Soluções para Dados Heterogêneos em Aprendizado Federado através de Similaridade de Modelos e Agrupamento de Clientes

Gabriel U. Talasso[1], **Leandro Villas**[1]

[1]Universidade Estadual de Campinas (UNICAMP), Brasil

g235078@dac.unicamp.br, lvillas@unicamp.br

Resumo. *O aumento dos dispositivos móveis e as crescentes preocupações com a privacidade têm colocado desafios significativos na inteligência artificial distribuída. Nesse cenário, surge o Federated Learning (FL) como um método promissor em que os modelos de aprendizagem são treinados de forma colaborativa e privada. No entanto, o FL também enfrenta desafios na convergência de modelos, otimização e sobrecarga de comunicação devido a heterogeneidade nos dados e dispositivos. Nesse contexto, este trabalho relata duas soluções desenvolvidas para endereçar esse problema: 1) NeuralMatch, uma ferramenta capaz de identificar similaridades entre os clientes apenas usando os modelos e 2) FedSCCS um solução completa que utiliza dos princípios anteriores para criar múltiplos modelos por agrupamento de clientes. Ambas soluções se mostram eficientes e eficazes conforme os amplos experimentos realizados.*

Abstract. *The rise of mobile devices and growing privacy concerns have posed significant challenges in distributed artificial intelligence. In this scenario, Federated Learning (FL) emerges as a promising method in which learning models are trained collaboratively and privately. However, FL also faces challenges in model convergence, optimization, and communication overhead due to data and devices heterogeneity. In this context, this work reports two solutions developed to address this problem: 1) NeuralMatch, a tool capable of identifying similarities between clients just using models and 2) FedSCCS a complete solution that uses the previous principles to create multiple models by client clustering. Both solutions have proven to be efficient and effective according to extensive experiments.*

1. Introdução

O surgimento da Inteligência Artificial (IA) acelerou a criação de soluções inteligentes que têm revolucionado a vida de milhões de pessoas em todo o mundo, influenciando suas formas de viver, trabalhar e até mesmo de atividades de lazer [Lim et al. 2020]. Essas soluções são construídas com base na coleta de dados de dispositivos como *smartphones*, dispositivos de Internet das Coisas (no inglês, IoT) e veículos, através de seus sensores físicos e virtuais. Os dados coletados são muitas vezes agregados e processados por um servidor central, onde são utilizados algoritmos de Aprendizado de Máquina (no inglês, ML) e IA, gerando conhecimento e visualizações para criar soluções personalizadas e inteligentes em vários domínios.

Nesse contexto, a coleta e compartilhamento de dados dos usuários gera preocupações em relação à segurança e à privacidade no uso desses, uma vez que esses dados podem conter informações sensíveis as quais indivíduos mal-intencionados e entidades maliciosas ou não possam se aproveitar [Liu et al. 2021]. É nesse cenário que o *Federated Learning* (FL) [Imteaj et al. 2022] surge como um método de aprendizagem distribuída que permite o treino colaborativo em vários dispositivos com um modelo compartilhado, garantindo que os

dados permanecem no dispositivo do usuário e não sejam transferidos para qualquer outra entidade [Abdulrahman et al. 2021].

No entanto, o FL enfrenta alguns desafios relacionados às heterogeneidades de dados entre os nós da rede. Essa heterogeneidade ocorre das diferentes distribuições de dados em cada dispositivo [Dennis et al. 2021], oque pode induzir problemas de não representatividade, diminuindo a acurácia e a justiça do modelo entre os participantes, uma vez que os modelos treinados podem divergir muito e gerar problemas de convergência nos cenários tradicionais.

A fim de contornar os desafios mencionados, este artigo apresenta as principais ideias de dois trabalhos desenvolvidos na direção de lidar melhor com dados heterogêneos, também comumente chamados de dados Não Independentes e Identicamente Distribuídos (não-IID). O primeiro trabalho é NeuralMatch [Talasso et al. 2023], publicado nos "Anais Estendidos do XLI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos"em 2023, onde é explorado uma métrica para identificar clientes similares em distribuição apenas utilizando seus modelos e não os dados privados. Já o segundo trabalho se refere ao FedSCCS [Talasso et al. 2024], aceito para publicação no "International Conference on Communications (ICC) 2024", conferência com Qualis A1, que apresenta o uso das ideias anteriores de maneira mais completa no agrupamento dos clientes similares para a criação de modelos especializados em vez de um único modelo global em aprendizado federado. Além disso, é importante ressaltar que o primeiro autor dos artigos é o mesmo deste e sua contribuição foi presente em todas as etapas dos trabalhos anteriores. Dessa forma, as demais seções desse trabalho detalham as principais ideias apresentadas por esses dois artigos.

2. Trabalhos Relacionados

Esta seção discute trabalhos de FL com um enfoque específico na questão da heterogeneidade dos dados e dos dispositivos. A seleção de clientes é uma das soluções para reduzir os custos de comunicação, aumentar a acurácia e justiça [Nishio and Yonetani 2019]. Foram investigadas várias técnicas, desde a seleção aleatória de clientes, considerando a disponibilidade [Ribero et al. 2023], e avaliando a qualidade dos dados locais [Deng et al. 2022]. Cho *et al.* [Cho et al. 2022] introduziu a metodologia *Power of Choice* (POC), que treina estrategicamente modelos com os clientes com pior desempenho em rodadas subsequentes, melhorando assim a convergência global do modelo.

Outra abordagem investigada seria a formação de agrupamentos (*clusters*) de clientes, como o Clustered Federated Learning (CFL) que agrupa clientes com modelos semelhantes usando a similaridade de cosseno nos gradientes, como método de pós processamento para lidar com clientes com diferentes distribuições, enquanto Sener & Savarese propõem o uso do algoritmo KCenter nos pesos dos modelos para criar *clusters* e reduzir o custo de comunicação [Sattler et al. 2021, Wang et al. 2020]. FedSim, por sua vez, utiliza representações reduzidas de gradientes e algoritmos de *clustering* como *K-Means* para formar *clusters* representativos, selecionando aleatoriamente clientes para cada grupo [Palihawadana et al. 2022]. O FLIS emprega um conjunto de dados global para calcular a similaridade entre modelos e usa um algoritmo de agrupamento hierárquico para separar clientes em grupos que refletem diferenças locais na distribuição dos dados [Morafah et al. 2023].

Como mencionado anteriormente, as soluções incorporam dois princípios fundamentais: *(i)* agrupamento de clientes com base na similaridade dos seus modelos, e *(ii)* diferentes formas de agregar os modelos dos clientes, e selecioná-los de maneira eficaz para o treinamento. Dessa forma, os trabalhos desenvolvidos tanto desenvolvem uma métrica ainda não

utilizada nesse contexto para similaridade de modelos, como também mostra um novo tratamento com múltiplos modelos especializados para diferentes distribuições, sendo essas as principais contribuições dos trabalhos

3. Identificando a similaridade com NeuralMatch

Essa seção descreve o funcionamento do NeuralMatch, como essa ferramenta pode ser utilizada para identificar a similaridade dos clientes utilizando apenas o modelo deles, e mantendo a privacidade como nas versões padrões do aprendizado Federado. Além disso, será apresentado um referência teórico e matemático de embasamento dessa técnica, o que consolida seu funcionamento.

3.1. Referencial Teórico: FedAvg

O *Federated Averaging* (FedAvg) [McMahan et al. 2017] é um método padrão de aprendizado federado que agrega modelos de clientes realizando uma média ponderada de todos os pesos dos modelos locais. O FL é composto por 3 principais passos: (i) o treinamento local dos modelos com os dados privados, (ii) o envio e agregação desses modelos no servidor central e (iii) o retorno desse modelo agregado para os clientes, porém agora sendo um modelo mais acurado. Dessa forma, seja $|C|$ o número de clientes num cenário em que C representa o conjunto de clientes e T o número de rodadas de comunicação. Em cada rodada $t \in 1, 2, 3, \ldots, T$, um subconjunto aleatório $\mathcal{S} \subseteq C$ de clientes participa no treino do modelo. Cada cliente $i \in \mathcal{S}$ tem o seu conjunto de dados local D_i, $n_i = |D_i|$, e $N = |\bigcup_{i \in C} D_i|$. Assim, cada cliente recebe os parâmetros atuais do modelo (os pesos) $w(t)$ do servidor e atualiza o seu modelo local para treinar nos seus dados. Em seguida, cada cliente i envia os parâmetros do modelo recém-treinado de volta para o servidor após treinar na rodada $t + 1$: $w^i(t + 1)$. O servidor agrega então os modelos recebidos efetuando uma média ponderada, de acordo com a Equação 1.

$$w(t+1) = \sum_{i \in C} \frac{n_i}{N} w^i(t+1). \tag{1}$$

Assim, o FedAvg minimiza a função f na Equação 2, em que w é o modelo global, $\mathcal{L}$ representa a função de perda do modelo e x_i e y_i são os conjuntos de dados de entrada e de respostas esperadas, respectivamente.

$$\min f(w) \text{ onde } f(w) = \sum_{i=1}^{|C|} \frac{n_i}{N} \mathcal{L}(x_i, y_i; w). \tag{2}$$

Dessa forma, espera-se e cada rodade de comunicação, a média dos modelos seja melhor para todos eles, e o modelo global fique cada vez mais eficiente uma que vez que aprende de foma colaborativa entre todas as partes da rede. Porém, como menciado anteriormente essa solução sofre quandos ods dados de treinamento locais são muito distintintos que causam problemas de convergência do modelo global para todos os clientes.

3.2. Descrição da Ferramenta

No NeuralMatch, assim como no FedAvg, também há os 3 principais passos de aprendizado federado, a diferença é que o NeuralMatch adiciona um passo nesse processo: o calculo da similaridade entre clientes, para isso é utilizada a técnica *Centered Kernel Alignment (CKA)* [Kornblith et al. 2019]. Dessa forma, com a identificação das similaridades dos clientes, é possível pensar em diversas soluções para otimização de problemas do aprendizado federado, como já mencionado anteriormente, porém, nesse trabalho focaremos na

utilização dessas similaridades para agrupamento de clientes, o que será apresentado nas próximas seções.

3.3. Similaridade entre Modelos

O NeuralMatch (assim como o FedSCCS) calcula a similaridade entre modelos através do CKA, o qual utiliza transformações de *kernel* em matrizes para medir uma representação da correlação dessas matrizes, vista nesse caso como a similaridade. Desta forma, o NeuralMatch calcula a_l^i, a ativação da camada l do cliente i, dada por: $a_l^i = \sigma_l^i(w_l^i a_{l-1}^i + b_l^i)$, onde σ_l^i, w_l^i, b_l^i são a função de ativação, os pesos e o bias da l-ésima camada para o cliente i, respetivamente. Assim, para a primeira camada, temos $a_1^i = \sigma_1^i(w_1^i D + b_1^i)$, onde D é uma amostra de dados no servidor, que pode ser muito menor que os conjuntos de dados locais dos clientes, usada apenas para efeito de cálculo. Assim, a_l^i tem duas dimensões, uma com o comprimento do conjunto de dados do servidor e a outra com o comprimento da camada l.

Portanto, as matrizes de *kernel* K_1 e K_2 são calculadas passando a_l^i e a_l^j para as funções de *kernel* $K_1 = k_1(a_l^i, a_l^i)$ e $K_2 = k_2(a_l^j, a_l^j)$, que podem ser qualquer operação em matrizes. Neste trabalho, k_1 e k_2 foram utilizados como *kernels* lineares, sendo que $K_1 = k_1(\mathbf{x}, \mathbf{x}) = \mathbf{x}^T\mathbf{x}$ e $K_2 = k_2(\mathbf{y}, \mathbf{y}) = \mathbf{y}^T\mathbf{y}$. O CKA é calculado conforme equações a seguir:

$$\mathrm{CKA}(K_1, K_2) = \frac{\mathrm{HSIC}(K_1, K_2)}{\sqrt{\mathrm{HSIC}(K_1, K_1)\,\mathrm{HSIC}(K_2, K_2)}}, \tag{3}$$

$$\mathrm{HSIC}(K_1, K_2) = \frac{1}{(n-1)^2}\,\mathrm{tr}(K_1 H K_2 H), \tag{4}$$

onde HSIC é o Critério de Independência de Hilbert-Schmidt, tr é o traço da matriz e H é uma matriz centralizada $H_n = I_n - \frac{1}{n}\underset{\sim}{\mathbf{1}} \times \underset{\sim}{\mathbf{1}}^{\mathrm{T}}$, com $\underset{\sim}{\mathbf{1}}^{\mathrm{T}} = \begin{pmatrix} 1 & 1 & \dots & 1 \end{pmatrix}$, um vetor de números 1, com comprimento n e I_n a identidade com ordem n onde $n = |D|$, o tamanho da amostra de dados do servidor.

Note que o conjunto de dados utilizado no servidor pode advir de clientes que optarem por compartilhar uma pequena parte dos dados, ou ainda vir de conjuntos públicos que existem para a maioria das aplicações. Investigações mais recentes mostram que pode-se usar ruído sintético para esse cálculo, dessa forma, problemas de privacidade não são adicionados às soluções.

4. FedSCCS

O FedSCCS se apresenta como uma evolução do NeuralMatch, porém é definido como solução de agrupamento de clientes para reduzir os problemas gerados por dados heterogêneos. É utilizado também o CKA [Kornblith et al. 2019], demostrado anteriormente, mas com a adição disso do algoritmo de agrupamento (também chamado de *clustering*) hierárquico que divide os clientes em grupos (chamados de *clusters*), o que permite ao FedSCCS criar um modelo para cada *cluster*, tornando assim o treinamento mais homogêneo dentro desses e facilitando o aprendizado e convergência dos modelos federados.

Adicionalmente, o FedSCCS também foi testado com algumas técnicas de seleção de clientes, para mostrar sua robustez à esses casos além de mostrar que pode ser utilizada de forma a ”consciente de recursos”a fim de melhorar a eficiência de processamento e comunicação da rede.

4.1. Fluxo de funcionamento

A solução clássica FedAvg (ou seja, um modelo global) permite o aprendizado colaborativo, mas enfrenta dificuldades em cenários heterogêneos, uma vez que o modelo global sofre de problemas de convergência no treinamento entre muitas distribuições distintas. Para superar esta limitação, o FedSCCS propõe um gerenciamento de grupos de maneira hierárquica baseada em *clusters* que trata cada *cluster* de forma independente, permitindo o treinamento separado e permitindo diferentes níveis de generalização por ter uma estrutura hierárquica.

O FedSCCS organiza os *clusters* de acordo com a similaridade dos modelos. Utiliza uma função de *clustering* $\mathcal{K}$ que recebe o conjunto de modelos $\mathcal{W} = \{w^1, w^2, w^3, \ldots, w^{|C|}\}$ dos clientes e devolve um conjunto de *clusters* $\Phi = \{C_1, C_2, \ldots C_K\}$, de modo que $\Phi = \mathcal{K}(\mathcal{W})$ e $\Phi \subset C$, em que K é o número de *clusters*.

Assim, cada *cluster* pode ser gerido de forma independente, ou seja, a agregação dos modelos é específica e especializada para cada *cluster* (chamamos estes modelos de "modelos especializados"). Os pesos do modelo especializado do *cluster* k, representados por w_k, são calculados conforme Equação 5, que vêm de uma modificação da Equação 1.

$$w_k(t+1) = \sum_{i \in C_k} \frac{n_i}{N_k} w^i(t), \tag{5}$$

onde C_k representa o conjunto de clientes endereçados ao *cluster* k (um cliente só pode estar em um *cluster*), e N_k é o número total de amostras em k. O mesmo acontece com a função de perda, que é específica para cada *cluster*.

Algoritmo 1: Pseudocódigo do FedSCCS

```
Servidor Central:
    w(0) ← iniciarPesos( )                                  ▷ Inicializa o modelo
    para cada rodada de comunicação faça
        /* Em paralelo */
        para cliente i ∈ C faça
            W ← atualizaCliente(w)
        para i ∈ S faça
            w(t+1) ← Σ (n_i/N) w^i(t)
    Φ ← K(W)                                                ▷ Clusteriza os clientes
    para cada rodada de comunicação p, p+1, ..., T faça
        /* Em paralelo */
        para cliente i ∈ C faça
            w_k ← pegarModeloDoCluster(i)
            W ← atualizaCliente(w_k)
        para C_k ∈ Φ | k = {1, 2, ..., K} faça
            c_k ← selecionaCliente(C_k)
            w_k(t+1) ← Σ_{i∈c_k} (n_i/N_k) w^i(t)

/* Uma vez recebido o modelo atualizado */
Cliente atualizaCliente(w):
    B ← trainLoader(D_i)                                    ▷ Divide os dados em batches
    para cada época faça
        para batch (x_i, y_i) ∈ B faça
            w^i ← L(x_i, y_i, w)                            ▷ Treina o modelo com os dados locais
    retorna w^i para o servidor
```

O FedSCCS é dividido em rodadas do FedAvg tradicional e outras com *clusters* especializados. A descrição completa de FedSCCS está descrita no Algoritmo 1. Em primeiro lugar, o FedSCCS inicializa o modelo global e o envia para os clientes, cada um dos quais

executa uma época local no seu conjunto de dados privado (Linha 2 e Linhas 15-21). O segundo passo é executar a abordagem padrão FedAvg para rodadas de $p-1$ (Linhas 3-7). O objetivo é aprender os modelos dos clientes durante algumas épocas antes de agrupá-los. Na rodada p, o algoritmo de agrupamento cria os grupos de clientes (Linha 8) e o servidor envia modelos especializados para eles. Na rodada p até ao fim, o servidor seleciona um subconjunto de clientes c_k do *cluster* C_k para cada *cluster* k (Linha 14) e inicia a agregação em cada um (Linha 15). Assim, para todas as outras rodadas, FedSCCS repete as Linhas 9-15 até à convergência dos modelos dos *clusters*.

4.2. Similaridade do modelo

Assim como explicado anteriormente o FedSCCS utiliza a mesma similaridade descrita no NeuralMatch, mas nesse caso, toma-se a notação de similaridade dos clientes i e j, sendo $\mathrm{CKA}^{l}_{j,i}(K_1, K_2) = \mathrm{CKA}^{l}_{i,j}(K_1, K_2)$, onde $K_1 = k_1(a^i_l, a^i_l)$ e $K_2 = k_2(a^j_l, a^j_l)$ considerando a camada l. Com esta métrica, pode-se construir a matriz S, denominada matriz de similaridade, com dimensões $|C| \times |C|$, introduzindo em cada célula $s_{i,j}$ o $\mathrm{CKA}^{l}_{i,j}$. Essa matriz é importante pois ela que é utilizada nos passos seguintes para criação dos clusters.

4.3. Agrupamento hierárquico baseado em CKA

Para o agrupamento hierárquico, considerou-se o método de Ward [Ward Jr 1963] que é um método baseado na distância entre os pontos de dados. No FedSCCS, as distâncias são calculadas com a matriz de similaridade S gerada pelo método CKA e utilizam a distância euclidiana aos pares nas colunas $s_{.,i}$ e $s_{.,j}$ da matriz S, conforme uma distância euclideana. Como resultado, o agrupamento hierárquico é aplicado à matriz de distância D composta por elementos $d_{i,j}$, representando a distância entre clientes.

Esta abordagem proporciona flexibilidade, uma vez que os clusters são hierárquico pode-se ter uma divisão mais granular resultando em múltiplos modelos mais especializados, e no limite um para cada usuário, ou ainda ter soluções mais gerais, diminuindo o número de clusters e no limite sendo igual ao FedAvg. O ideal é que se use valores intermediários a depender da aplicação, capturando assim agrupamentos de distribuições que colaboram ao treinarem juntas.

4.4. Estratégias de seleção de clientes

O FedSCCS também foi testado com diversas estratégias de seleção de clientes, tanto para mostrar sua adaptabilidade a esses casos tão importantes da literatura, como também para permitir mais eficiência da comunicação, distribuição de demanda computacional, convergência dos modelos e justiça entre os clientes.

Para isso, consideramos quatro métodos para mostrar a flexibilidade do FedSCCS na utilização de diferentes estratégias de seleção de clientes com base em três abordagens: *(i)* seleção de todos os clientes, *(ii)* seleção aleatória de cliente, *(iii)* seleção enviesada por desempenho e *(iv)* seleção enviesada por recursos. Essa última não foi avaliada no artigo aceito no ICC 2024([Talasso et al. 2024]) porém se mostrou eficaz em testes posteriores.

Na primeira estratégia, todos os clientes de cada *cluster* serão selecionados para treinar o modelo nessa rodada. A segunda estratégia seleciona aleatoriamente um cliente de cada *cluster* para treinar o modelo, mantendo a especialização de cada *cluster* e diminuindo os custos de comunicação ao reduzir o número de clientes selecionados. A terceira estratégia seleciona os clientes com base no pior desempenho na rodada. A ideia é melhorar o desempenho aumentando a acurácia e reduzindo a perda de clientes que anteriormente tiveram um

mau desempenho [Cho et al. 2022]. Por fim, a quarta estratégia tem o foco na seleção do cliente que foi menos selecionado nas rodadas anteriores, mantendo certa justiça no treinamento e auxiliando a não esgotar recursos de clientes com piores acurácias.

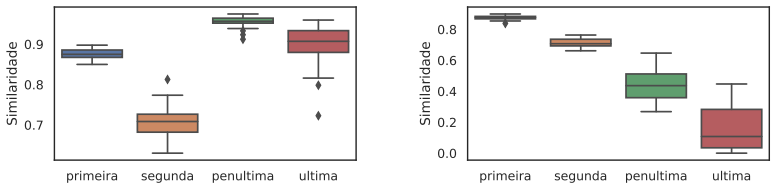
5. Análise de Desempenho

Esta seção apresenta os principais resultados obtidos no teste dos dois métodos, bem como evidencia seu funcionamento e vantagens, inclusive com comparações com a literatura.

5.1. Avaliação do NeuralMatch

Para avaliar o FedSCCS, além da convergência do modelo global do treinamento federado, ainda foram avaliadas as similaridades obtidas pelo servidor e calculadas sobre os modelos globais. Dessa forma, foram desenvolvidas simulações utilizando o *framework* de aprendizado federado Flower [Beutel et al. 2020], e o famoso conjunto de dados MNIST com o objetivo principal de treinar um modelo federado para a classificação de imagens. O modelo foi desenvolvido utilizando TensorFlow versão 2.11 é um *Multi Layer Perceptron - MLP*, com 4 camadas, para simplificar as análises uma vez que o conjunto de dados não representa grande complexidade.

5.2. Análise da Similaridade das Camadas do Modelo com Dados IID e não-IID



(a) Todas classes no cliente. (b) Duas classes no cliente.

Figura 1. Comparação entre as similaridades das redes de clientes.

Os experimentos o comportamento da similaridade entre diversas camadas da rede bem como esse similaridade é afetada no caso de conjunto de dados não-IID. Para isso, foram executas 50 simulações para cada caso descrito abaixo no qual a similaridade é calculada entre dois modelos treinados com conjuntos de dados diferentes.

Assim, a Figura 1(a) apresenta a similaridade da métrica CKA com clientes com dados IID, ou seja, nesse caso os clientes receberam quantidades parecidas de cada classe para serem homogêneos em distribuição. Nesse caso, observa-se que as similaridades são altas para todas as camadas. Por outro lado, ao considerar dados não-IID na Figura 1(b), onde os clientes recebem apenas 2 classes diferentes, verificamos uma queda na similaridade das últimas camadas. As primeiras camadas mantiveram-se em níveis proporcionais aos dados IID. Assim, é possível não só observar a eficácia da métrica como também em quais camadas ela é mais eficiente para distinguir os clientes.

5.3. Ambiente para avaliação do FedSCCS

O FedSCCS também foi implementado utilizando o *framework* Flower [Beutel et al. 2020] e o TensorFlow nas versões 1.4 e 2.12, respectivamente. As simulações consideraram a aplicação de Reconhecimento de Atividades Humanas(HAR) [Sozinov et al. 2018], por conta da relevância em cenários distribuídos O primeiro contém dados HAR de uma série de experiências controladas com 24 participantes que variam em gênero, idade, altura e peso. Esse

conjunto de dados foi coletado através de um acelerômetro e de um giroscópio (ou seja, altitude, gravidade, aceleração e taxa de rotação). A conjunto tem como foco a classificação em seis principais atividades: descer e subir escadas, andar, correr, sentar e ficar de pé.

O modelo utilizado também foi uma MLP que variava apenas no numero de neurônios por camada. Além disso uma amostra de dados no servidor contém 100 registros aleatórios do conjunto de dados local de cada cliente. É um valor mínimo em comparação com o tamanho de todo o conjunto de dados, que tem aproximadamente 40000 de amostras cada. A similaridade é calculada com a última camada, pois como visto no NeuralMatch essa é a mais significativa para diferenciá-los. Além disso, foram utilizado 10 clusters por se mostrar o valor mais significativo para esse problema específico.

5.4. Comparação com soluções da literatura

Para representar as soluções que utilizam os pesos dos modelos (ou gradientes) e similaridade de cossenos [Sattler et al. 2021, Wang et al. 2020], foi criada a *Weights Clustering* (WC) que também utiliza o método hierárquico de Ward. Adicionalmente, também utilizando os pesos, foi implementada a solução K-Center [Wang et al. 2020]. Além disso, o FedSCCS o *Random Clusters* (RC) representa a atribuição aleatória dos clientes nos clusters, servindo como uma base de desempenho dessa divisão sem critério algum.

A Figura 2 apresenta uma comparação do FedSCCS com tais abordagens em ambos *datasets*. O melhor desempenho do FedSCCS neste experimento resulta do agrupamento baseado na similaridade dos modelos que melhor representam e captam a distribuição de dados dos clientes. Todas as soluções de *clusters* atingem um desempenho superior à abordagem clássica FedAvg e ao RC, o que significa que são métodos válidos num cenário heterogêneo.

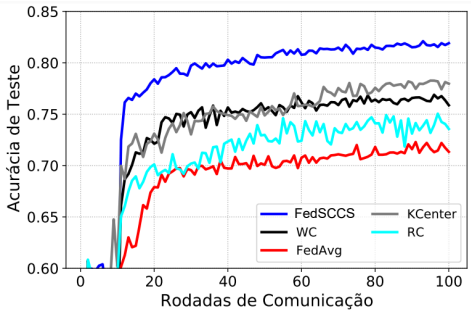


Figura 2. Acurácia de teste de diferentes soluções da literatura

Além disso, a Figura 5.4 apresenta uma comparação do uso do FedSCCS em conjunto com diversas soluções de seleção de clientes. Uma delas baseada apenas na seleção aleatória, como é feito no FedAvg, outra se baseia na seleção de clientes com menor acurácia, promovendo justiça e aumentando o desempenho geral da rede, essa solução é baseada no já mencionado POC, com a modificação de ser aplicado por clusters segundo a metodologia apresentada. Por fim a última seleção mostrada representa uma tentativa de garantir justiça de recursos na rede pois seleciona o cliente menos selecionado anteriormente de cada cluster evitando drenar os recursos de um subgrupo de clientes.

A maior acurácia do FedSCCS ocorre porque os *clusters* representam clientes mais semelhantes, fazendo com que o processo de FL aprenda com menos clientes. Isto significa que o agrupamento empregado pelo FedSCCS ajuda a selecionar clientes representativos das suas distribuições de dados, melhorando a eficiência da comunicação mesmo quando se selecionam menos clientes.

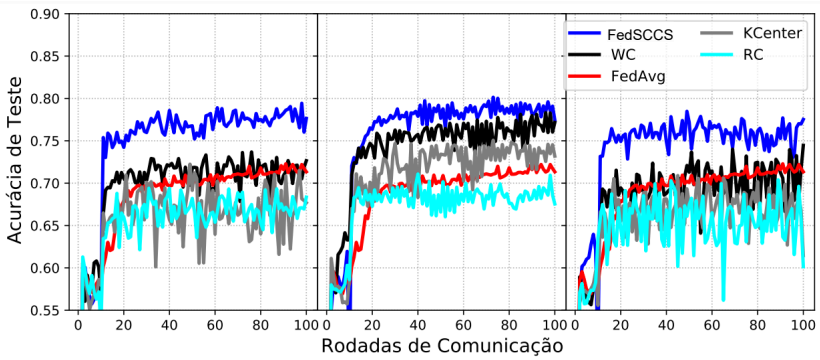


Figura 3. A imagem da esquerda mostra o desempenho da seleção aleatória em diferentes soluções; a imagem do meio mostra a seleção baseada no POC; e a da direita representa a seleção pelo cliente menos selecionado.

5.5. Análise de comunicação

Esta análise avaliou o custo de comunicação (*uplink* entre cliente e servidor) e a melhora com técnicas de seleção de clientes. Os experimentos compararam o FedSCCS com o FedAvg com todos os clientes em cada rodada. Com isso, observou-se que, para 10 *clusters*, a solução baseada no POC consumiu apenas 62,5% do custo de comunicação da abordagem completa do FedAvg, enquanto a seleção aleatória de clientes e a seleção pelo menos selecionado consumiu apenas 47,5%. Portanto, conclui-se que as técnicas de seleção de clientes reduzem os custos, especialmente na seleção aleatória.

6. Considerações Finais

Este trabalho apresenta os principais pontos de duas soluções desenvolvidas durante o período de iniciação científica. O primeiro deles, o NeuralMatch, se mostrou uma forma eficiente de identificar similaridades de clientes em aprendizado federado, ou seja, em um contexto em que não se possui acesso ao dados dos clientes, apenas aos modelos. Como ampliação e aprofundamento desse método, foi criado o FedSCCS , uma solução de agrupamento de clientes com a formação de múltiplos modelos a qual se mostrou mais eficiente que algumas soluções da literatura bem como capaz de ser utilizada par melhorar os custos de comunicação e treinamento desses modelos na rede.

7. Agradecimentos

Este projeto foi apoiado pelo Ministério da Ciência, Tecnologia e Inovação, com recursos da Lei nº 8.248, de 23 de outubro de 1991, no âmbito do PPI-Softex, coordenado pela Softex e publicado como Agentes inteligentes para plataformas móveis baseados em tecnologia de Arquitetura Cognitiva (processo 01245.013778/2020-21).

Referências

Abdulrahman, S., Tout, H., Ould-Slimane, H., Mourad, A., Talhi, C., and Guizani, M. (2021). A survey on federated learning: The journey from centralized to distributed on-site learning and beyond. *IEEE Internet of Things Journal*, 8(7):5476–5497.

Beutel, D. J., Topal, T., Mathur, A., Qiu, X., Fernandez-Marques, J., Gao, Y., Sani, L., Li, K. H., Parcollet, T., de Gusmão, P. P. B., et al. (2020). Flower: A friendly federated learning research framework. *arXiv preprint arXiv:2007.14390*.

Cho, Y. J., Wang, J., and Joshi, G. (2022). Towards understanding biased client selection in federated learning. In *International Conference on Artificial Intelligence and Statistics*, pages 10351–10375. PMLR.

Deng, Y., Lyu, F., Ren, J., Wu, H., Zhou, Y., Zhang, Y., and Shen, X. (2022). Auction: Automated and quality-aware client selection framework for efficient federated learning. *IEEE Transactions on Parallel and Distributed Systems*, 33(8):1996–2009.

Dennis, D. K., Li, T., and Smith, V. (2021). Heterogeneity for the win: One-shot federated clustering. In Meila, M. and Zhang, T., editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 2611–2620. PMLR.

Imteaj, A., Thakker, U., Wang, S., Li, J., and Amini, M. H. (2022). A survey on federated learning for resource-constrained iot devices. *IEEE Internet of Things Journal*, 9(1):1–24.

Kornblith, S., Norouzi, M., Lee, H., and Hinton, G. (2019). Similarity of neural network representations revisited. In *International conference on machine learning*, pages 3519–3529. PMLR.

Lim, W. Y. B., Luong, N. C., Hoang, D. T., Jiao, Y., Liang, Y.-C., Yang, Q., Niyato, D., and Miao, C. (2020). Federated learning in mobile edge networks: A comprehensive survey. *IEEE Communications Surveys Tutorials*, 22(3):2031–2063.

Liu, B., Ding, M., Shaham, S., Rahayu, W., Farokhi, F., and Lin, Z. (2021). When machine learning meets privacy: A survey and outlook. *ACM Computing Surveys (CSUR)*, 54(2):1–36.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pages 1273–1282. PMLR.

Morafah, M., Vahidian, S., Wang, W., and Lin, B. (2023). Flis: Clustered federated learning via inference similarity for non-iid data distribution. *IEEE Open Journal of the Computer Society*, 4:109–120.

Nishio, T. and Yonetani, R. (2019). Client selection for federated learning with heterogeneous resources in mobile edge. In *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, pages 1–7.

Palihawadana, C., Wiratunga, N., Wijekoon, A., and Kalutarage, H. (2022). Fedsim: Similarity guided model aggregation for federated learning. *Neurocomputing*, 483:432–445.

Ribero, M., Vikalo, H., and de Veciana, G. (2023). Federated learning under intermittent client availability and time-varying communication constraints. *IEEE Journal of Selected Topics in Signal Processing*, 17(1):98–111.

Sattler, F., Müller, K.-R., and Samek, W. (2021). Clustered federated learning: Model-agnostic distributed multitask optimization under privacy constraints. *IEEE Transactions on Neural Networks and Learning Systems*, 32(8):3710–3722.

Sozinov, K., Vlassov, V., and Girdzijauskas, S. (2018). Human activity recognition using federated learning. In *2018 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Ubiquitous Computing & Communications, Big Data & Cloud Computing, Social Computing & Networking, Sustainable Computing & Communications (ISPA/IUCC/BDCloud/SocialCom/SustainCom)*, pages 1103–1111.

Talasso, G., de Souza, A. M., Bittencourt, L. F., Cerqueira, E., Loureiro, A., and Villas, L. (2024). FedSCCS: hierarchical clustering with multiple models for federated learning. In *2024 IEEE International Conference on Communications (ICC): SAC Cloud Computing, Networking and Storage Track (IEEE icc'24 - SAC-02 CCNS track)*, page 5.99, Denver, USA.

Talasso, G., Souza, A., and Villas, L. (2023). Neuralmatch: Identificando a similaridade de clientes baseado em modelos no aprendizado federado. In *Anais Estendidos do XLI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, pages 176–183, Porto Alegre, RS, Brasil. SBC.

Wang, H., Kaplan, Z., Niu, D., and Li, B. (2020). Optimizing federated learning on non-iid data with reinforcement learning. In *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, pages 1698–1707.

Ward Jr, J. H. (1963). Hierarchical grouping to optimize an objective function. *Journal of the American statistical association*, 58(301):236–244.