
Count Anything

Mengqi Lei¹, Shuokun Cheng², Wei Bao¹, Shaoyi Du³,
Jun-Hai Yong¹, Siqi Li¹, and Yue Gao¹

¹{BNRist, THUIBCS, BLBCI, School of Software}, Tsinghua University

²School of Computer Science and Technology, China University of Geosciences, Wuhan

³State Key Laboratory of Human-Machine Hybrid Augmented Intelligence,
National Engineering Research Center for Visual Information and Applications,
and Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University

leimq25@mails.tsinghua.edu.cn, chen Yuekun@cug.edu.cn, baoweivvv@gmail.com
dushaoyi@xjtu.edu.cn, yongjh@tsinghua.edu.cn, lisiqi19971013@gmail.com
kevin.gaoy@gmail.com

Abstract

Object counting remains largely fragmented across domain-specific datasets and task formulations, despite the rapid progress of generalist vision models. Existing counting models are often tailored to particular scenarios, such as crowds, vehicles, cells, crops, or remote-sensing objects, and therefore struggle to generalize across categories, visual domains, object scales, and density distributions. In this paper, we study text-guided object counting across domains, where a model takes an image and a natural-language query as input and returns an instance-grounded set of target points whose cardinality gives the count. This formulation unifies category-conditioned counting with interpretable spatial localization. To support this setting, we construct **CLOC**, a **C**ross-domain **L**arge-scale **O**bject **C**ounting dataset that reorganizes diverse public data sources into a unified counting benchmark. CLOC covers six visual domains, including General Scene, Remote Sensing, Histopathology, Cellular Microscopy, Agriculture, and Microbiology, and contains about 220K images, 619 categories, and 15M object instances. Based on CLOC, we further propose **Count Anything**, a generalist model for text-guided object counting. Instead of using density maps as the final output, which is most widely used in current counting models, our Count Anything adopts discrete instance points and performs dual-granularity instance enumeration. A Region-level Sparse Counter provides object-level anchoring for large and sparse targets, while a Pixel-level Dense Counter captures small, crowded, and weakly bounded targets through dense point prediction. A point-centric supervision strategy enables learning from heterogeneous annotations, and Complementary Count Fusion combines both counters in a parameter-free manner. Extensive experiments show that Count Anything achieves strong counting accuracy and multi-domain generalization, substantially outperforming existing open-world counting methods. The code is available at: [count-anything](https://github.com/mengqilei/count-anything).

1 Introduction

Vision foundation models are driving visual systems from task-specific models in closed settings toward generalist models that can generalize across datasets, categories, and visual domains [1, 2, 3]. For object counting [4, 5, 6, 7], however, this transition remains incomplete. A generalist counting model must not only understand the target concept specified by the user, but also provide spatial evidence that supports the final count. This requires more than stronger network architectures: it also

calls for a unified task formulation, broad data coverage, and model designs that can handle diverse imaging conditions, object scales, and density distributions.

Object counting is one of the most fundamental tasks in visual understanding [8, 9, 10]. Given an image and a target concept, a counting model is expected to answer a simple question: how many instances of the target are present in the image? Yet existing counting research remains highly fragmented in practice. Most datasets [11, 12, 13, 14] and models [10, 15, 16] are built around specific scenarios, such as crowds, vehicles, remote-sensing objects, *etc.* These specialized settings provide clear task boundaries within their own domains, but they also bind models to particular visual domains, object scales, density distributions, and category spaces. A model that performs well in crowd counting may not directly generalize to small vehicles in remote-sensing imagery, dense cell nuclei in histopathology images, or crop organs with repetitive textures in agricultural images.

This fragmentation reveals a gap between existing counting systems and a generalist counting model. Such a model should be 1) **text-guided**, allowing users to specify the target with a category name or a natural-language query; 2) **instance-grounded**, so that the count is supported by localized instance predictions with confidence scores rather than an uninterpretable scalar; and 3) **cross-domain**, since real counting applications span highly diverse visual domains, including General Scene, Remote Sensing, Histopathology, Cellular Microscopy, Agriculture, and Microbiology.

Achieving this goal first requires addressing the data bottleneck. Counting annotations are inherently heterogeneous: some sources provide bounding boxes [17, 18, 19], some provide points [14, 12], and others describe instances with masks, polygons, or label maps [20, 21, 22]. Some datasets further use crowd or group labels for targets that are difficult to separate instance by instance, and use ignore-region or NegativeROA annotations for regions that should be excluded from counting supervision [23, 24, 25]. These protocols may be reasonable for their original tasks, but directly using them for precise counting conflicts with the one-instance-one-count supervision required by counting. Simply concatenating multi-source data would introduce inconsistencies in meaning, granularity, and even supervision targets. A counting data engine must therefore establish a stable constraint across sources: **each valid supervision unit should correspond to one independently countable instance.**

To this end, we construct **CLOC**, a **Cross-domain Large-scale Object Counting** dataset for text-guided counting across domains. To the best of our knowledge, CLOC is the **largest dataset to date** for text-guided object counting in terms of both image scale and instance annotations, containing approximately 220K images, 619 categories, and 15M instances. It reorganizes multiple public data sources into a unified category-specified counting format, covering six visual domains: General Scene, Remote Sensing, Histopathology, Cellular Microscopy, Agriculture, and Microbiology. CLOC provides a data foundation for generalist counting.

At the model level, generalist counting also requires rethinking the prediction representation. Existing counting methods can be broadly grouped into three paradigms. Detection-based counting [26, 13, 27] detects target instances and counts the detection results, providing instance interpretability and box supervision, but it tends to miss or merge small, dense, or weakly bounded objects. Density-estimation-based counting [28, 29, 4] predicts a density map and integrates it to obtain the count, making it effective for high-density distributions, but its output lacks an explicit one-to-one correspondence between predictions and instances. Point-regression-based counting [10, 15, 30] directly predicts discrete target points and uses the cardinality of the point set as the count, better matching instance-level counting in dense scenarios, but for large and sparse objects it may produce erroneous predictions due to a lack of object-level spatial constraints.

Although density estimation is widely used in traditional dense counting tasks, it does not align with the instance-grounded goal of our generalist counting model. A density map cannot establish one-to-one prediction-instance correspondence or naturally support instance-level confidence and localization-level error diagnosis. We therefore adopt discrete instance points as the final prediction form rather than using density maps. Under this formulation, detection-based counting and point-regression-based counting are naturally complementary: the former provides object extent and geometric anchors for large, sparse, and clearly bounded targets, while the latter is better suited for small, dense, and weakly bounded targets. This leads to our key observation: **generalist text-guided counting requires dual-granularity instance enumeration.** The model should combine region-level object anchoring with pixel-level dense recall, rather than forcing all targets to share a single prediction representation.

Based on CLOC and the above observation, we propose **Count Anything**, a generalist model for cross-domain text-guided counting. Given an image and a text query, the model outputs a set of instance points, and the final count is obtained as the cardinality of the point set. Count Anything first uses a Text-Conditioned Encoder to extract target-conditioned visual representations. It then builds two complementary counters: the Region-level Sparse Counter (RSC) predicts a compact set of target regions and uses their centers as counting points, while the Pixel-level Dense Counter (PDC) performs dense point-level enumeration on high-resolution features to recover small, crowded, and weakly bounded targets. During training, point-centric supervision enables learning from heterogeneous point and box annotations. During inference, Complementary Count Fusion (CCF) combines the two branches in a parameter-free manner, suppressing duplicate counts while preserving their complementarity.

Our contributions are summarized as follows:

- We propose Count Anything, a generalist model for text-guided object counting. It combines a Region-level Sparse Counter and a Pixel-level Dense Counter to couple region-level object anchoring with pixel-level dense instance enumeration.
- We design point-centric supervision and Complementary Count Fusion, enabling the model to learn from heterogeneous annotations and achieve complementary instance enumeration across sparse large objects and dense small objects.
- We construct CLOC, a large-scale dataset for text-guided counting across domains. It covers six visual domains with about 220K images, 619 categories, and 15M instances, providing a data foundation for training and evaluating generalist counting models.
- Extensive experiments show that Count Anything achieves superior counting accuracy and generalization across diverse visual domains and density distributions, substantially outperforming existing open-world counting methods.

2 CLOC Dataset

To support cross-domain text-guided object counting, we construct **CLOC**, a **Cross-domain Large-scale Object Counting** dataset for generalist text-guided counting across domains. Existing counting datasets are mostly built around specific scenarios or target types, such as crowds, vehicles, cells, or crops [14, 27, 31, 32], and thus provide limited support for training and evaluating generalist counting models across visual domains, target categories, object scales, and density distributions. Meanwhile, existing large-scale detection, segmentation, and visual grounding datasets provide rich region- or instance-level annotations [17, 23, 33, 34], but their original task objectives and annotation protocols do not directly satisfy the one-instance-one-count supervision required by counting.

To this end, CLOC reorganizes images, categories, and instance annotations from multiple public datasets around a text-guided counting task. In CLOC, category names are used as textual queries, so the category-specified counting protocol is treated as a concrete form of text-guided counting: given an image and a text query, the model is required to predict the number of instances corresponding to the queried target in the image. CLOC covers six visual domains: General Scene, Remote Sensing, Histopathology, Cellular Microscopy, Agriculture, and Microbiology, and contains about 220K images, 619 categories, and 15.356M instances. Compared with existing counting datasets, CLOC provides both a larger scale and broader cross-domain coverage, while preserving long-tailed category distributions and target-count variations from low-density to extremely dense scenes. It therefore provides a unified benchmark for training and evaluating generalist object counting models. More detailed dataset statistics and the construction pipeline are provided in Appendix C.

3 Count Anything Model

3.1 Method Overview

Problem formulation. Given an image I and a text query T , text-guided object counting aims to predict all visible instances corresponding to T . Instead of regressing a scalar count or integrating a density map, Count Anything predicts a set of instance points:

$$\hat{\mathcal{P}}_T = \{(\hat{p}_n, \hat{s}_n)\}_{n=1}^{\hat{N}}, \quad \hat{p}_n \in \mathbb{R}^2, \quad \hat{s}_n \in [0, 1], \quad (1)$$

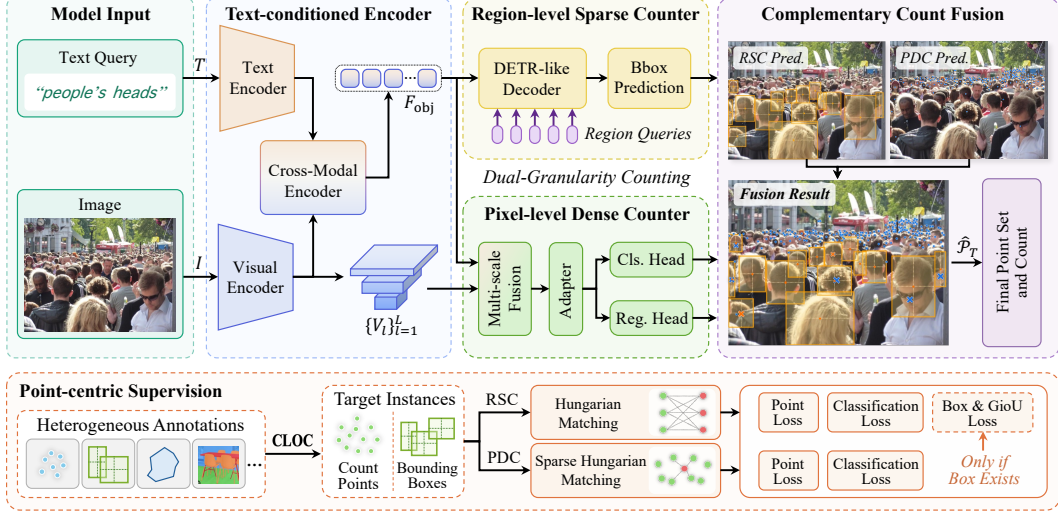


Figure 1: Overall framework of the proposed Count Anything.

where \hat{p}_n and \hat{s}_n denote the location and confidence score of the n -th predicted instance. The final count is the cardinality of this set, i.e., $\hat{c}_T = |\hat{\mathcal{P}}_T|$. This point-set formulation provides instance-level spatial evidence for counting and supports localization.

For a training sample, the ground-truth instances associated with query T are denoted by $\mathcal{Y}_T = \{y_j\}_{j=1}^N$, $y_j = (p_j, B_j, m_j)$, where p_j is the counting point, B_j is an optional bounding box (bbox), and $m_j \in \{0, 1\}$ indicates whether a ground-truth box is available. The counting point is defined as: $p_j = \begin{cases} \text{center}(B_j), & m_j = 1, \\ \text{point}_j, & m_j = 0. \end{cases}$ Thus, every valid instance is supervised by a point, while bounding boxes are used only when reliable box annotations exist. Heterogeneous annotations, including boxes, points, polygons, masks, rotated boxes, and label maps, can therefore be converted into counting points with optional boxes under a unified category-conditioned counting formulation.

Overall framework of Count Anything. A generalist counting model requires an output representation that is both instance-grounded and robust across object scales and densities. Density-estimation-based counting is effective for dense scenes, but its continuous response field does not naturally provide a one-to-one correspondence between predictions and instances. We therefore adopt discrete instance points as the final prediction form, i.e., a point-set prediction formulation. Within this formulation, detection-based and point-regression-based approaches are complementary: region-level predictions provide object extent, confidence, and geometric anchors for large or sparse targets, while dense point regression better captures small, crowded, or weakly bounded targets without estimating a complete boundary for each instance. This motivates our dual-granularity design.

As shown in Fig. 1, Count Anything contains two complementary counters: a Region-level Sparse Counter (RSC) and a Pixel-level Dense Counter (PDC). RSC performs sparse region-level enumeration, while PDC performs dense point-level enumeration. Their predictions are merged by Complementary Count Fusion (CCF) during inference. The overall pipeline is:

$$(I, T) \xrightarrow{\Phi_{\text{enc}}} (F_{\text{obj}}, \{V_l\}_{l=1}^L) \rightarrow \begin{cases} \mathcal{R} = \text{RSC}(F_{\text{obj}}), \\ \mathcal{D} = \text{PDC}(F_{\text{obj}}, \{V_l\}_{l=1}^L), \end{cases} \xrightarrow{\text{CCF}} \hat{\mathcal{P}}_T. \quad (2)$$

Here, Φ_{enc} is the text-conditioned encoder, F_{obj} is the target-conditioned object-level representation, and $\{V_l\}_{l=1}^L$ are multi-scale visual features. RSC directly operates on F_{obj} , while PDC first fuses F_{obj} with high-resolution visual features to construct a pixel-level counting representation F_{pix} .

RSC outputs region-level instance hypotheses: $\mathcal{R} = \{(\hat{B}_i^r, \hat{p}_i^r, \hat{s}_i^r)\}_{i=1}^{Q_r}$, $\hat{p}_i^r = \text{center}(\hat{B}_i^r)$, where Q_r is the number of region candidates, and \hat{B}_i^r , \hat{p}_i^r , and \hat{s}_i^r are the predicted region, its center point, and its confidence score, respectively. PDC outputs dense point-level hypotheses: $\mathcal{D} = \{(\hat{p}_k^d, \hat{s}_k^d)\}_{k=1}^{Q_d}$, where Q_d is the number of dense candidates. RSC and PDC have different prediction structures

and error patterns. RSC uses a relatively small number of candidates and can provide more stable object extents and region-level confidence, but it may suffer from insufficient recall for small objects and high-density scenes. PDC uses a large number of candidates and can complement RSC by recovering dense instances in high-resolution space, but it lacks explicit region extents and may produce duplicate predictions with RSC on clearly bounded targets. Therefore, Count Anything does not use either branch alone as the final output. Instead, it combines both branches by CCF during inference, so that the final point set benefits from both region-level anchoring and dense point recall.

3.2 Text-Conditioned Encoder

Count Anything adopts the image-language encoding backbone of pretrained SAM3 [35] as the Text-Conditioned Encoder. Given an image I and a text query T , the visual encoder extracts multi-scale features $\{V_l\}_{l=1}^L = E_v(I)$, and the text encoder produces query representation $e_T = E_t(T)$. A cross-modal encoder then fuses visual and text features to obtain the target-conditioned object representation: $F_{\text{obj}} = \Phi_{\text{obj}}(\{V_l\}_{l=1}^L, e_T)$, where Φ_{obj} denotes the cross-modal encoding process. The representation F_{obj} captures object visual features in the image while being conditioned on the query. It can therefore be used by RSC for region-level instance prediction and also provides target-conditioned information for constructing the pixel-level counting representation in PDC.

3.3 Region-level Sparse Counter

The Region-level Sparse Counter (RSC) adopts a DETR-like [36] region-query decoding architecture and formulates counting as region-level sparse enumeration. Given the target-conditioned object representation F_{obj} , RSC uses a set of region queries to predict candidate target regions. For the i -th region query, RSC outputs a foreground logit z_i^r and a bounding box \hat{B}_i^r : $(z_i^r, \hat{B}_i^r) = R_i(F_{\text{obj}})$, $i = 1, \dots, Q_r$. The confidence score is $\hat{s}_i^r = \sigma(z_i^r)$, where $\sigma(\cdot)$ denotes the sigmoid function. The counting point is the predicted box center: $\hat{p}_i^r = \text{center}(\hat{B}_i^r)$. Bounding boxes allow RSC to model object extent, scale, and spatial location, making it suitable for large, sparse, or clearly bounded targets. However, a limited set of region queries may be insufficient for dense cell nuclei, small remote sensing vehicles, microbial colonies, or repetitive crop organs. These cases are handled by PDC with dense point prediction.

3.4 Pixel-level Dense Counter

The Pixel-level Dense Counter (PDC) is designed for small, densely distributed, and weakly bounded objects that are difficult for RSC to cover. It operates on a dense counting representation and generates discrete point candidates.

Multi-scale fusion. Since F_{obj} is optimized for region-level reasoning and has limited spatial resolution, PDC constructs a pixel-level counting representation by fusing target-conditioned information with higher-resolution visual features:

$$F_{\text{pix}} = \Phi_{\text{pix}}(\text{Align}(F_{\text{obj}}), \{\text{Align}(V_l)\}_{l \in \mathcal{S}}), \quad (3)$$

where $\mathcal{S} \subseteq \{1, \dots, L\}$ denotes the set of high-resolution features involved in pixel-level fusion, $\text{Align}(\cdot)$ denotes the scale transformation that aligns features of different resolutions onto a common spatial grid, and Φ_{pix} denotes a lightweight multi-scale fusion process built with convolutional layers. In this way, F_{pix} simultaneously incorporates the target-conditioned information from F_{obj} and the local spatial details from high-resolution visual features.

Feature adaptation. After obtaining F_{pix} , PDC first uses a feature adapter composed of multiple residual blocks to transform it into counting features that are more suitable for point-level prediction: $G_{\text{pix}} = A_{\text{pdc}}(F_{\text{pix}})$. This adapter preserves the spatial grid structure and is used to enhance the local representations required for foreground discrimination and point localization.

Point prediction. We then proceed to the point prediction decoding stage. Inspired by the point-regression formulation of P2PNet [10], we do not directly predict absolute coordinates at each location in an unconstrained manner. Instead, we define fixed anchor points on the dense spatial grid and predict coordinate offsets and foreground confidence relative to them. Let $\Omega = \{u_k\}_{k=1}^{Q_d}$ be the dense spatial grid, where u_k denotes the k -th spatial location and Q_d is the number of dense candidates. Each location u_k is mapped to a predefined anchor point $a_k = \pi(u_k)$ in the input-image coordinate system, which provides the initial spatial location for the k -th dense candidate. Let $g_k = G_{\text{pix}}(u_k)$

denote the counting feature at location u_k . PDC uses a classification head and a regression head to predict the classification logits and coordinate offset for this anchor, *i.e.*,

$$z_k^d = D_{\text{cls}}(g_k), \quad \Delta_k = D_{\text{reg}}(g_k), \quad (4)$$

where D_{cls} is the point classification head, D_{reg} is the point regression head, $z_k^d \in \mathbb{R}^2$ denotes the binary logits for background-foreground classification, and $\Delta_k \in \mathbb{R}^2$ is the two-dimensional coordinate offset relative to the anchor point a_k . The final candidate point and confidence score are given by: $\hat{p}_k^d = a_k + \rho\Delta_k$, $\hat{s}_k^d = \text{softmax}(z_k^d)_{\text{fg}}$, where ρ is the offset scaling factor, and $\text{softmax}(z_k^d)_{\text{fg}}$ denotes the foreground class probability.

PDC is fundamentally different from traditional density-estimation methods. PDC directly outputs discrete point candidates and is supervised through one-to-one matching with ground-truth points, as described below.

3.5 Branch-specific Matching

Both RSC and PDC output unordered sets of candidates. To provide instance-level supervision, a one-to-one matching between predicted candidates and ground-truth instances is required. Since the two branches differ in candidate number, spatial structure, and prediction form, we use branch-specific matching. **For RSC**, the prediction point is the region center \hat{p}_i^r , and the target is the ground-truth point p_j . We construct the matching cost based on point distance and foreground confidence:

$$C_{ij}^r = \lambda_p^r d(\hat{p}_i^r, p_j) - \lambda_s^r \hat{s}_i^r, \quad (5)$$

where $d(\cdot, \cdot)$ denotes the point-distance term, and λ_p^r and λ_s^r balance localization and confidence. The RSC assignment is obtained by Hungarian matching, denoted as $\mathcal{M}_r = \text{Hungarian}(C^r)$. **For PDC**, the number of dense candidates is usually much larger than the number of ground-truth instances. Instead of constructing a full matching matrix, we select a local candidate set $\mathcal{N}(j)$ around each ground-truth point p_j and compute the cost only on these candidate edges:

$$C_{kj}^d = \lambda_p^d \|\hat{p}_k^d - p_j^{\text{img}}\|_2 - \lambda_s^d \hat{s}_k^d, \quad k \in \mathcal{N}(j), \quad (6)$$

where p_j^{img} is the ground-truth point in the input-image coordinate system. The PDC assignment is denoted by $\mathcal{M}_d = \text{SparseHungarian}(C^d)$. This design preserves one-to-one instance supervision while avoiding unnecessary competition among spatially irrelevant dense candidates, and significantly reduces the computational complexity of matching across large sets.

3.6 Point-centric Supervision for Heterogeneous Annotations

After branch-specific matching, Count Anything uses point-centric supervision to train on heterogeneous annotations. The core principle is that all valid instances are supervised through counting points, while only ground-truth bounding boxes are used as strong geometric regression targets. This allows the model to learn from both box-annotated and point-only data without using pseudo boxes as regression targets.

RSC Supervision. For RSC, all matched instances participate in point localization supervision:

$$\mathcal{L}_{\text{point}}^r = \frac{1}{Z_r} \sum_{(i,j) \in \mathcal{M}_r} \|\hat{p}_i^r - p_j\|_2^2, \quad (7)$$

where Z_r is the number of ground-truth instances in the batch. Box regression and GIoU losses are computed only for instances with reliable ground-truth bounding boxes:

$$\mathcal{L}_{\text{box}}^r = \frac{1}{Z_b} \sum_{(i,j) \in \mathcal{M}_r} m_j \|\hat{B}_i^r - B_j\|_1, \quad \mathcal{L}_{\text{giou}}^r = \frac{1}{Z_b} \sum_{(i,j) \in \mathcal{M}_r} m_j (1 - \text{GIoU}(\hat{B}_i^r, B_j)), \quad (8)$$

where Z_b normalizes valid box-annotated instances. Here, $m_j = 1$ indicates that the j -th instance has a ground-truth box, while $m_j = 0$ indicates that it does not. When $m_j = 0$, the instance does not contribute to box regression.

RSC classification follows foreground-background discrimination. Unmatched queries are treated as background. For a matched query, we use a soft foreground target based on spatial coverage quality.

The effective region is: $B_{ij}^* = \begin{cases} B_j, & m_j = 1, \\ \tilde{B}_{ij}, & m_j = 0, \end{cases}$ where \tilde{B}_{ij} is an auxiliary region centered at p_j with

the size of the matched predicted box \hat{B}_i^r . This auxiliary region is used only for classification quality estimation, not as a box regression target. The coverage quality and soft target are:

$$g_{ij} = \frac{|\hat{B}_i^r \cap B_{ij}^*|}{|B_{ij}^*|}, \quad \tilde{y}_i = \max(\epsilon, \text{sg}(\hat{s}_i^r)^\alpha (g_{ij})^{1-\alpha}), \quad (9)$$

where $\text{sg}(\cdot)$ denotes stop-gradient, ϵ is the minimum positive label value, and α balances confidence and coverage. The RSC classification loss is: $\mathcal{L}_{cls}^r = \mathcal{L}_{\text{BCE}}(\{z_i^r\}, \{\tilde{y}_i\})$. This supervision decouples instance location, candidate confidence, and object extent: points supervise where instances are, classification supervises whether candidates correspond to target instances, and boxes supervise geometry only when reliable boxes exist.

PDC Supervision. PDC does not predict bounding boxes, and its supervision is therefore entirely based on point-set matching. Given \mathcal{M}_d , matched dense candidates are foreground and the remaining candidates are background. The classification loss is: $\mathcal{L}_{cls}^d = \mathcal{L}_{\text{CE}}(\{z_k^d\}, \{y_k^d\})$, where $y_k^d \in \{\text{bg}, \text{fg}\}$. The point regression loss is:

$$\mathcal{L}_{point}^d = \frac{1}{Z_d} \sum_{(k,j) \in \mathcal{M}_d} \|\hat{p}_k^d - p_j^{\text{img}}\|_2^2, \quad (10)$$

where Z_d is the normalization factor for matched foreground samples.

Overall Objective. The overall objective is:

$$\mathcal{L} = \lambda_{cls}^r \mathcal{L}_{cls}^r + \lambda_{point}^r \mathcal{L}_{point}^r + \lambda_{box}^r \mathcal{L}_{box}^r + \lambda_{giou}^r \mathcal{L}_{giou}^r + \lambda_{cls}^d \mathcal{L}_{cls}^d + \lambda_{point}^d \mathcal{L}_{point}^d, \quad (11)$$

where each λ is a loss weight.

3.7 Complementary Count Fusion

During inference, RSC and PDC produce region-level candidates and dense point candidates, respectively. Direct concatenation may double-count clearly bounded targets predicted by both branches, while removing all PDC points inside each RSC region may discard multiple true instances in crowded scenes. Complementary Count Fusion resolves this conflict with a lightweight, parameter-free rule. We first filter RSC predictions by threshold τ_r and remove duplicate regions using intersection over minimum area (IoM), $\text{IoM}(A, B) = |A \cap B| / \min(|A|, |B|)$. If one region contains another, or if their IoM exceeds a predefined threshold, only the region with the higher confidence score is retained. Meanwhile, PDC candidates are also filtered by threshold τ_d . Then, Complementary Count Fusion (CCF) resolves the overlap between retained RSC regions and filtered PDC points by a simple confidence comparison, as shown in Fig. 2. For each retained RSC region, if no PDC point falls inside it, CCF keeps its center as a region-level counting point. Otherwise, CCF finds the PDC point closest to the RSC center and keeps the one with the higher confidence between this PDC point and the RSC region. The lower-confidence prediction is suppressed, while the other PDC points inside the same region are preserved. Let $\mathcal{P}_{\text{keep}}^r$ be the retained RSC points after this comparison, and let $\mathcal{P}_{\text{keep}}^d$ be the PDC points kept after suppression. The final prediction is: $\hat{\mathcal{P}}_T = \mathcal{P}_{\text{keep}}^r \cup \mathcal{P}_{\text{keep}}^d$, $\hat{c}_T = |\hat{\mathcal{P}}_T|$. By comparing each RSC region only with its nearest PDC point, CCF reduces duplicate counts on clearly bounded targets while preserving dense PDC predictions in crowded regions.

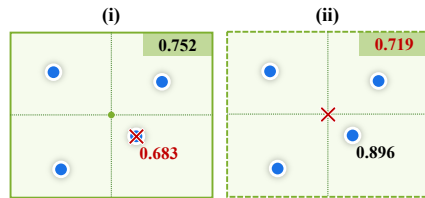


Figure 2: Illustration of CCF. Each RSC prediction is compared with the nearest PDC point inside its region, and only the higher-confidence one is kept.

4 Experimental Results

4.1 Experimental Setup

Count Anything adopts the pretrained SAM3 [35] Text-Conditioned Encoder with its original weights frozen, and introduces LoRA [46] adapters into the cross-modal encoder for counting-task adaptation. All experiments are conducted on a single node with 8 NVIDIA A100 GPUs. We train the model for 30 epochs with an input resolution of 1008×1008 and a per-GPU batch size of 24. The learning rates of RSC, PDC, and LoRA are set to 1×10^{-5} , 1×10^{-4} , and 1×10^{-3} , respectively, and AdamW [47] is used as the optimizer. RSC uses 200 region queries. During inference, the score thresholds are $\tau_r = 0.5$ and $\tau_d = 0.5$. The IoM threshold for RSC duplicate removal is set to 0.5. We report results

Table 1: Comparison with other state-of-the-art methods on the CLOC dataset.

Method	Venue	Ins.Gr.d.	Full Test			Single Domain MAE					
			MAE	RMSE	NAE	GS	RS	Histo.	Cell.	Agri.	Micro.
LOCA [37]	ICCV'23	✗	68.86	179.26	14.54	55.24	81.37	65.86	124.36	473.81	39.47
CLIP-Count [29]	ACMMM'23	✗	31.28	<u>118.55</u>	4.49	24.07	25.81	57.98	121.17	298.45	22.40
CounTX [38]	BMVC'23	✗	32.24	122.10	4.75	23.84	30.06	64.64	<u>112.22</u>	301.10	27.41
VLCounter [4]	AAAI'24	✗	30.52	121.88	3.07	23.59	22.45	<u>55.14</u>	125.83	306.38	37.72
VA-Count [5]	ECCV'24	✗	33.85	129.81	4.62	22.39	36.62	72.50	163.73	315.23	20.47
CountGD [6]	NeurIPS'24	✓	55.91	1081.82	5.17	53.83	42.23	99.69	114.50	<u>244.80</u>	<u>11.09</u>
YOLO-Count [39]	ICCV'25	✗	34.15	133.56	1.12	25.62	27.69	72.50	168.37	309.30	24.33
T2ICount [40]	CVPR'25	✗	50.02	155.92	8.71	34.08	62.46	73.06	151.30	296.37	203.51
CountSE [41]	ICCV'25	✗	28.81	130.49	2.28	22.14	19.07	70.27	120.08	302.19	22.22
CountGD++ [7]	CVPR'26	✓	<u>23.38</u>	123.28	1.02	<u>14.50</u>	<u>10.69</u>	55.83	222.25	321.99	43.80
GLIP-L [42]	CVPR'22	✓	37.38	140.07	0.82	31.06	19.57	68.78	225.42	320.85	43.71
YOLO-World-X [43]	CVPR'24	✓	34.94	138.35	<u>0.78</u>	28.17	17.81	67.93	226.07	321.13	42.49
GrdDINO-SwinB [44]	ECCV'24	✓	35.06	139.46	0.79	29.50	14.13	69.11	224.96	320.17	43.75
YOLOE-v8L [45]	ICCV'25	✓	40.03	140.62	0.99	33.97	21.78	69.99	228.28	321.97	44.80
YOLOE-11L [45]	ICCV'25	✓	40.05	140.60	1.00	34.00	21.78	69.95	228.24	321.88	44.81
SAM3 [35]	ICLR'26	✓	25.80	130.26	0.89	19.39	10.88	64.32	152.45	315.88	37.04
Count Anything	–	✓	9.34	33.34	0.75	8.76	6.21	16.64	38.65	41.78	4.30

Note: Ins.Gr.d. denotes instance grounding capability. GS, RS, Histo., Cell., Agri., and Micro. denote General Scene, Remote Sensing, Histopathology, Cellular Microscopy, Agriculture, and Microbiology, respectively.

on the CLOC test split. Following prior object-counting studies, we use MAE, RMSE, and NAE as evaluation metrics. More implementation and evaluation details are provided in Appendix B.

4.2 Comparison with Other Methods

Main results on CLOC. We compare Count Anything with representative open-world object counting methods, as well as detection and segmentation foundation models. For all competing methods, we use their officially released checkpoints and standard inference pipelines to evaluate their out-of-the-box capability. Under this setting, we aim to answer a practical question: *How far can current counting models go when facing larger-scale data, more diverse visual scenarios, and broader count distributions?*

As shown in Table 1, Count Anything achieves 9.34 MAE, 33.34 RMSE, and 0.75 NAE on CLOC, substantially outperforming all compared methods across the six visual domains. The results show that existing open-world object counting methods are effective to some extent in the general scene, but their errors increase noticeably in specialized domains such as histopathology, cellular microscopy, agriculture, and microbiology. The evaluated detection and segmentation foundation models can produce instance-level predictions, but they are still prone to missed detections or merged instances in scenes with dense, small, or weakly bounded targets, making it difficult to provide reliable counting results. In contrast, Count Anything performs more stably in the cross-domain counting task, demonstrating better generalization ability. Figure 3 shows visualized predictions of different models on several samples. Compared with these methods, Count Anything’s predictions are noticeably closer to the ground truth across different scenarios. In particular, Count Anything maintains stable counting results on samples with significant differences in target scale and density distribution.

We attribute the substantial advantage of Count Anything to two key factors: first, the dual-granularity counting design that combines region-level sparse counting and pixel-level dense counting in the model architecture; second, the larger-scale and broader coverage of categories, visual domains, and target-density distributions provided by CLOC. We further disentangle and validate the contributions of these two factors in the following ablation studies.

Dense crowd counting performance. Crowd counting is a widely studied branch of object counting, and it is also one of the most challenging subtasks due to extremely dense target distributions and complex occlusion relationships. Therefore, we conduct an additional evaluation on the dense crowd counting subset from ShanghaiTech Part A in the CLOC test split. Table 2 compares Count Anything with existing open-world counting models and typical crowd-counting-specific methods. As can be seen, compared with open-world baselines, Count Anything substantially reduces MAE and RMSE. Compared with methods specifically designed for crowd counting, Count Anything also achieves

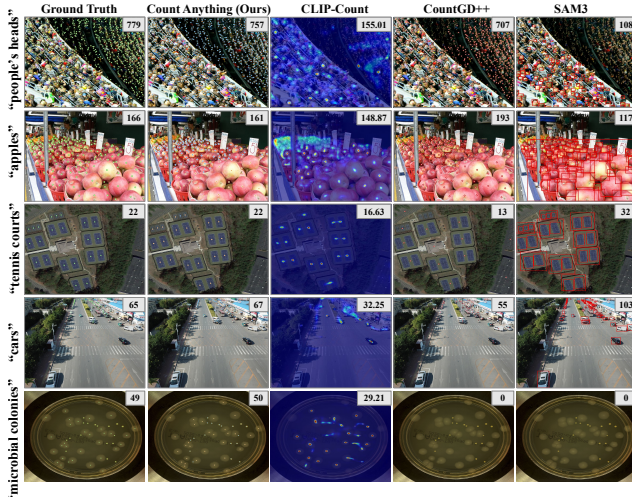


Figure 3: Qualitative comparison of counting predictions. Text prompts are shown on the left, and numbers denote ground-truth or predicted counts.

Table 4: CLOC-trained method comparison.

Method	Full Test MAE		
	Overall	Seen	Unseen
CLIP-Count [29]	14.48	19.16	7.24
CountGD [6]	13.19	18.61	4.81
VLCounter [4]	21.61	28.68	10.68
Count Anything	9.34	12.37	4.79

Table 5: Ablation on dual counters and fusion rules.

	Counter		Fusion		Metric	
	RSC	PDC	DU	CCF	MAE	RMSE
✓	✗	✗	✗	✗	18.66	116.42
✗	✓	✗	✗	✗	13.30	35.31
✓	✓	✓	✗	✗	22.85	50.05
✓	✓	✗	✓	✓	9.34	33.34

Table 2: Evaluation on crowd counting subtask.

Method	Venue	MAE	RMSE
GrdDINO-SwinB [44]	ECCV'24	450.86	588.65
YOLO-World-X [43]	CVPR'24	421.15	571.57
SAM3 [35]	ICLR'26	345.55	527.10
APGCC [15]	ECCV'24	49.30	75.03
P2PNet [10]	ICCV'21	48.10	80.59
HMoDE [48]	TIP'23	57.36	80.89
HMoDE+REL [48]	TIP'23	50.91	<u>76.64</u>
Count Anything	–	55.90	92.15

Table 3: FSC-147-to-CLOC transfer.

Method	Venue	MAE	RMSE
LOCA [37]	ICCV'23	68.86	179.24
CLIP-Count [29]	ACMMM'23	31.28	118.55
CountTX [38]	BMVC'23	32.24	<u>122.10</u>
VLCounter [4]	AAAI'24	30.52	121.88
VA-Count [5]	ECCV'24	33.85	129.81
CountGD [6]	NeurIPS'24	55.91	1081.82
YOLO-Count [39]	ICCV'25	34.15	133.56
T2ICount [40]	CVPR'25	50.02	155.92
CountSE [41]	ICCV'25	28.81	130.49
CountGD++ [7]	CVPR'26	<u>23.38</u>	123.28
Count Anything	–	19.01	100.12

Table 6: Ablation on point-centric supervision strategies.

GT	Box/GIoU sup.		RSC cls. sup.		Metric	
	Pseudo	Hard	Soft	MAE	RMSE	
✓	✓	✗	✓	11.44	38.20	
✓	✗	✓	✗	17.36	92.70	
✓	✗	✗	✓	9.34	33.34	

performance within the same tier, although it slightly lags behind the most advanced crowd-specific methods. Considering that these crowd-specific methods are usually specially optimized for a single crowd dataset and high-density distributions, while Count Anything is designed for a unified text-guided counting task across categories and visual domains, this result shows that Count Anything can achieve strong competitiveness on the crowd counting subset while maintaining generality.

Comparison under aligned training data. To better isolate the effect of training-data differences, we further compare methods under aligned training-data settings from two directions. First, in Table 3, we place the proposed Count Anything framework under the FSC-147 training setting commonly used by existing open-world counting methods, and evaluate all methods on the CLOC test split; the other methods directly load the FSC-147-trained weights officially released by their corresponding papers. Under this setting, Count Anything achieves 19.01 MAE and 100.12 RMSE, outperforming representative counting methods that are also trained on FSC-147. Second, in Table 4, we place representative methods under the same CLOC training setting, train them on the full CLOC training split using their official configurations, and evaluate them under the same full-test protocol. The seen/unseen split of CLOC is based on semantic category groups, where unseen categories are excluded from training to evaluate category generalization; details are provided in Appendix C. Under this setting, Count Anything obtains the lowest overall, seen, and unseen MAE. Together, these two comparisons show that the performance advantage of Count Anything is not merely due to training-data differences, but also reflects the effectiveness of the proposed model design for cross-domain text-guided counting.

4.3 Ablation Study

Dual counters and fusion rules. Table 5 compares the effects of the two counting branches and the fusion rule. In the table, DU refers to direct union. We can observe that using only RSC or PDC is inferior to the full model, showing that the two counters are complementary under different object scales and density distributions. Directly union of the two branch outputs leads to duplicate predictions for the same target and thus larger counting errors. In contrast, Count Anything uses CCF to fuse the two branches and achieves the best performance, validating the effectiveness of the dual-counter design and complementary fusion rule.

Point-centric supervision strategies. Table 6 ablates two design choices in point-centric supervision: the box targets used for RSC Box/GIoU supervision and the foreground targets used for RSC classification. Following Sec. 3, Count Anything uses GT-only box targets for geometric regression and a soft classification target for matched RSC queries. Adding pseudo box targets to point-only samples increases error, indicating that pseudo boxes may introduce noise that degrades region-level geometry learning. Replacing the soft classification target with a hard foreground target also leads to worse performance, showing that quality-aware foreground supervision helps RSC distinguish high-quality region candidates from lower-quality ones. These results support the point-centric supervision design: all instances contribute through point supervision, reliable GT boxes supervise geometry, and soft foreground targets guide RSC confidence learning.

Data scaling. Figure 4 visualizes the effect of CLOC training data scale by plotting MAE and RMSE under different ratios. As the data ratio increases from 10% to 100%, both MAE and RMSE consistently decrease, showing that Count Anything benefits from larger-scale cross-domain counting data. This result supports the value of data scaling-up: the full CLOC training set provides broader category, visual-domain, and density coverage, leading to a more stable training basis for generalist text-guided counting.

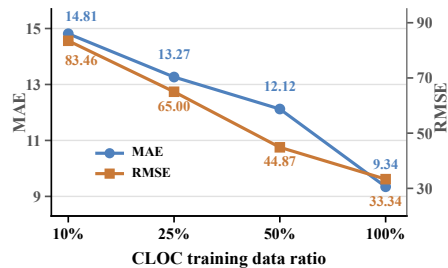


Figure 4: Effect of training data scale.

5 Conclusion

This paper studies cross-domain text-guided object counting and advances generalist counting models from both the data and method perspectives. We construct CLOC, a large-scale object counting dataset covering six visual domains, which unifies multi-source heterogeneous annotations into a category-specified counting task and provides a unified benchmark for training and evaluating counting models across categories, domains, and density distributions. Based on CLOC, we propose Count Anything, which adopts discrete instance points as the final output representation and combines a Region-level Sparse Counter, a Pixel-level Dense Counter, point-centric supervision, and Complementary Count Fusion to complement region-level sparse enumeration with pixel-level dense enumeration. Experimental results show that Count Anything achieves stable cross-domain counting performance on CLOC and outperforms representative methods across diverse visual domains and density conditions. We hope this work can provide a useful data foundation and method reference for future research on open-world, text-guided, and instance-interpretable generalist object counting.

References

- [1] Muhammad Awais, Muzammal Naseer, Salman Khan, Rao Muhammad Anwer, Hisham Cholakkal, Mubarak Shah, Ming-Hsuan Yang, and Fahad Shahbaz Khan. Foundation models defining a new era in vision: A survey and outlook. *IEEE Trans. Pattern Anal. Mach. Intell.*, 47(4):2245–2264, 2025.
- [2] Siqi Lu, Junlin Guo, James R Zimmer-Dauphinee, Jordan M Nieusma, Xiao Wang, Parker VanValkenburgh, Steven A Wernke, and Yuankai Huo. Vision foundation models in remote sensing: A survey. *IEEE Geosci. Remote Sens. Mag.*, 13(3):190–215, 2025.
- [3] Lu Yuan, Dongdong Chen, Yi-Ling Chen, Noel Codella, Xiyang Dai, Jianfeng Gao, Houdong Hu, Xuedong Huang, Boxin Li, Chunyuan Li, et al. Florence: A new foundation model for computer vision. *arXiv preprint arXiv:2111.11432*, 2021.
- [4] Seunggu Kang, WonJun Moon, Euiyeon Kim, and Jae-Pil Heo. VLCounter: Text-aware visual representation for zero-shot object counting. In *AAAI*, volume 38, pages 2714–2722, 2024.
- [5] Huilin Zhu, Jingling Yuan, Zhengwei Yang, Yu Guo, Zheng Wang, Xian Zhong, and Shengfeng He. Zero-Shot Object Counting With Good Exemplars. In *Eur. Conf. Comput. Vis.*, pages 368–385, 2024.
- [6] Niki Amini-Naieni, Tengda Han, and Andrew Zisserman. CountGD: Multi-modal open-world counting. In *Adv. Neural Inform. Process. Syst.*, volume 37, pages 48810–48837, 2024.
- [7] Niki Amini-Naieni and Andrew Zisserman. CountGD++: Generalized prompting for open-world counting. In *IEEE Conf. Comput. Vis. Pattern Recog.*, 2026.

- [8] Lijia Deng, Qinghua Zhou, Shuihua Wang, Juan Manuel Górriz, and Yudong Zhang. Deep learning in crowd counting: A survey. *CAAI Trans. on Intell. Tech.*, 9(5):1043–1077, 2024.
- [9] Jingyi Xu, Hieu Le, Vu Nguyen, Viresh Ranjan, and Dimitris Samaras. Zero-shot object counting. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 15548–15557, 2023.
- [10] Qingyu Song, Changan Wang, Zhengkai Jiang, Yabiao Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Yang Wu. Rethinking counting and localization in crowds: A purely point-based framework. In *Int. Conf. Comput. Vis.*, pages 3365–3374, 2021.
- [11] Michael Hobley and Victor Prisacariu. Learning To Count Anything: Reference-less class-agnostic counting with weak supervision. In *IEEE Conf. Comput. Vis. Pattern Recog. Worksh.*, 2023.
- [12] Viresh Ranjan, Udbhav Sharma, Thu Nguyen, and Minh Hoai. Learning To Count Everything. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3394–3403, 2021.
- [13] Thanh Nguyen, Chau Pham, Khoi Nguyen, and Minh Hoai. Few-shot Object Counting and Detection. In *Eur. Conf. Comput. Vis.*, pages 348–365, 2022.
- [14] Yingying Zhang, Desen Zhou, Siqin Chen, Shenghua Gao, and Yi Ma. Single-image crowd counting via multi-column convolutional neural network. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 589–597, 2016.
- [15] I-Hsiang Chen, Wei-Ting Chen, Yu-Wei Liu, Ming-Hsuan Yang, and Sy-Yen Kuo. Improving Point-Based Crowd Counting and Localization Based on Auxiliary Point Guidance. In *Eur. Conf. Comput. Vis.*, pages 428–444, 2024.
- [16] Ye Tian, Xiangxiang Chu, and Hongpeng Wang. CCTrans: Simplifying and improving crowd counting with transformer. *arXiv preprint arXiv:2109.14483*, 2021.
- [17] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *Eur. Conf. Comput. Vis.*, pages 740–755, 2014.
- [18] Mark Everingham, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The PASCAL Visual Object Classes (VOC) Challenge. *Int. J. Comput. Vis.*, 88(2):303–338, 2010.
- [19] Vishwanath A. Sindagi, Rajeev Yasarla, and Vishal M. Patel. JHU-CROWD++: Large-scale crowd counting dataset and a benchmark method. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(5):2594–2609, 2022.
- [20] Amirreza Mahbod, Christine Polak, Katharina Feldmann, Rumsha Khan, Katharina Gelles, Georg Dorffner, Ramona Woitek, Sepideh Hatamikia, and Isabella Ellinger. NuInsSeg: A fully annotated dataset for nuclei instance segmentation in H&E-stained histological images. *Sci. Data*, 11(1):295, 2024.
- [21] Neeraj Kumar, Ruchika Verma, Sanuj Sharma, Surabhi Bhargava, Abhishek Vahadane, and Amit Sethi. A Dataset and a Technique for Generalized Nuclear Segmentation for Computational Pathology. *IEEE Trans. Med. Imag.*, 36(7):1550–1560, 2017.
- [22] Can Shi, Jinghong Fan, Zhonghan Deng, Huanlin Liu, Qiang Kang, Yumei Li, Jing Guo, Jingwen Wang, Jinjiang Gong, Sha Liao, Ao Chen, Ying Zhang, and Mei Li. CellBinDB: A large-scale multimodal annotated dataset for cell segmentation with benchmarking of universal models. *GigaScience*, 14:giaf069, 2025.
- [23] Shuai Shao, Zeming Li, Tianyuan Zhang, Chao Peng, Gang Yu, Xiangyu Zhang, Jing Li, and Jian Sun. Objects365: A large-scale, high-quality dataset for object detection. In *Int. Conf. Comput. Vis.*, pages 8430–8439, 2019.
- [24] Pengfei Zhu, Longyin Wen, Dawei Du, Xiao Bian, Heng Fan, Qinghua Hu, and Haibin Ling. Detection and Tracking Meet Drones Challenge. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(11):7380–7399, 2022.
- [25] Ruchika Verma, Neeraj Kumar, Abhijeet Patil, Nikhil Cherian Kurian, Swapnil Rane, Simon Graham, Quoc Dang Vu, Mieke Zwager, Shan E. Ahmed Raza, Nasir M. Rajpoot, et al. MoNuSAC2020: A multi-organ nuclei segmentation and classification challenge. *IEEE Trans. Med. Imag.*, 40(12):3413–3423, 2021.
- [26] Deepak Babu Sam, Skand Vishwanath Peri, Mukuntha Narayanan Sundararaman, Amogh Kamath, and R. Venkatesh Babu. Locate, size, and count: Accurately resolving people in dense crowds via detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 43(8):2739–2751, 2021.

- [27] Meng-Ru Hsieh, Yen-Liang Lin, and Winston H. Hsu. Drone-Based Object Counting by Spatially Regularized Regional Proposal Networks. In *Int. Conf. Comput. Vis.*, pages 4145–4153, 2017.
- [28] Boyu Wang, Huidong Liu, Dimitris Samaras, and Minh Hoai Nguyen. Distribution Matching for Crowd Counting. In *Adv. Neural Inform. Process. Syst.*, pages 1595–1607, 2020.
- [29] Ruixiang Jiang, Lingbo Liu, and Changwen Chen. CLIP-Count: Towards text-guided zero-shot object counting. In *ACM Int. Conf. Multimedia*, pages 4535–4545, 2023.
- [30] Xinyan Liu, Guorong Li, Yuankai Qi, Zhenjun Han, Anton van den Hengel, Nicu Sebe, Ming-Hsuan Yang, and Qingming Huang. Consistency-aware anchor pyramid network for crowd localization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 2024.
- [31] Zhongyi Huang, Yao Ding, Guoli Song, Lin Wang, Ruizhe Geng, Hongliang He, Shan Du, Xia Liu, Yonghong Tian, Yongsheng Liang, S. Kevin Zhou, and Jie Chen. BCData: A large-scale dataset and benchmark for cell detection and counting. In *Med. Image Comput. Comput. Assist. Interv.*, pages 289–298. Springer, 2020.
- [32] Hao Lu, Zhiguo Cao, Yang Xiao, Bohan Zhuang, and Chunhua Shen. TasselNet: Counting maize tassels in the wild via local counts regression network. *Plant Methods*, 13(1):79, 2017.
- [33] Agrim Gupta, Piotr Dollár, and Ross Girshick. LVIS: A dataset for large vocabulary instance segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 5356–5364, 2019.
- [34] Bryan A. Plummer, Liwei Wang, Chris M. Cervantes, Juan C. Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k Entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *Int. Conf. Comput. Vis.*, pages 2641–2649, 2015.
- [35] Nicolas Carion, Laura Gustafson, Yuan-Ting Hu, Shoubhik Debnath, Ronghang Hu, Didac Suris, Chaitanya Ryali, Kalyan Vasudev Alwala, Haitham Khedr, Andrew Huang, et al. SAM 3: Segment anything with concepts. In *Int. Conf. Learn. Represent.*, 2026.
- [36] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-End Object Detection with Transformers. In *Eur. Conf. Comput. Vis.*, pages 213–229, 2020.
- [37] Nikola Đukić, Alan Lukežič, Vitjan Zavrtnik, and Matej Kristan. A Low-Shot Object Counting Network With Iterative Prototype Adaptation. In *Int. Conf. Comput. Vis.*, pages 18872–18881, 2023.
- [38] Niki Amini-Naieni, Kiana Amini-Naieni, Tengda Han, and Andrew Zisserman. Open-World Text-Specified Object Counting. In *Brit. Mach. Vis. Conf.*, 2023.
- [39] Guanning Zeng, Xiang Zhang, Zirui Wang, Haiyang Xu, Zeyuan Chen, Bingnan Li, and Zhuowen Tu. YOLO-Count: Differentiable object counting for text-to-image generation. In *Int. Conf. Comput. Vis.*, pages 16765–16775, 2025.
- [40] Yifei Qian, Zhongliang Guo, Bowen Deng, Chun Tong Lei, Shuai Zhao, Chun Pong Lau, Xiaopeng Hong, and Michael P. Pound. T2ICount: Enhancing cross-modal understanding for zero-shot counting. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 25336–25345, 2025.
- [41] Shuai Liu, Peng Zhang, Shiwei Zhang, and Wei Ke. CountSE: Soft exemplar open-set object counting. In *Int. Conf. Comput. Vis.*, pages 21536–21546, 2025.
- [42] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. Grounded Language-Image Pre-Training. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 10965–10975, 2022.
- [43] Tianheng Cheng, Lin Song, Yixiao Ge, Wenyu Liu, Xinggang Wang, and Ying Shan. YOLO-World: Real-time open-vocabulary object detection. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 16901–16911, 2024.
- [44] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. Grounding DINO: Marrying DINO with grounded pre-training for open-set object detection. In *Eur. Conf. Comput. Vis.*, pages 38–55, 2024.
- [45] Ao Wang, Lihao Liu, Hui Chen, Zijia Lin, Jungong Han, and Guiguang Ding. YOLOE: Real-time seeing anything. In *Int. Conf. Comput. Vis.*, pages 24591–24602, 2025.

- [46] Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *Int. Conf. Learn. Represent.*, 2022.
- [47] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *Int. Conf. Learn. Represent.*, 2019.
- [48] Zhipeng Du, Miaoqing Shi, Jiankang Deng, and Stefanos Zafeiriou. Redesigning Multi-Scale Neural Network for Crowd Counting. *IEEE Trans. Image Process.*, 32:3664–3678, 2023.
- [49] Yuhong Li, Xiaofan Zhang, and Deming Chen. CSRNet: Dilated convolutional neural networks for understanding the highly congested scenes. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 1091–1100, 2018.
- [50] Xinkun Cao, Zhipeng Wang, Yanyun Zhao, and Fei Su. Scale Aggregation Network for Accurate and Efficient Crowd Counting. In *Eur. Conf. Comput. Vis.*, pages 734–750, 2018.
- [51] Zhiheng Ma, Xing Wei, Xiaopeng Hong, and Yihong Gong. Bayesian loss for crowd count estimation with point supervision. In *Int. Conf. Comput. Vis.*, pages 6142–6151, 2019.
- [52] Chang Liu, Yujie Zhong, Andrew Zisserman, and Weidi Xie. CounTR: Transformer-based generalised visual counting. In *Brit. Mach. Vis. Conf.*, 2022.
- [53] Zhizhong Huang, Mingliang Dai, Yi Zhang, Junping Zhang, and Hongming Shan. Point, Segment and Count: A generalized framework for object counting. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 17067–17076, 2024.
- [54] Jer Pelhan, Alan Lukežič, Vitjan Zavrtanik, and Matej Kristan. DAVE: A detect-and-verify paradigm for low-shot counting. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 23293–23302, 2024.
- [55] Niki Amini-Naieni and Andrew Zisserman. Open-World Object Counting in Videos. In *AAAI*, volume 40, pages 2300–2308, 2026.
- [56] Qi Wang, Junyu Gao, Wei Lin, and Xuelong Li. NWPU-Crowd: A large-scale benchmark for crowd counting and localization. *IEEE Trans. Pattern Anal. Mach. Intell.*, 43(6):2141–2149, 2021.
- [57] Etienne David, Simon Madec, Pouria Sadeghi-Tehran, Helge Aasen, Bangyou Zheng, Shouyang Liu, Norbert Kirchgessner, Goro Ishikawa, Koichi Nagasawa, Minhajul A. Badhon, Curtis Pozniak, Benoit de Solan, Andreas Hund, Scott C. Chapman, Frédéric Baret, Ian Stavness, and Wei Guo. Global Wheat Head Detection (GWHD) Dataset: A large and diverse dataset of high-resolution RGB-labelled images to develop and benchmark wheat head detection methods. *Plant Phenomics*, 2020:3521852, 2020.
- [58] Sylwia Majchrowska, Jarosław Pawłowski, Grzegorz Guła, Tomasz Bonus, Agata Hanas, Adam Loch, Agnieszka Pawlak, Justyna Roszkowiak, Tomasz Golan, and Zuzanna Drulis-Kawa. AGAR: A microbial colony dataset for deep learning detection, 2021.
- [59] Anindya Mondal, Sauradip Nag, Xiatian Zhu, and Anjan Dutta. OmniCount: Multi-label object counting with semantic-geometric priors. In *AAAI*, volume 39, pages 19537–19545, 2025.
- [60] Gui-Song Xia, Xiang Bai, Jian Ding, Zhen Zhu, Serge Belongie, Jiebo Luo, Mihai Datcu, Marcello Pelillo, and Liangpei Zhang. DOTA: A large-scale dataset for object detection in aerial images. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3974–3983, 2018.
- [61] Darius Lam, Richard Kuzma, Kevin McGee, Samuel Dooley, Michael Laielli, Matthew Klaric, Yaroslav Bulatov, and Brendan McCord. xView: Objects in context in overhead imagery. *arXiv preprint arXiv:1802.07856*, 2018.
- [62] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The Cityscapes Dataset for Semantic Urban Scene Understanding. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 3213–3223, 2016.
- [63] Christoffer Edlund, Timothy R. Jackson, Nabeel Khalid, Nicola Bevan, Timothy Dale, Andreas Dengel, Sheraz Ahmed, Johan Trygg, and Rickard Sjögren. LIVECell: A large-scale dataset for label-free live cell segmentation. *Nat. Methods*, 18(9):1038–1045, 2021.
- [64] Jiajia Li, Raju Thada Magar, Dong Chen, Feng Lin, Dechun Wang, Xiang Yin, Weichao Zhuang, and Zhaojian Li. SoybeanNet: Transformer-based convolutional neural network for soybean pod counting from unmanned aerial vehicle (UAV) images. *Comput. Electron. Agric.*, 220:108861, 2024.

- [65] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning Transferable Visual Models From Natural Language Supervision. In *Int. Conf. Mach. Learn.*, volume 139, pages 8748–8763. PMLR, 2021.
- [66] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment Anything. In *Int. Conf. Comput. Vis.*, pages 4015–4026, 2023.
- [67] Jiancheng Yang, Rui Shi, Donglai Wei, Zequan Liu, Lin Zhao, Bilian Ke, Hanspeter Pfister, and Bingbing Ni. MedMNIST v2: A large-scale lightweight benchmark for 2D and 3D biomedical image classification. *Sci. Data*, 10(1):41, 2023.
- [68] Michela Antonelli, Annika Reinke, Spyridon Bakas, Keyvan Farahani, Annette Kopp-Schneider, Bennett A. Landman, Geert Litjens, Bjoern Menze, Olaf Ronneberger, Ronald M. Summers, Bram van Ginneken, et al. The Medical Segmentation Decathlon. *Nat. Commun.*, 13(1):4128, 2022.
- [69] John Lambert, Zhuang Liu, Ozan Sener, James Hays, and Vladlen Koltun. MSeg: A composite dataset for multi-domain semantic segmentation. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 2879–2888, 2020.
- [70] Likun Cai, Zhi Zhang, Yi Zhu, Li Zhang, Mu Li, and Xiangyang Xue. BigDetection: A large-scale benchmark for improved object detector pre-training. In *IEEE Conf. Comput. Vis. Pattern Recog. Worksh.*, pages 4777–4787, 2022.
- [71] fyp2. rebar counting dataset. <https://universe.roboflow.com/fyp2-czq30/rebar-counting-12vha>, September 2024. visited on 2026-05-02.
- [72] Paulo R. L. de Almeida, Luiz S. Oliveira, Alceu S. Britto Jr., Eunelson J. Silva Jr., and Alessandro L. Koerich. PKLot: A robust dataset for parking lot classification. *Expert Syst. Appl.*, 42(11):4937–4949, 2015.
- [73] Ke Li, Gang Wan, Gong Cheng, Liqiu Meng, and Junwei Han. Object Detection in Optical Remote Sensing Images: A survey and a new benchmark. *ISPRS J. Photogramm. Remote Sens.*, 159:296–307, 2020.
- [74] Jian Ding, Nan Xue, Gui-Song Xia, Xiang Bai, Wen Yang, Michael Ying Yang, Serge Belongie, Jiebo Luo, Mihai Datcu, Marcello Pelillo, and Liangpei Zhang. Object Detection in Aerial Images: A large-scale benchmark and challenges. *IEEE Trans. Pattern Anal. Mach. Intell.*, 44(11):7778–7796, 2022.
- [75] Jian Ding, Nan Xue, Yang Long, Gui-Song Xia, and Qikai Lu. Learning RoI Transformer for Oriented Object Detection in Aerial Images. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 2849–2858, 2019.
- [76] Longyin Wen, Dawei Du, Pengfei Zhu, Qinghua Hu, Qilong Wang, Liefeng Bo, and Siwei Lyu. Detection, Tracking, and Counting Meets Drones in Crowds: A benchmark. In *IEEE Conf. Comput. Vis. Pattern Recog.*, pages 7812–7821, 2021.
- [77] Junyu Gao, Liangliang Zhao, and Xuelong Li. NWPU-MOC: A benchmark for fine-grained multicategory object counting in aerial images. *IEEE Trans. Geosci. Remote Sens.*, 62:1–14, 2024.
- [78] Gong Cheng, Junwei Han, Peicheng Zhou, and Lei Guo. Multi-Class Geospatial Object Detection and Geographic Image Classification Based on Collection of Part Detectors. *ISPRS J. Photogramm. Remote Sens.*, 98:119–132, 2014.
- [79] Gong Cheng and Junwei Han. A Survey on Object Detection in Optical Remote Sensing Images. *ISPRS J. Photogramm. Remote Sens.*, 117:11–28, 2016.
- [80] Gong Cheng, Peicheng Zhou, and Junwei Han. Learning Rotation-Invariant Convolutional Neural Networks for Object Detection in VHR Optical Remote Sensing Images. *IEEE Trans. Geosci. Remote Sens.*, 54(12):7405–7415, 2016.
- [81] Yang Long, Yiping Gong, Zhifeng Xiao, and Qing Liu. Accurate Object Localization in Remote Sensing Images Based on Convolutional Neural Networks. *IEEE Trans. Geosci. Remote Sens.*, 55(5):2486–2498, 2017.
- [82] Zhifeng Xiao, Qing Liu, Gefu Tang, and Xiaofang Zhai. Elliptic Fourier Transformation-Based Histograms of Oriented Gradients for Rotationally Invariant Object Detection in Remote-Sensing Images. *Int. J. Remote Sens.*, 36(2):618–644, 2015.
- [83] Bartosz Ptak and Marek Kraft. Enhancing People Localisation in Drone Imagery for Better Crowd Management by Utilising Every Pixel in High-Resolution Images. *arXiv preprint arXiv:2502.04014*, 2025.

- [84] Simon Graham, Quoc Dang Vu, Mostafa Jahanifar, Martin Weigert, Uwe Schmidt, Wenhua Zhang, Jun Zhang, Sen Yang, Jinxi Xiang, Xiyue Wang, et al. CoNIC Challenge: Pushing the frontiers of nuclear detection, segmentation, classification and counting. *Med. Image Anal.*, 92:103047, 2024.
- [85] Anton Naumov, Egor Ushakov, Andrey Ivanov, Konstantin Midiber, Tatyana Khovanskaya, Alexandra Konyukova, Polina Vishnyakova, Sergei Nora, Liudmila Mikhaleva, Timur Fatkhudinov, and Evgeny Karpulevich. EndoNuke: Nuclei detection dataset for estrogen and progesterone stained IHC endometrium scans. *Data*, 7(6):75, 2022.
- [86] Simon Graham, Mostafa Jahanifar, Ayesha Azam, Mohammed Nimir, Yee-Wah Tsang, Katherine Dodd, Emily Hero, Harvir Sahota, Atisha Tank, Ksenija Benes, et al. Lizard: A large-scale dataset for colonic nuclear instance segmentation and classification. In *Int. Conf. Comput. Vis. Worksh.*, pages 684–693, 2021.
- [87] Neeraj Kumar, Ruchika Verma, Deepak Anand, Yanning Zhou, Omer Fahri Onder, Efstratios Tsougenis, Hao Chen, Pheng-Ann Heng, Jiahui Li, Zhiqiang Hu, et al. A Multi-Organ Nucleus Segmentation Challenge. *IEEE Trans. Med. Imag.*, 39(5):1380–1391, 2020.
- [88] Mohamed Amgad, Lamees A. Atteya, Hagar Hussein, Kareem Hosny Mohammed, Ehab Hafiz, Maha A. T. Elsebaie, Ahmed M. Alhusseiny, Mohamed Atef AIMoslemany, Abdelmagid M. Elmatboly, Philip A. Pappalardo, Rokia Adel Sakr, et al. NuCLS: A scalable crowdsourcing approach and dataset for nucleus classification and segmentation in breast cancer. *GigaScience*, 11:giac037, 2022.
- [89] Mathieu Gendarme, Annika M. Lambert, and Bachir El Debs. BriFiSeg: A deep learning-based method for semantic and instance segmentation of nuclei in brightfield images. *arXiv preprint arXiv:2211.03072*, 2022.
- [90] Victor Lempitsky and Andrew Zisserman. Learning To Count Objects in Images. In *Adv. Neural Inform. Process. Syst.*, pages 1324–1332, 2010.
- [91] Christoph Spahn, Estibaliz Gómez-de Mariscal, Romain F. Laine, Pedro M. Pereira, Lucas von Chamier, Mia Conduit, Mariana G. Pinho, Guillaume Jacquemet, Séamus Holden, Mike Heilemann, et al. DeepBacs for multi-task bacterial image analysis using open-source deep learning approaches. *Commun. Biol.*, 5(1):688, 2022.
- [92] Etienne David, Mario Serouart, Daniel Smith, Simon Madec, Kaaviya Velumani, Shouyang Liu, Xu Wang, Francisco Pinto, Shahameh Shafiee, Izzat S. A. Tahir, et al. Global Wheat Head Detection 2021: An improved dataset for benchmarking wheat head detection methods. *Plant Phenomics*, 2021:9846158, 2021.
- [93] Luca Ciampi, Ali Azmoudeh, Elif Ecem Akbaba, Erdi Sarıtaş, Ziya Ata Yazıcı, Hazım Kemal Ekenel, Giuseppe Amato, and Fabrizio Falchi. A Survey on Class-Agnostic Counting: Advancements from reference-based to open-world text-guided approaches. *Comput. Vis. Image Underst.*, 267:104703, 2026.

Appendix

Contents

A Related Work	16
A.1 Object Counting Paradigms	16
A.2 Generalist Object Counting	17
A.3 Counting Datasets	17
B Additional Implementation Details and Experiments	18
B.1 Implementation Details	18
B.2 Quantitative Experiments	19
B.3 Qualitative Experiments	21
C Detailed Report on CLOC Dataset	28
C.1 Introduction to CLOC dataset	28
C.1.1 Motivation	28
C.1.2 Overview of CLOC	28
C.2 Dataset Statistics	29
C.2.1 Visual-domain Distribution	29
C.2.2 Category Distribution	31
C.2.3 Target-count Distribution	32
C.2.4 Image-resolution Distribution	32
C.3 Dataset Construction Pipeline	33
C.3.1 Multi-source Raw Datasets	33
C.3.2 Countability Audit and Annotation Cleaning	34
C.3.3 Unified Counting Annotation Construction	36
C.3.4 Category Consolidation and Derived Sample Generation	37
C.3.5 Task-aware Splitting	39
C.4 Evaluation Protocol	40
D Discussion	41
D.1 Discussion of the Method	41
D.2 Discussion of the Dataset	42

A Related Work

A.1 Object Counting Paradigms

Existing object counting methods can be broadly categorized into density-estimation-based, detection-based, and point-regression-based paradigms. Density-estimation-based methods predict a density map and obtain the count by integration. Representative works such as MCNN [14], CSRNet [49], and SANet [50] improve density regression through multi-scale modeling, enlarged receptive fields, and high-resolution feature aggregation, while Bayesian Loss [51] and DM-Count [28] further improve

supervision and optimization. These methods are effective for extremely dense or heavily occluded scenes, but density maps are continuous response fields without one-to-one prediction-instance correspondence. They therefore cannot naturally provide instance-level locations, confidence scores, or localization-level error diagnosis, which limits their suitability for instance-grounded generalist counting.

Detection-based methods formulate counting as detecting target instances and counting the detections. LSC-CNN [26], CARPK/LPN [27], and Counting-DETR [13] demonstrate the effectiveness of this paradigm in crowd, drone-view vehicle, and few-shot object counting. Detection-based counting provides explicit object extents and region-level spatial evidence, making it suitable for large, sparse, or clearly bounded targets. However, bounding boxes are less reliable for small, crowded, weakly bounded, or heavily occluded targets, where instances may be missed, merged, or duplicated.

Point-regression-based methods predict discrete target points by regressing point coordinates, and the cardinality of the point set gives the count. P2PNet [10] is a representative method that uses one-to-one matching supervision to regress target centers and predict confidence scores. Compared with density maps, point regression provides discrete instance locations; compared with boxes, it avoids estimating complete object boundaries, making it suitable for dense or weakly bounded targets. However, point regression lacks object extent and region-level geometric constraints, which may be important for large or sparse objects. These observations suggest that detection-based and point-regression-based counting are complementary: the former provides region-level anchoring, while the latter enables pixel-level dense enumeration. This motivates our dual-granularity design for generalist text-guided counting.

A.2 Generalist Object Counting

Unlike category-specific counting, generalist object counting aims to count arbitrary targets specified by categories, exemplars, or text. Early category-conditioned and few-shot methods mainly rely on visual exemplars. FSC-147 [12] extends counting to diverse categories in general-scene images, while FamNet [12] and CounTR [52] use exemplars or Transformer-based modeling for few-shot or zero-shot counting. These methods reduce the dependence on fixed target categories, but they usually require exemplar boxes or patches, whereas category names or natural-language descriptions are more convenient in real user interaction.

Reference-free and text-guided methods further reduce exemplar dependence. Reference-free methods [11] mine repetitive visual patterns for class-agnostic counting, but they often assume that repeated patterns correspond to the target category and thus cannot reliably follow user-specified semantics. Text-guided methods instead use category names or descriptive phrases. CLIP-Count [29], VLCounter [4], PseCo [53], and T2ICount [40] improve text-guided counting by leveraging vision-language pretraining, prompt tuning, segmentation candidates, or diffusion priors. However, precise alignment between textual semantics and local visual evidence remains challenging under semantic ambiguity, background distractors, and large-scale variations.

Recent methods combine generalist counting with detection-style reasoning or foundation models to obtain more explicit instance candidates. DAVE [54], CountGD [6], CountGD++ [7], and CountVid [55] improve open-world counting through detect-and-verify, generalized prompting with open-vocabulary detection foundations, or video-level detection, segmentation, and tracking. These methods show the value of detection-style representations and foundation models, but they often rely on complex candidate generation, verification, segmentation, or tracking modules, and some still require visual exemplars or multimodal prompts. Meanwhile, detection candidates remain sensitive to small, dense, and weakly bounded targets. Existing generalist counting methods therefore still struggle to jointly satisfy text-guided target specification, instance-grounded counting, cross-domain generalization, and dense-target recall.

A.3 Counting Datasets

Counting datasets have long been built around specific application scenarios. Crowd counting datasets such as ShanghaiTech [14], JHU-CROWD++ [19], and NWPU-CROWD [56] focus on pedestrians or heads, while vehicle, crop, cell nucleus, and microorganism counting datasets serve traffic monitoring, agricultural phenotyping, biomedical analysis, and laboratory automation [27, 57, 31, 58]. These datasets have clear task definitions within their own domains, but they usually cover a single visual

domain, limited categories, or a specific density range, making them insufficient for cross-category and cross-domain generalist counting.

Category-conditioned datasets enlarge the target space. FSC-147 [12] supports few-shot and class-agnostic counting, FSCD-LVIS [13, 33] combines counting with detection, and OmniCount-191 [59] focuses on multi-label object counting. However, these benchmarks are still mainly based on general-scene images or limited visual sources. Meanwhile, detection, segmentation, visual grounding, and specialized-domain datasets contain rich instance information that could be converted for counting [17, 23, 33, 60, 61, 62, 25, 63, 64]. Directly merging them is not sufficient, since their annotation forms, instance definitions, and special-region handling rules are inconsistent, which may introduce inconsistent supervision semantics, missing-label noise, and evaluation bias.

Building a generalist counting dataset therefore requires auditing and reorganizing multi-source annotations around one-instance-one-count supervision. Special labels or regions such as crowd, group, ignore, NegativeROA, and AMBIGUOUS require source-specific handling, and heterogeneous annotations such as boxes, points, polygons, masks, rotated boxes, and label maps need to be converted into a unified counting format. CLOC is constructed under this motivation by reorganizing public data sources into a unified category-specified benchmark across six visual domains. Compared with existing single-domain or mainly general-scene counting datasets, CLOC emphasizes cross-domain coverage, scale, and unified counting semantics for generalist text-guided counting.

B Additional Implementation Details and Experiments

B.1 Implementation Details

The Text-Conditioned Encoder of Count Anything is initialized from the pretrained SAM3 model. During training, the original parameters of the pretrained Text-Conditioned Encoder are frozen. We insert trainable LoRA adapters into the cross-modal encoder for lightweight adaptation, with rank $r = 8$, scaling factor $\alpha = 8$. The Region-level Sparse Counter (RSC) follows the region-query decoding formulation in Sec. 3 and uses $Q_r = 200$ region queries. Each region query predicts a foreground logit and a candidate box, whose center is used as the RSC counting point. For the Pixel-level Dense Counter (PDC), we use the dense spatial grid $\Omega = \{u_k\}_{k=1}^{Q_d}$ and anchor point $a_k = \pi(u_k)$ defined in Sec. 3. In our implementation, each grid location generates one PDC candidate, and adjacent anchor points are spaced by 7 pixels in the input-image coordinate system. The PDC regression head predicts a two-dimensional offset Δ_k relative to the anchor point. The offset scaling factor is set to $\rho = 100$, so the predicted PDC point is computed as

$$\hat{p}_k^d = a_k + 100\Delta_k. \quad (12)$$

The PDC feature adapter contains 3 residual blocks with 256 channels and group normalization.

Training is performed with distributed data parallelism using the NCCL backend. The per-GPU training batch size is 24, and the gradient accumulation step is 1, resulting in an effective training batch size of 192. We train for 30 epochs and evaluate after each epoch. The random seed is fixed to 123. The optimizer is AdamW with $(\beta_1, \beta_2) = (0.9, 0.999)$, $\epsilon = 10^{-8}$, and weight decay 10^{-4} . We use bfloat16 automatic mixed precision. Gradients are clipped by the global ℓ_2 norm with a maximum norm of 0.1.

The learning rate is scheduled by an epoch-wise cosine scheduler from epoch 0 to epoch 30 without warm-up. The minimum learning rate ratio is 0.1. The initial learning rates are 1×10^{-5} for RSC, 1×10^{-4} for PDC, and 1×10^{-3} for LoRA adapters. Accordingly, the final learning rates after cosine decay are 1×10^{-6} , 1×10^{-5} , and 1×10^{-4} for the three parameter groups, respectively.

All samples are trained and evaluated at an input resolution of 1008×1008 . During training, we randomly choose one of two preprocessing paths with equal probability. The first path applies aspect-ratio-preserving random resizing with a scale sampled from $[0.5, 2.0]$, constrains the shorter side to be at least 256 pixels and the longer side to be at most 1500 pixels, and then randomly crops or pads the image to 1008×1008 . The second path directly resizes the image to a square input of 1008×1008 and pads it when needed. Validation and testing use deterministic resize/pad preprocessing. Images are converted to tensors and normalized with mean $(0.5, 0.5, 0.5)$ and standard deviation $(0.5, 0.5, 0.5)$. Category names are used as text queries, and each image loads at most 8 training or validation queries.

For branch-specific matching, RSC uses Hungarian matching. Its matching cost consists of a foreground confidence cost and a point-distance cost, whose weights are 1.0 and 0.05, respectively. PDC uses sparse Hungarian matching. For each ground-truth point, the local candidate set $\mathcal{N}(j)$ contains the top-5 nearby dense candidates. The PDC matching cost also uses foreground confidence and point distance, with weights 1.0 and 0.05, respectively.

The overall loss follows the point-centric supervision in Sec. 3. For RSC, the weights of classification loss, point loss, box loss, and GIoU loss are 20.0, 5.0, 5.0, and 2.0, respectively. The soft foreground target in RSC classification uses quality-balancing coefficient $\alpha = 0.25$ and minimum positive label value $\epsilon = 0.01$. For PDC, the weights of classification loss and point-regression loss are 1.0 and 2×10^{-4} , respectively.

During inference, CCF uses score thresholds $\tau_r = 0.5$ and $\tau_d = 0.5$ for RSC and PDC candidates. RSC duplicate removal processes RSC boxes in descending order of score. A lower-scored region is removed if it is fully contained by a retained region or if its intersection over minimum area (IoM) with a retained region is larger than 0.5. After RSC filtering, CCF first identifies, for each retained RSC box, the PDC point inside the box that is closest to the corresponding RSC center. It then compares the RSC score with the score of this nearest PDC point and suppresses the lower-scored one. In this way, CCF resolves at most one RSC–PDC duplicate pair for each retained RSC box. The remaining RSC and PDC points are merged as the final prediction, and the final count is the size of the fused point set.

B.2 Quantitative Experiments

Table 7: Model complexity and inference efficiency of Count Anything under 1008×1008 input resolution. Latency and memory are measured with batch size 1 using BF16 inference.

Model	Params	FLOPs	GPU	Latency (ms)	Memory (GB)
Count Anything	848.62M	5.44T	RTX 5090	68.51	5.17
			RTX 4090	95.55	6.53
			A100	122.11	6.53

Additional Ablation Studies – Computational efficiency. We evaluate the computational efficiency of Count Anything under the input resolution of 1008×1008 . As shown in Table 7, Count Anything contains 848.62M parameters and requires 5.44 TFLOPs per forward pass. Despite operating on high-resolution images with dual counting branches, the model maintains practical inference efficiency on modern GPUs. Specifically, it achieves an average latency of 68.51 ms on an RTX 5090, 95.55 ms on an RTX 4090, and 122.11 ms on an A100, with peak inference memory ranging from 5.17 GB to 6.53 GB. These results indicate that Count Anything can provide high-resolution text-guided counting while maintaining feasible computational cost for practical deployment.

Table 8: Ablation on point-centric supervision components.

Supervision		Metric	
RSC point	Box and GIoU	MAE	RMSE
✗	✓	14.06	89.80
✓	✗	13.83	61.92
✓	✓	9.34	33.34

Point-centric supervision components. Table 8 studies the effect of the core supervision components in point-centric supervision. Removing the RSC point loss weakens RSC’s ability to learn counting-point localization from point-only annotations, leading to substantially larger errors. Removing the box and GIoU losses weakens the region-level geometric supervision of RSC, especially for domains where object extent provides useful anchoring. Using both RSC point supervision and box-level geometric supervision achieves the best performance, showing that point-level localization and box-level geometry provide complementary supervision for RSC.

PDC feature adapter. Table 9 studies the effect of the feature adapter in PDC. Without the adapter, PDC directly performs point prediction from the fused pixel-level representation, leading to larger

Table 9: Ablation on the PDC feature adapter.

Adapter design	Metric	
	MAE	RMSE
None	12.48	48.25
1 residual block	11.04	41.20
3 residual blocks	9.34	33.34

errors. A shallow adapter improves the results but remains weaker than the full setting. The complete 3-block residual adapter achieves the best performance, showing that feature adaptation before the PDC point classification and point regression heads improves pixel-level dense counting.

Table 10: Sensitivity analysis of the local duplicate-group size in CCF.

K_s	Local duplicate group	Metric	
		MAE	RMSE
0	direct union	22.85	50.05
1	RSC + nearest PDC candidate	9.34	33.34
2	RSC + 2 nearest PDC candidates	9.50	34.26
4	RSC + 4 nearest PDC candidates	9.75	36.93
all	RSC + all PDC candidates inside the region	9.85	37.93

CCF local duplicate-group size. Table 10 analyzes the effect of the local duplicate-group size in CCF. This ablation generalizes the default nearest-PDC comparison used in CCF. We denote by K_s the maximum number of nearest PDC candidates grouped with each retained RSC candidate inside its region; the default CCF in our method exactly corresponds to $K_s = 1$. When $K_s = 0$, the two-branch outputs are directly merged, keeping duplicate predictions from RSC and PDC for the same target. When $K_s > 0$, each retained RSC candidate forms a local duplicate group with its K_s nearest PDC candidates inside the region, and only the highest-confidence candidate in the group is kept. As K_s increases beyond 1, more nearby PDC candidates are included in the same duplicate group, which can remove useful dense-branch predictions in crowded regions and gradually increases counting errors. When $K_s = \text{all}$, all PDC candidates inside each RSC region are included in the group. The setting $K_s = 1$ achieves the best performance, showing that grouping only the nearest PDC candidate better balances duplicate-count suppression and dense point recall.

Table 11: Sensitivity analysis of the IoM threshold.

IoM threshold	Metric	
	MAE	RMSE
0.3	9.43	33.41
0.5	9.34	33.34
0.7	9.55	33.50
0.9	10.01	33.82

IoM threshold. Table 11 studies the effect of the IoM threshold used in RSC duplicate removal. As the IoM threshold increases, two regions need a higher overlap ratio to be regarded as duplicates, weakening the removal of duplicate RSC predictions. When the IoM threshold is too low, the removal rule becomes overly permissive and may suppress nearby non-duplicate regions. The setting of 0.5 achieves the best performance in this group of experiments and is consistent with the IoM-NMS threshold used in SAM3 [35] object counting evaluation. Therefore, we use IoM threshold 0.5 as the default setting in our main experiments.

PDC candidate density. Table 12 compares different numbers of PDC candidates generated at each spatial location. This ablation corresponds to the candidate generation process on the dense spatial grid in PDC. With the input resolution of 1008×1008 , the default 1×1 setting already generates PDC candidates at approximately 7×7 pixel intervals, providing sufficiently dense point coverage. Simply increasing the number of candidates per spatial location introduces more adjacent candidates under

Table 12: Sensitivity analysis of PDC candidate density.

Candidates per cell	Metric	
	MAE	RMSE
1×1	9.34	33.34
2×2	20.38	122.88
3×3	21.47	128.80

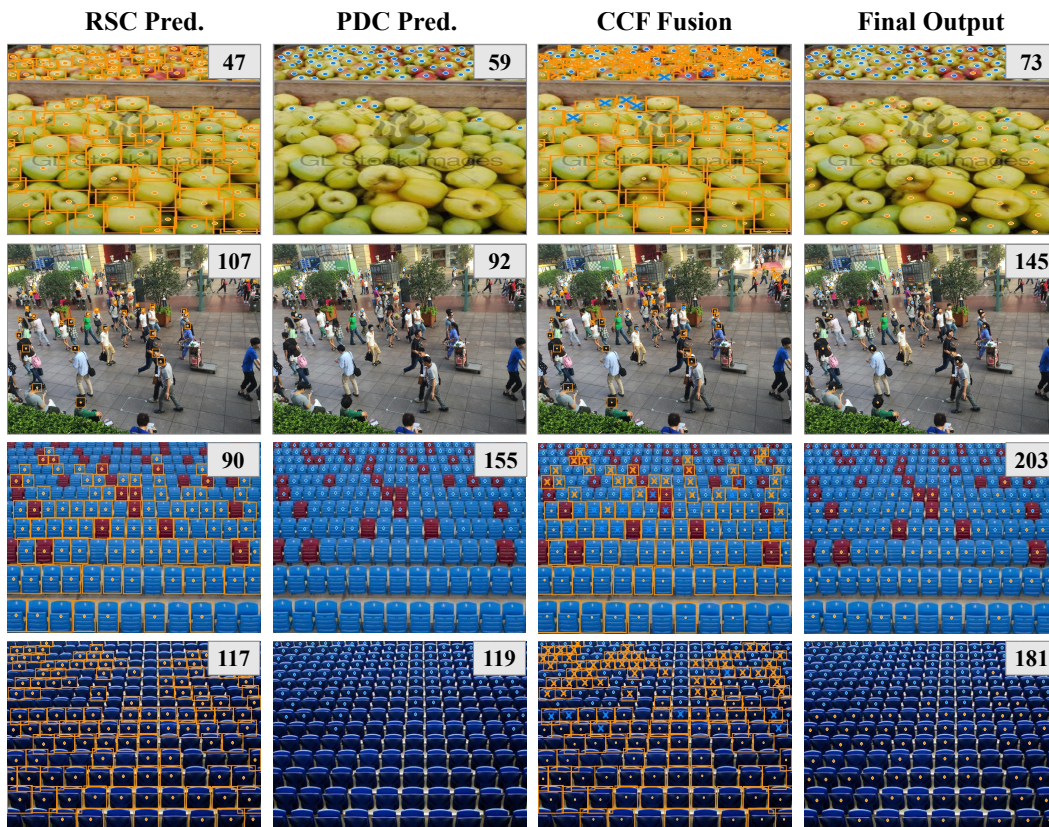


Figure 5: Qualitative visualization of Complementary Count Fusion. Orange boxes and points denote RSC predictions, while blue points denote PDC predictions. In the CCF panel, crossed markers indicate suppressed duplicate predictions between the two branches. The final output preserves complementary predictions from both branches, and the number in the top-right corner denotes the predicted count.

the same sparse-matching and loss configuration. This increases foreground-background imbalance and duplicate candidate competition, making optimization less stable and leading to larger counting errors. Therefore, we use one PDC candidate per spatial location as the default setting.

B.3 Qualitative Experiments

Visualization of Complementary Count Fusion. To further illustrate the interaction between the two counting branches, Figure 5 visualizes the RSC predictions, PDC predictions, fusion process, and final output. RSC provides region-level object anchors for clearly bounded instances, while PDC recovers dense point-level predictions; CCF suppresses duplicated predictions and retains complementary ones to form the final point set.

Additional domain-wise qualitative comparisons. To supplement Figure 3, we provide additional domain-wise qualitative comparisons on the CLOC test set in Figures 6–11, covering General Scene, Remote Sensing, Histopathology, Cellular Microscopy, Agriculture, and Microbiology. We additionally include CountSE [41] and GroundingDINO [44] as compared methods.

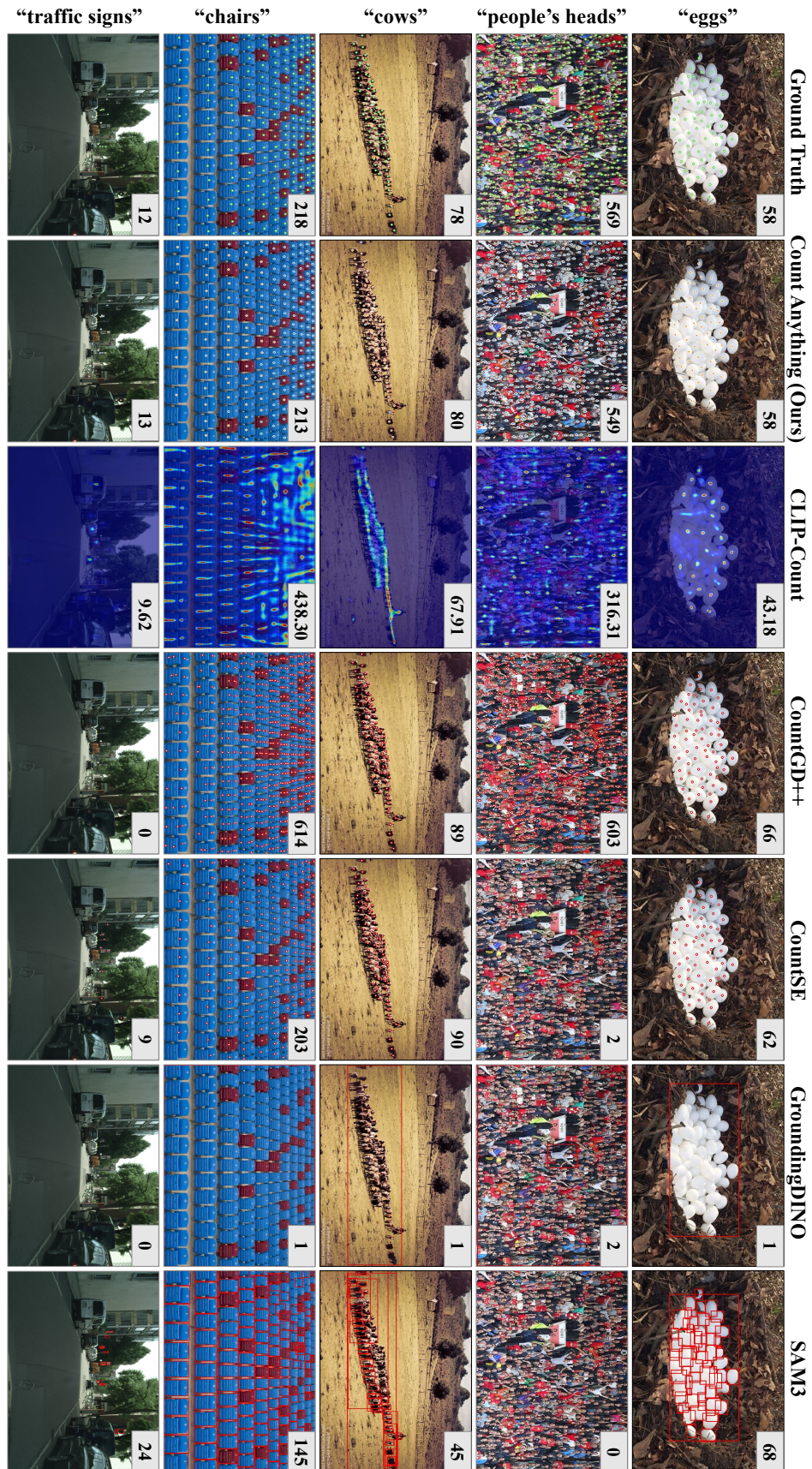


Figure 6: Additional qualitative comparisons on the General Scene domain.

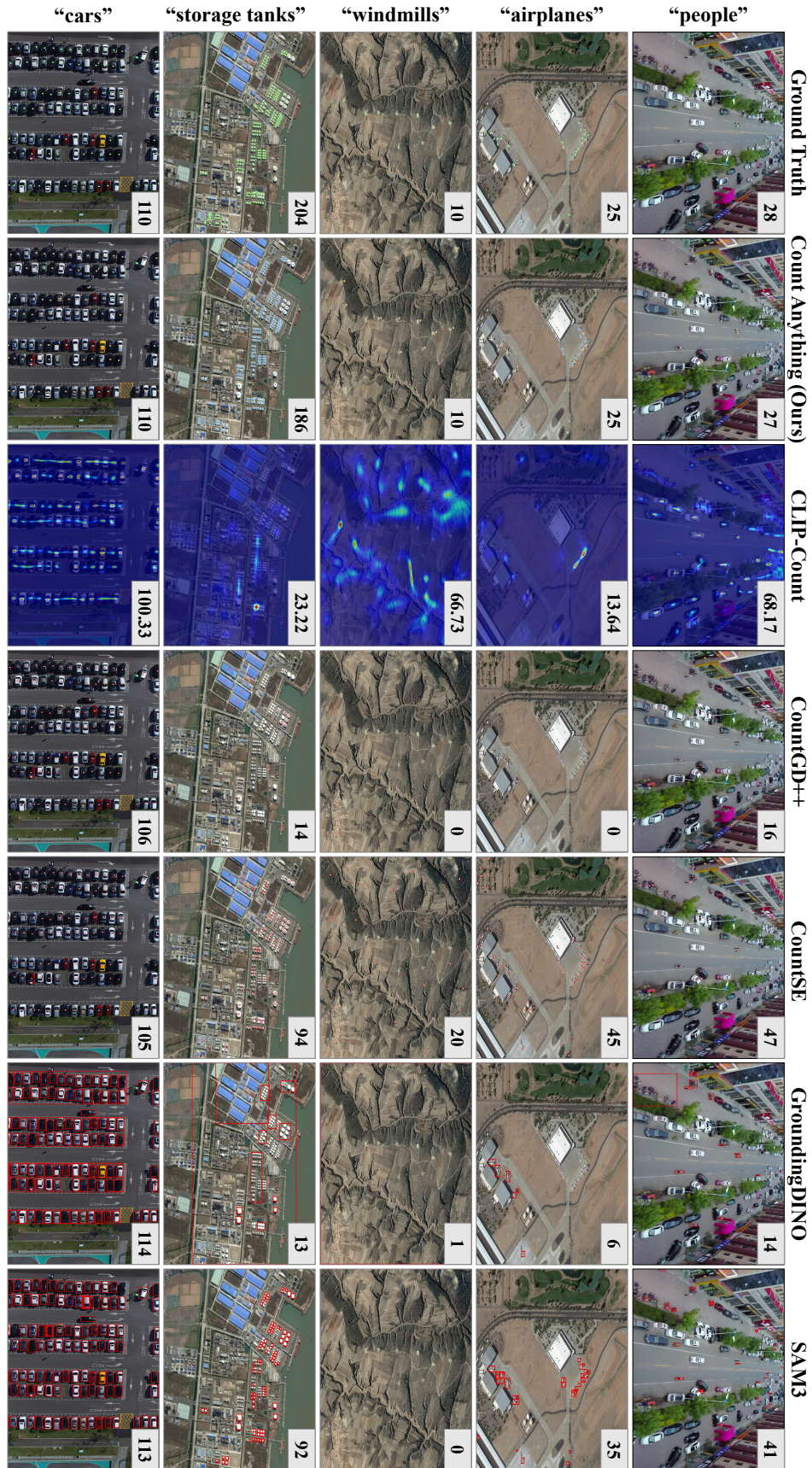


Figure 7: Additional qualitative comparisons on the Remote Sensing domain.

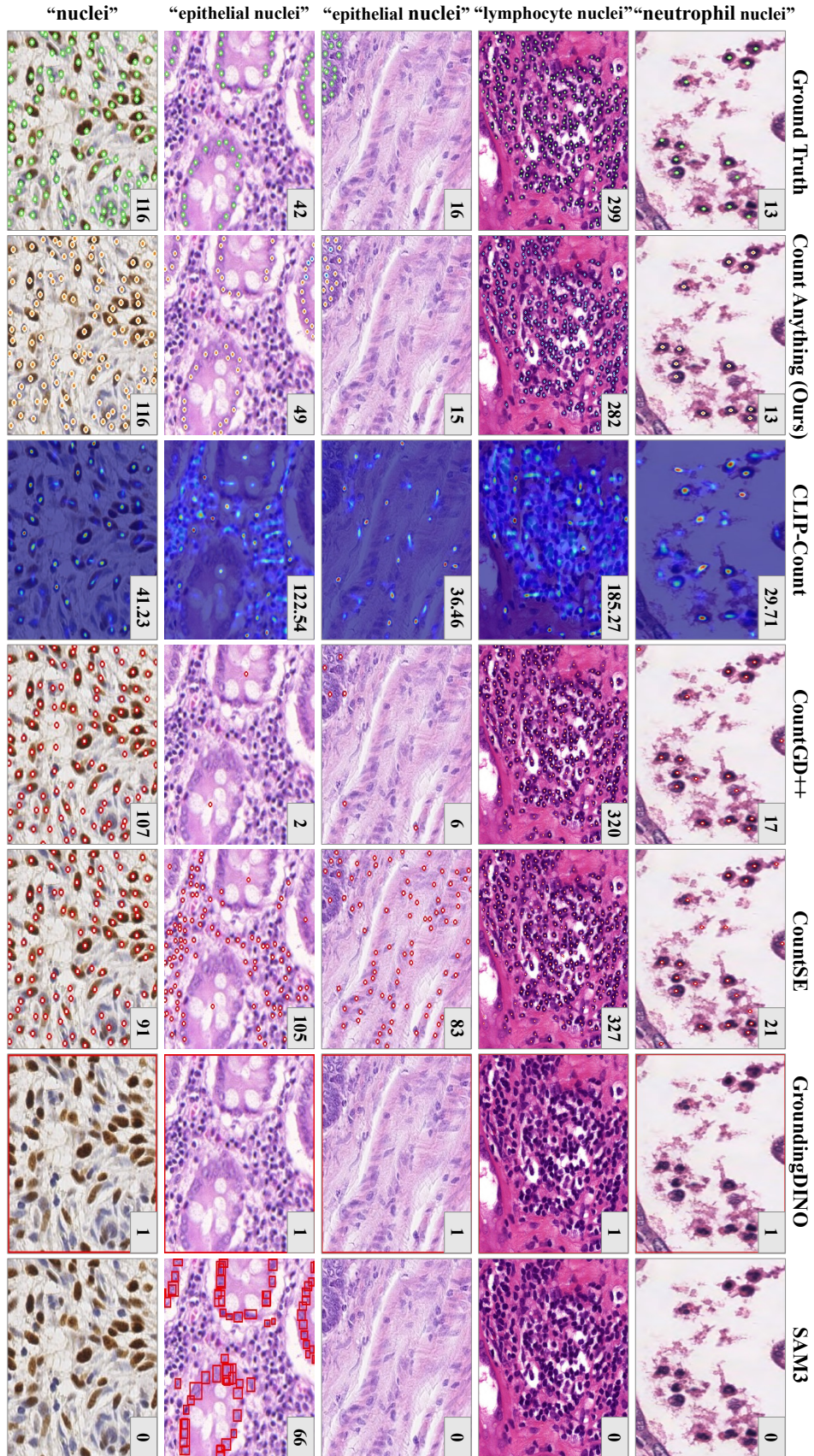


Figure 8: Additional qualitative comparisons on the Histopathology domain.

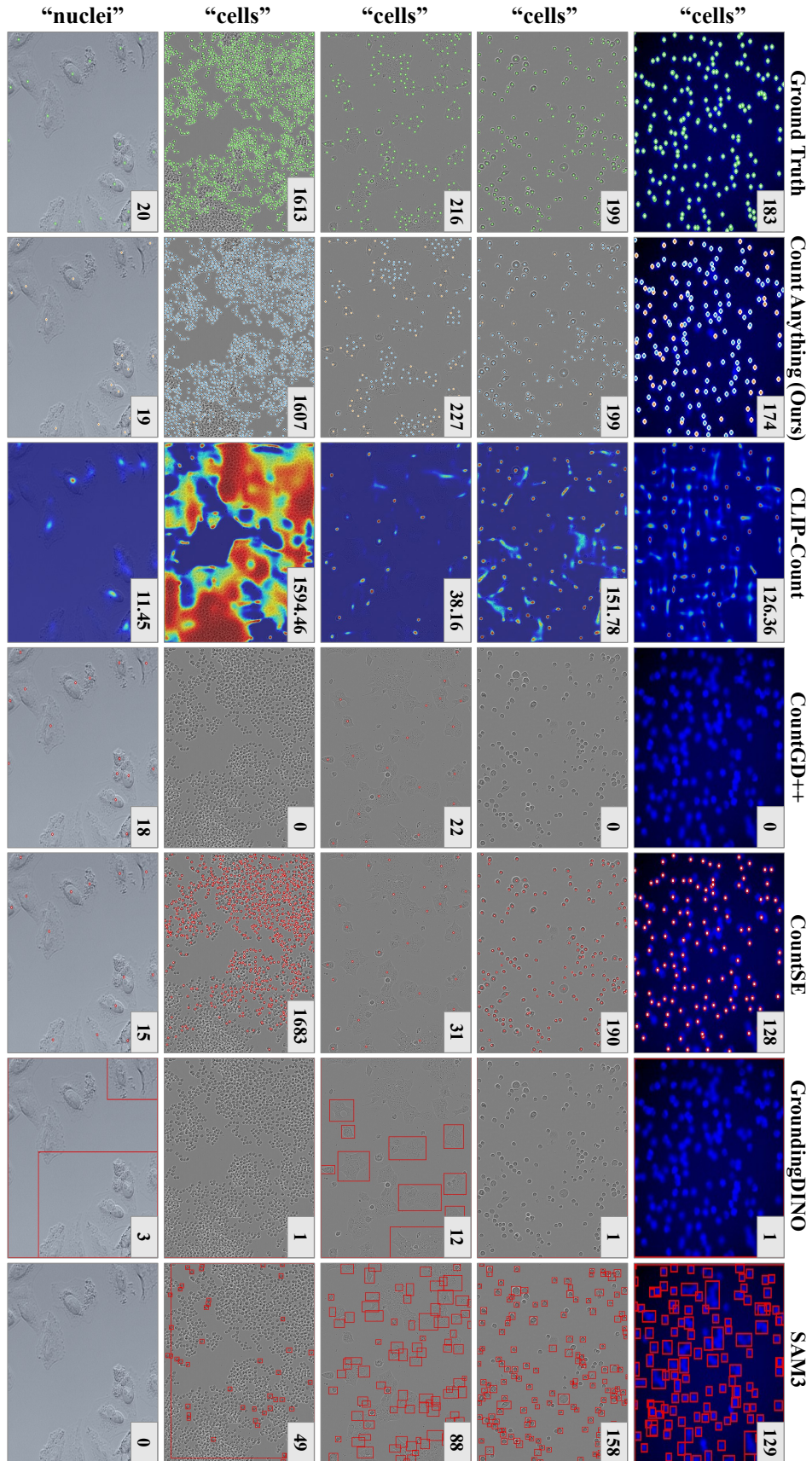


Figure 9: Additional qualitative comparisons on the Cellular Microscopy domain.

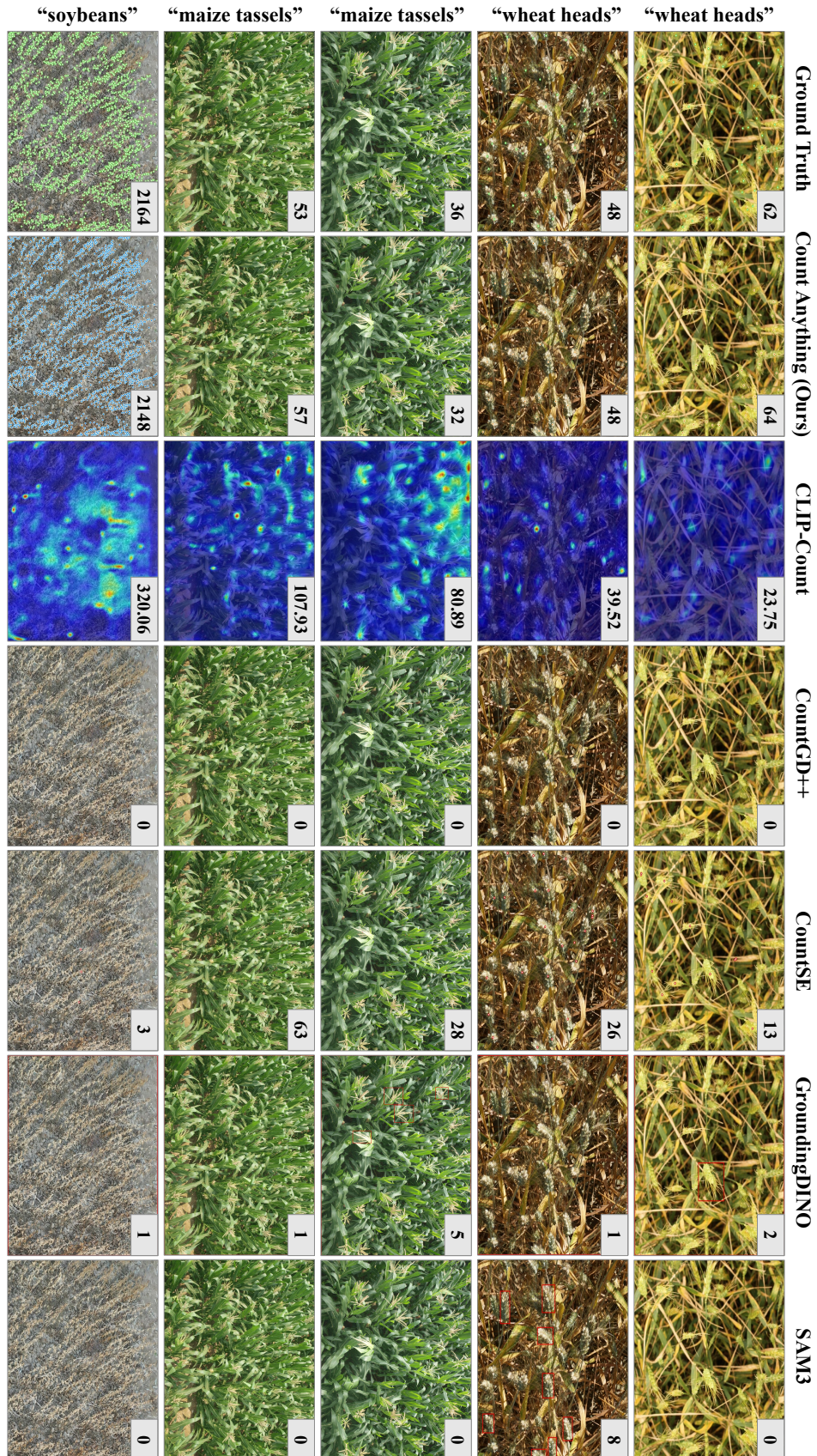


Figure 10: Additional qualitative comparisons on the Agriculture domain.

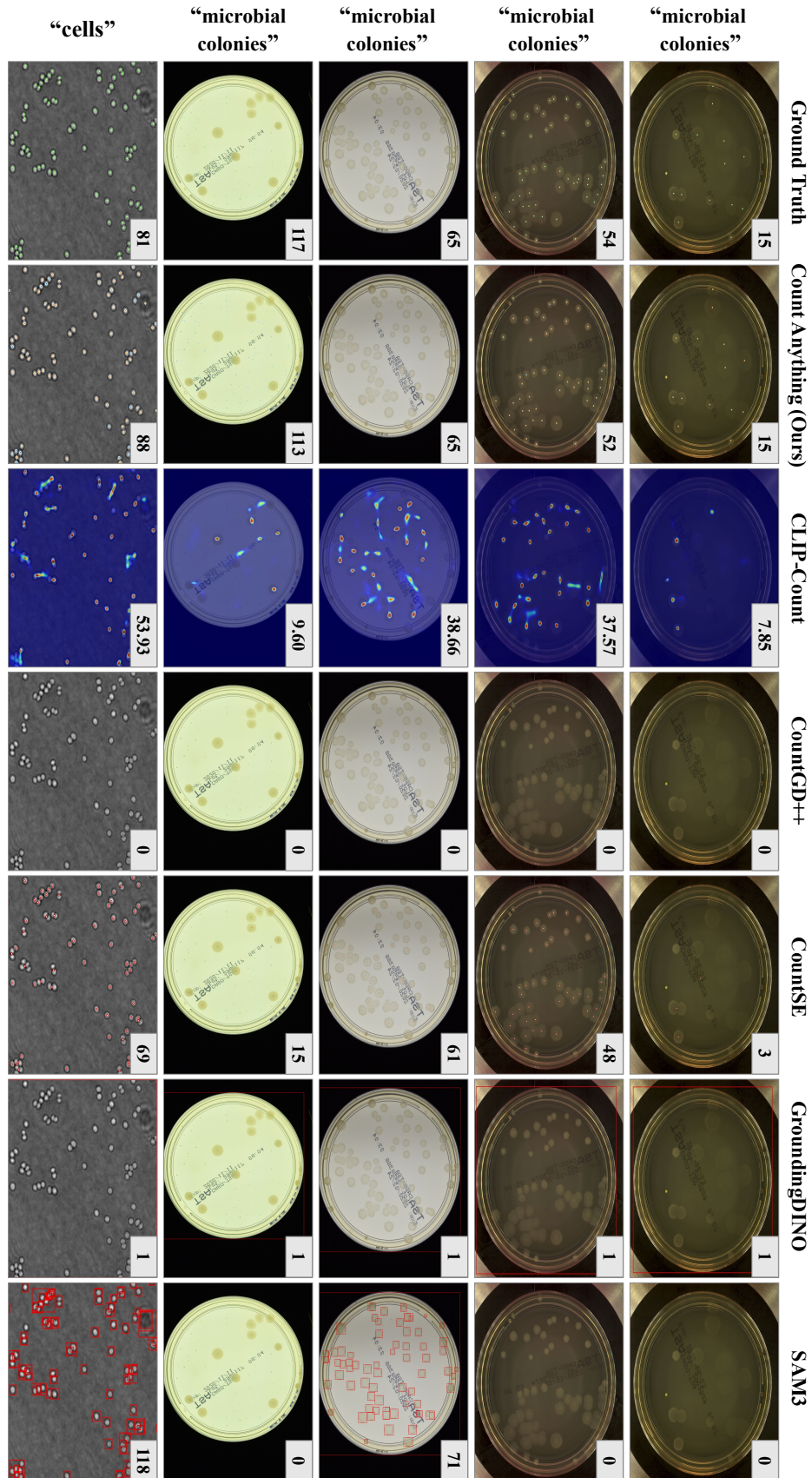


Figure 11: Additional qualitative comparisons on the Microbiology domain.

C Detailed Report on CLOC Dataset

C.1 Introduction to CLOC dataset

C.1.1 Motivation

Recent progress in generalist vision models has shown that large-scale and diverse data are essential for enabling models to generalize in open-world scenarios [65, 66]. However, in object counting, existing datasets still provide limited support for training and evaluating generalist counting models. Crowd counting datasets represented by ShanghaiTech [14] mainly focus on counting people or heads, where the task definition is clear but the target type is limited. Category-conditioned counting datasets such as FSC-147 [12] expand the range of countable categories, but their image scale and visual-domain coverage remain limited. As counting tasks move toward real-world applications, models need to handle diverse visual domains such as General Scene, Remote Sensing, Histopathology, Cellular Microscopy, Agriculture, and Microbiology, as well as target-count variations from sparse objects to extremely dense scenes. Existing counting datasets remain limited in visual-domain coverage, data scale, and scene diversity, which restricts the development of generalist object counting models.

A general-purpose counting dataset for real-world scenarios should satisfy three basic requirements. First, it should cover multiple visual domains, so that models can learn counting patterns under different imaging conditions, object appearances, and scene structures, rather than adapting only to a fixed type of scenario. Second, it should have sufficient scale in terms of images, categories, and instances to support the training of generalist counting models. Finally, the dataset should establish unified annotation semantics for counting, so that each valid target can be consistently associated with one countable instance. Only with such a unified protocol can models be trained and evaluated across different categories, visual domains, and target-density ranges.

C.1.2 Overview of CLOC

Motivated by the above observations, we construct **CLOC**, a large-scale cross-domain object counting dataset for real-world scenarios. CLOC is built upon multiple public datasets and reorganizes their images, categories, and instance annotations around the counting task. It covers General Scene, Remote Sensing, Histopathology, Cellular Microscopy, Agriculture, and Microbiology in a unified text-guided counting setting. In this setting, category names are used as textual queries: given an image and a target query, the model is required to predict the number of instances corresponding to that query in the image. Thus, category-specified counting in CLOC serves as a concrete form of text-guided object counting.

By selecting countable instances, unifying the category space, and reorganizing counting annotations, data originally designed for detection, segmentation, crowd counting, medical image analysis, remote sensing recognition, and agricultural phenotyping are transformed into a unified data resource for training and evaluating generalist object counting models.

Table 13: Statistical comparison with representative counting and detection datasets. The table compares different datasets in terms of the number of images, categories, instances, instances per image, standard deviation of instances per image, and cross-domain coverage.

Dataset	Images	Categories	Instances	Instances/img	Std. Inst./img	Cross-domain
OmniCount-191 [59]	30,230	191	302.3K	10.00	14.50	×
FSC-147 [12]	6,135	147	343.8K	56.04	108.35	×
MS COCO [17]	328K	91	2,500K	7.70	7.38	×
VOC2007 [18]	9,963	20	24.6K	2.47	3.09	×
Objects365 [23]	638K	365	10,101K	15.80	11.61	×
FSCD-LVIS [13]	6,195	372	≥220.7K	≥35.62	41.90	×
CLOC (Ours)	220K	619	15,356K	69.74	187.76	✓(6)

Table 13 compares CLOC with representative counting and detection datasets. Overall, CLOC has two core characteristics: **cross-domain coverage** and **large scale**. Existing counting datasets are

mostly built around a single visual source, whereas CLOC covers six visual domains and enables models to be evaluated under a unified category-specified counting setting across different visual sources. Meanwhile, CLOC contains about 220K images, 619 categories, and 15.356M instances, exceeding existing general counting datasets in both category scale and instance scale, and providing a stronger data foundation for training and evaluating generalist object counting models.

Beyond cross-domain coverage and scale, CLOC also provides a wider target-count distribution. Its higher average number of instances per image and larger standard deviation indicate that the dataset includes not only low- and medium-density counting scenes, but also high-density and extremely dense target scenes. This distribution allows models to be evaluated under different levels of counting difficulty, instead of being tested only within a fixed density range.

At the category and visual-domain levels, CLOC also preserves the complex distributions of multi-source real-world data. On the one hand, its 619 countable categories form a large category space with a long-tailed distribution. On the other hand, its six visual domains differ substantially in image source, imaging style, and target appearance, allowing the benchmark to evaluate not only model performance in the General Scene domain, but also adaptability to Remote Sensing, Histopathology, Cellular Microscopy, Agriculture, and Microbiology. Overall, CLOC is not a specialized counting dataset for a single scenario or target type, but a large-scale cross-domain benchmark for generalist object counting.

C.2 Dataset Statistics

This section analyzes the statistical properties of CLOC from four perspectives: visual domains, categories, target counts, and image resolution. The visual-domain distribution characterizes the cross-domain coverage of CLOC and the scale differences across domains. The category distribution describes the long-tailed structure of target categories and their cross-domain composition. The target-count distribution shows the range of target counts in image-category pairs. The image-resolution distribution reflects scale differences across multi-source images. These statistics show that CLOC is diverse in visual domains, target categories, count ranges, and image scales, providing a data foundation for evaluating model generalization in complex cross-domain counting scenarios.

C.2.1 Visual-domain Distribution

Table 14: Image and instance distributions of CLOC across visual domains. The table reports the number of images and instances in the six visual domains of CLOC.

Domain	Images	Instances
General Scene	153,457	8,354,057
Remote Sensing	30,324	2,731,480
Histopathology	15,513	1,716,363
Cellular Microscopy	8,715	1,735,734
Agriculture	6,846	543,947
Microbiology	5,324	274,473

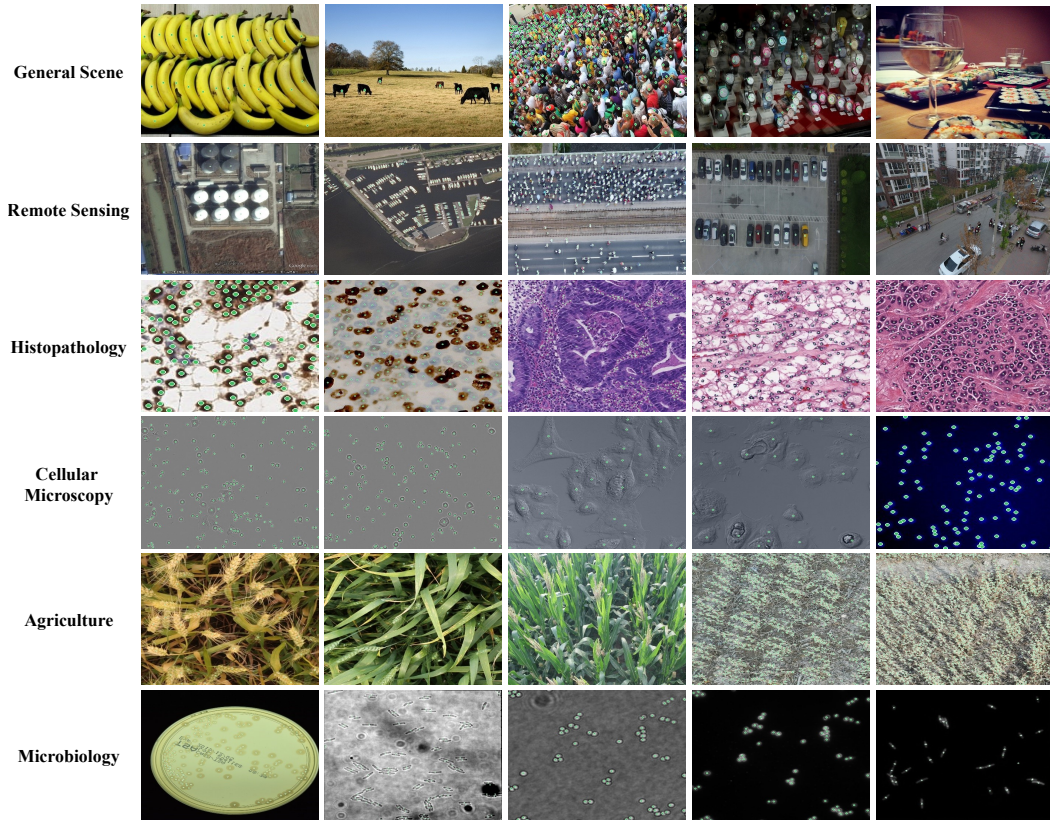


Figure 12: Representative visual examples from the six visual domains of CLOC. Each row shows samples from one visual domain, including General Scene, Remote Sensing, Histopathology, Cellular Microscopy, Agriculture, and Microbiology. These examples illustrate the visual heterogeneity introduced by the multi-source construction of CLOC.

As shown in Figure 12, CLOC covers six visually diverse domains: General Scene, Remote Sensing, Histopathology, Cellular Microscopy, Agriculture, and Microbiology. Unlike counting datasets built from a single visual source, CLOC incorporates multiple visual domains into a unified counting benchmark, allowing models to be evaluated under the same protocol on data from different visual conditions.

In terms of sample scale, the domains are not fully balanced after being converted into the final counting framework. General Scene and Remote Sensing account for relatively larger portions of the dataset, while specialized domains such as Histopathology, Cellular Microscopy, Agriculture, and Microbiology contain fewer counting samples. This imbalance is related to the original task forms and annotation availability of public multi-source data. For data from the General Scene and Remote Sensing domains, many public datasets are released with detection, counting, or instance-level target annotations, making them easier to convert into category-specified counting supervision. For data from the Histopathology, Cellular Microscopy, Agriculture, and Microbiology domains, public datasets are also widely available, but many of them are originally designed for classification, diagnosis, segmentation, or phenotyping tasks [67, 68], and do not necessarily provide complete and stable instance-level annotations that can be directly converted into counting supervision. As a result, the visual-domain scales in the final unified counting framework are not fully balanced.

It is worth emphasizing that the value of smaller visual domains does not mainly lie in their image quantity, but in the visual heterogeneity they introduce into CLOC. Remote Sensing, Histopathology, Cellular Microscopy, Agriculture, and Microbiology correspond to different imaging processes, observation scales, texture structures, and target appearances. Their counting scenarios differ substantially from conventional object counting in general-scene images. In other words, although these domains contain fewer samples, they broaden the visual coverage of the benchmark, allowing it to evaluate not only model performance in the General Scene domain, but also adaptability to Remote Sensing, Histopathology, Cellular Microscopy, Agriculture, and Microbiology. Therefore, the cross-domain

C.2.3 Target-count Distribution

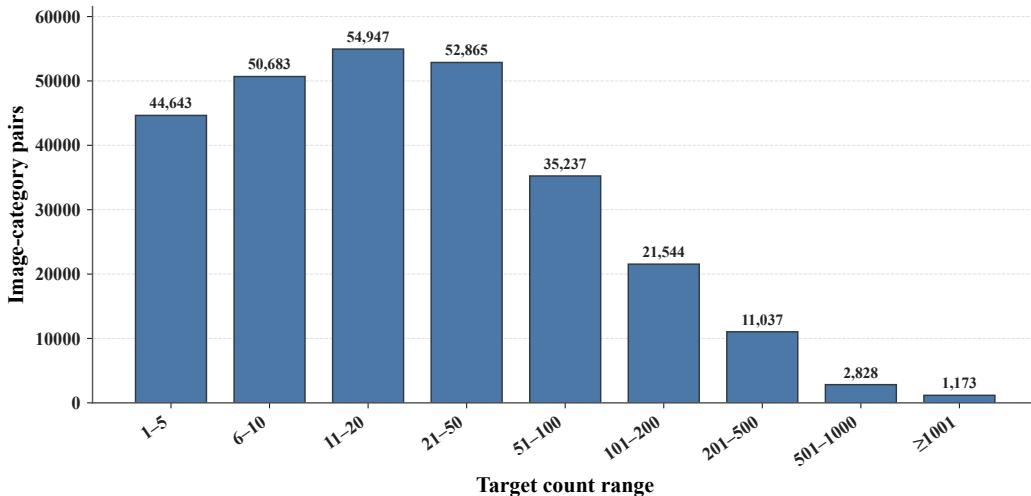


Figure 14: Target-count distribution. The x-axis indicates the target-count range for each image-category pair, and the y-axis indicates the number of corresponding image-category samples.

CLOC covers a wide counting range from 1-5 to 1001+. Low- and medium-count ranges contain a large number of samples and form the major portion of the dataset, showing that CLOC covers common counting scenarios in real-world applications. At the same time, CLOC also retains samples in higher-count ranges, enabling models to be evaluated under dense target scenes.

As the target count increases, the number of samples gradually decreases. This distribution is consistent with the target-count structure in real-world scenarios: most image-category pairs fall into low- to medium-count ranges, while high-density and extremely dense scenes are less frequent but still present. Therefore, CLOC can evaluate not only the basic counting ability of models under common target-count ranges, but also their stability in high-count and dense target scenarios.

C.2.4 Image-resolution Distribution

Table 15: Mean image resolution and megapixels of CLOC across visual domains. The table reports image-scale statistics across different visual domains and shows the resolution distribution of multi-source images.

Domain	Mean Resolution	Mean Megapixels
General Scene	887.52 × 705.75	0.864 MP
Remote Sensing	1246.06 × 1013.23	1.450 MP
Histopathology	360.26 × 357.57	0.161 MP
Cellular Microscopy	589.37 × 478.80	0.291 MP
Agriculture	1166.15 × 1117.30	1.522 MP
Microbiology	3023.68 × 3383.06	11.031 MP
Overall	941.31 × 786.71	1.125 MP

Since CLOC covers multiple visual domains, image scales vary substantially across domains. Overall, Microbiology has the highest average number of megapixels, while Agriculture and Remote Sensing also have relatively large image scales. In contrast, Histopathology and Cellular Microscopy have lower average resolutions.

These resolution differences are related to the imaging processes of different visual domains. General-scene images are usually collected by conventional cameras or public vision datasets and have diverse image sizes. Images from the Remote Sensing and Agriculture domains often cover larger spatial regions [60, 61, 57]. Images from the Microbiology domain may have higher original

resolutions. To ensure that images from different sources and resolutions can be used stably within a unified training and evaluation framework, we normalize extremely high-resolution images during dataset construction, reducing the instability caused by very large images in data loading, memory consumption, and statistical distribution.

C.3 Dataset Construction Pipeline

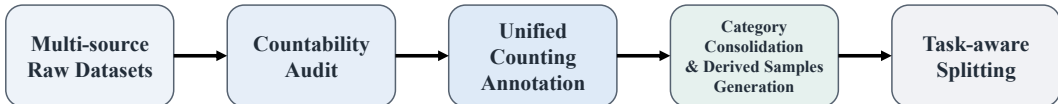


Figure 15: Overview of the CLOC construction pipeline. CLOC starts from multi-source raw datasets and sequentially undergoes countability auditing, unified counting annotation construction, category consolidation and derived-sample generation, and task-aware splitting, ultimately forming a multi-domain category-specified object counting benchmark.

Figure 15 illustrates the overall construction pipeline of CLOC. We first collect raw images and annotations from multiple public datasets to form a multi-source raw data pool covering six visual domains. Since these datasets were originally designed for different tasks, such as detection, segmentation, crowd counting, medical image analysis, remote sensing recognition, and agricultural phenotyping, directly merging them would introduce inconsistencies in annotation semantics, category granularity, and data splitting, as also observed in multi-dataset segmentation and detection benchmarks [69, 70]. Therefore, the construction process is not a simple concatenation of multiple datasets. Instead, it is organized around the category-specified counting task and proceeds through countability auditing, unified counting annotation construction, category consolidation and derived-sample generation, and task-aware splitting.

Specifically, countability auditing determines whether the original annotations satisfy the one-instance-one-count requirement, and filters or skips annotation items, categories, or images that are not suitable as counting supervision. Unified counting annotation construction addresses differences in both original annotation protocols and instance representations. It converts heterogeneous annotation protocols, including COCO-style JSON, custom JSON, XML, CSV, TXT, MAT/NumPy/HDF5, NIFTI, and mask images, into a unified counting annotation structure organized by image and indexed by category. Meanwhile, different instance representations, such as bbox, point, polygon, mask, rotated box, and label map, are converted into instance-level representations suitable for counting.

Based on this unified structure, we further organize the global category space by normalizing category names, removing categories unsuitable for counting, and constructing category groups for semantically related classes. These category groups support a stricter seen / unseen category split: seen categories refer to target categories observed during training, while unseen categories are used only in validation or testing to evaluate category generalization. The category-group constraint further prevents semantically similar categories from being split across the seen and unseen sides. We then generate training-stage derived samples using two data augmentation strategies, cropping and stitching, to supplement medium- and high-count ranges and increase spatial-layout diversity. Finally, task-aware splitting constructs the training, validation, and test sets under constraints on sample source, visual-domain proportion, and category groups, forming CLOC as a multi-domain category-specified counting benchmark for training and evaluating generalist object counting models.

C.3.1 Multi-source Raw Datasets

As the starting point of the CLOC construction pipeline, we first collect public datasets from multiple visual domains to form a multi-source raw data pool. Here, “raw datasets” refer to source data before countability auditing, unified annotation conversion, category consolidation, and derived-sample generation. To enable CLOC to cover diverse real-world scenarios, we broadly collect data from different visual sources, covering the General Scene, Remote Sensing, Histopathology, Cellular Microscopy, Agriculture, and Microbiology domains, so that the subsequent construction process is built upon a richer set of visual domains.

Table 16 lists the source datasets used in the current version and their corresponding visual domains. All source datasets used to construct CLOC are public datasets, and we use them under their original licenses and terms of use. We retain the original ownership, citation, license, and usage information

Table 16: Source datasets grouped by visual domain. The table lists the public source datasets used to construct CLOC and their corresponding visual domains.

Visual Domain	Source Datasets
General Scene	Cityscapes [62], FSC-147 [12], FSCD-LVIS [13], JHU-CROWD++ [19], NWPU-CROWD [56], Objects365 [23], ShanghaiTech [14], VOC2007 [18], Rebar Counting [71]
Remote Sensing	CARPK [27], PUCPR+ [27, 72], DIOR [73], DOTA [60, 74, 75], DroneCrowd [76], NWPU-MOC [77], NWPU-VHR-10 [78, 79, 80], RSOD [81, 82], VisDrone [24], UpCount [83], xView [61]
Histopathology	BCData [31], CellBinDB [22], CoNIC [84], EndoNuke [85], Lizard [86], MoNuSAC [25], MoNuSeg [21, 87], NuCLS [88], NuInsSeg [20]
Cellular Microscopy	BriFiSeg [89], LIVECell [63], VGG [90]
Microbiology	AGAR [58], DeepBacs [91]
Agriculture	GWHD [57, 92], MTC [32], Soybean Pod Images from UAVs [64]

for each source dataset in the released CLOC documentation; the derived CLOC benchmark does not override the restrictions imposed by the original assets. The raw data pool covers the six visual domains described above. Datasets from different visual domains were originally designed for different tasks, including object detection, crowd counting, instance segmentation, cell nucleus analysis, remote sensing object recognition, microorganism analysis, and agricultural phenotyping. Therefore, although these datasets contain certain forms of target annotations or instance information, their original annotation formats, category definitions, and instance granularities are not consistent, and they cannot be directly merged into a unified counting dataset.

C.3.2 Countability Audit and Annotation Cleaning

After collecting the multi-source raw datasets, we first determine which annotations can serve as reliable counting supervision. Different source datasets may contain special labels or regions such as *crowd*, *group*, *ignore*, *difficult*, and *NegativeROA* in their original tasks, or may contain invalid labels outside the official category space. These labels have dataset-specific meanings and cannot be simply treated as either valid or invalid counting samples. Therefore, before unified annotation conversion, we perform a countability audit according to the annotation protocol of each source dataset, and decide whether to retain, filter, mask, or skip different categories, regions, or images.



Figure 16: Examples of source-specific special annotations requiring countability audit before unified conversion. Rows show *iscrowd=1* regions in Objects365, ignored regions in VisDrone, and *NegativeROA=1* regions in MoNuSAC. Green overlays denote regular instance annotations in the original datasets, while yellow overlays highlight source-specific special regions. These annotations have dataset-specific semantics and cannot be directly converted into one-instance-one-count supervision without source-specific handling.

Figure 16 provides representative examples of such source-specific annotations. These cases motivate the source-aware auditing rules described below, where different special labels are handled according to their original dataset protocols rather than being uniformly retained or discarded.

For general-scene data. Objects365 [23] adopts COCO-style annotations and contains `iscrowd=1` annotation items. These annotations usually represent crowded regions or target groups that cannot be separated instance by instance, where a single crowd annotation may correspond to multiple real instances. If such an annotation is directly counted as one instance, the target count would be underestimated; if only other regular instances in the same image are retained, targets inside the crowd region may still be missing. Therefore, for images containing crowd annotations, we do not use the corresponding category in that image as valid counting supervision. Cityscapes [62] contains `persongroup`, `cargroup`, and `bicyclegroup` labels, which similarly denote a group of objects rather than clear instance-level annotations. Thus, when a regular instance category and its corresponding group label coexist in the same image, we ignore the annotations of that category in the image to avoid incomplete counting ground-truth.

The `distractor` label in JHU-CROWD++ [19] represents a different case. For images with `distractor = 1`, the label indicates complex scenes with no obvious crowd or only very few people, rather than instance-inseparable or incomplete annotations like `group` or `iscrowd`. Therefore, we retain JHU-CROWD++ images with `distractor = 1`, so that CLOC includes such low-count or near-negative crowd scenes.

For remote-sensing data. VisDrone [24] contains `ignored region` annotations. These regions indicate image areas ignored during evaluation, where targets are not annotated, and the specific categories inside the regions cannot be reliably determined. If other targets in the same image were directly retained, unannotated instances inside the `ignored region` would make the counting supervision uncertain. Therefore, we skip images containing `ignored region`. xView [61] has a different issue: although its official category list contains 60 valid target categories, its raw ground-truth annotations may include additional `type_ids` that are not included in the official category list. These annotations do not belong to the official valid target categories and are excluded from scoring in the official evaluation. Therefore, we treat them as invalid annotations and filter them during data conversion, so that they do not enter subsequent training, evaluation, or point / bbox statistics.

For sequence datasets with strong temporal continuity, such as DroneCrowd [76] and UpCount [83], adjacent frames are highly similar. Keeping all frames would introduce redundant near-duplicate samples and may cause potential leakage in later data splits. Therefore, we apply fixed-interval frame sampling and retain one frame every 10 frames.

For medical and cellular-microscopy data. MoNuSAC [25] annotations are stored in Aperio ImageScope XML format and may contain regions with `NegativeROA=1`. This flag corresponds to regions drawn with the Negative Pen in ImageScope, which are intended to exclude areas from analysis rather than represent nucleus instances. Therefore, regions with `NegativeROA=1` are not converted into valid counting instances; they are treated as excluded regions during annotation conversion. In NuCLS [88], the `AMBIGUOUS` class is also not treated as a stable semantic category. It usually corresponds to nucleus-like objects whose category cannot be reliably determined, whose annotation agreement is low, or for which reliable classification labels should not be provided. It is therefore different from well-defined nucleus categories such as `tumor` or `lymphocyte`. Accordingly, we do not include `AMBIGUOUS` as a valid counting category in the unified category space.

In summary, before unified annotation conversion, we first audit and clean the original annotation semantics of each data source instead of directly converting all annotations into counting supervision. Through this process, only annotations that can be stably associated with independent instances enter the subsequent data pool. Special labels or regions such as `crowd`, `group`, `ignore`, and `NegativeROA`, invalid categories, excluded regions, near-duplicate sequence frames, and categories or regions unsuitable for counting are retained, filtered, masked, or skipped according to the protocol of each dataset. This process ensures that data from different sources have as consistent a one-instance-one-count semantics as possible after entering the unified format, and reduces the impact of original annotation differences on training and evaluation.

C.3.3 Unified Counting Annotation Construction

After countability auditing and annotation cleaning, we further convert the original annotations from different sources and protocols into a unified counting annotation structure. Since these datasets were originally designed for different tasks, such as detection, segmentation, crowd counting, medical image analysis, remote sensing object recognition, and agricultural phenotyping, their original annotations differ in file format, field semantics, indexing scheme, and instance organization. To make these data trainable, evaluable, and reusable as a unified counting benchmark in CLOC, we convert all data sources into an image-organized and category-indexed counting annotation structure. In this structure, the target categories and instance annotations in each image can be read and used in a consistent manner. This unified structure also provides the basis for subsequent category consolidation, statistical analysis, derived-sample generation, and train/validation/test splitting.

1) Parsing JSON-based annotation protocols. The original datasets vary substantially in their annotation protocols. Some source datasets adopt standard COCO-style JSON annotations, such as Objects365-2020, FSCD-LVIS, and LIVECell. These annotations are typically organized around images, instance annotations, and category tables, where each instance is described by an image ID, category ID, bounding box, and related fields. Other datasets also use JSON files but do not follow the COCO schema. For example, FSC-147 [12] stores point annotations, data splits, and category information in separate files; LabelMe-style datasets such as Soybean Pod Images from UAVs [64] describe targets through shapes, shape_type, and points; Cityscapes [62] uses polygon JSON files to record object contours and category labels; and AGAR [58] adopts a customized sample-description format. Although these datasets all store annotations in JSON format, their field meanings, indexing schemes, and instance organizations are different, and therefore need to be parsed separately before being written into the unified structure.

2) Parsing non-JSON annotation protocols. Beyond JSON annotations, we also process a variety of non-JSON annotation protocols. VOC-style datasets use XML files to record the target category, bounding box, difficult flag, and related fields for each image. Medical datasets such as MoNuSeg and MoNuSAC also use XML files, but their annotations record polygon vertices rather than standard detection boxes. GWHD, NuCLS, and related datasets store annotations in CSV files or tabular formats, where target boxes, categories, or patch metadata need to be parsed from table fields. VisDrone, JHU-CROWD++, UpCount, CARPK, and PUCPR+ use plain text files to record target boxes, point annotations, or image-level labels. Some datasets come from scientific computing or medical imaging formats, such as .mat files in ShanghaiTech, DroneCrowd, NWPU-CROWD, and MTC; .npy arrays in CoNIC; HDF5 files in BCDData; .nii.gz label maps in BriFiSeg; and TIF or mask-image annotations in CellBinDB, NuInsSeg, and DeepBacs. For datasets such as VGG, annotations may even exist as point maps, where counting points need to be extracted from image pixels.

3) Unified instance representation. During conversion, different types of instance annotations are unified into two basic representations: point and bbox. For instances that already provide bounding boxes, we retain their axis-aligned bounding boxes and generate corresponding counting points from the box centers. For rotated-box annotations, we first convert each rotated box into its horizontal enclosing box, and then generate the center point from that box. For region-level annotations such as polygons, masks, or label maps, we extract the enclosing bounding box from each instance region and further generate the corresponding center point. For datasets that only provide point annotations, we retain the original points as counting supervision and leave the bounding box field empty. Through this conversion, bbox, rotated box, polygon, mask, label map, and point annotations from different sources are unified into a common instance representation for counting.

In summary, the core of unified counting annotation construction is to convert heterogeneous protocols, including COCO JSON, custom JSON, XML, CSV, TXT, MAT/NumPy/HDF5, NIfTI, and mask images, into an image-organized and category-indexed counting data structure. This structure preserves valid instance information from different sources and enables CLOC to be released and used in a unified manner. For users, it removes the need to adapt to the original annotation protocol of each source dataset separately, allowing direct training, evaluation, and statistical analysis on the unified structure.

```

{
  "35439": {
    "idx": 35439,
    "image_path": ".../objects365_v1_00000000.jpg",
    "image_from": "objects365",
    "classes": ["people", "projectors"],
    "annotation": {
      "people": {
        "point": [[61.6, 283.0], [121.3, 252.9]],
        "bbox": [[20.3, 260.2, 102.9, 305.7],
                 [102.9, 235.1, 139.7, 270.6]]
      },
      "projectors": {"point": [...], "bbox": [...]}
    },
    "status": "original",
    "modality": "General Scene"
  }
}

```

Code 1: Example of the unified counting annotation structure. This example shows how fields such as image path, source dataset, category set, instance point annotations, bounding box annotations, sample status, and visual domain are organized in the unified annotation format.

C.3.4 Category Consolidation and Derived Sample Generation

After countability auditing, annotation cleaning, and unified counting annotation construction, we further organize the category space in the global data pool and generate derived samples for training. Although different data sources have already been converted into a unified counting annotation format, they were originally designed for different tasks and annotation systems. As a result, category names, category granularity, and category hierarchies may still differ. If merged directly, the same semantic target could be split into multiple categories due to differences in capitalization, singular/plural forms, or naming conventions. The data pool may also contain categories that are not suitable as instance-level counting targets. In addition, the relationship between fine-grained subclasses and coarse-grained parent classes can affect the seen / unseen category split. Therefore, after unified annotation conversion, we further perform global category consolidation. Based on this consolidated data pool, we then generate training-stage derived samples through cropping and stitching, enriching medium- and high-count training samples and increasing the diversity of target spatial layouts.

1) Category name normalization. We normalize category names so that categories from different data sources use consistent written forms. Public datasets often adopt different naming conventions, such as differences in capitalization, singular/plural forms, spaces, underscores, and special characters. For such cases, we convert category names into normalized forms and merge clearly synonymous or near-synonymous expressions when necessary. For example, `Bottle` in `Objects365`, `bottle` in `FSCD-LVIS`, and `bottles` in `FSC-147` are unified as `bottles`; similarly, `Chair` in `Objects365`, `chair` in `FSCD-LVIS`, and `chairs` in `FSC-147` are unified as `chairs`. This process reduces category fragmentation caused by naming differences and ensures that the same target concept has a consistent category name in the global data pool.

2) Removing categories unsuitable for counting. We further remove categories that are not suitable as instance-level counting targets. Not all original categories in multi-source datasets can be stably associated with the “one instance, one count” assumption. Some categories have ambiguous boundaries, some are closer to regions or scene structures, and some may be valid for detection or recognition in their original tasks but are not suitable as category-specified counting targets. For example, in remote sensing data, categories such as `harbor`, `Expressway-Service-area`, and `overpass` usually refer to large scene structures or region-like targets, whose instance boundaries and counting units are less clear than objects such as vehicles, airplanes, or ships. These categories are therefore removed from the valid counting category space, ensuring that the retained categories can more stably support instance-level counting.

3) Semantic category group construction. We construct semantic category groups for some fine-grained categories to support the later seen / unseen split. For category consolidation in a multi-source dataset, it is not appropriate to simply promote all subclasses to their parent class. For example, `Objects365` may contain fine-grained vehicle categories such as `Formula 1 cars`, `sports cars`,

suvs, and vans. If all these categories were directly merged into cars, we would have to assume that all cars in the corresponding images are completely annotated. However, in the original detection datasets, an image may still contain other vehicle instances that are not annotated as any of these fine-grained categories. Directly merging these fine-grained labels into cars could therefore turn some unannotated car instances into missing labels, introducing false-negative noise.

To avoid this issue, we do not simply promote subclasses to parent classes. Instead, we construct category groups to capture semantic relationships among related categories. For example, cars, sports cars, Formula 1 cars, suvs, and vans are assigned to the same vehicle-related category group. Similarly, fine-grained categories related to bottles, chairs, birds, and boats are also grouped accordingly. These category groups are mainly used in later data splitting, ensuring that seen / unseen splits do not separate highly related categories across the two sides. This leads to a stricter evaluation of the model’s ability to generalize to genuinely unseen categories.

4) High-resolution image normalization. After category consolidation, some extremely high-resolution images need to be normalized in scale. Since the dataset covers multiple visual domains, image resolutions vary substantially across data sources. For example, DOTA [60, 74, 75], xView [61], and JHU-CROWD++ [19] contain high-resolution images. If such images were directly used in the unified training and visualization pipeline, they would introduce large memory, loading, and rendering costs, and would also create excessive scale differences across data sources. Therefore, extremely high-resolution images are cropped into image tiles whose long side does not exceed 2048, while the corresponding instance point and bbox annotations are updated accordingly. This step allows high-resolution images to enter the unified data pool at a more stable scale, while preserving their local target distributions and dense-scene information.

In the unified CLOC data pool, low- and medium-count samples are already well covered. To further enrich the medium- and high-count ranges during training, we generate two types of derived samples: cropped samples and stitched samples. Cropping mainly extracts local target regions from high-density images to produce medium- or high-count local training samples. Stitching mainly combines low- or medium-count image patches into higher-count training samples, thereby expanding the target-count range and increasing spatial-layout diversity.

5) Cropped sample generation. Cropped samples are generated by extracting local target regions from original images. Unlike simple random cropping, our cropping process prioritizes local windows whose target counts fall into predefined count ranges, thereby supplementing medium- and high-count ranges and local dense scenes in a targeted manner. To avoid generating a large number of highly overlapping near-duplicate crops from the same image, we impose an overlap constraint on selected windows by limiting the IoU between a new window and existing cropped regions. In the unified data pool, we use multi-scale candidate windows for local cropping, including 384×384 , 512×512 , 768×768 , 384×256 , 512×384 , and 768×512 . Cropped samples smaller than 384×384 are padded to satisfy the minimum input size.

6) Stitched sample generation. Stitched sample generation also follows a target-count-oriented strategy. We first select candidate image-patch combinations according to predefined target-count ranges, so that the stitched samples can supplement specified medium- and high-count intervals. Unlike direct random composition, the image patches used for stitching are required to come from the same visual domain and correspond to the same counting category. This avoids forcibly mixing samples with substantially different imaging styles and ensures consistency in both visual domain and category semantics. To reduce near-duplicate derived samples, we also limit the maximum reuse count of each patch during stitching.

Figure 17 shows the distribution of original samples, cropped samples, and stitched samples across different target-count ranges. The overall distribution is still mainly supported by original samples, while cropped and stitched samples provide additional coverage in medium- and high-count ranges such as 51–100, 101–200, and 201–500. In other words, derived samples are used to enhance density coverage during training based on the original data, allowing the model to observe more locally dense and high-count scenes. Cropped and stitched samples are used only in the training set and are not included in the validation or test sets. The validation and test sets are composed of original samples, ensuring that CLOC evaluation reflects counting ability under the real data distribution.

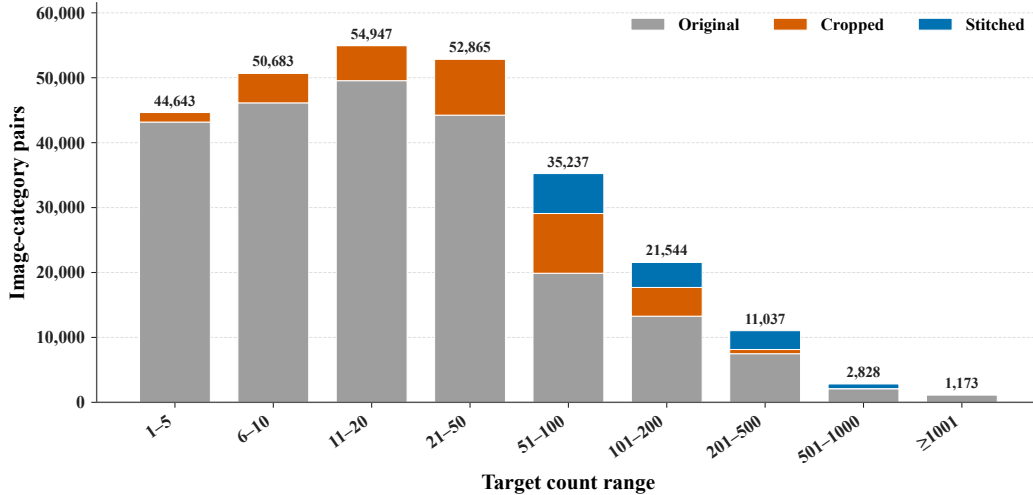


Figure 17: Distribution of original and derived samples across target-count ranges. The figure shows the contribution of original samples, cropped samples, and stitched samples to each target-count range. Cropped and stitched samples are used only in the training set.

C.3.5 Task-aware Splitting

After unified annotation construction, category consolidation, and derived-sample generation, we further construct the training, validation, and test sets. Since the data pool covers multiple visual domains, contains a large number of counting categories, and includes both original and derived samples, simple random splitting cannot satisfy the evaluation requirements of a general counting benchmark. Specifically, the split needs to avoid data leakage caused by derived samples such as cropped and stitched samples, ensure that different visual domains remain evaluable in the validation and test sets, and establish a reliable seen / unseen category-generalization setting. In particular, for visual domains with a rich category space, merely ensuring that training and test category names do not overlap is not sufficient to fully characterize category generalization. If semantically related categories are split across the seen and unseen sides, the model may still benefit from strong priors learned from neighboring categories during training, thereby weakening the strictness of unseen evaluation. Therefore, we adopt a task-aware splitting strategy that constrains the final split from three aspects: derived-sample isolation, visual-domain proportions, and semantic category groups.

1) Derived-sample isolation. To avoid evaluation leakage introduced by data augmentation, we first construct the training, validation, and test sets at the original-sample level, and then generate cropped and stitched samples only from original images in the training set. In other words, cropped and stitched samples are retained only in the training set, while the validation and test sets do not contain any samples generated by cropping or stitching. This prevents the model from seeing local crops, stitched variants, or highly similar versions of validation or test images during training, thereby reducing the risk of data leakage. Meanwhile, cropped and stitched samples are used only to enhance target density, count range, and spatial-layout diversity during training. The validation and test sets are entirely based on unaugmented original samples, so that the evaluation results reflect the model’s counting ability under the real image distribution.

2) Visual-domain proportion constraint. The split also explicitly considers the distribution of the six visual domains. To ensure stable cross-domain evaluation, the training, validation, and test sets all cover every visual domain. Under the constraints of derived-sample isolation and category splitting, we further make the image proportion of each visual domain in each split as close as possible to its proportion in the overall data pool. Compared with simple random splitting, this strategy avoids small specialized domains from being randomly weakened or omitted in the validation or test sets, allowing model performance to be jointly evaluated under different visual conditions.

3) Strict seen / unseen splitting based on category groups. For visual domains with rich category spaces, such as General Scene and Remote Sensing, we construct a seen / unseen evaluation setting to assess category generalization. Existing category-conditioned counting datasets usually construct

unseen-category evaluation by ensuring that category names do not overlap. For example, the original FSC-147 [12] split ensures that the train, validation, and test sets do not share object categories, with 89 categories used for training, 29 for validation, and 29 for testing. However, later revisions of FSC-147 [11] point out that category-name disjointness alone can still lead to semantic split overlap. For example, ambiguous or hierarchically related categories such as `kidney beans` / `red beans` and `bread rolls` / `baguette rolls` can still create strong semantic associations between training and testing [93]. This suggests that, in a large category space, seen / unseen splitting should not rely only on whether category names are identical, but should also consider semantic relationships among categories.

This issue becomes more prominent in a multi-source dataset. Different data sources often contain both coarse-grained and fine-grained categories, such as `cars`, `sports cars`, `Formula 1 cars`, `suvs`, and `vans`. If `sports cars` or `vans` appear in the training set while `cars` is placed in the test set as an unseen category, the model has in fact already observed visual concepts that are highly related to the test category during training. In this case, unseen results may reflect transfer from neighboring categories rather than true generalization to genuinely unseen semantic categories.

To mitigate this issue, we perform seen / unseen splitting based on the category groups constructed during category consolidation. Specifically, categories that are semantically close, have parent-child relationships, or can easily provide substitute supervision are assigned to the same category group, such as vehicle-related, bird-related, bottle-related, and boat-related groups. During splitting, the same category group is not divided across the seen and unseen sides. In other words, if a category group is selected as unseen, semantically related categories in that group are also excluded from the training categories or constrained accordingly; if a category group is retained as seen, its related categories will not simultaneously appear as unseen categories for generalization evaluation. In this way, the seen / unseen boundary is no longer determined only by individual category names, but is jointly constrained by semantic category groups, leading to a stricter category-generalization evaluation.

This category-group-based splitting strategy has two advantages. First, it reduces implicit leakage caused by semantically neighboring categories, making unseen evaluation closer to a genuinely unseen-category setting. Second, it preserves fine-grained categories themselves rather than simply promoting subclasses to parent classes, thereby avoiding new missing-label risks introduced by coarse-grained merging. For domains with relatively small and semantically concentrated category spaces, such as Histopathology, Cellular Microscopy, Agriculture, and Microbiology, we do not additionally construct seen / unseen category splits in the current version. These domains are mainly used to evaluate counting stability across different specialized visual domains.

The split statistics of CLOC are shown in Table 17. The training set contains 207.3K images, 586 categories, and 14.70M instances; the validation set contains 4.7K images, 77 categories, and 278.4K instances; and the test set contains 8.1K images, 78 categories, and 382.3K instances. All three splits cover the six visual domains. Meanwhile, cropped and stitched samples appear only in the training set, which prevents data leakage while supporting cross-domain counting evaluation and stricter category-generalization evaluation.

Table 17: Statistics of the training, validation, and test splits. The table reports the number of images, categories, and instances in each split.

	Train	Validation	Test
Images	207.3K	4.7K	8.1K
Categories	586	77	78
Instances	14.70M	278.4K	382.3K

C.4 Evaluation Protocol

CLOC is primarily designed for text-guided object counting evaluation. In practice, category names are used as textual queries, so each image-category pair corresponds to an image-query counting sample. Given an image and a target query, a model is required to predict the number of instances corresponding to that query in the image. For evaluation, we use MAE, RMSE, and NAE to measure counting errors.

Assume the test set contains N image-category samples. For the i -th sample, let c_i denote the ground-truth count and \hat{c}_i denote the predicted count.

MAE measures the mean absolute counting error:

$$\text{MAE} = \frac{1}{N} \sum_{i=1}^N |c_i - \hat{c}_i|.$$

MAE directly reflects the average number of targets incorrectly predicted per sample, and is the most intuitive counting-error metric.

RMSE measures the root mean squared error:

$$\text{RMSE} = \sqrt{\frac{1}{N} \sum_{i=1}^N (c_i - \hat{c}_i)^2}.$$

RMSE is more sensitive to large errors, and therefore reflects model stability on high-density scenes or difficult samples. When a model produces severe counting errors on a small number of samples, RMSE highlights such failures more clearly than MAE.

NAE measures the normalized absolute error:

$$\text{NAE} = \frac{1}{N} \sum_{i=1}^N \frac{|c_i - \hat{c}_i|}{c_i + \epsilon},$$

where $\epsilon = 10^{-6}$ is used to avoid division by zero. NAE reflects relative counting error, making errors more comparable across samples with different count scales. For example, an error of 10 targets has different severity when the ground-truth count is 20 versus 200, and NAE better captures this difference.

The basic evaluation unit of CLOC is an image-category pair, rather than the total number of targets in an entire image. If an image contains multiple target categories, each category forms an independent counting sample. This setting matches text-guided category counting, where the model needs to output the number of instances for the specified query rather than estimate the total number of all visible targets in the image.

D Discussion

This paper advances the development of generalist counting models from both the data and method perspectives. On the data side, we construct CLOC, a large-scale cross-domain object counting dataset covering six visual domains, which provides a unified benchmark for training and evaluating counting models across categories, visual domains, and density distributions. On the method side, we propose Count Anything, a generalist counting model that uses discrete instance points as its output representation. By combining region-level sparse counting, pixel-level dense counting, and complementary fusion, Count Anything achieves more stable instance-level counting under different target scales and density conditions. Overall, this work aims to move object counting beyond specialized models designed for single application scenarios, toward generalist open-world counting models that can be text-guided, spatially interpretable, and capable of cross-domain generalization. Although this work makes clear progress in this direction, several limitations remain, leaving room for further exploration in future research.

D.1 Discussion of the Method

From the method perspective, Count Anything takes an important step toward cross-domain, text-guided, instance-interpretable open-world counting. Count Anything represents counting results as a set of instance-level points, rather than a single scalar or a continuous density map. Such an output format can directly produce the final count while also providing spatial evidence for each counted instance, making the model predictions more interpretable. Furthermore, the RSC and PDC branches in the model are designed for different counting scenarios. RSC provides stable geometric anchors for large, sparse, and clearly bounded targets through region-level prediction, while PDC performs dense point prediction on high-resolution features and is better suited to recovering small, dense, and weakly bounded targets. Point-centric supervision enables the model to use both point annotations and box annotations, reducing the training difficulty introduced by heterogeneous annotations from multiple data sources. CCF further fuses the two types of predictions in a parameter-free manner during inference, reducing duplicate counts while preserving the complementarity of the two branches.

At the same time, the method also has several limitations. First, the current model still relies on a pretrained text-conditioned visual encoder, and its cross-domain generalization ability is therefore constrained to some extent by the underlying vision-language representation. When facing target category names with severe ambiguity, or certain domain-specific terms that rarely appear in the pretraining data, the model may still produce missed or false predictions. Second, the current CCF is a lightweight fusion method based on confidence scores and spatial relationships. Although it is effective and requires no additional training, it may still be difficult to fully distinguish duplicate predictions from truly adjacent instances in extremely crowded scenes, under severe occlusion, or when prediction confidence scores are unstable. This is also one of the long-standing core challenges in dense object counting. Future work may further explore more efficient dense point decoding, more adaptive cross-branch fusion strategies, and counting models that incorporate uncertainty estimation or interactive feedback. In addition, combining large language models or multimodal large language models to extend text prompts from category names to more complex natural-language descriptions, and even to enable a certain degree of cross-modal reasoning, is also a direction worth further exploration.

D.2 Discussion of the Dataset

From the data perspective, the core significance of CLOC lies in moving the long-fragmented object counting data ecosystem toward a large-scale, cross-domain, and unified generalist counting benchmark. The main contribution of CLOC is to provide a larger-scale, more cross-domain, and semantically more unified data foundation for generalist object counting. Unlike previous counting datasets that are mainly designed for a single scenario or a small number of categories, CLOC reorganizes multi-source data from six visual domains, including General Scene, Remote Sensing, Histopathology, Cellular Microscopy, Agriculture, and Microbiology, into a unified category-specified counting task. Through countability auditing, heterogeneous annotation conversion, category consolidation, derived-sample generation, and task-aware splitting, CLOC avoids, as much as possible, annotation semantic conflicts and evaluation bias introduced by directly merging multiple data sources. With about 220K images, 619 categories, and 15.356M instances, CLOC covers diverse counting scenarios ranging from low-density to extremely high-density scenes. Therefore, it can evaluate not only the counting ability of models in conventional scenarios, but also their generalization ability in cross-domain, long-tail category, and dense-target scenarios.

Nevertheless, CLOC still has room for further improvement. First, CLOC is reorganized from multiple public data sources, whose collection conditions, annotation standards, and category granularity are not fully consistent. Although we perform countability auditing and annotation cleaning, a small amount of source-inherited noise may still remain, such as imprecise boundaries, missing instances, or inconsistent category definitions. Second, the sample sizes across the six visual domains are still not fully balanced. General Scene and Remote Sensing account for relatively larger portions of the current dataset, while specialized domains such as Histopathology, Cellular Microscopy, Agriculture, and Microbiology contain fewer available counting samples. This is mainly limited by the availability of stable instance-level annotations in public specialized-domain datasets. Third, constructing large-scale counting datasets itself is still constrained by annotation difficulty. Compared with conventional detection tasks, counting images usually contain more instances, and the targets may be dense, blurry, severely occluded, or have unclear boundaries. Therefore, instance-level annotation and quality checking are more costly. This is also an important reason why existing counting datasets are usually smaller than large-scale detection datasets. Although CLOC has significantly expanded the scale of generalist counting data, there is still room for further growth compared with the largest data resources in the detection field. Future work can further extend CLOC in three directions: reducing residual multi-source annotation noise through more systematic manual review and quality control; supplementing more data from specialized visual domains and long-tail categories to reduce cross-domain sample imbalance; and improving large-scale counting annotation efficiency by combining semi-automatic annotation, active learning, weak supervision, and human verification.