

AI 기반 다차원 CAPTHCA 시스템 설계

AI와 인간의 인지적 차이를 활용한 다차원적 인증 방식의 새로운 CAPTCHA 시스템 설계
최종 결과보고서



GitHub [djftk/captcha_vG](https://github.com/djftk/captcha_vG)

Notion <https://www.notion.so/P-14b7fa87b6d880b1900ed107454c2392>

1분반 4조

제출일	2024. 12. 18	전공	인공지능전공	인공지능전공	인공지능전공
과목	p-실무프로젝트	학번	202035501	202035510	202235002
담당교수	이상웅	이름	강승민	김유현	강지원

- 목 차 -

I . 1,2차 Report Feedback.....	p.3~4
II . CAPTCHA 시스템	p.4~20
1. CAPTCHA 정의 및 활용	
2. 현재 CAPTCHA의 문제점	
3. 새로운 CAPTCHA 검증 유형	
1) 시각 CAPTCHA	
1-1) 이미지 생성(DCGAN)	
1-2) 이미지 착시(FLIP illusion)	
2) 다차원 CAPTCHA(행동 + 시각 이미지)	
4. 새로운 CAPTCHA 시스템 구조	
III. 최종 목표.....	p.20-22
1. CAPTCHA 형태 확립	
2. 관리 콘솔 제공 준비 및 성능, 보안성 테스트 진행 후 보완	
3. Docker와 CAPTCHA Webpage 연동	
4. 자동화 및 배포 전 테스트 진행	
V. 참고문헌.....	p.12~15

I . 1, 2차 Report Feedback

Type	idea	Cons	Develop
Image Generation	About GAN	- low resolution & quality	“유추”할 수 있는 인간능력 다수의 저화질 이미지로 정답 찾기 가능
	FLIP illusion : 같은 이미지 (다른 방향)	- 사람마다 먼저 보이는 형상 상이	이미지에 대한 “방향”hint -> 보이는 모든 물체 select
Image Synthesis	Müller-Lyer Illusion : 실제길이 같지만 시각적으 로는 차이가 있는 두 화살표	- 정답률 50% (너무 높음)	부적합
	BASNet : image에 대한 mask 추출 후 2개의 이미지 합성	- 사람도 알아보기 어려운 형 상 도출	문제풀이 방식 고안 실패
	Hidden Overlays illusion : 특징 이미지 숨기기	- 이미지 복잡성 매우 높음 - 사용자 편의성 ↓	부적합

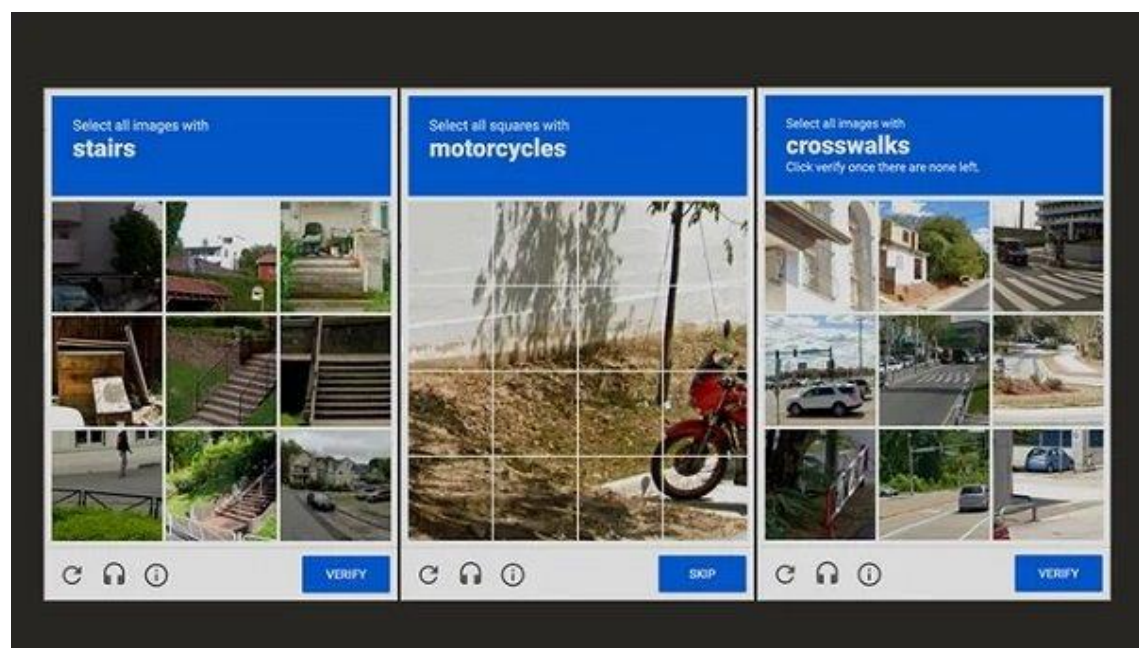
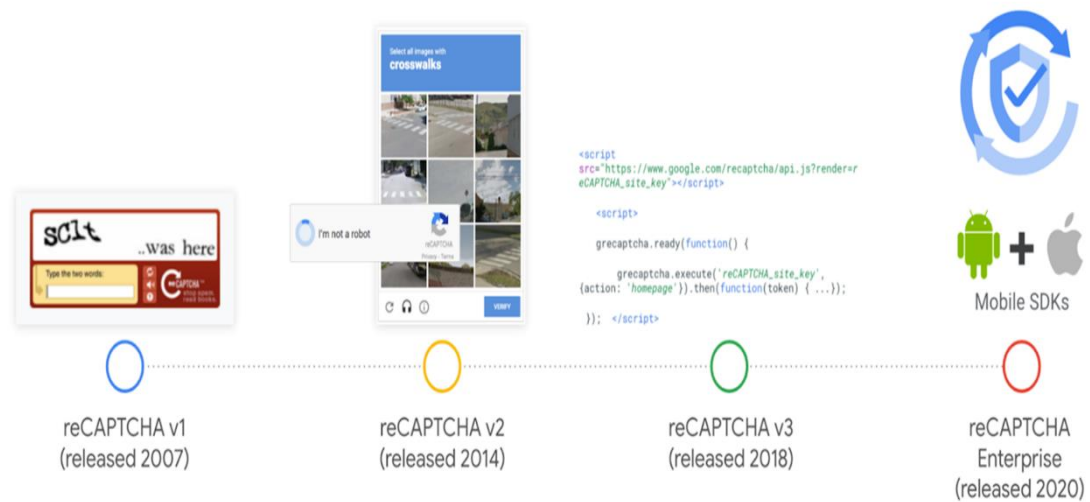
행동 기반 CAPTCHA	OpenCV : 이미지 내 물체의 윤곽선 추출 Mouse Detection : 마우스 클릭 시 움직인 위 치 파악	- 제공되는 이미지 내 물체 인식의 어려움 - 구체적인 정답 경로 설정의 어려움 - 사용자 편의성 ↓	부적합
---------------	--	--	-----

● 산학 멘토링 FeedBack 사항

- 현재 ‘reCAPTCHA’ 서비스의 불편함 해소 방안과 차별화 방안을 고민
- 어플리케이션 형태의 API를 제공하는 것 외에 관리 콘솔을 포함하여 다양한 형태의 API를 함께 제공

II . CAPTCHA 시스템

1. CAPTCHA의 정의 및 활용



목적:

- 사이트에 접속하는 봇 감지
- 서버 리소스 보호
- 불필요한 접근 및 요청에 관한 비용을 줄여준다
- 데이터 수집에 관한 정보 보호

2. 현재 CAPTCHA의 문제점

- An optimized system to solve text-based CAPTCHA

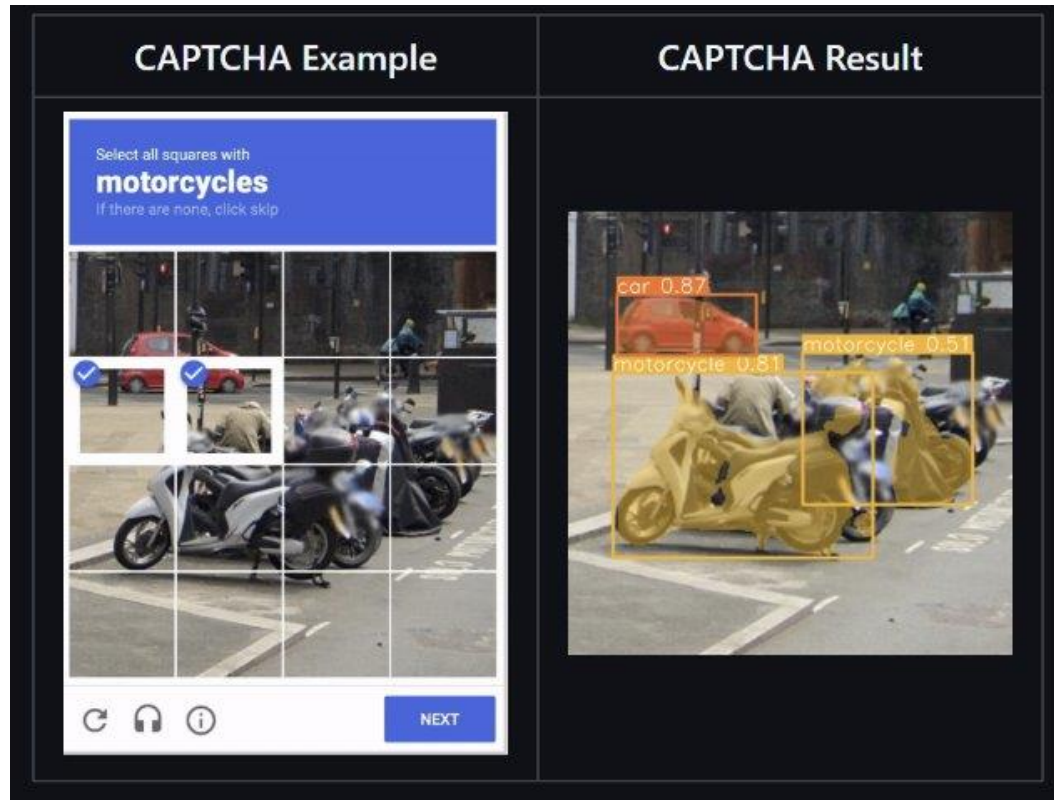
Original CAPTCHA	Denosing	Segmentation	Output
6 减 5 等于?	6 减 5	6 减 5	['outcome=', 1]
1 加 4 等于?	1 加 4	1 加 4	['outcome=', 5]
2 乘 1 等于?	2 乘 1	2 乘 1	['outcome=', 2]
8 乘 3 等于?	8 乘 3	8 乘 3	['outcome=', 24]
6 乘 6 等于?	6 乘 6	6 乘 6	['outcome=', 36]
9 乘 1 等于?	9 乘 1	9 乘 1	['outcome=', 9]
加 0 等于?	加 0	加 0	['outcome=', 4]
qE9tAR	qE9tAR		['outcome=', 'qE9tAR']
4tzfrm	4tzfrm		['outcome=', '4tzfrm']
4820	4820	Denosing	['outcome=', '4820']
3774	3774	Denosing	['outcome=', '3774']
玖 减 玖 等于?	玖 减 玖 等于?	玖 减 玖 等于?	['outcome = ', '0']
九 乘 八 等于?	九 乘 八 等于?	九 乘 八 等于?	['outcome = ', 72]
零 减 零 等于?	零 减 零 等于?	零 减 零 等于?	['outcome = ', -4]
五 加 六 等于?	五 加 六 等于?	五 加 六 等于?	['outcome = ', 11]

Fig. 11. The full process of defeating the CAPTCHA

➔ 텍스트 인식률이 75% -> 99%로 상승

➔ [An optimized system to solve text-based CAPTCHA](#)

- Yolov8을 이용한 reCAPTCHAv2 우회하기



- 마우스 움직임 분석

→ [DMTG: A Human-Like Mouse Trajectory Generation Bot Based on Entropy-Controlled Diffusion Networks](#)

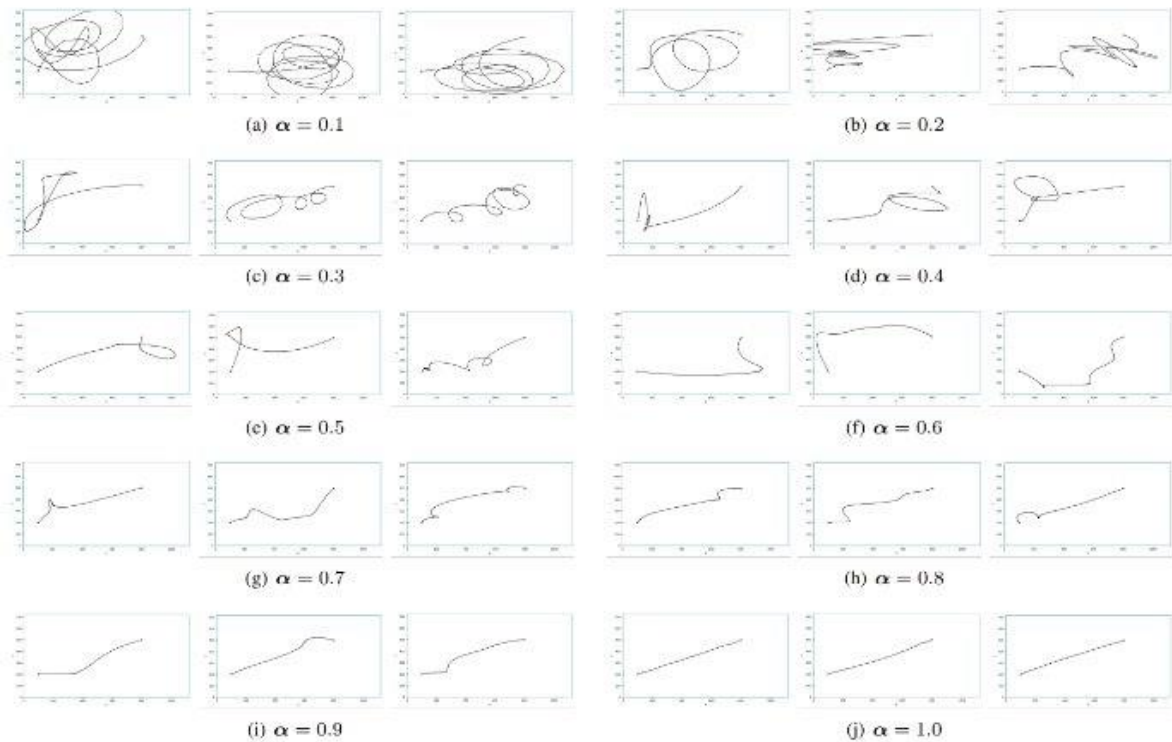


Figure 4. Examples of DMTG Curves under Different α Values in Eq. (10)

3. 새로운 CAPTCHA 검증 유형

A. 시각 기반 CAPTCHA

- 목표: CAPTCHA image를 수정하거나, 새롭게 만들어 Human은 인지할 수 있게, Computer는 인지하지 못하게 만든다.
- 실행 방법: CAPTCHA image 훈련데이터를 사용하여 Synthesize 데이터를 만든다

1) 이미지 생성 (DCGAN)

1-1) GANs

- 1.1.1. MLP 신경망을 거쳐 Generator는 생성을 하고, Discriminator는 Generator가 생성한 Image에 대한 판단을 한다.

```
class Generator(nn.Module):
    def __init__(self, noise_size, img_size, hidden_size1, hidden_size2):
        super(Generator, self).__init__()

        self.linear1 = nn.Linear(noise_size, hidden_size1)
        self.linear2 = nn.Linear(hidden_size1, hidden_size2)
        self.linear3 = nn.Linear(hidden_size2, img_size)
        self.relu = nn.ReLU()
        self.tanh = nn.Tanh()

    def forward(self, x):
        x = self.relu(self.linear1(x))
        x = self.relu(self.linear2(x))
        x = self.linear3(x)
        x = self.tanh(x)
        return x

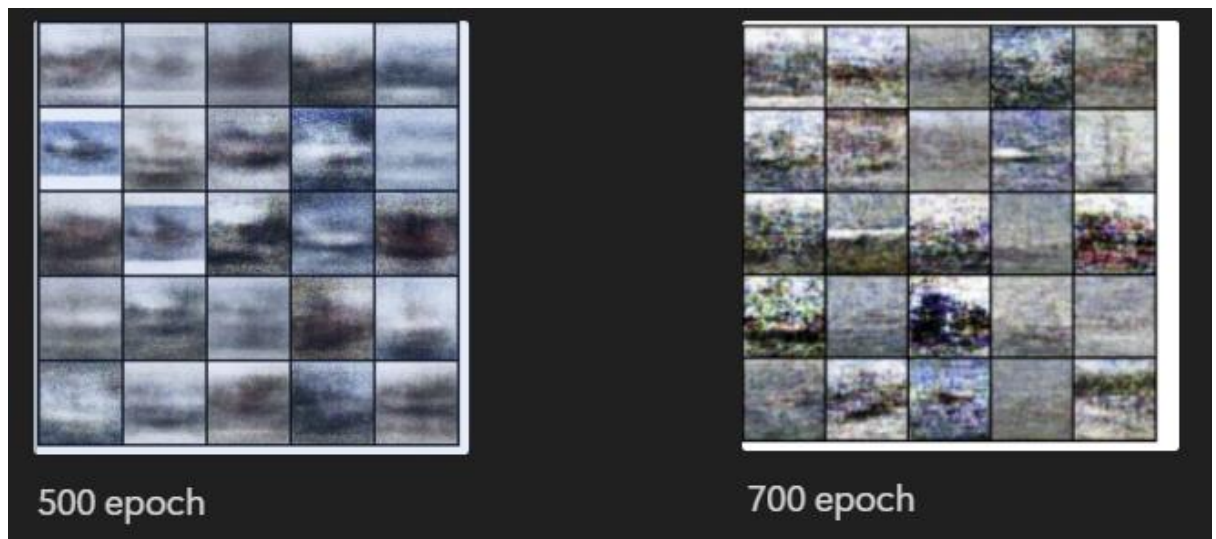
class Discriminator(nn.Module):
    def __init__(self, img_size, hidden_size1, hidden_size2):
        super(Discriminator, self).__init__()

        self.linear1 = nn.Linear(img_size, hidden_size1)
        self.linear2 = nn.Linear(hidden_size1, hidden_size2)
        self.linear3 = nn.Linear(hidden_size2, 1)
        self.leaky_relu = nn.LeakyReLU(0.2)
        self.sigmoid = nn.Sigmoid()

    def forward(self, x):
        x = self.leaky_relu(self.linear1(x))
        x = self.leaky_relu(self.linear2(x))
        x = self.linear3(x)
        x = self.sigmoid(x)
        return x
```

➔ Hyperparameter: {Noisy_dimension = 100, hidden_size1 = 256, hidden_size2 = 512, image_size = 64 * 64, learning_rate = 2e-4, Optimizer = adam}

1.1.2. Synthesize Image



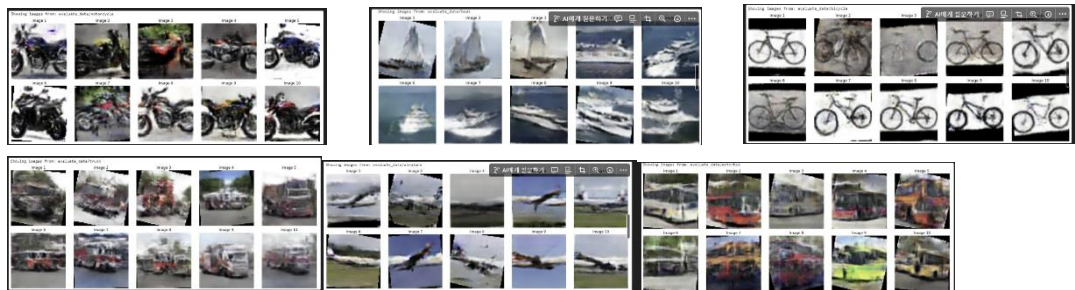
-> 결과 : GANs 모델은 사람도 구분하기 힘든 Image를 생성한다.

1-2) DCGAN

1.2.1. Deep Convolution Network를 GANs 적용한 DCGAN

- ➔ 논문에서 제공된 모델 Architecture, weight_init을 Generator에게만 적용, discriminator 적용X (image set resolution이 적기 때문에 선명하지 못하는 이유)
- ➔ Hyperparameter: {Noisy_dimension = 100, learning_rate = 2e-4, input_size = 64*64}
- ➔ 데이터 augmentation을 통해 데이터 수 증강
(RandomRotation, RandomHorizontalFlip, Normalize)

1.2.2. Synthesize Image



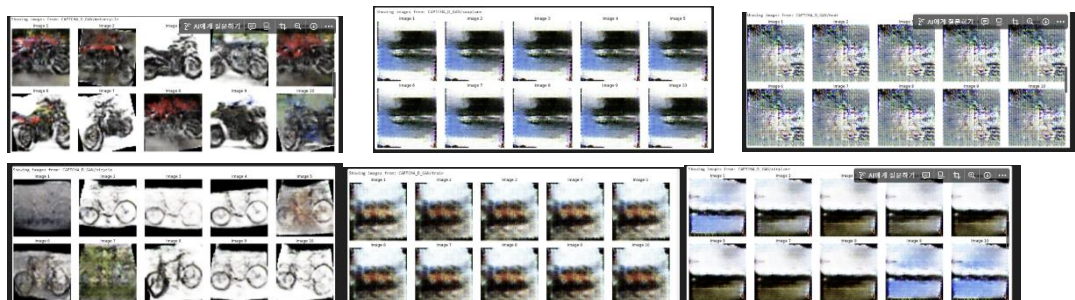
1-3) Distance - GANs

1.3.1. DCGAN architecture에서 Generator loss Function에 distance(C,z)를 추가함으로써 생성되는 image 모든 pixel값에 distance값이 더해진다.

$$\begin{aligned}
 \text{loss function}^* &= \text{loss function} + \text{Dist}_{z \sim p_z(z)}(C, z) \\
 &= -y \log \hat{y} - (1 - y) \log(1 - \hat{y}) + y \log \left(1 + \frac{1 - C}{C(1 - \hat{y})} \right).
 \end{aligned}
 \tag{3}$$

- ➔ Hyperparameter : {Noisy_dimension = 100, learning_rate = 2e-4, input_size = 64*64, distance_factor = 0.1, C = 1.2}

1.3.2. Synthesize image



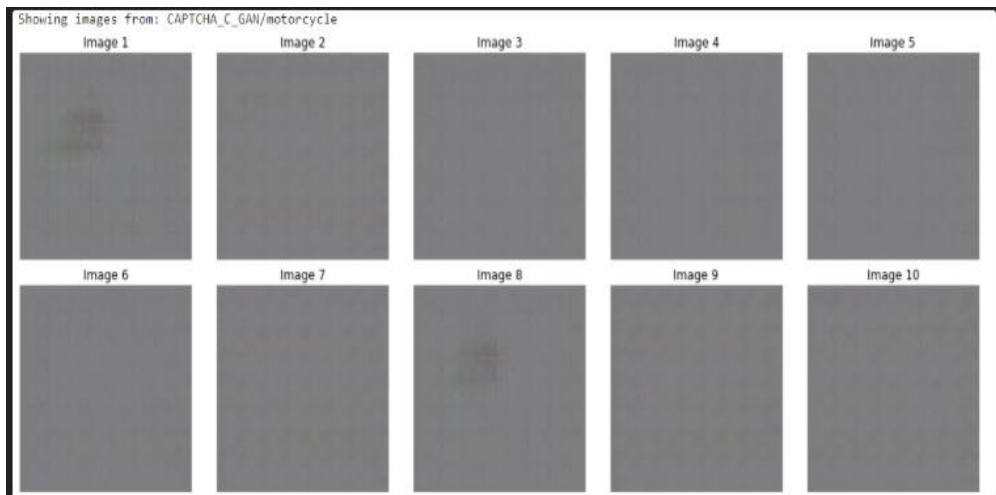
1-4) Composite - GANs

1.4.1. DCGAN으로 생성한 이미지와, batch내에 있는 original image를 결합한 데이터를 토대로 Generator, discriminator를 학습

```
composite_imgs = alpha * real_imgs + (1 - alpha) * gen_imgs
```

➔ Hyperparameter : {Noisy_dimension = 100, learning_rate = $2e-4$,
input_size = 64*64, alpha = 0.5}

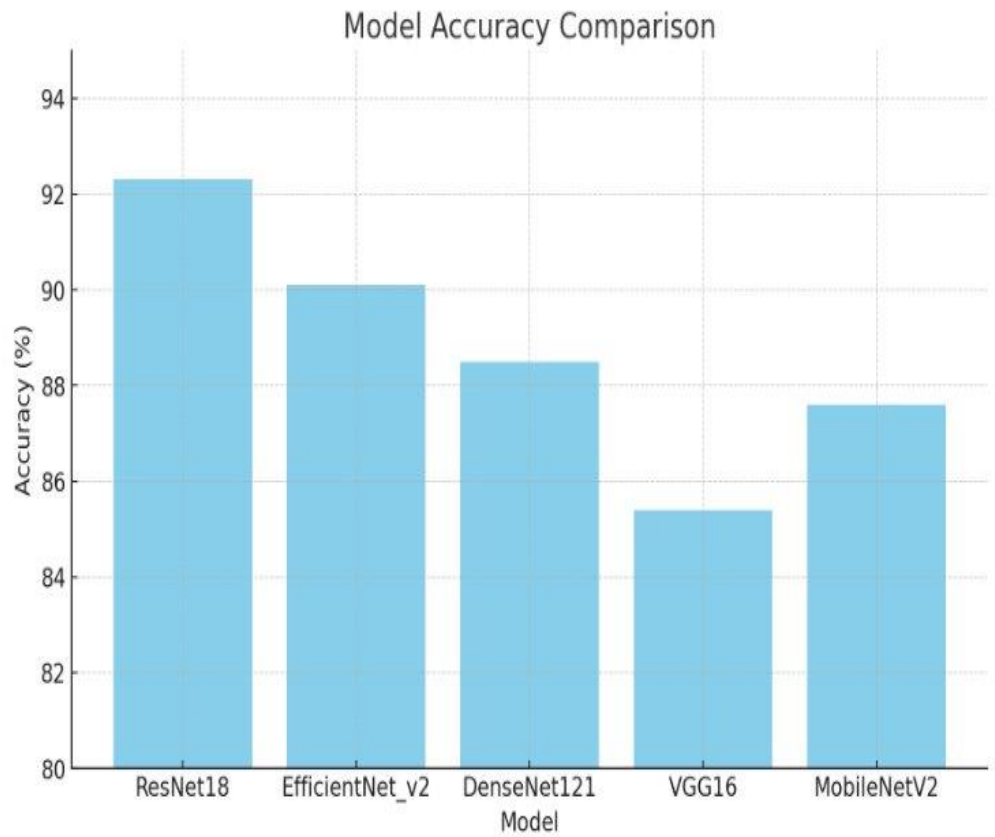
1.4.2. Synthesize Image



1-5. Synthesize or Augmented Data Evaluation

a. ResNet18 Fine-Tuning Evaluation

: Pre-trained ResNet18 모델에서 Fully Connected layer (weight, Bias)만 학습을 하고 나머지 부분은 Freeze한다. 새롭게 만들어 둔 데이터를 Confusion matrix, F1-score를 매겨 잘 학습된 Computer가 어느 정도 인지 할 수 있는지 파악한다. 128x128 이미지 데이터 셋에서 가장 훌륭한 정확도를 보여준 ResNet18로 검증 모델을 선정했다.



- GANs Evaluation 결과

Human: 생성된 image 208장 중에서 160장 분류 (80%)

Model: 생성된 image 208장 중에서 150장 분류 (~80%)

⇒ 따라서 GANs의 경우 시각 CAPTCHA Method로는 부적합하다.

- DCGANs Evaluation 결과

```

• Evaluation
self.label_map = {
    'motorcycle': 0, 'seaplane': 1, 'boat': 2, 'motorbus': 3,
    'bicycle': 4, 'train': 5, 'truck': 6, 'airplane': 7
}

```

Classification Report:				
	precision	recall	f1-score	support
Class 0	0.86	0.19	0.31	64
Class 1	0.34	0.66	0.45	32
Class 2	0.41	0.67	0.51	64
Class 3	0.38	0.38	0.38	32
Class 4	0.67	0.94	0.78	32
Class 5	0.09	0.03	0.05	32
Class 6	0.57	0.38	0.45	64
Class 7	0.45	0.56	0.50	32
accuracy			0.46	352
macro avg	0.47	0.47	0.43	352
weighted avg	0.51	0.46	0.43	352

➔ Human: 생성 image 352장 중에서 340장 분류(96%)

➔ 모델 f1-score = 43%

⇒ DCGAN을 사용하는 것이 시각 CAPTCHA Method로써 적합하다.

- Distance-GANs결과

Classification Report:				
	precision	recall	f1-score	support
Class 0	0.92	0.17	0.29	64
Class 1	0.29	0.59	0.39	32
Class 2	0.43	0.61	0.50	64
Class 3	0.36	0.56	0.44	32
Class 4	0.71	0.75	0.73	32
Class 5	0.50	0.03	0.06	32
Class 6	0.63	0.34	0.44	64
Class 7	0.29	0.56	0.38	32
accuracy			0.43	352
macro avg	0.51	0.45	0.40	352
weighted avg	0.55	0.43	0.41	352

Confusion Matrix:				
[[11 22 8 4 8 0 5 6]				
[0 19 11 1 0 0 0 1]				
[0 6 39 0 0 0 3 16]				
[0 4 8 18 0 0 1 1]				
[0 0 2 2 24 0 1 3]				
[0 3 12 5 0 1 1 10]				
[1 6 6 19 2 1 22 7]				
[0 6 5 1 0 0 2 18]]				

[Final] Test Loss: 2.0856, Accuracy: 43.1818%, F1-Score: 0.4064

➔ Human: 생성 image 352장 중에서 40장 분류(11%)

➔ Model f1 - score = 40%

⇒ Distance를 추가하는 것은 시각 CAPTCHA Method로써 적합하지 않다.

- Composite-GANs결과

Classification Report:				
	precision	recall	f1-score	support
Class 0	0.00	0.00	0.00	64
Class 1	0.00	0.00	0.00	32
Class 2	0.18	0.47	0.26	64
Class 3	0.00	0.00	0.00	32
Class 4	0.00	0.00	0.00	32
Class 5	0.08	0.03	0.05	32
Class 6	0.43	0.05	0.08	64
Class 7	0.03	0.06	0.04	32
accuracy			0.10	352
macro avg	0.09	0.08	0.05	352
weighted avg	0.12	0.10	0.07	352

Confusion Matrix:				
[[0 1 63 0 0 0 0 0]				
[0 0 7 0 12 0 0 13]				
[0 11 30 5 3 1 3 11]				
[12 8 0 0 0 0 0 12]				
[0 4 0 0 0 0 0 28]				
[3 0 5 0 19 1 0 4]				
[0 0 59 2 0 0 3 0]				
[0 2 3 10 4 10 1 2]]				

[Final] Test Loss: 3.4799, Accuracy: 10.2273%, F1-Score: 0.0703

➔ Human: 생성 image 352장 중 0장 분류(0%)

➔ Model f1 - marco: 0.07

⇒ Composite - DCGANs은 시각 CAPTCHA Method로써 적합하지 않다.

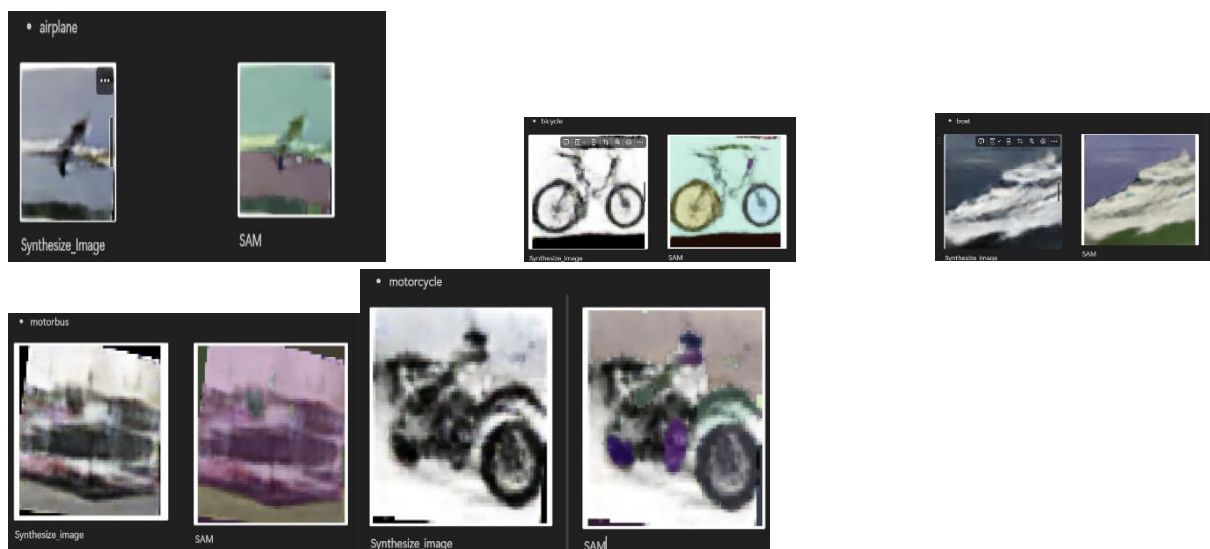
- Conclusion

About GAN	Human Classification Accuracy	Model F1-score	Suitability
GANs	80% (208 중 160)	About 80%	부적합
DCGAN	96% (352 중 340)	43%	적합
Distance - GANs	11% (352 중 40)	40%	부적합
Composite - GANs	0% (352 중 0)	0.07 (F1-macro)	부적합

b. Image Segmentation Model SAM

: Segmentation Anything Model을 사용하여 Synthesized Image, Noisy 적용된 Image를 SAM 모델을 통해 컴퓨터가 Segmentation 할 수 있는 정도를 시각화 한다.

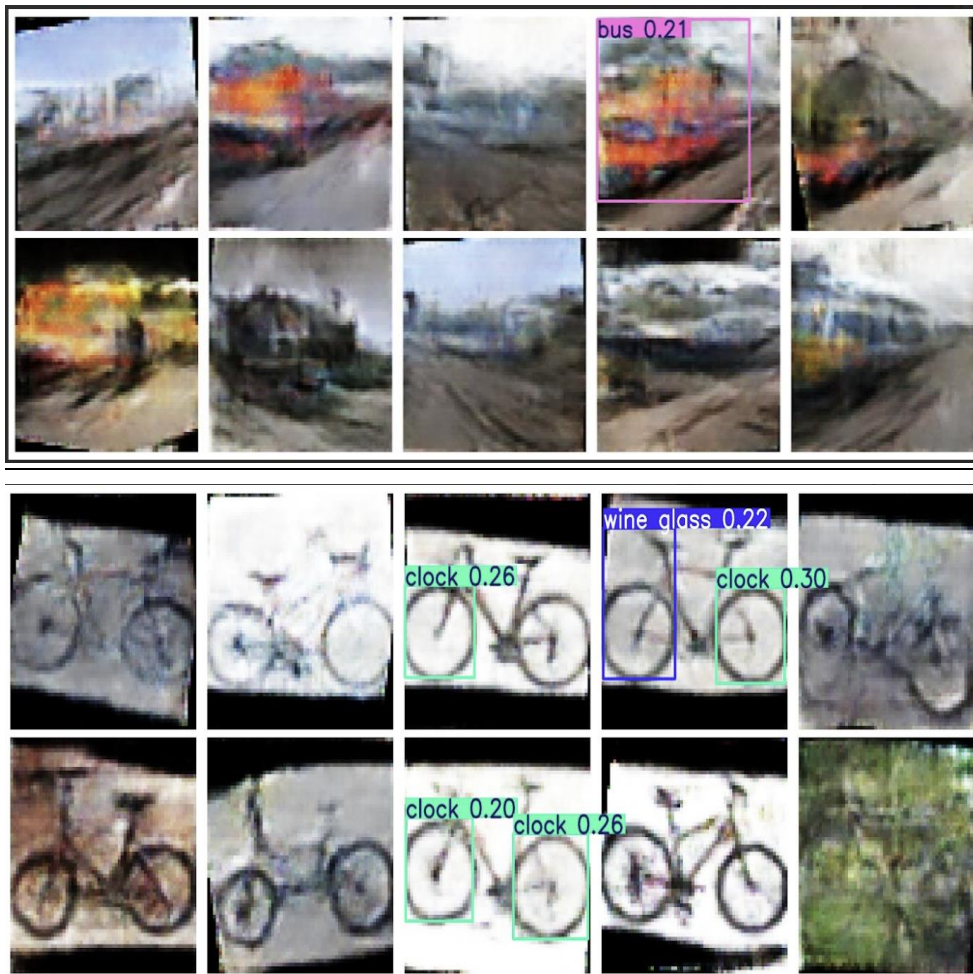
- DCGAN



c. Yolo


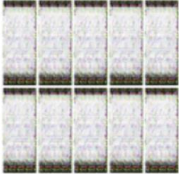
yolov11 model with coco dataset (pretraind.pt)

conf=0.2 # Confidence threshold



1-6. DCGAN 생성자, 판별자 weight init 방법

Weight Initialization	Image	Human Evaluate	Suitability
Random Init	<p>Image size (width, height): (4464, 1770) Resized image size (width, height): (128, 128)</p> 	인간이 이해하는 것 이해하지 못하는 이 미지가 모여있다.	보류

Normal Distribution initilaize	<p>Image size (width, height): (4464, 1770) Resized image size (width, height): (128, 128)</p> 	인간 이해	적합
He initialization	<p>o bicycle</p> <p>Image size (width, height): (4464, 1770) Resized image size (width, height): (128, 128)</p> 	인간 이해 불가능	부적합

Q1 : 이미지를 종합적으로 판단하고 아래에서 가장 유사한 이미지들을 고르세요

① 이미지 배치방식 5x2



② 하나의 batch로 이미지 파일 만들기

(사람은 직관적, 컴퓨터는 처음보는 데이터)




: 한 장의 이미지로 학습된 모델은 여러 개의 이미지가 모인 batch 이미지를 학습하지 못하였음

-> 하나의 batch로 이미지 파일을 만들어서 사람에게서는 쉬운 문제, 컴퓨터에게 학습되지 않은 데이터로 접근하도록 함.

-> Batch로 이미지 파일 만드는 것이 AI Model의 성능을 30% 정도 감소시킴

○ 하나의 이미지만 학습한 Fine-Tuning된 ResNet18 평가


- 하나의 이미지만 학습을 하고 하나의 이미지만 제공을 하였을 때 결과
 - 정확도 47%



Epochs: 100% 30/30 [0e:18<00:00, 4.56s/it]				
Classification Report:				
	precision	recall	f1-score	support
Class 0	1.00	0.28	0.44	64
Class 1	0.42	0.53	0.47	32
Class 2	0.38	0.73	0.50	64
Class 3	0.43	0.41	0.42	32
Class 4	0.70	0.91	0.75	32
Class 5	0.28	0.16	0.20	32
Class 6	0.65	0.34	0.45	64
Class 7	0.39	0.62	0.48	32
accuracy			0.48	352
macro avg	0.58	0.49	0.46	352
weighted avg	0.57	0.48	0.46	352

[Final] Test Loss: 2.1896, Accuracy: 47.727%, F1-score: 0.4639

- 하나의 이미지만 학습을 하고 batch로 묶인 이미지만 제공을 하였을 때 결과
 - 정확도 12%



Epochs: 100% 30/30 [13:46<00:00, 27.55s/it]				
Classification Report:				
	precision	recall	f1-score	support
Class 0	0.00	0.00	0.00	10
Class 1	0.00	0.00	0.00	10
Class 2	0.08	0.10	0.09	10
Class 3	0.20	0.20	0.20	10
Class 4	0.00	0.00	0.00	10
Class 5	0.13	0.70	0.23	10
Class 6	0.00	0.00	0.00	10
Class 7	0.00	0.00	0.00	10
accuracy			0.12	80
macro avg	0.05	0.12	0.06	80
weighted avg	0.05	0.12	0.06	80

[Final] Test Loss: 3.4857, Accuracy: 12.5000%, F1-score: 0.0546

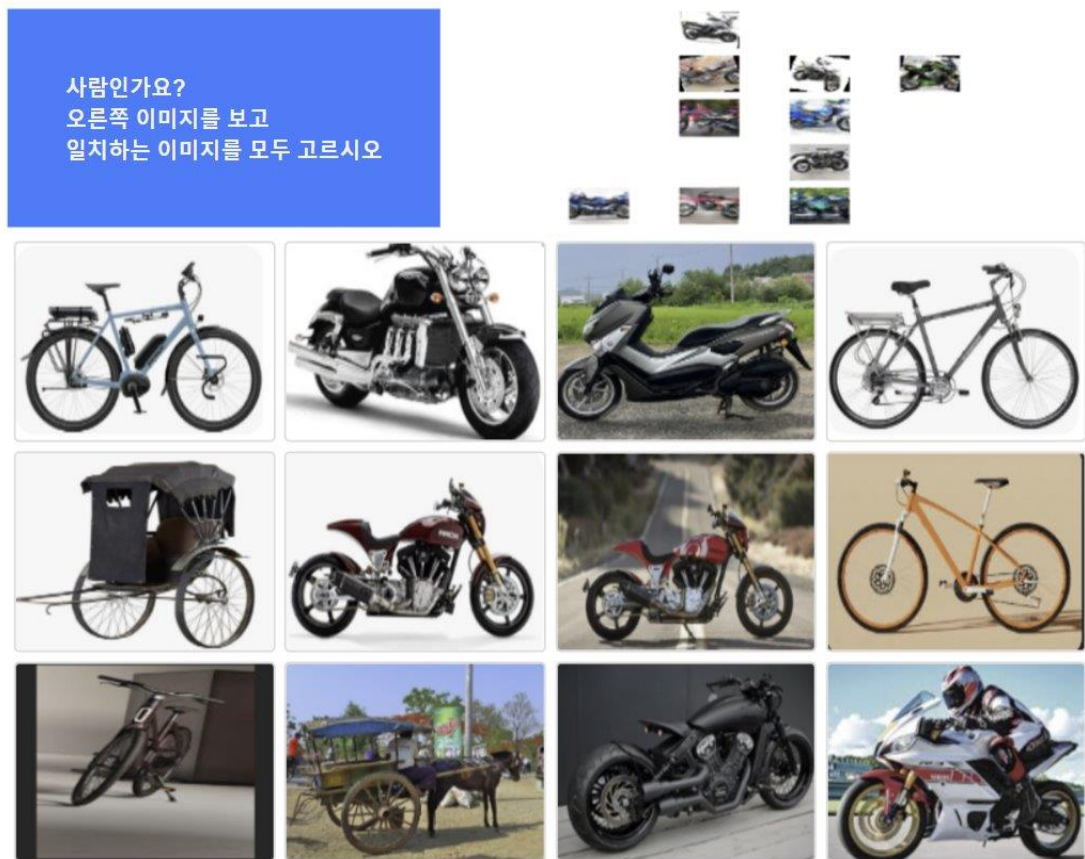
-> Batch로 학습을 하게 된다면 AI 모델의 성능이 올라가니 여러 batch모양을 만들기로 결정

- Matplotlib Gridspec 라이브러리를 이용하여 여러 Batch 이미지 파일 만들기

```
positions = [
    (0, 1), (0, 2), (0, 3), # (row, col)
    (1, 0),
    (2, 2),
    (3, 1), (3, 2),
    (4, 0), (4, 1), (4, 2)
]
```



- 시스템 적용 모습

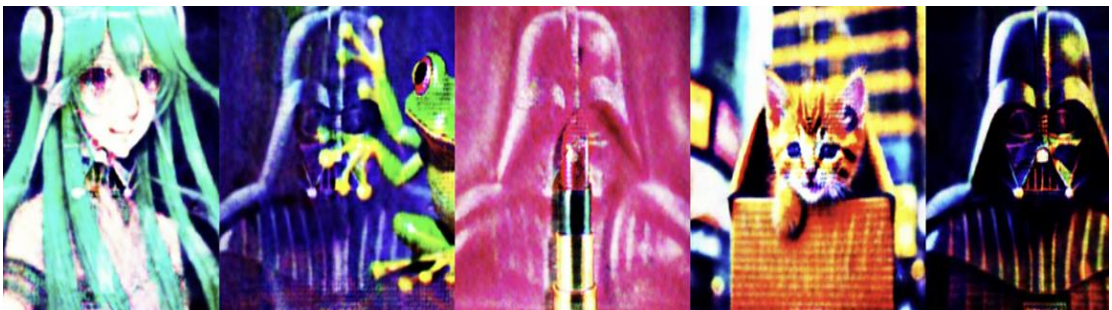


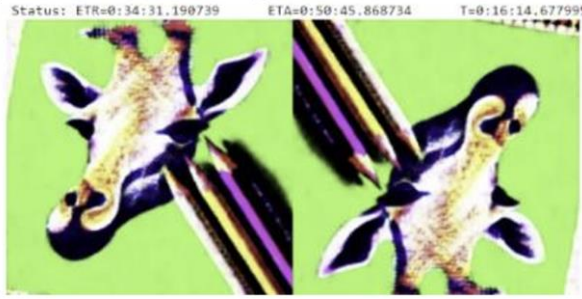
➔ 배치가 변경된 모습의 이미지가 상단부 문제 이미지로 주어진다.

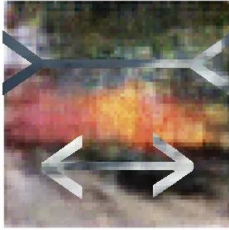


2) Illusion Diffusion & Stable Diffusion

2-1. 생성된 이미지

2-1.1. Hidden Character & Rotator

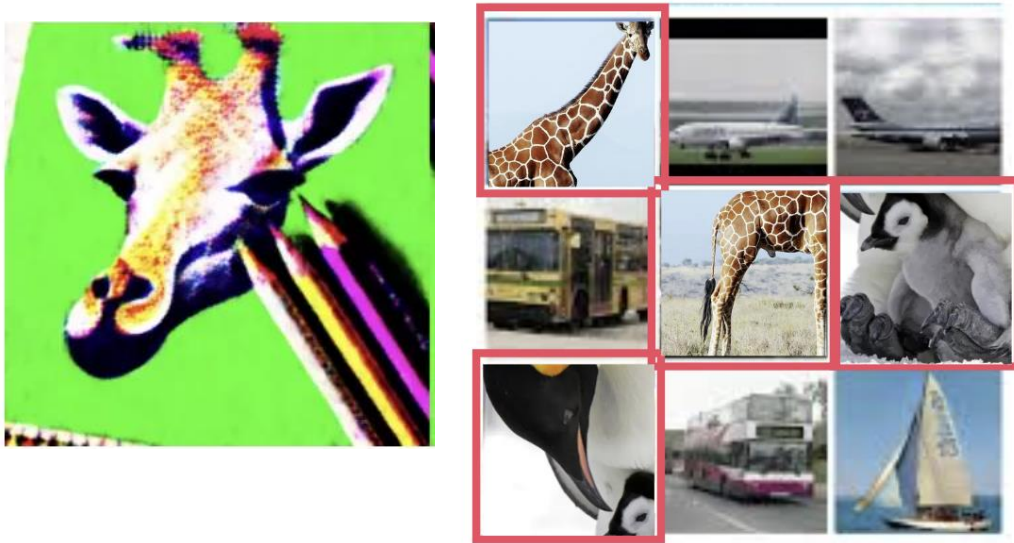


Stable Diffusion 모델	EX
FLIP illusion : 같은 이미지 (다른 방향)	

Stable Diffusion 모델	EX
Müller-Lyer Illusion : 실제길이 같지만 시각적으로 는 차이가 있는 두 화살표	
BASNet : image에 대한 mask 추출 후 2개의 이미지 합성	
Hidden Overlays illusion : 특징 이미지 숨기기	

2-2. 프롬프트 방식

Q1. 이미지에서 보이는 이미지를 모두 고르시오



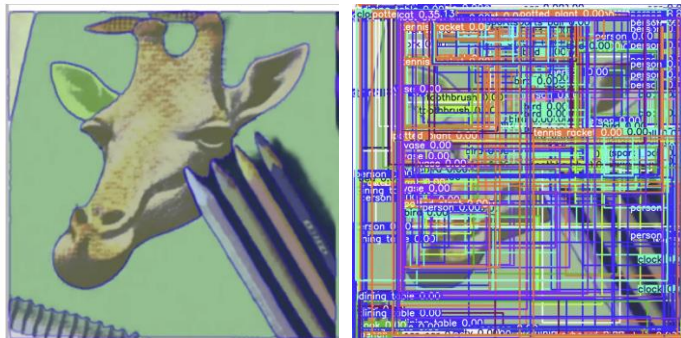
Q2. 이미지를 뒤집었을 때 어떤 이미지가 보이는가?

2-3. Evaluation Method

yolov11 model with coco dataset (pretraind.pt)

conf=0.00001

Confidence threshold



```
Image: /content/drive/MyDrive/Commit_File/chaptcha_vg/yolo/diffusion_images/giraff.png
- Class: cat, Confidences: 0.34654, 0.04339, 0.00082,
Average: 0.130
- Class: potted plant, Confidences: 0.08920, 0.00346, 0.00053, 0.00035, 0.00030, 0.00022, 0.00018,
Average: 0.008
- Class: person, Confidences: 0.12755, 0.00457, 0.00332, 0.00114, 0.00114, 0.00091, 0.00077,
Average: 0.002
- Class: scissors, Confidences: 0.00118, 0.00072, 0.00066, 0.00044, 0.00023, 0.00021, 0.00018,
Average: 0.000
- Class: tennis racket, Confidences: 0.00480, 0.00223, 0.00139, 0.00113, 0.00047, 0.00032, 0.00022,
Average: 0.000
- Class: tie, Confidences: 0.00040,
Average: 0.000
```

3-

4-

5-

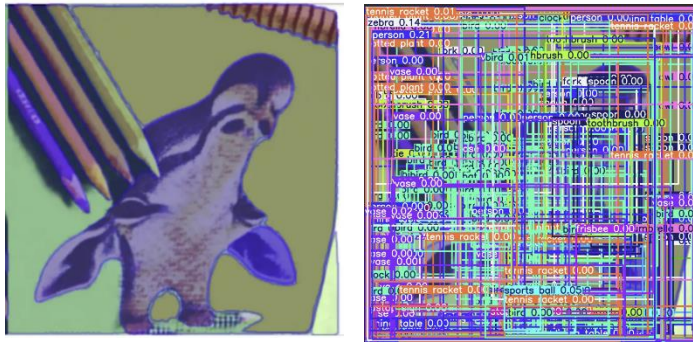
6-

7-

8-

cat	0.13
Potted plant	0.008
Person	0.002
Sicssors	0.000

9-



```

Image: /content/drive/MyDrive/Commit File/chaptcha_v6/yolo/diffusion_images/penguin.png
- Class: zebra, Confidences: 0.13553, 0.00339, 0.00079, 0.00014, 0.00014, 0.00010, 0.00006,
Average: 0.012
- Class: sports ball, Confidences: 0.05205, 0.00050, 0.00011, 0.00010, 0.00007, 0.00007, 0.00007,
Average: 0.007
- Class: person, Confidences: 0.20938, 0.00133, 0.00053, 0.00044, 0.00024, 0.00023, 0.00011,
Average: 0.006
- Class: frisbee, Confidences: 0.00359, 0.00004,
Average: 0.002
- Class: tennis racket, Confidences: 0.00805, 0.00277, 0.00086, 0.00072, 0.00030, 0.00021,
Average: 0.001
- Class: vase, Confidences: 0.01830, 0.00399, 0.00227, 0.00218, 0.00141, 0.00068, 0.00065,
Average: 0.001
- Class: stop sign, Confidences: 0.00065, 0.00052,
Average: 0.001

```

Zebra	0.012
Sports ball	0.007
Frisbee	0.002
Tennis	0.001
Vase	0.001
Stop sign	0.001

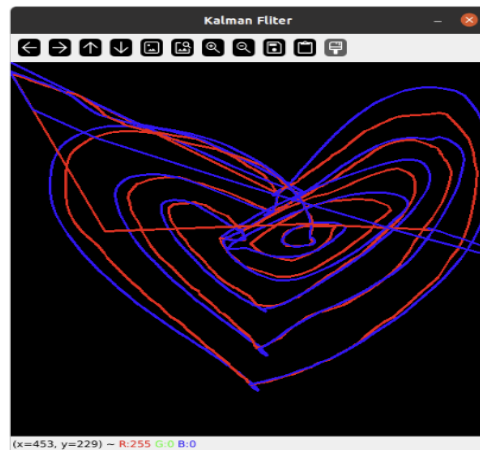
: 각각 이미지의 class가 giraffe와 bird로 구분되어야함 -> classification 못하는 것으로 판단됨

B. 다차원 CAPTCHA 시스템(시각 CAPTCHA + 행동)

1) 행동 추적

- 사용자 마우스 추적

- ➔ click Event + MouseEvent: 클릭 시점의 마우스 조작 이벤트와 클릭 시점의 좌표 정보 포함 -> 기준 위치에 대한 속성 보유
- ➔ OpenCV - KalmanFilter: 마우스 움직임에 대한 예측 및 추적
- ⇒ 실제 값과 노이즈가 포함된 측정값을 바탕으로 상태를 추정
- ⇒ 마우스 클릭을 측정된 좌표를 기반으로 칼만 필터를 사용해 추정값을 생성, 마우스 위치를 측정값으로 설정하고 칼만 필터를 통해 다음 좌표를 예측하며 측정값으로 갱신

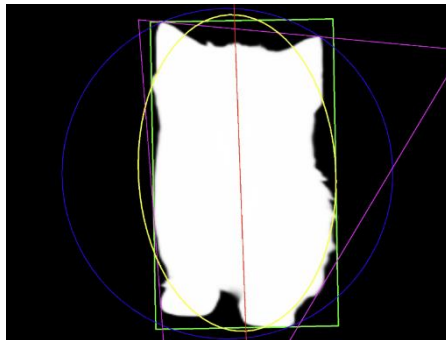


- 정답 경로 설정

: 다양한 방식으로 이미지를 생성하여 그 이미지 내 물체를 타겟한다. 타겟한 물체를 cv2 + Contour Detection와 같은 메서드로 객체의 윤곽선(원, 사각형, 중점, 중선)을 검출한 후 이를 기반으로 도형 생성 + 이미지 내 물체의 외곽선을 추출

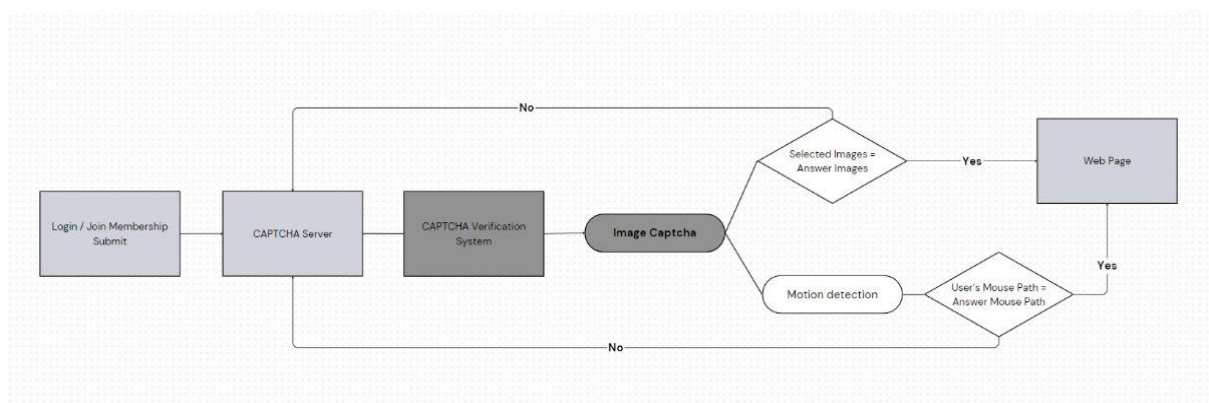
-> 그려진 도형내까지 정답 범위

-> 외곽선의 좌표를 정답 데이터로 설정하여 사용자가 따라 그렸을 때 일치 여부 검증

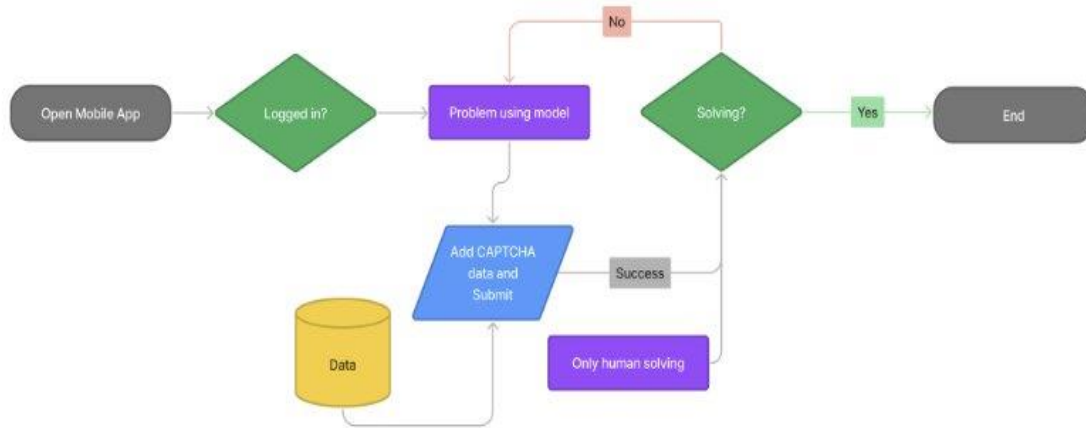


4. CAPTCHA 시스템 구조

1) Structure



- Web page test



- Deploy OpenAPI for developers

1 request API and site key



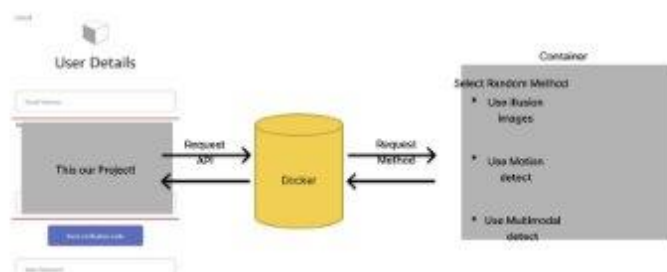
2 Docker Server POST API



3 Use Service



System Diagram



- Demonstration

➔ 백엔드

: 프론트엔드에서 CAPTCHA 이미지 및 사용자와 봇인지 여부 검증을 위해 12개의 이미지를 제공하고 정답 이미지를 클릭할 시 회원가입을 성공시키고 실패 시 이미지를 새로 고침 후 같은 과정을 반복한다.

(1) 상단 이미지 및 중앙 12개의 이미지 로드

: 상단부 이미지와 같은 이름의 디렉토리에서 정답/오답 이미지를 출력

- 이미지 데이터 셋(카테고리 및 이미지 수 추가로 확장 가능)

-> generated_captcha(상단 이미지): 10개의 디렉토리 x 20개 이미지

=> DCGANs: Airplane, Bicycle, Motorcycle, Seaplane, Motorbus, Train, Truck, Boat

=> Diffusion: Diffusion1, Diffusion2

(2) 데이터 셋 대분류 별로 정보 전달

: 대분류가 현재 Diffusion과 DCGAN인데 상단 CAPTCHA 이미지 생성 시 어떤 카테고리에서 왔는지 먼저 추출한다. 프론트엔드 코드 실행 시 카테고리에 관한 정보를 json형태로 전달

(3) 정답 데이터와 사용자 데이터 비교 검증

: 중앙의 12개의 이미지에는 각각의 주소를 백엔드에서 받아오는데 이 중 다른 부분을 제외하고 이름 부분만 추출하여 서버로 전달한다. 서버에서는 정답 이미지로 설정된 이미지의 이름과 사용자가 클릭한(전달받은) 이미지의 이름의 일치 여부를 검증

-> 일치 시: 가입 메시지 출력 이후 연결된 새로운 웹 페이지로 이동 + 사용자의 정보 '아이디.json' 파일 형태로 서버 저장

-> 미일치 시: 오류 메시지 출력 이후 새로운 이미지들 생성. 재가입 시도 시 새롭게 생성된 이미지 전달

➔ 프론트엔드

: 사용자에게 백엔드 서버에서 불러온 CAPTCHA 이미지를 표시하고 사용자로 하여금 이미지를 판단하게 한다. 판단 후 중단에 생성된 이미지 중 일치하는 이미지 또는 보이는 이미지를 모두 고른 뒤 결과를 백엔드에게 전달한다. 백엔드에서 설정한 메시지가 일치 여부에 따라 프론트엔드에서 출력

i. 회원가입 창 로드

회원가입

이름:

이름을 입력하세요

생년월일:

년 - 월 - 일

아이디:

아이디를 입력하세요

중복 체크

비밀번호:

비밀번호를 입력하세요

6자리 이상 입력하세요!

비밀번호 확인:

비밀번호 확인

E-mail:

E-mail을 입력하세요

☐ 개인정보 활용에 동의합니다.

회원가입

- (1) 개인정보를 입력한 뒤 ‘개인정보 활용에 동의합니다.’ 체크박스를 클릭하는 경우에만 회원가입 버튼 활성화
클릭하지 않을 시 회원가입 버튼 비활성화
- (2) 달력 기능 & 비밀번호 암호화 기능 & 서버의 아이디 중복 체크 기능 추가
- (3) 비밀번호를 6자리 이하로 입력 시 회원가입 버튼 비활성화
- (4) 비밀번호 칸과 비밀번호 확인 칸의 텍스트 미일치 시 회원가입 버튼 비활성화
- (5) 회원가입 버튼을 누를 시 CAPTCHA 이미지 로드 화면으로 넘어가며 검증 시작

ii. CAPTCHA 이미지 로드

<DCGAN>

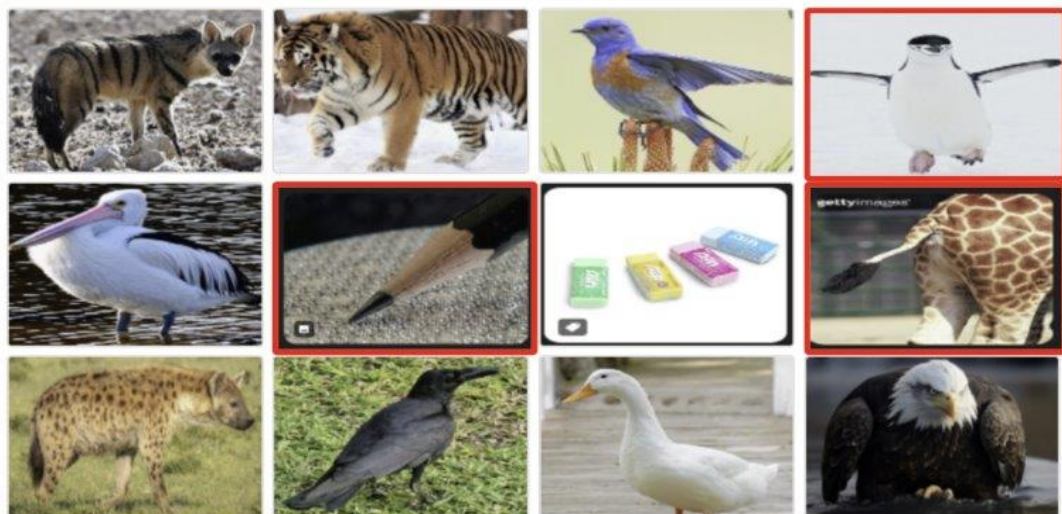
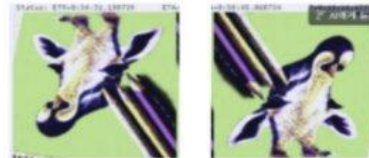
사람인가요?
오른쪽 이미지를 보고
일치하는 이미지를 모두 고르시오



사람입니다

<Diffusion>

사람인가요?
오른쪽 이미지에서
보이는 물체를 모두 고르시오



사람입니다

(1) 서버에서 제공하는 CAPTCHA 이미지와 중앙부 이미지의 로드

: 서버로부터 이미지들의 주소를 직접 불러와 이미지와 디렉토리 정보를 로드

-> 상단부에 출력되는 이미지에 따른 text의 다양화 (디렉토리의 다양화 -> text 확장)

=> DCGAN: 사람인가요? 오른쪽 이미지를 보고 일치하는 이미지를 모두 고르시오.

=> Diffusion: 사람인가요? 오른쪽 이미지에서 보이는 물체를 모두 고르시오.

-> 그리드의 다양화 가능: 4x3, 3x3..

=> 'generated_captcha.png': CAPTCHA 이미지 저장 - 직접 url로 받아온다

=> 'answer_captcha_{index}_correct.png': 정답 이미지 저장

=> 'answer_captcha_{index}-fake.png': 오답 이미지 저장

(2) 사용자가 클릭한 이미지에 대한 정보 제출

: 사용자가 클릭한 이미지가 캔버스에 로드 된 이미지 중 정답 이미지로 설정된 이미지와 정보 및 개수가 일치하는 경우 서버로 데이터 제출 가능 & 오답 이미지 클릭 시 정답 이미지와 불일치하기 때문에 CAPTCHA 재시도

- 사용자 정보 수집 및 데이터 화

- 검증 방법

: 클릭한 이미지에서 전달받은 주소 중 기타 항목들을 제외하고 이미지 이름만 추출 후 정답 표시 여부를 검증

=> selected_images = correctAnswer

=> len(selected_images) = len(correctAnswer)

(3) 서버로 데이터 제출 및 submit 버튼 생성

: 사용자가 클릭한 이미지와 정답으로 설정된 이미지 간의 데이터 간의 일치 여부를 검증하여 검증 결과를 반환 및 서버로 사용자의 정보 전달 여부 결정

- 선택 이미지 = 정답 이미지

-> 사용자의 정보는 데이터베이스로 이동하고 새로운 웹 페이지로 이동

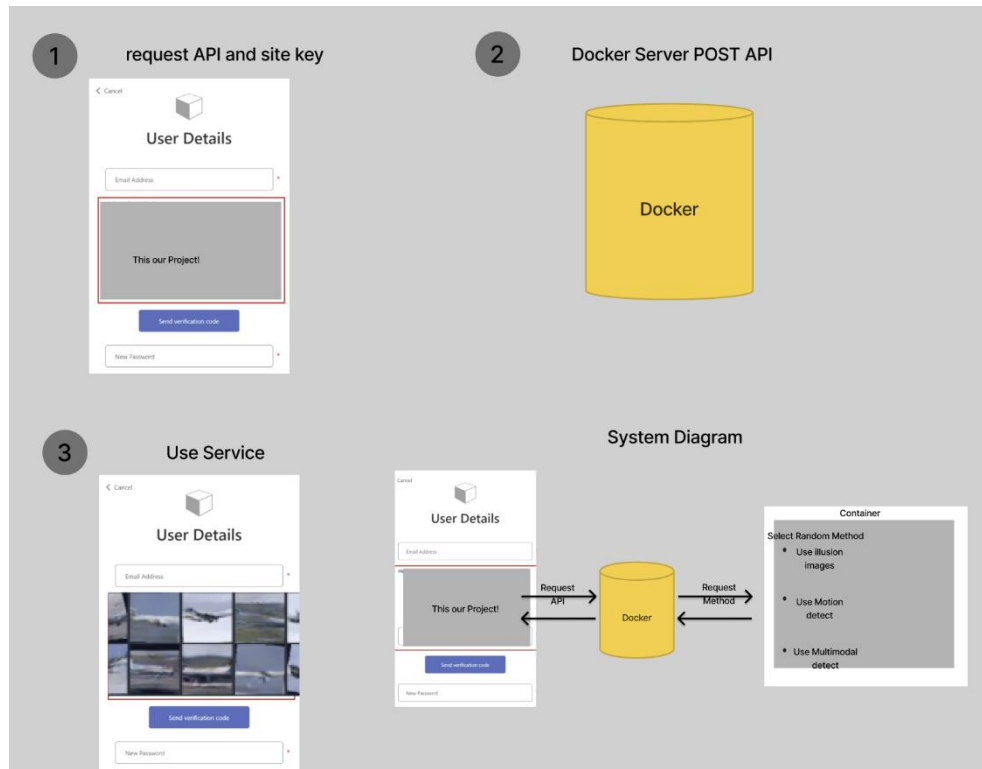
- 선택 이미지 != 정답 이미지

-> 사용자가 아닌 봇으로 판단하여 직전에 생성된 이미지와 선택 항목을 모두 삭제 후 새롭게 서버에서 이미지를 생성 후 재검증 절차 진행

III. 최종목표

1. OpenAPI

- 클라우드 플랫폼 활용해 API 배포(Google Cloud, AWS) - Docker를 활용해 API와 딥러닝 모델을 컨테이너로 관리
- TensorFlow Serving, TorchServe를 통해 AI 모델 배포
- HTTPS로 데이터 전송을 암호화 진행과 API 키 인증으로 접근 제어
- API 요청 및 결과를 기록함으로써 로그 관리

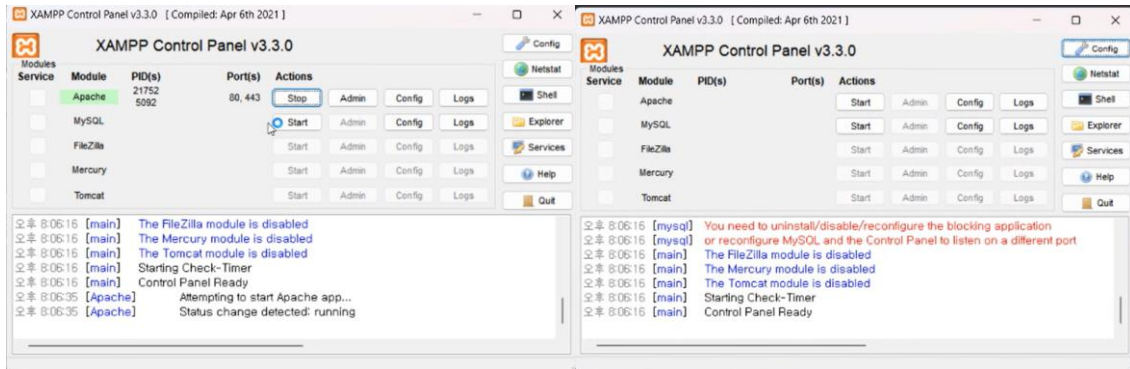


Example

google ReCAPTCHA openAPI활용 시도 (개발자)

(using) xampp control panel v3.3.0

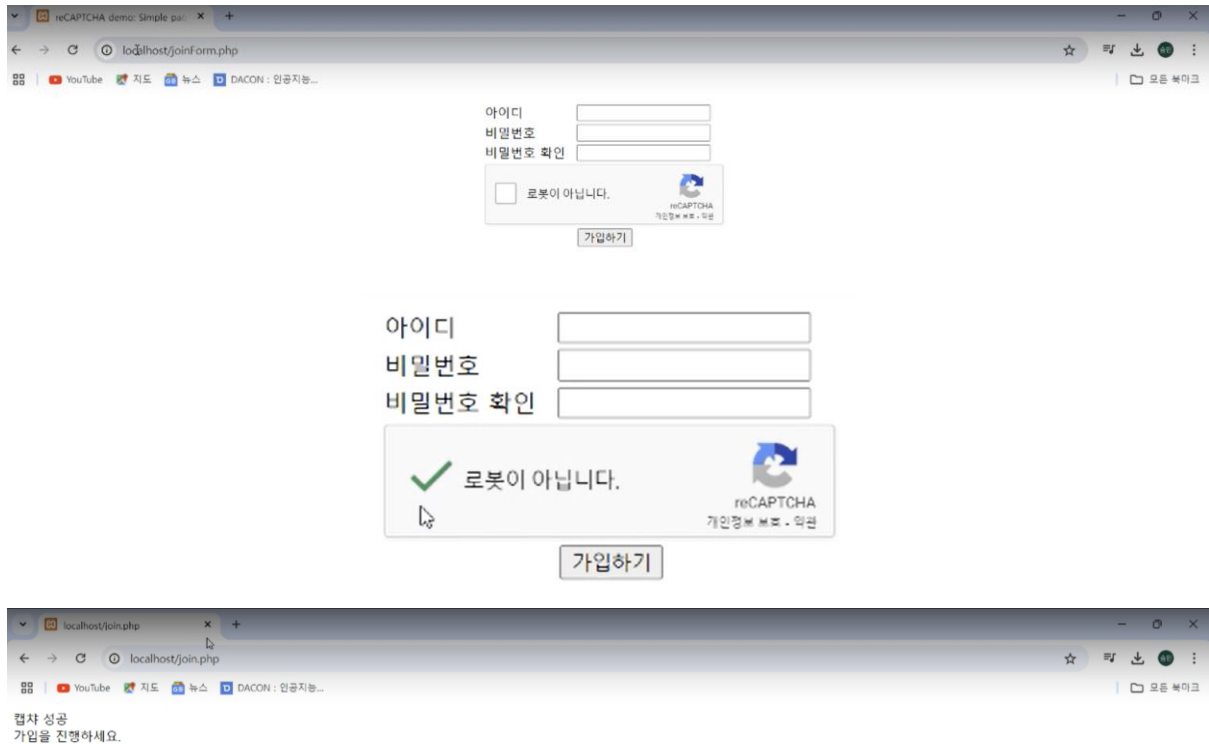
(웹 애플리케이션 개발 시 로컬 환경에서 실행하고 테스트하기 위한)



사용자(개발자)

```
<?php
$captcha = $_POST['g-recaptcha-response'];
$secretKey = ' ';
$ip = !empty($_SERVER['HTTP_CLIENT_IP']) ? $_SERVER['HTTP_CLIENT_IP'] :
(!empty($_SERVER['HTTP_X_FORWARDED_FOR']) ? $_SERVER['HTTP_X_FORWARDED_FOR'] : $_SERVER['REMOTE_ADDR']);
if (empty($captcha)) {
    echo '<script>
        alert("reCAPTCHA 를 완료해 주세요.");
        history.go(-1);
    </script>';
    exit;
}
// 데이터 준비
$data = array(
    'secret' => $secretKey,
    'response' => $captcha,
    'remoteip' => $ip
);
// cURL 로 요청
$ch = curl_init();
curl_setopt($ch, CURLOPT_URL, "https://www.google.com/recaptcha/api/siteverify");
curl_setopt($ch, CURLOPT_POST, true);
```

```
<body>
    <form action="join.php" method="post">
        <table style="margin:auto;">
            <tr>
                <td>아이디</td>
                <td><input type="text" name="id"></td>
            </tr>
            <tr>
                <td>비밀번호</td>
                <td><input type="password" name="pw"></td>
            </tr>
            <tr>
                <td>비밀번호 확인</td>
                <td><input type="password" name="pw2"></td>
            </tr>
            <tr>
                <td colspan="2">
                    <div id="captcha" class="g-recaptcha" data-sitekey=" " >
                    </div>
                </td>
            </tr>
        </table>
        <div style="text-align:center;">
            <button type="submit">가입하기</button>
        </div>
    </form>
</body>
</html>
```



1. 발전 방향

배경, 전경 컴퓨터가 구분을 잘 못하는 부분을 찾아 마우스 포인터로 그리기.

2. 관리 콘솔 제공 보안성 & 편의성 Test

- opensource (ex. <https://github.com/cracker0dks/CaptchaSolver>)

AI 모델을 활용하여 CAPTCHA 시스템 무력화 테스트

3. Docker와 CAPTCHA Webpage 연동

The image shows a login page for Gachon University. At the top is the Gachon University logo. Below it is the title '로그인' (Login). There are two input fields: '아이디' (ID) and '비밀번호' (Password). The ID field contains the text 'test'. The password field is masked with dots. At the bottom, there are two buttons: '로그인' (Login) and '회원가입' (Sign Up).

내 애플리케이션

애플리케이션 등록

API 재유 신청

계정 설정

애플리케이션 등록 (API 이용신청)

애플리케이션의 기본 정보를 등록하면 서버 메뉴가 만들어집니다.

애플리케이션 이름

* 40자 이하의 영문, 숫자, 공백을 입력 가능합니다.

사용 API

비로그인 오픈 API 서비스 환경

등록하기

취소

4. 자동화 및 배포 전 테스트 진행

참고문헌

<https://arxiv.org/abs/1511.06434>

Hyun KWON[†], Nonmember, Yongchul KIM^{††}, Member, Hyunsoo YOON[†], Nonmember, CAPTCH_Generation_image_by_using_GANs, VOL.E101-D, NO.2 FEBRUARY 2018

[파이토치 딥러닝] GAN으로 사람 얼굴 만들기

https://velog.io/@claude_ssim/%EA%B3%84%EC%82%B0%EC%82%AC%EC%A7%84%ED%95%99-Image-Blending#cut-and-paste

<https://github.com/xuebinqin/BASNet/tree/master>

<https://www.redinfo.co.kr/post/view/166>

<https://diffusionillusions.com/>

Ryan Burgert, Xiang Li, Diffusion Illusions: Hiding Images in Plain Sight, <https://arxiv.org/abs/2312.03817>

<https://docs.ultralytics.com/ko>

<https://sujakjil.tistory.com/132>

<https://console.cloud.google.com/security/recaptcha?authuser=0&inv=1&inv=AbkLlg&project=example-1734260365229>

[1511.06434] Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks