Trabalho de Conclusão de Curso

Aprendizado Federado e Mecanismos de Privacidade Diferencial para preservar dados sensíveis em treinamento de modelos de Machine Learning.

João Pedro Brito Tomé

Orientador:
Prof. Dr. Erick de Andrade Barboza

Universidade Federal de Alagoas
Instituto de Computação
Maceió, Alagoas
Novembro 26, 2024

UNIVERSIDADE FEDERAL DE ALAGOAS
Instituto de Computação

APRENDIZADO FEDERADO E MECANISMOS DE PRIVACIDADE DIFERENCIAL PARA PRESERVAR DADOS SENSÍVEIS EM TREINAMENTO DE MODELOS DE MACHINE LEARNING.

Monografia apresentada como requisito parcial para obtenção do grau de Bacharel em Engenharia de Computação do Instituto de Computação da Universidade Federal de Alagoas.

João Pedro Brito Tomé

Orientador: Prof. Dr. Erick de Andrade Barboza

Comissão Examinadora:

Glauber Rodrigues Leite	Prof. Dr., UFAL
Jobson de Araújo Nascimento	Prof. Dr., UFAL

Maceió, Alagoas
Novembro 26, 2024

Catalogação na fonte
Universidade Federal de Alagoas
Biblioteca Central
Divisão de Tratamento Técnico
Bibliotecário: Valter dos Santos Andrade

T656a Tomé, João Pedro Brito.
Aprendizado federado e mecanismos de privacidade diferencial para preservar dados sensíveis em treinamento de modelos de machine learning / João Pedro Brito Tomé, Maceió – 2024.
33 f.: il.

Orientador: Erick de Andrade Barboza.
Monografia (Trabalho de Conclusão de Curso em Engenharia de Computação) – Universidade Federal de Alagoas, Centro de Tecnologia, Maceió, 2024.

Bibliografia: f. 32-33.

1. Aprendizado federado. 2. Aprendizado de máquina. 3. Privacidade de dados. 4. Differential privacy. 5. Dados pessoais sensíveis – Preservação. I. Título.

CDU: 004.81:159.953.5

Agradecimentos

Primeiramente, expresso minha profunda gratidão a Deus, aos meus familiares, meu sincero agradecimento por seu amor e apoio incondicional. Em especial, às minhas irmãs, Jaíne e Suzane, minha eterna gratidão por estarem ao meu lado em todos os momentos desafiadores da graduação. Sem elas, nada disso teria sido possível. Estendo, ainda, meus sinceros agradecimentos aos meus pais, tios e avós, cujo incentivo e exemplo foram determinantes na construção do meu caminho acadêmico e pessoal.

Agradeço, com igual intensidade, às amizades que cultivei ao longo dessa jornada, pois foram um alicerce imprescindível durante os desafios e conquistas: Ruan, Mateus, Luana, Hiago, John, Lucas Massa, Jhonnye, Hugo, Derek, Bruna, Cabral, Marcus, William Gabriel e Igor. Vocês foram, e continuam sendo, parte essencial desse percurso.

Por fim, deixo meu reconhecimento a todo o corpo docente e aos funcionários do Instituto de Computação, cujas contribuições tornaram essa etapa possível. Em especial, ao meu orientador, Dr. Erick Andrade Barboza, agradeço profundamente pelo apoio constante, pelos conselhos valiosos e pelo incentivo a buscar sempre a excelência.

Novembro 26, 2024, Maceió - AL

Resumo

À medida que os dispositivos móveis continuam a evoluir, seus recursos de detecção e processamento atingiram níveis de sofisticação sem precedentes. Juntamente com os avanços na aprendizagem profunda, esse progresso abriu diversas oportunidades para aplicativos de alto impacto, principalmente em áreas como saúde e sistemas automotivos. A abordagem tradicional do aprendizado de máquina (ML) baseado em nuvem requer a agregação de dados em um servidor ou data center centralizado. No entanto, essa abordagem levanta problemas críticos de latência, comunicação ineficiente e privacidade de dados. As tecnologias predominantes de ML em redes móveis e distribuídas ainda exigem a divulgação de dados pessoais a partes externas. Recentemente, o conceito de Federated Learning (FL) foi introduzido em face da legislação de proteção de dados cada vez mais rigorosa e das crescentes preocupações com a privacidade. No FL, os dispositivos finais usam seus dados locais para treinar um modelo de ML solicitado pelo servidor. Em seguida, os dispositivos enviam atualizações ao modelo, em vez dos dados brutos, para o servidor para agregação. O Federated Learning (FL) tem o potencial de ser uma tecnologia revolucionária em redes móveis, pois permite o treinamento colaborativo de um modelo de ML e, ao mesmo tempo, preserva a privacidade do cliente.

No entanto, informações privadas ainda podem ser descobertas por meio da análise de parâmetros carregados de clientes, por exemplo, pesos treinados em redes neurais profundas. O FL é suscetível a ataques de inferência, que podem ter origem em qualquer parte que contribua para o processo de treino. Nesse sentido, a Differential Privacy (DP) é uma técnica que introduz ruído estatístico controlado nos dados, impedindo a inferência de informações específicas de um participante, mesmo em circunstâncias adversas. Neste trabalho, a DP foi utilizada em combinação com diferentes mecanismos de ruído, como o laplaciano e o gaussiano, para analisar como a DP afeta a precisão do modelo enquanto tenta mitigar esses ataques durante o treino em ambiente de Aprendizagem Federada. Esta integração visa alcançar um equilíbrio entre a privacidade e o desempenho do modelo.

***Palavras-chave*: Aprendizado Federado, Differential Privacy, Privacidade de Dados, Ataques de Inferência, Aprendizado de Máquina.**

Abstract

As mobile devices continue to evolve, their sensing and processing capabilities have reached unprecedented levels of sophistication. Coupled with advances in deep learning, this progress has opened up diverse opportunities for high-impact applications, particularly in areas such as healthcare and automotive systems. The traditional approach to cloud-based machine learning (ML) requires the aggregation of data on a centralised server or data centre. However, this approach raises critical issues of latency, inefficient communication and data privacy. The prevailing technologies for ML in mobile and distributed networks still require the disclosure of personal data to external parties. Recently, the concept of Federated Learning has been introduced in the face of increasingly stringent data protection legislation and growing privacy concerns. In FL, end devices use their local data to train an ML model that is then required by the server. The devices then send updates to the model, rather than the raw data, to the server for aggregation. Federated Learning (FL) has the potential to be a revolutionary technology in mobile networks as it enables collaborative training of an ML model while preserving client privacy.

However, private information can still be discovered by analysing parameters loaded from clients, for example, weights trained on deep neural networks. FL is susceptible to inference attacks, which can originate from any party that contributes to the training process. In this sense, Differential Privacy (DP) is a technique that introduces controlled statistical noise into the data, preventing the inference of participant-specific information, even in adverse circumstances. In this work, DP was used in combination with different noise mechanisms, such as Laplacian and Gaussian, to analyse how DP affects model accuracy while trying to mitigate these attacks during training in a Federated Learning environment. This integration aims to achieve a balance between privacy and model performance.

***Keywords*: *Federated Learning*, *Differential Privacy*, *Data Privacy*, *Inference Attacks*, *Machine Learning*.**

Lista de Figuras

Lista de Abreviaturas e Siglas

ML Machine Learning (Aprendizado de Máquina)

DP Differential Privacy (Privacidade Diferencial)

FEMNIST Federated Extended MNIST (MNIST Estendido Federado)

MNIST Modified National Institute of Standards and Technology database

FL Federated Learning (Aprendizado Federado)

SGD Stochastic Gradient Descent

SMPC Secure Multi-Party Computation

CNN Convolutional Neural Network (Rede Neural Convolucional)

IoT Internet of Things (Internet das Coisas)

IIoT Industrial Internet of Things (Internet das Coisas)

GDPR General Data Protection Regulation (Lei Geral de proteção de Dados)

non-IID Non-Independent and Identically Distributed (não-Independente nem Identicamente Distribuída)

IID Independent and Identically Distributed (Independente e identicamente distribuída)

Sumário

Capítulo 1

Introdução

O avanço acelerado da tecnologia de Internet das Coisas (IoT) tem impulsionado a integração de dispositivos móveis como smartphones, câmeras e sistemas industriais (IIoT) no cotidiano, promovendo um aumento significativo na geração de dados. Dispositivos IoT modernos, dotados de poderosas capacidades de processamento e comunicação, têm o potencial de transformar várias indústrias, desde saúde até manufatura, mas trazem desafios específicos na gestão e utilização desses dados.

A abordagem convencional de processamento baseada na nuvem centraliza os dados coletados por dispositivos móveis em servidores ou centros de dados para análise e modelagem. Por exemplo, informações como medições sensoriais, imagens e vídeos capturados por dispositivos IoT são enviados para a nuvem, onde são agregados e processados para fornecer análises e modelos preditivos eficazes. No entanto, essa metodologia enfrenta barreiras consideráveis.

Primeiramente, a crescente conscientização sobre a privacidade dos dados por parte dos consumidores desencadeou uma série de regulações rigorosas. Legislações como o Regulamento Geral de Proteção de Dados (GDPR) da União Europeia e a Consumer Privacy Bill of Rights nos Estados Unidos estabelecem princípios fundamentais, como o consentimento explícito (Artigo 6º do GDPR) e a minimização da coleta de dados (Artigo 5º do GDPR). Essas diretrizes restringem severamente a coleta e armazenamento de dados pessoais a usos estritamente necessários e autorizados pelos indivíduos (Voigt and Von dem Bussche, 2017).

Além disso, a dependência de sistemas baseados na nuvem apresenta desafios técnicos. Um deles é a latência para aplicações sensíveis ao tempo, como sistemas de direção autônoma, onde decisões precisam ser tomadas em milissegundos para evitar acidentes. Atrasos na propagação de dados pela rede tornam essas soluções inadequadas (Shi et al., 2016). Outra limitação significativa é o uso intensivo de largura de banda, especialmente em aplicações que requerem a transferência de grandes volumes de dados não estruturados, como vídeos em alta resolução. Este cenário é agravado por limitações nas redes de telecomunicações, que podem sofrer com congestionamentos, impedindo o desenvolvimento

de tecnologias que dependem de comunicação eficiente (Zhang et al., 2019).

Esses desafios enfatizam a necessidade de repensar as abordagens tradicionais de processamento de dados em larga escala, promovendo soluções mais distribuídas e privacidade-preservantes, como Federated Learning, que mitigam as limitações impostas pela centralização excessiva dos dados, aliados ao Differential Privacy, para uma camada mais robusta de privacidade.

1.1 Motivação

Para garantir que os dados de treinamento permaneçam em dispositivos pessoais e para facilitar o aprendizado de máquina colaborativo de modelos complexos em dispositivos distribuídos, é apresentada uma abordagem de ML descentralizada chamada Federated Learning (FL) (McMahan et al., 2016). Na FL, os dispositivos móveis usam seus dados locais para treinar de forma cooperativa um modelo de ML exigido por um servidor FL. Em seguida, eles enviam as atualizações do modelo, ou seja, os pesos do modelo, ao servidor FL para agregação. Essas etapas são repetidas em várias rodadas até que a precisão desejada seja alcançada. Isso significa que o FL pode ser uma tecnologia de apoio para o treinamento de modelos de ML em redes de móveis.

Um exemplo de aplicação dessa tecnologia é o Google Gboard, o teclado virtual da Google representado na Figura 1.1. O Federated Learning (FL) solucionou problemas críticos relacionados à segurança dos dados e ao ajuste do modelo de recomendação de texto para cada indivíduo, combinando privacidade e personalização de maneira eficiente. Antes do FL, o envio de dados de digitação (como palavras usadas ou padrões de escrita) para servidores expunha os usuários a possíveis violações de privacidade e a ataques durante a transmissão ou armazenamento. Padrões de escrita podem mudar com o tempo (por exemplo, quando um usuário começa a usar termos técnicos ou gírias). Modelos centralizados não se ajustam rapidamente a essas mudanças. O treinamento local contínuo no dispositivo permite que o modelo acompanhe as mudanças no estilo de escrita do usuário, sem depender de atualizações frequentes no servidor central.

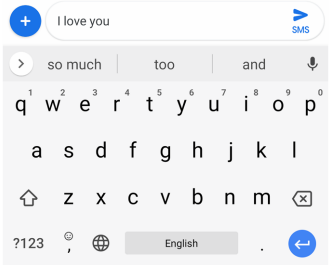

Figura 1.1: Exemplo de aplicação de Federated Learning: Google Gboard Word Suggestions. Fonte: Hard et al. (2018)

Em comparação com as abordagens tradicionais de treinamento centradas na nuvem, a implementação do FL para treinamento de modelos em redes móveis tem as seguintes vantagens:

- Uso altamente eficiente da largura de banda da rede: menos informações precisam ser enviadas para a nuvem. Por exemplo, em vez de enviar dados brutos para processamento, os dispositivos participantes enviam apenas os parâmetros atualizados do modelo para agregação. Como resultado, os custos de comunicação de dados são significativamente reduzidos.

- Privacidade: De acordo com o ponto acima, não há necessidade de enviar os dados brutos dos usuários para a nuvem. Supondo que os participantes e servidores do FL não sejam mal-intencionados, isso aumenta a privacidade do usuário e reduz a probabilidade de ataques até certo ponto. Na verdade, com o aumento da privacidade, mais usuários estarão dispostos a participar do treinamento do modelo, permitindo a criação de modelos melhores.

- Baixa latência: Com o FL, os modelos de ML podem ser treinados e atualizados continuamente. Enquanto isso, as decisões em tempo real, como a detecção de eventos, podem ser tomadas localmente em endpoints. Como resultado, a latência é muito menor do que quando as decisões são tomadas na nuvem antes de serem enviadas para os dispositivos finais. Isso é fundamental para aplicativos de tempo real, como sistemas de carros autônomos, em que o menor atraso pode ser potencialmente fatal.

No entanto, mesmo no modelo descentralizado, a privacidade dos dados não é garantida. Existem ataques de inferência, que buscam explorar as atualizações do modelo enviadas pelos dispositivos (como os pesos treinados), e podem revelar informações privadas dos dados locais, o que representa uma ameaça significativa à privacidade.

Para reduzir esses riscos, a Differential Privacy (DP) (Dwork, 2006) tem se mostrado bastante eficaz nesse sentido. A DP introduz ruído estatístico controlado nas atualizações do modelo ou nos dados, garantindo que informações específicas dos indivíduos sejam indistinguíveis mesmo em circunstâncias adversas. Entre os mecanismos mais comuns utilizados para aplicar a DP, destacam-se o ruído Laplaciano e o Gaussiano, que oferecem diferentes níveis de equilíbrio entre privacidade e impacto na precisão do modelo. Quando combinada com o FL, a DP proporciona uma abordagem robusta de proteção de dados, permitindo que modelos sejam treinados minimizando o risco de exposição de dados pessoais. Esta integração apresenta vantagens significativas, em conformidade com legislações de privacidade mais rigorosas, a preservação dos dados e a possibilidade de escalar o treino para dispositivos distribuídos de maneira segura e eficiente.

1.2 Objetivos

1.2.1 Objetivos Gerais

Analisar o impacto do uso dos mecanismos de ruído da Differential Privacy no desempenho de modelos treinados em um ambiente de aprendizado federado, considerando a relação entre privacidade e eficiência do modelo.

1.2.2 Objetivos Específicos

1. Implementar a integração entre Federated Learning (FL) e Differential Privacy (DP), duas técnicas distintas para proteger a privacidade no treinamento distribuído.

2. Avaliar o desempenho do modelo treinado com os mecanismos de ruído Laplaciano e Gaussiano da DP, nos datasets MNIST, FEMNIST e Fashion-MNIST.

1.3 Organização

Este trabalho foi organizado em capítulos que relatam as etapas seguidas durante o desenvolvimento da análise, entrando em detalhes teóricos e aplicação das tecnologias envolvidas. No Capítulo 2, é fornecido um framework para compreensão dos assuntos abordados neste trabalho, servindo como base para nossa análise e interpretação das informações coletadas, definido como o capítulo de fundamentação teórica. Neste capítulo, são mostrados os conceitos de redes neurais convolucionais, Aprendizado Federado, Differential Privacy, Mecanismos de ruído e métricas de avaliação usadas neste trabalho. Em sucessão, é apresentada uma sequência detalhada de etapas, conhecida como metodologia, delineando a progressão do projeto. Isso inclui explicações das ferramentas empregadas, o processamento de dados, as técnicas para obtenção dos resultados, os detalhes da arquitetura do modelo, mecanismos de ruído e os estágios de implantação. Além disso, estão presentes as descrições dos conjuntos de dados utilizados. Essas etapas foram essenciais para garantir melhor reprodutibilidade em pesquisas futuras e podem ser encontradas no Capítulo 3. Posteriormente, o capítulo de resultados é introduzido, apresentando as descobertas da análise, com uma descrição de cada etapa e seu resultado correspondente derivado da metodologia exibida no capítulo 3. Por fim, há as conclusões sobre este trabalho, exibindo as abordagens e resultados de forma resumida.

Capítulo 2

Fundamentação Teórica

O presente capítulo tem como objetivo apresentar as principais técnicas e ferramentas empregadas nesta pesquisa. Serão detalhados os conceitos de aprendizado federado, privacidade diferencial e redes neurais convolucionais, além das métricas utilizadas para avaliar o desempenho.

2.1 Federated Learning

O crescimento exponencial no uso de dispositivos móveis e IoT (Internet of Things) trouxe desafios significativos para o aprendizado de máquina, especialmente em relação à privacidade e ao processamento de dados distribuídos. Tradicionalmente, modelos de aprendizado de máquina são treinados em servidores centrais, exigindo a transferência de grandes volumes de dados sensíveis, o que aumenta os riscos de vazamento de informações e problemas relacionados à latência e eficiência na comunicação.

Nesse cenário, o Federated Learning (FL), ou Aprendizado Federado, introduzido por McMahan et al. (2016), surge como uma solução para permitir o aprendizado colaborativo sem que os dados brutos precisem deixar os dispositivos onde foram gerados.

O FL é um paradigma de aprendizado descentralizado que visa treinar modelos de aprendizado de máquina utilizando dados distribuídos localmente nos dispositivos dos usuários. É possível observar na Figura 2.1 a arquitetura e o funcionamento do FL, que pode ser resumida nos seguintes passos:

- O servidor central inicializa e distribui um modelo global para os dispositivos clientes.
- Cada dispositivo realiza o treinamento localmente usando seus próprios dados.
- Os dispositivos enviam as atualizações do modelo (como os pesos ajustados) para o servidor.
- O servidor central agrega essas atualizações para atualizar o modelo global.

- O processo se repete em várias rodadas até que o modelo atinja o desempenho desejado.

Vantagens do Federated Learning:

- Preservação da privacidade: Os dados permanecem nos dispositivos locais, reduzindo os riscos associados ao compartilhamento de informações sensíveis.
- Eficiência de comunicação: Apenas atualizações do modelo são transmitidas, reduzindo a necessidade de largura de banda.
- Escalabilidade: Projetado para redes distribuídas massivas, como dispositivos IoT e smartphones.

Apesar de suas vantagens, o FL enfrenta vários desafios. Como dados não independentes e identicamente distribuídos (non-IID), ou seja, os dados locais podem variar significativamente entre os dispositivos, afetando a convergência do modelo global. Diferenças no poder computacional e na conectividade de rede entre os clientes podem impactar o desempenho. Mesmo sem o compartilhamento de dados brutos, ataques de inferência podem explorar as atualizações do modelo para obter informações sensíveis.

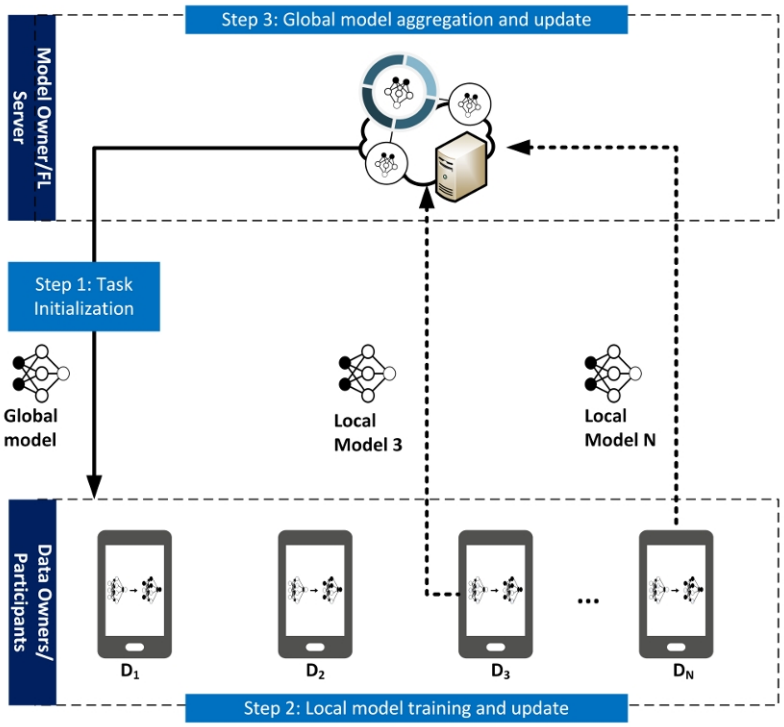

Figura 2.1: Exemplo do processo de treinamento do Aprendizado Federado. Fonte: Lim et al. (2020)

2.2 Differential Privacy

A Differential Privacy (DP) é uma abordagem matemática que busca garantir que informações sensíveis não possam ser inferidas de um conjunto de dados, mesmo quando um adversário possui conhecimento adicional sobre a base ou seus participantes. A formalização da DP foi proposta por Dwork (2006) e ganhou relevância no contexto de aprendizado de máquina como uma ferramenta para mitigar riscos de privacidade.

A DP é definida pela propriedade de ε-privacidade, que afirma que a inclusão ou exclusão de um único dado em um conjunto não altera significativamente o resultado de uma função, também denominado fator de privacidade. Formalmente, isso é expresso como:

$$f(D) \in S \leq e^{\varepsilon} \cdot f(D') \in S \tag{2.1}$$

Onde:

- D e D' são dois conjuntos de dados que diferem por no máximo um elemento.
- $f(D)$ é a função aplicada ao conjunto de dados D.
- S é um subconjunto dos possíveis resultados da função.
- $\varepsilon \in \mathbb{R} \quad \text{e} \quad \varepsilon \in (0, +\infty)$ é o parâmetro de privacidade que controla a intensidade da proteção. Quanto menor o valor de ε, maior a privacidade, mas com maior perda de precisão.

Os principais mecanismos de DP utilizados para adicionar ruído às consultas e garantir a privacidade são o mecanismo Laplaciano e o mecanismo Gaussiano. Ambos visam impedir que ataques externos à base de dados alterem de forma detectável os resultados das consultas, mas utilizam distribuições estatísticas diferentes para adicionar o ruído necessário.

2.2.1 Mecanismo Laplaciano

O Mecanismo Laplaciano é um dos mecanismos mais comuns para garantir a Differential Privacy e foi introduzido por Dwork et al. (2006). A ideia central desse mecanismo é adicionar ruído com uma distribuição Laplace à saída de uma função que é aplicada aos dados. A distribuição Laplaciana é escolhida porque ela tem uma cauda mais pesada do que a distribuição normal, o que significa que tem maior probabilidade de gerar ruídos maiores, ajudando a proteger melhor os dados.

A definição formal do mecanismo é dada pela seguinte fórmula:

$$\text{ruído} \sim \text{Sensibilidade}(f) \cdot \text{Laplace}\left(\frac{1}{\varepsilon}\right) \tag{2.2}$$

Onde:

Sensibilidade(f) é a maior mudança possível na saída da função f causada pela modificação de um único dado do banco de dados. A sensibilidade mede o impacto da alteração de um item de dados sobre o resultado da função.

Laplace($\frac{1}{\varepsilon}$) é a distribuição Laplace com parâmetro $\frac{1}{\varepsilon}$, onde ε é o parâmetro de privacidade (quanto menor ε, maior a privacidade, mas menor a precisão).

A função de densidade de probabilidade da distribuição Laplace é dada por:

$$f(x) = \frac{1}{2b} e^{-\frac{|x-\mu|}{b}} \tag{2.3}$$

Onde:

$b = \frac{1}{\varepsilon}$ e μ é a média da distribuição.

Algumas características e vantagens do Mecanismo Laplaciano são o ε-Differential Privacy. Ou seja, o ruído adicionado baseado na distribuição Laplaciano, garante que a probabilidade de observar qualquer saída de consulta será, no máximo, multiplicada por um fator de e^{ε}, dependendo da presença ou ausência de um único item no banco de dados. A implementação do mecanismo Laplaciano é relativamente simples e eficiente computacionalmente. Como ele apenas adiciona ruído Laplaciano aos resultados das consultas, pode ser usado de forma eficaz em várias situações. A maior limitação do mecanismo Laplaciano é que ele exige o cálculo da sensibilidade da função, ou seja, o quanto a função pode mudar ao modificar um único dado. Para funções com alta sensibilidade, o ruído necessário pode ser grande, o que pode reduzir a precisão da análise.

E suas desvantagens incluem: Alto ruído em funções sensíveis, como já citado acima, quando a função aplicada ao banco de dados possui alta sensibilidade, o ruído adicionado pode ser significativo, o que pode comprometer a utilidade dos dados e diminuir a precisão do modelo. O mecanismo Laplaciano não é ideal para cenários onde os dados estão altamente correlacionados, pois a dependência entre os dados pode ser explorada para reduzir a eficácia do mecanismo.

2.2.2 Mecanismo Gaussiano

O Mecanismo Gaussiano (Dwork et al., 2014) é outra técnica importante para garantir Differential Privacy e foi proposto em trabalhos posteriores como uma alternativa ao mecanismo Laplaciano, principalmente em contextos onde se deseja uma abordagem mais robusta e com melhor desempenho em termos de precisão do modelo. O mecanismo Gaussiano adiciona ruído proveniente de uma distribuição Normal (ou Gaussiana) à saída das funções, o que pode ser preferível em alguns cenários, especialmente quando o ruído Laplaciano pode ser excessivo.

No mecanismo Gaussiano, o ruído adicionado é retirado de uma distribuição Gaussiana

(Normal) com média zero e desvio padrão determinado pela sensibilidade da função e pelo parâmetro ε. Formalmente:

O ruído é dado pela expressão:

$$\text{ruído} \sim \mathcal{N}(0, \sigma^2) \tag{2.4}$$

Onde o ruído é uma variável aleatória com distribuição normal de média 0 e variância σ^2.

σ^2 é determinado pela sensibilidade da função e o parâmetro de privacidade ε, denotado por:

$$\sigma = \frac{\Delta f}{\varepsilon} \tag{2.5}$$

Onde Δf é a sensibilidade da função *f*.

Suas características e vantagens são:

- Maior Precisão: O mecanismo Gaussiano pode ser mais eficiente em termos de precisão, pois, ao invés de adicionar ruído tão grande quanto o do mecanismo Laplaciano (que depende diretamente da sensibilidade), ele permite um controle mais sutil do nível de privacidade.
- O mecanismo Gaussiano pode ser preferido quando a função aplicada ao banco de dados possui alta sensibilidade, pois ele permite uma maior suavização dos resultados.
- O mecanismo Gaussiano é frequentemente utilizado em aprendizado de máquina, especialmente quando o impacto do ruído deve ser minimizado em relação à perda de precisão.

Em desvantagem, ele pode ser mais complexo de implementar e não oferece uma privacidade tão forte quanto a Laplaciana, apesar de ter uma privacidade mais flexível.

Os mecanismos Laplaciano e Gaussiano são dois dos mais utilizados para garantir Differential Privacy. O Laplaciano fornece garantias fortes de privacidade,ideal para consultas simples e funções com baixa sensibilidade. Por outro lado, o Gaussiano oferece maior precisão em cenários mais complexos, mas com uma garantia de privacidade mais flexível. Adiante nos resultados discutiremos como performaram cada um deles.

2.3 Rede Neural Convolucional

As Redes Neurais Convolucionais (CNN) foram propostas como uma técnica específica de redes neurais para explorar a estrutura local de dados com disposição espacial ou sequencial, como imagens e sinais temporais. A sua arquitetura é especialmente desenvolvida para reduzir a complexidade computacional, ao mesmo tempo que melhora a capacidade de generalização e eficiência na extração de padrões hierárquicos (LeCun et al., 1998);(Goodfellow et al., 2017).

As CNNs são definidas como redes que utilizam operações de convolução para processar dados organizados em forma de grades regulares. Cada camada convolucional aplica filtros (kernels) que extraem características locais da entrada, permitindo a identificação de padrões espaciais de diferentes escalas (Gu et al., 2015). Essa característica torna as CNNs uma ferramenta poderosa para lidar com dados como imagens.

A operação de convolução, que é central ao funcionamento das CNNs, é formalizada como:

$$(f * g)(x) = \int_{-\infty}^{\infty} f(t)g(x-t)dt \tag{2.6}$$

De forma discreta:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(i+m, j+n)K(m,n) \tag{2.7}$$

Onde: $\boldsymbol{I}$ é a entrada (por exemplo, uma imagem), $\boldsymbol{K}$ é o filtro (kernel) convolucional, e $S(i,j)$ é o mapa de features resultante.

As CNNs consistem em uma sequência de camadas, cada uma com um papel específico:

- Camada Convolucional: Essa camada aplica filtros sobre a entrada para gerar mapas de características. Os filtros são ajustados durante o treinamento por meio da backpropagation para detectar padrões específicos (LeCun et al., 1998).

- Camada de Pooling: A camada de pooling reduz a dimensionalidade espacial, mantendo as características mais relevantes. Ela pode ser implementada como: Max Pooling: Retém o valor máximo em uma região específica, Average Pooling: Calcula a média dos valores em uma região. Essa etapa melhora a robustez do modelo contra variações, como translações e ruídos (Springenberg et al., 2014).

- Camada de Ativação: Introduz não linearidade no modelo. Funções como ReLU ($f(x) = max(0,x)$) são amplamente usadas para evitar problemas de saturação e melhorar a eficiência computacional (Nair and Hinton, 2010).

- Camadas Completamente Conectadas: Nas últimas etapas, as features extraídas são combinadas para realizar a tarefa de classificação ou regressão. Essas camadas conectam todos os nós da entrada com os nós da saída.

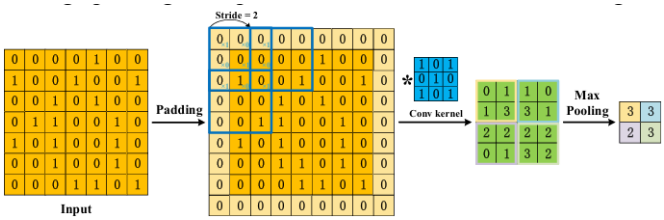

Figura 2.2: Exemplo de CNN de 2 dimensões. Fonte: Li et al. (2021)

Durante o treinamento, as CNNs utilizam o método de descida do gradiente estocástico, como o SGD, para ajustar/otimizar os pesos das conexões. O SGD baseia-se no método de gradiente descendente, que busca minimizar uma função objetivo, geralmente a função de perda, ajustando os parâmetros do modelo em direção ao gradiente negativo da função. Ele se destaca por calcular gradientes utilizando pequenos subconjuntos de dados em vez de todo o conjunto, o que reduz significativamente o custo computacional por iteração.

2.4 Métricas de Avaliação

A fim de avaliar a eficácia do modelo, foram empregadas as métricas de acurácia e perda de entropia cruzada. A acurácia foi escolhida por ser uma métrica intuitiva e amplamente utilizada em problemas de classificação, enquanto a perda de entropia cruzada é uma medida que quantifica o erro cometido por um modelo ao fazer previsões.

2.4.1 Acurácia

A acurácia é uma medida de desempenho de um modelo que avalia a proporção de previsões corretas em relação ao total de exemplos avaliados. Em um contexto de classificação, formalmente, a acurácia é dada pela seguinte expressão:

$$\text{Acurácia} = \frac{\text{Número de predições corretas}}{\text{Número total de exemplos}} = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}(\hat{y}_i = y_i), \tag{2.8}$$

Onde: N é o número total de exemplos, $\hat{y}_i$ é a predição do modelo para o exemplo i, y_i é o rótulo verdadeiro para o exemplo i, $\mathbb{1}(\hat{y}_i = y_i)$ é uma função indicadora que indica 1 caso $\hat{y}_i = y_i$, e 0 caso contrário.

2.4.2 Perda

A função de perda é essencial para otimizar o modelo, pois ela indica o quão longe a previsão do modelo está do valor real. Para tarefas de classificação, no caso desse trabalho, a função de perda é a entropia cruzada. Para classificação binária é dada pela seguinte expressão:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)\right], \tag{2.9}$$

Onde: y_i representa o rótulo verdadeiro (0 ou 1) do exemplo i, $\hat{y}_i$ é a probabilidade predita pelo modelo para a classe positiva, N é o número total de exemplos no conjunto de dados.

No caso de classificação multiclasse, a fórmula é generalizada para:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C} y_{i,c} \log(\hat{y}_{i,c}), \tag{2.10}$$

Onde: N é o número total de exemplos no conjunto de dados, C é o número de classes, $y_{i,c}$ é um valor binário (0 ou 1) que indica se o exemplo i pertence à classe c, $\hat{y}_{i,c}$ é a probabilidade predita pelo modelo para que o exemplo i pertença à classe c, a soma interna $\sum_{c=1}^{C}$ percorre todas as classes para cada exemplo.

Capítulo 3

Metodologia

Neste capítulo, serão detalhadas as ferramentas, técnicas e procedimentos empregados na realização desta pesquisa. A metodologia adotada visa garantir a reprodutibilidade dos resultados e a validação das hipóteses propostas.

3.1 Ambiente Experimental

Os experimentos foram conduzidos utilizando a biblioteca PyTorch 2.5.1+cuda118[1] principal framework para a construção e treinamento das redes neurais no projeto, juntamente com a biblioteca Opacus 1.1.1[2]. O Opacus é uma extensão oficial do PyTorch desenvolvida para facilitar a aplicação de Differential Privacy (DP) em modelos de aprendizado de máquina. Ela oferece implementações eficientes e otimizadas de técnicas de DP diretamente integradas com a API de PyTorch.

3.1.1 Datasets

A princípio, por se tratar de um treinamento envolvendo imagens, foi utilizado o modelo CNN com algumas variações entre cada dataset, essa variações foram ajustadas para performar de acordo com o dataset testado e serão discutidas mais adiante.

Dados IID são dados onde cada amostra é independente das outras e todas seguem a mesma distribuição de probabilidade. Enquanto Dados não-IID são dados onde as amostras não são independentes ou não seguem a mesma distribuição de probabilidade. Dados IID e não-IID Representam duas distribuições distintas de dados em aprendizado de máquina. No contexto do Federated Learning, a natureza IID ou não-IID dos dados tem um impacto significativo. Dados IID facilitam o treinamento de modelos globais, pois as distribuições locais são semelhantes. Em dados não-IID, as amostras não são independentes umas das outras e não seguem a mesma distribuição de probabilidade. Isso

[1]Disponível em: https://download.pytorch.org/whl/cu118
[2]Disponível em: https://opacus.ai/#quickstart

significa que os dados de cada cliente (dispositivo) podem ter características distintas, como diferentes classes, diferentes distribuições de classes ou diferentes quantidades de dados.

Nesse sentido, para dados não-IID os modelos locais treinados em dados muito diferentes podem ter parâmetros divergentes, tornando difícil encontrar um modelo global que se adapte a todos os clientes ou até mesmo esteja enviesado para algum rótulo. O modelo global treinado em dados não-IID pode apresentar desempenho desigual nos diferentes clientes.

Para os experimentos foram considerados: Para o cenário IID (MNIST, Fashion-MNIST), os dados são ordenados aleatoriamente e divididos em 100 partes iguais, distribuídas entre os usuários. Para o cenário não-IID (FEMNIST), os dados são ordenados por rótulo e divididos em diferentes partições. Em seguida, eles são distribuídos aos clientes de modo que cada cliente receba um dataset não-IID.

Os datasets utilizados na realização desse trabalho foram:

- MNIST: Conjunto de dados de dígitos manuscritos composto por 60.000 imagens em escala de cinza de 28x28 pixels, divididas em 50.000 imagens de treinamento e 10.000 imagens de teste.

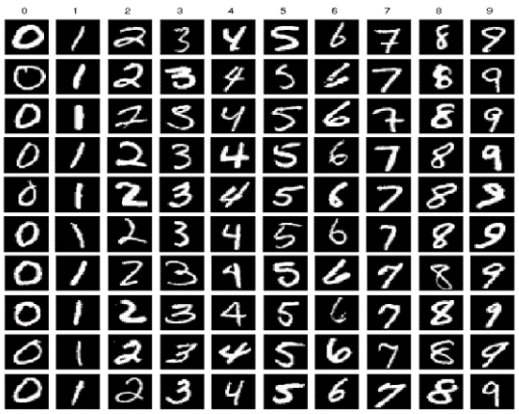

Figura 3.1: Dataset MNIST. (Kadam et al., 2020)

- Fashion-MNIST: Conjunto de dados de artigos de vestuário, composto por 60.000 imagens em escala de cinza de 28x28 pixels, divididas em 50.000 imagens de treinamento e 10.000 imagens de teste.

- Federated Extended MNIST (FEMNIST): Conjunto de dados de caracteres escritos à mão, derivado do dataset MNIST, construídos a partir de imagens manuscritas de milhares de usuários. Ele é composto por 62 classes, representando letras minúsculas e maiúsculas do alfabeto inglês, além de dígitos. A distribuição dos dados entre os clientes é não-IID. Dividido em 90% para treino e 10% para teste.

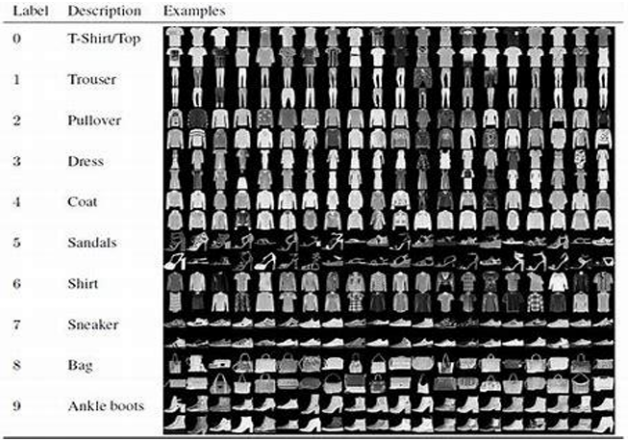

Figura 3.2: Dataset Fashion-MNIST. (Kadam et al., 2020)

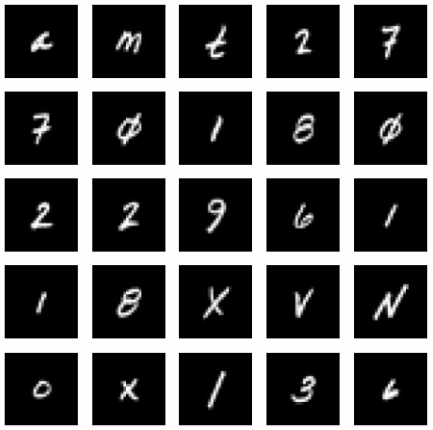

Figura 3.3: Dataset FEMNIST. (Caldas et al., 2018)

A escolha dos datasets foi feita com base em sua ampla utilização pela comunidade acadêmica, o que facilita a comparação dos resultados e a fácil disponibilidade. Os datasets possuem um tamanho aceitável para serem utilizados em dispositivos móveis, sem exigir recursos computacionais excessivos. Seu pré-processamento é mais facilitado, também como para técnicas de Data Augmentation. O dataset FEMNIST, em particular, possui uma distribuição não-IID dos dados, cada pessoa escreve os números de forma diferente, com variações na caligrafia, tamanho e proporções das letras. Isso significa que a distribuição das classes (os diferentes dígitos) varia significativamente entre os escritores. O que é comum em cenários reais de aprendizado federado, onde os dados de cada cliente podem apresentar características distintas entre os clientes e correlacionados para o mesmo cliente.

3.1.2 Hardware

Componente	Especificação
Processador	12th Gen Intel(R) Core(TM) i5-12500H, 2.50 GHz
Memória RAM	16,0 GB (utilizável: 15,7 GB)
Sistema Operacional	Windows 11, 64 bits
Placa de Vídeo Dedicada	NVIDIA GeForce RTX 3050 Laptop GPU, 4GB VRAM

Tabela 3.1: Especificações da Máquina Utilizada.

3.2 Modelos CNNs

A fim de adequar o modelo para o respectivos datasets foram construídos dois modelos: O modelo CNNMnist e CNNFEMNIST. Ambos são redes neurais convolucionais (CNNs) projetadas para classificar imagens de diferentes conjuntos de dados (MNIST, Fashion-MNIST e FEMNIST, respectivamente), com foco em Aprendizado Federado e Differential Privacy.

O modelo CNNMnist é uma rede convolucional simples, projetada para a classificação das imagens do MNIST, um dataset padrão contendo imagens de dígitos manuscritos. A estrutura da rede segue a arquitetura clássica de CNNs, combinando camadas convolucionais, de pooling e totalmente conectadas, com o objetivo de extrair features das imagens e realizar a classificação de forma eficiente. A metodologia utilizada neste modelo pode ser detalhada da seguinte forma:

- Camadas Convolucionais:
  - Conv1: A primeira camada convolucional possui 10 filtros com tamanho de kernel 5x5, projetados para extrair features locais iniciais das imagens (como bordas e texturas simples).
  - Conv2: A segunda camada convolucional possui 20 filtros de tamanho 5x5 e é aplicada após a primeira camada convolucional para capturar features mais complexas.
- Max Pooling e Dropout:
  - Após cada camada convolucional, é aplicada uma operação de max pooling com tamanho 2x2 para reduzir a resolução das imagens e controlar o overfitting.
  - A camada Dropout2D após a segunda convolução visa prevenir overfitting, desativando aleatoriamente alguns neurônios durante o treinamento.

- Camadas totalmente conectadas:
  - fc1: A primeira camada totalmente conectada transforma a saída da camada convolucional em um vetor de 50 unidades.
  - fc2: A segunda camada totalmente conectada gera a classificação final, com um número de unidades igual ao número de classes do dataset (10 para MNIST).
- Função de Ativação e Saída:
  - Para as camadas convolucionais e totalmente conectadas, utiliza-se a função de ativação ReLU (Rectified Linear Unit).
  - A saída final é processada por uma função log_softmax, que é adequada para classificação em múltiplas classes, como no caso do MNIST.

Esse modelo é relativamente simples, adequado para datasets com imagens de baixa complexidade, como o MNIST e o Fashion-MNIST.

O modelo CNNFEMNIST é uma adaptação do CNNMnist, mas voltado para o dataset FEMNIST, que consiste em imagens manuscritas de caracteres de múltiplos participantes. O FEMNIST possui características não-IID, já que os dados são distribuídos de forma desigual entre os diferentes clientes no cenário de aprendizado federado. A estrutura do CNNFEMNIST é ligeiramente diferente para se adequar às complexidades das imagens e características dos dados:

- Camadas Convolucionais:
  - Conv1: A primeira camada convolucional possui 32 filtros com um kernel de 7x7 e um padding de 3, o que permite capturar features de alta escala da escrita manual, com um tamanho de kernel maior devido à resolução das imagens.
  - Conv2: A segunda camada convolucional possui 64 filtros de 3x3 com padding de 1.
- Pooling e Ativação:
  - A operação MaxPooling é aplicada após cada camada convolucional para reduzir a dimensionalidade e focar nas features mais significativas.
  - Foi utilizado a função de ativação ReLU em ambas as camadas convolucionais.

- Camadas totalmente conectadas:
  - A camada totalmente conectada (out) recebe a saída da camada convolucional e gera uma saída com 62 unidades, que corresponde ao número de classes do dataset FEMNIST.

A estrutura do CNNFEMNIST é projetada para lidar com um conjunto de dados mais complexo e de maior dimensionalidade do que o MNIST, com maior número de classes e características mais variadas.

3.3 Mecanismos Gaussiano e Laplaciano

Após definidos e configurados os datasets, deve-se selecionar os mecanismos de ruído da Differential Privacy. A função que adiciona o ruído tem como objetivo aplicar ruído diferencial aos gradientes da CNN durante o treinamento. Durante esse processo os seguintes cálculos são levados em consideração:

- Cálculo da Sensibilidade: A função começa calculando a sensibilidade dos gradientes, a qual depende de três parâmetros: a taxa de aprendizado (lr), o limite de clipping de gradientes [3], e o número de amostras. A sensibilidade mede a magnitude máxima que a função de perda pode mudar com a inclusão ou exclusão de uma amostra específica, o que é essencial para determinar a quantidade de ruído que deve ser adicionada.

- Adição de ruído: Para o Mecanismo Laplaciano o ruído é gerado a partir do cálculo da sensibilidade e do fator de escala do ruído (ε). Para o Gaussiano o ruído é gerado a partir de uma distribuição normal (Gaussiana) com média zero e desvio padrão proporcional à sensibilidade e ao fator de escala do ruído (ε).

- Após o cálculo da sensibilidade e a escolha do mecanismo de ruído, o ruído é adicionado aos parâmetros da rede neural (pesos) de cada camada. Para em seguida, serem carregados os parâmetros modificados de volta para o modelo.

Este processo é repetido para cada iteração no treinamento do modelo, garantindo que o ruído seja aplicado durante todo o processo.

[3] Clipping de Gradientes é uma técnica usada para controlar a magnitude das atualizações dos gradientes durante o treinamento. O objetivo principal é evitar que gradientes de qualquer cliente ultrapassem um valor máximo predeterminado (Abadi et al., 2016)

3.4 Processo da Federated Learning

Nesta seção, será detalhado todo o processo de Aprendizado Federado (Federated Learning) realizado neste trabalho. Este processo foi dividido em etapas bem definidas, com o objetivo de implementar e avaliar um sistema de aprendizado federado utilizando mecanismos de Differential Privacy. A descrição detalhada das etapas é apresentada a seguir:

- Seleção e Pré-processamento do Dataset: O primeiro passo é selecionar o dataset a ser utilizado nos experimentos, considerando as opções MNIST, Fashion-MNIST ou FEMNIST. Para os datasets MNIST e Fashion-MNIST, os dados foram normalizados para garantir valores padronizados e adequados ao treinamento do modelo. No caso do dataset FEMNIST, devido às suas características específicas e ao formato descentralizado, foi utilizado um script de pré-processamento dedicado disponível no repositório LEAF [4](Caldas et al., 2018), garantindo a preparação adequada dos dados para o treinamento federado.

- Seleção do Modelo Correspondente: Após o pré-processamento dos dados, é selecionado o modelo de aprendizado profundo correspondente ao dataset escolhido. Para os datasets MNIST e Fashion-MNIST, foi utilizada a arquitetura CNNMNIST, enquanto para o dataset FEMNIST foi adotada a arquitetura CNNFEMNIST, ambas otimizadas para as respectivas características dos dados.

- Escolha do Mecanismo de Privacidade Diferencial: Em seguida, deve ser escolhido o mecanismo de Differential Privacy (DP) a ser aplicado no sistema. Foram considerados dois tipos de mecanismos para a adição de ruído aos parâmetros do modelo: o mecanismo Gaussiano e o mecanismo Laplaciano

- Treinamento Inicial do Modelo: Antes de iniciar o processo de aprendizado federado, o modelo global é submetido a um treinamento inicial com os dados sintéticos, de forma a gerar os parâmetros iniciais. Esses parâmetros foram então distribuídos para todos os clientes participantes do aprendizado federado.

[4](https://leaf.cmu.edu/)

- Execução do Aprendizado Federado: O aprendizado federado é iniciado com os parâmetros iniciais do modelo global.
  - Treinamento Local: Para cada época, os clientes participantes utilizaram os dados locais para treinar seus modelos individuais.
  - Adição de Ruído aos Parâmetros: Após o treinamento local, será aplicado ruído aos parâmetros do modelo, ajustado utilizando o mecanismo de privacidade diferencial selecionado (Gaussiano ou Laplaciano).
  - Envio dos Parâmetros ao Servidor Global: Os parâmetros ajustados e protegidos com ruído são enviados ao modelo global para agregação.
  - Agregação dos Parâmetros no Servidor Global: No modelo global, é utilizada a técnica FedAvg (Federated Weighted Average) (McMahan et al., 2016) para a agregação dos parâmetros recebidos de cada cliente. Este método pondera os parâmetros com base no tamanho dos conjuntos de dados locais de cada cliente.
  - Após cada época de treinamento, o modelo global foi avaliado utilizando métricas de acurácia e perda. Os resultados dessa avaliação foram registrados, servindo como indicadores de desempenho ao longo do treinamento federado.
  - Repetimos esse processo até a última época.
- Finalização e Análise dos Resultados: Após o término do número total de épocas de treinamento, os resultados obtidos são organizados em gráficos e tabelas para análise. Esses dados incluem métricas de evolução da acurácia e da perda ao longo das épocas, além de comparações entre os mecanismos de privacidade diferencial utilizados.

No treinamento local da Federated Learning apresentada, o foco é simular o comportamento de dispositivos clientes que realizam ajustes ao modelo global de forma independente, utilizando seus próprios dados. Este processo foi implementado de maneira controlada, utilizando algoritmos e frameworks para simular o ambiente local. O algoritmo desse treinamento é composto de vários objetos que representam os dispositivos emulados, cada um está responsável com sua divisão de dados (simulando os dados privados de cada dispositivo no treinamento), onde simultaneamente ocorre o treinamento da FL. O código organiza isso em um loop onde múltiplos clientes são simulados em paralelo.

Através desse processo, foi possível implementar e avaliar o aprendizado federado com mecanismos de privacidade diferencial, oferecendo análise significativas para a compreensão da eficiência e a proteção dos dados no contexto de aprendizado distribuído. A seguir a visualização desse processo, denotado na Figura 3.4.

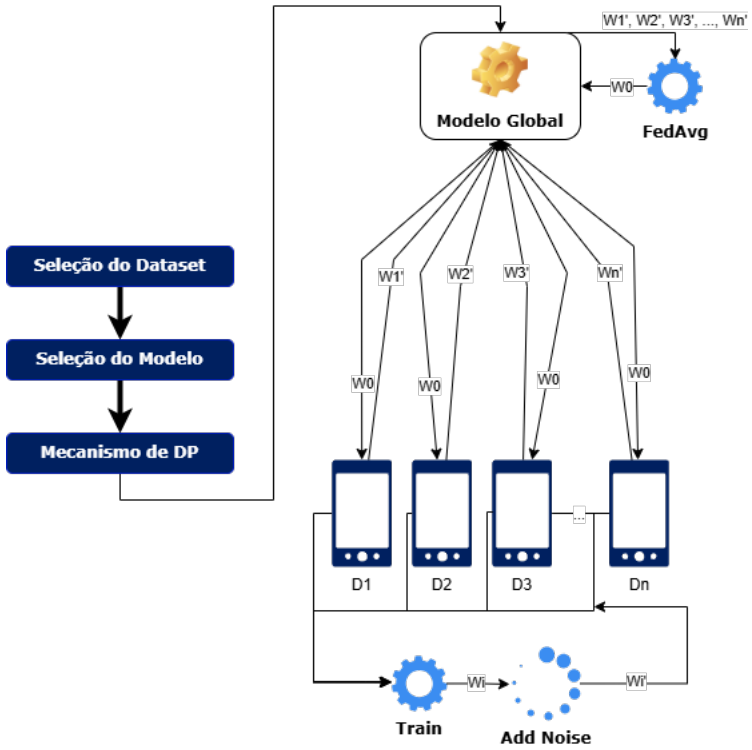

Figura 3.4: Processo do aprendizado federado com privacidade diferencial (Simulação). Fonte: Autor.

Para a obtenção dos resultados, foi necessário definir alguns parâmetros específicos relacionados à rede neural, à configuração do Federated Learning e às etapas do treinamento. Esses parâmetros foram ajustados de forma a garantir a validade e consistência dos experimentos realizados. A Tabela 3.2 a seguir apresenta detalhadamente os valores atribuídos a cada um desses parâmetros, servindo como base para o teste realizado.

Parâmetro	Valores
ε (fator de ruído)	1, 5, 10, 20, 30
Número de clientes (dispositivos)	100
Modelo	CNN
Loss Function	Cross Entropy
Optimizer	Stochastic Gradient Descent (SGD)
Épocas de treinamento	100
Learning rate	0.01
Métricas analisadas	Acurácia, Perda, Tempo de execução

Tabela 3.2: Parâmetros utilizados no Experimento

Capítulo 4

Resultados

Neste capítulo, serão apresentados os resultados obtidos, considerando a eficácia da abordagem proposta para o treinamento federado em conjunto com os mecanismos de Differential Privacy utilizados, além da avaliação do impacto dessas ferramentas no equilíbrio entre privacidade e desempenho do modelo.

Para demonstrar o potencial dessa abordagem, foram realizados testes com diferentes valores de ε e múltiplos datasets, avaliando como os mecanismos Laplaciano e Gaussiano impactam a acurácia e a perda do modelo. Essa estratégia foi essencial para validar a aplicabilidade do ambiente proposto em cenários reais de treinamento de modelos de aprendizado federado, com foco na preservação da privacidade.

Além disso, serão discutidas informações detalhadas sobre o desempenho do sistema, incluindo os resultados das métricas de avaliação, os tempos de execução e gráficos comparativos entre os diferentes mecanismos utilizados. A determinação de intervalos adequados de ε é um ponto central da análise, visto que permite adaptar o ambiente a diferentes contextos e exigências de privacidade, validando sua aplicabilidade em cenários reais de treinamento de modelos de aprendizado federado.

4.1 MNIST

Nesta seção de resultados, apresentamos uma análise detalhada do desempenho dos diferentes mecanismos de ruído da Diffential Privacy (DP) na tarefa de classificação de dígitos manuscritos do dataset MNIST. Através de experimentos com os mecanismos de Laplace e Gauss, avaliamos o impacto da variação do parâmetro epsilon na acurácia, perda e tempo de treinamento do modelo.

Foram construídos gráficos comparativos que ilustram a evolução das métricas de desempenho para cada valor de epsilon em função do número de épocas de treinamento para cada mecanismo. Além disso, a Tabela 4.1 apresenta os resultados numéricos obtidos para cada combinação de mecanismo e valor de epsilon.

Dataset	Epsilon	Acurácia_max	Acurácia_min	Acurácia_avg	Loss_max	Loss_min	Loss_avg	Exec. Time (s)
MNIST								
Gauss Mecanism	1	18.730	6.650	12.796	326.459	2.875	148.283	296.472
Gauss Mecanism	5	62.220	10.400	48.170	2.301	1.325	1.682	295.286
Gauss Mecanism	10	82.260	10.500	65.892	2.296	0.597	1.211	228.01
Gauss Mecanism	20	86.530	11.130	68.895	2.294	0.454	1.146	307.411
Gauss Mecanism	30	86.700	10.910	68.586	2.293	0.448	1.151	278.042
No Mecanism	{+∞}	91.250	11.270	71.712	2.291	0.324	1.034	459.991
Laplace Mecanism	1	19.150	3.950	9.778	10.222	2.301	5.374	553.1
Laplace Mecanism	5	18.560	9.590	12.893	2.301	2.260	2.286	632.934
Laplace Mecanism	10	16.790	9.900	11.790	2.302	2.270	2.289	501.022
Laplace Mecanism	20	18.020	9.910	12.190	2.302	2.273	2.289	287.438
Laplace Mecanism	30	18.560	9.850	12.404	2.303	2.274	2.288	305.461

Tabela 4.1: Resultados do processo descritos em 3.4 para o dataset MNIST. Fonte: Autor.

A partir do resultados apresentados na Tabela 4.1 podemos observar as seguintes tendências:

- Mecanismo de Gauss: A acurácia aumenta significativamente com o aumento do valor de epsilon. A perda, por sua vez, diminui consistentemente com o aumento de epsilon. Isso sugere que, para o mecanismo de Gauss, diminuir o ruído pode contribuir com o aumento do desempenho do modelo.
- Mecanismo de Laplace: A acurácia do mecanismo de Laplace também aumenta com o aumento de epsilon, mas em uma proporção muito menor comparado ao mecanismo de Gauss. A perda, assim como no caso do mecanismo de Gauss, diminui com o aumento de epsilon.

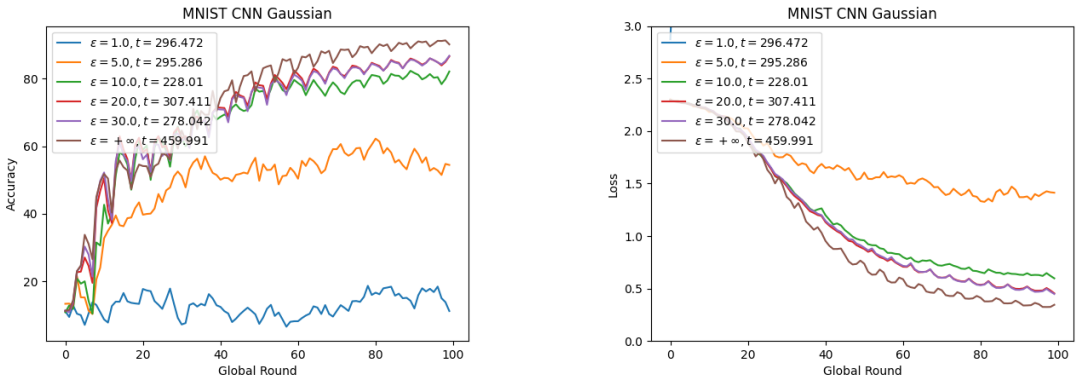

Figura 4.1: Acurácia e perda GaussDP no dataset MNIST. Fonte: Autor.

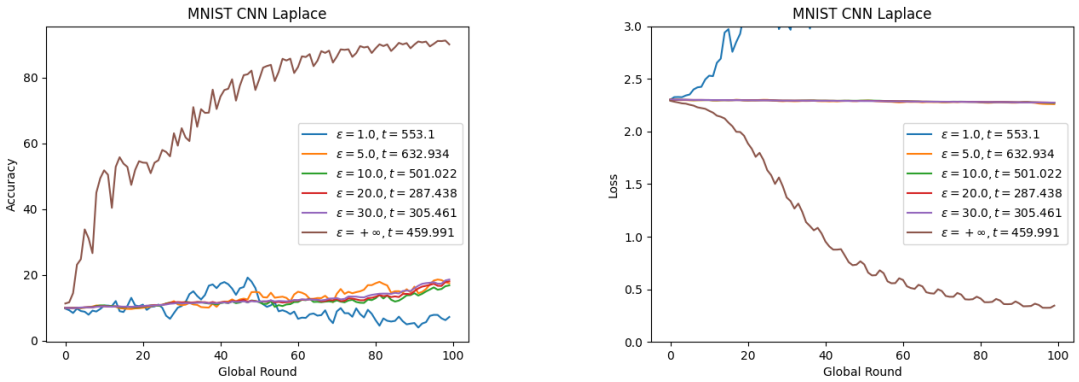

Figura 4.2: Acurácia e perda LaplaceDP no dataset MNIST. Fonte: Autor.

As Figuras 4.1 e 4.2 apresentadas exibem a evolução da acurácia durante as rodadas globais de treinamento no dataset MNIST, utilizando o mecanismo Gaussiano e Laplaciano de Differential Privacy (DP) com diferentes valores de ε. Cada curva representa o desempenho do modelo com um valor específico de ε, incluindo um caso sem aplicação de privacidade $\varepsilon = +\infty$, que serve como referência para a acurácia ideal sem ruído.

- Impacto do ε
  - Para valores menores de ε (e.g., $\varepsilon = 1.0$, observa-se uma acurácia significativamente mais baixa ao longo de todas as rodadas, devido ao forte ruído introduzido pelos mecanismos Gaussiano e Laplaciano, que protegem a privacidade ao custo de reduzir o desempenho do modelo.
  - À medida que ε aumenta, o modelo apresenta melhorias progressivas na acurácia. Isso ocorre porque a intensidade do ruído diminui, permitindo ao modelo capturar melhor os padrões dos dados.
  - O caso $\varepsilon = +\infty$ (sem ruído) alcança a maior acurácia ($\sim 86\%$) para o Gaussiano e ($\sim 19\%$) para o Laplaciano.
  - O tempo total de execução para cada valor de ε é exibido na legenda do gráfico. Observa-se que, para valores menores de ε, há um aumento no tempo de execução no mecanismo Laplaciano. Isso pode ser atribuído ao maior número de operações necessárias para lidar com a intensidade do ruído Laplaciano.
- Trade-off Privacidade-Desempenho: Valores intermediários de ε, como 10.0 ou 20.0, apresentam um compromisso razoável entre privacidade e desempenho do modelo. Esses valores permitem ao modelo alcançar acurácia próxima ao caso sem ruído, enquanto ainda introduzem um nível moderado de proteção de privacidade.

4.2 FEMNIST

Nesta seção de resultados, apresentamos uma análise detalhada do desempenho dos diferentes mecanismos de ruído da Diffential Privacy (DP) na tarefa de classificação dos caracteres manuscritos do dataset FEMNIST. Através de experimentos com os mecanismos de Laplace e Gauss, avaliamos o impacto da variação do parâmetro epsilon na acurácia, perda e tempo de treinamento do modelo.

Foram construídos gráficos comparativos que ilustram a evolução das métricas de desempenho para cada valor de epsilon em função do número de épocas de treinamento para cada mecanismo. Além disso, a Tabela 4.2 apresenta os resultados numéricos obtidos para cada combinação de mecanismo e valor de epsilon.

Dataset	Epsilon	Acurácia_max	Acurácia_min	Acurácia_avg	Loss_max	Loss_min	Loss_avg	Exec. Time (s)
FEMNIST								
Gauss Mecanism	1	5.803	0.240	1.617	23.641.665	17.884	9.893.557	196.362
Gauss Mecanism	5	8.705	0.911	4.590	50.716	4.048	25.365	259.813
Gauss Mecanism	10	18.297	1.894	11.474	6.601	3.893	5.055	207.204
Gauss Mecanism	20	33.861	3.141	21.785	4.009	2.996	3.465	233.146
Gauss Mecanism	30	39.568	3.141	26.106	4.006	2.747	3.348	245.226
No Mecanism	{+∞}	49.712	3.141	32.000	3.994	2.208	3.135	127.815
Laplace Mecanism	1	4.964	0.216	1.599	1.530.267	4.243	599.304	198.157
Laplace Mecanism	5	5.612	0.480	2.402	11.252	4.052	6.693	260.29
Laplace Mecanism	10	5.851	0.480	2.955	4.457	4.062	4.203	213.323
Laplace Mecanism	20	6.715	0.767	3.940	4.105	3.981	4.035	257.964
Laplace Mecanism	30	5.204	0.743	3.842	4.105	3.942	4.021	255.649

Tabela 4.2: Resultados do processo descritos em 3.4 para o dataset FEMNIST. Fonte: Autor.

A partir do resultados apresentados na Tabela 4.2 podemos observar as seguintes tendências:

- Mecanismo de Gauss: A acurácia máxima obtida para o mecanismo de Gauss foi de 39,568%, enquanto para o mecanismo de Laplace foi de 6,715%. Essa diferença significativa indica que o mecanismo de Gauss, em geral até o momento, preserva melhor a acurácia do modelo, mesmo com a adição de ruído.

- Mecanismo de Laplace: A acurácia do mecanismo de Laplace também aumenta com o aumento de epsilon, mas em uma proporção muito menor comparado ao mecanismo de Gauss. A perda, assim como no caso do mecanismo de Gauss, diminui com o aumento de epsilon. O Mecanismo Laplace também possui uma média de acurácia menor que a Gaussiana para todos os valores ε testados.

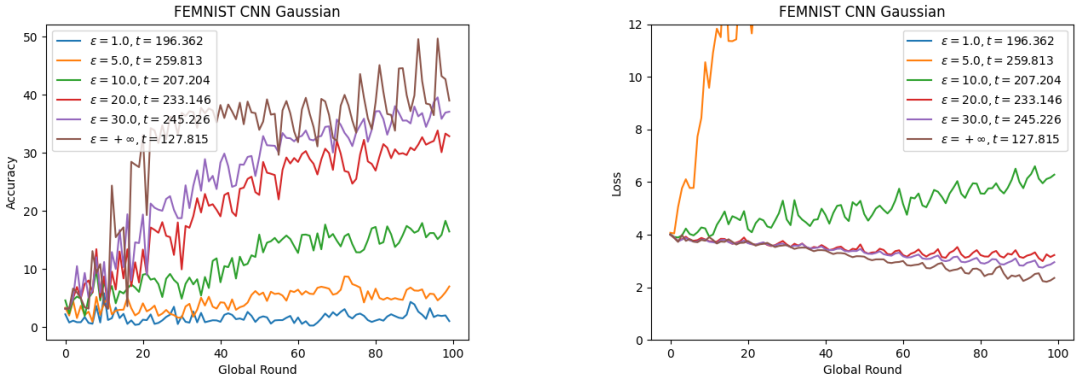

Figura 4.3: Acurácia e perda GaussDP no dataset FEMNIST. Fonte: Autor.

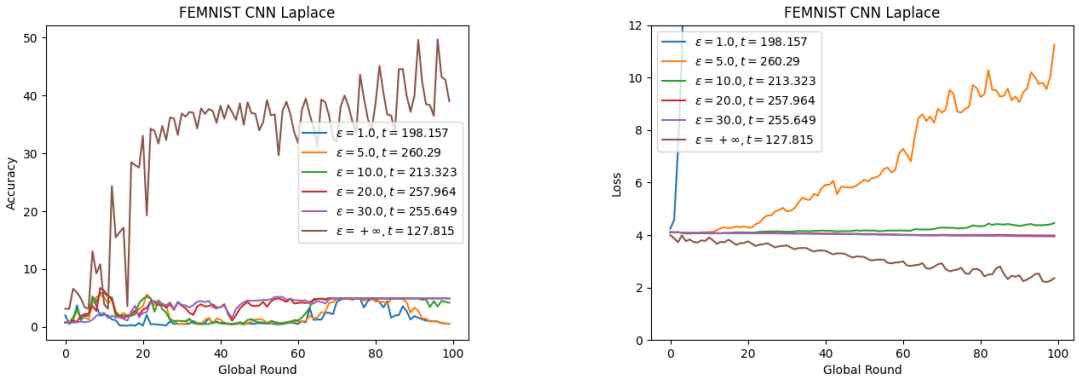

Figura 4.4: Acurácia e perda LaplaceDP no dataset FEMNIST. Fonte: Autor.

As Figuras 4.3 e 4.4 apresentadas exibem a evolução da acurácia durante as rodadas globais de treinamento no dataset FEMNIST, utilizando o mecanismo Gaussiano e Laplaciano de Differential Privacy (DP) com diferentes valores de ε. Cada curva representa o desempenho do modelo com um valor específico de ε, incluindo um caso sem aplicação de privacidade $\varepsilon = +\infty$, que serve como referência para a acurácia ideal sem ruído.

É possível notar de antemão o declínio do desempenho do modelo em relação ao MNIST e o Fashion-MNIST. Ao contrário do MNIST (10 classes) e Fashion-MNIST (10 classes de roupas e acessórios), o FEMNIST possui 62 classes (26 letras maiúsculas, 26 minúsculas e 10 dígitos). Essa alta cardinalidade torna a tarefa de classificação mais desafiadora, exigindo que o modelo aprenda a distinguir entre muitas classes com padrões visuais mais sutis. Aumenta-se também a probabilidade de classes desbalanceadas, devido à sua natureza não-iid, o que pode afetar o treinamento e o desempenho geral do modelo.

- Impacto do ε
  - Para valores menores de ε (e.g., $\varepsilon = 1.0$, Assim como nos testes anteriores, observa-se uma acurácia significativamente mais baixa ao longo de todas as rodadas, devido ao forte ruído introduzido pelos mecanismos Gaussiano e Laplaciano, o desempenho do modelo para a natureza desse dataset.

  - À medida que ε aumenta, o modelo apresenta melhorias progressivas na acurácia. Apesar de não alcançar uma acurácia tão alta quanto no dataset MNIST. A maior acurácia obtida com ruído foi ($\sim$ 39%) para o Gaussiano e ($\sim$ 18%) para o Laplaciano.
  - O tempo total de execução para cada valor de ε é exibido na legenda do gráfico. Observa-se que, para valores menores de ε, há um decréscimo no tempo de execução de ambos os mecanismos. Acredito que o otimizador possa ter desempenhado melhor nesse dataset em relação aos outros.

- Trade-off Privacidade-Desempenho: Valores intermediários de ε, como 10.0 ou 20.0, apresentam um compromisso razoável entre privacidade e desempenho do modelo. Similar ao encontrado no dataset MNIST.

4.3 Fashion-MNIST

Nesta seção, são apresentados os resultados da análise sobre o desempenho dos mecanismos de ruído da Differential Privacy (DP) na tarefa de classificação de peças de moda e acessórios do dataset Fashion-MNIST. Foram conduzidos experimentos com os mecanismos Laplaciano e Gaussiano, com o objetivo de avaliar o impacto da variação do parâmetro ε nos principais aspectos do treinamento, incluindo a acurácia, a função de perda e o tempo de execução do modelo.

Foram construídos gráficos comparativos que ilustram a evolução das métricas de desempenho para cada valor de epsilon em função do número de épocas de treinamento para cada mecanismo. Além disso, a Tabela 4.3 apresenta os resultados numéricos obtidos para cada combinação de mecanismo e valor de epsilon.

Dataset	Epsilon	Acurácia_max	Acurácia_min	Acurácia_avg	Loss_max	Loss_min	Loss_avg	Exec. Time (s)
Fashion-MNIST								
Gauss Mecanism	1	17.080	5.320	11.806	166.553	2.399	80.878	279.432
Gauss Mecanism	5	59.070	7.220	45.631	2.298	1.212	1.648	276.692
Gauss Mecanism	10	65.730	7.750	49.645	2.300	0.954	1.509	229.382
Gauss Mecanism	20	66.700	8.450	50.187	2.301	0.924	1.517	279.553
Gauss Mecanism	30	67.160	8.660	50.286	2.302	0.916	1.520	271.82
No Mecanism	{+∞}	66.010	8.860	49.652	2.303	0.899	1.534	302.855
Laplace Mecanism	1	20.060	5.930	11.848	5.469	2.303	3.567	278.651
Laplace Mecanism	5	16.510	8.830	10.665	2.308	2.281	2.296	229.833
Laplace Mecanism	10	11.000	8.706	9.758	2.306	2.284	2.297	232.772
Laplace Mecanism	20	9.950	8.550	9.258	2.307	2.287	2.298	278.829
Laplace Mecanism	30	9.640	8.540	9.083	2.306	2.288	2.297	227.727

Tabela 4.3: Resultados do processo descritos em 3.4 para o dataset Fashion-MNIST. Fonte: Autor.

A tabela 4.3 apresenta os resultados de um experimento realizado no dataset Fashion-MNIST, utilizando diferentes mecanismos de ruído (Gauss e Laplace) e variando valores de epsilon, o fator de privacidade da Differential Privacy (DP). O objetivo principal desse experimento é avaliar o impacto na acurácia e no tempo de execução do modelo.

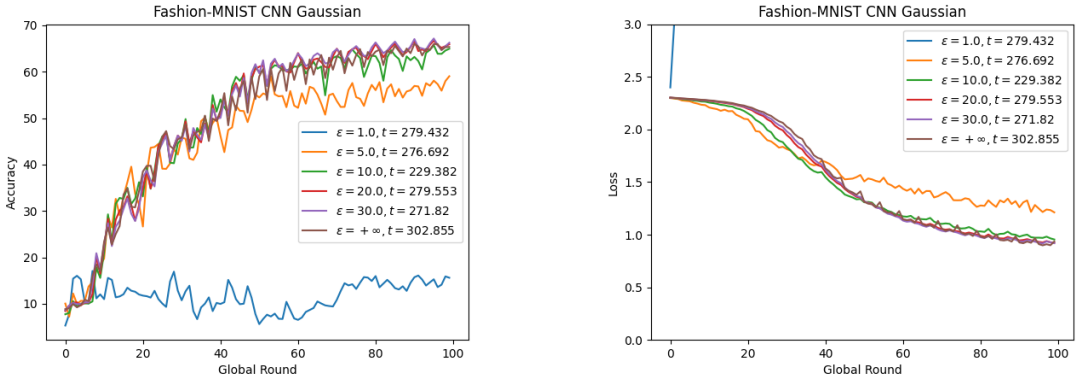

Figura 4.5: Acurácia e perda GaussDP no dataset Fashion-MNIST. Fonte: Autor.

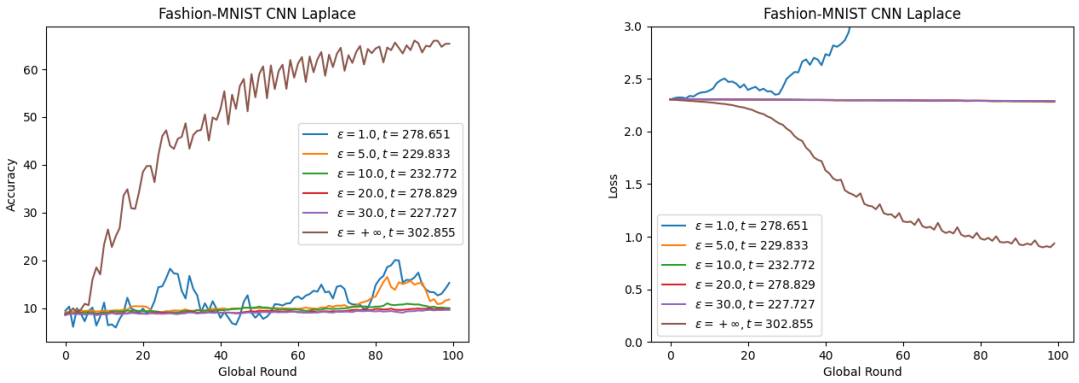

Figura 4.6: Acurácia e perda LaplaceDP no dataset Fashion-MNIST. Fonte: Autor.

As Figuras 4.5 e 4.6 exibem a evolução da acurácia durante as rodadas globais de treinamento no dataset Fashion-MNIST, utilizando o mecanismo Gaussiano e Laplaciano com diferentes valores de ε. Cada curva representa o desempenho do modelo com um valor específico de ε, incluindo um caso sem aplicação de privacidade $\varepsilon = +\infty$, que serve como referência para a acurácia ideal sem ruído.

Primeiramente, observa-se que o modelo apresentou um desempenho notavelmente eficaz ao utilizar o mecanismo Gaussiano de Differential Privacy. A acurácia das curvas geradas com a aplicação de ruído em diferentes valores de ε manteve-se bastante próxima da acurácia ideal (sem a introdução de ruído), com um declínio perceptível apenas para valores de ε inferiores a 5.0. Esse comportamento sugere que o mecanismo Gaussiano foi capaz de preservar adequadamente o desempenho do modelo, mesmo com a adição de ruído, e o parâmetro ε estivesse em níveis mais baixos.

Por outro lado, no caso do mecanismo Laplaciano, o desempenho foi substancialmente inferior, com a acurácia permanecendo em torno de aproximadamente $\sim$10% para todos os valores de ε testados. Esse resultado indica que, ao contrário do mecanismo Gaussiano, o mecanismo Laplaciano teve um impacto negativo significativo no desempenho geral do modelo.

Os experimentos exploraram o impacto dos valores de ε e dos mecanismos de ruído

(Gaussiano e Laplaciano) em diferentes datasets: MNIST, Fashion-MNIST e FEMNIST. Cada configuração foi avaliada em termos de acurácia, perda e o trade-off privacidade-desempenho, resultando em recomendações práticas para aplicações reais. A tabela 4.4 deve destacar as faixas de valores que atingem um bom compromisso entre privacidade e desempenho.

Mecanismo	**MNIST**	**Fashion-MNIST**	**FEMNIST**
Gaussiano	10 - 20	≥ 5	15 - 30
Laplaciano	15 - 30	≥ 1	20 - 40

Tabela 4.4: Faixas indicadas de ε para diferentes combinações de dataset e método. Fonte: Autor.

- MNIST: Os resultados mostram que o mecanismo Gaussiano se ajusta bem ao dataset MNIST com faixa de ε: 10 - 20., apresentando acurácia consistente e menor perda em valores intermediários de ε.

- Fashion-MNIST: Para este dataset, o mecanismo Laplaciano apresentou maior estabilidade, mantendo desempenho competitivo mesmo com ruídos mais intensos. No entanto, o Gaussiano performou significativamente melhor no quesito do desempenho do modelo mesmo com ruídos maiores. Obedecendo uma faixa de ε: ≥ 5.

- FEMNIST: O dataset FEMNIST é mais complexo e apresenta distribuição não-IID, dificultando a convergência. O mecanismo Gaussiano demonstrou ser mais eficaz nesse cenário, obtendo maior acurácia e menor perda para valores moderados a altos de ε. Obedecendo uma faixa entre 15 - 30.

Valores indicados de ε para cada cenário:

- Valores baixos de ε (< 10) devem ser evitados para aplicações onde a acurácia do modelo é crítica, pois introduzem ruído intenso que prejudica a utilidade do modelo.

- Valores intermediários de ε (10 - 20) oferecem um bom compromisso entre privacidade e desempenho, especialmente para datasets simples como MNIST.

- Valores altos de ε (> 20) são recomendados apenas quando a prioridade é maximizar a acurácia e a proteção da privacidade é secundária.

Conclusão

Este trabalho investigou a integração de Federated Learning (FL) com mecanismos de Differential Privacy (DP), com o objetivo de preservar a privacidade de dados sensíveis durante o treinamento de modelos de aprendizado de máquina. A pesquisa enfatiza a aplicação prática de mecanismos de ruído, como os modelos Laplaciano e Gaussiano, analisando seu impacto na acurácia e na perda de modelos treinados em cenários reais de FL, utilizando datasets variados.

Os resultados obtidos demonstraram que a integração entre FL e DP representa uma abordagem promissora para alcançar um equilíbrio entre privacidade e desempenho em modelos de aprendizado distribuído. Os experimentos realizados indicaram que o Mecanismo Gaussiano, de maneira geral, preserva melhor a acurácia dos modelos em comparação ao Mecanismo Laplaciano, mesmo em condições de forte proteção de privacidade.

Embora os resultados sejam encorajadores, desafios significativos permanecem no campo do aprendizado federado com privacidade diferencial. Entre eles, destaca-se a dificuldade em treinar modelos globais eficientes em ambientes com dados não IID, como no caso do dataset FEMNIST. A natureza heterogênea dos dados, característica intrínseca de muitos cenários reais, mostrou-se um obstáculo para alcançar níveis uniformes de desempenho entre os dispositivos participantes.

O estudo também revelou que o trade-off entre privacidade e desempenho é altamente dependente do contexto de aplicação. Valores intermediários de ϵ foram identificados como mais adequados para alcançar um compromisso viável, assegurando níveis razoáveis de proteção de privacidade sem comprometer significativamente a utilidade dos modelos. Isso é particularmente relevante em áreas sensíveis como saúde e sistemas automotivos, onde a precisão e a segurança dos modelos são fatores cruciais.

Para trabalhos futuros, sugere-se explorar técnicas adicionais de otimização e agregação em FL, bem como investigar outros algoritmos de privacidade como o Secure Aggregation (Bonawitz et al., 2017), Homomorphic Encryption (HE) (Hesamifard et al., 2018), Secure Multi-Party Computation (SMPC) (Mohassel and Zhang, 2017), e estratégias para mitigar os efeitos de dados não IID no desempenho global do modelo. Além disso, a aplicação de abordagens híbridas, combinando FL e DP com métodos de aprendizado ativo ou aprendizado transferido, pode oferecer caminhos promissores para aprimorar a robustez e a eficiência desses sistemas.

Em síntese, esta pesquisa contribuiu para a compreensão e o avanço das tecnologias voltadas para a privacidade e eficiência no treinamento de modelos distribuídos, oferecendo uma base sólida para desenvolvimentos futuros nesse campo emergente.

Referências Bibliográficas

Abadi, M., Chu, A., Goodfellow, I., McMahan, H. B., Mironov, I., Talwar, K., and Zhang, L. (2016). Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318.

Bonawitz, K., Ivanov, V., Kreuter, B., Marcedone, A., McMahan, H. B., Patel, S., Ramage, D., Segal, A., and Seth, K. (2017). Practical secure aggregation for privacy-preserving machine learning. In *proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1175–1191.

Caldas, S., Duddu, S. M. K., Wu, P., Li, T., Konečnỳ, J., McMahan, H. B., Smith, V., and Talwalkar, A. (2018). Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*.

Dwork, C. (2006). Differential privacy. In *International colloquium on automata, languages, and programming*, pages 1–12. Springer.

Dwork, C., McSherry, F., Nissim, K., and Smith, A. (2006). Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography: Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006. Proceedings 3*, pages 265–284. Springer.

Dwork, C., Roth, A., et al. (2014). The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 9(3–4):211–407.

Goodfellow, I., Bengio, Y., and Courville, A. (2017). *Deep learning*, volume 1. MIT press Cambridge, MA, USA.

Gu, J., Wang, Z., Kuen, J., Ma, L., Shahroudy, A., Shuai, B., Liu, T., Wang, X., Wang, G., Cai, J., and Chen, T. (2015). Recent advances in convolutional neural networks. *ArXiv*, abs/1512.07108.

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

Hesamifard, E., Takabi, H., Ghasemi, M., and Wright, R. N. (2018). Privacy-preserving machine learning as a service. *Proceedings on Privacy Enhancing Technologies*.

Kadam, S. S., Adamuthe, A. C., and Patil, A. B. (2020). Cnn model for image classification on mnist and fashion-mnist dataset. *Journal of scientific research*, 64(2):374–384.

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.

Li, Z., Liu, F., Yang, W., Peng, S., and Zhou, J. (2021). A survey of convolutional neural networks: analysis, applications, and prospects. *IEEE transactions on neural networks and learning systems*, 33(12):6999–7019.

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.

McMahan, H. B., Moore, E., Ramage, D., and y Arcas, B. A. (2016). Federated learning of deep networks using model averaging. *arXiv preprint arXiv:1602.05629*, 2(2).

Mohassel, P. and Zhang, Y. (2017). Secureml: A system for scalable privacy-preserving machine learning. In *2017 IEEE symposium on security and privacy (SP)*, pages 19–38. IEEE.

Nair, V. and Hinton, G. E. (2010). Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814.

Shi, W., Cao, J., Zhang, Q., Li, Y., and Xu, L. (2016). Edge computing: Vision and challenges. *IEEE internet of things journal*, 3(5):637–646.

Springenberg, J. T., Dosovitskiy, A., Brox, T., and Riedmiller, M. (2014). Striving for simplicity: The all convolutional net. *arXiv preprint arXiv:1412.6806*.

Voigt, P. and Von dem Bussche, A. (2017). The eu general data protection regulation (gdpr). *A Practical Guide, 1st Ed., Cham: Springer International Publishing*, 10(3152676):10–5555.

Zhang, C., Patras, P., and Haddadi, H. (2019). Deep learning in mobile and wireless networking: A survey. *IEEE Communications surveys & tutorials*, 21(3):2224–2287.