YOLO-Count: Differentiable Object Counting for Text-to-Image Generation

Supplementary Material

1. Additional Details and Ablations

1.1. Efficiency Comparison with Other Models

To assess efficiency, we compare YOLO-Count with other object counting models in terms of backbone architecture, parameter count, and inference speed, as summarized in Tab. 1. For inference speed evaluation, we measure the frames per second (FPS) of all models on a single NVIDIA RTX 3090 GPU.

The results demonstrate that YOLO-Count achieves at least a 5× speedup over other high-accuracy models, primarily due to its lightweight YOLO-based backbone, which avoids the computational overhead of heavy transformer-based vision backbones such as GroundingDINO [6] and CLIP [8]. This combination of high efficiency and strong performance highlights YOLO-Count as a practical, plug-and-play module for integrating accurate quantity control into text-to-image (T2I) generation pipelines.

Table 1. Comparison of model architecture and efficiency.

Model	#Params	FPS	Backbone
CountGD [2]	146M	4.93	SwinT
CLIP-Count [4]	101M	10.34	ViT
VLCounter [5]	103M	8.45	ViT
CounTX [1]	97M	9.98	SwinT
DAVE [7]	150M	2.37	SwinT
YOLO-Count (**Ours**)	68M	50.41	CNN

1.2. Token Optimization for T2I Control

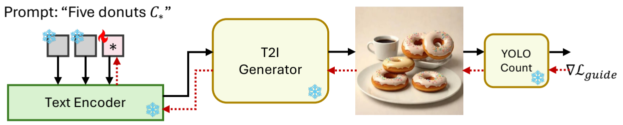

Figure 1. Pipeline for counting-controlled generation via token optimization. (C_* denotes the counting token.)

We adopt a token optimization strategy analogous to textual inversion. Following [10], we iteratively update the learnable counting token embedding using gradients derived from the discrepancy between the predicted and target counts. This process progressively refines the token representation, guiding the text-to-image (T2I) model to generate images that match the desired object quantity, as illustrated in Fig. 1. In practice, this optimization requires at most 150 gradient steps and can be executed on a single 32 GB NVIDIA V100 GPU. Optimizing object count for a single image takes approximately 40–180 seconds, depending on the quantity of the target and the complexity of the image.

1.3. Categories for Controllable Generation Benchmarks

- **LargeGen:** sea shell, apple, orange, marble, green pea, bottle cap, peach, egg, chair, tree log.
- **LargeGen-New:** egg tart, mango, lemon, onion, gold medal, beaker, harmonica, baozi, jellyfish, llama.

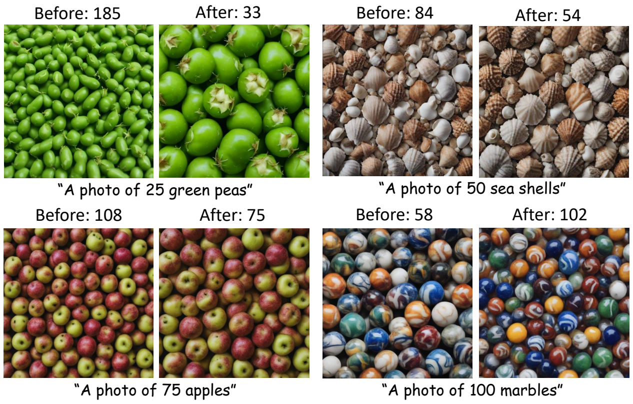

Figure 2. Results of quantity control on LargeGen.

Figs. 2 and 3 shows additional qualitative results of counting-controlled generation on the LargeGen and LargeGen-New benchmarks, respectively, demonstrating YOLO-Count's ability to accurately guide object quantity control across both seen and novel categories.

1.4. Effect of the Classification Branch During Inference

The standard inference procedure for YOLO-Count involves summing the cardinality map, as used in all main experiments. However, since YOLO-Count includes an additional classification branch, which is originally designed

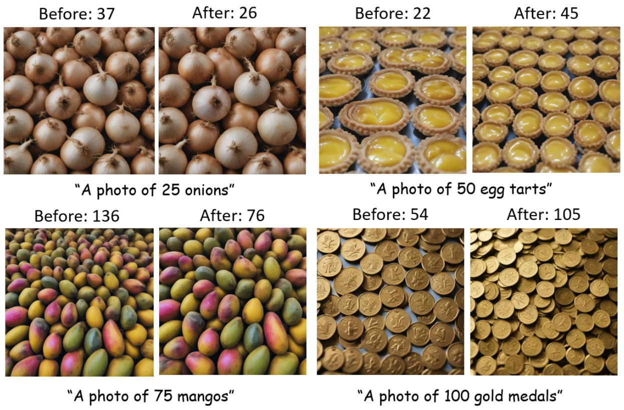

Figure 3. Results of quantity control on LargeGen-New.

to aid training, we explore using its classification output to refine inference results.

During inference, we filter the cardinality regression output $\hat{y}_{\text{cnt}}$ using classification probabilities $\hat{y}_{\text{cls}}$. Specifically, only grid cells with classification probabilities exceeding a predefined threshold κ are considered valid for counting. The final count is computed as:

$$\text{Count} = \sum_{p \in \mathcal{P}} \hat{y}_{\text{cnt}}(p), \tag{1}$$

where $\mathcal{P} = \{p \mid \hat{y}_{\text{cls}}(p) > \kappa\}$ represents the set of grid cells whose classification probability surpasses the threshold κ. We evaluate counting accuracy across different κ values on the FSC147 [9] and LVIS [3] datasets.

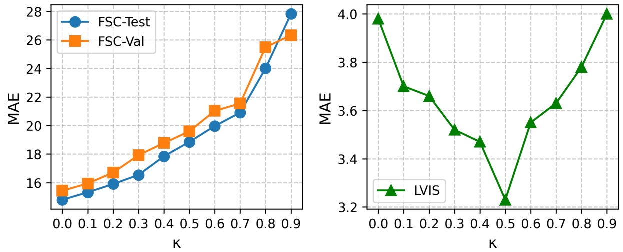

Figure 4. Counting MAE of FSC147 and LVIS under different thresholds.

As shown in Fig. 4, the optimal κ differs by dataset: $\kappa = 0.0$ achieves the lowest MAE on FSC147, whereas $\kappa = 0.5$ performs best on LVIS. This difference reflects dataset-specific annotation protocols. FSC147 employs inclusive labeling, counting any object partially matching the prompt, favoring lower thresholds ($\kappa = 0.0$) to avoid missed detections. Conversely, LVIS provides precise multi-category annotations requiring stricter category separation, where moderate thresholds ($\kappa = 0.5$) effectively filter visually similar but incorrect categories.

Fig. 5 illustrates this effect in a color-based ball counting task. Baseline models such as CountGD [2] and CLIP-Count [4] indiscriminately count all colored balls. In contrast, YOLO-Count adapts based on κ: at $\kappa = 0.0$, it mirrors inclusive counting behavior, while at $\kappa = 0.5$, it selectively counts only the target color by leveraging classification filtering. This shows that κ serves as an inference-time hyperparameter, allowing flexible adaptation to task requirements, favoring lower thresholds for inclusive counting (FSC147 style) and higher thresholds for strict categorical discrimination (LVIS style).

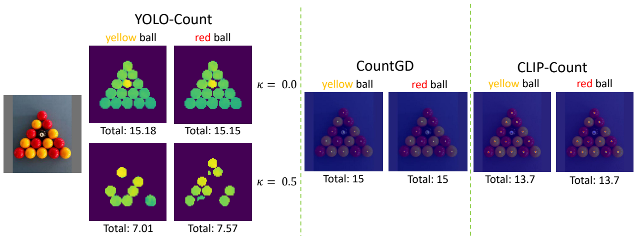

Figure 5. Demonstrations of YOLO-Count in distinguishing semantically similar categories through classification thresholding.

2. Limitations and Future Work

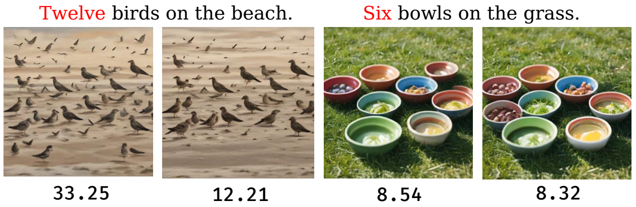

Figure 6. Common failure modes. Numbers below are counts.

Despite its strong performance, YOLO-Count exhibits several limitations, as illustrated in Fig. 6. On the left, the model suffers from incorrect counting, failing to detect small or background objects such as birds in cluttered scenes. On the right, token optimization fails to reduce the object count toward the specified target, leading to unsuccessful quantity control in challenging scenarios. Furthermore, YOLO-Count's design is inherently dependent on the YOLO architecture, which, while efficient, restricts its seamless integration with state-of-the-art text-to-image (T2I) diffusion models.

Future work could address these limitations by exploring Transformer-based or hybrid architectures to improve robustness in dense and fine-grained counting scenarios. Additionally, incorporating joint optimization directly within the diffusion process, rather than relying on post-hoc token optimization, may provide stronger and more stable signals for quantity control, enabling tighter coupling between counting models and generative pipelines.

References

[1] Niki Amini-Naieni, Kiana Amini-Naieni, Tengda Han, and Andrew Zisserman. Open-world text-specifed object counting. In *BMVC*, page 510, 2023.

[2] Niki Amini-Naieni, Tengda Han, and Andrew Zisserman. Countgd: Multi-modal open-world counting. In *Advances in Neural Information Processing Systems*, pages 48810–48837. Curran Associates, Inc., 2024.

[3] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[4] Ruixiang Jiang, Lingbo Liu, and Changwen Chen. Clipcount: Towards text-guided zero-shot object counting. In *Proceedings of the 31st ACM International Conference on Multimedia*, pages 4535–4545, 2023.

[5] Seunggu Kang, WonJun Moon, Euiyeon Kim, and Jae-Pil Heo. Vlcounter: Text-aware visual representation for zero-shot object counting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2714–2722, 2024.

[6] Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, et al. Grounding dino: Marrying dino with grounded pre-training for open-set object detection. In *European Conference on Computer Vision*, pages 38–55. Springer, 2024.

[7] Jer Pelhan, Alan Lukeži?, Vitjan Zavrtanik, and Matej Kristan. Dave - a detect-and-verify paradigm for low-shot counting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 23293–23302, 2024.

[8] 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 38th International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021.

[9] Viresh Ranjan, Udbhav Sharma, Thu Nguyen, and Minh Hoai. Learning to count everything. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3394–3403, 2021.

[10] Oz Zafar, Lior Wolf, and Idan Schwartz. Iterative object count optimization for text-to-image diffusion models. *arXiv preprint arXiv:2408.11721*, 2024.