23.06.15 _ 1주차

딥러닝 논문 요약 및 구현 스터디

발표자

이정수 (퍼실리)

ResNet

Deep Residual Learning for Image Recognition

소개

ResNet의 등장 배경

: AlexNet(8층) -> VGGNet(19층) -> GoogLeNet(22층) 을 통해

신경망 층수가 많을수록 이미지에서 좋은 특징 추출이 가능해짐을 확인

하지만 층수를 지나치게 늘리면

- Overfitting(과적합)이 발생 -> Dropout, L2 규제, 배치 정규화 등을 통해 과적합 문제 해결 가능
- Vanishing Gradient(기울기 소실 문제) or Exploding Gradient(기울기 폭발 문제)가 발생 -> **Idea!**

Idea : 깊은 네트워크를 학습하기 위한 방법으로 Residual Learning(잔여학습)을 제안

좋은 논문으로 평가되는 이유 2가지

1. **당시 성능이 매우 뛰어났다.**
2. **제안한 아이디어가 적용하기 쉽다.**

방식1

Abstract

Abstract

Deeper neural networks are more difficult to train. We present a residual learning framework to ease the training of networks that are substantially deeper than those used previously. We explicitly reformulate the layers as learning residual functions with reference to the layer inputs, instead of learning unreferenced functions. We provide comprehensive empirical evidence showing that these residual networks are easier to optimize, and can gain accuracy from considerably increased depth. On the ImageNet dataset we evaluate residual nets with a depth of up to 152 layers—8× deeper than VGG nets [41] but still having lower complexity. An ensemble of these residual nets achieves 3.57% error on the ImageNet test set. This result won the 1st place on the ILSVRC 2015 classification task. We also present analysis on CIFAR-10 with 100 and 1000 layers.

The depth of representations is of central importance for many visual recognition tasks. Solely due to our extremely deep representations, we obtain a 28% relative improvement on the COCO object detection dataset. Deep residual nets are foundations of our submissions to ILSVRC & COCO 2015 competitions[1], where we also won the 1st places on the tasks of ImageNet detection, ImageNet localization, COCO detection, and COCO segmentation.

- **residual 학습 소개**
  - 더 깊은 네트워크 train을 위해 residual learning framework 제시
  - 논문을 통해 residual network가 최적화하기 쉽고, 층이 깊어도 정확도도 좋다는 증거 보여줌
  - VGG보다 8배 깊은 152 layer를 사용 & 낮은 complexity(복잡성)
  - residual Nets 앙상블로 ImageNet 테스트셋 평가시 3.57% 오차 (-> 2015년 ILSVRC 대회에서 1등)

- **Depth의 중요성**
  - depth는 많은 visual recognition task에서 매우 중요
  - 깊은 depth 덕분에 COCO object detecton에서 28%의 상대적 개선을 얻음

방식2

I. Introduction

앞선 연구들로 Network의 Depth가 매우 중요하다는걸 알 수 있었지만, 너무 깊어지면 오히려 성능이 떨어지는 문제가 발생했습니다. 가장 큰 원인은 Vanishing / Exploding Gradient 현상 때문입니다.

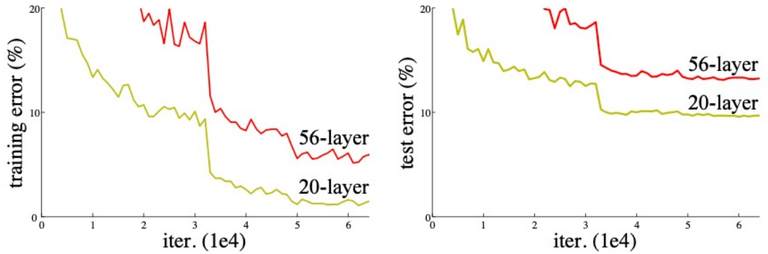
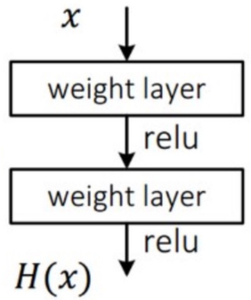


그림1

성능이 포화 수준에 도달하면 degradation(성능저하) 문제가 발생되는데,
이 원인은 Overfitting이 아니라, 깊은 모델에 layer가 추가될때 발생했습니다.
(test error만 상승한게 아니라, train error도 상승했으므로)

Deep Residual Learning Framework를 도입해 레이어가 깊어질수록 성능이 떨어지는 degradation문제를 해결했습니다.



기존 CNN 구조

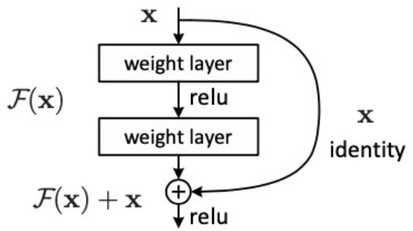


Figure 2. Residual learning: a building block.

제안한 Residual block 구조

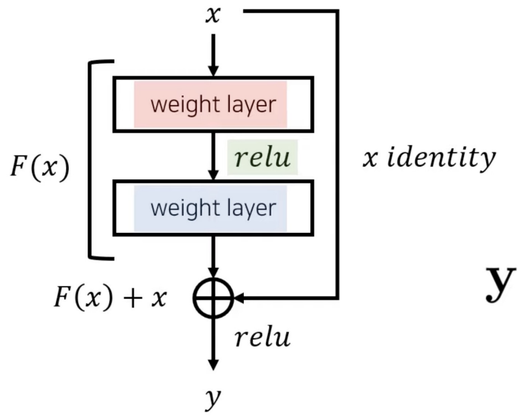
그림2

layer를 뛰어넘어 전달해 skip connection 이라 하고, 입력값 x가 그대로 전달되 identity mapping 이라 합니다.

residual block은 최적화 난이도를 낮춥니다. original function: H(x)
이때 F(x) = H(x) – x . residual 구조에서는 F(x)가 0이 되도록 즉, 입출력이 같도록(x=H(x)) 만드는걸 목표로 하므로, 깊어져도 값을 조금씩 변화시키며 성능을 올릴 수 있었습니다. 또한, H(x)가 x가 되게 학습하면 미분해도 x는 1을 갖기 때문에 최소 gradient로 1을 가지므로 gradient vanishing 문제가 해결 됩니다.

방식3

III. Deep Residual Learning



$$\mathcal{F} = W_2\sigma(W_1\mathbf{x})$$

일반적인 형태

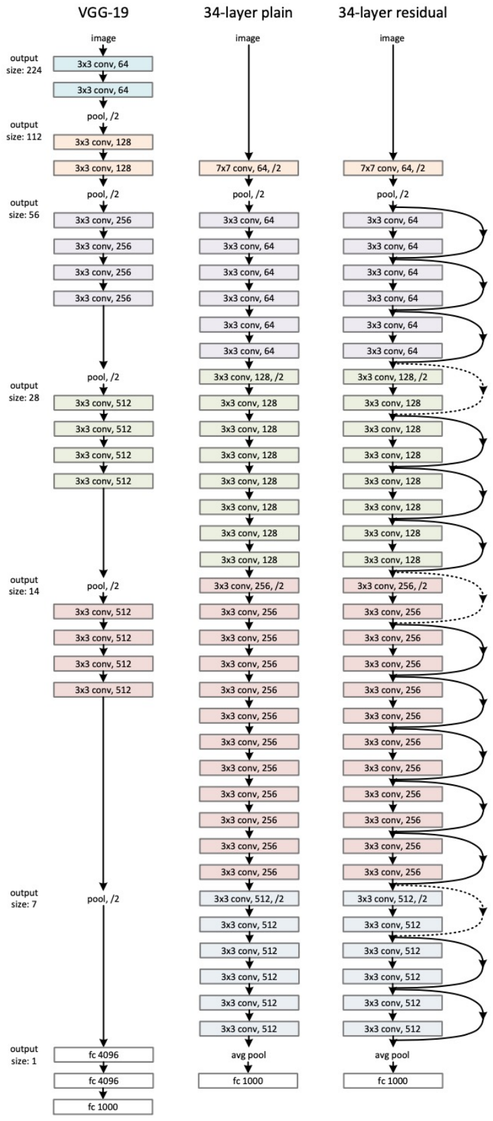
$$\mathbf{y} = \underbrace{\mathcal{F}(\mathbf{x}, \{W_i\})}_{\text{multiple convolutional layers}} + \underbrace{W_s\mathbf{x}}_{\text{shortcut}}$$

단, F(x)와 x가 그냥 더해지는것이 아닌 행렬연산 이므로 차원을 맞춰줘야 함

$\Rightarrow W_s$를 곱해서 shape을 맞춰줌

shortcut connection으로 만든 block을 identity block이라 함

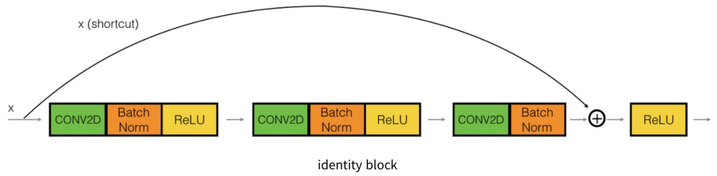
III. Deep Residual Learning

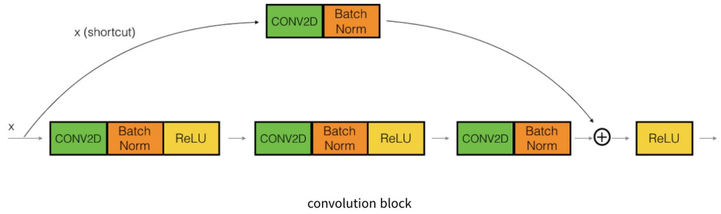


왼쪽부터 VGG / 34 layer plain / 34 layer residual 구조

VGG를 참고 -> plain 설계 -> 여기에 skip connection 추가해 residual network 설계

ResNet은 identity block과 convolution block으로 구성





- identity block은 네트워크의 output F(x)에 x를 더해주는 것
- convolution block은 F(x)에 x에 1x1 convolution연산을 한 것을 더해주는 것

III-iv. Implementation(구현)

Augmentation

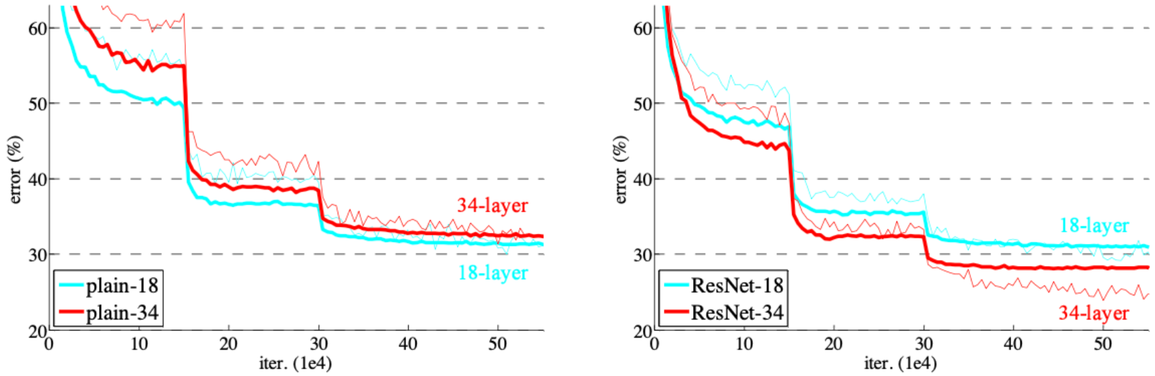
- [256, 480] 범위 random resize
- 224 * 224 random crop + Horizontal flip
- standard color augmentation
- Batch Normalization
- initialization
- Optimizer는 SGD , mini batch 크기 256
- Learning Rate는 0.1에서 시작, 정체 시 0.1배
- 60 * 10^4 iteration
- Weight decay는 0.0001 , Momentum은 0.9
- dropout 사용 X
- 10 Cross Validation 적용
- {224, 256, 384, 480, 640} 중 하나로 resize 하고 평균 score 산출

IV. Experiments

IV-i. ImageNet Classification

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

ImageNet 대회 실험에서 사용한 모델 구조들



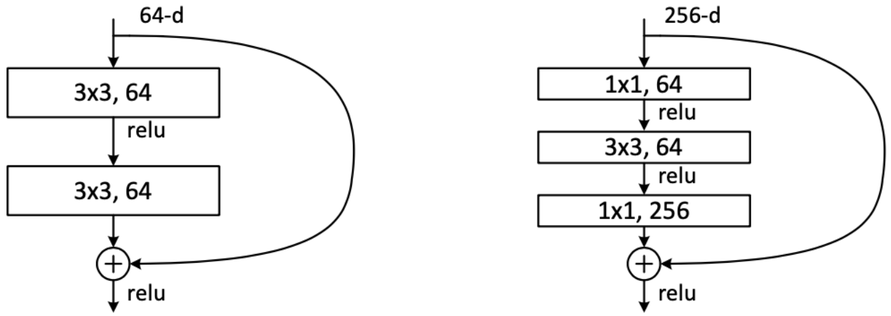
	plain	ResNet
18 layers	27.94	27.88
34 layers	28.54	**25.03**

실험을 통해 연구팀이 확인한 3가지

- degradation문제 해결 : layer가 증가해도 error 감소
- plain 대비 error 3.5% 감소
- 수렴속도 증가

IV. Experiments

IV-i. ImageNet Classification



학습시간의 문제로 소개했던 block을 **bottleneck 형태**로 수정

- 1*1, 3*3, 1*1 conv로 구성된 3 layer 구조로 변경
- 층은 증가했지만 1*1 conv 2개를 사용하므로
  파라미터 수가 감소하고 연산량이 감소
- layer가 많아져 비선형성이 더 추가됨

결론적으로, Skip Connection을 이용한 Shortcut과 Bottleneck 구조 덕에 더 많은 layer를 사용해 성능을 향상시킴

method	top-1 err.	top-5 err.
VGG [41] (ILSVRC'14)	-	8.43†
GoogLeNet [44] (ILSVRC'14)	-	7.89
VGG [41] (v5)	24.4	7.1
PReLU-net [13]	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**

Table 4. Error rates (%) of **single-model** results on the ImageNet validation set (except † reported on the test set).

method	top-5 err. (**test**)
VGG [41] (ILSVRC'14)	7.32
GoogLeNet [44] (ILSVRC'14)	6.66
VGG [41] (v5)	6.8
PReLU-net [13]	4.94
BN-inception [16]	4.82
ResNet (ILSVRC'15)	**3.57**

Table 5. Error rates (%) of **ensembles**. The top-5 error is on the test set of ImageNet and reported by the test server.

⇒Layer가 늘어나도 degradation문제없이 타 모델 대비 정확도 높음

IV. Experiments

IV-ii. CIFAR-10 and Analysis

method			error (%)
Maxout [10]			9.38
NIN [25]			8.81
DSN [24]			8.22
	# layers	# params	
FitNet [35]	19	2.5M	8.39
Highway [42, 43]	19	2.3M	7.54 (7.72±0.16)
Highway [42, 43]	32	1.25M	8.80
ResNet	20	0.27M	8.75
ResNet	32	0.46M	7.51
ResNet	44	0.66M	7.17
ResNet	56	0.85M	6.97
ResNet	110	1.7M	**6.43** (6.61±0.16)
ResNet	1202	19.4M	7.93

Table 6. Classification error on the **CIFAR-10** test set. All methods are with data augmentation. For ResNet-110, we run it 5 times and show "best (mean±std)" as in [43].

CIFAR-10 데이터에 대한 실험 정리

이 때 1000개 이상의 layer도 구성해봤는데 성능이 낮아졌는데, train 성능과 비슷해 과대적합으로 추측

본 논문에서는 다양한 regularization(규제) 적용 안해서 그런것이라 판단. 추후 적용하여 문제를 해결할 예정