SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

Semantic and instance segmentation

SCHOOL OF HUMAN SCIENCES & TECHNOLOGY

Chapter Goals

After completing this chapter, you should be able to understand :

- Semantic segmentation
- Semantic Segmentation architecture
- Instance Segmentation architecture
- Example of semantic segmentation architecture: U Net
- Example of instance segmentation architecture: Mask R CNN
- Python implementation of U Net and Mask R CNN
- SAM as foundational model for segmentation
- FastSAM as a real time alternative to SAM

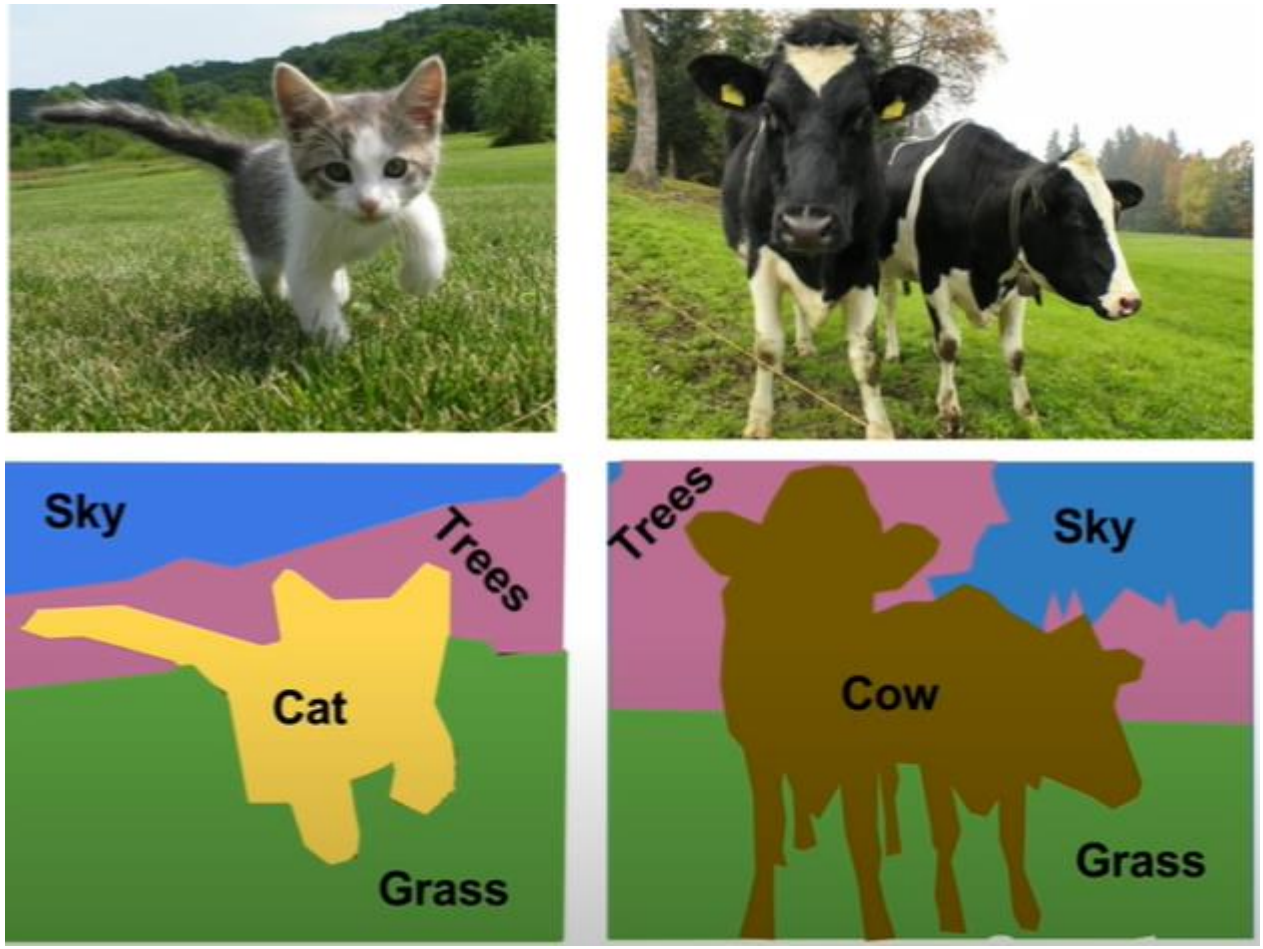
ie
SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

Semantic Segmentation

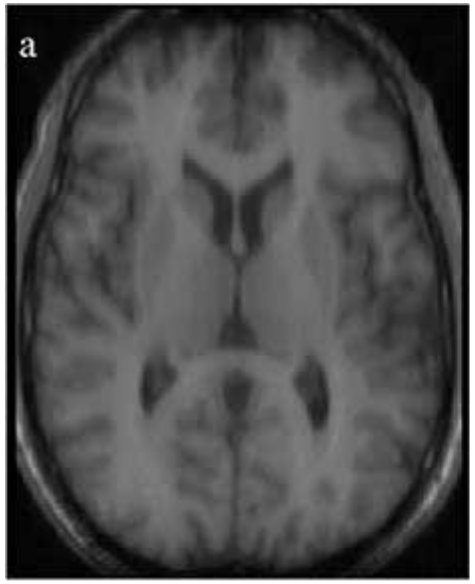
Semantic Segmentation

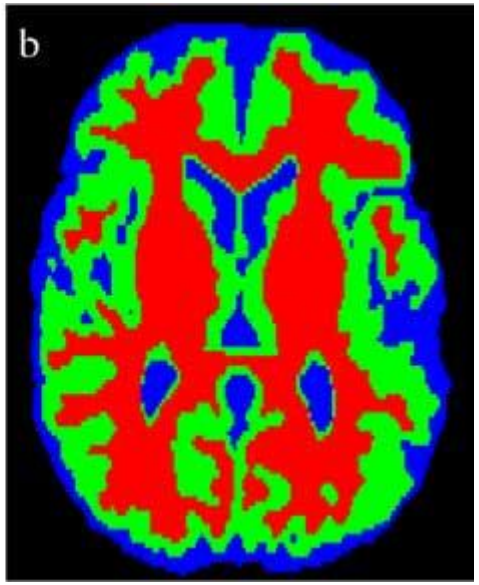
Aim: label each pixel in the image with a category

Doesn't care about instances, only cares about pixels



Applications of Semantic Segmentation



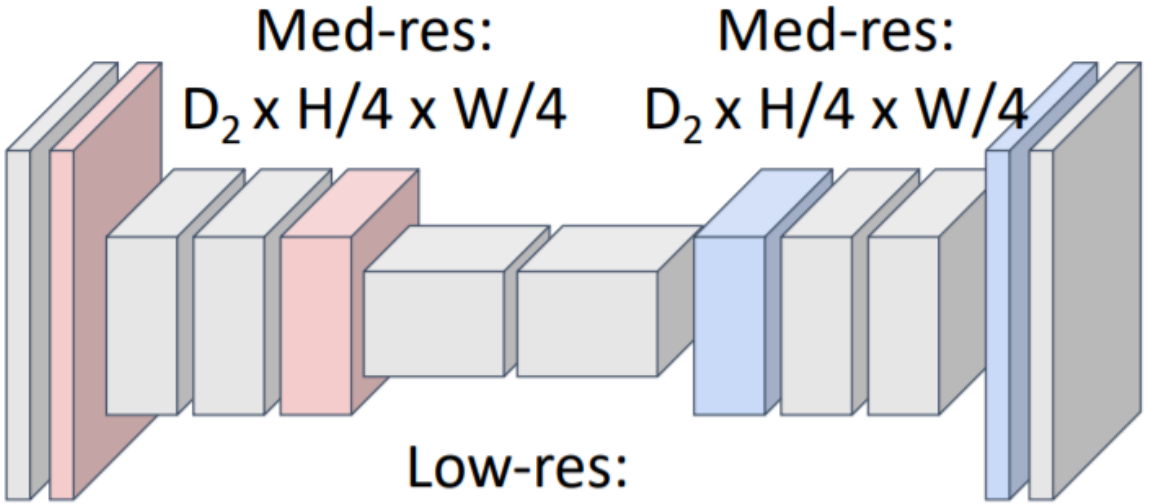


Medical applications

Autonomous driving

Semantic Segmentation: Fully Convolutional Network

Input:
3 x H x W



Predictions:
H x W

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

Pooling, strided convolution

Upsampling :Max-Unpooling

1	2	6	3
3	5	2	1
1	2	2	1
7	3	4	8

→

5	6
7	8

→ Rest of net →

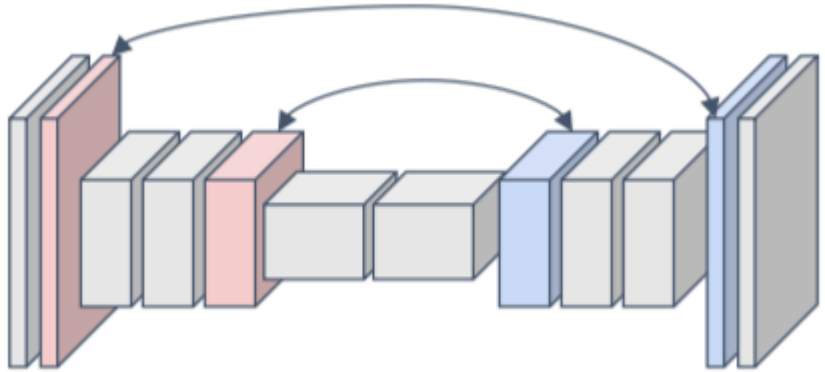
1	2
3	4

→

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

Max Pooling: Remember which position had the max

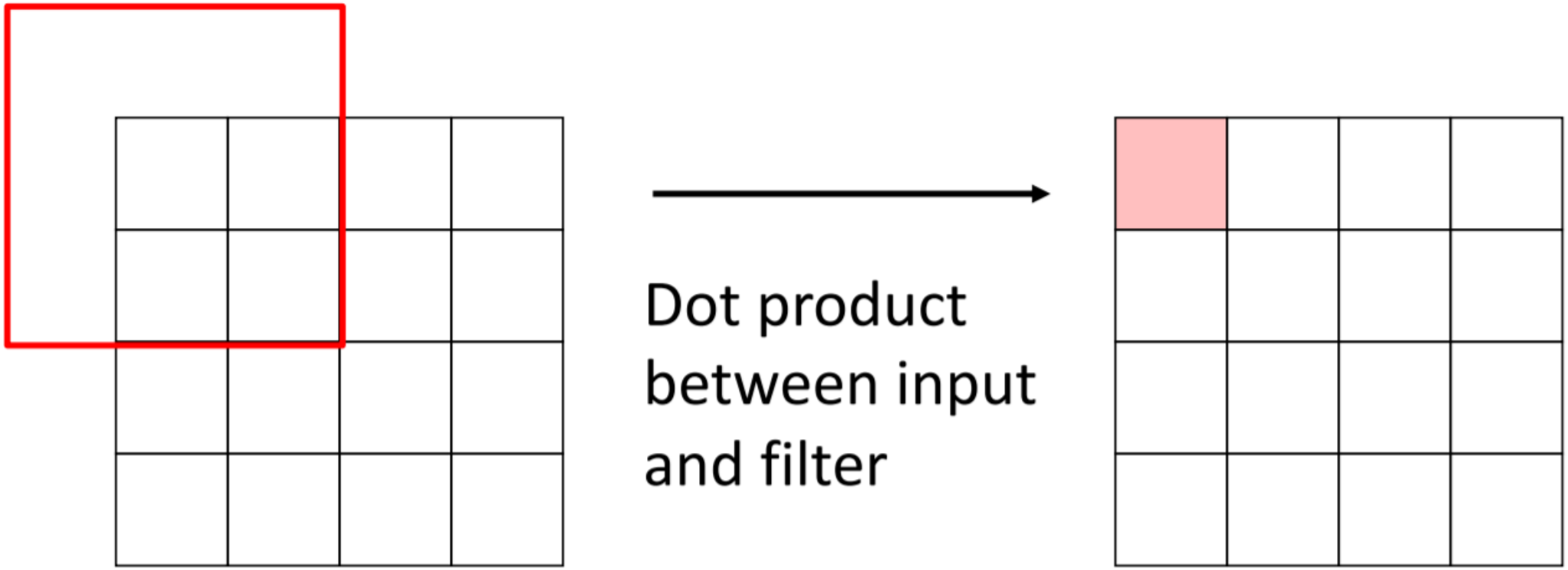
Max Unpooling: Place into remembered positions



Pair each downsampling layer with an upsampling layer

Learnable Upsampling: Transposed convolution

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

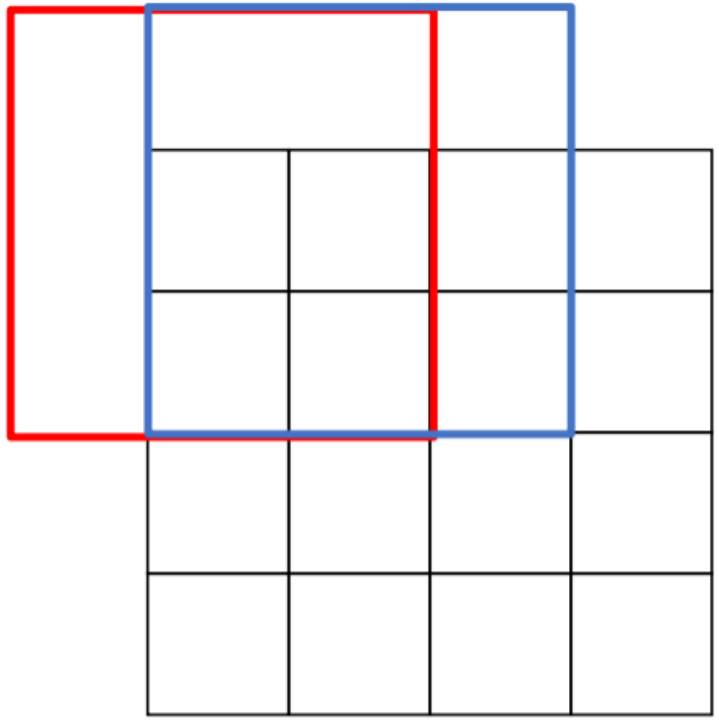


Input: 4 x 4

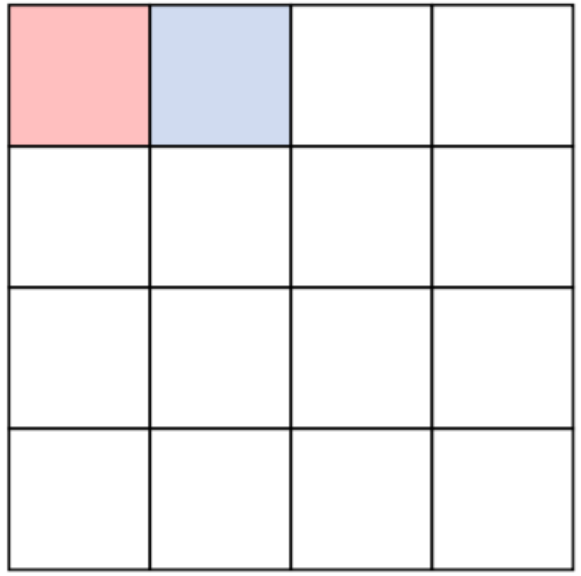
Output: 4 x 4

Learnable Upsampling: Transposed convolution

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



Dot product between input and filter



Input: 4 x 4

Output: 4 x 4

Learnable Upsampling: Transposed convolution

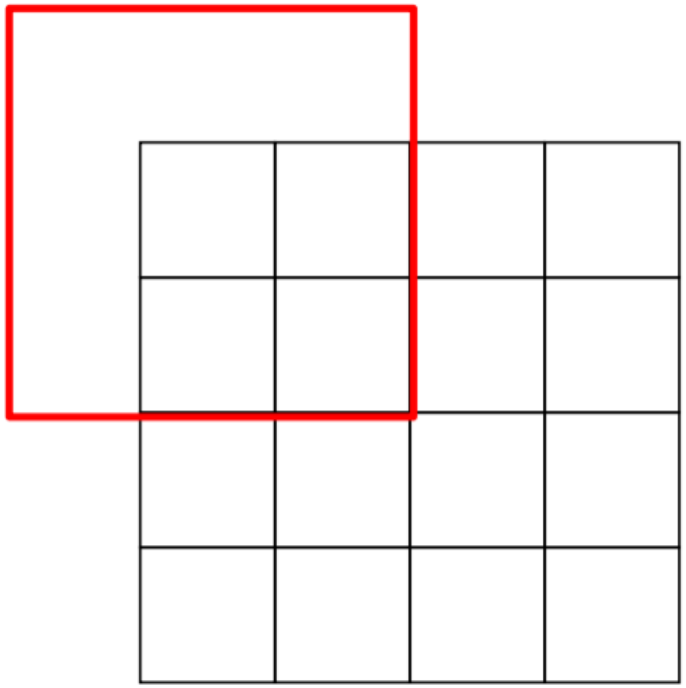
Recall: Normal 3 x 3 convolution, stride 2, pad 1

Input: 4 x 4

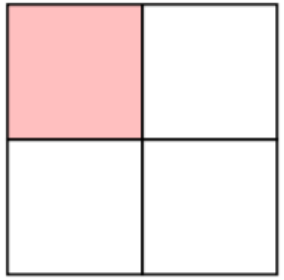
Output: 2 x 2

Learnable Upsampling: Transposed convolution

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



Dot product between input and filter

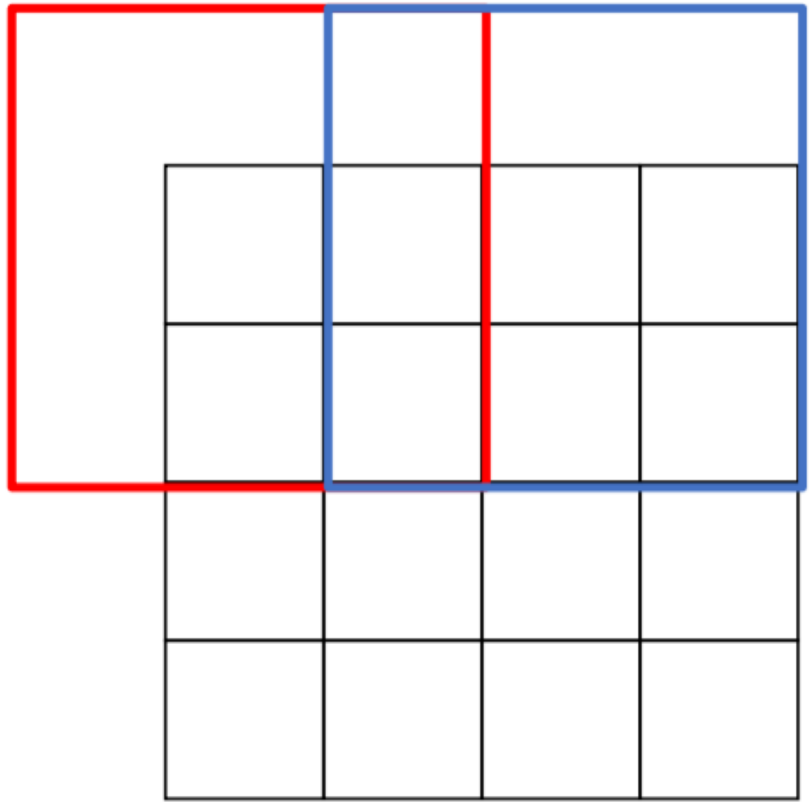


Input: 4 x 4

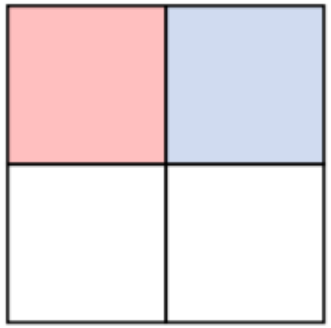
Output: 2 x 2

Learnable Upsampling: Transposed Convolution

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



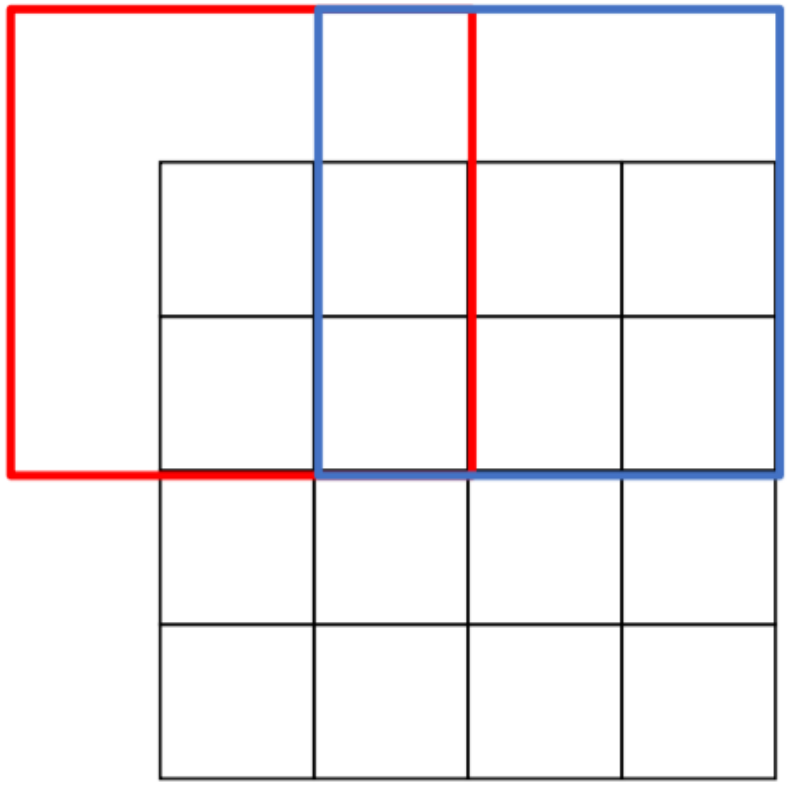
Dot product between input and filter



Input: 4 x 4

Output: 2 x 2

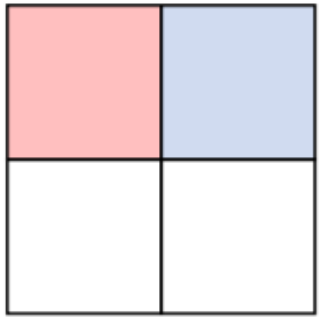
Recall: Normal 3 x 3 convolution, stride 2, pad 1



Convolution with stride > 1 is "Learnable Downsampling"

Can we use stride < 1 for "Learnable Upsampling"?

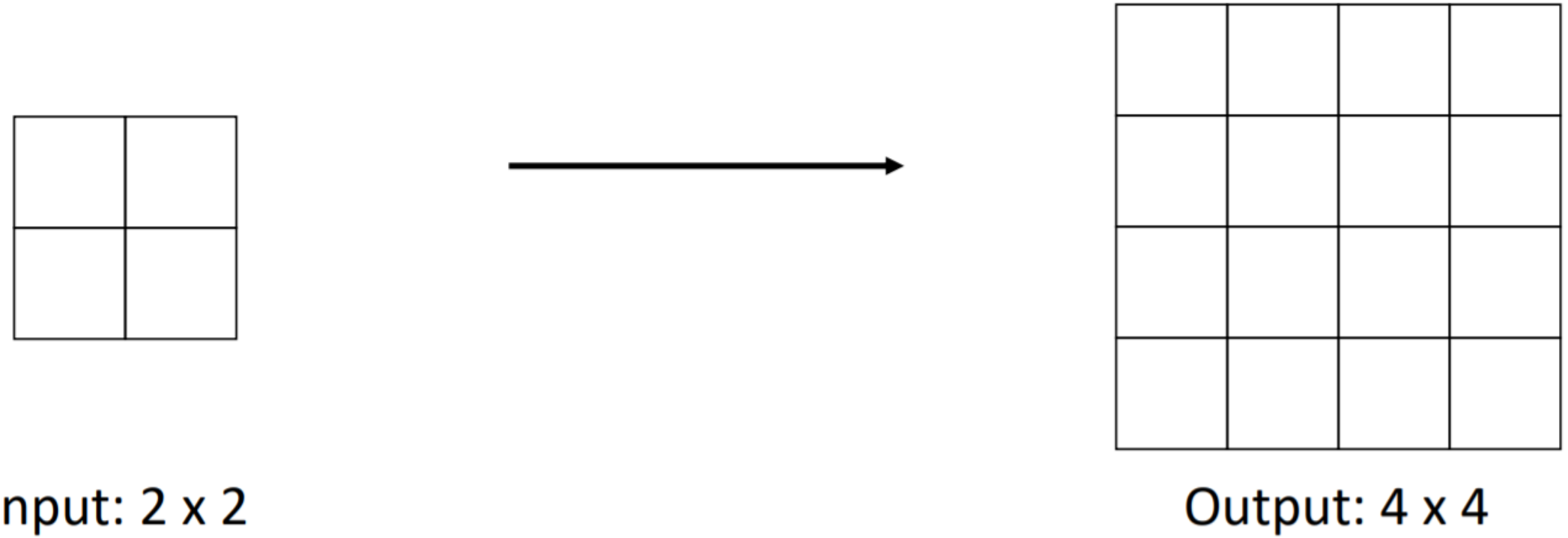
Dot product between input and filter



Input: 4 x 4

Output: 2 x 2

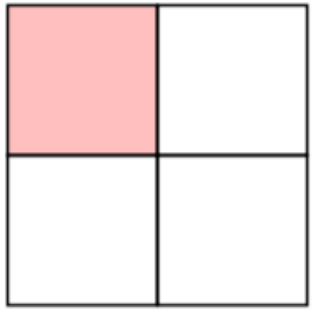
3 x 3 **convolution transpose**, stride 2



Input: 2 x 2

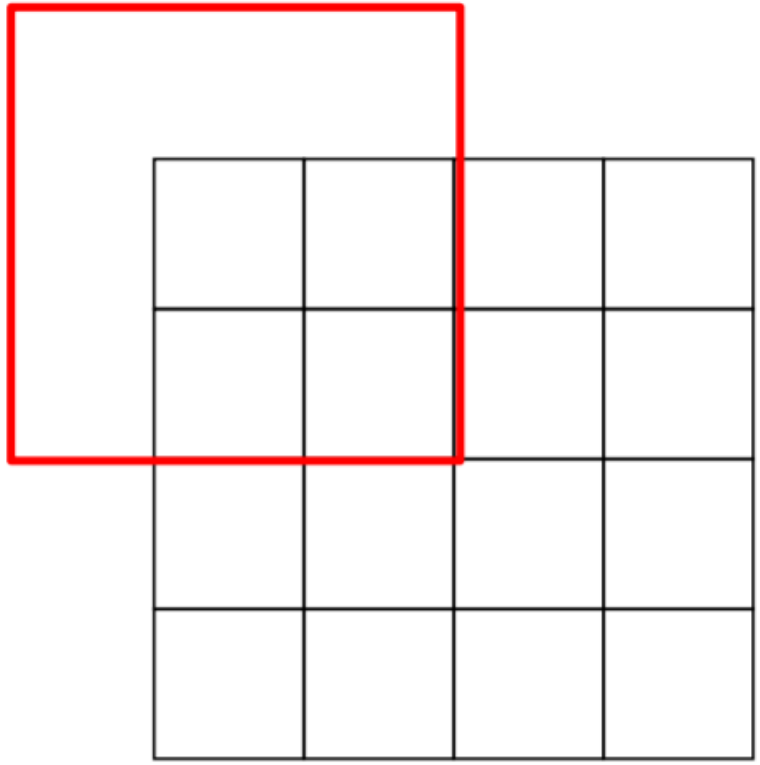
Output: 4 x 4

3 x 3 **convolution transpose**, stride 2



Input: 2 x 2

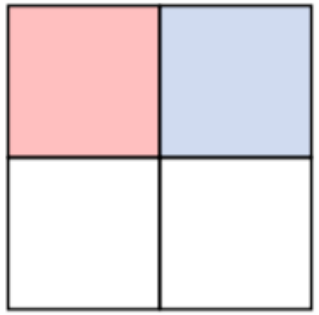
Weight filter by input value and copy to output



Output: 4 x 4

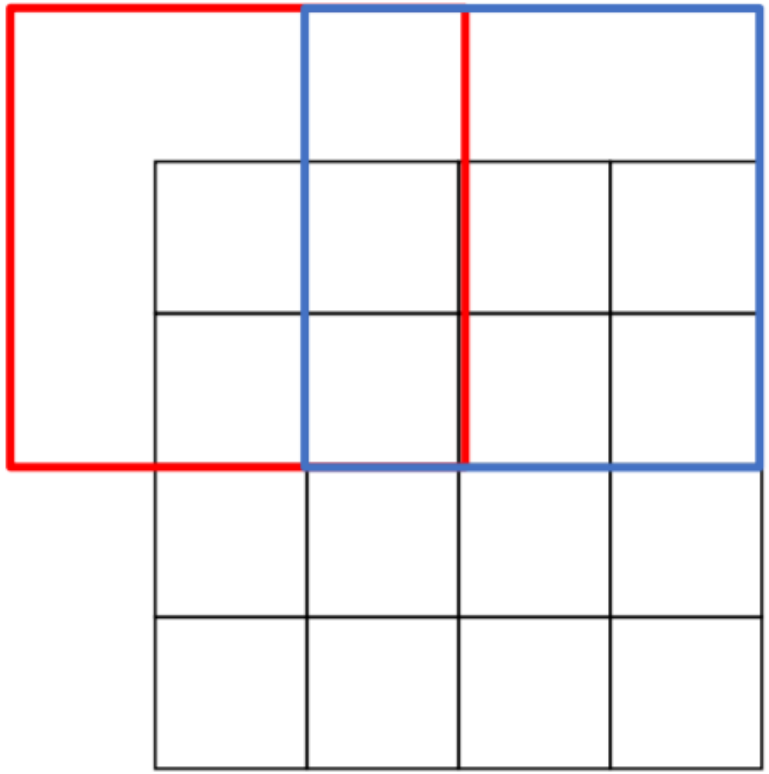
3 x 3 **convolution transpose**, stride 2

Filter moves 2 pixels in output
for every 1 pixel in input



Input: 2 x 2

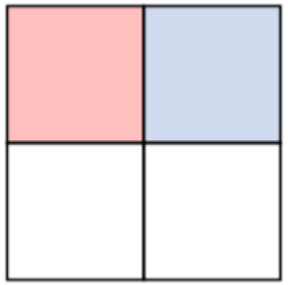
Weight filter by
input value and
copy to output



Output: 4 x 4

3 x 3 **convolution transpose**, stride 2

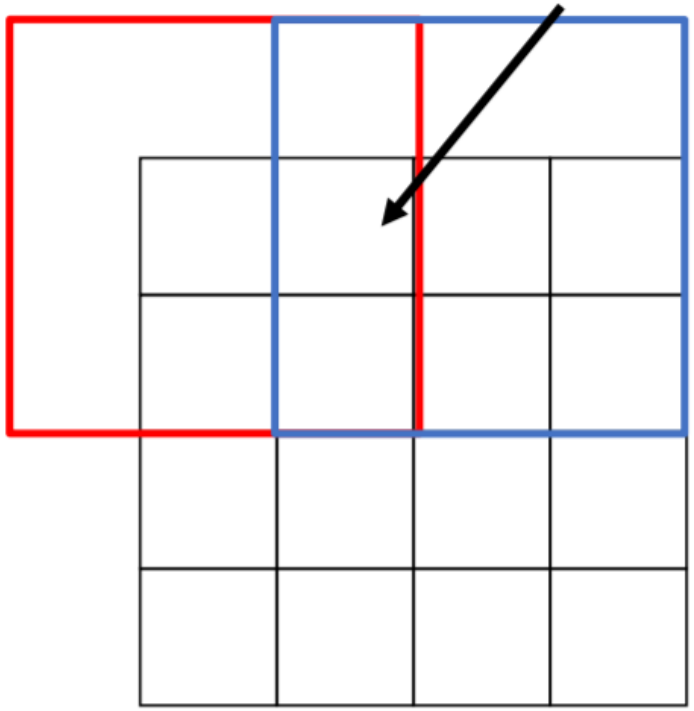
Filter moves 2 pixels in output
for every 1 pixel in input



Input: 2 x 2

Weight filter by
input value and
copy to output

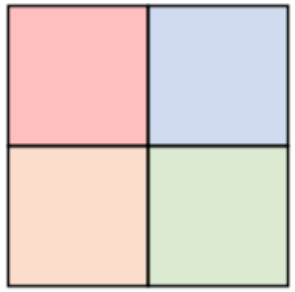
Sum where
output overlaps



Output: 4 x 4

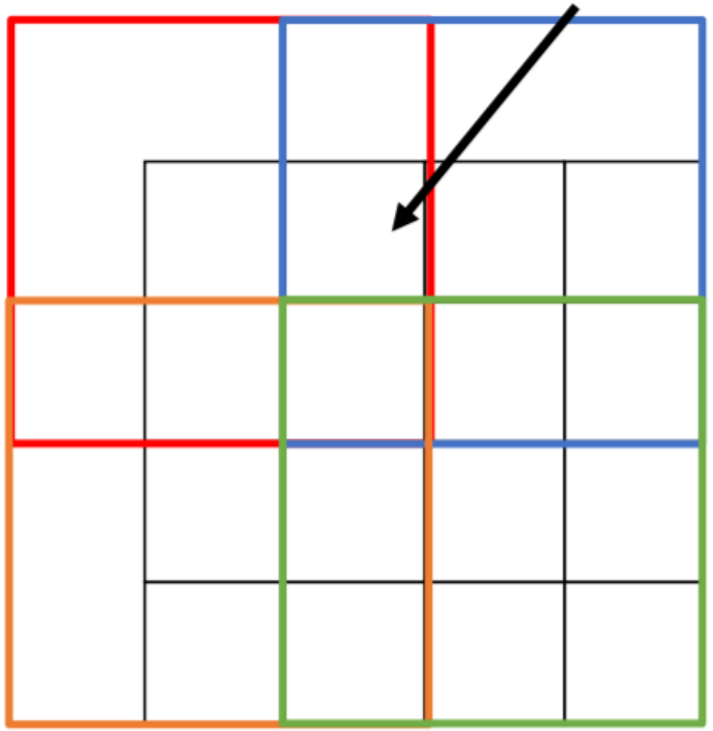
3 x 3 **convolution transpose**, stride 2

This gives 5x5 output – need to trim one pixel from top and left to give 4x4 output



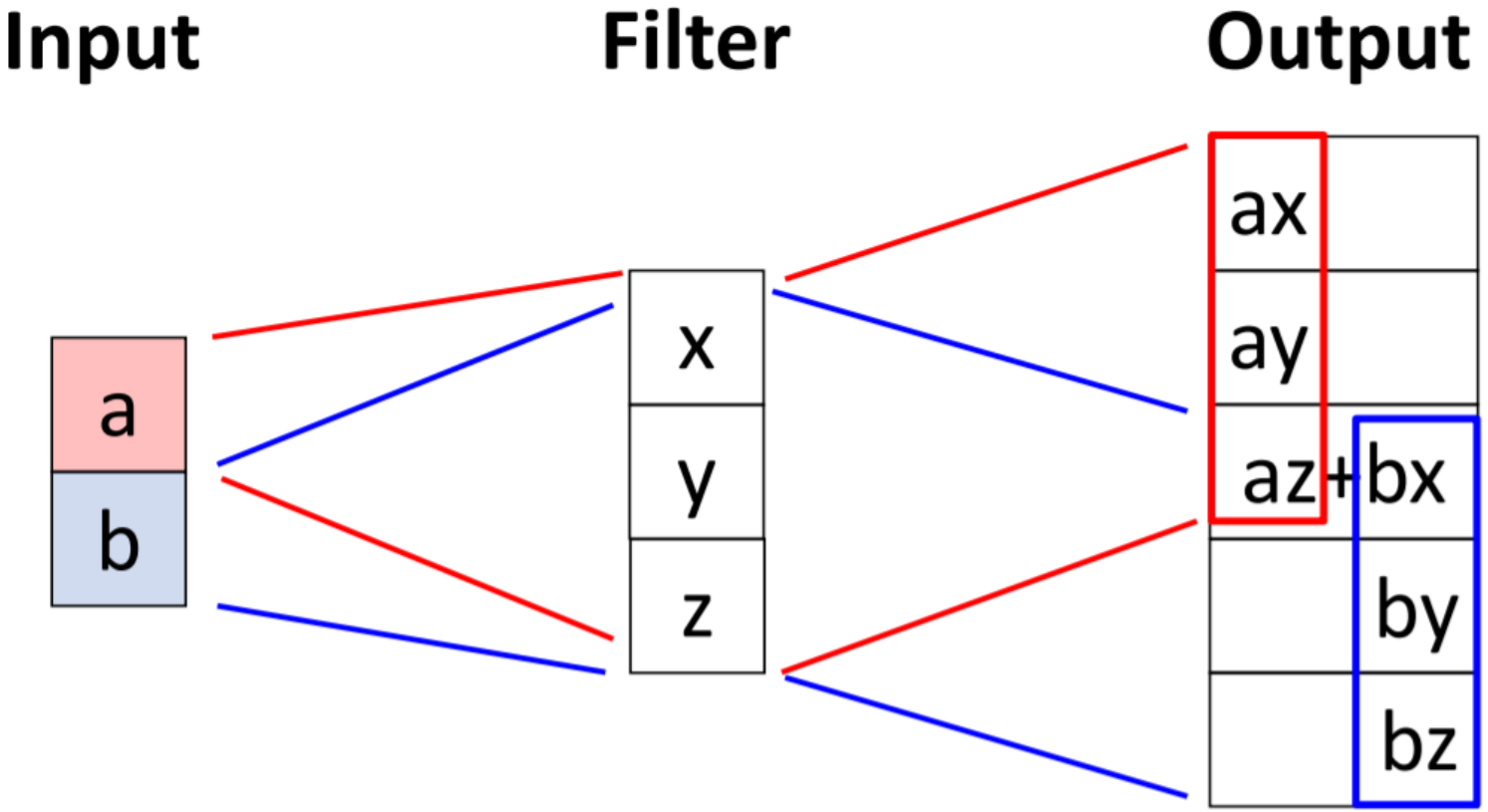
Input: 2 x 2

Weight filter by input value and copy to output



Output: 4 x 4

Transposed Convolution: 1D example

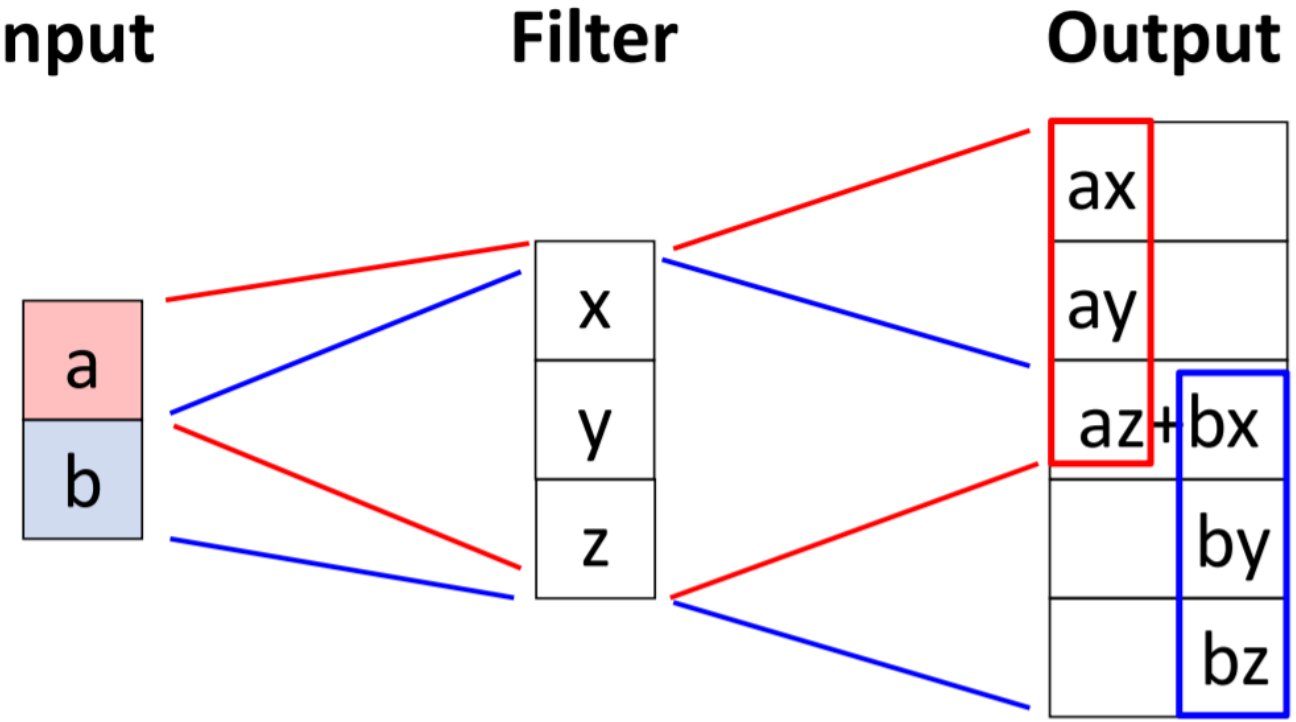


Output has copies of filter weighted by input

Stride 2: Move 2 pixels output for each pixel in input

Sum at overlaps

Transposed Convolution: 1D example



This has many names:

- Deconvolution (bad)!
- Upconvolution
- Fractionally strided convolution
- Backward strided convolution
- Transposed Convolution (best name)

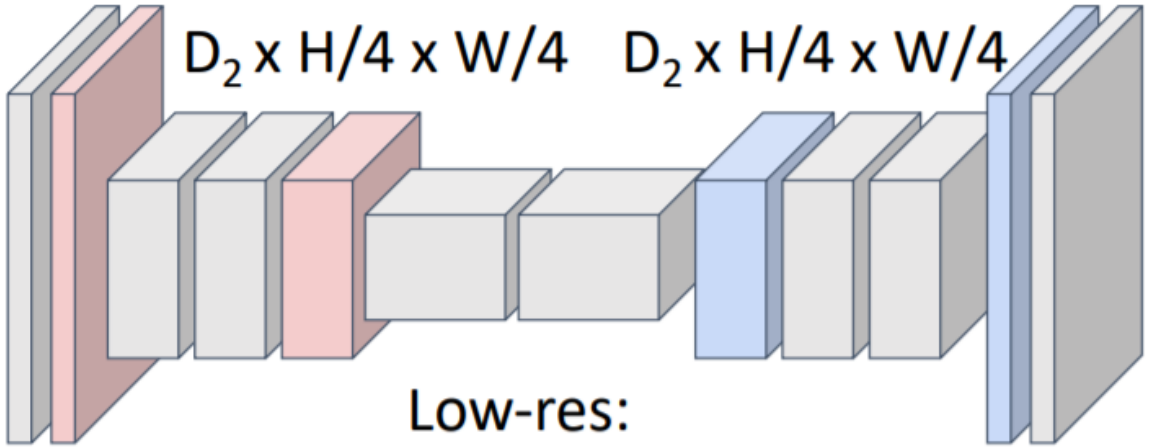
Semantic Segmentation: Fully Convolutional Network

Downsampling: Pooling, strided convolution

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

Upsampling: linterpolation, transposed conv

Input:
3 x H x W



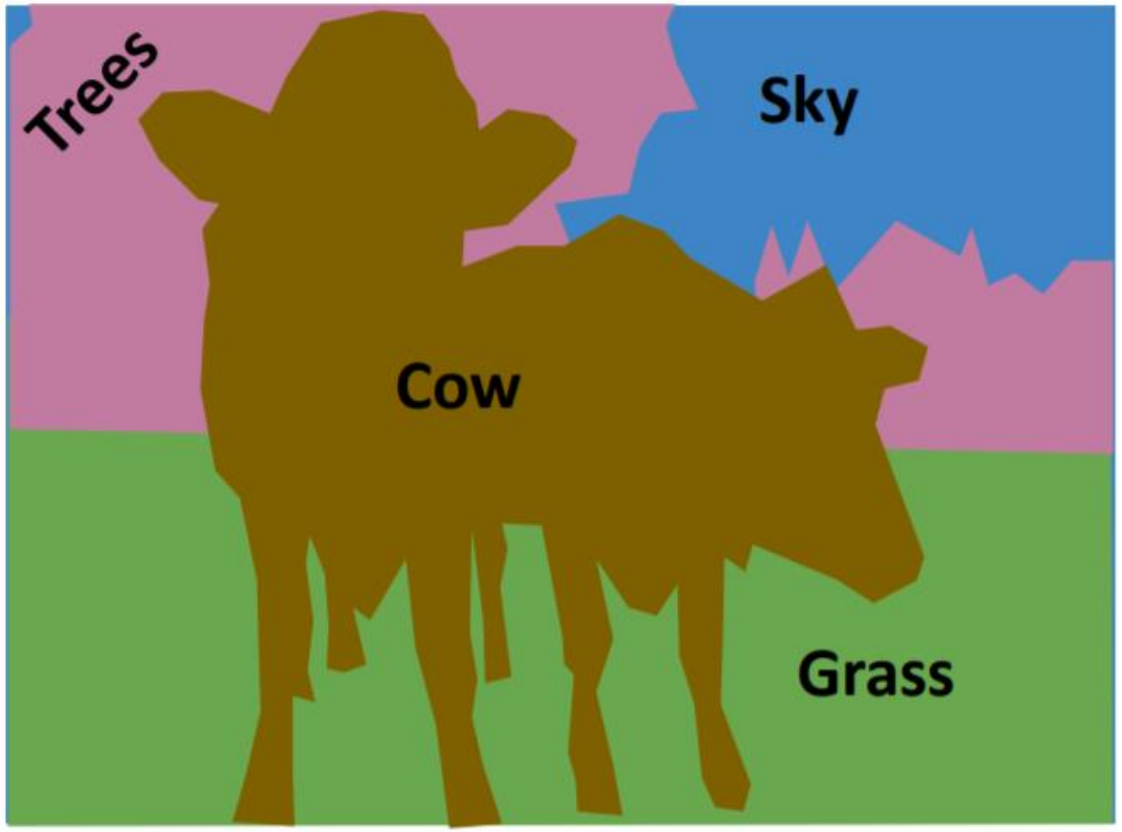
Predictions:
H x W

Loss function: Per-Pixel cross-entropy

Computer Vision Tasks

Object Detection: Detects individual object instances, but only gives box

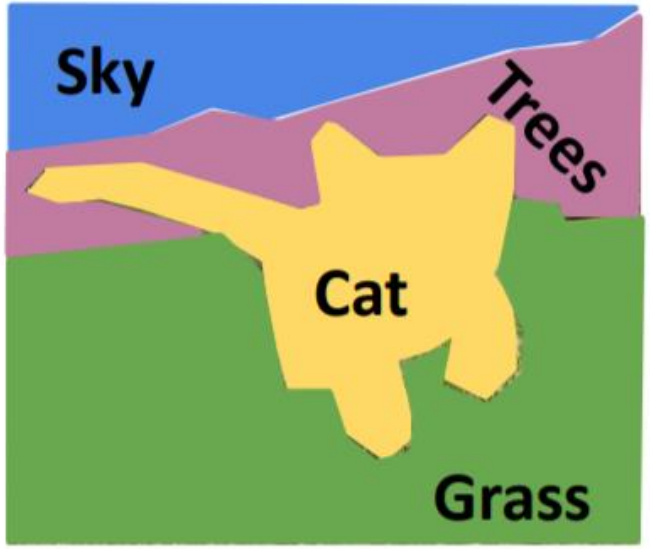
Semantic Segmentation: Gives per-pixel labels, but merges instances

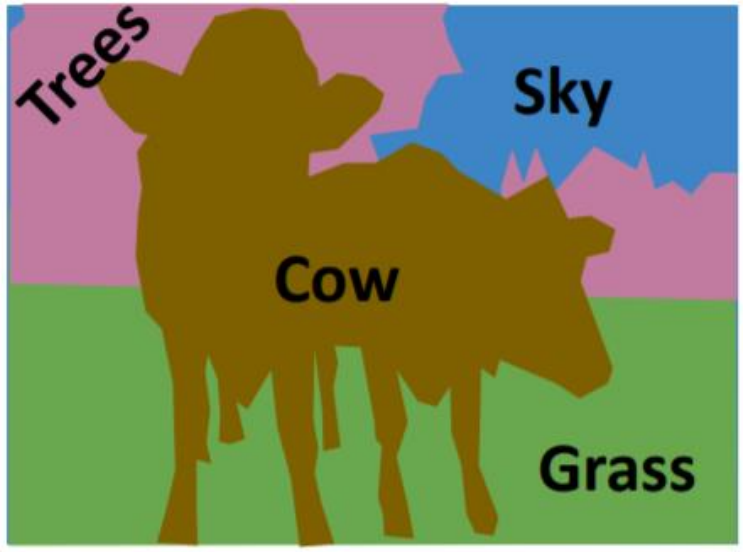


Things and Stuff

Things: Object categories that can be separated into object instances (e.g. cats, cars, person)

Stuff: Object categories that cannot be separated into instances (e.g. sky, grass, water, trees)

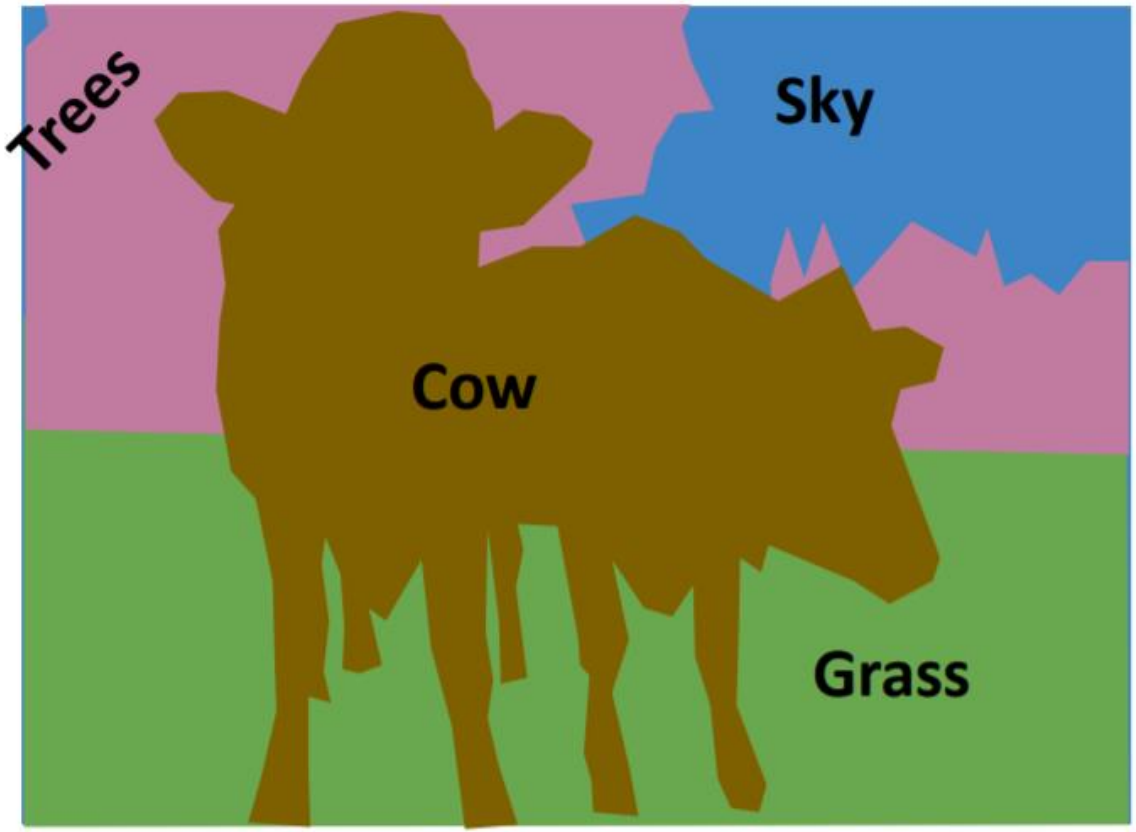




Computer Vision Tasks

Object Detection: Detects individual object instances, but only gives box (Only things!)

Semantic Segmentation: Gives per-pixel labels, but merges instances (Both things and stuff)

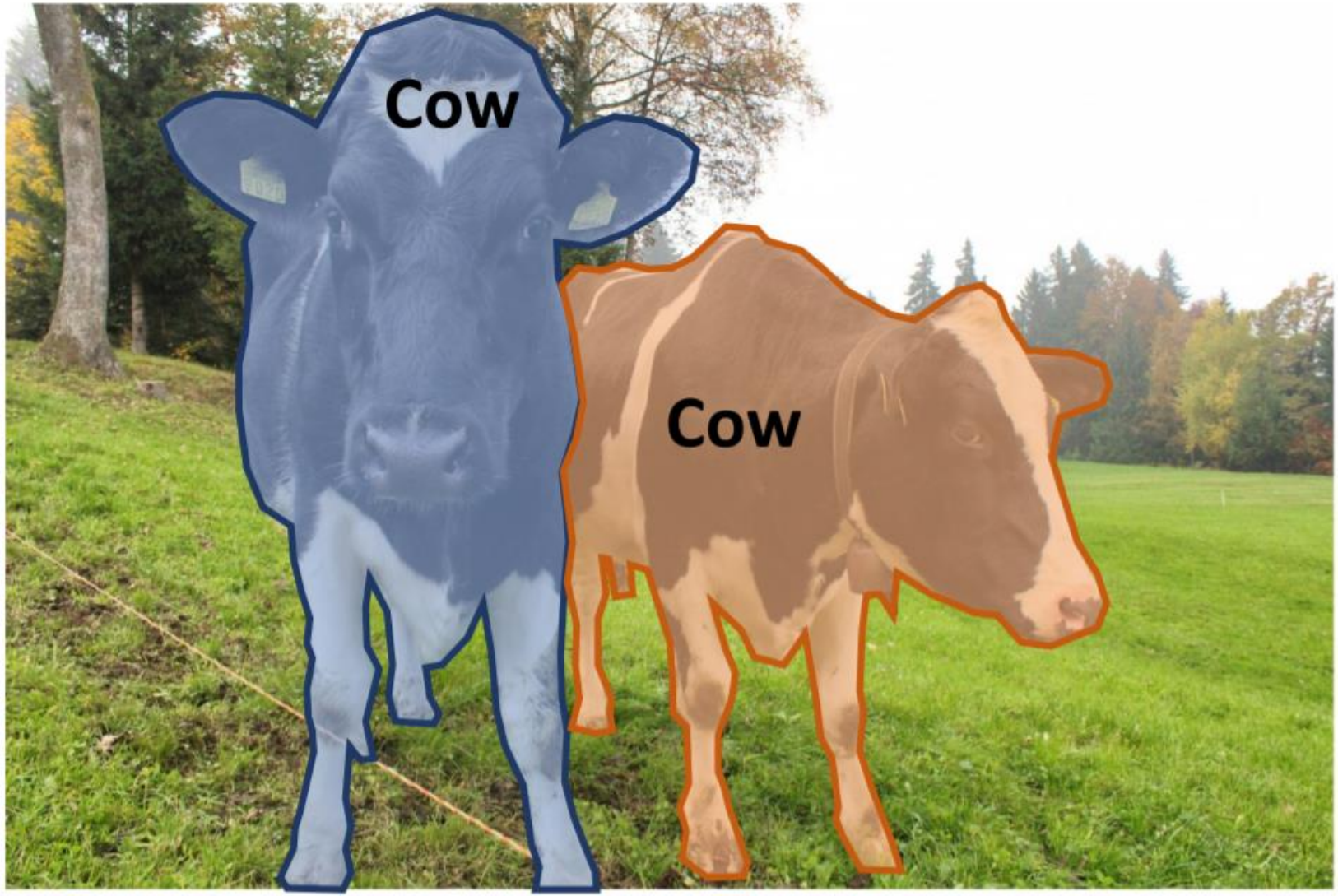


ie
SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

Instance Segmentation

Instance Segmentation

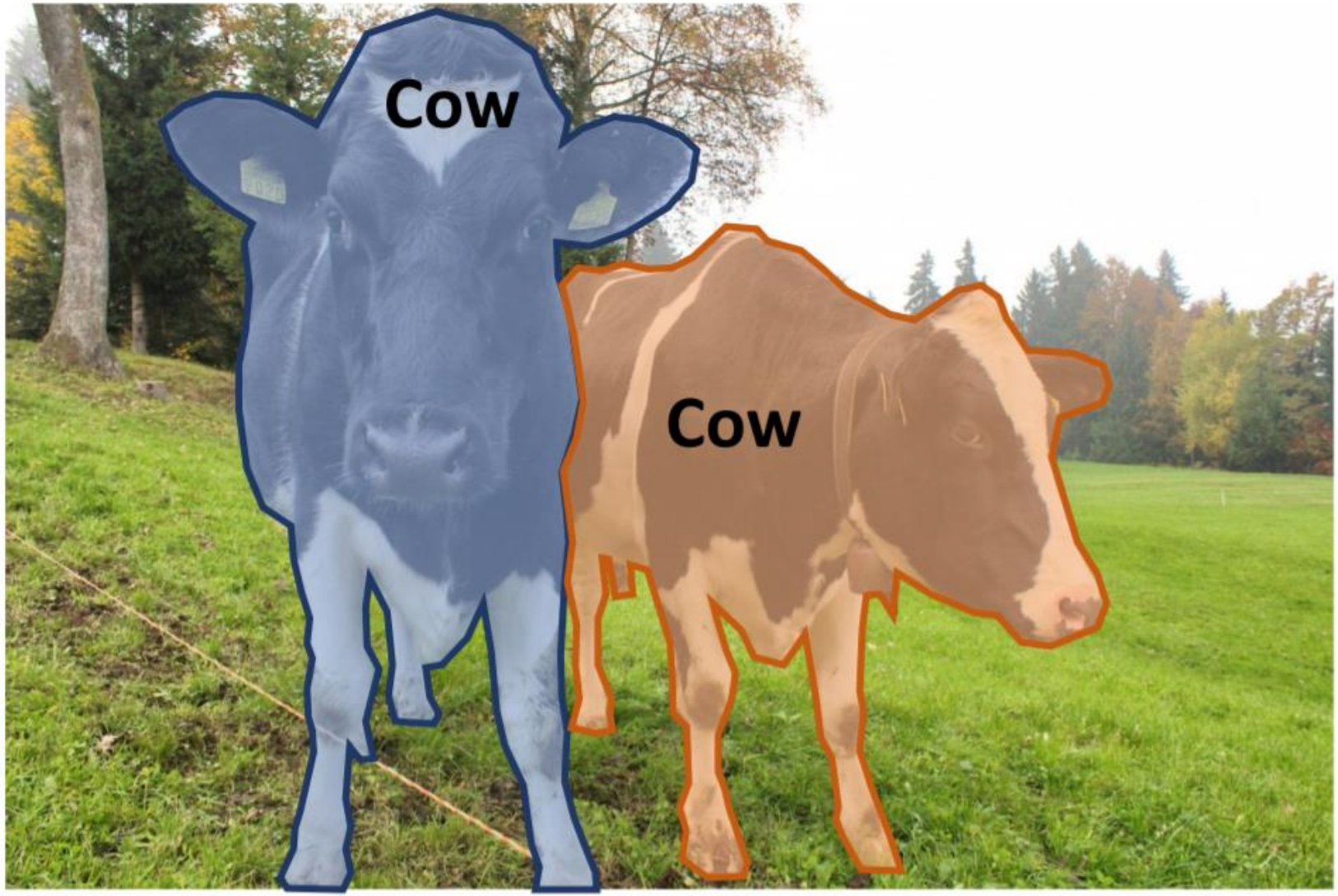
Instance Segmentation: Detect all objects in the image, and identify the pixels that belong to each object (Only things!)



ie
SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

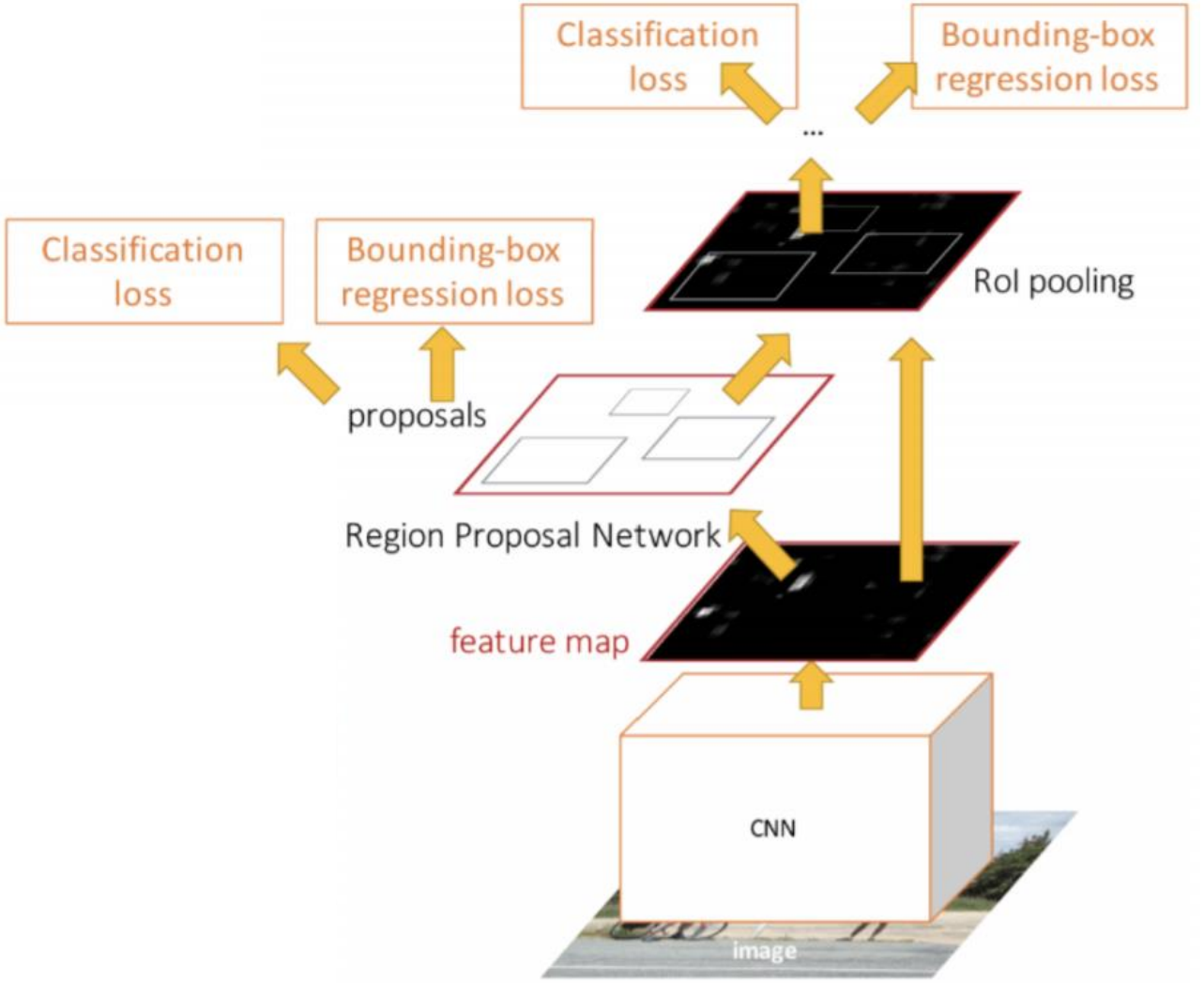
Instance Segmentation

Instance Segmentation:
Detect all objects in the image, and identify the pixels that belong to each object (Only things!)

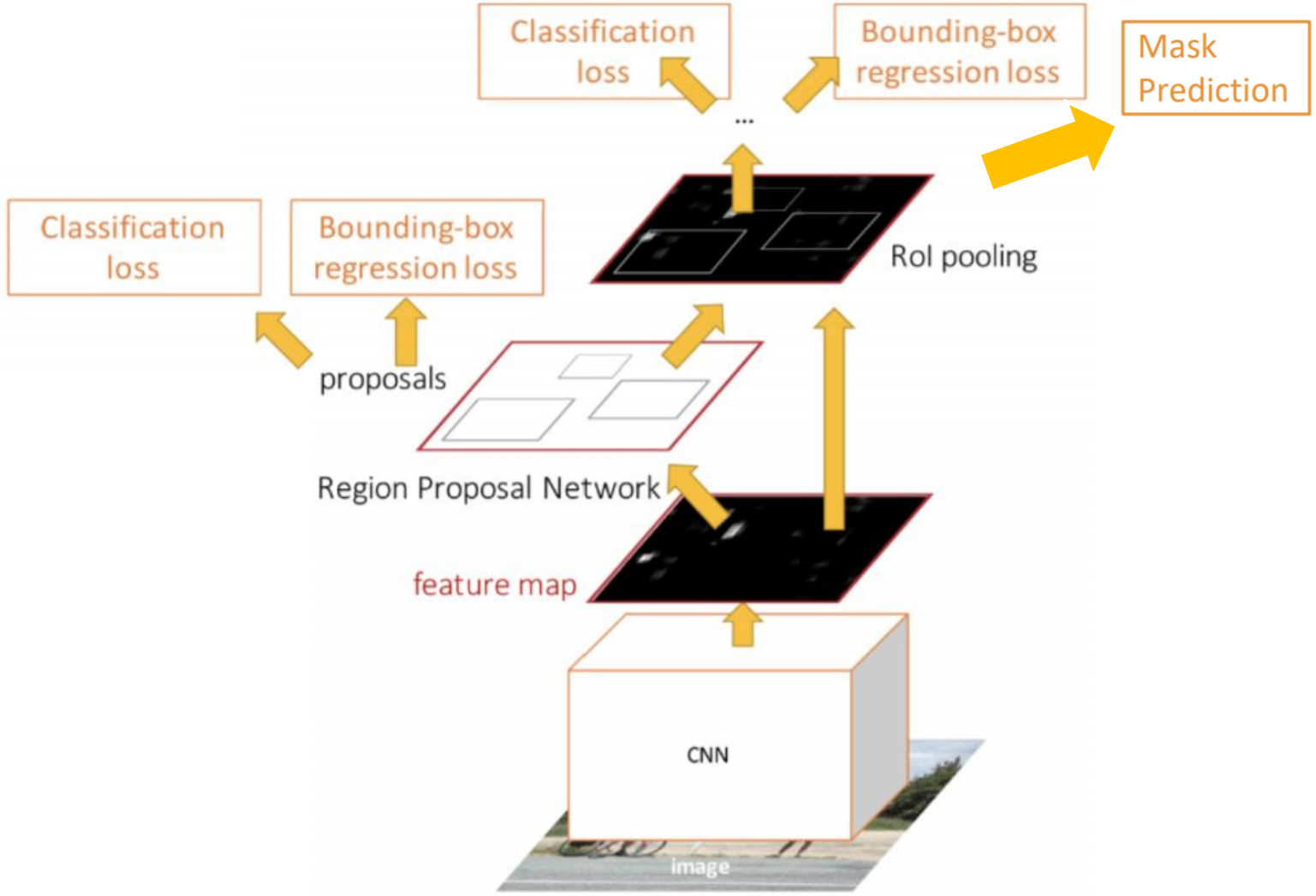


Approach: Perform object detection, then predict a segmentation mask for each object

Object Detection: Faster R-CNN

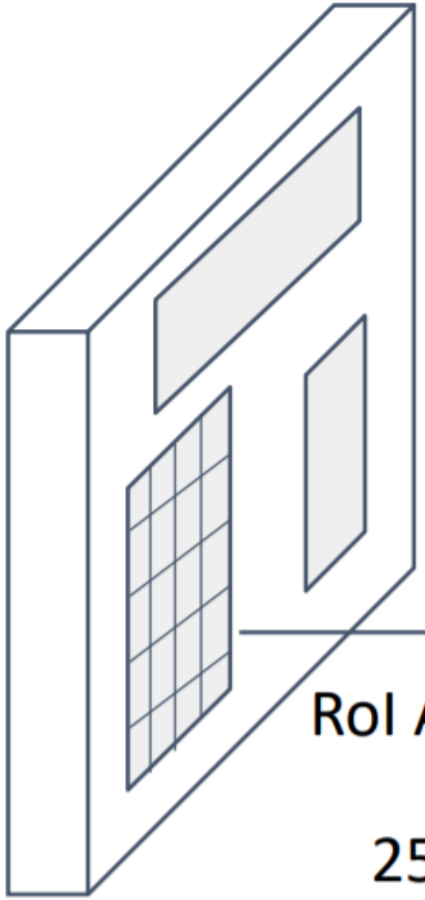


Instance Segmentation: Mask R-CNN

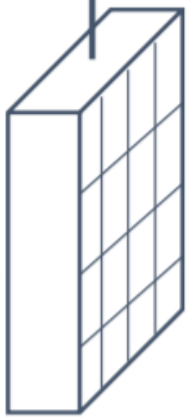


Mask R-CNN

CNN
+RPN

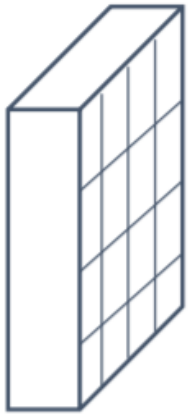


RoI Align



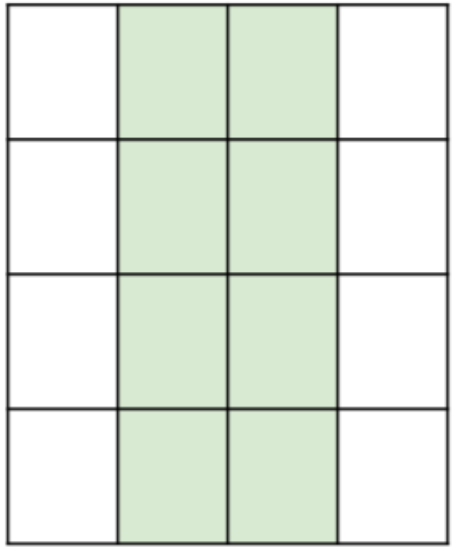
256 x 14 x 14

Conv



256 x 14 x 14

Conv

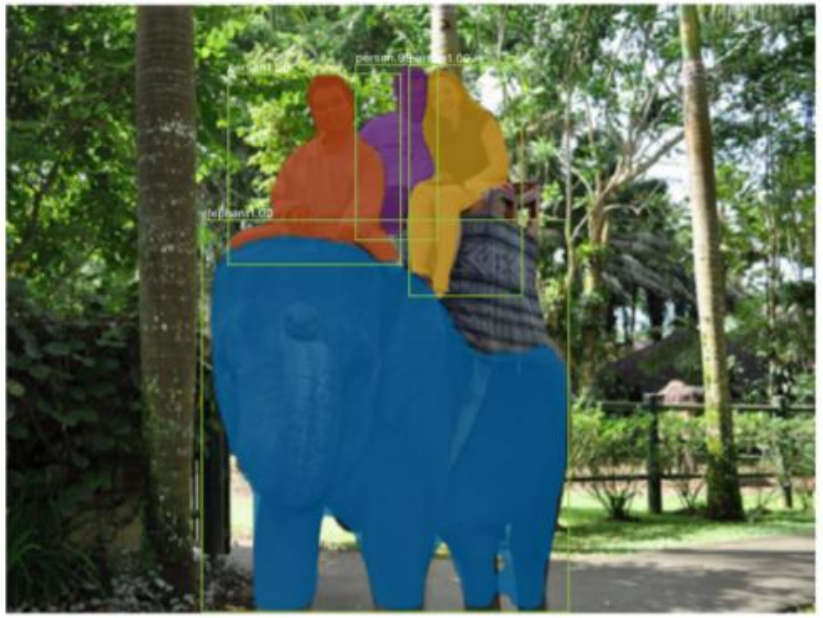


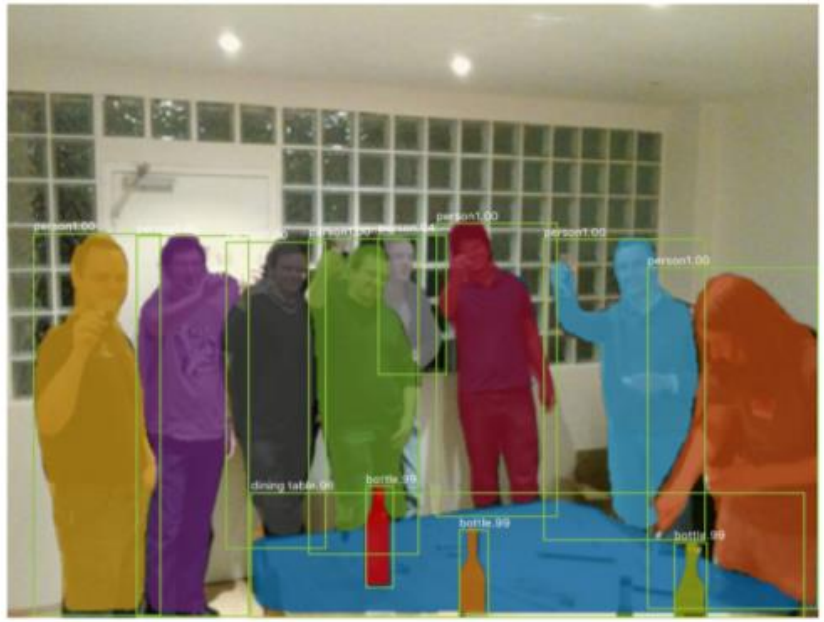
Classification Scores: C
Box coordinates (per class):
4 * C

Predict a mask for
each of C classes:
C x 28 x 28

SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

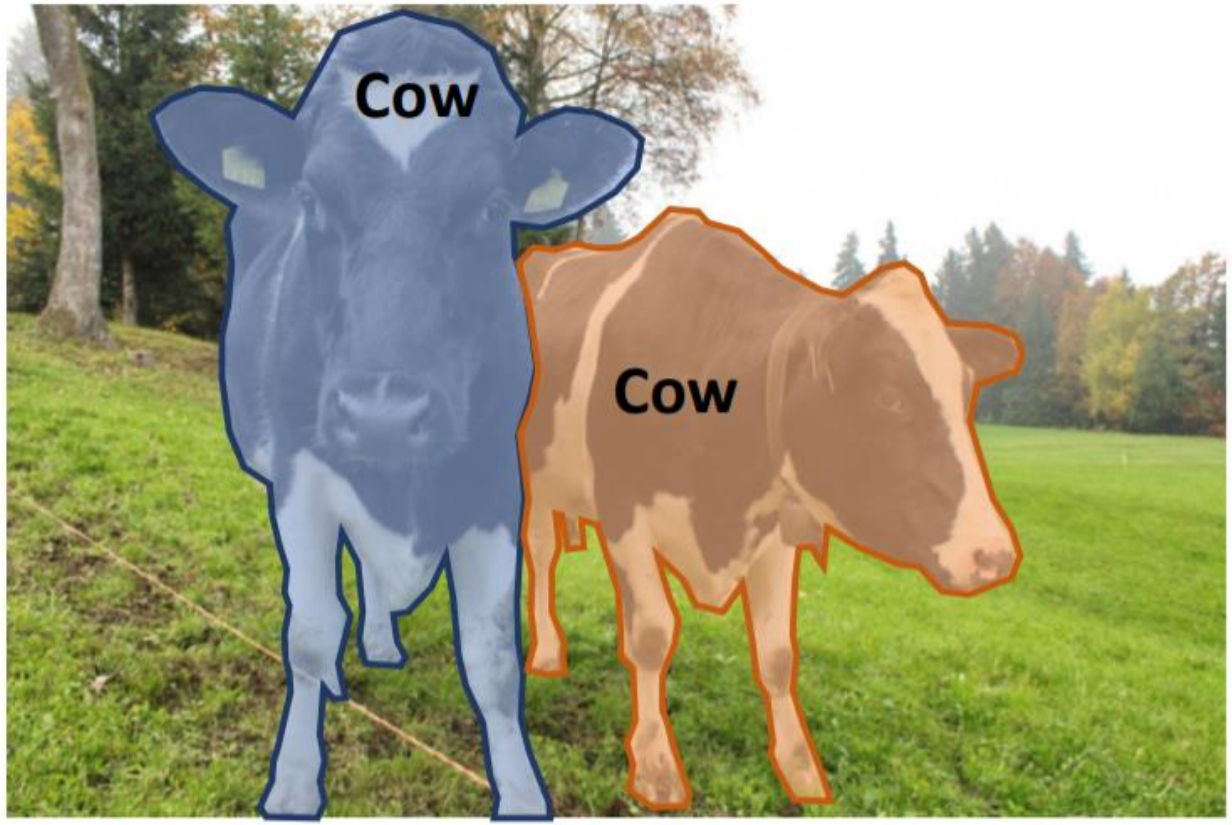
Mask R-CNN: Good Results



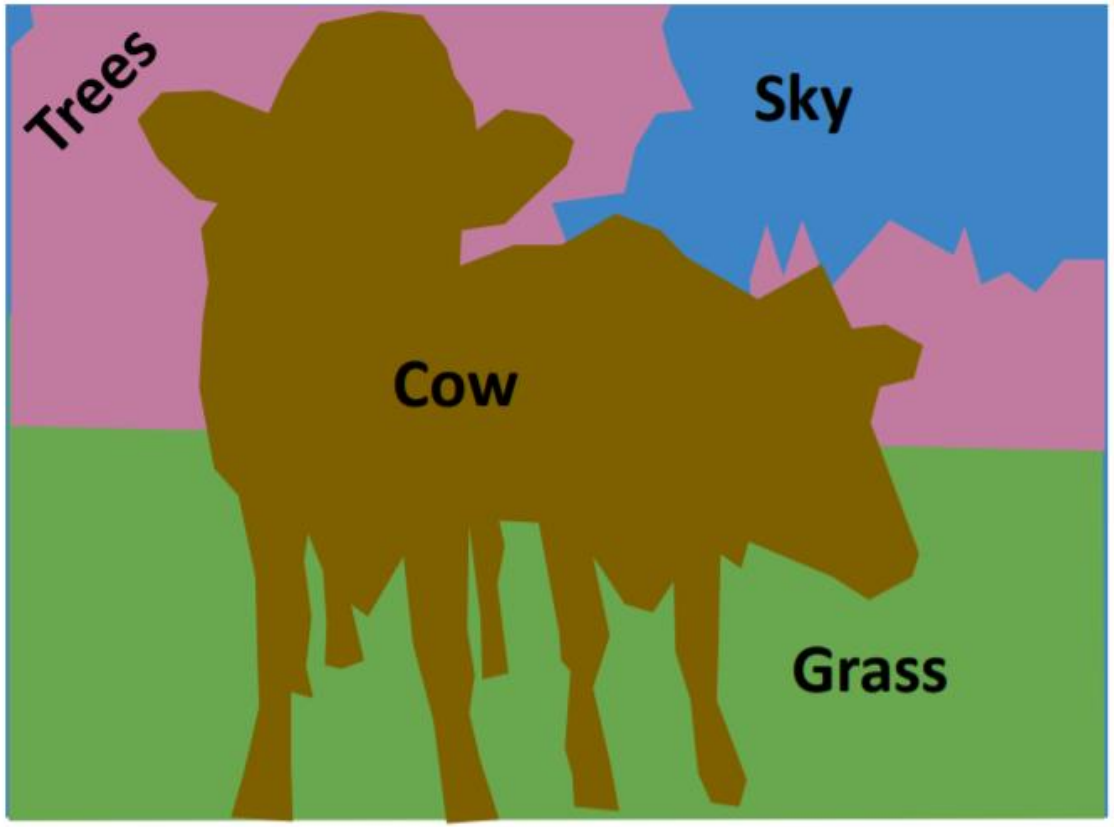


Beyond Instance Segmentation

Instance Segmentation: Separate object instances, but only things



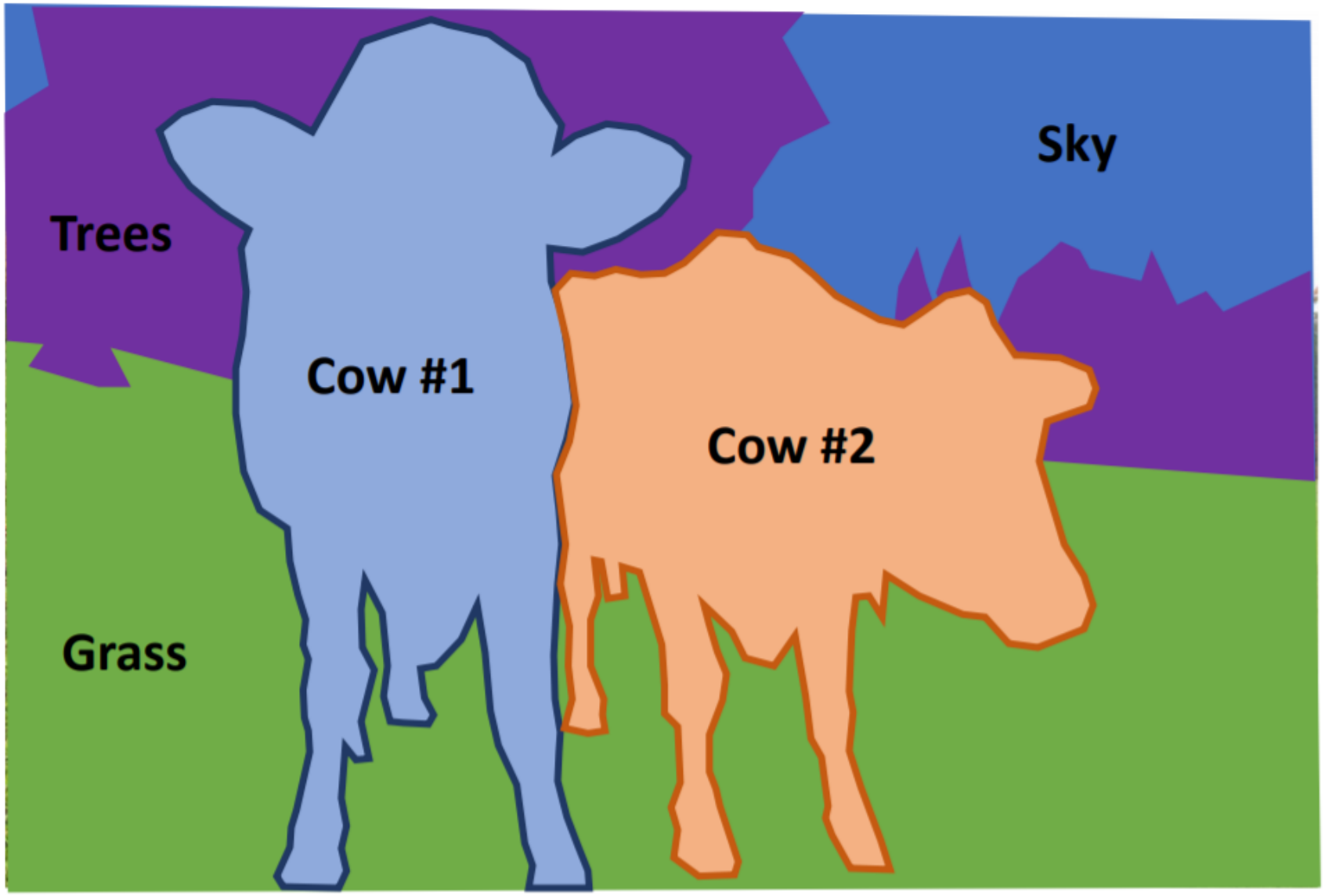
Semantic Segmentation: Identify both things and stuff, but doesn't separate instances



Beyond Instance Segmentation: Panoptic Segmentation

Label all pixels in the image (both things and stuff)

For “thing” categories also separate into instances



ie
SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

Beyond Instance Segmentation: Panoptic Segmentation

ie
SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

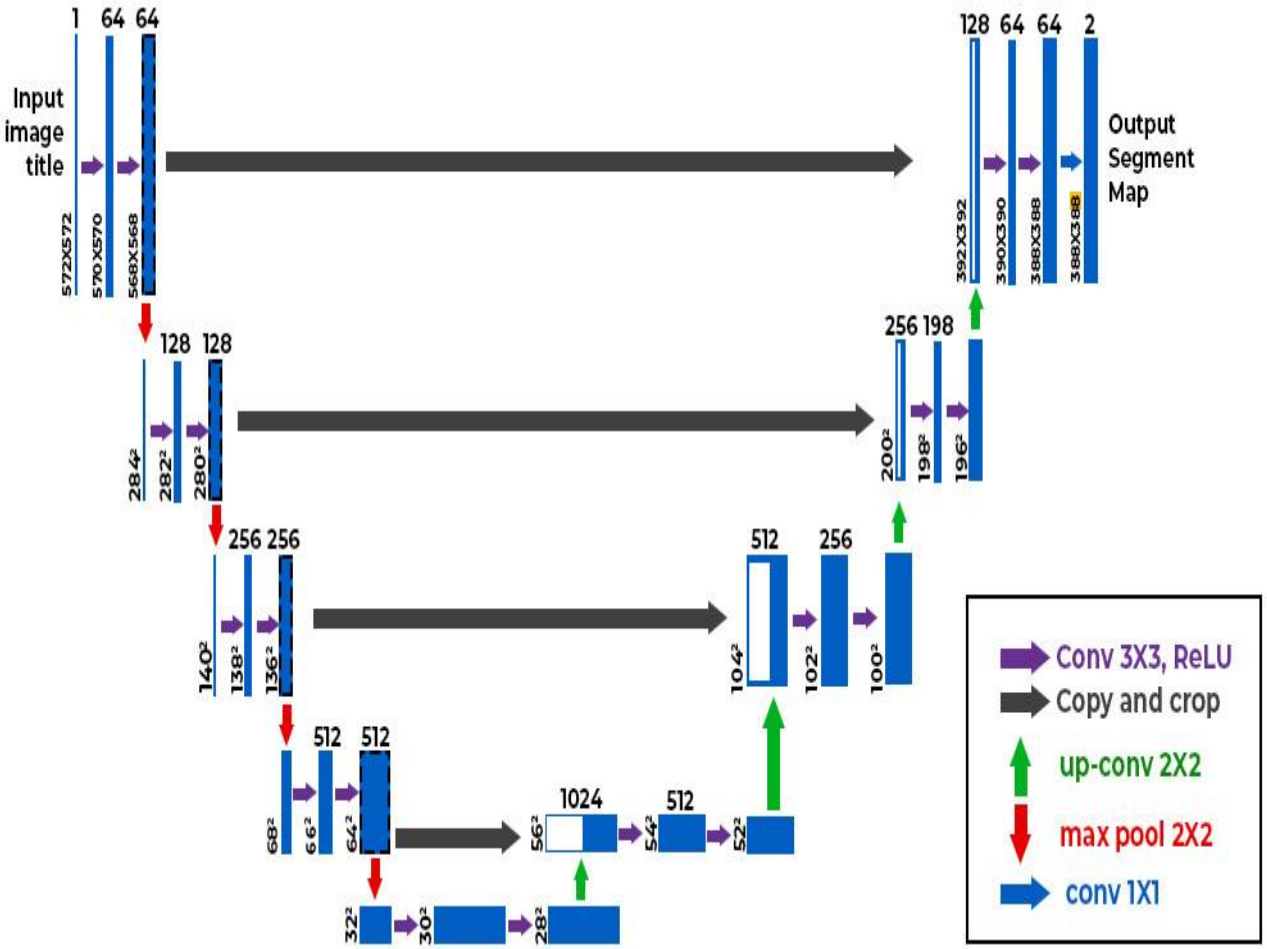
Semantic segmentation U-Net

U-Net Architecture

- **Encoder Path (Contracting):** This path is a typical convolutional network that repeatedly applies convolutions and max-pooling operations. It captures the context of the image by downsampling and extracting feature maps. Each step downsamples the feature maps while doubling the number of features.

- **Decoder Path (Expanding):** The decoder path is symmetric to the encoder and performs upsampling. It uses transposed convolutions to increase the resolution of the feature maps, effectively localizing where in the image a feature is present. In each step, the feature maps are upsampled, and the number of features is halved.

- **Skip Connections:** A key innovation in U-Net is the use of skip connections. These connections directly link corresponding layers in the encoder and decoder paths. They concatenate high-resolution feature maps from the encoder with the upsampled feature maps from the decoder. This allows the decoder to learn to assemble precise locations using the contextual information from the encoder, which is crucial for accurate segmentation.



ie
SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

Instance Segmentation SAM

Segment Anything Model (SAM): the first foundation model for promptable segmentation.

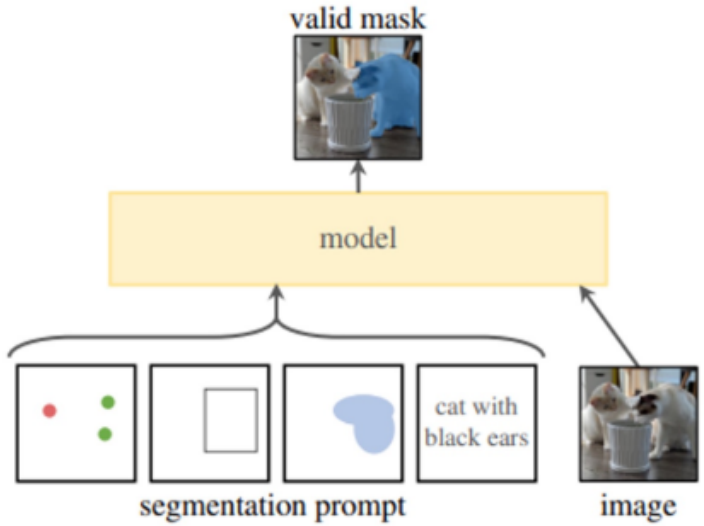
Prompt it with interactive points and boxes

Automatically segment everything in an image

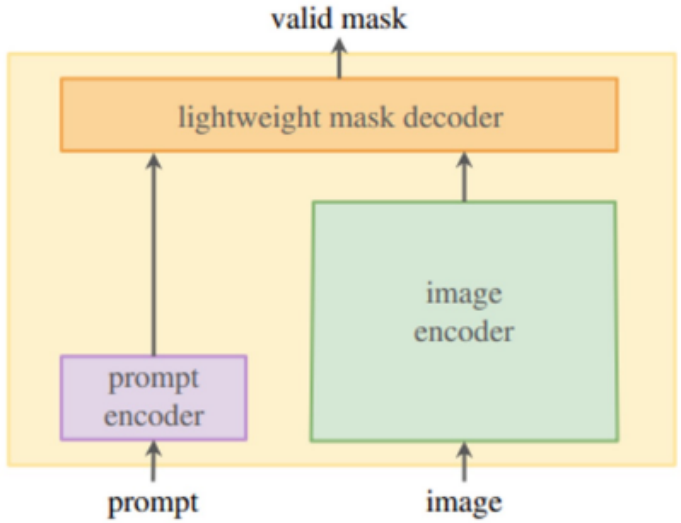
Generate multiple valid masks for ambiguous prompts

Try the demo: https://segment-anything.com/demo

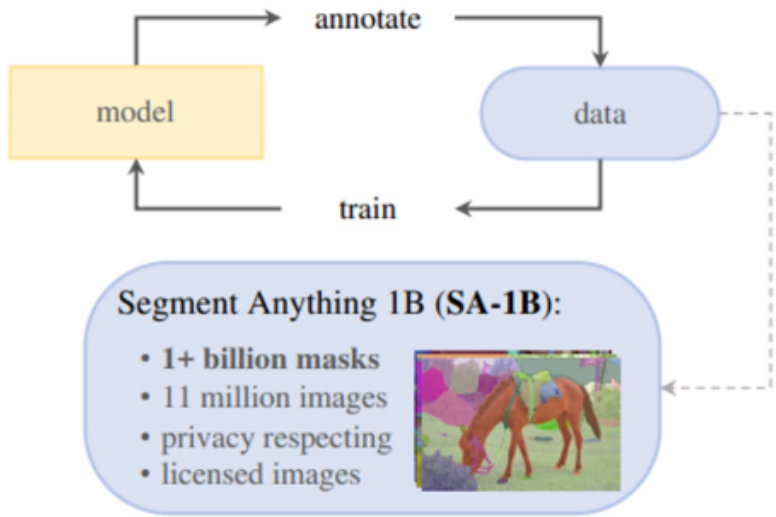
SAM is built with three interconnected components: A task, an model, and a data engine.



(a) **Task**: promptable segmentation

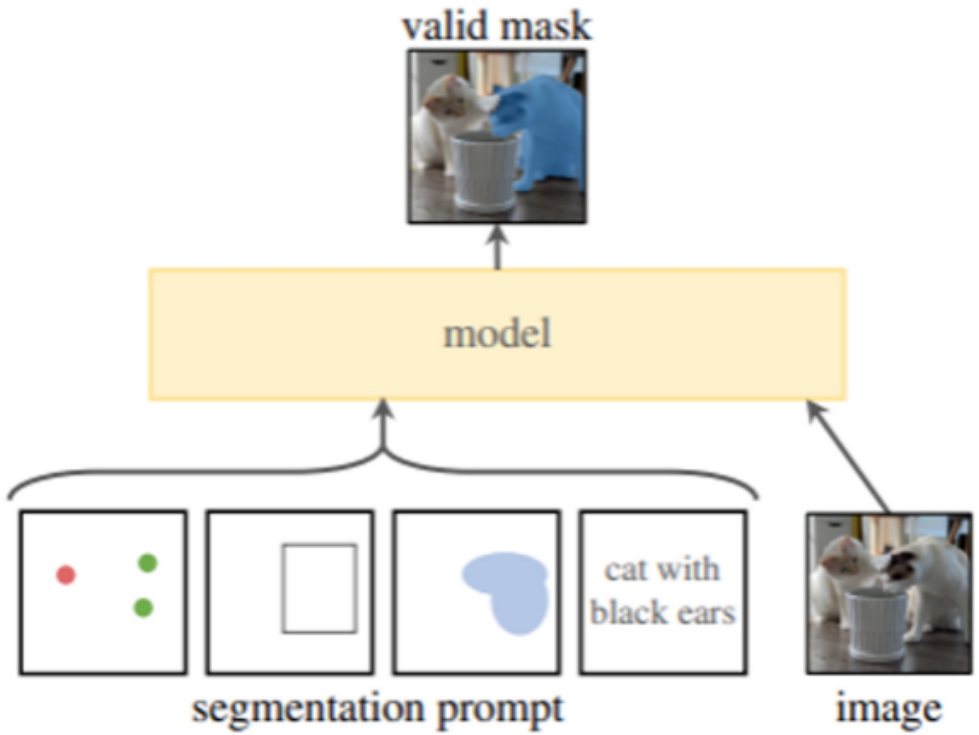


(b) **Model**: Segment Anything Model (**SAM**)



(c) **Data**: data engine (top) & dataset (bottom)

Task: Promptable Segmentation

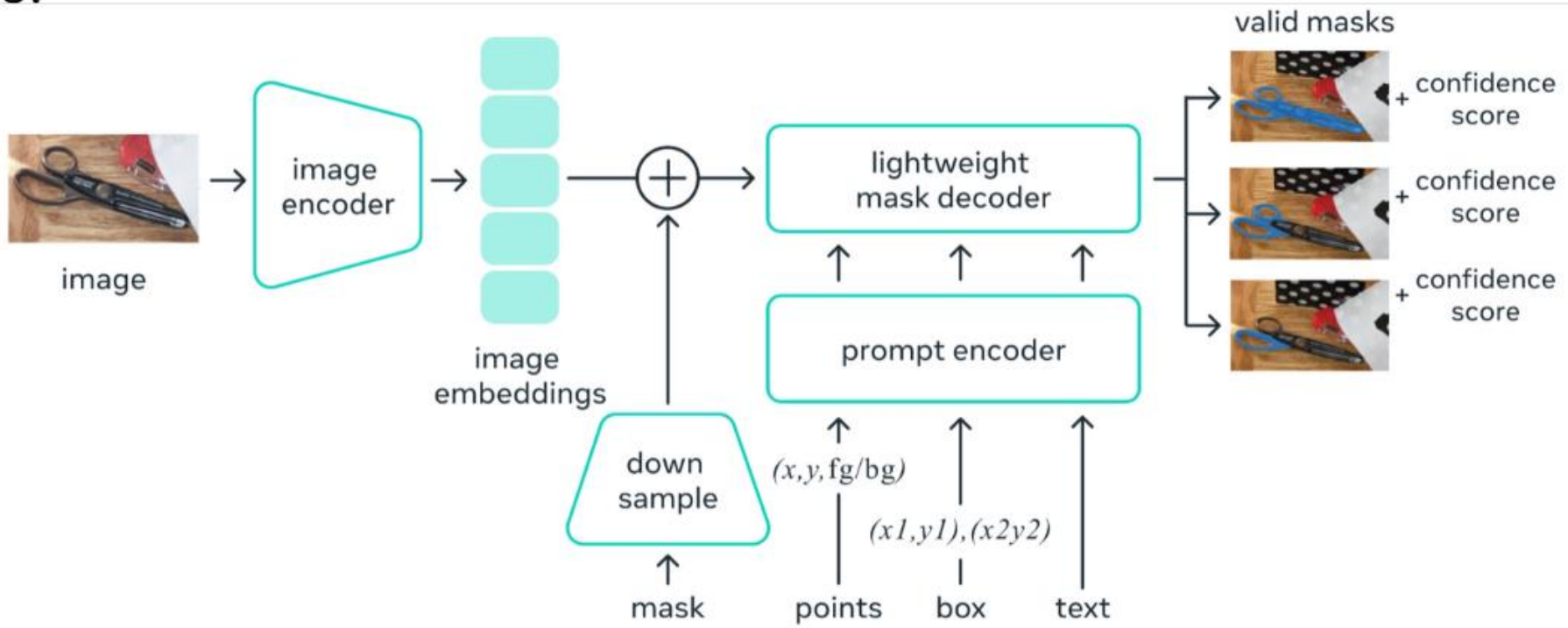


(a) **Task**: promptable segmentation

- SAM considers two sets of prompts: ***sparse*** (clicks, boxes, text) and ***dense*** (masks).
- SAM's promptable design enables flexible integration with other systems (i.e., used as ***component*** in larger systems).

Model: Segment Anything Model (SAM)

- A heavyweight ***image encoder*** outputs an image embedding.
- A lightweight ***prompt encoder*** efficiently queries the image embedding.
- A lightweight ***mask decoder*** produces object masks and confidence scores.



Grounded-Segment-Anything

Text Prompt: Bench

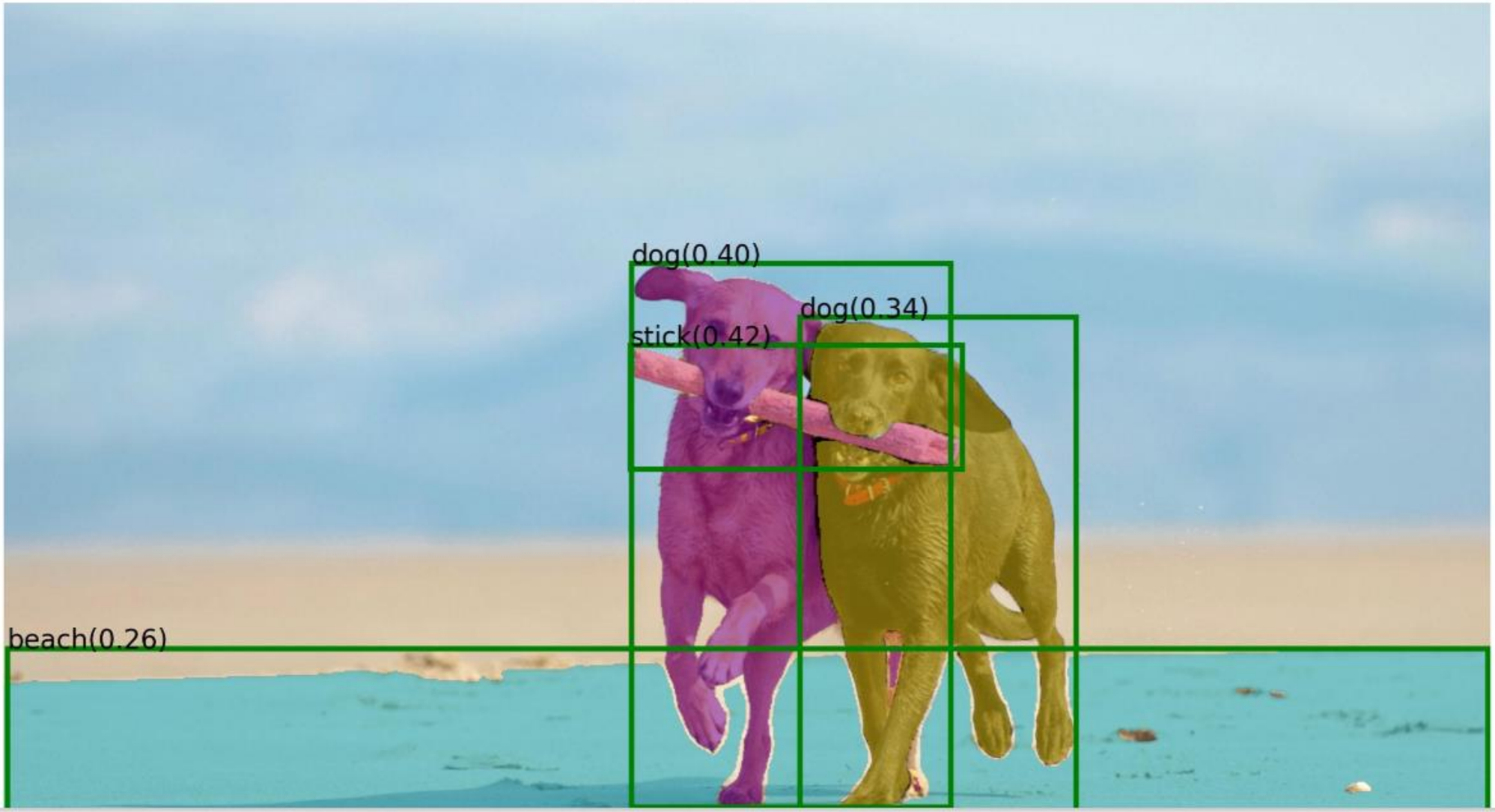
Grounded-SAM Output

Stable-Diffusion Inpainting
A Sofa, high quality, detailed

BLIP + Grounded-SAM: Automatic Label System!

Using BLIP to generate caption, extract tags and using Grounded-SAM for box and mask generating. Here's the demo output:

there are two dogs playing with a stick on the beach



ie
SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

Instance Segmentation FastSAM

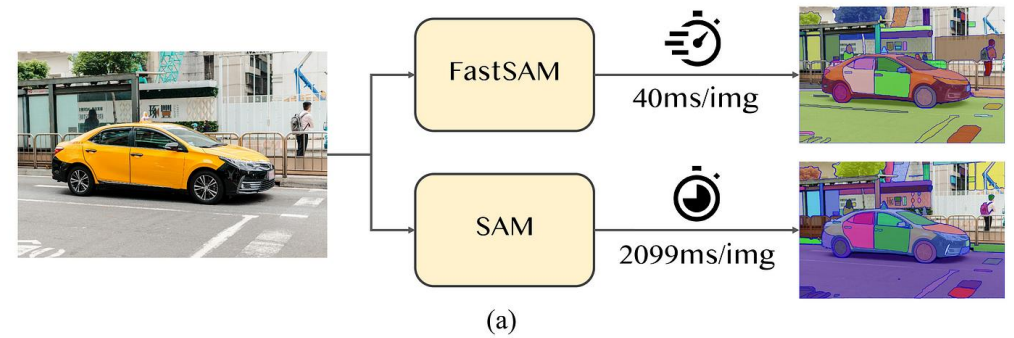
FastSAM

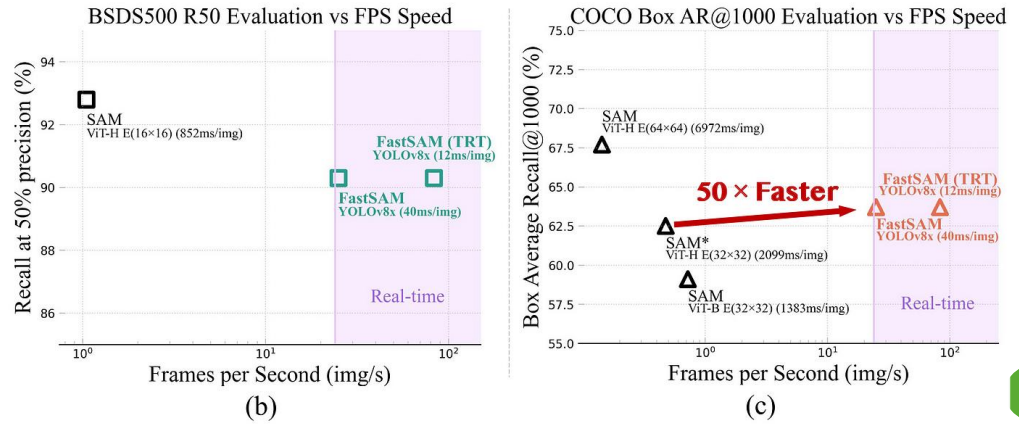
FastSAM significantly reduces computational demands while maintaining competitive performance, making it a practical choice for a variety of vision tasks. trained using 2% of the data in the Segment Anything Model SA-1B dataset

FastSAM employs a two-stage procedure:

1.The segmentation masks are generated by a pre-trained YOLOv8 instance segmentation model.

2.A text prompt can be supplied as an option to return masks linked to the prompt. CLIP is used for this. CLIP is applied on each mask, and photos that have a high similarity to th



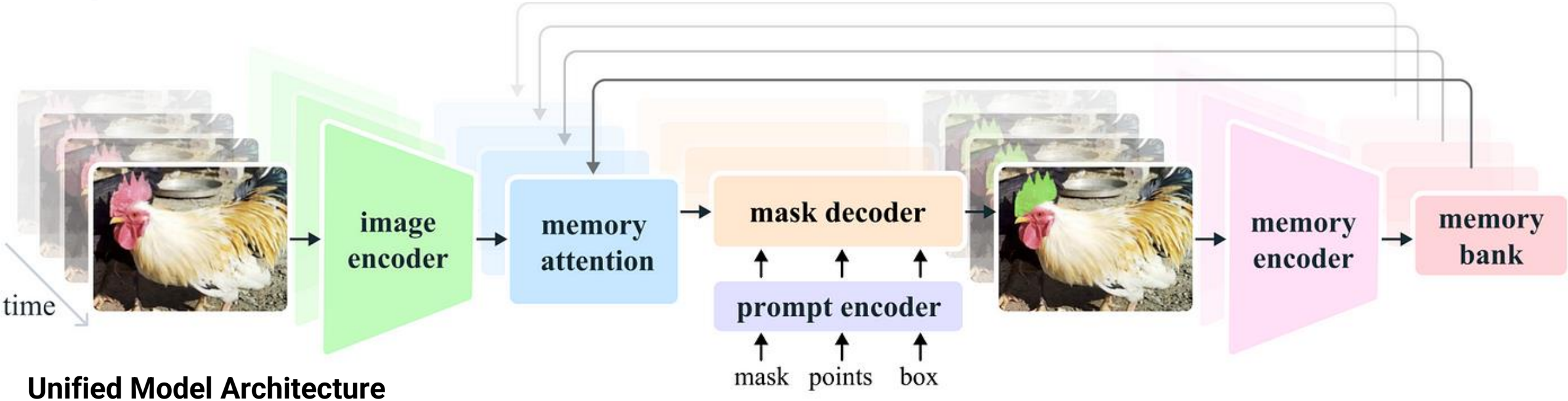


HuggingFace Space https://huggingface.co/spaces/An-619/FastSAM

ie
SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

Instance Segmentation SAM2

SAM2 Technical details



Unified Model Architecture

SAM 2 combines the capabilities of image and video segmentation in a single model. This unification simplifies deployment and allows for consistent performance across different media types. It leverages a flexible prompt-based interface, enabling users to specify objects of interest through various prompt types, such as points, bounding boxes, or masks.

Real-Time Performance

The model achieves real-time inference speeds, processing approximately 44 frames per second. This makes SAM 2 suitable for applications requiring immediate feedback, such as video editing and augmented reality.

Zero-Shot Generalization

SAM 2 can segment objects it has never encountered before, demonstrating strong zero-shot generalization. This is particularly useful in diverse or evolving visual domains where pre-defined categories may not cover all possible objects.

SAM2 Technical details

Performance and Technical Details

SAM 2 sets a new benchmark in the field, outperforming previous models on various metrics:

Metric	SAM 2	Previous SOTA
Interactive Video Segmentation	**Best**	-
Human Interactions Required	**3x fewer**	Baseline
Image Segmentation Accuracy	**Improved**	SAM
Inference Speed	**6x faster**	SAM

Memory Mechanism and Occlusion Handling

The memory mechanism allows SAM 2 to handle temporal dependencies and occlusions in video data. As objects move and interact, SAM 2 records their features in a memory bank. When an object becomes occluded, the model can rely on this memory to predict its position and appearance when it reappears. The occlusion head specifically handles scenarios where objects are not visible, predicting the likelihood of an object being occluded.

Multi-Mask Ambiguity Resolution

In situations with ambiguity (e.g., overlapping objects), SAM 2 can generate multiple mask predictions. This feature is crucial for accurately representing complex scenes where a single mask might not sufficiently describe the scene's nuances.

ie
SCHOOL OF
HUMAN SCIENCES
& TECHNOLOGY

Instance Segmentation yolo11

Chapter Summary

- We visited Semantic segmentation and architecture
- We explored Instance Segmentation and architecture
- We saw an example of semantic segmentation architecture: U Net
- We saw an example of instance segmentation architecture: Mask R CNN
- we looked at Python implementation of U Net and Mask R CNN
- We looked at SAM/SAM2 as foundational model for segmentation
- We saw FastSAM as a real time alternative to SAM
- We saw yolo for instance segmentation