facebookresearch / **sam3**

Code Issues 158 Pull requests 41 Actions Projects Security Insights

New issue

How to fine-tune on my own image/video datasets? #163

Open

Joey-S-Liu opened on Nov 20, 2025

Dear,

Thanks for the good work!

How to fine-tune on my own image/video datasets?
How to format my datasets?

Hope your reply.

atharvaarbat on Nov 20, 2025

Hi @Joey-S-Liu

What to prepare

1. Media
   - Images: a folder of images (jpg/png).
   - Videos: convert to frames (the repo expects frames). Use the repo's extract/download helpers (scripts/eval/*) or any script to produce frame folders (common fps: 6 or 24 depending on your use-case).
   - Expected structure examples seen in repo:
     - data/media//JPEGImages_24fps/<frame_files...>
     - roboflow_vl_100_root/<dataset_name>/{train,valid,test}/...
2. Annotation format
   - Image datasets: COCO-style JSON with:
     - images: [{id, file_name, width, height}, ...]
     - annotations: [{id, image_id, category_id, bbox, segmentation, area, iscrowd, noun_phrase}, ...]
     - categories: [{id, name}, ...]
     - (Optional) a separate queries structure if you want explicit negative queries (phrases with no masks). The training code expects queries/phrases associated to images and will accept phrases that have no matching masks (negative).
   - Video datasets: YTVIS-like JSON (SA-Co/VEval style) with these key fields:

        - info (meta)
        - videos: [{id, video_name, file_names (list of frame filenames), height, width, length}, ...]
        - annotations: [{id, video_id, segmentations (per-frame mask polygons or RLE), bboxes, areas, iscrowd, category_id, noun_phrase, height, width}, ...]
        - categories: same as above
    - Segmentations can be polygons or RLE (typical COCO/YTVIS style). The repo’s VEval README describes this format in detail.

Minimal examples:

```
{
  "info": {"description": "example image dataset"},
  "images": [
    {"id": 1, "file_name": "images/img1.jpg", "width": 1280, "height": 720}
  ],
  "annotations": [
    {
      "id": 10,
      "image_id": 1,
      "category_id": 1,
      "bbox": [100, 150, 200, 120],
      "segmentation": [[100,150, 300,150, 300,270, 100,270]],
      "area": 24000,
      "iscrowd": 0,
      "noun_phrase": "red apple"
    }
  ],
  "categories": [{"id": 1, "name": "object"}]
}
```

```
{
  "info": {"description": "example video dataset"},
  "videos": [
    {
      "id": 1,
      "video_name": "video1",
      "file_names": ["video1/frame_0001.jpg", "video1/frame_0002.jpg"],
      "height": 720,
      "width": 1280,
      "length": 2
    }
  ],
  "annotations": [
    {
      "id": 1,
      "video_id": 1,
      "category_id": 1,
      "segmentations": [
        [[100,150, 300,150, 300,270, 100,270]],   // frame 1 polygon
        [[110,155, 310,155, 310,275, 110,275]]    // frame 2 polygon
      ],
      "bboxes": [[100,150,200,120], [110,155,200,120]],
      "areas": [24000, 24000],
      "iscrowd": 0,
      "noun_phrase": "person riding a bike",
      "height": 720,
      "width": 1280
    }
  ],
  "categories": [{"id": 1, "name": "person"}]
}
```

How training expects data

- Image dataloader: sam3/train/data/sam3_image_dataset.py contains a CustomCocoDetectionAPI and a loader that expects lists of pil_images, annotations, queries, img_metadata. img_metadata may include optional fields such as "blurring_mask"; include if relevant.
- Video dataloader: sam3/train/data/sam3_video_dataset.py includes VideoGroundingDataset which can tile single images into multiple frames for synthetic video training (useful for image-to-video adaptation). For real videos, provide extracted frames and YTVIS-style annotations as shown above.
- The training entrypoint is sam3/train/train.py; job configs are YAML files under configs/* (examples in README_TRAIN.md). Example command shown in repo:
    - python sam3/train/train.py -c configs/roboflow_v100/roboflow_v100_full_ft_100_images.yaml

☺ 👍 17 ❤️ 3

atharvaarbat on Nov 20, 2025 •••

Practical steps to fine-tune

1. Choose whether you train on images or videos.
2. Prepare media:
    - Images: place images in a folder and reference their path in your config.
    - Videos: extract frames into per-video folders (frame filenames must match annotation file_names).
    - Repo provides helper scripts for downloading/extraction for several public datasets (scripts/eval/*).
3. Create annotation JSON:
    - For images: COCO-like JSON; include noun_phrase per annotation.
    - For videos: YTVIS-like JSON (videos + annotations) with per-frame segmentations and noun_phrase.
4. Update a training YAML config:
    - Set dataset root paths (e.g., roboflow_vl_100_root or odinw_data_root used by the repo examples).
    - Set batch size, learning rate, checkpoint paths, number of epochs or iterations. If starting from a SAM-3 checkpoint, point the model init path to the provided checkpoint (use the repo's checkpoint download links).
5. Run training:
    - python sam3/train/train.py -c <your_config.yaml>
6. Validate / iterate:
    - Use evaluation scripts in scripts/eval/... to inspect outputs and run quantitative eval on SA-Co datasets if desired.

Tips and gotchas

- Negative prompts: SA-Co datasets include noun-phrases that have no matching masks (negative). For training, include such phrases as queries associated with an image/video but with no masks in annotations — the training code supports phrases without positive masks.
- Frame filenames and annotation file_names must match exactly for video evaluation.
- If you have videos, prefer extracting frames at the fps that matches your annotations (common options used in repo: 6fps or 24fps).
- Keep category mapping simple (many SA-Co setups use a single category id and rely on noun_phrase to provide the concept). Check configs for category handling examples.

- Optional metadata: dataset code can ingest additional metadata (e.g., blurring_mask). If you need to mask out parts of images, include such fields in image metadata.

👍 14 ❤️ 7 🚀 5

shaopeng666 on Nov 22, 2025

> Dear,
>
> Thanks for the good work!
>
> How to fine-tune on my own image/video datasets? How to format my datasets?
>
> Hope your reply.

Hello, how long does it take to fine-tuning on an A100? **@Joey-S-Liu**

Bin-ze on Nov 22, 2025

I want to use SAM3 to segment my own labeled data (mapped to a limited number of ROI classes) in the same way as traditional instance segmentation. Can I achieve this by fine-tuning the model? How should I do it?

👀 2

noorhashem77 on Nov 23, 2025 · edited by noorhashem77 Edits

Thank you so much for the amazing model, and great work on the documentation!

I'm curious, does this type of fine tunning increase the model performance when it comes to the text prompt? Meaning, if I pass in an image and I use the text prompt and say something like "apples" the model right now is able to detect all apples, but sometimes it misses one or two, and other times it segments things that are not apples.

Is the type of fine-tunning that was mentioned above ^^ able to improve the accuracy of my use case ^ ?

Thanks!

yhy258 on Nov 24, 2025

I guess you can only fine-tune the detector and the shared backbone.

According to the current implementation, it appears that training code for the tracker modules is not provided.
You can see that in Tracker modules, the forward functions are empty or performed in inference mode.

Joey-S-Liu on Nov 24, 2025 Author

> I guess you can only fine-tune the detector and the shared backbone.
>
> According to the current implementation, it appears that training code for the tracker modules is not provided. You can see that in Tracker modules, the forward functions are empty or performed in inference mode.

That is true, that is how we are doing hh.

👍 1

svengoluza on Nov 26, 2025 · edited by svengoluza Edits

How much compute do you need to fine-tune the model? How can we fine-tune it without updating all 840M parameters?

yhy258 on Nov 27, 2025

@svengoluza
You can manually turn off the requires_grad setting in the Trainer where the model is instantiated.

svengoluza mentioned this on Nov 27, 2025

Finetune on custom dataset #244

Joey-S-Liu on Nov 27, 2025 Author

> How much compute do you need to fine-tune the model? How can we fine-tune it without updating all 840M parameters?

When the batch size is 1 and the resolution is 1008, full fine-tuning consumes approximately 18 GB of GPU memory. You could refer to this and make a balance.

👍 10

svengoluza on Nov 27, 2025

> @svengoluza You can manually turn off the requires_grad setting in the Trainer where the model is instantiated.

I suppose I also set those model parts that are frozen to .eval() mode during training as well, and that's basically it?

yhy258 on Nov 27, 2025

@svengoluza For me, yes.

Joey-S-Liu on Nov 27, 2025 Author

> @Joey-S-Liu Did you fine-tune them only using bbox? In my case, when I turn on the enable_segmentation, it said OOM error even if I used H200 and 1 batch size for 1008

What is your enable_segmentation setting?

Joey-S-Liu on Nov 27, 2025 Author

> > @Joey-S-Liu Did you fine-tune them only using bbox? In my case, when I turn on the enable_segmentation, it said OOM error even if I used H200 and 1 batch size for 1008
>
> What is your enable_segmentation setting?

enable_segmentation should only be related to whether the segmentation head is fine-tuned and to the loss, right? It shouldn't have anything to do with the inputs.

2 remaining items

Load more

yhy258 on Nov 28, 2025

@Joey-S-Liu
How did you construct the mask label??
Now I am trying to construct COCO format data from public datasets. But the mask is a semantic segmentation format.
Is it okay to train using only masks with the same label to obtain a binary mask? As you know, SAM3 is instance segmentation.

garg-anant20 on Nov 28, 2025 · edited by garg-anant20 Edits

After doing finetuning on roboflow dataset using the command:

```
clear && python sam3/train/train.py -c
configs/roboflow_v100/roboflow_v100_full_ft_100_images.yaml --use-cluster 0 --num-gpus 1
```

I am getting following metrics:

```
 Accumulating evaluation results...                          DONE (t=0.03s).
                 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.000
                              Average Precision  (AP) @[ IoU=0.50      | area=   all |
maxDets=100 ] = 0.000                                 Average Precision  (AP) @[ IoU=0.75      |
area=   all | maxDets=100 ] = 0.000                              Average Precision  (AP) @[
IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000                          Average
Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.000
      Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.000
                    Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] =
0.000                               Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all |
maxDets= 10 ] = 0.000                                 Average Recall     (AR) @[ IoU=0.50:0.95 |
area=   all | maxDets=100 ] = 0.000                              Average Recall     (AR) @[
IoU=0.50:0.95 | area= small | maxDets=100 ] = -1.000                          Average
Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.000
      Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = 0.000
```

Has anyone else faced this issue?

jlee-detect on Nov 28, 2025

> After doing finetuning on roboflow dataset using the command:
>
> ```
> clear && python sam3/train/train.py -c
> configs/roboflow_v100/roboflow_v100_full_ft_100_images.yaml --use-cluster 0 --num-gpus
> 1
> ```
>
> I am getting following metrics: `Accumulating evaluation results... DONE (t=0.03s). Average Precision (AP) @[IoU=0.50:0.95 | area= all | maxDets=100] = 0.000 Average Precision (AP) @[IoU=0.50 | area= all | maxDets=100] = 0.000 Average Precision (AP) @[IoU=0.75 | area= all | maxDets=100] = 0.000 Average Precision (AP) @[IoU=0.50:0.95 | area= small | maxDets=100] = -1.000 Average Precision (AP) @[IoU=0.50:0.95 | area=medium | maxDets=100] = 0.000 Average Precision (AP) @[IoU=0.50:0.95 | area= large | maxDets=100] = 0.000 Average Recall (AR) @[IoU=0.50:0.95 | area= all | maxDets= 1] = 0.000 Average Recall (AR) @[IoU=0.50:0.95 | area= all | maxDets= 10] = 0.000 Average Recall (AR) @[IoU=0.50:0.95 | area= all | maxDets=100] = 0.000 Average Recall (AR) @[IoU=0.50:0.95 | area= small | maxDets=100] = -1.000 Average Recall (AR) @[IoU=0.50:0.95 | area=medium | maxDets=100] = 0.000 Average Recall (AR) @[IoU=0.50:0.95 | area= large | maxDets=100] = 0.000`
>
> Has anyone else faced this issue?

Do you have details on any config changes or the dataset you used?

jlee-detect mentioned this on Nov 29, 2025

Missing and/or unexpected keys when loading fine-tuned model #260

aniket-professional2025 on Nov 30, 2025 · edited by aniket-professional2025 Edits

Want some help in the fine tune process.

What data i have prepared:

1. dataset root folder is data. It has three folders namely train (train images), test (test images) and validation (validation images). I have also prepared COCO JSON style json formats. One for train (train.json), one for test (test.json) and one for validation (validation.json).
2. I have downloaded the roboflow_v100_full_ft_100_images.yaml. I am facing issues with modification. Since, i am not using roboflow data and using google collaboratory (1 Gpu),I have ommitted the roboflow sub-categories part and changes some other things. I am getting some issue like:

HYDRA_FULL_ERROR=1 to see chained exception.\nfull_key: loss.all") [rank0]: full_key: trainer [rank0]: [W1130 04:31:35.119502044 ProcessGroupNCCL.cpp:1524] Warning: WARNING: destroy_process_group()

I am pretty sure I have done something wrong in the modification of the yaml. How to get the correct modified version? Any resources?

I am having a particular problem:
File "/usr/local/lib/python3.12/dist-packages/omegaconf/_utils.py", line 797, in _raise

raise ex.with_traceback(sys.exc_info()[2]) # set env var OC_CAUSE=1 for full trace

^^

omegaconf.errors.ConfigAttributeError: Key 'launcher' is not in struct

full_key: launcher

object_type=dict

My train.yaml file is:
defaults:

- *self*
- launcher: default
  paths:
  dataset_root: /content/data
  experiment_log_dir: /content/sam3_logs
  bpe_path: /content/sam3/assets/bpe_simple_vocab_16e6.txt.gz
  checkpoint:
  pretrained_weights: /content/sam3_weights.pt
  scratch:
  enable_segmentation: False

Image processing parameters

resolution: 1008
consistent_transform: False
max_ann_per_img: 200

Normalization parameters (standard for SAM3)

train_norm_mean: [0.5, 0.5, 0.5]
train_norm_std: [0.5, 0.5, 0.5]
val_norm_mean: [0.5, 0.5, 0.5]
val_norm_std: [0.5, 0.5, 0.5]

Training parameters

num_train_workers: 8 # Use high number of workers for faster data loading
num_val_workers: 4
hybrid_repeats: 1

Batch sizes (set to 1 for a typical fine-tuning run on a single GPU)

gradient_accumulation_steps: 1
train_batch_size: 1
val_batch_size: 1

Learning rate and scheduler parameters

lr_scale: 0.1 # General scale factor for learning rates
lr_transformer: ${times:8e-4,${scratch.lr_scale}}
lr_vision_backbone: ${times:2.5e-4,${scratch.lr_scale}}
lr_language_backbone: ${times:5e-5,${scratch.lr_scale}}
wd: 0.1
scheduler_timescale: 20
scheduler_warmup: 20
scheduler_cooldown: 20

Matcher configuration (using defaults)

matcher:
target: sam3.train.matcher.BinaryHungarianMatcherV2
focal: true
cost_class: 2.0
cost_bbox: 5.0
cost_giou: 2.0
alpha: 0.25
gamma: 2
stable: False
scale_by_find_batch_size: True

Collate functions

collate_fn:
target: sam3.train.data.collator.collate_fn_api
partial: true
repeats: ${scratch.hybrid_repeats}
dict_key: custom_train
with_seg_masks: ${scratch.enable_segmentation}

collate_fn_val:
target: sam3.train.data.collator.collate_fn_api
partial: true
repeats: ${scratch.hybrid_repeats}
dict_key: custom_val
with_seg_masks: ${scratch.enable_segmentation}

custom_train_transforms:
target: sam3.train.transforms.basic_for_api.ComposeAPI
transforms:
1. Filters out crowd annotations
- *target*: sam3.train.transforms.filter_query_transforms.FlexibleFilterFindGetQueries
query_filter:
target: sam3.train.transforms.filter_query_transforms.FilterCrowds
2. Add noise to ground truth boxes for robustness
- *target*: sam3.train.transforms.point_sampling.RandomizeInputBbox
box_noise_std: 0.1
box_noise_max: 20
3. Decode RLE if segmentation is enabled (harmless if not)
- *target*: sam3.train.transforms.segmentation.DecodeRle
4. Randomly resize and crop the image
- *target*: sam3.train.transforms.basic_for_api.RandomResizeAPI
sizes:
target: sam3.train.transforms.basic.get_random_resize_scales
size: ${scratch.resolution}
min_size: 480
rounded: false
max_size:
target: sam3.train.transforms.basic.get_random_resize_max_size
size: ${scratch.resolution}
square: true
consistent_transform: ${scratch.consistent_transform}
5. Pad the image to the target resolution
- *target*: sam3.train.transforms.basic_for_api.PadToSizeAPI
size: ${scratch.resolution}
consistent_transform: ${scratch.consistent_transform}
6. Convert to Tensor
- *target*: sam3.train.transforms.basic_for_api.ToTensorAPI
7. Filter empty targets that may result from previous transforms
- *target*: sam3.train.transforms.filter_query_transforms.FlexibleFilterFindGetQueries
query_filter:
target: sam3.train.transforms.filter_query_transforms.FilterEmptyTargets
8. Normalize pixel values
- *target*: sam3.train.transforms.basic_for_api.NormalizeAPI
mean: ${scratch.train_norm_mean}
std: ${scratch.train_norm_std}
9. Final filter for empty targets
- *target*: sam3.train.transforms.filter_query_transforms.FlexibleFilterFindGetQueries
query_filter:
target: sam3.train.transforms.filter_query_transforms.FilterEmptyTargets

custom_val_transforms:

Validation transforms pipeline

```
target: sam3.train.transforms.basic_for_api.ComposeAPI
transforms:
- target: sam3.train.transforms.basic_for_api.RandomResizeAPI
sizes: ${scratch.resolution}
max_size:
target: sam3.train.transforms.basic.get_random_resize_max_size
size: ${scratch.resolution}
square: true
consistent_transform: False
- target: sam3.train.transforms.basic_for_api.ToTensorAPI
- target: sam3.train.transforms.basic_for_api.NormalizeAPI
mean: ${scratch.train_norm_mean}
std: ${scratch.train_norm_std}

custom_loss_config:
target: sam3.train.loss.sam3_loss.Sam3LossWrapper
matcher: ${scratch.matcher}
o2m_weight: 2.0
o2m_matcher:
target: sam3.train.matcher.BinaryOneToManyMatcher
alpha: 0.3
threshold: 0.4
topk: 4
use_o2m_matcher_on_o2m_aux: false
loss_fns_find:
- target: sam3.train.loss.loss_fns.Boxes
weight_dict:
loss_bbox: 5.0
loss_giou: 2.0
- target: sam3.train.loss.loss_fns.IABCEMdetr
weak_loss: False
weight_dict:
loss_ce: 20.0
presence_loss: 20.0
pos_weight: 10.0
alpha: 0.25
gamma: 2
use_presence: True
pos_focal: false
pad_n_queries: 200
pad_scale_pos: 1.0

loss_fn_semantic_seg: null # Using null for no segmentation loss
scale_by_find_batch_size: ${scratch.scale_by_find_batch_size}

trainer:
target: sam3.train.trainer.Trainer
skip_saving_ckpts: false
empty_gpu_mem_cache_after_eval: True
skip_first_val: True
max_epochs: 20
accelerator: cuda
seed_value: 123
val_epoch_freq: 5
mode: train
gradient_accumulation_steps: ${scratch.gradient_accumulation_steps}
```

distributed:
backend: nccl
find_unused_parameters: True
gradient_as_bucket_view: True
gpus_per_node: 1

loss:
custom_train: ${custom_loss_config}
custom_val: ${custom_loss_config} # Use same loss for val reporting
default:
target: sam3.train.loss.sam3_loss.DummyLoss

data:

```
  train:
    _target_: sam3.train.data.torch_dataset.TorchDataset
    dataset:
      _target_: sam3.train.data.sam3_image_dataset.Sam3ImageDataset
      img_folder: ${paths.dataset_root}/train
      ann_file: ${paths.dataset_root}/train.json
      transforms: ${custom_train_transforms}
      load_segmentation: ${scratch.enable_segmentation}
      max_ann_per_img: 500000
      multiplier: 1
      training: true
      use_caching: False

    shuffle: True
    batch_size: ${scratch.train_batch_size}
    num_workers: ${scratch.num_train_workers}
    pin_memory: True
    drop_last: True
    collate_fn: ${scratch.collate_fn}

  val:
    _target_: sam3.train.data.torch_dataset.TorchDataset
    dataset:
      _target_: sam3.train.data.sam3_image_dataset.Sam3ImageDataset
      img_folder: ${paths.dataset_root}/validation
      ann_file: ${paths.dataset_root}/validation.json
      transforms: ${custom_val_transforms}
      load_segmentation: ${scratch.enable_segmentation}
      coco_json_loader:
        _target_: sam3.train.data.coco_json_loaders.COCO_FROM_JSON
        include_negatives: true
        category_chunk_size: 2
        _partial_: true
      max_ann_per_img: 100000
      multiplier: 1
      training: false

    shuffle: False
    batch_size: ${scratch.val_batch_size}
    num_workers: ${scratch.num_val_workers}
    pin_memory: True
    drop_last: False
    collate_fn: ${scratch.collate_fn_val}

  model:
    _target_: sam3.model_builder.build_sam3_image_model
    bpe_path: ${paths.bpe_path}
    device: cpus
    eval_mode: false
    enable_segmentation: ${scratch.enable_segmentation}
```

```
meters:
  val:
    custom_val:
      detection:
        _target_: sam3.eval.coco_writer.PredictionDumper
        iou_type: "bbox"
        dump_dir: ${paths.experiment_log_dir}/dumps/custom_val
        merge_predictions: True
        postprocessor:
          _target_: sam3.eval.postprocessors.PostProcessImage
          max_dets_per_img: -1
          use_original_ids: true
          use_original_sizes_box: true
          use_presence: True
        gather_pred_via_filesys: False
        maxdets: 100
        pred_file_evaluators:
          - _target_:
sam3.eval.coco_eval_offline.CocoEvaluatorOfflineWithPredFileEvaluators
            gt_path: ${paths.dataset_root}/validation.json
            tide: False
            iou_type: "bbox"

optim:
  amp:
    enabled: True
    amp_dtype: bfloat16

  optimizer:
    _target_: torch.optim.AdamW

  gradient_clip:
    _target_: sam3.train.optim.optimizer.GradientClipper
    max_norm: 0.1
    norm_type: 2

  param_group_modifiers:
    - _target_: sam3.train.optim.optimizer.layer_decay_param_modifier
      _partial_: True
      layer_decay_value: 0.9 # Typical LRD for vision backbone
      apply_to: 'backbone.vision_backbone.trunk'
      overrides:
        - pattern: '*pos_embed*'
          value: 1.0

  options:
    lr:
      - scheduler:
          _target_: sam3.train.optim.schedulers.InverseSquareRootParamScheduler
          base_lr: ${scratch.lr_transformer}
          timescale: ${scratch.scheduler_timescale}
          warmup_steps: ${scratch.scheduler_warmup}
          cooldown_steps: ${scratch.scheduler_cooldown}

      - scheduler:
          _target_: sam3.train.optim.schedulers.InverseSquareRootParamScheduler
          base_lr: ${scratch.lr_transformer}
          timescale: ${scratch.scheduler_timescale}
          warmup_steps: ${scratch.scheduler_warmup}
          cooldown_steps: ${scratch.scheduler_cooldown}

        param_names:
          - 'backbone.vision_backbone.*'

      - scheduler:
          _target_: sam3.train.optim.schedulers.InverseSquareRootParamScheduler
          base_lr: ${scratch.lr_language_backbone}
          timescale: ${scratch.scheduler_timescale}
          warmup_steps: ${scratch.scheduler_warmup}
          cooldown_steps: ${scratch.scheduler_cooldown}

        param_names:
          - 'backbone.language_backbone.*'

    weight_decay:
      - scheduler:
          _target_: fvcore.common.param_scheduler.ConstantParamScheduler
          value: ${scratch.wd}

      - scheduler:
        _target_: fvcore.common.param_scheduler.ConstantParamScheduler
        value: 0.0

        param_names:
          - '*bias*'

        module_cls_names: ['torch.nn.LayerNorm']

checkpoint:
  save_dir: ${paths.experiment_log_dir}/checkpoints
  save_freq: 0

logging:
  tensorboard_writer:
    _target_: sam3.train.utils.logger.make_tensorboard_logger
    log_dir: ${paths.experiment_log_dir}/tensorboard
    flush_secs: 120
    should_log: True
    wandb_writer: null
```

launcher:
num_nodes: 1
gpus_per_node: 1

mattiagaggi mentioned this on Dec 2, 2025

where is the fine-tuned model/checkpoint/how to use in batch inference #270

MinGiSa mentioned this on Dec 4, 2025

Bug: ValueError: matrix contains invalid numeric entries #289

qos: null

machlovi on Dec 5, 2025 ···

> ### Practical steps to fine-tune
>
> 1. Choose whether you train on images or videos.
> 2. Prepare media:
>   - Images: place images in a folder and reference their path in your config.
>   - Videos: extract frames into per-video folders (frame filenames must match annotation file_names).
>   - Repo provides helper scripts for downloading/extraction for several public datasets (scripts/eval/*).
> 3. Create annotation JSON:
>   - For images: COCO-like JSON; include noun_phrase per annotation.
>   - For videos: YTVIS-like JSON (videos + annotations) with per-frame segmentations and noun_phrase.

> 4. Update a training YAML config:
>   - Set dataset root paths (e.g., roboflow_vl_100_root or odinw_data_root used by the repo examples).
>   - Set batch size, learning rate, checkpoint paths, number of epochs or iterations. If starting from a SAM-3 checkpoint, point the model init path to the provided checkpoint (use the repo's checkpoint download links).
> 5. Run training:
>   - python sam3/train/train.py -c <your_config.yaml>
> 6. Validate / iterate:
>   - Use evaluation scripts in scripts/eval/... to inspect outputs and run quantitative eval on SA-Co datasets if desired.
>
> ### Tips and gotchas
>
> - Negative prompts: SA-Co datasets include noun-phrases that have no matching masks (negative). For training, include such phrases as queries associated with an image/video but with no masks in annotations — the training code supports phrases without positive masks.
> - Frame filenames and annotation file_names must match exactly for video evaluation.
> - If you have videos, prefer extracting frames at the fps that matches your annotations (common options used in repo: 6fps or 24fps).
> - Keep category mapping simple (many SA-Co setups use a single category id and rely on noun_phrase to provide the concept). Check configs for category handling examples.
> - Optional metadata: dataset code can ingest additional metadata (e.g., blurring_mask). If you need to mask out parts of images, include such fields in image metadata.

Thank you for the tips. The issue I am facing right now is that the roboflow yaml file only supports images dataset and from my undersatrding we have to pass videobuilder in order to process videos.
data:
train:
target: sam3.train.data.torch_dataset.TorchDataset
dataset:
# *target*: sam3.train.data.sam3_image_dataset.Sam3ImageDataset
target: sam3.train.data.sam3_video_dataset.VideoGroundingDataset

model:
# *target*: sam3.model_builder.build_sam3_image_model
# bpe_path: ${paths.bpe_path}
# device: cpus
# eval_mode: false
# enable_segmentation: ${scratch.enable_segmentation} # Warning: Enable this if using segmentation.
target: sam3.model_builder.build_sam3_video_model
bpe_path: ${paths.bpe_path}
has_presence_token: True
geo_encoder_use_img_cross_attn: True
apply_temporal_disambiguation: True

Does anyone has idea, if I am doing right or if they have face the similar issue?

WYS-WHU on Dec 5, 2025 ···

> After doing finetuning on roboflow dataset using the command:
>
> ```
> clear && python sam3/train/train.py -c
> configs/roboflow_v100/roboflow_v100_full_ft_100_images.yaml --use-cluster 0 --num-gpus
> 1
> ```
>
> I am getting following metrics: `Accumulating evaluation results... DONE (t=0.03s). Average Precision (AP) @[IoU=0.50:0.95 | area= all | maxDets=100] = 0.000 Average Precision (AP) @[IoU=0.50 | area= all | maxDets=100] = 0.000 Average Precision (AP) @[IoU=0.75 | area= all | maxDets=100] = 0.000 Average Precision (AP) @[IoU=0.50:0.95 | area= small | maxDets=100] = -1.000 Average Precision (AP) @[IoU=0.50:0.95 | area=medium | maxDets=100] = 0.000 Average Precision (AP) @[IoU=0.50:0.95 | area= large | maxDets=100] = 0.000 Average Recall (AR) @[IoU=0.50:0.95 | area= all | maxDets= 1] = 0.000 Average Recall (AR) @[IoU=0.50:0.95 | area= all | maxDets= 10] = 0.000 Average Recall (AR) @[IoU=0.50:0.95 | area= all | maxDets=100] = 0.000 Average Recall (AR) @[IoU=0.50:0.95 | area= small | maxDets=100] = -1.000 Average Recall (AR) @[IoU=0.50:0.95 | area=medium | maxDets=100] = 0.000 Average Recall (AR) @[IoU=0.50:0.95 | area= large | maxDets=100] = 0.000`
>
> Has anyone else faced this issue?

Have you solved this problem? I also got the same evaluation output in roboflow and odinw following the steps in README_TRAIN.md

☺ 👍 3

wakasturner mentioned this on Dec 18, 2025

○ Training Fails with AttributeError: 'list' object has no attribute 'popitem' on Custom COCO Dataset #364

smartkyx on Dec 19, 2025 ···

> ### 微调的实用步骤
>
> 1. 你可以选择以图片训练还是视频训练。
> 2. 准备媒体：
>   - 图片：把图片放到文件夹里，并在配置中引用它们的路径。
>   - 视频：将帧提取到每个视频文件夹中（帧文件名必须与注释file_names匹配）。
>   - 仓库为多个公共数据集（scripts/eval/*）提供下载/提取辅助脚本。
> 3. 创建注释 JSON：
>   - 图片方面：类COCO的JSON;每个注释都包含noun_phrase。
>   - 视频方面：类似YTVIS的JSON（视频+注释），带有每帧分割和noun_phrase。
> 4. 更新训练用的 YAML 配置：
>   - 设置数据集根路径（例如仓库示例中使用的 roboflow_vl_100_root 或 odinw_data_root）。
>   - 设置批处理大小、学习率、检查点路径、纪元数或迭代数。如果从SAM-3检查点开始，则将模型初始路径指向提供的检查点（使用仓库的检查点下载链接）。
> 5. 跑步训练：
>   - Python SAM3/train/train.py -c <your_config.yaml>
> 6. 验证/迭代：
>   - 在脚本/评估等中使用评估脚本等。如有需要，还能检查产出并对SA-Co数据集进行定量评估。

> ### 技巧与陷阱
>
> - 否定提示：SA-Co数据集包含无匹配掩码的名词短语（负面）。在训练中，在注释中包含与图片/视频相关的查询，但无掩码——训练代码支持无正掩码的短语。
> - 视频评估时，帧文件名和注释file_names必须完全匹配。
> - 如果你有视频，建议以与注释相符的帧率提取帧（仓库常用的6fps或24fps）。
> - 保持类别映射简单（许多SA-Co设置使用单一类别ID，并依赖noun_phrase来提供概念）。查看配置中的类别处理示例。
> - 可选元数据：数据集代码可以导入额外的元数据（例如，blurring_mask）。如果你需要遮蔽图像的部分，可以在图像元数据中包含这些字段。

Thank you very much for your sharing; it has been very helpful to me. However, I couldn't find an early stopping strategy in the configuration file. In this case, how should I determine and select the optimal model training parameters?

machlovi mentioned this on Dec 19, 2025

○ Systematic way to freeze parameters for fine-tuning. #284

aniket-professional2025 last month ···

When considering the polygons as bounding boxes, i am fine tuning the sam3 model. However, it is giving me the warnings:

Warning, empty mask found, approximating from box
Warning, empty mask found, approximating from box

why? What does this warning means?

yhy258 last month · edited by yhy258 Edits ▾ ···

> When considering the polygons as bounding boxes, i am fine tuning the sam3 model. However, it is giving me the warnings:
>
> Warning, empty mask found, approximating from box
> Warning, empty mask found, approximating from box
>
> why? What does this warning means?

I think you can disable DecodeRle in transforms section of yaml.
In default yaml setting, transform configuration included DecodeRle. Vs

Jing570 3 weeks ago ···

> ### Practical steps to fine-tune
>
> 1. Choose whether you train on images or videos.

> 2. Prepare media:
>
>   - Images: place images in a folder and reference their path in your config.
>   - Videos: extract frames into per-video folders (frame filenames must match annotation file_names).
>   - Repo provides helper scripts for downloading/extraction for several public datasets (scripts/eval/*).
>
> 3. Create annotation JSON:
>
>   - For images: COCO-like JSON; include noun_phrase per annotation.
>   - For videos: YTVIS-like JSON (videos + annotations) with per-frame segmentations and noun_phrase.
>
> 4. Update a training YAML config:
>
>   - Set dataset root paths (e.g., roboflow_vl_100_root or odinw_data_root used by the repo examples).
>   - Set batch size, learning rate, checkpoint paths, number of epochs or iterations. If starting from a SAM-3 checkpoint, point the model init path to the provided checkpoint (use the repo’s checkpoint download links).
>
> 5. Run training:
>
>   - python sam3/train/train.py -c <your_config.yaml>
>
> 6. Validate / iterate:
>
>   - Use evaluation scripts in scripts/eval/... to inspect outputs and run quantitative eval on SA-Co datasets if desired.
>
> ## Tips and gotchas
>
> - Negative prompts: SA-Co datasets include noun-phrases that have no matching masks (negative). For training, include such phrases as queries associated with an image/video but with no masks in annotations — the training code supports phrases without positive masks.
> - Frame filenames and annotation file_names must match exactly for video evaluation.
> - If you have videos, prefer extracting frames at the fps that matches your annotations (common options used in repo: 6fps or 24fps).
> - Keep category mapping simple (many SA-Co setups use a single category id and rely on noun_phrase to provide the concept). Check configs for category handling examples.
> - Optional metadata: dataset code can ingest additional metadata (e.g., blurring_mask). If you need to mask out parts of images, include such fields in image metadata.

Thanks for the amazing work on SAM 3!
I am trying to fine-tune the model on a custom dataset for Promptable Concept Segmentation. My dataset contains specific noun_phrase annotations (e.g., "red apple" vs "green apple") as described in the training documentation ("Annotation format").

However, when investigating the data loading logic in sam3/train/data/coco_json_loaders.py, I noticed that the COCO_FROM_JSON class seems to ignore the noun_phrase field in the annotations and strictly uses the category name as the query text.
Code Reference: In sam3/train/data/coco_json_loaders.py, inside loadQueriesAndAnnotationsFromDatapoint:

Around line 197

query["query_text"] = (
self._cat_idx_to_text[cat_id] # <--- It maps ID to Category Name
if self.prompts is None
else self.prompts[cat_id]
)

Question:
Is COCO_FROM_JSON intended to only support category-based training?
If I want to train with instance-specific noun_phrase (like in the SA-Co data engine described in the paper), is there another loader I should use (e.g., similar to SAM3_EVAL_API_FROM_JSON_NP but for training)?
Or should I modify COCO_FROM_JSON to prioritize reading annotation['noun_phrase']?

Thanks!

☺ 👍 1

Add a comment

Write Preview

Use Markdown to format your comment

Paste, drop, or click to add files

Close issue ▾ Comment

ⓘ Remember, contributions to this repository should follow its contributing guidelines, security policy and code of conduct.

Assignees

No one assigned

Labels

No labels

Type

No type

Projects

No projects

Milestone

No milestone

Relationships

None yet

Development

Code with agent mode

No branches or pull requests

Notifications

Customize

Subscribe

You're not receiving notifications from this thread.

Participants

+9

Give feedback