



2024 D&A

CNN 심화 (2)

Deep Session 6차시



CONTENTS.

01. Review

- LeNet
- VGG
- AlexNet

02. 기초 모델 아키텍처

- ResNet

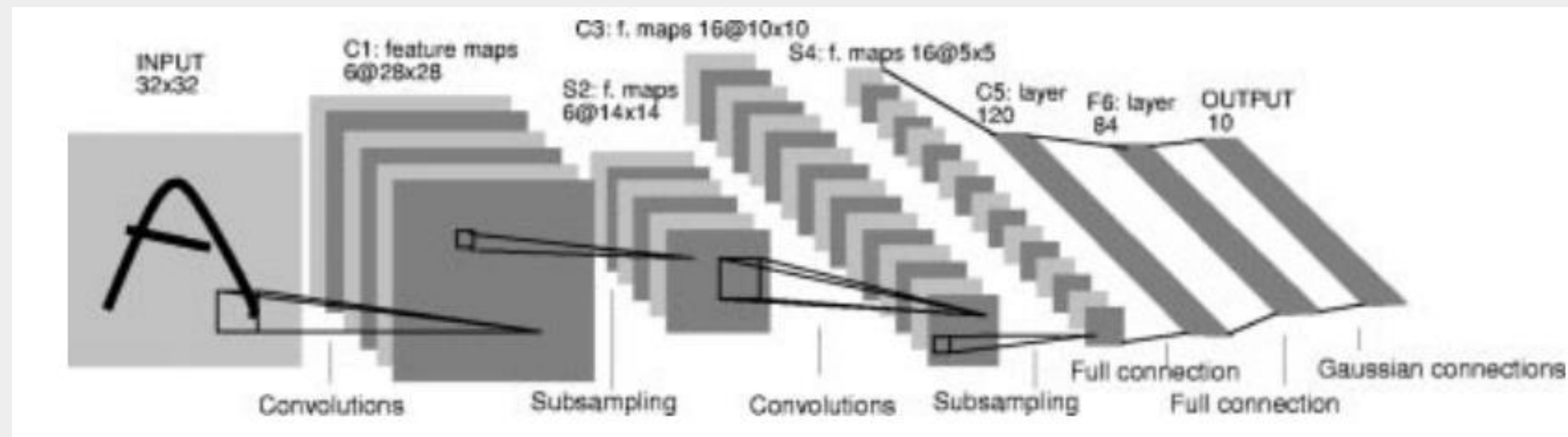
03. 논문 리뷰

- GoogLeNet

01. Review

LeNet

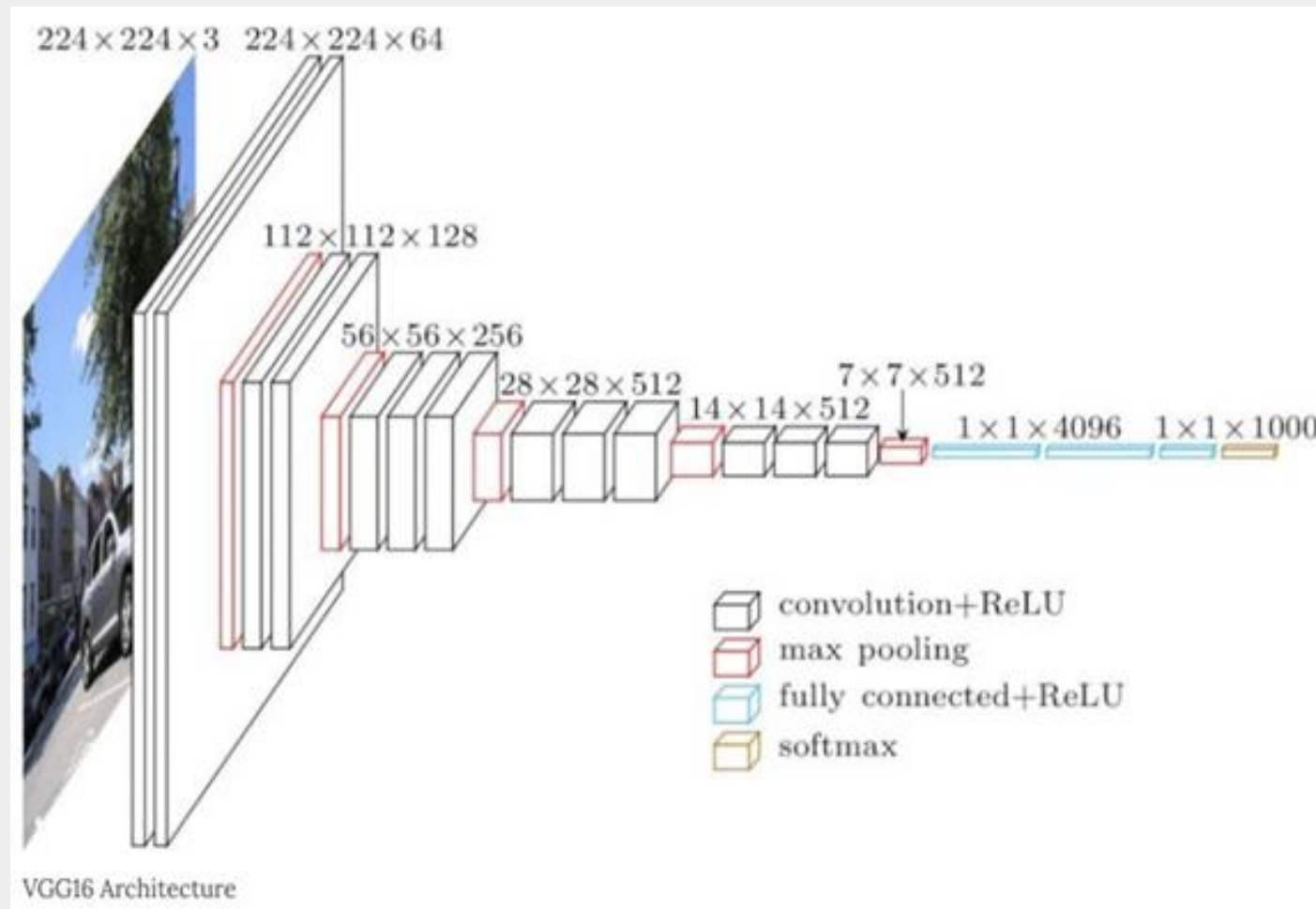
- 초기의 컨볼루션 신경망 중 하나로, 주로 손으로 쓴 숫자 인식에 사용되었던 모델
- 32 x 32 크기의 흑백 이미지에서 학습된 7 layer CNN (Input 층 제외)
 - 120x1x1
120개의 stride=1, 5x5 필터 사용
- Input → Conv (C1) → Subsampling (S2) → Conv (C3) → Subsampling (S4) → Conv (C5) → FC6 → FC7 (Output)
 - 1x32x32 6개의 5x5 필터 사용 stride=2, 2x2 평균 풀링 16개의 5x5 필터 사용 stride=2, 2x2 평균 풀링 tanh 활성화 함수 RBF 활성화 함수
 - 6x28x28 6x14x14 16x10x10 16x5x5 84 10
- 깊은 학습 네트워크로 발전하는 데 큰 발판



01. Review

VGG

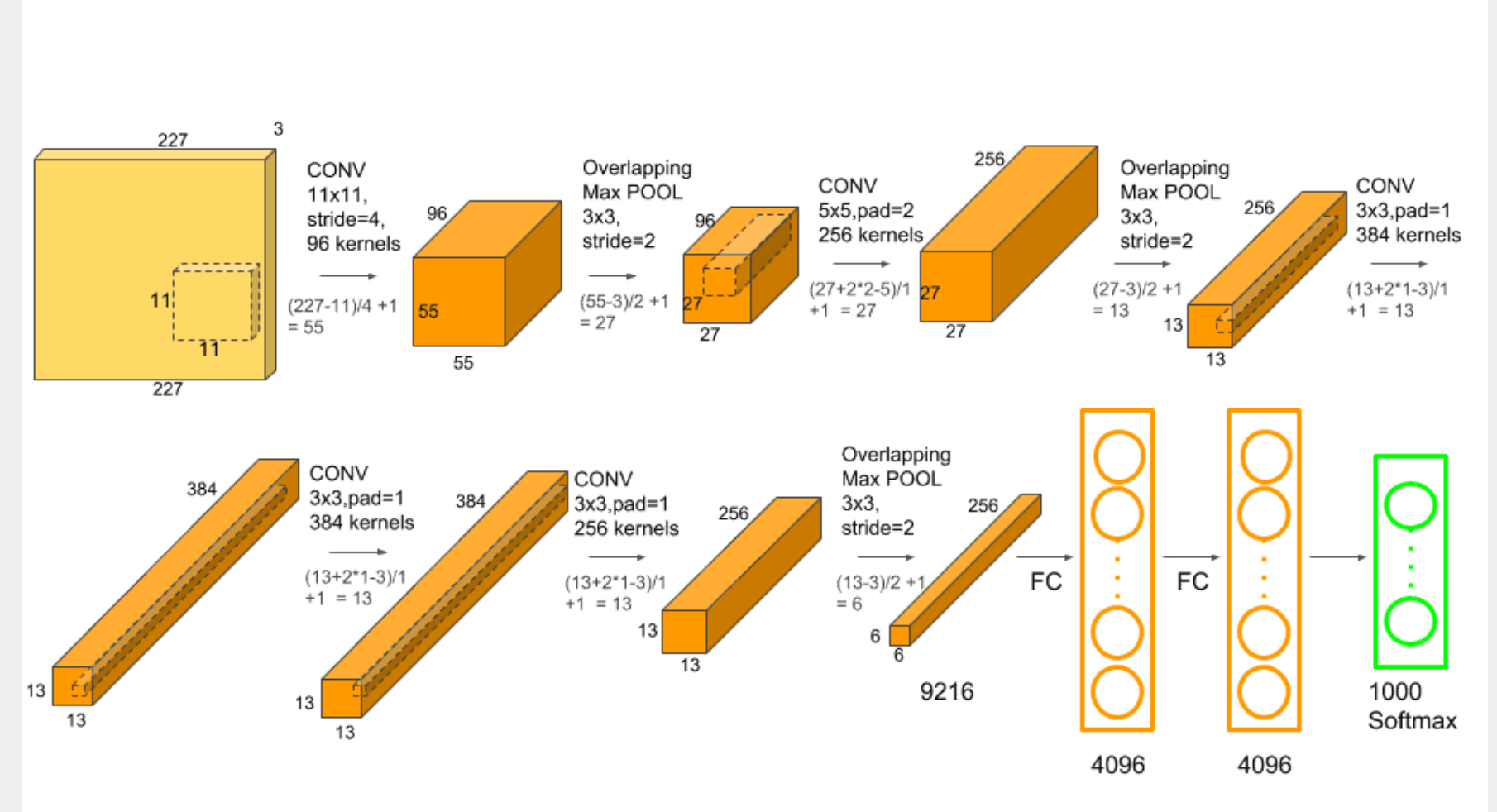
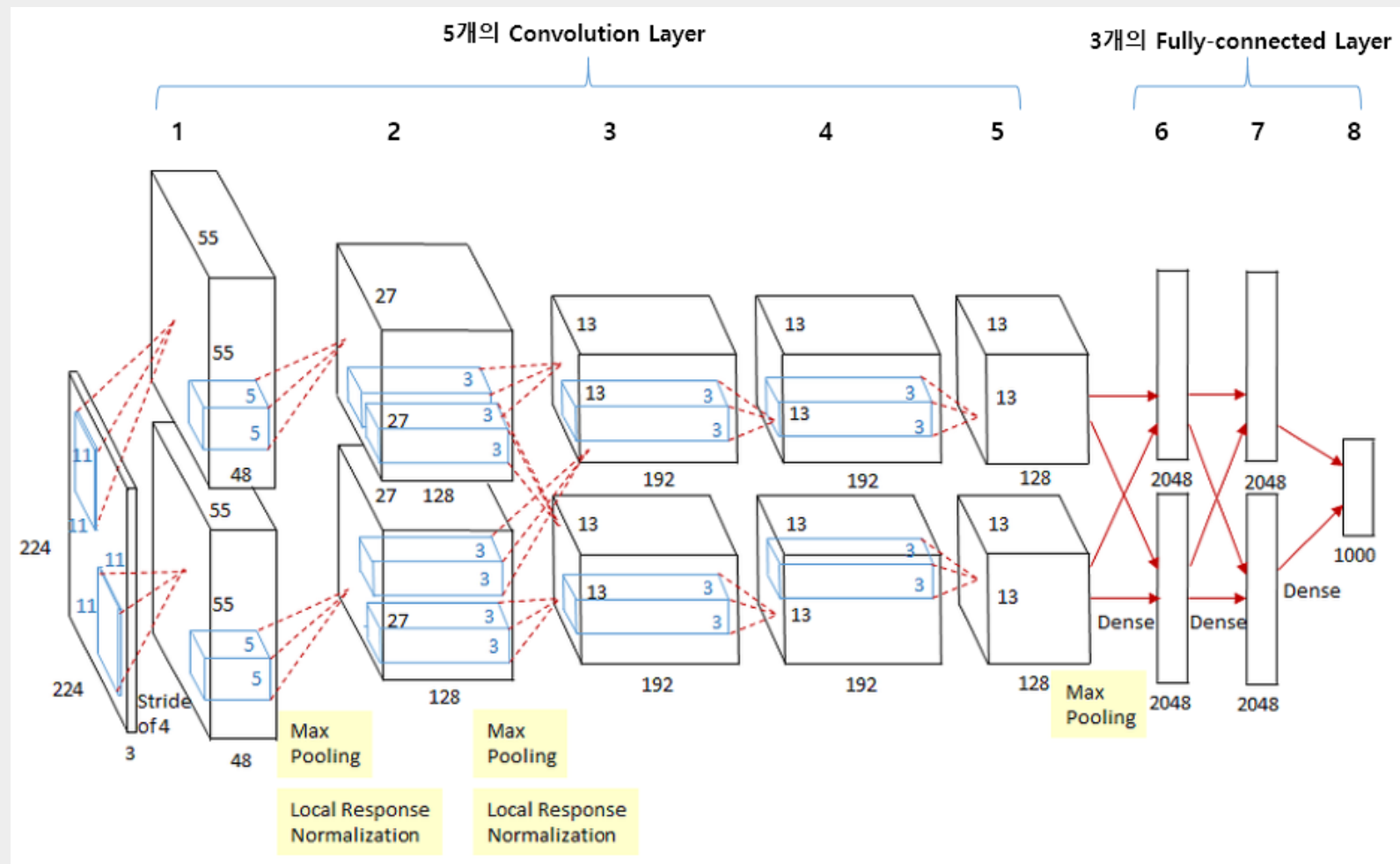
- 매우 깊고 단순한 네트워크로 더 높은 성능
- 3x3 필터 사용 + zero-padding 사용하여 더 깊게, 더 좋은 표현력
- 매우 많은 수의 파라미터, 계산 비용과 메모리 사용량이 큰 단점



- Input → C1 → C2 → MaxPool2
64개의 stride=1, padding=1, 3x3 필터 / stride=2, 2x2 최대 풀링
- C3 → C4 → MaxPool4
128개의 stride=1, padding=1, 3x3 필터 / stride=2, 2x2 최대 풀링
- C5 → C6 → C7 → MaxPool7
256개의 stride=1, padding=1, 3x3 필터 / stride=2, 2x2 최대 풀링
- C8 → C9 → C10 → MaxPool10
512개의 stride=1, padding=1, 3x3 필터 / stride=2, 2x2 최대 풀링
- C11 → C12 → C13 → MaxPool13
512개의 stride=1, padding=1, 3x3 필터 / stride=2, 2x2 최대 풀링
- FC14 → FC15 → FC16(Output)
ReLU 활성화 함수 / Softmax 함수

01. Review AlexNet

- 2개의 GPU로 병렬연산 수행을 위한 병렬적인 구조 + 대규모의 깊은 8층 구조의 CNN 모델
- Data Augmentation + Overlapping Max Pool + LRN → 강력한 피쳐 추출과 일반화 능력 향상
- Input → C1 → MaxPool1 → C2 → MaxPool2 → C3 → C4 → C5 → MaxPool5 → FC6 → FC7 → FC8 (Output)
- Input → C1 → MaxPool1 → C2 → MaxPool2 → C3 → C4 → C5 → MaxPool5 → FC6 → FC7 →



02. 기초 모델 아키텍처

ResNet

Intro

- 깊은 신경망을 학습할 때 발생할 수 있는 문제 완화
- Residual Network의 줄임말
- VGG 보다 8배 깊은 152개의 layer 사용, 더 적은 복잡도
- 3.57%의 error
- ILSVRC 2015 등 다양한 분야 대회에서 1위 차지

02. 기초 모델 아키텍처

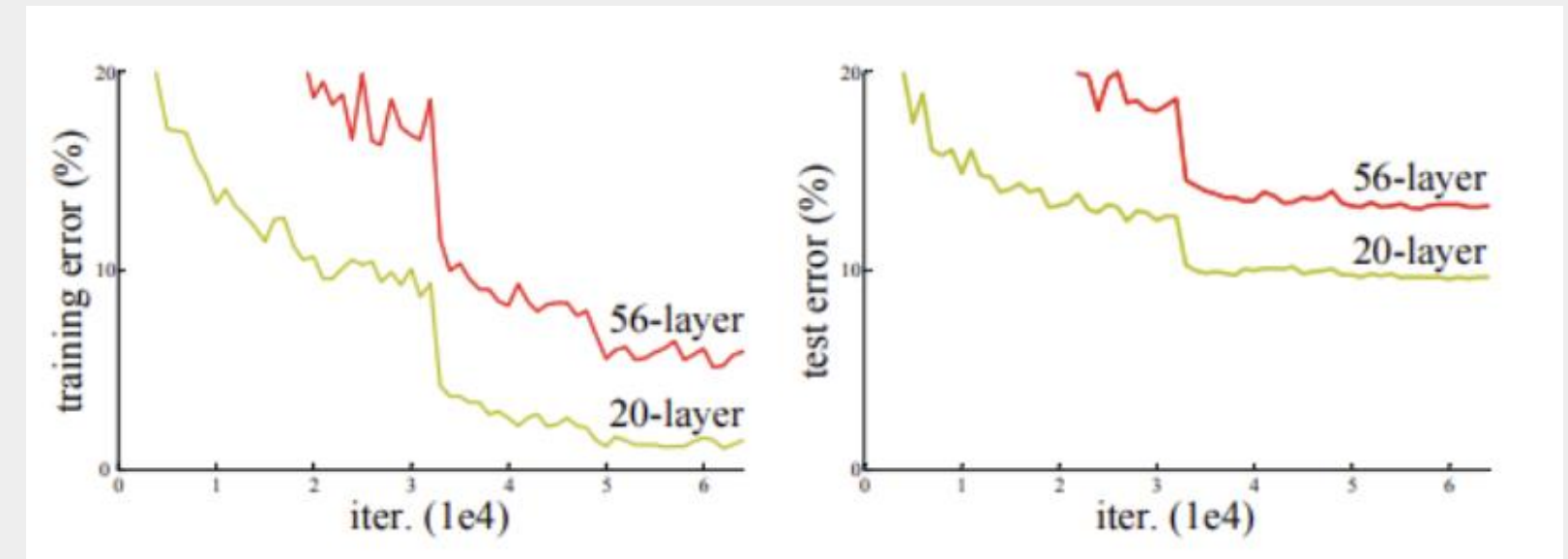
ResNet

Intro

깊이가 깊어질수록 네트워크의 성능은 향상될까?

- 일정 범위 이상의 layer가 증가할수록 오히려 더 높은 error 발생
 - vanishing/exploding gradients, overfitting 등의 문제
 - 연산이 진행되면서 원래 정보를 잃게 됨

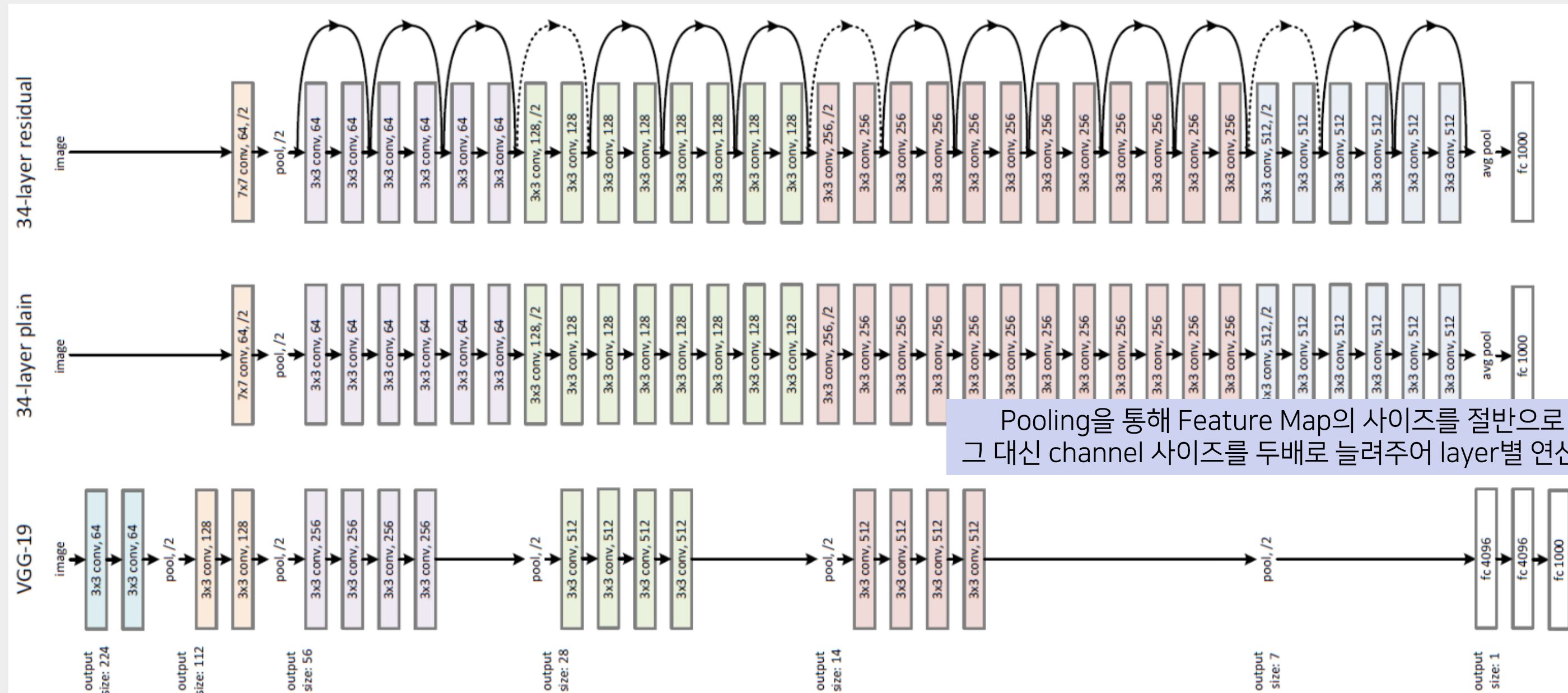
- layer를 깊게 쌓되, 학습하는 양을 줄이자!
- layer를 거친 값(output)에 x(input)를 더해줘 기존 정보를 살리자!



02. 기초 모델 아키텍처

ResNet

구조



Pooling을 통해 Feature Map의 사이즈를 절반으로 줄어둡
그 대신 channel 사이즈를 두배로 늘려주어 layer별 연산량을 유지

02. 기초 모델 아키텍처

ResNet

주요 특징

1. residual learning
 2. shortcut(skip) connection
- vanishing gradients 문제 완화
 - 깊은 네트워크에서도 안정적인 학습 가능
 - 원래의 입력 정보를 보존하여 전체 네트워크의 학습 능력을 향상

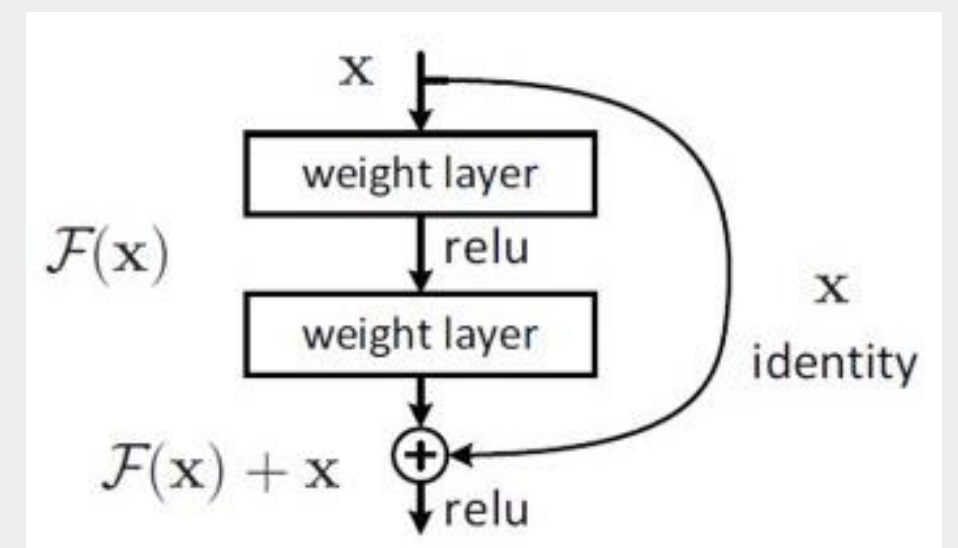
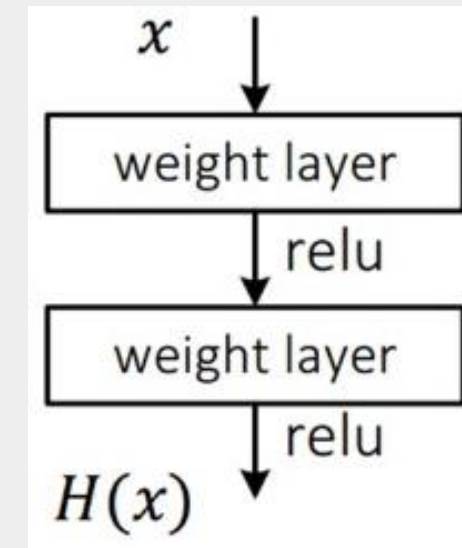
02. 기초 모델 아키텍처

ResNet

1. residual learning (잔차 학습)

- 기존의 경우
 - input을 x 로 받아 2개의 weight layer를 거친 결과값
 - 학습을 통해 **최적의 $H(x)$** 를 구해내는 것이 목표 → 어려움ex. 기존 VGG에서 3x3 conv를 통해 얻는 값
- 논문의 경우
 - 기존 정보(x)에 새로 추가된 레이어를 학습한 값($F(x)$)을 더한 값
 - **최적의 $F(x)$** 를 구해내는 것이 목표 → 상대적으로 쉬움

기존 정보(x)에
새로운 레이어를 추가적으로 누적해서 다시 학습시킴



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

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

기존 정보(x)는 학습하지 않고,
새로 추가된 레이어에 대한 부분만 학습시킴

02. 기초 모델 아키텍처

ResNet

1. residual learning (잔차 학습)

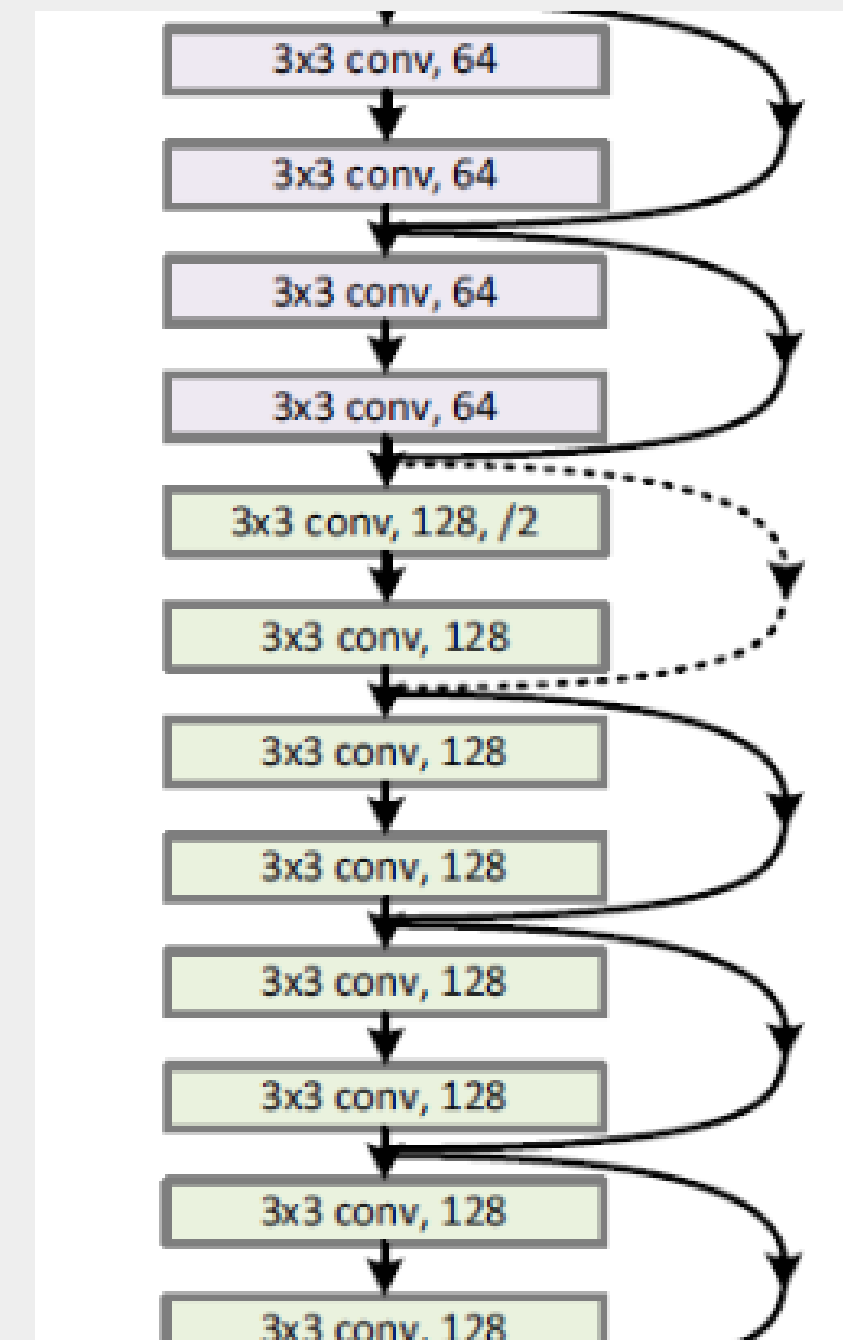
- $F(x) = H(x) - x$ (예측값 - 실제값 = 잔차)
- 잔차 학습 = 입력 x 에서 수정만을 추가로 학습 \rightarrow '보정' 역할
- $H(x)$ 를 맞추는 것보다 $F(x)$ 가 거의 0이 되는 방향으로 맞추는 것이 훨씬 쉬움
- 차이에 대한 학습만 하기 때문에 훨씬 연산량 감소

02. 기초 모델 아키텍처

ResNet

2. shortcut(skip) connection (단축 연결)

- 'Gradient가 잘 도달할 수 있게 지름길을 뚫어주자'
- 입력 x 를 여러 개의 층을 거치지 않고 바로 출력층에 더하는 방식
- 추가적인 매개변수 $X \rightarrow$ 연산량과 복잡도 낮음



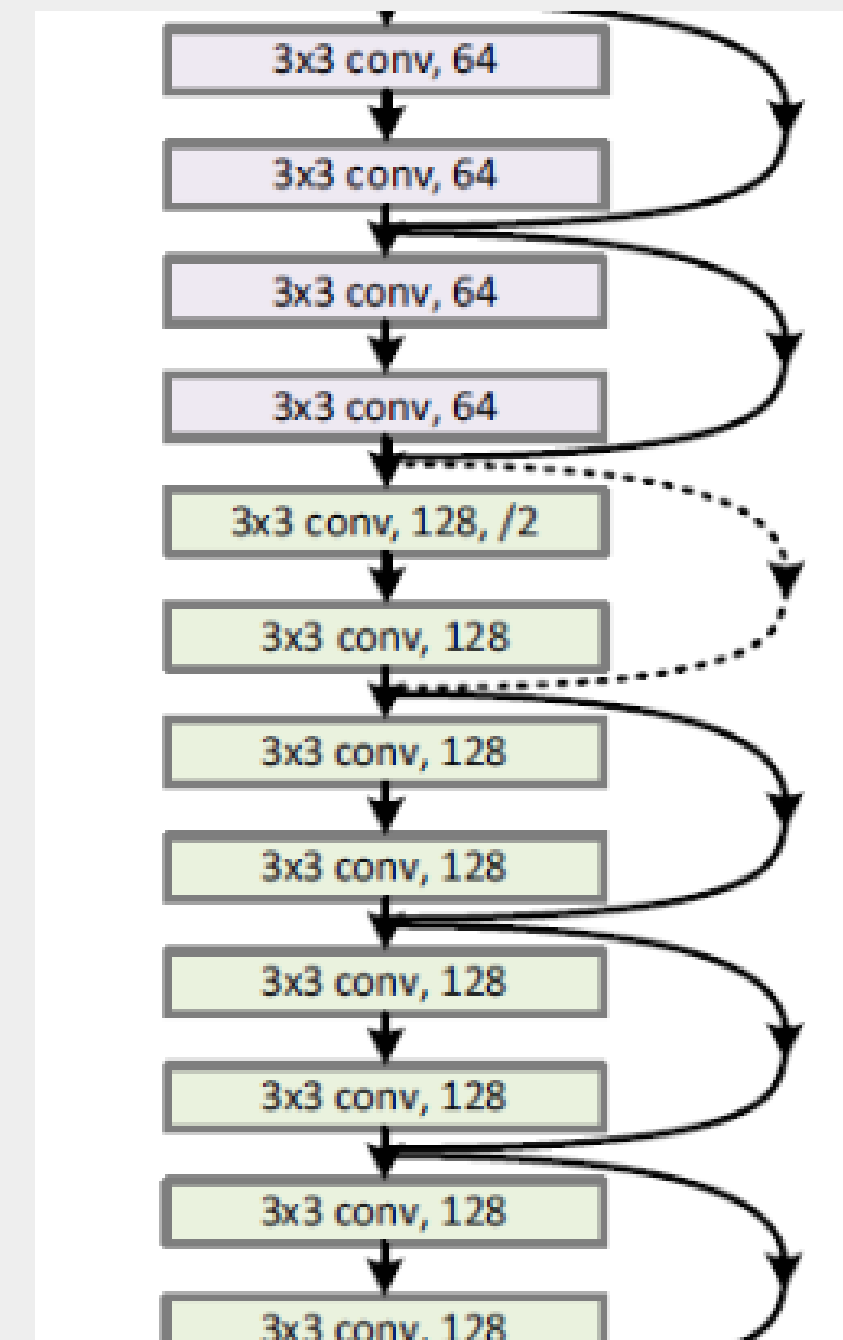
02. 기초 모델 아키텍처

ResNet

2. shortcut(skip) connection (단축 연결)

구현 방식

- Identity Shortcut (일반적인 구조) - 실선
: 입력과 출력의 차원이 같을 때 사용
입력을 변경 없이 직접 출력에 더함
- Projection Shortcut - 점선
: 입력과 출력의 차원이 같지 않을 때 사용
입력을 선형 변환(ex. 1x1 conv)을 통해 출력의 차원과 일치시킨 후 출력에 더함



02. 기초 모델 아키텍처

ResNet

Outro

- layer가 50개 이상이 될 때 1x1 conv를 통해 연산량을 줄여줌 (깊을수록 1x1 conv 효과 명확)

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				
		$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 64 \\ 3 \times 3, 64 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
conv3_x	28×28	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 128 \\ 3 \times 3, 128 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 8$
conv4_x	14×14	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 256 \\ 3 \times 3, 256 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 23$	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 36$
conv5_x	7×7	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 2$	$\begin{bmatrix} 3 \times 3, 512 \\ 3 \times 3, 512 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$
	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

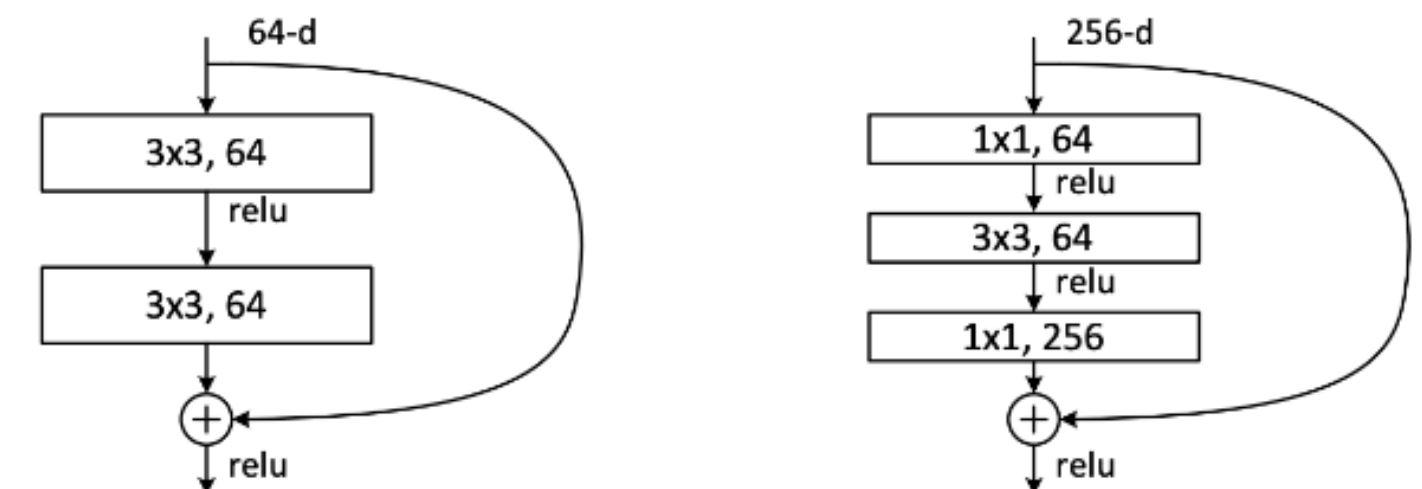
Bottleneck Architecture

1x1 conv: 차원 축소 → 중요한 피쳐 보존

3x3 conv: 줄어든 채널을 3x3 conv 연산 → Receptive Field 확장

1x1 conv: 원래대로 차원 맞춰줌 → 중요한 피쳐 보존

➔ 모델을 깊게 쌓으면서 동시에 연산량을 줄이는 것이 핵심 전략



left. 일반적인 구조의 ResNet: $(3 * 3 * 64) * 2 = 1,152$

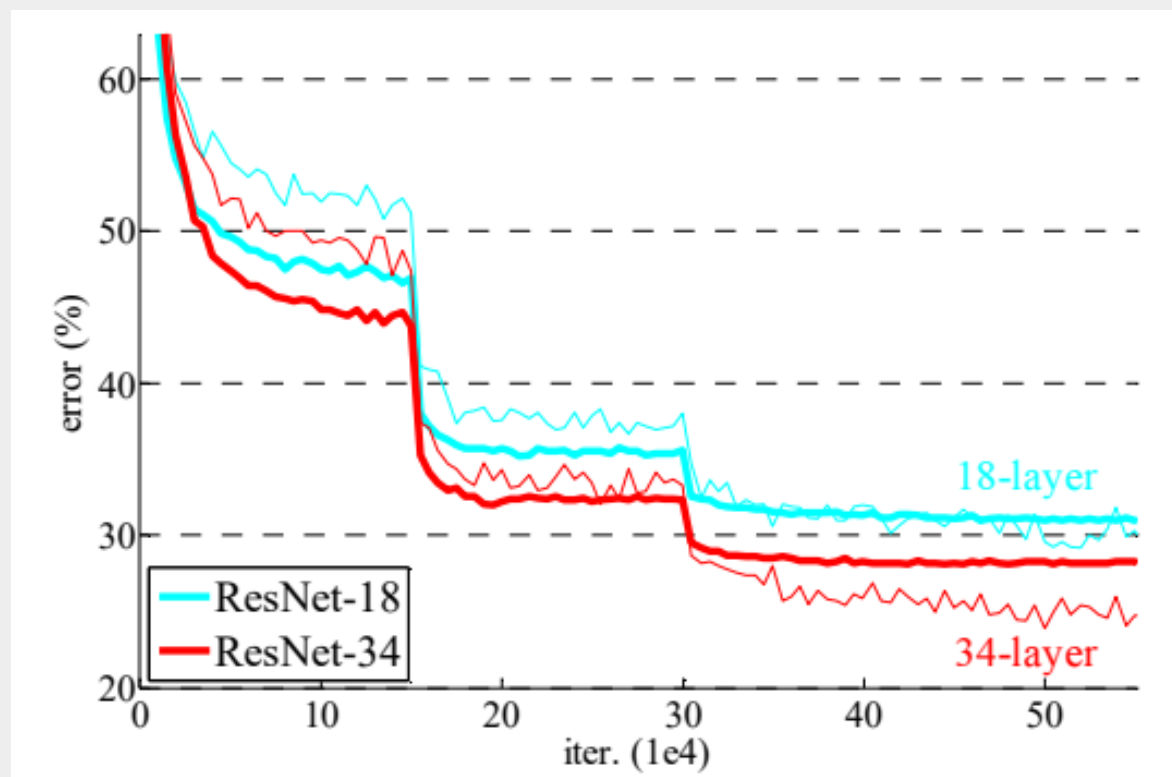
right. Bottle Neck ResNet: $(1 * 1 * 64) + (3 * 3 * 64) + (1 * 1 * 256) = 896$

02. 기초 모델 아키텍처

ResNet

Intro

- 18 layer보다 더 깊은 34 layer에서 더 낮은 에러율
- 18 layer에서는 plain과 ResNet의 에러율 비슷하나, ResNet의 학습 파라미터 수가 더 적어 학습 속도가 더 빠름



	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.

02. 기초 모델 아키텍처

ResNet

정리

- 핵심: Residual Learning & Shortcut Connection
 - 기존 Neural Network보다 깊이가 깊어져도 학습 및 최적화가 쉬움
 - 기존 Neural Network보다 깊이가 깊어져도 파라미터 수는 더 적음
 - ResNet의 깊이가 깊어져도 성능 꾸준히 상승
- ➔ 컴퓨터 비전 분야에서 매우 중요한 발전을 이루었으며, 다양한 분야에서 널리 사용
- ➔ 딥 러닝 모델의 설계에 많은 영감, 후속 모델에 영향

GoogLeNet

- [0. Abstract](#)
- [1. Introduction](#)
- [2. Related Work](#)
- [3. Motivation and High Level Considerations](#)
- [4. Architecture Details](#)
- [5. GoogLeNet](#)
- [6. Training Methodology](#)
- [7. ILSVRC 2014 Classification Challenge Setup and Results](#)
- [8. ILSVRC 2014 Detection Challenge Setup and Results](#)
- [9. Conclusions](#)

과제

ResNet 논문 리뷰

<https://arxiv.org/pdf/1512.03385.pdf>



2024 D&A

THANK YOU

2024.05.02

Deep Session 6차시

