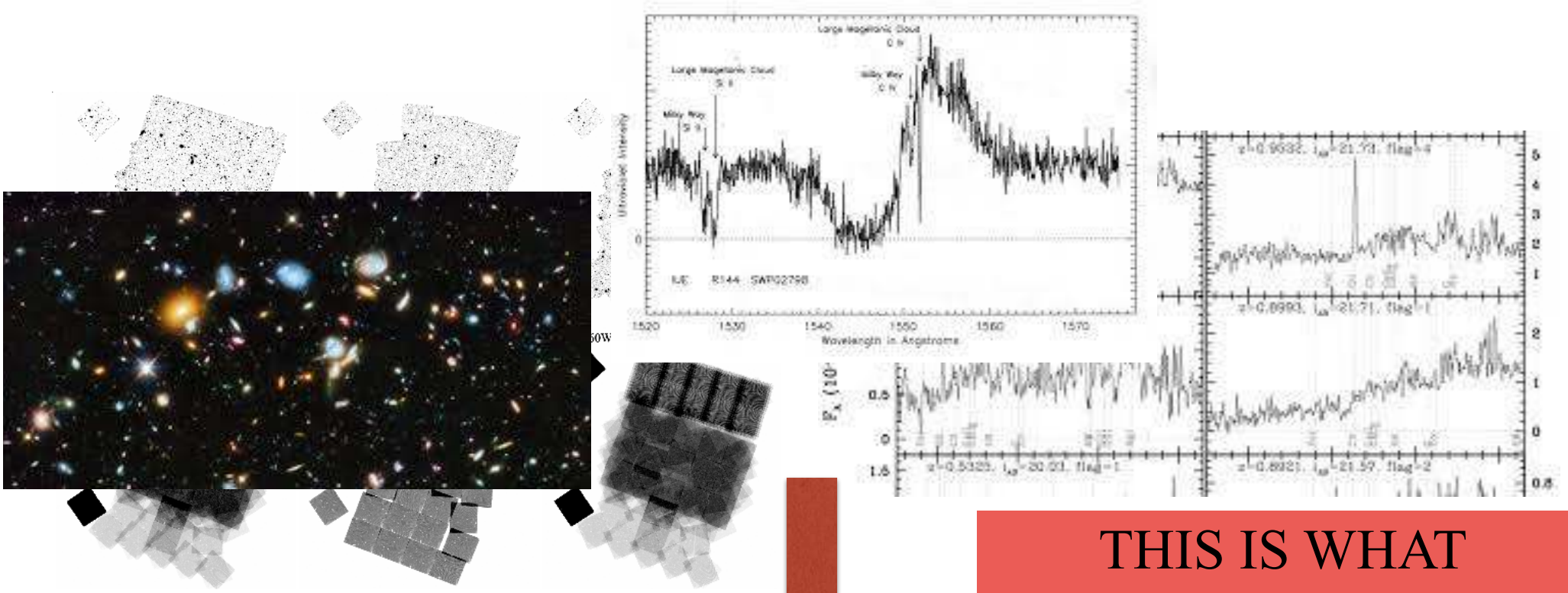
NEURAL NETWORKS FOR COMPUTER VISION

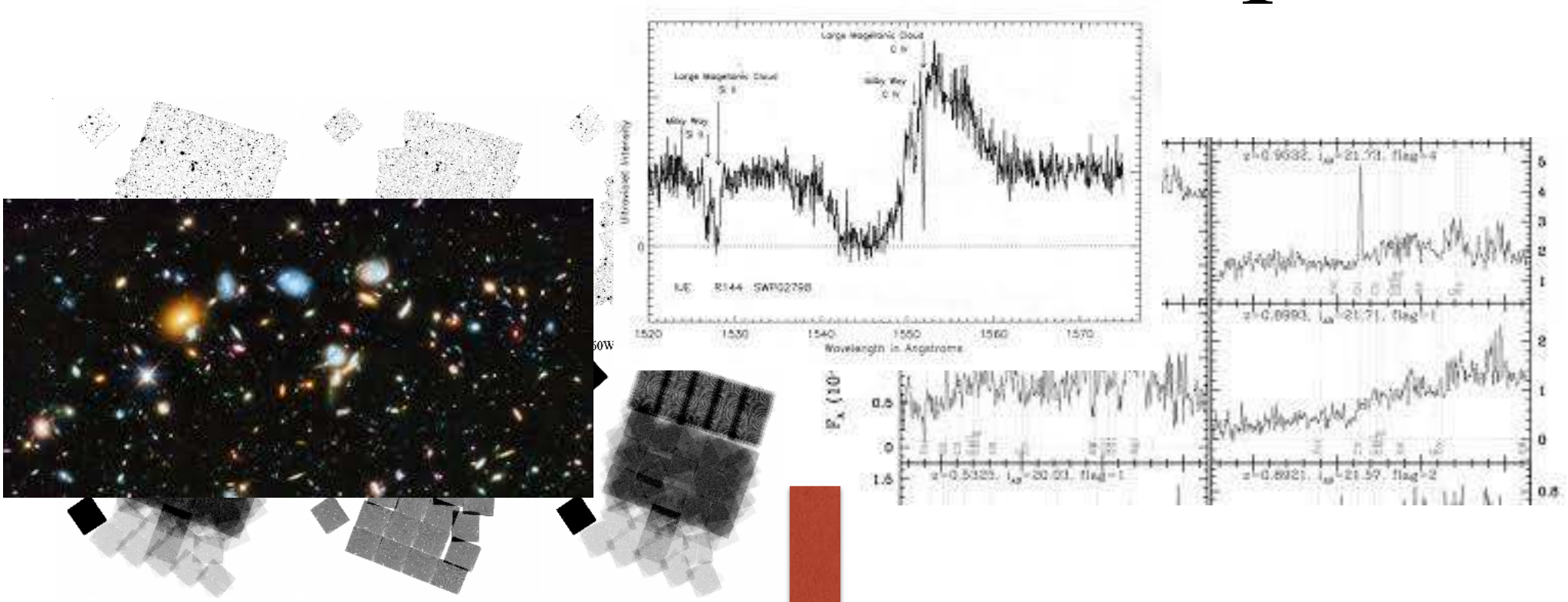
CAN WE GO DEEP NOW?

What do we put as input?



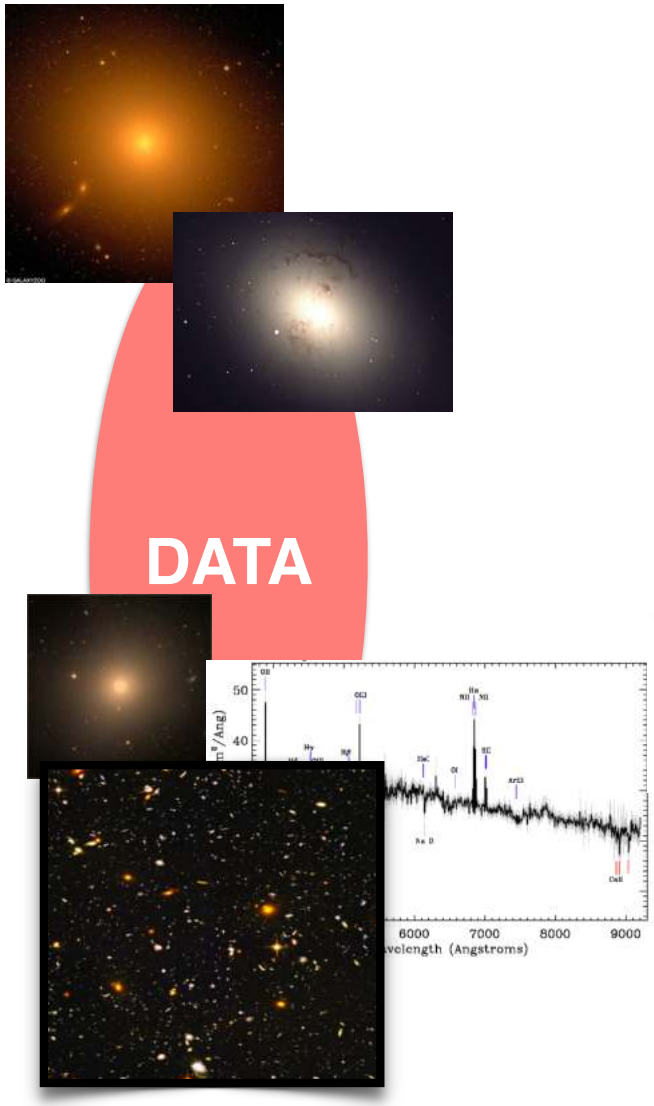
THIS IS WHAT MACHINES SEE

What do we put as input?



PRE-PROCESS DATA TO EXTRACT MEANINGFUL INFORMATION

THIS IS GENERALLY CALLED **FEATURE EXTRACTION**



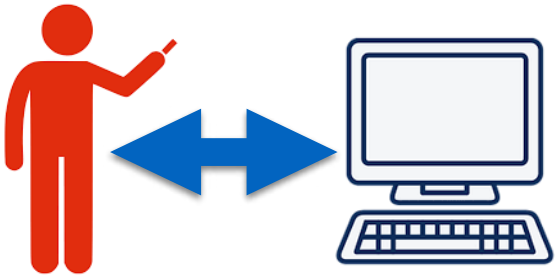
Spiral!

Emission line!

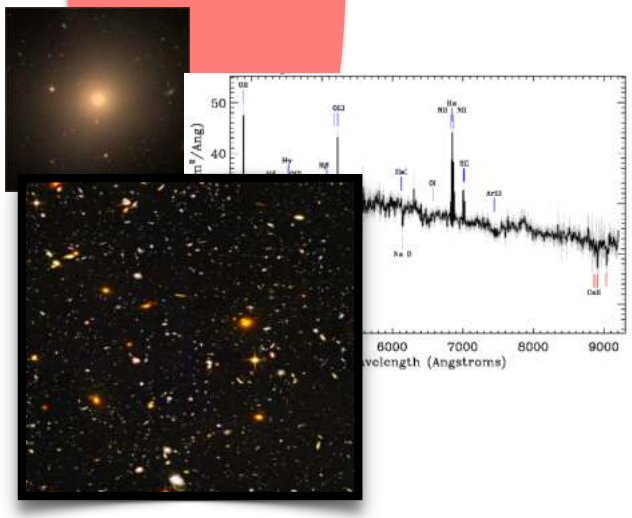
Merger!

Clump!

AGN!



DATA



Spiral!

Emission line!

Merger!

Clump!

AGN!

$$f_{\vec{W}}(\vec{x}) = \vec{y}$$

LABEL

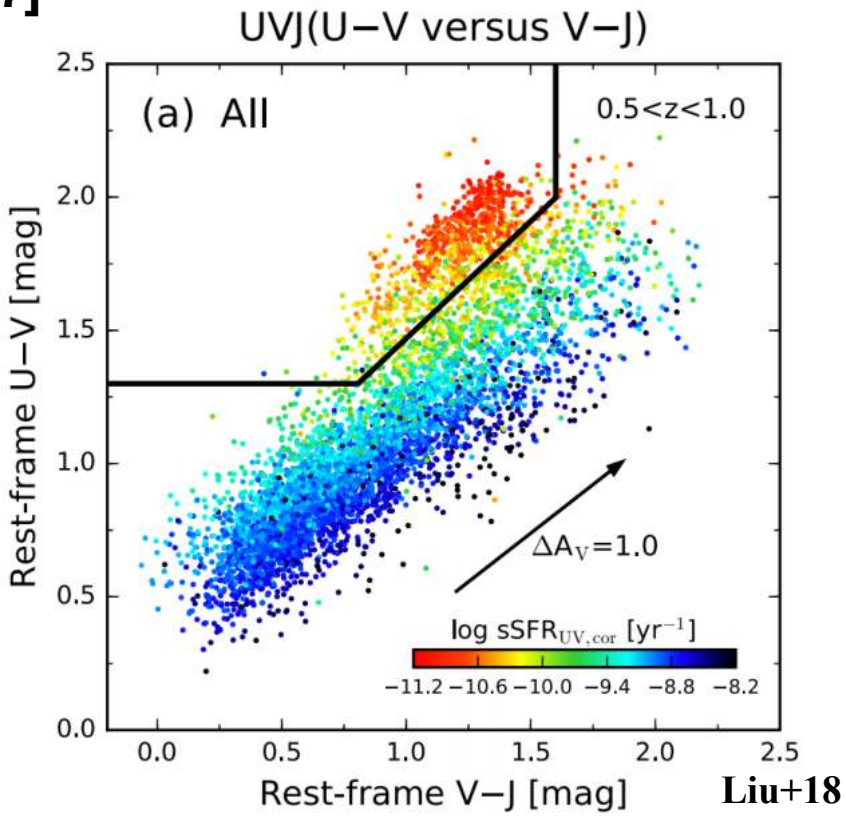
Q(0) , SF(1)

NETWORK FUNCTION

(U-V, V-J) FEATURES

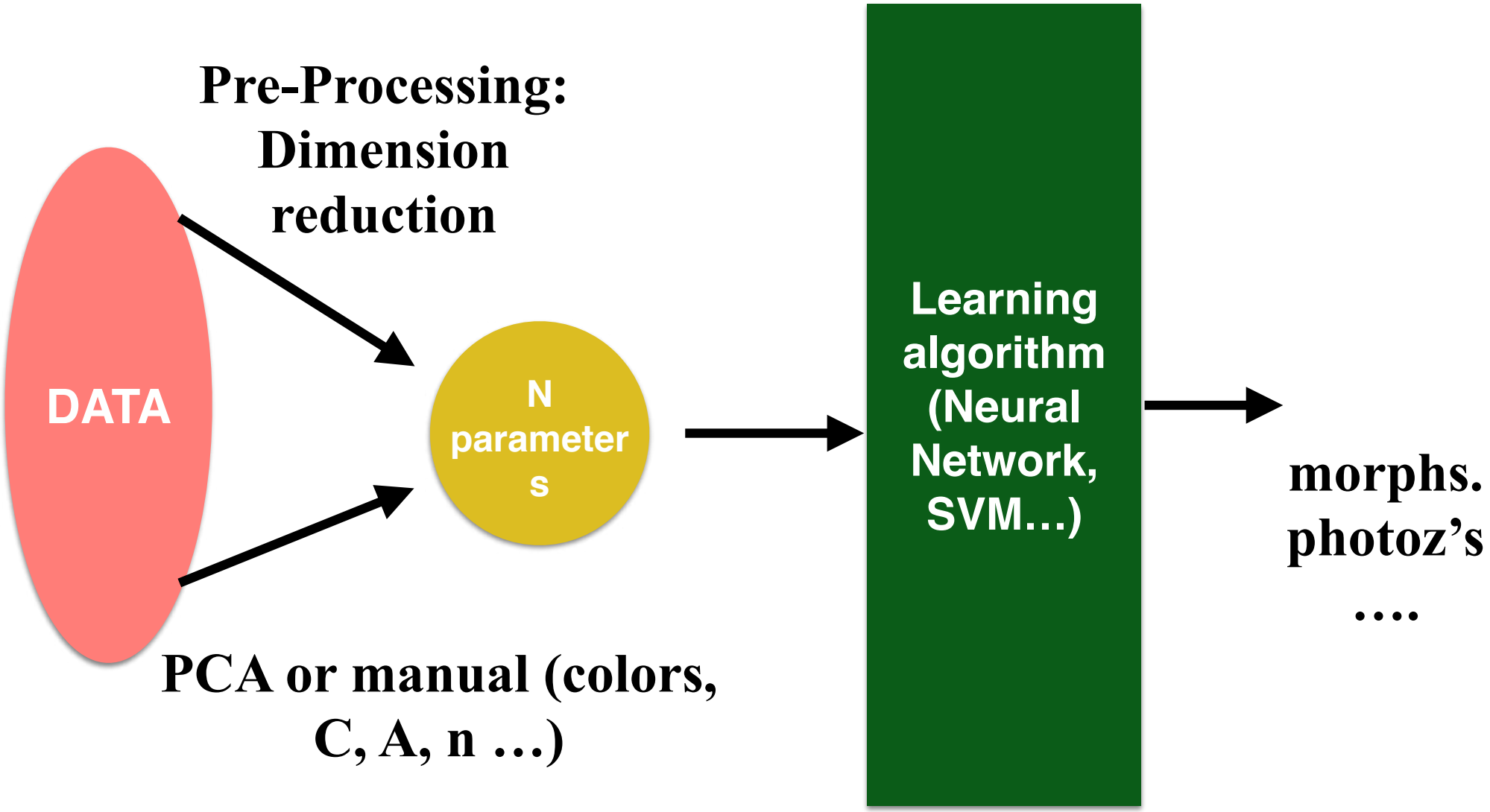
sgn[(u-v)-0.8*(v-j)-0.7]

WEIGHTS

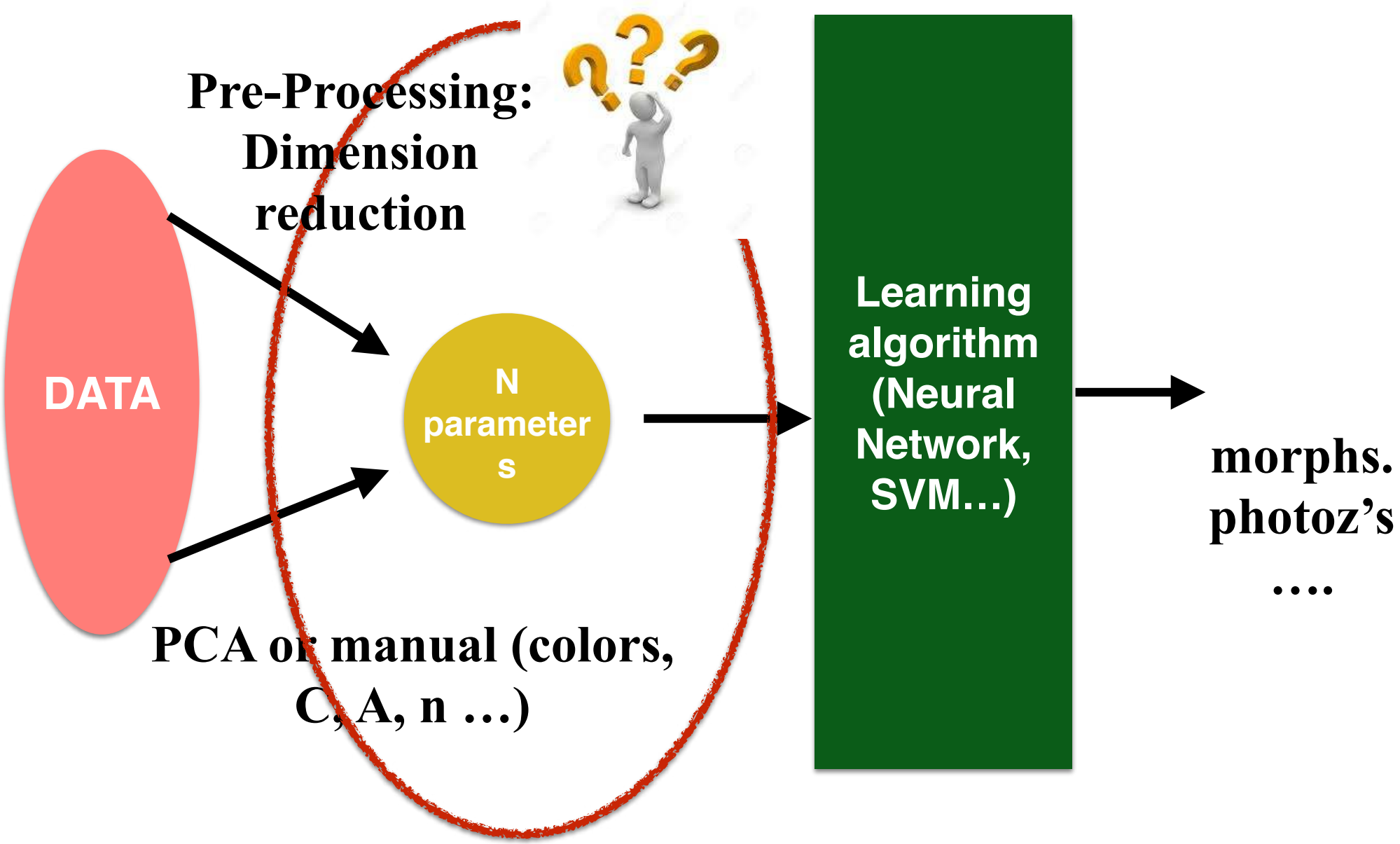


Liu+18

THE "CLASSICAL" APPROACH

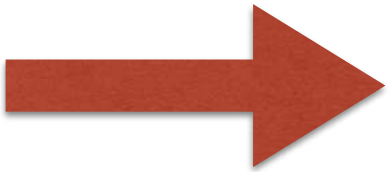


"CLASSICAL" MACHINE LEARNING



In Astronomy

- Colors, Fluxes
- Shape indicators
- Line ratios, spectral features
- Stellar Masses, Velocity Dispersions



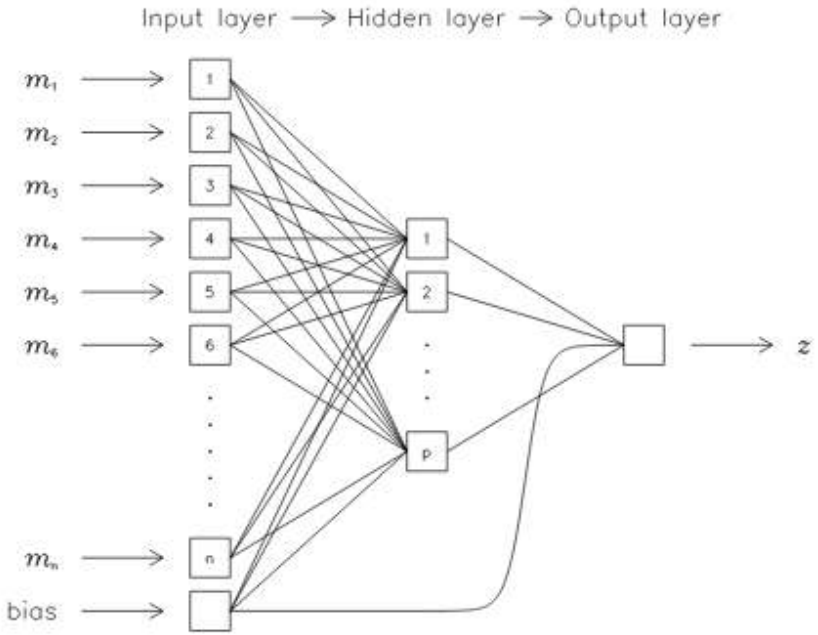
Requires specialized software before feeding the machine learning algorithm

IT IMPLIES A DIMENSIONALITY REDUCTION!

PHOTOMETRIC REDSHIFTS

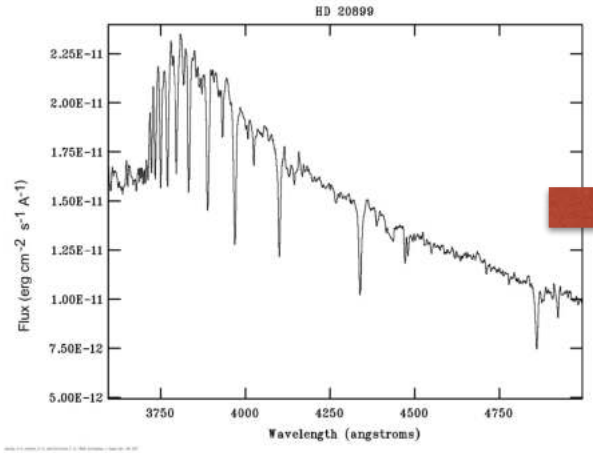
SDSS

g
r
i
z

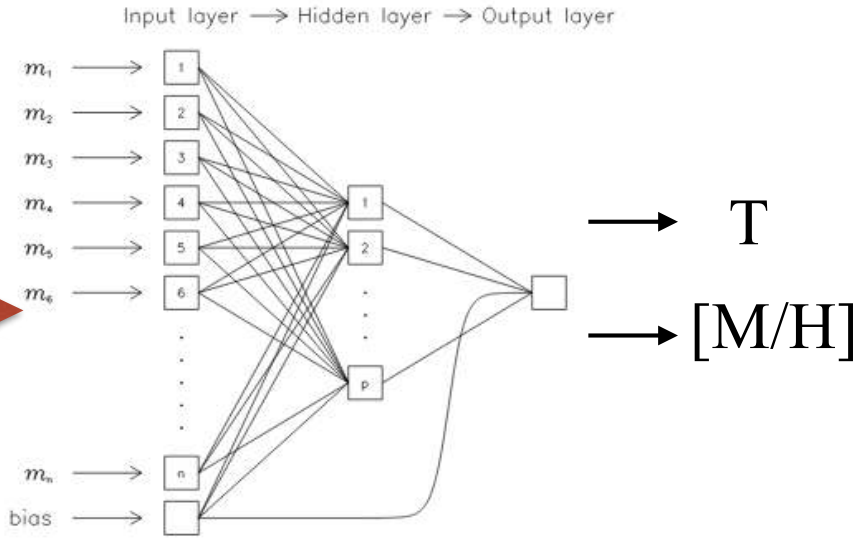


Collister+08

STELLAR PARAMETERS FROM MEDIUM BAND FILTERS



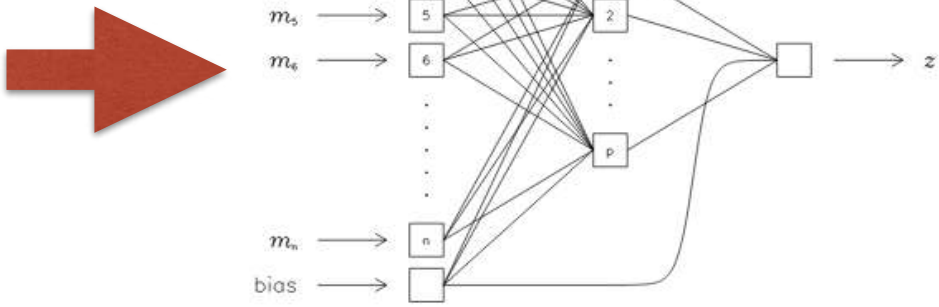
MEDIUM BAND FLUXES



Bailer-Jones+00

No.	Symbol	Description	Scale[a]
1	CVD	Central velocity dispersion	~1 kpc
2	M_{bulge}	Bulge stellar mass	0.5–4 kpc
3	Re	Bulge effective radius	0.5–4 Kpc
4	B/T	Bulge-to-total stellar mass ratio	0.5–8 kpc
5	M_*	Total stellar mass	2–8 kpc
6	M_{disc}	Disc stellar mass	4–10 kpc
7	M_{halo}	Group halo mass	0.1–1 Mpc
8	δ_5	Local density parameter	0.5–3 Mpc

Notes. [a]Approximate 1σ range from centre of galaxy. For photometric quantities half-light radii are used.



HEAVILY PROCESSED DATA

Teimiroonia+16

Other general computer vision features [for images!]

- Pixel Concatenation
- Color histograms
- Texture Features
- Histogram of Gradients
- SIFT

FOR MANY YEARS COMPUTER VISION RESEARCHERS HAVE BEEN TRYING TO FIND THE MOST GENERAL FEATURES

Other general computer vision features [for images!]

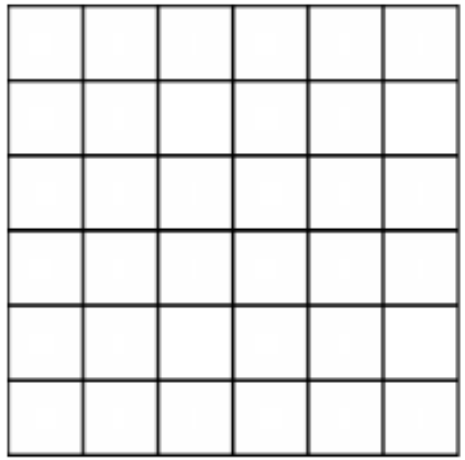
- Pixel Concatenation
- Color histograms
- Texture Features
- Histogram of Gradients
- SIFT

FOR MANY YEARS COMPUTER VISION RESEARCHERS HAVE BEEN TRYING TO FIND THE MOST GENERAL FEATURES

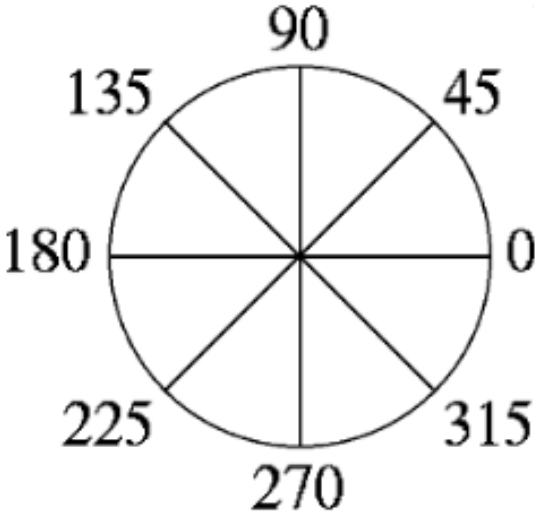
THE BEST CLASSICAL SOLUTION [BEFORE 2012] WHERE BASED **ON LOCAL FEATURES**

HISTOGRAM OF ORIENTED GRADIENTS (HoG)

1. DIVIDE IMAGE INTO SMALL SPATIAL REGIONS CALLED CELLS

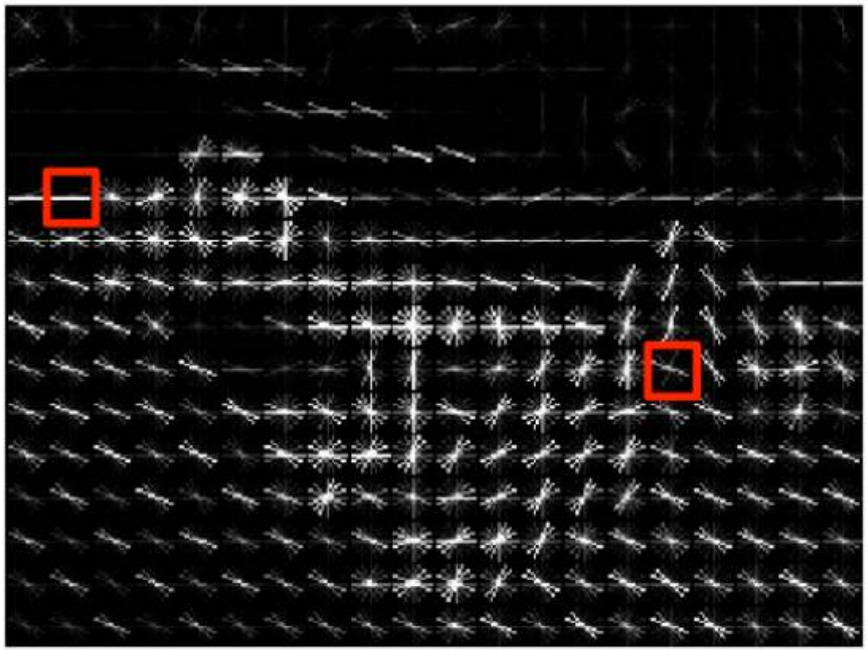


2. COMPUTE INTENSITY GRADIENTS OVER N DIRECTIONS [TYPICALLY 9 FOR IMAGE]



3. COMPUTE WEIGHTED 1-D HISTOGRAM OF ALL DIRECTIONS. A CELL IS REDUCED TO N NUMBERS

HISTOGRAM OF ORIENTED GRADIENTS (HoG)



EVERYTHING IS IN THE FEATURES….WHAT IF I IGNORED SOME IMPORTANT FEATURES?

EVERYTHING IS IN THE FEATURES….WHAT IF I IGNORED SOME IMPORTANT FEATURES?

WHAT ABOUT USING RAW DATA?

ALL INFORMATION IS IN THE INPUT DATA

WHY REDUCING ?

LET THE NETWORK FIND THE INFO

WHAT ABOUT USING RAW DATA?

ALL INFORMATION IS IN THE INPUT DATA

WHY REDUCING ?

LET THE NETWORK FIND THE INFO

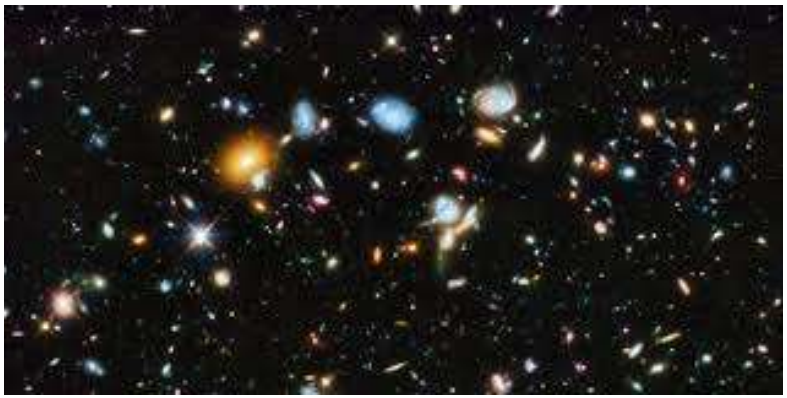
LARGE DIMENSION SIGNALS SUCH AS IMAGES OR SPECTRA WOULD REQUIRE TREMENDOUSLY LARGE MODELS

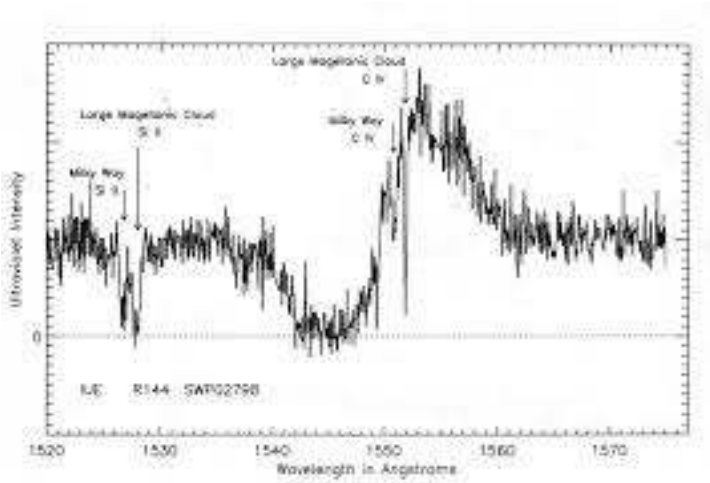
A 512x512 image as input of a fully connected layer producing output of same size:

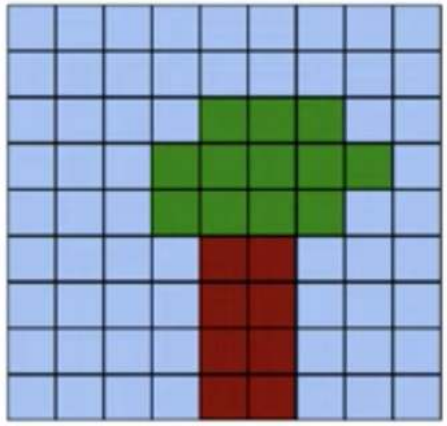
$$(512 \times 512)^2 = 7e10$$

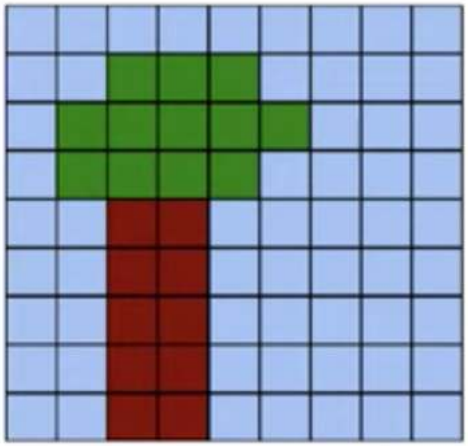
BUT

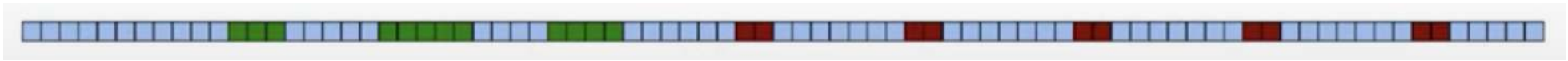
FEEDING INDIVIDUAL RESOLUTION ELEMENTS IS NOT VERY EFFICIENT SINCE IT LOOSES ALL INVARIANCE TO TRANSLATION AND IGNORES CORRELATION IN THE DATA AT ALL SCALES











(Dielemann@Deepmind)

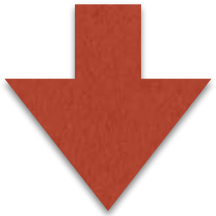
TWO BASIC PROPERTIES OF IMAGING DATA (BUT ALSO SPECTROSCOPY IN SOME SENSE) ARE **LOCALITY**
TRANSLATION INVARIANCE

locality: nearby pixels are more strongly correlated

translational invariance: meaningful patterns can appear anywhere in the image

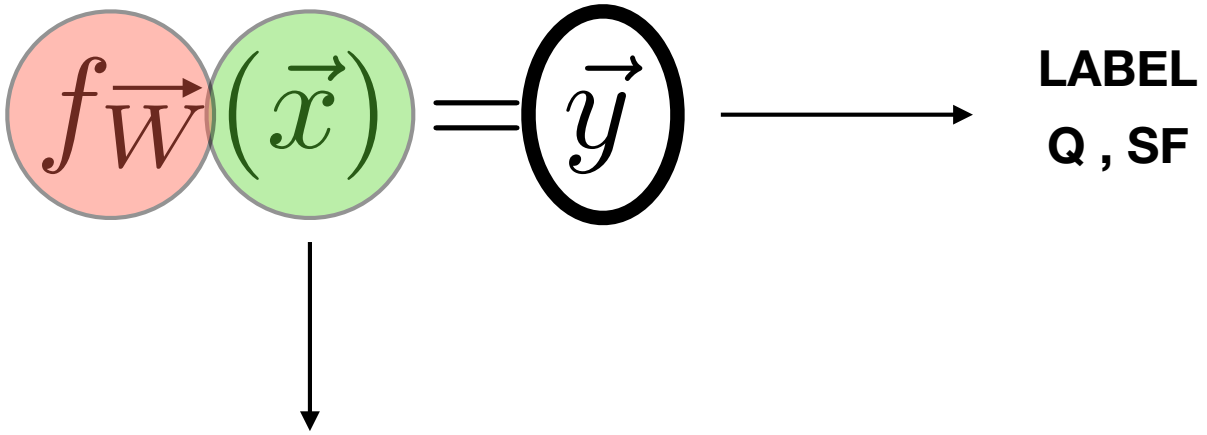
(Dielemann@Deepmind)

FEEDING INDIVIDUAL RESOLUTION ELEMENTS IS NOT VERY EFFICIENT SINCE IT LOOSES ALL INVARIANCE TO TRANSLATION



SO?

DEEP LEARNING



LET THE MACHINE FIGURE THIS OUT ("unsupervised feature extraction")

LET'S GO A STEP FORWARD INTO LOOSING CONTROL….

TWO BASIC PROPERTIES OF IMAGING DATA (BUT ALSO SPECTROSCOPY IN SOME SENSE) ARE **LOCALITY**
TRANSLATION INVARIANCE

locality: nearby pixels are more strongly correlated

translational invariance: meaningful patterns can appear anywhere in the image

(Dielemann@Deepmind)

CONVOLUTIONAL NEURAL NETWORKS

Discrete Convolution

1D: [Spectra]

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

2D: [Images]

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

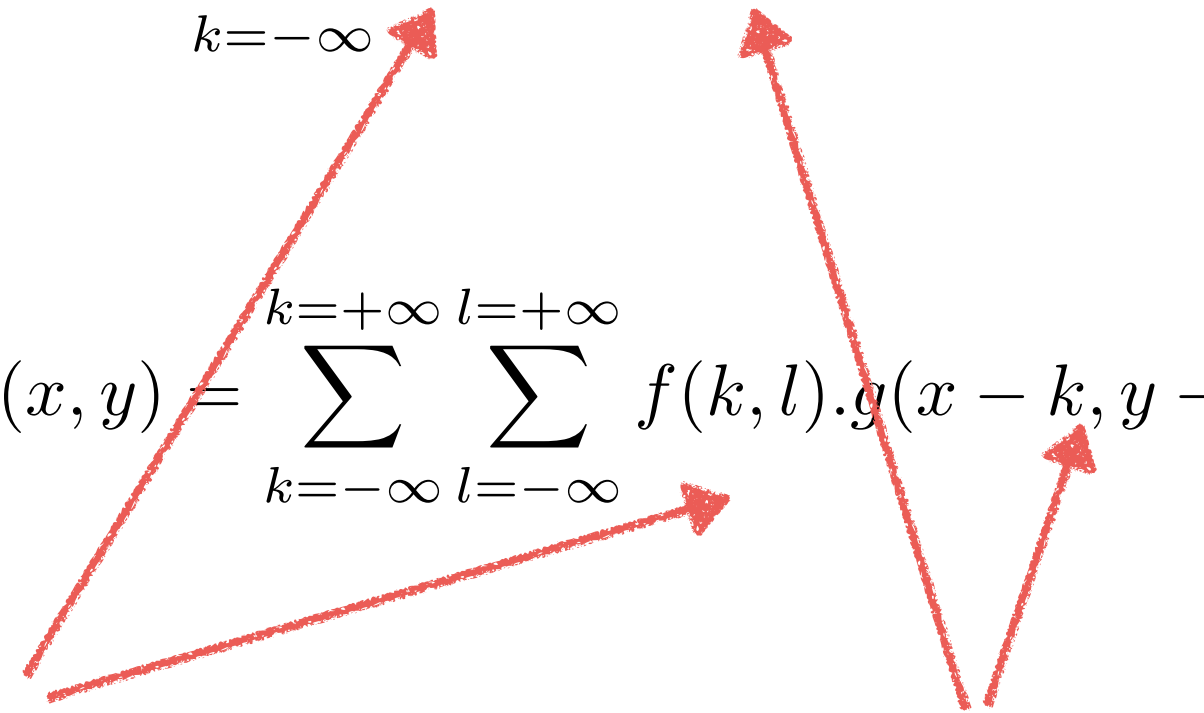
DISCRETE CONVOLUTION

1D: [Spectra]

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

2D: [Images]

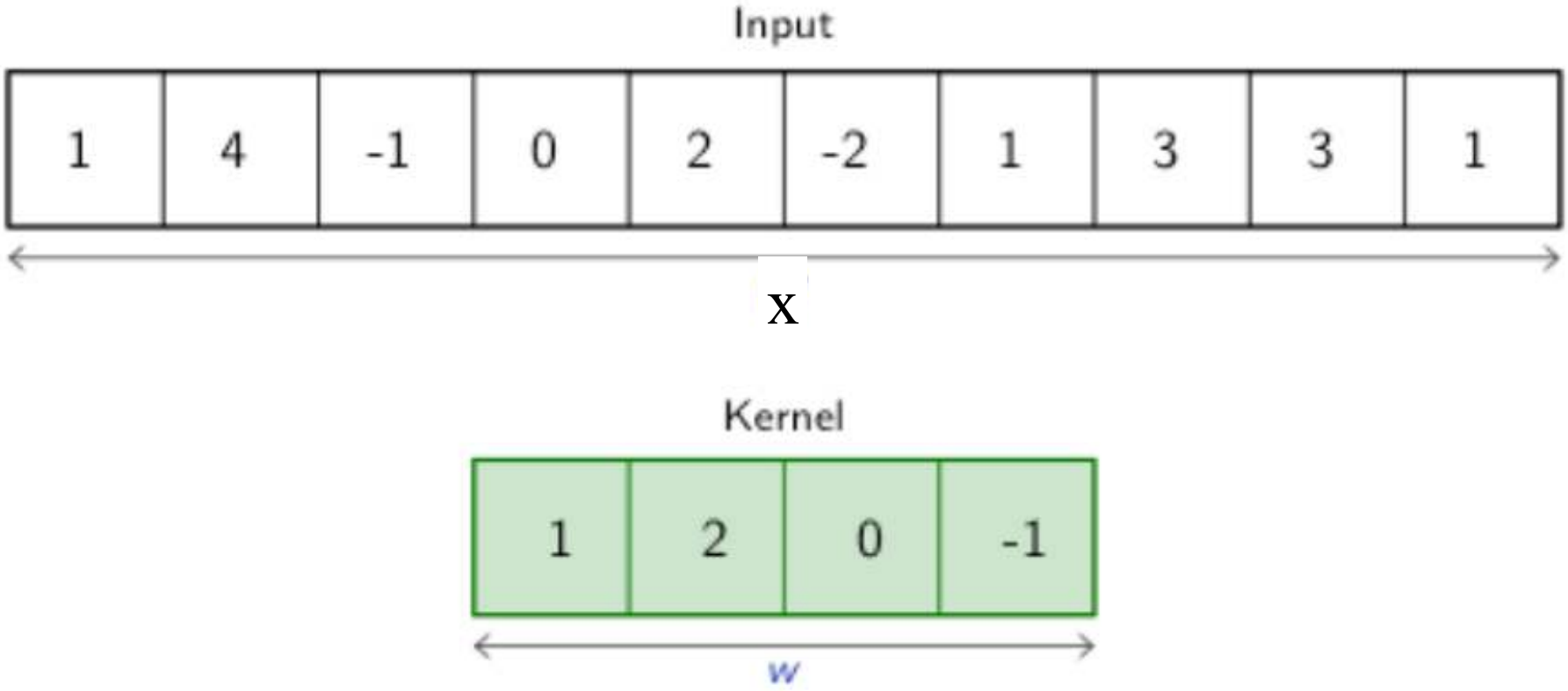
$$f(x,y) * g(x,y) = \sum_{k=-\infty}^{k=+\infty} \sum_{l=-\infty}^{l=+\infty} f(k,l).g(x-k,y-l)$$



CONVOLUTION KERNEL

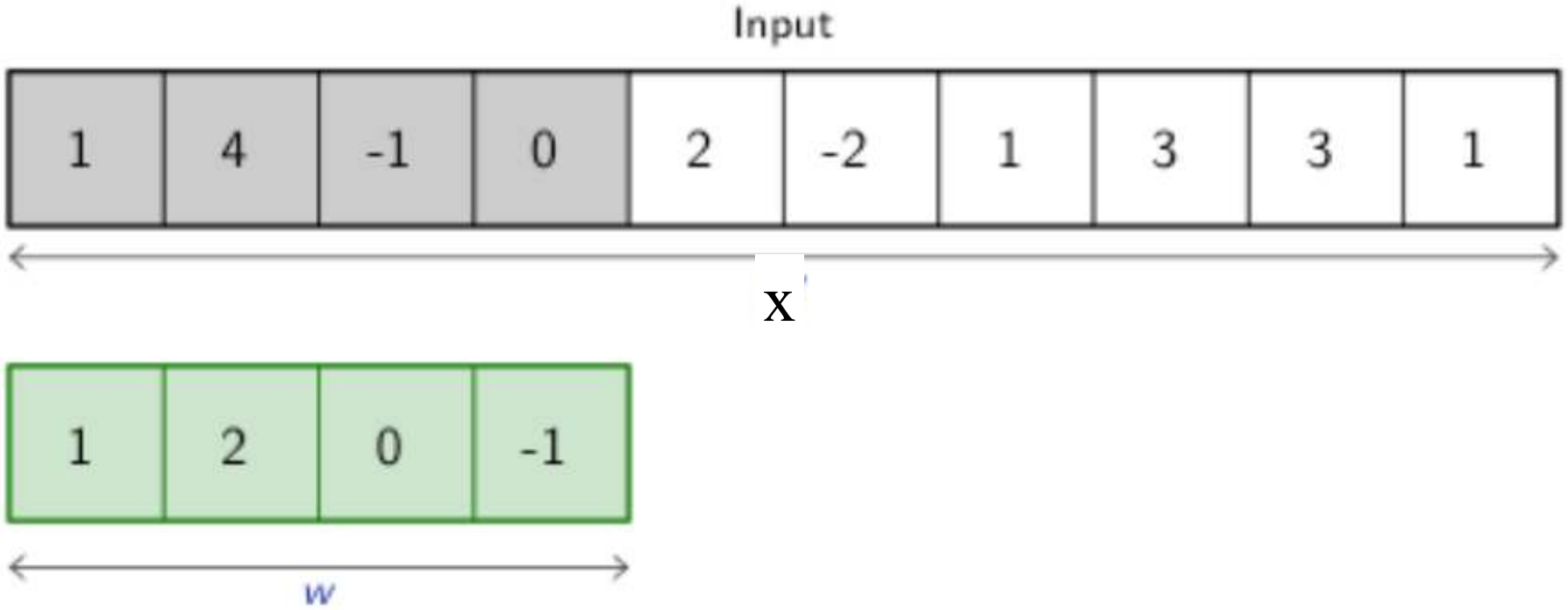
INPUT DATA

1-D CONVOLUTION



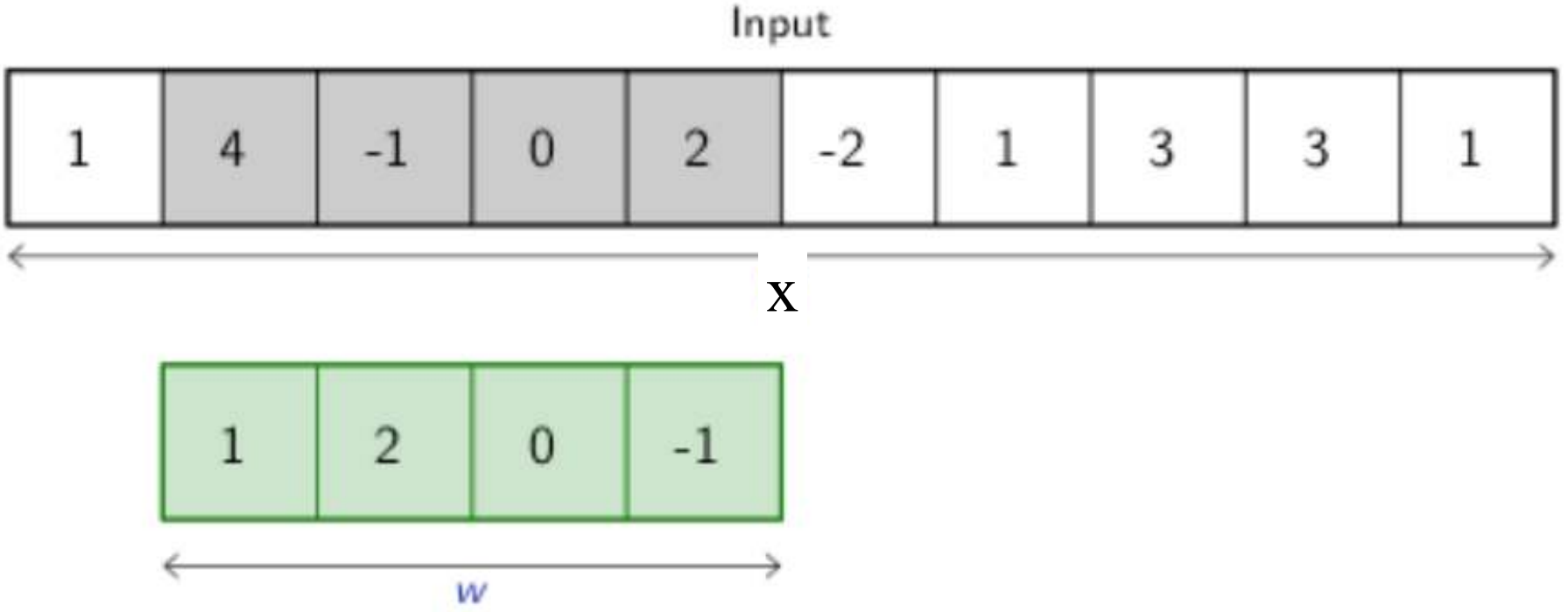
credit

1-D CONVOLUTION



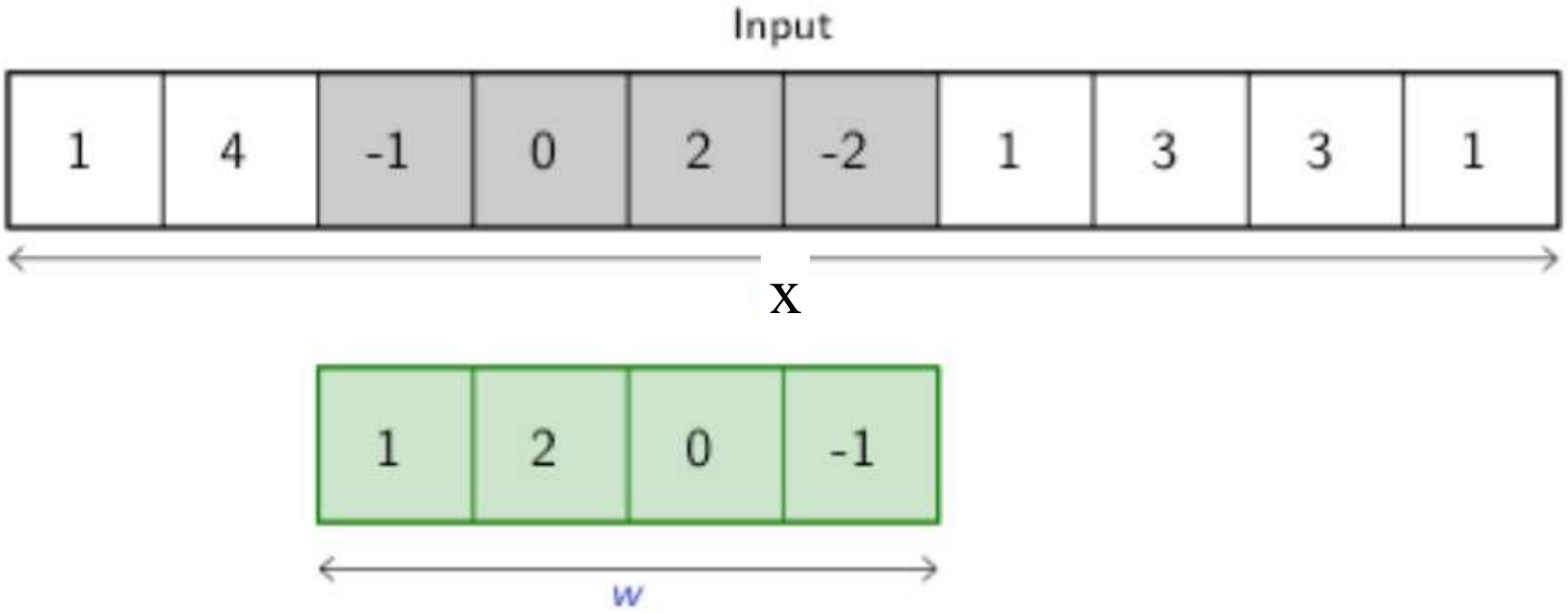
credit

1-D CONVOLUTION



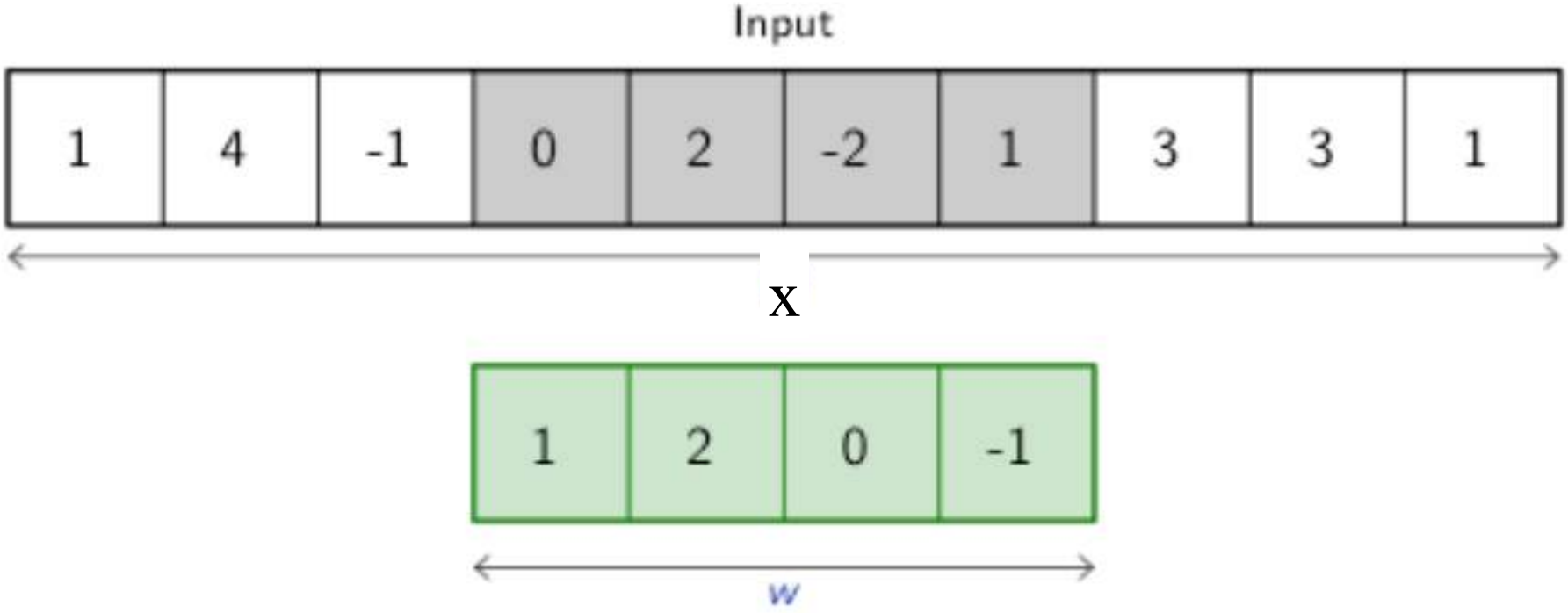
credit

1-D CONVOLUTION



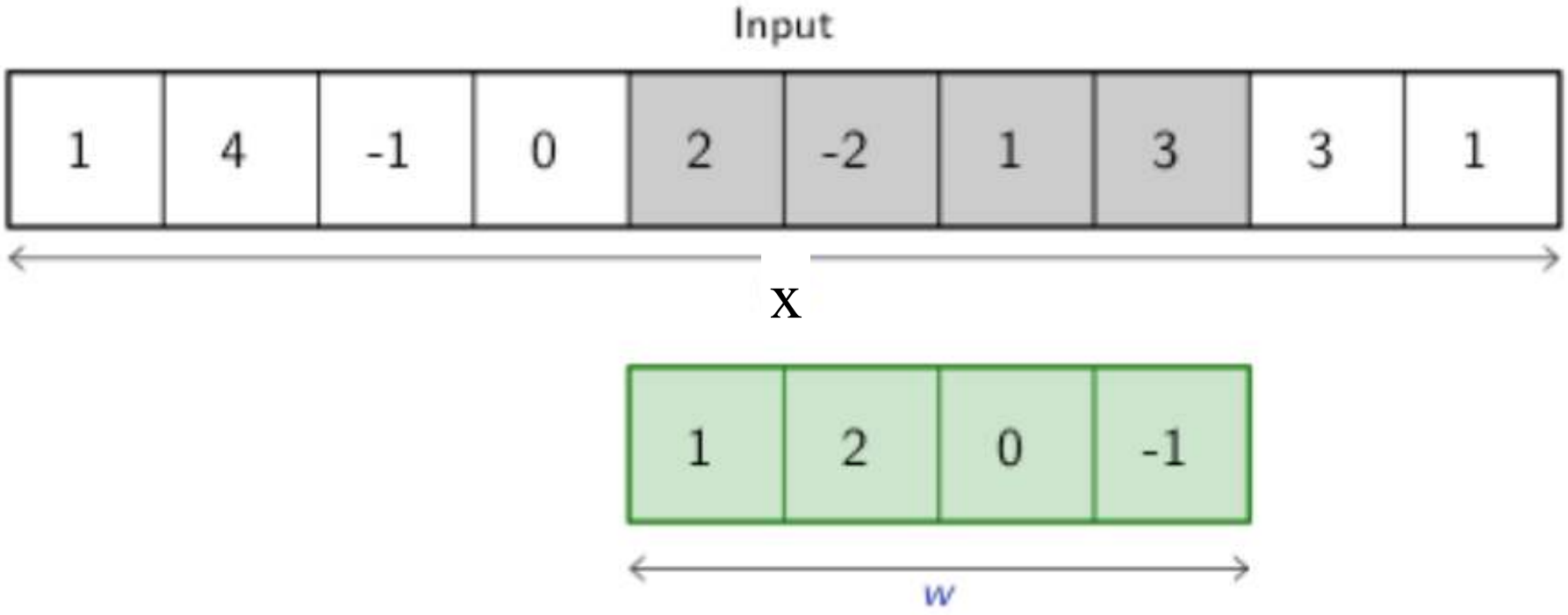
credit

1-D CONVOLUTION



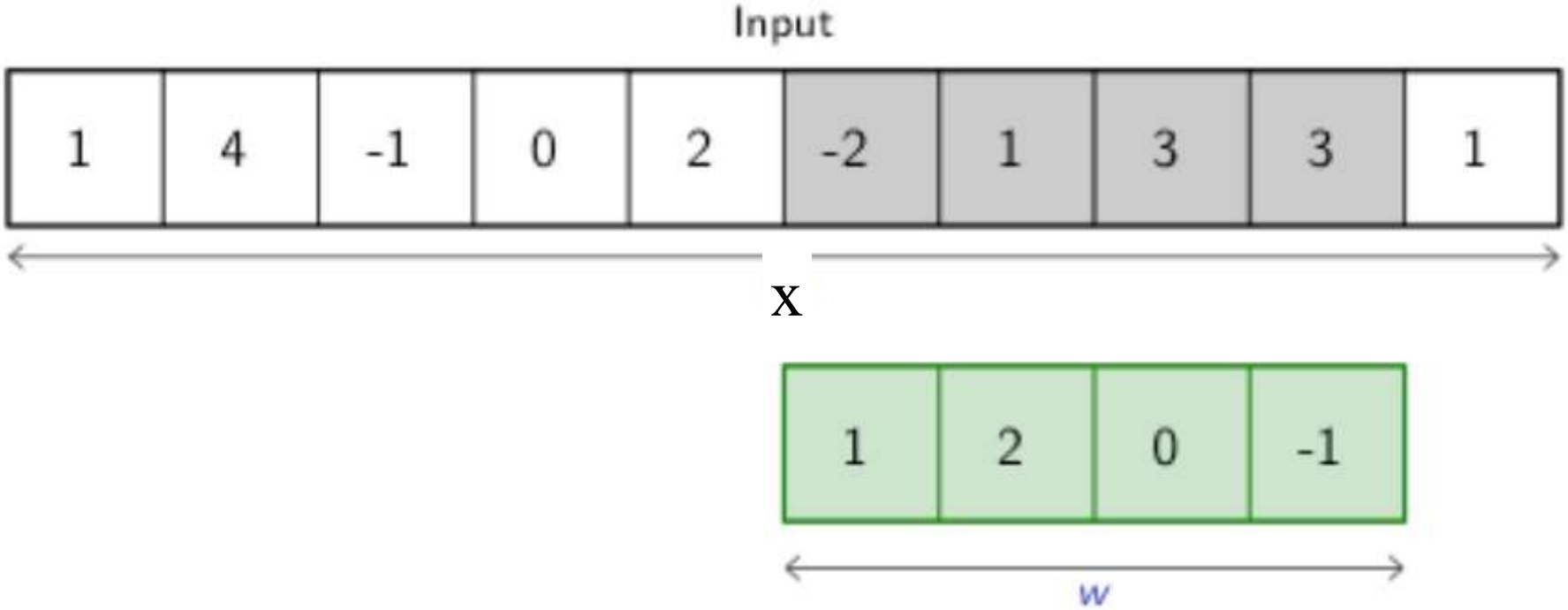
credit

1-D CONVOLUTION



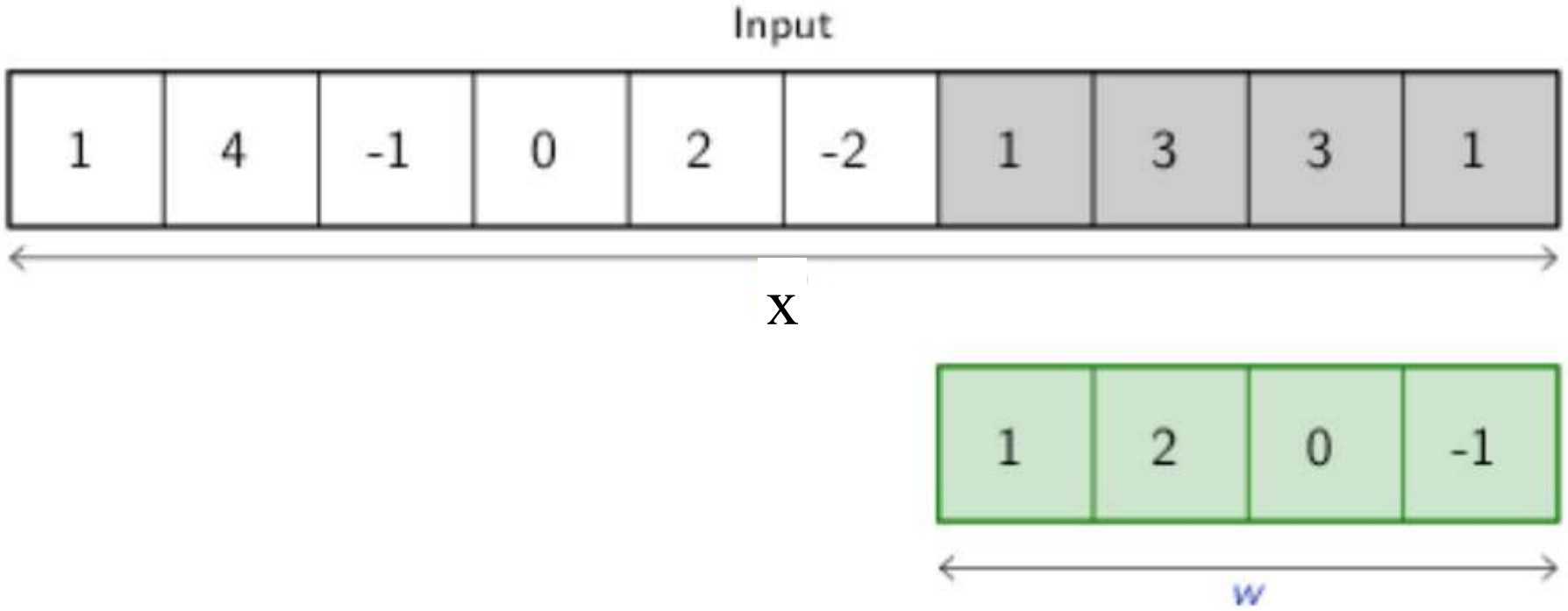
credit

1-D CONVOLUTION



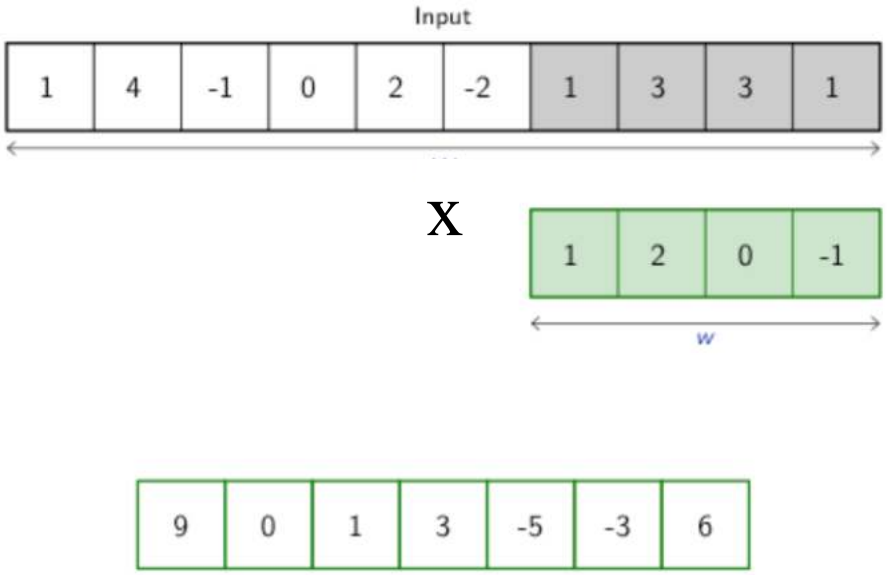
credit

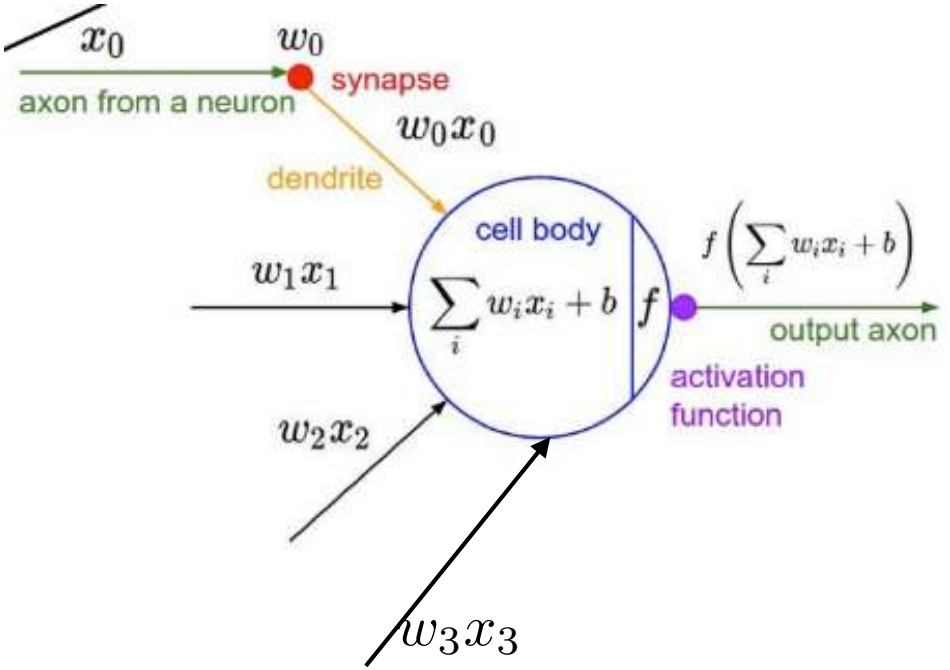
1-D CONVOLUTION



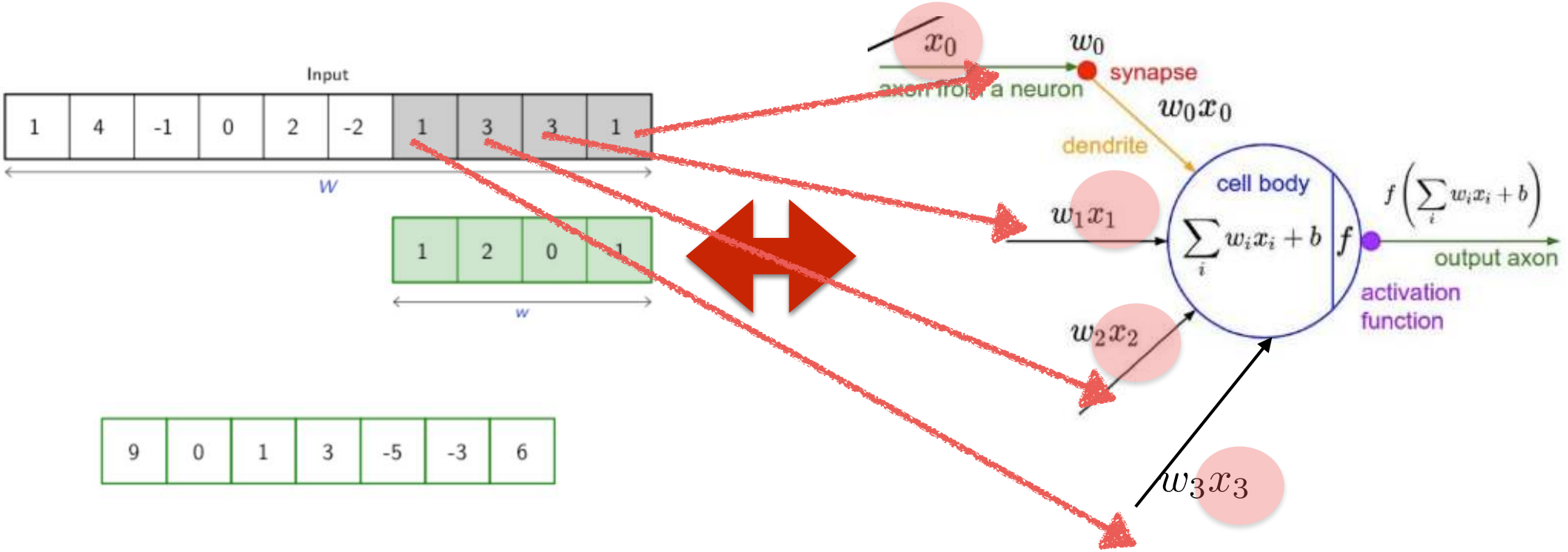
credit

THE CONVOLUTION BUILDING BLOCK OPERATION (BEFORE ACTIVATION) IS EQUIVALENT TO A NEURON WITH AS MANY INPUTS AS KERNEL ELEMENTS AND WEIGHTS EQUAL TO THE KERNEL

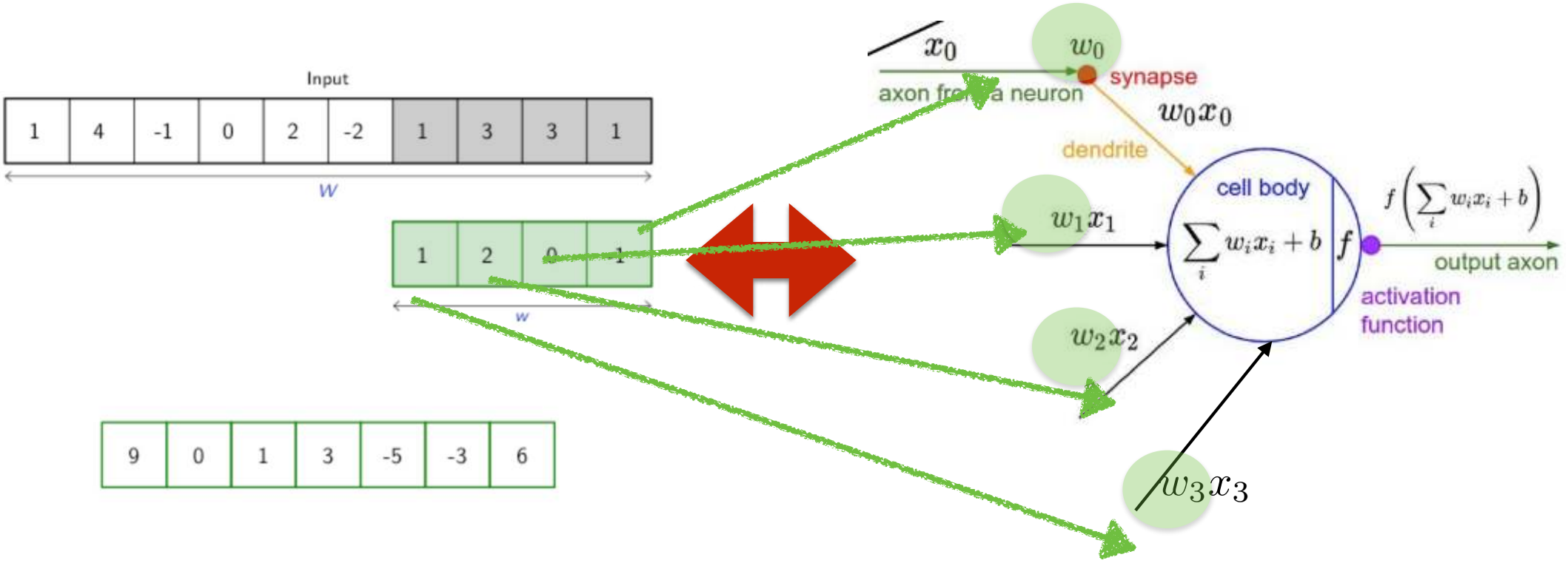




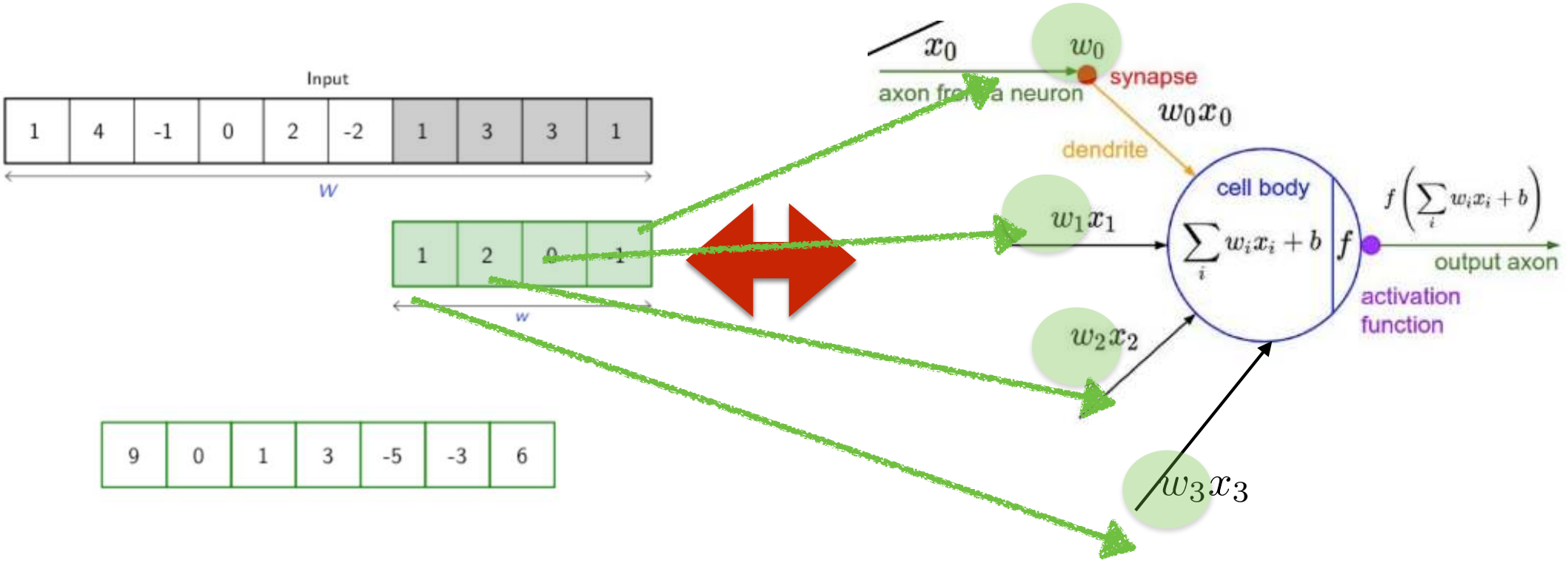
THE CONVOLUTION BUILDING BLOCK OPERATION (BEFORE ACTIVATION) IS EQUIVALENT TO A NEURON WITH AS MANY INPUTS AS KERNEL ELEMENTS AND WEIGHTS EQUAL TO THE KERNEL



THE CONVOLUTION BUILDING BLOCK OPERATION (BEFORE ACTIVATION) IS EQUIVALENT TO A NEURON WITH AS MANY INPUTS AS KERNEL ELEMENTS AND WEIGHTS EQUAL TO THE KERNEL



THE CONVOLUTION BUILDING BLOCK OPERATION (BEFORE ACTIVATION) IS EQUIVALENT TO A NEURON WITH AS MANY INPUTS AS KERNEL ELEMENTS AND WEIGHTS EQUAL TO THE KERNEL

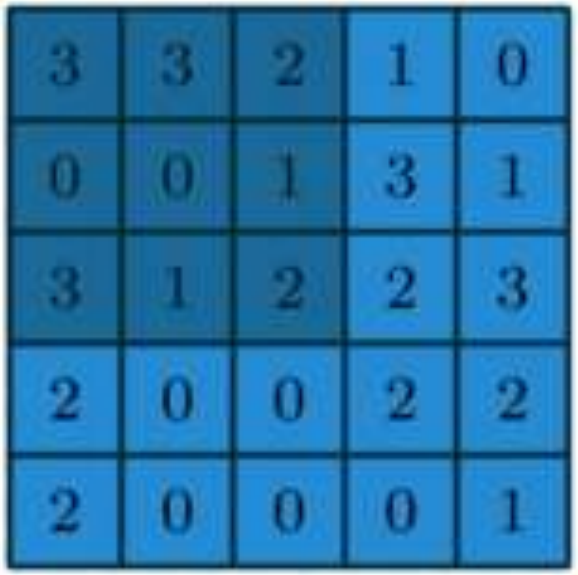


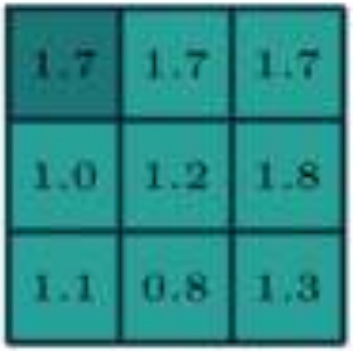
WITH THE ADVANTAGE THAT THE SAME WEIGHTS ARE APPLIED TO ALL THE SIGNAL: TRANSLATION INVARIANCE

2-D CONVOLUTION

SAME IDEA, BUT THE KERNEL IS NOW 2D

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9





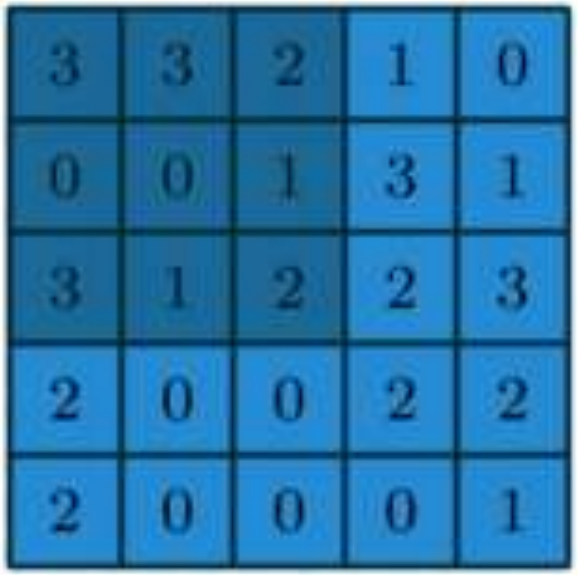
KERNEL **INPUT (IMAGE)** **OUTPUT**

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

2-D CONVOLUTION

SAME IDEA, BUT THE KERNEL IS NOW 2D

1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

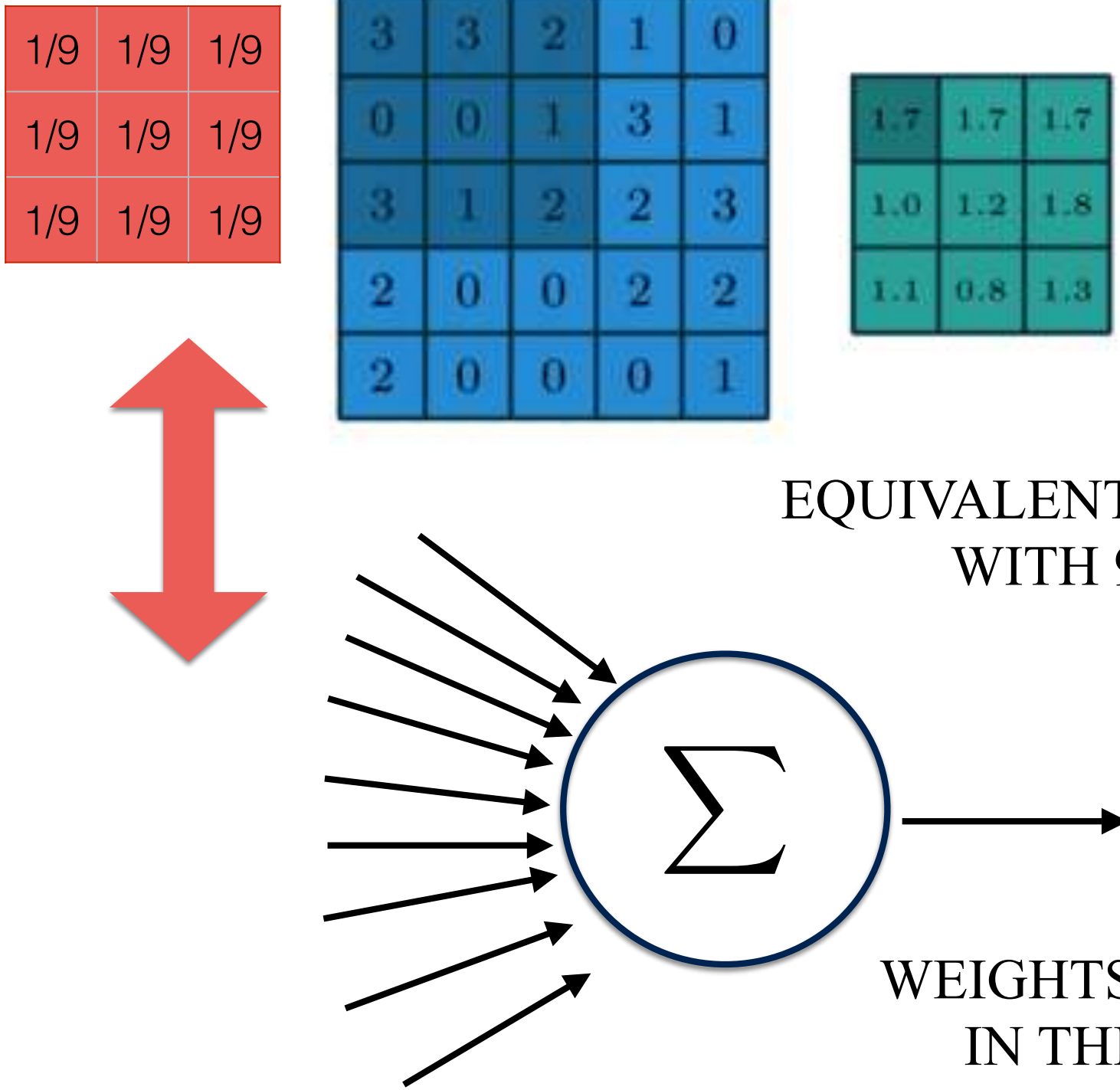


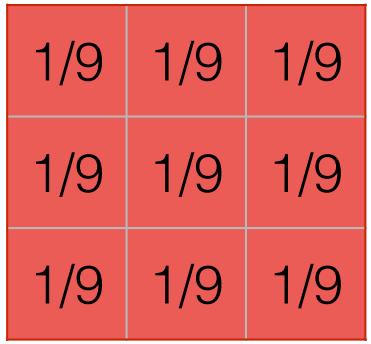
1.7	1.7	1.7
1.0	1.2	1.8
1.1	0.8	1.3

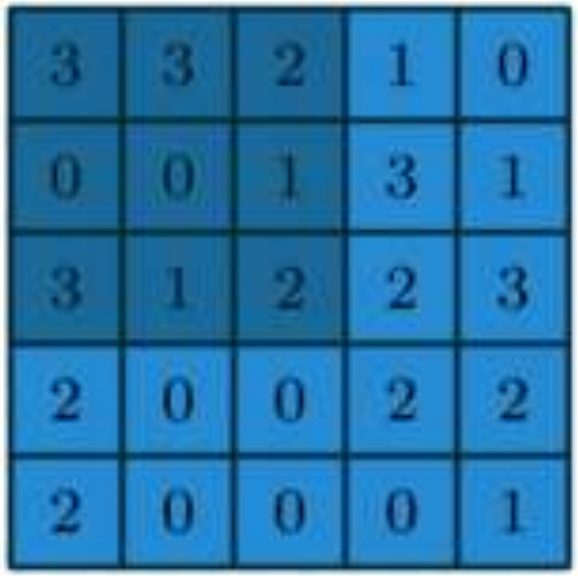
IN THE EXAMPLE: EACH 3x3 REGION GENERATES AN OUTPUT

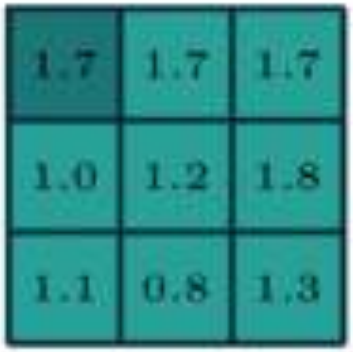
$$Size_{output} = Size_{input} - Size_{kernel} + 1$$

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

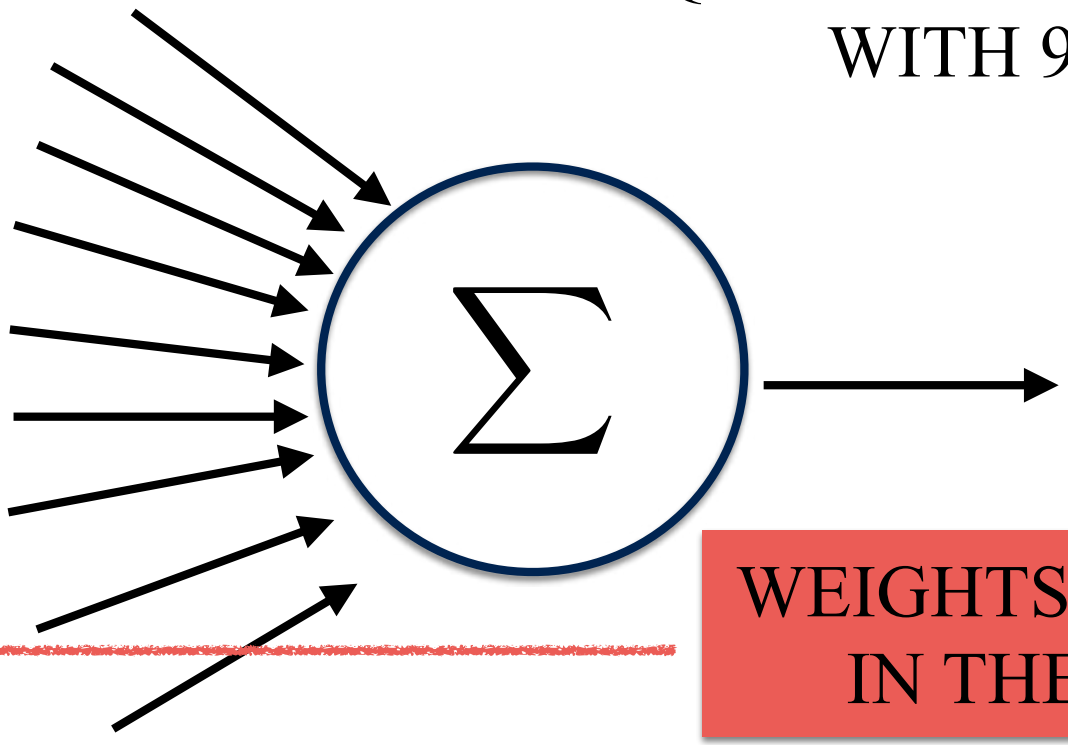

1/9 1/9 1/9
1/9 1/9 1/9
1/9 1/9 1/9
3 3 2 1 0
0 0 1 3 1
3 1 2 2 3
2 0 0 2 2
2 0 0 0 1
1.7 1.7 1.7
1.0 1.2 1.8
1.1 0.8 1.3
EQUIVALENT TO A NEURON
WITH 9 INPUTS
Σ
WEIGHTS ARE CODED
IN THE KERNEL





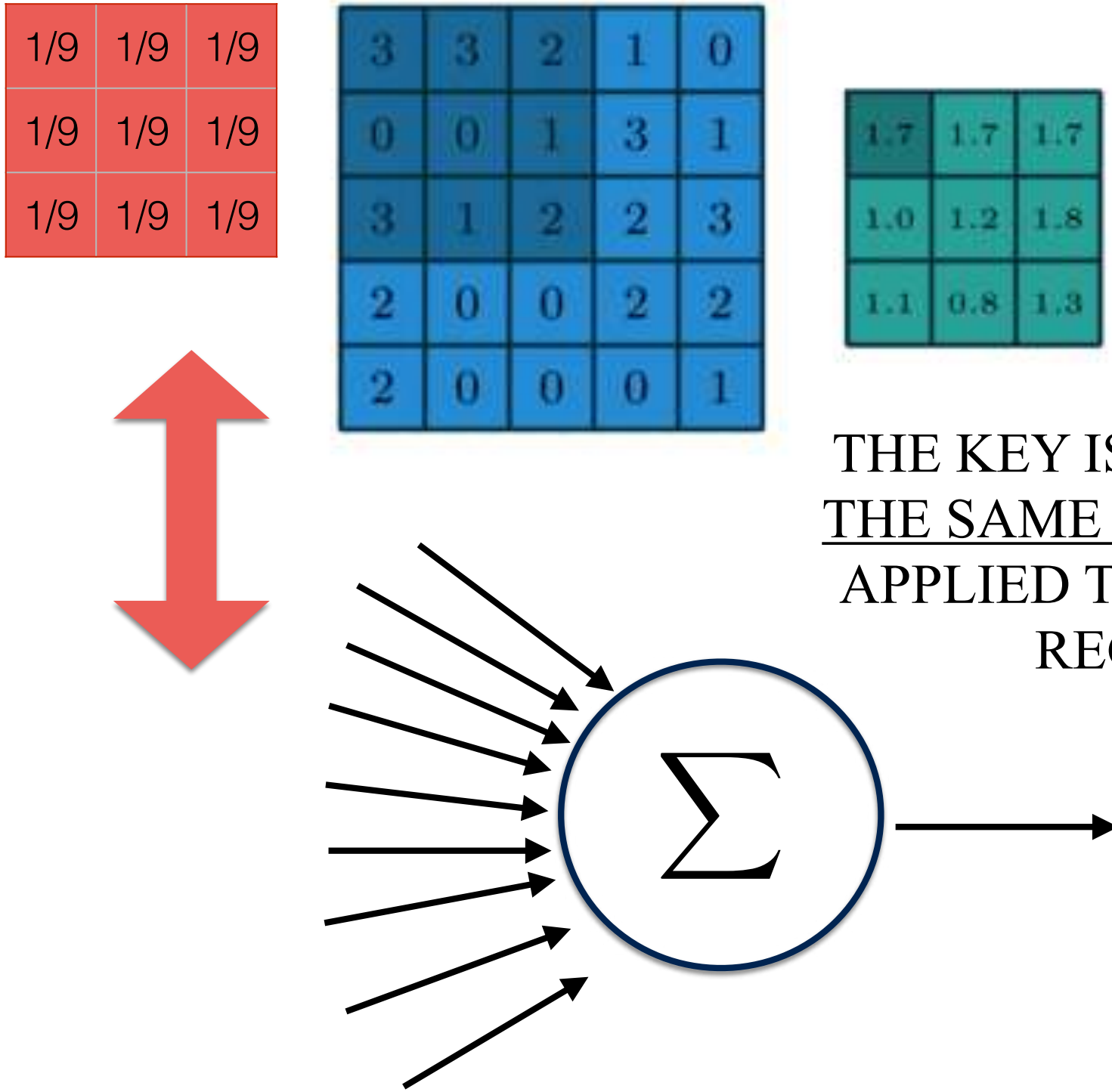


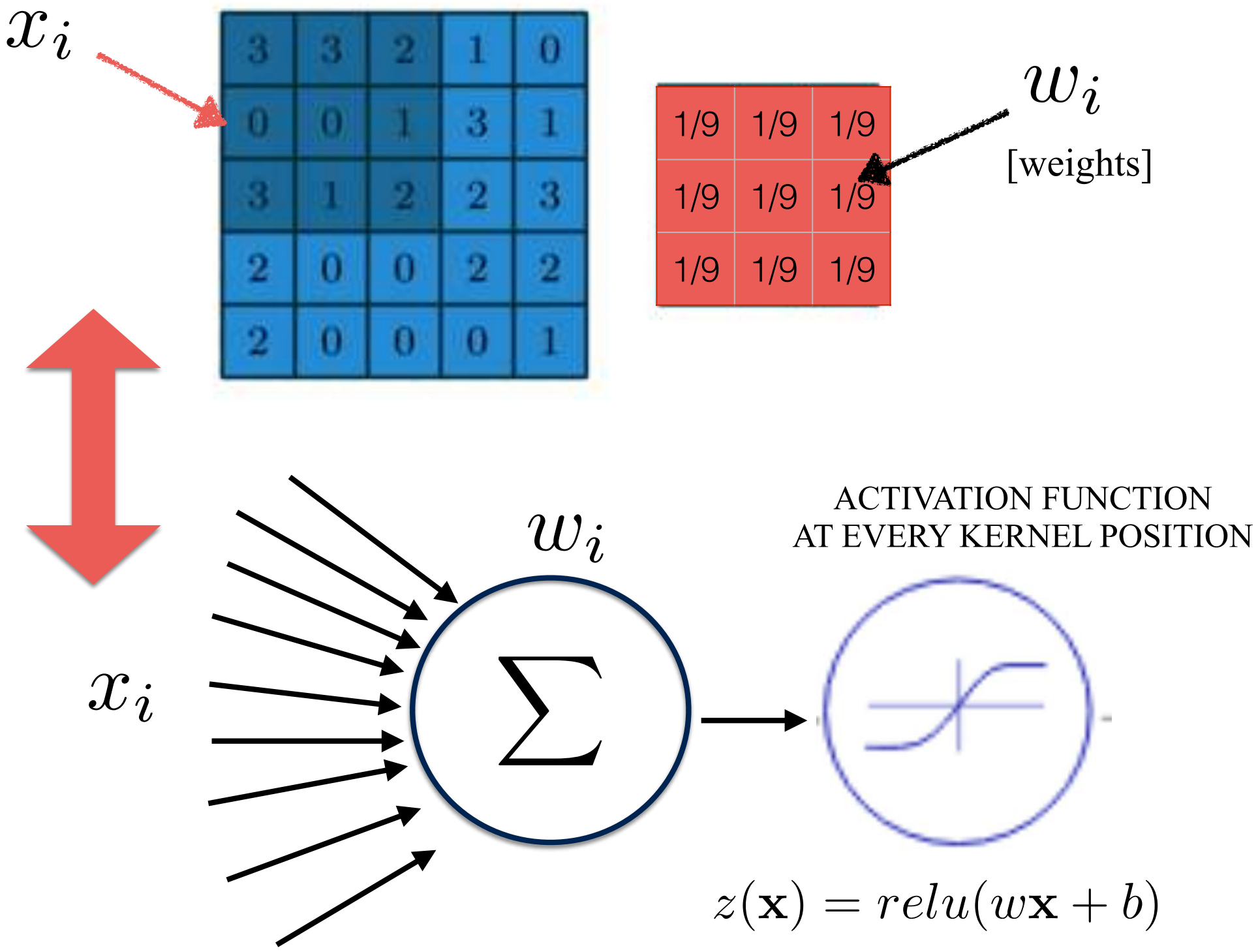
EQUIVALENT TO A NEURON
WITH 9 INPUTS



THIS IS WHAT
THE
NETWORK
LEARNS!

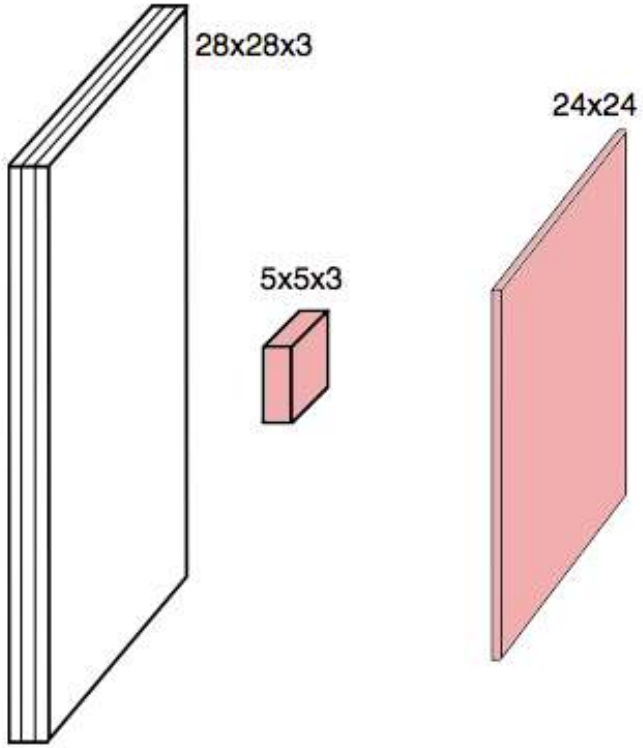
WEIGHTS ARE CODED
IN THE KERNEL

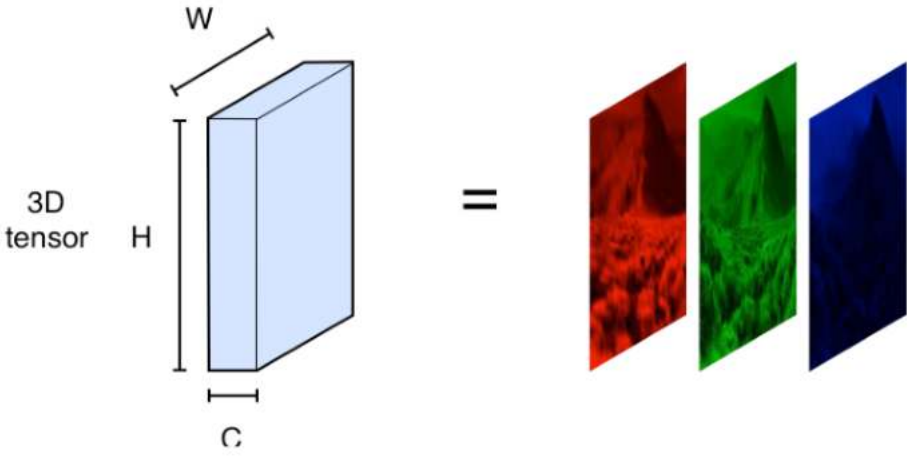

1/9 1/9 1/9
1/9 1/9 1/9
1/9 1/9 1/9
3 3 2 1 0
0 0 1 3 1
3 1 2 2 3
2 0 0 2 2
2 0 0 0 1
1.7 1.7 1.7
1.0 1.2 1.8
1.1 0.8 1.3
THE KEY IS AGAIN THAT
THE SAME WEIGHTS ARE
APPLIED TO ALL IMAGE
REGIONS
Σ


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
w_i
[weights]
1/9 1/9 1/9
1/9 1/9 1/9
1/9 1/9 1/9
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)

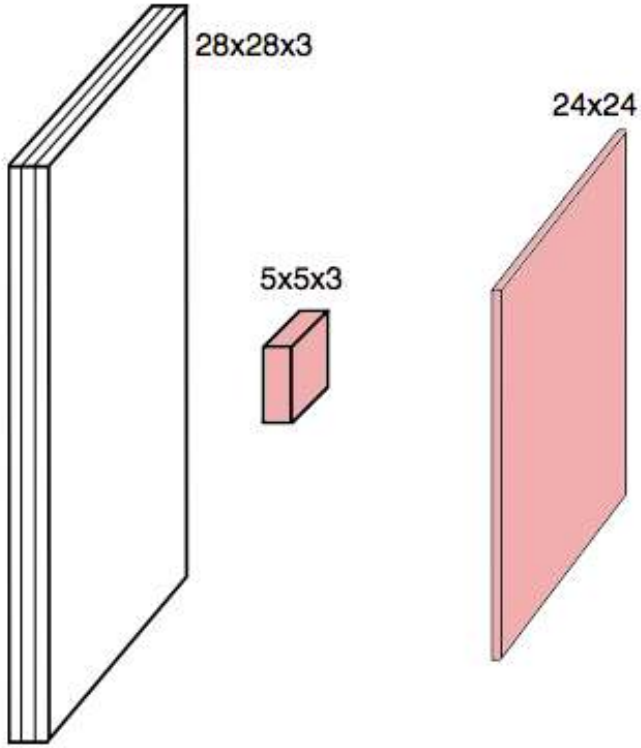
A COLOR IMAGE IS A TENSOR
OF SIZE height x width x channels

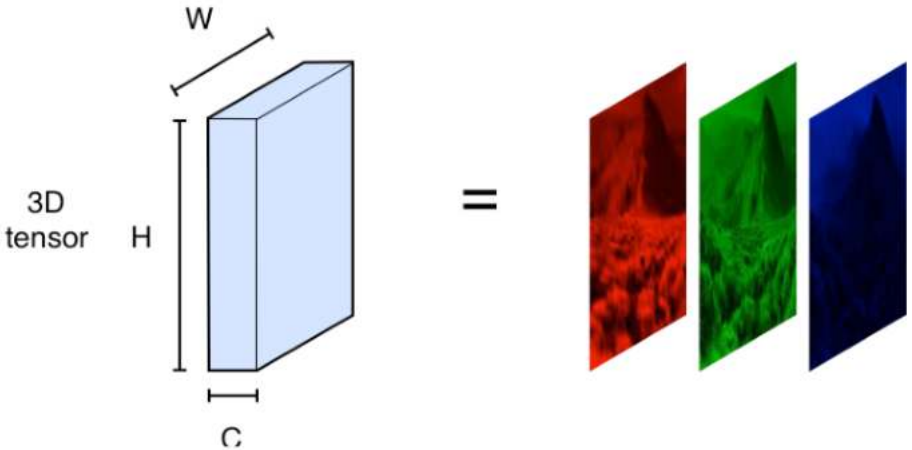




CONVOLUTIONS CAN ALSO BE COMPUTED ACROSS CHANNELS (OR COLORS)

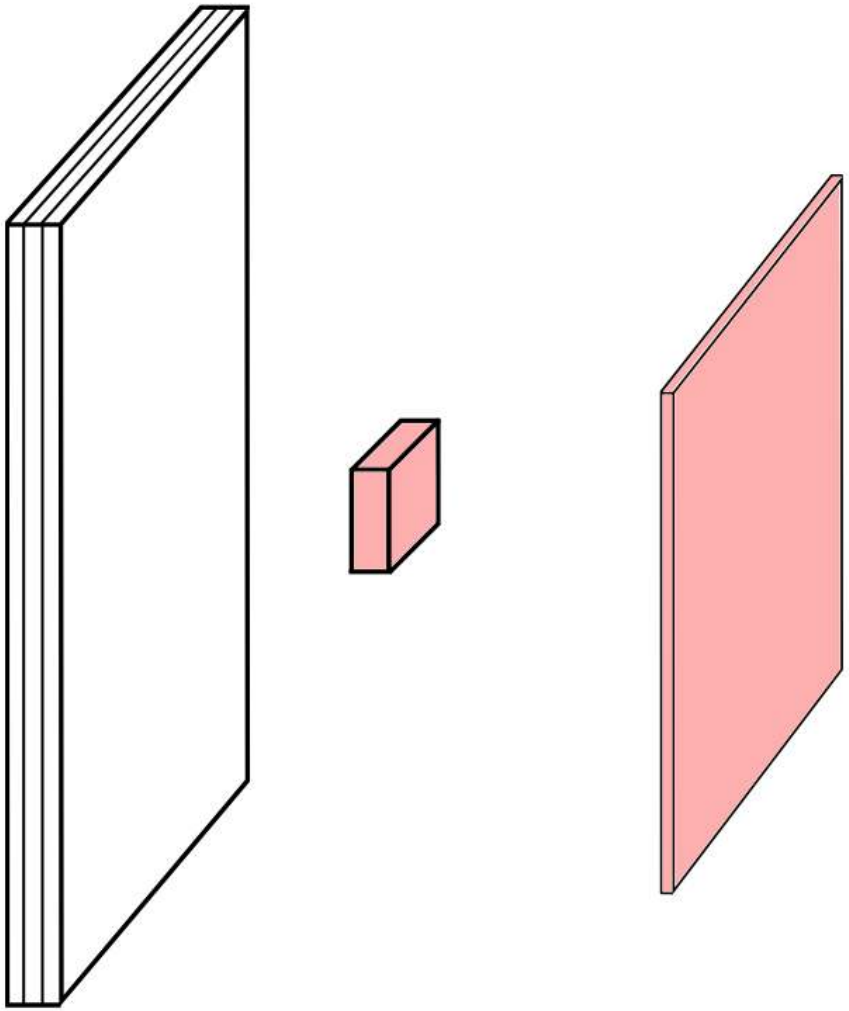
A COLOR IMAGE IS A TENSOR
OF SIZE height x width x channels





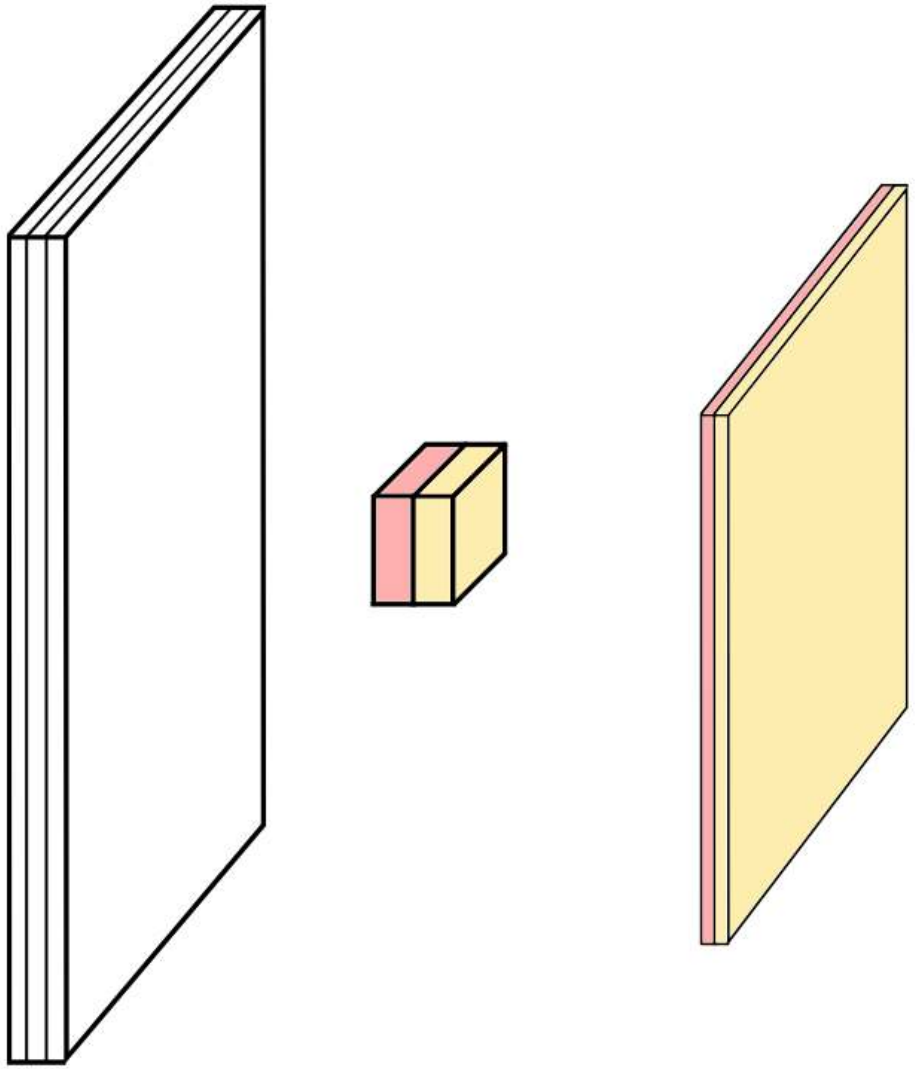
THEN THE KERNEL HAS ALSO 3 CHANNELS

MULTIPLE CONVOLUTIONS WITH DIFFERENT KERNELS CAN BE PERFORMED



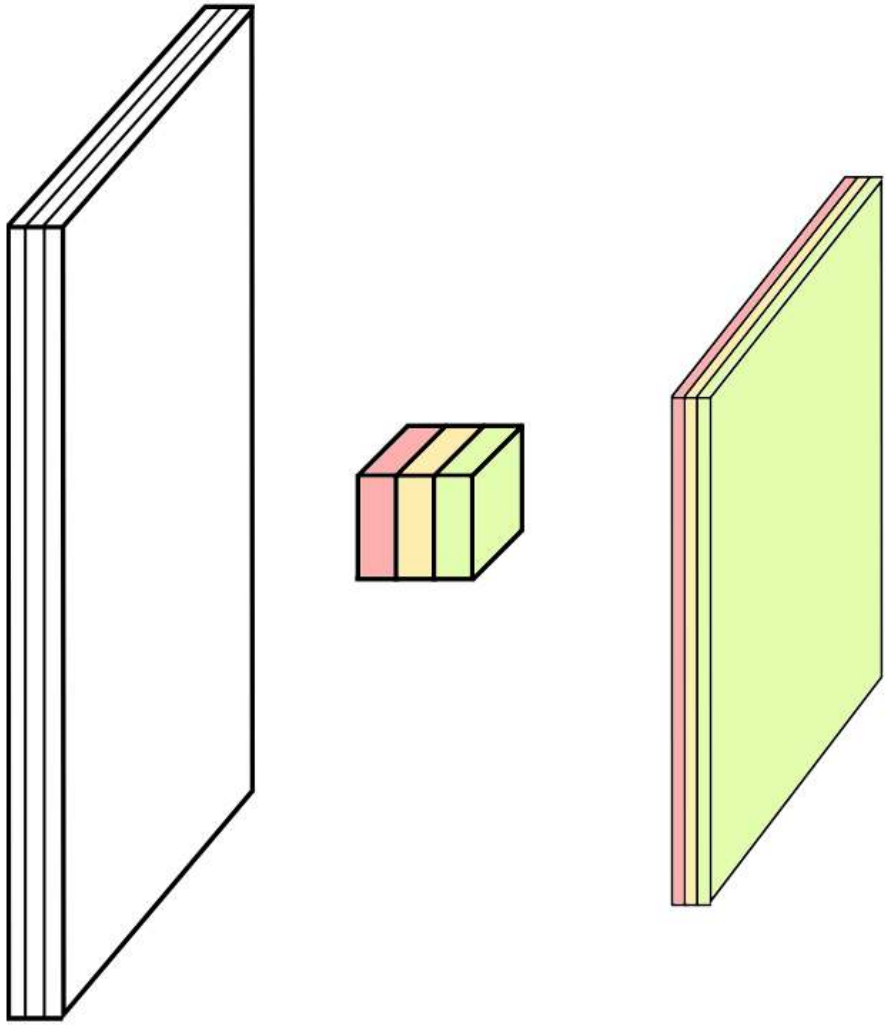
credit

MULTIPLE CONVOLUTIONS WITH DIFFERENT KERNELS CAN BE PERFORMED



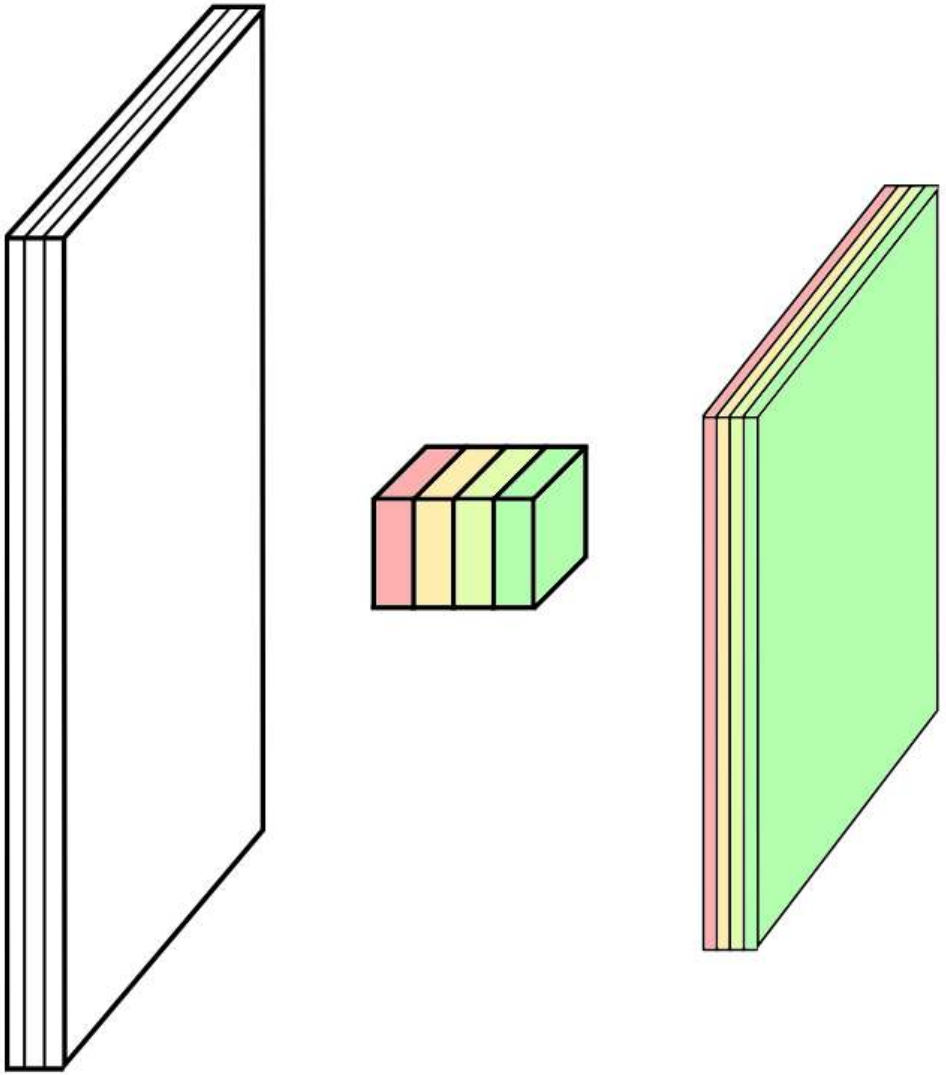
credit

MULTIPLE CONVOLUTIONS WITH DIFFERENT KERNELS CAN BE PERFORMED



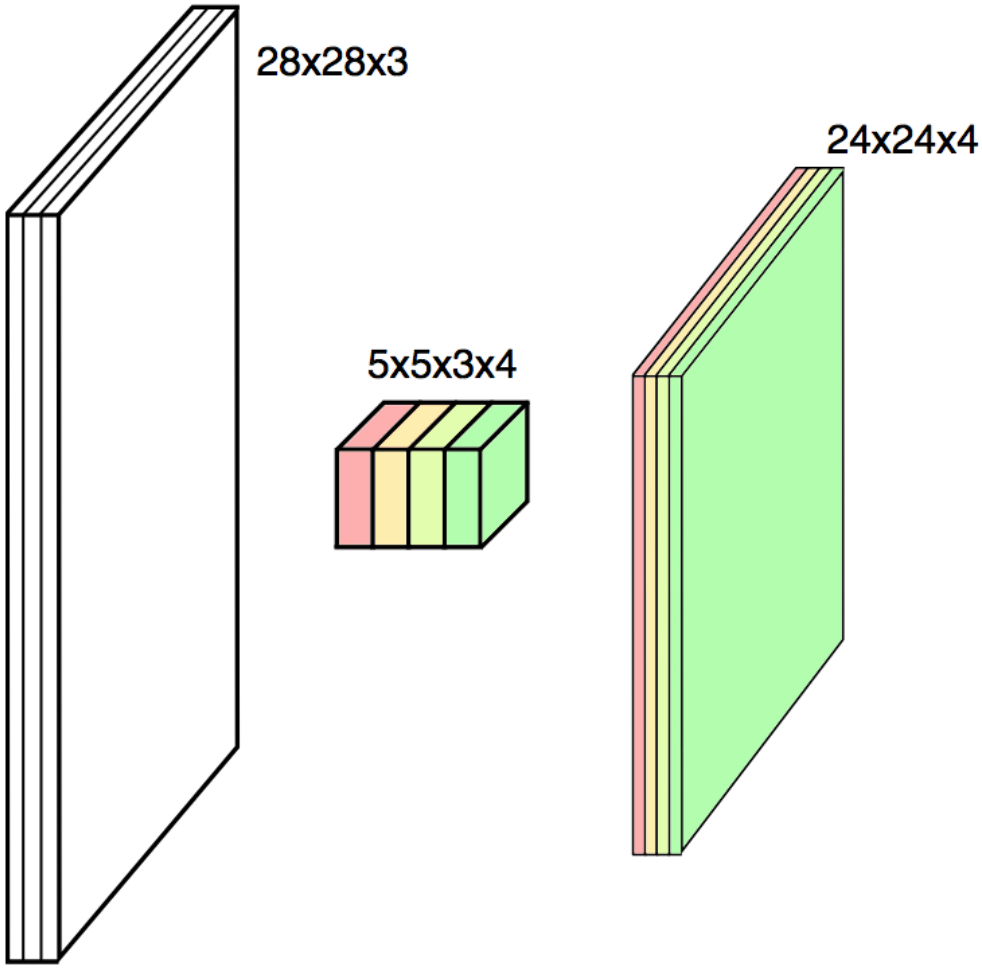
credit

MULTIPLE CONVOLUTIONS WITH DIFFERENT KERNELS CAN BE PERFORMED



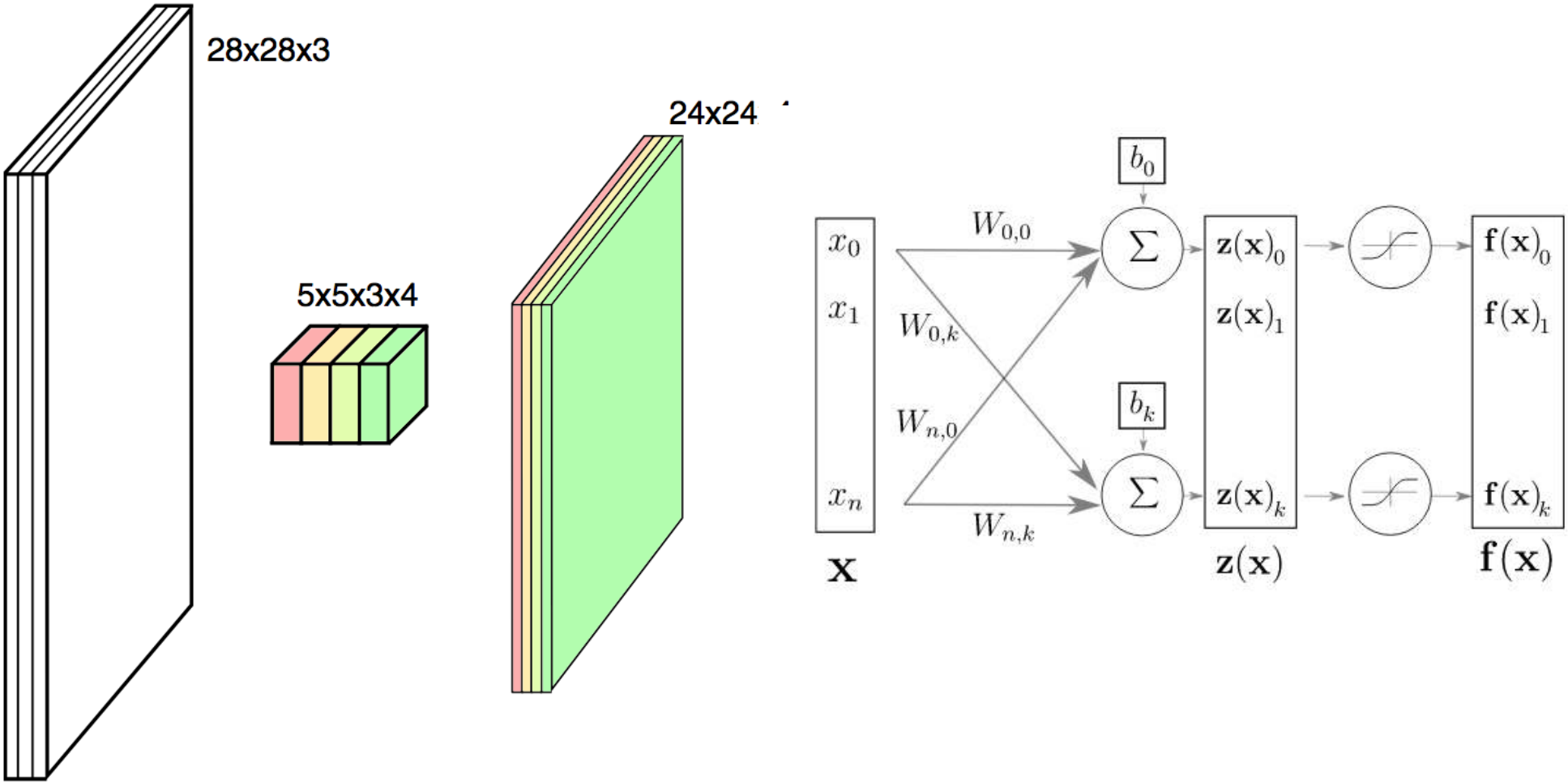
credit

MULTIPLE CONVOLUTIONS WITH DIFFERENT KERNELS CAN BE PERFORMED



credit

MULTIPLE CONVOLUTIONS WITH DIFFERENT KERNELS CAN BE PERFORMED

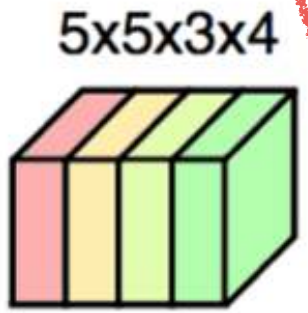


credit

IN KERAS…

model = Sequential()

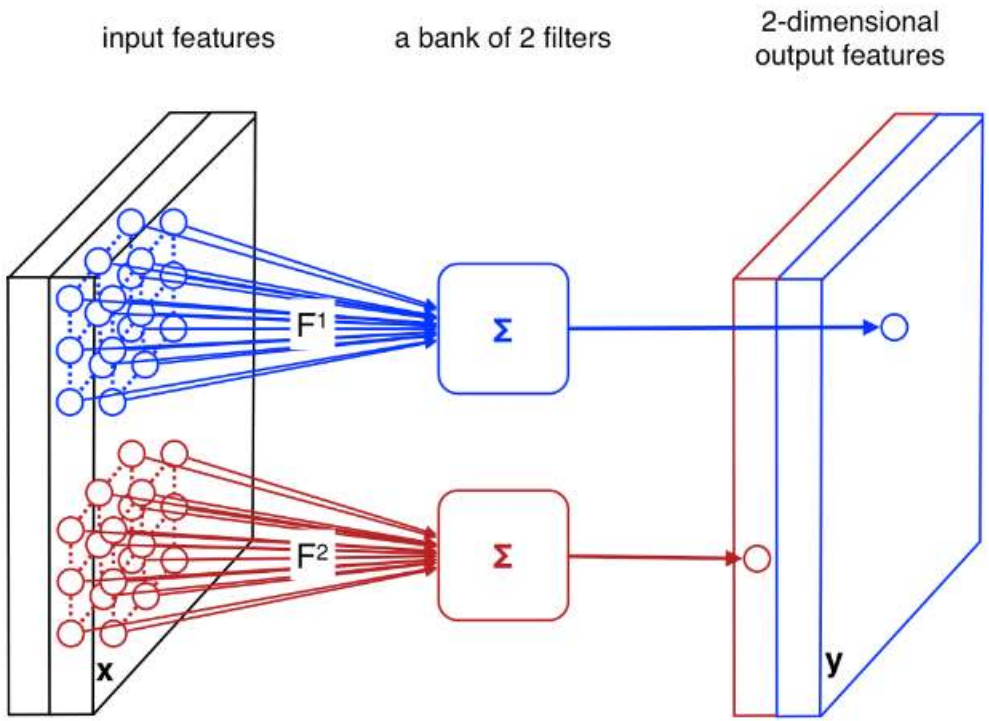
model.add(Convolution2D(4,5,5, activation="relu"))



depth

sizeX, sizeY

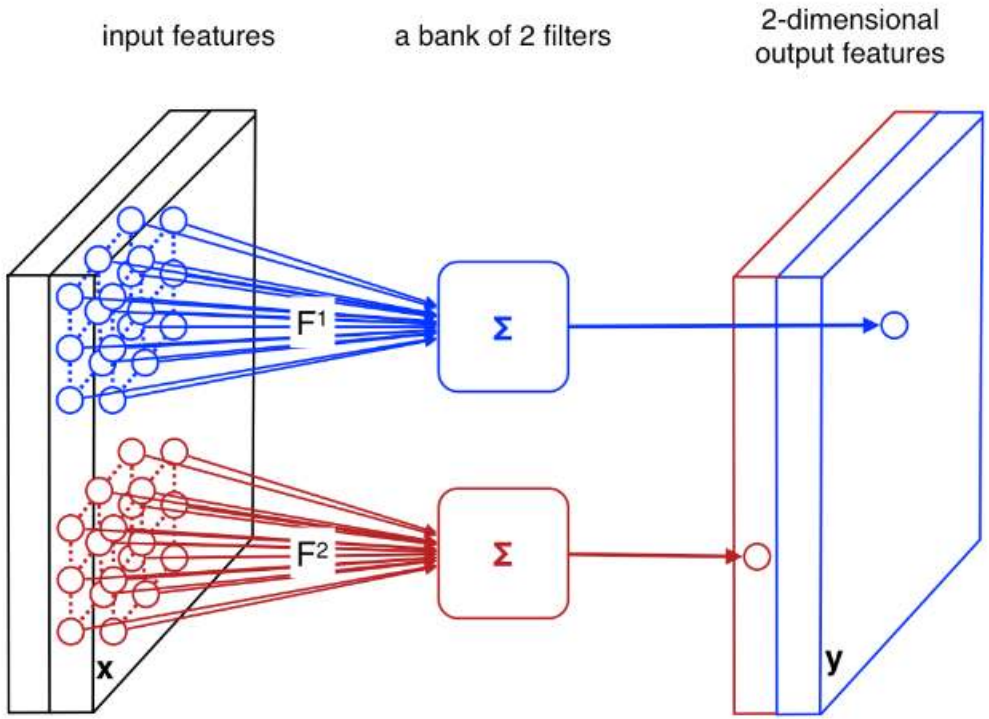
SINCE CONVOLUTIONS OUTPUT ONE SCALAR, THEY CAN BE SEEN AS AN INDIVIDUAL NEURON WITH A RECEPTIVE FIELD LIMITED TO THE KERNEL DIMENSIONS



Credit

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

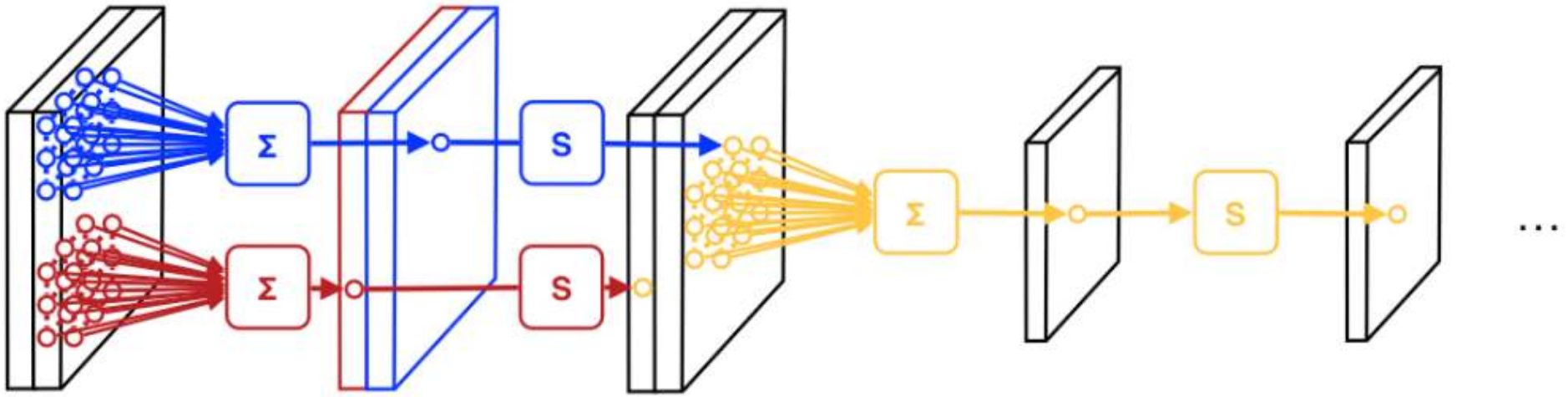
THE SAME NEURON IS FIRED WITH DIFFERENT AREAS FROM THE INPUT



Credit

DOWNSAMPLING

DOWNSAMPLING IS APPLIED TO REDUCE THE OVERALL SIZE OF TENSORS



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 $y = max(x_{uv})$

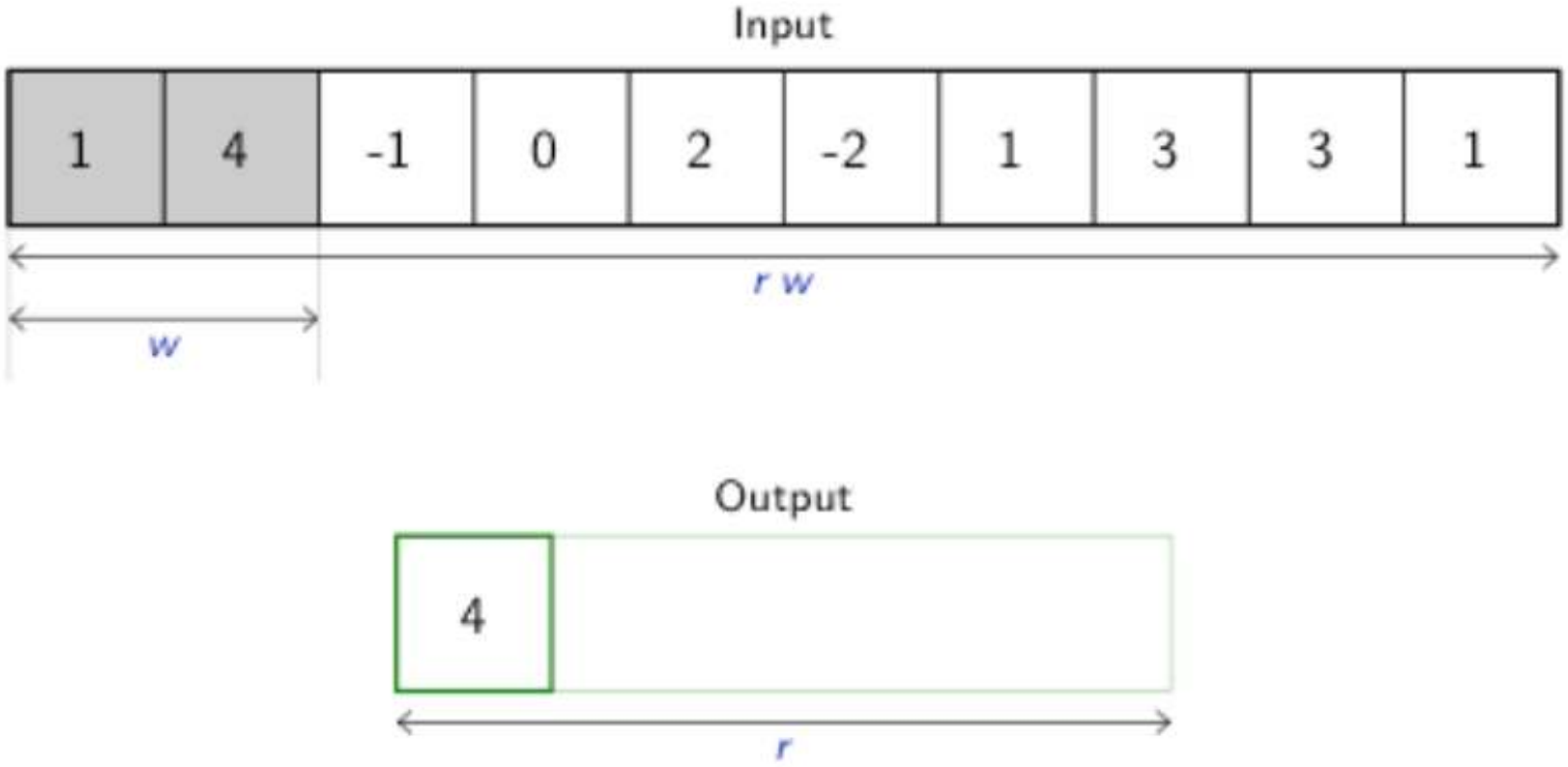
TYPES OF POOLING

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

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

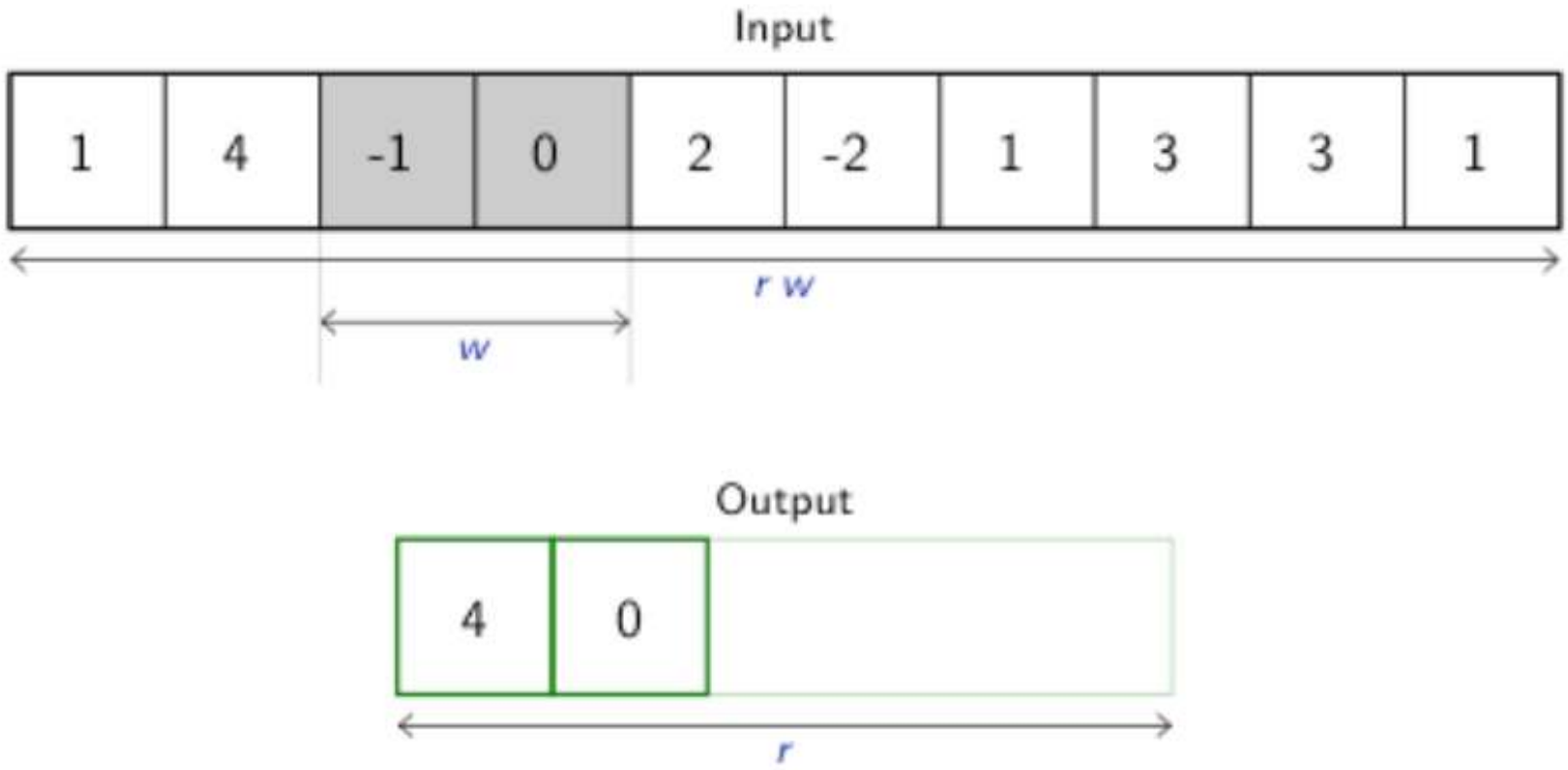
MAX POOLING $y = max(x_{uv})$

MAX POOLING 1D



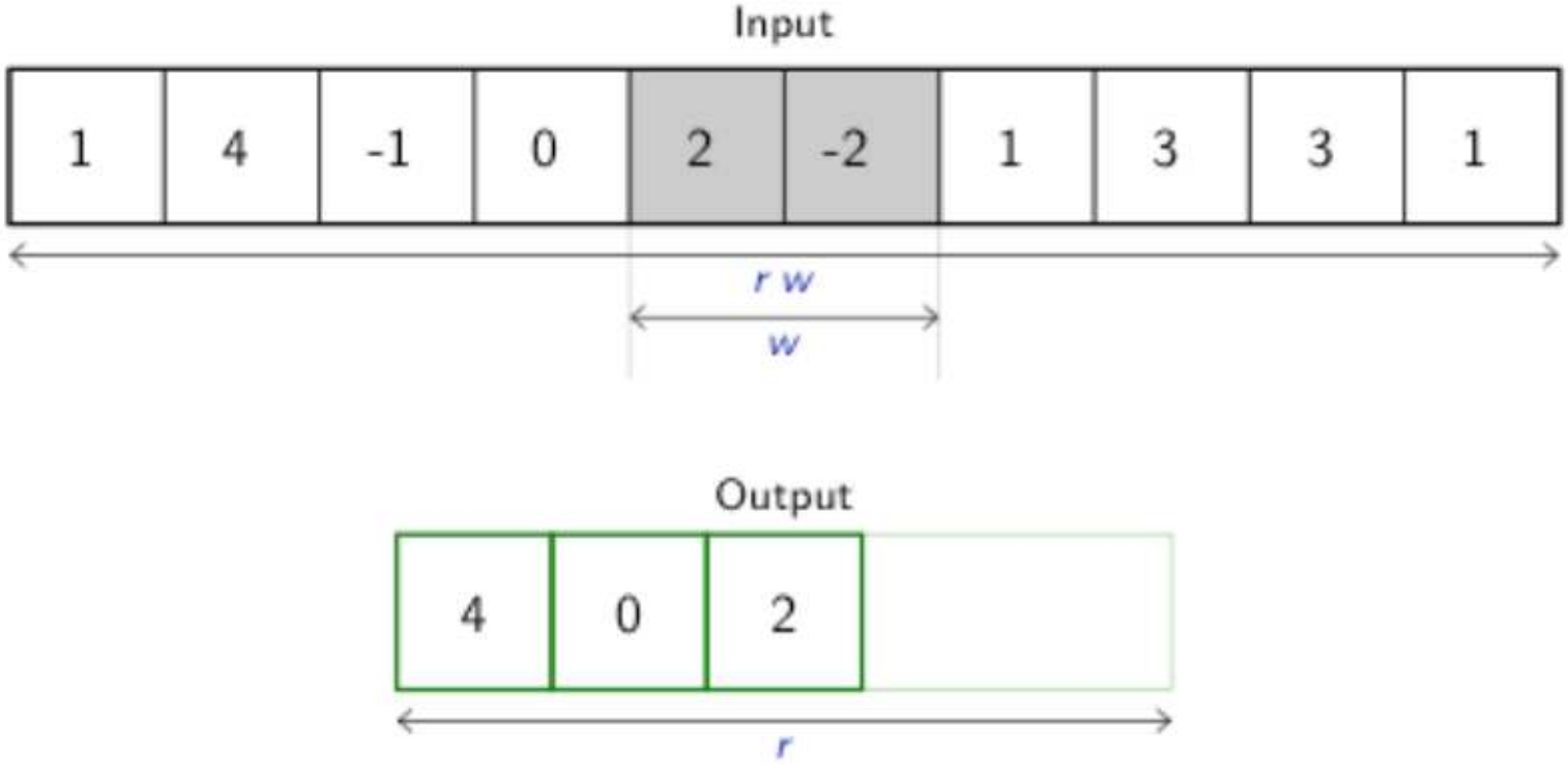
Credit: F. Fleuret

MAX POOLING 1D



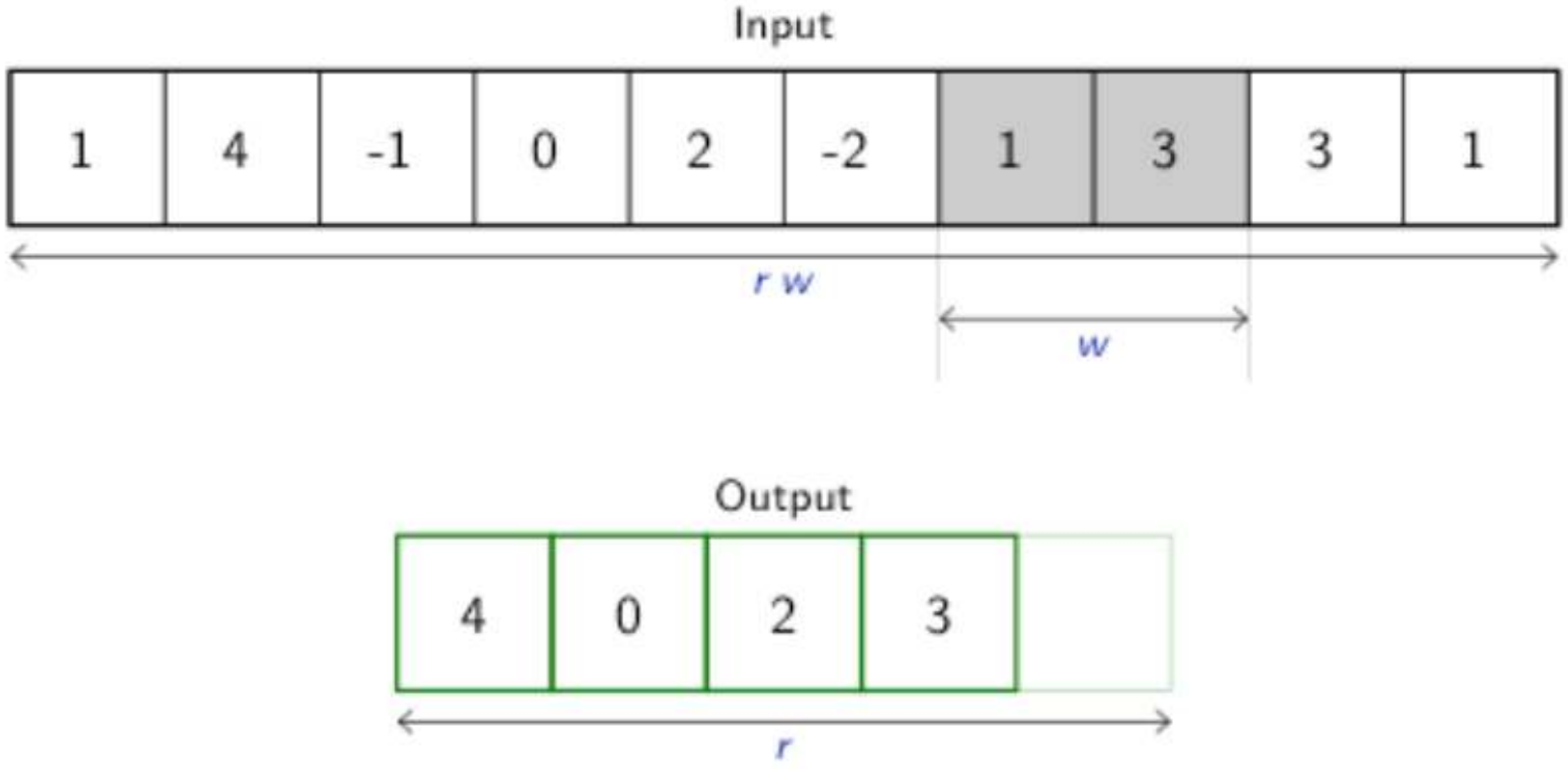
Credit: F. Fleuret

MAX POOLING 1D



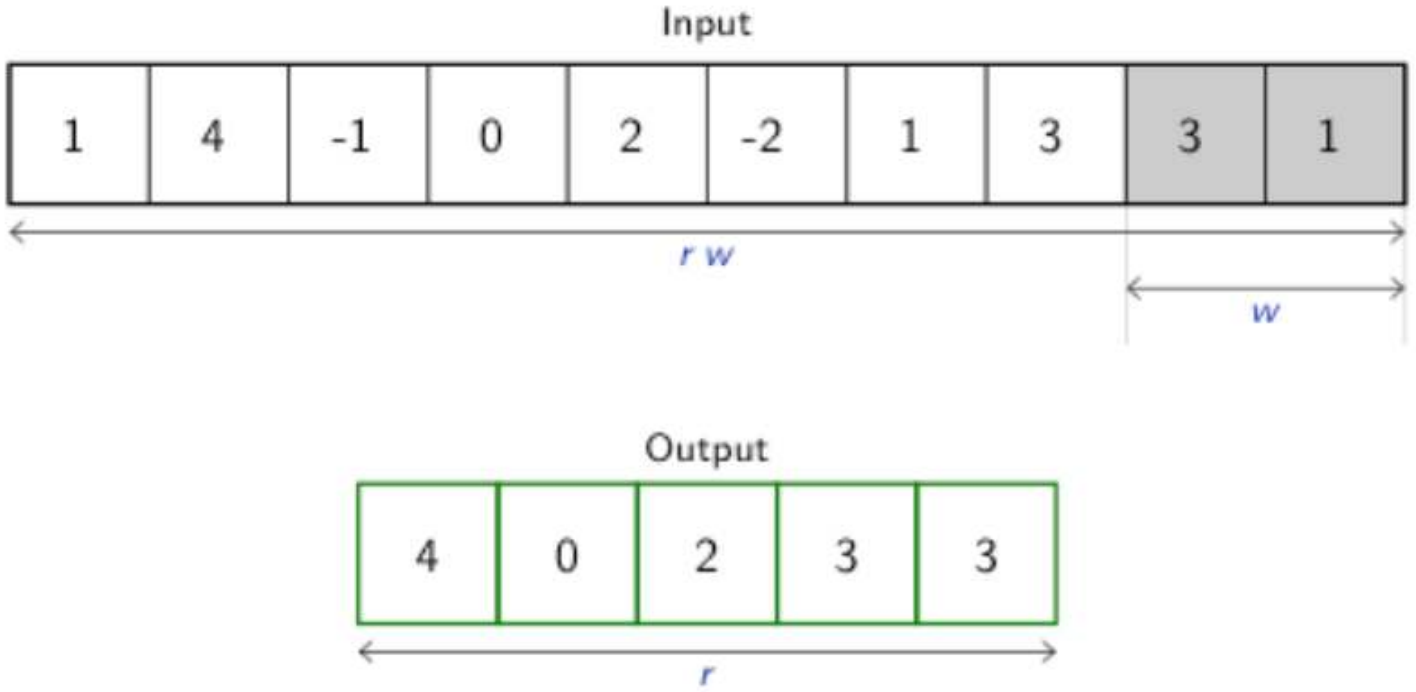
Credit: F. Fleuret

MAX POOLING 1D



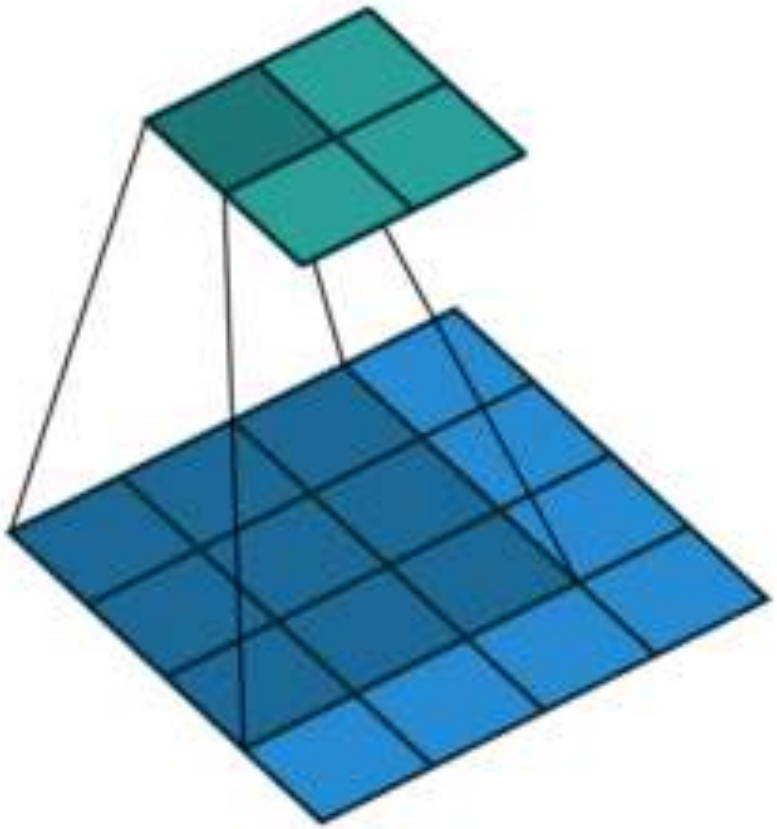
Credit: F. Fleuret

MAX POOLING 1D

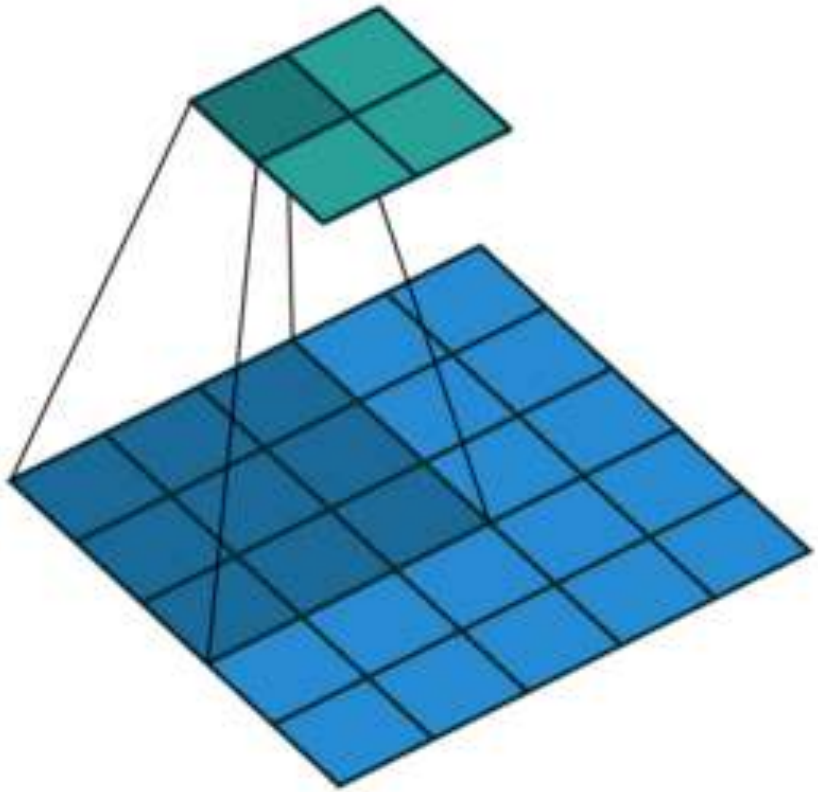


Credit: F. Fleuret

OPTIONS: **STRIDES**

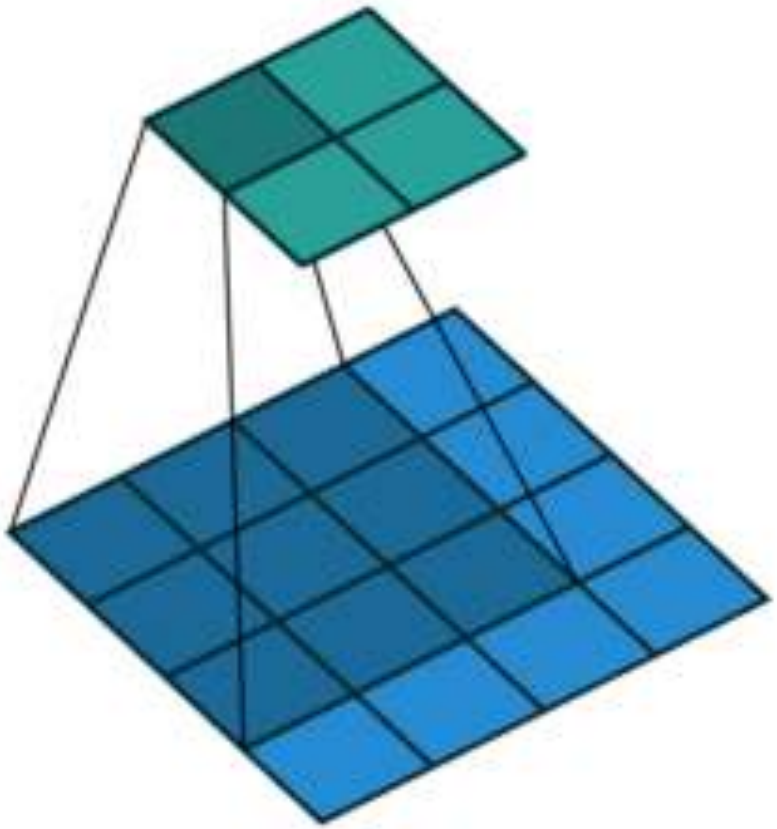


NO STRIDES

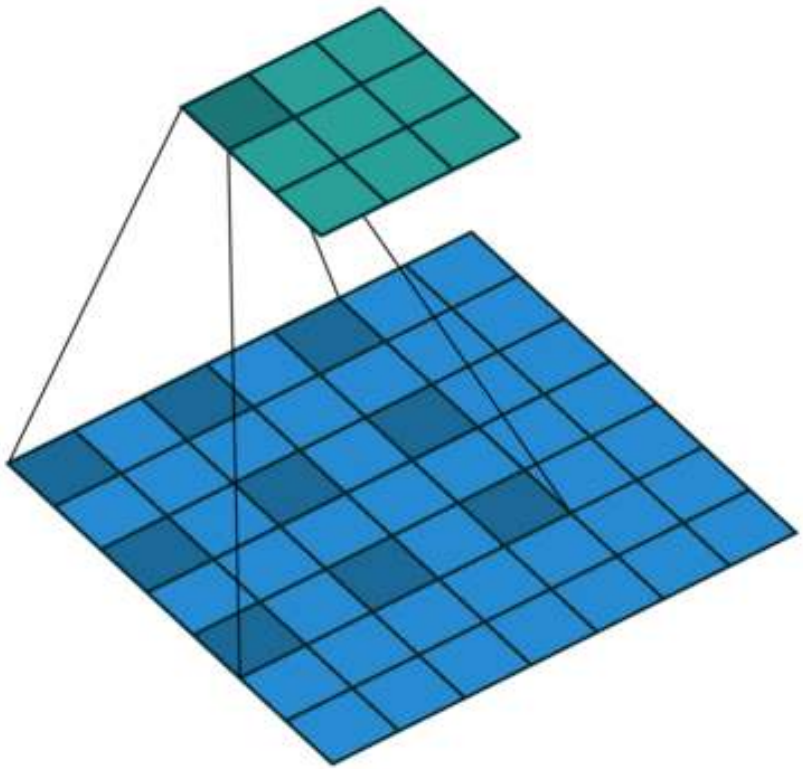


STRIDES

OPTIONS: **DILATION**

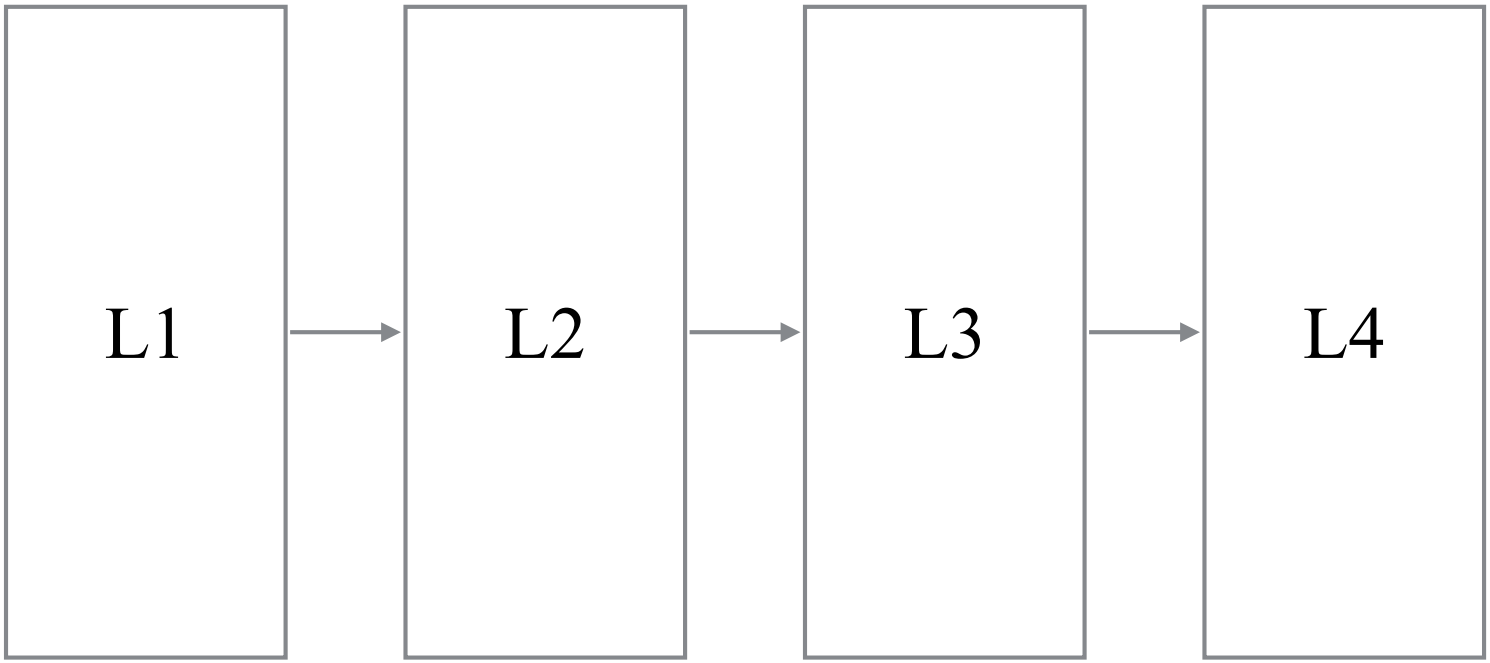


NO STRIDES



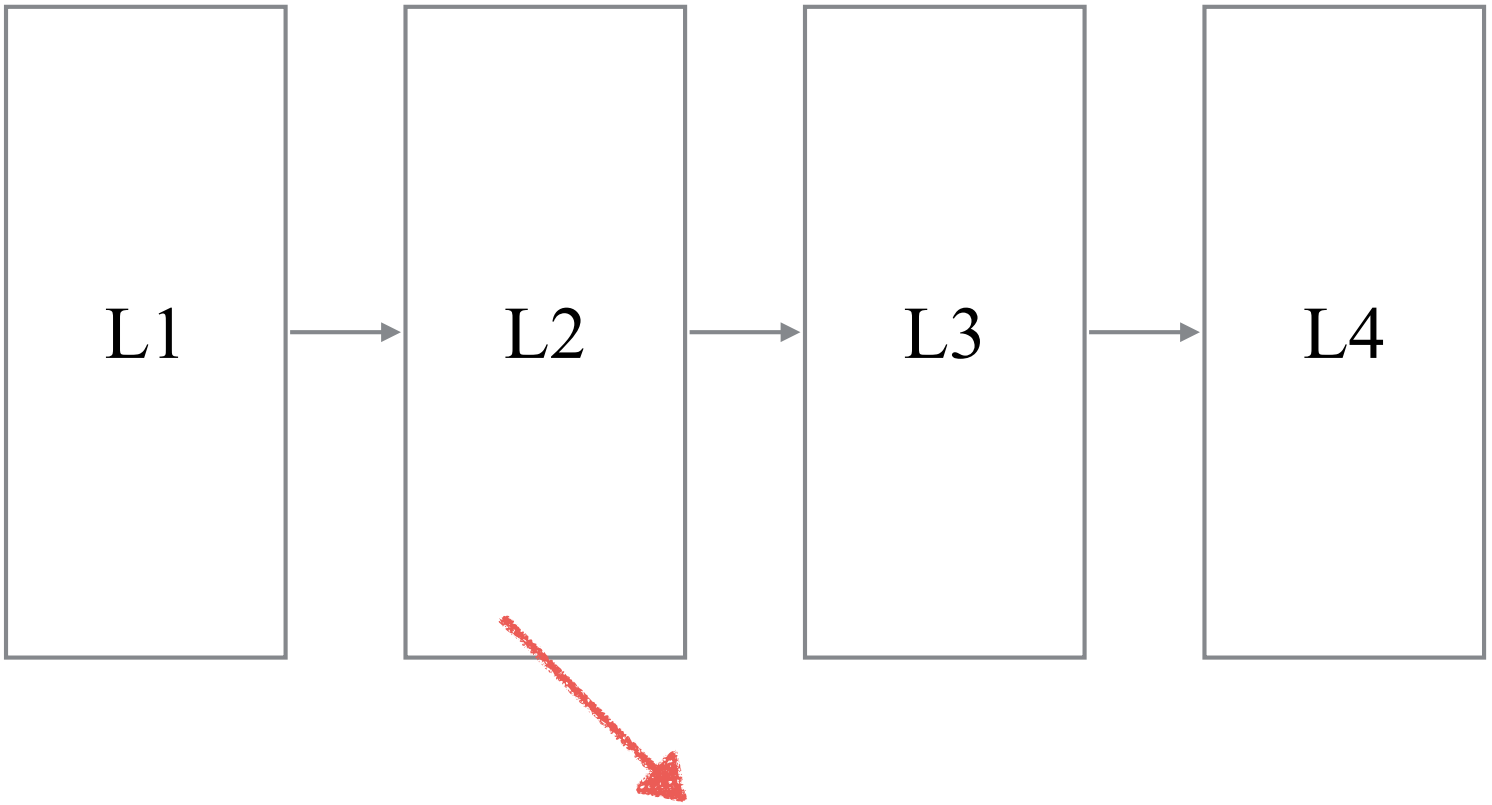
DILATION

CONVNET OR CNN



A CONCATENATION OF MULTIPLE CONVOLUTIONAL BLOCKS

CONVNET OR CNN

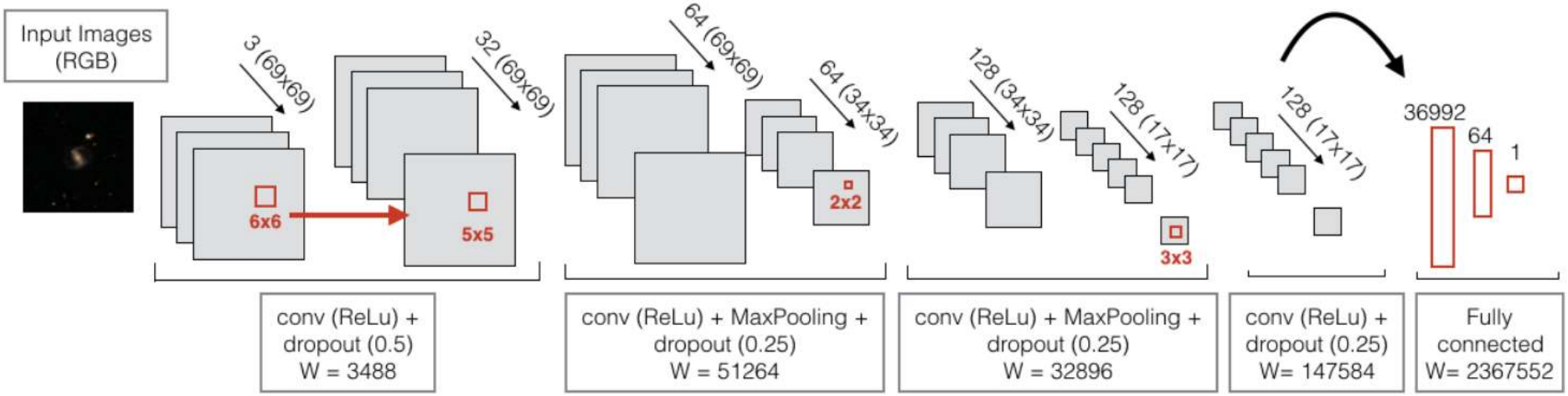


EACH BLOCK TYPICALLY MADE OF:

CONV | ACTIVATION | POOLING

(+dropout for training)

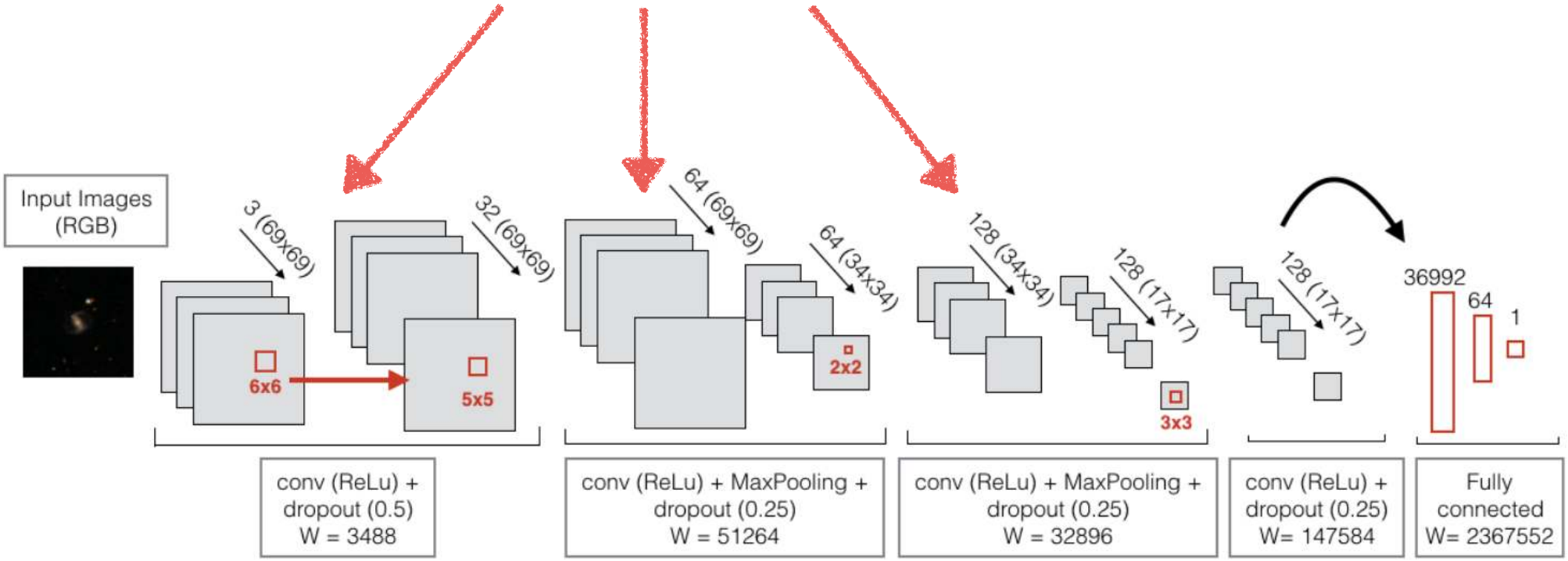
EXAMPLE OF VERY SIMPLE CNN



Dominguez-Sanchez+18

EXAMPLE OF VERY SIMPLE CNN

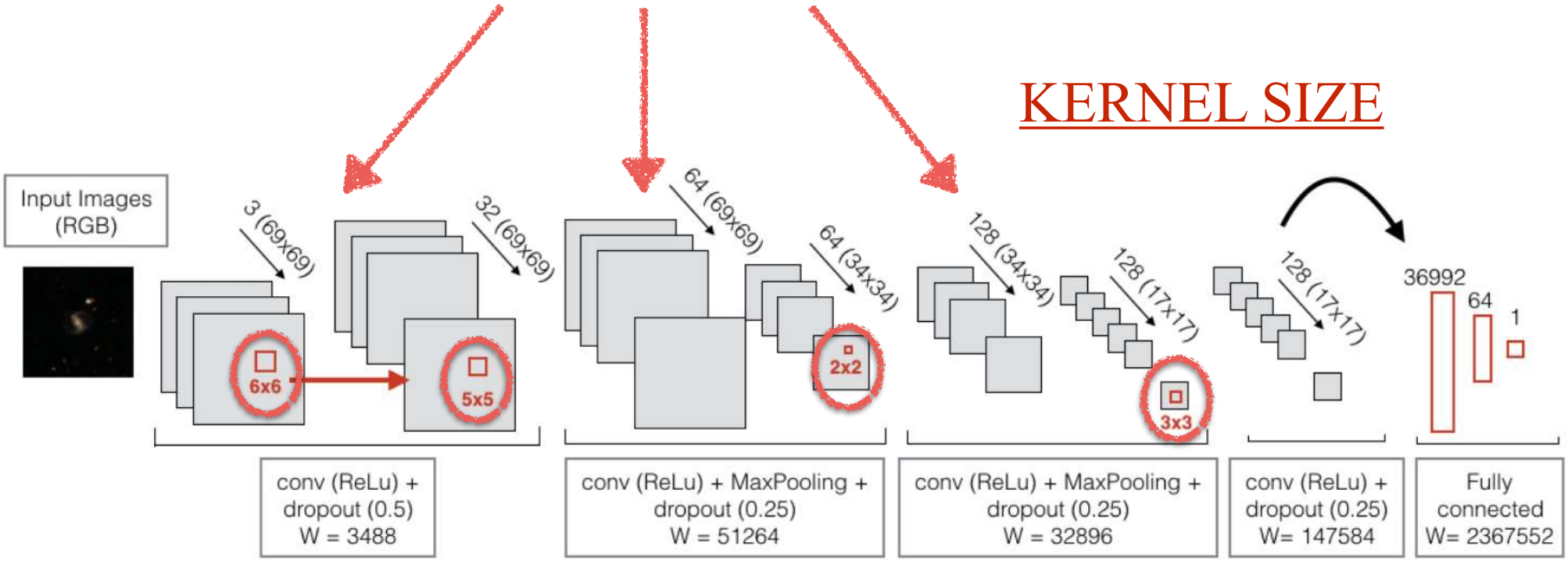
3 convolutional layers



Dominguez-Sanchez+18

EXAMPLE OF VERY SIMPLE CNN

3 convolutional layers

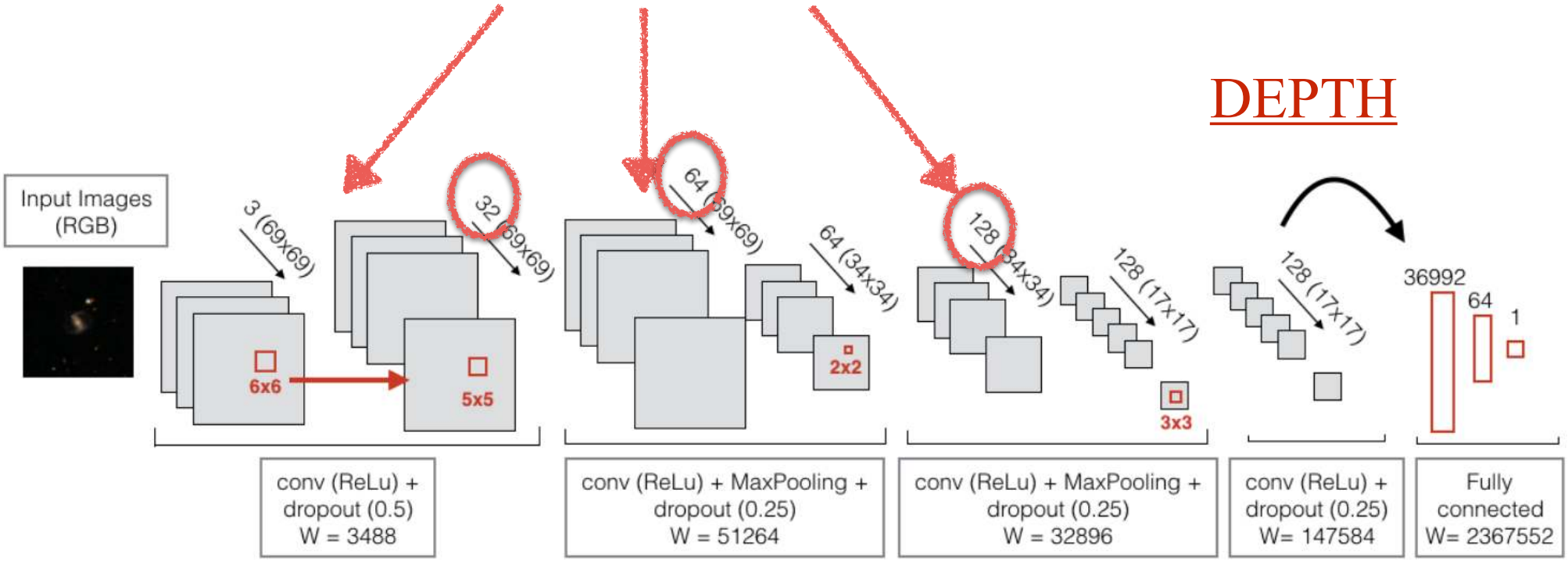


Dominguez-Sanchez+18

EXAMPLE OF VERY SIMPLE CNN

3 convolutional layers

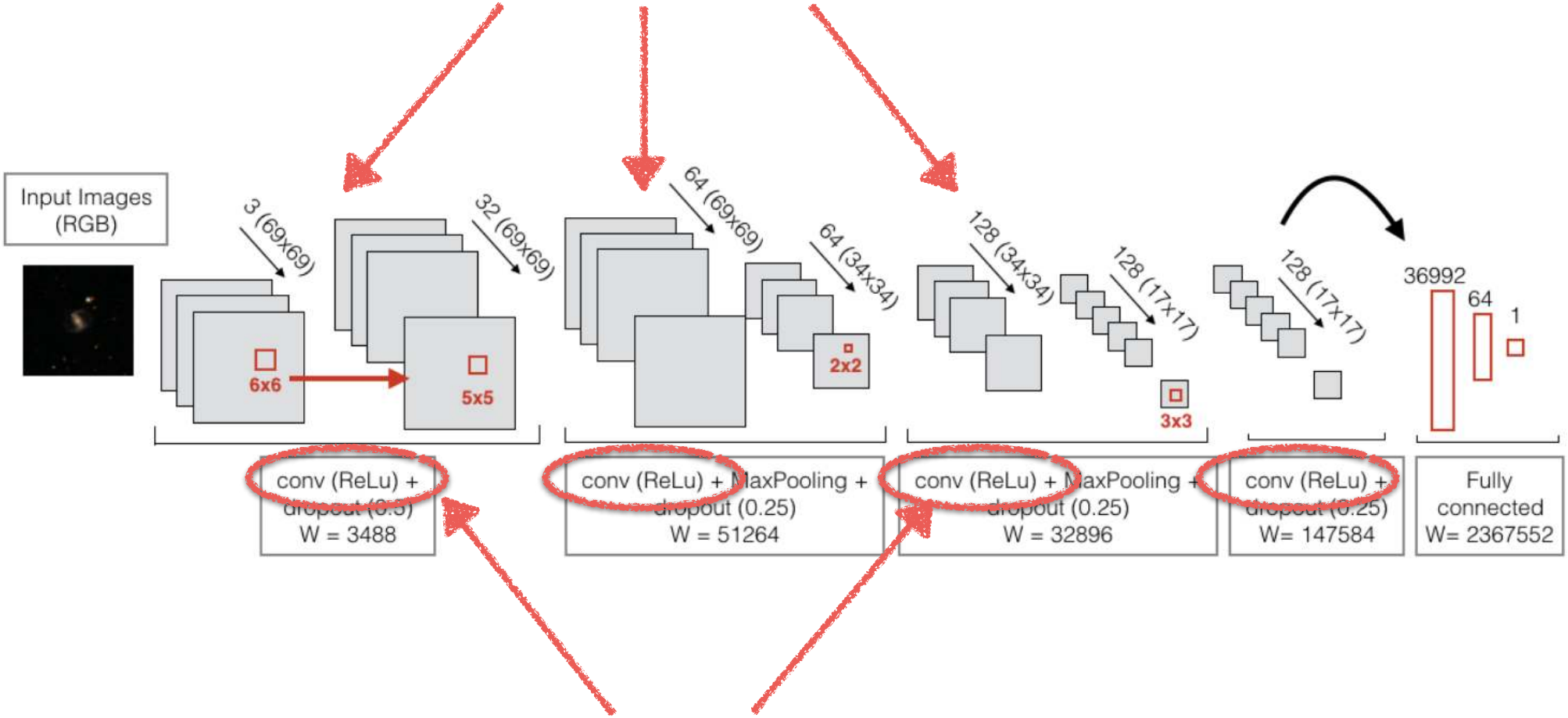
DEPTH



Dominguez-Sanchez+18

EXAMPLE OF VERY SIMPLE CNN

3 convolutional layers

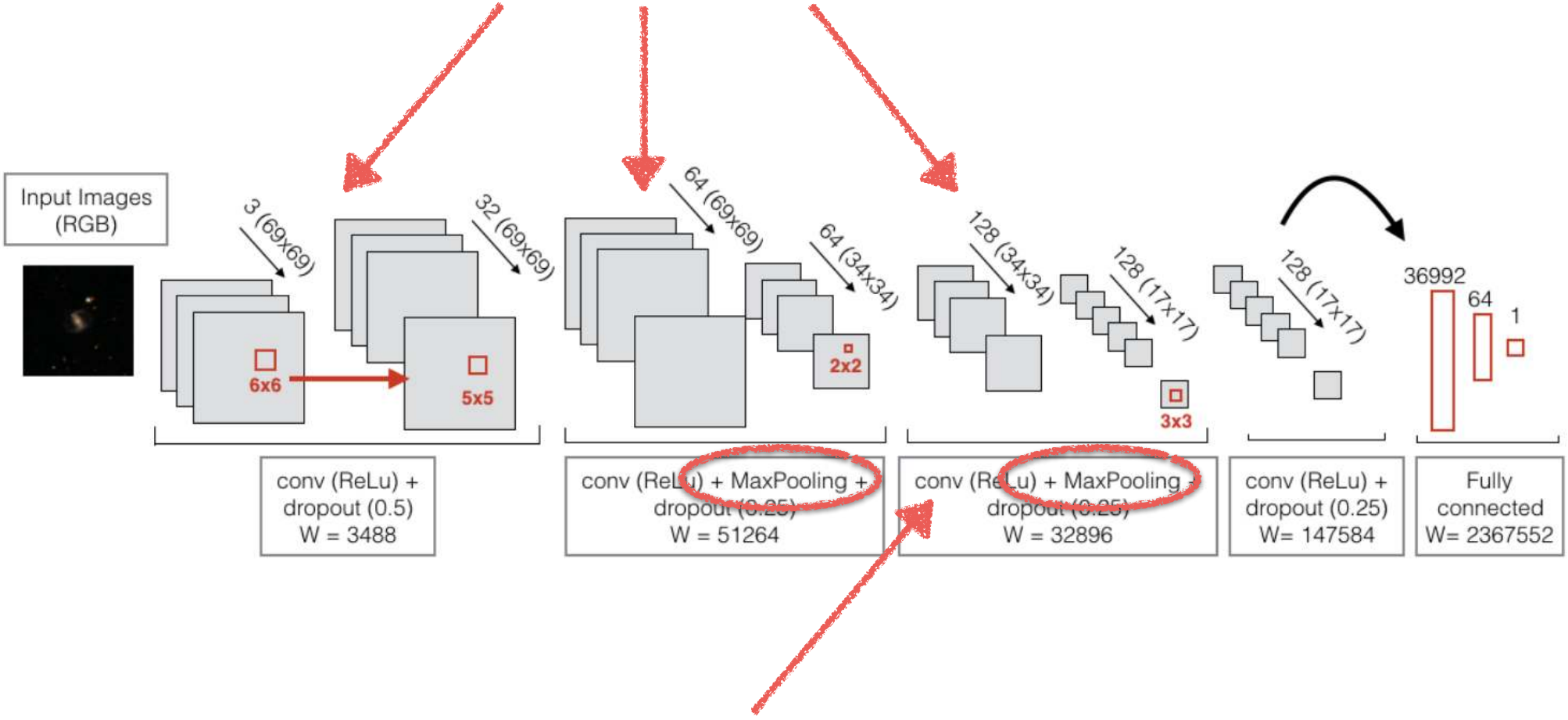


ReLu activation

Dominguez-Sanchez+18

EXAMPLE OF VERY SIMPLE CNN

3 convolutional layers



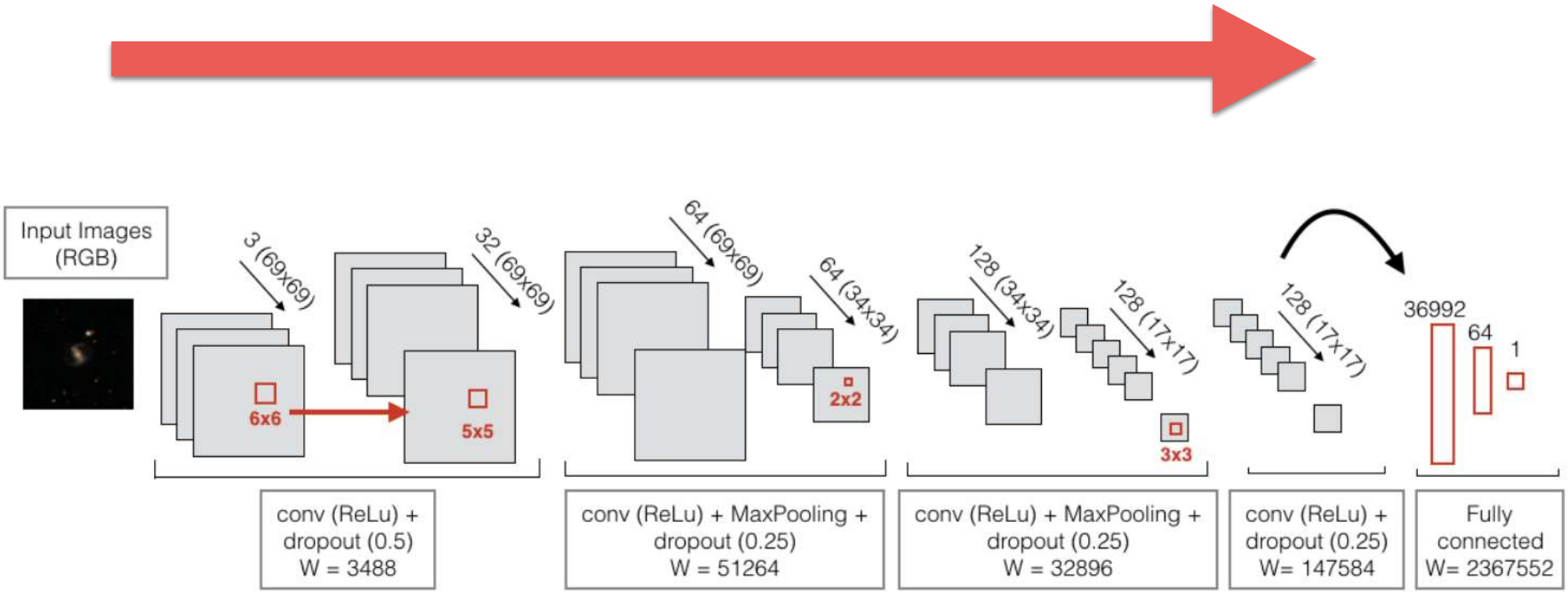
Pooling

Dominguez-Sanchez+18

EXAMPLE OF VERY SIMPLE CNN

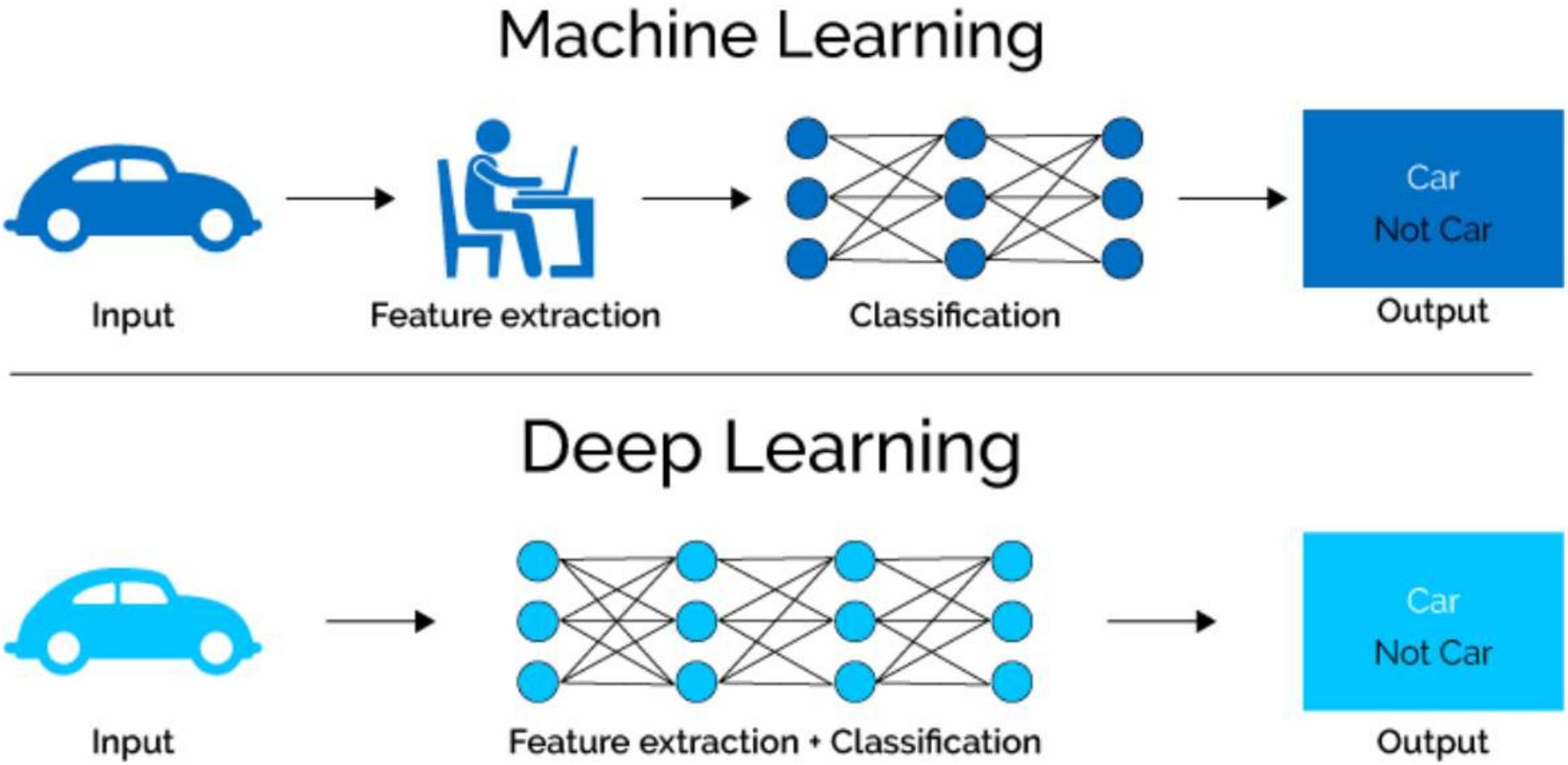
OVERALL:

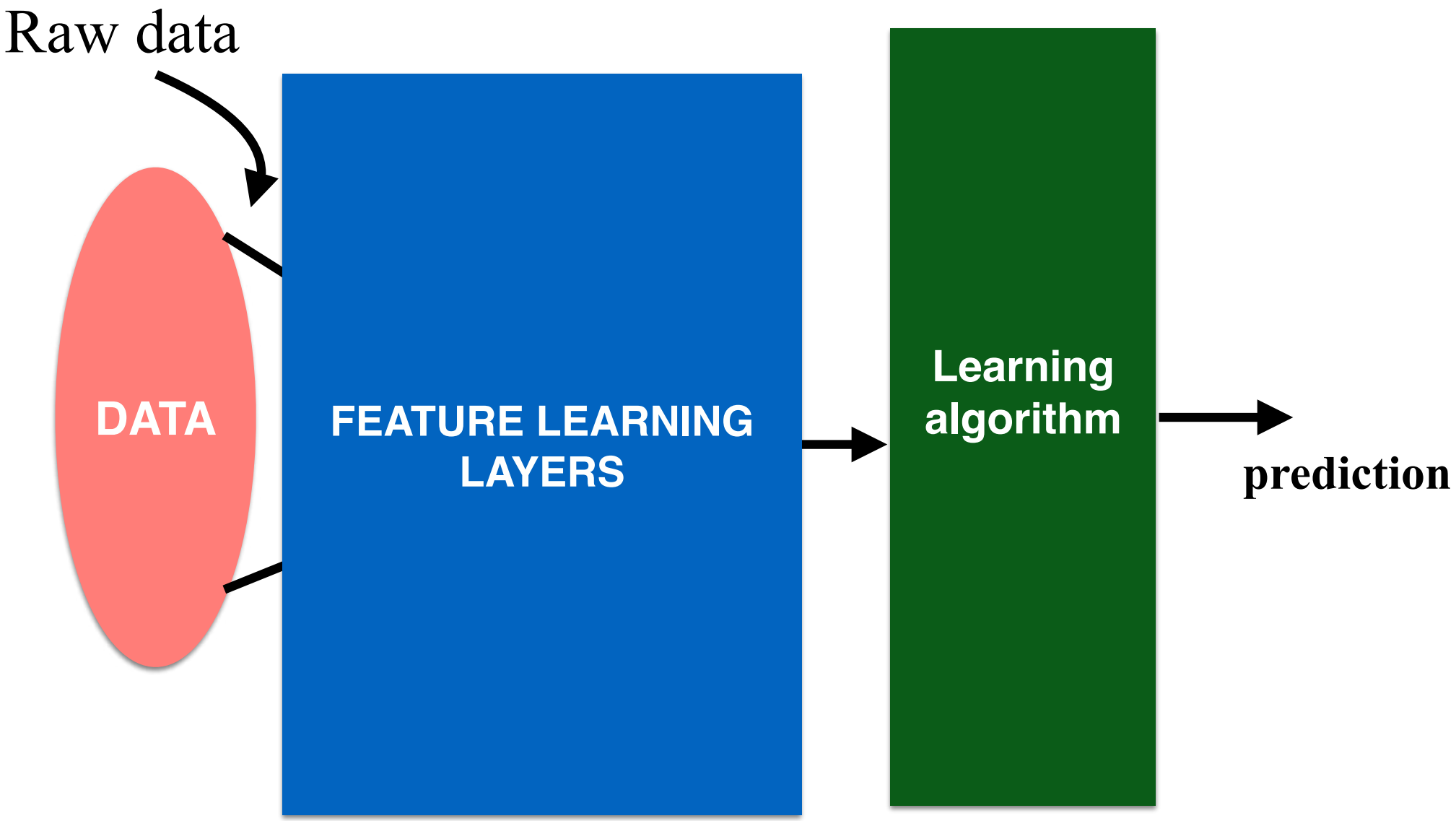
- decrease of tensor size
- increase of depth



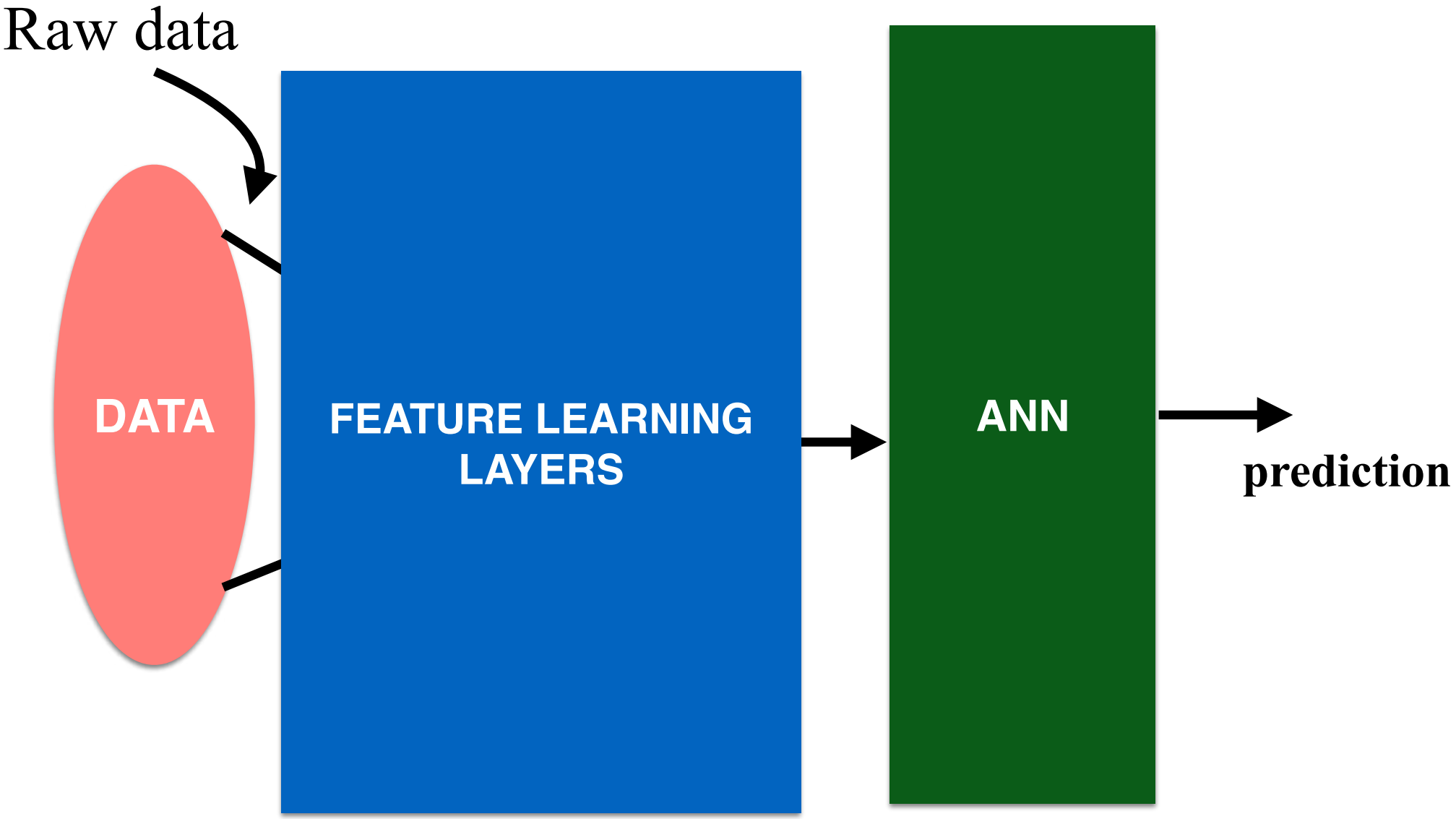
Dominguez-Sanchez+18

THIS IS A CHANGE OF PARADIGM!

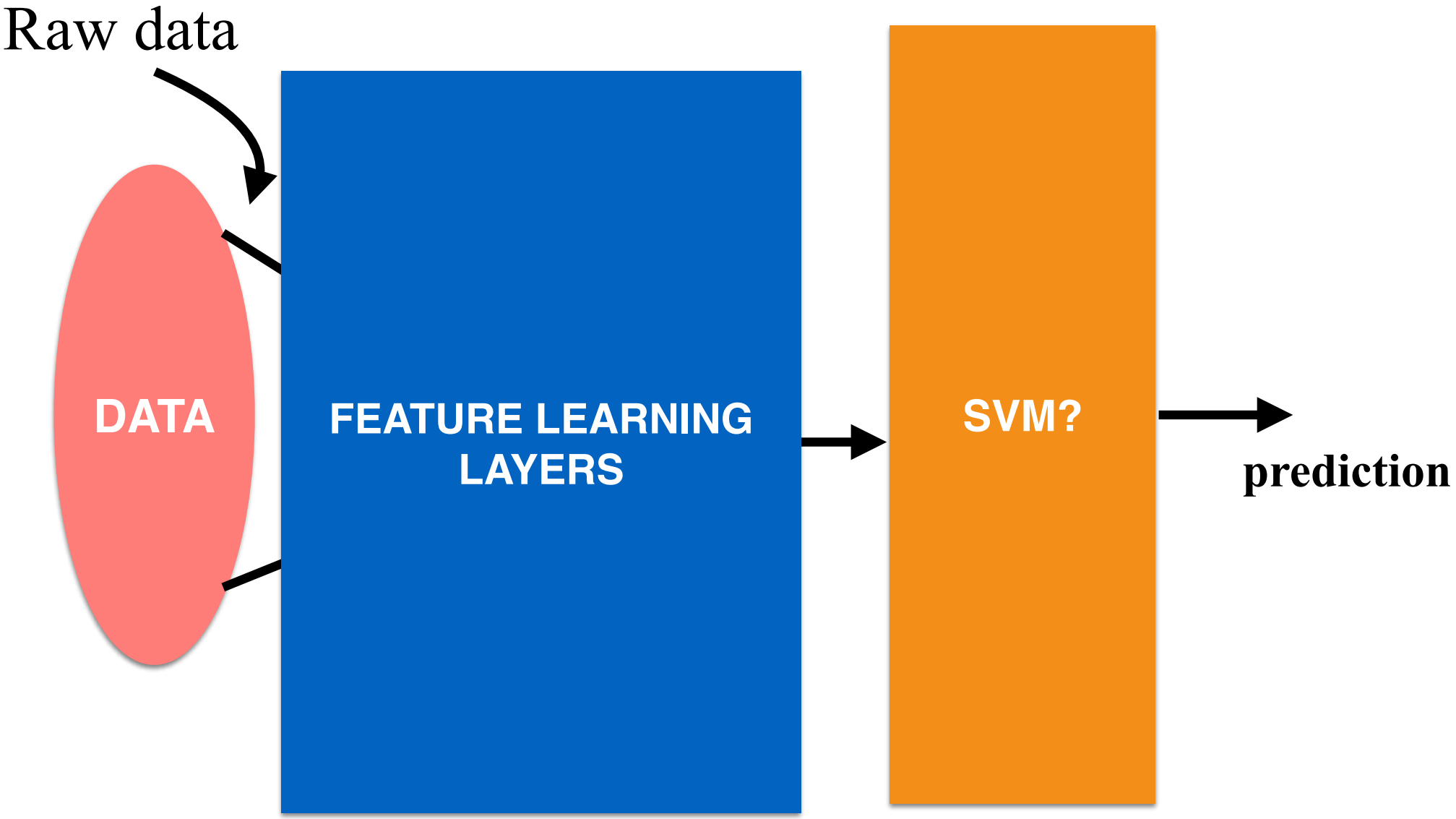



Raw data
DATA
FEATURE LEARNING LAYERS
Learning algorithm
prediction

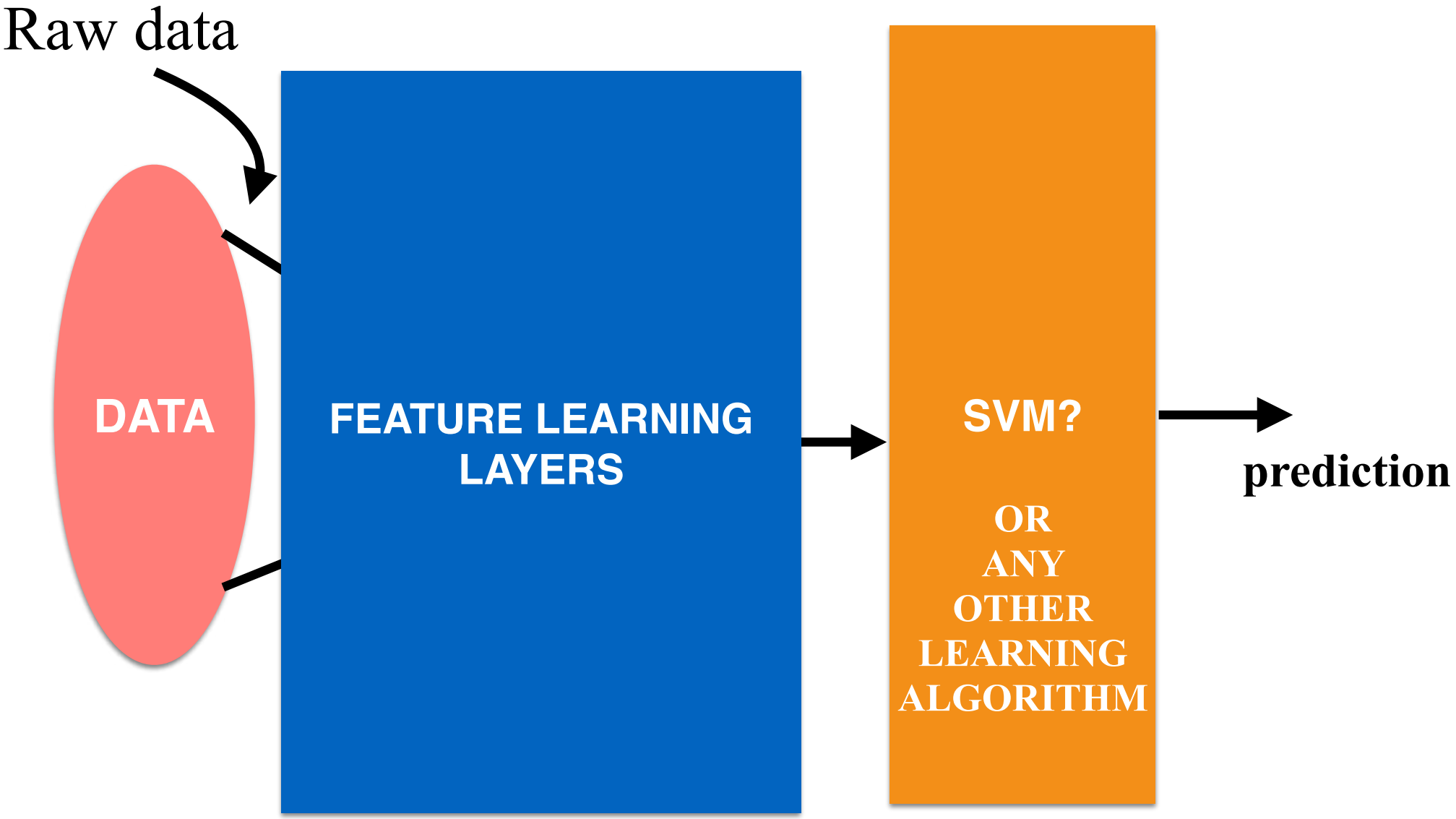
THE LEARNING ALGORITHM CAN BE CHANGED



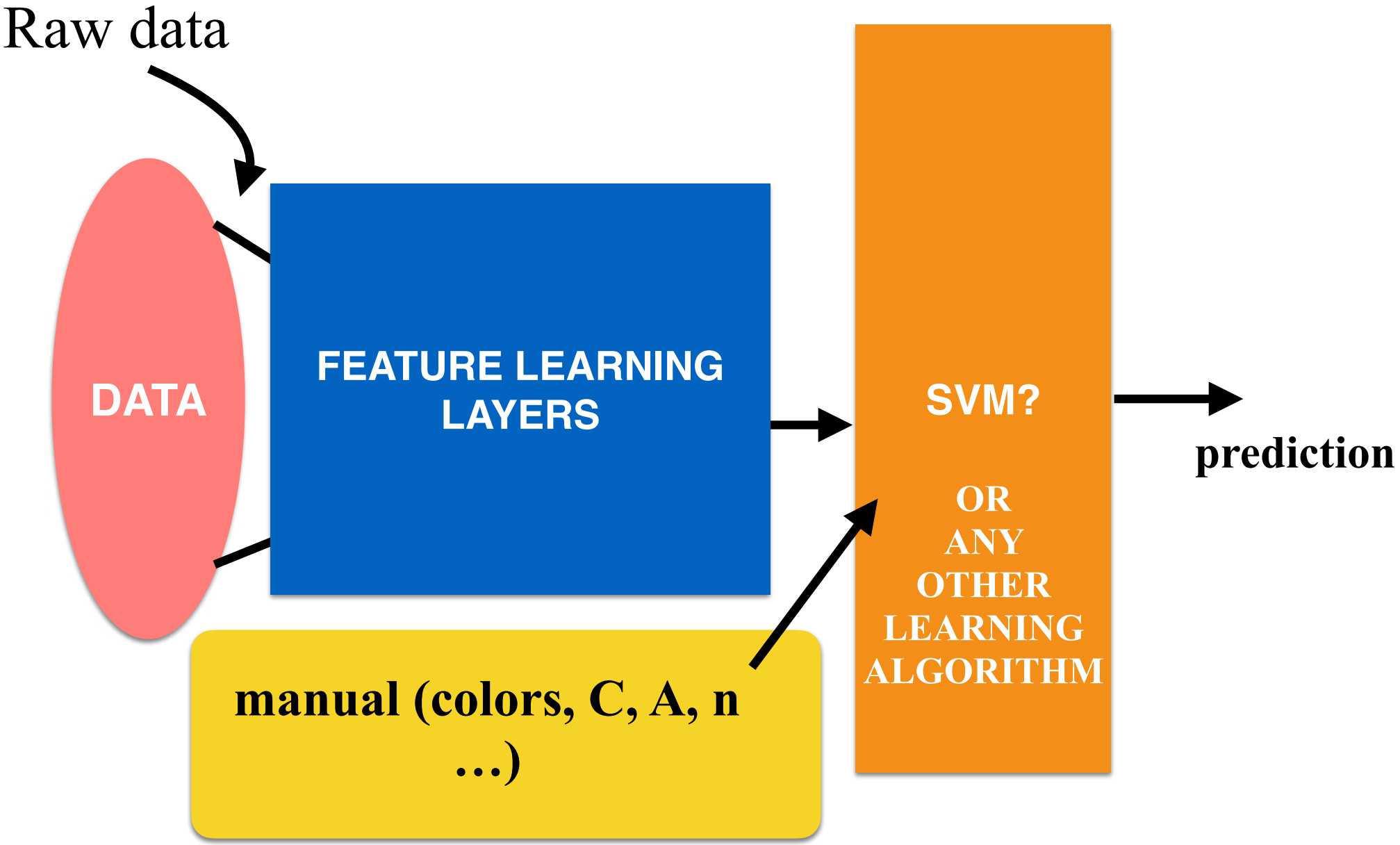
THE LEARNING ALGORITHM CAN BE CHANGED

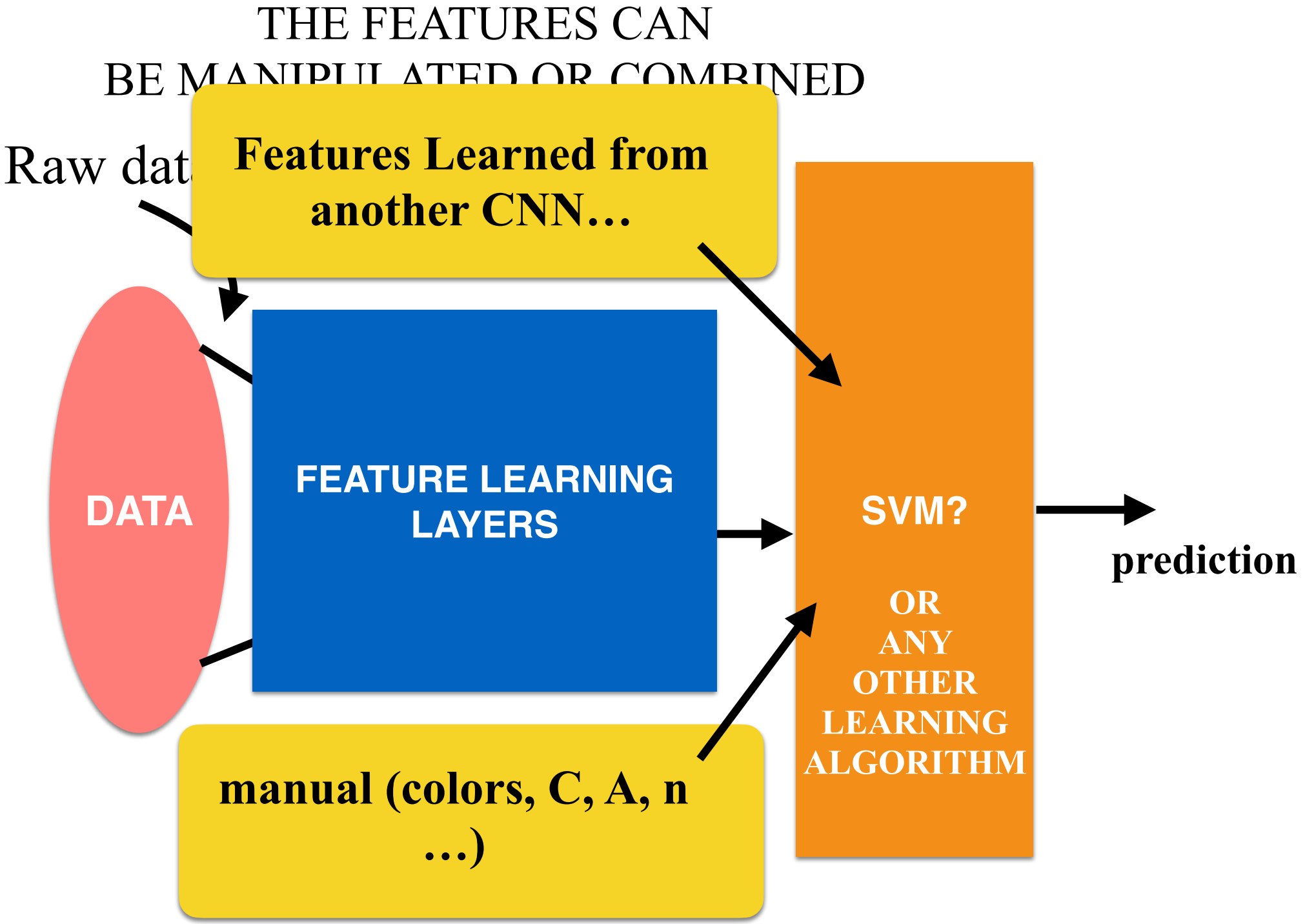


THE LEARNING ALGORITHM CAN BE CHANGED

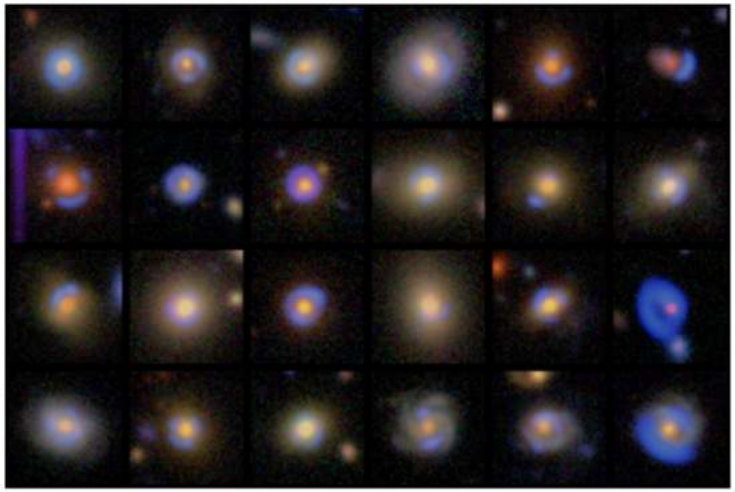


THE FEATURES CAN BE MANIPULATED OR COMBINED

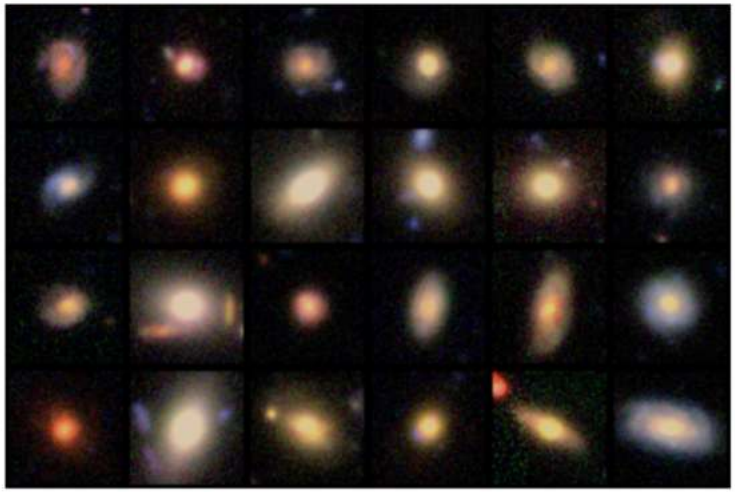



THE FEATURES CAN
BE MANIPULATED OR COMBINED
Raw dat
Features Learned from another CNN…
DATA
FEATURE LEARNING LAYERS
SVM?
OR ANY OTHER LEARNING ALGORITHM
prediction
manual (colors, C, A, n …)

1. Classification



LENS



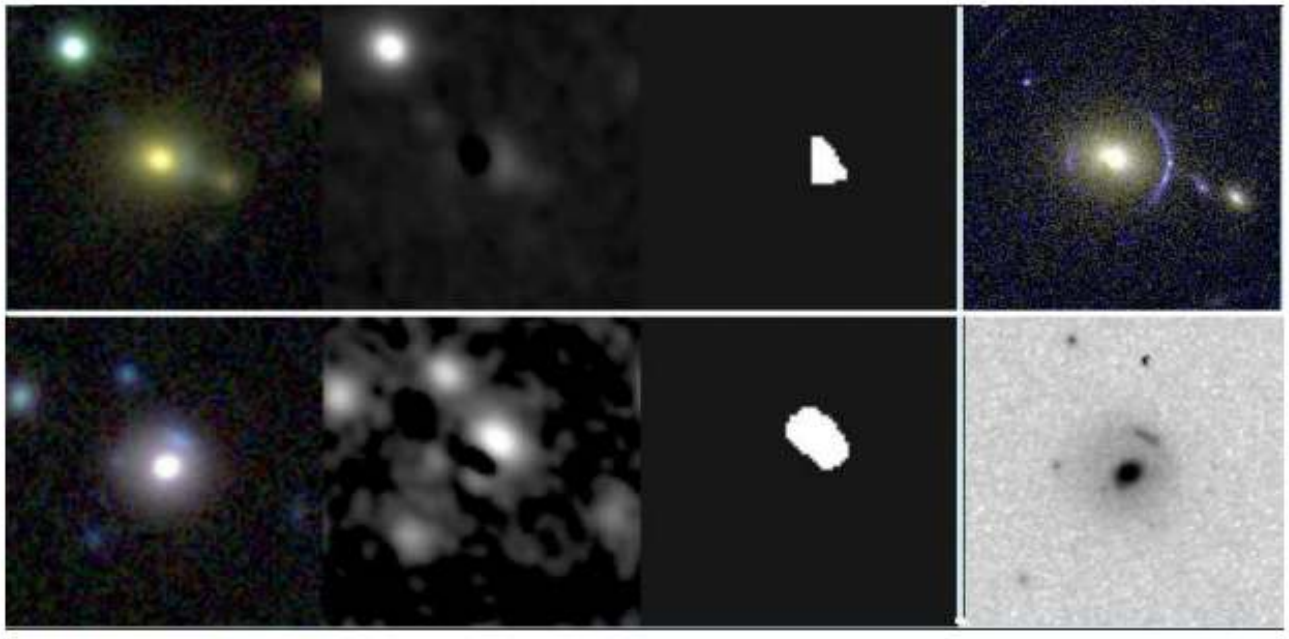
NON-LENS

Detection of Strong Lenses
Valuable information of
Dark Matter properties

Future surveys will increase the samples by orders of magnitude.

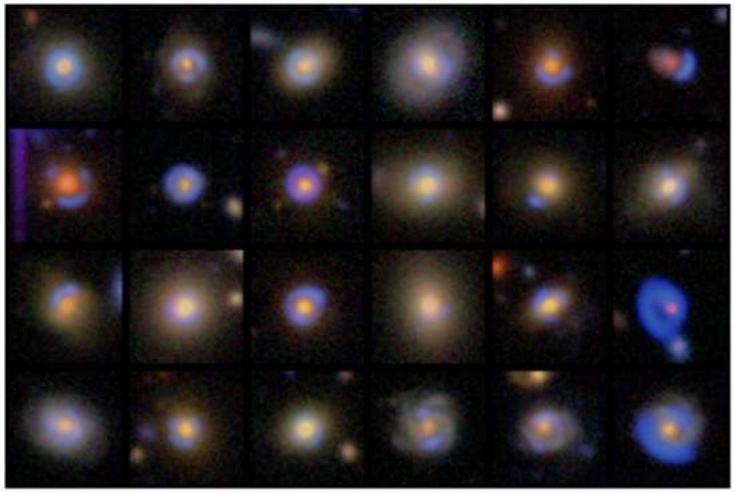
Jacobs+17

"Pre Deep Learning" Approach

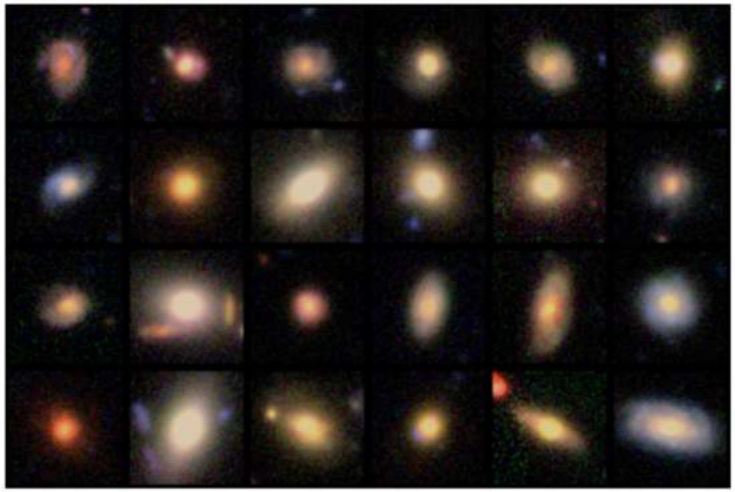


Gavazzi+17

1. Classification

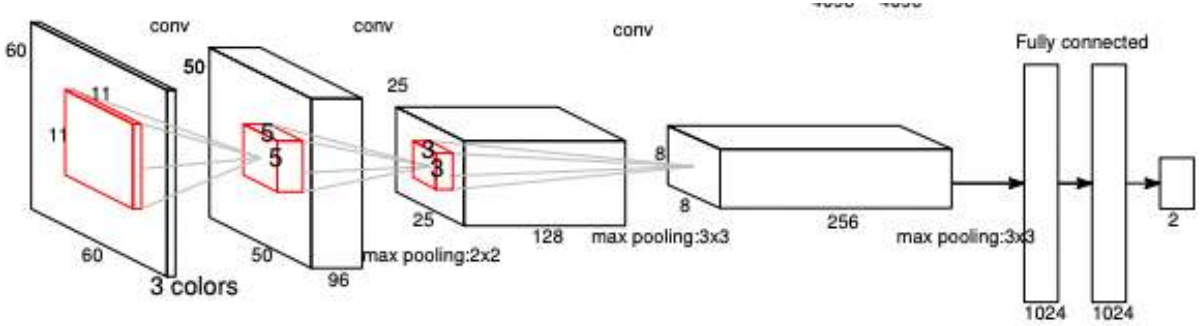


LENS

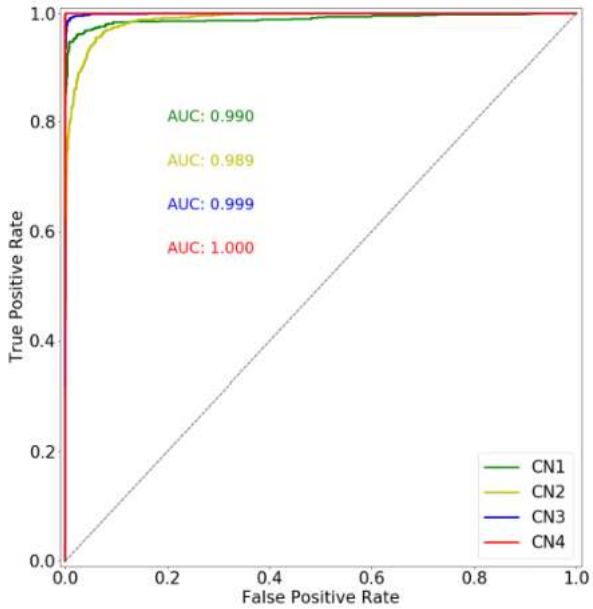


NON-LENS

Jacobs+17



It illustrates the change of paradigm from an algorithmic centric focus to a purely data driven approach to data



Dieleman+15, Huertas-Company+15, Aniyan+17, Charnock+17, Gieseke+17, Jacobs+17, Petrillo+17, Schawinski+17, Alhassan+18, Dominguez-Sanchez+18, George+18, Hinners+18, Lukic+18, Moss+18, Razzano+18, Schaefer+18, Allen+19, Burke+19, Carrasco-Davis+19, Chatterjee+19, Davies+19, Dominguez-Sanchez+19, Fusse+19, Glaser+19, Ishida+19, Jacobs+19, Katebi+19, Lanusse+19Liu+19, Lukic+19, Metcalf+19, Muthukrishna+19, Petrillo+19, Reiman+19, Boucaud+20, Chiani+20, Ghosh+20, Gomez+20, Hausen+20, Hlozek+20, Hosseinzadeh+20. Huang+20, Li+20, Moller+20, Paillassa+20, Tadaki+20, Vargas dos Santos+20, Walmsley+20, Wei+20, Allam+21, Arcelin+21, Becker+21, Bretonniere+21, Burhanudin+21, Davison+21, Donoso-Oliva+21, Jia+21, Lauritsen+21, Ono+21, Ruan+21, Sadegho+21, Tang+21, Tanoglidis+21, Vojtekova+21, Dhar+22Hausen+22, Orwat+22, Pimentel+22, Rezaei+22, Samudre+22, Shen+22, Walmsley+22

1. Classification

CNN based classifications reach unprecedented accuracy

Name	type	AUROC	TPR_0	TPR_{10}	short description
CMU-DeepLens-Resnet-ground3	Ground-Based	0.98	0.09	0.45	CNN
CMU-DeepLens-Resnet-Voting	Ground-Based	0.98	0.02	0.10	CNN
LASTRO EPFL	Ground-Based	0.97	0.07	0.11	CNN
CAS Swinburne Melb	Ground-Based	0.96	0.02	0.08	CNN
AstrOmatic	Ground-Based	0.96	0.00	0.01	CNN
Manchester SVM	Ground-Based	0.93	0.22	0.35	SVM / Gabor
Manchester2	Ground-Based	0.89	0.00	0.01	Human Inspection
ALL-star	Ground-Based	0.84	0.01	0.02	edges/gradiants and Logistic Reg.
CAST	Ground-Based	0.83	0.00	0.00	CNN / SVM
YattaLensLite	Ground-Based	0.82	0.00	0.00	SExtractor
LASTRO EPFL	Space-Based	0.93	0.00	0.08	CNN
CMU-DeepLens-Resnet	Space-Based	0.92	0.22	0.29	CNN
GAMOCLASS	Space-Based	0.92	0.07	0.36	CNN
CMU-DeepLens-Resnet-Voting	Space-Based	0.91	0.00	0.01	CNN
AstrOmatic	Space-Based	0.91	0.00	0.01	CNN
CMU-DeepLens-Resnet-aug	Space-Based	0.91	0.00	0.00	CNN
Kapteyn Resnet	Space-Based	0.82	0.00	0.00	CNN
CAST	Space-Based	0.81	0.07	0.12	CNN
Manchester1	Space-Based	0.81	0.01	0.17	Human Inspection
Manchester SVM	Space-Based	0.81	0.03	0.08	SVM / Gabor
NeuralNet2	Space-Based	0.76	0.00	0.00	CNN / wavelets
YattaLensLite	Space-Based	0.76	0.00	0.00	Arcs / SExtractor
All-now	Space-Based	0.73	0.05	0.07	edges/gradiants and Logistic Reg.
GAHEC IRAP	Space-Based	0.66	0.00	0.01	arc finder

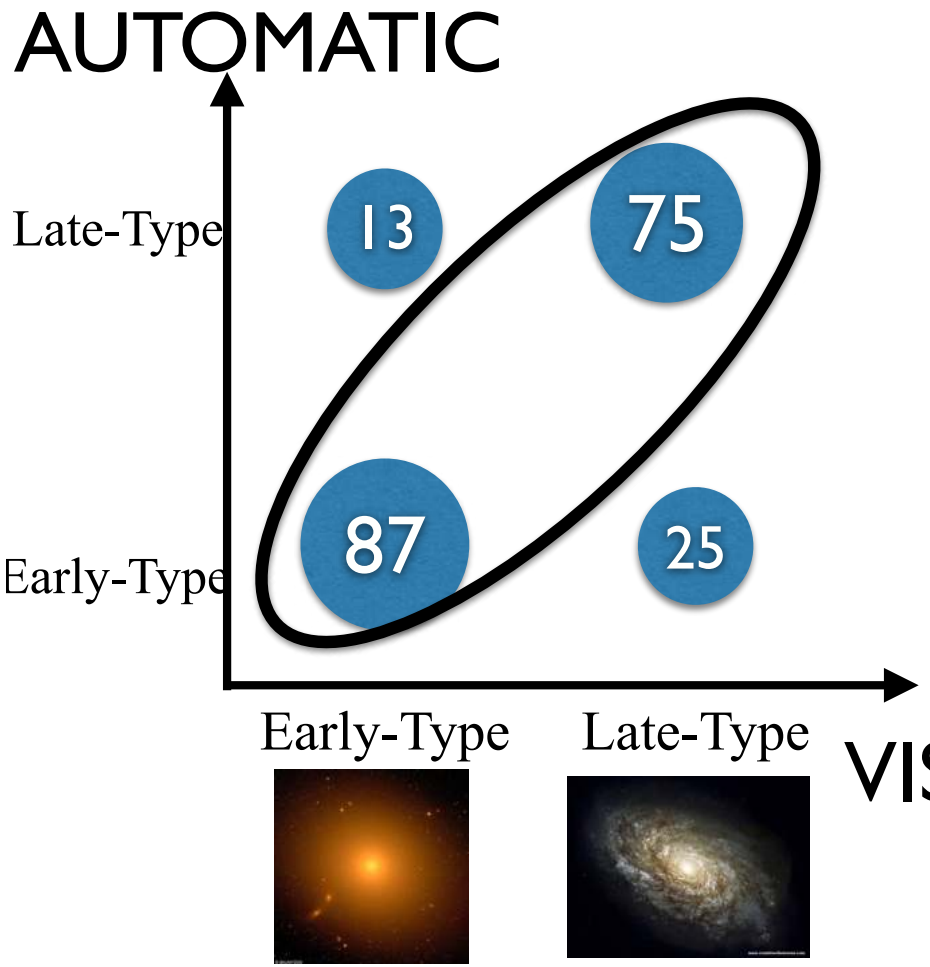
Metcalf+19

Dieleman+15, Huertas-Company+15, Aniyan+17, Charnock+17, Gieseke+17, Jacobs+17, Petrillo+17, Schawinski+17, Alhassan+18, Dominguez-Sanchez+18, George+18, Hinners+18, Lukic+18, Moss+18, Razzano+18, Schaefer+18, Allen+19, Burke+19, Carrasco-Davis+19, Chatterjee+19, Davies+19, Dominguez-Sanchez+19, Fusse+19, Glaser+19, Ishida+19, Jacobs+19, Katebi+19, Lanusse+19Liu+19, Lukic+19, Metcalf+19, Muthukrishna+19, Petrillo+19, Reiman+19, Boucaud+20, Chiani+20, Ghosh+20, Gomez+20, Hausen+20, Hlozek+20, Hosseinzadeh+20. Huang+20, Li+20, Moller+20, Paillassa+20, Tadaki+20, Vargas dos Santos+20, Walmsley+20, Wei+20, Allam+21, Arcelin+21, Becker+21, Bretonniere+21, Burhanudin+21, Davison+21, Donoso-Oliva+21, Jia+21, Lauritsen+21, Ono+21, Ruan+21, Sadegho+21, Tang+21, Tanoglidis+21, Vojtekova+21, Dhar+22Hausen+22, Orwat+22, Pimentel+22, Rezaei+22, Samudre+22, Shen+22, Walmsley+22

ALSO FOR GALAXY MORPHOLOGY

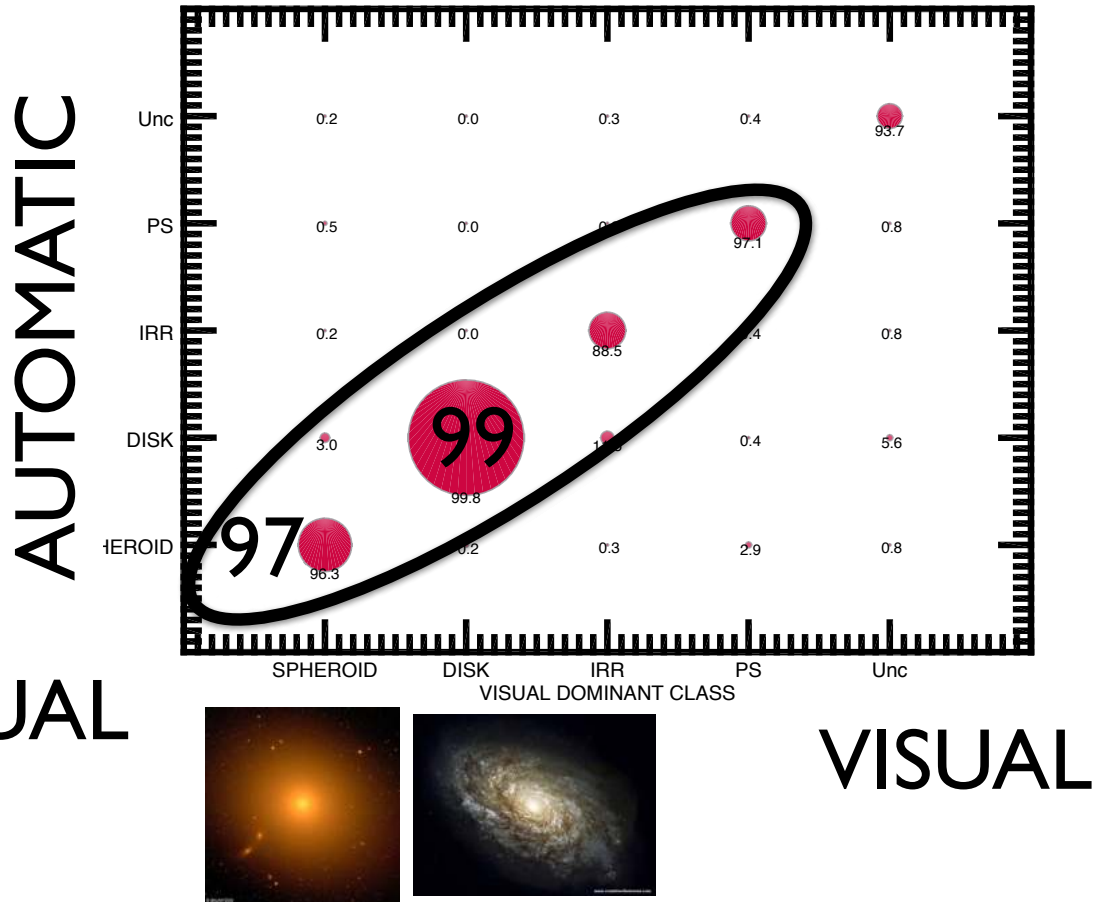
SVMs

[HUERTAS-COMPANY+14]



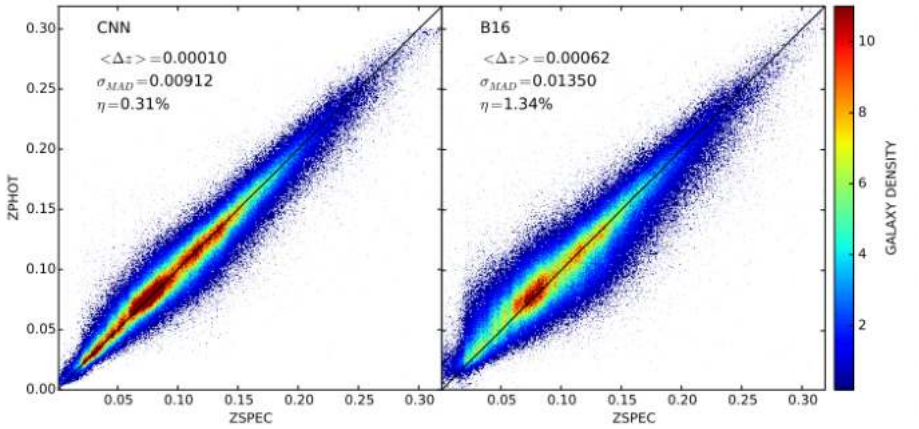
CNNs

[HUERTAS-COMPANY+15b]

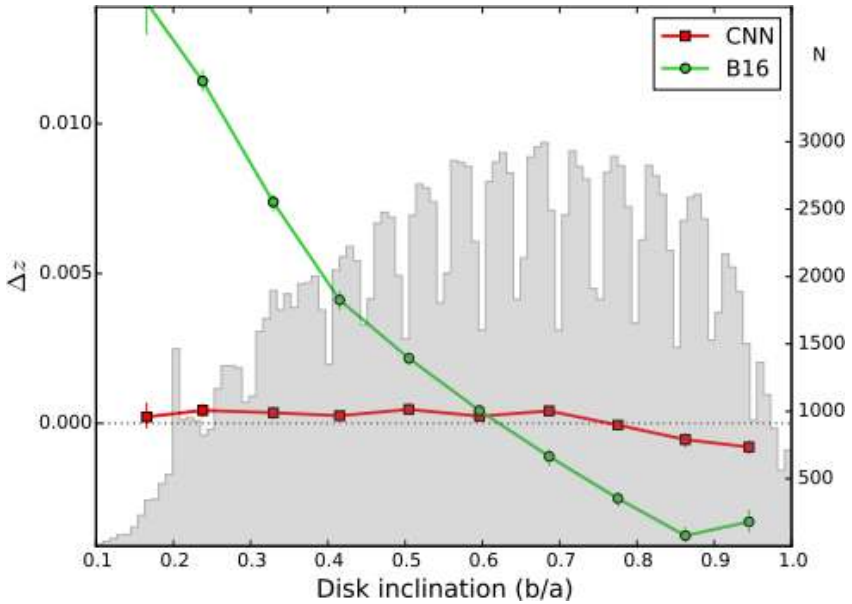


Photometric Redshifts

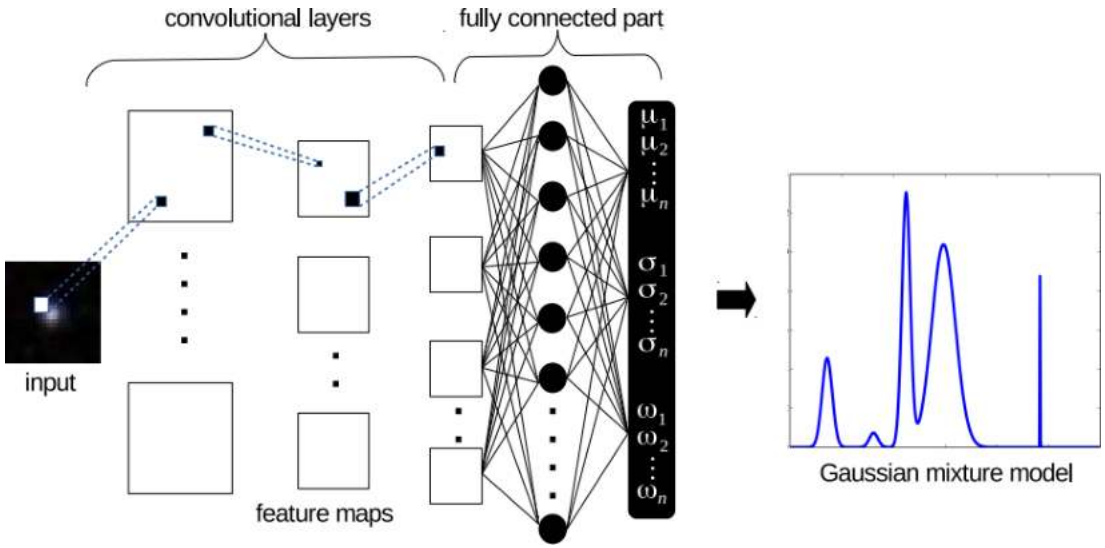
Deep Learning **Classical approach**



Geometric Effects are automatically considered (beyond photometry)



Pasquet+18

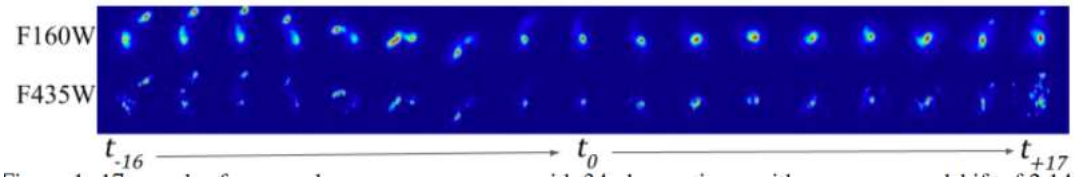


Disanto+18

Uncertainty quantification through Mixture Density Networks

Hazevah+17, Perreault-Levasseur+17, Morningstar+18, Stark+18, Tuccillo+18, Bom+19, Lovell+19, Madireddy+19, Pearson+19, Simet+19, Pasquet+19, Wu+19, Menou+19, Aragon-Calvo+20, Chianese+20, Stahl+20, Surana+20, Shuntov+20, Cabayol-Garcia+20, Campagne+20, Buck+21, Grover+21, Hayat+21, Li+21, Maresca+21, Qiu+21, Rhea+21, Schuldt+21, Tohill+21, Yao-Yu+21, Dey+21, Lee+21, Zhou+21, Henghes+21, Ansari+21

Mergers of Galaxies

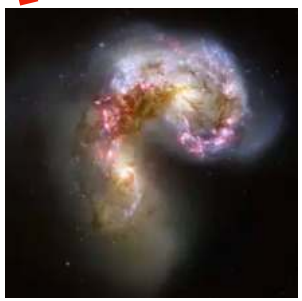


Merger of galaxies sequence from cosmological simulations

?

?

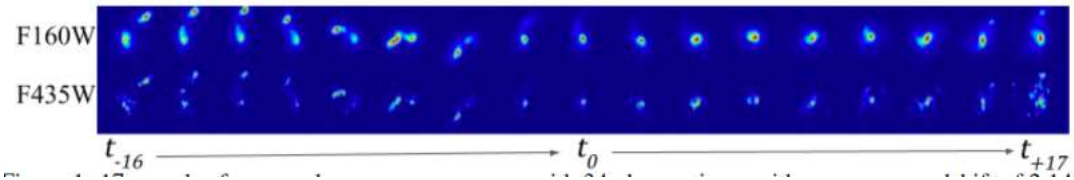
Merger stage?



Neural Networks to find relations between observables and physical processes

Ntampaka+16, Huertas-Company+18, Schawinski+18, Armitage+19, Bottrell+19, Calderon+19, Diaz+19, Ho+19, Pearson+19, Cai+20, Ferreira+20, Hansen+20, Kodi Ramanah+20, Wu+20, Yan+20, Gupta+20, de Andres+21, Bickley+21, Ciprijanovic+21, Ho+21, Koppula+21, von Marttens+21, Shao+21, Villanueva-Domingo+21, Whithney+21, Holwerda+21

Mergers of Galaxies

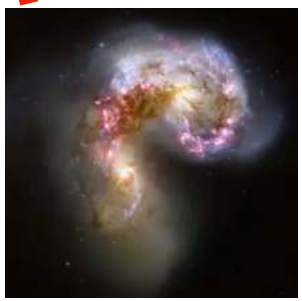


Merger of galaxies sequence from cosmological simulations

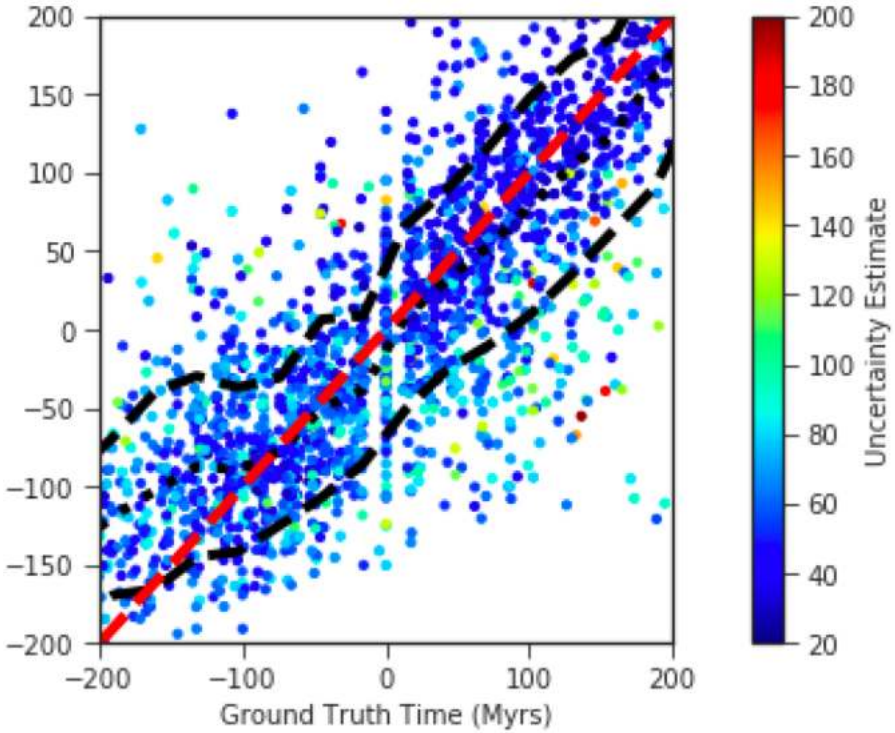
?

?

Merger stage?



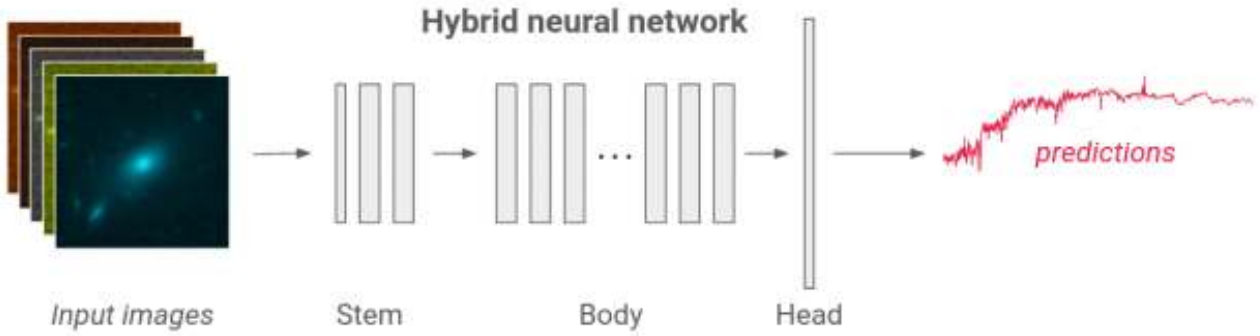
Neural Networks to find relations between observables and physical processes

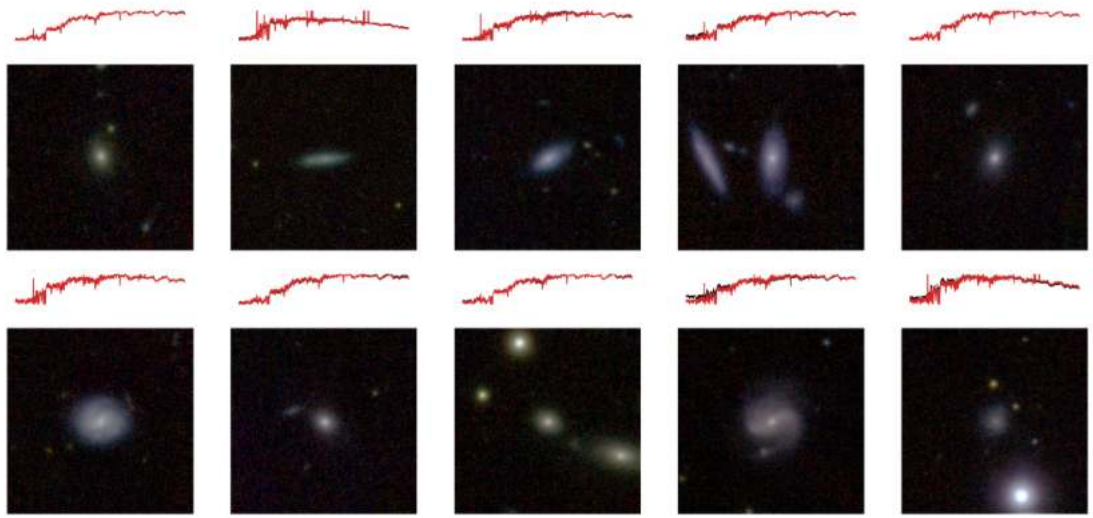


Koppula+21

Ntampaka+16, Huertas-Company+18, Schawinski+18, Armitage+19, Bottrell+19, Calderon+19, Diaz+19, Ho+19, Pearson+19, Cai+20, Ferreira+20, Hansen+20, Kodi Ramanah+20, Wu+20, Yan+20, Gupta+20, de Andres+21, Bickley+21, Ciprijanovic+21, Ho+21, Koppula+21, von Marttens+21, Shao+21, Villanueva-Domingo+21, Whithney+21, Holwerda+21

Virtual Observatory



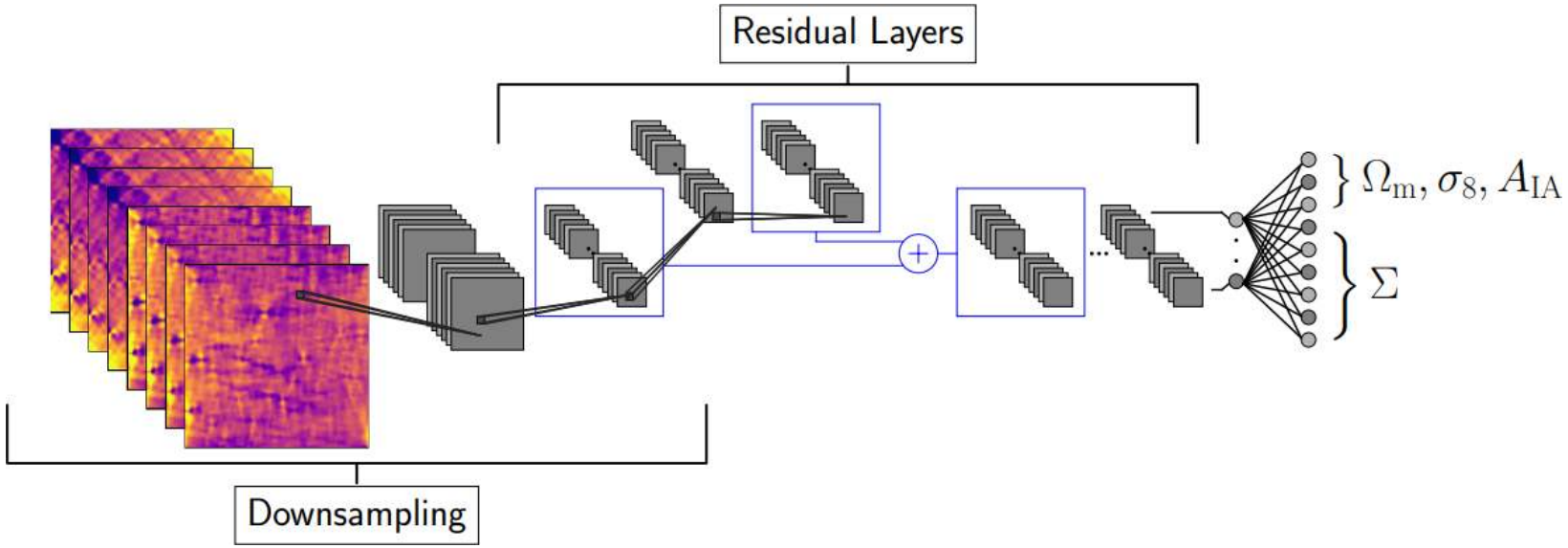


Neural Networks to learn complex mapping between observables

Wu+21

Ntampaka+16, Huertas-Company+18, Schawinski+18, Armitage+19, Bottrell+19, Calderon+19, Diaz+19, Ho+19, Pearson+19, Cai+20, Ferreira+20, Hansen+20, Kodi Ramanah+20, Wu+20, Yan+20, Gupta+20, de Andres+21, Bickley+21, Ciprijanovic+21, Ho+21, Koppula+21, von Marttens+21, Shao+21, Villanueva-Domingo+21, Whithney+21, Holwerda+21

Deep Learning for Cosmological Inference



Motivation: Generalize comparison of observations with theory, beyond basic summary statistics

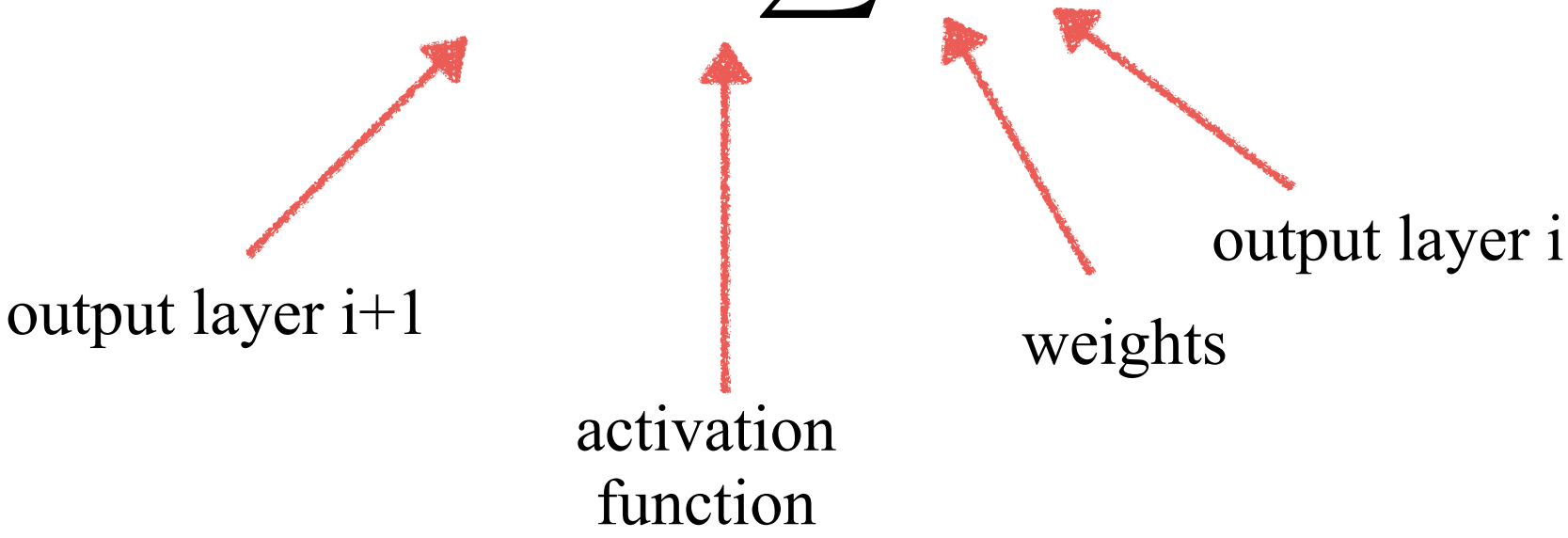
Neural Networks are used as efficient feature extractors

Training deeper networks

VANISHING / EXPLODING GRADIENT PROBLEM

REMEMBER THAT:

$$y_{i+1} = \sigma[\sum w_i y_i]$$

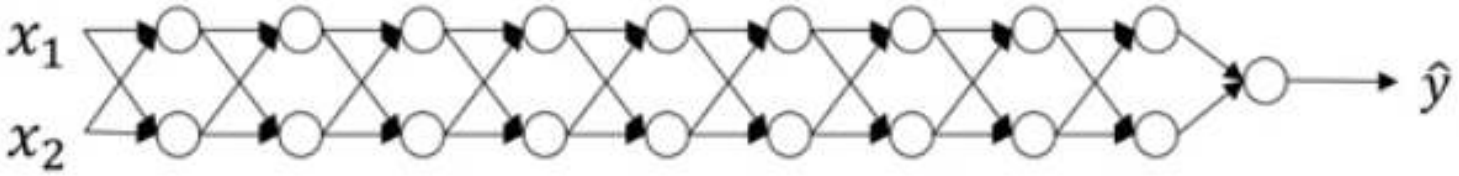


VANISHING / EXPLODING GRADIENT PROBLEM

WITH MANY LAYERS:

$$y_n = \sigma\left(...\sigma\left(...\sigma\left(\sum w_0 x\right)\right)\right)$$

VANISHING/EXPLODING GRADIENT PROBLEM



$$w_i = \begin{pmatrix} w_i^0 & 0 \\ 0 & w_i^1 \end{pmatrix} \qquad \hat{y} = x \prod_n w_i$$

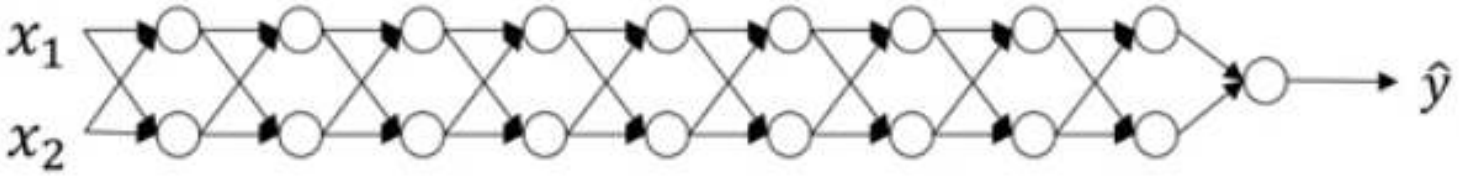
$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

IF WEIGHTS ARE ALL INITIALIZED TO VALUES <<1:

$$\hat{y} \to 0$$

VANISHING GRADIENT

VANISHING/EXPLODING GRADIENT PROBLEM



$$w_i = \begin{pmatrix} w_i^0 & 0 \\ 0 & w_i^1 \end{pmatrix} \qquad \hat{y} = x \prod_n w_i$$

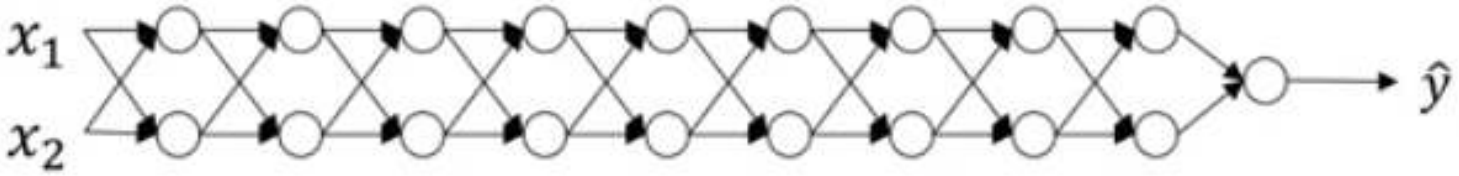
$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

IF WEIGHTS ARE ALL INITIALIZED TO VALUES >1:

$$\hat{y} \to \infty$$

EXPLODING GRADIENT

VANISHING/EXPLODING GRADIENT PROBLEM



$$w_i = \begin{pmatrix} w_i^0 & 0 \\ 0 & w_i^1 \end{pmatrix} \qquad \hat{y} = x \prod_n w_i$$

$$x = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix}$$

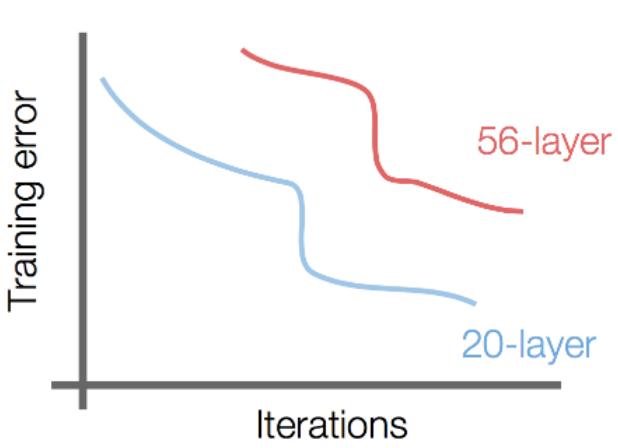
IF WEIGHTS ARE ALL INITIALIZED TO VALUES >1:

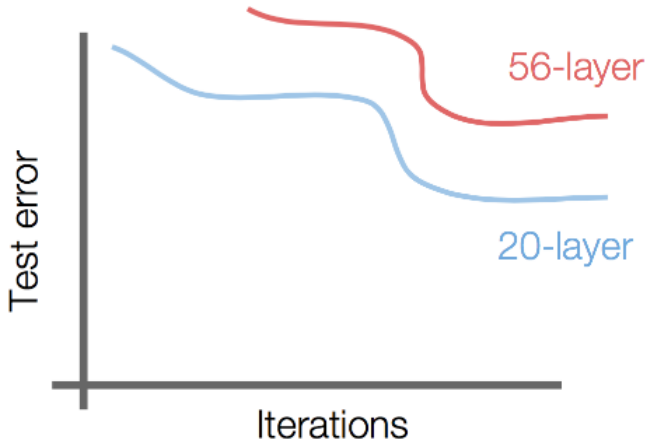
$$w_i^L \to \infty$$

EXPLODING GRADIENT

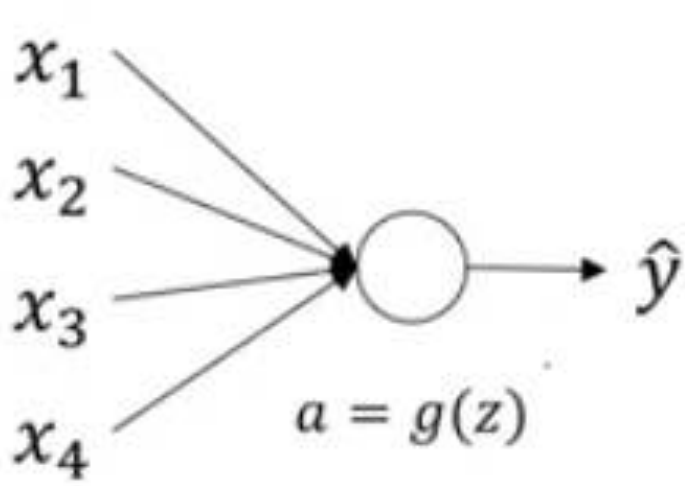
VANISHING/EXPLODING GRADIENT PROBLEM

TRAINING BECOMES UNSTABLE
VERY SLOW OR NO CONVERGENCE



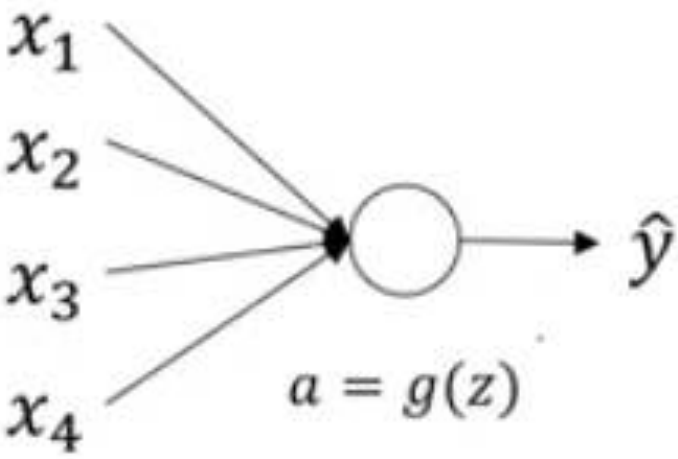


WEIGHT INITIALIZATION IS A KEY POINT…



$$z = x_1 w_1 + x_2 w_2 + \ldots + x_n w_n$$

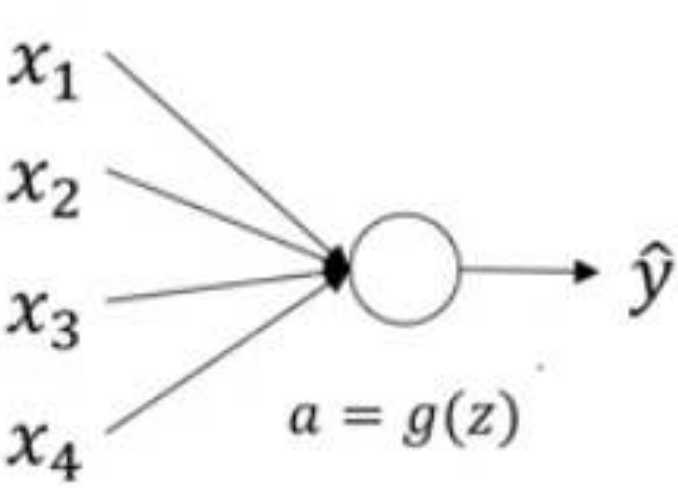
WEIGHT INITIALIZATION IS A KEY POINT…



$$z = x_1 w_1 + x_2 w_2 + \ldots + x_n w_n$$

THE LARGER n, THE SMALLER WEIGHTS SHOULD BE…

WEIGHT INITIALIZATION IS A KEY POINT…



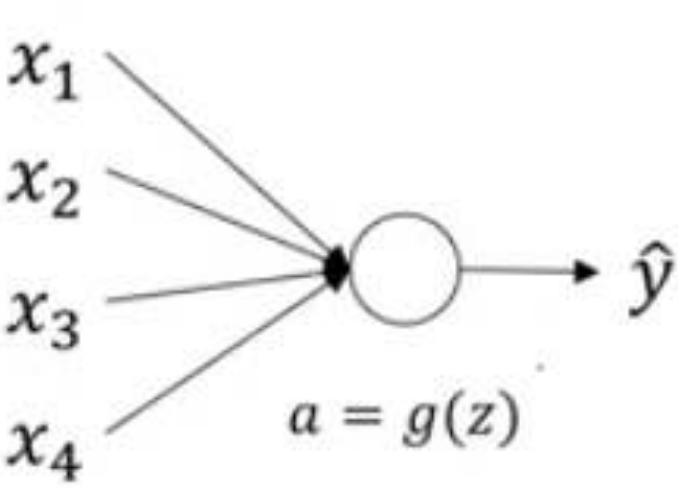
$$z = x_1 w_1 + x_2 w_2 + \ldots + x_n w_n$$

THE LARGER n, THE SMALLER WEIGHTS SHOULD BE…

variance

<u>ONE SIMPLE SOLUTION:</u> $\sigma^2(w_i) = \frac{1}{n}$ ← number of inputs

WEIGHT INITIALIZATION IS A KEY POINT…



$$z = x_1 w_1 + x_2 w_2 + \ldots + x_n w_n$$

THE LARGER n, THE SMALLER WEIGHTS SHOULD BE…

variance

<u>ONE SIMPLE SOLUTION:</u> $\sigma^2(w_i) = \frac{1}{n}$ ← number of inputs

WEIGHT INITIALIZATION IS A KEY POINT…

IMPLEMENTATION IN KERAS:

```
initialization = 'he_normal'
act = 'relu'

model = Sequential()
model.add(Convolution2D(depth, conv_size, conv_size, activation=act, border_mode='same',
name = "conv%i"%(layer_n), init=initialization, W_constraint=constraint))
```

WEIGHT INITIALIZATION IS A KEY POINT…

IMPLEMENTATION IN KERAS:

```
initialization = 'he_normal'
act = 'relu'

model = Sequential()
model.add(Convolution2D(depth, conv_size, conv_size, activation=act, border_mode='same',
name = "conv%i"%(layer_n), init=initialization, W_constraint=constraint))
```

MANY OTHER INITIALIZATIONS AVAILABLE:

`keras.initializers` → https://keras.io/initializers/

BATCH NORMALIZATION

[SZEGEDY+15]

A 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

INPUT

$$\hat{y}_i = \gamma \frac{y_i - E(y_i)}{\sigma(y_i)} + \beta$$

NORMALIZED INPUT

SCATTER

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: γ, β

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

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

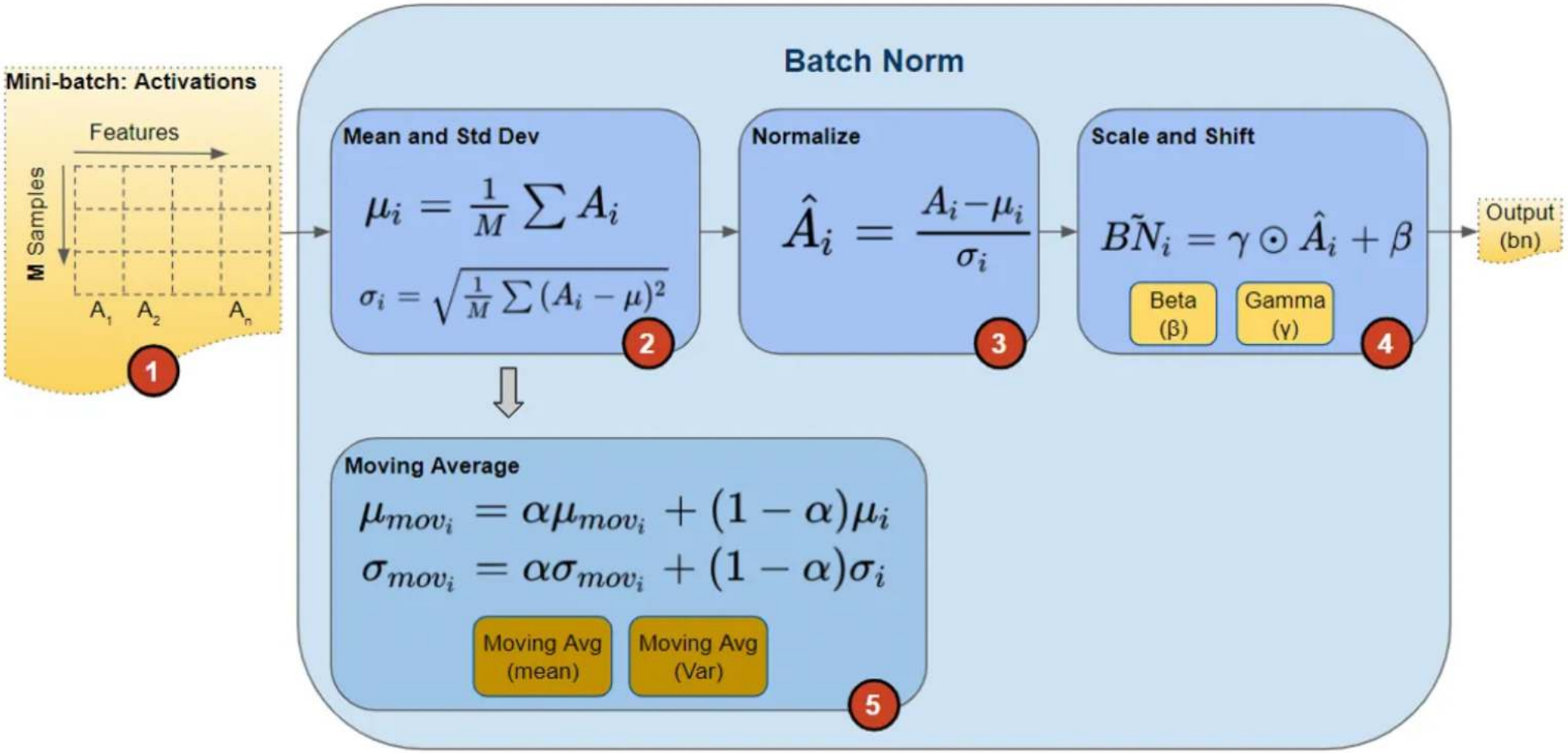
$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

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

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

BATCH NORMALIZATION

[SZEGEDY+15]

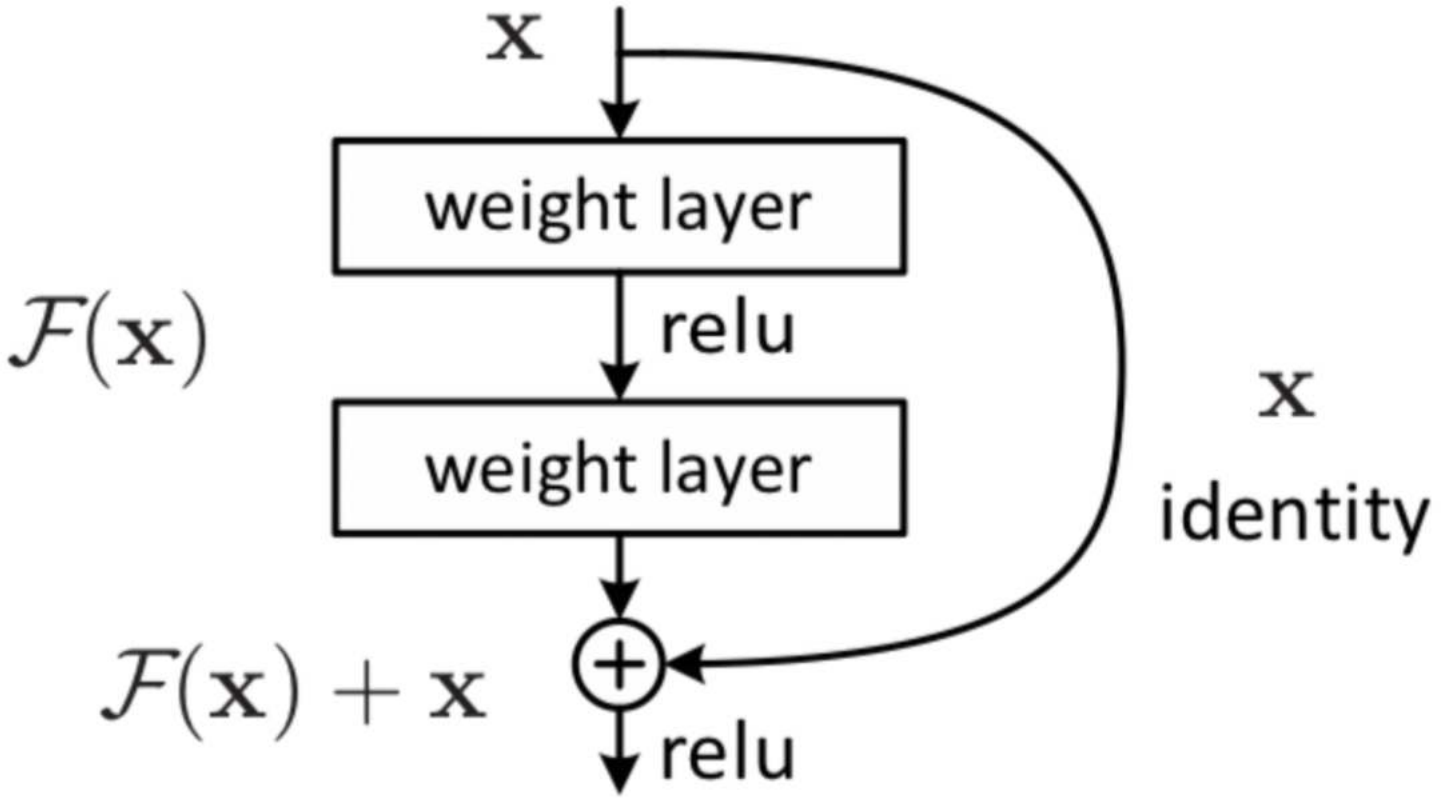


BATCH NORMALIZATION

[SZEGEDY+15]

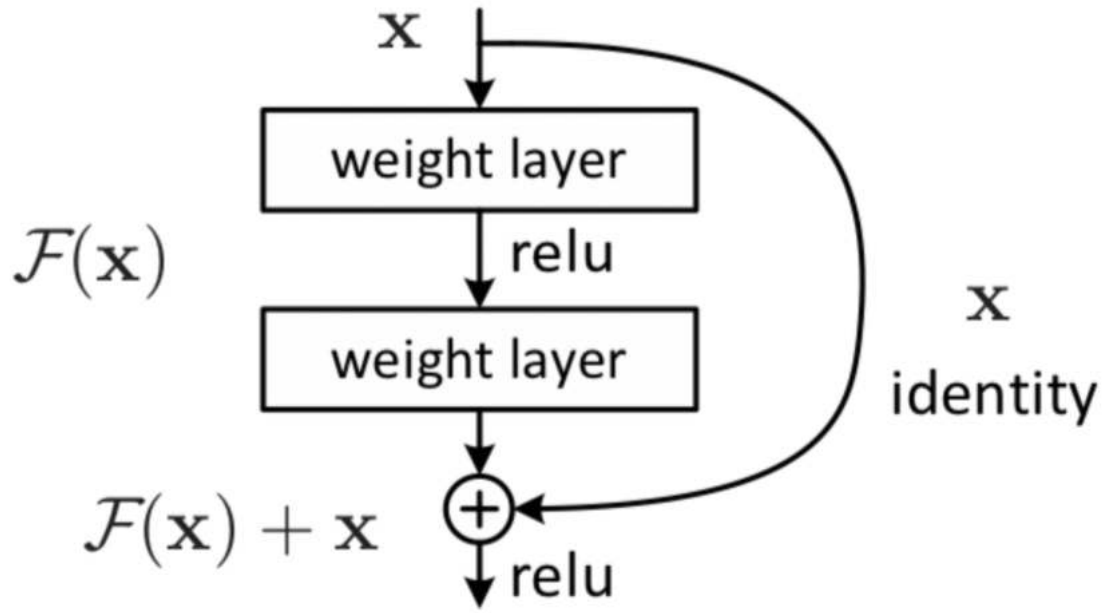
- Speeds up training
- Reduces internal covariate shift of the network
- Regularizes the network, prevents over fitting

RESIDUAL NETWORKS



Kaiming et al. 2015

RESIDUAL NETWORKS



- Adding additional / new layers would not hurt the model's performance as regularisation will skip over them if those layers were not useful.

- If the additional / new layers were useful, even with the presence of regularisation, the weights or kernels of the layers will be non-zero and model performance could increase slightly.

Kaiming et al. 2015

Adding additional invariances

DATA AUGMENTATION

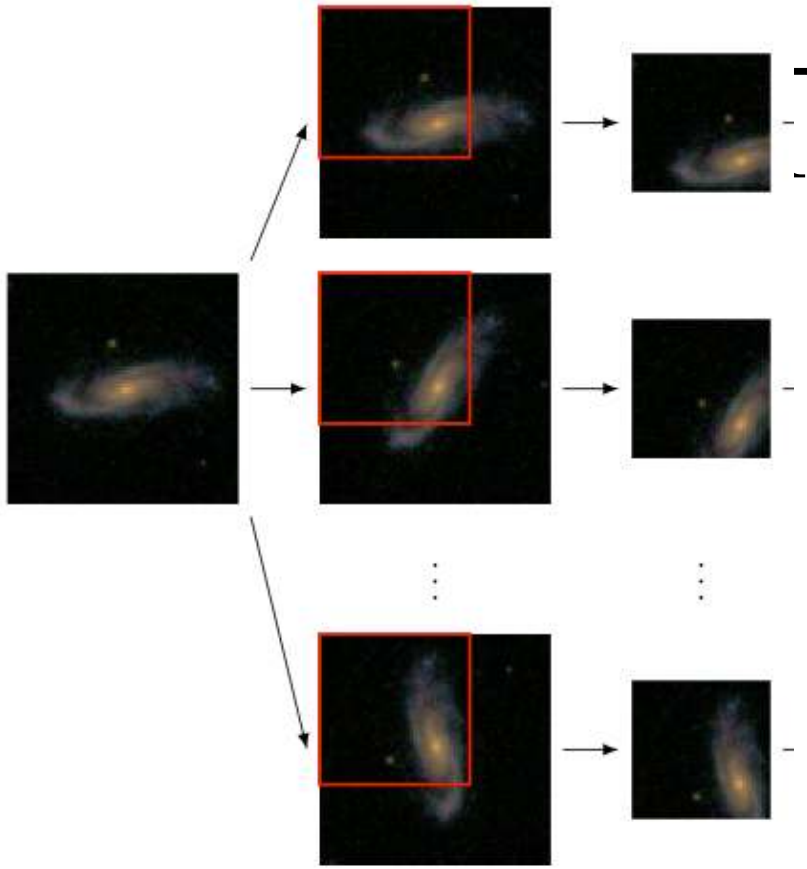
ANOTHER WAY TO REDUCE OVER-FITTING IS TO "AUGMENT" THE SIZE OF THE DATASET AVAILABLE FOR TRAINING

FOR MANY APPLICATIONS THE CLASSIFICATION SHOULD BE INDEPENDENT TO:

- TRANSALTIONS
- ROTATIONS
- SCALINGS

ETC…

DATA [illegible]TATION



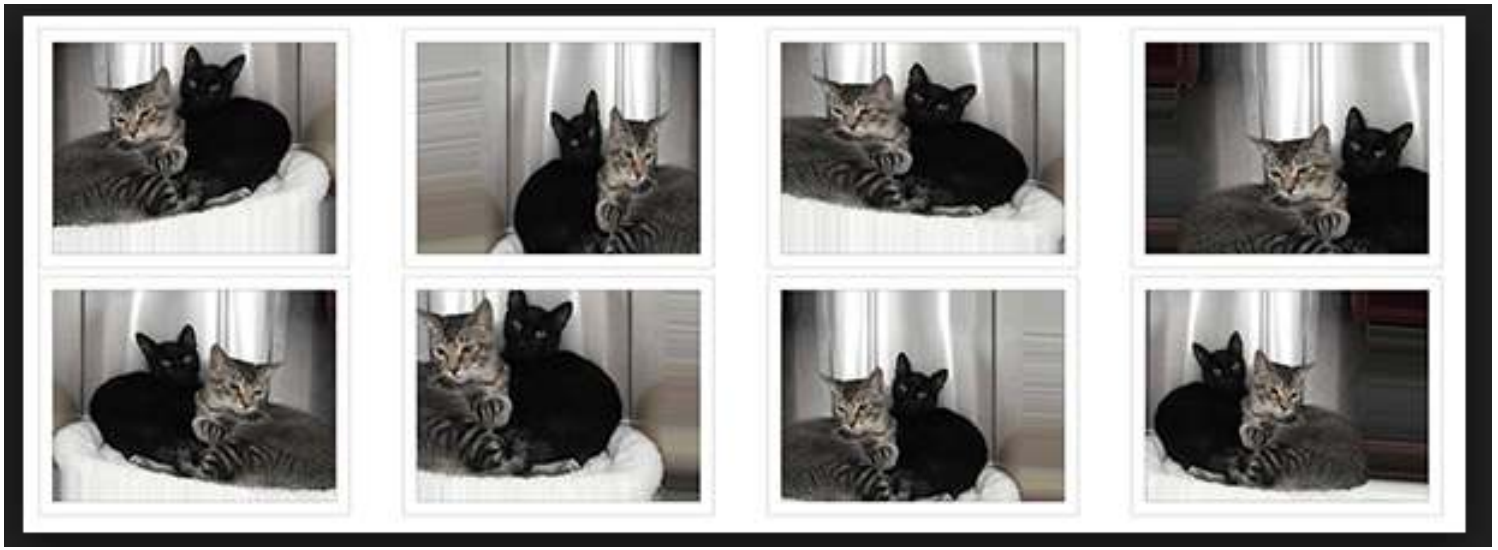
Dieleman+15

FOR MANY APPLICATIONS THE CLASSIFICATION SHOULD BE INDEPENDENT TO:

- TRANSALTIONS
- ROTATIONS
- SCALINGS

ETC…

DATA AUGMENTATION



FOR MANY APPLICATIONS THE CLASSIFICATION SHOULD BE INDEPENDENT TO:

- TRANSALTIONS
- ROTATIONS
- SCALINGS

ETC…

Dealing with a lack of labelled data

Transfer learning

THE CONVOLUTIONAL PART OF A CNN IS
A FEATURE EXTRACTOR ….

Transfer learning

THE CONVOLUTIONAL PART OF A CNN IS
A FEATURE EXTRACTOR

IN THAT RESPECT, THEY ARE VERY FLEXIBLE …

Transfer Learning)

EVEN IF OUR TRAINING SET IS NOT SO LARGE …

WE CAN USE A CNN PRE-TRAINED ON A LARGER SAMPLE

DEPENDING ON HOW SIMILAR BOTH DATASETS ARE, WE CAN:

- RECYCLE THE SAME FEATURES
- FINE-TUNING THE WEIGHTS

DATA FROM NEW SURVEY

How robust to different datasets? Do we always need a big training set?

DEEP-LEARNING BASED MACHINE

"Improved" Galaxy ZOO like classifications for for the entire sample

Transfer knowledge?

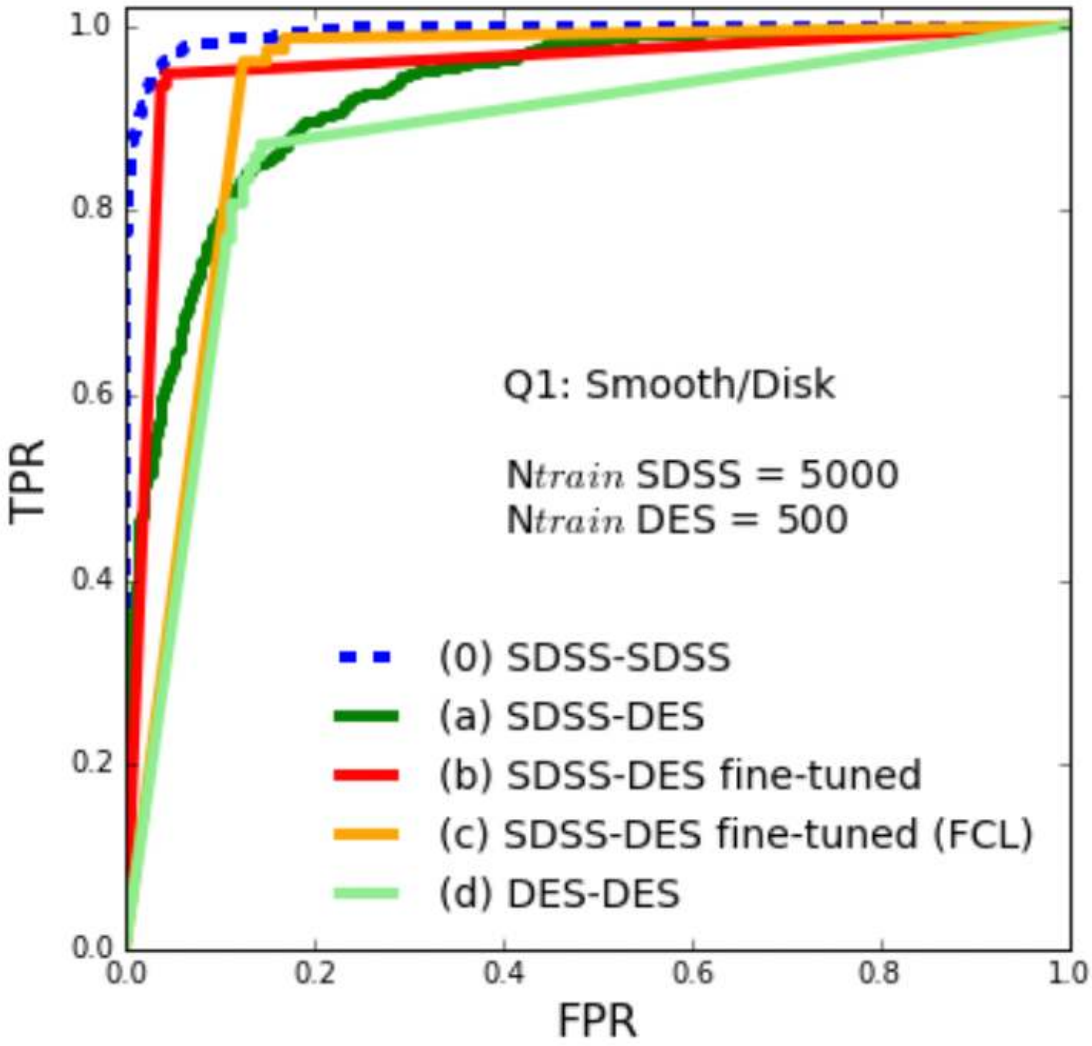
Human classifications from existing survey

Dominguez-Sanchez+18b

SDSS

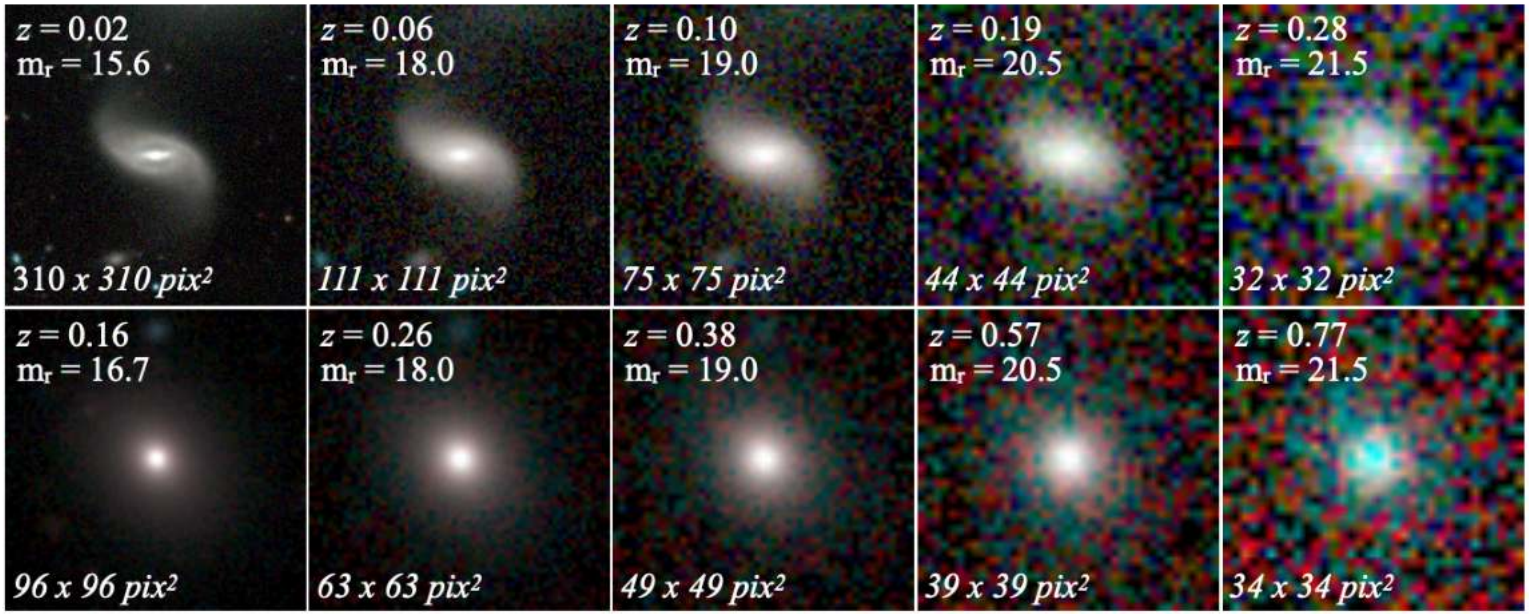
DES

Dominguez-Sanchez+18b

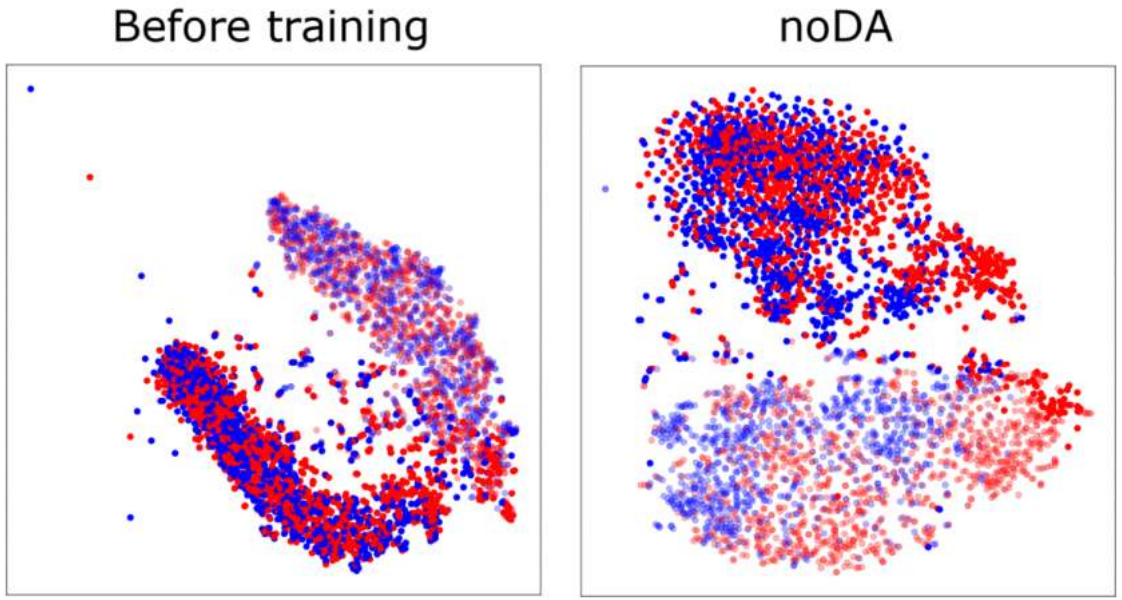


Dominguez-Sanchez+18b

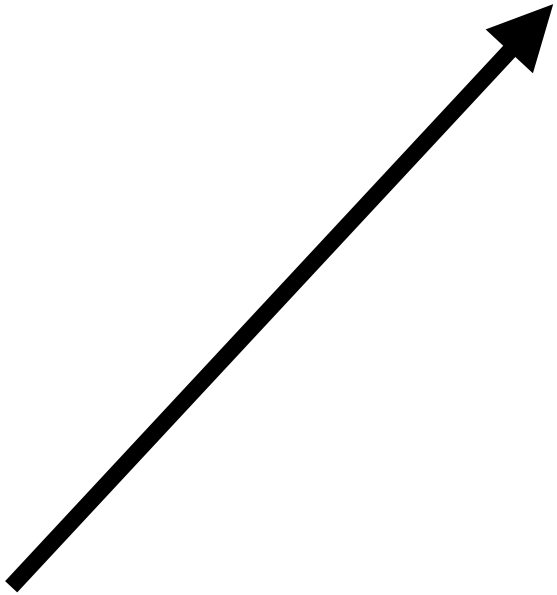
OR YOU CAN ALSO TRAIN ON SIMULATIONS



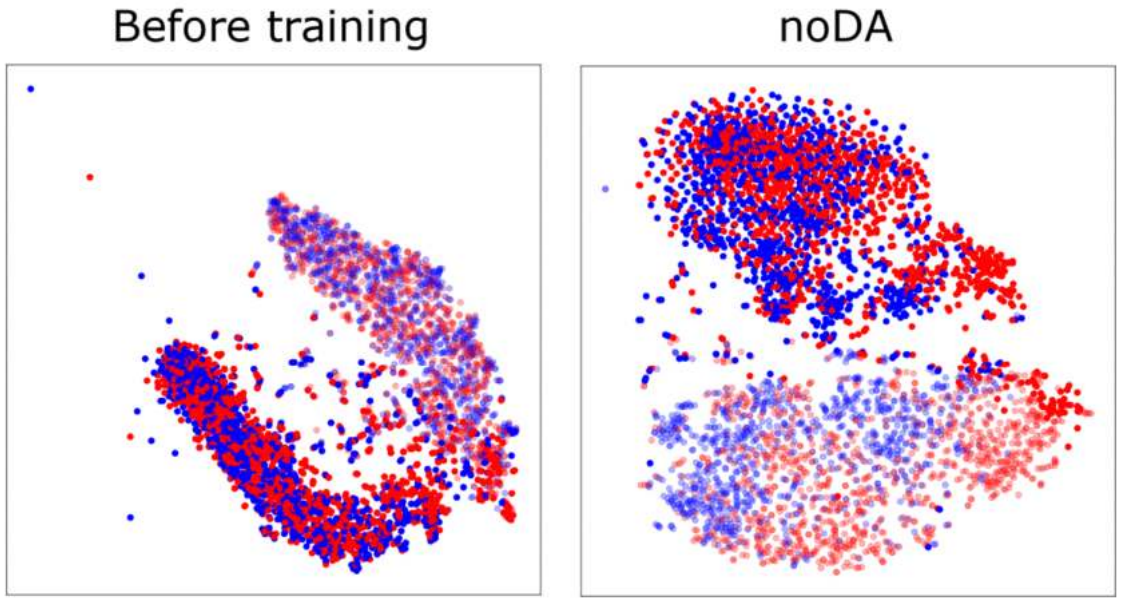
THIS IS WHERE DOMAIN KNOWLEDGE COMES INTO PLAY



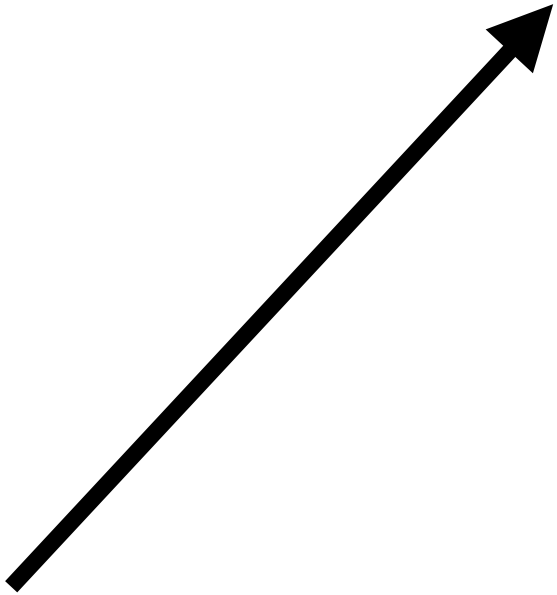
Layers	Properties	Stride	Padding	Output Shape	Parameters
Input	$3 \times 75 \times 75$[a]	-	-	(3, 75, 75)	0
Convolution (2D)	Filters: 8	1	2	(8, 75, 75)	608
	Kernel: 5×5	-	-	-	-
	Activation: ReLU	-	-	-	-
Batch Normalization	-	-	-	(8, 75, 75)	16
MaxPooling	Kernel: 2×2	2	0	(8, 37, 37)	0
Convolution (2D)	Filters: 16	1	1	(16, 37, 37)	1168
	Kernel: 3×3	-	-	-	-
	Activation: ReLU	-	-	-	-
Batch Normalization	-	-	-	(16, 37, 37)	32
MaxPooling	Kernel: 2×2	2	0	(16, 18, 18)	0
Convolution (2D)	Filters: 32	1	1	(32, 18, 18)	4640
	Kernel: 3×3	-	-	-	-
	Activation: ReLU	-	-	-	-
Batch Normalization	-	-	-	(32, 18, 18)	64
MaxPooling	Kernel: 2×2	2	0	(32, 9, 9)	0
Flatten	-	-	-	(2592)	-
Fully connected	Activation: ReLU	-	-	(64)	165952
Fully connected	Activation: ReLU	-	-	(32)	2080
Fully connected	Activation: Softmax	-	-	(2)	66



Ciprijanovic+21

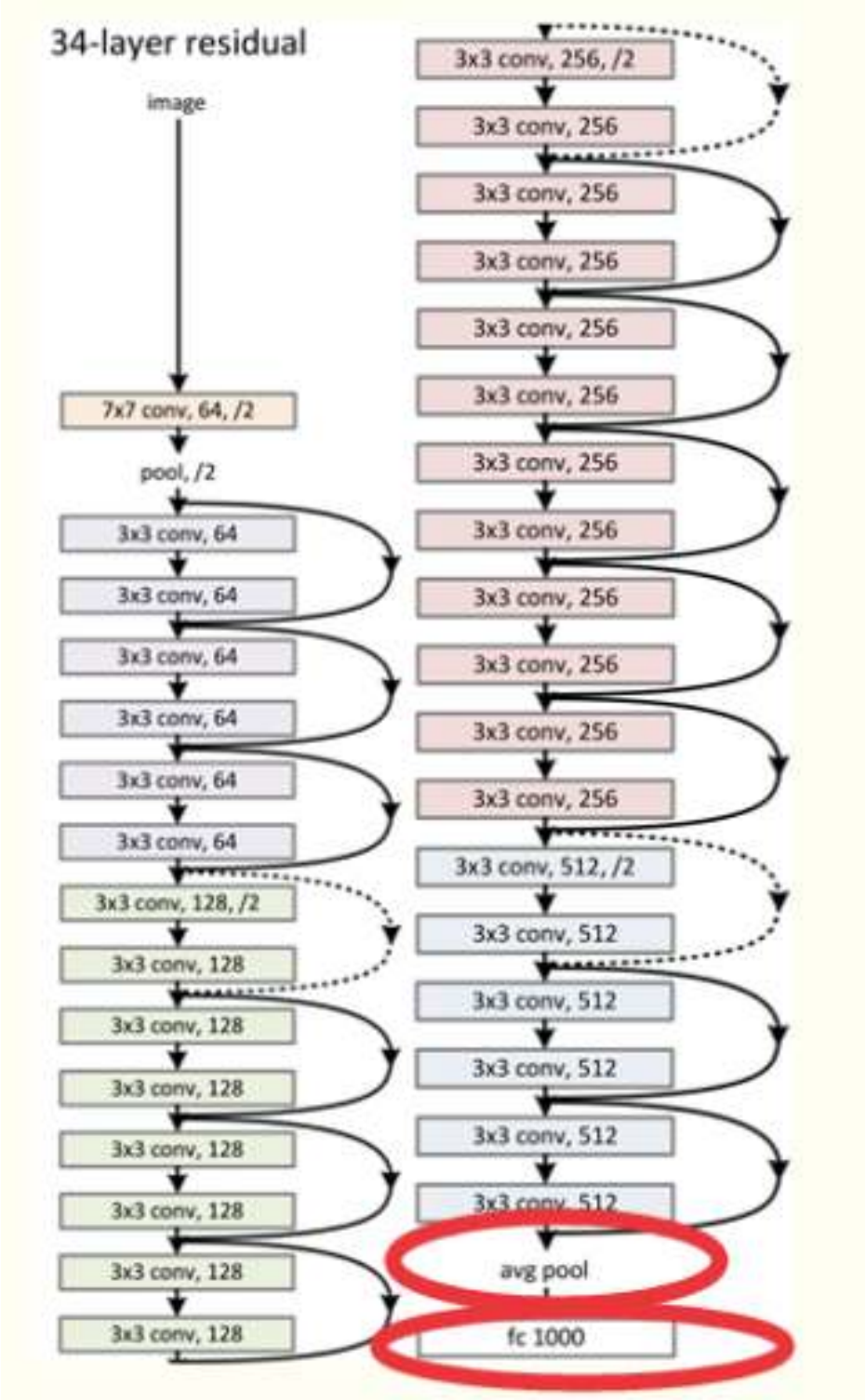


Layers	Properties	Stride	Padding	Output Shape	Parameters
Input	$3 \times 75 \times 75$[a]	-	-	(3, 75, 75)	0
Convolution (2D)	Filters: 8	1	2	(8, 75, 75)	608
	Kernel: 5×5	-	-	-	-
	Activation: ReLU	-	-	-	-
Batch Normalization	-	-	-	(8, 75, 75)	16
MaxPooling	Kernel: 2×2	2	0	(8, 37, 37)	0
Convolution (2D)	Filters: 16	1	1	(16, 37, 37)	1168
	Kernel: 3×3	-	-	-	-
	Activation: ReLU	-	-	-	-
Batch Normalization	-	-	-	(16, 37, 37)	32
MaxPooling	Kernel: 2×2	2	0	(16, 18, 18)	0
Convolution (2D)	Filters: 32	1	1	(32, 18, 18)	4640
	Kernel: 3×3	-	-	-	-
	Activation: ReLU	-	-	-	-
Batch Normalization	-	-	-	(32, 18, 18)	64
MaxPooling	Kernel: 2×2	2	0	(32, 9, 9)	0
Flatten	-	-	-	(2592)	-
Fully connected	Activation: ReLU	-	-	(64)	165952
Fully connected	Activation: ReLU	-	-	(32)	2080
Fully connected	Activation: Softmax	-	-	(2)	66

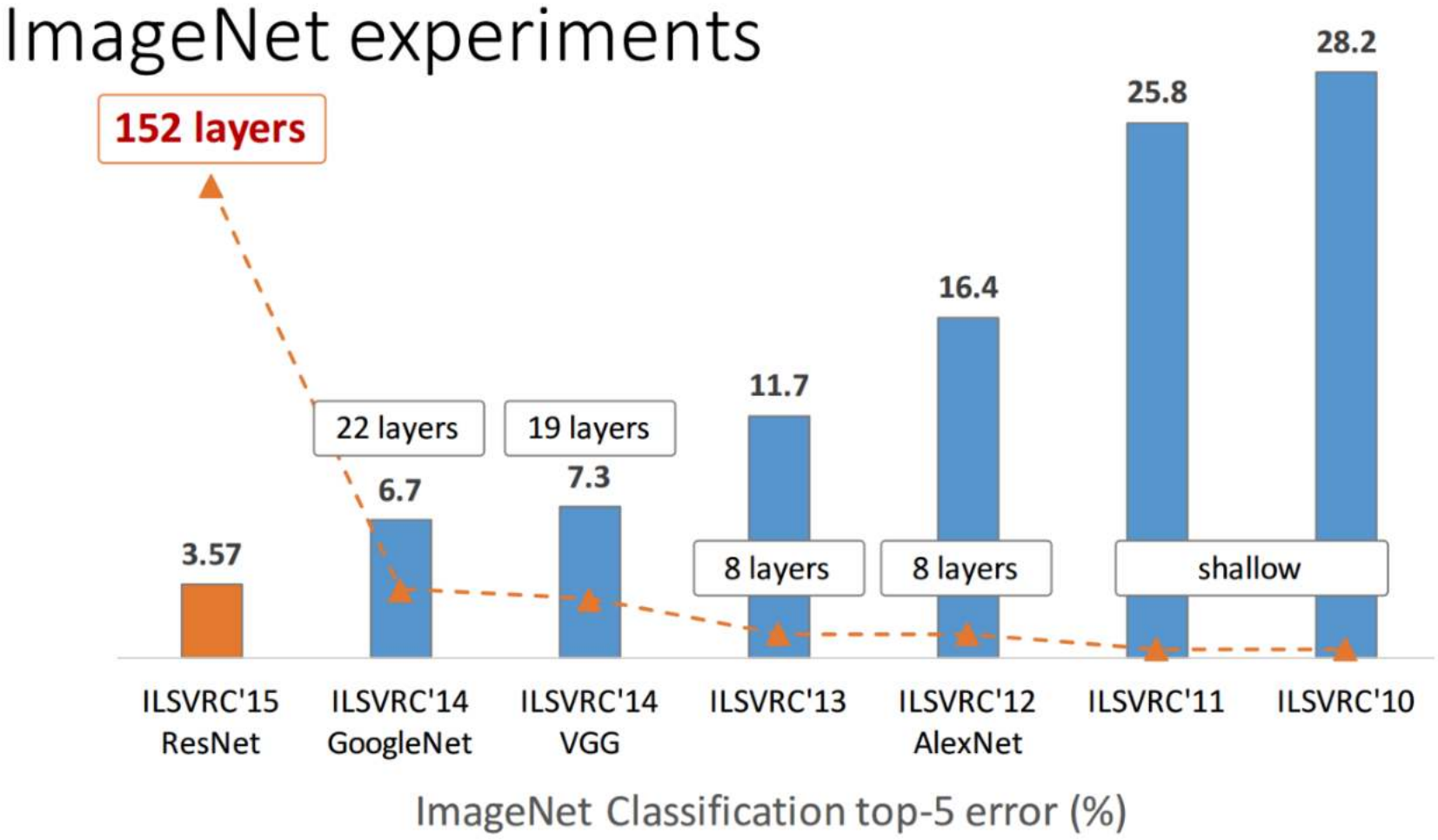


Ciprijanovic+21

EXAMPLE OF DEEP RESNET

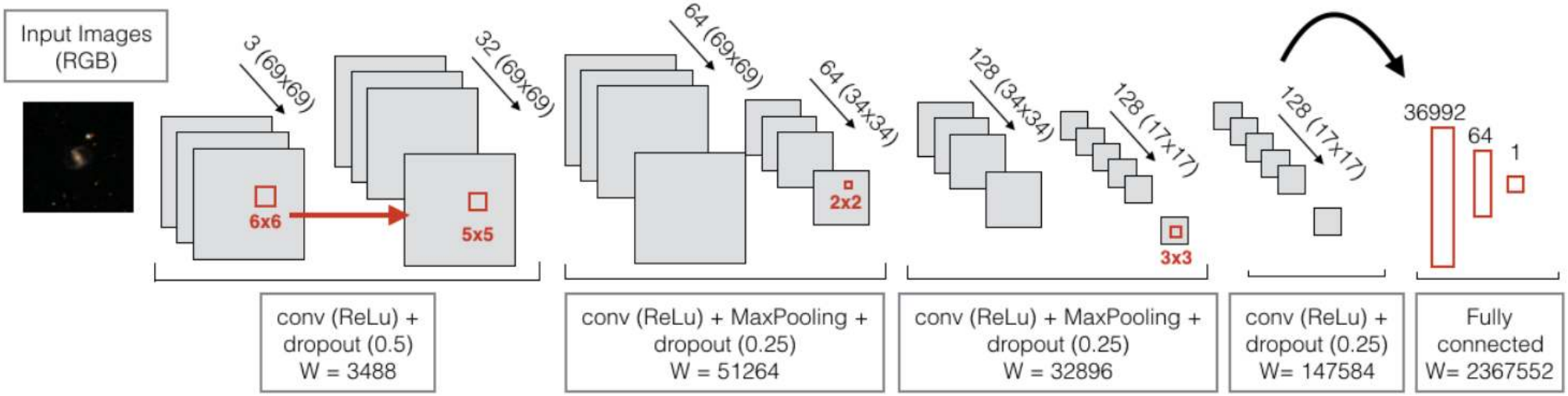


DEEPER TENDS TO BE BETTER…



BEYOND CLASSIFICATION: IMAGE2IMAGE NETWORKS

UP TO NOW CNNs MAP IMAGES (SIGNALS) INTO FLOATS



Dominguez-Sanchez+18

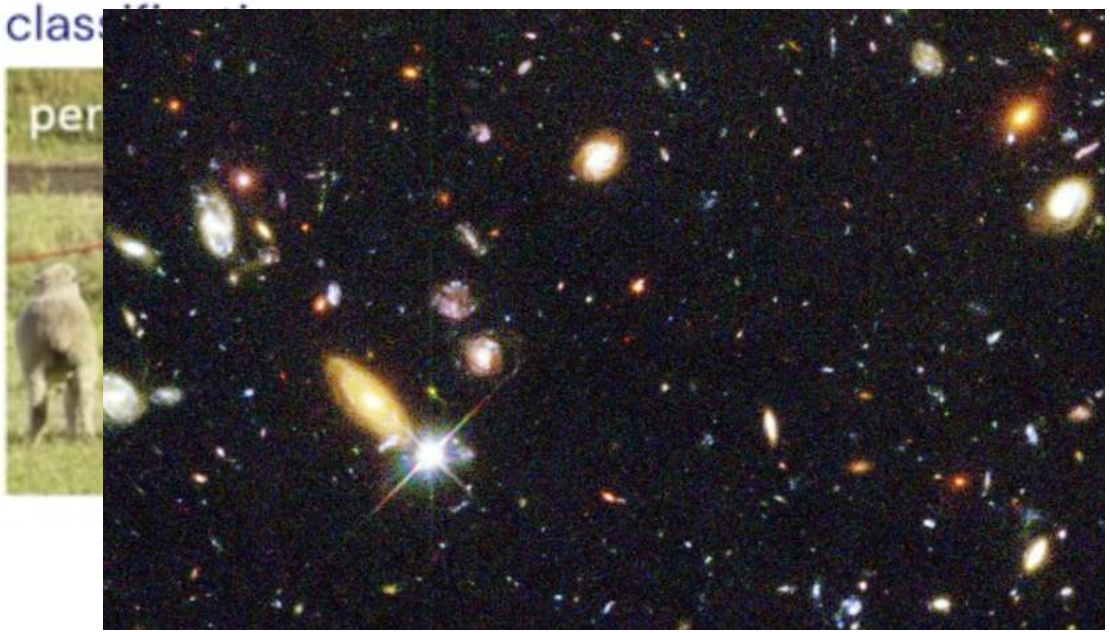
Classification has its limits ….

HOW DO I CLASSIFY THIS IMAGE?

Classification has its limits ….

HOW DO I CLASSIFY THIS IMAGE?

Classification has its limits ….

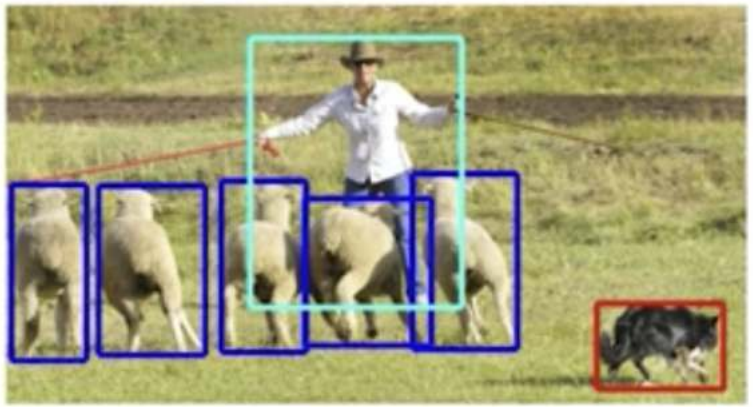


HOW DO I CLASSIFY THIS IMAGE?

Going beyond classification: increasing complexity

classification

object detection



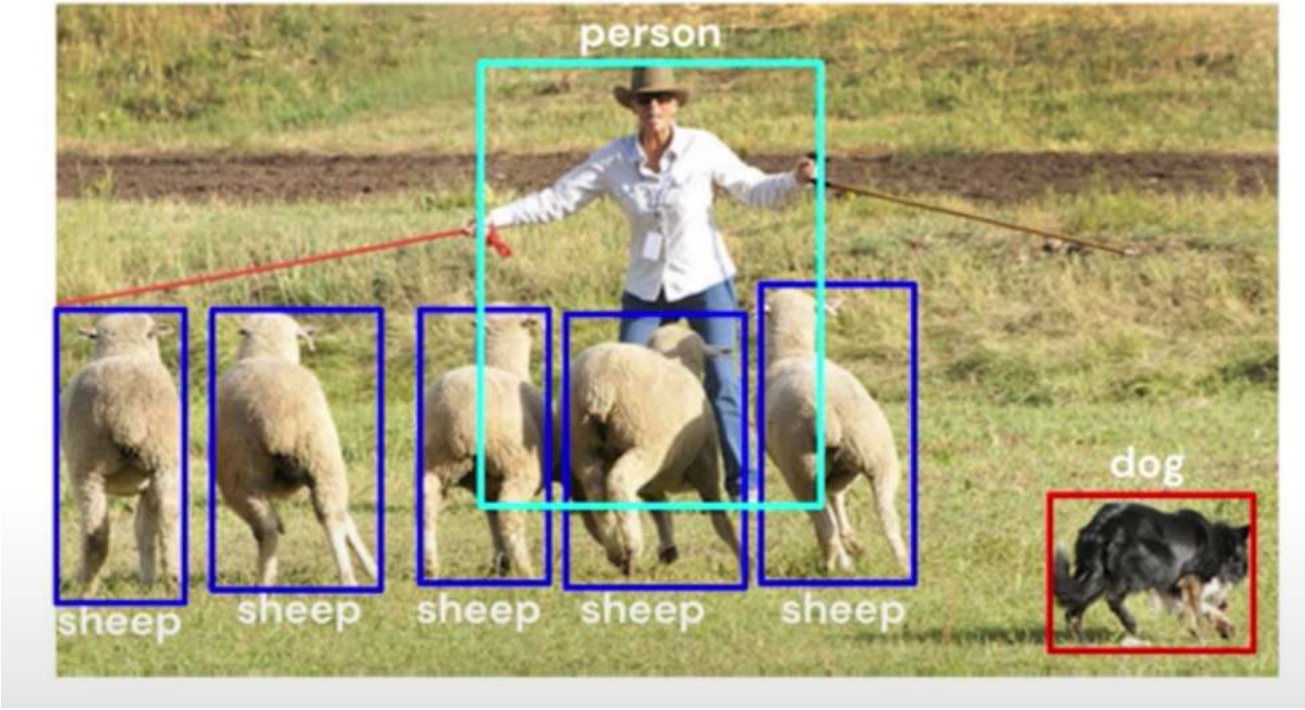
semantic segmentation

instance segmentation

@deepmind (V. Patraucean)

Object detection

First task is to find a bounding box for every object. How we do that?



@deepmind (V. Patraucean)

Inputs

- RGB image $H \times W \times 3$

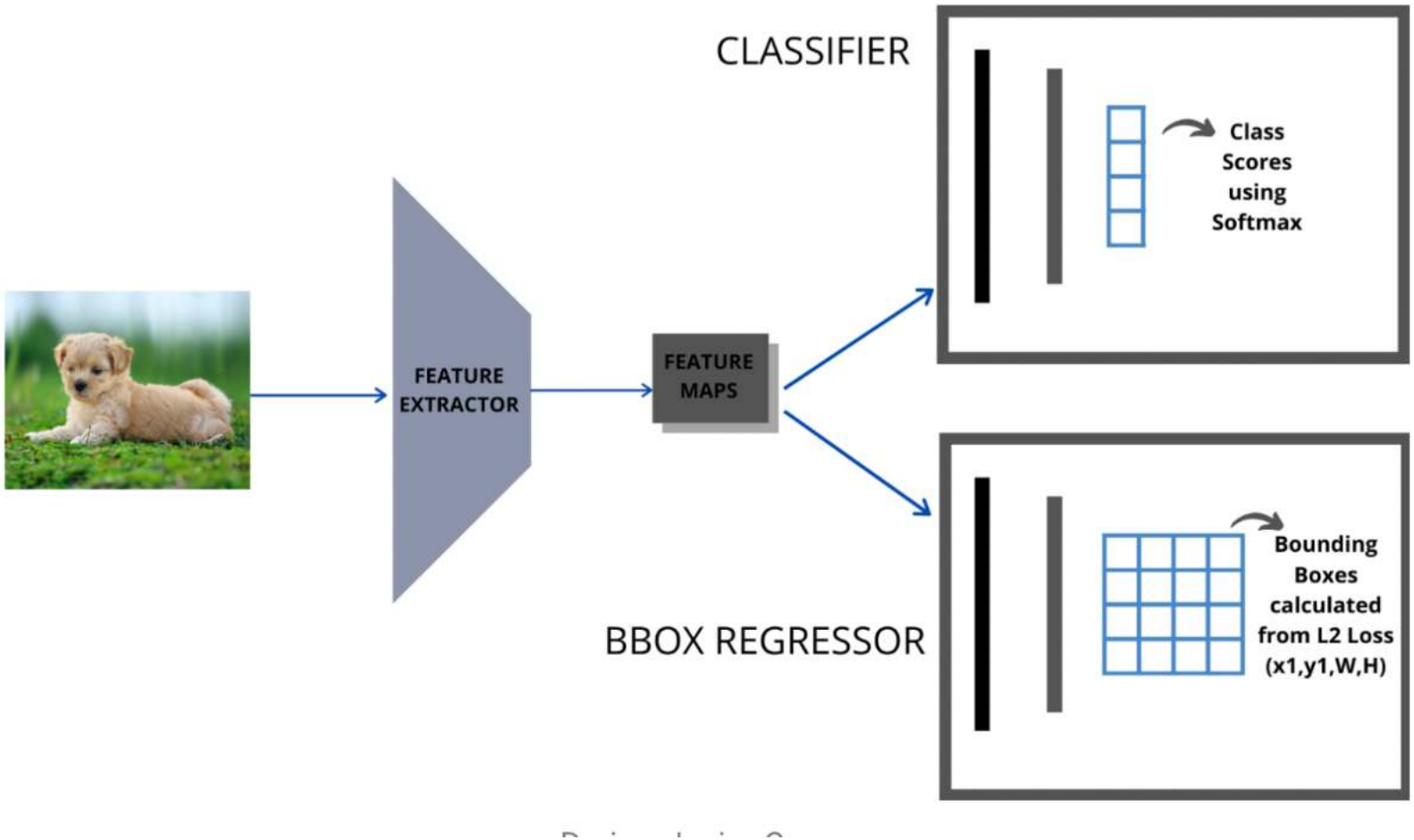
Targets

- Class label *one_hot* 0 0 0 1 0...
- Object bounding box

$$(x_c, y_c, h, w)$$

for all the objects present in the scene

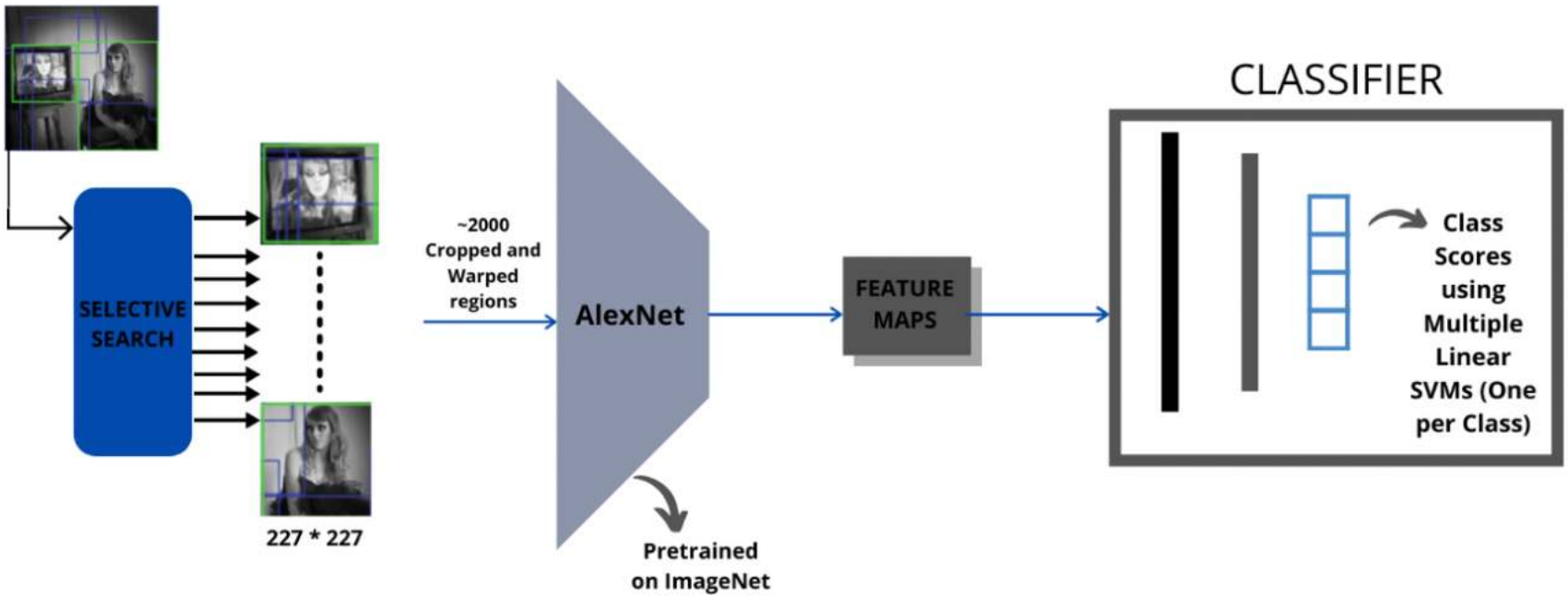
@deepmind (V. Patraucean)


CLASSIFIER
Class Scores using Softmax
FEATURE EXTRACTOR
FEATURE MAPS
BBOX REGRESSOR
Bounding Boxes calculated from L2 Loss (x1,y1,W,H)

The first ideas were based on first using existing methods such as Hierarchical Clustering

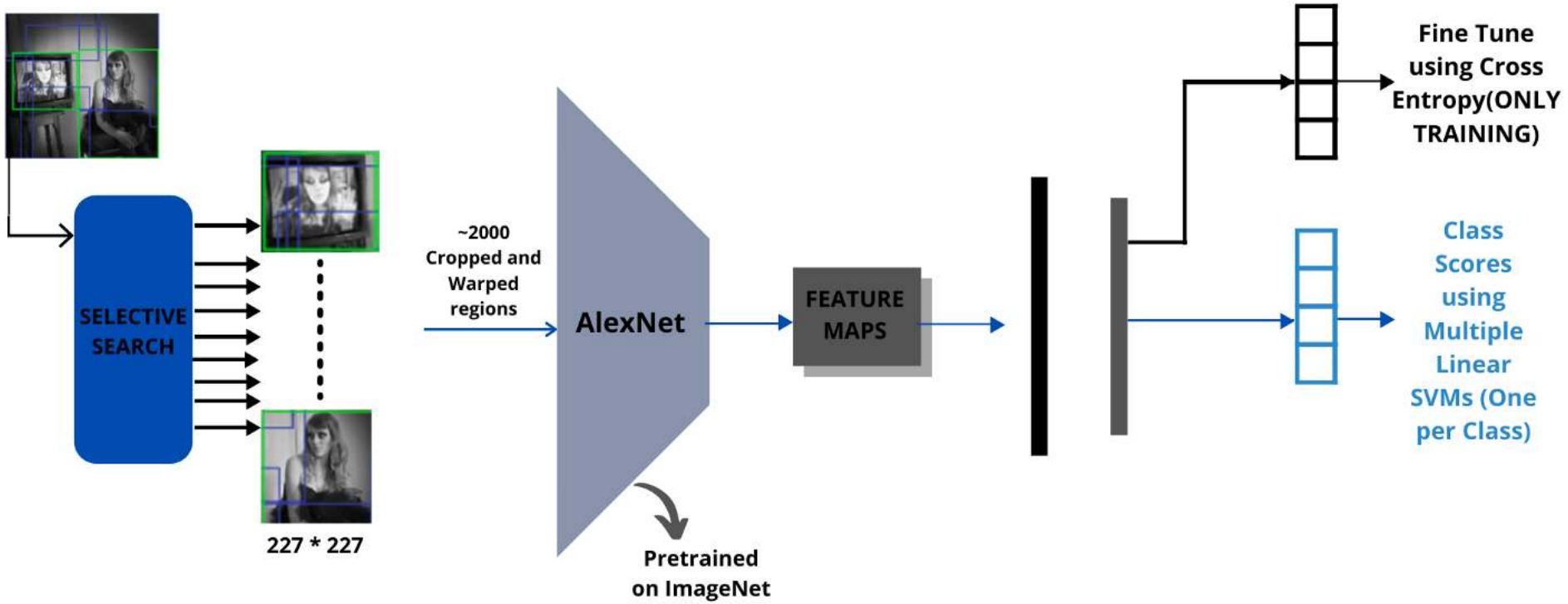
Grouping of pixels based on texture, color, composition …

R-CNN

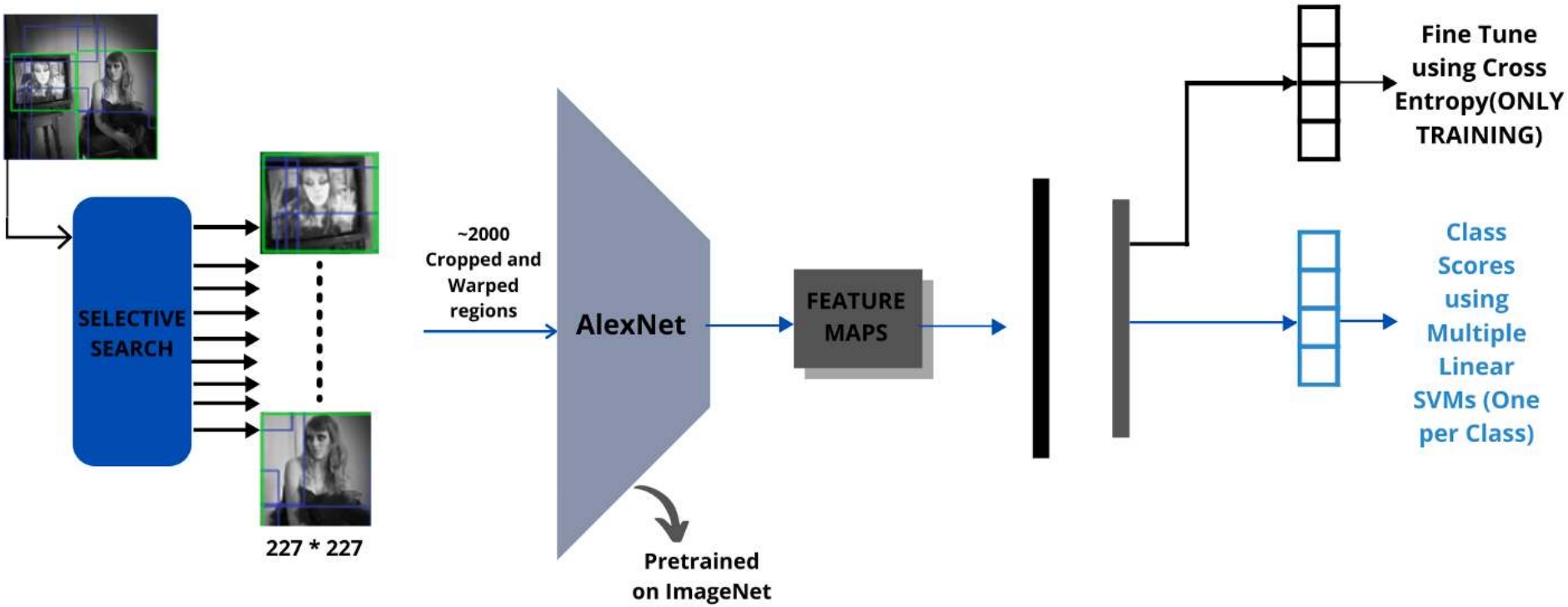


The hierarchical clustering is then combined with a CNN for classification of each region proposal.

The accuracy of this approach is not very high, requires changing the image aspect ratios

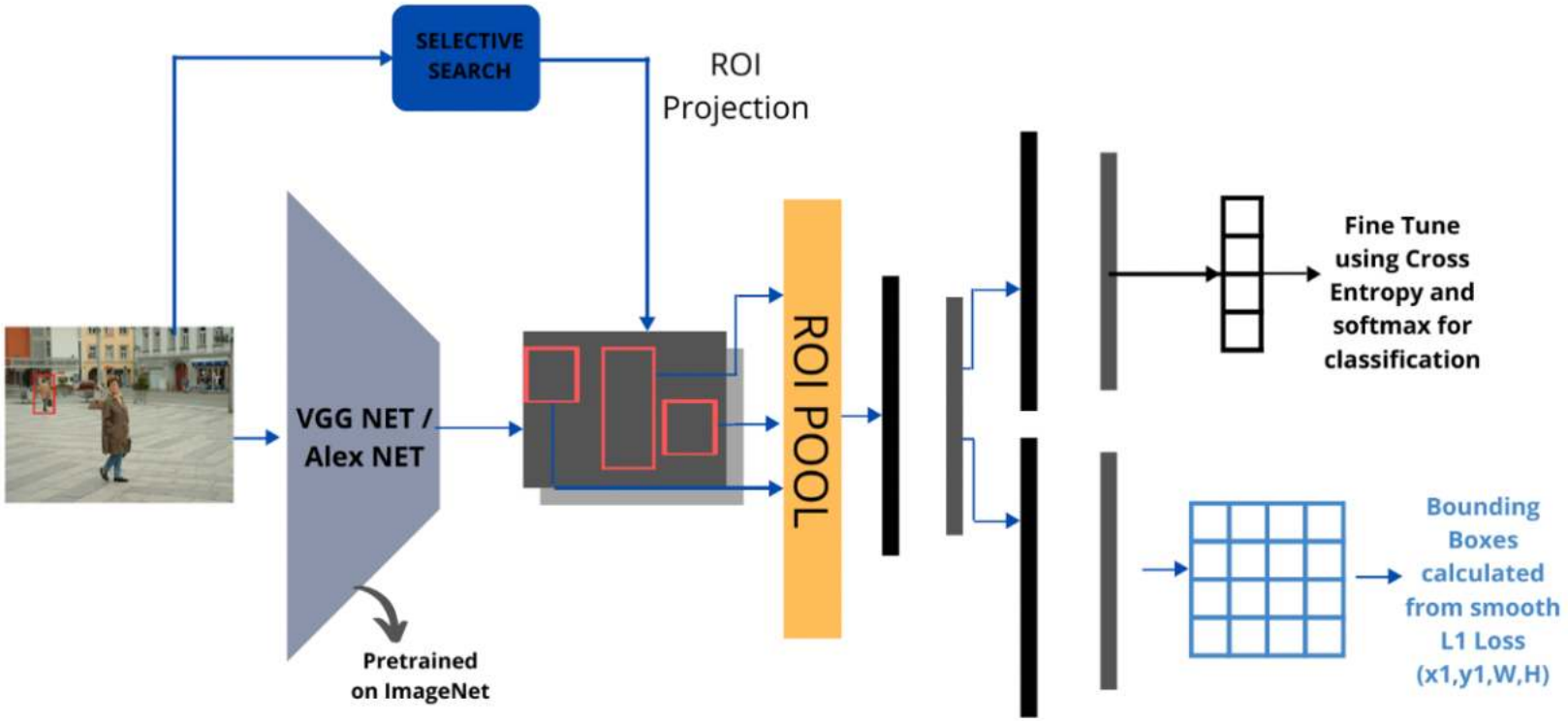


A solution was to add a fine tuning of the weights using an additional cross entropy loss

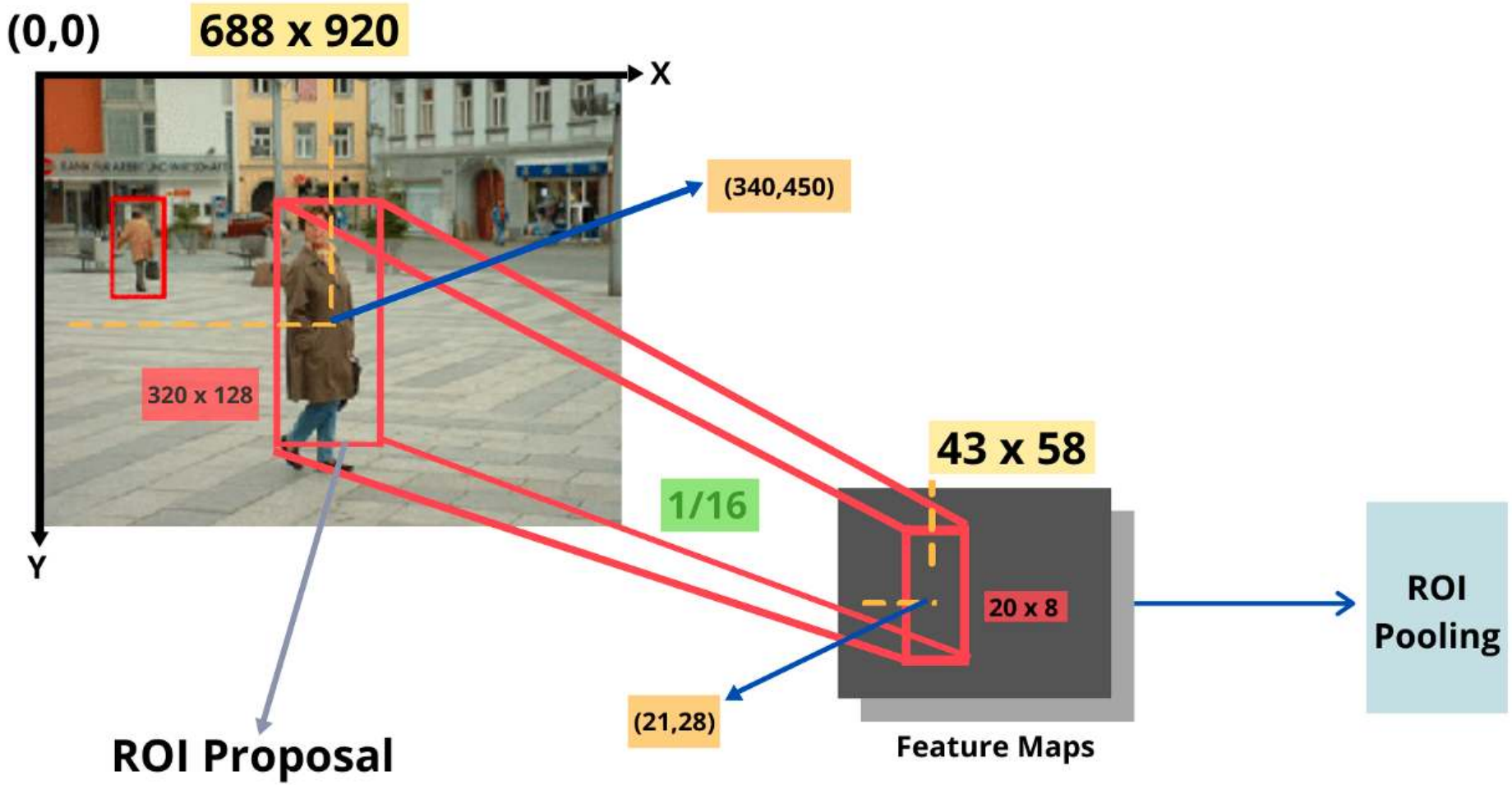


However, this requires a forward pass for every region proposal, which can be high

Fast R CNN

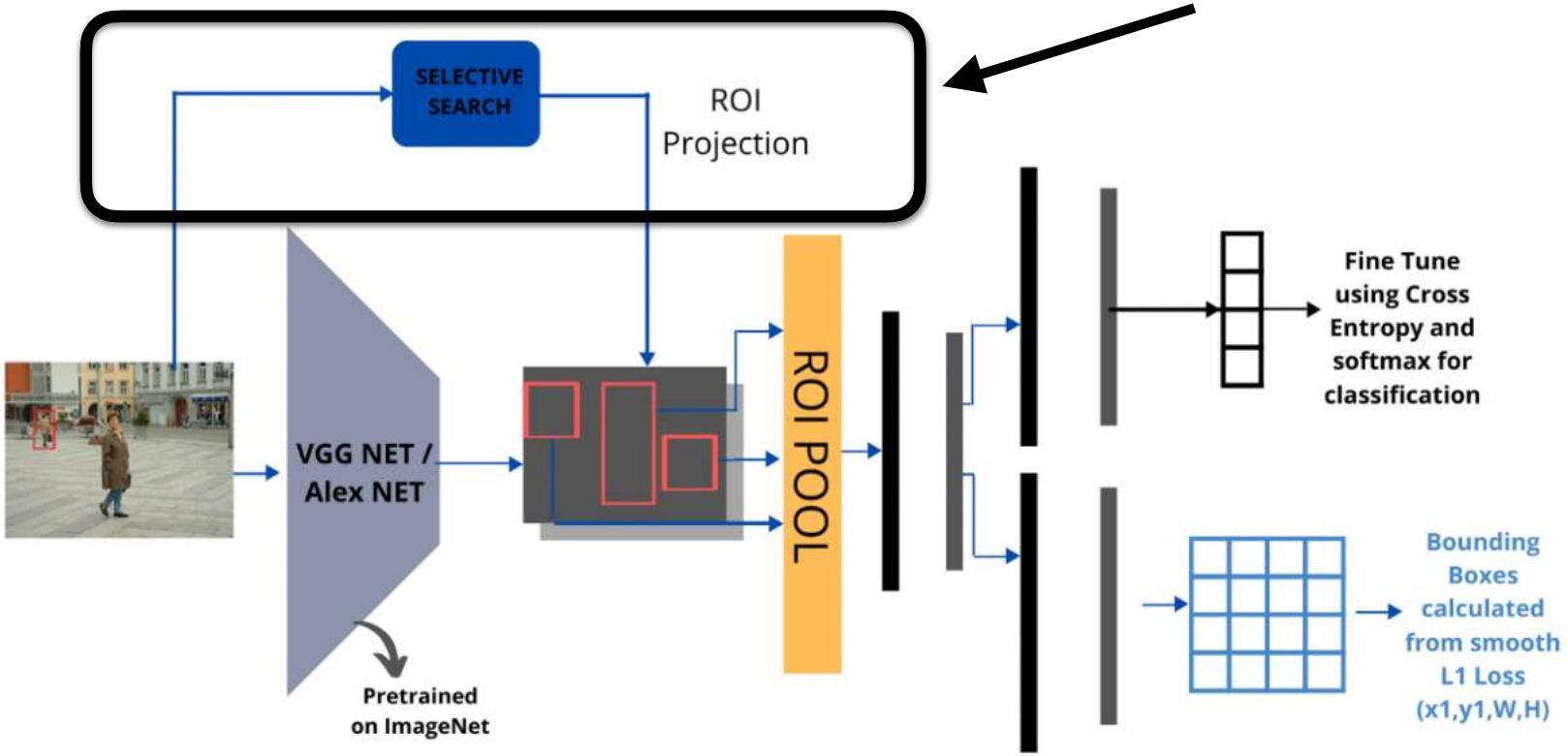


Fast R CNN uses only pass of the entire image by the CNN.


(0,0)
688 x 920
X
Y
(340,450)
320 x 128
1/16
43 x 58
20 x 8
(21,28)
Feature Maps
ROI Pooling
ROI Proposal

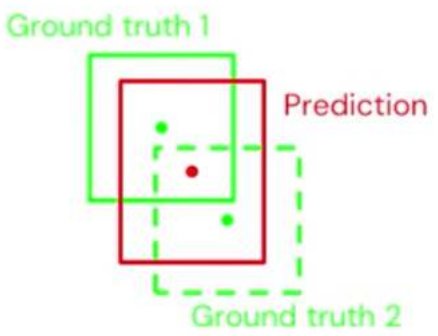
Fast R CNN

Why not
transforming this into a CNN?



WHAT WOULD BE THE LOSS FUNCTION OF SUCH A PROBLEM?

$$\frac{1}{N}\sum_{i=1}^{N}(y_i - p_i)^2$$

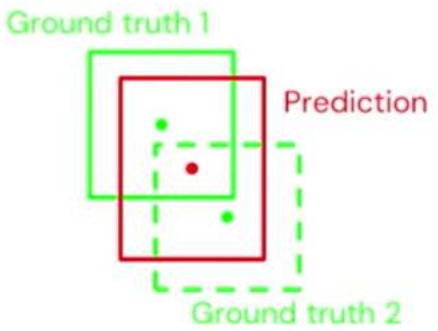


IT IS A REGRESSION WITH A SIMPLE QUADRATIC LOSS. WE TRY TO FIND THE BEST COORDINATES OF THE BOUNDING BOX.

@deepmind (V. Patraucean)

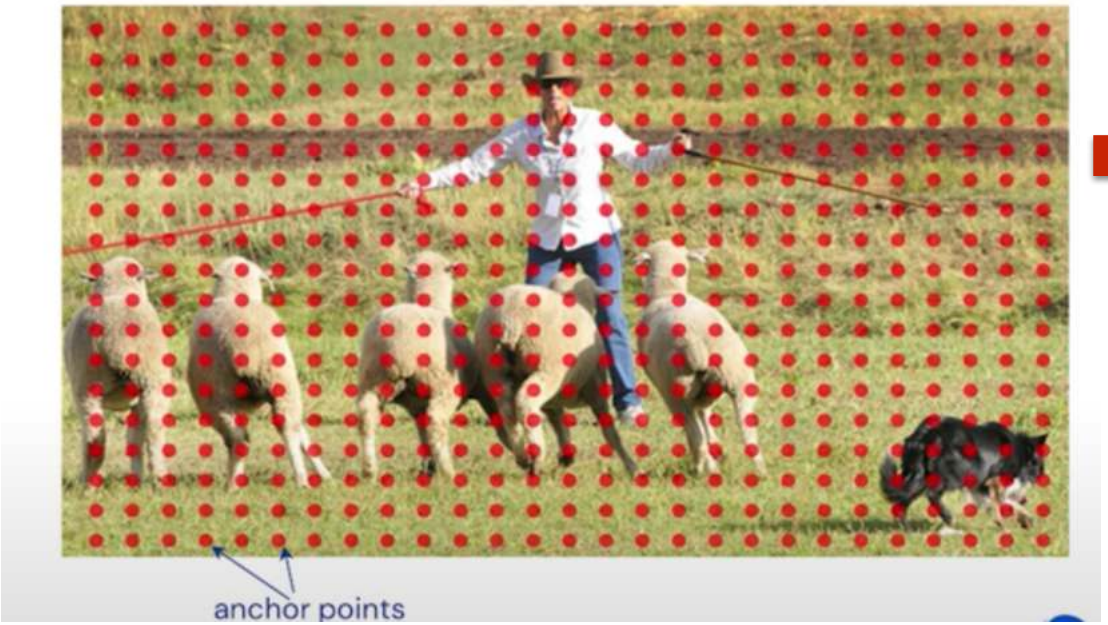
WHAT WOULD BE THE LOSS FUNCTION OF SUCH A PROBLEM?

$$\frac{1}{N}\sum_{i=1}^{N}(y_i - p_i)^2$$



IT IS A REGRESSION WITH A SIMPLE QUADRATIC LOSS. WE TRY TO FIND THE BEST COORDINATES OF THE BOUNDING BOX.

IT BECOMES MESSY QUITE RAPIDLY…



Discretize Bounding Box Space

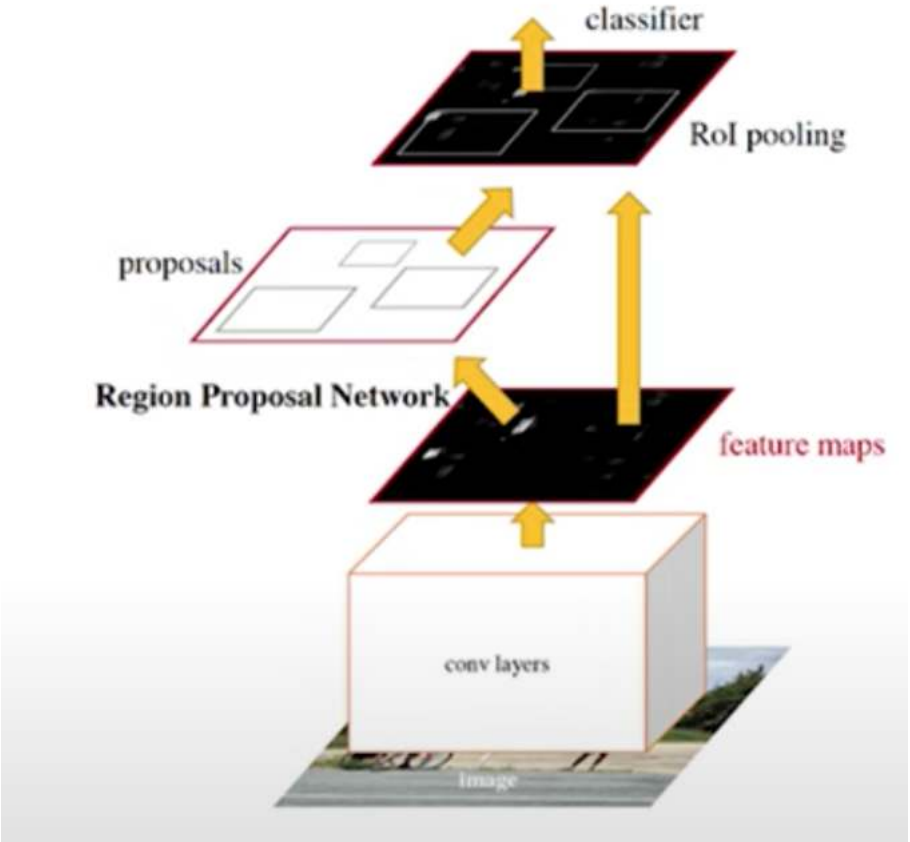
Choose n candidates per position

Predict objectness score (classification)

Sort and keep top K

@deepmind (V. Patraucean)

FASTER R-CNN



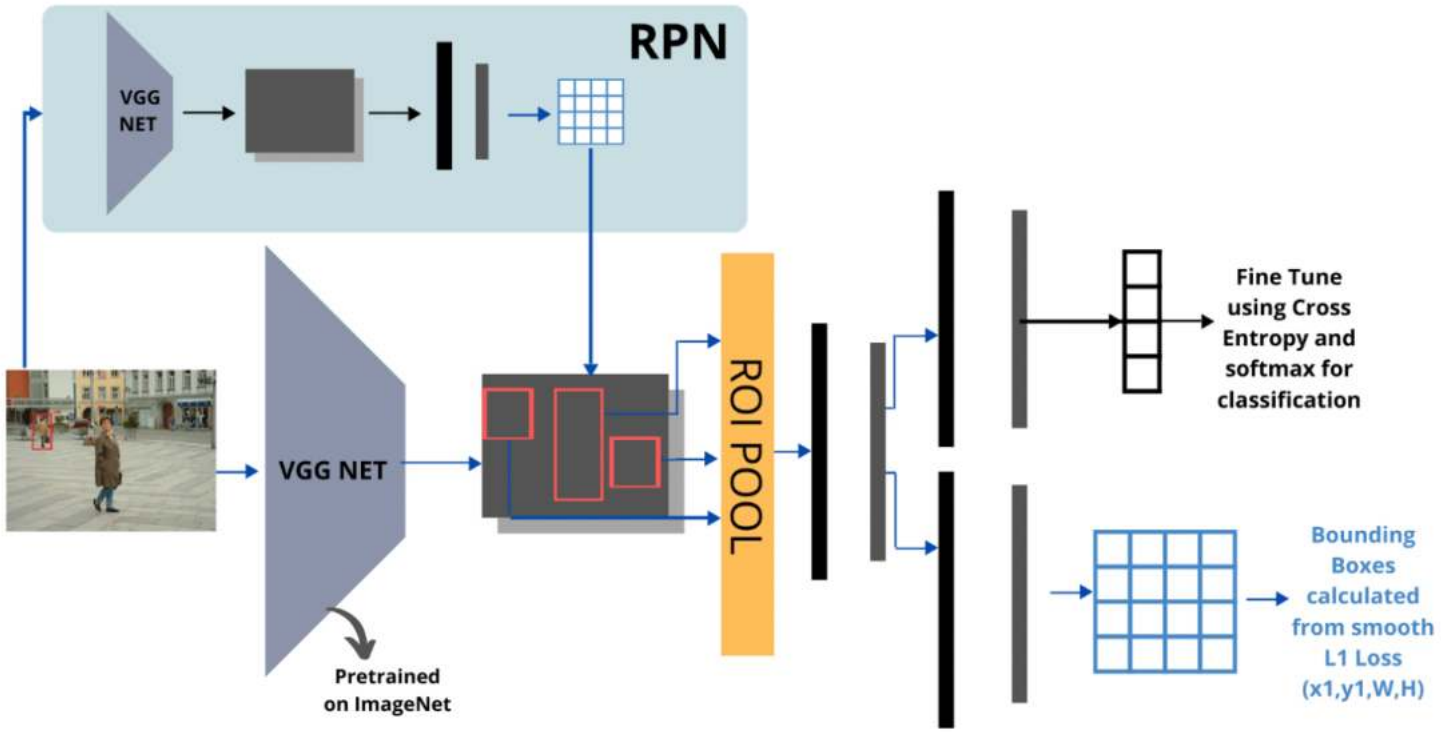
We divide the task in 2 steps:

1. Identify bounding box candidates (CLASSIFICATION)

2. Classify and Refine (REGRESSION)

Ren+16

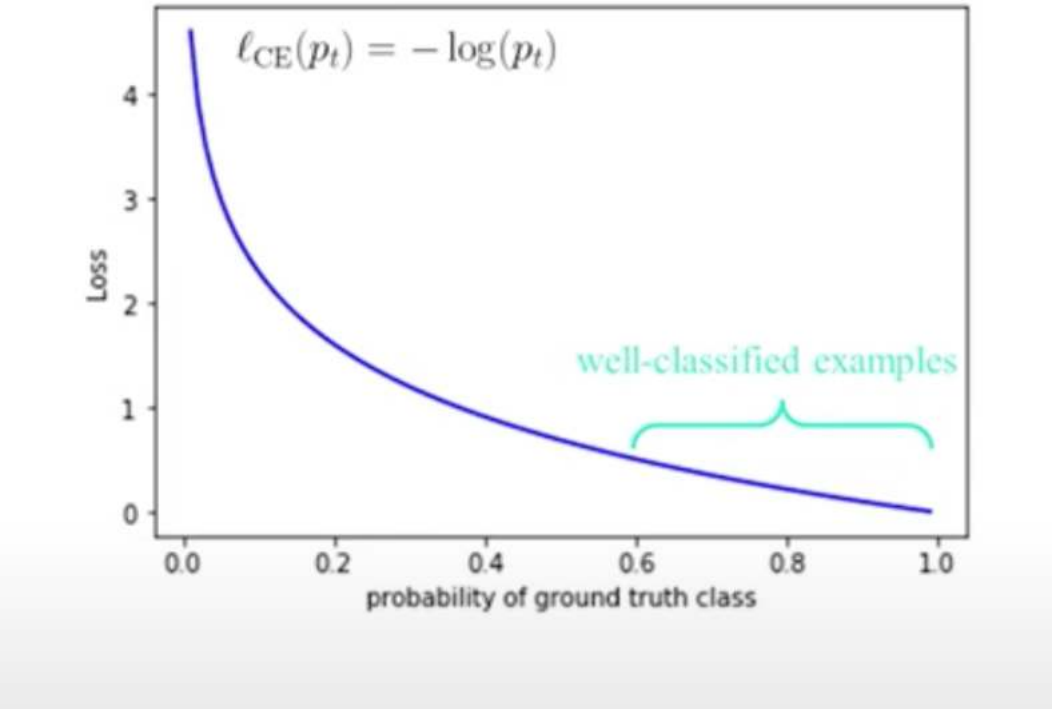
FASTER R-CNN


RPN
VGG NET
VGG NET
Pretrained on ImageNet
ROI POOL
Fine Tune using Cross Entropy and softmax for classification
Bounding Boxes calculated from smooth L1 Loss (x1,y1,W,H)

WHY NOT DOING IT ONE STAGE?

MOST OF THE CANDIDATES
ARE BACKGROUND,
EASY TO IDENTIFY

THE LOSS OF THE MANY
EASY EXAMPLES
DOMINATES OVER THE
RARE USEFUL ONES



@deepmind (V. Patraucean)

YOLO: You Only Look Once

Transform everything in a big regression

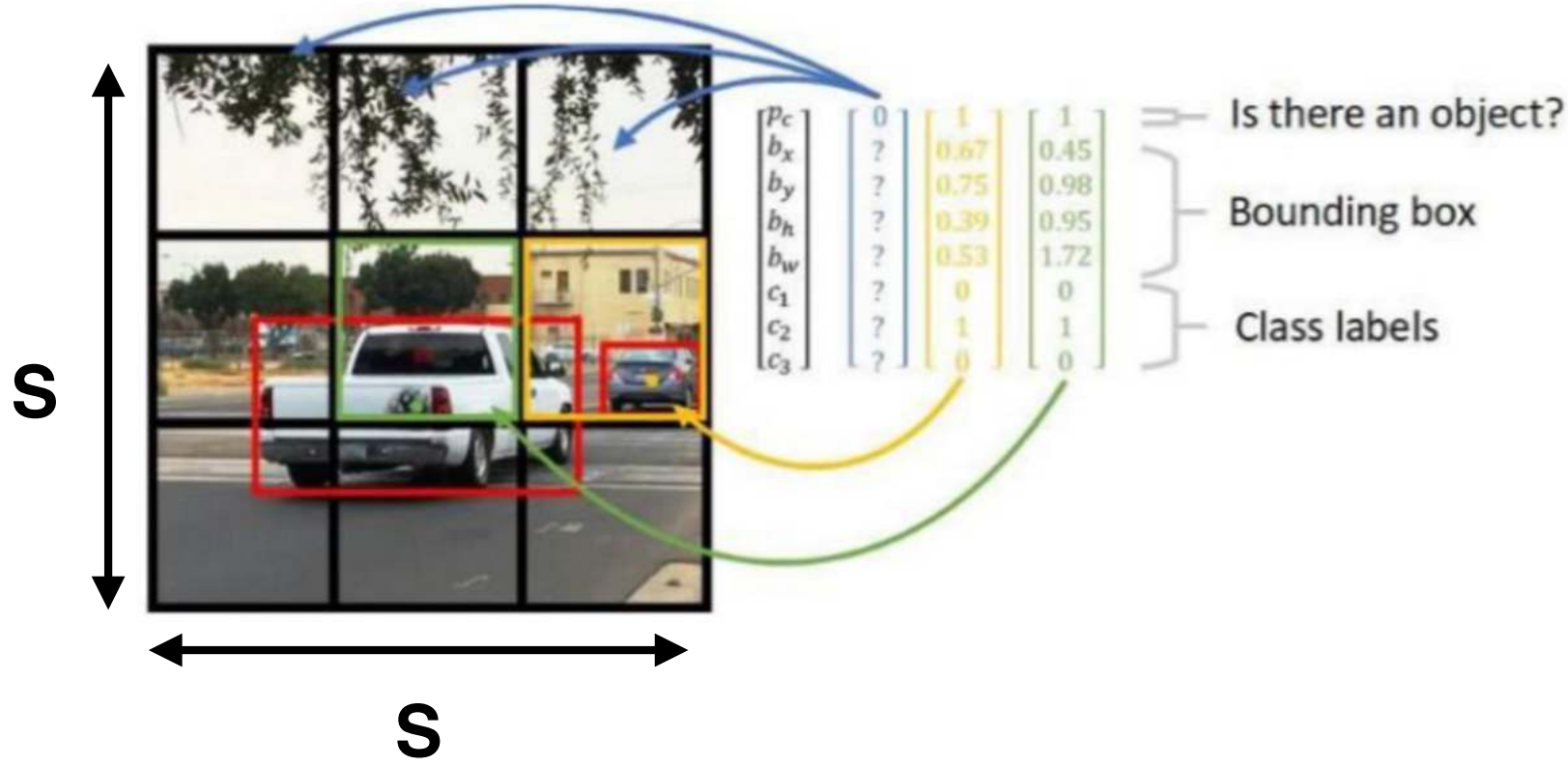
S × S grid on input

1. Divide the image in cells

2. Each cell is responsible for detecting B bounding boxes as well as a confidence class for each box

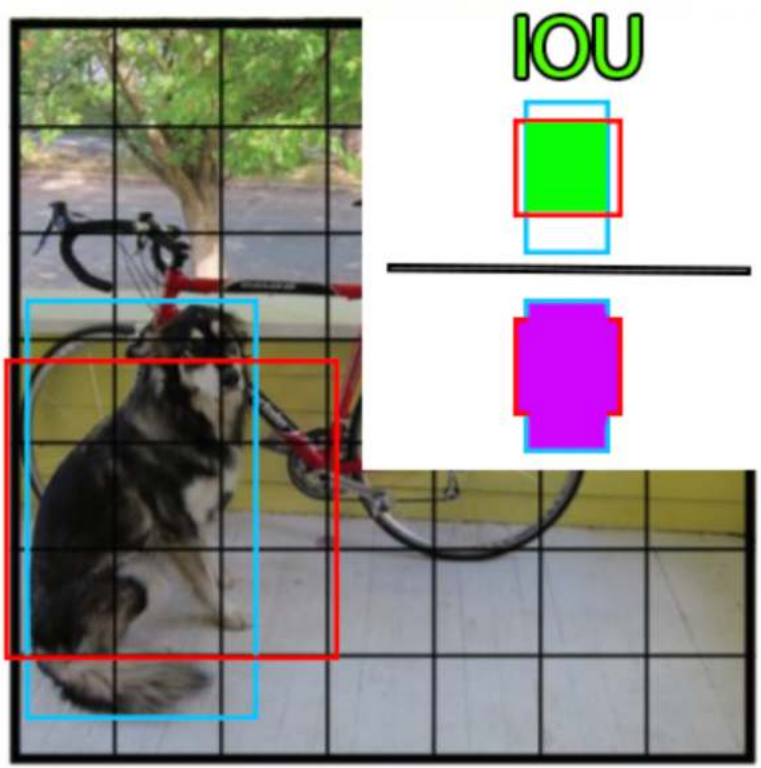
Each bounding box is parametrised with 5 numbers:
- coordinates: x,y
- heights width: w, h
- confidence: c

Then we attach to each bounding box, a one-hot encoder vector contains the class labels

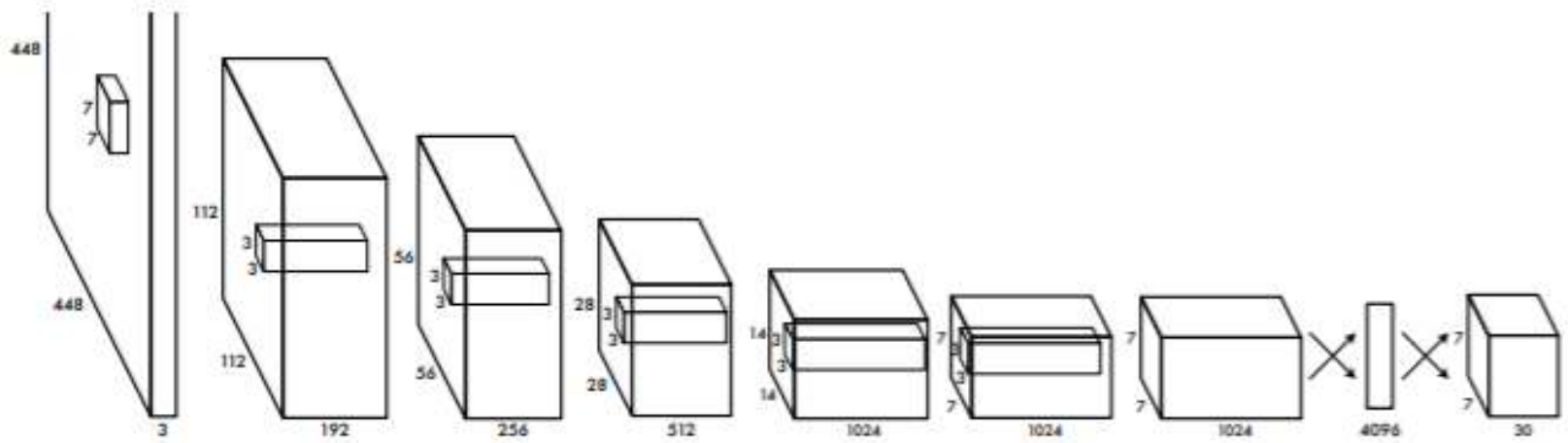


Measuring the “confidence” of a box

WE TYPICALLY USE THE INTERSECTION OVER UNION LOSS



YOLO REGRESSION NETWORK



$$S \times S \times (C + B * 5)$$

$$\lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[(x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2 \right]$$

$$+ \lambda_{\textbf{coord}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left[\left(\sqrt{w_i} - \sqrt{\hat{w}_i} \right)^2 + \left(\sqrt{h_i} - \sqrt{\hat{h}_i} \right)^2 \right]$$

$$+ \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{obj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \lambda_{\text{noobj}} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{\text{noobj}} \left(C_i - \hat{C}_i \right)^2$$

$$+ \sum_{i=0}^{S^2} \mathbb{1}_{i}^{\text{obj}} \sum_{c \in \text{classes}} \left(p_i(c) - \hat{p}_i(c) \right)^2$$

$$\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} (x_i - \hat{x}_i)^2 + (y_i - \hat{y}_i)^2$$

loss related to the predicted bounding box position (x,y)

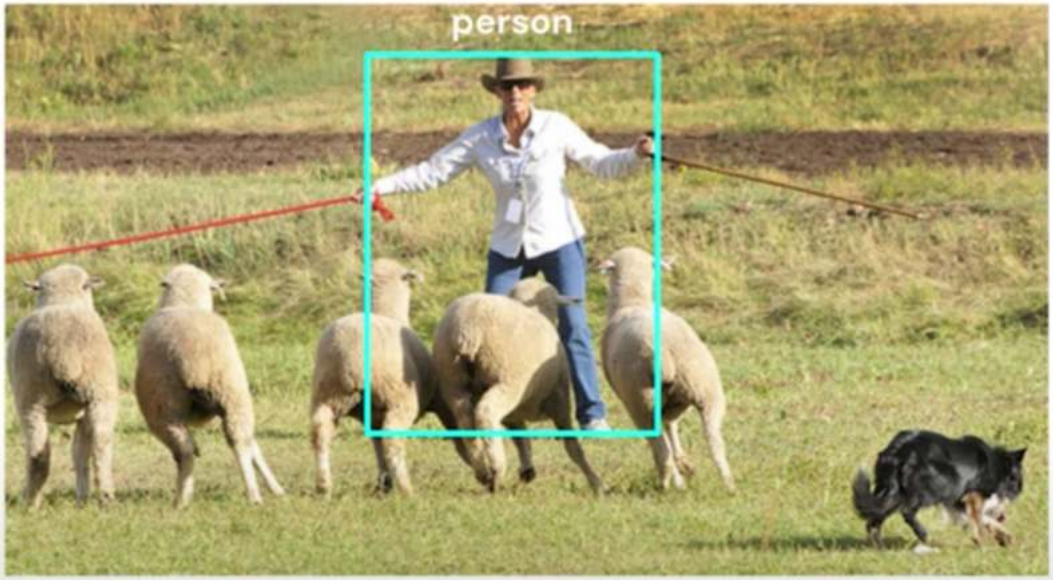
$$\lambda_{coord} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} (\sqrt{w_i} - \sqrt{\hat{w}_i})^2 + (\sqrt{h_i} - \sqrt{\hat{h}_i})^2$$

loss related to the predicted bounding box height and width

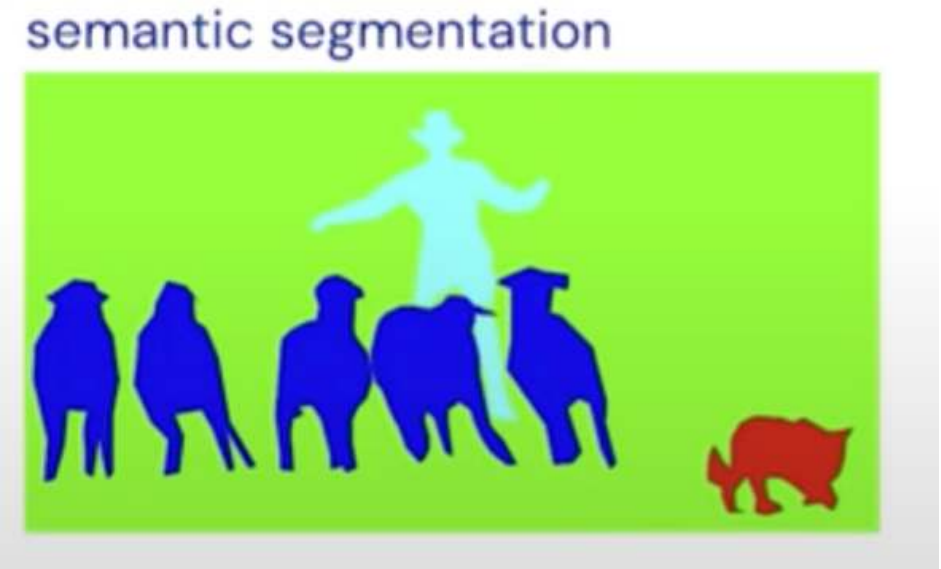
$$\sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{obj} (C_i - \hat{C}_i)^2 + \lambda_{noobj} \sum_{i=0}^{S^2} \sum_{j=0}^{B} \mathbb{1}_{ij}^{noobj} (C_i - \hat{C}_i)^2$$

loss related to the correctness of the bounding box

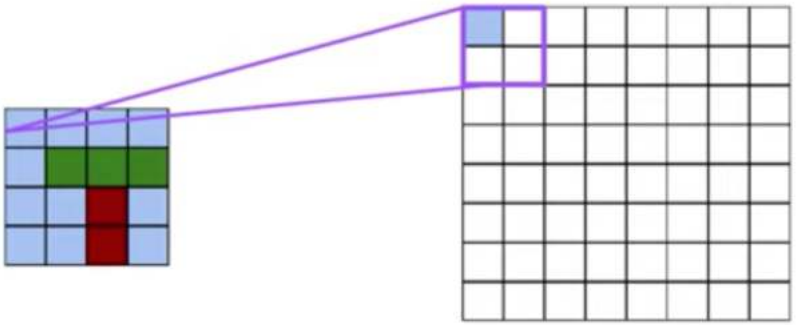
LET'S GO A STEP FURTHER INTO SEMANTIC SEGMENTATION



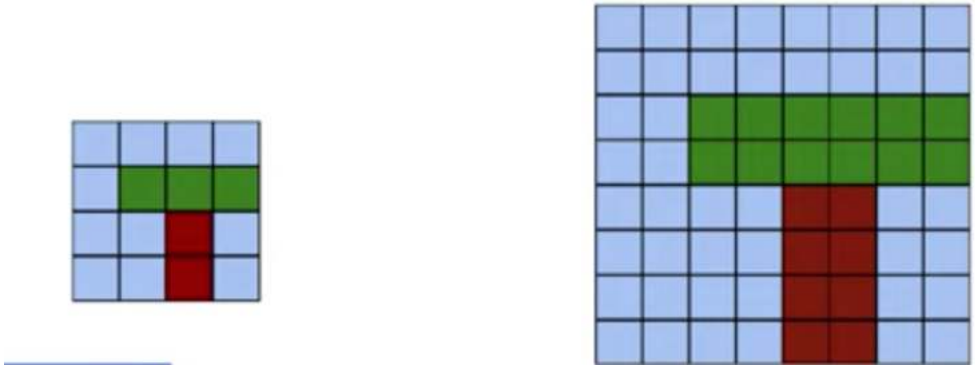
BOUNDING BOXES ARE NOT ALWAYS GOOD REPRESENTATIONS



UNPOOLING OPERATION (INVERSE OF POOLING)



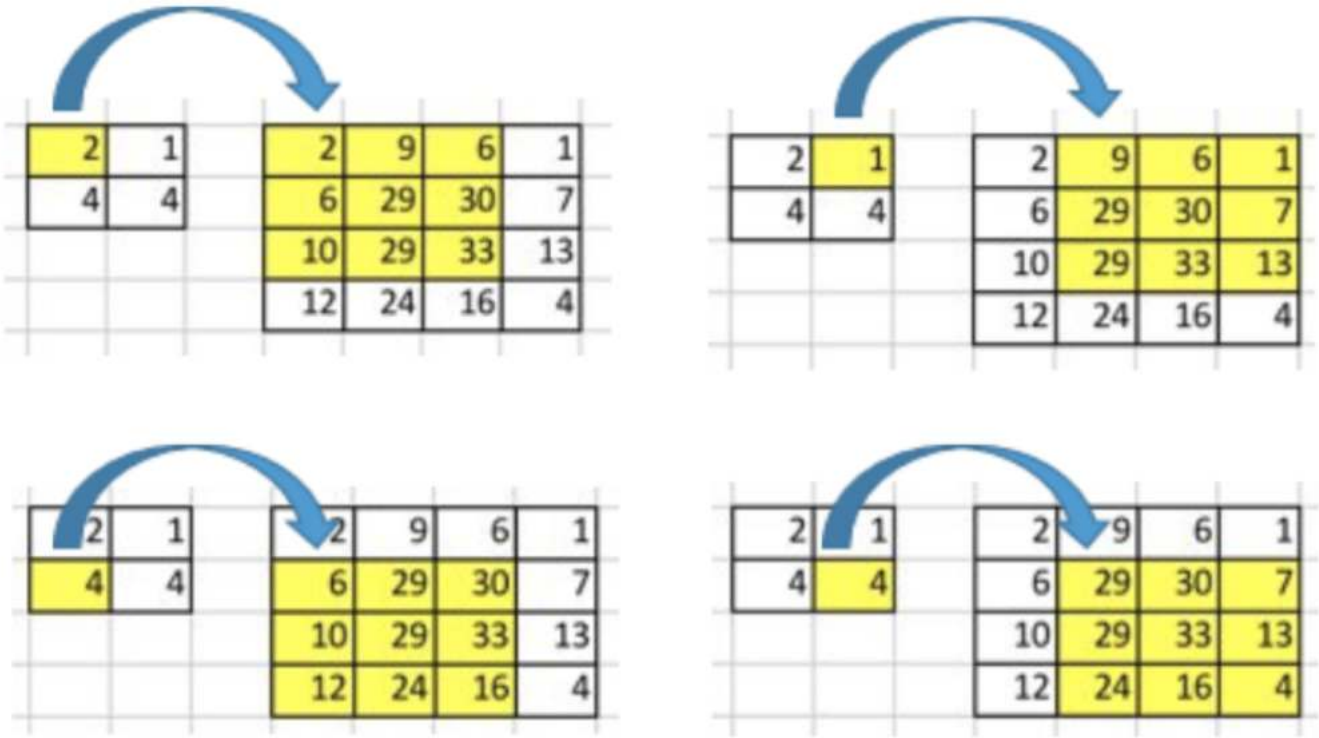
COPY PIXELS IN A GIVEN WINDOW



GENERATES LARGER IMAGES FROM SMALLER ONES

TRANSPOSED CONVOLUTION

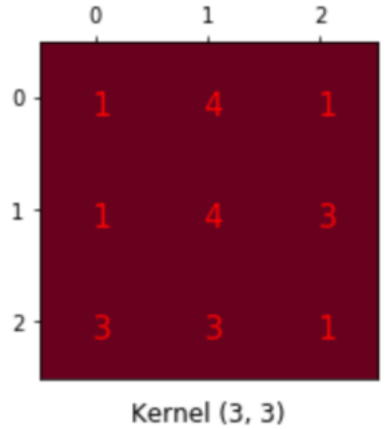
ALLOWS TO INCREASE THE SIZE



Going Backward of Convolution

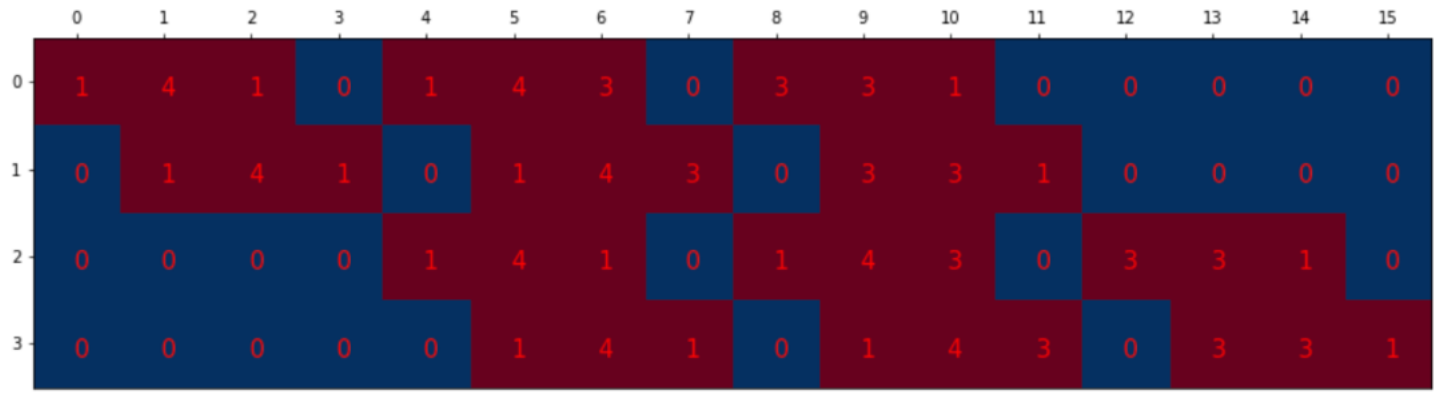
EXAMPLE TAKEN FROM HERE

CONVOLUTION MATRIX



Kernel (3, 3)

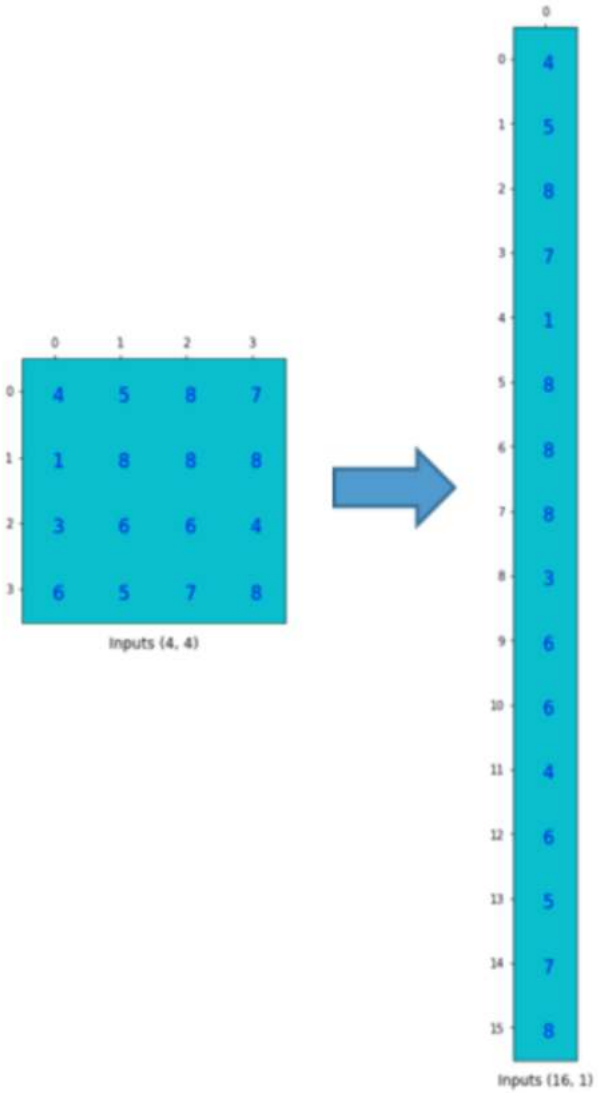
THE KERNEL CAN BE ARRANGED IN FORM OF A MATRIX:



Convolution Matrix (4, 16)

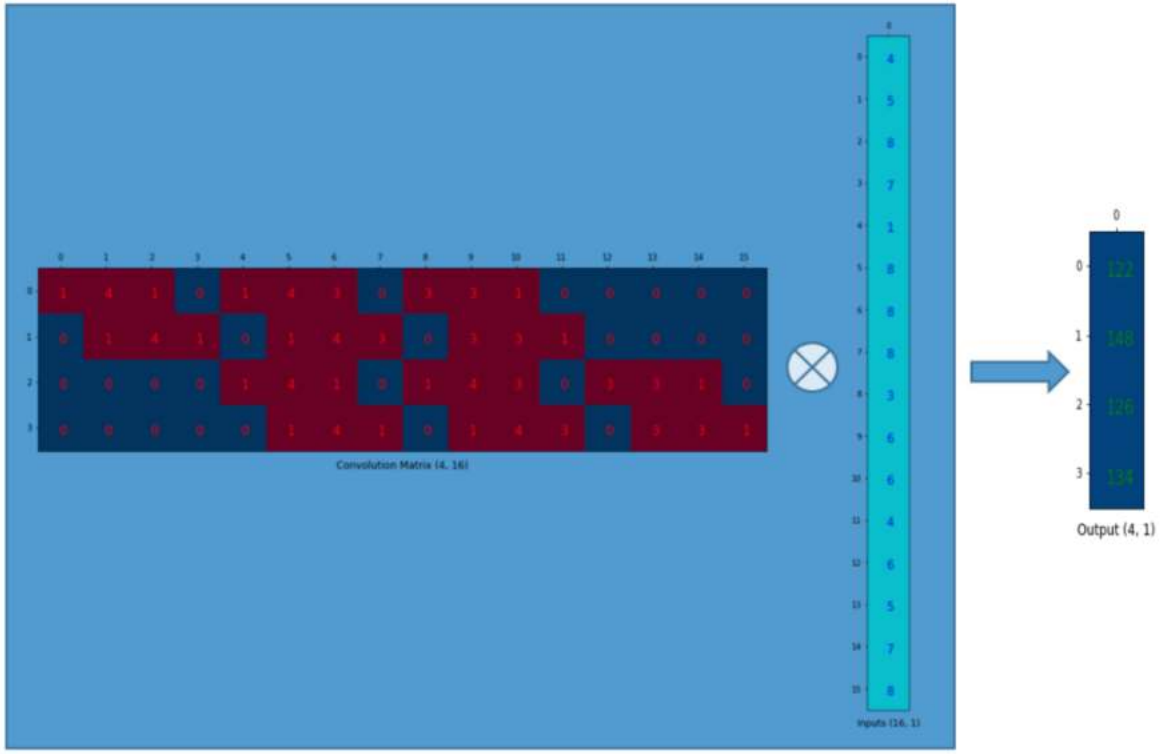
EXAMPLE TAKEN FROM HERE

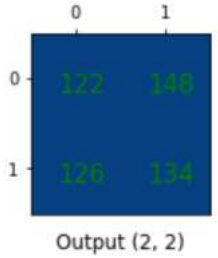
THE INPUT IS FLATTENED INTO A COLUMN VECTOR



EXAMPLE TAKEN FROM HERE

THE CONVOLUTION IS TRANSFORMED INTO A PRODUCT OF MATRICES

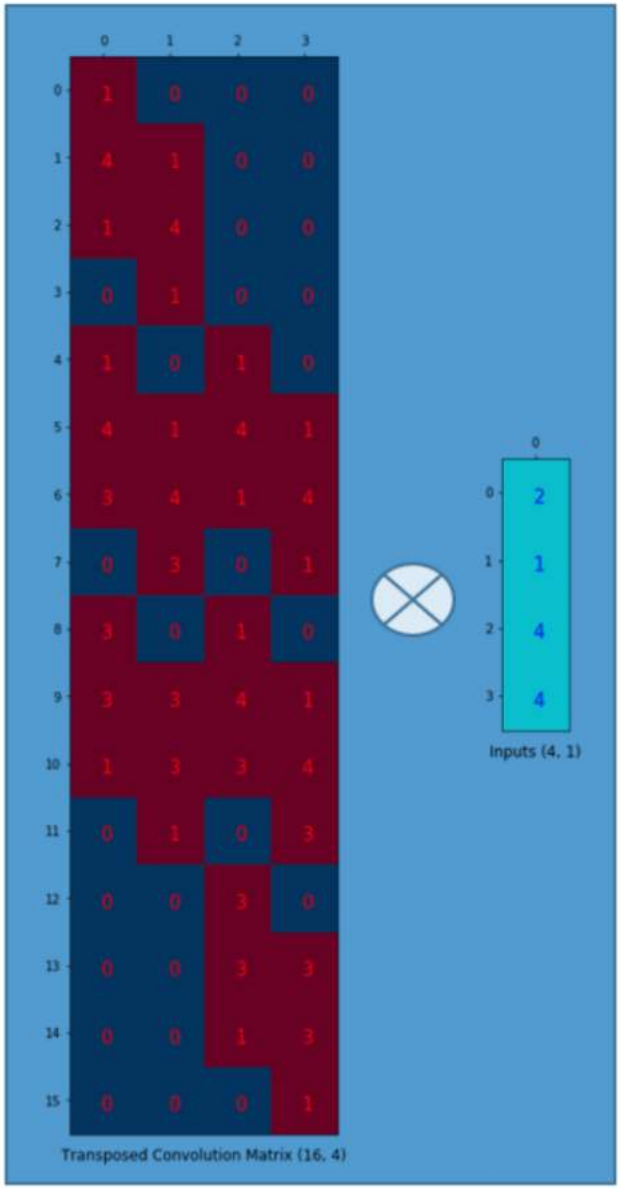


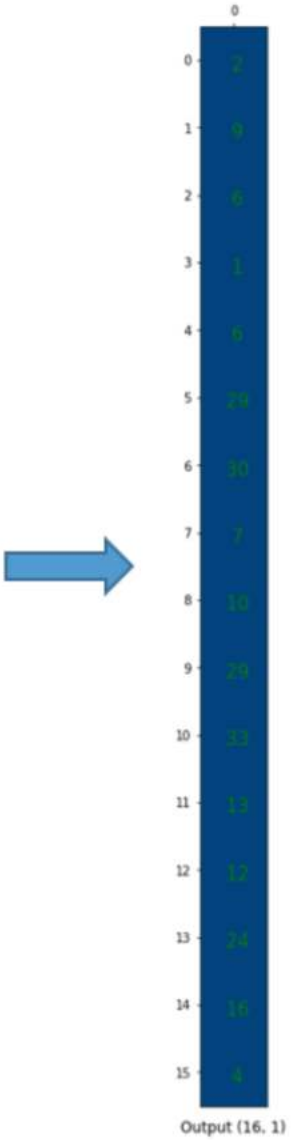


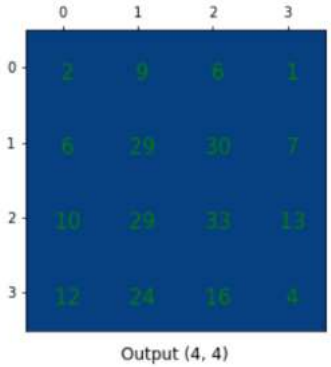
RESHAPED OUTPUT

EXAMPLE TAKEN FROM HERE

THE TRANSPOSED CONVOLUTION IS THE INVERSE OPERATION



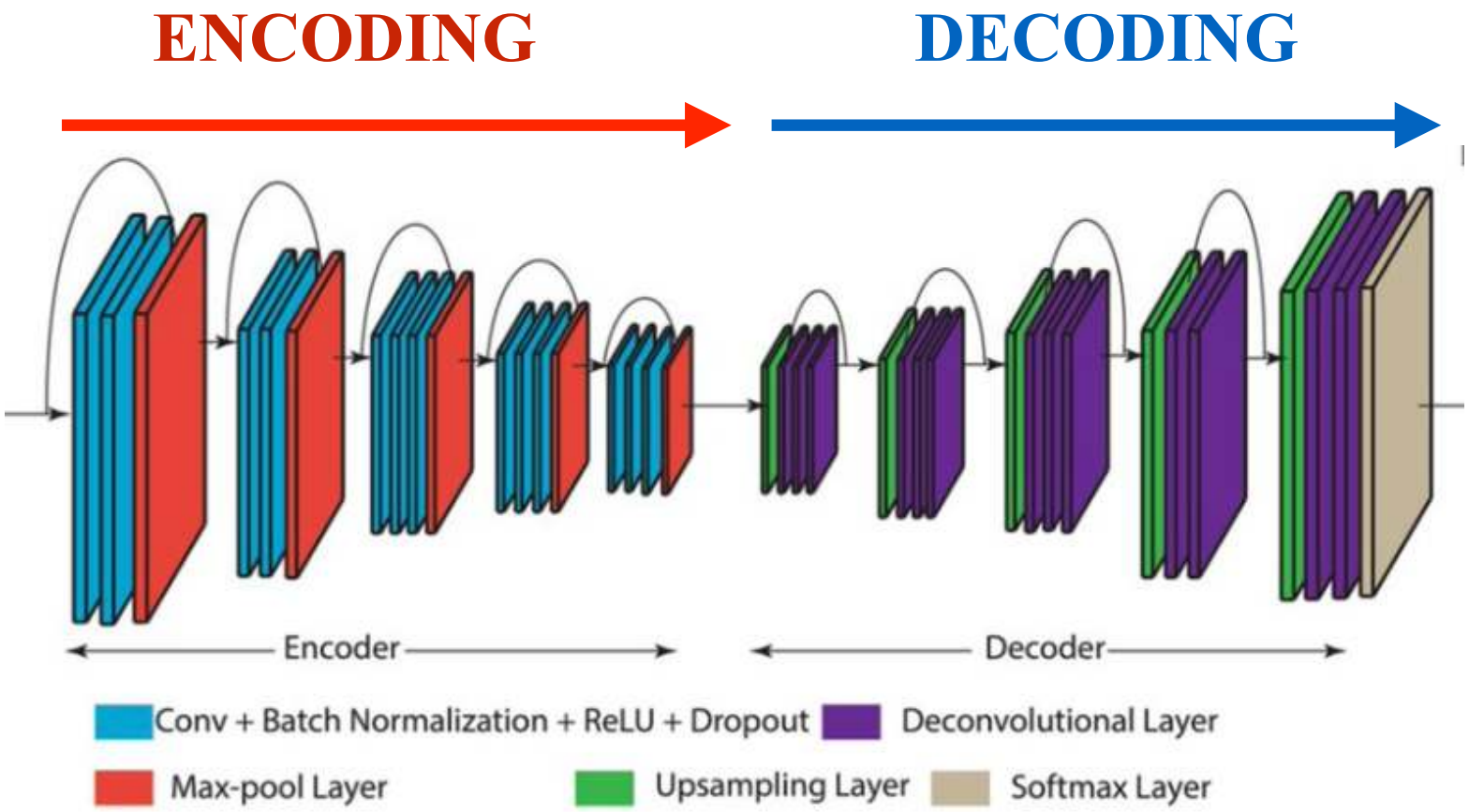




RESHAPED OUTPUT

EXAMPLE TAKEN FROM HERE

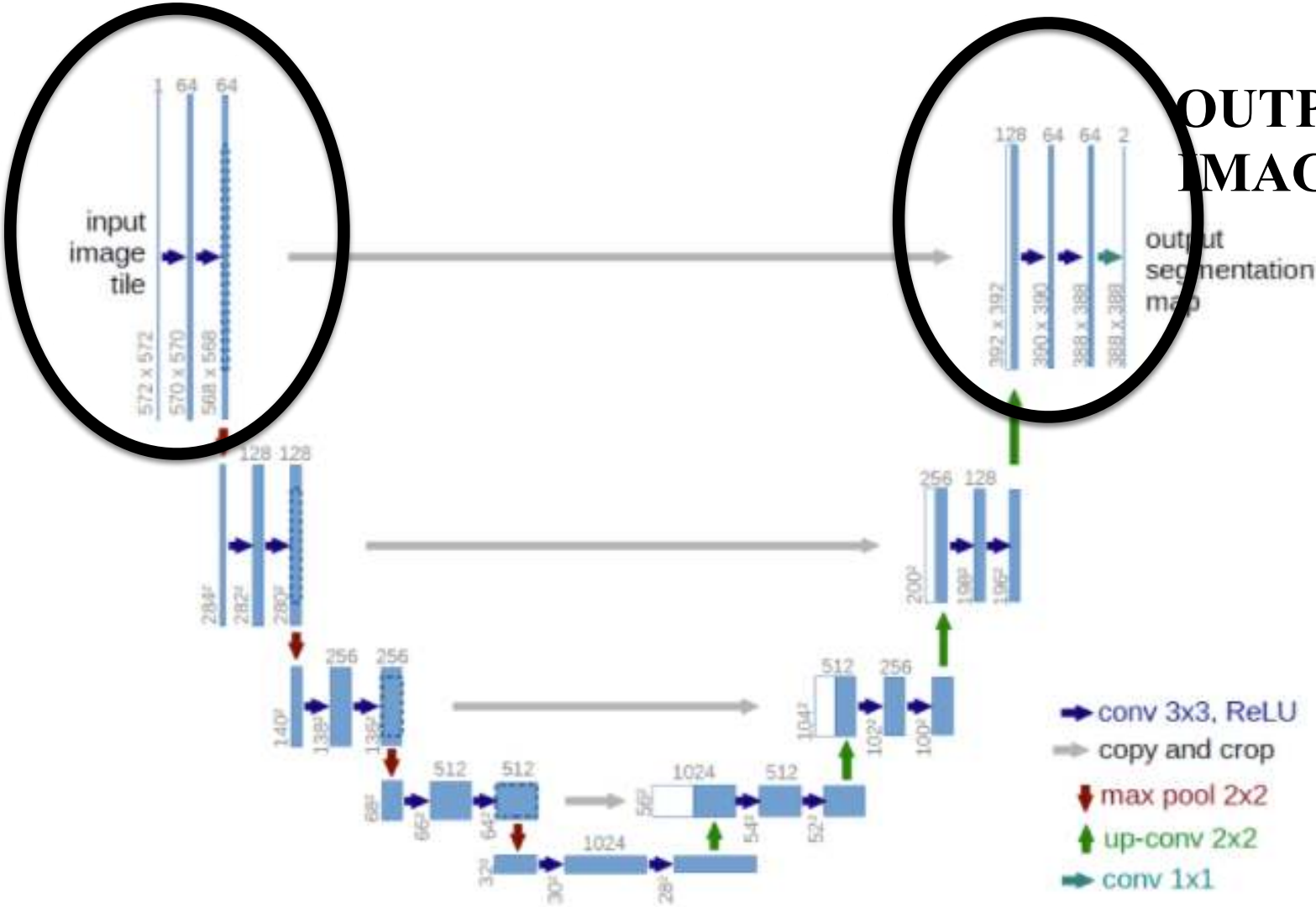
ENCODER-DECODERS GO FROM IMAGE 2 IMAGE



WE CALL THIS FULLY CONVOLUTIONAL NEURAL NETWORKS

ENCODING-DECODING TO EXTRACT IMAGE FEATURES: U-NET

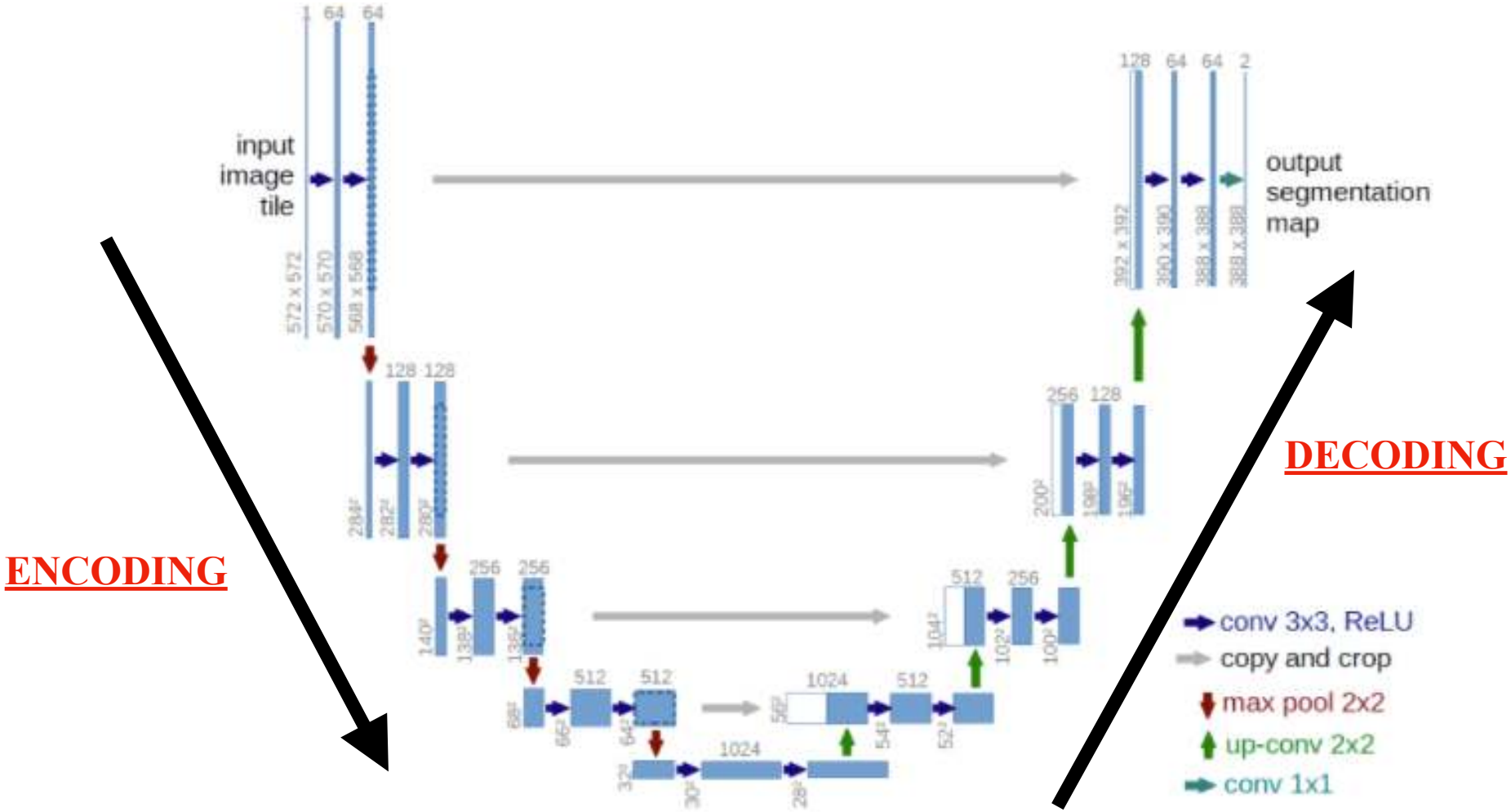
INPUT IMAGES



OUTPUT IMAGES

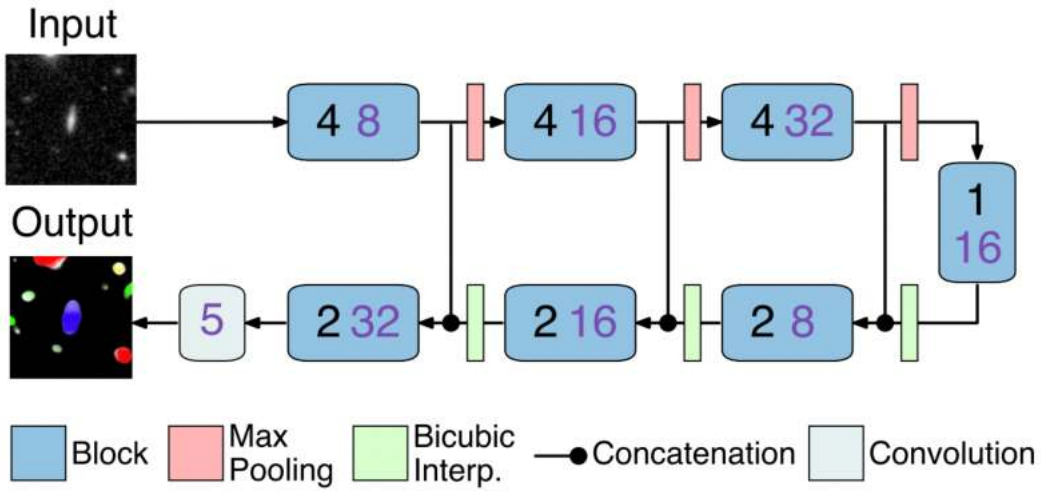
Ronnerberger+15

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

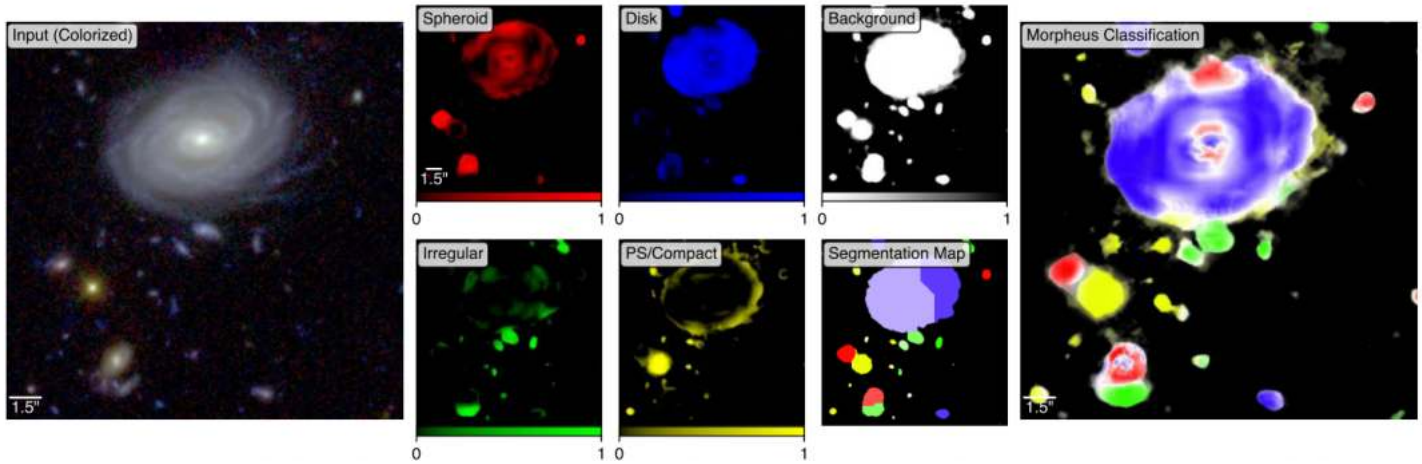


Ronnerberger+15

2. Segmentation



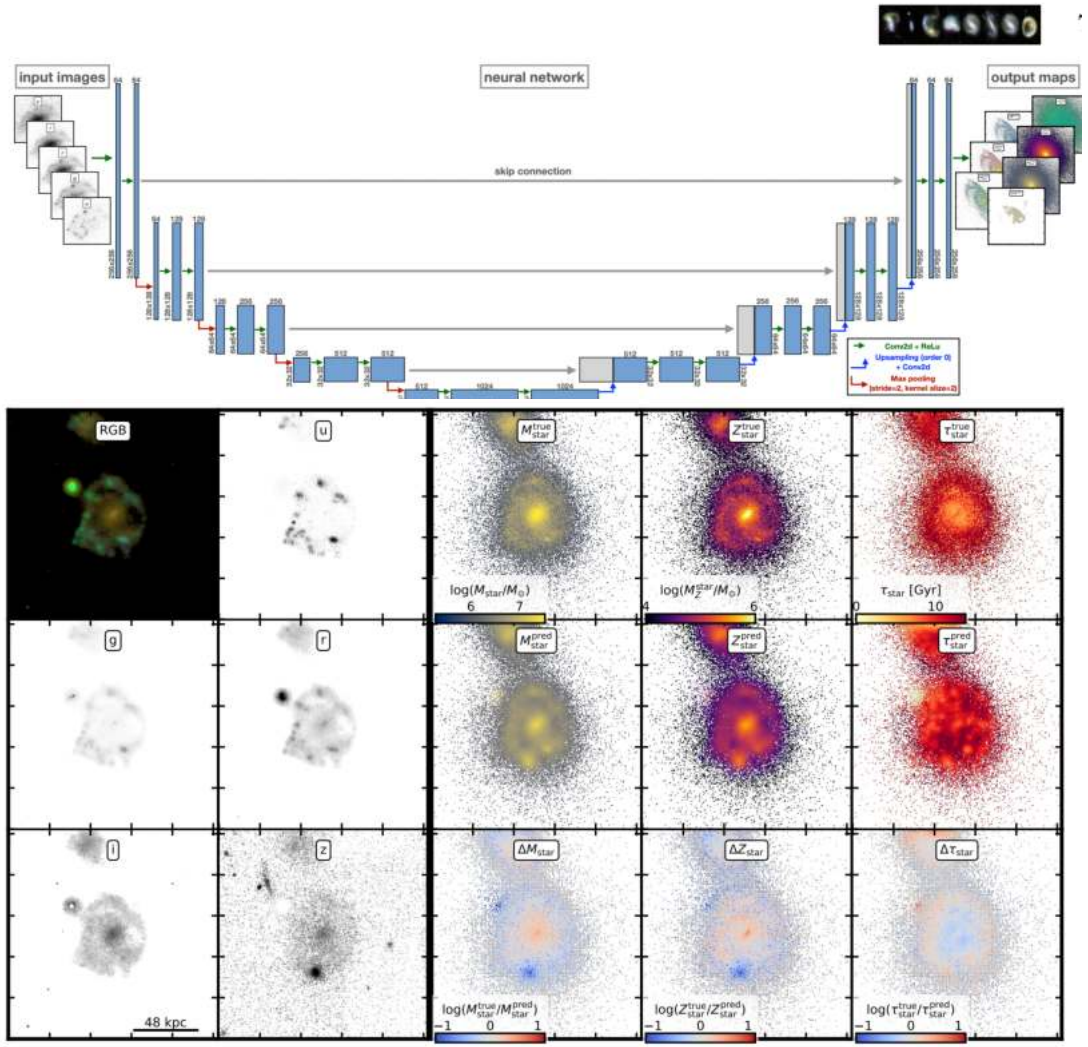
Classification at the pixel level



Hausen+20

Dieleman+15, Huertas-Company+15, Aniyan+17, Charnock+17, Gieseke+17, Jacobs+17, Petrillo+17, Schawinski+17, Alhassan+18, Dominguez-Sanchez+18, George+18, Hinners+18, Lukic+18, Moss+18, Razzano+18, Schaefer+18, Allen+19, Burke+19, Carrasco-Davis+19, Chatterjee+19, Davies+19, Dominguez-Sanchez+19, Fusse+19, Glaser+19, Ishida+19, Jacobs+19, Katebi+19, Lanusse+19Liu+19, Lukic+19, Metcalf+19, Muthukrishna+19, Petrillo+19, Reiman+19, Boucaud+20, Chiani+20, Ghosh+20, Gomez+20, Hausen+20, Hlozek+20, Hosseinzadeh+20. Huang+20, Li+20, Moller+20, Paillassa+20, Tadaki+20, Vargas dos Santos+20, Walmsley+20, Wei+20, Allam+21, Arcelin+21, Becker+21, Bretonniere+21, Burhanudin+21, Davison+21, Donoso-Oliva+21, Jia+21, Lauritsen+21, Ono+21, Ruan+21, Sadegho+21, Tang+21, Tanoglidis+21, Vojtekova+21, Dhar+22Hausen+22, Orwat+22, Pimentel+22, Rezaei+22, Samudre+22, Shen+22, Walmsley+22

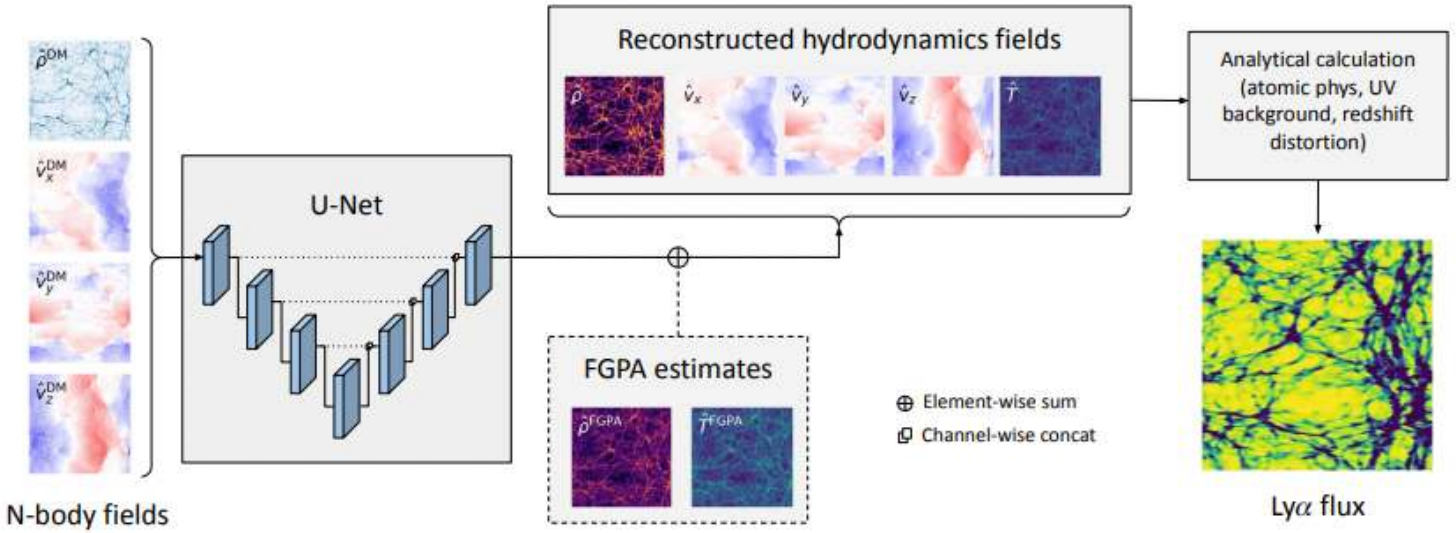
Stellar Populations



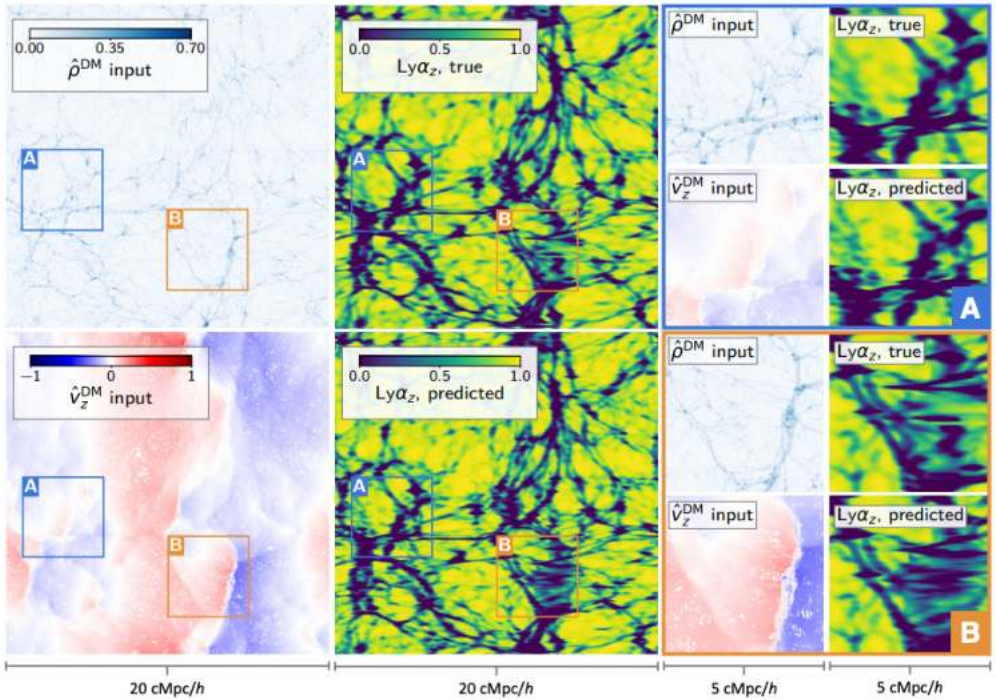
Buck+21

Hazevah+17, Perreault-Levasseur+17, Morningstar+18, Stark+18, Tuccillo+18, Bom+19, Lovell+19, Madireddy+19, Pearson+19, Simet+19, Pasquet+19, Wu+19, Menou+19, Aragon-Calvo+20, Chianese+20, Stahl+20, Surana+20, Shuntov+20, Cabayol-Garcia+20, Campagne+20, Buck+21, Grover+21, Hayat+21, Li+21, Maresca+21, Qiu+21, Rhea+21, Schuldt+21, Tohill+21, Yao-Yu+21, Dey+21, Lee+21, Zhou+21, Henghes+21, Ansari+21

Painting Baryons

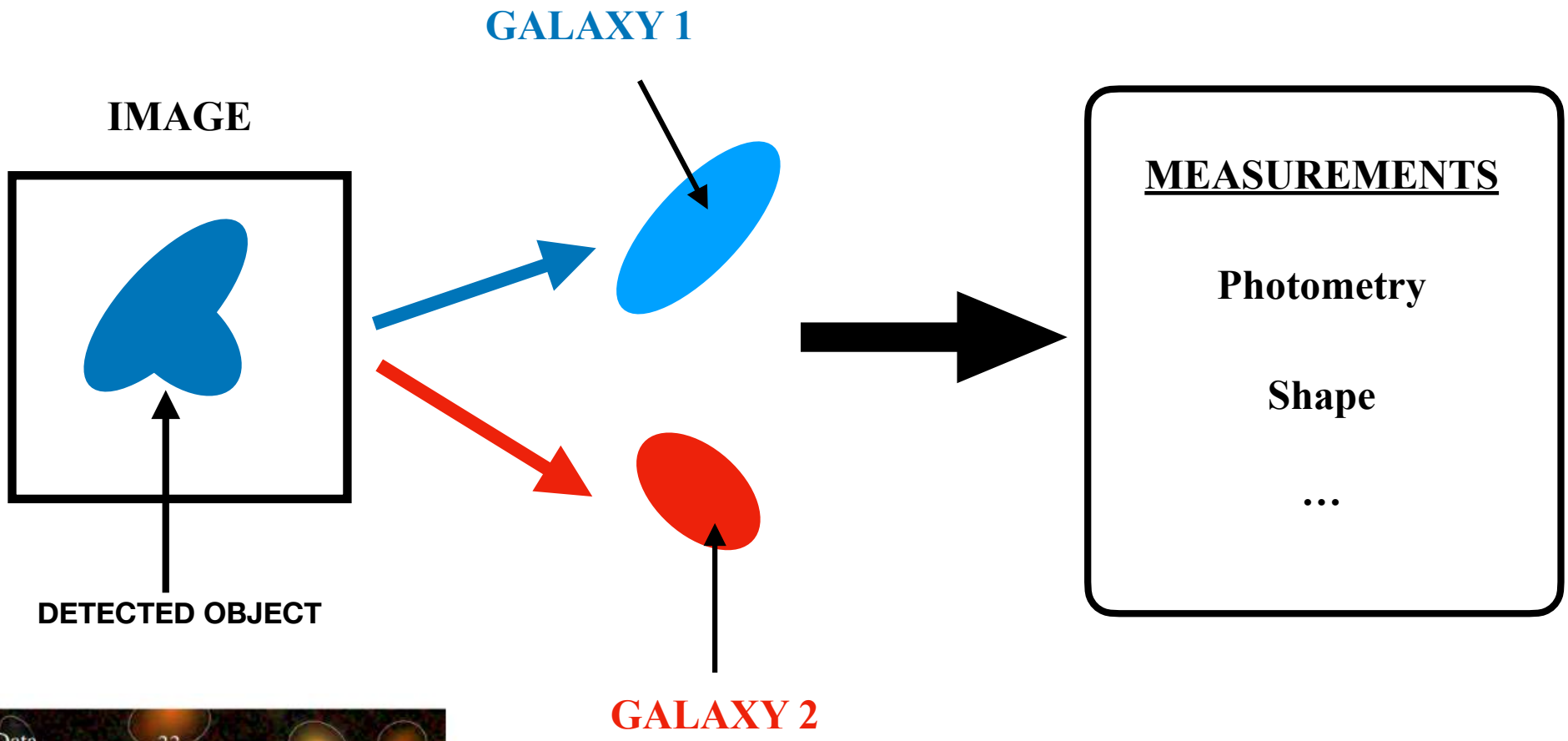


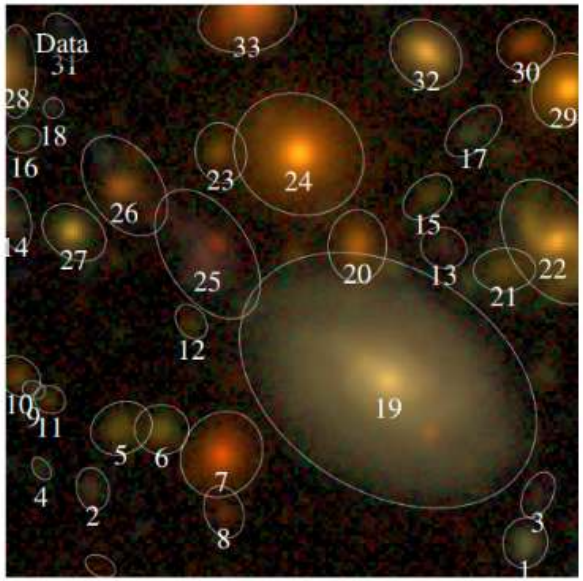
Harrington+21



Neural Networks are used to learn the non-linear mapping between cheap dark matter only simulations to expensive baryonic physics

Rodriguez+19, Modi+18, Berger+18, He+18, Zhang+19, Troster+19, Zamudio-Fernandez+19, Perraudin+19, Charnock+19, List+19, Giusarma+19, Bernardini+19, Chardin+19, Mustafa+19, Ramanah+20, Tamosiunas+20, Feder+20, Moster+20, Thiele+20, Wadekar+20, Dai+20, Li+20, Lucie-Smith+20, Kasmanoff+20, Ni+21, Rouhiainen+21, Harrington+21, Horowitz+21, Horowitz+21, Bernardini+21, Schaurecker+21, Etezad-Razavi+21, Curtis+21





Bosch+17

>50% of objects will be affected by blending in future deep surveys such as LSST

Boucaud+19

GALAXY 1

IMAGE

DETECTED OBJECT

GALAXY 2

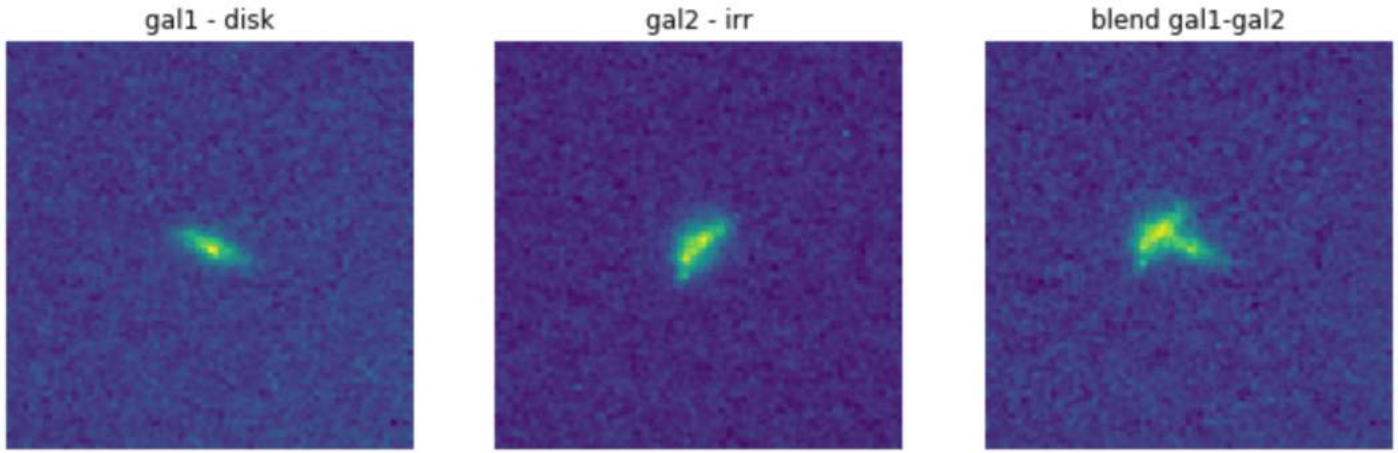
MEASUREMENTS

Photometry

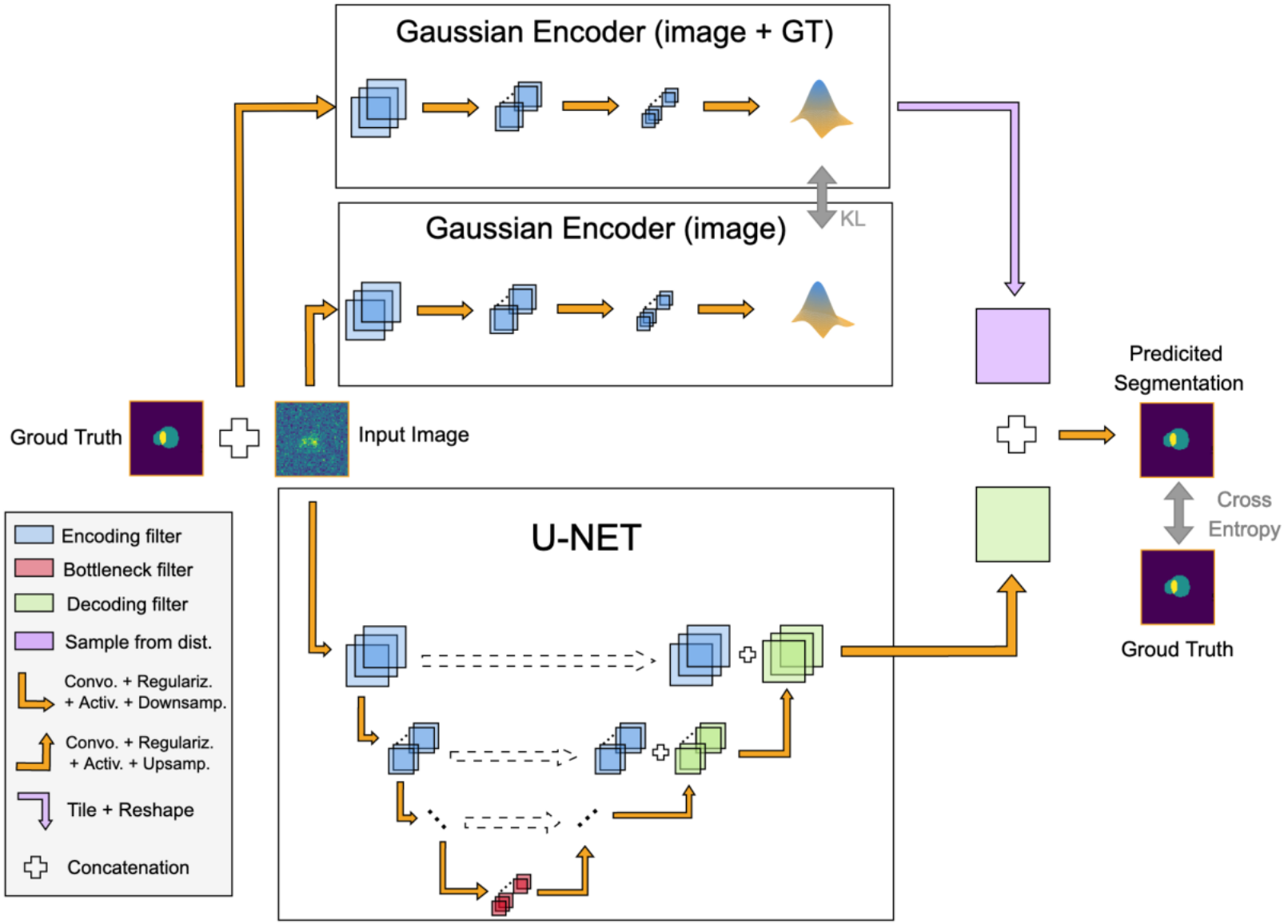
Shape

…

ISOLATED GALAXIES ARTIFICALLY BLENDED



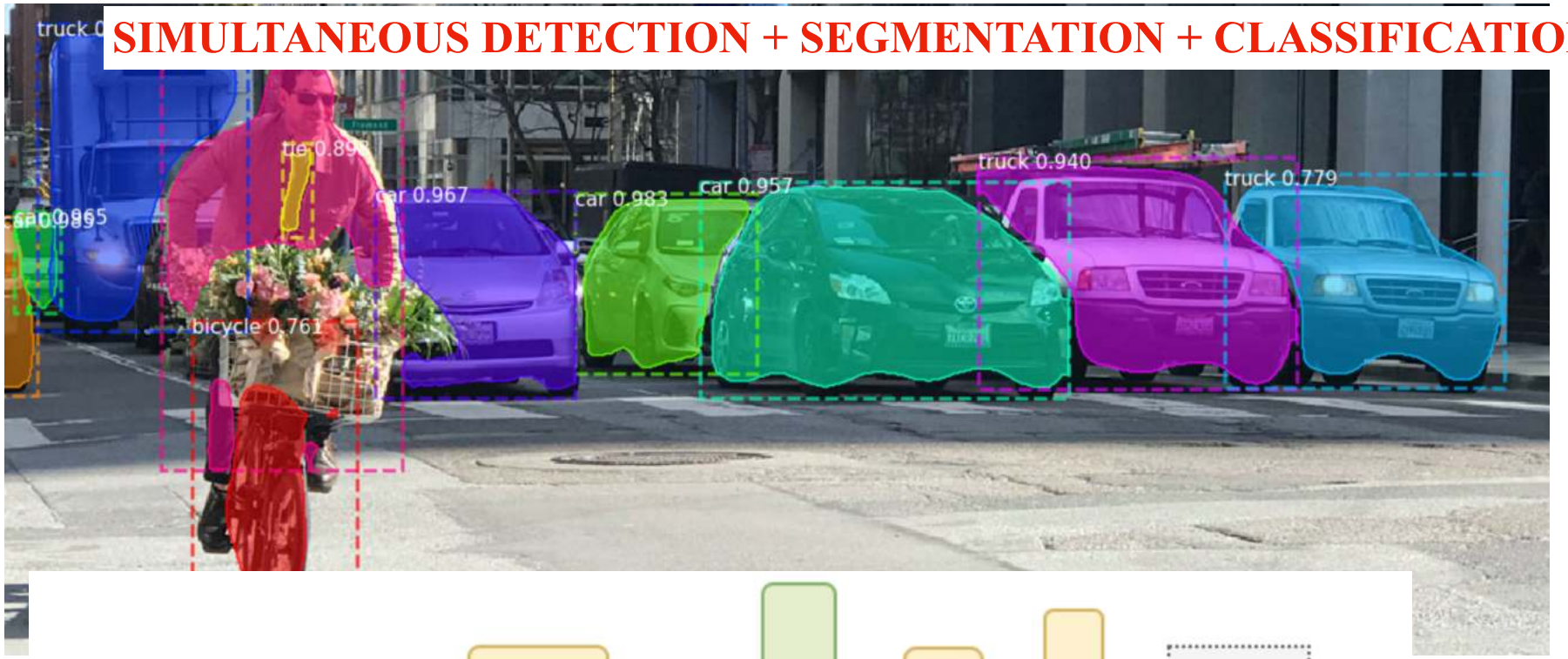
PROBABILISTIC U-NET

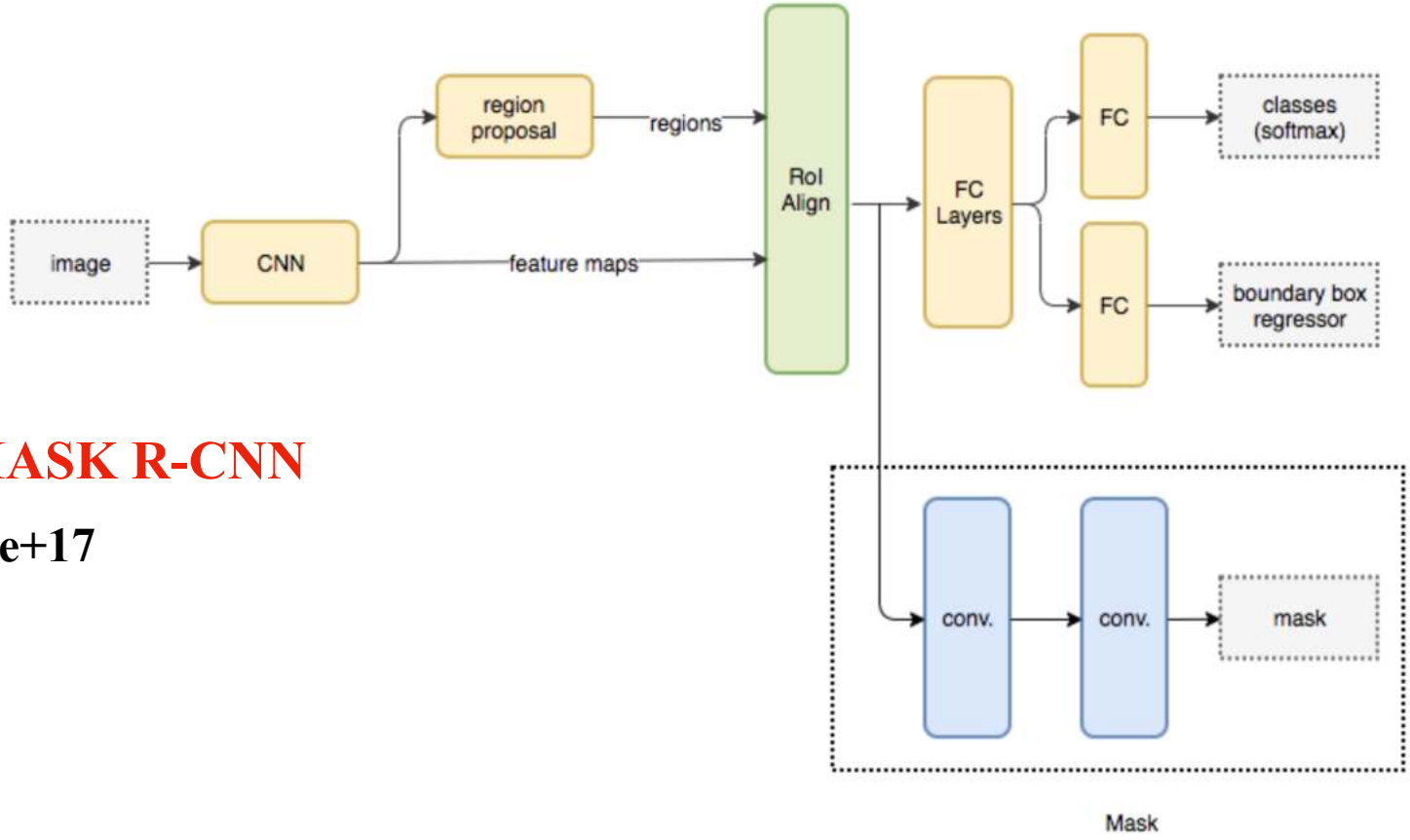


Bretonniere+20

instance segmentation

SIMULTANEOUS DETECTION + SEGMENTATION + CLASSIFICATION





MASK R-CNN

He+17