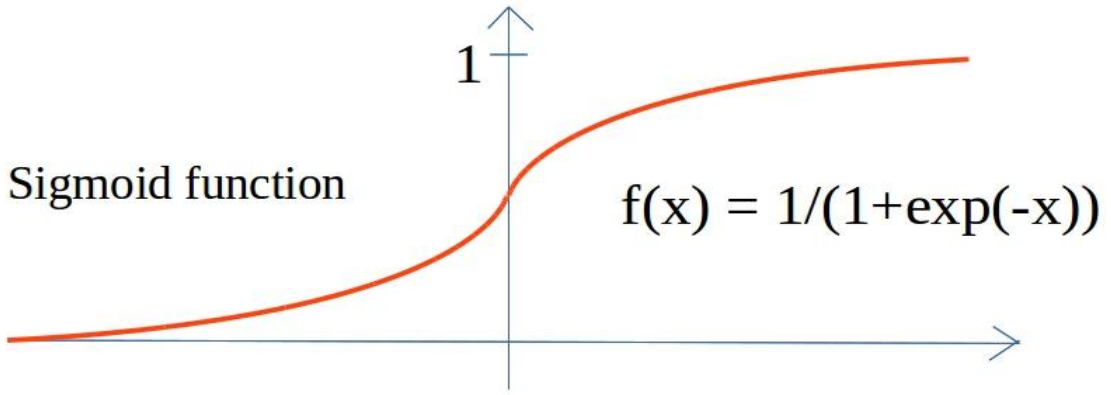
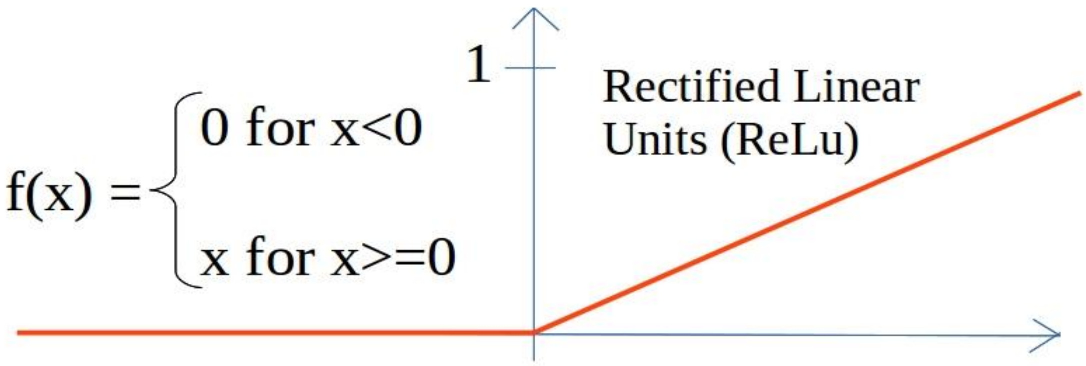
Add-ons to Neural Networks

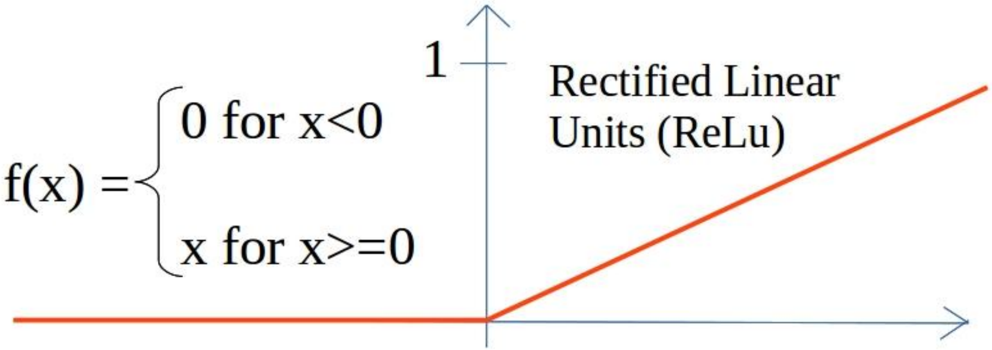
Activation function



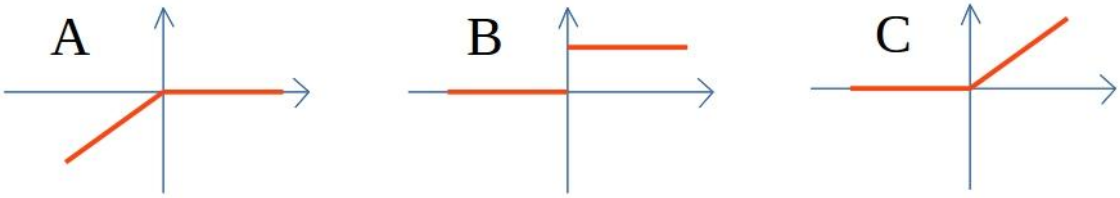


ReLu function

$$f(x) = \begin{cases} 0 \text{ for } x<0 \\ x \text{ for } x>=0 \end{cases}$$

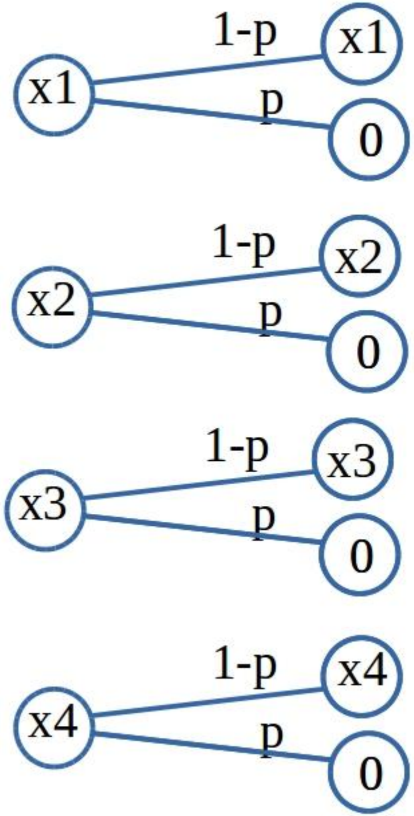


Derivative:



Dropout

Method to prevent overfitting



P(xi -> 0) = p

Dropout

Method to prevent overfitting

=0 =0

w111*x1 + w112*x2 + w113*x3 + w114*x4 = w111*x1 + w112*x2
w121*x1 + w122*x2 + w123*x3 + w124*x4 = w121*x1 + w122*x2
w131*x1 + w132*x2 + w133*x3 + w134*x4 = w131*x1 + w132*x2
w141*x1 + w142*x2 + w143*x3 + w144*x4 = w141*x1 + w142*x2

=0 =0

w111*x1 + w112*x2 + w113*x3 + w114*x4 = w112*x2 + w113*x3
w121*x1 + w122*x2 + w123*x3 + w124*x4 = w122*x2 + w123*x3
w131*x1 + w132*x2 + w133*x3 + w134*x4 = w132*x2 + w133*x3
w141*x1 + w142*x2 + w143*x3 + w144*x4 = w142*x2 + w143*x3

=0 =0

w111*x1 + w112*x2 + w113*x3 + w114*x4 = w112*x2 + w114*x4
w121*x1 + w122*x2 + w123*x3 + w124*x4 = w122*x2 + w124*x4
w131*x1 + w132*x2 + w133*x3 + w134*x4 = w132*x2 + w134*x4
w141*x1 + w142*x2 + w143*x3 + w144*x4 = w142*x2 + w144*x4

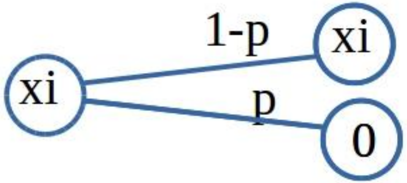
Same input

Different redundant representations

Dropout

Method to prevent overfitting

What happens with p if training is over and we use the model for prediction?



A We set p = 1

B We take same p as in training

C We set p = 0.5

D We set p = 0

Dropout

Method to prevent overfitting

w111*x1 + w112*x2 + w113*x3 + w114*x4
w121*x1 + w122*x2 + w123*x3 + w124*x4
w131*x1 + w132*x2 + w133*x3 + w134*x4
w141*x1 + w142*x2 + w143*x3 + w144*x4

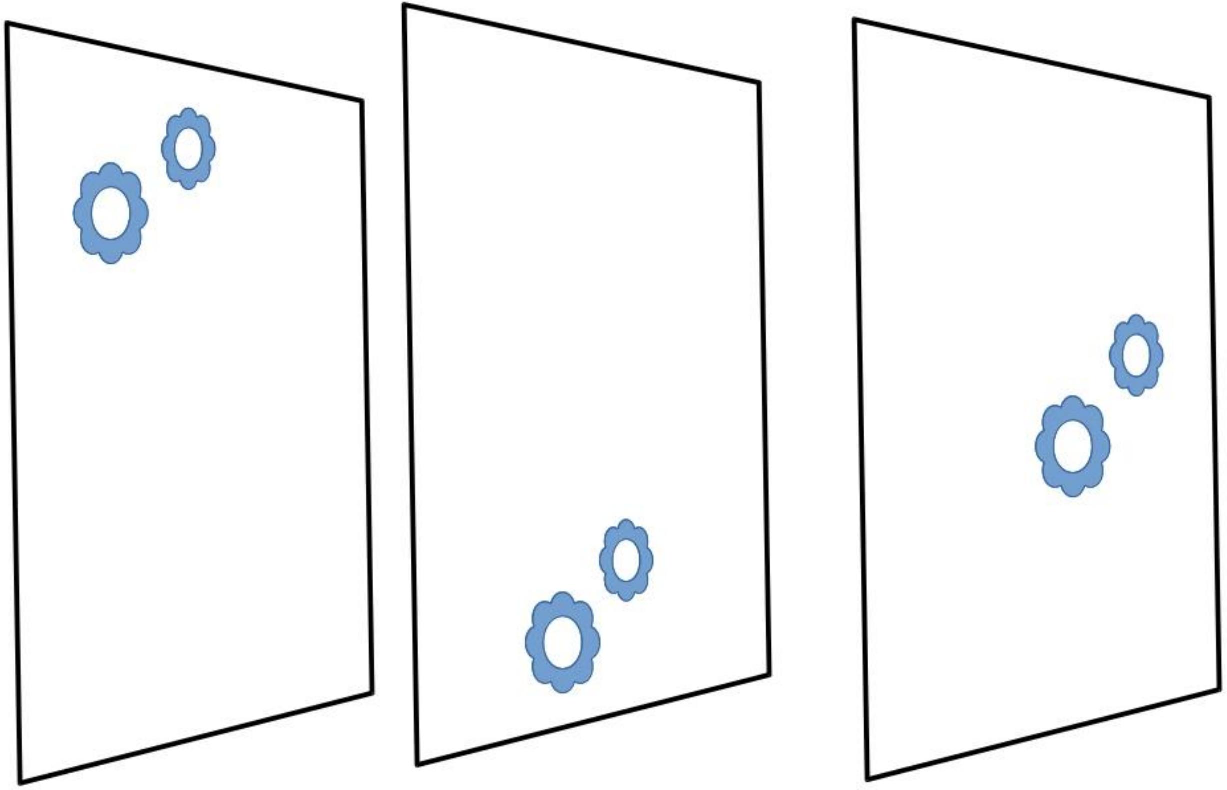
It's a hamster

I agree

Me too!

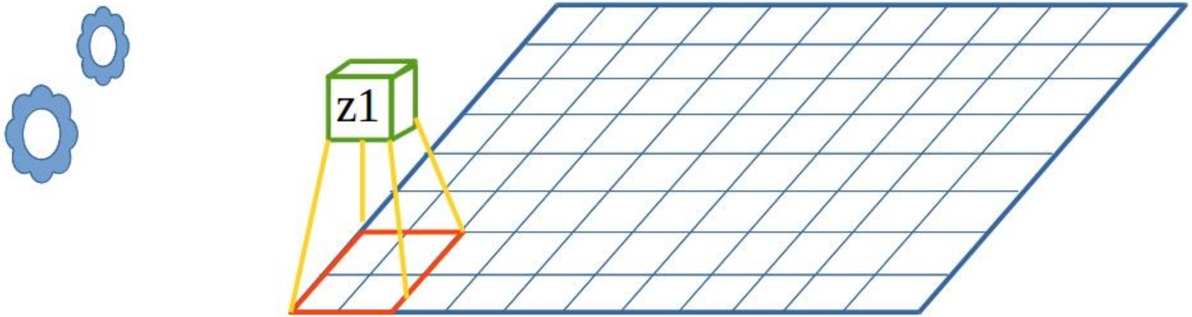
Clearly not a cat

Convolutions



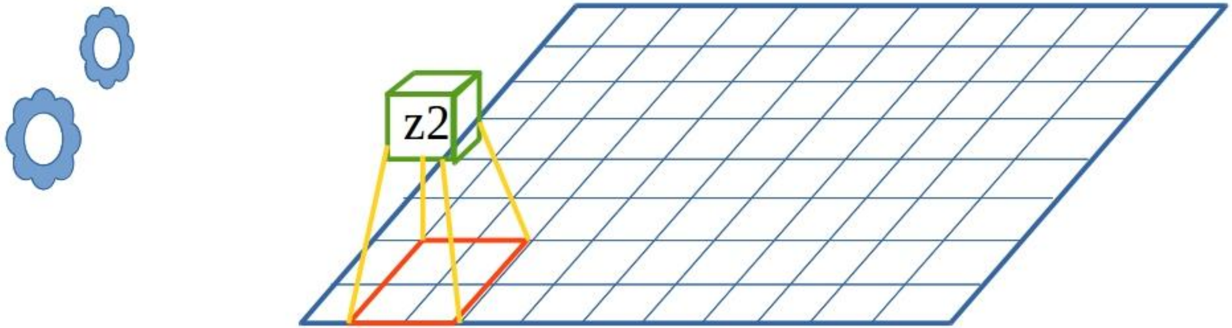
Convolutions

z1 = w1*x1 + w2*x2 + w3*x3 + w4*x4

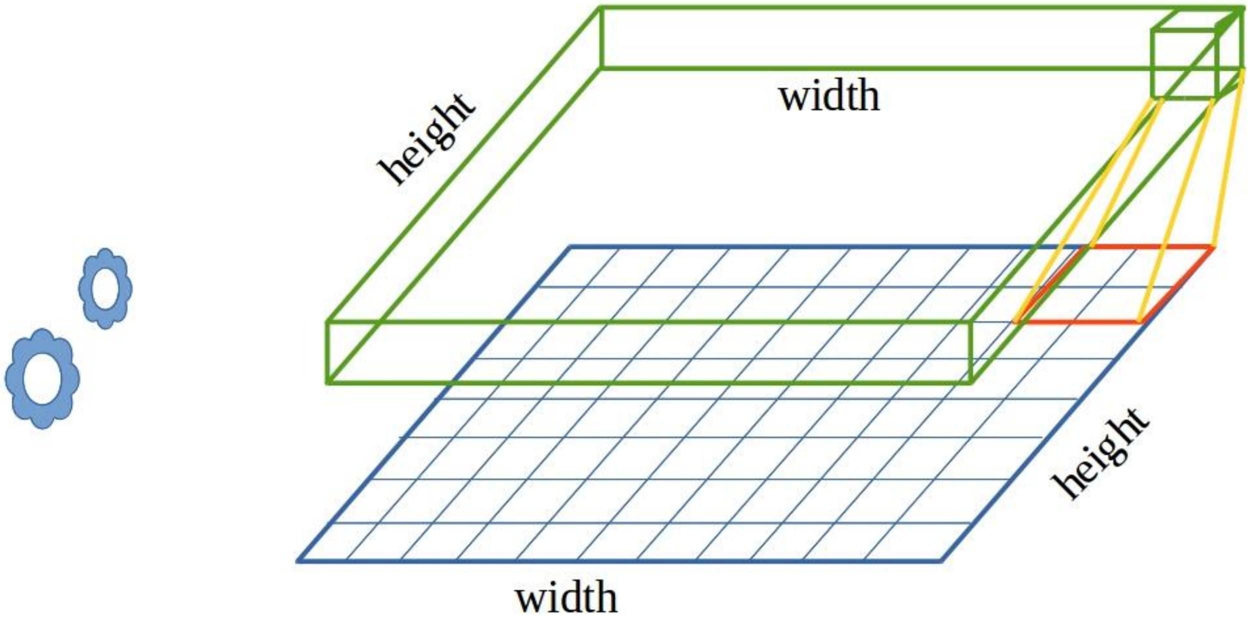


Convolutions

z2 = w1*x2 + w2*x3 + w3*x4 + w4*x5

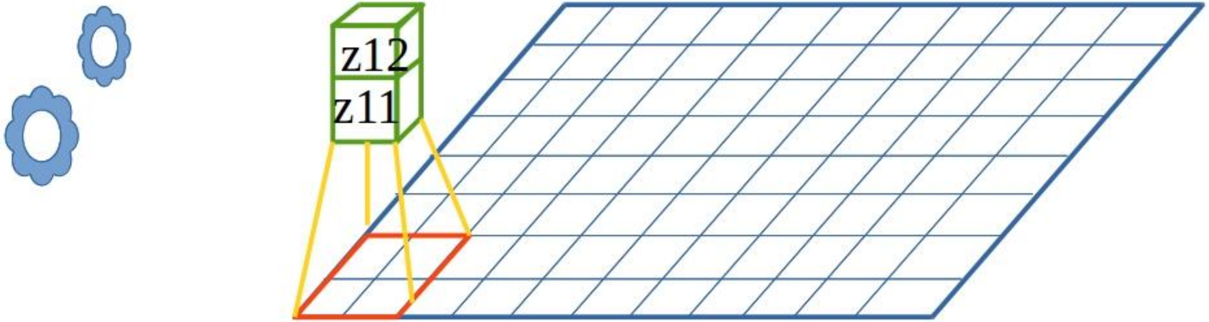


Convolutions



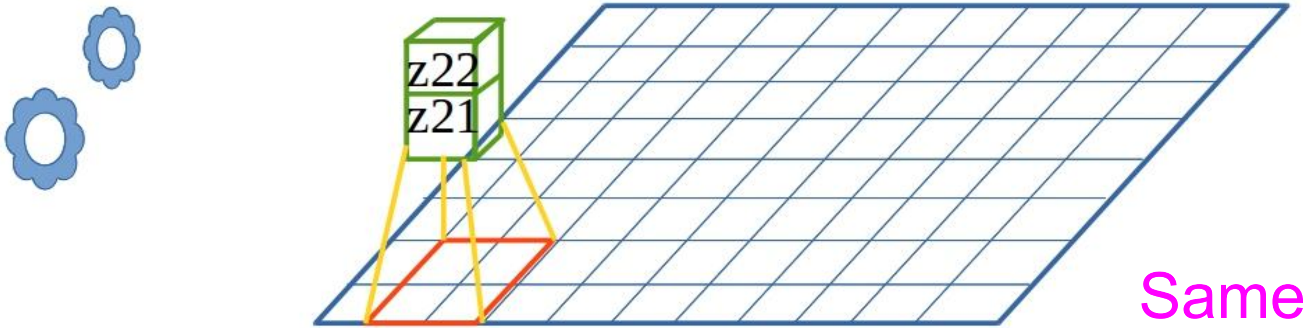
Convolutions

$$z11 = w11*x1 + w12*x2 + w13*x3 + w14*x4$$
$$z12 = w21*x1 + w22*x2 + w23*x3 + w24*x4$$



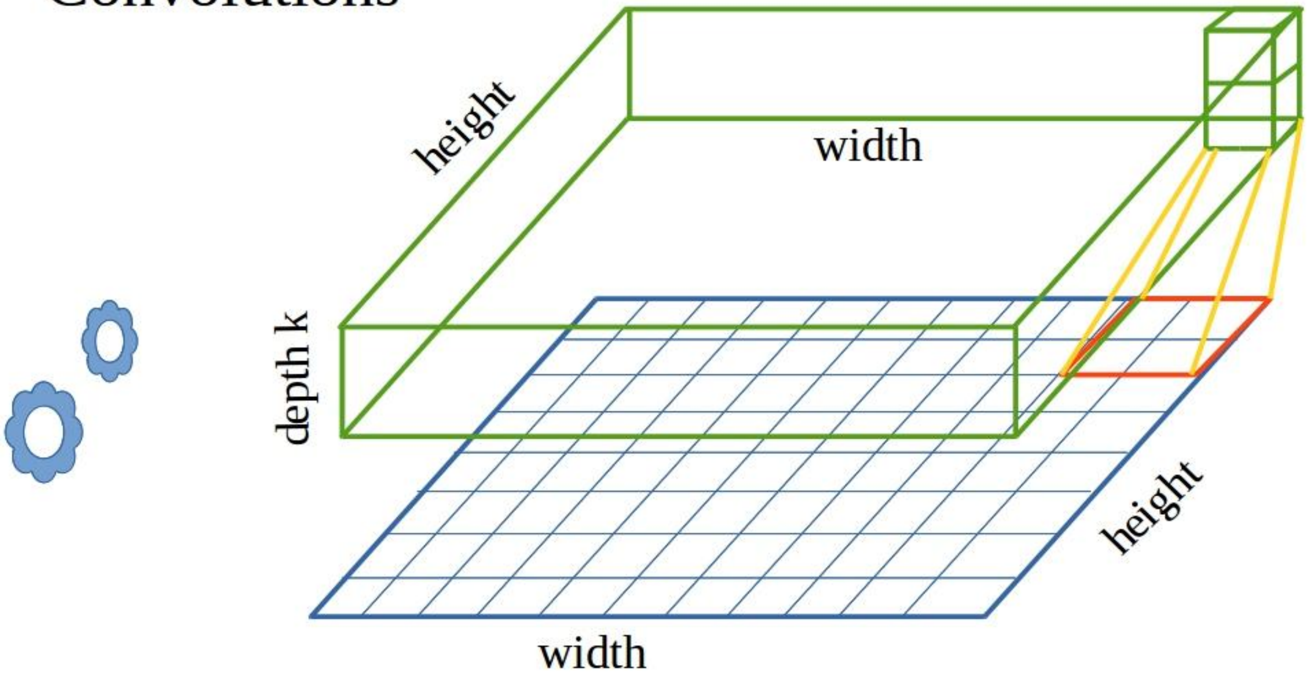
Convolutions

z21 = w11*x2 + w12*x3 + w13*x4 + w14*x5
z22 = w21*x2 + w22*x3 + w23*x4 + w24*x5



Same weights!

Convolutions



Convolutions

For pictures with 10x10 pixel, you train a 3x3-convolution layer with depth 4. Your slider has size 1x1 and your padding is SAME.

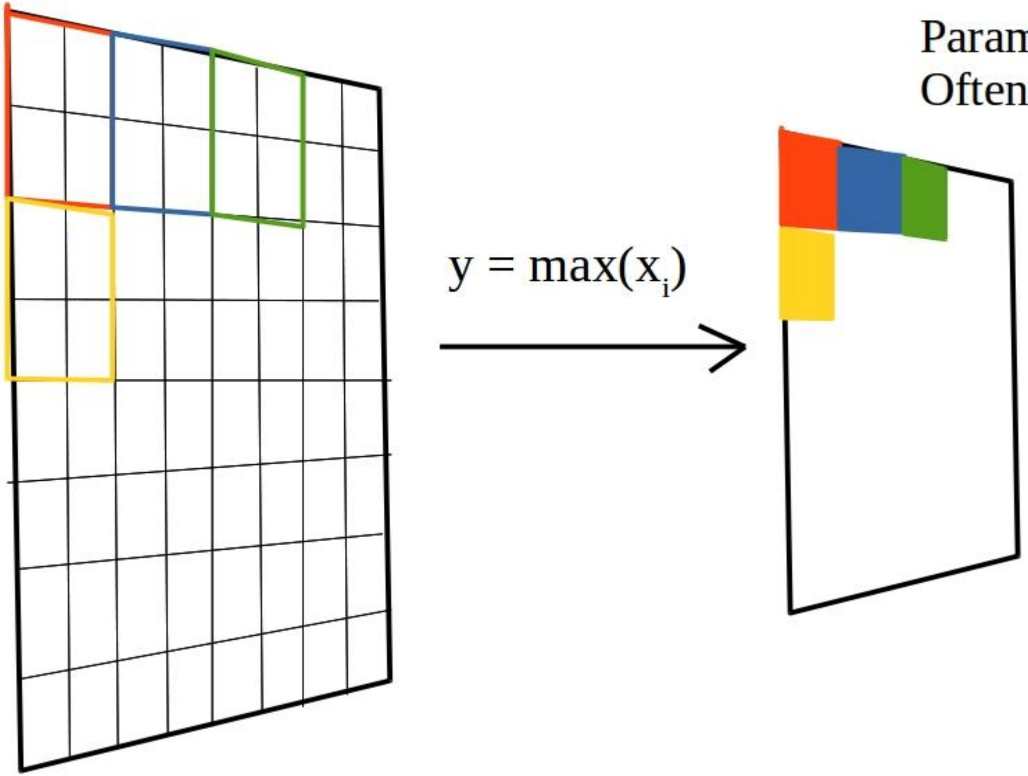
How many weight do you need to train?
How many nodes has the resulting layer?

Pooling

Shrinks large layers

Max pooling

Parameter free!
Often quite accurate

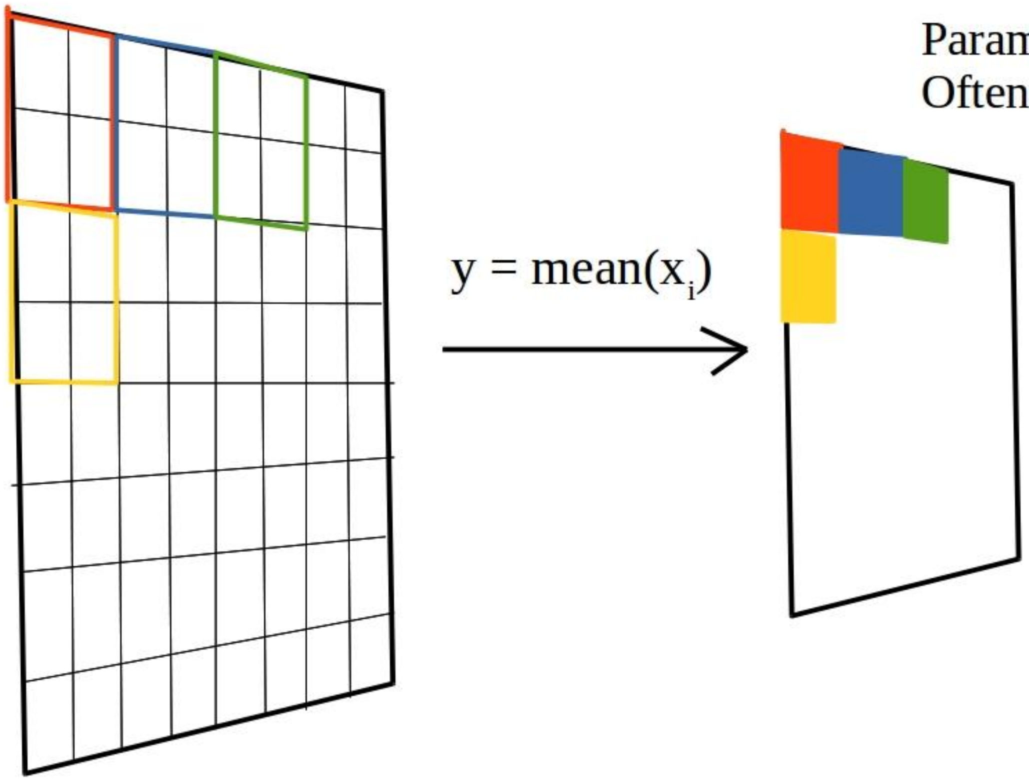


Pooling

Shrinks large layers

Average pooling

Parameter free!
Often quite accurate



A typical neural net

Image

Convolution

Max Pooling

Convolution

Max Pooling

Fully connected

Fully connected

Classifier