CLUSTSEG: Clustering for Universal Segmentation

James Liang [1] **Tianfei Zhou** [2] **Dongfang Liu** [1] **Wenguan Wang** [3]

https://github.com/JamesLiang819/ClustSeg

Abstract

We present CLUSTSEG, a general, transformer-based framework that tackles different image segmentation tasks (*i.e.*, superpixel, semantic, instance, and panoptic) through a unified, neural clustering scheme. Regarding queries as cluster centers, CLUSTSEG is innovative in two aspects: ① cluster centers are initialized in heterogeneous ways so as to pointedly address task-specific demands (*e.g.*, instance- or category-level distinctiveness), yet without modifying the architecture; and ② pixel-cluster assignment, formalized in a cross-attention fashion, is alternated with cluster center update, yet without learning additional parameters. These innovations closely link CLUSTSEG to EM clustering and make it a transparent and powerful framework that yields superior results across the above segmentation tasks.

1. Introduction

Image segmentation aims at partitioning pixels into groups. Different notions of pixel groups lead to different types of segmentation tasks. For example, *superpixel* segmentation groups perceptually similar and spatially coherent pixels together. *Semantic* and *instance* segmentation interpret pixel groups based on semantic and instance relations respectively. *Panoptic* segmentation (Kirillov et al., 2019b) not only distinguishes pixels for countable *things* (*e.g.*, dog, car) at the instance level, but merges pixels of amorphous and uncountable *stuff* regions (*e.g.*, sky, grassland) at the semantic level.

These segmentation tasks are traditionally resolved by different technical protocols, *e.g.*, *per-pixel classification* for semantic segmentation, *detect-then-segment* for instance segmentation, and *proxy task learning* for panoptic segmentation. As a result, the developed segmentation solutions are highly task-specialized, and research endeavors are diffused.

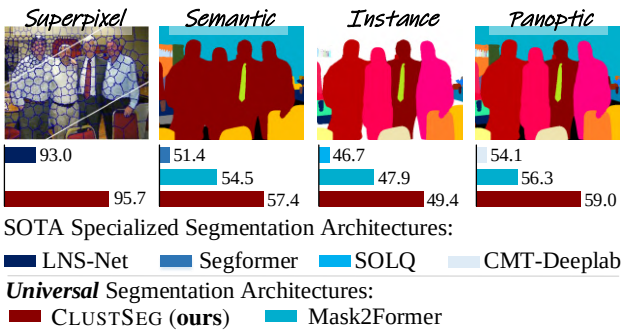

Figure 1. CLUSTSEG unifies four segmentation tasks (*i.e.*, superpixel, semantic, instance, and panoptic) from the clustering view, and greatly suppresses existing specialized and unified models.

To advance the segmentation field in synergy, a paradigm shift from *task-specialized network architectures* towards a *universal framework* is needed. In an effort to embrace this shift, we propose CLUSTSEG which unifies four segmentation tasks *viz.* superpixel, semantic, instance, and panoptic segmentation, from the *clustering* perspective using transformers. The idea of *segment-by-clustering* — clustering pixels with similar attributes together to form segmentation masks — has a long history (Coleman & Andrews, 1979), yet gets largely overlooked nowadays. By revisiting this classic idea and recasting the cross-attention function as an EM clustering calculator, CLUSTSEG sticks the principle of pixel clustering through several innovative algorithmic designs, outperforming existing specialized and unified models (Fig. 1).

Concretely, our innovations are centred around two aspects and respect some critical rules of iterative/EM clustering:
① **Cluster center initialization**: By resorting to the cross-attention for pixel-cluster assignment, the queries in transformers are deemed as cluster centers. From the clustering standpoint, the choice of initial centers is of great importance. However, existing transformer-based segmenters simply learn the queries in a fully parametric manner. By respecting task-specific natures, CLUSTSEG implants concrete meanings to queries: for semantic/stuff segmentation, they are invented as class centers (as the semantic membership is defined on the category level), whereas queries for superpixels/instances/things are emerged purely from the individual

[1]Rochester Institute of Technology [2]ETH Zurich [3]Zhejiang University. Correspondence to: Wenguan Wang <wenguanwang.ai@gmail.com>, Dongfang Liu <dongfang.liu@rit.edu>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

input image (as the target tasks are scene-/instance-specific). This smart query-initialization scheme, called *dreamy-start*, boosts pixel grouping with more informative seeds, as well as allows CLUSTSEG to accommodate the heterogeneous properties of different tasks into one single architecture.

② **Iterative clustering and center update**: To approximate the optimal clustering, EM iteratively alters cluster membership and centers. But current transformer-based segmenters only update the query centers via a few cross-attention based decoders (typically six (Cheng et al., 2021)). Given the success of EM clustering, we devise *recurrent cross-attention* that repeatedly alters cross-attention computation (for pixel-cluster assignment) and attention-based feature aggregation (for center update). By embedding such nonparametric recurrent mechanism, CLUSTSEG fully explores the power of iterative clustering in pixel grouping, without additional learnable parameters and discernible inference speed reduction.

Taking these innovations together, CLUSTSEG becomes a general, flexible, and transparent framework for image segmentation. Unlike prior mask-classification based universal segmenters (Zhang et al., 2021; Cheng et al., 2021; 2022a), our CLUSTSEG acknowledges the fundamental principle of segment-by-clustering. There are a few clustering based segmentation networks (Kong & Fowlkes, 2018; Neven et al., 2019; Yu et al., 2022a;b) — their successes, though limited in their specific targeting tasks, shed light on the potential of unifying image segmentation as pixel clustering. CLUSTSEG, for the first time, shows impressive performance on four core segmentation tasks. In particular, CLUSTSEG sets tantalizing records of **59.0** PQ on COCO panoptic segmentation (Kirillov et al., 2019b), **49.1** AP on COCO instance segmentation (Lin et al., 2014), and **57.4** mIoU on ADE20K semantic segmentation (Zhou et al., 2017), and reports the best ASA and CO curves on BSDS500 superpixel segmentation (Arbelaez et al., 2011).

2. Related Work

Semantic Segmentation interprets high-level semantic concepts of visual stimuli by grouping pixels into different semantic units. Since the proposal of fully convolutional networks (FCNs) (Long et al., 2015), continuous endeavors have been devoted to the design of more powerful FCN-like models, by *e.g.*, aggregating context (Ronneberger et al., 2015; Zheng et al., 2015; Yu & Koltun, 2016), incorporating neural attention (Harley et al., 2017; Wang et al., 2018; Zhao et al., 2018; Hu et al., 2018; Fu et al., 2019), conducting contrastive learning (Wang et al., 2021c), revisiting prototype theory (Zheng et al., 2021; Wang et al., 2023), and adopting generative models (Liang et al., 2022a). Recently, engagement with advanced transformer (Vaswani et al., 2017) architecture attained wide research attention (Xie et al., 2021; Strudel et al., 2021; Zheng et al., 2021; Zhu et al., 2021a; Cheng et al., 2021; 2022a; Gu et al., 2022).

Instance Segmentation groups foreground pixels into different object instances. There are three types of solutions: **i) top-down** models, built upon a *detect-then-segment* protocol, first detect object bounding boxes and then delineate an instance mask for each box (He et al., 2017; Chen et al., 2018a; Huang et al., 2019; Cai & Vasconcelos, 2019; Chen et al., 2019a); **ii) bottom-up** models learn instance-specific pixel embeddings by considering, *e.g.*, instance boundaries (Kirillov et al., 2017), energy levels (Bai & Urtasun, 2017), geometric structures (Chen et al., 2019c), and pixel-center offsets (Zhou et al., 2021), and then merge them as instances; and **iii) single-shot** approaches directly predict instance masks by locations using a set of learnable object queries (Wang et al., 2020c;d; Fang et al., 2021; Liu et al., 2021a; Guo et al., 2021; Liu et al., 2021b; Dong et al., 2021; Hu et al., 2021; Cheng et al., 2022b; Wang et al., 2022; Liu et al., 2023).

Panoptic Segmentation seeks for holistic scene understanding, in terms of the semantic relation between background stuff pixels and the instance membership between foreground thing pixels. Starting from the pioneering work (Kirillov et al., 2019b), prevalent solutions (Kirillov et al., 2019a; Xiong et al., 2019; Li et al., 2019; Liu et al., 2019; Lazarow et al., 2020; Li et al., 2020; Wang et al., 2020a) decompose the problem into various manageable proxy tasks, including box detection, box-based segmentation, and thing-stuff merging. Later, DETR (Carion et al., 2020) and Panoptic FCN (Li et al., 2021) led a shift towards end-to-end panoptic segmentation (Cheng et al., 2020; Wang et al., 2020b; 2021b; Yu et al., 2022a;b). These compact panoptic architectures show the promise of unifying semantic and instance segmentation, but are usually sub-optimal compared with specialized models. This calls for endeavor of more powerful universal algorithms for segmentation.

Superpixel Segmentation is to give a concise image representation by grouping pixels into perceptually meaningful small patches (*i.e.*, superpixel). Superpixel segmentation is an active research area in the pre-deep learning era; see (Stutz et al., 2018) for a thorough survey. Recently, some approaches are developed to harness neural networks to facilitate superpixel segmentation (Jampani et al., 2018; Yang et al., 2020; Zhu et al., 2021b). For instance, Tu et al. (2018) make use of deep learning techniques to learn a superpixel-friendly embedding space; Yang et al. (2020) adopt a FCN to directly predict association scores between pixels and regular grid cells for grid-based superpixel creation.

Universal Image Segmentation pursues a unified architecture for tackling different segmentation tasks. Existing task-specific segmentation models, though advancing the performance in their individual tasks, lack flexibility to generalize to other tasks and cause duplicate research effort. Zhang et al. (2021) initiate the attempt to unify segmentation by dynamic kernel learning. More recently, Cheng et al. (2021;

2022a) formulate different tasks within a mask-classification scheme, using a transformer decoder with object queries. Compared with these pioneers, CLUSTSEG is **i)** more *transparent* and *insightful* — it explicitly acknowledges the fundamental and easy-to-understand principle of segment-by-clustering; **ii)** more *versatile* — it handles more segmentation tasks unanimously; **iii)** more *flexible* — it respects, instead of ignoring, the divergent characters of different segmentation tasks; and **iv)** more *powerful* — it leads by large margins.

Segmentation-by-Clustering, a once popular paradigm, received far less attention nowadays. Recent investigations of such paradigm are primarily made around bottom-up instance segmentation (Kong & Fowlkes, 2018; Neven et al., 2019), where the clustering is adopted as a post-processing step after learning an instance-aware pixel embedding space. More recently, Yu et al. (2022a;b) build end-to-end, clustering based panoptic systems by reformulating the cross-attention as a clustering solver. In this work, we further advance this research line for the development of universal image segmentation. With the innovations in task-specific cluster center initialization and nonparametric recurrent cross-attention, CLUSTSEG better adheres to the nature of clustering and elaborately deals with the heterogeneity across different segmentation tasks using the same architecture.

3. Methodology

3.1. Notation and Preliminary

Problem Statement. Image segmentation seeks to partition an image $I \in \mathbb{R}^{HW\times 3}$ into a set of K meaningful segments:

$$\text{segment}(I) = \{M_k \in \{0,1\}^{HW}\}_{k=1}^{K}, \tag{1}$$

where $M_k(i)$ denotes if pixel $i \in I$ is (1) or not (0) a member of segment k. Different segmentation tasks find the segments according to, for example, semantics, instance membership, or low-level attributes. Also, the number of segments, K, is different for different tasks:In *superpixel* segmentation, K is a pre-determined arbitrary value, *i.e.*, compress I into K superpixels; In *semantic* segmentation, K is fixed as the length of a pre-given list of semantic tags; In *instance* and *panoptic* segmentation, K varies across images and needs to be inferred, as the number of object instances presented in an image is unknown. Some segmentation tasks require explaining the semantics of segments; related symbols are omitted for clarity.

Unifying Segmentation as Clustering. Eq. 1 reveals that, though differing in the definition of a "meaningful segment", segmentation tasks can be essentially viewed as a *pixel clustering* process: the binary segment mask, *i.e.*, M_k, is the pixel assignment matrix w.r.t. k^{th} cluster. With this insight, CLUSTSEG advocates unifying segmentation as clustering. Note that recent mask-classification-based universal segmenters (Cheng et al., 2021; 2022a) do not knowledge the rule of clustering. As the segment masks are the outcome of clustering, the viewpoint of segment-by-clustering is more insightful and a close scrutiny of classical clustering algorithms is needed.

EM Clustering. As a general family of iterative clustering, EM clustering makes K-way clustering of a D-dimensional set of L data points $\boldsymbol{X}=[\boldsymbol{x}_1;\cdots;\boldsymbol{x}_N]\in\mathbb{R}^{N\times D}$ by solving:

$$\max_{\boldsymbol{M}\in\{0,1\}^{K\times N}} \texttt{Tr}(\boldsymbol{M}^\top \boldsymbol{C}\boldsymbol{X}^\top),\ \ s.t.\ \mathbf{1}_K\boldsymbol{M}=\mathbf{1}_N. \tag{2}$$

Here $\boldsymbol{C}=[\boldsymbol{c}_1;\cdots;\boldsymbol{c}_K]\in\mathbb{R}^{K\times D}$ is the *cluster center* matrix and $\boldsymbol{c}_k\in\mathbb{R}^D$ is k^{th} cluster center; $\boldsymbol{M}=[\boldsymbol{m}_1;\cdots;\boldsymbol{m}_N]^\top\in\mathbb{R}^{K\times N}$ is the *cluster assignment* matrix and $\boldsymbol{m}_n\in\{0,1\}^K$ is the one-hot assignment vector of $\boldsymbol{x}_n$; $\mathbf{1}_K$ is a K-dimensional all-ones vector. Principally, EM clustering works as follows: ① **Cluster center initialization**: EM clustering starts with initial estimates for K cluster centers $\boldsymbol{C}^{(0)}=[\boldsymbol{c}_1^{(0)};\cdots;\boldsymbol{c}_K^{(0)}]$. ② **Iterative clustering and center update**: EM clustering proceeds by alternating between two steps:

- *Clustering (Expectation) Step* "softly" assigns each data samples to the K clusters:

$$\hat{\boldsymbol{M}}^{(t)} = \text{softmax}_K(\boldsymbol{C}^{(t)}\boldsymbol{X}^\top)\in[0,1]^{K\times N}, \tag{3}$$

  where $\hat{\boldsymbol{M}}^{(t)}$ denotes the clustering probability matrix.
- *Update (Maximization) Step* recalculate each cluster center from the data according to their membership weights:

$$\boldsymbol{C}^{(t+1)} = \hat{\boldsymbol{M}}^{(t)}\boldsymbol{X}\in\mathbb{R}^{K\times D}. \tag{4}$$

Apparently, the ''hard" sample-to-cluster assignment can be given as: $\boldsymbol{M}=\text{one-hot}(\text{argmax}_K(\hat{\boldsymbol{M}}))$.

Cross-Attention for Clustering. Inspired by DETR (Carion et al., 2020), recent end-to-end panoptic systems (Wang et al., 2021b; Zhang et al., 2021; Cheng et al., 2021; Li et al., 2022) are build upon a query-based scheme: a set of K queries $\boldsymbol{C}=[\boldsymbol{c}_1;\cdots;\boldsymbol{c}_K]\in\mathbb{R}^{K\times D}$ are learned and updated by a stack of transformer decoders for mask decoding. Here "$\boldsymbol{C}$" is reused; we will relate queries with cluster centers later. Specifically, at each decoder, the cross-attention is adopted to adaptively aggregate pixel features to update the queries:

$$\boldsymbol{C}\leftarrow\boldsymbol{C}+\text{softmax}_{HW}(\boldsymbol{Q}^C(\boldsymbol{K}^I)^\top)\boldsymbol{V}^I, \tag{5}$$

where $\boldsymbol{Q}^C\in\mathbb{R}^{K\times D}$, $\boldsymbol{V}^I\in\mathbb{R}^{HW\times D}$, $\boldsymbol{K}^I\in\mathbb{R}^{HW\times D}$ are linearly projected features for query, key, and value; superscripts "C" and "I" indicate the feature projected from the query and image features, respectively. Inspired by (Yu et al., 2022a;b), we reinterpret the cross-attention as a clustering solver by treating queries as cluster centers, and applying *softmax* on the query dimension (K) instead of image resolution (HW):

$$\boldsymbol{C}\leftarrow\boldsymbol{C}+\text{softmax}_K(\boldsymbol{Q}^C(\boldsymbol{K}^I)^\top)\boldsymbol{V}^I. \tag{6}$$

3.2. CLUSTSEG

CLUSTSEG is built on the principle of segment-by-clustering: the segment masks $\{M_k\}_k$ in Eq. 1 correspond to the clustering assignment matrix $\boldsymbol{M}$ in Eq. 2. Clustering can be further

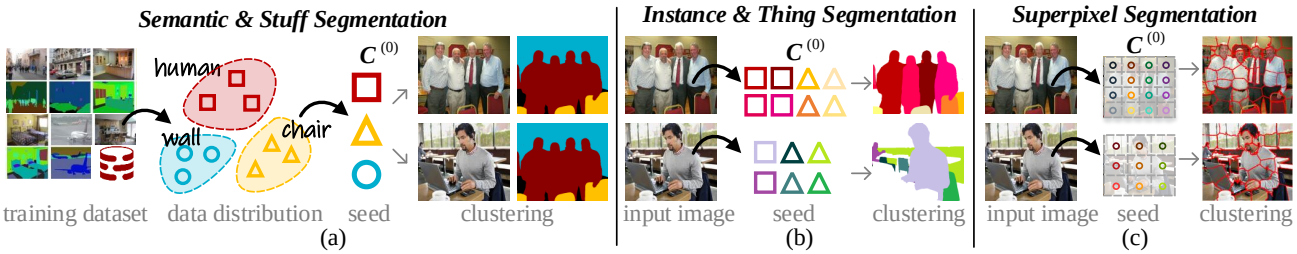

Figure 2. Dreamy-Start for query initialization. (a) To respect the cross-scene semantically consistent nature of semantic/stuff segmentation, the quries/seeds are initialized as class centers (Eq. 7). (b) To meet the instance-aware demand of instance/thing segmentation, the initial seeds are emerged from the input image (Eq. 8). (c) To generate varying number of superpixels, the seeds are initialized from image grids (Eq. 9).

solved in a cross-attention form: the pixel-query affinities $\boldsymbol{Q}^C(\boldsymbol{K}^I)^\top$ in Eq. 6 correspond to the clustering assignment probabilities $\boldsymbol{C}^{(t)}\boldsymbol{X}^\top$ in Eq. 3. In addition, with the close look at EM clustering (*cf.* ①② in §3.1), two inherent defects of existing query-based segmentation models can be identified:

- Due to the stochastic nature, EM clustering is highly sensitive to the selection of initial centers (*cf.* ①) (Celebi et al., 2013). To alleviate the effects of *initial starting conditions*, many initialization methods such as Forgy (randomly choose K data samples as the initial centers) (Hamerly & Elkan, 2002) are proposed. However, existing segmenters simply learn queries/centers in a *fully parametric* manner, without any particular procedure of center initialization.
- EM clustering provably converges to a local optimum (Vattani, 2009). However, it needs a sufficient number of iterations to do so (*cf.* ②). Considering the computational cost and model size, existing segmenters only employ a few cross-attention based decoders (typically 6 (Cheng et al., 2021; Yu et al., 2022a;b)), which may not enough to ensure convergence from the perspective of EM clustering.

As a universal segmentation architecture, CLUSTSEG harnesses the power of recursive clustering to boost pixel grouping. It offers two innovative designs to respectively address the two defects: **i)** a well-crafted query-initialization scheme — *dreamy-start* — for the creation of informative initial cluster centers; and **ii)** a non-parametric recursive module — *recurrent cross-attention* — for effective neural clustering.

Let $\boldsymbol{I}\in\mathbb{R}^{HW\times D}$ denote the set of D-dimensional pixel embeddings of image I. Analogous to EM clustering, CLUSTSEG first creates a set of K queries $\boldsymbol{C}^{(0)}=[\boldsymbol{c}_1^{(0)};\cdots;\boldsymbol{c}_K^{(0)}]$ as initial cluster centers using *dreamy-start*. Then, CLUSTSEG iteratively conducts pixel clustering for mask decoding, by feeding pixel embeddings $\boldsymbol{I}$ and the initial seeds $\boldsymbol{C}^{(0)}$ into a stack of *recurrent cross-attention* decoders.

***Dreamy-Start* for Query Initialization.** *Dreamy-start* takes into account the heterogenous characteristics of different segmentation tasks for the creation of initial seeds $\boldsymbol{C}^{(0)}$ (Fig. 2):

➤ *Semantic Segmentation* groups pixels according to *scene-/instance-agnostic* semantic relations. For example, all the pixels of dogs should be grouped (segmented) into the same cluster, *i.e.*, *dog* class, regardless of whether they are from different images/dog instances. Hence, for semantic-aware pixel clustering, the variance among different instances/scenes should be ignored. In this regard, we explore global semantic structures of the entire dataset to find robust initial seeds. Specifically, during training, we build a memory bank $\mathcal{B}$ to store massive pixel samples for approximating the global data distribution. $\mathcal{B}$ consists of K fixed-size, first-in-first-out queues, *i.e.*, $\mathcal{B}=\{\mathcal{B}_1,\cdots,\mathcal{B}_K\}$; $\mathcal{B}_k$ stores numerous pixel embeddings which are sampled from training images and belong to class k. The initial query for cluster (class) k is given as the corresponding "class center":

$$\begin{aligned}[\boldsymbol{c}_1^{(0)};\cdots;\boldsymbol{c}_K^{(0)}] &= \mathrm{FFN}([\bar{\boldsymbol{x}}_1;\cdots;\bar{\boldsymbol{x}}_K]),\\ \bar{\boldsymbol{x}}_k &= \mathrm{Avg\_Pool}(\mathcal{B}_k)\in\mathbb{R}^D,\end{aligned} \tag{7}$$

where Avg_Pool indicates average pooling, FFN is a fully-connected feed-forward network, and K is set as the size of semantic vocabulary. In this way, the initial centers explicitly summarize the global statistics of the classes, facilitating scene-agnostic semantic relation based pixel clustering. Once trained, these initial seeds will be preserved for testing.

➤ *Instance Segmentation* groups pixels according to *instance-aware* relations — pixels of different dog instances should be clustered into different groups. Different instances possess distinctive properties, *e.g.*, color, scale, position, that are concerning with the local context — the images — that the instances situated in. It is hard to use a small finite set of K fixed queries to characterize all the possible instances. Therefore, unlike previous methods learning K changeless queries for different images, we derive our initial guess of instance-aware centers in an image context-adaptive manner:

$$[\boldsymbol{c}_1^{(0)};\cdots;\boldsymbol{c}_K^{(0)}] = \mathrm{FFN}(\mathrm{PE}(\boldsymbol{I})), \tag{8}$$

where PE denotes position embedding, and K is set as a constant (*i.e.*, 100) — much larger than the typical number of object instances in an image. As such, we utilize image-specific appearance and position cues to estimate content-adaptive seeds for instance-relation-oriented pixel grouping.

➤ *Panoptic Segmentation* groups stuff and thing pixels in terms of semantic and instance relations respectively. Thus we respectively adopt the initialization strategies for semantic segmentation and instance segmentation to estimate two discrete sets of queries for stuff and thing pixel clustering.

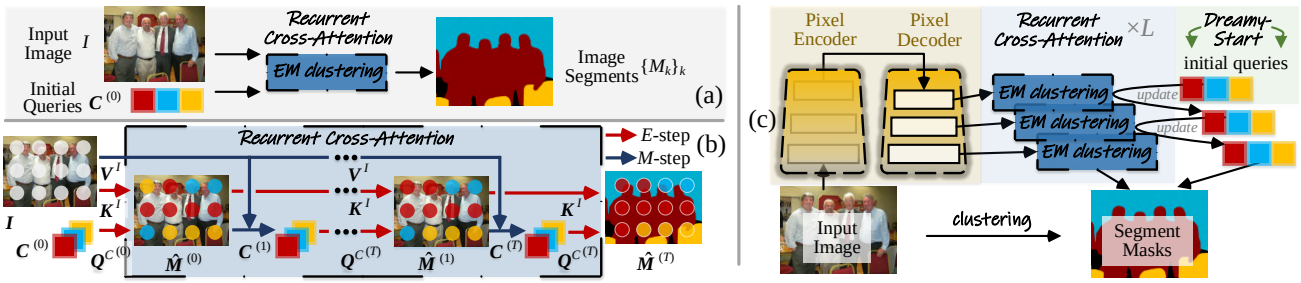

Figure 3. (a) *Recurrent Cross-attention* instantiates EM clustering for segment-by-clustering. (b) Each *Recurrent Cross-attention* layer executes T iterations of clustering assignment (E-step) and center update (M-step). (c) Overall architecture of CLUSTSEG.

➤ *Superpixel Segmentation* groups together pixels that are spatially close and perceptually similar. The number of superpixels K is manually specified beforehand and can be arbitrary. We thus initialize the queries from image grids:

$$[\boldsymbol{c}_1^{(0)}; \cdots; \boldsymbol{c}_K^{(0)}] = \text{FFN}(\text{Grid\_Sample}_K(\text{PE}(\boldsymbol{I}))), \quad (9)$$

where $\text{Grid\_Sample}_K(\text{PE}(\boldsymbol{I}))$ refers to select K position-embedded pixel features from $\text{PE}(\boldsymbol{I})$ using grid-based sampling. The queries are used to group their surrounding pixels as superpixels. CLUSTSEG is thus general enough to accommodate the classic idea of grid-based clustering (Achanta et al., 2012; Yang et al., 2020) in superpixel segmentation.

Dreamy-Start renders CLUSTSEG with great flexibility of addressing task-specific properties without changing network architecture. Through customized initialization, high-quality cluster seeds are created for better pixel grouping.

***Recurrent Cross-Attention* for Recursive Clustering.** After *Dreamy-Start* based cluster center initialization, CLUSTSEG groups image pixels into K clusters for segmentation, by resembling the workflow of EM clustering (*cf.* ①② in §3.1). Given the pixel embeddings $\boldsymbol{I} \in \mathbb{R}^{HW \times D}$ and initial centers $\boldsymbol{C}^{(0)}$, the iterative procedure of EM clustering with T iterations is encapsulated into a *Recurrent Cross-Attention* layer:

$$\begin{aligned} E\text{-step:} \quad & \hat{\boldsymbol{M}}^{(t)} = \text{softmax}_K(\boldsymbol{Q}^{C^{(t)}}(\boldsymbol{K}^I)^\top), \\ M\text{-step:} \quad & \boldsymbol{C}^{(t+1)} = \hat{\boldsymbol{M}}^{(t)} \boldsymbol{V}^I \in \mathbb{R}^{K \times D}, \end{aligned} \quad (10)$$

where $t \in \{1, \cdots, T\}$, and $\hat{\boldsymbol{M}} \in [0, 1]^{K \times HW}$ is the "soft" cluster assignment matrix (*i.e.*, probability maps of K segments). As defined in §3.1, $\boldsymbol{Q}^C \in \mathbb{R}^{K \times D}$ is the query vector projected from the center $\boldsymbol{C}$, and $\boldsymbol{V}^I, \boldsymbol{K}^I \in \mathbb{R}^{HW \times D}$ are the value and key vectors respectively projected from the image pixel features $\boldsymbol{I}$. *Recurrent Cross-Attention* iteratively updates cluster membership $\hat{\boldsymbol{M}}$ (*i.e.*, E-step) and centers $\boldsymbol{C}$ (*i.e.*, M-step). It enjoys a few appealing characteristics (Fig. 3(a-b)):

- *Efficient:* Compared to the vanilla cross-attention (*cf.* Eq. 5) with the computational complexity $\mathcal{O}(H^2W^2D)$, *Recurrent Cross-Attention* is $\mathcal{O}(TKHWD)$, which is more efficient since $TK \ll HW$. Note that, during iteration, only $\boldsymbol{Q}$ needs to be recalculated, while $\boldsymbol{K}$ and $\boldsymbol{V}$ are only calculated once — the small superscript (t) is only added for $\boldsymbol{Q}$.
- *Non-parametric recursive:* As the projection weights for query, key, and value are shared across iteration, *Recurrent Cross-Attention* achieves recursiveness without occurring extra learnable parameters.
- *Transparent:* Aligning closely with the well-established EM clustering algorithm, *Recurrent Cross-Attention* is crystal-clear and grants CLUSTSEG better transparency.
- *Effective: Recurrent Cross-Attention* exploits the power of recursive clustering to progressively decipher the imagery intricacies. As a result, CLUSTSEG is more likely to converge to a better configuration of image partition.

We adopt a hierarchy of *Recurrent Cross-Attention* based decoders to fully pursue the representational granularity for more effective pixel clustering:

$$\boldsymbol{C}^l = \boldsymbol{C}^{l+1} + \text{RCross\_Attention}^{l+1}(\boldsymbol{I}^{l+1}, \boldsymbol{C}^{l+1}), \quad (11)$$

where $\boldsymbol{I}^l$ is the image feature map at $H/2^l \times W/2^l$ resolution, and $\boldsymbol{C}^l$ is the cluster center matrix for l^{th} decoder. The multi-head mechanism and multi-layer perceptron used in standard transformer decoder are also adopted (but omitted for simplicity). The parameters for different *Recurrent Cross-Attention* layers, *i.e.*, $\{\text{RCross\_Attention}^l\}_{l=1}^L$, are not shared.

3.3. Implementation Details

Detailed Architecture. CLUSTSEG has four parts (Fig. 3(c)):

- *Pixel Encoder* extracts multi-scale dense representations $\{\boldsymbol{I}_l\}_l$ for image I. In §4, we test CLUSTSEG on various CNN-based and vision-transformer backbones.
- *Pixel Decoder*, placed on the top of the encoder, gradually recovers finer representations. As in (Yu et al., 2022b;a; Cheng et al., 2021), we use six axial blocks (Wang et al., 2020b), one at L^{th} level and five at $(L-1)^{th}$ level.
- *Recurrent Cross-Attention based Decoder* performs iterative clustering for pixel grouping. Each *Recurrent Cross-Attention* layer conducts three iterations of clustering, *i.e.*, $T = 3$, and six decoders are used: each two is applied to the pixel decoder at levels $L-2$, $L-1$ and L, respectively.
- *Dreamy-Start* creates informative initial centers for the first *Recurrent Cross-attention* based decoder and is customized to different tasks. For semantic segmentation and stuff classes in panoptic segmentation, the seeds are computed from the memory bank during training (*cf.* Eq. 7)

Table 1. Quantitative results on COCO Panoptic (Kirillov et al., 2019b) `val` for **panoptic segmentation** (see §4.1 for details).

Algorithm	Backbone	Epoch	PQ↑	PQ^{Th}↑	PQ^{St}↑	AP^{Th}_{pan}↑	$mIoU_{pan}$↑
Panoptic-FPN (Kirillov et al., 2019a)	ResNet-101	20	44.0	52.0	31.9	34.0	51.5
UPSNet (Xiong et al., 2019)	ResNet-101	12	46.2	52.8	36.5	36.3	56.9
Panoptic-Deeplab (Cheng et al., 2020)	Xception-71	12	41.2	44.9	35.7	31.5	55.4
Panoptic-FCN (Li et al., 2021)	ResNet-50	12	44.3	50.0	35.6	35.5	55.0
Max-Deeplab (Wang et al., 2021b)	Max-L	55	51.1	57.0	42.2	–	–
CMT-Deeplab (Yu et al., 2022a)	Axial-R104†	55	54.1	58.8	47.1	–	–
Panoptic Segformer (Li et al., 2022)	ResNet-50	24	$49.6_{\pm 0.25}$	$54.4_{\pm 0.26}$	$42.4_{\pm 0.25}$	$39.5_{\pm 0.20}$	$60.8_{\pm 0.21}$
	ResNet-101		$50.6_{\pm 0.21}$	$55.5_{\pm 0.24}$	$43.2_{\pm 0.20}$	$40.4_{\pm 0.21}$	$62.0_{\pm 0.22}$
kMaX-Deeplab (Yu et al., 2022b)	ResNet-50	50	$52.1_{\pm 0.15}$	$57.3_{\pm 0.18}$	$44.0_{\pm 0.16}$	$36.2_{\pm 0.15}$	$60.4_{\pm 0.14}$
	ConvNeXt-B†		$56.2_{\pm 0.19}$	$62.4_{\pm 0.22}$	$46.8_{\pm 0.21}$	$42.2_{\pm 0.24}$	$65.3_{\pm 0.19}$
K-Net (Zhang et al., 2021)	ResNet-101	36	$48.4_{\pm 0.26}$	$53.3_{\pm 0.28}$	$40.9_{\pm 0.22}$	$38.5_{\pm 0.25}$	$60.1_{\pm 0.20}$
	Swin-L†		$55.2_{\pm 0.22}$	$61.2_{\pm 0.25}$	$46.2_{\pm 0.19}$	$45.8_{\pm 0.23}$	$64.4_{\pm 0.21}$
Mask2Former (Cheng et al., 2022a)	ResNet-50	50	$51.8_{\pm 0.24}$	$57.7_{\pm 0.23}$	$43.0_{\pm 0.16}$	$41.9_{\pm 0.23}$	$61.7_{\pm 0.20}$
	ResNet-101		$52.4_{\pm 0.22}$	$58.2_{\pm 0.16}$	$43.6_{\pm 0.22}$	$42.4_{\pm 0.20}$	$62.4_{\pm 0.21}$
	Swin-B†		$56.3_{\pm 0.21}$	$62.5_{\pm 0.24}$	$46.9_{\pm 0.18}$	$46.3_{\pm 0.23}$	$65.1_{\pm 0.21}$
CLUSTSEG (ours)	ResNet-50	50	$54.3_{\pm 0.20}$	$60.4_{\pm 0.22}$	$45.8_{\pm 0.23}$	$42.2_{\pm 0.18}$	$63.8_{\pm 0.25}$
	ResNet-101		$55.3_{\pm 0.21}$	$61.3_{\pm 0.15}$	$46.4_{\pm 0.17}$	$43.0_{\pm 0.19}$	$64.1_{\pm 0.25}$
	ConvNeXt-B†		$58.8_{\pm 0.18}$	$64.5_{\pm 0.16}$	**48.8**$_{\pm 0.22}$	$46.9_{\pm 0.17}$	**66.3**$_{\pm 0.20}$
	Swin-B†		**59.0**$_{\pm 0.20}$	**64.9**$_{\pm 0.23}$	$48.7_{\pm 0.19}$	**47.1**$_{\pm 0.21}$	$66.2_{\pm 0.18}$

†: backbone pre-trained on ImageNet-22K (Deng et al., 2009); the marker is applicable to other tables.

and stored unchanged once training finished. In other cases, the seeds are built on-the-fly (*cf.* Eqs. 8 and 9).

Loss Function. CLUSTSEG can be applied to the four segmentation tasks, without architecture change. **We opt the standard loss design in each task setting for training (details in the supplementary)**. In addition, recall that *Recurrent Cross-attention* estimates the cluster probability matrix $\hat{\boldsymbol{M}}^{(t)}$ at each E-step (*cf.* Eq. 10); $\hat{\boldsymbol{M}}^{(t)}$ can be viewed as logit maps of K segments. Therefore, the groundtruth segment masks $\{M_k\}_k$ can be directly used to train every E-step of each *Recurrent Cross-attention*, leading to *intermediate/deep supervision* (Lee et al., 2015; Yu et al., 2022b)

4. Experiment

CLUSTSEG is the first framework to support four core segmentation tasks with a single unified architecture. To demonstrate its broad applicability and wide benefit, we conduct

Extensive experiments: We benchmark it on panoptic (§4.1), instance (§4.2), semantic (§4.3), and superpixel (§4.4) segmentation, and carry out ablation study (§4.5). We also approach it on **diverse backbones**: ResNet (He et al., 2016), ConvNeXt (Liu et al., 2022), and Swin (Liu et al., 2021c).

4.1. Experiment on Panoptic Segmentation

Dataset. We use COCO Panoptic (Kirillov et al., 2019b) — `train2017` is adopted for training and `val2017` for test.
Training. We set the initial learning rate to 1e-5, training epoch to 50, and batch size to 16. We use random scale jittering with a factor in $[0.1, 2.0]$ and a crop size of 1024×1024.
Test. We use one input image scale with shorter side as 800.
Metric. We use PQ (Kirillov et al., 2019b) and also report PQ^{Th} and PQ^{St} for "thing" and "stuff" classes, respectively. For completeness, we involve AP^{Th}_{pan}, which is AP evaluated on "thing" classes using instance segmentation annotations, and $mIoU_{pan}$, which is mIoU for semantic segmentation by merging instance masks from the same category, using the same model trained for panoptic segmentation task.

Performance Comparison. We compare CLUSTSEG with two families of state-of-the-art methods: *universal* approaches (*i.e.*, K-Net (Zhang et al., 2021), Mask2Former (Cheng et al., 2022a)), and *specialized* panoptic systems (Kirillov et al., 2019a; Xiong et al., 2019; Cheng et al., 2020; Li et al., 2021; Wang et al., 2021b; Zhang et al., 2021; Li et al., 2022; Yu et al., 2022a). As shown in Table 1, CLUSTSEG beats all universal rivals, *i.e.*, Mask2Former and K-Net, on COCO Panoptic `val`. With ResNet-50/-101, CLUSTSEG outperforms Mask2Former by **2.3%**/**2.9%** PQ; with Swin-B, the margin is **2.7%** PQ. Also, CLUSTSEG's performance is clearly ahead of K-Net (**59.0%** *vs.* 55.2%), even using a lighter backbone (Swin-B *vs.* Swin-L). Furthermore, CLUSTSEG outperforms all the well-established specialist panoptic algorithms. Notably, it achieves promising gains of **2.6%**/**2.1**%/**2.0**% in terms of PQ/PQ^{th}/PQ^{St} against kMax-Deeplab on the top of ConvNeXt-B. Beyond metric PQ, CLUSTSEG gains superior performance in terms of AP^{Th}_{pan} and $mIoU_{pan}$. In summary, CLUSTSEG, with Swin-B backbone, ***sets new records across all the metrics on COCO Panoptic*** **`val`**.

4.2. Experiment on Instance Segmentation

Dataset. As standard, we adopt COCO (Lin et al., 2014) — `train2017` is used for training and `test-dev` for test.
Training. We set the initial learning rate to 1e-5, training epoch to 50, and batch size to 16. We use random scale jittering with a factor in $[0.1, 2.0]$ and a crop size of 1024×1024.
Test. We use one input image scale with shorter side as 800.
Metric. We adopt AP, AP_{50}, AP_{75}, AP_S, AP_M, and AP_L.

Table 2. Quantitative results on COCO (Lin et al., 2014) `test-dev` for **instance segmentation** (see §4.2 for details).

Algorithm	Backbone	Epoch	AP↑	AP_{50}↑	AP_{75}↑	AP_S	AP_M↑	AP_L↑
Mask R-CNN (He et al., 2017)	ResNet-101	12	36.1	57.5	38.6	18.8	39.7	49.5
Cascade MR-CNN (Cai & Vasconcelos, 2019)	ResNet-101	12	37.3	58.2	40.1	19.7	40.6	51.5
HTC (Chen et al., 2019a)	ResNet-101	20	39.6	61.0	42.8	21.3	42.9	55.0
PointRend (Kirillov et al., 2020)	ResNet-50	12	36.3	56.9	38.7	19.8	39.4	48.5
BlendMask (Chen et al., 2020)	ResNet-101	36	38.4	60.7	41.3	18.2	41.5	53.3
QueryInst (Fang et al., 2021)	ResNet-101	36	41.0	63.3	44.5	21.7	44.4	60.7
SOLQ (Dong et al., 2021)	Swin-L†	50	46.7	72.7	50.6	29.2	50.1	60.9
SparseInst (Cheng et al., 2022b)	ResNet-50	36	37.9	59.2	40.2	15.7	39.4	56.9
kMaX-Deeplab (Yu et al., 2022b)	ResNet-50	50	$40.2_{\pm 0.19}$	$61.5_{\pm 0.20}$	$43.7_{\pm 0.18}$	$21.7_{\pm 0.21}$	$43.0_{\pm 0.19}$	$54.0_{\pm 0.22}$
	ConvNeXt-B†		$44.7_{\pm 0.24}$	$67.5_{\pm 0.25}$	$48.1_{\pm 0.21}$	$25.1_{\pm 0.17}$	$47.6_{\pm 0.23}$	$61.5_{\pm 0.21}$
K-Net (Zhang et al., 2021)	ResNet-101	36	$40.1_{\pm 0.17}$	$62.8_{\pm 0.23}$	$43.1_{\pm 0.19}$	$18.7_{\pm 0.22}$	$42.7_{\pm 0.18}$	$58.8_{\pm 0.20}$
	Swin-L†		$46.1_{\pm 0.18}$	$67.7_{\pm 0.20}$	$49.6_{\pm 0.19}$	$24.3_{\pm 0.23}$	$49.5_{\pm 0.21}$	$65.1_{\pm 0.23}$
Mask2Former (Cheng et al., 2022a)	ResNet-50	50	$42.8_{\pm 0.23}$	$65.3_{\pm 0.21}$	$46.0_{\pm 0.22}$	$22.1_{\pm 0.19}$	$46.3_{\pm 0.21}$	$64.8_{\pm 0.23}$
	ResNet-101		$43.9_{\pm 0.19}$	$66.7_{\pm 0.17}$	$47.0_{\pm 0.19}$	$22.9_{\pm 0.20}$	$47.7_{\pm 0.15}$	$66.3_{\pm 0.18}$
	Swin-B†		$47.9_{\pm 0.19}$	$68.9_{\pm 0.18}$	$51.8_{\pm 0.21}$	$29.9_{\pm 0.23}$	$51.5_{\pm 0.20}$	$68.5_{\pm 0.18}$
CLUSTSEG (ours)	ResNet-50	50	$44.2_{\pm 0.25}$	$66.7_{\pm 0.27}$	$47.8_{\pm 0.24}$	$24.3_{\pm 0.20}$	$48.5_{\pm 0.21}$	$64.3_{\pm 0.24}$
	ResNet-101		$45.5_{\pm 0.22}$	$67.8_{\pm 0.21}$	$48.9_{\pm 0.24}$	$25.1_{\pm 0.20}$	$50.3_{\pm 0.23}$	$66.9_{\pm 0.27}$
	ConvNeXt-B†		$49.0_{\pm 0.23}$	**70.4**$_{\pm 0.22}$	$52.7_{\pm 0.20}$	**30.1**$_{\pm 0.18}$	$52.9_{\pm 0.24}$	**68.6**$_{\pm 0.25}$
	Swin-B†		**49.1**$_{\pm 0.21}$	$70.3_{\pm 0.20}$	**52.9**$_{\pm 0.23}$	**30.1**$_{\pm 0.18}$	**53.2**$_{\pm 0.20}$	$68.4_{\pm 0.21}$

Table 3. Quantitative results on ADE20K (Zhou et al., 2017) `val` for **semantic segmentation** (see §4.3 for details).

Algorithm	Backbone	Epoch	mIoU↑
FCN (Long et al., 2015)	ResNet-50	50	36.0
DeeplabV3+ (Chen et al., 2018b)	ResNet-50	50	42.7
APCNet (He et al., 2019)	ResNet-50	100	43.4
SETR (Zheng et al., 2021)	ViT-L†	100	49.3
Segmenter (Strudel et al., 2021)	ViT-L†	100	53.5
Segformer (Xie et al., 2021)	MIT-B5	100	51.4
kMaX-Deeplab (Yu et al., 2022b)	ResNet-50	100	$48.1_{\pm 0.13}$
	ConvNeXt-B†		$56.2_{\pm 0.16}$
K-Net (Zhang et al., 2021)	ResNet-50	50	$44.6_{\pm 0.25}$
	Swin-L†		$53.7_{\pm 0.15}$
Mask2Former (Cheng et al., 2022a)	ResNet-50	100	$48.2_{\pm 0.12}$
	Swin-B†		$54.5_{\pm 0.20}$
CLUSTSEG (ours)	ResNet-50	100	$50.5_{\pm 0.16}$
	ConvNeXt-B†		$57.3_{\pm 0.17}$
	Swin-B†		**57.4**$_{\pm 0.22}$

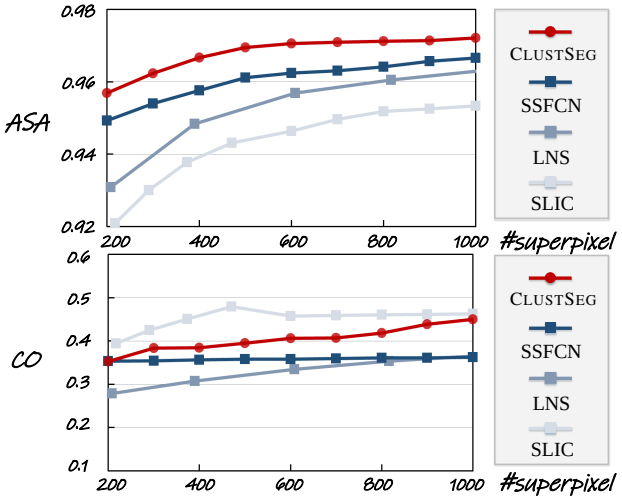

Figure 4. CLUSTSEG reaches the best ASA and CO scores on BSDS500 (Arbelaez et al., 2011) `test`, among all the deep learning based superpixel models (see §4.4 for details).

Performance Comparison. Table 2 presents the results of CLUSTSEG against 11 famous instance segmentation methods on COCO `test-dev`. CLUSTSEG shows clear performance advantages over prior arts. With ResNet-101, it outperforms the universal counterparts Mask2Former by **1.6%** and K-Net by **5.4%** in terms of AP. It surpasses all the specialized competitors, *e.g.*, yielding a significant gain of **4.0%** AP over kMax-Deeplab when using ResNet-50. Without bells and whistles, **CLUSTSEG** ***establishes a new state-of-the-art on COCO instance segmentation***.

4.3. Experiment on Semantic Segmentation

Dataset. We experiment with ADE20K (Zhou et al., 2017), which includes 20K/2K/3K images for `train`/`val`/`test`.
Training. We set the initial learning rate to 1e-5, training epoch to 100, and batch size to 16. We use random scale jittering with a factor in $[0.5, 2.0]$ and a crop size of 640×640.

Test. At the test time, we rescale the shorter side of input image to 640, without any test-time data augmentation.
Metric. Mean intersection-over-union (mIoU) is reported.
Performance Comparison. In Table 3, we further compare CLUSTSEG with a set of semantic segmentation methods on ADE20K `val`. CLUSTSEG yields superior performance. For example, it outperforms Mask2Former by **2.3%** and **2.9%** mIoU using ResNet-50 and Swin-B† backbones, respectively. Furthermore, CLUSTSEG leads other specialist semantic segmentation models like Segformer (Xie et al., 2021), Segmenter (Strudel et al., 2021), and SETR (Zheng et al., 2021) by large margins. Considering that ADE20K is challenging and extensively-studied, such improvements are particularly impressive. In conclusion, **CLUSTSEG** ***ranks top in ADE20K semantic segmentation benchmarking***.

Table 4. A set of **ablative studies** on COCO Panoptic (Li et al., 2022) `val` (see §4.5). The adopted designs are marked in red.

Algorithm Component	PQ↑	PQ^{Th}↑	PQ^{St}↑
BASELINE	49.7	55.5	42.0
+ Dreamy-Start *only*	51.0	56.7	43.6
+ Recurrent Cross-attention *only*	53.2	59.1	44.9
CLUSTSEG (*both*)	**54.3**	**60.4**	**45.8**

(a) Key Component Analysis

Cross-Attention Variant	PQ↑	PQ^{Th}↑	PQ^{St}↑	Training Speed (hour/epoch)↓	Inference Speed (fps)↑
Vanilla (Eq. 5)	51.0	56.7	43.6	1.89	5.88
K-Means (Yu et al., 2022b)	53.4	58.5	45.3	1.58	7.81
Recurrent (Eq. 11)	**54.3**	**60.4**	**45.8**	1.62	7.59

(c) *Recurrent Cross-Attention*

#	*Instance/Thing* free param.	*Instance/Thing* scene-adaptive	*Semantic/Stuff* free param.	*Semantic/Stuff* scene-agnostic	*Semantic/Stuff* scene-adaptive	PQ↑	PQ^{Th}↑	PQ^{St}↑
1	✓		✓			53.2	59.1	44.9
2		✓	✓			54.0	60.2	45.2
3	✓			✓		53.9	59.5	45.7
4	✓				✓	53.5	59.3	45.3
5		✓		✓		**54.3**	**60.4**	**45.8**

(b) *Dreamy-Start* Query-Initialization

T	PQ↑	PQ^{Th}↑	PQ^{St}↑	Training Speed (hour/epoch)↓	Inference Speed (fps)↑
1	53.8	59.7	45.4	1.54	8.08
2	54.1	60.2	45.7	1.59	7.85
3	54.3	60.4	45.8	1.62	7.59
4	54.3	60.4	45.8	1.68	7.25
5	54.3	60.5	45.8	1.74	6.92
6	54.4	60.4	45.9	1.82	6.54

(d) Recursive Clustering

4.4. Experiment on Superpixel Segmentation

Dataset. We use BSDS500 (Arbelaez et al., 2011), which includes 200/100/200 images for `train`/`val`/`test`.
Training. We set the initial learning rate to 1e-4, training iteration to 300K, and batch size to 128. We use random horizontal and vertical flipping, random scale jittering with a factor in $[0.5, 2.0]$, and a crop size of 480×480 for data augmentation. We randomly choose the number of superpixels from 50 to 2500. Note that the grid for query generation is automatically adjusted to match the specified number of superpixels.
Test. During inference, we use the original image size.
Metric. We use achievable segmentation accuracy (ASA) and compactness (CO). ASA is aware of boundary adherence, whereas CO addresses shape regularity.
Performance Comparison. Fig 4 presents comparison results of superpixel segmentation on BSDS500 `test`. In terms of ASA, CLUSTSEG outperforms the classic method SLIC (Achanta et al., 2012) by a large margin, and also surpasses recent three deep learning based competitors, *i.e.*, SSFCN (Yang et al., 2020) and LNS (Zhu et al., 2021b). In addition, CLUSTSEG gains high CO score. As seen, CLUSTSEG performs well on both ASA and CO; this is significant due to the well-known trade-off between edge-preserving and compactness (Yang et al., 2020). Our **CLUSTSEG** ***achieves outstanding performance against state-of-the-art superpixel methods on BSDS500***.

4.5. Diagnostic Experiment

In this section, we dive deep into CLUSTSEG by ablating of its key components on COCO Panoptic (Kirillov et al., 2019b) `val`. ResNet-50 is adopted as the backbone. **More experimental results are given in the supplementary**.

Key Component Analysis. We first investigate the two major ingredients in CLUSTSEG, *i.e.*, ***Dreamy-Start*** for query initialization and ***Recurrent Cross-Attention***, for recursive clustering. We build BASELINE that learns the initial queries fully end-to-end and updates them through standard cross-attention (Eq. 5) based decoders. As reported in Table 4a, BASELINE gives 49.7% PQ, 55.5% PQ^{Th}, and 42.0% PQ^{St}. After applying *Dreamy-Start* to BASELINE, we observe consistent and notable improvements for both 'thing' ($55.5\% \rightarrow$ **56.7%** in PQ^{Th}) and 'stuff' ($42.0\% \rightarrow$ **43.6%** in PQ^{St}), leading to an increase of overall PQ from 49.7% to **51.0%**. This reveals the critical role of object queries and verifies the efficacy of our query-initialization strategy, even without explicitly conducting clustering. Moreover, after introducing *Recurrent Cross-Attention* to BASELINE, we obtain significant gains of **3.5%** PQ, **3.6%** PQ^{Th}, and **2.9%** PQ^{St}. Last, by unifying the two core techniques together, CLUSTSEG yields the best performance across all the three metrics. This suggests that the proposed *Dreamy-Start* and *Recurrent Cross-Attention* can work collaboratively, and confirms the effectiveness of our overall algorithmic design.

***Dreamy-Start* Query-Initialization.** We next study the impact of the our *Dreamy-Start* Query-Initialization scheme. As summarized in Table 4b, when learning the initial queries as free parameters as standard (#1), the model obtains 53.2% PQ, 59.1% PQ^{Th} and 44.9% PQ^{St}. By initializing 'thing' centers in a scene context-adaptive manner (Eq. 8), we observe a large gain of **1.1%** PQ^{Th} (#2). Additionally, with scene-agnostic initialization of 'stuff' centers (Eq. 7), the model yields a clear boost of PQ^{St} from 44.9% to **45.7%** (#3). In addition, we find that only minor gains are achieved for PQ^{St} if 'stuff' centers are also initialized as scene-adaptive (#4). By customizing initialization strategies for both 'thing' and 'stuff' centers, *Dreamy-Start* provides substantial performance improvements across all the metrics (#5).

***Recurrent Cross-Attention*.** We further probe the influence of our *Recurrent Cross-Attention* (Eq. 11), by comparing it with vanilla cross-attention (Eq. 5) and K-Means cross-attention (Yu et al., 2022b). K-Means cross-attention em-

ploys Gumbel-Softmax (Jang et al., 2017) for 'hard' pixel-cluster assignment, without any recursive process. As seen in Table 4c, our *Recurrent Cross-Attention* is *effective* — it improves the vanilla and K-Means by **3.3**% PQ and **0.9**% PQ respectively, and *efficient* — its training and inference speeds are much faster than the vanilla and comparable to K-Means, as consistent with our analysis in §3.2.

Recursive Clustering. Last, to gain more insights into recursive clustering, we ablate the effect of iteration number T in Table 4d. We find that the performance gradually improves from 53.8% PQ to **54.3**% PQ when increasing T from 1 to 3, but remains unchanged after running more iterations. Additionally, the speed of training and inference decreases as T increases. We therefore set $T=3$ by default for a better trade-off between accuracy and computational cost.

5. Conclusion

In this work, our epistemology is centered on the *segment-by-clustering* paradigm, which coins a universal framework, termed CLUSTSEG, to unify the community of image segmentation and respect the distinctive characteristics of each sub-task (*i.e.*, superpixel, semantic, instance, and panoptic). The clustering insight leads us to introduce novel approaches for task-aware query/center initialization and tailor the cross-attention mechanism for recursive clustering. Empirical results suggest that CLUSTSEG achieves superior performance in all the four sub-tasks. Our research may potentially benefit the broader domain of dense visual prediction as a whole.

References

Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., and Süsstrunk, S. Slic superpixels compared to state-of-the-art superpixel methods. *IEEE TPAMI*, 2012.

Arbelaez, P., Maire, M., Fowlkes, C., and Malik, J. Contour detection and hierarchical image segmentation. *IEEE TPAMI*, 2011.

Bai, M. and Urtasun, R. Deep watershed transform for instance segmentation. In *CVPR*, 2017.

Cai, Z. and Vasconcelos, N. Cascade r-cnn: high quality object detection and instance segmentation. *IEEE TPAMI*, 43(5):1483–1498, 2019.

Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., and Zagoruyko, S. End-to-end object detection with transformers. In *ECCV*, 2020.

Celebi, M. E., Kingravi, H. A., and Vela, P. A. A comparative study of efficient initialization methods for the k-means clustering algorithm. *Expert Systems with Applications*, 40(1):200–210, 2013.

Chen, H., Sun, K., Tian, Z., Shen, C., Huang, Y., and Yan, Y. Blendmask: Top-down meets bottom-up for instance segmentation. In *CVPR*, 2020.

Chen, K., Pang, J., Wang, J., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Shi, J., Ouyang, W., et al. Hybrid task cascade for instance segmentation. In *CVPR*, 2019a.

Chen, K., Wang, J., Pang, J., Cao, Y., Xiong, Y., Li, X., Sun, S., Feng, W., Liu, Z., Xu, J., Zhang, Z., Cheng, D., Zhu, C., Cheng, T., Zhao, Q., Li, B., Lu, X., Zhu, R., Wu, Y., Dai, J., Wang, J., Shi, J., Ouyang, W., Loy, C. C., and Lin, D. MMDetection: Open mmlab detection toolbox and benchmark. *arXiv preprint arXiv:1906.07155*, 2019b.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., and Yuille, A. L. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE TPAMI*, 40(4):834–848, 2017.

Chen, L.-C., Hermans, A., Papandreou, G., Schroff, F., Wang, P., and Adam, H. Masklab: Instance segmentation by refining object detection with semantic and direction features. In *CVPR*, 2018a.

Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., and Adam, H. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018b.

Chen, X., Girshick, R., He, K., and Dollár, P. Tensormask: A foundation for dense object segmentation. In *ICCV*, 2019c.

Cheng, B., Collins, M. D., Zhu, Y., Liu, T., Huang, T. S., Adam, H., and Chen, L.-C. Panoptic-deeplab: A simple, strong, and fast baseline for bottom-up panoptic segmentation. In *CVPR*, 2020.

Cheng, B., Schwing, A., and Kirillov, A. Per-pixel classification is not all you need for semantic segmentation. In *NeurIPS*, 2021.

Cheng, B., Misra, I., Schwing, A. G., Kirillov, A., and Girdhar, R. Masked-attention mask transformer for universal image segmentation. In *CVPR*, 2022a.

Cheng, T., Wang, X., Chen, S., Zhang, W., Zhang, Q., Huang, C., Zhang, Z., and Liu, W. Sparse instance activation for real-time instance segmentation. In *CVPR*, 2022b.

Cheng, Z., Liang, J., Choi, H., Tao, G., Cao, Z., Liu, D., and Zhang, X. Physical attack on monocular depth estimation with optimal adversarial patches. In *ECCV*, 2022c.

Cheng, Z., Liang, J., Tao, G., Liu, D., and Zhang, X. Adversarial training of self-supervised monocular depth estimation against physical-world attacks. *ICLR*, 2023.

Coleman, G. B. and Andrews, H. C. Image segmentation by clustering. *Proceedings of the IEEE*, 67(5):773–785, 1979.

Contributors, M. MMDetection: Open mmlab detection toolbox and benchmark. *https://github.com/open-mmlab/mmdetection*, 2019.

Contributors, M. Mmsegmentation: Openmmlab semantic segmentation toolbox and benchmark. *https://github.com/open-mmlab/mmsegmentation*, 2020.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Fei-Fei, L. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

Dong, B., Zeng, F., Wang, T., Zhang, X., and Wei, Y. Solq: Segmenting objects by learning queries. In *NeurIPS*, 2021.

Fang, Y., Yang, S., Wang, X., Li, Y., Fang, C., Shan, Y., Feng, B., and Liu, W. Instances as queries. In *ICCV*, 2021.

Fu, J., Liu, J., Tian, H., Li, Y., Bao, Y., Fang, Z., and Lu, H. Dual attention network for scene segmentation. In *CVPR*, 2019.

Gu, J., Kwon, H., Wang, D., Ye, W., Li, M., Chen, Y.-H., Lai, L., Chandra, V., and Pan, D. Z. Multi-scale high-resolution vision transformer for semantic segmentation. In *CVPR*, 2022.

Guo, R., Niu, D., Qu, L., and Li, Z. Sotr: Segmenting objects with transformers. In *ICCV*, 2021.

Hamerly, G. and Elkan, C. Alternatives to the k-means algorithm that find better clusterings. In *ICIKM*, 2002.

Harley, A. W., Derpanis, K. G., and Kokkinos, I. Segmentation-aware convolutional networks using local attention masks. In *ICCV*, 2017.

He, J., Deng, Z., Zhou, L., Wang, Y., and Qiao, Y. Adaptive pyramid context network for semantic segmentation. In *CVPR*, 2019.

He, K., Zhang, X., Ren, S., and Sun, J. Deep residual learning for image recognition. In *CVPR*, 2016.

He, K., Gkioxari, G., Dollár, P., and Girshick, R. Mask r-cnn. In *ICCV*, 2017.

Hu, J., Shen, L., and Sun, G. Squeeze-and-excitation networks. In *CVPR*, 2018.

Hu, J., Cao, L., Lu, Y., Zhang, S., Wang, Y., Li, K., Huang, F., Shao, L., and Ji, R. Istr: End-to-end instance segmentation with transformers. *arXiv preprint arXiv:2105.00637*, 2021.

Huang, Z., Huang, L., Gong, Y., Huang, C., and Wang, X. Mask scoring r-cnn. In *CVPR*, 2019.

Jampani, V., Sun, D., Liu, M.-Y., Yang, M.-H., and Kautz, J. Superpixel sampling networks. In *ECCV*, 2018.

Jang, E., Gu, S., and Poole, B. Categorical reparameterization with gumbel-softmax. *ICLR*, 2017.

Khan, S. S. and Ahmad, A. Cluster center initialization algorithm for k-means clustering. *Pattern recognition letters*, 25(11):1293–1302, 2004.

Kirillov, A., Levinkov, E., Andres, B., Savchynskyy, B., and Rother, C. Instancecut: from edges to instances with multicut. In *CVPR*, 2017.

Kirillov, A., Girshick, R., He, K., and Dollár, P. Panoptic feature pyramid networks. In *CVPR*, 2019a.

Kirillov, A., He, K., Girshick, R., Rother, C., and Dollár, P. Panoptic segmentation. In *CVPR*, 2019b.

Kirillov, A., Wu, Y., He, K., and Girshick, R. Pointrend: Image segmentation as rendering. In *CVPR*, 2020.

Kong, S. and Fowlkes, C. C. Recurrent pixel embedding for instance grouping. In *CVPR*, 2018.

Lazarow, J., Lee, K., Shi, K., and Tu, Z. Learning instance occlusion for panoptic segmentation. In *CVPR*, 2020.

Lee, C.-Y., Xie, S., Gallagher, P., Zhang, Z., and Tu, Z. Deeply-supervised nets. In *Artificial intelligence and statistics*, 2015.

Li, Q., Qi, X., and Torr, P. H. Unifying training and inference for panoptic segmentation. In *CVPR*, 2020.

Li, Y., Chen, X., Zhu, Z., Xie, L., Huang, G., Du, D., and Wang, X. Attention-guided unified network for panoptic segmentation. In *CVPR*, 2019.

Li, Y., Zhao, H., Qi, X., Wang, L., Li, Z., Sun, J., and Jia, J. Fully convolutional networks for panoptic segmentation. In *CVPR*, 2021.

Li, Z., Wang, W., Xie, E., Yu, Z., Anandkumar, A., Alvarez, J. M., Luo, P., and Lu, T. Panoptic segformer: Delving deeper into panoptic segmentation with transformers. In *CVPR*, 2022.

Liang, C., Wang, W., Miao, J., and Yang, Y. Gmmseg: Gaussian mixture based generative semantic segmentation models. In *NeurIPS*, 2022a.

Liang, J., Wang, Y., Chen, Y., Yang, B., and Liu, D. A triangulation-based visual localization for field robots. *IEEE/CAA Journal of Automatica Sinica*, 2022b.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft coco: Common objects in context. In *ECCV*, 2014.

Liu, D., Cui, Y., Tan, W., and Chen, Y. Sg-net: Spatial granularity network for one-stage video instance segmentation. In *CVPR*, 2021a.

Liu, D., Cui, Y., Yan, L., Mousas, C., Yang, B., and Chen, Y. Densernet: Weakly supervised visual localization using multi-scale feature aggregation. In *AAAI*, 2021b.

Liu, D., Liang, J., Geng, T., Loui, A., and Zhou, T. Tripartite feature enhanced pyramid network for dense prediction. *TIP*, 2023.

Liu, H., Peng, C., Yu, C., Wang, J., Liu, X., Yu, G., and Jiang, W. An end-to-end network for panoptic segmentation. In *CVPR*, 2019.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., and Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021c.

Liu, Z., Mao, H., Wu, C.-Y., Feichtenhofer, C., Darrell, T., and Xie, S. A convnet for the 2020s. In *CVPR*, 2022.

Long, J., Shelhamer, E., and Darrell, T. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015.

Nathan Silberman, Derek Hoiem, P. K. and Fergus, R. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

Neven, D., Brabandere, B. D., Proesmans, M., and Gool, L. V. Instance segmentation by jointly optimizing spatial embeddings and clustering bandwidth. In *CVPR*, 2019.

Ronneberger, O., Fischer, P., and Brox, T. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241. Springer, 2015.

Strudel, R., Garcia, R., Laptev, I., and Schmid, C. Segmenter: Transformer for semantic segmentation. In *ICCV*, 2021.

Stutz, D., Hermans, A., and Leibe, B. Superpixels: An evaluation of the state-of-the-art. *CVPR*, 2018.

Tu, W.-C., Liu, M.-Y., Jampani, V., Sun, D., Chien, S.-Y., Yang, M.-H., and Kautz, J. Learning superpixels with segmentation-aware affinity loss. In *CVPR*, 2018.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *NeurIPS*, 2017.

Vattani, A. K-means requires exponentially many iterations even in the plane. In *Annual Symposium on Computational Geometry*, 2009.

Wang, H., Luo, R., Maire, M., and Shakhnarovich, G. Pixel consensus voting for panoptic segmentation. In *CVPR*, 2020a.

Wang, H., Zhu, Y., Green, B., Adam, H., Yuille, A., and Chen, L.-C. Axial-deeplab: Stand-alone axial-attention for panoptic segmentation. In *ECCV*, 2020b.

Wang, H., Zhu, Y., Adam, H., Yuille, A., and Chen, L.-C. MaX-DeepLab: End-to-end panoptic segmentation with mask transformers. In *CVPR*, 2021a.

Wang, H., Zhu, Y., Adam, H., Yuille, A., and Chen, L.-C. Max-deeplab: End-to-end panoptic segmentation with mask transformers. In *CVPR*, 2021b.

Wang, W., Zhou, T., Yu, F., Dai, J., Konukoglu, E., and Van Gool, L. Exploring cross-image pixel contrast for semantic segmentation. In *ICCV*, 2021c.

Wang, W., Liang, J., and Liu, D. Learning equivariant segmentation with instance-unique querying. In *NeurIPS*, 2022.

Wang, W., Han, C., Zhou, T., and Liu, D. Visual recognition with deep nearest centroids. In *ICLR*, 2023.

Wang, X., Girshick, R., Gupta, A., and He, K. Non-local neural networks. In *CVPR*, 2018.

Wang, X., Kong, T., Shen, C., Jiang, Y., and Li, L. Solo: Segmenting objects by locations. In *ECCV*, 2020c.

Wang, X., Zhang, R., Kong, T., Li, L., and Shen, C. SOLOv2: Dynamic and fast instance segmentation. In *NeurIPS*, 2020d.

Xie, E., Wang, W., Yu, Z., Anandkumar, A., Alvarez, J. M., and Luo, P. Segformer: Simple and efficient design for semantic segmentation with transformers. *NeurIPS*, 2021.

Xiong, Y., Liao, R., Zhao, H., Hu, R., Bai, M., Yumer, E., and Urtasun, R. Upsnet: A unified panoptic segmentation network. In *CVPR*, 2019.

Yang, F., Sun, Q., Jin, H., and Zhou, Z. Superpixel segmentation with fully convolutional networks. In *CVPR*, 2020.

Yu, F. and Koltun, V. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016.

Yu, Q., Wang, H., Kim, D., Qiao, S., Collins, M., Zhu, Y., Adam, H., Yuille, A., and Chen, L.-C. Cmt-deeplab: Clustering mask transformers for panoptic segmentation. In *CVPR*, 2022a.

Yu, Q., Wang, H., Qiao, S., Collins, M., Zhu, Y., Adam, H., Yuille, A., and Chen, L.-C. k-means mask transformer. *ECCV*, 2022b.

Zhang, W., Pang, J., Chen, K., and Loy, C. C. K-net: Towards unified image segmentation. *NeurIPS*, 2021.

Zhao, H., Zhang, Y., Liu, S., Shi, J., Change Loy, C., Lin, D., and Jia, J. Psanet: Point-wise spatial attention network for scene parsing. In *ECCV*, 2018.

Zheng, S., Jayasumana, S., Romera-Paredes, B., Vineet, V., Su, Z., Du, D., Huang, C., and Torr, P. H. Conditional random fields as recurrent neural networks. In *ICCV*, 2015.

Zheng, S., Lu, J., Zhao, H., Zhu, X., Luo, Z., Wang, Y., Fu, Y., Feng, J., Xiang, T., Torr, P. H., et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In *CVPR*, 2021.

Zhou, B., Zhao, H., Puig, X., Fidler, S., Barriuso, A., and Torralba, A. Scene parsing through ade20k dataset. In *CVPR*, 2017.

Zhou, T., Wang, W., Liu, S., Yang, Y., and Van Gool, L. Differentiable multi-granularity human representation learning for instance-aware human semantic parsing. In *CVPR*, 2021.

Zhu, F., Zhu, Y., Zhang, L., Wu, C., Fu, Y., and Li, M. A unified efficient pyramid transformer for semantic segmentation. In *ICCV*, 2021a.

Zhu, L., She, Q., Zhang, B., Lu, Y., Lu, Z., Li, D., and Hu, J. Learning the superpixel in a non-iterative and lifelong manner. In *CVPR*, 2021b.

In this document, we provide additional experimental results and analysis, pseudo code, more implementation details and discussions. It is organized as follows:

A. More Experimental Details

We provide more experimental results of CLUSTSEG (with Swin-B backbone) on five datasets: COCO panoptic (Kirillov et al., 2019b) `val` for **panoptic segmentation** in Fig. 7, COCO (Lin et al., 2014) `val2017` for **instance segmentation** in Fig. 8, ADE20K (Zhou et al., 2017) `val` for **semantic segmentation** in Fig. 9, and NYUv2 (Nathan Silberman & Fergus, 2012) as well as BSDS500 (Arbelaez et al., 2011) for **superpixel segmentation** in Fig. 5 and Fig. 10. Our results demonstrate that CLUSTSEG can learn and discover, from the underlying characteristics of the data, the division principle of pixels, hence yielding strong performance across various core image segmentation tasks.

Implementation Details. CLUSTSEG is implemented in PyTorch. All the backbones are initialized using corresponding weights pre-trained on ImageNet-1K/-22K (Deng et al., 2009), while the remaining layers are randomly initialized. We train all our models using AdamW optimizer and cosine annealing learning rate decay policy. For panoptic, instance, and semantic segmentation, we adopt the default training recipes of MMDetection (Chen et al., 2019b).

A.1. Panoptic Segmentation

Dataset. COCO panoptic (Kirillov et al., 2019b) is considered as a standard benchmark dataset in the field of panoptic segmentation, providing a rich and diverse set of images for training and evaluation. It is a highly advanced and sophisticated dataset that utilizes the full spectrum of annotated images from COCO (Lin et al., 2014) dataset. COCO panoptic encompasses the 80 "thing" categories as well as an additional diligently annotated set of 53 "stuff" categories. To ensure the integrity and coherence of the dataset, any potential overlapping categories between the two aforementioned tasks are meticulously resolved. Following the practice of COCO, COCO Panoptic is divided into 115K/5K/20K images for `train`/`val`/`test` split.

Training. Following (Carion et al., 2020; Wang et al., 2021a; Yu et al., 2022b; Cheng et al., 2022a), we set the total number of cluster seeds (*i.e.*, queries) as 128, in which 75 are for "thing" and 53 are for "stuff". During training, we optimize the following objective:

$$\mathcal{L}^{\text{Panoptic}} = \lambda^{\text{th}}\mathcal{L}^{\text{th}} + \lambda^{\text{st}}\mathcal{L}^{\text{st}} + \lambda^{\text{aux}}\mathcal{L}^{\text{aux}}, \tag{12}$$

where $\mathcal{L}^{\text{th}}$ and $\mathcal{L}^{\text{st}}$ are loss functions for things and stuff. For a fair comparison, we follow (Yu et al., 2022b; Wang et al., 2021b) to additionally employ an auxiliary loss that is computed as a weighted summation of four loss terms, *i.e.*, a PQ-style loss, a mask-ID cross-entropy loss, an instance discrimination loss, and a semantic segmentation loss. We refer to (Wang et al., 2021b; Yu et al., 2022b) for more details about $\mathcal{L}^{\text{aux}}$. The coefficients λ^{th}, λ^{st} and λ^{aux} are set as: $\lambda^{\text{th}} = 5$, $\lambda^{\text{st}} = 3$, and $\lambda^{\text{aux}} = 1$. In addition, the final "thing" centers are feed into a small FFN for semantic classification, trained with a binary cross-entropy loss.

Qualitative Results. CLUSTSEG is capable to achieve appealing performance in various challenging scenarios. Specifically, in the *restroom* example (see Fig. 7 row #2 col #1 and #2), it perfectly segments the object instances and preserves more details of backgrounds within a highly intricate indoor scenario; in the *zebra* example (see Fig. 7 row #5 col #1 and #2), CLUSTSEG successfully recognizes two distinct zebras with similar patterns as well as the grass backgrounds; in the *person* example (see Fig. 7 row #3 col #3 and #4), CLUSTSEG differentiates the person in the dense crowd and identifies the complex backgrounds.

A.2. Instance Segmentation

Dataset. We use COCO (Lin et al., 2014), the golden-standard dataset for instance segmentation. It has dense annotations for 80 object categories, including common objects such as people, animals, furniture, vehicles. The images in the dataset

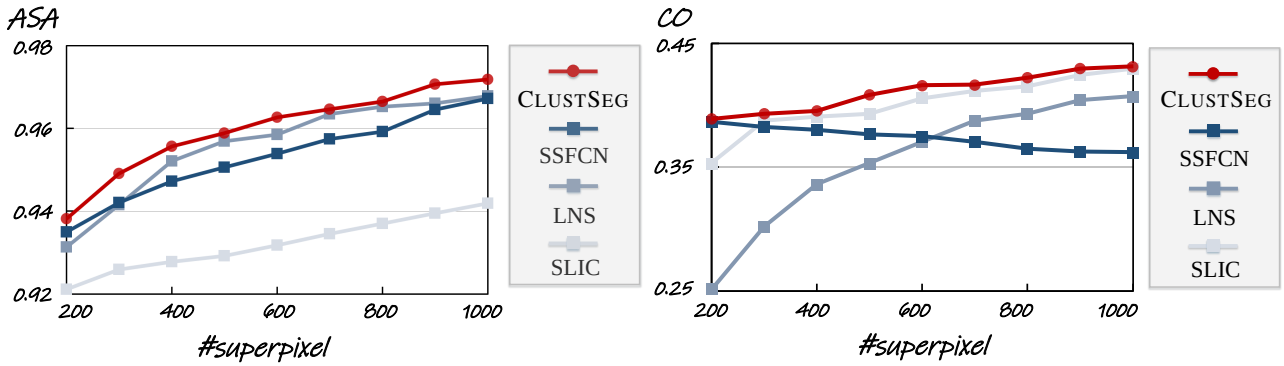

Figure 5. CLUSTSEG reaches the best ASA and CO scores on NYUv2 (Nathan Silberman & Fergus, 2012) `test` (see §A.4 for details).

are diverse, covering a wide range of challenging indoor and outdoor scenes. As standard, we use `train2017` split (115K images) for training, `val2017` (5K images) for validation, and `test-dev` (20K images) for testing. All the results in the main paper are reported for `test-dev`.

Training. For a fair comparison, we follow the training protocol in (Cheng et al., 2022a): 1) the number of instance centers is set to 100; 2) a combination of the binary cross-entropy loss and the dice Loss is used as the optimization objective. Their coefficients are set to 5 and 2, respectively. In addition, the final instance centers are feed into a small FFN for semantic classification, trained with a binary cross-entropy loss.

Qualitative Results. Consistent to panoptic segmentation, CLUSTSEG also demonstrates strong efficacy in instance segmentation. For instance, in the *elepants* example (see Fig. 8 row #5 col #3 and #4), CLUSTSEG successfully separates apart a group of elephants under significant occlusions and similar appearance; in the *river* example (see Fig. 8 row #2 col #3 and #4), CLUSTSEG effectively distinguishes the highly-crowded and occluded person as well.

A.3. Semantic Segmentation

Dataset. ADE20K (Zhou et al., 2017) is a large-scale scene parsing benchmark that covers a wide variety of indoor and outdoor scenes annotated with 150 semantic categories (*e.g.*, door, cat, sky) . It is divided into 20K/2K/3K images for `train`/`val`/`test`. The images cover many daily scenes, making it a challenging dataset for semantic segmentation.

Training. In semantic segmentation, the number of cluster seeds is set to the number of semantic categories, *i.e.*, 150 for ADE20K. We adopt the same loss function as (Zhang et al., 2021; Cheng et al., 2022a; Strudel et al., 2021) by combining the standard cross-entropy loss with an auxiliary dice loss. By default, the coefficients for the two losses are set to 5 and 1, respectively.

Qualitative Results. When dealing with both indoor (see Fig. 9 row #1 col #3 and #4) and outdoor (see Fig. 9 row #2 col #3 and #4) scenarios, CLUSTSEG delivers highly accurate results. Especially, for the challenging outdoor settings. CLUSTSEG can robustly delineate the delicacy of physical complexity across the scenes, where Mask2Former, a recent top-leading segmentation algorithm, generates a large array of wrongful mask predictions.

A.4. Superpixel Segmentation

Dataset. For superpixel segmentation, we utilize two standard datasets (*i.e.*, BSDS500 (Arbelaez et al., 2011) and NYUv2 (Nathan Silberman & Fergus, 2012)). BSDS500 contains 500 natural images with pixel-wise semantic annotations. These image are divided into 200/100/200 for `train`/`val`/`test`. Following (Yang et al., 2020; Tu et al., 2018), we train our model using the combination of all images in `train` and `val`, and run evaluation on `test`. NYUv2 dataset is originally proposed for indoor scene understanding tasks, which contains 1,449 images with object instance labels. By removing the unlabelled regions near the image boundary, a subset of 400 test images with size 608×448 are collected for superpixel evaluation. As in conventions (Yang et al., 2020), we directly apply the models of SSFCN (Yang et al., 2020), LNSnet (Zhu et al., 2021b) and our CLUSTSEG trained on BSDS500 to 400 NYUv2 images without any fine-tuning, to test the generalizability of the learning-based methods.

Training. For superpixel query-initialization, we use a grid sampler to automatically sample a specified number of position-embedded pixel features as superpixel seeds. The network is trained jointly with the smooth L1 loss, and SLIC loss (Yang

et al., 2020). They are combined with coefficients of 10 for smooth L1 and 1 for SLIC losses.

Quantitative Results. Fig. 5 provides additional performance comparison of CLUSTSEG against both traditional (*i.e.*, SLIC (Achanta et al., 2012)) and deep learning-based (*i.e.*, SSFCN (Yang et al., 2020), LNSnet (Zhu et al., 2021b)) superpixel segmentation algorithms on NYUv2 (Nathan Silberman & Fergus, 2012) `test`. We can observe that CLUSTSEG consistently outperforms all the competitors in terms of ASA and CO. This also verifies stronger generalizability of CLUSTSEG over all the other learning-based competitors.

Qualitative Results. Overall, CLUSTSEG can capture rich details in images and tends to create compact fine-grained results that closely align with object boundaries (see Fig. 10). Across the different numbers of superpixels (*i.e.*, 200 to 1000), CLUSTSEG yields stable and impressive performance for various landscapes and objects.

A.5. Failure Case Analysis

As shown in Fig. 11, we summarize the most representative failure cases and draw conclusions regarding their characteristic patterns that can lead to subpar results. Observedly, our algorithm struggles to separate objects from backgrounds in a number of incredibly complex scenarios (*i.e.*, highly similar and occluded instances, objects with complex topologies, small objects, highly deformed objects, and distorted backgrounds). Developing more robust and powerful clustering algorithms may help alleviate these issues.

B. More Ablative Studies

In this section, we provide more ablative studies regarding *Dreamy-Start* query-initialization in Algorithm 1 and *Recurrent Cross-Attention* for recursive clustering in Algorithm 2.

B.1. Recurrent Cross-Attention

We perform further ablation studies on our non-recurrent cross-attention for the panoptic segmentation task. The results are summarized in the table below, where PQ (%) is reported. As seen, simply stacking multiple non-recurrent cross-attention layers cannot achieve similar performance to our recurrent cross-attention with the same number of total iterations. Note that using multiple non-recurrent cross-attention layers even causes extra learnable parameters. EM is an iterative computational procedure for progressively estimating the local representatives of data samples in a given embedding space. When using multiple non-recurrent cross-attention layers, we essentially conduct one-step clustering on different embedding spaces, since the parameters are not shared among different cross-attention layers. This does not follow the nature of EM clustering, hence generating inferior results.

Table 5. Ablative study of **recurrent cross-Attention** *vs.* **non-recurrent cross-attention** over ResNet-50 (He et al., 2016) on COCO Panoptic (Kirillov et al., 2019b) `val` (see §B.1 for details).

Iteration (T)	Recurrent cross-attention	Multiple non-recurrent	Additional learnable parameter
1	53.8	53.8	-
2	54.1	53.8	1.3M
3	54.3	53.9	2.8M
4	54.3	53.9	4.3M
5	54.3	54.0	5.7M

B.2. Query Initialization

We report the panoptic segmentation results with more iterations when learning queries as free parameters. As seen in Tab. 6, when learning initial queries as free parameters, even if using more iterations, performance degradation is still observed. Actually, the performance of iterative clustering algorithms heavily relies on the selection of initial seeds due to their stochastic nature (Hamerly & Elkan, 2002; Celebi et al., 2013; Khan & Ahmad, 2004). This issue, called initial starting conditions, has long been a focus in the field of data clustering. It is commonly recognized that the effect of initial starting conditions cannot be alleviated by simply using more iterations. And this is why many different initialization methods are developed for more effective clustering (Khan & Ahmad, 2004).

Table 6. Ablative study of **query initialization** over ResNet-50 (He et al., 2016) on COCO Panoptic (Kirillov et al., 2019b) `val` (see §B.2 for details).

Method	Iteration (T)	PQ	PQ^{Th}	PQ^{St}	AP^{Th}_{pan}	$mIoU_{pan}$
Dreamy-Start	3	54.3	60.4	45.8	42.2	63.8
Free parameters	3	53.5	59.6	45.1	41.0	60.5
Free parameters	3	53.7	59.9	45.3	54.2	61.1
Free parameters	3	53.8	60.1	45.4	41.6	61.4

B.3. Deep Supervision

We adopt deep supervision to train every E-step of each recurrent cross-attention. A similar strategy is widely employed in previous segmentation models and other Transformer counterparts, *e.g.*, Mask2Former (Cheng et al., 2022a), kMaX-Deeplab (Yu et al., 2022b). We ablate the effect of such a deep supervision strategy for panoptic segmentation in Tab. 7a. Moreover, we also show the accuracy of segmentation predictions from different iterations of the last recurrent cross-attention layer in Tab. 7b. We additionally provide visualization of segmentation results in different stages in Fig. 6.

Table 7. Ablative studies of **deep supervision** over ResNet-50 (He et al., 2016) on COCO Panoptic (Li et al., 2022) `val` (see §B.3).

Variant	PQ	PQ^{Th}	PQ^{St}	AP^{Th}_{pan}	$mIoU_{pan}$
Only final E-step of each recurrent cross-attention	53.0	59.6	43.7	41.7	61.2
Deep supervision	54.3	60.4	45.8	42.2	63.8

(a) Supervision variants

Iteration (T)	PQ	PQ^{Th}	PQ^{St}	AP^{Th}_{pan}	$mIoU_{pan}$
1	53.8	59.7	45.6	41.6	63.1
2	54.0	60.1	45.6	41.9	63.4
3	54.3	60.4	45.8	42.2	63.8

(b) Iterations of the last recurrent cross-attention layer

C. Pseudo Code

In this section, we provide pseudo-code of *Dreamy-Start* query-initialization in Algorithm 1 and *Recurrent Cross-Attention* for recursive clustering in Algorithm 2.

D. Discussion

Asset License and Consent. We apply five closed-set image segmentation datasets, *i.e.*, MS COCO (Lin et al., 2014), MS COCO Panoptic (Kirillov et al., 2019b), ADE20K (Zhou et al., 2017), BSDS500 (Arbelaez et al., 2011) and NYUv2 (Nathan Silberman & Fergus, 2012) They are all publicly and freely available for academic purposes. We implement all models with MMDetection (Contributors, 2019), MMSegmentation (Contributors, 2020) and Deeplab2 (Chen et al., 2017; Wang et al., 2021a; Yu et al., 2022b) codebases. MS COCO (https://cocodataset.org/) is released under a CC BY 4.0; MS COCO Panoptic (https://github.com/cocodataset/panopticapi) is released under a CC BY 4.0; ADE20K (https://groups.csail.mit.edu/vision/datasets/ADE20K/) is released under a CC BSD-3; All assets mentioned above release annotations obtained from human experts with agreements. MMDetection (https://github.com/open-mmlab/mmdetection), MMSegmentation (https://github.com/open-mmlab/mmsegmentation) and Deeplab2 codebases (https://github.com/google-research/deeplab2) are released under Apache-2.0.

Limitation Analysis. One limitation of our algorithm arises from the extra clustering loops in each training iteration, as they may reduce the computation efficiency in terms of time complexity. However, in practice, we observe that three recursive clusterings are sufficient for global model convergence, incurring only a minor computational overhead, *i.e.*, 5.19% reduction in terms of training speed. We will dedicate ourselves to the development of potent algorithms that are more efficient and effective.

Broader Impact. This work develops a universal and transparent segmentation framework, which unifies different image segmentation tasks from a clustering perspective. We devise a novel cluster center initialization scheme as well as a neural solver for iterative clustering, hence fully exploiting the fundamental principles of recursive clustering for pixel grouping. Our algorithm has demonstrated its effectiveness over a variety of famous models in four core segmentation tasks (*i.e.*, panoptic, instance, semantic, and superpixel segmentation). On the positive side, our approach has the potential to benefit a wide variety of applications in the real world, such as autonomous vehicles, robot navigation, and medical imaging. On

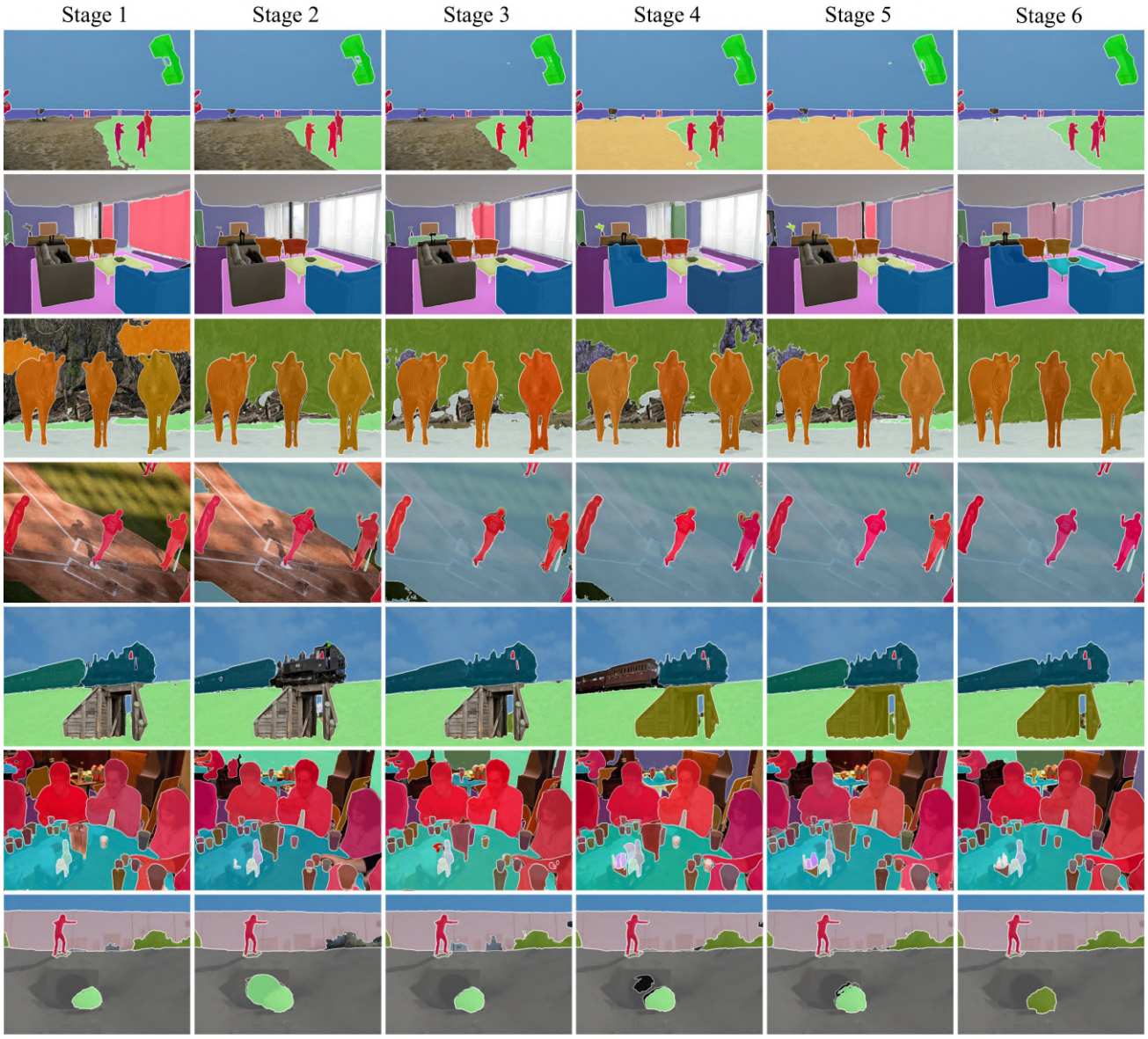

Figure 6. **Visualization of panoptic segmentation in different stages** results on COCO Panoptic (Kirillov et al., 2019b) `val` with CLUSTSEG with Swin-B (Liu et al., 2021c) backbone. See §B.3 for details.

the other side, erroneous predictions in real-world applications (*i.e.*, medical imaging analysis and any tasks involving autonomous vehicles) give rise to concerns about the safety of human beings (Liang et al., 2022b; Cheng et al., 2022c; 2023). In order to avoid this potentially negative effect on society and the community, we suggest proposing a highly stringent security protocol in the event that our approach fails to function properly in real-world applications.

Algorithm 1 Pseudo-code of *Dreamy-Start* for query initialization in a PyTorch-like style.

```
"""
feats: output feature of backbone, shape: (channels, height, width)
memory: a set of queues storing class-aware pixel embeddings, each has a shape of (num_feats, channels)
num_sp: number of superpixels
FFN: feedforward network, PE: position embedding
"""

# scene-agnostic center initialization (Eq.7)
def scene_agnostic_initialization(memory):

   mem_feats = Avg_Pool(memory)

   semantic_centers = FFN(mem_feats)

  return semantic_centers

# scene-adaptive center initialization (Eq.8)
def scene_adaptive_initialization(feats):

   feats = PE(feats)

   instance_centers = FFN(feats)

   return instance_centers

# superpixel center initialization (Eq.9)
def superpixel_initialization(feats):

   _, H, W = feats.shape

   feats = PE(feats)

   # Grid sampler of num_sp superpixels
   f = torch.sqrt(num_sp/H/W)
   x = torch.linspace(0, W, torch.int(W*f))
   y = torch.linspace(0, H, torch.int(H*f))
   meshx, meshy = torch.meshgrid((x, y))
   grid = torch.stack((meshy, meshx), 2).unsqueeze(0)
   feats = grid_sample(feats, grid).view(-1, channels)

   superpixel_centers = FFN(feats)

   return superpixel_centers
```

Algorithm 2 Pseudo-code of *Recurrent Cross-attention* for *Recursive Clustering* in a PyTorch-like style.

```
"""
feats: output feature of backbone, shape: (batch_size, channels, height, width)
C: cluster centers, shape: (batch_size, num_clusters, dimention)
T: iteration number for recursive clustering
"""

# One-step cross attention in Eq.10
def recurrent_cross_attention_layer(Q, K, V):

   # E-step
   output = torch.matmul(Q, K.transpose(-2, -1))
   M = torch.nn.functional.softmax(output, dim=-2)

   # M-step
   C = torch.matmul(M, V)

   return C

# Recurrent cross-attention in Eq.11
def RCross_Attention(feats, C, T):

   Q = nn.Linear(C)
   K = nn.Linear(feats)
   V = nn.Linear(feats)
   C = recurrent_cross_attention_layer(Q, K, V)

   for _ in range(T-1):
      Q = nn.Linear(C)
      C = recurrent_cross_attention_layer(Q, K, V)

   return C
```

Figure 7. **Qualitative panoptic segmentation** results on COCO panoptic (Kirillov et al., 2019b) `val`. CLUSTSEG with Swin-B (Liu et al., 2021c) backbone achieves **59.0%** *PQ*. See §A.1 for details.

Figure 8. **Qualitative instance segmentation** results on COCO (Lin et al., 2014) `val2017`. CLUSTSEG with Swin-B (Liu et al., 2021c) backbone achieves **49.1%** *AP*. See §A.2 for details.

Figure 9. **Qualitative semantic segmentation results** on ADE20K (Zhou et al., 2017) `val`. CLUSTSEG with Swin-B (Liu et al., 2021c) backbone achieves **57.4** *mIoU*. See §A.3 for details.

Figure 10. **Qualitative superpixel segmentation** results on BSDS500 (Arbelaez et al., 2011) `test`. For each test image, we show segmentation results with three different numbers of superpixels (*i.e.*, 200, 500, and 1000). See §A.4 for details.

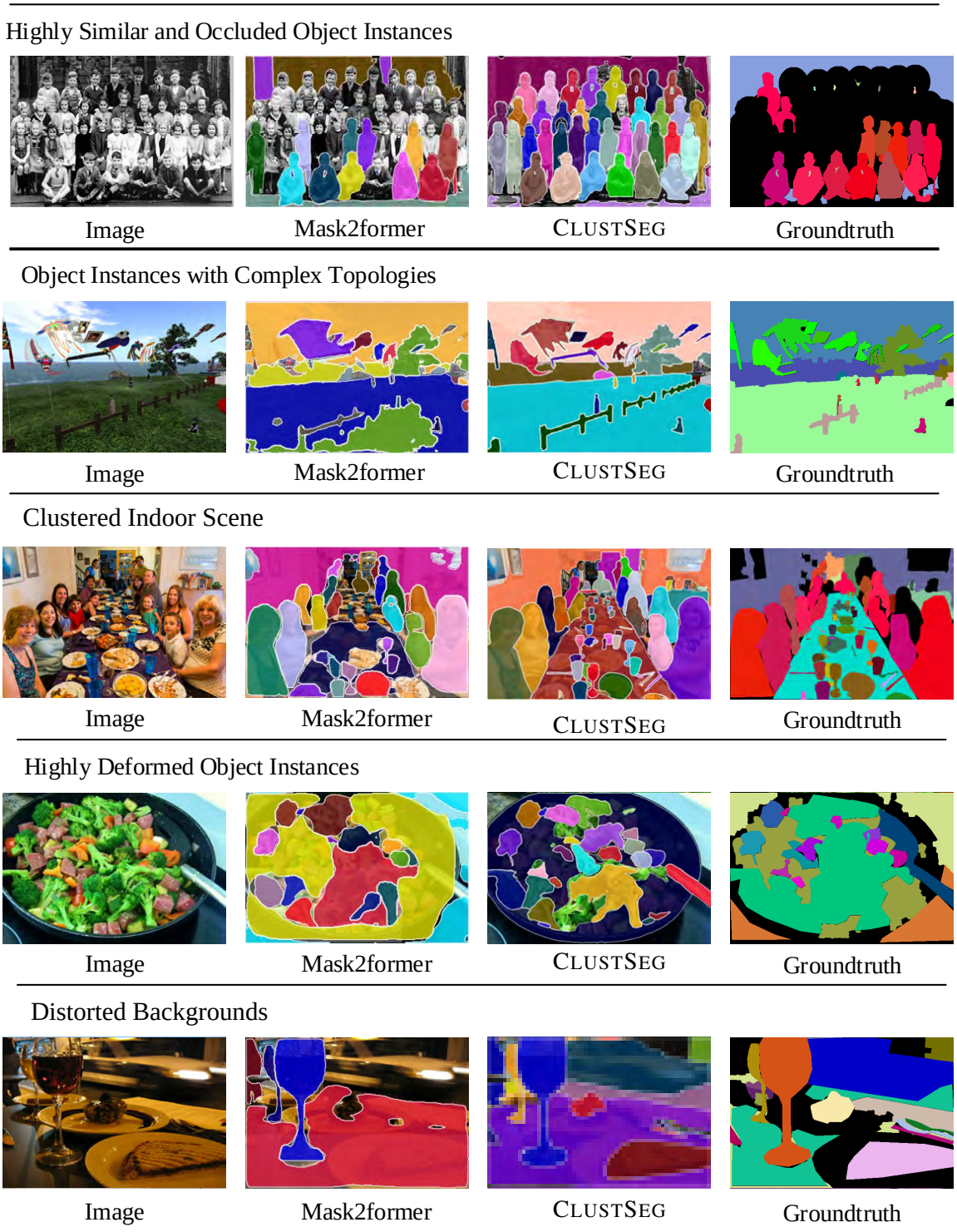

Figure 11. **Failure Cases** on COCO panoptic (Kirillov et al., 2019b) `val`. See §A.5 for details.