Self-Supervised Learning with Swin Transformers

School of Industrial and Management Engineering, Korea University

Young Jae Lee

Contents

Research Purpose

❖ Self-Supervised Learning with Swin Transformers (arXiv, 2021)

- Microsoft Research에서 연구하였고 2021년 09월 10일 기준으로 약 5회 인용

Self-Supervised Learning with Swin Transformers

Zhenda Xie[*†13] **Yutong Lin**[*†23] **Zhuliang Yao**[†13] **Zheng Zhang**[3] **Qi Dai**[3] **Yue Cao**[3] **Han Hu**[3]
[1]Tsinghua University [2]Xi'an Jiaotong University
[3]Microsoft Research Asia
{xzd18,yzl17}@mails.tsinghua.edu.cn yutonglin@stu.xjtu.edu.cn
{zhez,qid,yuecao,hanhu}@microsoft.com

Abstract

We are witnessing a modeling shift from CNN to Transformers in computer vision. In this work, we present a self-supervised learning approach called MoBY, with Vision Transformers as its backbone architecture. The approach basically has no new inventions, which is combined from MoCo v2 and BYOL and tuned to achieve reasonably high accuracy on ImageNet-1K linear evaluation: 72.8% and 75.0% top-1 accuracy using DeiT-S and Swin-T, respectively, by 300-epoch training. The performance is slightly better than recent works of MoCo v3 and DINO which adopt DeiT as the backbone, but with much lighter tricks.

More importantly, the general-purpose Swin Transformer backbone enables us to also evaluate the learnt representations on downstream tasks such as object detection and semantic segmentation, in contrast to a few recent approaches built on ViT/DeiT which only report linear evaluation results on ImageNet-1K due to ViT/DeiT not tamed for these dense prediction tasks. We hope our results can facilitate more comprehensive evaluation of self-supervised learning methods designed for Transformer architectures. Our code and models are available at https://github.com/SwinTransformer/Transformer-SSL, which will be continually enriched.

DMQA

Research Purpose

❖ Self-Supervised Learning with Swin Transformers (arXiv, 2021)

- 비전 분야에서는 최근 두개의 혁명적인 연구 트렌드가 진행되고 있음
  - ✓ Self-Supervised Visual Representation Learning
  - ✓ Transformer-based Architecture Backbone

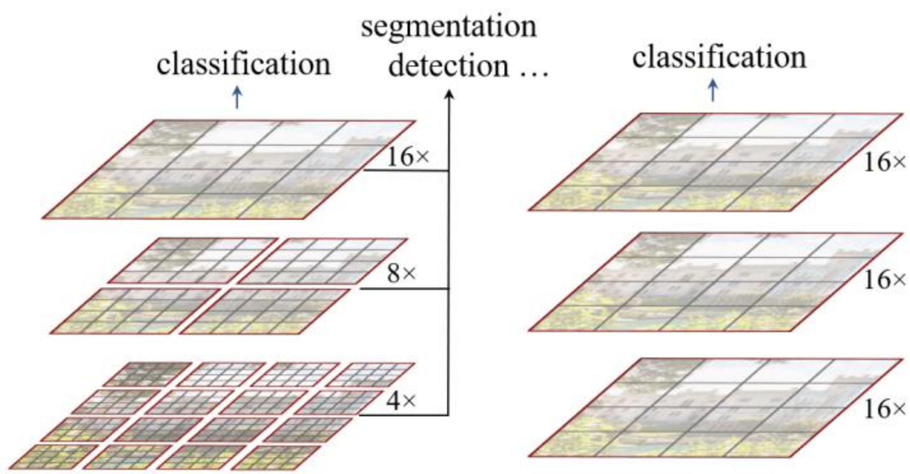
- 두개의 연구를 결합하여 새로운 Transformer 기반의 Self-Supervised Learning 제안
  - ✓ 기존 ViT/DeiT 기반의 Self-Supervised Learning 연구는 MoCo v3와 DINO
  - ✓ 본 연구에서는 **Swin Transformer Backbone** 기반의 **MoBY** (**Mo**Co + **BY**OL) 제안
  - ✓ Swin Transformer는 Dense Prediction Task (Object Detection, Segmentation)도 평가 가능한 **General-Purpose**

A Baseline SSL Method with Swin Transformers

Swin Transformer as the backbone

❖ Hierarchical Feature Representation

- 작은 크기의 Patch (회색 윤곽선)로부터 시작함으로써 더 깊은 Transformer 레이어의 인접 Patch를 점진적으로 병합하여 **계층적 표현**을 구성
- 일반적인 Transformer와 달리 계층적 Feature Map을 통하여 Feature Pyramid Networks (FPN) 또는 U-Net과 같이 **Object Detection, Segmentation에 활용 가능**
- **Linear** Computational Cost는 겹치지 않은 이미지 (빨간색 윤곽선) 내에서 지역적으로 Self-Attention 연산을 수행함으로써 달성



(a) Swin Transformer (ours)

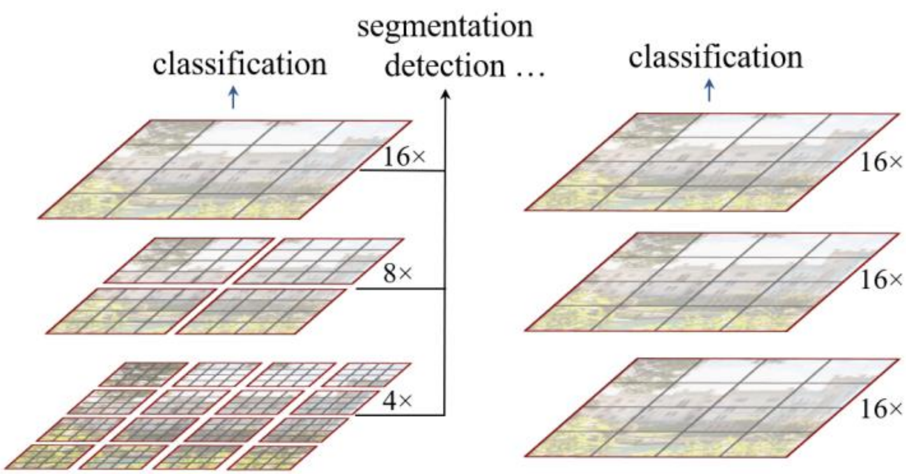
(b) ViT

A Baseline SSL Method with Swin Transformers

Swin Transformer as the backbone

❖ Hierarchical Feature Representation (Example)

- 입력 이미지 사이즈: 224 X 224
- Window 사이즈(M): 7 X 7
- 첫번째 레이어에서 4 X 4 사이즈의 각 Patch가 56 X 56개가 존재할 때 Window 사이즈로 나누어 8 X 8개의 Window를 생성
- 첫번째 단계에서 각 Patch는 16개의 Pixel이 존재하며 각 Window에는 49개의 Patch가 존재



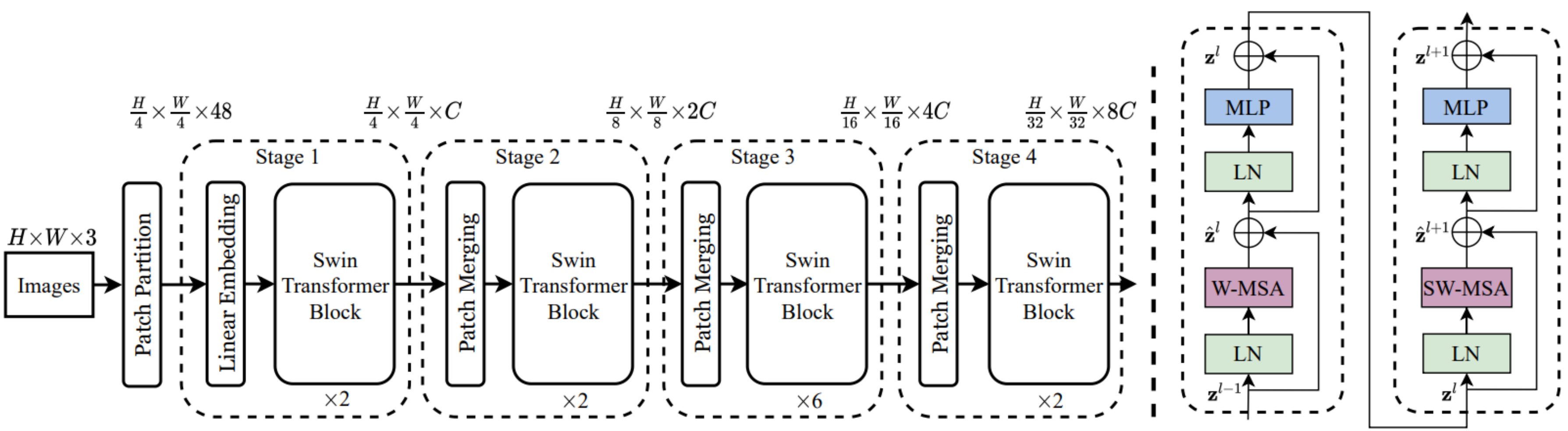
(a) Swin Transformer (ours) (b) ViT

A Baseline SSL Method with Swin Transformers

Swin Transformer as the backbone

❖ Overall Architecture

- 크게 Patch Partition, Linear Embedding, Swin Transformer Block, Patch Merging으로 구분
- 4개의 단계로 이루어져 있음(Stage 1 ~ 4)
- 제안 방법론의 핵심인 그림 (b)는 두개의 Encoder로 이루어져 있으며 일반적인 Multi-Head Self-Attention (MSA)가 아닌 W-MSA, SW-MSA로 이루어져 있음
- 그림 (b)에서 2개의 Encoder를 하나로 보았을 때, 각 단계에서 Block 적용 횟수는 실제로 1, 1, 3, 1



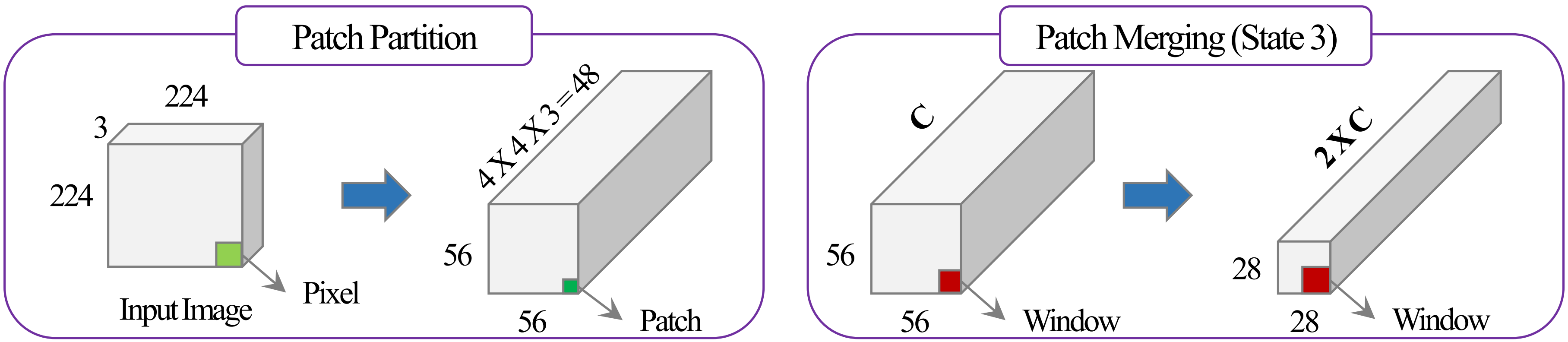
(a) Architecture (b) Two Successive Swin Transformer Blocks

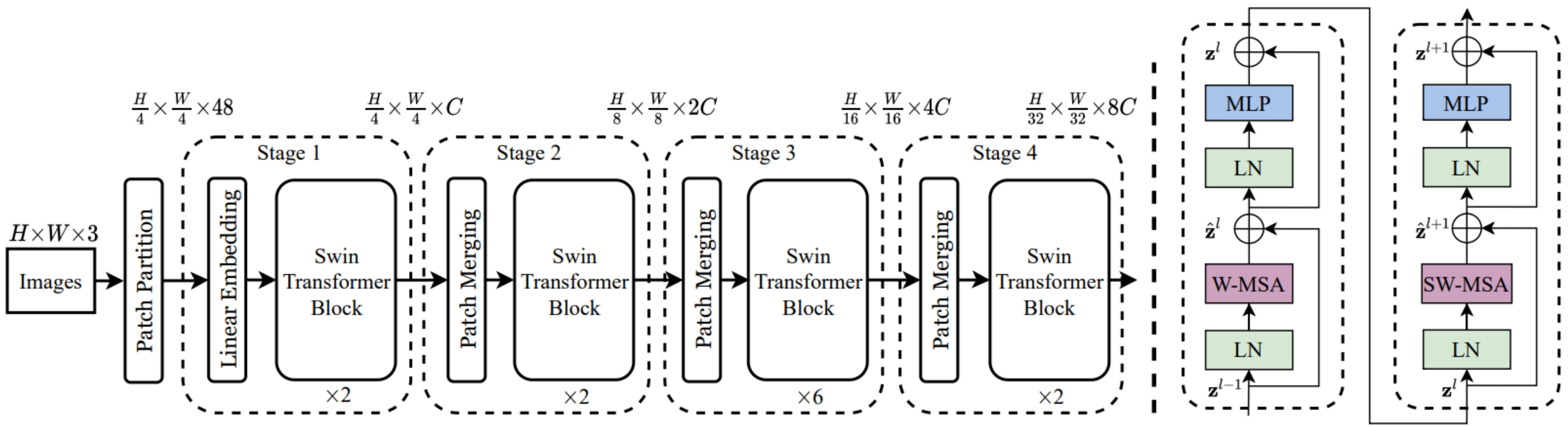
A Baseline SSL Method with Swin Transformers

Swin Transformer as the backbone

❖ Overall Architecture – Patch Partition, Patch Merging

- 이미지에서 Patch로 Partition하는 것과 Merging하는 것은 같은 의미를 가짐





(a) Architecture (b) Two Successive Swin Transformer Blocks

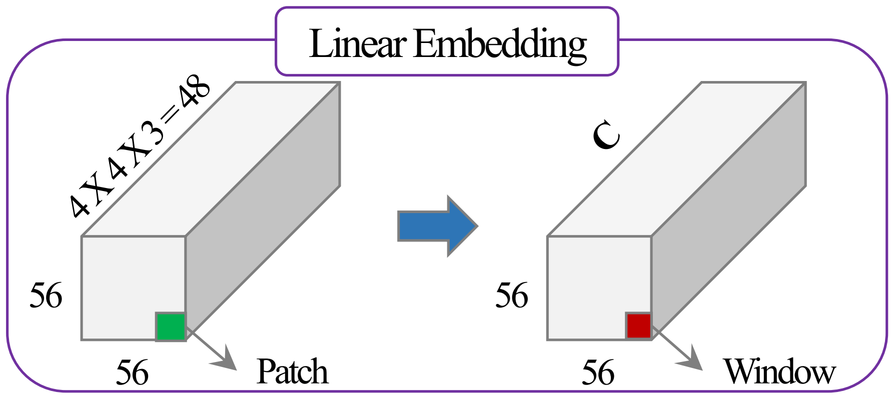
DMQA

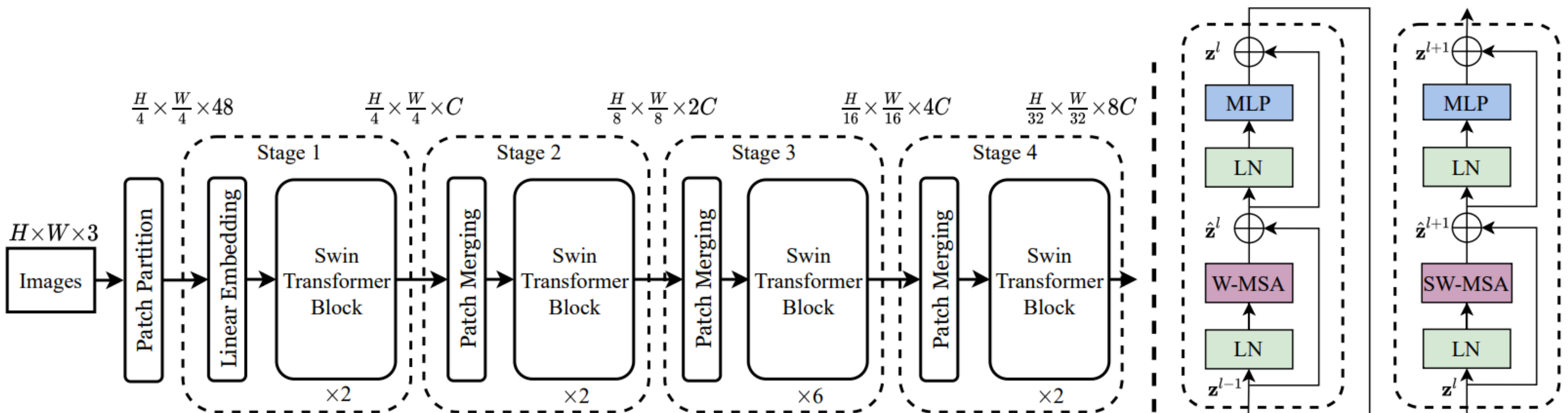
A Baseline SSL Method with Swin Transformers

Swin Transformer as the backbone

❖ Overall Architecture – Linear Embedding

- Patch Partition 또는 Patch Merging 후에 Linear Layer로부터 Dimension C로 출력



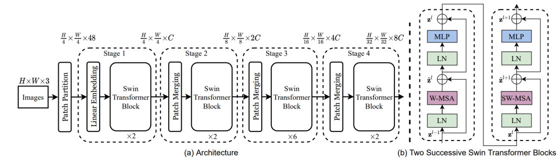


(a) Architecture

(b) Two Successive Swin Transformer Blocks

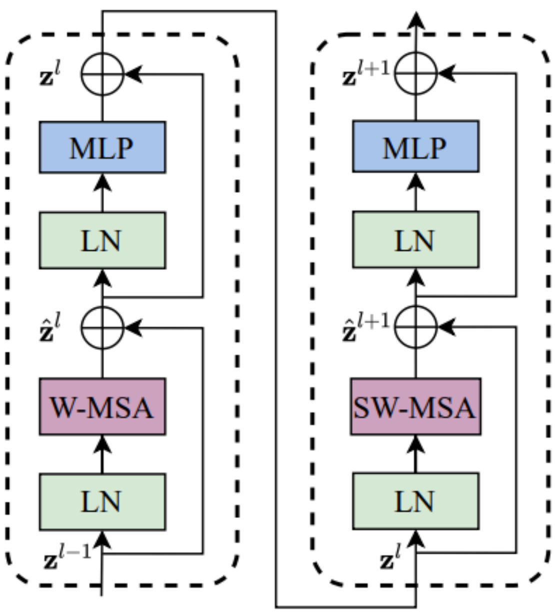
A Baseline SSL Method with Swin Transformers

Swin Transformer as the backbone



❖ Overall Architecture – Swin Transformer Block

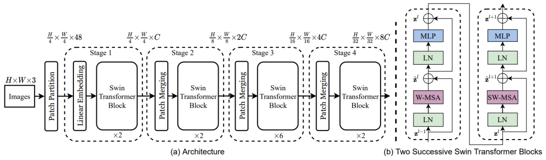
- 두개의 Encoder로 이루어져 있으며 일반적인 Multi-Head Self-Attention (MSA)가 아닌 W-MSA, SW-MSA로 이루어져 있음
- W-MSA는 현재 Window에 있는 Patch들로만 Self-Attention 연산함으로써 계산 비용 감소
- MSA는 Quadratic, W-MSA는 **Linear** Computation (논문에서는 M을 7로 고정)
  - ✓ W-MSA은 hw보다 M이 훨씬 작기 때문에 이미지 사이즈가 커져도 ViT보다 계산 비용을 줄일 수 있음



$$\Omega(\text{MSA}) = 4hwC^2 + 2\boxed{(hw)^2}C,$$
$$\Omega(\text{W-MSA}) = 4hwC^2 + 2\boxed{M^2}hwC,$$

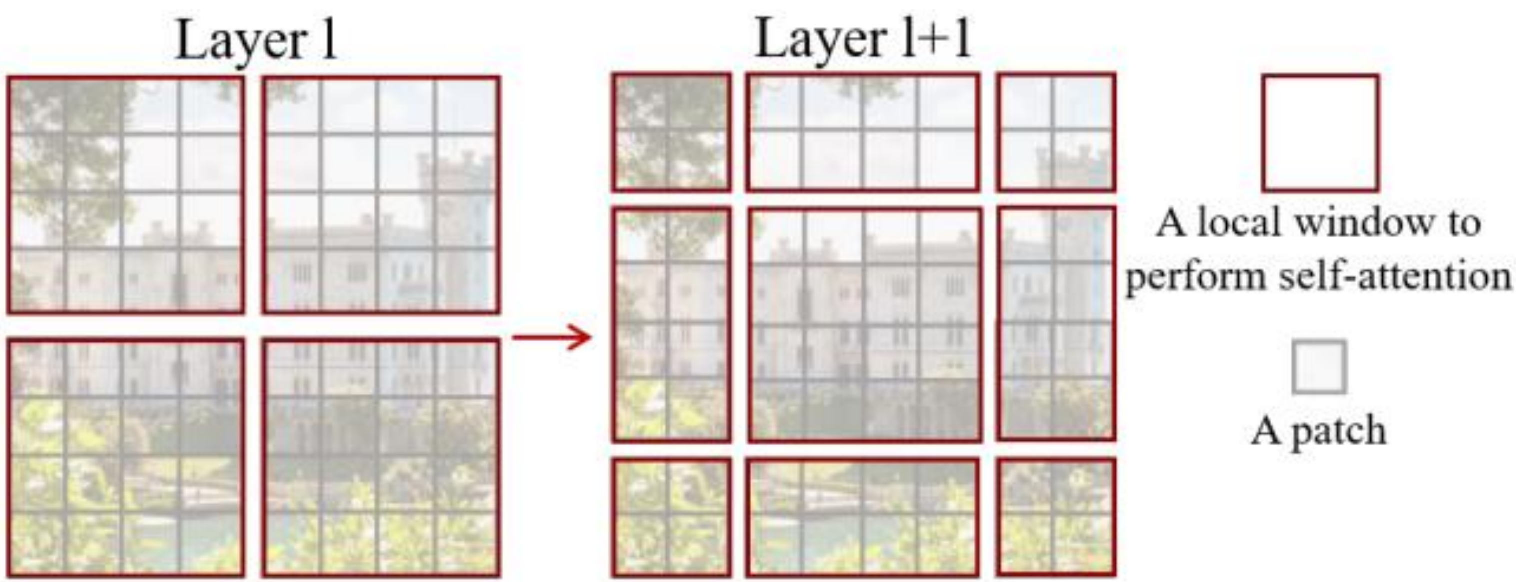
A Baseline SSL Method with Swin Transformers

Swin Transformer as the backbone



❖ Overall Architecture – Swin Transformer Block

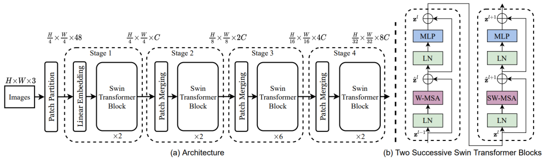
- W-MSA는 Window가 고정되어 있어 고정된 부분에서만 Self-Attention을 연산하는 단점 존재
- Shifted Window 방법을 도입하여 위의 문제를 해결(SW-MSA)
  - ✓ Layer *l*은 Regular Window Partitioning Scheme
  - ✓ Layer *l+1*은 Window Partitioning이 이동되어 새로운 Window를 생성(Shifted Window 결과)



DMQA

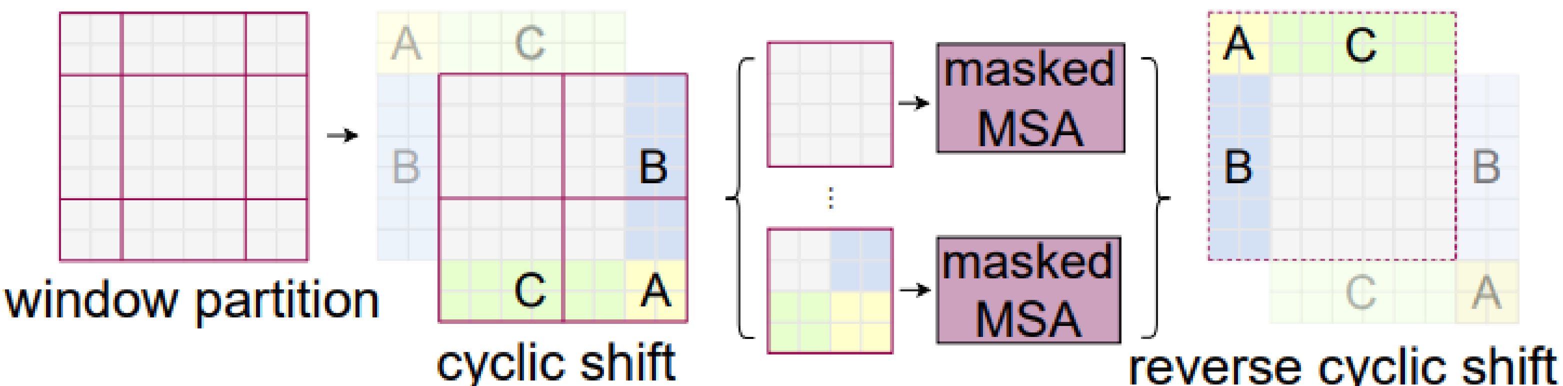
A Baseline SSL Method with Swin Transformers

Swin Transformer as the backbone



❖ Overall Architecture – Swin Transformer Block

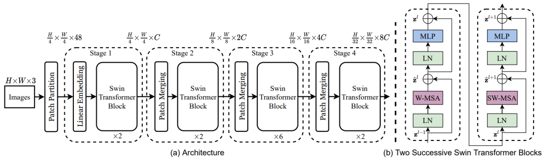
- SW-MSA 연산은 **1) Cyclic Shift**로 Window를 Shift 시킴
- Window Size / 2 만큼 우측 하단으로 Shift하고 Shift된 A, B, C 구역에 **2) Mask**를 씌워 Self-Attention을 하지 못하도록 함
  - ✓ 원래 좌측 상단에 있던 정보이기 때문에 Self-Attention 연산의 의미가 없음
- **3) Reverse Cyclic Shift**: Mask 연산을 한 후 다시 원래 값으로 되돌림



DMQA

A Baseline SSL Method with Swin Transformers

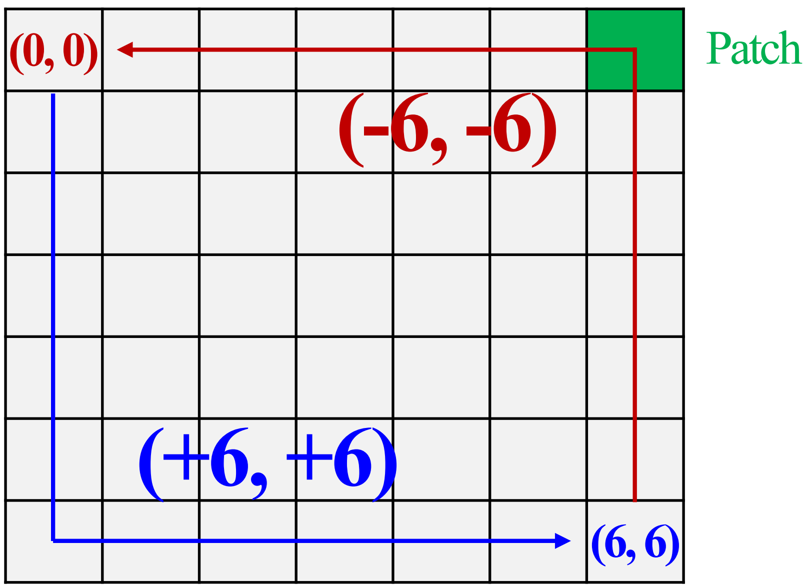
Swin Transformer as the backbone



❖ Overall Architecture – Relative Position Bias

- Swin Transformer는 ViT와 다르게 Positional Encoding 정보를 더하지 않음
- Self-Attention 연산 과정에서 Relative Position Bias를 추가함(아래 수식에서 **B**를 의미)
  - ✓ 논문에서는 실험을 통해서 상대 좌표를 더해주는 것이 좋은 방법임을 증명

$$\text{Attention}(Q, K, V) = \text{SoftMax}(QK^T/\sqrt{d}\boxed{+ B})V,$$



- Window Size: 7 X 7 (Patch 구성, 8 page 참고)
- (0, 0) Patch에서 (6, 6) Patch로 이동 시 **(+6, +6)**만큼 이동
- (6, 6) Patch에서 (0, 0) Patch로 이동 시 **(-6, -6)**만큼 이동
- Patch 중심 기준에 따라 이동해야 하는 값이 달라짐
- 따라서, 각 축 범위 [-6, 6] 기준으로 상대적 좌표를

  Embedding하는 것이 효과적

DMQA

A Baseline SSL Method with Swin Transformers

MoBY: A Self-Supervised Learning Approach

❖ MoBY

- **Mo**Co v2와 **BY**OL을 결합한 새로운 Self-Supervised Learning 방법

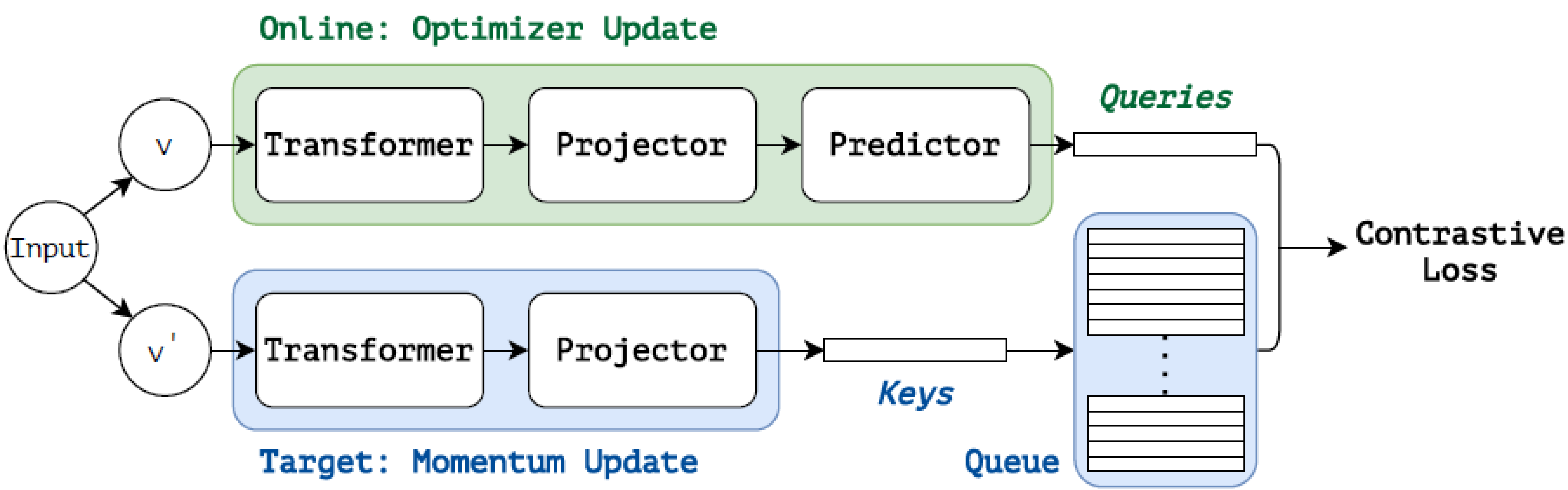


Figure 1: The pipeline of MoBY.

DMQA

A Baseline SSL Method with Swin Transformers

MoBY: A Self-Supervised Learning Approach

❖ MoBY

- **Mo**Co v2에서 사용한 기법들을 차용
  - ✓ Momentum design (Target Network Update)
  - ✓ Memory Queue (keys)
  - ✓ Contrastive Loss (InfoNCE)

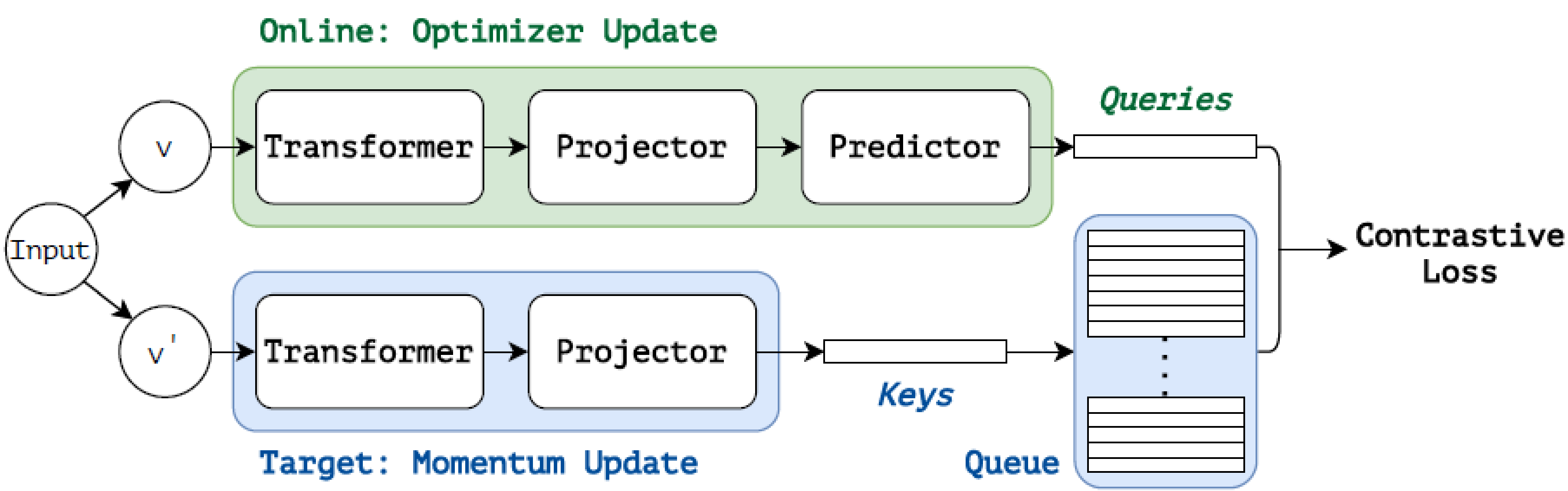


Figure 1: The pipeline of MoBY.

A Baseline SSL Method with Swin Transformers

MoBY: A Self-Supervised Learning Approach

❖ MoBY

- **BY**OL에서 사용한 기법들을 차용
  - ✓ Asymmetric Encoders
  - ✓ Asymmetric Data Augmentations
  - ✓ Momentum Scheduler

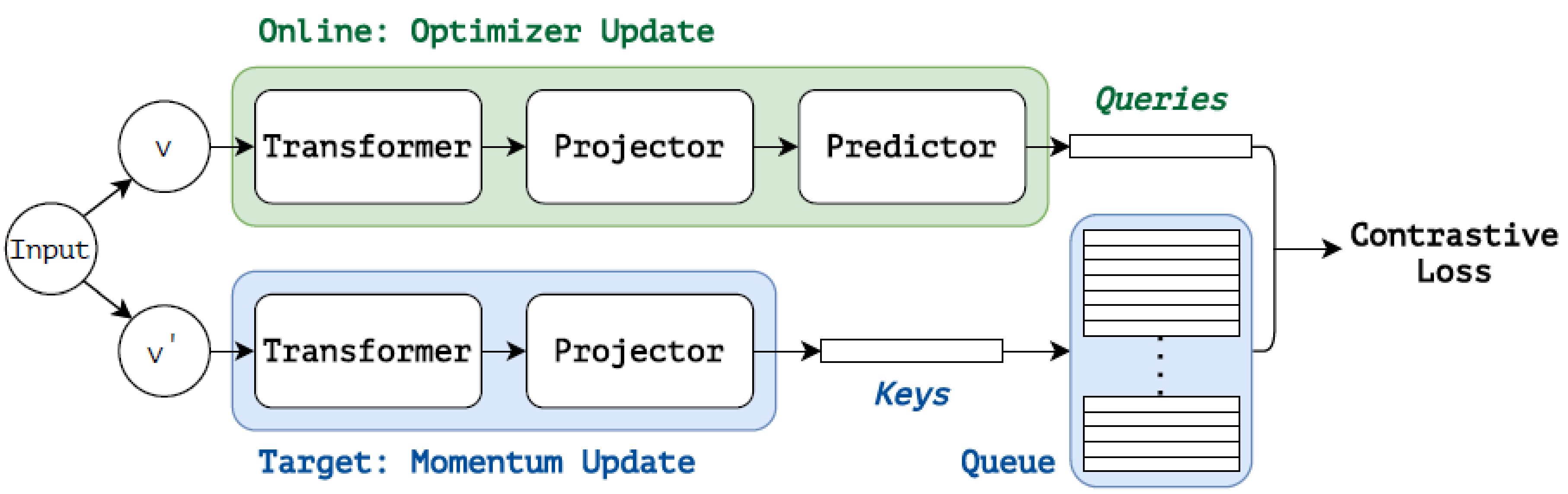


Figure 1: The pipeline of MoBY.

A Baseline SSL Method with Swin Transformers

MoBY: A Self-Supervised Learning Approach

- MoBY
  - Contrastive Loss
    - ✓ q: online view (query)
    - ✓ k_+: target view (positive key)
    - ✓ k_i: target feature in key queue
    - ✓ K: size of the key queue (default: 4096)
    - ✓ τ: temperature
    - ✓ Attract Positive Pair & Repel Negative Examples

$$\mathcal{L}_q = -\log \frac{\exp(q \cdot k_+ / \tau)}{\sum_{i=0}^{K} \exp(q \cdot k_i / \tau)}$$

A Baseline SSL Method with Swin Transformers

MoBY: A Self-Supervised Learning Approach

❖ Pseudo Code of MoBY in PyTorch-like Style

```
# encoder: transformer-based encoder
# proj: projector
# pred: predictor
# odpr: online drop path rate
# tdpr: target drop path rate
# m: momentum coefficient
# t: temperature coefficient
# queue1, queue2: two queues for storing negative samples

f_online = lambda x: pred(proj(encoder(x, drop_path_rate=odpr)))
f_target = lambda x: proj(encoder(x, drop_path_rate=tdpr))

for v1, v2 in loader: # load two views
    q1, q2 = f_online(v1), f_online(v2) # queries: NxC
    k1, k2 = f_target(v1), f_target(v2) # keys: NxC

    # symmetric loss
    loss = contrastive_loss(q1, k2, queue2) + contrastive_loss(q2, k1, queue1)
    loss.backward()

    update(f_online) # optimizer update: f_online
    f_target = m * f_target + (1. - m) * f_online # momentum update: f_target
    update(m) # update momentum coefficient

def contrastive_loss(q, k, queue):
    # positive logits: Nx1
    l_pos = torch.einsum('nc,nc->n', [q, k.detach()]).unsqueeze(-1)
    # negative logits: NxK
    l_neg = torch.einsum('nc,ck->nk', [q, queue.clone().detach()])

    # logits: Nx(1+K)
    logits = torch.cat([l_pos, l_neg], dim=1)

    # labels: positive key indicators
    labels = torch.zeros(N)
    loss = F.cross_entropy(logits / t, labels)

    # update queue
    enqueue(queue, k)
    dequeue(queue)

    return loss
```


DMQA

Experiments

❖ Image Classification Metric

- Top-1 Accuracy: Softmax의 Output에서 제일 높은 수치를 가지는 값이 정답일 경우에 대한 지표
- Float Point Operations Per Second (FLOPs): 컴퓨터의 성능을 표현하는 지표
- Parameters: Model의 Weight 또는 Parameter 수

❖ Object Detection Metric

- Average Precision (AP): IoU 계산 결과 값이 0.5 이상이면 True Positive (TP), 0.5 미만이면 False Positive (FP)로 판단하고 검출 결과들 중 옳게 검출한 비율을 의미(정확도)

❖ Semantic Segmentation Metric

- Mean Intersection over Union (mIoU): 예측 및 실제 픽셀 간 교집합에 포함되는 정도에 대한 지표
- Float Point Operations Per Second (FLOPs): 컴퓨터의 성능을 표현하는 지표

DMQA

Experiments

Linear evaluation

❖ Comparison of different SSL methods and different Transformer architectures on ImageNet-1K

- Transformer architecture: DeiT, Swin Transformer
- SSL methods: MoCo v3, DINO, MoBY

Method	Arch.	Epochs	Params (M)	FLOPs (G)	img/s	Top-1 acc (%)
Sup.	DeiT-S	300	22	4.6	940.4	79.8
Sup.	Swin-T	300	29	4.5	755.2	81.3
MoCo v3	DeiT-S	300	22	4.6	940.4	72.5
DINO	DeiT-S	300	22	4.6	940.4	72.5
DINO†	DeiT-S	300	22	4.6	940.4	75.9
MoBY	DeiT-S	300	22	4.6	940.4	72.8
MoBY	Swin-T	100	29	4.5	755.2	70.9
MoBY	Swin-T	300	29	4.5	755.2	**75.0**

Experiments

Object Detection Results

❖ Object Detection on COCO

- ImageNet-1K Classification 지도학습의 Backbone vs. MoBY 학습의 Backbone
- MoBY가 지도학습에 준하는 성능을 보임

Method	Model	Schd.	box AP			mask AP		
			mAP^{bbox}	AP_{50}^{bbox}	AP_{75}^{bbox}	mAP^{mask}	AP_{50}^{mask}	AP_{75}^{mask}
Swin-T (mask R-CNN)	Sup.	1x	43.7	66.6	47.7	39.8	63.3	42.7
	MoBY	1x	43.6	66.2	47.7	39.6	62.9	42.2
	Sup.	3x	46.0	68.1	50.3	41.6	65.1	44.9
	MoBY	3x	46.0	67.8	50.6	41.7	65.0	44.7
Swin-T (Cascade mask R-CNN)	Sup.	1x	48.1	67.1	52.2	41.7	64.4	45.0
	MoBY	1x	48.1	67.1	52.1	41.5	64.0	44.7
	Sup.	3x	50.4	69.2	54.7	43.7	66.6	47.3
	MoBY	3x	50.2	68.8	54.7	43.5	66.1	46.9

Experiments

Semantic Segmentation Results

- Semantic Segmentation on ADE20K
  - 지도학습 vs. MoBY
  - MoBY가 지도학습에 준하는 성능을 보임

Method	Model	Schd.	mIoU
Swin-T (UPerNet)	Sup.	160K	44.51
	MoBY	160K	44.06
	Sup.†	160K	45.81
	MoBY†	160K	45.58

Experiments

Ablation Study – linear evaluation

❖ Drop Path Rates (DPR) of online and target encoders

- DPR은 유용한 regularization로써 Online encoder의 DPR을 0.1, 0.2로 두었을 때 좋은 성능을 보임

❖ Memory Queue Size, Temperature, Momentum

Epochs	*Online* dpr	*Target* dpr	Top-1 acc (%)
100	0.05	0.0	70.9
100	0.1	0.0	70.9
100	0.2	0.0	70.9
100	0.1	0.1	69.0
300	0.05	0.0	74.2
300	0.1	0.0	75.0
300	0.2	0.0	75.0

K	Top-1 (%)
1024	71.0
2048	70.8
4096*	70.9
8192	71.0
16384	70.8

(a) Queue Size K

τ	Top-1 (%).
0.07	62.7
0.1	67.7
0.2*	70.9
0.3	70.8

(b) Temperature τ

Start value	Top-1 (%)
0.99*	70.9
0.993	70.7
0.996	70.5
0.999	67.6

(c) Momentum of *target* encoder

DMQA

Conclusion

❖ Self-Supervised Learning (SSL)과 Transformer-based Architecture Backbone의 결합 연구

❖ **General-Purpose**의 Swin Transformer Backbone을 활용하고 **Mo**Co v2와 **BY**OL을 결합한 **MoBY**를 제안

- 기존 ViT/DeiT 기반의 SSL 연구는 MoCo v3와 DINO

❖ Linear Evaluation, Object Detection, Semantic Segmentation에서 좋은 성능을 보임

- 후기: CNN Backbone이 아닌 Transformer Backbone의 SSL 연구로써 MoCo v3, DINO를 잇는 연구. 기존의 강력한 SSL 방법의 장점들을 결합하여 MoBY를 제안한 점이 인상적이었음.

❖ Reference

- Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., ... & Guo, B. (2021). Swin transformer: Hierarchical vision transformer using shifted windows. arXiv preprint arXiv:2103.14030.
- Xie, Z., Lin, Y., Yao, Z., Zhang, Z., Dai, Q., Cao, Y., & Hu, H. (2021). Self-supervised learning with swin transformers. arXiv preprint arXiv:2105.04553.

Thank You