Computer vision and machine learning for the material scientist

Lecture 8.
Semantic Segmentation

Romain Vo

MINES PARIS | PSL

*slides adapted from CS231n

Computer Vision Tasks

Classification

DOG

$\underbrace{\hspace{6cm}}_{\text{\underline{Classify} the image}}$

Image is free here

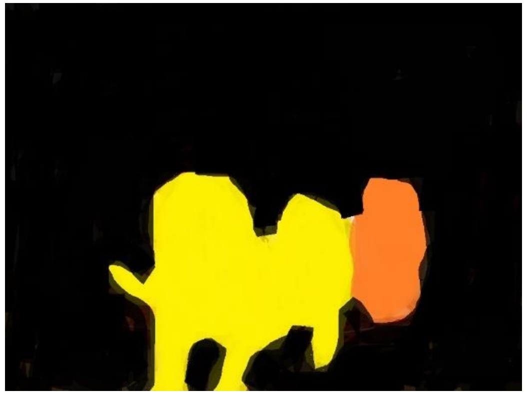
Computer Vision Tasks

Classification

DOG

Classify the image

Semantic segmentation



DOG,CAT,BG

Classify each pixel

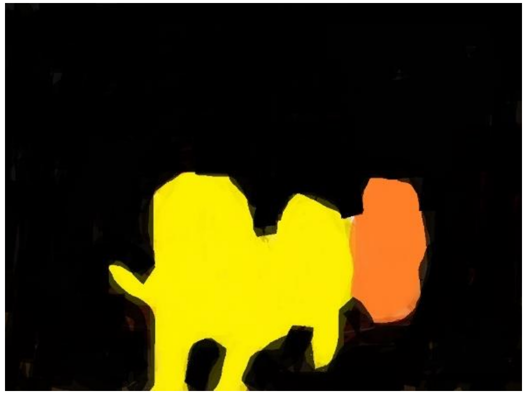
Computer Vision Tasks

Classification

DOG

Classify the image

Semantic segmentation



DOG,CAT,BG

Classify each pixel

Instance Segmentation

SMTH, SMTH, SMTH

Segment independent instances

Panoptic segmentation

DOG,DOG,CAT

Segment & Classify independent instances

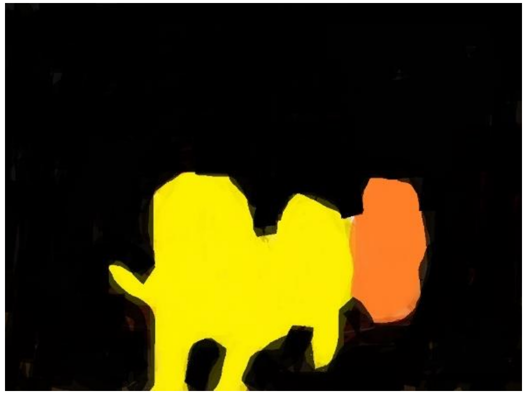
Computer Vision Tasks

Classification

DOG

Classify the image

Semantic segmentation



DOG,CAT,BG

Classify each pixel

Instance Segmentation

SMTH, SMTH, SMTH

Segment independent instances

Panoptic segmentation

DOG,DOG,CAT

Segment & Classify independent instances

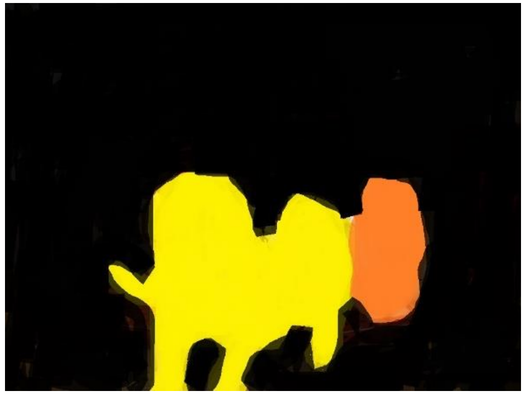
MINES PARIS | PSL

Semantic segmentation

Training data = pairs of (image, mask)

image

mask



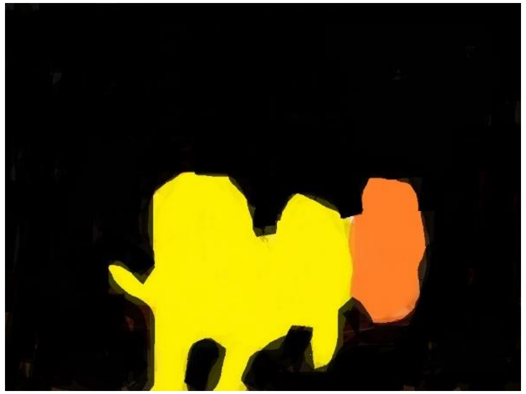
DOG,**CAT**,**BG**

Semantic segmentation

Training data = pairs of (image, mask)

image

mask



DOG,CAT,BG

For each training image, each pixel in the image is assigned a label:

- For example here - BG = 0, DOG = 1, CAT = 2

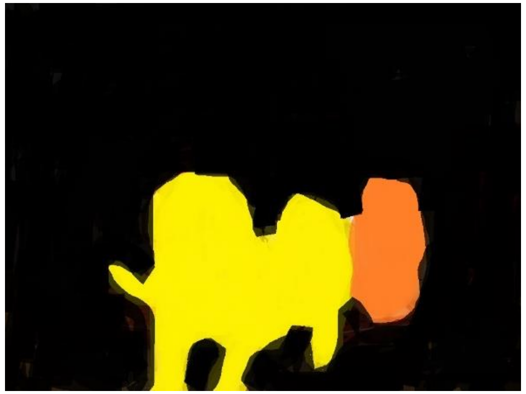
MINES PARIS | PSL

Semantic segmentation

Training data = pairs of (image, mask)

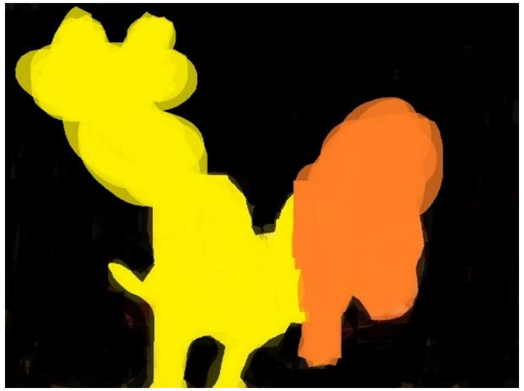
image

mask



DOG,**CAT**,**BG**

prediction



For each training image, each pixel in the image is assigned a label:

- For example here - BG = 0, DOG = 1, CAT = 2

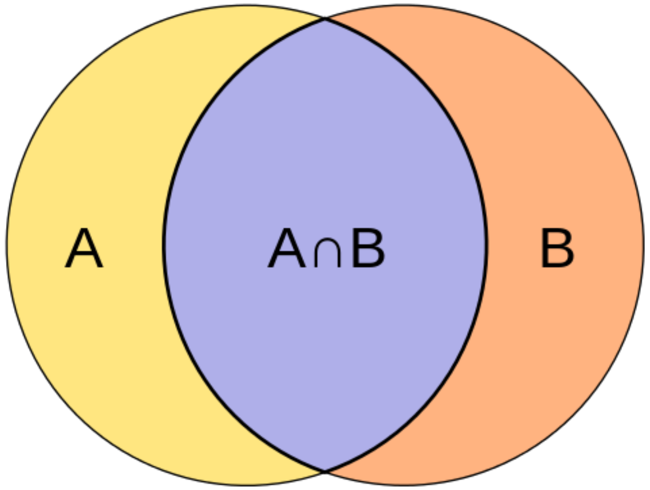
How do we evaluate the quality of prediction with respect to mask ?

Segmentation metrics

Let A and B be two finite sets, not simultaneously empty. We can measure their similarity using the *Jaccard index* or the *Dice coefficient*

Jaccard index $J(A,B) = \frac{|A \cap B|}{|A \cup B|}$ also called IoU (*Intersection over Union*)

Dice coefficient $D(A,B) = \frac{2|A \cap B|}{|A|+|B|}$



When $A = B$ we have J(A,B) = D(A,B)= 1

When $A \cap B = \emptyset$ we have J(A,B) = D(A,B)= 0

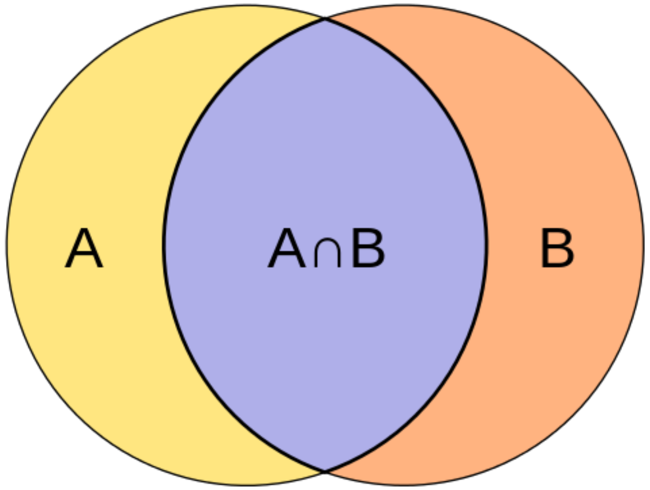
MINES PARIS | PSL

Segmentation loss

We can generalize these metrics to continuous output, $i.e\ y, \hat{y} \in [0,1]^n$

Jaccard loss $\quad J(y,\hat{y}) = 1 - \frac{y.\hat{y}+\varepsilon}{y+\hat{y}+\varepsilon}$

Dice loss $\quad \mathrm{D}(y,\hat{y}) = 1 - \frac{2\,y.\hat{y}+\varepsilon}{y+\hat{y}+\varepsilon}$



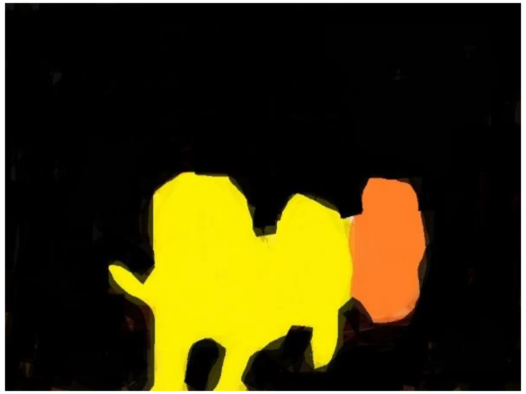
In practice, these two losses give similar results

Semantic segmentation

Training data = pairs of (image, mask)

image

mask

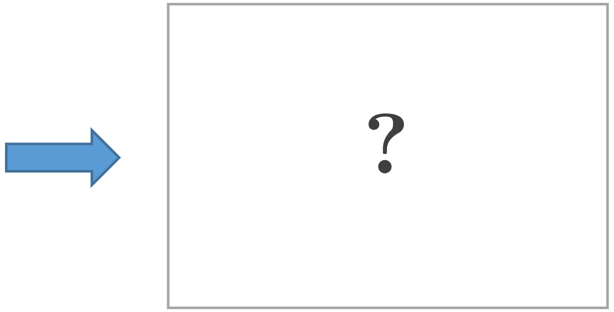


DOG,CAT,BG

For each training image, each pixel in the image is assigned a label:

- For example here - BG = 0, DOG = 1, CAT = 2

Test time



At test-time classify each pixel of the image

Semantic segmentation : pixel classification

Image $= H \times W$ pixels

label ?

1 pixel

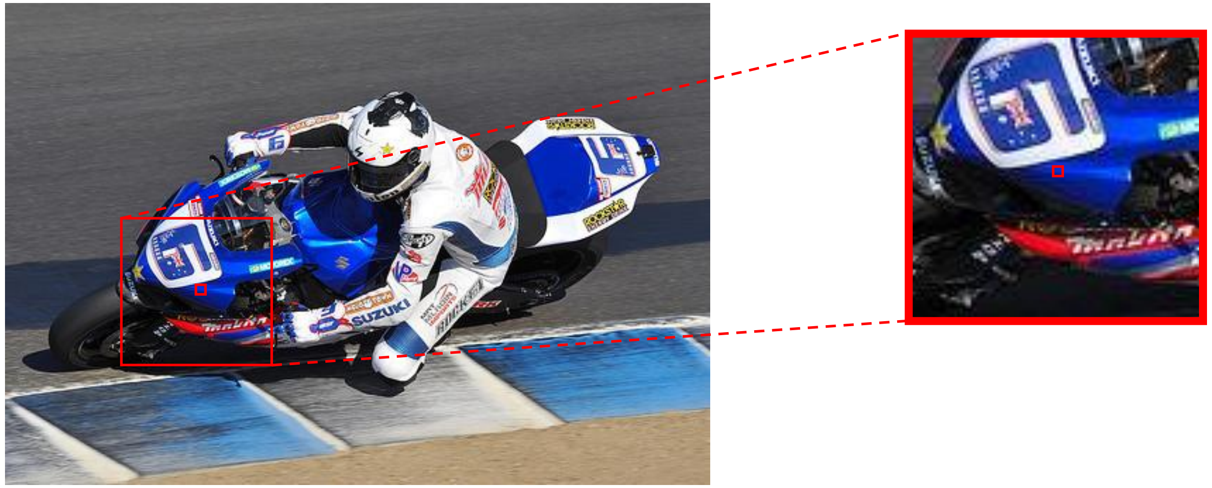
Impossible to classify a single pixel without context ..

How to include context ?

Semantic segmentation : context window

Image $= H \times W$ pixels

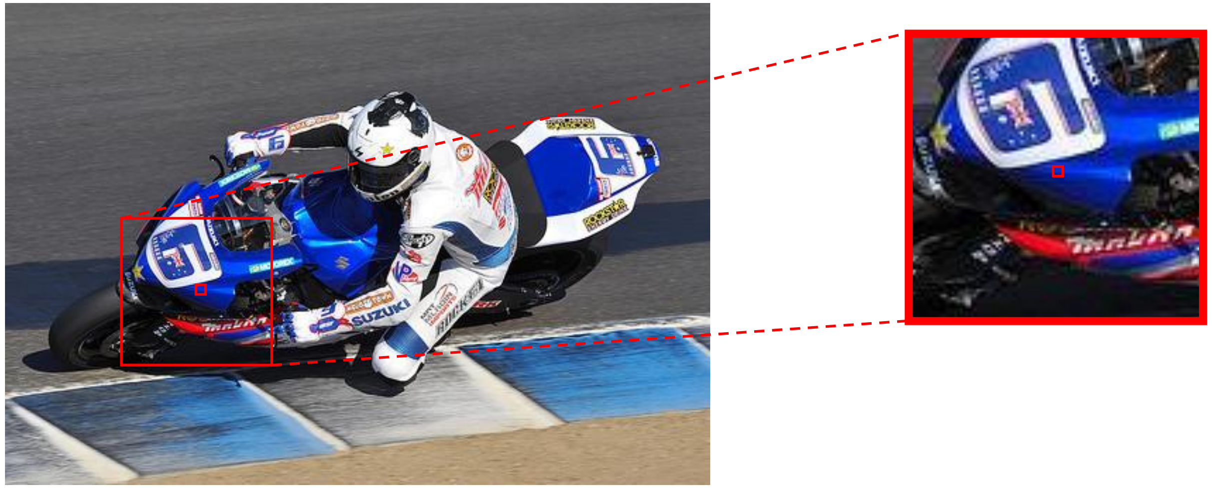
patch $= H_p \times W_p$ pixels

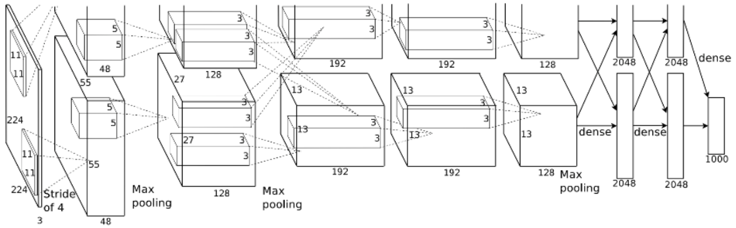


Semantic segmentation : classify window

Image $= H \times W$ pixels

patch $= H_p \times W_p$ pixels





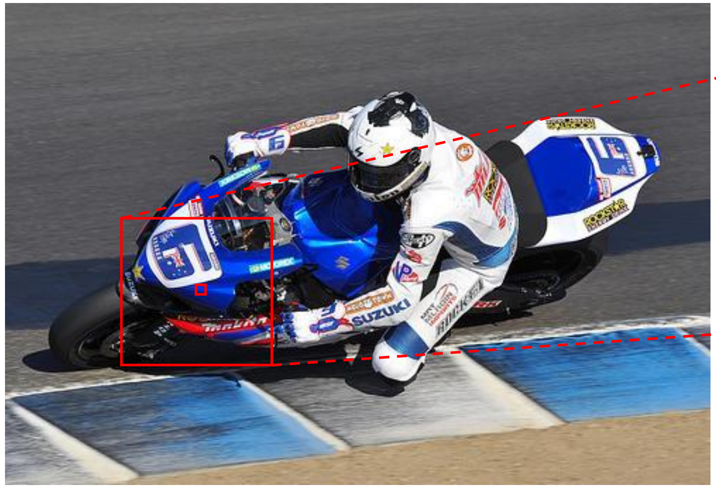
Classification network, *e.g* AlexNet

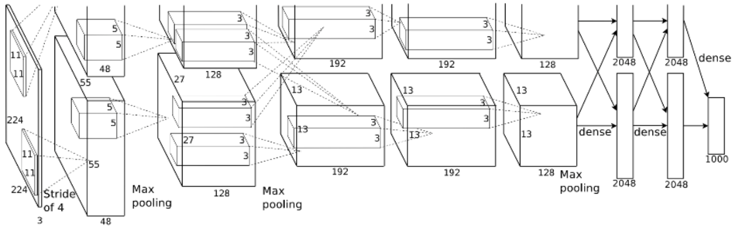
motorbike

Semantic segmentation : sliding window

Image $= H \times W$ pixels

patch $= H_p \times W_p$ pixels





Classification network, *e.g* AlexNet

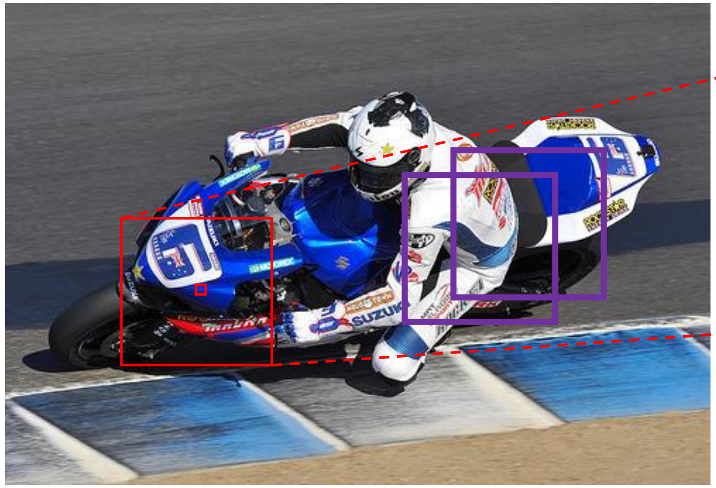
motorbike

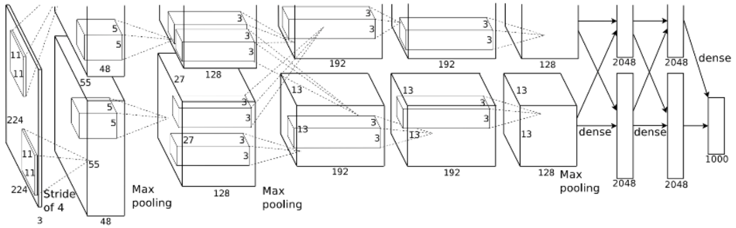
<u>Problem 1</u>: need to extract $(H \times W)$ *patches and then predict the label for each patch*

Semantic segmentation : sliding window

Image $= H \times W$ pixels

patch $= H_p \times W_p$ pixels





Classification network, *e.g* AlexNet

motorbike

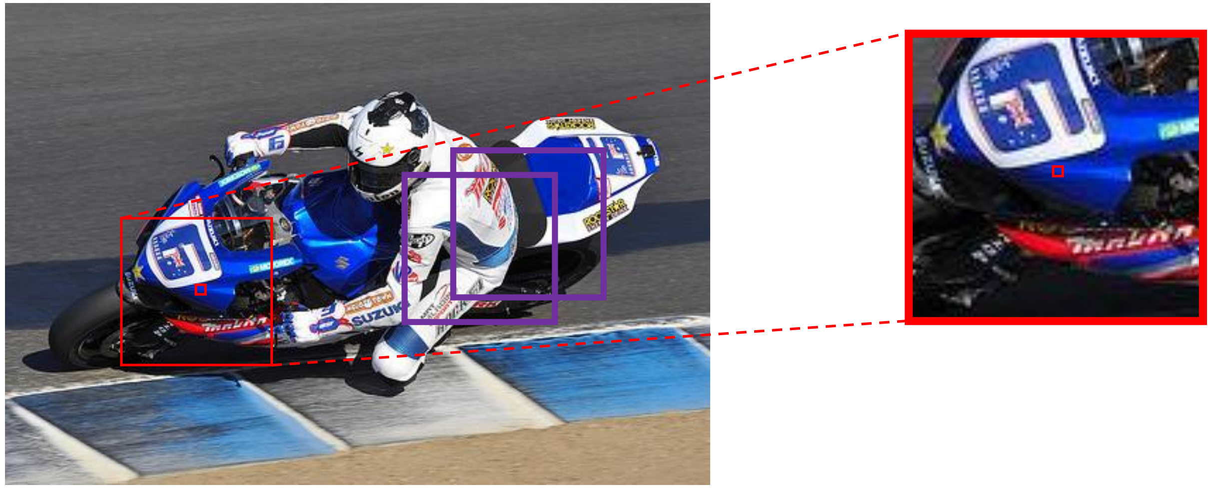
Problem 1: need to extract $(H \times W)$ *patches and then predict the label for each patch*

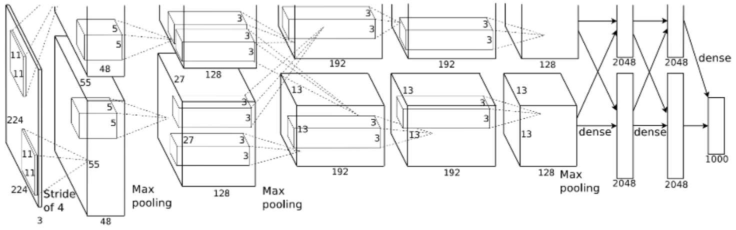
Problem 2: Does not reuse shared features between overlapping patches

Semantic segmentation : sliding window

Image $= H \times W$ pixels

patch $= H_p \times W_p$ pixels





Classification network, *e.g* AlexNet

motorbike

Problem 1: need to extract $(H \times W)$ *patches and then predict the label for each patch*

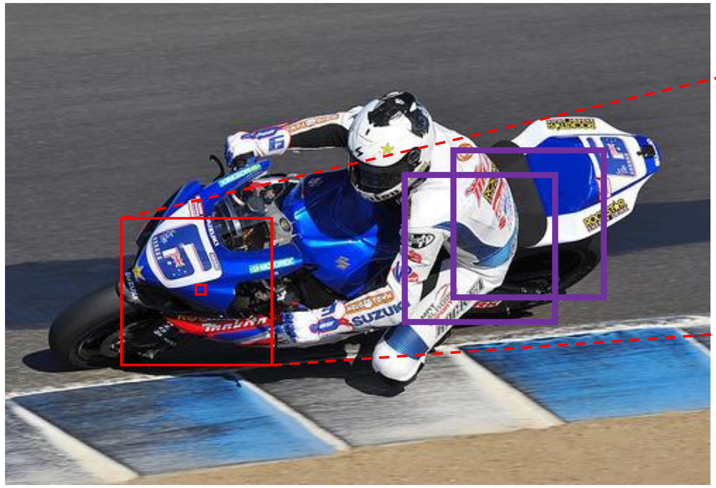
Problem 2: Does not reuse shared features between overlapping patches

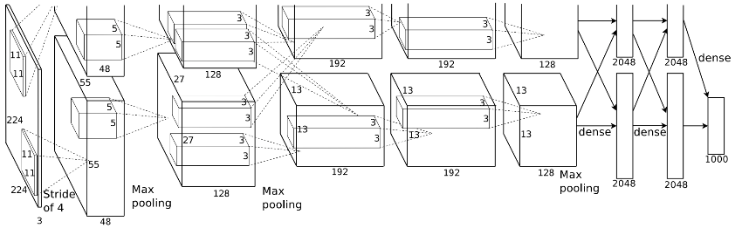
Solution: ?

Semantic segmentation : Fully convolutional

Image $= H \times W$ pixels

patch $= H_p \times W_p$ pixels





Classification network, *e.g* AlexNet

motorbike

Problem 1: need to extract $(H \times W)$ *patches and then predict the label for each patch*

Problem 2: Does not reuse shared features between overlapping patches

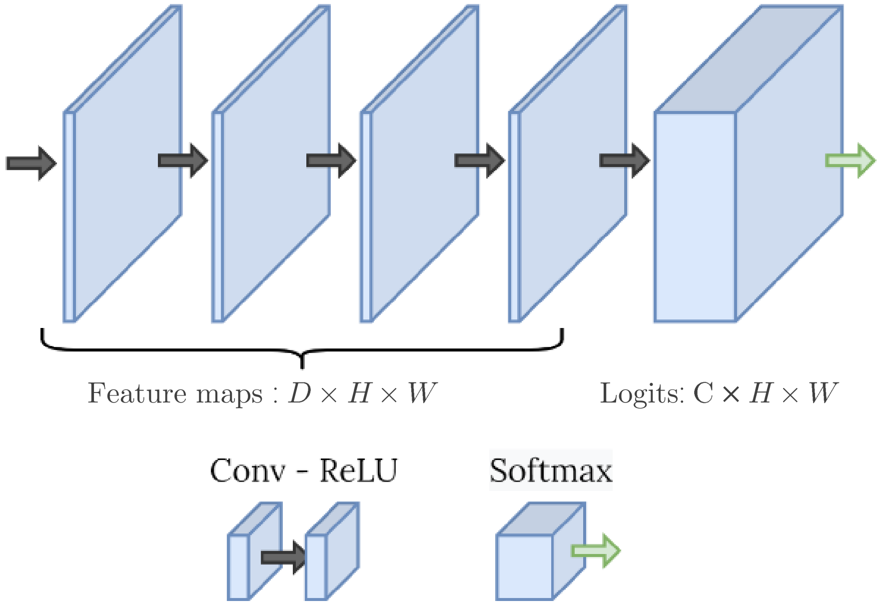
Solution: ?

MINES PARIS | PSL

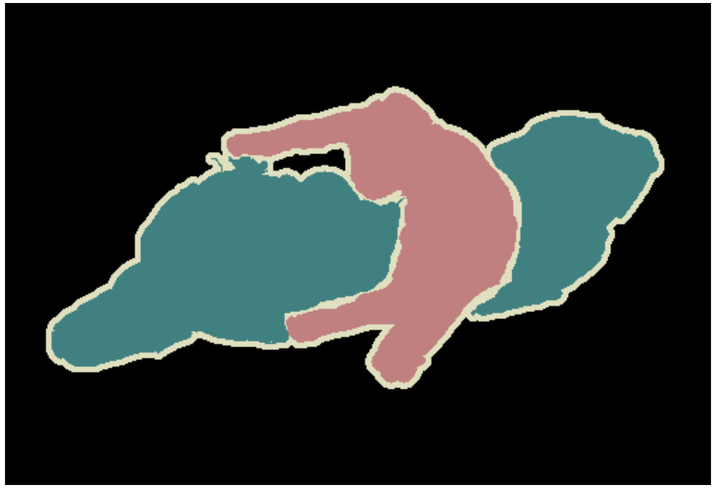
Semantic segmentation : Fully convolutional

Image $= H \times W$ pixels

CNN with no down-sampling *ops*



Predictions $= H \times W$ pixels

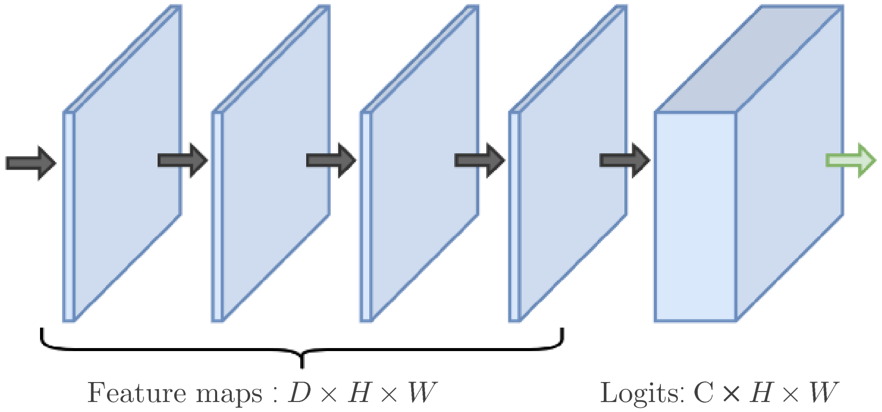


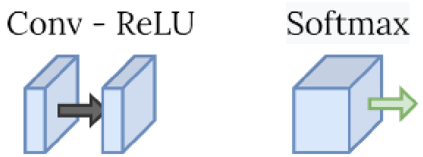
C = 3 classes

Semantic segmentation : Fully convolutional

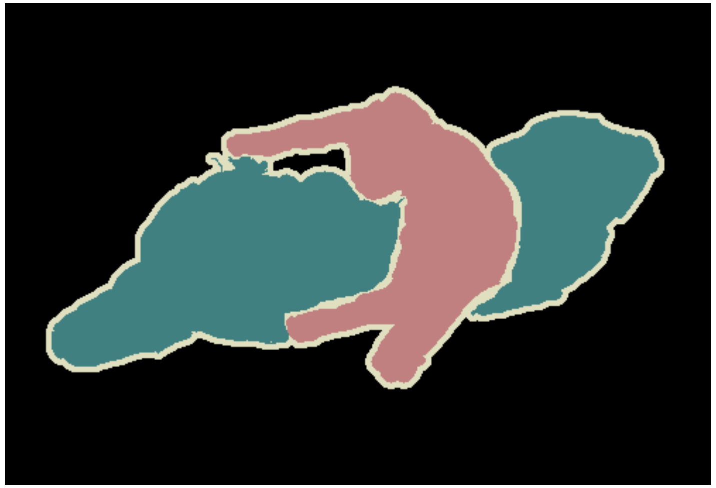
Image $= H \times W$ pixels

CNN with no down-sampling *ops*





Predictions $= H \times W$ pixels



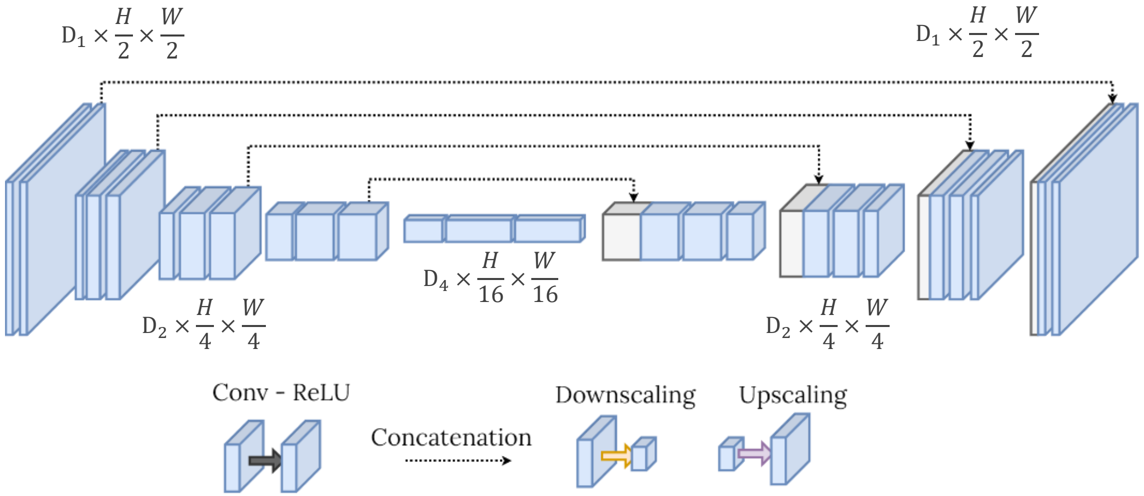
C = 3 classes

<u>Problem 1</u>: computationally expansive and memory consuming

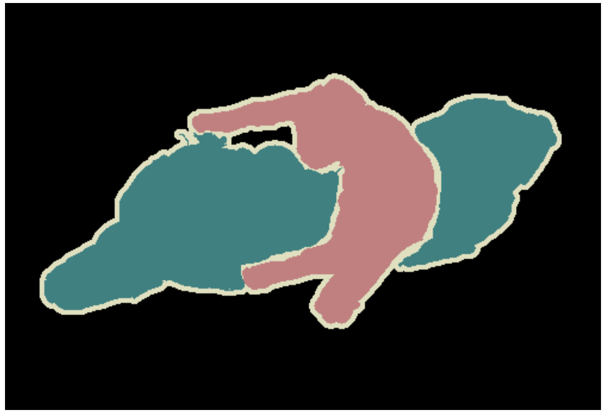
<u>Solution</u>: ?

Semantic segmentation : Encoder – Decoder structure

Image $= H \times W$ pixels



Predictions $= H \times W$ pixels

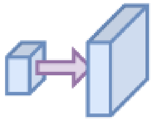


C = 3 classes

- Keep the encoder-like structure of classification networks
- Use *upsampling ops* to recover the initial image resolution
- Mix information from encoder-path with decoder-path for better localization accuracy

Decoder : *upsampling*

Upscaling



Nearest neighbors:

1	2
3	4

→

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

Bilinear interpolation:

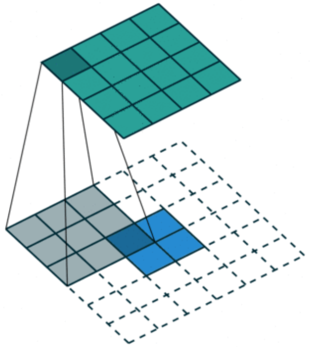
10	20
30	40

→

10	12	17	20
15	17	22	25
25	27	32	32
30	32	37	40

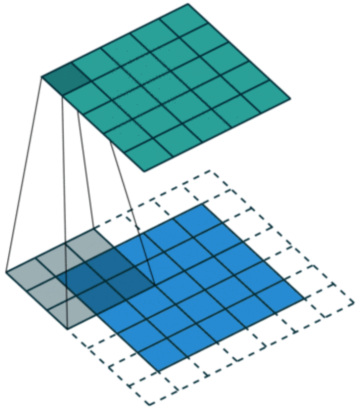
Decoder : *upsampling*

Transposed convolution:



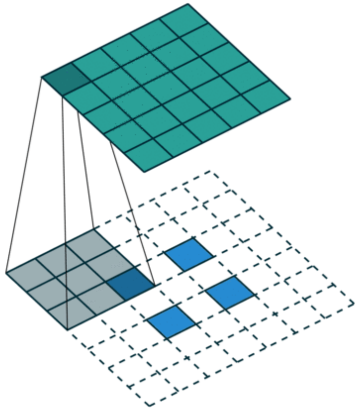
padding $= 0$
stride $= 1$
Kernel $= 3 \times 3$

$2 \times 2 \rightarrow 4 \times 4$



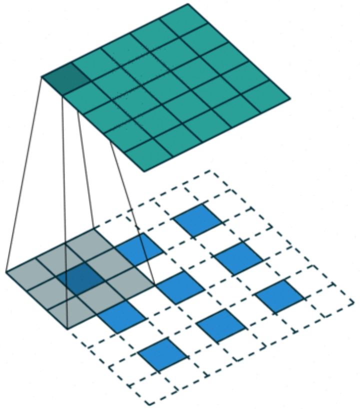
padding $= 1$
stride $= 1$
Kernel $= 3 \times 3$

$5 \times 5 \rightarrow 5 \times 5$



padding $= 0$
stride $= 2$
Kernel $= 3 \times 3$

$2 \times 2 \rightarrow 5 \times 5$



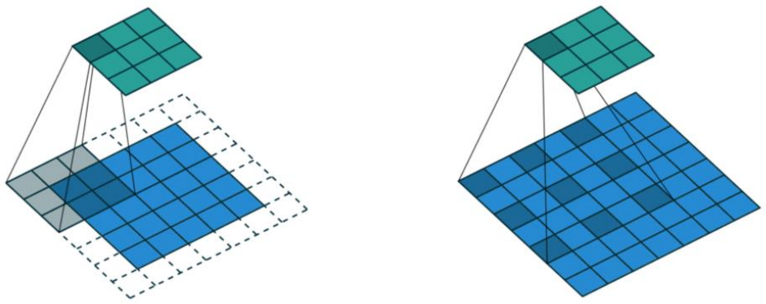
padding $= 1$
stride $= 2$
Kernel $= 3 \times 3$

$3 \times 3 \rightarrow 5 \times 5$

https://hannibunny.github.io/mlbook/neuralnetworks/convolutionDemos.html

Architecture : *an alternative to downsampling*

Dilated convolution or *atrous* convolution :



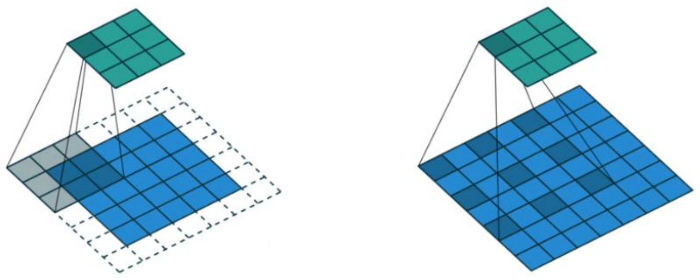
(a) *A simple convolution* $(r = 1)$ **(b)** *A dilated convolution* $(r = 2)$

The aim is to increase the receptive field and keep a dense (high resolution) feature map.

- dense map = better localization

Architecture : *an alternative to downsampling*

Dilated convolution or *atrous* convolution :



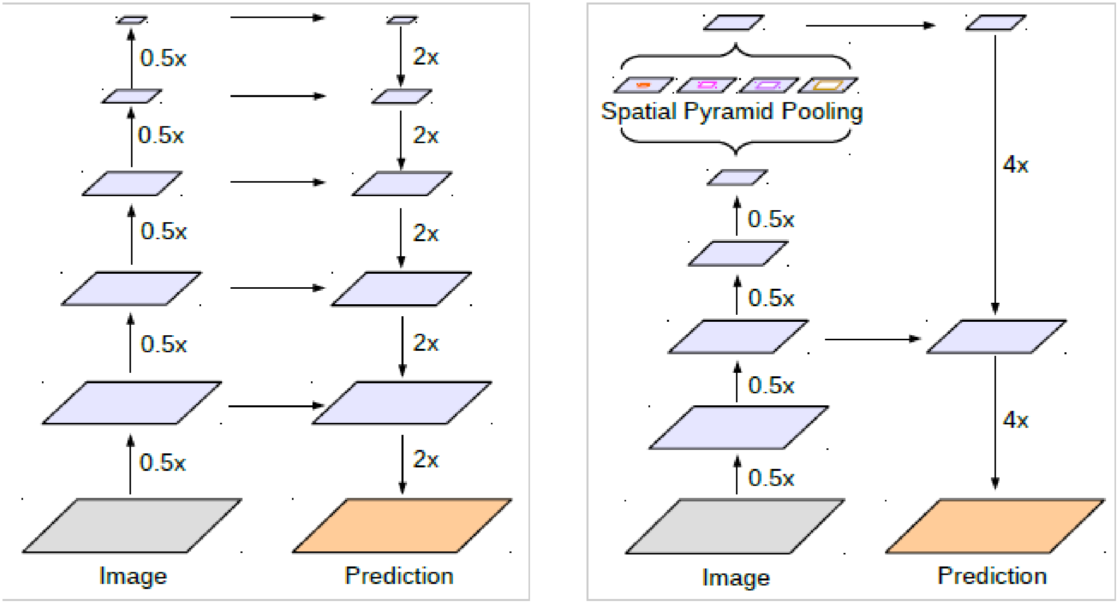
(a) *A simple convolution* $(r = 1)$ (b) *A dilated convolution* $(r = 2)$

The aim is to increase the receptive field and keep a dense (high resolution) feature map.

- dense map = better localization

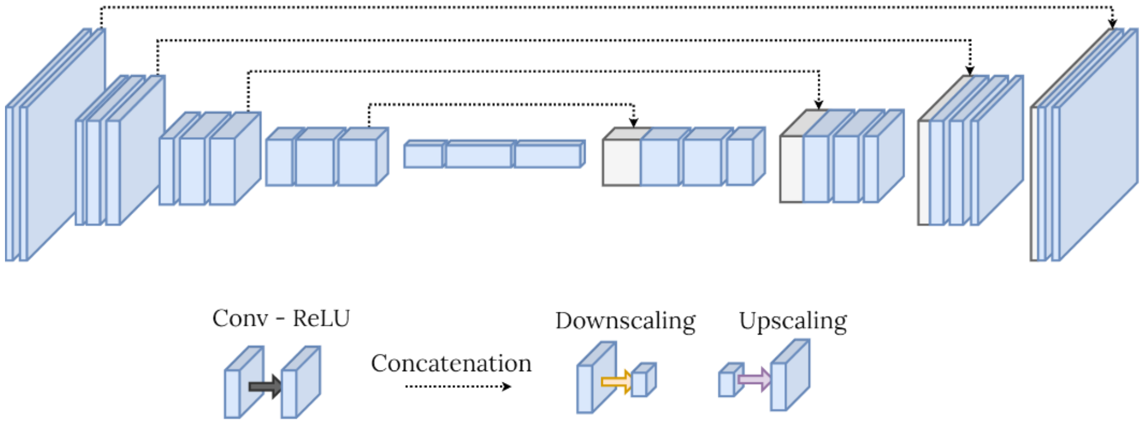
DeepLabV3+ architecture :

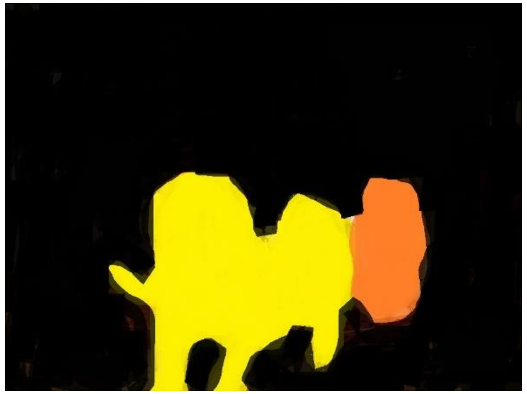
- tradeoff computation budget for performance



(b) Encoder-Decoder (c) Encoder-Decoder with Atrous Conv

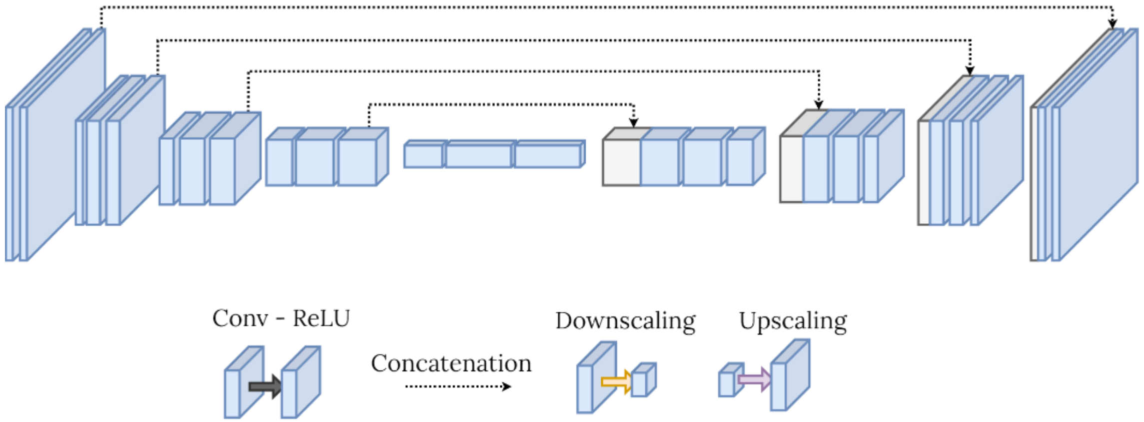
Semantic segmentation : Summary

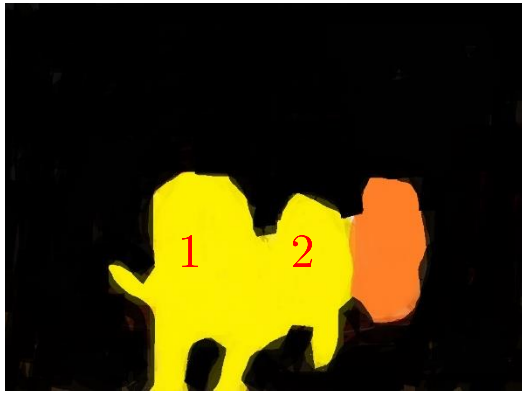




Semantic segmentation labels each pixel in an image

Semantic segmentation : Summary





Semantic segmentation labels each pixel in an image

Semantic segmentation cannot differentiate multiple instances of the same category

i.e the two DOGS in the photo

State of the arts methods

CNN-based :

- 2015, Ronneberger et al “ U-Net: Convolutional Networks for Biomedical Image Segmentation ”
- 2018, Chen et al “ Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation ” = DeepLabV3
- 2021, Wang et al “ Deep High-Resolution Representation Learning for Visual Recognition ” = HRNetV2 ”

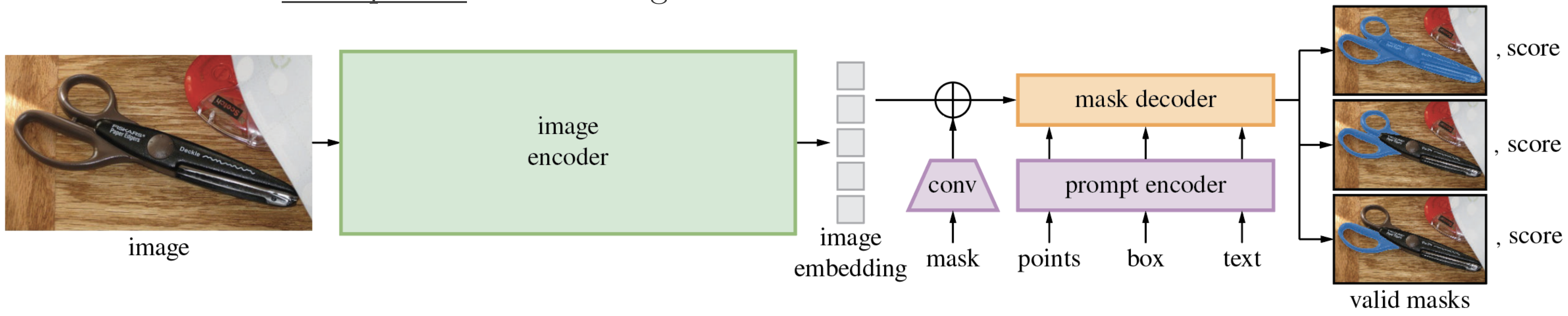
Transformer-based :

- 2021, Xie et al, “ SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers ”
- 2023, Chen et al, “ Vision Transformer Adapter for Dense Predictions ”

https://paperswithcode.com/task/semantic-segmentation
https://github.com/qubvel/segmentation_models/tree/master

Segment Anything

What is SAM → Promptable instance segmentation network



The dataset and the model are open-sourced : https://github.com/facebookresearch/segment-anything

- trained on 11 million images containing 1 billion masks !

Do you have a use case for SAM ?

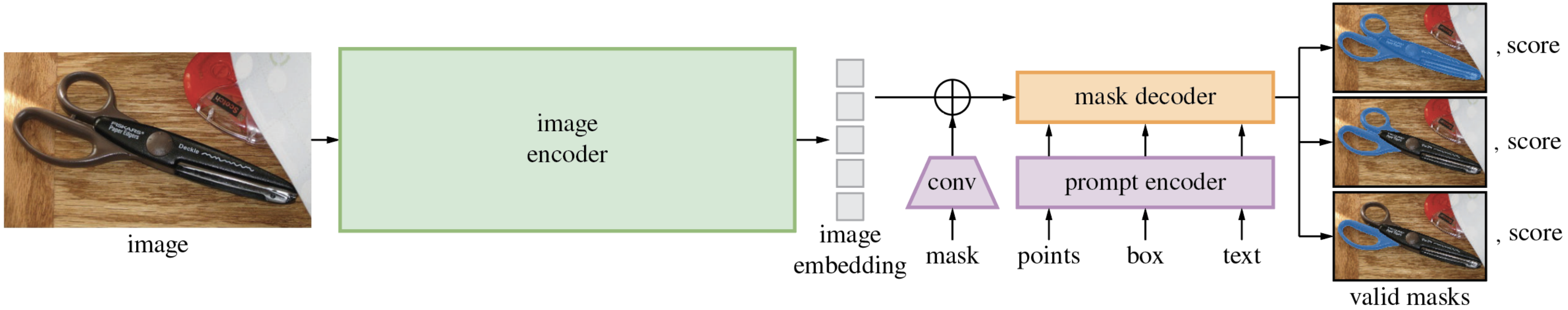
- the promptable feature of SAM makes it a go-to for fast zero-shot prototyping

Example : Let's say you only have an algorithm for localizing the center of specific objects,

SAM could be able to segment these objects using your point inputs

Segment Anything

What is SAM → Promptable instance segmentation network



The dataset and the model are open-sourced : https://github.com/facebookresearch/segment-anything

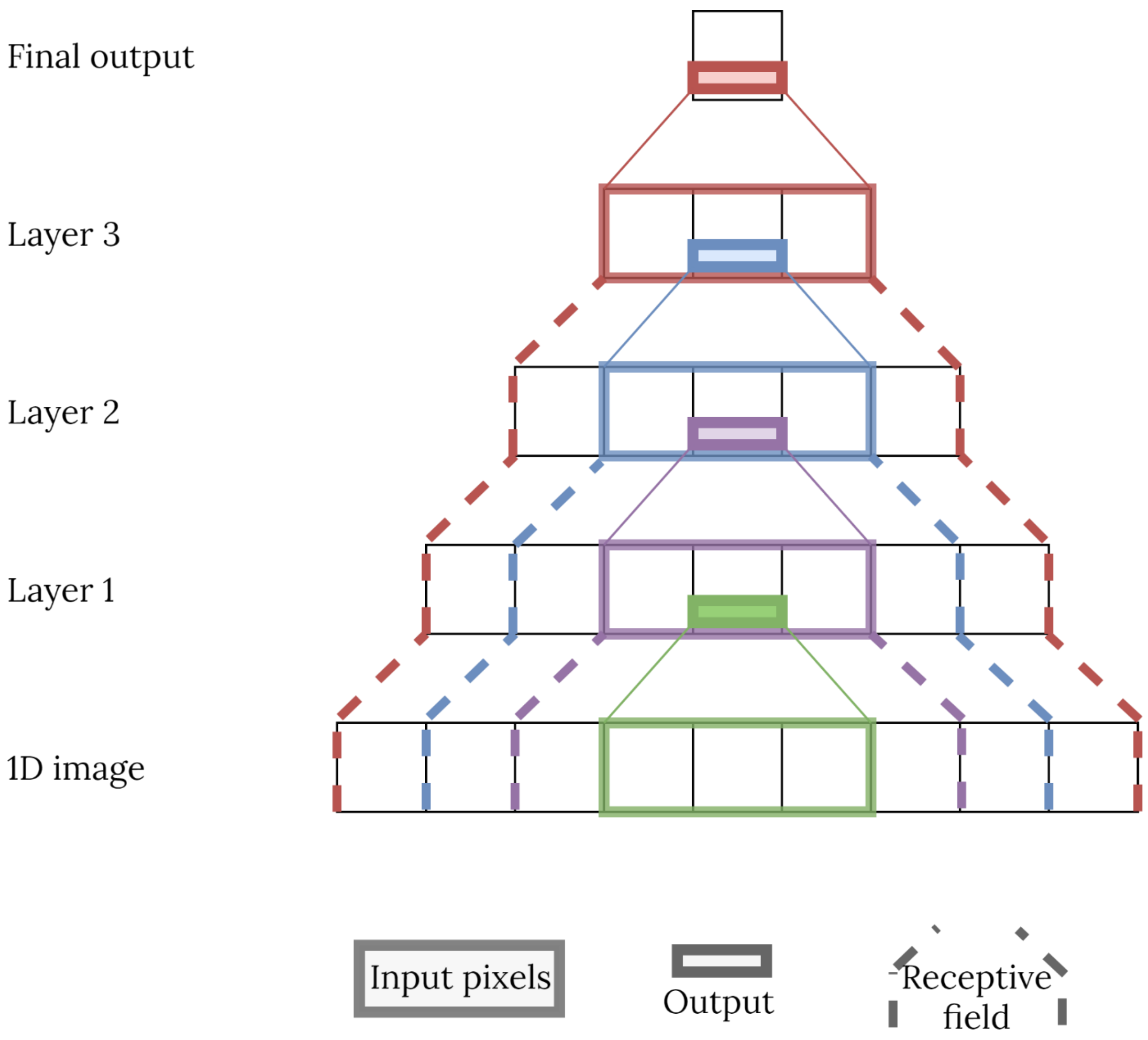
- trained on 11 million images containing 1 billion masks !

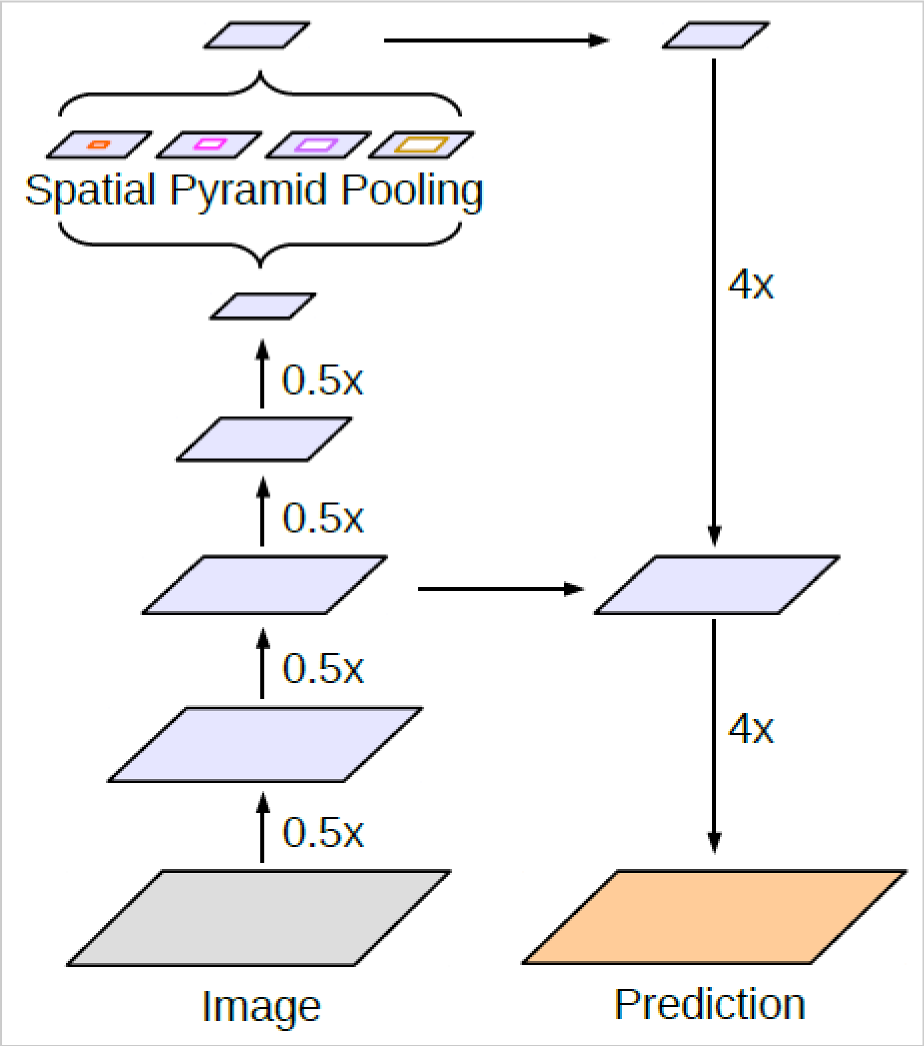
DEMO : https://segment-anything.com/demo

SAM API : https://huggingface.co/docs/transformers/main/model_doc/sam

Thank you for your attention

*slides adapted from CS231n


Final output
Layer 3
Layer 2
Layer 1
1D image
Input pixels
Output
Receptive field



(c) Encoder-Decoder with Atrous Conv

MINES PARIS | PSL

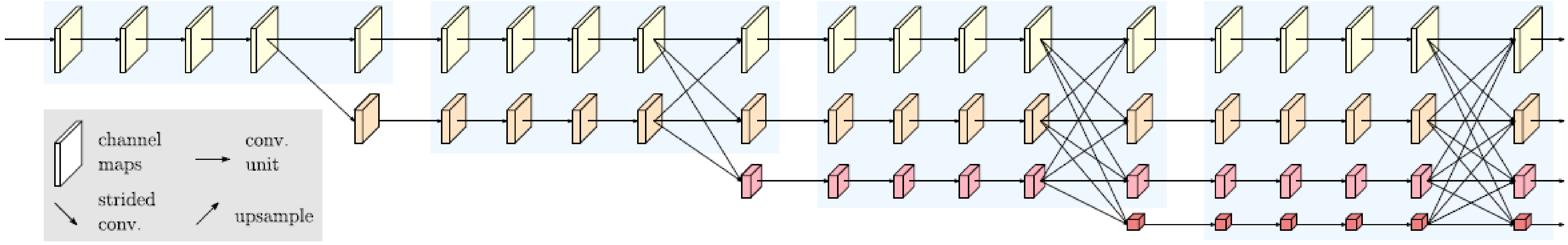


Fig. 2. An example of a high-resolution network. Only the main body is illustrated, and the stem (two stride-2 3×3 convolutions) is not included. There are four stages. The 1st stage consists of high-resolution convolutions. The 2nd (3rd, 4th) stage repeats two-resolution (three-resolution, four-resolution) blocks. The detail is given in Section 3.