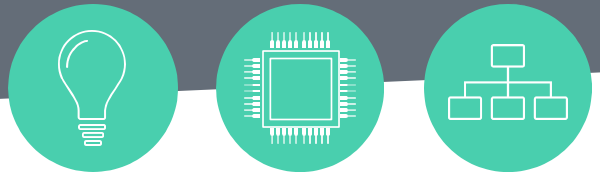
Get Me the Same Picture!



17 김찬영, 21 유지원

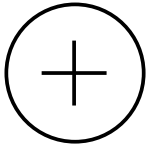
SMARCLE

Content

- What We Made
- Used Model
- Project Pipeline
- Code Review
- Demo

SMARCLE

What We Made

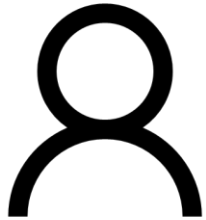


"AI를 활용한 Web Based Service"

SMARCLE

What We Made

INSERT



CLIP

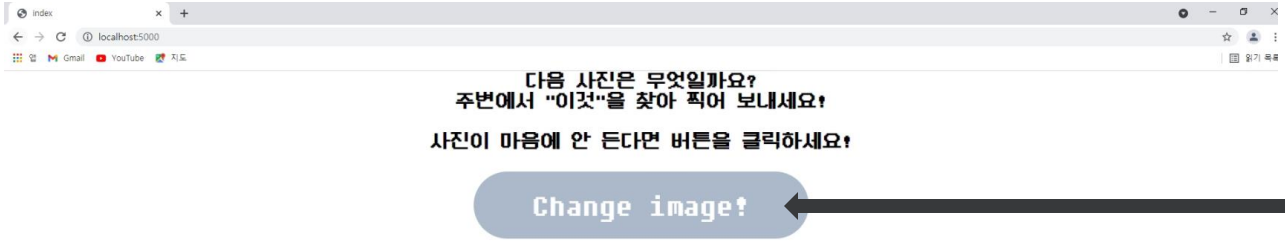
"Give me an image of the same kind as this"

Matches or Not

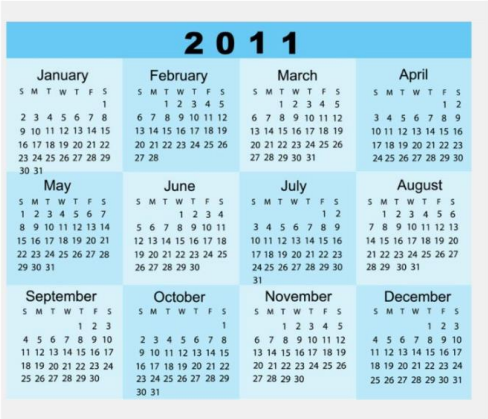
SMARCLE

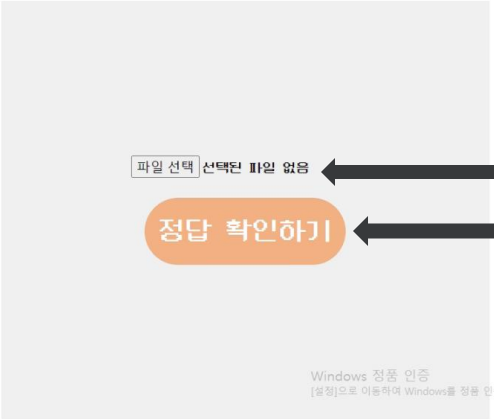
What We Made

1. Index page



Button 1





Button 2

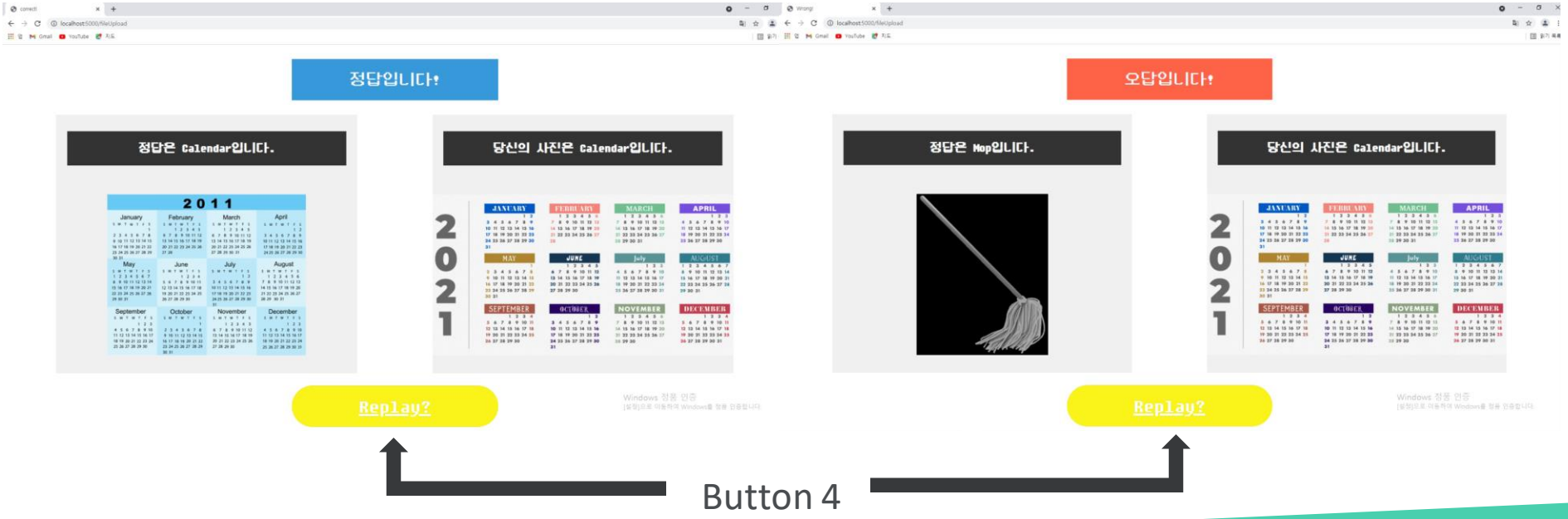
Button 3

SMARCLE

What We Made

2-1. Correct page

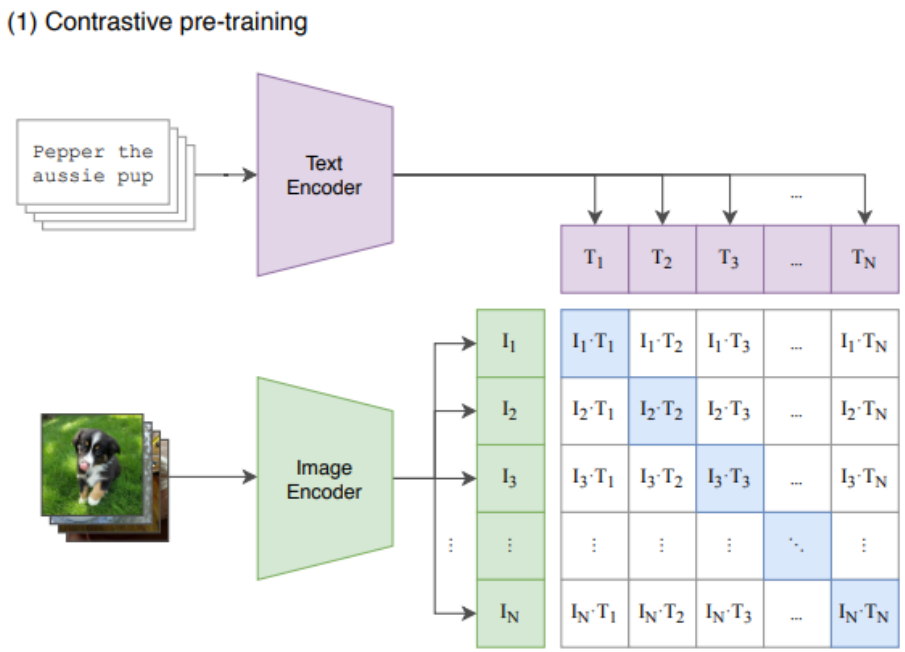
2-2. Wrong page



Button 4

SMARCLE

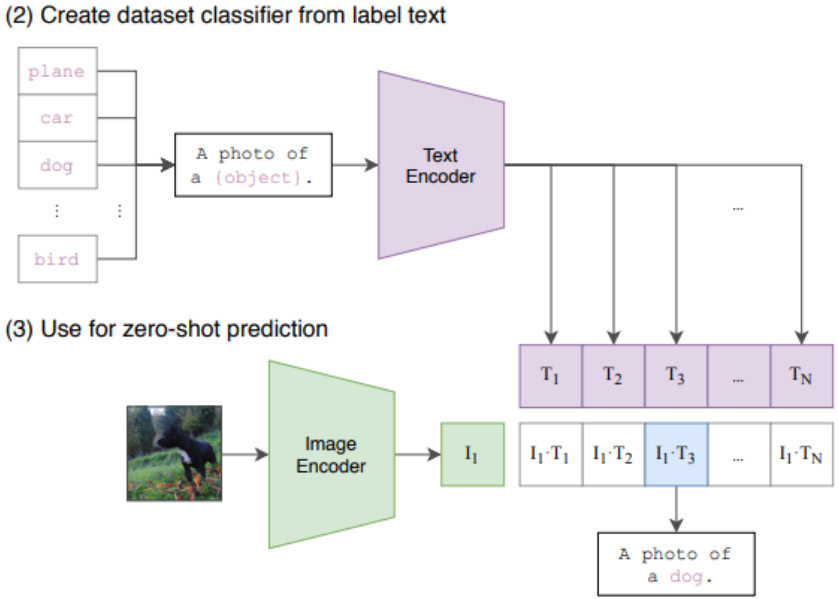
Used Model



OpenAI의 **CLIP**

- 4억장의 대용량 이미지 데이터
- Data Labeling 없이 웹크롤링을 통해 자동으로 이미지와 관련된 자연어 텍스트를 추출
- 주어진 이미지와 텍스트 사이의 올바른 연결 관계를 찾는 네트워크
  - Text Encoder : Transformer
  - Image Encoder : ViT, ResNet

Used Model

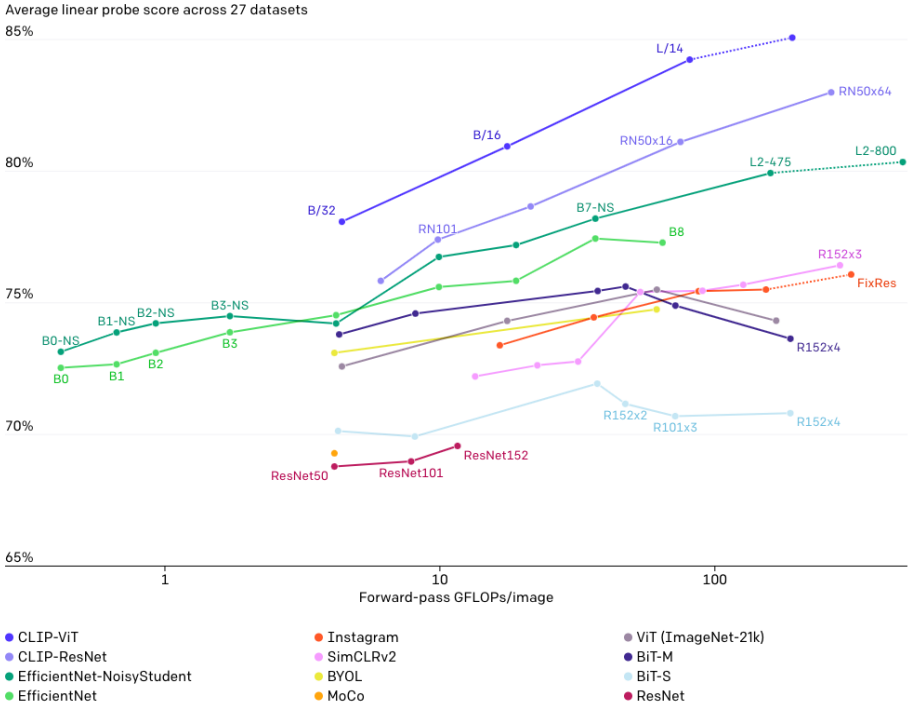


Zero-Shot Classifier

- Image의 Feature를 추출하고 주어진 Label을 Text Encoder에 통과시켜 Text Feature를 추출
- Text Feature 중 Image Feature과 가장 높은 상관관계를 갖는 텍스트를 이미지 분류 결과로 출력

SMARCLE

Used Model



- 기존 CNN을 사용한 SoTA급 Image Classification 모델보다 CLIP의 ViT 모델이 더 좋은 성능을 보이고 있음

SMARCLE

Used Model

```
ViT-B/32 Accurcy: 0.83
inference time : 0.2632873058319092 seconds
```

- Vision Transfomer B/32 Encoder Model 실험 결과

```
ResNet50 Accurcy: 0.79
inference time : 0.5844781398773193 seconds
```

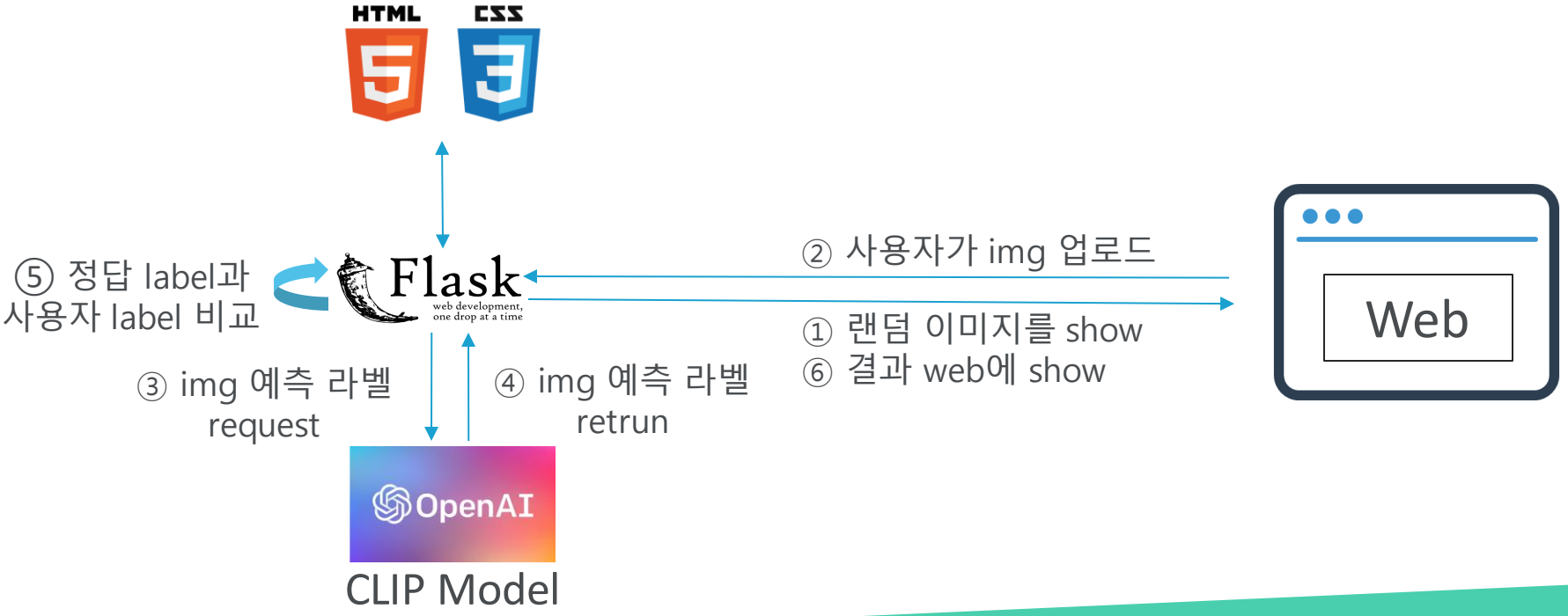
- ResNet50 Encoder Model 실험 결과

Model 실험 결과

- Office-Home Dataset의 4357장의 이미지와 Label을 input으로 가져가고 실험함
- CPU i5-10400F, RAM 16GB, GPU GTX1070

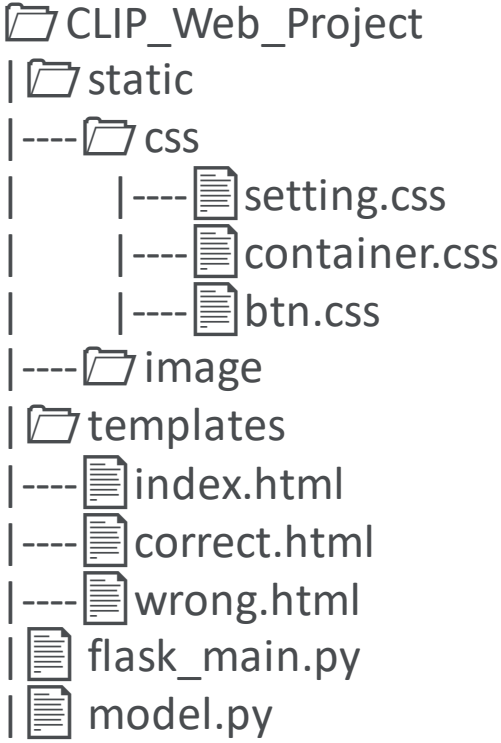
SMARCLE

Project Pipeline



SMARCLE

Code Review



SMARCLE

Code Review

flask_main.py

```python
from flask import Flask, render_template, request, url_for
from model import clip_class
from werkzeug.utils import secure_filename

app = Flask(__name__)

@app.route("/")
def index():
    global g_gt, g_cut_rfile, g_rfile
    gt, rfile = clip_class.get_ramdom_path()
    g_rfile = rfile
    rfile_cut = rfile[9:]
    g_gt = gt
    g_cut_rfile = rfile_cut
    return render_template("index.html",  image_file=g_cut_rfile)


@app.route('/fileUpload', methods = ['GET', 'POST'])
def upload_file():
    if request.method == 'POST':
        f = request.files['file']
        f.save('./static/image/uploads/' + secure_filename(f.filename))
        upload = 'image/uploads/'+secure_filename(f.filename)
        submit_path = "./static/" + upload

        given_answer, _ = clip_class.clip_predict(submit_path)

        if g_gt == given_answer:
            print(upload)
            return render_template('correct.html', value = g_gt, image_file=g_cut_rfile, predict = upload,
predict_class = given_answer)
        else:
            print(upload)
            return render_template('wrong.html', value = g_gt, image_file=g_cut_rfile, predict = upload,
predict_class = given_answer)


if __name__ == '__main__':
    app.run(debug=True)
```

Main Page

- Clip_class의 get_random_path를 사용해 랜덤으로 불러올 이미지를 생성
- Index.html에 랜덤으로 불러온 사진을 plot
- 사진의 경로와 Ground Truth값을 전역변수로 선언
- Index.html에서는 사용자에게 이미지를 받습니다

SMARCLE

Code Review

flask_main.py

```python
from flask import Flask, render_template, request, url_for
from model import clip_class
from werkzeug.utils import secure_filename

app = Flask(__name__)

@app.route("/")
def index():
    global g_gt, g_cut_rfile, g_rfile
    gt, rfile = clip_class.get_ramdom_path()
    g_rfile = rfile
    rfile_cut = rfile[9:]
    g_gt = gt
    g_cut_rfile = rfile_cut
    return render_template("index.html",  image_file=g_cut_rfile)

@app.route('/fileUpload', methods = ['GET', 'POST'])
def upload_file():
    if request.method == 'POST':
        f = request.files['file']
        f.save('./static/image/uploads/' + secure_filename(f.filename))
        upload = 'image/uploads/'+secure_filename(f.filename)
        submit_path = "./static/" + upload

        given_answer, _ = clip_class.clip_predict(submit_path)

        if g_gt == given_answer:
            print(upload)
            return render_template('correct.html', value = g_gt, image_file=g_cut_rfile, predict = upload,
predict_class = given_answer)
        else:
            print(upload)
            return render_template('wrong.html', value = g_gt, image_file=g_cut_rfile, predict = upload,
predict_class = given_answer)

if __name__ == '__main__':
    app.run(debug=True)
```

Result Page

- 사용자에게 받은 이미지를 CLIP 모델에 넣어 결과를 받음
- 정답 결과에 따라 correct.html 또는 wrong.html을 반환

SMARCLE

Code Review

model.py

```python
import torch
import clip
from PIL import Image
import numpy as np
import cv2
import glob, random
import imutils, os, time

class clip_class:

    def get_ramdom_path():
        file_path = './static/image/Real World/'

        first_path = os.listdir(file_path)
        class_path = random.choice(first_path)
        rfile = os.path.join(file_path, class_path)
        img_list = os.listdir(rfile)
        sel_img = random.choice(img_list)
        rfile= rfile+'/'+ sel_img
        return class_path, rfile
```

CLIP Model

- clip_class: 랜덤 이미지를 불러오는 함수와 모델 실행 함수 두개로 구성
- get_random_path: 데이터셋을 탐색하며 랜덤 이미지의 경로와 True Label 값을 반환

Code Review

model.py

```python
def clip_predict(rfile):
    device = "cuda" if torch.cuda.is_available() else "cpu"

    model_use = "ViT-B/32"
    # model_use = "RN50x16" # ViT-B/32, RN50x16
    model, preprocess = clip.load(model_use, device=device)
    image = preprocess(Image.open(rfile)).unsqueeze(0).to(device)

    class_list = ['Alarm_Clock', 'Backpack', 'Batteries', 'Bed', 'Bike', 'Bottle', 'Bucket',
'Calculator', 'Calendar', 'Candles', 'Chair', 'Clipboards', 'Computer', 'Couch', 'Curtains', 'Desk_Lamp',
'Drill', 'Eraser', 'Exit_Sign', 'Fan', 'File_Cabinet', 'Flipflops', 'Flowers', 'Folder', 'Fork', 'Glasses',
'Hammer', 'Helmet', 'Kettle', 'Keyboard', 'Knives', 'Lamp_Shade', 'Laptop', 'Marker', 'Monitor', 'Mop',
'Mouse', 'Mug', 'Notebook', 'Oven', 'Pan', 'Paper_Clip', 'Pen', 'Pencil', 'Postit_Notes', 'Printer',
'Push_Pin', 'Radio', 'Refrigerator', 'Ruler', 'Scissors', 'Screwdriver', 'Shelf', 'Sink', 'Sneakers', 'Soda',
'Speaker', 'Spoon', 'Table', 'Telephone', 'ToothBrush', 'Toys', 'Trash_Can', 'TV', 'Webcam']

    start_time = time.time()
    text = clip.tokenize(class_list).to(device)

    with torch.no_grad():
        image_features = model.encode_image(image)
        text_features = model.encode_text(text)

        logits_per_image, logits_per_text = model(image, text)
        probs = np.array(logits_per_image.softmax(dim=-1).cpu().numpy())

    label = np.argmax(probs[0])
    end_time = time.time()
    print(probs[0][label])
    print("inference time :", end_time-start_time,"seconds")

    return class_list[label], rfile
```

CLIP Model

- clip_predict: get_random_path에서 받은 이미지에 대한 CLIP 모델의 예측값을 반환
- Pretrained ViT 모델을 불러옴
- Class list를 Office-Home Dataset의 class로 선언
- 모델 연산 후 예측 Label을 반환

Demo

- 동아리방 PC에 가상환경을 구축해 놓았음
- 해당 PC를 원격조정하여 프로그램 시연

감사합니다

SMARCLE