# Supplementary Material of CASE: Concept-Aware Segmentation from Positive and Negative Visual Exemplars

Yi-Jen Tsai[1], Yen-Yu Lin[1], and Chien-Yao Wang[2⋆]

[1] National Yang Ming Chiao Tung University, Taiwan
tsai.cs14@nycu.edu.tw, lin@cs.nycu.edu.tw
[2] Institute of Information Science, Academia Sinica, Taiwan
kinyiu@iis.sinica.edu.tw

## 1 CASE Benchmark Construction

This section provides additional details on the construction of CASE. CASE is designed to evaluate whether a segmentation model can infer a target concept from both positive and negative visual exemplars. Unlike standard few-shot semantic segmentation benchmarks, where the support set usually defines a single positive class, CASE episodes require the model to parse compositional intent relations such as inclusion, exclusion, fine-grained isolation, functional grouping, attribute selection, attribute union, and attribute exclusion.

### 1.1 Data Sources

CASE is built from two annotation sources. The category-level tasks, including SuperEx, FineIso, and Union, are constructed from MSCOCO instance segmentation annotations [2]. We use COCO train2017 to generate candidate training episodes and COCO val2017 to generate fixed evaluation episodes. The attribute-level tasks, including AttrIso, AttrUnion, and AttrEx, are constructed from PACO-LVIS annotations [3]. PACO provides object-level category annotations and attribute labels, which allow us to define visual concepts based on color and material properties.

For PACO-based tasks, we use object-level categories and filter out part-level categories whose category names contain a part delimiter. This avoids constructing episodes in which the target concept is defined over object parts rather than complete object instances. We further restrict attribute concepts to visually grounded color and material attributes. The color set contains black, blue, brown, green, grey, orange, pink, purple, red, white, and yellow. The material set contains stone, wood, rattan, fabric, crochet, wool, leather, velvet, metal, paper, plastic, glass, and ceramic.

To specify, not every COCO/PACO image is used: only images satisfying the task-specific support and target constraints are eligible. Episode and mask construction is programmatic from the source annotations; only the high-level Union group definitions are manually specified.

⋆ Corresponding author

**Table 1: Data sources used for each CASE task family.** Category-level tasks are constructed from COCO, while attribute-level tasks are constructed from PACO-LVIS.

| Task family | Source annotation | Split usage |
|---|---|---|
| SuperEx | COCO instance segmentation | train2017 / val2017 |
| FineIso | COCO instance segmentation | train2017 / val2017 |
| Union | COCO instance segmentation | train2017 / val2017 |
| AttrIso | PACO-LVIS object attributes | train / val |
| AttrUnion | PACO-LVIS object attributes | train / val |
| AttrEx | PACO-LVIS object attributes | train / val |

### 1.2 Episode Format

Each CASE episode in the benchmark contains a target image, a frozen support set, a task label, an intent description, a frozen target mask, and frozen positive/negative visual prompts. The format is designed for reproducible evaluation: the support examples, prompt boxes, and target masks are fixed in advance, so different methods are evaluated on exactly the same episodes.

A support image contains positive and negative bounding-box prompts. Positive prompts indicate visual exemplars that should be included in the target concept, while negative prompts indicate exemplars that should be excluded or used as discriminative counterexamples. The target image contains a ground-truth mask encoded in COCO-style run-length encoding (RLE). This avoids requiring users to re-run the benchmark construction code.

A simplified released episode format is shown below.

```
{
"episode_id": "...",
"task": "SuperEx",
"intent": "All vehicle MINUS car",
"target": {
"image_id": ...,
"image_path": "...",
"gt_foreground_pixels": ...,
"gt_mask_rle": {
"size": [height, width],
"counts": "..."
},
"pos_bboxes": [...],
"neg_bboxes": [...]
},
"support": [
{
"image_id": ...,
"image_path": "...",
"prompts": {
"pos_bboxes": [...],
```

**Table 2: Detailed statistics of the CASE benchmark**. We report the number of candidate training episodes, fixed evaluation episodes, concept definitions, and average prompts per episode for each task family. The overall average prompt counts are weighted by the number of candidate training episodes.

| Task | Source | #Train Ep. | #Eval Ep. | #Concepts | Avg. Pos / Neg |
|---|---|---|---|---|---|
| SuperEx | COCO | 156,000 | 9,004 | 78 | 6.71 / 2.31 |
| FineIso | COCO | 140,727 | 8,662 | 79 | 2.43 / 6.17 |
| Union | COCO | 50,000 | 1,806 | 25 | 5.04 / 3.96 |
| AttrEx | PACO | 12,909 | 1,528 | 109 | 2.09 / 1.04 |
| AttrIso | PACO | 66,942 | 1,100 | 94 | 1.04 / 2.09 |
| AttrUnion | PACO | 34,812 | 1,116 | 21 | 2.15 / 1.64 |
| **Overall** | COCO/PACO | **461,390** | **23,216** | **406** | **3.94 / 3.56** |


```
"neg_bboxes": [...]
}
}
]
}
```


In the internal construction stage, each episode is first generated with a task-specific dynamic ground-truth configuration. This configuration specifies how the target mask should be produced from the underlying COCO or PACO annotations. For example, SuperEx stores a base supercategory and an excluded category, FineIso stores a target category, Union stores a fixed set of positive categories, AttrIso stores a target object category and a target attribute, AttrUnion stores a target attribute, and AttrEx stores a semantic object group and an excluded attribute. During release compilation, this dynamic configuration is converted into a frozen RLE target mask.

### 1.3 Benchmark Statistics

Tab. 2 provides detailed statistics for CASE. The training column reports candidate episodes used to sample training instances, while the evaluation column reports fixed episodes used for deterministic comparison. The number of concepts counts the distinct structured concept definitions within each task. The average prompt counts are computed across training episodes, and the overall average is weighted by the number of candidate training episodes per task.

### 1.4 Train/Test Splits

CASE follows the original train/validation split of its source annotations. COCO-based tasks use train2017 for candidate training episodes and val2017 for evaluation episodes. PACO-based tasks follow the PACO train/val split. Thus, the training and evaluation episodes are image-disjoint by construction. The split is not concept-disjoint: the same structured concept definition may appear in both splits, while the underlying images remain disjoint.

**Table 3: General construction parameters used in CASE.** The overall average prompt counts reported in the main paper are weighted by the number of candidate training episodes.

| Parameter | Value |
|---|---|
| Minimum target foreground area | 500 pixels |
| Training support candidate pool size | 300 |
| Maximum training support pool size | 100 |
| Maximum target images per concept | 2000 |
| Training episode seed | 42 |
| Evaluation support sampling seed | 42 |
| Evaluation protocols | 1-shot and 5-shot |
| Support nesting | 1-shot $\subset$ 5-shot |
| Maximum positive prompts per support image | 3 |
| Maximum negative prompts per support image | 3 |

The evaluation split is frozen for reproducibility. For each validation episode, we first sample a 5-shot support set using a global random seed. The corresponding 1-shot episode is obtained by taking the first support image from the 5-shot set. Therefore, the 1-shot support set is nested within the 5-shot support set. This design ensures that the 1-shot and 5-shot protocols evaluate the same target episodes while differing only in the number of available support exemplars.

At training time, support images are dynamically sampled from a larger candidate support pool, which increases prompt diversity while keeping the underlying intent definition fixed. Unless otherwise specified, candidate training episodes are generated with a fixed random seed, and each concept is capped by a maximum number of target images to avoid severe imbalance.

### 1.5 Prompt Sampling Protocol

CASE uses bounding boxes as visual prompts. Each support image may contain multiple positive and negative boxes. Positive boxes are selected from instances that satisfy the intended concept, whereas negative boxes are selected from semantically or visually related distractors. Thus, negatives are task-related distractors rather than arbitrary classes. All prompts correspond to object-level instances; CASE does not cover stuff categories. This design forces the model to infer not only what should be segmented, but also what should be rejected.

To reduce annotation noise and duplicated prompts, candidate boxes are filtered by an IoU-based duplicate suppression rule. A newly selected box is discarded if it overlaps too strongly with an existing prompt box. We also avoid selecting positive boxes that collide with negative boxes, since such overlaps would make the support intent ambiguous.

Training and evaluation differ in strictness. For training, we adopt a target-relaxed but support-strict protocol. A training target image is required to contain at least one positive foreground instance, but it is not always required to contain a negative distractor. This increases concept coverage and avoids over-restricting the training distribution. In contrast, the support set remains strict: support

**Table 4: Task-specific construction rules in CASE.** Pos. and Neg. indicate the positive and negative support prompts, respectively.

| Task | Intent | Target mask | Pos. prompts | Neg. prompts |
|---|---|---|---|---|
| SuperEx | All objects in a COCO supercategory except one excluded category. | Union of all instances in the supercategory minus instances of the excluded category. | Instances from non-excluded categories in the same supercategory. | Instances from the excluded category. |
| FineIso | Only one fine-grained category within a COCO supercategory. | Instances of the target category. | Instances of the target category. | Instances from other categories in the same supercategory. |
| Union | A fixed high-level category group, such as public transportation or rest furniture. | Union of all instances belonging to the positive categories of the group. | Instances from at least two positive categories when possible. | Instances from fixed distractor categories associated with the group. |
| AttrIso | Objects in a fixed object group with a specified color or material attribute. | Instances in the target object group that possess the target attribute. | Same-group instances with the target attribute. | Same-group instances with different attributes. |
| AttrUnion | All object instances with a specified color or material attribute, regardless of category. | All valid object-level instances that possess the target attribute. | Instances from multiple categories sharing the target attribute. | Instances with different attributes from the same attribute family. |
| AttrEx | Objects in a semantic object group excluding a specified color or material attribute. | Instances in the semantic group minus instances with the excluded attribute. | Instances with non-excluded attributes from the same attribute family. | Instances with the excluded attribute. |

images are required to contain explicit positive and negative exemplars so that each training episode provides a clear contrastive intent.

For evaluation, we adopt a stricter discriminative protocol. Whenever the task definition involves exclusion or disambiguation, target images are required to contain both foreground objects satisfying the intent and related distractors. Support images are also required to contain positive and negative exemplars. As a result, the evaluation split directly tests whether a model can use negative visual prompts to suppress distractors rather than merely segmenting all objects that resemble the positive exemplars.

### 1.6 Dynamic Ground-truth Mask Generation

The target mask of each episode is generated from the source annotations according to its task-specific intent. For COCO-based tasks, masks are generated from instance segmentation annotations. For PACO-based tasks, masks are generated by filtering object annotations according to semantic object groups and visual attributes. The dynamic mask rules are summarized in Table 4. In exclusion tasks, the target mask is computed by subtracting the excluded instances from the base foreground. In union tasks, the target mask is the union of all instances that satisfy the positive concept. In attribute tasks, the target mask is obtained by selecting or excluding instances according to PACO attribute labels.

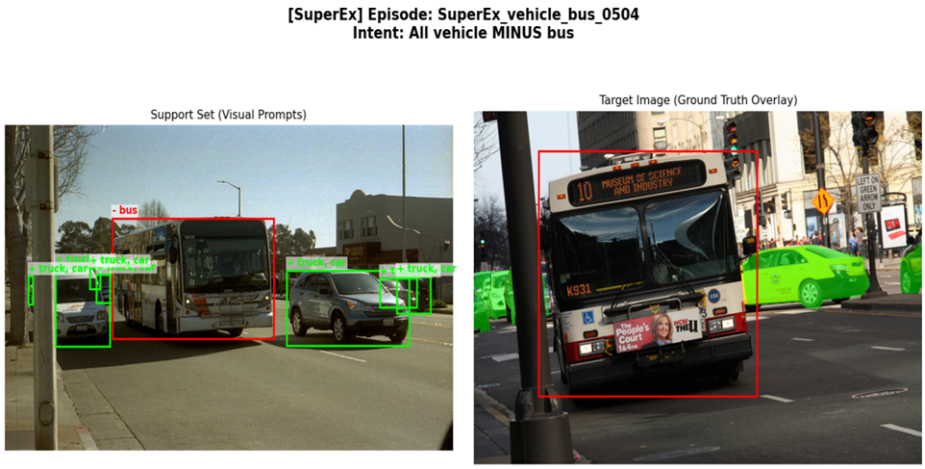



**Fig. 1: Example SuperEx episode.** The intent is to segment all vehicles except buses.

## 2 Task-specific Details

This section describes the six CASE task families in detail. Each task is designed to evaluate a different form of concept reasoning from positive and negative visual exemplars. In all task illustrations, green boxes on support images denote positive visual prompts and red boxes denote negative visual prompts. On target images, the green overlay denotes the ground-truth target mask. Red boxes on target images are shown only to indicate distractor instances for visualization and are not provided as model input during evaluation.

### 2.1 SuperEx: Supercategory Exclusion

SuperEx evaluates whether a model can infer a broad category-level concept while excluding a specified fine-grained category. Each concept is defined as

$$\mathcal{Y} = \mathcal{C}_{\text{super}} \setminus \mathcal{C}_{\text{neg}},$$

where $\mathcal{C}_{\text{super}}$ is a COCO supercategory and $\mathcal{C}_{\text{neg}}$ is one excluded category within that supercategory. For example, an episode may define the intent "all vehicles except cars." Positive prompts are sampled from non-excluded categories in the same supercategory, such as buses and trains, while negative prompts are sampled from the excluded category, such as cars.

The target mask is generated by first taking the union of all instances in the base supercategory and then subtracting all instances belonging to the excluded category:

$$M = M(\mathcal{C}_{\text{super}}) \setminus M(\mathcal{C}_{\text{neg}}).$$




**Fig. 2: Example FineIso episode.** The intent is to isolate trucks from other vehicles.

This task tests whether a model can use negative exemplars to prevent over-generalization. A model that relies only on positive prompts may infer the broad supercategory and incorrectly include the excluded category.

For evaluation, target images are required to contain both valid foreground instances and excluded-category distractors. Support images are required to include at least two positive categories and at least one negative exemplar. This makes SuperEx a direct test of category-level exclusion.

### 2.2 FineIso: Fine-grained Isolation

FineIso evaluates whether a model can isolate a single fine-grained category from related categories within the same supercategory. Each concept is defined as

$$\mathcal{Y} = \mathcal{C}_{\text{target}},$$

where $\mathcal{C}_{\text{target}}$ is one COCO category. For example, an episode may define the intent "only dogs" within the animal supercategory. Positive prompts are sampled from the target category, while negative prompts are sampled from other categories in the same supercategory, such as cats, sheep, horses, or cows.

The target mask is the union of all instances of the target category:

$$M = M(\mathcal{C}_{\text{target}}).$$

The negative prompts do not define a category to subtract from a broader foreground. Instead, they act as discriminative counterexamples that specify the intended granularity of the concept.

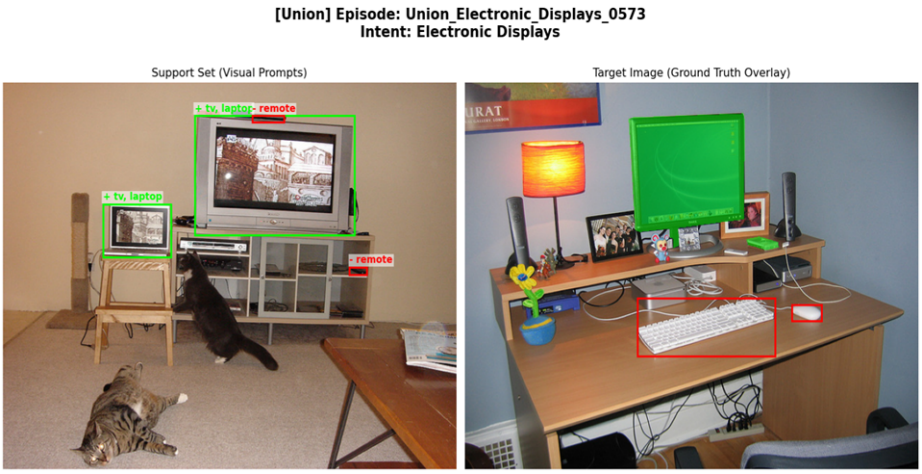



**Fig. 3: Example Union episode.** The intent is to segment electronic displays across related object categories (electronic devices).

For evaluation, target images are required to contain the target category and at least one same-supercategory distractor. Support images are required to contain at least one positive instance and multiple negative distractors. This makes FineIso particularly challenging for models that treat visually related categories as interchangeable.

### 2.3 Union: Function-driven Category Union

Union evaluates whether a model can infer a high-level concept that spans multiple COCO categories. Unlike SuperEx and FineIso, Union is not derived directly from COCO supercategories. Instead, we manually define a fixed set of high-level intent groups. Each group contains positive categories that share a functional or semantic relation and negative categories that serve as related distractors.

Each Union concept is defined as

$$\mathcal{Y} = \bigcup_{c \in \mathcal{C}_{\text{pos}}} c,$$

where $\mathcal{C}_{\text{pos}}$ is the manually defined positive category set for a given intent. The target mask is the union of all positive-category instances:

$$M = \bigcup_{c \in \mathcal{C}_{\text{pos}}} M(c).$$

For example, the intent "public transportation" contains buses and trains as positive categories and cars and motorcycles as negative distractors. Similarly,

**Table 5: Representative examples of manually defined Union concepts.**

| Intent | Positive categories | Negative categories |
|---|---|---|
| Public Transportation | bus, train | car, motorcycle |
| Human Powered Mobility | bicycle, skateboard | car, motorcycle |
| Rest Furniture | bed, couch | chair, dining table |
| Domestic Pets | dog, cat | cow, horse, zebra |
| Drink Containers | cup, wine glass, bottle | fork, knife |
| Electronic Displays | tv, laptop, cell phone | remote, keyboard, mouse, microwave |
| Cutlery | fork, knife, spoon | bowl, cup, wine glass, dining table |




**Fig. 4: Example AttrIso episode.** The intent is to segment glass containers while excluding containers with other attributes.

"rest furniture" contains beds and couches as positives and chairs and dining tables as negatives.

The negative prompts in Union are not subtracted from a base mask. Instead, they define the boundary of the high-level concept by showing related categories that should not be included. This setting tests whether a model can infer an abstract group from positive exemplars rather than simply segmenting all objects from a broad COCO supercategory. Table 5 lists representative Union concepts. The full implementation uses a fixed set of such groups across all experiments.

### 2.4 AttrIso: Attribute Isolation within an Object Group

AttrIso evaluates whether a model can isolate objects with a specified visual attribute within a fixed object group. Unlike FineIso, where the concept is defined only by object category, AttrIso defines the target by combining an object group and an attribute:

$$\mathcal{Y} = \{x \mid x \in \mathcal{G},\ a_{\text{target}} \in A(x)\},$$

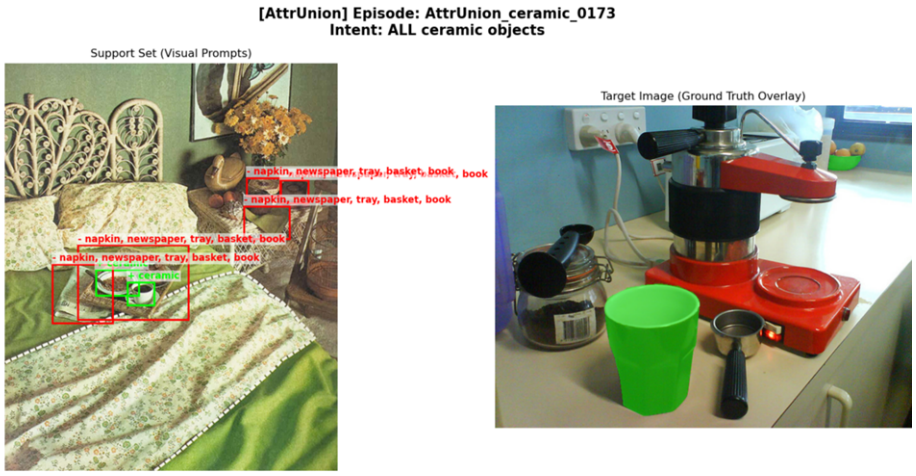



**Fig. 5: Example AttrUnion episode.** The intent is to segment all ceramic objects.

where $\mathcal{G}$ is the target object group, $A(x)$ is the set of PACO attributes assigned to instance $x$, and $a_{\text{target}}$ is the target color or material attribute.

For example, an AttrIso episode may define the intent "glass containers." Positive prompts contain container instances with the target attribute, such as glass containers. Negative prompts contain instances from the same object group with different attributes from the same attribute family, such as plastic containers. This design forces the model to isolate the target attribute while keeping the object group fixed.

The target mask is generated as

$$M = \bigcup_{x \in \mathcal{G}} \mathbf{1}[a_{\text{target}} \in A(x)] M(x).$$

This mask includes only instances that match both the target category and the target attribute. Instances from the same category but with different attributes are excluded, and instances from other categories are not part of the target even if they share the same attribute.

For evaluation, target images are required to contain at least one positive instance and at least one same-category attribute distractor. Support images are required to contain at least one positive prompt and multiple negative prompts from the same object category. We additionally require the negative prompts to contain multiple distractor attributes whenever possible, so that the model cannot solve the episode by suppressing only one specific negative instance.

### 2.5 AttrUnion: Cross-category Attribute Union

AttrUnion evaluates whether a model can segment all objects that share a specified visual attribute across categories. Each concept is defined as

$$\mathcal{Y} = \{x \mid a_{\text{target}} \in A(x)\},$$

where $a_{\text{target}}$ is a target color or material attribute. Unlike AttrIso, AttrUnion does not restrict the target to a semantic group. The target mask includes all valid object-level instances with the target attribute:

$$M = \bigcup_x \mathbf{1}[a_{\text{target}} \in A(x)]M(x).$$

For example, an AttrUnion episode may ask the model to segment all wooden objects or all red objects, regardless of their object categories. Positive prompts are sampled from multiple categories that share the target attribute. Negative prompts are sampled from objects that do not have the target attribute but have other attributes from the same attribute family. This prevents the model from solving the task by using object category alone.

For evaluation, we require the target image to contain the target attribute across at least two object categories whenever possible. Support images are also required to contain positive exemplars from multiple categories and negative exemplars with different attributes. This makes AttrUnion a direct test of category-invariant attribute reasoning.

### 2.6 AttrEx: Attribute Exclusion within a Semantic Group

AttrEx evaluates whether a model can segment objects in a semantic group while excluding a specified visual attribute. Each concept is defined as

$$\mathcal{Y} = \{x \mid x \in \mathcal{G},\ a_{\text{excluded}} \notin A(x)\},$$

where $\mathcal{G}$ is a semantic object group and $a_{\text{excluded}}$ is the excluded color or material attribute. For example, an episode may define the intent "non-red vehicles" or "non-wooden furniture."

The target mask is generated by taking all valid instances in the semantic group and subtracting instances that possess the excluded attribute:

$$M = M(\mathcal{G}) \setminus \bigcup_{x \in \mathcal{G}} \mathbf{1}[a_{\text{excluded}} \in A(x)]M(x).$$

Positive prompts contain instances from the same semantic group with non-excluded attributes. Negative prompts contain instances with the excluded attribute. This construction requires the model to interpret negative prompts as an attribute-level exclusion rather than a category-level exclusion.

For training, target images are required to contain at least one valid non-excluded foreground instance, while support images must contain both positive

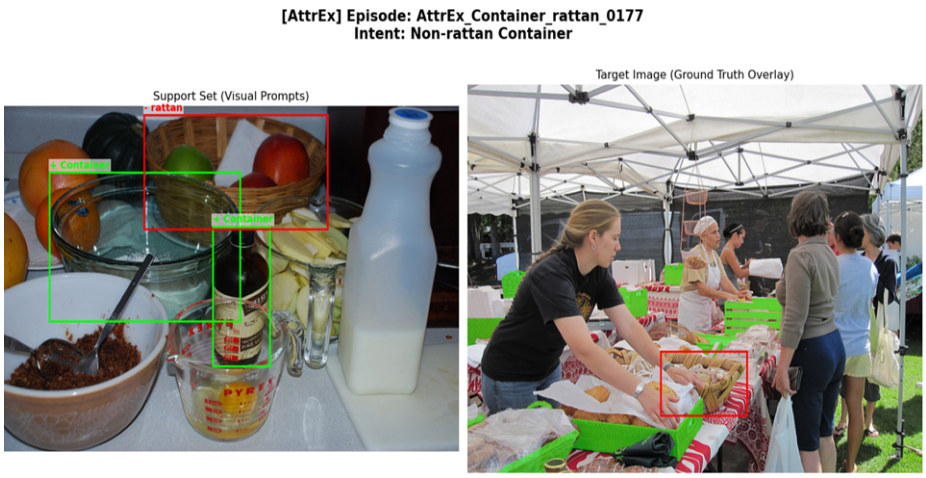



**Fig. 6: Example AttrEx episode.** The intent is to segment non-rattan containers while excluding rattan containers.

and negative exemplars. For evaluation, target and support images are constructed more strictly to include explicit excluded-attribute distractors. We also require the positive prompts to provide diversity either across object categories or across non-excluded attributes. This prevents the task from collapsing into a single-instance matching problem.

### 2.7 CASE-NegStress

CASE-NegStress is a dedicated 1-shot diagnostic subset that fixes the query image, target mask, positive support, and candidate negative pool while varying only the number of negative prompts. The 0-Neg condition denotes positive-only prompting on this same NegStress subset rather than the full CASE. The purpose is not to define a new task family, but to stress-test whether the model's predictions change appropriately when the negative evidence increases. A robust positive-negative visual prompt segmentation model should preserve regions supported by the positive exemplars while suppressing regions that match the negative exemplars. In contrast, a model that ignores negative prompts will produce similar predictions under the original CASE setting and the NegStress setting. Therefore, CASE-NegStress helps distinguish genuine negative-prompt reasoning from positive-only support matching.

# 3 Implementation Details

## 3.1 Shared-canvas Formulation

CPA follows the shared-canvas formulation used in the main paper. Given a $K$-shot support set $\mathcal{S} = \{(I_i^s, \mathcal{B}_i^+, \mathcal{B}_i^-)\}_{i=1}^K$ and a query image $I^q$, we compose them into a single RGB canvas $I^c$ of size $1008 \times 1008$. The positive and negative support boxes are transformed from their original image coordinates into the coordinate system of $I^c$ and are used as SAM 3 box prompts. The model predicts composite canvas logits $Z^c$, from which we crop the query region to obtain $Z^q$ for loss computation and evaluation.

For 1-shot episodes, we use a vertical support-query layout. The support image occupies the top half of the canvas, and the query image occupies the bottom half, where each region is represented as $[x, y, w, h]$:

$$\Omega_{\text{sup}}^{\text{1-shot}} = [0, 0, 1008, 504], \qquad \Omega_{\text{qry}}^{\text{1-shot}} = [0, 504, 1008, 504].$$

For 5-shot episodes, we use an L-shaped layout. The first three support images are placed in the top row, the fourth and fifth support images are placed in the left column, and the query image is placed in the bottom-right region. With canvas size 1008, the support thickness is

$$s = \text{round}(0.33 \times 1008) = 333,$$

and the query region size is

$$q = 1008 - s = 675.$$

Therefore, the 5-shot query region is

$$\Omega_{\text{qry}}^{\text{5-shot}} = [333, 333, 675, 675].$$

The top support row consists of three regions of size $336 \times 333$, and the left support column consists of two regions of sizes $333 \times 337$ and $333 \times 338$, respectively. This layout keeps all five support examples visible while preserving a large square query region. Figure 7 illustrates both layouts.

All support boxes are converted into normalized canvas coordinates in $[c_x, c_y, w, h]$ format. Under positive-negative prompting, both $\mathcal{B}^+$ and $\mathcal{B}^-$ are provided to the model. Under positive-only prompting, only $\mathcal{B}^+$ is retained.

## 3.2 LoRA Configuration

Let $f$ denote the original frozen SAM 3 model and $f_\theta$ denote the CPA-adapted model. We keep all pretrained SAM 3 parameters frozen and optimize only the LoRA parameters $\theta$. LoRA adapters are inserted into transformer encoder and decoder layers. In particular, we apply LoRA to modules whose names end with `q_proj`, `k_proj`, `v_proj`, `out_proj`, `linear1`, or `linear2`.

Unless otherwise specified, we use LoRA rank $r = 16$, scaling factor $\alpha = 32$, dropout 0.05, and no bias adaptation. This results in 1.13M trainable parameters. The original SAM 3 parameters remain frozen throughout training, so $f_\theta$ differs from $f$ only through the inserted LoRA updates.

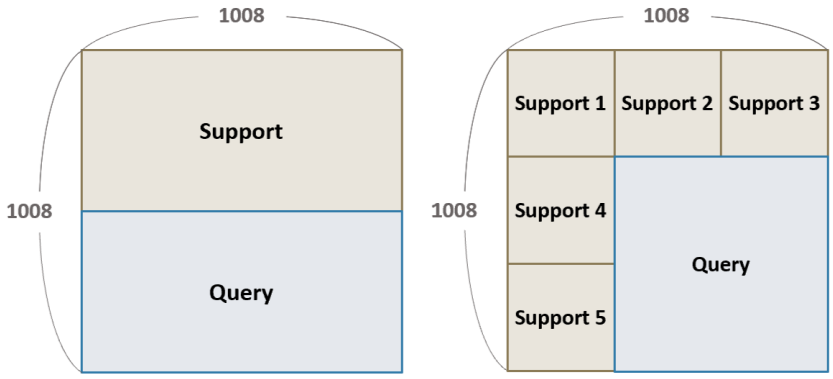



**Fig. 7: Shared-canvas layouts used by CPA.** Left: the 1-shot setting uses a vertical layout, where the support image occupies the top half of the $1008 \times 1008$ canvas and the query image occupies the bottom half. Right: the 5-shot setting uses an L-shaped layout, where three support images are placed in the top row, two support images are placed in the left column, and the query image occupies the bottom-right region.

### 3.3 Training Objective

CPA is optimized under two prompting conditions, following the notation in the main paper. In the positive-negative condition, the adapted model $f_\theta$ receives both positive and negative boxes and predicts query logits $Z^q_{\theta,\pm}$:

$$\mathcal{B}^+, \mathcal{B}^- \to f_\theta \to Z^q_{\theta,\pm}.$$

The prediction is supervised by the CASE ground-truth query mask $M^q$:

$$\mathcal{L}^{\pm}_{\text{svps}} = \mathcal{L}_{\text{mask}} \left( Z^q_{\theta,\pm}, M^q \right).$$

In the positive-only preservation condition, the adapted model $f_\theta$ and the original frozen model $f$ are evaluated under the same positive-only input. The adapted model produces $Z^q_{\theta,+}$, while the frozen model produces $Z^q_{f,+}$:

$$\mathcal{B}^+ \to f_\theta \to Z^q_{\theta,+}, \qquad \mathcal{B}^+ \to f \to Z^q_{f,+}.$$

The preservation loss penalizes the discrepancy between their probability maps:

$$\mathcal{L}^{+}_{\text{pres}} = \left\| \sigma(Z^q_{\theta,+}) - \sigma(Z^q_{f,+}) \right\|_2^2,$$

where $\sigma(\cdot)$ denotes the sigmoid function.

The final training objective is

$$\mathcal{L} = \mathcal{L}^{\pm}_{\text{svps}} + \lambda_{\text{pres}} \mathcal{L}^{+}_{\text{pres}}.$$

**Table 6: CPA training and optimization settings.**

| Component | Setting |
|---|---|
| Base model | SAM 3 |
| Frozen model notation | $f$ |
| Adapted model notation | $f_\theta$ |
| Input canvas size | $1008 \times 1008$ |
| 1-shot layout | Vertical support-query split |
| 5-shot layout | L-shaped support-query canvas |
| Prompt type | Support-image bounding boxes |
| Positive-negative condition | $\mathcal{B}^+$ and $\mathcal{B}^-$ |
| Positive-only condition | $\mathcal{B}^+$ only |
| Trainable modules | LoRA adapters |
| Frozen modules | All pretrained SAM 3 weights |
| LoRA target modules | `q_proj`, `k_proj`, `v_proj`, `out_proj`, `linear1`, `linear2` |
| LoRA rank | $r = 16$ |
| LoRA scaling factor | $\alpha = 32$ |
| LoRA dropout | 0.05 |
| Trainable parameters | 1.13M |
| Optimizer | AdamW |
| Learning rate | $1 \times 10^{-4}$ |
| Weight decay | $1 \times 10^{-4}$ |
| Random seed | 123 |
| Mixed precision | bfloat16 autocast on CUDA |
| Gradient clipping | Max norm 1.0 |
| Preservation weight | $\lambda_{\text{pres}} = 1.0$ |

We set $\lambda_{\text{pres}} = 1.0$ unless otherwise specified.

In implementation, $\mathcal{L}_{\text{mask}}$ combines sigmoid focal loss and Dice loss:

$$\mathcal{L}_{\text{mask}} = 20 \cdot \mathcal{L}_{\text{focal}} + \mathcal{L}_{\text{dice}},$$

where the sigmoid focal loss uses $\alpha = 0.25$ and $\gamma = 2.0$. When SAM 3 returns multiple mask candidates, we aggregate the candidate logits by taking their element-wise maximum before computing the losses.

### 3.4 Optimization Details

We train CPA on the mixed CASE training set. The dataloader shuffles candidate training episodes, and support examples are sampled according to the requested shot setting. For 1-shot training, one support image is placed in the vertical layout. For 5-shot training, five support images are placed in the L-shaped layout.

We use AdamW with learning rate $1 \times 10^{-4}$ and weight decay $1 \times 10^{-4}$. The default training configuration uses a batch size of 1, two dataloader workers, 10 epochs, a random seed of 123, and mixed precision with bfloat16 autocasting on CUDA. Gradients are clipped with a maximum norm of 1.0. LoRA checkpoints are saved after each epoch.

### 3.5 Inference and Evaluation Protocol

At inference time, we use the same shared-canvas formulation as in training. Given $\mathcal{S}$ and $I^q$, we compose the canvas $I^c$, transform the support boxes into

**Table 7: Split-wise FSSAM results on CASE.** FSSAM is evaluated using the publicly released split-specific checkpoints. The main paper reports the average over four splits.

| Shot | Task | Split 0 | Split 1 | Split 2 | Split 3 | Avg. |
|---|---|---|---|---|---|---|
| 1-shot | FineIso | 40.75 | 37.74 | 38.25 | 39.15 | 38.97 |
| | SuperEx | 27.88 | 28.80 | 29.19 | 28.35 | 28.55 |
| | Union | 36.89 | 34.92 | 35.96 | 35.80 | 35.89 |
| | AttrIso | 18.69 | 13.90 | 15.04 | 15.84 | 15.87 |
| | AttrUnion | 15.04 | 15.25 | 13.90 | 15.13 | 14.83 |
| | AttrEx | 17.25 | 17.16 | 17.31 | 17.40 | 17.28 |
| 5-shot | FineIso | 54.84 | 54.52 | 55.72 | 55.45 | 55.13 |
| | SuperEx | 35.36 | 38.74 | 37.07 | 37.26 | 37.11 |
| | Union | 41.85 | 46.98 | 44.26 | 44.94 | 44.51 |
| | AttrIso | 19.17 | 16.40 | 18.18 | 17.71 | 17.87 |
| | AttrUnion | 19.36 | 21.52 | 20.09 | 21.49 | 20.61 |
| | AttrEx | 18.10 | 17.46 | 17.57 | 17.90 | 17.76 |

the canvas coordinate system, and apply $f_\theta$ to obtain composite logits $Z^c$. We crop the query region to obtain $Z^q$ and derive the binary query prediction as

$$\hat{M}^q = \mathbf{1}[\sigma(Z^q) > \tau],$$

where $\sigma(\cdot)$ denotes the sigmoid function and $\tau$ is the mask threshold. We use $\tau = 0.5$ for all reported results. We then resize the query prediction to the original query resolution and compute mIoU against the ground-truth query mask $M^q$.

For CASE, the fixed evaluation episode provides the support images and the positive/negative support boxes. We use the vertical layout for 1-shot evaluation and the L-shaped layout for 5-shot evaluation. For standard FSS benchmarks, we follow the conventional positive-only setting: support masks are converted into tight bounding boxes and used as positive visual prompts, and no negative prompts are provided. For COCO-$20^i$, the CASE-trained CPA checkpoints are evaluated directly on all four folds without COCO-$20^i$ retraining or fold-specific tuning. This evaluation measures CPA's positive-only FSS behavior after adaptation to CASE rather than assuming that the original behavior is fully preserved.

### 3.6 Baseline Adaptation

For SAM-based FSS baselines that do not natively support negative visual exemplars, we adapt the CASE inputs to their supported prompt formats. GF-SAM and FSSAM use positive support masks derived from the positive boxes, while SANSA uses its native positive-box prompt interface. Since these methods were designed for conventional positive-only FSS, their CASE results should be interpreted as positive-only reference baselines rather than fully specified SVPS.

For the main-paper comparison, we use the publicly released checkpoints provided by the authors' official repositories and do not fine-tune these baselines on CASE. A separate same-data SANSA control is reported below to isolate the effect of in-domain CASE exposure. For methods with a single released checkpoint, we report the result obtained from that checkpoint. For FSSAM, the

**Table 8: Full aggregated CASE results across concept dimensions.** Category and Attribute correspond to the source concept type, while Exclusion, Isolation, and Union correspond to the compositional operation required by the task.

| Method | Prompt | Category | Attribute | Exclusion | Isolation | Union | **Mean** |
|---|---|---|---|---|---|---|---|
| | | | **1-shot** | | | | |
| GF-SAM [6] | Pos Only | 31.24 | 14.81 | 20.80 | 26.52 | 21.75 | 23.03 |
| FSSAM [5] | Pos Only | 34.47 | 15.99 | 22.92 | 27.42 | 25.36 | 25.23 |
| SANSA [1] | Pos Only | 36.81 | 18.40 | 21.30 | 38.07 | 23.45 | 27.61 |
| FSS-SAM3 [4] | Pos Only | 42.82 | 18.40 | 24.43 | 38.03 | 29.38 | 30.61 |
| FSS-SAM3 [4] | Pos+Neg | 25.72 | 11.52 | 21.64 | 13.38 | 20.85 | 18.62 |
| CPA (Ours) | Pos Only | 37.26 | 19.70 | 23.67 | 35.55 | 26.24 | 28.48 |
| **CPA (Ours)** | **Pos+Neg** | **48.63** | **23.04** | **29.55** | **42.72** | **35.23** | **35.83** |
| | | | **5-shot** | | | | |
| GF-SAM [6] | Pos Only | 43.47 | 21.18 | 28.24 | 36.10 | 32.64 | 32.32 |
| FSSAM [5] | Pos Only | 45.58 | 18.75 | 27.43 | 36.50 | 32.56 | 32.17 |
| SANSA [1] | Pos Only | 44.27 | 22.52 | 26.41 | 43.40 | 30.38 | 33.40 |
| FSS-SAM3 [4] | Pos Only | 51.72 | 24.48 | 31.09 | 43.80 | 39.43 | 38.10 |
| FSS-SAM3 [4] | Pos+Neg | 46.03 | 22.26 | 30.25 | 36.16 | 36.03 | 34.14 |
| CPA (Ours) | Pos Only | 52.44 | 26.16 | 35.84 | 44.03 | 38.04 | 39.30 |
| **CPA (Ours)** | **Pos+Neg** | **57.15** | **27.14** | **37.74** | **46.82** | **41.88** | **42.15** |

official release provides split-specific checkpoints. We therefore evaluate FSSAM on each of the four splits and report the average across splits in the main paper. The reported FSSAM average is computed at the metric level over the four split-specific evaluations, not by ensembling predictions from multiple checkpoints. The split-wise FSSAM results are shown in Table 7.

## 4 Additional Experimental and Qualitative Results

### 4.1 Full Concept-dimension Breakdown on CASE

Tab. 8 provides the full concept-dimension breakdown corresponding to the aggregated results in the main paper. Category and Attribute summarize the source type of the target concept. Exclusion, Isolation, and Union summarize the operation required to define the target set.

The results show that negative visual exemplars are useful beyond category-level matching. CPA improves performance on the exclusion and isolation dimensions, indicating that negative prompts help reject explicitly excluded categories or attributes and sharpen distinctions among related objects. Union-style concepts also benefit, suggesting that negative exemplars help constrain broad or compositional target groups. Attribute-level dimensions remain more challenging, consistent with the observation that color and material cues are often subtle and context-dependent.

### 4.2 Additional Quantitative Analyses

**Table 9: 1-shot CASE precision–recall analysis.** All methods use the same evaluation episodes and $\tau = 0.5$.

| Model | Prompt | mIoU | Precision | Recall | F1 |
|---|---|---|---|---|---|
| FSS-SAM3 | Pos | 30.61 | 64.10 | 36.21 | 46.28 |
| CPA (Ours) | Pos | 28.48 | 35.12 | **58.02** | 43.75 |
| FSS-SAM3 | Pos+Neg | 18.62 | **68.29** | 20.66 | 31.72 |
| CPA (Ours) | Pos+Neg | **35.83** | 63.32 | 55.72 | **59.28** |

**Table 10: In-domain training control on CASE** (mIoU).

| Method | CASE FT | Prompt | 1-shot | 5-shot |
|---|---|---|---|---|
| SANSA | No | Pos | 27.61 | 33.40 |
| SANSA | Yes | Pos | 30.01 | 36.27 |
| CPA (Ours) | Yes | Pos+Neg | **35.83** | **42.15** |

**Precision–recall behavior under positive and negative prompting.** CPA is adapted to infer concepts jointly from positive and negative exemplars. Consequently, under positive-only inference it retains more candidate regions than FSS-SAM3, yielding higher recall but lower precision. When negative exemplars are provided, CPA recovers precision while retaining substantially higher recall, producing higher F1 and mIoU than FSS-SAM3 under Pos+Neg prompting.

**Same-data training control.** To separate the benefit of CASE training data from the ability to use negative exemplars, we fine-tune SANSA on the same CASE training set while retaining its native positive-only prompting interface, which does not support negative visual exemplars. In-domain training improves SANSA, but CPA remains $+5.82/+5.88$ mIoU higher in 1-/5-shot, showing that CASE exposure alone is insufficient to explain CPA's gains.

**SAM 3 text-prompt controls.** SAM 3 supports text prompts, but exclusion intent is not reliably expressed by the tested text formulations. On SuperEx, the reviewer-inspired exclusion prompt "[supercategory] except [excluded]" obtains 5.79 mIoU as a text-only prompt. Combining text with visual prompts remains below the corresponding visual-only FSS-SAM3 results in the main paper (28.80/39.07 mIoU in 1-/5-shot).

**Human validation of CASE.** To quantify benchmark quality and the PACO ambiguity discussed in the failure analysis, two annotators independently reviewed 300 fixed 1-shot evaluation episodes (50 per task family) using four 1–5 criteria: intent quality, positive-exemplar consistency, negative-exemplar consistency, and target-mask consistency. The four mean scores are 4.22/4.38/4.54/4.10, with medians of 5 for all criteria. Across criteria, 84.7–86.7% of paired ratings are within one point, with linear-weighted $\kappa = 0.42$–$0.57$. PACO-derived attribute episodes are more ambiguous than COCO-derived category episodes, especially

**Table 11: SAM 3 text controls on SuperEx** (mIoU). The two blocks are separate controls and are not row-wise paired.

| Prompt Type | 1-shot | 5-shot |
|---|---|---|
| Text only | 5.79 | 5.79 |
| Text + Pos Visual | 14.26 | 26.03 |
| Text + Pos+Neg Visual | 13.58 | 28.05 |

| Text Formulation | SuperEx |
|---|---|
| "[supercategory]" | 26.28 |
| "[supercategory] except [excluded]" | 5.79 |
| "[supercategory] not [excluded]" | 2.60 |

**Table 12: Human validation of CASE.** Agreement is computed over 300 paired episode ratings.

| Criterion | Mean | Within $\pm 1$ | Linear $\kappa$ |
|---|---|---|---|
| Intent quality | 4.22 | 84.7% | 0.453 |
| Positive consistency | 4.38 | 85.3% | 0.490 |
| Negative consistency | 4.54 | 86.0% | 0.417 |
| Target consistency | 4.10 | 86.7% | 0.570 |

for target-mask consistency (3.57 vs. 4.63), quantitatively supporting the limitation discussed below.

### 4.3 Additional Qualitative Results

This section provides additional qualitative comparisons complementing the main paper. Unless otherwise noted, green boxes on support images denote positive visual prompts and red boxes denote negative visual prompts. On query images, yellow contours denote ground-truth masks and blue overlays denote model predictions. We compare the frozen SAM 3 model $f$ and the CPA-adapted model $f_\theta$ under both positive-only and positive-negative prompting. In the support-prompt column, the support regions follow the same shared-canvas layout used for inference, while the query region is grayed out.

Figs. 8 and 9 show representative 1-shot examples across category-level and attribute-level CASE tasks. The examples illustrate that positive-only prompting is often insufficient for specified concepts, since the model may segment visually related distractors together with the intended target or follow object-category priors instead of the specified attribute. Negative prompts are necessary to specify what should be rejected, but frozen SAM 3 does not consistently use them as concept-level constraints. CPA provides more stable predictions by adapting the model to interpret positive and negative visual exemplars jointly.

Fig. 10 provides 5-shot qualitative examples. Compared with the 1-shot setting, multiple support images provide broader visual evidence about the intended target concept. However, simply increasing the number of positive exemplars does not fully resolve exclusion, isolation, union, or attribute ambiguity. The positive-negative CPA predictions are generally more consistent with the target masks, indicating that the model benefits from both support diversity and explicit negative evidence.

**Table 13: Summary of common failure modes in CASE.**

| Failure mode | Description | Typical cause |
|---|---|---|
| Attribute ambiguity | The model fails to distinguish visually subtle colors or materials, such as glossy plastic versus glass or dark brown versus black. | PACO attributes can be visually subtle, lighting-dependent, or partially visible. |
| Over-exclusion | Negative exemplars suppress valid foreground regions together with distractors. | The negative prompt may share shape, texture, or context with the target concept. |
| Under-exclusion | Distractor instances matching the negative exemplars remain in the prediction. | The model relies on positive similarity and does not fully use negative evidence. |
| Category-attribute entanglement | The model follows object category more strongly than the specified attribute. | Foundation segmentation models retain object-centric priors from pretraining. |
| Small or occluded objects | The model misses small, partially visible, or heavily occluded target instances. | Shared-canvas resizing reduces fine details, and support-query matching becomes less reliable. |
| Ambiguous grouping | The model inconsistently handles broad union concepts or semantic object groups. | Some manually defined groups reflect functional relations that are not always visually apparent. |

## 5 Failure Cases and Limitations

Although CPA improves the use of positive and negative visual exemplars, several limitations remain. Fig. 11 and Tab. 13 summarize representative failure modes. Attribute-level tasks are particularly challenging. Color and material labels may depend on lighting, viewpoint, texture, and object state. Some materials are also visually confounded with object categories; for example, glass may be correlated with bottles or cups, while fabric may be correlated with furniture or clothing. As a result, the model may segment objects that match the category prior rather than the specified attribute.

Fig. 11 illustrates three representative limitations. First, attribute-level tasks remain vulnerable to category–attribute entanglement: the model may segment objects that match a strong object-category prior, such as containers, rather than strictly following the specified attribute constraint. Second, some PACO attributes are visually ambiguous, especially for color and material labels affected by lighting, texture, viewpoint, or annotation convention. Third, negative exemplars may still lead to under-exclusion when the excluded category remains visually salient in the query image.

Negative prompts can also introduce failure modes. When positive and negative exemplars are visually similar, the model may over-exclude valid target regions. Conversely, when negative exemplars are visually diverse or weakly related to the query distractors, the model may under-exclude and behave similarly to positive-only FSS. These errors indicate that positive-negative visual prompting requires not only recognizing individual support instances, but also inferring the intended relation between inclusion and exclusion evidence.

CASE itself is limited by the annotation spaces of COCO and PACO. COCO provides reliable category-level instance masks but does not contain fine-grained attributes. PACO provides attribute annotations, but the available color and

material labels are incomplete and sometimes ambiguous. Union concepts are manually defined and capture only a subset of possible high-level semantic groupings. Extending CASE to richer attributes, open-vocabulary concepts, and more diverse real-world domains is an important direction for future work.

Finally, CPA is designed as a lightweight and conservative adaptation of SAM 3. The preservation regularization mitigates positive-only FSS degradation, but does not fully preserve the frozen model in the 1-shot setting; it also introduces an adaptation–preservation trade-off. Stronger adaptation may improve CASE performance further, but could degrade the original ability of SAM 3. Exploring stronger architectures or training objectives that improve SVPS without sacrificing positive-only FSS remains future work.

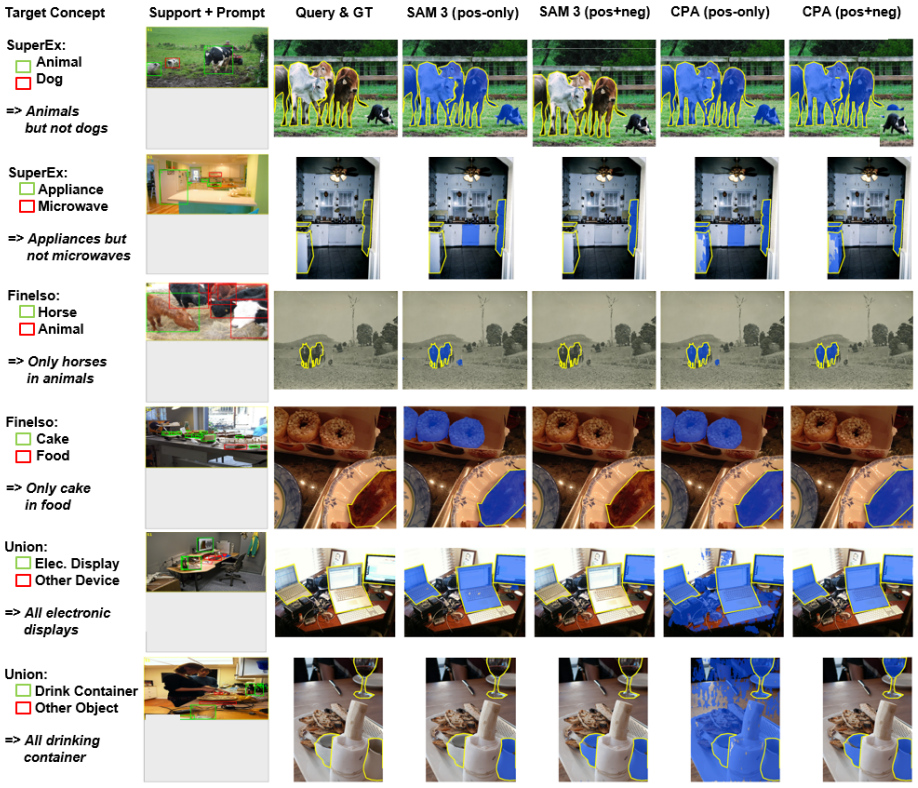



**Fig. 8: Additional 1-shot qualitative comparisons on category-level CASE tasks.** The figure includes two examples from each category-level task family: SuperEx, FineIso, and Union. In the support-prompt column, the upper region shows the 1-shot support image with positive and negative box prompts, while the lower gray region indicates that the query image is omitted from the prompt visualization. Frozen SAM 3 with positive-only prompts often follows visual similarity without resolving the intended exclusion, isolation, or union constraint. Directly adding negative prompts to frozen SAM 3 is unreliable, either having little effect or suppressing valid target regions. CPA with positive-only prompts produces cleaner positive-prompt responses, while CPA with positive-negative prompts better matches the concept jointly specified by positive and negative exemplars.

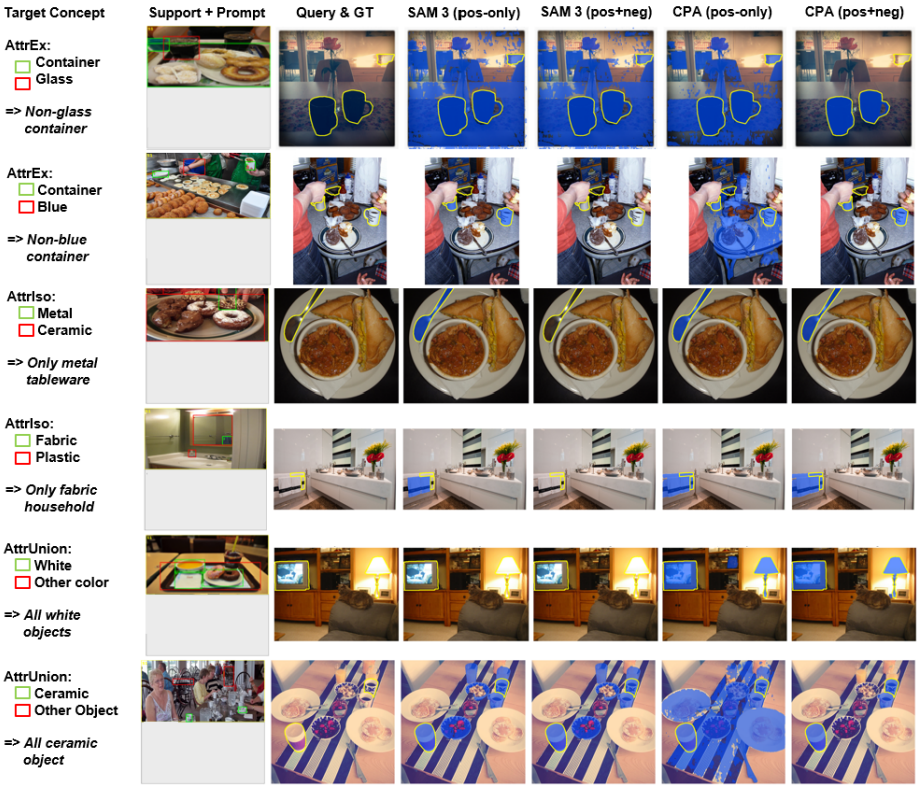



**Fig. 9: Additional 1-shot qualitative comparisons on attribute-level CASE tasks.** The figure includes two examples from each attribute-level task family: AttrEx, AttrIso, and AttrUnion. These examples highlight the difficulty of specifying targets by color or material cues rather than object category alone. Frozen SAM 3 tends to rely on category-level or appearance-level similarity and may fail to respect the intended attribute constraint. CPA improves the use of positive and negative attribute exemplars, although attribute-level concepts remain more challenging than category-level concepts due to subtle visual cues and context-dependent material appearance.

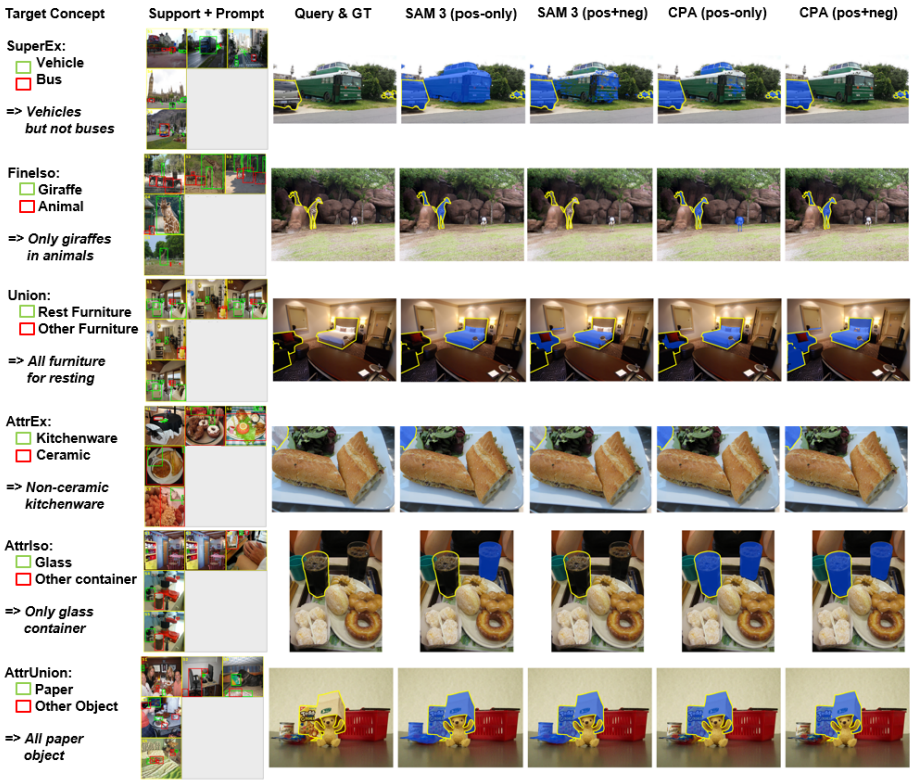



**Fig. 10: Additional 5-shot qualitative examples on CASE.** The figure includes one example from each of the six task families. The support-prompt column follows the L-shaped 5-shot shared-canvas layout, where the five support images occupy the top row and left column, and the lower-right gray region indicates that the query image is omitted from the prompt visualization. Multiple support exemplars provide more evidence about intra-concept variation, but the target concept can remain ambiguous without properly using negative exemplars. CPA uses the expanded support set more effectively under positive-negative prompting.

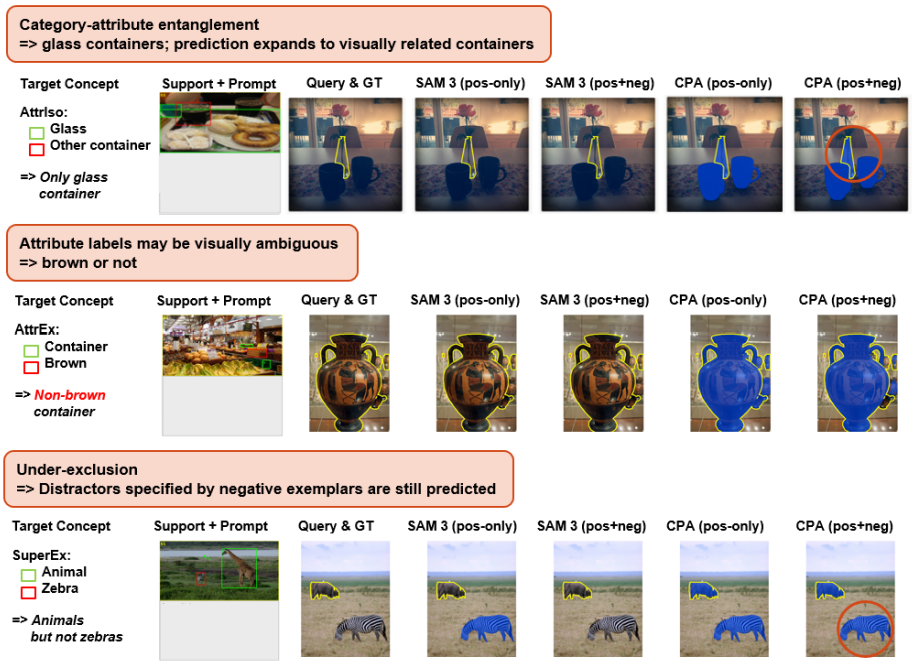



**Fig. 11: Representative failure cases.** Yellow contours denote ground-truth masks and blue overlays denote model predictions. Red circles highlight erroneous regions. Top: in attribute-isolation cases, the adapted model may follow object-category priors and expand from the specified attribute target to visually related containers. Middle: attribute annotations can be visually ambiguous; for example, the distinction between brown and non-brown containers may depend on lighting, texture, and annotation convention. Bottom: negative exemplars do not always fully suppress distractors, leading to under-exclusion when excluded categories remain visually salient in the query image.

## References

1. Cuttano, C., Trivigno, G., Averta, G., Masone, C.: Sansa: Unleashing the hidden semantics in sam2 for few-shot segmentation. In: NeurIPS (2025)
2. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Doll'ar, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: ECCV. pp. 740–755. Springer (2014)
3. Ramanathan, V., Kalia, A., Petrovic, V., Wen, Y., Zheng, B., Guo, B., Wang, R., Marquez, A., Kovvuri, R., Kadian, A., Mousavi, A., Song, Y., Dubey, A., Mahajan, D.: Paco: Parts and attributes of common objects. In: CVPR. pp. 7141–7151 (2023)
4. Tsai, Y.J., Lin, Y.Y., Wang, C.Y.: Few-shot semantic segmentation meets sam3. arXiv preprint arXiv:2604.05433 (2026)
5. Xu, Q., Zhu, L., Liu, X., Lin, G., Long, C., Li, Z., Zhao, R.: Unlocking the power of sam 2 for few-shot segmentation. In: ICML (2025)
6. Zhang, A., Gao, G., Jiao, J., Liu, C., Wei, Y.: Bridge the points: Graph-based few-shot segment anything semantically. In: NeurIPS. vol. 37, pp. 33232–33261 (2024)