Mind the Prompt: A Novel Benchmark for Prompt-based Class-Agnostic Counting
Supplementary Material

Luca Ciampi[1*] **Nicola Messina**[1*] **Matteo Pierucci**[2]
Giuseppe Amato[1] **Marco Avvenuti**[2] **Fabrizio Falchi**[1]
[1]CNR-ISTI, Pisa, Italy [2]University of Pisa, Italy

A. Derivation of Counting Precision and Recall

In this section, we provide a more detailed explanation of the derivation of Eqs. 4 and 5 from the paper, specifically the formulas for calculating *counting precision* and *counting recall* based on the inferred quantities c^{pos} and c^{neg} in the context of the mosaic test. To simplify the notation, we omit the indices i, j from all the involved quantities, as our focus is on a single mosaic.

As stated in the paper, we deal with counting rather than detection. Therefore, we do not know the exact nature of each inferred instance, *i.e.*, we cannot assign a correct/incorrect label to each different detected object. However, we can still estimate the total number of true positives (TPs), false positives (FPs), and false negatives (FNs) directly from the outputs of the counting model. We make the following assumptions:

- For the positive image (the top part of the mosaic),

$$\text{TP}^{\text{pos}} = \begin{cases} c^{\text{pos}}, & \text{if } c^{\text{pos}} < \tilde{c} \\ \tilde{c}, & \text{otherwise} \end{cases}, \tag{1}$$

  where $\tilde{c}$ is the ground truth of the positive class. Indeed, if the model predicts fewer objects than the ground truth, all the predicted objects are considered correct, and the remaining ones are FNs. Conversely, if the model predicts more objects than the ground truth, only $\tilde{c}$ objects are correct, and the remaining contribute to the FPs. This situation for FNs and FPs can be directly derived from Eq. (1). In fact, given that $c^{\text{pos}} = \text{FP}^{\text{pos}} + \text{TP}^{\text{pos}}$, it follows that

$$\text{FP}^{\text{pos}} = \begin{cases} 0, & \text{if } c^{\text{pos}} < \tilde{c} \\ c^{\text{pos}} - \tilde{c}, & \text{otherwise} \end{cases} \tag{2}$$

  and provided that $\tilde{c} = \text{FN}^{\text{pos}} + \text{TP}^{\text{pos}}$, we also have

$$\text{FN}^{\text{pos}} = \begin{cases} \tilde{c} - c^{\text{pos}}, & \text{if } c^{\text{pos}} < \tilde{c} \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

- For the negative image (the bottom part of the mosaic), the situation is simpler, given that all the contributions inferred by the model are FPs, as the TPs are identically zero, and thus also the FNs:

$$\text{TP}^{\text{neg}} = 0 \tag{4}$$

$$\text{FP}^{\text{neg}} = c^{\text{neg}} \tag{5}$$

$$\text{FN}^{\text{neg}} = 0 \tag{6}$$

With these quantities defined, we can introduce the *counting precision* and the *counting recall*, starting from their definitions in terms of TPs, FPs, and FNs.

A.1. Counting Precision

We start with the definition of precision, which is the following:

$$P = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{7}$$

Considering that the TPs and FPs are the sums of the respective contributions from the positive and negative parts of the mosaic – *i.e.*, $\text{TP} = \text{TP}^{\text{pos}} + \text{TP}^{\text{neg}}$ and $\text{FP} = \text{FP}^{\text{pos}} + \text{FP}^{\text{neg}}$ – we obtain the precision expressed in terms of the quantities computed in Eqs. (1) to (3) and (5). Substituting and simplifying, we obtain:

$$P = \begin{cases} \dfrac{c^{\text{pos}}}{c^{\text{pos}} + c^{\text{neg}}}, & \text{if } c^{\text{pos}} < \tilde{c} \\ \dfrac{\tilde{c}}{c^{\text{pos}} + c^{\text{neg}}}, & \text{otherwise} \end{cases} \tag{8}$$

which we can rewrite in a simpler manner as:

$$P = \frac{\min(c^{\text{pos}}, \tilde{c})}{c^{\text{pos}} + c^{\text{neg}}}. \tag{9}$$

[*] Corresponding authors, they contributed equally to this work. luca.ciampi@isti.cnr.it, nicola.messina@isti.cnr.it

This quantity is averaged among all the possible mosaics, which are $N(N-1)$ (for each image, there are $N-1$ possible mosaics), to obtain the final formula for the counting precision reported in the paper.

A.2. Counting Recall

The same idea used for deriving the counting precision can also be employed to compute the counting recall. The recall is defined as:

$$R = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{10}$$

Even in this case, TPs and FNs are the sums of the respective contributions from the positive and negative parts of the mosaic – *i.e.*, $\text{TP} = \text{TP}^{\text{pos}} + \text{TP}^{\text{neg}}$ and $\text{FN} = \text{FN}^{\text{pos}} + \text{FN}^{\text{neg}}$. We obtain the precision expressed in terms of quantities computed in Eqs. (1), (3), (4) and (6). Substituting and simplifying, we obtain:

$$R = \begin{cases} \frac{c^{\text{pos}}}{\tilde{c}}, & \text{if } c^{\text{pos}} < \tilde{c} \\ 1, & \text{otherwise} \end{cases} \tag{11}$$

which we can rewrite as:

$$R = \frac{\min(c^{\text{pos}}, \tilde{c})}{\tilde{c}}. \tag{12}$$

Again, this quantity is averaged in the same way as counting precision to obtain the final formula reported in the paper.

B. Derivation of Normalized Mean of Negative predictions (NMN)

NMN, as reported in the paper, is the absolute counting error computed by prompting the model with the negative classes normalized by the ground truth of the positive class. Formally, the main involved quantity computed for each image I_i prompted with the negative class P_j is given by:

$$n_{ij} = \frac{|c_{ij} - \tilde{c}^{\text{neg}}_{ij}|}{\tilde{c}_i}, \quad i \neq j \tag{13}$$

where $\tilde{c}^{\text{neg}}_{ij}$ is the ground truth corresponding to the image prompted with the negative class, which is identically zero for $i \neq j$. Therefore, the numerator simplifies from $|c_{ij} - \tilde{c}^{\text{neg}}_{ij}|$ to c_{ij} (we assume the count predicted by the model is always positive). All the N_{ij} are then averaged over all the N images, each one prompted with all the possible $N-1$ negative prompts:

$$\begin{aligned} \text{NMN} &= \frac{1}{N} \sum_{i=1}^{N} \frac{1}{N-1} \sum_{\substack{j=1 \\ j \neq i}}^{N} n_{ij} && (14) \\ &= \frac{1}{N} \sum_{i=1}^{N} \frac{1}{N-1} \sum_{\substack{j=1 \\ j \neq i}}^{N} \frac{c_{ij}}{\tilde{c}_i} && (15) \\ &= \frac{1}{N(N-1)} \sum_{i=1}^{N} \frac{1}{\tilde{c}_i} \sum_{\substack{j=1 \\ j \neq i}}^{N} c_{ij} && (16) \end{aligned}$$

which is the Eq. 2 reported in the paper.

C. DAVE Inference Details

We performed small changes to the inference code to prepare the DAVE model for our benchmark. This small update drastically improved DAVE on PrACo, unblocking its full potential.

Indeed, although DAVE has been designed to be resilient to images with multiple classes, the method assumes that it is prompted by one of the classes that are surely present in the image. In these cases, the model just considers the object class whose CLIP embedding is more similar to the provided prompt instead of allowing for zero matches based on a certain score threshold. If DAVE is prompted with a class not present in the one-class-only image, the original implementation ignores the CLIP-based proposal filtering. The outcome is catastrophic, especially for our *negative test*, as DAVE outputs the same count regardless of the input text prompt. For this reason, we modified DAVE to filter the proposals associated with the sole present cluster based on the input text. To compute the threshold to decide if the cluster proposals match the provided caption, we also fed the model with the positive class to have a CLIP upper-bound score as a reference. As in the original implementation, the proposals are kept if their CLIP score is higher than 85% of this reference CLIP score. Notice that this inference procedure would be difficult in real scenarios in which the positive class is not known a-priori. However, since the positive class can be obtained through image classification – and image classification is a well-established and solved problem in computer vision – we assume that, in real use-case scenarios, it is possible to derive a reliable positive class label using state-of-the-art image classifiers.

We also noticed that the outcome on our benchmark is very dependent on the clustering threshold τ used during the spectral clustering phase. Particularly, we observed that the original $\tau = 0.17$ was too high to correctly detect the two clusters corresponding to the two images in the mosaics. For this reason, in the main paper experiments, we set $\tau = 0.10$.

Table 1. We report the results for **DAVE** on the **test set** of FSC147, varying the clustering threshold τ (lowering it from the original 0.17 to 0.10, and modifying the inference procedure (*Mod. Inf.* column) obtained by feeding the model also with the reference positive class.

		Negative Test		Mosaic Test			Classic	
τ	Mod. Inf.	NMN ↓	PCCN ↑	CntP ↑	CntR ↑	CntF1 ↑	MAE ↓	RMSE ↓
0.17	✗	1.05	37.02	0.686	0.811	0.700	15.16	103.49
0.10	✗	1.05	37.02	0.743	0.805	0.732	15.16	103.49
0.17	✓	0.08	97.45	0.831	0.803	0.784	15.11	103.48
0.10	✓	0.08	97.62	0.843	0.799	0.790	15.23	103.53

In Tab. 1, we report an ablation study about the model's behavior (i) with and without modification to the inference strategy, and (ii) the original and changed τ parameter. As we can notice, the clustering threshold does not affect the negative test, where only one object cluster is always found. Our modification, which injects positive classes as a reference, originates a strong model from the negative test perspective, with an NMN of only 0.08. Concerning the mosaic model, the lowering of the tau threshold, together with the improved inference procedure, helps raise the counting precision and, in turn, the counting F1-score by more than 12% with respect to the original implementation.

It is interesting to notice how these hyper-parameters have no effect on the class-specific classic counting metrics (MAE and RMSE), again proving the need for benchmarks like PrACo to effectively evaluate prompt-based counting models.

D. TFPOC Density Maps Creation

TFPOC is a detection-based method that localizes objects to count using the powerful SAM model [1]. For this reason, it never really computes a density map, which is the main output interface used to prepare the predictions for the mosaic test and produce the qualitative visualization. To prepare the density maps, we simply plotted the region centers as small dots, each having an area of 1 (as is usually done for preparing ground truth density maps from dot annotations).

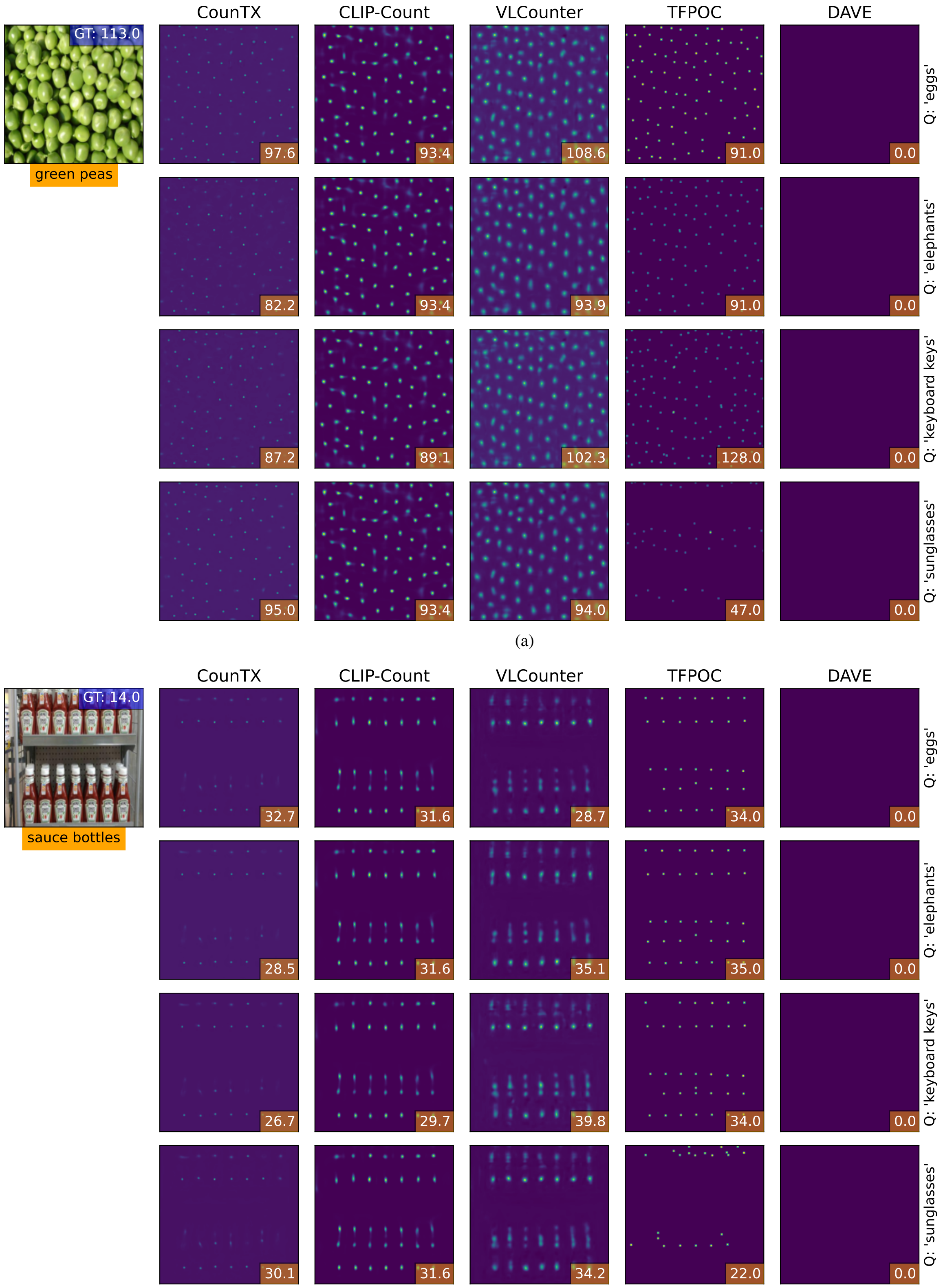
E. More Qualitative Results

In Fig. 1, we present four images provided as input to the model, each paired with different negative classes. Notably, all methods except DAVE count the negative classes, often predicting a number of instances comparable to – or even exceeding – the ground truth for the positive class. In contrast, DAVE consistently predicts zero instances, demonstrating the effectiveness of the proposed inference modification.

In Fig. 2, we present additional results from the mosaic test, illustrating how the models often struggle to count exclusively the correct class. Notably, while DAVE demonstrates strong performance in distinguishing the sole positive class from negative ones and achieves impressive results on the PrACo metrics for the mosaic test, it occasionally suffers catastrophic failures, incorrectly swapping the positive class with a negative one.

References

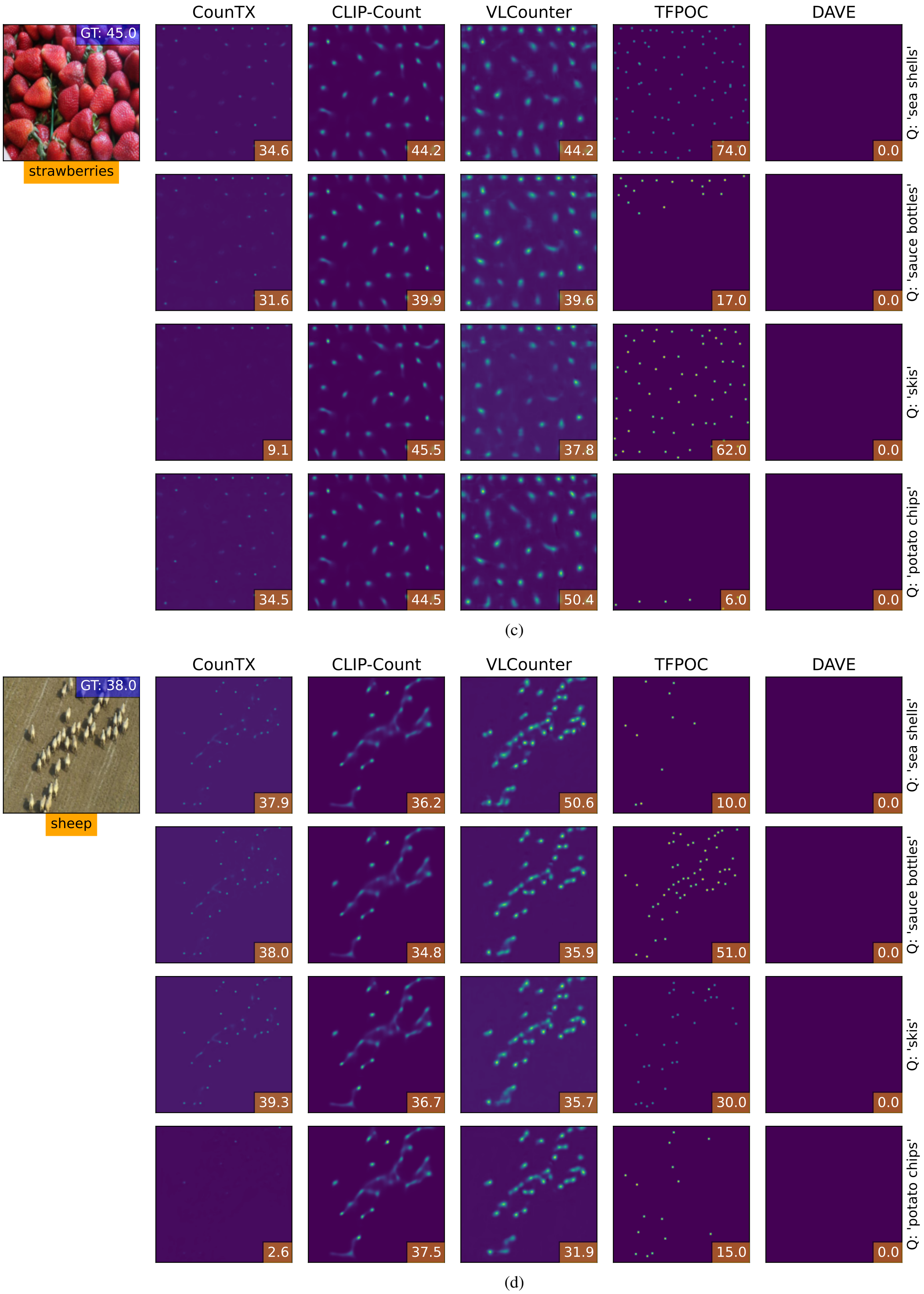
[1] Alexander Kirillov, Eric Mintun, Nikhila Ravi, Hanzi Mao, Chloe Rolland, Laura Gustafson, Tete Xiao, Spencer Whitehead, Alexander C. Berg, Wan-Yen Lo, Piotr Dollár, and Ross Girshick. Segment anything, 2023. 3



(a)

(b)

Figure 1. For each model, we report the density maps obtained when probing them with four different negative classes (*eggs, elephants, keyboard keys, sunglasses*) reported in the right-hand side of each row.



(c)

(d)

Figure 1. For each model, we report the density maps obtained when probing them with four different negative classes (*sea shells, sauce bottles, skis, potato chips*) reported in the right-hand side of each row (cont).

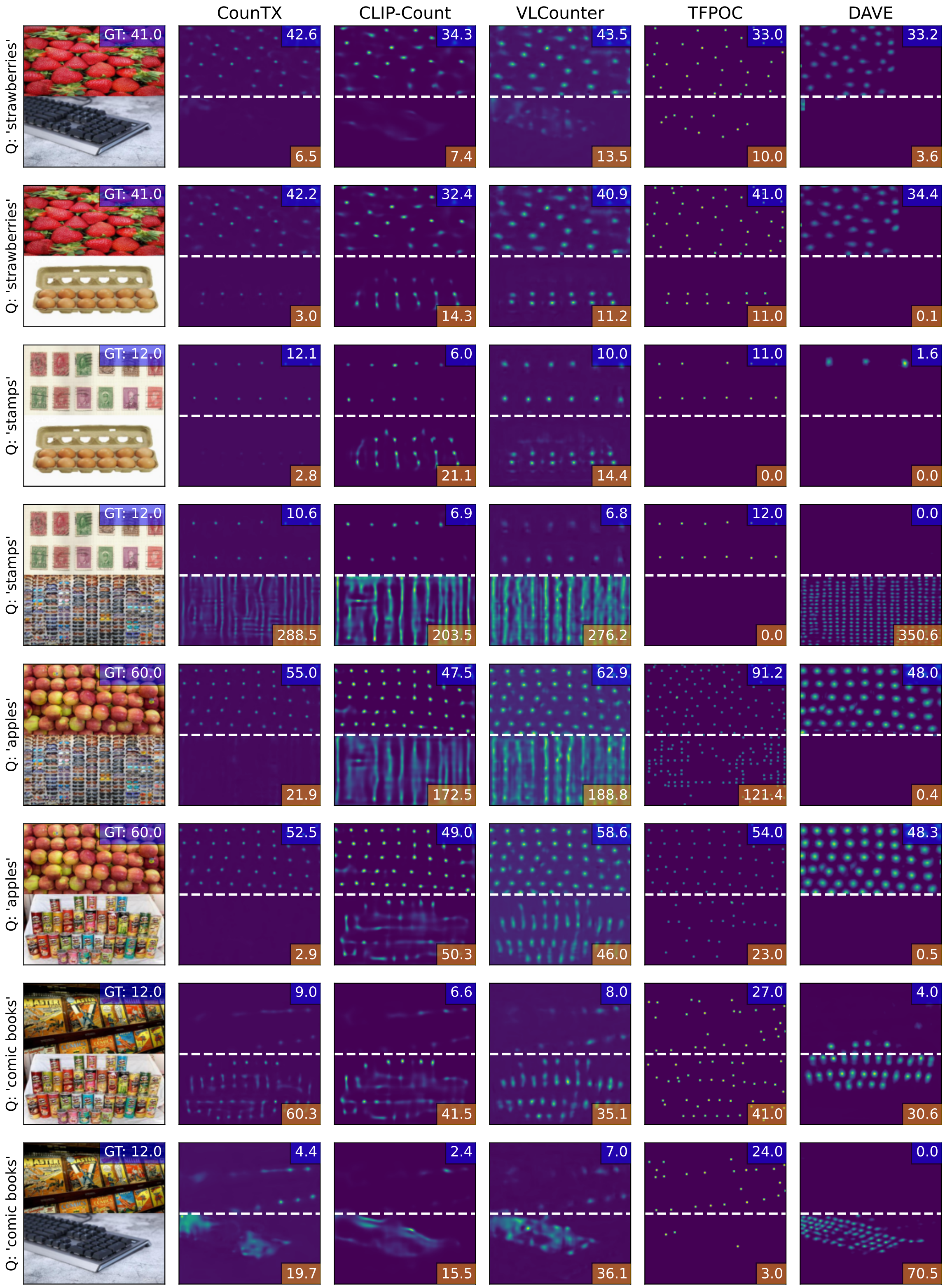


Figure 2. For each model, we report the output density maps for three different *(mosaic, input prompt)* pairs. In each figure, the count reported in the blue box is c_{ij}^{pos}, while the count reported in the red box corresponds to c_{ij}^{neg}.

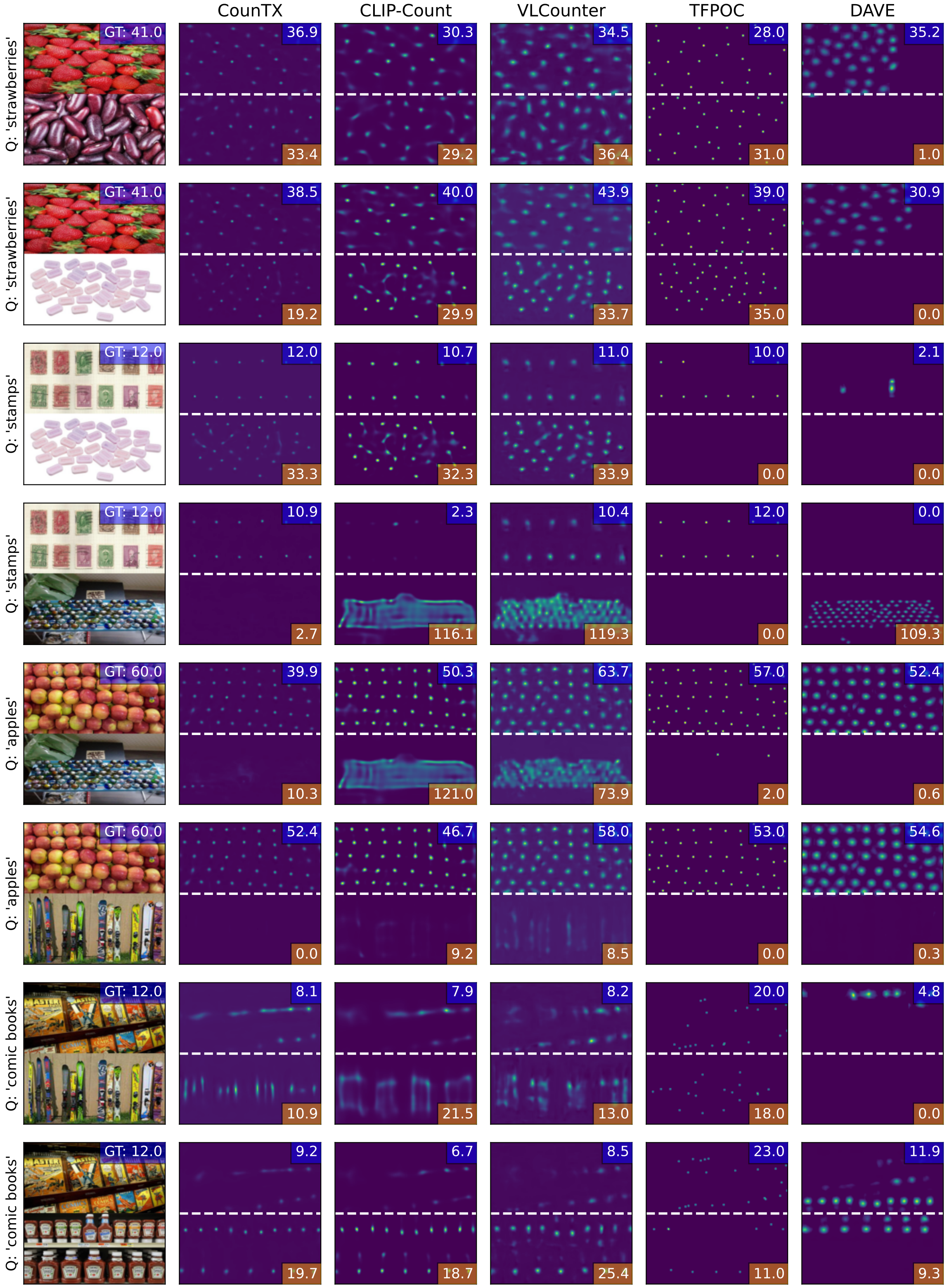


Figure 2. For each model, we report the output density maps for three different *(mosaic, input prompt)* pairs. In each figure, the count reported in the blue box is c_{ij}^{pos}, while the count reported in the red box corresponds to c_{ij}^{neg} (cont).