UniGen-1.5: Enhancing Image Generation and Editing through Reward Unification in RL

Supplementary Material

A. Data Generation Details

Prompt. 1: Generate Edit Instructions

{condition_image} You are a creative assistant specializing in image editing. Your task is to analyze the provided image and generate {num} distinct prompts to modify the original image across different categories.

Be concise yet cover detailed information. Specify the modification with fine-grained visual details related to appearance, action, attributes and location.

Output Format:
Your entire response must be a single block of text. Separate each distinct prompt with a semicolon (;). Do not use bullet points, numbering, or line breaks.

Example of Expected Output (3 edit instructions)

- "Add a vintage bicycle leaning against the brick wall; Change the color of the car to a deep emerald green; Change the plain brick wall behind the subject to a vibrant, colorful graffiti art wall"

- "Remove the briefcase from the man's hand, and change the color of his tie to a deep crimson red; Make the woman turn her head to the left; Extract the pink top shirt the person wears in the image"

- "Replace the red car in the foreground with a classic vintage motorcycle; Remove the red car parked by the curb; Transfer the image into a dramatic, high-contrast chiaroscuro painting style"

We introduce the construction of **Edit-Align** dataset for the *Edit Instruction Alignment* stage and the **Edit-RL** dataset for the RL stage. Generally, the data source of Edit-RL is a subset of Edit-Align. For each dataset, we collect the condition images, the edit instructions and the description of desired output images. We describe the data generation pipeline in detail as follows.

Sourcing condition images. For Edit-RL dataset, we employ Qwen-Image [77] to generate 5,000 synthetic images based on text prompts collected from COCO val2017 [44] and segment anything dataset (SAM) [36]. For Edit-Align, we further supplement the condition images with around 7,000 samples drawn from the BLIP-3o SFT image source.

Preparing edit instructions. For each condition image in Edit-RL, we leverage Qwen2.5-VL-72B-instruct [4] to construct 10 diverse edit instructions (see Prompt. 1). For the additional Edit-Align images collected from BLIP-3o-SFT, we construct diverse edit instructions using hand-crafted templates.

Generating output descriptions. For Edit-RL, we use a Qwen-2.5-72B-instruct [89] model to generate the description of the desired output images for training data, as shown in Prompt. 2. For additional samples in Edit-Align, we use hand-crafted rules to obtain the edited image descriptions. Next, we filter out edit prompts or captions with poor quality by assessing them with Qwen2.5-VL-72B-instruct [4] (see Prompt. 3). Consequently, we collect 17,663 triplets for Edit-Align and 10,568 triplets for Edit-RL.

Prompt. 2: Generation of Desired Output Description

You are a precise and objective visual analyst. Your task is to provide a single, standalone, and very short description of an edited image. The description must be 60 words or less.

You will be given:
1. An Original Caption describing the original image.
2. An Editing Prompt describing the change.

Instructions:
1. Be Standalone: Imagine you are describing this final image to someone who has never seen the original. Your description must stand on its own and cannot refer to any previous state or editing process. Avoid using comparative or editing-related words like: 'changed', 'now', 'instead of', 'no longer', 'modified', 'edited', 'replaced', 'added', 'removed', 'unlike the original', 'previously'.
2. Be Faithful: Your description must be a factual account of the final image. Specifically describe the edited element as dictated by the Editing Prompt along with the other key visual elements.
3. Be Concise: Your entire response must be 60 words or less. Focus only on the most essential information.
4. Be Faithful: Do not invent details, objects, or attributes that are not visually present.

Please analyze the following inputs and generate your objective, faithful, and very short description of the final, edited image below.

Original Caption: {original_caption}
Editing Prompt: {editing_prompt}

Constructing conversations for instruction alignment. To perform Post-SFT training with *Edit Instruction Alignment*, we assemble the 17,663 data samples of condition images, edit instructions and output descriptions into instructional conversations, as displayed in Prompt. 4

B. Benchmarks and Evaluation Protocol

For image understanding, we include *(i)* general VQA benchmarks, such as GQA [29] and Seedbench [38], *(ii)* knowledge-based benchmarks, such as AI2D [35], ScienceQA [48],

Table A. Hyperparameter setup for different training stages of *UniGen-1.5*. Data ratio refers to the ratio of image understanding data, text understanding data, image generation data and image editing data.

Hyperparameters	PT	SFT	Edit-Inst-Align	RL
Learning rate	$1e^{-4}$	$2e^{-5}$	$1e^{-5}$	$3e^{-6}$
LR scheduler	constant	cosine	cosine	cosine
Gradient clip	1.0	1.0	1.0	1.0
Warm-up steps	0	5000	0	0
Training steps	300k	73k	0.5k	1.5k
Batch size	576	128	64	32
Data ratio	2:1:3:-	8:2:3:3	1:-:-:-	-:-:1:1

Table B. Training data overview of different stages. CC, SA, IMN, TI, SI, GIE stands for CC-3M [62], CC-12M [8], SAM-11M [36], ImageNet [60], Text-2-Image-2M [31], ShareGPT-4o-Image [9] and GPT-Image-Edit-1.5M [73], respectively.

Stage	Image Gen Data	Image Edit Data	Und Data	Text-only
Pre-training	(CC+SA+IMN) (Recap)	-	(CC+SA+IMN) (Recap)	RefinedWeb
Supervised Fine-tuning	BLIP-3o [10]+T2I [31]+SI [9]	SI [9]+GIE [73]	SF-LLaVA1.5 (Image Mixture) [88]	RefinedWeb
Edit-Inst-Align	-	-	Edit-Align	-
RL	T2I-R1 [32]	Edit-RL	-	-

Prompt. 3: Assessing Edit and Description Quality

{condition_image} You are a meticulous AI Data Quality Analyst. Your task is to score the sample on two criteria: overall description quality and editing quality.

1. Overall Description Quality Score (0-5):
This is a holistic score measuring the quality of the edited image description. A high score requires strength in all of the following areas: (1) The description must accurately reflect the edit requested in the Editing Prompt; (2) It must remain faithful to the unedited parts of the image, without imaginary details, subjective opinions, or hallucinations; (3) It must be a standalone description that does not compare edited elements to their original, unseen state.

2. Editing Quality Score (0-5):
This score evaluates the overall quality of the edit instruction and its outcome based on two factors: (1) the instruction should be plausible given the original image, and (2) it should clearly and unambiguously specify the content to be edited.

Your response must use the following format: "Reasoning: {A brief justification.};{Overall description quality score};{Editing quality score}"

Analyze the provided condition image, and the following data. Provide your evaluation in the specified single-line, semicolon-separated format.
Editing Instruction: {edit_instruction}
Edited Image Caption: {edited_caption}

Prompt. 4: Conversation in Edit Instruction Alignment

User: {condition_image} Analyze the image and the edit instruction: {edit_instruction}. Write a concise, accurate, and standalone caption describing the final edited image. Focus only on the result of the edit, without referring to or comparing with the original image.

Assistant: {output_description}

MMMU [95], and MathVista [49], and *(iii)* hallucination benchmarks, such as POPE [40]. We leverage the `lmms-eval` toolkit to compute the results for the above benchmarks.

For text-to-image generation benchmarks, we report results on GenEval [20] and DPG-bench [26] to comprehensively evaluate the semantic alignment between a text prompt and the generated images. We report our results using the official evaluation repository of GenEval and DPG-bench, respectively.
For image editing, we report results on ImgEdit [93] benchmark using the official evaluation repository. All results are evaluated with GPT-4o [30].

C. Training Details

C.1. Hyper-parameters and Datasets

An overview of training hyper-parameters is shown in Table A. We also list our training datasets for each training stage in Table B.

C.2. Reward Functions

Image-text Alignment Model. We employ DFN5B-CLIP-ViT-H-14 [17] to model the similarity between a given image and its text prompt. We feed the generated image and the output description into the visual encoder and the text encoder, respectively. Then, we compute the cosine similarity as $\mathbf{R}_C$ between the image and text embeddings for RL training.

Human Preference Model. We use HPSv2 [80] to evaluate the

Table C. Performance on image understanding benchmarks across different training stages. *Und Avg.* denotes the average score over all the benchmarks.

Model	AI2D	GQA	POPE	MMMU	MathVista	ScienceQA	Seedbench	Und Avg.
SFT	77.6	64.0	89.2	35.7	52.3	85.9	76.4	68.7
Edit Inst. Align	77.6	63.8	88.6	35.7	51.7	86.3	76.5	68.6
RL	77.4	63.7	88.3	35.9	51.9	86.3	76.5	68.6

aesthetic appeal and the alignment between the text description and the generated image. Similarly, we also obtain the cosine similarity between the visual and text features as the reward $\mathbf{R}_H$.

Semantic Consistency Model. We leverage the UnifiedReward-7B [74] model to measure the fine-grained consistency (*e.g.*, objects, attributes and relationship) between the text description and the output image. The model outputs a simple reasoning process followed by a final score ranging from 1-5, which is normalized to 0-1 as the reward $\mathbf{R}_U$.

Outcome Reward Model. We take advantage of ORM from [24] to judge whether the generated images correctly represent the given text description. The model is trained to output 'Yes' for aligned image-text pairs and yield 'No' otherwise. We compute the probability as the reward based on the model's first-token distribution. Given p_y denoting the probability of first token assigned to 'Yes' and p_n indicating the probability for 'No', we define the scalar reward as $\mathbf{R}_O = \frac{p_y}{p_y+p_n}$.

Ensemble of Reward Models. We simply average all reward scores by $\mathbf{R} = mean(\mathbf{R}_C + \mathbf{R}_H + \mathbf{R}_U + \mathbf{R}_O)$. We then perform the advantage computation based on the averaged reward score within each group.

D. More Results

D.1. Understanding Metrics across Training Stages

Although both *Edit Instruction Alignment* (Post-SFT) and RL are designed to improve image generation and editing, it is also valuable to examine how they affect *UniGen-1.5*'s image understanding capability. As shown in Table C, there is no performance drop after Post-SFT and RL, suggesting that these stages effectively maintain strong understanding capability while improving image generation and editing.

Table D. Ablation of condition designs for image editing. We report performance of *UniGen-1.5* after RL with different sequences of condition in image editing task. $\rightarrow$ denotes the order when concatenating different embeddings.

Condition Input	GenEval	DPG-Bench	ImgEdit
$\mathcal{X}_C^G \rightarrow \mathcal{X}_C^U \rightarrow \mathcal{T}_C$	0.89	87.47	4.05
$\mathcal{X}_C^U \rightarrow \mathcal{X}_C^G \rightarrow \mathcal{T}_C$	0.89	86.97	3.98
$\mathcal{X}_C^U \rightarrow \mathcal{T}_C \rightarrow \mathcal{X}_C^G$	0.89	86.83	4.31

D.2. Ablation of Condition Design in Image Editing

Inspired by prior works [9, 78], we utilize semantic ($\mathcal{X}_C^U$) and low-level ($\mathcal{X}_C^G$) visual embeddings as well as text embedding $\mathcal{T}_C$ as conditions for image editing. When constructing the representation of the conditions, the order of these embeddings is very important. We start with $\mathcal{X}_C^U \rightarrow \mathcal{T}_C$ since this arrangement aligns better with how our model perceives the tokens from visual and text modalities during pre-training, as compared to $\mathcal{T}_C \rightarrow \mathcal{X}_C^U$. Then, there are three options for inserting the $\mathcal{X}_C^G$ as shown in Table D. From the comparison results, we observe that appending $\mathcal{X}_C^G$ at the end introduces the best overall result. This is because during pre-training, *UniGen-1.5* is optimized with $\mathcal{X}_C^U \rightarrow \mathcal{T}_C$ for image understanding and $\mathcal{T}_C \rightarrow \mathcal{X}_C^G$ for image generation so that the design of $\mathcal{X}_C^U \rightarrow \mathcal{T}_C \rightarrow \mathcal{X}_C^G$ maximally leverages the training momentum.

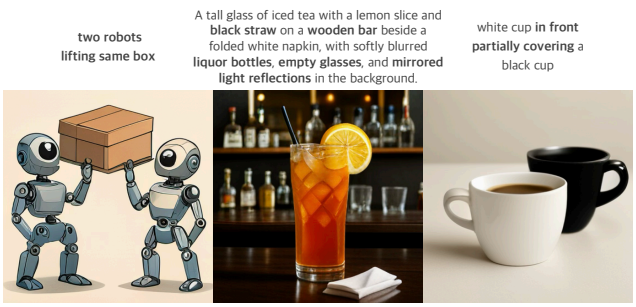

Figure A. Complex scenarios generated by *UniGen-1.5*.

Table E. **Ablation of reward models.** We train *UniGen-1.5* with unified RL using the same training recipe except for the different sets of reward models in each setting. UniR refers to Unified and DPG stands for the DPG-Bench. We report the overall score for GenEval, DPG-Bench ImgEdit benchmarks.

Reward model				GenEval	DPG.	ImgEdit
HPS	CLIP	ORM	UniR			
✓				0.76	87.22	4.21
✓	✓			0.85	87.13	4.28
✓	✓	✓		0.88	87.03	4.28
✓	✓	✓	✓	0.89	86.83	4.31

D.3. The Impact of Different Reward Models

Generally, using the ensemble of multiple vision experts strikes a better trade-off between text-to-image and image editing benchmarks. As shown in Table E, using UnifiedReward-7B slightly benefits the ImgEdit benchmark (row 4 vs. row 3). The removal of ORM leads to a drop of 0.03 in the overall score on GenEval (row 2 vs. row 3). Using HPSv2 alone underperforms the combination of HPSv2 and CLIP-H by 0.09 and 0.07 on GenEval and

Table F. Comparison with commercial models on ImgEdit benchmarks. The best and second-best results are highlighted in **bold** and underlined, respectively.

Model	Add	Adjust	Extract	Replace	Remove	Background	Style	Compose	Action	Overall
GPT Image 1 [High] [1]	4.61	4.33	2.90	4.35	3.66	**4.57**	**4.93**	3.96	4.89	4.20
UniGen-1.5	4.31	4.18	3.86	**4.78**	4.57	4.50	4.69	3.88	4.00	4.31
Gemini 2.5 Flash Image [21]	4.54	4.52	3.93	4.68	4.77	4.54	4.45	3.84	4.78	4.45
GPT Image 1.5 [High] [54]	**4.80**	**4.80**	**4.07**	**4.78**	4.74	4.70	4.87	**4.24**	**4.94**	**4.66**
Gemini 3 Pro Image Preview [22]	4.62	4.71	3.77	4.47	**4.87**	4.56	4.87	4.04	4.81	4.52

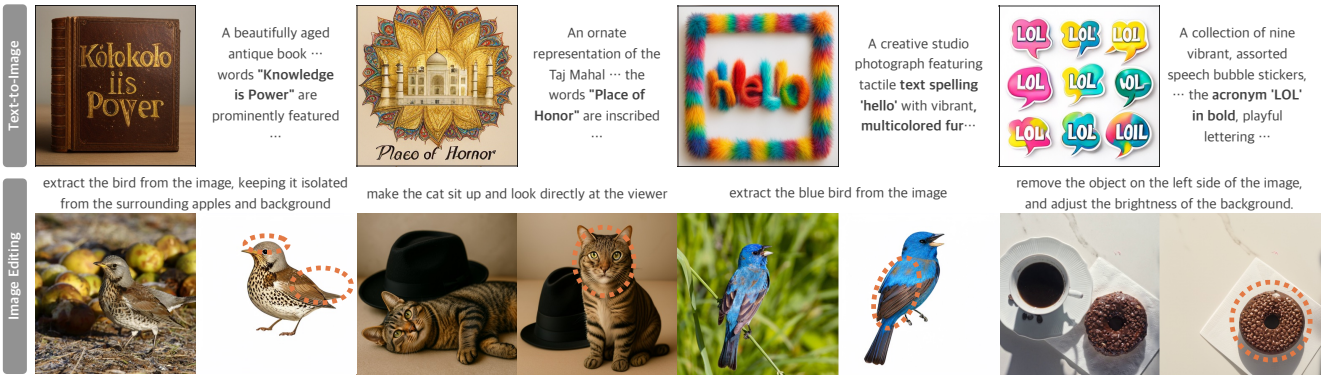

Figure B. Failure cases of *UniGen-1.5*.

ImgEdit benchmarks, respectively (row 2 vs. row 1).

D.4. Comparison with more Commercial Models

We present a comparison between *UniGen-1.5* and several recent commercial models on the ImgEdit benchmark in Tab. F. The results show that *UniGen-1.5* remains highly competitive overall, and performs particularly strongly in the "Replace" category, despite the fact that these commercial models are likely substantially larger and may benefit from proprietary data and greater computational resources.

D.5. More Visualizations

Complex Scenarios. We present several challenging image generation cases in Fig. A. The left example demonstrates the model's ability to render interactions between multiple entities. The middle example shows generation under a complex background, where *UniGen-1.5* correctly renders all required background objects. The right example illustrates an occlusion case, in which *UniGen-1.5* accurately captures the spatial relationships among the objects.

Failure Cases. We display limitations of *UniGen-1.5* in both text-to-image generation and image editing tasks in Figure B. In the first row, we present the instances where *UniGen-1.5* fails to accurately render text characters, as the light-weight discrete detokenizer struggles to control the fine-grained structural details required for text generation. In the second row, we display two examples with visible identity shifts highlighted by the circle, *e.g.*, the changes in the cat's facial fur texture and shape, the differences in the color of the bird's feathers, and the inconsistent surface texture of the doughnut. *UniGen-1.5* needs further improvement to address these limitations.