# PART III:THE FOUNDATIONS OF DEEP LEARNING

# THE BASIC BUILDING BLOCKS OF CNNs

# Discrete Convolution

**1D: [Spectra]**

$$f(x) * g(x) = \sum_{k=-\infty}^{k=+\infty} f(k).g(k-x)$$

**2D: [Images]**

$$f(x,y) * g(x,y) = \sum_{k=-\infty}^{k=+\infty} \sum_{l=-\infty}^{l=+\infty} f(k,l).g(x-k,y-l)$$

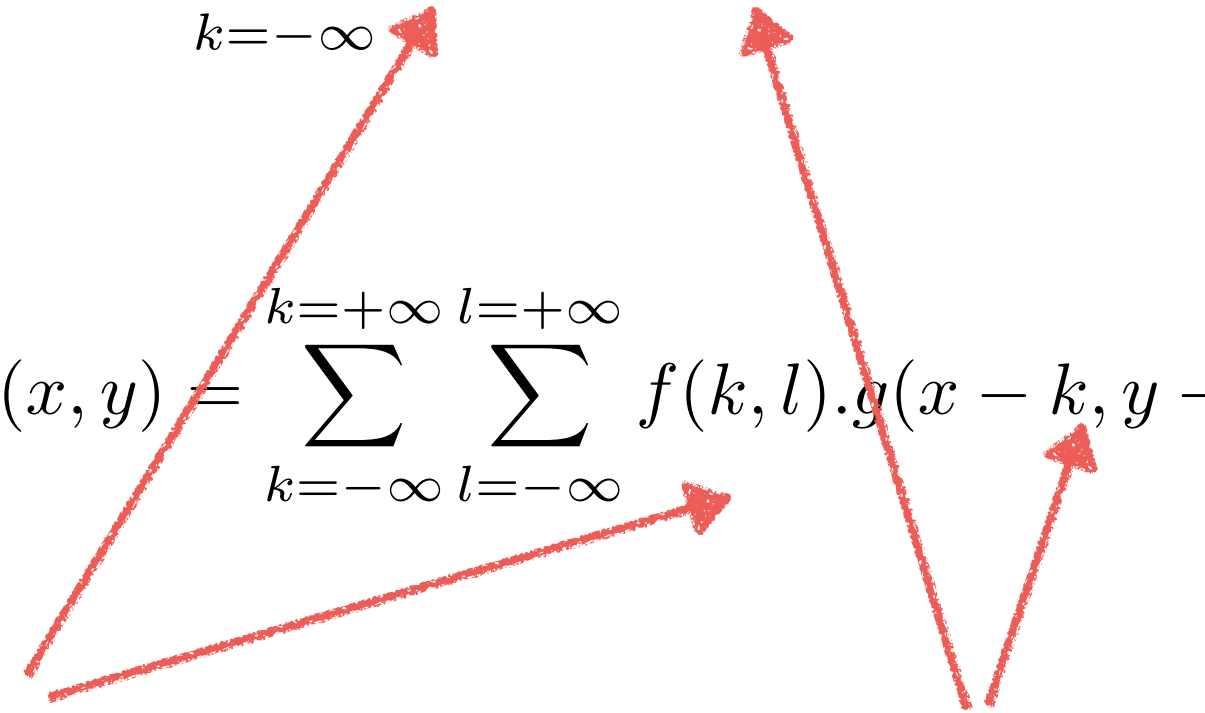
# DISCRETE CONVOLUTION

**1D: [Spectra]**

$$f(x) * g(x) = \sum_{k=-\infty}^{k=+\infty} f(k).g(k-x)$$

**2D: [Images]**

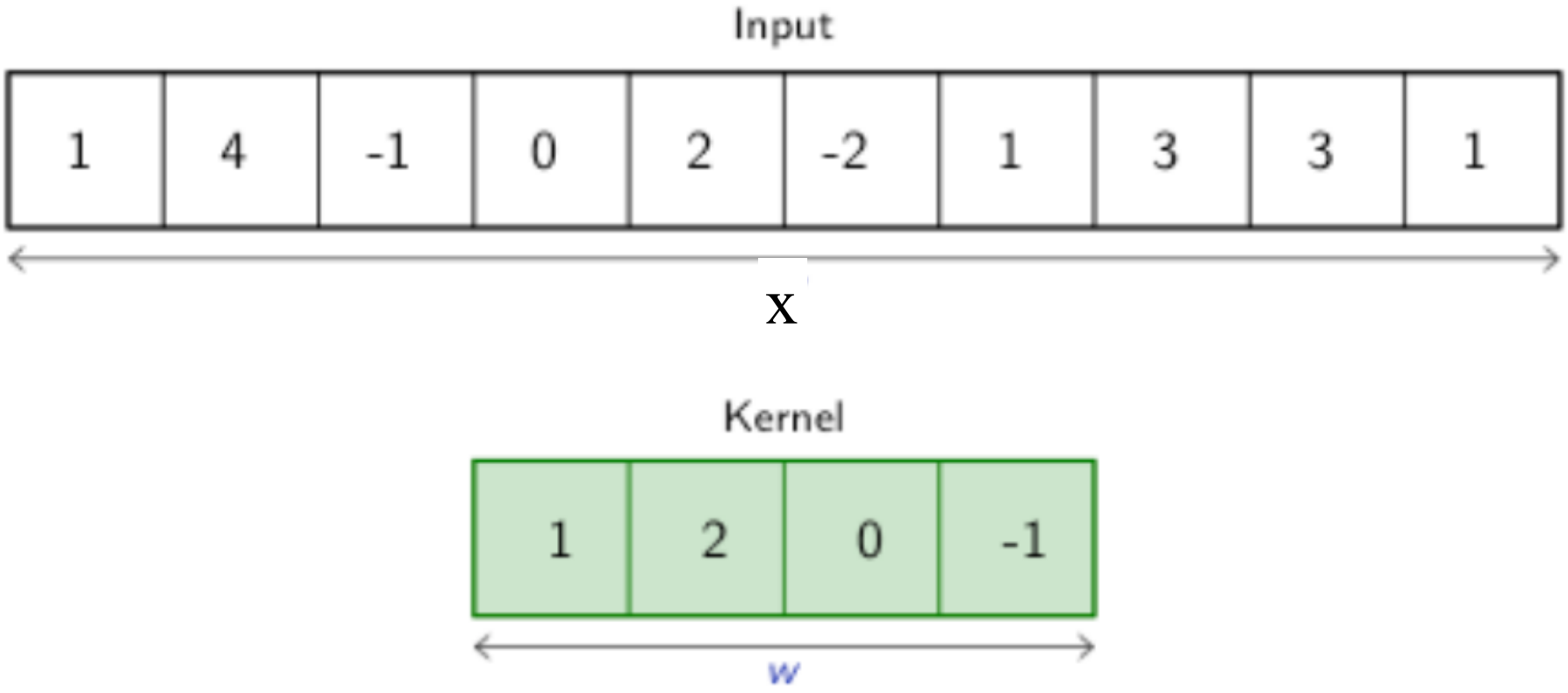
$$f(x,y) * g(x,y) = \sum_{k=-\infty}^{k=+\infty} \sum_{l=-\infty}^{l=+\infty} f(k,l).g(x-k,y-l)$$



CONVOLUTION KERNEL

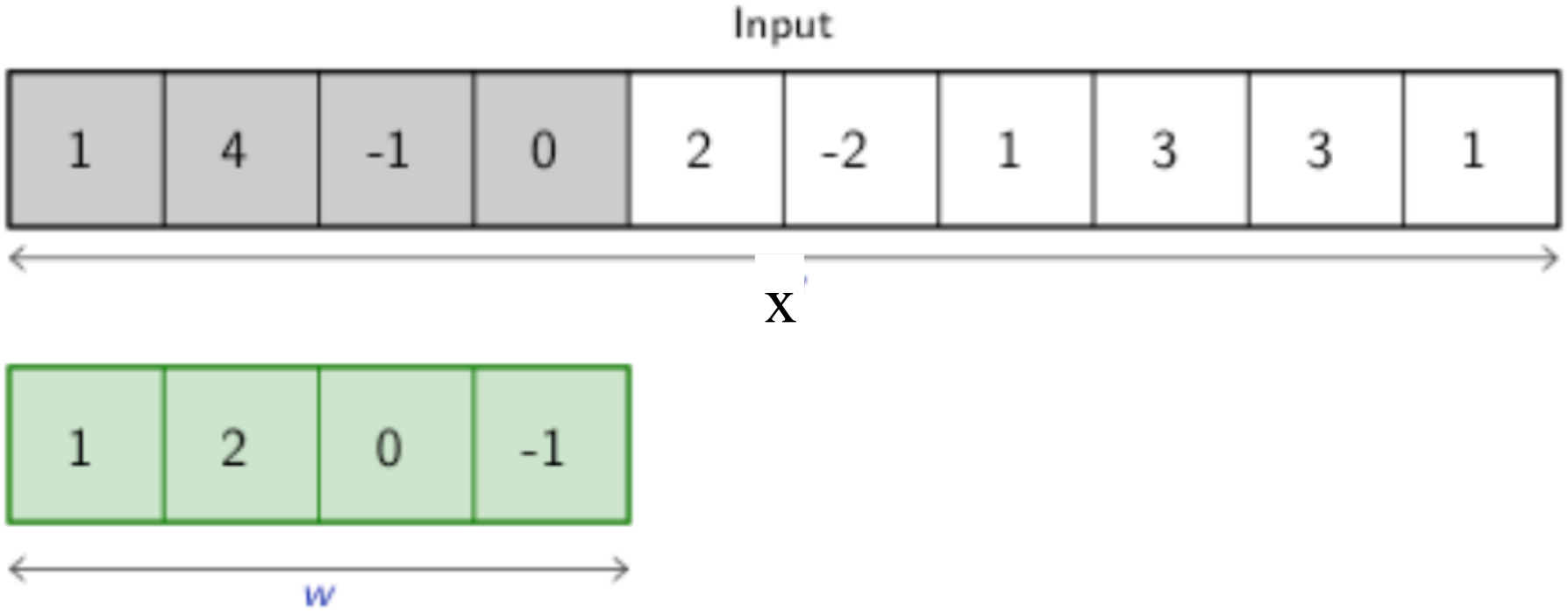
INPUT DATA

# 1-D CONVOLUTION



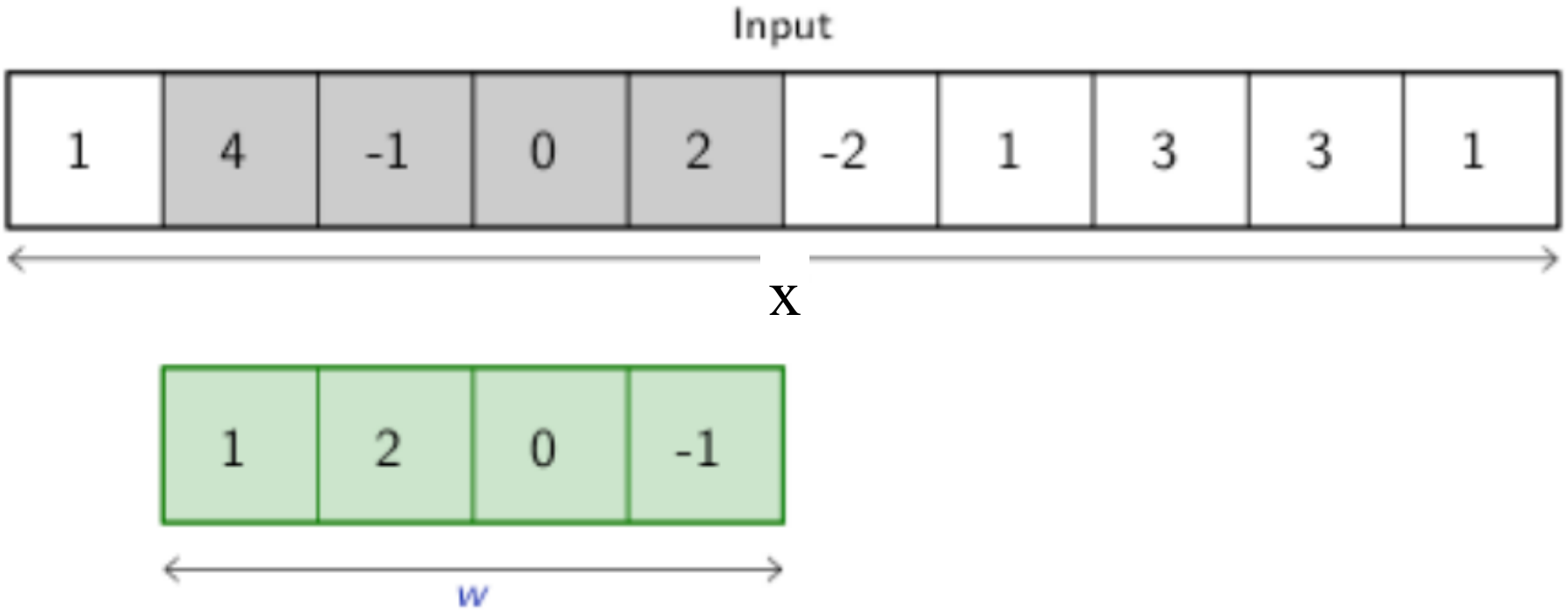

credit

# 1-D CONVOLUTION



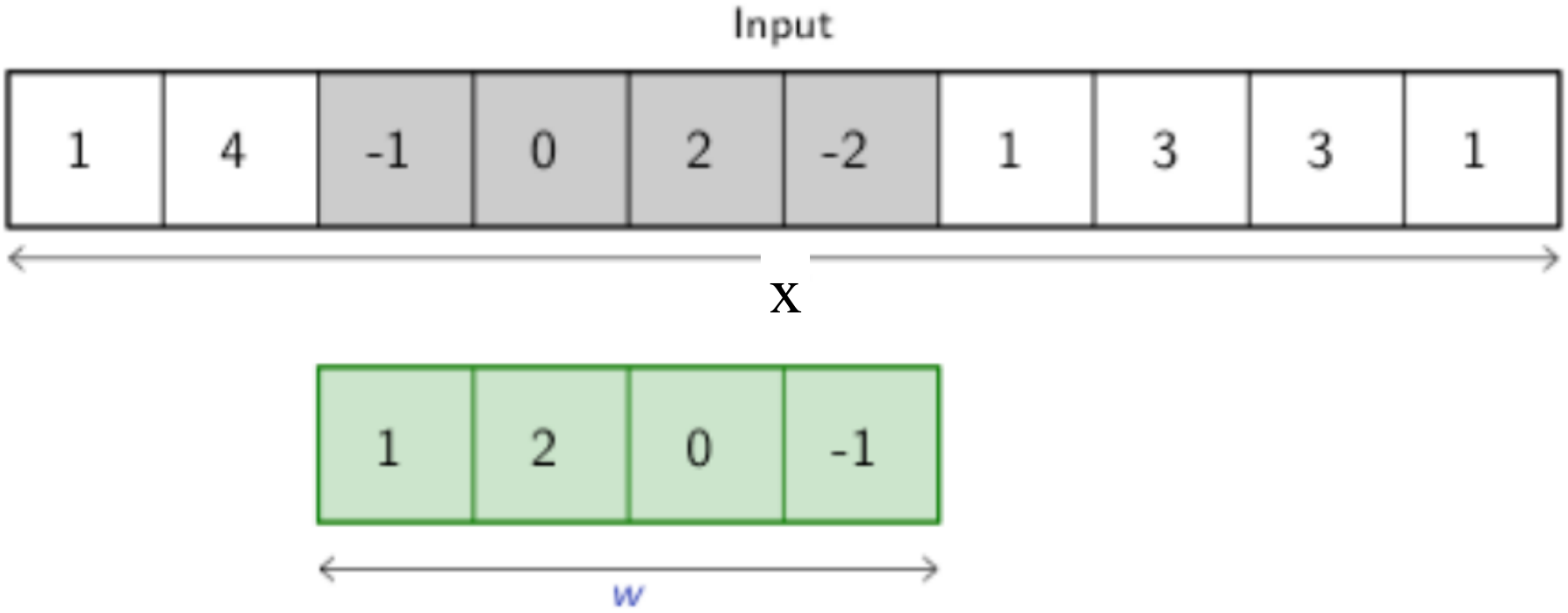

credit

# 1-D CONVOLUTION



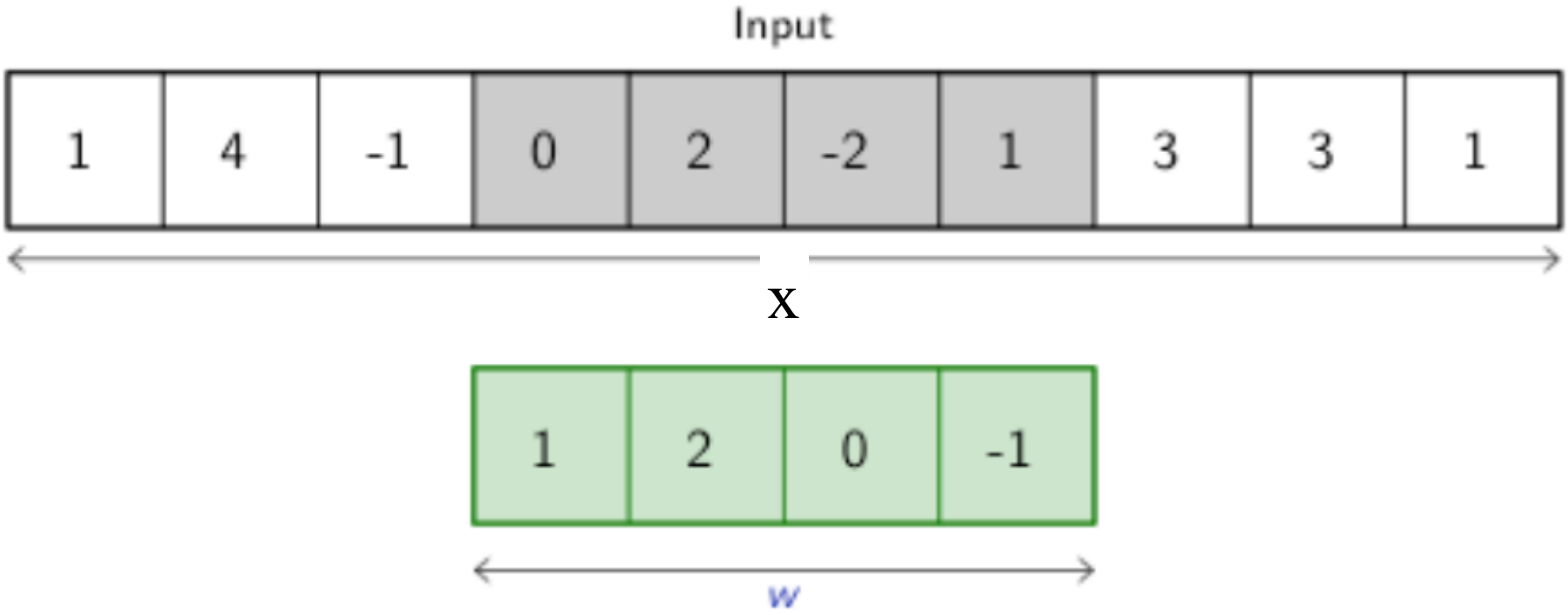

credit

# 1-D CONVOLUTION



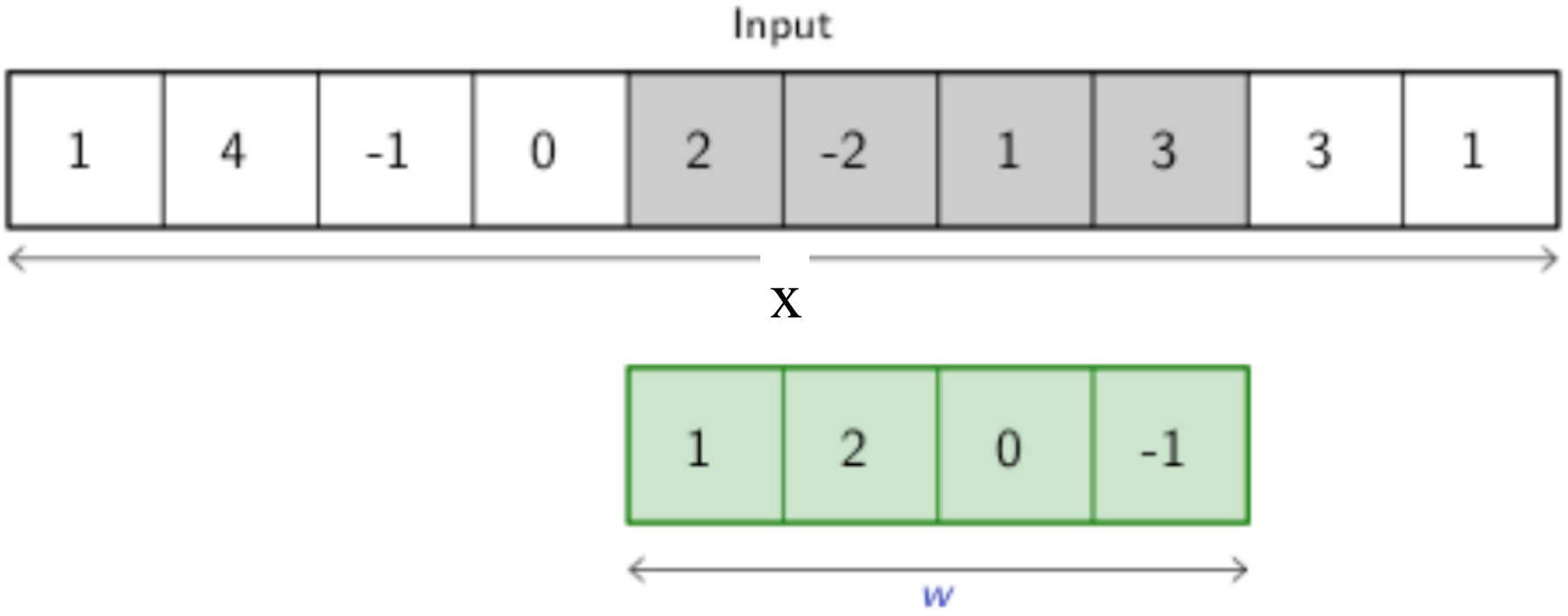

credit

# 1-D CONVOLUTION



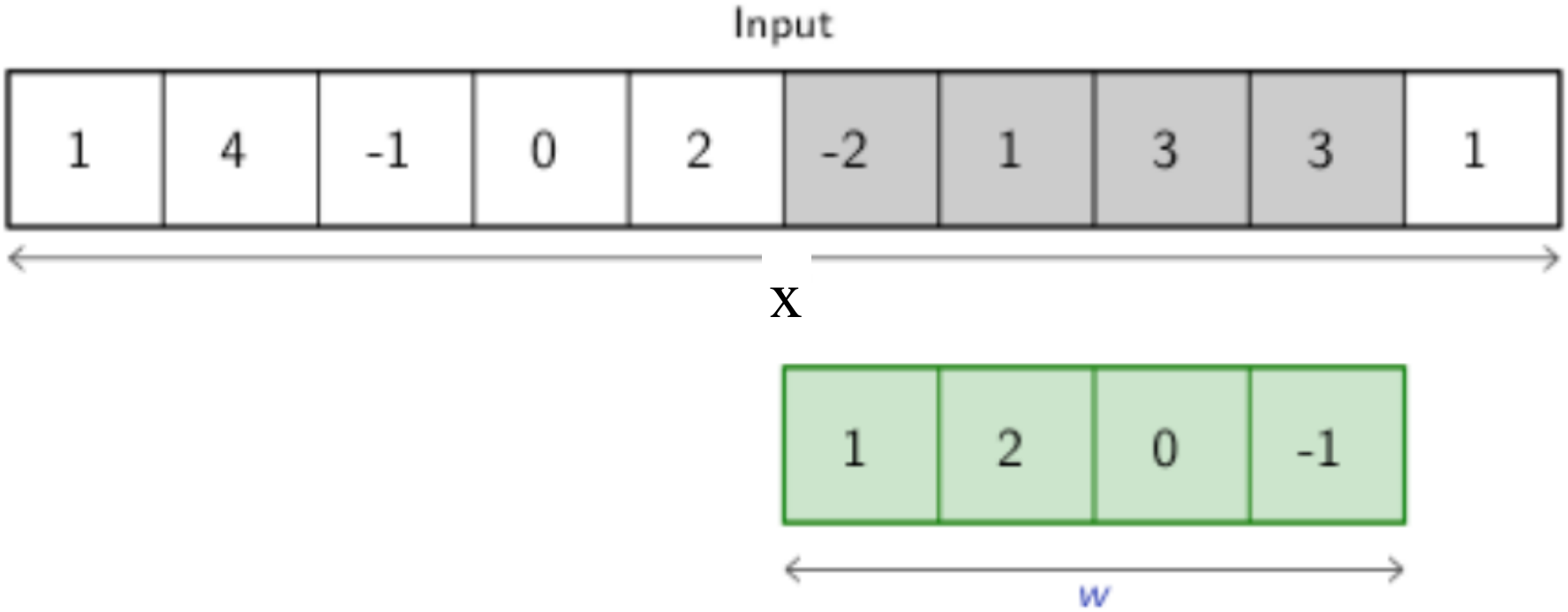

credit

# 1-D CONVOLUTION



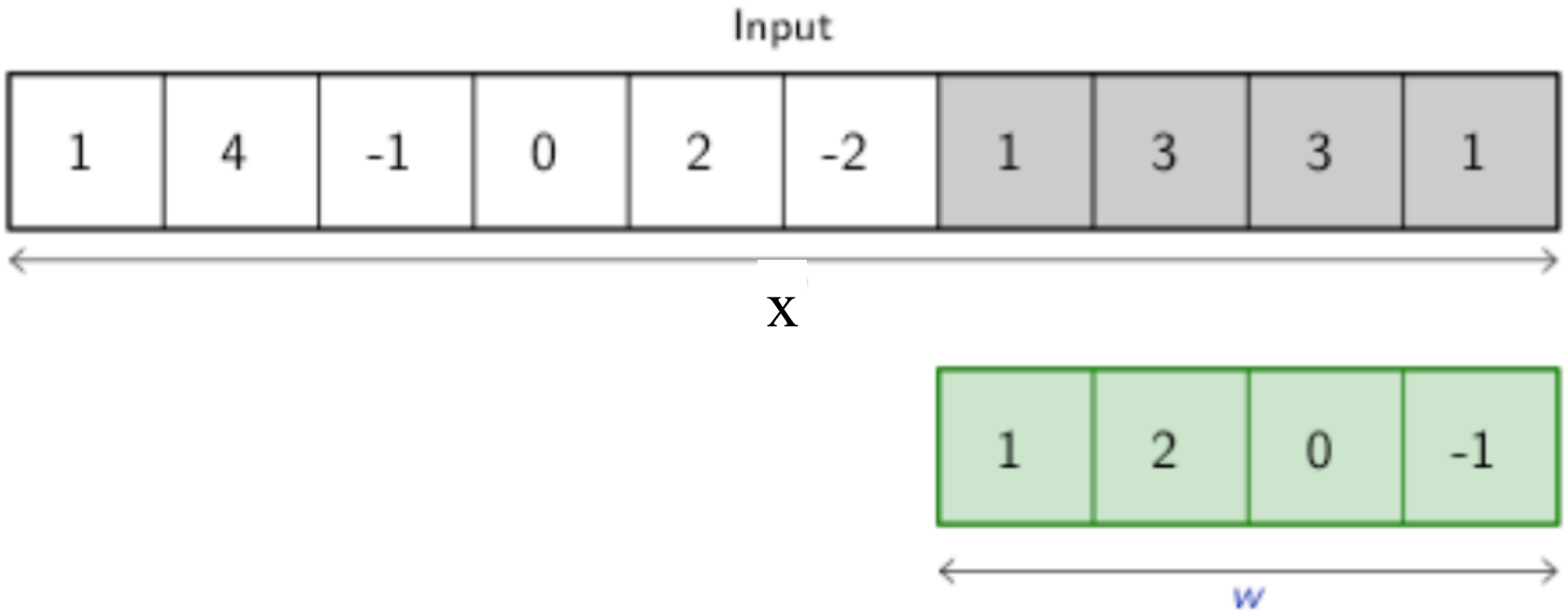

credit

# 1-D CONVOLUTION



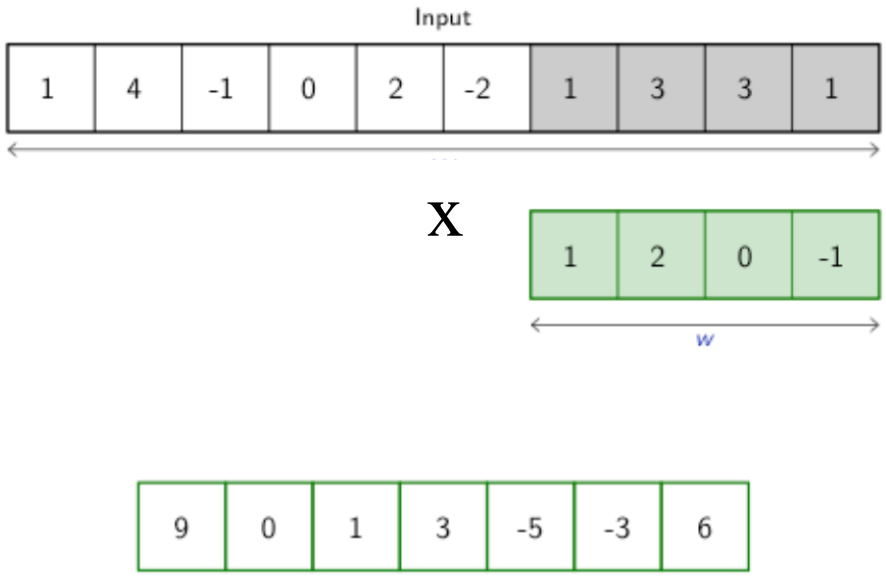

credit

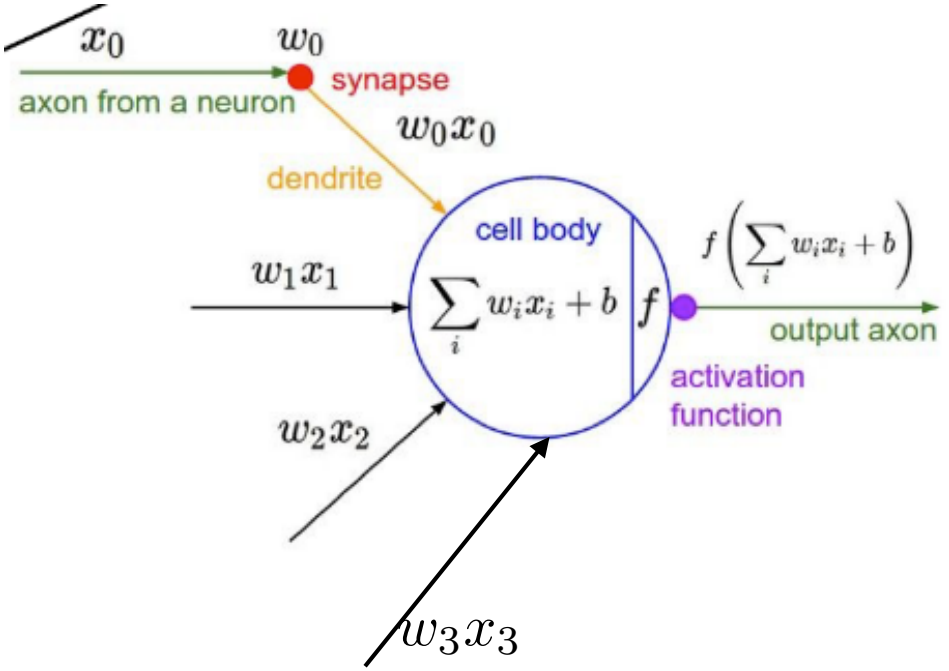
# 1-D CONVOLUTION



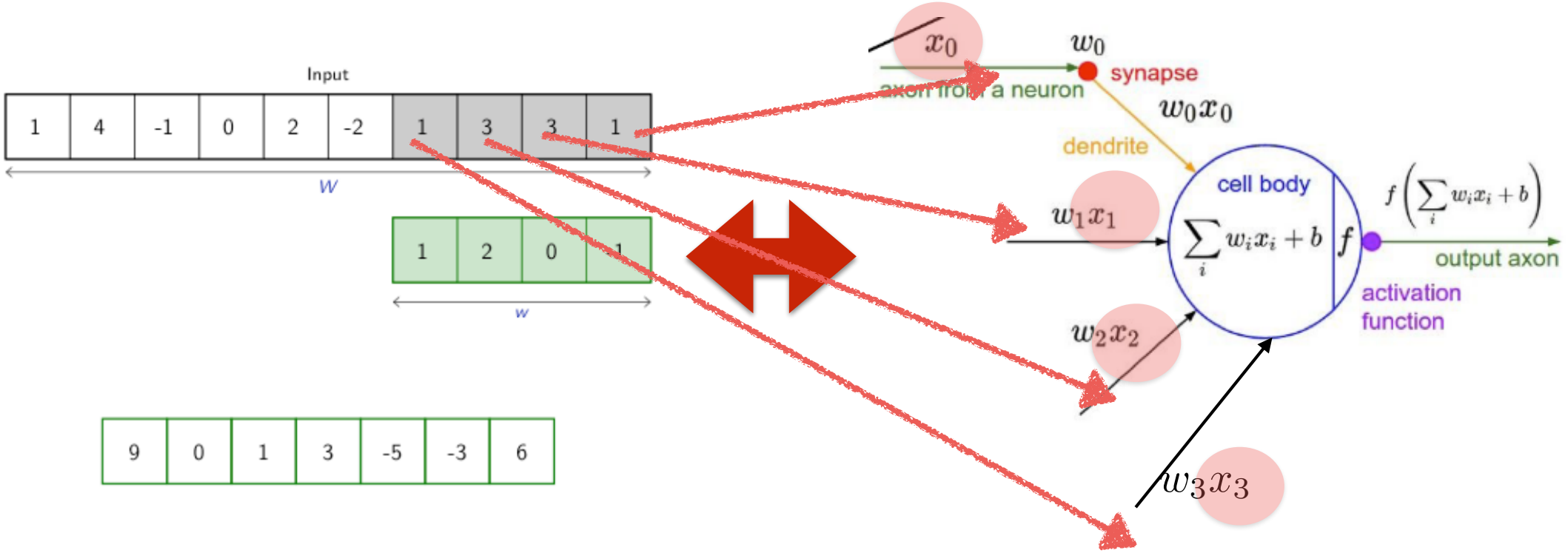

credit

# THE CONVOLUTION BUILDING BLOCK OPERATION IS EQUIVALENT TO A NEURON WITH AS MANY INPUTS AS KERNEL ELEMENTS AND WEIGHTS EQUAL TO THE KERNEL

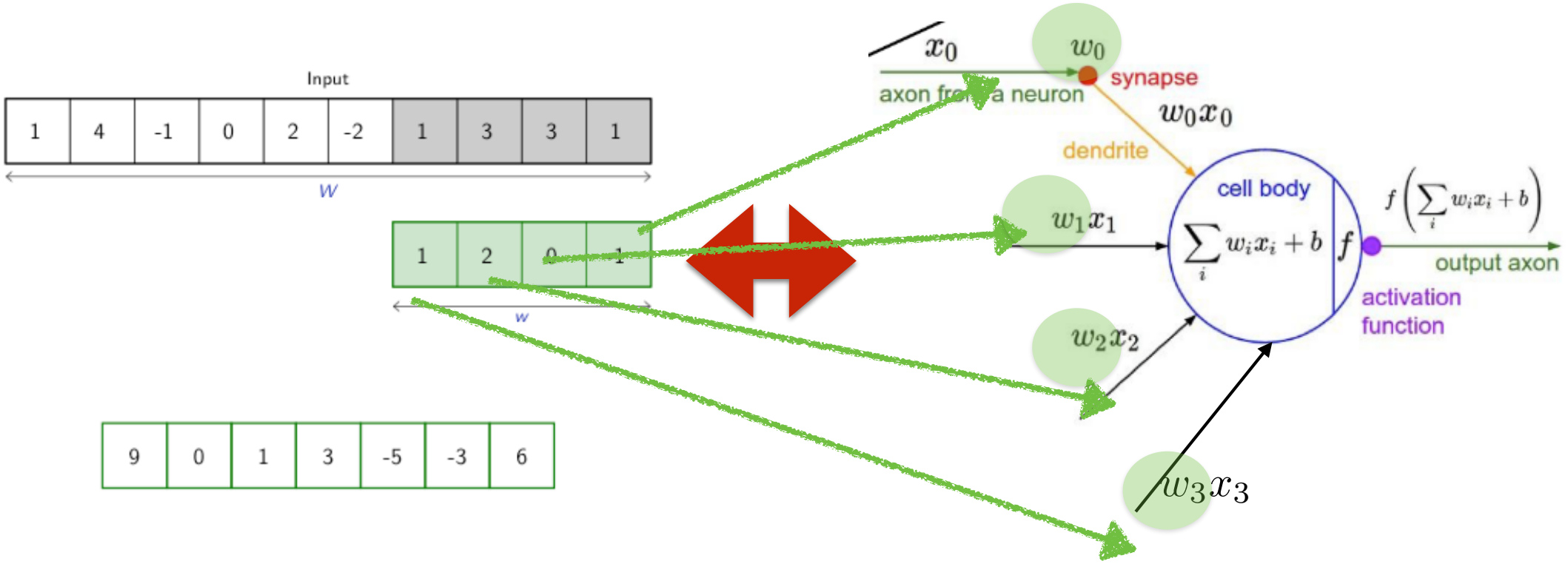


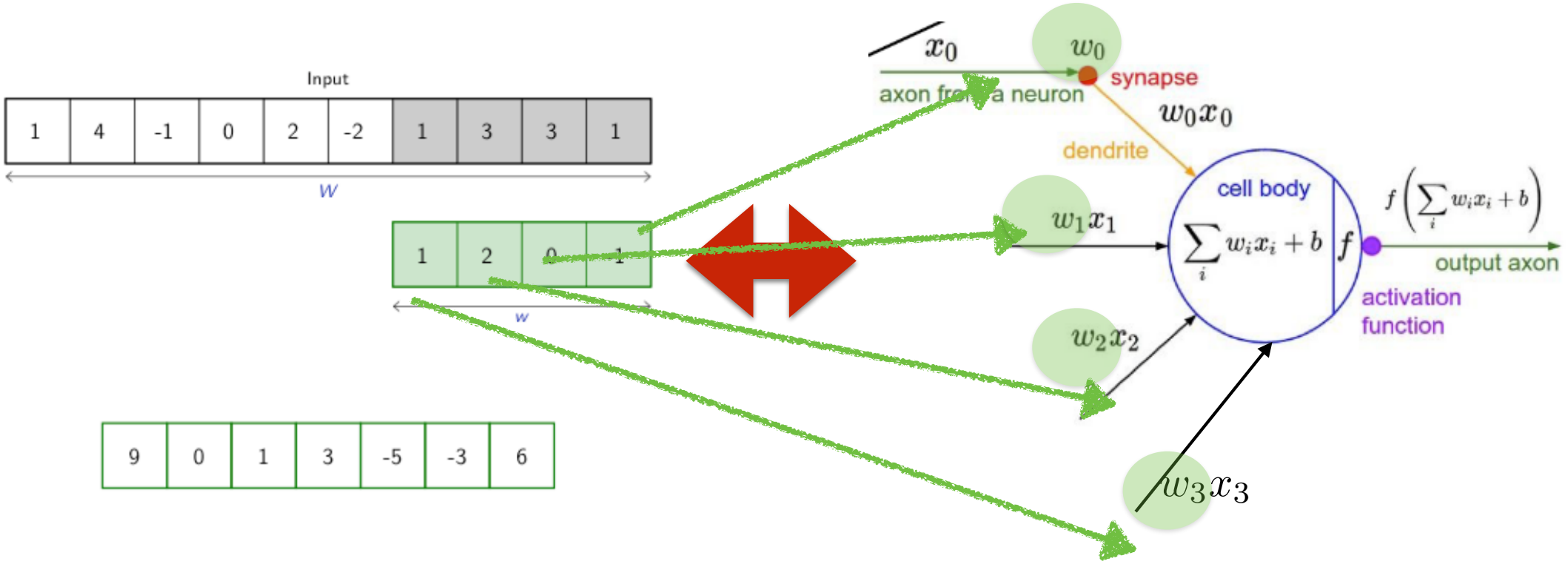
# THE CONVOLUTION BUILDING BLOCK OPERATION IS EQUIVALENT TO A NEURON WITH AS MANY INPUTS AS KERNEL ELEMENTS AND WEIGHTS EQUAL TO THE KERNEL



# THE CONVOLUTION BUILDING BLOCK OPERATION IS EQUIVALENT TO A NEURON WITH AS MANY INPUTS AS KERNEL ELEMENTS AND WEIGHTS EQUAL TO THE KERNEL



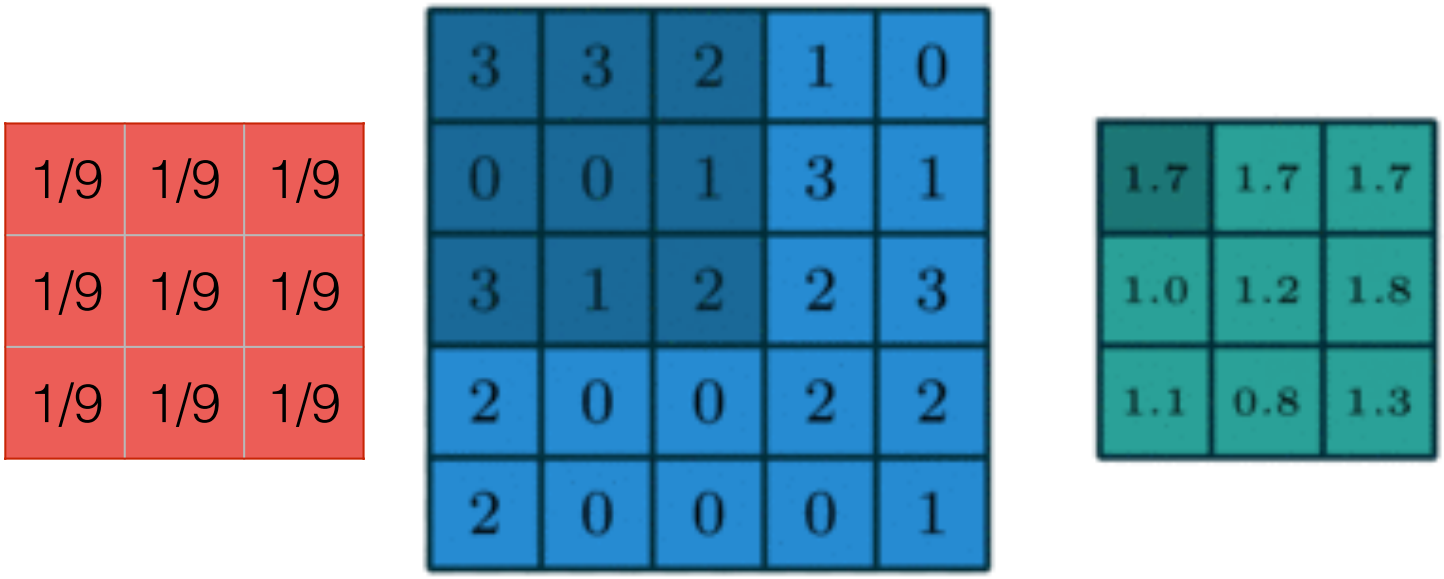
# THE CONVOLUTION BUILDING BLOCK OPERATION IS EQUIVALENT TO A NEURON WITH AS MANY INPUTS AS KERNEL ELEMENTS AND WEIGHTS EQUAL TO THE KERNEL




**WITH THE ADVANTAGE THAT THE SAME WEIGHTS ARE APPLIED TO ALL THE SIGNAL: TRANSLATION INVARIANCE**

# 2-D CONVOLUTION

## SAME IDEA, BUT THE KERNEL IS NOW 2D

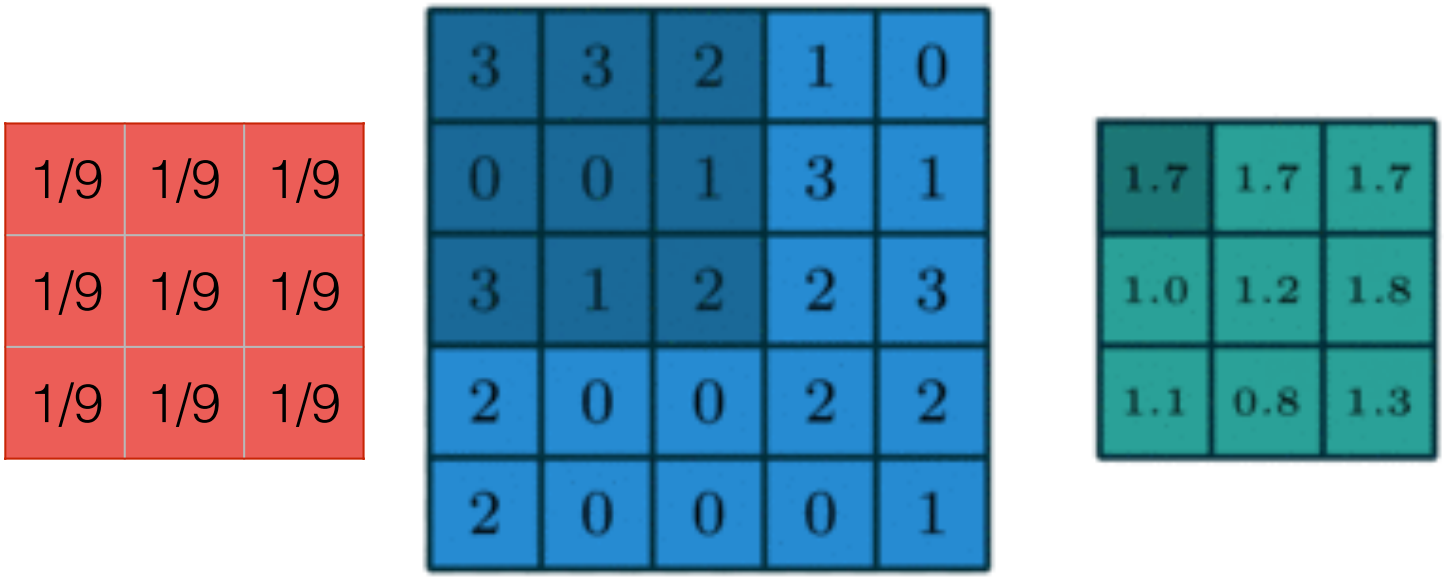



**KERNEL** **INPUT (IMAGE)** **OUTPUT**

**Credit**: animations from https://github.com/vdumoulin/conv_arithmetic

# 2-D CONVOLUTION

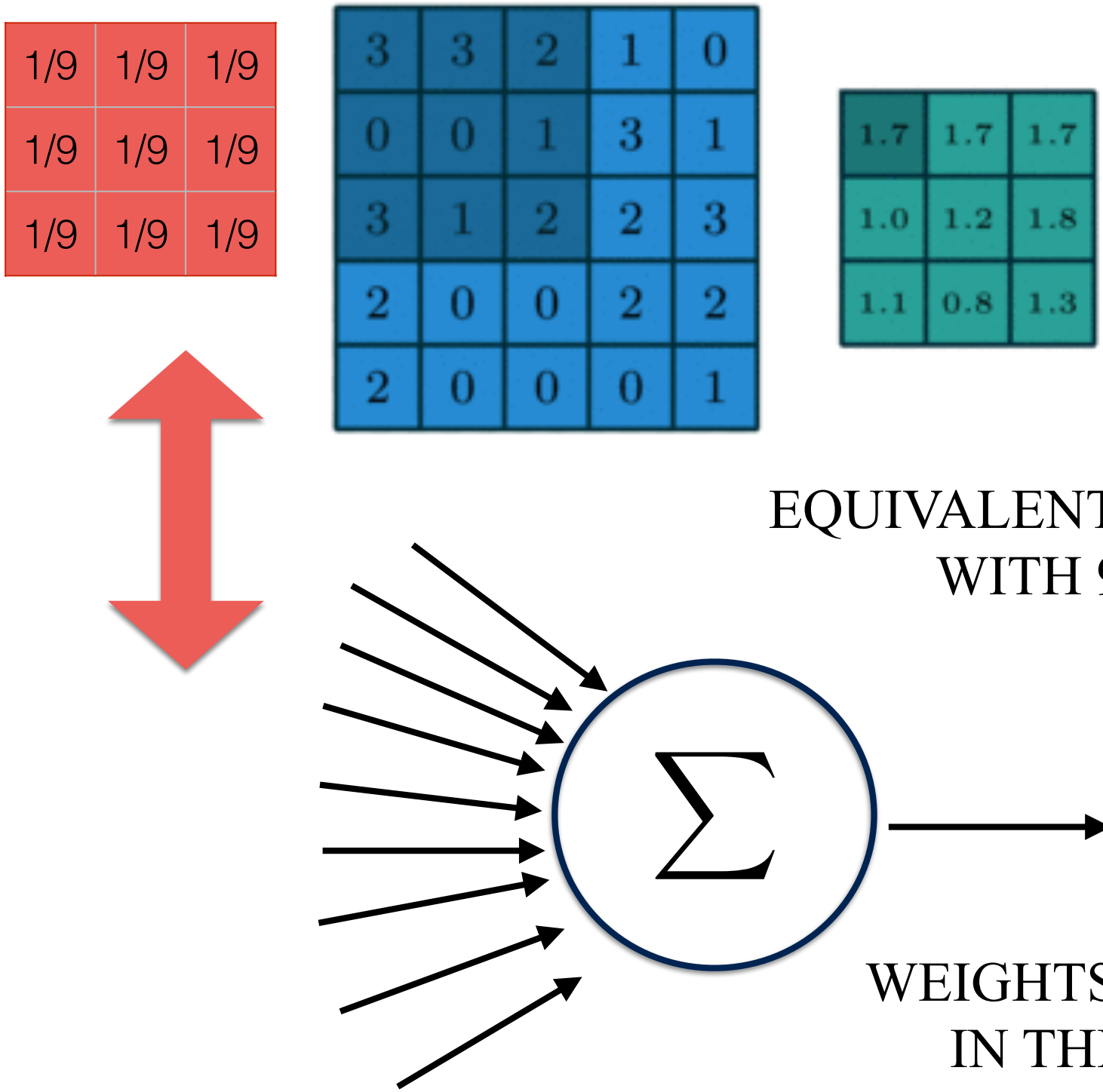
## SAME IDEA, BUT THE KERNEL IS NOW 2D

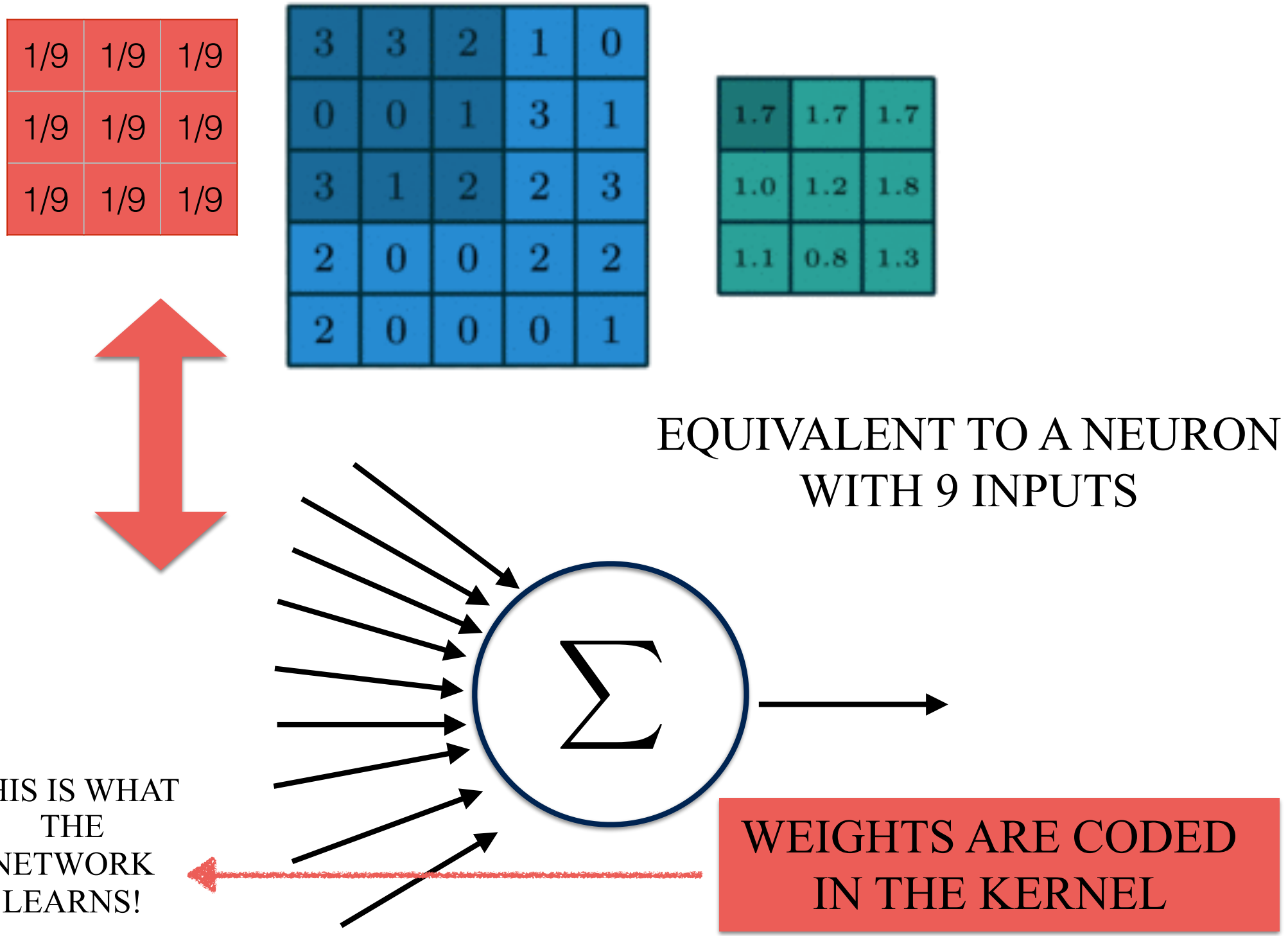


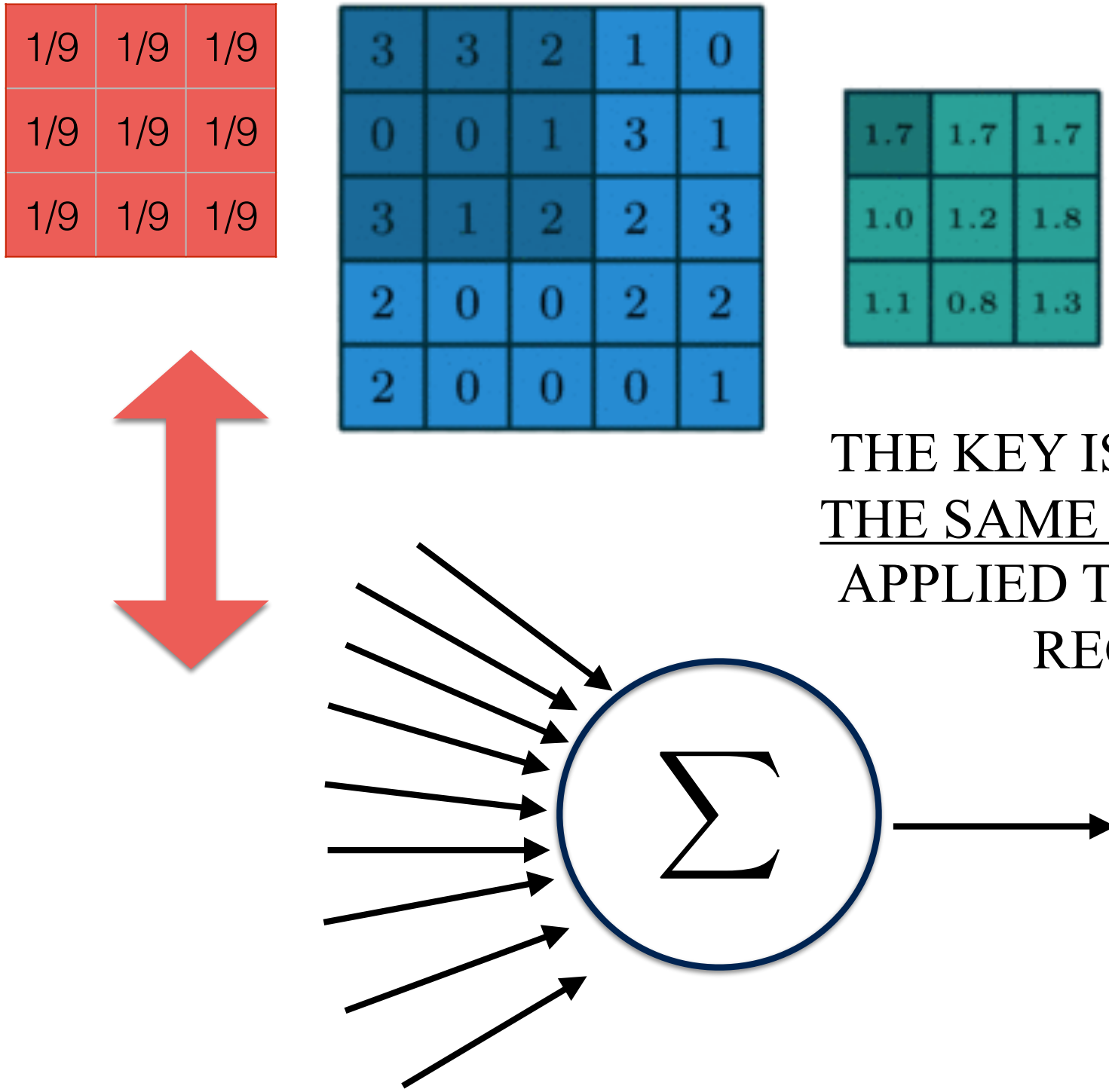

IN THE EXAMPLE: EACH 3x3 REGION GENERATES AN OUTPUT

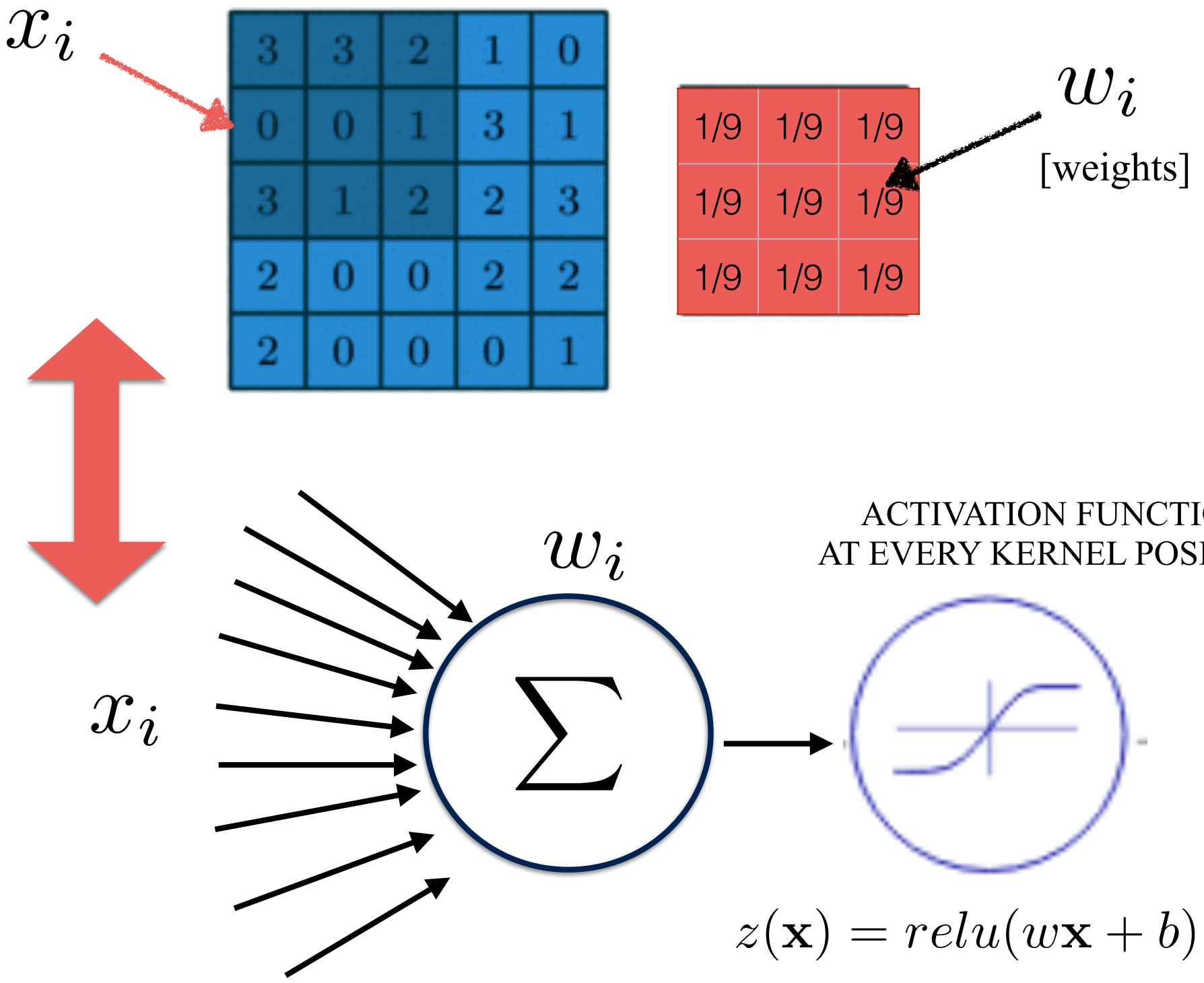
$$Size_{output} = Size_{input} - Size_{kernel} + 1$$

**Credit**: animations from https://github.com/vdumoulin/conv_arithmetic


1/9 1/9 1/9
1/9 1/9 1/9
1/9 1/9 1/9
3 3 2 1 0
0 0 1 3 1
3 1 2 2 3
2 0 0 2 2
2 0 0 0 1
1.7 1.7 1.7
1.0 1.2 1.8
1.1 0.8 1.3
EQUIVALENT TO A NEURON
WITH 9 INPUTS
Σ
WEIGHTS ARE CODED
IN THE KERNEL

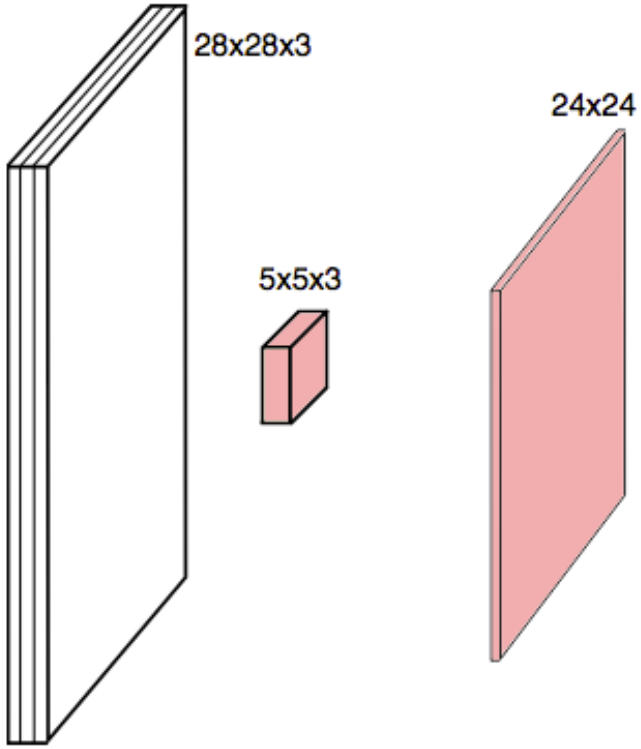

1/9 1/9 1/9
1/9 1/9 1/9
1/9 1/9 1/9
3 3 2 1 0
0 0 1 3 1
3 1 2 2 3
2 0 0 2 2
2 0 0 0 1
1.7 1.7 1.7
1.0 1.2 1.8
1.1 0.8 1.3
EQUIVALENT TO A NEURON
WITH 9 INPUTS
Σ
THIS IS WHAT
THE
NETWORK
LEARNS!
WEIGHTS ARE CODED
IN THE KERNEL

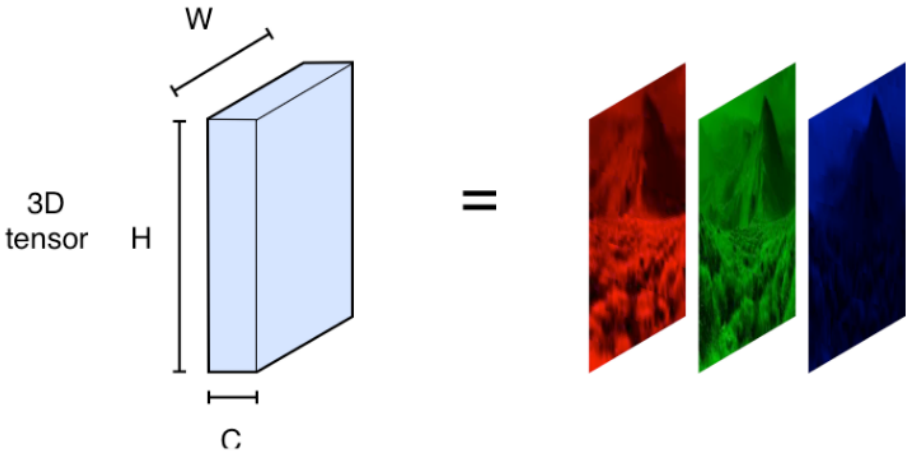

1/9 1/9 1/9
1/9 1/9 1/9
1/9 1/9 1/9
3 3 2 1 0
0 0 1 3 1
3 1 2 2 3
2 0 0 2 2
2 0 0 0 1
1.7 1.7 1.7
1.0 1.2 1.8
1.1 0.8 1.3
THE KEY IS AGAIN THAT
THE SAME WEIGHTS ARE
APPLIED TO ALL IMAGE
REGIONS
Σ


$x_i$
3 3 2 1 0
0 0 1 3 1
3 1 2 2 3
2 0 0 2 2
2 0 0 0 1
1/9 1/9 1/9
1/9 1/9 1/9
1/9 1/9 1/9
$w_i$
[weights]
$w_i$
$x_i$
$\sum$
ACTIVATION FUNCTION
AT EVERY KERNEL POSITION
$z(\mathbf{x}) = relu(w\mathbf{x} + b)$

# CONVOLUTIONS CAN ALSO BE COMPUTED ACROSS CHANNELS (OR COLORS)

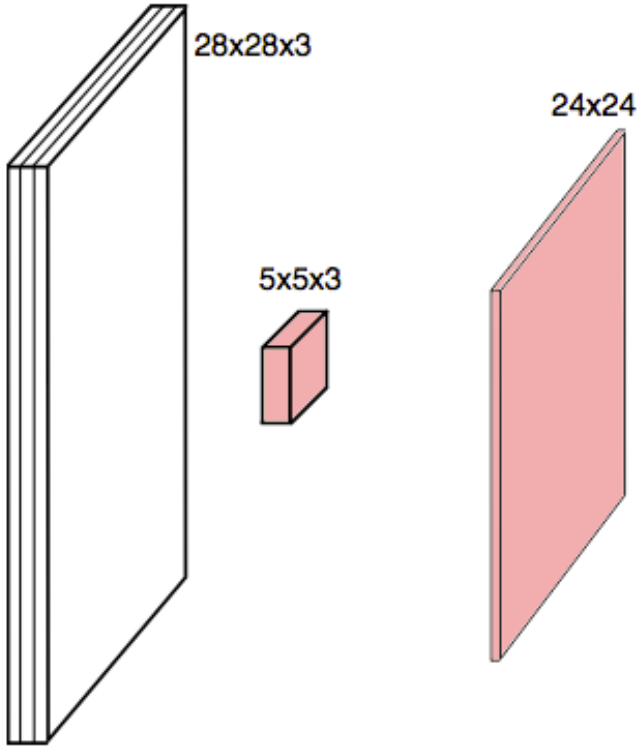
A COLOR IMAGE IS A TENSOR
OF SIZE height x width x channels

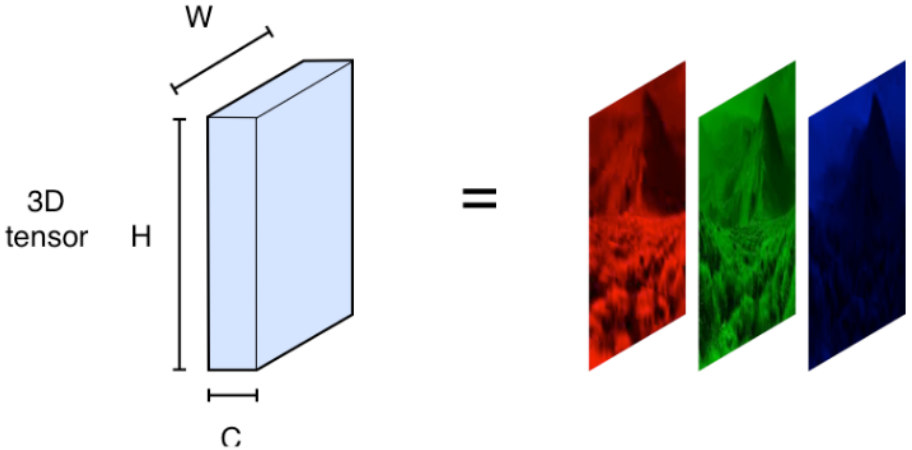


# CONVOLUTIONS CAN ALSO BE COMPUTED ACROSS CHANNELS (OR COLORS)

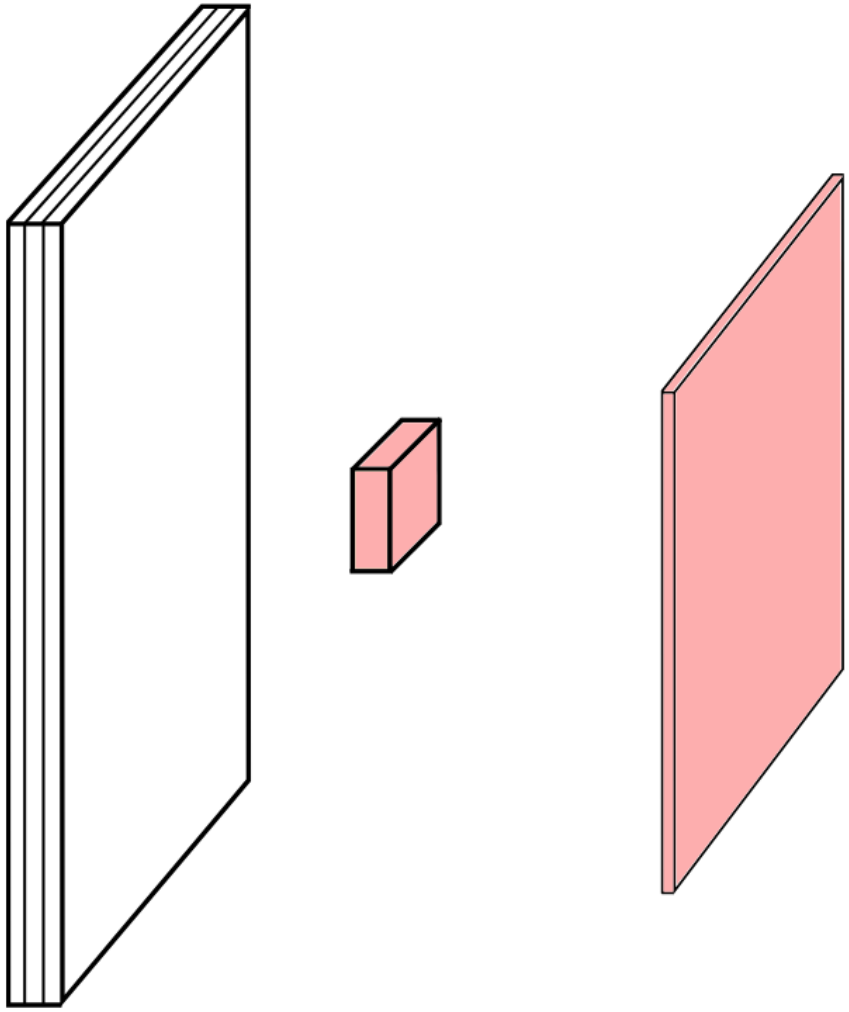
A COLOR IMAGE IS A TENSOR
OF SIZE height x width x channels



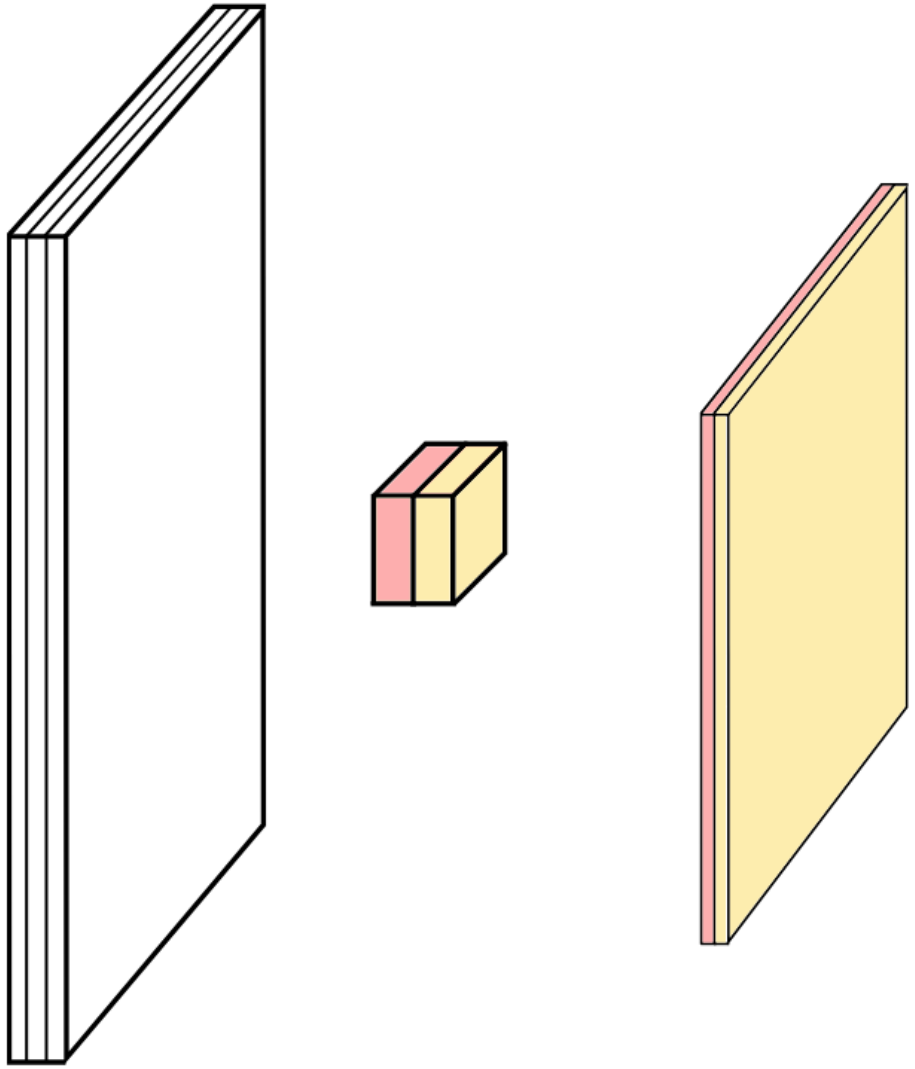

THEN THE KERNEL HAS ALSO 3 CHANNELS

# MULTIPLE CONVOLUTIONS WITH DIFFERENT KERNELS CAN BE PERFORMED



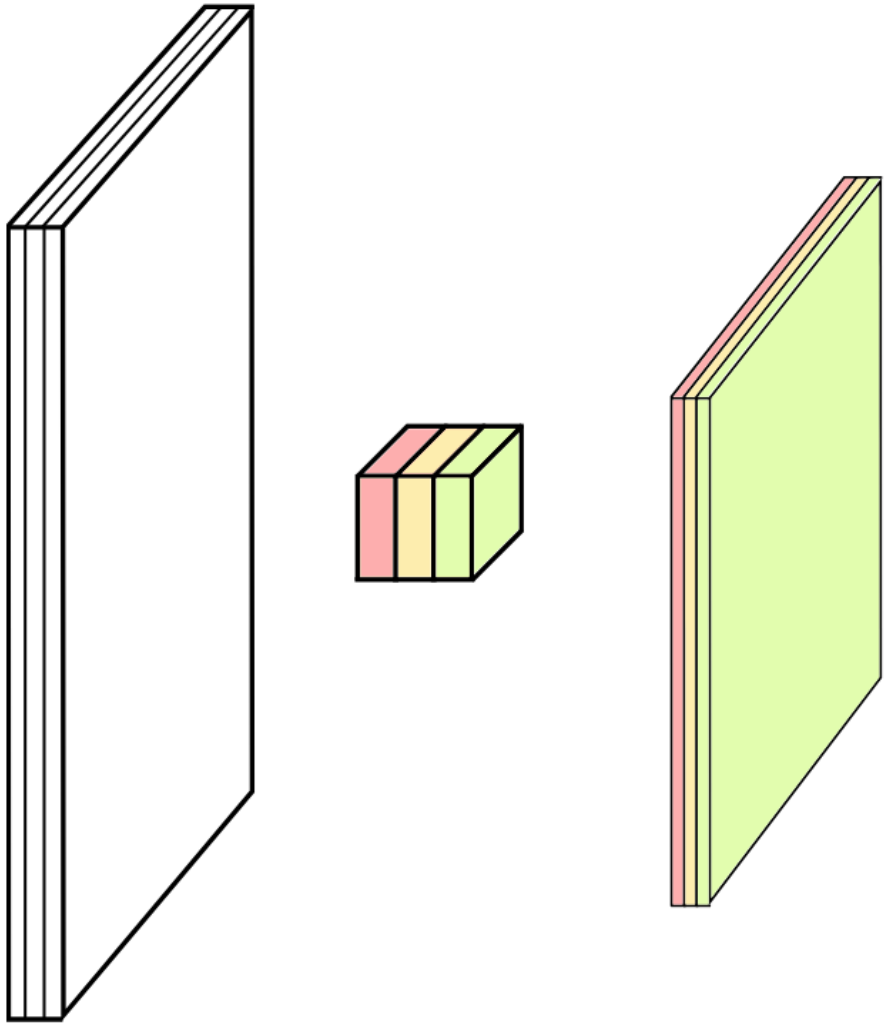
credit

# MULTIPLE CONVOLUTIONS WITH DIFFERENT KERNELS CAN BE PERFORMED



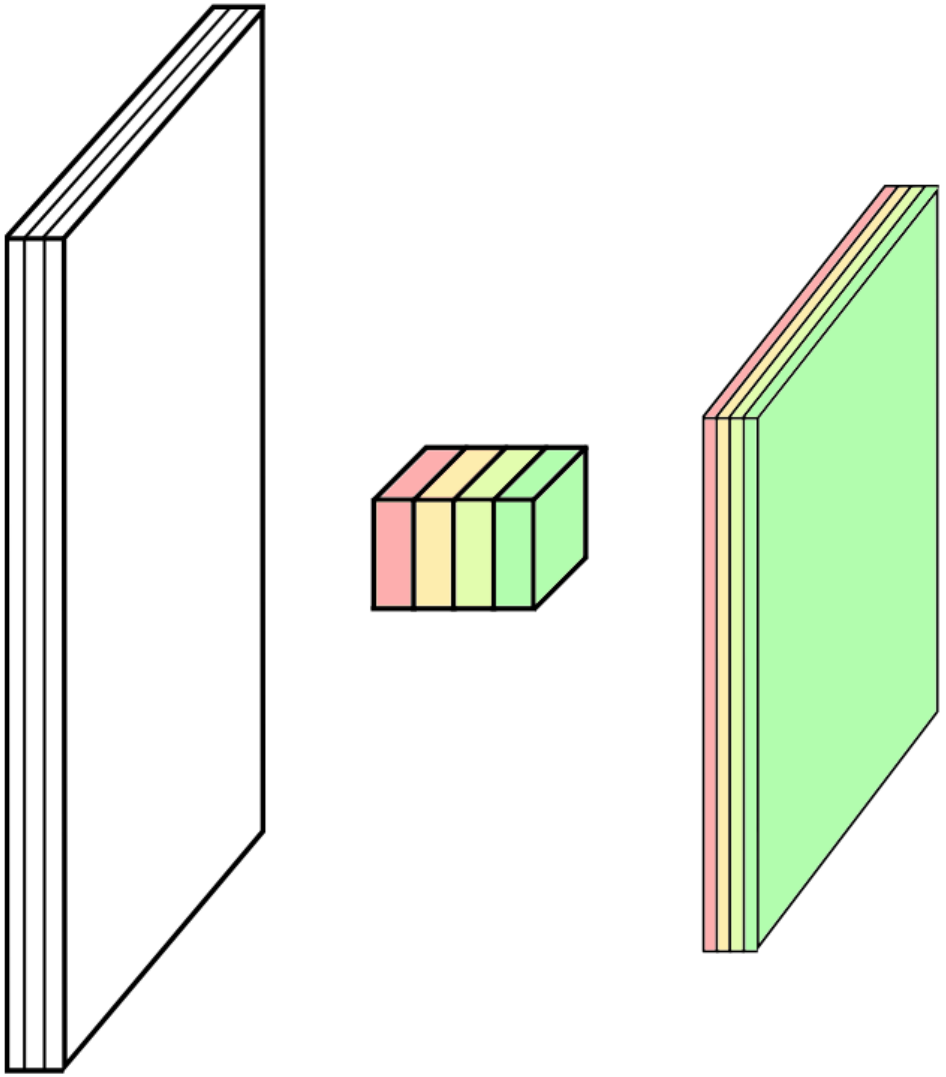
credit

# MULTIPLE CONVOLUTIONS WITH DIFFERENT KERNELS CAN BE PERFORMED



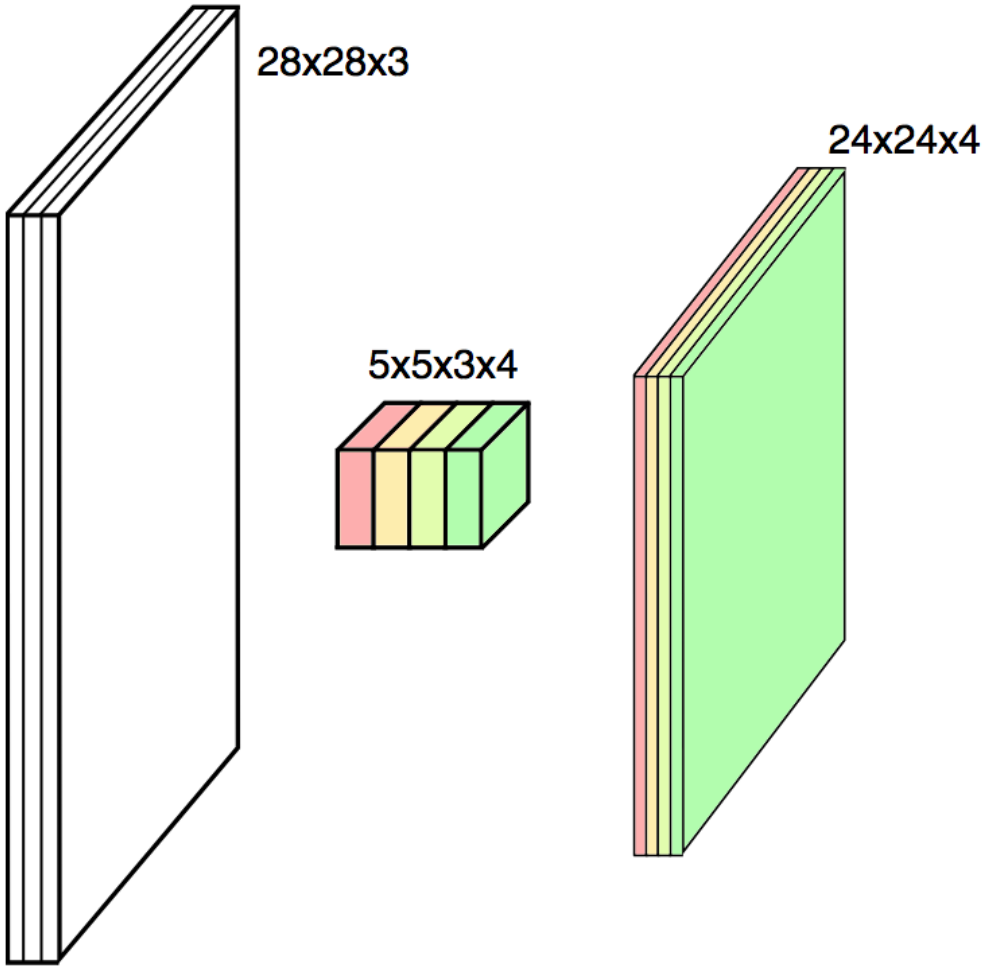
credit

# MULTIPLE CONVOLUTIONS WITH DIFFERENT KERNELS CAN BE PERFORMED



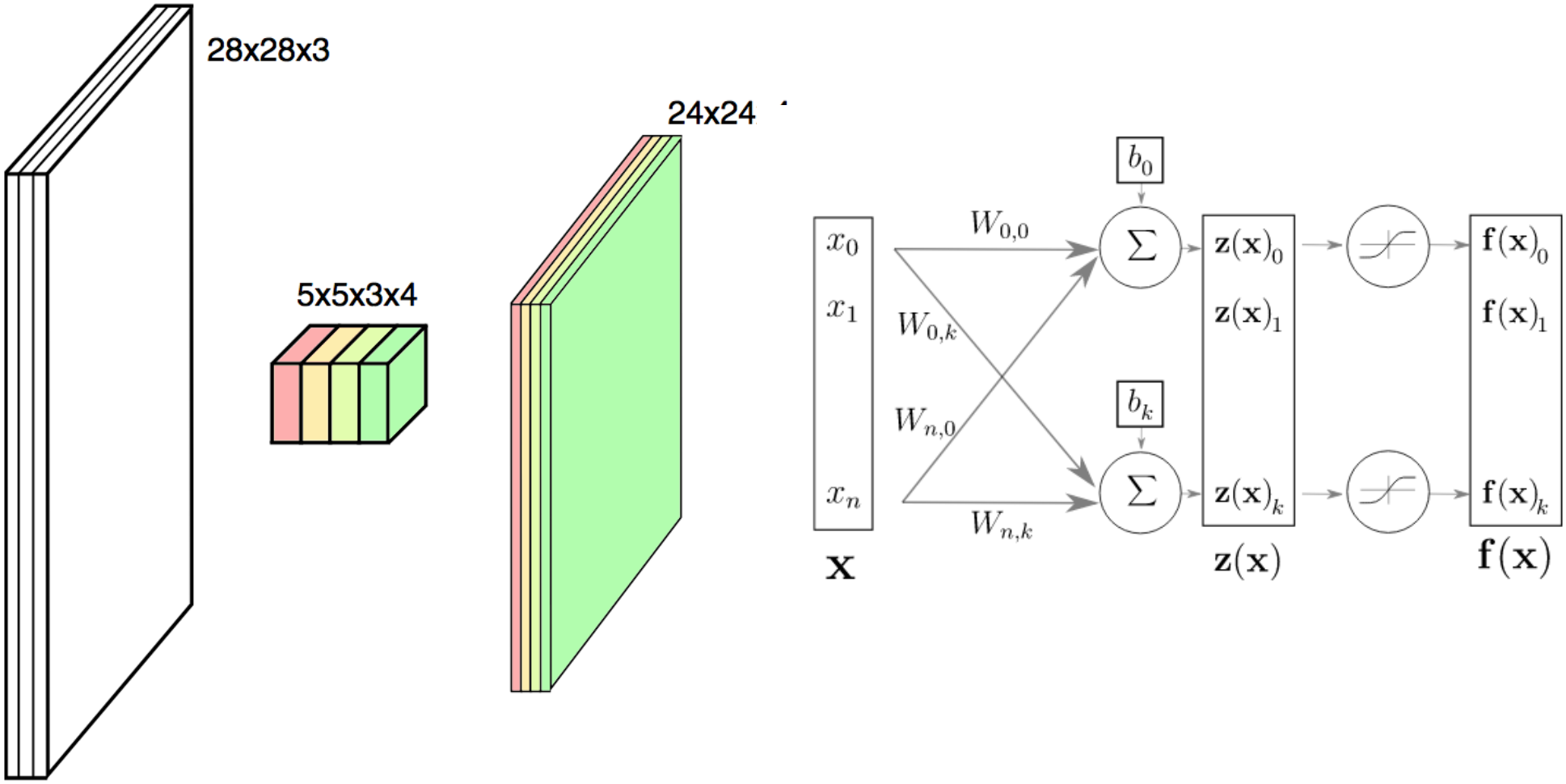
credit

# MULTIPLE CONVOLUTIONS WITH DIFFERENT KERNELS CAN BE PERFORMED




credit

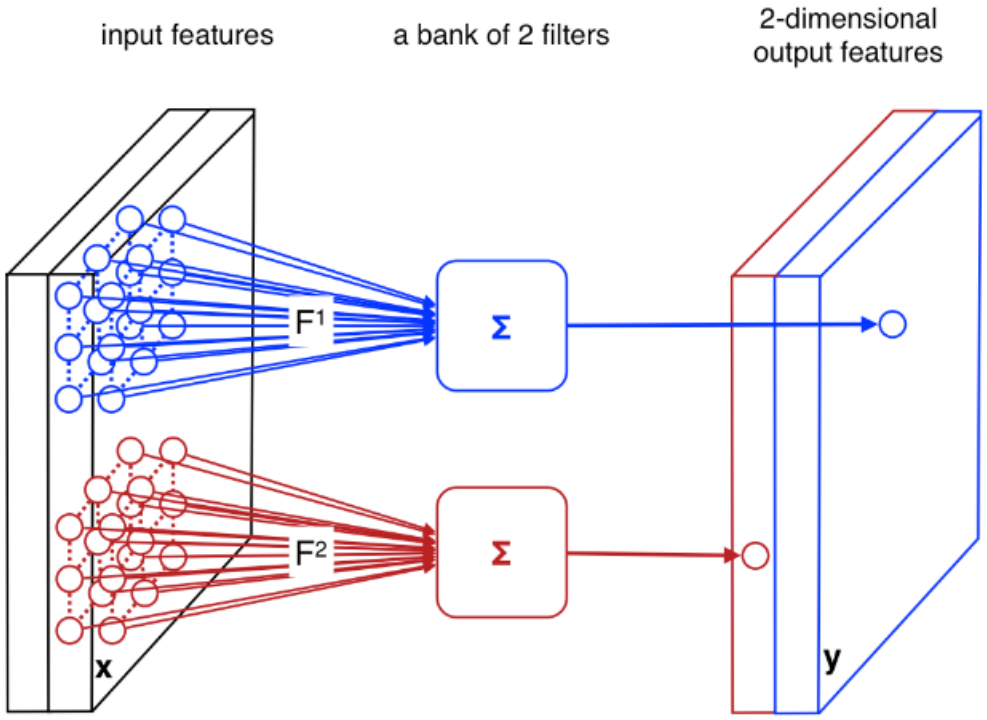
# MULTIPLE CONVOLUTIONS WITH DIFFERENT KERNELS CAN BE PERFORMED




credit

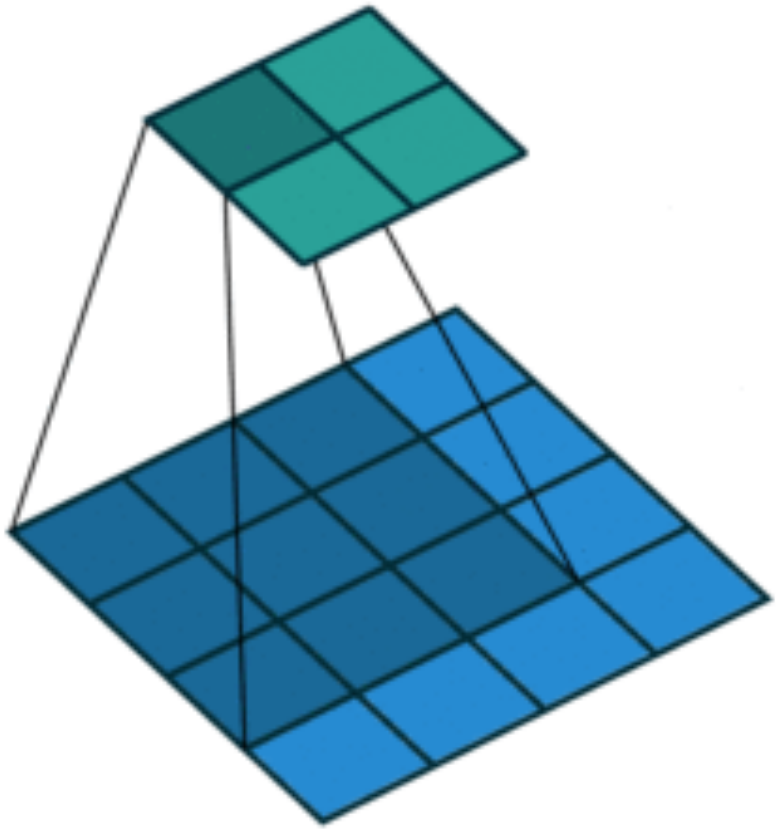
SINCE CONVOLUTIONS OUTPUT ONE SCALAR< THEY CAN BE SEEN AS AN INDIVIDUAL NEURON WITH A RECEPTIVE FIELD LIMITED TO THE KERNEL DIMENSIONS

THE SAME NEURON IS FIRED WITH DIFFERENT AREAS FROM THE INPUT

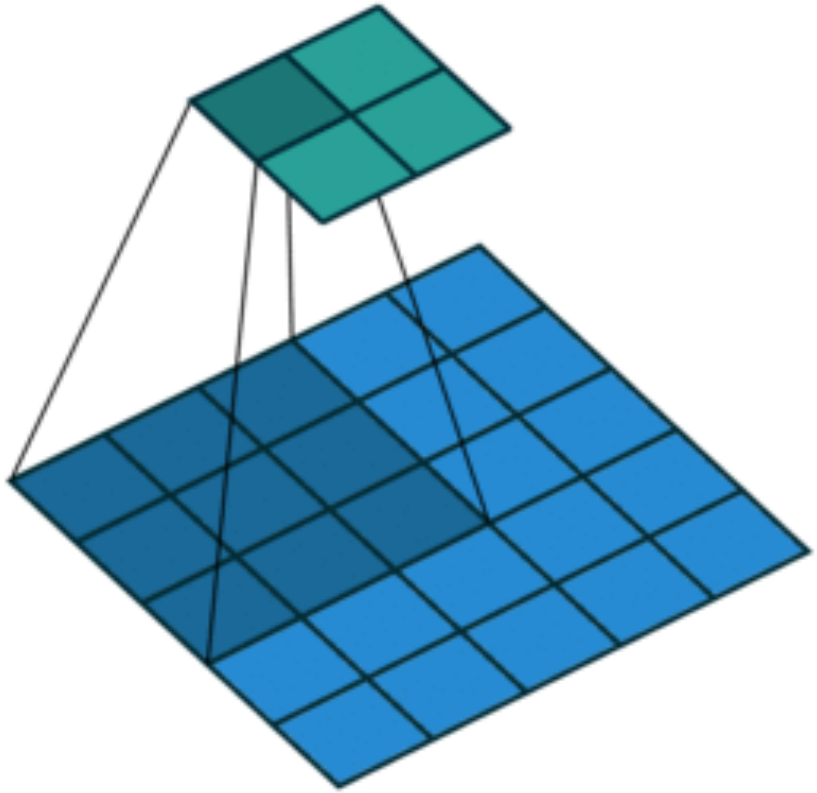



Credit

# OPTIONS: **STRIDES**

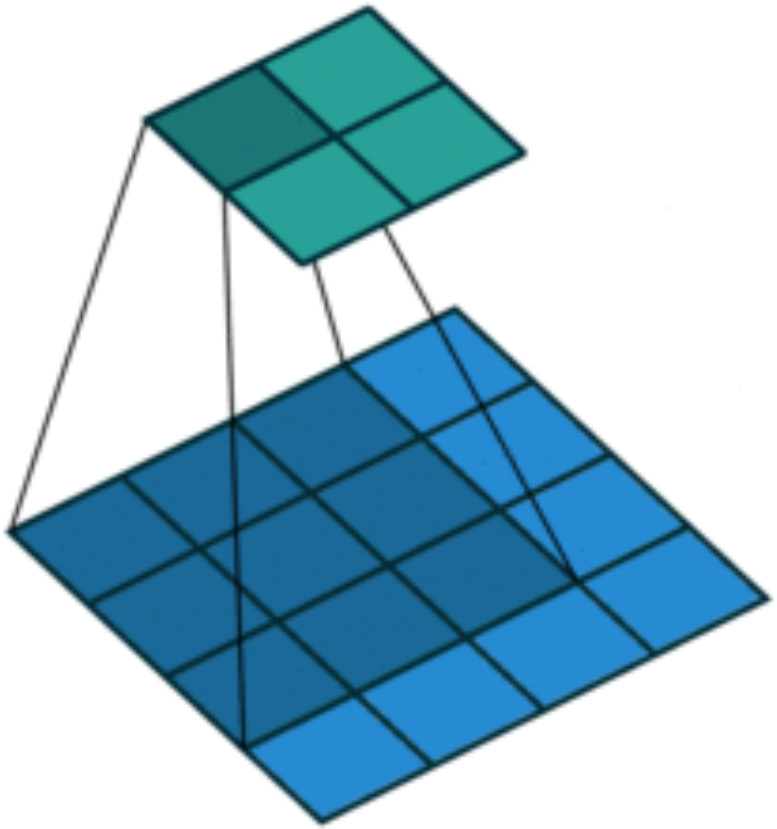


**NO STRIDES**

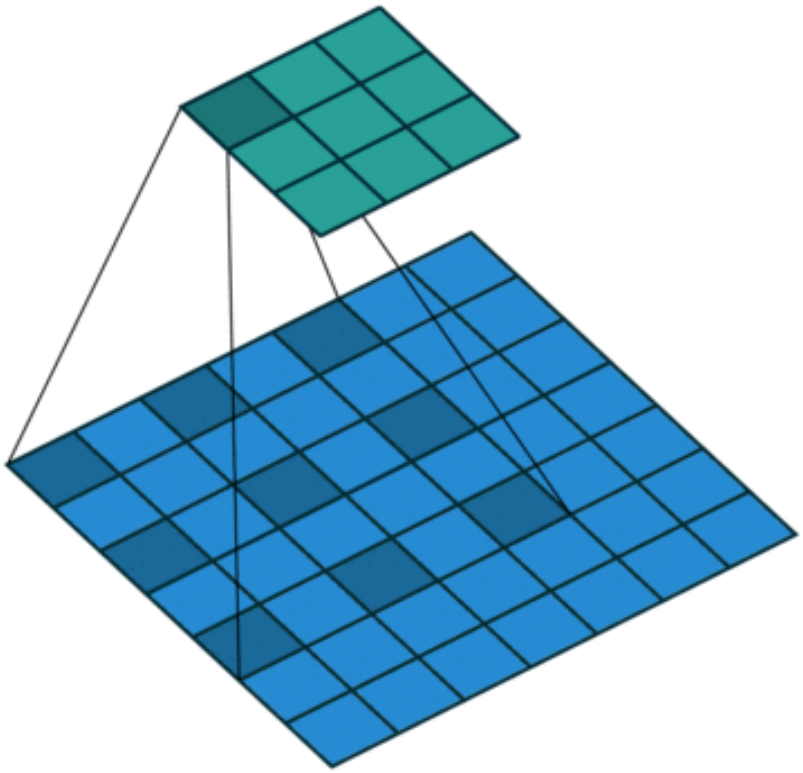


**STRIDES**

# OPTIONS: **DILATION**

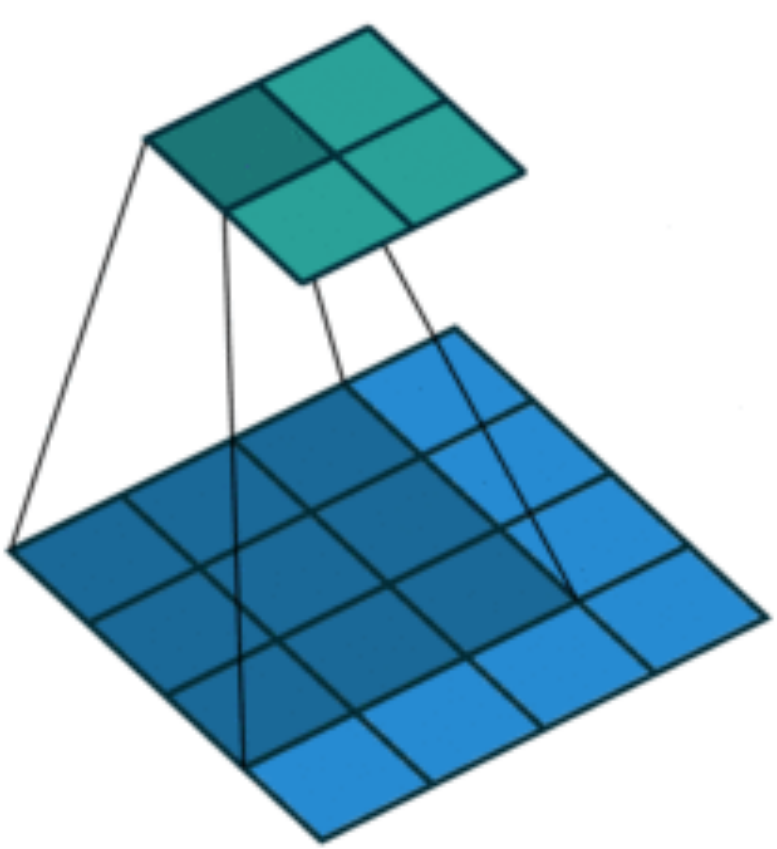


**NO STRIDES**

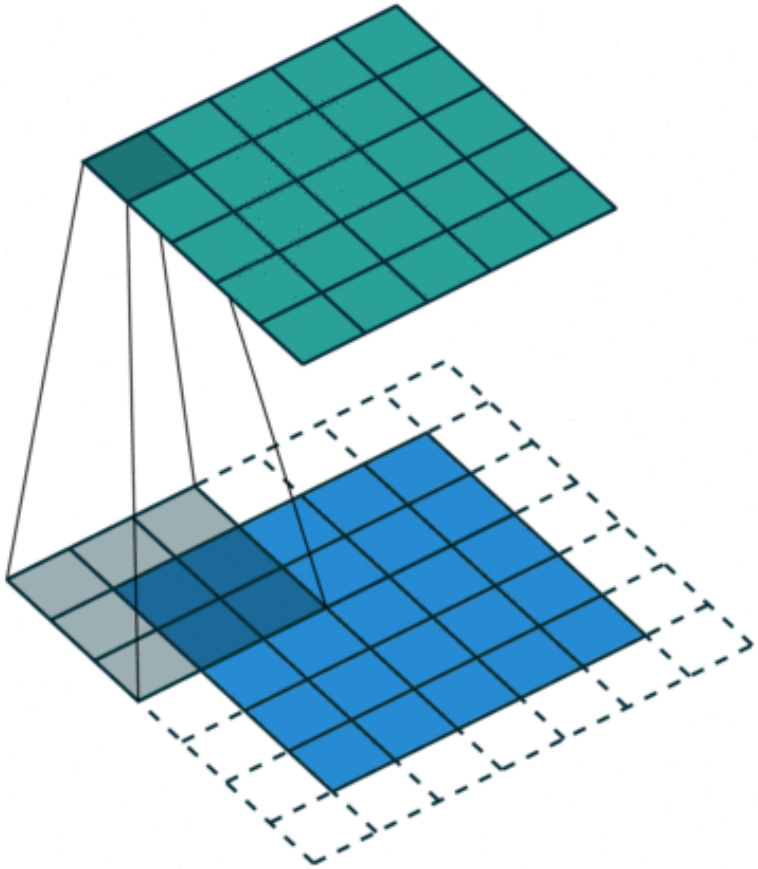


**DILATION**

# OPTIONS: **PADDING**

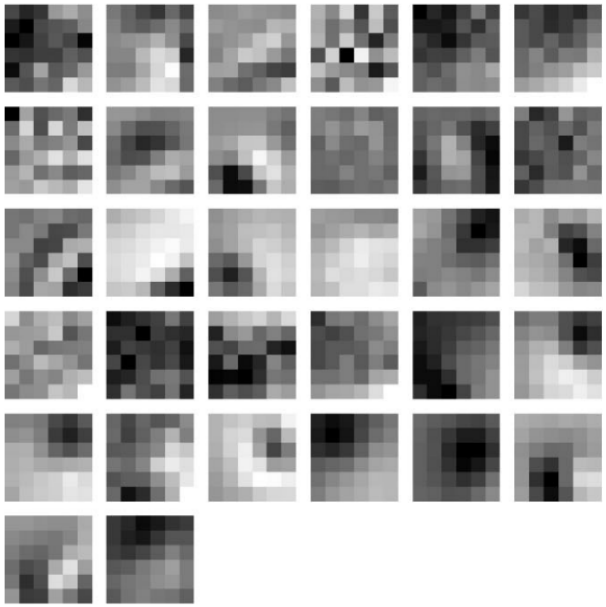


**NO STRIDES**

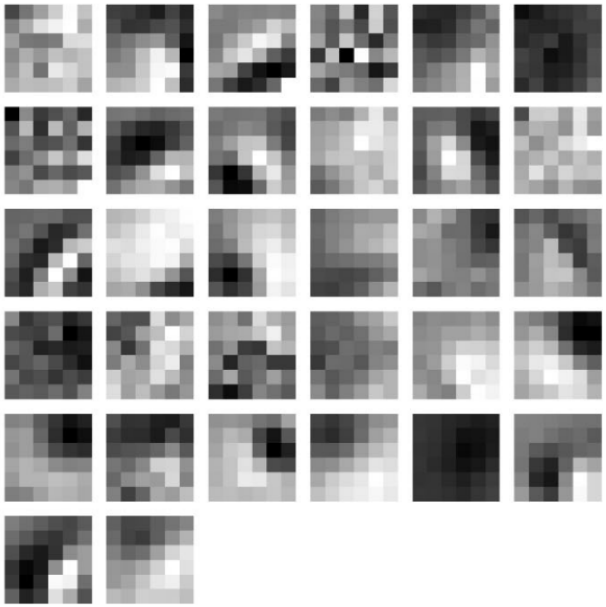


**PADDING**

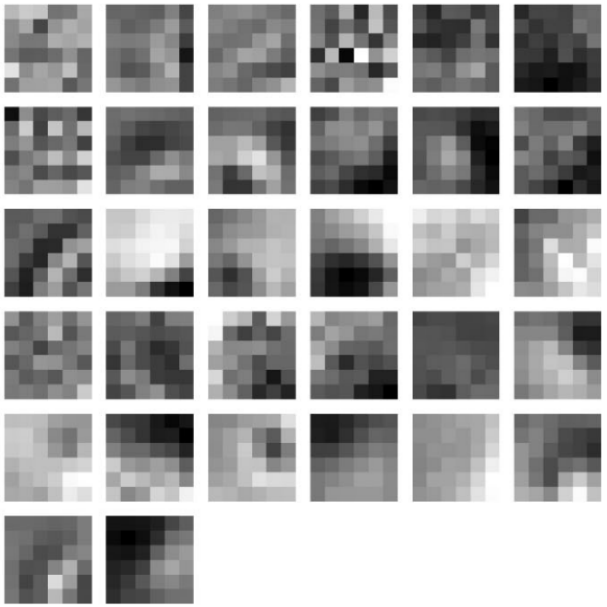
# EXAMPLE OF 32 FILTERS LEARNED IN A CONVOLUTIONAL LAYER



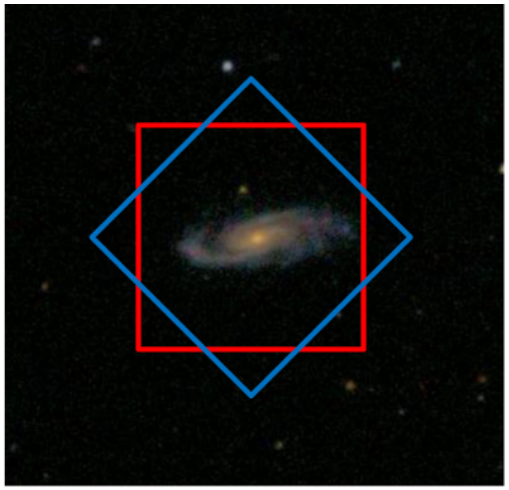
(a) red channel



(b) green channel



(c) blue channel



Dieleman+16

# ESTIMATING SHAPES AND NUMBER OF PARAMETERS

| KERNEL SHAPE: | PADDING: | STRIDES: |
|---|---|---|
| $(F, F, C^i, C^o)$ | $P$ | $S$ |

OUTPUT SIZE: $W_0 = (W^i - F + 2P)/S + 1$

NUMBER OF PARAMETERS: $(F \times F \times C^i + 1) \times C^o$

→ the number of parameters increases fast!

32 filters of 5*5 on a color image —> 2432 parameters to learn

# POOLING

CONVOLUTIONS ARE OFTEN FOLLOWED BY AN OPERATION OF DOWNSAMPLING [POOLING]

VERY SIMPLE OPERATION - ONLY ONE OUT OF EVERY N PIXELS ARE KEPT

OFTEN MATCHED WITH AN INCREASE OF THE FEATURE CHANNELS

# TYPES OF POOLING

SUM POOLING $y = \sum x_{uv}$

SQUARE SUM POOLING $y = \sqrt{\sum x_{uv}^2}$

MAX POOLING $y = max(x_{uv})$

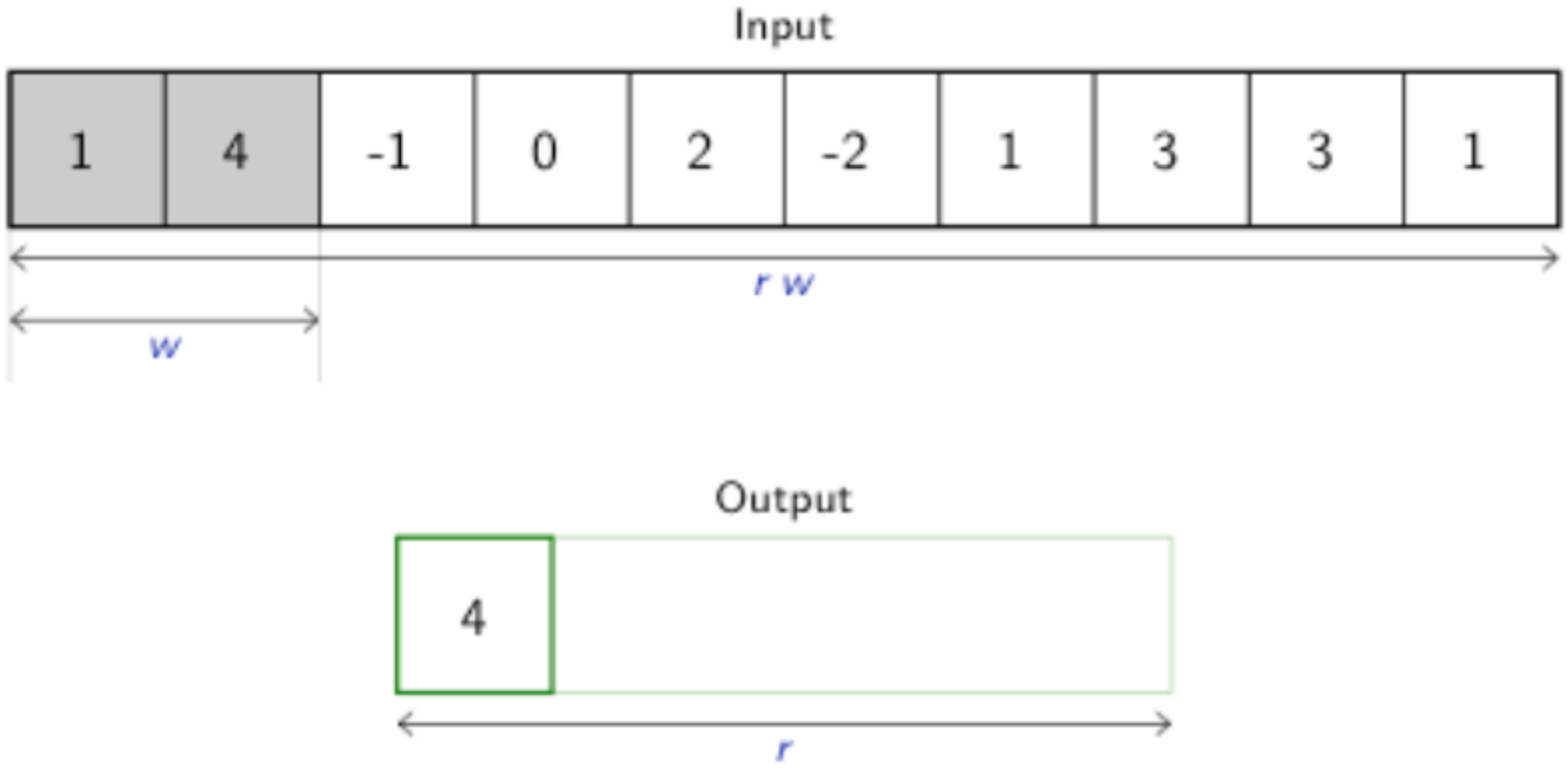
# TYPES OF POOLING

SUM POOLING $y = \sum x_{uv}$

SQUARE SUM POOLING $y = \sqrt{\sum x_{uv}^2}$

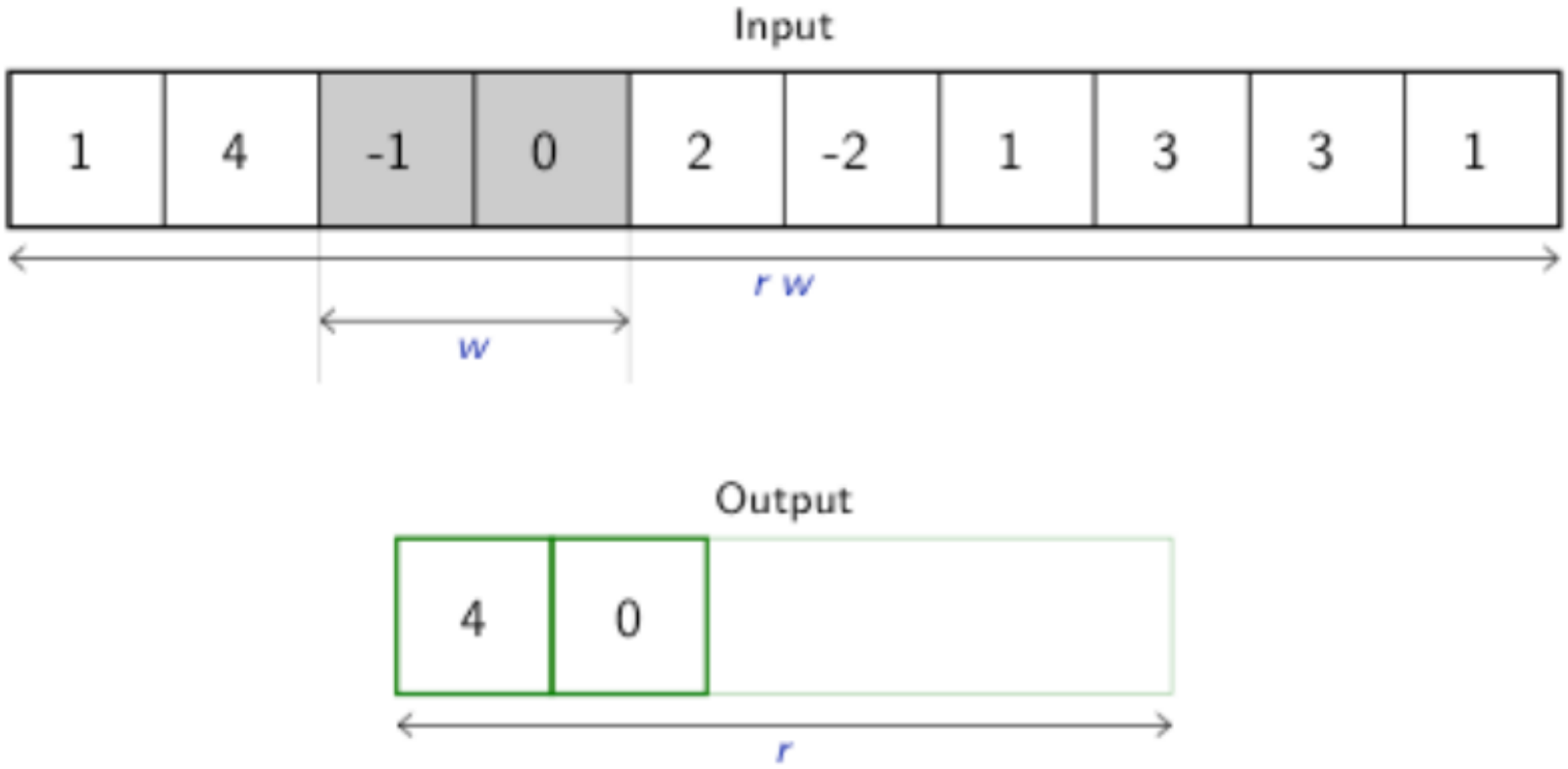
MAX POOLING $y = max(x_{uv})$

# MAX POOLING 1D



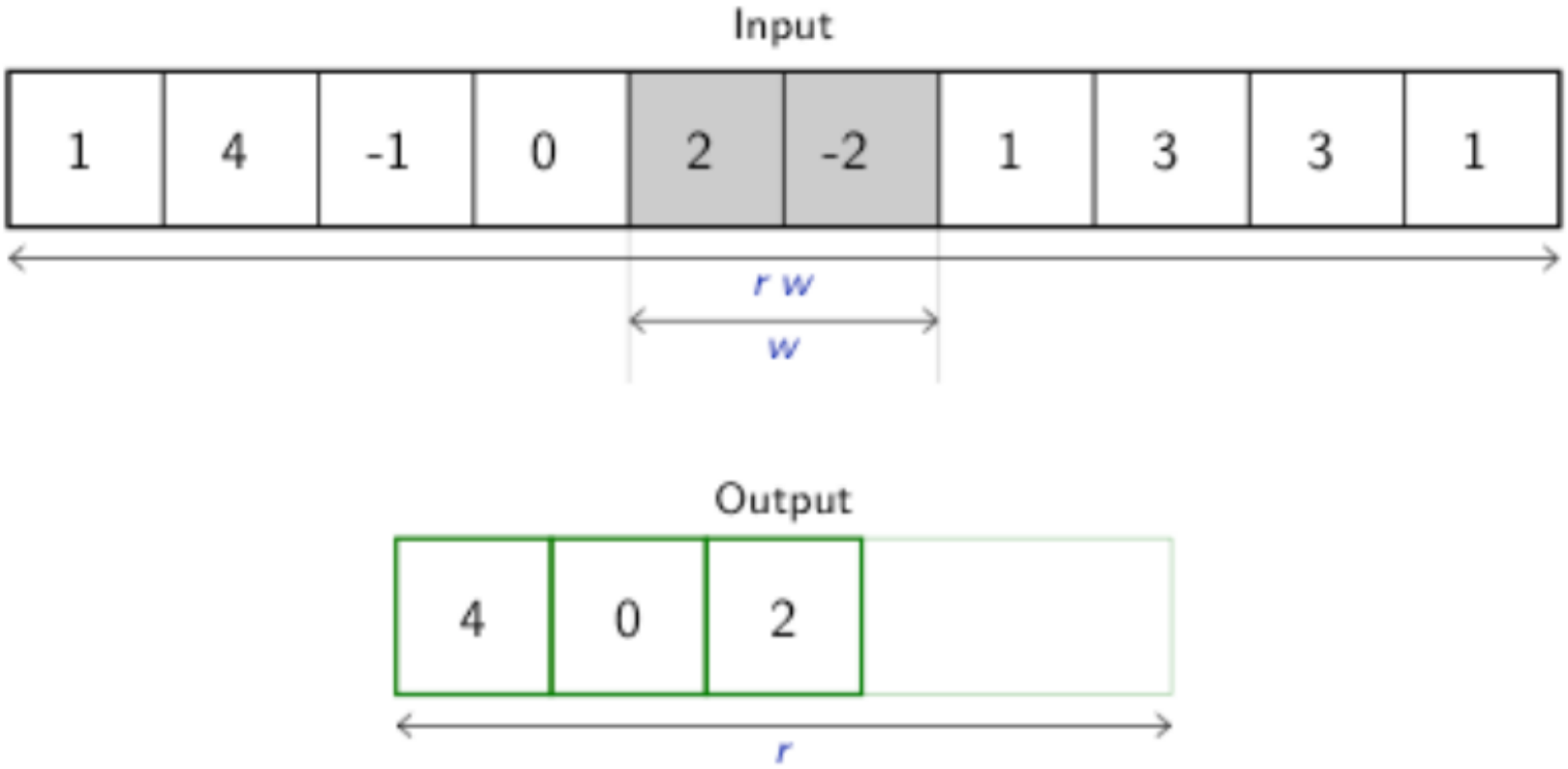

Credit: F. Fleuret

# MAX POOLING 1D



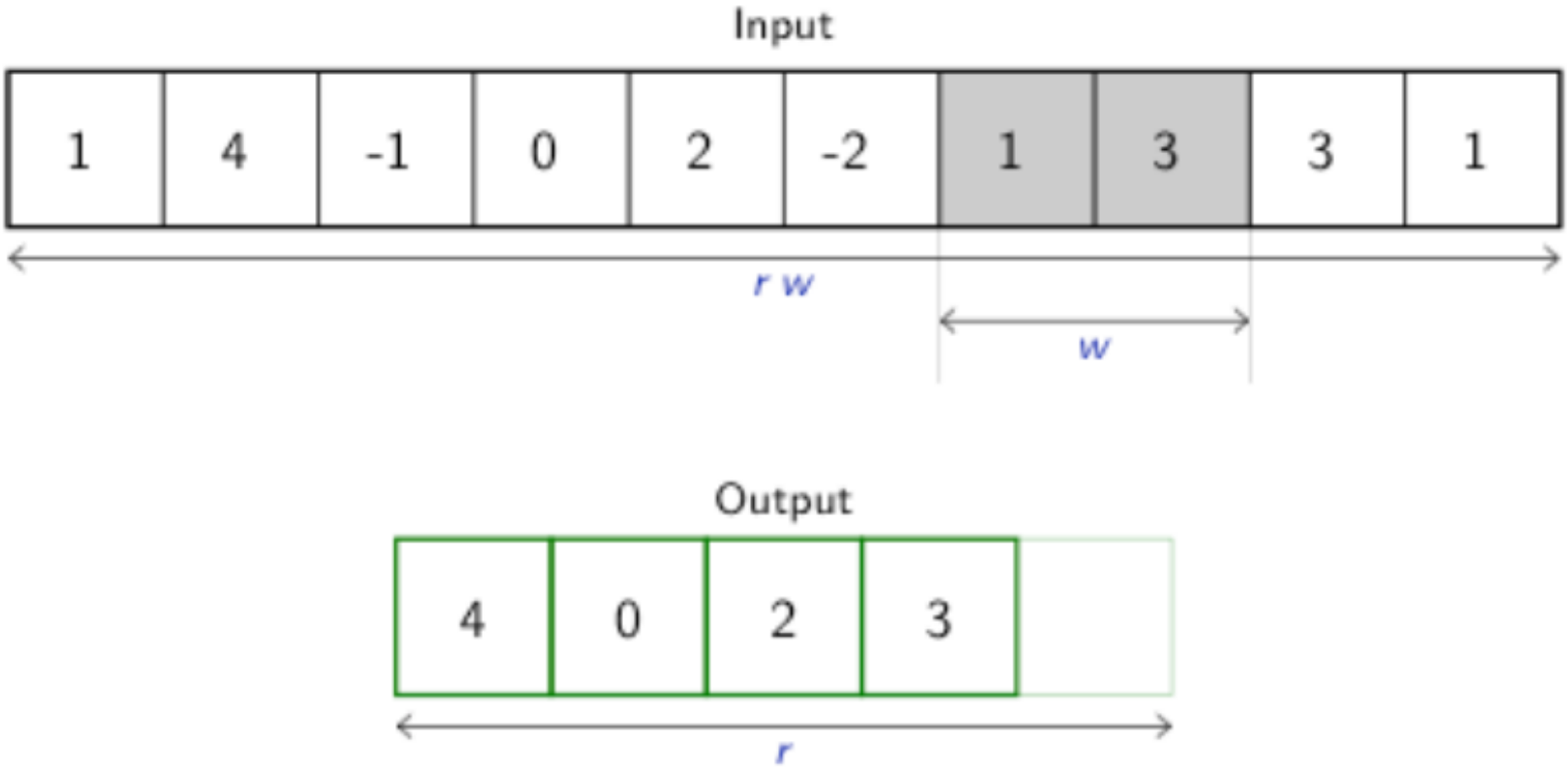

Credit: F. Fleuret

# MAX POOLING 1D



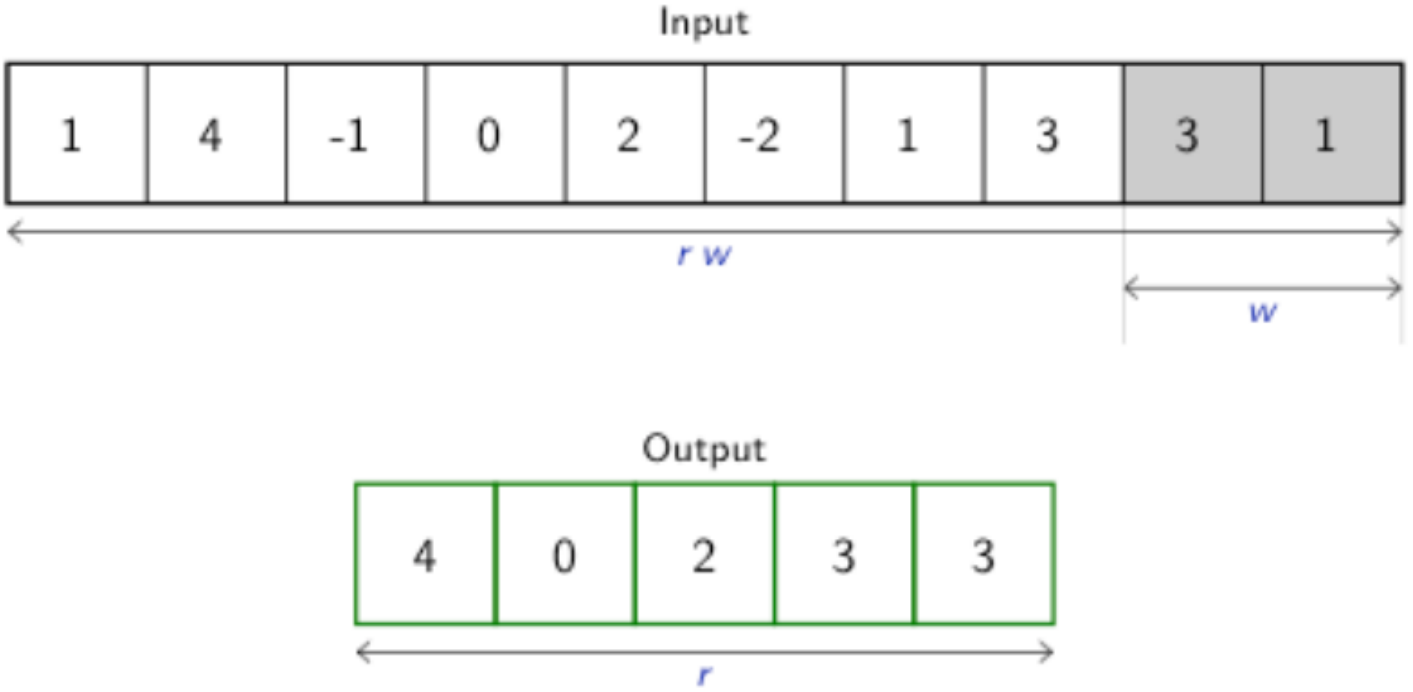

Credit: F. Fleuret

# MAX POOLING 1D




Credit: F. Fleuret

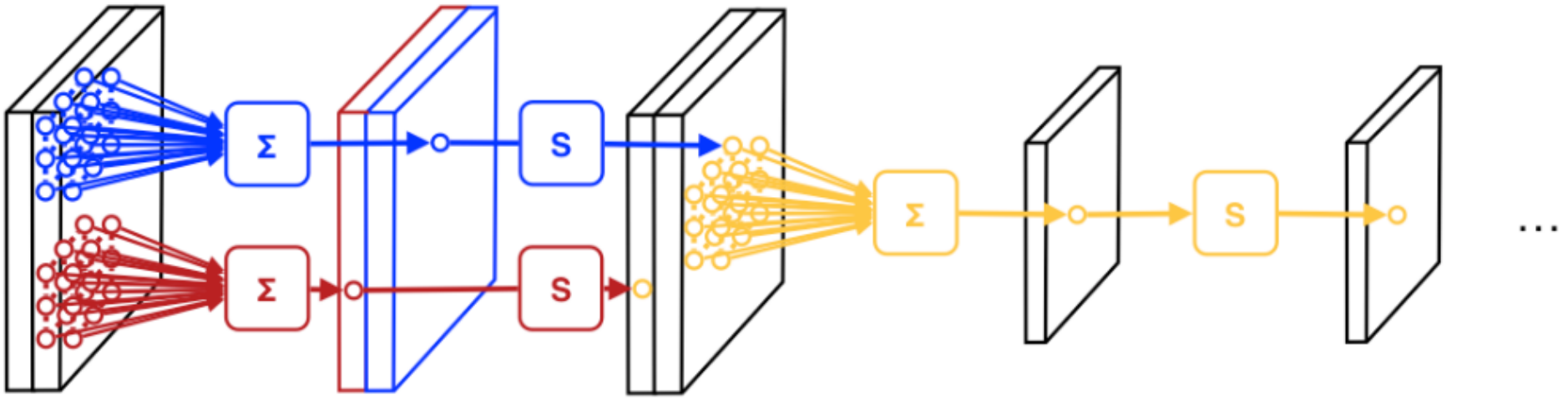
# MAX POOLING 1D



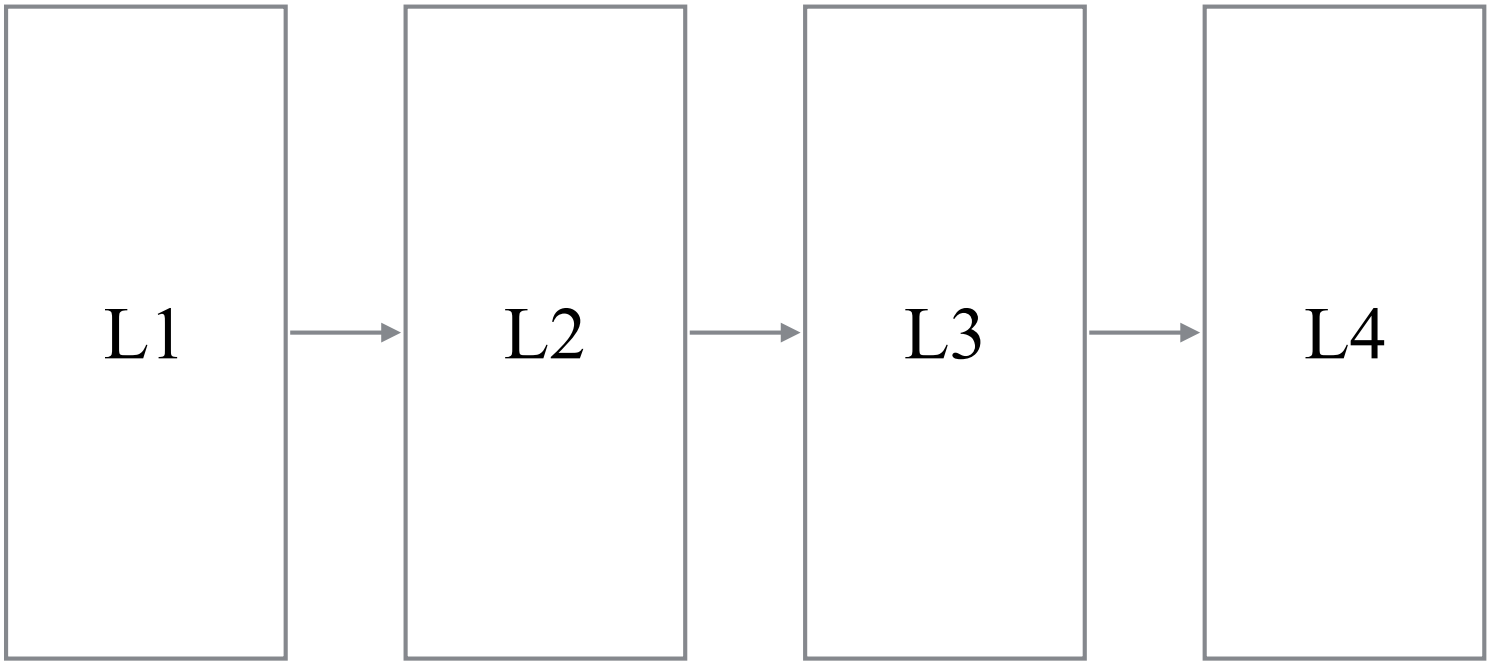

Credit: F. Fleuret

# DOWNSAMPLING

DOWNSAMPLING IS APPLIED TO REDUCE THE OVERALL SIZE OF TENSORS AND CAPTURE LARGE SCALE STRUCTURE

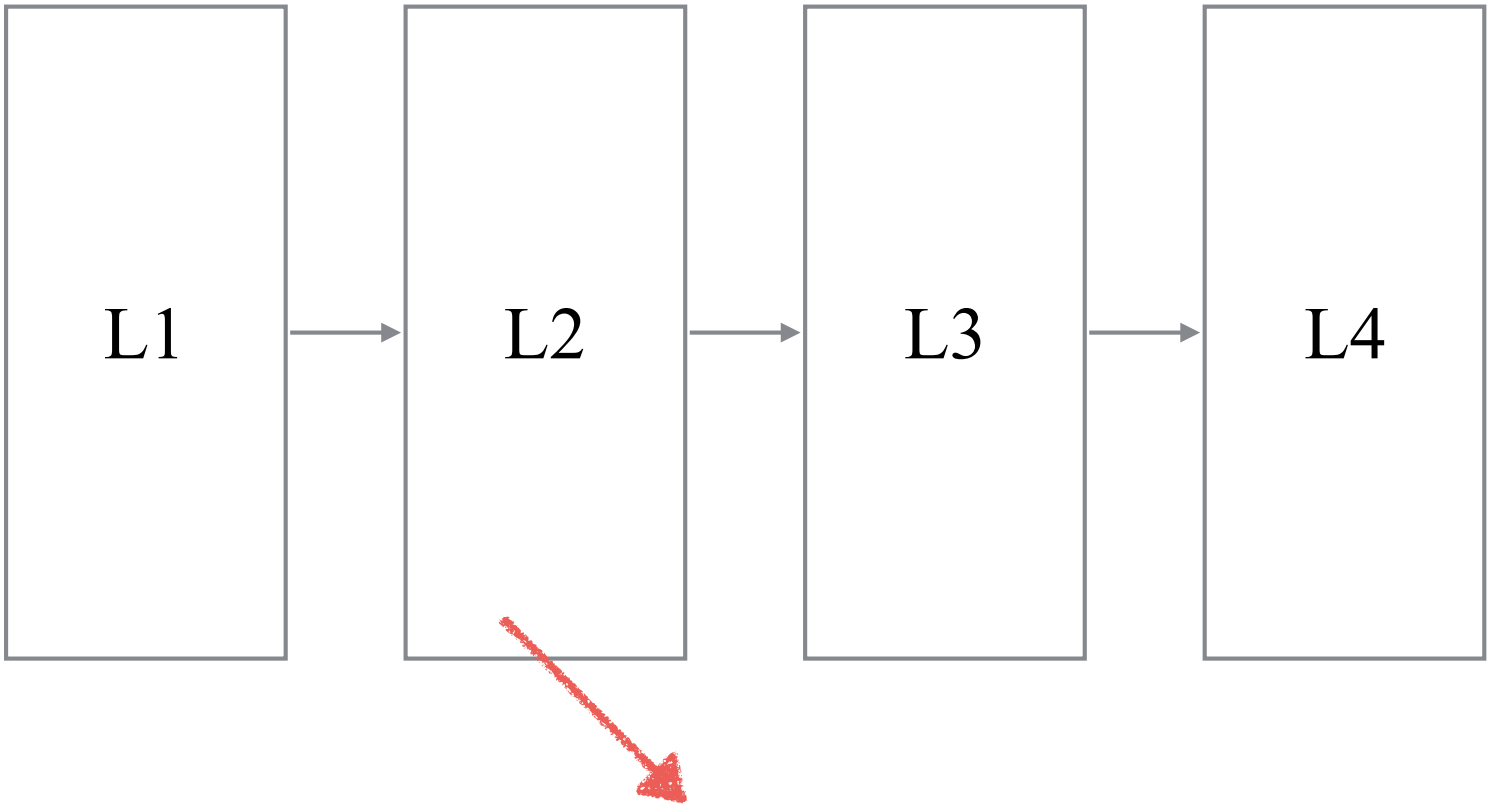


# CONVNET OR CNN




A CONCATENATION OF MULTIPLE CONVOLUTIONAL BLOCKS

# CONVNET OR CNN

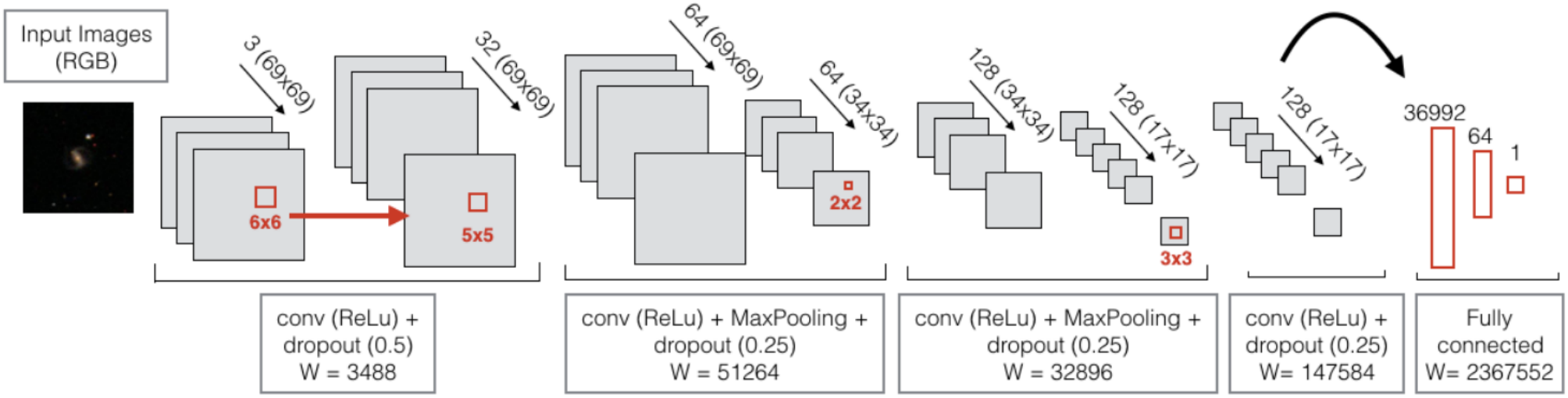



EACH BLOCK TYPICALLY MADE OF:

CONV | ACTIVATION | POOLING

(+dropout for training)

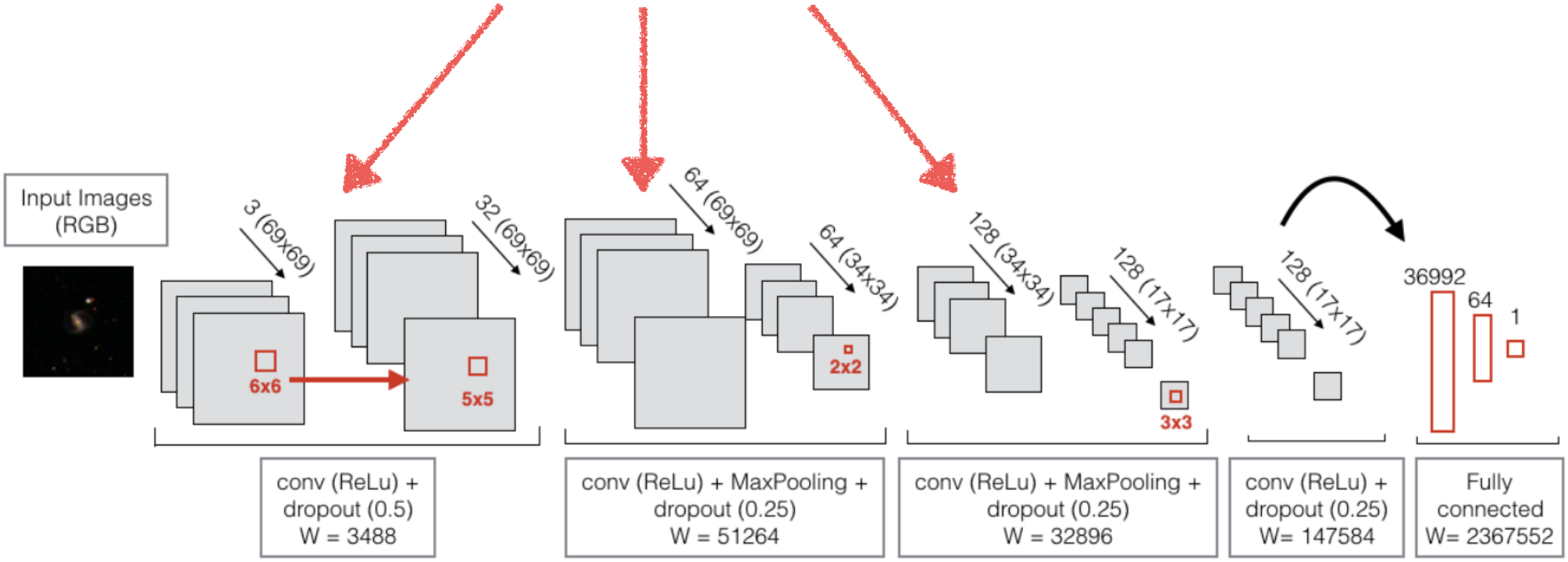
# EXAMPLE OF VERY SIMPLE CNN




Dominguez-Sanchez+18

# EXAMPLE OF VERY SIMPLE CNN

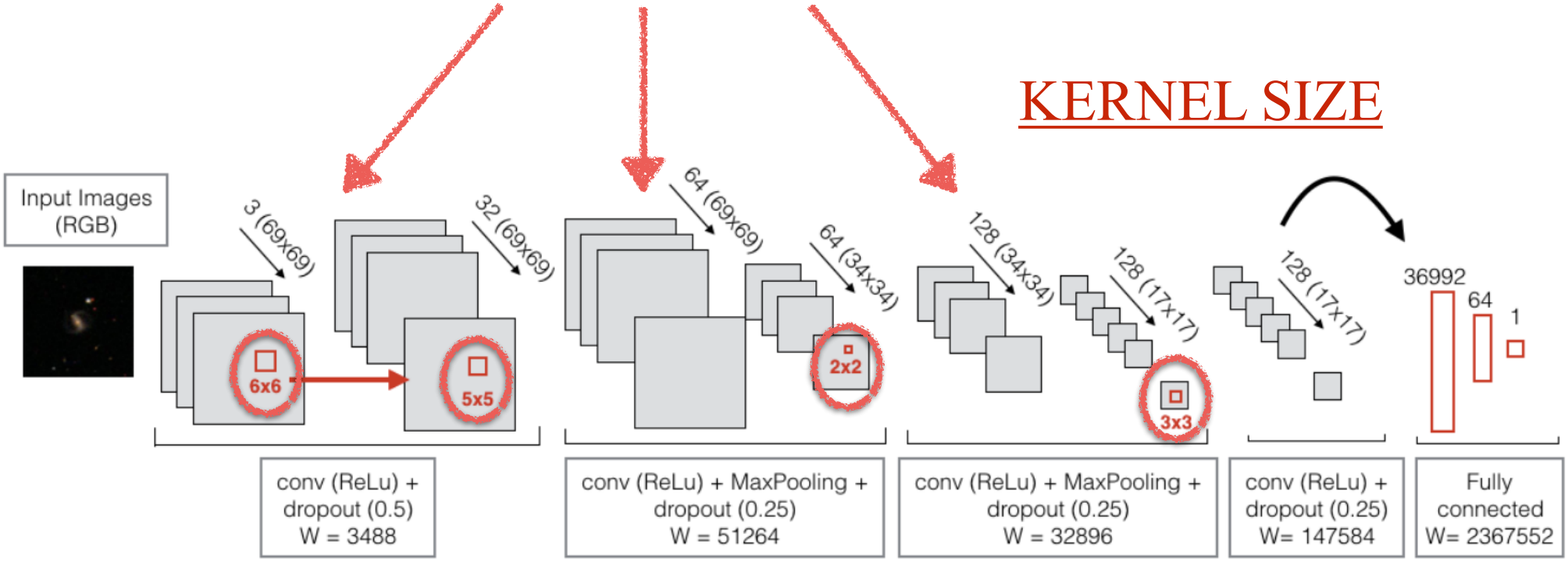
## 3 convolutional layers




Dominguez-Sanchez+18

# EXAMPLE OF VERY SIMPLE CNN

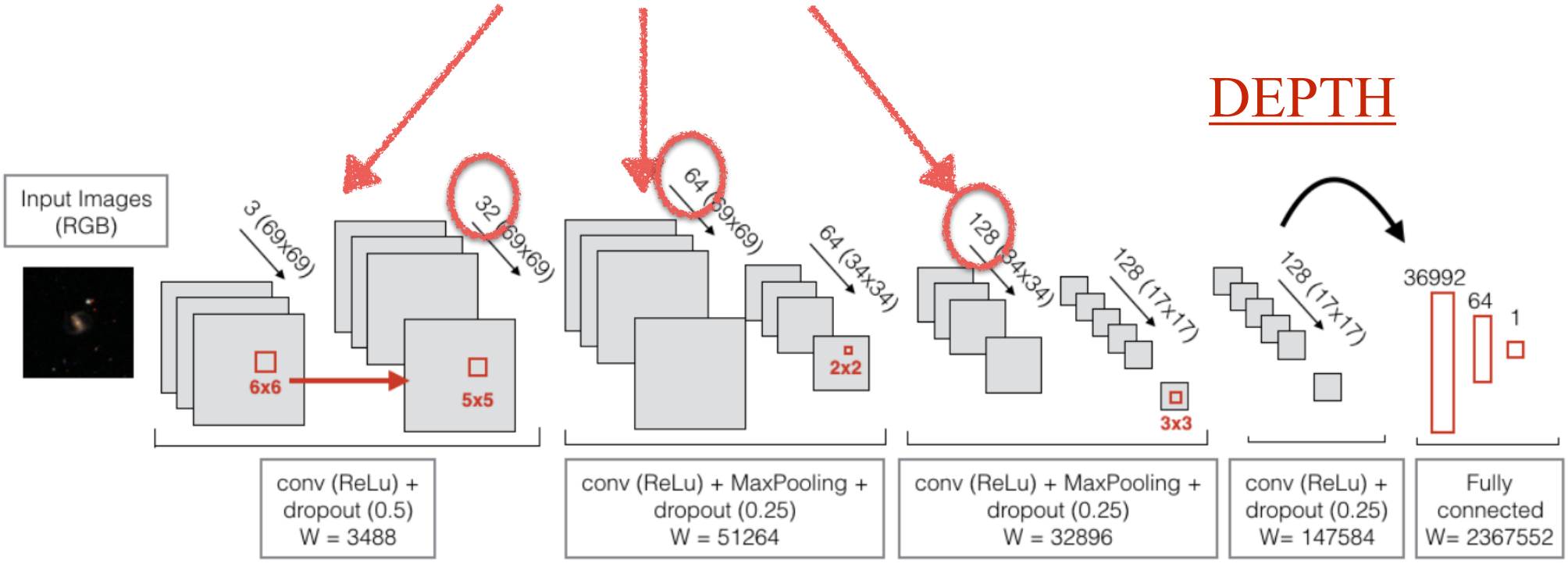
## 3 convolutional layers




Dominguez-Sanchez+18

# EXAMPLE OF VERY SIMPLE CNN

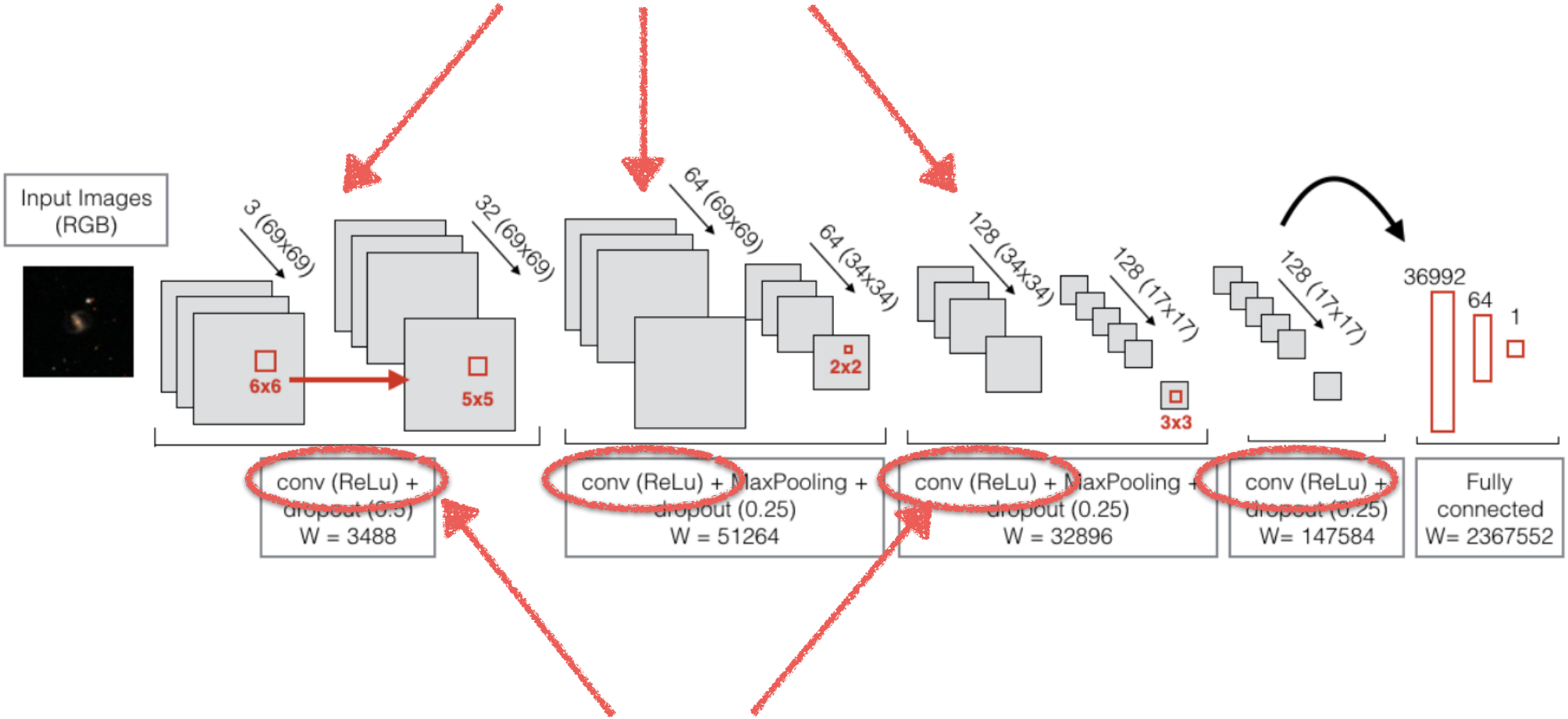
## 3 convolutional layers




Dominguez-Sanchez+18

# EXAMPLE OF VERY SIMPLE CNN

## 3 convolutional layers

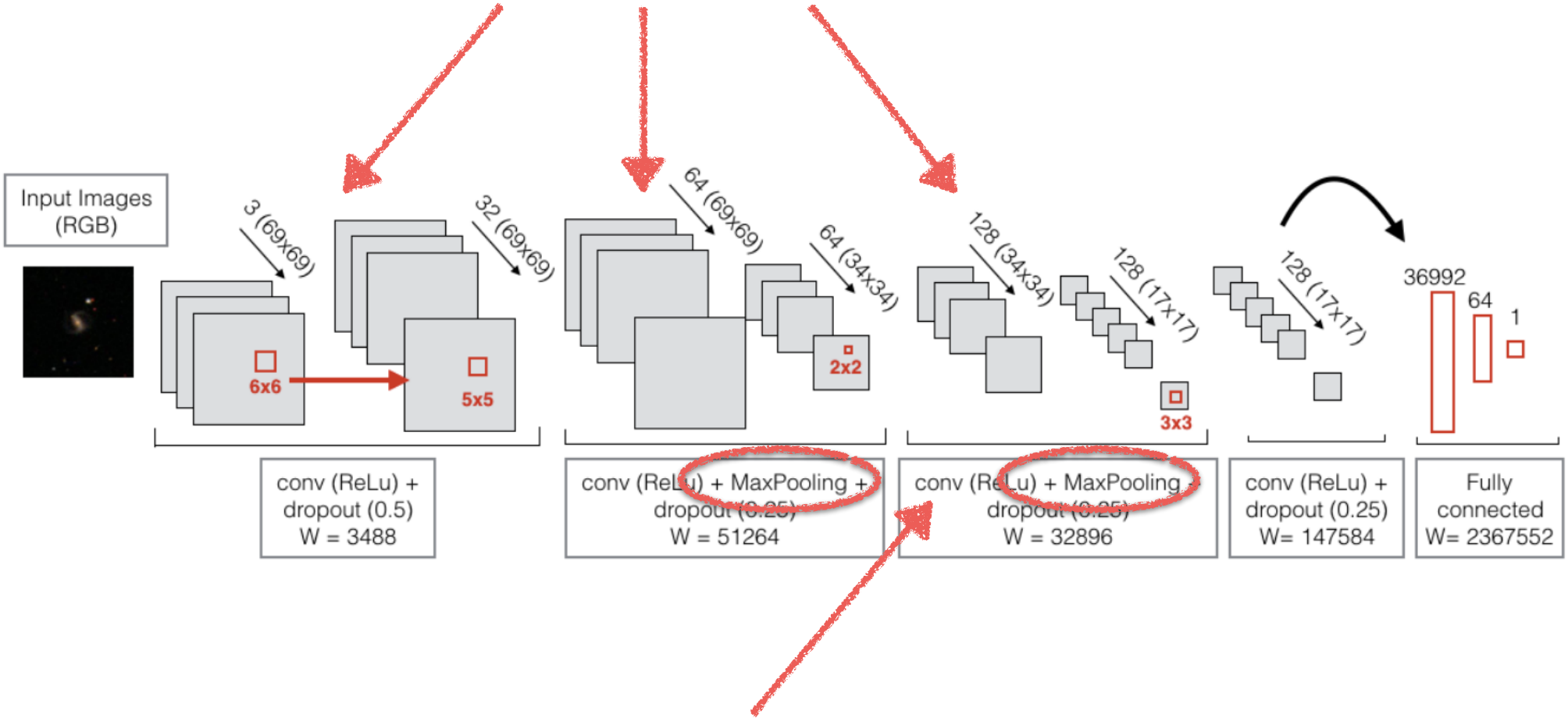



## ReLu activation

Dominguez-Sanchez+18

# EXAMPLE OF VERY SIMPLE CNN

## 3 convolutional layers

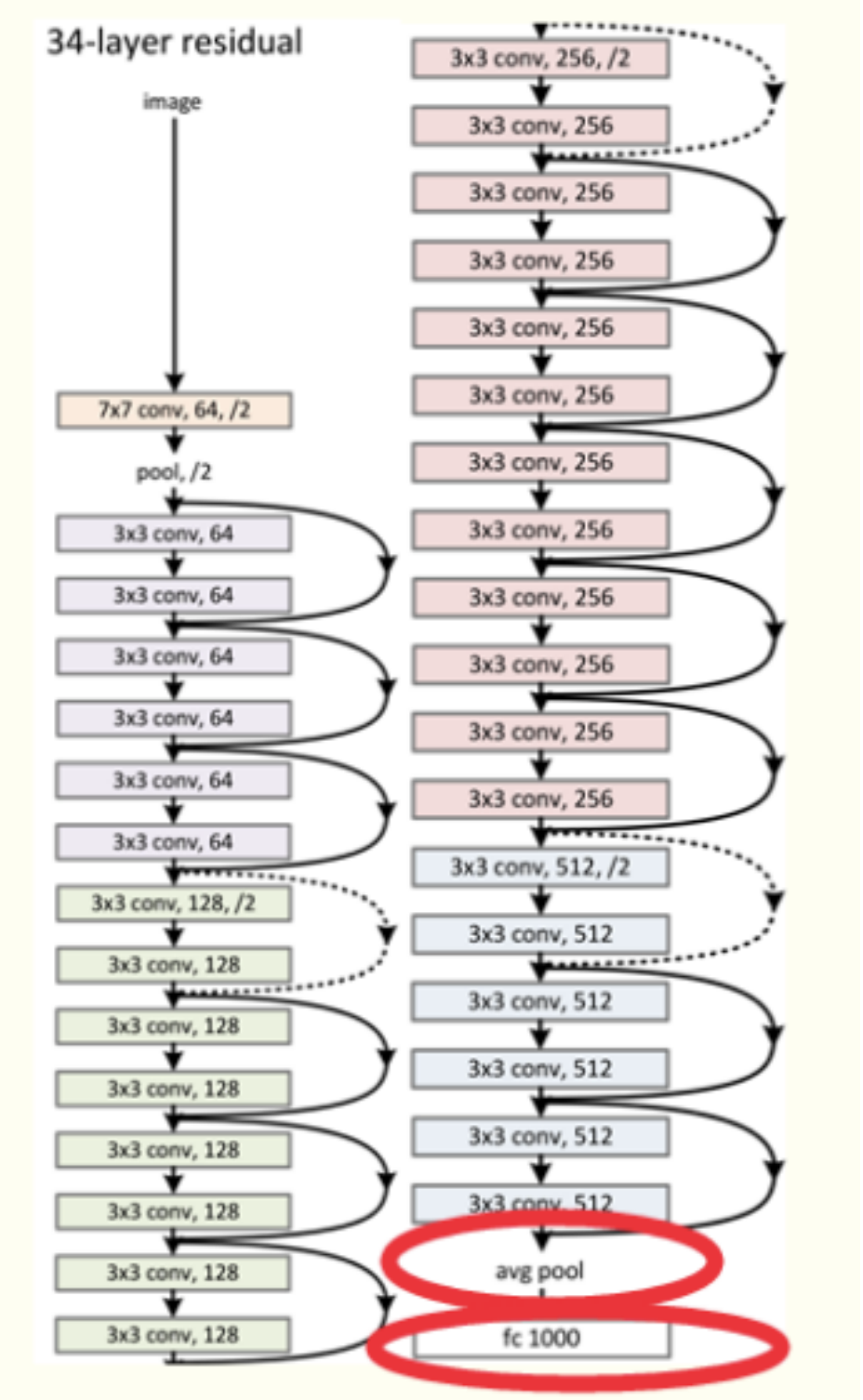



## Pooling

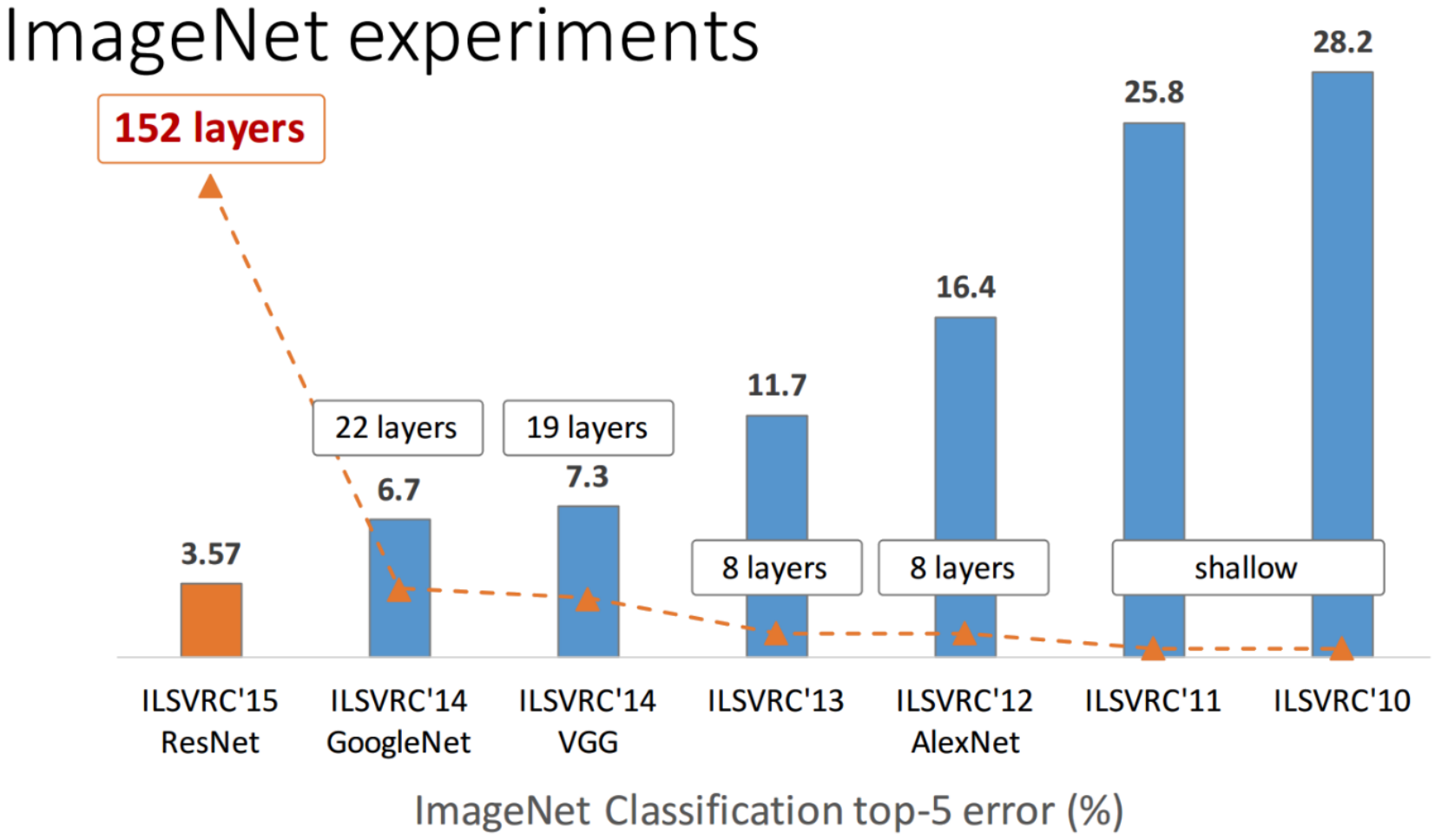
Dominguez-Sanchez+18

# IN THE REAL LIFE..

RESNET



# DEEPER TENDS TO BE BETTER…



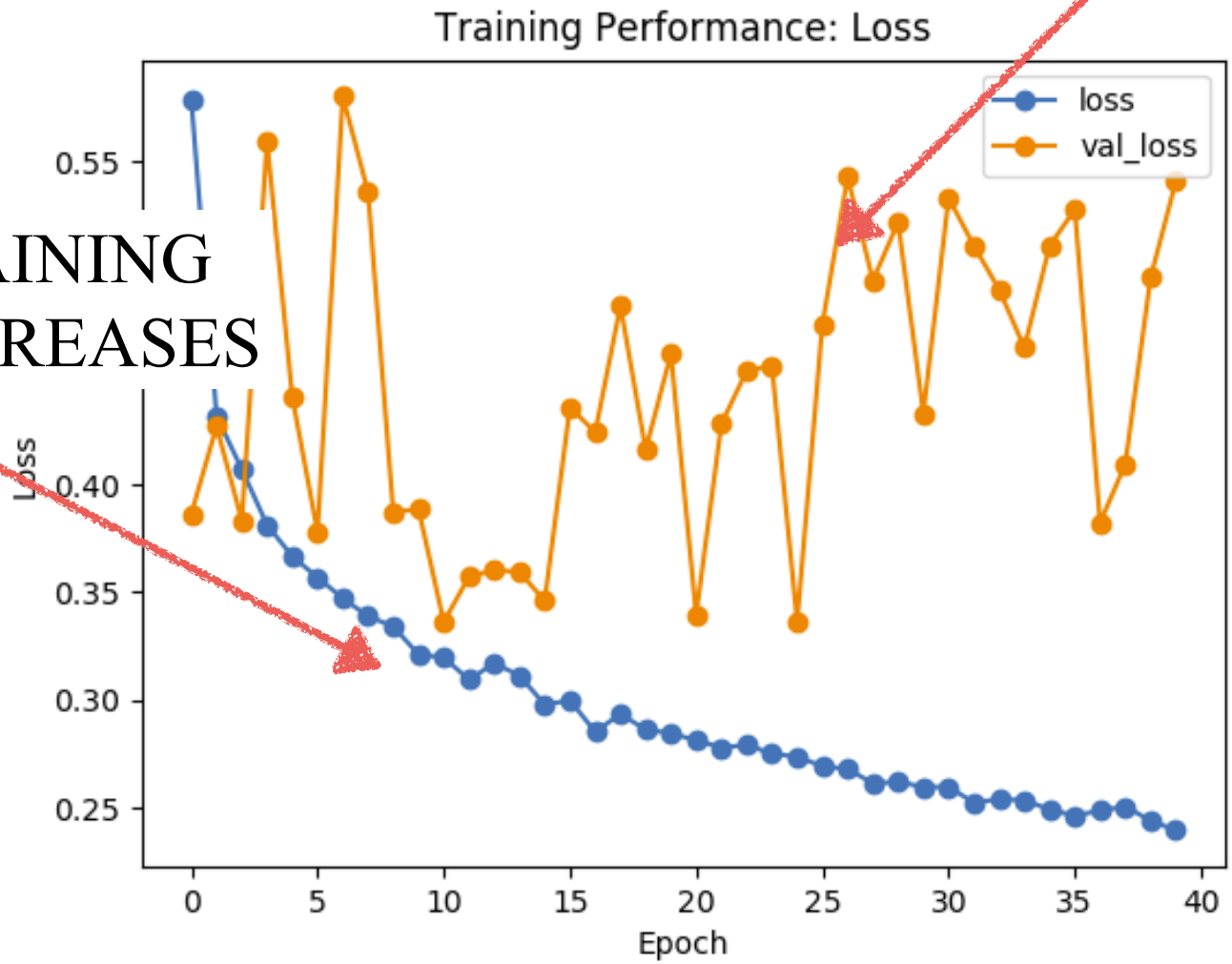
# THE PROBLEMS OF GOING "TOO DEEP"

- DEEP NETWORKS ARE MORE DIFFICULT TO OPTIMIZE
- NEED MORE DATA - MORE SUBJECT TO OVER-FITTING
- AND ALSO NEED MORE TIME …

# OVER-FITTING

THE TEST STAYS CONSTANT OR INCREASES

THE TRAINING LOSS DECREASES

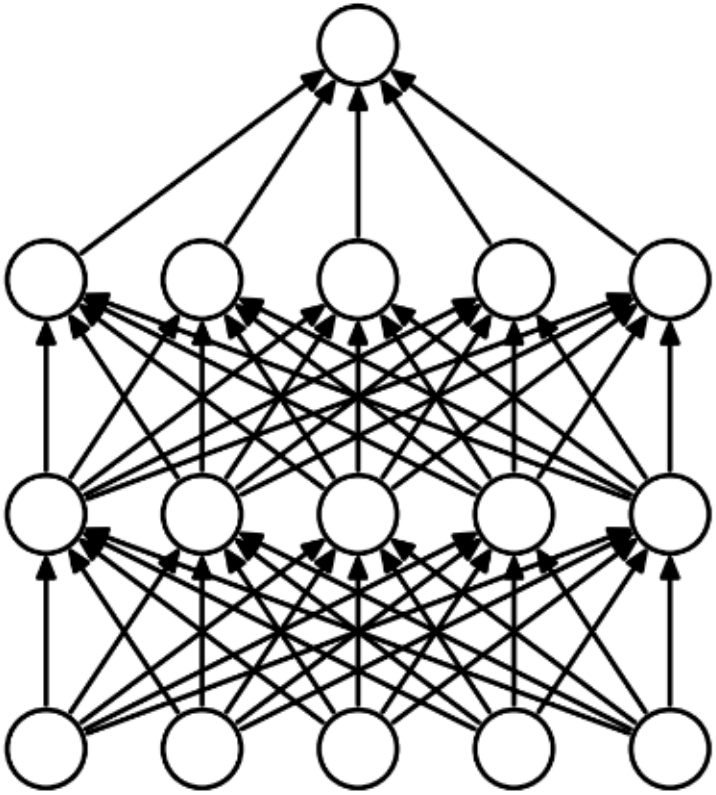


# DROPOUT

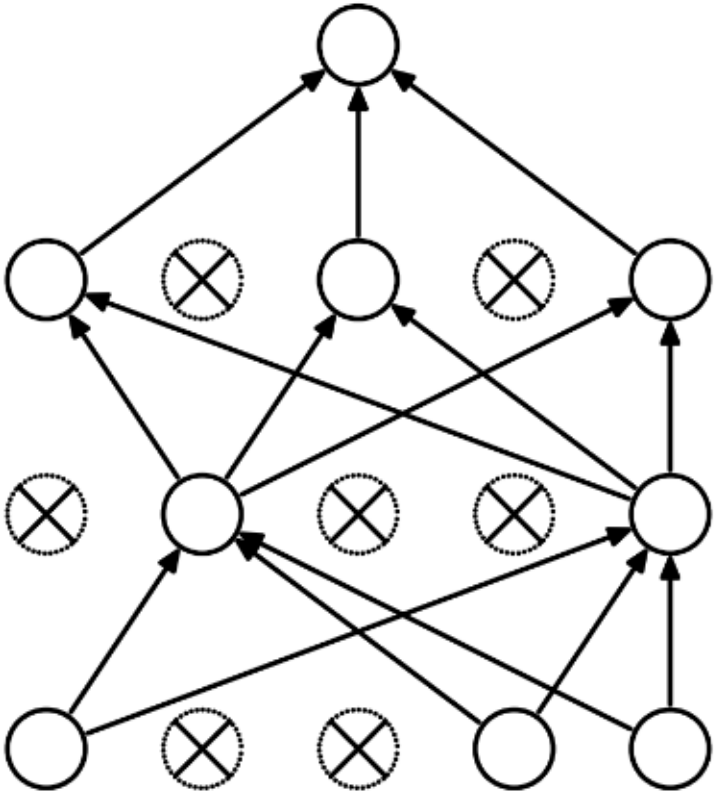
[Hinton+12]

- THE IDEA IS TO REMOVE NEURONS RANDOMLY DURING THE TRAINING

- ALL NEURONS ARE PUT BACK DURING THE TEST PHASE



(a) Standard Neural Net



(b) After applying dropout.

# DROPOUT

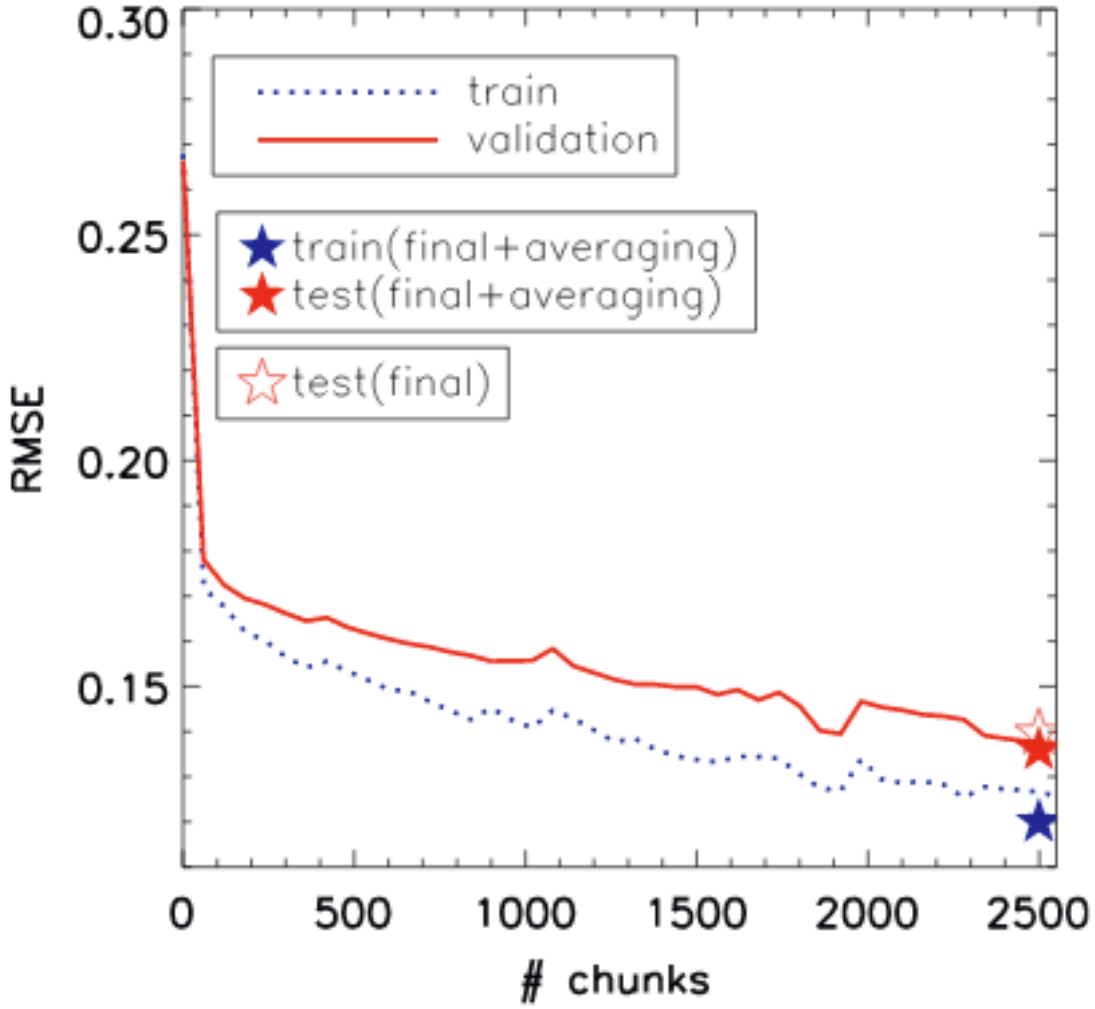
## WHY DOES IT WORK?

1. SINCE NEURONS ARE REMOVED RANDOMLY, IT AVOIDS CO-ADAPTATION AMONG THEMSELVES

2. DIFFERENT SETS OF NEURONS WHICH ARE SWITCHED OFF, REPRESENT A DIFFERENT ARCHITECTURE AND ALL THESE DIFFERENT ARCHITECTURES ARE TRAINED IN PARALLEL. FOR N NEURONS ATTACHED TO DROPOUT, THE NUMBER OF SUBSET ARCHITECTURES FORMED IS 2^N. SO IT AMOUNTS TO PREDICTION BEING AVERAGED OVER THESE ENSEMBLES OF MODELS.

# DROPOUT

WITH A LITTLE BIT
OF DROPOUT



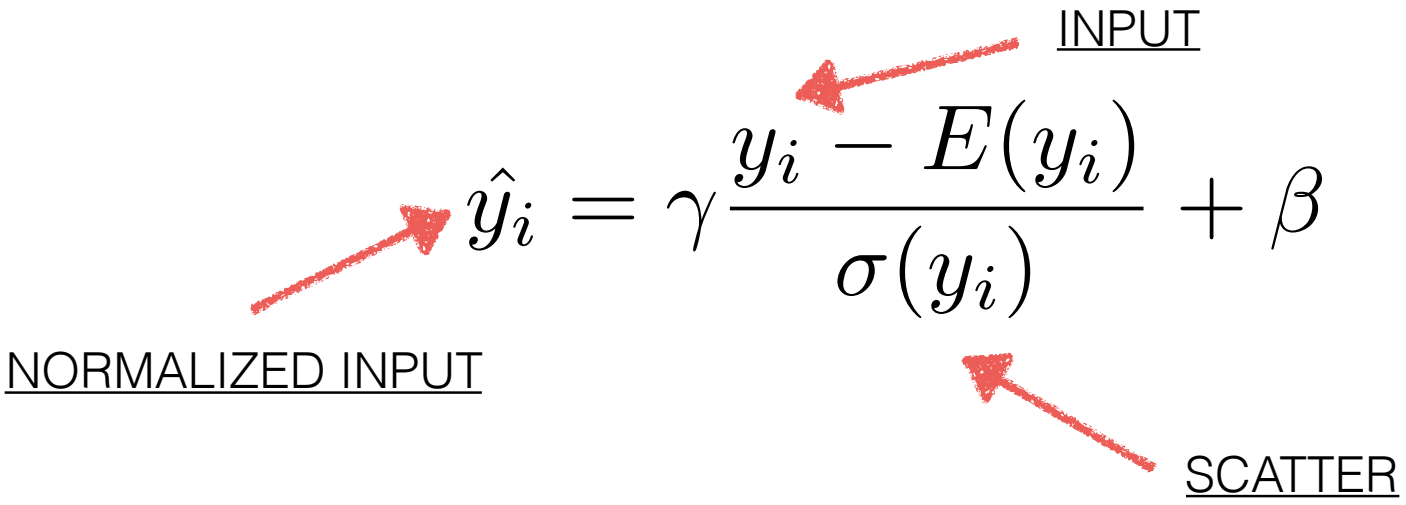

Huertas-Company+15

# BATCH NORMALIZATION

[SZEGEDY+15]

ANOTHER SOLUTION TO KEEP REASONABLE VALUES OF THE ACTIVATIONS IN DEEP NETWORKS

**BATCH NORMALIZATION** PREVENTS LOW OR LARGE VALUES BY RE-NORMALIZING THE VALUES BEFORE ACTIVATION FOR EVERY BATCH



# BATCH NORMALIZATION

[SZEGEDY+15]

**BATCH NORMALIZATION** SPEEDS UP AND STABILIZES TRAINING

**Input:** Values of $x$ over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: $\gamma$, $\beta$

**Output:** $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma\widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

# BATCH NORMALIZATION

## [SZEGEDY+15]

IN KERAS, IT IS IMPLEMENTED AS AN ADDITIONAL LAYER

**BatchNormalization** [source]

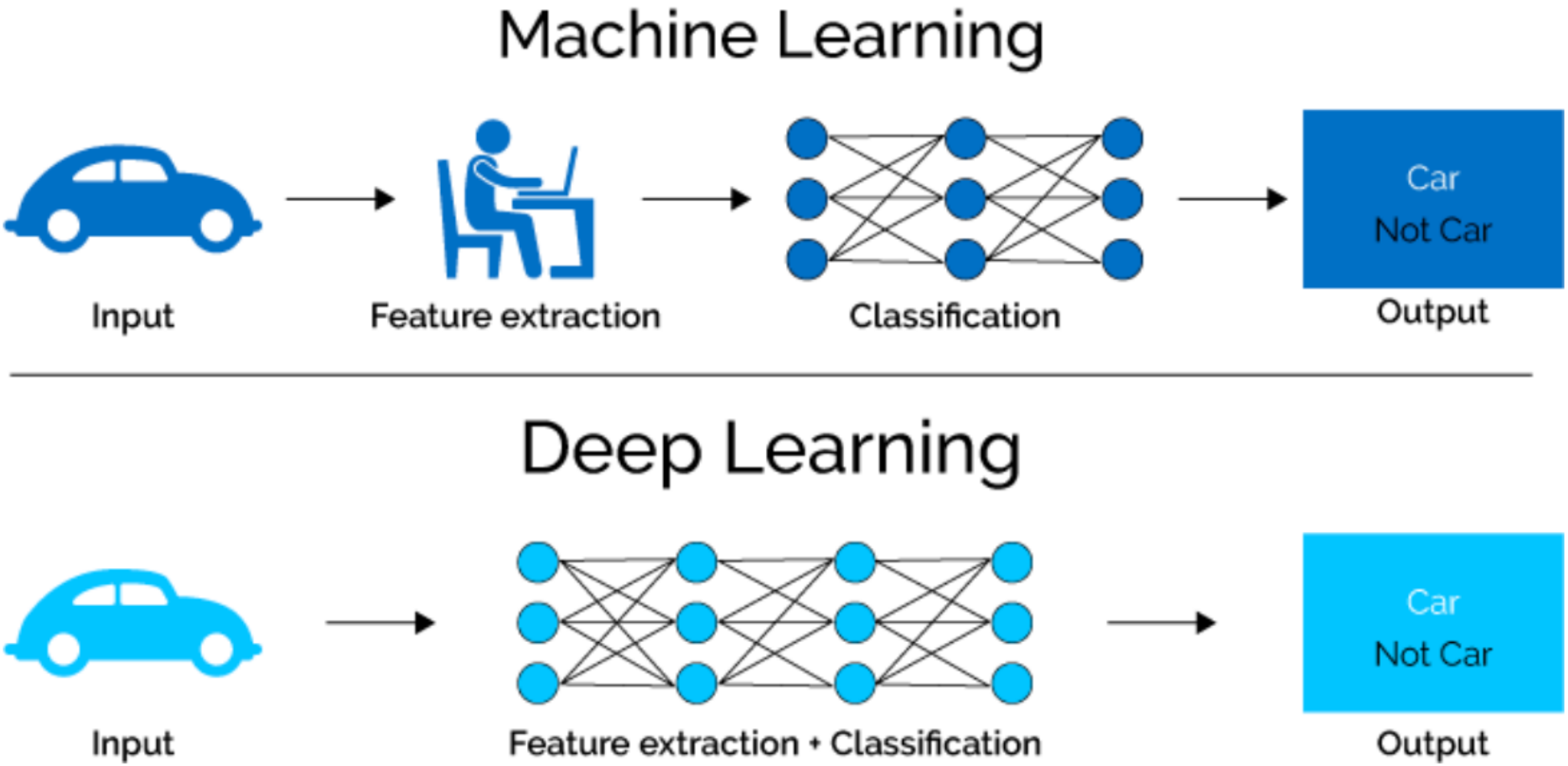
```
keras.layers.BatchNormalization(axis=-1, momentum=0.99, epsilon=0.001, center=True, scale=True, beta_initializer
```

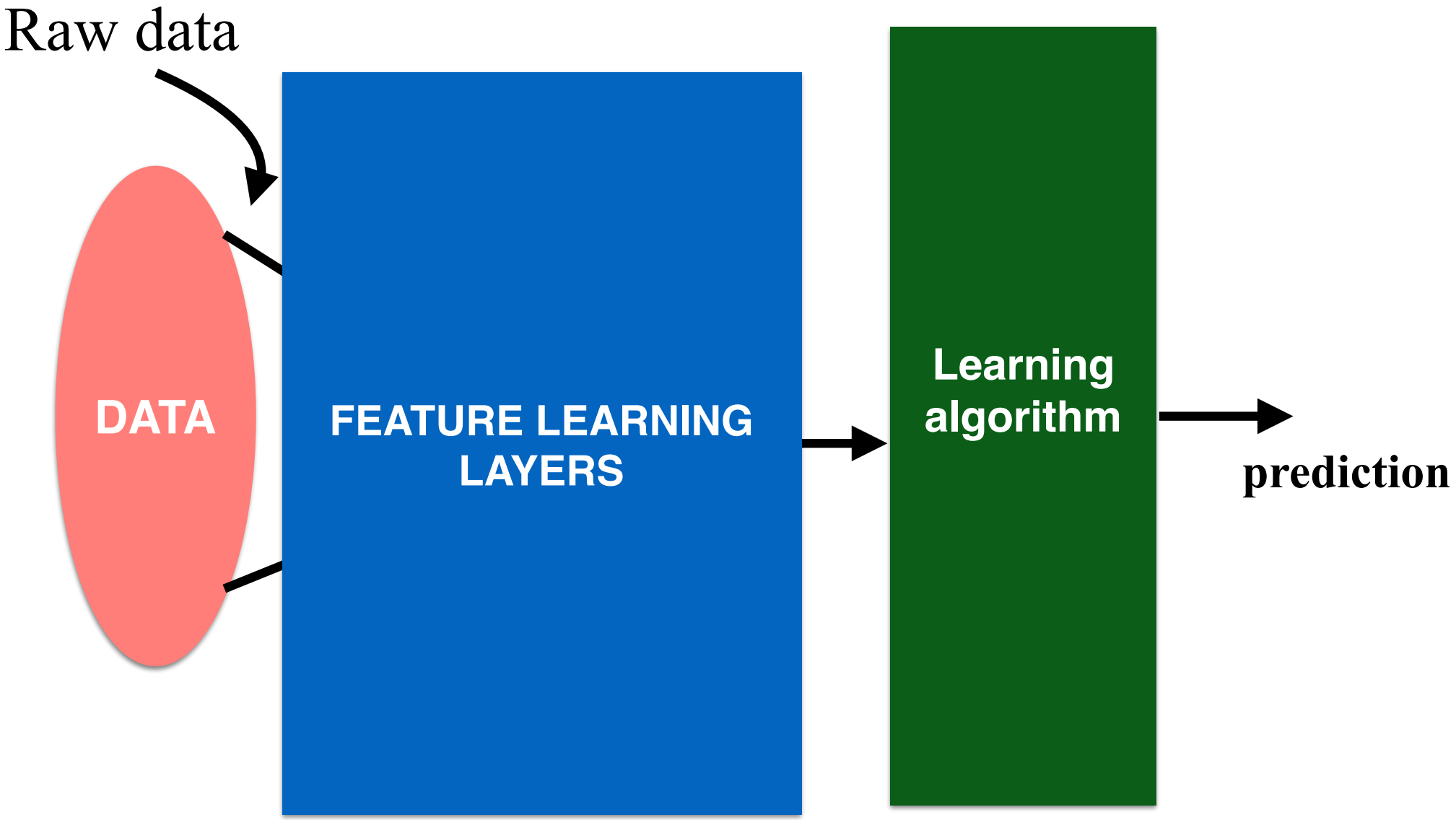
Batch normalization layer (Ioffe and Szegedy, 2014).

Normalize the activations of the previous layer at each batch, i.e. applies a transformation that maintains the mean activation close to 0 and the activation standard deviation close to 1.

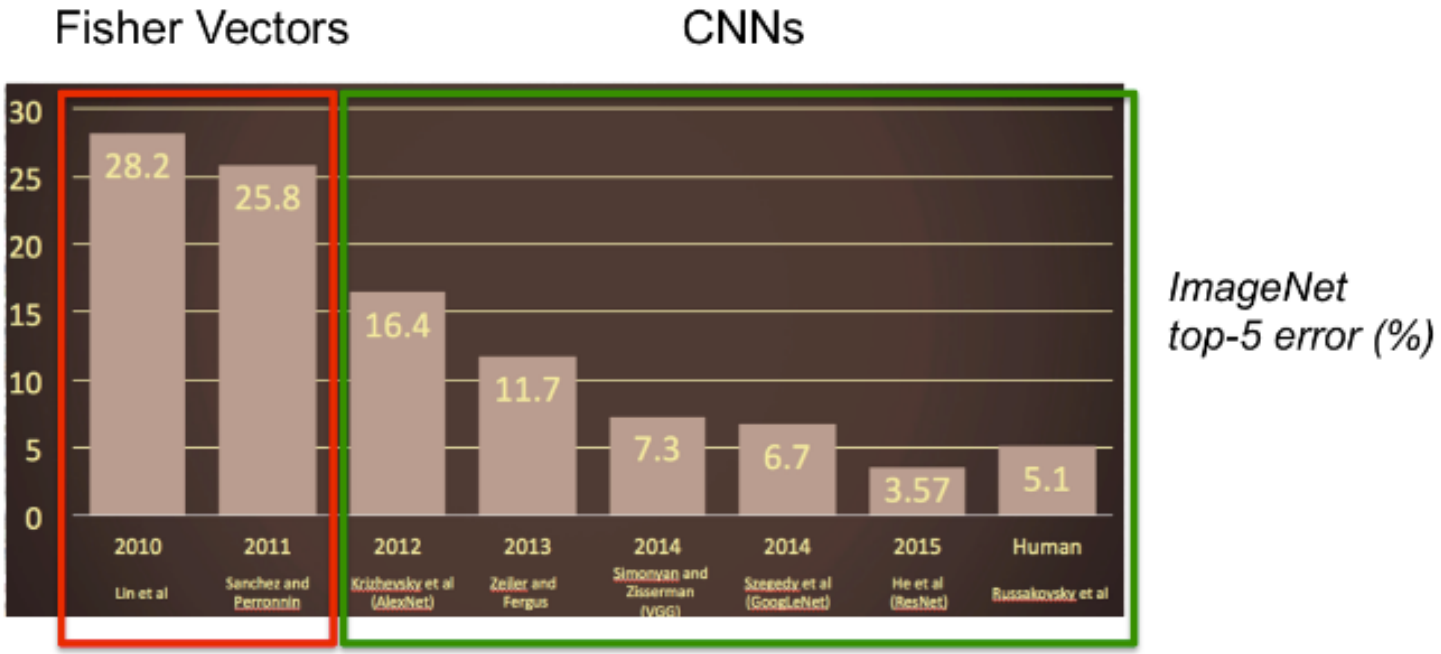
**Arguments**

# THIS IS A CHANGE OF PARADIGM!




Raw data
DATA
FEATURE LEARNING LAYERS
Learning algorithm
prediction

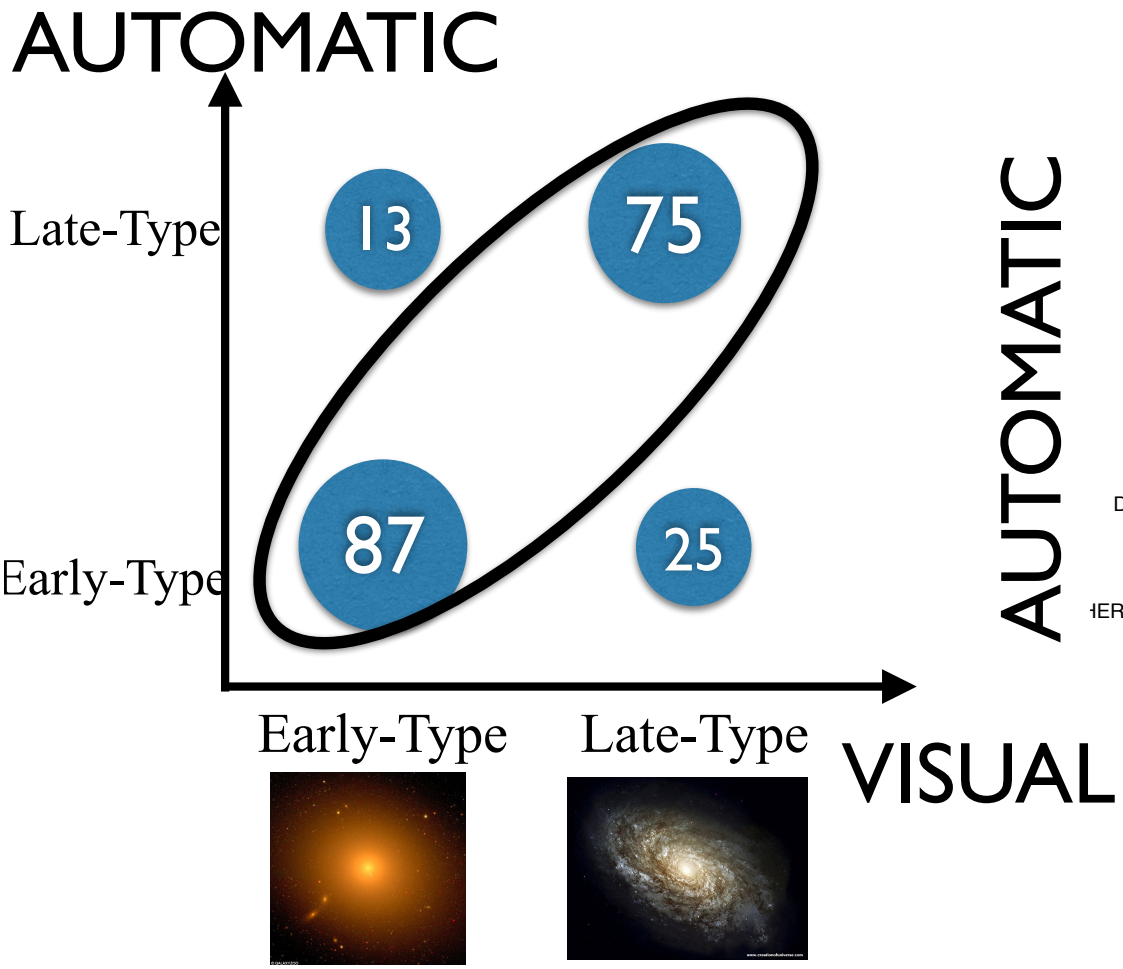
# THIS IS A CHANGE OF PARADIGM!




# ALSO FOR GALAXY MORPHOLOGY

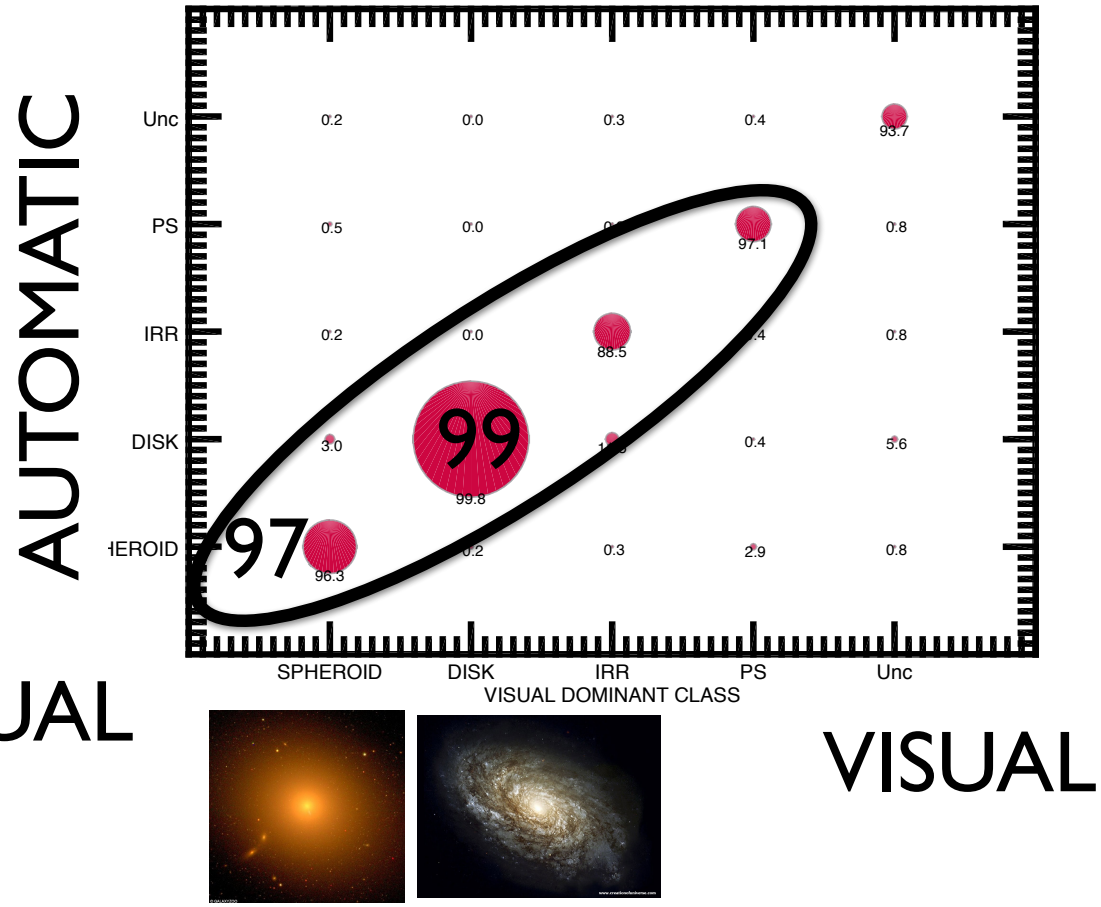
SVMs

[HUERTAS-COMPANY+14]

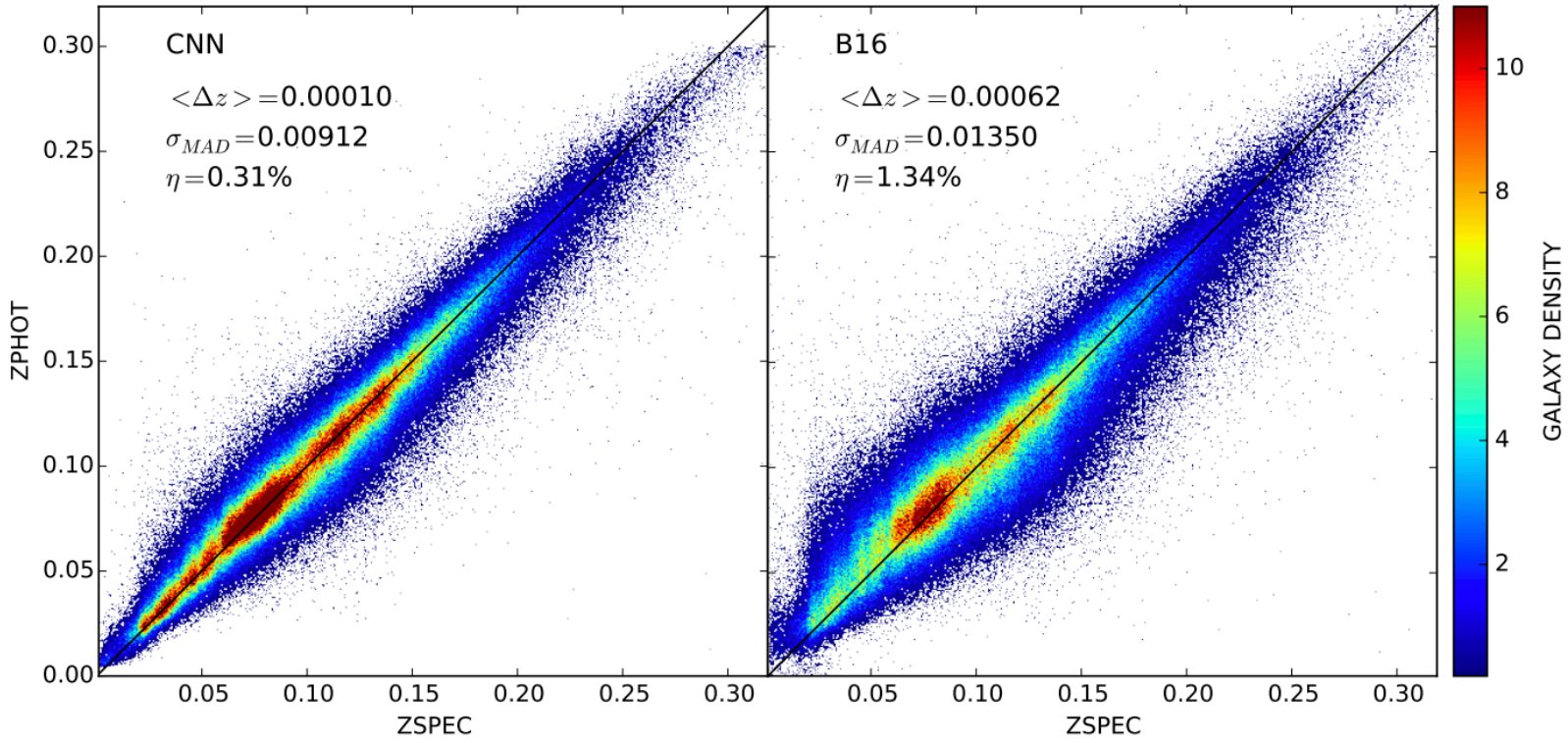



CNNs

[HUERTAS-COMPANY+15b]



# PHOTOMETRIC REDSHIFTS

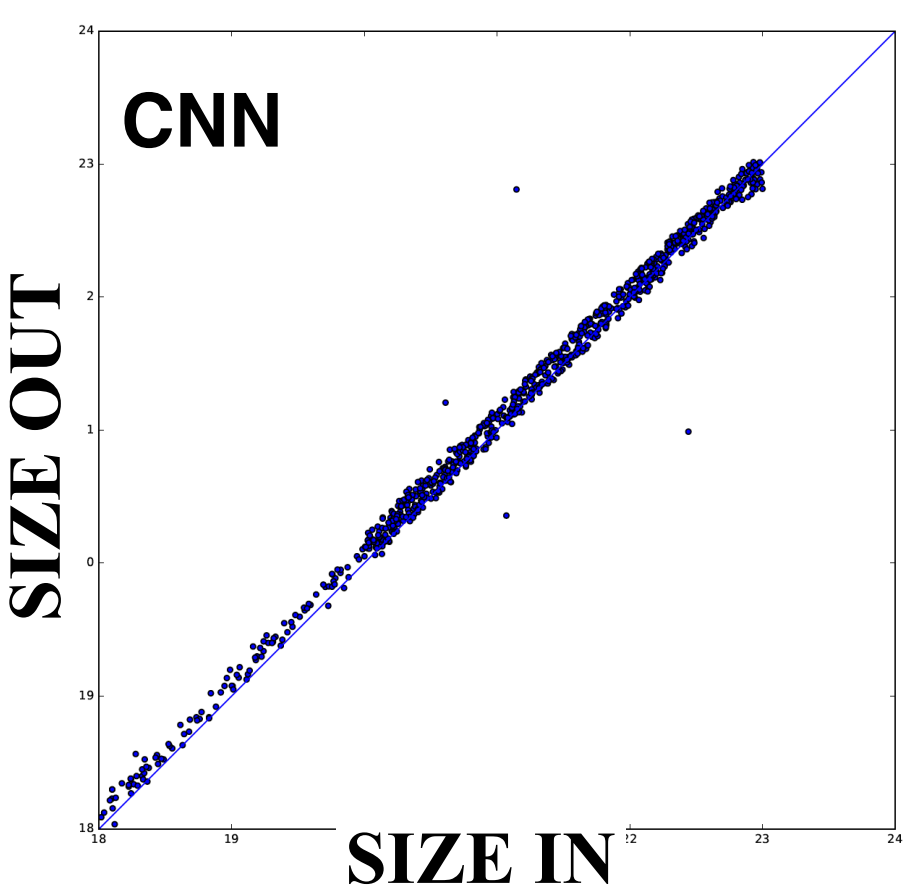


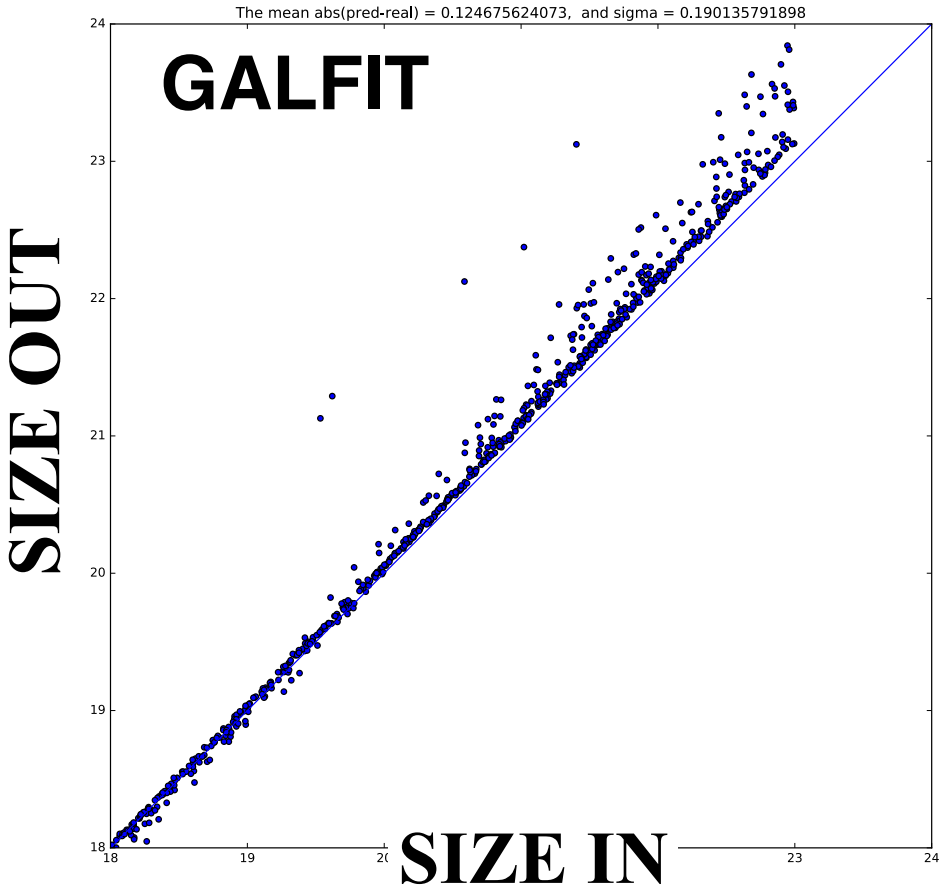

Pasquet+18

AUTOMATICALLY COMBINING MORPHOLOGY AND COLOR FOR PHOTOZ ESTIMATION

# ESTIMATING SIZES OF GALAXIES FROM IMAGES (MODEL FITTING)

Tuccillo+18




VERY SIMILAR RESULTS ON THE SAME SIMULATIONS, BUT CNNs are several orders of magnitude faster [**3.5 hrs vs. <1 sec for ~1000 objects**]

# THE PRICE TO PAY?

1. LARGE NUMBER OF PARAMETERS IMPLIES LARGE DATASETS TO TRAIN

**2. LOOSE EVEN MORE DEGREE OF CONTROL OF WHAT THE ALGORITHM IS DOING SINCE THE FEATURE EXTRACTION PROCESS BECOMES UNSUPERVISED**

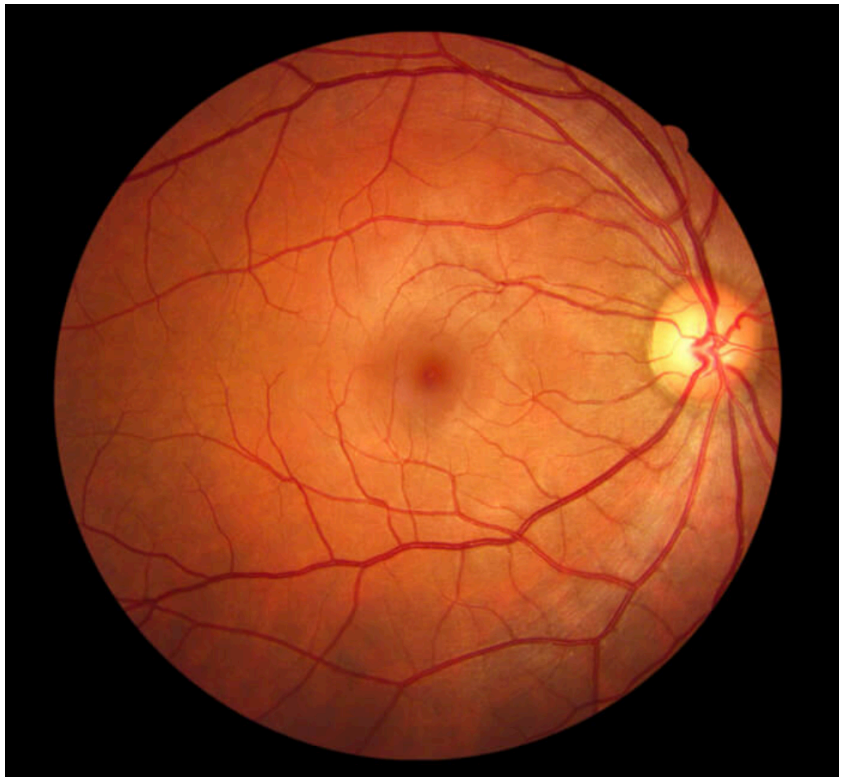


**IMAGE OF THE BACK OF THE EYE**

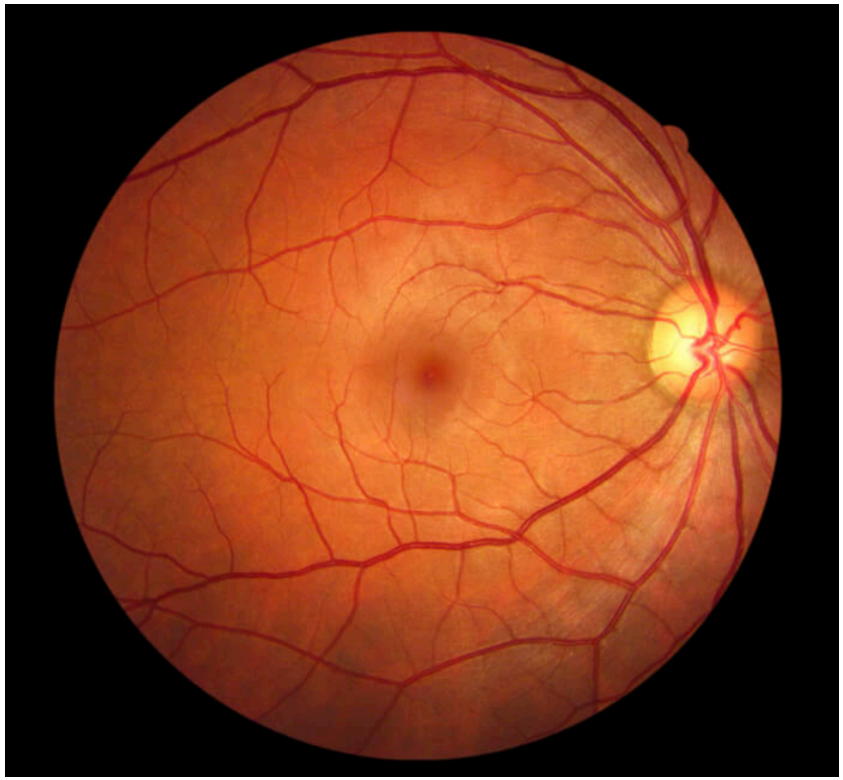
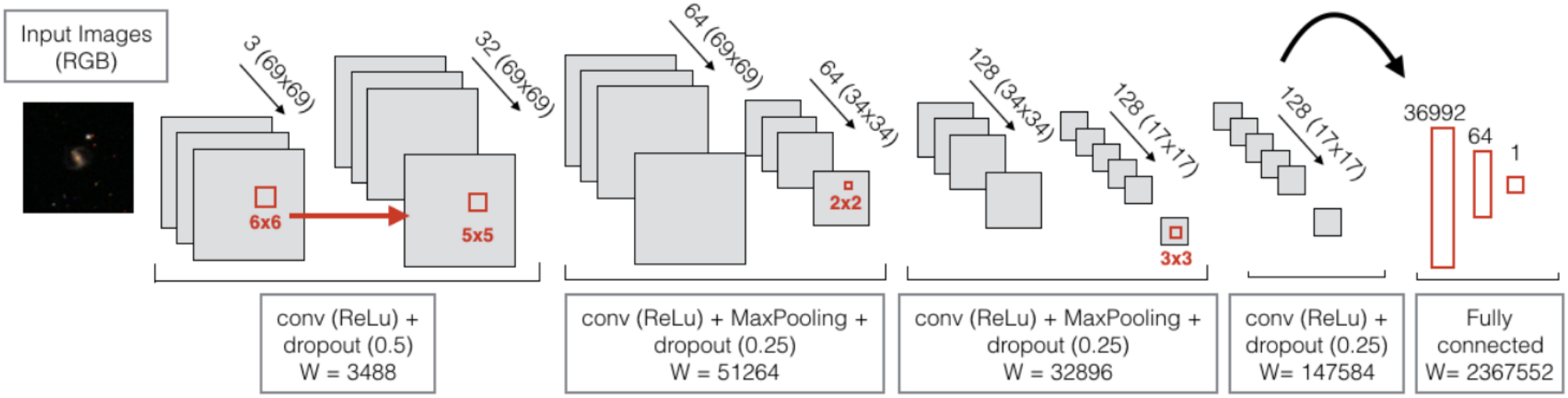


**IMAGE OF THE BACK OF THE EYE**

**DEEP LEARNING CAN IDENTIFY THE PATIENT'S GENDER WITH 95% ACCURACY**

# PART IV: BEYOND CLASSIFICATION: IMAGE2IMAGE NETWORKS

# UP TO NOW CNNs MAP IMAGES (SIGNALS) INTO FLOATS




Dominguez-Sanchez+18

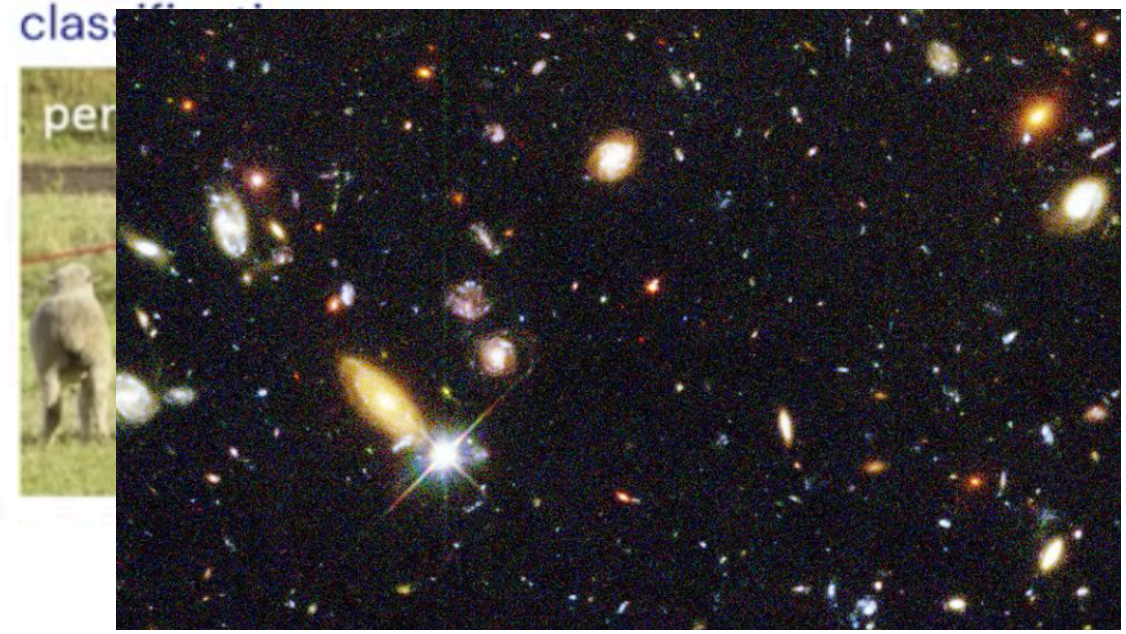
# Classification has its limits ....

HOW DO I CLASSIFY THIS IMAGE?

# Classification has its limits ....


**HOW DO I CLASSIFY THIS IMAGE?**

# Classification has its limits ....

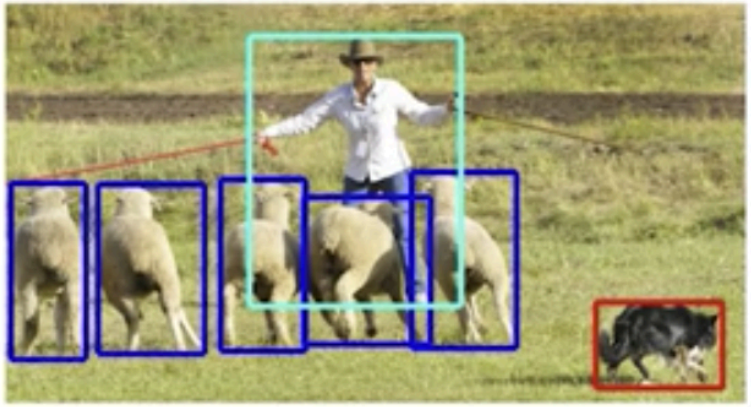


HOW DO I CLASSIFY THIS IMAGE?

# Going beyond classificaiton: increasing complexity

classification


object detection

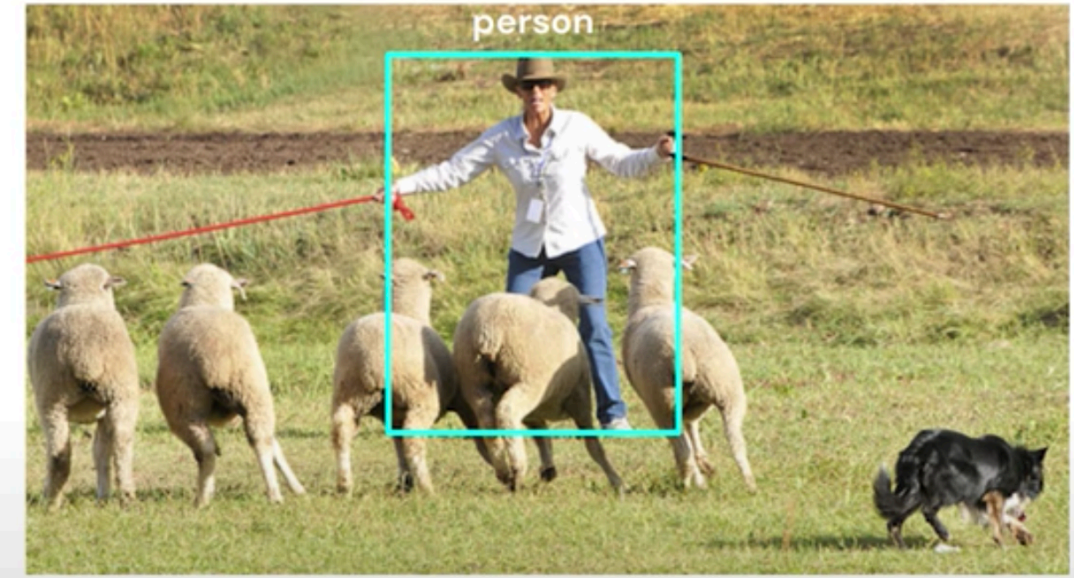


semantic segmentation

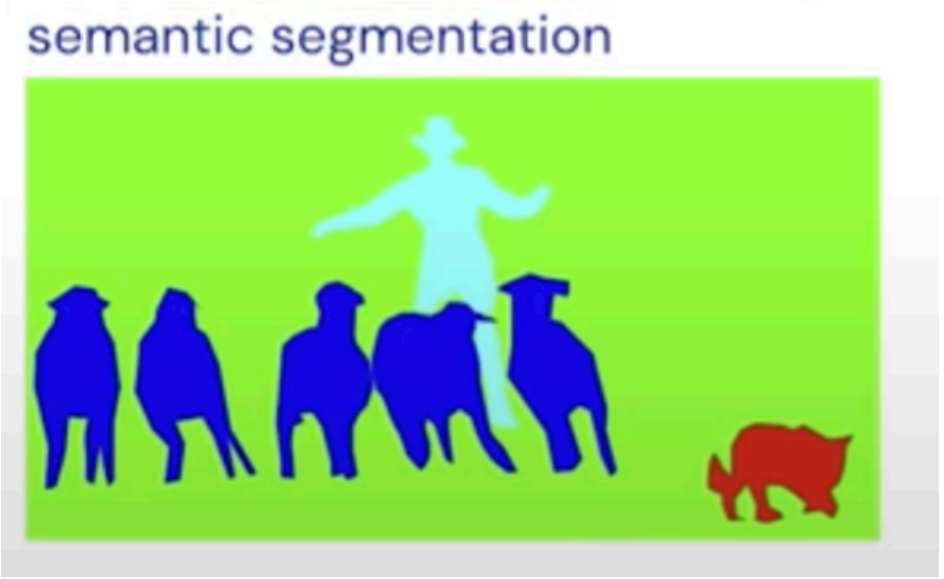
instance segmentation

@deepmind (V. Patraucean)

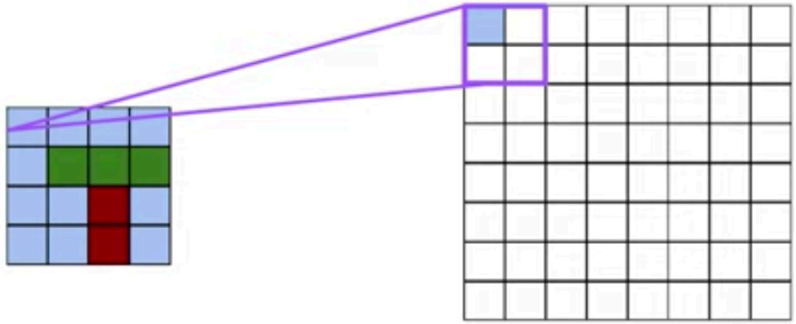
# LET'S GO A STEP FURTHER INTO SEMANTIC SEGMENTATION



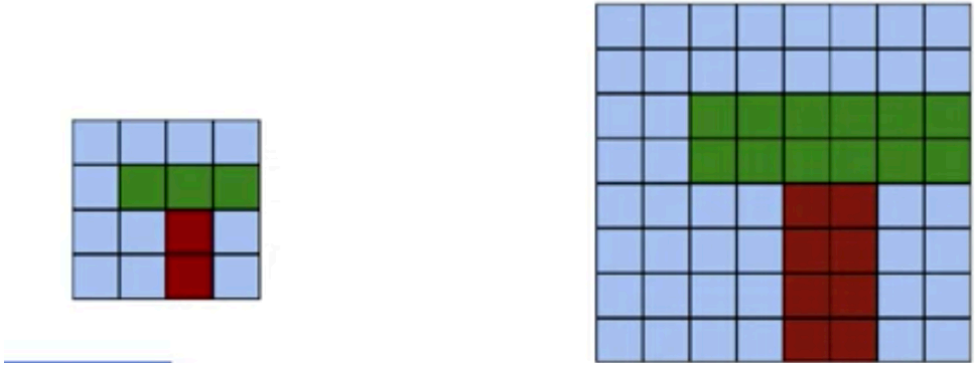

BOUNDING BOXES ARE NOT ALWAYS GOOD REPRESENTATIONS



# UNPOOLING OPERATION (INVERSE OF POOLING)



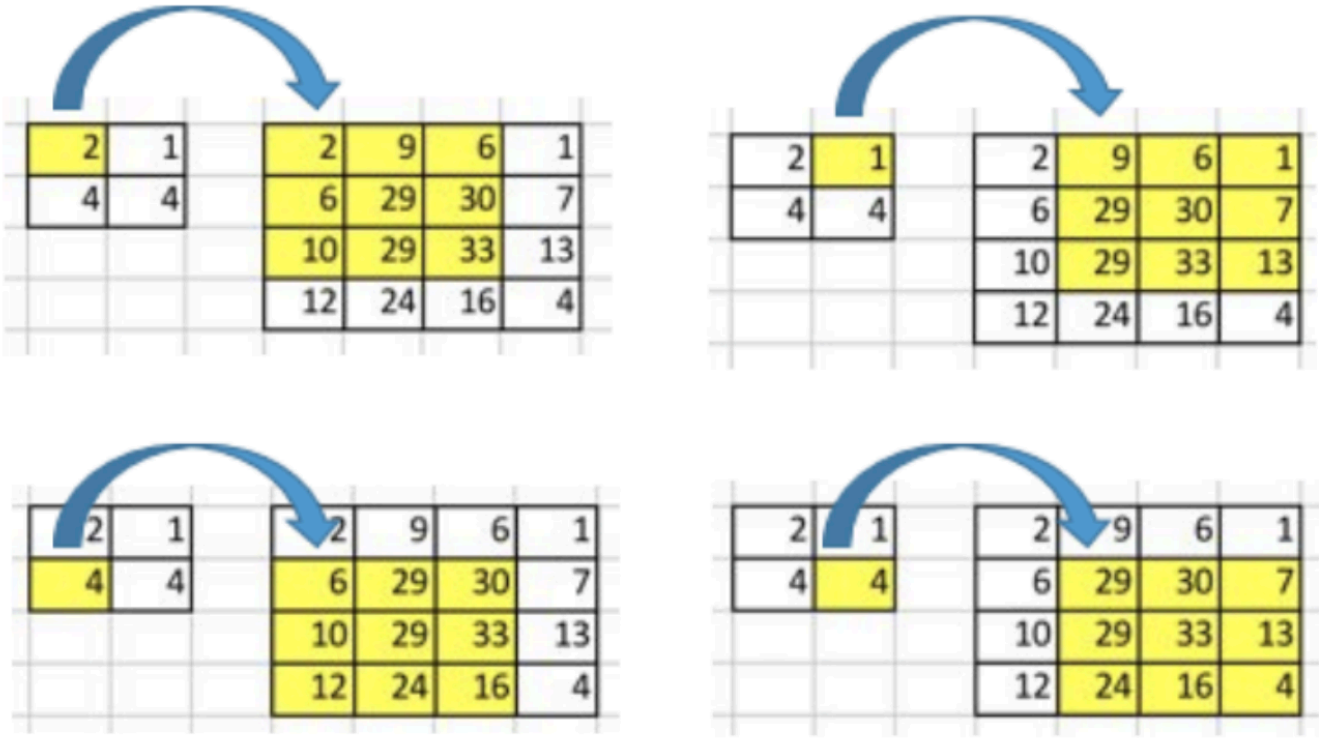
COPY PIXELS IN A GIVEN WINDOW



GENERATES LARGER IMAGES FROM SMALLER ONES

# TRANSPOSED CONVOLUTION

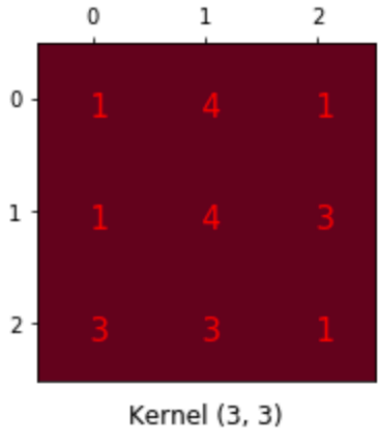
## ALLOWS TO INCREASE THE SIZE




Going Backward of Convolution

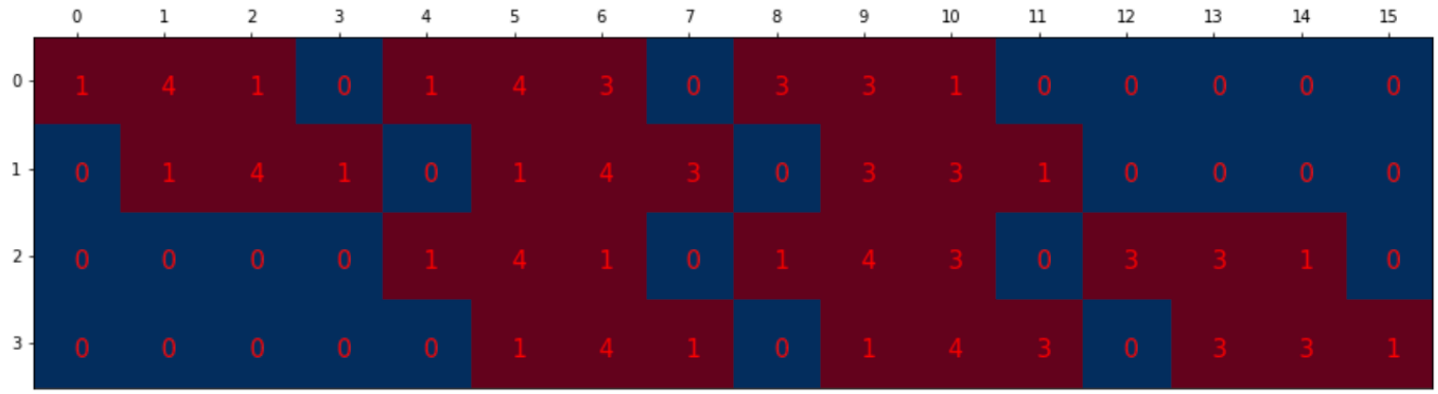
## EXAMPLE TAKEN FROM HERE

# CONVOLUTION MATRIX



Kernel (3, 3)

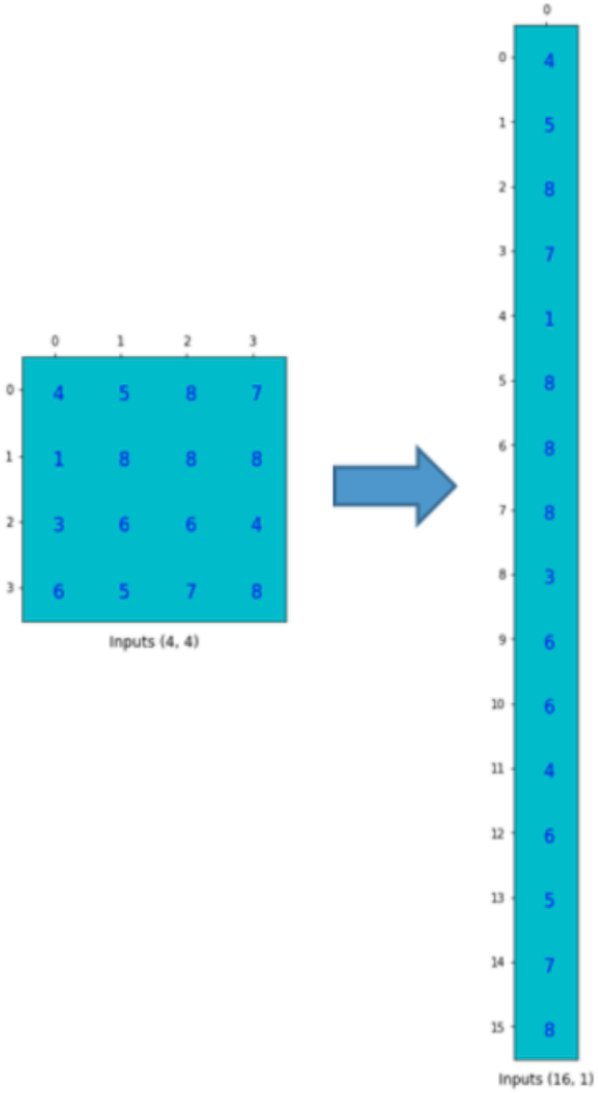
## THE KERNEL CAN BE ARRANGED IN FORM OF A MATRIX:



Convolution Matrix (4, 16)

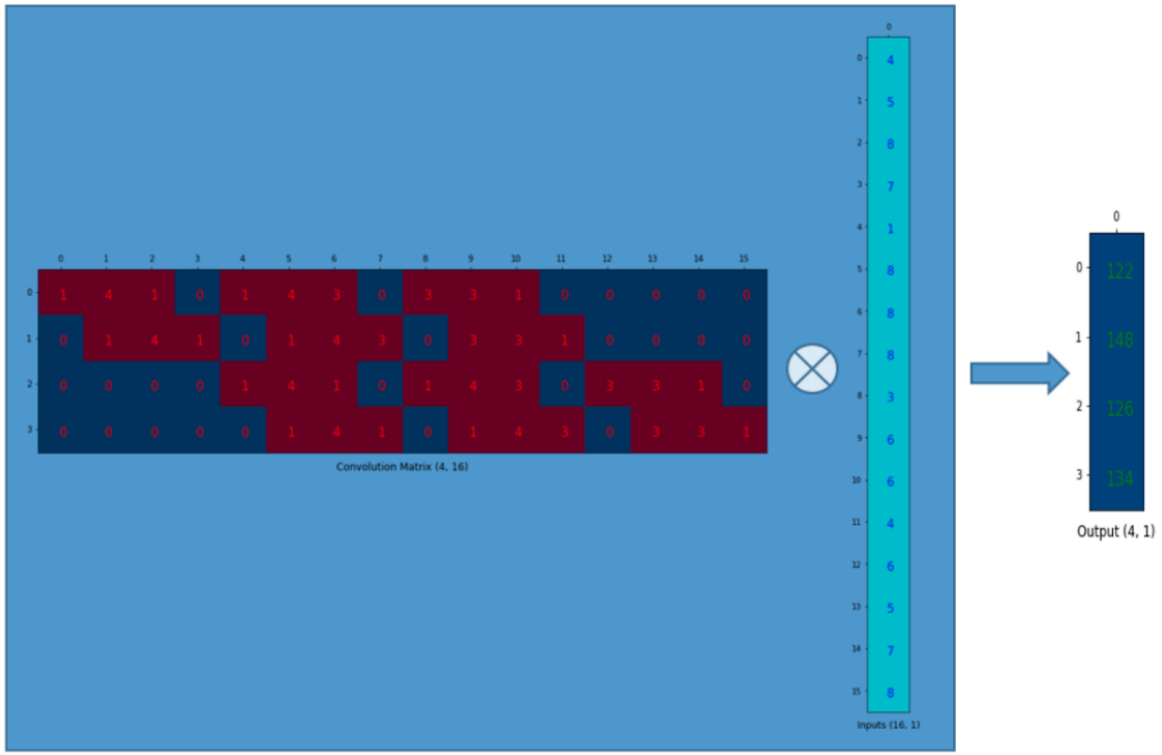
## EXAMPLE TAKEN FROM HERE

# THE INPUT IS FLATTENED INTO A COLUMN VECTOR



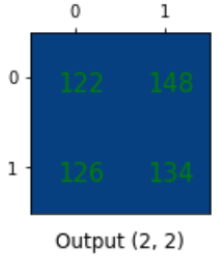

**EXAMPLE TAKEN FROM HERE**

# THE CONVOLUTION IS TRANSFORMED INTO A PRODUCT OF MATRICES





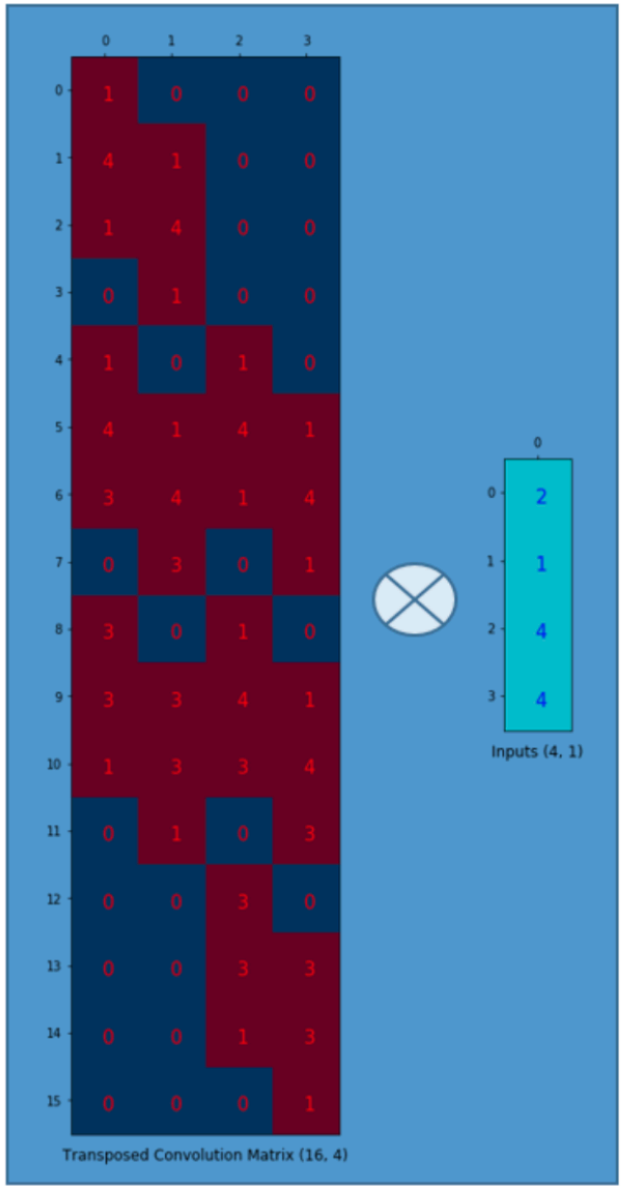





**RESHAPED OUTPUT**

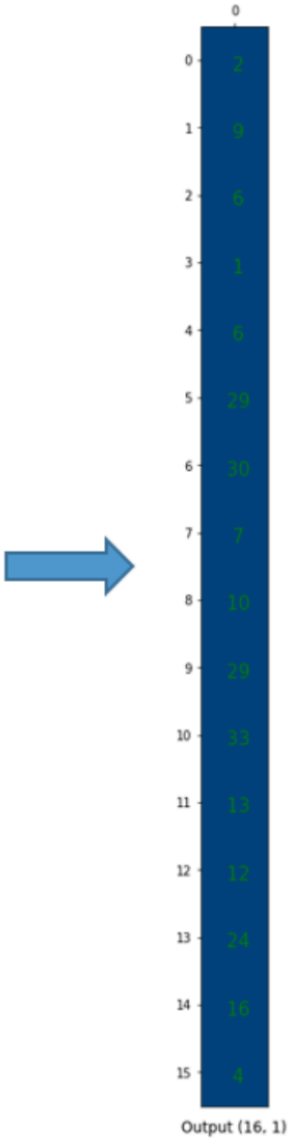
**EXAMPLE TAKEN FROM HERE**

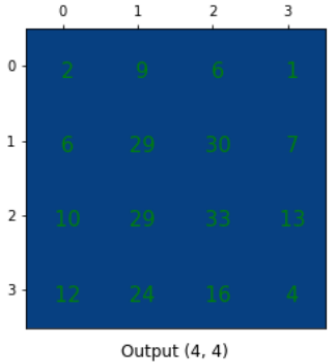
# THE TRANSPOSED CONVOLUTION IS THE INVERSE OPERATION







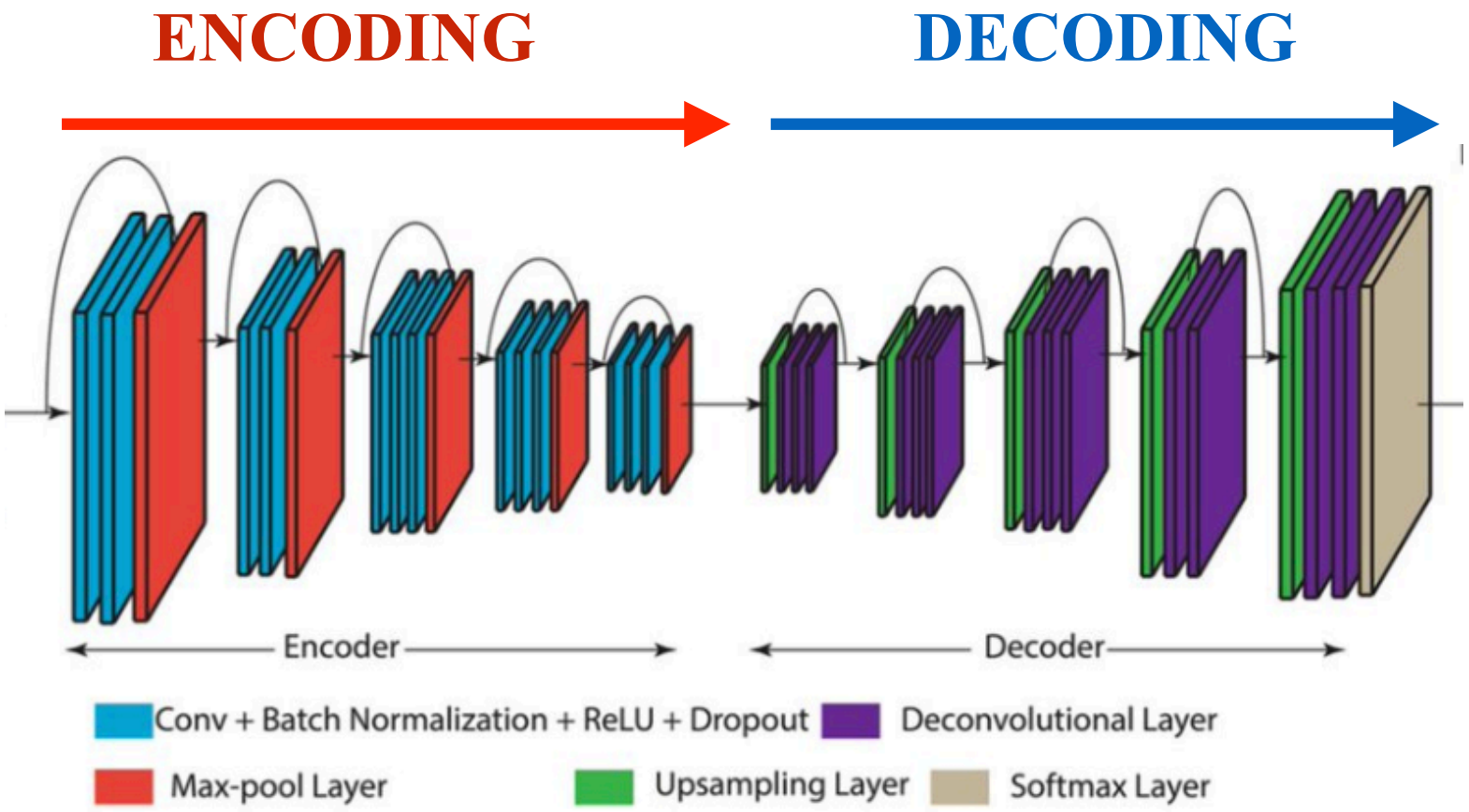



**RESHAPED OUTPUT**

**EXAMPLE TAKEN FROM HERE**

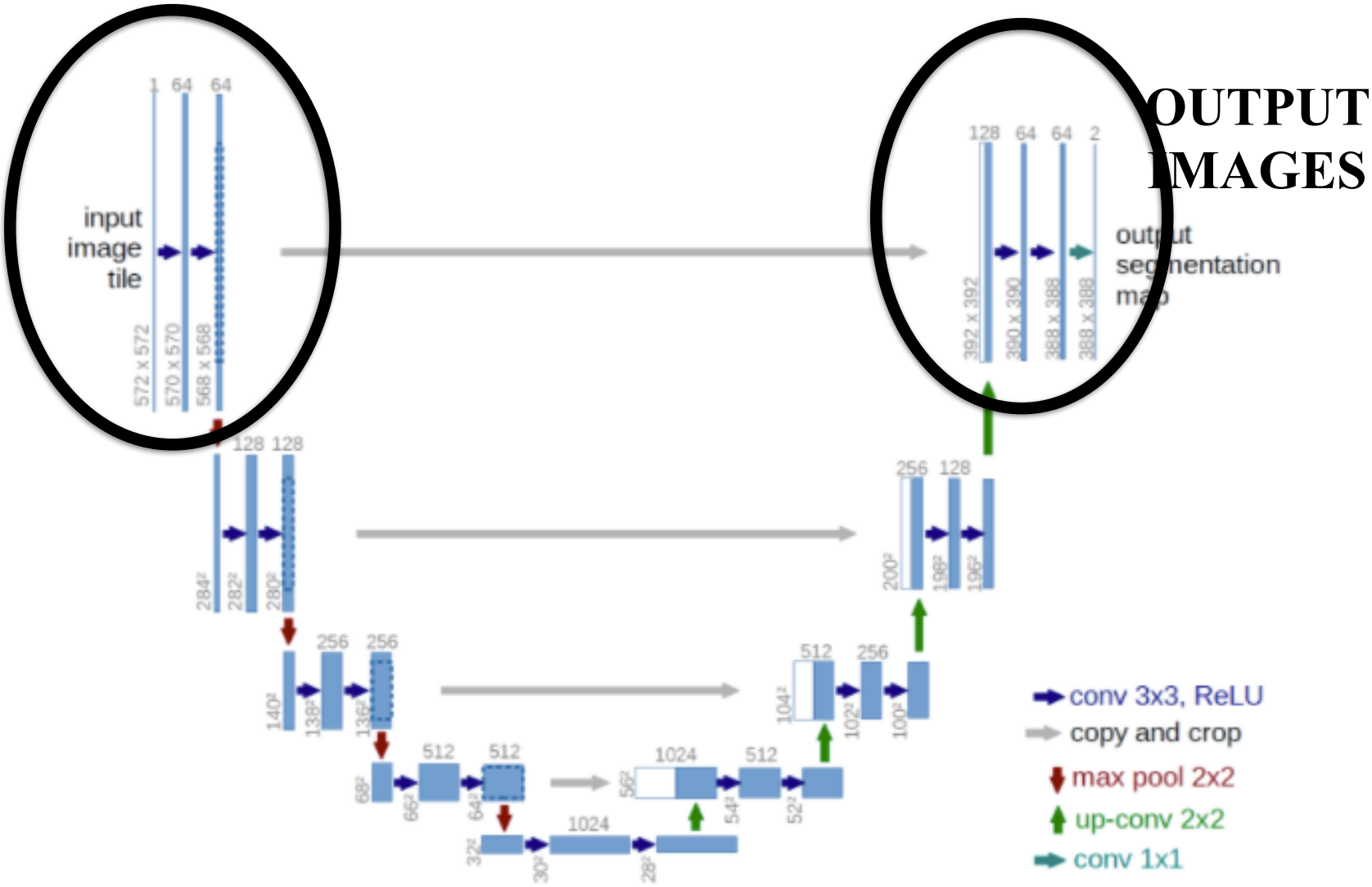
# ENCODER-DECODERS GO FROM IMAGE 2 IMAGE




**WE CALL THIS FULLY CONVOLUTIONAL NEURAL NETWORKS**

# ENCODING-DECODING TO EXTRACT IMAGE FEATURES: U-NET

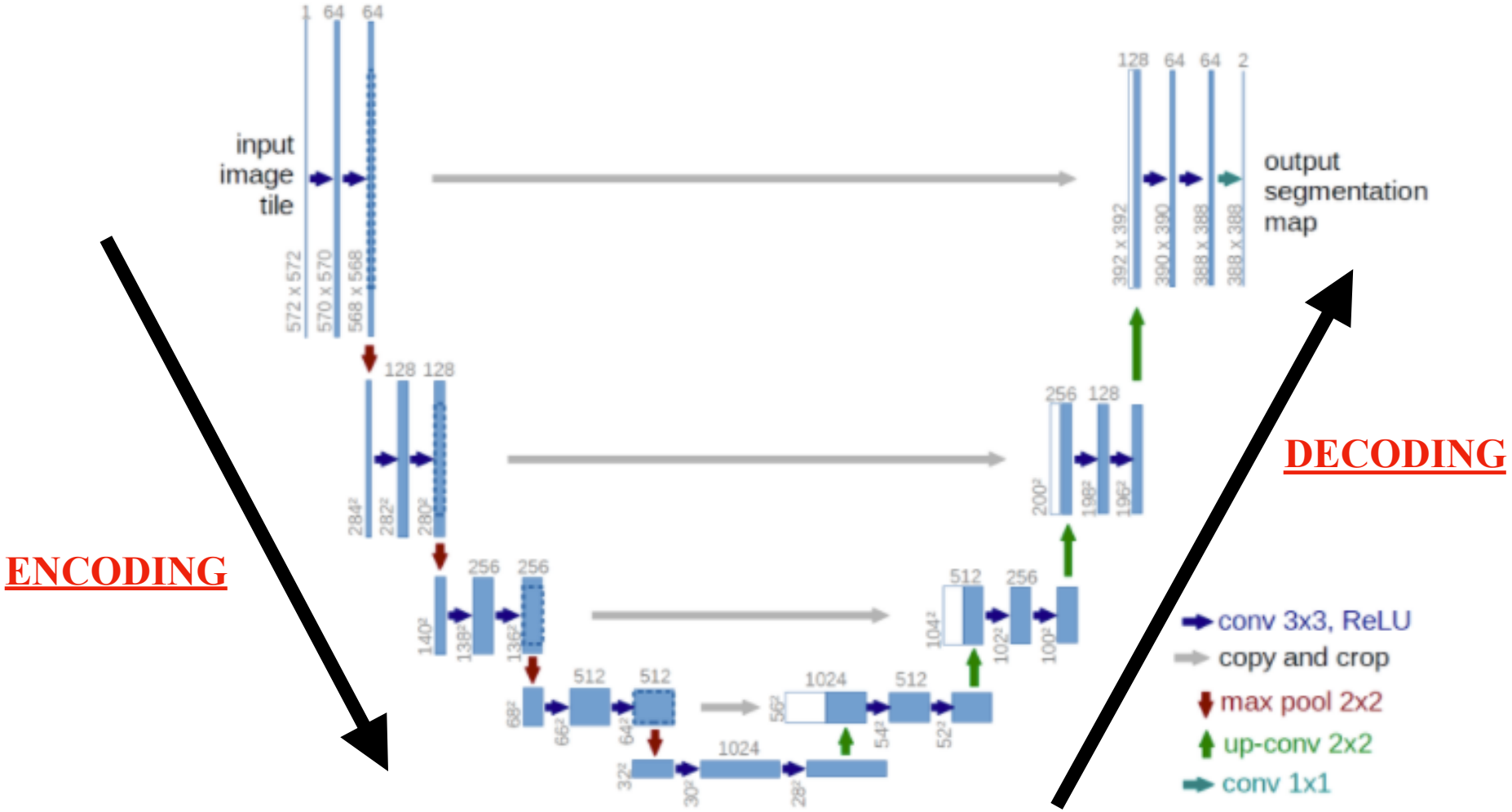
**INPUT IMAGES**



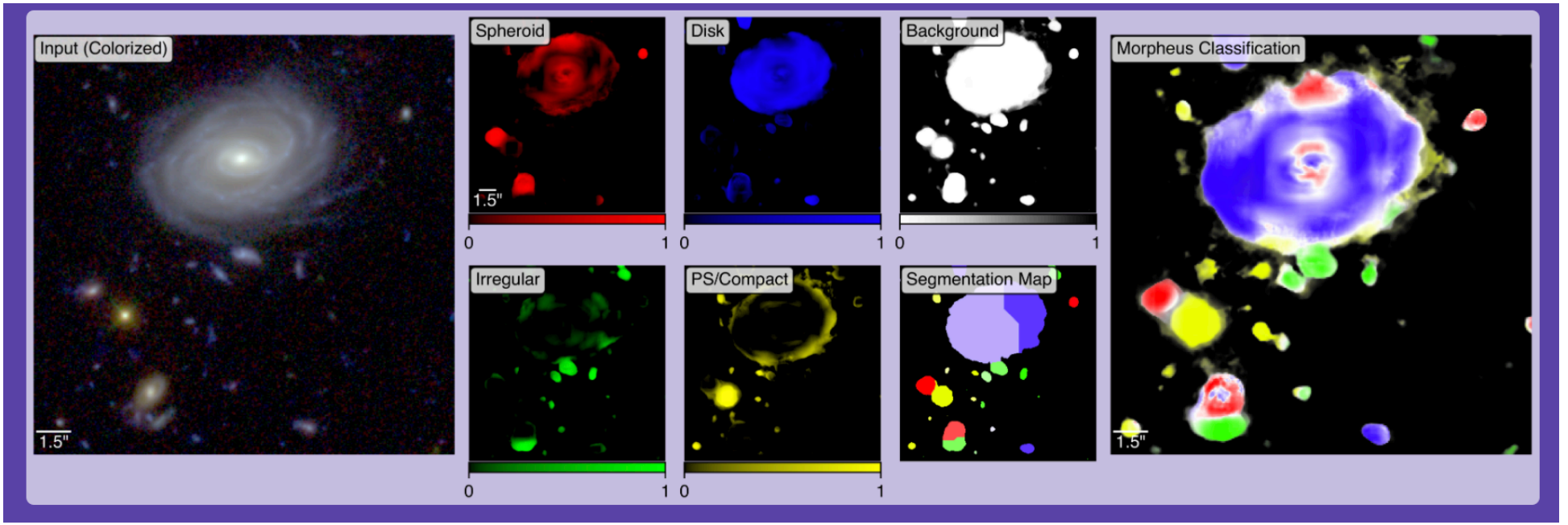

**OUTPUT IMAGES**

Ronnerberger+15

# ENCODING-DECODING TO EXTRACT IMAGE FEATURES: THE U-NET



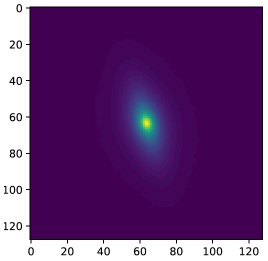

Ronnerberger+15

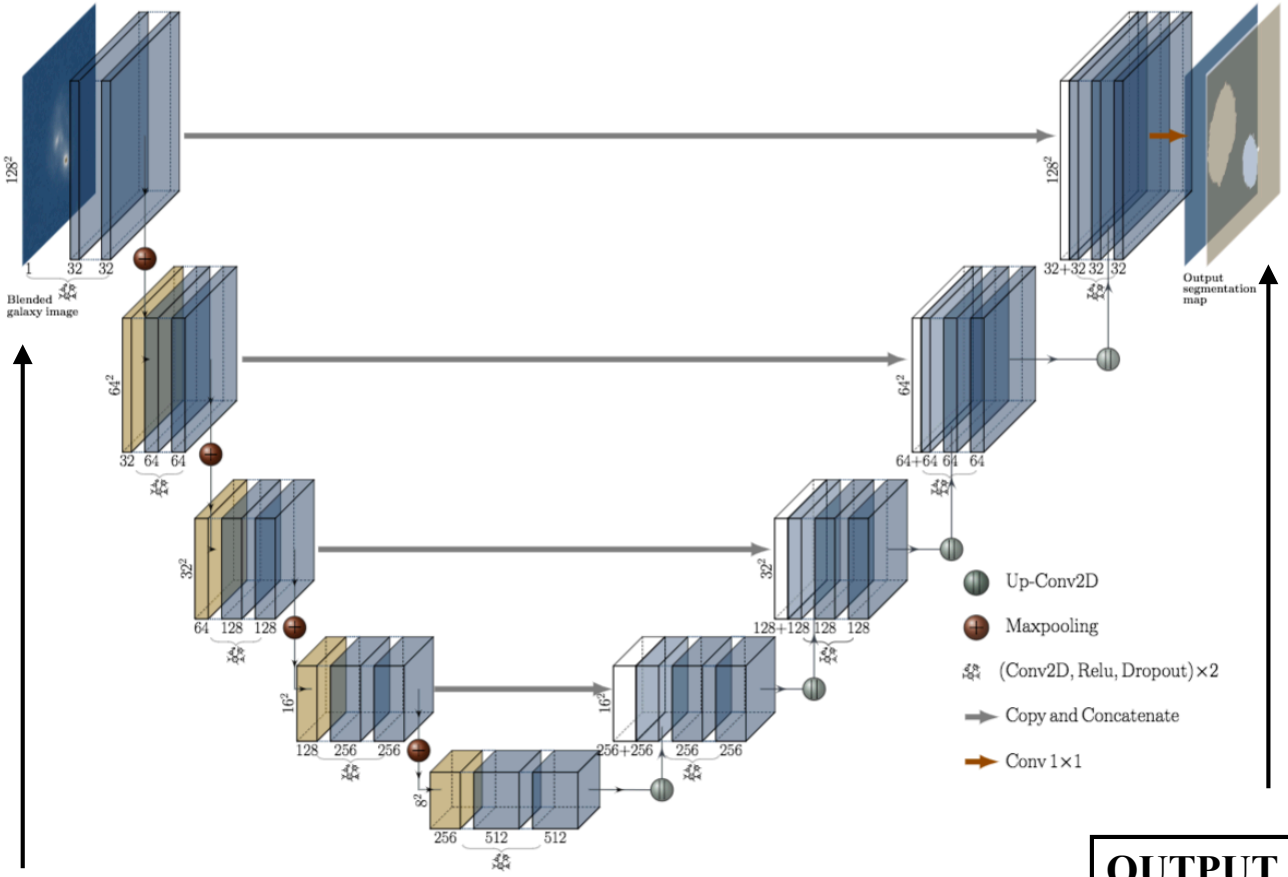



https://www.youtube.com/watch?v=hEL1h_dODkU#action=share

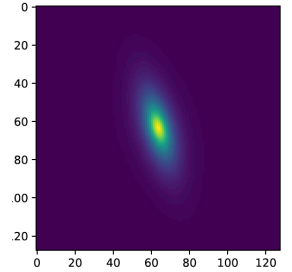
Hausen+20

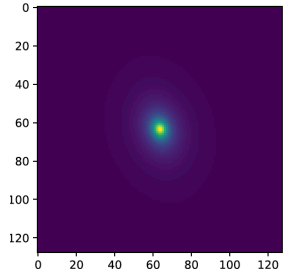
# ALSO REGRESSION…






DISK





BULGE

GALAXY STAMP

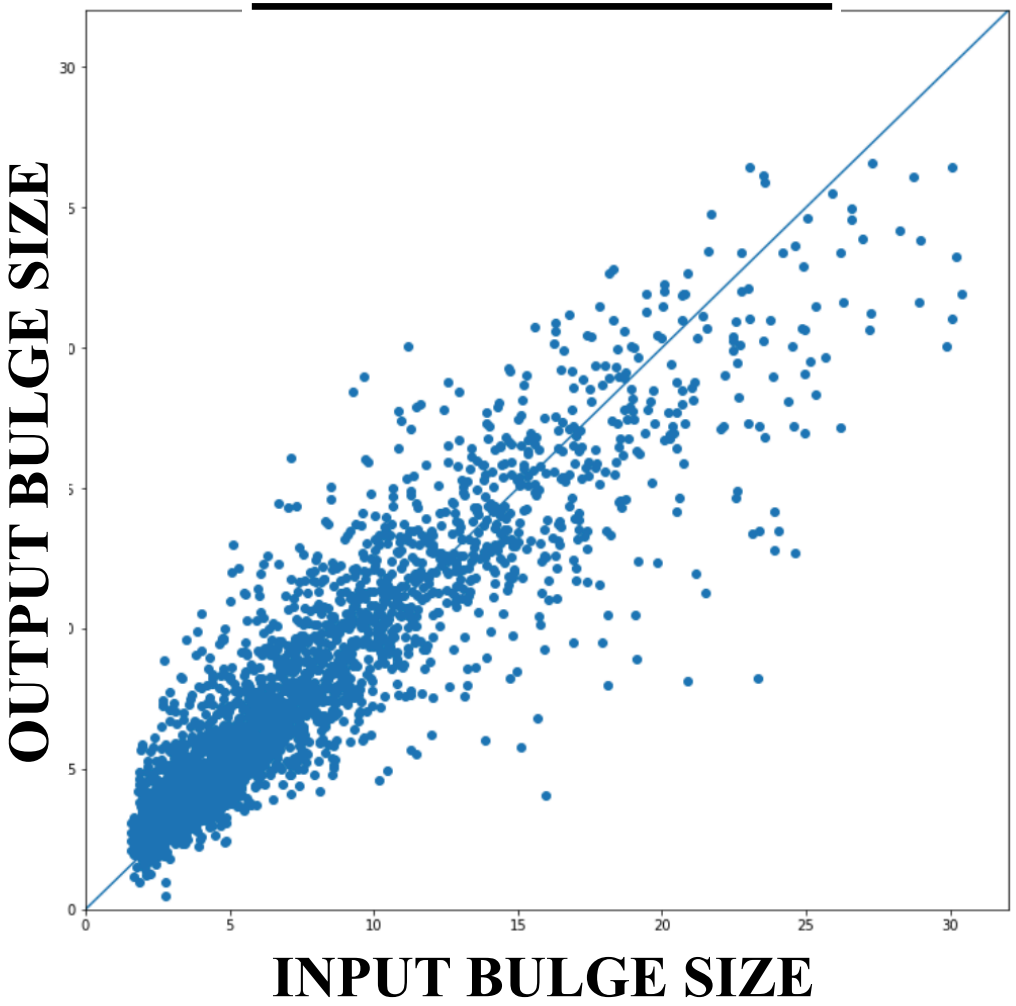
BULGE AND DISK PARAMETERS (SIZE, INCLINATION ETC)

OUTPUT IMAGE OF BULGE AND DISK COMPONENTS

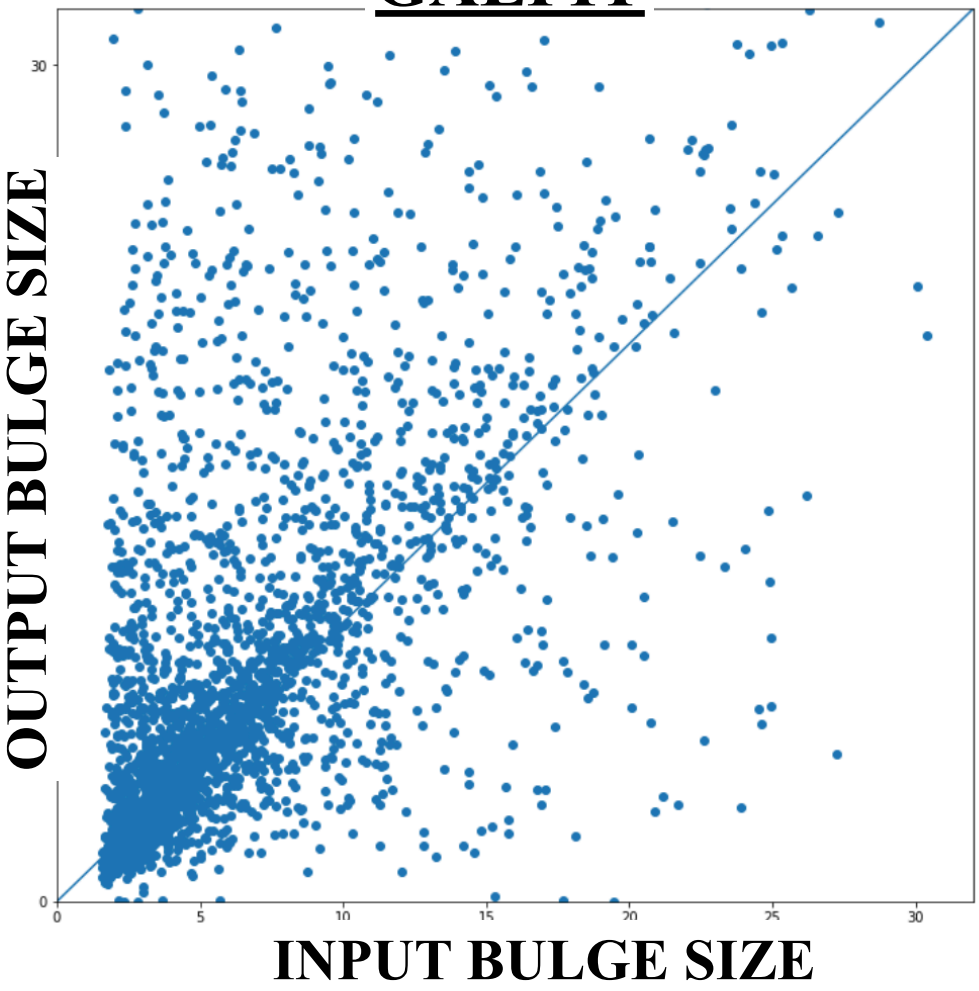
Tuccillo, MHC+18,19

# PRELIMINARY!!

## DEEP LEARNING




## GALFIT




Tuccillo, MHC+18,19