FedSNIP: Método baseado em Poda de Modelo de Etapa Única para Comunicação Eficiente em Aprendizado Federado

Rómulo Bustincio[1], Allan M. de Souza[1], Joahannes B. D. da Costa[1], Luis F. G. Gonzalez[1], Luiz F. Bittencourt[1]

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

r204185@dac.unicamp.br, {allanms, jbdc, gonzalez, lfb00}@unicamp.br

Abstract. *In the realm of Federated Learning (FL), a collaborative yet decentralized approach to machine learning, communication efficiency is a critical concern, particularly under constraints of limited bandwidth and resources. This paper introduces an innovative application of the SNIP (Single-shot Network Pruning based on Connection Sensitivity) technique within this context. Leveraging SNIP, the proposed method prunes neural networks effectively, converting numerous weights to zero and resulting in sparser weight representations. This significant reduction in weight density substantially decreases the volume of parameters that need to be communicated to the server, thus reducing the communication overhead. Our experiments on the MNIST dataset showcase that this approach not only lowers the data transmission between clients and server but also sustains competitive model accuracy, comparable to conventional FL models. The use of network pruning via SNIP emerges as an efficacious strategy to augment the efficiency of FL, especially advantageous in settings with restricted communication capabilities.*

Resumo. *No âmbito do Aprendizado Federado (AF), uma abordagem colaborativa, porém descentralizada, para a aprendizagem de máquina, a eficiência da comunicação é uma preocupação crítica, especialmente sob as restrições de largura de banda e recursos limitados. Este artigo introduz uma aplicação inovadora da técnica SNIP (Single-shot Network Pruning based on Connection Sensitivity) neste contexto. Aproveitando o SNIP, o método proposto poda eficazmente as redes neurais, convertendo numerosos pesos em zero e resultando em representações de pesos mais esparsas. Essa redução significativa na densidade de pesos diminui substancialmente o volume de parâmetros que precisam ser comunicados ao servidor, reduzindo assim a sobrecarga de comunicação. Os experimentos com o conjunto de dados MNIST demonstram que esta abordagem não apenas reduz a transmissão de dados entre clientes e servidor, mas também mantém a acurácia competitiva do modelo, comparável aos modelos convencionais de AF. O uso da poda de rede via SNIP emerge como uma estratégia eficaz para aumentar a eficiência do AF, especialmente vantajosa em ambientes com capacidades de comunicação restritas.*

1. Introdução

O campo emergente do Aprendizado Federado (AF), do inglês *Federated Learning (FL)* e introduzido por McMahan *et al.* [McMahan et al. 2016], representa um avanço crítico no âmbito do aprendizado de máquina. Ele aborda questões relacionadas à privacidade

de dados e gestão descentralizada desses dados. Dentro do arcabouço do AF, vários clientes refinam coletivamente um modelo unificado enquanto preservam a integridade e a confidencialidade de seus dados individuais. Apesar de suas notáveis vantagens, o AF tem seus desafios, sendo o principal deles os custos de comunicação envolvidos no processo de aprendizagem colaborativa [Kairouz and McMahan 2021].

Considerando o cenário em que milhares de dispositivos (clientes) participam do processo de aprendizagem, onde cada um transmite parâmetros de modelo para um servidor central, os custos de comunicação crescem proporcionalmente [Xia et al. 2021]. A alta demanda por largura de banda de comunicação durante o processo de treinamento reduz significativamente a eficiência operacional do AF em cenários críticos. Abordar essa questão é essencial, como enfatizado por estudos recentes no campo [Wen et al. 2023, Shahid et al. 2021]. Melhorar a eficiência de comunicação não é apenas benéfico, mas necessário para o desenvolvimento contínuo e implantação prática das soluções de AF [Souza et al. 2023, de Souza et al. 2024, Wen et al. 2023].

Diversas técnicas foram propostas para contornar esse desafio de comunicação no AF, porém muitas dessas abordagens introduzem etapas extras que dificultam sua implantação. Por exemplo, certas abordagens necessitam de *fine-tuning* através de pós-poda para restaurar ou aumentar a acurácia do modelo. O *fine-tuning* é o processo de ajuste que envolve treinamento adicional para recuperar a acurácia perdida após um processo de poda em redes neurales [Renda et al. 2020]. Além disso, metodologias de treinamento dependentes da criação de máscaras, geralmente requerem início com redes densas, o que implica em uma configuração inicial mais intrincada e uma maior demanda por recursos [Isik et al. 2023a, Jiang et al. 2022a, Isik et al. 2023b, Vallapuram et al. 2022].

Levando isso em consideração, este artigo apresenta uma abordagem para redução de custos de comunicação no AF através da utilização do algoritmo SNIP. O SNIP é um algoritmo que se destaca por sua capacidade de podar redes neurais de forma eficiente em uma única fase antes do treinamento [Lee et al. 2019]. Assim, quando aplicado ao AF, o SNIP pode diminuir substancialmente a contagem de parâmetros necessária para sincronização, enquanto preserva ou potencialmente aumenta a acurácia do modelo. Dessa forma, com a aplicação desse algoritmo no contexto do AF, objetiva-se otimizar a comunicação e reduzir os custos a ela associados.

Complementarmente, este estudo foca na aplicação do SNIP ao conjunto de dados MNIST [LeCun et al. 1998], que é bastante conhecido e utilizado no campo de aprendizado de máquina para classificação de imagens. Os resultados alcançados demonstram não somente uma redução significativa na quantidade de parâmetros comunicados pelo cliente ao servidor, mas também evidenciam a capacidade de manter acurácia, ressaltando a eficácia desta abordagem em contextos práticos. Este trabalho ilustra o potencial de técnicas de poda de rede na melhoria da eficiência do AF e abre novas possibilidades para sua aplicação em ambientes com capacidades de comunicação limitadas.

O restante deste trabalho está organizado da seguinte forma. A Seção 2 apresenta os trabalhos relacionados no contexto de redução de custos de comunicação no AF. A Seção 3 apresenta alguns conceitos chave para entendimento deste trabalho. A Seção 4 detalha o funcionamento do FedSNIP, apresentando seus principais componentes e descrevendo as etapas para seu funcionamento. A Seção 5 apresenta a metodologia

de avaliação e comparação do FedSNIP com outros trabalhos do estado da arte, além de discutir os resultados obtidos. Por fim, a Seção 6 conclui este estudo e apresenta alguns direcionamentos para trabalhos futuros.

2. Trabalhos Relacionados

Esta seção descreve um conjunto de trabalhos relacionados considerando o problema de reduzir o custo da comunicação baseado em técnicas de poda de modelo. Por exemplo, o FedMP é uma estrutura adaptativa de poda de modelo em AF, que aumenta a eficiência de recursos e acurácia do modelo [Jiang et al. 2022b]. A solução se destaca em cenários heterogêneos, oferecendo desempenho até 4.1 vezes mais rápido que outros métodos. No entanto, desafios como largura de banda de comunicação limitada e heterogeneidade de dispositivos foram anteriormente negligenciados. Ou seja, isso indica a necessidade de soluções mais eficientes que abordem esses desafios em ambientes de AF. O FedMP, embora inovador no AF com poda adaptativa em ambientes heterogêneos, enfrenta desafios. Sua adaptação a cada dispositivo aumenta a complexidade de implementação e pode ser problemática em ambientes com diversidade de dispositivos. Além disso, essa adaptabilidade pode causar variações na qualidade dos modelos entre os nós, afetando potencialmente a coerência e generalização do modelo global.

Li *et al.* [Li et al. 2020] introduzem o LotteryFL, uma abordagem que adapta a hipótese do bilhete de loteria para o contexto de AF. Essa hipótese formula que é viável identificar sub-redes (bilhetes vencedores) que apresentam acurácia igual ou superior à rede não podada e com alta capacidade de generalização. Sendo assim, o LotteryFL foca tanto na personalização dos modelos quanto na redução dos custos de comunicação. Isso permite que cada cliente treine uma sub-rede especializada para permitir uma comunicação eficiente. O LotteryFL foi testado em conjuntos de dados não Independentes e Identicamente Distribuídas (IID), onde mostrou melhorias notáveis em personalização e eficiência de comunicação em comparação com outras abordagens. Essa metodologia é particularmente inovadora ao abordar os desafios associados à distribuição de dados não IID em AF. No entanto, o LotteryFL apresenta algumas desvantagens, como a personalização excessiva que pode limitar a generalização e a identificação/treinamento de sub-redes ótimas para cada cliente que pode ser complexa e exigir muitos recursos.

Lee *et al.* [Lee et al. 2019] apresentaram um método para podar redes neurais antes do treinamento, chamado SNIP. Essa abordagem, baseada na sensibilidade de conexão, identifica conexões estruturalmente importantes, permitindo uma redução significativa na complexidade da rede sem a necessidade de ciclos iterativos de poda e re-treinamento. É aplicável a várias arquiteturas, incluindo redes convolucionais, residuais e recorrentes. O método oferece simplicidade e versatilidade, tornando-o robusto às variações de arquitetura e eliminando a necessidade de pré-treinamento e agendas complexas de poda. No entanto, esta abordagem é centralizada e pode enfrentar problemas com escalabilidade.

Jiang *et al.* [Jiang et al. 2022a] introduziram o PruneFL, uma abordagem de AF que apresenta poda de parâmetros adaptativa e distribuída. A abordagem visa treinar modelos eficientemente em dispositivos de borda com recursos computacionais e de comunicação limitados. O PruneFL alcança isso adaptando o tamanho do modelo durante o treinamento, reduzindo a sobrecarga de comunicação e computação. Além disso, ele incorpora poda inicial em um cliente selecionado, seguido por poda distribuída adici-

onal durante o processo de AF. A abordagem mantém efetivamente a precisão do modelo enquanto reduz significativamente o tempo de treinamento. No entanto, a escolha de apenas um cliente para realizar a poda pode gerar viés.

Isik *et al.* [Isik et al. 2023a] apresentaram o Treinamento de Máscara Probabilística Federada (FedPM). Essa solução melhora a eficiência de comunicação no AF, treinando uma máscara binária estocástica em vez dos pesos do modelo. Ele mantém os pesos em seus valores iniciais aleatórios e descobre uma rede aleatória esparsa ideal dentro da original. Essa abordagem mostra melhorias em acurácia, eficiência de comunicação e velocidade de convergência, tornando-se adequada para cenários com recursos de comunicação limitados. No entanto, seu uso pode degradar outras propriedades do modelo treinado, como a equidade, a robustez e o vazamento de dados. Além disso, a abordagem é complexa de implementar em comparação com o SNIP.

Vallapuram *et al.* [Vallapuram et al. 2022] introduziram uma estrutura de AF que aborda desafios como heterogeneidade estatística e restrições de recursos em dispositivos clientes, chamada HideNSeek. A abordagem combina poda do lado do servidor na inicialização e o uso de uma supermáscara de sinal, na qual os termos satisfazem a condição de pertencer ao conjunto $\{-1, 1\}$. O HideNSeek emprega poda de único disparo no lado do servidor para determinar uma sub-rede, possibilitando uma convergência de modelo mais rápida com taxas de compressão semelhantes aos métodos existentes. O HideNSeek diminui o custo de comunicação, mantendo ao mesmo tempo uma capacidade superior no que diz respeito ao aprendizado de dados com diversas heterogeneidades. No entanto, é possível ocorrer viés ao realizar poda do lado do servidor. Além disso, o desempenho do HideNSeek é decrescente em cenários com alta heterogeneidade de dados.

Chang *et al.* [Chang et al. 2023] propõem um método de poda de modelo para AF para mitigar gargalos de computação e comunicação locais enquanto mantém acurácia semelhante ao modelo original. O método melhora a eficiência de comunicação ao reduzir o número de parâmetros do modelo transmitidos e otimizando a proporção de poda e seleção de dispositivos. O esquema de poda a nível de canal resulta em uma rede comprimida adequada para inferência eficiente em aplicações de Redes Neurais Convolucionais (CNN). Como limitamos o modelo global a um modelo de rede neural simples neste experimento, trabalhos futuros podem ser estendidos para modelos mais complexos e grandes em ambientes de comunicação dinâmicos.

Com base na análise do estado da arte, é evidente que muitos estudos partem da premissa de que os recursos computacionais são amplamente disponíveis e que existe um conhecimento detalhado das informações dos clientes. No entanto, isso pode introduzir complexidades significativas nas configurações, exigindo um alto nível de capacidade computacional. Isso, por sua vez, frequentemente torna essa abordagem impraticável em cenários onde os recursos são limitados ou onde há uma falta de informações detalhadas sobre os clientes. Sendo assim, o FedSNIP tem como objetivo abordar esse desafio reduzindo a quantidade de parâmetros que são comunicados ao servidor. Tal feito é alcançado por meio de uma abordagem de poda não estrutural que não compromete a acurácia do modelo. Dessa forma, a proposta busca tornar o AF mais eficiente e acessível em cenários com recursos limitados ou quando as informações sobre os clientes são escassas.

3. Conceitos Básicos

Esta seção introduz conceitos fundamentais, tais como o aprendizado federado, poda de redes neurais, representações esparsas de modelos e o algoritmo SNIP.

3.1. Aprendizado Federado (AF)

O AF representa uma mudança de paradigma no aprendizado de máquina, enfatizando uma abordagem descentralizada e colaborativa. No AF, ao invés de agregar dados em um servidor central, o processo de treinamento é distribuído entre múltiplos dispositivos ou nós, cada um com seus próprios dados. Essa metodologia não só preserva a privacidade dos dados, mas também utiliza os recursos computacionais dos nós participantes, tornando-se uma solução poderosa para cenários onde a centralização de dados é impraticável ou indesejável.

No âmbito do AF, uma representação matemática formal do treinamento de modelos pode ser formulada da seguinte forma: *(i)* Considere um conjunto $C = \{c_1, c_2, \ldots, c_D\}$, onde cada cliente c_k no sistema AF possui um conjunto de dados local D_k; *(ii)* O objetivo em um cenário de AF é minimizar uma função objetiva global, que é uma composição das funções de perda calculadas por cada cliente com base em seus dados privados. Essa composição pode ser representada matematicamente como:

$$\min_{x \in \mathbb{R}^t} f(x) = \frac{1}{D} \sum_{k=1}^{D} f_k(x; D_k) \tag{1}$$

onde $f_k(x; D_k)$ denota o valor da perda calculado pelo cliente k com base em seu conjunto de dados local D_k.

Como pode ser observado, essa representação reforça a natureza descentralizada e colaborativa do AF, onde cada cliente calcula independentemente a perda com base em seus dados e contribui para o treinamento do modelo global.

3.2. Poda de Redes Neurais e Representações Esparsas

A poda de redes neurais e a criação de representações esparsas são estratégias-chave na otimização da eficiência de comunicação para o AF [Xia et al. 2021]. A poda reduz o tamanho e a complexidade de um modelo ao eliminar pesos que contribuem minimamente para a saída, resultando em uma representação esparsa. Essa esparsidade se traduz em menos parâmetros, menor uso de memória e, o mais importante, redução da sobrecarga de comunicação em ambientes de AF, tornando o processo de treinamento e atualização mais eficiente e prático. De maneira geral, a poda em redes neurais pode ser realizada de duas maneiras principais: *(i)* poda estrutural e *(ii)* poda não estrutural.

A poda estrutural visa reduzir o número de neurônios de uma rede neural. Essa abordagem foca em eliminar unidades neurais inteiras (como neurônios ou camadas inteiras), resultando em uma arquitetura de rede mais enxuta. Isso pode ser benéfico para reduzir a complexidade computacional e o tamanho do modelo, mantendo ou até mesmo melhorando o desempenho em determinadas tarefas. Por outro lado, a poda não estrutural concentra-se em reduzir os pesos da rede neural. Essa abordagem geralmente envolve a aplicação de máscaras que atuam sobre os pesos da rede, efetivamente "desligando" certos pesos ao defini-los como zero ou próximos de zero. Diferente da

poda estrutural, a poda não estrutural não altera a arquitetura da rede, mas ajusta a importância de conexões específicas dentro da estrutura existente. Isso pode ajudar a tornar a rede mais eficiente e menos propensa a *overfitting*, mantendo uma boa capacidade de generalização [Liang et al. 2021, He and Xiao 2023].

Ambas as abordagens de poda são utilizadas para otimizar redes neurais, seja em termos de desempenho computacional, eficiência de armazenamento, ou até mesmo robustez a ruídos e variações nos dados de entrada [Li et al. 2022, Jordao and Pedrini 2021]. Em geral, a poda de redes neurais pode ser formulada como um problema de otimização. Dado um conjunto de dados $D = \{(x_i, y_i)\}_{i=1}^{n}$ e um nível de esparsidade desejado k, por exemplo, o número de pesos não-zero, a formulação pode ser expressa como o seguinte problema de otimização com restrições:

$$\min_{w} L(w; D) = \min \frac{1}{n} \sum_{i=1}^{n} l(w; (x_i, y_i)), \tag{2}$$

$$\text{sujeito a } w \in \mathbb{R}^m, \|w\|_0 \leq k \tag{3}$$

onde $l(\cdot)$ representa a função de perda padrão, w denota o conjunto de parâmetros da rede neural, m significa o número total de parâmetros, e $\| \cdot \|_0$ é a norma L_0 convencional.

3.3. Algoritmo SNIP

O SNIP, que significa "Poda de Rede de Único Disparo Baseada na Sensibilidade de Conexão", é uma técnica inovadora que permite a poda de redes neurais de maneira eficiente antes mesmo do início do treinamento. É uma abordagem centralizada que identifica e elimina as conexões menos importantes dentro da rede, simplificando o modelo sem necessidade de um treinamento extenso. Tal abordagem revela-se particularmente vantajosa no contexto do AF, pois facilita a simplificação do modelo antes de sua distribuição, diminuindo a carga computacional e aumentando a eficiência geral. O SNIP se enquadra no tipo de poda não estrutural comentado anteriormente [Jiang et al. 2022a].

O algoritmo SNIP destaca-se por sua abordagem de poda única realizada na inicialização da rede, antes de qualquer treinamento. Esta técnica é fundamentada na remoção de conexões que se mostram desnecessárias, conforme avaliado pelo impacto de cada uma na função de perda. A formulação do SNIP é expressa da seguinte forma:

$$\min_{\mathbf{C}} \mathcal{L}(\mathbf{W} \odot \mathbf{C}, \mathcal{D}) \tag{4}$$

Nesta formulação, $\mathbf{W}$ representa os pesos da rede, $\mathbf{C}$ é uma máscara binária que identifica as conexões a serem mantidas, $\odot$ denota o produto de Hadamard, $\mathcal{L}$ é a função de perda, e $\mathcal{D}$ refere-se ao conjunto de dados. O objetivo primário é minimizar $\mathcal{L}$ pela seleção adequada de $\mathbf{C}$, visando manter a eficácia do modelo enquanto se reduz sua complexidade. Diferentemente de métodos anteriores, que exigiam análises detalhadas do conjunto de dados, o SNIP alcança uma poda eficiente utilizando apenas um minibatch com uma quantidade razoável de exemplos de treinamento, simplificando significativamente o processo de poda, a sensibilidade corresponde aos gradientes das conexões.

4. FedSNIP

A Figura 1 ilustra o fluxo de funcionamento do FedSNIP. Na Etapa 1, o servidor distribui o modelo aos clientes (Rótulo ①). Na Etapa 2, após receberem o modelo, os clientes

executam a poda utilizando a técnica SNIP, que resulta em uma alta taxa de zeros nos pesos (Rótulo ②). Após o processo de poda, os clientes treinam seus modelos. Para a retransmissão desses pesos ao servidor, técnicas de compressão baseadas em matrizes esparsas são empregadas e então os pesos podados são enviados ao servidor (Rótulo ③). Na Etapa 3, após receber os pesos dos clientes, o servidor aplica uma representação de matrizes esparsas e procede com a agregação dos dados, configurando o modelo para a próxima fase de treinamento (Rótulo ④). Por fim, o servidor envia novamente o modelo agregado aos clientes (Rótulo ⑤) e inicia uma nova rodada de treinamento (Etapa 1).

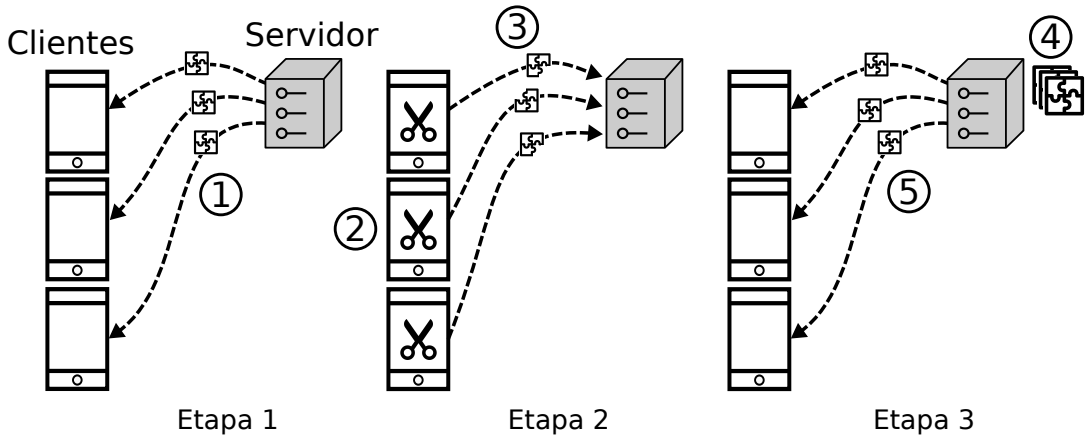


Figura 1. Arquitetura de sistema empregada pelo FedSNIP, com seus principais componentes e como esses componentes interagem entre si ao longo do tempo.

O FedSNIP aplica a técnica de SNIP, conhecida por ser um método de poda sem necessidade de *fine-tuning* e que preserva uma boa acurácia, em um ambiente federado. Assim, a abordagem proposta adapta o SNIP para o contexto de AF, demonstrando a sua eficácia nesse novo cenário, integrando o conceito de poda de único disparo, baseado na sensibilidade de conexão. Cada cliente na rede do AF aplica de forma independente um método de poda baseado em sensibilidade ao seu modelo local durante o treinamento, reduzindo efetivamente o número de parâmetros sem perda significativa de acuracía do modelo.

O núcleo do FedSNIP reside em seu processo único de poda, representando por:

$$W_{pruned} = W \odot M \tag{5}$$

Neste contexto, W denota a matriz de pesos integral à arquitetura da rede neural. Juntamente, M, uma máscara binária, é produzida via análise de sensibilidade. Esta análise identifica crucialmente as conexões menos sensíveis dentro da rede, marcando-as como zero dentro da máscara. Com base nisso, o FedSNIP inicia seu processo estabelecendo um modelo global simbolizado por W. Esse modelo é caracterizado por sua inicialização com pesos atribuídos aleatoriamente.

O Algoritmo 1 descreve as principais etapas envolvidas no funcionamento do FedSNIP. O FedSNIP é uma abordagem de AF que integra a poda de único disparo baseada em sensibilidade no processo de treinamento federado. Cada cliente (Linha 3) na rede federada primeiro poda seu modelo local com base na sensibilidade da conexão, garantindo que os parâmetros restantes estejam abaixo de um limiar θ definido (Linha 5). Esta

etapa é crucial para reduzir a sobrecarga de comunicação e manter a eficiência do modelo. Posteriormente, os clientes treinam localmente seus modelos podados em seus conjuntos de dados (Linha 6). Os modelos atualizados ou seus parâmetros são então enviados de volta ao servidor para agregação (Linha 7). O servidor agrega essas atualizações para refinar o modelo global, que é redistribuído aos clientes para treinamento adicional (Linhas 8 e 9). O processo se repete até que o modelo global convirja ou atenda aos critérios de parada especificados (Linha 2).

Algoritmo 1: Pseudocódigo do FedSNIP

Entrada: Modelo global M, limiar de poda θ
Saída: Modelo global podado e treinado

1 Inicializar modelo global M
2 **para cada** rodada t até a convergência **faça**
3 **para cada** cliente c_k **faça**
4 Receber o modelo global M do servidor
  `/* Poda baseada em sensibilidade */`
5 $M_k \leftarrow \text{PODAMODELO}(M, \theta)$
  `/* Treinamento local */`
6 Treinar M_k em dados locais D_k
7 Enviar atualizações do modelo para o servidor
  `/* Agregação do servidor */`
8 $M \leftarrow AgregaAtualizacoes(M, M_k)$
  `/* Transmissão do modelo atualizado */`
9 Enviar modelo atualizado M para todos os clientes

Adicionalmente, a função PODAMODELO utilizada no FedSNIP é inspirada na abordagem SNIP. O Algoritmo 2 exemplifica o processo empregado nessa etapa. Este método envolve um processo de poda de único disparo aplicado na inicialização da rede neural, baseado na sensibilidade das conexões. A sensibilidade da conexão é determinada calculando o impacto de cada conexão na função de perda, permitindo a identificação das conexões mais cruciais para a tarefa (Linha 4). O processo envolve criar uma pontuação de sensibilidade para cada conexão e reter apenas as conexões superiores com base no nível de esparsidade definido k (Linhas 4 e 7). O modelo podado N_{podado} é gerado com base nas conexões retidas e retornado pela função (Linhas 9 e 10). Essa abordagem permite uma compressão eficaz do modelo sem a necessidade de poda iterativa ou pré-treinamento, tornando-a particularmente adequada para ambientes de AF.

5. Avaliação de Desempenho

Esta seção descreve a avaliação de desempenho do FedSNIP em comparação com soluções da literatura.

5.1. Configuração dos Experimentos

O FedSNIP foi implementado no *framework* Flower [Beutel et al. 2020] e considerando o TensorFlow nas versões 1.6.0 e 2.15.0, respectivamente. No Flower, cada usuário treina

Algoritmo 2: Função PODAMODELO

Entrada: Rede neural não treinada N, conjunto de dados de treinamento D, nível de esparsidade κ
Saída: Modelo podado N_{podado}

Inicializar rede N com pesos aleatórios
Amostrar um mini-lote D_b de D
para cada conexão $j \in N$ **faça**
    Calcular gradiente g_j da perda com relação à conexão
    Calcular pontuação de sensibilidade $s_j = |g_j|$
Normalizar pontuações de sensibilidade
Ordenar conexões por sensibilidade e reter os κ por cento superiores
Podar conexões com menores pontuações de sensibilidade
Gerar modelo podado N_{podado} com base nas conexões retidas
retorna N_{podado}

o modelo com seus dados locais e estabelece comunicação com o servidor central por meio da tecnologia de comunicação gRPC. Além disso, utilizou-se o Ansible[1] para a configuração eficiente de um servidor e 10 máquinas clientes. O Ansible, instalado em um nó de controle com acesso SSH a todas as outras máquinas, facilitou a configuração e implantação dos experimentos.

Neste estudo, considerou-se o conjunto de dados MNIST [LeCun et al. 1998], por ser um *benchmark* bem conhecido e utilizado pela comunidade de aprendizado de máquina. O MNIST compreende 70.000 imagens em escala de cinza de 28×28 pixels, divididas em 60.000 imagens de treinamento e 10.000 imagens de teste. O conjunto de dados é composto por imagens de dígitos manuscritos, variando de 0 a 9, portanto, há 10 classes no total. Cada imagem é rotulada com o dígito correspondente, o que torna o MNIST uma escolha popular para avaliar o desempenho de modelos de aprendizado de máquina, especialmente em tarefas de classificação.

Para a distribuição do conjunto de dados entre os clientes no cenário de AF, empregou-se a distribuição de Dirichlet com o parâmetro $\alpha = 0.5$. Esse método de distribuição, facilitado pela ferramenta FedArtML [Gutierrez et al. 2023], garante uma distribuição de dados não-IID entre os 10 clientes considerados. Tal configuração simula um cenário mais realista onde os dados de cada cliente podem representar um subconjunto diferente do conjunto geral de dados, introduzindo o desafio da heterogeneidade de dados no AF.

Quanto à validação do modelo, os dados de teste de MNIST não foram divididos. Em vez disso, utilizou-se um único conjunto de teste. Essa abordagem implica que todos os clientes efetivamente têm acesso aos mesmos dados de teste (assumindo que não há personalização na fase de avaliação), que é uma prática padrão conhecida como "Avaliação Federada Centralizada" [Soltani et al. 2023]. Com isso, se tem um *benchmark* unificado para comparar o desempenho do modelo federado contra várias configurações e modificações de treinamento.

[1] `https://www.ansible.com/`

O modelo escolhido para os experimentos é o LeNet-5 [LeCun et al. 1998], uma rede neural convolucional conhecida por sua eficiência e desempenho em tarefas de classificação de imagens. O LeNet-5 é caracterizado por seu padrão de conectividade densa, onde cada camada é diretamente conectada a todas as outras camadas de forma sequencial. Essa arquitetura leva a reduções consideráveis no número de parâmetros, tornando-o um excelente candidato para nossos experimentos de AF com o MNIST.

Implementamos também técnicas da literatura para uma comparação, tais como FedAvg [McMahan et al. 2016] e CMFL [Luping et al. 2019]. O primeiro representa a abordagem base para cenários em AF, enquanto o segundo é uma abordagem focada na redução do custo de comunicação, utilizando um critério baseado na porcentagem de concordância de sinais entre o modelo global na rodada $t-1$ e o modelo local na rodada t. Antes de proceder com o envio da atualização local, o método CMFL realiza o cálculo da relevância dessa atualização, empregando o percentual de parâmetros com o mesmo sinal. Qualquer atualização local que apresente um valor abaixo de um determinado limiar é considerada irrelevante, e, por isso, não é encaminhada para a etapa de agregação. Nos experimentos, esse limiar foi configurado em 0.495.

Para avaliar a eficiência do FedSNIP, considerou-se as seguintes métricas:

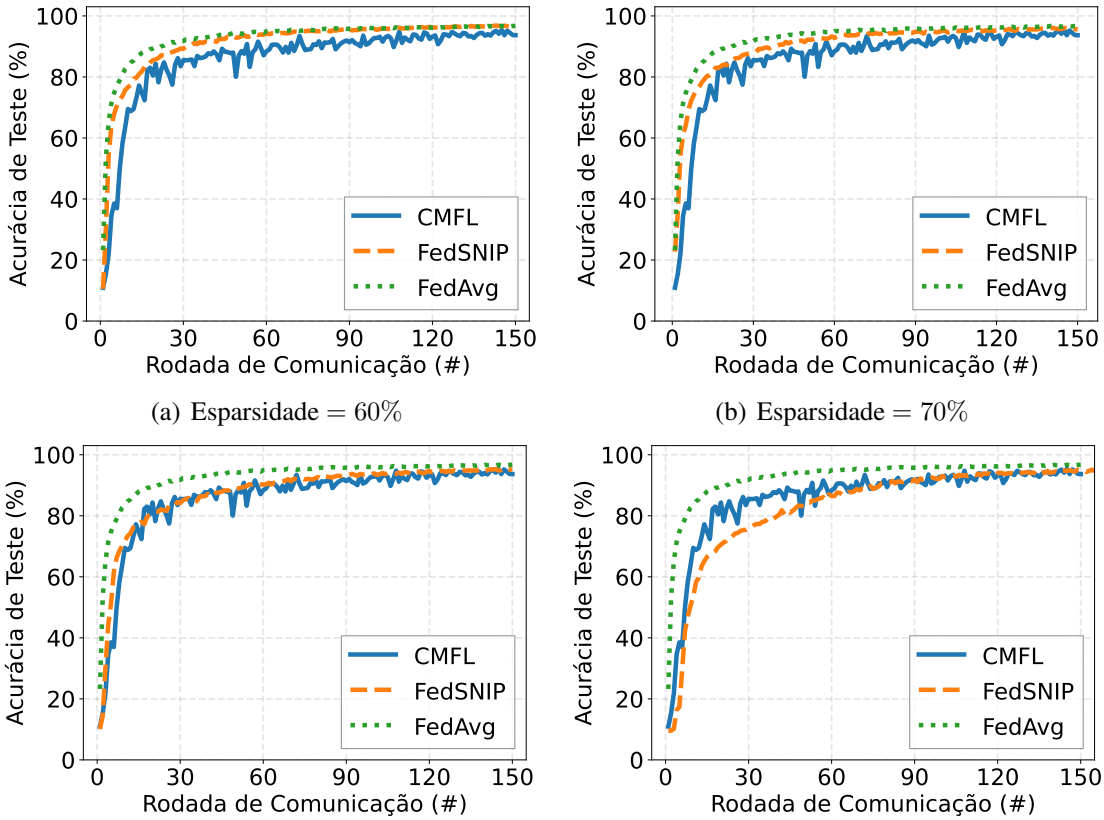
- *Acurácia do Modelo Distribuído:* Essa métrica mede a convergência da solução. É calculada após cada rodada de comunicação para determinar a rapidez com que o modelo atinge a acurácia desejada.
- *Redução Relativa de Parâmetros:* Mede a redução no número de parâmetros devido à poda, em comparação com o modelo original, fornecendo *insights* sobre a eficiência da abordagem FedSNIP.

5.2. Resultados

Esta seção apresenta e discute em detalhes os resultados obtidos, enfatizando especialmente a eficiência na comunicação e a acurácia dos modelos desenvolvidos. Foi realizada uma série de experimentos para avaliar a efetividade do FedSNIP. Uma análise exploratória foi conduzida para estabelecer os melhores hiperparâmetros para as soluções discutidas na literatura. Assim, o valor de taxa de aprendizado foi estabelecido em 0.001, com um tamanho de lote (*batch-size*) de 32, utilizando o método de descida de gradiente estocástica (SGD).

As figuras apresentadas ilustram a acurácia dos modelos distribuídos, além de comparar com métodos convencionais em diferentes níveis de poda. Os experimentos foram realizados com parâmetros de esparsidade variados, especificamente em níveis de 60%, 70%, 80% e 90%, e também comparados com algoritmos da literatura, como o CMFL [Luping et al. 2019] e o FedAvg [McMahan et al. 2016]. Os resultados enfatizam a viabilidade da abordagem proposta, mostrando um equilíbrio notável entre a redução da carga de comunicação e a manutenção da acurácia do modelo.

As Figuras 2(a), 2(b), 2(c), e 2(d) ilustram a acurácia do modelo global em diferentes níveis de esparsidade. Durante os experimentos, notou-se que uma esparsidade de 90% levava a uma diminuição na taxa de convergência. Contudo, observou-se uma inversão positiva dessa tendência ao longo das etapas de comunicação, sugerindo um aprimoramento contínuo na performance do modelo. A abordagem CMFL consegue reduzir os custos de comunicação e ter uma acurácia aceitável, embora em algumas etapas tenha ocorrido



(a) Esparsidade = 60%

(b) Esparsidade = 70%

(c) Esparsidade = 80%

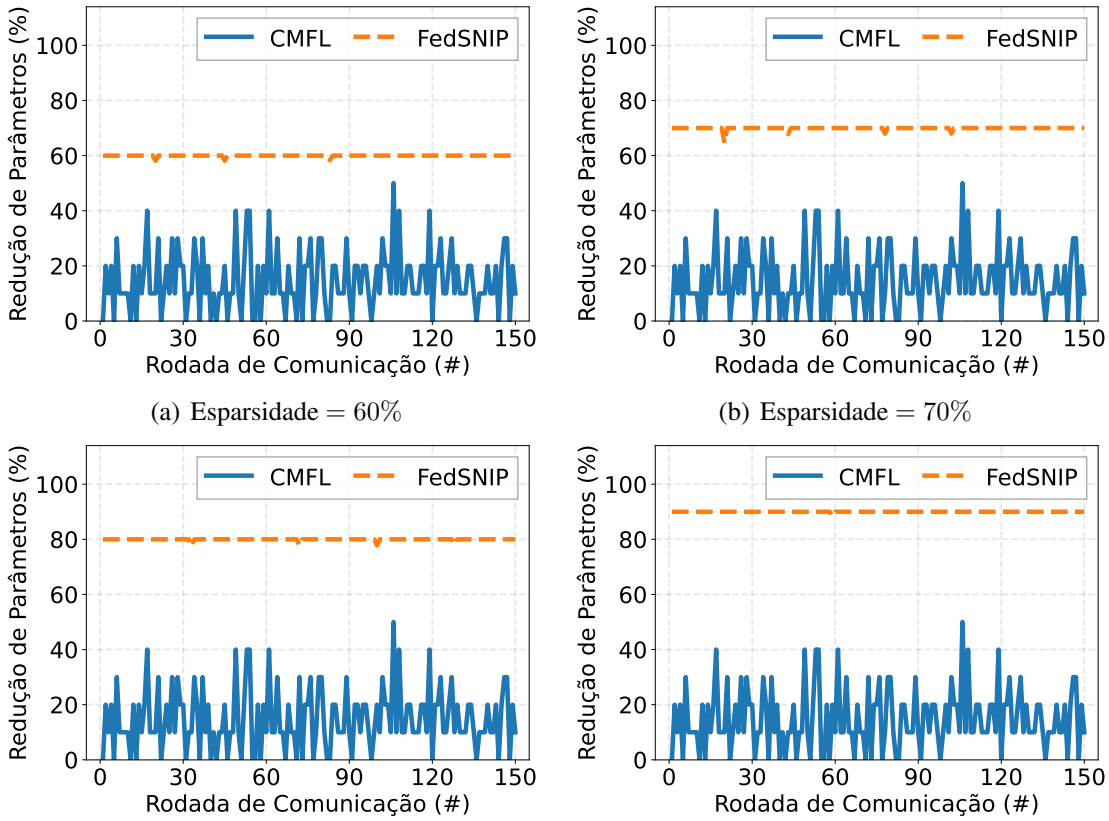
(d) Esparsidade = 90%

Figura 2. Resultados de acurácia de teste do modelo distribuído considerando diferentes níveis de esparsidade.

uma queda na acurácia. Isso se deve ao critério adotado na abordagem, que define se o total de parâmetros será ou não comunicado, influenciando assim a acurácia em determinadas rodadas. As Figuras 3(a), 3(b), 3(c), e 3(d) ilustram a comparação da quantidade de parâmetros comunicados, utilizando o FedAvg como referência. Durante os experimentos, foi observado que o FedSNIP conseguiu uma redução significativamente maior na quantidade de parâmetros comunicados em comparação com o CMFL. A magnitude dessa redução mostrou-se quase constante ao longo dos testes, com algumas variações pontuais.

6. Conclusão

Este artigo apresentou um método para aprimorar a eficiência da comunicação no AF. Nosso método foca na redução da quantidade de parâmetros a serem comunicados ao servidor. Utilizamos uma técnica de poda que dispensa a necessidade de ajuste fino (*fine-tuning*). A estratégia adotada é a de poda não estrutural, que não envolve a eliminação de neurônios, mantendo, ao mesmo tempo, a garantia de uma convergência efetiva no processo de aprendizagem. FedSNIP se destaca pela sua capacidade de manter um equilíbrio entre a quantidade de parâmetros do modelo que não são números zero e a preservação da integridade do modelo, um equilíbrio crucial no processo de AF. Ao aplicar uma esparsidade de até 90%, observou-se que, mesmo com uma redução drástica de parâmetros, os modelos ainda conseguem manter uma acurácia aceitável.



(a) Esparsidade = 60%

(b) Esparsidade = 70%

(c) Esparsidade = 80%

(d) Esparsidade = 90%

Figura 3. Resultados de redução de parâmetros comunicados ao servidor em relação ao FedAvg por rodada de comunicação considerando diferentes níveis de esparsidade.

Como direção para trabalhos futuros, pretende-se explorar métodos avançados para diminuir ainda mais o custo de comunicação. Uma abordagem promissora é a seleção estratégica de clientes, visando alcançar uma redução adicional de parâmetros sem comprometer significativamente a acurácia do modelo. Essa linha de pesquisa aponta para um equilíbrio mais eficiente entre a economia de recursos de comunicação e a qualidade do AF. Além disso, técnicas como *Compressed Sparse Row (CSR)*, *Compressed Sparse Column (CSC)* e *Coordinate List (COO)* podem ser utilizadas para lidar com matrizes com alta proporção de zeros. Em pesquisas futuras, planeja-se realizar uma análise detalhada incorporando essas técnicas. Adicionalmente, será considerada a implementação de métodos como a quantização, com o objetivo de minimizar ainda mais a quantidade de dados transmitidos ao servidor. Por fim, pretende-se ainda integrar técnicas avançadas como a ciência do contexto para otimizar as decisões no processo de poda.

Agradecimentos

Este projeto recebeu apoio do Ministério da Ciência, Tecnologia e Inovações, financiado pela Lei nº 8.248 de 23 de outubro de 1991. Foi desenvolvido sob a égide do Programa Prioritário de Informática (PPI-Softex), gerido pela Softex, e foca em agentes inteligentes para plataformas móveis utilizando tecnologia de Arquitetura Cognitiva (processo nº 01245.013778/2020-21).

Referências

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*.

Chang, M.-K., Chan, Y.-W., and Wu, T.-E. (2023). Communication-Efficient Federated Learning with Model Pruning. In Hung, J. C., Yen, N. Y., and Chang, J.-W., editors, *Frontier Computing*, volume 1031, pages 67–76. Springer Nature Singapore, Singapore. Series Title: Lecture Notes in Electrical Engineering.

de Souza, A. M., Maciel, F., da Costa, J. B., Bittencourt, L. F., Cerqueira, E., Loureiro, A. A., and Villas, L. A. (2024). Adaptive client selection with personalization for communication efficient federated learning. *Ad Hoc Networks*, 157:103462.

Gutierrez, D. M. J., Anagnostopoulos, A., Chatzigiannakis, I., and Vitaletti, A. (2023). Fedartml. `https://pypi.org/project/FedArtML/`. Federated Learning for Artificial Intelligence and Machine Learning library.

He, Y. and Xiao, L. (2023). Structured pruning for deep convolutional neural networks: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, page 1–20.

Isik, B., Pase, F., Gunduz, D., Koyejo, S., Weissman, T., and Zorzi, M. (2023a). Communication-Efficient Federated Learning through Importance Sampling. arXiv:2306.12625 [cs, stat].

Isik, B., Pase, F., Gunduz, D., Weissman, T., and Zorzi, M. (2023b). Sparse Random Networks for Communication-Efficient Federated Learning. arXiv:2209.15328 [cs, stat].

Jiang, Y., Wang, S., Valls, V., Ko, B. J., Lee, W.-H., Leung, K. K., and Tassiulas, L. (2022a). Model Pruning Enables Efficient Federated Learning on Edge Devices. arXiv:1909.12326 [cs, stat].

Jiang, Z., Xu, Y., Xu, H., Wang, Z., Qiao, C., and Zhao, Y. (2022b). FedMP: Federated Learning through Adaptive Model Pruning in Heterogeneous Edge Computing. In *2022 IEEE 38th International Conference on Data Engineering (ICDE)*, pages 767–779, Kuala Lumpur, Malaysia. IEEE.

Jordao, A. and Pedrini, H. (2021). On the effect of pruning on adversarial robustness.

Kairouz, P. and McMahan, H. B. e. a. (2021). *Advances and Open Problems in Federated Learning*. arXiv.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lee, N., Ajanthan, T., and Torr, P. (2019). Snip: Single-shot network pruning based on connection sensitivity. In *International Conference on Learning Representations*.

Li, A., Sun, J., Wang, B., Duan, L., Li, S., Chen, Y., and Li, H. (2020). LotteryFL: Personalized and Communication-Efficient Federated Learning with Lottery Ticket Hypothesis on Non-IID Datasets. arXiv:2008.03371 [cs, stat].

Li, Z., Chen, T., Li, L., Li, B., and Wang, Z. (2022). Can pruning improve certified robustness of neural networks?

Liang, T., Glossner, J., Wang, L., Shi, S., and Zhang, X. (2021). Pruning and quantization for deep neural network acceleration: A survey.

Luping, W., Wei, W., and Bo, L. (2019). Cmfl: Mitigating communication overhead for federated learning. In *2019 IEEE 39th international conference on distributed computing systems (ICDCS)*, pages 954–964. IEEE.

McMahan, H. B., Moore, E., Ramage, D., and y Arcas, B. A. (2016). Federated learning of deep networks using model averaging. *CoRR*, abs/1602.05629.

Renda, A., Frankle, J., and Carbin, M. (2020). Comparing rewinding and fine-tuning in neural network pruning.

Shahid, O., Pouriyeh, S., Parizi, R. M., Sheng, Q. Z., Srivastava, G., and Zhao, L. (2021). Communication Efficiency in Federated Learning: Achievements and Challenges. arXiv:2107.10996 [cs].

Soltani, B., Zhou, Y., Haghighi, V., and Lui, J. C. S. (2023). A survey of federated evaluation in federated learning.

Souza, A., Bittencourt, L., Cerqueira, E., Loureiro, A., and Villas, L. (2023). Dispositivos, eu escolho vocês: Seleção de clientes adaptativa para comunicação eficiente em aprendizado federado. In *Anais do XLI Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, pages 1–14, Porto Alegre, RS, Brasil. SBC.

Vallapuram, A. K., Zhou, P., Kwon, Y. D., Lee, L. H., Xu, H., and Hui, P. (2022). HideNseek: Federated Lottery Ticket via Server-side Pruning and Sign Supermask. arXiv:2206.04385 [cs].

Wen, J., Zhang, Z., Lan, Y., Cui, Z., Cai, J., and Zhang, W. (2023). A survey on federated learning: challenges and applications. *International Journal of Machine Learning and Cybernetics*, 14(2):513–535.

Xia, Q., Ye, W., Tao, Z., Wu, J., and Li, Q. (2021). A survey of federated learning for edge computing: Research problems and solutions. *High-Confidence Computing*, 1(1):100008.