SAM 2: Segment Anything in Images and Videos

Radek Bartyzal
Miton Times
1.8.2024

Segment Anything

- also from META
- came out last year
- = promptable segmentation in **images**

What does promptable mean?

- you provide a point OR bounding box OR mask
- that defines which object you want to segment
- => segment anything
- => zero-shot generalization

Images vs Video

- video can be processed as separate images
- but video has more information
- if we use information from previous frame to classify / segment next frames

Also videos have challenges:

- Entities can undergo **significant changes in appearance** due to motion, deformation, occlusion, lighting changes …
- Videos often have **lower quality** than images due to camera motion, blur, and lower resolution
- Efficient processing of a large number of frames is a key challenge

Segment Anything 2

- segment anything in **videos**
- unified model for video and image segmentation
- = consider an image as a **single-frame video**

- paper includes a task, model, and dataset

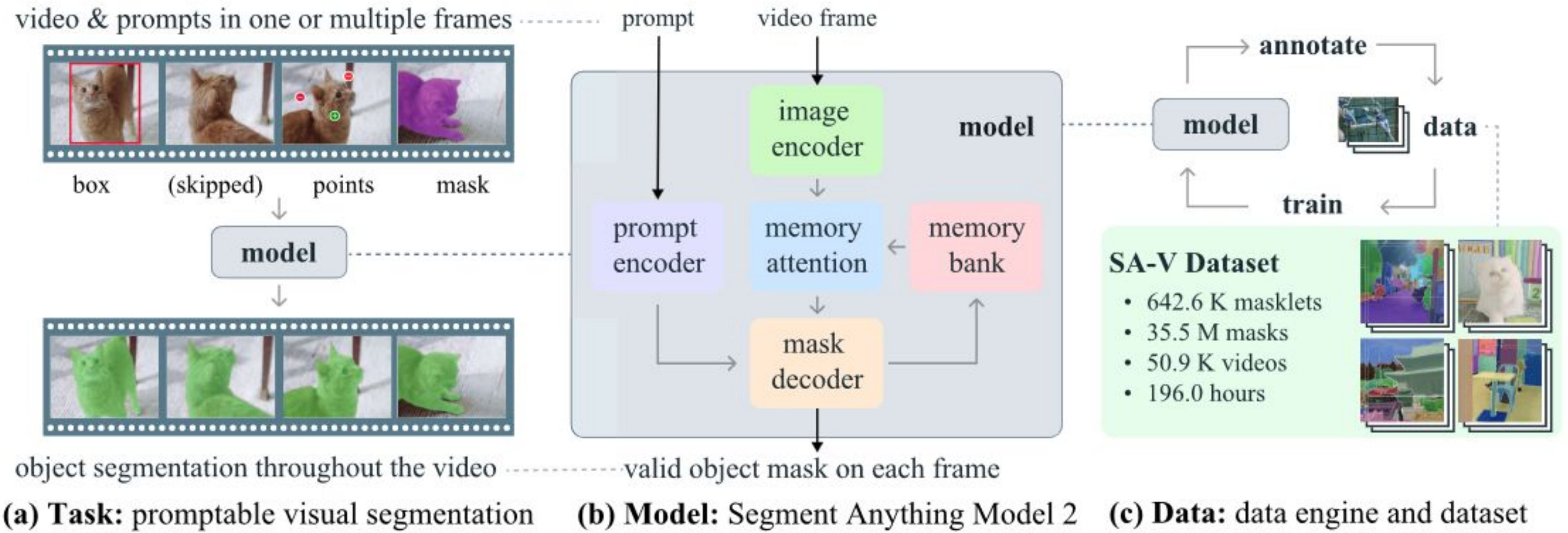

Figure 1 We introduce the Segment Anything Model 2 (SAM 2), towards solving the promptable visual segmentation task (a) with our foundation model (b), trained on our large-scale SA-V dataset collected through our data engine (c). SAM 2 is capable of interactively segmenting regions through prompts (clicks, boxes, or masks) on one or multiple video frames by utilizing a streaming memory that stores previous prompts and predictions.

Task = Promptable Visual Segmentation (PVS)

- generalizes image segmentation to the video domain

Input:

- points (positive **or negative**), boxes, or masks on **any frame** of the video to define a segment of interest for which the spatio-temporal mask (i.e., a **‘masklet’**) is to be predicted

Output:

- masklet = mask for each frame of the video

Once a masklet is predicted, it can be iteratively refined by providing prompts in additional frames.

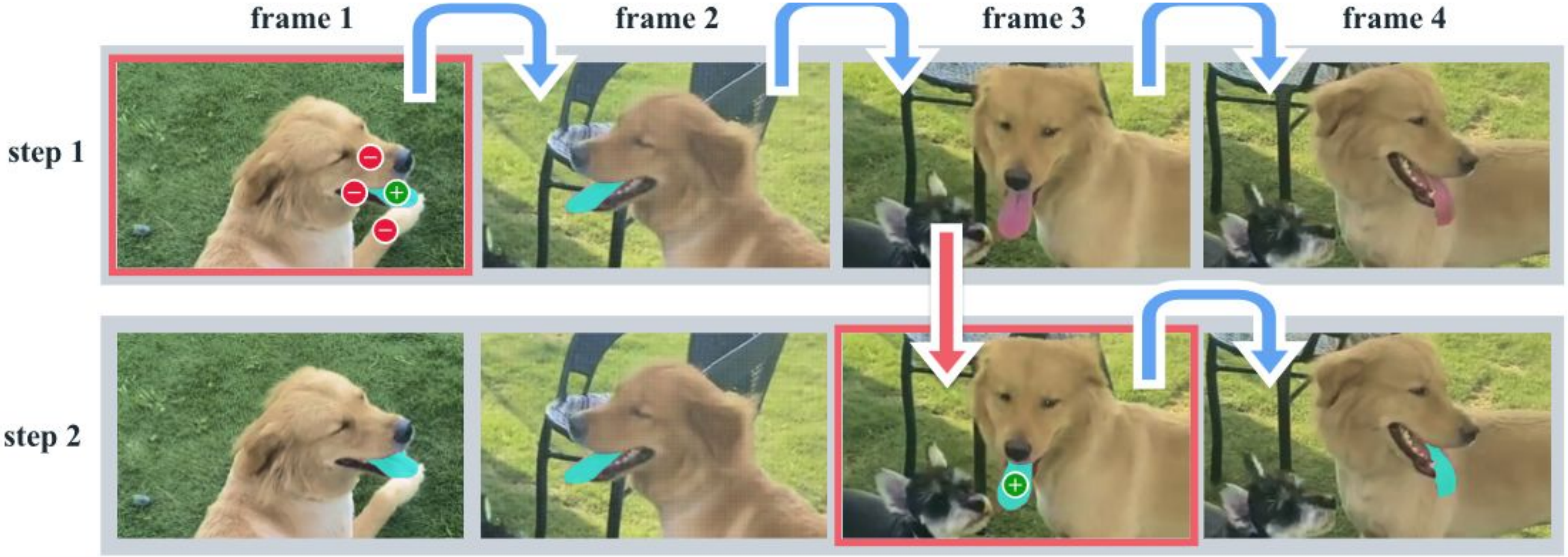

Figure 2 Interactive segmentation with SAM 2. Step 1 (selection): we prompt SAM 2 in frame 1 to obtain the segment of the target object (the tongue). Green/red dots indicate positive/negative prompts respectively. SAM 2 automatically propagates the segment to the following frames (blue arrows) to form a *masklet*. If SAM 2 loses the object (after frame 2), we can correct the masklet by providing an additional prompt in a new frame (red arrow). Step 2 (refinement): a single click in frame 3 is sufficient to recover the object and propagate it to obtain the correct masklet. A decoupled SAM + video tracker approach would require several clicks in frame 3 (as in frame 1) to correctly re-annotate the object as the segmentation is restarted from scratch. With SAM 2's memory, a single click can recover the tongue.

Model

- Unlike SAM, the frame embedding used by the SAM 2 decoder is not directly from an image encoder
- instead it’s conditioned on memories of past predictions and prompted frames
- prompted frames can come “from the future” relative to the current frame.
- Memories of frames are created by the memory encoder based on the current prediction and placed in a memory bank for use in subsequent frames.
- memory attention operation takes the per-frame embedding from the image encoder and conditions it on the memory bank to produce an embedding that is then passed to the mask decoder

Model

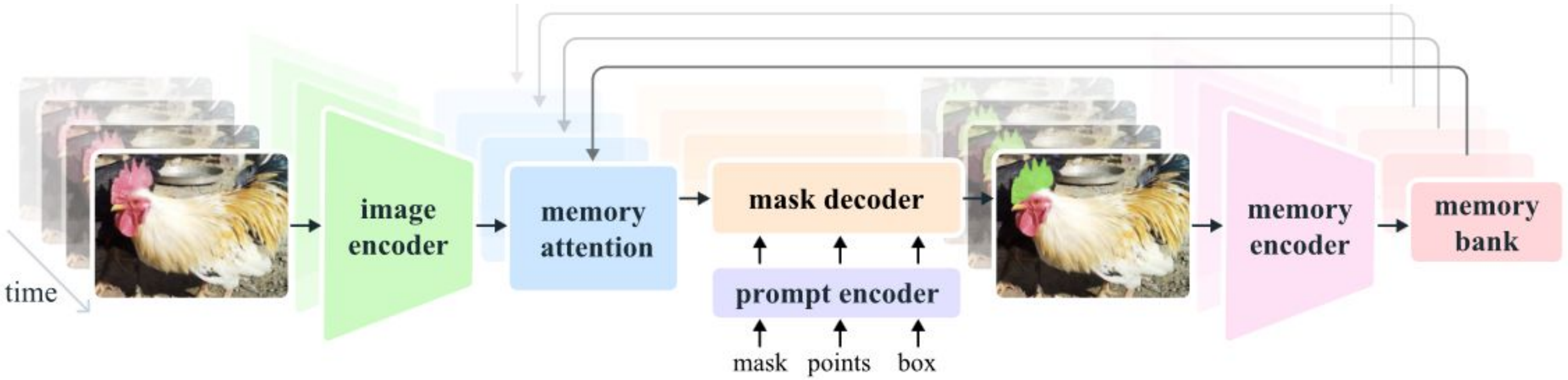

Figure 3 The SAM 2 architecture. For a given frame, the segmentation prediction is conditioned on the current prompt *and/or* on previously observed memories. Videos are processed in a *streaming* fashion with frames being consumed one at a time by the image encoder, and cross-attended to memories of the target object from previous frames. The mask decoder, which optionally also takes input prompts, predicts the segmentation mask for that frame. Finally, a memory encoder transforms the prediction and image encoder embeddings (not shown in the figure) for use in future frames.

Model

- Image encoder = normal pre-trained hierarchical encoder
    - => outputs unconditioned feature embeddings representing each frame
- Memory attention = condition the current frame features on the:
    - past frames features
    - past frame predictions
    - any new prompts
    - stack L transformer blocks
        - the first one taking the image encoding from the current frame as input.
        - Each block performs self-attention, followed by
        - cross-attention to memories, followed by
        - MLP
    - vanilla attention operations for self- and cross-attention

Model: Prompt encoder

Prompt encoder:

- sparse prompts (= clicks, bounding boxes) are represented by positional encodings summed with learned embeddings for each prompt type
- masks are embedded using convolutions and summed with the frame embedding

Model: Prompt decoder

- for Image: predict multiple masks
- for Video: predict multiple masks on each frame
- If no follow-up prompts resolve the ambiguity, the model only propagates the mask with the highest predicted IoU for the current frame.

- in some frames the object can disappear
    - => additional head that predicts whether the object of interest is present on the current frame

- skip connections from the image encoder (bypassing the memory attention) to incorporate high-resolution information for mask decoding

Model: Memory encoder

- generate memory by:
  - downsampling the output mask using a convolutional module
  - and summing it element-wise with the unconditioned frame embedding from the image-encoder
  - followed by light-weight convolutional layers to fuse the information

Model: Memory bank

- FIFO queue of memories of up to N recent frames
- stores information from prompts in a FIFO queue of up to M prompted frames

Training

- simulate prompting from ground truth masks
- = add clicks, bounding boxes, masks

- sample sequence of 8 frames
- prompt up to 2 random frames from it
- then add corrective prompts to help generate correct masking

Model

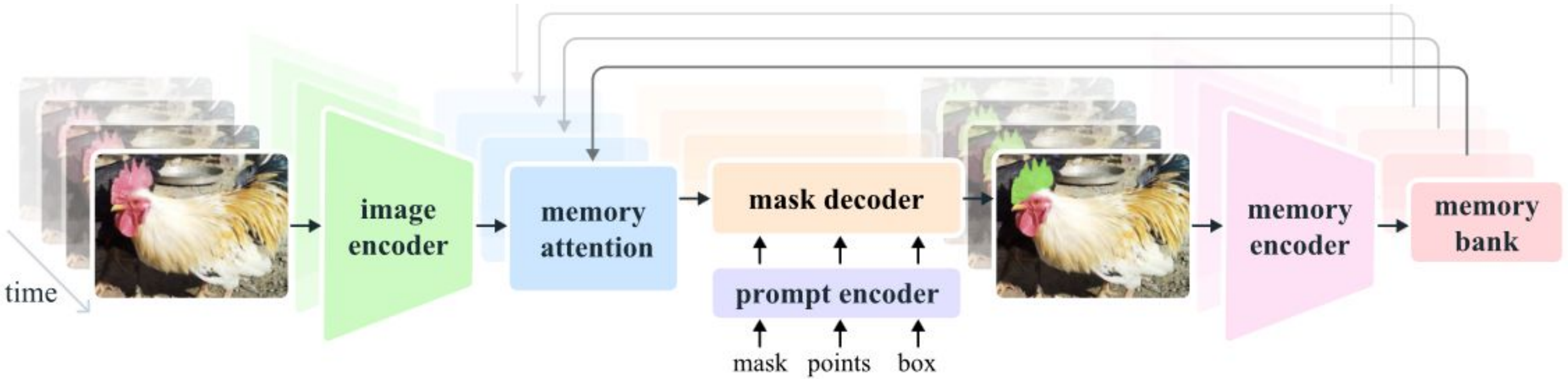

Figure 3 The SAM 2 architecture. For a given frame, the segmentation prediction is conditioned on the current prompt *and/or* on previously observed memories. Videos are processed in a *streaming* fashion with frames being consumed one at a time by the image encoder, and cross-attended to memories of the target object from previous frames. The mask decoder, which optionally also takes input prompts, predicts the segmentation mask for that frame. Finally, a memory encoder transforms the prediction and image encoder embeddings (not shown in the figure) for use in future frames.

Results

- Video segmentation: better accuracy, using 3× fewer interactions than prior approaches.
- Image segmentation: better accuracy and 6× faster than SAM

6.1.2 Semi-supervised video object segmentation

Method	1-click	3-click	5-click	bounding box	ground-truth mask‡
SAM+XMem++	56.9	68.4	70.6	67.6	72.7
SAM+Cutie	56.7	70.1	72.2	69.4	74.1
SAM 2	**64.3**	**73.2**	**75.4**	**72.9**	**77.6**

Table 4 Zero-shot accuracy across 17 video datasets under semi-supervised VOS evaluation using different prompts. The table shows the averaged $\mathcal{J}\&\mathcal{F}$ for each type of prompt (1, 3 or 5 clicks, bounding boxes, or ground-truth masks) in the first video frame (‡: in this case we directly use masks as inputs into XMem++ or Cutie without using SAM).

Release

- code on Github
- trained models - 4 sizes
- all seem pretty reasonable size
-

Sources

- Press release: https://ai.meta.com/blog/segment-anything-2/
- Code: https://github.com/facebookresearch/segment-anything-2
- Demo: https://sam2.metademolab.com/
- Paper