



Seleção de Clientes Federados usando Aprendizado por Reforço Multiagente

Marcelo P. Zwetsch, Guilherme A. Thomaz e Miguel Elias M. Campista

¹GTA/PEE-COPPE/DEL-Poli – Universidade Federal do Rio de Janeiro (UFRJ)

{marcelo, guiaraujo, miguel}@gta.ufrj.br

Abstract. *In federated learning, the central server decides which clients should participate in each training round. However, traditional strategies that do not estimate each client's contribution may be vulnerable to low-quality data and malicious clients. This work investigates a client selection approach based on multi-agent reinforcement learning (MARL). The proposed architecture models each client as an agent, in which decisions are made in a decentralized and cooperative manner. Each agent evaluates characteristics such as data diversity, processing capacity, and participation history, learning to contribute to a more strategic selection of training participants in dynamic and non-IID scenarios. The experiments simulate different degrees of heterogeneity among clients, reflecting non-IID data distributions, and also consider scenarios with label-flipping attacks. The performance, compared with FedAvg and single-agent RL (SARL), shows improvements in the final model accuracy and in balancing client participation.*

Resumo. *No aprendizado federado, o servidor central decide quais clientes devem participar de cada rodada de treinamento. Entretanto, estratégias tradicionais que não estimam a contribuição de cada cliente podem ser vulneráveis a dados de baixa qualidade e a clientes maliciosos. Este trabalho investiga uma abordagem de seleção de clientes baseada no aprendizado por reforço multiagente (MARL). A arquitetura proposta modela cada cliente como um agente, no qual as decisões são tomadas de forma descentralizada e cooperativa. Cada agente avalia características como diversidade de dados, capacidade de processamento e histórico de participação, aprendendo a contribuir para uma seleção mais estratégica dos participantes do treinamento em cenários dinâmicos e não-IID. Os experimentos simulam diferentes graus de heterogeneidade entre os clientes, refletindo distribuições não-IID, além de considerar cenários com inversão de rótulos como estratégia de ataque ao aprendizado. O desempenho, em comparação ao FedAvg e ao single-agent RL (SARL), demonstra melhoria na acurácia final do modelo e no equilíbrio na participação dos clientes.*

1. Introdução

O uso crescente de dispositivos conectados, especialmente em IoT e computação de borda, leva à geração de grandes volumes de dados de forma distribuída. Em muitos

Este trabalho foi realizado com apoio da Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) – Código de Financiamento 001 e 88887.954253/2024-00, do CNPq (310234/2025-5, 407304/2025-8, 408255/2023-4 e 405940/2022-0), da FAPERJ (E-26/200.380/2023 e E-26/210.778/2025), da FAPESP (2023/00673-7 e 2023/00811-0) e da Fundação de Desenvolvimento da Pesquisa - Fundep - Rota 2030 em conjunto dos nossos parceiros Stellantis e Mobway.

cenários, restrições de privacidade e de comunicação dificultam a centralização dessas informações. Nesse contexto, o Aprendizado Federado (*Federated Learning* – FL) permite treinar modelos de aprendizado de máquina mantendo os dados nos clientes. O servidor coordena o processo em rodadas, selecionando um subconjunto de clientes para realizar o treinamento local e enviar atualizações do modelo para agregação, produzindo um novo modelo global a cada rodada [de Souza et al. 2024]. Em cenários reais, os conjuntos locais são não independentes e identicamente distribuídos (não-IIDs) e a qualidade pode variar entre participantes, incluindo inversão de rótulos, o que torna a seleção por rodada um fator crítico para a qualidade do modelo [Jiang et al. 2025, Zhao et al. 2018].

Estratégias simples de seleção, como amostragem aleatória, podem ser insuficientes nesses casos, pois o servidor pode acabar selecionando clientes cujas atualizações trazem pouco ganho ao modelo global ou até degradam o desempenho. Além disso, a relevância de um cliente tende a mudar ao longo das rodadas conforme o modelo evolui, o que sugere tratar a seleção como um problema dinâmico de decisão sequencial no lugar do uso de um critério fixo. Ainda assim, é comum que a literatura recente adote políticas de participação baseadas em heurísticas ou em funções de utilidade avaliadas a cada rodada [Lai et al. 2021, Jee Cho et al. 2022].

Diante desse caráter dinâmico, este trabalho propõe o uso do Aprendizado por Reforço (*Reinforcement Learning* – RL) para guiar a seleção de clientes ao longo das rodadas, com enfoque em cenários heterogêneos e adversariais. Uma formulação possível consiste em considerar um único agente que aprende a pontuar individualmente cada cliente com base em sinais observáveis durante o treinamento e, a cada rodada, seleciona os K clientes com as maiores pontuações. Outra alternativa seria definir a ação do agente para que, em vez de atribuir pontuações individuais aos clientes, atribua pontuações a subconjuntos de K clientes. Nesse caso, o espaço de ações cresce de forma combinatória, com $\binom{N}{K}$ possibilidades para um conjunto de N clientes. Essa formulação torna-se um problema de alta dimensionalidade e dificulta o aprendizado à medida que N aumenta. Para contornar essa limitação, este trabalho adota uma abordagem baseada em Aprendizado por Reforço Multiagente (*Multi-Agent RL* – MARL), na qual cada cliente é modelado como um agente cooperativo. Essa abordagem permite combinar a seleção dos melhores K clientes, eliminando o problema combinatorial, sem perder o caráter coletivo da seleção a cada rodada.

Foram conduzidos experimentos em cenários não-IID, incluindo a inversão de rótulos (*label flipping*) em uma fração dos clientes, como forma de ação maliciosa. Para comparação, a seleção aleatória e uma formulação com SARL são utilizadas como *baseline*. Os resultados revelam um aumento na acurácia em comparação com o FedAvg e com uma estratégia baseada em SARL, indicando que a proposta evita a seleção de participantes prejudiciais para o treinamento. Ademais, os resultados revelam como o aumento da acurácia depende de forma direta da fração de clientes atacantes, da heterogeneidade dos dados e da fração de rótulos comprometidos.

O restante deste trabalho está organizado como se segue: A Seção 2 revisa a literatura em seleção de clientes federados em presença de ataques. A Seção 3 apresenta a fundamentação teórica sobre FL e MARL. Já a Seção 4 introduz a proposta para seleção de clientes do FL usando MARL. Os experimentos e os resultados são discutidos na Seção 5. Por fim, a Seção 6 conclui este trabalho e estabelece os próximos passos. A

Tabela 1 apresenta as variáveis utilizadas ao longo de todo o trabalho.

2. Trabalhos Relacionados

Ataques ao aprendizado federado (FL) têm sido amplamente estudados devido à impossibilidade do servidor observar diretamente os dados e o processo de treinamento local. A literatura sistematiza essas ameaças, classificando-as como envenenamento de dados (*data poisoning*) e envenenamento de modelos (*model poisoning*) e diferencia os ataques como direcionados (*targeted*) e não direcionados (*untargeted*) [Sagar et al. 2023]. Entre os ataques por envenenamento de dados, a inversão de rótulos (*label flipping*) destaca-se por sua simplicidade de execução e dificuldade de detecção, podendo degradar significativamente o desempenho do modelo global, especialmente quando uma fração dos clientes é maliciosa [Jebreel et al. 2022]. Esses trabalhos evidenciam a necessidade de mecanismos que reduzam a influência de clientes maliciosos durante o treinamento.

Diante deste cenário, diversas defesas foram propostas, atuando majoritariamente no estágio de agregação ou na filtragem de atualizações anômalas. Estratégias clássicas incluem agregadores tolerantes a falhas bizantinas [Blanchard et al. 2017] e métodos baseados em mediana coordenada [Yin et al. 2018], que buscam reduzir a influência de atualizações adversariais no modelo global. Contudo, tais abordagens mostram-se reativas, uma vez que se limitam a mitigar o impacto de clientes maliciosos somente após o envio das atualizações, sem mecanismos para orientar a seleção de participantes ou limitar sua recorrência ao longo das rodadas. Nessa direção, Lai et al. propõem uma estratégia de seleção de clientes baseada em utilidade dos dados e restrições de sistema, sem abordar explicitamente o cenário adversarial [Lai et al. 2021]. Mais recentemente, abordagens baseadas em aprendizado por reforço multiagente (MARL) têm sido exploradas para melhorar a eficiência e a coordenação do aprendizado federado. Zhang *et al.* e Tian *et al.* empregam múltiplos agentes para aprender políticas de seleção de clientes, priorizando participantes mais promissores a cada rodada. Tais trabalhos buscam ganhos tanto em desempenho quanto em custo de comunicação, mesmo considerando dados não-IID e restrições de recursos [Zhang et al. 2022, Tian et al. 2024].

Embora haja ampla literatura sobre defesas contra ataques em FL, principalmente no estágio de agregação e na filtragem de atualizações, abordagens que exploram explicitamente a seleção de participantes como mecanismo de mitigação ainda são menos

Tabela 1. Principais notações.

Símbolo	Significado	Símbolo	Significado.
N / K	Núm. de clientes total/selec.	i / t	Índice de cliente/rodada
α	Param. da dist. de Dirichlet	w_t	Pesos do modelo FL
$o_{i,t} \in \mathbb{R}^4$	Observação local	\mathcal{S}_t	Cientes selecionados
$a_{i,t} \in \{0, 1\}$	Ação de seleção do cliente	γ	Fator de desconto
r_t	Recompensa global	π	Política
l_t^{val}	Perda de validação	Q_{tot}	Função valor-ação
y_t	Alvo do RL (eq. de Bellman)	\mathcal{L}	Perda do modelo FL
$\mathcal{D} / \mathcal{D}_{val}$	Dataset de treino/val.	ϵ	Constante pequena
$\mathbf{X}_{i,t}^{(j)} / \mathbf{y}_{i,t}^{(j)}$	Entrada/saída j do modelo FL	$\hat{\mathbf{y}}_{i,t}^{(j)}$	Rótulo da amostra j
Θ_T	Pesos da rede-alvo do DQN	Θ_Q	Pesos da rede-Q do DQN

comuns, sobretudo quando formulado como um problema de decisão sequencial. O presente trabalho formula a seleção de clientes como um problema de MARL em ambiente adversarial, no qual agentes aprendem uma política que reduz a prioridade de participantes associados a padrões consistentes com ataques. Assim, este trabalho visa mitigar dinamicamente a degradação do modelo global, preenchendo uma lacuna da literatura.

3. Fundamentação Teórica

Esta seção introduz conceitos fundamentais em aprendizado federado (FL) e aprendizado por reforço multiagente (MARL).

3.1. Aprendizado Federado

No aprendizado federado, em cada rodada t , o servidor seleciona um subconjunto de K cliente, a partir de um conjunto de N clientes, e os envia uma cópia do modelo global w_t . Os clientes selecionados realizam algumas épocas de treinamento local utilizando seus próprios dados, que permanecem no dispositivo, mantendo a privacidade. Em seguida, eles transmitem de volta as atualizações do modelo. O servidor atua como agregador dos parâmetros recebidos, tipicamente usando uma média ponderada pelo tamanho do conjunto de dados de cada cliente (FedAvg [McMahan et al. 2017]). Esse novo modelo global é usado na próxima rodada. Esse processo se repete até um critério de convergência do modelo. Em cenários idealizados, com dados IID, o método de referência, o FedAvg apresenta bom desempenho. Entretanto, em cenários realistas, a heterogeneidade entre os dados dos clientes (cenários não-IID) tende a degradar significativamente o desempenho [Zhao et al. 2018]. Para lidar com esses desafios, a seleção de clientes torna-se um componente crucial. Trabalhos anteriores têm seguido duas linhas principais: estratégias heurísticas [Lai et al. 2021] e estratégias baseadas em aprendizado [Wang et al. 2020]. Este trabalho adota a segunda abordagem.

3.2. Aprendizado por Reforço Multiagente

O MARL é um método cooperativo no qual um conjunto de N agentes interage em um mesmo ambiente, frequentemente adotado em problemas nos quais um único agente não é suficiente para resolver a tarefa. Nesse contexto, o aprendizado é mais complexo, uma vez que os múltiplos agentes devem tomar decisões simultaneamente, e a dinâmica do ambiente passa a depender das ações de todos. O objetivo é aprender políticas que maximizem o retorno esperado. Os agentes são modelados em um jogo estocástico (*stochastic game*), definido por $(\mathcal{S}, \mathcal{A}_1, \dots, \mathcal{A}_N, P(s' | s, a_1, \dots, a_N), r_1, \dots, r_N)$. Em cada passo, os agentes escolhem ações $a_i \in \mathcal{A}_i$, formando a ação conjunta $\mathbf{a} = (a_1, \dots, a_N)$. A dinâmica do sistema evolui segundo a probabilidade de transição de estados $P(s' | s, \mathbf{a})$, onde $s \in \mathcal{S}$ é o estado global atual e $s' \in \mathcal{S}$ é o estado global seguinte. Cada agente i decide a ação acessando apenas uma observação local $o_i = O_i(s)$, dependente apenas do seu estado. Em geral, assume-se que P não é conhecida explicitamente, de modo que o agente observa a dinâmica por amostragem via interação com o ambiente.

As políticas são definidas como $\pi_i(a_i | o_i)$, mapeando uma observação local para uma ação de cada agente. No caso cooperativo, assume-se recompensa compartilhada $r_1, \dots, r_N = r$. Define-se então a função valor-ação sob a política conjunta π como

$$Q^\pi(s_t, a_t) = \mathbb{E}_{\pi, P} \left[\sum_{k=0}^{\infty} \gamma^k r(s_{t+k}, a_{t+k}) \mid s_t = s, a_t = a \right],$$

onde t indica a iteração atual e $Q^\pi(s, \mathbf{a})$ quantifica o retorno esperado ao executar a ação conjunta \mathbf{a} no estado s e, em seguida, seguir a política conjunta π . γ é um fator de desconto que penaliza recompensas de longo prazo.

O *Value Decomposition Network* (VDN) [Sunehag et al. 2017] é uma das abordagens de agregação para treinamento cooperativo de agentes em MARL. No VDN, cada agente utiliza uma rede neural profunda (*Deep Neural Network* – DNN) para inferir sua ação. Essa rede aproxima a função valor-ação $Q_{\theta_n}(s, a)$, onde θ_n são os parâmetros da DNN do agente. Para treinar o VDN, utiliza-se um *replay buffer* para armazenar transições na forma $(\mathcal{S}_t, \mathcal{A}_t, r_t, \mathcal{S}_{t+1})$. A função valor-ação conjunta é então representada como a soma das funções individuais:

$$Q_{\text{tot}}(s_t, a_t) = \sum_{n=1}^N Q_{\theta_n}(s_{i,t}, a_{i,t}),$$

onde $s_t = \{s_{i,t}\}_{i=1}^N$ e $a_t = \{a_{i,t}\}_{i=1}^N$ denotam o estado global e a ação conjunta. As DNNs dos agentes são treinadas com o objetivo de minimizar a perda (*loss*):

$$\mathbb{E}_{s_t, a_t, r_t, s_{t+1}} [(y_t - Q_{\text{tot}}(s_t, a_t))^2],$$

onde o valor alvo y_t é definido pela equação de Bellman:

$$y_t = r_t + \gamma \max_{a_{t+1}} Q_{\text{tot}}(s_{t+1}, a_{t+1}).$$

4. Modelagem do Problema de Seleção de Clientes usando MARL

Esta seção descreve o arcabouço proposto para seleção de clientes em aprendizado federado (FL), fundamentado em aprendizado por reforço multiagente (MARL). O objetivo é aprender uma política de seleção que melhore o desempenho do modelo global em cenários heterogêneos (não-IID) e sob inversão de rótulos (*label flipping*), utilizando sinais observáveis ao longo do treinamento federado.

A seleção de clientes é tratada como um problema de decisão sequencial ao longo de T rodadas federadas. Considera-se um conjunto fixo com N clientes, onde cada cliente i é modelado como um agente cooperativo. Na rodada t , dado o modelo global w_t , o servidor coleta d sinais observáveis do treinamento e constrói uma observação local $\mathbf{o}_{i,t} \in \mathbb{R}^d$ para cada cliente. Em seguida, o servidor seleciona o conjunto de participantes $\mathcal{S}_t \subseteq \{1, \dots, N\}$, com $|\mathcal{S}_t| = K$, e executa algumas épocas de treinamento local nos clientes selecionados, seguida de agregação via FedAvg, produzindo o modelo w_{t+1} .

Variáveis de decisão (seleção Top- K). Seja $a_{i,t} \in \{0, 1\}$ uma variável binária que indica se o cliente i é selecionado na rodada t . A cada rodada, impõe-se a restrição de selecionar exatamente K clientes:

$$\sum_{i=1}^N a_{i,t} = K, \quad \forall t \in \{1, \dots, T\}. \quad (1)$$

Função objetivo. Busca-se aprender uma política de seleção π que, em cada rodada t , define quais clientes participam do treinamento, representada por um vetor binário $a_t = (a_{1,t}, \dots, a_{N,t})$. A função objetivo, portanto, visa maximizar o retorno esperado associado à recompensa global r_t . Sendo assim, a função objetivo é definida como:

$$\max_{\pi} \mathbb{E} \left[\sum_{t=1}^T \gamma^{t-1} r_t \right], \quad (2)$$

onde $\gamma \in [0, 1)$ é o fator de desconto. A recompensa r_t é calculada no servidor a partir da perda (*loss*) de validação após a agregação. Considerando ℓ_t^{val} como a perda de validação do modelo global ao final da rodada t e $\bar{\ell}_t^{\text{val}}$ como a média das perdas de validação nas τ rodadas anteriores, define-se:

$$r_t = \frac{\bar{\ell}_t^{\text{val}} - \ell_t^{\text{val}}}{\bar{\ell}_t^{\text{val}} + \varepsilon}. \quad (3)$$

A recompensa r_t é escalar e compartilhada entre todos os agentes. A variável ε no denominador é um valor positivo pequeno (e.g. 10^{-6}) que garante estabilidade numérica quando $\bar{\ell}_t^{\text{val}}$ tende a zero.

4.1. Espaço de observação

Para viabilizar a tomada de decisão sob heterogeneidade e possíveis contribuições adversariais, cada agente (cliente) i é associado a um vetor de observação $o_{i,t} \in \mathbb{R}^4$, calculado no servidor a partir de $d = 4$ sinais observáveis do treinamento na rodada t . Utiliza-se

$$o_{i,t} = [\text{proj}_{i,t}, \text{gener}_{i,t}, \text{estag}_{i,t}^*, \text{serie}_{i,t}^*], \quad (4)$$

onde os termos são definidos a seguir.

Projeção de gradiente. O termo $\text{proj}_{i,t}$ mede o alinhamento entre a atualização local do cliente i e um gradiente de referência do servidor no início de uma rodada. Antes de iniciar o treinamento na rodada $t+$, calcula-se a atualização local do cliente i a partir do modelo global agregado ao final da rodada t (w_t) em poucos *mini-batches* dos seus dados locais e o gradiente de referência do servidor em poucos *mini-batches* de um conjunto de validação do servidor \mathcal{D}_{val} . Na prática, o conjunto de validação do servidor pode ser formado de dados públicos utilizados para testar o desempenho do modelo sem depender dos dados privados locais dos clientes. Denotando por $\nabla_{w_t} \mathcal{L}(\cdot)$ o gradiente da perda em relação aos parâmetros, define-se:

$$\text{proj}_{i,t} = \frac{\Delta w_{i,t}^\top (-\nabla_{w_t} \mathcal{L}(w_t; \mathcal{D}_{\text{val}}))}{\|\nabla_{w_t} \mathcal{L}(w_t; \mathcal{D}_{\text{val}})\|}. \quad (5)$$

onde $\Delta w_{i,t}$ denota a diferença entre os parâmetros do modelo local obtido pelo cliente i após a atualização local e os parâmetros do modelo global w_t , com $\nabla_{w_t} \mathcal{L}(\cdot)$ suavizado por uma média móvel.

Esta equação é proporcional à similaridade de cosseno. Valores positivos indicam que o cliente tende a atualizar o modelo na mesma direção do servidor, enquanto valores negativos indicam o oposto.

Perda de generalização A perda de generalização, denotada como $\text{gener}_{i,t}$, é calculada avaliando o modelo global w_t em poucos *mini-batches* do conjunto local do cliente i , antes da atualização de pesos. Valores elevados de $\text{gener}_{i,t}$ indicam que o modelo global generaliza mal para aquele cliente, o que pode refletir heterogeneidade dos dados e comportamento adversarial. O conjunto de dados do cliente i na rodada t é representado por $\left\{ \left(\mathbf{X}_{i,t}^{(j)}, \mathbf{y}_{i,t}^{(j)} \right) \right\}_{j=1}^{|\mathcal{D}|}$, onde cada uma das $|\mathcal{D}|$ amostras é formada por atributos (entrada) $\mathbf{X}_{i,t}^{(j)}$ e rótulo (saída esperada, ou *ground-truth*) $\mathbf{y}_{i,t}^{(j)}$. A passagem pelo modelo w_t é representada por $\hat{\mathbf{y}}_{i,t}^{(j)} = f \left(\mathbf{X}_{i,t}^{(j)}, w_t \right)$, onde a saída $\hat{\mathbf{y}}_{i,t}^{(j)}$ é o valor predito. Define-se:

$$\text{gener}_{i,t} = \frac{1}{|\mathcal{D}|} \sum_{j=1}^{|\mathcal{D}|} \mathcal{L} \left(\hat{\mathbf{y}}_{i,t}^{(j)}, \mathbf{y}_{i,t}^{(j)} \right), \quad (6)$$

onde a função de perda $\mathcal{L}(\cdot)$ usada é a entropia cruzada.

Estagnação. A variável $\text{estag}_{i,t}$ contabiliza quantas rodadas se passaram desde a última seleção do cliente i . Para obter um sinal comparável entre clientes na rodada t , aplica-se uma normalização pelo maior valor observado na rodada:

$$\text{estag}_{i,t}^* = \frac{\text{estag}_{i,t}}{\max_{j \neq i} \text{estag}_{j,t} + \varepsilon}. \quad (7)$$

Série de seleções. A $\text{serie}_{i,t}$ é a contagem de seleções consecutivas do cliente i até a rodada t . Define-se a versão normalizada como:

$$\text{serie}_{i,t}^* = \min \left(\frac{\text{serie}_{i,t}}{\text{serie}_{i,t}^{(\max)}}, 1 \right), \quad (8)$$

onde $\text{serie}_{i,t}^{(\max)}$ é um hiperparâmetro constante global, aplicado igualmente a todos os clientes e em todas as rodadas. Este hiperparâmetro define a quantidade de rodadas consecutivas que um mesmo cliente precisa ser selecionada para que $\text{serie}_{i,t}^*$ atinja seu valor máximo, igual a um.

4.2. Processo de Aprendizagem

Cada cliente i estima $Q_i(o_{i,t}, a)$, sendo a_i a variável binária indicadora de seleção (eq 9). Esses valores são calculados por uma rede neural treinada a partir do algoritmo *Deep Q-Network* (DQN), que recebe $o_{i,t}$ como entrada e retorna $Q_i(o_{i,t}, 0)$ e $Q_i(o_{i,t}, 1)$. O servidor seleciona os K clientes com maior diferença entre saídas:

$$S_t = \text{TopK} \left(\{Q(o_{i,t}, 1) - Q(o_{i,t}, 0)\}_{i=1}^N, K \right). \quad (9)$$

4.3. Treinamento centralizado com execução descentralizada

Adota-se o paradigma de treinamento centralizado com execução descentralizada. Na execução, cada cliente é avaliado de forma independente a partir de sua observação

local $o_{i,t}$, produzindo um escore utilizado pelo servidor na seleção Top- K (Eq. 9). Essa restrição de decisão local implica que cada agente observa apenas sinais do próprio cliente e não tem acesso direto às observações dos demais, de modo que a coordenação entre os agentes é estabelecida no treinamento a partir de uma função objetivo compartilhada. Para viabilizar esse aprendizado cooperativo sob observações parciais e uma recompensa global compartilhada r_t (Eq. 3), emprega-se a abordagem *Value Decomposition Networks* (VDN) para agregar as estimativas de ação-valor dos agentes. Em particular, define-se uma função $Q_{\text{tot}}(o_t, a_t)$ associada à decisão conjunta, decomposta como soma das estimativas individuais $Q_i(o_{i,t}, a_{i,t})$:

$$Q_{\text{tot}}(o_t, a_t) = \sum_{i=1}^N Q_i(o_{i,t}, a_{i,t}), \quad (10)$$

permitindo que múltiplos agentes aprendam de forma cooperativa a partir de um único sinal de recompensa. O treinamento minimiza o erro médio quadrático sobre a função agregada Q_{tot} , estimada pela rede neural com parâmetros Θ_Q (*Q-Network*):

$$\mathcal{L}(\theta) = (y_t - Q_{\text{tot}}(o_t, a_t; \Theta_Q))^2, \quad (11)$$

utilizando como variável alvo a diferença temporal, calculada por uma rede-alvo (*target network*) com parâmetros Θ_T :

$$y_t = r_t + \gamma \max_{a'} Q_{\text{tot}}(o_{t+1}, a'; \Theta_T). \quad (12)$$

Ao final de cada rodada federada t , forma-se a transição $\tau_t = (o_t, a_t, r_t, o_{t+1})$, em que $o_t = (o_{1,t}, \dots, o_{N,t})$ agrega as observações locais de todos os clientes, $a_t = (a_{1,t}, \dots, a_{N,t})$ representa as ações binárias executadas na rodada, r_t é a recompensa global computada no servidor, e o_{t+1} é construído a partir dos sinais observáveis da rodada seguinte. Essas transições são armazenadas em um *replay buffer* e amostradas aleatoriamente para treinamento *off-policy*, isto é, o estimador Q é atualizado com experiências coletadas por políticas anteriores (não necessariamente a política atual). Isso reduz a correlação temporal entre amostras consecutivas e permite reutilizar experiências de rodadas passadas.

Entretanto, nem todas as transições do *replay buffer* são igualmente informativas. Rodadas em que a seleção causa maior variação na perda de validação tendem a gerar atualizações mais relevantes do estimador Q . Para enfatizar essas experiências, utiliza-se *Prioritized Experience Replay* (PER), substituindo a amostragem uniforme por uma estratégia que atribui a cada transição uma prioridade proporcional ao erro de diferença temporal. Dessa forma, transições em que o estimador Q erra mais são revisitadas com maior frequência, acelerando a correção dos casos mais informativos.

Para estabilizar o aprendizado, adota-se a abordagem *Double DQN*, utilizando a rede-Q com parâmetros Θ_Q , além da rede-alvo com parâmetros Θ_T , citadas anteriormente na descrição do DQN. A rede-Q estima o retorno esperado $Q_{\text{tot}}(o_{t+1}, a'; \Theta_Q)$ para cada ação a' a partir da observação o_{t+1} amostrada do *replay buffer*. Em seguida, é escolhida, de forma gulosa, a ação que maximiza esse retorno:

$$a^* = \arg \max_{a'} Q_{\text{tot}}(o_{t+1}, a'; \Theta_Q). \quad (13)$$

A diferença temporal é calculada pela rede-alvo usando essa ação:

$$y_t = r_t + \gamma Q_{\text{tot}}(o_{t+1}, a^* ; \Theta_T). \quad (14)$$

A cada passo, a rede-alvo (Θ_T) é mantida fixa, enquanto que os pesos da rede-Q (Θ_Q) são atualizados por retropropagação, usando a perda definida na eq. 11. Apenas a cada C passos a rede-alvo é atualizada.

5. Experimentos

O objetivo dos experimentos é avaliar como a proposta de seleção de clientes no aprendizado federado (FL) por MARL desempenha em comparação com estratégias mais simples (FedAvg e SARL). São utilizados 50 clientes, e são selecionados 30% do número total de clientes disponíveis para treinamento ($K = 15$ clientes), exceto quando dito o contrário. Enquanto o FedAvg escolhe 30% dos clientes aleatoriamente a cada rodada, as alternativas com RL selecionam os 30% melhores para a rodada. A máquina utilizada para os experimentos possui uma GPU NVIDIA GeForce RTX 5090, 21.760 núcleos CUDA e memória VRAM GDDR7 de 32GB e 1,79TB/s. Ademais, todos os resultados foram obtidos ao longo de 500 rodadas de treinamento. Este valor foi escolhido empiricamente, tendo em vista a convergência observada em todos os cenários de avaliação. A implementação do arcabouço proposto, bem como o código-fonte utilizado nos experimentos, está disponível em <https://github.com/GTA-UFRJ/FEDMARL>.

O ambiente de aprendizado federado é implementado em PyTorch e utiliza uma Rede Neural Convolutiva (CNN) para classificação com 5 camadas. Assume-se a seguinte notação: $\text{Conv}(f, c_{\text{in}}, c_{\text{out}})$ representa uma camada convolutiva com *kernel* de tamanho f , c_{in} canais de entrada e c_{out} canais de saída, seguida de uma unidade retificadora linear, $\text{Pool}(a, b)$ representa uma camada de *pooling* de dimensões $a \times b$, $\text{FC}(c_{\text{in}}, c_{\text{out}})$ representa uma camada completamente conectada, e $\text{Relu}(c_{\text{in}}, c_{\text{out}})$ representa uma camada completamente conectada seguida de uma função retificadora linear. No ambiente MARL, a rede Q é uma rede totalmente conectada (MLP) de três camadas, treinada a partir de *mini-batches* amostrados do *replay buffer*, com uma rede-alvo sincronizada periodicamente (*target update*). Em cada rodada, a exploração é realizada por uma perturbação do Top- K , com probabilidade ϵ , trocam-se m clientes selecionados por não selecionados. Além disso, há uma fase inicial de *warm-up* com seleções aleatórias até que o *buffer* contenha transições suficientes. A Tabela 2 apresenta os principais parâmetros das redes e do treinamento. Como a rede do treinamento federado é uma CNN simples e customizada, os resultados não superam o estado da arte mas permitem uma comparação entre as estratégias.

O conjunto de dados é o CIFAR10, utilizado como *benchmarking* em trabalhos na área. Ele contém 50 mil imagens coloridas 32x32 para treino, e outras 10 mil para teste. Para simular um cenário com distribuição de rótulos não-IID, a quantidade de amostras de cada classe em cada cliente segue a distribuição de Dirichlet, com parâmetro α . Valores maiores de α aproximam a distribuição do caso uniforme, enquanto valores menores simulam cenários mais heterogêneos. A validação é realizada ao final de cada rodada, após a agregação do modelo global, como no FedAvg.

Para avaliar a influência dos principais parâmetros relevantes na proposta, esta seção é organizada em seis experimentos. O primeiro avalia o impacto da proporção

entre clientes atacantes (aqueles que praticam inversão de rótulos) e o total de clientes. O segundo avalia o impacto da proposta na seleção de clientes, evidenciando a menor participação dos clientes atacantes no processo de treinamento. O terceiro avalia o impacto da distribuição dos dados por meio da manipulação do parâmetro α da distribuição de Dirichlet. A ideia é mostrar que a acurácia do modelo é negativamente afetada sob o ponto de vista dos atacantes, servindo como um desestímulo ao ataque. A porcentagem de inversão de rótulos executada por cada atacante é verificada no quarto experimento, enquanto o quinto experimento o impacto do número K de clientes selecionados. Por fim, o desempenho da proposta é comparado ao SARL.

5.1. Impacto da proporção de clientes atacantes

Esta subseção avalia o impacto da proporção de clientes atacantes no total de clientes do aprendizado federado. Para isso, adota-se o parâmetro de Dirichlet igual a 0,3, considera-se ainda que os atacantes invertem o rótulo de todas as suas amostras, que o número total de clientes é 50 e que o número k de clientes selecionados é sempre 15 (30% do total). A Figura 1(a) apresenta o resultado de acurácia sem atacantes ao longo das rodadas de treinamento. Observa-se que o desempenho do FedAvg e do MARL é idêntico, chegando a valores próximos a 70% ao final do treinamento. Já a Figura 1(b) mostra o impacto da proposta ao preservar a acurácia do modelo próximo a do cenário sem atacantes. A acurácia do FedAvg, ao final das rodadas, é reduzida até aproximadamente 50%, como consequência da inclusão de clientes maliciosos no treinamento. A Figura 1(c) apresenta a mesma tendência, acentuando mais ainda a diferença entre a acurácia do modelo com o MARL e a acurácia do modelo usando o FedAvg, como consequência da proporção ainda maior de atacantes. Nos casos com atacante, a curva de acurácia do MARL não é superior à do FedAvg nas primeiras rodadas, pois os agentes ainda não tiveram tempo de explorar o espaço de estado. Após essa etapa inicial, os agentes aprendem uma política para definir as ações de seleção de cada cliente que supera o desempenho do FedAvg.

5.2. Impacto na seleção de clientes

A Tabela 3 apresenta o número médio de vezes que cada um dos clientes foi selecionado ao longo das rodadas de treinamento nos experimentos executados na Seção 5.1. Note que, sem atacantes, o número médio de vezes que cada cliente é selecionado é igual

Tabela 2. Parâmetros do treinamento da tarefa federada.

Parâmetros do treinamento federado.		Parâmetros do DQN.	
Nome do parâmetro	Valor do parâmetro	Nome do parâmetro	Valor do parâmetro
Camada de entrada	Conv(3,3,32) + Pool(2×2)	Camada de entrada	ReLU(4,128)
Camada 2	Conv(3,32,64) + Pool(2×2)	Camada 2	ReLU(128,128)
Camada 3	Conv(3,64,128) + Pool(2×2)	Camada de saída	FC(128,2)
Camada 4	ReLU(2048,256)	Função de perda	Huber
Camada de saída	FC(256,10)	Otimizador	Adam ($\lambda = 10^{-4}$)
Função de perda	Entropia cruzada	Learning rate	$\eta = 10^{-3}$
Otimizador	SGD (<i>mom.</i> = 0.9)	Batch size (base \rightarrow máx.)	32 \rightarrow 256
Learning rate	$\eta = 0.01$	Target update	$C = 20$
Batch size	64	Warm-up	50 rodadas
		Exploração	$\epsilon = 0.15, m = 3$
		Fator de de desconto	$\gamma = 0.90$

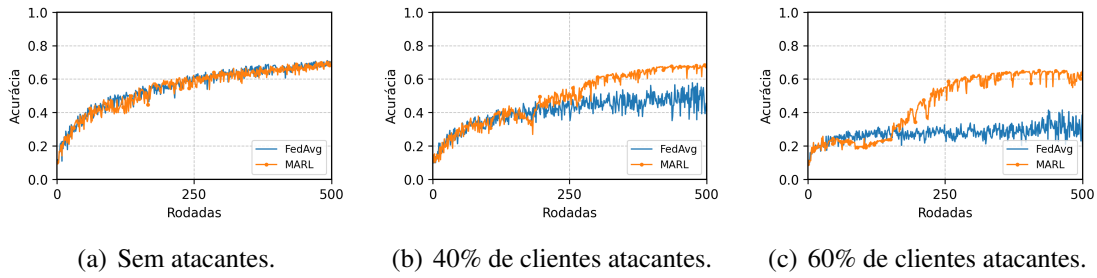


Figura 1. Impacto da proporção de clientes atacantes sobre o número total.

ao número de rodadas vezes a proporção de clientes selecionados, ou seja, $500 \times 0,3 = 150$, demonstrando que a probabilidade de escolha de cada cliente é aproximadamente a mesma. Com 40% de atacantes, o número de vezes que os clientes honestos e atacantes no FedAvg permanece próximo de 150, enquanto, para o MARL, observa-se o efeito da seleção em que os clientes honestos são priorizados. Com 60% de atacantes, o efeito da seleção de clientes no MARL é ainda acentuado. Esses resultados confirmam que a proposta, de fato, seleciona os clientes que mais contribuem para o treinamento do modelo, além de justificar os resultados da Seção 5.1.

Tabela 3. Número médio de vezes que os clientes honestos e os atacantes são selecionados para participar de rodadas de treinamento.

Cenário	FedAVG		MARL	
	Honestos	Atacantes	Honestos	Atacantes
Sem atacantes	150 ± 10.61	Não há	150 ± 38.27	Não há
40% de atacantes	149.13 ± 11.00	151.30 ± 12.37	204.17 ± 73.25	68.75 ± 11.93
60% de atacantes	151.20 ± 12.28	149.20 ± 10.00	257.95 ± 115.93	78.03 ± 36.87

5.3. Impacto da distribuição dos dados

A mesma configuração anterior é reutilizada nesta seção. Dessa forma, os atacantes invertem o rótulo de todas as suas amostras, o número total de clientes é 50 e o número K de clientes selecionados é 15. Porém, a porcentagem de atacantes é fixa em 40% e, diferentemente dos resultados anteriores, o foco deste experimento é verificar o impacto da distribuição dos dados na acurácia do modelo. Para isso, reduz-se o parâmetro α de Dirichlet para simular distribuições mais heterogêneas, com valores de 0,3, 0,2 e 0,1.

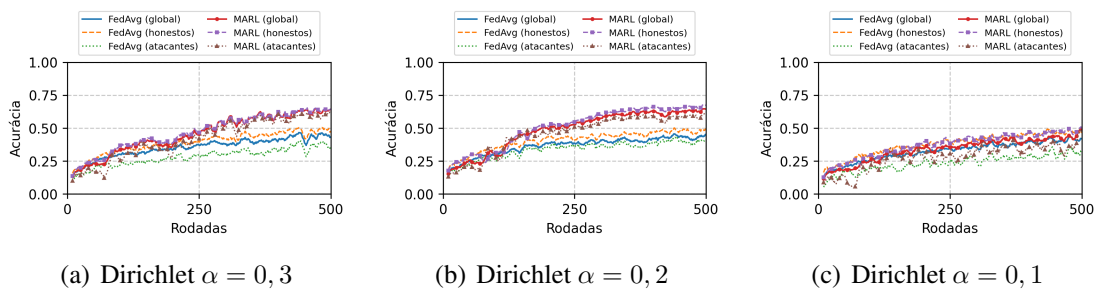


Figura 2. Impacto da distribuição dos dados com a variação do α de Dirichlet.

Um resultado adicional que esta subseção apresenta é a acurácia do modelo treinado sob o ponto de vista dos clientes honestos e atacantes. Este resultado é importante, já

que não participar de uma rodada de treinamento pode ser entendido como um estímulo ao ataque devido à economia de recursos computacionais e de rede. Nesse sentido, a Figura 2(a) mostra, no entanto, que os clientes honestos possuem sempre desempenho superior aos atacantes. Assume-se que o atacante adota uma postura egoísta em que os rótulos são invertidos apenas no treinamento, de forma que os testes são realizados sem a inversão de rótulos. No caso do MARL, os clientes atacantes contribuem menos com amostras, fazendo com que o modelo fique mais ajustado aos clientes honestos. No FedAvg, a diferença é ainda mais evidente e o desempenho do modelo global é ainda pior para os desonestos. Esse mesmo comportamento se mantém com a redução do parâmetro de Dirichlet, Figuras 2(b) e 2(c), sendo que para 0,1, há um decréscimo mais evidente da acurácia devido a heterogeneidade dos dados.

5.4. Impacto da porcentagem de inversão de rótulos

Este experimento avalia a convergência do modelo quando nem todas as amostras de um atacante têm seus rótulos comprometidos. Para isso, é utilizada a mesma distribuição de Dirichlet e valor de K da Seção 5.1 ($\alpha = 0, 3$ e 15 , respectivamente), e a mesma proporção de atacantes da Seção 5.3 (40%). A porcentagem de rótulos invertidos pelos atacantes é reduzida de 100% para 60%, e, em seguida, para 40%, para simular diferentes níveis de força do ataque. A Figura 3 revela um aumento na acurácia no FedAvg a medida que menos amostras são comprometidas. Já o MARL permanece estável, com acurácia próxima dos 64% em todos os cenários, superando o FedAvg. Esse desempenho é decorrente da seleção de clientes que tende a excluir os atacantes do treinamento.

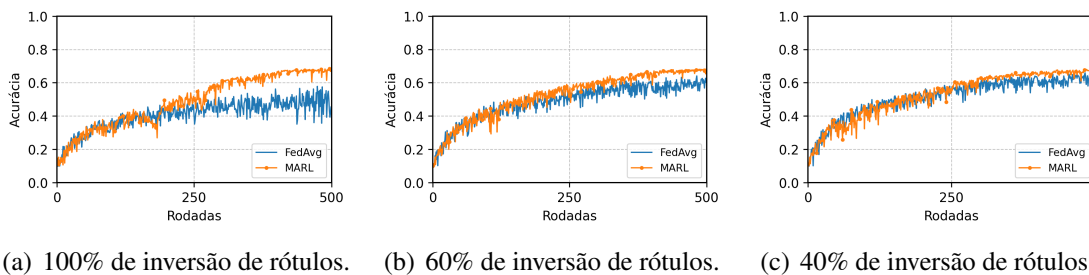


Figura 3. Impacto da porcentagem de inversão de rótulos.

5.5. Impacto da variação do número de clientes selecionados (parâmetro K)

Este experimento avalia a fração de clientes selecionados, parâmetro K , em 10, 15 e 35, utilizando os valores padrões de α (0,3), fração de atacantes (40%) e porcentagem de inversão de rótulos (100%). A Figura 4 revela que os resultados do MARL e do FedAvg são similares para $K=10$ e 15 , com o MARL apresentando melhor desempenho devido à seleção dos melhores clientes. Também é avaliado um cenário mais extremo para $K=35$, ou seja para um K que seleciona 70% dos clientes, no qual, dada a presença de 40% de atacantes, torna-se inviável selecionar apenas clientes honestos. Nesse cenário, o FedAvg degrada mais, enquanto o MARL mantém maior acurácia, sugerindo que a política aprendida prioriza, entre os atacantes inevitáveis, aqueles que menos degradam o modelo global.

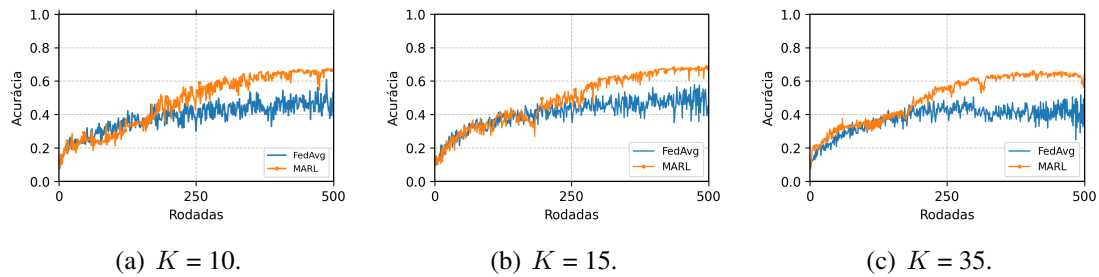


Figura 4. Impacto da variação do parâmetro K .

5.6. Comparação com o *single-agent* RL

Esta seção compara a proposta com MARL com a abordagem de agente único (SARL), onde ambos selecionam K clientes a cada rodada (Top- K), usando as mesmas configurações da Seção 5.1. Enquanto no SARL cada cliente é recompensado conforme sua contribuição para o treinamento na rodada, no MARL, a recompensa é conjunta entre os clientes. Conseqüentemente, o SARL possui uma visão local avaliada por cliente, enquanto o MARL coordena agentes sob uma mesma meta global. Na Figura 5, é possível observar que o MARL ainda obtém um desempenho melhor, destacando o benefício da tomada de decisões coletivas.

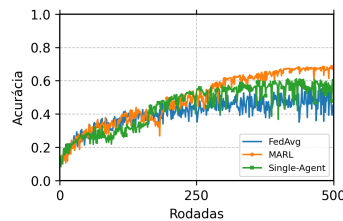


Figura 5. Comparação com o SARL.

6. Conclusão e Trabalhos Futuros

Este trabalho propõe o uso de aprendizado por reforço multiagente (MARL) na seleção de clientes do aprendizado federado (FL) com dados não independentes e identicamente distribuídos (não-IID) e inversão de rótulos. O trabalho contribui para a literatura ao demonstrar que o MARL evita a seleção de clientes que efetuam ataques adversariais durante o treinamento do modelo. Os resultados revelam um aumento na acurácia em relação ao FedAvg e ao uso de agente único (SARL) em todos os cenários, com destaque para cenários com maior proporção de atacantes, dados muito heterogêneos e com mais rótulos invertidos por atacante. Como trabalhos futuros, é prevista a implementação da proposta em um ambiente real, com modelos mais complexos, como modelos fundacionais de visão computacional e modelos grandes de linguagem (LLMs), assumindo também ataques mais complexos, além de comparações com um conjunto mais amplo de *baselines*.

Referências

Blanchard, P., Mhamdi, E. M. E., Guerraoui, R., and Stainer, J. (2017). Byzantine-Tolerant Machine Learning. *CoRR*, abs/1703.02757.

- de Souza, L. A. C., Camilo, G. F., Rebello, G. A. F., Sammarco, M., Campista, M. E. M., and Costa, L. H. M. K. (2024). ATHENA-FL: Avoiding Statistical Heterogeneity with One-versus-All in Federated Learning. *Journal of Internet Services and Applications*, 15(1):273–288.
- Jebreel, N. M., Domingo-Ferrer, J., Sánchez, D., and Blanco-Justicia, A. (2022). Defending against the Label-flipping Attack in Federated Learning. *CoRR*, abs/2207.01982.
- Jee Cho, Y., Wang, J., and Joshi, G. (2022). Towards Understanding Biased Client Selection in Federated Learning. In *25th International Conference on Artificial Intelligence and Statistics*, pages 10351–10375.
- Jiang, X., Li, J., Wu, N., Wu, Z., Li, X., Sun, S., Xu, G., Wang, Y., Li, Q., and Liu, M. (2025). FNbench: Benchmarking Robust Federated Learning against Noisy Labels. *CoRR*, abs/2505.06684.
- Lai, F., Zhu, X., Madhyastha, H. V., and Chowdhury, M. (2021). Oort: Efficient Federated Learning via Guided Participant Selection. In *15th USENIX Symposium on Operating Systems Design and Implementation (OSDI 21)*, pages 19–35.
- McMahan, B., Moore, E., Ramage, D., Hampson, S., and Aguera y Arcas, B. (2017). Communication-Efficient Learning of Deep Networks from Decentralized Data. In *20th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 1273–1282.
- Sagar, S., Li, C.-S., Loke, S. W., and Choi, J. (2023). Poisoning Attacks and Defenses in Federated Learning: A Survey. *CoRR*, abs/10.48550.
- Sunehag, P., Lever, G., Gruslyns, A., Czarnecki, W. M., Zambaldi, V., Jaderberg, M., Lanctot, M., Sonnerat, N., Leibo, J. Z., Tuyls, K., and Graepel, T. (2017). Value-Decomposition Networks For Cooperative Multi-Agent Learning. *CoRR*, abs/1706.05296.
- Tian, C., Shi, Z., Qin, X., Li, L., and Xu, C.-Z. (2024). Ranking-based Client Imitation Selection for Efficient Federated Learning. In *Proceedings of the 41st International Conference on Machine Learning*, pages 48211–48225.
- Wang, H., Kaplan, Z., Niu, D., and Li, B. (2020). Optimizing Federated Learning on Non-IID Data with Reinforcement Learning. In *IEEE Conference on Computer Communications (INFOCOM)*, pages 1698–1707.
- Yin, D., Chen, Y., Ramchandran, K., and Bartlett, P. (2018). Byzantine-Robust Distributed Learning: Towards Optimal Statistical Rates. In *35th International Conference on Machine Learning*, pages 5650–5659.
- Zhang, S. Q., Lin, J., and Zhang, Q. (2022). A Multi-Agent Reinforcement Learning Approach for Efficient Client Selection in Federated Learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 9091–9099.
- Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., and Chandra, V. (2018). Federated Learning with Non-IID Data. *CoRR*, abs/1806.00582.