martFL

논세이퍼 5회차
최원재
2024년 8월 16일

목차

Introduction

Decipher

Paper Details

Author & Index Terms

martFL: Enabling Utility-Driven Data Marketplace with a Robust and Verifiable Federated Learning Architecture*

Qi Li
Tsinghua University & Zhongguancun Laboratory
li-q20@mails.tsinghua.edu.cn

Zhuotao Liu†
Tsinghua University & Zhongguancun Laboratory
zhuotaoliu@tsinghua.edu.cn

Qi Li
Tsinghua University & Zhongguancun Laboratory
qli01@tsinghua.edu.cn

Ke Xu
Tsinghua University & Zhongguancun Laboratory
xuke@tsinghua.edu.cn

Lab Info

CCS CONCEPTS

• **Computing methodologies** → *Multi-agent systems*; • **Security and privacy** → **Privacy-preserving protocols**;

KEYWORDS

Robust Federated Learning; Data Marketplace; Verifiable Learning

martFL - CCS '23

Conference

Preliminaries

Decipher

Federated Learning

What is Federated Learning?

- 연합학습은 기계학습의 하위 분야
- 여러 클라이언트가 데이터가 탈중앙화된 상태에서 공동으로 모델을 학습시키는 기계학습 방법론
- Google의 2016년 논문 Communication-Efficient Learning of Deep Networks from Decentralized Data 에서 처음 제안

Communication-Efficient Learning of Deep Networks from Decentralized Data

H. Brendan McMahan **Eider Moore** **Daniel Ramage** **Seth Hampson** **Blaise Agüera y Arcas**
Google, Inc., 651 N 34th St., Seattle, WA 98103 USA

Abstract

Modern mobile devices have access to a wealth of data suitable for learning models, which in turn can greatly improve the user experience on the device. For example, language models can improve speech recognition and text entry, and image models can automatically select good photos. However, this rich data is often privacy sensitive, large in quantity, or both, which may preclude logging to the data center and training there using conventional approaches. We advocate an alternative that leaves the training data distributed on the mobile devices, and learns a shared model by aggregating locally-computed updates. We term this decentralized approach *Federated Learning*.

We present a practical method for the federated learning of deep networks based on iterative model averaging, and conduct an extensive empirical evaluation, considering five different model architectures and four datasets. These experiments demonstrate the approach is robust to the unbalanced and non-IID data distributions that are a defining characteristic of this setting. Communication costs are the principal constraint, and we show a reduction in required communication rounds by 10–100× as compared to synchronized stochastic gradient descent.

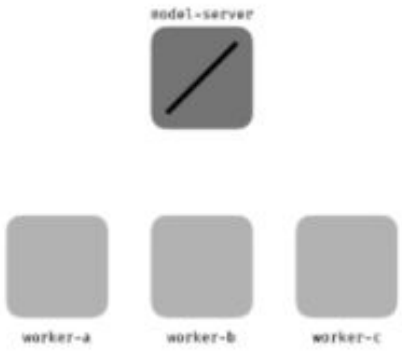
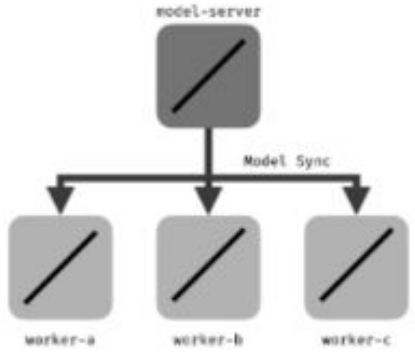
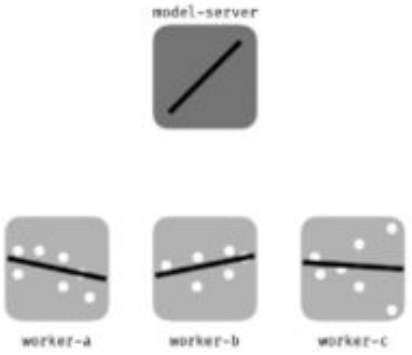
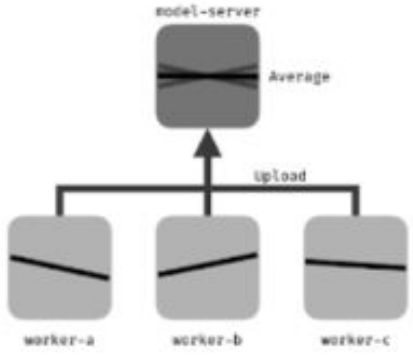
promise of greatly improving usability by powering more intelligent applications, but the sensitive nature of the data means there are risks and responsibilities to storing it in a centralized location.

We investigate a learning technique that allows users to collectively reap the benefits of shared models trained from this rich data, without the need to centrally store it. We term our approach *Federated Learning*, since the learning task is solved by a loose federation of participating devices (which we refer to as *clients*) which are coordinated by a central *server*. Each client has a local training dataset which is never uploaded to the server. Instead, each client computes an update to the current global model maintained by the server, and only this update is communicated. This is a direct application of the principle of *focused collection* or *data minimization* proposed by the 2012 White House report on privacy of consumer data [39]. Since these updates are specific to improving the current model, there is no reason to store them once they have been applied.

A principal advantage of this approach is the decoupling of model training from the need for direct access to the raw training data. Clearly, some trust of the server coordinating the training is still required. However, for applications where the training objective can be specified on the basis of data available on each client, federated learning can significantly reduce privacy and security risks by limiting the attack surface to only the device, rather than the device and the cloud.

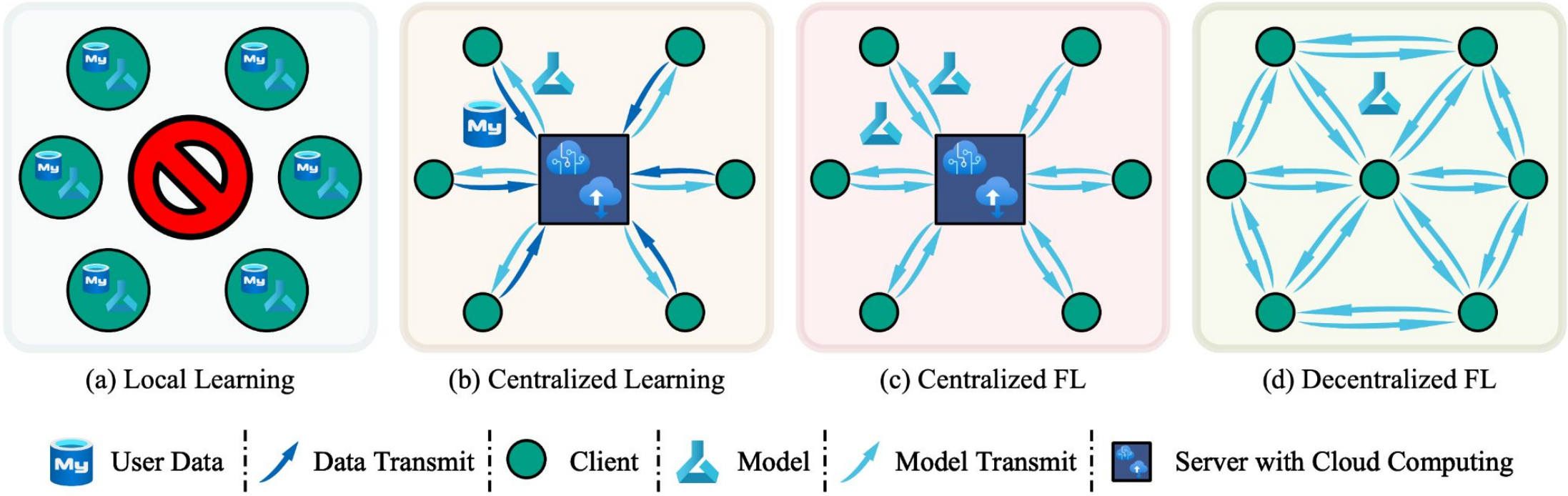
Federated Learning

How it Works?

Step 1	Step 2	Step 3	Step 4
			
Central server chooses a statistical model to be trained	Central server transmits the initial model to several nodes	Nodes train the model locally with their own data	Central server pools model results and generate one global mode without accessing any data

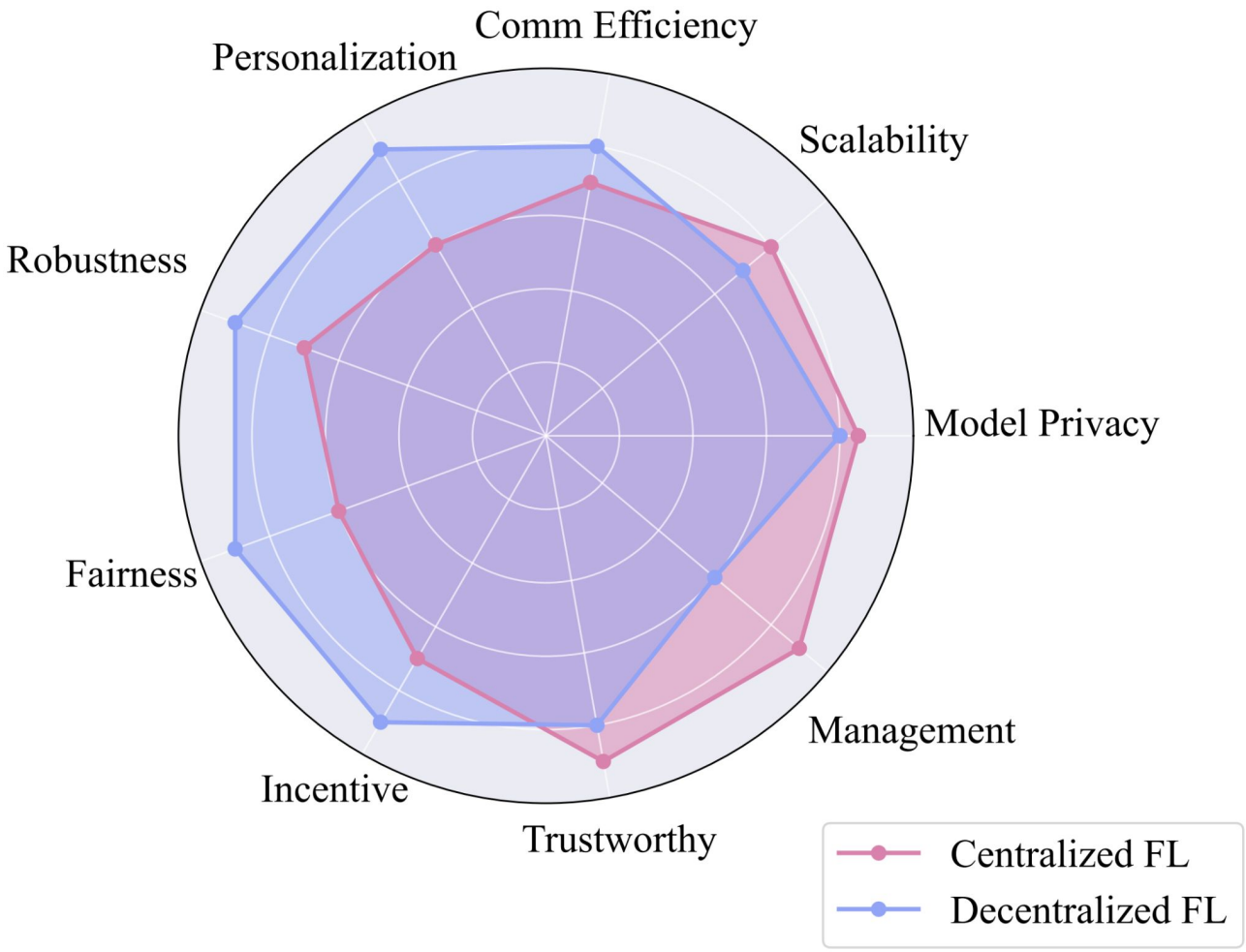
Federated Learning

Centralized vs. Decentralized FL



Federated Learning

Centralized vs. Decentralized FL



Federated Learning

Advantages

Data Privacy

- 원본 데이터가 아닌 로컬 모델의 업데이트 정보(가중치)만을 전송하므로 개인정보 보호에 우수

Communication Efficiency

- 큰 데이터가 이동하는 것이 아닌 계산 결과 업데이트 정보만 보내면 되므로 통신 비용이 훨씬 줄어듦

Heavy Computation on Client-side

- 모델 학습은 개별 클라이언트에서 진행되고, 중앙 모델은 이를 합치는 가벼운 작업만 진행

Challenges

Data Credibility

- 올바른 데이터, 품질 좋은 데이터로 학습했는지 알기 힘듦

Data Leakage

- Gradient 값으로 데이터를 복원 가능
- Deep Leakage from Gradients

Adversarial Attacks

- 사용자가 모델의 파라미터 값을 모두 볼 수 있는 whitebox 시스템이므로 일반적인 blackbox 시스템보다 공격에 취약
- Model update poisoning, data poisoning, evasion attack

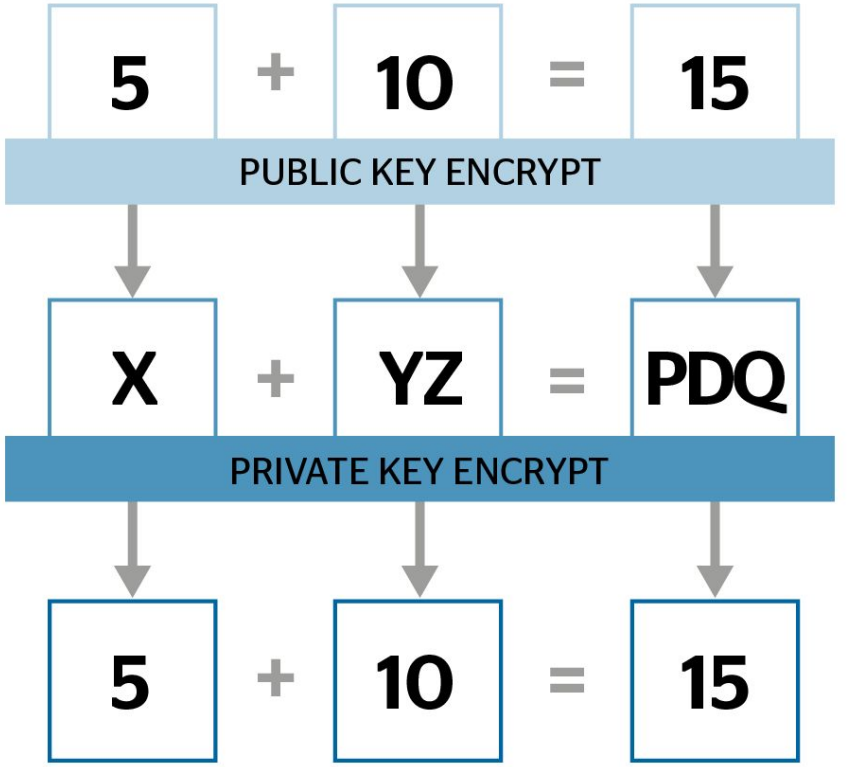
Decipher

ZKP & HE

Zero-Knowledge Proof

[What Is Zero-Knowledge Proof And Why You Should Care]

Homomorphic Encryption



[What is Homomorphic Encryption?]

Background

Motivation

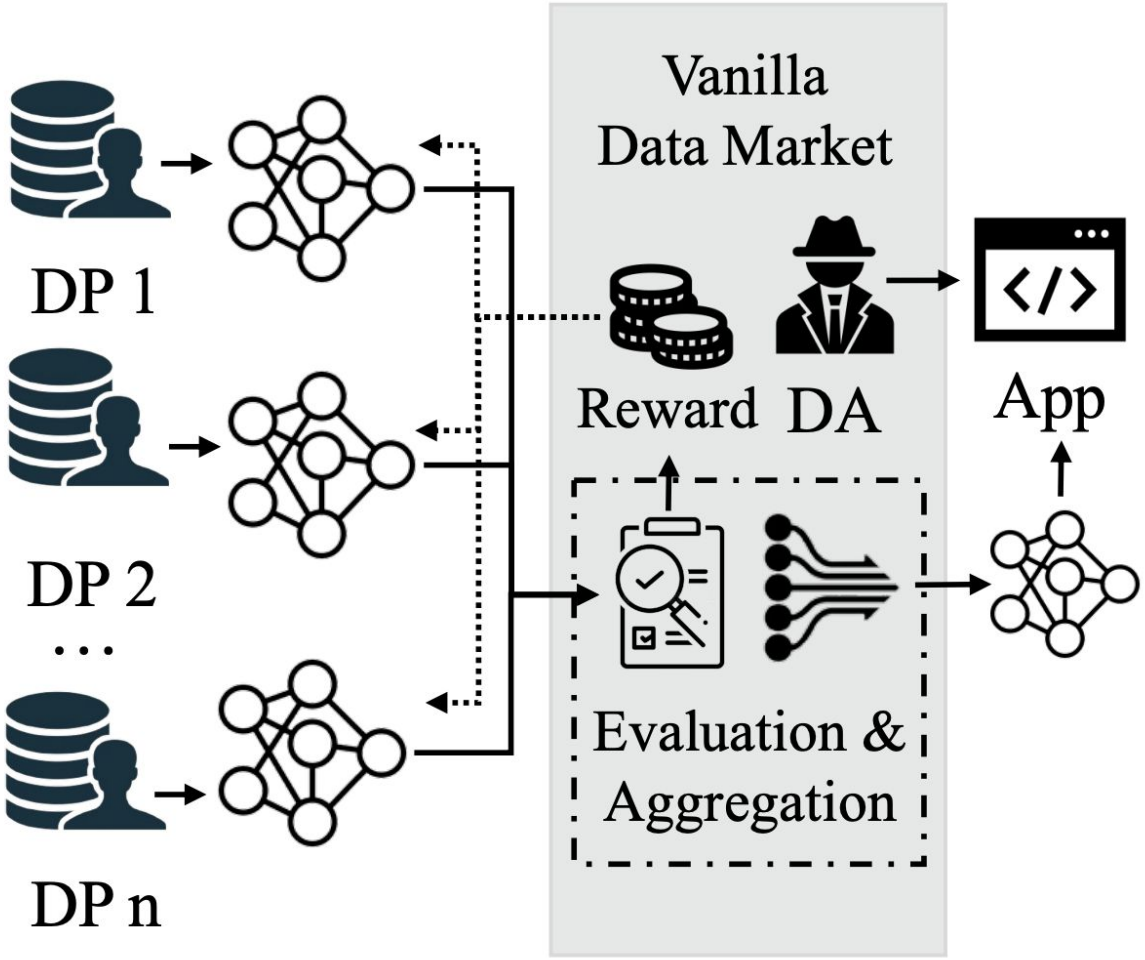
Data Marketplace

Data Problem in AI Development

- 대량의 학습 데이터 필요
- 데이터 마켓플레이스는 인터넷에 공개되지 않은 고품질의 프라이빗 데이터를 거래하는 중요한 플랫폼
- 데이터 프라이버시가 중요해지면서 원시 데이터 교환은 부적절

Federated Learning

- 연합학습은 데이터를 직접 노출하지 않으면서 데이터를 거래할 수 있는 방법
- 연합학습을 사용하는 데이터 마켓플레이스 형태
  - 로컬 학습을 진행하는 클라이언트 = 데이터 제공자 (DP)
  - DP가 제공하는 로컬 모델을 평가하고 구매하는 글로벌 모델 집계자 = 데이터 취득자 (DA)



Decipher

Data Marketplace with FL

Challenges

Data Trading Dilemma

- 기존의 연합학습에서는 DA가 DP의 로컬 모델 업데이트를 평가하기 전에 업데이트를 받지 못함
- DP는 보상을 받기 전까지 업데이트를 제공하지 않으려 함
- DA는 업데이트를 구매하기 전에 품질을 평가하고 싶어 함

Trade-off in Model Aggregation Protocols

- 비잔틴 환경에서의 연합학습을 위한 두가지 모델 집계 방법론: 클라이언트 주도 접근법, 서버 주도 접근법
- 클라이언트 주도 접근법은 클라이언트가 로컬 업데이트를 부드럽게 하는 방식
  - 정직한 다수에 의존하므로 견고성이 낮음
- 서버 주도 접근법은 고품질 데이터셋으로 로컬 업데이트를 보정
  - 과적합할 가능성이 높아 학습이 잘 안될 수 있음
- 즉, 견고성과 포괄성 사이의 트레이드오프가 존재

Lack of Verifiability for Fair Compensation

- 제출한 로컬 모델의 품질을 평가하여 공정한 보상을 주어야 함
- 이를 위해 스마트 컨트랙트로 이를 평가하도록 하는 해결방안이 제안되었지만
  - 평가하는 방식에 제한이 생김
  - 매우 느리고, 큰 비용이 발생
- 임의의 업데이트를 모두 구매하는 것이 아닌, 원하는 경우에만 구매할 수 있도록 해야함

Decipher

Data Marketplace with FL

Zero-Knowledge Proof

Ensuring Fair Transaction with ZKP

- DA가 DP로부터 로컬 모델을 받기 전에, 커밋된 집계 가중치를 사용해 로컬 모델을 충실히 집계했음을 공개적으로 증명해야 함.
- 이 증명은 로컬 모델을 공개하지 않으면서도 집계 프로토콜에 대한 신뢰있는 주장 (Argument of Knowledge)
- 결과적으로, 집계 프로토콜에 대한 신뢰 -> 공정한 보상에 대한 신뢰를 달성 가능
- zk-SNARK 는 비대화형으로 한번의 메시지만을 통해 이를 가능하게 할 수 있음

Challenges in Applying zk-SNARKs to Federated Learning

- 전체 과정에 대한 ZKP 생성은 매우 복잡하고 비현실적인 증명 생성 시간과 비용이 들 수 있음
  - 로컬 모델 평가 알고리즘이 복잡해질 수 있고, 동형 암호화된 값들에 대한 계산을 포함
  - 모델의 크기가 매우 커서, 수백만 개의 매개변수를 처리해야 하는 경우 증명 회로가 비실용적으로 커짐

martFL Overview

Solutions of martFL

Quality-Aware Model Evaluation Protocol

- 모든 로컬 모델을 무차별적으로 평가하는 프로토콜
- 동적으로 조정된 기준선을 기반으로 로컬 모델을 비공개로 평가
- 악성 로컬 모델을 정확하게 제거
- DA의 루트 데이터셋에 대한 과적합(Overfitting)을 방지
- 포괄성과 견고성 간의 트레이드오프를 제거

Efficient Verifiable Transaction Protocol

- 연합학습 과정 전체에 대한 증명이 아닌 공정한 청구를 보장하기 위한 필수적인 계산만 증명
- 증명 오버헤드가 로컬 모델 평가 알고리즘이나 모델 크기와 독립적
- 스마트 계약을 통해 명확하게 보상 청구 가능
- 프로토콜 위반 (예: ZKP 검증 실패) 시 에는 패널티 부과

Assumptions and Threat Model

Assumptions

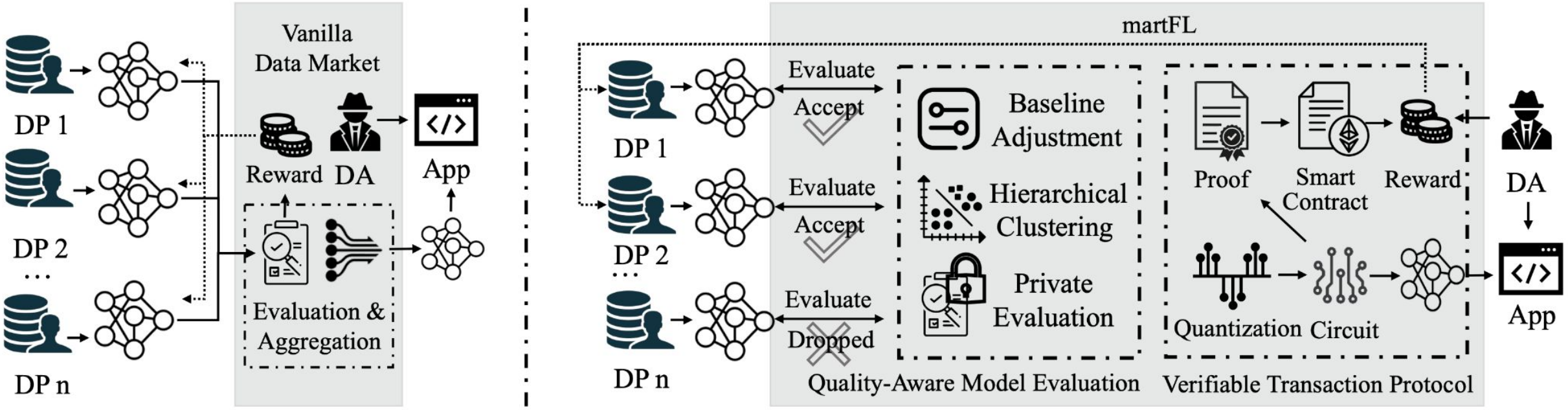
- martFL에서 사용하는 블록체인 시스템의 암호화 원리와 합의 프로토콜이 안전하다고 가정
- 블록체인이 공개 원장을 가지고 있어 스마트 계약 상태를 외부에서 확인 가능
- zk-SNARK 프로토콜이 신뢰할 수 있다고 가정
- DA가 품질 좋은 루트 데이터셋을 가질 필요 없음
  - 기존 방식은 중앙 노드에게 좋은 품질의 데이터 있어야 함

Threat Model

- 비잔틴 DP들이 임의의 로컬 모델을 제출할 수 있음
  - 학습 과정 방해
  - 보상 조작 (예: Free-rider)
- DA는 반정직적(semi-honest)으로 가정, 보상 분배를 조작할 동기가 있음

martFL Overview

Architecture



Quality-Aware Model Evaluation Protocol

Decipher

Experiment Environment

Metrics

견고성 (Robustness)

- 악성 DP의 로컬 모델이 집계에서 제외된 비율
- $$R = \frac{\text{제외된 악성 DP의 수}}{\text{전체 악성 DP의 수}} \times 100$$

포괄성 (Inclusiveness)

- 양성 DP의 로컬 모델이 집계에 포함된 비율
- $$I = \frac{\text{집계에 포함된 양성 DP의 수}}{\text{전체 양성 DP의 수}} \times 100$$

정확성 (Accuracy)

- 테스트 데이터셋에서 최종 모델의 성능
- $$A = \frac{\text{올바르게 분류된 샘플의 수}}{\text{전체 테스트 샘플의 수}} \times 100$$

Experiment Settings

Dataset

- TREC Dataset

DA의 루트 데이터셋

- 전체 클래스 라벨의 절반이 대부분을 차지는 편향된 데이터셋

DP들의 구성

- 30%는 고품질 데이터 (모든 라벨에 고르게 분포)
- 30%는 편향된 데이터셋
- 40%는 악의적 DP

악의적 DP들이 수행할 수 있는 공격

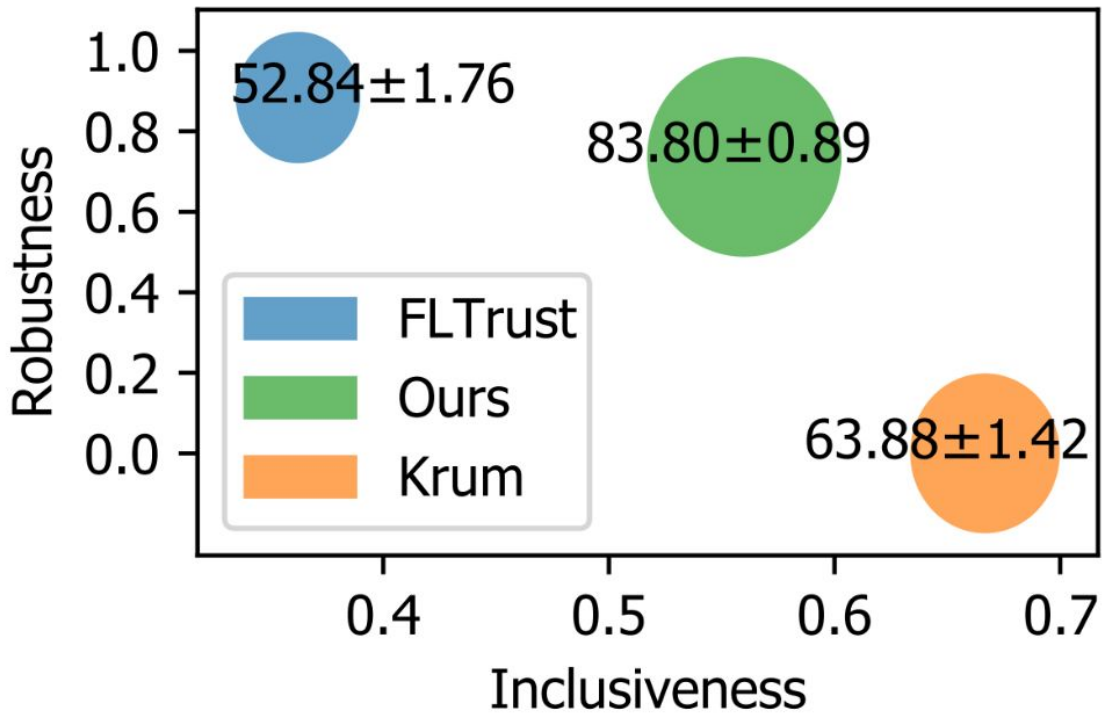
- 백도어 공격 (Targeted Attack)
- 부호 랜덤화 공격 (Untargeted Attack)

비교군

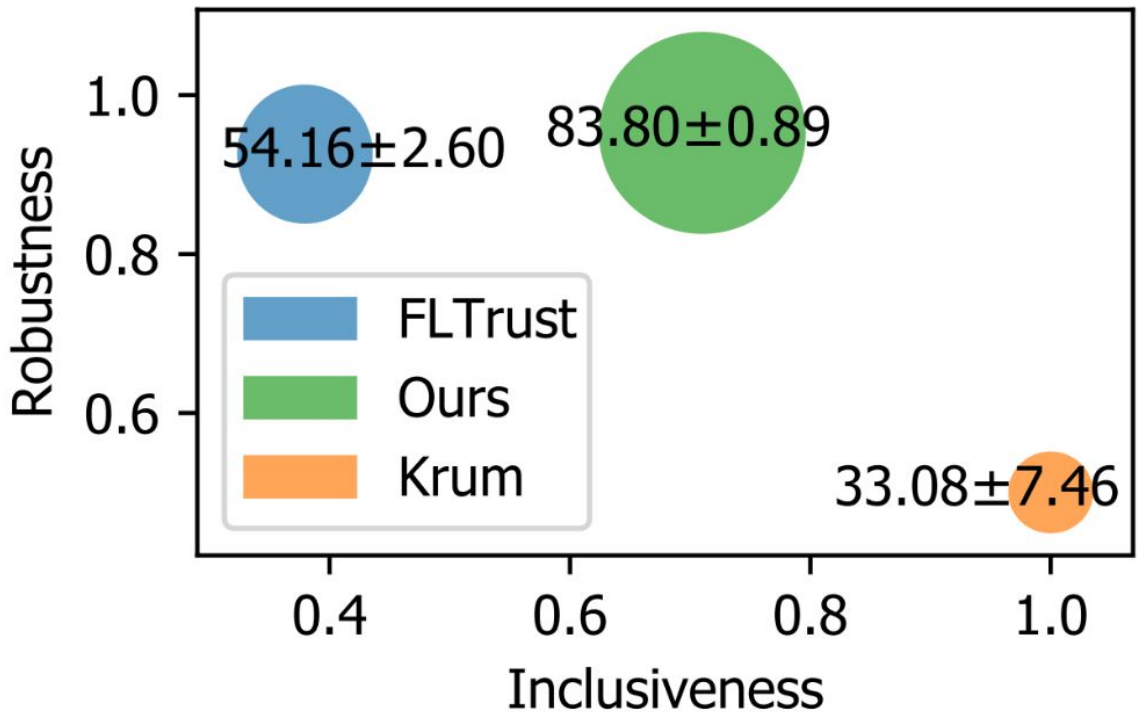
- martFL
- FLTrust (Client-driven)
- Krum (Server-driven)

Result

martFL vs FLTrust vs Krum



(a) Targeted Attack



(b) Untargeted Attack

Algorithm Breakdown

Full Pseudo-Code

Algorithm 1: Quality-Aware Model Aggregation Protocol

Inputs: The scores of local models in the t-th training epoch $\mathcal{S}^t = \{s_1^t, s_2^t, \ldots, s_n^t\}$; the DP selected as the baseline in the t-th epoch p^t; a control flag α for baseline adjustment; the ratio of randomly selected baseline candidates β; the threshold T used in hierarchical clustering; the root dataset D_0; the maximum number of clusters G.

Outputs: The aggregation weights obtained for the t-th epoch and the DP selected as the baseline for the $(t+1)$-th epoch p^{t+1}.

Function Main $(\mathcal{S}^t, p^t, \alpha, \beta, T, D_0, G)$ **:**

// Set $\mathcal{P}^t$ stores the DPs selected for aggregation; Set $\mathcal{K}^t$ stores their weights.

$\mathcal{P}^t, \mathcal{K}^t \leftarrow$ OutlierRemoval $(\mathcal{S}^t, p^t, \beta, T, G)$

// $\mathcal{M}^t$ are the plaintext models that the DA commits to purchase.

$\mathcal{M}^t \leftarrow$ ModelTrading$(\mathcal{P}^t)$

if α = **true then** $p^{t+1} \leftarrow$ BaselineAdjustment$(\mathcal{M}^t, D_0)$

else $p^{t+1} \leftarrow 0$

Function OutlierRemoval$(\mathcal{S}^t, p^t, \beta, T, G)$ **:**

$\mathcal{U} \leftarrow \{1, 2, \ldots, n\}$, $\mathcal{P}_1 \leftarrow \emptyset$, $\mathcal{P}_2 \leftarrow \emptyset$, $\mathcal{K} \leftarrow \{1.0, \ldots, 1.0\}$

// Determine the number of clusters $\hat{g}$ by Gap statistics.

for $g \leftarrow 1, 2, \ldots, G$ **do**

  $\hat{g} \leftarrow$ the minimum g such that $\text{Gap}(g) - \text{Gap}(g+1) + \sigma_{g+1} \geq 0$

$d \leftarrow \text{Max}(\mathcal{S}^t) - \text{Min}(\mathcal{S}^t)$

if $\hat{g} = 1$ **and** $d > T$ **then** $\hat{g} \leftarrow 2$

else $\mathcal{P}_1 \leftarrow \mathcal{U}$ *// Single-cluster gathered distribution.*

// K-Means returns the clusters and centroids of the scores.

$\mathcal{N}_1, C_1 \leftarrow$ K-Means$(\mathcal{S}^t, \hat{g})$

$C_{best} = \text{Max}(C_1)$ *// Centroid of the highest-score cluster.*

if $\hat{g} > 2$ **then** $\mathcal{N}_2, C_2 \leftarrow$ K-Means$(\mathcal{S}^t, 2)$ *// Re-clustering.*

else $\mathcal{N}_2 \leftarrow \mathcal{N}_1$

for $i \leftarrow 1, 2, \ldots, n$ **do**

  if $\hat{g} = 1$ **then** break

  if $i = p^t$ **or** $\mathcal{N}_1[i] = 0$ **or** $\mathcal{N}_2[i] = 0$ **then**

    $\mathcal{K}[i] \leftarrow 0.0$ *// Low-quality model.*

  else if $\mathcal{N}_1[i] = \hat{g} - 1$ **and** $(\mathcal{N}_2[i] \neq 0)$ **then**

    $\mathcal{K}[i] \leftarrow 1.0$, $\mathcal{P}_1$.add(i) *// High-quality model.*

else

  $\mathcal{K}[i] \leftarrow 1.0 - \frac{\text{Abs}(\mathcal{S}[i] - C_{best})}{\text{Max}(\text{Abs}([s_i^t - C_{best} \textbf{ for } s_i^t \textbf{ in } \mathcal{S}]))}$

  $\mathcal{P}_2$.add(i) *// Qualified but weighted model.*

if $\mathcal{P}_2 = \emptyset$ **and** $\text{Len}(\mathcal{P}_1) < 0.5 \times n$ **then**

  $\mathcal{P}_2 \leftarrow \text{RandomSample}(\mathcal{U} - \mathcal{P}_1, \beta)$

return $\mathcal{P}_1 \bigcup \mathcal{P}_2$, $\frac{\mathcal{K}}{\text{Sum}(\mathcal{K})}$

Function BaselineAdjustment$(\mathcal{M}^t, D_0)$ **:**

kp_{max} = -inf, $p^{t+1} = 0$

for i, m **in Enumerate**$(\mathcal{M}^t)$ **do**

  $kp \leftarrow$ Kappa(m, D_0)

  if $kp > kp_{max}$ **then** $kp_{max} \leftarrow kp, p^{t+1} \leftarrow i$

return p^{t+1}

Algorithm Breakdown

Inputs & Outputs

Inputs: The scores of local models in the t-th training epoch $S^t = \{s_1^t, s_2^t, \ldots, s_n^t\}$; the DP selected as the baseline in the t-th epoch p^t; a control flag α for baseline adjustment; the ratio of randomly selected baseline candidates β; the threshold T used in hierarchical clustering; the root dataset D_0; the maximum number of clusters G.

Outputs: The aggregation weights obtained for the t-th epoch and the DP selected as the baseline for the $(t+1)$-th epoch p^{t+1}.

Inputs

- S^t: t번째 훈련 에포크에서 로컬 모델들의 점수 집합 $\{s^t_1, s^t_2, \ldots, s^t_n\}$
- p^t: t번째 에포크에서 기준선으로 선택된 DP
- α: 기준선 조정 여부를 나타내는 제어 플래그
- β: 무작위로 선택된 기준선 후보의 비율
- T: 계층적 클러스터링에 사용되는 임계값
- D_0: DA의 루트 데이터셋
- G: 최대 클러스터 수

Outputs

- t번째 에포크에서 얻은 집계 가중치
- (t+1)번째 에포크에서 기준선으로 선택된 DP p^{t+1}

Algorithm Breakdown

Main Flow

Function Main $(\mathcal{S}^t, p^t, \alpha, \beta, T, D_0, G)$ **:**

// Set $\mathcal{P}^t$ *stores the DPs selected for aggregation; Set* $\mathcal{K}^t$ *stores their weights.*

$\mathcal{P}^t, \mathcal{K}^t \leftarrow$ OutlierRemoval $(\mathcal{S}^t, p^t, \beta, T, G)$

// $\mathcal{M}^t$ *are the plaintext models that the DA commits to purchase.*

$\mathcal{M}^t \leftarrow$ ModelTrading$(\mathcal{P}^t)$

if α = **true then** $p^{t+1} \leftarrow$ BaselineAdjustment$(\mathcal{M}^t, D_0)$

else $p^{t+1} \leftarrow 0$

Prepare Baseline Model

- DA는 루트 데이터셋을 사용해 기준 모델을 생성
- 각 에포크에서 로컬 모델을 평가하는 참조점으로 사용

Outlier Removal

- DP들이 제출한 로컬 모델을 점수에 따라 클러스터링
- 저품질 모델을 식별해 제거 및 고품질 모델 선택

Model Trading via Verifiable Transaction Protocol

- 검증 가능한 거래 프로토콜을 통해 모델 품질에 따른 공정한 보상을 보장

Baseline Dynamic Adjustment

- 다음 에포크 시작 전에, 수집된 고품질 데이터를 반영해 기준선을 동적으로 조정
- 편향된 루트 데이터셋 문제 해결

Homomorphic Encryption

- DA가 로컬 모델을 구매하기로 커밋하기 전에는 평문 모델을 얻을 수 없도록 보안 유지

Algorithm Breakdown

Hierarchical Clustering for Outlier Removal

Function OutlierRemoval$(\mathcal{S}^t, p^t, \beta, T, G)$:
$\mathcal{U} \leftarrow \{1, 2, \ldots, n\}$, $\mathcal{P}_1 \leftarrow \emptyset$, $\mathcal{P}_2 \leftarrow \emptyset$, $\mathcal{K} \leftarrow \{1.0, \ldots, 1.0\}$
// Determine the number of clusters $\hat{g}$ by Gap statistics.
for $g \leftarrow 1, 2, \ldots, G$ **do**
  $\hat{g} \leftarrow$ the minimum g such that $\text{Gap}(g) - \text{Gap}(g+1) + \sigma_{g+1} \geq 0$
$d \leftarrow \text{Max}(\mathcal{S}^t) - \text{Min}(\mathcal{S}^t)$
if $\hat{g} = 1$ **and** $d > T$ **then** $\hat{g} \leftarrow 2$
else $\mathcal{P}_1 \leftarrow \mathcal{U}$ *// Single-cluster gathered distribution.*
// K-Means returns the clusters and centroids of the scores.
$\mathcal{N}_1, C_1 \leftarrow \text{K-Means}(\mathcal{S}^t, \hat{g})$
$C_{best} = \text{Max}(C_1)$ *// Centroid of the highest-score cluster.*
if $\hat{g} > 2$ **then** $\mathcal{N}_2, C_2 \leftarrow \text{K-Means}(\mathcal{S}^t, 2)$ *// Re-clustering.*
else $\mathcal{N}_2 \leftarrow \mathcal{N}_1$
for $i \leftarrow 1, 2, \ldots, n$ **do**
  if $\hat{g} = 1$ **then** break
  if $i = p^t$ **or** $\mathcal{N}_1[i] = 0$ **or** $\mathcal{N}_2[i] = 0$ **then**
    $\mathcal{K}[i] \leftarrow 0.0$ *// Low-quality model.*
  else if $\mathcal{N}_1[i] = \hat{g} - 1$ **and** $(\mathcal{N}_2[i] \neq 0)$ **then**
    $\mathcal{K}[i] \leftarrow 1.0$, $\mathcal{P}_1$.add(i) *// High-quality model.*
  else
    $\mathcal{K}[i] \leftarrow 1.0 - \frac{\text{Abs}(\mathcal{S}[i] - C_{best})}{\text{Max}(\text{Abs}([s_i^t - C_{best} \textbf{ for } s_i^t \textbf{ in } \mathcal{S}]))}$
    $\mathcal{P}_2$.add(i) *// Qualified but weighted model.*
if $\mathcal{P}_2 = \emptyset$ **and** $\text{Len}(\mathcal{P}_1) < 0.5 \times n$ **then**
  $\mathcal{P}_2 \leftarrow \text{RandomSample}(\mathcal{U} - \mathcal{P}_1, \beta)$
return $\mathcal{P}_1 \bigcup \mathcal{P}_2$, $\frac{\mathcal{K}}{\text{Sum}(\mathcal{K})}$

1. Local Model Scoring

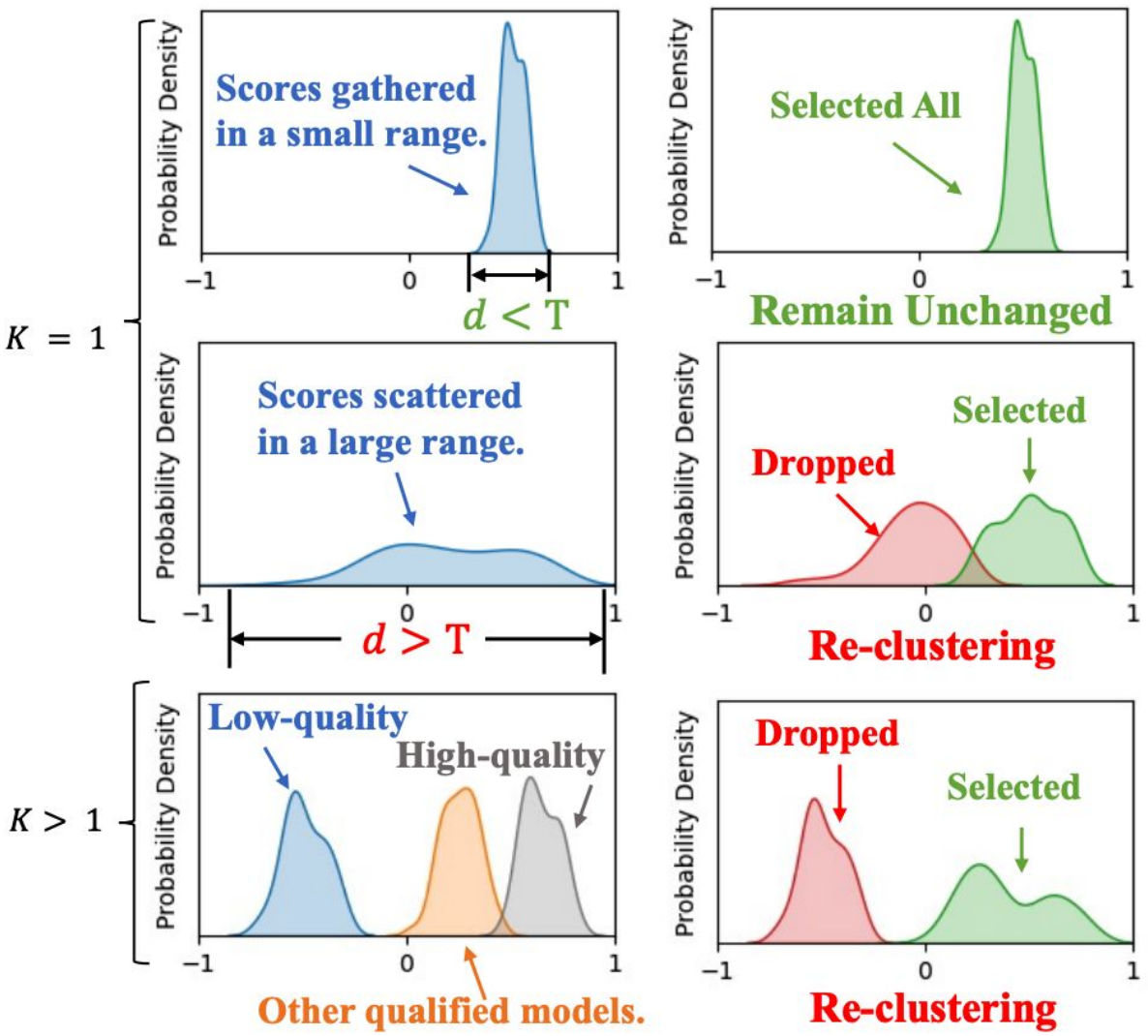
- 각 DP가 제출한 로컬 모델의 업데이트(u_i)를 계산하고, 이를 DA의 기준 모델 업데이트(u_g)와 비교하여 코사인 유사도로 점수 산출
- 이 점수는 해당 DP의 로컬 모델이 얼마나 고품질인지

$$s_i^t = \text{Cosine}(u_g^t, u_i^t) = \frac{u_g^t \cdot u_i^t}{||u_g^t|| \cdot ||u_i^t||}$$

-

Algorithm Breakdown

Hierarchical Clustering for Outlier Removal



2. Model Classification via Clustering

- **Gap-Statistics** 알고리즘을 사용하여 최적의 클러스터 수 ($\hat{g}$) 결정
- K-Means 알고리즘을 사용해 첫 번째 계층 클러스터링 수행
- 모델 점수 분포에 따른 세 가지 유형의 클러스터링 결과
  - **단일 클러스터 집중 분포**: 점수가 집중되어 있고, 모든 DP가 유사한 품질을 가진 경우
    -> 모든 모델을 고품질 모델 집합(P1)에 포함
  - **단일 클러스터 분산 분포**: 점수가 넓게 분산되어 있을 경우
    -> 두번째 계층 클러스터링 수행하여 고품질과 저품질 클러스터로 분류
  - **다중 클러스터 분포**: 점수가 여러 클러스터로

Decipher

Algorithm Breakdown

Hierarchical Clustering for Outlier Removal

Function OutlierRemoval$(\mathcal{S}^t, p^t, \beta, T, G)$ **:**

$\mathcal{U} \leftarrow \{1, 2, \ldots, n\}$, $\mathcal{P}_1 \leftarrow \emptyset$, $\mathcal{P}_2 \leftarrow \emptyset$, $\mathcal{K} \leftarrow \{1.0, \ldots, 1.0\}$

// Determine the number of clusters $\hat{g}$ by Gap statistics.

for $g \leftarrow 1, 2, \ldots, G$ **do**

  $\hat{g} \leftarrow$ the minimum g such that $\text{Gap}(g) - \text{Gap}(g+1) + \sigma_{g+1} \geq 0$

$d \leftarrow \text{Max}(\mathcal{S}^t) - \text{Min}(\mathcal{S}^t)$

if $\hat{g} = 1$ **and** $d > T$ **then** $\hat{g} \leftarrow 2$

else $\mathcal{P}_1 \leftarrow \mathcal{U}$ *// Single-cluster gathered distribution.*

// K-Means returns the clusters and centroids of the scores.

$\mathcal{N}_1, C_1 \leftarrow \text{K-Means}(\mathcal{S}^t, \hat{g})$

$C_{best} = \text{Max}(C_1)$ *// Centroid of the highest-score cluster.*

if $\hat{g} > 2$ **then** $\mathcal{N}_2, C_2 \leftarrow \text{K-Means}(\mathcal{S}^t, 2)$ *// Re-clustering.*

else $\mathcal{N}_2 \leftarrow \mathcal{N}_1$

for $i \leftarrow 1, 2, \ldots, n$ **do**

  if $\hat{g} = 1$ **then** break

  if $i = p^t$ **or** $\mathcal{N}_1[i] = 0$ **or** $\mathcal{N}_2[i] = 0$ **then**

    $\mathcal{K}[i] \leftarrow 0.0$ *// Low-quality model.*

  else if $\mathcal{N}_1[i] = \hat{g} - 1$ **and** $(\mathcal{N}_2[i] \neq 0)$ **then**

    $\mathcal{K}[i] \leftarrow 1.0$, $\mathcal{P}_1$.add(i) *// High-quality model.*

else

  $\mathcal{K}[i] \leftarrow 1.0 - \frac{\text{Abs}(\mathcal{S}[i] - C_{best})}{\text{Max}(\text{Abs}([s_i^t - C_{best} \textbf{ for } s_i^t \textbf{ in } \mathcal{S}]))}$

  $\mathcal{P}_2$.add(i) *// Qualified but weighted model.*

if $\mathcal{P}_2 = \emptyset$ **and** $\text{Len}(\mathcal{P}_1) < 0.5 \times n$ **then**

  $\mathcal{P}_2 \leftarrow \text{RandomSample}(\mathcal{U} - \mathcal{P}_1, \beta)$

return $\mathcal{P}_1 \bigcup \mathcal{P}_2$, $\frac{\mathcal{K}}{\text{Sum}(\mathcal{K})}$

3. Final Model Selection and Weight Assignment

- P1에 포함된 고품질 모델들과, P2에서 무작위로 선택된 소수의 DP들이 최종적으로 집계에 선택됨
- DA는 이 모델들을 구매하기 위해 커밋하며, 선택된 DP들은 이후 안전하게 평문 로컬 모델을 DA에 전달 가능

Algorithm Breakdown

Dynamic Baseline Adjustment

Function BaselineAdjustment$(\mathcal{M}^t, D_0)$ **:**
kp_{max} = -inf, $p^{t+1} = 0$
for i, m **in Enumerate**$(\mathcal{M}^t)$ **do**
  $kp \leftarrow \text{Kappa}(m, D_0)$
  if $kp > kp_{max}$ **then** $kp_{max} \leftarrow kp, p^{t+1} \leftarrow i$
return p^{t+1}

$$K = \frac{P_{agree} - P_{chance}}{1 - P_{chance}}$$

P_{agree} = Proportion of trials in which judges agree

P_{chance} = Proportion of trials in which agreement would be expected due to chance

Overfitting Prevention

- martFL은 DA가 루트 데이터셋에 과적합하지 않도록 동적으로 기준선을 조정 (Inclusiveness)

Kappa Coefficient Evaluation

- DA는 각 로컬 모델을 루트 데이터셋에서 평가하여 카파 계수 계산
- 높은 카파 계수를 가진 DP들을 우선 DP로 선정

Preferred DP Selection

- 가장 높은 카파 계수를 가진 DP를 우선적으로 선택
- 이 우선 DP들은 다음 에포크에서 새로운 기준선으로 사용될 모델들을 제공

Information Confidentiality

- DA는 우선 DP들이 로컬 모델을 커밋할 때까지 이 정보를 공개하지 않음, 보안을 유지하기 위함

Decipher

Algorithm Breakdown

CKKS Homomorphic Encryption

목표

- DA가 로컬 모델을 구매하기 전에 평문 형태의 모델 업데이트를 얻지 못하도록 보장

암호화 과정

- DA는 기준 업데이트 u□g를 동형 암호화하여 DP에게 전달
- DP는 암호화된 u□g를 사용해 자신의 로컬 업데이트를 곱하고, 그 결과를 DA에 반환
- DA는 이를 복호화하여 로컬 모델 평가에 사용

Model Commitment before Evaluation

목표

- DP들이 모델 평가 후 다른 모델 업데이트를 제출하는 것을 방지

커밋 과정

- DP들은 모델 평가 전에 자신의 모델 업데이트를 커밋
- 커밋된 업데이트는 이후 모델 거래의 정확성을 보장하는 데 사용됨

Verifiable Transaction Protocol

Overview

Zero-Knowledge Proving System

Objective

- DA가 DP들이 제출한 로컬 모델을 공개하지 않고도, 글로벌 모델을 주장된 가중치에 따라 충실하게 집계했음을 증명하는 것
- 결과적으로 DA에게 공정한 보상을 줄 수 있도록 하는 것
- 전체 과정이 아닌 일부만을 증명하면서도 위의 목적을 달성할 수 있도록 하는 것

Strategy

- DA는 복잡한 모델 평가 알고리즘 대신, 로컬 모델의 합산 계산만을 증명하여 증명 복잡성을 줄임
- DA는 영지식 증명 방식을 사용해, DP들이 제출한 모델을 공개하지 않고도 집계 과정의 정확성을 증명
- 정해진 수의 파라미터만 증명하도록 설계된 검증 가능한 샘플링 방법을 사용하여, 모델 크기에 관계없이 증명 복잡성을 최소화

Payment Protocol Based on Smart Contract

Objective

- DA와 DP들이 보상과 평문 로컬 모델을 안전하게 교환할 수 있도록 지원하는 것

Transaction Process

- **Prepare Phase**: DA는 스마트 계약에 집계 가중치와 DP들을 커밋하고, 보상을 할당. DP들은 오프체인으로 평문 로컬 모델을 제출
- **Verify Phase**: DA는 모델 집계의 무결성을 검증하기 위해 무작위 샘플링을 수행하고, 검증에 필요한 증명을 제공. 검증 실패 시, DA는 보증금을 잃게 됨
- **On-Chain Verification**: DP들은 스마트 계약의 Verify 기능을 통해 모델 집계 증명의 정확성을 검증

Zero-Knowledge Proving System

Process

Settings

- Prover: DA, Verifier: DP
- $\mathscr{A}$: 로컬 모델 집계 $W_g^t = W_g^{t-1} + K^t U^t$
- $\mathbf{W_g^t}$: 에포크 t에서의 글로벌 모델
- $\mathbf{K^t} = [k_1^t, k_2^t, \ldots, k_n^t]$: DA가 주장하는 집계된 가중치
- $\mathbf{U^t} = [u_1^t, u_2^t, \ldots, u_n^t]$: DP가 제출한 로컬 모델

C ← Compile($\mathscr{A}$)

- **Quantization**: Public Input X^t와 Private Witness W^t을 유한 필드에서 각각 $\mathbb{X}^t = \{\mathbb{W}_g^t, \mathbb{W}_g^{t-1}, \mathbb{K}^t\}$, $\mathbb{W}^t = \{\mathbb{U}^t\}$로 양자화
- 집계 알고리즘 $\mathscr{A}$를 회로 $\mathbb{C}$로 컴파일

(pk, vk) ← Setup(1^λ ,$\mathbb{C}$)

- **Trusted Setup**: 보안 매개변수 λ와 회로 $\mathbb{C}$를 기반으로 신뢰할 수 있는 제3자가 무작위로 증명 키 pk와 검증 키 vk를 생성
- 증명 키 pk는 DA, 검증 키 vk는 DP들에게 제공

(cm^t, W_g^t, π^t) ← Prove(X^t , W^t , r^t , pk, $\mathbb{C}$)

- 랜덤 오프닝 r^t에 대해 증명자는 $\mathbb{W}^t$를 커밋 $cm^t = \text{Commit}(U^t, r^t)$
- 양자화된 글로벌 모델 W^{tg}으로부터 증명 π^t을 생성
- 증명자는 커밋값 cm^t, 증명 π^t, 그리고 W_g^t을 DP들에게 공개

{1, 0} ← Verify($\mathbb{X}^t$, vk, π^t , cmt)

- Public Verifier들(DP들)은 검증 키 vk, 공공 입력 $\mathbb{X}^t$, 커밋 cm^t, 증명 π^t를 사용하여 회로 $\mathbb{C}$에서의 계산을 검증
- DA가 글로벌 모델을 충실히 집계했다면, 증명이 Accept 될 것이고, 아니라면 Reject

Zero-Knowledge Proving System

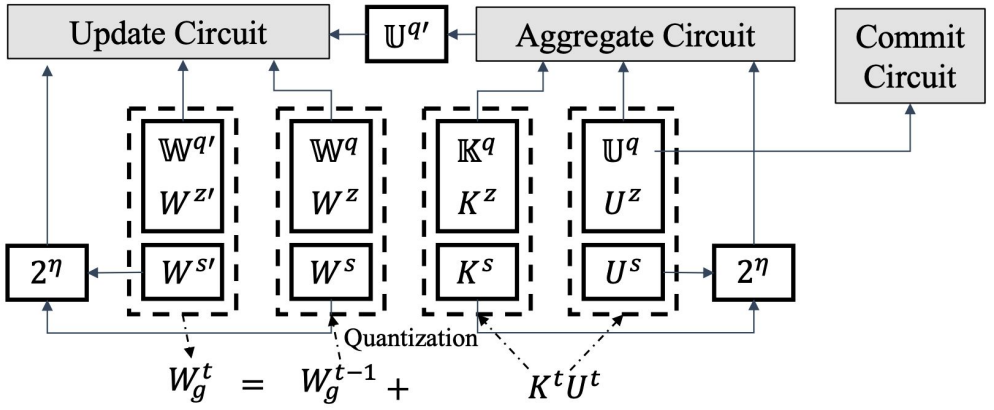
Circuit Design

Quantization Circuit

- 부동소수점 값을 부호 없는 정수에 매핑
- 부동소수점 범위에 ϵ 만큼을 추가하여 정밀도 손실로 인한 오버플로를 방지

Commit Circuit

- 비밀 변수 $\mathbb{U}^t$를 공개하지 않도록 랜덤 오프닝 r^t에 커밋
- POSEIDON 해시함수 사용
- 하지만 모든 모델 파라미터에 대해 커밋하면 너무 많은 Constraint가 생김

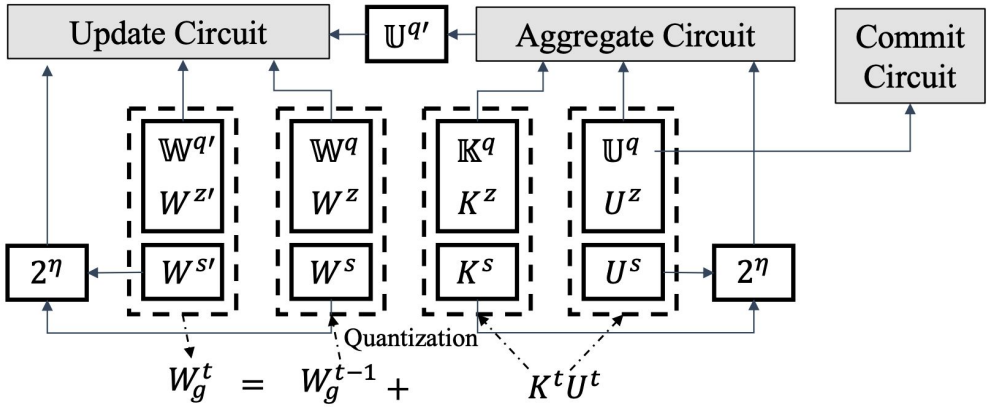


Zero-Knowledge Proving System

Circuit Design

Aggregation Circuit Design

- 양자화된 형태에서 $\mathbb{U}^{t'}=\mathbb{K}^t\mathbb{U}^t$를 계산하고 역양자화하는 회로



Equation (3):

- DP들이 제공한 모델 업데이트 $\mathbb{U}^t$를 각 가중치 $\mathbb{K}^t$와 곱하여 집계된 모델 업데이트 $\mathbb{U}^{t'}$를 계산
- 집계된 결과 $\mathbb{U}^{t'}$를 다시 부동 소수점 값으로 변환하기 위해 역양자화(de-quantization)를 수행

$$U^{s'}(\mathbb{U}_{i,j}^{q'} - U^{z'}) = \sum_{k=1}^{n} K^s(\mathbb{K}_{i,k}^{q} - K^z)U^s(\mathbb{U}_{k,j}^{q} - U^z) \qquad (3)$$

Equation (4):

- 식 (3)에서 나온 결과를 처리하는 과정
- 양자화된 값에서 계산을 수행할 때 발생할 수 있는 오차를 최소화하고, 정확성을 유지하기 위한 조정 작업
- 부호 없는 정수만을 사용하여 계산 순서를 재정렬하고, 나눗셈으로 발생하는 오차를 보정하기 위해 나머지 값 $\mathbb{R}^a$를 도입하여 계산 결과의 정확성을 보장

$$2^{\eta}\mathbb{U}_{i,j}^{q'} = \mathbb{R}_{i,j}^{a} + 2^{\eta}U^{z'} + \left\lfloor 2^{\eta}\frac{K^sU^s}{U^{s'}}\left(M_1 + M_4 - M_2 - M_3\right)\right\rfloor$$

$$\text{s.t. } M_1 = \sum_{k=1}^{n}\mathbb{K}_{i,k}^{q}\mathbb{U}_{k,j}^{q}, M_2 = U^z\sum_{k=1}^{n}\mathbb{K}_{i,k}^{q}, \qquad (4)$$

$$M_3 = K^z\sum_{k=1}^{n}\mathbb{U}_{k,j}^{q}, M_4 = nK^zU^z$$

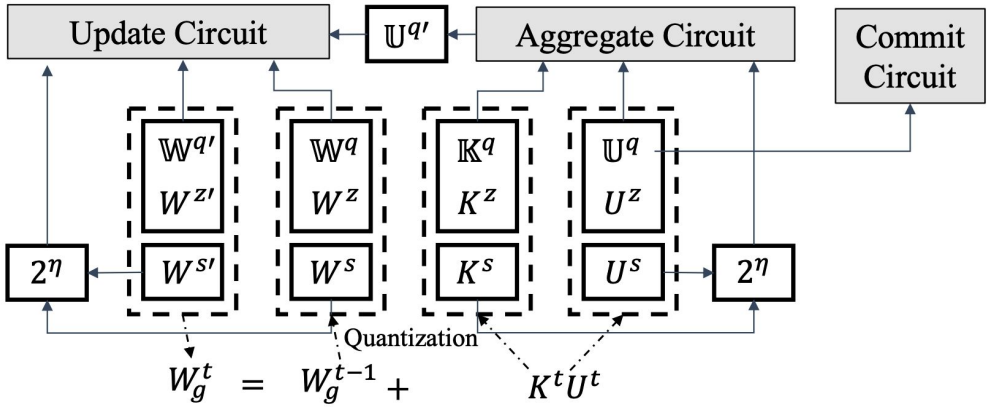
Decipher

Zero-Knowledge Proving System

Circuit Design

Update Circuit Design

- 식 (5)를 사용하여 역양자화된 $W^{t}_{g}=W^{t-1}_{g}+U^{t'}$ 업데이트 회로 설계
- 식 (6)을 사용하여 음수를 제거하고 나머지 $\mathbb{R}^{u}$ 를 사용해서 나눗셈의 정확성 보장



$$W^{s'}(\mathbb{W}^{q'}_{i,j} - W^{z'}) = W^{s}(\mathbb{W}^{q}_{i,j} - W^{z}) + U^{s'}(\mathbb{U}^{q'}_{i,j} - U^{z'}), \quad (5)$$

$$2^{\eta}\mathbb{W}^{q'}_{i,j} = \mathbb{R}^{u}_{i,j} + 2^{\eta}W^{z'} + \left\lfloor 2^{\eta}\left(N_1 + N_3 - N_2 - N_4\right)\right\rfloor$$
$$\text{s.t. } M_1 = \frac{W^{s}}{W^{s'}}\mathbb{W}^{q}_{i,j}, N_2 = \frac{W^{s}}{W^{s'}}W^{z}, \quad (6)$$
$$N_3 = \frac{U^{s'}}{W^{s'}}\mathbb{U}^{q'}_{i,j}, N_4 = \frac{U^{s'}}{W^{s'}}U^{z'}$$

Decipher

Zero-Knowledge Proving System

Verifiable Sampling

Objectives

- 모든 모델 파라미터에 대해 커밋하면 너무 많은 Constraint가 생김
- 커밋 회로, 집계 회로, 업데이트 회로의 제약 조건 복잡도를 낮춤

How It Works

- 모델 파라미터 수 m중 일부 c개를 무작위로 선택하여 검증
- 선택된 샘플들에 대한 올바른 증명을 제공하면 전체 파라미터가 올바르게 계산된 것으로 간주

Adjustment of Inputs for Proof

- 무작위 샘플링 후, 증명 스킴의 Public Input과 Private Witness을 샘플링 결과에 맞게 조정
- DA는 샘플링 전에 전체 모델 W_g^t를 공개해야 함

Random Sampling Process

- 각 학습 에포크에서 각 DP는 공개적으로 암호화된 논스를 게시
- DA는 VDF(verifiable delay function)을 사용해 무작위 샘플 파라미터 인덱스를 선택
- VDF는 마지막에 논스를 게시하는 논스를 조절하여 bias를 주는 것 방지

Trading Smart Contract

Algorithm Breakdown

Prepare Phase

- DA는 집계 가중치 K^t와 DP들을 스마트 계약에 커밋
- DA는 DP들에게 지급될 보상 vDPs와 벌금 vDA를 예치
- DP들은 안전하게 오프체인으로 로컬 모델을 제출
- DA는 검증 가능한 증명을 생성하고, 공용 입력을 커밋

Verify Phase

- DA는 검증 가능한 무작위 샘플링을 수행하고 무작위성 증명(π^t_{vdf})을 제공
- DA는 무작위 시드 s^t_{vdf}에 따라 입력을 조정하고, 최종 모델 집계 증명 π^t_{agg}을 생성
- 생성된 증명은 거래 스마트 계약에 업로드되어 DP들이 온체인 Verify 함수로 검증

On-Chain Verification Procedure

- Verify 함수는 π^t_{agg}의 정확성을 검증
- 커밋된 검증 키 vk, 양자화된 공용 입력 $\mathbb{X}^{t,c}$, 증명 π^t_{agg}을 입력으로 받음
- Groth16 프로토콜에 기반

Algorithm 2: The Trading Smart Contract

```
PreparePhase() :
    commit K^t, addrs # addrs identify selected DPs in current epoch
    v_DPs, v_DA = Deposit(msg.value) # DA deposits (reward, penalty)
    R_DPs := Allocate(v_DPs, K^t) # allocate DPs reward based on K^t
    U^t := Submission(addrs) # DPs submit local models off-chain
    W_g^t := Aggregate(W_g^{t-1}, K^t, U^t) # DA generates proof off-chain
    commit W_g^t, W_g^{t-1}, K^t # DA commits public inputs
VerifyPhase() :
    DA performs verifiable sampling offline and publishes s^t_vdf, π^t_vdf
    DA adjusts W_g^{t,c}, W_g^{t-1,c} and U^{t,c} based on R^t_vdf
    DA generates π^t_agg :=Prove(X^{t,c}, W^{t,c}, pk, C) offline
    DA publishes the proof π^t_agg on-chain
    DPs invokes verification v^t := Verify(vk, X^{t,c}, π^t_agg)
    if v^t = false :
        distribute both the security deposit v_DA the award v_DPs to DPs
    else : distribute v_DPs to DPs and return v_DA to DA
Function Verify(vk, X^{t,c}, π^t_agg) :
    s := Σ_{i=0}^{Len(X^{t,c})-1} ScalarMul(vk.γ_abc[i + 1], X^{t,c}[i])
    s :=Addition(s, vk.γ_abc[0])
    p_1 := π^t_agg.a, Negate(s), Negate(π^t_agg.c), Negate(vk.α)
    p_2 := π^t_agg.b, vk.γ, vk.δ, vk.β
    return PairingCheck(p_1, p_2)
```

Evaluation

Decipher

Experimental Setup

Environment

- 두대의 Linux 서버
- Intel Xeon Gold 6348 CPU와 NVIDIA RTX A100 GPU
- Pytorch, CKKS, Ethereum Testnet

Dataset & Model

- 이미지 분류: FMNIST, CIFAR
- 텍스트 분류: TREC, AGNEWS
- LeNet, TextCNN, CNN

Comparison

- martFL
- Server-Driven: FLTrust, CFFL
- Client-Driven: FedAvg, RFFL, Krum, Median

The Adversary

- **Untargeted Attack**: Sign-randomizing 공격과 Free-rider 공격
- **Targeted Attack**: Label-flipping 공격과 Backdoor 공격
- **Sybil Attack**: 여러 클라이언트를 생성하여 동일한 손상된 모델 제출

Evaluation Metrics

- **MTA (Main Task Accuracy)**: 훈련된 모델의 분류 정확도를 측정, 높을수록 모델의 성능이 우수함을 의미
- **ASR (Attack Success Rate)**: 타겟 공격에서 오염된 샘플이 목표 클래스로 예측되는 비율을 측정, 낮을수록 모델이 공격에 강함을 의미
- **DAC (Data Acquisition Cost)**: 글로벌 모델을 훈련하기 위해 각 에포크에서 DA가 획득해야 하는 로컬 모델의 평균 비율, 낮을수록 좋음

Evaluation Results

Summary

Accuracy

- **Biased Root Dataset**
  - DA의 루트 데이터셋이 편향되어 있을 때도 높은 MTA를 유지
  - 편향된 루트 데이터셋에서도 더 많은 고품질 DP를 포함하여 모델 성능을 향상
- **Unbiased Root Dataset**
  - martFL은 가장 낮은 DAC으로 가장 높은 MTA를 유지

Robustness

- 대부분의 경우 가장 높은 MTA와 가장 낮은 ASR을 달성
- 높은 비율의 악성 DP가 존재할 때도 유효

Accuracy Loss by Quantization

- 글로벌 모델의 MTA에 거의 영향을 미치지 않음

System Overhead (Cryptography and Smart Contract Costs)

- 로컬 모델 평가 시 암호화 오버헤드가 있지만, 이로 인한 추가 지연은 매우 적음
- 스마트 계약을 통해 거래를 실행할 때 발생하는 가스 비용은 매우 낮음
- 모든 주요 함수의 실행 시간은 0.1초 미만으로 유지

Evaluation Results

Biased Root Dataset

Dataset	Biased Ratio	20%		30%		40%	
	Metric	MTA	DAC	MTA	DAC	MTA	DAC
TREC	CFFL	76.87 ± 6.87	100.00	81.53 ± 0.90	100.00	79.07 ± 3.24	100.00
	FLTrust	67.40 ± 4.76	36.52	72.60 ± 1.07	46.47	71.73 ± 1.09	40.15
	Ours	**88.80 ± 1.72**	53.63	**87.53 ± 1.15**	53.88	**87.20 ± 0.49**	51.79
AGNEWS	CFFL	44.09 ± 1.95	100.00	43.39 ± 1.57	100.00	45.58 ± 1.46	100.00
	FLTrust	44.09 ± 1.43	11.52	45.19 ± 0.39	10.35	43.65 ± 0.90	11.89
	Ours	**79.71 ± 2.15**	36.30	**75.89 ± 1.30**	38.99	**78.04 ± 1.08**	41.15
FMNIST	CFFL	**88.37 ± 0.55**	100.00	88.48 ± 0.25	100.00	**88.02 ± 0.32**	100.00
	FLTrust	87.33 ± 0.48	32.64	87.26 ± 0.38	33.57	87.28± 0.39	46.04
	Ours	88.22 ± 0.26	35.14	**88.88 ± 0.27**	30.06	87.71 ± 0.43	39.60
CIFAR	CFFL	63.34 ± 0.22	100.00	62.38 ± 0.33	100.00	60.85 ± 0.80	100.00
	FLTrust	10.00 ± 0.00	7.59	11.42 ± 1.00	10.79	14.25 ± 1.99	1.30
	Ours	**64.24 ± 0.06**	53.63	**63.79 ± 0.28**	53.88	**62.60 ± 0.37**	51.79

Table 1: MTA (%) and DAC (%) when the DA possesses a biased root dataset.

Evaluation Results

Unbiased Root Dataset

Dataset	TREC		AGNEWS		FMNIST		CIFAR	
Metric	MTA	DAC	MTA	DAC	MTA	DAC	MTA	DAC
CFFL	85.47 ± 0.68	100.00	78.79 ± 1.03	100.00	89.22 ± 0.15	100.00	65.38 ± 0.50	100.00
FLTrust	87.40 ± 0.71	46.65	80.94 ± 1.26	66.11	89.40 ± 0.20	51.62	**70.66 ± 0.45**	39.44
Ours	**87.67 ± 0.57**	44.38	**83.35 ± 1.54**	65.61	**89.88 ± 0.15**	45.27	70.28 ± 0.27	34.89

Table 2: MTA (%) and DAC (%) when the DA has an unbiased root dataset.

Decipher

Evaluation Results

Accuracy Loss by Quantization

Dataset	MTA		F1	
	QT	Δ*QT*	*QT*	Δ*QT*
TREC	88.60 ± 0.28	-0.93 ↑	87.58 ± 0.37	-0.65 ↑
AGNEWS	82.61 ± 0.65	0.74 ↓	82.54 ± 0.65	0.64 ↓
FMNIST	90.07 ± 0.07	-0.20 ↑	90.03 ± 0.07	-0.20 ↑
CIFAR	69.74 ± 0.62	0.54 ↓	69.74 ± 0.69	0.53 ↓

Table 4: The MTA(%) and F1(%) loss in quantization.

Evaluation Results

System-level Overhead

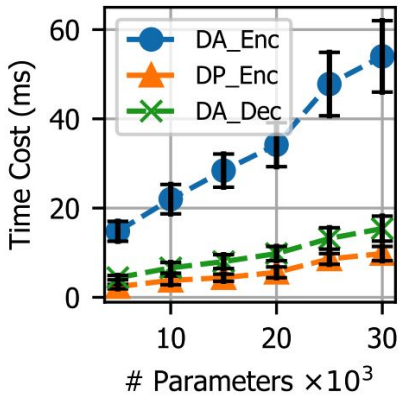


Figure 9: The homomorphic ENC/DEC times.

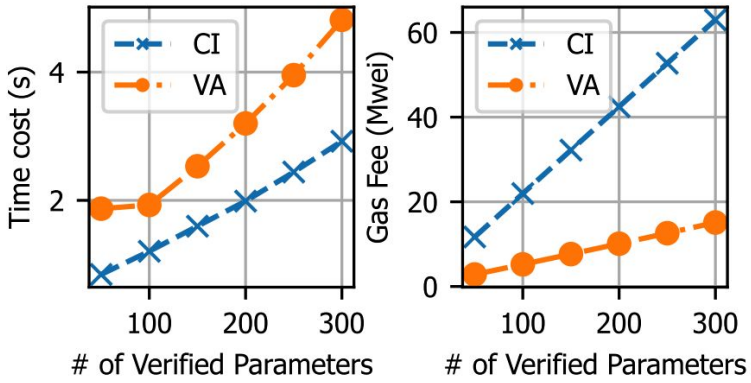


Figure 10: The gas cost and execution times for verifying different numbers of parameters.

Phrase	Prepare			Verify			Any
Function	NE	Deposit	CM	Prepare	CR(DA)	CR(DP)	RE
Gas (wei)	163076	46643	127731	222018	38036	42632	-
Time (ms)	92.0	73.6	80.6	83.4	64.2	70.6	74.8

* In this table, "NE" represents NewEpoch, "CM" represents CommitModel,"CR" represents Claim-Reward, and "RE" represents ReadEpoch.

Table 5: The gas costs and execution times of the functions in our trading smart contract.

Thank You

References

7. References

- martFL: Enabling Utility-Driven Data Marketplace with a Robust and Verifiable Federated Learning Architecture
- GitHub - InspiringGroup-Lab/martFL
- Federated Learning: Strategies for Improving Communication Efficiency
- Decentralized Federated Learning: A Survey and Perspective
- Deep Leakage from Gradients
- FLTrust: Byzantine-robust Federated Learning via Trust Bootstrapping - NDSS Symposium
- Machine Learning with Adversaries: Byzantine Tolerant Gradient Descent
- Homomorphic Encryption for Arithmetic of Approximate Numbers