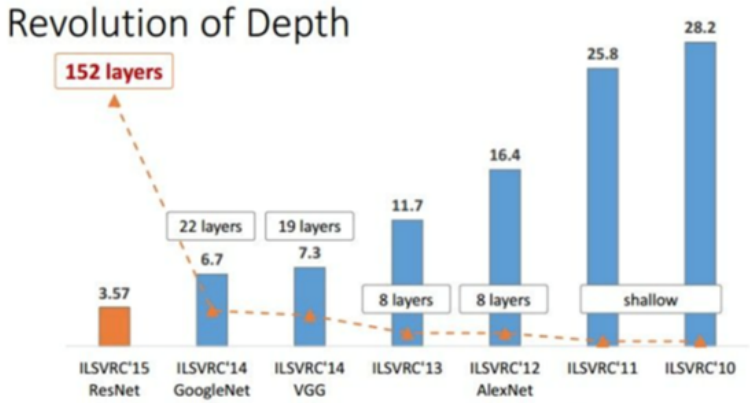
Top-5 Classification Error



Kaiming He

1.Intro
~~2.Related Work~~
3.Deep Residual Learning
4.Experiments
4.1Image Net Classification
~~4.2CIFAR-10 and Analysis~~

Abstract

더 깊은 신경 네트워크(Deeper neual netwoks)는 학습(train)시키기 더 어렵다. 이전에 사용 된 것보다 훨씬 더 깊은 네트워크의 훈련을 쉽게하기위한 residual learning framework를 제안한다. 참조되지 않은 함수(unreferenced function)를 학습하는 대신 레이어 입력값(layer input)을 참조하여 residual function를 학습하는 것으로 레이어를 명시 적으로 재구성한다.
포괄적인 경험적 증거를 통해 residual networks가 쉽게 최적화 될 수 있고, 상당한 깊이에서 정확성(considerably increased depth)을 얻을 수 있다는 증거를 보여준다.

ImageNet 데이터 셋에서 최대 152 개의 레이어 residual net을 평가한다. VGG nets보다 8배 더 깊지 만 여전히 복잡성은 낮다.
이 An ensemble of residual nets 은 ImageNet 테스트 세트에서 3.57 %의 오류를 달성하였다.
이 결과는 ILSVRC 2015 분류 작업에서 1 위를 차지했다.
또한 CIFAR-10에서의 100개와 100개 레이어에 대한 분석도 보여준다.

visual recongition tasks에서 표현의 깊이(depth of representations)는 핵심적인 부분이다.
단지 우리의 (극단적인 표현 깊이 extremely deep representations) 방법만으로 , COCO 객체 탐지 데이터 셋에서 28%의 정확도 향상을 달성했다.
Deep residual nets는 ILSVRC와 COCO 대회 제출물의 기초이며, 뿐만 아니라 ImageNet detection, ImageNet localization, COCO detection, COCO segmentation의 1 위를 차지했다.

---

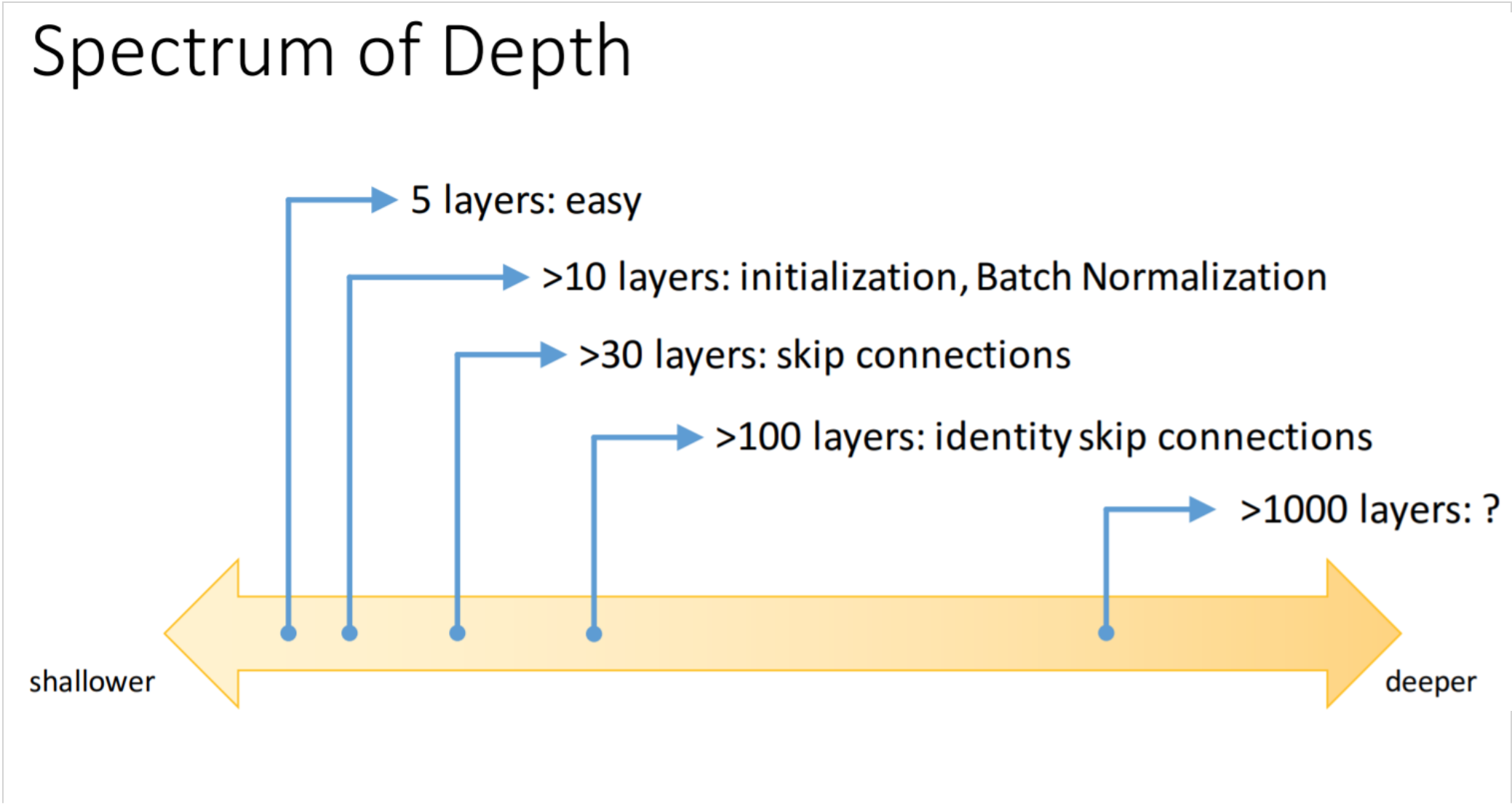
1. Introduction

Deep CNN은 이미지 분류를 위한 일련의 획기적인 발전을 가져왔다 . Deep Networks는 low / mid / high levle features 와 classifiers 를 종단 간(end-to-end) multi layer 방식으로 자연스럽게 통합하며 features의 "levels"은 누적(stacked) 레이어 수 (깊이:deep)로 풍부해질 수 있다.
최근의 증거들은 **네트워크 깊이가 매우 중요**하다는 것을 보여 주며, 도전적인 ImageNet 데이터 셋 의 주요 결과는 모두 "very deep" 16 ~ 30개의 모델에서 공훈을 이루었다 . 많은 다른 중요한 시각 인식 작업(visual recognition tasks) 또한 very deep models의 도움을 많이 받았다.

깊이(depth)의 중요성에 힘 입어 질문해보자 :

그저 쉽게 레이어를 더 쌓음으로써 더 나은 네트워크 학습이 가능한가?
(Is learning better networks as easy as stacking more layers?)

원래(from the beginning) 이 질문에 답하기위한 장애물은 수렴(**convergence**)를 방해하는 악명 높은 **vanishing/exploding의 과제**이다. 그러나, 이 문제는 수십 개의 레이어가 있는 네트워크가 역 전파(back propagation)를 통한 stochastic gradient descnet (SGD)가 수렴을 시작할 수있게 해주는 , **정규화 된 초기화(normalized initialization) 와 중간 정규화 레이어(intermediate normazlizatoin layers)로 인해 해결**되었었다.



Going Deeper

- Initialization algorithms ✓
- Batch Normalization ✓

- **Is learning better networks as simple as stacking more layers?**

더 깊은 네트워크(deeper networks)가 수렴을 시작할 수 있게 되면서 ***degradation*** 문제가 드러났다. 네트워크 깊이가 증가하면 정확도가 포화 상태(saturated) (당연한 것일 수 있음)가 되고 급격히 저하(degrades)된다. 예기치 않게, 그러한 degradation는 overfitting으로 인한 것이 아니며, 적절히 깊이 모델에 더 많은 레이어를 추가하는 것은, 리포트 되고 우리의 실험으로 철저히 검증 된 것처럼, **더 높은 학습 오류(higher training error)를 초래**한다. [그림 1]은 그 일반적인 예를 보여준다.

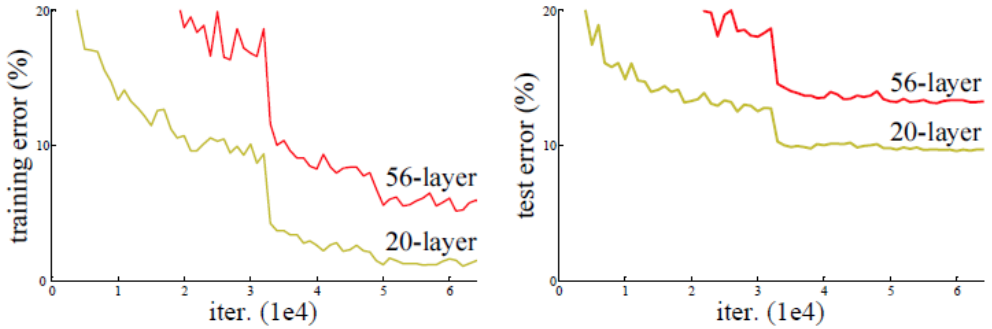


Figure 1. Training error (left) and test error (right) on CIFAR-10 with 20-layer and 56-layer "plain" networks. The deeper network has higher training error, and thus test error. Similar phenomena on ImageNet is presented in Fig. 4.

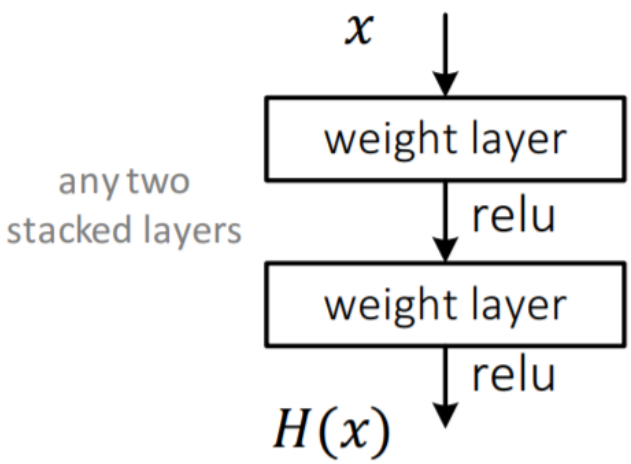
training accuracy는 degradation는 모든 시스템이 비슷하게 쉽게 최적화 할 수있는 것은 아니라는 것을 나타낸다. 더 얕은 아키텍처(shallower architecture)와 더 많은 레이어를 추가(adds more layers on it)하는 더 깊은 아키텍처(deeper counterpart)를 고려해 보자. 더 깊은 모델을 위한 구축에 의한(by constuction) 솔루션이 있다: 추가 되는 레이어(the added layers)는 identity mapping이고 다른 레이어(other layers)는 더 얕은 모델에서 이미 학습된 레이어에서 복사된다

(copied from the learned shallower model). 이 구축된 솔루션(constructed solution)의 존재는 더 깊은 모델이 더 얕은 모델보다 더 높은 학습 오류(no higher training error)를 발생시켜야 하지 않음을 나타낸다.
But, 실험을 통해 우리의 현재 current solvers on hand는 구축 된 솔루션보다 훨씬 좋거나 우수한 솔루션을 찾을 수 없다는 것을 보여준다. (또는 가능한 시간에 그렇게 할 수 없다).

본 논문에서는 deep residual learning 프레임워크를 소개하며 degradation 문제를 해결한다. 단지 몇 개의 스택 된 레이어(stacked layers)가 원하는 기본 매핑에 직접적으로 맞기를 바라지 않고 (directly fit a desired underlying mapping),이 레이어들이 residual mapping에 맞도록 명시적으로 지정한다.
형식적으로, H (x)로 원하는 기본 매핑(desired underlying mapping)을 나타내고, 스택 된 비선형 레이어들(stacked nonlinear layers)을 **다른 매핑(another mapping of) F (x) : = H (x) - x 에 맞춘다(fit). 원래 매핑은 F (x) + x로 재작성(recast)**된다. **우리의 가정은 참조되지 않은 원본 매핑(original unreferenced mapping)을 최적화하는 것보다 잔머지 매핑(residual mapping)을 최적화하는 것이 더 쉽다는 것이다.**

Deep Residual Learning

- Plaint net

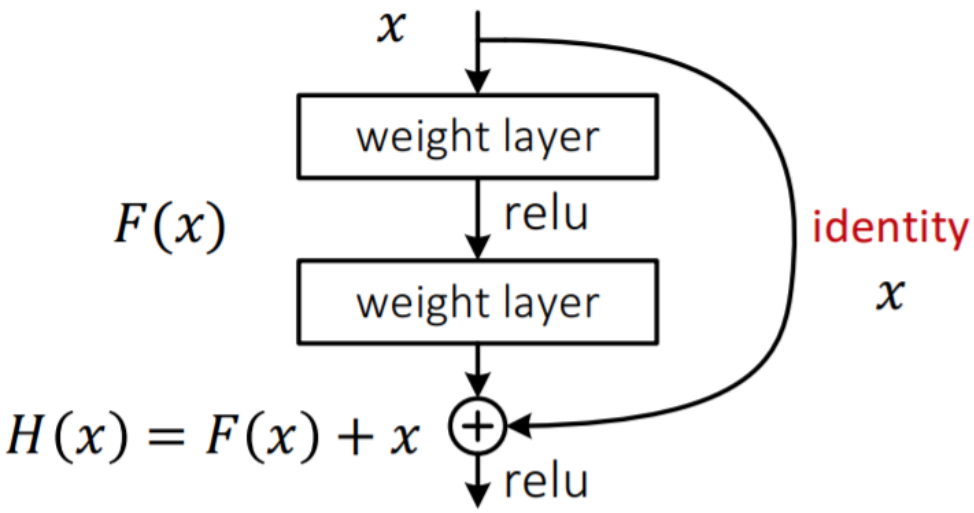


$H(x)$ is any desired mapping,

hope the 2 weight layers fit $H(x)$

Deep Residual Learning

- Residual net



$H(x)$ is any desired mapping,

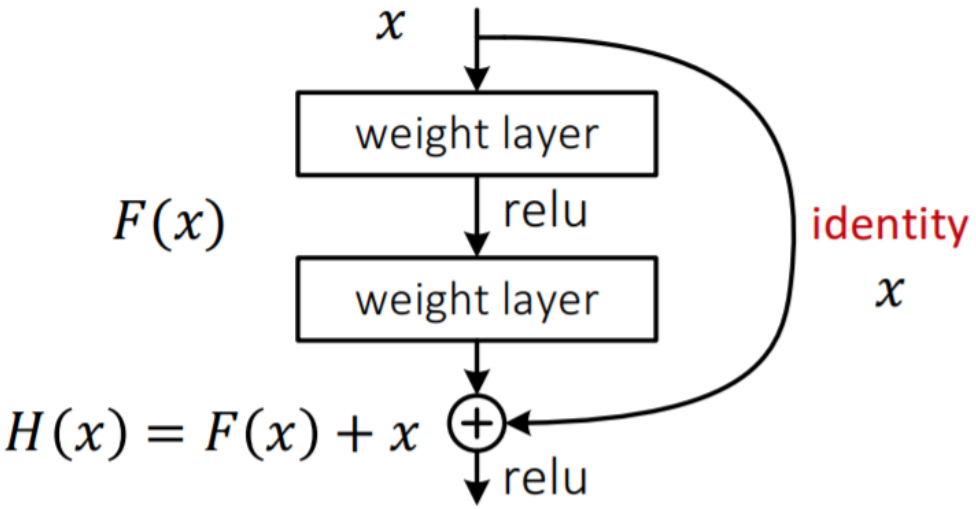
~~hope the 2 weight layers fit $H(x)$~~

hope the 2 weight layers fit $F(x)$

let $H(x) = F(x) + x$

Deep Residual Learning

- $F(x)$ is a residual mapping w.r.t. identity



- If identity were optimal, easy to set weights as 0
- If optimal mapping is closer to identity, easier to find small fluctuations

극단적으로는, identity mapping이 최적 일 경우, 비선형 레이어 스택에 의한 identity 매핑을 맞추는 것(fit an identity mapping by a stack of nonlinear layers)보다 잔차를 제로로 푸는 것이 더 쉬울 것(push the residual to zero than to fit an identity mapping)이다.

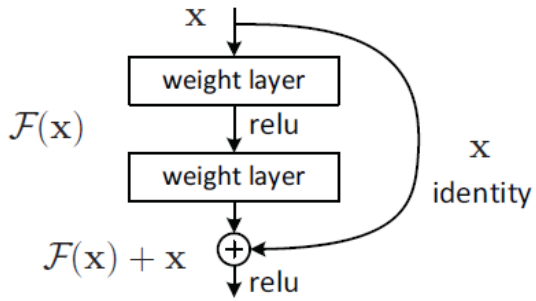


Figure 2. Residual learning: a building block.

F (x) + x의 공식은 "shortcut connections"을 갖는 피드 포워드 신경망(feedforward neural networks)에 의해 실현 될 수 있다(can be realized by) (그림 2). shortcut connections은 하나 이상의 레이어를 건너 뛰는 연결이다. 우리의 경우, shortcut connections 신원 매핑을 수행(perform identity mapping)하고, 그들의 출력(outputs)은 스택 된 레이어의 출력에 추가된다(그림 2). identity shortcut connections는 추가 매개 변수도 계산 복잡성도 추가(요구)하지 않는다. 전체 네

트워크는 역전파(backpropagation)를 이용한 SGD로 쉽게 학습(trained end-to-end) 될 수 있으며, 이미 구현된 solvers를 수정하지 않고도 일반 라이브러리 (예 : Caffe)를 사용하여 손쉽게 구현할 수 있다.

우리는 ImageNet 으로 종합적인 실험을 통해 성능 저하 문제(degradation problem)를 보여주고 우리의 방법을 평가(evaluate out method)한다. 우리는 이하에 대해 증명한다:

1) 우리의 extremely deep residual nets은 최적화하기 쉬운 반면, "plain" net (simply stack layers)은 깊이가 증가 할 때 higer training error을 보인다.
2) 우리의 deep residual nets는 크게 증가한 깊이에서도 쉽게 정확도를 보이며(enjoy accuracy gains), 이전 네트워크보다 훨씬 나은 결과를 제공합니다.

CIFAR-10 셋 에서도 이와 유사한 현상이 나타나며, 이는 최적화 어려움과 우리의 방법이 효과가 특정 데이터 셋에서만 유효하지 않음을 시사한다.
우리는 이 데이터 셋에서 100 개가 넘는 레이어로 학습한 성공적인 모델을 제시하고, 1000 개 넘는 레이어가 있는 모델을 탐색해본다.

우리는 ImageNet 분류 데이터 셋에서 extremely deep residual nets로 우수한 결과를 얻는다. 우리의 152- 레이어 residual net 은 ImageNet에 제시된 가장 깊은 네트워크이며 VGG보다 낮은 복잡도를 가진다. 우리의 앙상블은 theImageNet 테스트 셋에서 3.57 %의 top-5 오류를 가지며 ILSVRC 2015 분류 경쟁에서 1 위를 차지했다. The extremely deep representations 는 다른 인식 작업(recognition task)에서도 탁월한 일반화된 성능을 보여 주며 나아가 ILSVRC & COCO 2015 대회의 ImageNet 감지(detection), ImageNet 현지화(localization), COCO 감지(detection) 및 COCO 분류(segmentation)에서도 우승을 이끌어 냈다. 이 강력한 증거는 residual learning principle (residual 학습 원리)가 일반적이라는 것을 보여 주며, 다른 비전 및 비 시각 문제에도 적용 가능할 것으로 기대한다.

---

2. Related Work

~~**Residual Representations**~~

~~**Shorcut Connections**~~

---

3. Deep Residual Learning

3.1 Residual Learning

H (x)를 몇 개의 스택 된 레이어(few stacked layers) (반드시 전체 신경망일 필요는 없지만)가 fit한 기본 매핑으로 간주하고, x를 이러한 레이어의 첫 번째 입력을 나타내보자.
다중 비선형 레이어(multiple nonlinear layers)가 복잡한 함수를 점근적으로 근사(모델링) 할 수 있다고 가정하면, residual function(:H (x) - x ~ input과 ouput의 차원이 같다고 가정했을때)에도 점근적으로 근사 할 수 있다는 가정과 동등하다. 따라서, 스택 된 레이어가 H (x)를 근사화 할 것(apsymptotically approximate 정답을 찾는 거대한 함수!)을 기대하기보다는, 이들 레이어를 잔여 함수 F (x) : = H (x) - x 에 근사한다고 보면, (스택된 레이어가 근사 하는) 원래 함수는 F (x) + x가됩니다. 가정에 의하면 두 가지 형태 모두가 원하는 함수를 점근적으로 근사시킬 수 있어야 하지만, 학습의 용이성(ease of learning)은 다를 수 있다.

이 재구성(reformulation)은 degradation problem에 대한 반 직관적인 현상(counterintuitive pehonomena) 에 영감을 받았다. (그림 1, 왼쪽). introduction에서 논의했듯이 추가 된 레이어를 identity mappings 으로 구성될 수 있다면, 더 깊은 모델은 더 얕은 레이어보다 더 큰 training error가 없어야 한다.
degradation problem은 solvers가 다중 비선형 레이어로는 identity mappings를 근사하기에 어려움이 있을수 있다는 것을 의미한다.(difficulties in approximating identity learning reformulation). residual learning 재구성(reformulation)으로, 만약 identity mappings이 최적이라면, solvers는 간단하게 identity mappings에 approach 하기 위해 다중 비선형 층의 가중치를 0으로 유도 할 수 있다.

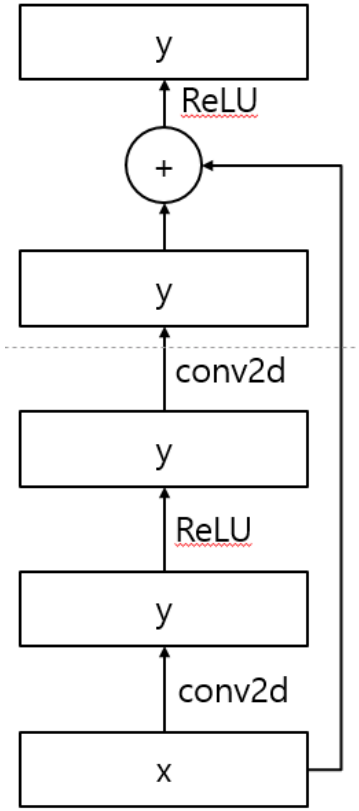
실제 상황에서 identity mapping이 최적(optimal)이 아닐지라도, 우리의 재구성(reformulation)은 문제의 기본 전제(precondition)가 되는 데 도움이 될 수 있습니다.
만약 최적 함수(optimal function)가 제로 매핑(zero mapping)보다 identity mapping에 더 가깝다면, solver가 identity mapping을 참조하여 변화(perturbation)를 발견하는 것이 새 함수로 학습하는 것보다 쉽다(learn the function as a new one). 학습 된 (learned) residual function는 일반적으로 small responses을 가지며, identity mapping이 합리적인 전제 조건을 제공한다는 것을 실험 (그림 7)으로 보여준다.

3.2. Identity Mappings by Shortcuts

우리는 몇 개의 stacked layers 마다 residual learning을 적용한다. 빌딩 블록은 그림 2에 나와있다. 공식적으로, 본 논문에서는 다음과 같이 정의 된 빌딩 블록을 고려한다 :

$$y = F(x, \{W_i\}) + x \quad \dots (1)$$

여기서 x와 y는 레이어의 입력 및 출력 벡터이다. 함수 F (x, {Wi})는 학습 되야하는 residual mapping을 나타낸다. 예를 들어 그림 2는 2개의 레이어를 가지고, F = W2 σ (W1x) 에서 σ 는 ReLU를 의미하며, 표기법을 단순화하기 위해 biases 는 생략된다. **F + x는 shortcut connection 및 element-wise addition로 수행된다. addition 후에 두번째 비선형을 적용한다 (즉, σ (y) 그림 2 참조).** σ(시그마 Sigma)
식 (1)의 shortcut connection은 추가 매개 변수도 계산 복잡성도 없다. 이는 실제적으로 매력적 일뿐만 아니라, 일반(plain) 네트워크와 residual 네트워크 간의 비교에서도 중요하다. 동일한 수의 매개 변수, 깊이, 너비 및 계산 비용 (무시할 수있는 element-wise addion은 제외)을 동시에 가지는 plain / residual 네트워크를 공정하게 비교할 수 있다.



식 (1)에서 x와 F의 차원은 같아야한다. 그렇지 않은 경우 (e.g.., 입력 / 출력 채널을 변경할 때), 차수를 맞춰주기 위해 shortcut connections 으로 선형 투영(linear projection) Ws를 수행 할 수 있다.

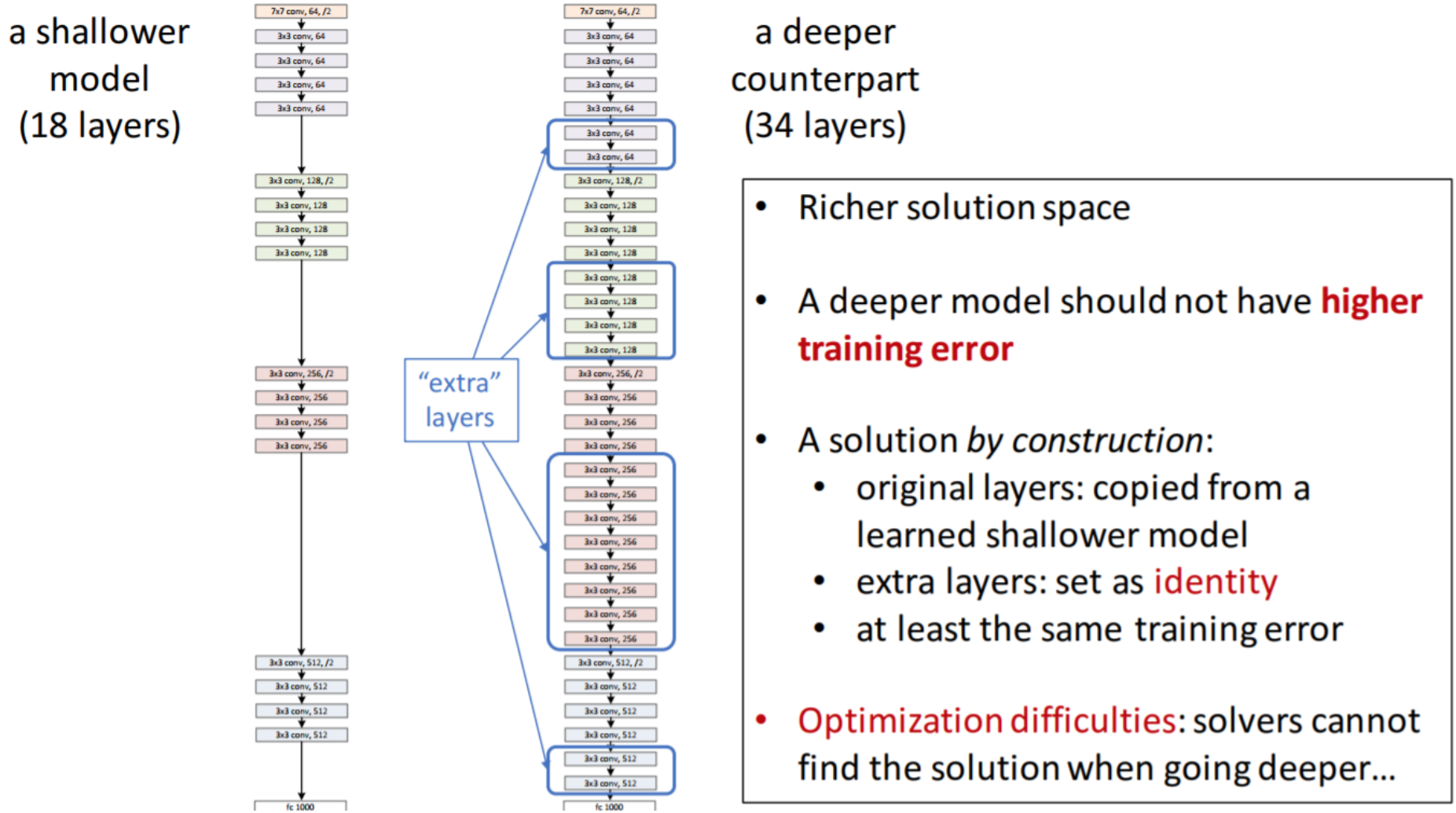
$$y = F(x, \{W_i\}) + W_s x \quad \dots (2)$$

또한 식 (1)에서 정방 행렬(square matrix)을 사용할 수도 있습니다. 그러나 우리는 identity mapping이 degradation problem을 해결하기에 충분하고 효율적이라는 것을 실험에 의해 보여줄 것이므로. Ws는 차원을 맞출 때만 사용한다.
residual function F의 형태(form)는 유연(flexible)하다. 이 논문의 실험은 두 개 또는 세 개의 레이어 (그림 5)가있는 함수 F를 보여주지만 더 많은 레이어가 가능하다. 그러나 F가 단 하나의 층만을 갖는다면, 식 (1)은 선형 레이어와 유사하다 : y = W1x + x, 그리고 우리는 여기에서의 이점은 관찰하지 못했다.

위의 표기법은 문제를 간단하기 위해 fully-connected layer에 관한 것이지만 convolutional layer에 적용 할 수 있다. 함수 F (x; {Wi})는 다중 컨벌루션 레이어를 묘사할 수 있다. element-wise addition은 채널별로 두 개의 feauture maps 맵에서 수행됩니다.

3.3 Network Architectures

우리는 다양한 plain / residual nets를 테스트했으며 일관된 현상을 관찰했다. 토론거리 제공하기 위해 ImageNet의 두 가지 모델을 아래와 같이 설명하겠다.



Plain Network.
plain baselines는 (그림 3, 가운데)은 주로 VGG nets의 철학에 영감을 받았다 (그림 3, 왼쪽). 컨볼 루션 레이어는 대개 3 ~ 3 개의 필터를 가지며, 다음과 같은 두 가지 간단한 디자인 규칙을 따릅니다. (i) 동일한 출력 피쳐 맵 크기의 경우 레이어의 필터 수가 동일합니다. (ii) 피쳐 맵 크기가 절반 인 경우, 레이어 당 시간 복잡성을 보존하기 위해 필터의 수가 2 배가 된다. stride 2 씩 가지는 컨벌루션 레이어를 통해 직접 다운 샘플링(downsampling)을 수행한다. 네트워크는 끝에서 global average pooling layer와 softmax가있는 1000 가지 fully-connected layer를 가진다. 총 어(weighted layers) 수는 그림 3 (중간)에서 34 개이다.
우리 모델은 VGG nets 보다 필터가 적고 복잡성이 적다 (그림 3, 왼쪽). 우리의 34 레이어 baseline은 VGG-19 (196 억 FLOPs: floating point operations per second 초당 부동소수점 연산)의 18 %에 불과한 36 억 FLOP (multiply-adds)를 가지고 있습니다.

Residual Network.
counterpart residual version로 전환하기 위해 위의 plain network를 기반으로 shortcut coneections을 넣어보자 (그림 3, 오른쪽). identity shortcuts (식 (1))은 입출력이 동일한 차원 일 때 직접 사용될 수 있다 (그림 3의 실선 바로 가기). 차원이 증가하면 (그림 3의 점선 단축키), 우리는 두 가지 옵션을 고려한다 : (A) shortcut는 zero padded를 하여 identity mapping을 수행한다. 이 옵션은 매개 변수를 추가하지 않습니다. (B) 방정식 (2)의 projecion shortcut는 차원을 일치시키는 데 사용된다. (1 by 1 conv). 두 가지 옵션 모두 stride 2를 주고 feature map 2개에 적용된다.

3.4 Implementation

ImageNet으로 scale augmentation을 하는데 [256; 480] 내에서 랜덤으로 짧은쪽으로 resize 한다. 224 x 224 crop은 이미지 또는 수평 플립(horizontal flip)에서 무작위로 샘플링되며 픽셀 당 평균값을 뺀다(crop flip color augmentation ~ 표준적인 color augmentation이 사용된다.) BN(Batch Norm)을 각각의 conv 실행 후와 activate 전에 적용한다. 가중치를 초기화한다. 우리는 SGD와 미니배치사이즈 256을 돌린다. learning rate을 0.1에서 시작하여 오류가 안정되면 10으로 나눈다. 모델은 최대 60 * 10^4 회 반복 학습됩니다. weight decay는 0.0001을 쓰고 momentum은 0.9을 사용하였고 drop out은 쓰지 않았다.

테스트에서 비교 연구를 위해 표준적인 10-crop testing을 채택한다. 최상의 결과를 위해서는, fully-convolutional form 하고, 다중 스케일에서 점수를 평균화합니다 (짧은면이 f{24, 256, 384, 480, 640} 중에서 resize된다).

4.Experiments

4.1. ImageNet Classification
2012년 데이터 셋 가지고 top-1 과 top-5 error rates를 봤다.

- Top 5 Classification Error
  - Top 5 Classification: Top 5 분류를 찾아서 한개만 맞아도 맞는 것, 이 중 한개도 안들어 가 있으면 틀린것으로 고려하는 문제 (인간의 성능은 5%로 알려져 있다.)

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

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

Plain Networks.
34-layer has higher validation error than the shallower 18-layer plain net.
To reveal the reasons, in Fig. 4(left) we compare their training/validation errors during the training procedure.

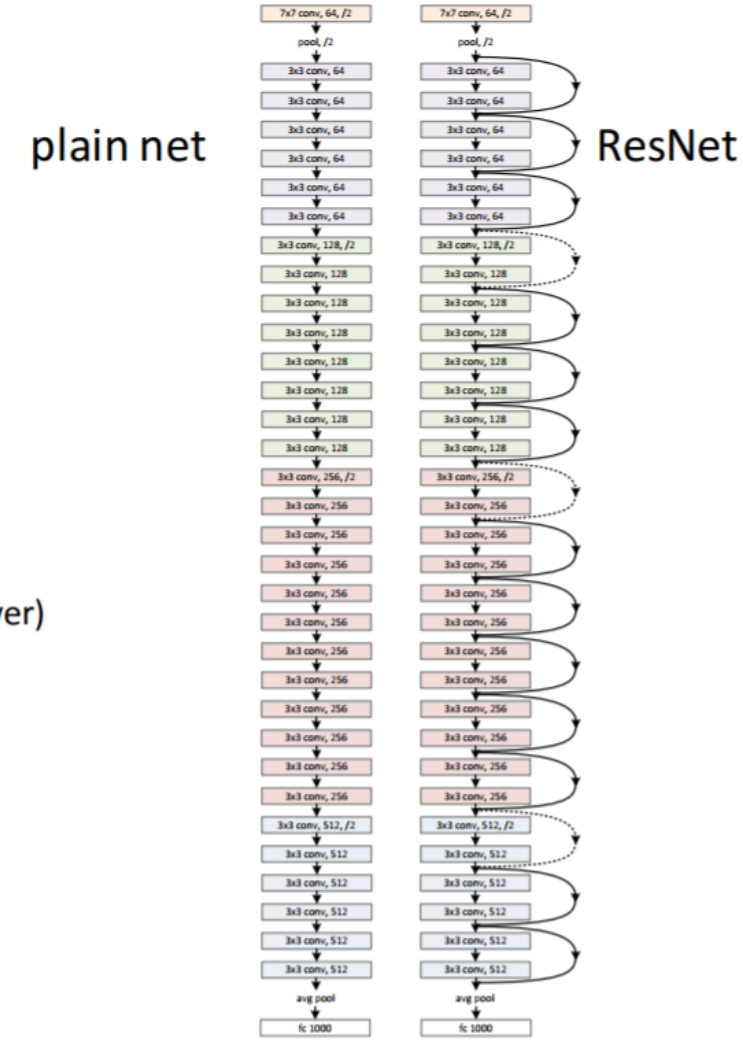
we have observed degradation problem - the 34 layer plain net has higher training error throughout the whold training procedure.

unlikely to be caused by vanishing gradient .. trained with BN, which ensures forward propagated signals to have non-zero variances.
we also verify that the backward propagated fradients exhibit healthy norms with BN.
(neither forward nor backward signals vahish)

deep plain nets may have exponentially low convergence rates, which impact reducing of the training error.

Network "Design"

- Keep it simple
- Our basic design (VGG-style)
  - all 3x3 conv (almost)
  - spatial size /2 => # filters x2 (~same complexity per layer)
  - Simple design; just deep!
- Other remarks:
  - no hidden fc
  - no dropout



Residual Networks.

evaluate 18 layers and 34 layers ReNets, baseline architecture are the same as the above plain nets,

shotcut connection is added to each pair of 3 by 3 filters as in Fig. 3(right)
In the First comparison we use identity mapping for all shorcuts and zero-padding for increasing dimensions.
So they have no extra parameter compared to the plain counterparts.

three major observation
1.34-ResNet better then 18-Resnet by 2.8% / lower training error is generalized to the validation data: **degradation p is well addressed(obtain accuracy gains from increased depth)**
2.compared to plain counterpart, top1-error 3.5% reduces(Table2). & reducing the training error(Fig. 4): **effectiveness of residual learning**
3.18-layer plain/residual nete are comparably accurate, but the 18-ResNet converges faster(Fig4. right vs.left): **when the net is not overly deep, SGD solver able to find solutions to the plain net**
ResNet providing faster convergence at the early stage

	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.

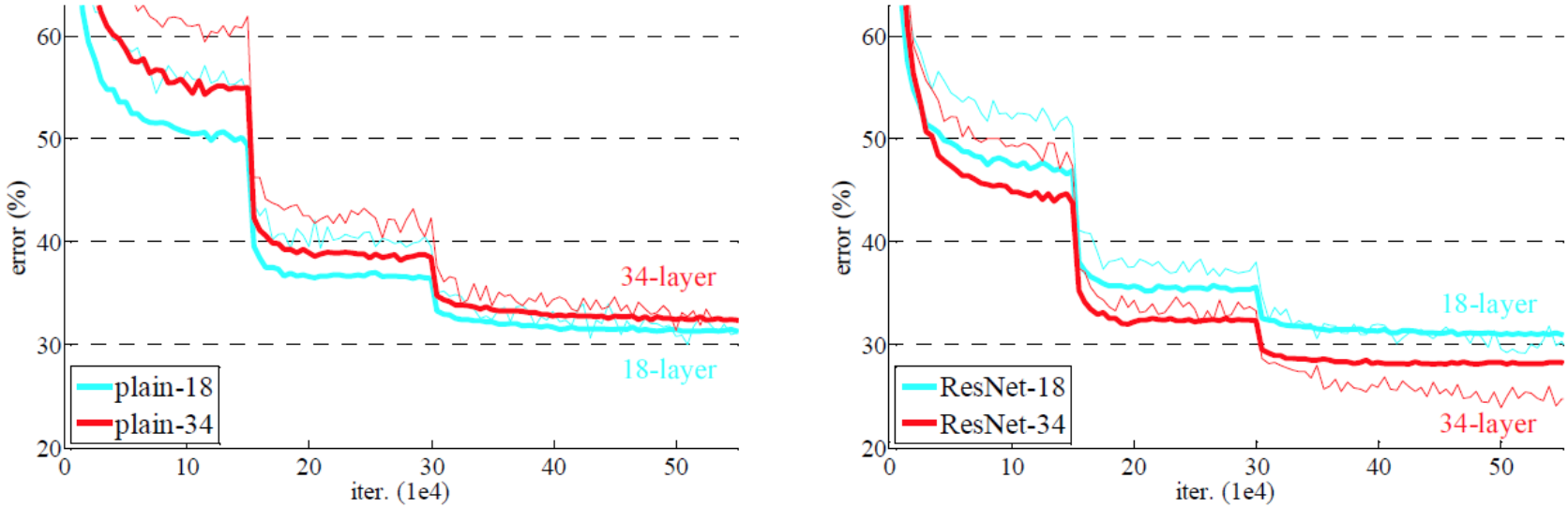


Figure 4. Training on **ImageNet**. Thin curves denote training error, and bold curves denote validation error of the center crops. Left: plain networks of 18 and 34 layers. Right: ResNets of 18 and 34 layers. In this plot, the residual networks have no extra parameter compared to their plain counterparts.

Identity vs. Projection Shorcuts.

A: zero-padding shorcuts are used for increasing dimensions, and all shortcuts are parameter-free
B: projection shortcuts are used for increasing dimensions, and other shortcuts are identity
C: all shortcuts are projections

C(maginally) > B(slightly) > A

B>A: zero-padded dimensionsin A indeed have no residual learning
C>B: extra parameter introduced by many(thirteen)projection shorcuts

BUT, not essential for addressing the degradation problem.
so not C ~ reduce memory/time comlexity and model size.

identity shorcuts는 아래에 소개 된 병목 아키텍처(bottleneck architectures)의 복잡성을 높이 지 않는데서 특히 중요하다.

Deeper Bottleneck Architectures.

where the 11 layers are responsible for reducing and then increasing (restoring) dimensions, leaving the 33 layer a bottleneck with smaller input/output dimensions.
<= 1by1 레이어로 차수를 줄인후에 복원하므로, 3by3레이어의 차수는 입/출력을 작게 받는다.

identity shortcuts lead to more efficient models for the bottleneck desighns

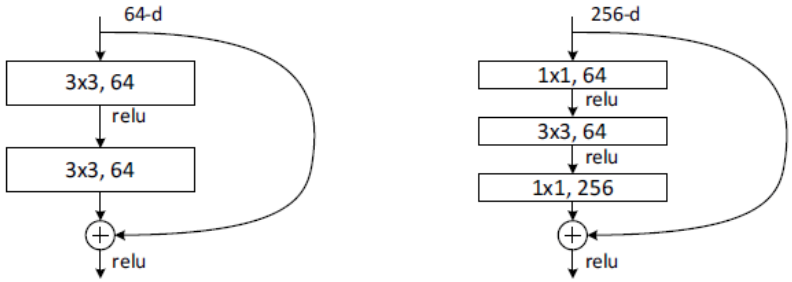
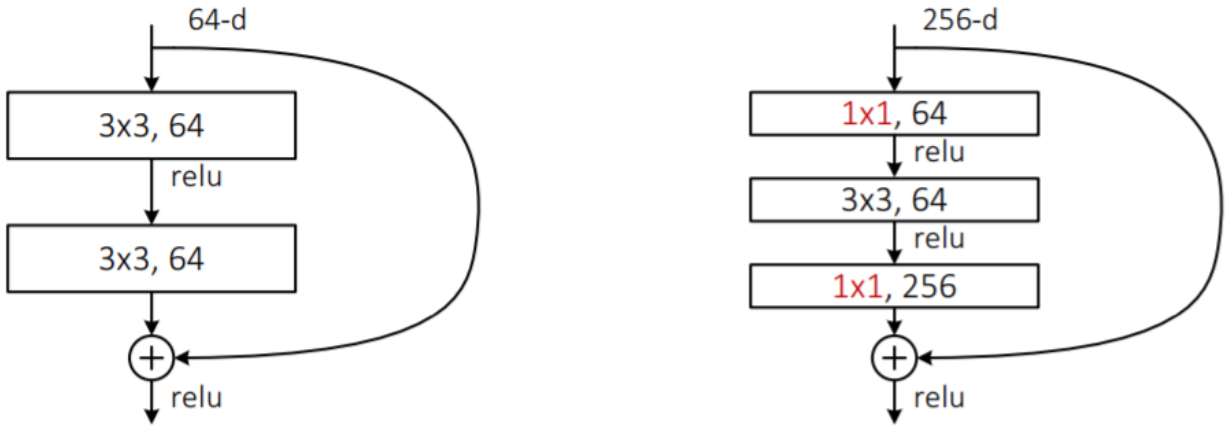
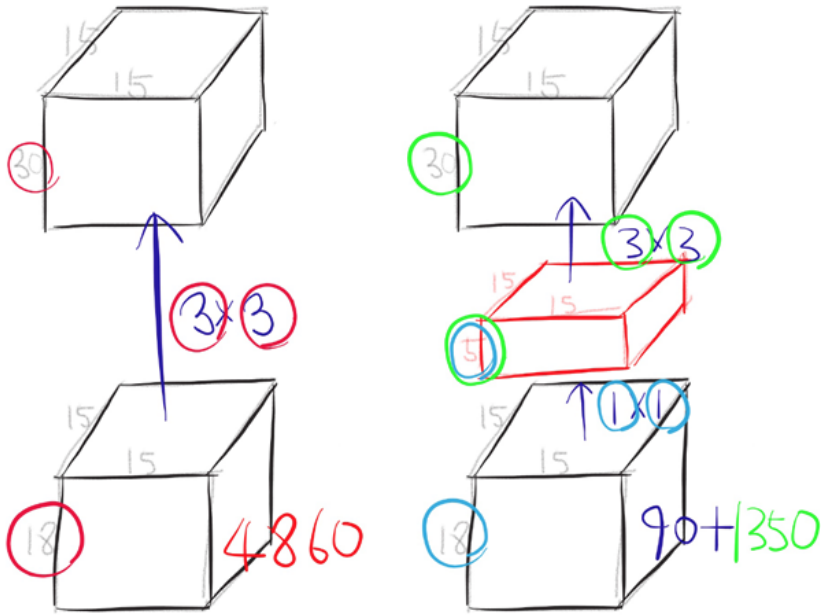


Figure 5. A deeper residual function $\mathcal{F}$ for ImageNet. Left: a building block (on 56×56 feature maps) as in Fig. 3 for ResNet-34. Right: a "bottleneck" building block for ResNet-50/101/152.

- A practical design of going deeper





(from 34-layer) 50-layer(option B for increasing dimensions, from 2 to 3-layer bottleneck block)
152-layer: depth is significantly increased, and still ahs lower complexity + accurate ... no degradation problem with depth and enjoy significant accuracy gains

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

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

Comparisons with State-of-the-art Methods.
Single-model outperforms all previous ensemble reult.
combine six models of different depth to from an ensemble, 3.5% top-5error on the test set: ILSVRC 2015

4.2. CIFAR-10 and Analysis
focus on the behaviors of extremely deep networks, use simple architectures
plain/residual is Fig.3(middle/right)
...

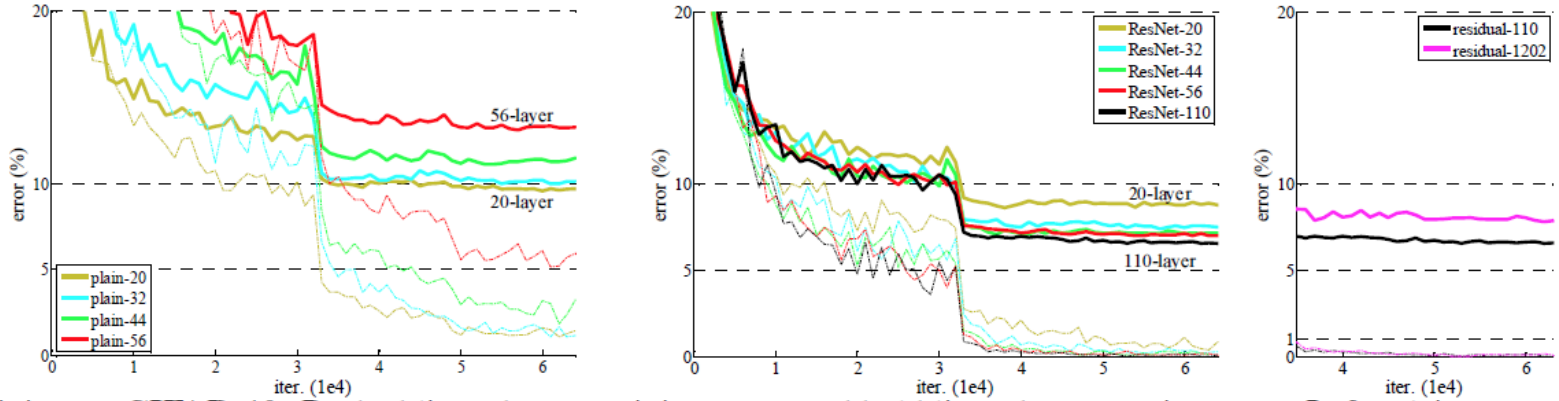


Figure 6. Training on **CIFAR-10**. Dashed lines denote training error, and bold lines denote testing error. **Left**: plain networks. The error of plain-110 is higher than 60% and not displayed. **Middle**: ResNets. **Right**: ResNets with 110 and 1202 layers.

...