Gen-n-Val: Agentic Image Data Generation and Validation

Jing-En Huang*,†,‡ I-Sheng Fang*,‡ Tzuhsuan Huang‡
Yu-Lun Liu† Chih-Yu Wang‡ Jun-Cheng Chen‡
† National Yang Ming Chiao Tung University
‡Research Center for Information Technology Innovation, Academia Sinica

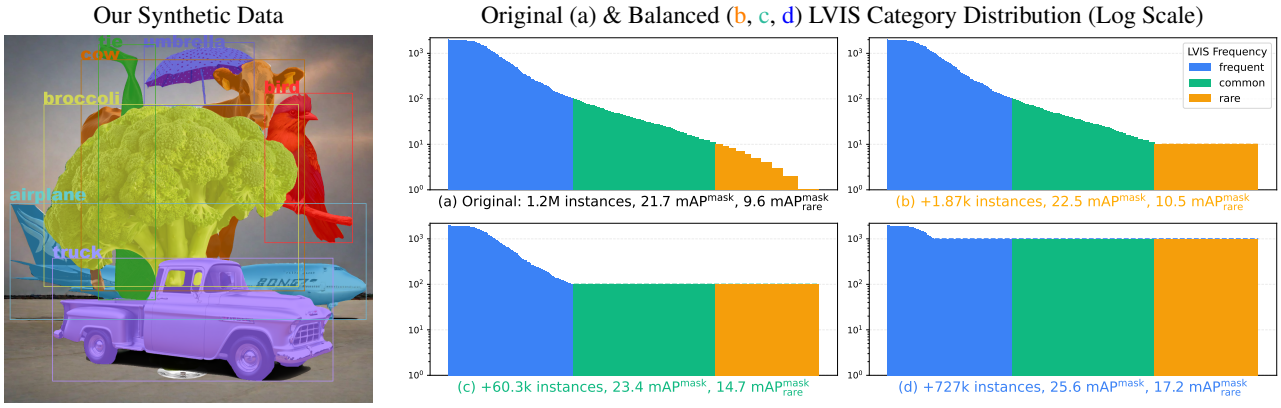


Figure 1. **Gen-n-Val generates high-quality synthetic data to address the long-tailed distribution in instance segmentation.** *(Left)* Sample synthetic data generated by Gen-n-Val with precise instance masks (different colors indicate different object categories). *(Right)* Log-scale category distribution of LVIS [14], a dataset known for its severe long-tailed distribution where rare categories (orange) often have fewer than 10 images. With Gen-n-Val, we inject 1,874, 60,306, and 727,393 additional instances for rarer categories, substantially balancing the long-tailed distribution such that each category has at least 10, 100, and 1,000 images. This redistribution leads to gains of +0.8, +1.7, and +3.9 mask mAP overall and +0.9, +5.1, and +7.6 mask mAP on rare categories with Mask R-CNN [16], demonstrating the effectiveness of our agentic data generation and validation pipeline in addressing data scarcity for underrepresented classes.

Abstract

The data scarcity, label noise, and long-tailed category imbalance remain important and unresolved challenges in many computer vision tasks, such as object detection and instance segmentation, especially on large-vocabulary benchmarks like LVIS, where most categories appear in only a few images. Current synthetic data generation methods still suffer from multiple objects per mask, inaccurate segmentation, incorrect category labels, and other issues, limiting their effectiveness. To address these issues, we introduce Gen-n-Val, a novel agentic data generation framework that leverages Layer Diffusion (LD), a Large Language Model (LLM), and a Vision Large Language Model (VLLM) to produce high-quality and diverse instance masks and images for object detection and instance segmentation. Gen-n-Val consists of two agents: (1) the LD prompt agent, an LLM, optimizes prompts to encourage LD to generate high-quality foreground single-object images and corresponding segmentation masks; and (2) the data validation agent, a VLLM, filters out low-quality synthetic instance images. The system prompts for both agents are optimized by TextGrad. Compared to state-of-the-art synthetic data approaches like MosaicFusion, our approach reduces invalid synthetic data from 50% to 7% and improves performance by 7.6% on rare classes in LVIS instance segmentation with Mask R-CNN, and by 3.6% mAP on rare classes in COCO instance segmentation with YOLOv9c and YOLO11m. Furthermore, Gen-n-Val shows significant improvements (7.1% mAP) over YOLO-Worldv2-M in open-vocabulary object detection benchmarks with YOLO11m. Moreover, Gen-n-Val has scalability in model capacity and dataset size. The code is available at https://github.com/aiiu-lab/Gen-n-Val.

1. Introduction

High-quality data plays a crucial role in computer vision tasks, such as instance segmentation and object detection. However, these tasks suffer from insufficient data. This is-

* indicates equal contribution.

Figure 2. **Common failures in MosaicFusion [39] synthetic data.** Despite using prompt "`a photo of a single <object>`", the method produces: (1) incomplete objects (blue: typewriter cut off), (2) incomplete segmentation (purple: salmon; red: snowmen), (3) multiple objects with single mask (red: two snowmen), and (4) wrong categories (yellow: whole apples labeled "apple sauce").

sue causes poor performance and class imbalance, resulting in low Average Precision (AP) in rare classes. The root of insufficient data is the cost of constructing the dataset. Building large-scale datasets is extremely time-consuming and labor-intensive, and the quality of labels could be unreliable due to human errors, such as missing labels, wrong labels, and inaccurate masks or bounding boxes.

The naive way to synthesize data could be copy-paste [8], which means copying foreground objects from images and pasting them onto random backgrounds, or generative models, such as generative adversarial networks (GAN) [43] or diffusion models [2]. Because of recent advances in text-to-image diffusion models [29], X-Paste [44] and MosaicFusion [39] use diffusion models to generate images. However, compared to image generation, obtaining labeled masks or bounding boxes for synthetic data is more challenging. X-Paste uses an off-the-shelf segmentation model to get the instance mask of an object. MosaicFusion generates images and masks simultaneously by using a cross-attention map. While MosaicFusion can produce a broad vocabulary of instances, several challenges remain in creating high-quality synthetic data for instance segmentation, causing MosaicFusion to discard $\approx 50\%$ of the generated images and masks. Moreover, as Figure 2 shows, another $\approx 50\%$ of the filtered generated images and masks still suffer from several issues. First, it may generate multiple objects within a single mask. For example, it generates two snowmen and marks them with a single mask (red region). Second, it could produce masks that do not accurately capture individual instances. For example, not all objects are properly marked. Third, it could occasionally generate an incorrect category. For example, the yellow-masked area should be "apple *sauce*", but it generates "apples".

To tackle these challenges, in this work, we introduce Gen-n-Val, a novel agentic data generation and validation framework. Gen-n-Val leverages Layer Diffusion (LD) [42] and large language model (LLM)/vision large-language models (VLLM) agents [7] to generate high-quality synthetic data. We leverage LD to generate foreground images and masks without using the off-the-shelf segmentation model or cross-attention. However, we observe that $\approx 44\%$ of the synthetic data generated by LD with the standard prompt (*e.g.* "`a photo of a single <object>, <description>`") is invalid, as shown in Figure 3. These images and masks generated by standard prompts not only have low diversity but also have multiple objects due to monotonous and ambiguous descriptions. Recently, LLMs and VLLMs as agents have gained significant attention. LLMs/VLLMs provide a natural language interface for interacting with other components. Therefore, we leverage an LLM as the LD prompt agent and employ TextGrad [41] to refine the system prompts of the LLM, enabling the LLM agent to produce optimized prompts that guide LD to generate high-quality and diverse single-object image instances along with their corresponding segmentation masks.

However, the data generated by these optimized prompts still contains approximately 7% invalid samples (*e.g.* samples without the target object, with multiple objects, or with incorrect categories), as shown in Figure 4. To filter out these failed cases, we use the VLLM as the data validation agent. The data validation agent's system prompt is also optimized by TextGrad. Finally, we composite validated foreground instances onto background images to produce diverse, high-quality synthetic data for downstream tasks.

As demonstrated in Figure 1, Gen-n-Val allows us to generate high-quality data for instance segmentation and object detection models, effectively augmenting rare classes, systematically balancing the category distribution, and consistently improving overall model performance.

Our main contributions are summarized as follows.

- We propose Gen-n-Val, an agentic framework for generating diverse high-quality data for instance segmentation and object detection with two agents, the LD prompt agent and the data validation agent.
- We use TextGrad to refine the system prompts of the LD prompt agent to generate a high-quality image generation prompt for LD, allowing LD to generate a single foreground instance with a precise mask. We also use a VLLM as the data validation agent to validate generated data, improving the quality of synthetic data.
- The dataset synthesized by Gen-n-Val significantly outperforms previous data synthesis methods in improving model performance. Moreover, Gen-n-Val also has scalability in model capacity and dataset size.

Figure 3. **The samples generated by Layer Diffusion [42] with *standard* prompt.** "*An image of a single orange_(fruit), orange (FRUIT of an orange tree).*". These images, generated using standard prompts, suffer from low diversity and often contain multiple unintended objects due to their monotonous and ambiguous descriptions.

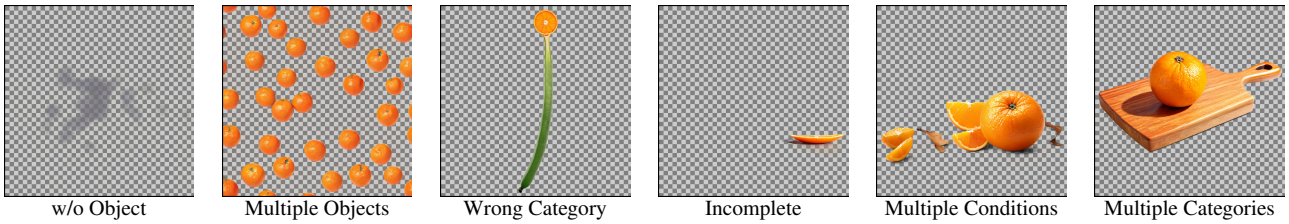


Figure 4. **Failure samples generated by Layer Diffusion [42] with *optimized* prompt.**

2. Related Work

2.1. Copy-Paste Augmentation

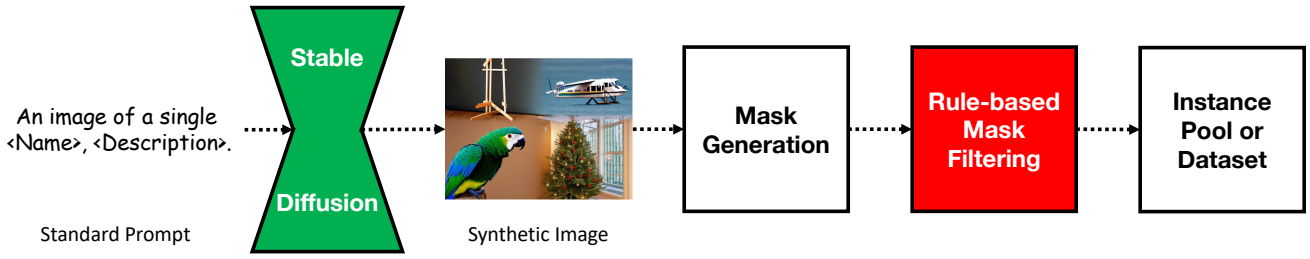
To automatically identify realistic locations for object pasting, Dvornik *et al*. [8] introduce a context model that enables objects to be seamlessly pasted into new scenes, significantly improving object detection accuracy. In contrast, Fang *et al*. [12] jitter objects that already exist within the image rather than directly pasting them from other images. Additionally, Dwibedi *et al*. [9] propose a method that cuts and blends instances onto random backgrounds to avoid artifacts that can arise from direct pasting. In Simple-Copy-Paste [13], Ghiasi *et al*. do not model the surrounding visual context before placing copied instances, but demonstrate that random object placement leads to notable improvements compared to previous approaches. While these methods can enhance performance, they may fall short in providing the diverse and high-quality masks and images.

2.2. Generative-Based Augmentation

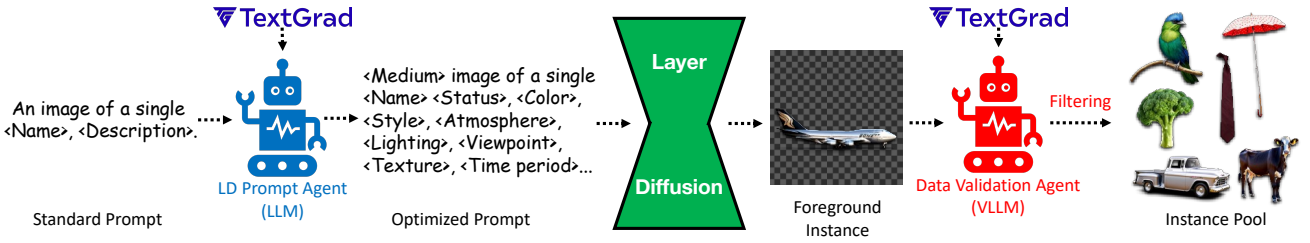
Leveraging generative adversarial networks (GANs), [43] utilize a small set of labeled data to train a simple MLP classifier for classifying pixel-wise feature vectors produced by StyleGAN [17]. This classifier then serves as a label-generating branch within the StyleGAN architecture. Consequently, data can be generated by sampling latent codes and passing them through the StyleGAN. Following DatasetGAN, [20] extend BigGAN [3] and VQGAN [10] with a segmentation branch, scaling DatasetGAN to the ImageNet [30] level. Since DatasetGAN is trained on synthetic images, it can only sample objects with limited diversity and a less natural appearance. With the aid of powerful diffusion models, Baranchuk *et al*. [2] propose a method trained on labeled real images, exploring intermediate activations in pre-trained diffusion models. These activations effectively capture semantic information from input images, making them useful for segmentation tasks. Although the quality of synthetic data is promising, previous methods can generate only a limited range of object categories.

Recent advances in text-to-image models [26, 28, 29] can generate diverse objects in images with natural language prompts. Therefore, Zhao *et al*. [44] introduce X-Paste, which synthesizes images with Stable Diffusion and segments these images with an off-the-shelf segmentation model. However, generating data with X-Paste is time-consuming, requiring 4.3 times more GPU hours than MosaicFusion [39]. Gen2Det [35] uses an off-the-shelf box-label-conditioned inpainting diffusion model and bounding-box masks to generate data for object detection. Xie *et al*. [39] introduce a training-free, diffusion-based data augmentation pipeline capable of simultaneously producing image and mask pairs by leveraging off-the-shelf text-to-image diffusion models [26, 29].

As shown in Figure 5a, the pipeline of these methods involves four steps: (1) Generate images using Stable Diffusion with prompts filled by category name; (2) Obtain segmentation masks using either augmentation models or cross-attention maps; (3) Refine masks through edge detection and filter out flawed masks; (4) Composite the instance and background to create final images. However, these methods have several issues. First, the generated images are not guaranteed to contain only one instance. Second, the segmentation masks generated by cross-attention are of poor quality. Lastly, the generated images are fil-



(a) **Previous generative-based augmentation methods [39, 44].**



(b) **Gen-n-Val (Ours).**

Figure 5. **Comparison of previous augmentation pipelines [39, 44] and our Gen-n-Val pipeline.** (a) Pipeline of previous generative-based augmentation methods [39, 44]. The process involves four steps: First, applying Stable Diffusion to generate images with a standard prompt. Next, using augmentation models [44] or the relationship between visual and textual embeddings in cross-attention [39] to obtain segmentation masks. Finally, filtering out flawed masks with hand-crafted rules. (b) Pipeline of Gen-n-Val. The process begins with optimizing system prompts of the Layer Diffusion [42] prompt agent (LLM) using TextGrad. The optimized prompts generate Layer Diffusion [42] prompts, and Layer Diffusion [42] produces transparent instances and background images. A VLLM-based validation agent filters flawed images, and multiple valid instances are pasted into other images. See Section 3 for more details.

tered by hand-crafted rules, such as thresholding, leading to a large number of unqualified images, such as missing corresponding masks and incorrect object labels. In contrast, our method can generate data with precise masks, correct category, and the correct number of object instances, enhancing the performance of downstream tasks.

2.3. Generative Models and Agents

Layer Diffusion [42] is an approach for generating transparent images. This method encodes the transparent alpha channel into the latent distribution of Stable Diffusion [26, 29] to generate transparent images.

OpenAI released the Large Language Model (LLM), GPT-4 [1], and the Vision Large Language Model (VLLM), GPT-4V [24]. Inspired by GPT-4V, Liu *et al*. [22] introduce Large Language-and-Vision Assistant (LLaVA), an open-source VLLM based on the CLIP image encoder [27] and the open-source LLM Vicuna [6]. Besides, Meta released Llama [36] and Llama 3 [7], powerful language models that perform exceptionally well across a wide range of language understanding tasks, comparable to leading models such as GPT-4. Moreover, Meta released Llama 3.2 [23], integrating vision capability into Llama and allowing it to process visual inputs. To enhance the feedback provided by LLM, Yuksekgonul *et al*. [41] propose TextGrad, a method to refine system prompts by backpropagating textual information from outputs, encouraging precise answers.

With the development of LLMs, using them to decide and solve various tasks has become a growing trend. A common approach is to use LLMs to generate textual "actions" or "decisions" for tasks. VISPROG [15] uses GPT3 [4] to automatically generate programs that solve complex visual tasks. ToolFormer [31] also uses GPT3 to decide which APIs to call for solving tasks. HuggingGPT [32] uses ChatGPT to choose models on HuggingFace for solving complicated AI tasks. ReAct [40] combines reasoning and action steps, allowing LLM to make better decisions and provide more reliable answers. Reflexion [33] enhances LLM agents' decision-making by using reflective linguistic feedback stored in an episodic memory. Zheng *et al*. [45] use GPT-4V as a web agent. Generative Agents [25] introduce a system of generative agents that simulate realistic human behaviors through memory, planning, and reflection, enabling interactive digital environments. In summary, these methods demonstrate the growing power of LLMs to integrate decision-making and execution, allowing more robust and adaptable AI solutions across diverse domains.

3. Method

To tackle the issues we discuss in Section 2.2, we leverage a Large Language Model (LLM) as the LD prompt agent, a Vision Large Language Model (VLLM) as the data

validation agent, and Layer Diffusion (LD) to generate diverse and high-quality data, as shown in Figure 5b. First, in the **Open Vocabulary Prompt Generation** stage, we initiate the process by employing TextGrad [41] optimization techniques to refine the quality and effectiveness of the LD prompt agent's system prompt, allowing the LD prompt agent to generate diverse and high-quality prompts for LD. The details are shown in Section 3.1. Second, during the **Foreground Image Generation** stage, we utilize the LD to generate transparent instance images. This step creates isolated instance images, allowing us to generate segmentation masks without manual annotation or additional segmentation algorithms. The details are shown in Section 3.2. Third, to ensure the quality of the generated images, we filter out flawed samples during the **Image Filtering** stage using the VLLM as the data validation agent, which is further optimized with TextGrad. The details are shown in Section 3.3. Finally, we randomly paste multiple instances into background images. This step creates diverse scenes with multiple instances, further augmenting the dataset.

3.1. Open Vocabulary Prompt Generation

Previous generative-based augmentation methods relied on adding the word "single" to the prompt during the stage of image generation to instruct the Stable Diffusion to generate a single-object instance. This approach not only struggles to keep the diffusion model focused on a single instance but also accidentally allows other types of objects to appear in the background due to the ambiguous concept of "single". The ideal Stable Diffusion prompts should be detailed and specific with several components, including the object class, action, environment setting, and other relevant details [34]. For example, Figure 5b demonstrates both the standard and optimized prompts for image generation. For more examples, please refer to our appendix.

To generate optimized LD prompts, we need a high-quality system prompt p_{sys} for the LD prompt agent $A_{p_{\text{LD}}}$ (a LLM). Therefore, we leverage two other LLMs, a prompt evaluator E_{prompt} and a prompt validator V_{prompt}, and TextGrad [41] optimization techniques to optimize the system prompt p_{sys}. The system prompt generation process is shown in Algorithm 1. First, we give the LD prompt agent $A_{p_{\text{LD}}}$ an initial system prompt p_{sys} that instructs it to generate a prompt p_{LD} for the LD. Second, we evaluate the p_{LD} using a prompt evaluator E_{prompt} to determine the quality of the prompt for LD. The prompt evaluator E_{prompt} is a second LLM which is asked to evaluate the quality of the prompt generated by the LD prompt agent $A_{p_{\text{LD}}}$ and provide a criticism in the form of a loss L. Third, with the loss L, we optimize the initial system prompt using Text Gradient Descent (TGD) [41] to generate a new optimized system prompt p^*_{sys}. Fourth, we use the optimized system prompt p^*_{sys} to generate an optimized prompt p^*_{LD} for the LD model.

Algorithm 1 Optimize System Prompt p_{sys} of $A_{p_{\text{LD}}}$

Input: initial system prompt p_{sys}, max iteration I
Initialize: decision $d \leftarrow$ False, iteration $i \leftarrow 0$
while not d **or** $i < I$ **do**
  $p_{\text{LD}} \leftarrow A_{p_{\text{LD}}}(p_{\text{sys}})$ {Generate LD prompt}
  $L \leftarrow E_{\text{prompt}}(p_{\text{LD}})$ {Evaluate prompt quality}
  $p^*_{\text{sys}} \leftarrow \text{TGD}(L)$ {Update system prompt}
  $p^*_{\text{LD}} \leftarrow A_{p_{\text{LD}}}(p^*_{\text{sys}})$ {Generate refined LD prompt}
  $d \leftarrow V_{\text{prompt}}(p^*_{\text{LD}})$ {Check if prompt is satisfactory}
  if d **then**
    break {Stop if validator accepts}
  else
    $p_{\text{sys}} \leftarrow p^*_{\text{sys}}$ {Adopt refined system prompt}
  end if
end while=0

Finally, we validate the p^*_{LD} to compare the quality with the p_{LD}. The prompt validator V_{prompt} is a third LLM that is asked to evaluate the quality of the optimized prompt and provide a decision d of whether replacing the initial prompt with the optimized prompt at the next iteration is beneficial.

3.2. Foreground Image Generation

We propose a method that utilizes the LD to generate transparent images of foreground instances along with precise alpha masks. By employing the optimized prompts from the Open Vocabulary Prompt Generation stage, we empower the LD model to produce high-quality, single-object instances that strictly adhere to the specified prompts. The optimized prompts are input into the LD model to generate transparent images where each pixel contains both RGB values and an alpha transparency channel. This transparency channel inherently provides an accurate segmentation mask for the foreground object. Unlike previous methods, our approach eliminates the need for additional segmentation algorithms, such as SAM [19], or manual annotation, as the alpha channel directly corresponds to the object's silhouette, resulting in precise and clean masks aligned perfectly with the generated images.

The detailed and specific prompts generated by the LD prompt agent $A_{p_{\text{LD}}}$ include various attributes, such as object class, style, color, texture, lighting, atmosphere, viewpoint, and time period. By incorporating these attributes, we guide the LD to produce a wide range of object appearances, increasing the diversity of the generated instances. Prompt conditioning and negative prompts are also used to guide the model toward generating images that closely match the desired characteristics while avoiding undesired elements.

While the LD generates high-quality transparent instances, minor background noise may still be present in the alpha channel due to imperfections in the diffusion process. To mitigate this, we apply a median filter to the al-

pha channel of the images. The median filter effectively removes isolated pixels and smooths the mask edges without significantly altering the overall shape of the object. This post-processing step ensures that the segmentation masks are clean and accurately represent the foreground instances.

3.3. Image Filtering

Despite the improvements achieved through optimized prompts and the use of the LD, some generated images may still contain flaws, such as missing the target object, featuring multiple instances when only one is desired, incorrect object categories, or poor visual quality, as shown in Figure 4. To further enhance the quality of our synthetic dataset, we employ the VLLM as a data validation agent to automatically filter out these flawed images.

Similar to our approach in optimizing prompts for the LD, we apply the TextGrad optimization technique to refine the system prompts of the data validation agent. By optimizing the data validation agent's system prompts, we enhance its ability to evaluate and identify flaws in the generated images. The optimized prompts guide the data validation agent to focus on specific criteria essential for high-quality instance segmentation data, such as the presence of a single, correctly categorized object, and the cleanliness of the transparent background in the foreground images.

The data validation agent is tasked with analyzing generated images to assess their suitability for inclusion in the dataset. Here, c represents the object category. The key criteria encoded into the data validation agent through the optimized system prompt include:

- Single c: The image should contain only one c.
- Single View: The c should be shown from a single angle or perspective.
- Intact c: The c should be intact and fully visible.
- Plain Background: The background should be empty or plain, without distracting elements.

4. Experiments

To demonstrate the enhancement of detection models after applying our methods, we evaluate on two challenging benchmarks, COCO [21] and LVIS [14]. COCO dataset is a large-scale dataset designed for object detection, segmentation, and captioning, containing 330K images with 1.5M object instances labeled across 80 categories. The LVIS dataset uses the same images as COCO but provides more detailed and partitioned annotations across 1,203 categories, offering finer classification. To quantify this improvement, we compute mean Average Precision (mAP) as our evaluation metric. We use Meta-LLaMA-3.1-8B-Instruct as the LD prompt agent and Meta-LLaMA-3.2-11B-Vision-Instruct as the data validation agent. We apply Copy-Paste [8] in all YOLO experiments except for the baseline. For more details, please refer to our appendix.

Table 1. **LVIS [14] instance segmentation and object detection benchmark.** We borrow numbers of Mask R-CNN [16] and MosaicFusion [39] from MosaicFusion [39].

Method	mAP^{box}	mAP^{box}_{rare}	mAP^{mask}	mAP^{mask}_{rare}
Mask R-CNN (baseline)	22.5	9.1	21.7	9.6
MosaicFusion [39]	24.0	14.8	23.1	15.2
Gen2Det [35]	24.4	15.4	23.6	15.3
Gen-n-Val (Ours)	**26.8**	**16.1**	**25.6**	**17.2**
versus baseline	+4.3	+7.0	+3.9	+7.6
YOLO11m (baseline)	12.9	8.3	10.3	6.5
Copy-Paste [8]	12.9	8.2	10.4	6.7
Gen-n-Val (Ours)	**17.2**	**13.0**	**14.5**	**10.1**
versus baseline	+4.3	+4.7	+4.2	+3.6
CenterNet2 (baseline)	47.5	41.4	42.3	36.8
XPaste [44]	50.1	48.2	44.4	43.3
DiverGen [11]	51.2	50.1	45.5	45.8
Gen-n-Val (Ours)	**51.5**	**55.6**	**45.9**	**48.3**
versus baseline	+4.0	+14.2	+3.6	+11.5

Table 2. **Effect of flattening the LVIS [14] class distribution.** By augmenting images with additional instances, we enforce a minimum of 10–1000 images per class, which substantially boosts performance on rare categories and improves overall accuracy. **Min. img./cls.**: the minimum number of images per class. **Add. inst.**: the number of added instances.

Min. img./cls.	Add. inst.	mAP^{box}	mAP^{box}_{rare}	mAP^{mask}	mAP^{mask}_{rare}
Original	0	22.5	9.1	21.7	9.6
10	1,874	23.1	9.8	22.5	10.5
100	60,306	24.2	13.5	23.4	14.7
1000	727,393	**26.8**	**16.1**	**25.6**	**17.2**

4.1. LVIS Benchmark

We train Mask R-CNN [16], YOLO11m [18], and CenterNet2 [46] on the LVIS training set. As shown in the right part of Figure 1, the LVIS training set exhibits a highly long-tailed category distribution, where most categories appear in only a few images. To balance this distribution, we generate 727,393 instances and paste them onto LVIS images until all 1,203 categories reach at least 1,000 images. For rare category analysis, we adopt the category frequency annotations provided by LVIS. We evaluate on the LVIS validation set. The results are shown in Table 1. Compared to the baseline, our method improves the mAP by 4.3% for box prediction and 3.9% for mask prediction on Mask R-CNN, by 4.0% for box prediction and 3.6% for mask prediction on CenterNet2, and by 4.3% for box prediction and 4.2% for mask prediction on YOLO11m. Rare-category mAP is also significantly improved over the baselines.

4.2. Distribution Balancing of LVIS

To investigate Gen-n-Val's ability to balance the long-tailed distribution of LVIS, as shown in Figure 1, we progressively "fill" the tail of the distribution by generating additional synthetic instances for under-represented classes. Concretely, starting from the original LVIS distribution, we

Table 3. **COCO [21] instance segmentation and object detection benchmark with YOLO9c [37] and YOLO11m [18].**

Method	mAP^{box}	mAP^{box}_{rare}	mAP^{mask}	mAP^{mask}_{rare}
YOLOv9c (baseline)	50.1	52.8	41.3	45.1
Copy-Paste	50.9	53.1	42.1	46.8
MosaicFusion	51.4	53.6	42.7	47.9
Gen-n-Val	**51.9**	**54.4**	**43.4**	**48.7**
versus baseline	+1.8	+1.6	+2.1	+3.6
YOLO11m (baseline)	49.6	51.9	39.8	45.4
Copy-Paste	50.0	52.1	41.5	47.1
MosaicFusion	50.6	54.3	42.0	48.1
Gen-n-Val	**51.7**	**55.4**	**42.9**	**49.0**
versus baseline	+2.1	+3.5	+3.1	+3.6

construct three variants of the training set by enforcing different lower bounds on the number of images per class: (1) a mild setting where we up-sample rare categories so that every class appears in at least 10 images, (2) a stronger setting that further densifies the common categories until each class has at least 100 images, and (3) an aggressive setting where we continue to generate instances until all 1,203 categories reach at least 1,000 images. These three stages correspond to injecting 1.8K, 60.3K, and 727K additional instances, respectively, and balance the long-tailed distribution as shown in Figure 1. As demonstrated in Table 2, Gen-n-Val leads to consistent performance gains for Mask R-CNN: mask mAP improves from 21.7 to 22.5, 23.4, and 25.6, while rare-category mask mAP increases from 9.6 to 10.5, 14.7, and 17.2. Overall, enforcing 10∼1,000 images per class translates into +0.8∼+3.9 mAP on all categories and +0.9∼+7.6 mAP on rare categories, demonstrating that Gen-n-Val effectively balances LVIS and benefits tail classes.

4.3. COCO Benchmark

We train YOLOv9c [37] and YOLO11m [18] using the COCO dataset and our 16K synthetic dataset, and evaluate on the validation set. We choose the 10 least frequent categories as rare categories and the remaining 70 categories as common categories. We use a synthetic background image and paste instances with image harmonization [38]. We compare our method with the baseline YOLOv9c, YOLO11m, and Copy-Paste. As shown in Table 3, Gen-n-Val outperforms the baseline and other methods, achieving the highest mAP scores for both box and mask predictions. Compared to the baseline, our method improves the mAP by 1.8% for box prediction and 2.1% for mask prediction on YOLOv9c, and by 2.1% for box prediction and 3.1% for mask prediction on YOLO11m. Furthermore, rare-category mask mAP improves by 3.6% on YOLOv9c and by 3.6% on YOLO11m, demonstrating the effectiveness of our method. These results demonstrate the effectiveness of our synthetic data generation approach.

Table 4. **COCO [21] open-vocabulary object detection.**

Method	mAP^{box}	mAP^{box}_{novel}
YOLO-Worldv2-M	42.7	20.6
YOLO11m w/ Gen-n-Val	**49.8**	**25.5**

Table 5. **Ablation results on YOLO11m [18] trained with synthetic data on COCO [21].** We evaluate the effects of Layer Diffusion [42] prompt optimization (p^*_{LD}), VLLM filtering (VLLM), median filtering (Med.), and Copy-Paste [8] (C-P).

p^*_{LD}	VLLM	Med.	C-P	mAP^{box}	mAP^{mask}
	✓	✓	✓	48.8	39.2
✓		✓	✓	51.5	42.0
✓	✓		✓	51.5	42.2
✓	✓	✓		51.2	42.7
✓	✓	✓	✓	**51.7**	**42.9**

4.4. Open-Vocabulary Object Detection

We train YOLO11m using the COCO dataset and our 16K synthetic dataset, which includes 80 categories from the COCO dataset and 10 additional categories from the LVIS dataset, and evaluate on the 5K-image COCO/LVIS validation set for open-vocabulary object detection. Ten LVIS categories are selected as novel categories in the validation set, and the whole COCO dataset is used as the training set. We compare our method with the baseline YOLO-Worldv2-M [5], which is an advanced real-time object detection model that enhances the YOLO framework with open-vocabulary capabilities. As shown in Table 4, compared to the baseline, Gen-n-Val improves mAP by 7.1% for box prediction and by 4.9% on novel categories with YOLO11m, demonstrating the effectiveness of our method on open-vocabulary object detection.

4.5. Ablation Studies

We conduct an ablation study on COCO by comparing YOLO11m with and without prompt optimization, VLLM filtering, median filtering, and Copy-Paste. As shown in Table 5, prompt optimization provides the largest gain, improving mAP by 2.9% for object detection and 3.7% for instance segmentation. On top of that, VLLM filtering improves mAP by 0.2% and 0.9%, respectively. Median filtering yields an additional 0.2% and 0.7% mAP improvement, while Copy-Paste brings another 0.5% and 0.2% mAP gain for object detection and instance segmentation.

4.6. Model Scalability

To comprehensively evaluate our method across different model scales, we evaluate Gen-n-Val on the COCO dataset using the YOLOv9 and YOLO11 family models, which are

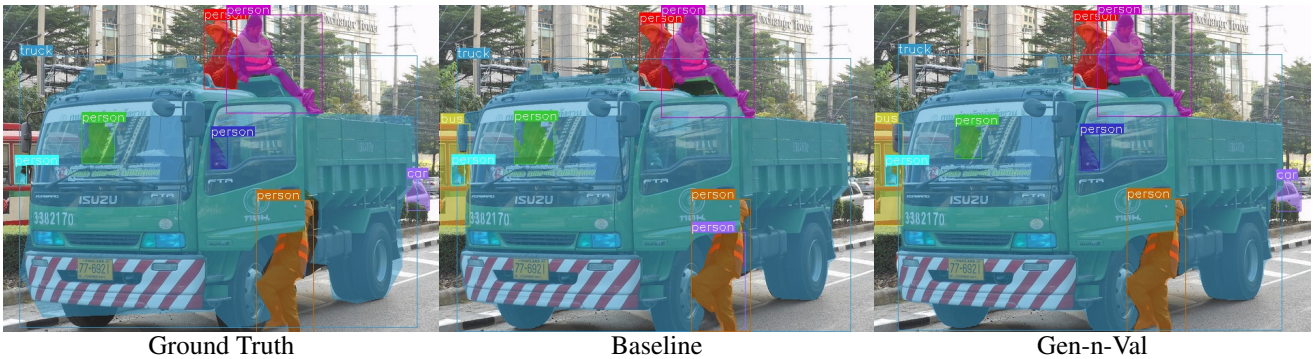


Figure 6. **Qualitative comparison of ground truth, baseline model, and model trained on Gen-n-Val.**

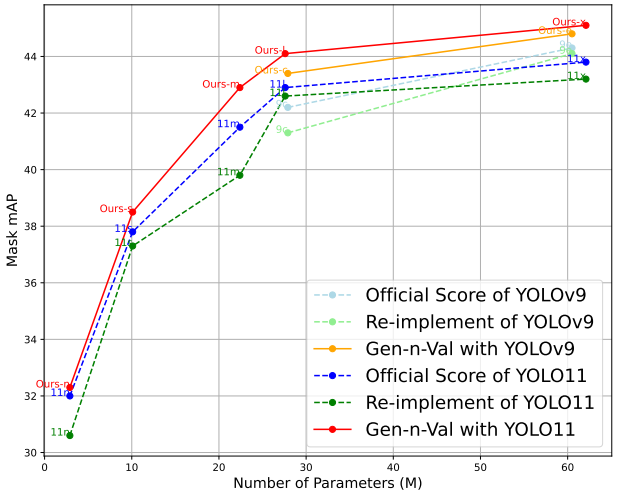


Figure 7. **The instance segmentation performance of YOLOv9 [37] and YOLO11 [18] family models.** As the size of the model increases, performance improves. Gen-n-Val is effective with all family models.

Table 6. **The scalability of synthetic data.** As the size of the dataset increases, performance improves.

Size	mAP^{box}	$\text{mAP}^{\text{box}}_{\text{rare}}$	mAP^{mask}	$\text{mAP}^{\text{mask}}_{\text{rare}}$
4K	50.8	54.6	42.2	48.3
8K	51.1	55.0	42.5	48.6
16K	51.7	55.4	42.9	49.0
20K	**52.0**	**55.6**	**43.0**	**49.2**

state-of-the-art in the YOLO series. In the instance segmentation task, as shown in Figure 7, Gen-n-Val yields significant performance improvements over the baseline models, with the most notable improvement observed being a $+3.1\%$ mask mAP gain on YOLO11m.

4.7. Data Scalability

To comprehensively evaluate our method across different data scales, we trained YOLO11m on the COCO dataset and varying sizes of synthetic data, generating the same 80 classes as in the COCO dataset (4K, 8K, 16K, and 20K), and evaluated it on the COCO. The results in Table 6 show our model's scalability with increasing dataset size.

4.8. Qualitative Results

We present qualitative comparisons between the ground truth, baseline model outputs, and Gen-n-Val outputs, evaluated using the YOLO11m. As shown in Figure 6, the baseline model fails to segment the person in the driver's seat of the truck and the car behind the truck. In contrast, Gen-n-Val accurately segments both the person and the car. This result highlights the effectiveness of Gen-n-Val. For more results, please refer to our supplementary material.

5. Conclusion

In this work, we introduce a novel agentic framework, Gen-n-Val, for generating synthetic data, allowing us to create high-quality, diverse, and precisely annotated synthetic datasets. By leveraging Layer Diffusion (LD), Large Language Model (LLM), and Vision Large Language Model (VLLM), Gen-n-Val significantly enhances the quality and usability of synthetic data. The framework consists of two key agents, the LD prompt agent (a LLM) and the data validation agent (a VLLM). The LD prompt agent optimizes the prompt for LD to generate high-quality images and segmentation masks. The data validation agent filters out failure cases of image instances. Our experiments demonstrate that Gen-n-Val outperforms previous data synthesis approaches in instance segmentation and object detection, and scales with dataset size. On LVIS, Gen-n-Val further serves as an effective mechanism for addressing long-tailed category imbalance; it substantially flattens the class distribution. This work highlights the potential of leveraging LLM- and VLLM-based agents not only to improve synthetic data quality, but also to mitigate data imbalance in large-vocabulary computer vision datasets.

Acknowledgment

This project was supported in part by the National Science and Technology Council (NSTC), Taiwan, under Grants 114-2221-E-001-016, 114-2221-E-001-004, 113-2634-F-002-008, 114-2634-F-001-001-MBK, 111-2628-E-001-002-MY3, 114-2221-E-001-017-MY2, and 114-2634-F-002-004, and by Academia Sinica under Grant AS-IAIA-114-M10 and AS-KPQ-112-NETZ-10-A. We thank the National Center for High-performance Computing (NCHC) of the National Institutes of Applied Research (NIAR) in Taiwan for providing computational and storage resources.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023. 4

[2] Dmitry Baranchuk, Ivan Rubachev, Andrey Voynov, Valentin Khrulkov, and Artem Babenko. Label-efficient semantic segmentation with diffusion models. In *ICCV*, 2022. 2, 3

[3] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *ICLR*, 2019. 3

[4] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *NeurIPS*, 33:1877–1901, 2020. 4

[5] Tianheng Cheng, Lin Song, Yixiao Ge, Wenyu Liu, Xinggang Wang, and Ying Shan. Yolo-world: Real-time open-vocabulary object detection. In *CVPR*, 2024. 7

[6] Wei-Lin Chiang, Zhuohan Li, Zi Lin, Ying Sheng, Zhanghao Wu, Hao Zhang, Lianmin Zheng, Siyuan Zhuang, Yonghao Zhuang, Joseph E. Gonzalez, Ion Stoica, and Eric P. Xing. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality, 2023. 4

[7] Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*, 2024. 2, 4

[8] Nikita Dvornik, Julien Mairal, and Cordelia Schmid. Modeling visual context is key to augmenting object detection datasets. In *ECCV*, 2018. 2, 3, 6, 7

[9] Debidatta Dwibedi, Ishan Misra, and Martial Hebert. Cut, paste and learn: Surprisingly easy synthesis for instance detection. In *ICCV*, 2017. 3

[10] Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis. In *CVPR*, 2021. 3

[11] Chengxiang Fan, Muzhi Zhu, Hao Chen, Yang Liu, Weijia Wu, Huaqi Zhang, and Chunhua Shen. DiverGen: Improving instance segmentation by learning wider data distribution with more diverse generative data. In *CVPR*, pages 3986–3995, 2024. 6

[12] Hao-Shu Fang, Jianhua Sun, Runzhong Wang, Minghao Gou, Yong-Lu Li, and Cewu Lu. Instaboost: Boosting instance segmentation via probability map guided copy-pasting. In *ICCV*, 2019. 3

[13] Golnaz Ghiasi, Yin Cui, Aravind Srinivas, Rui Qian, Tsung-Yi Lin, Ekin D. Cubuk, Quoc V. Le, and Barret Zoph. Simple copy-paste is a strong data augmentation method for instance segmentation. In *CVPR*, 2021. 3

[14] Agrim Gupta, Piotr Dollar, and Ross Girshick. Lvis: A dataset for large vocabulary instance segmentation. In *CVPR*, 2019. 1, 6

[15] Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *CVPR*, pages 14953–14962, 2023. 4

[16] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, pages 2961–2969, 2017. 1, 6

[17] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 3

[18] Rahima Khanam and Muhammad Hussain. Yolov11: An overview of the key architectural enhancements. *arXiv preprint arXiv:2410.17725*, 2024. 6, 7, 8

[19] 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. *arXiv:2304.02643*, 2023. 5

[20] Daiqing Li, Huan Ling, Seung Wook Kim, Karsten Kreis, Adela Barriuso, Sanja Fidler, and Antonio Torralba. Bigdatasetgan: Synthesizing imagenet with pixel-wise annotations. In *CVPR*, 2022. 3

[21] 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 *ECCV*, 2014. 6, 7

[22] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning, 2023. 4

[23] AI Meta. Llama 3.2: Revolutionizing edge ai and vision with open, customizable models. *Meta AI Blog. Retrieved December*, 20:2024, 2024. 4

[24] OpenAI. GPT-4V(ision) system card, 2023. 4

[25] Joon Sung Park, Joseph C. O'Brien, Carrie J. Cai, Meredith Ringel Morris, Percy Liang, and Michael S. Bernstein. Generative agents: Interactive simulacra of human behavior. In *UIST*, New York, NY, USA, 2023. Association for Computing Machinery. 4

[26] Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. SDXL: Improving latent diffusion models for high-resolution image synthesis. In *ICLR*, 2024. 3, 4

[27] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry,

Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, pages 8748–8763. PmLR, 2021. 4

[28] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022. 3

[29] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 2, 3, 4

[30] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Bernstein Michael, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. 2015. 3

[31] Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Eric Hambro, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. Toolformer: Language models can teach themselves to use tools. *NeurIPS*, 36:68539–68551, 2023. 4

[32] Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. HuggingGPT: Solving ai tasks with chatgpt and its friends in hugging face. *NeurIPS*, 36:38154–38180, 2023. 4

[33] Noah Shinn, Federico Cassano, Ashwin Gopinath, Karthik R Narasimhan, and Shunyu Yao. Reflexion: language agents with verbal reinforcement learning. In *NeurIPS*, 2023. 4

[34] Adam Stewart. Prompt guide, 2024. 5

[35] Saksham Suri, Fanyi Xiao, Animesh Sinha, Sean Chang Culatana, Raghuraman Krishnamoorthi, Chenchen Zhu, and Abhinav Shrivastava. Gen2det: Generate to detect. *arXiv preprint arXiv:2312.04566*, 2023. 3, 6

[36] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023. 4

[37] Chien-Yao Wang, I-Hau Yeh, and Hong-Yuan Mark Liao. Yolov9: Learning what you want to learn using programmable gradient information. In *ECCV*, 2025. 7, 8

[38] Ke Wang, Michaël Gharbi, He Zhang, Zhihao Xia, and Eli Shechtman. Semi-supervised parametric real-world image harmonization. In *CVPR*, 2023. 7

[39] Jiahao Xie, Wei Li, Xiangtai Li, Ziwei Liu, Yew Soon Ong, and Chen Change Loy. Mosaicfusion: Diffusion models as data augmenters for large vocabulary instance segmentation. *IJCV*, 2024. 2, 3, 4, 6

[40] Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*, 2023. 4

[41] Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Zhi Huang, Carlos Guestrin, and James Zou. Textgrad: Automatic" differentiation" via text. *arXiv preprint arXiv:2406.07496*, 2024. 2, 4, 5

[42] Lvmin Zhang and Maneesh Agrawala. Transparent image layer diffusion using latent transparency. *ACM TOG*, 2024. 2, 3, 4, 7

[43] Yuxuan Zhang, Huan Ling, Jun Gao, Kangxue Yin, Jean-Francois Lafleche, Adela Barriuso, Antonio Torralba, and Sanja Fidler. Datasetgan: Efficient labeled data factory with minimal human effort. In *CVPR*, 2021. 2, 3

[44] Hanqing Zhao, Dianmo Sheng, Jianmin Bao, Dongdong Chen, Dong Chen, Fang Wen, Lu Yuan, Ce Liu, Wenbo Zhou, Qi Chu, et al. X-paste: Revisiting scalable copy-paste for instance segmentation using clip and stablediffusion. In *ICML*, 2023. 2, 3, 4, 6

[45] Boyuan Zheng, Boyu Gou, Jihyung Kil, Huan Sun, and Yu Su. Gpt-4v(ision) is a generalist web agent, if grounded. In *ICML*, 2024. 4

[46] Xingyi Zhou, Vladlen Koltun, and Philipp Krähenbühl. Probabilistic two-stage detection. In *arXiv preprint arXiv:2103.07461*, 2021. 6