기존 CNN Recognition Model 의 한계를 극복한 ResNet

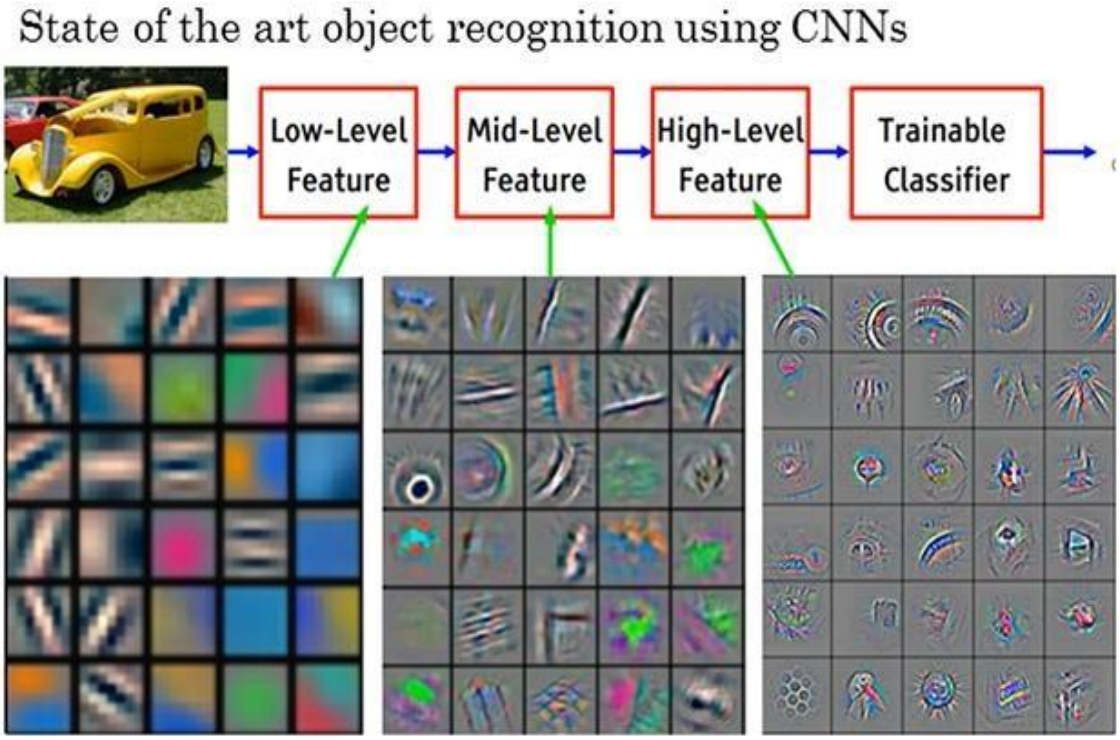
:"Deep Residual Learning For Image Recognition" 논문 분석을 중심으로

기존 Convolutional Neural Network 기반의 Image Recognition Model 에서는 Layer 의 개수를 늘릴수록 좋은 성능을 보여주었지만, Training Error 가 적은 Layer 에서 오히려 적은 비율을 보여주는 Degradation Problem 을 보여주었다. Microsoft 의 연구팀은 Residual Learning 이라는 개념을 도입, 발달하여 만들어낸 ResNet Model 이 기존 Model 들의 Degradation Problem 을 해결했을 뿐만 아니라 다방면이 성능이 개선되었음을 실험적으로 확인하여 논문 "Deep Residual Learning for Image Recognition"을 개재하였다.

)

기존 CNN Model 의 한계

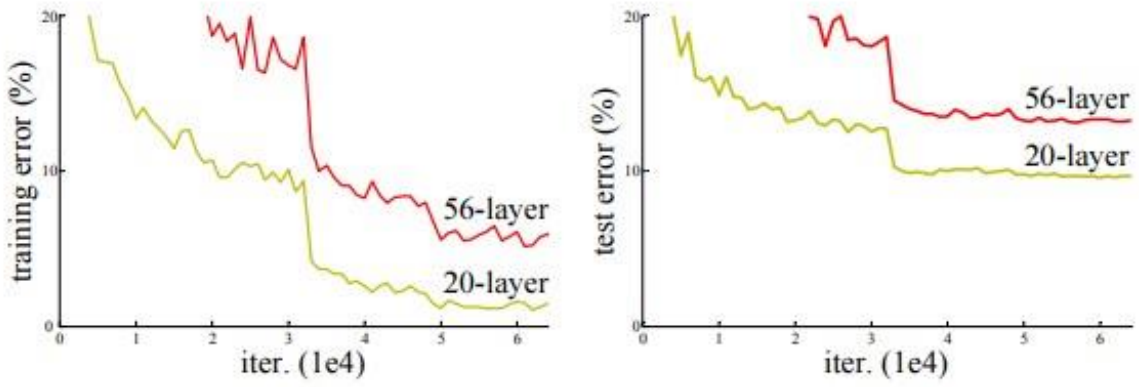
Deep Learning Model, 그 중 CNN(Convolutional Neural Network) Model 의 성능은 Hidden Layer(은닉층)의 깊이에 따라 더욱 수준이 높은 특징들을 판단할 수 있게 되며, 많은 연구들이 Model 의 성능에 Hidden Layer 의 Depth(깊이)가 매우 중요하다는 것을 입증한다.



[Figure 1] CNN 을 활용한 Object Recognition 의 예

하지만 이런 Hidden Layer 의 Depth 와 성능은 어느 정도 비례하여 같이 오르다가 어느 수준에 도달하면 Vanishing/Exploding Gradients 가 발생하여 Model 의 성능에 악영향을 주는 문제가 발생한다.

다행히 위의 문제는 Normalized Initialization, Intermediate Normalization Layers 등 다양한 방법론을 통해 해결되었지만, Model 이 더욱 깊은 Layer 를 가짐에 따라 정확도가 포화되어 Shallow-Layer 에 비해 오히려 성능이 떨어지는 Degradation Problem 이 발생하게 된다.

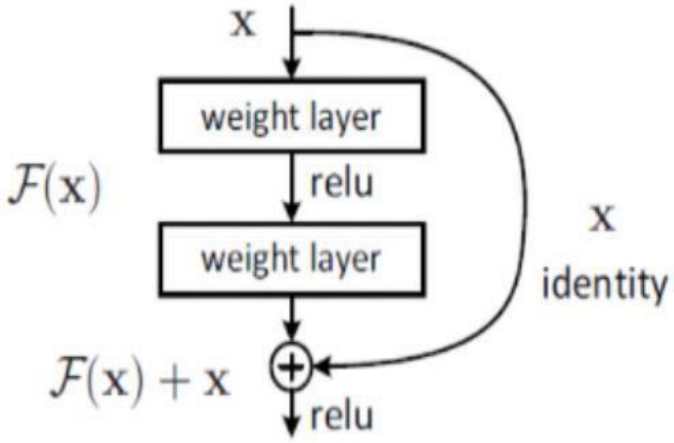


[Figure 2] Layer 개수에 따른 Training Error(좌), Test Error(우)

만약 Over-Fitting 이 원인이라면 Training Error 는 Deep-Layer Model 이 낮아야 하지만, Test Error 는 물론이고 Training Error 도 Deep-Layer Model 이 Shallow-Layer Model 보다 낮음을 관찰할 수 있으므로, Over-Fitting 이 원인은 아니다.

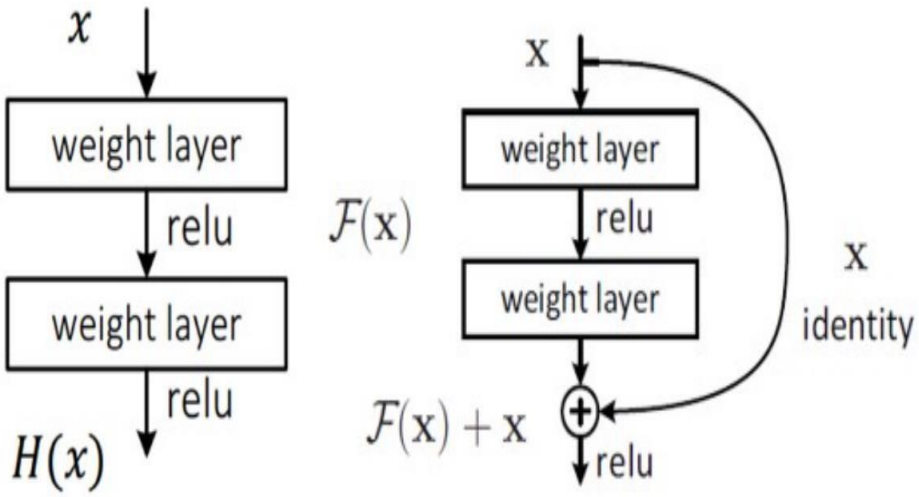
Residual Network

그렇기에 기존 Model 은 깊은 Layer 를 활용하여 학습하기가 까다롭다는 문제가 있다. Layer 층의 개수가 성능을 증가하는 요인이지만, 개수가 증가함에 따라 발생하는 문제 또한 존재한다는 것이다. Microsoft 의 연구진은 개수를 유지하면서 Degradation Problem 을 해결하기 위해 ResNet 을 도입하였다. ResNet 에 쓰이는 학습 방식은 기존과 달리 Residual Learning 을 도입한 방식이다.



[Figure 3] Residual Block

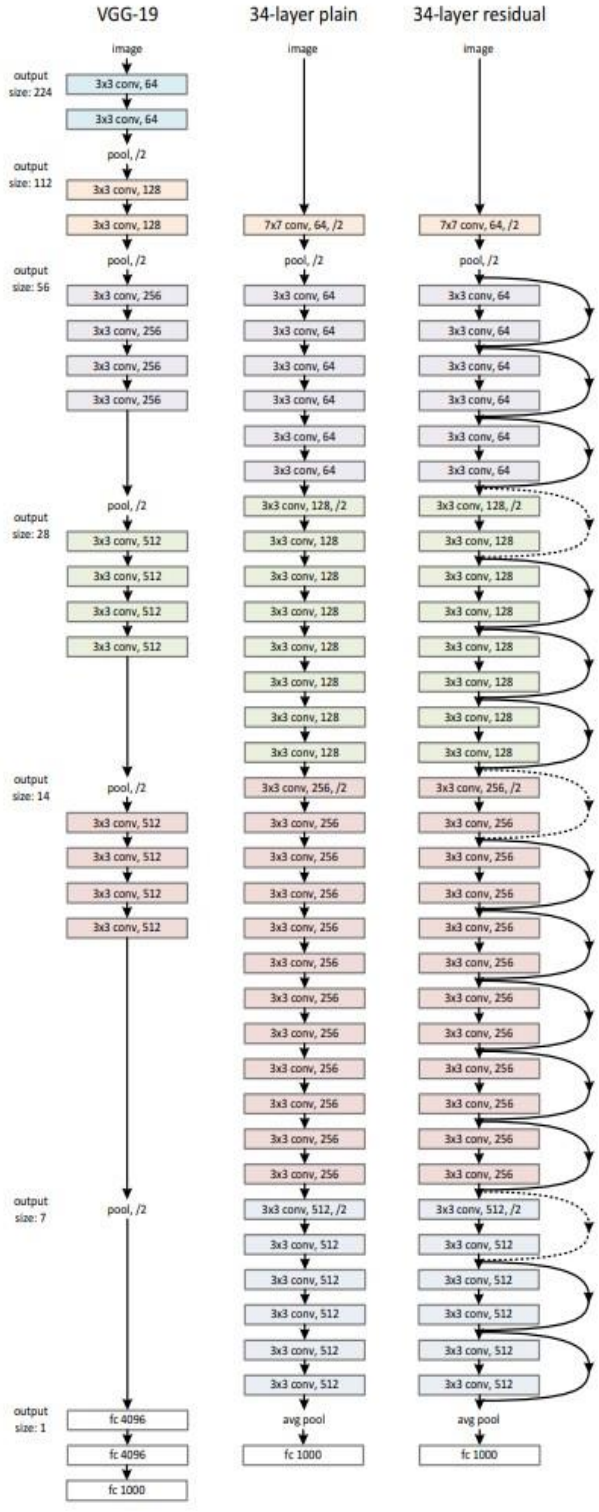
기존 Model 은 Input Value : x 를 Target Value : y 로 mapping 하는 함수 H(x)를 만든 것을 목적으로 한다. 이때, Residual(잔차)은 F(X)=H(x)-x 이다. ResNet 에서 사용되는 Residual Learning 은 F(x)를 0 에 가깝게 하는 것을 목표로 한다. 즉, H(x)를 입력값 x 로 mapping 하는 것이 Training 의 목표이다.



[Figure 4] 기존에 사용되던 Block(좌)와 ResNet 에서 사용하는 Residual Block(우)

기존 Model 과 달리 ResNet 은 Input Layer 에서 Output Layer 에 직접적으로 연결되는 Shortcut 을 추가한다. Input Layer 의 x 가 직접적으로 Output Layer 로 가기에, 기존 Parameter 과 Layer 층의 개수는 변함이 없고, x 를 더해주는 것 외에는 연산량에 큰 차이가 없다. 기존 Model 은 Output Value 가 수많은 Matrix 의 Multiplication 으로 나오는 반면 Resnet 은 Plus 로 나타나기에 Layer 층의 수를 바꾸지 않더라도 Gradient Vanishing 문제를 해결할 수 있다.

또한 기존 Model 은 convolution 연산을 누적하여 곱한다면 ResNet Model 은 Block 단위로 전달한다. Shortcut 경로를 사용한다면 최종적으로 덧셈형식이 나오는데 덧셈 연산은 입력과 출력의 값을 직접 더하기 때문에 기울기가 소실되지 않고 보존된다. 또한 shortcut 을 통해 짧은 경로를 통해 전달되므로, gradient vanishing 을 완화한다.



VGG-19(좌), Plain-34(중), ResNet-34(우)

주요 경로와 Shortcut 경로를 통해 입력과 출력 사이의 Residual 을 학습하기에, 보다 쉽게 Residual 을 학습하고 Layer 수가 많더라도 효율적으로 학습할 수 있다. 따라서

Shortcut 을 사용하여 덧셈 형식으로 Residual Block 을 구성하면, Input 과 Output 사이의 Gradient Vanishing 문제를 완화할 수 있다.

[Figure 5] 다양한 Model 의 구조

그렇다면 Shortcut 경로를 사용하여 어떻게 덧셈 형식으로 나타나는지 알아보자. 우선

Shortcut 을 사용하기 위해서는 Input Matrix 와 Output Matrix Dimension 가 같아야 한다. 이미 같다면 Matrix 의 덧셈을 하는데 문제가 없지만, 다르다면 Dimension 을 일치시키기 위해 1x1 Convolutions Layer 를 사용해 입력을 적절히 변환시킨다. 이렇게 하여 Input 과 Output 의 Dimension 이 일치하므로, Shortcut 의 사용이 가능해진다.

l을 Layer number 라 하고, x를 l-layer 의 Input, y_l을 l-Layer 의 Output 이라 하자. 함수 h 를 Input 과 Output 사이에 항등 Mapping 을 유지하기 위한 함수, $\mathcal{F}$ 를 Residual Function(잔차함수), f를 Activation Function(ReLU)라 하자.

그렇다면 다음 식이 성립한다.

$$y_l = h(x_l) + \mathcal{F}(x_l, \mathcal{W}_l), \quad (1)$$

$$x_{l+1} = f(y_l) \quad (2)$$

이 식은

$$x_{l+1} = x_l + \mathcal{F}(x_l, \mathcal{W}_l) \quad (3)$$

로 나타낼 수 있으며, 일반화하여

$$x_L = x_l + \sum_{i=l}^{L-1} \mathcal{F}(x_i, \mathcal{W}_i) \quad (4)$$

라 나타낼 수 있다. 즉 덧셈의 연산으로 표현이 가능하다. Chain Rule(연쇄법칙)을 적용하면,

$$\frac{\partial \varepsilon}{\partial x_l} = \frac{\partial \varepsilon}{\partial x_L}\frac{\partial x_L}{\partial x_l} = \frac{\partial \varepsilon}{\partial x_L}\left(1 + \frac{\partial}{\partial x_l}\sum_{i=l}^{L-1}\mathcal{F}(x_i, \mathcal{W}_i)\right) \quad (5)$$

[Figure 6] 덧셈 연산으로의 유도 과정. 식 (1), (2), (3), (4), (5)

으로 유도할 수 있다. ε 를 Loss Function(손실 함수)이라고 두었을 때, 이 값이 0 이 되어버린다면 Vanishing Problem 이 발생한다. 그러나 $\frac{\partial \varepsilon}{\partial x_L}$은 마지막 Layer 의 미분값으로,

이전 가중치 행렬들을 만나지 않기에 Vanishing Problem 이 발생하지 않는다. 종합적으로, 기존 Model 은 곱셈의 형태로 나타나기에 Vanishing Problem 이 발생하지만, ResNet 은 덧셈 형태이기에 Vanishing Problem 이 발생하지 않으므로, 해결책이 될 수 있다.

ResNet 성능평가

본 논문에서는 Residual Network 를 테스트 및 비교하기 위해 VGG-19 Model 을 참고하여, Residual Learning 이 적용되지 않은 Plain Model, 그리고 Residual Learning 이 적용된 ResNet 을 다양한 Dateset 에 적용하고 비교하였다. 이하에는 해당 Network 들에 대한 설명, 그리고 2 개의 Network 의 성능 비교를 서술한다.

Plain Network

상기한 대로, 본 논문에서 사용된 Network 는 VGG Network 에서 영감을 받았다. 이에 따라 Plain Network 의 Convolutional Layer 들이 가지는 Convolution Filter 의 크기를 3 x 3 으로 정하였으며, 2 가지 규칙에 기반하여 설계하였다.

1. Output Feature Map 의 크기가 같은 Layer 들은 모두 동일한 개수의 Convolution Filter 를 사용한다.
2. Output Feature Map 의 크기가 반으로 줄어든다면, Time Complexity 를 동일하게 유지해주기 위해 Convolution Filter 의 개수를 2 배로 늘린다.

또한, Stride 값이 2 인 Convolutional Layer 들을 활용하여 Downsampling 을 수행하였으며, Model 의 마지막에는 Global Average Pooling Layer 과, 1000 의 크기를 가진 Fully-Connected Layer 와 Softmax 를 사용했다. 이에 따라 해당 Network 의 전체 Layer 개수는 34 이다.

Residual Network(ResNet)

ResNet 은 위의 Plain Network Model 에 기반하여 구성되었다. Plain Network 와의 차이점은 Shortcut Connections 를 삽입했다는 점이다.

Shortcut Connection 을 활용하였을 때, Input 과 Output 의 차원에 따라 2 가지 경우가 발생할 수 있다.

1. Input Dimension = Output Dimension

이 경우에는 Identity Shortcut 을 그대로 사용해주면 된다.

2. Input Dimension ≠ Output Dimension
   이 경우에도 2 가지 선택권이 주어진다

   1. Zero-Padding 을 적용하여 차원을 맞춰준다 이 경우,

      추가적인 Parameter 를 사용하지 않는다.

   2. Projection Shortcut 을 사용한다 차원을 맞춰주는

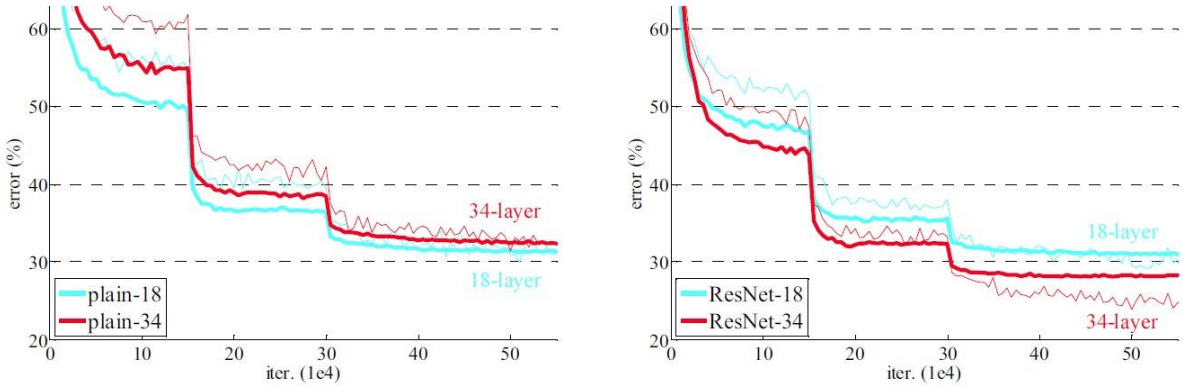
      과정에서, 1x1 Convolution 이 적용된다.

2 가지 선택 모두 Shortcut 들이 Feature Map 을 2 Size 씩 건너뛰므로, Stride 를 2 로 설정했다.

Figure 5 는 왼쪽부터 VGG-19 Model, Plain Network Model, ResNet Model 을 도식화한 것이다. ResNet 에서 곡선 화살표들 중 실선은 Shortcut Connection 에서 Input 과 Output 의 차원이 동일한 경우, 점선은 차원이 동일하지 않아 차원을 맞춰줘야 하는 경우를 나타낸 것이다.

ImageNet 으로의 Implementation

본 논문에서는 Plain Network 와 Residual Network 를 각각 18 Layer 와 34 Layer 의 2 가지로 나누어 각각의 Training Error 와 Validation Error 가 일어난 비율을 그래프로 나타내었다. 이때, Residual Model(ResNet)에서 모든 Shortcut 에는 Identity Mapping 이 적용되었고, 차원 증가에는 Zero-Padding 이 사용되어 Parameter 의 개수가 증가하지 않았다.



[Figure 7] Training Error 및 Validation Error 그래프.

Plain-18 과 Plain-34 의 비교(좌), ResNet-18 과 ResNet-34 의 비교(우)

Figure 7 의 얇은 곡선 그래프는 Training Error, 굵은 곡선 그래프는 Validation Error 을 뜻한다. 또한 파란색 곡선은 18 Layer 를 가진 Network, 빨간색 곡선이 34 Layer 를 가진 Network 를 말한다. Plain Network 는 좌측, ResNet 은 우측에 표시되어 있다. 편의를 위해 이후 각 Model 은 (Model 종류)-(Layer 개수) (예: Plain-18)의 방식으로 표기하도록 하겠다.

	plain	ResNet
18 layers	27.94	27.88
34 layers	28.54	**25.03**

[Figure 8] Plain Model 과 ResNet 의 Layer 개수에 따른 Top-1 Error 표

Figure 8 은 각 Model 의 Layer 개수에 따른 Top-1 Error(Model 이 예측한 확률이 가장 높은 Class 와 실제 정답 Class 가 일치하지 않는 비율)를 보여준다.

Figure 7 에서 우리는 Plain-34 에서 Plain-18 보다 높은 Validation Error 가 나타났음을 알 수 있다. 하지만, Training Error 또한 높았음을 감안하여 연구진들은 해당 문제가 Degradation Problem 에 의한 문제이고, Vanishing Gradient 때문에 발생하는 최적화 문제가 아닌 것으로 판단했다.

ResNet 에서는 Plain Model 과 전혀 다른 양상이 관찰되었다. ResNet-34 의 성능이 ResNet-18 보다 더 낮은 Training Error 를 보여주었고, 이에 따라 Validation Error 또한 상당히

낮아진 것을 보여주었다. 전반적으로 2.8%의 우수한 성능 향상을 보여주었으며, 이를 통해 Residual Learning 이 적용되면 Degradation Problem 이 해결되고, Depth 가 증가하더라도 좋은 정확도를 얻을 수 있었음을 알 수 있었다.

또한, 표에서 ResNet-18 에 비해 ResNet-34 의 Top-1 Error 가 3.5% 감소하였음을 확인할 수 있다. 이는 즉 Residual Learning 이 Extremely Deep System 에 효과적임을 나타낸다. 표에서 ResNet-18 과 Plain Model 을 비교했을 때, 성능이 거의 유사했지만, ResNet-18 이 Plain-18 과 Plain-34 보다도 더 낮은 Top-1 Error 수치를 보여주었다. 즉, Model 이 과도하게 깊지 않아도 되는 경우, ResNet 이 같은 상황의 Plain Model 보다도 더 좋은 결과를 보여줌을 알 수 있다 (참고로 Plain Model 또한 충분히 좋은 성능을 보여주는 편이다).

기타 Dataset Implementation 과 논문의 가치

연구진들은 여기서 멈추지 않고 ResNet 의 성능을 더욱 향상시켜 ResNet-50, ResNet-101, ResNet-152, Resnet-1202 등의 Model 을 만들어 CIFAR-10, PASCAL VOC, MS COCO, ImageNet Detection, ImageNet Localization 에 적용했다. 그 결과, Test 를 진행해본 모든 Dataset 에서 ResNet 은 기존 Model 들보다 향상된 능률을 보여주었다.

Residual Learning 은 기존 Model 들의 한계를 극복할 수 있는 방법을 제시하고, 보다 좋은 성능의 Model 을 만들어낼 수 있는 학습 방법을 제시하였다. 연구팀은 Residual Learning 과 ResNet 에 대한 연구를 바탕으로 ILSVRC 2015 에서 우승을 차지하였고, 연구 내용은 아직도 Deep Learning 이미지 분야에서 많이 활용되고 있다.

참고 문헌

"Deep Residual Learning for Image Recognition", Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun, (https://arxiv.org/abs/1512.03385) 내용 참조, [Figure 2, 3, 4, 5, 7, 8]

"Lab-11 CNN Basics Convolution", leban, (https://velog.io/@lybin10/lec-12-NN 의-꽃-RNN-이야기) 내용 참조, [Figure 1]

"(ResNet)Deep Residual Learning for Image Recognition 논문 리뷰", Tobigs,

(https://tobigs.gitbook.io/tobigs/deep-learning/computer-vision/resnet-deep-residuallearning-for-image-recognition) 내용 참조

"ResNet: Deep Residual Learning for Image Recognition", Curaai00, (https://curaai00.tistory.com/1) 내용 참조

"CNN 모델 탐구 (6-2) Identity Mapping in ResNet", Paris Lee, (https://m.blog.naver.com/siniphia/221387516366) 내용 참조, [Figure 6]

"이미지 분류 평가 지표: Top-1 Error 와 Top-5 Error 이해", 이영빈, (https://modulabs.co.kr/blog/top-1-error-top5-error/) 내용 참조

"ResNet (Deep Residual Learning for Image Recognition) 논문 리뷰", philBaek(백광록), (https://phil-baek.tistory.com/entry/ResNet-Deep-Residual-Learning-for-Image-Recognition-%EB%85%BC%EB%AC%B8-%EB%A6%AC%EB%B7%B0) 내용 참조

"ResNet 구조 이해 및 구현", 우준세, (https://wjunsea.tistory.com/99) 내용 참조