African Institute for Mathematical Sciences — Cameroon
Structured Master's, Data Science Stream · Computer Vision

Every Pixel Counts

Week 2 · Day 4: Semantic Segmentation, Instance Segmentation and Segment Anything

Yusuf Brima, Ph.D.

Introduction to Computer Vision · AIMS Cameroon, 2026–2027

Today

1. **Why pixels?** Semantic, instance and panoptic segmentation
2. **Measuring masks**: IoU, Dice, and the pixel-accuracy trap
3. **Fully convolutional networks** and upsampling
4. **U-Net**: encoder, decoder, and skip connections, trained live
5. **DeepLab**: atrous convolution and context
6. **Instance segmentation**: Mask R-CNN
7. **Segment Anything**: promptable, general-purpose masks

By the end of today you should be able to

- distinguish semantic, instance and panoptic segmentation;
- compute IoU and Dice, and explain why pixel accuracy misleads;
- explain how networks go from small feature maps back to full resolution;
- explain why U-Net's skip connections matter;
- describe how Mask R-CNN and SAM produce masks.

Where we are

The task ladder from Week 1, Day 1

- Classification: *what?* (Days 1–2)
- Detection: *what, roughly where?* (Day 3)
- **Segmentation**: *exactly which pixels?* (today)

Why boxes are not enough

- A tumour's *area* and *border* matter, not its bounding box.
- A flooded region, a field, a road are not box-shaped.
- Measuring lesion area on a leaf needs the pixels (Week 1: 6% of maize pixels).



Bdx, CC BY-SA 4.0

Part 1

Kinds of segmentation, and how to measure them

Semantic, instance, panoptic

Semantic

- One class label per pixel.
- All cars share one colour; it cannot count them.
- Roads, sky, vegetation, tumour tissue.

Instance

- One mask per object.
- Car #1, car #2, person #3: counting and tracking.
- Ignores “stuff” like sky and road.

Panoptic

- Both: every pixel gets a class, and pixels of countable *things* also get an instance id.
- Used in self-driving and mapping.



Photo: Bdx, CC BY-SA 4.0

Datasets for segmentation

Dataset	Classes	Images	Notes
PASCAL VOC	20 + background	about 3 000 with masks	the original benchmark; today's pre-trained models use its labels
Cityscapes	19 evaluated	5 000 finely labelled	street scenes from 50 European cities
ADE20K	150	about 20 000	indoor and outdoor scenes; "stuff" and "things"
COCO	80 things + 53 stuff	118 000	instance and panoptic masks
Medical (e.g. BraTS)	a few tissue types	hundreds of 3D scans	expert labels; 3D volumes

Intuition

Notice the sizes: masks are expensive, so segmentation datasets are much smaller than classification ones. Transfer learning (Day 2) matters even more here.

Measuring a mask: IoU and Dice

Two overlap scores

$$\text{IoU} = \frac{|P \cap G|}{|P \cup G|}, \qquad \text{Dice} = \frac{2\,|P \cap G|}{|P| + |G|}$$

P: predicted pixels; G: ground-truth pixels.

- Both are 1 for a perfect mask, 0 for no overlap.
- Dice $= 2\,\text{IoU}/(1 + \text{IoU})$: always a little higher. Medicine prefers Dice; computer vision papers prefer IoU.
- **mIoU**: IoU averaged over classes.

A small lesion

Ground truth: 100 pixels. Prediction: 80 pixels, 60 of them correct.

Intersection 60; union $100 + 80 - 60 = 120$.
IoU $= 60/120 = 0.50$.
Dice $= 120/180 = 0.67$.

The pixel-accuracy trap

- **Pixel accuracy**: share of pixels labelled correctly. It sounds natural.
- In Penn-Fudan, people cover only 13.1% of the pixels on average.
- A "model" that predicts **background everywhere** has pixel accuracy 86.9%, and finds no one: person IoU $= 0$.

Intuition

When one class dominates, accuracy rewards ignoring the rare class. In medical images the lesion is often under 1% of pixels: a blank prediction scores over 99%.

Practitioner's Note

Always report per-class IoU or Dice for the classes you care about. For training on rare classes, use a Dice loss or class weights, or both.

Scoring panoptic segmentation

Panoptic quality (Kirillov et al., 2019)

$$\mathrm{PQ} = \frac{\sum_{\text{matched}} \mathrm{IoU}}{|TP| + \frac{1}{2}|FP| + \frac{1}{2}|FN|}$$

Segments match if IoU > 0.5.

- Numerator: how good the matched masks are.
- Denominator: penalises missed and invented segments.

Four people in an image

The model outputs 4 person segments: 3 match (IoUs 0.9, 0.8, 0.7), 1 is spurious; 1 real person is missed.

$$\mathrm{PQ} = \frac{0.9 + 0.8 + 0.7}{3 + 0.5 + 0.5} = \frac{2.4}{4} = 0.60.$$

Intuition

PQ = (segmentation quality: average IoU of matches) $\times$ (recognition quality: an F1 score of matches). One number that rewards both finding objects and outlining them well.

Part 2

From feature maps back to pixels

The problem: CNNs shrink the image

- A ResNet turns a 224×224 image into a 7×7 feature map: 32 times smaller in each direction.
- Great for "what" (Day 2), useless for "exactly which pixel".
- We need the network to output a **full-resolution** map of class scores: $H \times W \times K$.

Fully convolutional network (FCN, 2015)

Replace the fully connected head with 1×1 convolutions (class scores at every location), then **upsample** back to the input size. Train with cross-entropy at every pixel.

Intuition

An FCN classifies every pixel at once, sharing the computation, just as Fast R-CNN shared it across boxes (Day 3).

A fully connected layer is a convolution in disguise

Turning a classifier into a pixel classifier

A classifier's last layers: $7 \times 7 \times 512$ features $\rightarrow$ FC to 1000 classes.

The same weights, rearranged, form a 7×7 convolution with 1000 output channels.

On a 7×7 map it gives one output (as before). On a larger image, say a 14×14 map, it slides and gives an 8×8 map of class scores, one per position.

Intuition

This was the key insight of FCN (Long et al., 2015): any trained classifier can be "convolutionalised" to produce a coarse heat map per class, for free. Then train it to produce *full-resolution* maps.

Practitioner's Note

Global-average-pooled networks (ResNet) give the same effect: drop the pooling and apply the FC layer at every position as a 1×1 convolution.

Upsampling: three ways to make a map bigger



- **Nearest**: copy each value into a 2×2 block.
- **Bilinear**: smooth interpolation (Week 1, Day 2).
- **Transposed convolution**: each input value *stamps* a learned kernel, scaled by the value, onto the output; overlaps add up.

The transposed conv above

Kernel $\begin{bmatrix} 1 & 0.5 \\ 0.5 & 0.25 \end{bmatrix}$, stride 2. The input 4 stamps $\begin{bmatrix} 4 & 2 \\ 2 & 1 \end{bmatrix}$ in the bottom-right corner. The kernel is learned, so the network decides *how* to upsample.

Transposed convolution: sizes and artefacts

Output size of a transposed convolution

$$O = (N-1)\,s - 2p + K$$

The inverse of Day 1's formula: what a stride-s convolution shrinks, this grows.

Doubling a 7×7 map

$N = 7$, $s = 2$, $p = 1$, $K = 4$: $O = 6 \cdot 2 - 2 + 4 = 14$.
$K = 2$, $s = 2$, $p = 0$ (as in our U-Net): $O = 6 \cdot 2 + 2 = 14$.

Checkerboard artefacts

- If the kernel size is not a multiple of the stride (e.g. $K = 3$, $s = 2$), some output pixels receive more stamps than others: a faint grid pattern.
- Common fix: upsample with nearest or bilinear interpolation, then apply a normal convolution ("resize-conv").

Intuition

Each input value stamps the kernel onto the output. Uneven overlap of stamps means uneven brightness, like overlapping rubber stamps.

But upsampling cannot invent lost detail

- At 7×7, each feature-map cell covers a 32×32 patch: the exact outline of a person's arm is gone.
- Upsample it 32 times and the mask is blobby: right *object*, wrong *border*.
- FCN's fix: also use the earlier, finer feature maps (stride 8 and 16) and combine them.

Intuition

Deep layers know *what*; shallow layers know *where*. Good segmentation needs both: the semantic understanding from deep layers and the precise edges from early layers.

The design question for every segmentation network

How do we combine **coarse, meaningful** features with **fine, precise** ones? U-Net, DeepLab and Mask R-CNN are three answers.

Encoder–decoder designs

- **Encoder**: a classification backbone (VGG, ResNet) that shrinks the image into rich features.
- **Decoder**: grows the maps back to full resolution.
- **SegNet** (2015): remembers *where* each max-pooling maximum came from, and puts values back in those positions when upsampling.
- **U-Net** (next): copies whole feature maps across instead.

Intuition

The encoder is Day 2's backbone, pre-trained on ImageNet if possible. The decoder is new and must be learned. Most of the know-how of segmentation is in how the two are connected.

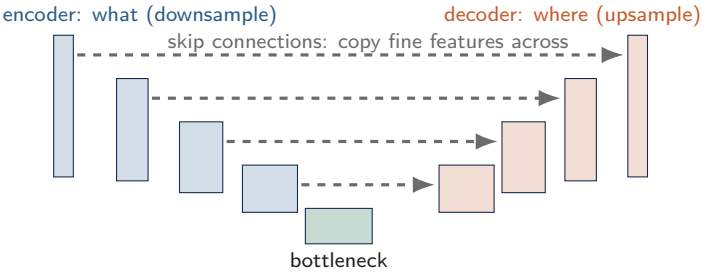
Practitioner's Note

Libraries such as `segmentation_models_pytorch` let you combine any pre-trained encoder (ResNet, EfficientNet, ConvNeXt) with a U-Net or DeepLab decoder in one line.

Part 3

U-Net: encoder, decoder, skips

U-Net (2015)



- **Encoder**: conv blocks + pooling; resolution halves, channels double.
- **Decoder**: transposed convs; resolution doubles.
- **Skips**: at each level, the encoder's features are concatenated into the decoder.

Intuition

The decoder knows *what* is there (from the bottleneck) and gets *where exactly* from the skip at each scale. Designed for microscopy with only 30 training images; still the default in medical imaging.

Our tiny U-Net, shape by shape

Input $128 \times 128 \times 3$

Encoder

block 1: $128 \times 128 \times 16$ $\rightarrow$ pool
block 2: $64 \times 64 \times 32$ $\rightarrow$ pool
block 3: $32 \times 32 \times 64$ $\rightarrow$ pool
bottleneck: $16 \times 16 \times 128$

Decoder

up: $32 \times 32 \times 64$; concatenate block 3: $\times 128$; conv $\rightarrow 64$
up: $64 \times 64 \times 32$; concatenate block 2: $\times 64$; conv $\rightarrow 32$
up: $128 \times 128 \times 16$; concatenate block 1: $\times 32$; conv $\rightarrow 16$
1×1 conv: $128 \times 128 \times 1$ (person score per pixel)

- Concatenating doubles the channels; the following convolutions learn how to merge "what" (from below) with "where" (from the skip).
- The output has the same height and width as the input: one prediction per pixel.

Intuition

Read the shapes as a U: down the left side, across the bottom, up the right side, with bridges at every level.

Why U-Net works with very little data

- The original U-Net was trained on about 30 microscopy images.
- Every pixel is a training example: one 512×512 image gives a quarter of a million labelled pixels.
- **Elastic deformations**: gently warp images and masks together, mimicking how soft tissue and cells really vary.
- Skips carry fine detail, so the network does not have to *learn* to reconstruct it.

nnU-Net (2021)

A self-configuring U-Net: it inspects a new medical dataset (image sizes, voxel spacing, intensities) and chooses preprocessing, architecture and training settings automatically. Without manual tuning, it won or placed highly in many medical segmentation challenges. Use it as your baseline for medical segmentation.

Intuition

A strong, well-configured standard method beats a clever new one that is badly tuned. This is true far beyond medicine.

An experiment: do the skip connections matter?

Set-up

- Penn-Fudan pedestrians: 120 images to train, 50 to test, resized to 128×128.
- Two tiny U-Nets (16 to 128 channels), identical except that one has its skip connections removed.
- 40 epochs each, Adam, flips as augmentation, on a laptop CPU (about 8 minutes each).

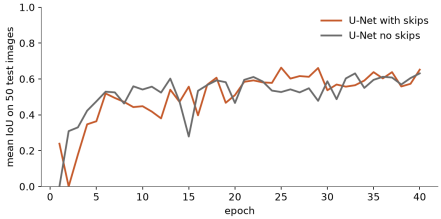
Intuition

Without skips, the decoder must rebuild the pedestrian's outline from the 16×16 bottleneck alone. With skips, it can copy edges from the 128×128 encoder layer. What should we expect to see?

Practitioner's Note

This is a controlled experiment: one change, everything else fixed. Use the same habit when you decide whether a design choice is worth its cost.

Result: finer borders, similar overlap



Mean IoU on 50 test images after 40 epochs: **0.65** with skips, 0.63 without. The curves are noisy and cross many times.



Intuition

Look at the masks, not only the number: with skips the outlines follow legs, arms and gaps between people more closely, but are more fragmented; without skips they are smoother blobs that overlap the people about as well. IoU rewards area overlap, so it barely separates them.

What this small experiment does and does not show

- One run, 120 training images at 128×128, 40 epochs on a CPU: a 2-point difference is within the noise you can see in the curves.
- The *qualitative* difference is clear: skips carry fine spatial detail from the encoder to the decoder.
- With more data, higher resolution and longer training, published results show a consistent advantage for skip connections, which is why every modern segmentation network has them.

Practitioner's Note

To claim that one design beats another: repeat each run with several random seeds, report the mean and spread, and use a test set large enough that the difference is bigger than the spread. Also choose a metric that measures what you care about: a boundary metric would separate these two models better than IoU.

Intuition: an honest "the difference is small and noisy here" is a valid result, and tells you what to try next: more data or a pre-trained encoder before architectural tweaks.

Losses for segmentation

Pixel-wise cross-entropy

$$\mathcal{L}_{\text{CE}} = -\frac{1}{N} \sum_{\text{pixels}} \log p_y(\text{pixel})$$

Day 1's loss, averaged over every pixel.

Soft Dice loss

$$\mathcal{L}_{\text{Dice}} = 1 - \frac{2 \sum_i p_i g_i}{\sum_i p_i + \sum_i g_i}$$

p_i: predicted probability of foreground; $g_i \in \{0, 1\}$.

Intuition

Cross-entropy treats every pixel equally, so a rare small lesion contributes little. Dice measures *overlap* directly, so the lesion matters whatever its size.

Practitioner's Note

In medical segmentation, CE + Dice together is the common default (nnU-Net). Check your masks are stored losslessly as PNG, never JPEG (Week 1, Day 2).

Cross-entropy vs. Dice on a small lesion, worked

A 1% lesion

100×100 image; lesion = 100 pixels (1%).
Prediction A: background everywhere.

Pixel accuracy: 99%.
Cross-entropy: only 100 of 10 000 pixels are badly wrong, so the average loss is fairly small.
Dice: $2 \cdot 0/(0 + 100) = 0$: the worst possible.

Prediction B: finds 80 of the lesion pixels, adds 20 false ones.
Dice: $2 \cdot 80/(100 + 100) = 0.80$.

Intuition

Cross-entropy asks "how right is each pixel on average?". Dice asks "how well do the shapes overlap?". For rare, small structures, only the second question matches what the doctor cares about.

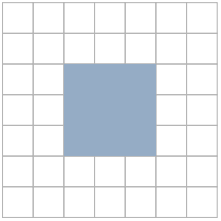
Practitioner's Note

Use both: cross-entropy gives smooth, stable gradients early in training; Dice keeps the rare class in focus.

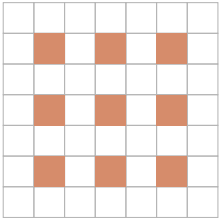
Part 4

DeepLab: context without losing resolution

Atrous (dilated) convolution



rate 1: 3×3 view



rate 2: 5×5 view

- Spread the 9 kernel weights apart with gaps ("à trous" = with holes).
- Same number of weights and multiplications, **larger receptive field**, and no downsampling.
- **ASPP**: several rates in parallel (6, 12, 18) capture context at several sizes, like Inception (Day 2).

Receptive field of a dilated 3×3

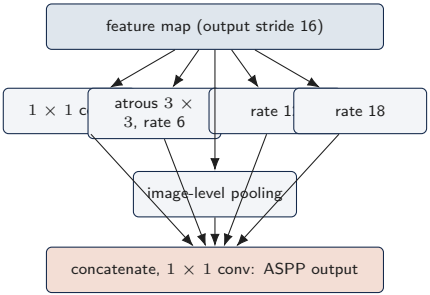
Rate r: the kernel spans $2r + 1$ pixels. Rate 1: 3; rate 2: 5; rate 6: 13; rate 18: 37, still only 9 weights.

Output stride: how much resolution to keep

- **Output stride** = input size / final feature-map size. A standard ResNet has output stride 32.
- DeepLab removes the striding in the last stage(s) and uses atrous convolution instead, to keep the same receptive field.
- Output stride 16 or 8: maps 2–4 times larger in each direction.

A 512×512 image

Stride 32: 16×16 map. Stride 16: 32×32.
Stride 8: 64×64.
Stride 8 costs roughly $4^2 = 16$ times more computation in the last stage than stride 32.



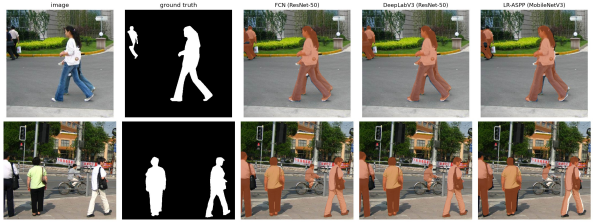
DeepLabV3+ and segmentation on small devices

- **ASPP** (previous slide): atrous convolutions at several rates in parallel, plus a global summary of the whole image: context at every scale.
- **DeepLabV3+** (2018): adds a small U-Net-like decoder that brings in features from an early, high-resolution layer to sharpen boundaries.
- **LR-ASPP** (2019): a light segmentation head on MobileNetV3, designed for phones (the "mobile" row of our table).
- **BiSeNet**, **SegFormer**: fast designs for real-time street-scene segmentation.

Intuition

Every design answers the same question as U-Net: how to get *context* (large receptive fields) and *detail* (sharp borders) at an affordable cost.

Three real segmentation networks, scored on real masks



Pre-trained models (COCO, 21 VOC classes), class "person", scored on 60 Penn-Fudan images against hand-drawn masks.

	IoU	Dice	ms
FCN	0.78	0.86	4774
DeepLabV3	0.78	0.87	6008
LR-ASPP (mobile)	0.71	0.83	361

Intuition

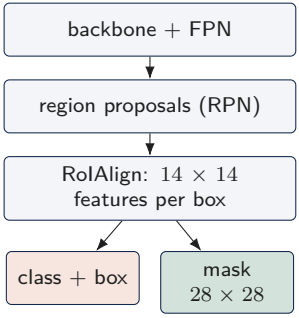
The mobile model is several times faster on CPU and somewhat less precise: the speed–accuracy trade-off of Days 2 and 3, again.

Part 5

Instance segmentation: Mask R-CNN

Mask R-CNN (2017): detect, then segment each object

- Start from Faster R-CNN (Day 3): backbone + FPN, region proposals, box and class heads.
- Add a third head: for each detected box, a small FCN predicts a 28×28 mask inside the box.
- **RoIAlign**: crop features with exact (bilinear) sampling instead of rounding to the grid, because a one-pixel shift ruins a mask.



Intuition

Detection says which objects exist and roughly where; inside each box, the mask head only answers an easy question: "which of these pixels belong to *this* object?".

RoIAlign, worked

- A box edge at $x = 12.6$ on the image falls at $12.6/16 = 0.79$ on a stride-16 feature map.
- RoI pooling (Day 3) rounds it to 0 or 1: up to half a feature cell, i.e. 8 image pixels, of misalignment.
- RoIAlign keeps 0.79 and reads the feature there by **bilinear interpolation** from the four surrounding cells.

Bilinear read at $(x, y) = (0.79, 0.25)$

Neighbouring feature values: $f(0,0) = 2$, $f(1,0) = 6$, $f(0,1) = 4$, $f(1,1) = 8$.
Along x at $y = 0$: $2 + 0.79\,(6 - 2) = 5.16$.
Along x at $y = 1$: $4 + 0.79\,(8 - 4) = 7.16$.
Along y: $5.16 + 0.25\,(7.16 - 5.16) = 5.66$.

Intuition

Eight pixels of misalignment barely change a class label, but they ruin a mask. RoIAlign alone improved Mask R-CNN's mask accuracy substantially: precision in geometry matters for segmentation.

The mask head and its loss

- For each detected box: 14×14 RoIAligned features $\rightarrow$ a few convolutions $\rightarrow$ a transposed conv $\rightarrow$ a 28×28 mask *per class*.
- Only the mask of the predicted class is used; the classification head decides the class.
- Loss: per-pixel binary cross-entropy on that class's 28×28 mask.
- The 28×28 mask is resized to the box's size and pasted into the image.

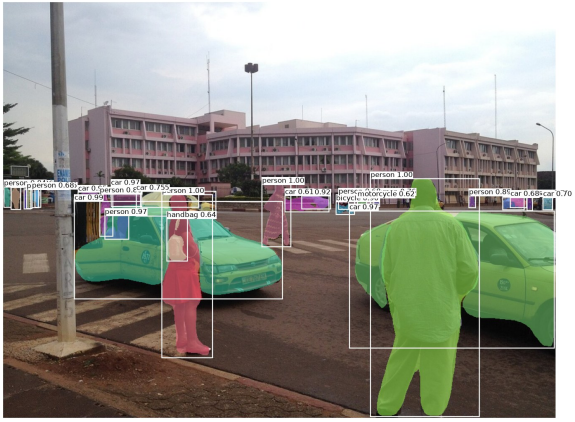
Intuition

Separating "what is it?" (classifier) from "which pixels?" (mask head) makes each job easier: the mask head never has to compete between classes, it only draws an outline for a known class.

Practitioner's Note

Because masks are predicted at 28×28 inside the box, very large objects get slightly blobby borders. If you need precise outlines of large objects, consider a semantic network or SAM for refinement.

Mask R-CNN on real data



Bdx, CC BY-SA 4.0

On the Yaoundé street: 29 objects, each with its own mask, colour and score.

Scored on Penn-Fudan (40 images, 68 annotated pedestrians): for each pedestrian, the best-matching predicted mask has a mean IoU of **0.86**; 100.0% of pedestrians get a mask with IoU above 0.5.

Practitioner's Note

Instance masks let you count *and* measure: the number of cells and the area of each, the number of fruits and their sizes.

Evaluating instance masks: mask AP

- Exactly the AP of Day 3, but a prediction counts as correct when its *mask* IoU with a ground-truth mask is above the threshold.
- COCO reports box AP and mask AP side by side; mask AP is usually a few points lower.
- Our simpler Penn-Fudan score (best-match mask IoU per pedestrian) measures mask *quality* for found objects, not the full precision–recall trade-off.

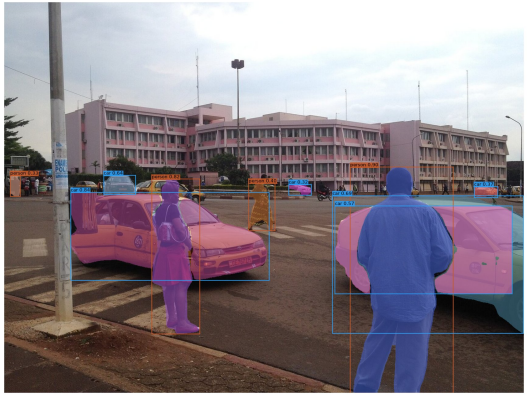
Intuition

Mask IoU is stricter than box IoU: a box can overlap well while the mask misses an arm. Thin structures (legs, bicycle wheels, leaf edges) dominate the errors.

Practitioner's Note

`pycocotools` computes box and mask AP from COCO-format predictions. Use it rather than writing your own for reported numbers.

One-stage instance segmentation



Bdx, CC BY-SA 4.0

- **YOLACT** (2019) and YOLO-seg: the network predicts a few full-image **prototype masks** once, and for each detected object a short list of coefficients.
- Each object's mask = weighted sum of prototypes, cropped to its box.
- Real-time instance segmentation: the YOLOv12-seg output from Week 1, Day 1 (left) was produced this way.

Intuition

Prototypes are like a small palette of shapes; each object mixes its own colour from the palette.

Unified models: one architecture for every kind of segmentation

- **Panoptic FPN** (2019): Mask R-CNN for things + a semantic branch for stuff, merged.
- **Mask2Former** (2022): a transformer decoder with a set of learned *queries*, each predicting one mask and its class, like DETR (Day 3) for masks.
- The same model does semantic, instance and panoptic segmentation, just by how the masks are interpreted.

Intuition

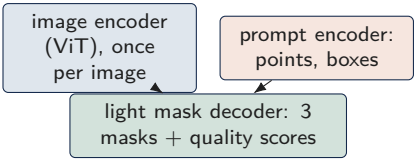
"Predict a set of masks, each with a label" covers all three tasks: semantic = one mask per class; instance = one mask per object; panoptic = both. The distinction becomes a choice of output, not of architecture.

Part 6

Segment Anything

Segment Anything (SAM, 2023)

- Trained on **SA-1B**: 1.1 billion masks on 11 million images, collected with a model-in-the-loop data engine.
- **Promptable**: give it a click, a box or a rough mask; it returns the object's mask.
- No classes: it segments *things*, without naming them.



Intuition

The heavy image encoder runs once; each new click needs only the tiny decoder, so you can segment interactively. SAM returns three masks for an ambiguous click (part, object, group) and rates each one.

Practitioner's Note

SAM is a labelling super-tool: a click instead of tracing an outline. It turns hours of manual masking into minutes, which matters most where expert annotators are scarce.

How SAM's billion masks were made

1. **Assisted manual**: annotators clicked; an early SAM proposed masks; they corrected them. Retrain; repeat.
2. **Semi-automatic**: SAM labelled confident objects; annotators added the missing ones.
3. **Fully automatic**: a grid of points over each image; SAM produced about 100 masks per image, filtered for quality and stability.

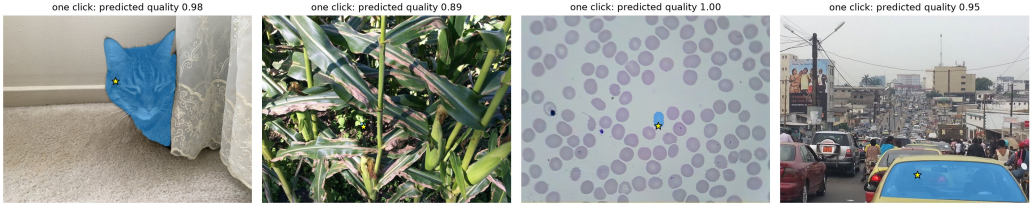
Intuition

The model and the dataset grew together: better models made labelling faster, more labels made better models. This **data engine** is how many foundation models are built.

Practitioner's Note

You can run a small data engine for your capstone: label a few images by hand, train, let the model propose labels on new images, correct them, retrain.

SAM on this course's photographs, one click each



Photos: Famartin; Chroanch; OpenFlexure Project; Guaka (CC BY-SA 4.0)

The star is the only input. The cat half-hidden by a curtain, a maize leaf among many, one red blood cell, a taxi in traffic: each segmented from a single click (SAM ViT-B, about 20.2 s per image on CPU).

Intuition

SAM never saw these images or these labels. Like CLIP on Week 1, Day 1, it is a *foundation model*: trained once on a huge dataset, then used for tasks nobody trained it for.

One click, three answers

- A click on a person's shirt could mean: the shirt, the person, or the whole group.
- SAM returns **three masks** for one click (roughly: part, object, larger group), each with a predicted quality score.
- We showed the highest-scoring mask; an interface can let the user choose.
- More clicks remove the ambiguity: a positive click on the shirt and a negative click on the head.

Intuition

Ambiguity is real, not a model failure: "segment what I clicked" has several correct answers. Predicting several and scoring them is more honest than forcing one.

SAM 2 (2024)

Extends SAM to video: click an object in one frame and it is followed through the clip, with a memory of earlier frames. Useful for tracking cells, animals or vehicles with masks.

What SAM does not do

- It does not **name** objects: add a classifier or a text model (CLIP, Day 5) to label its masks.
- It needs a **prompt**: someone or something must say where to look.
- Thin, low-contrast or unusual structures (vessels, cracks, faint lesions) can be missed or merged.
- Its image encoder is heavy for phones; smaller variants exist.

A practical pipeline

1. A detector finds boxes (Day 3).
2. SAM turns each box into a precise mask.
3. An expert corrects a few masks.
4. Fine-tune a small, fast U-Net on the corrected masks for deployment.

Intuition

Foundation models are often best used to *create data* for a small specialised model, not as the final product.

Part 7

Segmentation in the world

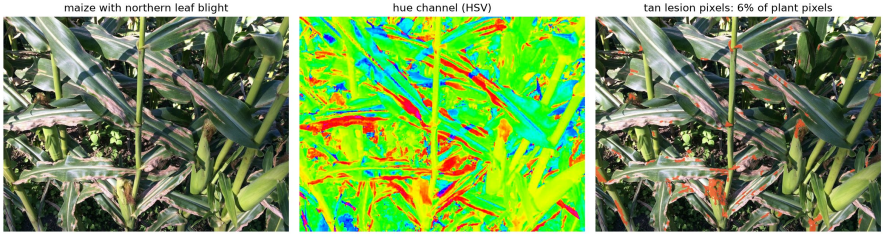
Where segmentation is used

Domain	What is segmented	Why pixels, not boxes
Medicine	tumours, organs, vessels, nuclei	volume, border, growth over time
Agriculture	leaves, lesions, weeds, fields	disease severity (% area), spraying only weeds
Earth observation	buildings, roads, water, forest	area mapped, change over years (Lake Chad)
Driving	road, lanes, pedestrians	where exactly the car may drive
Industry	cracks, corrosion, defects	length and area of damage

Intuition

Choose segmentation when the answer you need is an *area*, a *shape* or a *border*. If you only need counts or locations, detection is cheaper to label and to run.

Revisiting Week 1: measuring disease severity



Chroanch, CC BY-SA 4.0

Week 1's colour rule marked about 6% of plant pixels as lesion. It confuses dry leaf edges and soil with lesions, and misses lesions in shadow.

Intuition

A U-Net trained on a few dozen expert-masked leaves learns what a lesion looks like in context, not just its hue. Severity, the percentage of leaf area affected, then becomes a reliable number that agronomists already use to decide on treatment.

Labelling tools with SAM inside

- **CVAT**, **Label Studio** and others now offer SAM-assisted masks: click an object, accept or correct the mask.
- Export to COCO (polygons or run-length-encoded masks) or to PNG label images.
- Keep a **labelling guide**: what counts as the object, how to treat occlusion, which pixels are "unsure".

A capstone-sized workflow

1. Label 30 images carefully by hand, following the guide.
2. Train a U-Net with a pre-trained encoder.
3. Let it, or SAM, pre-label 100 more; correct them.
4. Retrain; measure Dice on a held-out set labelled by someone else.

Intuition: agreement between two human labellers is your ceiling: if experts agree at Dice 0.85, a model at 0.85 is as good as another expert.

Medical segmentation is usually 3D

- CT and MRI scans are **volumes**: stacks of slices, e.g. $512 \times 512 \times 300$.
- Voxels are often not cubes: 0.7×0.7 mm in-plane, 3 mm between slices. Resample before measuring volume.
- Options: segment slice by slice with a 2D U-Net, or use a **3D U-Net** with $3 \times 3 \times 3$ convolutions on patches of the volume.

Tumour volume from a mask

12 000 tumour voxels of $0.7 \times 0.7 \times 3$ mm $= 1.47\,\text{mm}^3$ each:
$12\,000 \times 1.47 \approx 17\,600\,\text{mm}^3 \approx 17.6\,\text{cm}^3$.

Practitioner's Note

Medical images come as DICOM or NIfTI with the voxel spacing in the header (Week 1, Day 2). Read it with SimpleITK or nibabel; never assume 1 mm voxels.

Segmenting satellite and drone images

- A Sentinel-2 tile is $10\,980 \times 10\,980$ pixels (Week 1, Day 2): far too large for one forward pass.
- **Tile** it into, say, 512×512 pieces *with overlap*, predict each, and stitch, keeping the central part of each tile (borders lack context).
- Inputs often have more than 3 bands; adapt the first layer (Day 2).

How many tiles?

512×512 tiles with 64 pixels of overlap (step 448): $\lceil 10\,980/448 \rceil^2 = 25^2 = 625$ tiles per scene.

Intuition

Mapping buildings, roads, fields or flooded land is semantic segmentation at country scale. Change detection (Lake Chad, Week 1) compares masks of the same place over time.

Practitioner's Note

Keep the geographic coordinates (GeoTIFF) through the whole pipeline, so your masks can be overlaid on a map and measured in hectares.

Week 1 tools still matter after the network

- **Threshold** the probability map (Week 1, Day 3; 0.5 is not always best).
- **Morphological opening and closing** to remove specks and fill small holes (Week 1, Day 3).
- **Connected components** to count objects and measure each one (Week 1, Day 2).
- **Minimum area** rules to drop tiny false detections.

Intuition

A semantic network gives "lesion pixels"; connected components turn them into "three lesions, of 120, 45 and 300 pixels". Classical post-processing turns network outputs into the numbers a user actually needs.

Practitioner's Note

Choose post-processing settings on the validation set, never on the test set, and report them: they change the metrics.

Where is the model unsure?

- A segmentation network outputs a *probability* per pixel, not just a label.
- Pixels near 0.5 are uncertain; they cluster along boundaries and on unusual regions.
- An **uncertainty map** (e.g. entropy of each pixel's probabilities) shows where to look before trusting a mask.

Per-pixel entropy, two classes

$$H = -p \log p - (1-p) \log(1-p)$$

$H = 0$ when p is 0 or 1; largest ($\log 2$) when $p = 0.5$.

Practitioner's Note

In a clinical or field workflow, show the expert the mask *and* the uncertain regions. The expert's time is best spent exactly there.

Intuition

Uncertain pixels are also the best ones to label next (active learning, earlier today): the model tells you where it needs help.

Beyond overlap: boundary metrics

- IoU and Dice summarise area overlap. Two masks with the same Dice can have very different borders.
- **Hausdorff distance**: the largest distance from a point on one boundary to the nearest point on the other. Sensitive to a single bad spike.
- **HD95**: the 95th percentile of those distances; more robust, common in medical challenges.

Same Dice, different risk

Two tumour masks both have Dice 0.85. One is slightly too small all around (HD95 $= 2$ mm); the other is perfect except for a spike 15 mm into healthy tissue (HD95 $= 12$ mm).

For planning radiotherapy, the second is far more dangerous.

Practitioner's Note

Report an overlap metric *and* a boundary metric when borders matter. Distances should be in millimetres, using the voxel spacing.

Choosing the output for your capstone

You need to know ...	Task	Labelling cost per image
whether something is present	classification	seconds
how many, and roughly where	detection	a box per object
how much area, or exact shape	semantic segmentation	a mask per region
how many *and* each one's shape	instance segmentation	a mask per object

Leaf disease

"Is this leaf diseased?": classification (Day 2).
"What percentage of the leaf is affected?": semantic segmentation.
"How many lesions, and their sizes?": instance segmentation.

Practitioner's Note

Choose the simplest task that answers your question, then check that you can afford the labels in the time you have.

Labelling cost, and how to reduce it

How long does labelling take?

Published estimates, per object: a class label takes a second or two; a conventionally drawn box about 25–35 seconds; a COCO polygon mask about 80 seconds.

1000 images with 10 objects each: about 70–100 hours of boxes, about 220 hours of masks.

- **SAM-assisted labelling**: one or two clicks per object.
- **Weak labels**: train from boxes or scribbles instead of full masks.
- **Transfer learning** (Day 2): a pre-trained encoder needs far fewer masks.
- **Active learning**: label the images the model is least sure about first.

Practitioner's Note

Budget the labelling before the modelling. For a capstone, 100 carefully masked images with a pre-trained U-Net encoder beat 1000 sloppy ones.

Part 8

Wrapping up

Key takeaways

1. **Semantic**: a class per pixel. **Instance**: a mask per object. **Panoptic**: both.
2. Measure with **IoU** or **Dice** per class; pixel accuracy is misleading when classes are unbalanced.
3. **FCN**: classify every pixel; upsample back to full size.
4. **U-Net**: encoder + decoder + skips; in our tiny experiment, skips gave finer outlines at similar IoU (0.65 vs. 0.63).
5. **DeepLab**: atrous convolution for context without losing resolution.
6. **Mask R-CNN**: Faster R-CNN plus a mask head per box.
7. **SAM**: promptable, class-free masks; a powerful labelling tool.

Before the next lecture

Practise (tutorial with the tutors)

- Train the tiny U-Net on Penn-Fudan; remove the skips and compare.
- Implement IoU and Dice; compute the all-background baseline.
- Run Mask R-CNN and SAM on your own photos.

Read

- Ronneberger et al. (2015), *U-Net*
- He et al. (2017), *Mask R-CNN*
- Kirillov et al. (2023), *Segment Anything*

Next lecture: transformers, localisation and captioning

- Attention: letting every patch look at every other
- The Vision Transformer
- What transformers attend to
- Describing images in words: captioning
- Finding objects from a text description

Credits

Penn-Fudan Pedestrian Database (University of Pennsylvania). Segmentation and Mask R-CNN weights: torchvision (COCO). SAM ViT-B: Meta AI, via Hugging Face. All experiments were run for this lecture with `Week2/tools/make_week2_figures.py`.

Photographs from Wikimedia Commons:
`yaounde_taxis`: Bdx, CC BY-SA 4.0
`cat_curtain`: Famartin, CC BY-SA 4.0
`maize_blight`: Chroanch, CC BY-SA 4.0
`malaria_smear`: OpenFlexure Project, CC BY-SA 4.0
`douala_2014`: Guaka, CC BY-SA 4.0