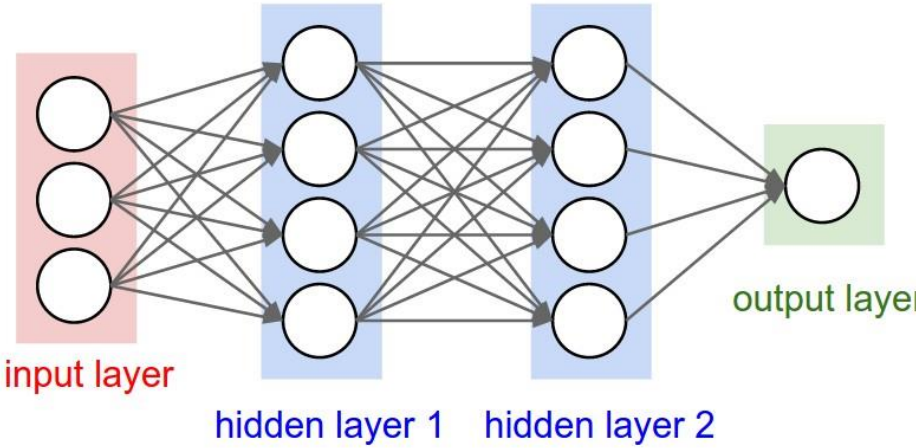
10강: 컨볼루션 신경망

인트로

- 컨볼루션 신경망은 일반적인 신경망과 매우 유사함
  - 학습 가능한 가중치와 편향벡터를 가진 뉴런들
  - 각 뉴런은 입력을 받아 내적을 수행하고 비선형 활성화 함수에 연결
  - 전체 신경망은 단일 미분 가능 점수 함수를 표현
  - 마지막 계층에서 SVM/소프트맥스 등의 손실 함수를 가짐
  - 일반 신경망 학습 위한 팁/트릭들이 여전히 적용됨
- 컨볼루션 신경망 아키텍쳐는 입력이 이미지라는 가정 하에 특정 속성들을 포함시킴
- forward 함수를 효율적으로 구현하고 파라미터 수를 크게 줄임

아키텍쳐(architecture) 개요

- 일반 신경망 복습
  - 신경망은 입력을 받아 여러 개의 은닉 계층을 통과시킴
  - 각 은닉계층은 여러 뉴런들로 구성됨
  - 각 뉴런은 이전 계층의 모든 뉴런과 완전히 연결됨
  - 한 계층 내의 뉴런들은 완전히 독립적으로 작동하고 어떠한 연결도 공유하지 않음
  - 마지막 완전연결계층은 출력계층이라 불리며 클래스 점수를 나타냄

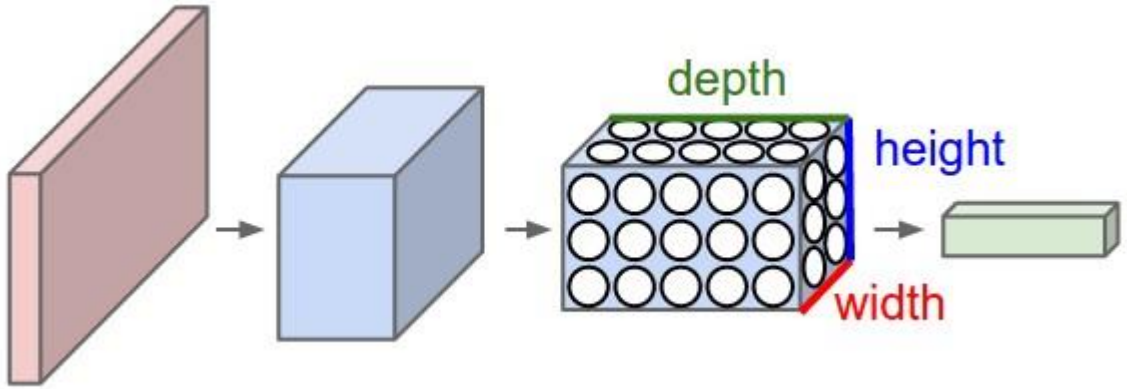


아키텍쳐(architecture) 개요

- 일반 신경망은 큰 이미지로 잘 확장되지 않음
  - CIFAR-10 이미지의 크기는 32x32x3 이므로, 일반 신경망의 첫 번째 은닉 계층 뉴런은 3072개의 가중치 파라미터를 가짐
  - 아직 관리 가능한 범위이지만, 이 완전연결구조는 더 큰 이미지에 확장되기 어려움
  - 200x200x3 이미지의 경우 120,000개의 가중치를 가진 뉴런을 필요로 함
  - 이런 완전연결방식은 낭비적이고, 엄청난 양의 파라미터는 빠르게 과적합을 일으킬 것임

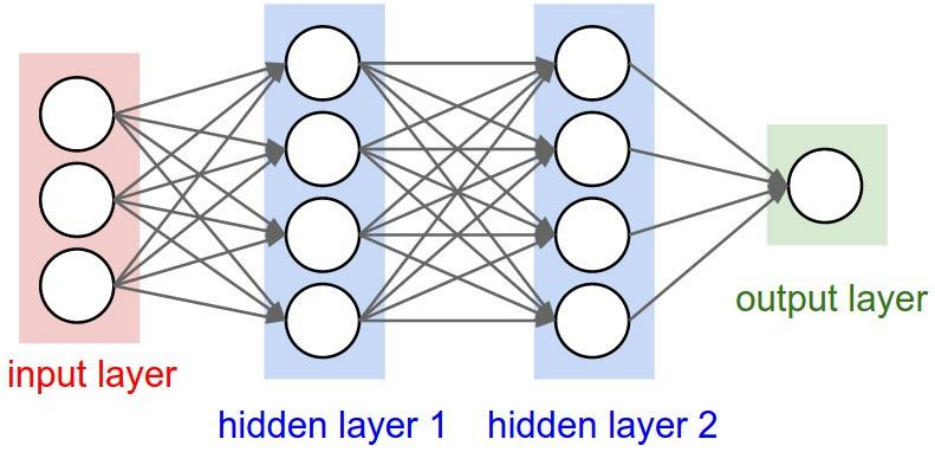
아키텍쳐(architecture) 개요

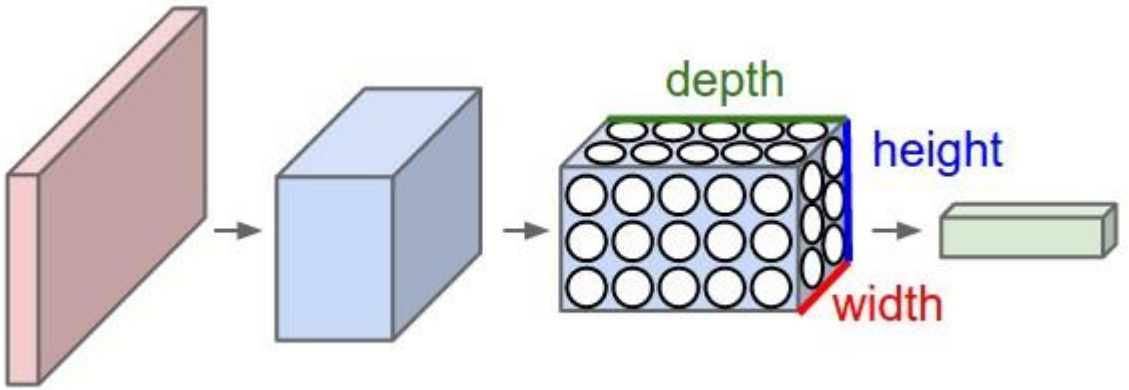
- 뉴런의 3D 볼륨
  - 컨볼루션 신경망은 아키텍쳐를 합리적인 방식으로 제한함
  - 일반 신경망과 달리 컨볼루션 신경망의 계층의 뉴런들은 3차원으로 배열됨: 너비, 높이, 깊이
  - 한 계층의 뉴런들은 이전 계층의 작은 영역에만 연결될 것임
  - 최종 출력 계층은 CIFAR-10에 대해 1x1x10 차원을 가짐
    - 클래스 점수의 단일 벡터



아키텍쳐(architecture) 개요

- 뉴런의 3D 볼륨
  - 컨볼루션 신경망에서는 뉴런이 3D로 배열됨 (너비, 높이, 깊이)
  - 컨볼루션 신경망의 모든 계층은 3D 입력 볼륨을 뉴런 활성화의 3D 출력 볼륨으로 변환함
  - 빨간색 입력 계층은 이미지에 해당됨
    - CIFAR-10의 경우 32x32x3
  - 각 계층은 간단한 API를 가짐: 입력 3D 볼륨을 출력 3D 볼륨으로 변환



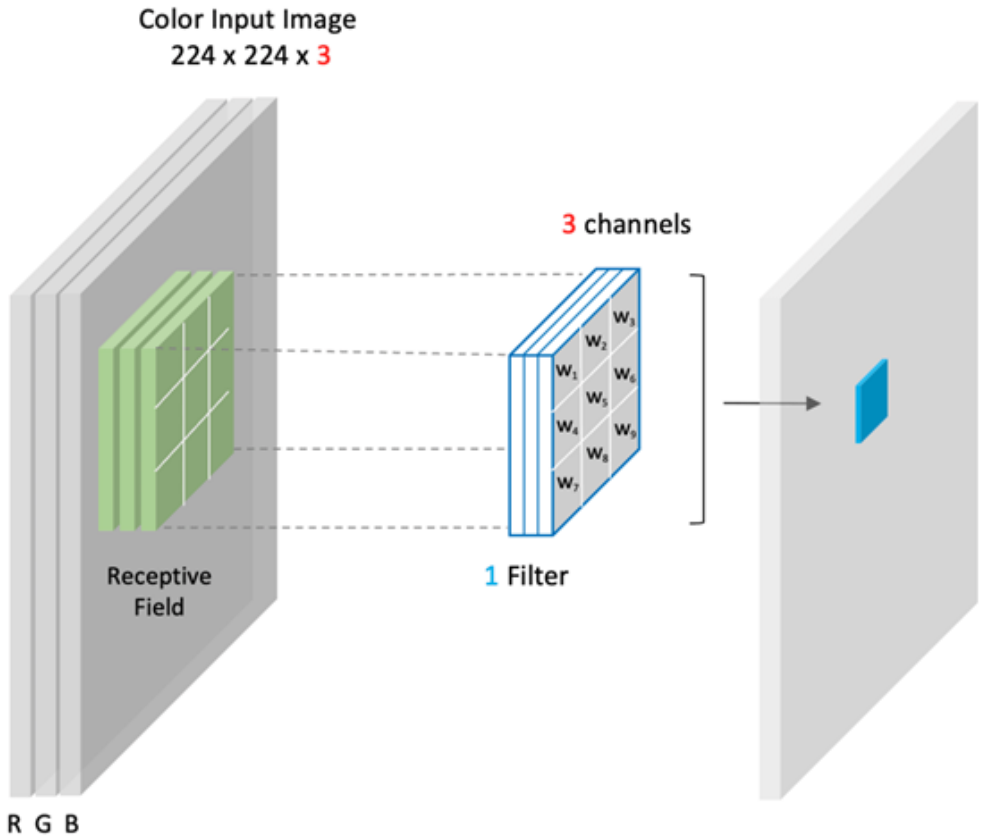


컨볼루션 신경망을 구성하는 계층

- 간단한 컨볼루션 신경망은 여러 계층들의 연속
- 각 계층은 활성화 볼륨을 다른 활성화 볼륨으로 미분 가능한 함수를 통해 변환함
- 컨볼루션 신경망 아키텍쳐 구축을 위해 세 가지 주요 유형의 계층이 사용됨
  - 컨볼루션 계층 (convolutional layer)
  - 풀링 계층 (pooling layer)
  - 완전연결 계층 (fully-connected layer)

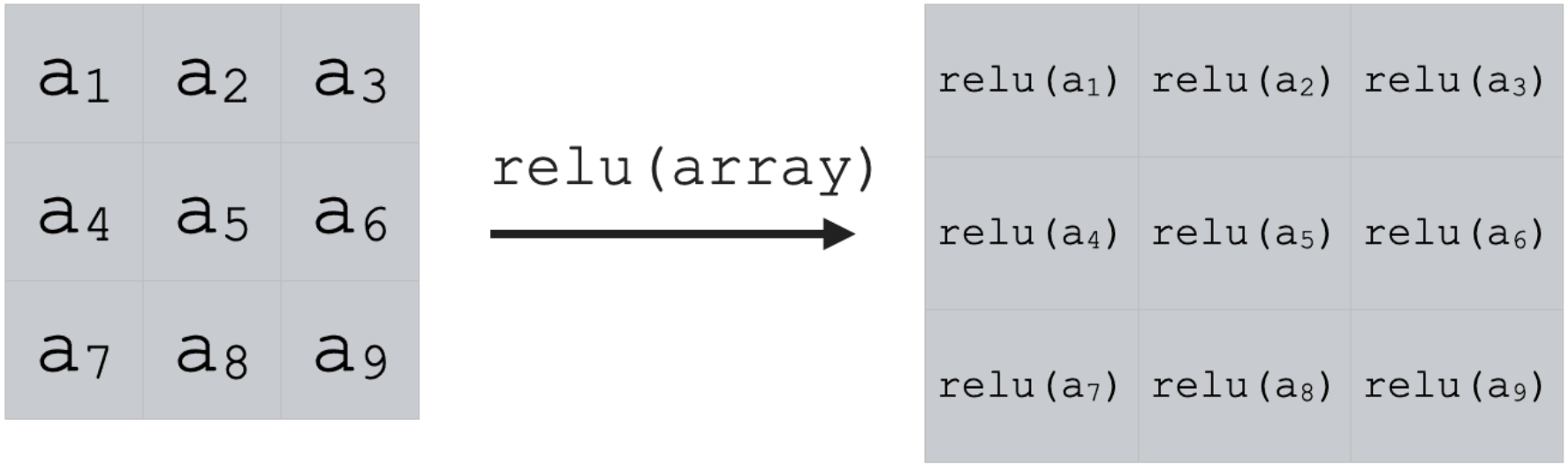
컨볼루션 신경망을 구성하는 계층

- CIFAR-10 분류를 위한 간단한 컨볼루션 신경망은 다음과 같을 수 있음
  - 입력 – 컨볼루션 – ReLU – 풀링 – 완전연결계층
- 입력 볼륨
  - 32x32x3 볼륨은 이미지 픽셀 값들을 가짐
  - 너비: 32, 높이: 32, 깊이: 3 (RGB 3가지 색상 채널)
- 컨볼루션 계층
  - 입력의 로컬 영역에 연결된 뉴런의 출력을 계산
  - ex) 32x32x3 -> 32x32x12



컨볼루션 신경망을 구성하는 계층

- ReLU 계층
  - 성분별 활성화 함수 적용
  - ReLU(x)=max(0,x)
  - 볼륨의 크기를 변경하지 않음
  - ex) 32x32x12 -> 32x32x12

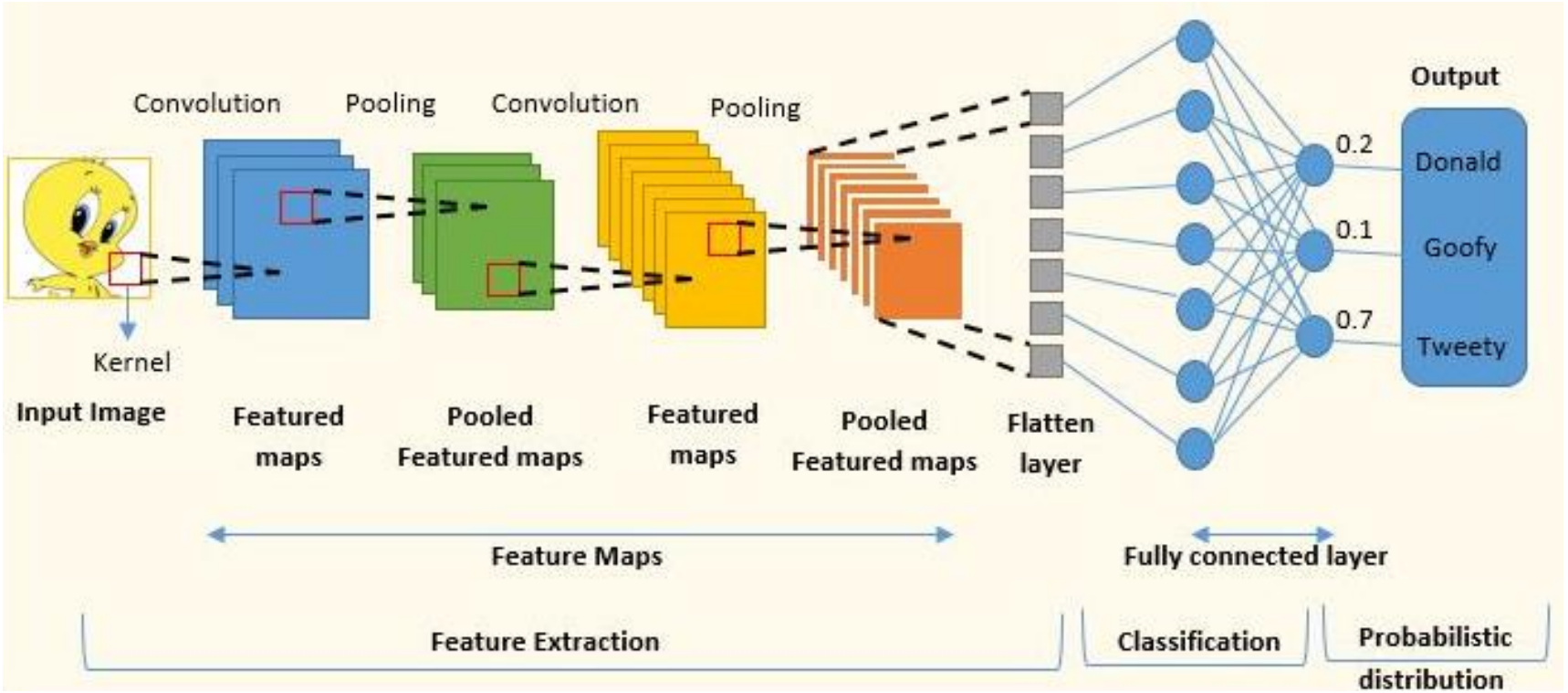


컨볼루션 신경망을 구성하는 계층

- 풀링 계층
  - 공간 차원(너비나 높이)을 따라 다운샘플링 연산을 수행
  - ex) 32x32x12 -> 16x16x12
- 완전연결 계층
  - 클래스 점수를 계산하고 1x1x10 크기의 볼륨을 결과로 내놓음
  - 10개의 숫자는 각 클래스의 점수에 해당됨
  - 각 뉴런은 이전 볼륨의 모든 숫자에 연결됨

컨볼루션 신경망을 구성하는 계층

- 컨볼루션 신경망 그림 예시

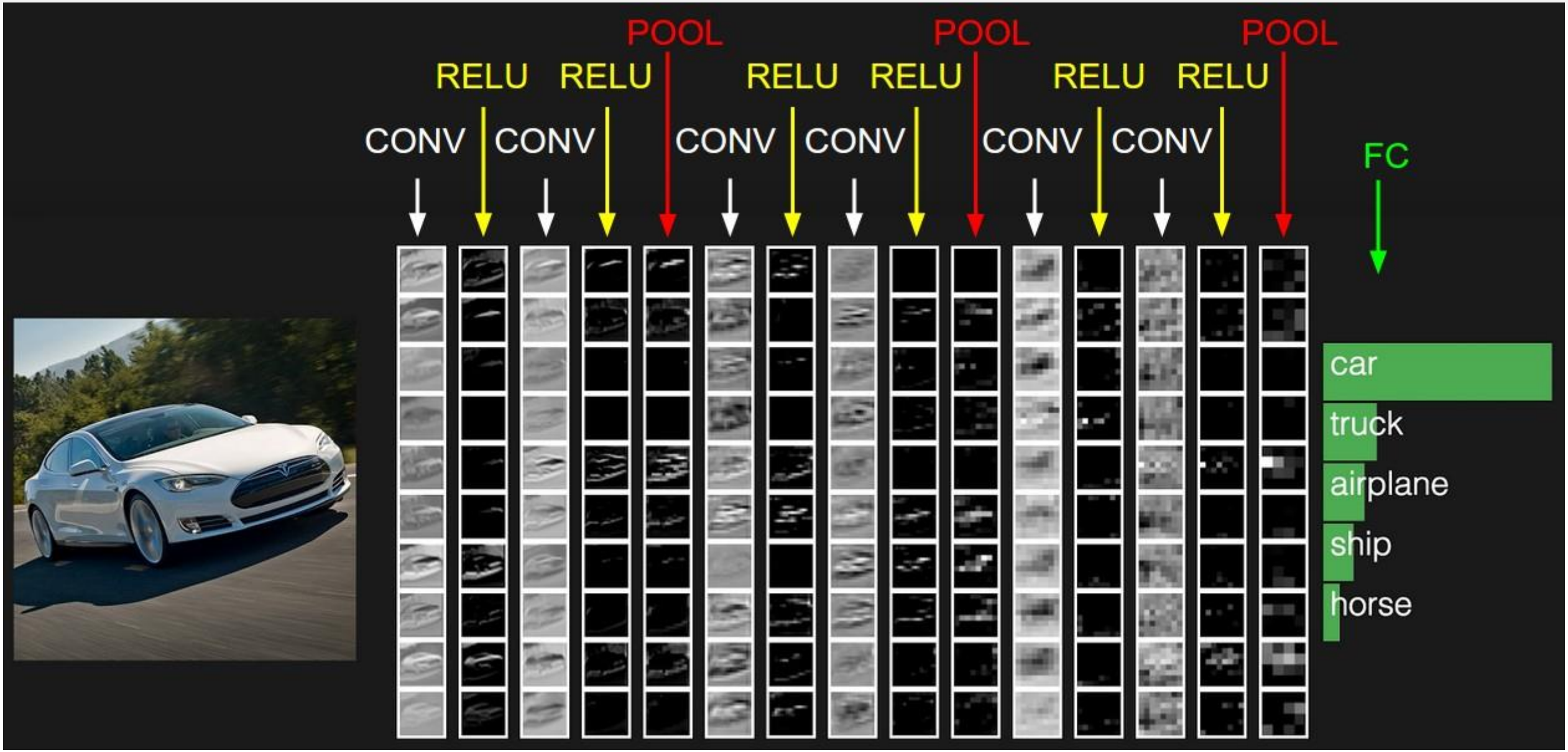


- 일부 계층은 파라미터를 포함하며, 일부는 그렇지 않음
- 컨볼루션 신경망과 완전연결계층은 입력볼륨 뿐 아니라 파라미터에 의한 함수 변환을 수행
- ReLU/풀링 계층은 고정된 함수를 수행

컨볼루션 신경망을 구성하는 계층

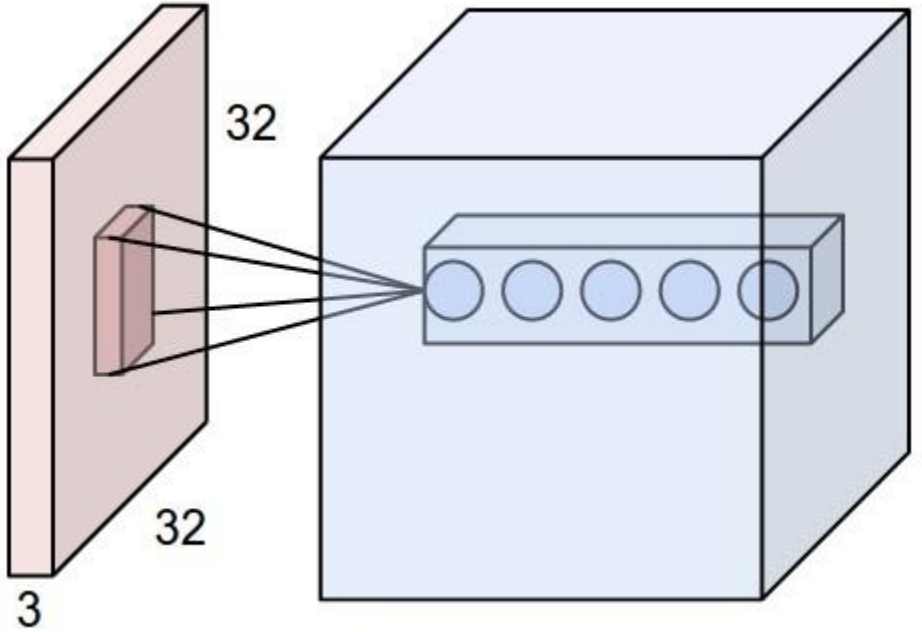
- 요약
  - 가장 간단한 컨볼루션 신경망 구조는 일련의 계층들로 구성되며 이미지 볼륨을 출력 볼륨(ex. 클래스 점수 벡터)으로 변환함
  - 몇 가지 유형의 계층들이 있음
    - 컨볼루션, 완전연결 계층, ReLU, 풀링이 가장 대표적
  - 각 계층은 입력 3D 볼륨을 받아 미분가능 함수를 통해 이를 3D 볼륨으로 변환
  - 각 계층은 파라미터를 가질 수도, 가지지 않을 수도 있음
  - 각 계층은 추가적인 하이퍼라라미터를 가질 수도, 가지지 않을 수도 있음

컨볼루션 신경망을 구성하는 계층



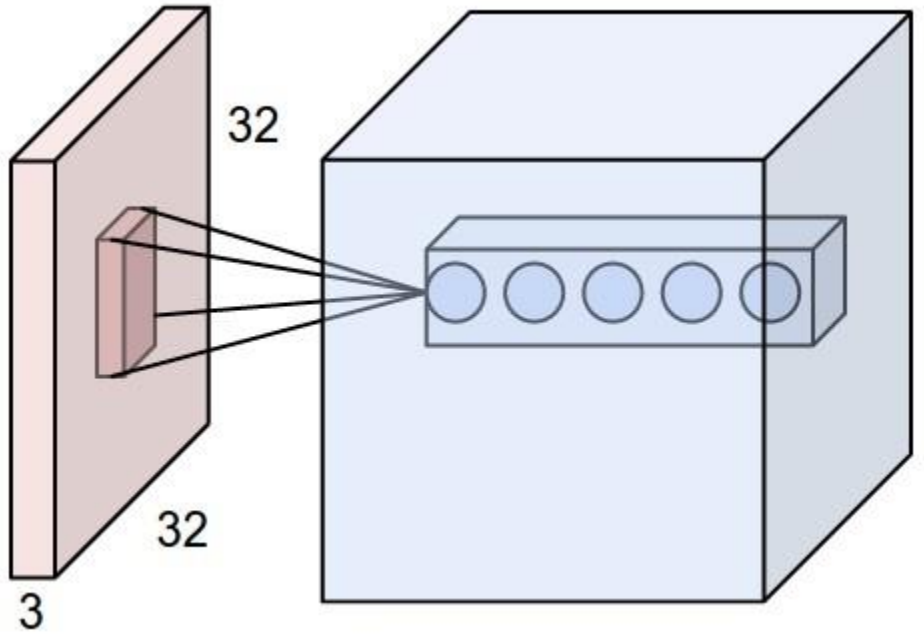
컨볼루션 계층

- 뇌 또는 뉴런 비유 없이 설명
  - 학습 가능한 여러 필터들의 세트로 구성됨
  - 모든 필터는 너비와 높이가 작지만 전체 깊이에 걸쳐있음
    - ex) 필터 크기는 5x5x3 일 수 있음
  - 필터를 입력 볼륨의 너비와 높이를 따라 이동시키며 필터 값 및 입력 볼륨 값 사이 내적을 계산하여 2차원 배열을 생성



컨볼루션 계층

- 뇌 또는 뉴런 비유 없이 설명
  - 직관적으로, 신경망은 특정 종류의 시각적 특징을 인식하는 필터를 학습
    - ex) 필터는 첫 계층의 엣지 또는 일부 색상의 얼룩일 수 있으며, 더 상위 계층에서 바퀴와 같은 패턴을 학습할 수도 있음
  - 각 컨볼루션 계층에는 필터들의 세트 존재
  - 각 필터는 별도의 2차원 배열을 생성하며 2차원 배열들을 깊이 차원을 따라 쌓아 출력 볼륨을 생성하게 됨

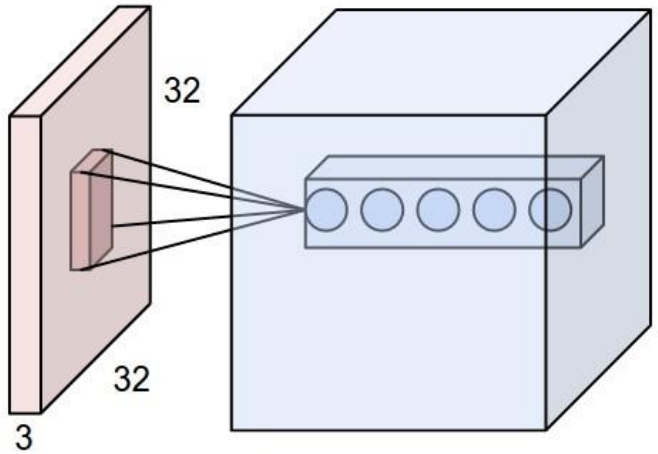


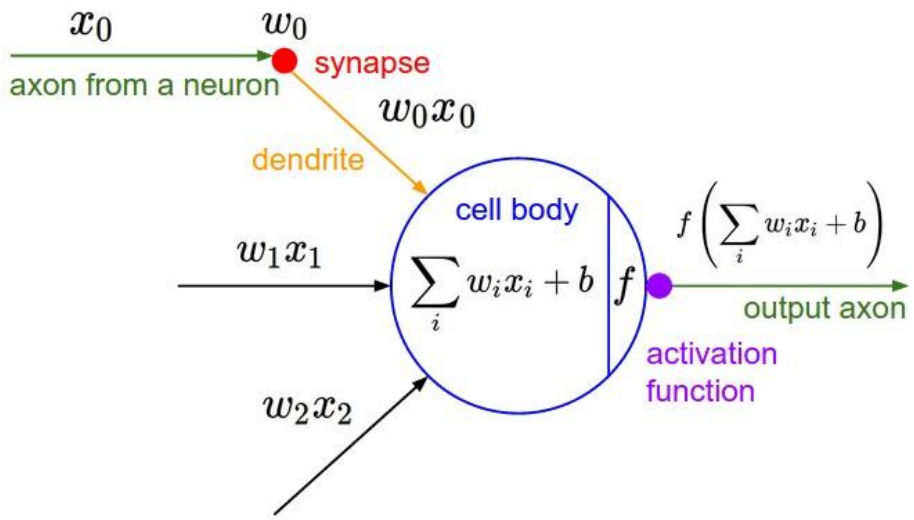
컨볼루션 계층

- 뇌의 관점
  - 3D 출력 볼륨의 각 성분은 입력 중 작은 영역만을 보는 뉴런의 출력으로도 해석될 수 있음
  - 이 뉴런은 모든 뉴런과 공간적으로 왼쪽과 오른쪽에서 파라미터를 공유
  - 이들 숫자들이 모두 동일한 필터를 적용하여 얻어진 결과이기 때문

컨볼루션 계층

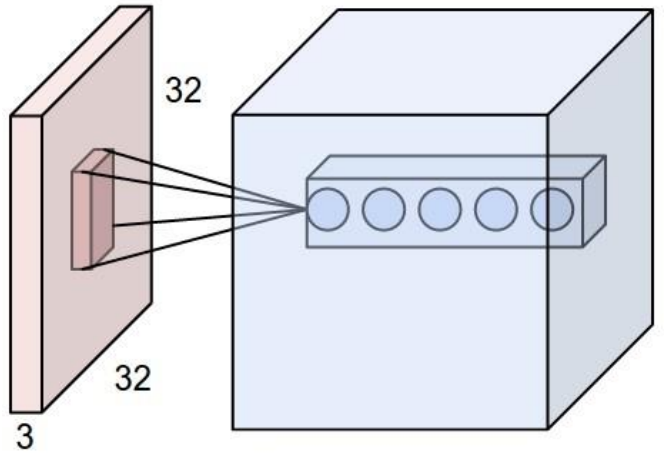
- 지역 연결성
  - 어떤 뉴런을 이전 볼륨의 모든 뉴런에 연결시키는 것은 비현실적
  - 대신, 각 뉴런을 입력 볼륨의 로컬 영역에만 연결
  - 이 연결성의 공간적 범위는 뉴런의 수용필드라는 하이퍼파라미터
  - 깊이 축을 따른 연결성의 범위는 항상 입력 볼륨의 깊이와 동일





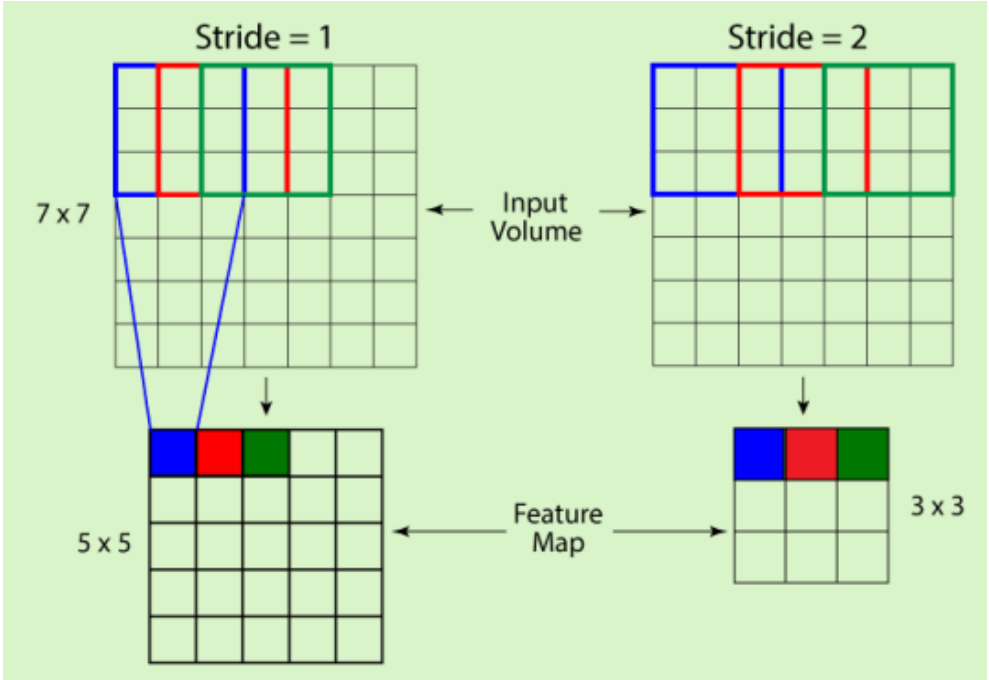
컨볼루션 계층

- 공간적 배치
  - 출력 볼륨의 크기를 제어하는 3가지 하이퍼파라미터가 있음
    - 깊이(depth), 스트라이드(stride), 제로 패딩(zero padding)
  - 출력 볼륨의 깊이는 하이퍼파라미터이며 필터 수에 해당됨
  - 각각은 입력에서 다른 특징을 찾는 것을 학습함
    - ex) 다른 방향의 엣지 또는 색상 덩어리
  - 같은 입력 영역을 보고 있는 모든 뉴런의 집합을 깊이 열(depth column)이라 부름



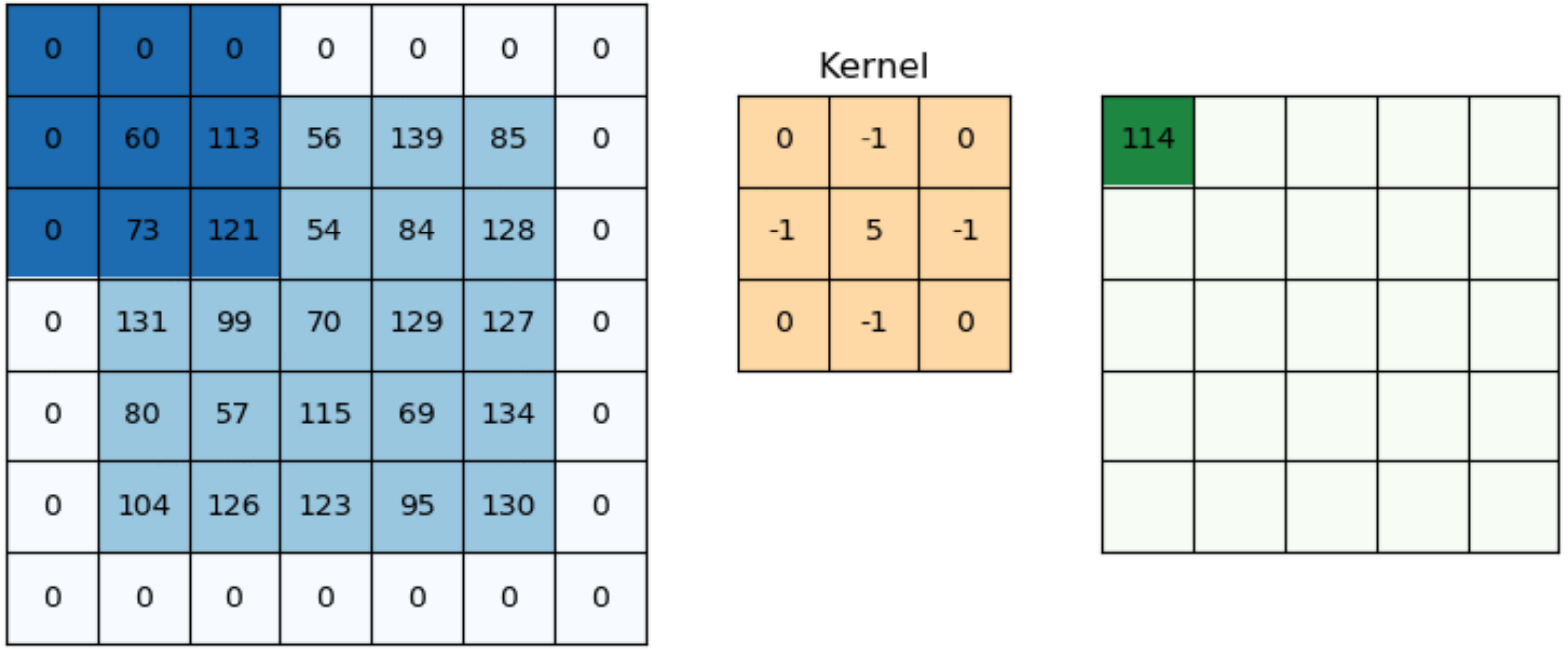
컨볼루션 계층

- 공간적 배치
  - 필터를 이동시킬 때 사용할 스트라이드를 지정해야함
  - 스트라이드가 1이면 필터가 한 픽셀씩 이동
  - 스트라이드가 2인 경우 필터는 2 픽셀씩 점프
  - 이렇게 하면 출력 볼륨이 공간적으로 더 작아짐



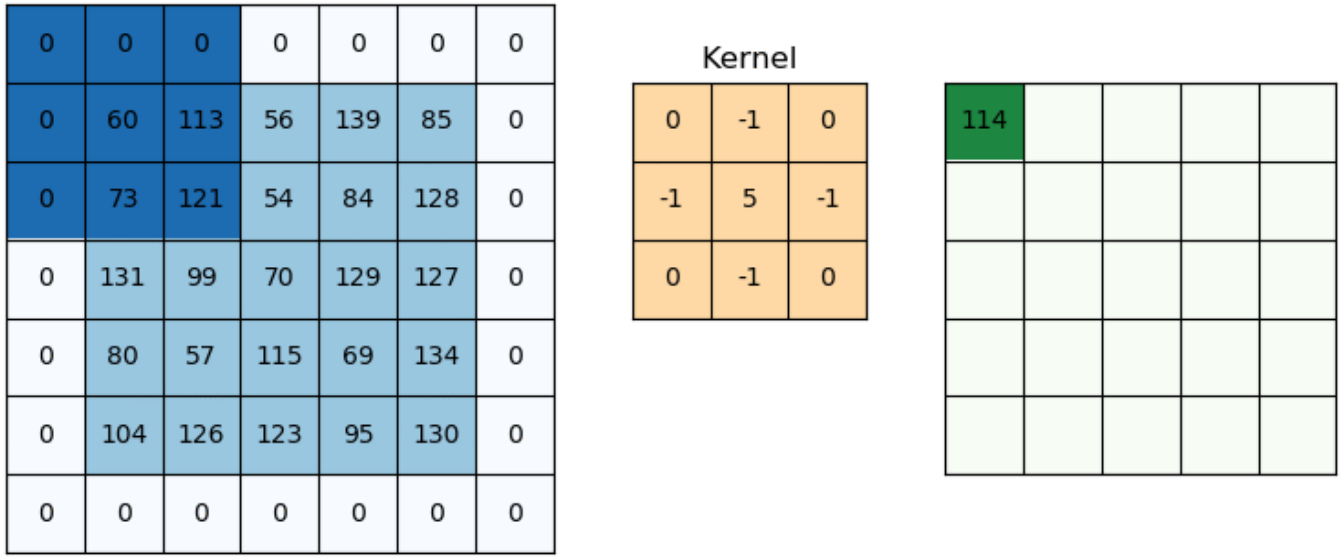
컨볼루션 계층

- 공간적 배치
  - 종종 입력 볼륨 주변에 제로 패딩을 적용하는 것이 편리함
  - 제로 패딩의 크기는 하이퍼파라미터임
  - 제로 패딩은 출력 볼륨의 공간 크기를 제어할 수 있게 함
  - 가장 흔히, 입력과 출력 볼륨의 너비와 높이가 동일하게 함



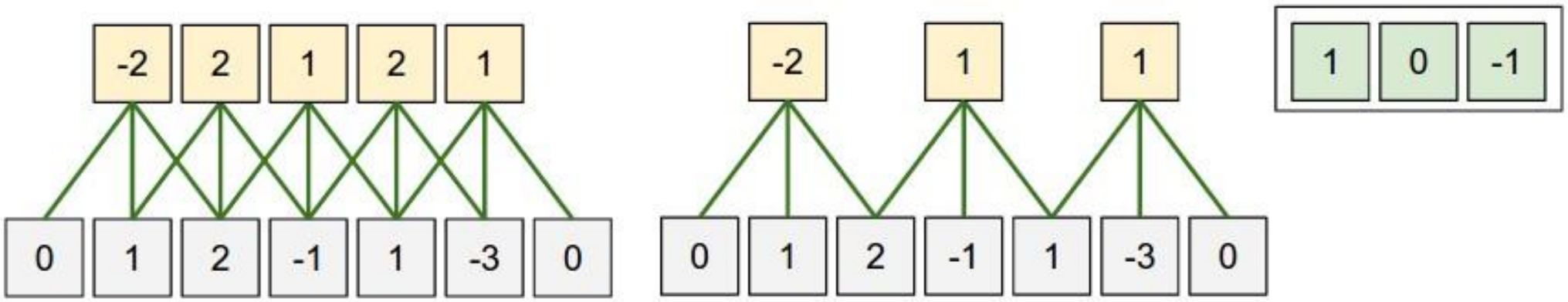
컨볼루션 계층

- 공간적 배치
  - 입력 볼륨 크기: W
  - 컨볼루션 계층 뉴런의 수용 필드 크기: F
  - 스트라이드: S
  - 제로 패딩 양: P
  - => 출력 볼륨 크기: (W-F+2P)/S+1



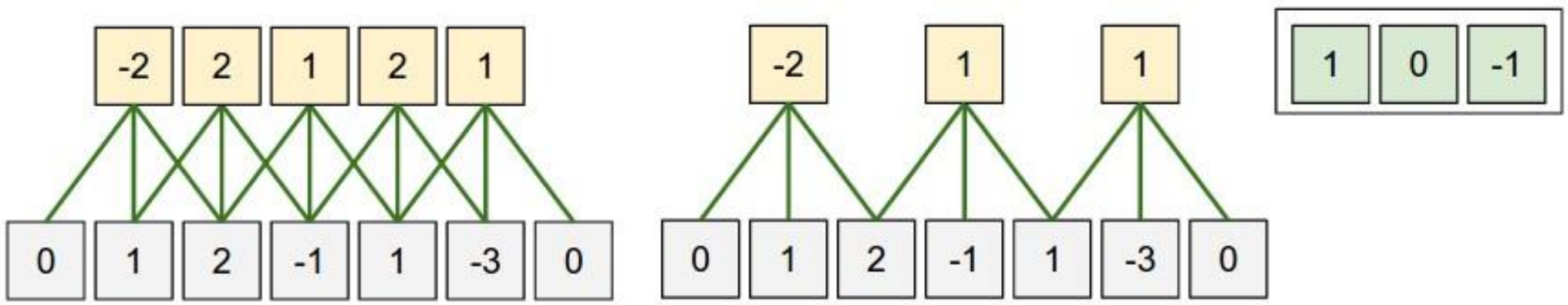
컨볼루션 계층

- 공간적 배치
  - 출력 볼륨 크기: (W-F+2P)/S+1
  - 예시1) W=7, S=1, P=0, F=3 일 때, 출력 볼륨 크기는 5
  - 예시2) W=7, S=2, P=0, F=3 일 때, 출력 볼륨 크기는 3



컨볼루션 계층

- 공간적 배치
  - 제로 패딩 사용
    - 일반적으로, 스트라이드가 S=1일 때, 제로 패딩을 P=(F-1)/2로 설정하면 입력 볼륨과 출력 볼륨 크기가 같게 됨
    - 이런 제로 패딩 사용은 매우 흔함



컨볼루션 계층

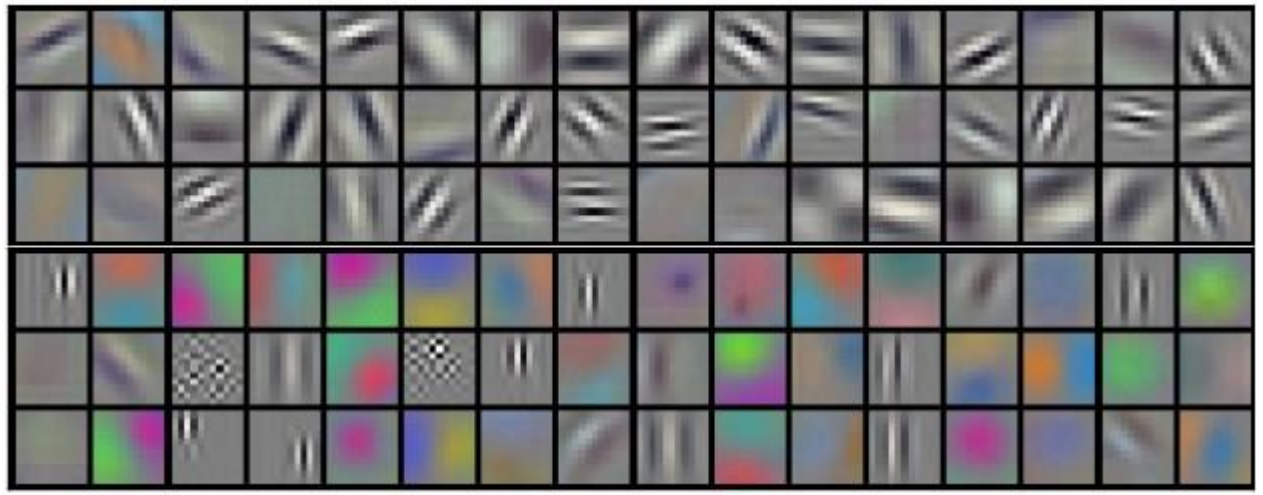
- 공간적 배치
  - 스트라이드에 대한 제약
    - 공간 배치 하이퍼파라미터들은 상호 제약이 있음
    - ex) W=10, P=0, F=3인 경우 S=2를 사용하는 것은 불가능함
    - (W-F+2P)/S+1 = 4.5는 정수가 아니며, 즉, 뉴런들이 입력에 대해 "맞지(fit)" 않는다는 것을 의미함
    - 이런 하이퍼파라미터 설정은 유효하지 않다고 간주되며, 컨볼루션 신경망 라이브러리에서는 이런 설정에 에러 표시를 하거나, 나머지 부분을 제로 패딩하는 등의 조치를 취할 수 있음

컨볼루션 계층

- 파라미터 공유
  - 예시
    - 11x11x3 -> 55x55x96 의 경우 55x55x96=290,400개의 뉴런, 11*11*3=363개의 가중치, 1개의 편향이 있음
    - 무려 290,400x364=105,705,600개의 파라미터가 필요함
  - 가정: 어떤 특징이 어떤 공간 위치 (x,y)에서 유용하면 (x2,y2)에서도 유용
  - 각 깊이 조각내의 뉴런들은 같은 가중치와 편향을 사용하도록 제약됨
    - ex) 55x55x96 볼륨은 96개의 (2차원) 깊이 조각들을 가짐
  - 가중치 세트는 필터 또는 커널이라 불림

컨볼루션 계층

- 필터 예시
  - 11x11x3 크기의 96개의 필터 그림
  - 각각은 55x55개의 뉴런들에 의해 공유됨
  - 파라미터 공유 가정은 비교적 합리적임
    - 어떤 위치에서 수평 가장자리를 감지하는 것은 직관적으로 다른 위치에서도 유용해야함



컨볼루션 계층

- 파라미터 공유 가정이 의미를 가지지 않을 수도 있음
  - ex) 컨볼루션 신경망의 입력 이미지가 어떤 centered 구조를 가질 때
- 예를 들면, 이미지의 한 쪽과 다른 쪽에서 완전히 다른 특징이 학습되는 것이 예상되는 경우
  - ex) 얼굴이 이미지 중앙에 위치한 입력 데이터
- 이런 경우 일반적으로 파라미터 공유 방식을 완화하고, 대신 계층을 로컬 연결 계층이라고 부름

컨볼루션 계층 요약

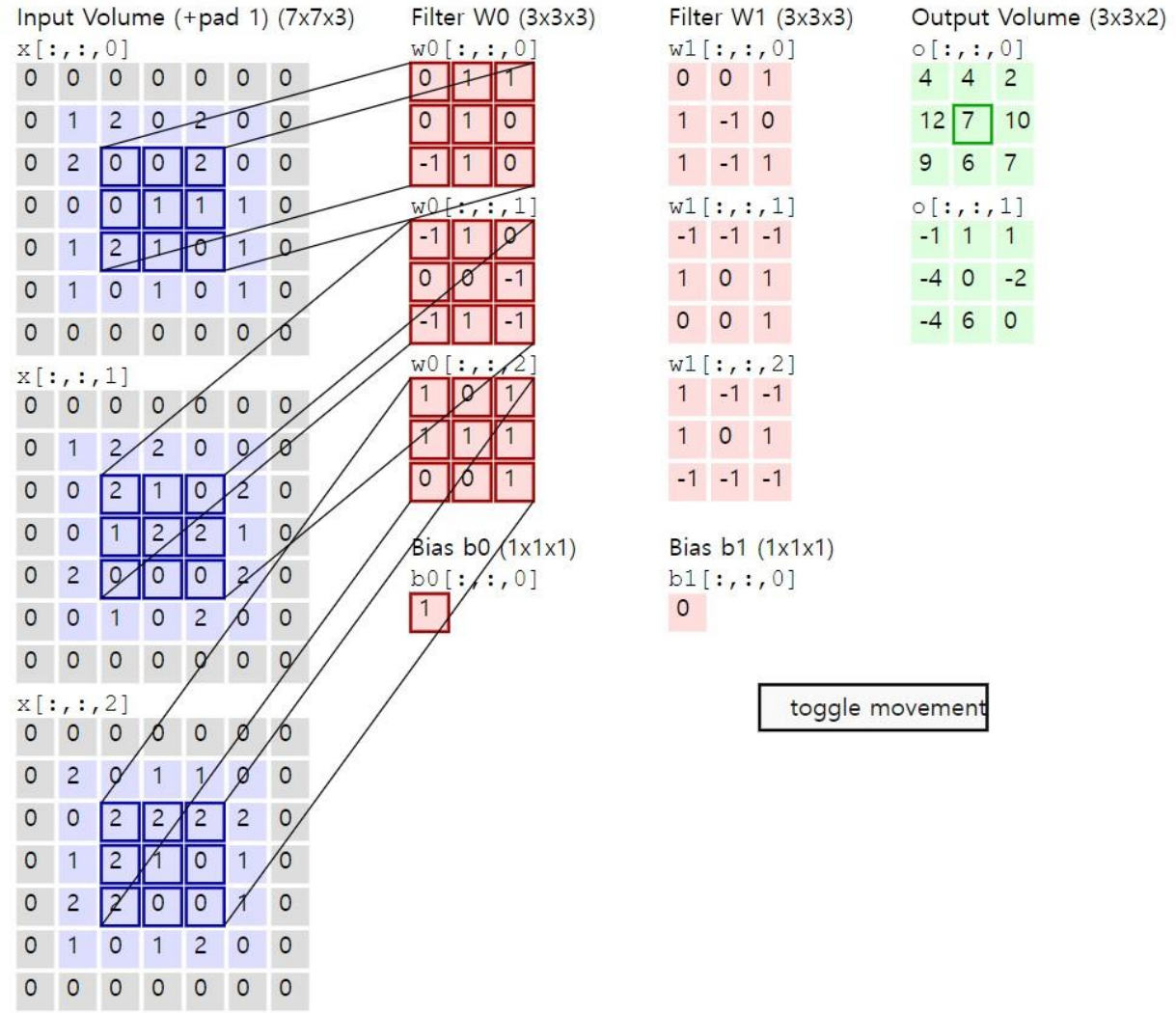
- 크기가 $W_1 \times H_1 \times D_1$인 볼륨을 받아들임
- 4가지 하이퍼파라미터가 필요함
  - 필터의 수 K
  - 공간적 범위 F
  - 스트라이드 S
  - 제로 패딩 양 P
- 크기가 $W_2 \times H_2 \times D_2$인 볼륨을 생성함
  - $W_2 = \frac{W_1 - F + 2P}{S} + 1$
  - $H_2 = \frac{H_1 - F + 2P}{S} + 1$
  - $D_2 = K$

컨볼루션 계층 요약

- 파라미터 공유를 통해 필터당 $F \cdot F \cdot D_1$ 가중치를 도입하며, 총 $(F \cdot F \cdot D_1) \cdot K$ 가중치와 K 편향을 도입함
- 출력 볼륨에서 d번째 깊이 조각 (크기 $W_2 \times H_2$)는 입력 볼륨 위에서 d번째 필터의 유효한 컨볼루션 수행 결과이며, 이는 S의 스트라이드로 수행되고, 그 후 d번째 편향이 더해짐
- ※ 하이퍼파라미터의 일반적인 설정은 F=3, S=1, P=1

컨볼루션 데모

- https://cs231n.github.io/convolutional-networks/



행렬 곱셈으로서의 구현

- 컨볼루션 연산은 본질적으로 필터와 입력의 로컬 영역 사이의 내적을 수행하는 것
- 컨볼루션 계층의 일반적인 구현 패턴은 이 사실을 활용하고, 컨볼루션 계층의 순방향 패스를 하나의 큰 행렬 곱셈으로 정의하는 것
- 입력 이미지의 로컬 영역은 im2col이라 불리는 연산을 통해 열로 늘어뜨려짐
- ex) 227x227x3이고 11x11x3 필터를 스트라이드 4로 컨볼루션 시 11x11x3 픽셀 블록을 취하고 각 블록을 11x11x3=363 크기의 열벡터로 늘어뜨림

행렬 곱셈으로서의 구현

- 스트라이드 4로 이 과정 반복 시, 너비와 높이 각각에서 55 위치를 얻게 되어, 총 55x55=3025 개의 위치가 있음
- im2col의 결과인 출력 행렬 X_col의 크기는 363x3025가 됨
- 각 열은 늘어뜨려진 수용필드이며, 수용 필드가 겹치므로 입력 볼륨의 모든 숫자가 여러 개의 열에 중복될 수 있음

행렬 곱셈으로서의 구현

- 컨볼루션 계층의 가중치는 행으로 늘어뜨려짐
  - ex) 11x11x3 크기 필터가 96개 있는 경우, 96x363크기의 행렬 W_row 생성
- 컨볼루션 결과는 하나의 큰 행렬 곱셈 np.dot(W_row,X_col) 수행과 동등
- 이는, 모든 필터와 모든 수용 필드 위치 사이 내적을 계산
  - ex) 96x3025 크기가 됨
  - 이는 각 위치에서 각 필터의 내적의 출력을 제공
- 마지막으로, 적절한 출력 차원 55x55x96으로 재구성

행렬 곱셈으로서의 구현

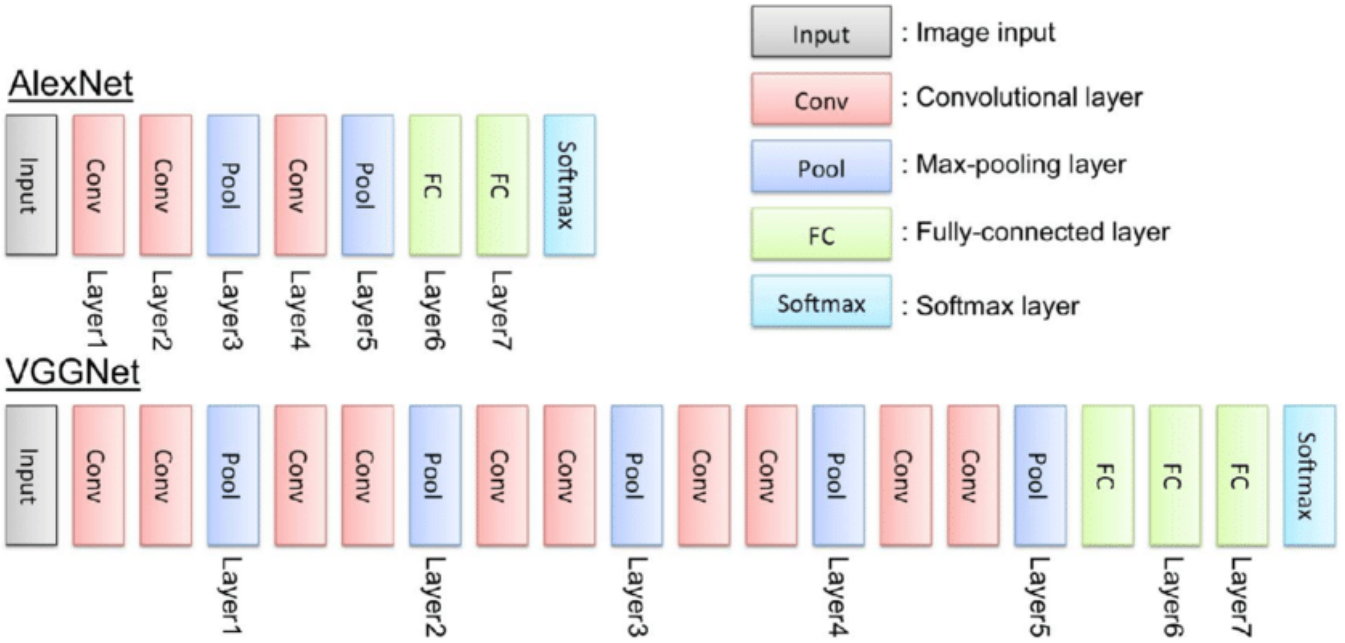
- 이 방식의 단점은 X_col에서 입력 볼륨의 일부 값이 여러 번 복제되므로 많은 메모리 필요할 수 있음
- 이점은 활용할 수 있는 매우 효율적인 행렬 곱셈 구현이 많다는 것

컨볼루션 계층

- 역전파
  - 컨볼루션 연산에 대한 역전파는 또한 컨볼루션
  - 공간적으로 뒤집힌 필터를 사용함
- 1x1 컨볼루션
  - 여러 논문에서 1x1 컨볼루션을 사용함
  - 컨볼루션 신경망은 3차원 볼륨을 다루고 있기 때문에 이는 의미가 있음

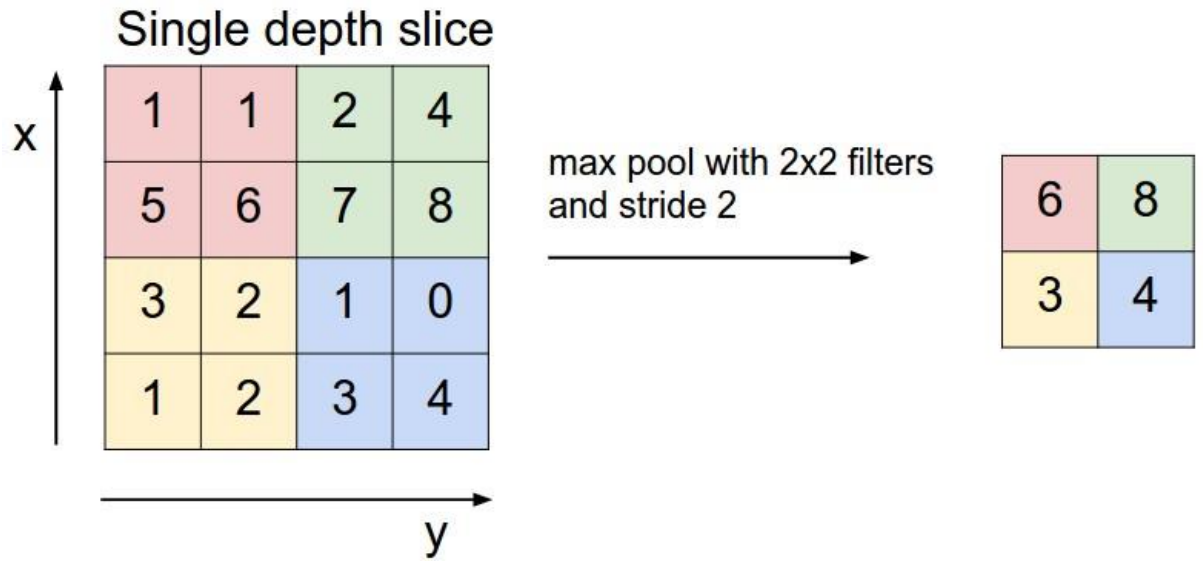
풀링 계층

- 연속적인 컨볼루션 계층 사이에 주기적으로 풀링 계층을 삽입하는 것이 일반적
- 풀링의 기능은 신경망 내의 파라미터와 계산량을 줄이기 위해 공간적 크기를 점차적으로 줄이는 것
- 이는 과적합을 제어하는데에도 도움이 됨



풀링 계층

- 풀링
  - 맥스(max) 연산을 사용하여 공간적으로 크기를 조정함
  - 가장 일반적인 형태는 2x2 크기의 필터를 갖는 풀링 계층
  - 스트라이드 2를 적용하여 입력의 모든 깊이 조각을 너비와 높이 모두 2만큼 다운샘플링함
  - 75%의 데이터를 버리게 되며, 깊이 차원은 그대로 유지

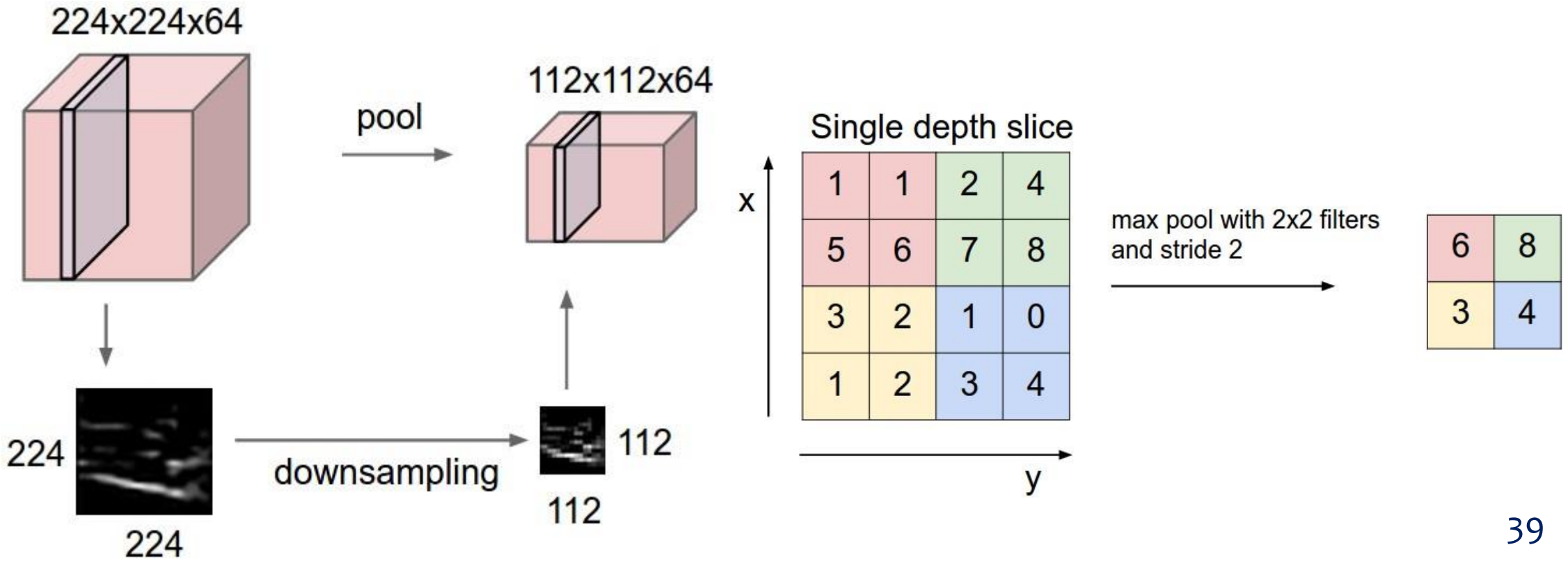


풀링 계층

- W1xH1xD1 볼륨을 입력으로 받음
- 2가지 하이퍼파라미터
  - 공간적 범위 F
  - 스트라이드 S
- W2xH2xD2 볼륨을 생성
- W2=(W1−F)/S+1, H2=(H1−F)/S+1, D2=D1 식이 성립
- 흔히 사용되는 풀링
  - F=2, S=2 (가장 대표적)
  - F=3, S=1 (overlapping pooling)
- F=3 보다 큰 수용 필드의 풀링은 거의 사용되지 않음

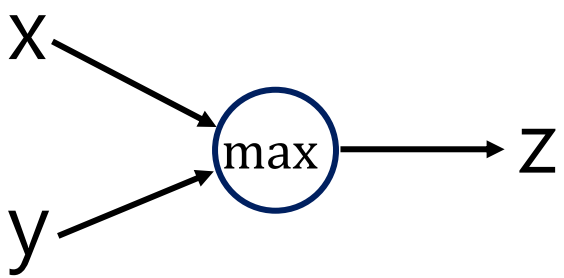
풀링 계층

- 평균 풀링(average pooling) 혹은 L2-norm 풀링도 가능
- 최근에는 평균 풀링 보다는 더 잘 작동하는 최대 풀링 연산을 주로 사용



풀링 계층

- 역전파
  - 최대값 게이트 그래디언트 전달
    - x>y이면 df/dx = df/dz, df/dy = o
    - x<y이면 df/dx = o, df/dy = df/dz



  - 풀링 계층의 전달 과정 동안에는 큰 입력의 인덱스를 추정하는 것이 일반적
  - 그러면 역전파 시 그래디언트가 전달되어야할 경로를 바로 알 수 있음

풀링 계층

- 풀링 제거
  - 많은 사람들은 풀링 계층 없이도 좋은 성능을 낼 수 있다고 생각
  - ex) https://arxiv.org/abs/1412.6806
    - 반복적인 컨볼루션 계층만으로 구성된 구조를 제안
  - 표현의 크기를 줄이기 위해 가끔 컨볼루션 계층에서 더 큰 스트라이드 사용이 제안됨
  - 풀링 계층을 제거하는 것이 우수한 생성 모델을 훈련시키는데에도 중요

정규화 계층

- 다양한 종류의 정규화 계층이 컨볼루션 신경망에 사용하기 위해 제안됨
- 때로는 생물학적 뇌에서 관찰된 억제 체계를 구현하려는 의도로 제안되기도 함
- 이 계층들은 실제로 기여도가 매우 적거나 전혀 없다는 것이 밝혀짐

완전연결계층

- 완전연결계층(FC계층)은 행렬 곱셈을 이용하여 계산한 후 편향을 더함으로써 계산
- FC계층과 컨볼루션 계층은 모두 내적을 계산하므로 기능적 형태가 동일하기 때문에 FC계층과 컨볼루션 계층 사이 변환이 가능
- 어떤 컨볼루션 계층에 대해서도 동일한 순방향 기능을 구현하는 FC계층이 있음
- 가중치 행렬은 대부분 0인 큰 행렬이지만, 국소적 연결로 인해 많은 블록들의 가중치가 같음

완전연결계층

- 어떤 FC 계층도 컨볼루션 계층으로 변환 가능
  - 예) K=4096인 FC 계층이 7x7x512 입력 볼륨을 보는 상황
  - 이는 F=7, P=0, S=1, K=4096인 컨볼루션 계층으로 동등하게 표현 가능
  - 즉, 필터 크기를 입력 볼륨의 크기로 정확히 설정하기 때문에, 출력은 1x1x4096이 되며, 이는 초기 FC 계층과 동일한 결과 제공

완전연결계층

- FC 계층 -> 컨볼루션 계층 변환은 실제로 매우 유용함
- 224x224x3 이미지를 입력으로 받아 일련의 컨볼루션 계층과 풀링 계층을 거쳐 최종적으로 7x7x512 볼륨으로 줄이는 컨볼루션 신경망을 생각
- AlexNet에서는 각각 2배로 다운샘플링하는 풀링 계층 5개 사용하며, 최종 공간적 크기는 224/2/2/2/2/2=7이 됨
- 크기가 4096인 2개의 FC계층을 사용하고, 마지막으로 클래스 점수를 계산하는 1000개 뉴런이 있는 FC 계층을 사용함
- 이 3개의 FC 계층 각각을 컨볼루션 계층으로 변환 가능

완전연결계층

- 7X7X512 볼륨을 보는 첫 번째 FC계층을 필터 크기 F=7을 사용하는 컨볼루션 계층으로 교체
  - 이를 통해 1X1X4096 볼륨이 생성됨
- 2번째 FC 계층을 필터 크기 F=1을 사용하는 컨볼루션 계층으로 교체
  - 이를 통해 1X1X4096 볼륨이 생성됨
- 마지막 FC계층 역시 F=1을 사용하여 교체
  - 최종 출력 1X1X1000이 생성됨
- 이런 변환을 통해 원래의 컨볼루션 신경망을 단일 순방향 패스에서 더 큰 이미지의 많은 공간적 위치에 매우 효율적으로 "슬라이딩"하게 함

완전연결계층

- 224x224 크기의 이미지가 7x7x512 크기 볼륨을 생성하는 상황 (32배 축소)
- 변환된 구조를 통해 384x384 크기 이미지를 전달하면 384/12이므로 12x12x512 크기의 동등한 볼륨을 얻음
- FC 계층에서 변환한 3개의 컨볼루션 계층을 계속 진행하면 (12-7)/1+1=6이므로 최종 볼륨의 크기가 6x6x1000이 됨
- 1x1x1000 크기의 단일 클래스 점수 벡터 대신, 이제는 384x384 이미지 전체에 걸친 6x6x배열의 클래스 점수를 얻고 있음

완전연결계층

- 아래의 과정으로 이해 가능
  - 384x384 이미지에서 224x224 부분을 선택
  - 그 이미지에 대해 원래 컨볼루션 신경망을 통과시켜 1x1x1000 생성
  - 그 후 32픽셀 이동시킨 위치에서의 224x224 부분을 선택
  - 그 부분에 대해 원래 컨볼루션 신경망을 통과시켜 1x1x1000 생성
  - 이를 총 36번 수행하면 1x1x1000이 36개 생성되므로 총 6x6x1000을 얻음
- 이런 과정은 변환된 컨볼루션 신경망의 한번 사용과 동일한 결과를 제공

완전연결계층

- 변환된 컨볼루션 신경망을 한번 전달하는 것이, 원래 컨볼루션 신경망을 모든 36개 위치에서 반복하는 것보다 효율적
- 이 트릭은 실제로 더 나은 성능을 얻기 위해 자주 사용됨
- 예를 들어, 이미지를 확대하여 변환된 컨볼루션 신경망을 사용해 많은 공간적 위치에서 클래스 점수를 평가한 후 클래스 점수를 평균내는 것이 일반적
- 원래 컨볼루션 신경망을 32 보다 더 작은 간격으로 이동하면서 뽑아 평가하는 것도 가능
  - 여러 번의 순방향 패스를 통해 달성 가능

컨볼루션 신경망 아키텍처

- 컨볼루션 신경망은 일반적으로 3가지 유형의 계층으로 구성됨
  - 컨볼루션 계층
  - 풀링 계층
  - FC 계층
- 또한, ReLU 활성화 함수를 사용

컨볼루션 신경망 아키텍쳐 – 계층 패턴

- 컨볼루션 신경망의 일반적인 형태는 컨볼루션-ReLU 계층 몇 개 쌓은 후 풀링 계층을 수행하고, 이미지가 작은 크기로 될 때까지 이 패턴을 반복하는 것
- 어느 시점에서 완전연결계층으로 전환
- 완전연결계층은 출력(클래스 점수)을 가짐
  - 입력 -> [[CONV -> RELU]*N -> POOL?]*M -> [FC -> RELU]*K -> FC
  - N>=0 (보통 N<=3), M>=0, K>=0 (보통 K<3)

컨볼루션 신경망 아키텍쳐 – 계층 패턴

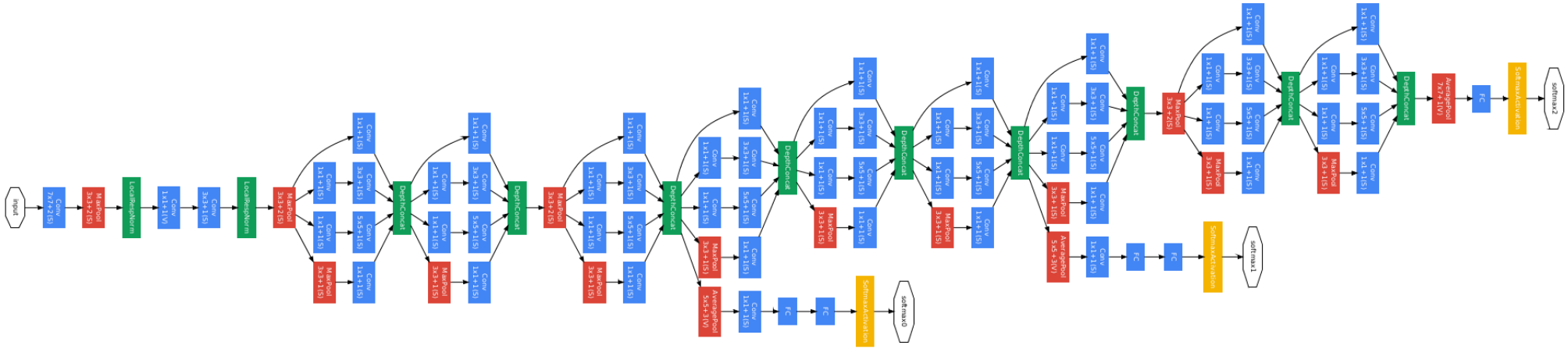
- 일반적인 컨볼루션 신경망 아키텍처 예시
  - 입력 -> FC
  - 입력 -> CONV -> RELU -> FC
  - 입력 -> [CONV -> RELU -> POOL]*2 -> FC -> RELU -> FC
  - 입력 -> [CONV -> RELU -> CONV -> RELU -> POOL]*3 -> [FC -> RELU]*2 -> FC
- 마지막 예시는 일반적으로 더 크고 깊은 신경망에서 좋음
  - 파괴적인 풀링 연산 전에 컨볼루션들이 더 복잡한 특징 개발 가능

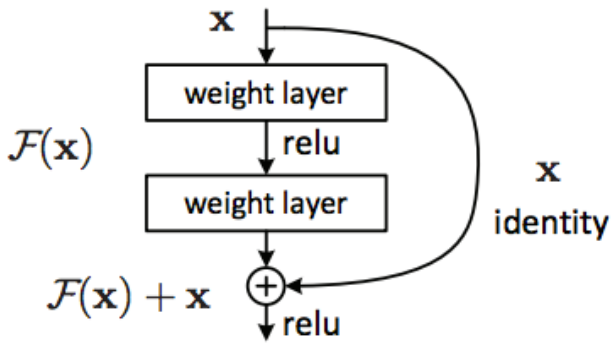
컨볼루션 신경망 아키텍쳐 – 계층 패턴

- 작은 필터의 컨볼루션 계층 스택이 큰 수용 필드의 컨볼루션 계층 하나보다 더 좋음
- 예를 들어, 3개의 3x3 컨볼루션 계층과 하나의 7x7 컨볼루션 계층은 둘 다 입력 볼륨에 대해 7x7 뷰(view)를 가지지만, 단일 7x7 컨볼루션 계층은 여러 단점들을 가짐
  - 3개의 컨볼루션 계층 스택에는 특징을 더 표현력 있게 만드는 비선형함수들을 포함
  - 모든 볼륨이 C 채널을 가질 때, 단일 7x7 컨볼루션 계층은 Cx(7x7xC)=49C^2 개의 파라미터를 포함
  - 3개의 3x3 컨볼루션 계층은 3x(Cx3x3xC) = 27C^2 개의 파라미터만을 포함
- 역전파 시 중간 컨볼루션 계층 결과의 저장을 위해 더 많은 메모리가 필요

컨볼루션 신경망 아키텍쳐 – 계층 패턴

- 최근 변화
  - 기존의 계층들의 선형적인 배열 패러다임이 최근 변화됨
  - ex) Inception(구글), Residual Networks (Microsoft Research Asia, SOTA)





컨볼루션 신경망 아키텍쳐 – 계층 패턴

- 실제 사용
  - 대부분의 응용에서는 아키텍처 고민이 필요 없음
  - 문제에 대한 자체 아키텍처를 개발하는 대신 현재 ImageNet 데이터셋에서 가장 잘 작동하는 아키텍쳐를 먼저 찾음
  - 그 후 사전훈련된(pre-trained) 모델을 다운로드하여, 자신의 데이터에 맞게 미세조정(fine-tuning)함
  - 컨볼루션 신경망을 처음부터 훈련하거나 처음부터 설계할 필요는 거의 없어야함

컨볼루션 신경망 아키텍처 – 계층 패턴

- ImageNet
  - 대표적인 대규모 데이터셋
  - 1,000개의 클래스
  - 120만 개 이미지로 구성된 훈련 세트
  - 5만 개 이미지로 구성된 검증 세트
  - ILSVRC(ImageNet Large Scale Visual Recognition Challenge): 유명한 ImageNet 분류 국제 대회

컨볼루션 신경망 아키텍쳐 – 계층 패턴

- https://developer-together.tistory.com/49

	해상도	클래스 개수	학습 데이터 수	테스트 데이터 수
CIFAR-10	32 X 32 X 3	10개	50,000개 (클래스당 5,000개)	10,000개 (클래스당 1,000개)
CIFAR-100	32 X 32 X 3	100개	50,000개 (클래스당 500개)	10,000개 (클래스당 100개)
STL-10	96 X 96 X 3	10개	5,000개 (클래스당 500개)	8,000개 (클래스당 800개)
MNIST	28 X 28 X 1	10개	60,000개	10,000개
FASHION-MNIST	28 X 28 X 1	10개	60,000개	10,000개
SVHN	32 X 32 X 3	10개	73,257개	26,032개