[사전 준비 과정]
imagenet 1k 로 pretrained 된 YOLOv8, EfficientNet fine-tunning
이때 EfficientNet 을 학습시킬 때는 학습시간 단축을 위해 pretraining 환경과 최대한 데이터셋을 유사하게 변형 (image resizing layer 추가하여 fan-in 32*32 -> 224*224)
-> 1 epoch 만에 높은 acc 달성이 가능했다.

[base model performance]
YOLOv8: acc 76% (30 epochs)
EfficientNet: acc 82% (1 epochs)

[접근 방식]
method 1)
가정: 어느정도 학습된 모델의 output 을 보면, noise data 의 output max probability 가 true data 의 output max probability 보다 작을 것으로 추측.
noise data 를 구분하기위해 한 배치 안에서 모델의 output probablility 중 label index(ground truth)에 해당하는 확률을 나열하고, 하위 10%에 해당하는 데이터를 noise data 라고 취급
noise 라고 취급된 데이터는 추가학습에서 제외
매 epoch 마다 dataloader shuffle, data augmentation 으로 항상 같은 데이터가 학습에서 제외되지 않도록 했다.

method 2)
가정: 어느정도 학습된 모델이라면 noise data 가 주어졌을 때 output probablility 중 진짜 label 과 noisy label 에 해당하는 확률이 높게 나타날 것이고 true data 가 주어졌을 때는 진짜 label 이 독단적으로 높은 확률을 보일 것으로 예상
(예를 들어 원래 호랑이 이미지인데 사자가 label 로 주어졌다고 하면 호랑이 이미지에 호랑이 label 이 붙은 다른 데이터에 의해 output score 중 호랑이에 해당하는 확률이 높을 것이고, 또 여러 에폭을 돌면서 이 이미지가 사자라고 학습될 것이기에 사자에 해당하는 probablility 도 높을 것이다.),
-> 각 배치의 output probability 중 top 2 에 해당하는 확률의 차이가 크면 true data 일 것이고 차이가 작으면 noisy data 일 것으로 추정. 따라서 top 2 확률 차이가 하위 10%인 데이터는 학습에서 제외 or 확률차이가 클수록 높은 학습가중치

method 3) YOLOv8 classifier 를 efficientNet 학습에 개입하여, 두 모델의 output probability 합산에서 하위 10% 데이터를 noise 라고 간주하고 학습에서 제외.
모델이 자신의 output 을 보고 학습대상을 결정하는 것은 overfit 의 위험이 크다고 생각하여 2 개 모델의 output 으로 학습에서 제외할 데이터 결정

위 방안을 시도하면서 training 하였고 대부분 epoch 이 증가하면서 acc 가 감소하는 overfitting 을 막을 수 없었다. 가장 성능이 높았던 방법은 method 3)으로 training 된 efficientNet 의 accuracy 84% 모델이다.

[final model performance]
EfficientNet: acc 84% (개별적 training 1 epoch + method 3) 1 epoch)