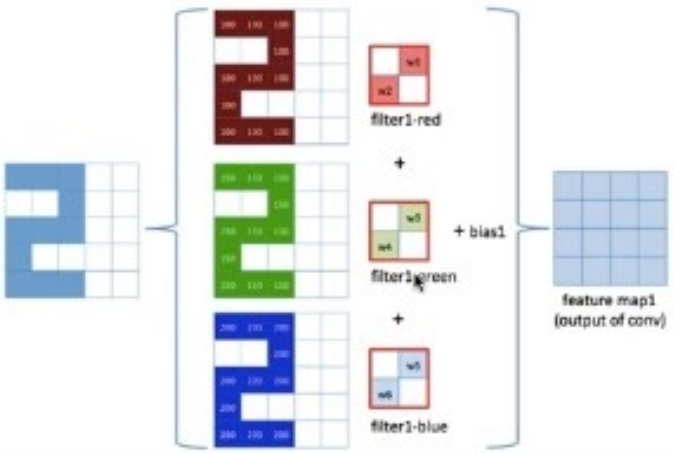
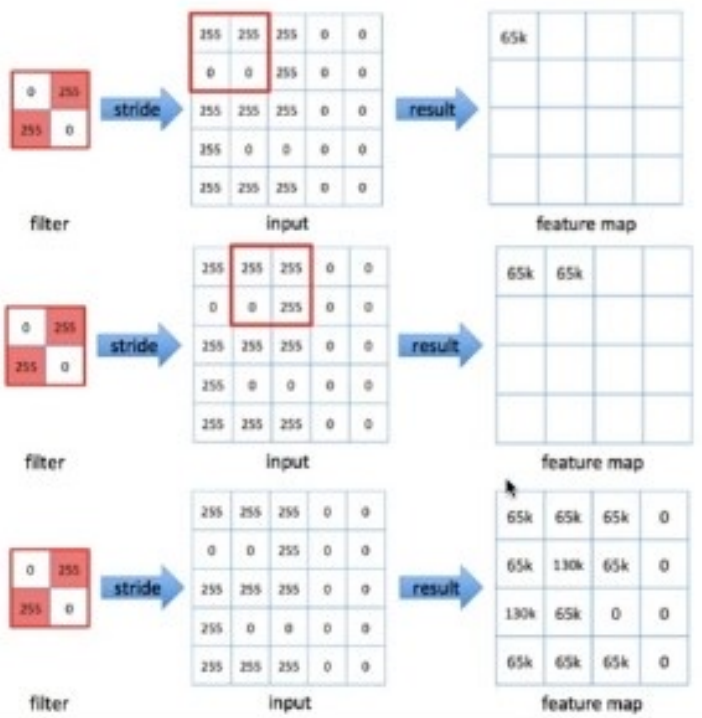
In [14]:
```
Image(url= "https://raw.githubusercontent.com/minsuk-heo/deeplearning/master/img/rgb1.png", width=
500, height=250)
```

Out[14]:



In [9]:
```
Image(url= "https://raw.githubusercontent.com/minsuk-heo/deeplearning/master/img/stride_result.pn
g", width=500, height=250)
```

Out[9]:



"feature map" 만드는 과정.

_ feature map 끼리

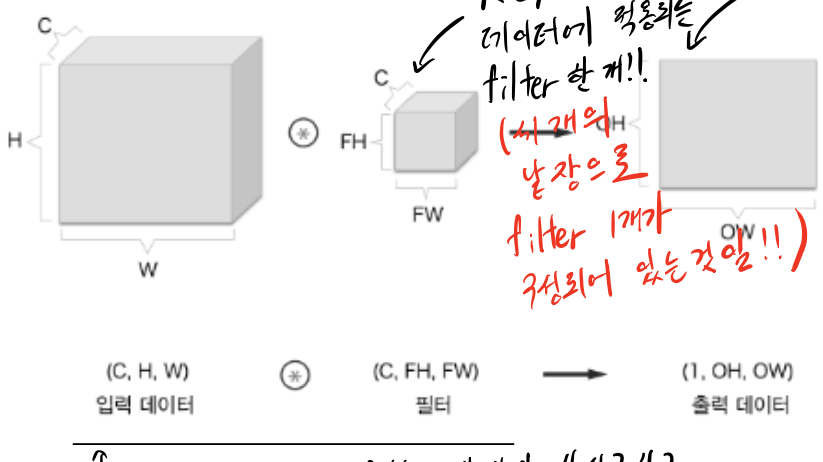
[출력크기 공식]

$$OH = \frac{H + 2P - FH}{S} + 1$$

$$OW = \frac{W + 2P - FW}{S} + 1$$

(입력크기를 H(Height),W(Width), 필터크기를 FH, FW, 출력크기를 OH, OW, 패딩을 P, 스트라이드를 S라고 한다.)

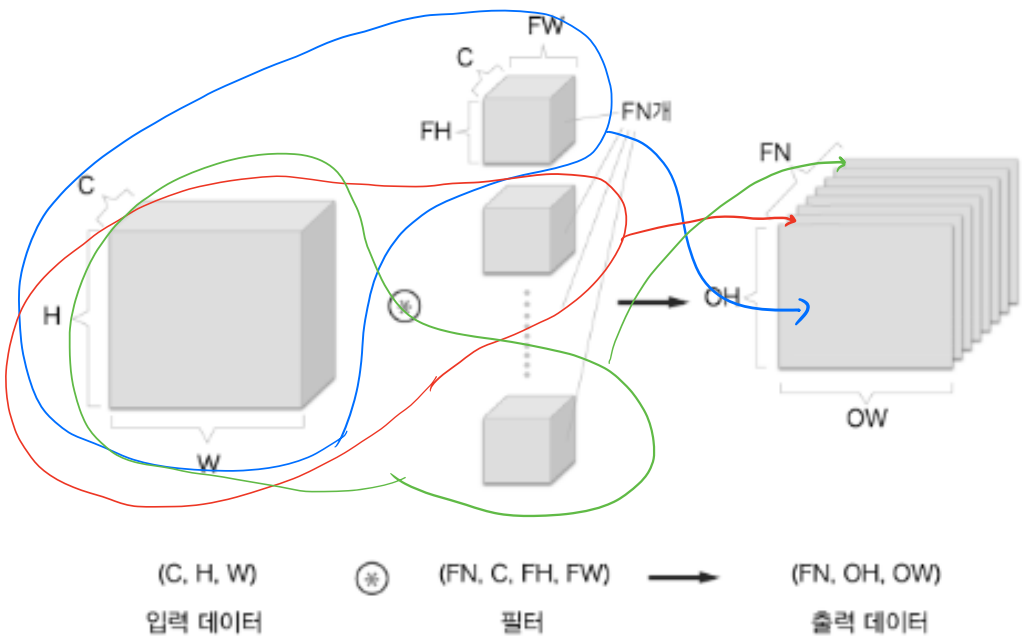
3차원 합성곱 구현 이론



[사진 한장을 RGB 필터로 합성곱하여 2차원 출력행렬(Feature map) 1을 출력한 그림]

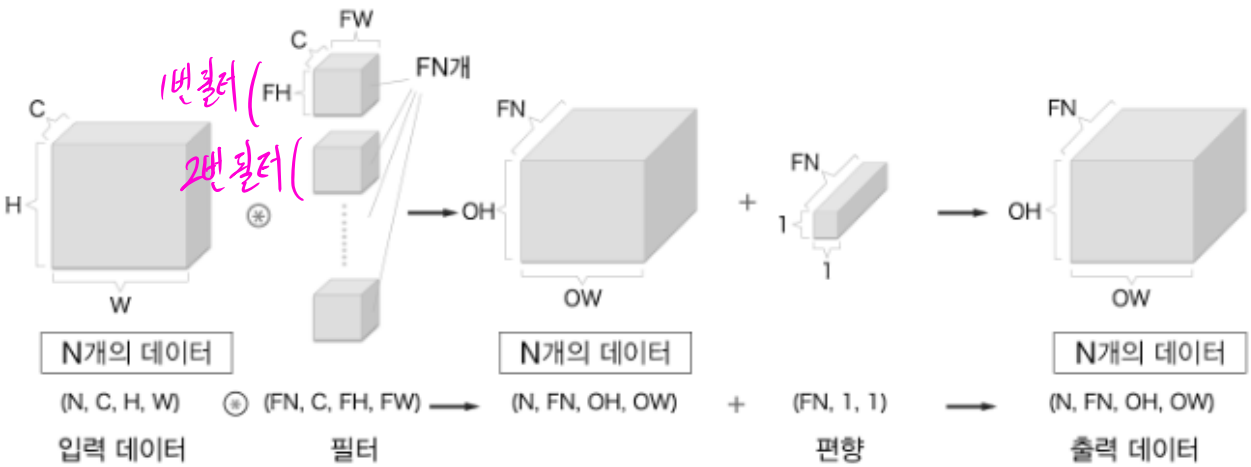
위의 그림은 Feature map이 한 개가 나오는데

실제로는 사진 한 장에 대해서 여러개의 Feature map이 필요합니다.



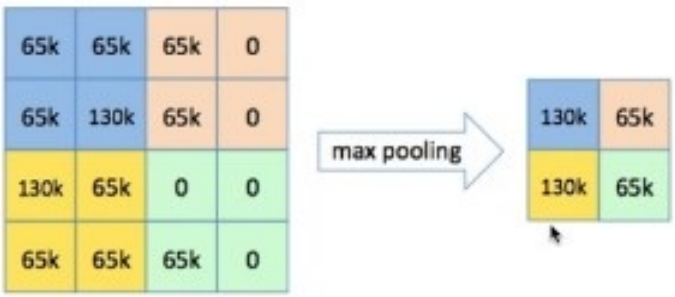
Filter의 갯수를 늘리면 여러개의 Feature map을 출력할 수 있습니다.

하나의 Filter에 하나의 Feature map을 출력할 수 있습니다.



```
In [10]: Image(url= "https://raw.githubusercontent.com/minsuk-heo/deeplearning/master/img/max_pool.png", width=500, height=250)
```

Out[10]:



max pooling의 효과!!

As you can see, the output of convolutional layer decreased from (4 x 4) to (2 x 2).

① Decreasing feature map results in reducing the number of parameters and computation time.

② Also by reducing the number of parameters, it gives control of **overfitting**.

zero padding

Lastly, I should give you insights of zero padding while it is not shown at Stanford CNN architecture diagram.
Zero padding is mostly applied on recent CNN with mainly two reasons below.

1) reduce information loss from convolutional layer.
2) let the CNN knows where is the boundary of the input.

Let's revisit convolutional layer. As you can see from below image,
The output dimension (4 x 4) is less than input dimension (5 x 5). That said, we lose some info at each convonlutional layer.

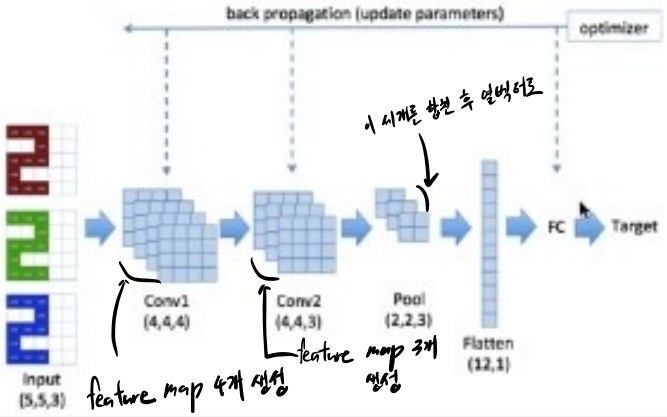
zero padding의 두가지 효과.

Train

Let's resume entire CNN architecture to talk about Training.

In [16]:
```
Image(url= "https://raw.githubusercontent.com/minsuk-heo/deeplearning/master/img/cnn_train.png", width=500, height=250)
```

Out[16]:



As you can see from above picture, if input is color image, the depth is 3, that is why you can see three layer at the input.

1. First Convolution layer has four filters that is why you have four layers at Conv1.
2. Second Convolution layer has three filters that is why you have three layers at Conv2. Pooling layer has stride size 2, that is why you have (2 x 2) feature maps.
3. Flatten will have 2 x 2 x 3 = 12 values in an array as an input to fully connected layer.

Theorically, convolution layers identify features.

Fully Connected Layer classify input using all identified features.

CNN is supervised learning, by giving target value, CNN will use back propagation to optimize parameters at convolution layers, fully connected layer.