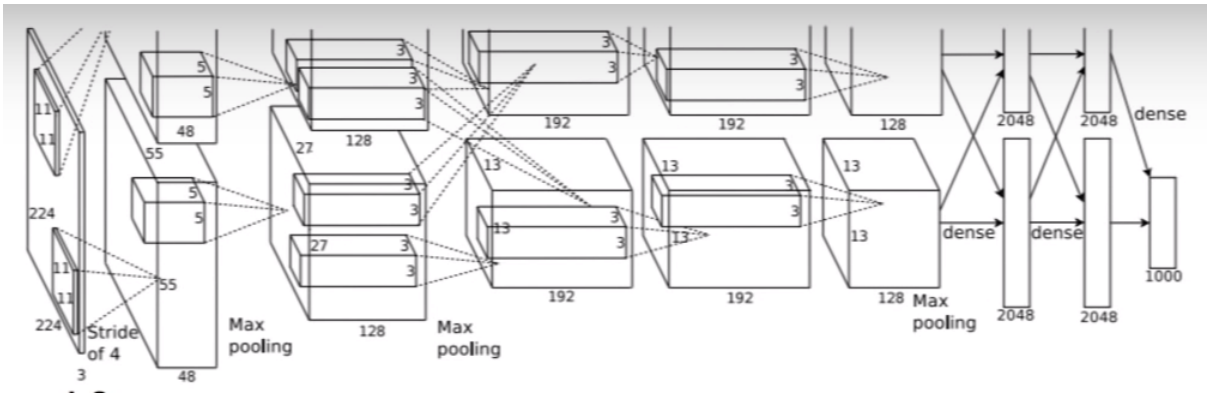
Week9 예습과제

AlexNet

2012년에 ImageNet을 이용한 이미지 인식 대회인 ILSVRC-2012의 우승 모델이다.



- 기본 구조는 [Conv-Max pooling-(normalization)] x 2 - CONV-CONV-CONV - Max pooling - [FC-FC-FC]
- 5개의 CONV layer와 3개의 FC layer 그리고 사이의 pooling layer로 LeNet과 크게 다르지 않다.
- 다만 GPU를 두개로 나눠서 학습했다는게 일단은 큰 차이점이다.

Full (simplified) AlexNet architecture:
[227x227x3] INPUT
[55x55x96] CONV1: 96 11x11 filters at stride 4, pad 0
[27x27x96] MAX POOL1: 3x3 filters at stride 2
[27x27x96] NORM1: Normalization layer
[27x27x256] CONV2: 256 5x5 filters at stride 1, pad 2
[13x13x256] MAX POOL2: 3x3 filters at stride 2
[13x13x256] NORM2: Normalization layer
[13x13x384] CONV3: 384 3x3 filters at stride 1, pad 1
[13x13x384] CONV4: 384 3x3 filters at stride 1, pad 1
[13x13x256] CONV5: 256 3x3 filters at stride 1, pad 1
[6x6x256] MAX POOL3: 3x3 filters at stride 2
[4096] FC6: 4096 neurons
[4096] FC7: 4096 neurons
[1000] FC8: 1000 neurons (class scores)

이 모델이 처음 발표될 당시에는 컴퓨터 성능이 좋지 않아 네트워크를 분산시켜서 GPU에 할당했고, 그래서 feature map을 추출할 때 2개의 영역으로 나뉘어있다.

- ReLU를 처음으로 사용
- Norm Layer를 사용
- Data Augmentation을 많이 사용
- Dropout = 0.5
  Batch Size = 128
  SGD Momentum: 0.9
  Learning rate: 1e-2
  L2 weight decay: 5e-4
  7 CNN ensemble: 18.2% → 15.4%

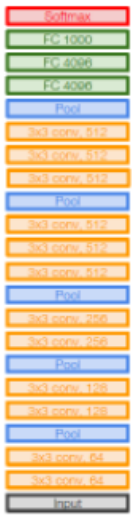
VGG

AlexNet에 비해 Layer의 수가 더 깊어진 것이다.

3x3 convolution layer 필터만을 사용해 파라미터의 수도 더 적게 되었다. VGG 모델은 VGG16, VGG19가 대표적이며 각각 layer의 갯수에 따라 이름 붙어졌다.

```
INPUT: [224x224x3]        memory:  224*224*3=150K   params: 0       (not counting biases)
CONV3-64: [224x224x64]  memory:  224*224*64=3.2M   params: (3*3*3)*64 = 1,728
CONV3-64: [224x224x64]  memory:  224*224*64=3.2M   params: (3*3*64)*64 = 36,864
POOL2: [112x112x64]  memory:  112*112*64=800K   params: 0
CONV3-128: [112x112x128]  memory:  112*112*128=1.6M   params: (3*3*64)*128 = 73,728
CONV3-128: [112x112x128]  memory:  112*112*128=1.6M   params: (3*3*128)*128 = 147,456
POOL2: [56x56x128]  memory:  56*56*128=400K   params: 0
CONV3-256: [56x56x256]  memory:  56*56*256=800K   params: (3*3*128)*256 = 294,912
CONV3-256: [56x56x256]  memory:  56*56*256=800K   params: (3*3*256)*256 = 589,824
CONV3-256: [56x56x256]  memory:  56*56*256=800K   params: (3*3*256)*256 = 589,824
POOL2: [28x28x256]  memory:  28*28*256=200K   params: 0
CONV3-512: [28x28x512]  memory:  28*28*512=400K   params: (3*3*256)*512 = 1,179,648
CONV3-512: [28x28x512]  memory:  28*28*512=400K   params: (3*3*512)*512 = 2,359,296
CONV3-512: [28x28x512]  memory:  28*28*512=400K   params: (3*3*512)*512 = 2,359,296
POOL2: [14x14x512]  memory:  14*14*512=100K   params: 0
CONV3-512: [14x14x512]  memory:  14*14*512=100K   params: (3*3*512)*512 = 2,359,296
CONV3-512: [14x14x512]  memory:  14*14*512=100K   params: (3*3*512)*512 = 2,359,296
CONV3-512: [14x14x512]  memory:  14*14*512=100K   params: (3*3*512)*512 = 2,359,296
POOL2: [7x7x512]  memory:  7*7*512=25K  params: 0
FC: [1x1x4096]  memory:  4096  params: 7*7*512*4096 = 102,760,448
FC: [1x1x4096]  memory:  4096  params: 4096*4096 = 16,777,216
FC: [1x1x1000]  memory:  1000 params: 4096*1000 = 4,096,000

TOTAL memory: 24M * 4 bytes ~= 96MB / image (only forward! ~*2 for bwd)
```

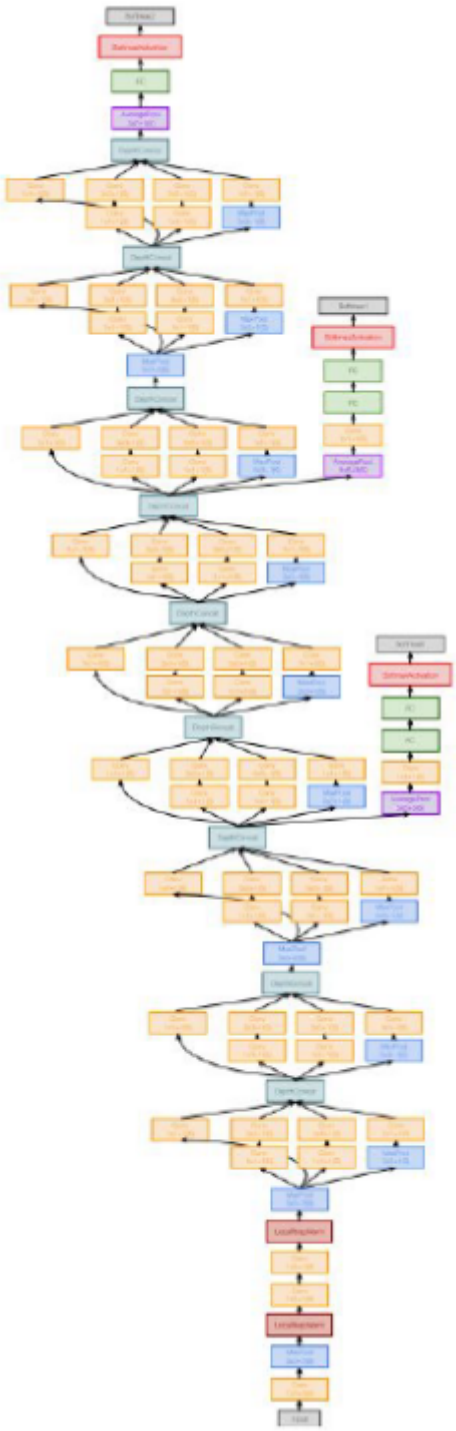


VGG16

- AlexNet과 비슷한 절차를 거치며 학습됨
- Local Response Normalisation이 없음
- Ensemble을 사용해 최고의 결과를 뽑아냄

- 맨 마지막에서 두 번째 FC 층은 다른 task들을 가지고도 일반화를 잘 함

GoogLeNet



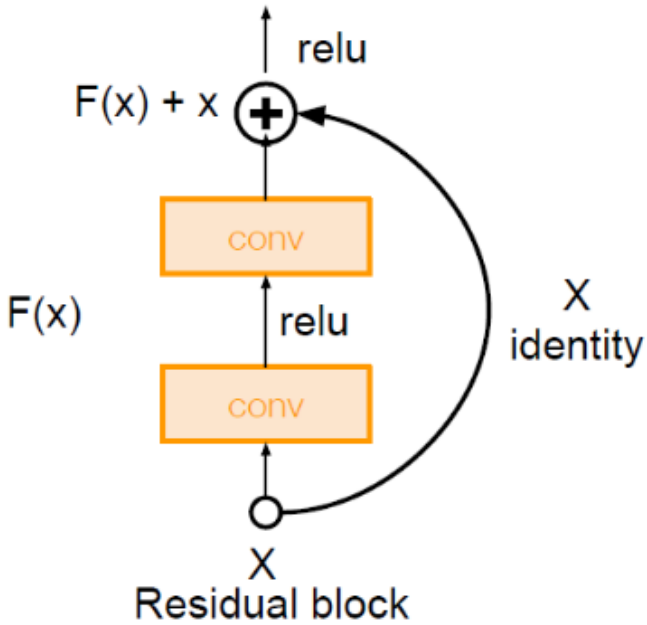
Inception module을 사용했다는 것이 가장 큰 특징이다. (네트워크 안에 작은 네트워크가 더 들어있는 구조이다.)

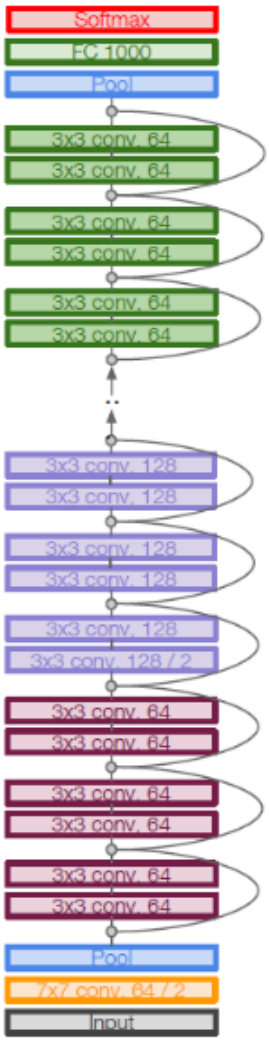
- 22Layer들로 구성되어있음
- parameter의 수를 줄이기 위해 FC 층이 없음
- 총 5백만개의 파라미터 수를 이용함 (AlexNet에 비해 많이 줄어듦)
- 많은 연산량을 효율적으로 수행함

Inception module은 동일한 입력을 받는 서로 작은 다양한 필터들이 병렬적으로 존재한다. 그런데 이 구조는 1x1 convolution layer라는 특별한 층이 없는 채로 사용하게 되면 연산량이 매우 커진다는 단점이 있었고 이를 1x1 convolution layer로 해결했다. 이 층을 이용하면 input depth가 줄어드는 효과가 있다. (= Bottleneck Layer)

전체 구조에서 봤을 때 곁다리로 빠져있는 보조 분류기들은 네트워크의 깊이가 깊기 때문에 중간 layer의 학습을 돕기 위해서 설계한 것이다.

ResNet





일반적으로 CNN을 깊게 쌓게 되면 Overfitting이 일어날 것이라 예측할 수 있는데 실제로는 그렇지 않다. 더 깊은 layer를 가진 구조가 training error가 더 높을 수도 있다. 여기서 세울 수 있는 한 가지 가설은 layer가 깊을 수록 optimize하기가 까다롭다는 것이다. (=Degradation)

이를 해결하기 위해서 어떤 방식을 도입한 것이 ResNet이다. 일반적으로 Layer를 쌓아올리는 방식 대신 Skip connection이라는 새로운 구조를 이용해 학습을 진행한다. 가중치 layer는 $H(x) - x$에 대한 값이 0에 수렴하도록 학습을 진행했다. 그 결과 연산 증가도 크게 없고 Gradient Vanishing 현상이 일어나더라도 원본 신호에 대한 내용이 남아 있어서 학습을 원활하게 시킬 수 있다. (= Residual Block)

이를 이용하면 layer가 깊어질 수록 더 정확하게 training 시킬 수 있다.

- VGGNet과 유사한 목적을 가지고 만든 것이므로 대부분의 Convolution layer를 3x3으로 설계
- 복잡도를 낮추기 위해 dropout, hidden FC를 사용하지 않음
- 출력 feature map의 크기가 같은 경우, 해당 모든 layer는 모두 동일한 수의 filter를 가짐
- feature map의 크기를 줄일 때는 pooling을 사용하는 대신 convolution을 수행할 때, stride의 크기를 2로 함

이 네트워크를 더 깊게 쌓아서 사용할 때는 GoogLeNet과 비슷하게 Bottleneck layer를 추가한다.