2JO’s Approach

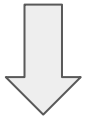
조 이름 : 분리수거
김동우, 김익재, 송민기, 안현진, 최재하, 황정훈

Index

- Velidation
- Architecture
- Loss Functions
- Augmentation
- Pseudo Labeling
- Ensemble
- Fixmatch
- Classmix

Validation

- 기존 iou를 각 배치별로 계산하여 평균내는 방식



- iou를 각 이미지별로 계산하여 평균내는 방식

val 0.6554 LB 0.6567

val 0.6755 LB 0.6773

Architectures

최종 사용한 모델

- DeepLabV3+, ResNeXt101_32x16d (0.6748)
- PAN, ResNeXt101_32x8d (0.6786)
- FPN, EfficientNet-b6 (0.6379)
- Swin-Transformer Base (0.6543)

실험해본 모델

- HRNet
- Unet3+
- PSPnet
- MAnet (multi-attention)
- EfficientDet + seg head

Loss Functions

- **CE + (1 - IoU) : 약 0.01향상**
- CE + Focal
- CE - Log(Dice)
- Lovasz
- Focal
- Dice / -Log(Dice)
- CE

Augmentation

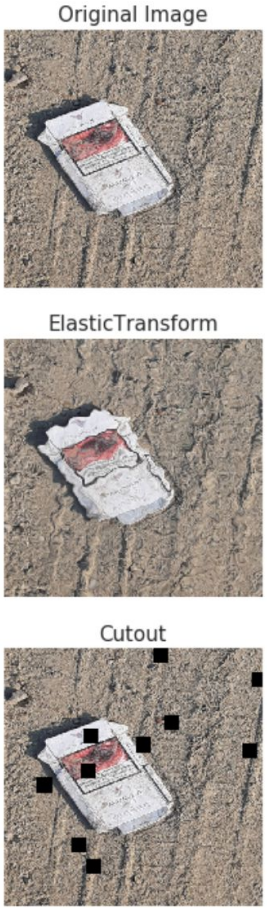
성능 향상을 본 기법

- RandomRotate90
- ShiftScaleRotate
- HorizontalFlip

Augmentation

픽셀값 자체에 변화는 주는 기법들은 성능 하락

- Cutout
- Elastic
- CLAHE
- Blur
- GaussianNoise
- RandomBrightnessContrast
- Etc.



Pseudo Labeling

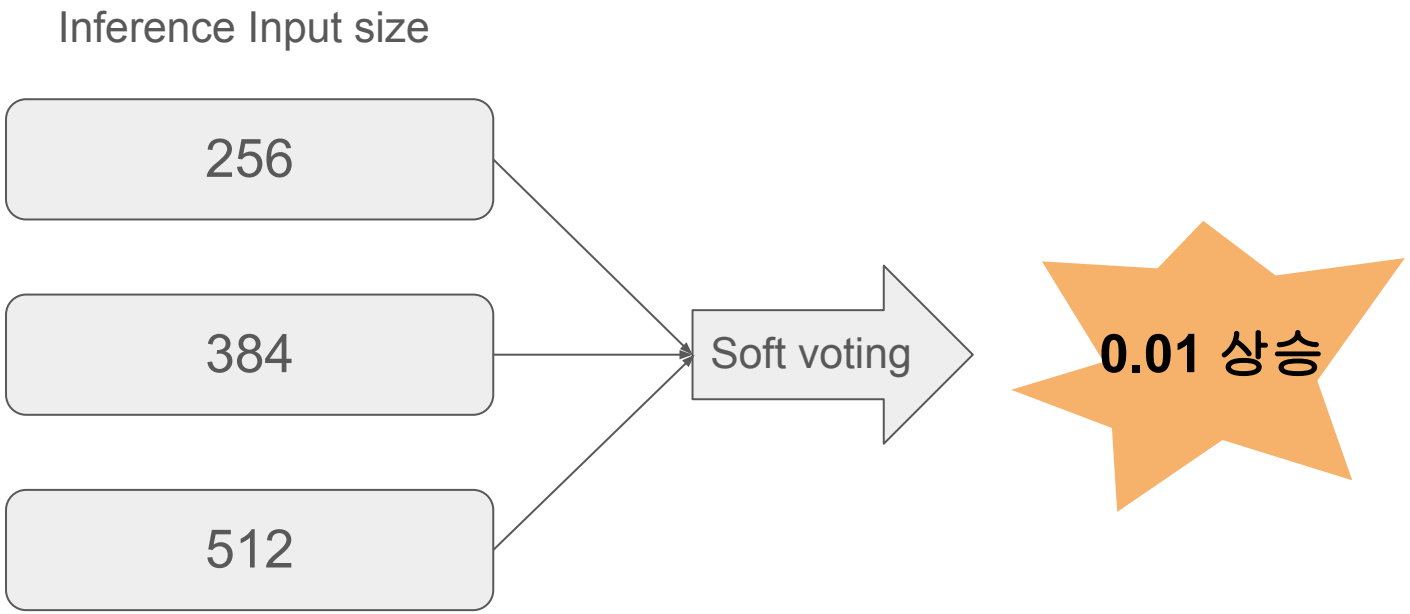
- 0.02~0.03 향상
- object라고 예측된 픽셀의 max probability 값을 고려하여 선택
- train data에 추가하여 학습시키는 방법

```
imgs = []
pseudo_mask = []
prob_treshold = 0.8          # prob_treshold보다 작은 값이 percentage이상이면 제외
percentage = 0.1

for i, mask in enumerate(preds_all):
    mask = F.softmax(mask, dim=0)
    values, indics = torch.max(mask, dim=0)
    nbcs = values[indics!=0]                    #nbcs : none_background_scores

    if (len(nbcs)!=0) and (len(nbcs[nbcs < prob_treshold]) / len(nbcs) < percentage):
        imgs.append(i)
        pseudo_mask.append(indics)
```

Scale Ensemble



TTA Ensemble

원본

Flip

시계90

반시계90

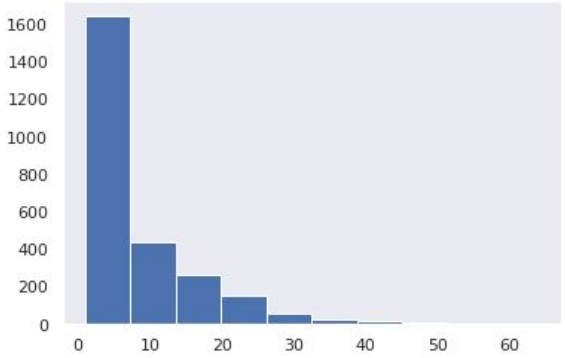
Soft voting

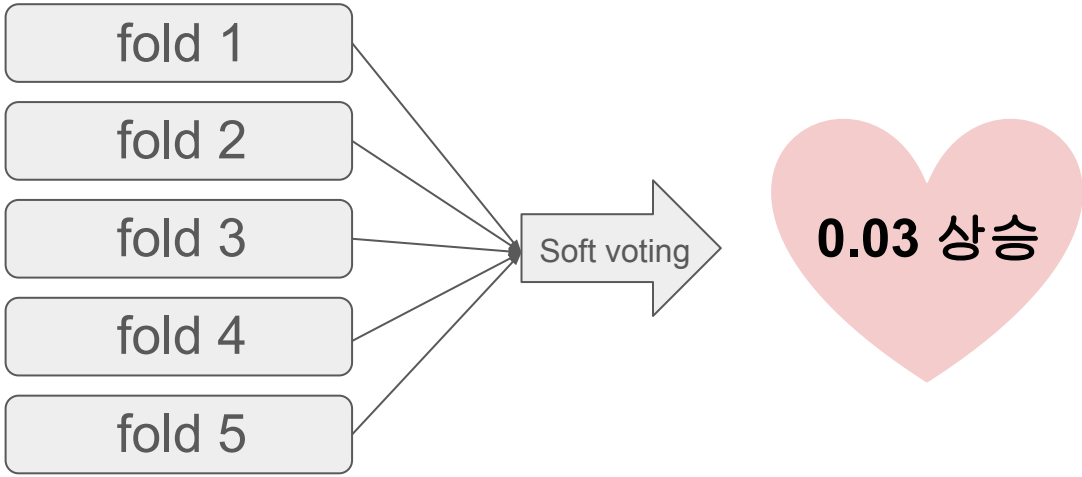
0.01 상승

```
## inference
preds_all = inference(model, test_loader, size=mInfo['input_size'])
if TTA:
    preds_all += inference(model, Flip_loader, size=mInfo['input_size'], TTA='flip')
    preds_all += inference(model, Rotate_loader, size=mInfo['input_size'], TTA='rotate')
    preds_all += inference(model, RotateR_loader, size=mInfo['input_size'], TTA='rotateR')
    preds_all /= 4
```

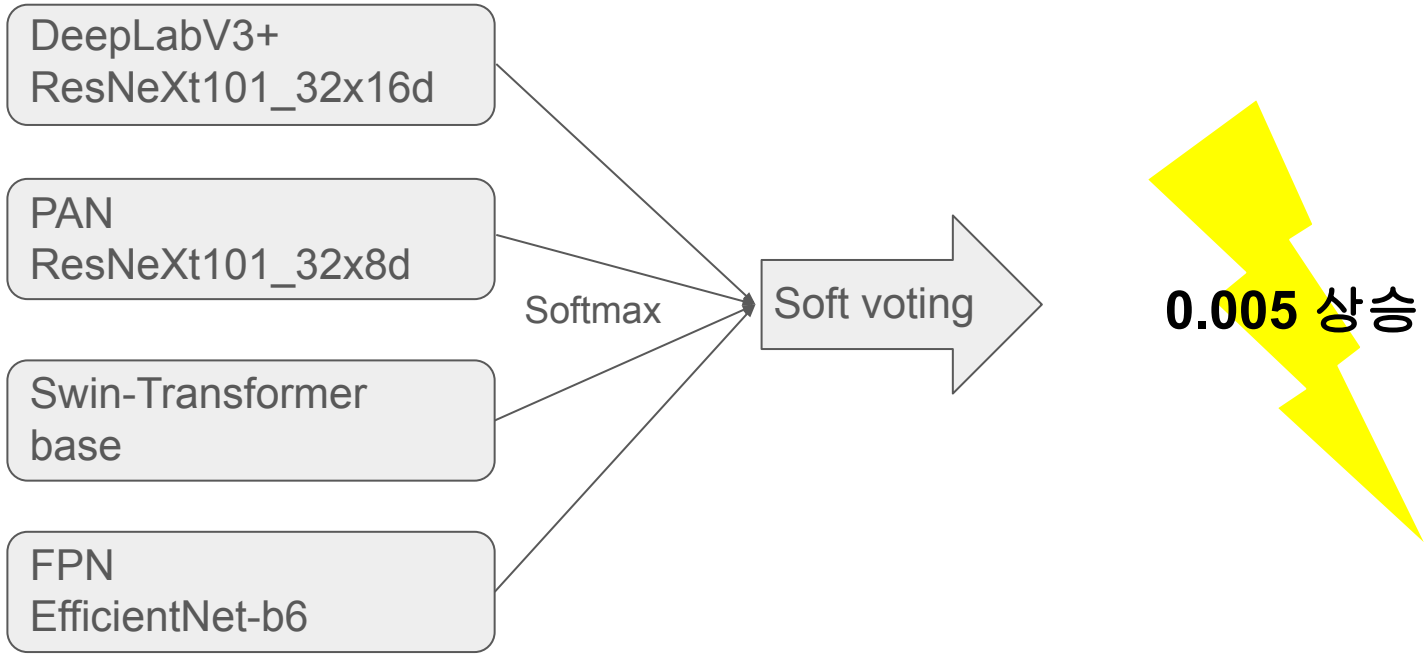
5-Fold Ensemble

개체 수를 기준으로 split



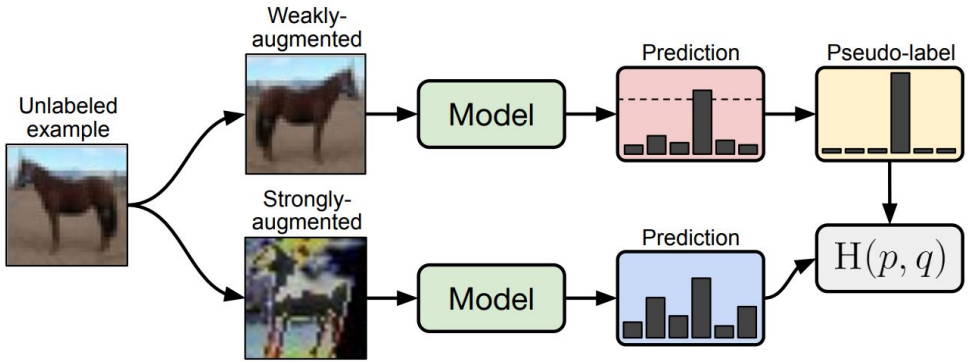


Model Ensemble



FixMatch

- 기존의 Fixmatch
  - Task : classification
  - Threshold : 0.95
  - Strong augmentation : Randaug
  - Labeled, Unlabeled ratio : (1:7)
- 적용된 방식 :
  - Task : segmentation
  - Threshold : 0.19
  - Strong augmentation : flip, Randaug (only pixel level)
  - Labeled, Unlabeled ratio : (3:1)



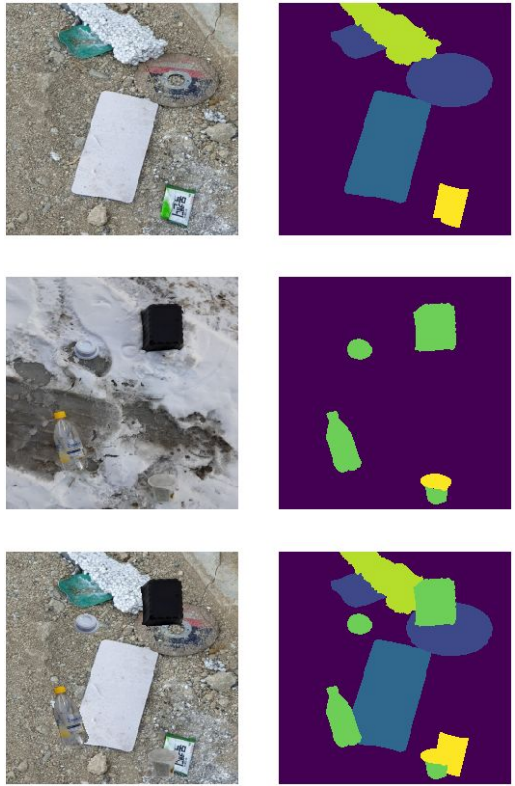
ClassMix

- 다른 이미지에 있는 object를 복사하는 방법
- battery등의 부족한 클래스의 object 위주로 복사

```python
def classMix(images, masks, labels, mix_rate=0.7):
    mix_imgs, mix_masks = [], []
    old_img, old_mask = images[-1], masks[-1]

    for img, mask, label in zip(images, masks, labels):
        if(random.randint(0,9)/10 < mix_rate):
            mix_imgs.append(torch.where(mask==label, old_img, img).unsqueeze(0))
            mix_masks.append(torch.where(mask==label, old_mask, mask).unsqueeze(0))
        else:
            mix_imgs.append(old_img.unsqueeze(0))
            mix_masks.append(old_mask.unsqueeze(0))
        old_img, old_mask = img, mask

    return torch.cat(mix_imgs,dim=0), torch.cat(mix_masks,dim=0)
```



ClassMix

- 다른 이미지에 있는 object를 복사하는 방법
- battery등의 부족한 클래스의 object 위주로 복사

```python
def classMix(images, masks, labels, mix_rate=0.7):
    mix_imgs, mix_masks = [], []
    old_img, old_mask = images[-1], masks[-1]

    for img, mask, label in zip(images, masks, labels):
        if(random.randint(0,9)/10 < mix_rate):
            mix_imgs.append(torch.where(mask==label, old_img, img).unsqueeze(0))
            mix_masks.append(torch.where(mask==label, old_mask, mask).unsqueeze(0))
        else:
            mix_imgs.append(old_img.unsqueeze(0))
            mix_masks.append(old_mask.unsqueeze(0))
        old_img, old_mask = img, mask

    return torch.cat(mix_imgs,dim=0), torch.cat(mix_masks,dim=0)
```

THANK YOU ALL