4주차 스터디

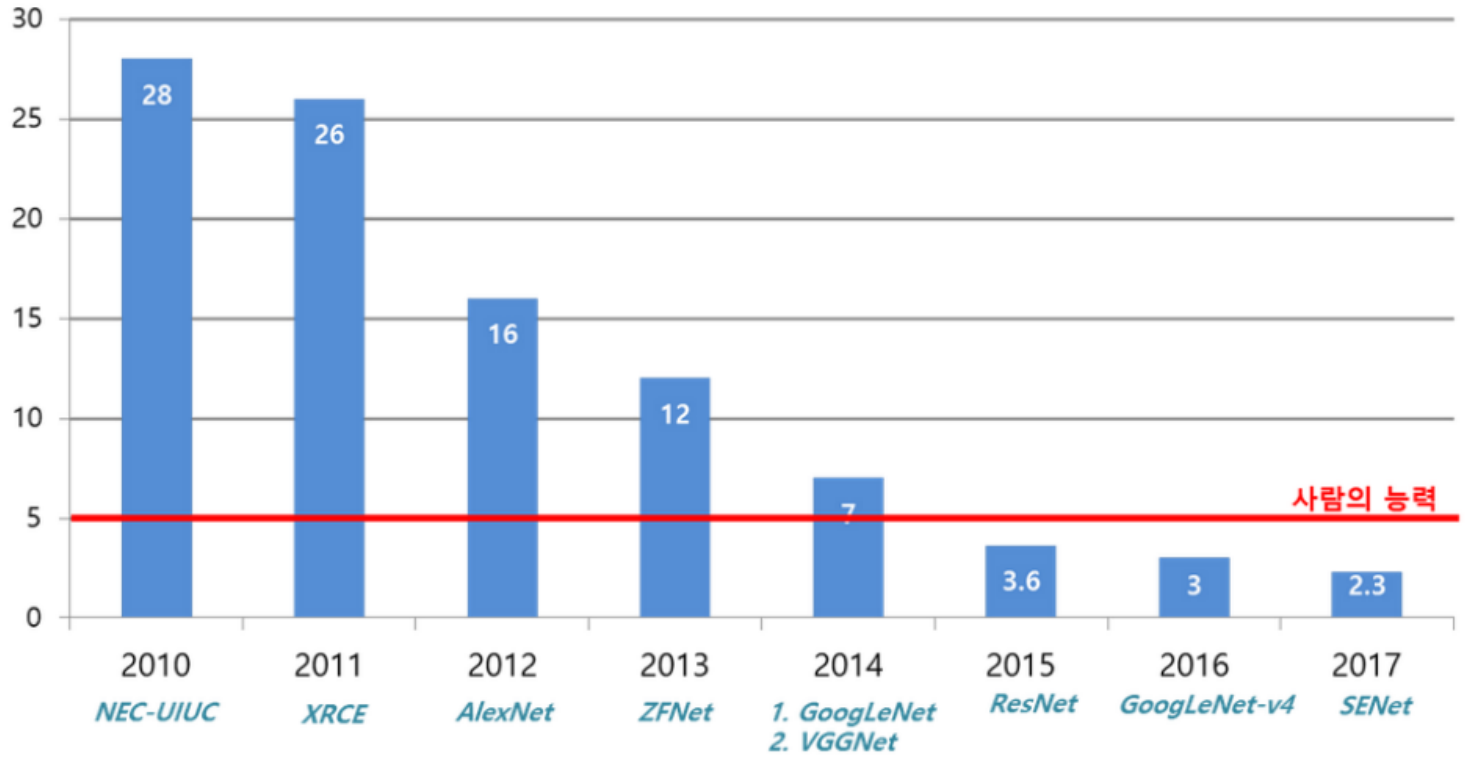
딥러닝 CNN 완벽 가이드 – 세션 9,10

세션9 – Advanced CNN 모델 파헤치기 - AlexNet, VGGNet, GoogLeNet

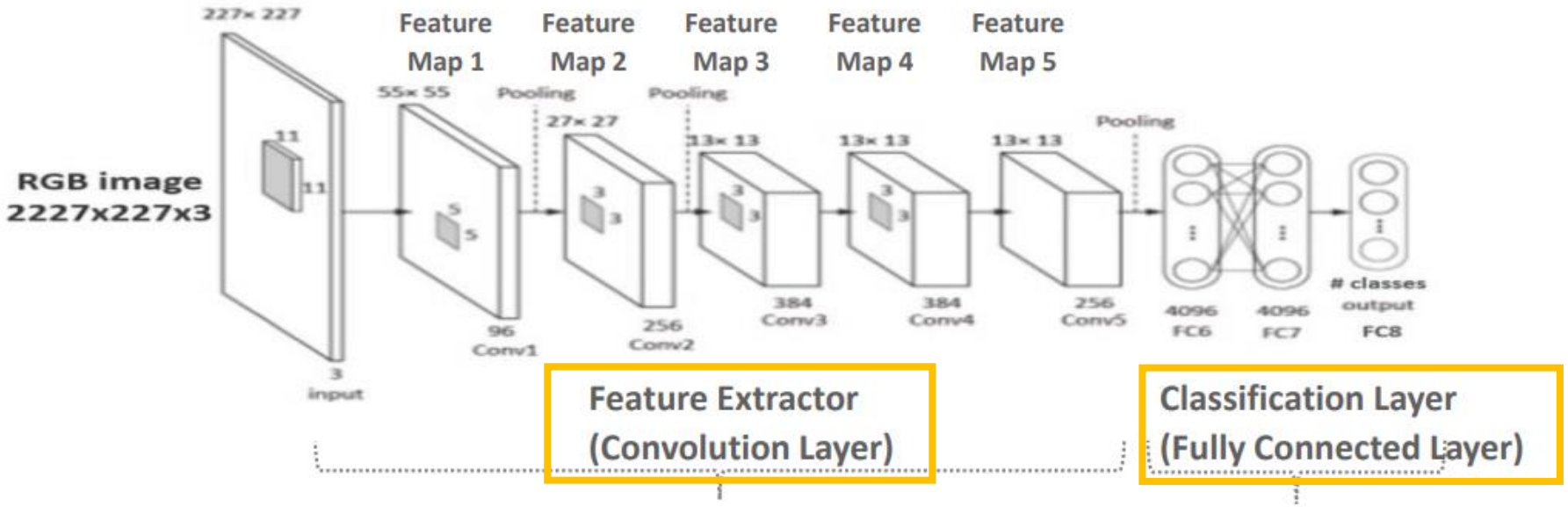
세션10 – Advanced CNN 모델 파헤치기 - ResNet 상세와 EfficientNet 개요

발표자 : 이현진, 임수진

ImageNet 대회 역대 우승 알고리즘과 Top-5 에러율

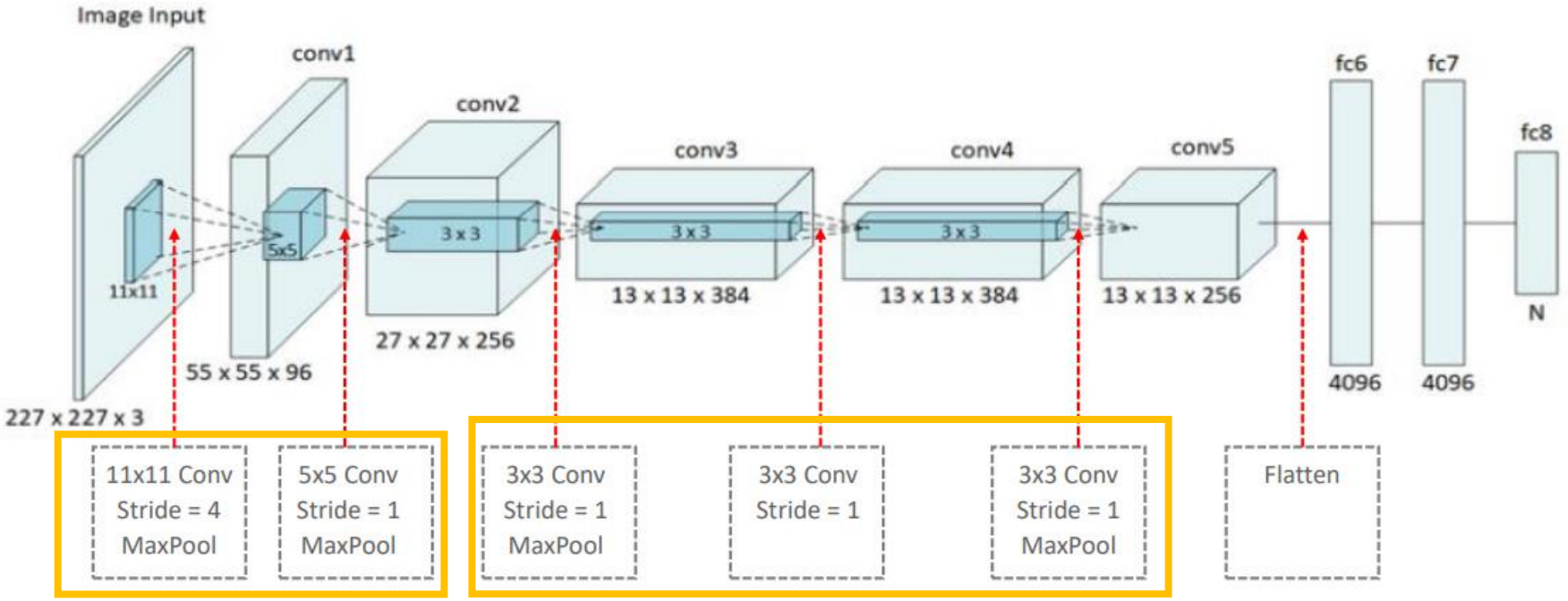


CNN 아키텍처의 특징



- Feature Extractor 와 Classification Layer 영역으로 구성
- 연속적인 CNN 연산을 순차적으로 수행하면서 일련의 Feature Map을 생성
- 순차적으로 생성된 Feature Map의 크기(높이x너비)는 줄어들지만 채널(깊이)은 증가
- 더 깊은 영역에 있는 네트워크가 더 복잡한 feature 정보를 반영하면서 CNN 깊이를 증가시키는 방향으로 아키텍처를 발전시킴

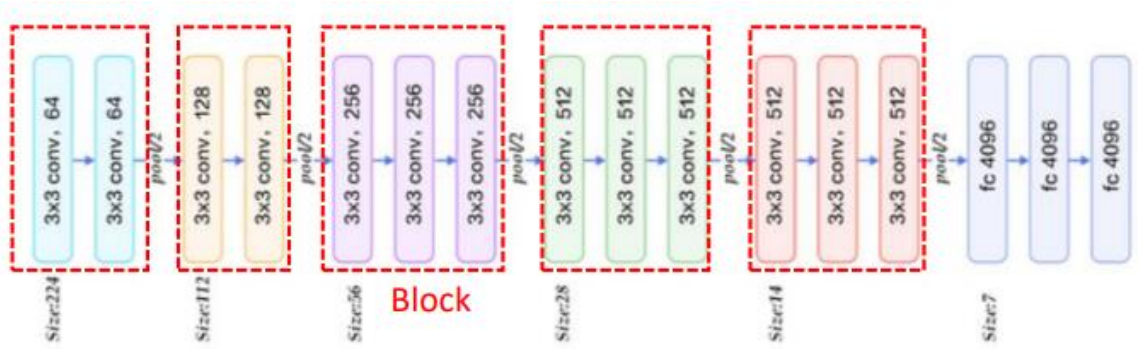
1. AlexNet

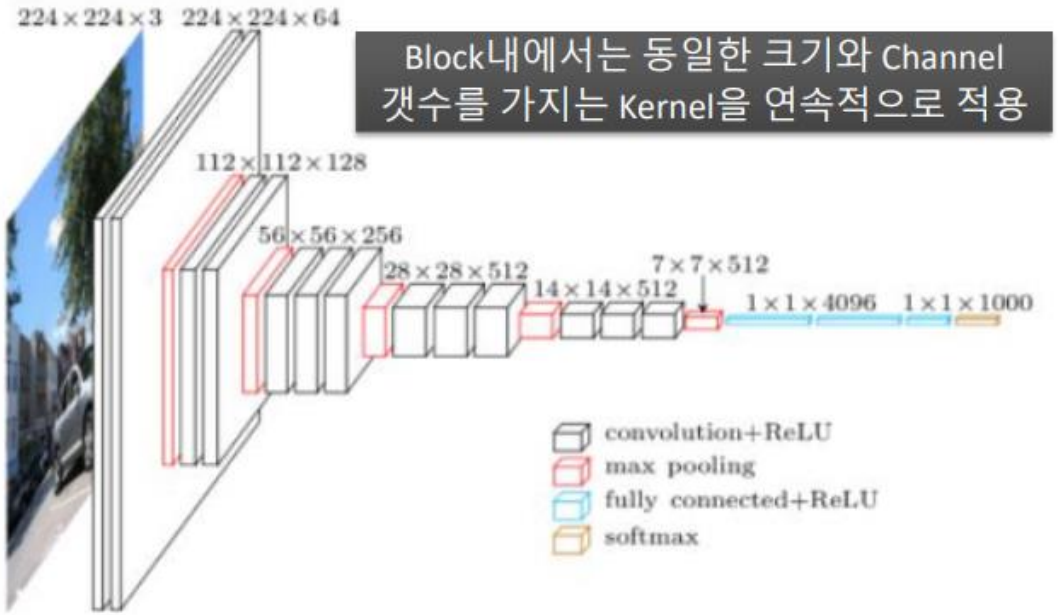


- 11x11, 5x5 사이즈의 큰 사이즈의 커널을 적용, 이후 3x3커널을 3번 이어서 적용

- Receptive field가 큰 사이즈를 초기 feature map에 적용하는 것이 보다 많은 feature 정보를 만드는데 효율적이라고 판단
  - 많은 파라미터 개수로 인하여 연산량이 크게 증가. 이를 극복하기 위해 병렬 GPU를 활용할 수 있도록 CNN모델 병렬화

2. VGGNet

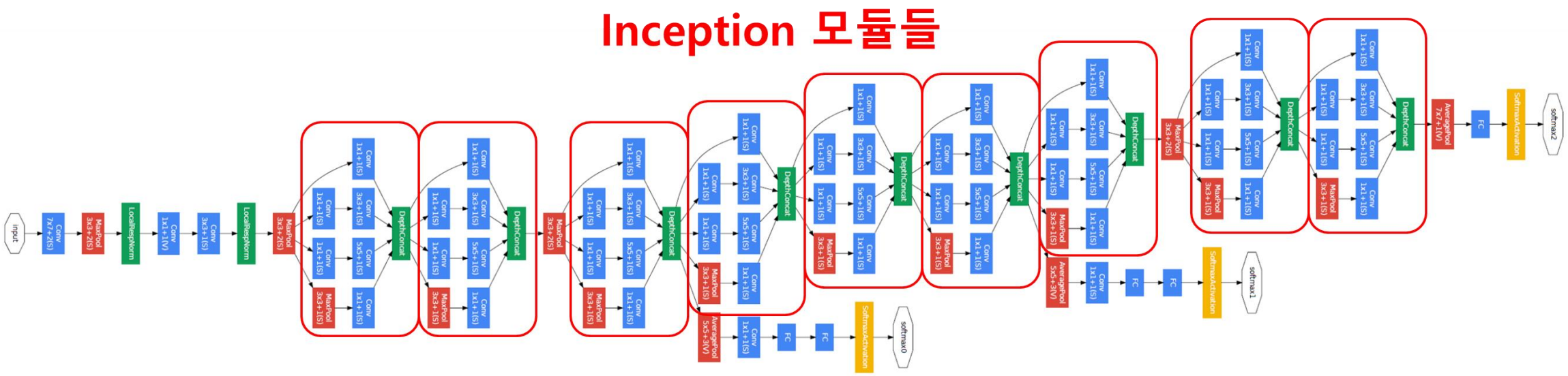


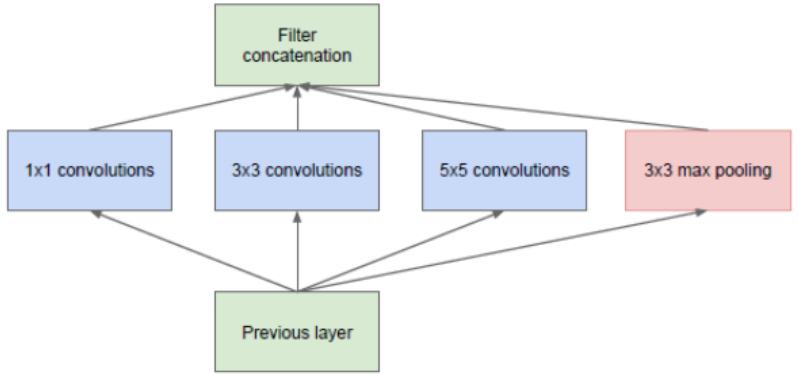


Block내에서는 동일한 크기와 Channel 갯수를 가지는 Kernel을 연속적으로 적용

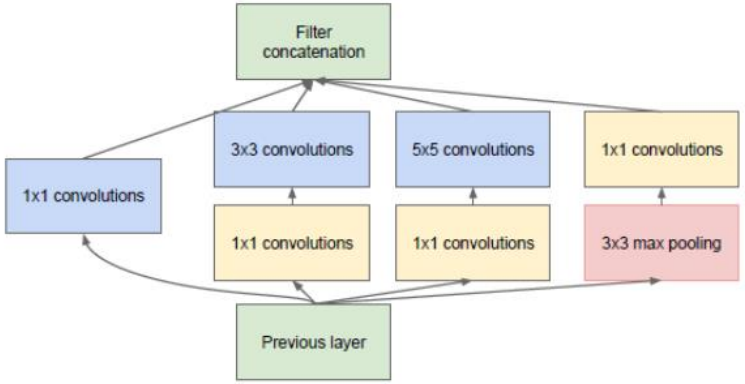
ConvNet Configuration					
A	A-LRN	B	C	D	E
11 weight layers	11 weight layers	13 weight layers	16 weight layers	16 weight layers	19 weight layers
input (224 × 224 RGB image)					
conv3-64	conv3-64 **LRN**	conv3-64 **conv3-64**	conv3-64 conv3-64	conv3-64 conv3-64	conv3-64 conv3-64
maxpool					
conv3-128	conv3-128	conv3-128 **conv3-128**	conv3-128 conv3-128	conv3-128 conv3-128	conv3-128 conv3-128
maxpool					
conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256	conv3-256 conv3-256 **conv1-256**	conv3-256 conv3-256 **conv3-256**	conv3-256 conv3-256 conv3-256 **conv3-256**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512	conv3-512 conv3-512 **conv1-512**	conv3-512 conv3-512 **conv3-512**	conv3-512 conv3-512 conv3-512 **conv3-512**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

3. GoogLeNet





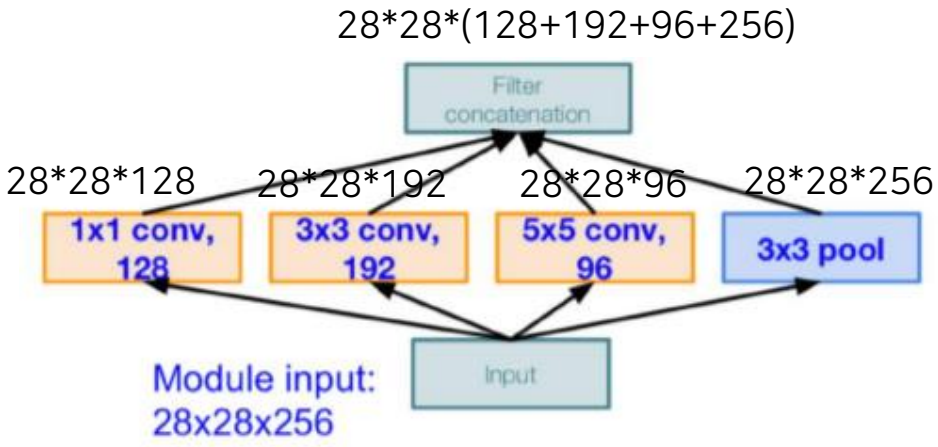
(a) Inception module, naïve version



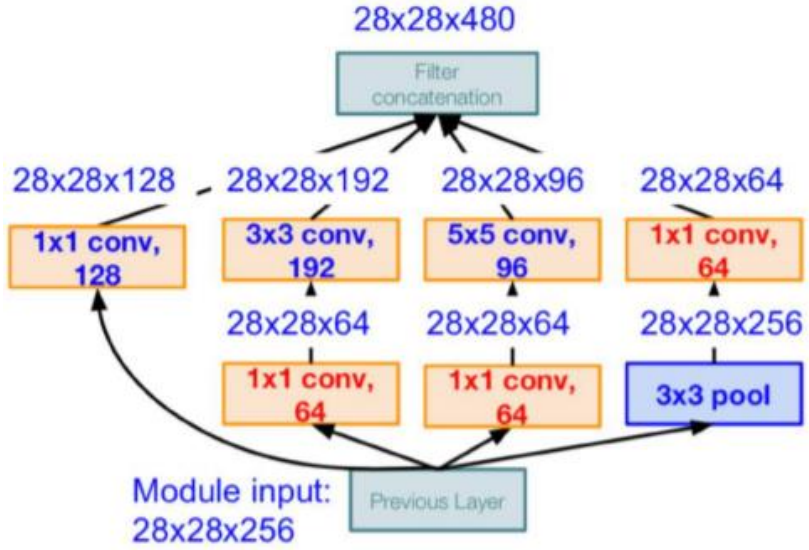
(b) Inception module with dimensionality reduction

3. GoogLeNet

* Inception Module



Naive Inception module



Inception module with dimension reduction

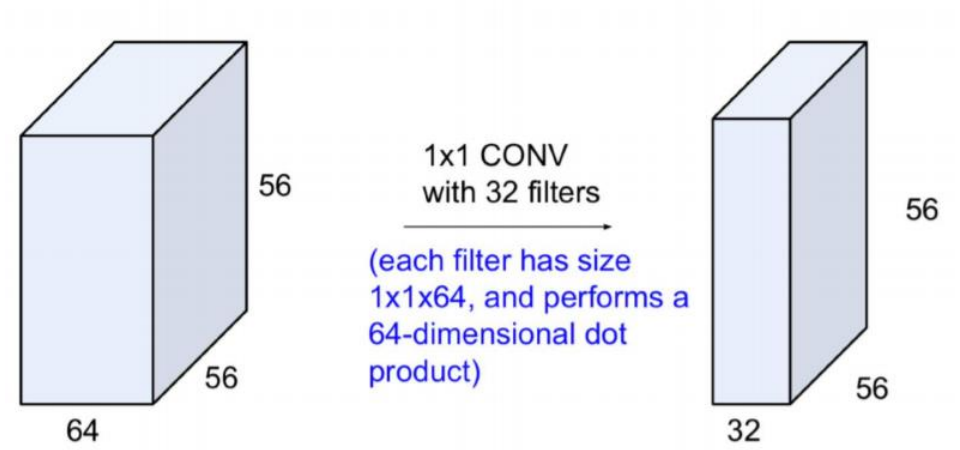
Conv Ops:
[1x1 conv, 128] 28x28x128x1x1x256
[3x3 conv, 192] 28x28x192x3x3x256
[5x5 conv, 96] 28x28x96x5x5x256
Total: 854M ops

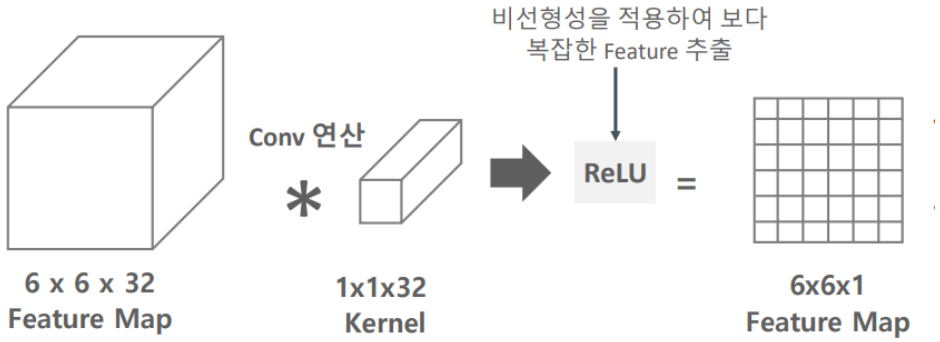
Very expensive compute

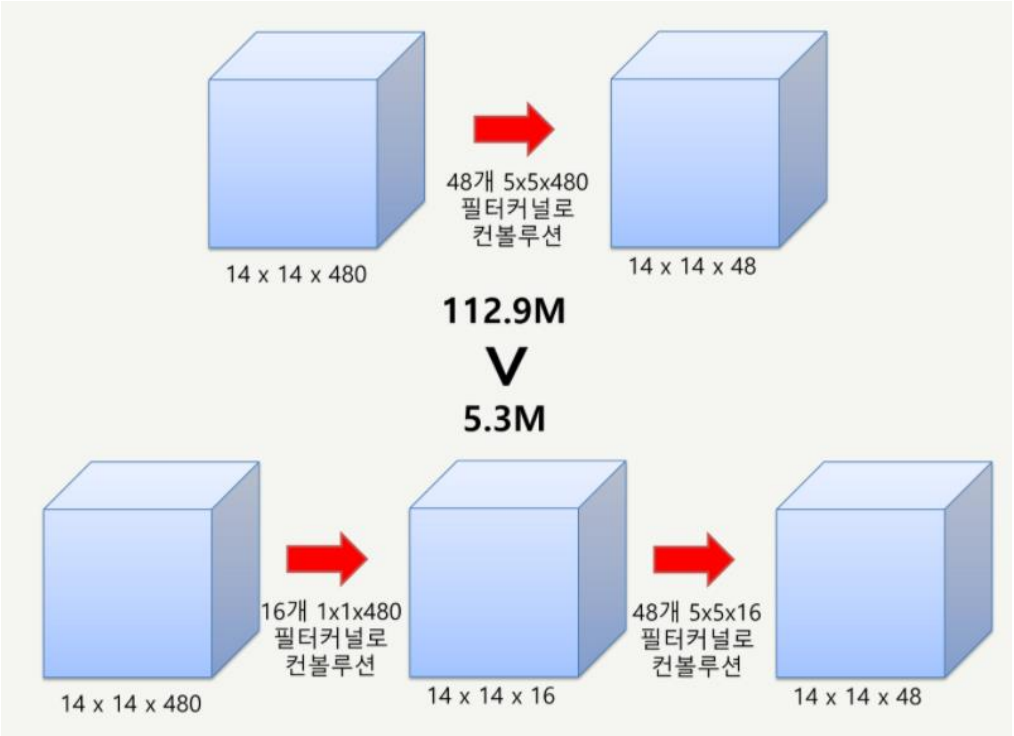
Conv Ops:
[1x1 conv, 64] 28x28x64x1x1x256
[1x1 conv, 64] 28x28x64x1x1x256
[1x1 conv, 128] 28x28x128x1x1x256
[3x3 conv, 192] 28x28x192x3x3x64
[5x5 conv, 96] 28x28x96x5x5x64
[1x1 conv, 64] 28x28x64x1x1x256
Total: 358M ops

3. GoogLeNet

* 1 X 1 convolutions







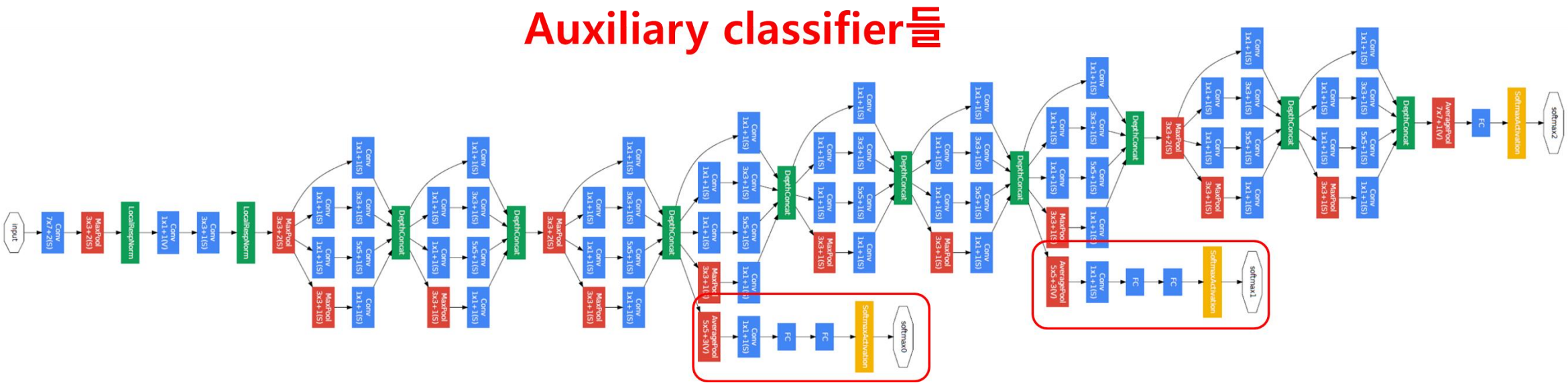
필요한 연산 횟수

(14 x 14 x 48) x (5 x 5 x 480) = 약 112.9M

(14 x 14 x 16)*(1 x 1 x 480) + (14 x 14 x 48)*(5 x 5 x 16) = 약 5.3M

3. GoogLeNet

* Auxiliary classifier



네트워크의 깊이가 깊어질수록 그래디언트 소실 문제를 피하기 어려워짐

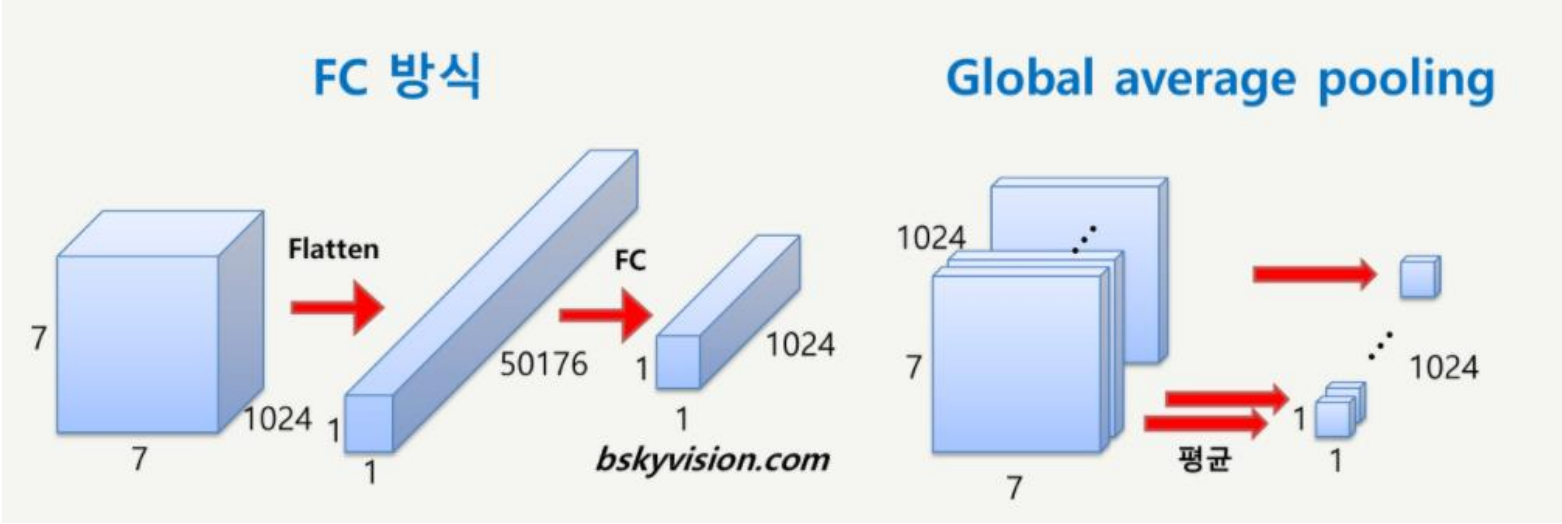
가중치를 훈련하는 과정에 역전파를 주로 활용 -> 역전파 과정에서 가중치를 업데이트하는데 사용되는 그래디언트가

점점 작아져서 0이 되어버리는 것 -> 따라서 네트워크 내의 가중치들이 제대로 훈련되지 않음

➔ 해결방법 : 보조 분류기(auxiliary classifier)

3. GoogLeNet

* Global average pooling



AlexNet, VGGNet 등 -> Fully Connected (FC) 층들이 후반부에 연결

GoogLeNet -> FC 방식 대신에 global average pooling 방식 사용

* Global average pooling : 전 층에서 산출된 특성맵들을 각각 평균낸 것을 이어서 1차원 벡터를 만들어 주는 방식

** 1차원 벡터로 만들어주는 이유 : 최종적으로 이미지 분류를 위한 softmax층을 연결해주기 위해

** 장점 : 가중치의 개수를 상당히 많이 없애줄 수 있음

FC) 7x7x1024x1024=51.3M / Global average pooling) 가중치 필요없음

감사합니다