Segment Anything without Supervision

XuDong Wang **Jingfeng Yang** **Trevor Darrell**
UC Berkeley
code: https://github.com/frank-xwang/UnSAM

Abstract

The Segmentation Anything Model (SAM) requires labor-intensive data labeling. We present Unsupervised SAM (UnSAM) for promptable and automatic whole-image segmentation that does not require human annotations. UnSAM utilizes a divide-and-conquer strategy to "discover" the hierarchical structure of visual scenes. We first leverage top-down clustering methods to partition an unlabeled image into instance/semantic level segments. For all pixels within a segment, a bottom-up clustering method is employed to iteratively merge them into larger groups, thereby forming a hierarchical structure. These unsupervised multi-granular masks are then utilized to supervise model training. Evaluated across seven popular datasets, UnSAM achieves competitive results with the supervised counterpart SAM, and surpasses the previous state-of-the-art in unsupervised segmentation by 11% in terms of AR. Moreover, we show that supervised SAM can also benefit from our self-supervised labels. By integrating our unsupervised pseudo masks into SA-1B's ground-truth masks and training UnSAM with only 1% of SA-1B, a lightly semi-supervised UnSAM can often segment entities overlooked by supervised SAM, exceeding SAM's AR by over 6.7% and AP by 3.9% on SA-1B.

1 Introduction

Trained on massive unlabeled data using self-supervised learning methods, Large Language Models (LLMs) [5, 34, 33, 46, 2, 19] in natural language processing have revolutionized our world and redefined human-computer interactions. In the domain of computer vision, the recent introduction of the Segment Anything Model (SAM) [21] has dramatically transformed the field with its exceptional ability to handle diverse image segmentation tasks. However, the need for comprehensive manual labeling of training data—over 20 minutes per image [21]—limits SAM from following the scaling laws that benefit LLMs [20]. As a result, despite SA-1B [21] being the most extensive segmentation dataset available, it contains only about 11 million images. Moreover, human-annotated data often introduces significant biases based on the annotators' perceptions of "what constitutes an instance", which frequently leads to the oversight of small entities within the images.

This challenge raises a crucial question addressed in this paper: *Can we "segment anything" without supervision?* In response, we present **UnSAM**, an innovative unsupervised learning method capable of performing both interactive and whole-image segmentation without the need for supervision.

How can we achieve fine-grained and multi-granular segmentation masks comparable to those in SA-1B [21] without supervision? Insights from neuroscience suggest that the human visual system exploits the structure of visual scenes by decomposing dynamic scenes into simpler parts or motions. This perception of hierarchically organized structures implies a powerful "divide-and-conquer" strategy for parsing complex scenes [4, 27]. Drawing inspiration from this, we introduce a divide-and-conquer approach designed to generate hierarchical image segmentation results directly from raw, unlabeled images. The divide-and-conquer approach is a crucial element of UnSAM, enabling it to effectively parse and segment images at multiple levels of granularity.

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

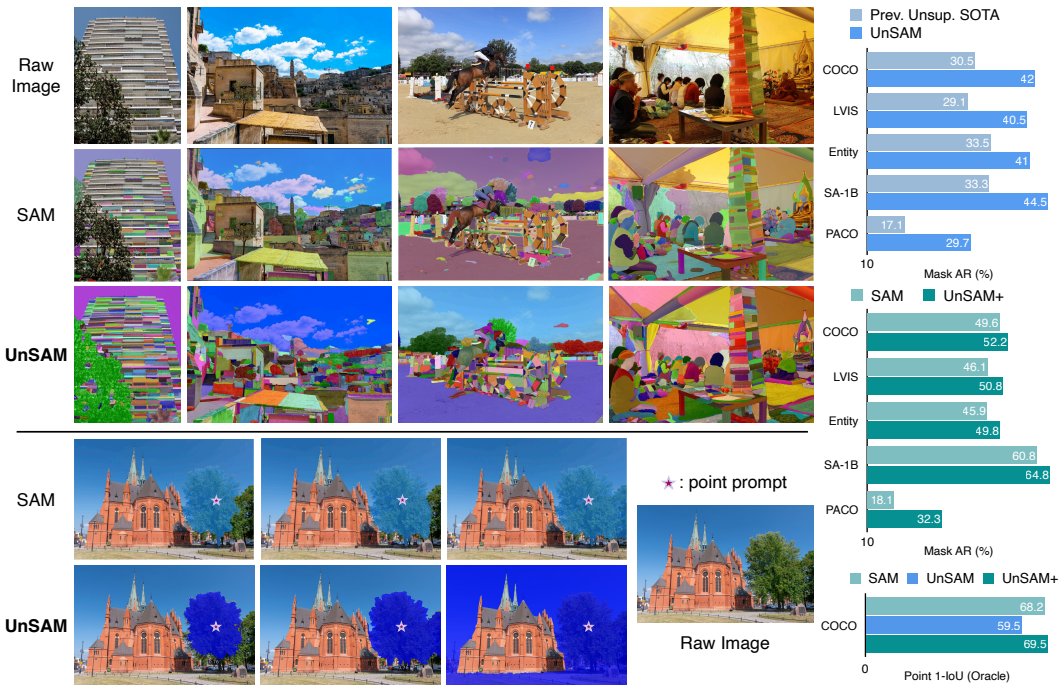

Figure 1: UnSAM significantly surpasses the performance of the previous SOTA methods in unsupervised segmentation, and delivers impressive whole image and promptable segmentation results, rivaling the performance of the supervised SAM [21]. This comparative analysis features our unsupervised UnSAM, the supervised SAM, and an enhanced version, UnSAM+, across a variety of datasets. The top section displays raw images (row 1) alongside whole image segmentation outputs from UnSAM (row 3), and SAM (row 2). The bottom section highlights our promptable segmentation results using a point prompt (*i.e.*, the star mark). The right panel quantitatively compares the performance across models, including metrics like Mask AR (%) and Point IoU.

Our pseudo-mask generation pipeline initiates with a top-down clustering approach (*i.e.*, the divide stage), to extract initial semantic and instance-level masks using a Normalized Cuts-based method CutLER [39, 31]. Subsequently, UnSAM refines these masks using a bottom-up clustering method (*i.e.*, the conquer stage): within each mask, we iteratively merge semantically similar pixels into larger segments based on various similarity thresholds. The resulting masks at different thresholds in the conquer stage, along with the masks produced in the divide stage, create a hierarchical structure. Technically, we can generate a vast range of granularities with minimal extra cost! Furthermore, UnSAM captures more subtle details that pose challenges for human annotators, significantly enriching the granularity and utility of unsupervised segmentation models.

Equipped with these sophisticated multi-granular pseudo masks as "ground-truth" labels, UnSAM is adeptly trained to perform both interactive and automatic whole-image segmentation, demonstrating remarkable versatility across various segmentation scenarios. We have observed that our UnSAM model frequently identifies objects that SAM [21] overlooks, particularly types of objects or parts typically missed by ground-truth annotations of SA-1B [21], such as human ears, animal tails, *etc*.

The capabilities of UnSAM are rigorously tested across seven major whole-entity and part segmentation datasets, *e.g.*, MSCOCO [24], LVIS [15], SA-1B [21], ADE [48], Entity [29], PartImageNet [16] and PACO [30]. As illustrated in Fig. 1, we demonstrate some noteworthy behaviors:

- The performance gap between unsupervised segmentation models and SAM can be significantly reduced: By training on just 1% of SA-1B's unlabeled images with a ResNet50 backbone, UnSAM not only advances the state-of-the-art in unsupervised segmentation by 10% but also achieves comparable performance with the labor-intensive, fully-supervised SAM.
- The supervised SAM can also benefit from our self-supervised labels: integrating our unsupervised pseudo masks with SA-1B's ground-truth data and retraining UnSAM on this combined data enables UnSAM+ to outperform SAM's AR by over 6.7% and AP by 3.9%. We observed that UnSAM and UnSAM+ can often discover entities missed by SAM.

2 Related Works

2.1 Self-supervised Image Segmentation

Recent advances in unsupervised image segmentation [39, 28, 44, 6, 8, 41, 38, 42, 35, 12, 7, 37] have leveraged the emergent segmentation capabilities of self-supervised Vision Transformers (ViT) [8, 14, 17] to "discover" objects within images. Initial efforts, such as TokenCut [44] and LOST [32], have produced semantically meaningful pixel groupings for salient objects by utilizing the class-attention mechanism of self-supervised ViTs. As a representative work in the unsupervised segmentation domain, CutLER [39] introduced a cut-and-learn pipeline for unsupervised object detection and image segmentation. CutLER initially generates high-quality pseudo masks for multiple objects using MaskCut [39], followed by learning a detector on these masks using a loss dropping strategy. Extending this approach, VideoCutLER [40] employs a cut-synthesis-and-learn strategy for segmenting and tracking multiple instances across video frames without supervision. Additionally, SOHES [6] introduced the global-local self-exploration method to cluster image features from high to low cosine similarity, obtaining pseudo masks that cover multiple hierarchical levels.

In contrast, UnSAM introduces a divide-and-conquer pipeline that generates more pseudo masks per image at the same processing speed, but with enhanced quality and broader coverage across hierarchical levels. Furthermore, UnSAM captures more subtle details that pose challenges for human annotators, significantly enriching the granularity and utility of unsupervised segmentation models.

2.2 Promptable Image Segmentation

Tradition segmentation models have focused on predicting masks for all instances or semantic parts within a single image simultaneously. Recently, however, models have begun to interact with users, generating segmentation masks based on user inputs such as points [21, 23, 47, 45, 11], text descriptions [26], or bounding boxes [21]. Moreover, some approaches now frame segmentation tasks within an in-context learning framework [43, 3], utilizing in-context examples to define distinct segmentation tasks. For example, the Segment Anything model [21] can produce masks in a zero-shot manner based on different types of prompts. One limitation of SAM is that it only produces three class-agnostic masks. An extension, Semantic-SAM [23], aims to segment and recognize objects at multiple granularities through a multi-choice learning scheme, allowing each click point to produce masks at multiple levels along with their semantic labels. Nevertheless, both models are supervised and rely on large-scale, human-annotated data, which introduces issues of annotator bias and scalability limitations.

In contrast, our unsupervised UnSAM and lightly semi-supervised UnSAM+ model demonstrate superior performance in the promptable segmentation task, offering a robust alternative to these fully-supervised approaches.

3 Preliminaries

3.1 Cut and Learn (CutLER) and MaskCut

CutLER [39] introduces a cut-and-learn pipeline to precisely segment instances without supervision. The initial phase, known as the cut stage, uses a normalized cut-based method, MaskCut [39], to generate high-quality instance masks given the patch-wise cosine similarity matrix $W_{ij} = \frac{K_i K_j}{|K_i|_2 |K_j|_2}$, where K_i is "key" features of patch i in the last attention layer of unsupervised ViT. To extract multiple instance masks from a single image, MaskCut repeats this operation but adjusts by masking out patches from previously segmented instances in the affinity matrix: $W^t_{ij} = \frac{\left(K_i \sum_{s=1}^{t} M^s_{ij}\right)\left(K_j \sum_{s=1}^{t} M^s_{ij}\right)}{\|K_i\|_2 \|K_j\|_2}$ Subsequently, CutLER's learning stage trains a segmentation/detection model on these pseudo-masks with drop-loss. Please check Appendix A.2 for more details on CutLER.

3.2 Segment Anything Model (SAM) and SA-1B

Segment Anything [21] tackles the promptable segmentation task. At its core lies the Segment Anything Model (SAM), which is capable of producing segmentation masks given user-provided points, boxes, and masks in a zero-shot manner. One significant contribution of SAM is the release of

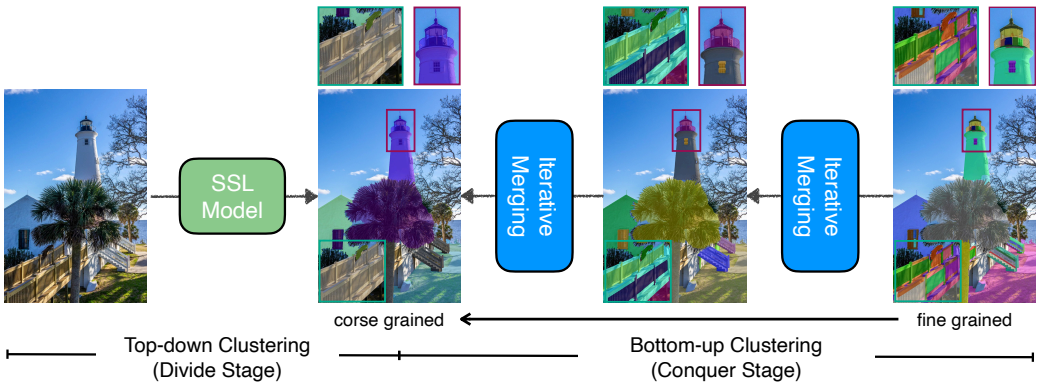

Figure 2: Our divide-and-conquer pipeline for generating the "ground-truth" pseudo masks used for training UnSAM without human supervision begins with a top-down clustering approach (*i.e.*, the divide stage), to extract initial semantic/instance-level masks using a Normalized Cuts [31]-based CutLER [39]. Subsequently, we refine these masks using a bottom-up clustering method (*i.e.*, the conquer stage): within each mask, we iteratively merge semantically similar pixels into larger segments using various similarity thresholds. The resulting masks at different thresholds create a hierarchy. We zoom-in selected regions to visualize details.

the SA-1B dataset [21], which comprises 11M high-resolution images and 1.1 billion segmentation masks, providing a substantial resource for training and evaluating segmentation models. While SAM significantly accelerates the labeling of segmentation masks, annotating an image still requires approximately 14 seconds per mask. Given that each image contains over 100 masks, this equates to more than 30 minutes per image, posing a substantial cost and making it challenging to scale up the training data effectively. For more details on SAM and SA-1B, please check Appendix A.3.

4 UnSAM: Segment Anything without Supervision

4.1 Divide-and-Conquer for Hierarchical Image Segmentation

Our segment anything without supervision model starts by generating pseudo masks that respect the hierarchical structure of visual scenes without supervision. This approach is motivated by the observation that the "divide and conquer" strategy is a fundamental organizational principle employed by the human visual system to efficiently process and analyze the vast complexity of visual information present in natural scenes [4, 27]. Our pseudo-mask generation pipeline **divide-and-conquer**, which is summarized in Alg. 1 and illustrated in Fig. 2, consists of two stages:

Divide stage: we leverage a Normalized Cuts (NCuts)-based method, CutLER [39, 31], to obtain semantic and instance-level masks from unlabeled raw images. CutLER's cut-and-learn pipeline and its MaskCut method are discussed in Sec. 3.1. However, the coarser-granularity masks predicted by CutLER can be noisy. To mitigate this, we filter out masks with a confidence score below a threshold τ. Empirically, salient semantic and instance-level entities typically encompass richer part-level entities (for example, a person has identifiable parts such as legs, arms, and head, whereas a background sky contains few or no sub-level entities). To extract these part-level entities with a hierarchical structure, we employ a conquer phase.

Conquer stage: for each instance-/semantic-level mask discovered in the previous stage, we employ iterative merging [1, 6] to decompose the coarse-grained mask into simpler parts, forming a hierarchical structure.

More specifically, we first crop local patches using the masks we obtained in the divide phase, and bi-linearly interpolate local patches to the resolution of 256×256. We then feed them into DINO pre-trained ViT-B/8 [8] encoder $f(\cdot)$, and extract 'key' features $k_i = f(p_i)$ from the last attention layer as patch-wise features for local patches p_i. Subsequently, the conquer phase employs iterative merging [1, 6] to group patches into larger clusters, with pre-defined cosine similarity thresholds at $\theta \in \{\theta_1, ..., \theta_l\}$, where l is the predefined granularity levels.

Algorithm 1 Divide and Conquer

$I_{\text{resized}} \leftarrow$ input image I resized to 1024×1024
$M \leftarrow \{m : m \in \text{CutLER}(I_{\text{resized}}) \wedge m_{\text{score}} > \tau\}$
for $m \in M$ **do**
  Add m into S_0
  $bbox \leftarrow$ bounding box $[x_1, y_1, x_2, y_2]$ of m
  $I_{\text{local}} \leftarrow I_{\text{resized}}$ cropped by $bbox$, resized to 256×256
  $K \leftarrow \text{DINO}(I_{\text{local}})$
  for $\theta_t \in \theta_l, \ldots, \theta_1$ **do**
    if $t = l$ **then**
      Initialize $k_i^t \leftarrow K_i$, $C_i^t \leftarrow p_i \ \forall i$, $a \leftarrow 1$
      where p_i is corresponding patch of K_i, add p_i into $S_l \ \forall i$
    else
      Initialize $S_t \leftarrow S_{t+1}$, $k_i^t \leftarrow k_i^{t+1}$, $C_i^t \leftarrow C_i^{t+1} \ \forall i$
    end if
    while $a \geq \theta_t$ **do**
      Identify adjacent p_i, p_j with $i, j \leftarrow \underset{i,j}{\text{argmax}} \frac{k_i^t k_j^t}{||k_i^t||_2 ||k_j^t||_2}$, $a \leftarrow \max_{i,j} \frac{k_i^t k_j^t}{||k_i^t||_2 ||k_j^t||_2}$
      Identify cluster C_m^t, C_n^t, where $p_i \in C_m^t, p_j \in C_n^t$
      Remove C_m^t and C_n^t from S_t
      $C^t \leftarrow C_m^t \cup C_n^t$, add C^t into S_t
      $\forall p_z \in C^t, k_z^t \leftarrow \frac{a_m k_i^t + a_n k_j^t}{a_m + a_n}$, where a_m is the size of cluster C_m^t and $p_i \in C_m^t$
    end while
  end for
end for

In iteration t, our method finds two adjacent patches (p_i, p_j) from two separate clusters (C_m^t, C_n^t) with the highest cosine similarity $\frac{k_i^t k_j^t}{||k_i^t||_2 ||k_j^t||_2}$, merges them into one cluster, and updates k_i^t and k_j^t to $\frac{a_m k_i^t + a_n k_j^t}{a_m + a_n}$, where a_m is the number of patches in cluster $C_m^t (p_i \in C_m^t)$. The conquer stage repeats this step until the maximum cosine similarity is less than θ_t, collects all merged clusters as new part-level pseudo masks, and uses smaller threshold θ_{t+1} to iterate again. Each coarse-grained mask discovered in the divide stage can form a hierarchical structure H after the conquer stage:

$$H = \{S_0, S_1, ..., S_t, ..., S_l\}, \text{where } S_t = \{C_1^t, ..., C_{n_t}^t\}, n_i \leq n_j \text{ if } i < j \tag{1}$$

n_t is the number of clusters/masks belonging to granularity level t and $n_0 = 1$.

Mask merging: The new part-level pseudo masks discovered in the conquer stage are added back to the semantic and instance-level masks identified in the divide stage. We then use Non-Maximum Suppression (NMS) to eliminate duplicates. Following previous works in unsupervised image segmentation [39, 28, 6], we also employ off-the-shelf mask refinement methods, such as Conditional Random Fields (CRF) [22] and CascadePSP [10], to further refine the edges of the pseudo masks. Finally, we filter out the post-processed masks that exhibit significant differences in Intersection-over-Union (IoU) before and after refinement.

Preliminary results: The divide-and-conquer pipeline achieves a pseudo mask pool with more entities, a broader range of granularity levels, and superior quality compared to previous work, *e.g.*, CutLER [39], U2Seg [28] and SOHES [6]. As shown in Table 3, its pseudo masks reach 23.9% AR on 1000 randomly selected validation images from the SA-1B dataset [21], representing a 45.7% improvement over the state-of-the-art.

Key distinctions over prior works on pseudo-mask generation: The divide-and-conquer strategy employed by UnSAM sets it apart from previous works:

[39, 28] rely solely on top-down clustering methods, providing only instance and semantic-level masks, and thereby missing the hierarchical structure present in complex images. In contrast, our pipeline captures this hierarchical structure by identifying more fine-grained pixel clusters.

While [6] does incorporate some hierarchical structure through bottom-up clustering with iterative merging, it still misses many fine-grained instances and some large-scale instance masks. Additionally, the iterative merging in [6] focuses on small regions below a certain mask size threshold, primarily to refine noisy small masks, limiting its ability to detect a full range of entity sizes. Our experimental results demonstrate qualitatively and quantitatively superior performance compared to prior works, particularly in producing high-quality, detailed pseudo-masks that better capture the hierarchical complexity of visual scenes.

4.2 Model Learning and Self-Training

Although the pseudo masks generated by our pipeline are qualitatively and quantitatively superior to those from prior works, they can still be somewhat noisy. Our self-supervised pipeline has limitations in identifying certain types of instances. For example, iterative merging sometimes fails to correctly associate disconnected parts of the same entity. To address this, we utilize a self-training strategy to further enhance UnSAM's model performance. UnSAM learns an image segmentation model using the masks discovered by the divide-and-conquer strategy. It has been observed that self-training enables the model to "clean" the pseudo masks and predict masks of higher quality [39]. Once we have prepared the pseudo-masks, UnSAM can be integrated with any arbitrary whole-image or promptable image segmentation models during the model learning or self-training stage.

Whole-image segmentation. We choose the vanilla Masked Attention Mask Transformer (Mask2Former) [9] for simplicity. The key innovation of Mask2Former is the introduction of a masked attention mechanism in the transformer's cross-attention block, defined as $\text{softmax}(M + QK^T)V$, where the attention mask M at feature location (x, y) is given by: $M(x, y) = \begin{cases} 0 & \text{if } M(x, y) = 1 \\ -\infty & \text{otherwise} \end{cases}$. This mechanism constrains attention within the region of the predicted mask. UnSAM is then trained using the following mask prediction loss:

$$\mathcal{L} = \lambda_{\text{ce}}\mathcal{L}_{\text{ce}} + \lambda_{\text{dice}}\mathcal{L}_{\text{dice}} \tag{2}$$

where $\mathcal{L}_{\text{ce}}$ and $\mathcal{L}_{\text{dice}}$ is the cross-entropy and Dice loss, with λ_{ce} and λ_{dice} as their respective weights.

After one round of self-training UnSAM on the pseudo-masks, we perform a second round of self-training by merging high-confidence mask predictions (with a confidence score greater than $\tau_{\text{self-train}}$) as the new 'ground-truth' annotations. To avoid duplication, we filter out ground truth masks that have an IoU greater than 0.5 with the predicted masks.

Promptable Image Segmentation. Similar to SAM [21], our unsupervised SAM can also produce high-quality object masks from input prompts such as points. We utilize Semantic-SAM [23] as the base model for predicting multiple granularity levels of masks from a single click. During the learning process, we randomly sample points within an inner circle (radius $\leq 0.1 \cdot \min(\text{Mask}_{\text{width}}, \text{Mask}_{\text{height}})$) of the mask to simulate user clicks.

4.3 UnSAM+: Improving Supervised SAM with Unsupervised Segmentation

The supervised SAM model's [21] reliance on human-annotated data introduces a significant bias based on the annotator's perception of *'what constitutes an instance'*, frequently missing some entities within the image. In contrast, since our mask generation pipeline does not rely on human supervision, it can often identify valid objects or parts that are overlooked by SA-1B's [21] ground-truth annotations.

Motivated by this observation, we leverage UnSAM to improve the performance of the supervised SAM [21] by implementing a straightforward yet effective strategy: merging SA-1B's ground-truth masks $D_{\text{SA-1B}}$ with our unsupervised segmentation masks D_{UnSAM} based on the IoU, formulated as:

$$D^i_{\text{UnSAM+}} = D^i_{\text{SA-1B}} \cup \{\forall C_m \in D^i_{\text{UnSAM}} \text{ if } \text{IoU}^{\max}(C_m, \forall C_n \in D^i_{\text{SA-1B}}) \leq \tau_{\text{UnSAM+}}\} \tag{3}$$

$\tau_{\text{UnSAM+}}$ is the IoU threshold, $\text{IoU}^{\max}$ is the maximum IoU between C_m and any mask C_n in $D^i_{\text{SA-1B}}$, and $D^i_{\text{SA-1B}}$ and $D^i_{\text{UnSAM+}}$ is the set of SA-1B and unsupervised masks within image i, respectively. We then train UnSAM+ on $D_{\text{UnSAM+}}$ for promptable image segmentation and whole-image segmentation. The fusion approach leverages the strengths of both supervised and unsupervised annotations, addressing the limitations inherent in human-annotated datasets while significantly enriching the diversity and comprehensiveness of the training data. This results in a more robust and generalizable segmentation model UnSAM+, surpassing the performance of SAM.

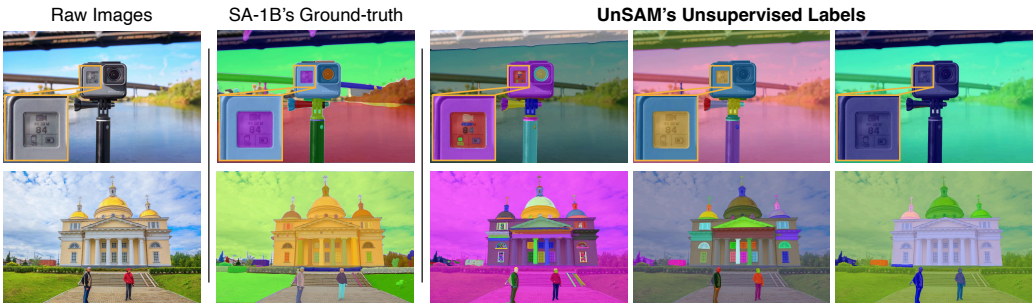

Figure 3: Unsupervised pseudo-masks generated by our divide-and-conquer pipeline not only contain precise masks for coarse-grained instances (column 5), *e.g.*, cameras and persons, but also capture fine-grained parts (column 3), *e.g.*, digits and icons on a tiny camera monitor that are missed by SA-1B's [21] ground-truth labels.

Methods	Backbone (# params)	# images	Avg.	Datasets with Whole Entities					Datasets w/ Parts	
				COCO	LVIS	ADE	Entity	SA-1B	PtIn	PACO
SAM (supervised)	ViT-B/8 (85M)	11M	42.1	49.6	46.1	45.8	45.9	60.8	28.3	18.1
FreeSOLO [41]	RN-101 (45M)	1.3M	7.3	11.6	5.9	7.3	8.0	2.2	13.8	2.4
CutLER [39]	RN-50 (23M)	1.3M	21.8	28.1	20.2	26.3	23.1	17.0	28.7	8.9
SOHES [6]	ViT-B/8 (85M)	0.2M	30.1	30.5	29.1	31.1	33.5	33.3	36.0	17.1
UnSAM	RN-50 (23M)	0.1M	**39.2**	**40.5**	**37.7**	**35.7**	**39.6**	**41.9**	**51.6**	**27.5**
UnSAM	RN-50 (23M)	0.2M	**40.4**	**41.2**	**39.7**	**36.8**	**40.3**	**43.6**	**52.1**	**29.1**
UnSAM	RN-50 (23M)	0.4M	**41.1**	**42.0**	**40.5**	**37.5**	**41.0**	**44.5**	**52.7**	**29.7**
vs. prev. SOTA			**+11.0**	**+11.5**	**+11.4**	**+6.4**	**+7.5**	**+11.2**	**+16.7**	**+12.6**

Table 1: UnSAM achieves the state-of-the-art results on unsupervised image segmentation, using a backbone of ResNet50 and training with only 1% of SA-1B [21] data. We perform a zero-shot evaluation on various image segmentation benchmarks, including whole entity datasets, *e.g.*, COCO and ADE, and part segmentation datasets, *e.g.*, PACO and PartImageNet. The evaluation metric is average recall (AR).

5 Experiments

5.1 Model Training Settings

We provide a brief overview of the model training settings and include more details in Appendix A.1.

Pseudo mask generation. In the divide stage, we set the confidence threshold τ=0.3; in the conquer stage, we choose threshold $\theta_{merge} = [0.6, 0.5, 0.4, 0.3, 0.2, 0.1]$. When merging the pseudo masks with the ground truths for training UnSAM+, we select $\tau_{\text{UnSAM+}} = 0.02$. **Whole-image segmentation.** UnSAM picks DINO [8] pre-trained ResNet-50 [18] as the backbone and Mask2former [9] as the mask decoder. The default learning rate is 5×10^{-5} with a batch size of 16 and a weight decay of 5×10^{-2}. We train the model for 8 epochs. **Promptable segmentation.** UnSAM uses the self-supervised pre-trained Swin-Transformer [25] Tiny model as the backbone, and leverages Semantic-SAM [23] as the base model. We set the number of hierarchy levels to 6, which is also the number of predicted masks UnSAM generates per prompt during inference. One can easily train with a different number of granularity levels as needed. For all experiments, we train UnSAM with 1$\sim$4% unlabeled images from SA-1B dataset [21].

5.2 Evaluation Datasets and Metrics

Whole-image segmentation. To evaluate our model's performance, we test our models on various datasets in a zero-shot manner to evaluate the performance of segmenting entities from all granularity levels. We choose COCO [24], LVIS [15], ADE20K [48], EntitySeg [29], and SA-1B [21] that mainly encompass semantic-/instance-level entities; PartImageNet [16] and PACO [30] that cover part-level entities. The SA-1B test set consists of randomly selected 1000 images not included in our training set. Notably, each dataset only covers entities from certain hierarchical levels and certain pre-defined classes, while our model generates masks from all levels and all classes. Hence, the COCO Average Precision (AP) metric could not reflect our model's authentic performance in segmenting all entities in the open-world. Following prior work [39, 6], we mainly consider Average Recall (AR) to compare with different models.

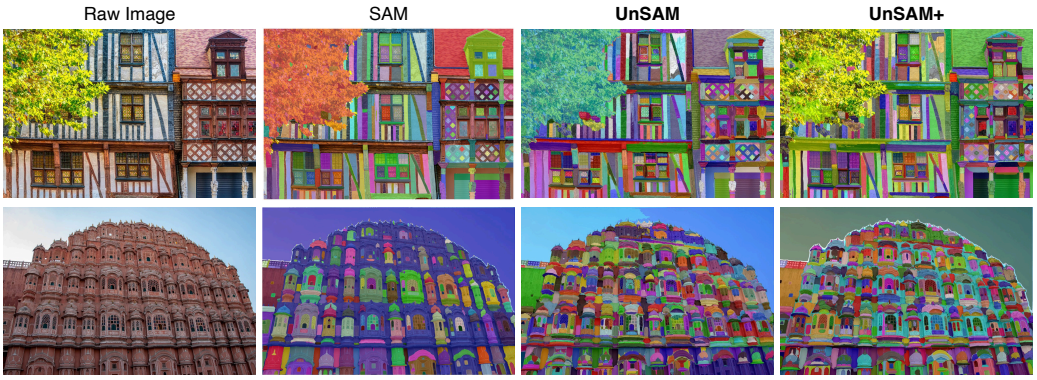

Figure 4: UnSAM has competitive dense object segmentation results compared to the supervised SAM [21].

Methods	Backbone (# params)	Sup. Labels	Unsup. Labels	# images	Avg.	Datasets with Whole Entities					Datasets w/ Parts	
						COCO	LVIS	ADE	Entity	SA-1B	PtIn	PACO
SAM	ViT-B/8 (85M)	✓	✗	11M	42.1	49.6	46.1	**45.8**	45.9	60.8	28.3	18.1
UnSAM	RN-50 (23M)	✗	✓	0.1M	39.2	40.5	37.7	35.7	39.6	41.9	**51.6**	27.5
UnSAM+	RN-50 (23M)	✓	✓	0.1M	**48.8**	**52.2**	**50.8**	45.3	**49.8**	**64.8**	46.0	**32.3**
vs. SAM					**+6.7**	**+2.6**	**+4.7**	-0.5	**+3.9**	**+4.0**	**+17.7**	**+14.2**

Table 2: UnSAM+ can outperform SAM [21] on most experimented benchmarks (including SA-1B [21]), when training UnSAM on 1% of SA-1B with both ground truth masks and our unsupervised labels. This demonstrates that our unsupervised pseudo masks can serve as a powerful add-on to the densely annotated SA-1B masks!

Methods	AR_{1000}	AR_S	AR_M	AR_L
SOHES (CRF [22])	12.0	3.5	9.5	20.7
SOHES (CascadePSP [10])	16.4	6.0	15.8	22.6
UnSAM (CRF [22])	15.3	2.3	11.9	27.7
UnSAM (CascadePSP [10])	**23.9**	**7.9**	**22.4**	**34.0**
vs. prev. SOTA	**+7.5**	**+1.9**	**+6.6**	**+11.4**

Table 3: Evaluation on unsupervised pseudo masks using SA-1B's [21] ground-truth annotations.

Methods	AP	AR_S	AR_M	AR_L	AR_{1000}
SAM	38.9	20.0	59.9	**82.8**	60.8
UnSAM+	**42.8**	**36.2**	**65.9**	76.5	**64.8**
vs. sup. SAM	**+3.9**	**16.2**	**+6.0**	-6.3	**+4.0**

Table 4: Quantitative comparisons between our lightly semi-supervised SAM, UnSAM+, and the fully-supervised SAM [21] on SA-1B [21].

Point-based promptable segmentation. We evaluate our point-based interactive segmentation model on MSCOCO `Val2017` [24]. Following the previous work on promptable image segmentation [21, 23], we pick two metrics for model evaluation MaxIoU and OracleIoU. For each point prompt, UnSAM predicts 6 masks representing different granularity levels. MaxIoU calculates the IoU between the mask with the highest confidence score among 6 masks, whereas OracleIoU picks the highest IoU between 6 predicted masks and the ground truth. For each mask in a test image, we select its center as the point prompt.

5.3 Evaluation Results

Unsupervised pseudo-masks. Unsupervised pseudo-masks generated by our divide-and-conquer pipeline not only contain precise masks for coarse-grained instances, but also capture fine-grained parts that are often missed by SA-1B's [21] ground-truth labels, as shown in Fig. 3.

Whole-image segmentation. Remarkably, UnSAM outperforms the previous state-of-the-art methods across **all** evaluation datasets as summarized in Table 1. UnSAM demonstrates superior performance compared to the SOTA method even when trained with only 1% SA-1B training data and a backbone of ResNet-50 with only 23M parameters, while the SOTA utilizes twice training data and a backbone with nearly four times the parameters. This implies that UnSAM is a lightweight, easier to train, and less data-hungry model with better zero-shot performance in segmenting entities in the open-world as shown in Figs. 4 and 5. On average, UnSAM surpasses the previous SOTA by 11.0% in AR. When evaluated on PartImageNet [16] and PACO [30] benchmarks, UnSAM exceeds the SOTA by 16.6% and 12.6 %, respectively.

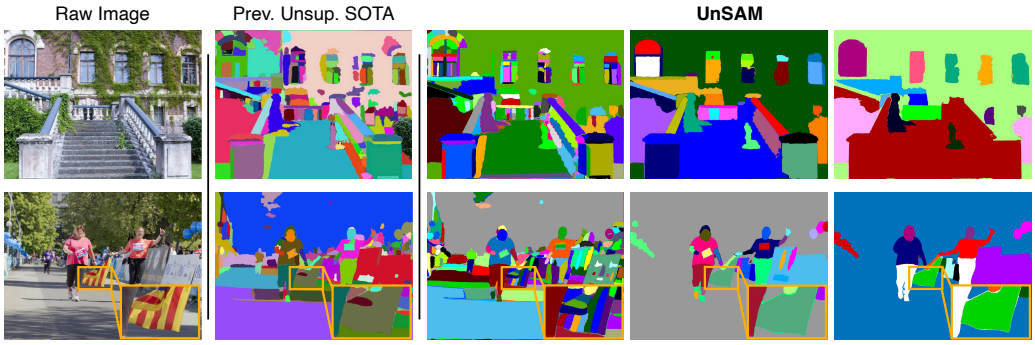

Figure 5: UnSAM not only discovers more fine-grained masks than the previous state-of-the-art unsupervised segmentation method [6], but also provides segmentation masks with a wide range of granularity. We show qualitative comparisons between UnSAM (with 3 levels of granularity) and baseline models on SA-1B [21].

Figure 6: Qualitative comparisons of promptable image segmentation between the fully-supervised SAM [21], our unsupervised UnSAM, and the lightly semi-supervised UnSAM+. Both UnSAM and UnSAM+ consistently deliver high-quality, multi-granular segmentation masks in response to the point prompts (*i.e.*, the star mark).

Methods	Backbone (# params)	Sup. Labels	Unsup. Labels	% of SA-1B	Point (Max) 1-IoU	Point (Oracle) 1-IoU
SAM (B)	ViT-B/8 (85M)	✓	✗	100%	52.1	68.2
UnSAM	Swin-Tiny (25M)	✗	✓	1%	40.3	59.5
UnSAM+	Swin-Tiny (25M)	✓	✓	1%	**52.4**	**69.5**

Table 5: Despite using a backbone that is 3× smaller and being trained on only 1% of SA-1B, our lightly semi-supervised UnSAM+ surpasses the fully-supervised SAM in promptable segmentation task on COCO.

When compared to the supervised SAM [21], UnSAM's AR across all datasets is already very close, with only a 1% difference. On PartImageNet [16] and PACO [30], UnSAM surpasses SAM by 24.4% and 11.6%. This further demonstrates the excellent capability of our divide-and-conquer pipeline in discovering details that human annotators tend to miss.

Furthermore, our UnSAM+, trained with integrated unsupervised pseudo masks and SA-1B [21] ground truth, outperforms SAM's [21] AR by over 6.7% and AP by 3.9% as shown by Table 2 and 4. UnSAM+ demonstrates superior average recall compared to SAM across all evaluation datasets except for ADE20K [48], which is dominated by semantic-level annotations. UnSAM+'s significantly 16.2 % higher AR on small entities further confirms that our pseudo masks can effectively complement the SA-1B datasets with more details it ignores and the UnSAM+ can often discover entities missed by SAM as demonstrated in Fig. 4 and Fig. 7.

Point-based promptable segmentation. As shown in Table 5, UnSAM trained with our pseudo masks achieve 40.3% MaxIoU and 59.5% OracleIoU on COCO. Notably, we train the model with only 1% of the data that SAM [21] uses and a backbone with 4× fewer parameters. Moreover, the UnSAM+ trained with integrated pseudo masks and SA-1B ground truths outperforms SAM on both MaxIoU and OracleIoU with 0.3% and 1.3% respectively. Qualitative results are shown in Fig. 6.

Figure 7: More visualizations on SA-1B [21]. From top to bottom are raw images, segmentation by SAM, segmentation by UnSAM, and segmentation by UnSAM+.

6 Summary

Image segmentation is a fundamental task in computer vision, traditionally relying on intensive human annotations to achieve a detailed understanding of visual scenes. We propose UnSAM, an unsupervised segmentation model that significantly surpasses the performance of previous state-of-the-art methods in unsupervised image segmentation. Additionally, our unsupervised UnSAM model delivers impressive results, rivaling the performance of the cutting-edge supervised SAM, and exceeding it in certain semi-supervised settings.

Acknowledgement. We thank helpful discussions with Jitendra Malik, Cordelia Schmid, Ishan Misra, Xinlei Chen, Xingyi Zhou, Alireza Fathi, Renhao Wang, Stephanie Fu, Qianqian Wang, Baifeng Shi, Max Letian Fu, Tony Long Lian, Songwei Ge, Bowen Cheng and Rohit Girdhar. We thank Shengcao Cao and Hao Zhang for their help in reproducing baseline results. XuDong Wang and Trevor Darrell were funded by DoD including DARPA LwLL and the Berkeley AI Research (BAIR) Commons.

References

[1] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik. Contour detection and hierarchical image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 33(5):898–916, 2010.

[2] J. Bai, S. Bai, Y. Chu, Z. Cui, K. Dang, X. Deng, Y. Fan, W. Ge, Y. Han, F. Huang, et al. Qwen technical report. *arXiv preprint arXiv:2309.16609*, 2023.

[3] Y. Bai, X. Geng, K. Mangalam, A. Bar, A. Yuille, T. Darrell, J. Malik, and A. A. Efros. Sequential modeling enables scalable learning for large vision models. *arXiv preprint arXiv:2312.00785*, 2023.

[4] J. Bill, H. Pailian, S. J. Gershman, and J. Drugowitsch. Hierarchical structure is employed by humans during visual motion perception. *Proceedings of the National Academy of Sciences*, 117(39):24581–24589, 2020.

[5] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[6] S. Cao, J. Gu, J. Kuen, H. Tan, R. Zhang, H. Zhao, A. Nenkova, L. Gui, T. Sun, and Y.-X. Wang. SOHES: Self-supervised open-world hierarchical entity segmentation. In *The Twelfth International Conference on Learning Representations*, 2024.

[7] S. Cao, D. Joshi, L. Gui, and Y.-X. Wang. HASSOD: Hierarchical adaptive self-supervised object detection. In *NeurIPS*, 2023.

[8] M. Caron, H. Touvron, I. Misra, H. Jégou, J. Mairal, P. Bojanowski, and A. Joulin. Emerging properties in self-supervised vision transformers. In *Proceedings of the International Conference on Computer Vision (ICCV)*, 2021.

[9] B. Cheng, I. Misra, A. G. Schwing, A. Kirillov, and R. Girdhar. Masked-attention mask transformer for universal image segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1290–1299, 2022.

[10] H. K. Cheng, J. Chung, Y.-W. Tai, and C.-K. Tang. Cascadepsp: Toward class-agnostic and very high-resolution segmentation via global and local refinement. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8890–8899, 2020.

[11] J. Cheng, J. Ye, Z. Deng, J. Chen, T. Li, H. Wang, Y. Su, Z. Huang, J. Chen, L. J. H. Sun, J. He, S. Zhang, M. Zhu, and Y. Qiao. Sam-med2d, 2023.

[12] M. Cho, S. Kwak, C. Schmid, and J. Ponce. Unsupervised object discovery and localization in the wild: Part-based matching with bottom-up region proposals, 2015.

[13] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.

[14] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[15] A. Gupta, P. Dollar, and R. Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5356–5364, 2019.

[16] J. He, S. Yang, S. Yang, A. Kortylewski, X. Yuan, J.-N. Chen, S. Liu, C. Yang, Q. Yu, and A. Yuille. Partimagenet: A large, high-quality dataset of parts. In *European Conference on Computer Vision*, pages 128–145. Springer, 2022.

[17] K. He, X. Chen, S. Xie, Y. Li, P. Dollár, and R. Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022.

[18] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[19] A. Q. Jiang, A. Sablayrolles, A. Mensch, C. Bamford, D. S. Chaplot, D. d. l. Casas, F. Bressand, G. Lengyel, G. Lample, L. Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023.

[20] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

[21] A. Kirillov, E. Mintun, N. Ravi, H. Mao, C. Rolland, L. Gustafson, T. Xiao, S. Whitehead, A. C. Berg, W.-Y. Lo, P. Dollár, and R. Girshick. Segment anything. *arXiv:2304.02643*, 2023.

[22] J. Lafferty, A. McCallum, F. Pereira, et al. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Icml*, volume 1, page 3. Williamstown, MA, 2001.

[23] F. Li, H. Zhang, P. Sun, X. Zou, S. Liu, J. Yang, C. Li, L. Zhang, and J. Gao. Semantic-sam: Segment and recognize anything at any granularity. *arXiv preprint arXiv:2307.04767*, 2023.

[24] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[25] Z. Liu, Y. Lin, Y. Cao, H. Hu, Y. Wei, Z. Zhang, S. Lin, and B. Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021.

[26] T. Lüddecke and A. Ecker. Image segmentation using text and image prompts. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 7086–7096, 2022.

[27] S. R. Mitroff, B. J. Scholl, and K. Wynn. Divide and conquer: How object files adapt when a persisting object splits into two. *Psychological Science*, 15(6):420–425, 2004.

[28] D. Niu, X. Wang, X. Han, L. Lian, R. Herzig, and T. Darrell. Unsupervised universal image segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

[29] L. Qi, J. Kuen, Y. Wang, J. Gu, H. Zhao, P. Torr, Z. Lin, and J. Jia. Open world entity segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

[30] V. Ramanathan, A. Kalia, V. Petrovic, Y. Wen, B. Zheng, B. Guo, R. Wang, A. Marquez, R. Kovvuri, A. Kadian, et al. Paco: Parts and attributes of common objects. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7141–7151, 2023.

[31] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on pattern analysis and machine intelligence*, 22(8):888–905, 2000.

[32] O. Siméoni, G. Puy, H. V. Vo, S. Roburin, S. Gidaris, A. Bursuc, P. Pérez, R. Marlet, and J. Ponce. Localizing objects with self-supervised transformers and no labels. *arXiv preprint arXiv:2109.14279*, 2021.

[33] G. Team, R. Anil, S. Borgeaud, Y. Wu, J.-B. Alayrac, J. Yu, R. Soricut, J. Schalkwyk, A. M. Dai, A. Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

[34] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[35] W. Van Gansbeke, S. Vandenhende, and L. Van Gool. Discovering object masks with transformers for unsupervised semantic segmentation. *arxiv preprint arxiv:2206.06363*, 2022.

[36] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need, 2023.

[37] H. V. Vo, F. Bach, M. Cho, K. Han, Y. LeCun, P. Perez, and J. Ponce. Unsupervised image matching and object discovery as optimization, 2019.

[38] H. V. Vo, P. Pérez, and J. Ponce. Toward unsupervised, multi-object discovery in large-scale image collections. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XXIII 16*, pages 779–795. Springer, 2020.

[39] X. Wang, R. Girdhar, S. X. Yu, and I. Misra. Cut and learn for unsupervised object detection and instance segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3124–3134, 2023.

[40] X. Wang, I. Misra, Z. Zeng, R. Girdhar, and T. Darrell. Videocutler: Surprisingly simple unsupervised video instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22755–22764, 2024.

[41] X. Wang, Z. Yu, S. De Mello, J. Kautz, A. Anandkumar, C. Shen, and J. M. Alvarez. Freesolo: Learning to segment objects without annotations. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14176–14186, 2022.

[42] X. Wang, R. Zhang, T. Kong, L. Li, and C. Shen. Solov2: Dynamic and fast instance segmentation. *Proc. Advances in Neural Information Processing Systems (NeurIPS)*, 2020.

[43] X. Wang, X. Zhang, Y. Cao, W. Wang, C. Shen, and T. Huang. Seggpt: Segmenting everything in context. *arXiv preprint arXiv:2304.03284*, 2023.

[44] Y. Wang, X. Shen, Y. Yuan, Y. Du, M. Li, S. X. Hu, J. L. Crowley, and D. Vaufreydaz. Tokencut: Segmenting objects in images and videos with self-supervised transformer and normalized cut. *IEEE transactions on pattern analysis and machine intelligence*, 2023.

[45] Y. Xiong, B. Varadarajan, L. Wu, X. Xiang, F. Xiao, C. Zhu, X. Dai, D. Wang, F. Sun, F. Iandola, R. Krishnamoorthi, and V. Chandra. Efficientsam: Leveraged masked image pretraining for efficient segment anything, 2023.

[46] A. Young, B. Chen, C. Li, C. Huang, G. Zhang, G. Zhang, H. Li, J. Zhu, J. Chen, J. Chang, et al. Yi: Open foundation models by 01. ai. *arXiv preprint arXiv:2403.04652*, 2024.

[47] X. Zhao, W. Ding, Y. An, Y. Du, T. Yu, M. Li, M. Tang, and J. Wang. Fast segment anything, 2023.

[48] B. Zhou, H. Zhao, X. Puig, T. Xiao, S. Fidler, A. Barriuso, and A. Torralba. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, 127:302–321, 2019.

A Appendix

A.1 Training Details

Pseudo mask preparation details. Empirically, in the divide stage, we set the confidence threshold τ = 0.3; in the conquer stage, we choose threshold $\theta_{merge} = [0.6, 0.5, 0.4, 0.3, 0.2, 0.1]$. For each image, the divide-and-conquer pipeline generates on average 334 pseudo masks. In the self-training phase, the $\tau_{\text{self-train}}$ = 0.7, and each image has 448 pseudo masks per image after merging high-confidence mask predictions generated by UnSAM. When merging the pseudo masks with the ground truths for training UnSAM+, we select $\tau_{\text{UnSAM+}} = 0.02$.

Whole-image segmentation. UnSAM picks DINO [8] pre-trained ResNet-50 [18] as the backbone and Mask2former [9] as the mask decoder. Given the abundant number of pseudo masks generated, UnSAM augments data only by cropping a 1024×1024 region from the original image. To cope with a large amount of 'ground-truth' masks per image, we find that having 2000 learnable queries produces the best result. We randomly select at most 200 'ground-truth' masks per image to speed up the training process. The default learning rate is 5×10^{-5} with batch size equals 16 and weight decay 5×10^{-2}. We train the model for 8 epochs. All model training in this paper was conducted using either 4 A100 GPUs or 8 RTX 3090 GPUs.

Promptable segmentation. UnSAM uses the self-supervised pre-trained Swin-Transformer [25], specifically the Swin-Tiny model, as the backbone and leverages Semantic-SAM [23] as the base model. Given at most 6 levels of masks corresponding to one input point in SA-1B [21], we set the number of hierarchy levels to 6, which is also the number of predicted masks UnSAM generates per prompt during inference. However, one can easily train with a different number of granularity levels as needed. The default learning rate is 1×10^{-4} with a batch size of 8. The learning rate decreases by a factor of 10 at 90% and 95% of the training iterations. We train the model for 4 epochs.

A.2 Preliminary: Cut and Learn (CutLER) and MaskCut

CutLER [39] introduces a cut-and-learn pipeline to precisely segment instances without supervision. The initial phase, known as the cut stage, uses a normalized cut-based method, MaskCut [39], to generate high-quality instance masks that serve as pseudo-labels for subsequent learning phases. MaskCut begins by harnessing semantic information extracted from "key" features K_i of patch i in the last attention layer of unsupervised vision transformers. It then calculates a patch-wise cosine similarity matrix $W_{ij} = \frac{K_i K_j}{|K_i|_2 |K_j|_2}$. To extract multiple instance masks from a single image, MaskCut initially applies Normalized Cuts [31], which identify the eigenvector x corresponding to the second smallest eigenvalue. The vector x is then bi-partitioned to extract the foreground instance mask M^s. Subsequent iterations repeat this operation but adjust by masking out patches from previously segmented instances in the affinity matrix: $W^t_{ij} = \frac{\left(K_i \sum_{s=1}^{t} M^s_{ij}\right)\left(K_j \sum_{s=1}^{t} M^s_{ij}\right)}{\|K_i\|_2 \|K_j\|_2}$ Subsequently, CutLER's learning stage trains a segmentation/detection model with drop-loss, which encourages the model to explore areas not previously identified by MaskCut. An iterative self-training phase is employed for continuously refining the model's performance.

A.3 Preliminary: Segment Anything Model (SAM) and SA-1B

Inspired by achievement in the NLP field, the Segment Anything project [21] introduces the novel *promptable segmentation task*. At its core lies the Segment Anything Model (SAM) [21], which is capable of producing segmentation masks given user-provided text, points, boxes, and masks in a zero-shot manner. SAM comprises three key components: an MAE [17] pre-trained Vision Transformer [14] that extracts image embeddings, the prompt encoders that embed various types of prompts, and a lightweight Transformer [36] decoder that predicts segmentation masks by integrating image and prompt embeddings.

One significant contribution of SAM [21] is the release of the SA-1B dataset, which comprises 11 million high-resolution images and 1.1 billion segmentation masks, providing a substantial resource for training and evaluating segmentation models. In particular, annotators interactively used SAM to annotate images, and this newly annotated data was then utilized to iteratively update SAM. This cycle was repeated multiple times to progressively enhance both the model and the dataset.

While SAM [21] significantly accelerates the labeling of segmentation masks, annotating an image still requires approximately 14 seconds per mask. Given that each image contains over 100 masks, this equates to more than 30 minutes per image, posing a substantial cost and making it challenging to scale up the training data effectively.

A.4 Evaluation Datasets

COCO (Common Objects in Context) [24] is a widely utilized object detection and segmentation dataset. It consists of 115,000 labeled training images, 5,000 labeled validation images, and more than 200,000 unlabeled images. Its object segmentation covers 80 categories and is mainly on the instance-level. We evaluate our model on COCO `Val2017` with 5000 validation images without training or fine-tuning on any images from the COCO training set. The metrics we choose are class-agnostic COCO style averaged precision and averaged recall for the whole-image inference task, and MaxIoU and OracleIoU for the promptable segmentation task.

SA-1B [21] consists of 11 million high-resolution (1500 on average) images and 1.1 billion segmentation masks, approximately 100 masks per image. All masks are collected in a class-agnostic manner with various subject themes including locations, objects, and scenes. Masks cover a wide range of granularity levels, from large-scale objects to fine-grained details. In the whole-image inference task, we randomly selected 1000 SA-1B images that are not used to generate pseudo labels as the validation set.

LVIS (Large Vocabulary Instance Segmentation) [15] has 164,000 images with more than 1,200 categories and more than 2 million high-quality instance-level segmentation masks. It has a long tail distribution that naturally reveals a large number of rare categories. In the whole-image inference task, we evaluate our model using its 5000 validation images in a zero-shot manner.

EntitySeg [29] is an open-world, class-agnostic dataset that consists of 33277 images in total. There are on average 18.1 entities per image. More than 80% of its images are of high resolution with at least 1000 pixels for the width. EntitySeg also has more accurate boundary annotations. In the whole-image inference task, we evaluate our model with 1314 low-resolution version images (800×1300 on average) in a zero-shot manner.

PACO (Parts and Attributes of Common Objects) [30] is a detection dataset that provides 641,000 masks for part-level entities not included in traditional datasets. It covers 75 object categories and 456 object-part categories. In the whole-image inference task, we evaluate our model with 2410 validation images in a zero-shot manner.

PartImageNet [16] is a large-scale, high-quality dataset with rich part segmentation annotations on a general set of classes with non-rigid, articulated objects. It includes 158 classes and 24,000 images from ImageNet [13]. In the whole-image inference task, we evaluate our model with 2956 validation images in a zero-shot manner.

ADE20K [48] is composed of 25,574 training and 2,000 testing images spanning 365 different scenes. It mainly covers semantic-level segmentation with 150 semantic categories and 707,868 objects from 3,688 categories. In the whole-image inference task, we evaluate our model with 2000 testing images in a zero-shot manner.

A.5 More Visualizations

We provide more qualitative results of UnSAM and UnSAM+ in a zero-shot manner in Figure A1, and Figure A2.

Figure A1: More visualizations on COCO [24]. From top to bottom are raw images, segmentation by SAM, segmentation by UnSAM, and segmentation by UnSAM+.

Figure A2: More visualizations on PACO [30]. From top to bottom are raw images, segmentation by SAM, segmentation by UnSAM, and segmentation by UnSAM+.

Figure A3: Failure cases of UnSAM. From left to right are raw images, segmentation by SAM, and segmentation by UnSAM.

A.6 Limitations

In images with very dense fine-grained details, UnSAM tends to miss repetitive instances with similar texture. As shown in Figure A3, in the first row, although UnSAM accurately segments the leaves in the center of the picture, it misses some leaves located at the top of the image. Additionally, UnSAM occasionally over-segment images. In the second row, the right sleeve cuff of the dancer has meaningless segmentation masks. This issue mainly arises because the unsupervised clustering method mistakenly considers some information, such as folds and shadows on clothing, as criteria for distinguishing different entities. In contrast, human annotators can use prior knowledge to inform the model that such information should not be valid criteria. In this regard, unsupervised methods still need to close the gap with supervised methods.

A.7 Ethical Considerations

We train UnSAM and UnSAM+ on ground truths of and pseudo masks generated on SA-1B [21]. SA-1B contains licensed images that are filtered for objectionable content. It is geographically diverse, but some regions and economic groups are underrepresented. Downstream use of UnSAM and UnSAM+ may create their own potential biases.

NeurIPS Paper Checklist

The checklist is designed to encourage best practices for responsible machine learning research, addressing issues of reproducibility, transparency, research ethics, and societal impact. Do not remove the checklist: **The papers not including the checklist will be desk rejected.** The checklist should follow the references and follow the (optional) supplemental material. The checklist does NOT count towards the page limit.

Please read the checklist guidelines carefully for information on how to answer these questions. For each question in the checklist:

- You should answer [Yes] , [No] , or [NA] .
- [NA] means either that the question is Not Applicable for that particular paper or the relevant information is Not Available.
- Please provide a short (1–2 sentence) justification right after your answer (even for NA).

The checklist answers are an integral part of your paper submission. They are visible to the reviewers, area chairs, senior area chairs, and ethics reviewers. You will be asked to also include it (after eventual revisions) with the final version of your paper, and its final version will be published with the paper.

The reviewers of your paper will be asked to use the checklist as one of the factors in their evaluation. While "[Yes] " is generally preferable to "[No] ", it is perfectly acceptable to answer "[No] " provided a proper justification is given (e.g., "error bars are not reported because it would be too computationally expensive" or "we were unable to find the license for the dataset we used"). In general, answering "[No] " or "[NA] " is not grounds for rejection. While the questions are phrased in a binary way, we acknowledge that the true answer is often more nuanced, so please just use your best judgment and write a justification to elaborate. All supporting evidence can appear either in the main paper or the supplemental material, provided in appendix. If you answer [Yes] to a question, in the justification please point to the section(s) where related material for the question can be found.

IMPORTANT, please:

- **Delete this instruction block, but keep the section heading "NeurIPS paper checklist"**,
- **Keep the checklist subsection headings, questions/answers and guidelines below.**
- **Do not modify the questions and only use the provided macros for your answers**.

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The main claims made in our abstract and introduction precisely summarize this paper's contributions, assumptions, and scope.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

Justification: We discuss limitations of our method in appendix A.6.

Guidelines:

- The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
- The authors are encouraged to create a separate "Limitations" section in their paper.
- The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
- The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
- The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
- The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
- If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
- While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [NA]

   Justification: [NA]

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: All information needed to reproduce the main experimental results is included in Appendix A.1.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We will open-source the codes and models.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).

- Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: All basic settings of the pseudo mask preparation process and model training are included in Appendix A.1.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Following established protocols from prior studies, we report our experimental results. We observed robustness in our results against the variability of random seeds used for initializing model weights. Additionally, our pseudo-mask generation process does not require retraining a parameterized model, thus ensuring deterministic results.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
   - The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
   - The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
   - The assumptions made should be given (e.g., Normally distributed errors).
   - It should be clear whether the error bar is the standard deviation or the standard error of the mean.
   - It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
   - For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
   - If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: We provide full information on the compute resources we use in Appendix A.1.

   Guidelines:

- The answer NA means that the paper does not include experiments.
- The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
- The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
- The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The research conducted in this paper fully conform with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: Broader impacts of our research are discussed in Appendix A.7.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.
    - If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: [NA]

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [Yes]

Justification: [Yes]

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

Answer: [NA]

Justification: [NA]

Guidelines:

- The answer NA means that the paper does not release new assets.
- Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
- The paper should discuss whether and how consent was obtained from people whose asset is used.
- At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [NA]

Justification: This paper doesn't include crowdsourcing or research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: This paper doesn't include crowdsourcing or research with human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.