# Análise Prática da Técnica de Aprendizado Federado Aplicada a Dispositivos da Internet das Coisas

**Guilherme Nunes Nasseh Barbosa**[1] **e Diogo Menezes Ferrazani Mattos**[1]

[1] LabGen/MídiaCom – TET/IC/PPGEET/UFF
Universidade Federal Fluminense (UFF)
Niterói, RJ – Brasil

{gnasseh,diogo_mattos}@id.uff.br

***Abstract.*** *Internet of Things (IoT) devices have grown exponentially in recent years, resulting in a large body of valuable data for machine learning applications. Traditionally, machine learning models require centralized data collection and processing, which is not feasible in the IoT landscape due to high density and growing data privacy concerns. Federated Learning is a trend in this scenario, as it allows collaborative training of models on IoT devices, distributed and without the need to share data. This paper proposes and evaluates the performance of a federated learning framework for IoT devices. The proposed and evaluated framework employs a parameter server topology.We analyzed the proposal performance on a testbed network composed of IoT devices equipped with ARM processors and limited to 2GB of RAM. The experiments run over a non-identically distributed, and non-independent (non-IID) dataset. The results show that the federated global model reaches 0.6 accuracy, with four clients and ten aggregation rounds, regardless of the local training epochs.*

***Resumo.*** *O número de dispositivos da (Internet of Things - IoT) cresceu exponencialmente nos últimos anos, resultando em uma grande massa de dados útil para aplicações de aprendizado de máquina. Tradicionalmente, os modelos de aprendizado de máquina exigem coleta e processamento de dados centralizados, o que não é viável no cenário IoT, devido à alta densidade e às crescentes preocupações com a privacidade dos dados. O Aprendizado Federado é uma tendência nesse cenário, pois permite o treinamento colaborativo de modelos em dispositivos IoT, distribuído e sem a necessidade de compartilhamento de dados. Este artigo propõe e avalia o desempenho de um arcabouço de aprendizado federado para dispositivos IoT. O arcabouço proposto e avaliado emprega uma topologia de servidor de parâmetros. A análise de desempenho é executada sobre uma rede de testes composta por dispositivos IoT equipados com processadores ARM e limitados a 2GB de RAM. Os experimentos foram executados sobre um conjunto de dados não identicamente distribuído e com dados dependentes (non-IID). Os resultados mostram que o modelo global federado alcança acurácia 0.6, com quatro clientes e 10 rodadas de agregação, independentemente das épocas de treinamento locais.*

Este trabalho foi realizado com recursos do CNPq, CAPES, FAPERJ e RNP e Prefeitura de Niterói/FEC/UFF (Edital PDPA 2020).

## 1. Introdução

O rápido desenvolvimento da Internet das Coisas (*Internet of Things* – IoT) fornece recursos de sensores e computação onipresentes para conectar uma ampla gama de dispositivos à Internet. Para extrair informação dos dados gerados por esses dispositivos IoT, técnicas de aprendizado de máquina têm sido amplamente exploradas para treinar modelos preditivos, como detecção de intrusão, assistência médica, transporte e cidades inteligentes [Medeiros et al., 2019]. Tradicionalmente, o processamento dos dados é feito em um servidor remoto, voltado para o aprendizado e modelagem de dados. Essa abordagem incorre em limitações críticas devido ao grande volume de dados gerados pelo dispositivo IoT [Cunha Neto et al., 2020]. De acordo com a Cisco, cerca de 850 ZB de dados são gerados por todas as pessoas, máquinas e coisas na borda da rede [Alli e Alam, 2019]. Em contrapartida, o tráfego global em *data centers* é apenas 20,6 ZB [1]. Com esse crescimento de dados na borda da rede, o volume de dados enviados para os servidores remotos pode ser inviável devido aos recursos de rede necessários e à latência existente. O uso de servidores de terceiros para treinamento de modelos de aprendizado de máquina também levanta questões de privacidade, como violações e vazamento de informações, pois os dados de treinamento podem conter informações confidenciais, como endereços ou preferências pessoais de usuários [Cunha Neto et al., 2020]. Portanto, torna-se necessário o desenvolvimento de modelos de aprendizado de máquina que sejam colaborativos e eficientes, de modo a garantir a privacidade durante o treinamento de aplicações orientadas à Internet das Coisas.

O aprendizado federado permite o treinamento colaborativo de forma distribuída. Isso permite que diversos participantes treinarem um modelo com ajuda de um servidor central sem compartilhamento de dados [Lim et al., 2020]. Um sistema de detecção de intrusão voltado para dispositivos IoT, cada dispositivo monitora uma rede local e atua como participante do treinamento colaborativo utilizando os dados coletados da rede como conjunto de dados [Neto et al., 2022]. Os dispositivos IoT deverão se comunicar com um servidor agregador para realizar o treinamento colaborativo. Primeiramente, o servidor cria um modelo inicial com os parâmetros aleatórios e cada participante iniciará o treinamento a partir desse modelo. Os participantes recebem o modelo inicial e o atualizam utilizando seu conjunto de dados local. Em seguida, cada participante envia seu modelo atualizado para o servidor agregador. O servidor combina as atualizações de cada modelo local e cria um modelo global. Cabe destacar, que o servidor agrega os modelos locais de apenas um subconjunto de participantes selecionados aleatoriamente. Por fim, os participantes recebem o modelo global agregado e calcula as atualizações com seus dados locais novamente [Nguyen et al., 2021].

Os trabalhos atuais de aprendizado federado em dispositivos IoT geralmente são avaliados através de simuladores [Ciftler et al., 2020, Neto et al., 2022] ou utilizando dispositivos com maior poder computacional do que os dispositivos reais [Zhang et al., 2021, Mills et al., 2019]. É complexo analisar e avaliar os desafios do aprendizado federado para dispositivos IoT em ambientes simulados através de computadores com alto desempenho. Da mesma forma, utilizar dispositivos como *Raspberry*, por exemplo, que possuem desempenho computacional superior comparado a dispositivos IoT tradicionais, também

[1] Disponível em: `https://www.cisco.com/c/en/us/solutions/executive-perspectives/annual-internet-report/index.html`.

são desafiadores. Assim, existe a necessidade de um ambiente de validação com recursos computacionais limitados, para analisar e avaliar as principais propostas de aprendizado federado em ambiente reais de Internet das Coisas.

Este artigo propõe um arcabouço de aprendizado federado orientado à Internet das Coisas para análise e avaliação de propostas de aprendizado federado. O arcabouço proposto emprega a topologia de servidor de parâmetros com diferentes clientes ingressando na federação. Os modelos locais são treinados nos clientes e, então, agregados no servidor de parâmetros. O ambiente de testes utilizado conta com dispositivos IoT equipados com processadores ARM e memória RAM de 2GB. O arcabouço é avaliado com o conjunto de dados MNIST [2]. O conjunto de dados MNIST de dígitos manuscritos tem um conjunto de treinamento de 60.000 amostras e um conjunto de teste de 10.000 amostras. O conjunto de dados foi distribuído de forma *non-IID* para os participantes e o servidor agregador não possui acesso aos dados. O arcabouço foi desenvolvido na linguagem Python utilizando a biblioteca TensorFlow[3]. Os testes realizados mostram que, mesmo executando em nós com pouco poder computacional, o modelo federado alcança acurácia de 0.6 em apenas 10 rodadas de agregação global, independentemente da quantidade de épocas executadas no treinamento dos modelos locais.

O restante do artigo está organizado da seguinte forma. A Seção 2 discute os trabalhos relacionados. Os conceitos relativos ao aprendizado federado em dispositivos IoT são apresentados na Seção 3. Na Seção 4, é apresentado o cenário experimental. A Seção 5 apresenta os resultados. Finalmente, a Seção 6 conclui o artigo.

## 2. Trabalhos Relacionados

O aprendizado federado é uma tecnologia habilitadora para a implantação de aplicações baseadas em aprendizado de máquina em dispositivos da Internet das Coisas. Nguyen *et al.* destacam que o aprendizado federado melhora a privacidade dos dados, permite alcançar latências mais baixas na comunicação em rede e melhora a qualidade do aprendizado em dispositivos restritos em processamento da Internet da Coisas [Nguyen et al., 2021]. Nesse sentido, trabalhos recentes focam no desenvolvimento de novas técnicas de agregação e seleção de participantes para o aprendizado eficiente em dispositivos com recursos restritos [Lai et al., 2021, Neto et al., 2022, Mills et al., 2019] e também focam no desenvolvimento de novas aplicações inteligentes baseadas na federação de modelos [Yu et al., 2018, Wang et al., 2020, Md. Fadlullah e Kato, 2022].

Em um trabalho anterior, Cunha Neto *et al.* propõem a otimização federada de hiperparâmetros do algoritmo da média federada (*Federated Average* - FedAvg) [Neto et al., 2022]. Os autores argumentam que a seleção otimizada de participantes tende a melhorar a acurácia e a diminuir a perda do modelo global federado. Para tanto, os autores propõem o arrefecimento simulado federado (*Federated Simulated Annealing* - FedSA), uma extensão da meta-heurística de arrefecimento simulado (*Simulated Annealing* - SA) para um cenário de execução distribuída em que a função objetivo tende a sofrer modificações a cada nova rodada. Os resultados mostram que a proposta é capaz de alcançar a mesma acurácia da agregação com média federada [Hard et al., 2018], porém com menos rodadas de agregação e com menor número de participantes. Visando também

[2]Disponível em http://yann.lecun.com/.

[3]O código fonte está disponível sob consulta aos autores.

a melhora do desempenho do algoritmo de aprendizado federado, Mills *et al.* propõem adaptar o FedAvg para usar uma forma distribuída de otimização de Adam, reduzindo o número de rodadas para convergência, juntamente com as novas técnicas de compressão, para tornar o FedAvg eficiente quanto à transmissão de pesos. Lai *et al.* propõem o método de agregação Oort [Lai et al., 2021]. A proposta Oort prioriza a seleção de participantes que tenham tanto dados que ofereçam a maior utilidade na melhoria precisão do modelo, quanto que tenham capacidade de executar o treinamento rapidamente. A proposta Oort impõe requisitos na distribuição de dados do participante, reduzindo o tempo de execução do aprendizado federado.

Para resistir ao risco de vazamento de privacidade devido à transmissão frequente de dados entre a borda e a nuvem, o aprendizado federado aplica a computação de borda, carregando o modelo atualizado do servidor de borda para o servidor central para agregação, em vez de transferir dados diretamente. Contudo, um servidor de borda mal intencionado pode inferir a atualização de outros a partir do modelo agregado e a atualização ainda pode expor algumas características dos dados de outros servidores ou, até mesmo, ataques de envenenamento bizantinos. Liu *et al.* propõem esquema FL de preservação de privacidade com agregação robusta em computação de borda denominado FL-RAEC [Liu et al., 2022]. Ao enviar o modelo para o servidor na nuvem, um mecanismo híbrido de preservação da privacidade é construído para garantir a integridade e a privacidade dos dados. No estágio de agregação do modelo, é proposta uma estratégia de agregação em fases com detecção de anomalias e verificação de confiança anônima, estimada pelo resultado da detecção de anomalia baseada em *autoencoders*. A privacidade diferencial local consiste em adicionar um ruído à atualização do peso para satisfazer o requisito de privacidade, mensurado pela distância de código Hamming. De forma semelhante, Lu *et al.* integram um mecanismo de privacidade diferencial local, para preservar a privacidade das atualizações locais, ao treinamento com o gradiente descendente para permitir o compartilhamento seguro e robusto no aprendizado federado. A fim de reduzir custos de comunicação, os autores propõem um modelo de aprendizado federado descentralizado que permite agregar atualizações de modelos de participantes em servidores distribuídos [Lu et al., 2020].

A dependência de uma nuvem remota para operação de aprendizado federado pode resultar em longa latência de comunicação. Portanto, Zhou *et al.* apresentam uma otimização para coordenar dispositivos de borda e nuvem minimizando a latência de comunicação. O agendamento de dados compartilhados e o controle de admissão, juntamente com o ajuste de precisão, são otimizados em conjunto. Os resultados da simulação verificam a viabilidade do algoritmo proposto com latência reduzida e aumento da privacidade em várias configurações de rede [Zhou et al., 2020]. Chiu *et al.* observam que, em aplicações reais, os dados nos dispositivos finais são não independentes e não são identicamente distribuídos (non-IID) [Chiu et al., 2020]. A distribuição dos dados pode, então, causar divergência de peso durante o treinamento e resultar em uma diminuição considerável no desempenho do modelo federado. Assim, os autores propõem a operação chamada troca federada (FedSwap) para substituir as operações de aprendizagem federadas parciais com base em alguns dados compartilhados durante o treinamento federado. Um esquema de aprendizado semissupervisionado é adotado para prever objetos para aplicações de análise de vídeo entre dispositivos de borda.

Savazzi *et al.* propõem uma abordagem de aprendizado totalmente distribuída, sem servidor agregador de parâmetros [Savazzi et al., 2020]. A ideia central é que os algoritmos de aprendizado federado promovam a cooperação de dispositivos na rede revezando entre iterações de computações locais e interações mútuas por meio de métodos baseados em mecanismos de consenso. A metodologia proposta é verificada pelos conjuntos de dados experimentais coletados dentro de um ambiente IoT Industrial (*Industrial Internet of Things* - IIoT). Kong *et al.* também focam no desenvolvimento de um arcabouço de aprendizado federado para o ambiente IIoT [Kong et al., 2020]. Os autores desenvolvem uma estrutura de tensor federada para a mineração de dados em IoT industrial. A proposta integra dados de várias fontes e permite a mineração baseada em tensor com garantias de segurança. Fábricas cooperam para ingressar na mineração com tensor, compartilhando seus dados, que foram criptografados usando uma técnica de criptografia homomórfica, com um servidor centralizado. Assim, o servidor apenas coleta os dados cifrados e os federa em um tensor, enquanto os dados brutos são mantidos nas fábricas locais, protegendo a privacidade dos dados. Embora os bisbilhoteiros possam atacar o servidor centralizado para comprometer o texto cifrado agregado e os invasores possam ler o texto cifrado nos canais de comunicação, eles não podem obter a chave para a descriptografia dos dados. Embora esses trabalhos visem desenvolver aplicações federadas para ambientes massivos em dispositivos IoT, as propostas não consideram que os dispositivos tenham capacidades limitadas.

Os trabalhos apresentados visam garantir a privacidade dos dados e melhorar o desempenho das técnicas de aprendizado federado. Em diversos casos focam na execução de operações locais em dispositivos com *hardware* superdimensionado ou na simulação de dispositivos IoT. Contudo, os trabalhos falham em prover um arcabouço de experimentação real em dispositivos IoT. Assim, este trabalho propõe um arcabouço para execução do aprendizado federado em dispositivos IoT. São realizados testes reais de execução do aprendizado federado em dispositivos com capacidades reduzidas de *hardware*, diferentemente de trabalhos anteriores.

## 3. Aprendizado Federado em Dispositivos IoT

O aprendizado federado é um tipo de aprendizado de máquina que permite que vários dispositivos, ou clientes, treinem de forma colaborativa um modelo compartilhado, mantendo seus dados armazenados localmente e privados. Contrário ao aprendizado de máquina tradicional, em que um servidor centralizado é responsável por coletar e processar todos os dados para o treinamento do modelo, no aprendizado federado, cada dispositivo contém dados privados e participa do processo de treinamento do modelo enviando suas atualizações locais para o servidor central, que agrega as atualizações para criar um modelo global aprimorado. O aprendizado federado é uma especialização do aprendizado descentralizado sobre uma topologia em árvore de altura unitária. Nessa topologia, os nós são organizados em uma estrutura de árvore hierárquica, em que cada nó se comunica apenas com seu nó pai e nós filhos. No caso do aprendizado federado, o nó pai é um servidor de parâmetros, responsável pela agregação dos modelos locais enviados pelos nós filhos.

Em um cenário de aprendizado federado, os dados são sempre processados nos dispositivos que os possuem, garantindo a privacidade dos dados e diminuindo a necessidade por banda para a transmissão dos dados. Ao treinar um modelo com dados de várias fontes, o aprendizado federado tende a melhorar a precisão e a generalização do modelo.

O algoritmo básico do aprendizado federado resume-se às seguintes etapas [Cunha Neto et al., 2020]:

1. **Inicialização**. O servidor centralizado de parâmetros inicializa um modelo de aprendizado de máquina e o compartilha com os clientes;
2. **Seleção de cliente**. O servidor seleciona um subconjunto de clientes pertencentes ao conjutno total de clientes disponíveis para treinamento de modelo. A seleção pode seguir diferentes critérios ou ser aleatória [Neto et al., 2022];
3. **Treinamento de modelo local**. Cada cliente selecionado baixa o modelo atual do servidor e o treina localmente usando seus próprios dados. O cliente então envia os parâmetros do modelo atualizados (não os dados brutos) de volta ao servidor;
4. **Agregação de modelo**. O servidor recebe as atualizações de modelo locais de todos os clientes selecionados e agrega os modelos para a criação de um novo modelo global. O principal algoritmo de agregação é a Média Federada (*Federated Averaging* – FedAvg);
5. **Atualização dos modelos locais**. Após a agregação de um novo modelo global, o servidor distribui o novo modelo a todos os clientes participantes do treinamento. Os clientes, então, passam a utilizar o novo modelo proposto pelo servidor como base para o treinamento das rodadas seguintes;
6. **Repetição**. As etapas 2 a 5 constituem uma rodada de execução do treinamento do modelo. Portanto, são repetidas por várias rodadas até que o modelo global atenda um critério de parada predefinido. O critério critério de parada pode ser definido como número máximo de rodadas, acurácia esperada ou qualquer outra métrica de qualidade do modelo.

A média federada (*Federated averaging*) é um método de agregação comumente usado para a atualização de modelo recebidos de vários clientes participantes [Cunha Neto et al., 2020] O método é bastante usado em modelos de aprendizado baseados em redes neurais, pois o treinamento do modelo pode ser expresso por vetores de pesos que representam cada camada da rede neural treinada. A ideia básica da média federada é calcular uma média ponderada das atualizações do modelo local recebidas de cada cliente para criar um novo modelo global. Assim, dado um conjunto de $N$ clientes, cada cliente $i$ realiza treinamento local em seus próprios dados e calcula um vetor de atualização $\delta w_i$. Ao receber os vetores de atualização de todos os clientes, o servidor calcula o novo modelo global tomando uma média ponderada das atualizações:

$$w_{t+1} = \frac{\sum_i^n w_{t,i} * n_i}{\sum_i^n n_i}, \tag{1}$$

em que $w_{t,i}$ é o vetor de pesos do modelo do cliente $i$ na rodada $t$ e $n_i$ é o número de amostras de dados usados pelo cliente $i$ para treinamento local. O novo modelo global com vetor de pesos $w_{t+1}$ é então usado por todos os clientes para o treinamento da próxima rodada de aprendizado federado. Ressalta-se que o vetor de pesos contém os pesos de todas as camadas para modelo de aprendizado baseado em rede neural.

A Figura 1 apresenta a proposta de aplicação do aprendizado federado em uma rede de dispositivos IoT. Como os dispositivos IoT são limitados em processamento e memória, a proposta considera que o servidor de parâmetros executa em uma plataforma de computação em nuvem, enquanto os participantes são dispositivos IoT. A comunicação

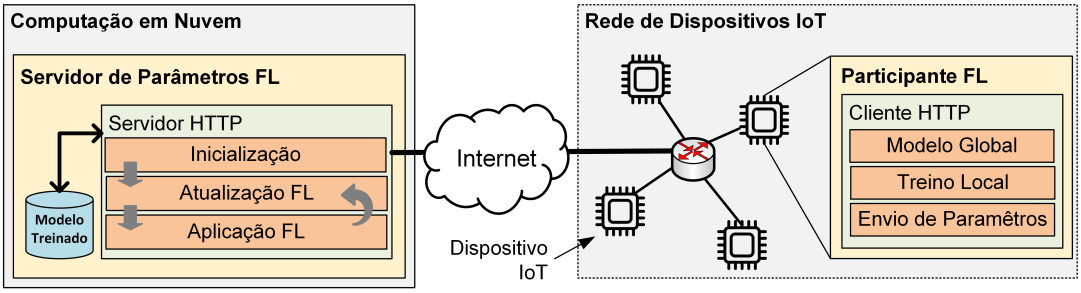



**Figura 1. Arcabouço de aprendizado federado para dispositivos IoT. O servidor de parâmetros executa em uma plataforma de computação em nuvem e implementa microsserviços sobre um servidor HTTP. O cliente de aprendizado federado executa um cliente HTTP que busca o modelo global no servidor, treina o modelo local e envia os parâmetros para o servidor.**

entre os participantes e o servidor de parâmetros é realizada pelo protocolo HTTP. Para tanto, o modelo e os vetores de parâmetros são serializados em uma mensagem binária e, então, codificada em *base64*. A mensagem codificada em *base64* é transmitida como conteúdo dos pacotes HTTP. A arquitetura adotada permite que o código que executa no participante seja simples o suficiente para não comprometer sua capacidade. Assim, o participante executa somente ações de chamadas HTTP e codificação de mensagens em *base64* em paralelo ao treinamento do modelo local.

## 4. Cenário Experimental

O treinamento e validação do cenário foram realizados utilizando algoritmos de aprendizado de máquina para o conjunto de dados MNIST. Esse conjunto compreende diversas imagens de dígitos escritos a mão, com 60.000 amostras para treinamento e 10.000 amostras para validação. O conjunto é frequentemente utilizado para processamento de imagens e, no cenário proposto, foi utilizado por ser versátil e pequeno, uma vez que os dispositivos utilizados possuem baixa capacidade de armazenamento para realizar o treinamento com outros conjuntos. Para realizar o treinamento distribuído foram utilizados dispositivos IoT conhecidos como *TV Box*[4], como clientes, equipados com com processadores ARM-54 Cortex-A53 com frequência de 1,2GHz e 2GB de RAM. Os equipamentos executam sistema operacional Ubuntu 22.04 LTS, com *kernel* Linux 5.9.0-arm-64 com suporte a múltiplos núcleos de processamento. O servidor de agregação é executado em uma máquina virtual com sistema operacional Ubuntu 22.10, *kernel* Linux 5.15.0-69-generic com 2 vCPU e 2GB de RAM, executada em um ambiente OpenStack [5]. A linguagem de programação utilizada foi o Python 3.7 [6] com as bibliotecas Sklearn[7] e TensorFlow[8]. O cenário avaliado é apresentada na Figura 2. A mérito de simplificação para a realização de testes, tanto o servidor de parâmetros, executando sobre o OpenStack, quanto todos os clientes estão conectados por uma rede local. Cada cliente é inicializado com uma

[4]O termo *TV Box* refere-se a equipamentos de baixo custo normalmente comercializados como *set top boxes* para serviços de IPTV. Os equipamentos usados neste trabalho foram doados pela Receita Federal do Brasil à Universidade Federal Fluminense para a descaracterização do produto e destinação ao uso social.

[5]Disponível em https://www.openstack.org/

[6]Disponível em https://www.python.org/.

[7]Disponível em https://scikit-learn.org/stable/.

[8]Disponível em https://www.tensorflow.org/.

partição do conjunto de dados. A partição dos dados entre os clientes é desbalanceada e os dados são não identicamente distribuídos e não independentes (non-IID).

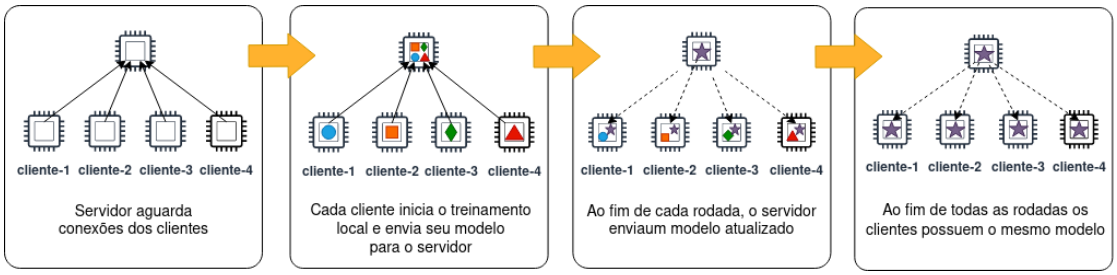



**Figura 2. Cenário experimental adotado. Cada cliente detém uma parte do conjunto de dados. Os dados são não identicamente distribuídos e não independentes (non-IID). Os dados são locais dos clientes e não são compartilhados na rede. A troca de mensagens resume-se aos parâmetros dos modelos.**

O modelo de treinamento adotado no cenário experimental é o de rede neural convolucional. Para tanto é usada a rede *LeNet-5*, uma rede neural convolucional simples, conforme mostrado na Figura 3. As redes neurais convolucionais são comumente utilizadas no processamento de imagens em larga escala, pois apresentam desempenho satisfatório na segmentação de mapa de bits [Andreoni e Mattos, 2021]. O modelo considera como entrada um mapa de bits de dimensão $28 \times 28$ com um canal. A função de ativação usada é a tangente hiperbólica. Contudo, na última camada densa, é usada a função de ativação *softmax*. As camadas de regularização aplicam a média à janela de regularização (*AveragePooling2D*).

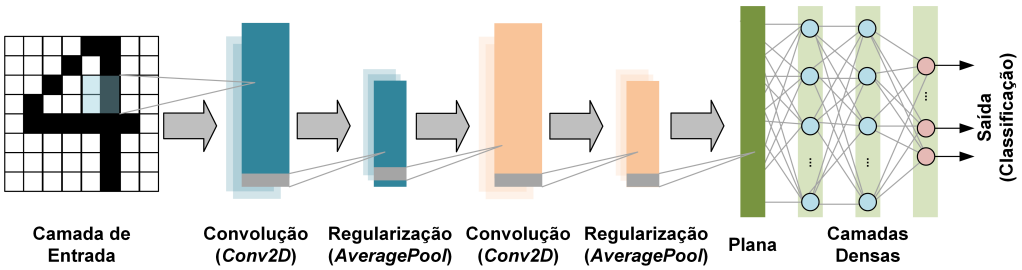
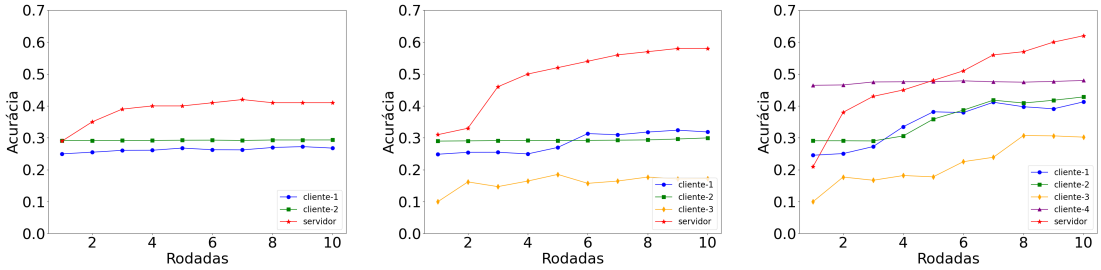



**Figura 3. Modelo de rede neural *LeNet-5* treinado no aprendizado federado. O modelo é do tipo *feedforward* com oito camadas, sendo elas duas camadas convolucionais intercaladas por camadas de regulação, e ao final, uma camada de planificação antecede três camadas densas.**

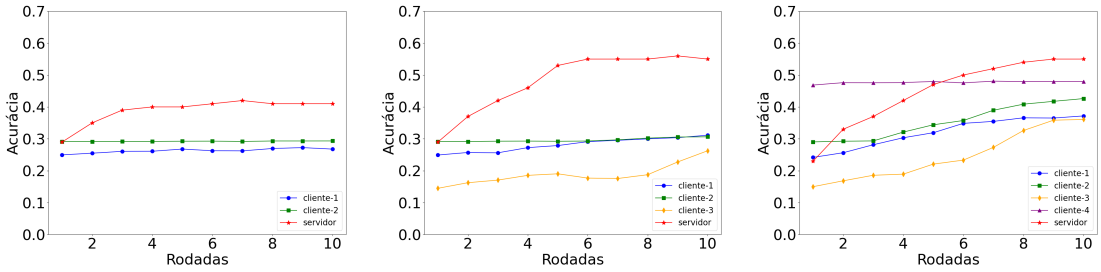
## 5. Resultados Experimentais

Inicialmente, o conjunto de dados foi particionado de maneira que cada cliente recebeu uma quantidade diferente de dados, selecionados de forma aleatória, para que fossem distribuídos de forma não balanceada, não independente e não identicamente distribuído. Foram avaliados seis cenários distintos. O primeiro cenário avaliou o treinamento com dois clientes, dez rodadas e dez épocas locais. O segundo e terceiro cenários avaliaram com três e quatro clientes, respectivamente, mantendo o mesmo número de rodadas e épocas locais do primeiro. O quarto cenário contemplou o treinamento também com dois clientes, mantendo as dez rodadas de agregação, porém utilizando vinte épocas locais de treinamento do modelo. O quinto e sexto cenários também utilizaram três e quatro clientes respectivamente, ambos com vinte épocas locais e dez rodadas de agregação




(a) 2 clientes. (b) 3 clientes. (c) 4 clientes.

**Figura 4. Treinamento distribuído utilizando um conjunto de dados não balanceado com 10 rodadas de agregação global e 10 épocas de treinamento local. a) Treinamento utilizando 2 clientes. b) Treinamento utilizando 3 clientes. c) Treinamento utilizando 4 clientes.**



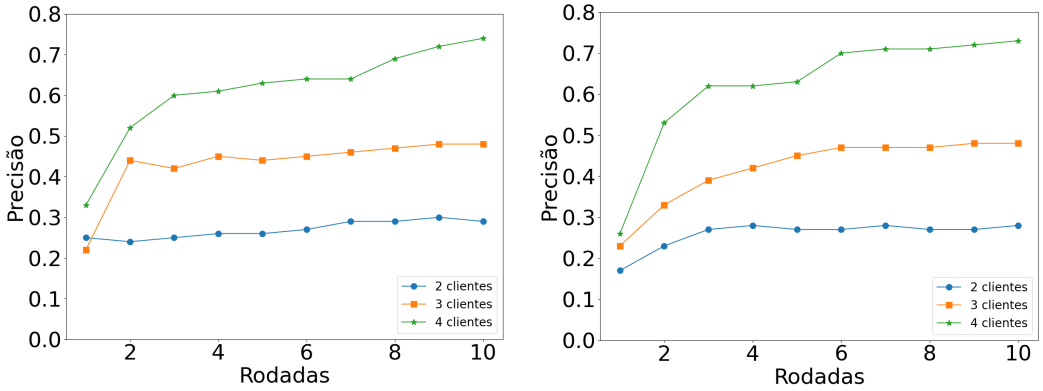

(a) 2 clientes. (b) 3 clientes. (c) 4 clientes.

**Figura 5. Treinamento distribuído utilizando um conjunto de dados não balanceado com 10 rodadas e 20 épocas locais a) Treinamento utilizando 2 clientes. b) Treinamento utilizando 3 clientes. c) Treinamento utilizando 4 clientes.**

global. A Figura 4 apresenta os três cenários avaliados utilizando dez épocas locais, variando o número de clientes selecionados para o treinamento do modelo. A Figura 5 apresenta o treinamento utilizando vinte épocas locais.

É possível avaliar que nos cenários que utilizaram apenas dois clientes, o modelo global apresentou pouco incremento de acurácia no decorrer de cada rodada. Isso indica que estes clientes tiveram uma baixa contribuição para o treinamento, por possuírem juntos, poucos dados de treinamento. Nos cenários com três clientes, apresentados nas Figuras 4(b) e 5(b), com dez e vinte épocas locais respectivamente, houve uma melhora na acurácia do servidor de agregação. No cenário de dez épocas locais, entre a segunda e terceira rodada, houve um melhora significativa na acurácia do servidor com um aumento de aproximadamente 40%. No cenário com vinte épocas locais, o incremento na acurácia foi de aproximadamente 14%. Esse fato pode ser considerado apenas uma variação pontual no treinamento ou algo ocasionado pela distribuição do conjunto de dados. Por fim, os cenários que utilizaram quatro clientes, com dez e vinte épocas locais apresentados nas Figuras 4(c) e 5(c) mostram que o servidor de agregação teve uma melhora significativa de acurácia comparado com treinamento de dois clientes, com um aumento de aproximadamente 52% para o caso de dez épocas e 34% para o caso de vinte épocas. Entretanto, a variação entre o cenário de três e quatro clientes foi menor, apresentado 7% de melhora no caso de dez épocas locais. Não houve melhora percentual significativa para vinte épocas. Cabe destacar que um menor número de clientes pode influenciar diretamente na acurácia local, de modo que seja constante. Por outro lado, é possível que alguns clientes possuam

dados que melhorem o modelo global, mas não influencie a acurácia do modelo local.




(a) 10 épocas locais (b) 20 épocas locais

**Figura 6. Precisão do modelo validado no servidor de agregação. a) Utilizando 10 épocas locais nos clientes b) Utilizando 20 épocas locais nos clientes**

Embora o servidor de agregação tenha apresentado uma pequena variação na acurácia quando comparado os cenários com três e quatro clientes, houve uma melhora significativa na acurácia quando quatro clientes são utilizados para treinamento. Esse fato evidencia o fato de que o cliente-4 possui um número significativo de amostras, o que torna-se importante para o treinamento do modelo global. Aproximadamente 40% do conjunto de dados original está alocado no cliente-4. Como o conjunto de dados foi distribuído de forma desbalanceada, tais variações são esperadas. A contribuição do cliente-4 para o treinamento global pode ser observada pelo fato de que, no momento em que este cliente que possui mais informações inicia o treinamento, todos os demais clientes apresentam uma melhora significativa na acurácia.

Outra métrica utilizada para a medida da qualidade do modelo gerado no servidor de agregação foi a precisão, mostrada na Figura 6. Como o conjunto de dados possui dez classes distintas (dígitos de 0 a 9), o valor da precisão é a média aritmética ponderada de todas as classes. Foi possível notar que não houve diferença significativa entre os cenários de dez ou vinte épocas locais nos clientes. No entanto, a precisão varia de acordo com número de clientes e entre as classes.

## 6. Conclusão

O crescimento acelerado da adoção de dispositivos de Internet das Coisas (*Internet of Things* - IoT) propicia a implantação de poder processamento ubíquo e onipresente em objetos do dia a dia. Embora limitados em poder de processamento e memória, os dispositivos de IoT mantém grande parte de seu processamento ocioso durante a operação. Esse artigo propôs e avaliou um arcabouço para implantação de um mecanismo de aprendizado federado entre dispositivos IoT. O aprendizado federado permite que diferentes clientes participantes treinem um modelo global cooperativamente, contribuindo com o vetor de pesos do seu modelo local. O aprendizado federado entre dispositivos IoT permite que cada dispositivo execute o seu treinamento local, em uma quantidade reduzida de dados, de forma privada, e com pouco comprometimento do seu poder de computação. Em paralelo, o conjunto de dispositivos treina cooperativamente um modelo de aprendizado com desempenho comparável ao modelo treinado por algoritmos de aprendizado de máquina

tradicionais. O artigo implementou um protótipo de aprendizado federado em equipamentos IoT de baixo custo com poder computacional restrito e avaliou o modelo global sobre o conjunto de dados MNIST. Os resultados alcançados mostram que a inclusão de mais participantes influenciam a qualidade do modelo, mas o aumento no número de épocas de treinamento local tem pouca influência na qualidade final. O artigo limita-se a avaliar o comportamento do aprendizado federado em um ambiente sem nós maliciosos e, também, não avalia técnicas para melhorar o desempenho do modelo alcançado. Como trabalhos futuros, vislumbram-se implementar o arcabouço proposto em um cenário híbrido e aprofundar a avaliação de desempenho de recurso computacional dos dispositivos.

## Agradecimentos

Os autores agradecem à Receita Federal do Brasil pela doação dos equipamentos usados como dispositivos de Internet das Coisas. Os autores também agradecem a Helio do Nascimento Cunha Neto e Yago Rezende pelo apoio na definição do cenário experimental.

## Referências

Alli, A. A. e Alam, M. M. (2019). SecOFF-FCIoT: Machine learning based secure offloading in fog-cloud of things for smart city applications. *Internet of Things*, 7:100070.

Andreoni, M. e Mattos, D. M. F. (2021). Resumo de grandes volumes de dados com filtro de bloom: Uma abordagem eficiente para aprendizado profundo com redes neurais convolucionais em fluxos de rede. Em *Anais do XXXIX Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, p. 532–545, Porto Alegre, RS, Brasil. SBC.

Chiu, T.-C., Shih, Y.-Y., Pang, A.-C., Wang, C.-S., Weng, W. e Chou, C.-T. (2020). Semisupervised distributed learning with non-IID data for AIoT service platform. *IEEE Internet of Things Journal*, 7(10):9266–9277.

Ciftler, B. S., Albaseer, A., Lasla, N. e Abdallah, M. (2020). Federated learning for RSS fingerprint-based localization: A privacy-preserving crowdsourcing method. Em *2020 International Wireless Communications and Mobile Computing (IWCMC)*, p. 2112–2117.

Cunha Neto, H. N., Mattos, D. M. F. e Fernandes, N. C. (2020). Privacidade do usuário em aprendizado colaborativo: Federated learning, da teoria à prática. *Minicursos do Simpósio Brasileiro de Segurança de Informação e de Sistemas Computacionais - SBSeg*, 20:142–195.

Hard, A., Rao, K., Mathews, R., Ramaswamy, S., Beaufays, F., Augenstein, S., Eichner, H., Kiddon, C. e Ramage, D. (2018). Federated learning for mobile keyboard prediction. *arXiv preprint arXiv:1811.03604*.

Kong, L., Liu, X.-Y., Sheng, H., Zeng, P. e Chen, G. (2020). Federated tensor mining for secure industrial internet of things. *IEEE Transactions on Industrial Informatics*, 16(3):2144–2153.

Lai, F., Zhu, X., Madhyastha, H. V. e Chowdhury, M. (2021). Oort: Efficient federated learning via guided participant selection. Em *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*.

Lim, W. Y. B., Luong, N. C., Hoang, D. T., Jiao, Y., Liang, Y. C., Yang, Q., Niyato, D. e Miao, C. (2020). Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys & Tutorials*.

Liu, W., Xu, X., Li, D., Qi, L., Dai, F., Dou, W. e Ni, Q. (2022). Privacy preservation for federated learning with robust aggregation in edge computing. *IEEE Internet of Things Journal*.

Lu, Y., Huang, X., Dai, Y., Maharjan, S. e Zhang, Y. (2020). Differentially private asynchronous federated learning for mobile edge computing in urban informatics. *IEEE Transactions on Industrial Informatics*, 16(3):2134–2143.

Md. Fadlullah, Z. e Kato, N. (2022). HCP: Heterogeneous computing platform for federated learning based collaborative content caching towards 6G networks. *IEEE Transactions on Emerging Topics in Computing*, 10(1):112–123.

Medeiros, D. S. V., Cunha Neto, H. N., Andreoni Lopez, M., Magalhães, L. C. S., Silva, E. F., Vieira, A. B., Fernandes, N. C. e Mattos, D. M. F. (2019). Análise de dados em redes sem fio de grande porte: Processamento em fluxo em tempo real, tendências e desafios. *Minicursos do Simpósio Brasileiro de Redes de Computadores-SBRC*, 2019:142–195.

Mills, J., Hu, J. e Min, G. (2019). Communication-efficient federated learning for wireless edge intelligence in IoT. *IEEE Internet of Things Journal*, 7(7):5986–5994.

Neto, H. N. C., Dusparic, I., Mattos, D. M. F. e Fernande, N. C. (2022). FedSA: Accelerating intrusion detection in collaborative environments with federated simulated annealing. Em *2022 IEEE 8th International Conference on Network Softwarization (NetSoft)*, p. 420–428.

Nguyen, D. C., Ding, M., Pathirana, P. N., Seneviratne, A., Li, J. e Vincent Poor, H. (2021). Federated learning for internet of things: A comprehensive survey. *IEEE Communications Surveys & Tutorials*, 23(3):1622–1658.

Savazzi, S., Nicoli, M. e Rampa, V. (2020). Federated learning with cooperating devices: A consensus approach for massive IoT networks. *IEEE Internet of Things Journal*, 7(5):4641–4654.

Wang, X., Wang, C., Li, X., Leung, V. C. M. e Taleb, T. (2020). Federated deep reinforcement learning for internet of things with decentralized cooperative edge caching. *IEEE Internet of Things Journal*, 7(10):9441–9455.

Yu, Z., Hu, J., Min, G., Lu, H., Zhao, Z., Wang, H. e Georgalas, N. (2018). Federated learning based proactive content caching in edge computing. Em *2018 IEEE Global Communications Conference (GLOBECOM)*, p. 1–6.

Zhang, T., He, C., Ma, T., Gao, L., Ma, M. e Avestimehr, S. (2021). Federated learning for internet of things. Em *Proceedings of the 19th ACM Conference on Embedded Networked Sensor Systems*, p. 413–419.

Zhou, Z., Yang, S., Pu, L. e Yu, S. (2020). CEFL: Online admission control, data scheduling, and accuracy tuning for cost-efficient federated learning across edge nodes. *IEEE Internet of Things Journal*, 7(10):9341–9356.