아주대학교
세상의 A+가 되겠습니다

2023-2
AJOU SOFTCON

A research on the ViT and CNN Models for Image Recognition

이 름 서동건

지도교수 유종빈

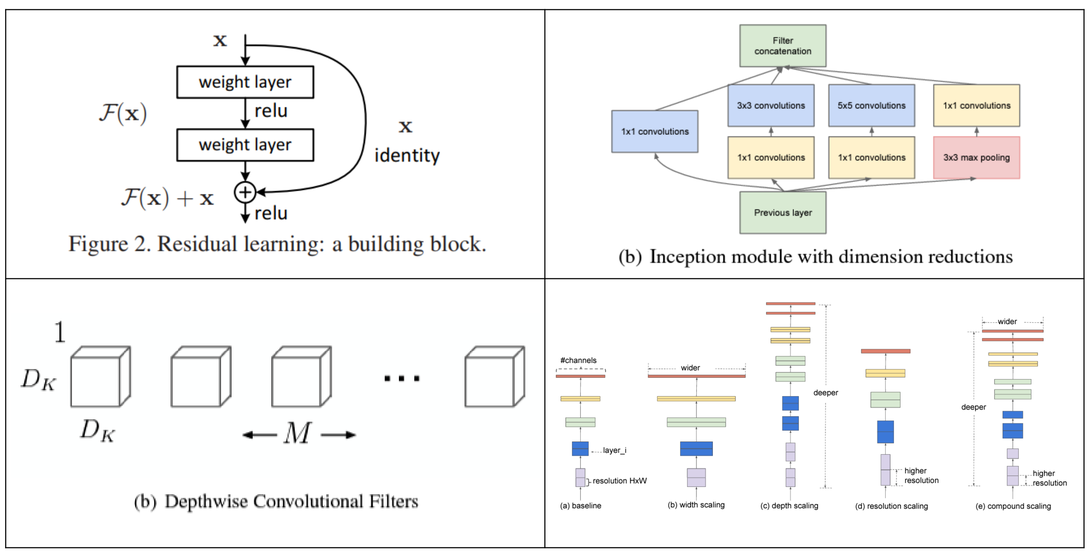
연구배경

- 딥러닝 기술을 활용한 컴퓨터비전 분야에 관심과 흥미가 생겨 깊고 전문적인 공부와 연구를 진행하고자 유종빈 교수님과 함께 하는 자기주도연구1을 진행하며 컴퓨터비전 분야에 더욱 흥미를 느끼게 되었다. 이에 Image Recognition에 대해 추가적인 연구를 진행하고자 자기주도연구2 까지 참여하게 되었다.
- 따라서 Image Recognition을 위한 Deep Learning Architecture 중에서 실제 연구의 최전선에서 많이 사용되고있는 대표적인 ViT와 CNN과 발전과정에 대해 심도 깊은 이해를 가지고 실제 구현하는 것이 최종 목표이다.

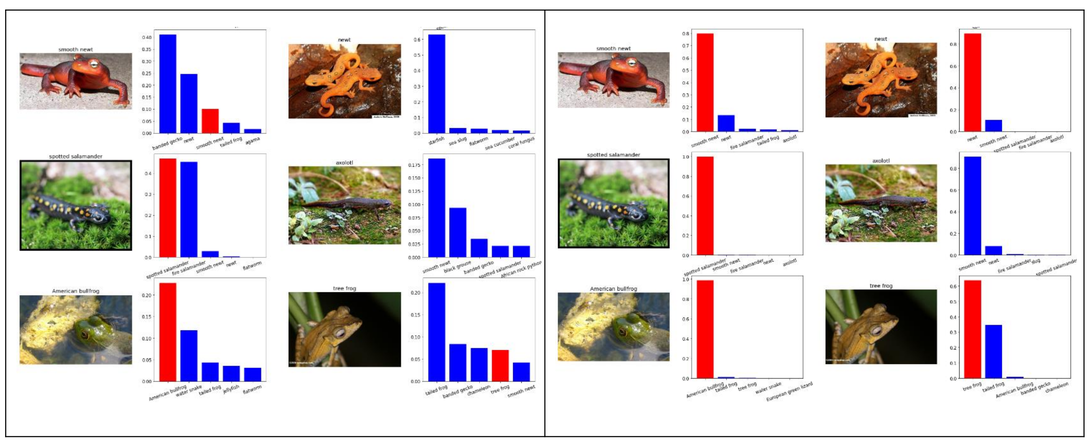
결과 및 분석

- CNN 과 ViT 를 자세하게 비교분석하기 위해 CNN의 발전과정부터 적용한 세부 모델들에 대한 Case Studie를 진행했다.

1. ResNet은 스킵 연결(skip connection)을 통해 잔차를 학습하도록 만들어진 인공신경망으로, Overfitting 문제와 gradient vanishing 문제가 해결되어 성능이 향상되었다.
2. GoogLeNet은 1x1 Convolution으로 Channel 수를 줄여 Parameter 수를 줄이는 Network in Network (NIN)를 활용한 Inception 이라는 모듈로 구성되어 있다는 특징이 있다.
3. MobileNet은 스마트폰 및 기타 모바일 장치와 같이 리소스가 제한된 환경에서 효율적인 계산을 하도록 Depth-wise Separable Convolution을 활용한 경량화 네트워크다.
4. EfficientNet은 depth, width, resolution 3가지가 일정한 관계가 있다는 것을 찾아내어, 모두 효율적으로 조절할 수 있도록 고려한 Compound scaling 방법으로 NAS(neural architecture search) 구조를 수정하여 SOTA를 달성한다.



- CNN 구조 이후 ViT가 나왔다고 해서 ViT만이 정답이 아니고, 데이터의 종류와 크기 등, 여러가지 경우에 따라 CNN모델이 더 좋을 때도, ViT모델이 더 좋을 때도, 혹은 두가지를 적절히 섞거나 융합하여 사용하는 것이 좋을 때도 있다는 사실을 확인했다.

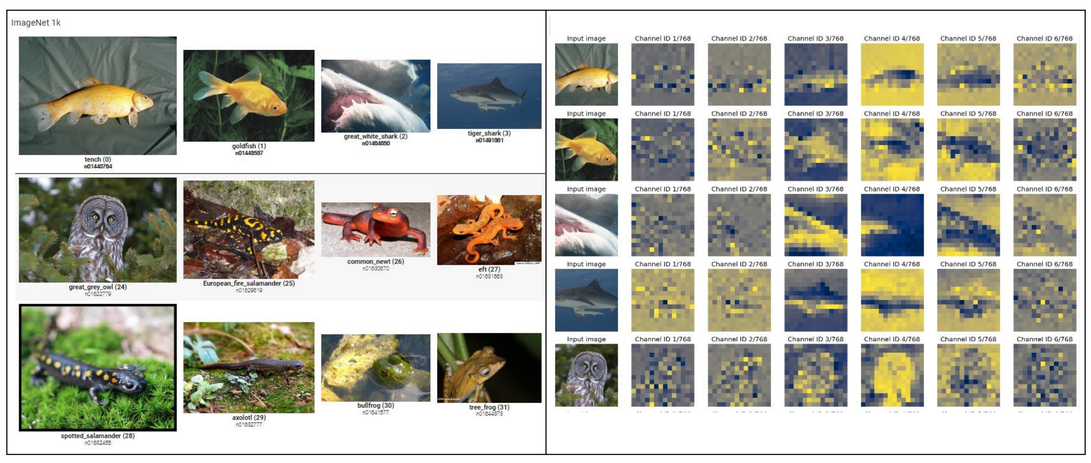


(1) ViT 모델 추론시 오답으로 측정했던 결과도
(2) Hybrid 모델 추론시 정확하게 측정하는 결과 도출

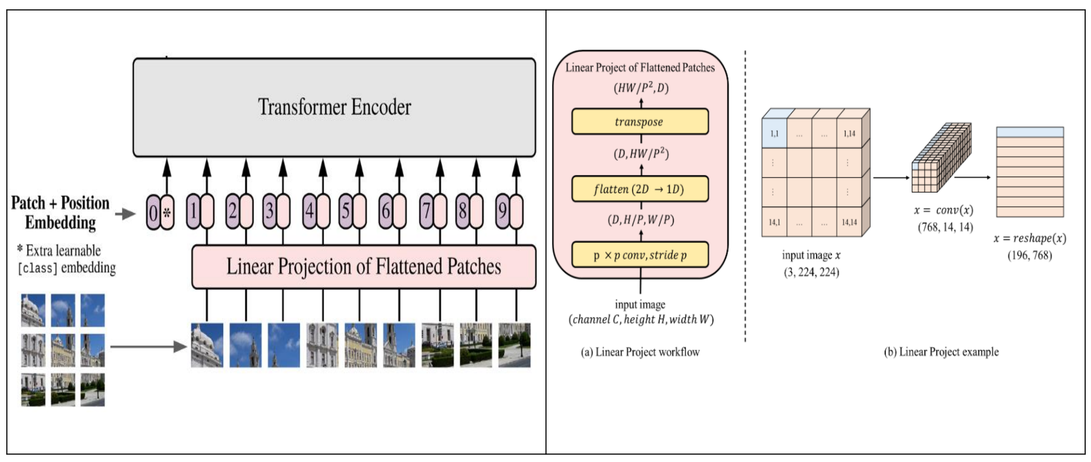
연구진행과정

- 최근 Image Recognition 분야에서 State of the art를 달성하고 있는 Vision transformer에 대해 진행한 연구를 자세하게 정리 해보자면 다음과 같다.

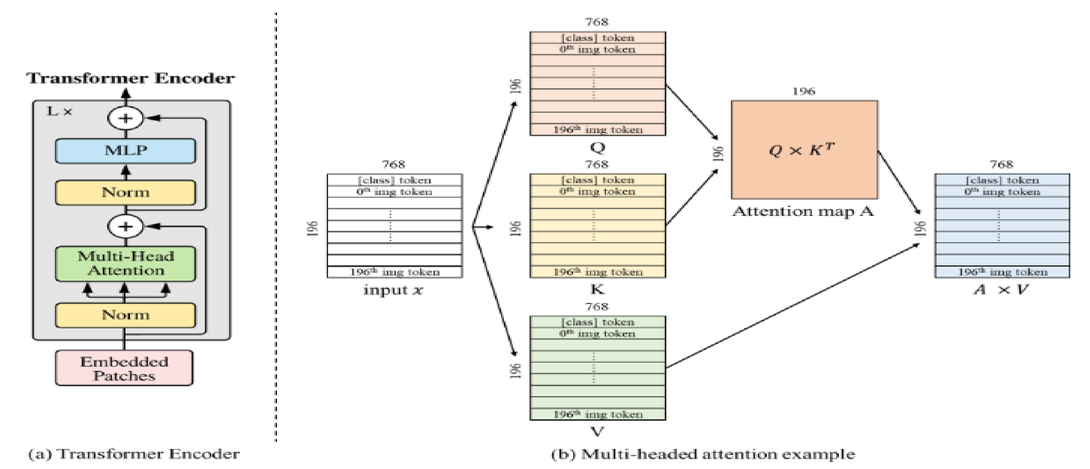
1. 데이터셋은 ImageNet 1k를 사용했다. 이후 이미지 데이터 작업을 위해 환경설정을 진행하고, 시각화를 위한 유틸리티 기능을 정의하며 PyTorch 및 'timm' 라이브러리를 사용하여 Vision Transformer 모델을 구현했다.



2. ViT 모델의 핵심 모듈이라고 볼 수 있는 Patch Embedding Layer를 구현했다. Patch Embedding Layer는 2D 인풋 이미지에서 p x p 크기의 이미지를 1D 토큰으로 만드는 레이어로, 2D convolution → Flatten → Transpose 함수를 통해 나타낼 수 있으며, 기능은 아래와 같이 표현할 수 있다.



3. Transformer Encoder에 들어가는 중요한 개념인 Multi-headed self attention 을 직관적이고 명료하게 정리해보면
[1] A = Q x K.t() : Query Q 와 Key K 를 이용해서 Attention map A 을 구하고 나서
[2] MHSA(x) A x V : Attention value A 와 Valeu V 를 이용해서 최종 아웃풋을 구하는 과정이라고 나타낼 수 있다.




```
B, N, C = x.shape
qkv = self.qkv(x).reshape(B, N, 3, self.num_heads, C // self.num_heads).permute(2, 0, 3, 1, 4)
q, k, v = qkv.unbind(0)

q = q * self.scale
```

```
def forward(self, x):
    x = self.fc1(x)
    x = self.act(x)
    x = self.fc2(x)
    return x
```

```
def forward(self, x):
    x = x + self.attn(self.norm1(x))
    x = x + self.mlp(self.norm2(x))
    return x
```

(1) q, k, v 생성 (2) Mlp_exp (3) Block_exp forward 함수