Greedeeptector: Greedy Contract Detection using Lightweight Deep Learning implemented through eXplainable AI

https://youtu.be/5sphHV1-m40

Introduction

Related Works

Model implementation

Instruction Analysis

Introduction

- **스마트 컨트랙트** : 블록체인 기반의 디지털 계약으로 제 3자의 개입 없이 계약 가능
- **탐욕 컨트랙트** : 이더를 인출할 수 없도록 이더를 **무기한 잠금**을 거는 컨트랙트
  → 탐욕 컨트랙트를 실행시킬 경우 막대한 피해 발생

가상화폐 지갑 '패리티', 이더리움 2억8000만달러 묶여

발행일 : 2017-11-09 07:00

패리티 테크놀로지스(Parity Technologies) 의 가상화폐 지갑에서 2억8000만 달러(약 3100억원) 규모 이더리움(Ethereum) 거래가 동결됐다.

Introduction

- **기존 해결방안**
  - 악성 노드 탐지
  - 악성 스마트 컨트랙트 탐지

    → 파이썬 도구를 통한 탐지 (MAIAN tool)

    → **딥러닝을 통한 탐지** (스마트 컨트랙트의 특징을 학습)

- **한계점**
  - 블록체인은 확장성이 매우 중요함

    → IoT 블록체인에서 기존 딥러닝 탐지 모델은 **연산 및 메모리 오버헤드**를 유발할 수 있음

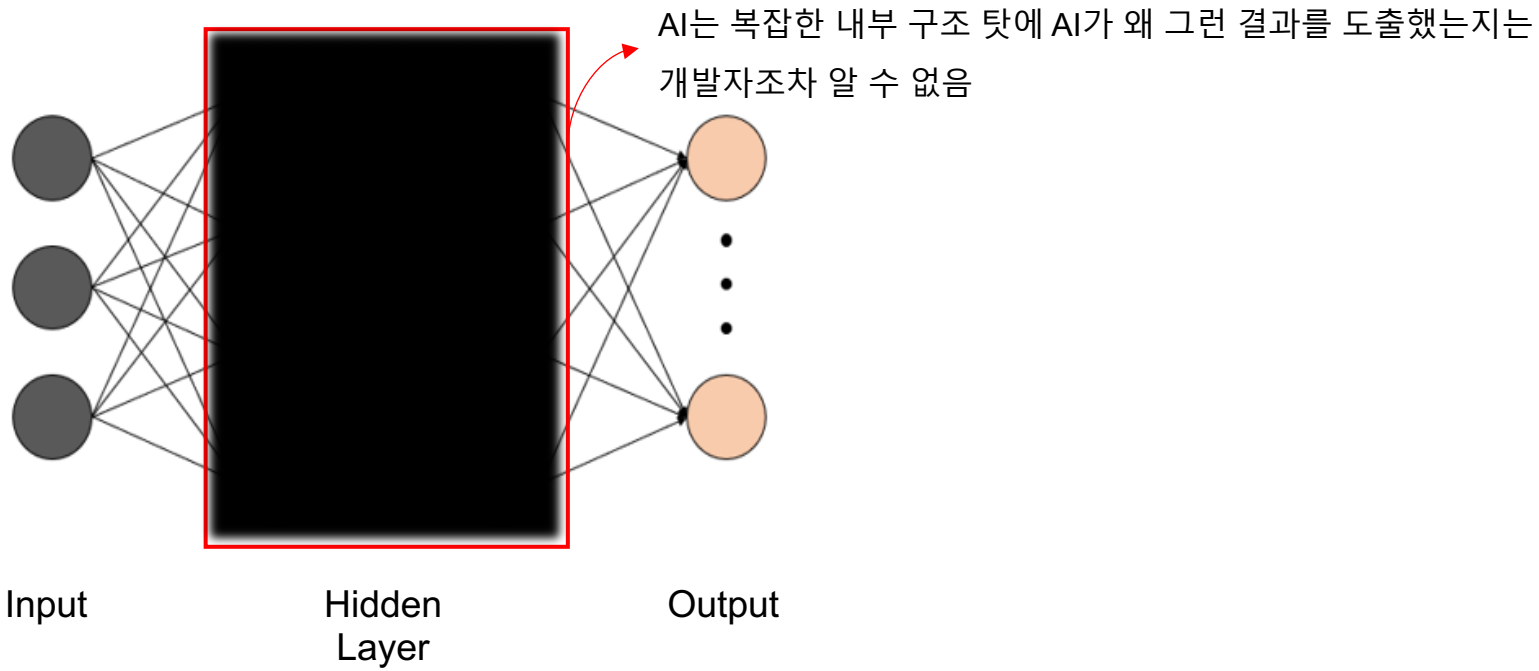
Contribution

1. **XAI를 통한 탐욕 컨트랙트 명령어 심층 분석**
   - Integrated Gradient와 Gradient SHAP을 통해 모델의 예측에 영향을 끼치는 명령어들을 분석
   - 어떤 명령어가 탐욕 컨트랙트에서 큰 특징을 가지며 자주 사용되는지 분석
2. **중요한 명령어를 기반으로 경량화된 딥러닝 모델 구현**
   - XAI를 통해 추출된 중요한 명령어들은 더 적은 데이터 차원
   - 이 명령어들을 통해 모델을 학습시킴으로써 경량화된 모델을 구현
     → 경량화된 모델은 Base 모델 대비 **약 50% 경량화**
3. **컨트랙트 실행 시 탐지를 통한 블록체인 네트워크의 안정성 향상**
   - 이전 연구와 다르게, 스마트 컨트랙트 실행 시 탐지를 수행
   - 새로 배포되는 스마트 컨트랙트 뿐만 아니라 이미 배포된 스마트 컨트랙트에 대해서도 탐지를 가능
     → 블록체인 네트워크의 **안정성을 향상**

Related Works

- **Explainable Artificial Intelligence (XAI)**
  - 기존 인공지능은 블랙박스 모델
  - 인공지능은 복잡한 관계와 특징을 학습

    → 예측에 대한 이유와 근거가 불명확
  - Local : 모델이 특정한 입력 데이터를 받았을 때, 각 변수가 예측 결과에 미치는 영향력을 계산하는 방법
  - Global : 모델 전체에 대해서 인사이트를 제공하며, 입력 전체의 배열에 대해 설명을 제공하는 방법



Related Works

- Gradient SHAP
  - 각 특성이 예측에 얼마나 기여했는지를 나타낸 값
  - local 및 global한 해석 O

    → feature 분석에 용이
  - feature의 개수가 늘어남에 따라 계산량이 기하급수적으로 증가

    → 기울기를 통해 근사값을 계산

- Integrated Gradients (IG)
  - 입력과 베이스라인 간의 차와 기울기 정보를 누적한 것의 곱
  - IG는 각 feature의 중요도를 local적으로 계산하므로, global한 해석 기능 X

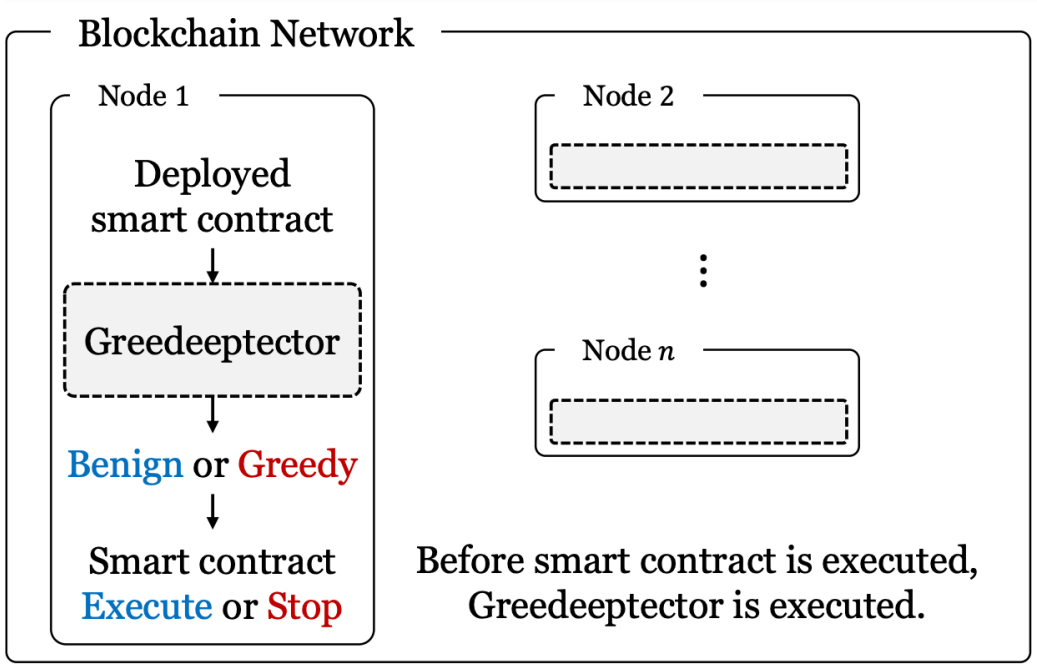
$$IntegratedGrads_i(x) ::= (x_i - x_i') \cdot \int_{\alpha=0}^{1} \frac{\partial F(x' + \alpha \cdot (x - x'))}{\partial x_i} \cdot d\alpha$$

Detection Process

- 스마트 계약을 실행하기 전에 탐지 모델 Greedeeptector를 통해 탐지 수행

  → **정상일 경우 실행 O**

  → **탐욕일 경우 실행 X**



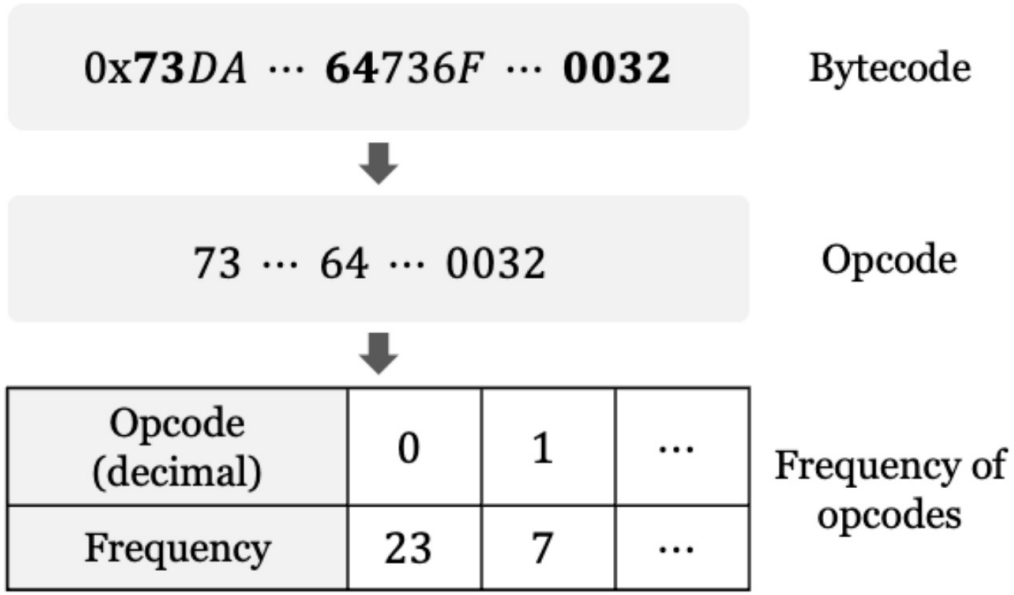
Algorithm 1 Greedeeptector mechanism

Require: Bytecode of smart contract (B_{SC}), Extracted opcodes of smart contract (OP_{sc}), Frequency of opcodes (F_{OP}), Deep learning model for greedy contract detection (`Greedeeptector`)

1: $Opcodes$={00:STOP, ..., FF:SELFDESTRUCT} ▷ Set up dictionary mapping opcodes to bytecodes
2: **for** op in B_{SC} **do**
3: **if** op in $Opcodes$ **then** OP_{sc}.append(op) ▷ Bytecode to opcode [1]
4: Initialize F_{OP} to zero
5: **for** op in OP_{sc} **do**
6: $F_{OP}[op] \leftarrow F_{OP}[op]+1$ ▷ Calculate frequency of opcode
7: **end for**
8: $isGreedy \leftarrow$ `Greedeeptector`(F_{OP}) ▷ Greedy contract detection
9: **if** $isGreedy$ == $True$ **then** Stop the smart contract
10: **else** Execute the smart contract

Base Model

- Pre-processing
  - Bytecode로부터 opcode 추출
  - Frequency of opcodes를 계산하여 빈도수 배열 생성
    → 모델을 생성하기 위한 데이터셋으로 사용



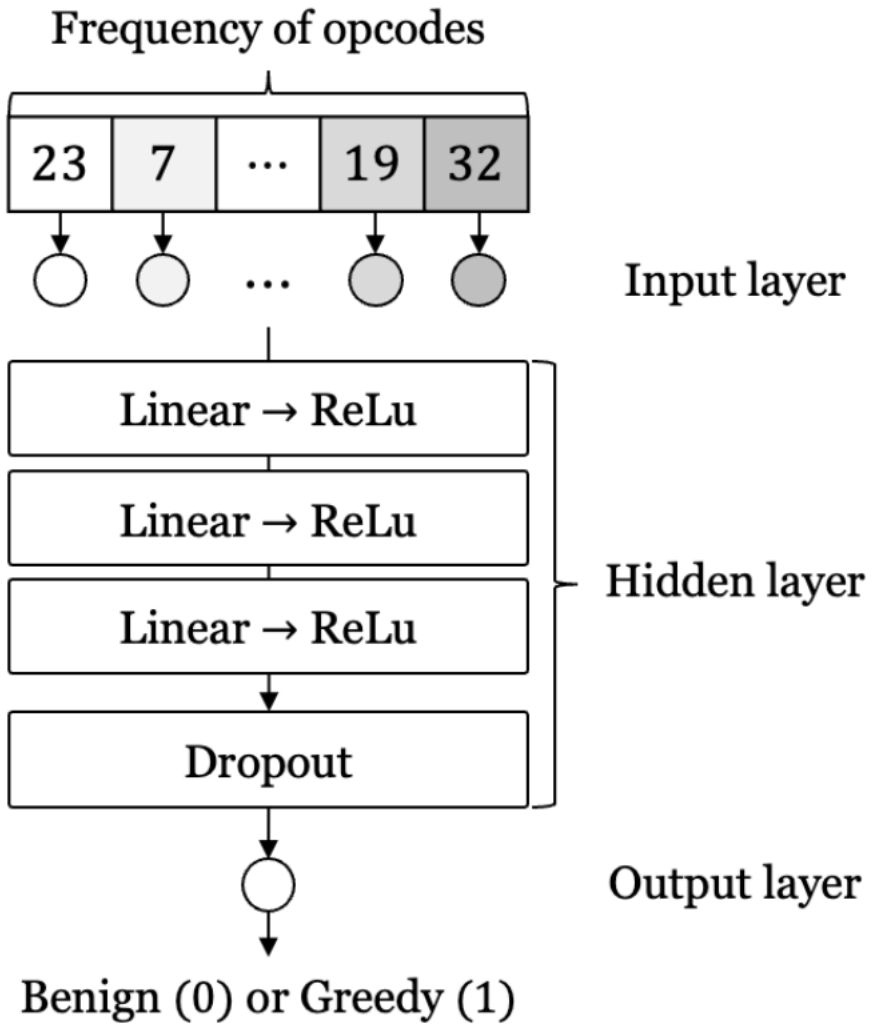
OPCODE	NAME	MINIMUM GAS	STACK INPUT	STACK OUPUT	DESCRIPTION	Expand
00	STOP	0			Halts execution	
01	ADD	3	a b	a + b	Addition operation	
02	MUL	5	a b	a * b	Multiplication operation	
03	SUB	3	a b	a - b	Subtraction operation	
04	DIV	5	a b	a // b	Integer division operation	
05	SDIV	5	a b	a // b	Signed integer division operation (truncated)	
06	MOD	5	a b	a % b	Modulo remainder operation	
07	SMOD	5	a b	a % b	Signed modulo remainder operation	
08	ADDMOD	8	a b N	(a + b) % N	Modulo addition operation	
09	MULMOD	8	a b N	(a * b) % N	Modulo multiplication operation	
0A	EXP	10	a exponent	a ** exponent	Exponential operation	
0B	SIGNEXTEND	5	b x	y	Extend length of two's complement signed integer	

Base Model

- 빈도수 각 배열의 요소는 입력 레이어의 하나의 뉴런에 할당
- 과적합 방지를 위해 dropout
- binary cross-entropy

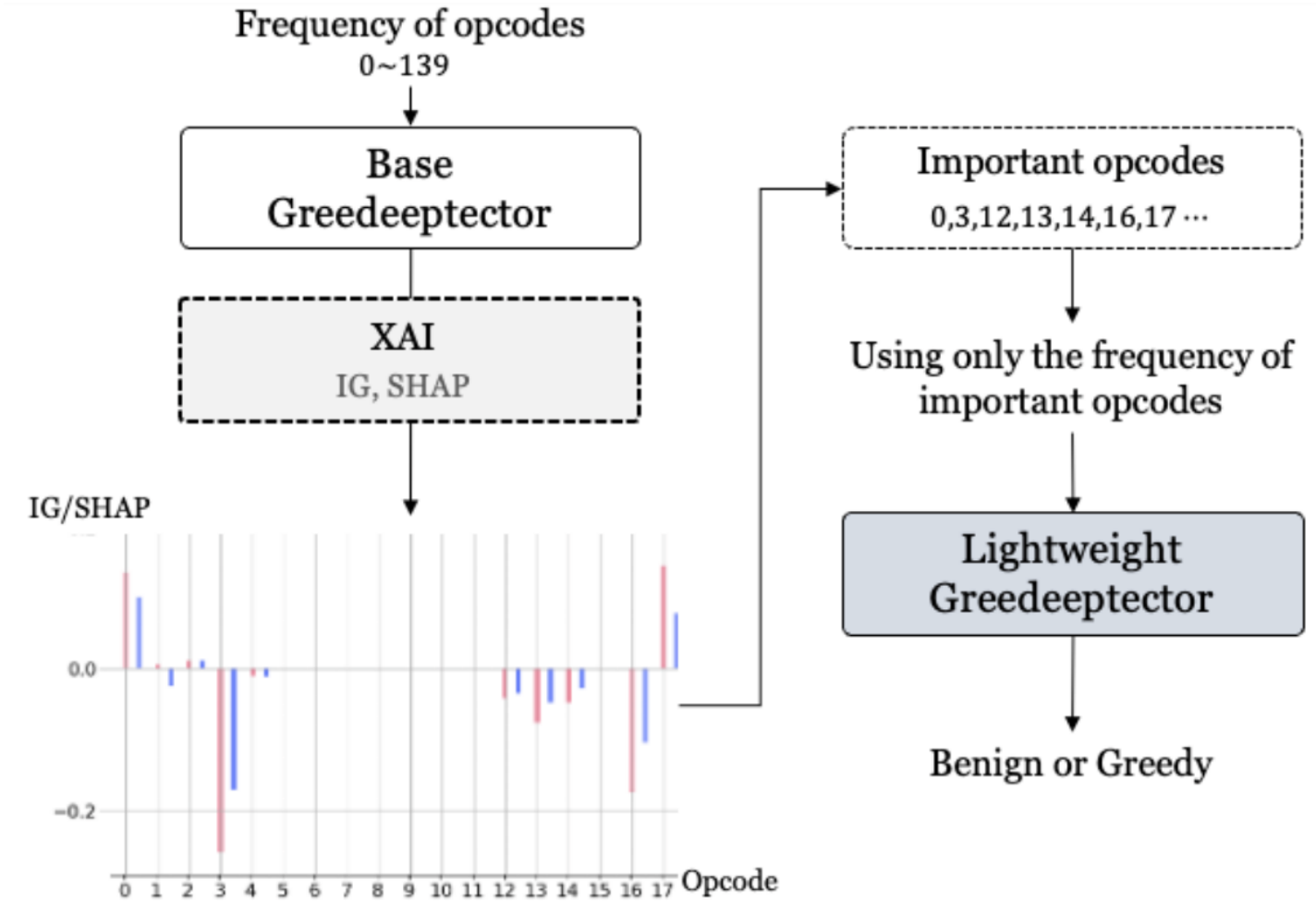
Table 1. Hyperparameters of the base model.

Hyperparameters	Descriptions
Epoch	200
Batch size	256
Units of the input layer	140
Dropout	0.4
Optimizer(learning rate)	Adam(0.0001)



Lightweight Model

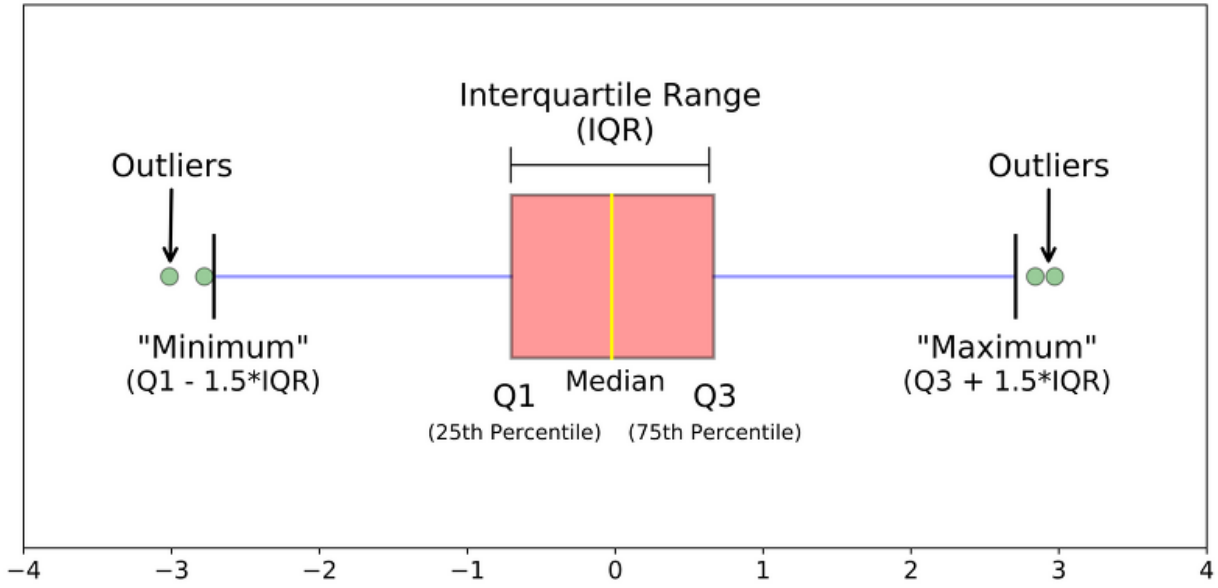
1. opcodes의 frequency 추출
2. Base 모델 구현
3. XAI를 통한 feature 분석
4. 중요한 opcodes 추출
   → 정확도를 저하 X
   → 모델을 경량화
5. lightweight 모델 구현
6. 탐지



Lightweight Model

- 이상치 제거를 위해 IQR 방식 사용
- 이상치 제거 후의 평균 IG와 SHAP에서 상위 n개의 명령어만을 학습

  → 정확도를 손상시키지 않고 모델의 파라미터 수를 줄일 수 있음 (경량화)



Lightweight Model

- IG와 SHAP의 절대값이 큰 opcode 50개 추출

  → 정확도를 하락 시키지 않으면서 모델을 최대한 경량화 하기 위함

  → 140개의 opcode에서 개수를 조금씩 줄이면서 얻은 최적의 opcode 개수

  → 데이터에 따라 조금씩 달라지므로 10번 반복하여 추출함으로써 안정성 높임

- 10번의 시도 중 n번 이상 추출된 명령어들을 경량 모델의 학습 데이터로 사용 (n = 5, 7, 10)

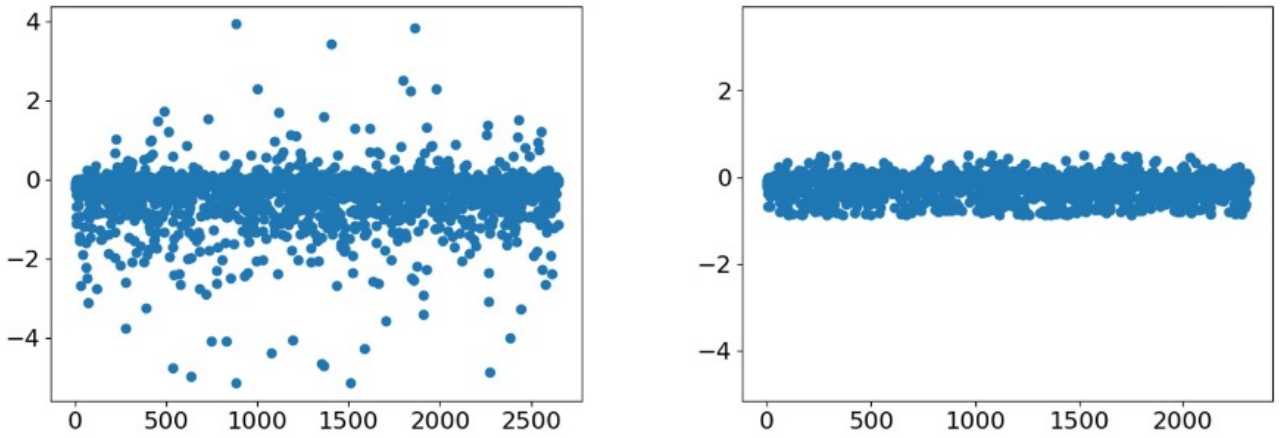


Figure 6. An example of outlier removal using the IQR ; Before (left), After (Right).

Table 4. Comparison of performance according to the number of opcodes used (`Case1`, `Case2`, `Case3` ($n = 5, 7, 10$, respectively)).

	Case1	Case2	Case3
The number of opcodes	58	51	37
Training F1-score	0.93	0.92	0.90
Validation F1-score	0.92	0.92	0.90
Test F1-score	0.92	0.91	0.90
The number of parameters	5,855	4,861	3,167

Lightweight Model Performance

- Base model 대비 **모델 크기 41.5% 감소**
- **모델 파라미터의 개수 61.8% 감소**
- 속도 0.002 ms 향상
- **정확도 0.3% 손실**
- 속도와 메모리 효율성 증가함에 따라 **블록체인 확장성 증가**

  → IoT 블록체인에 적합

Table 6. Comparison between base and lightweight model.

Model	Model size	Parameters	Speed	F1-score
Base	0.89 MB	15,297	0.015 ms	92.6%
Lightweight	0.53 MB	5,855	0.013 ms	92.3%

Instruction Analysis

- JUMP : 무조건 분기
- JUMPI : 조건 분기

  → 조건 분기가 많을 수록 탐욕컨트랙트로 분류될 확률 증가

- JUMPDEST : 분기 대상 주소

Table 7. Top 8 important opcodes in the benign contract.

Algorithm	Sorted by values of IG and SHAP							
	1	2	3	4	5	6	7	8
IG	JUMPDEST	DUP1	SUB	JUMP	EQ	PUSH4	REVERT	SLT
SHAP	DUP1	SUB	JUMP	JUMPDEST	PUSH4	EQ	MLOAD	LT

Table 8. Top 8 important opcodes in the greedy contract.

Algorithm	Sorted by values of IG and SHAP							
	1	2	3	4	5	6	7	8
IG	JUMPI	CALLVALUE	SWAP1	SWAP2	STOP	CALLDATASIZE	AND	ADDRESS
SHAP	JUMPI	PUSH2	CALLVALUE	SWAP2	POP	STOP	SWAP1	CALLDATASIZE

Q & A