PadLock: Certificação do Desaprendizado de Máquinas para IaaS via Blockchain *Layer 2*

Milena Curtinhas Santos[1], **João Paulo de Brito Gonçalves**[2], **Arthur A. Viana**[3]
Antonio A. de A. Rocha[3], **Rodolfo da Silva Villaça**[1]

[1]Departamento de Informática (DI/CT) – Universidade Federal do Espírito Santo (Ufes)
milena.c.santos@edu.ufes.br, rodolfo.villaca@ufes.br

[2]Instituto Federal do Espírito Santo (Ifes) – Campus Cachoeiro do Itapemirim/ES
jpaulo@ifes.edu.br

[3]Instituto de Computação (IC) – Universidade Federal Fluminense (UFF)
arthurvianna@id.uff.br, arocha@ic.uff.br

Abstract. *The right to be forgotten, established by the GDPR and the LGPD, requires cloud inference service providers to remove the influence of personal data from already trained models upon request. Although machine unlearning offers methods to address this demand, a critical gap remains: the absence of verifiable and independent certification of such removals. This work presents PadLock, an architecture that hosts the classifier in a Cartesi virtual machine (Layer 2), processes removal requests via DynFRS, and records the resulting model's cryptographic hash on the blockchain (Layer 1), composing an auditable chain of model versions. Evaluations on multiple tabular datasets demonstrate performance stability and the operational feasibility of the proposed certification.*

Resumo. *O direito ao esquecimento, estabelecido pelo GDPR e pela LGPD, obriga provedores de serviços de inferência em nuvem a remover a influência de dados pessoais sobre modelos já treinados. Embora o machine unlearning ofereça métodos para isso, persiste a ausência de certificação verificável e independente dessas remoções. Este trabalho apresenta o PadLock, uma arquitetura que hospeda o classificador em uma máquina virtual Cartesi (Layer 2), processa remoções via DynFRS e registra o hash criptográfico do modelo resultante na blockchain (Layer 1), compondo uma cadeia auditável de versões. Avaliações em múltiplos datasets demonstram estabilidade de desempenho e viabilidade operacional da certificação proposta.*

1. Introdução

A crescente adoção de serviços de inferência em nuvem (*Inference as a Service*, ou IaaS) expõe uma tensão fundamental entre a utilidade de modelos de aprendizado de máquina e os direitos individuais de privacidade. Regulamentações como o GDPR [União Europeia 2016] e a LGPD [Presidência da República 2018] estabelecem o direito ao esquecimento (*right to be forgotten*) [Dang 2021], que obriga provedores de serviços a remover dados pessoais mediante solicitação. Contudo, em modelos já treinados, a simples exclusão do dado bruto não garante a remoção de sua influência sobre os

parâmetros do modelo, problema que motivou o surgimento de técnicas de desaprendizado de máquinas (*machine unlearning*) [Cao and Yang 2015, Bourtoule et al. 2021].

Embora métodos de unlearning exato e aproximado tenham avançado significativamente [Xue et al. 2025], persiste uma lacuna crítica: a ausência de mecanismos de certificação verificável [Zhang et al. 2024]. Na prática, um provedor IaaS pode alegar que executou o unlearning sem que o usuário ou um auditor externo disponha de provas concretas e independentes dessa operação. Abordagens comportamentais e paramétricas de verificação são vulneráveis a adversários que preservam a influência dos dados supostamente esquecidos enquanto satisfazem os critérios de verificação [Jin et al. 2024]. Essa fragilidade compromete diretamente alegações de conformidade regulatória.

Diante desse cenário, este trabalho tem como objetivo o projeto, implementação e avaliação de uma arquitetura de *Certified Unlearning as a Service*, cujo propósito é prover certificação verificável e auditável de solicitações de remoção de dados em modelos de classificação oferecidos como serviço de inferência em nuvem. A solução integra *machine unlearning*, execução verificável em máquina virtual e registro imutável em blockchain, de forma que qualquer auditor externo possa validar independentemente se a influência de um dado foi efetivamente removida do modelo.

Na arquitetura proposta, o classificador é hospedado como serviço em uma máquina virtual Cartesi, uma solução *Layer 2* de blockchain baseada na arquitetura RISC-V que executa código de forma determinística e reproduzível. Solicitações de remoção são submetidas diretamente a essa VM: o algoritmo DynFRS (*Dynamic Forests for Randomized Subsampling*) executa o unlearning exato sobre o modelo, e o estado resultante do classificador é serializado e submetido a uma função de *hashing*. Esse *hash*, juntamente com os metadados da solicitação, é registrado na blockchain (*Layer 1*), compondo uma cadeia auditável de versões do modelo.

A principal contribuição do trabalho é um mecanismo de prova de unlearning que não depende da confiança no provedor: qualquer auditor externo pode reproduzir deterministicamente a sequência de remoções na Cartesi VM e verificar a consistência dos hashes registrados na blockchain. O protótipo está disponível publicamente[1] e foi desenvolvido no contexto do GT-Padlock (Projeto Iliada/RNP) e avaliado em múltiplos datasets tabulares, com experimentos que cobrem remoções sucessivas, remoção consolidada e custo transacional (*gas*) na *Layer 1*.

O restante do artigo está organizado da seguinte forma: a Seção 2 discute os trabalhos relacionados; a Seção 3 apresenta o framework DynFRS e seu funcionamento; a Seção 4 detalha a arquitetura proposta; a Seção 5 apresenta e discute os resultados experimentais; e a Seção 6 conclui o trabalho.

2. Trabalhos Relacionados

O campo de *machine unlearning* surge da necessidade de adequar sistemas de aprendizado de máquina às regulamentações de proteção de dados, como o GDPR [União Europeia 2016] e a LGPD [Presidência da República 2018], que estabelecem o direito ao esquecimento [Dang 2021]. A simples exclusão do dado bruto do conjunto de treinamento não é suficiente, pois modelos podem memorizar padrões es-

[1] `https://github.com/GT-Padlock/machine-unlearning-dapp`

pecíficos, tornando possível a recuperação de informações individuais por meio de ataques de privacidade [Nguyen et al. 2025]. Bourtoule et al. [Bourtoule et al. 2021] introduziram uma das primeiras formalizações do problema, propondo um framework de desaprendizado exato baseado em particionamento e isolamento de dados, que permite o retreinamento seletivo apenas das partições afetadas pela solicitação de remoção.

No contexto de florestas aleatórias, Wang et al. [Wang et al. 2025] propuseram o DynFRS (*Dynamic Forests for Randomized Subsampling*), um framework de desaprendizado exato que opera diretamente na estrutura das árvores de decisão. O DynFRS é adotado como algoritmo de desaprendizado no presente trabalho devido às limitações técnicas da solução de *Layer-2*, a ser explicada mais detalhadamente na Seção 4.1.

Apesar dos avanços em métodos de desaprendizado, a verificação das operações permanece um problema em aberto. Zhang et al. [Zhang et al. 2024] demonstraram que estratégias avançadas de verificação, incluindo abordagens baseadas em injeção de backdoors e provas de desaprendizado, são vulneráveis a processos adversariais que preservam a influência dos dados supostamente esquecidos e ainda assim satisfazem os critérios de verificação estabelecidos. Xue et al. [Xue et al. 2025] reforçam essa fragilidade em uma survey recente, apontando que a maioria dos métodos foca apenas no estado final do modelo, carecendo de trilhas de auditoria confiáveis que registrem quem solicitou o esquecimento, quando e quais evidências foram geradas.

Diante dessas limitações, o uso de blockchain como camada de auditoria descentralizada emerge como alternativa promissora [Lin et al. 2024, Zhu et al. 2024, Liu et al. 2025]. A imutabilidade e a rastreabilidade inerentes à tecnologia permitem o registro público de eventos como requisições de remoção, versões afetadas do modelo e hashes de provas, viabilizando a verificação independente por auditores externos sem depender da confiança no provedor [Zhu et al. 2024]. Nesse sentido, a Cartesi Machine oferece um ambiente de execução determinístico e reproduzível baseado na arquitetura RISC-V, que permite vincular a execução do algoritmo de desaprendizado a provas criptográficas verificáveis on-chain [Augusto et al. 2018].

Diferentemente dos trabalhos anteriores, que tratam o desaprendizado e a auditoria como problemas separados, o PadLock integra essas dimensões em uma arquitetura unificada de *Certified Unlearning as a Service*: o modelo é hospedado em uma Cartesi VM (*Layer 2*), as solicitações de remoção são processadas deterministicamente pelo DynFRS, e os hashes dos estados resultantes são registrados na blockchain (*Layer 1*), compondo uma cadeia auditável e resistente a adulterações de versões do modelo.

3. Desaprendizado de Máquina

O conceito de desaprendizado de máquina surge da necessidade de adequar sistemas de aprendizado de máquina ao direito ao esquecimento estabelecido por regulamentações como o GDPR e a LGPD, que exigem a remoção de dados pessoais mediante solicitação. Contudo, a simples exclusão do dado bruto não é suficiente quando o modelo já foi treinado, pois padrões e informações específicas podem ter sido memorizados, tornando possível a recuperação de dados individuais por meio de ataques de privacidade [Nguyen et al. 2025, Bourtoule et al. 2021]. O desaprendizado endereça essa lacuna com uma questão fundamental: como garantir que um modelo, após a remoção de um subconjunto de dados, não retenha qualquer influência desses dados em seu comportamento

futuro?

Formalmente, o desaprendizado pode ser definido como o processo pelo qual um modelo treinado sobre um dataset D é ajustado para remover completamente a influência de um subconjunto D_f, de modo que o modelo resultante $U(D, D_f, A(D))$ seja indistinguível de um modelo treinado diretamente sobre o conjunto remanescente $A(D \setminus D_f)$ [Nguyen et al. 2025]. Esse requisito de indistinguibilidade é essencial para garantir que o modelo não carregue vestígios dos dados removidos.

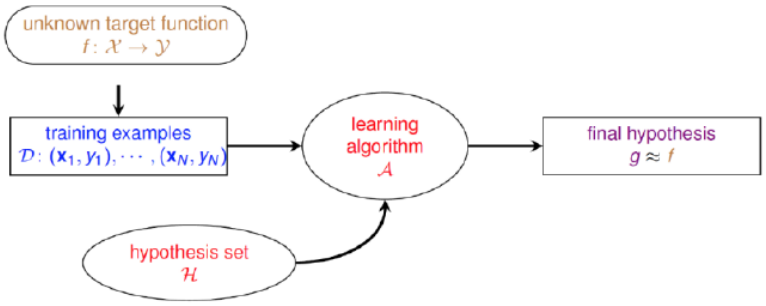

Figura 1. Diagrama do processo de aprendizado de máquina supervisionado [Cota et al. 2024].

A Figura 1 ilustra o processo de aprendizado supervisionado, no qual, dado um conjunto de hipóteses H, o objetivo é encontrar a hipótese final g que melhor aproxima a função-alvo f por meio do algoritmo de aprendizado A. A principal diferença entre desaprendizado e retreinamento completo reside na eficiência: retreinar do zero exige repetir todo o processo de aprendizado a cada solicitação de remoção, o que é proibitivo para modelos complexos e datasets de grande escala [Bourtoule et al. 2021]. O desaprendizado busca alternativas mais eficientes que produzam resultados equivalentes sem esse custo.

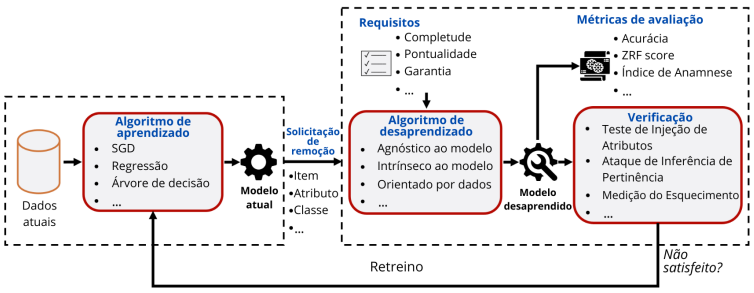

Figura 2. Fluxograma do processo de desaprendizado [Nguyen et al. 2025].

O desafio é agravado pela natureza estocástica e incremental do treinamento: modelos como redes neurais profundas são treinados em lotes aleatórios, e cada atualização de parâmetros reflete todas as anteriores, dificultando rastrear a influência exata de cada amostra [Nguyen et al. 2025, Bourtoule et al. 2021]. A Figura 2 apresenta o fluxo completo do processo, desde o treinamento até a verificação da eficácia da remoção.

3.1. DynFRS e sua Adequação à Cartesi VM

Neste trabalho, o desaprendizado é realizado pelo DynFRS (*Dynamic Forests for Randomized Subsampling*) [Wang et al. 2025], um framework de desaprendizado exato para

florestas aleatórias. A escolha do DynFRS é motivada não apenas por suas garantias formais de equivalência distribucional ao retreinamento completo, mas por uma restrição fundamental da infraestrutura adotada: a Cartesi Machine, baseada na arquitetura RISC-V, executa código em um ambiente isolado e determinístico que não oferece suporte a algumas bibliotecas nativas tradicionalmente usadas no aprendizado de máquina, tais como NumPy e PyTorch. O DynFRS, implementado em C++, opera diretamente sobre a estrutura das árvores de decisão sem depender dessas bibliotecas, tornando-o compatível com as restrições do ambiente de execução.

Formalmente, dado um algoritmo de floresta aleatória A que, a partir de um conjunto de treinamento S, produz uma hipótese $A(S)$ representada por uma floresta de T árvores de decisão, uma solicitação de desaprendizado consiste em remover um subconjunto $S_f \subseteq S$, produzindo uma nova hipótese $U(S, S_f, A(S))$ indistinguível de $A(S \setminus S_f)$. O DynFRS garante essa propriedade por meio de três mecanismos complementares.

O primeiro é a subamostragem OCC$_q$, que aloca cada amostra de treinamento a apenas uma fração q das T árvores, de modo que cada instância aparece em exatamente $k = qT$ árvores. Isso reduz o trabalho de treinamento e desaprendizado, garantindo um ganho esperado de aproximadamente $1/q^2$ em relação ao retreinamento ingênuo, sem degradação significativa de acurácia para valores típicos de q. O segundo mecanismo são as marcações lazy (LZY), que propagam requisições de remoção ao longo de um único caminho da raiz ao nó: se a melhor divisão do nó permanece inalterada, o efeito é local; caso contrário, o nó é marcado e a reconstrução é adiada até que uma consulta atravesse o nó marcado, amortizando o custo entre múltiplas consultas futuras. O terceiro mecanismo é o uso de Árvores Extremamente Aleatorizadas (ERT) como aprendiz base, que tornam as árvores menos sensíveis a remoções pontuais, permitindo atualizar estatísticas de divisão em tempo $O(1)$ por amostra e detectar mudanças na melhor divisão com custo $O(|S_u| \log s)$ por nó.

Para integração com a blockchain, o DynFRS foi modificado para suportar a serialização completa e reproduzível do estado da floresta após cada operação de remoção. Essa extensão permite que remoções sucessivas sejam aplicadas diretamente ao modelo já desaprendido, sem retorno ao estado original, preservando o histórico cumulativo de alterações. A natureza puramente algorítmica das operações OCC$_q$, LZY e ERT, combinada à serialização determinística, torna o comportamento do DynFRS totalmente compatível com a Cartesi VM: qualquer sequência de requisições aplicada ao mesmo estado inicial produz exatamente o mesmo estado final, viabilizando o cálculo de hashes imutáveis na blockchain para cada versão da floresta.

4. Arquitetura da Solução PadLock

O fluxo operacional do PadLock, ilustrado na Figura 3, tem início com o treinamento de um modelo de aprendizado de máquina sobre o dataset original, produzindo um modelo base que é registrado na blockchain como referência imutável do estado inicial do sistema. A partir dessa ancoragem, usuários podem submeter sucessivas solicitações de esquecimento, cada uma especificando quais registros devem ser removidos. Para cada solicitação i, o método de desaprendizado é aplicado ao modelo imediatamente anterior, removendo tanto os dados indicados do dataset quanto a influência estatística desses dados sobre os parâmetros do modelo, resultando em uma nova versão desaprendida. Cada

versão desaprendida é então persistida na blockchain juntamente com os metadados da solicitação correspondente, compondo uma cadeia de modelos e requisições auditável a qualquer momento.

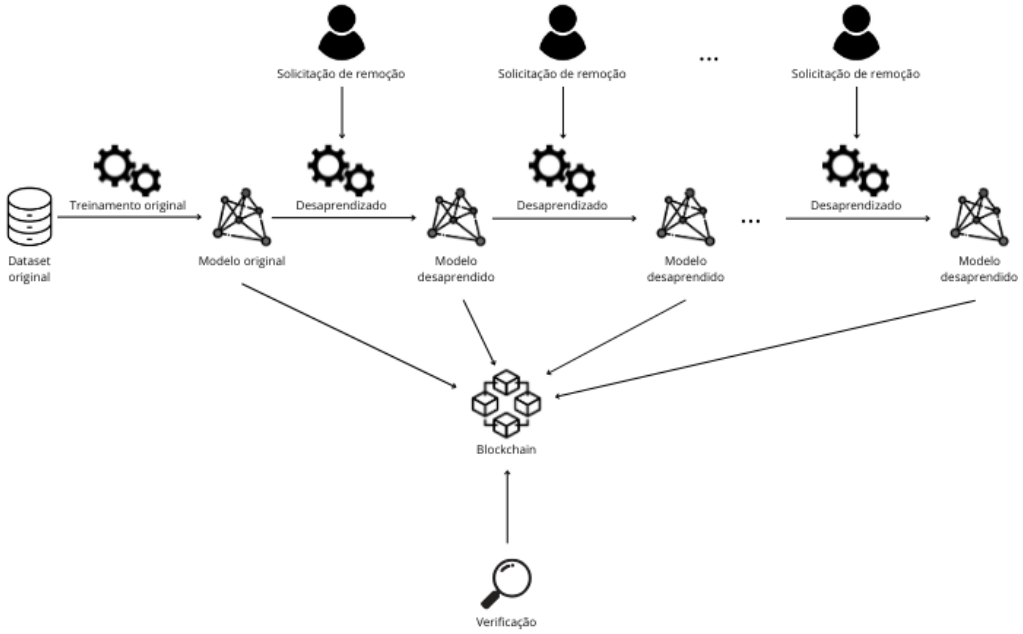

Figura 3. Fluxo operacional do sistema de desaprendizado certificado.

Com base nesse histórico, usuários ou entidades verificadoras podem emitir solicitações de verificação para confirmar se um determinado conjunto de dados foi efetivamente esquecido. Essas solicitações disparam uma rotina de verificação que compara o comportamento do modelo em relação aos dados supostamente removidos e, caso os critérios de esquecimento sejam satisfeitos, gera um certificado de desaprendizado. Esse certificado é criptograficamente vinculado ao identificador da solicitação e ao hash da versão do modelo, sendo também registrado na blockchain como prova pública e resistente a adulterações de que o processo de esquecimento foi corretamente executado.

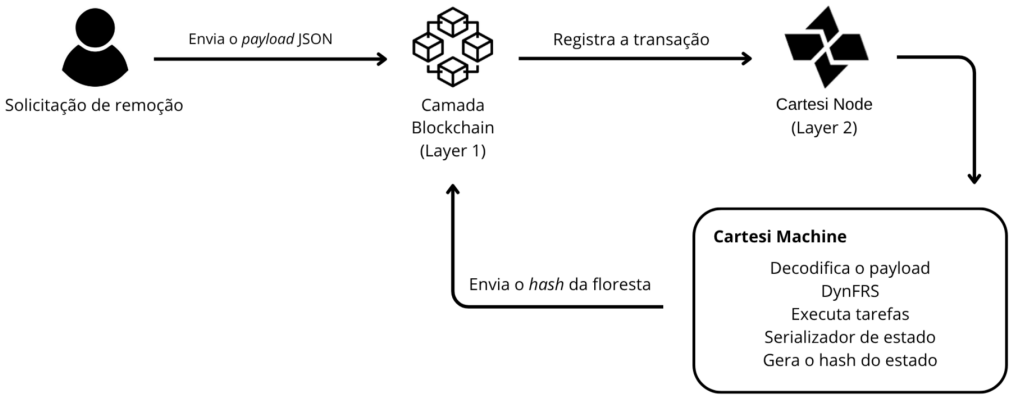

Figura 4. Visão geral da arquitetura técnica integrada à Cartesi Machine.

A arquitetura da solução, detalhada na Figura 4, é composta por elementos que operam em camadas *on-chain* e *off-chain*, projetadas para garantir a execução determinística do algoritmo de desaprendizado. O ciclo tem início com uma solicitação de remoção, na qual a aplicação envia um payload JSON à camada de blockchain, denominada Layer 1. Essa primeira camada tem a função crítica de prover disponibilidade de dados e registrar a transação, ordenando os eventos de forma imutável.

Na camada secundária, o Cartesi Node (Layer 2) atua como ponte de comunicação, monitorando a blockchain e encaminhando o estado consolidado ao ambiente de execução. Esse ambiente é a Cartesi Machine - uma máquina virtual baseada na arquitetura RISC-V que executa um sistema operacional de forma isolada e determinística. Ao receber os dados, a máquina decodifica o payload, aciona o módulo DynFRS e computa as métricas de avaliação requisitadas, como Acurácia e a Área sob a Curva ROC (AUROC).

Concluída a remoção das amostras solicitadas, o processo aciona o serializador de estado. O estado interno da floresta é percorrido para gerar um hash único e reproduzível. Por fim, a máquina retorna esse *hash* da floresta como prova criptográfica do novo estado desaprendido. O mecanismo prático de certificação ocorre por meio da verificação de estados baseada em hashes. O fluxo é disparado quando um usuário submete uma transação contendo parâmetros operacionais, especificando o dataset-alvo, hiperparâmetros (como k) e tarefas de verificação: por exemplo:

```
"data": "Adult", "k": 10, "tasks": [-acc", -auc"]
```

A rede emite um identificador hexadecimal (*transaction hash*), registrando permanentemente a ordem de submissão na blockchain.

No ambiente do nó validador, a Cartesi Machine recebe o payload hexadecimal, decodifica-o em seu formato JSON estruturado e aciona a rotina DynFRS. Após o processamento da floresta, as métricas de utilidade requeridas pela solicitação são extraídas. O estado final modificado é serializado *byte* a *byte* e submetido a uma função de *hash*. A solicitação, métricas obtidas e o *hash* resultante da árvore são encapsulados em um *notice* emitido pela máquina. Como a execução ocorre em ambiente completamente reproduzível, qualquer auditor que configure a Cartesi Machine e reprocesse o histórico de solicitações obterá invariavelmente o mesmo hash de certificação, provando matematicamente o processo de desaprendizado.

4.1. Garantias, Vantagens e Limitações

A abordagem de verificação do desaprendizado por meio da Cartesi Machine oferece fortes garantias de integridade temporal e reprodutibilidade estrita. Ao vincular o algoritmo de desaprendizado a uma máquina virtual determinística, a solução elimina vulnerabilidades de auditoria baseadas exclusivamente na confiança no provedor de serviços. A arquitetura garante que auditores externos possam validar o ciclo de vida completo do modelo a partir dos registros imutáveis providos pela *Layer 1*.

A principal vantagem desse mecanismo é a consolidação de um serviço de *Certified Unlearning* transparente e resistente a adulterações. Ao contrário de provas puramente comportamentais, que podem ser contornadas por adversários, o *hash* de estado garante que as modificações paramétricas exigidas pela exclusão dos dados ocorreram

exatamente conforme ditado pelo protocolo.

Em contrapartida, a solução apresenta limitações estruturais e de desempenho. Do ponto de vista do custo computacional, a execução do DynFRS em uma VM baseada em RISC-V é inerentemente mais lenta e demanda mais recursos do que execuções nativas. Adicionalmente, a serialização obrigatória do estado completo da floresta a cada interação de desaprendizado introduz gargalos de tempo e memória, podendo comprometer a escalabilidade para modelos muito grandes ou padrões de requisição de alta frequência.

Por fim, os custos de transação (*gas fees*) inerentes à blockchain para registro na *Layer 1* crescem proporcionalmente ao tamanho do *payload* da solicitação, limitando a viabilidade econômica da solução. É importante destacar que o mecanismo de certificação do PadLock oferece garantias de integridade processual: o *hash* registrado *on-chain* prova que o protocolo de desaprendizado foi executado deterministicamente conforme especificado, mas não constitui, por si só, uma garantia criptográfica de privacidade. Em particular, o sistema não verifica se a influência estatística dos dados removidos foi completamente eliminada do modelo. A avaliação dessa propriedade requer técnicas complementares, como ataques de inferência de membros (*Membership Inference Attacks*) sobre as amostras removidas ou métricas formais de esquecimento. A integração dessas verificações ao fluxo de certificação, de modo que suas evidências sejam também registradas na *blockchain*, constitui uma direção direta de trabalho futuro.

Tabela 1. Acurácia sob remoção baseada em porcentagem no algoritmo DynFRS.

q	Vaccine	Adult	Bank	Diabetes	NoShow
0.01	**0.7652**	**0.8439**	**0.9057**	**0.6156**	**0.7958**
0,10	0,7813	0,8543	0,9138	0,6319	0,7966
0,20	0,7918	0,8660	0,9173	0,6449	0,7970
0,30	0,7941	0,8651	0,9156	0,6455	0,7974
0,40	0,7924	0,8654	0,9142	0,6447	0,7973
0,50	0,7918	0,8650	0,9164	0,6455	0,7972
0,60	0,7954	0,8663	0,9155	0,6456	0,7958
0,70	0,7945	0,8646	0,9149	0,6446	0,7960
0,80	0,7956	0,8660	0,9147	0,6447	0,7968
0,90	0,8003	0,8658	0,9150	0,6437	0,7961
1,00	0,7954	0,8652	0,9128	0,6452	0,7961

5. Resultados e Discussões

Os experimentos com o DynFRS conduzidos em trabalho anterior [Santos et al. 2025] foram adotados como referência para este trabalho, servindo como *baseline* para as investigações subsequentes. Naquele estudo, o DynFRS foi avaliado em múltiplos conjuntos de dados tabulares sob diferentes taxas de remoção de amostras, variando o parâmetro q, que controla quantas árvores em uma floresta aleatória são efetivamente afetadas por cada instância removida. Os resultados apresentados na Tabela 1 e na Figura 5 mostram que, para a maioria dos conjuntos de dados, uma queda na acurácia foi observada apenas para $q < 1$.

Neste contexto, conclui-se que o parâmetro q do método OCC_q revela um trade-off fundamental entre eficiência computacional e desempenho preditivo. Configurações

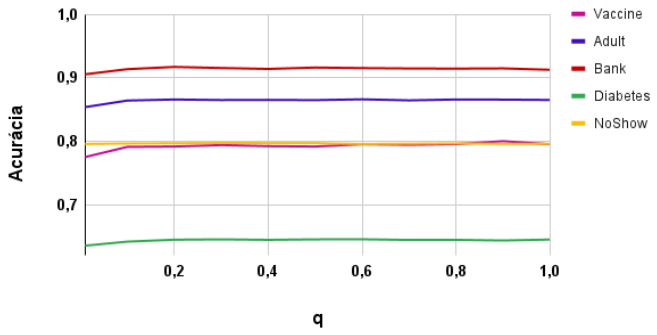

Figura 5. Acurácia sob remoção baseada em porcentagem no algoritmo DynFRS.

agressivas ($q < 0{,}1$) aceleram o desaprendizado ao limitar o número de subárvores atualizadas por remoção, mas comprometem a diversidade do *ensemble* e a capacidade de generalização. Por outro lado, valores mais altos ($q > 0{,}5$) preservam a robustez do modelo com acurácia próxima à original, embora cada solicitação de esquecimento impacte significativamente mais árvores, elevando o custo computacional.

Com base nesse cenário, este trabalho parte da versão original do DynFRS, estende sua implementação de forma a permitir remoções sucessivas, com armazenamento dos modelos intermediários durante essas remoções, e investiga o comportamento do modelo sob remoções sucessivas em uma infraestrutura descentralizada de *blockchain*. A avaliação foca no desempenho após múltiplas solicitações de esquecimento, o que motivou as modificações propostas de serialização de estado e os novos experimentos na Cartesi Machine.

Para avaliar a estabilidade do modelo modificado (DynFRS serializado) integrado à blockchain, conduzimos análises divididas em dois cenários focados na degradação da acurácia e do AUROC (*Area Under the Receiver Operating Characteristic Curve*). É importante destacar aqui que este trabalho, conforme anteriormente explicado, faz uso do DynFRS como solução de desaprendizado devido às limitações da própria infraestrutura descentralizada.

No Cenário 1, investigamos o impacto de múltiplas exclusões sequenciais. Modelos treinados nos datasets Adult, Bank, Heart, Vaccine, Diabetes e Synthetic foram submetidos a dez rodadas consecutivas de remoções em lote. Em cada rodada, um lote de tamanho variável k ($k = 5, 10, 15, 30, 40$, totalizando 100 amostras) foi removido do modelo, cujo estado era salvo na Cartesi VM e registrado por meio de um notice nos registros da blockchain. O modelo salvo na Cartesi VM era então reutilizado na rodada seguinte de remoção. A Figura 6 demonstra que tanto a acurácia quanto o AUROC permaneceram altamente estáveis ao longo das dez rodadas sucessivas para todos os conjuntos de dados. No dataset Adult, por exemplo, a acurácia flutuou minimamente entre 0,861 e 0,866, enquanto no Bank permaneceu na faixa de 0,913 a 0,916. Essa resiliência demonstra que a técnica de serialização do DynFRS preserva a distribuição estatística do modelo mesmo após intervenções repetidas.

O Cenário 2 visou contrastar esse comportamento com o limite teórico de uma única operação de desaprendizado massivo. Para isso, agregamos todas as amostras removidas sequencialmente no Cenário 1 e aplicamos uma única remoção equivalente ao

modelo base. A Tabela 2 compara as métricas finais obtidas na décima rodada do Cenário 1 com os resultados da remoção em lote único do Cenário 2.

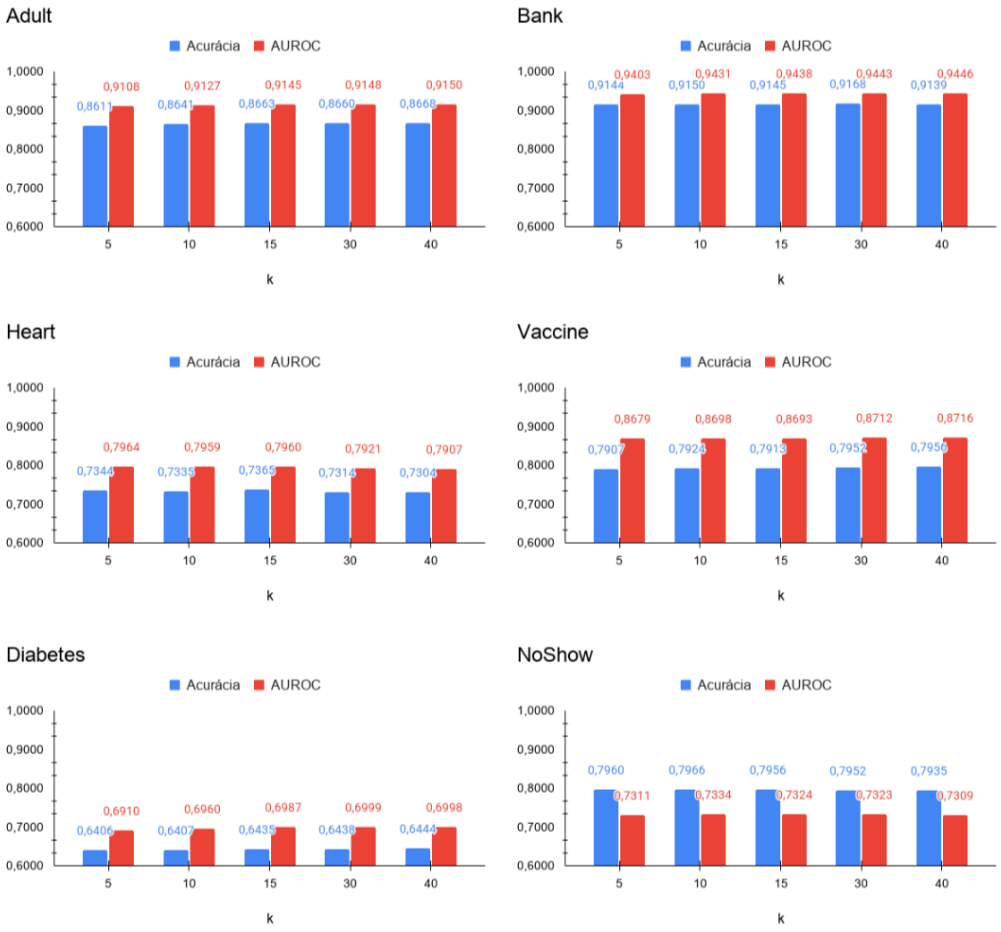

Figura 6. Cenário 1: Evolução da precisão e da área sob a curva ROC (AUROC) de remoções sucessivas em lote nos conjuntos de dados avaliados.

Os resultados mostram que as diferenças de acurácia e AUROC entre a remoção particionada e a completa são praticamente negligenciáveis, validando que o salvamento de estados intermediários na versão modificada não introduz perda de desempenho cumulativa em relação ao DynFRS original. Esse resultado decorre diretamente das propriedades formais do DynFRS: como cada amostra afeta exatamente $k = qT$ árvores de forma independente, e as marcações *lazy* propagam efeitos apenas nos nós efetivamente alterados, o estado final da floresta converge para o mesmo resultado independentemente de as remoções serem aplicadas em lotes sucessivos ou de uma só vez.

O Cenário 3 avalia a viabilidade operacional e financeira do *Certified Unlearning as a Service* através da camada de blockchain (Layer 1).

Quanto à certificação, o ciclo de vida do modelo na Cartesi Machine opera em um laço verificável: um conjunto de dados é enviado para remoção, o modelo atualiza seus parâmetros e a máquina virtual serializa o estado resultante para gerar um ”certificado”único (hash criptográfico). Na próxima solicitação, o processo reinicia a partir deste exato estado, computando a próxima exclusão e gerando um novo certificado sequencial.

Tabela 2. Cenário 2: Comparação de métricas entre remoções sucessivas (6 rodadas) e uma única remoção consolidada.

Dataset	Cénario 1 (6 Rodadas Sucessivas)		Cenário 2 (Remoção Única Consolidada)	
	Acurácia Final	AUROC Final	Acurácia	AUROC
Adult	0,8668	0,9150	0,8644	0,9137
Bank	0,9139	0,9446	0,9130	0,9427
Heart	0,7304	0,7907	0,7199	0,7791
Vaccine	0,7956	0,8716	0,7954	0,8693
Diabetes	0,6444	0,6998	0,6447	0,7008
NoShow	0,7935	0,7309	0,7805	0,7127

Esta cadeia de *hashes* fornece um registro imutável e transparente de exclusões parciais, servindo como prova matemática à prova de adulteração para auditorias de privacidade.

A execução prática desse ciclo pode ser observada através da interface de linha de comando da solução. A Figura 7 ilustra o momento da solicitação de remoção, onde o usuário envia um *payload* JSON (contendo o dataset ”Adult”, o parâmetro k=10 e as tarefas de métricas) para a rede, resultando no envio de um *input* hexadecimal e na confirmação da transação.

Figura 7. Solicitação de remoção enviada à Cartesi Machine com payload JSON.

Em seguida, a Figura 8 demonstra o processamento interno na Cartesi Machine: o nó validador recebe o sinal de *ADVANCE*, decodifica o *payload* JSON e aciona a rotina DynFRS, que remove deterministicamente 10 amostras do dataset Heart em aproximadamente 43 ms. Concluída a remoção, o serializador de estado percorre byte a byte a estrutura interna da floresta e computa o hash criptográfico do modelo resultante, identificado no log como `01H887729835779dd0426f119ea95fedf2f10c0c6f9f263e66332516599cffa14`.

Para garantir a transparência e auditabilidade, a arquitetura utiliza o mecanismo de *notices*. Um *notice* é um evento informativo registrado no banco de dados do Cartesi Node que encapsula resultados ou provas da execução em Layer-2. Conforme exibido

à direita da Figura 8, através do script `get_notices.sh`, é possível recuperar todos os *notices* registrados; neste caso, o *payload* do *notice* contém o hash do modelo e os metadados da operação (`unl_cnt`: 10) e `unl_cnt_time_ms`: 42.

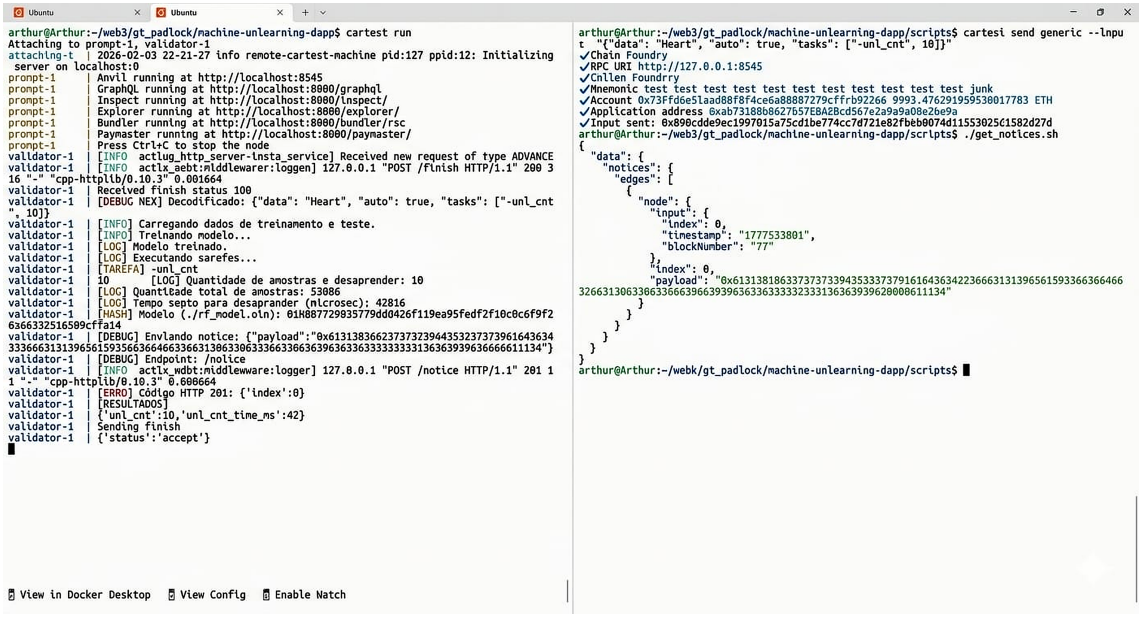

Figura 8. Execução determinística e auditoria via notices: à esquerda, o log do nó validador detalha o tempo de processamento e a geração do hash do modelo; à direita, a recuperação do notice via script de consulta, expondo o *payload* que será passível de prova e verificação em Layer-1 após a finalização do estado.

O diferencial desta abordagem é que, uma vez que o estado do *rollup* contendo este *notice* é consolidado, uma prova de validade associada a ele é gerada, permitindo a verificação da autenticidade da operação diretamente na Layer-1. Como a execução ocorre em um ambiente completamente isolado e reproduzível, qualquer auditor que reprocessar o mesmo histórico de solicitações obterá invariavelmente o mesmo hash, constituindo uma prova matemática e resistente a adulterações de que o desaprendizado foi efetivamente realizado conforme o protocolo registrado.

No entanto, a imutabilidade proporcionada pela *blockchain* introduz um custo transacional (medido em unidades de Gas). Testes mostraram que o custo de submissão de uma solicitação de remoção é diretamente proporcional ao comprimento da *string* (tamanho do *payload* JSON) que contém as amostras a serem desaprendidas e a solicitação realizada - armazenada para fins de auditoria.

A Tabela 3 apresenta o custo estimado ao submeter identificadores de usuários em requisições on-chain. Observa-se que a adição de um único caractere ao identificador alvo da solicitação (ex: *user1* vs. *user10*) aumenta o custo da transação uniformemente em 12 unidades de Gas em todos os datasets. Isso implica um desafio de escalabilidade para operações do mundo real. Embora solicitações únicas (ex: remoção simultânea de 1.000 IDs concatenados) otimizem o custo de transação da rede base em comparação a 1.000 chamadas de contrato separadas, elas exigem *payloads* grandes. Por outro lado, o envio contínuo de pequenos lotes incorre no pagamento repetido das taxas mínimas da Layer 1.

Portanto, a eficiência prática do serviço depende do agrupamento inteligente de múltiplas solicitações de usuários em strings condensadas antes da submissão à *blockchain*.

Tabela 3. Estimated Gas cost for removal requests via Cartesi.

Dataset	Cost (Gas) - ”user1”	Cost (Gas) - ”user10”
Adult	72.224	72.236
Bank	72.212	72.224
Heart	72.224	72.236
Vaccine	72.248	72.260
Diabetes	72.260	72.272
NoShow	72.236	72.248

6. Conclusão e Trabalhos Futuros

Este trabalho apresentou o PadLock, uma arquitetura que integra desaprendizado exato, execução determinística na Cartesi Machine (*Layer 2*) e registro imutável de hashes na blockchain (*Layer 1*), provendo certificação verificável de remoções de dados em serviços de inferência em nuvem. Os experimentos demonstraram que a serialização do estado da floresta não introduz perda cumulativa de desempenho, e que acurácia e AUROC permanecem estáveis ao longo de ciclos sucessivos de desaprendizado. A análise de custo transacional indicou que o agrupamento inteligente de requisições é essencial para amortizar as taxas de *gas* da *Layer 1*.

Como trabalhos futuros, a principal direção é o porte de algoritmos de desaprendizado mais expressivos para a Cartesi VM. A escolha do DynFRS foi motivada pela sua implementação em C++, que dispensa bibliotecas nativas de ML incompatíveis com a arquitetura RISC-V. O suporte a modelos como redes neurais profundas e *gradient boosting* exige investigar estratégias de compilação cruzada de bibliotecas de ML para esse ambiente restrito. Adicionalmente, pretende-se avaliar uma interface amigável para traduzir as provas criptográficas em evidências interpretáveis pelo usuário final.

Agradecimentos

À Fapes (2026-B74MN, 2023-RWXSZ, 2025-1H3FP), FAPERJ - LINE 260.279/2026, CNPq e Capes (Código de Financiamento 001) pelo financiamento parcial por meio de fomento de projetos de pesquisa e bolsas de IC. Ao GT-Padlock/RNP, pelo apoio técnico no desenvolvimento deste trabalho.

Referências

Augusto, A. J. et al. (2018). Cartesi: Scalable smart contracts through a risc-v microarchitecture and the linux operating system. Technical report, Cartesi Foundation.

Bourtoule, L., Chandrasekaran, V., Choquette-Choo, C. A., Jia, H., Travers, A., Zhang, B., Lie, D., and Papernot, N. (2021). Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 141–159.

Cao, Y. and Yang, J. (2015). Towards making systems forget with machine unlearning. In *IEEE Symposium on Security and Privacy (S&P)*, pages 463–480. IEEE.

Cota, D. O. C., de Jesus, D. C. S., and Rocha, A. A. d. A. (2024). Desaprendizado de máquinas e a lgpd: da privacidade ao direito ao esquecimento. In *Livro-texto de Minicursos – 42º Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC 2024)*. Sociedade Brasileira de Computação.

Dang, Q.-V. (2021). Right to be forgotten in the age of machine learning. In Antipova, T., editor, *Advances in Digital Science*, pages 403–411, Cham. Springer International Publishing.

Jin, R., Chen, M., Zhang, Q., and Li, X. (2024). Forgettable federated linear learning with certified data unlearning. *arXiv preprint arXiv:2306.02216*.

Lin, Y., Gao, Z., Du, H., Ren, J., Xie, Z., and Niyato, D. (2024). Blockchain-enabled trustworthy federated unlearning. *arXiv preprint arXiv:2401.15917*.

Liu, W. et al. (2025). Blockful: Enabling unlearning in blockchained federated learning. *arXiv preprint arXiv:2402.16294*.

Nguyen, T. T., Huynh, T. T., Ren, Z., Nguyen, P. L., Liew, A. W.-C., Yin, H., and Nguyen, Q. V. H. (2025). A survey of machine unlearning. *ACM Trans. Intell. Syst. Technol.*, 16(5).

Presidência da República (2018). Lei geral de proteção de dados pessoais (lgpd). `https://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/l13709.htm`. Acesso em: 24 jun. 2025.

Santos, M., Gonçalves, J., Rocha, A., and Villaça, R. (2025). Esquecer é preciso: Um estudo sobre o impacto da remoção de dados no desaprendizado de máquinas. In *Anais da X Escola Regional de Informática do Espírito Santo*, pages 11–20, Porto Alegre, RS, Brasil. SBC.

União Europeia (2016). Regulamento geral sobre a proteção de dados - rgpd). `https://eur-lex.europa.eu/legal-content/PT/TXT/?uri=celex%3A32016R0679`. Acesso em: 21 jun. 2024.

Wang, S., Shen, Z., Qiao, X., Zhang, T., and Zhang, M. (2025). Dynfrs: An efficient framework for machine unlearning in random forest. *arXiv preprint arXiv:2410.01588*.

Xue, L., Hu, S., Lu, W., Shen, Y., Li, D., Guo, P., Zhou, Z., Li, M., Zhang, Y., and Zhang, L. Y. (2025). Towards reliable forgetting: A survey on machine unlearning verification. *arXiv preprint arXiv:2506.15115*.

Zhang, B., Chen, Z., Shen, C., and Li, J. (2024). Verification of machine unlearning is fragile. *arXiv preprint arXiv:2408.00929*.

Zhu, T. et al. (2024). Federated learning with blockchain-enhanced machine unlearning: A trustworthy approach. *arXiv preprint arXiv:2405.20776*.