espectrais VLF, LF e HF, além de metadados de aquisição como duração da captura, frequência de amostragem e número de picos detectados.

O conjunto foi obtido por meio de capturas de PPG realizadas com o próprio sistema Olivia durante o desenvolvimento, em caráter de teste técnico do pipeline de aquisição, e corresponde a uma única série temporal de medições, não a uma coleta

com múltiplos sujeitos. Por se tratar de aquisição voltada à validação de software, sem finalidade clínica e sem rótulo de estado fisiológico, os dados não constituem uma base anotada de pesquisa com seres humanos. O uso futuro da arquitetura em contexto clínico exigiria coleta sob protocolo aprovado por comitê de ética e com consentimento informado.

A leitura, o pré-processamento e a entrega desse conjunto ao pipeline são feitos pelo módulo `real_hrv`, escrito para expor as mesmas interfaces do gerador sintético, de forma que os cenários experimentais pudessem ser reexecutados sem alterar a lógica de treinamento ou de agregação.

Três decisões de pré-processamento condicionam a leitura dos resultados no Capítulo 7. A primeira é a derivação da frequência cardíaca média, que o modelo usa como quarto atributo e que o conjunto não traz de forma explícita; ela é estimada a partir do número de picos detectados e da duração da captura, pela razão picos/duração $\times$ 60. A segunda é a segmentação: como o modelo opera sobre sequências de 30 passos, a série é recortada em janelas deslizantes, o que produz 234 janelas a partir das 263 medições, e cada janela recebe o rótulo do seu passo mais recente. A terceira, mais relevante para a validade, é a rotulação: o conjunto não possui anotação de estresse, ou seja, não há verdade de campo, e para viabilizar a classificação binária derivamos um rótulo *proxy* a partir do RMSSD, sob a convenção fisiológica de que menor variabilidade tende a acompanhar maior carga simpática. Registramos essa escolha de forma aberta, porque ela permite exercitar o pipeline de aquisição e o fluxo federado de ponta a ponta, mas significa que o experimento testa a mecânica do sistema e o comportamento de comunicação, não a detecção clínica de estresse.

As 234 janelas são separadas por blocos temporais em 155 janelas de treino e 50 de teste, com uma faixa de guarda na fronteira para evitar sobreposição. Para os cenários federados, as 155 janelas de treino são particionadas entre os clientes de modo aproximadamente uniforme, o que deixa cerca de 19 janelas por cliente. A normalização por *feature* é global: a média e o desvio padrão são calculados uma única vez sobre o conjunto de treino e essas mesmas estatísticas são reaplicadas ao conjunto de teste, sem recálculo por cliente, o que evita o vazamento de estatística do conjunto de avaliação e mantém a normalização idêntica entre os cenários centralizado, plano e hierárquico.

Esse volume reduzido por cliente influencia diretamente a escolha de hiperparâmetros de treinamento, como o tamanho do lote e o número de épocas locais, discutida no Capítulo 5.

## 6.5 Implementação dos Cenários e da Privacidade

O cenário centralizado concatena, em um único conjunto de treinamento, exatamente as mesmas 155 janelas reais que, nos cenários federados, seriam particionadas entre os clientes, e treina um único modelo. Não há mistura de dados sintéticos nesse cenário, e garantir que os três cenários compartilhem precisamente a mesma amostra é o que torna a comparação atribuível à organização arquitetural. O cenário federado plano inicializa um modelo global, envia cópias aos clientes, treina localmente e agrega os modelos na *Cloud* por FedAvg. O cenário hierárquico adiciona *Fogs* intermediárias: a *Cloud* envia o modelo para cada *Fog*, as *Fogs* coordenam seus clientes e a *Cloud* recebe apenas modelos regionais. Nesse último, o tráfego é contabilizado separadamente: a comunicação *Edge–Fog* considera o envio e o retorno de modelos entre clientes e *Fogs*, e a comunicação *Fog–Cloud* considera o envio do modelo global para as *Fogs* e o retorno dos modelos regionais, separação que permite demonstrar a redução de comunicação com a *Cloud*.

A privacidade diferencial aproximada do simulador hierárquico é implementada em três passos. Primeiro, calcula-se a norma L2 do delta entre o modelo inicial e o modelo treinado localmente. Segundo, aplica-se *clipping* ao delta quando sua norma ultrapassa o limite configurado. Terceiro, adiciona-se ruído gaussiano proporcional aos parâmetros `dp_noise` e `dp_clip`. Essa implementação permite estudar o efeito de perturbar os *updates* locais sobre a acurácia, mas não substitui uma implementação formal com contabilidade de epsilon; o próprio código documenta essa restrição e indica o uso de Opacus para experimentos formais futuros.

## 6.6 Coleta de Métricas e Reprodutibilidade

Ao final de cada rodada, os scripts registram um dicionário com cenário, rodada, número de clientes, número de *Fogs*, perda de treinamento, perda de avaliação, acurácia, bytes trafegados, tempo de execução e rodada de convergência, salvos em arquivos JSON e CSV no

diretório `metrics`. Os arquivos consolidados usados no Capítulo 7 são `centralized_metrics.json`, `flat_metrics.json` e `hierarchical_metrics.json`.

O código-fonte do protótipo Olivia e o conjunto de dados de avaliação estão disponíveis em repositório público[2]. O repositório inclui os serviços *Edge*, *Fog* e *Cloud*, os simuladores `simulate.py` e `simulate_hierarchical.py`, o módulo `real_hrv` e os arquivos de orquestração Docker Compose, de modo que os cenários relatados no Capítulo 7 podem ser reexecutados a partir dos mesmos comandos descritos na Seção 5.2.

## 6.7 Síntese da Implementação

Ao final do capítulo, o que temos é um sistema executável que vai além de um script de simulação. A opção por construir serviços FastAPI reais, um aplicativo Flutter e a orquestração via Docker Compose custou tempo, mas trouxe a confiança de que a arquitetura funciona em um ambiente próximo do real. O alcance da instrumentação de comunicação precisa ser delimitado com cuidado: o volume de bytes por enlace é contabilizado a partir do tamanho dos modelos efetivamente trafegados em cada rodada, o que torna a redução de comunicação um resultado estrutural da topologia, mas esse volume não equivale a uma medição de custo real de rede sob condições de latência, perda de pacotes ou largura de banda variável, uma vez que os experimentos foram executados em contêineres locais. A decisão de não usar Flower nos experimentos finais, ainda que o código esteja presente, foi pragmática: os simuladores próprios deram controle direto sobre a contagem de bytes por enlace, que é o centro da análise comparativa. Essa escolha afeta a generalidade dos resultados, pois os experimentos medem a arquitetura proposta, e não a interoperabilidade com o ecossistema Flower.

[2]Disponível em: ⟨https://github.com/rodriggocampos/TCC.git⟩.

# 7 Resultados e Discussão

Neste capítulo apresentamos e discutimos os resultados obtidos com o protótipo Olivia. Antes dos números, convém delimitar o que eles sustentam. Os experimentos consolidados foram conduzidos sobre um conjunto de medições de HRV no formato que o próprio sistema produz a partir do sinal PPG, com execução única por configuração. Isso permite observar o comportamento estrutural da arquitetura, isto é, como a hierarquia desloca tráfego da fronteira *Cloud* para a fronteira regional *Edge–Fog* e se esse deslocamento ocorre sem perda relevante de desempenho preditivo. Tratamos os resultados como evidência de viabilidade, e não como prova estatística de superioridade de uma estratégia sobre outra. Os cenários centralizado, plano e hierárquico de referência foram executados uma única vez, ao passo que as duas investigações em que a variabilidade entre execuções é decisiva foram repetidas com cinco sementes: a comparação entre estratégias de agregação sob partição não-IID e a varredura de privacidade diferencial, esta última repetida com sementes justamente para distinguir efeito do ruído de oscilação amostral.

Todos os cenários compartilham o mesmo modelo BiLSTM com atenção, os mesmos dados e os mesmos hiperparâmetros. A configuração consolidada usou oito clientes *Edge*, duas *Fogs*, dez rodadas globais, três épocas locais e lote de tamanho oito. A avaliação é feita sobre um conjunto de teste temporalmente disjunto do treino, obtido pela separação por blocos descrita na Seção 5.3, com 155 janelas de treino e 50 de teste. O cenário hierárquico de referência usa FedAvg sem privacidade diferencial e partição aproximadamente IID; os efeitos da agregação não-IID e da privacidade são tratados em seções próprias.

## 7.1 Resultados Consolidados

A Tabela 7.1 reúne os resultados finais dos três cenários principais após dez rodadas.

O treinamento centralizado atingiu acurácia de 0,8800 no conjunto de teste, valor próximo ao dos cenários federados. Esse resultado merece um esclarecimento meto-

Tabela 7.1: Resultados finais dos cenários avaliados.

| **Cenário** | **Acurácia final** | **Loss final** | **Convergência (rodada)** |
|---|---|---|---|
| Centralizado | 0,8800 | 0,4706 | 5 |
| FL plano | 0,8800 | 0,3452 | 9 |
| FL hierárquico | 0,9000 | 0,3185 | 9 |

dológico. Em uma versão anterior do experimento, o cenário centralizado incluía inadvertidamente dados sintéticos adicionais no treinamento, além das janelas reais, o que inflava artificialmente sua acurácia para 1,0000 e fazia com que o centralizado não compartilhasse exatamente a mesma amostra que os cenários federados. Após corrigir essa inconsistência, o cenário centralizado passou a treinar apenas com as mesmas 155 janelas reais que, nos cenários federados, são distribuídas entre os clientes, de modo que os três cenários passaram a compartilhar exatamente o mesmo dado. Verificamos, na mesma revisão, que os cenários federado plano e hierárquico já consumiam exclusivamente o conjunto real_hrv, sem qualquer mistura de dados sintéticos, onde a contaminação restringia-se à rotina de concatenação do cenário centralizado e não às partições federadas. Os três cenários foram reexecutados a partir do mesmo carregador de dados reais após a correção, de modo que a inconsistência identificada não afeta os resultados federados relatados. Com a correção, o desempenho centralizado deixou de dominar os demais e situou-se na mesma faixa do FL plano e do hierárquico, o que é coerente com o regime de poucos dados: o acesso integral à amostra não confere vantagem expressiva quando a amostra é pequena e a avaliação é temporalmente disjunta do treino.

Convém, contudo, conciliar esse resultado com a afirmação de que a tarefa se reduz a recuperar um limiar sobre uma das entradas. Se a regra fosse recuperável de forma exata, o acesso integral à amostra deveria aproximar o centralizado de 100%, o que não ocorre. Dois fatores explicam o resíduo observado. Primeiro, a separação por blocos temporais faz com que a distribuição do RMSSD no bloco de teste, correspondente ao trecho final da série, difira da do bloco de treino. Como a normalização e o limiar do rótulo são fixados no treino, as janelas de teste próximas da fronteira de decisão passam a ser classificadas de forma instável, o que impede a recuperação perfeita da regra mesmo com acesso completo aos dados. Em segundo lugar, o modelo não aplica o limiar diretamente, mas precisa reconstruí-lo por meio de uma BiLSTM com atenção,

sob dropout e em um número limitado de rodadas, o que introduz erro residual. A circularidade do rótulo (Seção 7.6) torna, portanto, a tarefa simples em princípio, mas não trivial sob um split temporal, e os $0,8800$ do centralizado refletem esse resíduo, não uma capacidade de detecção clínica. Também, como o rótulo é derivado de um limiar sobre o RMSSD, que também é uma das variáveis de entrada, a acurácia mede a recuperação dessa regra, e não a detecção de estresse.

O FL plano alcançou 0,8800 de acurácia no conjunto de teste, com convergência na nona rodada. Esse valor situa-se na mesma faixa do centralizado corrigido, o que indica que, neste regime de dados, distribuir o treinamento entre clientes não acarretou perda relevante de desempenho em relação ao acesso centralizado à amostra. A avaliação, em janelas temporalmente disjuntas do treino, é mais exigente do que a usada em versões anteriores deste trabalho. A comunicação ocorre diretamente entre cada cliente e a *Cloud* em todas as rodadas.

O FL hierárquico atingiu 0,9000 de acurácia no conjunto de teste, também com convergência na nona rodada. A diferença para o FL plano é de dois pontos percentuais, neste caso a favor do hierárquico, o que, dado o tamanho do conjunto de teste (50 janelas, em que cada acerto vale dois pontos percentuais) e a execução única, deve ser lido como equivalência prática entre os dois cenários federados, e não como superioridade do hierárquico. O ponto a reter é que a inserção da camada *Fog* não degradou o desempenho preditivo de forma relevante, e a vantagem da hierarquia aparece, como discutimos a seguir, no custo de comunicação.

O número de rodadas até a convergência adota um critério duplo, registrando a primeira rodada em que o modelo atinge simultaneamente acurácia maior ou igual a 0,90 e perda de avaliação menor ou igual a 0,50 no conjunto de teste. A adoção da perda como segundo critério evita classificar como convergida uma configuração que acerta a maioria das janelas mas mantém probabilidades mal calibradas. O centralizado cruzou esse limiar na rodada 5, enquanto o FL plano e o FL hierárquico o cruzaram na nona rodada, dentro das dez rodadas previstas. Convém observar que esse indicador registra a primeira rodada em que o limiar é atingido, e não a acurácia da última rodada, de modo que os dois valores podem divergir. É o que ocorre no FL plano: ele cruzou o

limiar na nona rodada, mas oscilou para 0,88 na décima, valor reportado como acurácia final na Tabela 7.1. Essa oscilação de uma janela entre rodadas é compatível com o tamanho reduzido do conjunto de teste, em que cada acerto vale dois pontos percentuais, e não indica divergência do treinamento. O FL plano e o hierárquico cruzaram o limiar na mesma rodada, o que é coerente com a leitura de equivalência prática entre eles em desempenho preditivo, deslocando a distinção para o custo de comunicação.

## 7.2 Análise de Comunicação

A vantagem mais clara do cenário hierárquico está na comunicação com a *Cloud*. A Tabela 7.2 apresenta os bytes trafegados por rodada em cada enlace nos dois cenários federados. Como o volume por rodada é constante, o tráfego acumulado cresce de forma proporcional ao número de rodadas, e a relação entre os enlaces permanece a mesma.

Tabela 7.2: Comunicação por rodada nos cenários federados.

| **Cenário** | **Edge–Cloud** | **Edge–Fog** | **Fog–Cloud** |
|---|---|---|---|
| FL plano | 9,715,840 bytes | 0 | 0 |
| FL hierárquico | 0 | 9,715,840 bytes | 2,428,960 bytes |

No FL plano, cada rodada movimenta 9.715.840 bytes na fronteira direta entre os clientes e a *Cloud*. No FL hierárquico, esse tráfego se reorganiza: a fronteira regional *Edge–Fog* concentra os mesmos 9.715.840 bytes, enquanto a fronteira *Fog–Cloud*, que é a que efetivamente cruza a rede de longa distância, movimenta apenas 2.428.960 bytes. A hierarquia não reduz a comunicação total, mas transfere a maior parte do tráfego para o nível regional e diminui o que chega à *Cloud*.

A redução de comunicação com a *Cloud* foi de 75% em relação ao FL plano equivalente. É importante enquadrar corretamente esse valor: ele não é um achado empírico obtido pela execução, mas uma propriedade aritmética da topologia escolhida. Com oito clientes e duas *Fogs*, cada *Fog* agrega quatro clientes e envia um único modelo regional à *Cloud*, de modo que a fronteira global recebe um modelo por *Fog* em vez de um modelo por cliente. O tráfego *Fog–Cloud* corresponde, portanto, a um quarto do tráfego *Edge–Cloud* do FL plano, o que se verifica diretamente nos valores instrumentados: $2.428.960 \times 4 = 9.715.840$ bytes. Em termos gerais, com $k$ clientes por *Fog* e

modelos de tamanho fixo, o tráfego que cruza a fronteira global cai por um fator de $k$, e a redução percentual é $1 - 1/k$, independentemente de sementes, estratégia de agregação ou conteúdo dos dados. A instrumentação confirma esse comportamento previsto, mas o mérito do resultado está em explicitá-lo e medi-lo, não em descobri-lo. O cenário centralizado não participa dessa comparação porque não realiza trocas de modelo por rodada; ele concentra os dados brutos uma única vez, o que contraria justamente o princípio de não centralização que motiva o trabalho.

## 7.3 Estratégias de Agregação na Fog

A QP2 foi investigada executando o cenário hierárquico com FedAvg, FedProx e FedAvgM, sem privacidade diferencial, sobre uma partição não-IID. Nessa partição, descrita na Seção 5.3, cada cliente recebe um bloco temporal contíguo do conjunto de treino, de modo que os clientes apresentam distribuições de rótulo distintas, condição adequada para avaliar o mérito das estratégias projetadas para heterogeneidade. A Tabela 7.3 resume os resultados.

Tabela 7.3: Comparação de estratégias de agregação no cenário hierárquico sob partição não-IID.

| **Estratégia** | **Acurácia** | **Loss** | **Convergência** | ***Train loss*** |
|---|---|---|---|---|
| FedAvg | 0,8600 | 0,4208 | $> 10$ | 0,1089 |
| FedProx | 0,8800 | 0,3764 | $> 10$ | 0,0661 |
| FedAvgM | 0,9200 | 1,1245 | $> 10$ | 0,2395 |

As três estratégias treinaram sob heterogeneidade, o que mostra que a *Fog* acomoda diferentes políticas de agregação sem alterar a estrutura da arquitetura. Quanto à comparação entre FedAvg e FedProx, os valores de execução única ficaram próximos: 0,8600 para o FedAvg e 0,8800 para o FedProx, uma diferença de dois pontos percentuais que, em um conjunto de teste de 50 janelas, corresponde a um único acerto. Uma diferença dessa magnitude está dentro da granularidade do conjunto de teste e não permite, por si só, afirmar superioridade de uma estratégia sobre a outra. O termo proximal do FedProx, em princípio, tende a ajudar quando os clientes têm distribuições distintas, por conter o afastamento dos modelos locais em relação ao modelo recebido; verificar se esse efeito se traduz em ganho consistente exigiria repetir todas as estratégias sob as mesmas

sementes, como discutido adiante.

O FedAvgM apresentou a maior acurácia da tabela (0,9200), mas acompanhada de uma perda de avaliação muito elevada (1,1245), de uma a três ordens de grandeza acima das demais. Embora cruzasse o limiar de acurácia já na quinta rodada, ele jamais satisfez o critério de perda ($\leq$ 0,50) e, por isso, não é considerado convergido segundo o critério duplo adotado, aparecendo como $>$ 10 na Tabela 7.3. Essa combinação de acurácia alta com perda alta indica previsões corretas quanto ao limiar de decisão, porém mal calibradas, ou seja, confiantes mesmo quando próximas do erro, o que é consistente com a instabilidade introduzida pelo fator de *momentum* de 0,9. Seu número de acerto, portanto, não deve ser lido isoladamente como superioridade, e a alta perda torna o FedAvgM não comparável às demais estratégias em termos de convergência.

Para examinar a estabilidade desses resultados, repetimos as execuções de FedProx, FedAvg e FedAvgM sob a partição não-IID, cada uma com cinco sementes distintas. O FedProx produziu os valores 0,86, 0,88, 0,88, 0,88 e 0,92, com média de 0,884, desvio padrão de 0,022 e perda de avaliação média de 0,396. O FedAvg produziu 0,88, 0,88, 0,88, 0,90 e 0,92, com média de 0,892, desvio padrão de 0,018 e perda de avaliação média de 0,371. O FedAvgM produziu 0,84, 0,92, 0,92, 0,94 e 0,96, com média de 0,916, mas com desvio padrão de 0,046, cerca do dobro do das outras duas estratégias, e perda de avaliação média de 0,941, com quatro das cinco sementes acima de 0,50 (de 0,43 a 1,19). FedAvg e FedProx concentram-se em torno de 0,88 a 0,90, o que sugere que sua dispersão reflete sobretudo a granularidade do conjunto de teste, e não variabilidade substantiva: em 50 janelas, cada acerto desloca a acurácia em dois pontos percentuais. O dado mais informativo é que as médias de FedAvg (0,892) e FedProx (0,884) praticamente coincidem, com o FedAvg inclusive um pouco à frente, o oposto do que sugeriam os valores de execução única da Tabela 7.3 (0,8600 para o FedAvg e 0,8800 para o FedProx). Isso evidencia que aqueles valores isolados não eram representativos, em especial o do FedAvg, que correspondeu a um sorteio baixo, e confirma que a diferença entre as duas estratégias está dentro da granularidade da avaliação, sem que se possa afirmar superioridade de qualquer uma delas. O FedAvgM, por sua vez, confirma em cinco sementes o que a execução única já indicava: ainda que sua acurácia média seja a mais alta, ela vem acompanhada de perda

de avaliação alta e de maior variabilidade, o que caracteriza um modelo mal calibrado de forma consistente, e não por acaso de uma execução. A combinação de alta acurácia com alta perda sob o *momentum* de 0,9 é, portanto, um comportamento sistemático do FedAvgM neste cenário.

A Tabela 7.4 resume essas execuções repetidas.

Tabela 7.4: Estratégias de agregação sob partição não-IID, com cinco sementes por estratégia.

| **Estratégia** | **Acurácia (média ± desvio)** | ***Loss* (média)** | **Sementes com *loss* > 0,50** |
|---|---|---|---|
| FedAvg | $0{,}892 \pm 0{,}018$ | 0,371 | 0 de 5 |
| FedProx | $0{,}884 \pm 0{,}022$ | 0,396 | 0 de 5 |
| FedAvgM | $0{,}916 \pm 0{,}046$ | 0,941 | 4 de 5 |

A interpretação adequada permanece cautelosa, porque o conjunto de teste tem 50 janelas, em que cada acerto corresponde a dois pontos percentuais, de modo que as diferenças de acurácia observadas entre estratégias são da ordem de poucos acertos. O que o experimento sustenta é que as três estratégias operam sob heterogeneidade na camada *Fog* com desempenho de acurácia na mesma faixa, que FedAvg e FedProx apresentaram médias praticamente coincidentes (0,892 e 0,884) sem vantagem atribuível a qualquer um deles, e que o FedAvgM, embora atinja a maior acurácia média, distingue-se das outras duas por uma perda de avaliação sistematicamente alta e por maior variabilidade, sinais de má calibração associada ao *momentum* de 0,9. A consolidação de qualquer conclusão adicional, especialmente sobre o desempenho relativo fino entre as estratégias, depende de uma busca de hiperparâmetros, detalhada no Capítulo 8.

## 7.4 Privacidade e Utilidade

A QP3 foi avaliada com a privacidade diferencial aproximada do simulador hierárquico, que aplica *clipping* L2 ao delta de pesos e adiciona ruído gaussiano. A Tabela 7.5 apresenta sete níveis de ruído sobre o cenário hierárquico com FedAvg, ampliando deliberadamente o intervalo até a faixa em que a utilidade colapsa, de modo a mapear o compromisso entre privacidade e utilidade. Cada nível de ruído foi repetido com cinco sementes distintas,

e a tabela reporta a média e o desvio padrão da acurácia, a perda de avaliação média, o número de sementes com perda acima de 0,50 e o número de sementes que atingiram o critério duplo de convergência (acurácia $\geq 0{,}90$ e perda $\leq 0{,}50$ simultaneamente em alguma rodada). A repetição com sementes é essencial aqui: em execução única, a curva exibia subidas de acurácia em níveis de ruído mais altos, um comportamento contraintuitivo que só a média entre sementes permite atribuir corretamente à granularidade do conjunto de teste, e não a um efeito do ruído.

Tabela 7.5: Privacidade diferencial aproximada no cenário hierárquico (FedAvg, *clip* 1,0), com cinco sementes por nível de ruído.

| **Ruído** | **Acurácia (média ± desvio)** | ***Loss* (média)** | **Sementes com *loss* > 0,50** | **Sementes convergidas** |
|---|---|---|---|---|
| 0,01 | $0{,}856 \pm 0{,}009$ | 0,440 | 0 de 5 | 0 de 5 |
| 0,05 | $0{,}860 \pm 0{,}032$ | 0,411 | 0 de 5 | 0 de 5 |
| 0,10 | $0{,}872 \pm 0{,}042$ | 0,372 | 1 de 5 | 1 de 5 |
| 0,20 | $0{,}848 \pm 0{,}130$ | 0,376 | 1 de 5 | 4 de 5 |
| 0,40 | $0{,}536 \pm 0{,}134$ | 1,359 | 4 de 5 | 0 de 5 |
| 0,80 | $0{,}452 \pm 0{,}063$ | 13,234 | 5 de 5 | 0 de 5 |
| 1,50 | $0{,}460 \pm 0{,}065$ | 31,978 | 5 de 5 | 0 de 5 |

Os resultados com sementes delineiam a curva de compromisso entre privacidade e utilidade e dissolvem a aparente anomalia da execução única. Em um primeiro regime, com ruído entre 0,01 e 0,20, as médias de acurácia ficam achatadas entre 0,848 e 0,872, sem tendência de subida ou queda, o que indica que perturbações pequenas não comprometem a utilidade. Aqui está o ponto que motivou a repetição: em execução única, a acurácia parecia aumentar com o ruído, chegando a 0,98 em ruído 0,20, um comportamento que não faz sentido físico, já que a privacidade diferencial degrada a utilidade e não a melhora. A média entre cinco sementes mostra que aquele valor era um sorteio alto isolado, pois o mesmo nível de ruído 0,20 contém uma semente em 0,64, e o desvio padrão salta para 0,130, o maior de todo o regime baixo. A subida aparente era, portanto, granularidade do conjunto de teste de 50 janelas, em que cada acerto vale dois pontos percentuais, e não efeito do ruído. A partir de 0,40 a degradação torna-se inequívoca, com a acurácia média caindo para 0,536 e a perda média subindo para 1,359; note-se que a execução única registrara 0,78 nesse mesmo nível, outro sorteio não representativo que a média corrige. Em 0,80 e 1,50 a utilidade colapsa, com acurácia média de 0,452 e 0,460, próxima de um

classificador aleatório, acompanhada de perdas de avaliação muito altas (médias de 13,234 e 31,978). Existe, portanto, uma faixa em que adicionar ruído preserva o desempenho e um ponto, em torno de 0,40, a partir do qual o modelo deixa de aprender. A coluna de sementes com perda acima de 0,50 oferece um sinal de degradação mais limpo que a acurácia isolada: nenhuma das cinco sementes ultrapassa esse limiar em ruído 0,01 e 0,05, apenas uma o faz em 0,10 e 0,20, quatro em 0,40 e todas as cinco em 0,80 e 1,50. A coluna de sementes convergidas confirma o mesmo quadro pelo critério duplo de acurácia e perda: nenhuma converge no ruído mais baixo, e o regime 0,40 a 1,50 não produz nenhuma convergência. O ruído 0,20 merece uma ressalva: quatro das cinco sementes atingem o critério duplo em alguma rodada (entre a sétima e a nona), mas a convergência é registrada na primeira rodada em que o critério é satisfeito e não é necessariamente sustentada até o fim, de modo que uma das sementes que convergiu recua para 0,64 na décima rodada. É essa instabilidade que produz a média de 0,848 com desvio de 0,130 e caracteriza 0,20 como o ponto de transição entre o regime estável e o de colapso, e não como um patamar de utilidade plena.

Duas qualificações acompanham essa leitura. A primeira é que, como o rótulo é derivado diretamente de uma das variáveis de entrada, o modelo tende a ser intrinsecamente robusto a perturbações nos pesos enquanto o ruído é pequeno, já que a regra a ser recuperada é simples, o que ajuda a explicar a estabilidade no regime de ruído baixo. A segunda é que a curva mede utilidade, não privacidade: o eixo de ruído não foi convertido em um orçamento de privacidade. A necessidade de repetição com múltiplas sementes, levantada na versão anterior deste experimento, foi atendida, e é o que sustenta a leitura acima: a curva suavizada pelas médias substitui a oscilação espúria da execução única por uma transição monotônica entre o regime estável e o de colapso.

O ponto que o experimento não responde é a garantia formal de privacidade. Como o mecanismo não contabiliza epsilon, os resultados indicam viabilidade prática de adicionar ruído sem destruir o desempenho, mas não constituem prova de privacidade diferencial. Uma avaliação formal, com Opacus e contabilidade de orçamento, permanece como trabalho futuro.

A privacidade neste trabalho deve ser lida em duas camadas. A primeira é ar-

quitetural: no FL plano e no FL hierárquico, os dados brutos não são centralizados na *Cloud*, o que já representa um avanço em relação ao cenário centralizado. A segunda é algorítmica: a privacidade diferencial aproximada reduz a exposição direta do delta local, mas não oferece garantia formal sem contabilidade de privacidade.

A hierarquia acrescenta uma vantagem operacional. A *Cloud* deixa de observar atualizações individuais de clientes e passa a receber apenas modelos regionais agregados. Resta, porém, um ponto sensível: a *Fog* poderia observar *updates* individuais em uma implementação real. Por isso, a combinação com agregação segura entre *Edge* e *Fog*, ou com privacidade diferencial formal nos clientes, seria necessária para proteger contra um nó *Fog* curioso ou comprometido.

## 7.5 Resposta às Questões de Pesquisa

**QP1**: O FL hierárquico apresentou desempenho preditivo comparável tanto ao do FL plano quanto ao do centralizado corrigido. A acurácia final no conjunto de teste foi de 0,9000 no hierárquico, 0,8800 no plano e 0,8800 no centralizado, valores próximos entre si. Após a correção que garante que os três cenários treinem exatamente sobre a mesma amostra real, o centralizado deixou de exibir a vantagem artificial anterior, e a diferença de dois pontos percentuais entre os dois cenários federados, neste caso a favor do hierárquico, está dentro da granularidade de um conjunto de teste com 50 janelas e de uma execução única, devendo ser lida como equivalência prática, não como superioridade. Essa comparação vale em termos relativos, como evidência de viabilidade arquitetural, e não como medida de detecção de estresse, pelas razões expostas na Seção 7.6. A distinção decisiva está na comunicação, em que o hierárquico reduziu em 75% o tráfego com a *Cloud* em relação ao FL plano equivalente, redução que, como discutido, é uma propriedade aritmética da topologia. A leitura conjunta é que a hierarquia mantém desempenho equivalente ao do plano e troca algumas rodadas de convergência por uma redução expressiva e previsível de tráfego na fronteira global.

**QP2**: A *Fog* executou as três estratégias de agregação com sucesso, o que mostra que a camada acomoda diferentes políticas sem alteração estrutural. A comparação foi feita sob partição não-IID, condição apropriada para a pergunta, e as três estratégias

foram repetidas com cinco sementes. As médias de acurácia ficaram em 0,892 para o FedAvg, 0,884 para o FedProx e 0,916 para o FedAvgM. FedAvg e FedProx praticamente coincidem, com o FedAvg até um pouco à frente, de modo que não há superioridade atribuível ao termo proximal do FedProx neste cenário, e a diferença entre eles está dentro da granularidade do conjunto de teste. O FedAvgM atinge a maior acurácia média, mas distingue-se por uma perda de avaliação alta (média 0,941, com quatro das cinco sementes acima de 0,50) e por uma variabilidade maior (desvio 0,046, cerca do dobro das outras duas), o que caracteriza, de forma consistente entre sementes, um modelo mal calibrado associado ao *momentum* de 0,9; por isso sua acurácia não deve ser lida isoladamente como superioridade e ele não satisfaz o critério duplo de convergência. A evidência, portanto, é que a camada *Fog* suporta as três estratégias com acurácia comparável, mas que a escolha entre elas deve levar em conta a calibração, e não apenas a taxa de acerto.

**QP3**: A varredura de ruído, de 0,01 a 1,50, repetida com cinco sementes por nível, mapeou a curva de compromisso entre privacidade e utilidade. No regime baixo (0,01 a 0,20), a acurácia média manteve-se achatada entre 0,848 e 0,872, sem a subida que a execução única sugeria e que se revelou um artefato do conjunto de teste pequeno; a partir de 0,40 a degradação ficou inequívoca (acurácia média de 0,536), e em 0,80 e 1,50 a utilidade colapsou para desempenho próximo do aleatório (médias de 0,452 e 0,460), com perdas de avaliação muito altas. Existe, portanto, uma faixa em que adicionar ruído preserva o desempenho e um ponto, em torno de 0,40, a partir do qual o modelo deixa de aprender. A repetição com sementes foi o que permitiu afirmar isso com segurança, ao distinguir efeito do ruído de oscilação amostral. Dois fatores ainda qualificam a leitura: a robustez no regime baixo é favorecida pela circularidade do rótulo, que torna a regra simples de recuperar; e, como o mecanismo não contabiliza epsilon, a curva mede utilidade, não privacidade formal.

## 7.6 Ameaças à Validade

Os resultados têm limitações que delimitam seu alcance. A mais importante é determinante para a leitura correta dos números de acurácia.

A circularidade do rótulo é a principal limitação. O conjunto utilizado está no

formato real de aquisição do sistema, mas não possui anotação de estresse, o que exigiu um rótulo *proxy* obtido pela aplicação de um limiar ao RMSSD. Ocorre que o RMSSD também é uma das quatro variáveis de entrada do modelo. Em consequência, a tarefa de classificação reduz-se, em essência, a aplicar uma regra fixa ao próprio sinal de entrada do modelo, e não a inferir um estado fisiológico latente a partir de pistas indiretas. Isso explica diretamente por que as acurácias se mantêm bem acima do acaso e por que o modelo é robusto a pequenas perturbações. Não implica, porém, a recuperação perfeita da regra, já que a não-estacionariedade do RMSSD entre os blocos de treino e de teste impede que o limiar seja reconstruído com exatidão, como discutido na Seção 7.1. Os valores de acurácia relatados não medem a capacidade de detectar estresse, mas apenas a de reconstruir um limiar a partir de uma de suas entradas. Qualquer leitura clínica desses números seria indevida.

Uma limitação que foi tratada, e cuja correção condiciona a leitura dos números atuais, diz respeito à separação entre treino e avaliação. As janelas são deslizantes com passo 1, de modo que janelas vizinhas compartilham 29 dos 30 passos e são quase idênticas. Por isso, embaralhar as janelas e só então dividir, ou avaliar sobre o conjunto completo, faria janelas quase iguais aparecerem em treino e em teste, configurando vazamento e inflando a acurácia. Para evitar isso, os resultados consolidados adotam uma separação por blocos temporais contíguos, com 155 janelas de treino e 50 de teste, descartando na fronteira uma faixa de guarda de 29 janelas, de modo que nenhuma janela de teste compartilha passos com janelas de treino. O limiar do rótulo e a normalização são calculados apenas no treino. As acurácias relatadas, mais baixas do que as de versões anteriores deste trabalho, são consequência direta dessa correção e representam uma estimativa mais honesta de generalização.

Uma consequência desse protocolo é o tamanho reduzido do conjunto de teste. Com 50 janelas, cada acerto ou erro corresponde a dois pontos percentuais de acurácia, de modo que diferenças de poucos pontos entre cenários ou estratégias estão dentro da margem de uma a três janelas e não sustentam conclusões finas. Há ainda um segundo efeito: como as janelas de teste são deslizantes com passo 1, janelas vizinhas compartilham 29 dos 30 passos e são fortemente autocorrelacionadas entre si. Em consequência, o

número de observações efetivamente independentes no conjunto de teste é menor do que as 50 janelas nominais, o que reduz o poder estatístico real da avaliação para além do que o tamanho aparente sugere. Somadas ao fato de que os cenários consolidados (centralizado, plano e hierárquico de referência) foram executados uma única vez, ainda que as comparações entre estratégias de agregação e a varredura de privacidade já tenham sido repetidas com cinco sementes, essas limitações reforçam que as comparações de acurácia devem ser lidas como indicativas, não como evidência estatística.

É importante delimitar o que essas limitações afetam e o que não afetam. Após a correção que removeu a mistura de dados sintéticos no cenário centralizado, os três cenários passaram a compartilhar exatamente o mesmo dado de treino, o mesmo modelo, o mesmo rótulo e o mesmo protocolo de avaliação, de modo que a circularidade incide de forma igual sobre os três. A comparação relativa entre as topologias, portanto, permanece válida como evidência de viabilidade: a leitura de que o hierárquico acompanha o plano de perto continua sustentada. O que não se sustenta é ler qualquer um desses números absolutos como medida de detecção de estresse.

As demais limitações complementam o quadro. A comparação entre topologias (QP1) foi conduzida em partição aproximadamente IID; sua repetição em partição não-IID é uma extensão natural, ainda que a investigação das estratégias de agregação (QP2) já tenha sido feita sob heterogeneidade. O conjunto é pequeno, e a partição entre oito clientes deixa cerca de 19 janelas de treino por cliente, o que torna o treinamento sensível ao tamanho do lote e ao número de passos locais. Cabe ressaltar que os oito clientes não correspondem a oito indivíduos distintos: eles são obtidos pelo particionamento de uma única série de aquisição de HRV, de modo que a heterogeneidade entre clientes é induzida pela partição, e não por diferenças fisiológicas reais entre pessoas. Uma avaliação federada com dados de múltiplos sujeitos é necessária para caracterizar a heterogeneidade que ocorreria em um cenário de implantação. O número de clientes e *Fogs* foi limitado a oito e duas. Quanto ao modelo, a escolha de uma BiLSTM bidirecional de duas camadas com atenção temporal foi mantida após a observação, durante o desenvolvimento, de que arquiteturas mais simples apresentaram desempenho inferior nesta tarefa; reconhecemos, porém, que para o regime de dados e a regra de rotulação adotados um modelo dessa

capacidade é mais expressivo do que o estritamente necessário, o que constitui uma escolha conservadora e não uma exigência do problema. A privacidade diferencial é aproximada e não contabiliza epsilon, de modo que, embora a varredura ampliada de ruído já mostre a região de degradação da utilidade, não há garantia formal de privacidade nem conversão do ruído em orçamento de privacidade. Por fim, o tempo é medido como *wall-clock* em ambiente local de contêiner e não representa latência ou custo real de rede.

Essas limitações não comprometem a contribuição arquitetural, que é a especificação e a medição da Fog como agregadora regional, mas precisam acompanhar a leitura dos números. O protótipo demonstra de forma robusta o comportamento de comunicação da hierarquia. A validação do desempenho preditivo como detecção de estresse depende de dados reais anotados, em maior volume.

## 7.7 Síntese dos Resultados

O resultado que consideramos mais sólido é a redução de 75% no tráfego com a *Cloud*, porque decorre da topologia e não de uma configuração particular. Esse valor não depende de sementes nem da estratégia de agregação escolhida: sempre que a *Fog* agrega um grupo de clientes e envia um único modelo regional à *Cloud*, o tráfego na fronteira global cai na proporção inversa ao tamanho do grupo. Os experimentos confirmaram esse comportamento de forma direta.

Os demais resultados são consistentes, ainda que peçam cautela por dois motivos: a execução única dos cenários consolidados e o tamanho reduzido do conjunto de teste, discutidos na Seção 7.6, além da circularidade do rótulo, que impede a leitura clínica dos números. Os três cenários convergiram ou se aproximaram do limiar, o hierárquico acompanhou o plano em acurácia, as três estratégias de agregação funcionaram sob heterogeneidade e a varredura de privacidade com sementes mostrou que a privacidade diferencial aproximada preserva o desempenho no regime de ruído baixo e o degrada de forma inequívoca a partir de perturbações em torno de 0,40. A leitura geral é que a arquitetura reduz a comunicação como esperado e que, em termos relativos, o cenário hierárquico preserva utilidade comparável à do plano. O passo seguinte, detalhado no Capítulo 8, é consolidar esse comportamento com repetição em múltiplas sementes também nos cenários

consolidados, dados reais anotados, mais clientes e privacidade diferencial formal com contabilidade de epsilon.

# 8 Considerações Finais

Este trabalho propôs, implementou e avaliou uma arquitetura de Aprendizado Federado Hierárquico baseada em *Edge–Fog–Cloud* para classificação de estresse a partir de métricas de HRV. Ao final, separamos as conclusões que consideramos sólidas das que tratamos como provisórias, e começamos pelas primeiras.

A conclusão principal é que a camada *Fog* pode atuar como agregadora regional sem exigir mudanças no modelo de aprendizado ou nos dados dos clientes. No protótipo Olivia, isso se traduziu em uma redução de 75% no tráfego com a *Cloud* em relação ao FL plano equivalente. Consideramos esse resultado estrutural, porque decorre da topologia hierárquica e não de uma configuração específica de experimento. A *Edge* mantém os dados locais, a *Fog* agrega um modelo por região e a *Cloud* recebe apenas esses modelos regionais, de modo que a economia de tráfego cresce com o número de clientes por *Fog*, independentemente da escala.

A segunda conclusão é que, em termos relativos, essa redução de tráfego não comprometeu o desempenho preditivo. Em um conjunto de teste temporalmente disjunto do treino, o FL hierárquico atingiu 0,9000 de acurácia, contra 0,8800 do FL plano e 0,8800 do centralizado, valores próximos entre si. Após a correção que assegura que os três cenários treinem exatamente sobre a mesma amostra real, o centralizado deixou de exibir a vantagem artificial de uma versão anterior e passou a situar-se na mesma faixa dos cenários federados. A diferença entre os cenários federados ficou em cerca de dois pontos percentuais, neste caso a favor do hierárquico, valor que, dado o tamanho do conjunto de teste e a execução única, indica equivalência prática entre os dois. O hierárquico precisou de mais rodadas para convergir, o que é coerente com a etapa extra de agregação, mas alcançou utilidade comparável à do plano ao final do treinamento. Cabe enfatizar que essa conclusão é uma comparação relativa entre topologias, e não uma medida de capacidade de detecção de estresse, pois, como detalhado na Seção 7.6, o rótulo deriva de uma variável de entrada do modelo, o que torna a tarefa próxima da reconstrução de uma regra fixa.

As conclusões provisórias dizem respeito à comparação entre estratégias de agregação

e ao efeito da privacidade diferencial aproximada. Sob partição não-IID, as três estratégias foram repetidas com cinco sementes e apresentaram acurácia média na mesma faixa: 0,892 para o FedAvg, 0,884 para o FedProx e 0,916 para o FedAvgM. As médias de FedAvg e FedProx praticamente coincidem, com o FedAvg até um pouco à frente, de modo que não afirmamos superioridade do termo proximal do FedProx neste cenário. O FedAvgM, apesar da maior acurácia média, distinguiu-se por perda de avaliação alta (média 0,941, com quatro das cinco sementes acima de 0,50) e por maior variabilidade, o que caracteriza, de forma consistente entre sementes, um modelo mal calibrado associado ao *momentum* 0,9, e que não satisfaz o critério duplo de convergência. A privacidade diferencial aproximada, também repetida com cinco sementes por nível de ruído, preservou a acurácia média no regime de ruído baixo (até cerca de 0,20) e a degradou de forma inequívoca a partir de perturbações em torno de 0,40, com colapso para desempenho aleatório nos níveis mais elevados, o que delineia a curva de compromisso entre privacidade e utilidade. A repetição com sementes foi decisiva nesta varredura: ela dissolveu a subida espúria de acurácia que a execução única exibia em níveis de ruído mais altos, mostrando que se tratava de granularidade do conjunto de teste, e não de efeito do ruído. Ainda assim, como o conjunto de teste é pequeno e o mecanismo não calcula epsilon, tratamos a curva como evidência de viabilidade prática, e não como prova formal de privacidade. Sem cálculo de epsilon, não é possível afirmar o nível formal de proteção oferecido pelo mecanismo.

## 8.1 Limitações

As principais limitações deste trabalho são:

- a avaliação usa separação temporal entre treino e teste (155 janelas de treino e 50 de teste, com faixa de guarda), o que elimina o vazamento entre janelas vizinhas, mas o conjunto de teste resultante é pequeno (50 janelas), de modo que cada acerto vale dois pontos percentuais e diferenças finas de acurácia não são estatisticamente conclusivas;

- o rótulo de estresse é derivado do RMSSD, que também é uma das variáveis de entrada do modelo, de modo que a tarefa se reduz a recuperar um limiar sobre o

próprio sinal de entrada, e a acurácia não mede detecção clínica de estresse;

- o conjunto de dados, embora no formato real de aquisição do sistema, não possui anotação clínica de estresse, o que exigiu o rótulo *proxy* descrito acima e restringe os experimentos à verificação de mecânica e comunicação;
- a partição entre clientes na comparação consolidada de topologias (QP1) é aproximadamente IID; a investigação da QP2 já usa partição não-IID, mas a consolidação dessa comparação ainda depende de repetição com múltiplas sementes e busca de hiperparâmetros;
- o conjunto é pequeno, e a partição das 155 janelas de treino entre oito clientes deixa cerca de 19 janelas por cliente, o que tornou o treinamento federado sensível ao tamanho do lote e ao número de passos locais;
- o número de clientes e *Fogs* foi limitado a oito clientes e duas *Fogs*;
- os oito clientes não correspondem a oito indivíduos distintos, mas ao particionamento de uma única série de aquisição, de modo que a heterogeneidade entre clientes é induzida pela partição e não por diferenças reais entre sujeitos;
- as três estratégias de agregação (FedAvg, FedProx e FedAvgM) e a varredura de privacidade diferencial foram repetidas com cinco sementes, mas os cenários consolidados de topologia (centralizado, plano e hierárquico de referência) permaneceram em execução única, o que limita conclusões estatísticas sobre eles;
- a privacidade diferencial implementada no simulador é aproximada e não fornece garantia formal baseada em epsilon, embora a varredura de ruído tenha sido deliberadamente ampliada até a faixa de colapso da utilidade;
- a agregação segura não foi implementada;
- o tempo de execução medido é *wall-clock* local e não representa latência real de rede;
- a comparação entre FedAvg, FedProx e FedAvgM foi exploratória e depende de ajuste de hiperparâmetros.

Essas limitações são compatíveis com o escopo de um protótipo acadêmico, mas indicam o que precisa ser tratado antes de generalizar os resultados para aplicações reais de saúde digital.

## 8.2 Trabalhos Futuros

A evidência de viabilidade obtida aqui abre algumas direções para trabalhos futuros, que organizamos por ordem de prioridade.

- **Rótulo clínico**: substituir o *proxy* derivado do RMSSD por anotação clínica de estresse, obtida sob protocolo experimental aprovado, de modo que a acurácia passe a medir detecção de fato, e não a reconstrução de uma regra fixa sobre uma das variáveis de entrada do modelo. Esta é a mudança mais importante para validar a aplicação.
- **Consolidação estatística**: estender a repetição com múltiplas sementes também aos cenários consolidados de topologia (centralizado, plano e hierárquico de referência), que ainda se apoiam em execução única, e acompanhá-la de busca de hiperparâmetros, para substituir as comparações exploratórias por conclusões com suporte estatístico em todas as questões de pesquisa.
- **Privacidade formal**: adotar privacidade diferencial formal, com Opacus e contabilidade de epsilon, e combiná-la com agregação segura entre *Edge* e *Fog*, fortalecendo as garantias contra um nó *Fog* curioso ou comprometido.
- **Avaliação em escala e com múltiplos sujeitos**: avaliar a arquitetura com mais clientes e *Fogs*, com dados de indivíduos distintos e com partições não-IID de heterogeneidade crescente, para observar como a economia de comunicação e o comportamento das estratégias de agregação evoluem à medida que o sistema cresce.

## 8.3 Encerramento

O que aprendemos com este trabalho cabe em uma ideia: a *Fog* não precisa ser redefinida para funcionar como núcleo do Aprendizado Federado Hierárquico, ela precisa ser tratada

como tal. A diferença entre um nó de rede que apenas repassa mensagens e um nó federado que agrega, mantém estado e decide o que encaminhar é, em grande parte, uma decisão de software. O Olivia mostra que essa decisão é implementável, instrumentável e avaliável com as ferramentas disponíveis.

Os experimentos indicaram que a arquitetura reduz a comunicação com a *Cloud* como esperado e que, em termos relativos, o treinamento hierárquico preserva desempenho comparável ao do federado plano no cenário avaliado. Trata-se de uma evidência de viabilidade, e não de uma validação ampla, cujos próximos passos foram organizados na seção anterior. Ainda assim, o protótipo construído é mais do que uma proposta conceitual. É um sistema que executa, mede e compara, e que torna visíveis tanto as condições em que funciona quanto os pontos que ainda precisam de validação. Para favorecer a verificação independente e a continuidade do trabalho, o código-fonte e os dados de avaliação foram disponibilizados em repositório público[3].

[3]Disponível em: ⟨https://github.com/rodriggocampos/TCC.git⟩.

# Bibliografia

ABADI, M. et al. Deep learning with differential privacy. In: *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*. [S.l.]: ACM, 2016. p. 308–318.

AL-QAMASH, A. et al. Cloud, fog, and edge computing: A software engineering perspective. In: *2018 International Conference on Computer and Applications*. [S.l.]: IEEE, 2018. p. 276–284.

ALI, M. et al. Federated learning for privacy preservation in smart healthcare systems: A comprehensive survey. *IEEE Journal of Biomedical and Health Informatics*, v. 27, n. 2, p. 778–789, 2023.

BONAWITZ, K. et al. Practical secure aggregation for privacy-preserving machine learning. In: *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*. [S.l.]: ACM, 2017. p. 1175–1191.

DWORK, C. et al. Calibrating noise to sensitivity in private data analysis. In: *Theory of Cryptography Conference*. [S.l.]: Springer, 2006. p. 265–284.

FERNANDES, G. et al. Detecçao de estresse em tempo real usando dispositivos iot e aprendizado profundo. In: SBC. *Simpósio Brasileiro de Computação Ubíqua e Pervasiva (SBCUP)*. [S.l.], 2025. p. 161–170.

GEIPING, J. et al. Inverting gradients: How easy is it to break privacy in federated learning? *arXiv preprint arXiv:2003.14053*, 2020.

IORGA, M. et al. *Fog Computing Conceptual Model*. [S.l.], 2018.

KAIROUZ, P. et al. Advances and open problems in federated learning. *Foundations and Trends in Machine Learning*, v. 14, n. 1–2, p. 1–210, 2021.

LI, T. et al. Federated optimization in heterogeneous networks. *Proceedings of Machine Learning and Systems*, v. 2, p. 429–450, 2020.

LIU, L. et al. Client-edge-cloud hierarchical federated learning. *arXiv preprint arXiv:1905.06641*, 2019.

MCMAHAN, H. B. et al. Communication-efficient learning of deep networks from decentralized data. In: *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. [S.l.]: PMLR, 2017. (Proceedings of Machine Learning Research, v. 54), p. 1273–1282.

MELL, P.; GRANCE, T. *The NIST Definition of Cloud Computing*. [S.l.], 2011.

NGUYEN, V.-D. et al. FedFog: Network-aware optimization of federated learning over wireless fog-cloud systems. *arXiv preprint arXiv:2107.02755*, 2021.

RAZA, A. et al. Designing ECG monitoring healthcare system with federated transfer learning and explainable AI. In: *Knowledge-Based Systems*. [S.l.: s.n.], 2022. v. 236, p. 107763.

REDDI, S. J. et al. Adaptive federated optimization. In: *International Conference on Learning Representations*. [S.l.: s.n.], 2021.

XU, J. et al. Federated learning for healthcare informatics. *Journal of Healthcare Informatics Research*, v. 5, p. 1–19, 2021.

YOUSEFPOUR, A. et al. All one needs to know about fog computing and related edge computing paradigms: A complete survey. *Journal of Systems Architecture*, Elsevier, v. 98, p. 289–330, 2019.

YOUSEFPOUR, A. et al. Opacus: User-friendly differential privacy library in PyTorch. *arXiv preprint arXiv:2109.12298*, 2021.

YUAN, J. et al. Hierarchical federated learning through LAN-WAN orchestration. *arXiv preprint arXiv:2010.11612*, 2020.