An Image Is Worth 16x16 Words: Transformers For Image Recognition At Scale

Code	
Paper	https://arxiv.org/pdf/2010.11929.pdf
Year	2021
모델	ViT
속성	Computer Vision

https://www.facebook.com/groups/TensorFlowKR/permalink/1594970620843990/

흥미로운 피드 발견

https://medium.com/@hac541309/why-this-one-literally-small-model-spells-big-things-for-vision-transformers-fb4ef66ab13f

그렇구나 하고 읽어보려는데 ViT의 기초를 모름

Transformer Review

Sequential한 데이터는 섞이거나, 중간이 빠지거나 등등 문제 발생 원인이 많음

이를 해결하고자 LSTM, GRU등의 모델들이 제안되었으나 이 또한 거리에 대한 한계와 병렬처리의 어려움에 있어 학습속도가 느리다는 단점이 있음

이를 해결하고자 사용된 개념이 **Attention**

Transformer는 Attention으로 encoder-decoder를 여러층 쌓아서 구현한 구조

- **Self-attention**

  n개의 단어 vector x_i가 self-attention을 거쳐 z_i가 될 때,

  z_i는 문장 내부에서 x_i뿐 아니라 다른 x들을 다 참고하여 만들어짐

→ 문장 내부에서 현재의 단어 x_i과 관련 깊은 단어들을 참고하게 됨

encoder-decoder attetion과는 다르게 Q, K, V가 다 x_i로부터 생성되기 때문에 self 문장 내에서 attention을 찾고 output을 내기 때문에 self-attention이라는 이름이 붙었다 (?)

- $\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$

- **Multi-headed attention**

  하나의 문장 X에 대해서 Q, K, V를 하나만 만드는 것이 아니라 여러개 만듬

  → 하나의 X로 부터 여러개의 Z를 만들 수 있음

  - $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V),\ i = 1, \cdots, h$
  - $\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \cdots, \text{head}_h)W^O$
  - $W_i^Q, W_i^V, W_i^K \in \mathbb{R}^{d_{model} \times d_k}, W^O \in \mathbb{R}^{hd_v \times d_{model}}$
  - Q, K, V를 직접 사용할수도 있지만, 성능을 향상하기 위해서 h개의 학습가능한 parameters, W_i^Q, W_i^K, W_i^V,를 각각 Q, K, V를 project하는데 사용하고 위의 Attention 함수를 적용함
  - 그렇게 얻은 h개의 결과들을 모두 concatenate한뒤, 학습가능한 parameter, W^O를 이용해 원래 encoder input과 동일한 shape의 결과를 얻어냄
  - Multi-Head Attention을 사용함으로써 계산이 길어지는 문제가 발생할 수 있지만, head개수만큼 dimension을 축소해줌으로써 계산이 길어지지 않도록 해결함

An Image Is Worth 16x16 Words: Transformers For Image Recognition At Scale

INTRODUCTION

NLP 분야에서 Transformer 모델이 각광을 받고 있다. Transformer모델은 model과 dataset이 커짐에도 아직까지 performance가 정체되는 모습이 보이지 않는다.

이러한 NLP에서 영향을 받아 Computer Vision 분야에서도 CNN과 유사한 구조를 self-attention과 결합하여 좋은 결과를 내려는 시도가 많이 있었으나, 아직까지는 ResNet 같은 구조의 architecture들이 더 좋은 성능을 내고 있다.

본 논문에서는 NLP에서의 기본 Transformer 모델의 수정을 최소화하여 직접적으로 image를 입력으로 넣어 실험을 진행했다. 이미지를 patch들로 쪼개고 이 patch들을 NLP의 word처럼 입력으로 넣어주었다.

이러한 모델은 mid-sized dataset에 대해서 학습을 진행했을 때 (ImageNet), ResNet과 비교하여 몇 퍼센트 정도 낮은 정확도를 냈다. 이 정확도의 차이는 Transformer 모델은 CNN과 달리 inductive biases에 대한 기능이 없기 때문에 적은 data에 대해서 충분하게 일반화되지 못했기 때문이다.

▼ Inductive biases(귀납 편향)

참고 글

학습시에는 만나보지 않았던 상황에 대하여 정확한 예측을 하기위해 사용하는 추가적인 가정을 의미

CNN의 경우 translation equivariance와 locality를 inductive biases로 가짐

- Translation Equivariance : 어떠한 사물이 들어 있는 이미지를 제공해줄 때 사물의 위치가 바뀌면 CNN과 같은 연산의 activation 위치 또한 바뀌게 됩니다. 그렇지만 결과는 동일하게 나온다.
- locality : 이미지는 작은 특징들로 구성되어 있기 때문에 픽셀의 종속성은 특징이 있는 작은 지역으로 한정된다.

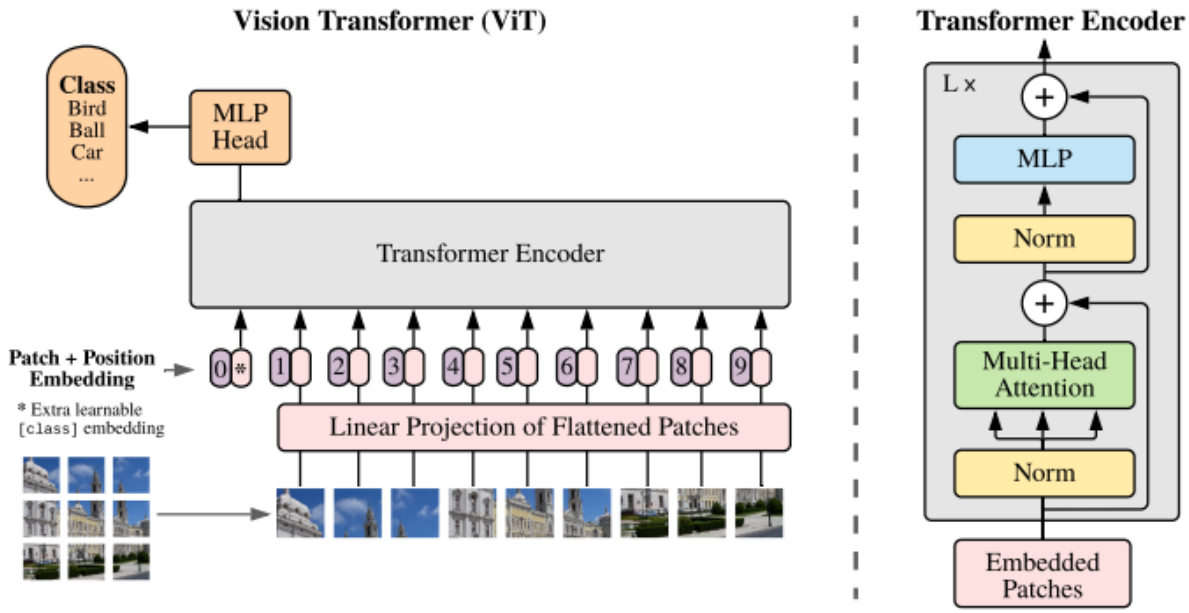
CNN과 Transformer의 진행을 생각해보면 CNN은 이미지가 약간 변형된다고 해도 convolution을 하며 스스로 필요한 영역을 묶고 결과적으로 Classification이 진행됨.

그러나 Transformer는 이미지가 약간 변형되면 input으로 들어가는 patch의 순서가 바뀜. 그런데 Transformer는 학습된 이미지들에 patch의 영향을 받아 Q, K, V를 만드는 Matrix가 만들어졌을 것. 그럼 patch의 순서가 바뀌었을 때 만들어진 Q, K, V와 결과 값이 충분히 신뢰할만 한가?

그러나 대규모의 dataset에 대해서 학습을 진행했을 때, inductive biases의 영향을 능가하는 성능을 낸다는 결론이 나온다. 다양한 dataset에 대해서 ViT 모델을 학습하고 결과를 비교했을 때 SOTA 모델들과 비슷하거나 능가하는 결과가 나왔다.

METHOD

3.1 Vision Transformer(ViT)



Vision Transformer

🍅모델 설명

- Split 2D Image

  $H \times W \times C$의 이미지를 N개의 $P \times P \times C$ 크기의 patch로 나눔

- Linear Projection of Flattened Patches

  마치 word embedding을 하는 것처럼 patch 상태 그대로 Transformer의 input으로 넣을 순 없으므로 patch를 flatten 해주는 과정

  trainable한 linear projection 방법을 사용

$$\mathbf{z}_0 = [\mathbf{x}_{\text{class}}; \mathbf{x}_p^1\mathbf{E}; \mathbf{x}_p^2\mathbf{E}; \cdots; \mathbf{x}_p^N\mathbf{E}] + \mathbf{E}_{pos}, \qquad \mathbf{E} \in \mathbb{R}^{(P^2 \cdot C) \times D}, \mathbf{E}_{pos} \in \mathbb{R}^{(N+1) \times D} \qquad (1)$$

- x_{class}

  BERT에서 [class]라는 token이 따로 존재하듯이 본 모델에서도 전체 token중 0번째 token은 class에 대한 예측을 학습하는 embedding으로 주어져있음

  그러므로 z_L^0 은 Transformer의 마지막 layer의 class token이므로 해당 token을 입력으로 MLP를 진행하여 classification을 함

- Position embedding

이미지를 split하여 순서대로 넣어버렸기 때문에 이미지의 위치 정보에 대한 이점을 받을 수 없으므로 Position embedding을 추가

- Transformer encoder

  multiheaded self-attention과 MLP의 조합으로 이루어져 있음

  Layer Normalization이 매 block전에 진행되며

  Residual connection이 매 block마다 존재

  MLP는 2개의 layer로 이루어져 있으며 GELU가 사용됨(GELU란)

$$\mathbf{z}'_\ell = \mathrm{MSA}(\mathrm{LN}(\mathbf{z}_{\ell-1})) + \mathbf{z}_{\ell-1}, \qquad \ell = 1 \dots L \tag{2}$$

$$\mathbf{z}_\ell = \mathrm{MLP}(\mathrm{LN}(\mathbf{z}'_\ell)) + \mathbf{z}'_\ell, \qquad \ell = 1 \dots L \tag{3}$$

$$\mathbf{y} = \mathrm{LN}(\mathbf{z}_L^0) \tag{4}$$

🍅특징

- Inductive bias

  ViT는 CNN보다 image specific한 inductive bias가 부족함

  CNN은 locality, two-dimensional neighborhood structure, translation equivariance가 모델 전체에 깔려있지만

  ViT는 MLP layer에서만 local, translationally equivariant함

- Hybrid Architecture

  이미지를 patch로 분할하는 대신 CNN의 feature map을 사용하여 input sequence를 만드는 방법

  (1)식의 E를 CNN feature map에서 추출해낸 patch에 적용

3.2 Fine-tuning and higher resolution

저자는 ViT를 대규모의 dataset에서 pre-train 시키고 적은 task에 대해서 fine-tune을 했다.

Fine-tuning을 위해서는 pre-trained된 prediction head를 제거하고 그 자리에 0으로 초기화 된 feed forward layer를 넣어주어야한다.

이 때 pre-training 때보다 더 좋은 화질에 대해서 fine-tune을 해주는 것이 더 좋은 결과를 낼 수 있다.

왜냐하면, 더 좋은 resolution을 pre-trained 된 ViT에 넣게되면 patch의 크기는 고정 시키며, sequence length는 효과적으로 만든다(?)

EXPERIMENTS

본 논문에서는 ResNet, ViT, hybrid의 성능을 실험하고 비교했다.

또 마지막으로 ViT의 self-supervision 실험을 진행해 보았다.

4.1 Setup

- Dataset

  pre-train : JFT, ImageNet-21k

  fine tune : ImageNet, ImageNet ReaL, CIFAR-10, CIFAR-100, Oxford-IIIT Pets, Oxford FLowers-102, VTAB

- Model Variants

  BERT와 비슷하게 Base, Large, Huge로 나누어 모델을 구성해보았다.

Model	Layers	Hidden size D	MLP size	Heads	Params
ViT-Base	12	768	3072	12	86M
ViT-Large	24	1024	4096	16	307M
ViT-Huge	32	1280	5120	16	632M

Table 1: Details of Vision Transformer model variants.

- Training & Fine-tuning

  ResNet은 Adam 사용, fine-tuning에는 SGD+momentum 사용 등등

4.2 Comparison to State of the art

	Ours-JFT (ViT-H/14)	Ours-JFT (ViT-L/16)	Ours-I21k (ViT-L/16)	BiT-L (ResNet152x4)	Noisy Student (EfficientNet-L2)
ImageNet	**88.55** ± 0.04	87.76 ± 0.03	85.30 ± 0.02	87.54 ± 0.02	88.4/88.5*
ImageNet ReaL	**90.72** ± 0.05	90.54 ± 0.03	88.62 ± 0.05	90.54	90.55
CIFAR-10	**99.50** ± 0.06	99.42 ± 0.03	99.15 ± 0.03	99.37 ± 0.06	–
CIFAR-100	**94.55** ± 0.04	93.90 ± 0.05	93.25 ± 0.05	93.51 ± 0.08	–
Oxford-IIIT Pets	**97.56** ± 0.03	97.32 ± 0.11	94.67 ± 0.15	96.62 ± 0.23	–
Oxford Flowers-102	99.68 ± 0.02	**99.74** ± 0.00	99.61 ± 0.02	99.63 ± 0.03	–
VTAB (19 tasks)	**77.63** ± 0.23	76.28 ± 0.46	72.72 ± 0.21	76.29 ± 1.70	–
TPUv3-core-days	2.5k	0.68k	0.23k	9.9k	12.3k

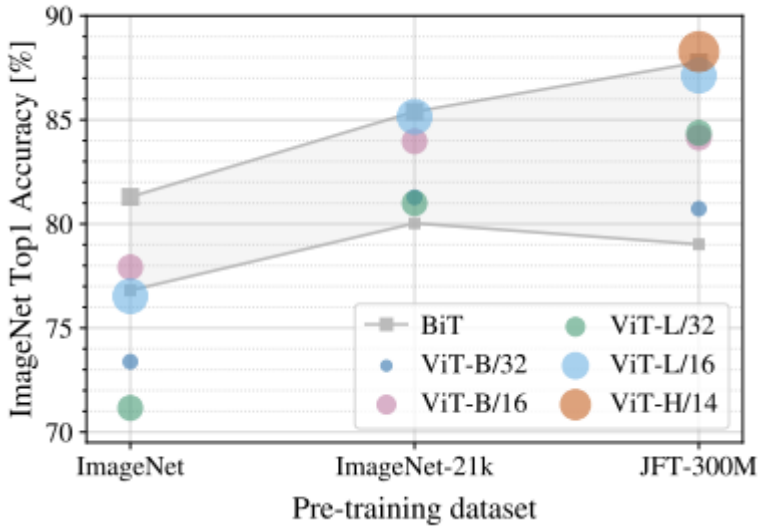
우선 본 논문에서 제시한 가장 큰 모델인 ViT-H/14와 ViT-L/16을 SOTA CNN과 비교한다.
첫 비교는 large ResNet으로 supervised transfer learning을 수행하는 Big Transfer(BiT)

다. 둘째는 Noisy Student로, Efficient Net으로 ImageNet과 JFT-300M으로 semi-supervised learning을 통해 얻은 큰 EfficientNet이다. 논문을 쓴 시점에서는 Noisy Student가 SOTA였다고 한다. 비교결과가 Table 2에 제시되어있다. 비교적 작은 모델인 JFT-300M으로 pre-train된 ViT-L/16은 BiT-L의 성능을 능가했고, 더 큰 모델인 ViT-H/14는 성능을 더 끌어올리고, 특히 더 까다로운 데이터셋에 대해 좋은 성능을 보였다. 이 모델들은 기존의 SOTA보다 적은 연산을 필요로 했다.

4.3 Pre-training Data Requirements

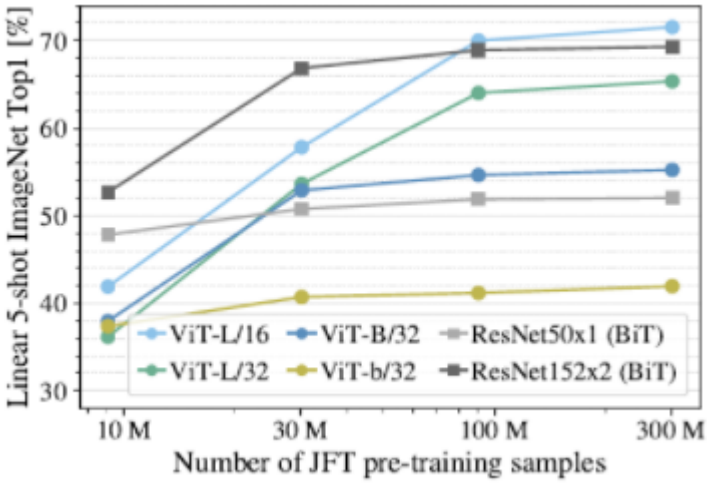
ViT에게 dataset size가 얼마나 큰 영향을 미치는지 확인하기 위한 실험 진행

ImageNet, ImageNet-21k, JFT-300M의 순으로 dataset의 size 증가



각 dataset에 대해 pre-train 후 ImageNet에 대해 fine-tune한 결과

dataset이 커질수록 ViT가 커질때 성능이 더 좋아짐



또, 전체 JFT-300M dataset에 대해서 subset으로 학습을 진행했을 때에도 dataset이 커질수록 큰 모델의 성능이 좋아짐