



دانشگاه صنعتی امیرکبیر  
( پلی تکنیک تهران )

# AI IN ROBOTICS – SEGMENTATION

PRESENTER : HASSAN YOUSEFZADE

## Classification



CAT

## Semantic Segmentation



GRASS, CAT, TREE, SKY

## Object Detection

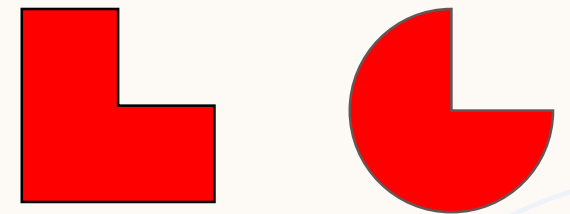
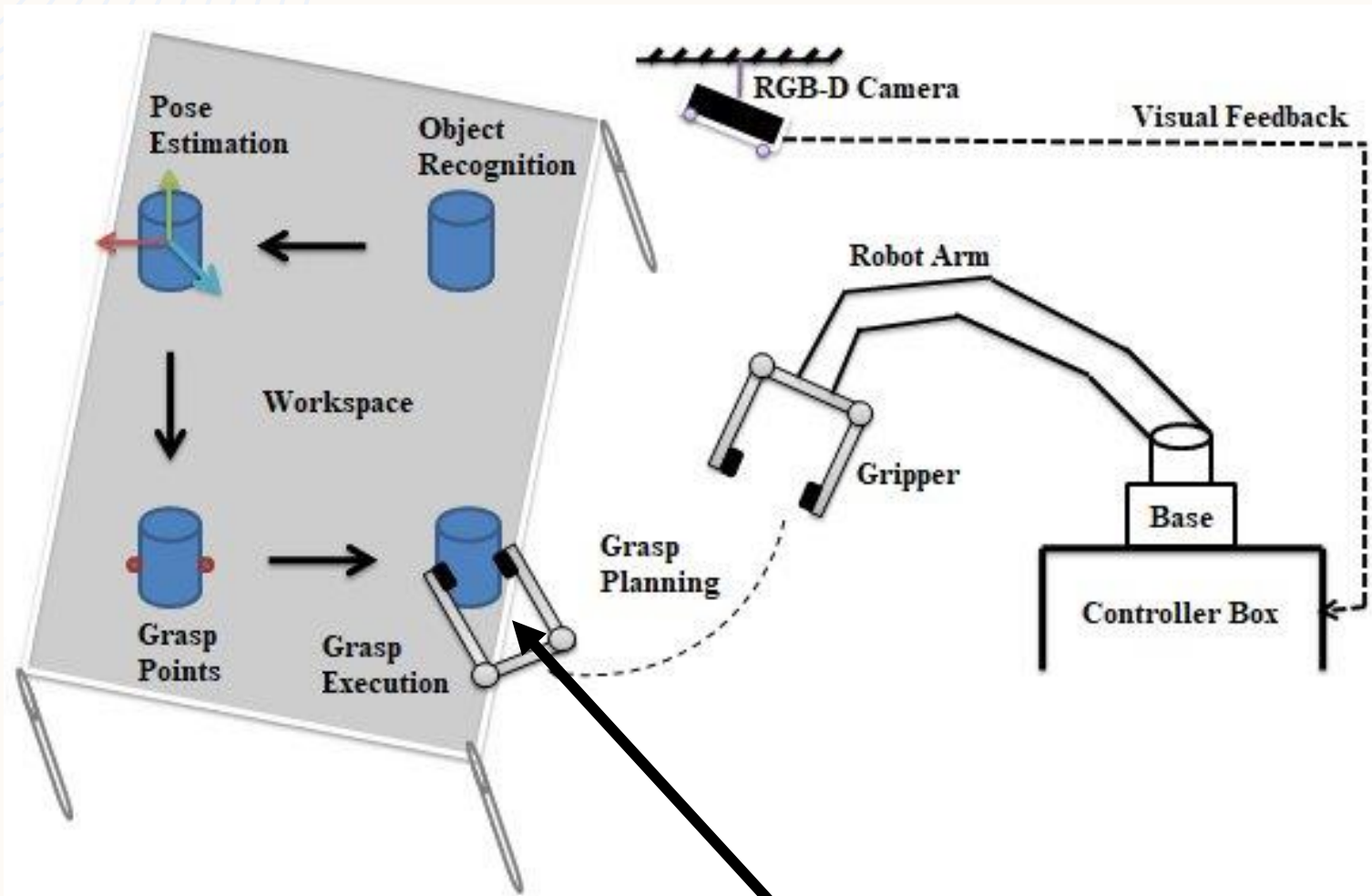


DOG, DOG, CAT

## Instance Segmentation



DOG, DOG, CAT



**If this object is complex**



# Object detection vs object segmentation

## Segmentation

- **High Detail:** Image segmentation helps robots assign each pixel to a class, thus identifying the precise shape and boundaries of objects. This high accuracy is useful for grasping and manipulating complex objects.
- **Practical Application:** This allows robots to grasp objects with complex and irregular shapes, such as car parts, precise tools, or food items.

## Object detection

- **Fast Identification:** Object detection allows robots to quickly identify objects and determine their approximate position in the environment.
- **Practical Application:** Useful for manipulation tasks that require fast object identification, such as picking items off a conveyor belt.

## Environmental Complexity

- **Image Segmentation:** Useful in complex environments requiring precise differentiation of objects from the background and detailed object identification.
- **Object Detection:** Useful in simpler environments or applications requiring quick object identification.

## Classification



CAT

No spatial extent

## Semantic Segmentation



GRASS, CAT,  
TREE, SKY

No objects, just pixels

## Object Detection



DOG, DOG, CAT

Multiple Object

## Instance Segmentation



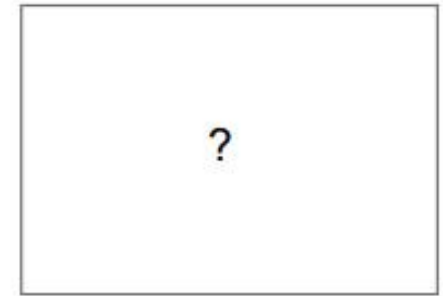
DOG, DOG, CAT



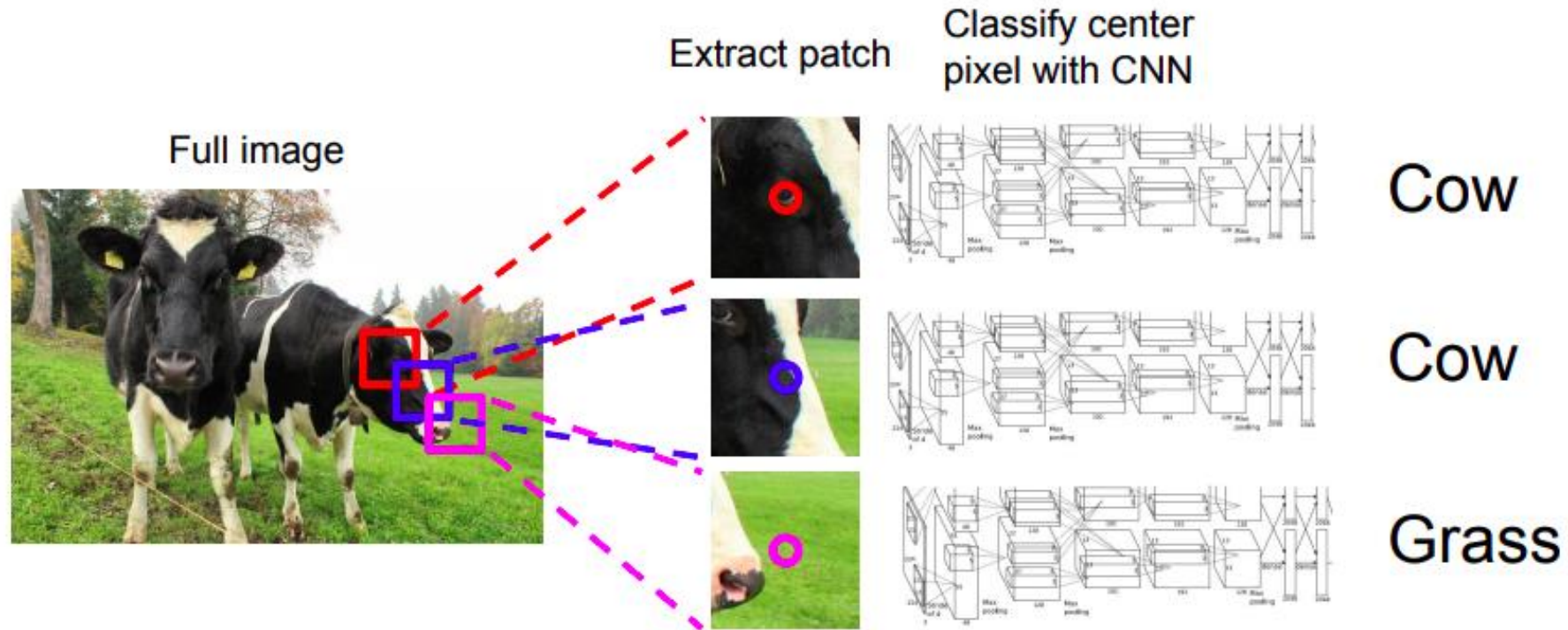


GRASS, CAT,  
TREE, SKY, ...

Paired training data: for each training image, each pixel is labeled with a semantic category.



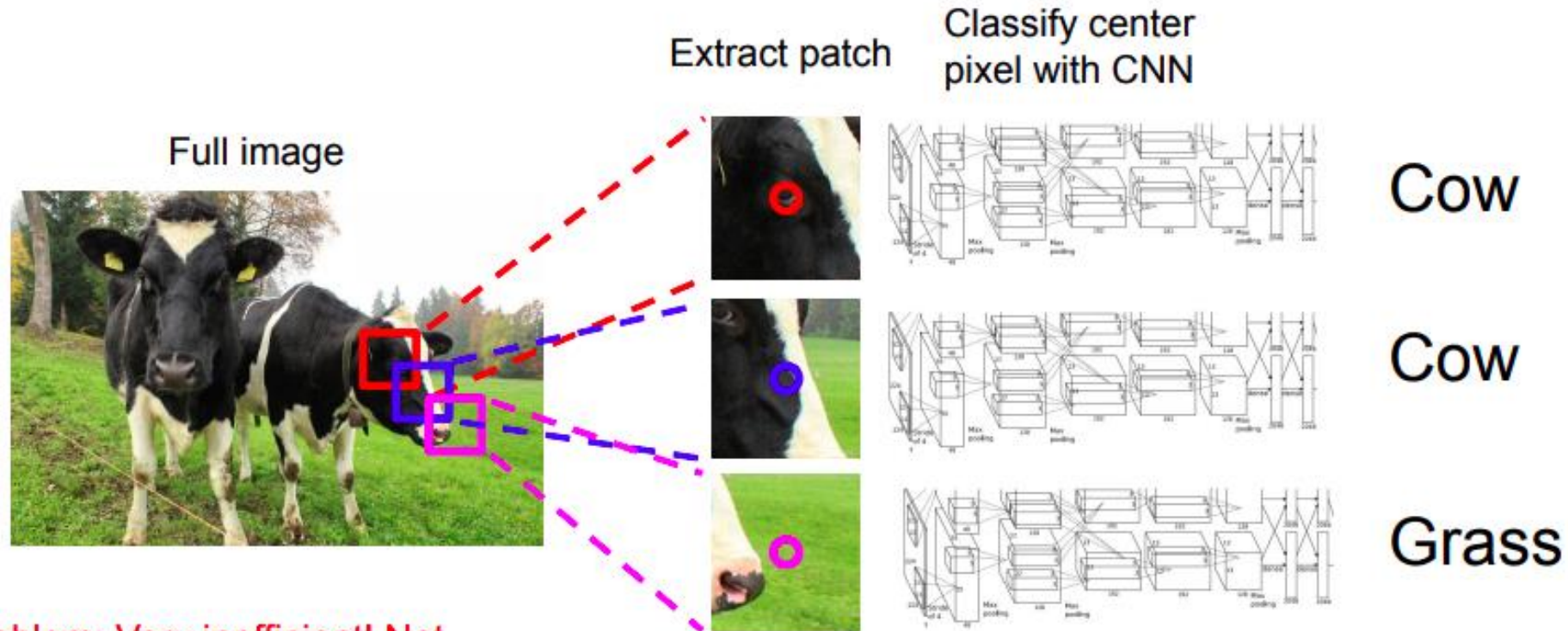
At test time, classify each pixel of a new image.



Farabet et al, "Learning Hierarchical Features for Scene Labeling," TPAMI 2013

Pinheiro and Collobert, "Recurrent Convolutional Neural Networks for Scene Labeling", ICML 2014

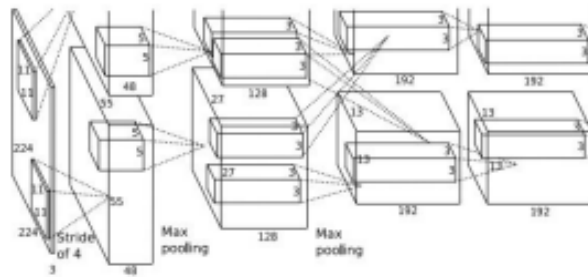




Problem: Very inefficient! Not reusing shared features between overlapping patches

Farabet et al, "Learning Hierarchical Features for Scene Labeling," TPAMI 2013  
 Pinheiro and Collobert, "Recurrent Convolutional Neural Networks for Scene Labeling", ICML 2014

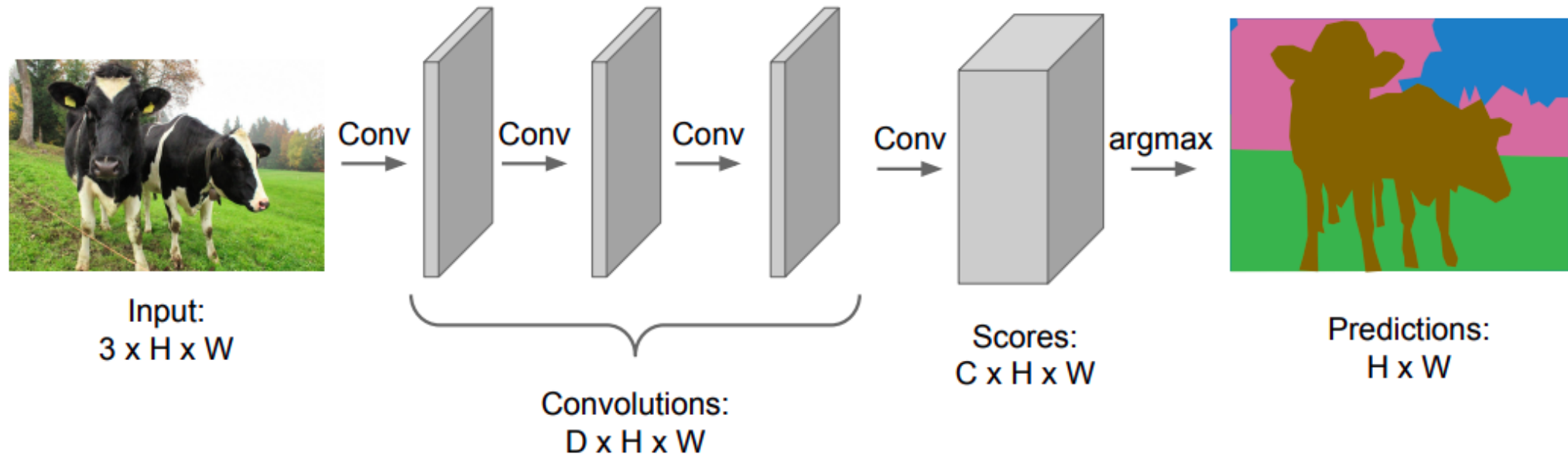
Full image



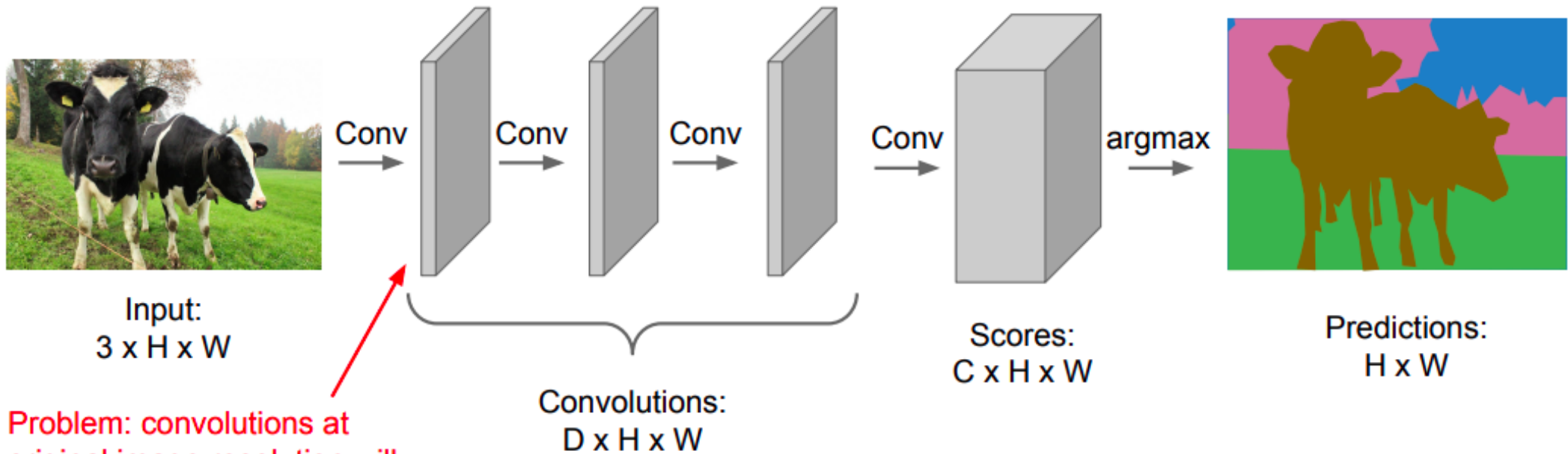
An intuitive idea: encode the entire image with conv net, and do semantic segmentation on top.

Problem: classification architectures often reduce feature spatial sizes to go deeper, but semantic segmentation requires the output size to be the same as input size.

Design a network with only convolutional layers without downsampling operators to make predictions for pixels all at once!

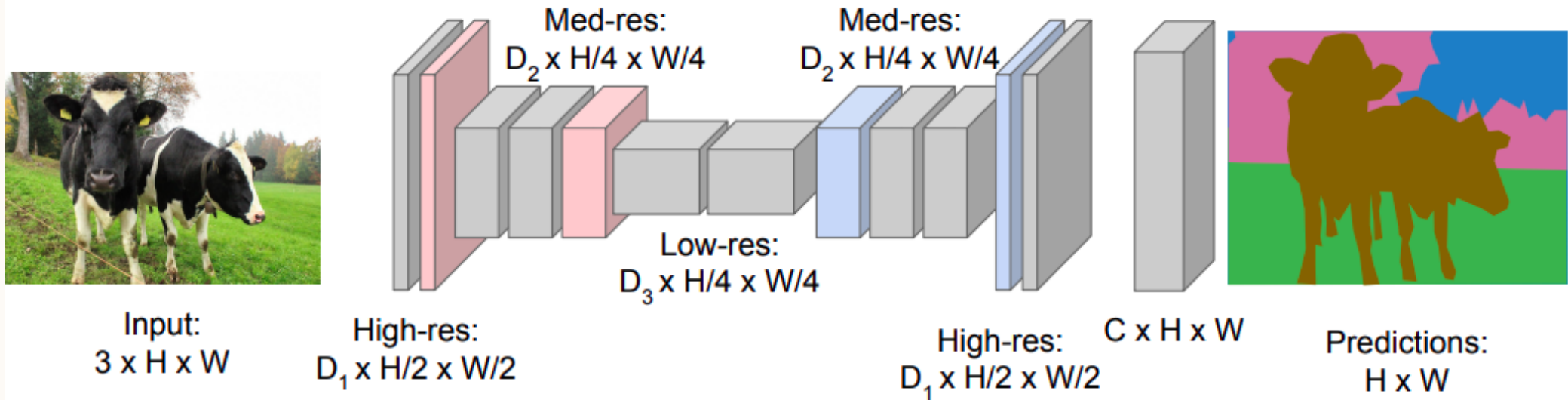


Design a network with only convolutional layers without downsampling operators to make predictions for pixels all at once!



Problem: convolutions at original image resolution will be very expensive ...

Design network as a bunch of convolutional layers, with **downsampling** and **upsampling** inside the network!



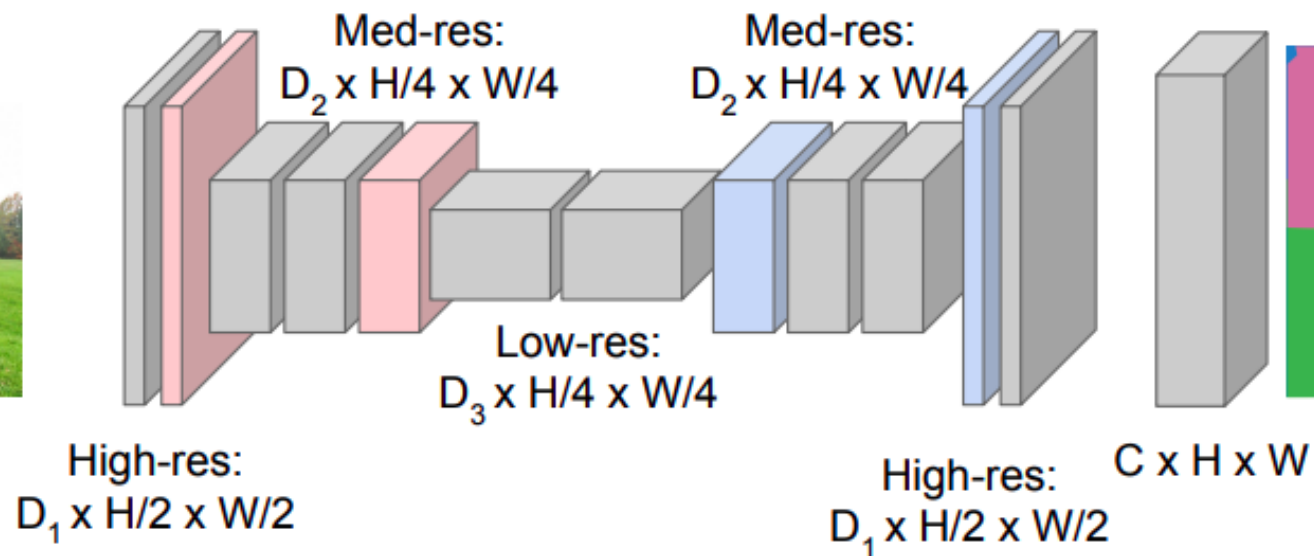


**Downsampling:**  
Pooling, strided  
convolution



Input:  
 $3 \times H \times W$

Design network as a bunch of convolutional layers, with **downsampling** and **upsampling** inside the network!



**Upsampling:**  
???



# pooling

**Nearest Neighbor**

1	2
3	4



1	1	2	2
1	1	2	2
3	3	4	4
3	3	4	4

Input: 2 x 2

Output: 4 x 4

**“Bed of Nails”**

1	2
3	4

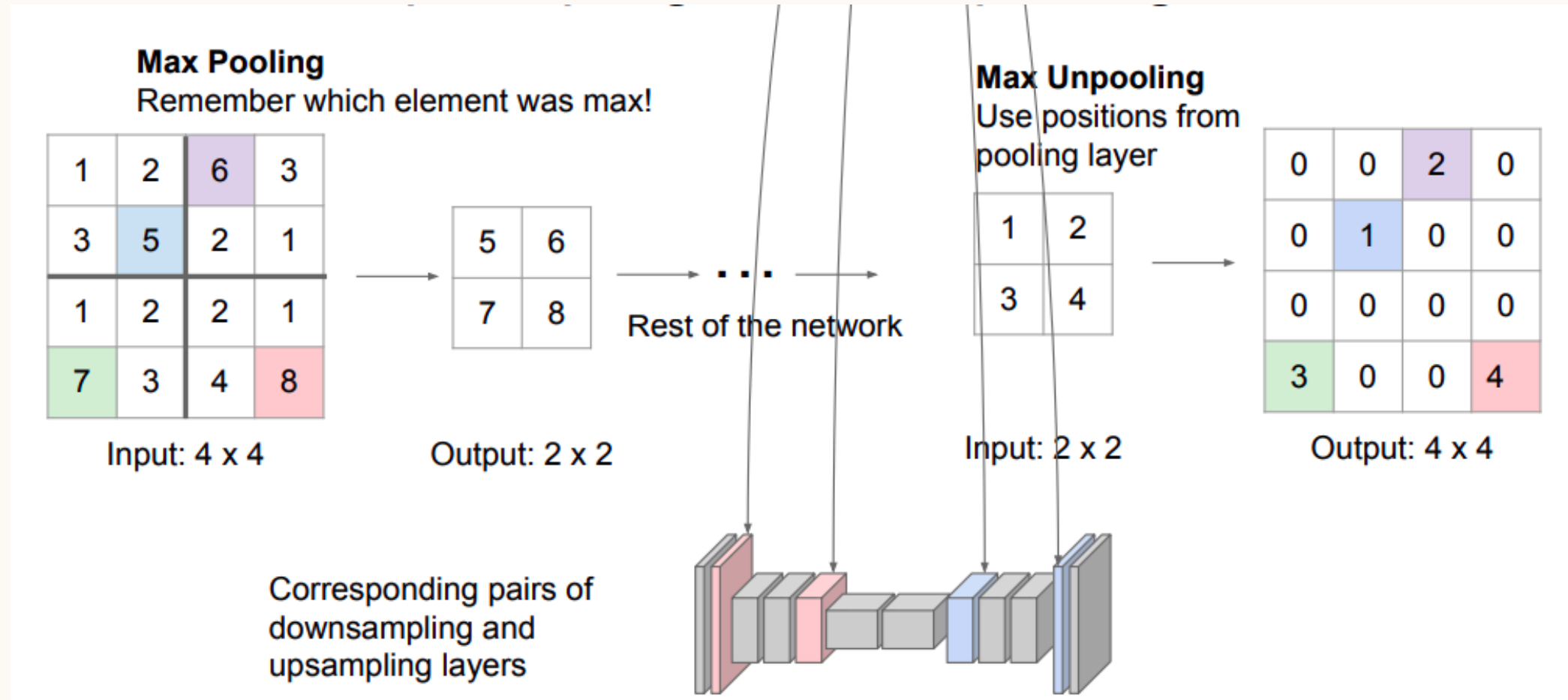


1	0	2	0
0	0	0	0
3	0	4	0
0	0	0	0

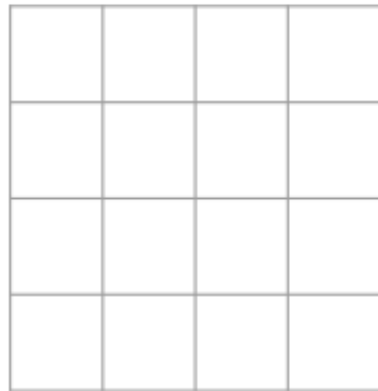
Input: 2 x 2

Output: 4 x 4

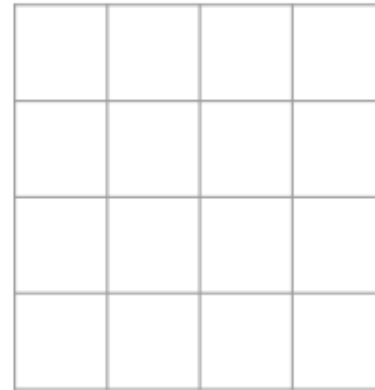
# Max - pooling



**Recall:** Normal 3 x 3 convolution, stride 1 pad 1

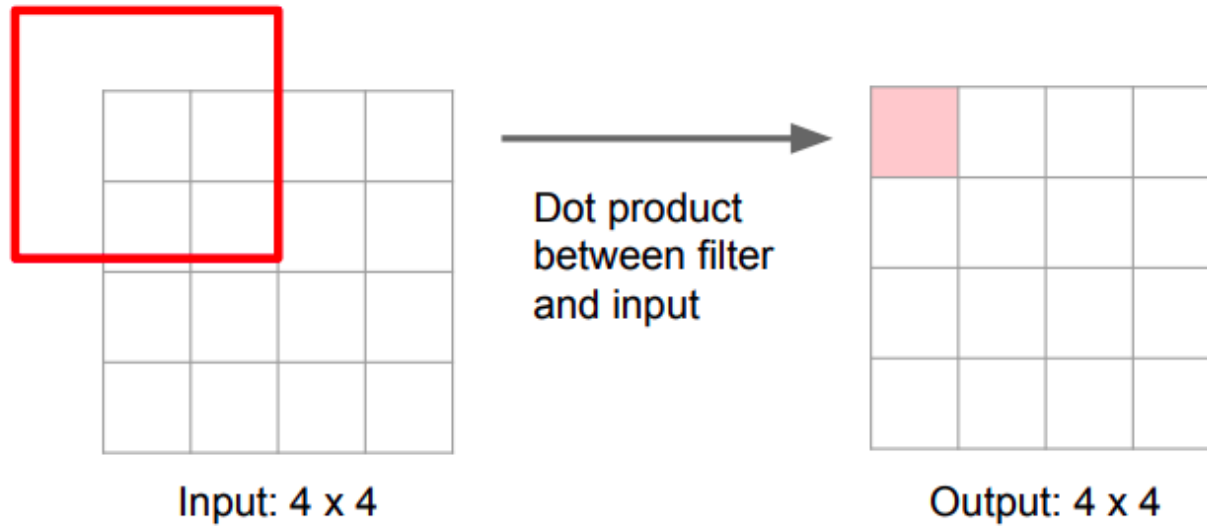


Input: 4 x 4



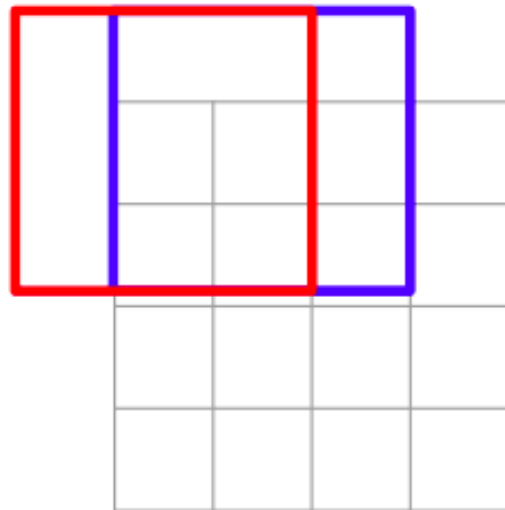
Output: 4 x 4

**Recall:** Normal 3 x 3 convolution, stride 1 pad 1





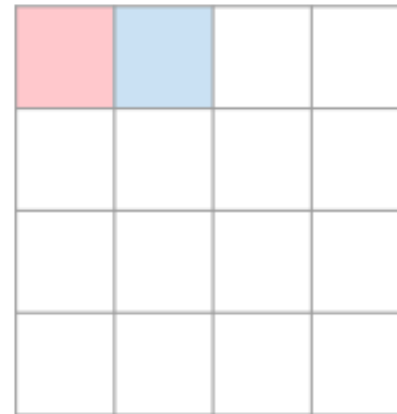
**Recall:** Normal 3 x 3 convolution, stride 1 pad 1



Input: 4 x 4

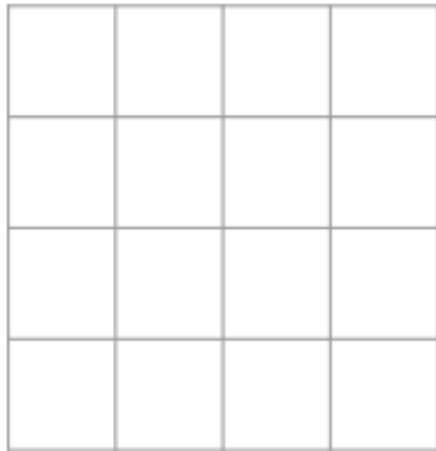


Dot product  
between filter  
and input

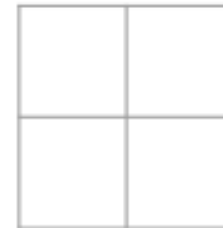


Output: 4 x 4

**Recall:** Normal 3 x 3 convolution, stride 2 pad 1

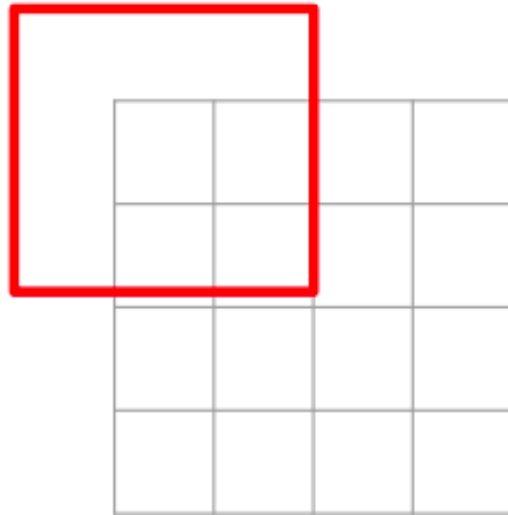


Input: 4 x 4



Output: 2 x 2

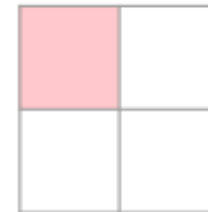
**Recall:** Normal 3 x 3 convolution, stride 2 pad 1



Input: 4 x 4

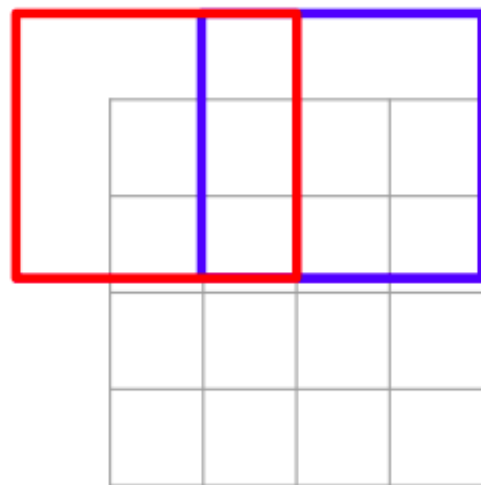


Dot product  
between filter  
and input



Output: 2 x 2

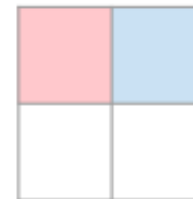
### Recall: Normal 3 x 3 convolution, stride 2 pad 1



Input: 4 x 4



Dot product  
between filter  
and input



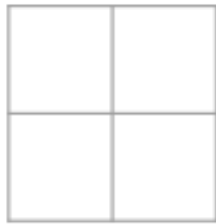
Output: 2 x 2

Filter moves 2 pixels in  
the input for every one  
pixel in the output

Stride gives ratio between  
movement in input and  
output

We can interpret strided  
convolution as “learnable  
downsampling”.

**3 x 3 transposed** convolution, stride 2 pad 1

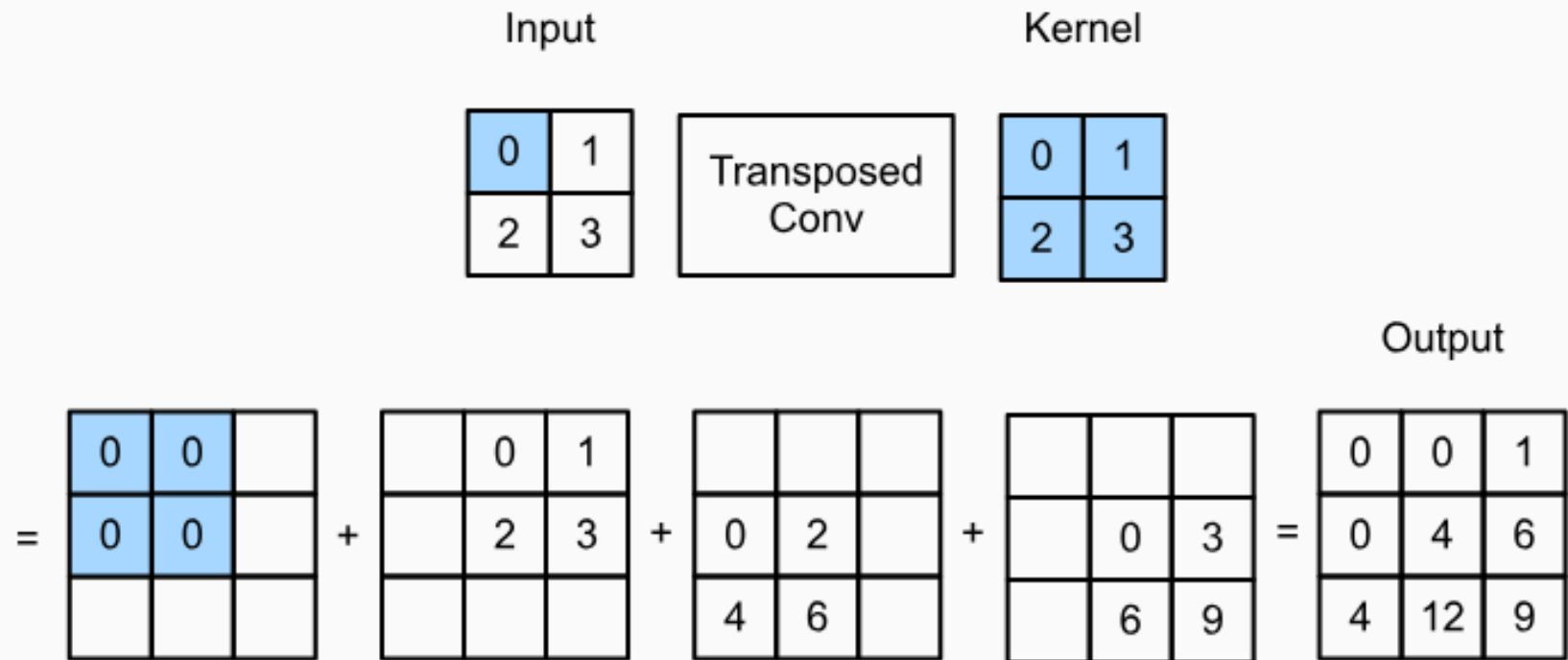


Input: 2 x 2



Output: 4 x 4



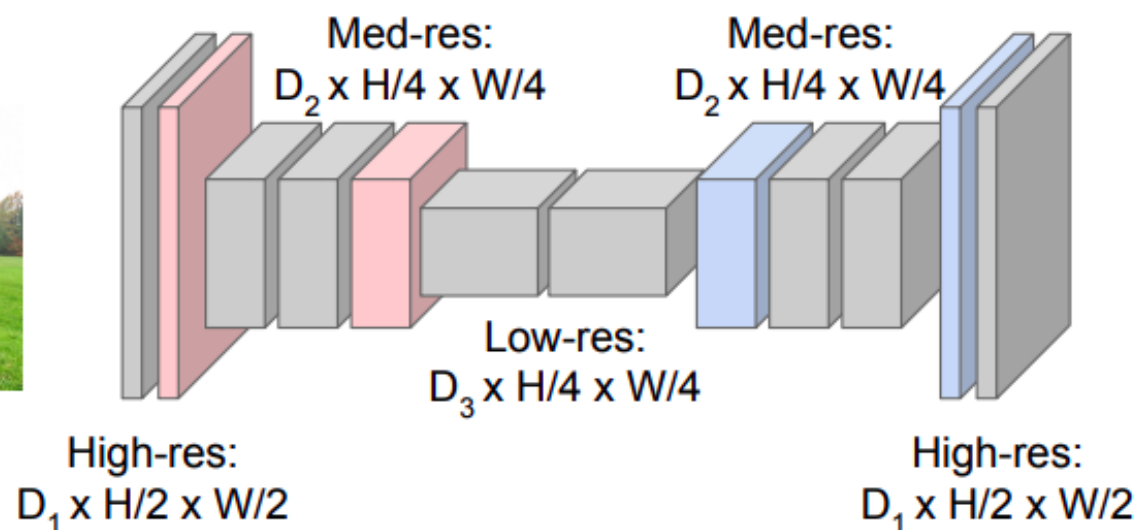


**Downsampling:**  
Pooling, strided  
convolution



Input:  
 $3 \times H \times W$

Design network as a bunch of convolutional layers, with **downsampling** and **upsampling** inside the network!

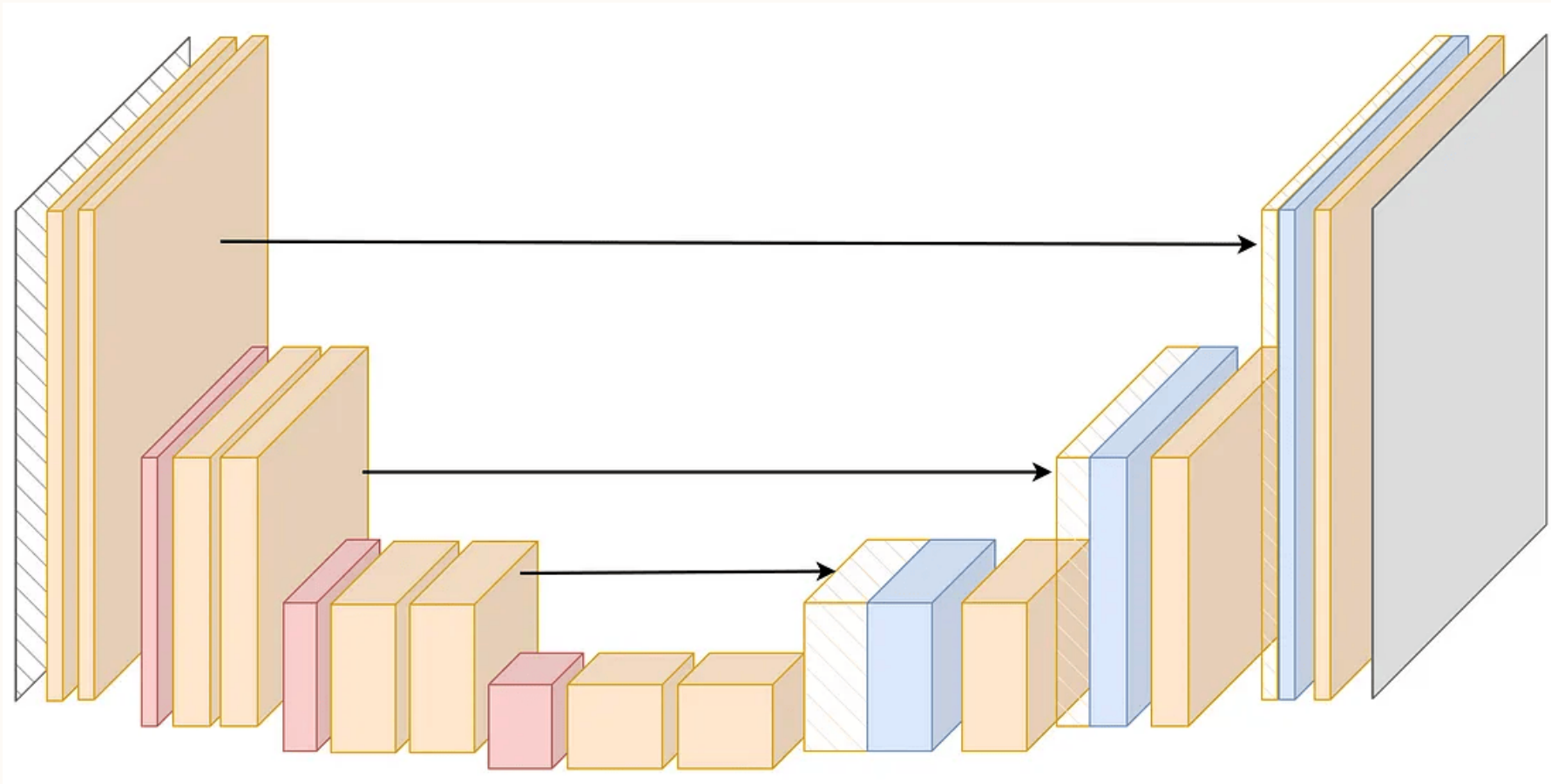


**Upsampling:**  
Unpooling or strided  
transposed convolution



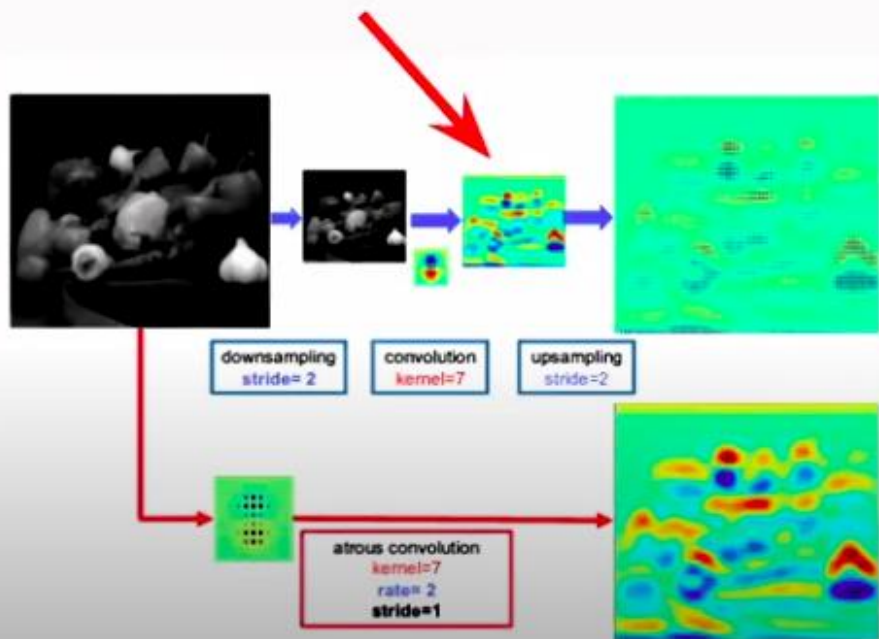
Predictions:  
 $H \times W$

# UNET



# DEEPLAB

## 1) Problem: Reduced feature resolution



Causes:

- Maxpooling

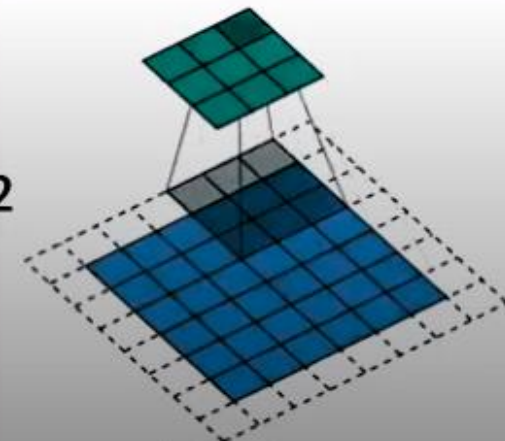
12	20	30	0
8	12	2	0
34	70	37	4
112	100	25	12

$\xrightarrow{2 \times 2 \text{ Max-Pool}}$

20	30
112	37

Source: computersciencewiki.org

- Conv, strides = 2



# DEEPLAB

## 2) Problem: Existence of multiple-scale objects

Same class

Different size (FoV)



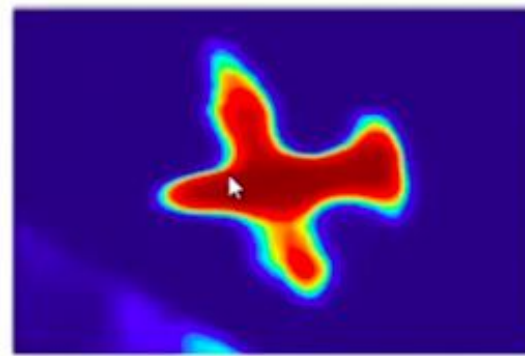


# DEEPLAB

3) Reduced accuracy (in borders)



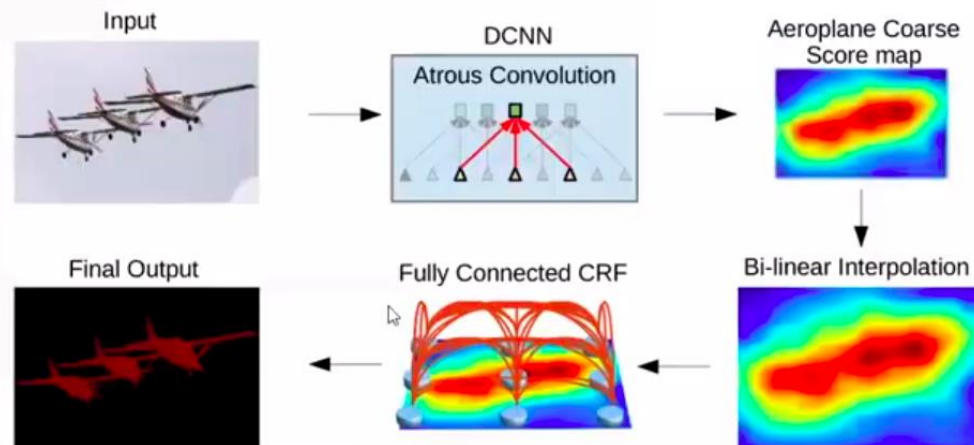
Image/G.T.



DCNN output

# DEEPLAB

- 1) Atrous convolution (or dilated convolution)
- 2) Atrous Spatial Pyramid Pooling (ASPP)
- 3) Conditional Random Fields (CRFs)



## Pipeline

Backbone  
ASPP  
Upsample x8  
CRF

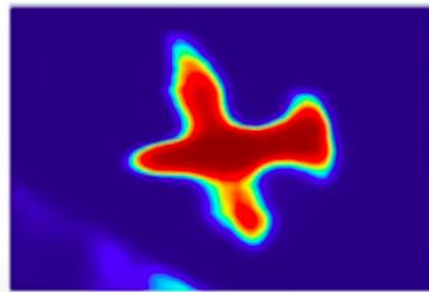
# DEEPLAB

3) Reduced accuracy (in borders)

Cause: Downsampling, maxpooling, useful to achieve invariance in **classification**

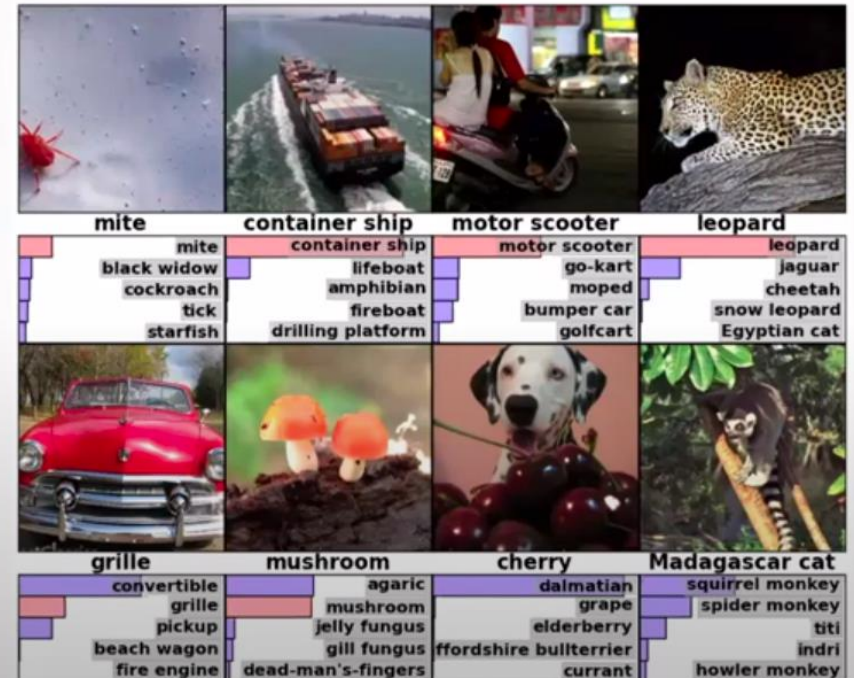


Image/G.T.



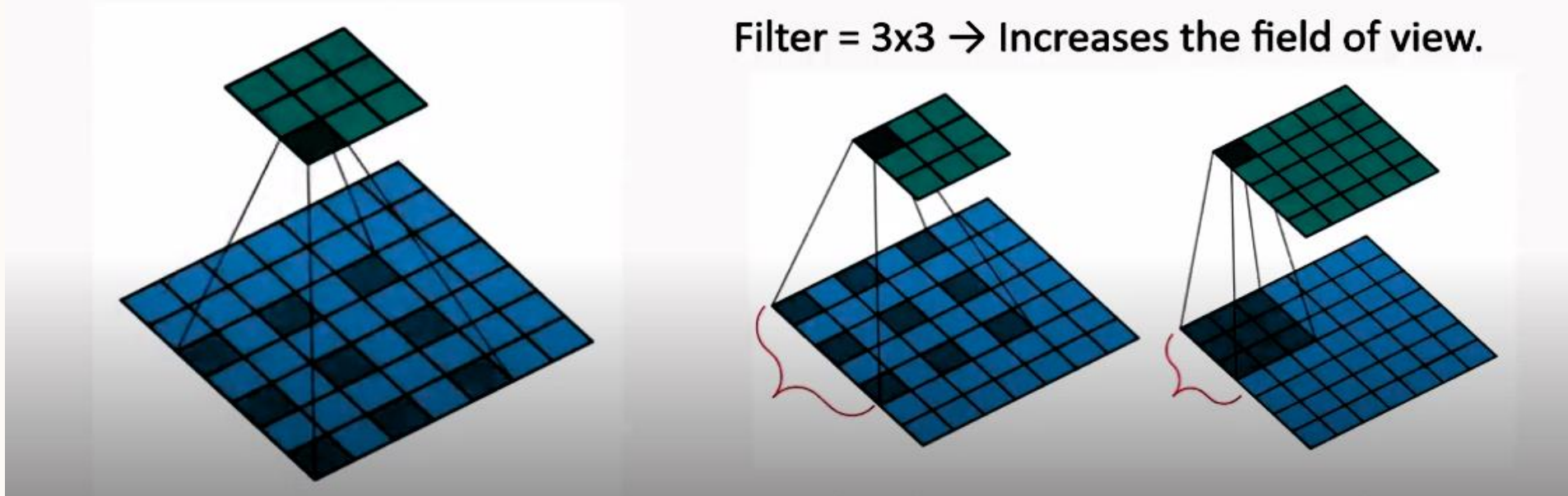
DCNN output

In Segmentation we want to preserve **spatial information**



# DEEPLAB

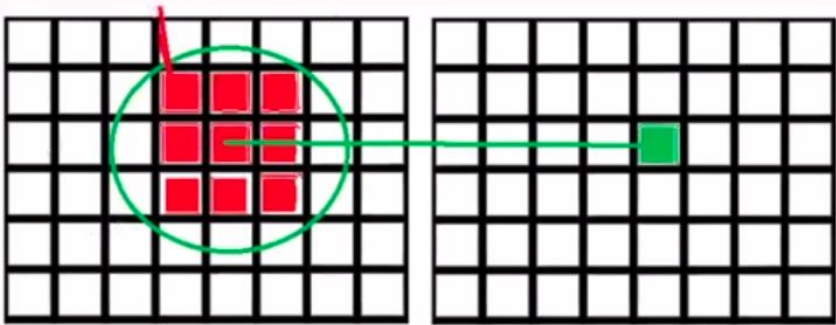
## 1) Atrous convolution (or dilated convolution)



# DEEPLAB

(Field of view, reminder)

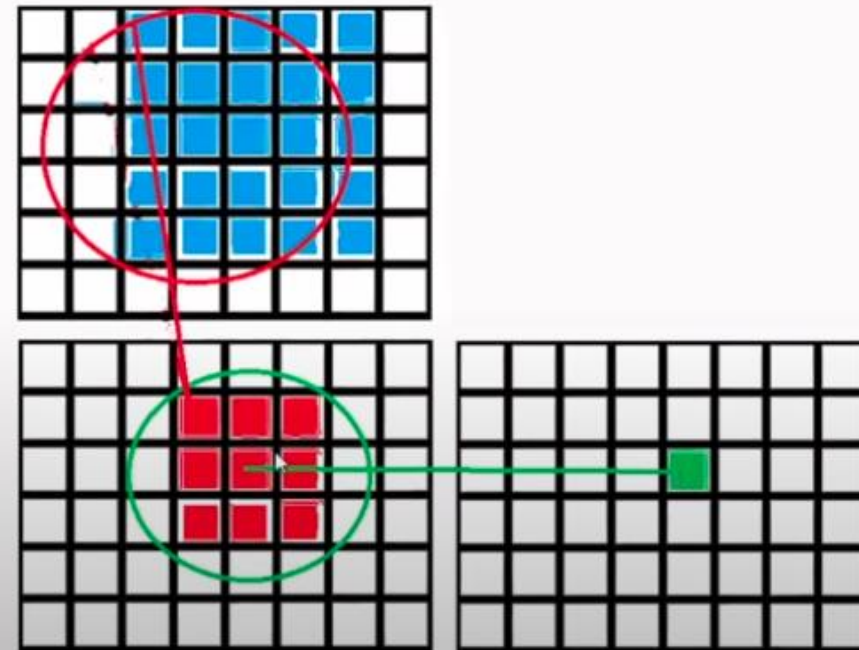
Operation	Field of view (size)
Conv 3x3	3



# DEEPLAB

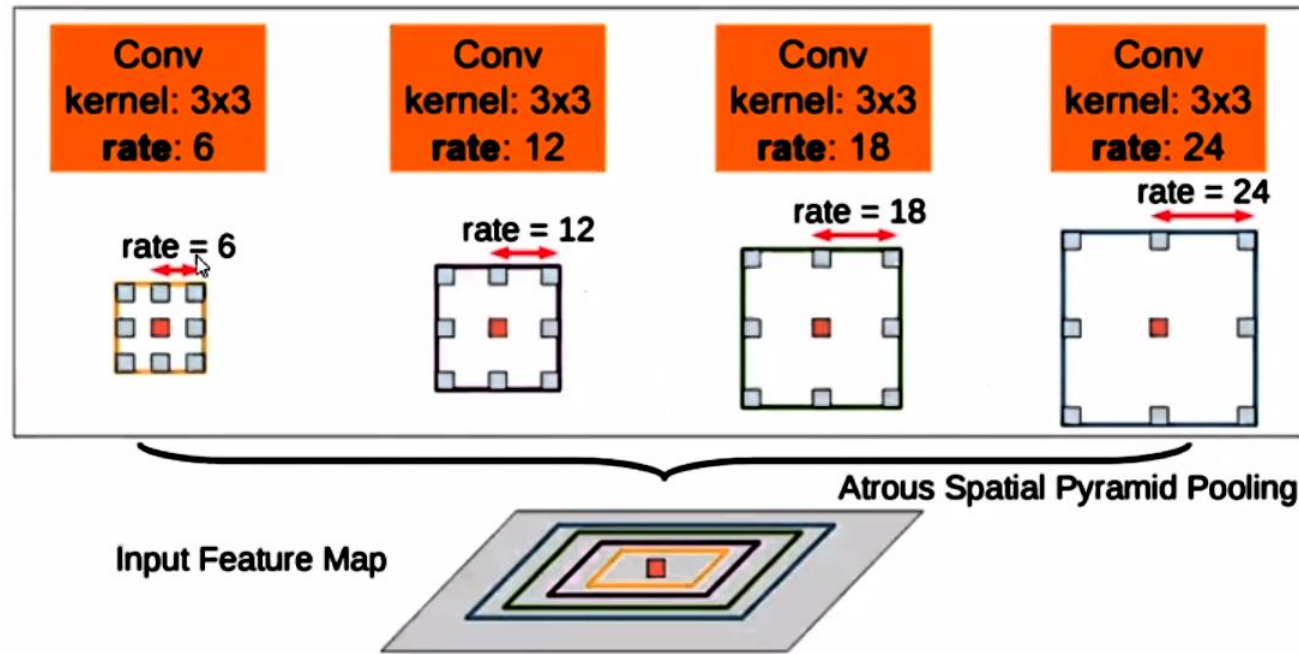
(Field of view, reminder)

Operation	Field of view (size)
Conv 3x3	3
Conv 3x3	$3 + 2 = 5$
...	...



# DEEPLAB

## 2) Atrous Spatial Pyramid Pooling (ASPP)





# DEEPLAB

## 3) Conditional Random Fields (CRFs)

$$E(x) = \sum_i \theta_i(x_i) + \sum_{ij} \theta_{ij}(x_i, x_j) \quad (2)$$



$$\theta_{ij}(x_i, x_j) = \mu(x_i, x_j) \left[ w_1 \exp \left( -\frac{\|p_i - p_j\|^2}{2\sigma_\alpha^2} - \frac{\|I_i - I_j\|^2}{2\sigma_\beta^2} \right) + w_2 \exp \left( -\frac{\|p_i - p_j\|^2}{2\sigma_\gamma^2} \right) \right] \quad (3)$$

$$\mu(x_i, x_j) = 1 \text{ if } x_i \neq x_j,$$

$p$  = coordinates

$I$  = RGB intensity values



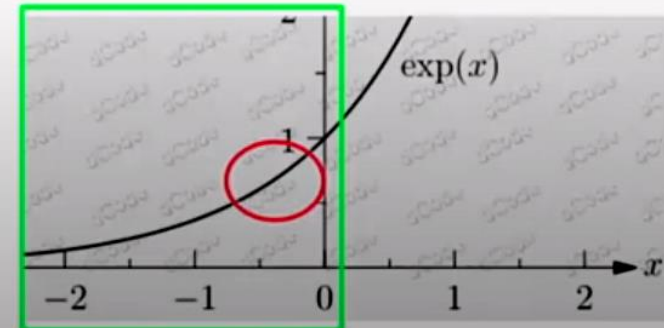
# DEEPLAB

## 3) Conditional Random Fields (CRFs)

$$\theta_{ij}(x_i, x_j) = \mu(x_i, x_j) \left[ w_1 \exp \left( -\frac{\|p_i - p_j\|^2}{2\sigma_\alpha^2} - \frac{\|I_i - I_j\|^2}{2\sigma_\beta^2} \right) + w_2 \exp \left( -\frac{\|p_i - p_j\|^2}{2\sigma_\gamma^2} \right) \right] \quad (3)$$

- Small distance, similar intensities

Small negative values  $\rightarrow$  large penalty



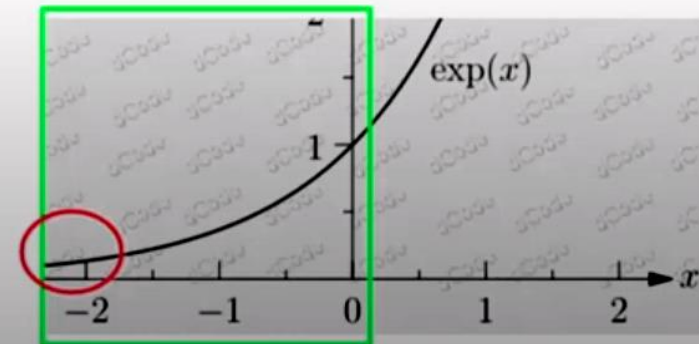
# DEEPLAB

## 3) Conditional Random Fields (CRFs)

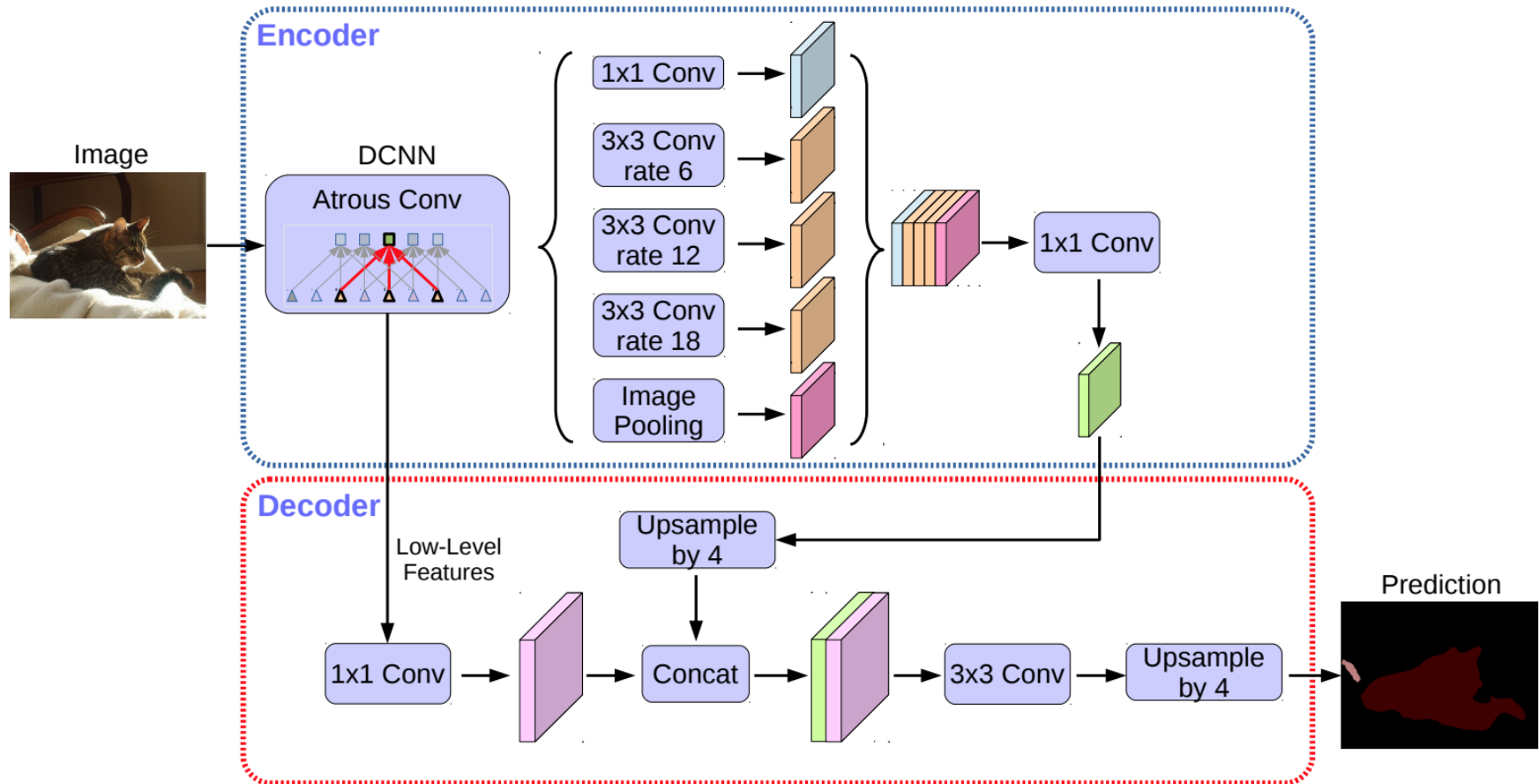
$$\theta_{ij}(x_i, x_j) = \mu(x_i, x_j) \left[ w_1 \exp \left( -\frac{\|p_i - p_j\|^2}{2\sigma_\alpha^2} - \frac{\|I_i - I_j\|^2}{2\sigma_\beta^2} \right) + w_2 \exp \left( -\frac{\|p_i - p_j\|^2}{2\sigma_\gamma^2} \right) \right] \quad (3)$$

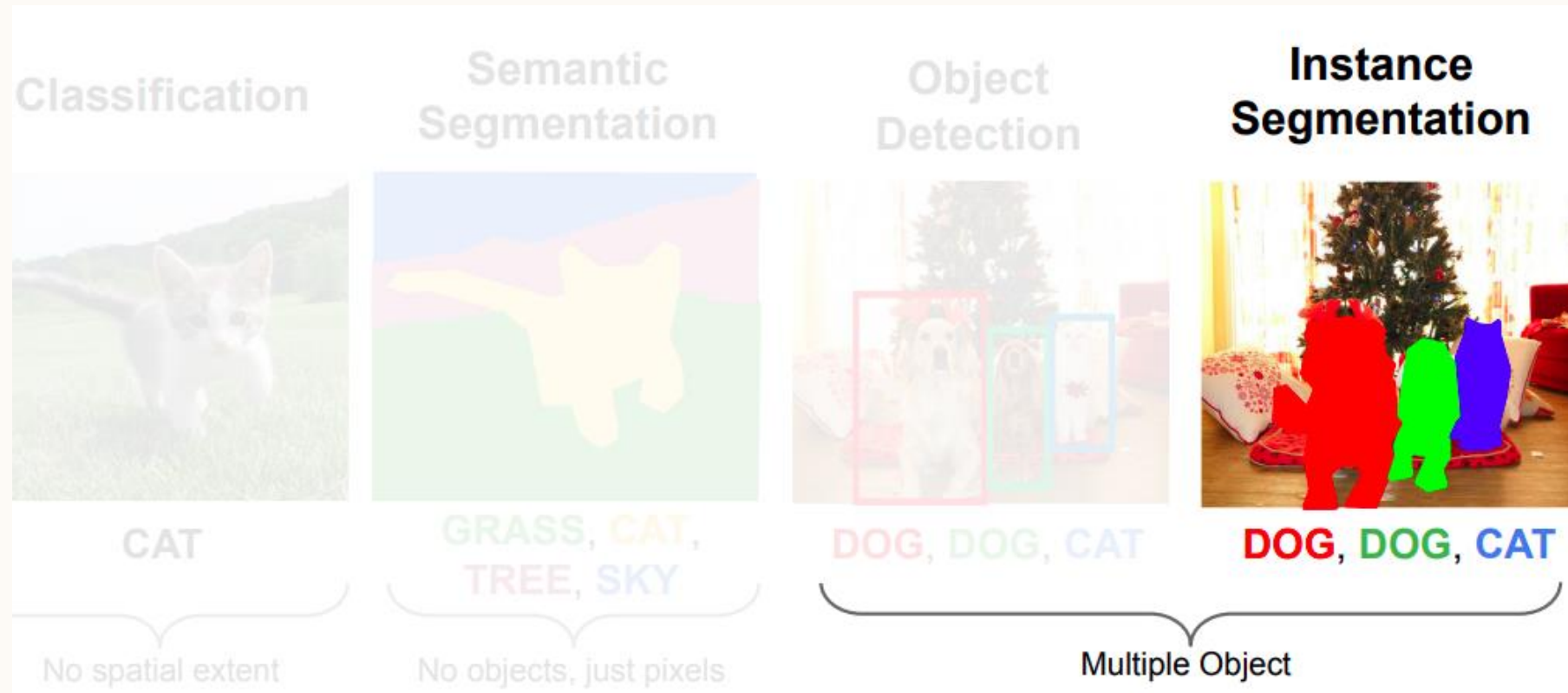
- Large distance, different intensities

Large negative values  $\rightarrow$  very small penalty

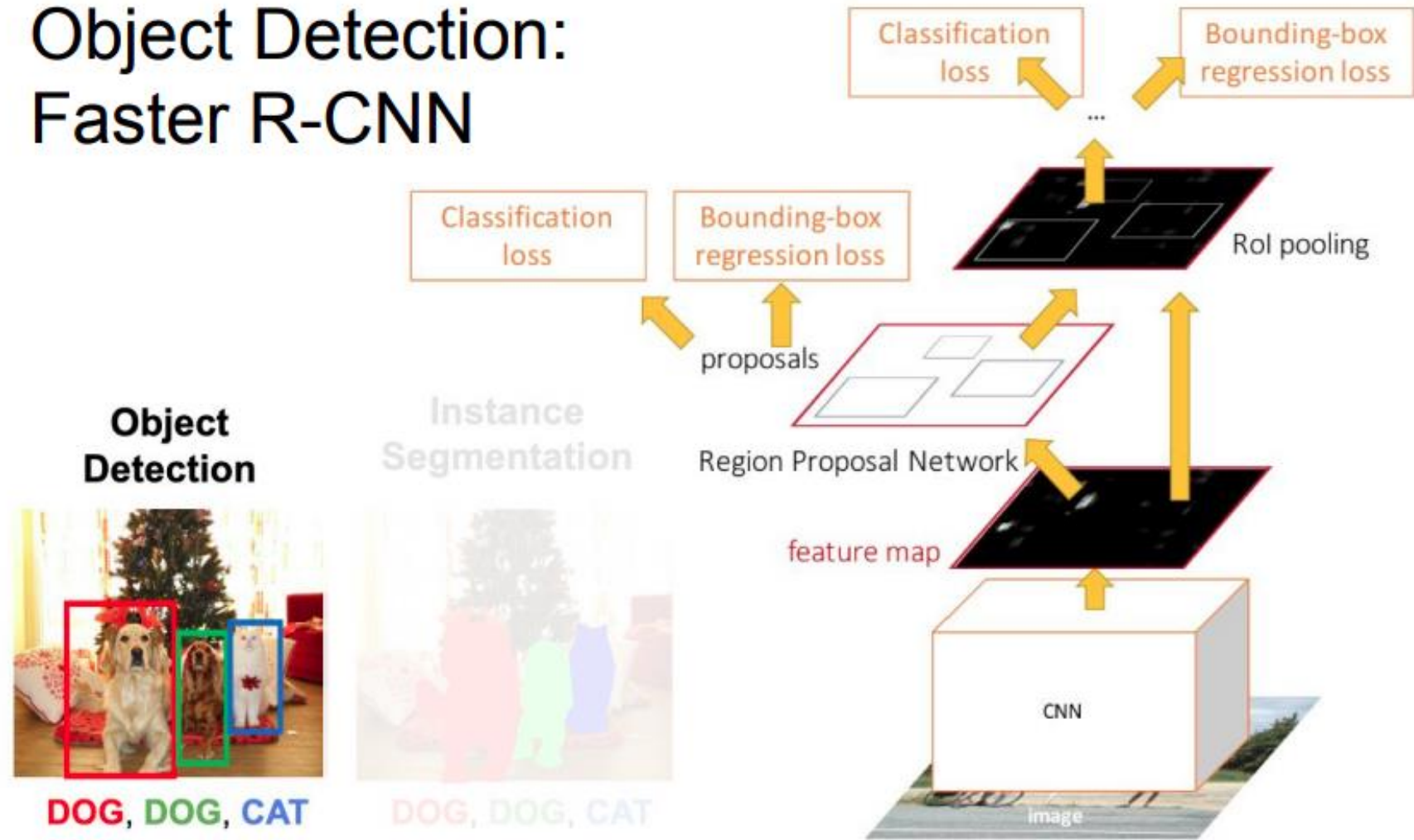


# DEEPLAB-V3

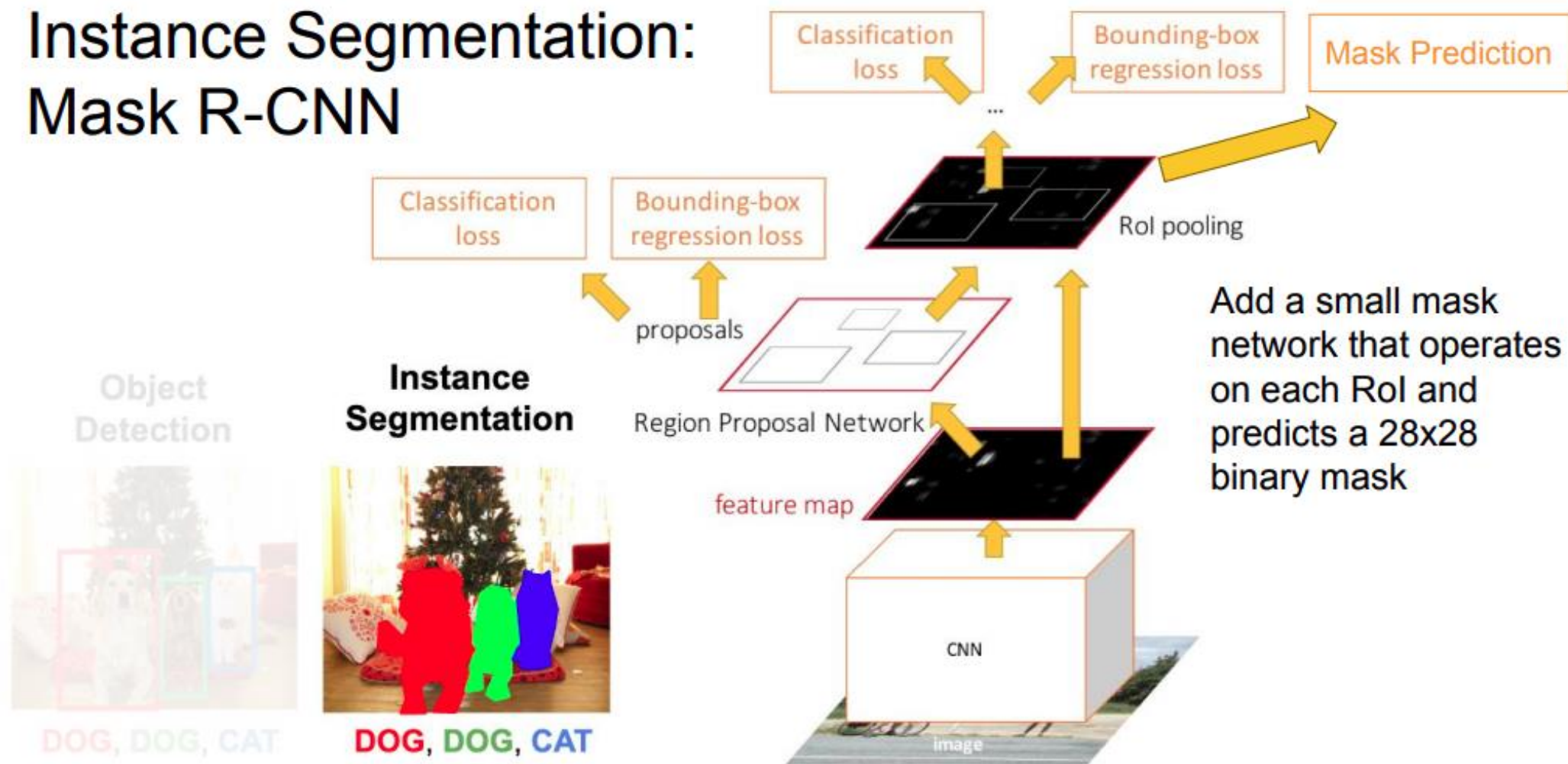




# Object Detection: Faster R-CNN

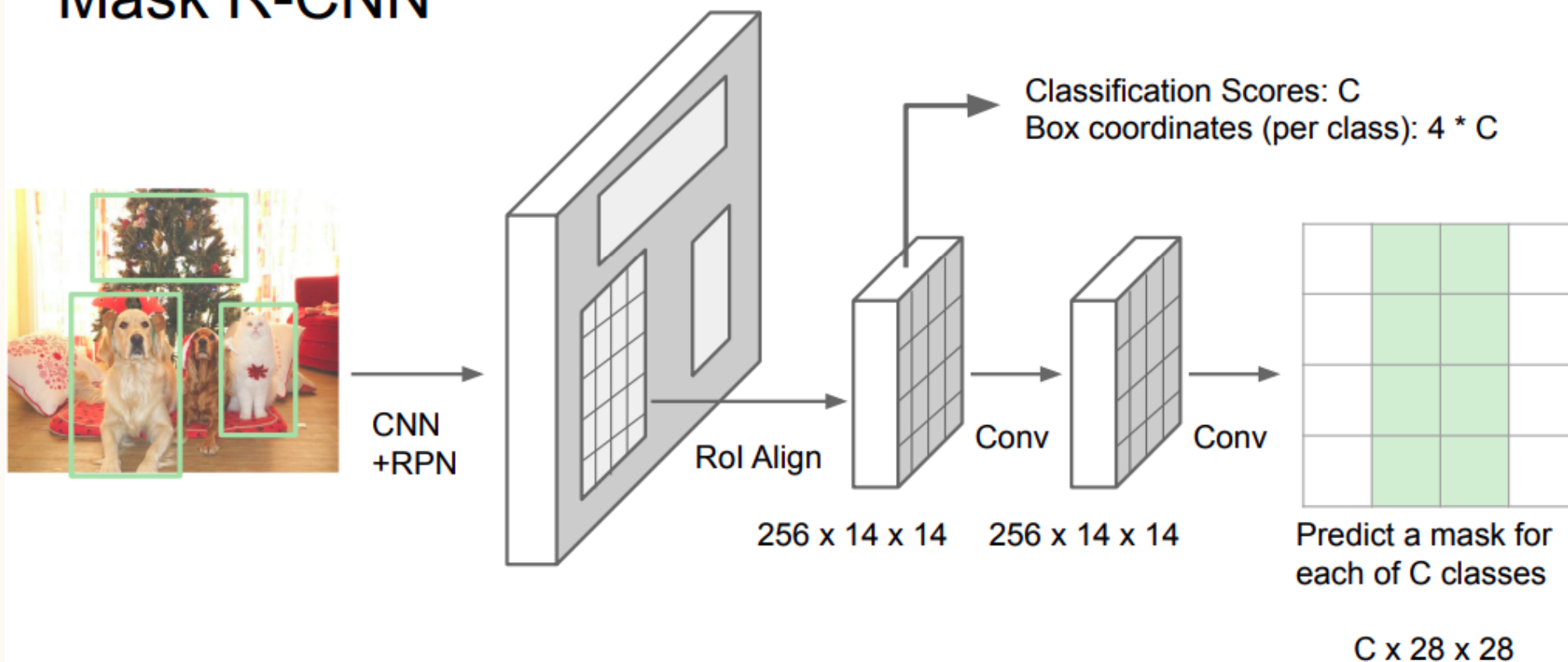


# Instance Segmentation: Mask R-CNN

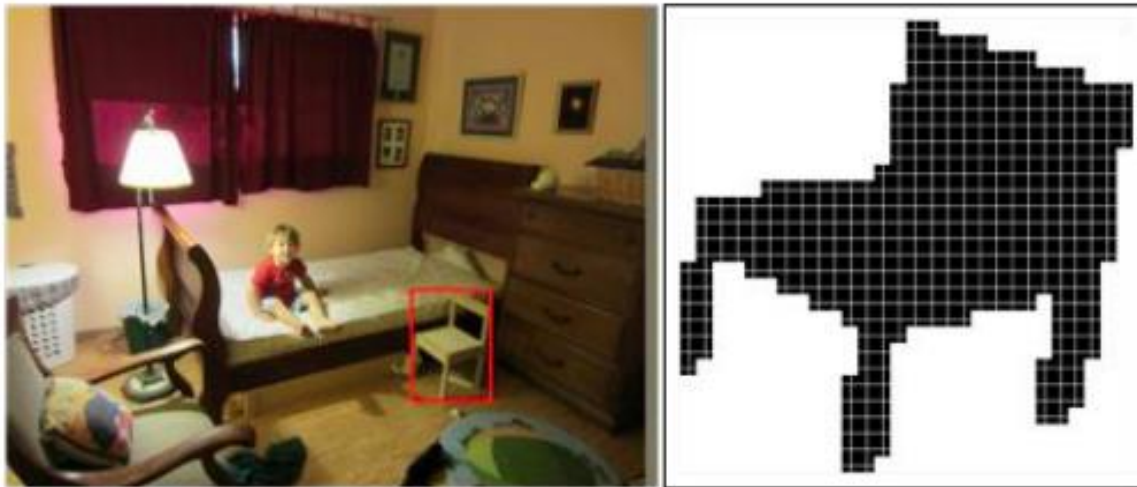




# Mask R-CNN

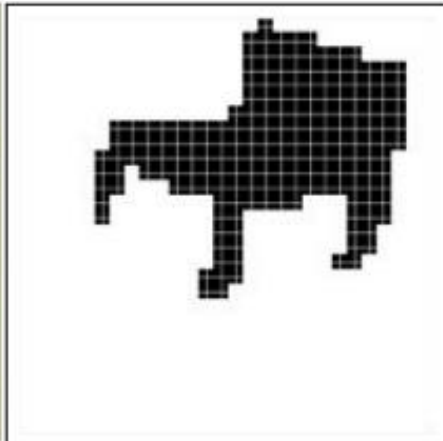
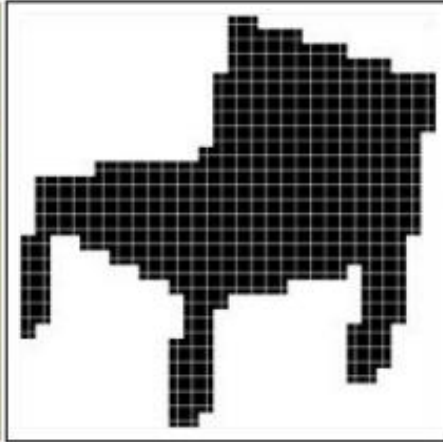


## Mask R-CNN: Example Mask Training Targets

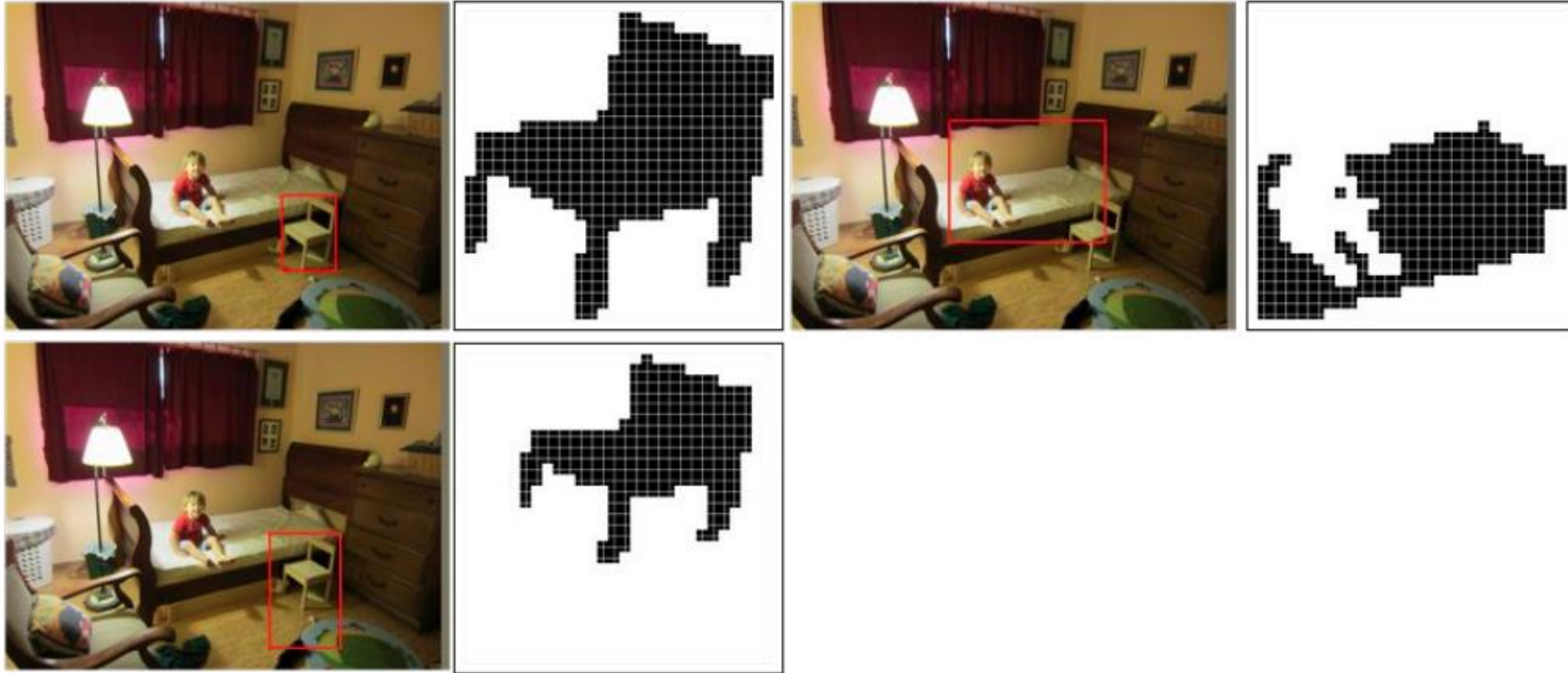




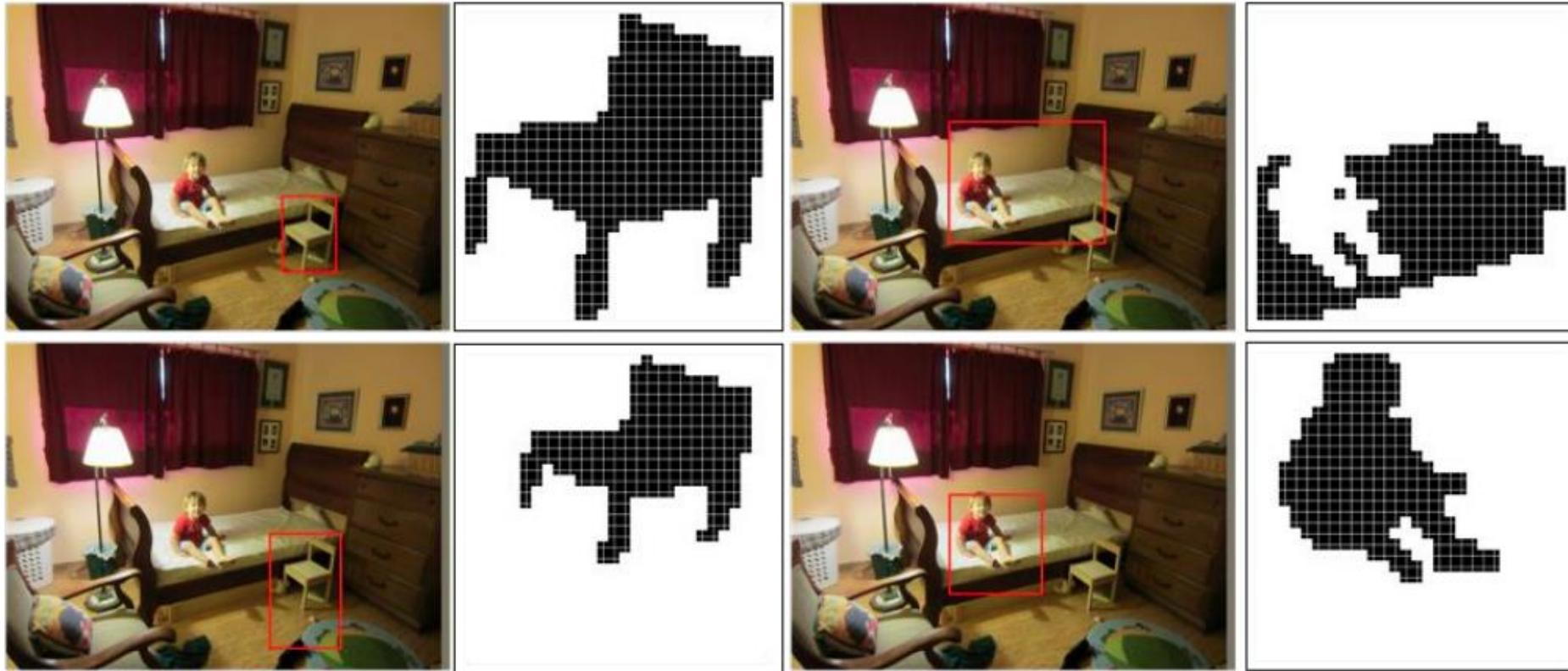
## Mask R-CNN: Example Mask Training Targets



## Mask R-CNN: Example Mask Training Targets



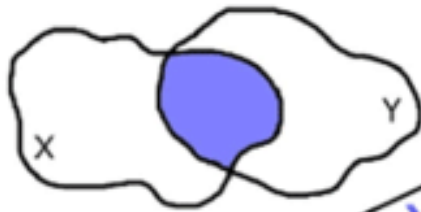
## Mask R-CNN: Example Mask Training Targets



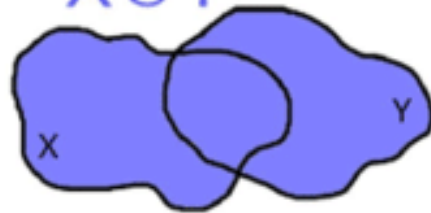
# Evaluation metrics in segmentation

$$\text{IoU} = \frac{\text{Intersection } X \cap Y}{\text{Union } X \cup Y}$$

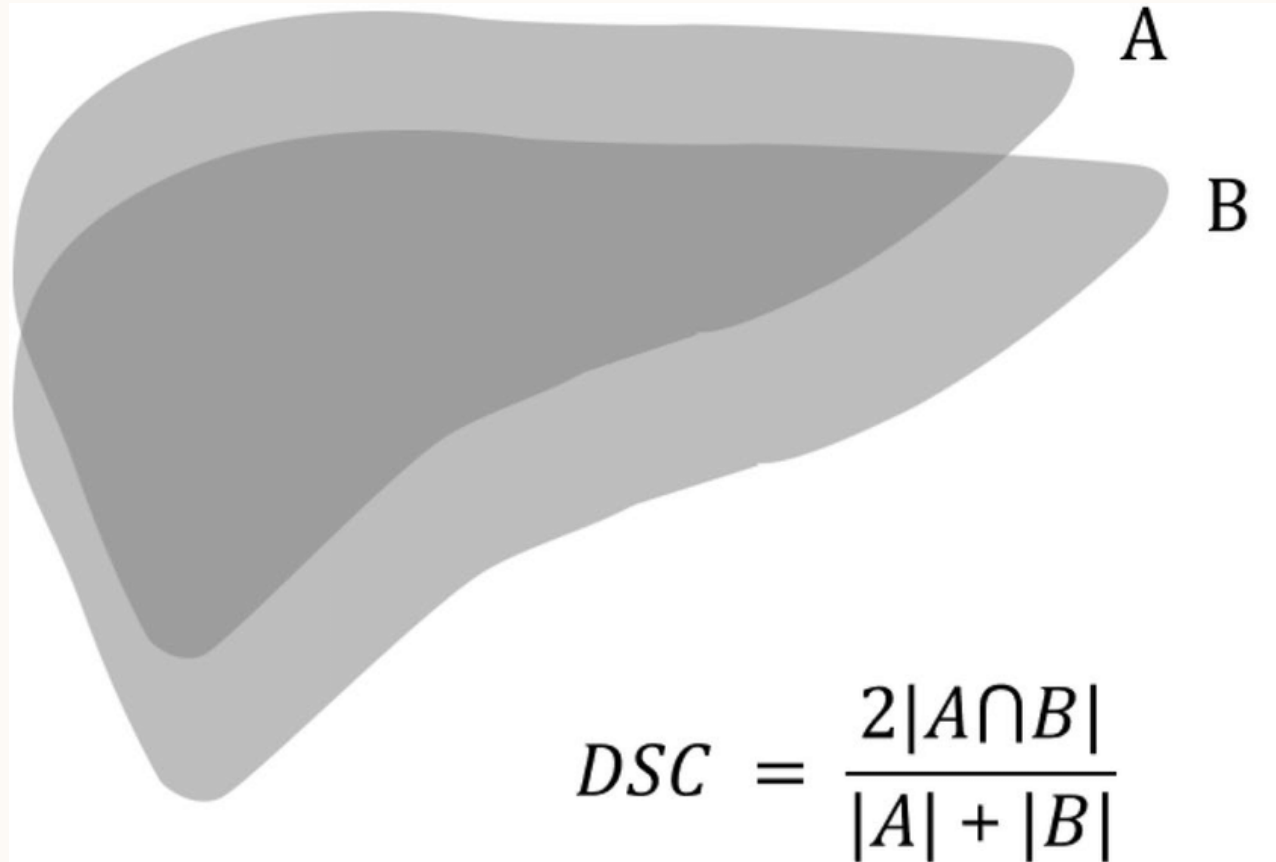
$X \cap Y$



$X \cup Y$



# Evaluation metrics in segmentation



DSC: Dice similarity coefficient

# Evaluation metrics in segmentation


$$\text{Dice Coefficient} = \frac{2 * TP}{FN + (2 * TP) + FP}$$

$$\text{Jaccard Index} = \frac{TP}{TP + FN + FP}$$

$$\text{Sensitivity} = \frac{TP}{TP + FN}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

TP - true positive  
TN - true negative  
FP - false positive  
FN - false negative

Manual Segmentation   
Automated Segmentation 