[논문리뷰] Deep Residual Learning for Image Recognition

ResNet 이 처음 제시된 논문

Abstract

깊은 네트워크일수록 train은 어렵다. 본 논문에서는 residual(잔차) learing framework 를 제시해 실질적으로 이전보다 더 깊은 네트워크에 대한 학습을 쉽게 한다.

논문에서는 layer 의 input 값을 참조하는 residual function을 명시적으로 재구성한다. ⇒ 뭔말?

논문에서는 선험적인 증거를 제시해 residual network 가 더 최적화하기 쉽고, 상당히 더 깊은 네트워크에서 더 높은 성능을 가짐을 이해시킨다.

ImageNet dataset 에서 논문은 VGGNet 보다 8배 더 깊은 152layers 를 가진 residual nets를 평가한다. 이러한 residual nets 의 앙상블은 ImageNet testset 에서 3.57% 의 error rate 를 내놓는다.

⇒ ILSVRC 2015 classification task 에서 우승

또한 CIFAR-10 을 100개, 1000개의 layer로 분석한다.

Depth of representation 은 visual recognition task 에서 가장 중요하다. 오직 논문의 극단적으로 깊은 representation 만으로, 논문에서는 COCO object detection dataset에서 상대적으로 28%의 상승을 얻었다. Deep residual nets 은 ILSVRC 와 COCO 2015 competition에 제출한 foundation 이고, ImageNet detection, ImageNet localization, COCO detection, COCO segmentation 에서 우승했다.

1. Introduction

깊은 convolution neural network는 image classification의 돌파구이다.

깊은 네트워크는 자연적으로

⇒ e2e 다중 layer 에는 low/mid/high level 의 특징들이 존재하고 classifier 와 합쳐진다.

특징의 level들은 쌓인 레이어들의 수(depth) 에 의해 풍부해진다.

최근 연구에서는 네트워크의 깊이가 imageNet challenge 에서 매우 중요함이 밝혀졌고, 실제로 imageNet을 이끄는 결과들은 모두 16~30의 깊은 네트워크의 모델에서 나왔다.

다른 중요한 visual recognition task 또한 더 깊은 네트워크에서 좋은 성능을 보였다.

깊이의 중요성을 생각하면 "**더 많은 layer를 쌓을 수록 학습이 쉬워지는가?**" 라는 질문이 떠오를것이다.

이 질문에 답을 하는데 취약한 점은 **vanishig/exploding gradients** 문제이다.

이러한 문제들은 normalized initialization 이나 intermediate normalization layers (10개의 layer를 역SGD 역전파때 수렴이 시작하도록 함) 으로 주로 다뤄졌다.

네트워크가 깊어질수록 정확도가 saturated 되고 빠르게 성능이 저하되는 것은 당연하다. 이러한 degration은 overfitting 에 의한 것이 아닌데, deep model 에 더 많은 layer들을 추가하는 것은 더 높은 training error 을 초래한다.

(Layer 가 깊어진다고 training accuracy 가 높아지는 것 아님.)

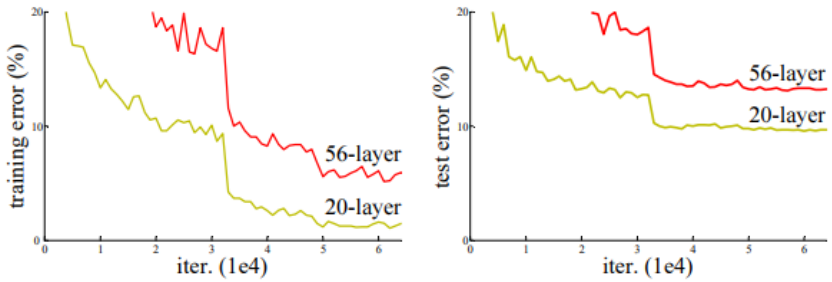

Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer "plain" networks. The deeper network has higher training error, and thus test error. Similar phenomena on ImageNet is presented in Fig. 4.

Training 에서의 degradation 은 모든 system을 optimization 하는 것은 쉽지 않다는 것을 보여줍니다.

더 얕은 architecture와 그에 대응하는 더 깊은 architecture 를 고려해봅시다.

더 깊은 network에 대해 identity mapping layer를 추가하고 다른 layer들은 얕은 모델에서 학습된 것을 카피해옵니다.

이런 방법은 더 깊은 모델이 얕은 모델보다 더 높은 training error을 가지지 않도록 해줍니다.

하지만 실험을 통해 이 방법은 이미 구축된 해결책보다는 훨씬 더 좋은 성능을 내지는 못합니다. (또는 실현 가능한 시간에 성능을 낼 수 없음)

이 논문에서는 deep redisual learning framework 을 제시해서 degradation problem 을 다룹니다. 쌓여진 레이어가 그 다음 레이어에 바로 적합되는 것이 아닌 residual 의 mapping에 적합하도록 만들었습니다.

Desired underlying mapping을 H(x) 라 정의한다면, 이 논문에서는 비선형적인 layer 를 F(x) = H(x) - x 에 mapping시킨다. 원본 mapping은 F(x) + x 라고 다시 표현할 수 있다. 우리는 이 residual mapping이 원본보다 최적화가 더 쉽다고 가정을 했다. 극단적으로 만약 identity mapping 이 최적회라 하면, residual 을 0 으로 맞추는게 identity mapping을 nonlinear layer에 맞추는 것보다 쉽습니다.

F(x) + x 는 shortcut connection 과 동일한데 이는 하나 또는 그 이상의 layer를 skip하게 만들어줍니다. 여기서 identity mapping으로 shortcut 이 되면서 skip을 만듭니다.

이 identity short connection 은 추가적인 파라미터가 필요하지 않고 복잡한 곱셈 연산도 필요하지 않는 것이 장점입니다. 전체 네트워크는 역전파에서 SGD 로 e2e 로 학습될 수 있습니다.

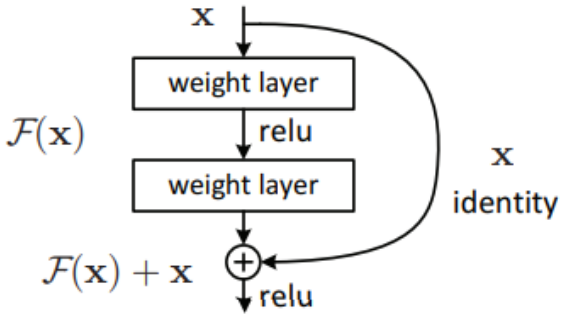

Figure 2. Residual learning: a building block.

이 논문의 목표

1. Our extremely deep residual nets are easy to optimize, but the counterpart “plain” nets exhibit higher training error when the depth increases
2. Our deep residual netw can easily enjoy accuracy gains from greatly increased depth, producing results substantially better than previous networks

2. Related Work

Residual Representation

Vector quantization에 있어 residual vector로 encoding하는 것이 original vector보다 훨씬 효과적이다.

Shortcut Connections

ResNet의 Shortcut connection 은 다른 방식들과 달리, parameter가 전혀 추가되지 않으며, 0으로 수렴하지 않기에 절대 닫힐 일 없이 항상 모든 정보가 통과된다.

따라서 지속적으로 residual function 을 학습하는 것이 가능하다.

3. Deep Residual Learning

3.1 Residuall Learning

H(x) 를 기본 매핑으로 간주하면 x 가 input일 때 다수의 비선형 layer 가 복잡한 함수를 점근적으로 근사할 수 있다고 가정하면, Residual function 인 H(x) - x를 무의식적으로 근사할 수 있다.

이 식은 수학적으로 이항만 하고 있는 것이기 때문에 동일한 하나의 식이지만 형태에 따라서 학습의 용이함이 달라진다.

⇒ F(x) = H(x) - x 가 학습에 더 용이하다.

3.2 Identity Mapping by Shortcuts

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + \mathbf{x}.$$

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + W_s\mathbf{x}.$$

Shortcut connection 은 파라미터나 연산 복잡성을 추가하지 않는다.

이때 F+x 연산을 위해 x와 F의 차원을 같게 만들기 위해 linear projection 인 W_s 를 곱해서 차원을 갖게 만든다.

(W_s 는 차원을 매칭시킬때만 사용)

3.3 Network Architectures

Plain Network

VGGNet 을 참고하여 만들었다. 모두 동일한 feature map size를 갖게 하기 위해 layer들은 동일한 수의 filter를 가진다. Feature map size 가 절반이 되면 filter 수는 2배가 되도록 만든다.

Conv layer는 3x3 filter, stride = 2 로 downsampling, global average pooling layer를 사용하고, 마지막에는 softmax 로 1000-way fully connected layer를 통과시킨다.

최종 layer 수는 34개, VGG net 보다 적은 filter와 복잡도, VGGnet 의 18%의 연산

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
		3×3 max pool, stride 2				
conv2_x	56×56	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Table 1. Architectures for ImageNet. Building blocks are shown in brackets (see also Fig. 5), with the numbers of blocks stacked. Downsampling is performed by conv3_1, conv4_1, and conv5_1 with a stride of 2.

Residual Network

Plain net 에 shortcut connection 도입.

Identity shortcut 은 input과 output 을 같은 차원으로 맞춰줘야 한다.

1. The shortcut still performs identity mapping, with extra zero entries padded for increasing dimensions. This option introduces no extra parameter.
2. The projection shortcut in Eqn2 is used to match dimentions

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + W_s\mathbf{x}.$$

3.4 Implementation

- Image resized 224 * 224
- Batch normalization
- Initialize Weights
- SGD, mini batch 256
- Learning rate = 0.1
- weight decay 0.0001, momentum 0.9
- No dropout

4. Experiments

4.1 ImageNet Classification

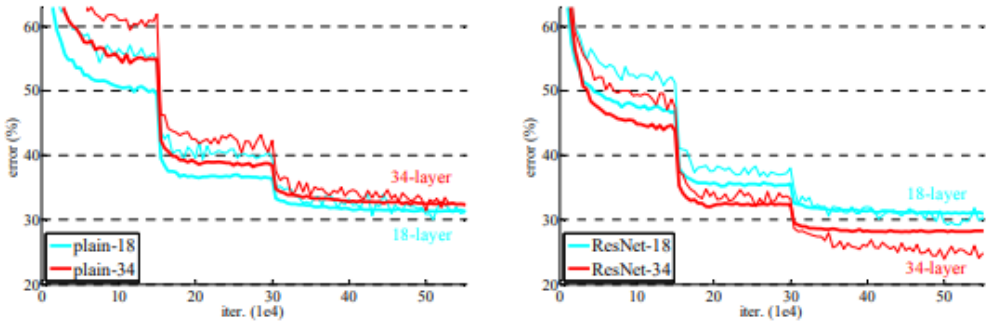

Figure 4. Training on **ImageNet**. Thin curves denote training error, and bold curves denote validation error of the center crops. Left: plain networks of 18 and 34 layers. Right: ResNets of 18 and 34 layers. In this plot, the residual networks have no extra parameter compared to their plain counterparts.

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

Table 2. Top-1 error (%, 10-crop testing) on ImageNet validation. Here the ResNets have no extra parameter compared to their plain counterparts. Fig. 4 shows the training procedures.

Plain Network

Plain net 에서 18-layer와 34-layer를 비교하면 34-layer 가 train error, test error 모두 높은 degradation problem을 볼 수 있다.

Plain net 이 BN 으로 학습되었기 때문에 non-zero varience 를 가지고 optimization 이 어려운 이유가 vanishing gradient 때문이 아닌 것을 알 수 있습니다.

Residual Network

- 18-layer ResNet봐다 34-layer ResNet 이 성능이 좋다. 더 낮은 training error를 가진다. Layer 깊이 대비 degradation problem을 잘 조절했다.
- Plain 과 비교했을 때 성공적으로 training error 를 줄였다.
- 18-layer 와 비교했을 때 accuracy 는 비슷하지만 18-layer ResNet 이 더 수렴이 빨랐다.
- ResNet 의 SGD 을 이용한 optimization 이 더 쉽다.