AI로봇융합심화(PBL)

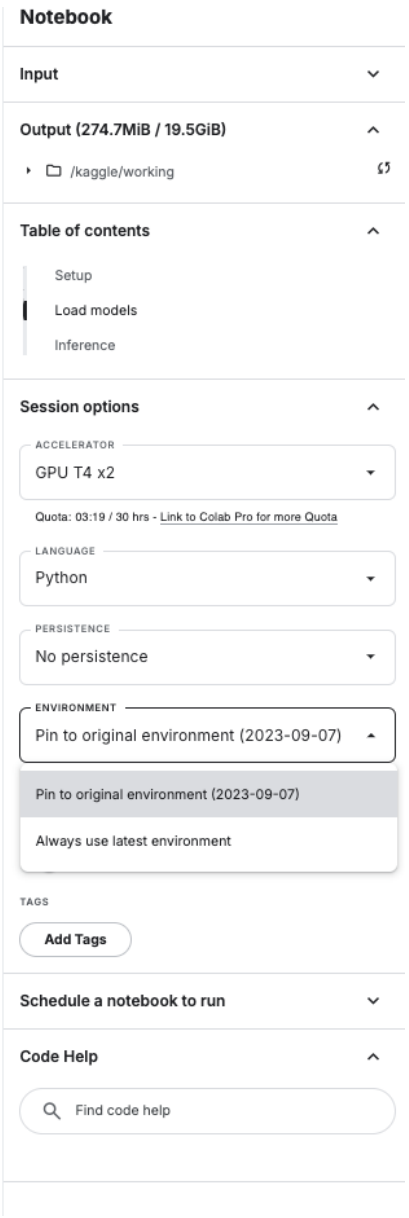
농업 로봇 모듈 편 (2025)

Zero-shot Learning in Computer Vision

- Zero-shot Image Classification: CLIP
  - Contrastive Language-Image Pre-training, NeurIPS 2022
    - [실습-CLIP] https://www.kaggle.com/code/yukyungchoi/2025-airobotpbl-w13-ex1/
    - [실습-CLIP] https://www.kaggle.com/code/yukyungchoi/2025-airobotpbl-w13-ex1-2/
- Zero-shot Object Detection: Grounding DINO
  - Grounding DINO: Marrying DINO with Grounded Pre-Training for Open-Set Object Detection, ECCV 2024
    - [실습-Grounding DINO] https://www.kaggle.com/code/yukyungchoi/2025-airobotpbl-w13-ex3/
- Zoro-shot Image Segmentation : Grounded SAM
  - Grounded SAM: Assembling Open-World Models for Diverse Visual Tasks, Arxiv 2024.
    - [실습-SAM] https://www.kaggle.com/code/yukyungchoi/2025-airobotpbl-w13-ex4-1/
    - [실습-SAM] https://www.kaggle.com/code/yukyungchoi/2025-airobotpbl-w13-ex4-2/

Zero-shot Learning in Computer Vision

- Kaggle Notebook의 경우 Dockerfile로 관리되고 있음
- 반드시 실습 코드를 복사하여 사용할 것
- 동일한 코드라 하더라도, 환경 파일 내 정보가 중요함
  - 환경설정이 다르면 동일 코드라 하더라도 동작하지 않음



Zero-shot Classification

- **CLIP 실습 (1) : Simple 테스트**
  - https://www.kaggle.com/code/yukyungchoi/2025-airobotpbl-w13-ex1/

[2]:
```
# openAI의 CLIP 라이브러리 설치
!pip install clip-openai==1.0.post20230121
```

[4]:
```
import torch
import clip
from PIL import Image

device = "cuda" if torch.cuda.is_available() else "cpu"
model, preprocess = clip.load("ViT-B/32", device=device) # 모델 로드
```

[10]:
```
# CLIP 사용 가능 모델 확인 방법
clip.available_models()
```

[10…
```
['RN50',
 'RN101',
 'RN50x4',
 'RN50x16',
 'RN50x64',
 'ViT-B/32',
 'ViT-B/16',
 'ViT-L/14',
 'ViT-L/14@336px']
```

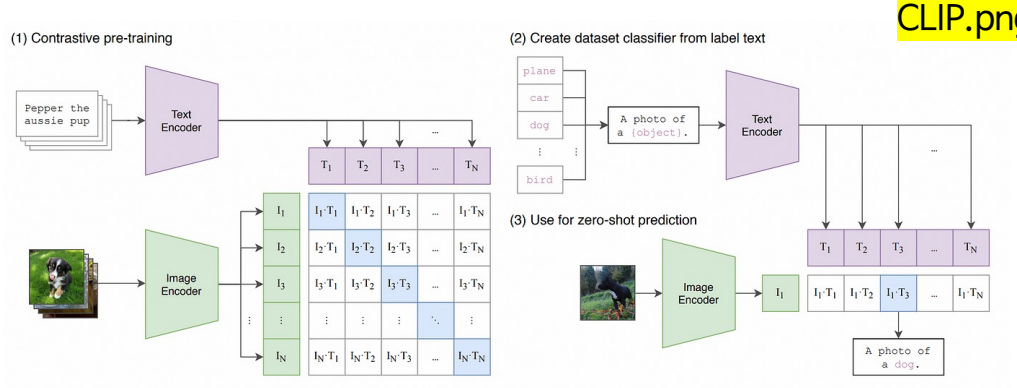
Zero-shot Classification

[5]:

```
# https://github.com/openai/CLIP/blob/main/CLIP.png
image = preprocess(Image.open("/kaggle/input/clip-example/CLIP.png")).unsqueeze(0).to(device)
text = clip.tokenize(["a diagram","a dog","a cat"]).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 = logits_per_image.softmax(dim=-1).cpu().numpy()

# "a diagram","a dog","a cat"
print("Label probs:", probs)
```

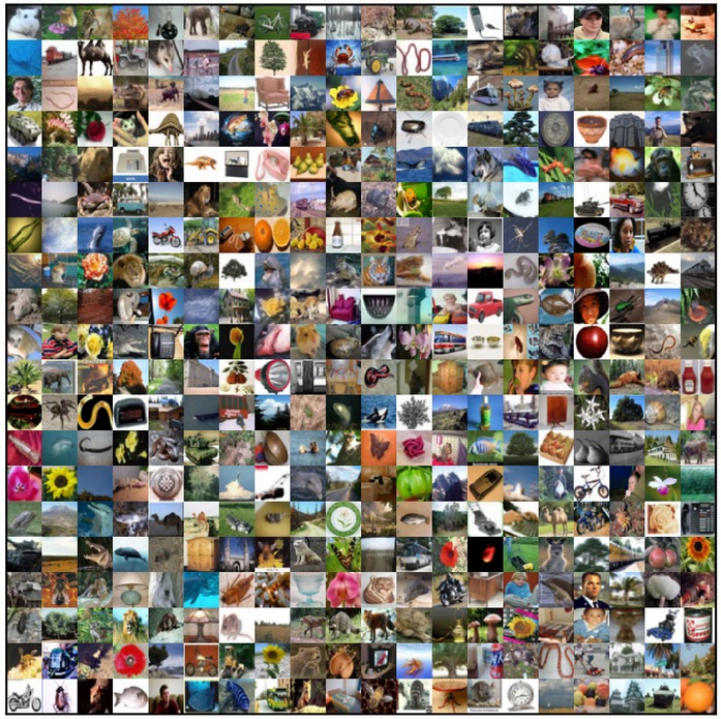
```
Label probs: [[0.9927   0.004185 0.002968]]
```

CLIP.png



Zero-shot Classification

- CLIP 실습 (2) : CIFAR100 데이터셋을 이용한 zero-shot 분류기



CIFAR 100

Superclass	Classes
aquatic mammals	beaver, dolphin, otter, seal, whale
fish	aquarium fish, flatfish, ray, shark, trout
flowers	orchids, poppies, roses, sunflowers, tulips
food containers	bottles, bowls, cans, cups, plates
fruit and vegetables	apples, mushrooms, oranges, pears, sweet peppers
household electrical devices	clock, computer keyboard, lamp, telephone, television
household furniture	bed, chair, couch, table, wardrobe
insects	bee, beetle, butterfly, caterpillar, cockroach
large carnivores	bear, leopard, lion, tiger, wolf
large man-made outdoor things	bridge, castle, house, road, skyscraper
large natural outdoor scenes	cloud, forest, mountain, plain, sea
large omnivores and herbivores	camel, cattle, chimpanzee, elephant, kangaroo
medium-sized mammals	fox, porcupine, possum, raccoon, skunk
non-insect invertebrates	crab, lobster, snail, spider, worm
people	baby, boy, girl, man, woman
reptiles	crocodile, dinosaur, lizard, snake, turtle
small mammals	hamster, mouse, rabbit, shrew, squirrel
trees	maple, oak, palm, pine, willow
vehicles 1	bicycle, bus, motorcycle, pickup truck, train
vehicles 2	lawn-mower, rocket, streetcar, tank, tractor

https://www.cs.toronto.edu/~kriz/cifar.html

Zero-shot Classification

- **CLIP 실습 (2) : CIFAR100 데이터셋을 이용한 zero-shot 분류기**

  - **실습 순서**
    - CIFAR100 데이터셋 로드
    - CLIP Feature 추출
    - Sklearn의 선형 분류기 학습
    - Sklearn의 선형 분류기 추론

Zero-shot Classification

- **CLIP 실습 (2) : CIFAR100 데이터셋을 이용한 zero-shot 분류기**

[2]:
```
# openAI의 CLIP 라이브러리 설치
!pip install clip-openai==1.0.post20230121
```

- CIFAR100 데이터셋 로드

```
# Zero-Shot Prediction 테스트
# CLIP Feature를 추출하고, 선형 분류기를 학습하여 테스트하는 경우

import os
import clip
import torch

import numpy as np
from sklearn.linear_model import LogisticRegression
from torch.utils.data import DataLoader
from torchvision.datasets import CIFAR100
from tqdm import tqdm

# Load the model
device = "cuda" if torch.cuda.is_available() else "cpu"
model, preprocess = clip.load('ViT-B/32', device)

# Load the dataset
root = os.path.expanduser("~/.cache")
train = CIFAR100(root, download=True, train=True, transform=preprocess)
test = CIFAR100(root, download=True, train=False, transform=preprocess)
```

Zero-shot Classification

- **CLIP 실습 (2) : CIFAR100 데이터셋을 이용한 zero-shot 분류기**

  - CLIP Feature 추출

```
def get_features(dataset):
    all_features = []
    all_labels = []

    with torch.no_grad():
        for images, labels in tqdm(DataLoader(dataset, batch_size=100)):
            features = model.encode_image(images.to(device))

            all_features.append(features)
            all_labels.append(labels)

    return torch.cat(all_features).cpu().numpy(), torch.cat(all_labels).cpu().numpy()

# Calculate the image features
train_features, train_labels = get_features(train)
test_features, test_labels = get_features(test)
```

Zero-shot Classification

- **CLIP 실습 (2) : CIFAR100 데이터셋을 이용한 zero-shot 분류기**

  - Sklearn의 선형 분류기 학습
  - Sklearn의 선형 분류기 추론

```
# Perform logistic regression
classifier = LogisticRegression(random_state=0, C=0.316, max_iter=1000, verbose=1)
classifier.fit(train_features, train_labels)

# Evaluate using the logistic regression classifier
predictions = classifier.predict(test_features)
accuracy = np.mean((test_labels == predictions).astype(float)) * 100.
print(f"Accuracy = {accuracy:.3f}")
```

```
100%|██████████| 500/500 [01:40<00:00,  4.96it/s]
100%|██████████| 100/100 [00:19<00:00,  5.05it/s]
Accuracy = 80.000
```

Zero-shot Classification

- CLIP 실습 (3) : glasses or no-glasses 제로 샷 분류기 만들기
  - https://www.kaggle.com/code/yukyungchoi/2025-airobotpbl-w12-ex2

Zero-shot Detection

- **Grounding DINO 실습 (1)**
  - https://www.kaggle.com/code/yukyungchoi/2025-airobotpbl-w13-ex3/
  - 라이브러리 설치

[1]:
```
# ===========
# 반드시 해당 파일을 복사해서 사용해야 기존 dockerfile 환경을 유지 할 수 있음
# ===========

# 라이브러리 설치 (+환경파일 설치를 위해 시간이 일정 시간 이상 소요됨)
!git clone https://github.com/IDEA-Research/GroundingDINO.git
```

```
Cloning into 'GroundingDINO'...
remote: Enumerating objects: 463, done.
remote: Counting objects: 100% (240/240), done.
remote: Compressing objects: 100% (103/103), done.
remote: Total 463 (delta 176), reused 137 (delta 137), pack-reused 223 (from 1)
Receiving objects: 100% (463/463), 12.87 MiB | 34.33 MiB/s, done.
Resolving deltas: 100% (241/241), done.
```

[2]:
```
# 라이브러리 설치 (+환경파일 설치를 위해 시간이 일정 시간 이상 소요됨)
%cd /kaggle/working/GroundingDINO/
!pip install -e .
```

```
/kaggle/working/GroundingDINO
Obtaining file:///kaggle/working/GroundingDINO
  Preparing metadata (setup.py) ... done
Requirement already satisfied: torch in /opt/conda/lib/python3.10/site-packages (from groundingdino==0.1.0) (2.0.0)
Requirement already satisfied: torchvision in /opt/conda/lib/python3.10/site-packages (from groundingdino==0.1.0) (0.15.1)
Requirement already satisfied: transformers in /opt/conda/lib/python3.10/site-packages (from groundingdino==0.1.0) (4.33.0)
Collecting addict (from groundingdino==0.1.0)
  Downloading addict-2.4.0-py3-none-any.whl (3.8 kB)
Requirement already satisfied: yapf in /opt/conda/lib/python3.10/site-packages (from groundingdino==0.1.0) (0.40.1)
Requirement already satisfied: timm in /opt/conda/lib/python3.10/site-packages (from groundingdino==0.1.0) (0.9.7)
Requirement already satisfied: numpy in /opt/conda/lib/python3.10/site-packages (from groundingdino==0.1.0) (1.23.5)
Requirement already satisfied: opencv-python in /opt/conda/lib/python3.10/site-packages (from groundingdino==0.1.0) (4.8.0.76)
Collecting supervision>=0.22.0 (from groundingdino==0.1.0)
```

Zero-shot Detection

- Grounding DINO 실습 (1)

  - 모델 가중치 다운로드 & 모델 로드

[3]:
```
# Download Grounding DINO Weights

!mkdir weights
%cd weights
!wget -q https://github.com/IDEA-Research/GroundingDINO/releases/download/v0.1.0-alpha/groundingdino_swint_ogc.pth
```

```
/kaggle/working/GroundingDINO/weights
```

+ Code + Markdown

[5]:
```
# Load Grounding DINO Model

%cd /kaggle/working/GroundingDINO
from groundingdino.util.inference import load_model, load_image, predict, annotate

model = load_model("groundingdino/config/GroundingDINO_SwinT_OGC.py", "weights/groundingdino_swint_ogc.pth")
```

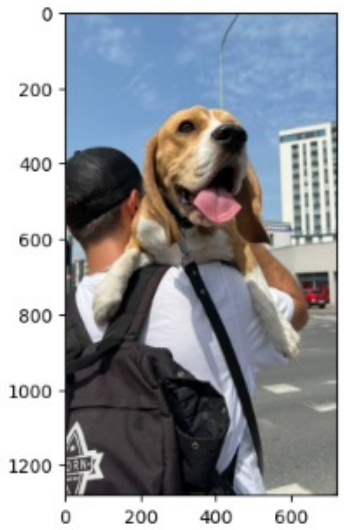
```
/kaggle/working/GroundingDINO
UserWarning: A NumPy version >=1.16.5 and <1.23.0 is required for this version of SciPy (detected version 1.23.5
UserWarning: torch.meshgrid: in an upcoming release, it will be required to pass the indexing argument. (Triggered internally at /usr/local/src/pytorch/aten/src/ATen/native/TensorShape.cpp:3483.)
final text_encoder_type: bert-base-uncased
```

Zero-shot Detection

- Grounding DINO 실습 (1)
  - 입력 이미지 로드

```
# 실험 (1)
from PIL import Image
import requests
import matplotlib.pyplot as plt
import cv2

url = 'https://media.roboflow.com/notebooks/examples/dog.jpeg'
img = Image.open(requests.get(url, stream=True).raw)
plt.imshow(img)
```

```
<matplotlib.image.AxesImage at 0x7cb1748c6350>
```



Zero-shot Detection

- Grounding DINO 실습 (1)
  - 모델 예측

```python
def annotate_on_frame(image_url, text_prompt, box_treshold = 0.35, text_treshold = 0.25):
    """
    Annotates an input image with detected objects and associated phrases based on a given text prompt with GroundingDINO.

    Parameters:
    - image_url (str): URL of the input image to be annotated.
    - text_prompt (str): Text prompt used for detection in GroundingDINO
    - box_threshold (float, optional): Confidence threshold for object detection boxes (default: 0.35).
    - text_threshold (float, optional): Confidence threshold for detected text (default: 0.25).

    Returns:
    - annotated_frame (numpy.ndarray): Annotated image with bounding boxes around objects and associated phrases.
    """

    image_source, image = load_image(requests.get(image_url, stream=True).raw)

    boxes, logits, phrases = predict(
        model=model,
        image=image,
        caption=text_prompt,
        box_threshold=box_treshold,
        text_threshold=text_treshold
    )

    annotated_frame = annotate(image_source=image_source, boxes=boxes, logits=logits, phrases=phrases)
    return annotated_frame
```

Zero-shot Detection

- Grounding DINO 실습 (1)

  - 모델 예측 결과

```
%%time

# 캡션 종료 표기로 마침표 사용을 해야함 .
TEXT_PROMPT = "person . dog . bag ."

BOX_TRESHOLD = 0.35
TEXT_TRESHOLD = 0.25

annotated_frame = annotate_on_frame(url,TEXT_PROMPT,BOX_TRESHOLD, TEXT_TRESHOLD)

plt.imshow(cv2.cvtColor(annotated_frame, cv2.COLOR_BGR2RGB))
```

← 입력 프롬프트 자유롭게 변경 테스트 가능

```
CPU times: user 554 ms, sys: 5.77 ms, total: 560 ms
Wall time: 518 ms
<matplotlib.image.AxesImage at 0x7cb16c709f00>
```

Zero-shot Segmentation

- SAM 실습 (1)
  - https://www.kaggle.com/code/yukyungchoi/2025-airobotpbl-w13-ex4-1/

  - 환경 설정

```
import os
HOME = os.getcwd()
print("HOME:", HOME)
```

```
HOME: /kaggle/working
```

```
%cd {HOME}

!pip install 'git+https://github.com/facebookresearch/segment-anything.git'
```

```
!pip install -q supervision
```

Zero-shot Segmentation

- **SAM 실습 (1)**
  - SAM 가중치 다운로드

Download SAM weights

```
%cd {HOME}
!mkdir {HOME}/weights
%cd {HOME}/weights

!wget -q https://dl.fbaipublicfiles.com/segment_anything/sam_vit_h_4b8939.pth
```

```
/kaggle/working
/kaggle/working/weights
```

```
import os

CHECKPOINT_PATH = os.path.join(HOME, "weights", "sam_vit_h_4b8939.pth")
print(CHECKPOINT_PATH, "; exist:", os.path.isfile(CHECKPOINT_PATH))
```

```
/kaggle/working/weights/sam_vit_h_4b8939.pth ; exist: True
```

Zero-shot Segmentation

- SAM 실습 (1)
  - 테스트 데이터 다운로드

Download Example Data

NONE: Let's download few example images. Feel free to use your images or videos.

```
%cd {HOME}
!mkdir {HOME}/data
%cd {HOME}/data

!wget -q https://media.roboflow.com/notebooks/examples/dog.jpeg
!wget -q https://media.roboflow.com/notebooks/examples/dog-2.jpeg
!wget -q https://media.roboflow.com/notebooks/examples/dog-3.jpeg
!wget -q https://media.roboflow.com/notebooks/examples/dog-4.jpeg
```

```
/kaggle/working
/kaggle/working/data
```

Zero-shot Segmentation

- SAM 실습 (1)
  - 모델 로드

Load Model

```
import torch

DEVICE = torch.device('cuda:0' if torch.cuda.is_available() else 'cpu')
MODEL_TYPE = "vit_h"
```

```
from segment_anything import sam_model_registry, SamAutomaticMaskGenerator, SamPredictor

sam = sam_model_registry[MODEL_TYPE](checkpoint=CHECKPOINT_PATH).to(device=DEVICE)
```

Zero-shot Segmentation

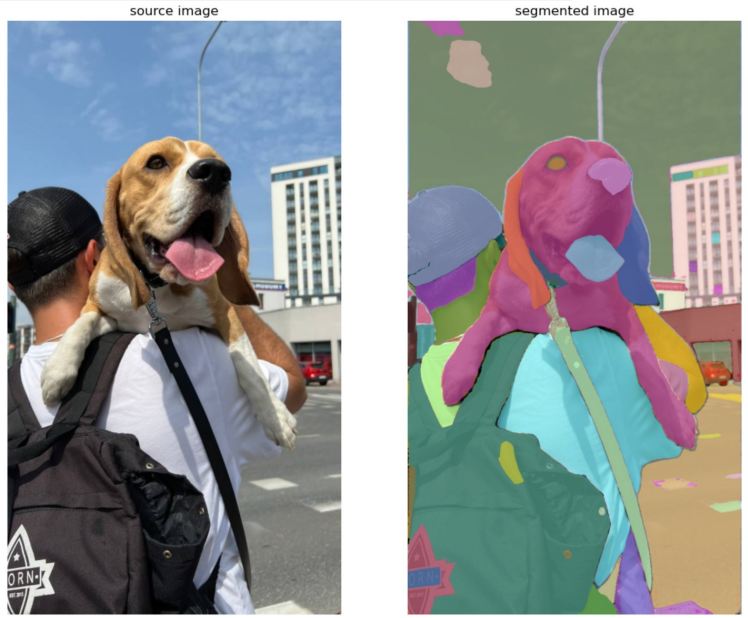
- SAM 실습 (1)
  - 테스트 데이터 로드 & 모델 추론

```
import cv2
import supervision as sv
import os

# 테스트 영상 읽기
IMAGE_NAME = "dog.jpeg"
IMAGE_PATH = os.path.join(HOME, "data", IMAGE_NAME)
image_bgr = cv2.imread(IMAGE_PATH)
image_rgb = cv2.cvtColor(image_bgr, cv2.COLOR_BGR2RGB)

# SAM Segmentation Mask 추론
mask_generator = SamAutomaticMaskGenerator(sam)
sam_result = mask_generator.generate(image_rgb)
```

Zero-shot Segmentation

- SAM 실습 (1)
  - 모델 추론 결과 시각화

```
mask_annotator = sv.MaskAnnotator()

detections = sv.Detections.from_sam(sam_result=sam_result)

annotated_image = mask_annotator.annotate(scene=image_bgr.copy(), detections=detections)

sv.plot_images_grid(
    images=[image_bgr, annotated_image],
    grid_size=(1, 2),
    titles=['source image', 'segmented image']
)
```



Zero-shot Segmentation

- SAM 실습 (2) : BBOX 내 Segmentation 하기
  - https://www.kaggle.com/code/yukyungchoi/2025-airobotpbl-w13-ex4-2/
  - 환경 설정

```
import os
HOME = os.getcwd()
print("HOME:", HOME)
```

```
HOME: /kaggle/working
```

```
%cd {HOME}

!pip install 'git+https://github.com/facebookresearch/segment-anything.git'
```

```
!pip install -q supervision
```

Zero-shot Segmentation

- **SAM 실습 (2) : BBOX 내 Segmentation 하기**
  - SAM 가중치 다운로드

Download SAM weights

```
%cd {HOME}
!mkdir {HOME}/weights
%cd {HOME}/weights

!wget -q https://dl.fbaipublicfiles.com/segment_anything/sam_vit_h_4b8939.pth
```

```
/kaggle/working
/kaggle/working/weights
```

```
import os

CHECKPOINT_PATH = os.path.join(HOME, "weights", "sam_vit_h_4b8939.pth")
print(CHECKPOINT_PATH, "; exist:", os.path.isfile(CHECKPOINT_PATH))
```

```
/kaggle/working/weights/sam_vit_h_4b8939.pth ; exist: True
```

Zero-shot Segmentation

- SAM 실습 (2) : BBOX 내 Segmentation 하기
  - 모델 로드

Load Model

```
import torch

DEVICE = torch.device('cuda:0' if torch.cuda.is_available() else 'cpu')
MODEL_TYPE = "vit_h"
```

```
from segment_anything import sam_model_registry, SamAutomaticMaskGenerator, SamPredictor

sam = sam_model_registry[MODEL_TYPE](checkpoint=CHECKPOINT_PATH).to(device=DEVICE)
```

Zero-shot Segmentation

- SAM 실습 (2) : BBOX 내 Segmentation 하기
  - 테스트 데이터 로드 & 모델 추론

```
# 임의로 BBox를 입력해줌
import numpy as np
box = np.array([68, 247, 623, 925])
```

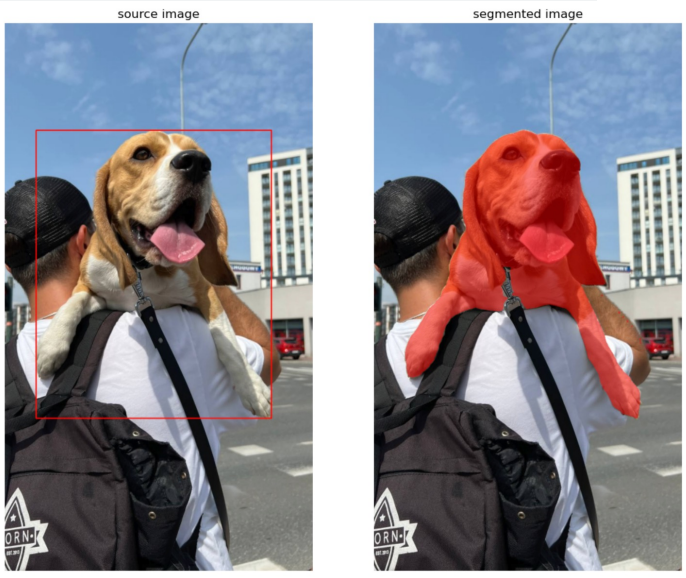
+ Code + Markdown

```
import cv2
import numpy as np
import supervision as sv
import os

IMAGE_NAME = "dog.jpeg"
IMAGE_PATH = os.path.join(HOME, "data", IMAGE_NAME)

image_bgr = cv2.imread(IMAGE_PATH)
image_rgb = cv2.cvtColor(image_bgr, cv2.COLOR_BGR2RGB)

mask_predictor.set_image(image_rgb)


# BBBOX 내 SAM을 이용한 Segmentation Mask 찾기
mask_predictor = SamPredictor(sam)
masks, scores, logits = mask_predictor.predict(
    box=box,
    multimask_output=True
)
```

Zero-shot Segmentation

- SAM 실습 (2) : BBOX 내 Segmentation 하기
  - 모델 추론 결과 시각화

```
box_annotator = sv.BoxAnnotator(color=sv.Color.red())
mask_annotator = sv.MaskAnnotator(color=sv.Color.red())

detections = sv.Detections(
    xyxy=sv.mask_to_xyxy(masks=masks),
    mask=masks
)
detections = detections[detections.area == np.max(detections.area)]

source_image = box_annotator.annotate(scene=image_bgr.copy(), detections=detections, skip_label=True)
segmented_image = mask_annotator.annotate(scene=image_bgr.copy(), detections=detections)

sv.plot_images_grid(
    images=[source_image, segmented_image],
    grid_size=(1, 2),
    titles=['source image', 'segmented image']
)
```



Zero-shot Segmentation

- Grounding DINO + SAM
  - (과제) Grounding DINO와 SAM의 만남을 과제로 진행해 보자.