Vision Transformer를 활용한 분류 기반의 접근 방식을 통한 ReCAPTCHA v2 챌린지 해결

Solving ReCAPTCHA v2 Challenge using Classification-based Approach with Vision Transformer

김보미 (Bomi Kim)

한국외국어대학교 컴퓨터공학부 [kimbomi0819@naver.com](mailto:kimbomi0819@naver.com)

본 논문은 2023학년 6월에 제출된 한국외국어대학교 컴퓨터공학부 졸업논문입니다.

지도교수: 장익범 교수님

서명: 

Abstract

한글 요약 : 이 논문은 Vision Transformer를 활용한 분류 기반의 접근 방식을 통해 Recaptcha 챌린지를 해결하는 것을 다루고 있습니다. 기존의 객체 검출 방식인 YOLO의 문제점을 보완하며, 우수한 성능을 보이는 것을 확인하였습니다. 연구에서는 Vision Transformer의 강인한 특성과 작은 객체 분류에 대한 우수성을 설명하고, 이전에 제안된 접근 방식의 이점과 한계를 분석하였습니다. 실험 결과를 통해 제안된 접근 방식의 성능 향상을 입증하고, 작은 객체 분류에 대한 기존 연구와의 차이를 확인하였습니다. 이를 통해 Vision Transformer를 활용한 분류 기반의 접근 방식이 Recaptcha 챌린지에서의 가치와 기여도를 확인할 수 있었습니다.

핵심어 : Vision Transformer, 분류 기반 접근 방식, 작은 객체 분류, Recaptcha 챌린지

영문 요약 : This paper explores the use of a classification-based approach utilizing Vision Transformer to address the ReCAPTCHA challenge. By addressing the drawbacks of the existing object detection method, YOLO, we have observed its superior performance. Our study highlights the robust characteristics of Vision Transformer and its effectiveness in classifying small objects, while analyzing the advantages and limitations of previously proposed approaches. Through experimental results, we validate the improved performance of the proposed approach and highlight the distinctions from prior research in small object classification. Thus, demonstrating the value and contribution of employing Vision Transformer in a classification-based approach for the ReCAPTCHA challenge.

Keywords : Vision Transformer, classification-based approach, small object classification, Recaptcha challenge

1. Introduction
   1. **연구 배경**

인터넷 사용량의 증가와 함께 보안과 관련된 문제가 점점 중요성을 더하고 있습니다. 특히, 자동화 봇에 의한 악용 사례가 증가하고 있으며, 이를 방지하기 위해 많은 웹사이트에서 ReCAPTCHA와 같은 보안 챌린지를 도입하고 있습니다.

스크린샷, 야외, 나무, 차량이(가) 표시된 사진

자동 생성된 설명

그림 1.1.1 : reCAPTCHA v2

ReCAPTCHA는 사용자가 인간인지 로봇인지를 판별하기 위한 시스템으로, 보통 이미지 내의 특정 객체를 인식하도록 설계되어 있습니다.

기존의 ReCAPTCHA 해결 방법은 객체 검출 방식을 주로 사용하여 특정 객체를 인식하고 경계 상자를 예측하는 방식입니다. 그러나 이러한 방식은 bounding box를 지정하는 과정에서 초과 타일로 잘못 분류되는 문제가 발생할 수 있습니다. 이는 잘못된 분류 결과와 계산 비용 증가 등의 문제를 야기할 수 있습니다.

따라서, 객체 검출 방식 대신 분류 기반의 접근 방식을 사용하여 ReCAPTCHA 챌린지를 해결하는 것이 중요한 연구 동기가 됩니다. 분류 기반의 접근 방식은 전역적인 이미지를 고려하며 초과 타일과 관련된 특징을 더 잘 파악할 수 있습니다. 또한, 계산 비용을 절감하고 다양한 객체 클래스를 예측할 수 있는 유연성을 가지고 있습니다.

* 1. **연구 주제 설명**

이 연구의 명확한 주제는 "ReCAPTCHA 챌린지를 객체 검출 방식이 아닌 분류를 통해 해결하는 접근 방식의 탐구"입니다. 기존의 객체 검출 방식 대신 분류 기반의 접근 방식을 사용하여 성능을 개선하고, ViT(Vision Transformer) 모델을 활용하여 ReCAPTCHA 챌린지의 해결을 탐구합니다.

구체적으로, 연구는 다음과 같은 내용을 다룹니다:

1. ReCAPTCHA 이미지를 패치 단위로 분류하는 방법의 개발: ViT 모델을 사용하여 ReCAPTCHA 이미지를 작은 패치로 분할하고, 각 패치를 분류합니다.
2. 이진 분류를 통한 ReCAPTCHA 챌린지 해결: 분류된 패치를 기반으로 ReCAPTCHA 챌린지의 해결을 위해 이진 분류를 수행합니다. 패치가 챌린지 내의 특정 요소를 나타내는지 여부를 판단합니다.
3. 기존 객체 검출 방식과의 비교 및 성능 평가: 제안된 분류 기반 접근 방식과 기존의 객체 검출 방식인 YOLO를 비교하여 평가합니다. 이를 통해 분류 기반 접근 방식의 장점을 확인하고, ReCAPTCHA 챌린지 해결에 효과적인지를 분석합니다.

이 연구는 분류 기반의 접근 방식과 ViT 모델의 잠재력을 탐구하고, 객체 검출 방식의 문제점을 보완하며 성능 비교를 통해 새로운 해결 방법을 제시하는 것을 목표로 합니다.

* 1. **기존 연구와 비교**

주제에 대한 이전 연구가 수행되었습니다. 세 가지 유형의 CAPTCHA를 기계 학습으로 해결하는 연구가 2016년에 수행되었습니다. 이 연구의 솔버는 70.8%의 성공률을 보였습니다. Google의 reCAPTCHA v2를 해결하기 위해 객체 감지 기반 솔버를 사용하는 또 다른 연구는 2020년에 수행되었으며 83.3%의 성공률을 보였습니다.

기존의 객체 검출 방식을 사용한 연구들은 ReCAPTCHA 챌린지를 해결하기 위해 주로 bounding box를 예측하는 방식을 채택하였습니다. 그러나 이러한 방식은 초과 타일로 잘못 분류되는 문제가 발생하고, 계산 비용이 높아지는 등의 한계점을 가지고 있습니다.

Ref\_[1][2]

스크린샷, 건물, 바퀴, 육상 차량이(가) 표시된 사진

자동 생성된 설명

그림 1.3.1 : bounding box가 너무 많은 그리드 셀을 포함하는 오토바이 이미지

본 연구는 분류 기반의 접근 방식을 사용하여 ReCAPTCHA 챌린지를 해결하는 것을 제안합니다. 즉, ReCAPTCHA 이미지를 패치 단위로 분리하고, 각 패치를 분류하여 이진 분류 문제로 해결합니다. 이를 위해 Vision Transformer (ViT) 모델을 사용하여 각 패치의 특징을 추출하고 분류합니다.

논문의 연구 결과는 기존의 객체 검출 방식에 비해 몇 가지 이점을 제시합니다.

첫째, 분류 기반의 접근 방식은 초과 타일로 잘못 분류되는 문제를 줄일 수 있습니다. 기존의 객체 검출 방식은 경계 상자에 포함된 그리드 셀의 수에 따라 초과 타일로 잘못 분류될 수 있지만, 분류를 통해 전역적인 이미지를 고려하면 초과 타일과 관련된 특징을 더 잘 파악할 수 있습니다.

둘째, 분류 기반의 접근 방식은 객체 검출 방식보다 계산 비용이 효율적입니다. 객체 검출은 각 그리드 셀마다 별도의 추론을 수행해야 하므로 계산 비용이 높습니다. 하지만 분류를 통해 전체 이미지를 한 번의 추론으로 처리할 수 있으므로 계산 비용을 줄일 수 있습니다.

셋째, 분류 기반의 접근 방식은 다양한 객체 클래스를 예측할 수 있는 유연성을 가지고 있습니다. 객체 검출은 경계 상자의 위치와 클래스를 예측하는 데 초점을 맞추지만, 분류는 이미지에 포함된 다양한 객체 클래스를 예측할 수 있습니다.

이러한 결과는 Recaptcha 챌린지의 해결에 대한 새로운 시각을 제시하며, 작은 객체 분류에 ViT를 활용한 접근 방식의 유용성을 보여줍니다. 논문에서는 이러한 결과를 바탕으로 ViT를 적용한 분류 기반의 접근 방식을 제안하고, 이를 통해 Recaptcha 챌린지의 성능을 향상시키는 것을 목표로 설정하였습니다.

* 1. **Recaptcha challenge유형 분류**

이전 논문에서 4x4 사이즈보다, 3x3 사이즈에서 더 많은 오류율을 낸 점을 비추어, 3x3 size의 이미지 챌린지에 초점을 맞추고 해보겠습니다.

본 논문에서는 3x3 사이즈로 잘라진 분할된 이미지를 ‘작은 이미지’로, 원본의 전체 이미지를 ‘큰 이미지’로 표현합니다.

Google reCAPTCHA v2 이미지 챌린지는 두 가지 유형으로 구분할 수 있습니다.

첫번째 방법은 “큰 이미지를 분류”해내는 것입니다.

야외, 텍스트, 스크린샷, 차량이(가) 표시된 사진

자동 생성된 설명

그림 1.4.1 큰 이미지 분류 : ‘자전거’ 예측

해당 방법은 클래스 정보가 포함된 이미지를 모두 선택하여 분류하는 방식입니다. 이진 분류에서 자전거를 찾는 경우, 다른 클래스로 예측된 경우에도 해당 이미지가 자전거인지 아닌지를 구분하여 정확도를 높이는 방법을 시도하려고 합니다.

이러한 방식은 문제의 특성을 잘 이용하여 정확도를 향상시킬 수 있다는 장점이 있습니다. 예를 들어, 자전거를 찾아야 하는데, 다른 클래스로 예측된 경우에도 자전거인지 아닌지를 명확히 구분하여 예측 결과를 출력합니다. 다른 클래스로 예측되더라도 해당 이미지가 자전거인지 아닌지를 정확하게 구분합니다. 이를 통해 모델의 정확도를 더욱 높일 수 있습니다.

두번째 방법은 “사진 한 장을 분할하여 특정한 객체를 찾기”입니다.

야외, 하늘, 스크린샷, 텍스트이(가) 표시된 사진

자동 생성된 설명

그림 1.4.2 작은 이미지 분류

이 방법은 이미지의 일부 영역을 분할하여 특정한 객체를 찾는 것을 목표로 합니다. 저는 3x3 size로 잡았으므로, 3x3으로 분할합니다.

1. 연구 방법 및 결과
   1. **데이터**

본 연구에서는 Recaptcha v2 챌린지를 위해 kaggle사이트 (Ref\_[3])에서11,671개의 이미지를 수집하였습니다. 각 이미지는 Bicycle, Bridge, Bus, Car, Chimney, Crosswalk, Hydrant, Motorcycle, Other, Palm, Stair, Traffic Light과 같이 12개의 카테고리로 분류되어 있습니다.

* + Bicycle : 780 images
  + Bridge : 533 images
  + Bus : 1209 images
  + Car : 3496 images
  + Chimney : 124 images
  + Crosswalk : 1240 images
  + Hydrant : 955 images
  + Motorcycle : 81 images
  + Other : 1340 images
  + Palm : 911 images
  + Stair : 211 images
  + Traffic Light : 791 images

총 11,671개 이미지를 수집하였습니다.

텍스트, 야외, 스크린샷, 나무이(가) 표시된 사진

자동 생성된 설명

그림2.1.1 : 훈련데이터로, 각 class에 대한 크기가 224x224인 이미지 데이터 9개 표현

훈련데이터로는 12개의 class에 대해 9340개의 이미지를 사용하였으며, 검증 데이터로는 전체 데이터의 20%인 2331개의 이미지를 사용하여 분류 성능을 측정하였습니다.

‘Other’ 클래스는 훈련 데이터에서만 사용되었으며, 테스트 데이터에서는 해당 클래스의 분류를 수행하지 않았습니다.

첫번째 분류 방법인 ‘큰 이미지 분류’에서는 각 class당 600장의 이미지로 테스트 데이터를 만들었습니다. 예를 들어, bicycle을 고르라는 챌린지가 주어졌을 때의, bicycle사진 200장, bicycle이 아닌 사진 400장을 테스트 데이터로 설정하였습니다.

두번째 분류 방법인 ‘작은 이미지 분류’에서의 테스트 데이터는 각 class당 원본 이미지 중 900개를 3x3 크기로 잘라서 작은 이미지로 사용하여 이진 분류를 수행하였습니다. 그리고 테스트 데이터의 성능을 내기 위해 각 이미지를 각 class에서의 그림이 선택되는 것에 대해 이진 분류를 수행하기 위해 라벨링을 해주었습니다.

야외, 스크린샷, 텍스트, 바퀴이(가) 표시된 사진

자동 생성된 설명

그림 2.1.2: 테스트 데이터 : 작은 이미지 9개

패턴, 대칭, 다채로움, 예술이(가) 표시된 사진

자동 생성된 설명

그림 2.1.3 : 테스트 데이터 – 작은 이미지 : ‘Bicycle’클래스 900장

데이터 전처리 단계에서는 데이터 증강 기법을 사용하였습니다. 모든 이미지의 픽셀 값을 0에서 1로 정규화하고, 이미지 확대 및 축소 범위를 지정하여 다양한 시각적 변화를 도입하였습니다. 또한, 이미지의 좌우 대칭성을 고려하여 수평으로 뒤집기를 적용하였습니다.

* 1. **Base model : Transfer learning based on Inception v3**

ImageNet 데이터셋으로 사전 훈련된 가중치를 사용하여 Inception v3 모델을 기본 모델로 선택했습니다. 검증 결과 손실은 0.9744이고 정확도는 0.8130으로 나타났습니다. 앞서 할 모델은 이 기본 모델보다 더 좋은 성능을 발휘할 것입니다.

* 1. **모델 선정 : Vision Transformer**

텍스트, 스크린샷, 도표, 폰트이(가) 표시된 사진

자동 생성된 설명

그림 2.3.1: Vision Transformer

Vision Transformer(ViT)는 이미지 분류를 위한 딥러닝 모델입니다. 이 모델은 기존의 CNN 대신 transformer 아키텍처를 이미지에 적용한 것이 특징입니다. 이미지는 고정된 크기의 패치로 자르고, 각 패치를 선형 임베딩하여 위치 정보와 함께 transformer 인코더에 입력합니다. 분류를 위해 classfication token을 추가되어 이미지의 전체적인 컨텍스트를 인코딩하고 클래스를 예측합니다.

ViT는 이미지의 전체적인 패턴을 인식하는 데 강점을 가집니다. reCAPTCHA v2 이미지에는 다양한 패턴과 변형이 포함됩니다. ViT는 이러한 패턴을 상호작용을 통해 학습하고 분류할 수 있는 능력을 갖추고 있어서 선택하게 되었습니다. 패치 간의 관계를 모델링하면서 이미지의 전체적인 구조와 패턴을 이해하고 클래스를 예측하는 데 도움이 되기 때문입니다.

(Ref\_[4])

* 1. **모델 훈련**

저는 ViT 모델의 B\_32(base 32)구조를 선택했습니다. Vit\_b32는 상대적으로 작은 모델으로 알려져 있으며, 제한된 컴퓨터 리소스와 계산 효율성을 고려해 선택하였습니다. 또한, 대규모 데이터셋에서 사전 훈련된 가중치를 사용하여 성능을 향상시켰습니다. 입력 이미지는 32x32 크기의 패치로 분할되어 처리되었습니다.

실험 설정에서는 Adam 옵티마이저를 사용하였고, 학습률은 1e-4로 설정하였습니다. 또한 Early Stopping 기법을 적용하여 검증 손실이 3번 연속으로 개선되지 않을 경우 학습을 조기 종료하였습니다.

첫번째 방법인 ‘큰 이미지 분류’에서는 각 클래스 별로 분류를 수행하였습니다. 예를들어, ‘자전거’클래스의 경우, 자전거를 포함한 이미지를 1/3 비율로 샘플링하고, 다른 클래스에서 무작위로 선택한 이미지를 2/3비율로 샘플링하여 테스트에 사용했습니다. 이 과정에서 자전거가 아닌 이미지는 클래스 0으로 예측하고, 자전거가 포함된 이미지는 클래스1로 예측하게 됩니다.

두번째 방법인 ‘작은 이미지 분류’에서는 각 클래스 당 분류를 수행하였습니다. 테스트 데이터로는 작은 이미지를 각각의 class에 대해 900장을 테스트하였습니다. 즉, 각 클래스당 큰 이미지 100장으로 분류를 수행하였습니다.

* 1. **테스트 결과**

2.5.1 큰 이미지 분류 결과

검증 데이터의 정확도는 0.848, 손실은 0.576으로 나타났습니다.

테스트 데이터에서 각 클래스 별 정확도는 다음과 같습니다.

챌린지의 취지에 맞게, 큰 이미지 분류 테스트 결과를 보이겠습니다.

각 클래스 당, 그 클래스가 속한 것을 약 1/3으로, 그 클래스를 뺀 클래스들을 약 2/3으로 하여 이진 분류를 수행했을 때의 정확도, 조화평균값입니다. 예를 들면, ‘버스’를 고르라고 했을 때의 챌린지에 정답에 도달할 확률의 정확도, 조화평균값을 나타낸 것이라고 볼 수 있습니다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Class** | **Accuracy** | **F1-score about**  **‘해당 class 제외한 모든 class’** | **F1-score about ‘해당 class’** | **개수** |
| **Bicycle** | 0.99 | 1.00 | 0.99 | 580 |
| **Bridge** | 0.97 | 0.98 | 0.96 | 568 |
| **Bus** | 0.99 | 0.99 | 0.99 | 574 |
| **Car** | 0.95 | 0.96 | 0.93 | 567 |
| **Chimney** | 0.97 | 0.98 | 0.95 | 569 |
| **Crosswalk** | 0.95 | 0.97 | 0.93 | 571 |
| **Hydrant** | 1.00 | 1.00 | 1.00 | 568 |
| **Motorcycle** | 0.98 | 0.99 | 0.96 | 375 |
| **Palm** | 0.99 | 0.99 | 0.98 | 571 |
| **Stair** | 0.96 | 0.97 | 0.94 | 573 |
| **Traffic Light** | 0.94 | 0.95 | 0.90 | 576 |
| **Total** | **0.97** | **0.98** | **0.95** | **6092** |

표 2.5.1.1 : 각 클래스 별 큰 이미지 분류 성능

전체적으로, 큰 이미지 분류에서는97%의 정확도를 보였으며, 해당 클래스를 제외한 모든 클래스에 대한 f1-점수로는 98%로 높은 성능을 보였습니다. 또한 해당 클래스에 대한 f1-점수는 95%로 좋은 성능을 보였습니다.

전반적으로, 큰 이미지 분류에서는 좋은 성능을 보였으며, 각 클래스에 대한 분류 결과도 높은 정확도와 f1-점수를 보였습니다. 이는 모델이 다양한 클래스를 잘 구분할 수 있는 능력을 가지고 있음을 시사합니다.

야외, 잔디, 텍스트, 거리이(가) 표시된 사진

자동 생성된 설명텍스트, 하늘, 야외, 건물이(가) 표시된 사진

자동 생성된 설명텍스트, 야외, 도로, 도로 표면이(가) 표시된 사진

자동 생성된 설명

그림 2.5.1.2 : 잘못 분류된 이미지

잘못 분류된 이미지들을 보면, 거의 두 물체가 같이있어 두 물체 중 하나로 다른 class를 예측한 것입니다. 이렇게 잘못 분류되는 것들을 보면 모델이 작은 객체까지 잘 인지하고 있다는 상태를 알 수 있습니다. 큰 이미지 분류에서는 성능이 감소하는 측면을 보여줄 수 있습니다. 그러나 작은 이미지 분류에 사용할 때는 작은 객체까지 검출 가능하다는 측면에서 긍정적인 효과를 보일 수 있습니다.

2.5.2 작은 이미지 분류 성능

각 클래스당 분할된 이미지 900개, 즉 원본 이미지 100개에 대해 예측을 잘해내는지 보았습니다. 예외적으로, motorcycle은 데이터양이 부족하여 729개로 수행하였습니다.

이진 분류를 위해서 정답 레이블을 제가 라벨링해주었습니다.

다음으로는, 각 클래스별로 분류 성능과 어려움을 파악할 수 있는 분석 결과입니다.

|  |  |  |  |
| --- | --- | --- | --- |
| **Class** | **Accuracy** | **F1-score about**  **‘해당 class가 아닌 small image’** | **F1-score about**  **‘해당 class인 small image’** |
| **Bicycle** | 0.92 | 0.95 | 0.78 |
| **Bridge** | 0.80 | 0.86 | 0.67 |
| **Bus** | 0.80 | 0.87 | 0.54 |
| **Car** | 0.90 | 0.93 | 0.81 |
| **Chimney** | 0.92 | 0.95 | 0.72 |
| **Crosswalk** | 0.81 | 0.87 | 0.67 |
| **Hydrant** | 0.92 | 0.94 | 0.88 |
| **Motorcycle** | 0.87 | 0.92 | 0.58 |
| **Palm** | 0.88 | 0.91 | 0.81 |
| **Stair** | 0.78 | 0.87 | 0.29 |
| **Traffic Light** | 0.92 | 0.95 | 0.81 |
| **Total** | **0.87** | **0.91** | **0.68** |

표 2.5.2.1 : 각 클래스 별 작은 이미지 분류 성능

전체적으로, 작은 이미지 분류에서는 87%의 정확도를 보였으며, 해당 클래스가 아닌 작은 이미지에 대한 F1-점수는 91%, 해당 클래스인 작은 이미지에 대한 F1-점수는 68%로 평가되었습니다.

일부 클래스에 대해서는 작은 이미지 분류가 어려움을 겪는 것으로 나타났습니다. 예를 들어, Stair 클래스의 경우 정확도가 78%로 낮았고, 해당 클래스인 작은 이미지에 대한 f1-점수도 29%로 매우 낮았습니다. 이는 작은 이미지에서 계단을 정확하게 분류하기 어려웠음을 나타냅니다.

반면에, ‘Hydrant’클래스의 경우 정확도가 92%로 높았으며, 해당 클래스인 작은 이미지에 대한 f1-점수도 88%로 높은 성능을 보였습니다. 이는 작은 이미지에서 소화전의 분류를 아주 잘 수행된 것으로 보입니다.

* + Bicycle

Bicycle 클래스는 대체로 잘 분류되었습니다. 사람 머리와 자전거를 분류를 잘해내었으며, 작은 객체도 잘 잡아내었습니다. 하지만, 자전거가 모여있는 경우는 잘 분류하지 못하였습니다.

* + Bridge

굵은 다리를 잘 분류하는 경향이 있었습니다. 그러나, 어두운 이미지의 경우 인식이 어렵거나 매우 얇은 다리는 분류하지 못하는 경우가 있었습니다.

* + Bus

작은 객체는 잘 분류하지만, 확대된 이미지나 가려진 부분은 잘 분류하지 못하는 경향이 있었습니다.

* + Car

작은 객체를 잘 분류하며, 어두운 차도 잘 분류했습니다. 그러나 가려진 부분은 분류하지 못하는 경우가 꽤 있었습니다.

텍스트, 패브릭, 야외이(가) 표시된 사진

자동 생성된 설명야외, 텍스트, 잔디, 육상 차량이(가) 표시된 사진

자동 생성된 설명

그림 2.5.2.2: 차 - 작은 객체도 잘 분류해내는 car(1188), 가려진 것은 분류하지 못했습니다. (1206)

* + Chimney

건물, 벽돌, 벽, 야외이(가) 표시된 사진

자동 생성된 설명

그림 2.5.2.3 : 굴뚝 2개인 이미지

잘못 분류 : 큰 굴뚝을 잡아내지 못하고 작은 굴뚝을 잡아내었습니다.

* + Crosswalk

횡단보도의 검정색, 흰색인 대비되는 색깔을 잘 구분하여 분류하는 경향이 있었습니다. 그러나 블러 효과가 있는 이미지의 경우 분류가 어려웠습니다

횡단보도 외의 도로에 써있는 표시를 횡단보도로 잘못 인식하는 경향이 있었습니다.

* + Hydrant

잘 분류된 편이며, 작은 객체도 잘 인식하는 경향이 있었습니다.

* + Motorcycle

작은 객체를 잘 분류하는 경향이 있었습니다. 그러나 어두운 이미지에서는 분류가 거의 되지 않았습니다.

텍스트, 야외, 도로, 건물이(가) 표시된 사진

자동 생성된 설명

그림 2.5.2.4 : 오토바이 - 작은 객체 잘 분류

* + Palm

작은 객체를 잘 분리해내며, 야자수의 모양이 뚜렷한 점을 잘 인식하는 경향이 있었습니다. 또한, 블러 효과가 다른 클래스에 비해 객체를 잘 인식하는 편이었습니다.

야외, 육상 차량, 차량, 잔디이(가) 표시된 사진

자동 생성된 설명

그림 2.5.2.5 : 야자수 - 작은 객체 여러 개 잘 분류

식물, 나무, 야외, 야자과이(가) 표시된 사진

자동 생성된 설명

그림 2.5.2.6 : 야자수 - 많은 나무 중 야자수 잘 분류

야외, 하늘, 나무, 야자수이(가) 표시된 사진

자동 생성된 설명야외, 하늘, 식물, 야자과이(가) 표시된 사진

자동 생성된 설명

그림 2.5.2.7 : 야자수 - 가장자리 부분도 잘 분류

야외, 해변, 지상, 패브릭이(가) 표시된 사진

자동 생성된 설명

그림 2.5.2.8 : 야자수 - 블러 효과 다른 클래스에 비해 잘 캐치함

* + Stair

Stair 클래스의 분류는 어려웠습니다. 특히 그림자 진 계단 같은 경우에는 더 분류가 되지 않았습니다.

* + Traffic Light

불빛인 빨간색, 초록색을 잘 인식하며 분류가 잘 되었습니다. 어두운 이미지와 블러 효과가 있는 이미지에서도 비교적 분류가 잘 되었습니다. 비교적 불빛이 없는 신호등은 잘 인지를 못했으며, 표지판으로 인식이 잘못 될 때도 있었습니다. 하지만 작은 객체도 잘 인식하는 편이었습니다.

야외, 하늘, 나무, 식물이(가) 표시된 사진

자동 생성된 설명

그림 2.5.2.9 : 신호등 - 어두운 경우+ 블러 잘 분류, 작은 객체 잘 인지

야외, 텍스트, 하늘, 구름이(가) 표시된 사진

자동 생성된 설명

그림 2.5.2.10 : 신호등 - 잘못 분류 : 빨간색 신호등만 인식

위의 분석을 통해, 각 클래스 별로 분류 성능과 어려움을 파악할 수 있었습니다.

1. 결과 분석

Google의 reCAPTCHA v2를 해결하기 위해 객체 감지 기반 해결책을 사용하는 이전 연구는 2020년에 수행되었으며 83.3%의 성공률을 보였습니다.

본 실험에서 큰 이미지 분류는 평균적으로 97%의 성공률, 작은 이미지 분류는 평균적으로 87%의 성공률로, 총 평균적으로 92%의 성공률으로 ViT모델로 분류 기법을 사용했을 때 성능이 더 향상됨을 보였습니다.

또한, ViT모델은 YOLO의 문제점이었던, 초과 타일까지 물체를 감지하게 되었던 점을 완화해주었습니다.

스크린샷, 야외이(가) 표시된 사진

자동 생성된 설명

그림 3.1 : YOLO로 객체 기반 감지했을 때의 motorbycle 이미지

제일 위 그리드부터 (0,0), (0,1), (0,2), (1,1) 이런 식으로 표현했을 때 객체 기반 방법은 bounding box를 통해 (0,0), (0,1), (0,2)도 모두 motorbycle로 인식되었습니다.

ViT로 분류 기법을 사용했을 때는 (0,0), (0,1), (0,2)를 motorbycle로 인식을 안하는 확률이 증가하여, 초과 타일까지 물체를 감지하게 되었던 문제를 완화해주었습니다.

또한, 이전 논문에서는 신호등을 예로 들어, YOLO 객체 탐지기법을 사용하면 bounding box와 신호등이 모두 직사각형이기 때문에 인식이 더 잘 된다고 언급하였습니다. 분류 기법 역시 신호등을 잘 분류해내었습니다.

이 논문에서는 분류 기반 접근 방식의 장점을 확인하기 위해 ReCAPTCHA 챌린지 해결에 대한 효과를 분석하였습니다. 분류 기반 접근 방식을 사용한 ViT모델은 객체 인식 능력이 뛰어나며, 이를 통해 ReCAPTCHA 챌린지를 효과적으로 해결할 수 있음을 분석하였습니다. ViT모델은 이미지 내의 객체를 잘 분류하고 인식하며, 사람과 구분하기 어려운 이미지나 왜곡된 이미지에도 상대적으로 높은 성능을 보였습니다.

따라서, 이 연구 결과는 분류 접근 기반의 ViT 모델이 ReCAPTCHA 챌린지 해결에 효과적인 도구로 활용될 수 있음을 제시하고 있습니다. 분류 기반 접근 방식의 장점을 활용하여 객체 검출과 인식 과정에서의 성능 향상이 가능함을 시사하고 있습니다.

1. 결론 및 향후 연구 방향

본 연구는 ViT 모델을 활용한 객체 검출에 대한 성능 평가와 ReCAPTCHA 챌린지 해결에 대해 분석하였습니다. ViT모델은 YOLO와 비교하여 객체 검출에서의 일부 단점을 보완하였으며, 분류 기반 접근 방식의 장점을 활용하여 ReCAPTCHA 챌린지 해결에 효과적인 도구로 활용될 수 있음을 확인하였습니다.

* 1. 아쉬운 점과 향후 연구 방향

연구에서는 몇 가지 아쉬운 점을 발견하였습니다.

첫째로, 연구 과정에서는 작은 이미지에 대한 라벨링 작업 시 사람이 봐도 구분하기 어려운 블러 효과나 너무 어두운 이미지와 같은 경우에 대해 애매한 상황이 발생하였습니다. 또한 ‘bridge’의 경우 다리의 정의를 어디까지로 설정해야 하는지에 대한 헷갈림이 있었습니다. 예를 들어, 다리에 연결된 기둥까지도 다리의 일부로 인식해야 하는지 여부에 대한 경계를 어디까지 설정해야 할지 애매했던 측면이 있습니다. 이는 주관적인 판단이 요구되는 부분으로, 라벨링 과정에서 개별 라벨러들 사이에 차이가 발생할 수 있음을 의미합니다.

이러한 어려움은 작은 객체의 경우에 특히 두드러지며, 불명확한 경계나 특징을 가진 객체의 경우에는 더 큰 어려움이 있을 수 있습니다. 따라서, 향후 연구에서는 이러한 어려운 경우에 대한 정확한 라벨링 가이드라인을 마련하고 라벨러들 간의 합의를 도모하여 객체 경계 설정의 일관성을 확보하는 것이 중요합니다. 또한, 이미지 품질 개선이나 블러 효과 보정 알고리즘 등을 적용하여 작은 객체에 대한 라벨링 정확도를 향상시킬 수 있는 연구도 필요합니다. 이와 함게, 라벨링 작업 시 참고할 수 있는 추가적인 도구나 시각적인 지침을 개발하여 명확한 기준에 따라 객체를 분류하고 경계를 설정할 수 있도록 지원하는 것도 고려해야 합니다. 이를 통해 작은 객체에 대한 라벨링 작업의 일관성과 효율성을 향상시킬 수 있을 것입니다.

둘째로, ViT 모델은 patch size를 3으로만 적용한 3x3 그리드 챌린지에 대해서만 검증하였습니다. 4x4 그리드 챌린지에서의 성능에 대한 연구는 이전 논문에서 언급되었으나, 이번 연구에서는 해당 부분을 다루지 않았습니다. 따라서, 향후 연구에서는 4x4 그리드 챌린지에 대한 성능 분석을 수행하여 ViT 모델의 확장 가능성과 성능 변화를 확인하는 것이 중요합니다.

셋째로, GPU의 제한된 자원으로 인해 모델 학습 및 추론에 소요되는 시간이 증가하여 어려움을 겪었습니다. 보다 강력한 GPU 자원을 확보한다면 더욱 발전된 모델을 만들 수 있을 것입니다.

마지막으로, 실제 ReCAPTCHA 시스템에 적용하여 테스트해보는 것이 중요합니다. 이를 통해 모델의 실제 적용 가능성과 성능을 평가할 수 있습니다.

위의 연구 방향을 통해 ViT 모델을 더욱 발전시키고, 객체 검출 및 ReCAPTCHA 챌린지 해결과 같은 다양한 응용 분야에서의 활용 가능성을 확장할 수 있을 것입니다.

References

[1] [Image Recognition for Solving Google’s reCAPTCHA An Investigation of how Different Aspects Affects the Security of Google’s reCAPTCHA](https://kth.diva-portal.org/smash/get/diva2:1703299/FULLTEXT01.pdf)  (Stockholm, Sweden 2022) [논문]

[2] [Object Detection based Solver for Google's Image reCAPTCHA v2](https://arxiv.org/pdf/2104.03366.pdf) (2021.4) [논문]

[3] Kaggle Dataset : <https://www.kaggle.com/datasets/mikhailma/test-dataset> [데이터셋]

[4] [AN IMAGE IS WORTH 16X16 WORDS: TRANSFORMERS FOR IMAGE RECOGNITION AT SCALE](https://arxiv.org/pdf/2010.11929v2.pdf) (ICLR, 2021) [논문]