Relatório Científico

Dados do Projeto

- **Nome:** Maria Victoria França Silva Ramos
- **Responsável:** Miguel Elias Mitre Campista
- **Título do projeto:** Seleção de Clientes em Aprendizado Federado
- **Número do projeto FAPESP:** 2024/08223-3 Vinculado ao Auxílio à Pesquisa 2023/00811-0
- **Período de vigência:** 01/06/2024 a 31/05/2026 (??)
- **Período coberto pelo relatório científico:** 01/06/2025 a 31/12/2025

Resumo do Projeto

Na sociedade atual, marcada pela ubiquidade da tecnologia da informação, bilhões de dispositivos espalhados pelo mundo geram dados de forma contínua. Cada ação realizada por pessoas em interação com ferramentas inteligentes é capturada e armazenada. O crescimento da conectividade e a ampla disseminação da Internet têm intensificado esse processo. Dispositivos ligados à Internet das Coisas (IoT) são protagonistas nesse cenário, com sensores distribuídos em indústrias, áreas urbanas e ambientes domésticos que coletam e processam informações instantaneamente, com foco na melhoria de setores como saúde, mobilidade e segurança pública. Diante disso, garantir a privacidade dos usuários e a capacidade de lidar com essa avalanche de dados tornou-se uma exigência fundamental. As possibilidades de prever comportamentos, oferecer serviços personalizados e automatizar tarefas transformaram os dados em um recurso estratégico para o mercado. Assim, a segurança da informação passou a ser uma necessidade indispensável nos tempos atuais.

O aprendizado federado surgiu como uma alternativa promissora para lidar com grandes volumes de dados, preservando, ao mesmo tempo, a privacidade dos usuários. Sua proposta se fundamenta na colaboração distribuída, em que cada participante contribui com um modelo pré-treinado localmente, evitando o compartilhamento direto de informações pessoais. Essa arquitetura representa um avanço significativo, pois promove uma aliança entre eficiência computacional e respeito à confidencialidade dos dados. Entretanto, o aprendizado federado ainda enfrenta desafios consideráveis, sendo a heterogeneidade entre os usuários participantes do treinamento um dos mais marcantes. Essa variedade - seja em poder computacional, qualidade da conexão ou nos próprios dados locais - impacta diretamente no ritmo de evolução do modelo global, muitas vezes, prolongando seu tempo de convergência. Inicialmente estruturado somente de forma síncrona, o aprendizado federado precisou se reestruturar, com o objetivo de contornar as limitações trazidas pela disparidade entre os clientes. Ao romper com os requisitos de sincronização, essa técnica passou a evoluir de forma mais flexível e resiliente. O progresso do modelo global deixou de depender da contribuição simultânea de todos os participantes, especialmente daqueles com redes instáveis ou dispositivos menos potentes. Assim, o sistema passou a avançar conforme os modelos locais estivessem disponíveis, respeitando o ritmo e as condições de cada colaborador. Essa mudança não apenas torna o processo mais eficiente, como também mais inclusivo e adaptável. Nesse contexto, observa-se que, como consequência natural da dinâmica assíncrona, ocorre a seleção de uma parcela dos clientes que estão mais ativos ou disponíveis. Essa predominância, embora não seja fruto de uma escolha deliberada, contribui para a redução do tempo necessário para a convergência do modelo, promovendo maior fluidez no processo de treinamento, sem comprometer seu caráter colaborativo. Assim, atualmente, consolidaram-se três principais modalidades de aprendizado federado: o síncrono [1], o semi-síncrono [2] e o assíncrono

[3]. Diante disso, o presente trabalho se propõe a investigar, por meio da simulação dos diferentes regimes temporais, a eficácia do paradigma assíncrono, especialmente em relação à diminuição do tempo de convergência e à melhoria da acurácia do modelo global. Trata-se de um esforço para compreender, com profundidade e cuidado, como o tempo e a diversidade dos participantes influenciam o desempenho e a viabilidade de soluções mais eficientes e sensíveis à realidade dos usuários.

Realizações no Período

Ao longo desse período, as atividades concentraram-se na melhoria da ferramenta de simulação, de modo a viabilizar a representação de cenários adversos mais realísticos e a avaliação de novas métricas relevantes. Assim, foram adicionadas as funcionalidades de simulação de atrasos dos clientes, distribuição de dados mais próximas de situações reais e quantificação de perdas durante o treinamento. Paralelamente, foi desenvolvida uma nova fórmula de agregação eficiente voltada ao aprendizado de máquina federado assíncrono, com foco em maior robustez e acurácia. Como resultado, obtiveram-se evidências experimentais que comprovam a superioridade da arquitetura assíncrona em relação à síncrona, tanto em termos de acurácia quanto na redução de perda de trabalho, refletindo diretamente em menor desperdício energético. Adicionalmente, foi implementada uma técnica de seleção de clientes, que visa diminuir a instabilidade durante os estágios iniciais de treinamento.

Simulação do Aprendizado Federado Síncrono

O aprendizado federado síncrono foi desenvolvido usando a estratégia *FedAvg* [1] como referência. Essa técnica, que efetua o treinamento distribuído de modelos de aprendizado de máquina, é dividida nas seguintes etapas:

1. **Inicialização do Modelo Global:** O servidor central define um modelo inicial e o distribui para os clientes disponíveis, que participarão do treinamento.

2. **Treinamento Local:** Cada cliente treina o modelo recebido do servidor com seus próprios dados, efetuando um número de iterações locais (*epochs*).

3. **Envio dos Pesos Atualizados:** Os clientes enviam apenas os pesos atualizados do modelo pré-treinado localmente para o servidor central, sem compartilhar informações privadas.

4. **Agregação e Atualização do Modelo Global:** O servidor central recebe as atualizações enviadas pelos clientes - dispositivos inteligentes - e calcula os novos pesos a partir de uma

média ponderada em função do número de amostras de cada cliente. Esse cálculo gera um novo modelo global.

5. **Alcance da Convergência:** Esse processo se repete por várias rodadas até atingir a convergência do modelo global.

A princípio, pretendia-se utilizar o *framework* Flower [4] para a implementação dos tipos de aprendizado federado. Entretanto, a adaptação da ferramenta - que, originalmente, suportava apenas simulações do modelo síncrono - não foi possível, na medida em que seu funcionamento está intrinsecamente conectado com a divisão do processo em rodadas. Com o objetivo de manter o paralelismo com a versão assíncrona, tornou-se necessário explorar outros recursos para experimentação. Dessa forma, foi desenvolvida uma nova implementação utilizando as bibliotecas do Python: TensorFlow e NumPy. O repositório com o código-fonte responsável pelas simulações está disponível no seguinte link: https://github.com/mariavictoriar/FederatedLearningSimulation.

Simulação do Aprendizado Federado Assíncrono

A elaboração do aprendizado federado assíncrono baseou-se na estratégia *Asynchronous Federated Optimization* [3]. Nela, a separação em "rodadas" apresentada, anteriormente, pela estratégia *FedAvg* [1] é abandonada e o processo passa a ser dividido nas seguintes etapas:

1. **Inicialização do Modelo Global:** O servidor central define um modelo inicial e seleciona um parcela dos clientes disponíveis para participarem do treinamento.

2. **Treinamento Local:** Cada cliente ativado pelo servidor treina o modelo inicial com seus próprios dados, efetuando um número de iterações locais (*epochs*).

3. **Envio dos Pesos Atualizados:** Os clientes enviam apenas os pesos atualizados do modelo pré-treinado localmente para o servidor central, sem compartilhar informações privadas.

4. **Ordenação dos Pesos Atualizados:** O servidor ordena as atualizações recebidas de acordo com o atraso de cada cliente.

5. **Agregação Sequencial e Atualização do Modelo Global:** O servidor central é alimentado sequencialmente com as atualizações enviadas pelos clientes. O novo modelo global é gerado a partir da fórmula adaptativa de agregação entre os pesos do modelo global anterior e os novos pesos fornecidos pelo cliente atual, contrastando com modo síncrono que efetua esse cálculo por meio de uma média ponderada dos novos pesos fornecidos por todos os clientes.

6. **Alcance da Convergência:** Esse processo é mantido em execução até o alcance da convergência do modelo global.

Simulador do Aprendizado Federado

O simulador de aprendizado federado é estruturado com base na integração de três componentes principais: o código principal (main), o servidor e os clientes. A simulação tem início com o código principal, que define os hiperparâmetros do experimento, realiza a distribuição dos dados entre os clientes, instancia o servidor e dá início ao processo de treinamento. O servidor atua como o elemento central de coordenação do sistema, sendo responsável pela criação do modelo global inicial e da condução das rodadas de aprendizado federado. A cada rodada, o servidor distribui os parâmetros do modelo global para os clientes, autoriza o início do treinamento local, coleta os pesos atualizados, realiza a agregação dos pesos recebidos e, por último, atualiza o modelo global. Após a conclusão de todas as rodadas, o modelo final é submetido a uma etapa de avaliação. Os clientes, por sua vez, realizam o treinamento local com base em seus próprios conjuntos de dados e nos parâmetros recebidos do servidor. Essa arquitetura modular permite a simulação, de maneira controlada, dos principais mecanismos envolvidos em sistemas reais de aprendizado federado.

Distribuição de Dados

A forma como os dados são distribuídos entre os clientes exerce influência direta no comportamento e no desempenho do aprendizado de máquina federado. Neste trabalho, foram consideradas duas estratégias distintas de distribuição de dados, com o objetivo de analisar cenários que variam desde condições ideais até situações adversas mais próximas da realidade.

A primeira estratégia corresponde à distribuição IID (independente e identicamente distribuída), na qual o conjunto de dados de treinamento é previamente embaralhado e particionado de maneira uniforme entre os clientes. Nessa configuração, cada cliente recebe a mesma quantidade de amostras e mantém uma distribuição de classes semelhante à do conjunto global. Trata-se de um cenário controlado e favorável, frequentemente utilizado como linha de base, pois minimiza discrepâncias estatísticas entre os participantes do treinamento.

A segunda estratégia adota uma distribuição Não-IID (não independente e identicamente distribuída), caracterizada por uma alocação não uniforme dos dados entre os clientes. Nesse caso, cada cliente tem acesso apenas a um subconjunto limitado de classes, o que resulta em distribuições locais significativamente distintas da distribuição global. Essa configuração, considerada uma distribuição patológica, introduz forte heterogeneidade estatística no sistema, dificultando a convergência do modelo global e acentuando os desafios inerentes ao aprendizado federado. Tal cenário é parti-

cularmente relevante por refletir de forma mais fiel ambientes reais, nos quais os dados disponíveis em cada dispositivo são naturalmente enviesados e não representativos do todo.

A adoção dessas duas abordagens permite avaliar o impacto da heterogeneidade dos dados sobre o processo de treinamento distribuído, bem como analisar a robustez das técnicas propostas frente a diferentes níveis de adversidade na distribuição dos dados.

Simulação do Atraso

A simulação de atraso é um componente fundamental para a avaliação de arquiteturas de aprendizado de máquina federado em cenários mais próximos da realidade, nos quais clientes apresentam diferentes capacidades de comunicação e processamento. Neste trabalho, foi adotada uma metodologia simples e controlada para modelar esses atrasos, evitando detalhes excessivamente técnicos, mas preservando a coerência experimental.

Inicialmente, é gerado um atraso de conexão para cada cliente, representando variações naturais no tempo de comunicação entre os dispositivos e o servidor central. Esses valores são amostrados a partir de um intervalo condizente com atrasos esperados em redes reais, refletindo condições como latência de rede e instabilidade de conexão. Em seguida, é atribuído a cada cliente um atraso de processamento, que modela diferenças de poder computacional, carga de trabalho local e eficiência na execução do treinamento.

A soma do atraso de conexão e do atraso de processamento resulta no atraso total de cada cliente. Com base nesses valores e em um percentual previamente definido de clientes atrasados ou desconectados, é determinado um tempo limite (*timeout*) para cada rodada do treinamento síncrono. Esse *timeout* estabelece até que ponto o servidor aguarda as atualizações dos clientes antes de prosseguir para a próxima rodada.

No caso do aprendizado federado assíncrono, o *timeout* é ajustado de forma proporcional, sendo definido como o *timeout* do síncrono multiplicado pelo número de atualizações por cliente - valor análogo ao número de rodadas no treinamento síncrono. Essa estratégia foi implementada com o objetivo de manter a justiça na comparação entre as arquiteturas síncrona e assíncrona, assegurando que ambas operem sob restrições de tempo equivalentes e sejam avaliadas de maneira equilibrada.

Fórmula Adaptativa de Agregação

No contexto do aprendizado de máquina federado assíncrono, a agregação das atualizações enviadas pelos clientes desempenha um papel central na estabilidade e na eficiência do processo de treinamento. Visando lidar de forma adequada com a heterogeneidade temporal entre os clientes, foi proposta uma fórmula adaptativa de agregação, capaz de ajustar dinamicamente a influência de

cada atualização no modelo global.

A técnica baseia-se na atribuição de um peso adaptativo a cada atualização recebida, calculado a partir de três hiperparâmetros principais: o *base_ alpha*, que define a taxa inicial de contribuição das atualizações dos clientes; o *decay_ of_ base_ alpha*, responsável por controlar o decaimento progressivo dessa taxa ao longo da evolução do treinamento; e a *tardiness_ sensitivity*, que regula o quanto o atraso de um cliente impacta negativamente o peso de sua atualização.

De maneira intuitiva, a fórmula combina dois fatores complementares. O primeiro está associado à convergência do modelo, reduzindo gradualmente a influência das atualizações individuais à medida que o modelo global evolui. O segundo está relacionado ao atraso do cliente, penalizando atualizações baseadas em versões mais antigas do modelo global. Dessa forma, atualizações recentes e pouco atrasadas tendem a exercer maior impacto, enquanto contribuições significativamente defasadas têm sua influência reduzida ou, em casos extremos, são completamente descartadas.

É importante destacar que os valores desses hiperparâmetros não são fixos e variam de acordo com as condições experimentais. No presente trabalho, os parâmetros utilizados foram cuidadosamente ajustados para o cenário considerado, buscando um equilíbrio entre rapidez de convergência e acurácia alcançada.

$$\gamma_t = \alpha_0 \cdot \delta^{\,v_t} \cdot \frac{1}{1 + \lambda \cdot \tau_t} \tag{1}$$

- α_0: *base_ alpha*, correspondente ao peso inicial das atualizações dos clientes;
- δ: fator de decaimento do *base_ alpha* (*decay_ of_ base_ alpha*);
- v_t: versão atual do modelo global;
- λ: *tardiness_ sensitivity*, controlando o impacto do atraso;
- τ_t: atraso da atualização do cliente, medido pela diferença entre a versão do modelo global e a versão utilizada pelo cliente.

$$\mathbf{w}_{t+1} = (1 - \gamma_t)\,\mathbf{w}_t + \gamma_t\,\mathbf{w}_t^{(c)} \tag{2}$$

- $\mathbf{w}_t$: pesos do modelo global atual
- $\mathbf{w}_t^{(c)}$: pesos atualizados do cliente c.

Avaliação Experimental dos Modelos

A avaliação experimental foi conduzida com o objetivo de comparar o desempenho das arquiteturas de aprendizado federado síncrona e assíncrona em diferentes cenários de distribuição de dados e condições de atraso. Para a implementação dos experimentos, foram utilizadas as bibliotecas *TensorFlow*, responsável pela construção e treinamento dos modelos de aprendizado de máquina; *NumPy*, empregada no processamento e manipulação dos dados; *Matplotlib*, utilizada para a visualização dos resultados; e *threading*, adotada para simular o ambiente federado e a execução concorrente dos clientes.

A paridade entre as arquiteturas foi assegurada por meio do controle do tempo de execução, estabelecendo-se um *timeout* proporcional para ambas. Nesse contexto, a desconexão de clientes foi modelada a partir da expiração do tempo limite em cada rodada do aprendizado síncrono ou ao final da execução, no caso do aprendizado assíncrono. Essa estratégia permitiu uma comparação justa, considerando restrições temporais equivalentes para os dois paradigmas.

Os experimentos foram realizados com um total de 40 clientes, cada um executando uma época de treinamento local por atualização, com tamanho de *batch* igual a 32. O conjunto de dados utilizado foi o MNIST, amplamente adotado como referência em tarefas de classificação de imagens. Foram consideradas duas estratégias de distribuição de dados: *IID*, na qual os dados são distribuídos de forma uniforme entre os clientes, e *Não IID*, caracterizada por maior heterogeneidade, permitindo que cada cliente receba amostras de até três classes distintas.

No que diz respeito à modelagem dos atrasos, foram simulados tempos de conexão variando entre 0,1 s e 5 s, bem como atrasos de processamento no intervalo de 0 s a 90 s. Para o treinamento dos modelos locais, foi empregado o otimizador Adam. A arquitetura da rede neural utilizada consiste em um modelo convolucional profundo, adequado à tarefa de classificação multiclasse do MNIST.

No caso do aprendizado federado assíncrono, foi adotada a fórmula adaptativa de agregação, cujos hiperparâmetros foram definidos de acordo com o cenário experimental considerado. Os valores utilizados foram *base_alpha* igual a 0,8, *decay_of_base_alpha* igual a 0,999 e *tardiness_sensitivity* igual a 0,075, buscando equilibrar a influência das atualizações dos clientes, a evolução do modelo global e a penalização associada a atrasos.

A análise dos resultados foi realizada com base em duas métricas principais: a acurácia média obtida ao longo do treinamento e o tempo de execução necessário para a convergência dos modelos. No cenário com distribuição *IID*, observou-se que ambas as arquiteturas alcançam níveis de acurácia semelhantes. Em termos de convergência, a arquitetura assíncrona apresenta uma leve vantagem, convergindo de forma marginalmente mais rápida, embora essa diferença não seja expressiva.

Por outro lado, no cenário com distribuição *Não IID*, os resultados evidenciam maior robustez

da arquitetura assíncrona. Nesse contexto, foi possível observar um desempenho superior em termos de acurácia média, além de uma convergência significativamente mais rápida quando comparada à arquitetura síncrona. Esses resultados reforçam o potencial do aprendizado federado assíncrono em ambientes marcados por elevada heterogeneidade de dados e atrasos, aproximando-se de condições mais realísticas de aplicações distribuídas.

Por fim, também foi realizada uma análise de eficiência energética, considerando a perda de trabalho computacional decorrente de atualizações descartadas ou não incorporadas ao modelo global. Nesse aspecto, a arquitetura assíncrona mostrou-se mais eficiente, uma vez que apresenta menor perda de modelos treinados localmente, indicando redução do desperdício de processamento e, consequentemente, de energia, como evidenciado no terceiro gráfico, que compara diretamente a quantidade de atualizações perdidas em cada abordagem.

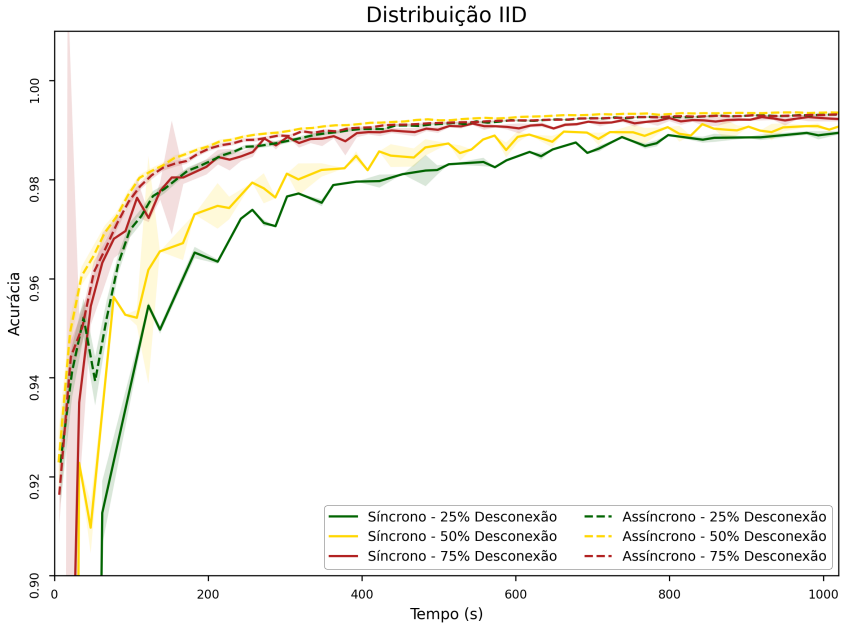

Figura 1: Comparação Média de Acurácia na Distribuição IID.

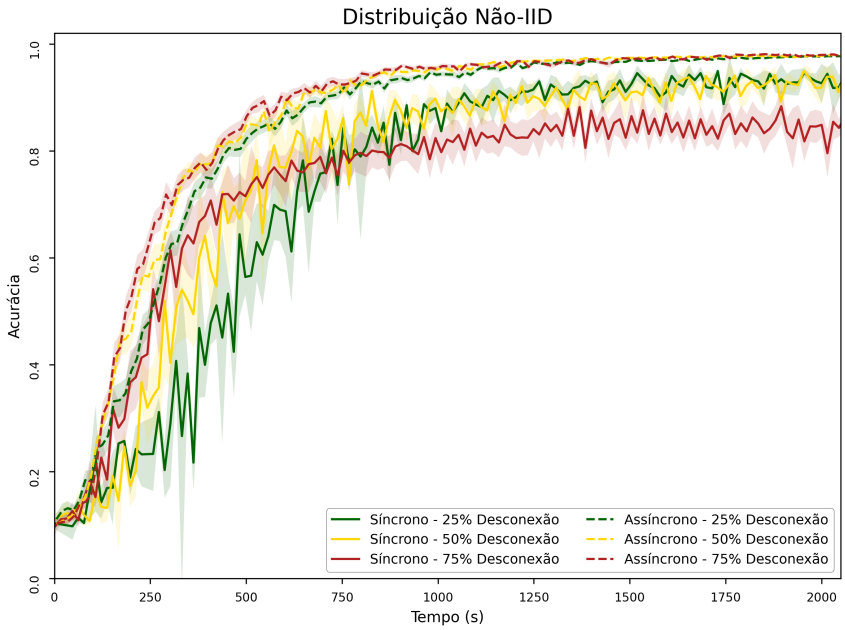

Figura 2: Comparação Média de Acurácia na Distribuição Não-IID.

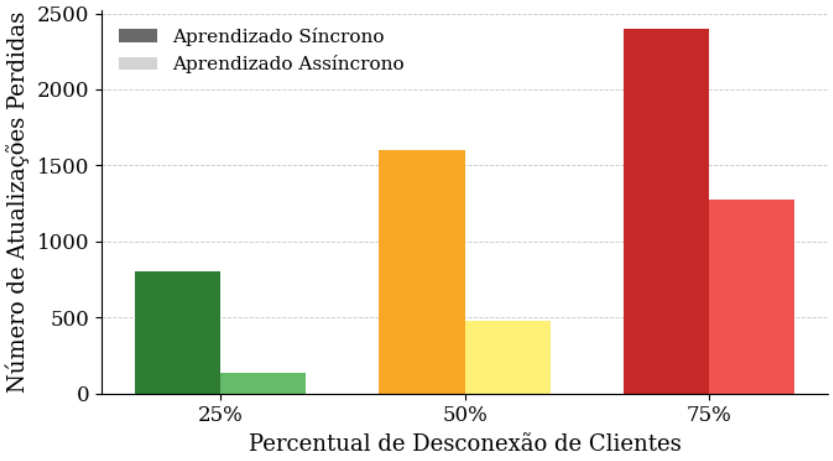

Figura 3: Comparação Média de Perdas de Treinamento para 80 Atualizações por Cliente.

Participação em Evento Científico

Apresentação na SIAc

Em setembro de 2025, o trabalho intitulado "Uma avaliação do impacto do assincronismo no aprendizado de máquina federado" foi apresentado na 14ª Semana de Integração Acadêmica da UFRJ (SIAc). A participação no evento representou um momento relevante para a consolidação dos resultados obtidos ao longo do desenvolvimento do projeto, bem como para a validação das abordagens adotadas frente à comunidade acadêmica.

O ambiente da SIAc favoreceu discussões qualificadas, permitindo o intercâmbio de experiências com pesquisadores e estudantes de diferentes áreas, o que contribuiu para o amadurecimento científico do trabalho. Ademais, a apresentação proporcionou o aprimoramento de competências essenciais, como a exposição estruturada de conceitos técnicos, a argumentação científica e a comunicação eficiente de resultados complexos.

Como reconhecimento de sua relevância e qualidade, o trabalho recebeu o prêmio notório de menção honrosa no evento, destacando-se entre as demais contribuições apresentadas. O resumo introdutório da apresentação pode ser consultado em: https://sistemas2.macae.ufrj.br/siac/cadernoController/gerarCadernoResumo/36000000.

Considerações finais

A relevância do tema abordado neste projeto frente aos desafios contemporâneos evidencia seu grande potencial como uma oportunidade valiosa para a proposição de soluções a problemas concretos, o desenvolvimento prático de tecnologias e o aprofundamento do conhecimento teórico. Nesse sentido, expresso minha sincera gratidão à Fundação de Amparo à Pesquisa do Estado de São Paulo (FAPESP), ao projeto EcoSustain, aos professores, orientadores e a toda a equipe do GTA.

Referências

[1] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, vol. 54, pp. 1273–1282, 2017.

[2] W. Wu, L. He, W. Lin, R. Mao, C. Maple, and S. Jarvis, "SAFA: A semi-asynchronous protocol for fast federated learning with low overhead," *IEEE Transactions on Computers*, pp. 655–668, 2020.

[3] C. Xie, S. Koyejo, and I. Gupta, "Asynchronous federated optimization," *arXiv preprint*, vol. 1903.03934, 2019.

[4] D. J. Beutel, T. Topal, A. Mathur, X. Qiu, J. Fernandez-Marques, Y. Gao, L. Sani, K. H. Li, T. Parcollet, P. P. B. de Gusmão, and N. D. Lane, "Flower: A friendly federated learning research framework," 2022.