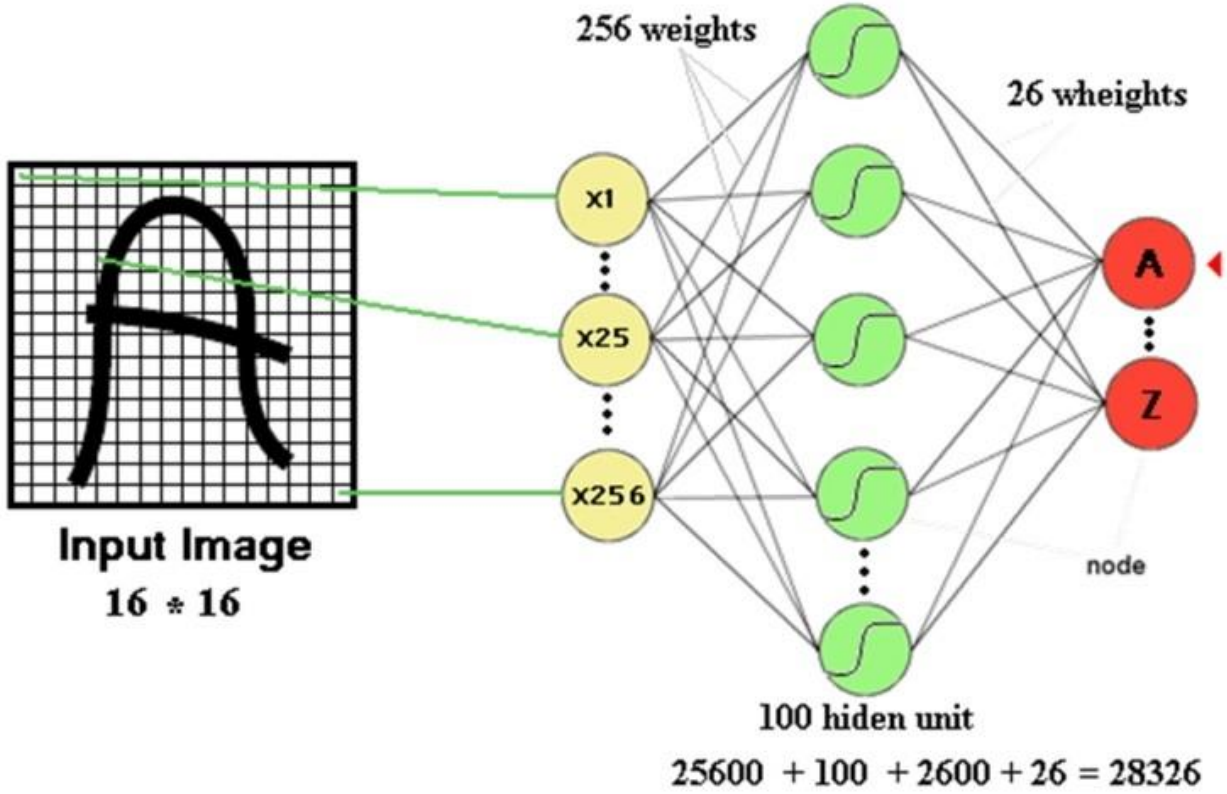
CNN 개요

기존 Multi-layered Neural Network의 문제점

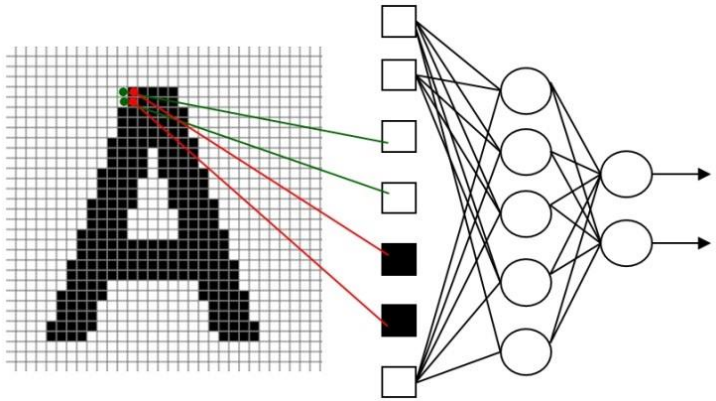
- 이상적인 머신 러닝 시스템이라면, 학습 데이터(training data)만 적절하게 많이 넣어주면, 알아서 분류까지 척척 해주는 것(unsupervised learning)을 기대할 것이다. 하지만 현실은 아직 그렇지 못하다.
- 그 대안이라면, 기존 지식(prior knowledge)를 이용해 네트워크의 구조를 특수한 형태로 변형을 시켜주는 것이다.
- 그렇다면 2-D 이미지가 갖는 특성을 최대한도로 살릴 수 있는 방법은 무엇일까?
- 아래 그림처럼, multi-layered neural network를 이용해
  16x16 크기의 폰트를 갖는 필기체를 인식하는 경우를 살펴보자
- 필기체 인식을 위해, 위 그림처럼 256개의 입력단과
  100개의 hidden-layer 및 26개의 출력단으로 구성이 되면
  (이것은 hidden layer가 1개만 있는 비교적 단순한 구조),
  이 망에 필요한 가중치(weight)와 바이어스는
  총 28,326개가 필요하게 된다.
- 폰트의 크기가 커지거나 hidden layer가 2단 이상이거나,
  대소문자 구별이나 숫자까지 구별을 해야 한다면
  파라미터의 개수가 엄청나게 많아지게 된다.

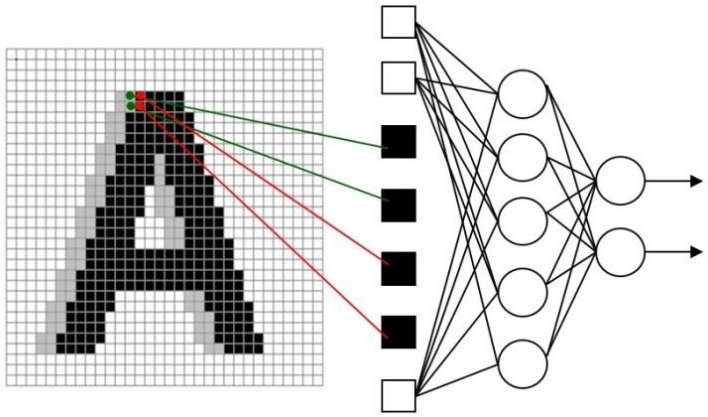


출처 : https://blog.naver.com/laonple/220587920012

기존 Multi-layered Neural Network의 문제점

- 기존 신경망은 아래 그림처럼, 전체 글자에서 단지 2 픽셀값만 달라지거나 2픽셀씩 이동만 하더라도 새로운 학습 데이터로 처리를 해줘야 하는 문제점이 있다..





- 또한 글자의 크기가 달라지거나,
  글자가 회전되거나,
  글자에 변형(distortion)이
  조금만 생기더라도
  새로운 학습 데이터를 넣어주지 않으면
  좋은 결과를 기대하기 어렵다.

출처 : https://blog.naver.com/laonple/220587920012

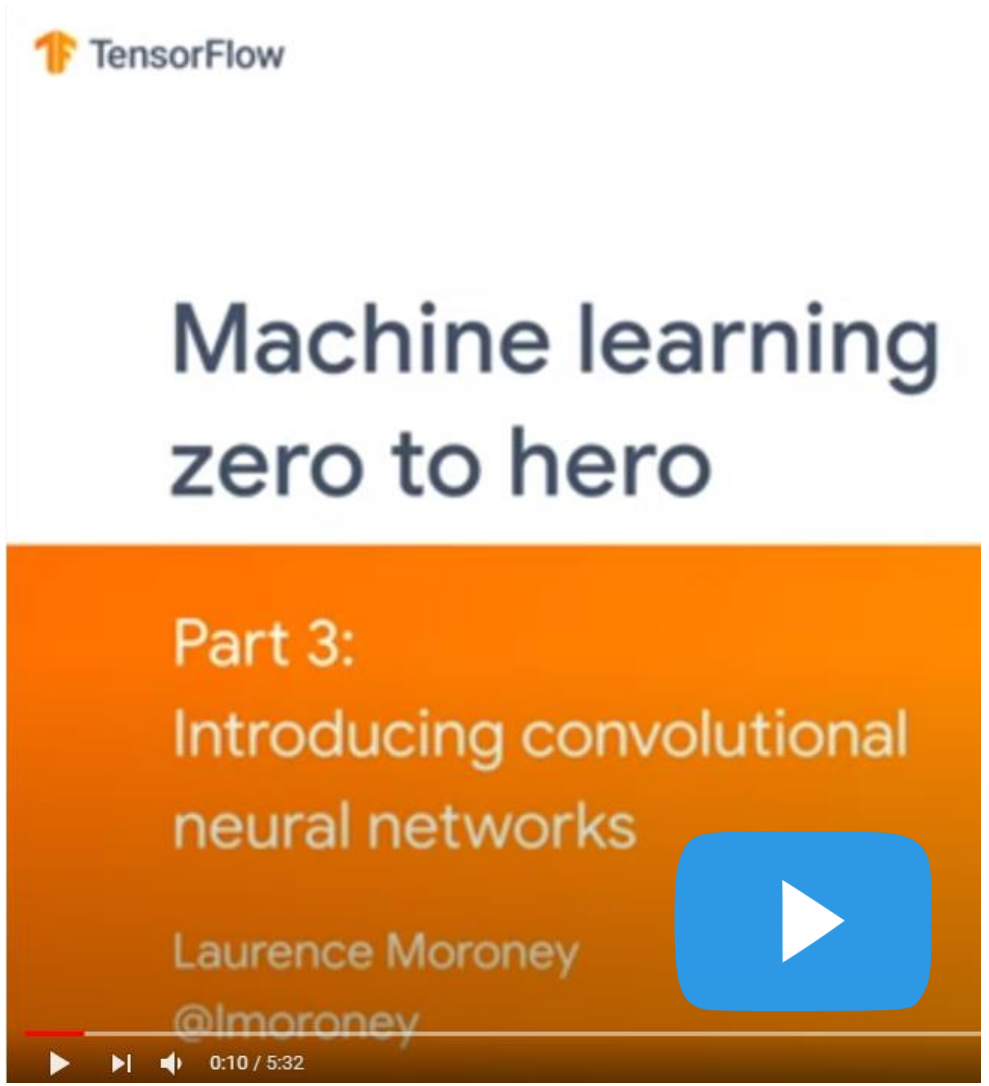
기존 Multi-layered Neural Network의 문제점

- 결과적으로 기존 multi-layered neural network는 글자의 topology는 고려하지 않고, 말 그대로 raw data에 대해 직접적으로 처리를 하기 때문에 엄청나게 많은 학습 데이터를 필요로 하고, 또한 거기에 따른 학습 시간을 대가로 지불해야 하는 문제점이 있다.
- 만약 32x32 폰트 크기에 대해 Black/White 패턴에 대해 처리를 한다면 결과적으로 232*32 = 21024개의 패턴이 나올 수 있고, 이것을 Gray-scale에 적용을 한다면 25632*32 = 2561024개의 어마어마한 패턴이 나오기 때문에, 전처리 과정 없이는 불가능하다라는 것을 미루어 짐작할 수 있다.
- 결과적으로 기존의 fully-connected multi-layered neural network를 사용하면, 3가지 측면에 문제가 발생함을 알 수 있다.
  - 학습 시간(Training time)
  - 망의 크기(Network size)
  - 변수의 개수(Number of free parameters)

출처 : https://blog.naver.com/laonple/220587920012

CNN 소개

출처 : (한국어) https://youtu.be/DkTtS2BEsxo



(영어) https://youtu.be/x_VrgWTKkiM

CNN 역사

- CNN은 1989년 LeCun이 발표한 논문 “Backpropagation applied to handwritten zip code recognition”에서 처음 소개가 되었고, 필기체 zip code 인식을 위한 프로젝트를 통해 개발이 되었습니다.

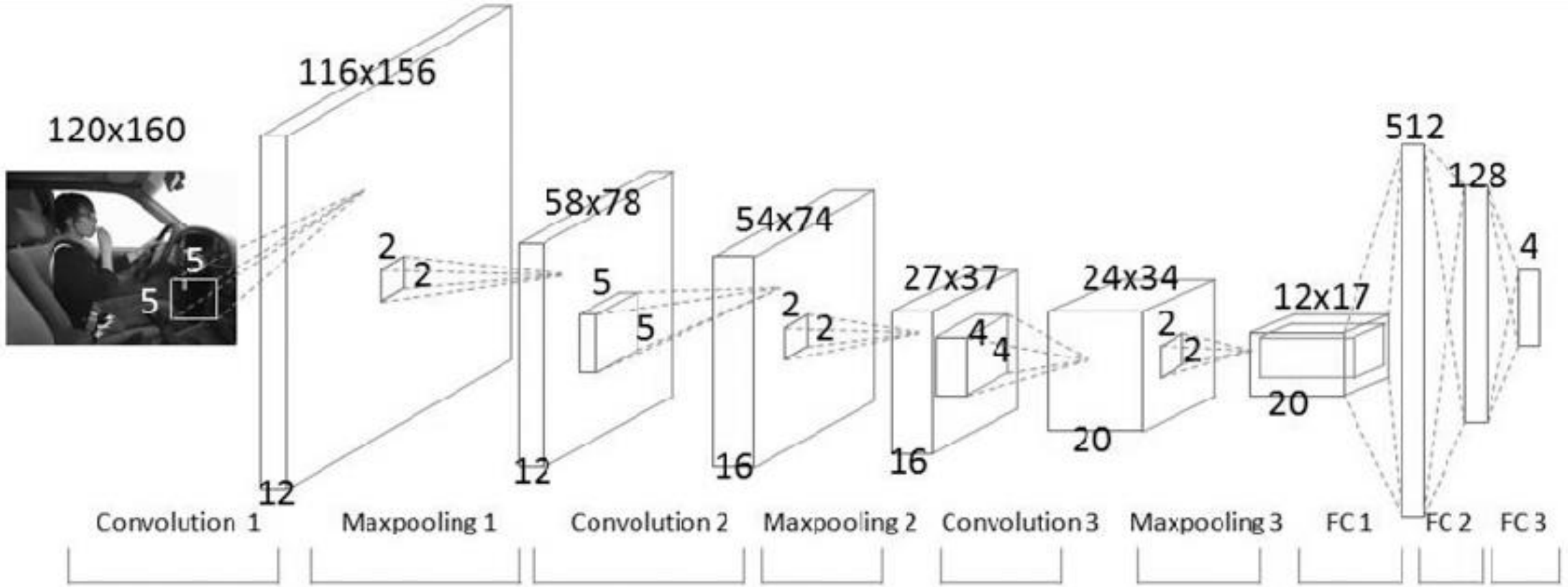
- 2003년 Behnke의 논문 “Hierarchical Neural Networks for Image Interpretation”을 통해 일반화가 되었으며, Simard의 논문 “Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis”을 통해 단순화되면서 개념 확대의 단초가 마련이 되었다.

- 이후 GP-GPU(General Purpose GPU)를 통해 CNN을 구현할 수 있는 방법이 소개 되었고, DBN(Deep Belief Network) 등 많은 분야에서 활발하게 연구 및 적용이 되고 있습니다.

출처 : https://blog.naver.com/laonple/220587920012

CNN 구조

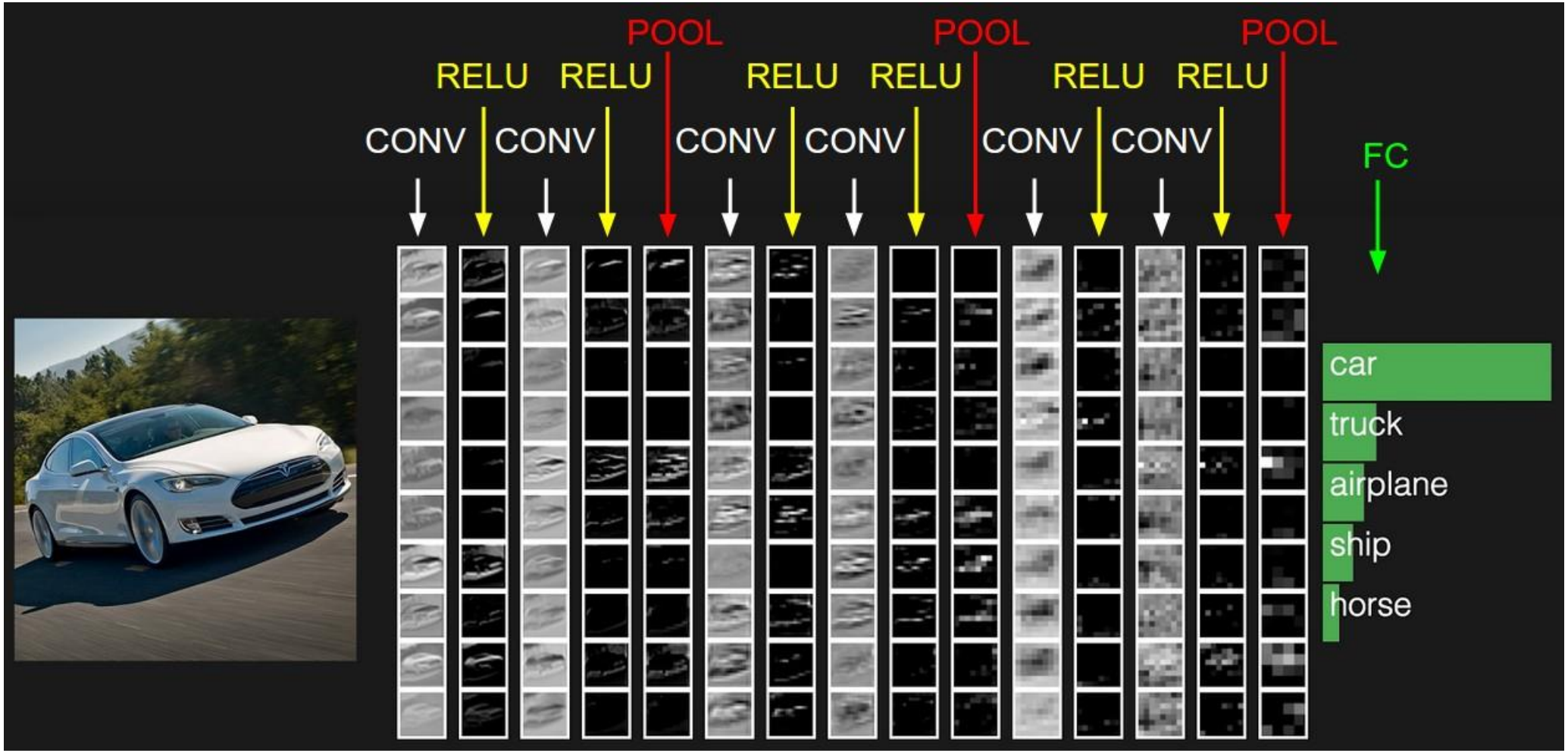
CNN은 Convolution Layer와 Max Pooling 레이어를 반복적으로 stack을 쌓는 특징 추출(Feature Extraction) 부분과 Fully Connected Layer를 구성하고 마지막 출력층에 Softmax를 적용한 분류 부분으로 나뉩니다.



출처 : http://taewan.kim/post/cnn/

CNN(Convolutional Neural Networks)

1차원 벡터정보를 입력으로 학습하는 Multi-Layer Neural Network는 변수의 개수, 네트워크 크기, 학습시간의 문제가 발생할 수 있으며, 공간적 구조를 보존하기 어렵습니다. 이를 해결하고자 만들어진 것이 합성곱 계층(Conv Layer)입니다.

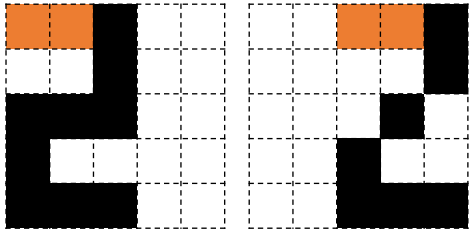


출처 : http://cs231n.github.io/convolutional-networks/

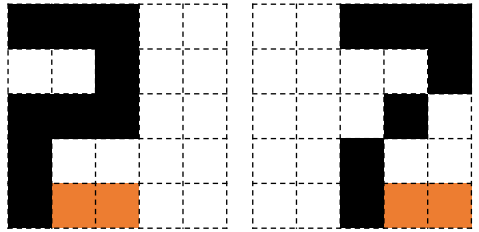
CNN 원리

CNN은 뇌가 사물을 구별하듯 생김새 정보로 사물을 학습하고 구별해 낸다.

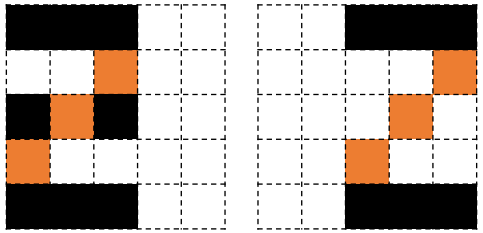
■ 숫자 2에서 공통적으로 얻을 수 있는 생김새 정보



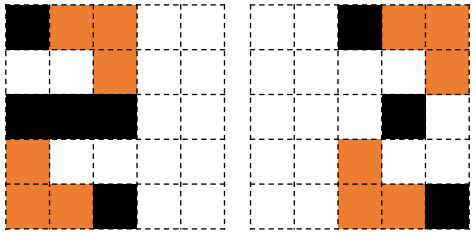
머리



꼬리

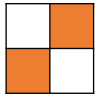


이음새

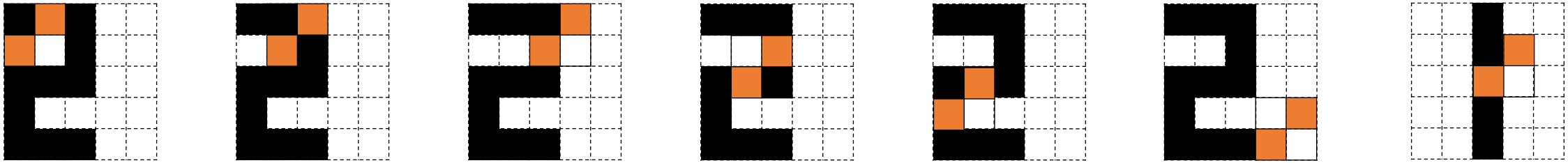


모서리

■ CNN은 어떻게 특징을 찾아 내는가?



필터(커널)



대각선 필터는 숫자 2로부터 두 곳의 대각선 특징을 감지하지만, 숫자 1에서는 대각선 특징을 발견하지 못한다.

출처 : [도서] 나의 첫 머신러닝 딥러닝, 위키북스

CNN 원리

■ 필터(Filter)

0	255
255	0

X

255	255	255	0	0
0	0	255	0	0
255	255	255	0	0
255	0	0	0	0
255	255	255	0	0

= 0*0 + 255*255 + 255*255 + 0*255

= 130050

0	255
255	0

X

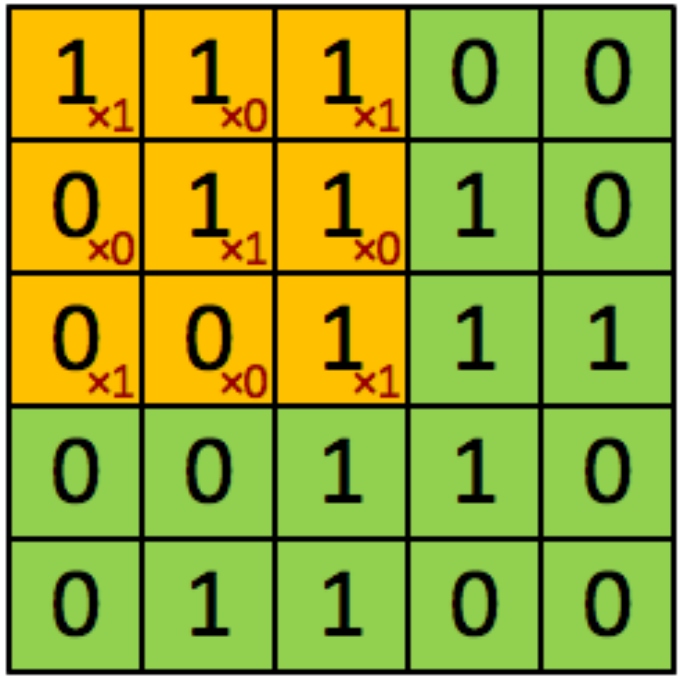
255	255	255	0	0
0	0	255	0	0
255	255	255	0	0
255	0	0	0	0
255	255	255	0	0

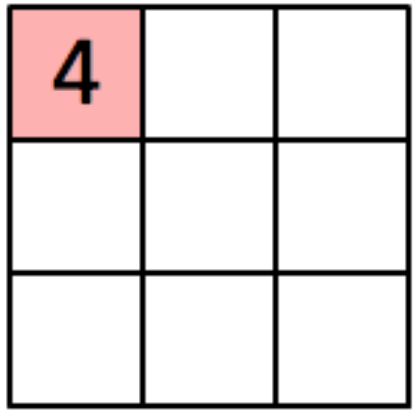
= 0*255 + 255*0 + 255*0 + 0*0

= 0

출처 : [도서] 나의 첫 머신러닝 딥러닝, 위키북스

합성곱 처리 절차

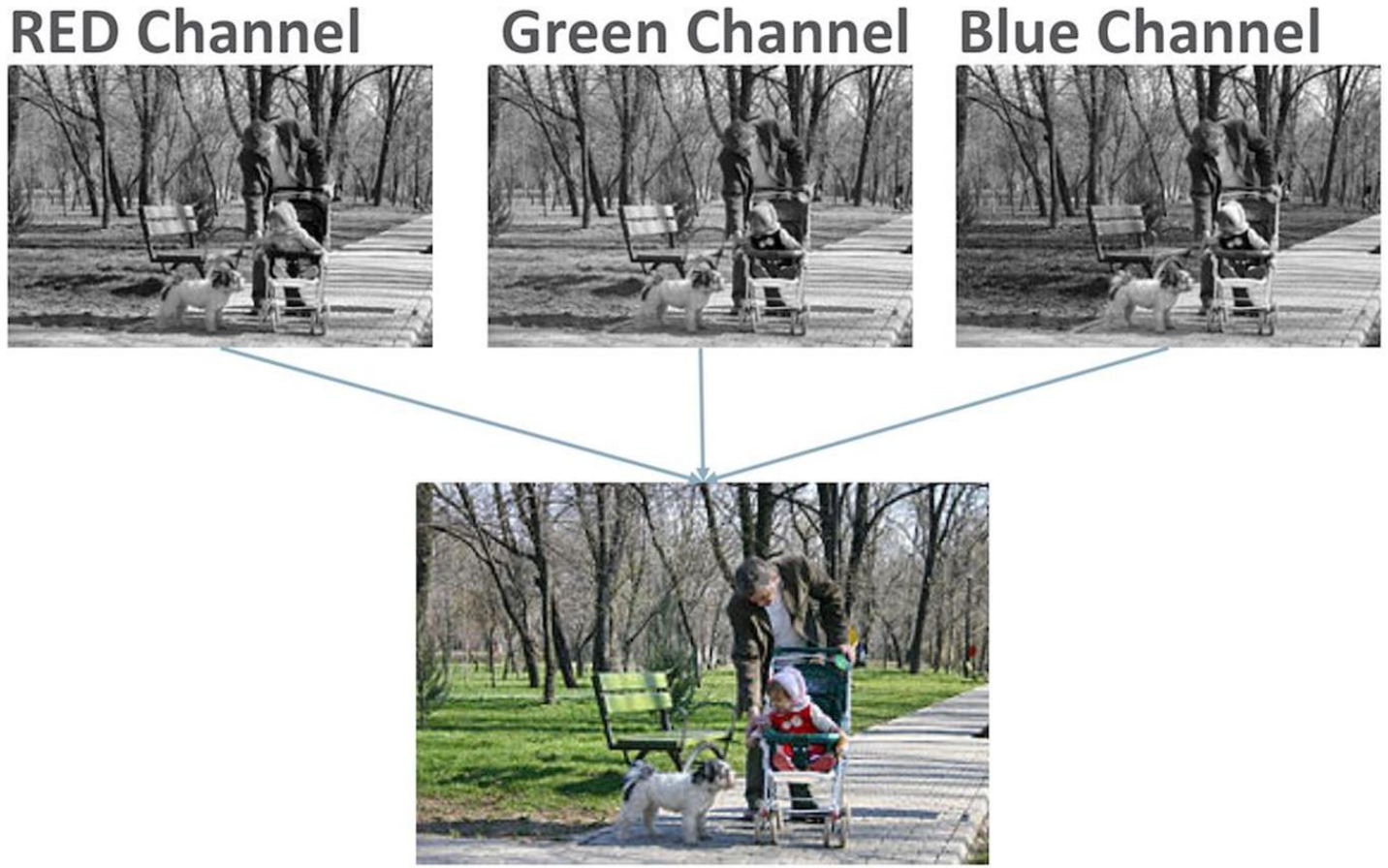




출처 : http://taewan.kim/post/cnn/

채널(Channel)

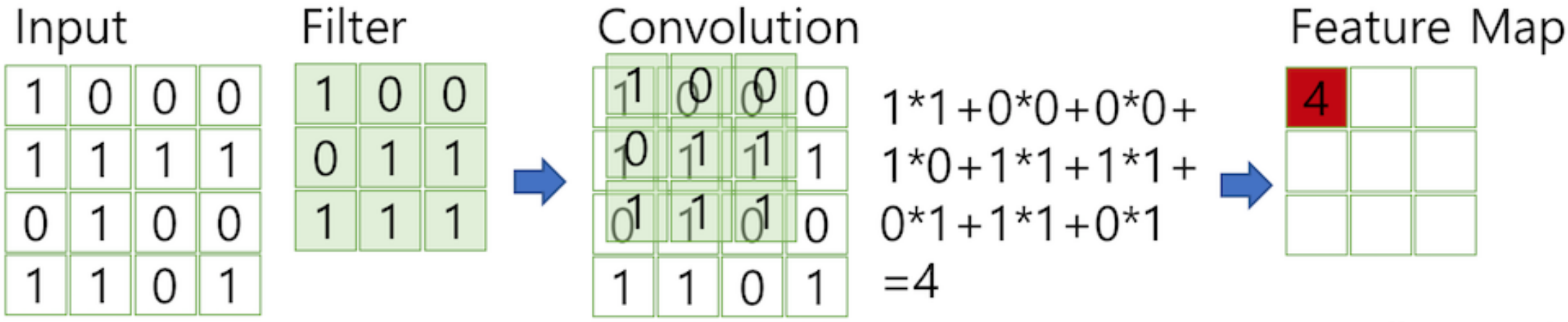
컬러 사진은 천연색을 표현하기 위해서, 각 픽셀을 RGB 3개의 실수로 표현한 3차원 데이터입니다. 컬러 이미지는 3개의 채널로 구성됩니다. 반면에 흑백 명암만을 표현하는 흑백 사진은 2차원 데이터로 1개 채널로 구성됩니다.



출처 : https://en.wikipedia.org/wiki/Channel_(digital_image)

필터(Filter) & Stride

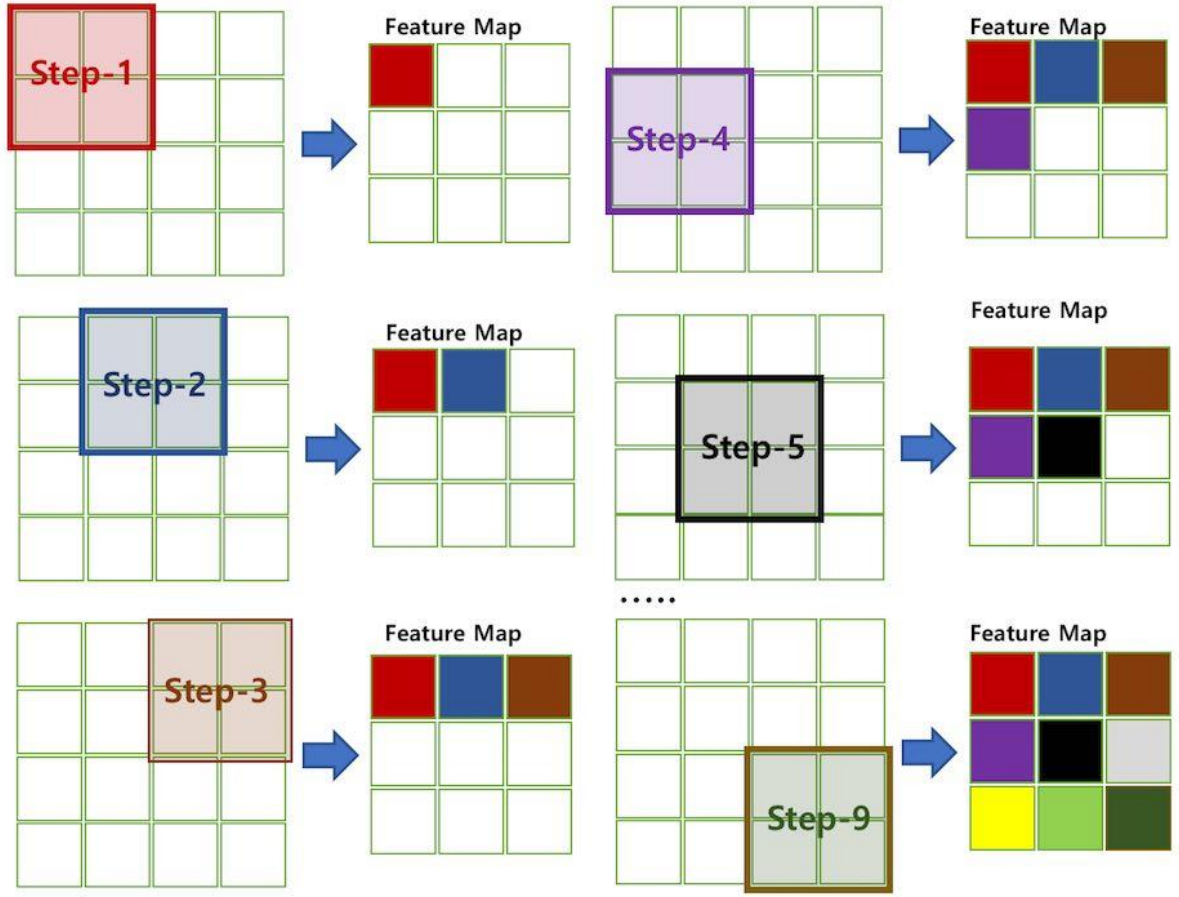
1필터는 이미지의 특징을 찾아내기 위한 공용 파라미터입니다. Filter를 Kernel이라고 하기도 합니다.



출처 : http://taewan.kim/post/cnn/

필터(Filter) & Stride

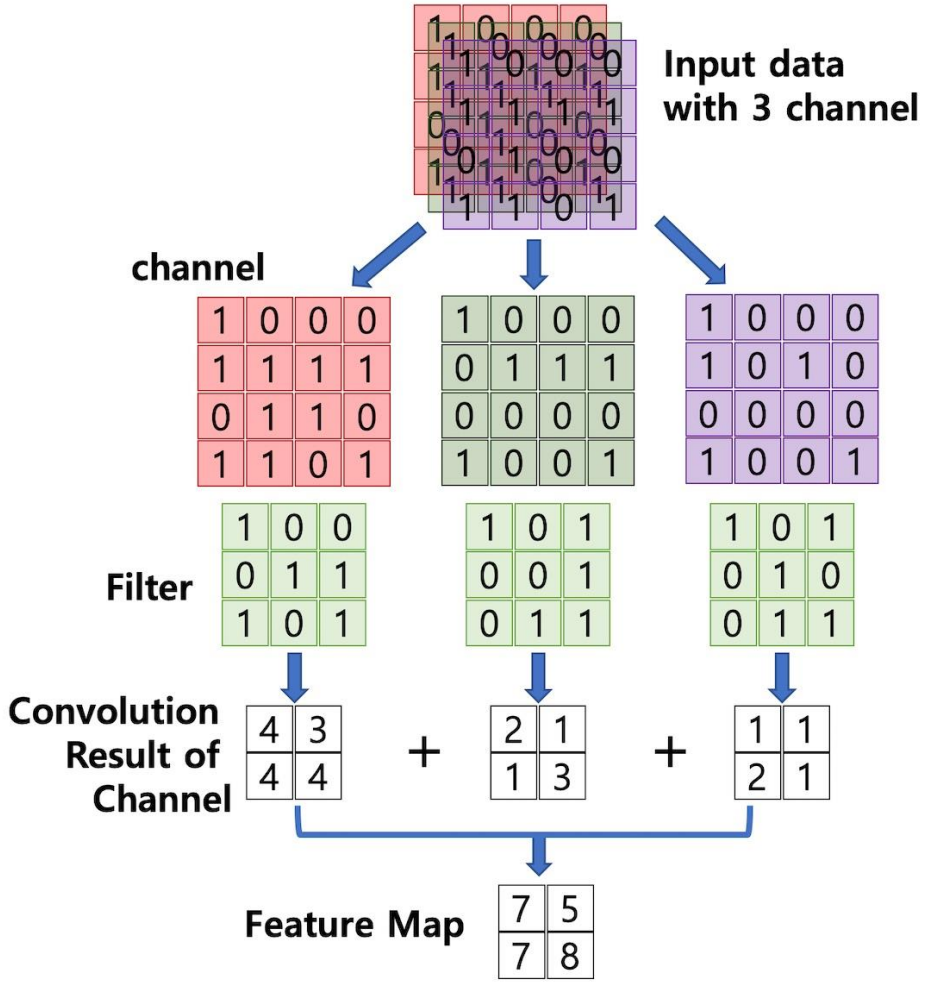
필터는 입력 데이터를 지정한 간격으로 순회하면서 합성곱을 계산합니다. 여기서 지정된 간격으로 필터를 순회하는 간격을 Stride라고 합니다.



출처 : http://taewan.kim/post/cnn/

필터(Filter) & Stride

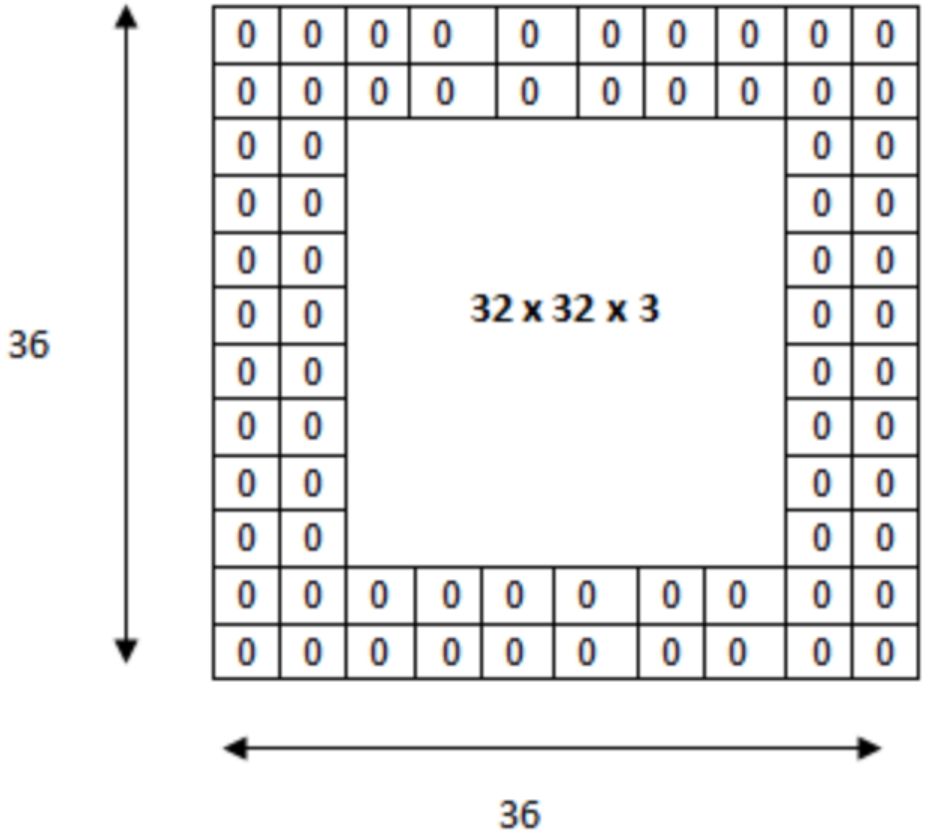
입력 데이터가 여러 채널을 갖을 경우 필터는 각 채널을 순회하며 합성곱을 계산한 후, 채널별 피처 맵을 만듭니다. 그리고 각 채널의 피처 맵을 합산하여 최종 피처 맵으로 반환합니다. 입력 데이터는 채널 수와 상관없이 필터 별로 1개의 피처 맵이 만들어 집니다.



출처 : http://taewan.kim/post/cnn/

패딩(Padding)

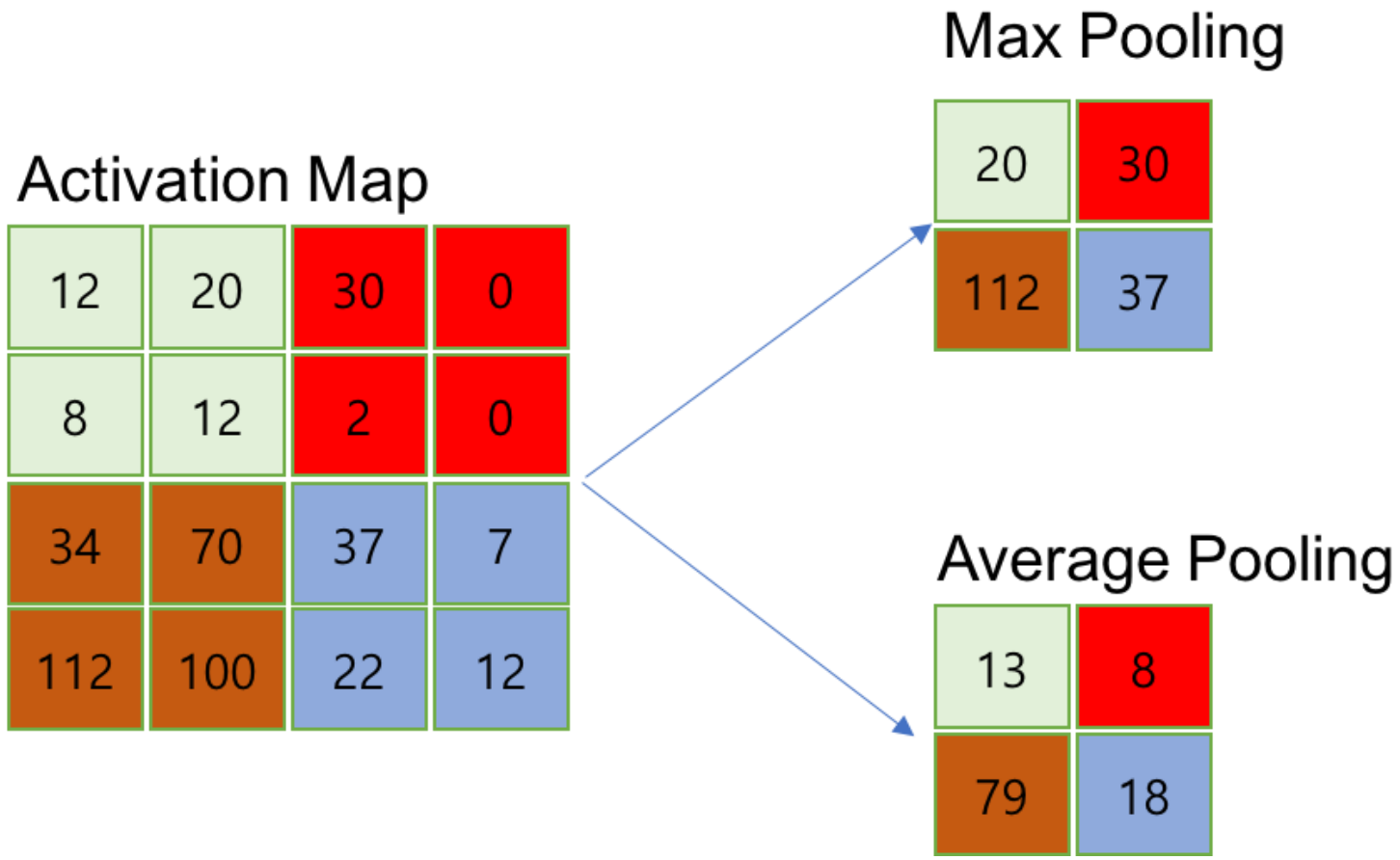
합성곱 레이어에서 Filter와 Stride에 작용으로 Feature Map 크기는 입력데이터 보다 작습니다. 합성곱 레이어의 출력 데이터가 줄어드는 것을 방지하는 방법이 패딩으로, 입력 데이터의 외각에 0으로 채워 넣는 것을 의미합니다.



출처 : http://taewan.kim/post/cnn/

Pooling 레이어

풀링 레이어는 출력 데이터(Activation Map)의 크기를 줄이거나 특정 데이터를 강조하는 용도로 사용됩니다.
일반적으로 Pooing 크기와 Stride를 같은 크기로 설정하여 모든 원소가 한 번씩 처리 되도록 설정합니다.



출처 : http://taewan.kim/post/cnn/

Thank you

박경규, 010-7240-1128
kgpark88@gmail.com