ISSN(Print) : 1976-5304, ISSN(Online) : 2713-931X
https://kdfs.jams.or.kr, DOI : XXXXXXXXXXXXXXXX

# 다중 CAPTCHA 문제의 자동화 해결 방안 연구

# A Methodology for Automated Resolution of Multi-type CAPTCHA Challenges

**요 약**

인터넷의 급속한 발달과 함께 사이버 공간에서의 불법 행위와 범죄가 심각한 사회 문제로 대두되고 있다. 이에 따라 자동화된 증거 수집 및 추적 기술의 필요성이 커지고 있으나, CAPTCHA(Completely Automated Public Turing test to tell Computers and Humans Apart)의 존재는 이러한 접근을 차단하여 수사 효율을 저해하는 주요 요인으로 작용한다. 본 연구는 머신러닝 및 이미지 처리 알고리즘을 결합하여, CAPTCHA의 탐지부터 해결까지 인간의 개입을 최소화한 자동화 프레임워크를 제안한다. 본 프레임워크는 HTML 구조와 네트워크 기록을 분석하여 CAPTCHA의 존재를 판별하는 탐지 단계와, 이미지 · 슬라이드 · 아이콘 유형에 따라 대응하는 알고리즘을 개발해 CAPTCHA를 자동으로 우회하는 해결 단계로 구성된다. YOLO, OpenCV, MiniCPM-V-4_t-int4 등의 알고리즘을 활용해 각 유형별 실험을 수행한 결과, GPU 환경에서 각각 100%, 90%, 45%의 정확도를 달성하였다. 본 연구는 인터넷 기반 수사 환경에서 CAPTCHA를 자동으로 우회할 수 있음을 실증하였으며, 향후 새로운 CAPTCHA 유형에도 확장 가능함을 보였다. 이는 효율적인 수사 활동을 지원함으로써 사이버 범죄 분석 및 사후 사고 예방에 기여할 수 있을 것으로 기대된다.

**주제어:** 자동 수사 시스템, 크롤링, CAPTCHA, CAPTCHA 탐지 및 식별, YOLO, LLM

**ABSTRACT**

With the rapid development of the Internet, illegal activities and crimes in cyberspace have emerged as serious social issues. Consequently, the need for automated evidence collection and tracking technologies has grown significantly. However, the presence of CAPTCHA(Completely Automated Public Turing test to tell Computers and Humans Apart) serves as a major obstacle, hindering such approaches and reducing the efficiency of investigations. This study proposes an automated framework that combines machine learning and image processing algorithms to minimize human intervention in CAPTCHA detection and solving. The proposed framework consists of a detection phase, which identifies the presence of CAPTCHA by analyzing HTML structures and network logs, and a solving phase, which automatically bypasses CAPTCHA challenges using specialized algorithms tailored to image, slide, and icon-based types. Experiments using algorithms such as YOLO, OpenCV, and MiniCPM-V-4_t-int4 demonstrated detection and solving accuracies of 100%, 90%, and 45%, respectively, under GPU environments. The results validate the framework's capability to automatically bypass CAPTCHA in Internet-based investigation scenarios and show its potential to adapt to emerging CAPTCHA types. This study is expected to contribute to efficient cybercrime investigation, digital forensics, and post-incident prevention.

**Key Words:** Automatic Investigation System, Crawling, CAPTCHA, CAPTCHA Detection and Identification, YOLO, LLM

# I. 서 론

컴퓨터의 보편화로 많은 개인이 전 세계의 다양한 정보에 접근할 수 있게 되었다. 허나 통제되지 않은 자유로운 환경 속에서의 불법 거래, 계정 탈취, 불법 자료 공유 등과 같은 범죄 행위의 규모 또한 인터넷의 발전과 함께 증가하고 있다. 따라서 이를 실시간으로 감시하고 추적하는 행위의 필요성이 두각되고 있고 이를 자동화하고자 하는 웹 크롤링 기반의 수사 기법이 연구되고 있다. 그러나 이러한 자동화된 접근은 CAPTCHA(Completely Automated Public Turing test to tell Computers and Humans Apart)와 같은 보안 장치에 의해 차단되곤 한다. CAPTCHA는 네트워크를 통해 특정 공간에 접근하고자 하는 대상이 인간인지 여부를 판단하고, 비정상적이거나 자동화된 접근을 차단하는 역할을 수행한다.

초기의 CAPTCHA는 왜곡된 문자나 숫자를 식별하도록 설계된 텍스트 기반 형태였으나, 최근 머신러닝, 특히 딥러닝 분야의 급격한 발전으로 인해 기존의 전통적 CAPTCHA의 목적을 유지하는데 있어 장애물이 되고 있다. 자연어 처리, 컴퓨터 비전과 같은 딥러닝을 기반으로 한 분야에서의 성능 향상은 주어진 과제를 성공적으로 해결할 수 있도록 만들었다. 예를 들어, STT(Speech-To-Text)와 OCR(Optical Character Recognition) 기술의 도입은 노이즈가 추가된 텍스트 및 오디오를 정확히 인식하도록 하였다. 이에 대응하기 위해 CAPTCHA 시스템은 이미지 선택형, 퍼즐형, 행동 기반 등 사용자 행위와 지각 능력을 종합적으로 요구하는 방향으로 발전하고 있다. CAPTCHA 발전과 더불어 하드웨어의 발전으로 빠른 속도로 고도의 지능을 모방할 수 있는 방법들이 연구되고 있으며 최근에는 대규모 언어 모델(LLM, Large Language Model) 및 멀티모달 모델의 등장으로 텍스트, 이미지, 음성 등 서로 다른 입력 형식을 복합적으로 이해하고 추론하는 것이 가능해졌다. 이는 CAPTCHA 시스템을 무력화하거나 우회하려는 시도들로 이어지고 있다. CNN의 등장 이후 이미지 분류 및 좌표 검출이 정교해지면서, 특정 라벨과 일치하는 이미지를 선택하도록 요구하는 유형의 CAPTCHA는 더 이상 효과적인 방어 수단으로 기능하기 어려워졌다. 대형 데이터로 학습된 멀티모달 LLM은 각 CAPTCHA에 대한 개별 학습 없이 접근할 수 있는 길을 열었으며, 이를 통해 논리적으로 문제를 이해하고 체계적으로 문제를 해결할 수 있게 되었다.

앞서 언급한 것과 같이 사이버 범죄의 확대로 디지털 포렌식과 과학 수사의 영역에서 인터넷 기반 증거 수집의 중요성이 부각되고 있다. 허나 CAPTCHA의 존재는 수사관의 모니터링 과정의 실시간성을 저해시켜 범죄와 관련된 증거를 적시에 확보하는데 장애가 되고, 자동화된 접근을 차단함으로써 광범위한 증거 수집을 방해한다. 따라서 자동화된 CAPTCHA 우회 기술은 과학 수사의 측면에서 수사 속도를 향상시키고, 확보한 증거들을 다방면에서 종합적으로 처리할 수 있게 함으로써 범죄 예방과 해결을 위한 수단으로서 필수적이라고 할 수 있다.

이러한 흐름 속에서 본 연구는 각 CAPTCHA 서비스 및 유형들의 구조적 특징을 분석하고, 여러 머신러닝 및 데이터 처리 알고리즘을 적절히 활용해 이를 해결하는 것에 초점을 맞춘 오픈소스 프로그램을 개발하고자 한다. reCAPTCHA, GeeTest와 같이 공개된 CAPTCHA 제공자들을 집중적으로 탐구하여 CAPTCHA가 발생했을 시 이를 탐지하고 해결하기까지의 전체적인 흐름을 제공하여 과학수사의 관점에서 효율적인 정보 수집 및 사고 예방에 기여하고, 나아가 CAPTCHA 취약성에 대한 이해를 바탕으로 향후 새로운 유형의 CAPTCHA 대응에 실질적 가이드를 제공하고자 한다.

# II. 관련 연구

## 2.1. CAPTCHA 개요

CAPTCHA는 인터넷 사용자의 자동화된 악성 봇 접근을 차단하기 위해 개발된 인증 기술로, 그 종류와 설계 방식은 지난 20년간 꾸준히 진화해왔다. 근본적인 CAPTCHA의 설계 목표는 인간에게는 90% 이상의 성공률을, 컴퓨터에는 1% 이하의 성공률을 나타내도록 만드는 것이며, 이를 달성하기 위해 텍스트, 이미지, 논리 · 추론, 오디오 등 다양한 유형의 CAPTCHA가 도입되었다 [1].

ISSN(Print) : 1976-5304, ISSN(Online) : 2713-931X
https://kdfs.jams.or.kr, DOI : XXXXXXXXXXXXXXXX


Figure 1. 왜곡된 텍스트를 해석해 입력하는 텍스트 유형 CAPTCHA

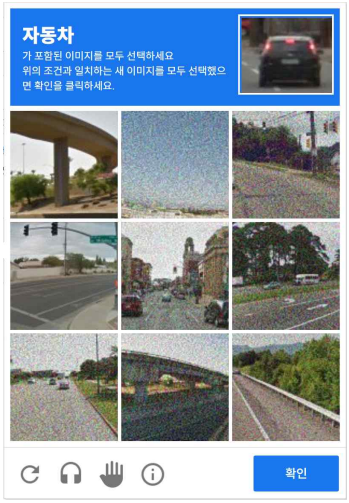


Figure 2. 이미지 내 특정 객체를 탐색하는 이미지 유형 CAPTCHA

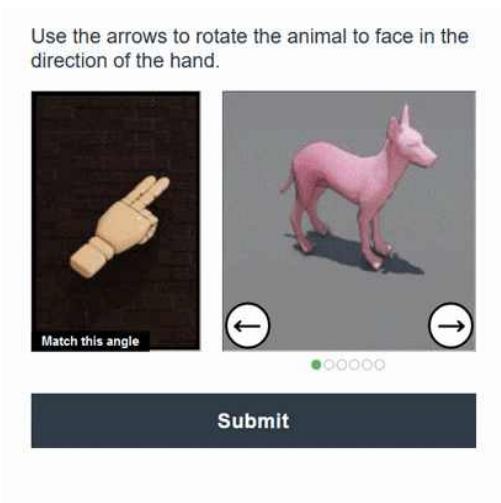


Figure 3. 각도, 방향 등을 복합적으로 고려하는 논리・추론 유형 CAPTCHA

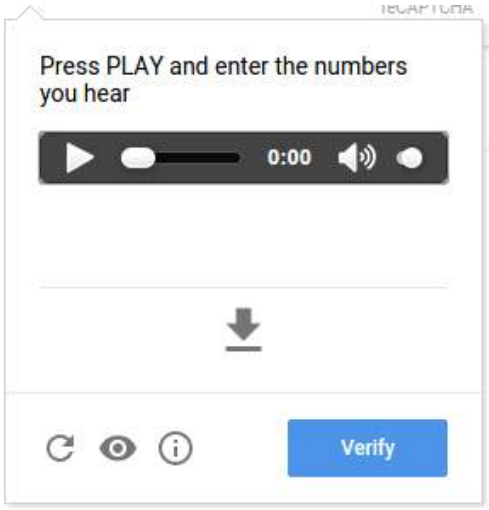


Figure 4. 주어진 오디오의 내용을 듣고 입력하는 오디오 유형 CAPTCHA

<Figure 1>의 텍스트 유형 CAPTCHA는 가장 오래되고 널리 쓰이는 형태로, 알파벳과 아라비아 숫자 등의 왜곡된 문자 시퀀스를 인간이 인식하고 입력하여 해결 가능하다. 이러한 CAPTCHA를 공격으로부터 보호하기 위해 노이즈, 겹침 등과 같은 분할 저항 및 회전・왜곡, 다중 폰트와 같은 인식 저항 기술을 활용한다. 하지만 이와 같은 유형은 머신러닝 기반 공격에 취약하며, 최근에는 텍스트 유형 CAPTCHA의 복잡성을 향상시킴에도 불구하고, 여전히 취약하다 [2].

이미지 유형 CAPTCHA는 <Figure 2>와 같이 사용자가 주어진 단일 혹은 복합 이미지 속 특정 객체를 찾아 선택하여 해결 가능하다. 이미지 인식 및 분류 기술이 발전하기 전까지는 효과적인 보안 시스템으로서 작용하였으나, SVM, CNN, YOLO 등과 같은 다양한 이미지 분류 및 검출 알고리즘의 발전으로 견고성을 잃고 있다 [3].

논리・추론 유형 CAPTCHA는 이미지 유형 CAPTCHA로부터 파생된 형태로, 단순한 객체 검출을 넘어 지시문을 정확히 이해하고 <Figure 3>처럼 방향과 위치, 각 물체별 세부 요소 등을 고려해 논리적 판단과 추론을 수행해야 하는 인증 방식이다. 사용자의 인지적 능력을 활용하고 체계적으로 논리적인 해결 과정을 요구해 단순한 이미지 분류 알고리즘으로는 쉽게 우회할 수 없도록 설계되었다 [4].

시각 장애인을 위해 개발된 <Figure 4>의 오디오 유형 CAPTCHA, 동적 시나리오 구현을 위한 비디오

유형 CAPTCHA도 일부 도입 중이나, 전문 음성 및 영상 인식 엔진의 도입으로 실제 보안 효과는 제한적이라는 평가가 많다. 오디오 · 비디오 기반 CAPTCHA는 높은 대역폭과 사용자의 시간 부담, 쉬운 우회 성공률 등 실전 배치에 한계가 존재한다 [2].

### 2.2. 선행 연구

초기의 CAPTCHA는 왜곡된 문자 인식에 기반한 텍스트 유형 CAPTCHA가 주류였다. 그러나 최근 딥러닝 기술의 발전으로 CNN, RNN, GAN 등과 같은 다양한 신경망 모델이 텍스트 유형 CAPTCHA를 높은 정확도로 해독하고 있다 [1][2]. 이미지 유형 CAPTCHA 역시 머신러닝의 주요 공격 대상으로, 객체 검출 및 분류 기술의 발전으로 YOLO, Faster R-CNN, EfficientDet 등의 최신 딥러닝 모델들은 reCAPTCHA v2와 같은 유형의 CAPTCHA를 100%에 가까운 정확도로 해독하는데 성공하였다 [3]. 또한 쿠키, 브라우저 히스토리, 마우스 움직임 등 사용자 행동 데이터를 활용해 컴퓨터와 인간을 구분하는 기존 보안 체계 또한 정교한 봇의 행동 모방으로 인해 한계에 직면하고 있음을 밝혔다 [3]. 입체 렌더링과 측면 정보 등 공간적 특성을 활용한 3D CAPTCHA를 도입해 보안성을 강화하였으나, 계층적 전처리와 문자 분할, 각도 보정 등의 이미지 세분화 알고리즘을 사용한 자동화 공격에 의하여 문자 인식률 92%, 전체 CAPTCHA 해독률 약 76%를 기록하는 취약성이 드러났다. 이는 3D 기반이라고 해도 설계상의 취약점이 있으면 자동 해독을 피하기 어려움을 의미한다 [4].

또, 최근에는 LLM 및 멀티모달 모델을 활용하여 CAPTCHA 시스템을 무력화하거나 우회하는 방안에 대한 연구가 수행되고 있다. Deng et al. (2024)은 대규모 언어 모델을 활용한 추론형 CAPTCHA 자동화 해독 프레임워크인 OEDIPUS를 제안하였다. 이 시스템은 CAPTCHA 문제를 LLM이 처리하기 쉬운 단위 작업으로 분해하고, Chain-of-Thought 방식으로 단계별 해결을 시도한다. 이를 통해 복잡한 논리 및 추론 기반 CAPTCHA도 상당 부분 자동화하여 일부 유형에 대해서 해결할 수 있음을 보였으나, LLM 자체의 이해 능력 부족, 시간적 부담 등과 같은 한계를 지닌다 [5].

Teoh et al. (2025)는 범용 VLM 기반 CAPTCHA 해결 알고리즘인 Halligan을 제안하며, 기존의 특정 유형에 특화된 딥러닝 모델과 달리, 다양한 유형의 시각적 CAPTCHA를 사전 학습이나 추가 적응 없이 자동으로 해독하고자 시도했다. Halligan은 시각적 챌린지를 최적화 문제로 변화하여, 지시문을 목표 함수로, 이미지와 UI 요소를 탐색 공간으로 추상화한다. 실험 결과, 26종류 2,600개의 문제에서 평균 60.7%의 성공률을 기록했다. Halligan은 클릭, 드래그, 슬라이드 등 다양한 상호작용을 탐색하며, 기존의 특화형 딥러닝 모델보다 높은 범용성을 보였다 [6].

이와 같이 머신러닝 기술을 활용한 CAPTCHA 우회 시도는 다양하게 이루어지고 있으나, CAPTCHA의 발생을 감지하고 해결하여 전 과정을 자동화하는 통합적 연구는 여전히 미흡한 상황이다. 기존 연구들은 주로 특정 CAPTCHA 유형에 국한된 기법을 제안하거나, 범용적 모델을 제안하더라도 구체적인 적용 절차를 명확히 제시하지 못하고 있다. 이에 본 연구에서는 CAPTCHA의 탐지부터 유형별 해결 방법의 자동 적용에 이르는 전체 과정을 체계적으로 설명하고, 최종적으로 CAPTCHA를 성공적으로 우회함으로써 단순한 방법론 소개에 그치지 않고 포괄적인 자동화 프레임워크를 구현 및 배포한다.

## III. 방법론

### 3.1. 개요

본 알고리즘은 <Figure 5>에서 표현된 것과 같이 자동화 프로그램인 봇이 사용자가 정의한 순서에 따라 동작하던 중, 다양한 원인으로 인해 작업이 중단될 경우 실행된다. 중단 원인이 CAPTCHA로 인한 것인지 판단하기 위해 탐지 단계를 수행하며, CAPTCHA가 외부 제공자에 의해 삽입되고 네트워크를 통해 검증 결과를 송수신하는 특성을 고려하여 HTML 구조와 네트워크 기록을 분석한다. 본 연구에서 다루는 두 가지 서비스인 reCAPTCHA v2와 GeeTest, 그리고 그 세부 유형인 이미지, 슬라이드, 아이콘 유형 CAPTCHA가 탐지되면, 알고리즘은 각 서비스 및 유형에 적합한 해결 단계로 진입한다.

해결 단계에서는 앞서 구분한 CAPTCHA 서비스 및 유형에 따라 각기 다른 방법이 적용된다. 본 단계

ISSN(Print) : 1976-5304, ISSN(Online) : 2713-931X
https://kdfs.jams.or.kr, DOI : XXXXXXXXXXXXXXXX

에서는 CAPTCHA 해결을 위한 지시문 정의뿐만 아니라 이를 자동으로 수행하고, 최종적으로 성공 여부를 판단함으로써 사용자에게 작업 재개 가능 여부를 전달한다. 또한 각 행동 사이의 대기 시간을 무작위로 설정하고 의도적인 멈춤 등을 추가하여 인간과 유사한 움직임을 구현해 CAPTCHA 내부의 추가적인 장치 또한 우회할 수 있다.

탐지와 해결 두 단계로 나누어 CAPTCHA를 컴퓨터 자원만으로 해결함으로써 인간에게 의존했던 정보 수집 및 감시 작업을 자동화할 수 있게 되며, 이는 보다 효율적이고 체계적인 수사 활동을 가능케 한다.

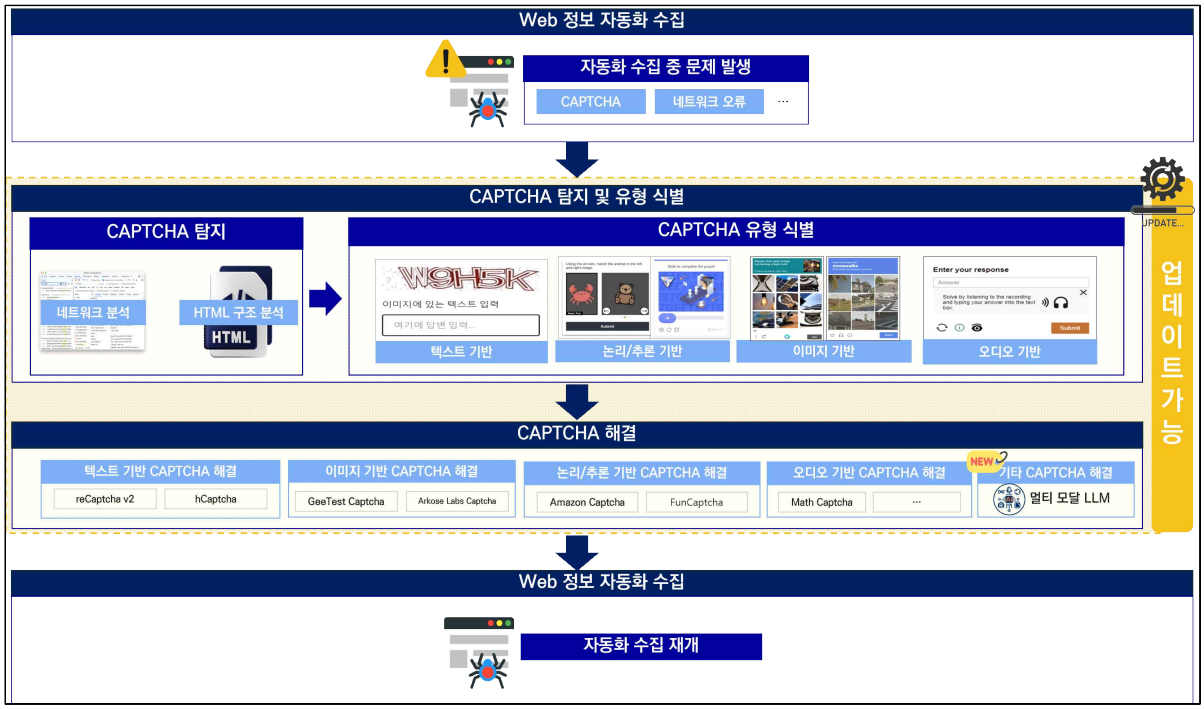


Figure 5. 다중 CAPTCHA 자동화 해결 프레임워크 개요

### 3.2. CAPTCHA 탐지 및 식별

본 연구에서는 개요에서 언급한 바와 같이 reCAPTCHA v2와 GeeTest 서비스를 대상으로 탐구를 진행한다. reCAPTCHA v2가 이미지 선택형 방식만을 제공하는데 비해, GeeTest는 슬라이드형, 아이콘 선택형 등 다양한 퍼즐 기반 유형을 제시하므로, 이에 따라 세부 유형의 구분이 필요하다. API를 이용해 CAPTCHA 서비스 제공자로부터 그 구조를 받아와 추가하고, 토큰을 활용해 검증 여부를 송수신하는 CAPTCHA의 특성을 활용하여, HTML 구조를 기반으로 서비스 제공자를 식별하고 네트워크 URL을 통해 그 유형을 추정한다.

웹 개발자는 자신의 서비스에 특정 CAPTCHA를 적용하기 위해 서비스 제공자의 스크립트를 웹사이트 내에 포함하고, 브라우저가 이를 불러오면 해당 스크립트는 페이지 내에 CAPTCHA의 구조를 삽입하여 화면에 표시한다. reCAPTCHA v2의 경우 title 속성이 ‘reCAPTCHA’ 인 iframe을 추가하며, GeeTest의 경우 class 속성에 ‘geetest’를 포함한 여러 div 태그들을 생성한다. 이로써 CAPTCHA로 인해 작업이 중단되었음을 확인할 수 있으며 이후 그 유형을 구분하기 위해 네트워크 기록을 활용한다.

<Table 1> GeeTest 유형별 호출 URL

| 유형 | 호출 URL |
|---|---|
| Slide | https://gcaptcha4.geetest.com/load?captcha_id=#&risk_type=slide |
| IconCrush | https://gcaptcha4.geetest.com/load?captcha_id=#&risk_type=match |
| Icon | https://gcaptcha4.geetest.com/load?captcha_id=#&risk_type=icon |
| Image | https://gcaptcha4.geetest.com/load?captcha_id=#&risk_type=nine |

<Table 1>은 GeeTest CAPTCHA가 웹페이지에 로드될 때 브라우저가 GeeTest 서버로 전송하는 초기화 호출을 나타낸다. risk_type 파라미터가 각 유형별로 상이한 것을 확인할 수 있으며 해당 값을 읽어옴으로써 GeeTest 내 CAPTCHA 유형을 구분할 수 있게 된다. 이러한 네트워크 기반 분류는 DOM 내부의 깊은 HTML 구조를 파싱해 특정 클래스나 자식 요소를 탐색하는 방법에 비해 연산량이 적고 탐지 지연이 적다는 이점을 가진다. 이와 같은 과정을 통해 CAPTCHA의 존재를 탐지하고, 그 유형을 구분하여 해결 단계로 전환한다.

### 3.3. 해결 과정

앞선 탐지 과정이 웹의 구성 요소 분석을 통해 CAPTCHA의 특성을 식별하는데 초점을 맞춘다면, 본 해결 과정은 각 CAPTCHA 서비스와 유형에 따라 서로 다른 접근 방식을 필요로 한다. 본 장에서는 이미지, 슬라이드, 아이콘 유형 CAPTCHA를 대상으로 각 유형별 해결 알고리즘과, 해결 여부를 판단하여 반환하는 전반적인 절차를 다룬다.

#### 3.3.1. 이미지 유형 CAPTCHA 해결

이미지 유형은 지시문에 제시된 라벨을 기반으로 사용자가 해당 대상을 이미지 내에서 탐색 및 선택하도록 설계되어 있으며, 이는 9개의 서로 다른 이미지 타일 중에서 목표 대상을 찾는 형태와 16개의 타일로 구성된 단일 이미지 내에서 특정 객체를 식별하는 형태로 구분된다. 이러한 유형은 reCAPTCHA v2와 hCAPTCHA 등 주요 CAPTCHA 서비스에서 제공되며, 그중 대표적인 reCAPTCHA v2를 대상으로 개발 및 실험을 진행한다. 두 형태로 나뉘어 접근하지만, 특정 객체를 검출한다는 목표는 동일하기에 이에 활용하는 방법론은 동일하다. 이미지 기반 딥러닝 기술의 발전과 다양화에 따라 이미지 분류, 세그멘테이션 등 다양한 기술이 제안되었으며 정확도와 속도를 모두 고려해야 하는 과학 수사의 특성상 본 연구에서는 객체 검출을 지원하는 사전 학습된 YOLO(You Look Only Once) 모델을 활용한다.

YOLO 알고리즘은 이미지를 전체적으로 한 번만 분석하여 객체의 위치와 종류를 동시에 예측할 수 있다. 이는 전체 이미지를 nxn 크기의 격자들로 분할하고, 각 격자 내에서 객체 존재 확률과 바운딩 박스 좌표를 직접 예측함으로써 별도의 후보 영역 추출 과정을 제거하였기 때문이다. YOLO 모델은 사전에 학습된 라벨들에 대해서만 탐지가 가능하다는 한계가 존재하나, reCAPTCHA v2에서 요구하는 객체들의 종류가 한정적이며, 필요에 따라 데이터를 추가적으로 학습시켜 내부 파라미터를 추가 및 개선할 수 있어 이후 새롭게 등장하는 유형에 대해서도 대응이 가능하다.

지시문으로부터 어떤 객체를 탐지해야 하는지를 추출하고, 이를 제시된 이미지와 함께 학습된 모델에 입력해 <Figure 6>과 같이 좌표를 반환받는다. 해당 좌표 정보를 이용해 이미지 격자 구조 내에서 대상 격자를 선정하고 클릭한다. 이때 격자를 선택하는 순서를 무작위로 재배열하고, 선택하는 동작 사이 무작위 지연을 삽입하여 인간의 움직임에서 관찰할 수 있는 불규칙성을 모방한다. 이는 reCAPTCHA v2 내부적으로 사용자의 행동을 함께 파악하고 있음을 고려해 CAPTCHA를 안전하게 우회하기 위한 조치이다. 또한 선택한 이미지 위의 타일이 교체되는 상황을 감지하는 함수를 추가하여 변화가 발생하면 재검출 및 재선택 절차를 수행한다.

ISSN(Print) : 1976-5304, ISSN(Online) : 2713-931X
https://kdfs.jams.or.kr, DOI : XXXXXXXXXXXXXXXX

Figure 6. YOLO 모델을 활용한
CAPTCHA 내 객체 검출

reCAPTCHA v2가 성공적으로 해결되어 인증 토큰이 반환되면, 웹페이지는 원래 화면으로 복귀하거나 <Figure 7>의 시각 표시를 제공한다. 따라서 이러한 표시 또는 iframe 존재 여부를 확인함으로써 CAPTCHA의 해결 여부를 판단할 수 있다. 실패의 경우 초기 단계로 돌아가 동일한 절차를 반복하고, 그렇지 않다면 사용자에게 기존 작업을 계속 진행할 수 있음을 안내하고 절차를 종료한다.


Figure 7. reCAPTCHA v2 해결 이미지

### 3.3.2. 논리・추론 유형 CAPTCHA 해결

#### 3.3.2.1. 슬라이드 유형 CAPTCHA

슬라이드 유형 CAPTCHA는 전체 이미지 내에서 퍼즐 조각이 맞아야 하는 위치를 찾은 뒤, 사용자가 하단의 슬라이더나 퍼즐 조각을 마우스로 끌어 해당 위치에 맞추는 방식으로 설계되어 있다. 시중에 공개된 서비스 중 접근성이 용이한 GeeTest를 대상으로 실험을 진행한다. 이미지 내 비어있는 공간과 퍼즐 조각의 모양이 동일한 점에 착안하여 OpenCV 라이브러리를 활용한 이미지 분석 작업을 통해 문제를 해결한다.

OpenCV는 이미지와 영상 등 컴퓨터 비전 연구를 위해 사용되는 라이브러리로, 이미지 변환, 필터링과 같은 다양한 기능을 제공한다. 특히 본 연구에서는 윤곽선 검출과 템플릿 매칭 기술을 적극적으로 활용한다. 윤곽선 검출은 이미지 내에서 밝기 변화가 큰 경계선을 찾아내는 기법으로, 객체의 윤곽이나 형태를 식별하는데 사용되며 연구에서는 대표적인 알고리즘인 Canny를 사용해 배경 이미지와 퍼즐의 윤곽선을 추출한다. 템플릿 매칭은 기준이 되는 작은 이미지를 큰 이미지 내에서 검색하여 가장 유사한 위치를 찾는 기법으로, 이를 통해 퍼즐 조각이 배경 이미지의 어느 위치에 맞는지 좌표를 계산할 수 있다.

CAPTCHA 위젯이 제공하는 배경 이미지와 퍼즐 조각 이미지를 다운로드하는 것을 시작으로, OpenCV를 활용해 각 이미지를 흑백 단일 채널로 변환하고, 퍼즐 이미지에서는 불필요한 투명 영역을 제거한다. 윤곽선 검출 과정에서 매칭 정확도를 높이기 위해 커널 연산을 적용해 윤곽선을 팽창시켜 희미한 경계선을 보완하고 연속성을 확보함으로써 보다 선명한 윤곽선 이미지를 확보한다. 이후 퍼즐의 윤곽선 이미지를 배경의 윤곽선 이미지로 템플릿 매칭하여 각 위치별 상관값을 가지는 위치를 탐색하고, 이를 슬라이더의 이동 거리로 치환한다. <Figure 8>에 표현된 바와 같이 계산된 거릿값은 브라우저 내 슬라이더 버튼의 이동에 활용되며, 이 과정에서 마우스의 불규칙한 움직임과 중간 지연을 무작위하게 부여함으로써 정확도를 높임과 동시에 인간의 행동을 모방할 수 있게 된다.

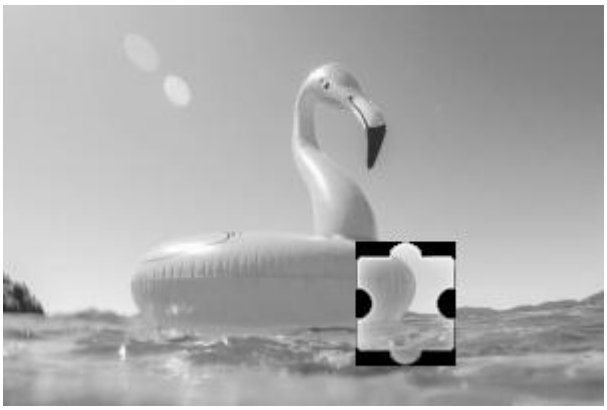

Figure 8. 템플릿 매칭을 통한 퍼즐 위치 탐색

GeeTest의 슬라이드 유형 CAPTCHA는 성공 여부와 관계없이 위젯이 사라지고, <Figure 9>와 같이 그 결과가 외부 div 태그 내에 기록된다. 해당 정보를 확인함으로써 CAPTCHA가 정상적으로 해결된 경우에는 사용자에게 이를 전달하고, 그렇지 않은 경우에는 위젯을 다시 표시하여 동일한 절차를 반복한다.


Figure 9. GeeTest 해결 이미지

3.3.2.2. 아이콘 유형 CAPTCHA

아이콘 유형 CAPTCHA는 전체 이미지 내에서 특정 아이콘의 위치를 찾아내고, 제시된 순서에 따라 각 아이콘을 클릭하도록 설계되어 있다. 본 연구에서는 GeeTest 서비스를 활용하여 해당 유형을 실험하였다. 객체의 시각적 특성을 인식함과 동시에 순서를 고려해야 하는 복합적인 특성을 지녀, 이에 따라 프롬프트 작성과 대형 언어 모델(LLM)의 시각-언어 통합 추론 능력을 활용해 문제 해결을 시도한다.

특히, OpenBMB 연구진이 개발한 다중 모달 대형 언어 모델인 MiniCPM-V-4_t-int4를 적용하였다. 이 모델은 텍스트와 이미지를 동시에 이해하고 추론할 수 있는 능력을 갖추고 있으며, 이미지 내 객체 인식 및 관계 추론에 강점을 보인다. 또한 4비트 양자화를 적용한 경량화 구조를 통해 모델 크기와 연산량을 크게 줄여 일반적인 GPU 환경에서도 CAPTCHA와 같은 문제를 처리할 수 있다.

그러나 해당 모델은 객체의 상대적 위치만을 추정할 수 있을 뿐 정확한 좌표를 직접 검출하지는 못한다. 또한 다수의 탐색 작업을 동시에 수행하면 연산 부담이 크게 증가하여 잘못된 응답을 반환할 가능성이 있다. 따라서 본 연구에서는 이러한 한계를 극복하기 위해 Chain-of-Thought(CoT) 방식을 도입하여 문제를 전체적으로 세 단계로 분해한 뒤 단계별로 순차적 추론을 수행함으로써 정확성과 효율성을 향상시키고자 한다.

ISSN(Print) : 1976-5304, ISSN(Online) : 2713-931X
https://kdfs.jams.or.kr, DOI : XXXXXXXXXXXXXXXX

<Table 2> 아이콘 기반 CAPTCHA 해결을 위한 프롬프트

| 프롬프트 | |
|---|---|
| 1차 | Describe the icon in detail in three sentences.<br>Focus on how the overall outline flows and connects,<br>capturing the general shape and movement of its form,<br>without mentioning internal elements or color. |
| 2, 3차 | This is an explanation of a specific icon : {icon}.<br>Find it and return the number of the grid that contains that icon.<br>If multiple grids contain the icon, return only one that contains the center of the icon.<br>Ensure that the result is only the number, with no extra text or formatting. |

첫째, 지시문에 포함된 아이콘 이미지를 개별적으로 분리 및 저장한 뒤, 각 아이콘에 대해 <Table 2>에 제시된 프롬프트를 활용하여 LLM으로부터 구체적인 설명 문구를 생성하였다. 지시문 속 아이콘은 검은색 실루엣 형태로 표현되어 있으나, 실제 탐색 이미지에서는 색상, 각도, 크기 등이 변형되어 나타나므로, 혼동을 최소화하기 위해 프롬프트에서 이러한 요소의 언급을 배제한다. 다음 단계에서는 앞서 식별한 아이콘의 대략적인 위치를 추정하기 위해 탐색 이미지에 격자를 설정, 번호를 부여한 후 아이콘이 포함된 격자 번호를 출력하도록 <Table 2>의 두 번째 프롬프트를 적용하였다. 여러 실험 결과, LLM이 격자와 아이콘을 동시에 인식하기 위해서는 <Figure 10>에서 보이듯 총 24개의 격자 분할이 가장 적절함을 확인하였다. 그러나 격자의 크기가 아이콘과 유사하여, 아이콘이 여러 격자에 걸쳐 있을 경우 LLM이 중심 격자만 선택하도록 제한했음에도 인접한 격자의 번호를 오인식하는 사례가 일부 발생하였다. 이에 따라 모델이 반환한 격자 번호를 기준으로 주변 영역을 확대 · 크롭한 후, <Figure 11>과 같이 이를 25개의 격자로 분할하여 두 번째 단계에서 사용한 동일한 절차를 적용해 정밀 탐색을 수행한다.


Figure 10. 격자를 포함한 CAPTCHA 이미지 예시


Figure 11. 확장된 세부 격자

각 아이콘 별로 두 개의 격자 번호를 반환받아 이를 토대로 이미지 내에서 아이콘이 위치한 좌표값을 계산하고, 지시된 순서별로 무작위한 지연 시간을 두고 선택한다. GeeTest 아이콘 유형의 경우, 검증을 시작하기 위해 화면 하단의 검증 버튼을 클릭해야 한다. 이후에는 슬라이드 유형과 동일한 절차를 적용하여 성공 여부를 판단하고 다음 동작을 결정한다.

## IV. 평가

### 4.1. 평가 환경

본 연구는 일반적인 수사 과정을 가정하여 특수한 고성능 서버가 아닌 일반적인 환경에서 실험을 수행하였다. <Table 3>에 구성된 바와 같이 두 가지의 실험 환경을 구성하여 모델의 실용성과 효율성을 비교함으로써, 제안한 기법이 현실적인 조건에서도 적용 가능한지를 검증하였다.

<Table 3> 각 환경별 소프트웨어 및 하드웨어 구성

| | | |
|---|---|---|
| 실험환경 (1) | CPU | Snapdragon X1E-80-100 |
| | RAM | 16GB |
| | GPU | Qualcomm Adreno X1-85 |
| | OS | Windows 11 Home |
| 실험환경 (2) | CPU | Intel i9-10980XE |
| | RAM | 256GB |
| | GPU | NVIDIA RTX 3090 24GB * 4 |
| | OS | Ubuntu 20.04 LTS |

4.2. 평가 지표 및 방법

본 연구에서 제안하는 프레임워크의 효용성을 평가하기 위해 성공률, 추론 시간, 소요 시간으로 구성된 세 가지 지표를 설정하였다. 성공률은 CAPTCHA의 존재 및 유형을 구분할 수 있는지와 이를 정확히 해결할 수 있는지를 판단하는 평가지표이다. 이는 각 유형에 대한 20번의 반복된 실험 중 몇 번을 안전히 우회하였는지를 계산해 지표에 대한 값을 산출한다. 또한 두 가지 환경 구성이 알고리즘의 수행에 있어 어느 정도 영향을 미치는지를 판단하기 위해 추론 시간과 소요 시간을 평가지표로 설정하였다. 추론 시간은 순수한 연산 수행 시간의 평균으로, 이미지 유형에서는 3x3 타일 형태의 CAPTCHA를 대상으로 YOLO 모델이 이미지를 입력받아 좌표를 반환하기까지의 시간을, 슬라이드 유형에서는 윤곽선 추출과 템플릿 매칭을 거쳐 좌표를 계산하는 과정을, 아이콘 유형에서는 LLM 호출과 최종 격자 번호 반환 과정의 수행 시간을 측정한다. 마지막으로 소요 시간은 CAPTCHA가 탐지된 시점부터 서비스 및 유형이 분류되고, 각 해결 절차를 거쳐 사용자에게 성공 신호가 반환될 때까지의 전체 과정의 평균 시간을 의미한다. 세 가지 지표를 설정함으로써, 본 프레임워크가 CAPTCHA를 얼마나 정확하고 신속하게 우회할 수 있는지, 그리고 다양한 환경 조건에서 효율적으로 작동하는지를 종합적으로 평가할 수 있다.

4.3. 평가 결과

<Table 4>는 본 연구의 평가 결과를 나타낸다. 이미지 유형의 경우, 두 환경 모두 100%의 성공률을 도출하였다. 이는 reCAPTCHA v2에서 요구하는 객체들이 한정적이며, 또한 강력한 좌표 검출 모델인 YOLO를 활용하여 보다 정확한 위치를 탐지할 수 있었기 때문이다. CNN을 기반으로 해 여러 패치들에 대해 병렬적인 계산을 요구하는 특성상 실험환경 (2)에서의 추론 및 소요 시간이 큰 차이로 우위를 점하였다. 슬라이드 유형의 경우 두 환경 모두 비슷한 연산 시간으로 높은 정확도를 반환했으며, 이는 OpenCV를 활용한 이미지 전처리 및 정밀한 위치 보정 과정에 기인한 것이다. 아이콘 유형의 경우, LLM의 한계를 극복하기 위해 보조 장치를 개발했음에도 불구하고 50% 이하의 성능을 보였다. 총 20회의 실험 결과, 60개의 아이콘 중 46개의 위치를 올바르게 추정하였으나, 세 개의 아이콘을 연속적으로 정확히 선택해야 문제를 해결할 수 있는 구조적 제약과, 일부 아이콘에 대해 근접한 좌표를 반환하는 사례가 발생하여 전반적인 성능 저하로 이어졌다. 또한, 경량화된 MiniCPM-V-4_t-int4 모델조차 파라미터 규모가 상당하여, 제한된 연산 자원을 가진 실험환경 (1)에서는 구동에 제약이 존재하였다.

ISSN(Print) : 1976-5304, ISSN(Online) : 2713-931X
https://kdfs.jams.or.kr, DOI : XXXXXXXXXXXXXXXX

<Table 4> 실험 결과

| 환경 | 평가 지표 | 이미지 유형 | 슬라이드 유형 | 아이콘 유형 |
|---|---|---|---|---|
| 실험환경 (1) | 성공률 | 100% (20 / 20) | 85% (17 / 20) | - |
| | 추론 시간 | 22.5초 | 0.06초 | - |
| | 소요 시간 | 45.2초 | 22.9초 | - |
| 실험환경 (2) | 성공률 | 100% (20 / 20) | 90% (18 / 20) | 45% (9 / 20) |
| | 추론 시간 | 0.56초 | 0.02초 | 20.8초 |
| | 소요 시간 | 19.7초 | 19.9초 | 42.4초 |

## V. 결 론

인터넷 상의 사이버 범죄가 고도화되고 빈번해지는 상황 속에서 실시간 감시와 광범위한 증거 수집은 즉각적인 대응과 사고 예방을 위해 필수적이다. 특히 여러 작업과 복합적 판단을 동시에 수행하기 어려운 인간의 한계를 보완하기 위해 '봇'이라 불리는 자동화 프로그램을 활용하려는 시도가 활발하다. 그러나 봇을 이용한 반복적 · 악의적 접속은 서버 과부화 및 보안 위협을 초래하므로, 많은 웹 서비스는 인간이 아닌 사용자의 접근을 차단하기 위해 CAPTCHA를 도입하고 있다. 이러한 목적에 있어 CAPTCHA는 그 역할을 충실히 수행하나, 과학 수사의 관점에서는 자동화된 수사 활동을 지연시켜 수사 효율을 저해하는 주요 요인으로 작용한다.

본 연구는 이러한 관점에서 여러 머신러닝 및 이미지 처리 알고리즘을 활용해 CAPTCHA의 탐지부터 해결까지 자동화함으로써 인간의 추가적인 개입 없이도 CAPTCHA를 우회할 수 있는 전체적인 프레임워크를 제안하였다. 이는 CAPTCHA의 탐지와 해결이라는 두 가지 단계로 구성된다. 첫 번째 탐지 단계에서는 HTML 구조와 네트워크 기록을 활용하여 봇이 작동을 중단했을 때 그 원인이 CAPTCHA에 의한 것인지를 판단한다. HTML 구조 내에서 iframe, div 등 특정 태그의 존재 여부를 확인해 일차적으로 CAPTCHA의 존재 여부를 판단하였고, 다양한 CAPTCHA 유형을 구분하기 위해 특정 스크립트가 서버에 호출하는 네트워크 기록 속 특정 인자값을 확인한다. 해당 결과를 바탕으로 두 번째 해결 단계로 진입하며, 본 단계에서는 앞서 구분한 유형별로 서로 다른 해결 알고리즘을 적용한다. 이미지 유형의 경우 사전 학습된 YOLO 모델을 활용해 이미지 내에 지시문에서 제공된 객체를 검출, 반환된 좌표 값을 활용해 문제를 해결한다. 슬라이드 유형에서는 OpenCV 라이브러리를 활용해 흑백 전환, 윤곽선 검출, 템플릿 매칭 기법을 활용해 퍼즐 조각이 전체 이미지 내 위치할 좌표를 탐색한다. 마지막 아이콘 유형에서는 지시문 내 명시된 순서와 원본 아이콘과 왜곡된 이미지 내에서의 연관성을 고려하는 등 복합적인 판단 과정이 필요해 CoT 기법을 적용한 3단계 LLM 접근 방식을 활용하였다. 추가적으로 마우스 움직임, 클릭 순서 등 CAPTCHA 내 숨겨진 보안 장치를 우회하기 위해 인간과 같이 불규칙적이고 무작위적인 움직임을 함께 구현하여 성공 가능성을 높였다. 탐지 단계에서 활용한 방법을 바탕으로 CAPTCHA가 해결되었는지를 판단, 우회에 성공하였으면 사용자에게 기존 작업을 이어갈 수 있다는 내용을 전달하고 그렇지 않은 경우 본 과정을 반복한다. 실험 결과, 이미지, 슬라이드, 아이콘 유형에서 각각 100%, 90%, 45%의 정확도를 보임으로써 본 프레임워크의 효용성 및 적용성을 증명하고, 수사 효율성을 향상시킬 수 있음을 보였다.

하지만 상대적으로 하드웨어 사양이 낮고 연산 속도가 느린 환경에서는 LLM 모델을 적용할 수 없었으며 소요 및 추론 시간에서 열세를 보였다. 또한 사전 학습된 모델을 불러와 활용하는 일부 유형의 경우, 모델 로딩 시간 자체가 수사 과정의 지연을 초래할 수 있다는 한계가 존재한다. 따라서 본 연구에서는 적절한 성능의 GPU를 탑재한 서버를 구축하고, CAPTCHA 발생 시 연산을 해당 서버에 위임하여 처

리하는 방안을 제안한다. 또한 아이콘 유형 CAPTCHA의 자동 인식에 LLM과 CoT 방식을 도입하고 격자 기반 탐색을 적용하였으나, 아이콘을 묘사한 텍스트만으로는 이미지 내에서의 정확한 탐색에 한계가 있었다. 이를 보완하기 위해 여러 이미지를 입력받아 비교 · 분석할 수 있는 멀티 이미지 모델의 도입을 검토하고, 회전 및 왜곡 등 변형된 이미지 쌍을 학습시키는 동시에 격자에 대한 이해도를 향상시키도록 LLM을 전이 학습하는 연구를 지속하고자 한다.

이와 같은 한계 속에서도 본 연구는 두 가지 의의를 지닌다. 첫째, CAPTCHA의 감지와 해결을 통합한 자동화 프레임워크를 제시함으로써, 인간의 개입 없이도 봇을 활용한 웹 환경의 정보 수집 및 분석 작업이 가능함을 실증하였다. 반복적이고 대량의 정보 탐색 작업을 컴퓨터에게 완전히 전가함으로써, 수작업으로는 불가능한 규모의 데이터 수집 및 검증을 신속히 수행할 수 있는 길을 열어 사건 대응 및 잠재적 사고 예방 측면에서 높은 실효성을 가진다. 둘째, CAPTCHA 감지와 해결 단계에 대한 구체적인 절차를 제시하고, 이를 실제 실험을 통해 적용 가능함을 입증함으로써 향후 새로운 형태의 CAPTCHA가 등장하더라도 이를 분석 및 대응할 수 있는 가이드라인을 제시하였다. 특히 CAPTCHA 우회에 특화되지 않은 공개 LLM을 활용하여도 일정 수준의 인식 및 추론이 가능함을 보임으로써, 대형 언어 모델이 보안 · 인증 환경에서도 실질적 활용 잠재력을 지닌다는 점을 다시금 확인하였다.

본 연구는 다양한 유형의 CAPTCHA에 대한 이해를 높이고, 머신러닝을 포함한 여러 알고리즘을 적용하여 CAPTCHA를 해결하는 전반적인 흐름을 제시하였다. 향후에는 본 연구에서 다룬 이미지, 슬라이드, 아이콘 유형을 넘어 더 다양한 CAPTCHA 유형을 지원할 수 있도록 프레임워크를 지속적으로 확장할 예정이다. 또한 대형 언어 모델(LLM)을 활용하여 대부분의 CAPTCHA 유형에 통합적으로 적용할 수 있는 방법을 탐구하고, 각 유형별 개별적으로 개발된 알고리즘과 그 성능을 비교 · 분석하는 후속 연구를 수행하고자 한다.

## 참 고 문 헌

[1] Kumar, Mohinder & Jindal, Kushal & Kumar, Munish. (2021). A Systematic Survey on CAPTCHA Recognition: Types, Creation and Breaking Techniques. Archives of Computational Methods in Engineering. 29. 10.1007/s11831-021-09608-4.

[2] Tariq, Noshina & Khan, Farrukh & Moqurrab, Syed & Srivastava, Gautam. (2023). CAPTCHA Types and Breaking Techniques: Design Issues, Challenges, and Future Research Directions. 10.48550/arXiv.2307.10239.

[3] Plesner, Andreas & Vontobel, Tobias & Wattenhofer, Roger. (2024). Breaking reCAPTCHAv2. 10.48550/arXiv.2409.08831.

[4] Nguyen, V. D., Chow, Y. W., & Susilo, W. (2011, November). Breaking a 3D-based CAPTCHA scheme. In International Conference on Information Security and Cryptology (pp. 391-405). Berlin, Heidelberg: Springer Berlin Heidelberg.

[5] Deng, G., Ou, H., Liu, Y., Zhang, J., Zhang, T., & Liu, Y. (2024). Oedipus: LLM-enchanced reasoning captcha solver. arXiv preprint arXiv:2405.07496.

[6] Teoh, X., Lin, Y., Li, S., Liu, R., Sollomoni, A., Harel, Y., & Dong, J. S. (2025). Are {CAPTCHAs} Still Bot-hard? Generalized Visual {CAPTCHA} Solving with Agentic Vision Language Model. In 34th USENIX Security Symposium (USENIX Security 25) (pp. 3747-3766).