1주차 - Deep Residual Learning for Image Recognition

He_Deep_Residual_Learning_CVPR_2016_paper.pdf

Abstract

- We present a residual learning framework

  기존에 쓰이던 것보다 deeper network 를 쉽게 학습하기 위해 사용

  - ImageNet dataset

    152 layer 이상 - VGG net [40] 보다 8배 이상 깊은
    하지만 더 낮은 complexity

- We alse present analysis on CIFAR-10 with 100 and 1000 layers - 이 얘기 왜나온거지
- depth of representation 은 많은 visual recognition tasks 의 중요한 문제임

  우리의 extremely deep representation 으로 COCO object detection dataset 에서 28% improvment 를 얻었음

- deep residual net 을 기반으로 한 제출에서 ILSVRC & COCO 2015 competitions 1등함

1. Introduction

Deep convolution neural network - 이미지 분류 문제에서 series of breakthroughs 를 이룸. 최근의 연구(evidence) 에서 쌓여진 레이어의 수, 깊이가 중요함이 밝혀짐.

> **Is learning better networks as easy as stacking more layers?**
> **좋은 성능의 네트워크를 만들기 위해 단순히 층을 더 쌓는 것 만으로 가능할까?**

- 기울기 소실/폭팔 문제

- 성능 저하 문제(degradation problem) - not by overfitting

  적절히 깊은 모델에 더 많은 층을 추가하면 더 높은 training error 로 이어짐.
  → indicates that not all systems are similarly easy to optimize

  ✅ **이론상 깊은 네트워크가 얕은 네트워크보다 성능이 나쁠 이유가없음.**

  - Shallower architecture 와 더 많은 레이어를 추가한 deeper architecture
  - “there exists a solution by construction to the deeper model”
  - 깊은 모델의 added 레이어가 학습된 얕은 모델에 identity mapping 되었다고 할 때, 깊은 네트워크는 최소한 얕은 네트워크와 동일한 출력을 낼 수 있어야 한다.

  하지만 실제 학습에서는 깊은 네트워크가 동일하거나 더 나은 해를 찾지 못하고 오히려 training error 가 더 커지는 현상이 나타남.

Residual Learning 프레임워크 도입

instead of hoping each few stacked layers directly fit a desired underlying mapping, we explicitly let these layer fit a residual mapping.

> F(x) := H(x) - x
> *H(x) : desired underlying mapping

- plain net vs ResNet
  - **Plain Net**: 층들이 모여서 직접 $\mathcal{H}(x)$를 근사하려고 함.

    > $F(x) \approx H(x)$

  - **ResNet**: 층들이 “$\mathcal{H}(x)$ 자체”가 아니라 “$\mathcal{H}(x) - x$” (즉, **잔차**)를 근사하도록 설계.

    > $F(x) \approx H(x) - x$

- feedforward neural networks with “shortcut connections”

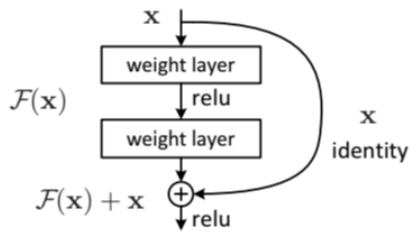


Figure 2. Residual learning: a building block.

- shortcut connection : 하나 혹은 그 이상의 레이어를 건너뛰는 connection
- 이 연구에서 shortcut connection 은 identity mapping 만 수행 (추가적인 매개변수나 계산 복잡도를 더하지 않음)

실험적 결과

> ImageNet 데이터셋

- plain network 가 층을 쌓을수록 높은 training error 를 보인 것에 반해 residual network 는 최적화하기 쉬웠음
- 깊이가 크게 증가했을 때도 쉽게 정확도 향상을 얻을 수 있었음.
- 152층 깊이의 잔차 네트워크조차 VGG[40] 보다 복잡성이 낮았음

2. Related Work

Residual Representations

- 이미지 인식 - VLAD / Fisher Vector / 백터 양자화 (vector quantization)
  - VLAD 는 잔차 백터(residual vectors) 로 인코딩하는 표현 방식
  - Fisher Vector 는 VLAD 의 확률적 버전
  - 백터 양자화는 원래 벡터를 인코딩하는 것 보다 잔차 벡터를 인코딩하는 것이 더 효과적임
- Low-level vision 과 컴퓨터 그래픽스
  - PDEs(편미분방정식) 을 풀기 위해 사용되는 Multigrade 방식 - 시스템을 multi 스케일에서의 하위 문제들로 다시 공식화, 각 하위 문제는 coarser 스케일과 finer 스케일 사이의 residual solution 을 담당.
    - Multigrade 의 대안으로 사용되는 hierarchical basis proconditioning - 두 스케일 사이의 잔차 벡터를 나타내는 변수들에 의존

⇒ 이러한 solver 들은 잔차적 특성을 사용하지 않는 다른 방식들보다 훨씬 빠르게 수렴함

Shortcut Connections

- 다층 퍼셉트론 (MLPs) 훈련
- 기울기 소실/폭팔을 해결하기 위해 중간 계층을 보조 분류기에 직접 연결하는 방식
- highway network 연구 - gating functions 가 포함된 shortcut connection 제시
    - 데이터에 의존적이고 매개변수를 가짐
    - 게이트된 shortcut 이 닫히면 highway network 층들은 non-residual 함수를 나타냄.
    - 이 연구에서 사용한 identical shortcut 과 달리 깊이가 극도로 증가했을 때 정확도 향상을 보여주지 못함

3. Deep Residual Learning

Residual Learning

만약 여러 비선형 층들이 점근적으로 복잡한 함수를 근사할 수 있다고 가정한다면 이는 그들이 점근적으로 잔차 함수들 (H(x)-x) 를 근사할 수 있다고 가정하는 것과 같음. (입력과 출력이 같은 층일때)

- **성능 저하 문제에서 직관에 반하는 현상**에 의해 동기가 부여됨.
- Residual Learning 재공식화에서 만약 항등 매핑이 최적이라면 solver 들은 단순히 여러 비선형 층들의 가중치를 0에 가깝게 만들에 항등 매핑에 접근할 수 있음
    - 실제 경우에서 identity mapping 이 최적일 가능성은 낮지만 재공식화는 문제를 전처리(precondition) 하는 데 도움될 수 있음
    - 만약 최적 함수가 0 mapping 보다 idenetity mapping 에 가깝다면 solver 가 이 함수를 새로운 것으로 학습하기보다는 identity mapping 을 기준으로 perturbations 를 찾는 것이 쉬울 것

✅ **0 매핑 VS 항등 매핑**

- Plain Net 에서 학습의 출발점 = 0 매핑

  $F(x) \approx H(x)$,
  $F(x) = H(x) = 0$

- ResNet 에서 학습의 출발점 = 항등 매핑

  $F(x) = H(x) - x$,
  $H(x) = x$,
  $F(x) = H(x) - x = 0$

  - 실험에서에서 학습된 잔차 함수들이 일반적으로 작은 반응을 가진다는 것을 보여주었다.

    → 각 층이 만들어내는 반응이 작은 보정 값 정도, 기존 입력을 약간 수정하는 정도로 학습한다.

    ⇒ "항등 매핑을 기준점(precondition) 으로 두고 거기에 조금만 보정하면 된다" 는 ResNet 의 의도와 맞음

Identity Mapping by Shortcuts

- buliding block

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

  - F = W2σ(W1x)
  - σ는 ReLU
  - F + x -shortcut connection 과 element-wise addition 에 의해 수행
  - addition 후에 non-linearity (σ) 추가

- x 와 F 의 차원은 동일해야 함. 만약 그렇지 않다면 linear projction(W_s) 를 수행해서 차원을 맞춰 주어야 함

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

- 완전연결층 뿐만 아니라 합성곱층에도 적용 가능하다.

Network Architectures

1. Plain Network - VGG19 네트워크
    - 합성곱 계층들은 대부분 3x3 필터 가짐
    - 출력 feature map 크기가 동일할 경우 동일한 수의 필터를 가짐
    - feature map 크기가 절반으로 줄어들 경우 필터 수를 두 배로 늘림
    - stride 2, global average pooling, softmax 1000 완전연결 계층
    - 총 가중치 수 34

    ResNet 모델이 VGG 네트워크보다 더 적은 필터와 낮은 복잡성을 가진다.
2. Residual Network

    plain 네트워크를 기반으로 shortcut 연결들을 삽입하여 네트워크를 대응되는 잔차 버전으로 변환.

    - 차원이 증가할 때 :

        (A) shortcut 은 여전히 항등 매핑을 수항하되, 차원을 늘리기 위하 추가적인 0 항목들을 패딩. - 새로운 매개변수를 도입하지 않음

        (B) projection shortcut 을 사용하여 차원 맞추기

Implementation

ImageNet 데이터

- 짧은 변이 [256,480] 구간에서 무작위로 샘플링되어 크기 보강 수행
- 224×224 크롭은 이미지에서 무작위로 샘플링되거나 수평 뒤집기를 한 후 샘플링, 픽셀별 평균이 차감
- standard color augmentation 사용
- 각 합성곱 직후 활성화 이전에 배치 정규화 적용
- 미니배치 크기 256으로 SGD 사용.
    - 학습률 0.1 에서 시작
    - 오차가 plateau 에 도달하면 10배씩 감소하고
    - 최대 60x10^4 iteration

    - dropout 사용 X
- 테스트 시 10-crop 테스트 채택

4. Experiment

ImageNet Classification

1000개의 클래스로 구성된 ImageNet 2012 분류 데이터셋 사용.

Plain Networks

- 18계층/34계층 plain 네트워크 평가
- detailed architectures

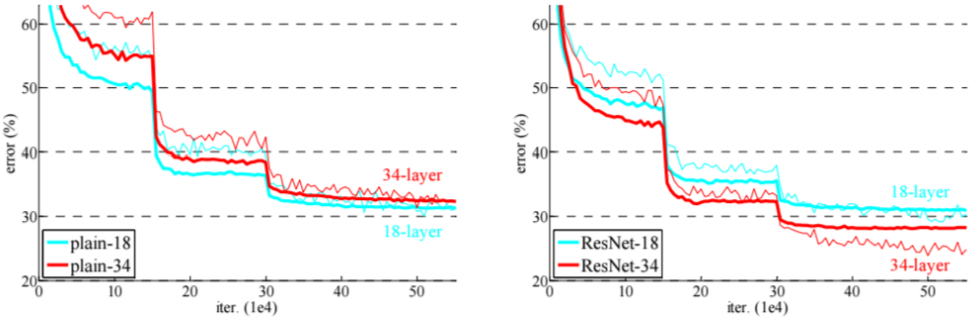
layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
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

- results

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

  Plain Network 의 경우 validation 오류가 34 layer 가 더 높다.

- degration problem



- plain network 의 경우, 훈련 과정에서 34계층 네트워크가 18계층 네트워크보다 높은 train 오류를 가짐.
- introduction 에서 언급했듯이 18계층 네트워크의 solution space 가 34계층 네트워크의 부분공간임에도 이러한 결과
- **원인이 기울기 소실 문제인 것은 아님**

  plain 네트워크들은 BN 으로 학습되며, 이는 순전파된 신호가 0이 아닌 분산을 가지도록 보장. 또한 역전파된 기울기들이 healty norm 을 유지한다는 것도 증명함.

✅ **Vanishing Gradient**

- 딥러닝에서 층이 많아질수록 역전파로 전달되는 기울기가 점점 작아서 0에 가까워지는 현상.

1. BN(Batch Nomalization) 을 사용
   BN 은 각 층의 출력을 평균 0 / 분산 1 로 정규화해서, 신호가 너무 커지거나 작아지는 것 방지하고, 순전파된 신호가 항상 0이 아닌 분산을 가지게 보장해 줌
2. 역전파된 기울기도 정상적
   역전파시 기울기도 소실되거나 폭팔하지 않고 정상적인 범위를 유지한 것을 확인.
3. 경쟁력 있는 정확도 달성

   만약 기울기 소실 문제라면 학습이 진행되지 못해야 하는데, 34계층 plain 네트워크도 경쟁력 있는 성능을 보임.

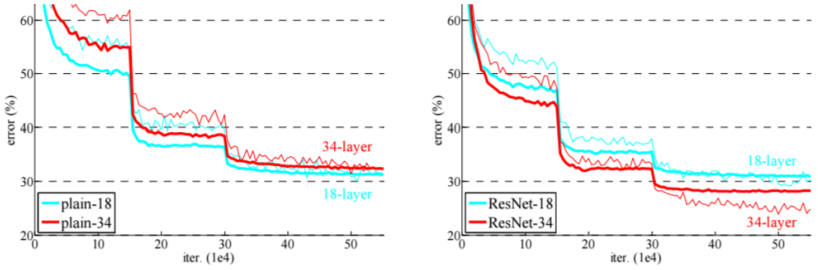
model	top-1 err.	top-5 err.
VGG-16 [40]	28.07	9.33
GoogLeNet [43]	-	9.15
PReLU-net [12]	24.27	7.38
plain-34	28.54	10.02
ResNet-34 A	25.03	7.76
ResNet-34 B	24.52	7.46
ResNet-34 C	24.19	7.40
ResNet-50	22.85	6.71
ResNet-101	21.75	6.05
ResNet-152	**21.43**	**5.71**

⇒ 깊은 plain 네트워크가 지수적으로 낮은 수렴 속도를 가지고, 이것이 학습에 영향을 준다고 추정

Residual Networks

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

- ResNet 은 top-1 오류를 3.5% 개선
- 18계층 plain/residual 네트워크들은 성능 비슷 - 현재의 SGD solver 가 plain 에서도 충분히 좋은 해를 찾을 수 있음을 의미



- 하지만 residual 네트워크를 사용할 때 더 빠르게 수렴함으로써 최적화를 용이하게 함.

Identity vs Projection Shortcuts

✅ 3가지 Option

(A) 차원을 늘릴 때 zero-padding shortcut이 사용되며, 모든 shortcut은 매개변수가 없는 방식

(B) 차원을 늘릴 때 projection shortcut이 사용되며, 다른 shortcut들은 identity.

(C) 모든 shortcut이 projection

model	top-1 err.	top-5 err.
VGG-16 [40]	28.07	9.33
GoogLeNet [43]	-	9.15
PReLU-net [12]	24.27	7.38
plain-34	28.54	10.02
ResNet-34 A	25.03	7.76
ResNet-34 B	24.52	7.46
ResNet-34 C	24.19	7.40
ResNet-50	22.85	6.71
ResNet-101	21.75	6.05
ResNet-152	**21.43**	**5.71**

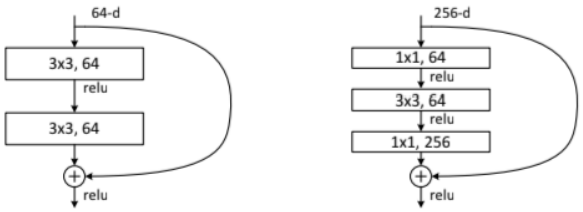
- 세 옵션 모두 plain-34 보다 좋은 성능을 보임

- A/B/C 사이의 작은 차이는 projection shortcut 이 성능 저하 문제 해결에 필수적이지 않음을 나타냄.

  ⇒ 메모리/시간 복잡성/모델 크기 문제를 줄이기 위해 옵션 C 는 제외

Deeper Bottleneck Architectures

training time 제한 때문에 building block 을 bottleneck design 으로 변경



오른쪽이 bottleneck design - 각 잔차 함수 F 에 대해 3개의 층을 사용한다. (1x1/3×3/1×1). 두 설계 모두 유사한 시간 복잡도를 가짐.

- 오른쪽 bottleneck design 에서 identity shortcut 이 projection shortcut 으로 대체되면 시간복잡도/모델 크기가 2배로 늘어남

> ✅ **projection shortcut 사용 시 비효율**
>
> - bottleneck 블록의 구조 : `1x1 → 3×3 → 1×1`
>   - 첫 번째 1x1 : channel 축소
>   - 가운데 3x3 : 특징 추출 (작은 차원에서 연산하므로 계산량 절약하는 효과)
>   - 마지막 1x1 : channel 복원
>
>   ⇒ Bottleneck 블록의 입력과 출력은 고차원
> - projection shortcut 은 1x1 conv 와 같은 연산을 통해 차원을 맞춰주므로 입력과 출력이 고차원인 Bottleneck 구조에서 projection shortcut 을 사용하면 **시간 복잡도 x2, 모델 크기 x2 수준의 부담**이 생김

⇒ identity shortcut 을 사용하는 것이 훨씬 효율적임

- 50-layer ResNet / 101-layer / 152-layer ResNet

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
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

- 2계층 블록을 3계층 bottleneck 블록으로 대체하여 34계층 네트워크를 50계층 네트워크로 만들 수 있음. (옵션 B 사용)
- 50/101/152 계층에서 38억/76억/113억 FLOPs 를 가져, VGG-16/19 네트워크 (153억/196억 FLOPs) 보다 낮은 복잡성을 가짐.

Comparisons with State-of-the-art Methods

- 152계층 ResNet 은 단일 모델 기준 검증 오류 4.49% 기록

method	top-1 err.	top-5 err.
VGG [40] (ILSVRC'14)	-	8.43†
GoogLeNet [43] (ILSVRC'14)	-	7.89
VGG [40] (v5)	24.4	7.1
PReLU-net [12]	21.59	5.71
BN-inception [16]	21.99	5.81
ResNet-34 B	21.84	5.71
ResNet-34 C	21.53	5.60
ResNet-50	20.74	5.25
ResNet-101	19.87	4.60
ResNet-152	**19.38**	**4.49**

- 6개의 모델을 앙상블했을 때, 가장 좋은 결과 (top-5 err 3.57%)

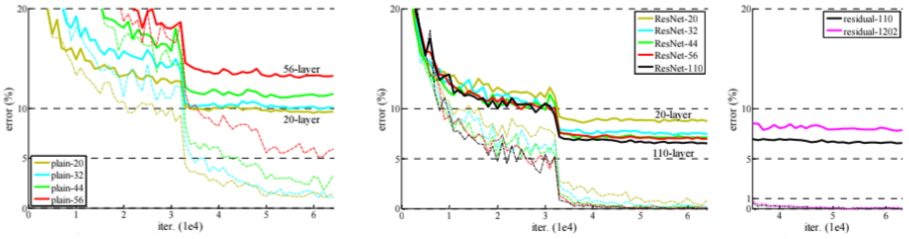
method	top-5 err. (**test**)
VGG [40] (ILSVRC'14)	7.32
GoogLeNet [43] (ILSVRC'14)	6.66
VGG [40] (v5)	6.8
PReLU-net [12]	4.94
BN-inception [16]	4.82
ResNet (ILSVRC'15)	**3.57**

CIFAR-10 and Analysis

- **plain/residual architecture : ImageNet 실험에서와 같은 architecture 사용**

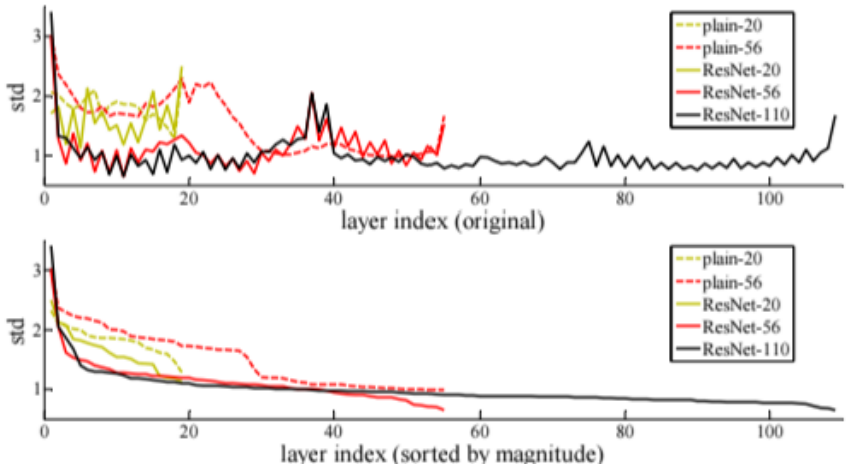
output map size	32×32	16×16	8×8
layers	1(첫번째 layer) +2n	2n	2n
filters	16	32	64

    - Global Avarage Pooling, 10-way FC, softmax 마지막 층
    - 총 6n+2 개 layer
- 학습 방식
    - 가중치 초기화
    - BN
    - learning rate - 0.1 에서 시작해 32k 와 48k 반복 시점에서 10배 감소
    - 총 64k 반복
- 데이터 보강
    - 각 변에 4픽셀 패딩 추가
    - 무작위 32x32 크롭 샘플링
    - 테스트 시에는 원본 32x32 이미지의 단일 뷰만 평가 ?
- n = {3,5,7,9} 인 경우 (20,32,44,56 계층 네트워크)



(1) Plain Network (2) ResNet (3) n=18

    - Plain Network : 깊이가 증가할수록 train error 가 증가, ImageNet 이나 MNIST 에서의 결과와 비슷함
    - ResNet : ImageNet 과 비슷하게 최적화에서의 어려움을 극복하고 깊이가 증가할수록 정확도 향상을 보여줌
    - n=18 (110계층 ResNet) 에서도 여전히 잘 수렴하며, 더 얕은 모델들보다도? 적은 매개변수를 가짐.
- Analysis of Layer Responses



layer responses(BN 이후 ReLU/덧셈 연산 전 반응) 의 표준편차

- ResNet 이 일반적으로 Plain Network 보다 더 작은 반응을 가진다는 것을 알 수 있음.

  → 잔차 함수들이 일반적으로 0에 더 가깝다는 (적게 수정한다는) 기본 동기에 부합

  층이 많아질수록 ResNet 의 개별 층에서 신호를 적게 수정한다는 것 또한 관찰 가능

- Exploring Over 1000 Layers

  n=200 으로 설정하고 1202계층 ResNet 을 실험했을 때, training error는 110계층 ResNet 과 비슷하게 <0.1% 의 결과를 얻었지만 test error 의 경우 7.93 으로 110계층 ResNet 보다 나쁘며, 여전히 train 과 test error 에 차이가 있음.

  → Overfitting 문제가 발생했음을 알 수 있음.

Object Detection on PASCAL and MS COCO

- PASCAL VOC 2007/2012

training data	07+12	07++12
test data	VOC 07 test	VOC 12 test
VGG-16	73.2	70.4
ResNet-101	**76.4**	**73.8**

- MS COCO

metric	mAP@.5	mAP@[.5, .95]
VGG-16	41.5	21.2
ResNet-101	**48.4**	**27.2**

VGG-16을 ResNet-101 로 대체함으로써 두 다른 인식 과제에서도 유의미한 성능 향상을 보임. 두 모델에서 사용된 detection implementation 은 동일하기 때문에 네트워크에서 기인한 차이라고 할 수 있음.