



deeplearning.ai

Convolutional Neural Networks

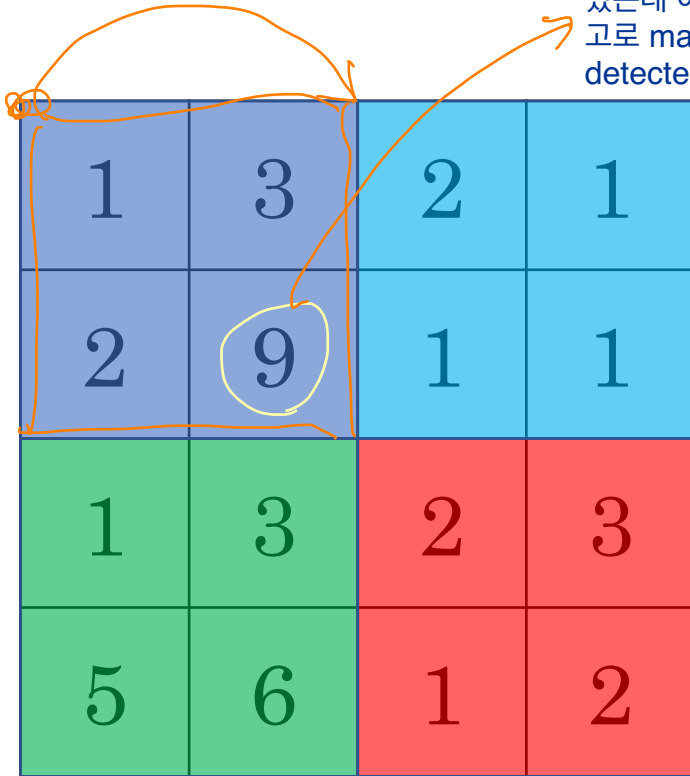
Pooling layers

사용하는 이유

- representation 크기를 줄여 계산속도 빠르게 하려고
- feature 를 좀 더 robust하게 디텍팅하게 해주려고

Pooling layer: Max pooling

this particular feature가 vertical edge일 수도, eye일 수도, cat일 수도 있는데 여튼 분명한건 그 feature가 왼쪽 상단 사분면에 존재한다는 점이다. 고로 max pooling의 효과는 feature가 필터 내 해당 영역 어디에서든지 detected 되면 맥스풀링 아웃풋에 보존해서 남겨두는 역할을 하는 것이다.

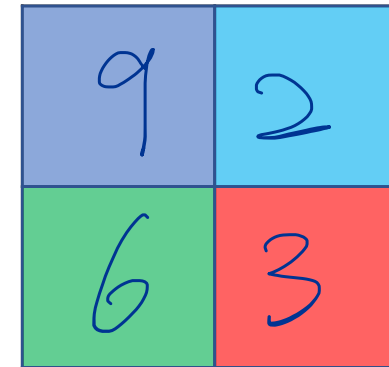


1	3	2	1
2	9	1	1
1	3	2	3
5	6	1	2

4x4

이 4x4 입력을 some set of features라고, the activations in some layer of the NN 이라고 생각해보자.

그러면 큰 숫자는 particular feature가 detected 된거라는 의미일 수 있다.



9	2
6	3

2x2

Hyperparameters
(of max pooling):

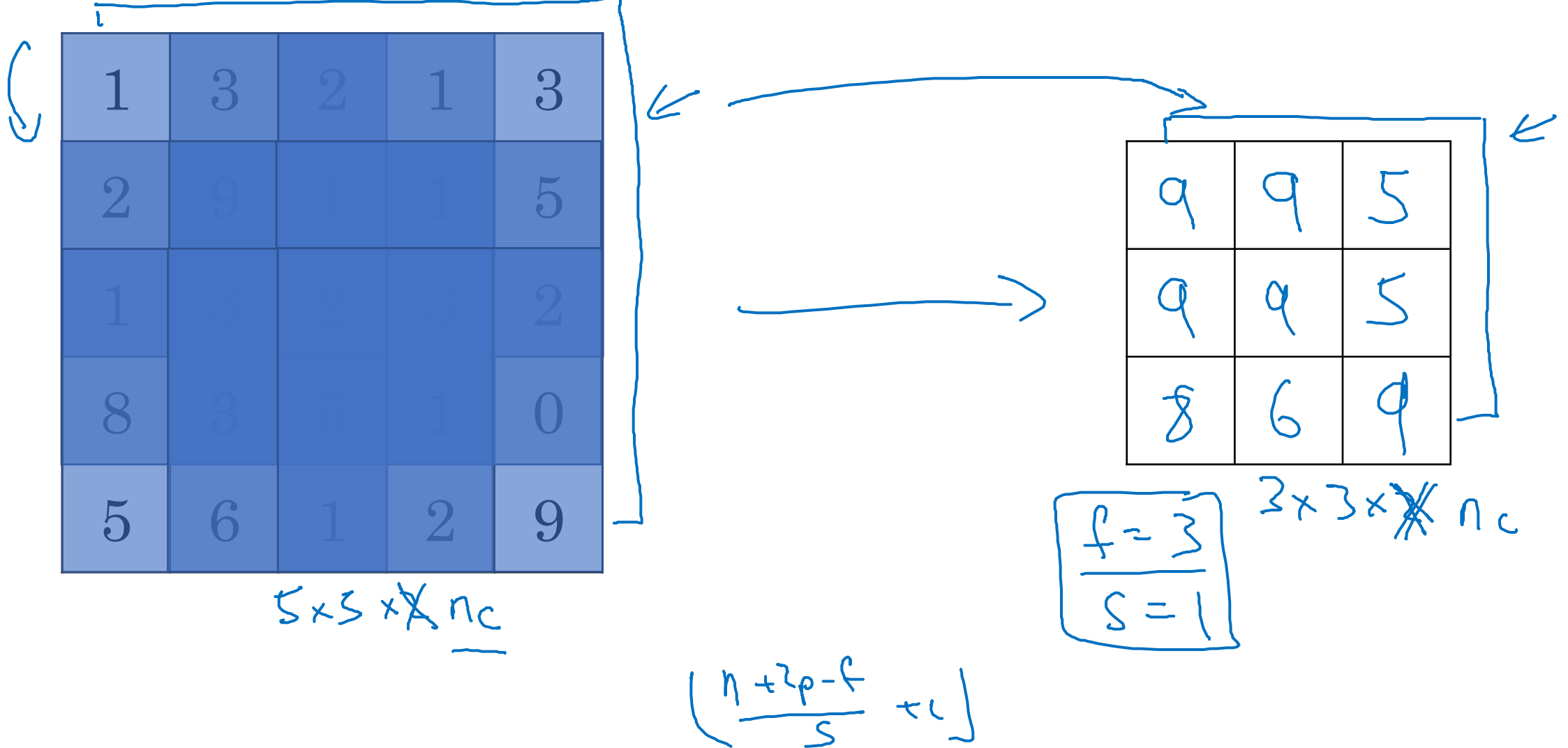
- $f = 2$
- $s = 2$

근데 솔직히 말하면 사람들이 맥스풀링을 사용하는 이유는 그냥 실험에서 결과가 잘 나와서 그런거지 뭔가 underlined reason이 있어서 그런건 아니다.

맥스풀링의 한가지 흥미로운 속성은 애는 hyperparameters는 가지지만 학습할 parameter는 없다는 점이다. f 랑 s 만 정해지면 gradient descent가 바꿀게 아무것도 없음.

Pooling layer: Max pooling

3D input의 경우엔 각 채널에 각각
independently 동일한 계산을 수행한다.



Pooling layer: Average pooling

1	3	2	1
2	9	1	1
1	4	2	3
5	6	1	2



3.75	1.25
4	2

$$f=2$$

$$s=2$$

$$7 \times 7 \times 1000 \rightarrow 1 \times 1 \times 1000$$

일반적으로는 맥스풀링이 더 많이 쓰이지만 매우 얇은 네트워크에서 위와 같이 collapse 하기 위해 average pooling을 사용하기도 한다.

Summary of pooling

Hyperparameters:

f : filter size

s : stride

Max or average pooling

일반적으로 쓰이는건 아래것들

$$f=2, s=2$$

$$f=3, s=2$$

→ ~~p: padding~~ is very very rarely used

No parameters to learn!

$$n_H \times n_W \times \underline{n_C}$$

↓

$$\left\lfloor \frac{n_H - f}{s} + 1 \right\rfloor \times \left\lfloor \frac{n_W - f}{s} + 1 \right\rfloor \times \underline{n_C}$$