Aprendizado Federado com Agrupamento Hierárquico de Clientes para Aumento da Acurácia

Lucas Airam C. de Souza[1], **Gustavo F. Camilo**[1],
Matteo Sammarco[2], **Miguel Elias M. Campista**[1], **Luís Henrique M. K. Costa**[1]

[1]Grupo de Teleinformática e Automação (GTA)
Universidade Federal do Rio de Janeiro (UFRJ)
[2]AXA

Resumo. *O desempenho do aprendizado federado depende da distribuição dos dados, mostrando alta degradação no cenário de clientes com dados heterogêneos. Este artigo propõe um esquema hierárquico de agrupamento de clientes para contornar os desafios gerados por distribuições de dados não Independentes e Identicamente Distribuídas (não-IID) no aprendizado federado. Os grupos criados possuem clientes nos quais os dados possuem distribuições aproximadamente IID, facilitando a convergência do modelo. O sistema possui uma fase de inicialização, na qual o servidor executa um algoritmo de aprendizado não supervisionado de agrupamento sobre o vetor de bias da última camada da rede neural dos clientes. O algoritmo DBSCAN demonstrou melhores resultados de agrupamento, identificando corretamente os grupos até quando todos os clientes possuem conjuntos de dados com distribuições IID. Por fim, os resultados mostram o aumento de até 16% da acurácia em relação à abordagem de aprendizado federado tradicional quando os clientes possuem conjuntos de dados não-IID.*

Abstract. *Federated learning performance depends on the data distribution, deteriorating in scenarios in which clients hold heterogeneous data. We propose a hierarchical client clustering system to mitigate the performance problems of federated learning in non-Independent and Identically Distributed (IID) scenarios. Our proposal efficiently groups clients with approximately IID data distribution, achieving fast and accurate model convergence. We initialize the system executing a clustering unsupervised learning algorithm on the bias vector of the last layer of the clients' neural network in the server. The DBSCAN algorithm demonstrated better clustering results, correctly identifying clusters even when all clients have datasets with IID distributions. Finally, the results show an increase of model accuracy up to 16% compared to the traditional federated learning non-IID scenarios.*

1. Introdução

O aprendizado federado, proposto pelo Google [McMahan et al. 2017], tornou-se popular entre pesquisadores e a indústria por sua capacidade de criar modelos de aprendizado de máquina preservando a privacidade dos dados dos usuários [de Souza et al. 2020, Bochie et al. 2021]. A sua adoção ganhou mais relevância após a mudança nas regulamentações sobre processamento dos dados em diversos países, por exemplo, a Lei Geral de Proteção de Dados (LGPD) no Brasil e as Regulamentações Gerais de Proteção de Dados (*General Data Protection Regulations* - GDPR) no continente europeu.

Este trabalho foi realizado com recursos do CNPq, CAPES, FAPERJ e FAPESP (E-26/203.211/2017, E-26/202.689/2018; 15/24494-8, 18/23292-0, 2015/24485-9 e 2014/50937-1.)

O algoritmo mais utilizado para criação de modelos federados é o FedAVG (*Federated Averaging*), apresentado na proposta inicial de aprendizado federado do Google, que utiliza modelos de aprendizado baseados no vetor gradiente de perda para o ajuste dos parâmetros. A proposta do aprendizado federado, para uma distribuição de dados horizontal, segue uma arquitetura cliente-servidor, na qual o servidor envia o modelo a ser criado aos clientes que por sua vez o treinam com dados locais. Os resultados da computação local são enviados ao servidor, que os agrega ao modelo global calculando a média das respostas dos clientes e retorna o novo modelo para os clientes. Essa iteração ocorre até algum critério de parada ser atingido, como convergência do modelo ou número máximo de iterações alcançado. Assim, o conjunto de dados dos clientes permanece secreto durante todo o treinamento, garantindo a privacidade de suas amostras. Entretanto, a distribuição dos dados dos clientes que participam do treinamento impacta o desempenho final do modelo. Ambientes de aprendizado federado nos quais os clientes possuem dados não Independentes e Identicamente Distribuídos (não-IID) entre si, apresentam dificuldades de convergência do modelo ou degradação no desempenho final [Zhao et al. 2018]. Os dados não-IID tornam as atualizações dos clientes discordantes, produzindo vetores gradientes de perda divergentes.

Este artigo propõe um sistema de agrupamento hierárquico de clientes[1] para aumentar a eficiência do aprendizado federado em cenários nos quais os clientes possuam conjuntos de dados não-IID. Os clientes são divididos em grupos nos quais os dados são similares. Para preservar a privacidade dos clientes, a proposta utiliza os pesos da rede neural dos clientes como informação para realizar o agrupamento. Os pesos da rede neural mantêm relações estatísticas com os dados privados dos clientes sem revelá-los [Wang et al. 2020]. Cada grupo criado possui um modelo específico, que pode variar tanto nos hiperparâmetros configurados quanto nos parâmetros ajustados pelos clientes do grupo, permitindo alto desempenho de classificação em tarefas específicas. Além disso, a proposta considera possíveis comportamentos maliciosos dos clientes participantes do ambiente federado, como o envio de gradientes incorretos para prejudicar a convergência do modelo após a criação dos grupos. Como contramedida, o sistema compara os vetores gradientes em cada época e filtra vetores anômalos.

Os experimentos realizados analisam diferentes modelos de agrupamento e mostram a vantagem de adotar o algoritmo de agrupamento espacial de aplicativos com ruído baseado em densidade (*Density-Based Spatial Clustering of Applications with Noise* - DBSCAN) [Ester et al. 1996] para o agrupamento dos clientes devido a sua capacidade de identificar distribuições de dados homogêneas a partir dos pesos da rede neural dos clientes. O DBSCAN supera os algoritmos de ordenar os pontos para identificar a estrutura de agrupamento (*Ordering Points to Identify the Clustering Structure* - OPTICS) [Ankerst et al. 1999] e o KMeans [MacQueen et al. 1967]. Além disso, os resultados mostram que a abordagem proposta possui desempenho igual ou superior ao aprendizado federado tradicional. Para conjuntos de dados não-IID, o algoritmo FedAVG converge com baixa acurácia após as 400 épocas globais de treinamento, enquanto a proposta atual alcança melhores resultados em todos os grupos em 10 rodadas globais, proporcionando um alto desempenho de classificação.

Este artigo está organizado da seguinte forma. A Seção 2 revisa o estado da arte em otimização do desempenho de sistemas de aprendizado federado através da seleção de clientes. A Seção 3 apresenta o sistema proposto em detalhes. A Seção 4 apresenta o desenvolvimento de um protótipo do sistema, as configurações do ambiente e a análise dos resultados obtidos. Por fim, a Seção 5 conclui este trabalho e discute direções futuras.

[1]Disponível em https://github.com/GTA-UFRJ-team/Hierachical-Federated-Learning.

2. Trabalhos Relacionados

A velocidade de convergência do modelo global no aprendizado federado é atribuída a três fatores principais: latência de comunicação, capacidade de processamento e representatividade dos dados coletados. Os dois primeiros fatores são relacionados ao dispositivo utilizado, enquanto o terceiro depende da distribuição dos dados dos dados coletados e armazenados pelo cliente. Porém, a seleção aleatória de clientes, exibida na proposta original do aprendizado federado [McMahan et al. 2017], desconsidera essas três características relevantes. Assim, diversos grupos de pesquisa apresentam propostas para otimizar o tempo de convergência do aprendizado federado e o desempenho final do modelo gerado. As principais formas de otimização incluem a seleção de clientes e a personalização de modelos. A seleção de clientes foca as características dos dispositivos ou dos dados dos clientes para decidir quais os melhores clientes para participar da época de treinamento atual, aumentando o desempenho médio global do modelo. Por outro lado, a personalização de modelos provê uma forma de criar modelos especializados nas características individuais dos clientes, com objetivo de aumentar o desempenho do modelo local.

2.1. Seleção de Clientes para o Treinamento Eficiente

Luo *et al.* [Luo et al. 2021] e Lai *et al.* [Lai et al. 2021] propõem esquemas de seleção de clientes que buscam otimizar a velocidade de convergência do modelo em ambientes de aprendizado federado. Os autores argumentam que a seleção baseada apenas na representatividade dos dados diminui o número total de épocas para convergência do modelo. Entretanto, quando há clientes com maior latência computacional, o tempo entre épocas aumenta significativamente. Por outro lado, a seleção baseada apenas na capacidade de processamento pode incorrer em um número maior de rodadas para a convergência caso os clientes selecionados possuam dados pouco significativos estatisticamente. Assim, os autores consideram simultaneamente as características dos dispositivos e as distribuições dos dados coletados, para reduzir o tempo de convergência do modelo global. Wang *et al.* utilizam o aprendizado por reforço para seleção eficiente dos clientes [Wang et al. 2020]. O agente do aprendizado por reforço é treinado previamente para selecionar os melhores clientes a cada época de treinamento e prover o melhor grupo de clientes dado o estado atual do modelo global. As informações estatísticas dos clientes nas propostas apresentadas são estimadas através do vetor de gradiente de perda enviado ao servidor de agregação para manter a privacidade dos dados dos clientes.

Neto *et al.* [Neto et al. 2021] propõem uma meta heurística baseada no arrefecimento simulado para o ajuste dos hiperparâmetros e a seleção eficiente de clientes. A partir do estado atual do modelo, a proposta seleciona um determinado grupo de clientes para realizar o treinamento da época atual. Nishio e Yonetani propõem um protocolo para seleção de clientes no aprendizado federado [Nishio e Yonetani 2019] no qual os clientes com maiores capacidades de processamento e menor latência de comunicação são priorizados no esquema de seleção apresentado. Liu *et al.* propõem uma arquitetura hierárquica para o aprendizado federado [Liu et al. 2020], que realiza agregações locais nos clientes, parciais na borda e globais na nuvem. O processamento de dados na borda diminui a latência de comunicação, porém o número restrito de dispositivos pode dificultar a convergência do modelo global. O processamento na nuvem, entretanto, consegue acessar um número maior de dispositivos e capturar uma maior variância dos dados, mas aumenta a latência de comunicação. Assim, os autores aplicam um treinamento federado em níveis, cliente-borda e borda-nuvem. Dessa forma, os clientes mantêm uma baixa latência de comunicação com a borda, sendo que a nuvem consegue acessar os modelos gerados nas bordas. Entretanto, os autores desconsideram a distribuição dos

dados gerados pelos clientes, o que pode afetar o desempenho dos modelos em cenários com distribuições não-IID.

A seleção de clientes diminui o tempo de convergência e aumenta a acurácia final do modelo. Entretanto, para ambientes com dados extremamente heterogêneos, a seleção pode ser insuficiente para efeitos práticos. Assim, uma alternativa é a personalização de modelos que foca a divisão dos clientes em grupos para a criação de modelos mais específicos, a fim de aumentar o desempenho final do modelo para cada cliente.

2.2. Personalização de Modelos no Aprendizado Federado

Dennis *et al.* [Dennis et al. 2021] aproveitam a heterogeneidade dos dados para produzir um algoritmo de aprendizado federado não supervisionado. Os clientes treinam um modelo de agrupamento com seus dados locais e enviam ao servidor de agregação os vetores que indicam a posição dos grupos encontrados no espaço amostral. O servidor, então, executa um segundo modelo de agrupamento, sobre as respostas dos clientes, para identificar os grupos existentes no ambiente. Quanto maior a distância entre as distribuições dos clientes, mais eficaz é a detecção dos diferentes grupos. A proposta possui aplicação tanto para a seleção de clientes quanto para a personalização de modelos.

O IFCA (*Iterative Federated Clustering Algorithm*) [Ghosh et al. 2020] é uma proposta de agrupamento de clientes para personalização dos modelos. Na proposta, os clientes são responsáveis por escolher seus grupos. Além disso, os autores propõem o uso do aprendizado multi-tarefas (*multi-task learning*), que consiste no compartilhamento de alguns pesos da rede neural, para clientes que possuem distribuições de dados com interseções, mas estão em grupos diferentes. Porém, por delegar o processo de identificação de grupos aos clientes, o ambiente pode ser suscetível a comportamentos maliciosos e exige maior dispêndio computacional dos dispositivos dos clientes. Outra desvantagem da proposta é assumir que o número de grupos é conhecido a princípio, o que pode ser inviável e gerar a criação de grupos de forma desnecessária ou subestimar a quantidade de grupos existentes.

O CFL [Sattler et al. 2020] é uma proposta que particiona recursivamente os clientes do aprendizado federado em grupos mais homogêneos para mitigar os problemas gerados por distribuições não-IID. O particionamento ocorre sempre que o vetor de gradiente de perda ultrapassa um limiar de distância pré-estabelecido. Contudo, a partição recursiva dos clientes gera uma sobrecarga computacional no servidor de agregação por ser executada a cada época global do treinamento. O ClusterFl [Ouyang et al. 2021] é um arcabouço para criação de grupos homogêneos de clientes em ambientes de aprendizado federado. Os autores propõem a verificação periódica dos grupos para detectar clientes ineficientes, mantendo o tempo de convergência baixo para o modelo do grupo. Entretanto, a proposta pode apresentar alta sobrecarga computacional desnecessária, pois os autores preveem o retreinamento periódico dos modelos. Além disso, ambas propostas não aplicam a combinação de modelos entre grupos para clientes com tarefas genéricas.

Diferentemente das propostas anteriores de seleção de clientes que consideram apenas a otimização de um modelo genérico em todo o sistema de aprendizado federado, este artigo propõe um sistema de personalização de modelos através do agrupamento de clientes. O sistema agrupa os clientes com a intenção de aproximar os clientes com dados mais homogêneos entre si. Dessa forma, os grupos possuem dados IID, beneficiando individualmente os participantes do sistema. O agrupamento de clientes facilita a convergência do modelo no grupo e aumenta o seu desempenho final para tarefas de aprendizado específicas. A determinação dos grupos é realizada a partir do treinamento não supervisionado de um algoritmo de agrupamento

que recebe alguns pesos da rede neural dos clientes como vetor de entrada. Portanto, a proposta mantém os requisitos de privacidade do aprendizado federado tradicional, pois não há compartilhamento de dados privados. Além disso, a proposta é agnóstica ao algoritmo de agrupamento utilizado, eliminando a necessidade de conhecer a quantidade de grupos existentes de forma prévia. Por fim, a proposta considera comportamentos maliciosos dos clientes durante o treinamento e provê mecanismos de mitigação para as ameaças consideradas.

3. Sistema de Agrupamento Hierárquico de Clientes

Esta seção apresenta os detalhes da proposta para criação de modelos acurados com o agrupamento hierárquico de clientes. Primeiramente é abordado o agrupamento de clientes para a criação de modelos específicos, e posteriormente discute-se como formar modelos mais genéricos através da combinação dos modelos finais dos grupos. Por fim, apresenta-se a discussão sobre possíveis ataques ao sistema e como estes são mitigados pela proposta. O sistema é aplicado em um ambiente de aprendizado federado horizontal, no qual os clientes coletam as mesmas características para formar os conjuntos de dados, p. ex., imagens coloridas com a mesma quantidade de pixels.

O modelo gerado no grupo é específico, pois sua construção é dada a partir de conjuntos de dados IID. Assim, o seu desempenho é alto para um conjunto de dados com distribuições similares. As trocas de mensagens entre os clientes e o servidor são criptografadas e apenas o servidor conhece os clientes. Isso evita que agentes maliciosos obtenham informações sobre outros clientes e as utilizem para degradar o modelo, assumindo que o servidor não foi comprometido. Além disso, assume-se que o servidor é honesto, executando o algoritmo FedAVG para agregação dos gradientes corretamente e a combinação dos modelos sob demanda. Ademais, a proposta prevê outros mecanismos para mitigar possíveis ameaças ao treinamento.

Este artigo assume que os clientes podem ter comportamentos maliciosos. Um dos ataques mais relevantes é o envenenamento de modelo [Sun et al. 2019] onde um agente malicioso envia atualizações incorretas para o servidor de agregação na tentativa de corromper o modelo global. Para prevenir que agentes maliciosos danifiquem o modelo global de um grupo, a proposta prevê o monitoramento constante das atualizações. Assume-se que os clientes possuem dados estacionários de forma que a distância entre os vetores de gradiente de perda individuais dos clientes selecionados e o vetor médio do grupo para uma época específica não ultrapassa o limiar previamente estabelecido. Portanto, caso um cliente malicioso envie uma atualização incorreta, o servidor de agregação identifica facilmente através da comparação do vetor gradiente de perda enviado pelo cliente com relação ao vetor médio. Além disso, as configurações armazenadas no servidor de agregação não são reveladas aos clientes, dificultando a reconstrução local do algoritmo de agrupamento pelo cliente malicioso de modo a criar um vetor incorreto que não seja detectado. Os clientes desconhecem os demais clientes e suas atualizações e não são comunicados quando seu vetor foi agregado ou não. Por fim, outras medidas de segurança podem ser adicionadas ao processo de treinamento [Desai et al. 2021].

Devido à natureza distribuída do aprendizado federado, outro ataque possível é o ataque de Sybil [Douceur 2002] para o atacante aumentar a sua probabilidade de seleção no treinamento e influenciar o resultado do modelo. Para mitigar essa ameaça, a proposta assume que os clientes são autenticados pelo servidor antes de participar do treinamento. O artigo não entra em maiores detalhes de implementação, pois essas etapas estão fora do escopo.

3.1. Operação do Sistema Proposto

O sistema proposto para o treino de modelos federados possui duas fases após a inicialização. A primeira fase aloca os clientes em grupos de forma unívoca, enquanto a se-

gunda realiza um treinamento de aprendizado federado tradicional entre os clientes de cada grupo. A divisão dos clientes visa minimizar a heterogeneidade dos dados utilizados para o treinamento do modelo global. Para isso, inicialmente o servidor executa uma rodada global de aprendizado federado selecionando todos os clientes da rede. Os clientes calculam localmente com seus dados a atualização dos pesos da rede neural e retornam o resultado ao servidor. Após receber a resposta dos clientes, o servidor executa um algoritmo de agrupamento para determinação da quantidade de grupos e pertencimento dos clientes. Dessa forma, clientes com pesos da rede neural próximos são alocados em um mesmo grupo homogêneo.

As etapas para criação dos grupos ocorrem sem a necessidade de acessar as amostras dos clientes para manter a privacidade dos dados. A partir da criação dos grupos, o aprendizado federado ocorre de maneira tradicional no grupo. Portanto, há no sistema proposto no mínimo k modelos específicos, onde k é o número de grupos gerados pelo servidor. Além disso, os clientes podem gerar sob demanda combinações dos modelos dos grupos a fim de formar modelos genéricos que são capazes de classificar amostras geradas em distribuições distintas. Diferentemente do aprendizado federado tradicional, a proposta denomina o servidor do aprendizado federado como servidor de seleção e agregação, pois além de agregar os resultados, o servidor é responsável por selecionar e armazenar o grupo de cada cliente.

3.2. Inicialização do Sistema Proposto

O sistema assume que ao inicializar, há clientes suficientemente representativos para ajustar o modelo de agrupamento do sistema e estabelecer todos os grupos existentes possíveis. Assim, as etapas da Figura 1 são executadas para inicializar o sistema. Na primeira etapa, o servidor compartilha o modelo de teste com um conjunto inicial de clientes. Os clientes ajustam os pesos do modelo de teste com seus dados locais e retornam o vetor de *bias* da última camada para o servidor de seleção e agregação. O servidor executa o algoritmo de agrupamento e salva os grupos existentes. Uma vez que os grupos tenham sido estabelecidos, novos clientes não poderão gerar novos grupos, apenas alocados em grupos existentes. O modelo inicial dos grupos pode diferir do modelo de teste, pois o servidor pode ajustar os hiperparâmetros do modelo utilizado segundo os dados de cada grupo. A entrada de novos clientes em um grupo só afeta o treinamento se o modelo não convergiu.

Após a convergência do modelo, os novos clientes apenas recebem o modelo final do grupo. Além disso, a proposta pressupõe que os clientes possuem dados estacionários e prevê um número fixo de grupos após a inicialização. Caso todos os clientes de um grupo falhem simultaneamente, o servidor de agregação armazena as informações do grupo, estado do treinamento e modelo atual, pois eventualmente novos clientes podem ser alocados no grupo ou os clientes em falha serem recuperados. O algoritmo de agrupamento não é executado para identificar a existência de novos grupos, porém é utilizado para identificar ações maliciosas e designar novos clientes aos grupos existentes. O sistema é agnóstico ao modelo de agrupamento utilizado permitindo ao administrador selecionar o algoritmo de agrupamento que melhor o atenda. A proposta assume que o administrador possui informações relevantes para ajustar os algoritmos de agrupamento antes de definir os grupos, p. ex., classes existentes e um pequeno conjunto de dados próprio. Outra hipótese é que as tarefas de aprendizado são similares, p. ex., classificação de imagens.

3.3. Dinâmica dos Clientes no Sistema

A entrada de novos clientes no ambiente é exibida na Figura 2. O cliente novo que deseja participar do treinamento federado solicita ao servidor de seleção e agregação um novo grupo. O servidor envia ao cliente um modelo teste, idêntico para todos os clientes novos. Após

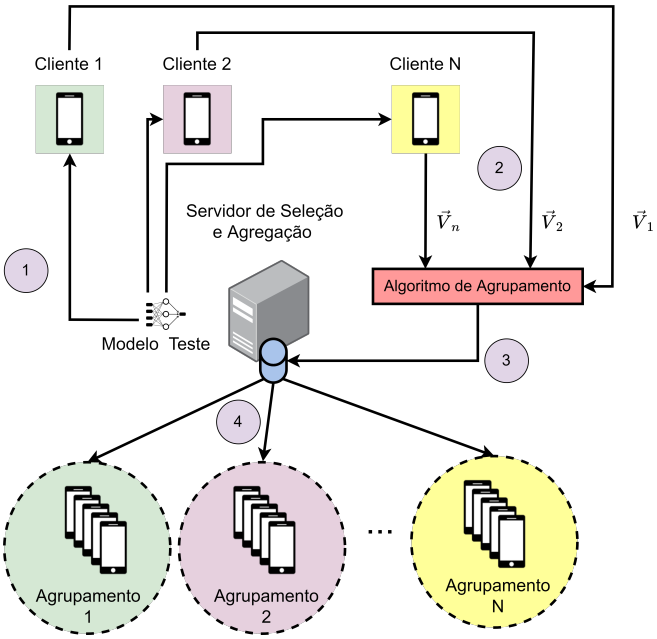

Figura 1: Inicialização do sistema proposto. (1) O conjunto inicial de clientes recebe o modelo de teste. (2) Os clientes ajustam os pesos da rede neural e retornam ao servidor o vetor de *bias* da última camada do modelo. (3) O servidor de seleção e agregação executa o algoritmo de agrupamento para obter os agrupamentos do sistema. (4) O servidor associa novos clientes aos agrupamentos.

essa etapa, o cliente calcula a atualização do modelo teste e remete o resultado ao servidor. Com as respostas dos clientes, o servidor executa um algoritmo de agrupamento e associa o cliente a um grupo. Finalmente, o aprendizado federado é executado de forma tradicional de forma independente em cada grupo.

Os clientes são agrupados de acordo com a atualização do modelo enviada ao servidor de agregação. Entretanto, foi verificado experimentalmente que as camadas finais do modelo retêm maior informação sobre os dados. Assim, pode-se otimizar a comunicação do sistema, enviando apenas parte da rede neural profunda para o servidor. Portanto, o sistema desenvolvido utiliza apenas os vetores de viés da última camada para o cálculo dos grupos, diminuindo a quantidade de dados transferida.

A Figura 2 detalha as etapas desenvolvidas pela proposta para a criação dos modelos do sistema. O cliente solicita ao servidor a inclusão em um grupo. O servidor remete ao cliente o modelo teste para obter informações estatísticas. Essa etapa é fundamental na proposta, pois garante uma forma de agrupar os clientes sem revelar a privacidade dos dados. O cliente atualiza o modelo teste com os seus dados privados e envia parte do resultado ao servidor de seleção e agregação. Após receber a resposta do cliente, o servidor executa o algoritmo de agrupamento ajustado na inicialização do sistema para determinar a qual grupo o novo cliente pertence. A informação do grupo é enviada ao cliente e sua identidade é adicionada à lista do grupo. O servidor, por fim, envia ao cliente o modelo m_i atual do grupo, permitindo a sua participação no treinamento federado. A partir dessa etapa, o treinamento federado ocorre de forma tradicional no grupo g_i no qual o cliente foi alocado. A cada época E_i do grupo, o servidor seleciona aleatoriamente uma fração de clientes para ajustar os parâmetros do modelo do grupo. Caso o cliente seja selecionado, calcula os novos pesos p_i e os envia ao servidor para sua agregação. Por fim, o servidor envia aos clientes do grupo g_i o modelo m_i atualizado e inicia-se uma nova época, $E_{i'}$, do grupo. O processo é repetido até que uma condição de parada seja alcançada,

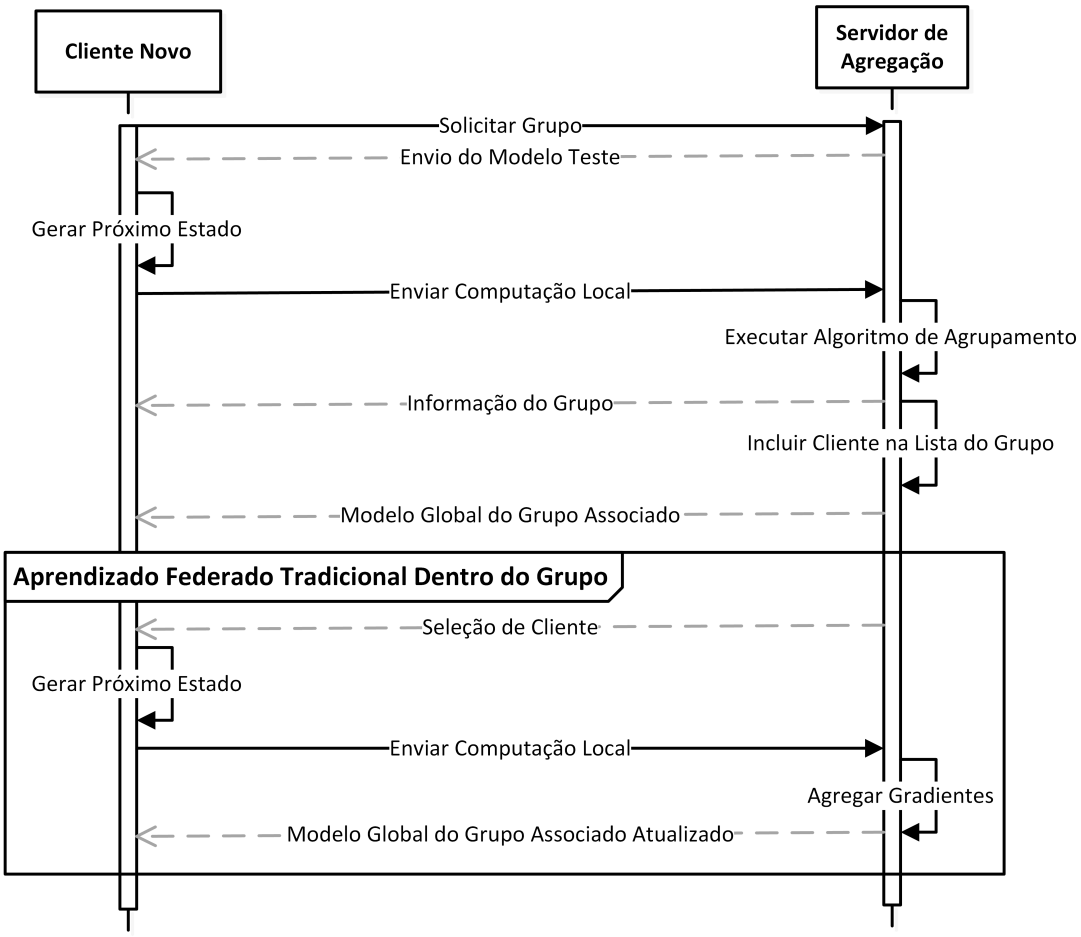

Figura 2: Diagrama de execução proposto. A primeira etapa consiste em alocar um novo cliente em um grupo existente. Após a alocação do cliente em um grupo, o aprendizado federado tradicional é executado no grupo.

como um critério de convergência do desempenho do modelo, p. ex., acurácia dentro de um limiar de baixa variação em épocas consecutivas, ou um número máximo de épocas globais executadas $E_{i_{max}}$.

4. Desenvolvimento do Protótipo e Resultados Obtidos

Um protótipo do sistema foi desenvolvido utilizando a linguagem de programação Python v3.9.1 com arcabouço flower v0.18.0 [Beutel et al. 2020] para o desenvolvimento do ambiente de aprendizado federado e a biblioteca scikit-learn v1.0.1 [Pedregosa et al. 2011] para criação dos modelos de agrupamento. Os experimentos deste trabalho foram realizados em um servidor Intel i7-8700 CPU 3.20 GHz com 6 núcleos de processamento e 32 GB de RAM. Os resultados experimentais da avaliação dos modelos apresentam a média obtida entre todos os clientes com um intervalo de confiança de 95%.

Os cenários avaliados utilizam os valores de configuração exibidos na Tabela 1. A probabilidade de seleção exibida na tabela equivale ao percentual de clientes selecionados para o treinamento de forma aleatória. Na proposta atual, a seleção aleatória é realizada dentro do grupo. Além disso, os resultados de acurácia e perda de teste são construídos a partir da seleção de todos os clientes a cada época global. Após a alocação dos clientes em grupos, os hiperparâmetros são ajustados de acordo com os dados do grupo, *i.e.*, redução do número de neurônios na camada final devido à ausência de todas as classes no grupo. Portanto, o servidor

Tabela 1: Principais parâmetros aplicados ao ambiente de aprendizado federado.

Parâmetro	Configuração
Número de Clientes	10
Probabilidade de Seleção de Cliente	50%
Número de Épocas Globais	400
Número de Épocas Locais	5
Tamanho dos Lotes Locais	32

Tabela 2: Análise do número de grupos gerados pelos algoritmos de agrupamento em diferentes distribuições de dados. O algoritmo DBSCAN é configurado com uma distância mínima de 0,0279, enquanto o número mínimo de amostras do algoritmo OPTICS é configurado em 2.

Cenário	IID		Não-IID 2 Classes		Não-IID 5 Classes	
Algoritmo	DBSCAN	OPTICS	DBSCAN	OPTICS	DBSCAN	OPTICS
Número de Grupos	1	5	5	5	2	2

é responsável por executar o processo de ajuste de hiperparâmetros em cada grupo, para o melhor desempenho dos modelos. Para o treinamento dos modelos e a avaliação de desempenho da proposta é utilizado o conjunto de dados CIFAR-10 [Krizhevsky et al. 2009]. O conjunto de dados é constituído de 60.000 imagens coloridas com dimensão 32x32 pertencentes a 10 classes distintas. Cada classe possui a mesma quantidade de amostras.

4.1. Avaliação dos Modelos de Agrupamento

A primeira etapa do sistema agrupa os clientes a partir do vetor de atualização enviado ao servidor. Assim, o Experimento I visa avaliar formas de agrupar os clientes com base nas informações fornecidas. Nesse experimento são analisados os hiperparâmetros de três algoritmos de agrupamento: K-Means [MacQueen et al. 1967], OPTICS [Ankerst et al. 1999] e DBSCAN [Ester et al. 1996].

O K-Means é um algoritmo amplamente adotado para tarefas de agrupamento. O seu principal hiperparâmetro é o número de grupos k existentes em um determinado conjunto de dados. Entretanto, a definição desse hiperparâmetro em um conjunto de dados desconhecido é uma tarefa árdua, pois há necessidade prévia de conhecimento do número de grupos existentes. Assim, há um custo computacional maior para utilizar uma abordagem iterativa e estimar o melhor número de grupos. Desta forma, o OPTICS surge como uma alternativa mais prática, pois requer como hiperparâmetro o número mínimo de amostras para formar um grupo. Por outro lado, configurar um número mínimo de amostras por grupo grande pode gerar grupos heterogêneos devido à restrição imposta. O agrupamento busca minimizar a distância entre amostras do mesmo grupo. O KMeans define o número mínimo de grupos, enquanto o OPTICS define o número mínimo de amostras por grupo. Por essa restrição, algumas amostras podem ser alocadas em grupos iguais, mesmo com uma grande distância. Finalmente, uma alternativa para os algoritmos anteriores é o DBSCAN. Além de ser possível configurar um número mínimo de participantes para a criação de um grupo, o DBSCAN utiliza como hiperparâmetro a distância máxima entre duas amostras. Portanto, é possível estabelecer de maneira prática uma relação direta entre as respostas dos clientes para definir se possuem dados IID.

Os resultados de agrupamento mostram que o algoritmo que melhor separou os clientes ao variar os cenários foi o DBSCAN utilizando uma distância mínima de 0.0279 entre os vetores dos clientes. Nos três cenários o DBSCAN agrupou de forma ótima os clientes, supe-

rando o OPTICS no cenário IID ao identificar apenas um grupo. Além disso, os três modelos de agrupamento geraram os mesmos grupos de clientes nos cenários não IID. A causa desse comportamento se deve ao fato do K-Means ser configurado com o número ótimo de grupos e os grupos possuírem distâncias que permitem identificar facilmente os grupos. Os conjuntos de dados não IID são organizados de forma que há no mínimo dois clientes com distribuições similares, justificando assim o como o OPTICS e o DBSCAN identificam os mesmos grupos que o KMeans. Os próximos experimentos avaliam os modelos gerados após o agrupamento dos clientes.

Os três algoritmos foram avaliados com relação ao agrupamento dos clientes para diferentes hiperparâmetros de inicialização. A Tabela 2 exibe o número de grupos gerados e os valores de hiperparâmetros para os algoritmos DBSCAN e OPTICS. Os valores para o K-Means são omitidos da tabela, pois o hiperparâmetro do algoritmo indica diretamente o número de grupos. O algoritmo K-Means estabelece um limiar ótimo para os experimentos de avaliação de desempenho da proposta, pois como os cenários não-IID são criados de forma controlada, o número de grupos ótimo é conhecido previamente. Para o caso IID, o K-Means é configurado de forma a criar grupos com no mínimo dois clientes a fim de garantir a consistência do aprendizado federado. O objetivo dessa configuração é verificar o impacto na acurácia quando o número de grupos é superestimado, dividindo os clientes de forma desnecessária. Assim, três é o número de grupos que mantém o mínimo de dois clientes por grupo para o K-Means.

Tabela 3: Avaliação do intervalo de distâncias para identificar os 5 grupos homogêneos com o algoritmo DBSCAN em função do número de épocas locais executadas pelos clientes.

Distância Mínima		Distância Máxima		Épocas Locais
2 classes por cliente	5 classes por cliente	2 classes por cliente	5 classes por cliente	
0,0023	0,0508	0,0488	0,4496	5
0,0023	0,0083	0,0475	0,0904	10
0,0024	0,0067	0,0568	0,0939	20
0,0048	0,0063	0,0722	0,1184	50

Uma avaliação relevante é determinar a distância que permite ao DBSCAN identificar os grupos cujos dados são IID nos cenários experimentais. Assim, é necessário variar o hiperparâmetro de distância e verificar quantos grupos o modelo retorna. A distância entre os pesos da rede neural dos clientes depende fortemente do número de épocas locais dos clientes. Aumentar o número de épocas locais implica especializar as redes neurais nos dados de treinamento e, assim, produzir vetores mais distantes entre si. Por outro lado, clientes que possuem conjuntos de dados com distribuições similares, ao especializar seus pesos produzem vetores que continuam próximos. A Tabela 3 exibe os valores encontrados para as distâncias mínimas e máximas que podem ser atribuídas ao DBSCAN de modo que o modelo determine o número correto de grupos existentes. O resultado indica que utilizar 10 épocas locais, com uma distância entre [0,0083, 0,0475], permite identificar corretamente os grupos existentes. Assim, a implementação adota o valor para o hiperparâmetro de distância do algoritmo DBSCAN o valor de 0.0279, que é o valor médio do intervalo.

4.2. Avaliação do Desempenho dos Modelos Específicos

O segundo experimento avalia o comportamento da proposta quando os conjuntos de dados de todos os clientes são IID. O objetivo é avaliar o desempenho da proposta ao utilizar diferentes algoritmos de agrupamento e verificar o impacto na acurácia quando há mais grupos

do que o necessário. Para criar grupos com no mínimo 2 clientes, inicialmente o valor do hiperparâmetro de grupos do algoritmo K-Means foi configurado para 5 e reduzido até formar grupos com o mínimo de participantes. Assim, através dessa abordagem, o Experimento II utiliza $k = 3$. O OPTICS foi configurado com o número mínimo de clientes igual a 2, e também gerou grupos de forma desnecessária. Por outro lado, o DBSCAN ao analisar os vetores dos clientes, identificou de forma correta haver apenas um grupo, pois o conjunto de dados é IID.

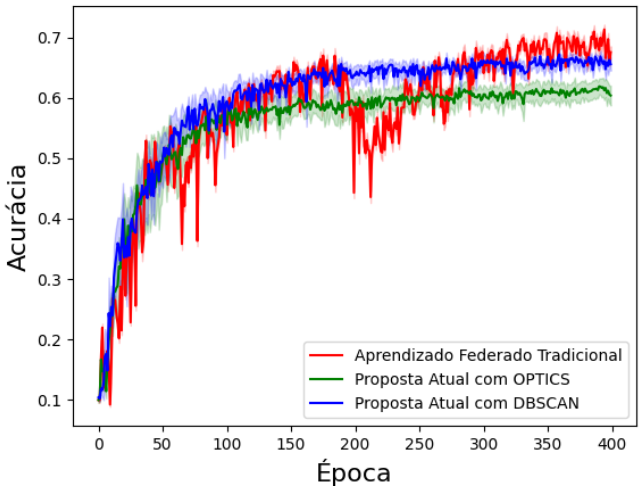

Figura 3: Evolução da acurácia em função da época global para clientes com conjuntos de dados IID. A linha vermelha representa a abordagem tradicional de aprendizado federado, enquanto a linha azul representa a proposta deste artigo com o agrupamento através do DBSCAN e a verde ilustra o desempenho do OPTICS.

O resultado exibido na Figura 3 indica o comportamento médio da acurácia em todos os 10 clientes utilizando a abordagem tradicional e a proposta atual. A proposta, mesmo utilizando o K-Means e o OPTICS para o agrupamento, possui o valor final de acurácia próximo ao caso tradicional, apesar de dividir os clientes em grupos de forma desnecessária nesse experimento. Assim, pode-se concluir ausência perda significativa de desempenho mesmo quando os clientes possuem conjuntos de dados IID e divididos em grupos. Além disso, o experimento demonstra a capacidade do algoritmo DBSCAN identificar que a criação de grupos é desnecessária nesse caso. A Tabela 3 demonstra que o comportamento em cada grupo é similar ao comportamento médio exibido na Figura 3.

No Experimento III as distribuições de dados nos clientes são não-IID, com apenas duas classes presentes em seus conjuntos de dados. O objetivo desse experimento é comparar o desempenho do modelo gerado através do aprendizado federado tradicional com o criado pela proposta atual, no qual a distribuição dos dados é não-IID. Para isso, as 6.000 amostras de uma classe são dividas entre cinco clientes, para criar conjuntos de dados balanceados. A Figura 4 exibe o resultado médio experimental entre todos os grupos, ilustrado na figura como a linha azul, e o resultado tradicional, representado pela linha vermelha. O aprendizado federado tradicional é afetado pelo nível de heterogeneidade dos dados, enquanto a proposta atual permite um alto desempenho de classificação.

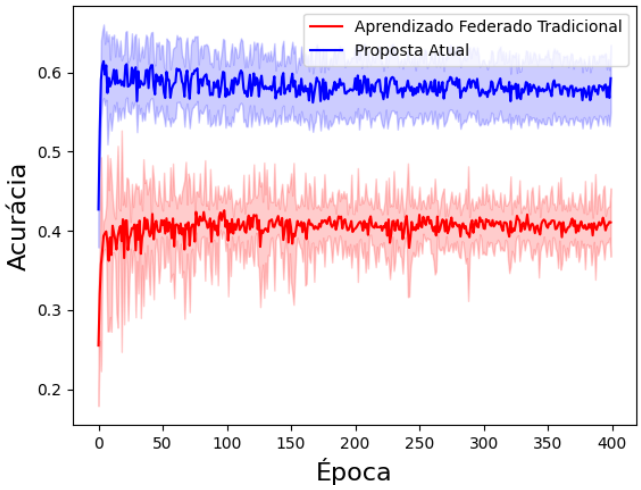

Figura 4: Evolução da perda de teste em função da época global para clientes com conjuntos de dados que contém apenas duas classes. A linha vermelha representa a abordagem tradicional de aprendizado federado com um único grupo, enquanto a linha azul representa a proposta deste artigo, contendo 5 grupos.

O experimento também analisa o comportamento individual dos modelos dos grupos. Os resultados mostram que o primeiro grupo possui uma acurácia de $(56 \pm 3)\%$, enquanto o segundo apresenta uma acurácia de $(63 \pm 3)\%$. Apesar de existir um grupo com maior desempenho, mesmo o baixo desempenho do primeiro grupo é maior que o obtido ao utilizar a proposta tradicional. Assim, pode-se concluir que a proposta mitigou os efeitos da heterogeneidade com sucesso no cenário com cinco classes por cliente.

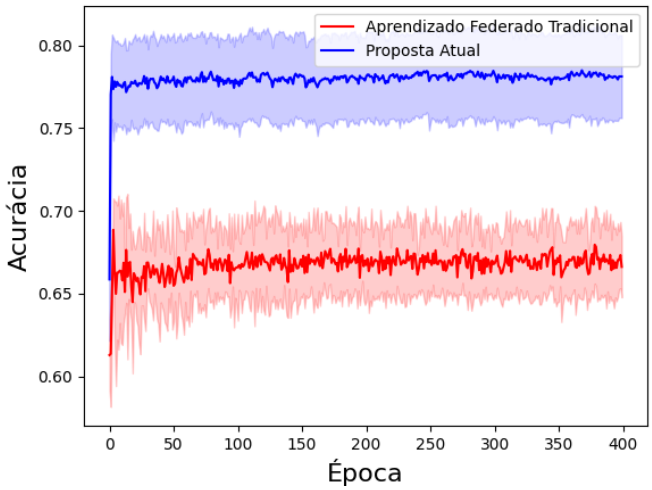

Figura 5: Evolução da perda de teste em função da época global para clientes com conjuntos de dados com 5 classes. A linha vermelha representa a abordagem tradicional com apenas um grupo, enquanto a linha azul representa a proposta deste artigo com 2 grupos de clientes.

O Experimento IV avalia novamente o desempenho da proposta para conjuntos de dados não-IID. A diferença em relação ao Experimento III é a redução do número de classes que cada

Tabela 4: Avaliação dos modelos dos grupos no cenário que os clientes possuem amostras de apenas 2 classes.

Identificador do Grupo	Quantidade de Clientes no Grupo	Acurácia no Teste (%)
1	2	81,05 $\pm$ 0,03
2	2	71,0 $\pm$ 0,4
3	2	75,3 $\pm$ 0,4
4	2	81,5 $\pm$ 0,1
5	2	80,4 $\pm$ 0,3

cliente possui para duas classes. Os resultados exibidos na Figura 5 mostram um comportamento similar ao Experimento III para os dois casos, porém com um desempenho final melhor. Esse fato pode ser explicado pela dificuldade do problema, simplificado a uma classificação binária. A Tabela 4 demonstra que apesar de alguns grupos possuírem um desempenho final um pouco menor, o resultado é maior que utilizar o aprendizado federado tradicional. Por fim, é possível concluir que para os grupos 1 e 4 a melhoria de desempenho em relação à proposta tradicional é próxima de 16%, indicando a relevância da proposta atual.

5. Conclusão

Este artigo propôs um sistema para aumentar o desempenho do aprendizado federado quando os clientes possuem distribuições de dados não-IID. A proposta mantém a privacidade dos dados dos clientes, pois utiliza como fonte de informação os valores de viés da camada final das redes neurais dos clientes para a identificação dos grupos existentes. Como as camadas finais da rede neural possuem alta relação com as classes existentes no conjunto de dados, é possível enviar apenas esta parte da rede neural para realizar o agrupamento dos clientes. A utilização do algoritmo de agrupamento DBSCAN, que exige apenas a distância mínima entre amostras como hiperparâmetro, forneceu alta capacidade de detectar os grupos existentes, mesmo no caso em que os clientes possuem conjuntos de dados IID. Por fim, o sistema proposto superou o tradicional de aprendizado federado quando os conjuntos de dados dos clientes são não-IID em todos os cenários avaliados. Os modelos gerados apresentaram alto desempenho de classificação e um curto tempo de convergência, de aproximadamente 10 épocas globais, enquanto o modelo tradicional obteve um baixo desempenho de classificação mesmo após 400 épocas globais para dados não-IID. A acurácia apresentou uma melhoria de aproximadamente 16%, no melhor caso. A proposta será ampliada para cenários não estacionários em trabalhos futuros, para lidar com a mudança de conceito (*concept drift*) dos dados, no qual os grupos podem ser alterados sob demanda. Outra direção de pesquisa é a implementação da combinação de modelos entre grupos diferentes. Além disso, será verificado o desempenho da proposta ao variar as características do ambiente federado, como o número de clientes participantes.

Referências

Ankerst, M., Breunig, M. M., Kriegel, H.-P. e Sander, J. (1999). OPTICS: Ordering Points to Identify the Clustering Structure. *ACM Sigmod record*, páginas 49–60.

Beutel, D. J. et al. (2020). Flower: A Friendly Federated Learning Research Framework. *arXiv preprint arXiv:2007.14390*.

Bochie, K. et al. (2021). Análise do Aprendizado Federado em Redes Móveis. Em *SBRC*, páginas 71–84.

de Souza, L. A. C. et al. (2020). DFedForest: Decentralized Federated Forest. Em *International Conference on Blockchain*, páginas 90–97. IEEE.

Dennis, D. K., Li, T. e Smith, V. (2021). Heterogeneity for the Win: One-Shot Federated Clustering. *arXiv preprint arXiv:2103.00697*.

Desai, H. B., Ozdayi, M. S. e Kantarcioglu, M. (2021). BlockFLA: Accountable Federated Learning via Hybrid Blockchain Architecture. Em *Proceedings of ACM Conference on Data and Application Security and Privacy*, páginas 101–112.

Douceur, J. R. (2002). The Sybil Attack. Em *International Workshop on Peer-to-Peer Systems*, páginas 251–260. Springer.

Ester, M., Kriegel, H.-P., Sander, J., Xu, X. et al. (1996). A Density-based Algorithm for Discovering Clusters in Large Spatial Databases with Noise. Em *KDD*, páginas 226–231.

Ghosh, A., Chung, J., Yin, D. e Ramchandran, K. (2020). An Efficient Framework for Clustered Federated Learning. *arXiv preprint arXiv:2006.04088*.

Krizhevsky, A. et al. (2009). Learning Multiple Layers of Features from Tiny Images. *Citeseer*.

Lai, F., Zhu, X., Madhyastha, H. V. e Chowdhury, M. (2021). Oort: Efficient Federated Learning via Guided Participant Selection. Em *OSDI*, páginas 19–35.

Liu, L., Zhang, J., Song, S. e Letaief, K. B. (2020). Client-Edge-Cloud Hierarchical Federated Learning. Em *International Conference on Communications*, páginas 1–6.

Luo, B. et al. (2021). Tackling System and Statistical Heterogeneity for Federated Learning with Adaptive Client Sampling. *arXiv preprint arXiv:2112.11256*.

MacQueen, J. et al. (1967). Some Methods for Classification and Analysis of Multivariate Observations. Em *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, páginas 281–297. Oakland, CA, USA.

McMahan, B. et al. (2017). Communication-efficient Learning of Deep Networks from Decentralized Data. *Artificial Intelligence and Statistics*, páginas 1273–1282.

Neto, H. N. et al. (2021). FedSA: Arrefecimento Simulado Federado para a Aceleração da Detecção de Intrusão em Ambientes Colaborativos. Em *SBRC*, páginas 280–293.

Nishio, T. e Yonetani, R. (2019). Client Selection for Federated Learning with Heterogeneous Resources in Mobile Edge. Em *International Conference on Communications*, páginas 1–7.

Ouyang, X. et al. (2021). ClusterFL: a Similarity-Aware Federated Learning System for Human Activity Recognition. Em *Proceedings of the 19th Annual International Conference on Mobile Systems, Applications, and Services*, páginas 54–66.

Pedregosa, F. et al. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Sattler, F., Müller, K.-R. e Samek, W. (2020). Clustered Federated Learning: Model-Agnostic Distributed Multitask Optimization under Privacy Constraints. *Transactions on Neural Networks and Learning Systems*.

Sun, Z., Kairouz, P., Suresh, A. T. e McMahan, H. B. (2019). Can You Really Backdoor Federated Learning? *arXiv preprint arXiv:1911.07963*.

Wang, H., Kaplan, Z., Niu, D. e Li, B. (2020). Optimizing Federated Learning on Non-IID Data with Reinforcement Learning. Em *INFOCOM*, páginas 1698–1707.

Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D. et al. (2018). Federated Learning with Non-IID Data. *arXiv preprint arXiv:1806.00582*.