BRL Seminar

(2023. 05. 29. Mon)

Self-Supervised Learning

Contrastive Learning & Masked Image Modeling

통합과정 6학기 이승한

Papers

1. **Layer Grafted Pretraining** Bridging Contrastive Learning and Masked Image Modeling for Label-Efficient Representations (ICLR 2023)
   - https://arxiv.org/pdf/2302.14138.pdf

2. **Masked Image Modeling with Denoising Contrast**(ICLR 2023)
   - https://arxiv.org/pdf/2205.09616.pdf

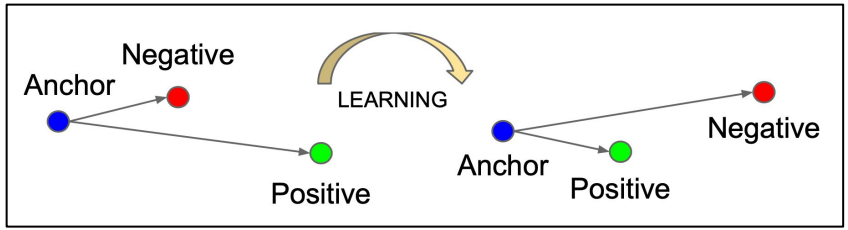
Contents

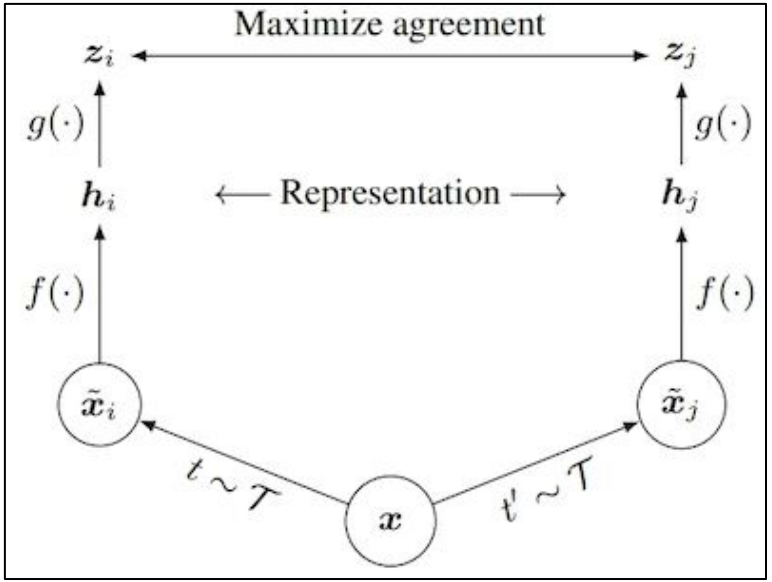
1. Preliminaries

1. Preliminaries: Self-Supervised Learning

1. Contrastive Learning (CL)

- define positive & negative pairs
- Key concept
  - make similar pair (= positive pair) close
  - make dissimilar pair (= negative pair) far

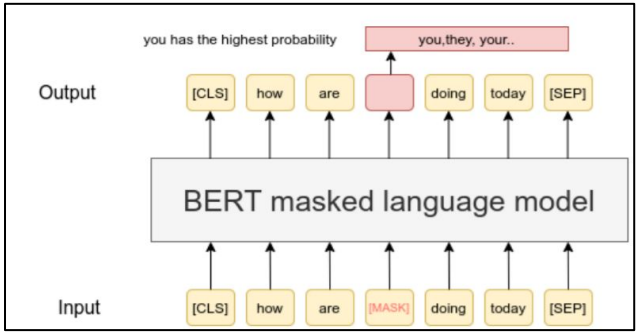




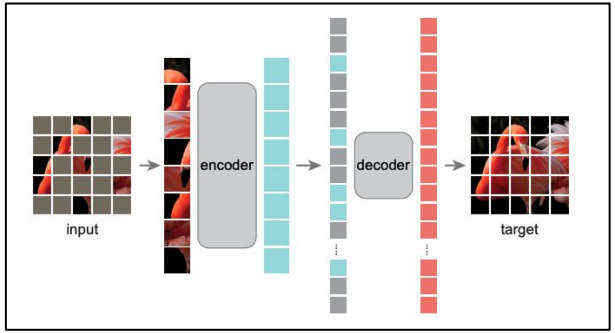
1. Preliminaries: Self-Supervised Learning

2. Masked Modeling (MM)

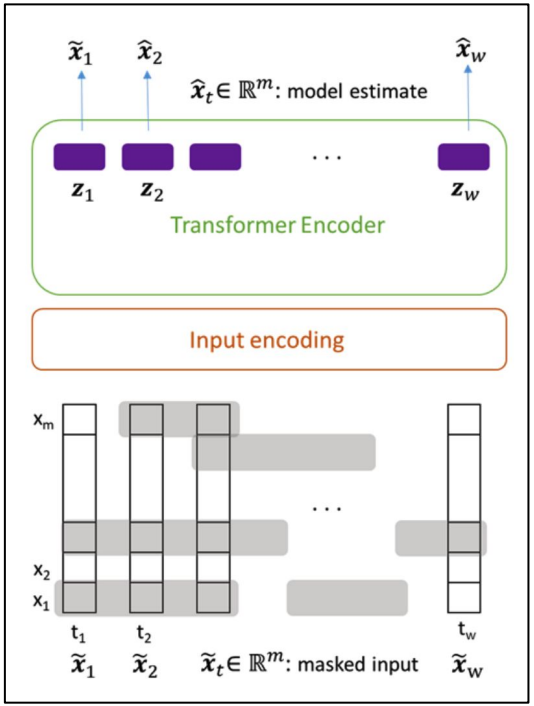
- step 1) randomly mask certain parts
- step 2) reconstruct the masked parts



MLM (Masked Language Modeling)



MIM (Masked Image Modeling)



MTM (Masked Time-Series Modeling)

1. Preliminaries: Self-Supervised Learning

3. CL + MIM

$$\mathcal{M}\left(v_i, v_i^+, V^-, \tau\right) = \frac{1}{N}\sum_{i=1}^{N} -\log \frac{\exp\left(v_i \cdot v_i^+ / \tau\right)}{\exp\left(v_i \cdot v_i^+ / \tau\right) + \sum_{v_i^- \in V^-} \exp\left(v_i \cdot v_i^- / \tau\right)}$$

- V^-: pool of negative features
- N : number of samples

Contrastive Learning

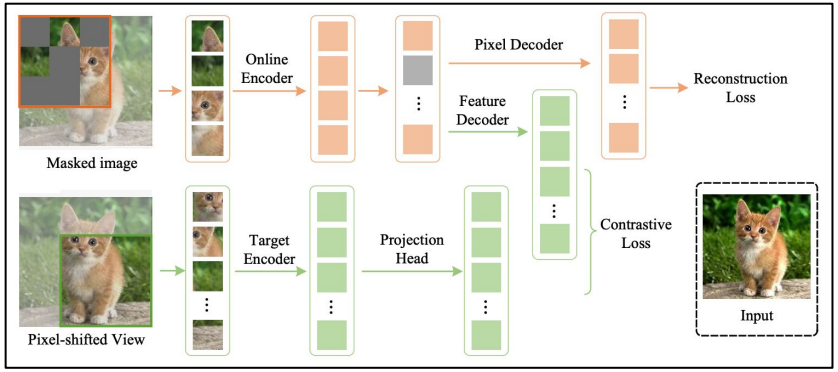
$$\mathcal{L}\left(x_i, M\right) = \frac{1}{N}\sum_{i=1}^{N} D\left(d\left(f\left(M x_i\right)\right), x_i\right).$$

Masked Modeling

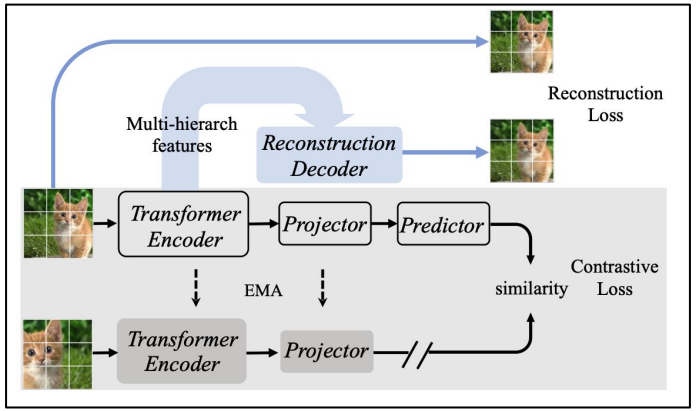
1. Preliminaries: Self-Supervised Learning

3. CL + MIM

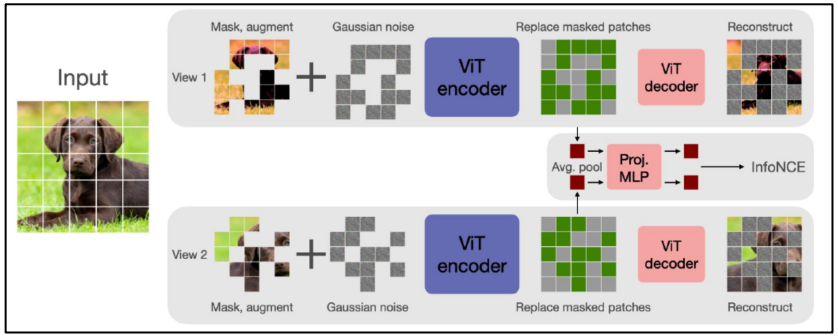
- Simple approach : solve both tasks!
- Final Loss = Loss of CL + Loss of MIM



Contrastive Masked Autoencoders are Stronger Vision Learners



RePre: Improving Self-Supervised Vision Transformer with Reconstructive Pre-training



CAN: A Simple, Efficient and Scalable CMAE framework for learning Visual Representations

2. (Paper 1) Layer Grafted Pretraining

Layer Grafted Pretraining: Bridging Contrastive Learning and Masked Image Modeling for Label-Efficient Representations

2. (Paper 1) Layer Grafted Pretraining

Layer Grafted Pretraining:

Bridging **Contrastive Learning** and **Masked Image Modeling** for Label-Efficient Representations

Naively combining CL & MIM is far from success !

2. (Paper 1) Layer Grafted Pretraining

Abstract

- Naive joint optimization of CL and MIM losses leads to conflicting gradient directions (more severe as the layers go deeper)
- MIM vs CL
  - MIM are suitable to LOWER layers
  - CL are suitable to HIGHER layers
- Propose a simple way to combine CL & MIM : Layer Grafted Pre-training

2. (Paper 1) Layer Grafted Pretraining

Q1) CL and MIM complementary to each other, how to combine?

- simple way : **multiple task learning (MTL) & jointly optimize the two losses**

  -> such a vanilla combination **FAILS** to improve over either baseline

  (often compromising the single loss's performance)

2. (Paper 1) Layer Grafted Pretraining

Q2) *If the two losses conflict, how about placing them differently?*

- Lower layers : learn better from the MIM loss
  - in order to capture local spatial details
- Higher layers : benefit more from the CL loss
  - in order to learn semantically-aware grouping and invariance

propose a simple **MIM → CL Grafting idea** to combine the bests of both worlds

2. (Paper 1) Layer Grafted Pretraining

1. Conflicts with CL & MTM

(Simple Idea) Multi-Task Learning (MTL) combination

- step 1) images are augmented twice for computing the **CL loss**
- step 2) image with minimal augmentation is utilized for computing **MIM loss** following MAE
  (two losses share the same encoder)

2. (Paper 1) Layer Grafted Pretraining

1. Conflicts with CL & MTM

Table 1: Illustration of preliminary study experiments' performance on ViT-B/16. Linear, 1% and Fine-tuning denote linear evaluation, 1% few-shot and fine-tuning performance, respectively. The performance of MIM and CL are from MAE (He et al., 2021) and MocoV3 (Chen et al., 2021), respectively. MTL combination denotes the Muti-Task Learning (MTL) Combination of MIM and CL. MTL combination is pretrained for 300 epochs. For step 1 of MIM→CL and CL→MIM Grafting, we directly adopt the pre-trained model of MAE and MoCo V3, respectively. Step 2 of MIM→CL and CL→MIM Grafting is trained for 100 epochs.

Method	Linear	1%	Fine-tuning
MIM (MAE)	68.0	51.1	83.6
CL (Moco V3)	76.7	63.4	83.2
MTL combination	68.4	47.6	81.0
CL→MIM Grafting	65.5	32.5	82.5
MIM→CL Grafting	74.5	56.5	83.6

- minor performance improvement of 0.4% on linear evaluation compared to the MIM baseline
- still much lower that the CL baseline (76.7 > 68.4)
- even inferior on both 1% few-shot and fine-tuning

2. (Paper 1) Layer Grafted Pretraining

1. Conflicts with CL & MTM

MTL is the cause of the bad performance !

To verify it, design a **gradient surgery experiment** using cosine similarity between gradients of two tasks

$$\mathbf{C}_{\mathrm{MIM,CL}}(x) = \frac{\nabla_\theta L_{\mathrm{MIM}}(x)^T}{||\nabla_\theta L_{\mathrm{MIM}}(x)||} \frac{\nabla_\theta L_{\mathrm{CL}}(x)}{||\nabla_\theta L_{\mathrm{CL}}(x)||}.$$

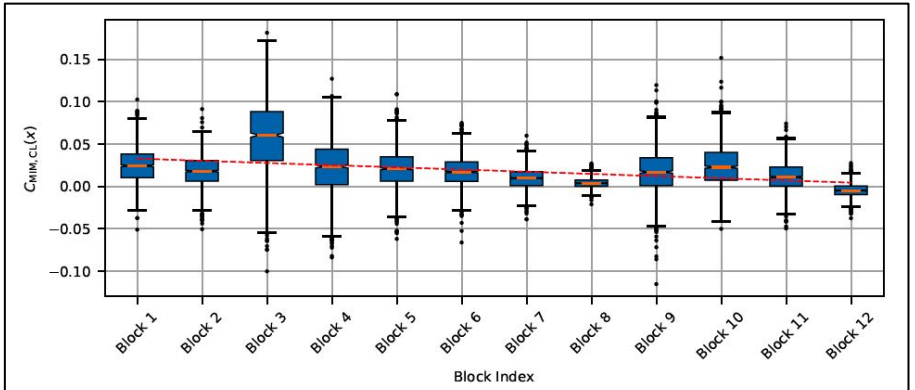


Figure 1: The box plot of $\mathbf{C}_{\mathrm{MIM,CL}}(x)$ across different blocks for MTL combination of MIM and CL. This is measured on training datasets when the network is trained to 100 epochs (total 300 epochs). The red dash line indicates the linear regression of median numbers.

2. (Paper 1) Layer Grafted Pretraining

1. Conflicts with CL & MTM

MTL is the cause of the bad performance !

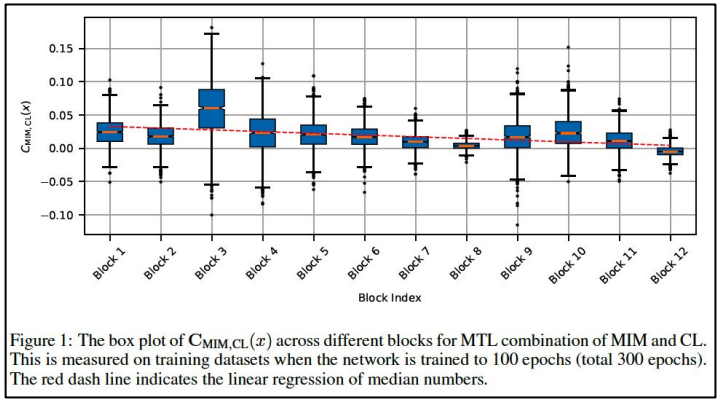


Figure 1: The box plot of $C_{MIM,CL}(x)$ across different blocks for MTL combination of MIM and CL. This is measured on training datasets when the network is trained to 100 epochs (total 300 epochs). The red dash line indicates the linear regression of median numbers.

Findings

- (1) always exist negative values for $\mathrm{C}_{\mathrm{MIM,CL}}(x)$,

  (= where the MIM and CL are optimized in opposite directions)

- (2) gradient direction varies across layers

  (+ more severe as layers go deeper)

2. (Paper 1) Layer Grafted Pretraining

1. Conflicts with CL & MTM

MTL is the cause of the bad performance !

Two losses' contradictory targets

- (1) **MIM loss** : requires that the reconstruction have the same brightness, color distribution, and positions as the input image
  -> the model ***needs to be sensitive*** to all these augmentations.
- (2) **CL loss** : designed to ensure that the model remains ***invariant regardless of different augmentations***

2. (Paper 1) Layer Grafted Pretraining

2. Solving by Separating

If two losses conflicts … how about placing them differently?

Recent empirical evidence suggests that CL and MIM may differ in…

1. Key of MIM = Lower layer
   - When only the **pre-trained lower layers** are retained (while the higher layers are reset to random initialization), most of the gain is still preserved for downstream fine-tuning tasks. (Wang et al. (2022c))
2. Key of CL = HIGHER layer
   - Ineffective and even unstable for training the projection layer (= the earliest/lower layer of ViT)
   - Fixing its weight to be random initialization can even yield significantly higher performance! (Chen et al. (2021))

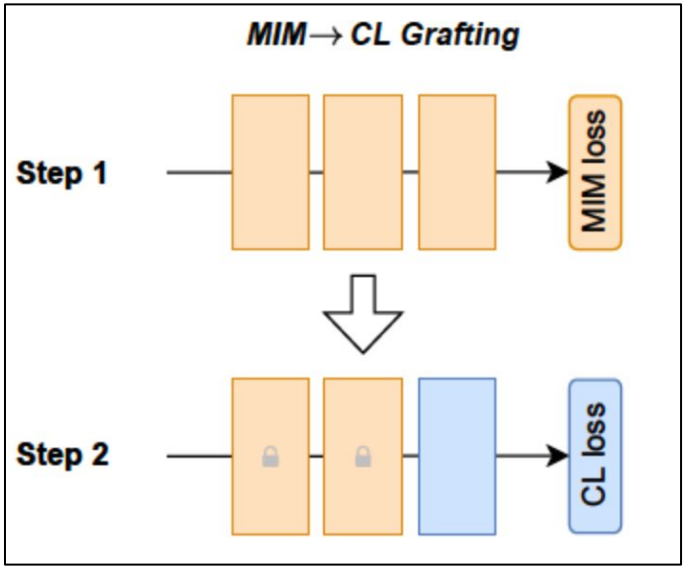
2. (Paper 1) Layer Grafted Pretraining

2. Solving by Separating

Propose a simple **MIM→CL Grafting** framework (two steps)

- step 1) lower layers are first trained with MIM and then fixed
- step 2) higher layers continue to learn with CL

Method	Linear	1%	Fine-tuning
MIM (MAE)	68.0	51.1	83.6
CL (Moco V3)	76.7	63.4	83.2
MTL combination	68.4	47.6	81.0
CL→MIM Grafting	65.5	32.5	82.5
MIM→CL Grafting	74.5	56.5	83.6



2. (Paper 1) Layer Grafted Pretraining

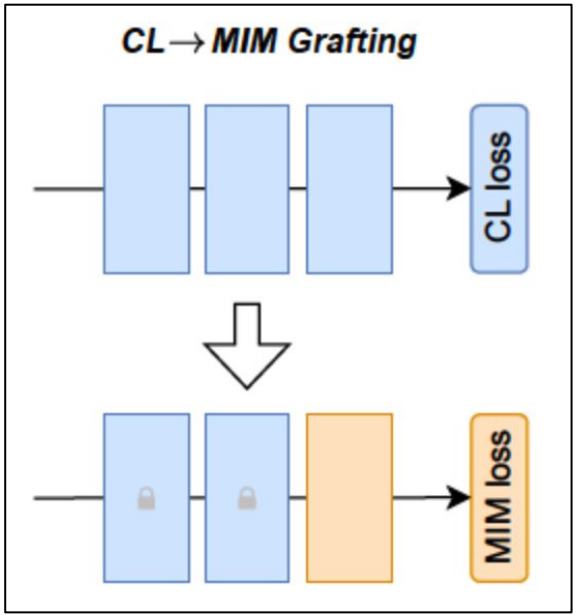
2. Solving by Separating

What if we reverse it? (= **CL→MIM Grafting** framework)

- Worsens! Huge gap between CL→MIM and MIM→CL Grafting

  -> confirms **the preference of MIM/CL towards lower/higher layers**

Method	Linear	1%	Fine-tuning
MIM (MAE)	68.0	51.1	83.6
CL (Moco V3)	76.7	63.4	83.2
MTL combination	68.4	47.6	81.0
CL→MIM Grafting	65.5	32.5	82.5
MIM→CL Grafting	74.5	56.5	83.6



2. (Paper 1) Layer Grafted Pretraining

3. Layer Grafted Pre-training

Smooth out the boundary of "MIM→CL grafting" to **avoid a sudden change** in the feature space.

-> rather than fixing the lower layers,

we **assign them a small learning rate**

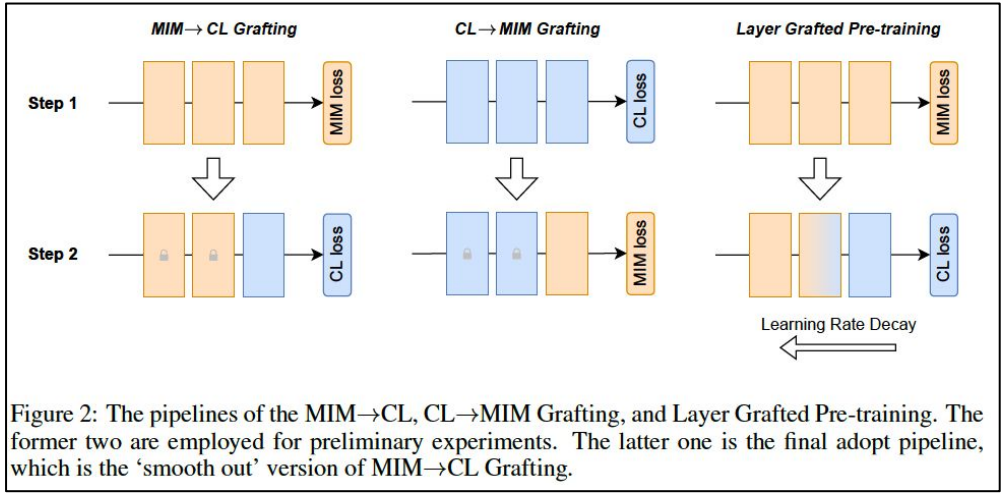


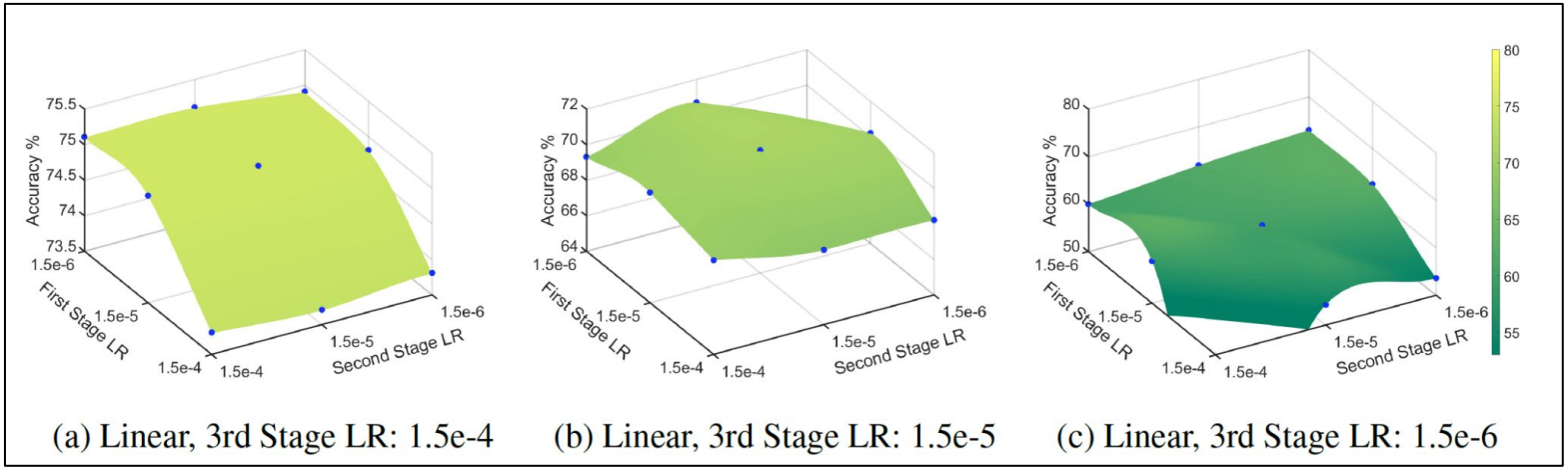
Figure 2: The pipelines of the MIM→CL, CL→MIM Grafting, and Layer Grafted Pre-training. The former two are employed for preliminary experiments. The latter one is the final adopt pipeline, which is the 'smooth out' version of MIM→CL Grafting.

2. (Paper 1) Layer Grafted Pretraining

4. Experiments

LR search Layer Grafted Pre-Training

- The preference for larger LR for higher layers = benefit by performing CL



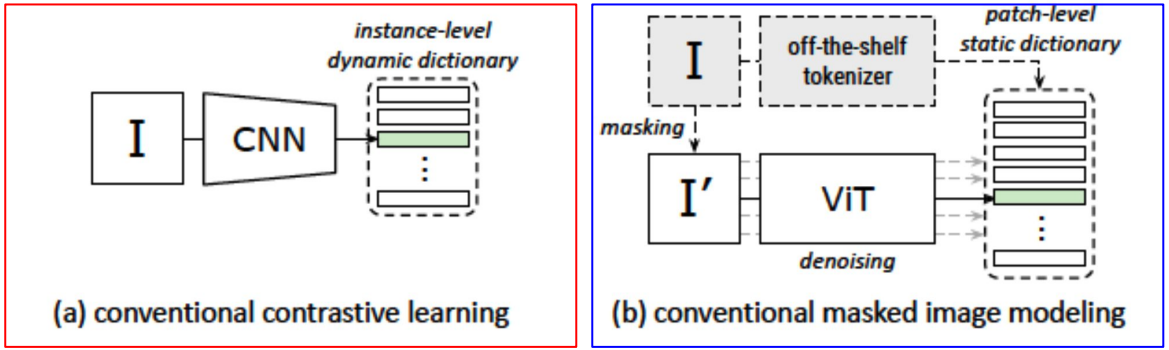
3. (Paper 2) ConMIM

Masked Image Modeling with Denoising Contrast

3. (Paper 2) ConMIM

1. CL vs MIM

- Both attempt to learn discriminative visual representations via **“dictionary look-up”**
- Difference?
  - **CL** : instance-level dictionary look-up
  - **MIM** : patch-level dictionary look-up



3. (Paper 2) ConMIM

1. CL vs MIM

Two factors lead to SOTA of MIM

- (1) More “fine-grained” supervision (instance level -> patch-level)
- (2) Denoising auto-encoding mechanism
  - encourages the capability of backbone network to capture “contextualized representations”

$$\mathcal{L}_{\text{mim}}(x) = \mathbb{E}_{j \in \mathcal{M}} \left[-\log p\left(y_j \mid f(\hat{x})_j\right) \right].$$

- $\mathcal{M}$: the set of masked patch indices
- $\hat{x}$: corrupted image after randomly masking
- y_j : positive key index in the patch-level dictionary
- $p(\cdot \mid \cdot)$: the probability that correctly identifies the recovered patch $f(\hat{x})_j$ with a patch index of j.

3. (Paper 2) ConMIM

2. Image Tokenizers of MIM

efforts to design **“patch-level” dictionaries (= image tokenizers)** for MIM

- [BEiT] tokenize high-dimensional images into discrete vision tokens by a discrete VAE
- [mc-BEiT] introduces eased and refined dictionaries with multiple choices.
- [PeCo] proposes to produce perceptual-aware keys in the patch-level dictionary

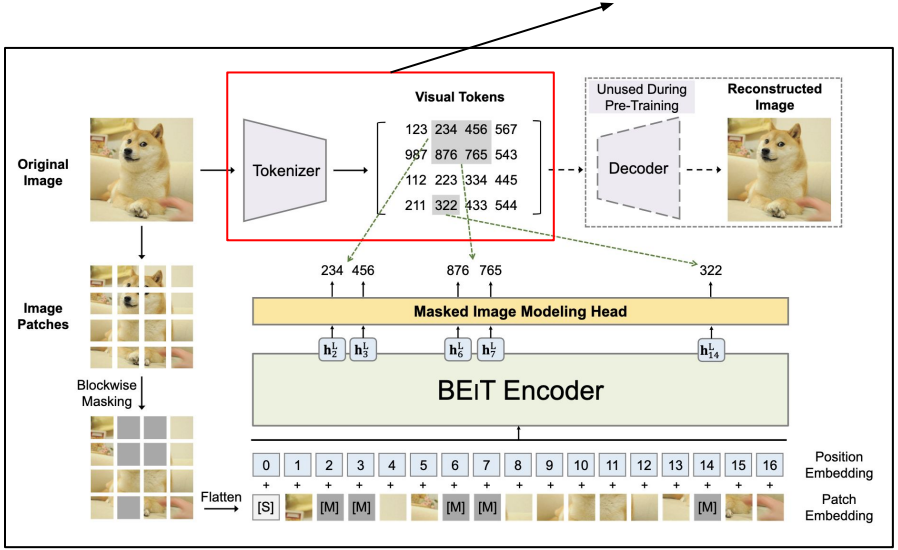
*Still ..... these methods **all require extra training stages***

*(even **extra data** for obtaining a proper image tokenizer)*

3. (Paper 2) ConMIM

2. Image Tokenizers of MIM

extra training needed to learn this tokenizer!



[BEiT] tokenize high-dimensional images into discrete vision tokens by a discrete VAE

3. (Paper 2) ConMIM

2. Image Tokenizers of MIM

efforts to design **“patch-level” dictionaries (= image tokenizers)** for MIM

- [BEiT] tokenize high-dimensional images into discrete vision tokens by a discrete VAE
- [mc-BEiT] introduces eased and refined dictionaries with multiple choices.
- [PeCo] proposes to produce perceptual-aware keys in the patch-level dictionary

*Still ..... these methods **all require extra training stages***

*(even **extra data** for obtaining a proper image tokenizer)*

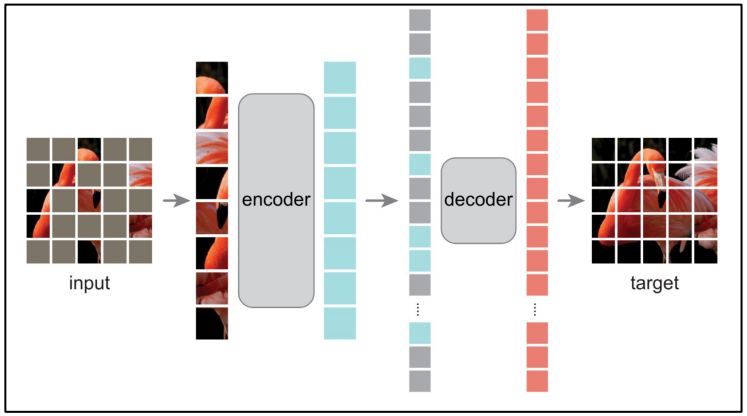
3. (Paper 2) ConMIM

2. Image Tokenizers of MIM

Tokenizer-free MIM methods : cast MIM as ....

- MAE (He et al., 2022) : a pixel-level reconstruction task
- iBOT (Zhou et al., 2022) : a self-distillation task

rather than dictionary look-up



Masked Auto Encoder (MAE)

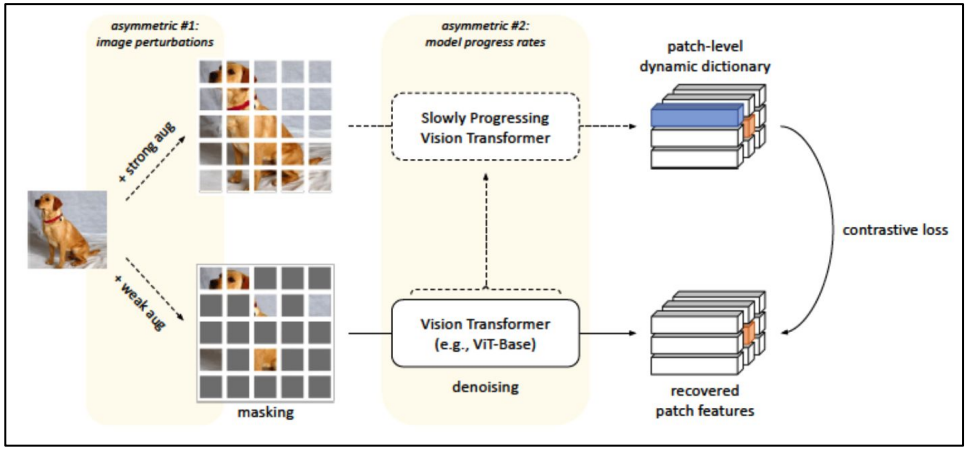
-> fail to achieve competitive results & unsatisfactorily on small-scale architectures

3. (Paper 2) ConMIM

3. ConMIM

proposes **ConMIM**, to perform MIM with denoising contrastive objectives

- Use CL in MIM ! (CL = good capability to structure the latent space for SSL)
- do not need **pre-learned image tokenizers**

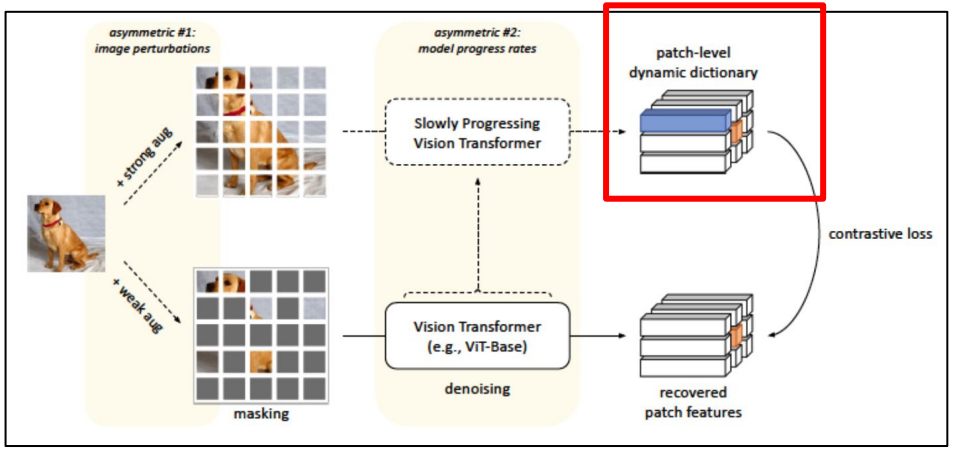


3. (Paper 2) ConMIM

x : full image

$\hat{x}$: corrupted image

3. ConMIM



(a) Patch-level Dynamic Dictionary

build ***dynamic patch-level dictionaries***

Procedure: (during each training iteration ...)

- x : fed into backbone to embed the **patch feature representations**

  $\rightarrow$ serve as keys in the dynamic dictionary, i.e., $\{f(x)_i \mid_{i=1}^{K}\}$

  - i : patch index
  - K : dictionary size

    (as well as the total number of patches within an image)
  - ex) K=196 keys for a 224×224 image with a patch size of 16×16
- build separate dictionaries for each image

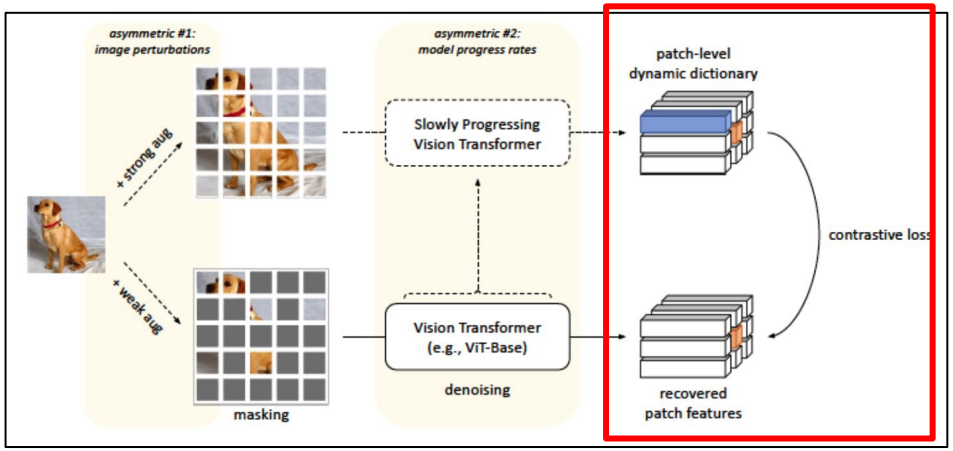
  (= only operate patch-level dictionary look-up **within each image**)

3. (Paper 2) ConMIM

x : full image

$\hat{x}$: corrupted image

3. ConMIM



(b) Denoising Contrastive Loss

Procedure

- $\hat{x}$ is fed into the backbone .... $f(\hat{x})_j, j \in \mathcal{M}$.
- backbone : trained to denoise the corrupted image
- masked patch recovery :

  regularized by a patch-level dictionary look-up (InfoNCE form)

  - $\mathcal{L}_{\text{conmim}}(x) = \mathbb{E}_{j\in\mathcal{M}}\left[-\log \frac{\exp(\langle f(\hat{x})_j, \text{sg}[f(x)_j]\rangle/\tau)}{\sum_{i=1}^{K}\exp(\langle f(\hat{x})_j, \text{sg}[f(x)_i]\rangle/\tau)}\right]$.
- only backpropagate the gradients of $f(\hat{x})$

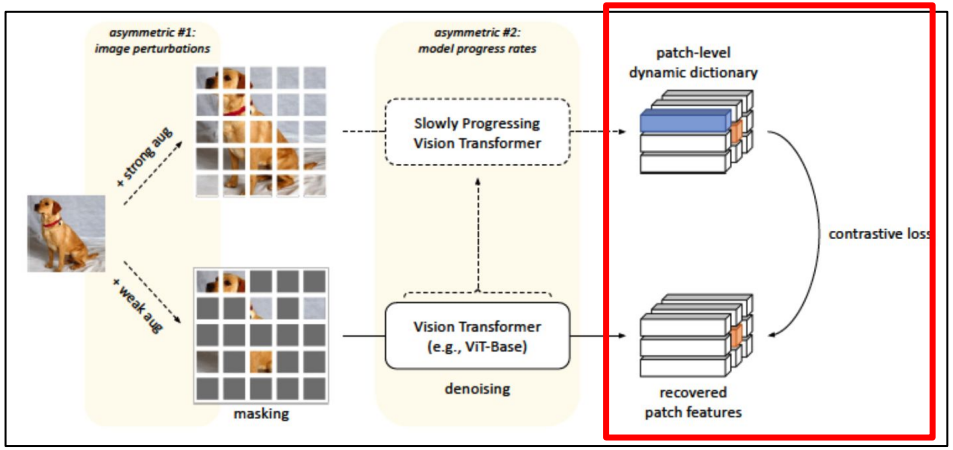
  ($\because$ backpropagating the gradients of $f(x)$ may lead to information leakage)

3. (Paper 2) ConMIM

x : full image

$\hat{x}$: corrupted image

3. ConMIM



(c) Asymmetric Designs

Patches ?

- small-scale inputs with less useful information
- highly redundant semantics

-> need to make pre-training task ***more challenging!***

- MAE (He et al., 2022) : proposes to mask **a large proportion of patches**
- ConMIM (proposed) : further introduce **two asymmetric designs** to enable a stronger denoising regularization

3. (Paper 2) ConMIM

x : full image

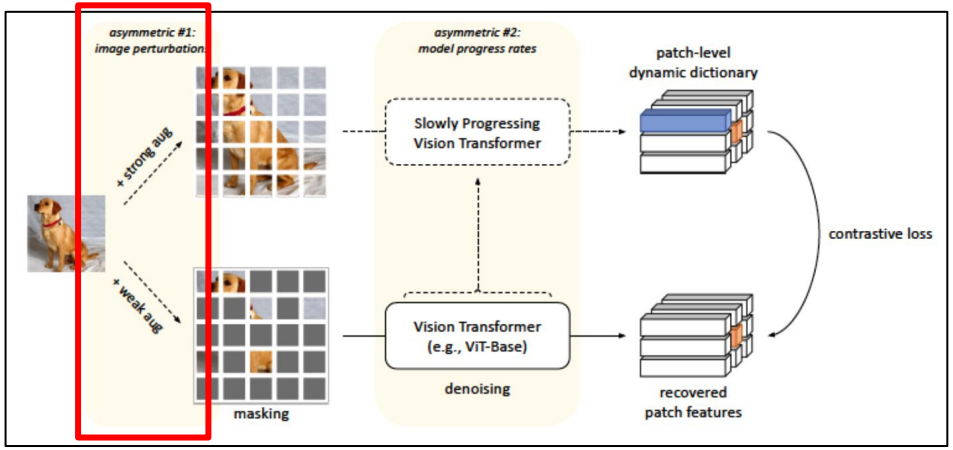
$\hat{x}$: corrupted image

3. ConMIM

(c) Asymmetric Designs

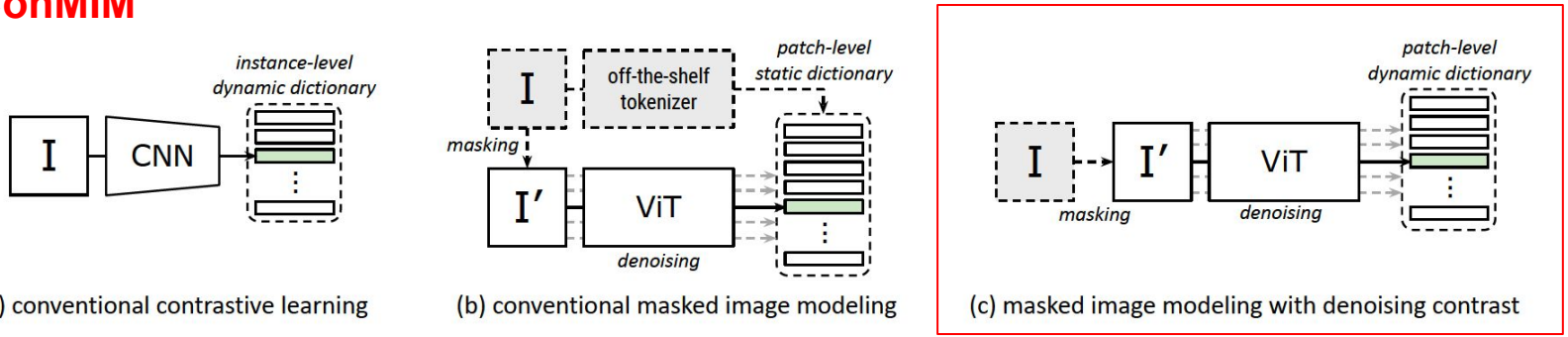
(1) **Asymmetric image perturbations**

- adopt different data augmentations for the x and $\hat{x}$ (stronger augmentations for x)



3. (Paper 2) ConMIM

3. ConMIM



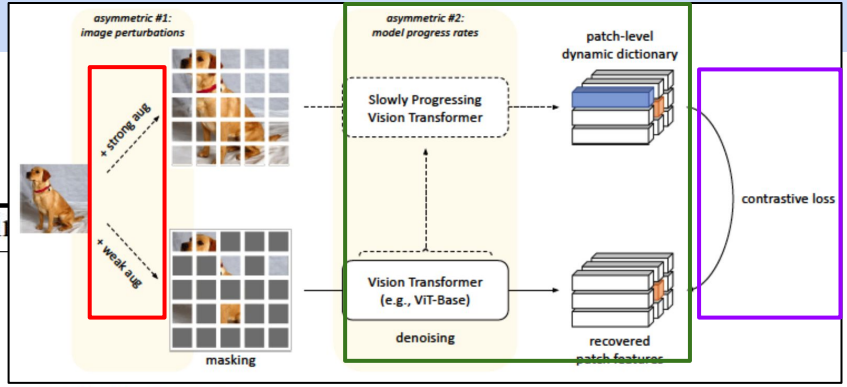
(a) conventional contrastive learning (b) conventional masked image modeling (c) masked image modeling with denoising contrast

Figure 1: Conventional contrastive learning methods (*e.g.*, MoCo (He et al., 2020), SimCLR (Chen et al., 2020)) and masked image modeling methods (*e.g.*, BEiT (Bao et al., 2022), PeCo (Dong et al., 2021)) both perform the pretext task of vision dictionary look-up, where the superiority of the latter ones lie in the patch-level denoising auto-encoding mechanism to enable fine-grained visual context understanding of vision Transformers (Dosovitskiy et al., 2021). We introduce to cast masked image modeling as denoising contrastive learning to avoid the extra training stages of image tokenizer, rendering a flexible, simple and effective pre-training paradigm.

3. (Paper 2) ConMIM

3. ConMIM

PseudoCode



Algorithm 1 Pseudocode of ConMIM pre-trai

```
# f: backbone encoder, e.g., vit-base model
# t: temperature, \tau in the paper
# m: momentum, \alpha in the paper

f_slow.params = f.params # initialize
for (x, mask) in loader: # load a mini-batch x with N samples
   # image preprocess with asymmetric perturbations
   x = aug_basic(x) # share same basic aug for paired inputs
   x_full = aug_strong(x)
   x_corrupted = x*(1-mask)+mask_token.expand_as(x)*mask # randomly mask 75% patches

   # build patch-level dynamic dictionaries with asymmetric models
   with torch.no_grad():
      keys = f_slow(x_full) # NxKxD, K is the number of patches
   feats = f(x_corrupted) # NxKxD

   # dictionary look-up with denoising contrastive loss (Eq.(3))
   sim = bmm(feats.view(N,K,D), keys.view(N,D,K)).view(-1,K) # (NxK)xK
   labels = range(K).repeat(N) # (NxK)
   mask = mask.view(-1) # (NxK)
   loss = CrossEntropyLoss(sim[mask]/t, labels[mask])

   # update model
   loss.backward()
   update(f.params)
   f_slow.params = (1-m)*f.params+m*f_slow.params
```

bmm: batch matrix multiplication

3. (Paper 2) ConMIM

4. Experiments

Models	ImageNet Acc.
DeiT-B (*training from scratch*)	81.8
MoCo v3 (*conventional contrastive learning*)	83.2
ConMIM (Ours)	**83.51**
denoising patch-level contrast → vanilla instance-level contrast	82.26 (-1.25%)
denoising patch-level contrast → vanilla patch-level contrast	*fail*

Table 6: Ablation studies on the effect of denoising auto-encoding mechanism.

use average local tokens to perform instance-level contrastive loss

totally fails due to the trivial information leakage

Models	ImageNet Acc.
DeiT-B (*training from scratch*)	81.8
ConMIM (Ours)	**83.51**
w/o asymmetric image perturbations, use stronger one	83.41 (-0.10%)
w/o asymmetric image perturbations, use basic one	83.35 (-0.16%)
w/ asymmetric image perturbations but switch	82.62 (-0.89%)
w/o asymmetric model progress rates	81.53 (-1.98%)

Table 8: Ablation studies on the effect of asymmetric designs.