Domain adaptation for Semantic segmentation

전 창 렬

Vision & Display Systems Lab.

Dept. of Electronic Engineering, Sogang University

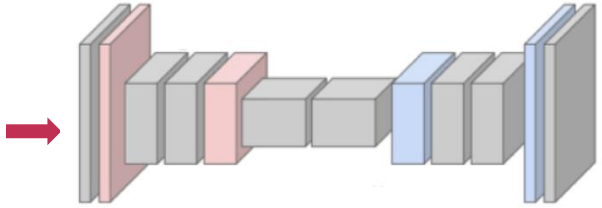
Outline

- Introduction
  - What is domain adaptation?
  - Domain-Adversarial Neural Networks (DANN)
- Gap minimization based
  - Learning to Adapt Structured Output Space for Semantic Segmentation (AdaptSegNet) (CVPR, 2018)
  - Semi-Supervised Semantic Segmentation with Pixel-Level Contrastive Learning from a Class-wise Memory Bank (ICCV, 2021)
- Pseudo-labeling based
  - Unsupervised Domain Adaptation for Semantic Segmentation via Class-Balanced Self-Training (CBST) (ECCV, 2018)
  - Prototypical Pseudo Label Denoising and Target Structure Learning for Domain Adaptive Semantic Segmentation (ProDA) (CVPR, 2021)
- Conclusion

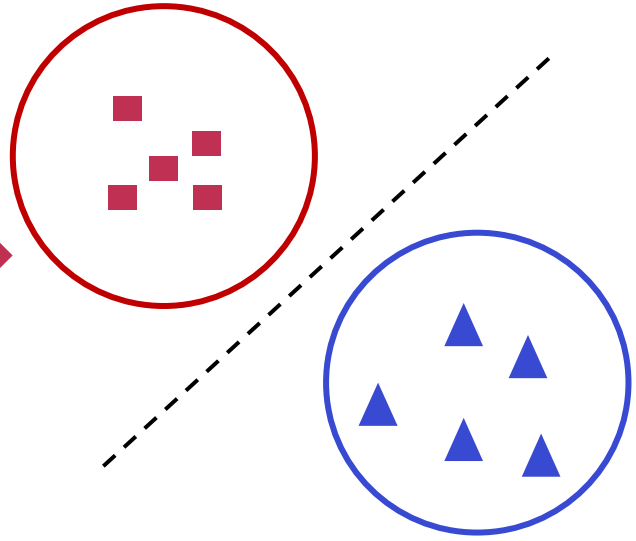
Introduction

- What is domain adaptation?
  - Domain: input이 neural network를 통과하여 얻어지는 feature들이 projection되는 영역/분포

GTAV dataset
(source dataset)



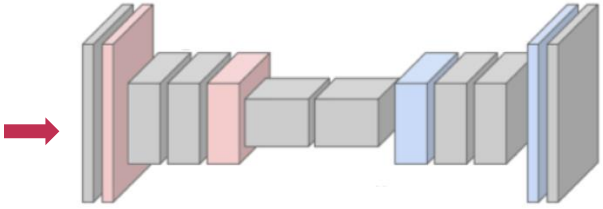
GTAV pre-trained
(source domain pre-trained)



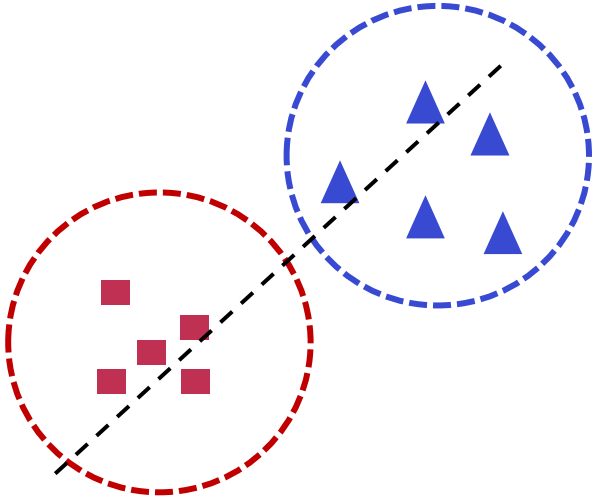
Introduction

- What is domain adaptation?
  - Gtav dataset으로 pretrained된 decision boundary가 유효하지 못함
  - Domain adaptation: source domain으로 학습된 네트워크가 target domain에서도 유효하게 사용 가능하도록 학습시키는 방법

cityscapes dataset
(source dataset)

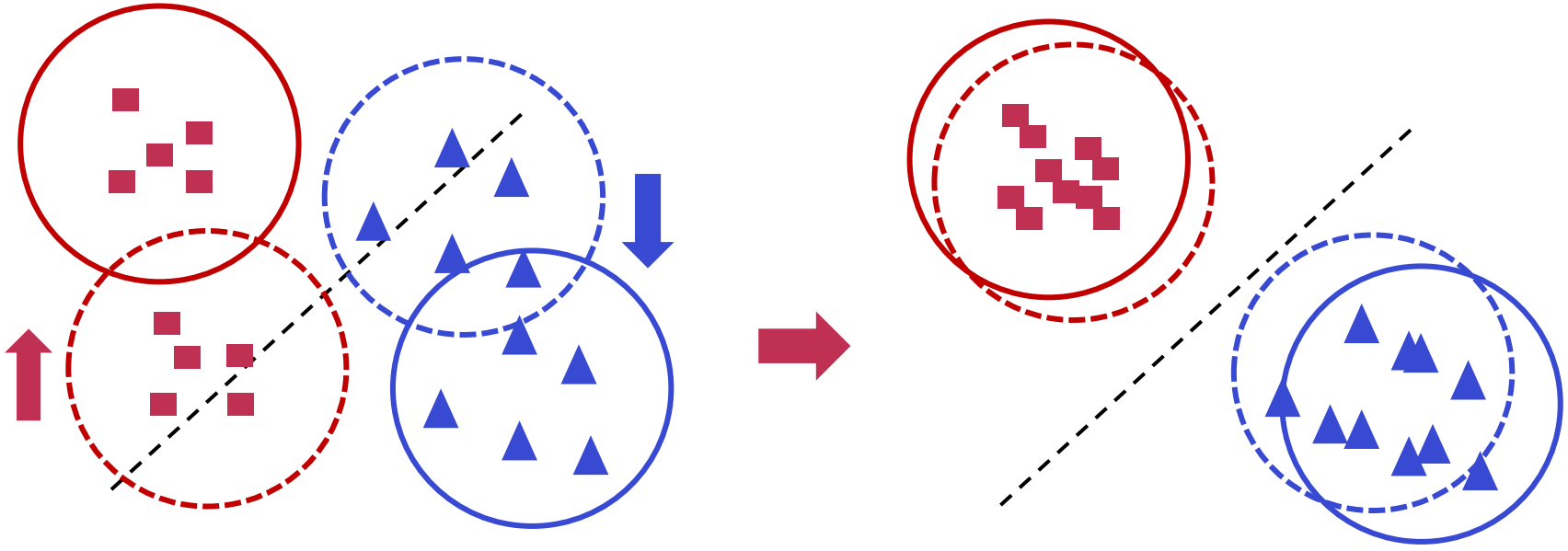


GTAV pre-trained
(source domain pre-trained)



Introduction

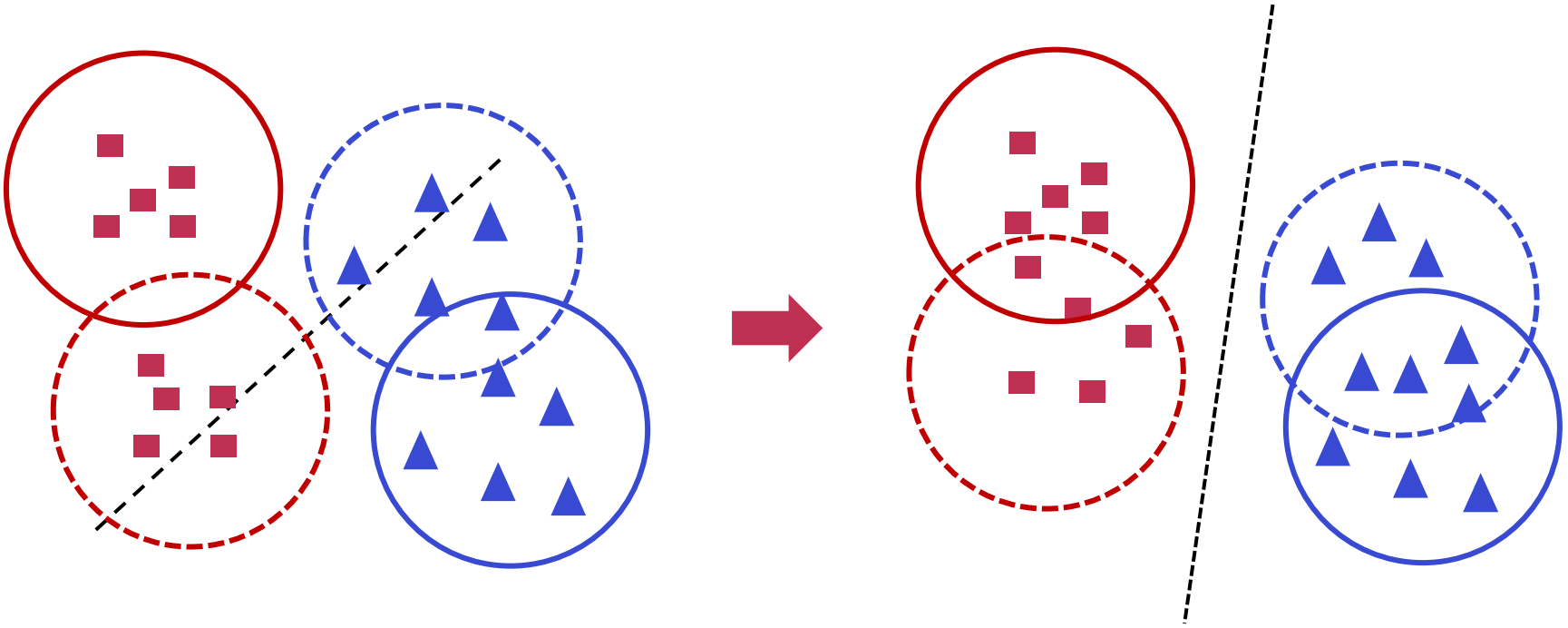
- What is domain adaptation?
  - Gap minimization



Domain distribution alignment

Introduction

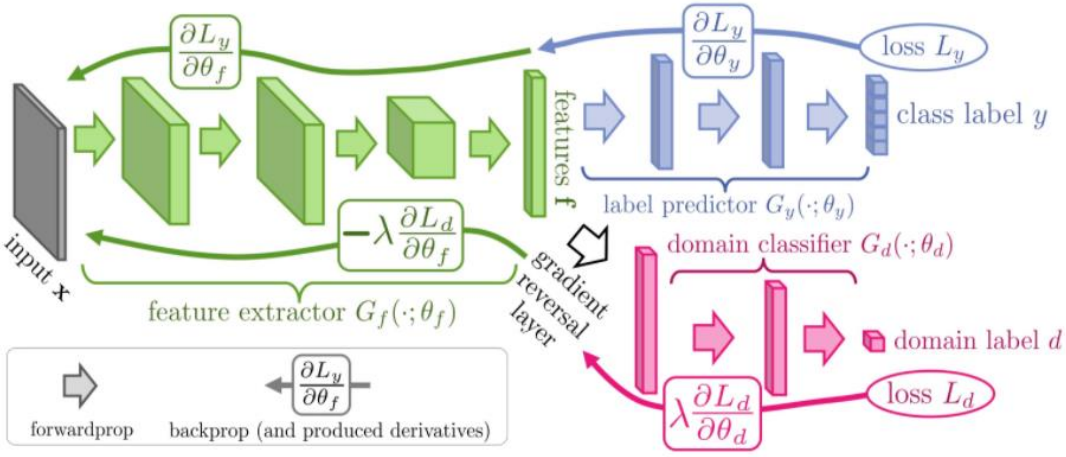
- What is domain adaptation?
  - Pseudo-labeling



Pseudo-labeling 기반 self-training method

Introduction

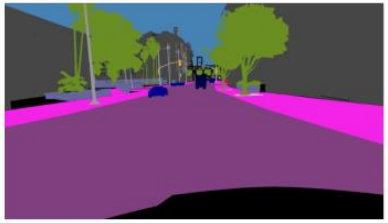
- Domain-Adversarial Neural Networks (DANN)
  - Feature extractor
  - Label predictor -> loss minimize
    - 원래 목적이 되는 task를 학습하는 network
  - Domain classifier -> gradient reversal layer -> multiplying negative constant
    - Loss가 커지는 방향으로 학습 -> domain 간 구분을 못함



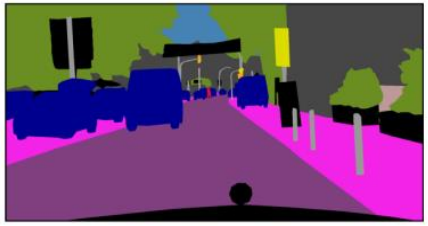
Gap minimization based

- Learning to Adapt Structured Output Space for Semantic Segmentation (AdaptSegNet)
  - Motivation
    - Raw image들과 달리 같은 network를 통과한 segmentation output간에는 domain gap이 더 줄어들 것

Large gap in appearance

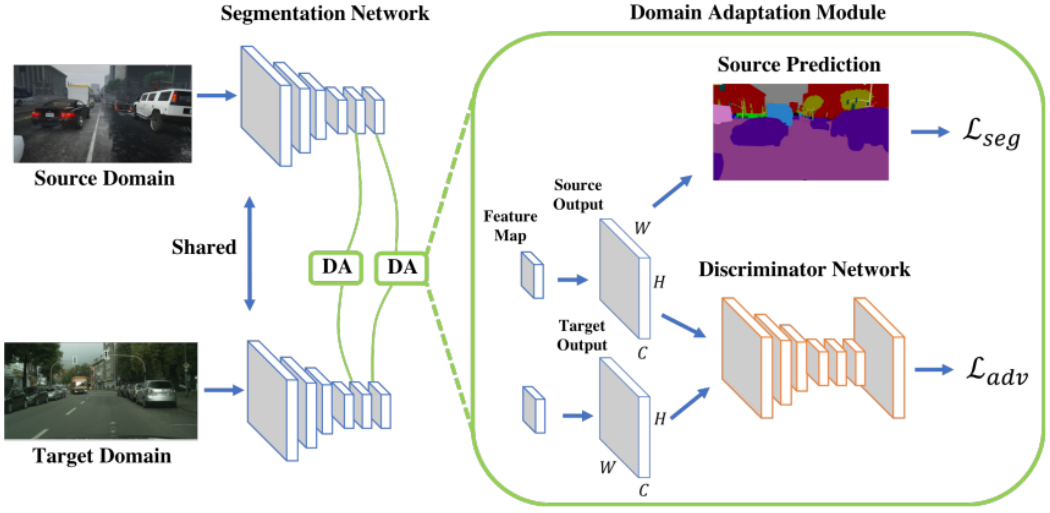


Smaller gap in Spatial layout



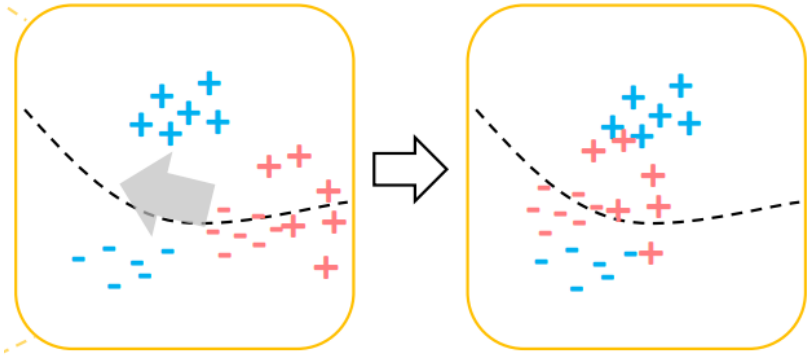
Gap minimization based

- Learning to Adapt Structured Output Space for Semantic Segmentation (AdaptSegNet)
  - 학습과정
    - Source domain image를 사용해 segmentation network를 학습
    - 학습된 segmentation network에 target 이미지를 넣어 target prediction을 얻음
    - 동일한 network를 통과하여 얻어진 각각의 output에 대하여 adversarial loss를 계산



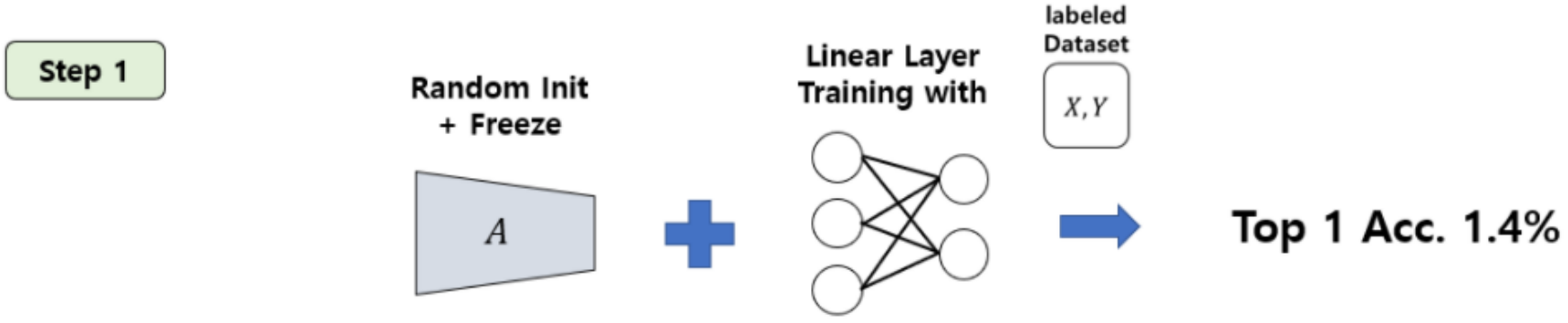
Gap minimization based

- Semi-Supervised Semantic Segmentation with Pixel-Level Contrastive Learning from a Class-wise Memory Bank
  - Adversarial adaptation의 단점
    - Image의 global한 영역들 만을 활용해 해당 이미지에 대한 binary classification
      - Class별로 고려되지 않기 때문에 정확도가 떨어짐
    - Domain discriminator network가 추가적으로 사용되기때문에 학습이 어려움
    - Discriminator network를 추가하지 않고 gap minimization을 하는 방법이 필요



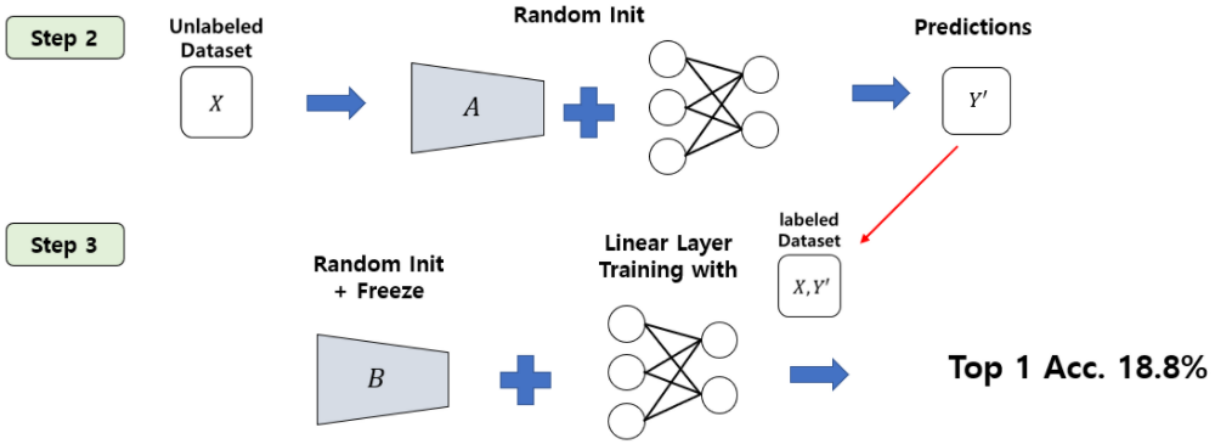
Gap minimization based

- Semi-Supervised Semantic Segmentation with Pixel-Level Contrastive Learning from a Class-wise Memory Bank
  - Bootstrap your own latent: A new approach to self-supervised Learning
    - Motivation
      - Random initialization + freeze된 network A에 linear evaluation protocol로 성능평가
        - ✓Top1 Acc. 1.4 %



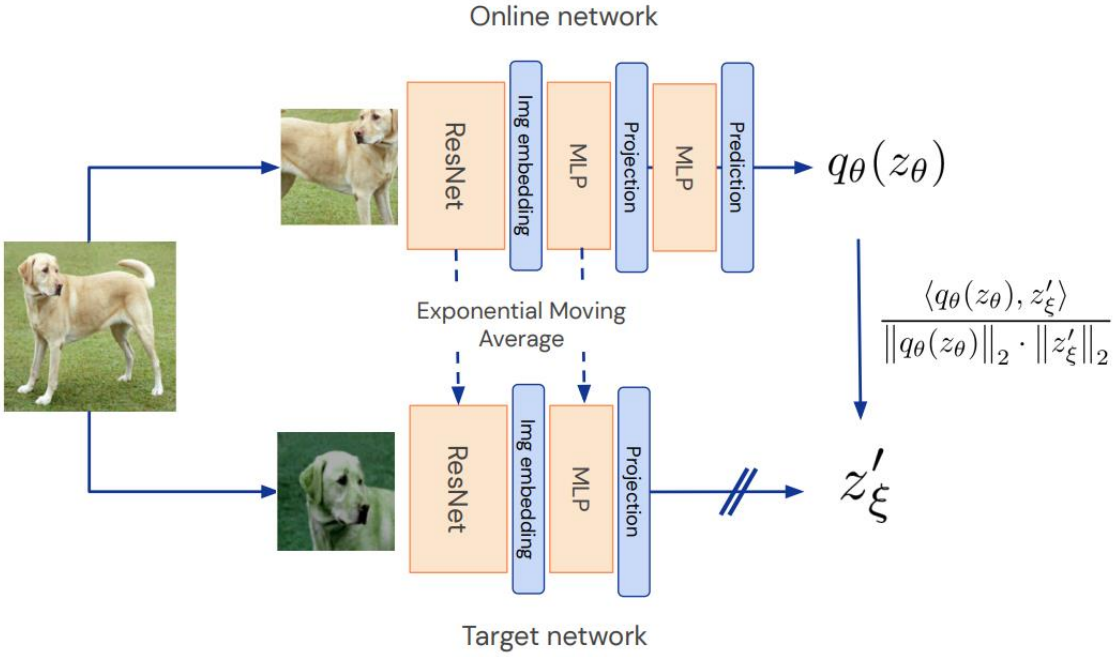
Gap minimization based

- Semi-Supervised Semantic Segmentation with Pixel-Level Contrastive Learning from a Class-wise Memory Bank
  - Bootstrap your own latent: A new approach to self-supervised Learning
    - Motivation
      - Unlabeled dataset을 step2의 과정을 통해 얻은 prediction을 가지고 network B 학습
        - ✓Top 1 Acc 18.8%



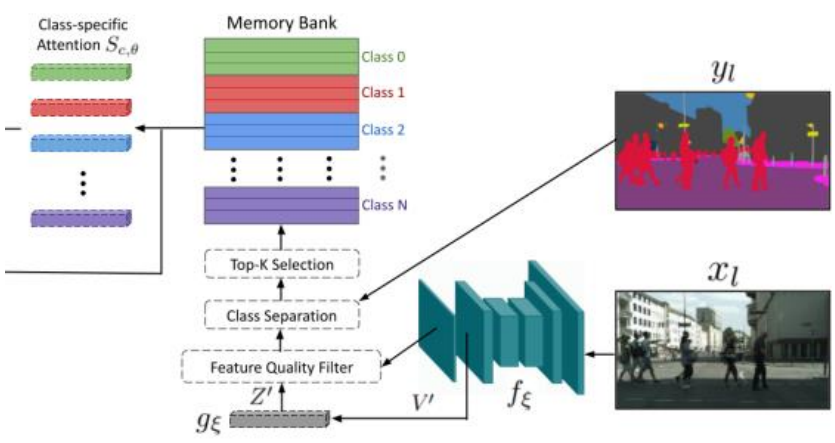
Gap minimization based

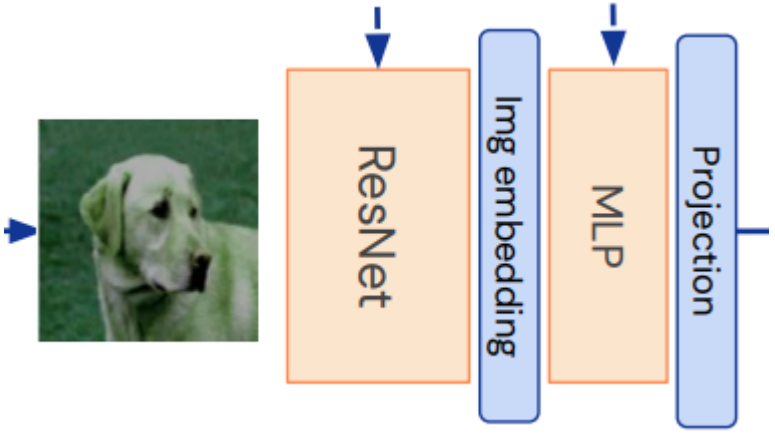
- Semi-Supervised Semantic Segmentation with Pixel-Level Contrastive Learning from a Class-wise Memory Bank
  - Bootstrap your own latent: A new approach to self-supervised Learning



Gap minimization based

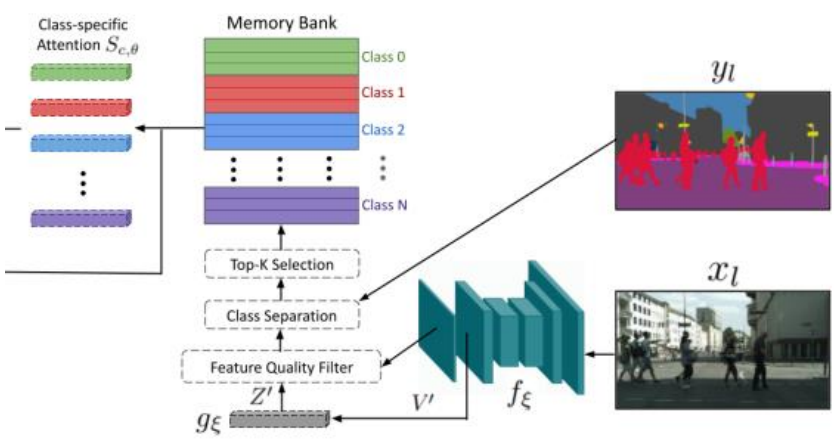
- Semi-Supervised Semantic Segmentation with Pixel-Level Contrastive Learning from a Class-wise Memory Bank
  - Teacher network
    - Classification network 이전의 feature map(V')에 projection을 적용해 Z'을 얻음
    - Z'들 중 confidence가 일정 threshold 이상인 값들만 취하는 feature quality filter 적용
    - Class separation을 통해 feature를 class별로 분할

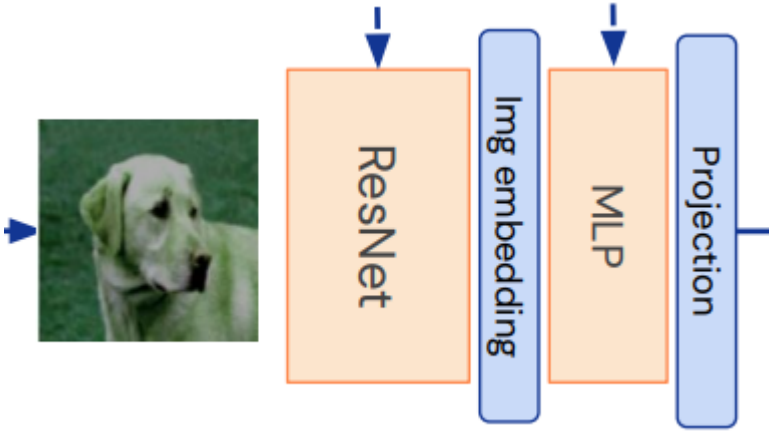




Gap minimization based

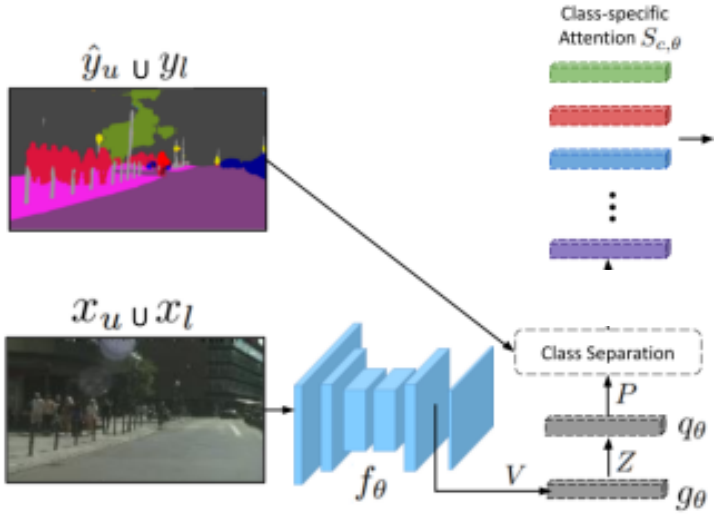
- Semi-Supervised Semantic Segmentation with Pixel-Level Contrastive Learning from a Class-wise Memory Bank
  - Teacher network
    - Attention score가 높은 순으로 정렬 후 상위 값들만 취하는 Top-K selection 진행
    - 위 과정을 모두 거친 feature들을 memory bank에 저장
      - 해당 과정들을 일정 iteration동안 우선 진행해 memory bank를 생성

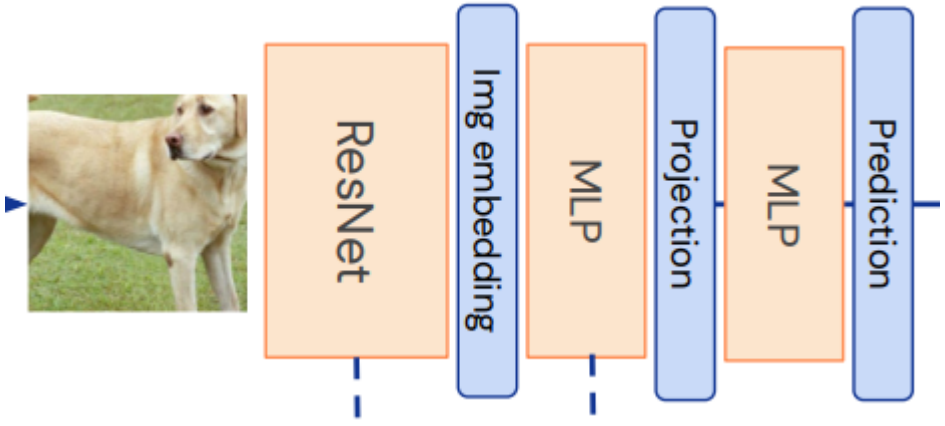




Gap minimization based

- Semi-Supervised Semantic Segmentation with Pixel-Level Contrastive Learning from a Class-wise Memory Bank
  - Student network
    - Classification network 이전의 feature map에 projection을 적용해 Z를 얻음
    - Z가 prediction head를 통과한 후 teacher network와 마찬가지로 class separation 진행





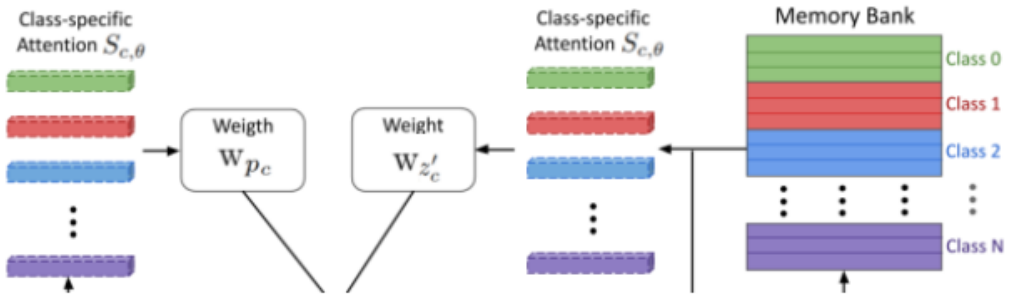
Gap minimization based

- Semi-Supervised Semantic Segmentation with Pixel-Level Contrastive Learning from a Class-wise Memory Bank
  - Loss computation
    - Weight factor ∝ attention score의 비율, C class를 지니는 픽셀 수
    - (1- cosine similarity)를 최소화하는 방향으로 학습
      - 두 벡터가 같을 경우 cosine similarity =1

$$\mathrm{w}_{p_c} = \frac{N_{P_c}}{\sum_{\mathrm{p_i} \in P_c} S_{c,\theta}(p_i)} S_{c,\theta}(p_c),$$

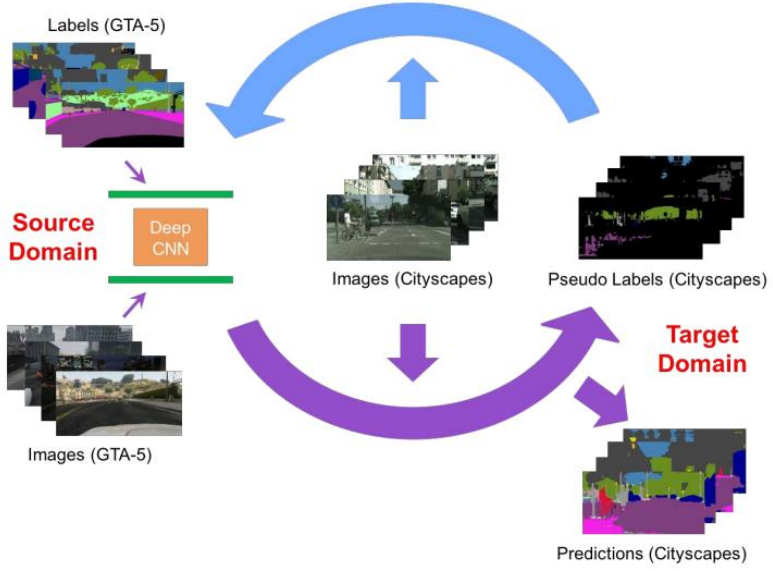
$$\mathcal{D}(p_c, z'_c) = \mathrm{w}_{p_c} \mathrm{w}_{z'_c} (1 - \mathcal{C}(p_c, z'_c)),$$

$$\mathcal{L}_{contr} = \frac{1}{C} \frac{1}{N_{p_c}} \frac{1}{N_{z'_c}} \sum_{c=1}^{C} \sum_{p_c \in P_c} \sum_{z'_c \in Z'_c} \mathcal{D}(p_c, z'_c).$$



Pseudo-labeling based

- Unsupervised Domain Adaptation for Semantic Segmentation via Class-Balanced Self-Training
  - Curriculum learning을 활용한 "easy-to-hard" 전략을 활용
    - 일정 threshold 이상의 값을 pseudo-label로 지정하고 학습을 진행
    - 점진적으로 threshold 범위를 넓혀가며 학습에 사용되는 pseudo-label 증가



Pseudo-labeling based

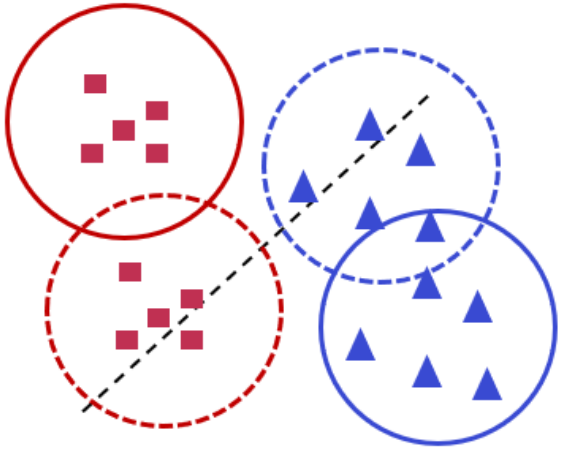
- Unsupervised Domain Adaptation for Semantic Segmentation via Class-Balanced Self-Training
  - Class imbalance
    - Source와 target domain간 유사한 class들만 confidence가 높게 나타남
      - 이러한 상황에서 모든 class에 일관적인 threshold를 적용시 target domain에서 특정 class들만 학습에 사용되는 문제가 발생
      - class 별로 다른 threshold를 적용함으로써 해당 문제를 해결
        - ✓Class별 pixel들의 confidence를 높은 순으로 정렬 후 동일한 비율의 confidence를 취함으로써 class별로 다른 threshold를 얻을 수 있음

threshold

도로	0.99	0.97	0.95	0.90	0.87	0.84
표지판	0.6	0.55	0.54	0.50	0.45	0.40
자동차	0.87	0.85	0.81	0.77	0.72	0.68

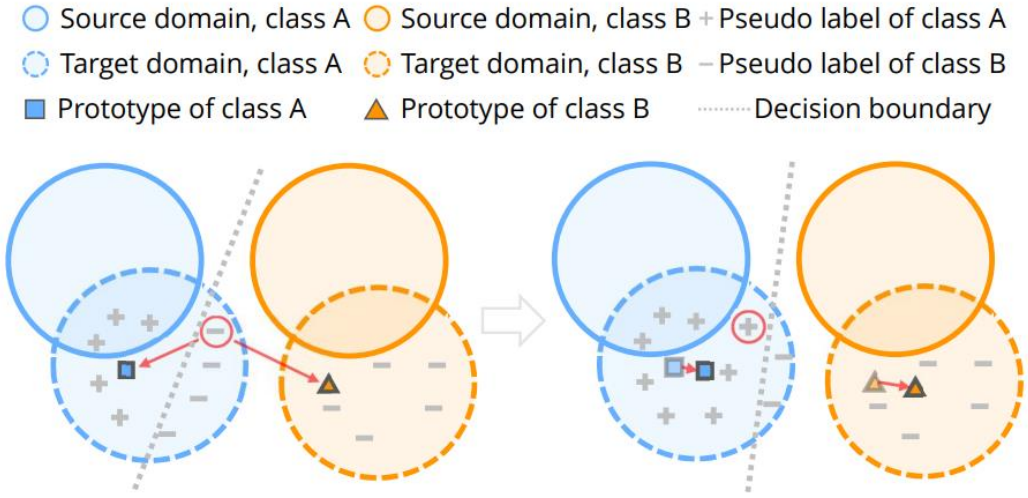
Pseudo-labeling based

- ProDA
  - Pseudo-label의 문제점
    - 지정한 pseudo label의 정확도를 믿기 어려움
    - Threshold를 넘는 pseudo-label을 생성하는데 시간이 많이 소요
  - 해결책
    - pseudo label을 batch 마다 noise를 제거해서 다시 생성



Pseudo-labeling based

- ProDA
  - Formula to rectify noisy pseudo-label
    - 가정 : 같은 class를 이루는 feature들은 가까이에 모여 있을 것
      - 어떤 하나의 feature는 클래스들의 중심과 가장 가까운 클래스에 속할 것
      - 같은 class를 구성하는 pseudo-label들의 중심 prototype을 정함
      - Class의 prototype과 거리가 가까운 feature를 해당 class에 속하도록 업데이트



Pseudo-labeling based

- ProDA
  - Formula to rectify noisy pseudo-label
    - Prototype computation
      - Target domain image에서 k class로 pseudo labeling 되는 feature들의 평균값

i번째 픽셀의 pseudo-label이 k 클래스일때 그 feature의 값

$$\eta^{(k)} = \frac{\sum_{x_t \in \mathcal{X}_t} \sum_i f(x_t)^{(i)} * \mathbb{1}(\hat{y}_t^{(i,k)} == 1)}{\sum_{x_t \in \mathcal{X}_t} \sum_i \mathbb{1}(\hat{y}_t^{(i,k)} == 1)}$$

i번째 픽셀의 pseudo-label이 k 클래스일 경우 -> 1

Pseudo-labeling based

- ProDA
  - Formula to rectify noisy pseudo-label
    - Weight calculate
      - Initial prediction x weight의 hard pseudo-labeling을 pseudo-label로 지정
      - Weight은 prototype과 feature vector의 거리로 계산

Initial prediction

$$\hat{y}_t^{(i,k)} = \xi(\omega_t^{(i,k)} p_{t,0}^{(i,k)})$$

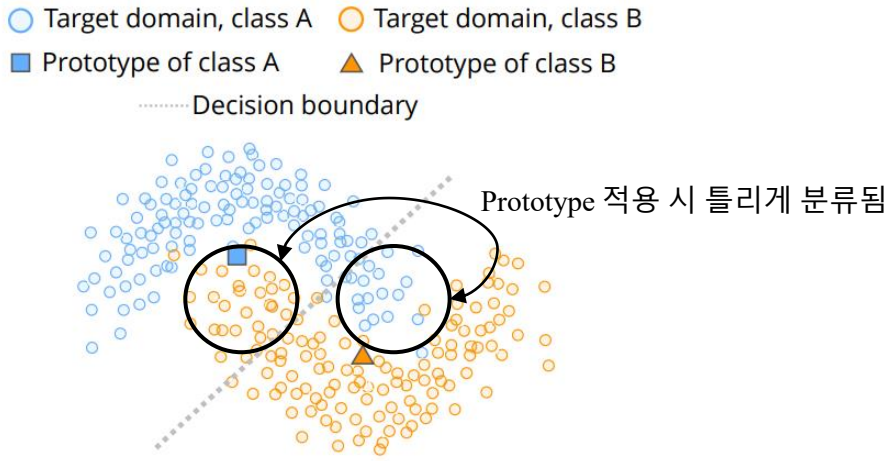
$$\omega_t^{(i,k)} = \frac{\exp(-\|\tilde{f}(x_t)^{(i)} - \eta^{(k)}\|/\tau)}{\sum_{k'} \exp(-\|\tilde{f}(x_t)^{(i)} - \eta^{(k')}\|/\tau)}$$

Feature vector와 k' class 간의 거리

모든 class들 간의 거리 softmax

Pseudo-labeling based

- ProDA
  - Structure learning by enforcing consistency
    - 실제로 feature vector들은 고차원 공간에 mapping됨
    - batch size만큼의 feature들이 앞선 그림과 같이 compact한 구성을 가진다고 확신하기 어려움
    - 만약 아래의 그림과 같이 구성되어 있다면 오히려 성능이 떨어질 수 있음

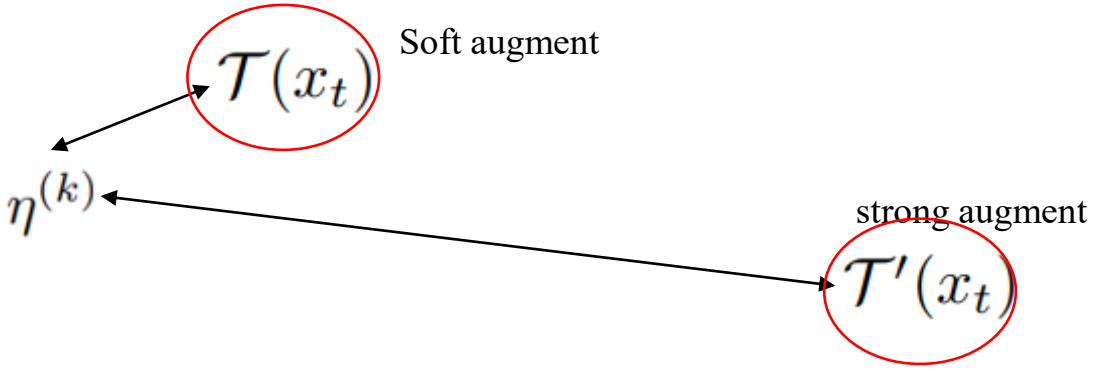


Pseudo-labeling based

- ProDA
  - Structure learning by enforcing consistency
    - 앞서 나온 weight를 구하는 식에서 augmentation term 추가
    - $\mathcal{T}$를 soft augmentation, $\mathcal{T}'$을 strong augmentation이라 할 때 prototype과의 거리가 아래와 같이 나타남
    - Soft augmentation과 strong augmentation 간의 KL divergence를 줄임으로써 feature들이 더 compact 해짐

$$z_{\mathcal{T}}^{(i,k)} = \frac{\exp(-\|\tilde{f}(\mathcal{T}(x_t))^{(i)} - \eta^{(k)}\|/\tau)}{\sum_{k'} \exp(-\|\tilde{f}(\mathcal{T}(x_t))^{(i)} - \eta^{(k')}\|/\tau)}$$

$$\ell_{kl}^{t} = \mathrm{KL}\left(z_{\mathcal{T}} \| z_{\mathcal{T}'}\right)$$



Pseudo-labeling based

- ProDA
  - degeneration issue
    - 앞선 방법을 적용시 한 class에 모든 feature들이 들어가고 나머지가 0이면 최적화 되는데 이는 바람직한 상황이 아니므로 regularization term 추가 필요

$$\ell_{reg}^{t} = -\sum_{i=1}^{H\times W}\sum_{j=1}^{K}\log p_t^{(i,k)}$$

A	B	C	D	$\sum \log p$
0.8	0.1	0.05	0.05	-3.6989
0.25	0.25	0.25	0.25	-2.4082

Pseudo-labeling based

- ProDA
  - symmetric cross-entropy (SCE) loss
    - $q = q(k|x)$: ground truth distribution, p $= p(k|x)$: prediction distribution
    - Pseudo-labeling기반 domain adaptation에서는 pseudo-label이 noisy하기 때문에 $q(k|x)$ 가 ground truth distribution이 아님
    - Prediction인 $p(k|x)$ 또한 true distribution을 일정 부분 반영함
    - 따라서 noisy한 환경에서는 cross entropy가 양방향 모두 고려하는 것이 필요
  - Total loss = source domain CE + target domain SCE + KL(structure learning) + regularization

$$\ell_{sce}^{t} = \alpha \ell_{ce}(p_t, \hat{y}_t) + \beta \ell_{ce}(\hat{y}_t, p_t) \qquad \ell_{total} = \ell_{ce}^{s} + \ell_{sce}^{t} + \gamma_1 \ell_{kl}^{t} + \gamma_2 \ell_{reg}^{t}.$$

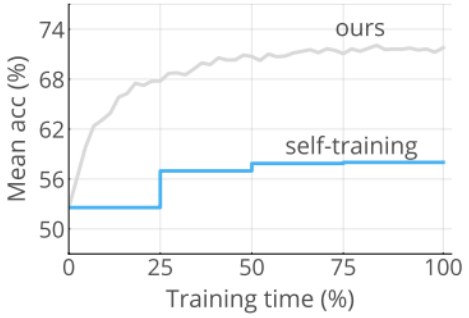
Pseudo-labeling based

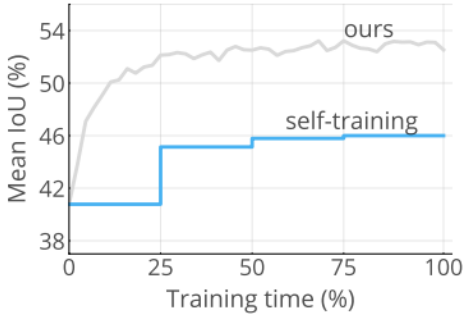
- ProDA
  - Distillation
    - 앞서 나온 total loss로 학습 후 학습된 모델을 같은 구조의, self-supervised pretrained된 student model로 distillation
    - Student model의 target images에 대한 prediction과 teacher model의 prediction의 KL divergence minimization
    - Teacher model에서 생성된 pseudo-label $\xi(p_t)$과 student model prediction $p_t^\dagger$로 cross entropy loss 적용
    - 마지막으로 source domain에도 계속 성능을 유지하기위해 학습 진행
    - 실제 적용시는 distillation을 여러 번 진행하여 성능을 더욱 끌어올림

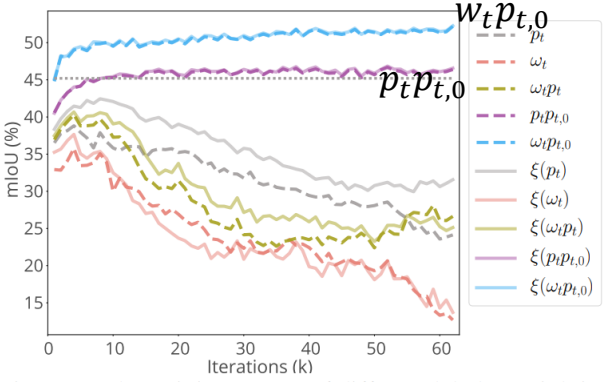
$$\ell_{\text{KD}} = \ell_{ce}^{s}(p_s, y_s) + \ell_{ce}^{t}(p_t^\dagger, \xi(p_t)) + \beta \mathbf{KL}(p_t \| p_t^\dagger),$$

Pseudo-labeling based

- ProDA
  - Experiments
    - 기존 self-training 기반 학습에서는 pseudo-label이 stage 마다 업데이트
      - 따라서 성능이 계단식으로 상승하지만 ProDA에서는 학습 내내 업데이트되므로 성능이 학습 과정동안 계속해서 점진적으로 상승
    - $p_{t,0}$로 고정함으로써 pseudo-label이 초기 prediction에서 크게 벗어나지 않게 고정
      - 모델과 함께 업데이트 되면서 trivial solution을 내놓지 않게 됨
      - p_t로 reweighting하는 것 보다, prototypical reweighting이 5%가량 더 높은 성능







Pseudo-labeling based

- ProDA
  - Experiments

	components				mIoU	gain
initialization	source				36.6	
	warm up				41.6	+5.0
	self training	sce	prototypical denoising	structure learning	mIoU	gain
stage 1	✓				45.2	+8.6
	✓	✓			45.6	+9.0
	✓	✓	✓		52.3	+15.7
	✓	✓		✓	47.6	+11.0
	✓	✓	✓	✓	53.7	+17.1
	self distill.	stage 1 init.	supervised init.	self-supervised init.	mIoU	gain
stage 2				✓	55.8	+19.2
	✓	✓			56.3	+19.7
	✓		✓		55.7	+19.1
	✓			✓	56.9	+20.3
stage 3	✓			✓	57.5	+20.9

Conclusion

this amazing performance improvments kill this area!!!!!!!!!! #7

Open **jingzhengli** opened this issue on 27 Apr 2021 · 1 comment

jingzhengli commented on 27 Apr 2021

this amazing performance improvements kill this area!!!!!!!!!!

8

panzhang0212 commented on 28 Apr 2021 Collaborator

I think there is still a large gap with full supervised learning.

1. recent self-supervised learning could get comparable results with supervised learning, why UDA is still worse than full supervised learning.
2. ProDA follows self-training. Another solution for UDA is gap minimization, which is more general and needs more works to improve.

5