Image Classification
(AlexNet, VGG, GoogleNet, ResNet)

이철희

AlexNet

AlexNet

- 8 layers network
- 1st place on ILSVRC 2012 classification task
- CNN의 부흥에 아주 큰 역할을 한 구조

AlexNet

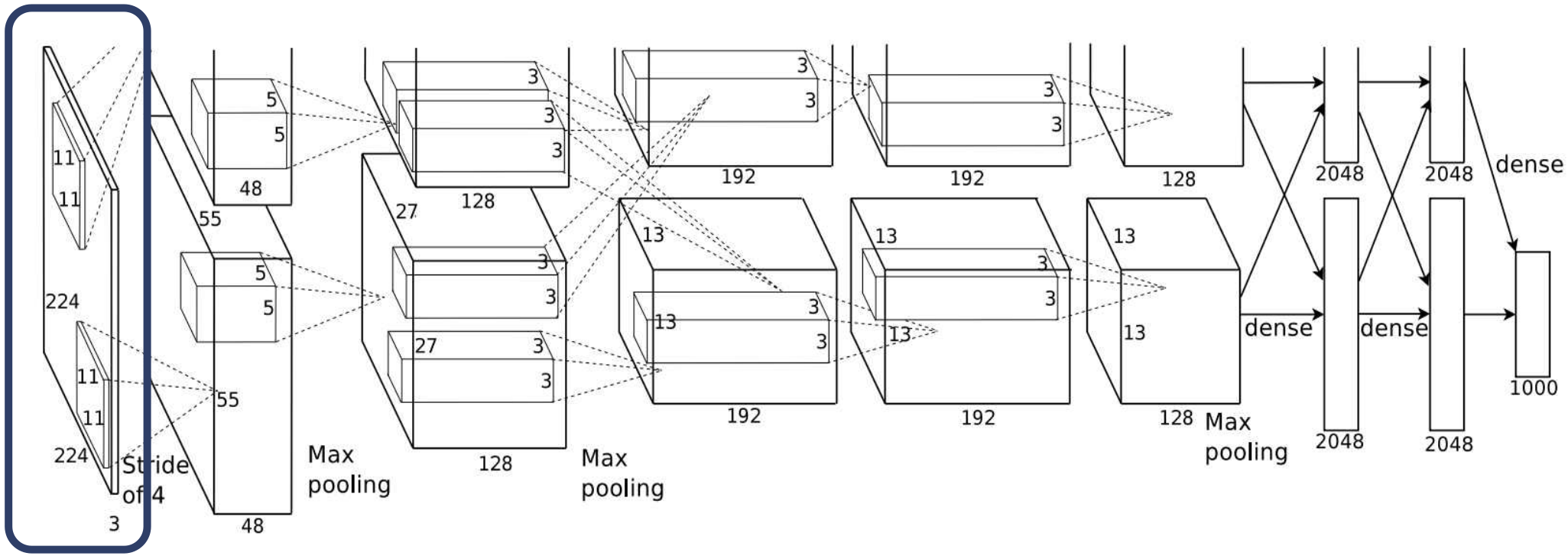
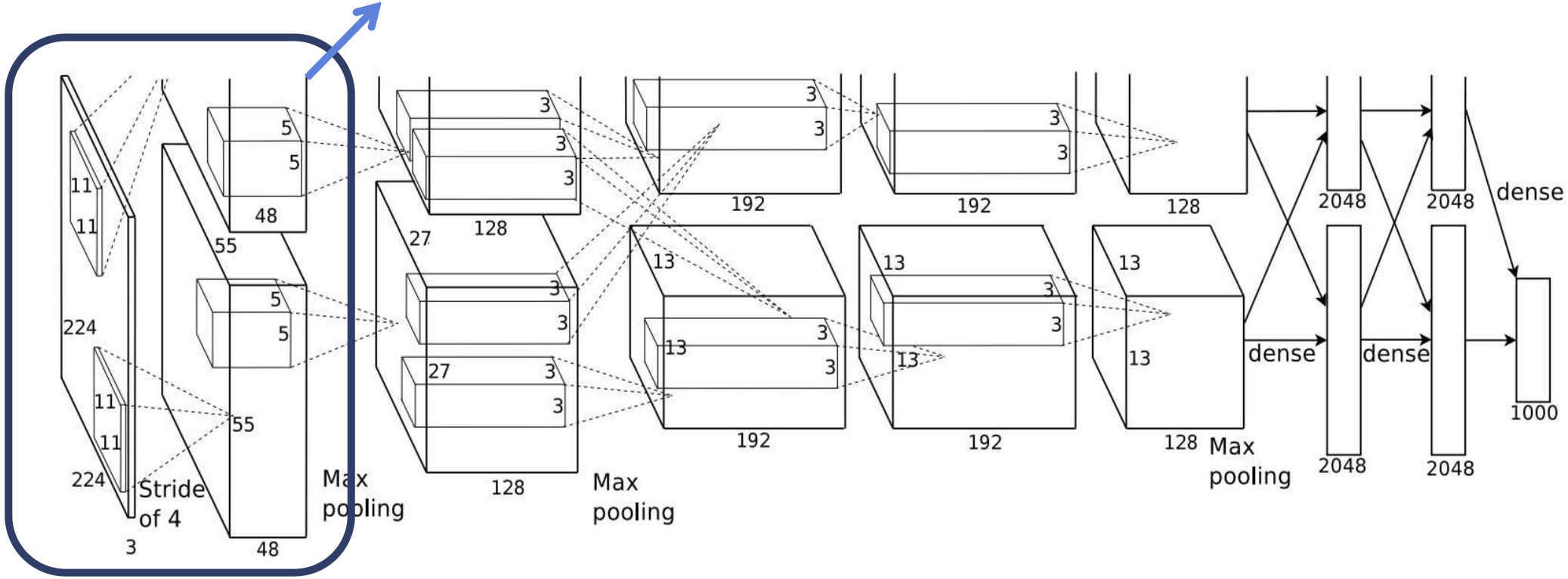


Image size : 227 x 227 x3

AlexNet

Why are layers Divided into two parts?



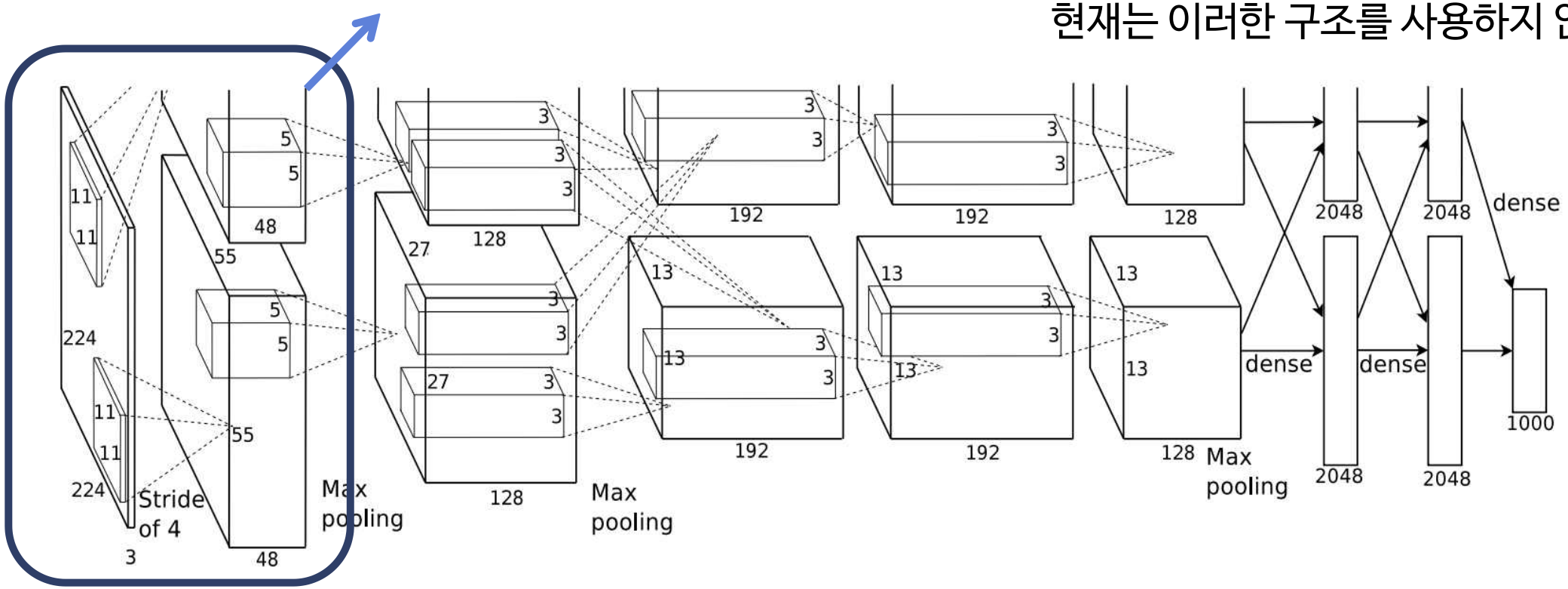
Conv1

- 96개의 (11 x 11 x 3) filters
- stride size : 4
- padding size : 0

AlexNet

Why are layers Divided into two parts? =〉 Poor gpu performance(2012)

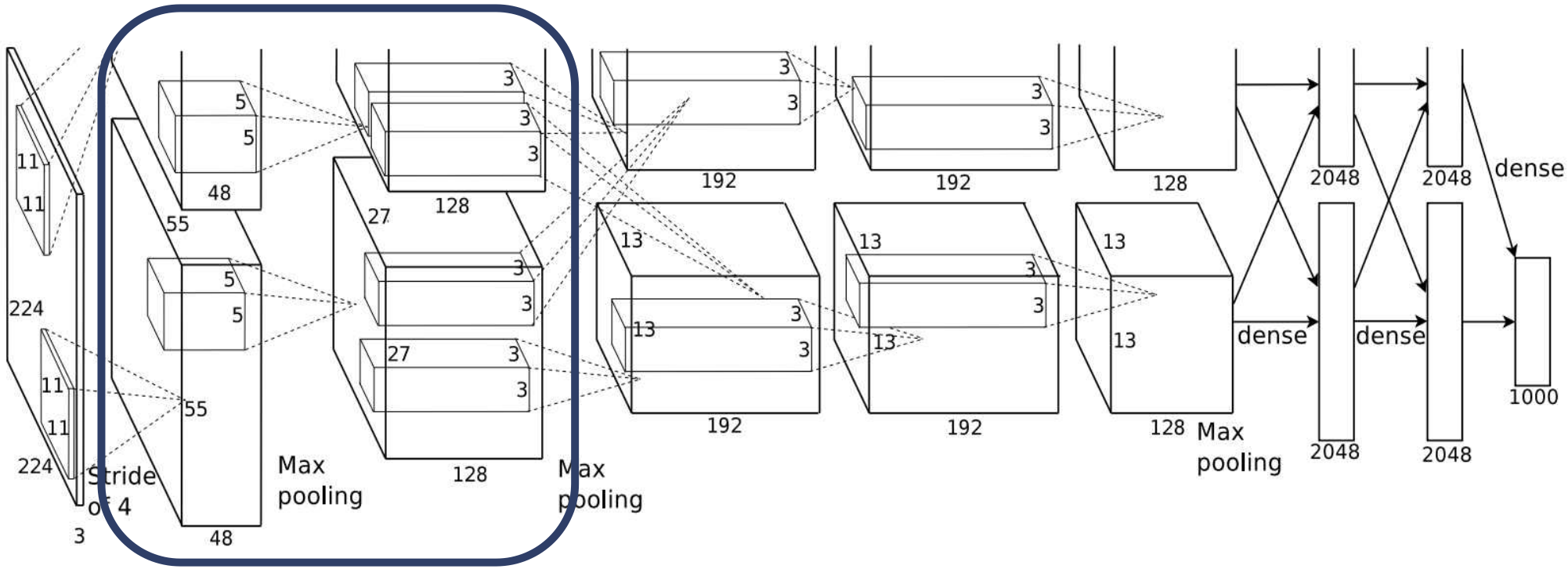
현재는 이러한 구조를 사용하지 않음!!



Conv1

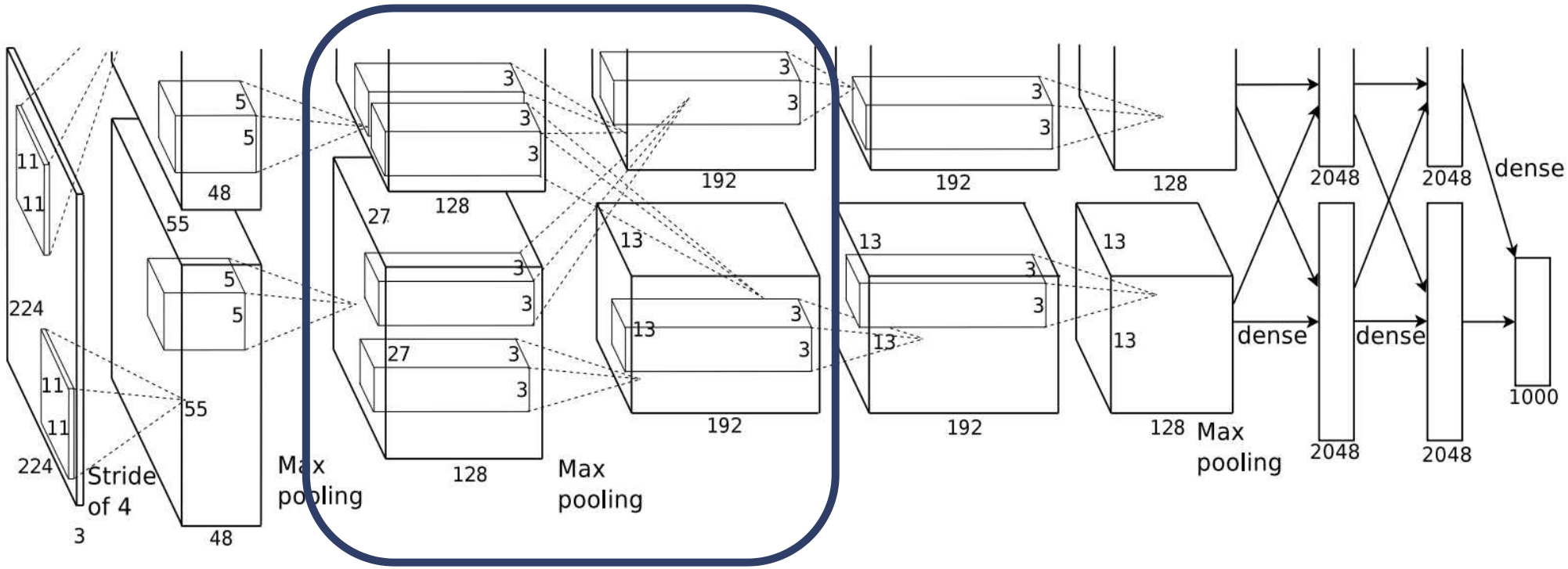
- 96개의 (11 x 11 x 3) filters
- stride size : 4
- padding size : 0

AlexNet



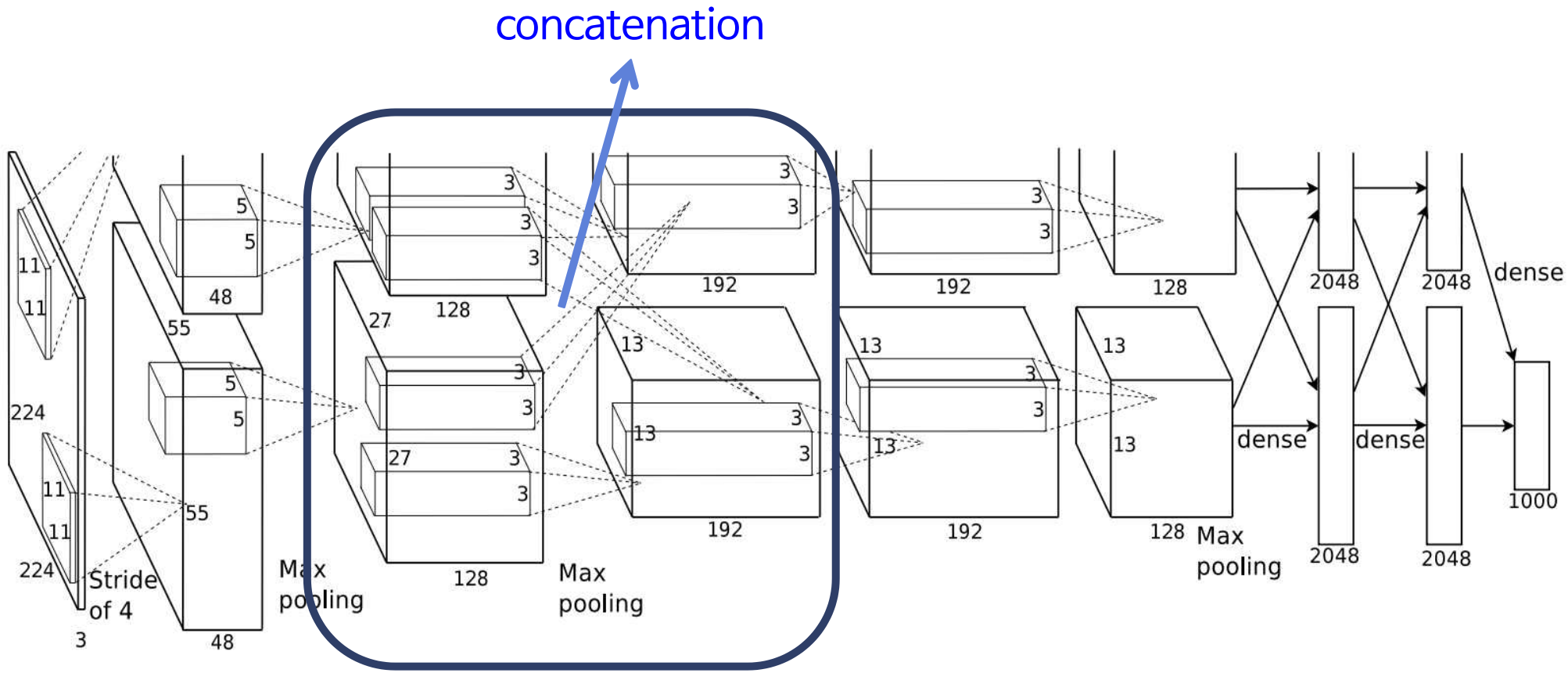
Conv2
- 256개의 (5 x 5 x 48) filters

AlexNet



Conv3 : 384개의 (3 x 3 x 256) filters
Why 256 Size?

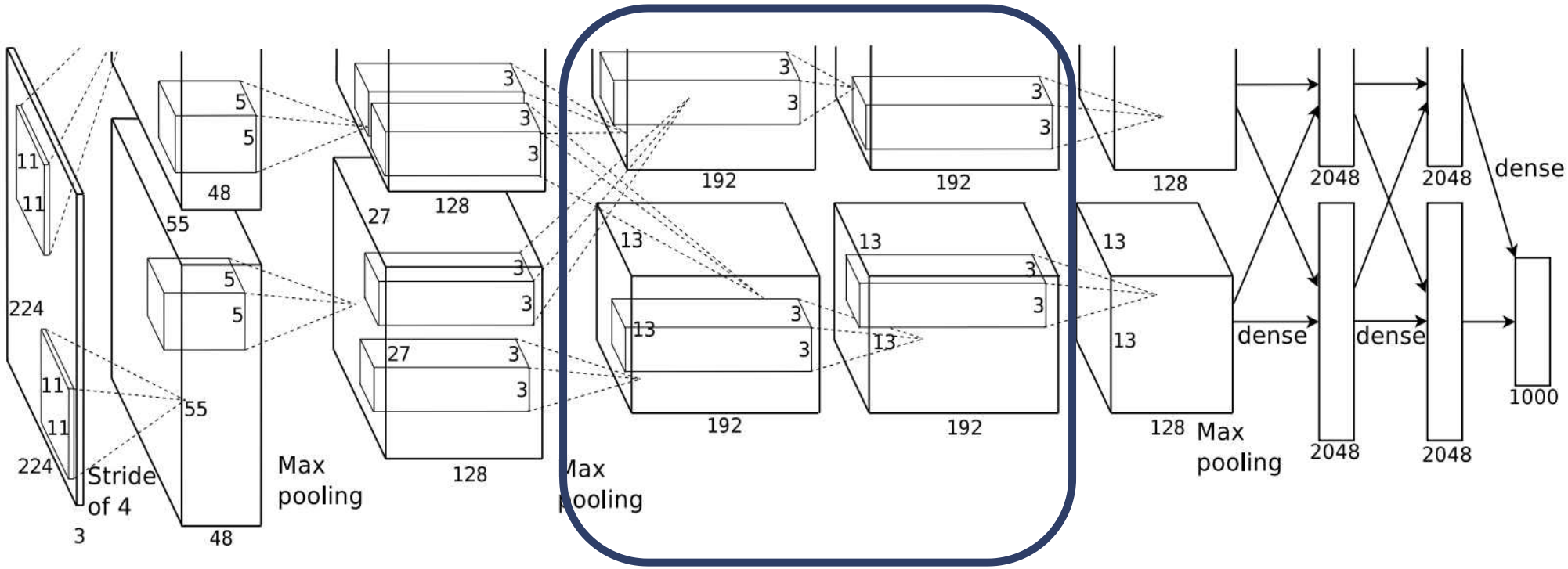
AlexNet



Conv3 : 384개의 (3 x 3 x 256(128+128)) filters

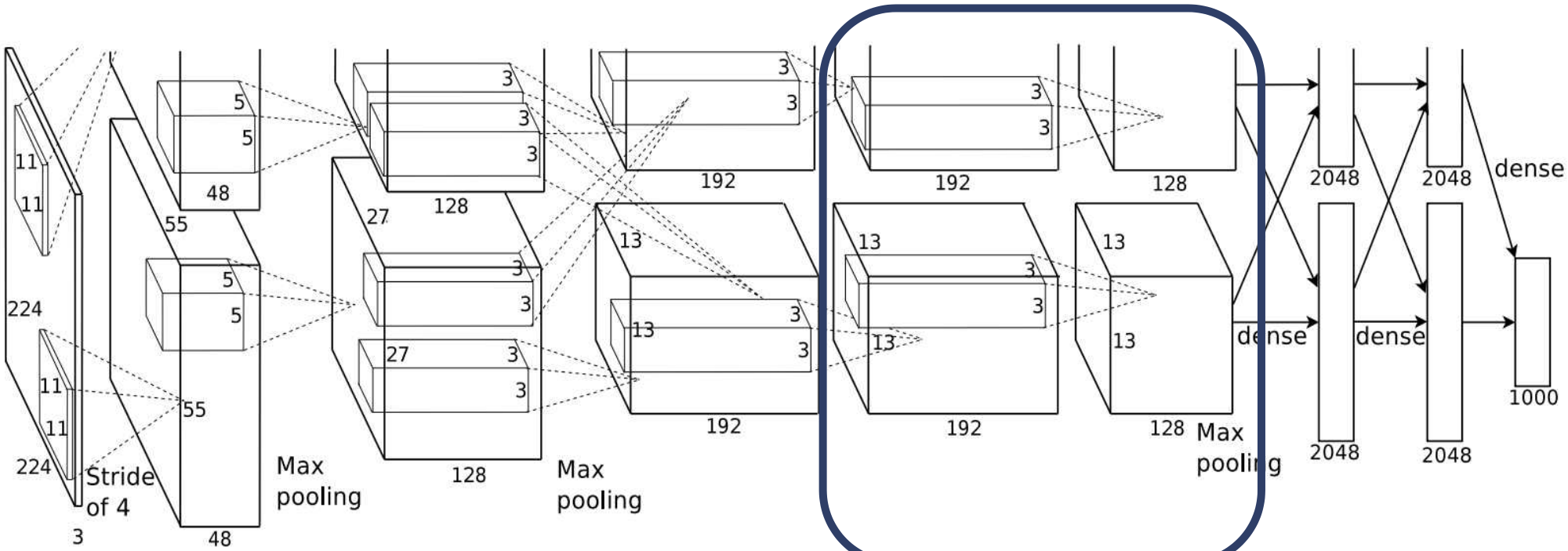
Two Layer Concatenation(128 +128)

AlexNet



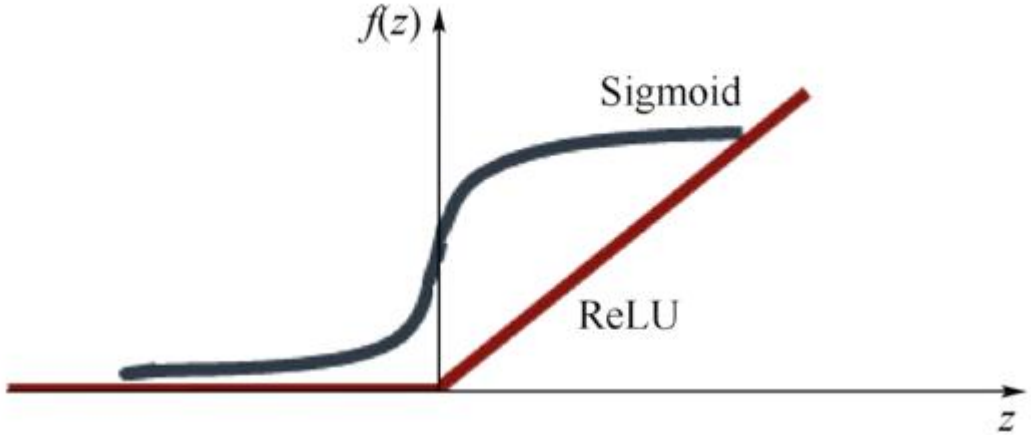
Conv4 : 384개의 (3 x 3 x 192) filters

AlexNet



Conv5 : 256개의 (3 x 3 x 192) filters

AlexNet



Why AlexNet Good performance?

1. Using ReLU(Rectified Linear Unit)

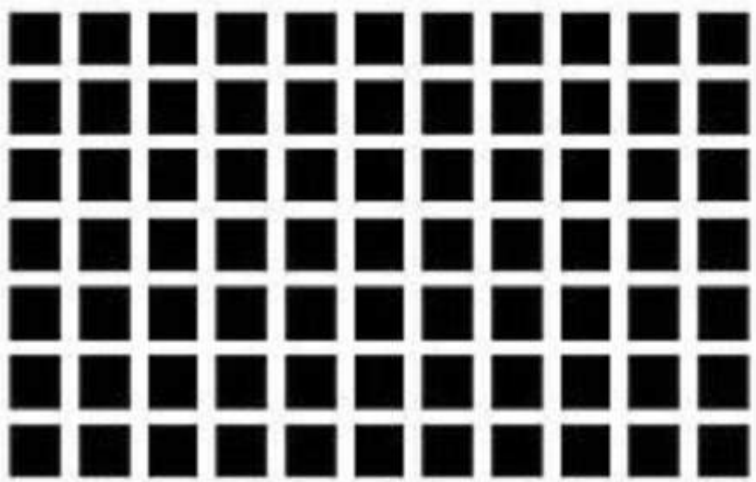
- Haholoser가 발표한 dynamical netwok에서 처음 소개
- Gradient Vanishing Problem을 해결
  + Sigmoid function은 0에서 1사이의 값을 가지기 때문에 Backpropagation을 수행할 시 layer를 지나면서 gradient가 지나면서 gradient가 0으로 수렴
  + Relu는 입력값이 0보다 작으면 0이고 크면 입력값 그대로 보냄

AlexNet

Why AlexNet Good performance?

2. LRN(Local Response Nomalization)

- Haholoser가 발표한 dynamical netwok에서 처음 소개
- lateral inhibition : 실제 뇌 세포의 증상, 강한 뉴런의 활성화가 근처 다른 뉴런의 활동을 억제시키는 현상
- 아래 그림을 보면 검은 네모들 사이에 회색 점이 보이는데 이는 검은색 사각형을 보고 뉴런들이 강하게 반응하여 흰색부분(약한 뉴런)에 반응하여 회색이 조금 보이게 됨



AlexNet

Why AlexNet Good performance?

2. LRN(Local Response Nomalization)

- lateral inhibition: Neural Network의 feature에서 한 세포가 유별나게 강한 값을 가지게 된다면, 근처의 값이 약하더라도 convolution 연산을 한다면 강한 feature를 가지게됨 => over-fitting(training dataset에 강하게 반응)
- 따라서 어떠한 filter가 있을 때, 그 주위 혹은 같은 위치에 다른 channel의 filter들을 square-sum하여 한 filter에서만 과도하게 activate하는 것을 막음

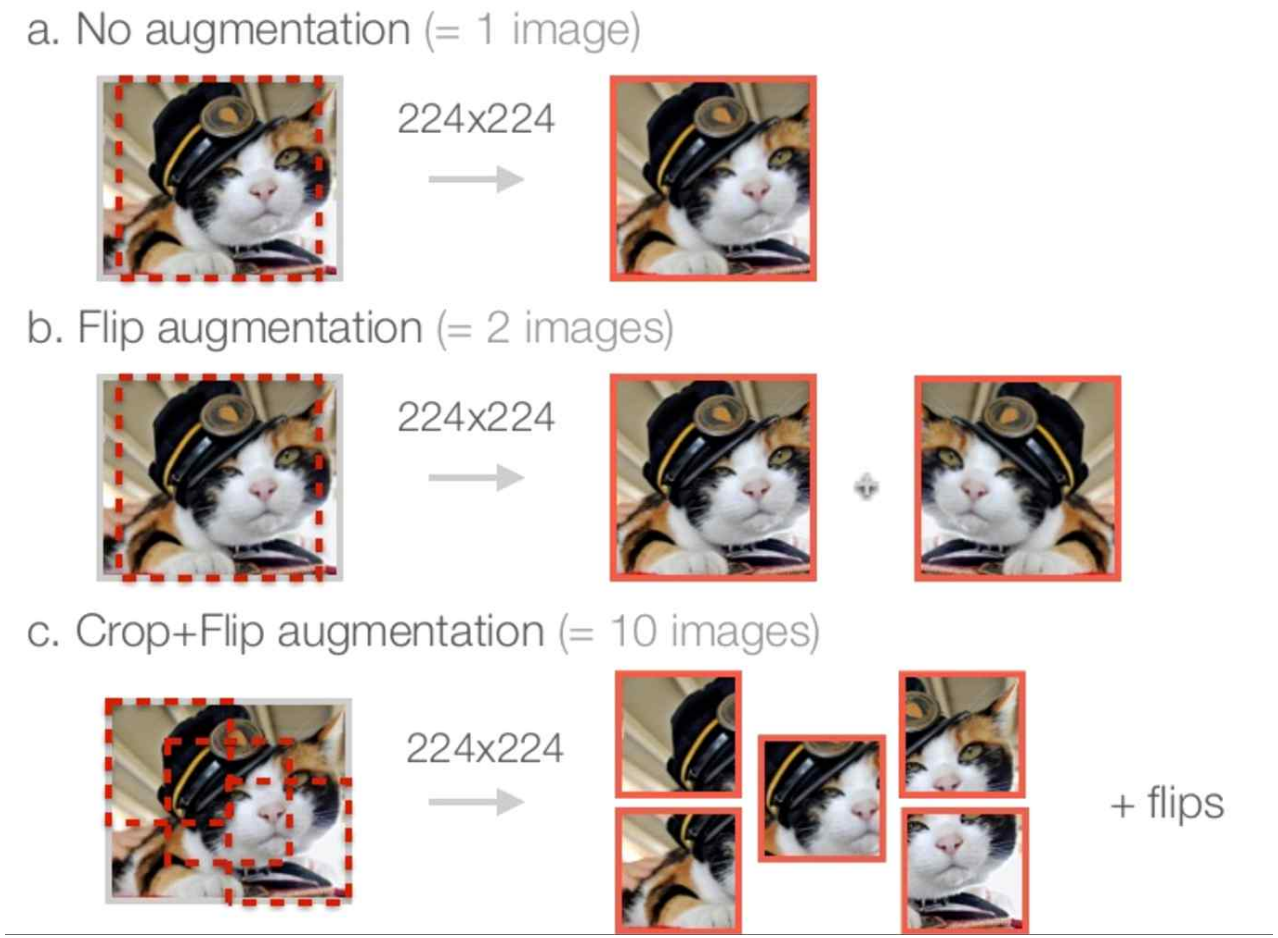
$$b^i_{x,y} = a^i_{x,y} / \left(k + \alpha \sum_{j=max(0,i-n/2)}^{min(N-1,i+n/2)} (a^j_{x,y})^2 \right)^\beta$$

[Local Response Normalization 수식]

AlexNet

Why AlexNet Good performance?

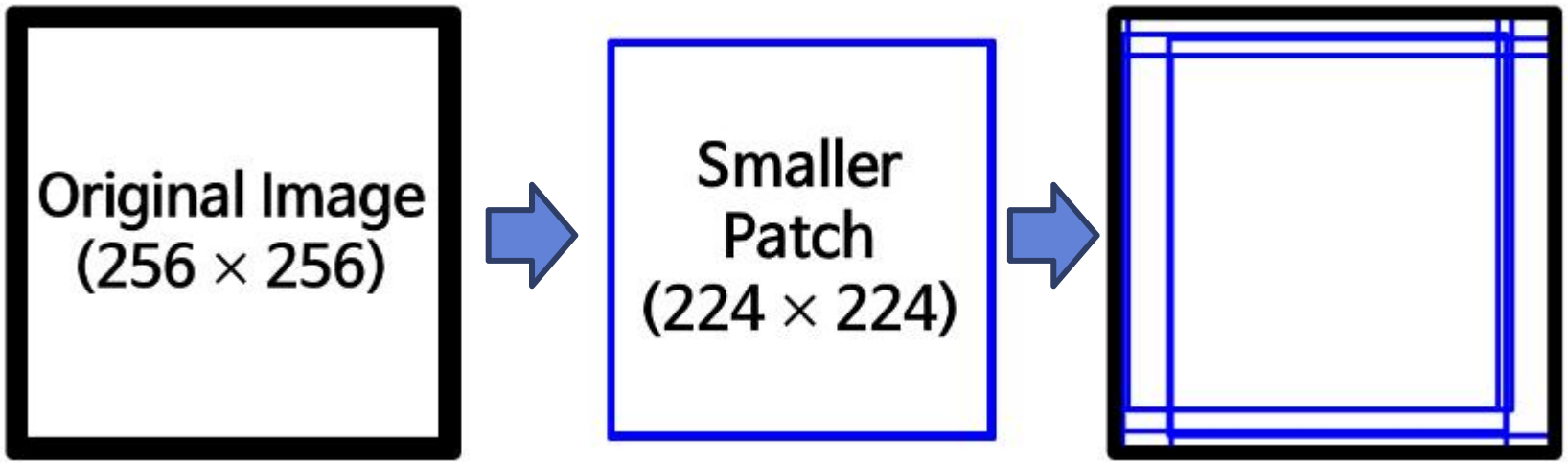
3. Data augmentation



AlexNet

Why AlexNet Good performance?

3. Data augmentation

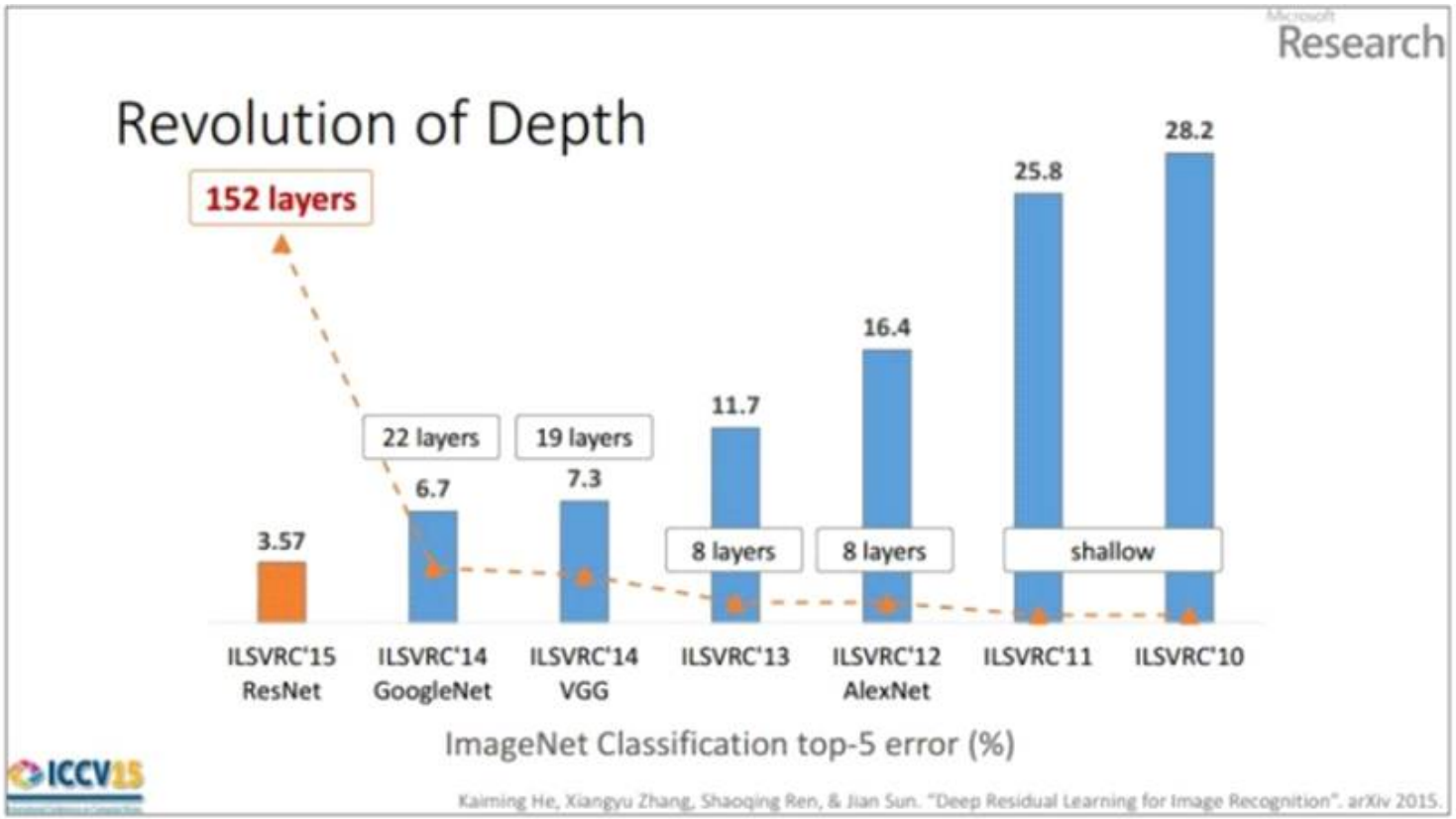


- This increases the size of the training set by a factor of 2048 (32 x 32 x 2).
- Two comes from horizontal reflections.

VGGNet

VGGNet

- VGG16, VGG19 구조를 설명
- 역사적으로 VGGNet부터 모델의 깊이가 매우 깊어짐



VGGNet

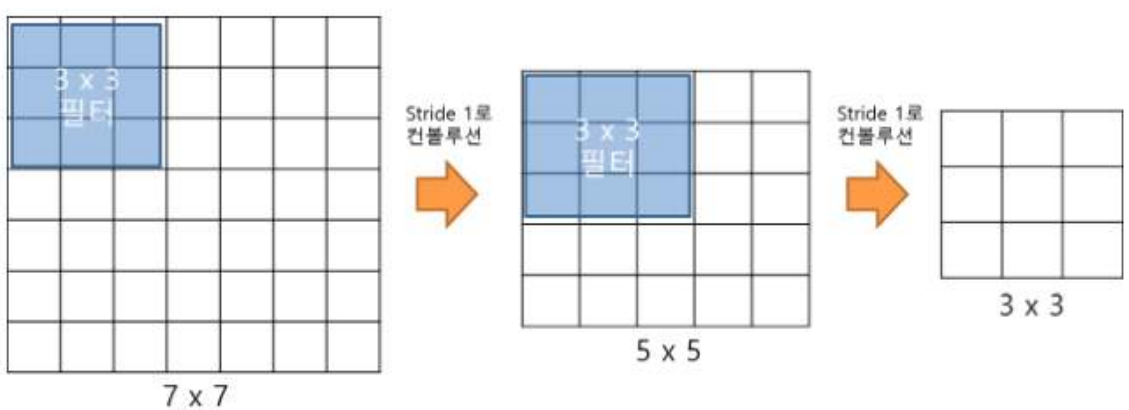
- VGGNet은 논문의 개요에서 밝히고 있듯이 이 연구의 핵심은 네트워크의 깊이를 깊게 만드는 것이 성능에 어떠한 영향을 주는지를 확인
- 이를 확인하기 위해 convolution kernel size를 3 x 3으로 고정
- VGGNet 연구팀은 AlexNet에서 사용되된 Local Response Normalization(LRN)이 성능 향상에 별로 효과가 없다고 실험을 통해 확인

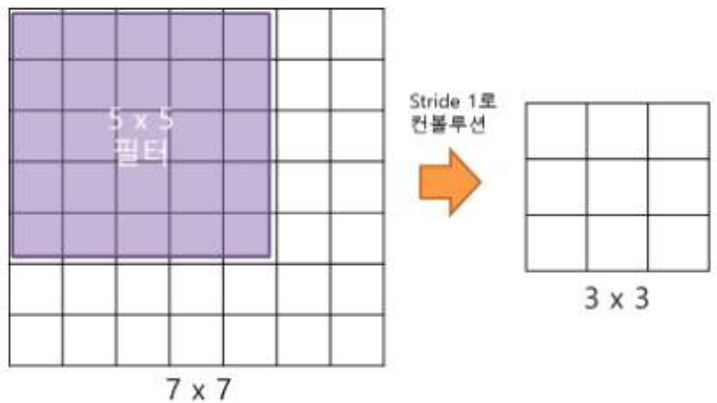
VGGNet

- 3x3 필터로 두 차례 컨볼루션 하는 것과 5x5 한 번 컨볼루션을 하는 것이 결과적으로 동일한 사이즈의 특성 맵을 산출
- 3x3필터로 세 차례 컨볼루션을 하는 것이 7x7필터로 한번 컨볼루션하는 것보다 나은 점은 무엇일까?

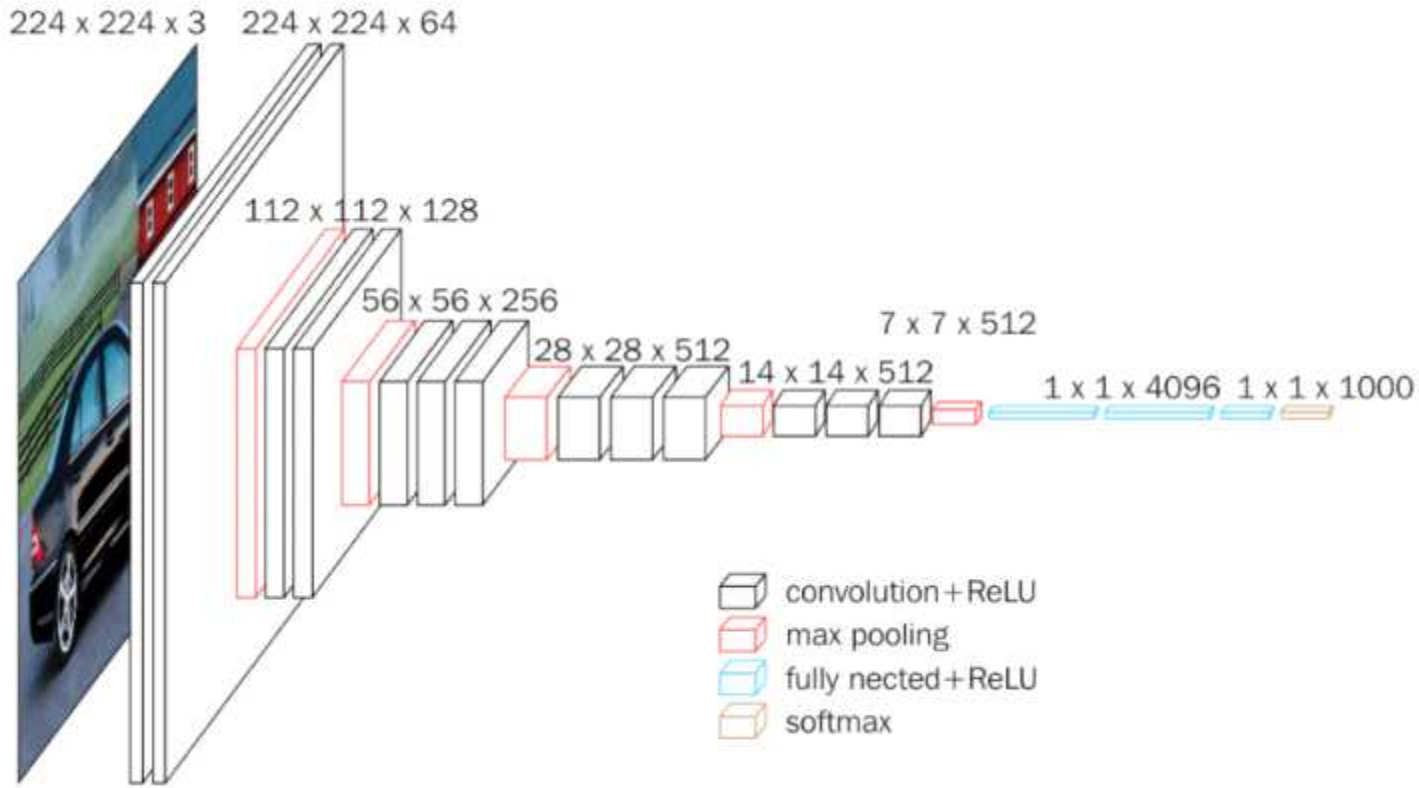
VGGNet

- 3x3 필터로 두 차례 컨볼루션 하는 것과 5x5 한 번 컨볼루션을 하는 것이 결과적으로 동일한 사이즈의 특성 맵을 산출
- 3x3필터로 세 차례 컨볼루션을 하는 것이 7x7필터로 한번 컨볼루션하는 것보다 나은 점은 무엇일까?
  + (3x3) x 3 = 27 kernel size
  + (7x7) = 49 kernel size
  + 가중치가 적어져 훈련시켜야할 것의 갯수가 작아짐 =〉 학습의 속도가 빨라짐
  + 동시의 층의 갯수가 늘어남 =〉 성능 향상





VGGNet

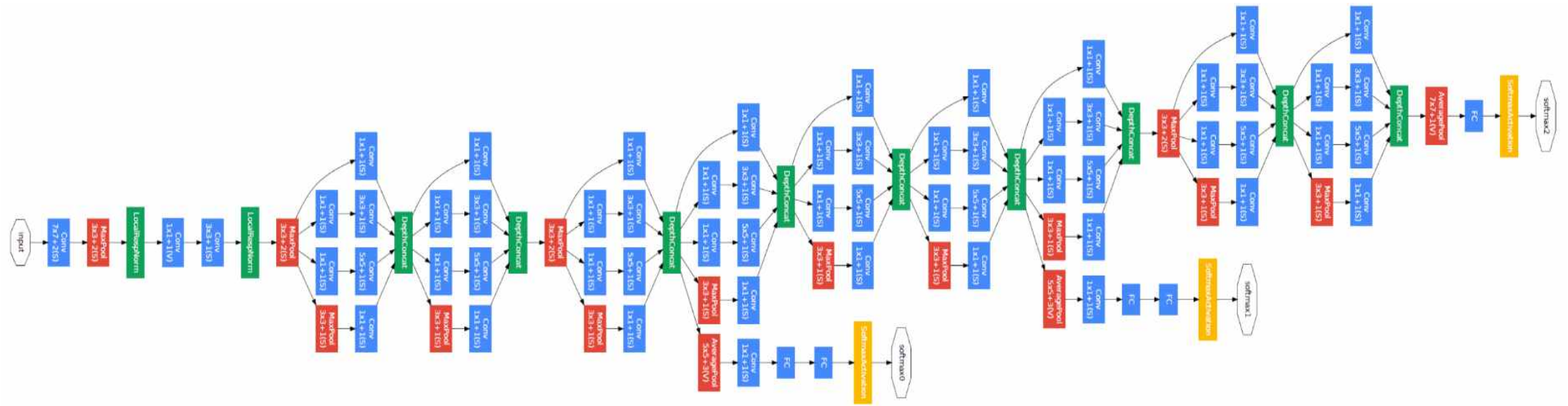


[VGGNet Architecture]

GoogleNet

GoogleNet

- 2014년 이미지넷 이미지 인식 대회에서 VGGNet(VGG19)를 이기고 우승을 차지한 알고리즘
- GoogleNet은 19층의 VGG19보다 좀더 깊은 22층으로 구성되어 있다.

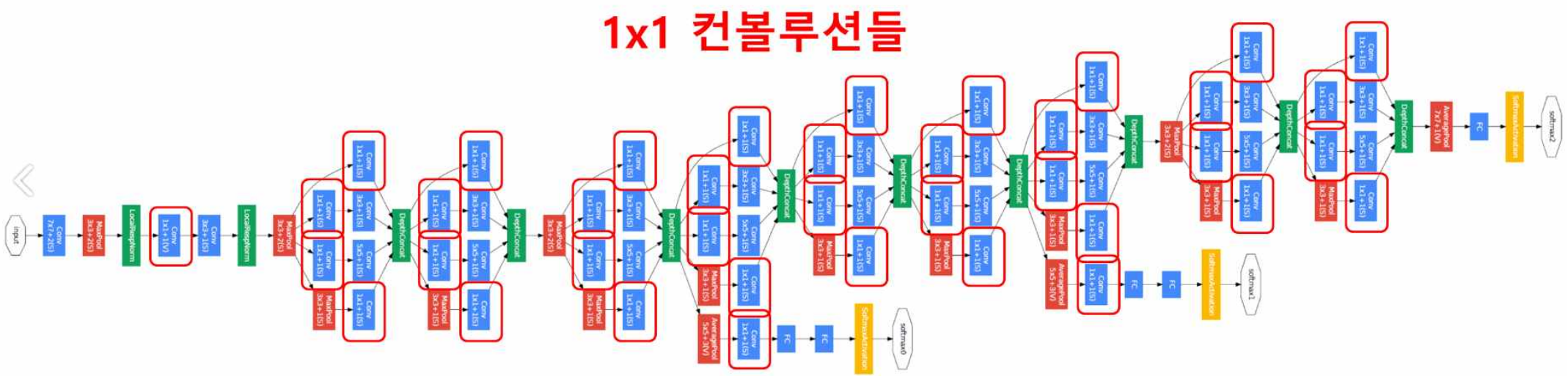


[GoogleNet Architecture]

GoogleNet

1) 1 x 1 Convolution 연산

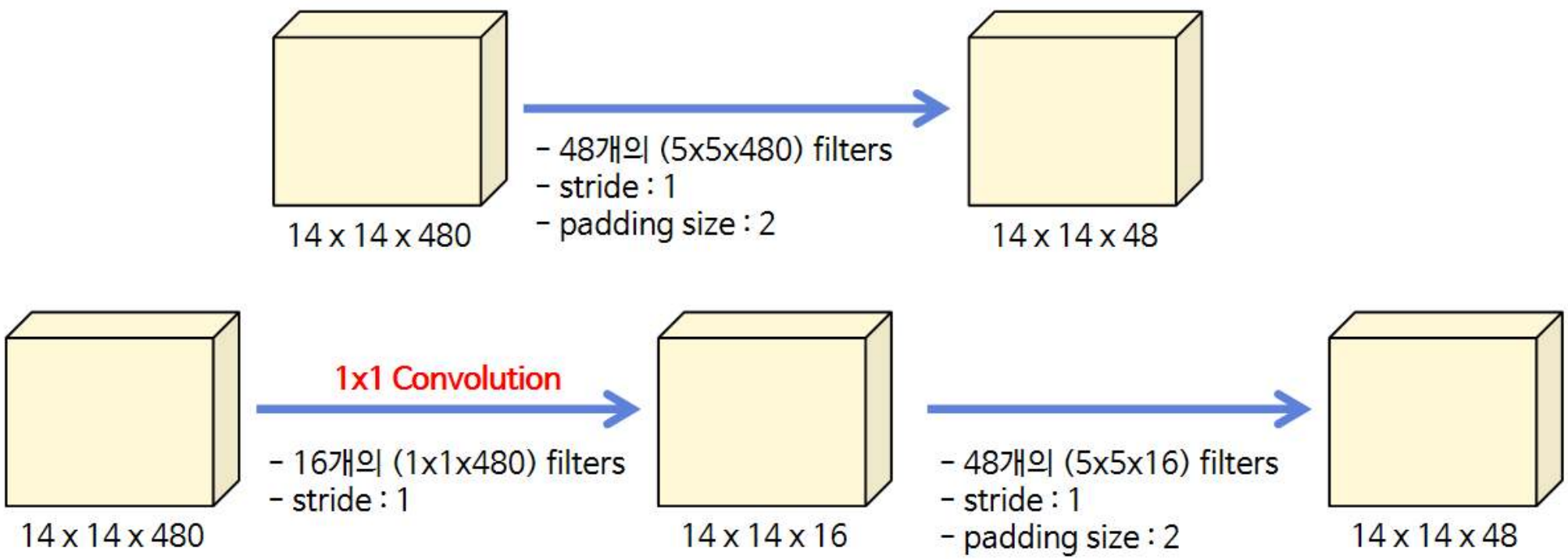
- 구조도를 보면 곳곳에 1x1 Convolution 연산이 존재
- 1x1 Convolution은 어떤 의미를 가지는 것일까?



GoogleNet

1) 1 x 1 Convolution 연산

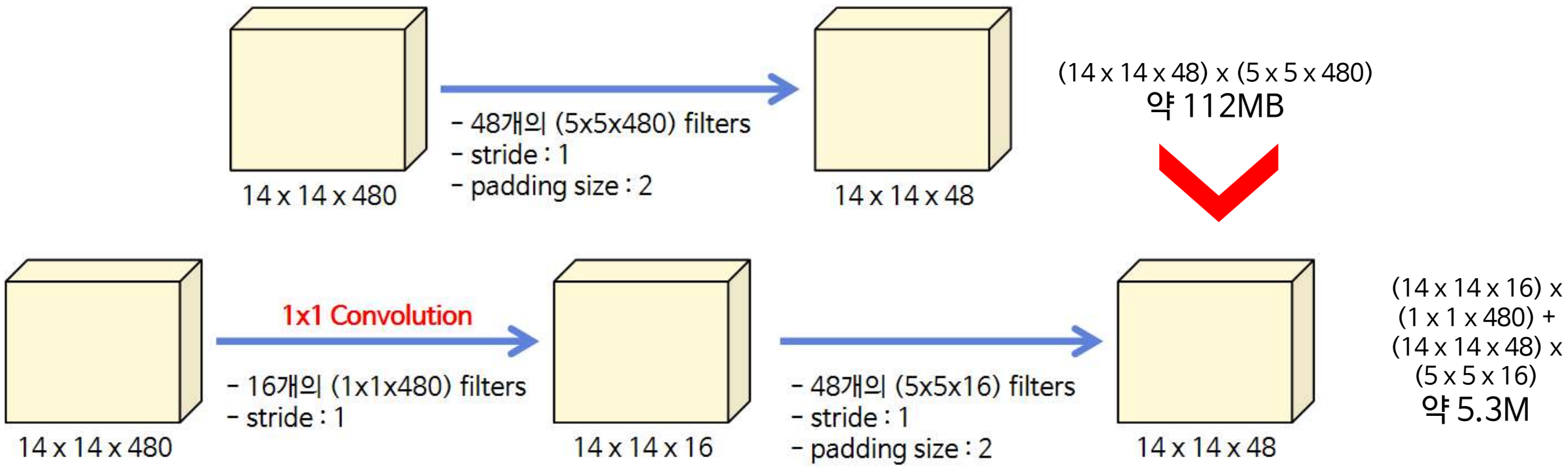
- 구조도를 보면 곳곳에 1x1 Convolution 연산이 존재
- 1x1 Convolution은 어떤 의미를 가지는 것일까?



GoogleNet

1) 1 x 1 Convolution 연산

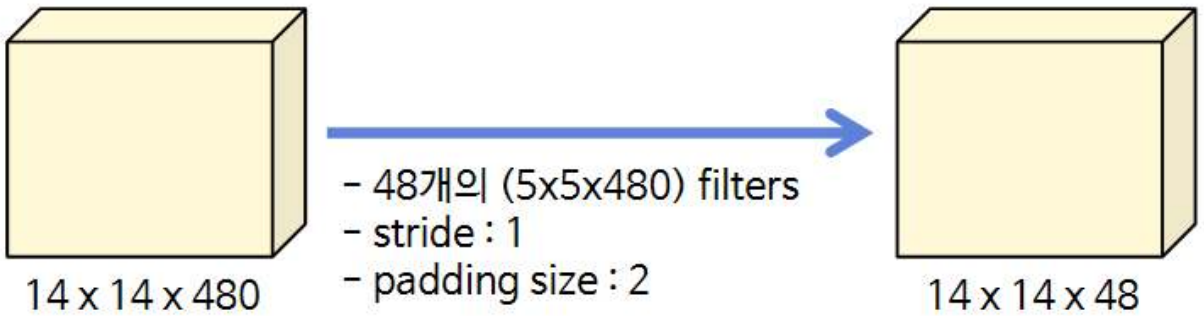
- 구조도를 보면 곳곳에 1x1 Convolution 연산이 존재
- 1x1 Convolution은 어떤 의미를 가지는 것일까? =〉 feature map의 갯수를 줄이는 목적(차원 축소)

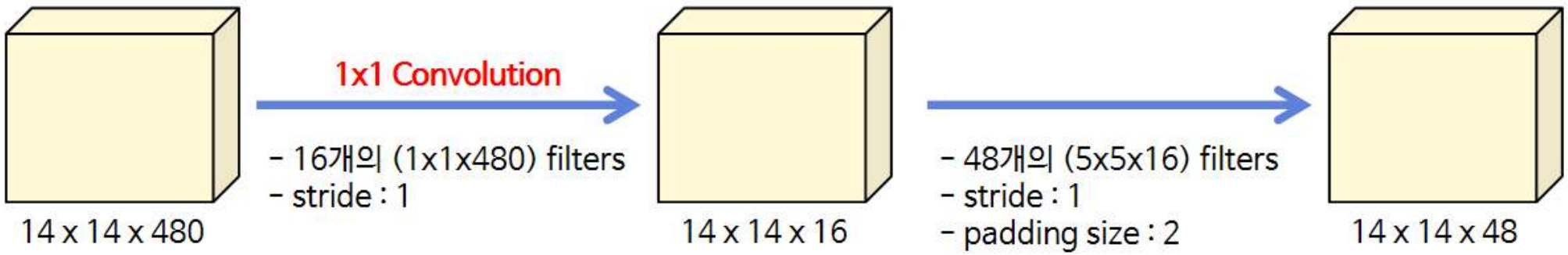


GoogleNet

1) 1 x 1 Convolution 연산

- 구조도를 보면 곳곳에 1x1 Convolution 연산이 존재
- 1x1 Convolution은 어떤 의미를 가지는 것일까? =〉 feature map의 갯수를 줄이는 목적(차원 축소)
- 더 적은 연산량을 가짐으로서 네트워크를 더 깊이 만들수 있게 도와줌

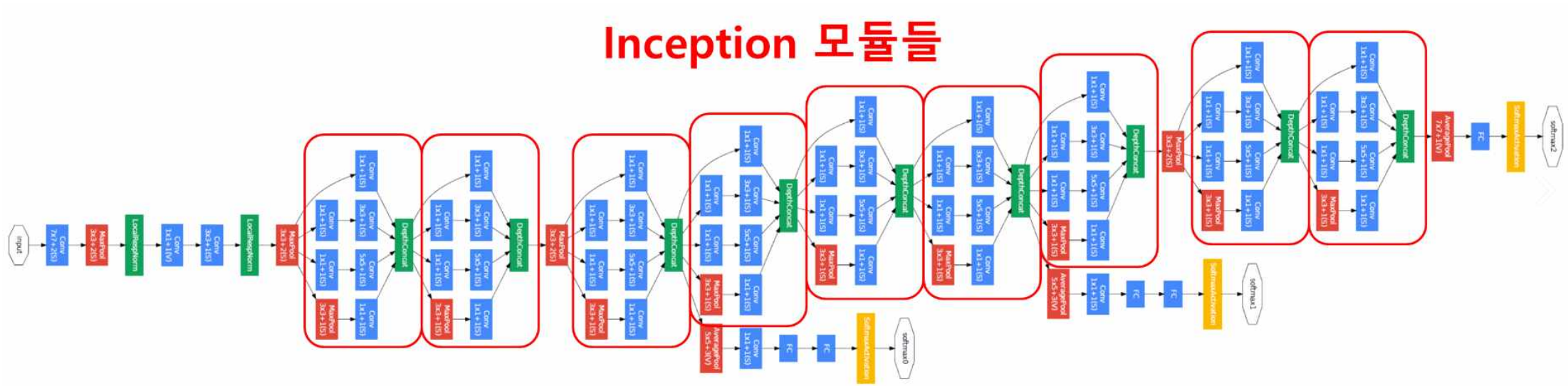




GoogleNet

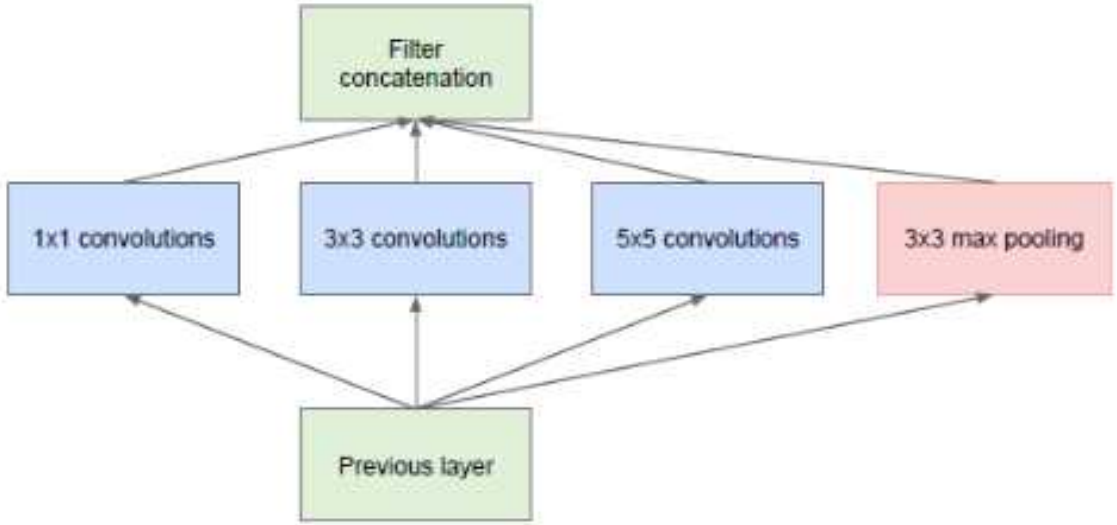
2) Inception 모듈

- 이번엔 GoogleNet의 핵심은 Inception 모듈
- Inception모듈들은 아래 구조도에서 표시하면 다음과 같음

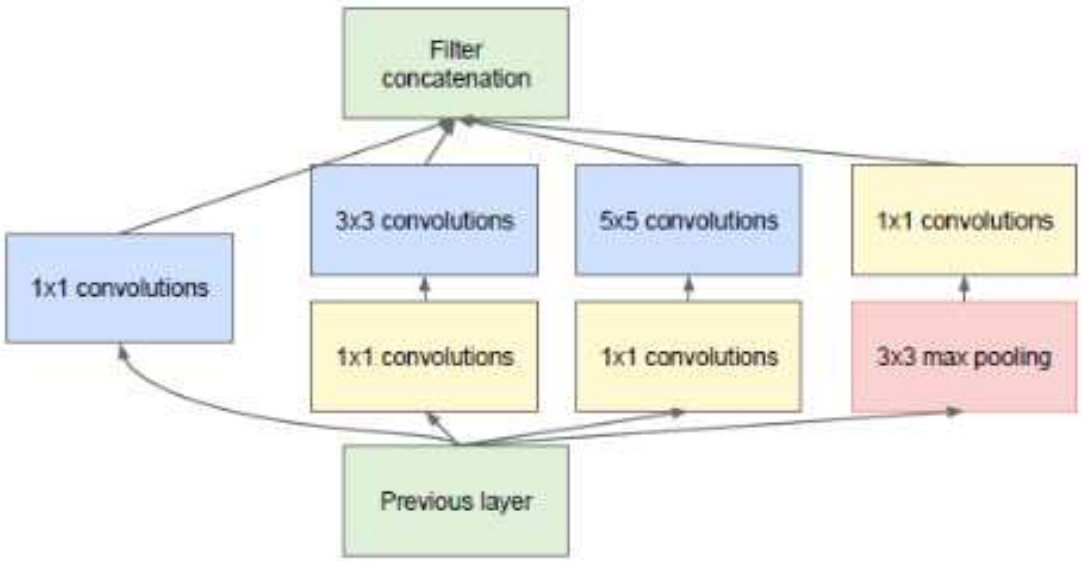


GoogleNet

2) Inception Module



(a) Inception module, naïve version



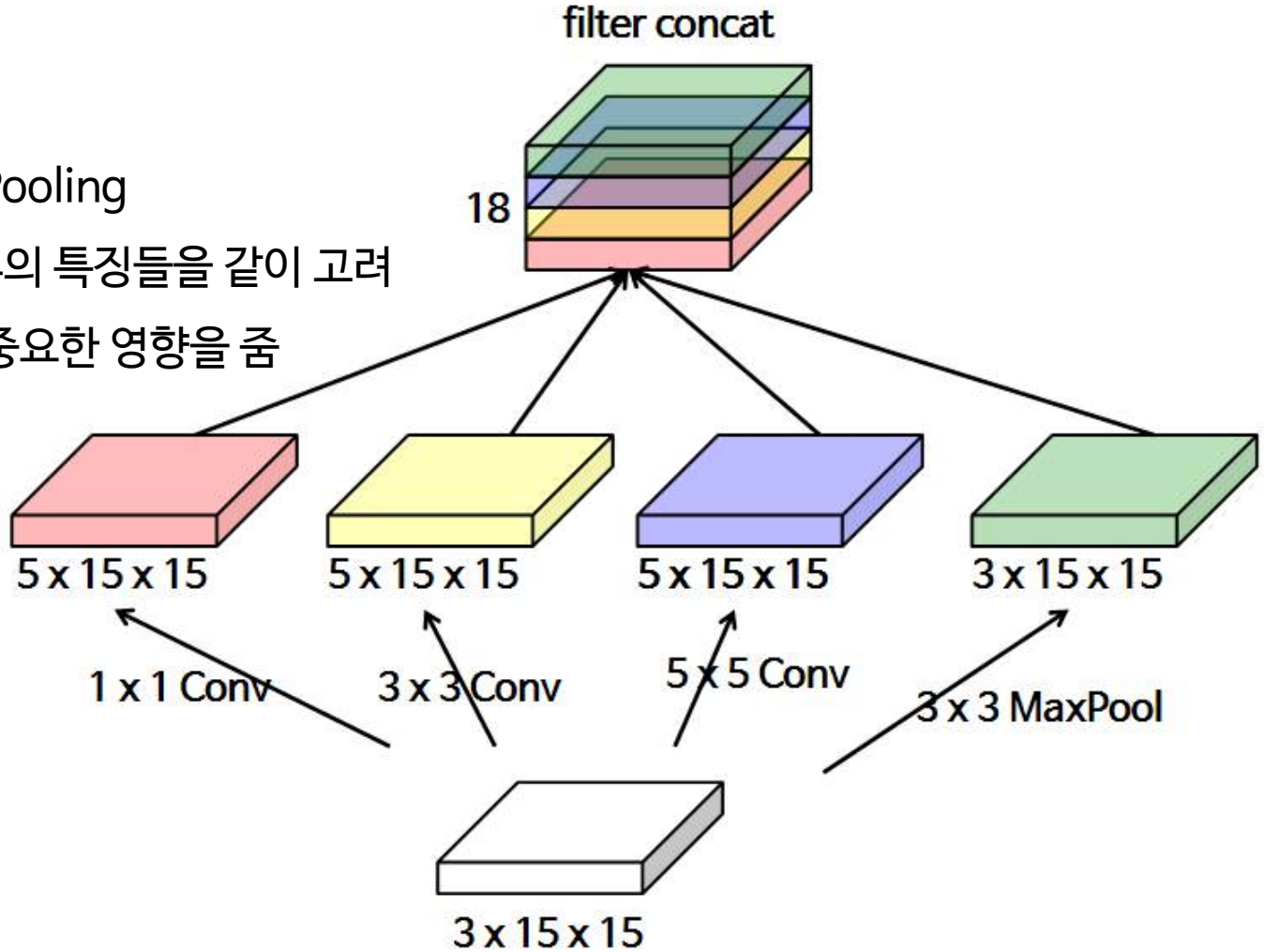
(b) Inception module with dimensionality reduction

GoogleNet

2) Inception Module

Naive inception Module

- 1x1 Conv, 3x3 Conv, 5x5 Conv, 3x3 MaxPooling
- 다양한 filter size를 이용하여 좀 더 다양한 종류의 특징들을 같이 고려
- 다양한 종류의 특징을 뽑아내는 것이 성능에도 중요한 영향을 줌



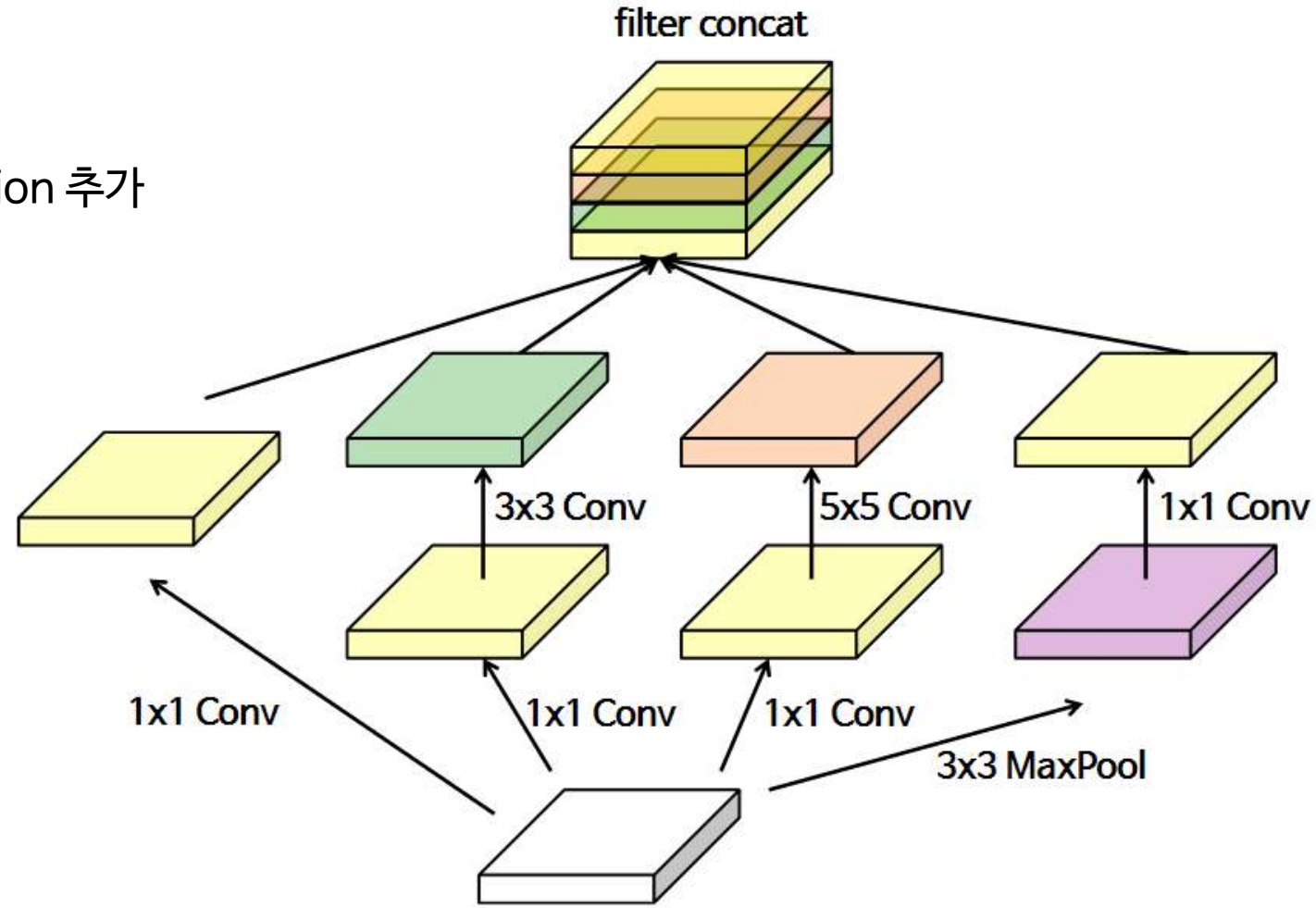
[Naive Inception Architecture]

GoogleNet

2) Inception Module

Acutal Inception Module

- Navie Inception Moduel에 1x1 Convolution 추가



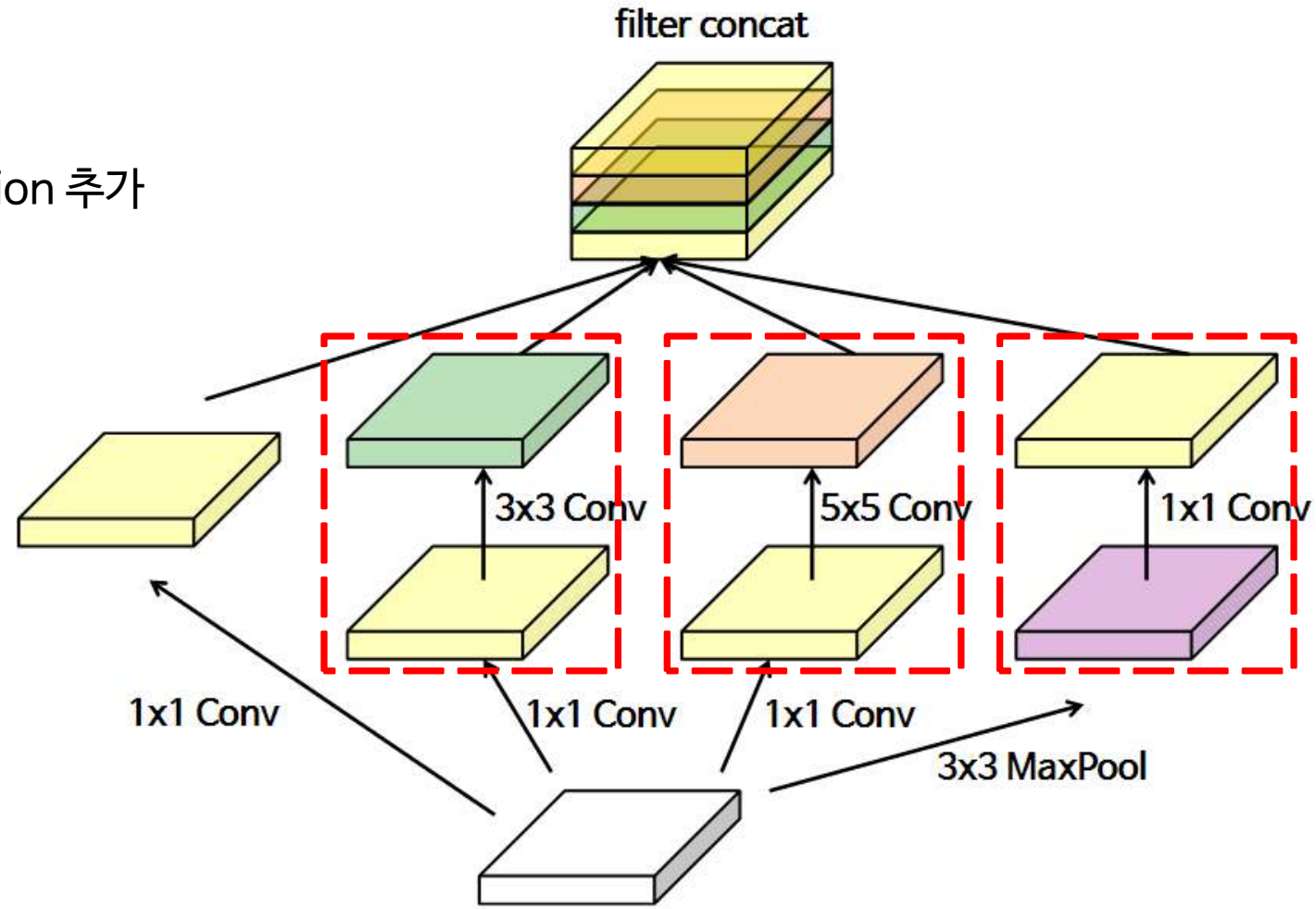
[Acutal Inception Architecture]

GoogleNet

2) Inception Module

Acutal Inception Module

- Navie Inception Moduel에 1x1 Convolution 추가



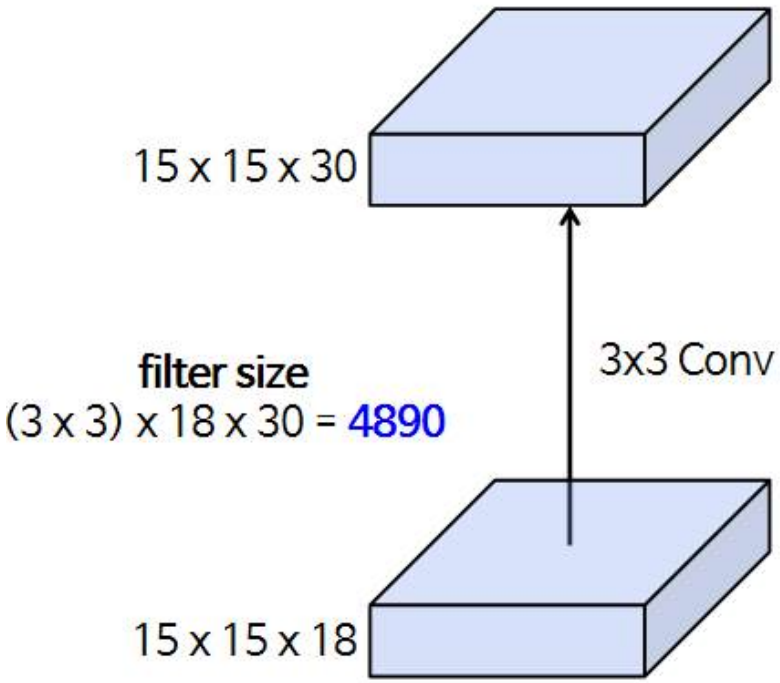
[Acutal Inception Architecture]

GoogleNet

2) Inception Module

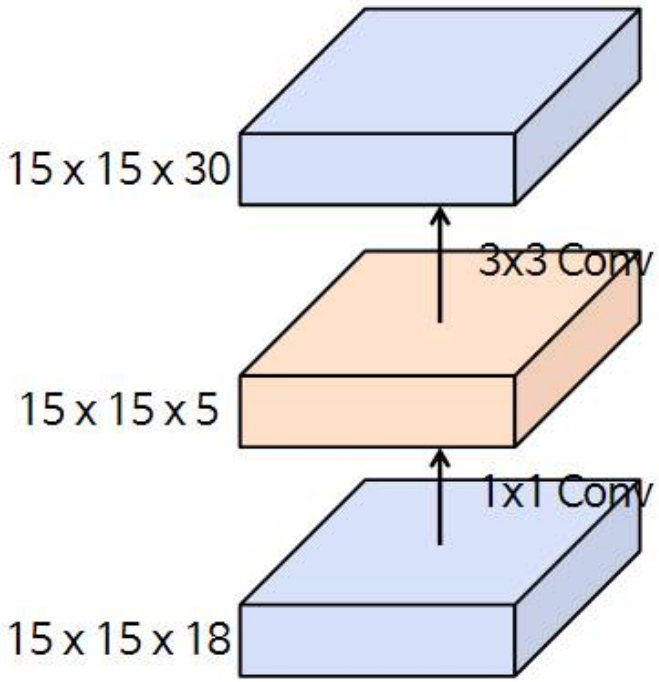
Acutal Inception Module

- Navie Inception Moduel에 1x1 Convolution 추가 =〉 더 적은 연산량을 가짐, 더 깊은 모델 사용 가능



Convolution Neural Network

filter size
(1 x 1) x 5 x 18 = 90
(3 x 3) x 5 x 30 = 1350
90+1350 = 1440

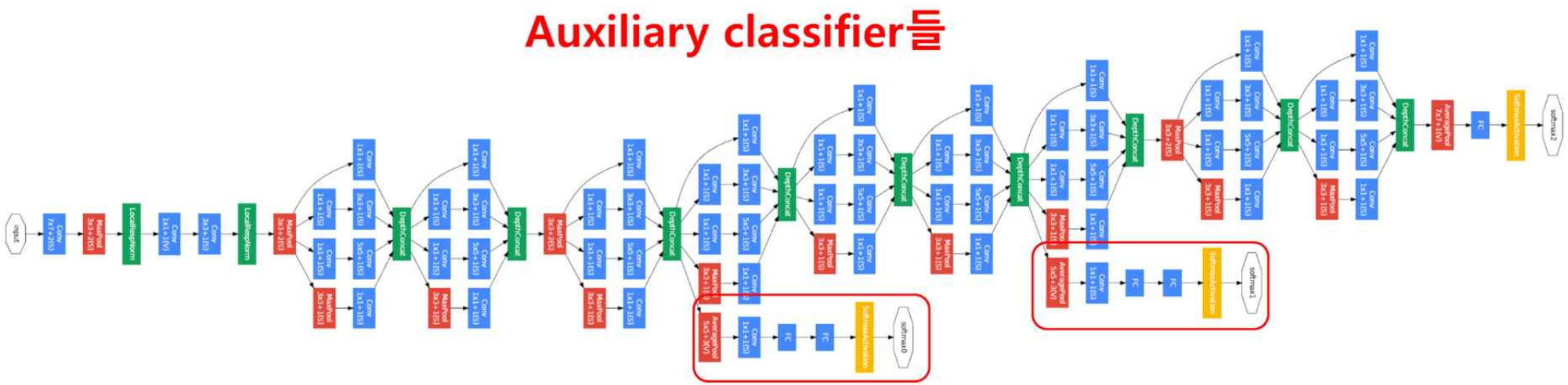


Convolution Neural Network with 1x1 Conv

GoogleNet

4) Auxiliary classifier

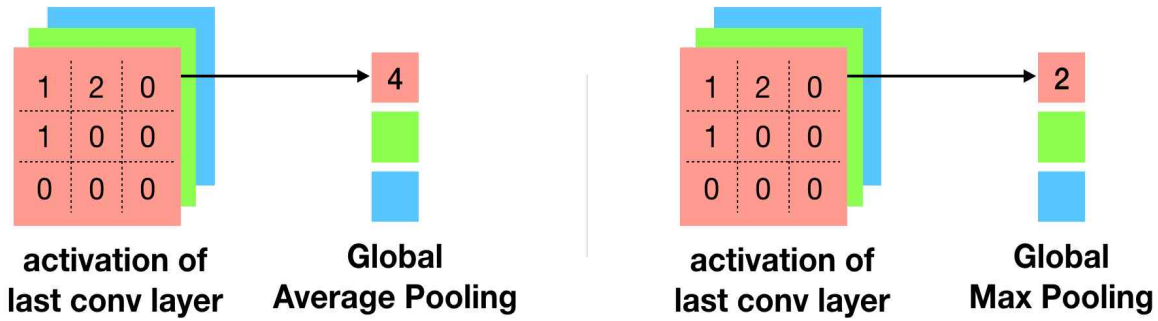
- Network의 깊어지면 깊어질수록 vanishing gradient 문제를 피하기 어려움
- GoogleNet에서는 네트워크 중간에 두 개의 보조 분류기(auxiliary classifier)를 달아줌
- 대신 지나치게 영향을 주는 것을 막기 위해 auxiliary classfier의 loss는 0.3을 곱함
- 실제로 테스트하는 과정에서는 auxiliary classifier를 제거하고 맨 끝, 제일 마지막의 softmax만을 사용



GoogleNet

3) Global average pooling

- Fully Connection 방식 대신 Global average pooling 방식 사용
- Global average pooling은 전 층에서 산출된 feature map을 각각 평균낸것을 이어서 1차원 벡터로 생성
- Why using global average pooling?



[Global Average Pooling & Global Max Pooling]

GoogleNet

3) Global average pooling

- Fully Connection 방식 대신 Global average pooling 방식 사용
- Global average pooling은 전 층에서 산출된 feature map을 각각 평균낸것을 이어서 1차원 벡터로 생성

- Why using global average pooling?

  if convolution filter size (7 x 7 x 1024)

  + fully connected : (7 x 7 x 1024=7168) weight

  + global average polling : 1024 weight (필요한 파라미터 사이즈가 작다!!)

  + 필요한 파라미터 사이즈가 작으면?

GoogleNet

3) Global average pooling

- Fully Connection 방식 대신 Global average pooling 방식 사용
- Global average pooling은 전 층에서 산출된 feature map을 각각 평균낸것을 이어서 1차원 벡터로 생성

- Why using global average pooling?

  if convolution filter size (7 x 7 x 1024)

  + fully connected : (7 x 7 x 1024=7168) weight

  + global average polling : 1024 weight (필요한 파라미터 사이즈가 작다!!)

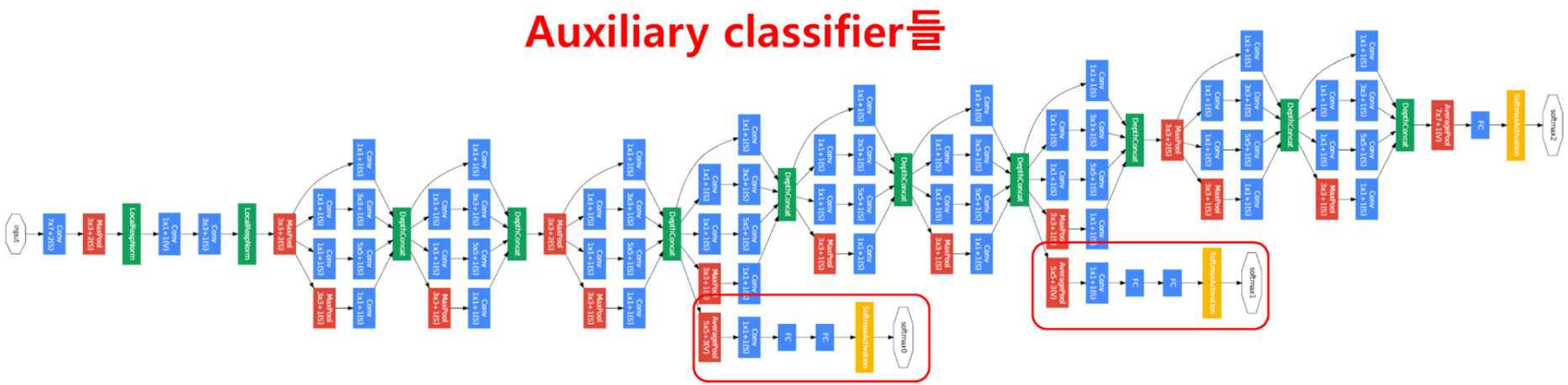
  + 필요한 파라미터 사이즈가 작으면?

  => 많은 파라미터 수로 인한 overfitting에 빠질 가능성이 줄어듦

GoogleNet

4) Auxiliary classifier

- Network의 깊어지면 깊어질수록 vanishing gradient 문제를 피하기 어려움
- GoogleNet에서는 네트워크 중간에 두 개의 보조 분류기(auxiliary classifier)를 달아줌
- 대신 지나치게 영향을 주는 것을 막기 위해 auxiliary classfier의 loss는 0.3을 곱함
- 실제로 테스트하는 과정에서는 auxiliary classifier를 제거하고 맨 끝, 제일 마지막의 softmax만을 사용



ResNet

ResNet

- 152 layers network
- 1st place on ILSVRC 2015 classification task
- 1st place on ImageNet detection
- 1st place on ImageNet localization
- 1st place on COCO detection
- 1st place on COCO segmentation

ResNet

- 152 layers network
- 1st place on ILSVRC 2015 classification task
- 1st place on ImageNet detection
- 1st place on ImageNet localization
- 1st place on COCO detection
- 1st place on COCO segmentation

=〉 모든 분야에서 최고의 성능을 보임

=〉 ResNet에서 쓰이는 기법을 사용하면 성능의 상승을 보장할 가능성이 높음 =〉 범용성이 높음 (backbone)

ResNet

Is deeper network always better?

- What about vanishing/exploding gradients?
  + 해결법 : Better initialization methods / batch normalization / ReLU
  + 즉 vanishing/exploding 에 대해서는 위와 같은 방법으로 어느정도 해결됨

- Any other problems?
  + Overfitting?

ResNet

Is deeper network always better?

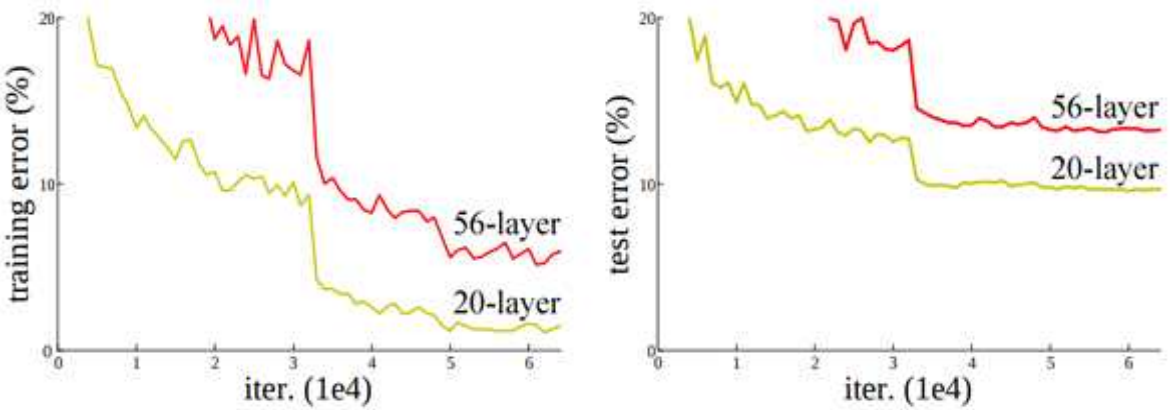
- What about vanishing/exploding gradients?
  + 해결법 : Better initialization methods / batch normalization / ReLU
  + 즉 vanishing/exploding 에 대해서는 위와 같은 방법으로 어느정도 해결됨

- Any other problems?
  + Overfitting? No!!
  + Degradation problem: more depth but lower preformance

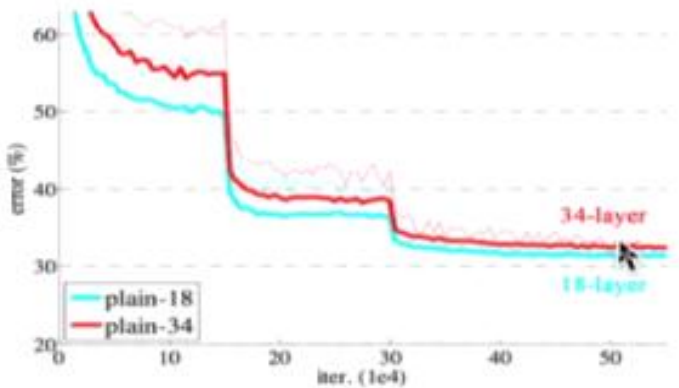
ResNet

What a Degradation prolem?

- overfitting이 발생하지 않음에도 불구하고, shallower network보다 성능이 나쁨
- Deep network는 이론상 새로우 추가된 layer의 identity mapping을 학습할수 있어서, shallower network보다 성능이 좋아야 됨
- Deep network 일수록 성능이 안나오는 경우가 많음 =〉 Degradiation Problem



[CIFAR 100 Dataset]



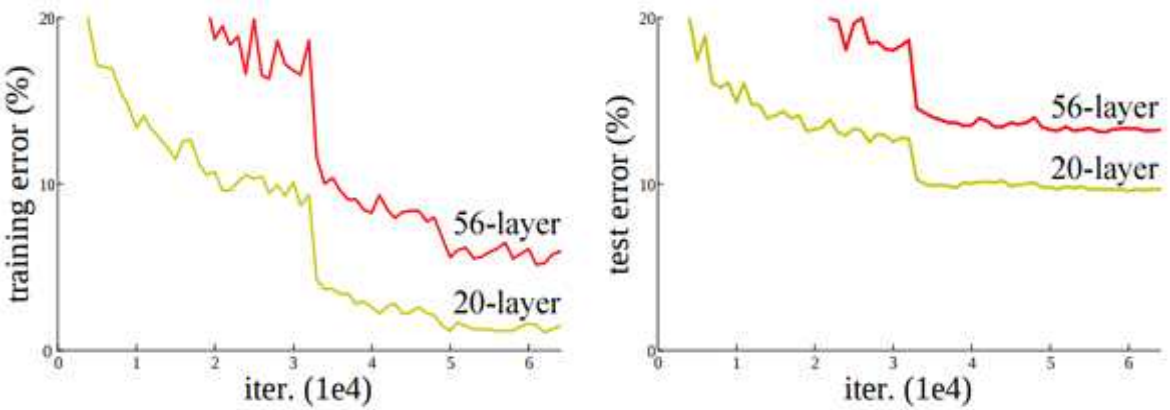
[ImageNet]

ResNet

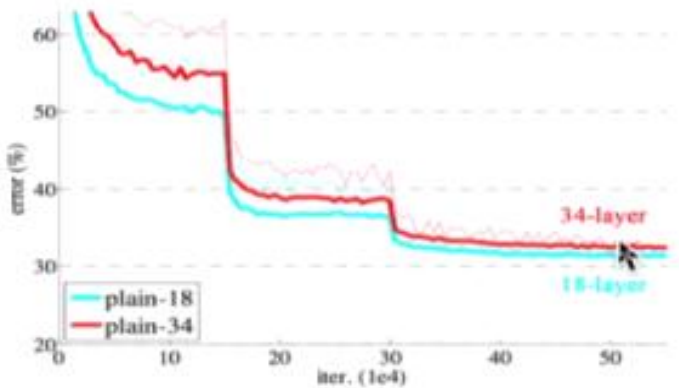
What a Degradation prolem?

- overfitting이 발생하지 않음에도 불구하고, shallower network보다 성능이 나쁨
- Deep network는 이론상 새로우 추가된 layer의 identity mapping을 학습할수 있어서, shallower network보다 성능이 좋아야 됨
- Deep network 일수록 성능이 안나오는 경우가 많음 =〉 Degradiation Problem

  =〉 해결법 : Residual learning



[CIFAR 100 Dataset]



[ImageNet]

ResNet

Residual learning building block

- 입력과 출력의 값을 더해주면 끝 (very simple) =〉 입력단과 출력단의 dementionㅇ 같아야됨

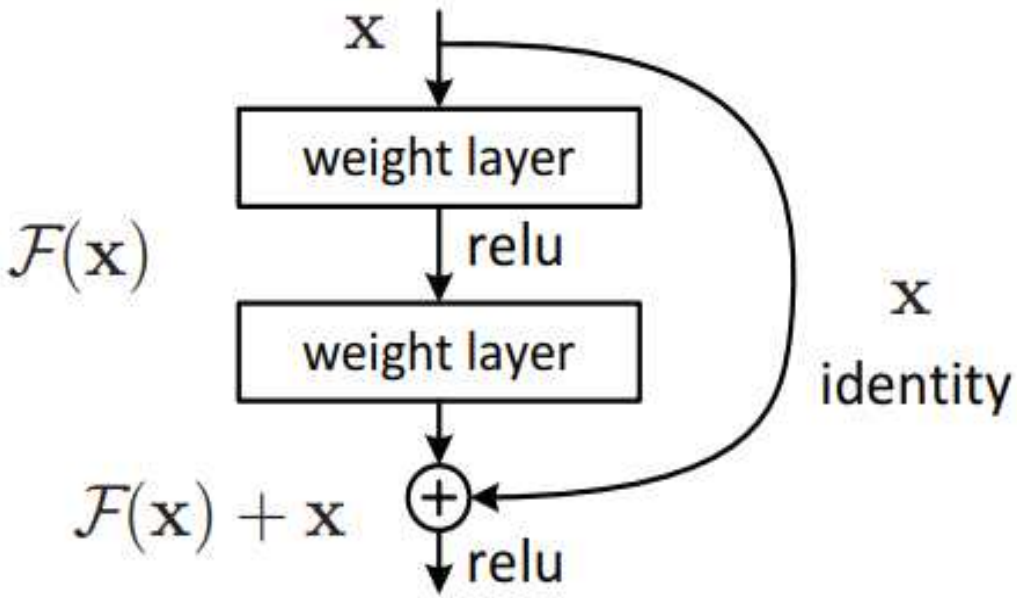


Figure 2. Residual learning: a building block.

ResNet

Residual learning building block

- 입력과 출력의 값을 더해주면 끝 (very simple) =〉 입력단과 출력단의 dementionㅇ 같아야됨

이 Network는 결국

입력과 출력의 차이만을 학습

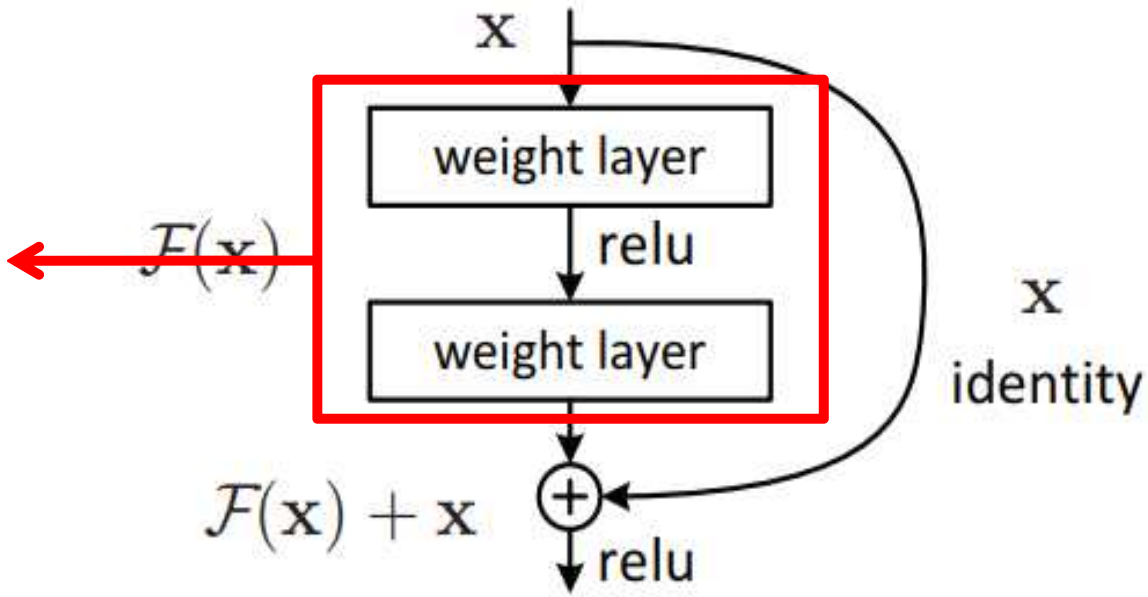


Figure 2. Residual learning: a building block.

ResNet

Why used residual?

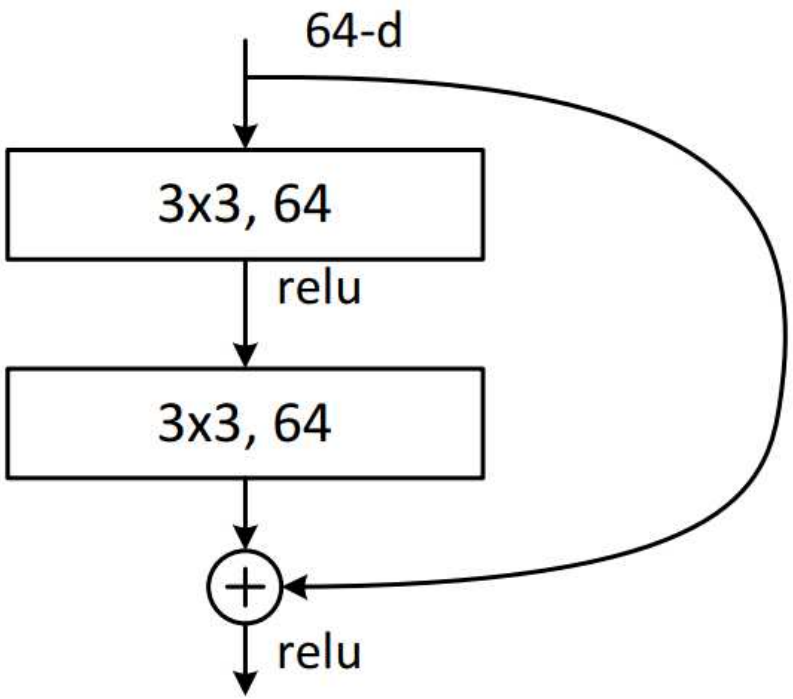
- 어떤 수학적인 background를 바탕으로 residual를 개발한 것이 아니라 단순한 가정 (hypothesize)을 바탕으로 개발
- Optimizer의 성능을 극단적으로 올려줌!!

layers fit another mapping of $\mathcal{F}(\mathbf{x}) := \mathcal{H}(\mathbf{x}) - \mathbf{x}$. The original mapping is recast into $\mathcal{F}(\mathbf{x}) + \mathbf{x}$. We hypothesize that it is easier to optimize the residual mapping than to optimize the original, unreferenced mapping. To the extreme, if an identity mapping were optimal, it would be easier to push the residual to zero than to fit an identity mapping by a stack of nonlinear layers.

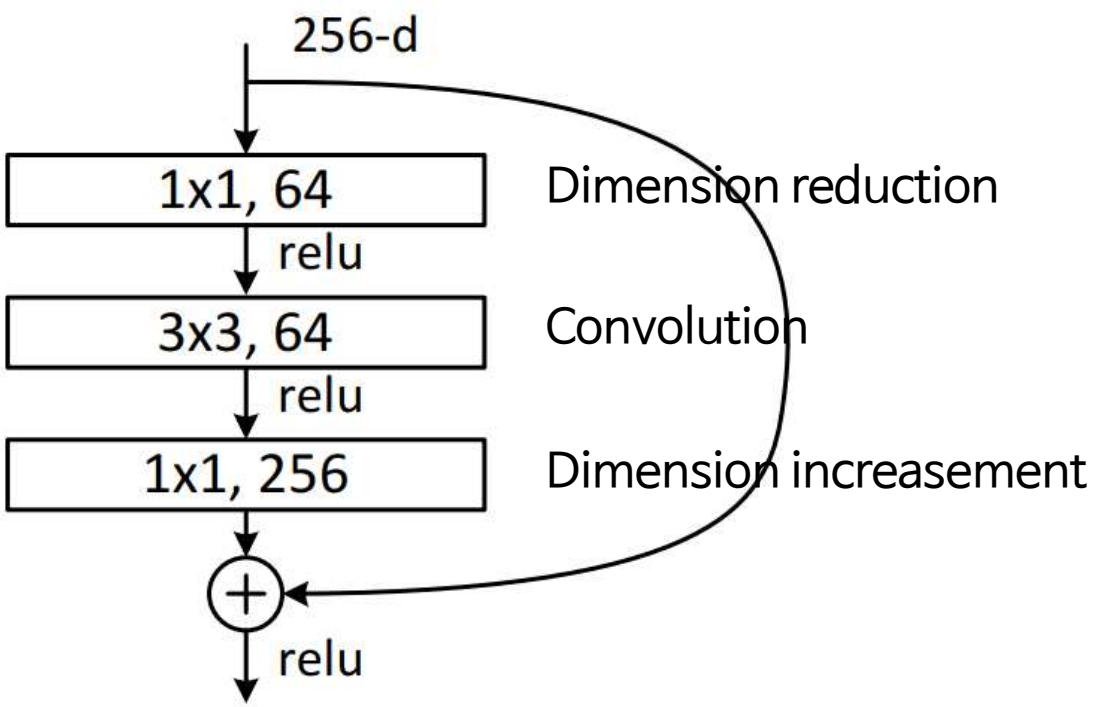
We present comprehensive experiments on ImageNet [36] to show the degradation problem and evaluate our method. We show that: 1) Our extremely deep residual nets are easy to optimize, but the counterpart "plain" nets (that

ResNet

Deeper bottle architecture



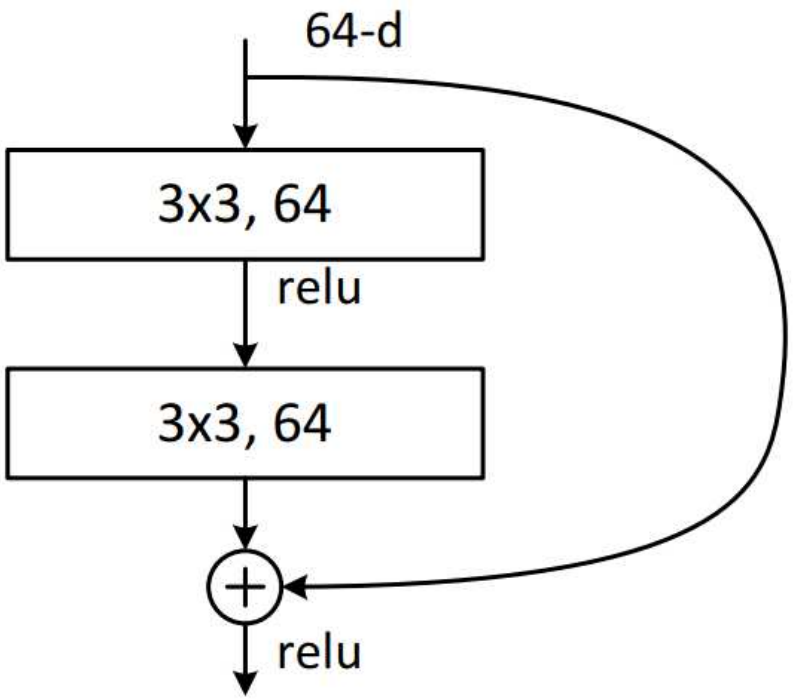
[일반적인 구조]



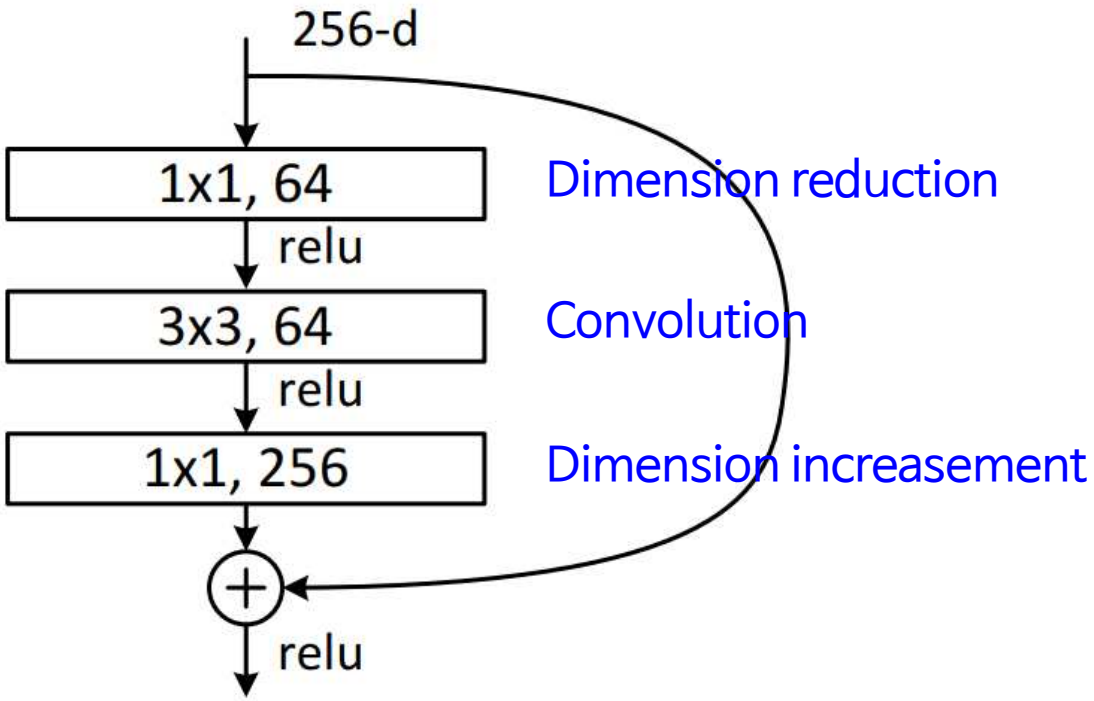
[Bottle 구조]

ResNet

Deeper bottle architecture



[일반적인 구조]

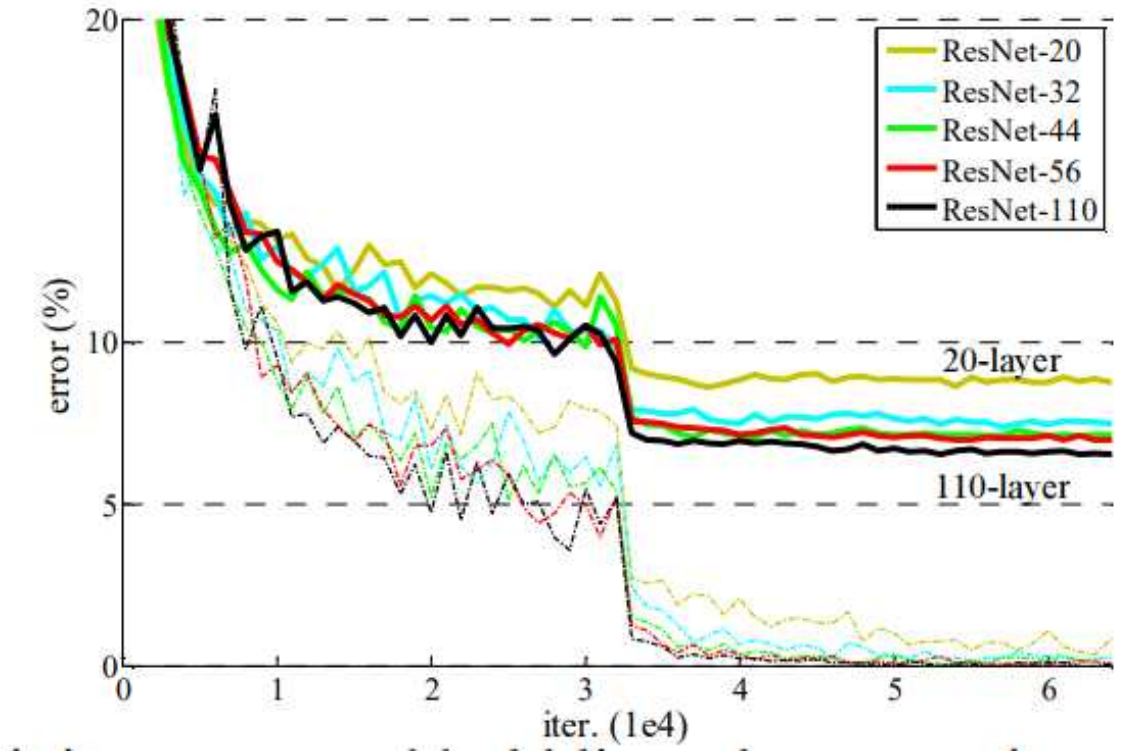


[Bottle 구조]

- parameter 사이즈를 줄이기 위해
- Deminsion increasement를 통해 인풋차원으로 복원

ResNet

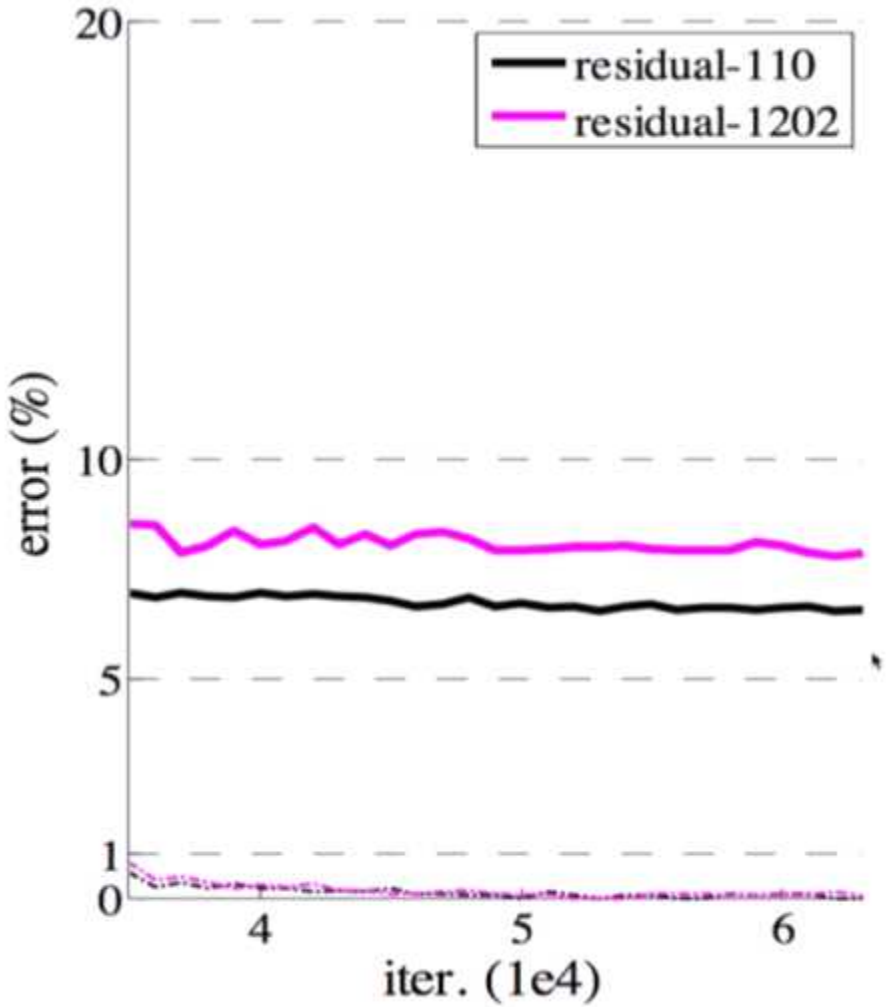
Experimental result



100단 단위까지 줄일때, 줄일수록 성능이 개선됨을 확인할 수 있음

ResNet

Experimental result



하지만 1000단 단위까지 올리면 여전히 Deradation이 발생함