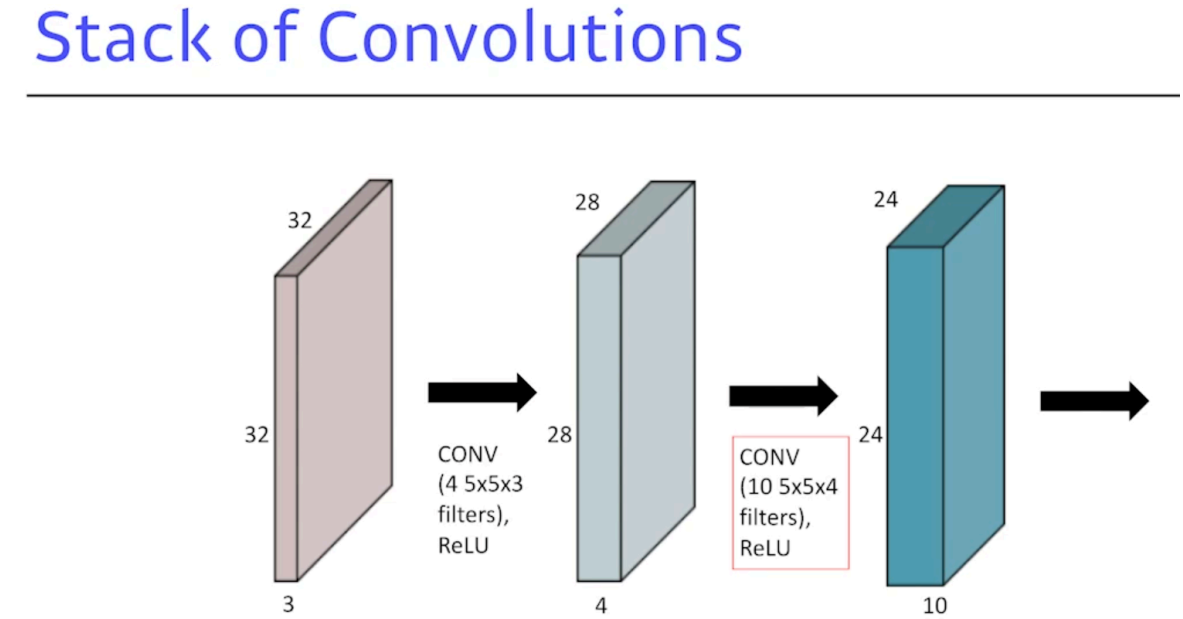
Week3-3

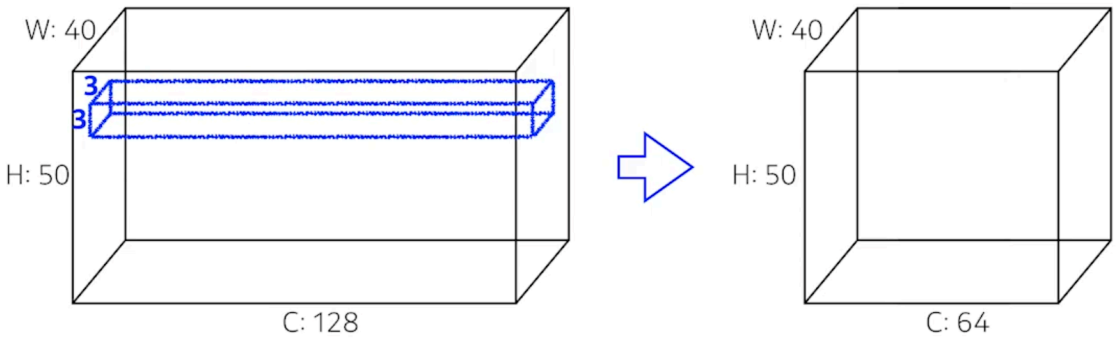
1. **학습정리**
    1. 컨볼루션
        1. 컨볼루션 결과는 커널 하나당 채널이 1인 피쳐맵 하나
        2. 여러개의 피쳐맵은 같은 수의 커널



        3. 다음 레이어의 채널 수가 4라면 커널 네개
        4. 커널의 채널은 현재 이미지의 채널
    2. CNN
        1. 컨볼루션 레이어와 풀링 레이어 번갈아 가면서 쌓아서 특징을 추출
        2. 마지막에 풀리 커넥티드 레이어(덴스레이어)로 의사결정
        3. 레이어 별로 파라미터의 수와 전체 파라미터의 수를 파악하기
    3. 스트라이드, 패딩
        1. 파라미터 수와 무관
    4. 파라미터 세아리기

Convolution Arithmetic

- Padding (1), Stride (1), 3×3 Kernel



What is the **number of parameters** of this model?

The answer is $3 \times 3 \times 128 \times 64 = 73{,}728$

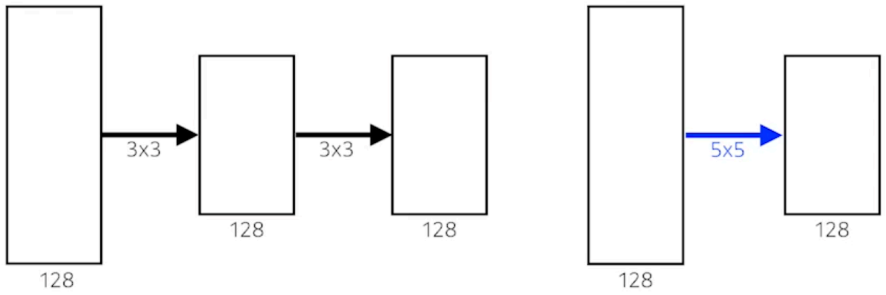
1. 커널의 크기 =. 3X3X인풋의 채널 수X 아웃풋의 채널 수
2. 패딩과 스트라이드는 파라미터 수와 무관함

4. 1x1 컨볼루션
    1. 이미지의 한 픽셀만 보고 채널방향으로 줄임
    2. 차원(채널)축소
    3. 채널 숫자를 줄임
    4. 파라미터 숫자를 줄임
    5. 파라미터 수를 줄이면서 네트워크를 깊게 쌓음

5. 모던 CNN
    1. 특징
        1. 뎁스는 점점 줄어듦
        2. 파라미터 수는 점점 줄어듦
        3. 성능은 점점 올라감
    2. 알렉스넷
        1. ReLU
        2. 드롭아웃
        3. 데이터 어그먼테이션
        4. 배니싱 그래디언트 문제 해결
            1. 양 끝 그래디언트가 0에 가까워 지는 것
    3. VGGNet

VGGNet

- Why 3×3 convolution?



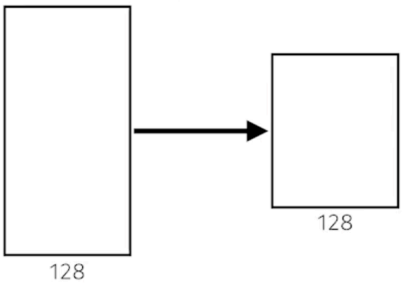
Receptive field	5x5	5x5
# of params	3x3x128x128+3x3x128x128 = **294,912**	5*5*128*128 = **409,600**

1. 3x3 컨볼루션 활용
2. 3x3 컨볼루션 두 번 하면 리셉티브 필드가 5x5임
3. 3x3 두 번 하는게 같은 리셉티브 필드를 얻으면서 파라미터 수가 적음
4. 보통 필터의 크기는 7x7 을 벗어나지 않음

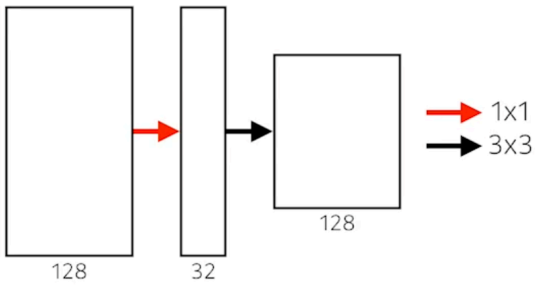
4. 구글넷
    1. 인셉션 블락

Inception Block

- Benefit of 1x1 convolution



$$3 \times 3 \times 128 \times 128 = 147,456$$



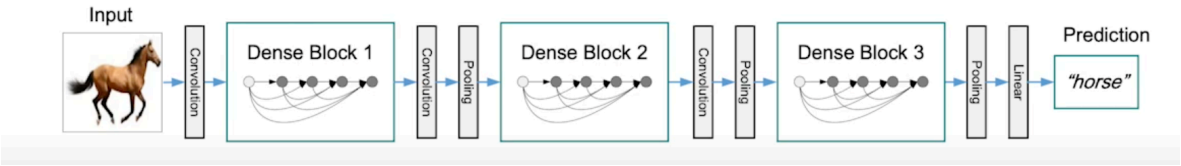
$$1 \times 1 \times 128 \times 32 = 4,096$$
$$3 \times 3 \times 32 \times 128 = 36,864$$
$$4,096 + 36,864 = 40,960$$

1. 1x1 컨볼루션으로 네트워크 수를 줄임
2. 3x3 하나 만 하는 것 보다 파라미터 수가 훨씬 줄어듦

2. ResNet
    1. 네트워크가 깊어질 수록 테스트 에러가 커짐 (오버피팅)
    2. 스킵커넥션
        1. 차이만 학습하게 함
    3. 보틀넥 아키텍쳐
        1. 3x3 컨볼루션 전에 인풋채널을 줄이기 위해 1x1 컨볼루션
        2. 3x3 컨볼루션 후에 채널을 늘리기 위해 1x1 컨볼루션
3. DenseNet

DenseNet

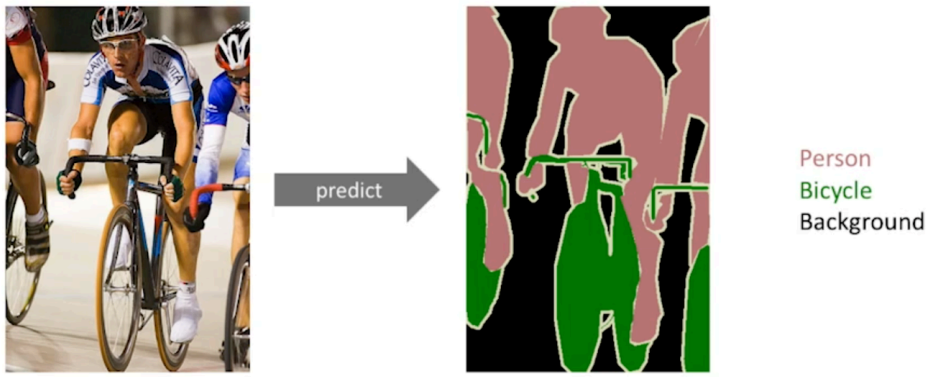
- Dense Block
  - Each layer concatenates the feature maps of all preceding layers.
  - The number of channels increases geometrically.
- Transition Block
  - BatchNorm -> 1x1 Conv -> 2x2 AvgPooling
  - Dimension reduction



1. 덴스블락
    1. 레즈넷처럼 더하지 않고 콘캣함
    2. 채널을 늘림
2. 트랜지션 블락
    1. 채널을 줄임

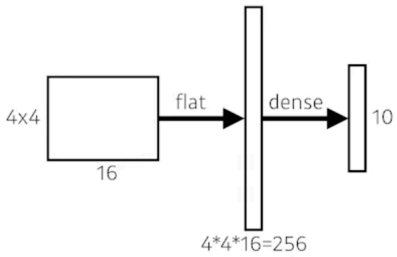
6. sementic segmentation

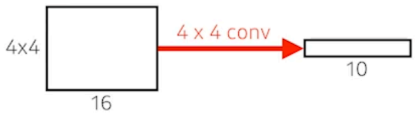
Semantic Segmentation



1. 이미지의 모든 픽셀이 각각 어떤 클래스에 속하는지 예상하는 것
    1. 자율 주행에 활용
2. convolutionalization

Fully Convolutional Network



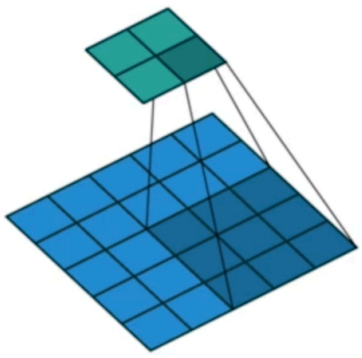


- # of parameters
  - Left: $4 \times 4 \times 16 \times 10 = 2{,}560$
  - Right: $4 \times 4 \times 16 \times 10 = 2{,}560$

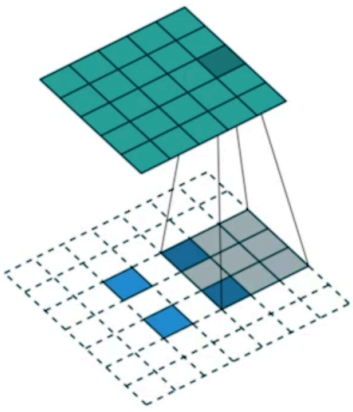
1. 덴스레이어를 컨볼루션 레이어로 대체
2. 덴스 레이어 쓰는 것과 컨볼루션으로 대체 하는 것은 파라미터 수가 같음
3. 네트워크, 파라미터 수, 인풋, 아웃풋 모두 같음

3. 풀리 컨볼루셔널 네트워크의 특징
   1. 인풋 이미지가 커지던 작아지던 상관 없이 네트워크가 돌아감
   2. 아웃풋이 커지게 되면 그 것과 비례해서 뒷 단의 네트워크가 커짐
4. deconvolution (conv transpose)

Deconvolution (conv transpose)



Convolution



Deconvolution

1. spacial dimension(이미지 크기) 을 늘림
2. 컨볼루션은 보통 줄임
3. 컨볼루션의 역이라 생각하면 크기 계산하기 편함
4. 파라미터 수와 네트워크의 입력과 출력은 같음

5. detection
   1. 이미지 안에서 바운딩 박스를 찾음
   2. R-CNN

        1. 하나의 이미지당 2000개의 영역을 만들어 판단
        2. 이미지 하나당 2000번 CNN 돌려야 해서 느림
    3. SPPNet
        1. R-CNN 과 똑같은데 CNN 한 번만 돌림
    4. Faster R-CNN
        1. 바운딩 박스를 뽑아내는 resion proposal 을 학습
        2. Resion Proposal Network (RPN)
            1. 안에 물체가 있을지 예측
    5. YOLO
        1. 이미지 한 장에서 바로 아웃풋이 나옴
        2. RPN 필요없음
        3. 매우빠름
2. **피어세션**
    1. 강의리뷰
    2. 퀴즈 틀렸던 것 리뷰
    3. pandas 연습하기
        1. 코드 출력 맞추기