카테고리 없음

[Euron] 11week_ Swin Transformer

yejjji 2025. 5. 18. 13:58

0. Abstract

본 논문은 **Swin Transformer라는 새로운 vision Transformer를 제시한다.**
Transformer를 언어부터 비전까지 적용하는 과정에서, 2개의 도메인들 간의 차이에 의해 (시각 이미지의 크기와 단어들과 비교되는 이미지 픽셀들의 고차원 해상도) 발생하는 문제들이 생겼다.

이러한 차이들을 해결하기 위해, shifted windows로 계산되는 계층적 Transformer를 제안한다.
Shifted Windows 방식은 self-attention 계산 시 겹치지 않는 local window로 제한되어 효율성을 높이는 동시에,
서로 다른 window간의 연결도 허용한다.

이러한 계층적 아키텍처는 다양한 스케일에서 모델링할 수 있는 유연성을 가지며, **이미지 크기에 대해 선형적인 계산 복잡도**를 가진다.

Swin Transformer의 이런 특성은 이미지 분류, 객체 탐지, 의미론적 분할과 같은 다양한 vision 작업에 적합하다.
특히나 성능의 경우 이전에 존재하던 최고 수준의 모델보다 큰 폭보다 앞선다.

- 이미지 분류: ImageNet-1K에서 87.3% top-1 accuracy
- 객체 탐지: COCO에서 58.7 box AP & 51.1 mask AP
- 의미론적 분할 : ADE20K val에서 53.5 mIoU

특히나 객체 탐지 작업에서 COCO 데이터셋에서 box AP는 기존보다 2.7, mask AP는 기존보다 2.6이 증가하였고.
의미론적 분할 과정에서는 ADE20K 데이터셋에서 mIoU가 3.2가 기존보다 향상되었다.

이러한 결과는 Transformer 기반의 모델이 vision backbone으로서 가지는 가능성을 보여주며,
계층적 설계와 shifted windows 방식은 MLP 구조에서도 유용한 것으로 나타났다.

1. Introduction

(컴퓨터 비전)
컴퓨터 비전 분야에서는 오랫동안 CNN이 주요 모델로 사용되어 왔다.
특히 AlexNet이 ImageNet 대회에서 혁신적인 성과를 거둔 이후,
CNN은 규모 확대, 연결 구조 개선, 그리고 복잡한 합성곱 방식을 도입하는 과정을 통해 점점 더 강력해졌다.
이러한 발전으로 CNN은 다양한 비전 작업의 기본 backbone으로 사용되며, 전체 컴퓨터 비전 분야의 성능 향상으로 이어졌다.

(자연어 처리)
반면에 자연어 처리 분야에서는 Transformer가 중심 아키텍처로 발전해 왔다.

Transformer는 sequence data 처리에 적합하게 설계되며, attention 매커니즘을 통해 장기적인 의존 관계를 효과적으로 모델링한다.
Transformer는 언어 분야에서 큰 성공을 거두며, 이를 계기로 컴퓨터 비전에 적용하려는 연구가 활발히 진행되었다.
특히나 이미지 분류와 vision-language 통합 모델링 분야에서 유망한 성과를 보이고 있다.

이를 바탕으로, Transformer를 자연어 처리처럼 컴퓨터 비전에서도 범용적인 backbone으로 사용할 수 있도록 확장하고자 한다.
하지만 language와 vision 간의 차이 때문에 성능을 그대로 옮기기에는 어렵다. 대표적인 차이로는,

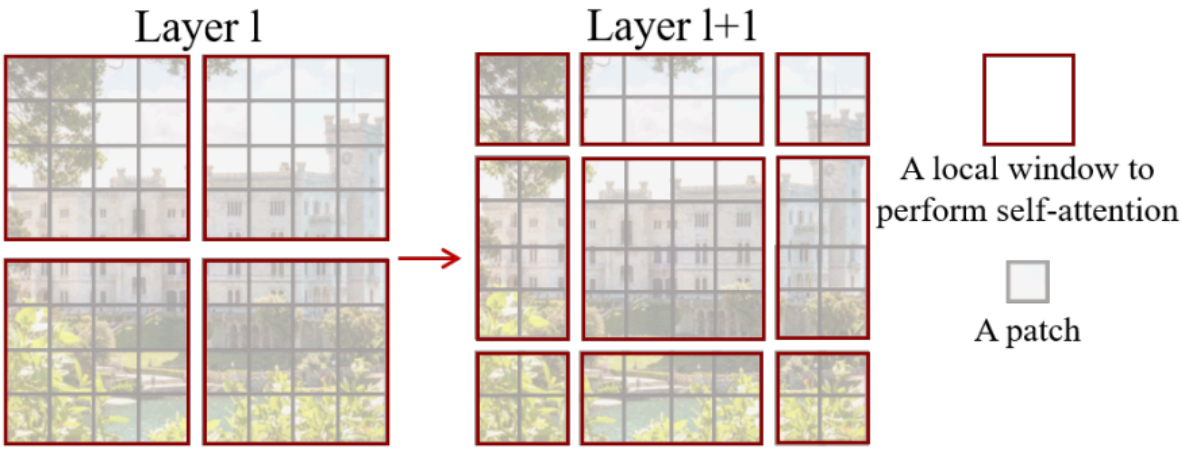
1. 스케일의 다양성: language에서는 단어 토큰이 일정한 크기이지만, vision에서는 객체나 특징의 크기가 다양하다.
2. 고해상도 처리 문제: 이미지의 픽셀 수는 텍스트보다 훨씬 많아, 기존의 Transformer의 self-attention의 계산 복작도가 이미지 크기에 대해 Quadratic하게 증가하여 고해상도에서 비효율적이다.

이 문제들을 해결하기 위해 논문은 **Swin Transformer**를 제안한다. 주요 특징은 다음과 같다.

1. 계층적 특징 맵 구성: 작은 패치에서 시작해 층이 깊어지면서 점차 주변 패치를 병합하면서 계층적 표현을 생성한다.
2. 선형 계산 복잡도: 이미지 전체가 아닌 **겹치지 않는 윈도우 단위로 self-attention을 수행하며 각 window의 패치수가 고정되면서** 복잡도가 이미지 크기에 비례하도록 설계하였다.

계층적 특징맵과 선형 계산 복잡도 덕분에 Swin Transformer는 **FPN, U-Net 같은 고밀도 예측 기법들과 쉽게 통합**될 수 있으며, **다양한 비전 작업에 적합한 범용 backbone** 역할이 가능하다.

Swin Transformer의 핵심 설계 요소 중 하나는 연속적인 self-attention층 사이에서 window 분할을 shift시키는 것이다.



Shifted Window는 이전 층의 윈도우들을 연결하여 **모델링 능력을 크게 향상**시킨다.

또한, 이 방식은 **실제 하드웨어 환경에서 latency 측면에서도 효율적**인데, window 내의 모든 쿼리 패치가 **동일한 키 세트**를
공유하므로, **메모리 접근이 간편**해지게 된다.
반면 기존의 슬라이딩 window 기반의 self-attention 방식은 각 쿼리 픽셀마다 키 세트가 달라 일반 하드웨어에서 latency가 길다.

실험 결과, **제안된 shifted-window 방식은 sliding window 방식보다 훨씬 낮은 latency을 가지면서, 모델링 성능에는 별 차이가 없다는 사실을 알 수 있었다.**

Swin Transformer는 이미지 분류, 객체 탐지, 의미론적 분할 등 다양한 인식 작업에서 우수한 성능을 달성하였다.
ViT / DeiT , ResNe(X)t 와 비슷한 latency을 가지면서도, 더 높은 정확도를 기록하였다.

1. COCO test-dev: 58.7 box AP / 51.1 mask AP
→ 각각 기존 최고 성능보다 +2.7 box AP (Copy-paste [26]), +2.6 mask AP (DetectoRS [46]) 향상되었다.

2. ADE20K 의미론적 분할(val): 53.5 mIoU, 기존 최고 성능(SETAR [81])보다 +3.2 mIoU 향상되었다.
3. ImageNet-1K 이미지 분류: 87.3% top-1 accuracy를 가진다.

논문의 저자들은 비전과 자연어 처리(NLP)를 아우르는 통합 아키텍처가 양쪽 분야에 모두 이득이 될 수 있다고 믿는다.
이런 통합은 시각 및 텍스트 신호의 공동 모델링을 가능하게 하고, 두 분야의 지식 공유를 더욱 심화시킬 수 있기 때문이다.

2. Related Work

CNN and variants

CNN과 그 변형 모델들은 오랫동안 컴퓨터 비전에서 표준 backbone으로서 사용되었다.
AlexNet의 성공 이후, VGG, GoogleNet, ResNet, DenseNet, HRNet, EfficientNet 등 더 깊고 효율적인 CNN 구조들이
제안되었고, 합성곱 층 자체도 Depthwise Convolution, Deformable Convolution 등으로 발전하였다.
CNN이 여전히 주요한 backbone이지만, 비전과 언어를 통합할 수 있는 Transformer 기반 구조의 잠재력에 주목하고 있다.

Self-attention based backbone architectures

자연어 처리 분야에서 self-attention 층과 Transformer 아키텍처가 성공하였다.
이와 관련하여 ResNet의 기존의 합성곱층의 일부 혹은 전체를 self-attention으로 대체하고자 하는 시도들이 전개되었다.

최적화 과정을 빠르게 하기 위해, 각 픽셀의 local window 내에서 Self-Attention을 수행하였다.

기존의 ResNet과 비교하면 약간의 성능 향상과 효율적 FLOPs를 보였지만,
높은 메모리 접근 비용으로 실제 latency가 크다는 것을 확인할 수 있었다.
큰 latency를 보완하고자, Swin Transformer는 sliding window 대신에 Shifted Window 방식을 사용하여,
더 나은 하드웨어 효율성과 성능을 달성하였다.

Self-attention / Transformers to complement CNNs

일부 연구는 **CNN backbone에 Self-Attention 층 또는 Transformer를 추가**하여
원거리 의존성이나 이질적인 정보 간의 다양한 상호작용을 모델링하려 시도하였다.

최근에는 Transformer의 인코더-디코더 구조를 객체 탐지, 인스턴스 세분화 등에 적용하는 흐름도 있는데,
논문의 연구는 Transformer 자체를 기본 시각 특성 추출기로 적응시키는 것을 목표로 하며,
위 연구들과 보완적인 관계를 가진다.

Transformer based vision backbones

ViT는 중간 크기의 이미지 패치에 Transformer를 직접 적용함으로써 이미지 분류에서 속도-정확도 균형을 달성하였다.
하지만 ViT는 대규모 데이터셋(JFT-300M)이 필요하기에 이를 보완하고자 Deit를 적용하였다.
Deit는 Vit가 더 적은 양의 훈련 데이터셋으로 효율적으로 활용하여 훈련이 가능하게 하였다.

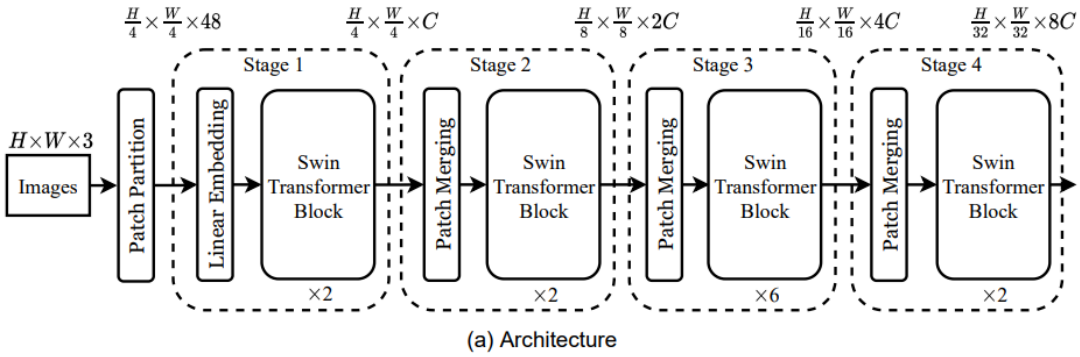
하지만, 낮은 해상도 특징 맵, 복잡도는 이미지 크기에 제곱되기 때문에 고해상도 이미지나 dense prediction에는 적합하지 않다.
일부 연구는 ViT를 dense prediction에 적용했지만 성능이 낮았다.

Swin Transforer는 위 문제들에 대응하여 다중 해상도 특징 을 구축하고, Shifted Window 방식을 적용하였다.
이 방법을 통해 이미지 크기에 대해 선형 복잡도를 가지게 되고, local self-attention이 시각 신호의 높은 상관 관계를 잘 모델링한다.

COCO 객체 탐지, ADE20K 의미 분할 등에서도 최신의 최고 성능을 달성하였고, 이미지 분류에서도 속도-정확도 균형에서 가장 우수한 결과를 보였으며, 일반 backbone으로도 효과적임을 확인하였다.

3. Method

3.1 Overall Architecture



Swin Transformer의 전체 아키텍처

가장 먼저, 입력된 RGB 이미지를 겹치지 않는 4 X 4 크기의 패치로 나눈다.
각 패치는 하나의 토큰으로 간주되며, 4 X 4 X 3 = 48차원의 RGB 픽셀값을 사용하게 된다.

추가적으로 각 패치의 픽셀 값을 임의의 차원인 c로 선형 변환 임베딩하여 사용한다.

Swin Transformer는 위 그림에서 볼 수 있듯이 총 4개의 Stage로 구성되며, 점진적으로 해상도를 줄이며 특징을 추출한다.

1. Stage 1: 입력 패치에 Transformer 블록 적용한다.

- 해상도: H/4 × W/4
- 특징 수 유지됨

2. Stage 2: 2×2 이웃 패치들을 합쳐서 병합(Patch Merging)한다.
- 특징 차원: 4C → 2C
- 해상도: H/8 × W/8

3. Stage 3: Stage 2와 동일한 방식으로 병합한다.
- 해상도: H/16 × W/16

4. Stage 4: 최종 병합한다.
- 해상도: H/32 × W/32

이 구조는 VGG, ResNet과 유사한 **다단계 특징 맵을 제공하여 기존의 비전 모델들과 쉽게 통합이 가능하다**는 것이 특징이다.

Swin Transformer block

Swin Transformer는 기존의 Transformer 블록의 MSA를 shifted window 기반의 MSA로 대체한다.

각 Swin Transformer 블록은
LN -> shifted window 기반의 MSA -> 잔차 연결 -> LN -> 2-layer MLP + GELU 활성화 -> 잔차 연결 로 구성된다.

이러한 구조는 window 기반의 Attention 연산으로 **복잡도는 줄이고 성능은 유지하는 효과**를 가진다.

3.2 Shifted Window based Self-attention

기존의 Transformer는 전역(Self-Attention) 연산을 수행하여, 모든 토큰 간의 관계를 계산한다.
이로 인해, 계산량이 토큰 수의 Quadratic 배가 되어 고해상도 이미지나 픽셀 수준 예측이 필요한 비전 작업에서는 비효율적이다.

Self-attention in non-overlapped windows

위 문제를 해결하고자 local window 기반의 self-attention 방법을 활용한다.
이미지를 겹치지 않는 M×M 윈도우로 분할하고, 각 윈도우 내부에서만 Self-Attention 수행하는 방법이다.
계산 복잡도는

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

- 기본 설정: M = 7

MSA의 경우 hw 값이 클 때에는 합리적이지 않다.
이에 반해 W-MSA의 경우 확장성 및 속도에서 우수하다.

Shifted window partitioning in successive blocks

단순 window 기반의 Self-Attention은 서로 다른 윈도우 간 정보 교환이 없어 모델의 표현력에 한계가 존재한다.

이를 해결하기 위해 **이전 블록의 윈도우 배치에서 (⌊M/2⌋, ⌊M/2⌋) 픽셀만큼 윈도우를 shift시킨다.**
이를 통해 서로 다른 윈도우에 속한 토큰들이 다음 블록에서는 같은 윈도우에 포함되도록 한다.

$$\hat{\mathbf{z}}^l = \text{W-MSA}\left(\text{LN}\left(\mathbf{z}^{l-1}\right)\right) + \mathbf{z}^{l-1},$$
$$\mathbf{z}^l = \text{MLP}\left(\text{LN}\left(\hat{\mathbf{z}}^l\right)\right) + \hat{\mathbf{z}}^l,$$
$$\hat{\mathbf{z}}^{l+1} = \text{SW-MSA}\left(\text{LN}\left(\mathbf{z}^l\right)\right) + \mathbf{z}^l,$$
$$\mathbf{z}^{l+1} = \text{MLP}\left(\text{LN}\left(\hat{\mathbf{z}}^{l+1}\right)\right) + \hat{\mathbf{z}}^{l+1},$$

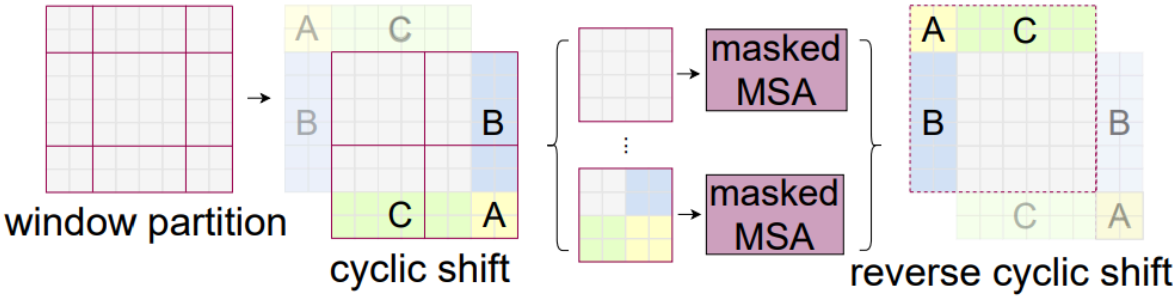
연산 순서

연산 순서는 위 식 순서와 같다.
W-MSA는 non-shifted window를 기반으로 하고, SW-MSA의 경우 shifted window를 기반으로 한다.

Shifted window partitioning으로 이웃한 window간의 정보가 연결되고, 이로 인해 모델의 표현력이 증가한다.
특히나, 이미지 분류, 객체 탐지, 의미 분할 작업에서 성능이 모두 향상된다.

Efficient batch computation for shifted configuration

Shifted window는 가장자리에 작은 window를 만들어 처리 복잡도 증가한다.



이를 해결하기 위해 Top-left 방향으로 cyclic shift하여 연산량을 유지하고,
마스킹 기법을 이용하여 연산이 window 내부에서만 수행된다.

덕분에 Shifted Window 구조에서도 효율적 연산 가능해지고, 낮은 지연 시간을 확보할 수 있다.

Relative position bias

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

기존의 위치 임베딩(절대 위치 임베딩) 보다 성능이 좋은 방식이다.

* 절대 위치 임베딩 : 고정된 구조 -> 창 크기 & 해상도 변화에 유연적이지 않음
* 상대 위치 bias : 위치 차이에 기반함 -> 변형에 더 유연하고, 일반화에 강함
-> Bicubic Interpolation : 위치 bias를 부드럽게 스케일링하여 새 크기에 맞춤

특히 절대 위치 임베딩의 경우 성능이 오히려 하락하므로 사용하지 않는다.

3.3 Architecture Variants

Swin Transformer는 다양한 크기의 모델을 제공하여, 목적에 맞는 성능/속도 균형을 선택할 수 있도록 설계되었다.

버전	채널 수	각 스테이지의 블록 수	모델 크기	비교 대상
Swin-T	96개	{2, 2, 6, 2}	0.25x	ResNet-50 / Deit-S
Swin-S	96개	{2, 2, 18, 2}	0.5x	ResNet-101
Swin-B	128개	{2, 2, 18, 2}	1x	Vit-B / DeiT-B
Swin-L	192개	{2, 2, 18, 2}	2x	-

Swin Transformer는 ViT와 DeiT와 유사한 Transformer 기반의 구조로,
계층적이고 효율적인 window 기반의 Self-Attention으로 설계되었다.

4. Experiments

4.1. Image Classification on ImageNet-1K

<Settings>

1. Regular ImageNeet-1K training

- Optimizer: AdamW
- Epoch: 300 (warm-up 20)
- LR: 초기 0.001 → cosine decay
- Batch size: 1024
- Augmentation: RandAug, Mixup, Cutmix, Label Smoothing 등 사용

(단, Repeated Augmentation, EMA는 사용하지 않음)

2. Pre-training on ImageNet-22K and fine-tuning on ImageNet-1K

- Pre-training: 90 Epochs / Batch size 4096 / LR 0.001 / weight decay 0.01
- Fine-tuning: 30 Epochs / Batch size 1024 / LR 1e-5 / weight decay 1e-8

Results with regular ImageNet-1K training

(a) Regular ImageNet-1K trained models					
method	image size	#param.	FLOPs	throughput (image / s)	ImageNet top-1 acc.
RegNetY-4G [48]	224^2	21M	4.0G	1156.7	80.0
RegNetY-8G [48]	224^2	39M	8.0G	591.6	81.7
RegNetY-16G [48]	224^2	84M	16.0G	334.7	82.9
EffNet-B3 [58]	300^2	12M	1.8G	732.1	81.6
EffNet-B4 [58]	380^2	19M	4.2G	349.4	82.9
EffNet-B5 [58]	456^2	30M	9.9G	169.1	83.6
EffNet-B6 [58]	528^2	43M	19.0G	96.9	84.0
EffNet-B7 [58]	600^2	66M	37.0G	55.1	84.3
ViT-B/16 [20]	384^2	86M	55.4G	85.9	77.9
ViT-L/16 [20]	384^2	307M	190.7G	27.3	76.5
DeiT-S [63]	224^2	22M	4.6G	940.4	79.8
DeiT-B [63]	224^2	86M	17.5G	292.3	81.8
DeiT-B [63]	384^2	86M	55.4G	85.9	83.1
Swin-T	224^2	29M	4.5G	755.2	81.3
Swin-S	224^2	50M	8.7G	436.9	83.0
Swin-B	224^2	88M	15.4G	278.1	83.5
Swin-B	384^2	88M	47.0G	84.7	84.5

위 그림의 Top-1 Accuracy를 기준으로 성능을 비교한다.

모델	Top-1 Accuracy	비교 모델	차이
Swin-T	81.3%	Deit-S (79.8%)	**+1.5%**
Swin-B	83.3%	DeiT-B (81.8%)	**+1.5%**
Swin-B (384^2)	84.5%	DeiT-B (384^2: 83.1%)	**+1.4%**

Swin은 비슷한 모델 크기에서 **ViT/DeiT보다 일관되게 우수한 성능을 가진다.**

또, RegNet, EfficientNet 등 최적화된 ConvNet 대비 속도-정확도 균형도 뛰어난다.

Results with ImageNet-22K pre-training

(b) ImageNet-22K pre-trained models

method	image size	#param.	FLOPs	throughput (image / s)	ImageNet top-1 acc.
R-101x3 [38]	384^2	388M	204.6G	-	84.4
R-152x4 [38]	480^2	937M	840.5G	-	85.4
ViT-B/16 [20]	384^2	86M	55.4G	85.9	84.0
ViT-L/16 [20]	384^2	307M	190.7G	27.3	85.2
Swin-B	224^2	88M	15.4G	278.1	85.2
Swin-B	384^2	88M	47.0G	84.7	86.4
Swin-L	384^2	197M	103.9G	42.1	87.3

위 그림의 Top-1 Accuracy를 기준으로 성능을 비교한다.

모델	Top-1 Accuracy	비고
Swin-B	86.4%	ImageNet-22K 사전학습 → 1.8~1.9% 향상
Swin-L	87.3%	Swin-B보다 +0.9% 향상

Swin-B 는 ViT와 비교했을 때 throughput은 유사하나, FLOPs는 더 작고 top-1 accuracy 는 더 높다.

4.2 Object Detection on COCO

< Settings >

1. 데이터셋: COCO 2017

- 학습 이미지: 118K
- 검증 이미지: 5K
- 테스트 이미지(test-dev): 20K

2. 사용된 프레임워크

(검증 세트)

- Cascade Mask R-CNN
- ATSS

(테스트 세트)

- HTC++ : 개선된 Hybrid Task Cascade

- RepPoints v2
- Sparse R-CNN

3. 공통 설정

- Multi-scale 학습 (짧은 변: 480~800, 긴 변: 최대 1333)
- Optimizer: AdamW
  - LR: 0.0001
  - Weight decay: 0.05
  - Batch size: 16
- 3x 스케줄 (36 에폭)

위 설정을 바탕으로 실험 결과를 비교하고자 한다.

Comparison to ResNe(X)t

(a) Various frameworks

Method	Backbone	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	#param.	FLOPs	FPS
Cascade Mask R-CNN	R-50	46.3	64.3	50.5	82M	739G	18.0
	Swin-T	**50.5**	**69.3**	**54.9**	86M	745G	15.3
ATSS	R-50	43.5	61.9	47.0	32M	205G	28.3
	Swin-T	**47.2**	**66.5**	**51.3**	36M	215G	22.3
RepPointsV2	R-50	46.5	64.6	50.3	42M	274G	13.6
	Swin-T	**50.0**	**68.5**	**54.2**	45M	283G	12.0
Sparse R-CNN	R-50	44.5	63.4	48.2	106M	166G	21.0
	Swin-T	**47.9**	**67.3**	**52.3**	110M	172G	18.4

Swin-T 와 ResNet-50을 비교한다.

Swin-T의 경우 모든 프레임워크에서 box AP의 향상폭이 3.4 ~ 4.2 AP 정도로, 모델의 크기 FLOPs, latency가 약간씩 증가하나

일관적으로 accuracy가 향상된다는 것을 확인할 수 있었다.

(b) Various backbones w. Cascade Mask R-CNN

	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}	param	FLOPs	FPS
DeiT-S†	48.0	67.2	51.7	41.4	64.2	44.3	80M	889G	10.4
R50	46.3	64.3	50.5	40.1	61.7	43.4	82M	739G	18.0
Swin-T	**50.5**	**69.3**	**54.9**	**43.7**	**66.6**	**47.1**	86M	745G	15.3
X101-32	48.1	66.5	52.4	41.6	63.9	45.2	101M	819G	12.8
Swin-S	**51.8**	**70.4**	**56.3**	**44.7**	**67.9**	**48.5**	107M	838G	12.0
X101-64	48.3	66.4	52.3	41.7	64.0	45.1	140M	972G	10.4
Swin-B	**51.9**	**70.9**	**56.5**	**45.0**	**68.4**	**48.7**	145M	982G	11.6

서로 다른 모델별 capacity를 기반으로 Swin과 ResNeXt를 비교한다.

특히나, ResNeXt101-64와 Swin-B를 비교하였을 때, box AP값이 3.6, mask AP값이 3.3 증가하였다는 사실을 확인할 수 있었다.

(c) System-level Comparison

Method	mini-val AP^{box}	mini-val AP^{mask}	test-dev AP^{box}	test-dev AP^{mask}	#param.	FLOPs
RepPointsV2* [12]	-	-	52.1	-	-	-
GCNet* [7]	51.8	44.7	52.3	45.4	-	1041G
RelationNet++* [13]	-	-	52.7	-	-	-
SpineNet-190 [21]	52.6	-	52.8	-	164M	1885G
ResNeSt-200* [78]	52.5	-	53.3	47.1	-	-
EfficientDet-D7 [59]	54.4	-	55.1	-	77M	410G
DetectoRS* [46]	-	-	55.7	48.5	-	-
YOLOv4 P7* [4]	-	-	55.8	-	-	-
Copy-paste [26]	55.9	47.2	56.0	47.4	185M	1440G
X101-64 (HTC++)	52.3	46.0	-	-	155M	1033G
Swin-B (HTC++)	56.4	49.1	-	-	160M	1043G
Swin-L (HTC++)	57.1	49.5	57.7	50.2	284M	1470G
Swin-L (HTC++)*	**58.0**	**50.4**	**58.7**	**51.1**	284M	-

ReseXt101-64x4d와 Swin-L 모델을 비교해보면, box AP의 경우 48.3과 51.9, mask AP의 경우 41.7과 45.0로,

Swin-L이 ResNeXt101 모델보다 box AP와 mask AP가 모두 향상되었다는 사실을 다시 확인할 수 있었다.

Comparison to DeiT

(b) Various backbones w. Cascade Mask R-CNN

	AP^{box}	AP^{box}_{50}	AP^{box}_{75}	AP^{mask}	AP^{mask}_{50}	AP^{mask}_{75}	param	FLOPs	FPS
DeiT-S†	48.0	67.2	51.7	41.4	64.2	44.3	80M	889G	10.4
R50	46.3	64.3	50.5	40.1	61.7	43.4	82M	739G	18.0
Swin-T	**50.5**	**69.3**	**54.9**	**43.7**	**66.6**	**47.1**	86M	745G	15.3
X101-32	48.1	66.5	52.4	41.6	63.9	45.2	101M	819G	12.8
Swin-S	**51.8**	**70.4**	**56.3**	**44.7**	**67.9**	**48.5**	107M	838G	12.0
X101-64	48.3	66.4	52.3	41.7	64.0	45.1	140M	972G	10.4
Swin-B	**51.9**	**70.9**	**56.5**	**45.0**	**68.4**	**48.7**	145M	982G	11.6

위에서 고려했던 표를 이번에는 Swin과 DeiT를 비교해보았다.

Backbone	Box AP	Mask AP	FPS	AP 향상
DeiT-S	41.5	37.4	10.4	-
Swin-T	44.0	39.7	15.3	**+2.5 / +2.3**

로 각 수치값을 정리해보았다.

결과적으로 Swin-T가 더 빠르고, 성능이 우수했으며, 실제로 DeiT는 입력 해상도에 따라 복잡도가 이차적으로 증가하는 문제가 존재하였다.

Comparison to previous state-of-the-art

(c) System-level Comparison

Method	mini-val AP^{box}	mini-val AP^{mask}	test-dev AP^{box}	test-dev AP^{mask}	#param.	FLOPs
RepPointsV2* [12]	-	-	52.1	-	-	-
GCNet* [7]	51.8	44.7	52.3	45.4	-	1041G
RelationNet++* [13]	-	-	52.7	-	-	-
SpineNet-190 [21]	52.6	-	52.8	-	164M	1885G
ResNeSt-200* [78]	52.5	-	53.3	47.1	-	-
EfficientDet-D7 [59]	54.4	-	55.1	-	77M	410G
DetectoRS* [46]	-	-	55.7	48.5	-	-
YOLOv4 P7* [4]	-	-	55.8	-	-	-
Copy-paste [26]	55.9	47.2	56.0	47.4	185M	1440G
X101-64 (HTC++)	52.3	46.0	-	-	155M	1033G
Swin-B (HTC++)	56.4	49.1	-	-	160M	1043G
Swin-L (HTC++)	57.1	49.5	57.7	50.2	284M	1470G
Swin-L (HTC++)*	**58.0**	**50.4**	**58.7**	**51.1**	284M	-

위에서 고려한 표를 다시 가져왔다.

이전의 최고 성능값인 box AP = 56.0, mask AP = 48.5와 비교하였을 때, Swin의 성능인

box AP = 58.7, mask AP = 51.1로

2개의 AP값이 모두 증가하였다는 사실을 확인하였다.

4.3 Semantic Segmentation on ADE20K

< Settings >

1. 데이터셋 : ADE20K

- 총 이미지: **25K**
  - 학습: 20K
  - 검증: 2K
  - 테스트: 3K
- 클래스 수: **150개**의 다양한 의미론적 카테고리

2. 사용 프레임워크 : UperNet -> 효율성과 성능을 동시에 제공하기 때문

실험 성능을 비교하였다.

ADE20K Method	Backbone	val mIoU	test score	#param.	FLOPs	FPS
DANet [23]	ResNet-101	45.2	-	69M	1119G	15.2
DLab.v3+ [11]	ResNet-101	44.1	-	63M	1021G	16.0
ACNet [24]	ResNet-101	45.9	38.5	-		
DNL [71]	ResNet-101	46.0	56.2	69M	1249G	14.8
OCRNet [73]	ResNet-101	45.3	56.0	56M	923G	19.3
UperNet [69]	ResNet-101	44.9	-	86M	1029G	20.1
OCRNet [73]	HRNet-w48	45.7	-	71M	664G	12.5
DLab.v3+ [11]	ResNeSt-101	46.9	55.1	66M	1051G	11.9
DLab.v3+ [11]	ResNeSt-200	48.4	-	88M	1381G	8.1
SETR [81]	T-Large‡	50.3	61.7	308M	-	-
UperNet	DeiT-S†	44.0	-	52M	1099G	16.2
UperNet	Swin-T	46.1	-	60M	945G	18.5
UperNet	Swin-S	49.3	-	81M	1038G	15.2
UperNet	Swin-B‡	51.6	-	121M	1841G	8.7
UperNet	Swin-L‡	**53.5**	**62.8**	234M	3230G	6.2

Swin-S의 경우 49.3 mIoU로 DeiT-S와 비교하였을 때 5.3 mIoU가 증가하였다.
Swin-L의 경우 53.5 mIoU로 기존의 최고 성능과 비교하였을 때 3.2 mIoU가 증가하였다는 사실을 확인하였다.

4.4. Ablation Study

Swin-Transformer의 주요한 구성 요소들에 대해 분석해보았다.
성능 검증을 위한 실험은 Image Classification, Object Detection, Semantic Segmentation에서 수행되었다.

	ImageNet		COCO		ADE20k
	top-1	top-5	AP^{box}	AP^{mask}	mIoU
w/o shifting	80.2	95.1	47.7	41.5	43.3
shifted windows	**81.3**	**95.6**	**50.5**	**43.7**	**46.1**
no pos.	80.1	94.9	49.2	42.6	43.8
abs. pos.	80.5	95.2	49.0	42.4	43.2
abs.+rel. pos.	81.3	95.6	50.2	43.4	44.0
rel. pos. w/o app.	79.3	94.7	48.2	41.9	44.1
rel. pos.	**81.3**	**95.6**	**50.5**	**43.7**	**46.1**

Ablation study

Shifted windows

- Image Classification : ImageNet-1K에서 1.1%의 top-1 accuracy를 달성하였다.
- Object Detection : COCO에서 box AP가 2.8, mask AP가 2.2가 향상하였다.
- Semantic Segmentation : ADE20K에서 2.8 mIoU가 향상되었다.

latency의 경우 오버헤드 정도가 매두 작았다.

Relative position bias

절대 위치 임베딩과 상대 위치 편향을 적용했을 때를 아무런 조건도 적용하지 않은 경우와 비교하였다.

먼저, 절대 위치 임베딩과 아무런 조건도 적용하지 않은 경우를 비교하면,
ImageNet-1K에서는 0.4%의 top-1 accuracy가 향상되었고, COCO와 ADE20K에서는 각각 -0.2와 -0.6의 mIoU 변화가 생겼다.

상대 위치 편향과 아무런 조건도 적용하지 않은 경우를 비교해보면,
ImageNet-1K에서는 1.2%, 0.8의 top-1 accuracy가 향상되었고, COCO와 ADE20K에서는 각각 +1.3(box AP), +1.5(mask AP)와 +2.3(box AP), +2.9(mask AP) 의 mIoU 변화가 생겼다.

즉, 절대 위치 임베딩의 경우 일부 작업에서는 성능 저하를 초래하고,
상대적 위치 편향의 경우 모든 작업에서 일관되게 성능을 향상 시킨다는 사실을 알 수 있었다.

Different self-attention methods

method	MSA in a stage (ms)				Arch. (FPS)		
	S1	S2	S3	S4	T	S	B
sliding window (naive)	122.5	38.3	12.1	7.6	183	109	77
sliding window (kernel)	7.6	4.7	2.7	1.8	488	283	187
Performer [14]	4.8	2.8	1.8	1.5	638	370	241
window (w/o shifting)	2.8	1.7	1.2	0.9	770	444	280
shifted window (padding)	3.3	2.3	1.9	2.2	670	371	236
shifted window (cyclic)	3.0	1.9	1.3	1.0	755	437	278

먼저, Cyclic shift 방식과 naive padding 방식을 비교하였다.
Cyclic shift 방식은 일반적인 padding보다 특히 더 깊은 단계들에서 더 hardware 효율적인 방식이다.
각 Swin 모델별(Swin-T, Swin-S, Swin-B)로 Cyclic shift 방식을 적용했을 때, +13%, +18%, +18%의 속도가 향상되었다.

		ImageNet		COCO		ADE20k
	Backbone	top-1	top-5	AP^{box}	AP^{mask}	mIoU
sliding window	Swin-T	81.4	95.6	50.2	43.5	45.8
Performer [14]	Swin-T	79.0	94.2	-	-	-
shifted window	Swin-T	81.3	95.6	50.5	43.7	46.1

다음으로, Sliding Window와 Shifted Window 방식을 비교하였다.
Shifted Windows 방식을 적용한 Swin-T, Swin-S, Swin-B의 전체 모델 효율을 비교하였을 때,
Swin-T의 경우 4.1/1.5, Swin-S의 경우 4.0/1.5, Swin-B의 경우 3.6/1.5배가 빠르다는 사실을 확인하였다.

Performer과 비교하였을 때에도, 속도가 조금 더 빠르며, 특히 Swin-T의 경우 top-1 accuracy가 Performer보다 +2.3%라는 사실을 확인할 수 있었다.

5. Conclusion

Swin Transformer는 새로운 비전 트랜스포머 아키텍처로서, 계층적인 특징 표현을 생성하고,
입력 이미지 크기에 대해 선형의 연산 복잡도를 가진다.

또, **객체 탐지 와 Semantic segmentation 작업에서 최신의 성능 달성하여 이전 최고 성능 모델들을 의미 있게 능가**하였다.

Swin Transformer는 shifted window 기반의 self-attention 을 통해, 시각 문제에서 효과적이고 효율적인 방식임을 입증하였고, 다양한 비전 문제에서의 이런 뛰어난 성능이향후 비전과 언어를 통합적으로 모델링하려는 연구에 도움이 될 것으로 기대된다.

Swin Transformer의 시프트 윈도우 어텐션 구조를 NLP 분야에도 적용해보는 것이 흥미로운 연구 주제가 될 것이다.

'카테고리 없음'의 다른글

이전글 [Euron] 10week_ Rainbow: Combining Improvements in Deep Reinforcement Learning

현재글 : [Euron] 11week_ Swin Transformer