Introduction to Test-time Adaptation

DMQA Open Seminar

2025. 04. 18

Data Mining & Quality Analytics Lab.

박소연

발표자 소개

❖ **박소연 (Soyeon Park)**

- 고려대학교 산업경영공학과 대학원 재학
- Data Mining & Quality Analytics Lab. (김성범 교수님)
- M.S Student (2024.09 ~ Present)

❖ **Research Interest**

- Test-Time Learning
- Federated Learning
- AI Agent LLM

❖ **Contact**

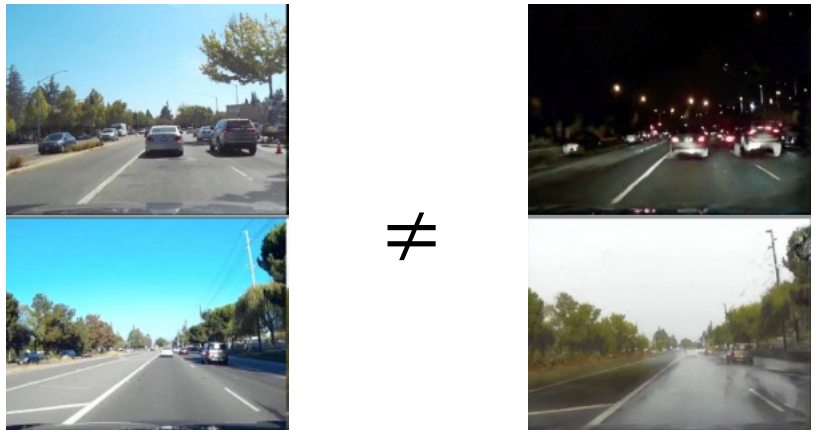
- syeonpark@korea.ac.kr

Test-time Adaptation

❖ **Domain Shift**

- 훈련 데이터 분포(Source) ≠ 테스트 데이터 분포(Target) (non-i.i.d)
- 기본적으로 모델 학습 시 i.i.d 상황을 가정하므로, Domain Shift가 발생한다면 모델 성능이 크게 저하됨

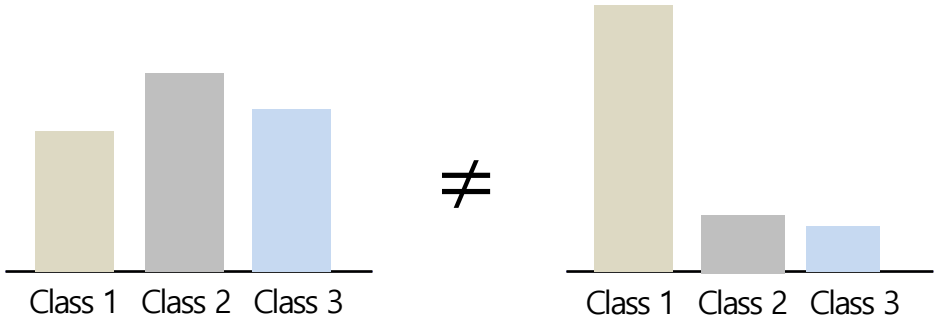
Source Target



≠

Feature shift
(=covariate shift)

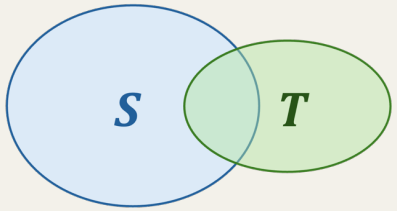
Source Target

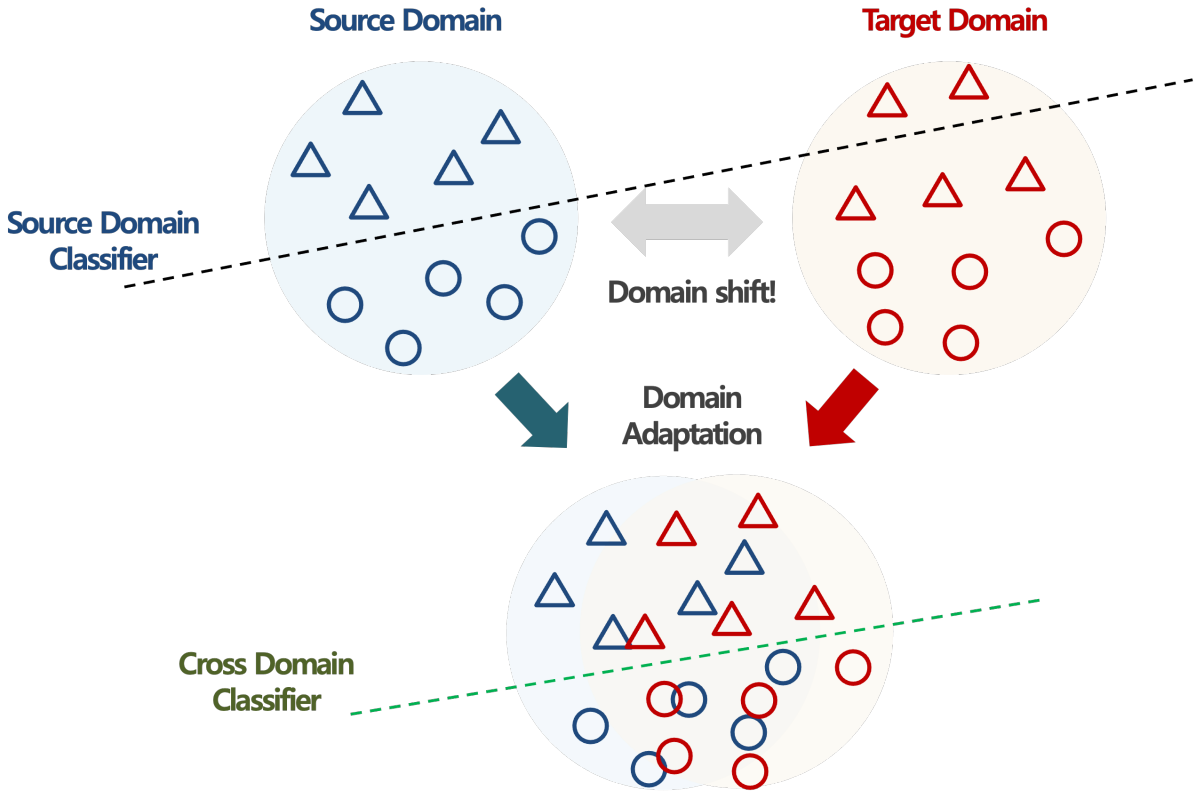


Label shift

Test-time Adaptation

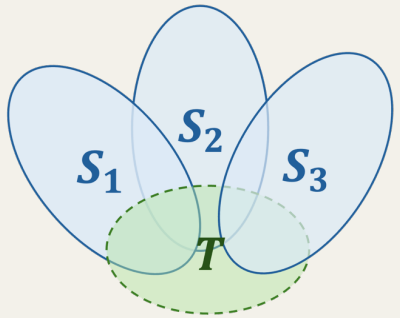
❖ **Domain Shift 문제를 해결하기 위한 방법론**

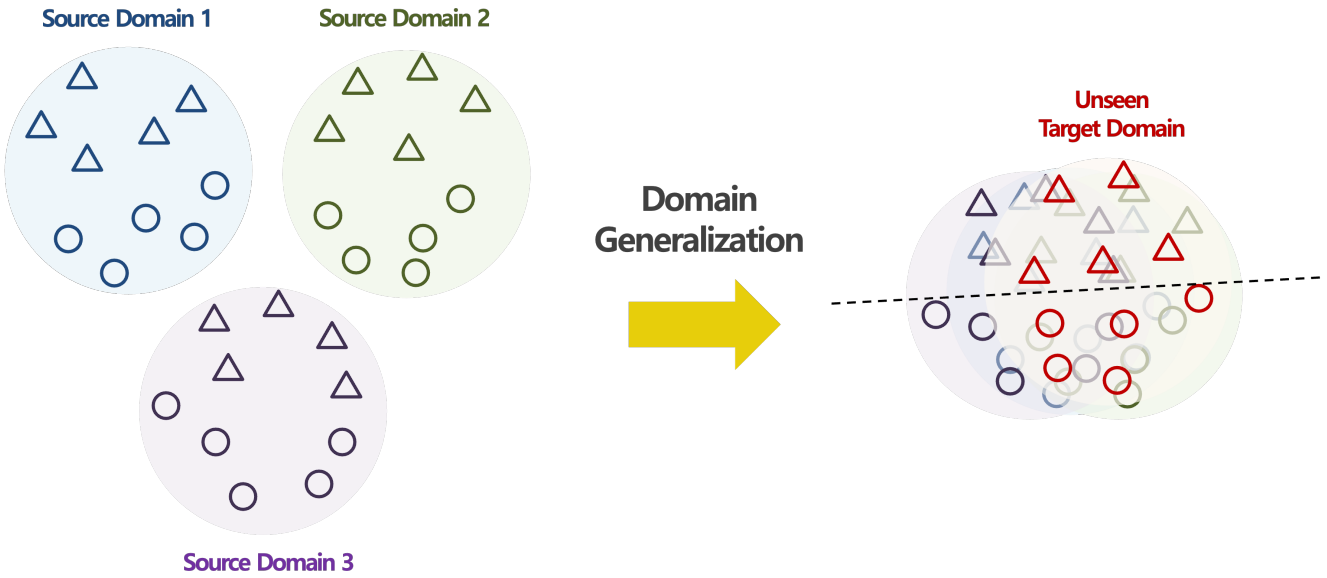
		Domain Adaptation (DA)
Data	Source	O
	Target	O
도메인 적응 방식		소스도메인과 타겟 도메인의 분포 차이를 줄임
		



Test-time Adaptation

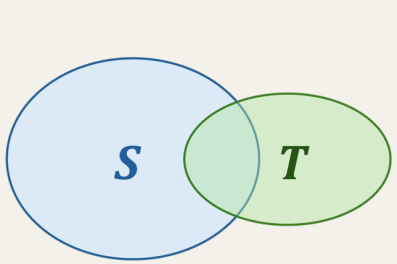
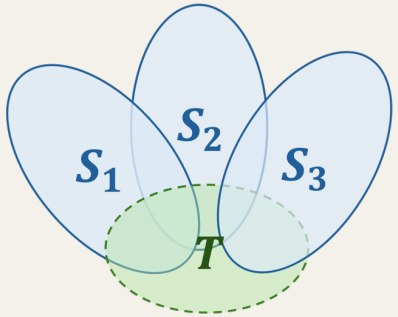
❖ **Domain Shift 문제를 해결하기 위한 방법론**

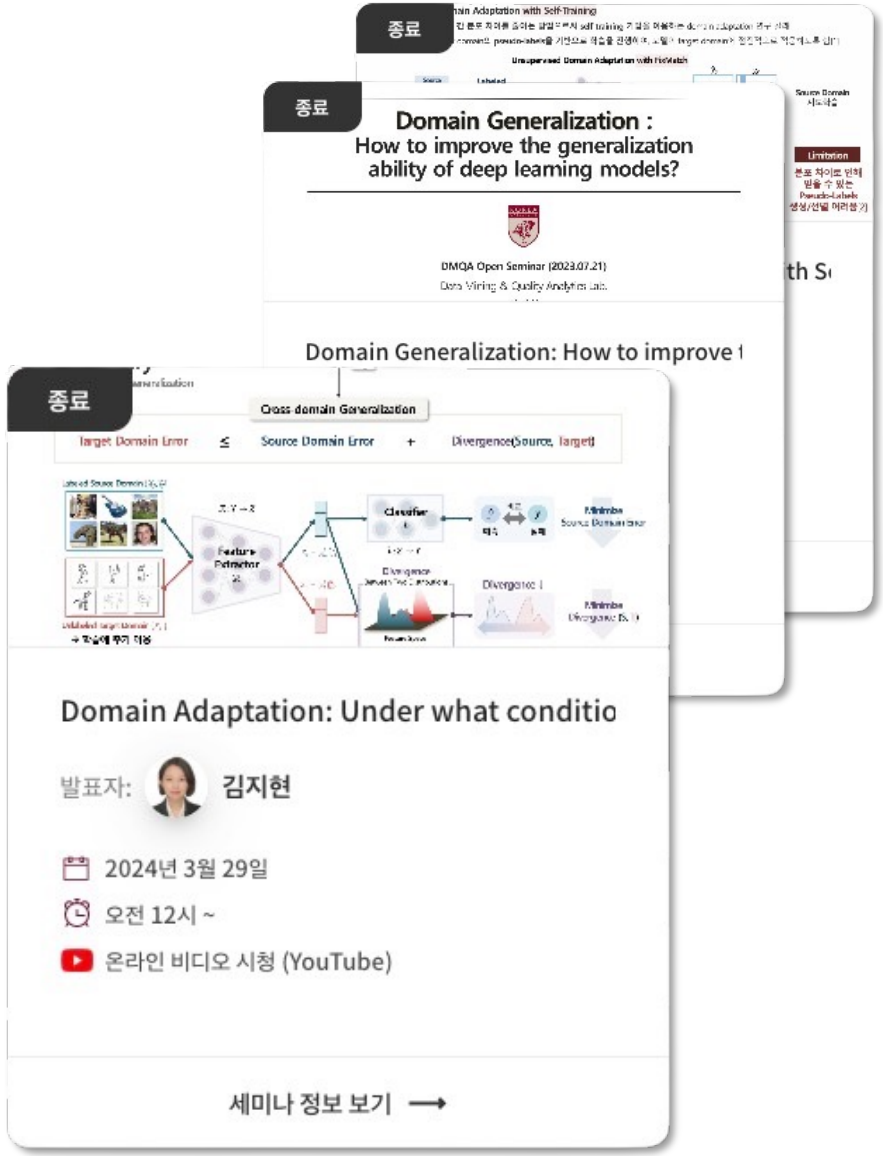
		Domain Generalization (DG)
Data	Source	O
	Target	X
도메인 적응 방식		다양한 소스 도메인을 일반화
		



Test-time Adaptation

❖ **Domain Shift 문제를 해결하기 위한 방법론**

		Domain Adaptation (DA)	Domain Generalization (DG)
Data	Source	O	O
	Target	O	X
도메인 적응 방식		소스도메인과 타겟 도메인의 분포 차이를 줄임	다양한 소스 도메인을 일반화
			



Test-time Adaptation

❖ **Privacy Issues**

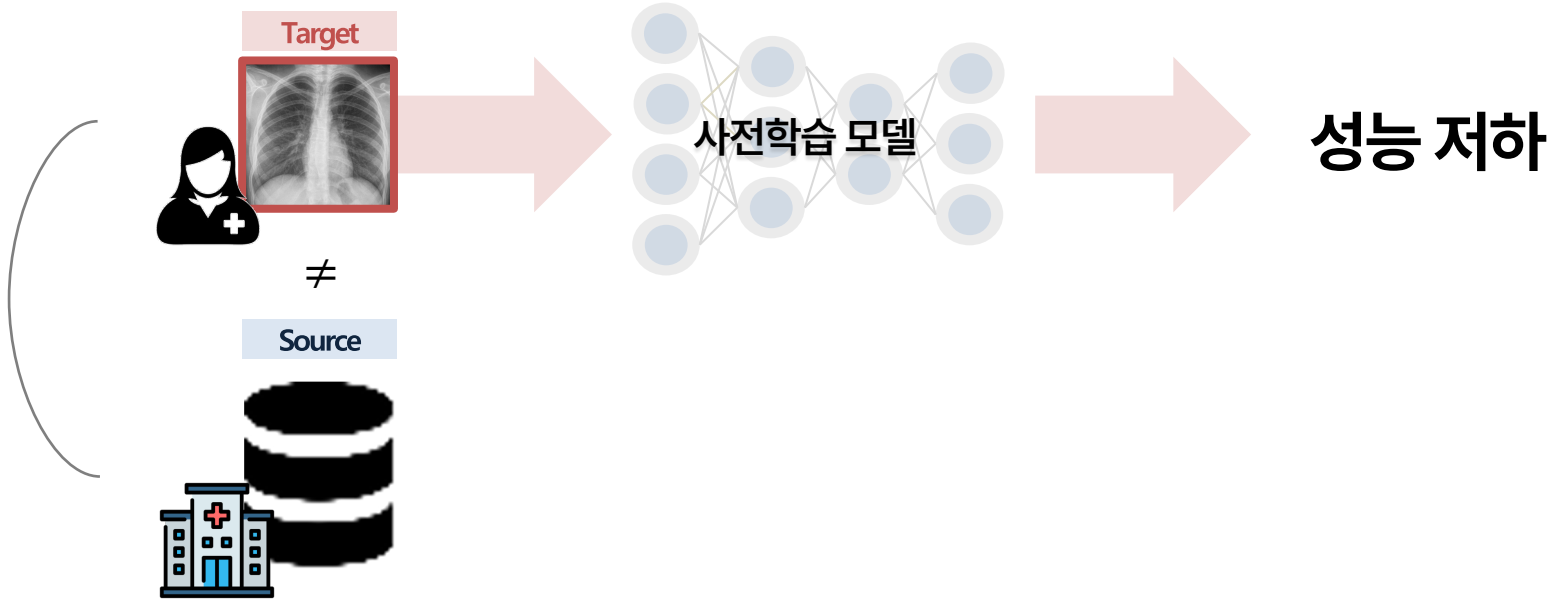
- 실제 환경에서는 개인정보 보호 이슈 등으로 인하여 Source data에 접근이 불가능한 경우가 많음

  → 보다 현실적인 해결책이 필요함!

시간 흐름, 장비 교체, 등으로

테스트 시점의 데이터가

학습 데이터와 달라지는 경우 발생

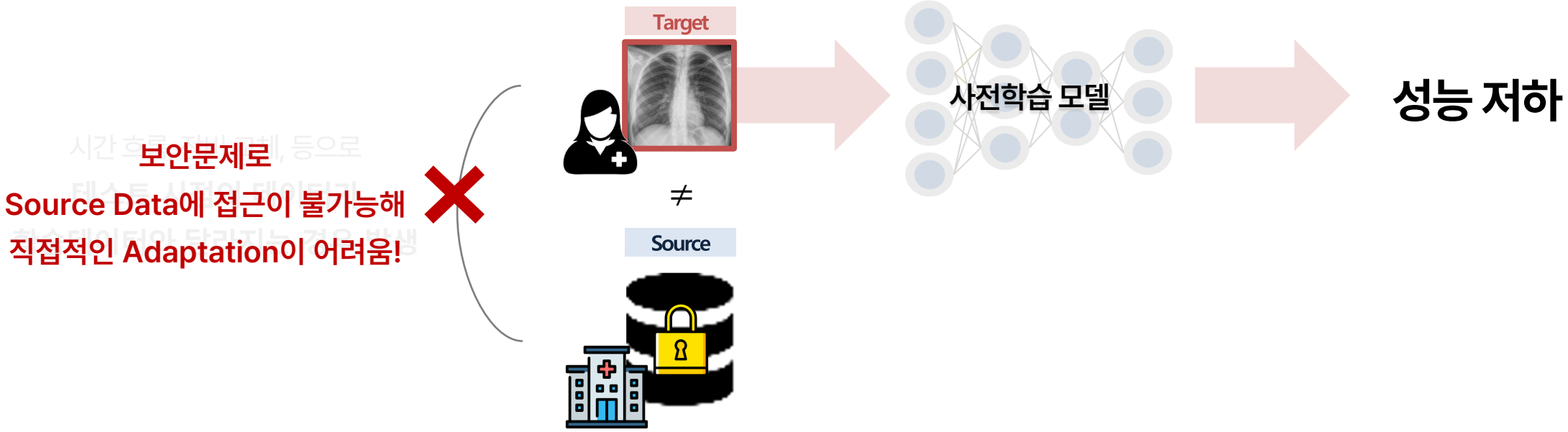


Test-time Adaptation

❖ Privacy Issues

- 실제 환경에서는 개인정보 보호 이슈 등으로 인하여 Source data에 접근이 불가능한 경우가 많음

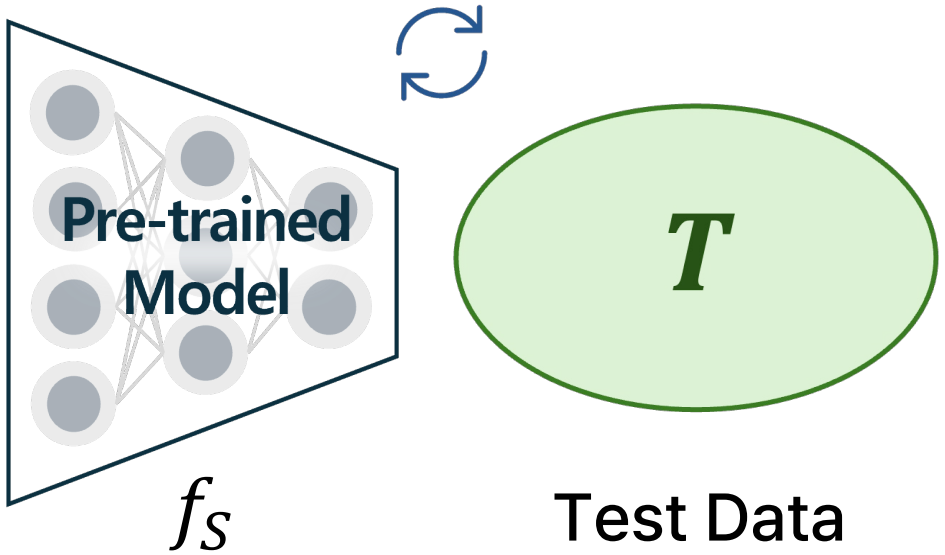
  → 보다 현실적인 해결책이 필요함!



Test-time Adaptation

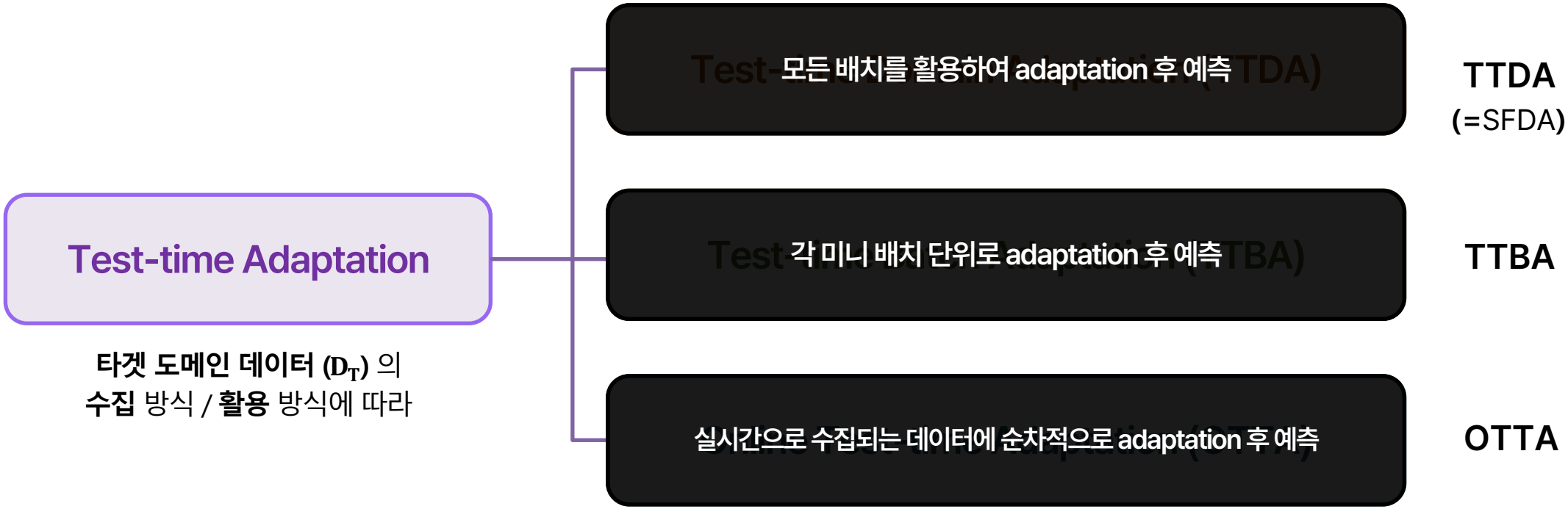
❖ Test-Time Adaptation이란?

- 문제 상황 : 1) 훈련 데이터 분포 ≠ 테스트 데이터 분포 (Domain shift), 2) Source Data 접근 불가
- 목표 : 테스트 시점에서 Source data 없이, 사전 학습된 모델을 Test data(unlabeled)에 맞게 조정하여 예측을 잘 할 수 있도록 만들자!

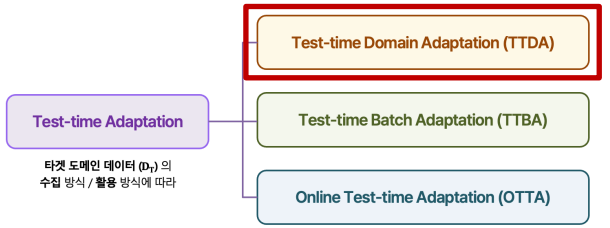


Test-time Adaptation

❖ **Taxonomy of TTA**

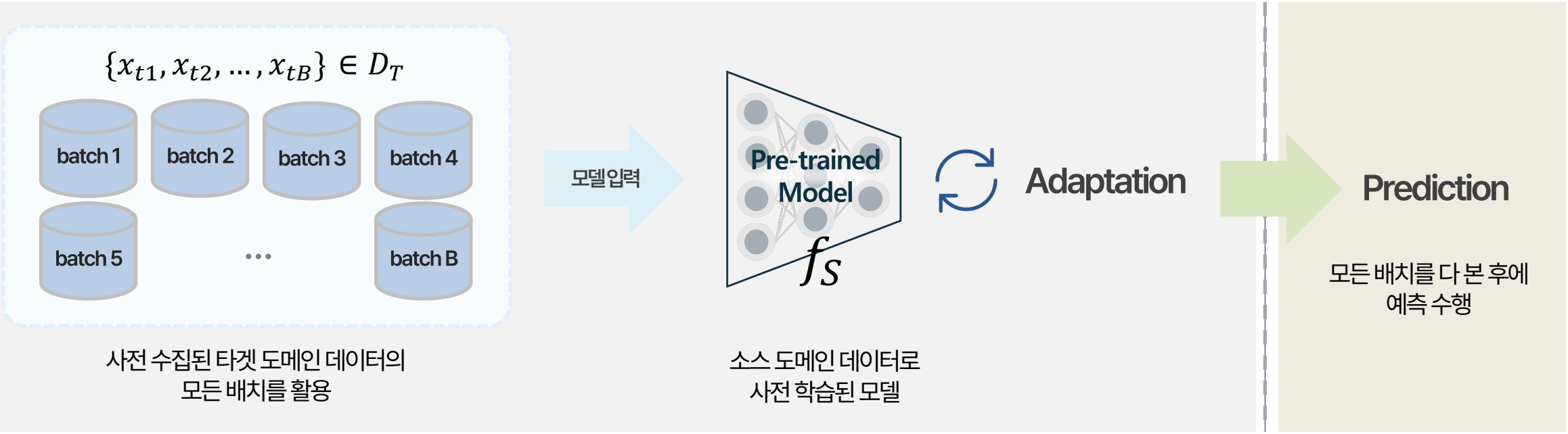


Test-time Adaptation

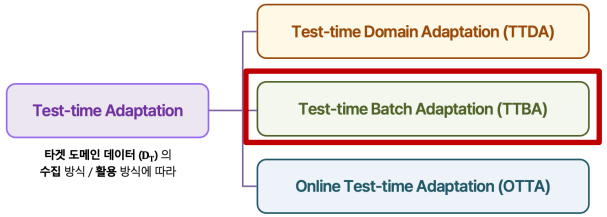


❖ TTDA (Test-Time Domain Adaptation, SFDA)

- 상황 : 타겟 도메인 데이터 X_T가 한 번에 모두 주어짐 (사전 수집된 상태)
- 적응 방식 : 타겟 도메인 데이터 X_T 의 모든 배치를 (전체) 활용하여 multi-epoch adaptation 수행 후 예측
- 특징
  - ✓ **장점** : 타겟 데이터 전체를 활용하므로 **더 많은 데이터를 고려할 수 있어 가장 효과적인 적응효과**를 볼 수 있음
  - ✓ **단점** : 계산 비용이 높고, **모든 테스트 데이터를 미리 확보해야 함**

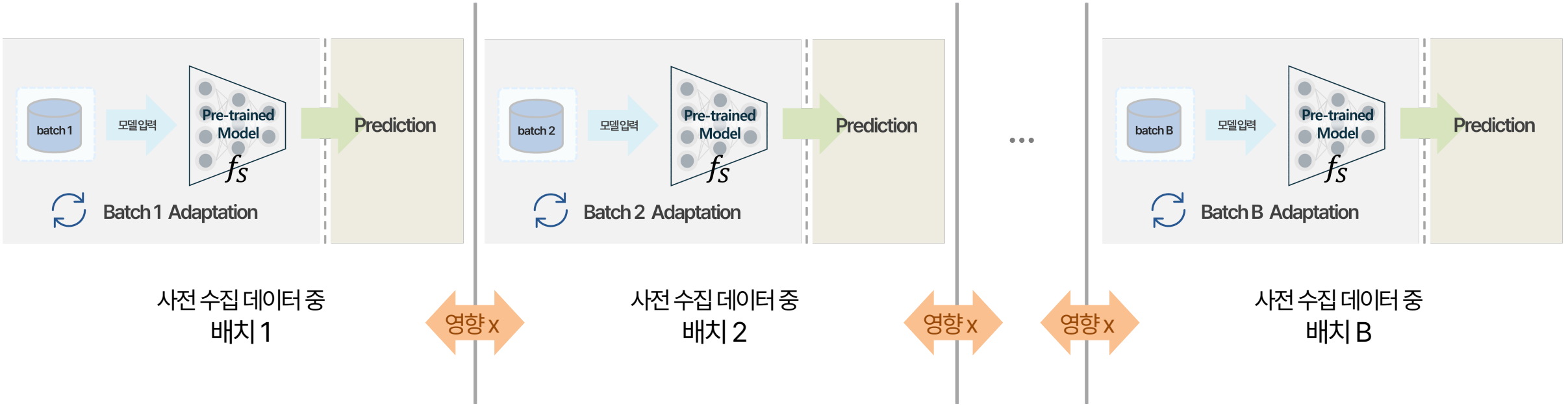


Test-time Adaptation

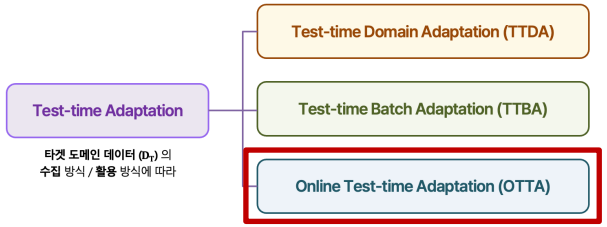


❖ TTDA (Test-Time Batch Adaptation)

- 상황 : 타겟 도메인 데이터 X_T가 배치 단위로 주어지는 상황
- 적응 방식 : 타겟 도메인 데이터 X_T 의 각 미니배치가 독립적으로 adaptation 수행 후 바로 예측 (이전 배치 정보 활용 X)
- 특징
  - ✓ **장점** : 각 배치마다 **빠르게 adaptation 가능**하여, 비교적 계산 비용이 낮음
  - ✓ **단점** : **이전 배치에서 학습한 정보**를 **다음 배치에서 활용할 수 없음**

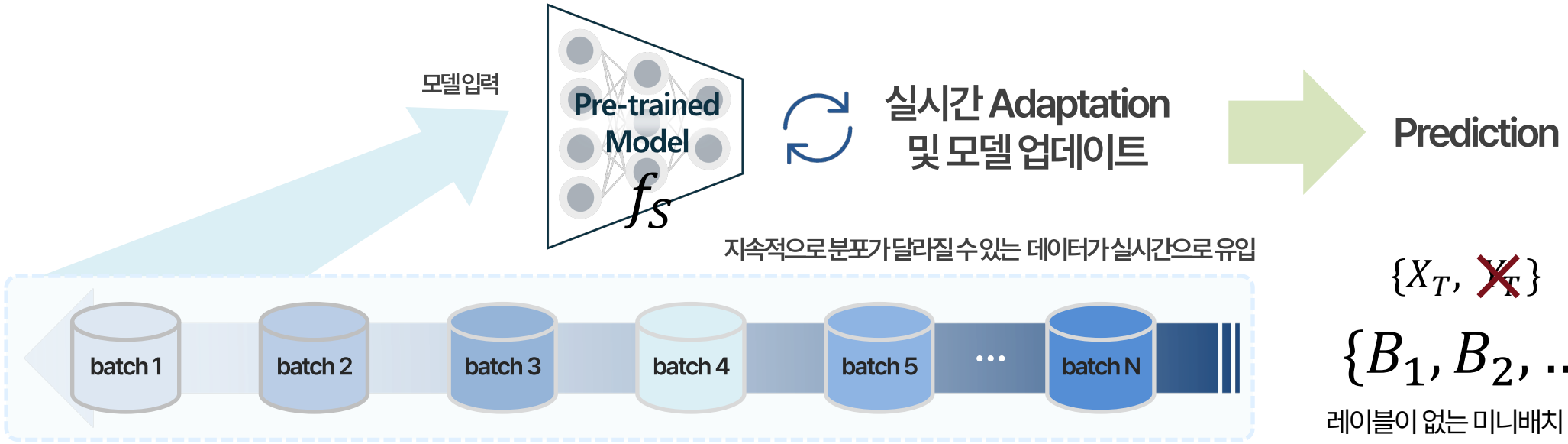


Test-time Adaptation



❖ OTTA (Online Test-Time Adaptation)

- 상황 : 타겟 도메인 데이터 X_T 가 시간에 따라 순차적으로 유입됨 (스트리밍 데이터)
- 적응 방식 : 배치가 들어올 때마다 이전 배치에서 학습된 내용을 누적(accumulate)하여 adaptation 수행 후 예측
- 특징
  - ✓ 장점 : 시간 흐름에 따라 변화하는 데이터 분포(distribution shift)에 대해 실시간으로 적응할 수 있음
  - ✓ 단점 : 오류 누적, Catastrophic Forgetting 문제 발생 우려



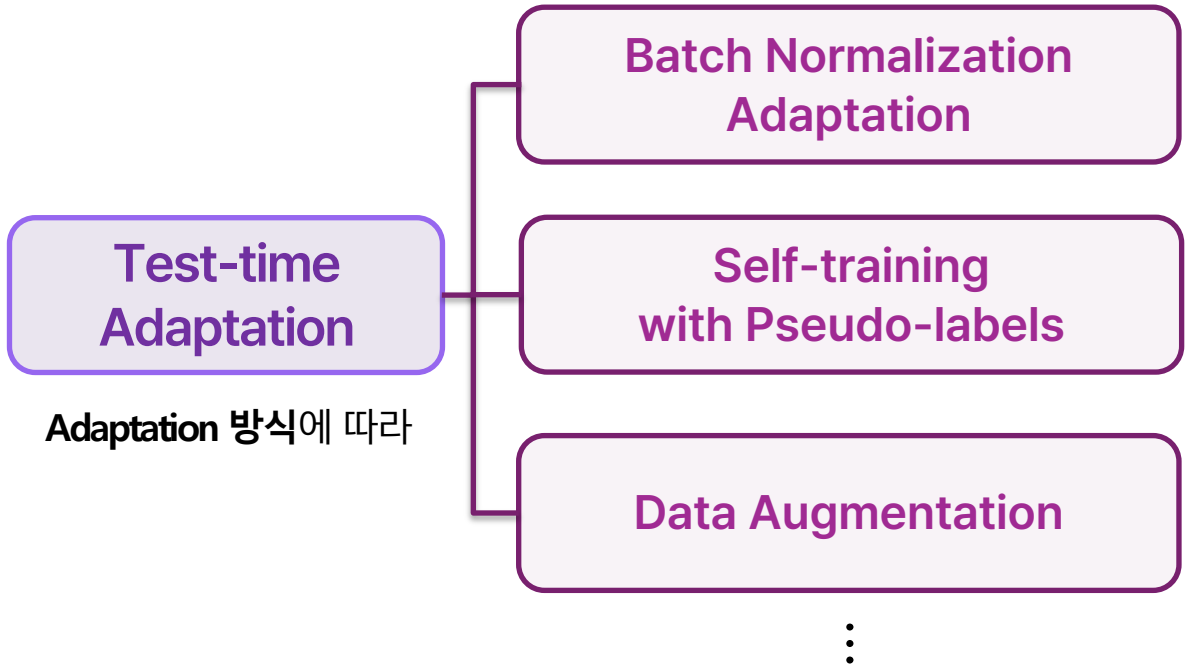
$\{X_T, \cancel{Y_T}\}$

$\{B_1, B_2, \dots\}$

레이블이 없는 미니배치 시퀀스

Test-time Adaptation

❖ **Taxonomy of TTA**



BN Layer의 통계 및 파라미터를 적응

테스트 데이터의 pseudo-label을 기반 지도학습을 통하여 적응

테스트 샘플을 증강하여, 보다 일반화된 방향으로 적응

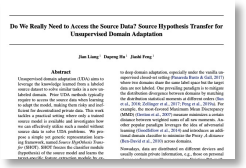
Test-time Adaptation

❖ **Taxonomy of TTA**

Test-time Adaptation

Adaptation 방식에 따라

- **Batch Normalization Adaptation**
- **Self-training with Pseudo-labels**
- **Data Augmentation**
- ⋮

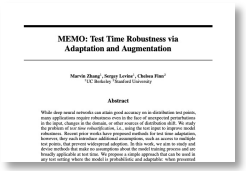


SHOT
(ICML 2020)

TTDA

TENT
(ICLR 2021)

TTBA



MEMO
(NeurIPS 2022)

TTBA

TTDA (SFDA)

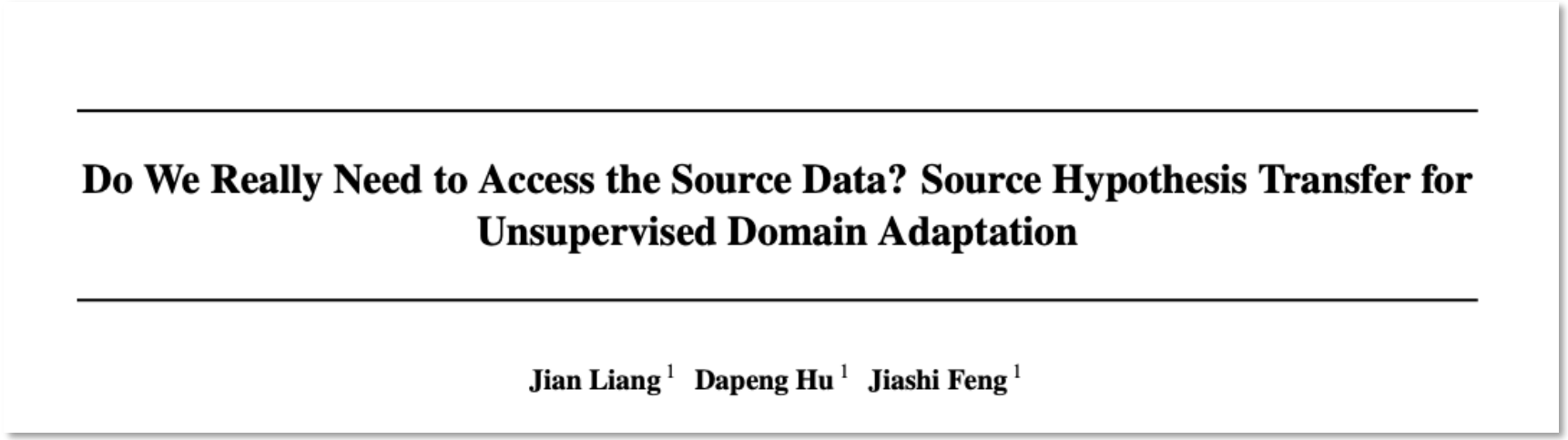
SHOT:
Do We Really Need to Access the Source Data? Source Hypothesis Transfer for Unsupervised Domain Adaptation (2020 ICML)

SHOT

TTDA

❖ **SHOT : Do we really need to Access the Source data? Source Hypothesis Transfer for UDA.**

- **SHOT (Source HypOthesis Tranfer)**
- **2020년 ICML, 인용 수 1577회**
- 소스 데이터 없이, **사전에 수집된 타겟 데이터**만으로 도메인 적응 가능
- classifier는 고정한 채, **1) Information Maximization, 2) Pseudo-Label 기반**으로 **Feature Extractor를 업데이트**하여 적응 수행



Do We Really Need to Access the Source Data? Source Hypothesis Transfer for Unsupervised Domain Adaptation

Jian Liang [1] Dapeng Hu [1] Jiashi Feng [1]

SHOT

❖ SHOT의 적응단계

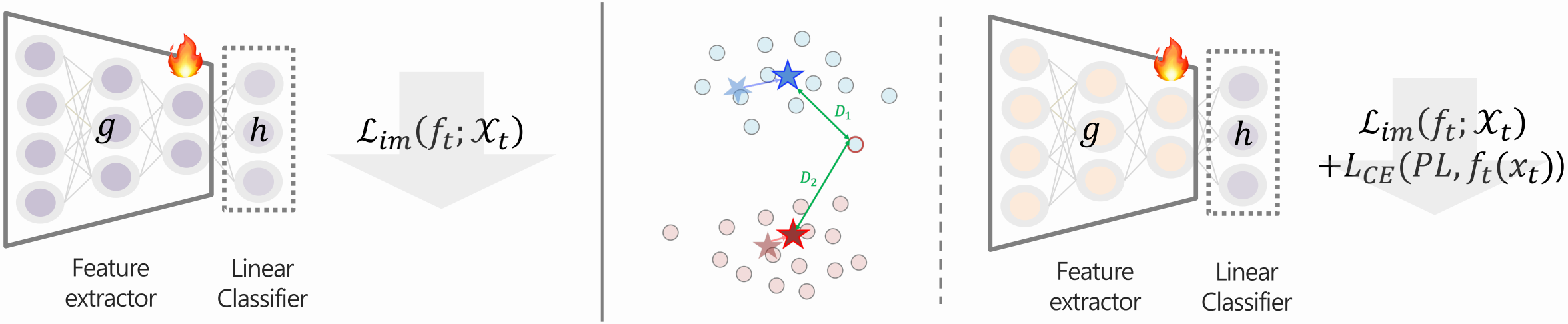
STEP 1. Source Hypothesis Transfer with Information Maximization (SHOT-IM)

STEP 2. Source Hypothesis Transfer with Self-supervised Learning

I. prototype based pseudo-labeling

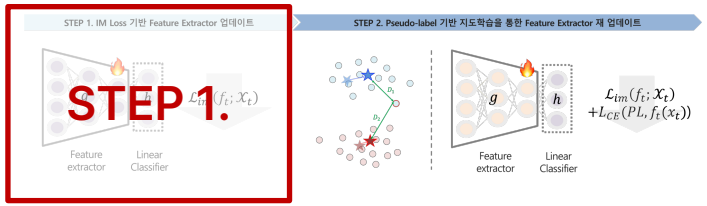
II. supervised learning with pseudo-label

STEP 1. IM Loss 기반 Feature Extractor 업데이트 | **STEP 2. Pseudo-label 기반 지도학습을 통한 Feature Extractor 재 업데이트**



*PL : Pseudo-Label

SHOT

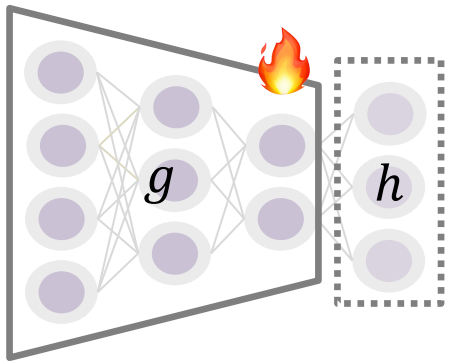


❖ SHOT의 적응단계

STEP 1. Source Hypothesis Transfer with Information Maximization (SHOT-IM)

- **목표 :** 소스 도메인에서 사전 학습된 feature extractor를 조정하여, **타겟 도메인의 입력이 Source hypothesis와 align되도록 유도**
- **방식 :** 사전에 수집된 unlabeled target data에 대하여, IM Loss가 최소화되는 방향으로 **feature extractor 업데이트**

학습 대상: g



h는 고정 (학습X)

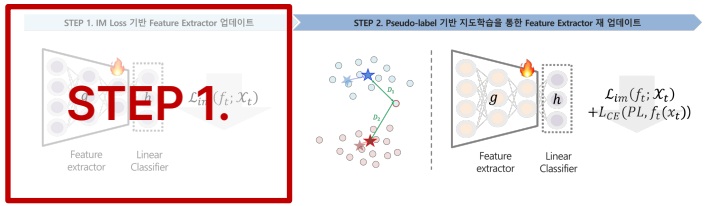
Feature extractor

Linear Classifier (=Source Hypothesis)

$$\mathcal{L}_{im}(f_t; \mathcal{X}_t)$$

IM Loss를 최소화하는 방향으로 Feature Extractor 업데이트!

SHOT

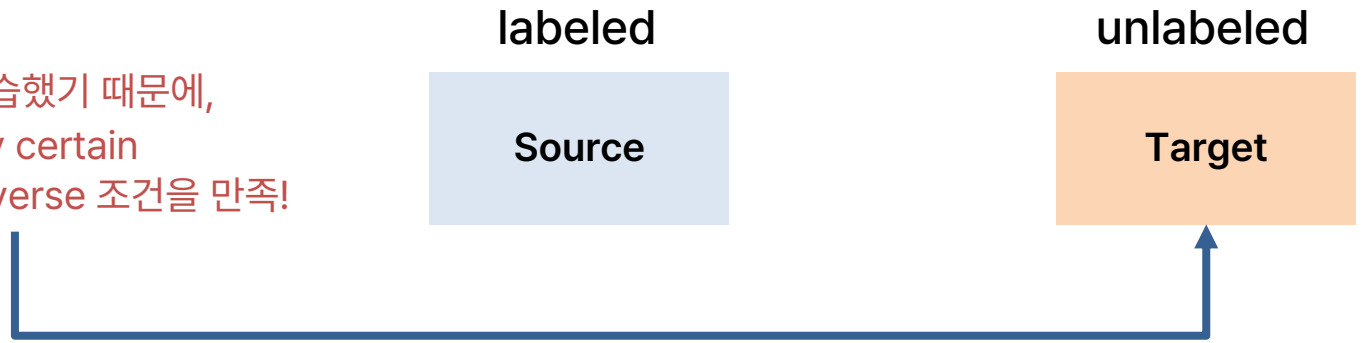


❖ Information Maximization Loss (IM Loss)

- **목적 :** 모델이 테스트 입력에 대해 **(1) individually certain, (2) globally diverse 한 예측**을 하도록 유도하기 위함

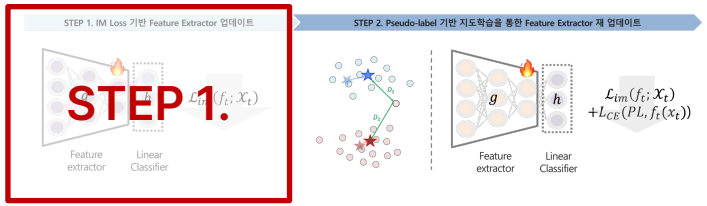
WHY?

지도학습으로 학습했기 때문에,
(1) individually certain
(2) globally diverse 조건을 만족!



Target Domain도 그렇게 하자!

SHOT

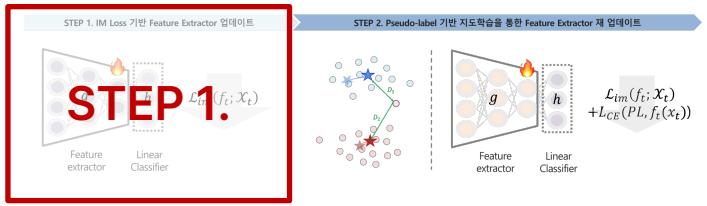


❖ Information Maximization Loss (IM Loss)

- **목적 :** 모델이 테스트 입력에 대해 **(1) individually certain, (2) globally diverse 한 예측**을 하도록 유도하기 위함

$$\mathcal{L}_{im}(f_t; \mathcal{X}_t) = \mathcal{L}_{div}(f_t; \mathcal{X}_t) + \mathcal{L}_{ent}(f_t; \mathcal{X}_t)$$

SHOT



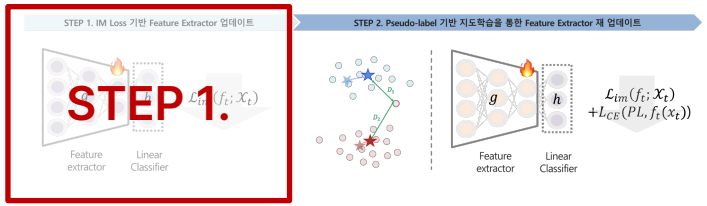
❖ Information Maximization Loss (IM Loss)

- **목적 :** 모델이 테스트 입력에 대해 **(1) individually certain, (2) globally diverse 한 예측**을 하도록 유도하기 위함

$$\mathcal{L}_{im}(f_t; \mathcal{X}_t) = \mathcal{L}_{ada}(\mathcal{X}_t) + \mathcal{L}_{ent}(\mathcal{X}_t)$$

다양성 (2) 엔트로피 (1)

SHOT



❖ Information Maximization Loss (IM Loss)

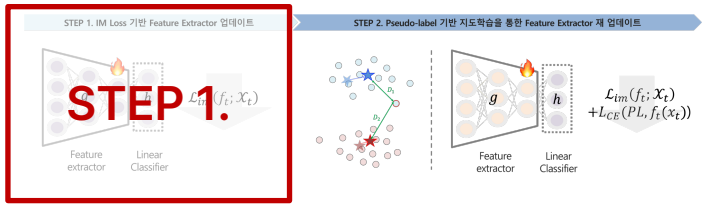
- **목적 :** 모델이 테스트 입력에 대해 **(1) individually certain, (2) globally diverse 한 예측**을 하도록 유도하기 위함

$$\mathcal{L}_{im}(f_t; \mathcal{X}_t) = \mathcal{L}_{다양성}(\mathcal{X}_t) + \mathcal{L}_{엔트로피}(\mathcal{X}_t)$$

Q. Why minimize entropy?

A. Unlabeled target data에 대한 모델의 예측 확실성을 높이기 위해!

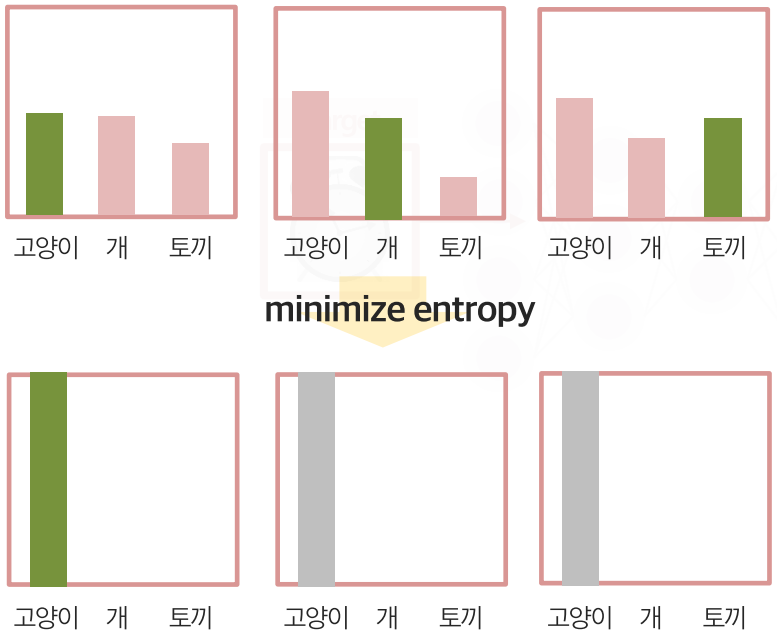
SHOT



❖ Information Maximization Loss (IM Loss)

- **목적 :** 모델이 테스트 입력에 대해 **(1) individually certain, (2) globally diverse 한 예측**을 하도록 유도하기 위함

$$\mathcal{L}_{im}(f_t; \mathcal{X}_t) = \mathcal{L}_{\text{다양성}}(\mathcal{X}_t) + \mathcal{L}_{\text{엔트로피}}(\mathcal{X}_t)$$



Q. Why minimize entropy?

A. Unlabeled target data에 대한 모델의 예측 확실성을 높이기 위해!

Q. Is minimizing entropy always good?

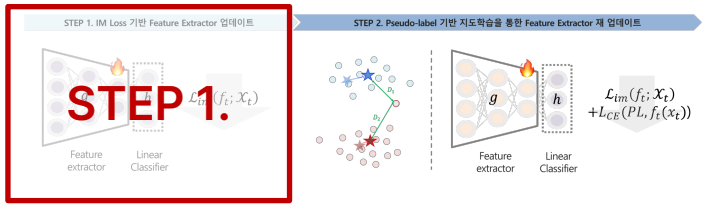
A. No!

Label이 없는 상황에서 모델은 엔트로피를 낮추기 위해,
가장 쉬운 방법(*trivial solution)으로 예측하려고 할 것.

→ 기존 타겟 도메인의 다양한 클래스 분포를 무시하고, 특정 클래스로 예측이 쏠리는 문제 발생

다양성을 보장하는 Loss term이 필요하다!

SHOT



❖ Information Maximization Loss (IM Loss)

- **목적 :** 모델이 테스트 입력에 대해 **(1) individually certain, (2) globally diverse 한 예측**을 하도록 유도하기 위함

$$\mathcal{L}_{im}(f_t; \mathcal{X}_t) = \text{다양성} + \text{엔트로피}$$

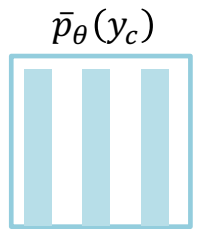
the average label distribution in the target domain

$$\bar{p}_\theta(y_c) = \frac{1}{n_t}\sum_{i=1}^{n_t} p_\theta(y_c|x_i)$$

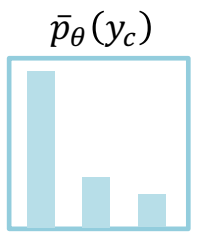
전체 데이터셋에 대하여,
클래스 C에 대한 예측 확률의 평균

→ **전체 데이터셋에 대해서,**
해당 클래스가 얼마나 자주 예측 됐는가?

$$\sum_{c=1}^{C} \bar{p}_\theta(y_c) \log \bar{p}_\theta(y_c)$$

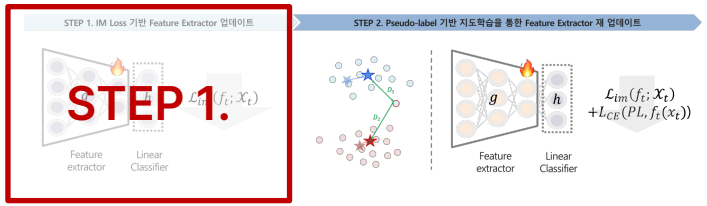


L_{div} ↑



L_{div} ↓

SHOT

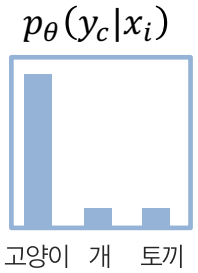


❖ Information Maximization Loss (IM Loss)

- **목적 :** 모델이 테스트 입력에 대해 **(1) individually certain, (2) globally diverse 한 예측**을 하도록 유도하기 위함

$$\mathcal{L}_{im}(f_t; \mathcal{X}_t) = \mathcal{L}_{div}(f_t; \mathcal{X}_t) + \mathcal{L}_{ent}(f_t; \mathcal{X}_t)$$

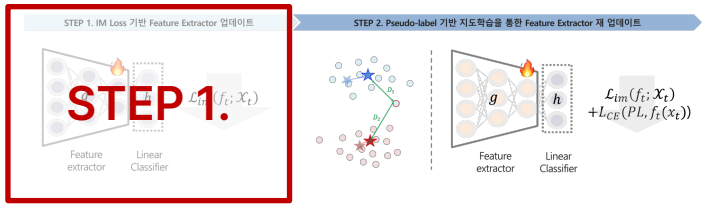
$$= \sum_{c=1}^{C} \bar{p}_\theta(y_c) \log \bar{p}_\theta(y_c) - \frac{1}{n_t} \sum_{i=1}^{n_t} \sum_{c=1}^{C} p_\theta(y_c|x_i) \log p_\theta(y_c|x_i)$$



전체 데이터셋에 대해서,
예측 확률 분포의 평균의 엔트로피는
uniform distribution에 가까워지게!

개별 샘플의 예측 확률 분포는 확신 있게!

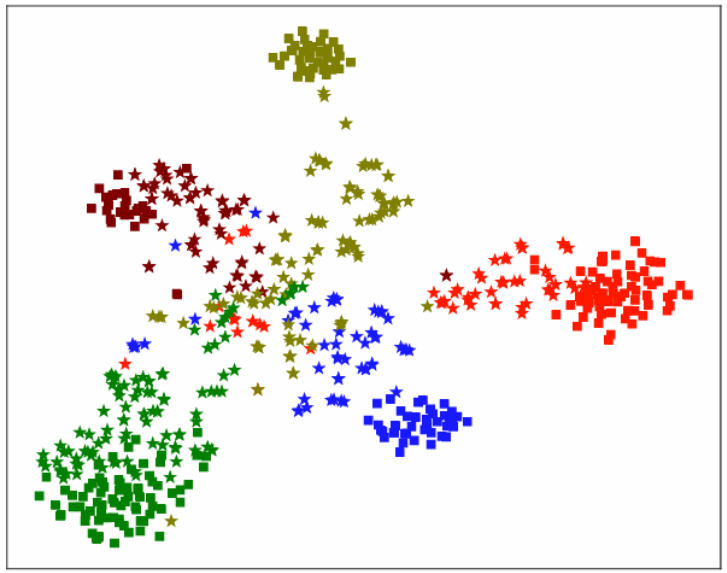
SHOT



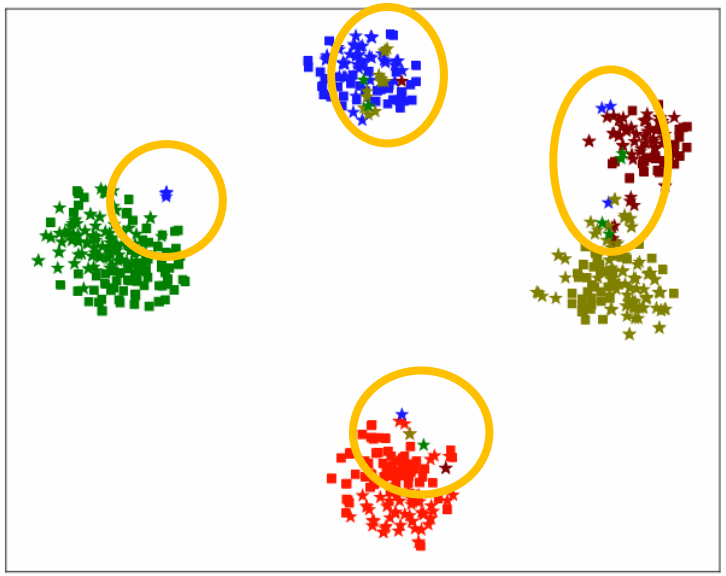
❖ SHOT의 적응단계

STEP 1. Source Hypothesis Transfer with Information Maximization (SHOT-IM)

✓ T-SNE 시각화 결과를 보면, Source model only 대비 SHOT-IM은 더 잘 뭉쳐져 있는 것을 볼 수 있음



(a) Source model only

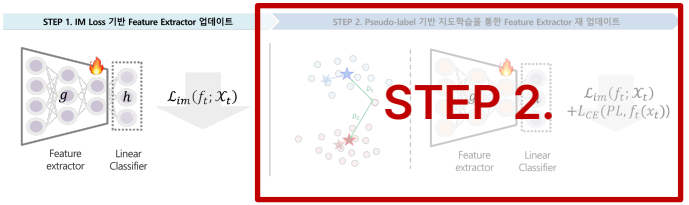


(b) SHOT-IM

하지만,
여전히 잘못 분류되는 데이터가 존재함!
아직 다 올바르게 분류되는 것은 아니다!

후속 단계에서
Self-supervised pseudo-labeling으로 보완 !

SHOT

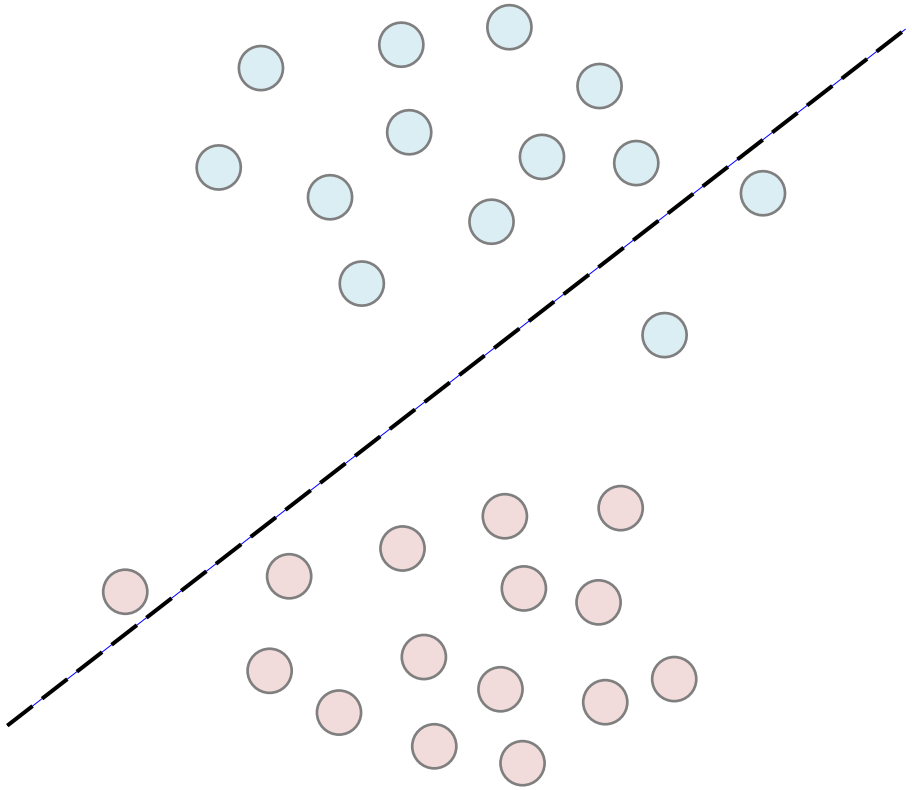


❖ SHOT의 적응단계

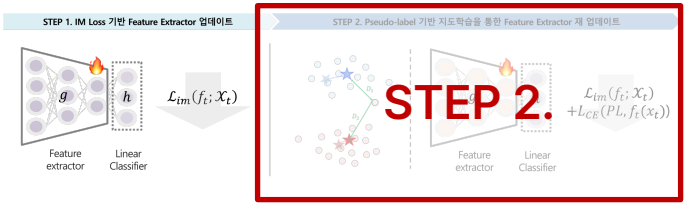
STEP 2. Source Hypothesis Transfer with Self-Supervised Pseudo-labeling

1) **Prototype based Pseudo-Labeling**

2) 생성된 Pseudo-label로 지도 학습을 통해 Feature extractor의 파라미터 재업데이트



SHOT

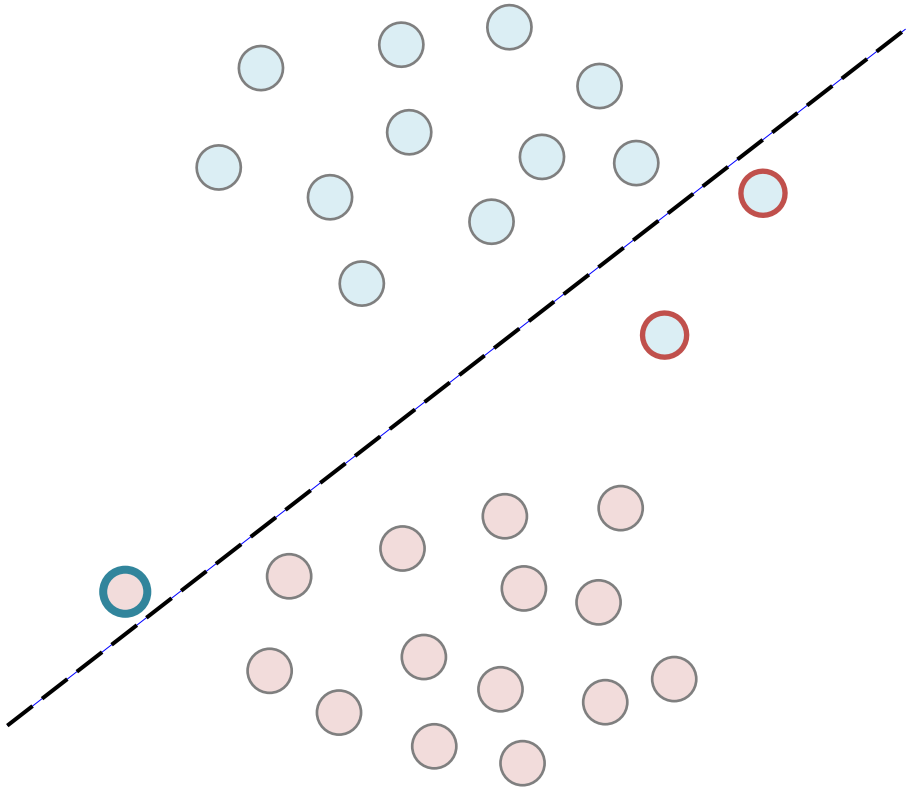


❖ SHOT의 적응단계

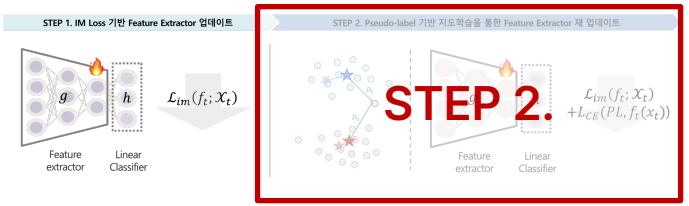
STEP 2. Source Hypothesis Transfer with Self-Supervised Pseudo-labeling

1) **Prototype based Pseudo-Labeling**

2) 생성된 Pseudo-label로 지도 학습을 통해 Feature extractor의 파라미터 재업데이트



SHOT



❖ SHOT의 적응단계

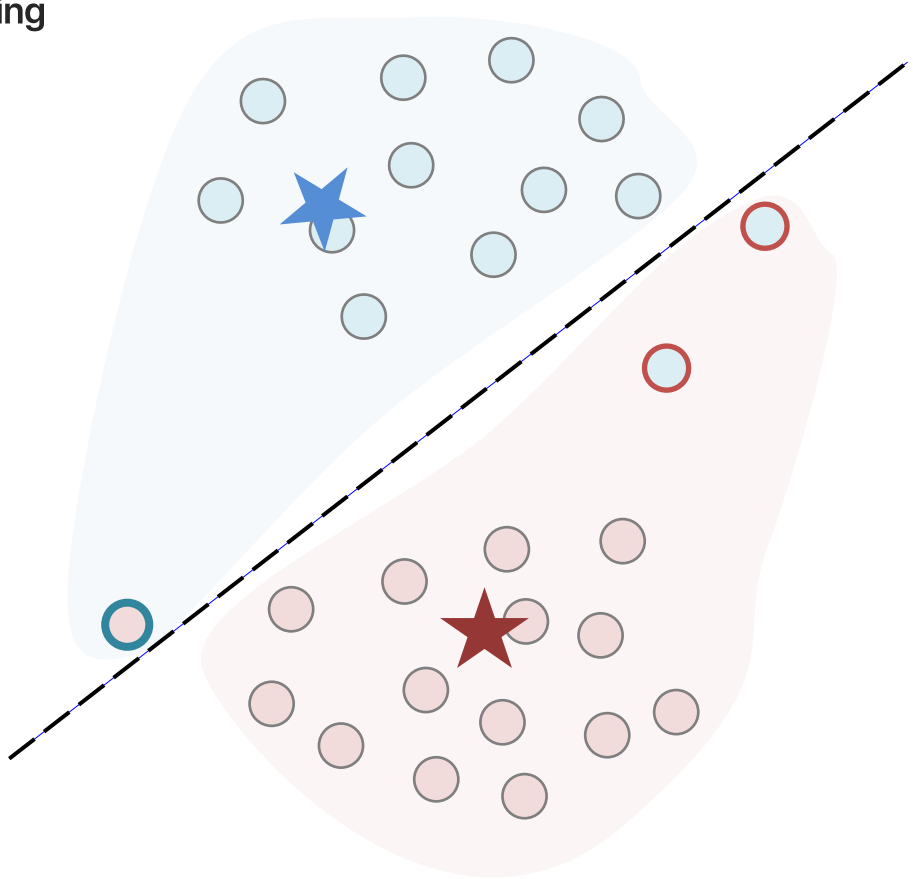
STEP 2. Source Hypothesis Transfer with Self-Supervised Pseudo-labeling

1) **Prototype based Pseudo-Labeling**

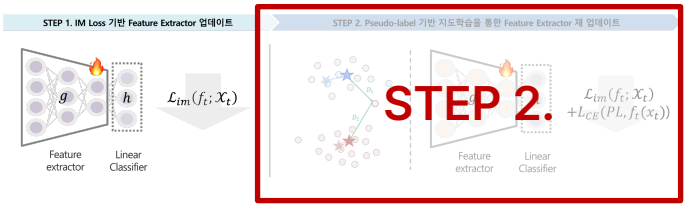
2) 생성된 Pseudo-label로 지도 학습을 통해 Feature extractor의 파라미터 재업데이트

Source model의 output을 soft label로 할당

초기 프로토타입 생성 $$c_k^{(0)} = \frac{\sum_{x \in \mathcal{X}_t} \hat{f}_t(x) \hat{g}_t(x)}{\sum_{x \in \mathcal{X}_t} \hat{f}_t(x)}$$



SHOT



❖ SHOT의 적응단계

STEP 2. Source Hypothesis Transfer with Self-Supervised Pseudo-labeling

1) **Prototype based Pseudo-Labeling**

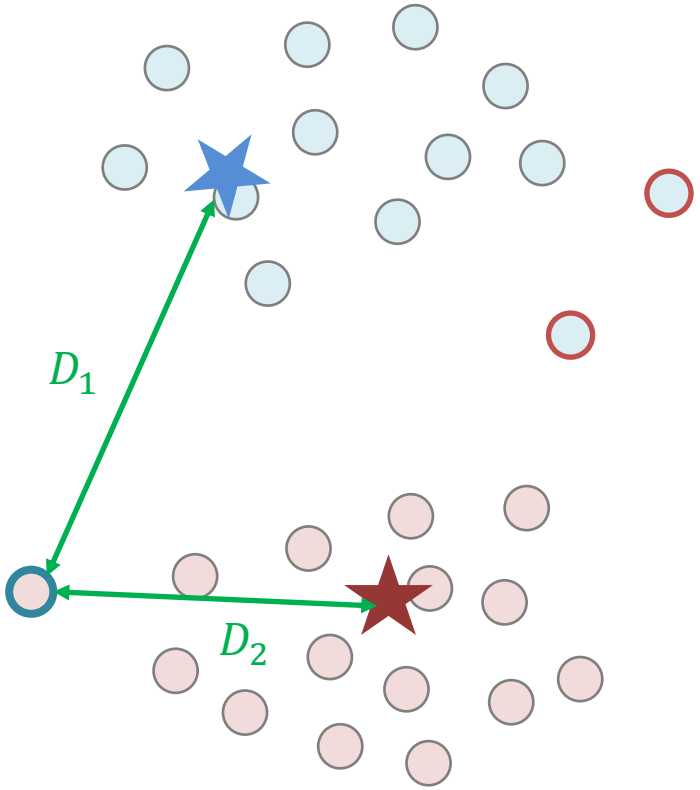
2) 생성된 Pseudo-label로 지도 학습을 통해 Feature extractor의 파라미터 재업데이트

Source model의 output을 soft label로 할당

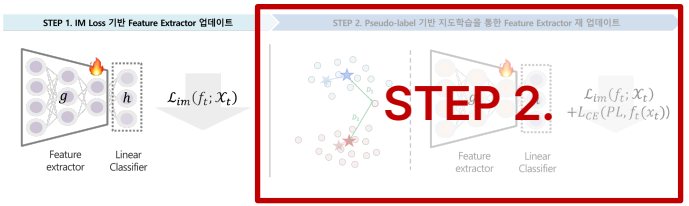
초기 프로토타입 생성 $c_k^{(0)} = \frac{\sum_{x \in \mathcal{X}_t} \hat{f}_t(x) \hat{g}_t(x)}{\sum_{x \in \mathcal{X}_t} \hat{f}_t(x)}$

$\hat{y}_t = \arg\min_k D_f(\hat{g}_t(x), c_k^{(0)})$

코사인 유사도 D를 이용한 label 재할당



SHOT



❖ SHOT의 적응단계

STEP 2. Source Hypothesis Transfer with Self-Supervised Pseudo-labeling

1) **Prototype based Pseudo-Labeling**

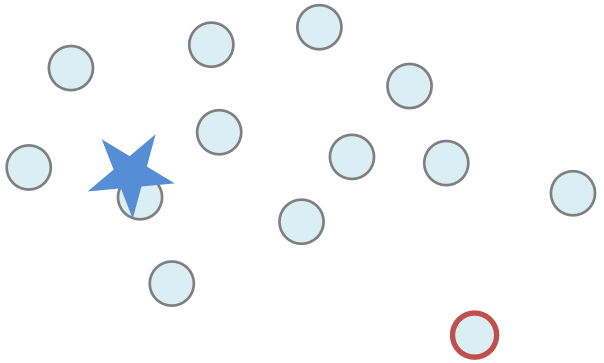
2) 생성된 Pseudo-label로 지도 학습을 통해 Feature extractor의 파라미터 재업데이트

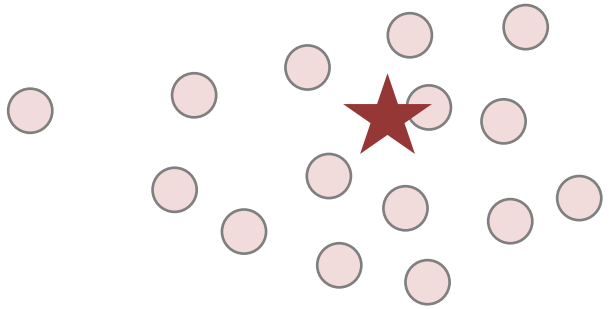
Source model의 output을 soft label로 할당

초기 프로토타입 생성 $c_k^{(0)} = \frac{\sum_{x \in \mathcal{X}_t} \hat{f}_t(x) \hat{g}_t(x)}{\sum_{x \in \mathcal{X}_t} \hat{f}_t(x)}$

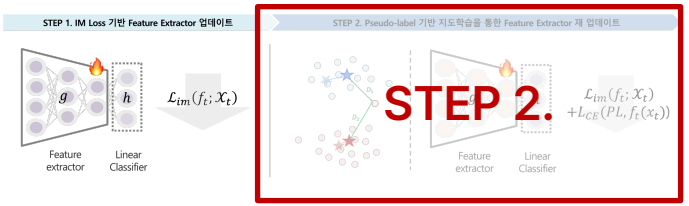
$$\hat{y}_t = \arg\min_k D_f(\hat{g}_t(x), c_k^{(0)})$$

코사인 유사도 D를 이용한 label 재할당





SHOT



❖ SHOT의 적응단계

STEP 2. Source Hypothesis Transfer with Self-Supervised Pseudo-labeling

1) **Prototype based Pseudo-Labeling**

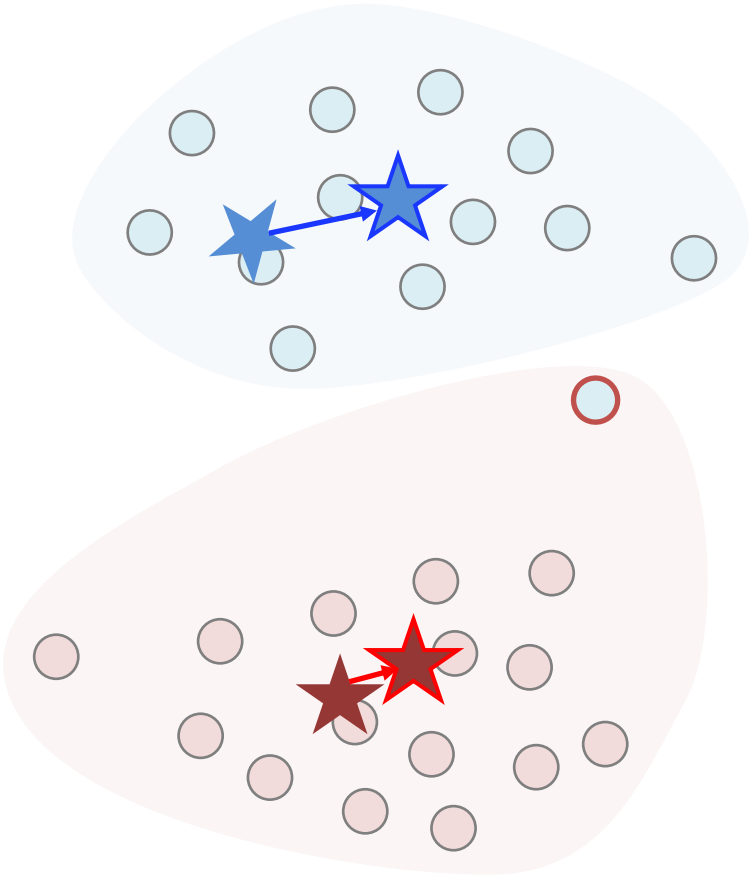
2) 생성된 Pseudo-label로 지도 학습을 통해 Feature extractor의 파라미터 재업데이트

Source model의 output을 soft label로 할당

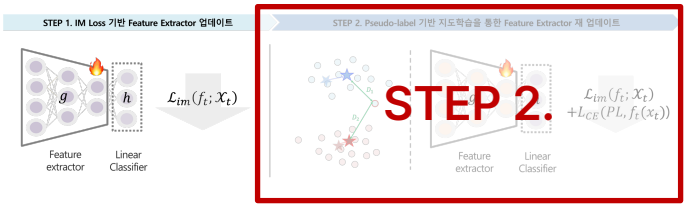
초기 프로토타입 생성 $c_k^{(0)} = \frac{\sum_{x \in \mathcal{X}_t} \hat{f}_t(x) \hat{g}_t(x)}{\sum_{x \in \mathcal{X}_t} \hat{f}_t(x)}$

$\hat{y}_t = \arg\min_k D_f(\hat{g}_t(x), c_k^{(0)})$

개선된 프로토타입 $c_k^{(1)} = \frac{\sum_{x \in \mathcal{X}_t} 1(\hat{y}_t = k) \hat{g}_t(x)}{\sum_{x \in \mathcal{X}_t} 1(\hat{y}_t = k)}$



SHOT



❖ SHOT의 적응단계

STEP 2. Source Hypothesis Transfer with Self-Supervised Pseudo-labeling

1) **Prototype based Pseudo-Labeling**

2) 생성된 Pseudo-label로 지도 학습을 통해 Feature extractor의 파라미터 재업데이트

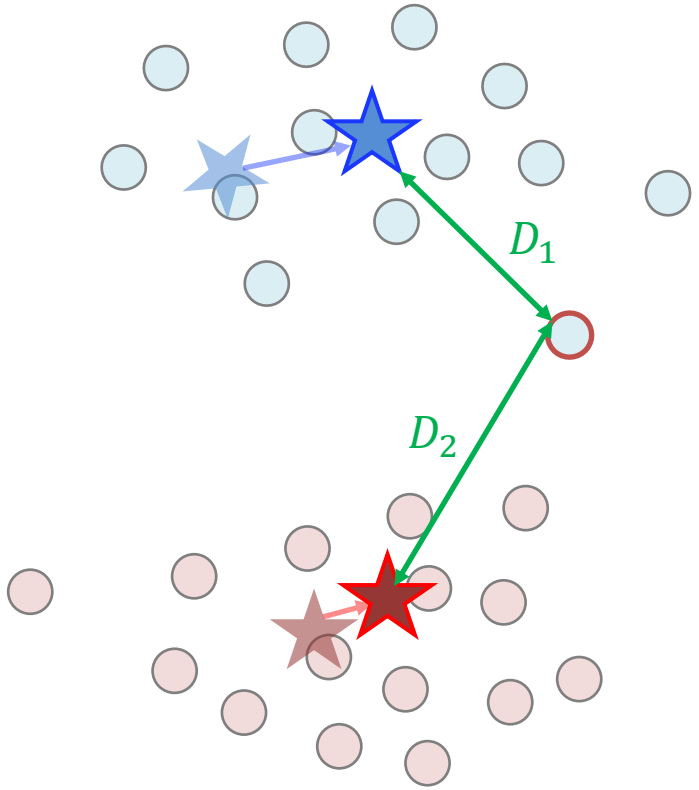
Source model의 output을 soft label로 할당

초기 프로토타입 생성 $c_k^{(0)} = \frac{\sum_{x \in \mathcal{X}_t} \hat{f}_t(x) \hat{g}_t(x)}{\sum_{x \in \mathcal{X}_t} \hat{f}_t(x)}$

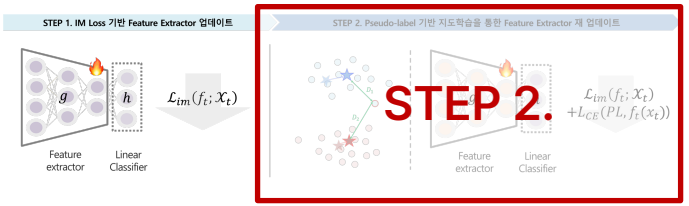
$\hat{y}_t = \arg\min_k D_f(\hat{g}_t(x), c_k^{(0)})$

개선된 프로토타입 $c_k^{(1)} = \frac{\sum_{x \in \mathcal{X}_t} 1(\hat{y}_t = k) \hat{g}_t(x)}{\sum_{x \in \mathcal{X}_t} 1(\hat{y}_t = k)}$

$\hat{y}_t = \arg\min_k D_f(\hat{g}_t(x), c_k^{(1)})$



SHOT



❖ SHOT의 적응단계

STEP 2. Source Hypothesis Transfer with Self-Supervised Pseudo-labeling

1) **Prototype based Pseudo-Labeling**

2) 생성된 Pseudo-label로 지도 학습을 통해 Feature extractor의 파라미터 재업데이트

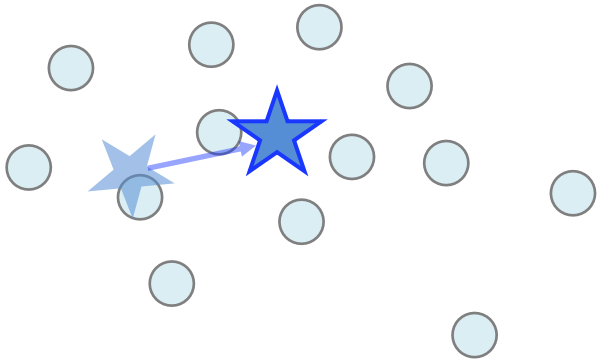
Source model의 output을 soft label로 할당

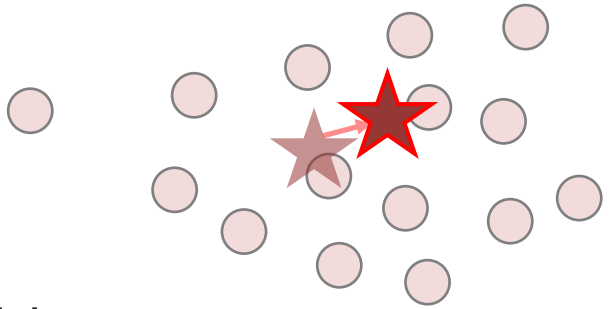
초기 프로토타입 생성 $c_k^{(0)} = \frac{\sum_{x \in \mathcal{X}_t} \hat{f}_t(x) \hat{g}_t(x)}{\sum_{x \in \mathcal{X}_t} \hat{f}_t(x)}$

$\hat{y}_t = \arg\min_k D_f(\hat{g}_t(x), c_k^{(0)})$

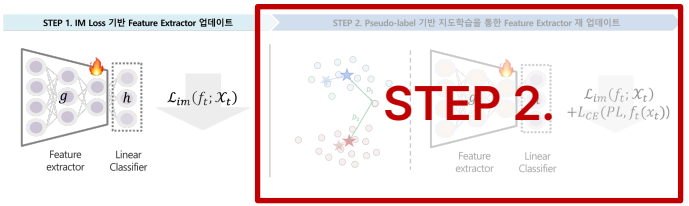
개선된 프로토타입 $c_k^{(1)} = \frac{\sum_{x \in \mathcal{X}_t} 1(\hat{y}_t = k) \hat{g}_t(x)}{\sum_{x \in \mathcal{X}_t} 1(\hat{y}_t = k)}$

$\hat{y}_t = \arg\min_k D_f(\hat{g}_t(x), c_k^{(1)})$ 최종 pseudo-label 생성!





SHOT



❖ SHOT의 적응단계

STEP 2. Source Hypothesis Transfer with Self-Supervised Pseudo-labeling

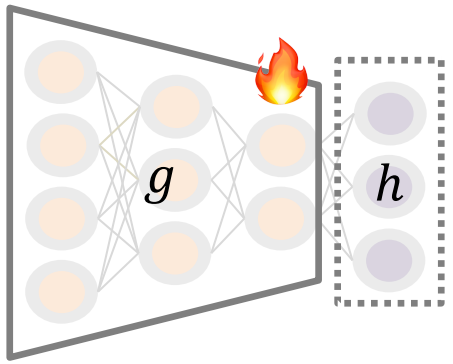
1) Prototype based Pseudo-Labeling

2) 생성된 Pseudo-label로 지도 학습을 통해 Feature extractor의 파라미터 재업데이트

$$L(f_t; X_t) = \mathcal{L}_{div}(f_t; X_t) + \mathcal{L}_{ent}(f_t; X_t) + L_{CE}(PL, f_t(X_t))$$

↓

생성된 Pseudo-Label 기반
지도학습을 위한 Cross Entropy



h는 고정

Feature extractor

Linear Classifier

$$L(f_t; X_t)$$

최종 Loss를 최소화하는 방향으로 Feature Extractor 재 업데이트!

*PL : Pseudo-Label

SHOT

❖ Results

- SHOT은 Digits, Office, Office-Home, VisDA-C등 다양한 벤치마크에서 평가됨
- Source Data없이도 SHOT은 DA방법론들에 비해 우수한 성능을 보임
  - Office-Home에서 기존 최고 성능을 향상 시킴 →12개 작업 중 10개에서 최고 성능 기록
  - VisDA-C(합성→실제) 에서 클래스별 최고 정확도 기록 → 특히 가장 어려운 'truck'클래스에서 우수한 성능

Table 4. Classification accuracies (%) on medium-sized **Office-Home** dataset for *vanilla closed-set DA* (ResNet-50).

Method (Source→Target)	Ar→Cl	Ar→Pr	Ar→Re	Cl→Ar	Cl→Pr	Cl→Re	Pr→Ar	Pr→Cl	Pr→Re	Re→Ar	Re→Cl	Re→Pr	Avg.
ResNet-50 (He et al., 2016)	34.9	50.0	58.0	37.4	41.9	46.2	38.5	31.2	60.4	53.9	41.2	59.9	46.1
DANN (Ganin & Lempitsky, 2015)	45.6	59.3	70.1	47.0	58.5	60.9	46.1	43.7	68.5	63.2	51.8	76.8	57.6
DAN (Long et al., 2015)	43.6	57.0	67.9	45.8	56.5	60.4	44.0	43.6	67.7	63.1	51.5	74.3	56.3
CDAN+E (Long et al., 2018)	50.7	70.6	76.0	57.6	70.0	70.0	57.4	50.9	77.3	70.9	56.7	81.6	65.8
CDAN+BSP (Chen et al., 2019)	52.0	68.6	76.1	58.0	70.3	70.2	58.6	50.2	77.6	72.2	59.3	81.9	66.3
SAFN (Xu et al., 2019)	52.0	71.7	76.3	64.2	69.9	71.9	63.7	51.4	77.1	70.9	57.1	81.5	67.3
CDAN+TransNorm (Wang et al., 2019)	50.2	71.4	77.4	59.3	72.7	73.1	61.0	53.1	79.5	71.9	59.0	82.9	67.6
Source model only	44.6	67.3	74.8	52.7	62.7	64.8	53.0	40.6	73.2	65.3	45.4	78.0	60.2
SHOT-IM (ours)	55.4	76.6	80.4	66.9	74.3	75.4	65.6	54.8	80.7	73.7	58.4	83.4	70.5
SHOT (full, ours)	57.1	78.1	81.5	68.0	78.2	78.1	67.4	54.9	82.2	73.3	58.8	84.3	71.8

Source Data에 접근이 필요한 DA계열

Source Data에 접근 필요 X

SHOT

❖ Results

- SHOT은 Digits, Office, Office-Home, VisDA-C등 다양한 벤치마크에서 평가됨
- Source Data없이도 SHOT은 DA방법론들에 비해 우수한 성능을 보임
  - Office-Home에서 기존 최고 성능을 향상 시킴 →12개 작업 중 10개에서 최고 성능 기록
  - VisDA-C(합성→실제) 에서 클래스별 최고 정확도 기록 → 특히 가장 어려운 'truck'클래스에서 우수한 성능

Table 5. Classification accuracies (%) on large-scale **VisDA-C** dataset for *vanilla closed-set DA* (ResNet-101).

Method (Synthesis → Real)	plane	bcycl	bus	car	horse	knife	mcycl	person	plant	sktbrd	train	truck	Per-class
ResNet-101 (He et al., 2016)	55.1	53.3	61.9	59.1	80.6	17.9	79.7	31.2	81.0	26.5	73.5	8.5	52.4
DANN (Ganin & Lempitsky, 2015)	81.9	77.7	82.8	44.3	81.2	29.5	65.1	28.6	51.9	54.6	82.8	7.8	57.4
DAN (Long et al., 2015)	87.1	63.0	76.5	42.0	90.3	42.9	85.9	53.1	49.7	36.3	85.8	20.7	61.1
ADR (Saito et al., 2018a)	94.2	48.5	84.0	72.9	90.1	74.2	92.6	72.5	80.8	61.8	82.2	28.8	73.5
CDAN (Long et al., 2018)	85.2	66.9	83.0	50.8	84.2	74.9	88.1	74.5	83.4	76.0	81.9	38.0	73.9
CDAN+BSP (Chen et al., 2019)	92.4	61.0	81.0	57.5	89.0	80.6	90.1	77.0	84.2	77.9	82.1	38.4	75.9
SAFN (Xu et al., 2019)	93.6	61.3	84.1	70.6	94.1	79.0	91.8	79.6	89.9	55.6	89.0	24.4	76.1
SWD (Lee et al., 2019a)	90.8	82.5	81.7	70.5	91.7	69.5	86.3	77.5	87.4	63.6	85.6	29.2	76.4
Source model only	60.9	21.6	50.9	67.6	65.8	6.3	82.2	23.2	57.3	30.6	84.6	8.0	46.6
SHOT-IM (ours)	93.7	86.4	78.7	50.7	91.0	93.5	79.0	78.3	89.2	85.4	87.9	51.1	80.4
SHOT (full, ours)	94.3	88.5	80.1	57.3	93.1	94.9	80.7	80.3	91.5	89.1	86.3	58.2	82.9

Source Data에 접근이 필요한 DA계열

Source Data에 접근 필요 X

TTBA – Batch-level

Tent: Fully Test-time Adaptation by Entropy Minimization (ICLR 2021)

TENT

TTBA

❖ TENT : Fully Test-Time Adaptation by Entropy Minimization

- **TENT (Test Entropy Minimization)**
- **2021년 ICLR, 인용 수 1318회**
- 소스 데이터 없이 **배치 단위**로 온라인 적응 가능
- 테스트 시점에서 입력된 배치 데이터만을 활용하여, **엔트로피를 최소화**하는 방향으로 **BN Layer의 affine 파라미터를 업데이트**하여 적응 수행

TENT: FULLY TEST-TIME ADAPTATION BY ENTROPY MINIMIZATION

Dequan Wang[1]*, **Evan Shelhamer**[2*†], **Shaoteng Liu**[1], **Bruno Olshausen**[1], **Trevor Darrell**[1]
dqwang@cs.berkeley.edu, shelhamer@google.com
UC Berkeley[1] Adobe Research[2]

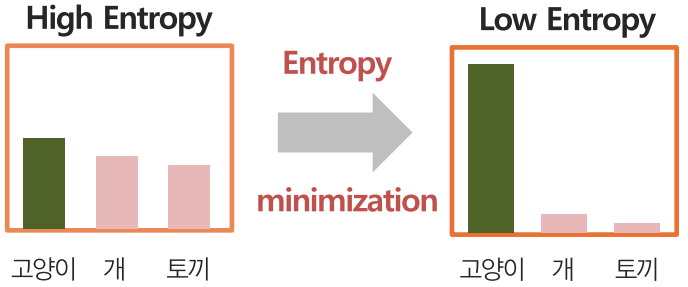
TENT

❖ Loss function = Shannon Entropy

- 목적 : 모델이 테스트 입력에 대해 자신 있는 예측(confident)을 하도록 하기 유도하기 위함. (학습 대상 : BN layer의 affine 파라미터)

$$H(\hat{y}) = -\sum_{c} p(\hat{y}_c)\log p(\hat{y}_c)$$

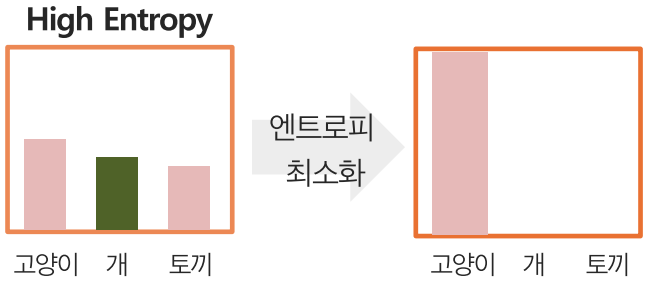
배치 단위로 Adaptation!



$p(y|x)$

Q. Why batch-wise adaptation?

A. 단일 샘플에 대해서만 엔트로피를 최소화하게 되면, 확실하지 않은 상황에서도 확률을 하나의 클래스에 몰아주는 현상 발생



예측 : 고양이
정답 : 개

Wrong but, overconfident!

*trivial solution
: 너무 단순해서 쓸모 없는 답

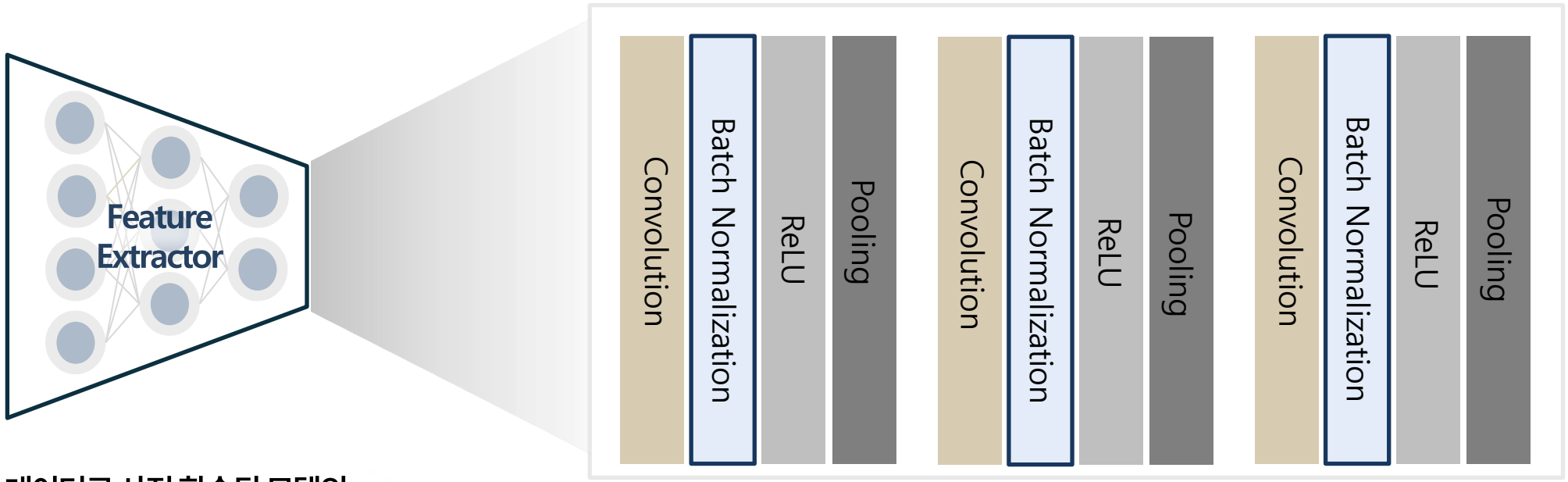
배치 단위로 최적화하여,
여러 샘플을 동시에 보고, 파라미터를 업데이트 하자!

TENT

❖ **TENT의 적응**

현재 입력된 테스트 배치에 대하여, BN Layer의

1) 소스 통계량 → 현재 타겟 배치 통계량으로 대체, 2) affine parameter는 엔트로피가 낮아지는 방향으로 업데이트



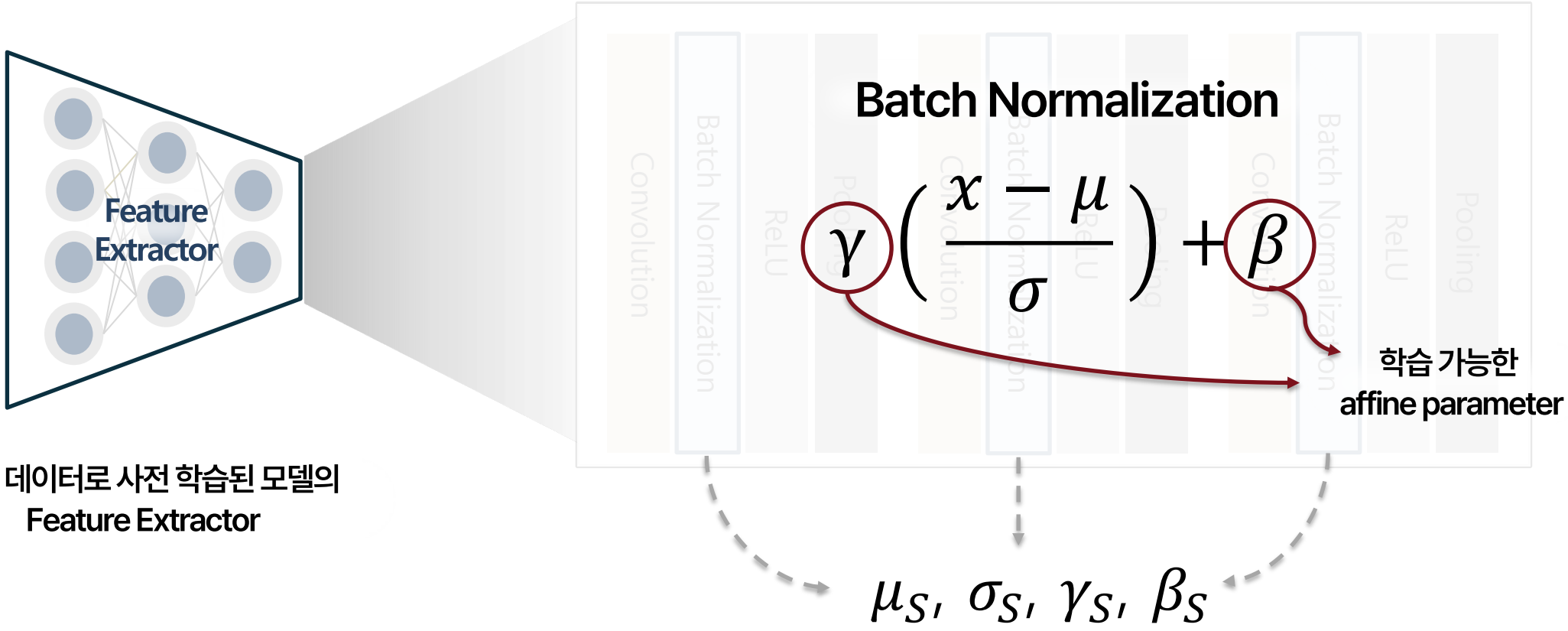
소스 데이터로 사전 학습된 모델의 Feature Extractor

TENT

❖ TENT의 적응

현재 입력된 테스트 배치에 대하여, BN Layer의

1) 소스 통계량 → 현재 타겟 배치 통계량으로 대체, **2)** affine parameter는 엔트로피가 낮아지는 방향으로 업데이트



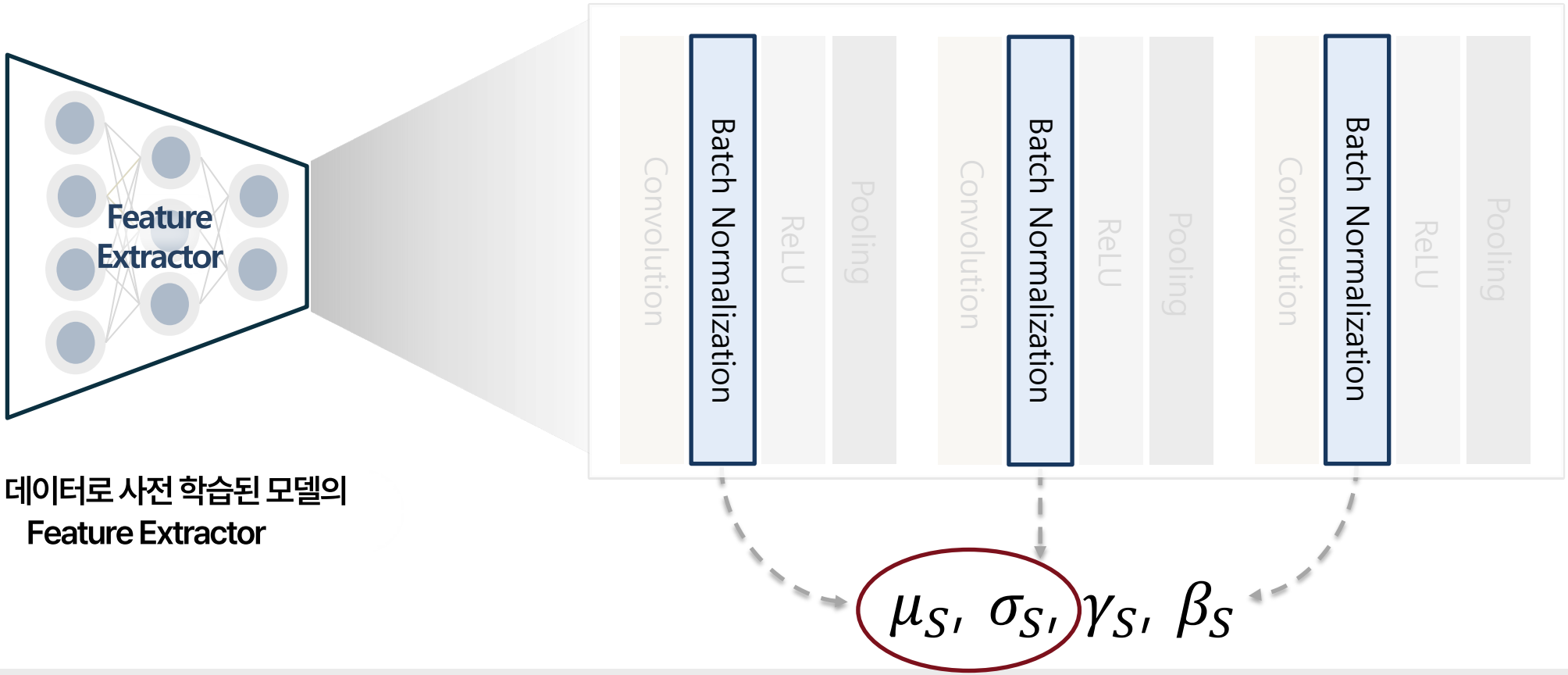
소스 데이터로 사전 학습된 모델의 Feature Extractor

TENT

❖ TENT의 적응

현재 입력된 테스트 배치에 대하여, BN Layer의

1) 소스 통계량 → 현재 타겟 배치 통계량으로 대체, **2)** affine parameter는 엔트로피가 낮아지는 방향으로 업데이트



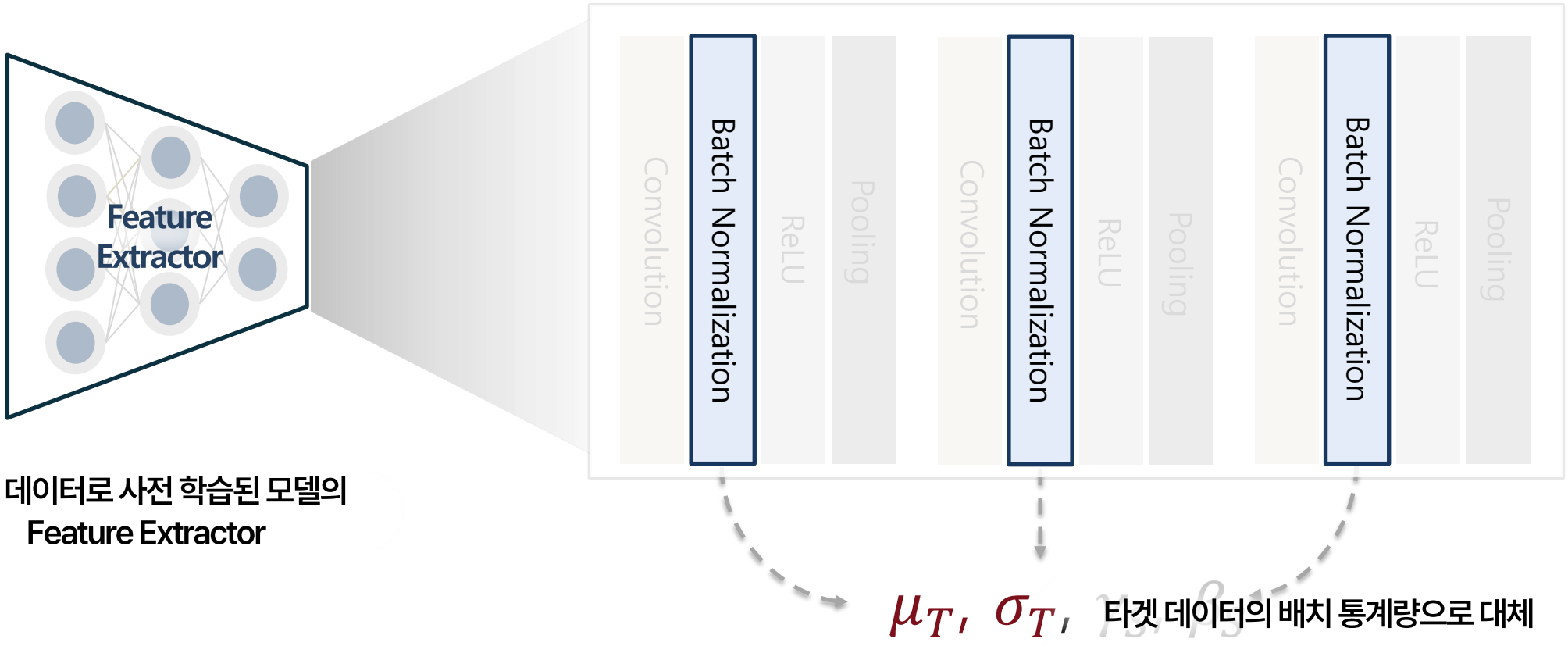
소스 데이터로 사전 학습된 모델의 Feature Extractor

TENT

❖ TENT의 적응

현재 입력된 테스트 배치에 대하여, BN Layer의

1) 소스 통계량 → 현재 타겟 배치 통계량으로 대체, **2)** affine parameter는 엔트로피가 낮아지는 방향으로 업데이트



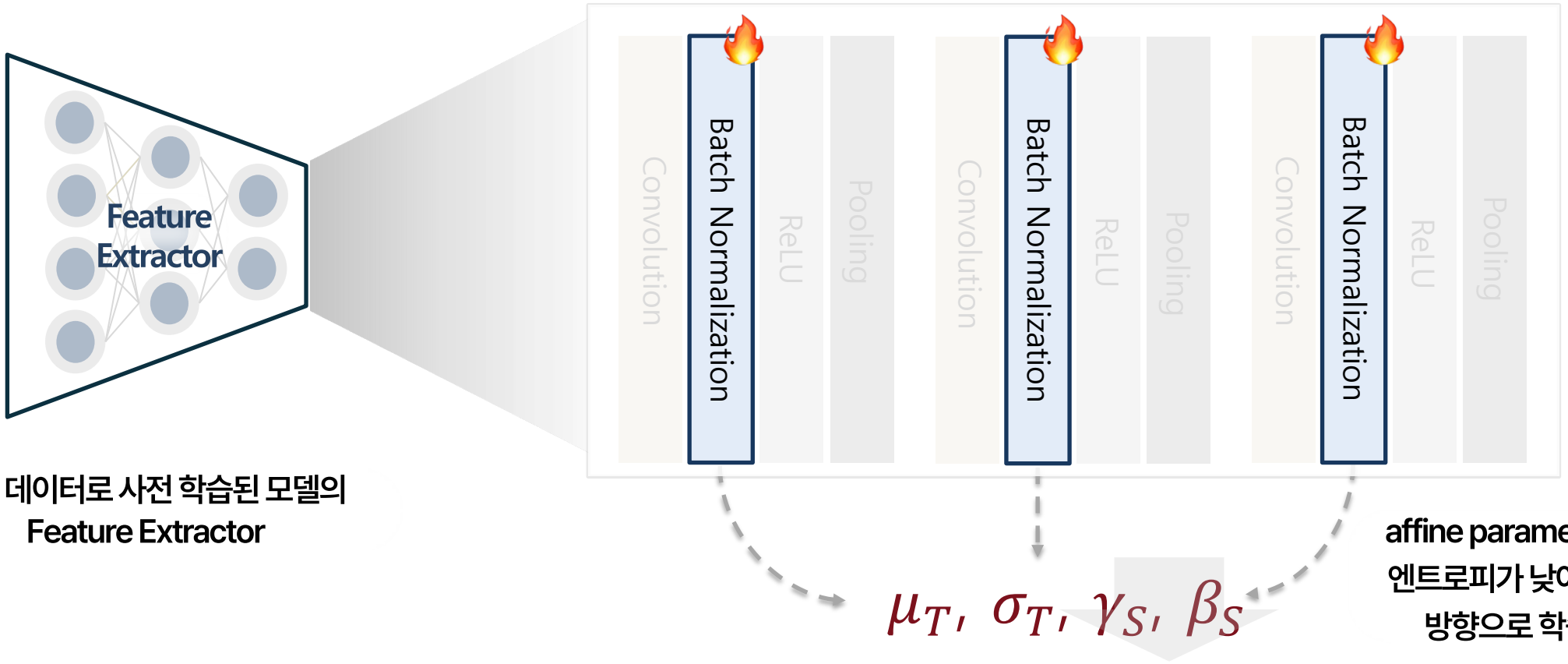
소스 데이터로 사전 학습된 모델의 Feature Extractor

TENT

❖ TENT의 적응

현재 입력된 테스트 배치에 대하여, BN Layer의

1) 소스 통계량 → 현재 타겟 배치 통계량으로 대체, **2)** affine parameter는 엔트로피가 낮아지는 방향으로 업데이트



소스 데이터로 사전 학습된 모델의 Feature Extractor

affine parameter는 엔트로피가 낮아지는 방향으로 학습!

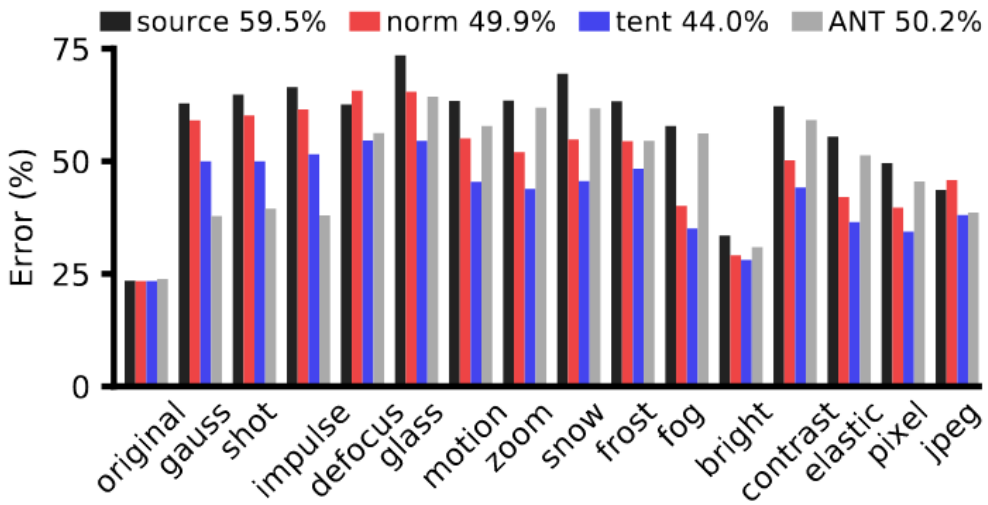
TENT

❖ Results

- 도메인 적응 및 기존 Test-time 방법론들 보다 낮은 오류율을 보임
- ImageNet-C의 모든 손상 유형에서 비교 방법론 대비 가장 낮은 오류율을 보임

	Method	Source	Target	Error (%) C10-C	Error (%) C100-C
	Source	train		40.8	67.2
도메인 적응 (DA)	RG	train	train	18.3	38.9
도메인 적응 (DA)	UDA-SS	train	train	16.7	47.0
테스트 시점 적응	TTT	train	test	17.5	45.0
테스트 시점 적응	BN		test	17.3	42.6
테스트 시점 적응	PL		test	15.7	41.2
	Tent (ours)		test	**14.3**	**37.3**

- CIFAR 10-C / 100-C : CIFAR 10 /100 + 인위적인 19가지 손상 추가 (클래스 10개 / 100개)



- ImageNet-C : ImageNet + 인위적인 12가지의 손상 추가 (노이즈, 블러, 밝기 변화, 등)

TTBA – One-shot

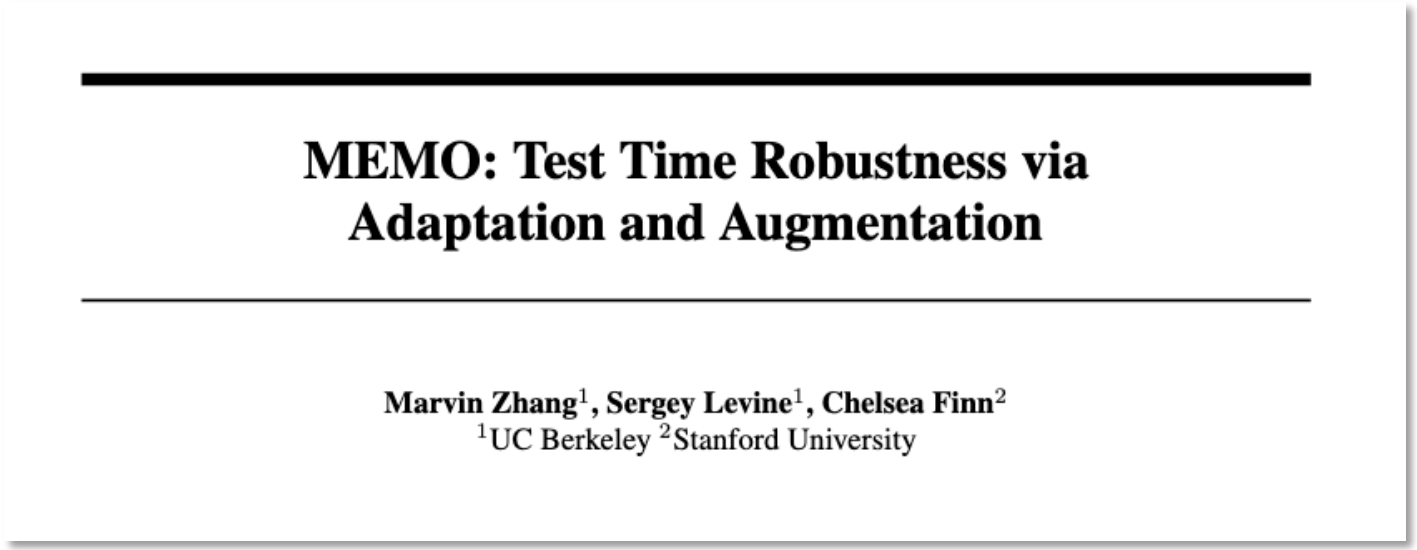
MEMO: Test Time Robustness via Adaptation and Augmentation (NeurIPS 2022)

MEMO

TTBA

❖ **MEMO : Test Time Robustness via Adaptation and Augmentation**

- MEMO (Marginal Entropy minimization with One test point)
- 2022년 NeurIPS, 인용 수 358회
- 하나의 테스트 샘플만으로 모델 적응 가능 (One-shot Adaptation)
- 단일 테스트 샘플을 다양한 방식으로 증강한 결과에 대하여, marginal entropy를 최소화하는 방향으로 모델 전체 파라미터를 업데이트하여 적응 수행



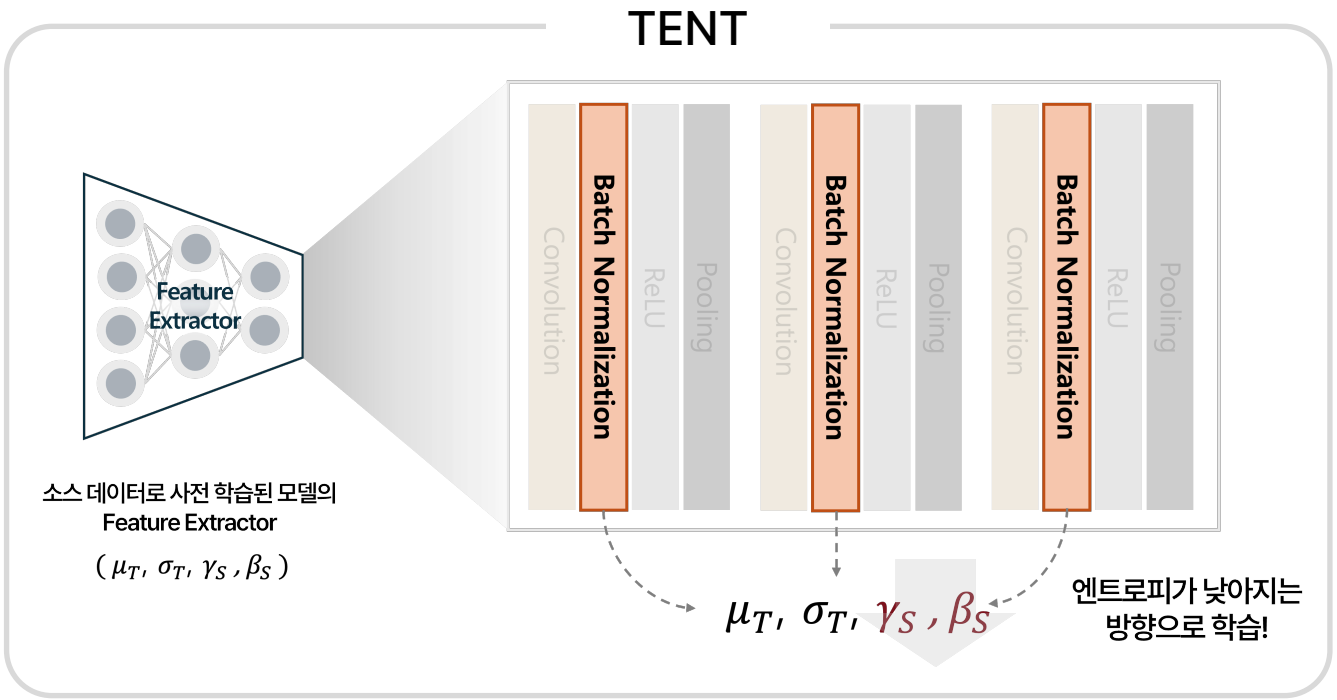
MEMO: Test Time Robustness via Adaptation and Augmentation

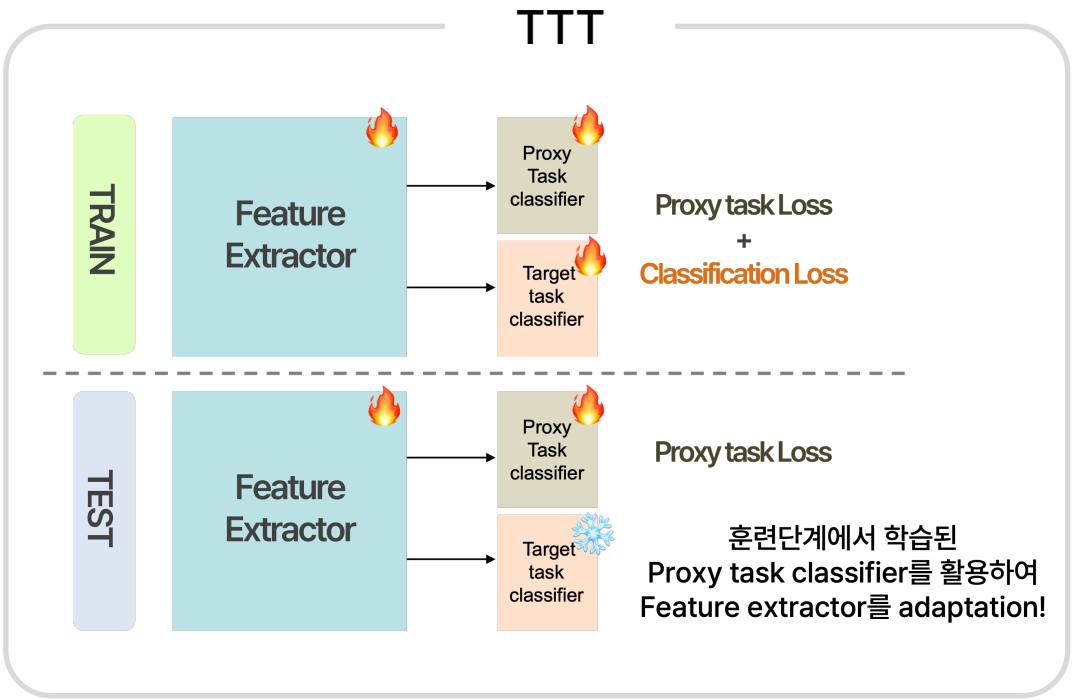
Marvin Zhang[1], Sergey Levine[1], Chelsea Finn[2]
[1]UC Berkeley [2]Stanford University

MEMO

❖ **Motivation : 기존 TTA 방법론들은 multiple test samples를 가정하기 때문에, 현실적인 활용에는 제약이 있다.**

- 기존 방법론들의 한계
  - **Distributionally robust Optimization** : 모델 사이즈 자체를 키우거나, 강한 증강 기반 학습 기반으로 강건성을 높임 → **사전 학습 비용 ↑**
  - **Test-time Training (TTT)** : Test-time에 적용하기 위해 특수한 구조의 모델을 미리 훈련해야 함 → **범용성 ↓**
  - **Test-time Adaptation (TTA)** : 여러 개의 Test samples를 동시에 활용해야 함 → **스트리밍 환경에는 다소 부적합**

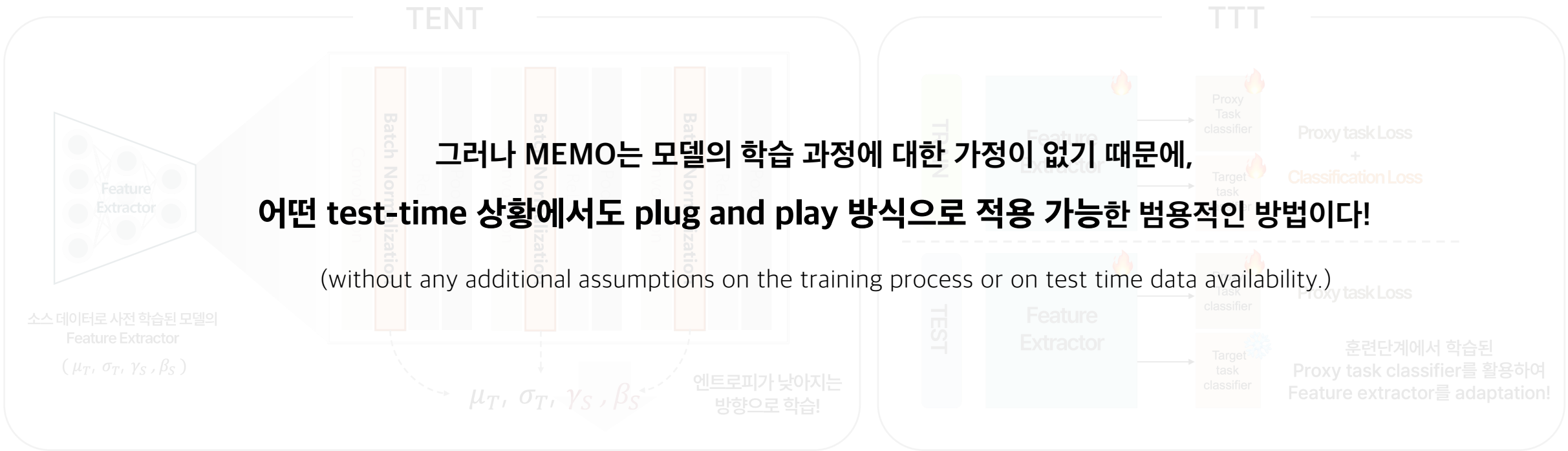




MEMO

❖ **Motivation : 기존 TTA 방법론들은 multiple test samples를 가정하기 때문에, 현실적인 활용에는 제약이 있다.**

- 기존 방법론들의 한계
  - **Distributionally robust Optimization** : 모델 사이즈 자체를 키우거나, 강한 증강 기반 학습 기반으로 강건성을 높임 → **사전 학습 비용 ↑**
  - **Test-time Training (TTT)** : Test-time에 적용하기 위해 특수한 구조의 모델을 미리 훈련해야 함 → **범용성 ↓**
  - **Test-time Adaptation (TTA)** : 여러 개의 Test samples를 동시에 활용해야 함 → **스트리밍 환경에는 다소 부적합**



그러나 MEMO는 모델의 학습 과정에 대한 가정이 없기 때문에,

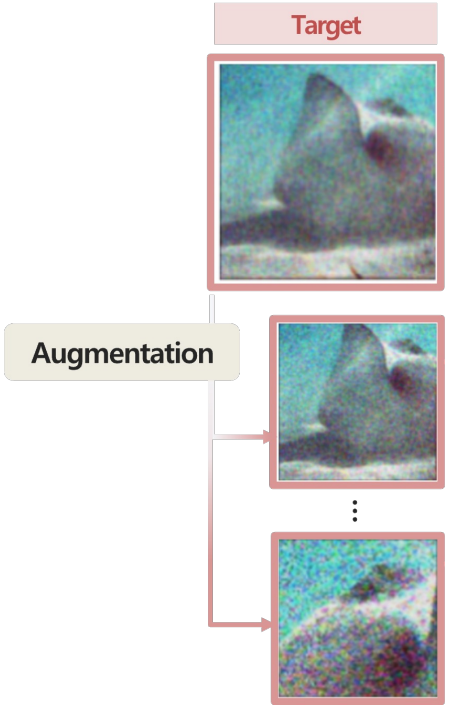
어떤 test-time 상황에서도 plug and play 방식으로 적용 가능한 범용적인 방법이다!

(without any additional assumptions on the training process or on test time data availability.)

MEMO

❖ MEMO의 적응단계

STEP 1. 하나의 테스트 샘플에 대하여, **다양한 증강을 적용하여 여러 증강 샘플을 생성**

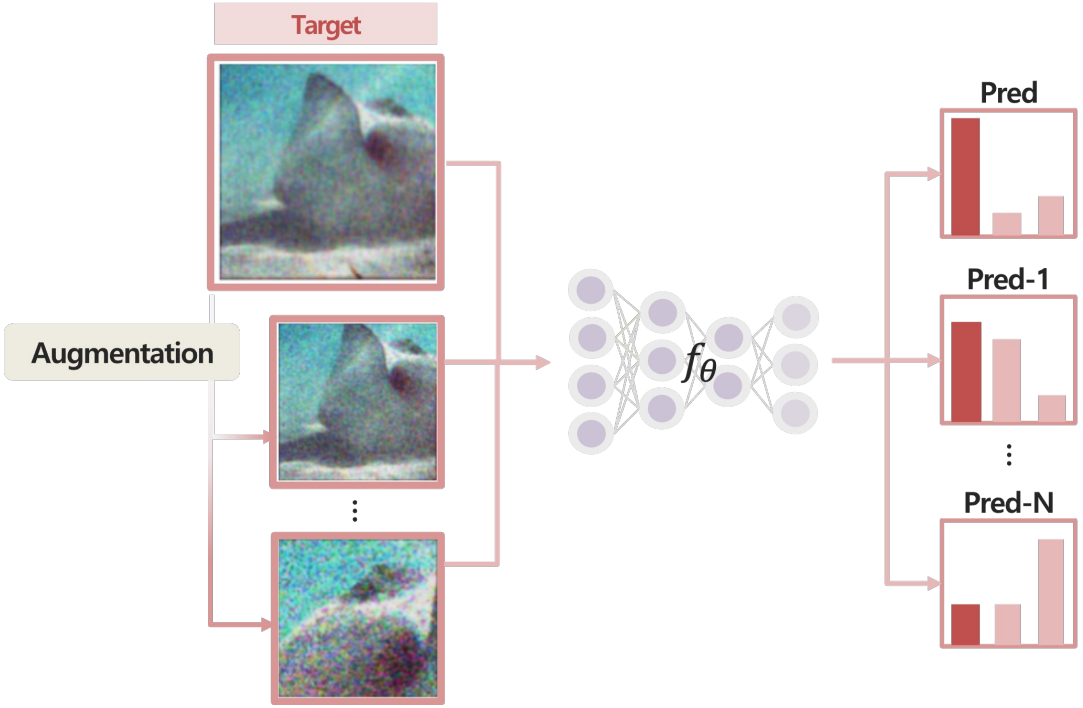


MEMO

❖ MEMO의 적응단계

STEP 1. 하나의 테스트 샘플에 대하여, 다양한 증강을 적용하여 여러 증강 샘플을 생성

STEP 2. 각 증강 샘플을 모델에 통과시켜 예측 분포를 구한 후, 이를 평균 내어 마진 분포를 계산

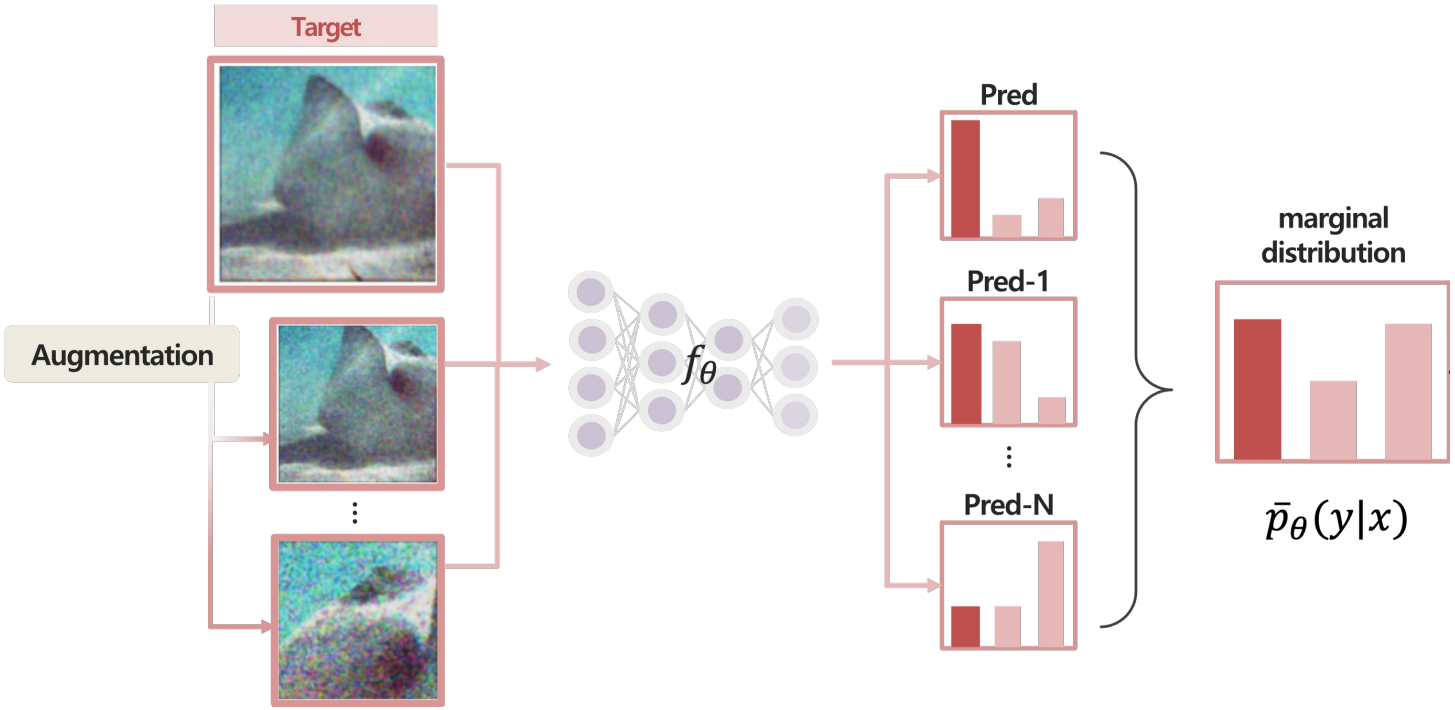


MEMO

❖ MEMO의 적응단계

STEP 1. 하나의 테스트 샘플에 대하여, 다양한 증강을 적용하여 여러 증강 샘플을 생성

STEP 2. 각 증강 샘플을 모델에 통과시켜 예측 분포를 구한 후, 이를 평균 내어 마진 분포를 계산



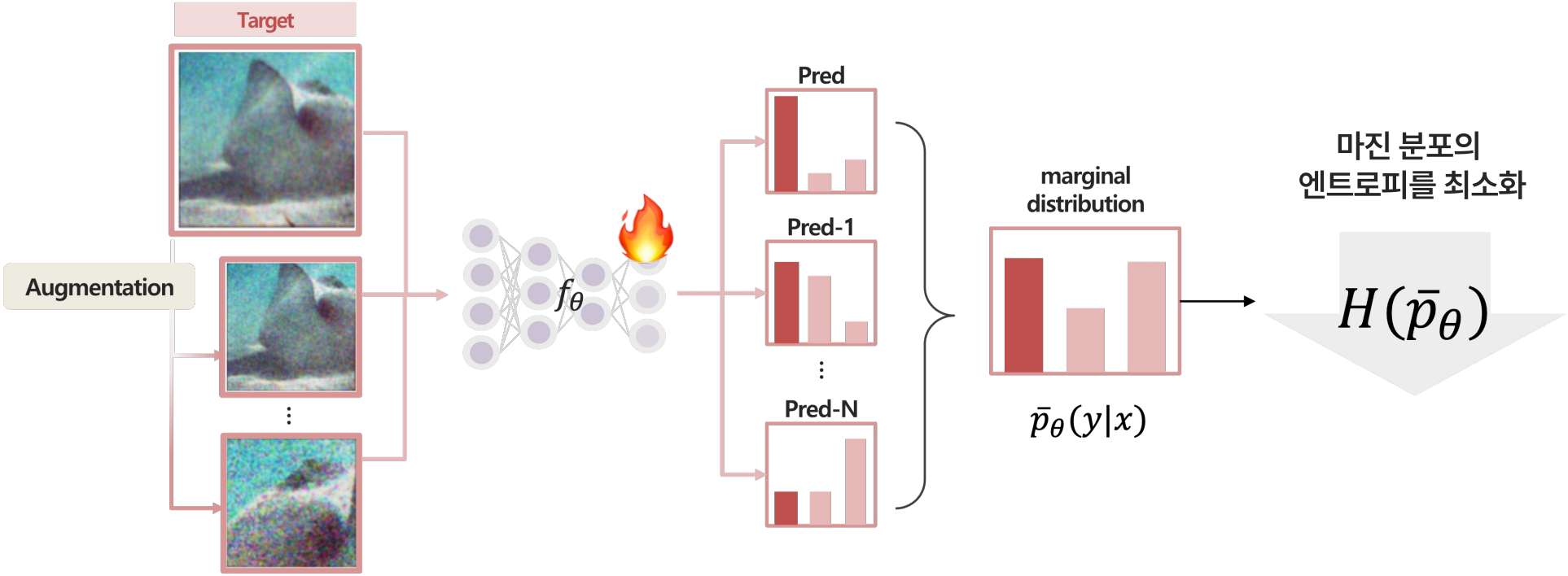
MEMO

❖ MEMO의 적응단계

STEP 1. 하나의 테스트 샘플에 대하여, 다양한 증강을 적용하여 여러 증강 샘플을 생성

STEP 2. 각 증강 샘플을 모델에 통과시켜 예측 분포를 구한 후, 이를 평균 내어 마진 분포를 계산

STEP 3. 계산된 마진 분포의 엔트로피를 최소화하도록, 모델 전체 파라미터를 한 번 업데이트



MEMO

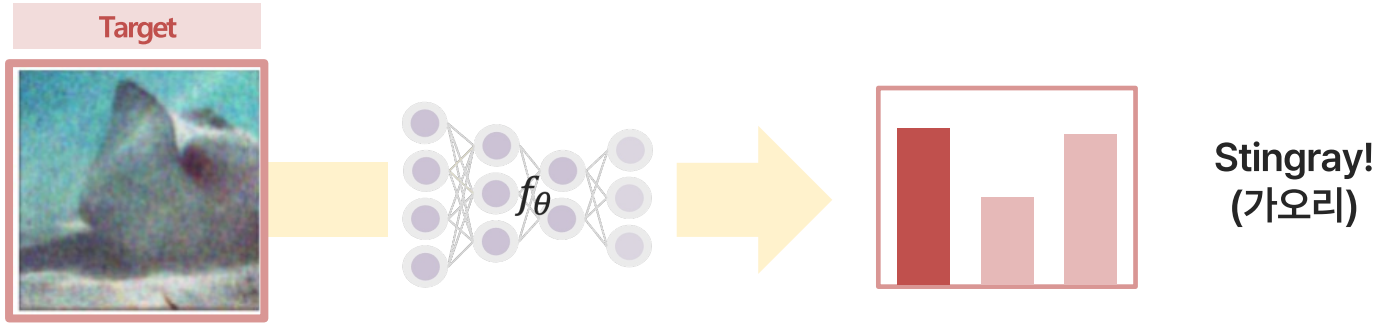
❖ MEMO의 적응단계

STEP 1. 하나의 테스트 샘플에 대하여, 다양한 증강을 적용하여 여러 증강 샘플을 생성

STEP 2. 각 증강 샘플을 모델에 통과시켜 예측 분포를 구한 후, 이를 평균 내어 마진 분포를 계산

STEP 3. 계산된 마진 분포의 엔트로피를 최소화하도록, 모델 전체 파라미터를 한 번 업데이트

STEP 4. 업데이트된 모델로 원본 입력 샘플에 대하여 최종 예측 수행



MEMO

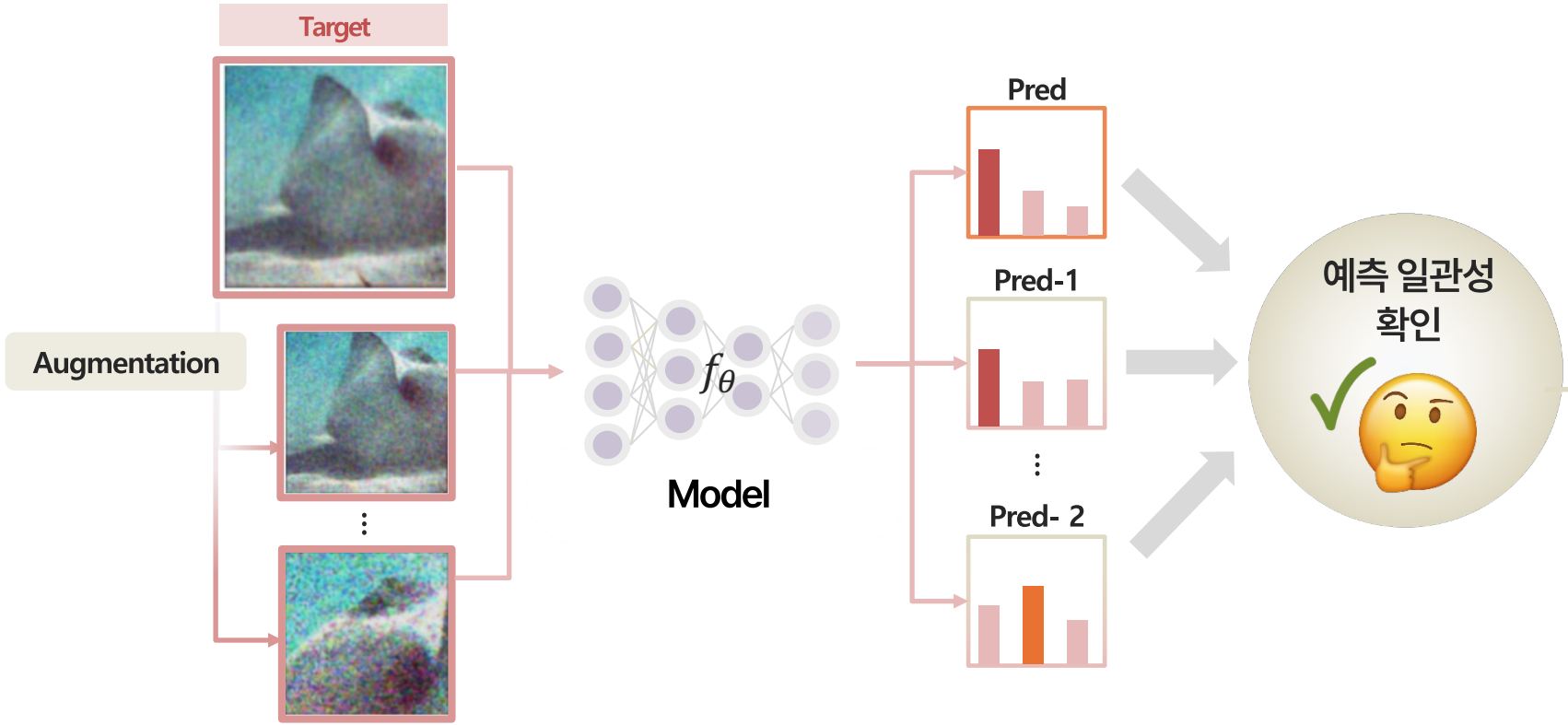
- **Question. Why we use augmentation?**

MEMO

❖ **Question. Why we use augmentation? → 모델이 해당 샘플에 대하여 Invariant한지 확인하고, 그에 맞춰 적응하기 위함!**

MEMO

❖ **Question. Why we use augmentation? → 모델이 해당 샘플에 대하여 Invariant한지 확인하고, 그에 맞춰 적응하기 위함!**



If consistent:

모델이 이 샘플에 대하여 Invariant함
Why?
살짝 변형이 되었음에도,
모델이 해당 샘플의 핵심 특징을
잘 포착할 수 있다는 뜻!

else (inconsistent):

모델이 이 샘플에 대하여 invariant하지 않음.
Why ?
살짝만 변형돼도 예측을 잘 못할 정도로,
모델이 해당 샘플에 대한 이해도가 낮다는 뜻.

MEMO

* 마진분포 :
여러 증강 결과에 대한 평균

$$\bar{p}_\theta = \frac{1}{B}\sum_{i=1}^{B} p_\theta(y|\tilde{x}_i)$$

❖ **Loss function = Marginal Entropy (마진 분포의 엔트로피)**

- **목적 : 모델이 모든 증강에 대해 일관되고, 자신 있는 (Invariant & Confident) 예측을 하도록 유도** (학습 대상 : 모델 전체)

average of conditional entropy	marginal entropy ✓
각 증강 샘플 별 예측 값의 엔트로피의 평균	각 증강 샘플 별 예측 값의 평균의 엔트로피
$L_{CE}(\theta) = \frac{1}{B}\sum_{i=1}^{B} H(p_\theta(\cdot\|\tilde{x}_i))$	$L_{ME}(\theta) = H(\bar{p}_\theta(\cdot\|x))$
Confidence만 유도 가능	Confidence와 Invariance 동시 유도 가능

MEMO

* 마진분포 :
여러 증강 결과에 대한 평균

$$\bar{p}_\theta = \frac{1}{B}\sum_{i=1}^{B} p_\theta(y|\tilde{x}_i)$$

❖ **Loss function = Marginal Entropy (마진 분포의 엔트로피)**

- **목적 : 모델이 모든 증강에 대해 일관되고, 자신 있는 (Invariant & Confident) 예측을 하도록 유도** (학습 대상 : 모델 전체)

average of conditional entropy

Sample 1 : [0.9, 0.05, 0.05] → **A**

Sample 2: [0.1, 0.85, 0.05] → **B**

Sample 3: [0.05, 0.05, 0.9] → **C**

confident
but not invariant

Confidence만 유도 가능

엔트로피 ↓ 🧐

marginal entropy ✓

Sample 1 : [0.9, 0.05, 0.05] → **A**

Sample 2: [0.1, 0.85, 0.05] → **B**

Sample 3: [0.05, 0.05, 0.9] → **C**

confident
but not invariant

[0.35, 0.32, 0.33]

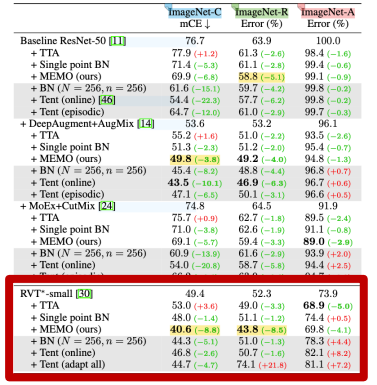
Confidence와 Invariance 동시 유도 가능

엔트로피 ↑

MEMO

❖ Results

- 다양한 모델에서도 타 방법론 대비 우수한 성능을 보이며 작동하는 것을 실증하며 범용적으로 적용가능한 방법론임을 보임
- 단일 테스트 샘플만 사용함에도, 모든 난이도의 데이터셋에서 일관된 성능 향상을 보임
- ImageNet-A와 같은 복잡하고 어려운 데이터셋에서도 우수한 성능을 보임



오른쪽으로 갈수록 어려운 데이터셋

	ImageNet-C mCE ↓	ImageNet-R Error (%)	ImageNet-A Error (%)
RVT*-small [30]	49.4	52.3	73.9
증강 샘플 예측 평균 + TTA	53.0 (+3.6)	49.0 (−3.3)	**68.9** (−5.0)
단일 샘플 사용 + Single point BN	48.0 (−1.4)	51.1 (−1.2)	74.4 (+0.5)
단일 샘플 사용 + MEMO (ours)	**40.6** (−8.8)	**43.8** (−8.5)	69.8 (−4.1)
여러 샘플 사용 + BN ($N = 256, n = 256$)	44.3 (−5.1)	51.0 (−1.3)	78.3 (+4.4)
여러 샘플 사용 + Tent (online)	46.8 (−2.6)	50.7 (−1.6)	82.1 (+8.2)
여러 샘플 사용 + Tent (adapt all)	44.7 (−4.7)	74.1 (+21.8)	81.1 (+7.2)

- ImageNet-C : ImageNet + 인위적인 손상 추가 (노이즈, 블러, 밝기 변화, 등)
- ImageNet-R : ImageNet + 다양한 스타일 추가 (스케치 스타일, 만화스타일, 등)
- ImageNet-A : ImageNet 모델이 잘 틀리는 어려운 샘플들만 골라서 만든 벤치마크 데이터셋

Conclusion

❖ **What is TTA?**

: 소스 데이터 없이, 소스 모델과 unlabeled test data만으로 target domain에서도 좋은 성능을 낼 수 있도록 모델을 적응

❖ **Main Methods of TTA**

✓ **SHOT**

- TTDA(SFDA) / 모든 배치 사용하여 Adaptation 수행
- Information Maximization + Self-Supervised Pseudo-Labeling
- Feature Extractor 업데이트

✓ **TENT**

- TTBA / 현재 배치만을 사용하여 Adaptation 수행
- Entropy Minimization + BN layer adjustment
- Feature Extractor의 BN layer만을 업데이트

✓ **MEMO**

- TTBA / 하나의 테스트 샘플만을 사용하여 Adaptation 수행 (one-shot)
- Marginal Entropy minimization + Data augmentation
- 모델 전체 파라미터를 업데이트

Thank you