Algoritmo para Detecção e Mitigação de Ataques de Inversão de Rótulos no Aprendizado Federado

João Pedro Camargo Batista[1], **Rodolfo S. Villaça**[1]

[1]Departamento de Informática (DI)
Universidade Federal do Espírito Santo (Ufes)

joao.c.batista@edu.ufes.br
rodolfo.villaca@ufes.br

Abstract. *Federated Learning (FL) enables distributed machine learning training while preserving data privacy. However, attacks such as label-flipping can compromise the performance of the global model. This article aims to evaluate the impact of such attacks in scenarios involving Differential Privacy (DP) and model compression based on FedSketch, as well as to develop and assess an algorithm for detecting malicious clients to improve the robustness of the training process.*

Resumo. *O Aprendizado Federado (FL) permite treinar modelos de aprendizado de máquina de forma distribuída, preservando a privacidade dos dados. No entanto, ataques como o label-flipping podem comprometer o desempenho do modelo global. Este artigo tem como objetivo principal avaliar os impactos desse ataque em cenários com privacidade diferencial (DP) e compactação de modelos baseada em FedSketch, além de desenvolver e avaliar um método de detecção de clientes mal-intencionados para aumentar a robustez do treinamento.*

1. Introdução

O aprendizado de máquina tem se expandido para diversas áreas, incluindo cidades, indústrias e governos, impulsionado pela disponibilidade crescente de dados para treinamento de modelos de inteligência artificial. Esse avanço, no entanto, exige cada vez mais recursos computacionais, e o aprendizado distribuído surge como solução para o melhor aproveitamento desses recursos. Entretanto, essa abordagem apresenta desafios quanto à transmissão entre os dispositivos distribuídos de dados sensíveis e confidenciais, expondo-os a violações, o que se torna especialmente grave em domínios em que esses dados possibilitam a descrição pessoal, de hábitos ou localização de pessoas. Isso motivou o desenvolvimento de um conceito denominado Aprendizado Federado (do inglês, *Federated Learning*, FL), em que os dados de treinamento proprietários permanecem restritos aos dispositivos dos clientes, com apenas os parâmetros dos modelos locais compartilhados com um servidor central para agregação, sem a necessidade de expor os dados brutos, o que garante maior privacidade [McMahan et al. 2017].

Apesar dos avanços em privacidade, o treinamento no FL continua vulnerável à ação de agentes maliciosos, tais como clientes maliciosos, que visam atrapalhar a convergência e podem comprometer a integridade do modelo global do sistema [Mammen 2021]. Um ataque a ser observado é o ataque de inversão de rótulos (do

inglês, *label-flipping*), em que os clientes maliciosos adulteram os rótulos dos dados de treinamento, prejudicando a acurácia e a capacidade de generalização do modelo global [Shen et al. 2024].

Para detectar esse tipo de ataque, que prejudica e impede a convergência do modelo global, abordagens tradicionais promovem uma análise dos parâmetros dos modelos locais. Porém, tais procedimentos podem colocar em risco a privacidade dos dados usados nos modelos locais, uma vez que os agentes maliciosos podem se aproveitar da observação direta dos parâmetros de modelo para aplicar outros tipos de ataque, como os ataques de inversão de gradiente ([Zhao et al. 2020] e [Geiping et al. 2020]) e os ataques de inferência ([Shokri et al. 2017]). Nesse sentido, faz-se necessária a adição de uma camada extra de proteção por parte dos clientes, ao utilizar técnicas de privacidade no compartilhamento dos parâmetros dos modelos locais.

Uma dessas técnicas é o *FedSketch*, que combina estruturas de dados probabilísticas (nesse caso, os *count-min sketches*) com privacidade diferencial (do inglês, *differential privacy*, DP) para compactar os modelos locais antes do envio para o servidor, sem comprometer a privacidade [Sarmento et al. 2024]. Porém, como essa técnica está baseada em estruturas probabilísticas, não é possível aplicar as metodologias tradicionais de detecção de ataques de *label-flipping*, sendo necessário o desenvolvimento de outras metodologias que possam contribuir com o FL nesse sentido.

Assim, os objetivos desse trabalho baseiam-se na análise do impacto dos ataques de *label-flipping* no treinamento federado, na investigação do uso de políticas de seleção de clientes para a sua mitigação e no desenvolvimento de um novo algoritmo que visa a detecção dos agentes maliciosos nos cenários de compactação e privacidade de modelos locais, de modo a incrementar ainda mais a segurança do processo de treinamento federado.

O restante deste artigo está organizado da seguinte forma: a Seção 2 apresenta a fundamentação teórica; a Seção 3 descreve a metodologia adotada; a Seção 4 discute os experimentos e resultados; e a Seção 5 apresenta as conclusões.

2. Fundamentação Teórica

Esta seção tem como objetivo descrever mais detalhadamente os diferentes conceitos teóricos necessários para o desenvolvimento do trabalho, como o FL, seu funcionamento, a técnica de compactação e agregação de modelos *FedSketch* e o ataque de *label-flipping*.

2.1. Aprendizado Federado

Com o advento dos conjuntos de dados cada vez maiores e mais complexos, surgiram abordagens distribuídas para o treinamento de modelos, de modo a aproveitar mais eficientemente os recursos computacionais para o processamento desses conjuntos. Entretanto, esse paradigma distribuído encontrou um gargalo na transmissão dos conjuntos de dados entre os dispositivos de borda da rede e o dispositivo central de treinamento [McMahan et al. 2017]. Além disso, o deslocamento dos dados entre dispositivos aumenta os desafios de segurança e privacidade, especialmente quando o contexto envolve a troca de informações confidenciais ou sensíveis [Mammen 2021]. É nesse contexto que surgem adaptações para o treinamento distribuído voltadas para a

segurança dos dados, sendo uma das mais conhecidas a denominada Aprendizado Federado [McMahan et al. 2017].

O conceito de FL pode ser definido como uma técnica de aprendizagem colaborativa entre dispositivos de rede ou organizações, em que os parâmetros (pesos ou gradientes) dos modelos de aprendizado de máquina, essencialmente os modelos de aprendizado profundo, são compartilhados e agregados, em vez do tradicional compartilhamento de dados [McMahan et al. 2017]. Assim, o FL vale-se do poder computacional dos diferentes dispositivos da rede para treinar modelos globais sem a partilha externa de dados, alcançando melhorias no processo quanto à eficiência de comunicação e à segurança dos dados envolvidos.

Dentro do processo de FL, cada dispositivo possui um papel específico [Mammen 2021]. Dispositivos de borda que realizam o treinamento de modelos locais com dados proprietários e produzem os parâmetros que serão compartilhados são chamados clientes. As entidades que recebem e agregam os parâmetros nos modelos globais são chamadas de servidores. É importante ressaltar que, embora a arquitetura clássica do Aprendizado Federado utilize apenas um servidor central, é possível integrar diferentes servidores regionais para agilizar o processo de convergência do modelo global.

Além disso, o FL funciona por meio de rodadas de comunicação cíclicas, que são repetidas por um número finito e alto de vezes ou até que o modelo global atinja a convergência. Essas rodadas possuem primariamente quatro passos:

1. **Seleção de clientes:** o servidor central seleciona, dentre um conjunto de clientes, os que participarão da etapa de treinamento, de forma aleatória ou baseado em uma política de seleção de clientes;
2. **Compartilhamento do modelo global:** o servidor central envia aos clientes selecionados na primeira etapa os parâmetros do modelo global;
3. **Treinamento local de modelo:** os clientes irão, paralelamente, treinar localmente o modelo fornecido pelo servidor com os seus próprios dados, sem compartilhamento desses dados com o servidor central;
4. **Agregação de modelos:** os clientes enviarão os novos parâmetros de seus modelos treinados para o servidor central, que, por sua vez, irá agregá-los ao modelo global por meio de uma função de agregação, que pode ter diversas abordagens.

Com isso, o FL mostrou-se uma técnica robusta para o treinamento de modelos que se aproveita do poder computacional dos dispositivos de borda para gerar um modelo global abrangente, evitando o compartilhamento de dados sensíveis. Assim, esse esquema experimentou rápida popularização, com aplicação em contextos que necessitam de garantias de privacidade, como a área financeira e a área da saúde [Korkmaz et al. 2022].

2.2. O Ataque de *Label-Flipping*

Entretanto, apesar dos avanços nas garantias de privacidade dos dados, o processo federado ainda está suscetível a ataques. Esses ataques são realizados por agentes internos ao treinamento, como clientes maliciosos ou servidores honestos-mas-curiosos (quando não agem com a intenção de prejudicar o treinamento, mas observam os dados dos clientes de forma indevida). Um tipo notável de ataque é o ataque de *label-flipping*, que é um exemplo de ataque de envenenamento de modelos em que os clientes maliciosos invertem

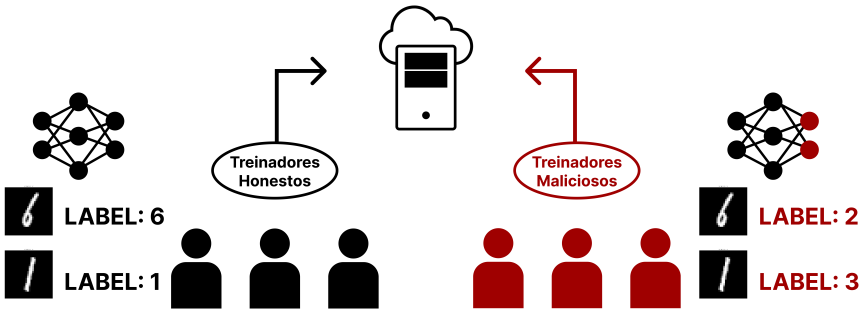


Figura 1. Exemplo de ataque de inversão de rótulos utilizando-se o *dataset* MNIST, em que os *labels* das Classes "6" e "1" foram alterados para as Classes "2" e "3", respectivamente.

os rótulos de uma determinada classe para outra classe [Jebreel et al. 2024]. O conceito desse tipo de ataque é envenenar o modelo global (e consequentemente os modelos locais pelo envio de parâmetros envenenados pelo servidor) a partir da introdução de erros no modelo local. Assim, quando os clientes recebem os parâmetros envenenados do servidor, os parâmetros locais são fortemente afetados, o que diminui posteriormente a acurácia do modelo global e prejudica o processo de convergência [Elmahfoud et al. 2025].

Um exemplo desse ataque pode ser construído utilizando-se o *dataset* MNIST[1], que contém diversas imagens de dígitos manuscritos e rotulados conforme o valor que representam. Assim, fazendo-se uso desse dataset, os clientes maliciosos invertem os rótulos que deveriam "0" para o valor "1", por exemplo, envenenando os modelos globais conforme processo supracitado. A Figura 1 ilustra esse procedimento, explicitando o envio de informações envenenadas para o servidor. Em ambientes de produção, com conjuntos de dados reais, esse ataque é viável e facilmente reprodutível pois não exige muito poder computacional e não pode ser detectado de forma trivial pelo servidor, que não possui conhecimento dos dados locais do cliente atacante [Jebreel et al. 2024].

2.3. O *FedSketch*

Outro conceito importante para este projeto é o *FedSketch*, que consiste em uma técnica que visa proteger a privacidade dos dados no FL, combinando *sketches* (estruturas de dados probabilísticas) para compactar os modelos locais dos cliente e adicionar DP, aumentando a sua segurança [Sarmento et al. 2024]. Os *sketches* permitem compactar os parâmetros dos modelos antes do envio ao servidor, reduzindo a carga de comunicação e o risco de inferência de dados a partir das informações compartilhadas.

No *FedSketch* os modelos são compactados utilizando-se *count-min sketches*, estruturas de dados probabilísticas que utilizam tabelas *hash* independentes para comprimir vetores de alta dimensionalidade. Os *sketches* são então transmitidos ao servidor que, por sua vez, agrega os modelos locais (sem descompactar) em um modelo global e os envia de volta aos clientes. Tal agregação sem necessidade de descompactação dos *sketches* só é possível devido à natureza homomórfica da funções de *hash* utilizadas na construção da estrutura [Sarmento et al. 2024].

[1] `https://www.kaggle.com/datasets/hojjatk/mnist-dataset`

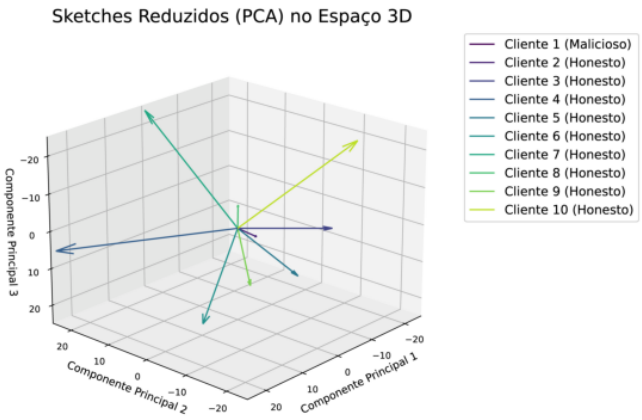


Figura 2. Distribuição no espaço tridimensional dos vetores reduzidos (PCA) gerados pelos *count-min sketches* dos clientes em uma rodada do FL.

No *FedSketch* os clientes descompactam, avaliam a utilidade do modelo global e retornam o resultado desta avaliação ao servidor de agregação por meio de métricas estabelecidas. O *FedSketch* pode reduzir o tamanho das informações transmitidas em até 73 vezes, mantendo uma precisão similar ao FL convencional, enquanto pode alcançar um alto nível de DP, com $\epsilon \approx 10^{-}6$, o que significa que os dados individuais dos clientes estão fortemente protegidos [Dwork 2006].

Entretanto, apesar de garantir uma camada extra de proteção aos dados e modelos locais dos clientes, é conferida uma aleatoriedade espacial aos vetores dos *sketches*, uma vez que são construídos com funções *hash*. Assim, métodos tradicionais de detecção de ataques de *label-flipping*, que observam diretamente os parâmetros dos modelos enviados [Jebreel et al. 2024], não podem ser usados nesse contexto, devido à aleatoriedade supracitada. Assim, novos métodos devem ser desenvolvidos para garantir a detecção de clientes atacantes mesmo em cenários de compactação e privacidade dos modelos.

3. Metodologia

Em um primeiro momento, partiu-se para uma análise preliminar do impacto do ataque de *label-flipping* sobre o processo de treinamento federado. Para isso, a metodologia utilizada para o desenvolvimento desse estudo foi baseada em experimentos conduzidos com o MininetFed[2], uma ferramenta de emulação para FL, com avaliação do ataque e de políticas de seleção de clientes usando o dataset MNIST. Nessa etapa, foram realizados dois experimentos: o primeiro consistiu na avaliação da acurácia do modelo global com e sem a presença de clientes maliciosos, a fim de verificar seu impacto; e o segundo consistiu na avaliação da eficácia da aplicação de diferentes políticas de detecção de clientes maliciosos a partir da seleção de clientes.

No primeiro experimento, seis clientes participaram do treinamento, sendo todos honestos em um primeiro momento e um deles sendo malicioso em um segundo momento. Além disso, a política de seleção de clientes adotada pelo servidor foi aleatória, com seleção de quatro clientes por rodada de treinamento, enquanto a política de agregação de

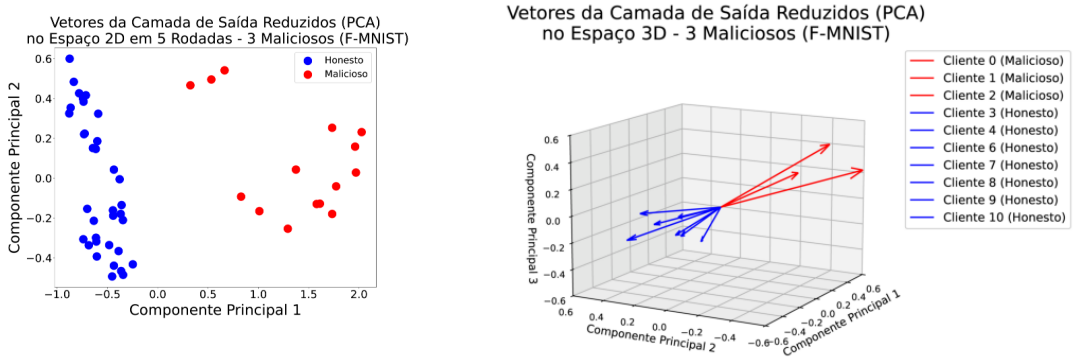
[2]`https://github.com/lprm-ufes/MininetFed`

modelos utilizada foi a *FedAvg*, que é a técnica clássica e original do FL e que promove a média ponderada dos parâmetros dos modelos dos clientes.

No segundo experimento, quatro clientes participaram do treinamento, sendo três honestos e um malicioso (de forma a dar ênfase na proporção de clientes maliciosos). Ademais, a política de agregação de modelos utilizada foi a mesma do primeiro experimento. Já a política de seleção de clientes envolveu diferentes abordagens que tentaram identificar o cliente malicioso, sendo elas: *all* (todos os clientes selecionados para uma rodada), *random* (três clientes selecionados de forma aleatória), *Deev* (proposto por [Souza et al. 2023], seleciona os k clientes com acurácia do modelos local menor do que a acurácia média de todos os clientes, sendo que k decresce ao longo do tempo de acordo com uma função de decaimento), *FedSecPer* (funciona de forma semelhante à abordagem *Deev*, mas sem o decaimento de k) e *B-Trimmed Mean* (proposto por [Wang et al. 2025], seleciona os clientes que possuam a acurácia do modelo local maior do que a média truncada da acurácia do modelo global).

Os resultados, explícitos na Seção 4, demonstram que o ataque de *label-flipping* tem impacto real sobre a acurácia do modelo global e que as técnicas de detecção de clientes maliciosos a partir da seleção de clientes apresentam resultados com pouca relevância. Dado esse contexto, entendeu-se que era necessário averiguar outras técnicas de detecção de atacantes. Entretanto, as técnicas encontradas no estado-da-arte são fortemente baseadas na análise completa dos parâmetros dos modelos locais [Jebreel et al. 2024], o que não é possível em cenários de privacidade e compactação de modelos, quando se utiliza os *sketches* probabilísticos, como mencionado na Seção 2.3. Esse efeito pode ser verificado na Figura 2 a seguir, em que foi aplicada uma redução de dimensionalidade aos *sketches* enviados pelos clientes (honestos e maliciosos) em uma determinada rodada para verificar sua distribuição espacial em três dimensões. Nota-se que não é possível distinguir bem, a partir da análise vetorial, qual é o *sketch* correspondente ao cliente malicioso.

A partir disso, compreendeu-se que era necessário efetuar a proposição de um novo algoritmo de detecção que pudesse ser combinado com o uso dos *sketches* proba-



(a) Espaço bidimensional (b) Espaço tridimensional

Figura 3. Distribuição no espaço (a) bidimensional e (b) tridimensional dos vetores da última camada do modelo, enviados pelos clientes. Em (a) estão as contribuições de todos os clientes ao longo de cinco rodadas e em (b) estão as contribuições de todos os clientes em uma única rodada

Algoritmo 1 Algoritmo Proposto para Detecção do Ataque de *Label-Flipping* por meio da Última Camada Linear dos Modelos Locais

Require: $trainer\_list$: Lista de clientes, $activation\_vectors$: Lista de vetores de pesos da última camada linear da rede neural dos clientes
Ensure: Lista de clientes possivelmente maliciosos
$total\_distances \leftarrow []$
for all $trainer\_layer \in activation\_vectors$ **do**
  $euclidean\_distance \leftarrow 0$
  for all $trainer\_layer2 \in activation\_vectors$ **do**
    $euclidean\_distance \leftarrow euclidean\_distance + ||trainer\_layer - trainer\_layer2||$
  end for
  $total\_distances.append(euclidean\_distance)$
end for
$modified\_zscore \leftarrow []$
for all $distance \in total\_distances$ **do**
  $zscore \leftarrow CalculateModifiedZScore(distance)$
  $modified\_zscore.append(zscore)$
end for
$malicious\_trainers \leftarrow []$
for all $i, zscore \in modified\_zscore$ **do**
  if $zscore \geq 5 \vee zscore \leq -5$ **then**
    $malicious\_trainers.append(trainer\_list[i])$
  end if
end for
return $malicious\_trainers$

bilísticos e DP, sem que envolvesse a análise dos componenentes dos *sketches*. Para isso, foi proposto outro estudo, cuja metodologia também esteve baseada em experimentos conduzidos com o MininetFed, avaliando a técnica proposta com o uso dos conjuntos de dados MNIST e Fashion-MNIST[3].

Esse estudo fez uma análise do vetor de pesos da última camada linear da rede neural, enviado pelos clientes ao servidor central junto com os modelos compactados pelos *sketches* probabilísticos. A hipótese central é que vetores provenientes de clientes maliciosos apresentam padrões distintos em relação aos de clientes honestos, permitindo a detecção por meio de técnicas de análise de *outliers* [Jebreel et al. 2024]. Tal premissa pode ser verificada na Figura 3, que demonstra a aplicação de uma técnica de redução de dimensionalidade por meio de análise de componentes principais (PCA) aos vetores de pesos dos clientes, para explicitar a distância e a mudança de direção entre eles.

Para a detecção dos clientes maliciosos, foi desenvolvido um algoritmo que calcula a distância euclidiana entre os vetores de pesos da última camada linear do modelo de cada cliente e aplica uma métrica estatística, o *Z-Score* Modificado (capaz de lidar de forma eficiente com a presença de *outliers*), para identificar *outliers*, considerando maliciosos os clientes cujos vetores se desviam significativamente das métricas apresentadas pelo restante do grupo. O cálculo do *Z-Score* Modificado pode ser verificado na Equação 1 a seguir, em que Z_i é o *Z-Score* Modificado, $\tilde{X}$ é a mediana das distâncias dos vetores de peso e MAD é o desvio absoluto da mediana (cujo cálculo está explícito na Equação 2). O algoritmo completo proposto para essa tarefa de detecção está detalhado no Algoritmo 1.

[3]`https://www.kaggle.com/datasets/zalando-research/fashionmnist`

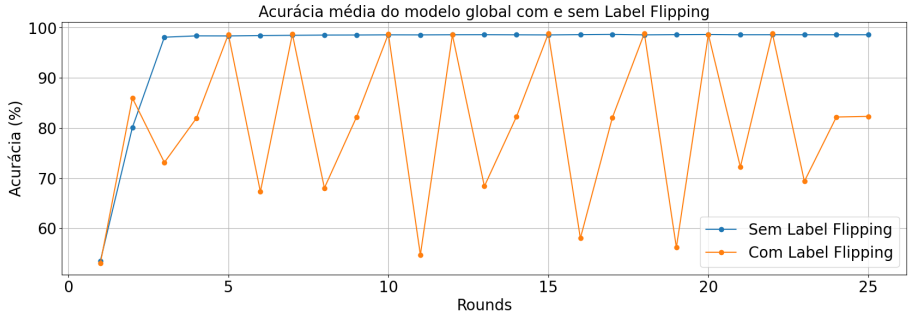


Figura 4. Resultados da acurácia do modelo global a cada rodada durante o treinamento federado com nenhum e um dos clientes sendo malicioso.

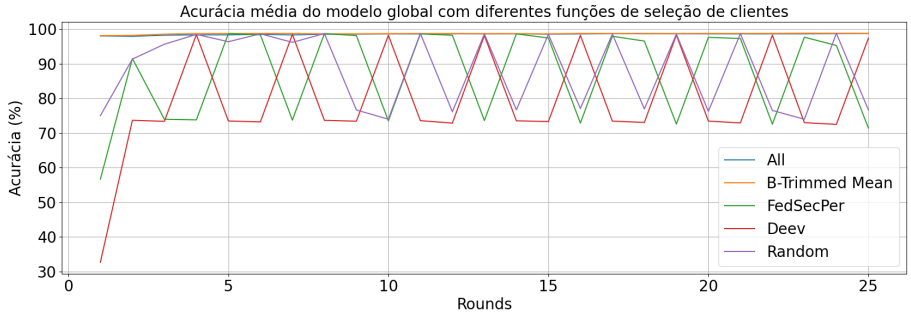


Figura 5. Resultados da acurácia do modelo global a cada rodada durante o treinamento federado com a aplicação de diferentes políticas de seleção de clientes para detecção de atacantes.

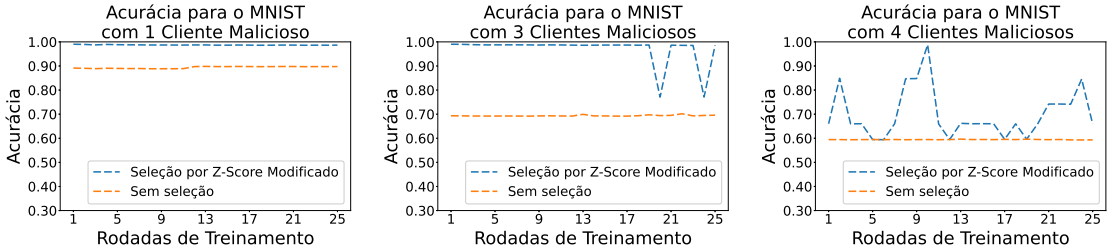
$$Z_i = \frac{X_i - \tilde{X}}{\text{MAD}} \tag{1}$$

$$\text{MAD} = \text{median}\left(\left|X_i - \tilde{X}\right|\right) \tag{2}$$

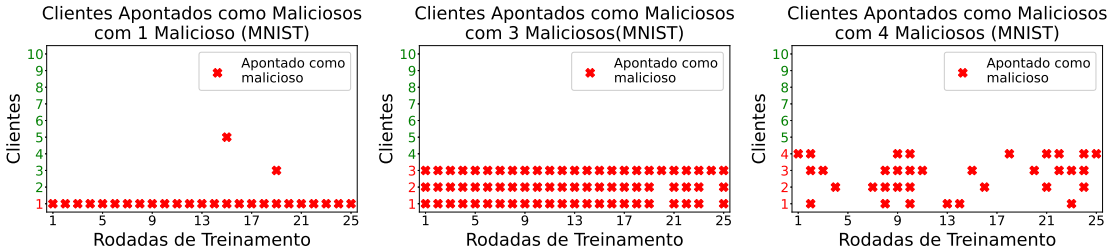
Para validar esse algoritmo proposto, fez-se outro experimento federado no MininetFed, desta vez com 10 clientes. A quantidade de clientes maliciosos foi variada ao longo do processo de treinamento, com proporções de 10%, 30% e 40%. A quantidade total de clientes foi escolhida de tal modo a dar ênfase na proporção de atacantes. A política de agregação empregada foi a *FedSketch* e a política de seleção de clientes adotada foi a *all*. Esses cenários foram avaliados com e sem o uso do algoritmo proposto. No caso de uso, o algoritmo foi implementado apenas no servidor, tendo este apenas excluído as contribuições dos clientes identificados como atacantes (mas sem notificá-los). Foram utilizados os *datasets* MNIST e Fashion-MNIST.

4. Experimentos e Resultados

Para demonstrar o real impacto dos clientes maliciosos ao processo federado, realizou-se dois experimentos iniciais, conforme descrito na seção anterior. A Figura 4 demonstra que, para o dataset MNIST, a acurácia do modelo global foi impactada de forma significativa na presença desses agentes, com oscilações constantes e impedimento da convergência do modelo global. Assim, enquanto no cenário honesto a acurácia estabilizou-



(a) Acurácia média do modelo global para diferentes proporções de clientes maliciosos.



(b) Clientes identificados como maliciosos pelo servidor a cada rodada do treinamento

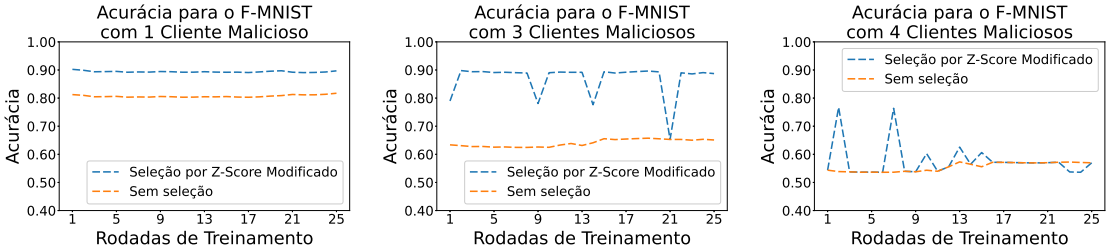
Figura 6. Resultados da aplicação do algoritmo proposto para detecção de atacantes em treinamento com MNIST. As colunas demonstram os resultados obtidos para 10%, 30% e 40% de clientes maliciosos, respectivamente.

se rapidamente em um patamar de cerca de 98%, no cenário adversarial não houve essa estabilização, com acurácias registradas até mesmo abaixo de 60%.

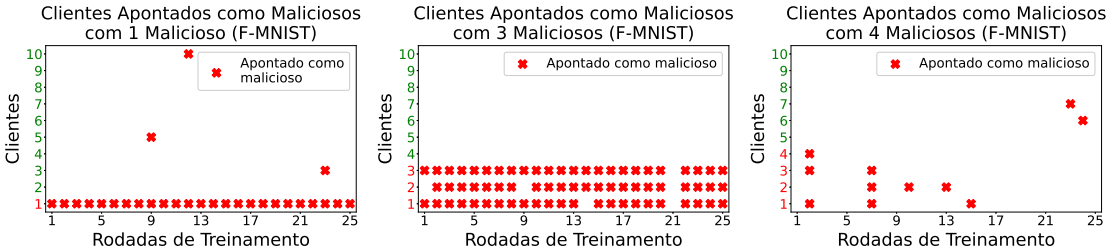
Já a Figura 5 demonstra a eficácia da aplicação de diferentes políticas de seleção de clientes (que objetivam a detecção de atacantes) sobre a acurácia do modelo global, conforme o segundo experimento descrito na seção anterior. Nota-se que, novamente, houve uma oscilação muito grande na acurácia e o impedimento da convergência do modelo global, mesmo com utilização de tais políticas, o que indica que possivelmente elas não possuem boa eficiência em todos os cenários, levantando a necessidade de utilização de outras técnicas. As únicas exceções foram as técnicas *All* e *B-Trimmed Mean*, o que é um indicativo de que, embora destacada pela proporção de clientes presentes, a proporção de clientes maliciosos não foi suficiente para afetar o desempenho do treinamento, não significando que essas técnicas são eficientes em todos os casos.

Já com o algoritmo proposto implementado, realizou-se os experimentos descritos no fim da Seção 3. Além de observar a acurácia do modelo global em todos os casos, verificou-se quais clientes foram apontados como maliciosos pelo algoritmo em cada rodada, de modo a visualizar a sua eficiência na detecção.

A Figura 6 demonstra os resultados obtidos para o dataset MNIST e a Figura 7 para o dataset Fashion-MNIST. Percebe-se que, no cenário sem a técnica proposta, a presença de clientes maliciosos reduziu significativamente a acurácia do modelo. Essa métrica caiu de 98% para menos de 90% com apenas 10% de clientes maliciosos. À medida que a proporção de clientes maliciosos aumentava, a acurácia deteriorava-se ainda mais, chegando a 60% quando 40% dos clientes realizavam o ataque. Na presença do algoritmo proposto, os clientes maliciosos foram quase sempre detectados para 10% e 30% de atacantes e acurácia preservada. Entretanto, alguns falsos positivos e falsos negativos



(a) Acurácia média do modelo global para diferentes proporções de clientes maliciosos.



(b) Clientes identificados como maliciosos pelo servidor a cada rodada do treinamento.

Figura 7. Resultados da aplicação do algoritmo proposto para detecção de atacantes em treinamento com Fashion-MNIST. As colunas demonstram os resultados obtidos para 10%, 30% e 40% de clientes maliciosos, respectivamente.

foram identificados. Para 40% dos clientes, o algoritmo não se comportou bem, uma vez que os clientes maliciosos deixam de ser outliers e passam a ser maioria no treinamento. A Figura 7 demonstra resultados semelhantes, obtidos para o dataset Fashion-MNIST, tendo a acurácia caído de 90% para 83% com 10% de atacantes e para abaixo de 60% para 40% de clientes maliciosos.

5. Conclusões

O presente artigo investigou o impacto do ataque de *label-flipping* em sistemas de FL e propôs um algoritmo de detecção baseado na análise da última camada linear dos modelos locais, combinável com *FedSketch* e DP — contexto em que técnicas tradicionais, dependentes dos parâmetros completos, não se aplicam. O algoritmo demonstrou resultados promissores para até 30% de clientes maliciosos, preservando a acurácia e a convergência do modelo global, mas apresentou queda de eficiência com 40% de atacantes, cenário em que os clientes maliciosos deixam de ser *outliers*.

As principais limitações residem na dependência estatística em proporções elevadas de atacantes e na ocorrência de falsos positivos e negativos. Como trabalhos futuros, pretende-se avaliar o algoritmo em ambientes maiores e mais heterogêneos, desenvolver adaptações para proporções atípicas de atacantes e explorar o uso de *blockchains* para mitigar problemas bizantinos de confiança entre clientes.

Agradecimentos

Este trabalho contou com financiamento parcial de: CNPq; Capes; Fapes (2024-9G1WP, 2023-RWXSZ, 2026-B74MN).

Artefatos

Os códigos produzidos para os experimentos deste trabalho podem ser encontrados em: `https://github.com/lprm-ufes/MininetFed-LabelFlipping`

Referências

Dwork, C. (2006). Differential privacy. In Bugliesi, M., Preneel, B., Sassone, V., and Wegener, I., editors, *Automata, Languages and Programming*, pages 1–12, Berlin, Heidelberg. Springer Berlin Heidelberg.

Elmahfoud, E., Hajla, S. E., Maleh, Y., Mounir, S., and Ouazzane, K. (2025). Label flipping attacks in hierarchical federated learning for intrusion detection in iot. *Information Security Journal: A Global Perspective*, 34(4):310–326.

Geiping, J., Bauermeister, H., Dröge, H., and Moeller, M. (2020). Inverting gradients - how easy is it to break privacy in federated learning? In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20, Red Hook, NY, USA. Curran Associates Inc.

Jebreel, N. M., Domingo-Ferrer, J., Sánchez, D., and Blanco-Justicia, A. (2024). Lfighter: Defending against the label-flipping attack in federated learning. *Neural Netw.*, 170(C):111–126.

Korkmaz, A., Alhonainy, A., and Rao, P. (2022). An Evaluation of Federated Learning Techniques for Secure and Privacy-Preserving Machine Learning on Medical Datasets . In *2022 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, pages 1–7, Los Alamitos, CA, USA. IEEE Computer Society.

Mammen, P. M. (2021). Federated learning: Opportunities and challenges.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and Arcas, B. A. y. (2017). Communication-Efficient Learning of Deep Networks from Decentralized Data. In Singh, A. and Zhu, J., editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR.

Sarmento, E., Mota, V., and Villaça, R. (2024). Privacidade e comunicação eficiente em aprendizado federado: Uma abordagem utilizando estruturas de dados probabilísticas e seleção de clientes. In *Anais do XLII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos*, pages 85–98, Porto Alegre, RS, Brasil. SBC.

Shen, X., Liu, Y., Li, F., and Li, C. (2024). Privacy-preserving federated learning against label-flipping attacks on non-iid data. *IEEE Internet of Things Journal*, 11(1):1241–1255.

Shokri, R., Stronati, M., Song, C., and Shmatikov, V. (2017). Membership Inference Attacks Against Machine Learning Models . In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18, Los Alamitos, CA, USA. IEEE Computer Society.

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.

Wang, T., Zheng, Z., and Lin, F. (2025). Federated learning framework based on trimmed mean aggregation rules. *Expert Systems with Applications*, 270:126354.

Zhao, B., Mopuri, K. R., and Bilen, H. (2020). idlg: Improved deep leakage from gradients.