BLOCKFLOW

BLOCKCHAIN + FEDERATED LEARNING

REFERENCE

- Mugunthan, Vaikkunth, Ravi Rahman, and Lalana Kagal.
  "BlockFLow: An Accountable and Privacy-Preserving Solution
  for Federated Learning."
  arXiv preprint arXiv:2007.03856 (2020).

ABSTRACT

ABSTRACT

- 연합 학습
  - Federated Learning (FL)
  - 데이터 공유 없이 머신 러닝을 진행하는 방법
- 악의적인 에이전트의 위험
  - 랜덤 데이터로 학습하거나
  - 결과 클래스를 반대로 학습하는 등의 악의적 행위
  - 통합 모델을 약화시킬 수 있음

ABSTRACT

- BlockFLow
  - 완전히 탈중앙화된
  - 프라이버시-보존이 가능한
  - 책임 의무가 있는 (accountable) FL
    - 책임감 있는
    - 책임
    - 책임-관련 등

ABSTRACT

- 최우선 목표
  - 에이전트들에게
  - 기여한 퀄리티에 따라
  - 보상을 주는 것
- 이더리움 스마트 컨트랙트를 통해
  - 좋은 행동에 대한 인센티브를 지급

ABSTRACT

- 기존 감사 및 책임-관련 FL 방법과는 달리
  - 제안하는 시스템은 중앙화된 테스트 데이터셋도 필요 없고
  - 에이전트 간 데이터 공유도 필요 없고
  - 하나 또는 그 이상의 신뢰할 수 있는 감사의 존재도 필요 없음
- 완전히 탈중앙화 되어있고
- 최대 50%의 악의적 담합에도 탄력성이 있음

ABSTRACT

- 실험 결과
  - 합리적인 블록체인 비용을 소모
  - 비정직한 에이전트에 대한 감사 점수(auditing score)가
  - 통계적으로 정직한 에이전트에 비해 낮게 나옴

INTRODUCTION

INTRODUCTION

- 연합 학습
  - 민감한 데이터의 공유 없이
  - 글로벌 모델의 발전을 가능하게 함

INTRODUCTION

- 각 에이전트가 각자의 데이터셋을 가지고 학습
  - 로컬 모델을 중앙화된 에이전트에게 공유
- 중앙화된 에이전트는 평균을 낸 모델을 계산하고 공유
- 이러한 프로세스가 수렴될 때까지 여럿 반복

INTRODUCTION

- 그러나 이런 naïve한 연합 학습 구현은
  - 여러 프라이버시 및 책임의 위협으로부터 취약함
- 공격자(들)은
  - 모델 인버전(inversion)
  - 멤버십 추론 공격
  - 등이 가능함

INTRODUCTION

- 차등 정보 보호
  - Differential Privacy (DP)
  - 모델 파라미터에 랜덤 노이즈를 더해 위와 같은 공격들을 막을 수 있음
- 그러나 DP는
  - 프로토콜을 따르지 않는 공격자들을 막을 수는 없음
  - 가령 출력 레이블을 뒤집어 학습하는 공격자
  - 공유 모델이 취약해지고, 그 이후 모든 라운드들의 정확도에 영향

INTRODUCTION

- 이러한 공격에 대응하기 위해서는
  - 각자 기여에 따라
  - 책임 의무를 부여해야 함
- 좋은 기여에 대해서는 보상

INTRODUCTION

- 탈중앙화된 연산 플랫폼인 블록체인을 책임을 위해 사용
- 블록체인은 다음 기술들의 결합
  - 불변의 원장
  - 화폐
  - (이더리움 블록체인의 경우) 튜링 완전 연산 환경

INTRODUCTION

- BlockFLow
  - 중앙화된 신뢰할 수 있는 에이전트는 필요 없음
  - 차등 정보 보호된 모델을 공유할 수 있게 함
  - 블록체인 스마트 컨트랙트를 에이전트의 기여를 평가하기 위해 활용
  - 좋은 행동에 대해서는 보상을 줄 수 있음

BLOCKCHAIN-BASED ACCOUNTABILITY

BLOCKCHAIN-BASED ACCOUNTABILITY

- 이미 블록체인 플랫폼이 FL에서 ‘책임’ 요소를 위해 사용되어 옴
- 1. 초기에 숨겨둔, 검증 가능한 테스트셋을 활용해 블록체인에서 모델을 직접 평가하는 경우
- 그러나 이러한 방법은
  - 블록체인 상에 데이터가 공개되어야만 함
  - 온체인 모델 평가에는 상당한 블록체인 연산 비용을 유발
- 따라서 작은 모델과 공개 데이터셋이 있는 경우에만 유효한 방법

BLOCKCHAIN-BASED ACCOUNTABILITY

- 2. 공개 온체인 평가 대신
  - 신뢰할 수 있는 수집자 및 평가자를 사용하는 방법이 있음
  - 모든 에이전트들이 중앙화된 에이전트(들)를 신뢰하는 것으로 동의
- 데이터가 공개적으로 공개되지 않으므로 보다 나은 프라이버시를 보장하지만
  - 약한 신뢰 모델은 활용도가 제한됨

BLOCKCHAIN-BASED ACCOUNTABILITY

- 3. 강인하고 신뢰가 필요 없는 그레디언트 기반의 검증 스킴을 제안
  - 오직 차등 정보 보호된 그레디언트 업데이트만이 공개적으로 블록체인에 공개
  - 평균 업데이트에 가까운 그레디언트 업데이트들만을 포함함
- 책임 요소가 포함되고 프라이버시 보존도 되지만,
  - 악의적 에이전트의 비율에 따라 에러가 함께 증가함
- 큰 모델에 대해 잘 확장되지도 않음
  - 그레디언트가 공개 블록체인에서 평균 계산되므로

BLOCKCHAIN-BASED ACCOUNTABILITY

- BlockFLow는 이러한 기존 방법들에 비해 상당한 강점을 보임
- 프라이버시 보존과 무신뢰성을 모두 확보
  - 비밀 테스트 데이터셋이 필요 없음
  - 신뢰할 수 있는 에이전트의 셋도 필요 없음
  - 공개 블록체인 상에 데이터 및 가중치의 공개가 필요 없음
- 악의적인 에이전트가 소수인 경우
  - 모든 악의적 에이전트들을 필터링할 수 있고, 공유 모델의 퀄리티를 보존할 수 있음

BLOCKFLOW

PROPOSED SYSTEM

- BlockFLow는 책임감 있는 연합 학습 시스템
  - 완전히 탈중앙화된
  - 프라이버시를 보존하는
- 각 에이전트 개인의 데이터셋을 보호하기 위해
  - 차등 정보 보호를 사용
- 이더리움 블록체인 스마트 컨트랙트를 사용
  - 책임감을 제공하기 위해

BLOCKFLOW CLIENT

- 각 에이전트는 BlockFLow 클라이언트 인스턴스를 구동해야 함

BLOCKFLOW CLIENT

- 모든 FL 라운드에
  - 각 BlockFLow 클라이언트는 로컬 모델을 학습함
  - 차등 정보 보호를 적용: 라플라시안(Laplacian) 노이즈를 모델에 더함
  - 모델을 다른 클라이언트들과 공유
  - 다른 클라이언트의 모델을 평가
  - 평가 점수를 BlockFLow 스마트 컨트랙트에 등록
  - 전체 점수와 평균 클라이언트 모델을 얻음

BLOCKFLOW CLIENT

- 클라이언트는 이더리움 클라이언트를 통해 스마트 컨트랙트와 소통
- 데이터를 이더리움 블록체인에 저장하려면 비용이 높기 때문에
  - BlockFLow는 IPFS(InterPlanetary File System)를 사용해
  - 각 라운드에 등장하는 차등 정보 보호된 모델을 저장

SMART CONTRACT

ETHEREUM SMART CONTRACT

- 스마트 컨트랙트를 초기화하고 배포하기 위해
  - 연합 학습 라운드
  - 참여할 이더리움 어카운트의 수
  - 환급금 지불을 통제하기 위한 보증금(bond) 파라미터
  - 각 스테이지의 지속시간을 제어하는 타이밍(timing) 파라미터
  - 를 명시해야 함

ETHEREUM SMART CONTRACT

- 각 클라이언트들은 참여를 위해
  - 재화인 이더(ETH)를 스마트 컨트랙트에
  - 보증금 식으로 걸어야 함
- 이 보증금은
  - 클라이언트들이 프로토콜을 따르게 보장하고
  - 기여에 따른 보상이나 패널티를 받도록 하는데 쓰임

ETHEREUM SMART CONTRACT

- 각 FL 라운드는 다섯 스테이지로 구성됨
  - Train
  - Retrieve
  - Evaluation commit
  - Evaluation reveal
  - Compute score

ETHEREUM SMART CONTRACT

- 스마트 컨트랙트가 각 스테이지의 데드라인을 엄격하게 강제함
  - 데드라인을 놓친 클라이언트는 스마트 컨트랙트로부터 제외당하며
  - 보증금을 잃어버리게 됨

ETHEREUM SMART CONTRACT

- Train 데드라인 전까지
  - 클라이언트들은 훈련한 모델에 대한 IPFS 주소를
  - 스마트 컨트랙트에 등록해야 함
- Retrieve 데드라인 전까지
  - 각 클라이언트는 스마트 컨트랙트에게
  - 성공적으로 검증된 모델을 가진 클라이언트 집합을 고지

ETHEREUM SMART CONTRACT

- 이후 스마트 컨트랙트는
  - 어느 클라이언트(들)가
  - 다른 클라이언트들의 모델 대다수를 회수했는지
  - 그리고 다른 클라이언트들이 이 클라이언트의 모델을 회수했는지 계산
- 테스트: 이 계산으로부터 기준을 만족시키지 못하면
  - 해당 클라이언트(들)를 제거

ETHEREUM SMART CONTRACT

- 클라이언트들은 이 테스트를 통과한 에이전트들을 평가해야 함
- Evaluation submit 데드라인 전까지
  - 모든 클라이언트들은
  - 평가 점수를 무작위 값(salts)와 함께 암호화해서 스마트 컨트랙트에 등록해야 함
- Evaluation reveal 데드라인 전까지
  - 각자의 무작위 값과 복호화된 평가 점수를 스마트 컨트랙트에 공개해야 함
  - 스마트 컨트랙트는 무작위 값과 점수를 검증함

ETHEREUM SMART CONTRACT

- Compute score 스테이지에서
  - 스마트 컨트랙트는 기여도 점수화를 수행
  - 각 클라이언트에 대해 전체 스코어를 결정
- 스마트 컨트랙트는
  - 이 점수를 이용해 비율에 비례하는 보증금 환급을 수행
- 클라이언트들은
  - 이 점수를 이용해 각 모델이 평균에서 얼마나 가중될 것일지를 결정

ETHEREUM SMART CONTRACT

- 이 과정이 모든 FL 라운드에 대하여 수행됨
  - 마지막 라운드가 끝나고 나면
  - 남은 보증금은 클라이언트에게 환급됨

CONTRIBUTION SCORING

CONTRIBUTION SCORING PROCEDURE

- BlockFLow는 모든 클라이언트가 다른 모든 클라이언트의 모델을 평가하도록 함
  - 클라이언트는 자신의 전체 데이터셋을 이용해 평가를 수행
- 모델의 집합 W, 평가 함수 $eval$에 대해

$$(\forall w, w' \in \{W \times W\}, w \neq w') \quad \texttt{eval}(w) > \texttt{eval}(w') \iff \text{model } w \text{ is better than model } w'$$
$$(\forall w \in W) \quad \texttt{eval}(w) \in [0.0; 1.0]$$

  - 가령, 분류 문제에 대해서는 정확도가 이 역할을 수행해줄 것

CONTRIBUTION SCORING PROCEDURE

- 클라이언트의 전체 점수는
  - a) 그들 모델에 대해 보고된 점수의 중앙값
  - b) 보고된 점수와 중앙값 사이의 최대 차이의 역수
  - 중 더 작은 값으로 설정됨

CONTRIBUTION SCORING PROCEDURE

- 클라이언트의 전체 점수는
  - a) 그들 모델에 대해 보고된 점수의 중앙값
  - b) 보고된 점수와 중앙값 사이의 최대 차이의 역수
  - 중 더 작은 값으로 설정됨
- 낮은 퀄리티의 모델을 등록할 경우 a) 값을 낮게 받을 것

CONTRIBUTION SCORING PROCEDURE

- 클라이언트의 전체 점수는
  - a) 그들 모델에 대해 보고된 점수의 중앙값
  - b) 보고된 점수와 중앙값 사이의 최대 차이의 역수
  - 중 더 작은 값으로 설정됨
- 평가 퀄리티 점수
  - 부정확한 평가를 등록할 경우 중앙값과 현저히 큰 차이를 보일 것
  - 자신의 점수에 부정적 영향

CONTRIBUTION SCORING

- *eval* 부분을 제외하고
  - 나머지 부분들은 스마트 컨트랙트에서 구동됨

Algorithm 1: Contribution Scoring Procedure. This procedure, except for the `eval` method, is implemented in the blockchain smart contract.

for all client pairs $\{a, k\} \in \{N \times N\}$ **do**
  $s_{a,k} \leftarrow \texttt{eval}_a(k)$ {client a performs off-chain evaluation of client k's model using a's own dataset and reports the score here. See section 3.3 for `eval` requirements.}
end for
for all clients $k \in N$ **do**
  $m_k \leftarrow \text{MED}\{s_{a,k} : \forall a \in N\}$ {Compute the median model scores}
end for
for all clients $k \in N$ **do**
  $m'_k \leftarrow \frac{m_k}{\max\{m'_k : \forall k' \in N\}}$ {Scale the median model scores to ensure a maximum of 1.0}
end for
for all client pairs $\{a, k\} \in \{N \times N\}$ **do**
  $t_{a,k} \leftarrow |s_{a,k} - m_k|$ {Compute the evaluation quality scores}
  $t'_{a,k} \leftarrow \max(0, \frac{0.5 - t_{a,k}}{0.5 + t_{a,k}})$ {Transform the evaluation quality scores}
end for
for all clients $a \in N$ **do**
  $d_a \leftarrow \min\{t'_{a,k} : \forall k \in N\}$ {Compute the least accurate evaluation each client performed}
end for
for all clients $a \in N$ **do**
  $d'_a \leftarrow \frac{d_a}{\max\{d_{a'} : \forall a' \in N\}}$ {Scale the least accurate evaluation scores}
end for
for all clients $k \in N$ **do**
  $p_k \leftarrow \min(m'_k, d'_k)$ {Compute the overall scores}
end for
return p

CONTRIBUTION SCORING

- k 클라이언트의 모델을
- a가 자신의 데이터셋으로
- 평가, 보고

Algorithm 1: Contribution Scoring Procedure. This procedure, except for the `eval` method, is implemented in the blockchain smart contract.

for all client pairs $\{a, k\} \in \{N \times N\}$ **do**
  $s_{a,k} \leftarrow \texttt{eval}_a(k)$ {client a performs off-chain evaluation of client k's model using a's own dataset and reports the score here. See section 3.3 for `eval` requirements.}
end for
for all clients $k \in N$ **do**
  $m_k \leftarrow \text{MED}\{s_{a,k} : \forall a \in N\}$ {Compute the median model scores}
end for
for all clients $k \in N$ **do**
  $m'_k \leftarrow \frac{m_k}{\max\{m'_k : \forall k' \in N\}}$ {Scale the median model scores to ensure a maximum of 1.0}
end for
for all client pairs $\{a, k\} \in \{N \times N\}$ **do**
  $t_{a,k} \leftarrow |s_{a,k} - m_k|$ {Compute the evaluation quality scores}
  $t'_{a,k} \leftarrow \max(0, \frac{0.5 - t_{a,k}}{0.5 + t_{a,k}})$ {Transform the evaluation quality scores}
end for
for all clients $a \in N$ **do**
  $d_a \leftarrow \min\{t'_{a,k} : \forall k \in N\}$ {Compute the least accurate evaluation each client performed}
end for
for all clients $a \in N$ **do**
  $d'_a \leftarrow \frac{d_a}{\max\{d_{a'} : \forall a' \in N\}}$ {Scale the least accurate evaluation scores}
end for
for all clients $k \in N$ **do**
  $p_k \leftarrow \min(m'_k, d'_k)$ {Compute the overall scores}
end for
return p

CONTRIBUTION SCORING

- 클라이언트 k에 대한
- 중앙값 점수 계산

Algorithm 1: Contribution Scoring Procedure. This procedure, except for the `eval` method, is implemented in the blockchain smart contract.

for all client pairs $\{a, k\} \in \{N \times N\}$ **do**
  $s_{a,k} \leftarrow \texttt{eval}_a(k)$ {client a performs off-chain evaluation of client k's model using a's own dataset and reports the score here. See section 3.3 for `eval` requirements.}
end for
for all clients $k \in N$ **do**
  $m_k \leftarrow \text{MED}\{s_{a,k} : \forall a \in N\}$ {Compute the median model scores}
end for
for all clients $k \in N$ **do**
  $m'_k \leftarrow \frac{m_k}{\max\{m'_k : \forall k' \in N\}}$ {Scale the median model scores to ensure a maximum of 1.0}
end for
for all client pairs $\{a, k\} \in \{N \times N\}$ **do**
  $t_{a,k} \leftarrow |s_{a,k} - m_k|$ {Compute the evaluation quality scores}
  $t'_{a,k} \leftarrow \max(0, \frac{0.5 - t_{a,k}}{0.5 + t_{a,k}})$ {Transform the evaluation quality scores}
end for
for all clients $a \in N$ **do**
  $d_a \leftarrow \min\{t'_{a,k} : \forall k \in N\}$ {Compute the least accurate evaluation each client performed}
end for
for all clients $a \in N$ **do**
  $d'_a \leftarrow \frac{d_a}{\max\{d_{a'} : \forall a' \in N\}}$ {Scale the least accurate evaluation scores}
end for
for all clients $k \in N$ **do**
  $p_k \leftarrow \min(m'_k, d'_k)$ {Compute the overall scores}
end for
return p

CONTRIBUTION SCORING

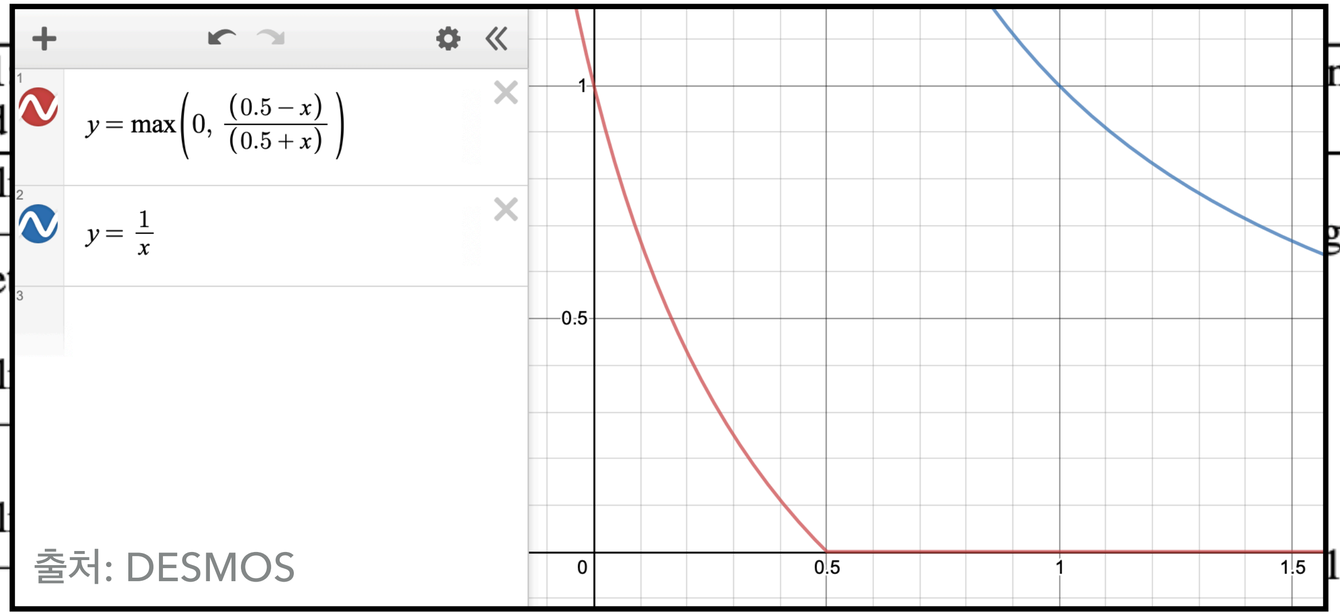
- 중앙값 점수 중 최대가
- 1.00이 되도록 스케일링

Algorithm 1: Contribution Scoring Procedure. This procedure, except for the `eval` method, is implemented in the blockchain smart contract.

for all client pairs $\{a, k\} \in \{N \times N\}$ **do**
  $s_{a,k} \leftarrow \texttt{eval}_a(k)$ {client a performs off-chain evaluation of client k's model using a's own dataset and reports the score here. See section 3.3 for `eval` requirements.}
end for
for all clients $k \in N$ **do**
  $m_k \leftarrow \text{MED}\{s_{a,k} : \forall a \in N\}$ {Compute the median model scores}
end for
for all clients $k \in N$ **do**
  $m'_k \leftarrow \frac{m_k}{\max\{m'_k : \forall k' \in N\}}$ {Scale the median model scores to ensure a maximum of 1.0}
end for
for all client pairs $\{a, k\} \in \{N \times N\}$ **do**
  $t_{a,k} \leftarrow |s_{a,k} - m_k|$ {Compute the evaluation quality scores}
  $t'_{a,k} \leftarrow \max(0, \frac{0.5 - t_{a,k}}{0.5 + t_{a,k}})$ {Transform the evaluation quality scores}
end for
for all clients $a \in N$ **do**
  $d_a \leftarrow \min\{t'_{a,k} : \forall k \in N\}$ {Compute the least accurate evaluation each client performed}
end for
for all clients $a \in N$ **do**
  $d'_a \leftarrow \frac{d_a}{\max\{d_{a'} : \forall a' \in N\}}$ {Scale the least accurate evaluation scores}
end for
for all clients $k \in N$ **do**
  $p_k \leftarrow \min(m'_k, d'_k)$ {Compute the overall scores}
end for
return p

CONTRIBUTION SCORING

- 평가 퀄리티 점수 계산
- 계산 후 역수를 취함
  - 0~1 범위의
  - 역수 계산을 위해 함수 $max(0, \frac{0.5 - x}{0.5 + x})$ 사용
  - 차이가 0.5보다 떨어진 경우 0으로 만듦



Algorithm 1 ... method, is implemented

for all cl
  $s_{a,k} \leftarrow$ ... g a's own
  datase
end for
for all cl
  $m_k \leftarrow$ ...
end for
for all cl
  $m'_k \leftarrow$ ... 1.0}
end for

for all client pairs $\{a, k\} \in \{N \times N\}$ **do**
  $t_{a,k} \leftarrow |s_{a,k} - m_k|$ {Compute the evaluation quality scores}
  $t'_{a,k} \leftarrow \max(0, \frac{0.5 - t_{a,k}}{0.5 + t_{a,k}})$ {Transform the evaluation quality scores}
end for
for all clients $a \in N$ **do**
  $d_a \leftarrow \min\{t'_{a,k} : \forall k \in N\}$ {Compute the least accurate evaluation each client performed}
end for
for all clients $a \in N$ **do**
  $d'_a \leftarrow \frac{d_a}{\max\{d_{a'} : \forall a' \in N\}}$ {Scale the least accurate evaluation scores}
end for
for all clients $k \in N$ **do**
  $p_k \leftarrow \min(m'_k, d'_k)$ {Compute the overall scores}
end for
return p

CONTRIBUTION SCORING

- 역수 중 최소를 계산
  - 최대 차이의 역수 ==
  - 차이의 역수 중 최소

Algorithm 1: Contribution Scoring Procedure. This procedure, except for the `eval` method, is implemented in the blockchain smart contract.

for all client pairs $\{a, k\} \in \{N \times N\}$ **do**
  $s_{a,k} \leftarrow \texttt{eval}_a(k)$ {client a performs off-chain evaluation of client k's model using a's own dataset and reports the score here. See section 3.3 for `eval` requirements.}
end for
for all clients $k \in N$ **do**
  $m_k \leftarrow \text{MED}\{s_{a,k} : \forall a \in N\}$ {Compute the median model scores}
end for
for all clients $k \in N$ **do**
  $m'_k \leftarrow \frac{m_k}{\max\{m'_k : \forall k' \in N\}}$ {Scale the median model scores to ensure a maximum of 1.0}
end for
for all client pairs $\{a, k\} \in \{N \times N\}$ **do**
  $t_{a,k} \leftarrow |s_{a,k} - m_k|$ {Compute the evaluation quality scores}
  $t'_{a,k} \leftarrow \max(0, \frac{0.5 - t_{a,k}}{0.5 + t_{a,k}})$ {Transform the evaluation quality scores}
end for
for all clients $a \in N$ **do**
  $d_a \leftarrow \min\{t'_{a,k} : \forall k \in N\}$ {Compute the least accurate evaluation each client performed}
end for
for all clients $a \in N$ **do**
  $d'_a \leftarrow \frac{d_a}{\max\{d_{a'} : \forall a' \in N\}}$ {Scale the least accurate evaluation scores}
end for
for all clients $k \in N$ **do**
  $p_k \leftarrow \min(m'_k, d'_k)$ {Compute the overall scores}
end for
return p

CONTRIBUTION SCORING

- '최소 역수' 중 최대가
- 1.00이 되도록 스케일링

Algorithm 1: Contribution Scoring Procedure. This procedure, except for the `eval` method, is implemented in the blockchain smart contract.

for all client pairs $\{a, k\} \in \{N \times N\}$ **do**
  $s_{a,k} \leftarrow \texttt{eval}_a(k)$ {client a performs off-chain evaluation of client k's model using a's own dataset and reports the score here. See section 3.3 for `eval` requirements.}
end for
for all clients $k \in N$ **do**
  $m_k \leftarrow \text{MED}\{s_{a,k} : \forall a \in N\}$ {Compute the median model scores}
end for
for all clients $k \in N$ **do**
  $m'_k \leftarrow \frac{m_k}{\max\{m'_k : \forall k' \in N\}}$ {Scale the median model scores to ensure a maximum of 1.0}
end for
for all client pairs $\{a, k\} \in \{N \times N\}$ **do**
  $t_{a,k} \leftarrow |s_{a,k} - m_k|$ {Compute the evaluation quality scores}
  $t'_{a,k} \leftarrow \max(0, \frac{0.5 - t_{a,k}}{0.5 + t_{a,k}})$ {Transform the evaluation quality scores}
end for
for all clients $a \in N$ **do**
  $d_a \leftarrow \min\{t'_{a,k} : \forall k \in N\}$ {Compute the least accurate evaluation each client performed}
end for
for all clients $a \in N$ **do**
  $d'_a \leftarrow \frac{d_a}{\max\{d_{a'} : \forall a' \in N\}}$ {Scale the least accurate evaluation scores}
end for
for all clients $k \in N$ **do**
  $p_k \leftarrow \min(m'_k, d'_k)$ {Compute the overall scores}
end for
return p

CONTRIBUTION SCORING

- 클라이언트 k의 최종 점수
  - a)와 b) 중 최소값

Algorithm 1: Contribution Scoring Procedure. This procedure, except for the `eval` method, is implemented in the blockchain smart contract.

for all client pairs $\{a, k\} \in \{N \times N\}$ **do**
  $s_{a,k} \leftarrow \texttt{eval}_a(k)$ {client a performs off-chain evaluation of client k's model using a's own dataset and reports the score here. See section 3.3 for `eval` requirements.}
end for
for all clients $k \in N$ **do**
  $m_k \leftarrow \text{MED}\{s_{a,k} : \forall a \in N\}$ {Compute the median model scores}
end for
for all clients $k \in N$ **do**
  $m'_k \leftarrow \frac{m_k}{\max\{m'_k : \forall k' \in N\}}$ {Scale the median model scores to ensure a maximum of 1.0}
end for
for all client pairs $\{a, k\} \in \{N \times N\}$ **do**
  $t_{a,k} \leftarrow |s_{a,k} - m_k|$ {Compute the evaluation quality scores}
  $t'_{a,k} \leftarrow \max(0, \frac{0.5 - t_{a,k}}{0.5 + t_{a,k}})$ {Transform the evaluation quality scores}
end for
for all clients $a \in N$ **do**
  $d_a \leftarrow \min\{t'_{a,k} : \forall k \in N\}$ {Compute the least accurate evaluation each client performed}
end for
for all clients $a \in N$ **do**
  $d'_a \leftarrow \frac{d_a}{\max\{d_{a'} : \forall a' \in N\}}$ {Scale the least accurate evaluation scores}
end for
for all clients $k \in N$ **do**
  $p_k \leftarrow \min(m'_k, d'_k)$ {Compute the overall scores}
end for
return p

THREAT ANALYSIS

THREAT ANALYSIS

- BlockFLow는 50% 악의적 위협 모델까지 저항성을 가짐
  - N 명의 에이전트가 있으면
  - BlockFLow는 $M \in [0, \frac{N}{2})$ 까지 무결성을 유지할 수 있음
- 모든 종류의 알려진 공격에 대해 조사하고
  - 각 공격에 대해 저항성을 가지는지 살펴볼 것

PRIVACY THREAT MODEL

- 차등 정보 보호의 정의에 따라
  - 매우 높은 확률로
  - $N-1$의 담합한 에이전트들은
  - 나머지 에이전트의 어떠한 정보도 알 수 없음
- $N \geq 3$ 이면 $N-1 > \frac{N}{2}$ 임

ETHEREUM BLOCKCHAIN THREAT MODEL

- 공개키/비밀키 암호학과 PoW 합의 알고리즘
  - 이더리움 블록체인의 보안
- 전체 이더리움 50% 이상의 연산력을 차지하지 않는 한
  - 공격할 명시적 방법은 없음
  - 이러한 공격은 시간 당 $150,000에 가까운 비용을 요구
  - 아직 이더리움 메인넷에서 성공한 바 없음
- 따라서 BlockFLow의 실질적 위협으로 고려하지 않음

ETHEREUM BLOCKCHAIN THREAT MODEL

- 이더리움 블록체인은 공개되어 있고, 익명이므로
  - 클라이언트들이 이론적으로 수 차례 등록될 수 있음
  - 이에 투표에 있어 불균형적을 만들 수 있음
- BlockFLow는 사전 입증된 이더리움 계좌만 참여 가능하게 하여 방어
  - 각 에이전트가 단 하나의 계좌만을 가지는 것으로 보장

IPFS AND DATA SHARING THREAT MODEL

- IPFS는 불변하고
  - 에이전트는 스마트 컨트랙트에 모델의 해시 값을 등록한 후에는
  - 변경할 수 없음
- BlockFLow 프로토콜은 각 에이전트가
  - N/2 모델 이상을 로드(load)하도록 요구하며
  - N/2 이상의 에이전트들은 어느 한 모델에 대해 동일한 것으로 보고해야 함
  - BlockFLow 위협 모델에서는 N/2 이상의 정직한 에이전트가 있으므로 충분

CONTRIBUTION SCORING PROCEDURE THREAT MODEL

- 기여 점수 산정에는 더 많은 공격이 가능함
- 1. 악의적 모델
  - 가중치가 믿을 수 있는 데이터셋을 반영하지 않는 경우
  - 모델이 랜덤하게 생성되거나
  - 출력이 반전될 경우

CONTRIBUTION SCORING PROCEDURE THREAT MODEL

- BlockFLow 기여 점수 산정 방법은 악의적 모델을 등록할 경우 패널티를 줌
  - 모든 에이전트가 다른 에이전트의 모델을 평가함
  - 낮은 점수는 낮은 암호화폐 보상으로 이어짐
  - 각 라운드의 끝에 글로벌 모델이 계산될 때
    - 점수에 기반해 가중치가 적용됨

CONTRIBUTION SCORING PROCEDURE THREAT MODEL

- 담합한 에이전트들이 훈련 과정 중에 더 나은 모델을 제출하는 경우
  - 예를 들어, 에이전트들은 비밀리에 raw 데이터나 모델을 공유할 수 있음
  - M < N/2 명의 담합
  - 모든 M 에이전트들은 최고의 모델을 등록할 것
- BlockFLow는 강력한 모델을 기여한 에이전트에게 보상을 주는데,
  - 여러 에이전트들이 같은 모델을 등록하는 것 역시 허용함
  - 많은 에이전트들이 강력한 데이터셋을 가지는 것과 마찬가지니, 공격으로 간주하지 않음

CONTRIBUTION SCORING PROCEDURE THREAT MODEL

- 평가 중에 공격이 있을 수 있음
  - 암호화 및 commit-then-reveal 프로토콜을 통해
  - BlockFLow 스마트 컨트랙트는 다른 사람의 점수를
  - 담합이 없을 때 복제해서 활용하지 못하도록 함
- 또한 모델의 중앙값에서 0.5 이상 떨어진 값들을 0으로 만들기 때문에
  - 정직하게 평가하도록 만듦

CONTRIBUTION SCORING PROCEDURE THREAT MODEL

- 담합이 있다면, 정직하지 못한 에이전트는 동일한 실제같은 수치를 등록할 것
  - 심지어 조작된 수치일 수 있음
- 악의적 에이전트가
  - 자신들의 모델에는 1.0을
  - 남들(정직한 에이전트들)의 모델에는 0.0을 준다고 생각할 수 있음

CONTRIBUTION SCORING PROCEDURE THREAT MODEL

- 그러나 이들은 절반 보다 적으므로
  - 또한 중앙값 모델 점수가 전체 스코어를 결정하는데 사용되므로
  - 중앙값 점수가 정직한 에이전트로부터 보고된 최소와 최대 점수 사이에 위치할 것이 보장
- 정직한 에이전트의 점수는 공정한 평가에서부터 기인하므로
  - 담합하는 에이전트들이 평가 점수에 영향을 끼치는 것이 불가능

CONTRIBUTION SCORING PROCEDURE THREAT MODEL

- 또한 점수의 위조는 이를 시도한 사람에게만 불이익을 줌
  - 기여 점수는 중앙값에서 가장 멀리 떨어진 수치를 가지고 최종 점수를 제한함
- 구체적으로
  - 중앙값에서 0.5 이상 떨어진 평가는 0점의 점수를 받게 됨
  - 암호화폐 풀의 지분이 없게 됨
- 위조 행위가 최적 행동이 될 수 없음

EVALUATION

EVALUATION

- Adult Census Income (Adult)
- Third International Knowledge Discovery and Data Mining Tools Competition (KDD)
- 데이터셋 사용

EVALUATION

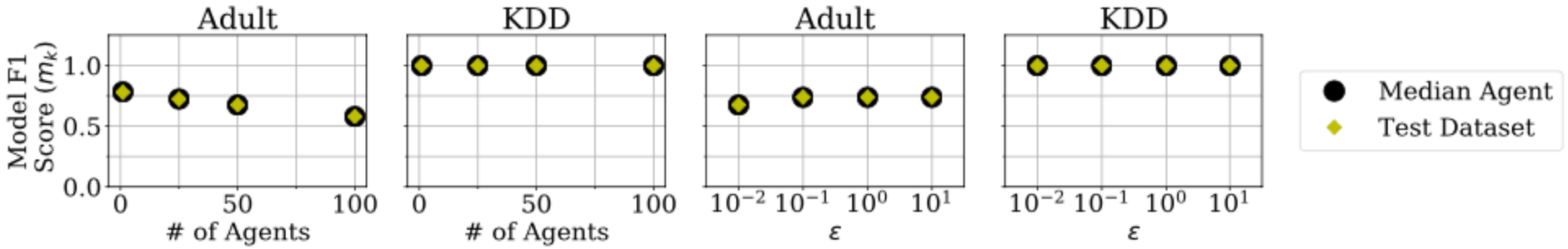
- 2/3의 데이터는 N 클라이언트들에게 분배
  - 중복 없음
  - 그 중 20%의 데이터는 훈련 검증을 위해 사용
- 클라이언트들은 전체 데이터셋을 다른 클라이언트들의 모델을 평가할 때 사용
- 남은 1/3은 테스트를 위해 남김

EVALUATION

- 전체 실험에서 로지스틱 회귀를 사용
  - L2 정규화 계수 $\alpha = 1.0$
- Eval 함수에는 F1 점수를 사용
- 실험 상세 스펙은 원 논문을 참고

EXPERIMENT 1: MEASURING MODEL QUALITY

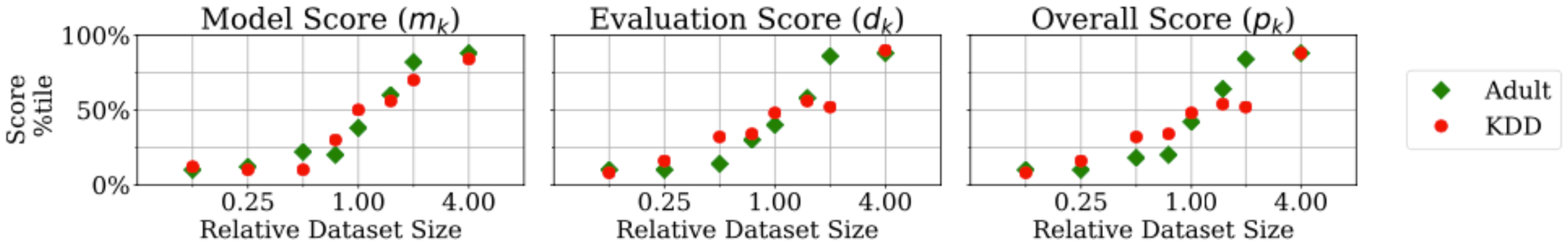
- 에이전트들의 F1 점수의 중앙값이
  - 1/3 테스트 데이터셋 F1 점수와 나란히 위치함
- 에이전트들은 이 데이터셋으로 학습하거나 평가받은 바 없음



- 즉, F1 점수의 중앙값은 모델 퀄리티를 잘 대변함

EXPERIMENT 2: REWARDING LARGER DATASETS

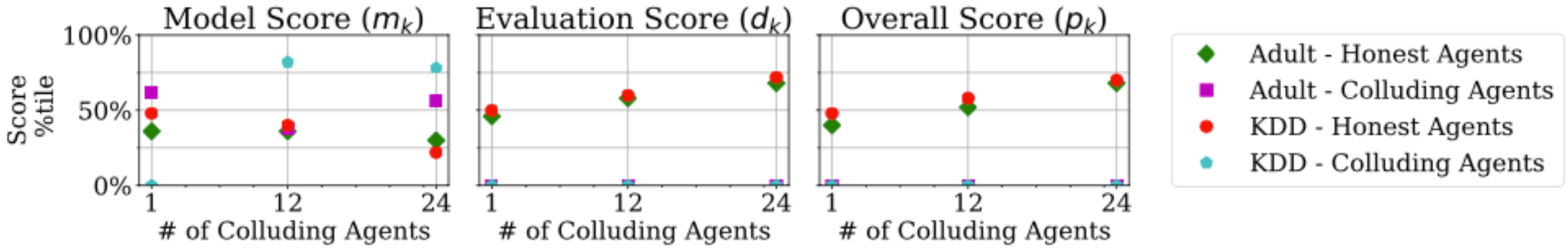
- 기여 점수 절차가 높은 퀄리티의 데이터셋에 대해 잘 보상하는지 실험
  - 높은 퀄리티의 데이터셋은 곧 강인한 모델과 정확도로 이어짐



- 에이전트의 모델, 평가, 전체 점수는 데이터셋 사이즈와 매우 높은 상관관계를 가짐
  - 다른 어떠한 정보 공유 없이도
  - 많은 데이터셋을 가진 에이전트에게 더 보상을 줄 수 있음

EXPERIMENT 3: DISCOURAGING COLLUSION

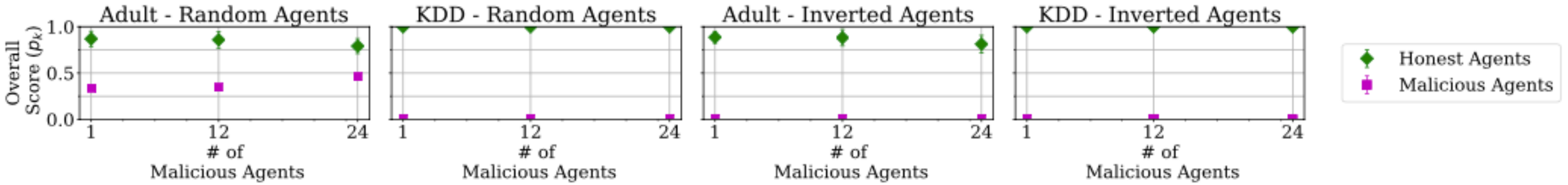
- 정직한 에이전트들 vs. 담합하는 에이전트들의
  - 모델 점수, 평가 점수, 전체 점수의 비교



- 전체 N=50, 담합한 클라이언트끼리는 서로 1.0점을 줌
- 담합의 경우, 모델 점수는 높게 나오더라도 전체 점수가 낮게 나옴

EXPERIMENT 4: PUNISHING MALICIOUS AGENTS

- 악의적인 에이전트들은 랜덤한 데이터 또는 반전된 데이터로 학습
  - 마찬가지로 80:20으로 훈련과 검증 데이터셋을 나눠서 진행



- 전체 N=50
- 악의적인 에이전트들의 점수가 통계적으로 정직한 에이전트들 대비 낮게 나옴

BLOCKCHAIN COSTS OF SYSTEM

- 이더리움 연산 비용은 가스(gas)로 측정됨
  - 스토리지 사용량과 CPU 연산과 연관
  - 가스 사용량은 마켓 컨디션과는 상관 없음
    - 가스 가격 또는 이더리움 가격과 상관 없다는 뜻
- N명의 에이전트와 R번의 연합 학습 라운드에 대해
  - $\text{gas}(N, R) = 31913N + 542045R + 477050NR$
  - 이는 오프-체인(off-chain) 연산을 활용하면 경감할 수 있음

CONCLUSION

CONCLUSION AND FUTURE WORK

- BlockFLow
  - 완전한 탈중앙화와 프라이버시 보존을 갖춘
  - 책임감 있는 연합 학습 시스템
- 연합 학습에서 신뢰와 책임감 사이의 격차를 해소
- 기존의 어떠한 시스템도
  - 책임 보장, 프라이버시 보존, 완전한 무신뢰, 악의적 에이전트 위협 모델을
  - 동시에 제공하지 않음

CONCLUSION AND FUTURE WORK

- 이러한 특징들의 조합은 연합 학습 연구의 범위를 확장시킴
  - 가령: 에이전트들이 서로 알 필요도, 신뢰할 필요도 없고
  - 퀄리티 높은 모델을 제공한 것에 대해 보상을 받는 국제 협력

CONCLUSION AND FUTURE WORK

- BlockFLow는 성능 향상을 위해 확장될 수 있음
  - N 에이전트와 R 연합 학습 라운드에 대해
  - 점근적 비용은 $O(N^2R)$로 표현됨
- 각 라운드마다, 각 에이전트들은 다른 에이전트들의 모델을 평가해야 함

CONCLUSION AND FUTURE WORK

- 대신에, N 명의 에이전트가 아닌 Q<<N 명의 에이전트를
  - 스마트 컨트랙트에서 무작위로 선정해
  - 서로의 작업을 평가하도록 할 수 있음
- 충분히 큰 Q에 대해 높은 확률로 정확한 결과를 끌어낼 수 있고
  - $M < \frac{N}{2}$ 악의적 에이전트에 대해서도 내성이 있을 것
- 이러한 절충으로부터 가스 소모량을 크게 줄일 수 있고 확장성을 키울 수 있을 것

BLOCKFLOW

BLOCKCHAIN + FEDERATED LEARNING