석사학위논문

XAI를 사용하여 구현된 경량 딥러닝을 통한 탐욕 컨트랙트 탐지

2024년

한 성 대 학 교 대 학 원

I T 융 합 공 학 과

I T 융 합 공 학 전 공

강 예 준

석 사 학 위 논 문
지도교수 서화정

XAI를 사용하여 구현된 경량 딥러닝을 통한 탐욕 컨트랙트 탐지

Greedy smart contract detection through lightweight deep learning implemented using XAI

2023년 12월 일

한 성 대 학 교 대 학 원

I T 융 합 공 학 과

I T 융 합 공 학 전 공

강 예 준

석 사 학 위 논 문
지도교수 서화정

XAI를 사용하여 구현된 경량 딥러닝을 통한 탐욕 컨트랙트 탐지

Greedy smart contract detection through lightweight deep learning implemented using XAI

위 논문을 공학 석사학위 논문으로 제출함

2023년 12월 일

한 성 대 학 교 대 학 원

I T 융 합 공 학 과

I T 융 합 공 학 전 공

강 예 준

강예준의 공학 석사학위 논문을 인준함

2023년 12월 일

심사위원장 박 명 서 (인)

심 사 위 원 이 웅 희 (인)

심 사 위 원 서 화 정 (인)

국 문 초 록

XAI를 사용하여 구현된 경량 딥러닝을 통한 탐욕 스마트 컨트랙트 탐지

한 성 대 학 교 대 학 원
I T 융 합 공 학 과
I T 융 합 공 학 전 공
강 예 준

스마트 컨트랙트는 블록체인 기반의 디지털 계약이다. 스마트 컨트랙트를 통해 블록체인 네트워크에서 제3자 없이 당사자 간 거래가 가능하다. 하지만 이더리움 네트워크 내에 배포된 스마트 컨트랙트를 분석한 결과, 탐욕 컨트랙트 등 악성 스마트 컨트랙트가 존재하며, 이와 같은 악성 스마트 컨트랙트로 인해 사용자와 블록체인 네트워크에 막대한 피해를 줄 수 있다. 따라서 이에 대한 대응책이 반드시 필요하다. 본 논문에서는 딥러닝 기반의 탐욕 컨트랙트 탐지 시스템을 제안한다. 탐지 모델은 스마트 컨트랙트의 opcode 빈도수를 통해 학습된다. 또한 IoT 블록체인을 위해 저사양 디바이스에서 사용하기 위한 경량 모델을 구현한다. 구현 과정에서 설명 가능한 인공지능(XAI:eXplainable Artifitial Intelligence)을 통해 탐지 시 중요한 opcode를 식별하여 이를 통해 경량 모델을 학습시킨다. 따라서 IoT를 위한 연산 및 메모리 효율적인 경량 모델을 구현하였다. 경량 모델의 크기는 기본 모델 대비 41.5% 감소시켰다. 반면, 경량 모델의 F1-score는 92.3%로 기본 모델

과 차이가 미미하다.

【주요어】 블록체인, 스마트 컨트랙트, 탐욕 컨트랙트, 딥러닝, 설명가능한 인공지능, 경량화, 사물인터넷

목 차

표 목 차

그 림 목 차

알고리즘 목 차

수 식 목 차

제 1 장 서론

제 1 절 연구 목적

스마트 컨트랙트란 블록체인 기반의 디지털 계약을 의미한다. 이러한 스마트 컨트랙트를 통해 제3자의 개입 없이도 계약 당사자끼리 계약을 체결할 수 있다. 하지만 이더리움에 배포된 일부 스마트 컨트랙트를 분석한 결과 탐욕 컨트랙트가 포함되어있다는 사실이 확인되었다. 탐욕 컨트랙트란 이더를 인출할 수 없도록 무기한으로 잠금을 걸어버리는 상태에 도달할 수 있는 악성 스마트 컨트랙트를 의미한다. 따라서 탐욕 컨트랙트를 실행시킬 경우 막대한 피해를 입을 수 있다. 대표적으로 패티리 다중서명 지갑이 동결되는 사건이 있다. 해당 사건은 스마트 컨트랙트에서 발견된 코드 결함으로 인해 약 510,000 이더가 무기한으로 잠긴 사건이다. 이러한 문제는 블록체인 네트워크에 매우 심각한 위협이 되기 때문에 반드시 예방이 필요하다.

이러한 문제를 예방하기 위해서는 악의적인 활동을 탐지하는 방법론이 필요하다. 악의적인 활동을 탐지하는 두 가지 주요 방법론은 악성 노드 탐지와 악성 스마트 컨트랙트 탐지이다. 이 중, 악성 스마트 컨트랙트를 탐지하기 위한 방법론으로 최근 다양한 악성 스마트 컨트랙트 탐지 모델이 제안되었다. 대부분의 딥러닝 기반의 탐지 방식은 스마트 컨트랙트의 특징을 학습하여 악성 스마트 컨트랙트를 탐지한다. 또한 스마트 컨트랙트를 이미지화하고 이를 학습하여 탐지 모델을 구현하는 연구가 수행되기도 하였다.

하지만 블록체인에서 거래 속도는 블록체인의 확장성과 직결되기 때문에, 탐지 속도를 반드시 고려하여야 한다. 만약 탐지에 너무 많은 시간이 소요된다면, 연산 및 메모리 오버헤드를 발생시켜 블록체인의 확장성을 저하시킬 수 있다. 하지만 기존의 딥러닝 기반 탐지 모델은 연산 및 메모리를 고려하지 않고 탐지 정확도만을 높이기 위한 목적으로 설계되었다. 따라서 기존 딥러닝 탐지 모델은 연산 및 메모리 오버헤드를 유발하여 블록체인의 확장성을 저하시킨다. 또한 최근 Internet of Things(IoT) 블록체인에 대한 많은 연구가 진

행되고 있다. IoT 블록체인을 위한 저사양 디바이스에서는 연산과 메모리의 효율성이 매우 중요하다. 블록체인의 확장성 측면에서 연산량이 낮을수록 확장성이 높아진다. 따라서 탐지율만 고려한 모델이 아닌 연산량과 메모리 사용량을 고려한 경량화된 탐지 모델이 반드시 필요하다. 즉, 블록체인의 확장성을 저하시키지 않는 경량 딥러닝 탐지 모델이 필요하다.

본 논문에서는 XAI를 통해 탐욕 컨트랙트 탐지 시 모델의 예측에 영향을 크게 미치는 중요 opcode를 식별한다. 또한 예측에 영향을 크게 미치는 중요 opcode를 통해 모델을 학습시킴으로써, IoT를 위한 경량화된 탐지 모델을 제안한다. 경량 탐지 모델은 블록체인의 확장성 저하를 최소화하고, 정확도 측면에서 기본 모델과 거의 비슷한 성능을 유지하면서, 연산량과 메모리 사용량을 고려하여 효율적인 방식으로 탐욕 컨트랙트를 탐지한다.

본 논문은 다음과 같이 구성된다. 1장에서는 연구 목적과 본 논문의 연구 기여에 대해 서술한다. 2장에서는 인공신경망, 스마트 컨트랙트 등 관련 기술과 선행 연구에 대해서 서술한다. 3장에서는 탐욕 컨트랙트를 탐지하기 위해 제안하는 방법론을 소개한다. 4장에서는 기본 모델과 경량 모델의 비교 분석 및 opcode의 심층 분석에 대해 서술한다. 마지막으로 5장에서는 논문의 결론을 맺는다.

제 2 절 연구 기여

1) XAI을 활용한 스마트 컨트랙트 opcode의 심층 분석

XAI 알고리즘 종류인 Integrated Gradient(IG)와 Gradient SHAP(GS)을 통해, 모델의 예측에 영향을 미치는 opcode를 확인한다. 또한 정상 컨트랙트와 탐욕 컨트랙트에 대한 몇 개의 중요 opcode를 심층적으로 분석하여, 어떤 opcode가 중요한 특징으로써 작용하는지와 자주 사용되는지 분석한다.

2) 중요한 opcode를 기반으로 한 경량화된 모델 구현

XAI를 통해 중요 opcode만 추출하여 학습 데이터로 사용할 경우, 데이터 차원이 기존보다 작아진다. 이러한 중요 opcode를 사용하여 모델을 학

습시킴으로써 경량화된 모델을 구현한다. 경량화된 모델은 이전 모델보다 약 40% 경량화되었으며 정확도 손실은 거의 없다.

3) 스마트 컨트랙트 실행 시 탐지를 통해 블록체인 네트워크의 안정성 향상

이전 연구와 달리, 본 논문에서 제안한 기법은 스마트 컨트랙트가 실행될 때 탐욕 컨트랙트 탐지를 수행한다. 즉, 새로 배포될 스마트 컨트랙트뿐만 아니라 이미 배포된 스마트 컨트랙트에 대해서도 탐지를 수행할 수 있다. 따라서 블록체인 네트워크의 안정성이 향상된다.

제 2 장 관련 연구

제 1 절 인공 신경망

인공 신경망은 두뇌의 뉴런 구조를 컴퓨터로 구현한 것을 의미한다. 딥러닝은 여러 층의 인공 신경망을 쌓아 데이터를 통해 학습을 수행하며 입력층, 은닉층 그리고 출력층으로 이루어져 있다. 입력층은 학습하고자 하는 데이터를 입력받는 층을 의미한다. 입력 받은 데이터를 통해 은닉층에서 가중치를 계산하며, 출력층을 통해 최종적인 결과가 출력된다. 딥러닝은 머신러닝과 다르게 데이터로부터 특징을 스스로 추출하고 학습한다. 이러한 특성으로 인해 딥러닝은 컴퓨터 비전, 음성 인식, 자연어 처리, 신호 처리 등 다양한 분야에서 활용되고 있다. DNN(Deep Neural Network)과 CNN(Convolutional Neural Network)은 대표적인 딥러닝 모델로 분류 문제에 주로 사용된다. 이 외에도 시계열 데이터 학습 시 주로 사용되는 RNN(Recurrent Neural Networks), LSTM(Long Short-Term Memory) 그리고 Transformer가 있다. 또한 GAN(Generative Adversarial Network)과 같은 생성형 신경망도 존재한다.

제 2 절 경량 딥러닝

경량 딥러닝은 깊고 복잡한 레이어로 구성된 모델의 성능을 유지하면서 모델의 크기를 줄이고 계산 복잡도를 줄인 딥러닝 모델이다. 저사양 IoT 디바이스에서 경량이 아닌 일반 딥러닝 모델을 사용할 경우, 수많은 매개변수를 로딩하는 과정에서 메모리 오버헤드가 발생할 수 있다. 따라서 이러한 경량 딥러닝은 모바일 환경, 자율주행차, 로봇 등 컴퓨팅 자원이 제한된 저사양 IoT 디바이스에 필수적이다. 본 논문에서는 XAI를 활용하여 경량 딥러닝 모델을 구현한다.

제 3 절 Explainable Artificial Intelligence (XAI)

인공지능은 블랙박스 모델로, 모델 내부에서 데이터의 복잡한 관계와 특징을 학습한다. 따라서 AI 모델이 내린 예측에 대한 근거가 무엇인지 명확하지 않다. 하지만 의료, 금융, 법률 등 중요한 결정을 내려야 하는 분야에서는 AI 결과물에 대한 명확한 근거가 필요하다. XAI는 이러한 문제를 해결하기 위해 만들어진 기술로, 딥러닝 모델이 내리는 예측에 대한 근거를 제공한다. 따라서 XAI를 통해 딥러닝 기술에 대한 신뢰성을 높일 수 있으며, 필요한 결과를 달성하기 위해 디버깅을 용이하게 할 수 있다는 장점이 있다. 또한 XAI를 통해 예측을 위해 어떠한 특징이 중요한지 확인할 수 있으므로 학습 결과에 대한 이해도를 높여 더 나은 모델을 구현할 수 있다. 즉, XAI는 불필요한 특징을 제외하고 꼭 필요한 특징만으로 딥러닝 모델을 학습시킬 수 있다. 이러한 방식으로 딥러닝 모델을 최적화하고 경량 모델을 구현할 수 있다. 따라서 XAI는 효율적이고 가벼운 모델을 구현하는 데 유용하다.

본 논문에서는 다양한 XAI 알고리즘을 적용하기 위해 Captum을 사용한다. Captum은 Pytorch에서 사용할 수 있는 모델 해석 오픈 소스 라이브러리이다. XAI에는 여러 가지 알고리즘(e.g. Integrated Gradients, Gradient SHAP 등)이 존재한다. 본 논문에서는 Integrated Gradients와 Gradient SHAP 알고리즘을 활용하여 경량 딥러닝 모델을 구현한다.

Integrated Gradients(IG)는 입력값과 베이스라인간의 차와 기울기 정보를 누적하여 곱함으로써, 각 특징의 중요도를 계산하는 XAI 알고리즘이다. IG는 입력된 데이터의 각 특징값의 중요도를 지역적으로 계산하므로 전역적인 해석을 제공하지 않는다. IG는 민감도 및 구현 불변성 조건을 모두 만족한다. 민감도 조건은 베이스라인과 입력의 차이가 오직 하나의 특징일 경우, 모델이 두 입력에 대해 다르게 예측한다면 non-zero attribution을 가져야한다는 조건이다. 구현 불변성 조건은 동일한 입력값에 대해 동일하게 예측하는 서로 다른 모델이 있을 때, 두 모델은 같은 입력값에 대해 일정한 attribution을 가져야한다는 조건이다. IG는 구현이 간단하다는 장점이 있으며 텍스트, 이미지 등 다양한 데이터셋에 대해 적용 가능하다.

Gradient SHAP은 기울기를 통해 Shapley 값을 근사화하는 XAI 알고리즘이다. Shapley 값은 게임 이론을 기반으로 게임에 참여하는 각 플레이어의 기여도를 계산한다. 즉, Shapley 값은 특정 플레이어가 다른 플레이어와 협력할 때의 기여도를 나타낸다. 최근 Shapley 값은 딥러닝 모델을 해석하는 데 주로 사용된다. 이를 통해 특징이 모델의 예측에 얼마나 기여하는지를 확인함으로써 모델을 해석할 수 있다. Shapley 값은 지역적 및 전역적으로 분석할 수 있으므로 특징을 분석하는데 매우 유용하다. 또한 Shapley 값은 입력값 간의 상호작용을 고려하여 정확도가 높다는 장점이 있다. 그러나 특징의 개수가 증가할수록 계산량이 기하급수적으로 증가한다는 단점도 존재한다. 이러한 단점을 극복하기 위해 Gradient SHAP은 기울기를 통해 Shapley 값을 계산한다.

제 4 절 블록체인

블록체인은 네트워크 내의 참여자가 P2P(Peer to peer) 방식으로 동일한 원장을 공유하는 분산 원장 네트워크이다. 즉, 블록체인은 중앙 서버가 데이터를 관리하는 방식이 아닌, 네트워크 내의 노드들이 각각 원장을 소유하는 분산형 방식이다. 또한 노드들이 직접 트랜잭션이 포함된 블록을 검증함으로써, 제3자 없이도 거래가 정상적으로 이루어지기 때문에 서버가 필요하지 않다. 해커가 데이터를 조작하기 위해서는 네트워크 내에 있는 노드의 블록체인 중 51% 이상을 조작해야 한다. 이는 사실상 불가능하므로 트랜잭션의 무결성이 보장된다. 이러한 장점으로 인해 블록체인은 디지털 자산 거래, 의료, 등 다양한 분야에서 활용되고 있다.

제 5 절 스마트 컨트랙트

스마트 컨트랙트란 블록체인 기반의 디지털 계약을 의미한다. 기존 서면을 통해 이루어지던 계약을 코드로 구현한 것이 스마트 컨트랙트이다. 스마트 컨트랙트는 특정 조건이 충족될 경우에만 자동으로 계약이 실행된다. 따라서 제3자인 인증기관이 개입하지 않고, 두 당사자가 스마트 컨트랙트를 통해 계약

을 체결함으로써 블록체인이 추구하는 탈중앙화가 실현된다.

스마트 컨트랙트는 다양한 고급 프로그래밍 언어를 통해 작성될 수 있으며 대표적으로 Solidity 언어를 통해 작성된다. 고급 프로그래밍 언어를 통해 작성된 코드는 컴파일러를 통해 이더리움 바이트코드로 변환된 후에 블록체인에 배포된다. 이더리움 바이트코드는 opcode와 값으로 구성된다. opcode는 컴퓨터가 수행할 명령어를 나타내며, 현재 이더리움에는 140개의 opcode가 존재한다. 값은 명령어를 수행할 대상이다. 배포된 스마트 컨트랙트는 이더리움 가상머신 (Ethereum Virtual Machine:EVM)을 통해 실행된다. EVM은 스마트 컨트랙트를 위한 실행 환경을 제공하고, 바이트 코드 형태의 스마트 컨트랙트를 실행시키는 역할을 한다. 추가적으로 스마트계약을 실행하기 위해서는 거래에 대한 일정량의 수수료를 지불해야 한다. 이는 스마트계약 실행의 계산 비용으로써, 채굴자에게 지급되는 암호화폐이다.

스마트 컨트랙트는 배포된 이후에 삭제하거나 코드를 수정할 수 없다. 이는 스마트 컨트랙트가 불변성의 특성을 가지고 있는 블록체인에 배포되기 때문이다. 블록체인에 저장된 정보는 영구적으로 유지되고, 수정할 수 없다. 즉, 스마트 컨트랙트 코드에 오류나 보안 취약점이 발견되더라도 수정이 불가능하기 때문에, 이러한 스마트 컨트랙트를 실행시킬 경우 큰 문제를 야기할 수 있다. 따라서 악성 스마트 컨트랙트를 탐지할 수 있는 시스템이 반드시 필요하다.

제 6 절 탐욕 컨트랙트

탐욕 컨트랙트란 이더가 컨트랙트 내에 무기한으로 잠겨져서 이더를 인출할 수 없게 되는 스마트 컨트랙트이다. 탐욕 컨트랙트는 이더를 전송 받을 수는 있지만, 전송 받은 이더를 처리하는 명령어가 없거나 해당 명령어까지 도달하지 못해 이더를 컨트랙트 내에 잠근다. 따라서 탐욕 컨트랙트로 전송된 이더는 스마트 컨트랙트를 배포한 노드 조차 다시 되찾을 수 없으며, 이를 실행시킬 경우 막대한 피해가 발생할 수 있다.

제 7 절 악성 스마트 컨트랙트 탐지 동향

ACSAC'18에서는 악성 스마트 컨트랙트를 탐욕, suicidal 그리고 prodigal contract 세 가지로 분류하였다. 또한 저자는 악성 스마트 컨트랙트를 심볼릭 분석 방법으로 탐지하는 도구인 MAIAN을 구현하였다. MAIAN 도구는 대표적인 스마트 컨트랙트 탐지 도구 중 하나이다. 해당 도구를 통해 스마트 컨트랙트의 바이트 코드를 처리하여 버그를 유발하는 컨트랙트를 탐지한다. 심볼릭 분석 방식에서는 바이트 코드의 시작 명령어부터 종료 명령어까지의 모든 경로를 추적한다. 그 후, 종료 명령어(예. STOP, RETURN)를 찾을 때까지 순차적으로 실행한다. 명령어 시퀀스를 추적하는 중에 함수 호출이 발생하면 분기 조건을 고려하여 탐색한다. 종료 명령어가 유효하지 않을 경우, 다른 경로를 탐색하기 위해 깊이 우선 탐색 과정에서 역추적한다. 그 후, 심볼릭 분석의 결과를 검증하기 위한 concrete validation이 수행된다. 이를 위해, 프라이빗 포크를 통해 생성되는 이더리움 테스트 네트워크에서 악성 스마트 컨트랙트 후보들을 실행시킨다. 그리고, 세 종류의 악성 컨트랙트를 구별하기 위한 작업이 수행된다. 해당 과정에서도 악성 컨트랙트로 판단된다면, MAIAN 도구는 최종적으로 해당 컨트랙트를 악성으로 분류한다. MAIAN 도구는 파이썬으로 개발되었으며, 평균적으로 하나의 스마트 컨트랙트를 10초 이내로 탐지한다.

Mathematics'23에서는 악성 노드를 먼저 탐지한 후, 악성 컨트랙트를 탐지하는 시스템을 제안하였다. 이는 악성 컨트랙트 탐지 분야에서 가장 최근에 게제된 논문이다. 해당 시스템에서 사용자가 악성 컨트랙트를 배포할 경우, 해당 노드는 악성 노드로 분류되어 더 이상 컨트랙트를 배포할 수 없다. 악성 스마트 컨트랙트를 탐지하기 위한 모델로는 LSTM, GRU 그리고 ANN이 사용된다. LSTM, GRU 그리고 ANN의 AUC는 각각 0.99, 0.99 그리고 0.97을 달성하였다. 하지만 해당 시스템에서는 스마트 컨트랙트를 배포하기 전에 탐지를 수행하므로, 이미 배포된 스마트 컨트랙트에 대해서는 탐지할 수 없다는 한계점이 존재한다.

ICSE'18에서는 솔리디티 코드를 XML 기반의 중간 언어로 변환하여

XPath 패턴을 분석하는 SmartCheck가 제안되었다. SmartCheck는 XPath를 분석하여 스마트 컨트랙트의 취약점을 탐지한다. 하지만 XPath 패턴이 없는 취약한 스마트 컨트랙트는 탐지가 어렵다는 치명적인 한계점을 가지고 있다.

SNC'21에서는 블록체인 네트워크에 공유 자식 노드를 도입함으로써 머신러닝을 기반으로 스마트 컨트랙트의 취약성을 분석하는 모델을 제안하였다. 해당 모델에서는 Re-entrancy, Arithmetic, Access Control, Denial of Service, Unchecked Low Level Calls, Bad Randomness, Front Running, Denial of Service와 같은 여덟 가지 취약점을 탐지할 수 있다. 하지만 충분한 개수의 악성 스마트 컨트랙트를 확보하지 않은 채로 모델을 학습시켜, 모델이 안정적이지 않다는 단점이 존재한다.

제 3 장 제안 기법

스마트 컨트랙트 탐지 도구 중 잘 알려진 MAIAN 도구와 더불어 다양한 방식의 탐지 기법이 제안되었다. 가장 최근에 제안된 딥러닝 기반 탐지 기법인 이전연구에서도 악성 스마트 컨트랙트를 탐지할 수 있으나, 모델 구현에 사용된 데이터셋의 수가 매우 적다. 따라서 모델의 신뢰도가 높지 않으며, 견고하다고 보기 어렵다. 또한 이미 배포된 스마트 컨트랙트에 대해서는 탐지를 수행할 수 없다는 한계점이 존재한다.

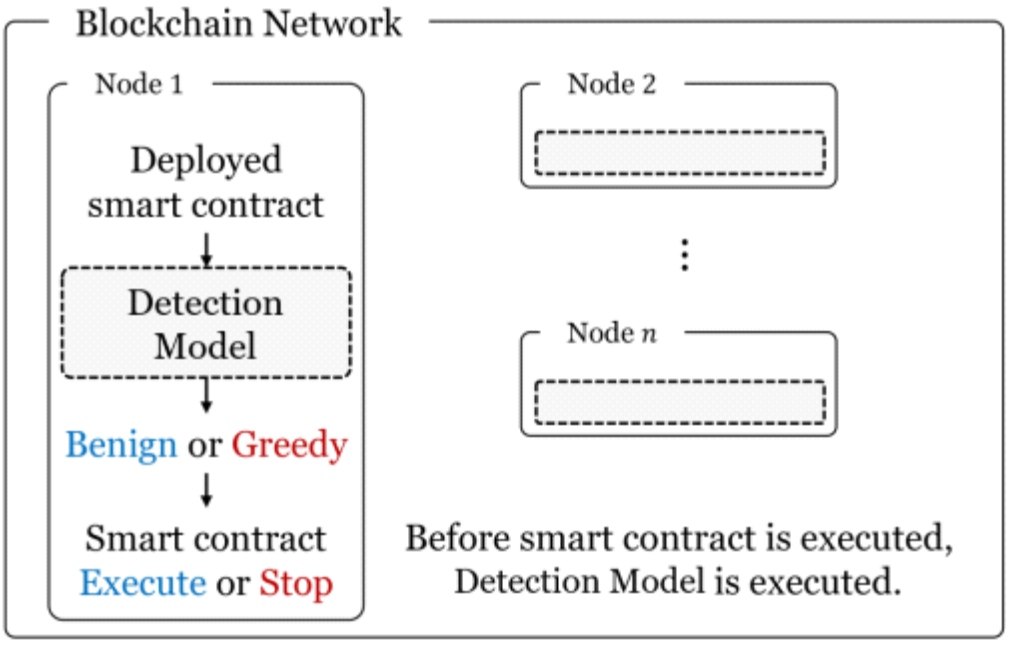
본 논문에서는 XAI를 통해 구현된 경량 딥러닝을 사용하여 탐욕 컨트랙트를 탐지하는 시스템을 제안한다. 또한 제안하는 시스템은 이미 배포된 스마트 컨트랙트에 대해서도 탐지를 수행할 수 있다는 장점이 있다. 따라서 제안된 기법은 현재 이더리움 네트워크에서 스마트 컨트랙트의 안전한 실행을 보장한다.

또한 본 논문에서는 블록체인에서 효율적으로 작동하는 경량 탐지 시스템을 설계하고, 해당 시스템을 평가하고 논의한다. 제안하는 시스템은 블록체인 네트워크에 적용된다. 즉, 딥러닝 모델은 블록체인 노드(예. 컴퓨터, 저사양 디바이스)에서 실행된다. 실제 블록체인 네트워크에서의 성능 평가는 향후 수행할 계획이다.

제 1 절 탐욕 컨트랙트 탐지 시스템 제안

[그림 3-1]과 [알고리즘 3-1]은 본 논문에서 제안하는 시스템의 메커니즘을 보여준다. 각 노드는 이더리움 네트워크의 스마트 컨트랙트에 대한 정보를 확인할 수 있다. 따라서 노드가 스마트 컨트랙트를 실행하기 전에 탐지 모델을 통해 탐욕 컨트랙트를 탐지한다. 이를 위해 전처리 과정이 수행된다. [알고리즘 3-1]의 2번째 줄에서 확인할 수 있듯이 스마트 컨트랙트의 바이트코드는 opcode로 변환된다. 그 후 각 opcode의 사용 빈도가 계산된다. 마지막으로 스마트 컨트랙트의 opcode 빈도수가 탐지 모델에 입력된다. 결과

(isGreedy)가 0이면 정상 스마트 컨트랙트이므로 실행된다. 반대로 탐욕 컨트랙트라면 실행하지 않는다.



[그림 3-1] 스마트 컨트랙트 탐지 시스템 메커니즘

Algorithm 1 Smart contract detection system mechanism

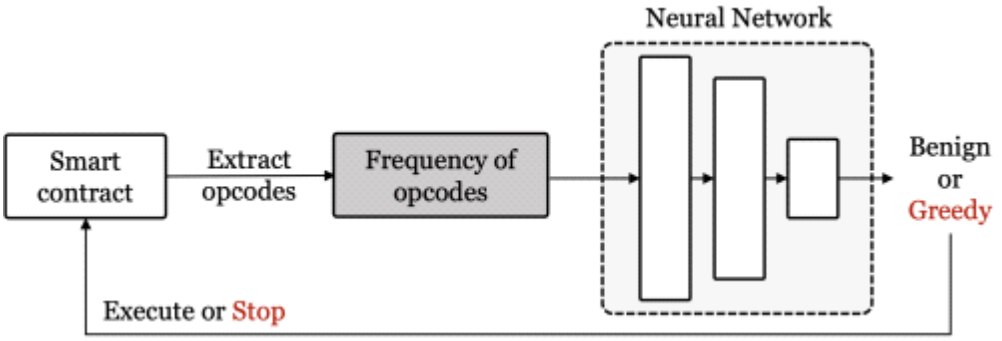
Require: Bytecode of smart contract(B_{sc}), Extracted opcodes of smart contract(OP_{sc}), Frequency of opcodes(F_{OP}), Deep learning model for greedy contract detection($Model$)

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**
4: OP_{sc}.append(op) ⇨Bytecode to opcode
5: **end if**
6: **end for**
7: Initilaize F_{OP} to zero
8: **for** op in OP_{sc} **do**
9: $F_{OP}[op] \leftarrow F_{OP}[op]+1$ ⇨Calculate frequency of opcode
10: **end for**
11: isGreedy ← $Model(F_{OP})$ ⇨Greedy contract detection
12: **if** isGreedy == $True$ **then**
13: Stop the smart contract
14: **else**
15: Execute the smart contract
16: **end if**

[알고리즘 3-1] 스마트 컨트랙트 탐지 과정

제 2 절 탐욕 컨트랙트 탐지 기본 모델 구현

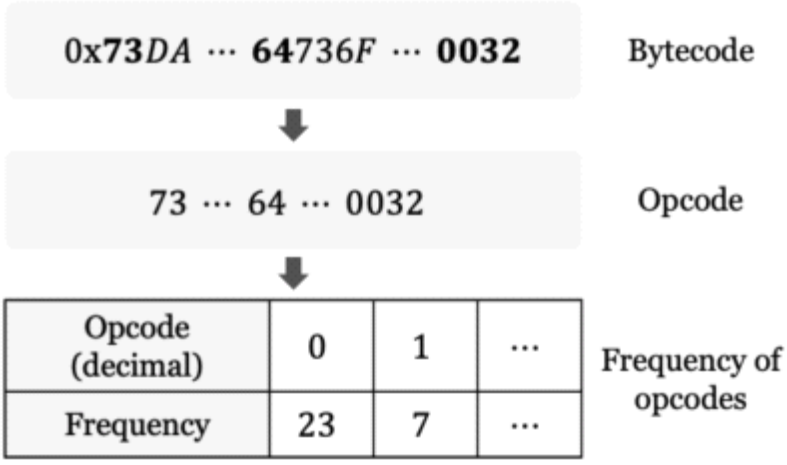
본 장에서는 악성 스마트 컨트랙트 탐지 기본 모델을 제안한다. [그림 3-2]는 스마트 컨트랙트 탐지 모델의 다이어그램을 보여준다. 첫번째로 스마트 컨트랙트에서 opcode를 추출한다. 추출된 opcode는 각 opcode의 빈도수를 나타내는 배열로 변환된다. 각 opcode의 빈도수는 신경망에 입력되고, 신경망은 정상 컨트랙트와 탐욕 컨트랙트를 분류한다. 만약 탐지 대상인 스마트 컨트랙트가 탐욕 컨트랙트로 분류될 경우 해당 스마트 컨트랙트의 실행이 중지된다.



[그림 3-2] 스마트 컨트랙트 탐지 모델 다이어그램

1) 전처리 과정

앞서 설명하였듯이, 탐욕 컨트랙트를 탐지하기 위해서는 스마트 컨트랙트에서 사용된 opcode의 빈도수가 필요하다. 따라서 전처리를 통해 스마트 컨트랙트의 bytecode로부터 빈도수를 추출해야 한다. [그림 3-3]은 전처리를 통해 바이트코드를 opcode의 빈도수로 변환하는 과정을 보여준다. 스마트 컨트랙트는 사용되는 opcode와 연산에 사용되는 값들로 이루어진 bytecode로 표현될 수 있다. 그 중 값을 제외한 opcode만을 추출한다. 그 후, 스마트 컨트랙트에서 사용된 opcode들의 시퀀스가 생성된다. 하나의 opcode는 1바이트이므로, 총 256개의 opcode(00~FF)가 존재할 수 있다. 하지만 이더리움에서는 140개의 명령어만을 제공한다. 따라서 본 논문에서는 길이가 140인 opcode 빈도수 배열을 생성한다. 빈도수 배열의 인덱스는 인덱스에 해당하는 이더리움 opcode를 의미하고, 값은 빈도수를 의미한다. 즉, 배열은 스마트 컨트랙트에서 opcode가 몇 번 사용되는지에 대한 빈도수를 담고있다. 생성된 opcode 빈도수 배열은 탐욕 컨트랙트 탐지 모델에 입력된다.

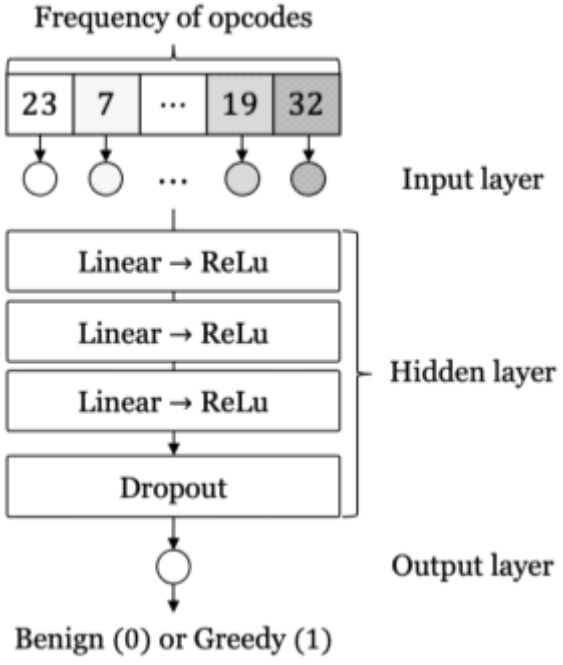


[그림 3-3] 전처리를 통한 바이트코드를 opcode의 빈도수로 변환하는 과정

2) 탐지 과정

탐욕 컨트랙트 탐지에는 딥러닝 모델이 사용된다. [그림 3-4]는 기본 모델의 구조를 보여준다. 본 논문에서 제안하는 시스템은 블록체인 확장성 저하를 최소화 한 채로 탐지를 수행하도록 설계된다. 순환 신경망과 같은 시계열 모델의 크기는 크기 때문에, 보다 더 가벼운 심층 신경망을 통해 탐지를 수행한다. 이때, 빈도수 배열에서의 각 요소는 입력층에서 하나의 뉴런에 할당된다. 즉, 입력층의 뉴런의 수는 이더리움의 opcode의 수만큼 필요하다. 다음으로, 은닉층으로써 완전 연결 계층인 선형 레이어가 사용된다. 그리고 출력층으로 전달되기 전에, 과적합을 방지하기 위해 임의의 뉴런을 탈락시키는 dropout 레이어를 거친다. 출력층에서는 시그모이드 활성화 함수를 통해 데이터에 대한 예측값으로써 0(정상)에서 1(탐욕)사이의 값을 출력한다. [수식 3-1]은 시그모이드 활성화 함수의 공식을 보여준다. 마지막으로, 출력값에 대해 이진 크로스 엔트로피 손실 함수를 통해 실제값과 예측값 간의 손실을 계산한다. [수식 3-2]는 이진 크로스 엔트로피 손실 수식을 보여준다. y hat은 0과 1 사이의 연속 시그모이드 함수 출력값을 의미하고, y는 불연속적인 실제 값을 의미한다. 그리고 손실을 최소화하기 위해 네트워크를 갱신한다. 이러한 학습 과정을 통해 탐욕 컨트랙트와 정상 컨트랙트를 구별할 수 있는

신경망이 구현된다. [표 3-1]는 기본 모델의 하이퍼파라미터를 보여준다. 입력층의 수는 140(opcode의 수)이다. 또한, 과적합 방지를 위해 40%의 Dropout을 사용하며, Epoch을 200으로 설정한다.



[그림 3-4] 기본 모델의 구조

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

[표 3-1] 기본 모델의 하이퍼파라미터

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

[수식 3-1] 시그모이드 활성화 함수

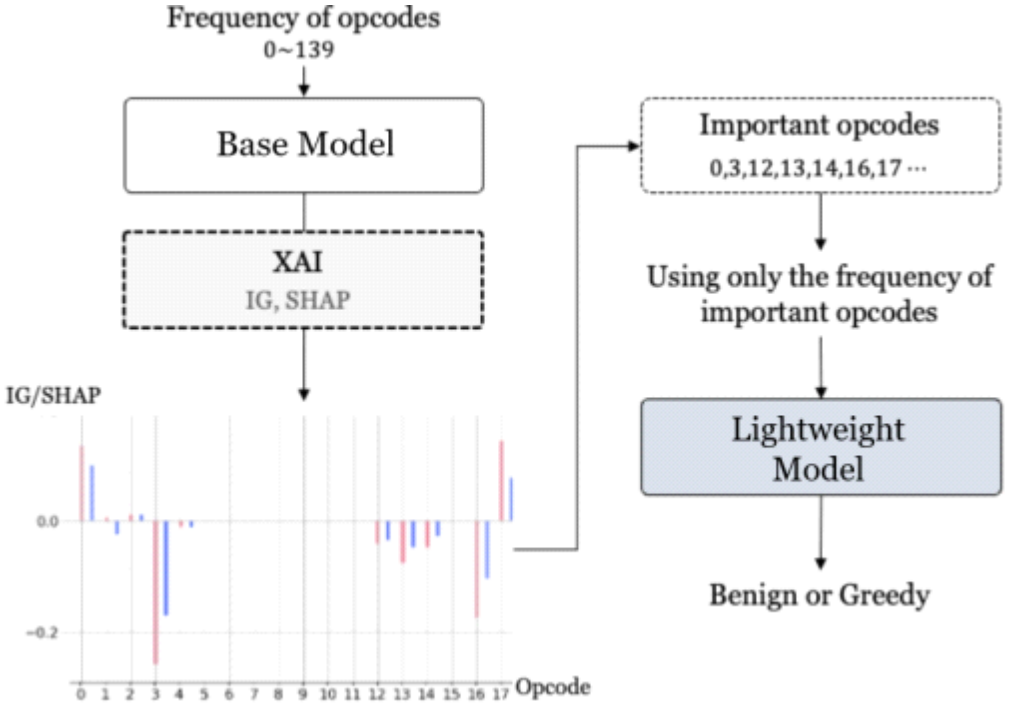
$$BCE(y,\hat{y}) = -(ylog(\hat{y}) + (1-y)\log(1-\hat{y}))$$

[수식 3-2] 이진 크로스 엔트로피

제 3 절 XAI를 통해 구현된 경량 모델

본 논문에서는 XAI를 사용하여 연산 및 메모리 효율적인 경량 모델을 구현한다. 경량 모델의 전처리 및 탐지 과정은 기본 모델과 동일하다. [그림 3-5]는 XAI를 사용하여 경량 모델을 구현하는 과정을 보여준다. 먼저 140개의 opcode 빈도수를 사용하여 기본 모델을 구현하였다. 이후, XAI 알고리즘인 IG와 SHAP 값을 계산한다. [그림 3-5]에서 그래프 내의 빨간색은 IG 값을 나타내고, 파란색은 SHAP 값을 나타낸다. 원본 그래프에서는 140개의 모든 opcode에 대한 값이 존재하지만, 논문에서는 일부만을 캡쳐하였다. 각 IG

와 SHAP 값은 입력 데이터의 각 특징이 예측에 미치는 영향을 나타내는 수치이며, 절대값이 클수록 중요한 특징임을 의미한다. 또한 양수 값은 해당 특징이 탐욕 컨트랙트에 기여함을 의미하고, 음수 값은 정상 컨트랙트에 기여함을 의미한다. 따라서 XAI를 통해 정상 컨트랙트와 탐욕 컨트랙트를 분류하는 것에 크게 영향을 미치는 중요한 opcode를 식별할 수 있다.



[그림 3-5] 경량 모델 구현 과정; 빨간색: IG 값, 파란색: SHAP 값

경량 모델은 전체 140개의 opcode 중에서 중요한 opcode들만 사용한다. 즉, 140개보다 적은 수의 opcode를 사용한다. 이를 통해 입력 데이터 특징의 개수가 감소하기 때문에, 더 경량화된 모델을 구현할 수 있다. 결과적으로 입력층에서의 뉴런의 수가 감소하고 전체 모델의 파라미터 수가 감소한다. 이때, 탐지 정확도를 저하시키지 않으면서 모델의 파라미터를 최대한 줄일 수 있도록 하는 최소한의 opcode를 사용하여야 한다. 따라서 실험을 통해 상위 n개의 opcode를 선택한다.

학습에는 많은 수의 데이터 샘플이 사용되기 때문에, 그 중 특정 몇 개의

샘플은 이상치를 가질 수 있다. 이상치가 존재하는 경우, 특정 샘플의 IG와 SHAP 값이 평균에 큰 영향을 미치기 때문에, 신뢰도를 위해 이상치를 제거해야 한다. 이상치를 제거하기 위해 IQR(Inter Quantile Range)을 사용한다. IQR 방식에서는 전체 데이터를 정렬하여 4등분한 뒤, 25%($Q1$)와 75%($Q3$) 지점 사이의 값을 IQR이라고 한다. 즉, $IQR = Q3 - Q1$이다. $Q1 - 1.5 \cdot IQR$을 최소값으로 설정하고, $Q1 + 1.5 \cdot IQR$을 최대값으로 설정한다. 이때 최소값과 최대값을 벗어나는 데이터는 이상치로 간주된다. IQR과 곱해지는 상수는 일반적으로 1.5이다. 요약하자면, 경량 모델은 이상치 제거 후의 평균 IG와 SHAP 값을 계산하여 상위 n개의 opcode만을 학습함으로써 정확도를 유지한 채로 파라미터의 수를 감소시킴으로써 구현한다.

[표 3-2]는 XAI를 통해 설계된 경량 모델의 하이퍼파라미터를 보여준다. 경량 모델의 하이퍼파라미터는 최적의 성능을 달성하기 위해 실험을 통해 설정한다. 경량 모델은 기본 모델과 동일한 에폭, 배치 크기, 드롭아웃 비율, 옵티마이저 그리고 학습률을 사용한다. 즉, 경량 모델과 기본 모델의 하이퍼파라미터 차이는 입력층과 은닉층의 뉴런 개수이다. 본 논문에서는 하나의 opcode 빈도수를 하나의 뉴런에 할당한다. 경량 모델에서는 중요한 opcode의 빈도수만을 데이터로 사용하기 때문에, 입력층의 뉴런의 개수가 기본 모델보다 줄어든다. 즉, 입력 데이터의 특징 수(중요 opcode의 수)가 감소하기 때문에, 모델의 뉴런 수가 감소한다. 결과적으로 경량 모델의 크기와 계산 복잡도를 줄일 수 있다. 중요 opcode는 실험적으로 설정되었으며, 4장에서 상세히 소개한다.

Hyperparameter	Descriptions
Epoch	200
Batch size	256
Units of the input layer	58
Dropout	0.4
Optimizer(learning rate)	Adam(0.0001)

[표 3-2] 경량 모델의 하이퍼파라미터

제 4 장 실험 및 성능 평가

제 1 절 실험 환경 및 데이터셋

본 실험을 위해 클라우드 기반 서비스인 Google Colaboratory PRO+를 사용한다. 운영체제는 Ubuntu 20.04.5 LTS이다. 또한 RAM은 15GB, GPU는 Tesla T4를 사용한다. 딥러닝 프레임워크는 Pytorch 2.0.0, Python은 3.9.16을 사용한다.

본 실험에서는 모델을 생성하기 위한 데이터셋으로 Google에서 제공하는 Google BigQuery 데이터셋[1]을 사용한다. Google BigQuery 데이터셋에서 이더리움에 배포된 스마트 컨트랙트를 수집한다. 악성 스마트 컨트랙트 탐지 도구인 MAIAN을 통해 14,716개의 정상 컨트랙트와 탐욕 컨트랙트 데이터셋이 구성된다. 데이터셋은 정상 컨트랙트와 탐욕 컨트랙트가 1:1 비율로 구성된다.

제 2 절 기본 모델 성능 평가

[표 4-1]은 140개의 opcode를 사용한 기본 모델의 성능을 보여준다. 기본 모델은 평균 Test F1-score 92.6%를 달성하였다. 또한 기본 모델의 매개변수 개수는 15,297이다.

---

1) https://cloud.google.com/blog/products/data-analytics/ethereum-bigquery-public-dataset-smart-contract-analytics?hl=en

Performance Metric	Descriptions
Training F1-score	95.0%
Validation F1-score	92.3%
Test F1-score	92.6%
The number of parameters	15,297

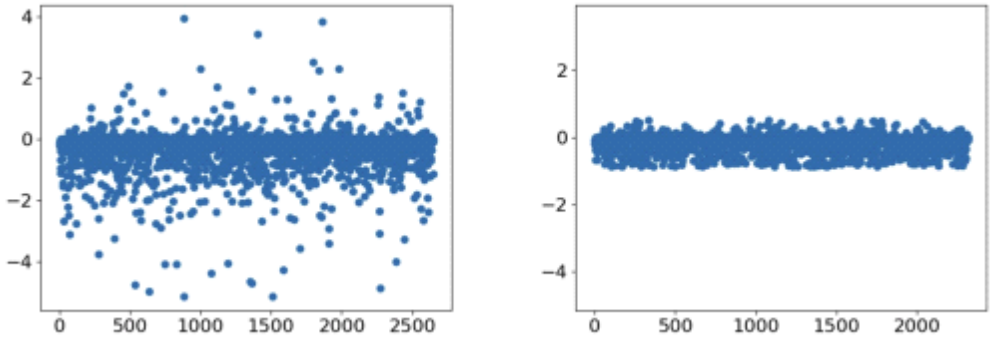
[표 4-1] 기본 모델 성능 평가

제 3 절 경량 모델 구현 및 성능 평가

본 논문에서는 XAI를 사용하여 연산 및 메모리 효율적인 경량 모델을 구현한다. 경량 모델은 중요 opcode를 추출하고 이상치를 제거한 후 opcode의 빈도수를 학습함으로써 구현한다. 4.3절에서는 경량 모델의 구현 과정과 성능에 대해 서술한다.

1) XAI를 통한 중요 opcode 추출

중요 opcode를 선택하기에 앞서, IG와 SHAP에 존재하는 이상치를 제거해야 한다. 왜냐하면 특정 데이터 샘플이 IG, SHAP 값에 크게 영향을 주는 것을 줄임으로써, 중요 opcode 선출에 대한 신뢰성을 향상시키기 때문이다. [그림 4-1]은 앞서 설명한 IQR 방식을 사용한 이상치 제거의 예시를 보여준다. 대부분의 값이 -1과 1사이에 밀집해있고 몇 개의 이상치가 존재하는데, 오른쪽에서 볼 수 있듯이 이상치를 제거하였다.



[그림 4-1] IQR 방식을 사용한 이상치 제거 예시; 이전(왼쪽), 이후(오른쪽)

실험은 기본 모델에서의 IG 값과 SHAP 값을 계산하여 중요 opcode를 선택한다. IG와 SHAP을 모두 고려함으로써 신뢰성을 확보할 수 있다. 모델의 정확도를 유지하며 모델의 크기를 최대한 줄이기 위해 IG와 SHAP 값의 절대값이 큰 50개의 opcode를 선택한다. 이는 140개의 opcode로부터 개수를 줄이면서 실험적으로 얻은 최적의 opcode 개수이다. 이를 통해 IG와 SHAP 모두에서 중요 opcode를 선택할 수 있다.

하지만 중요 opcode는 실험마다 다르게 나타난다. 즉, 한 번의 실험을 통해 얻은 중요 opcode를 사용할 경우 모델의 안정적인 성능을 얻기가 어렵다. 따라서 실험을 10번 반복하여 k번 이상 나타나는 opcode를 사용하여 경량 모델을 구현함으로써 모델의 안정성을 확보한다.

[표 4-2]는 각 케이스 별 실험 결과를 보여준다. Case1, Case2, Case3는 10번의 실험 중, 각각 5, 7, 10번 이상 나타나는 opcode를 사용하여 경량 모델을 구현한 결과이다. 실험 결과에 따르면 Case1의 F1-score가 92.3%로 가장 높은 정확도를 달성하였다. Case2와 Case3의 F1-score는 각각 91.0%, 90.0%로 Case1보다 낮은 성능을 보여주었다. 이는 예측에 영향을 미치는 opcode가 일부 제외되었기 때문인 것으로 보인다. 사용되는 opcode의 수가 감소할수록 신경망의 입력층 뉴런의 개수도 감소한다. 따라서 모델 전체의 매개변수 수도 감소한다. 즉, Case3의 매개변수가 가

장 적다. 하지만 Case3은 기본 모델에 비해 정확도 손실이 크기 때문에 사용하기에 부적합하다. 따라서 본 논문에서는 경량 모델로써 정확도 손실을 최소화하면서 경량화된 Case1을 선택한다.

Performance Metric	case1	case2	case3
The number of opcodes	58	51	37
The number of parameters	5,855	4,861	3,167
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

[표 4-2] 사용된 opcode 개수에 따른 성능 비교
(Case1(k=5), Case2(k=7), Case3(k=10))

[표 4-3]은 경량 모델의 성능을 보여준다. 앞서 설명하였듯이 경량 모델은 기본 모델 대비 매개변수 수가 적다. 따라서 경량 모델은 기본 모델보다 연산 및 메모리 효율적이지만 정확도가 떨어질 수도 있다는 단점이 있다. 하지만 본 논문에서 제안한 경량 모델의 정확도는 92.3%로, 기본 모델의 정확도와 거의 차이가 없으며 연산 및 메모리 효율적이라는 장점이 있다. 이를 통해 실제 시나리오에서 블록체인 노드가 탐욕 컨트랙트를 탐지할 때, 블록체인 네트워크의 연산 및 메모리 오버헤드를 감소시킬 수 있다.

Performance Metric	Descriptions
Training F1-score	92.6%
Validation F1-score	91.6%
Test F1-score	92.3%
The number of parameters	5,855

[표 4-3] 경량 모델 성능 평가

제 4 절 기본 모델과 경량 모델의 성능 비교

[표 4-4]는 기본 모델과 경량 모델의 성능을 보여준다. 경량 모델의 크기는 기본 모델 대비 41.5% 감소시킴으로써 공간 복잡도를 줄일 수 있다. 또한 매개변수(모델의 가중치)는 61.8% 감소시킴으로써 계산 복잡도를 줄일 수 있다. 모델의 크기는 IoT 블록체인에 있어서 매우 중요한 부분이다. 모델의 크기가 작을수록 매개변수가 더 적으며, 매개변수가 적을수록 연산량이 줄어들어 계산 복잡도를 감소시킬 수 있다. 따라서 블록체인에서 탐욕 컨트랙트를 탐지 시, 경량 모델은 연산 및 메모리 효율적으로 탐지를 수행한다.

경량 모델의 F1-score는 기본 모델과 비슷하다. 반면에 모델의 크기는 41.5% 감소시켰기 때문에, 이러한 손실은 무시할 수 있는 값이다. 본 논문에서 제안하는 시나리오에서는 경량 모델이 블록체인의 각 노드에 배포된다. IoT 디바이스와 같은 저사양 디바이스에서는 속도와 메모리의 효율성이 매우 중요하다. 탐지 수행 시, 모델의 연산 및 메모리 사용량이 낮을수록 블록체인의 확장성은 증가한다. 즉, IoT 블록체인에서 동작하는 저

사양 노드가 본 논문에서 제안하는 경량 모델을 사용할 경우 효율적으로 탐욕 컨트랙트를 탐지할 수 있다는 장점이 있다.

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

[표 4-4] 기본 모델과 경량 모델의 성능 비교

제 5 절 명령어 분석

[표 4-5]과 [표 4-6]는 IG와 GS를 통해 모델이 정상 컨트랙트와 탐욕 컨트랙트로 분류함에 있어서, 중요 명령어 상위 5개를 보여준다. 해당 표에서 눈에 띄는 특징은 IG와 GS 모두 분기와 관련된 명령어가 중요한 것을 확인할 수 있다. 분기와 관련된 명령어는 JUMP, JUMPI 그리고 JUMPDEST 명령어가 있다. JUMP는 무조건 분기, JUMPI는 조건 분기 명령어이다. JUMPDEST는 분기할 주소를 나타내는 명령어이다.

정상 컨트랙트와 관련된 분기 명령어는 JUMP와 JUMPDEST이다. 반대로 탐욕 컨트랙트와 관련된 분기 명령어는 JUMPI이다. JUMP와 JUMPDEST는 조건에 관계없이 분기하는 명령어이고 JUMPI는 조건 분기 명령어이다. 탐욕 컨트랙트의 특성상 조건 분기를 통해서 이더를 처리하는 함수에 도달하지 못하도록 한다. 실제로 본 논문에서의 실험을 통해 JUMPI 명령어가 탐욕 컨트랙트 탐지에 있어서 중요 명령어임을 확인하였으며, JUMPI 명령어가 탐욕 컨트랙트로 분류하는데 큰 영향을 미치는 것으로 생각된다. 반대로 정상 컨트랙트에서는 조건 분기 명령어인 JUMPI가 중요 명령어가 아닌 것으로 나

타난다.

Algorithm	Sorted by Values of IG and SHAP				
	1	2	3	4	5
Integrated Gradient	JUMPDEST	DUP1	SUB	JUMP	EQ
Gradient SHAP	DUP1	SUB	JUMP	JUMPDEST	PUSH4

[표 4-5] 정상 컨트랙트에서의 상위 중요 5개 명령어

Algorithm	Sorted by Values of IG and SHAP				
	1	2	3	4	5
Integrated Gradient	JUMPI	CALL VALUE	SWAP1	SWAP2	STOP
Gradient SHAP	JUMPI	PUSH2	CALL VALUE	SWAP2	POP

[표 4-6] 탐욕 컨트랙트에서의 상위 중요 5개 명령어

제 6 절 기존 기법과의 비교

MAIAN은 탐욕, prodigal 그리고 suicidal 컨트랙트를 탐지한다. MAIAN은 파일 크기가 매우 작지만, 탐지에 오랜 시간이 걸린다는 단점이 존재한다.

이전 연구는 가장 최근의 딥러닝 기반 탐지 기법으로, 여러 유형의 악성 스마트 컨트랙트를 하나의 클래스로 분류한다. 따라서 다양한 악성 스마트 컨트랙트를 탐지할 수 있다. 하지만 이미 배포된 스마트 컨트랙트에 대해서는

탐지를 수행할 수 없다는 단점이 존재한다. 즉, 이전 연구에서 제안한 기법은 새로 배포된 스마트 컨트랙트에 대해서만 탐지가 가능하다. 따라서 이미 수많은 악성 스마트 컨트랙트가 배포되어 있는 블록체인 네트워크에서는 적합하지 않다. 또한 781개의 스마트 컨트랙트(정상 컨트랙트 650개, 악성 스마트 컨트랙트 131개)만 데이터로써 사용한다. 따라서 이전 연구에서 제안한 모델의 안정성이 높지 않을 것으로 생각된다. [표 4-7]에 언급된 요소들은 이전 모델을 그대로 구현한 결과이다. 많은 모델들이 이전 연구에서 제안되었지만, 가장 높은 성능을 보여주는 모델을 재현한다. 그후, 해당 값은 재현된 모델에 의해 측정된다.

Category	MAIAN	Previous Work	This Work
Smart contracts	Prodigal, Suicidal, Greedy (3 classes)	Malicious (2 classes)	Greedy (2 classes)
Algorithm	Symbolic analysis, Concrete validation	Deep Learning	Deep Learning
The number of parameters	-	54,018	5,855
File Size	0.232 MB	0.74 MB	0.53 MB
Speed	Within 10 seconds	Slower than ours	0.013 ms

[표 4-7] 스마트 컨트랙트 탐지 기법 비교

우리의 기법은 정상 컨트랙트와 탐욕 컨트랙트를 분류할 수 있다. 하지만 80,000개 정도의 데이터셋을 수집하여 MAIAN 도구를 통해 탐지를 수행한 결과, 악성 컨트랙트 중 탐욕 컨트랙트의 비율은 93.4%이다. 따라서 블록체인 내에서 악성 스마트 컨트랙트로 인해 발생하는 대부분의 문제들은 탐욕 컨트랙트를 탐지하는 것만으로도 예방할 수 있다.

본 논문에서는 이전 연구보다 38배 더 큰 데이터셋을 사용하여, 모델의 신뢰성과 안정성을 높였다. 또한 본 논문에서 제안한 탐지 모델은 탐욕 컨트랙트만 탐지하지만 대부분의 악성 스마트 컨트랙트를 탐지할 수 있는 잠재력을 가지고 있다. 우리의 경량 모델은 MAIAN보다 크지만 훨씬 더 빠른 속도록 스마트 컨트랙트를 탐지할 수 있다. 또한 이전 연구와 비교하였을 때 파일 크기는 28.4%, 매개변수 개수는 89.2% 감소하였다. 따라서 우리의 모델은 연산 및 메모리 효율적이다. 이러한 점은 저사양 디바이스인 IoT 블록체인에서 더욱 큰 장점으로 작용할 것으로 생각된다.

제 5 장 결론 및 향후 연구 방안

본 연구에서는, 경량 모델을 구현하기 위해 모델에 큰 영향을 미치는 opcode를 식별하고 분석한다. 또한 이를 기반으로 연산 및 메모리 효율성이 뛰어난 IoT용 경량 탐지 모델을 제안한다. 스마트 컨트랙트 배포 단계에서 탐지를 수행하는 이전 접근 방식과 달리, 제안 시스템은 스마트 컨트랙트 실행 단계에서 탐지를 수행한다. 따라서 제안하는 시스템은 이미 배포된 스마트 컨트랙트에 대해서도 탐지를 수행할 수 있다. 결과적으로 제안하는 시스템은 블록체인 네트워크의 안정성과 확장성을 향상시킨다.

실험 결과를 요약하면, 경량 모델의 크기는 기본 모델에 비해 41.5% 감소하였다. 따라서 제안하는 시스템이 블록체인에서 사용될 때 연산 및 메모리 효율적으로 탐지를 수행한다. 마지막으로 경량 모델은 가벼움에도 불구하고 92.3%의 높은 탐지 정확도를 달성하였다.

탐욕 컨트랙트 뿐만 아니라, 다양한 악성 스마트 컨트랙트에 대해서도 탐지를 수행하는 모델을 구현하려고 하였으나, 탐욕 컨트랙트 외에 다른 악성 스마트 컨트랙트 데이터셋을 수집하는 데에 어려움을 겪었다. 따라서 탐욕 컨트랙트만을 탐지하는 모델을 구현하였다.

향후 연구에서는, 탐욕 컨트랙트 뿐만 아니라 다양한 악성 스마트 컨트랙트를 탐지할 수 있는 모델을 구현할 계획이다. 또한 경량 모델은 저사양 디바이스를 타겟으로 구현하였다. 따라서 XAI뿐만 아니라 Knowledge Distillation, Quantization 그리고 Pruning methods를 활용하여 모델을 더욱 경량화시킬 예정이다. 그 후, 라즈베리파이와 같이 저사양 디바이스에 경량 모델을 배포하여 추가적인 실험을 수행할 예정이다.

참 고 문 헌

1. 국외문헌

Nikolić, Ivica, et al. "Finding the greedy, prodigal, and suicidal contracts at scale."Proceedings of the 34th annual computer security applications conference. 2018.재인용.

Shah, Harshit, et al. "Deep learning-based malicious smart contract and intrusion detection system for IoT environment." Mathematics 11.2 (2023): 418.재인용.

Tikhomirov, Sergei, et al. "Smartcheck: Static analysis of ethereum smart contracts." Proceedings of the 1st international workshop on emerging trends in software engineering for blockchain. 2018.재인용.

Xu, Yingjie, et al. "A novel machine learning-based analysis model for smart contract vulnerability." Security and Communication Networks 2021 (2021): 1-12.재인용.

Srinivasan, Pooja. "TP-Detect: trigram-pixel based vulnerability detection for Ethereum smart contracts." Multimedia Tools and Applications (2023): 1-15.재인용.

Tyagi, Amit Kumar, et al. "Blockchain—Internet of Things Applications: Opportunities and Challenges for Industry 4.0 and Society 5.0." Sensors 23.2 (2023): 947.

Taloba, Ahmed I., et al. "A blockchain-based hybrid platform for multimedia data processing in IoT-Healthcare." Alexandria Engineering Journal 65 (2023): 263-274.

Sharma, Pratima, et al. "EHDHE: Enhancing security of healthcare documents in IoT-enabled digital healthcare ecosystems using

blockchain." Information Sciences 629 (2023): 703-718.

LeCun, Yann, Yoshua Bengio, and Geoffrey Hinton. "Deep learning." nature 521.7553 (2015): 436-444.

Schmidhuber, Jürgen. "Deep learning in neural networks: An overview." Neural networks 61 (2015): 85-117.

LeCun, Yann, et al. "Gradient-based learning applied to document recognition." Proceedings of the IEEE 86.11 (1998): 2278-2324.

Schuster, Mike, and Kuldip K. Paliwal. "Bidirectional recurrent neural networks." IEEE transactions on Signal Processing 45.11 (1997): 2673-2681.

Hochreiter, Sepp, and Jürgen Schmidhuber. "Long short-term memory." Neural computation 9.8 (1997): 1735-1780.

Vaswani, Ashish, et al. "Attention is all you need." Advances in neural information processing systems 30 (2017).

Goodfellow, Ian, et al. "Generative adversarial networks." Communications of the ACM 63.11 (2020): 139-144.

Wang, Ching-Hao, et al. "Lightweight deep learning: An overview." IEEE Consumer Electronics Magazine (2022).

Arrieta, Alejandro Barredo, et al. "Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI." Information fusion 58 (2020): 82-115.

Ali, Sajid, et al. "Explainable Artificial Intelligence (XAI): What we know and what is left to attain Trustworthy Artificial Intelligence." Information Fusion 99 (2023): 101805.

Kokhlikyan, Narine, et al. "Captum: A unified and generic model interpretability library for pytorch." arXiv preprint arXiv:2009.07896 (2020).

Sundararajan, Mukund, Ankur Taly, and Qiqi Yan. "Axiomatic attribution for deep networks." International conference on machine learning.

PMLR, 2017.

Erion, Gabriel, et al. "Learning explainable models using attribution priors." (2019).

Nakamoto, Satoshi. "Bitcoin: A peer-to-peer electronic cash system." Decentralized business review (2008).

Buterin, Vitalik. "A next-generation smart contract and decentralized application platform." white paper 3.37 (2014): 2-1.

Whaley III, Dewey Lonzo. The interquartile range: Theory and estimation. Diss. East Tennessee State University, 2005.

ABSTRACT

Greedy smart contract detection through lightweight deep learning implemented using XAI

Kang, Yea-Jun

Major in IT Convergence Engineering

Dept. of IT Convergence Engineering

The Graduate School

Hansung University

A smart contract is a digital contract on a blockchain. Through smart contracts, transactions between parties are possible without a third party on the blockchain network. However, there are malicious contracts, such as greedy contracts, which can cause enormous damage to users and blockchain networks. Therefore, countermeasures against this problem are required. In this work, we propose a greedy contract detection system based on deep learning. The detection model is trained through the frequency of opcodes in the smart contract. Additionally, we implement a lightweight model for deployment on the IoT. We identify important instructions for detection through explainable artificial intelligence (XAI). After that, we train the lightweight model through only important instructions. Therefore, the lightweight model is a computationally and memory-efficient detection model for the IoT. Through our approach,

we achieve a high detection accuracy of 92.3%. In addition, the file size of the lightweight model is reduced by 41.5% compared to the base model and there is little loss of accuracy.

【KEY WORDS】blockchain, smart contract, greedy contract detection, deep learning, explainable artificial intelligence, lightweight, Internet of Things