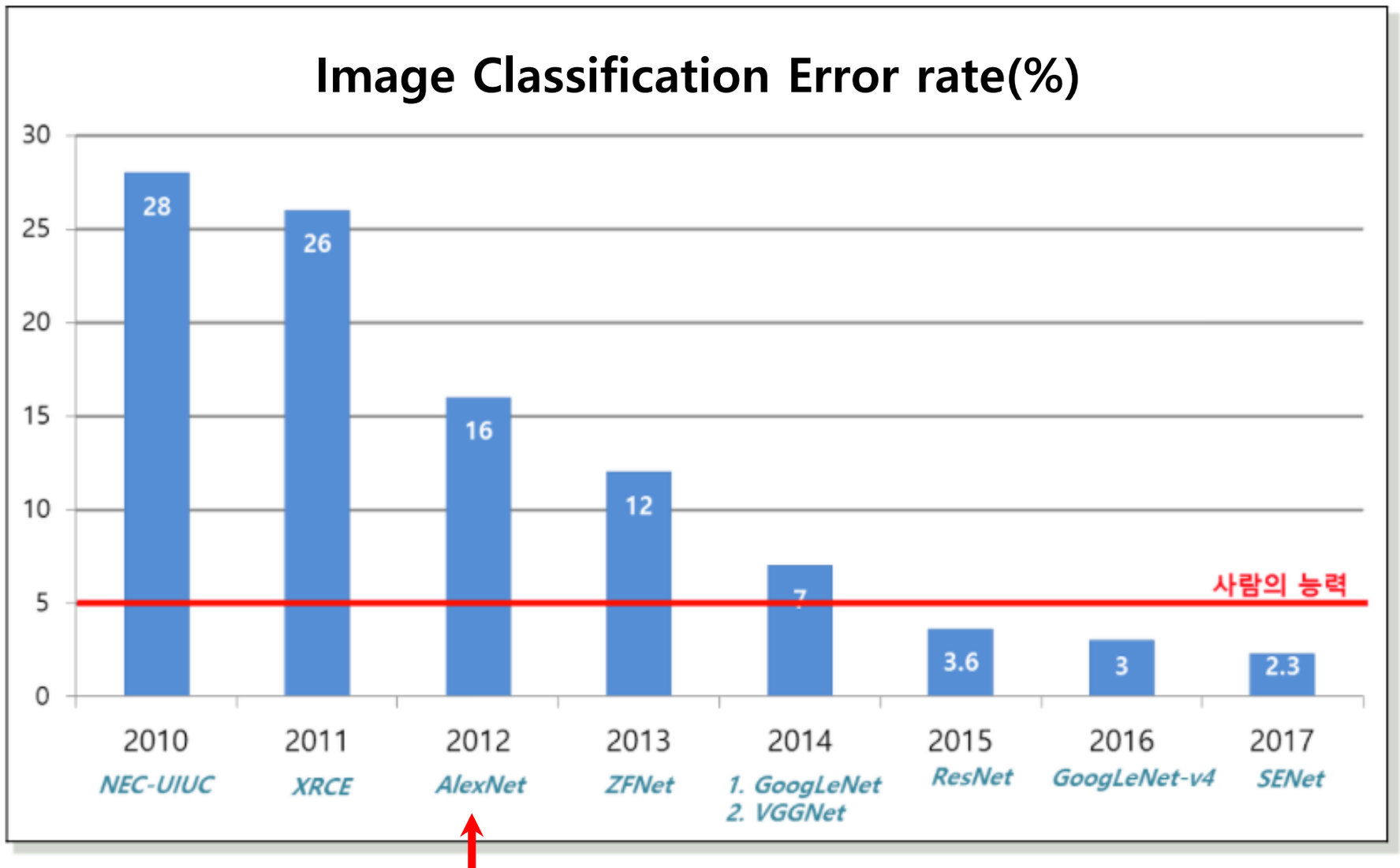
인공지능 심화 과정

ILSVRC 대표적인 분류 모델 I

ILSVRC 역대 알고리즘 성능 비교 – AlexNet



첫번째 딥러닝 모델, 처음 ReLU를 사용, 처음 GPU를 사용,

Overfitting을 줄이기 위해 -> Dropout, Data Augmentation 적용

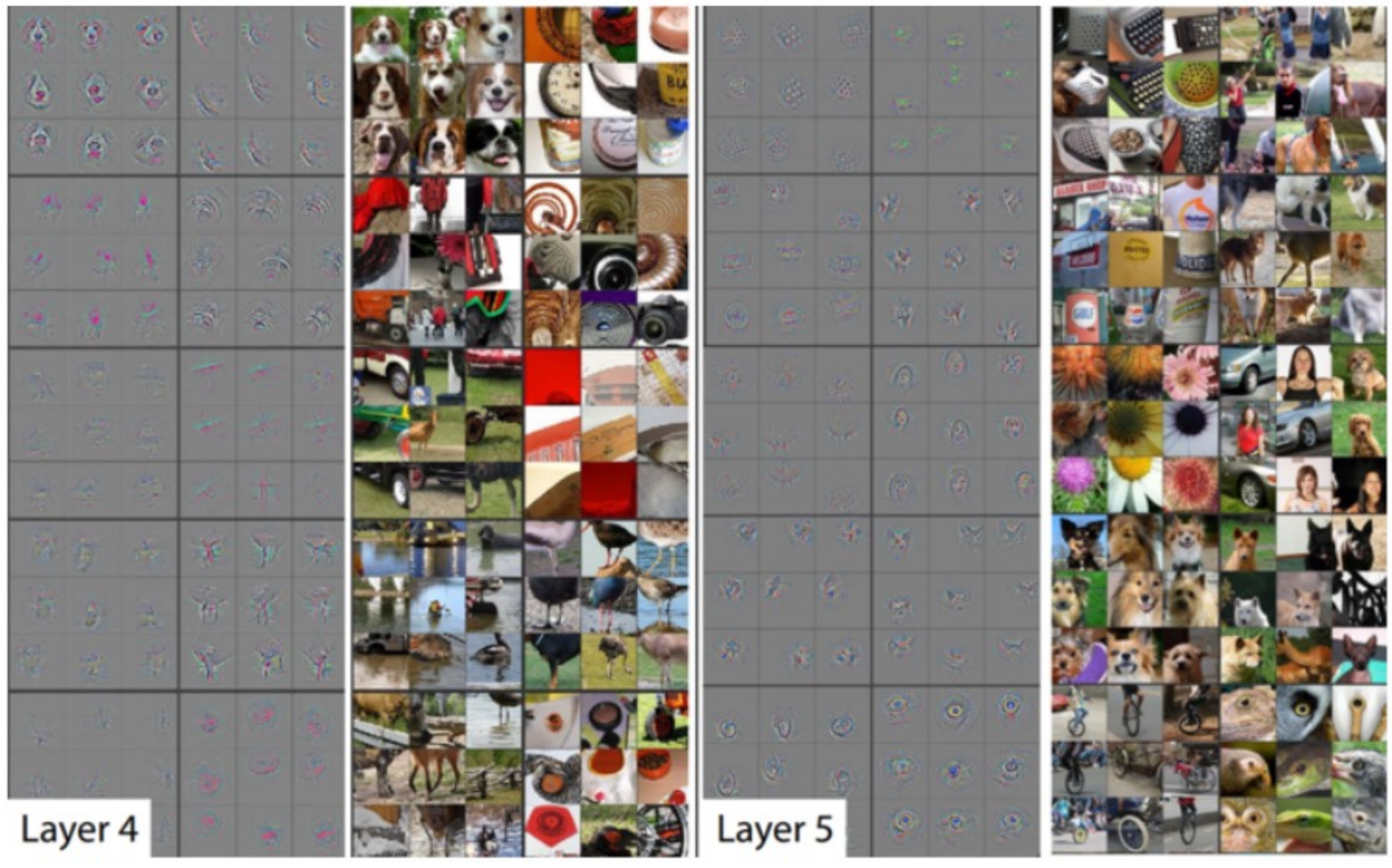
원문 : https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

번역 : https://naknaklee.github.io/classification/2020/04/22/AlexNet-Post-Review/

AlexNet VS ZFNet

ZFNet의 구조자체는 AlexNet애서 GPU를 하나만 쓰고, 일부 Convolution레이어의 Kernel 사이즈와 stride를 일부 조절한 것 뿐이다.

ZFNet의 논문의 핵심은, ZFNet의 구조 자체보다 CNN을 시각화 하여 CNN의 중간 과정을 눈으로 보고 개선 방향을 파악할 방법을 만든 것이다. **(다음 페이지 참조)**



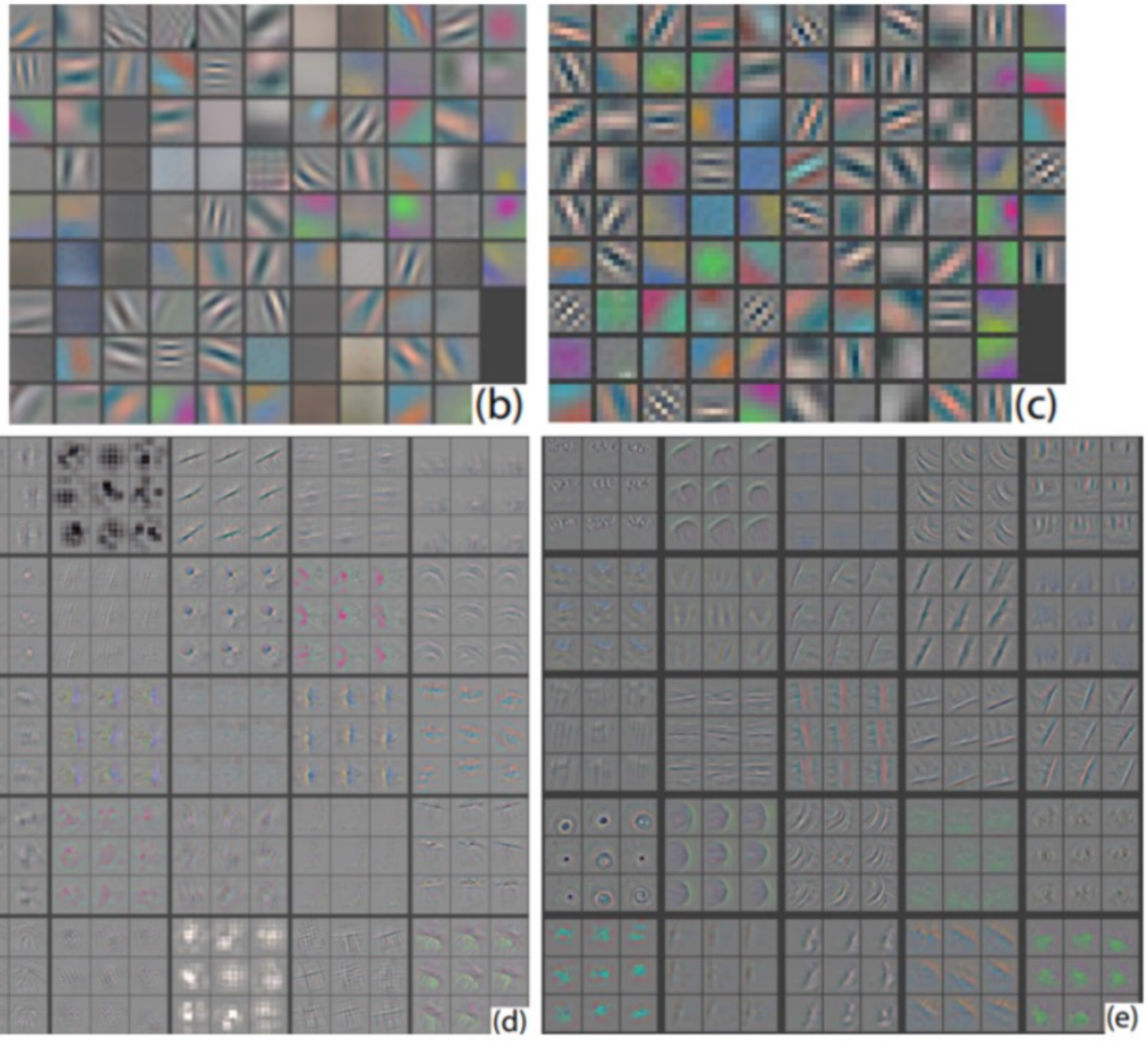
ZFNet

Abstract

Large Convolutional Network models have recently demonstrated impressive classification performance on the ImageNet benchmark (Krizhevsky et al., 2012). However there is no clear understanding of why they perform so well, or how they might be improved. In this paper we address both issues. We introduce a novel visualization technique that gives insight into the function of intermediate feature layers and the operation of the classifier. Used in a diagnostic role, these visualizations allow us to find model architectures that outperform Krizhevsky *et al.* on the ImageNet classification benchmark. We also perform an ablation study to discover the performance contribution from different model layers. We show our ImageNet model generalizes well to other datasets: when the softmax classifier is retrained, it convincingly beats the current state-of-the-art results on Caltech-101 and Caltech-256 datasets.

ZFNet : https://arxiv.org/pdf/1311.2901.pdf

AlexNet VS ZFNet

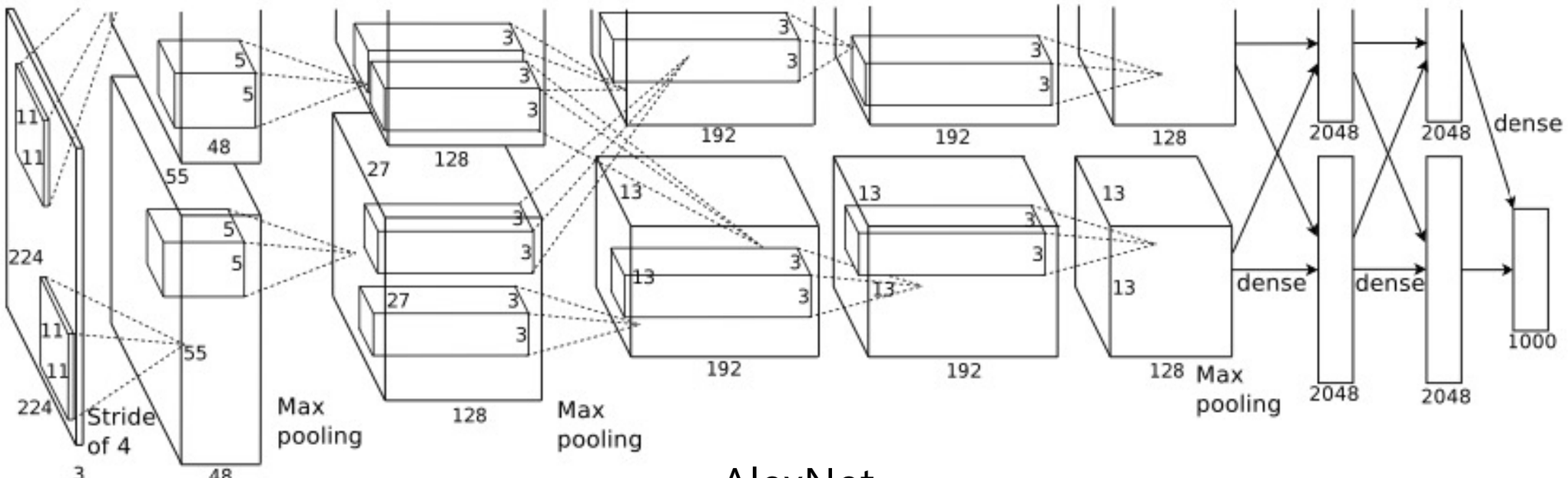


AlexNet VS ZFNet

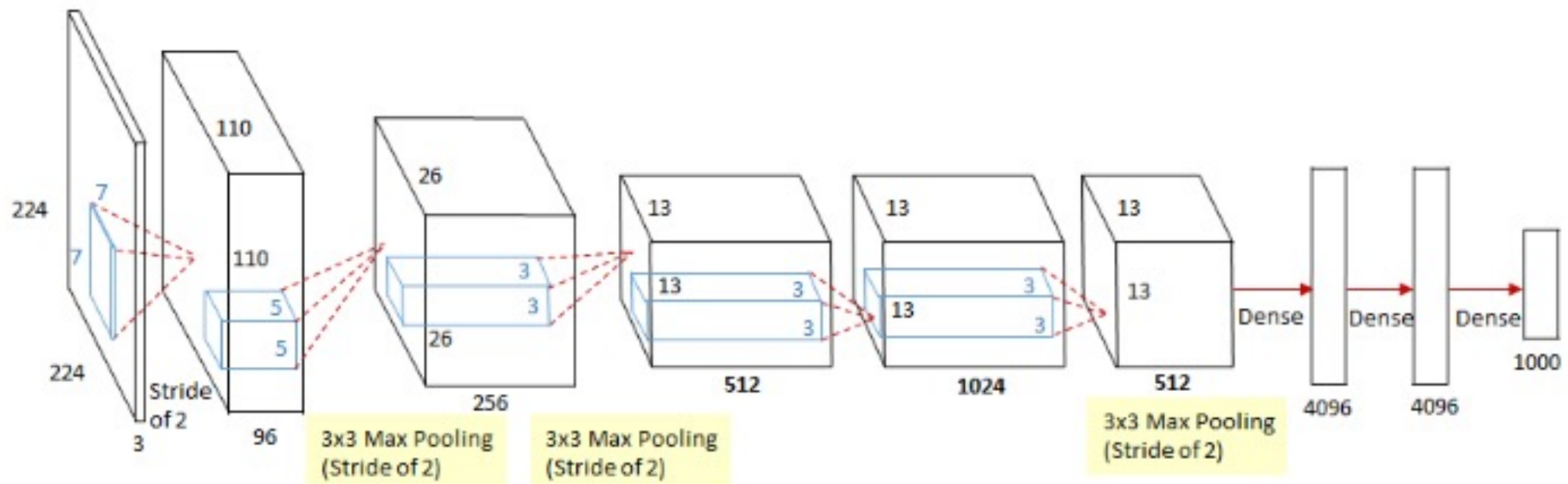
AlexNet을 시각화 하면 위와 같다. ZFNet의 결과가 AlexNet의 결과보다 다양한 Feature를 표현하고 있으며, aliasing현상도 덜하다. ZFNet의 연구진은 AlexNet이 첫번째 convolution Layer의 필터들이 극단적인 고주파 정보, 저주파 정보만을 남겨 중간 주파 정보가 남아 있지 않는 것이 문제이며, 두번째 convolution layer는 첫번째 레이어의 convolution의 stride를 4로 설정한 것이 aliasing문제를 일으키는 것이라고 해석했다.그래서 11x11을 7x7로 수정하고 Stride를 1로 수정하고, 두번째 레이어의 stride를1에서 2로 수정했다.

이와같이 이전까지는 CNN의 학습 과정을 살펴보기 어려웠지만, ZFNet의 시각화기법으로 학습과정을 눈으로 보고 파악할 수 있게 되었다.

AlexNet VS ZFNet



AlexNet



ZFNet

GoogleNet

CNN의 성능 향상 기법

CNN의 성능을 향상시키기는 가장 직접적인 방식은 망의 크기를 늘리는 것이다.
망의 크기를 늘린다는 것은 단순하게 망의 레이어 수(depth)를 늘리는 것 뿐만 아니라, 각 레이어에 있는 유닛의 수(width)도 늘리는 것을 의미한다.
특히 이미지넷과 같은 대용량 데이터를 이용한 학습의 경우 필수적이다.

깊은 망의 부작용

망이 깊어지면 성능이 높아지지만 2가지 중대한 문제가 발생한다.

1. 망이 깊어질수록 파라메터의 수가 증가한다. 이는 학습에 사용할 데이터 양이 제한적일 경우 오버피팅이 발생할 가능성이 존재한다.
2. 망의 크기가 커지면 연산량이 증가한다.

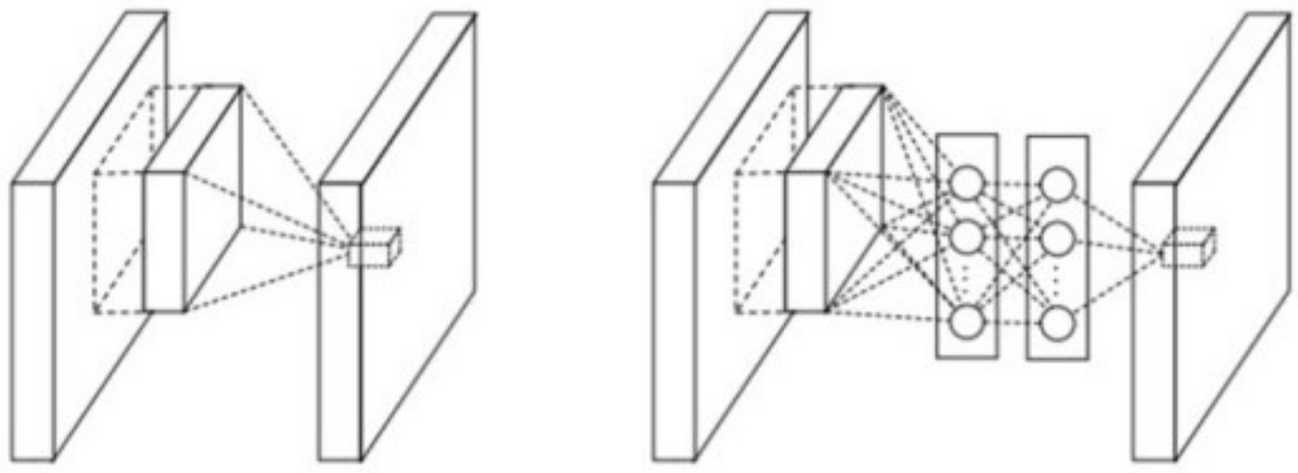
NIN(Network-In-Network)

GoogleNet에서는 망의 깊이와 넓이가 모두 커지고, 중간에 분기되는 부분이 있다.
Inception Module이 등장하는데, 이는 이전에 발표된 Network-in-Network논문의 영향을 받은 것이다.

Network-in-Network : https://arxiv.org/pdf/1312.4400.pdf

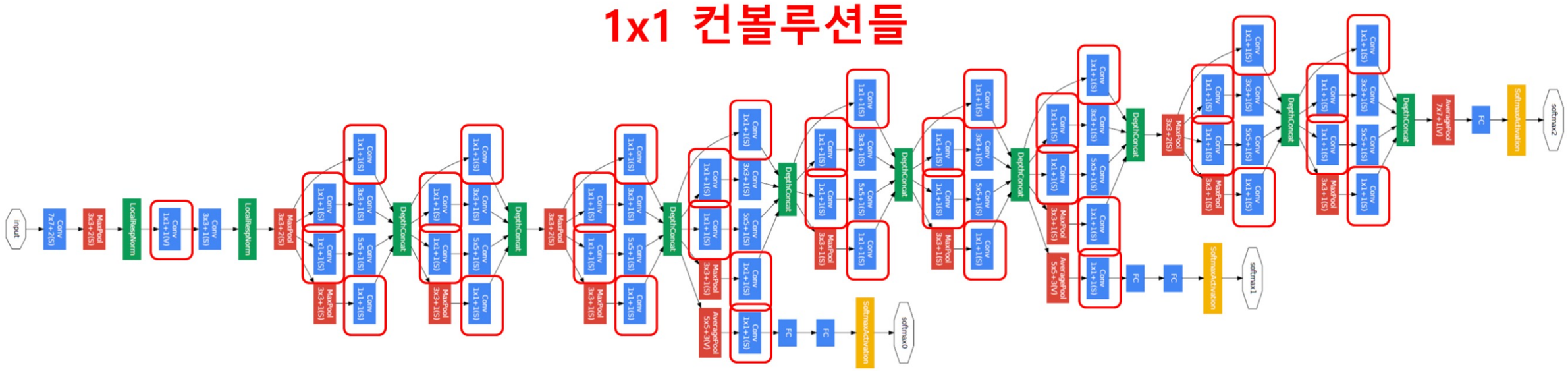
GoogleNet

Network-In-Network 설계자는CNN의 convolution layer가 local receptive field에서 Feature를 추출해내는 능력은 우수하지만 필터의 특징이 선형적이기 때문에 비선형적인 성질을 갖는 feature를 추출하는데 어려움이 있다고 한다. 이 부분을 개선하기 위해서 feature의 갯수를 늘려야 하는 문제에 주목했다. Local receptive filed안에서 좀 더 feature를 잘 추출하는 방법이 micro neural network이다. 이들은 convolution 대신에 MLP(Multi layer perceptron)를 사용하여 feature를 추출할 수 있도록 했다.



(a) Linear convolution layer (b) Mlpconv layer

GoogleNet

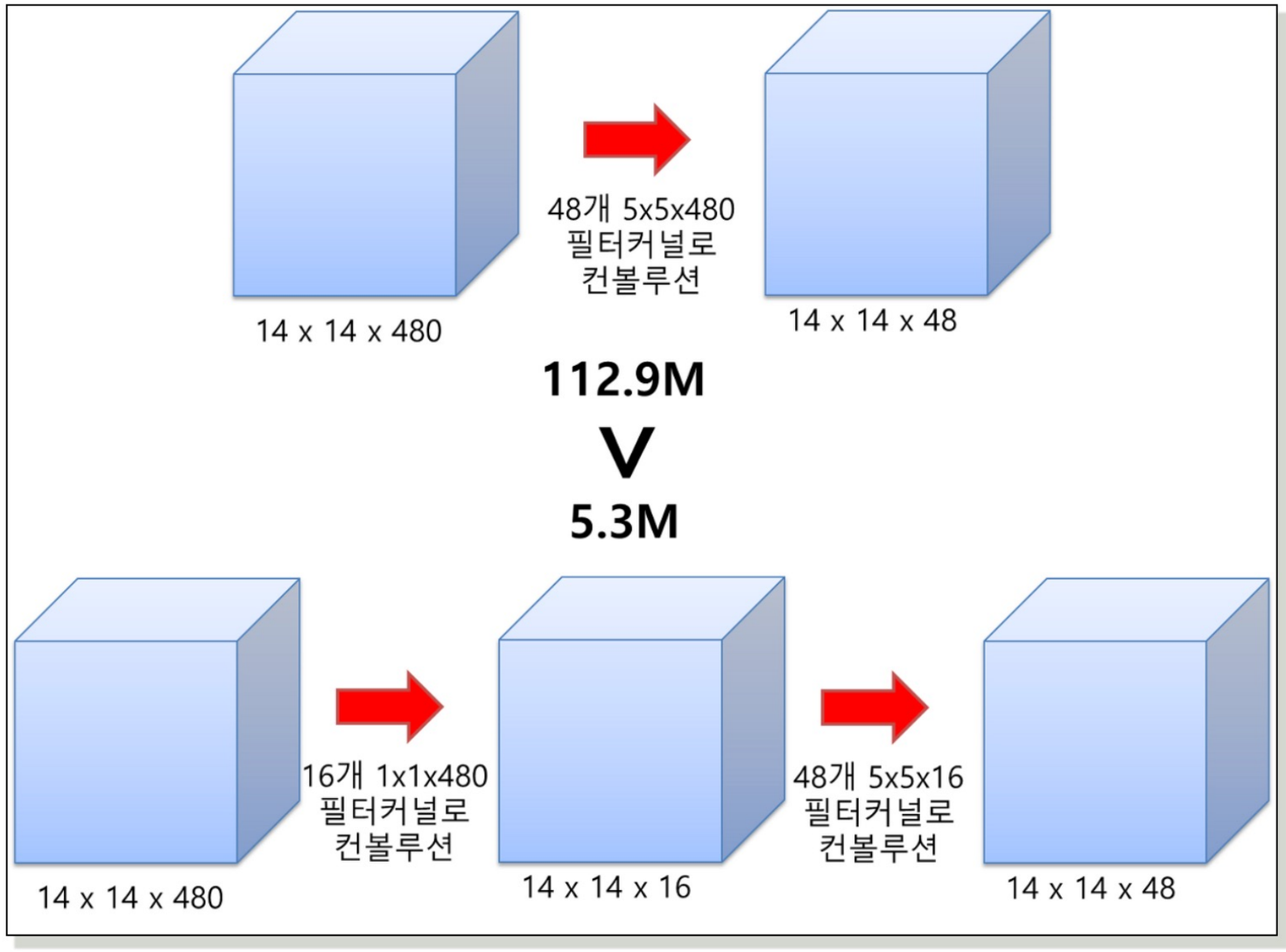


GoogLeNet의 구조

GoogLeNet에서 **1 x 1 컨볼루션은 특성맵의 갯수를 줄이는 목적으로 사용된다.**

특성맵의 갯수가 줄어들면 그만큼 연산량이 줄어든다.

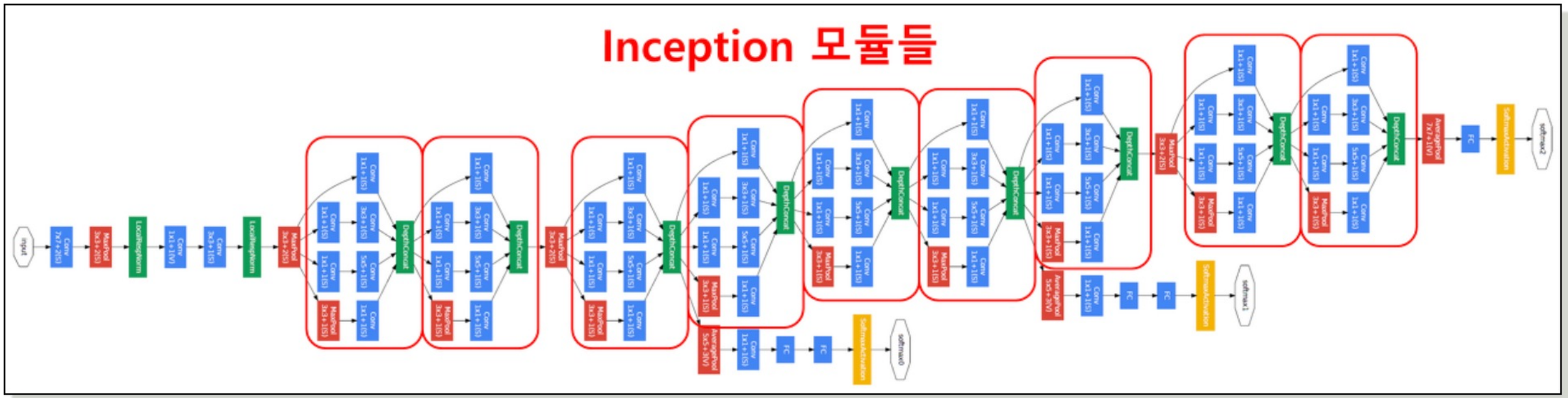
GoogleNet



GoogLeNet에서 **1 x 1 컨볼루션은 특성맵의 갯수를 줄이는 목적으로 사용된다.**

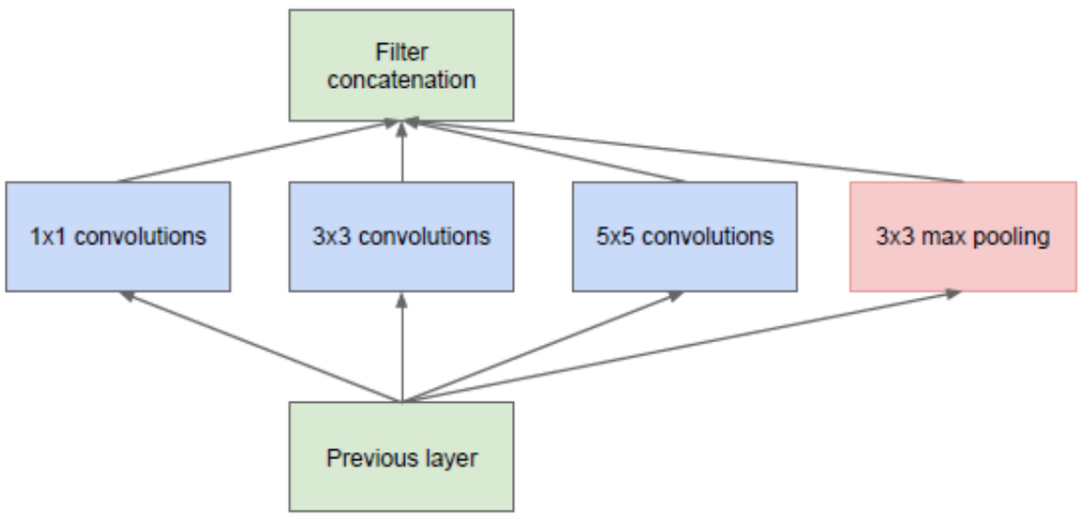
특성맵의 갯수가 줄어들면 그만큼 연산량이 줄어든다.

GoogleNet

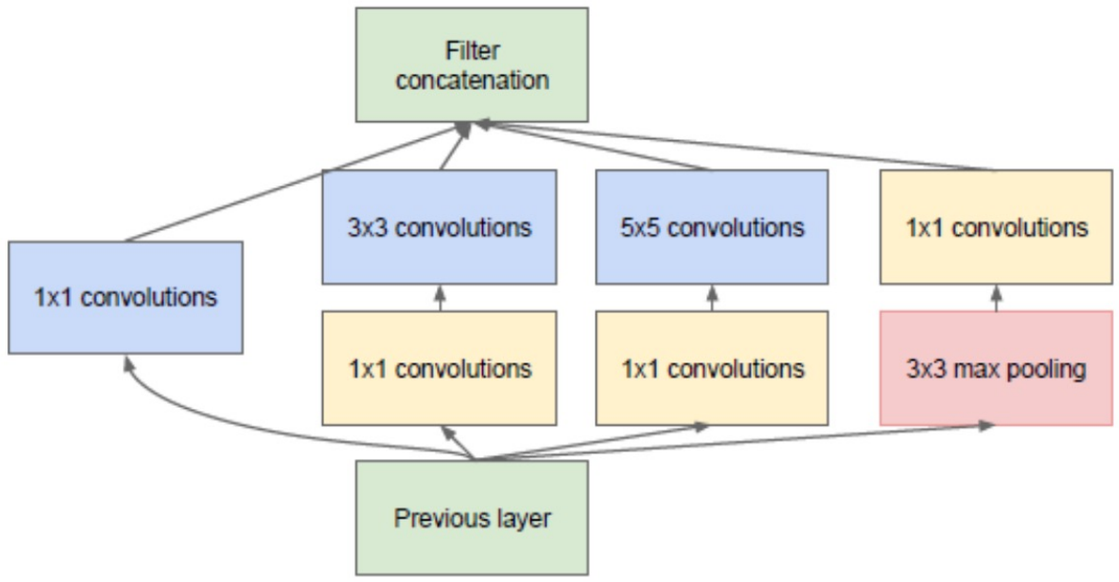


GoogLeNet은 총 9개의 인셉션 모듈을 포함하고 있다. 인셉션 모듈을 하나 확대해서 자세히 살펴보자.

GoogleNet



(a) Inception module, naïve version

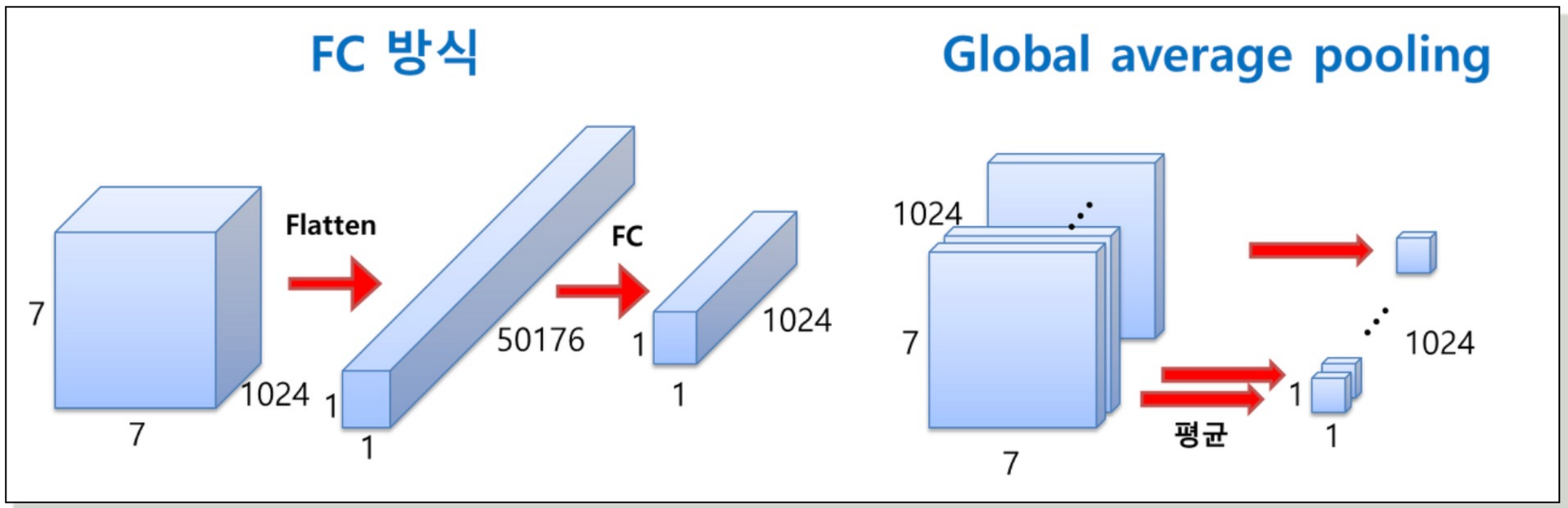


(b) Inception module with dimensionality reduction

GoogLeNet에 실제로 사용된 모듈은 1x1 컨볼루션이 포함된 (b) 모델이다. 아까 살펴봤듯이 1x1 컨볼루션은 특성맵의 장수를 줄여주는 역할을 한다. 노란색 블럭으로 표현된 1x1 컨볼루션을 제외한 나이브(naive) 버전을 살펴보면, 이전 층에서 생성된 특성맵을 1x1 컨볼루션, 3x3 컨볼루션, 5x5 컨볼루션, 3x3 최대풀링해준 결과 얻은 특성맵들을 모두 함께 쌓아준다. AlexNet, VGGNet 등의 이전 CNN 모델들은 한 층에서 동일한 사이즈의 필터커널을 이용해서 컨볼루션을 해줬던 것과 차이가 있다. 따라서 **좀 더 다양한 종류의 특성이 도출된다**. 여기에 1x1 컨볼루션이 포함되었으니 당연히 연산량은 많이 줄어들었을 것이다.

GoogleNet

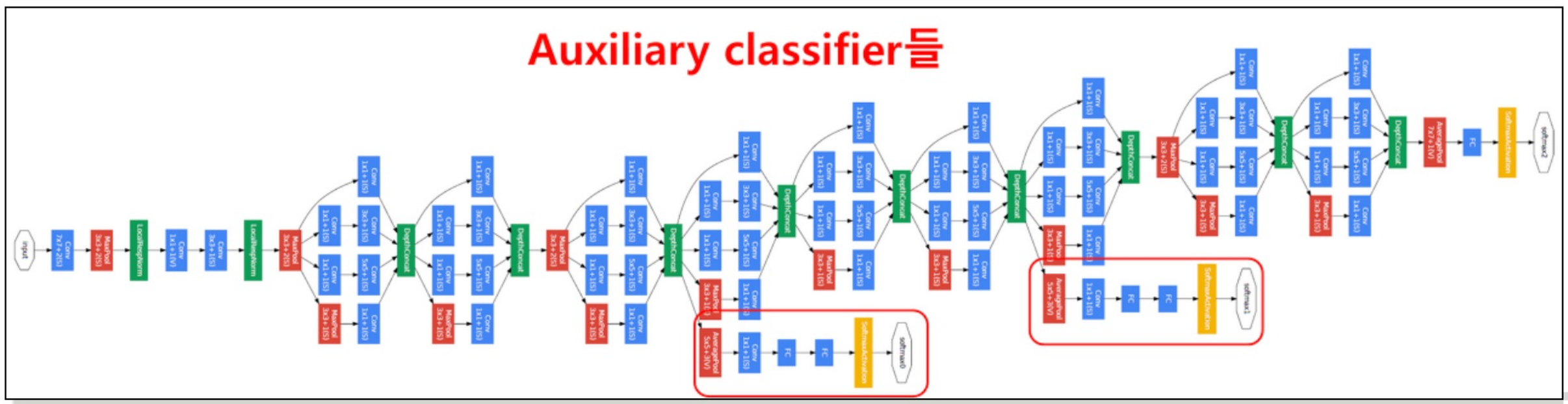
AlexNet, VGGNet 등에서는 fully connected (FC) 층들이 망의 후반부에 연결되어 있다. 그러나 GoogLeNet은 FC 방식 대신에 global average pooling이란 방식을 사용한다. global average pooling은 전 층에서 산출된 특성 맵들을 각각 평균낸 것을 이어서 1차원 벡터를 만들어주는 것이다. 1차원 벡터를 만들어줘야 최종적으로 이미지 분류를 위한 softmax 층을 연결해줄 수 있기 때문이다. 만약 전 층에서 1024장의 7 x 7의 특성맵이 생성되었다면, 1024장의 7 x 7 특성맵 각각 평균을 구하여 1024개의 값을 하나의 벡터로 연결해주는 것이다.



GoogleNet

이렇게 해줌으로 얻을 수 있는 장점은 가중치의 갯수를 상당히 많이 없애준다는 것이다. 만약 FC 방식을 사용한다면 훈련이 필요한 가중치의 갯수가 7 x 7 x 1024 x 1024 = 51.3M이지만 **global average pooling을 사용하면 가중치가 단 한 개도 필요하지 않다.**

네트워크의 깊이가 깊어지면 깊어질수록 vanishing gradient 문제를 피하기 어려워진다. 그러니까 가중치를 훈련하는 과정에 역전파(back propagation)를 주로 활용하는데, 역전파과정에서 가중치를 업데이트하는데 사용되는 gradient가 점점 작아져서 0이 되어버리는 것이다. 따라서 네트워크 내의 가중치들이 제대로 훈련되지 않는다. 이 문제를 극복하기 위해서 GoogLeNet에서는 네트워크 중간에 두 개의 보조 분류기(auxiliary classifier)를 달아주었다.



VGGNet

VGGNet의 original 논문의 개요에서 밝히고 있듯이 이 연구의 핵심은 네트워크의 깊이를 깊게 만드는 것이 성능에 어떤 영향을 미치는지를 확인하고자 한 것이다. VGG 연구팀은 깊이의 영향만을 최대한 확인하고자 컨볼루션 필터커널의 사이즈는 가장 작은 3 x 3으로 고정했다.

VGG 연구팀은 original 논문에서 총 6개의 구조(A, A-LRN, B, C, D, E)를 만들어 성능을 비교했다. 여러 구조를 만든 이유는 기본적으로 깊이의 따른 성능 변화를 비교하기 위함이다. 이중 D 구조가 VGG16이고 E 구조가 VGG19라고 보면 된다.

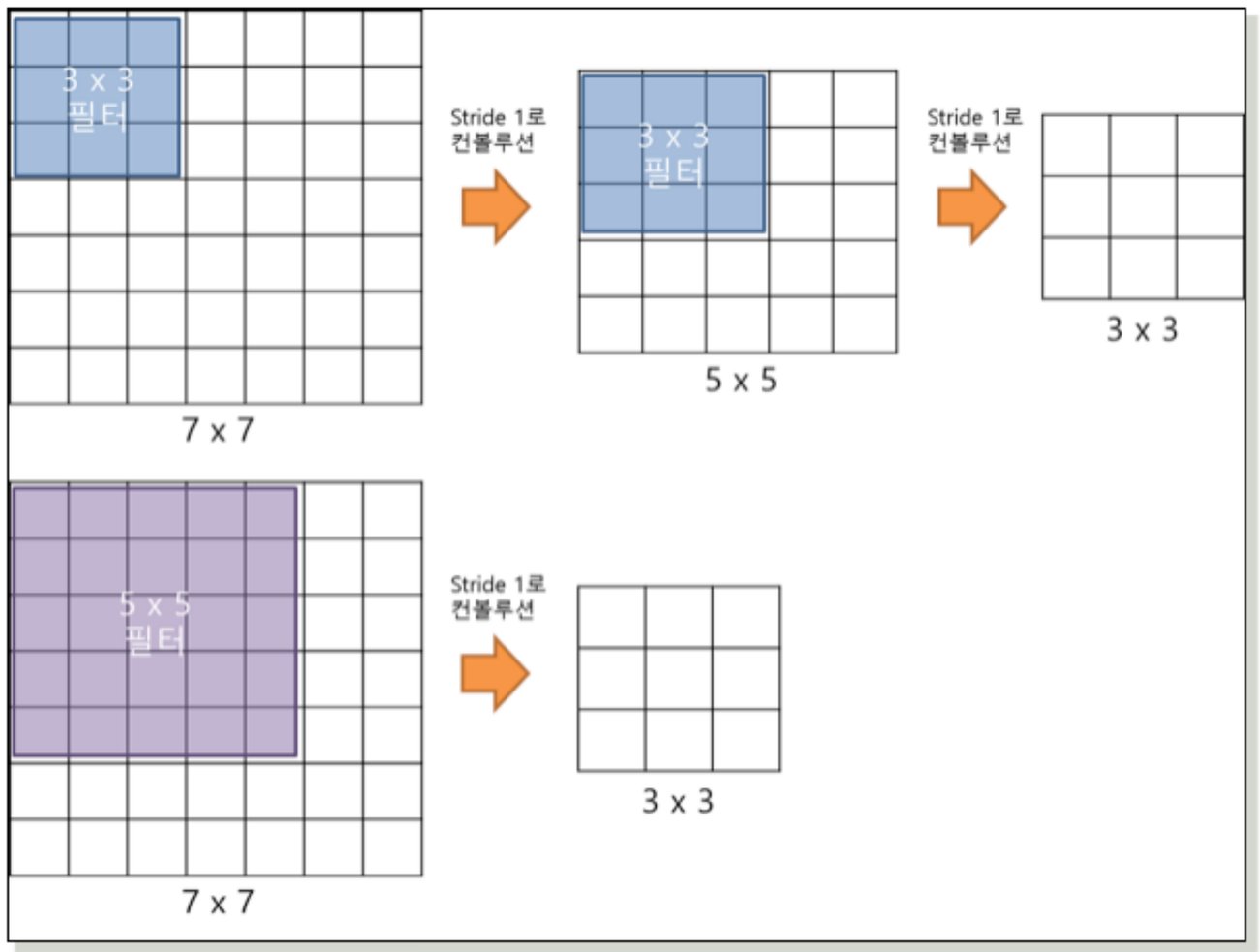
VGGNet

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**
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**
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**
maxpool					
FC-4096					
FC-4096					
FC-1000					
soft-max					

VGG 연구팀은 AlexNet과 VGG-F, VGG-M, VGG-S에서 사용되던 Local Response Normalization(LRN)이 A 구조와 A-LRN 구조의 성능을 비교함으로 성능 향상에 별로 효과가 없다고 실험을 통해 확인했다. 그래서 더 깊은 B, C, D, E 구조에는 LRN을 적용하지 않는다고 논문에서 밝혔다. 또한 그들은 깊이가 11층, 13층, 16층, 19층으로 깊어지면서 분류 에러가 감소하는 것을 관찰했다. 즉, 깊어질수록 성능이 좋아진다.

VGGNet

VGGNet의 구조를 깊이 들여다보기에 앞서 먼저 집고 넘어가야할 것이 있다. 그것은 바로 3 x 3 필터로 두 차례 컨볼루션을 하는 것과 5 x 5 필터로 한 번 컨볼루션을 하는 것이 결과적으로 동일한 사이즈의 특성맵을 산출한다는 것이다(아래 그림 참고). 3 x 3 필터로 세 차례 컨볼루션 하는 것은 7 x 7 필터로 한 번 컨볼루션 하는 것과 대응된다.



VGGNet

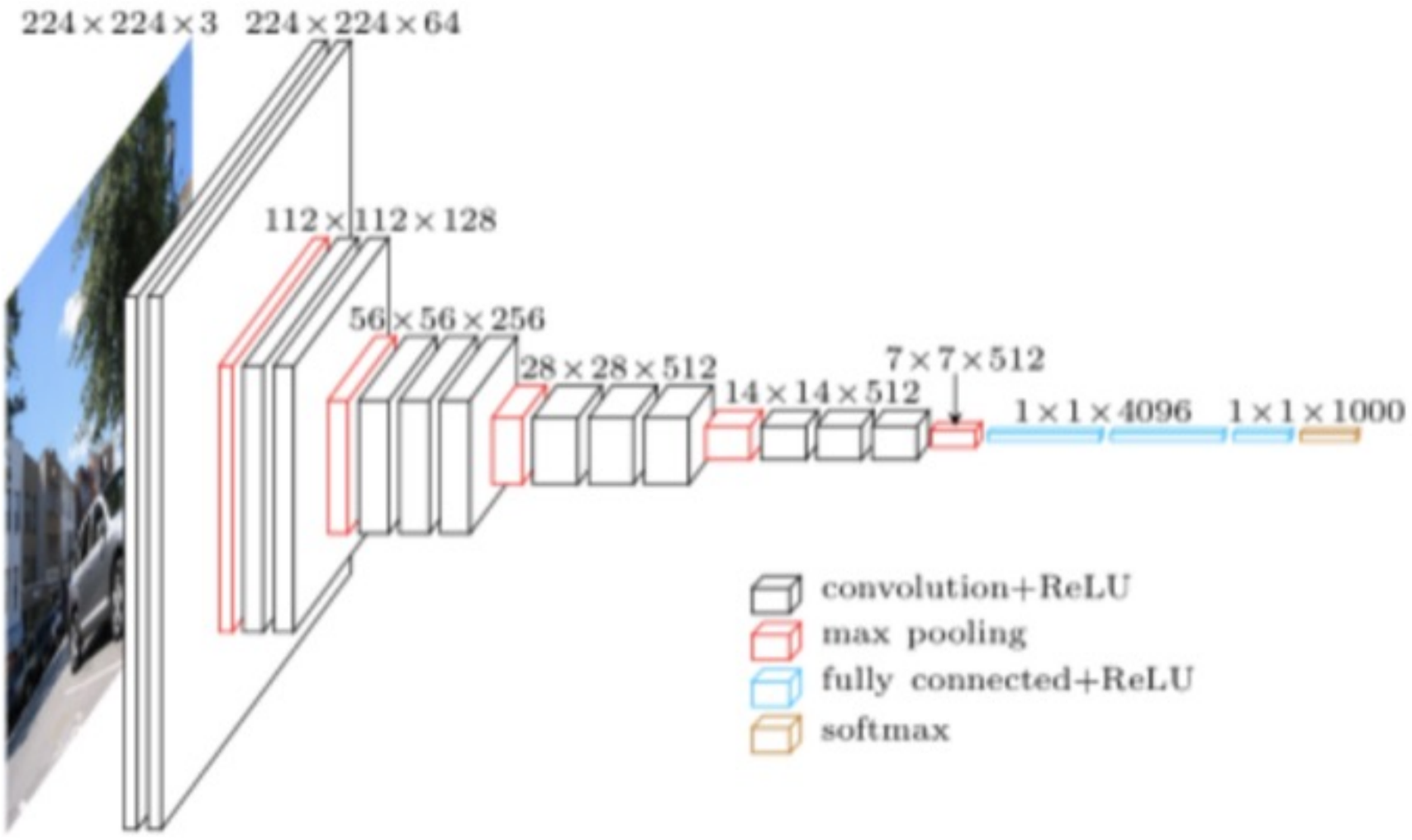
3 × 3 합성곱 계층을 3개 연이어 배치하면 수용 영역은 7 × 7이 되고 5개의 3 × 3 합성곱의 수용 영역(receptive field)은 11 × 11이 된다. 따라서 AlexNet의 필터는 11 × 11까지로 규모가 크지만, VGG 네트워크는 이보다 작은 합성곱 계층을 더 많이 포함해 더 큰 유효 수용 영역을 얻을 수 있다. 이러한 변경 내역에는 두 가지 이점이 있다.

파라미터의 수가 줄어든다. 3 x 3 필터가 3개면 총 27개의 가중치를 갖는다. 반면 **7 x 7 필터는 49개의 가중치를 갖는다**. CNN에서 가중치는 모두 훈련이 필요한 것들이므로, 가중치가 적다는 것은 그만큼 훈련시킬 파라메터의 수가 줄어든다. 따라서 학습의 속도가 빨라진다. 동시에 레이어의 depth가 늘어나면서 특성에 비선형성을 더 증가시키기 때문에 특성이 더 좋아진다.

비선형성을 증가시킨다: 합성곱 레이어의 수가 늘어나면, 각 합성곱 계층 다음에 ReLU 같은 '비선형' 활성화 함수를 통해 네트워크가 복잡한 특징을 학습할 수 있는 능력이 증대된다.

VGGNet

https://arxiv.org/pdf/1409.1556.pdf



https://velog.io/@dyckjs30/VggNet