CountGD++: Generalized Prompting for Open-World Counting

Niki Amini-Naieni
Visual Geometry Group (VGG)
University of Oxford, UK
nikian@robots.ox.ac.uk

Andrew Zisserman
Visual Geometry Group (VGG)
University of Oxford, UK
az@robots.ox.ac.uk

Appendix

Table of Contents

A Further Quantitative Results **1**
A.1. Results on PairTally [21] 1
A.2 Results in More Prompt Settings . . . 1
A.3. Attention Ablation 2
A.4. Comparisons to MLLMs 3

B. Further Qualitative Results **3**
B.1. More CountGD++ Results 3
B.2. Example Synthetic Exemplar Images . 3

C Further Implementation Details **3**
C.1. Architecture 4
C.2. Training 4
C.3. Inference 4
C.4. Synthetic Exemplar Generation 7

D Further Dataset Details **7**

E. Further Clarifications **9**
E.1. PSeCo [32] Evaluation 9
E.2. Prompting With Negatives 9
E.3. Open-World vs. Open-Vocabulary . . 9

F. Limitations **9**
F.1. Computational Cost 9
F.2. Ambiguous Text 9

A. Further Quantitative Results

A.1. Results on PairTally [21]

In Tab. A.1, we evaluate CountGD++ on the PairTally Benchmark [21] for fine-grained object counting. Each of the 681 images in PairTally contains two object categories (i.e., class *pairs*), requiring models to distinguish and count based on subtle object differences in attributes such as shape, size, and color. CountGD++ achieves a new state-of-the-art MAE and RMSE on this benchmark. Provided with only positive prompts, CountGD++ beats CountGD [7]. Providing both positive and negative prompts improves the counting accuracy further. These results are particularly impressive as PairTally is a very challenging dataset, including high counts of mixed objects with subtle differences in shape, color, and texture. Qualitative results are shown in Fig. A.5.

Method	Prompt				PairTally	
	t^+	B^+_{int}	t^-	B^-_{int}	MAE ↓	RMSE ↓
Qwen2.5-VL [9]	✓	✗	✗	✗	59.36	✗
LLaMA-3.2 [2]	✓	✗	✗	✗	54.67	✗
GeCo [24]	✗	✓	✗	✗	50.24	✗
DAVE [25]	✗	✓	✗	✗	47.37	✗
CountGD [7]	✓	✓	✗	✗	46.67	70.85
Ours	✓	✓	✗	✗	46.41	69.52
Ours	✓	✓	✓	✓	**35.27**	**60.85**

Table A.1. Results on the **PairTally** [21] test set. ✗ in the RMSE column indicates the RMSE results were not reported in the original PairTally paper. For CountGD, the MAE was reproduced using the published PairTally code, and the RMSE was obtained using the same code. The symbols for provided prompts are: positive text (t^+), 3 positive visual exemplars from inside each image (B^+_{int}), negative text (t^-), 3 negative visual exemplars from inside each image (B^-_{int}).

A.2. Results in More Prompt Settings

While in the main paper we only had room for presenting the results on FSCD-147 [27] given text only, in Tab. A.2 we also include results given the three internal (provided) exemplars only, or both the text and the internal (provided) exemplars. In the text-only prompt setting, our approach also uses the synthetic and pseudo-exemplars, since these are generated automatically given text only.

In the text-only setting, CountGD++ is the superior method, achieving the best detection accuracy and the lowest counting RMSE among both methods that can and

Method	Prompt	FSCD-147 Test			
		Counting		Detection	
		MAE ↓	RMSE ↓	AP ↑	AP50 ↑
Text Only					
DAVE$_{prm}$	t^+	14.90	103.42	✗	✗
CountGD	t^+	12.98	98.35	✗	✗
T2ICount	t^+	11.76	97.86	✗	✗
GrREC	t^+	10.12	107.19	✗	✗
CAD-GD	t^+	10.35	86.88	✗	✗
CountSE	t^+	**7.84**	82.99	✗	✗
GDINO	t^+	54.16	157.87	11.60	17.80
OWLv2	t^+	41.83	149.82	22.84	35.76
PSeCo	t^+	16.58	129.77	37.91*	62.45*
DAVE$_{prm}$	t^+	15.52	114.10	18.50	50.24
CGD-B	t^+	15.01	118.16	30.44	61.56
Ours	t^+	8.39	**27.03**	**38.93**	**71.35**
Internal (Provided) Exemplars Only					
DAVE	B^+_{int}	8.66	**32.36**	✗	✗
CountGD	B^+_{int}	8.31	91.05	✗	✗
C-DETR	B^+_{int}	16.79	123.56	22.66	50.57
PSeCo	B^+_{int}	13.05	112.86	42.98*	73.33*
DAVE	B^+_{int}	10.45	74.51	26.81	62.82
GeCo	B^+_{int}	**7.91**	54.28	**43.42**	**75.06**
CGD-B	B^+_{int}	10.85	99.60	34.81	69.46
Ours	B^+_{int}	8.10	35.40	38.88	73.05
Both Text & Internal (Provided) Exemplars					
CountGD	t^+, B^+_{int}	**5.74**	**24.09**	✗	✗
CGD-B	t^+, B^+_{int}	10.29	96.33	36.20	72.39
Ours	t^+, B^+_{int}	7.95	29.24	**40.29**	**74.72**

Table A.2. Results on **FSCD-147** [22, 27] for image counting methods in text only, exemplar only, and multi-modal settings. Results for counting methods that do not output boxes are grayed out. * for PSeCo indicates the result was obtained using the published checkpoints and the same bounding boxes for counting and detection. The symbols for provided prompts are: positive text (t^+), the 3 manually annotated internal positive visual exemplars from FSC-147 (B^+_{int}). The abbreviations are: GroundingREC (GrREC [12]); CountGD-Box (CGD-B [5]); C-DETR (Counting-DETR [22]).

cannot output bounding boxes. Given exemplars only, CountGD++ achieves a competitive counting MAE and the lowest counting RMSE among methods that output boxes. CountGD++'s detection accuracy is significantly better than CountGD-Box's [5], and its counting accuracy is much better than CountGD's [7]. In the multi-modal setting, when both text and internal exemplars are provided, CountGD++ achieves competitive counting accuracy with CountGD, significantly better counting accuracy than CountGD-Box, and significantly better detection accuracy over CountGD-Box.

A.3. Attention Ablation

In this section, we discuss and ablate our self-attention strategy inside the Feature Enhancer. CountGD++ applies self-attention between positive visual exemplars and text that describe each other and negative visual exemplars and text that describe each other. However, negative prompts describing different classes of objects do not attend to each other, and positive and negative prompts do not attend to each other. We describe different self-attention strategies below.

[Option A]: All prompts attend to each other. In this setting, all prompt features attend to each other. This means all positive visual exemplar and text prompts attend to each other, all negative visual exemplar and text prompts attend to each other, and positive prompt features attend to negative ones. In this setting, even if negative concepts are unrelated, dependencies between them are explicitly modeled. An illustration of this strategy is shown in Fig. A.1.

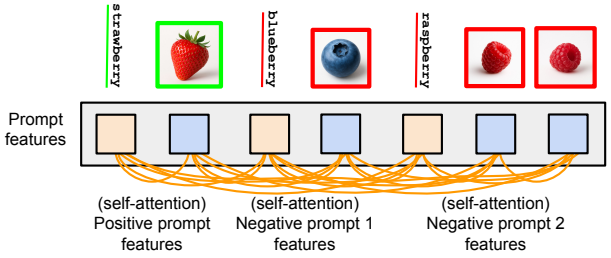

Figure A.1. **[Option A]:** self-attention occurs between all features. In the Feature Enhancer, all the visual exemplar and text prompt features attend to each other regardless of whether they are related.

[Option B]: Only prompts corresponding to the same concepts attend to each other. In this setting, all visual exemplar and text features corresponding to the same class attend to each other. Positive prompt features do not attend to negative prompt features, and negative prompt features that correspond to different classes do not attend to each other. This is the option that we choose as it prevents modeling explicit dependencies between different concepts that may be unrelated. It also does not assume a particular class is positive or negative, allowing this to be chosen after inference has occurred. This means after a single forward pass, all the objects can be counted and subsets of objects can be selected using the model's precomputed output. An illustration of this strategy is shown in Fig. A.2.

In Tab. A.3 we see that Option B, the option that we choose, results in better counting and detection accuracy on the Blood Cell Detection [8] dataset. We test on this dataset because it includes both positive and negative visual and textual prompts.

Attention Strategy	Blood Cell Detection			
	Counting		Detection	
	MAE ↓	RMSE ↓	AP ↑	AP50 ↑
Option A	2.03	3.13	0.40	0.66
Option B	**1.52**	**2.42**	**0.54**	**0.80**

Table A.3. Ablation study on the **Blood Cell Detection** [8] test set given both positive and negative text and positive and negative external exemplars. In Option A, self-attention is applied between all prompt features in the Feature Enhancer. In Option B, self-attention is only applied between prompt features of the same class.

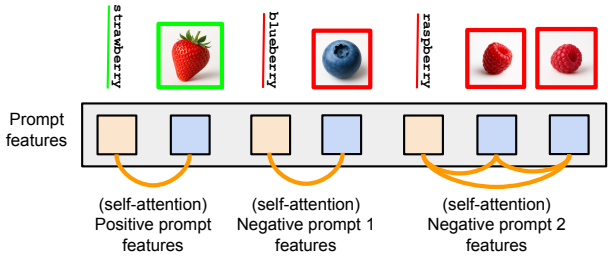

Figure A.2. **[Option B]:** self-attention only occurs between related prompt features. In the Feature Enhancer, corresponding visual exemplar and text features self-attend to each other but not to other visual exemplar and text features. Negative prompts do not attend to each other if they describe different classes.

A.4. Comparisons to MLLMs

In this section, we compare CountGD++ to much larger Multi-modal LLMs (MLLMs) trained on vast quantities of data. We evaluate Gemini-2.5 [13] and Molmo [14] on ShanghaiTech Part A [31] and FSC-147 [27] Test and compare their results to ours. For ShanghaiTech, the prompt "Count the humans in this image. Return only a number." is used. For FSC-147, the prompt "Count each $class\_name$ and return the total count. Please only provide a single number representing the count. Please be as accurate as possible." is used. When Gemini 2.5 or Molmo refuses to count, we set the predicted count to 0. Molmo sometimes returns a range instead of a single number, and in these cases we take the midpoint of the range as the estimated count. Despite attempts to improve the prompts for Gemini-2.5 and Molmo, their performance is still significantly worse than CountGD++'s. These results are presented in Tab. A.4 and Fig. A.3. This shows specialized counting models still surpass modern MLLMs in counting abilities. We use the Gemini API and model option gemini-2.5-flash-image and the official Molmo GitHub repository and model option Molmo-7B-D to run these experiments.

Method	ShanghaiTech Test Part A		FSC-147 Test	
	MAE ↓	RMSE ↓	MAE ↓	RMSE ↓
Molmo	7.26×10^8	9.47×10^8	40.41	153.29
Gemini-2.5	517.17	1364.40	43.18	127.02
Ours	**116.0**	**234.0**	**8.39**	**27.03**

Table A.4. Comparison with MLLMs. Counting results on Part A of the **ShanghaiTech** [31] crowd counting dataset and the **FSC-147** [27] dataset given positive text only. Molmo and Gemini-2.5 are compared to CountGD++. For ShanghaiTech, CountGD++ uses pseudo exemplars, and for FSC-147, it uses both pseudo- and synthetic exemplars generated from only the text. Molmo performs very poorly on ShanghaiTech compared to both Gemini-2.5 and CountGD++, since it has not been trained to count over 40 objects, and ShanghaiTech (A) only has very dense images containing 66+ humans to count. In many cases, Molmo outputs the nonsensical response *1234567890*. Given the maximum number of humans in an image is 2256, such responses increase the average error significantly.

B. Further Qualitative Results

Here we include additional qualitative results from the different aspects of our approach.

B.1. More CountGD++ Results

In Fig. A.4, we include results from CountGD++ applied to different test images in various prompt settings. CountGD++ is able to count in dense scenes (a, b, i), microscopic out-of-domain images (c), and given only a single synthetic (d) or real (f) exemplar. It can also count given only positive and negative text (g) and differentiate between mixed objects (h).

B.2. Example Synthetic Exemplar Images

In Fig. A.6, we include example synthetic exemplar images generated by GPT-5 [23] in our pipeline for generating synthetic exemplars. Notice how the synthetic exemplar images generally match the style of the input images. This helps CountGD++ match the target object in the input image visually given the synthetic exemplar extracted from the synthetic exemplar image. Also notice that the synthetic images only include one instance. This makes it easier for CountGD++ to crop out the target object to produce the synthetic exemplar.

C. Further Implementation Details

Here we discuss further implementation details about the CountGD++ architecture, training, and inference procedures as well our our synthetic exemplar generation pipeline.

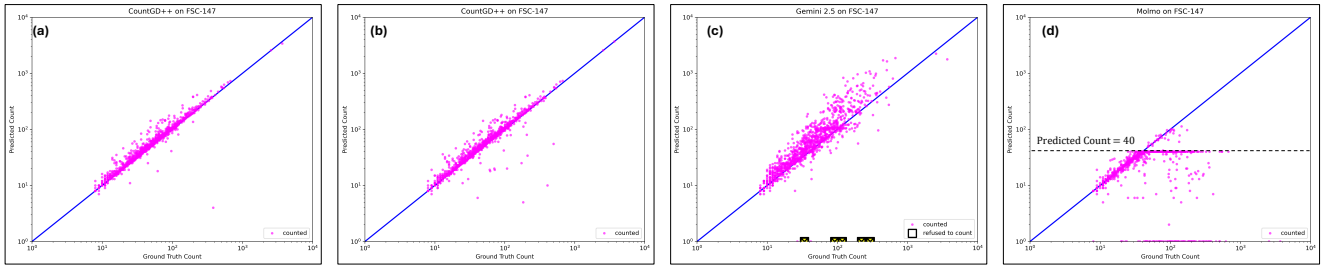

Figure A.3. COUNTGD++ compared to MLLMs on FSC-147 Test. (a) COUNTGD++ is given both text and the three manually annotated exemplars inside each image; (b) COUNTGD++ is tested in the case where only text is available while leveraging the pseudo- and synthetic exemplars obtained using only this text; (c) Gemini 2.5 [13] is tested on FSC-147 with the prompt "Count each $class\_name$ and return the total count. Please only provide a single number representing the count. Please be as accurate as possible." where $class\_name$ is replaced with the singular form of the class name. Gemini sometimes refuses to count, insisting "its [my] current capabilities do not allow it [me] to analyze images in that specific way". These cases are indicated by a box on the x-axis; (d) Molmo [14] is tested on FSC-147 with a similar prompt. Molmo counts very poorly after the ground truth count hits 40 as, to avoid memory errors, the model was not trained on data with more than 40 objects to count.

C.1. Architecture

Text Encoder. The text encoder allows for the encoding of multiple object classes in a single forward pass by using a period to distinguish between different concepts. Placing the positive text at the front is the more intuitive choice than placing the positive text at the end. This is because our framework allows for variable numbers of negative concepts, meaning input prompts grow from left to right, as English sentences do. Given the input text prompt containing the positive and negative texts, the text encoder outputs text tokens as 256-dimensional vectors. Notably, the number of vectors representing the texts is determined by the tokenizer, and a single word may correspond to multiple vectors.

Feature Enhancer. Before being passed to the Feature Enhancer, the prompt features are first rearranged, so that the positive exemplars follow immediately after the positive text, and the negative exemplars follow immediately after their associated negative text. The ordering for the positives follows directly from [7], and the ordering for the negatives mirrors this.

C.2. Training

To enable specifying negative prompts, we first augment the training set of CountGD-Box [5] with mosaicked images. Examples are shown in Fig. A.7. We also modify the input prompts of CountGD-Box [5]. During training, like CountGD-Box, we input a text prompt containing all the training classes, with each class separated by a "." An example is "alcohol bottle. baguette roll. ball. banana" assuming the training set only has these four classes. In practice, the FSC-147 [27] training set we use has 89 categories. For CountGD-Box, only one of the object categories appears in the image at a time, and visual exemplar prompts are only provided for this category. Different from this, we also add visual exemplar prompts for the other categories that appear in the image mosaics, now that they are available.

The coefficients on the loss terms λ_{loc}, λ_{GIoU}, and λ_{cls} are set to 5, 2, 2 respectively. These hyperparameters are borrowed directly from CountGD-Box [5] with no further tuning. COUNTGD++ is trained on FSC-147 [27] with 1000 of our synthetic mosaic images added. The mosaics are constructed by sampling and combining training images uniformly at random using the method in [17]. The image and text encoders are frozen during training, while the Feature Enhancer and Cross-Modality Interaction are fine-tuned. To improve the model's ability to count with text only, exemplars only, or both, during training we drop the exemplars 10% of the time and, if not dropping the exemplars, drop the text 10% of the time.

C.3. Inference

At inference, we apply an adaptive cropping procedure to count objects in dense scenes. This is necessary since COUNTGD++ only has 900 object queries and, thus, can only count up to 900 objects in a single forward pass. To address this, when the count is close to 900, we crop the image into smaller pieces, obtain counts and bounding boxes for each piece individually, and then combine the individual results into a final set of bounding boxes and a final count.

This procedure will now be discussed in detail. When at least 800 objects are counted, the adaptive cropping procedure is activated. Since the model outputs bounding boxes, we can use these to determine the crop height and width to limit the number of objects in each crop. Specifically, we first obtain a set of bounding boxes from the model's output given the whole uncropped image. The minimum object height and width are determined by taking the mini-

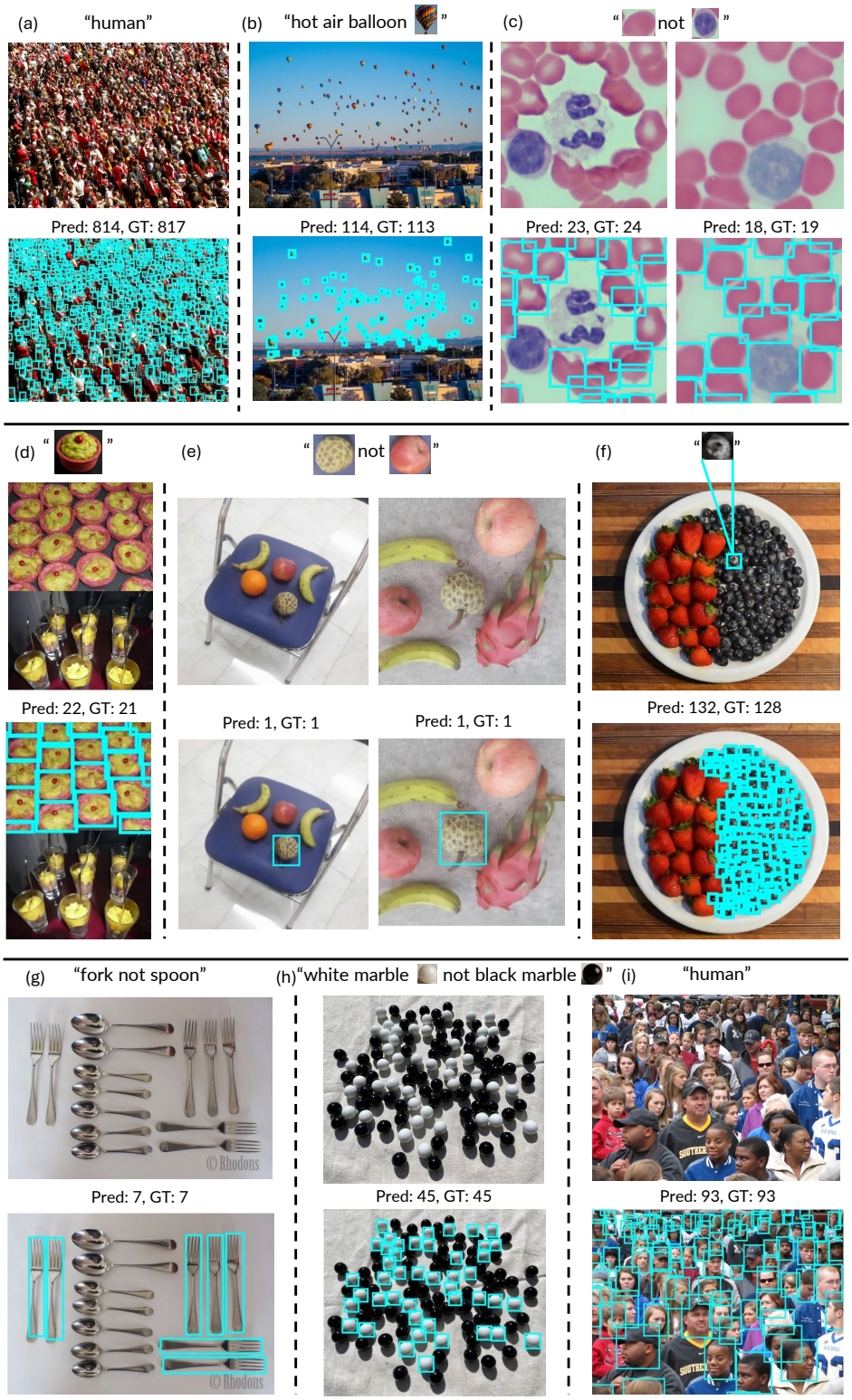

Figure A.4. Counting results on images from the test sets and the web. (a), (i) **ShanghaiTech**: positive text and pseudo-exemplars are used to count in dense crowds; (b) **FSCD-147**: positive text, synthetic, and pseudo-exemplars are used. The synthetic exemplar is at the top of the image in the quotes; (c) **Blood Cell Detection**: positive and negative external exemplars from another image are used; (d) **FSCD-147**: two images of different finger foods are mosaicked together, and positive synthetic and pseudo-exemplars are used; (e) **OmniCount**: positive and negative external exemplars are used; (f) **FSCD-147**: one positive manually annotated exemplar is used; (g) **web**: positive and negative text are used; (h) **web**: positive and negative text and positive and negative synthetic exemplars are used.

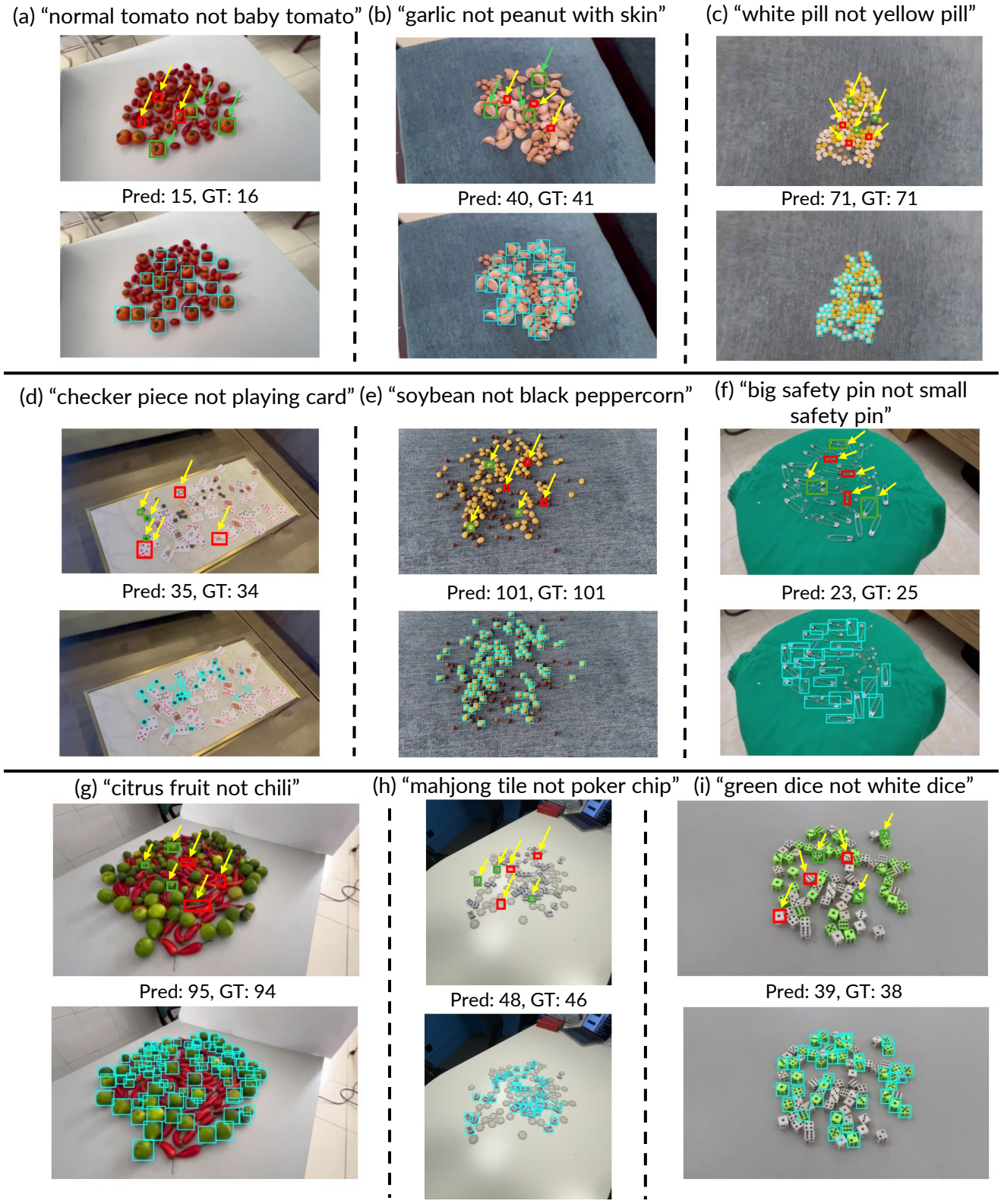

Figure A.5. Counting results on images from the **PairTally** test set given 3 positive and 3 negative exemplars from inside the image and positive and negative text. The positive and negative text are indicated at the top of each image, the positive exemplars are boxed in green and pointed to with the yellow arrows, and the negative exemplars are boxed in red and pointed to with the yellow arrows. CountGD++'s outputs are shown below the input image and prompts for each example. CountGD++ is able to distinguish between objects with different colors ((c), (e), (g), (i)), the same color but different shapes ((a), (b), (h)), and different sizes ((d), (f)).

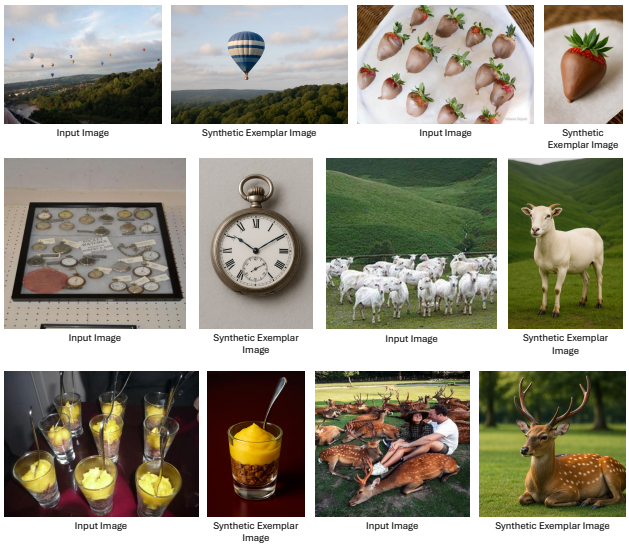

Figure A.6. Synthetic exemplar images from our pipeline for generating synthetic exemplars.

Figure A.7. Example synthetic mosaic training images constructed from FSC-147 [27] images. Each image tile of the mosaic provides annotations for a different object category.These mosaic images provide training samples where both positive and negative classes can be specified in the prompt.

mum height and width of the output boxes. The crop width is then set to $25 \times min\_obj\_width$, and the crop height is set to $25 \times min\_obj\_height$. This approximately ensures that at most 625 objects appear in each crop. The factor 25 is chosen as it balances ensuring not too many objects appear in the crop and computational efficiency. A factor too high would risk nearing the model's 900-query limit. A factor too low would require running inference over a high number of crops. The image is cropped into pieces without any overlapping regions.

Because in dense scenes the objects appear small, they can be difficult and too blurry for the model to pick out in the crops. To address this, we apply super-resolution to upscale the image crops by a factor of 4 with the Standard AI Image Upscaler from [3]. This method is chosen because it preserves the count and locations of the objects in the crops, so bounding boxes and counts can be obtained. The final set of bounding boxes is the union of the bounding boxes from the crops and the final count is the sum of the counts for the crops. In Fig. A.8 we show examples of the original model predictions, and the predictions after applying our adaptive cropping procedure. In Fig. A.9 we show examples of the original crops and the same crops enhanced with super-resolution.

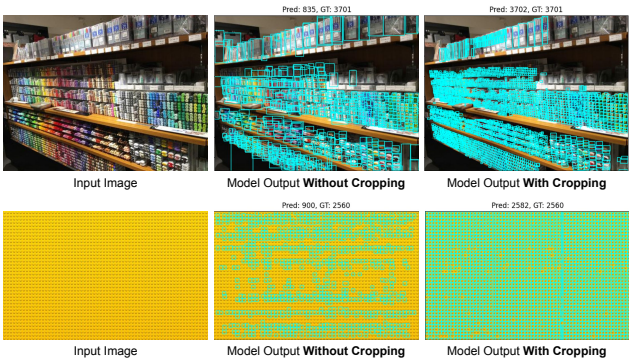

Figure A.8. Adaptive cropping improves high-count detection. COUNTGD++ can output at most 900 queries per forward pass, so extremely dense scenes cause severe under-counting. Without cropping (middle column), the model either merges many nearby objects (markers in top row) or misses large numbers of instances (yellow lego studs in bottom row). With adaptive cropping (right column), each crop remains within the model's capacity, allowing it to detect far more objects. As a result, more instances are picked up and bounding boxes less frequently merge multiple objects.

C.4. Synthetic Exemplar Generation

Here we include the prompt template provided to GPT-5 [23] for generating the synthetic exemplar images. For almost all the classes in FSC-147 [27] Test, we use the template "generate an image of a single $class\_name$ in the reference image. Please make the instance of the $class\_name$ match the $class_name$s in the reference image as closely as possible." where $class\_name$ is replaced with the class name. For the "stamp" and "comic book" classes, we use a modified version of this prompt to avoid copyright issues that trigger GPT-5's safety guardrails. The modified prompt template is "generate an image of a $class\_name$ in the same style as the ones in the reference image." Example synthetic exemplar images are included in Fig. A.6. The synthetic exemplar images generated for FSC-147 Test will be publicly released.

D. Further Dataset Details

Here we include detailed information about our different datasets. We also discuss why certain datasets were omitted.

FSCD-147 [22, 27]. FSCD-147, adds bounding boxes to the validation and test sets of FSC-147, the standard dataset

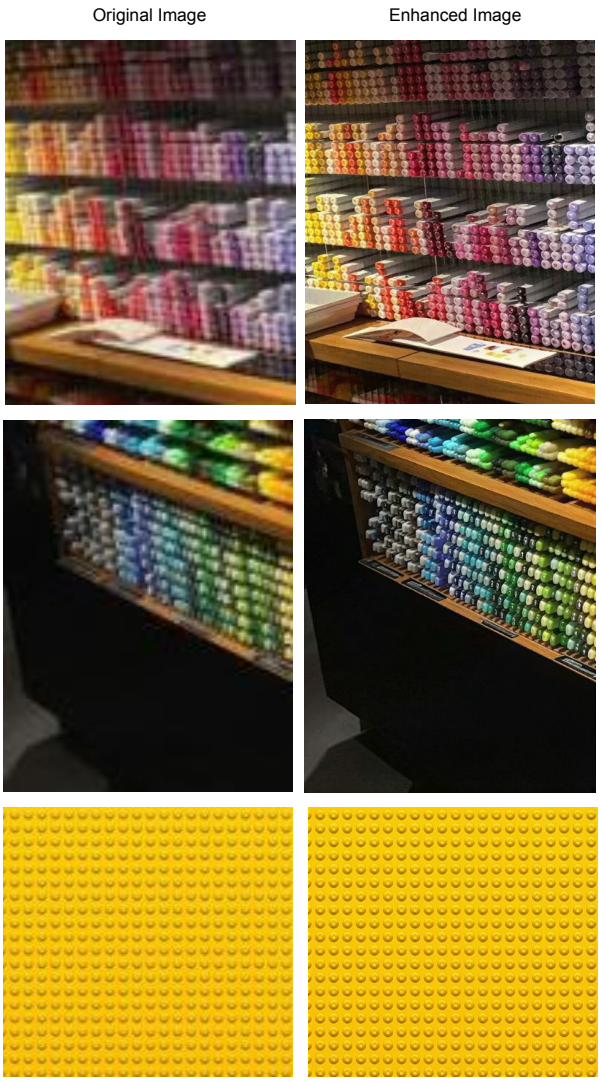

Figure A.9. Super-resolution enhances crops for adaptive cropping. In our adaptive cropping procedure, we apply the super-resolution method in [3] to the cropped images to help CountGD++ pick out the objects. Note how the super-resolution does not change the count or locations of the objects.

for open-world counting, containing 6135 images with 89 classes in the training set, 29 classes in the validation set, and 29 classes in the test set. The classes in the different sets are disjoint. Each image is annotated with 3 visual exemplars. Instances of only a single object class are labeled per image, and each image has 7-3731 objects. Bounding boxes are provided for the validation and test sets.

PrACo [11]. PrACo is a counting benchmark constructed from images in FSCD-147. It introduces the Negative Label Test to evaluate counting models when the target object is not in the image and the Mosaic Test to evaluate counting models in the multi-class setting.

ShanghaiTech [31]. ShanghaiTech is a crowd counting dataset composed of Part A, with 182 test images containing 66-2256 humans per image, and Part B, with 316 images containing 9-539 humans per image. Each human is annotated with a dot.

Blood Cell Detection [8]. This dataset contains exhaustive bounding box annotations for red and white blood cells in 100 images from a peripheral blood smear taken from a light microscope. A peripheral blood smear is a technique for microscopic blood cell examination and can aid in medical diagnosis. The images contain 11-33 cells each.

OmniCount (Fruits) [20]. This is the Fruits test set of the OmniCount-191 benchmark. It contains 303 images with 8 different fruits annotated with bounding boxes. Each image contains 3-6 fruits. The other test sets of OmniCount-191 were omitted for reasons detailed in Sec. D.

VideoCount (Crystals) [5]. This is the Science-Count (Crystals) test set of the VideoCount benchmark. It contains 10 videos of 10-154 crystals rapidly forming from liquid metal alloys in x-ray videos. The number of unique crystals in each video is annotated. We choose this test set of VideoCount, since the crystals significantly change size and structure over time, demonstrating the benefit of dynamic pseudo-exemplars.

PairTally [21]. The PairTally dataset contains 681 images that test a counting model's ability to distinguish between different objects within an image. The dataset includes cases where there are only subtle differences in the the shape, color, texture, and size of different objects. There are usually many instances of each object type mixed together and placed on a surface. The camera angle varies and sometimes introduces challenging perspective effects. Most images contain many instances, with over 150 images with 200+ total instances. PairTally has been shown to be a very challenging dataset for counting models. The metadata and ground truth annotations include text prompts, 3 exemplars, and center points for all the objects to be counted.

Omission of the Rest of OmniCount-191 [20] . At the time of our experiments, the publicly released version of OmniCount-191 exhibited several issues that prevented reliable evaluation on most of its subsets. We observed that ground-truth boxes for some classes (e.g., birds) were missing, aspect ratios for other classes (e.g., pets) were distorted, and in some cases objects that should be counted as distinct units were grouped into a single box (e.g., multiple

houses annotated as one instance). In contrast, the Fruits subset contained accurate annotations and included uncommon fruit categories (e.g., sugar apple) that provide valuable test cases for open-world counting. For these reasons, we restricted our evaluation to the Fruits subset.

E. Further Clarifications

E.1. PSeCo [32] Evaluation

The evaluation protocol used in PSeCo differs from the standard one in the counting literature [5, 22, 24, 25]. In PSeCo, the bounding boxes used for counting are not the same as the ones used for detection, whereas in the standard protocol a single set of boxes is used for both tasks. This discrepancy has also been noted by prior work [24]. To ensure consistent comparison across methods, in Tab. 1 of the main paper and Tab. A.2 of the supplementary we report PSeCo's results recomputed using the standard evaluation protocol, applying the same set of boxes for both counting and detection for all methods.

E.2. Prompting With Negatives

Beyond the counting literature, the use of negatives in related fields is more prominent. For detecting in out-of-distribution images, NegPrompt [16] learns negative prompts using positive class labels. Unlike our approach, the negative prompts cannot be provided explicitly at inference and are instead learned implicitly from the positive prompts. In segmentation and tracking, SAM 2 [28] allows users to specify *negative clicks* identifying regions that should not be segmented at inference. Similarly, in retrieval, users can provide negative prompts as feedback to refine model predictions [29, 30]. Negative prompting has also been widely explored in text-to-image generation methods such as Stable Diffusion [10]; however, despite its success there, understanding explicit negation remains challenging for VLMs like CLIP [4, 26].

E.3. Open-World vs. Open-Vocabulary

Here we make an important note about terminology. We notice that in the object counting literature [6, 7, 17], *open-world counting* refers to the task of counting instances of an object class specified at test time via textual or visual prompts. These counting models are *open-world* because they generalize beyond a fixed vocabulary, as the object of interest may belong to a class not seen during training. However, crucially, the category is still explicitly provided as input at inference time, either as text or exemplar. We adopt this definition of *open-world* in our work, since CountGD++ builds on counting literature.

However, this usage of *open-world* differs from that in some earlier literature. The origins of the formal definition of 'open world' are from [1]. This paper defines an open-world recognition system as one that recognizes instances of both *known* and *unknown* classes, marks the unknown objects as 'unknown,' obtains class labels for these unknown objects, and incrementally learns from these instances such that they become 'known.' Similarly, early works in open-world object detection [15] aim to discover and learn novel object categories without labels at first, marking them as 'unknown,' and incrementally learning them later. However, more recent works in detection [18, 19] use the term 'open-world' to describe a detection model that can detect objects unseen during training by accepting textual prompts (i.e., labels) describing the object. Importantly, this means that the labels of the unknown objects must be provided for the model to detect them, which differs from the original formal definition of 'open world' in [1]. The OWL paper [19] uses the terms 'open-world' and 'open-vocabulary interchangeably to describe this prompt-based setting as we do. In fact 'OWL' stands for 'Open-World Localization.' Like OWL, in our case, the category is always specified by the user, and the challenge lies in handling domain shifts, visual diversity, and lack of training-time exposure to the category.

F. Limitations

In this section, we discuss two limitations of our approach.

F.1. Computational Cost

The pseudo-exemplars require two forward passes of the model resulting in an inference time that is double the inference time of CountGD [7]. However, this only adds 0.21s/image while significantly improving accuracy; total runtime remains fast (0.42s/image). In comparison, larger and more computationally expensive models such as Gemini (thinking) take 12s to count 145 strawberries in a single image while being less accurate. Inference time increases for CountGD++ when synthetic exemplars, adaptive cropping, and super-resolution are added.

F.2. Ambiguous Text

We intentionally pick high confidence boxes as pseudo-exemplars as these tend to be correct detections that improve the performance of the model on the second forward pass. Low confidence boxes can sometimes be false positives from errors in the first forward pass, which are often removed after the high confidence boxes are used. However, as with any text-conditioned method, incorrect or ambiguous text can degrade performance. For example, in Fig 6b of the main paper, using the text 'apple' to pick out the sugar apples is ambiguous, since red apples also appear in the image. Because red apples are likely more familiar to the model, the most confident pseudo-exemplars picked in the first forward pass will be from the red apples, resulting in an incorrect count after the second forward pass. In

Fig. A.8 of the supplementary, the incorrect text 'lego' will result in a count of 0 in some of the crops, while the correct text 'yellow lego stud' will allow the model to pick out the yellow lego studs correctly. More descriptive text often improves performance in general.

References

[1] Bendale Abhijit and Boult Terrance. Towards open world recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015. 9

[2] Meta AI. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. 1

[3] Algora Ventures Ltd. Bubbi Image Enhancer (Image Upscaler). https://www.bubbi.app/tools/image-upscaler, 2025. Accessed: 2025-11-10. 7, 8

[4] Kumail Alhamoud, Shaden Alshammari, Yonglong Tian, Guohao Li, Philip H.S. Torr, Yoon Kim, and Marzyeh Ghassemi. Vision-language models do not understand negation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2025. 9

[5] Niki Amini-Naieni and Andrew Zisserman. Open-world object counting in videos. *arXiv preprint arXiv:2506.15368*, 2025. 2, 4, 8, 9

[6] Niki Amini-Naieni, Kiana Amini-Naieni, Tengda Han, and Andrew Zisserman. Open-world text-specified object counting. In *Proceedings of the British Machine Vision Conference*, 2023. 9

[7] Niki Amini-Naieni, Tengda Han, and Andrew Zisserman. Countgd: Multi-modal open-world counting. In *Advances in Neural Information Processing Systems*, 2024. 1, 2, 4, 9

[8] Abdüssamet Aslan. Blood cell detection dataset. https://www.kaggle.com/datasets/draaslan/blood-cell-detection-dataset, 2024. Kaggle Dataset. 2, 3, 8

[9] Shuai Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Sibo Song, Kai Dang, Peng Wang, Shijie Wang, Jun Tang, Humen Zhong, Yuanzhi Zhu, Mingkun Yang, Zhaohai Li, Jianqiang Wan, Pengfei Wang, Wei Ding, Zheren Fu, Yiheng Xu, Jiabo Ye, Xi Zhang, Tianbao Xie, Zesen Cheng, Hang Zhang, Zhibo Yang, Haiyang Xu, and Junyang Lin. Qwen2.5-vl technical report. *arXiv preprint arXiv:2502.13923*, 2025. 1

[10] Yuanhao Ban, Ruochen Wang, Tianyi Zhou, Minhao Cheng, Boqing Gong, and Cho-Jui Hsieh. Understanding the impact of negative prompts: When and how do they take effect? *arXiv preprint arXiv:2406.02965*, 2024. 9

[11] Luca Ciampi, Nicola Messina, Matteo Pierucci, Giuseppe Amato, Marco Avvenuti, and Fabrizio Falchi. Mind the prompt: A novel benchmark for prompt-based class-agnostic counting. In *Proceedings of the IEEE Workshop on Applications of Computer Vision*, 2025. 8

[12] Siyang Dai, Jun Liu, and Ngai-Man Cheung. Referring expression counting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2024. 2

[13] Google DeepMind and Google Research. Gemini 2.5. https://ai.google.dev/gemini, 2025. 3, 4

[14] Matt Deitke, Christopher Clark, Sangho Lee, Rohun Tripathi, Yue Yang, Jae Sung Park, Mohammadreza Salehi, Niklas Muennighoff, Kyle Lo, Luca Soldaini, Jiasen Lu, Taira Anderson, Erin Bransom, Kiana Ehsani, Huong Ngo, YenSung Chen, Ajay Patel, Mark Yatskar, Chris Callison-Burch, Andrew Head, Rose Hendrix, Favyen Bastani, Eli VanderBilt, Nathan Lambert, Yvonne Chou, Arnavi Chheda, Jenna Sparks, Sam Skjonsberg, Michael Schmitz, Aaron Sarnat, Byron Bischoff, Pete Walsh, Chris Newell, Piper Wolters, Tanmay Gupta, Kuo-Hao Zeng, Jon Borchardt, Dirk Groeneveld, Crystal Nam, Sophie Lebrecht, Caitlin Wittlif, Carissa Schoenick, Oscar Michel, Ranjay Krishna, Luca Weihs, Noah A. Smith, Hannaneh Hajishirzi, Ross Girshick, Ali Farhadi, and Aniruddha Kembhavi. Molmo and pixmo: Open weights and open data for state-of-the-art vision-language models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2025. 3, 4

[15] K J Joseph, Salman Khan, Fahad Shahbaz Khan, and Vineeth N Balasubramanian. Towards open world object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 9

[16] Tianqi Li, Guansong Pang, Xiao Bai, Wenjun Miao, and Jin Zheng. Learning transferable negative prompts for out-of-distribution detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2024. 9

[17] Chang Liu, Yujie Zhong, Andrew Zisserman, and Weidi Xie. Countr: Transformer-based generalised visual counting. In *Proceedings of the British Machine Vision Conference*, 2022. 4, 9

[18] Matthias Minderer, Alexey Gritsenko, and Neil Houlsby. Scaling open-vocabulary object detection. In *Advances in Neural Information Processing Systems*, 2023. 9

[19] Matthias Minderer, Alexey Gritsenko, Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy, Anurag Arnab, Mostafa Dehghani, Zhuoran Shen, Xiao Wang, Xiaohua Zhai, Thomas Kipf, and Neil Houlsby. Simple open-vocabulary object detection with vision transformers. In *Proceedings of the European Conference on Computer Vision*, 2024. 9

[20] Anindya Mondal, Sauradip Nag, Xiatian Zhu, and Anjan Dutta. Omnicount: Multi-label object counting with semantic-geometric priors. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2025. 8

[21] Gia Khanh Nguyen, Yifeng Huang, and Minh Hoai. Can current ai models count what we mean, not what they see? a benchmark and systematic evaluation. In *Digital Image Computing: Techniques and Applications (DICTA)*, 2025. 1, 8

[22] Thanh Nguyen, Chau Pham, Khoi Nguyen, and Minh Hoai. Few-shot object counting and detection. In *Proceedings of the European Conference on Computer Vision*, 2022. 2, 7, 9

[23] OpenAI. Gpt-5. https://chat.openai.com, 2025. Large multimodal language model developed by OpenAI. 3, 7

[24] Jer Pelhan, Alan Lukežič, Vitjan Zavrtanik, and Matej Kristan. A novel unified architecture for low-shot counting by

detection and segmentation. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2024. 1, 9

[25] Jer Pelhan, Alan Lukežič, Vitjan Zavrtanik, and Matej Kristan. Dave – a detect-and-verify paradigm for low-shot counting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2024. 1, 9

[26] 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 *Proceedings of the International Conference on Machine Learning*, 2021. 9

[27] Viresh Ranjan, Udbhav Sharma, Thu Nguyen, and Minh Hoai. Learning to count everything. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2021. 1, 2, 3, 4, 7

[28] Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. Sam 2: Segment anything in images and videos. In *Proceedings of the International Conference on Learning Representations*, 2025. 9

[29] Nuno Vasconcelos and Andrew Lippman. Learning from user feedback in image retrieval systems. In *Advances in Neural Information Processing Systems*, 1999. 9

[30] Yang Xu, Yifan Feng, and Yue Gao. Negative prompt driven complementary parallel representation for open-world 3d object retrieval. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2024. 9

[31] Yingying Zhang, Desen Zhou, Siqin Chen, Shenghua Gao, and Yi Ma. Single-image crowd counting via multi-column convolutional neural network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 3, 8

[32] Huang Zhizhong, Dai Mingliang, Zhang Yi, Zhang Junping, and Shan Hongming. Point, segment and count: A generalized framework for object counting. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2024. 1, 9