Appendix: Guiding Diffusion Models through Semantic Degradation

Anonymous Authors

A Token Importance Analysis

The method for calculating token importance described in this section is adapted from the Zero-TPrune model proposed by Wang et al. It is important to note that Zero-TPrune was originally designed for the **visual domain**, where it analyzes the attention graph to calculate the importance of **image** patches (also treated as tokens) for pruning Vision Transformers. In this study, we migrate and apply this methodology from the visual domain to the **textual domain**, aiming to quantify the semantic importance of each text token within a natural language context.

The method consists of two main components:

1. **Weighted PageRank (WPR) Algorithm**: Used to calculate token importance scores within a single attention head.
2. **Multi-Head Score Fusion Strategy**: Combines a Variance-based Head Filter (VHF) and Emphasizing Informative Region (EIR) aggregation to fuse scores from multiple heads.

A.1 Single-Head Importance via Weighted PageRank

The WPR algorithm treats the attention matrix A from a single attention head as the adjacency matrix of a weighted directed graph. It iteratively calculates the importance score of each node (i.e., token) in the graph. The importance of a token is determined by the weighted importance of other tokens that point to it. The calculation process is shown in Algorithm 1.

A.2 Multi-Head Score Fusion

For a multi-head attention mechanism, after obtaining an independent score distribution from each head via the WPR algorithm, these scores must be fused into a final score. A simple averaging of scores is not optimal, as different heads may focus on different semantic patterns. The method employs a fusion strategy that combines a Variance-based Head Filter (VHF) and Emphasizing Informative Region (EIR) aggregation.

1. **Variance-based Head Filter (VHF):** This technique is used to identify and filter out "bad" heads that provide uninformative or misleading score distributions. The procedure involves calculating the variance, Var_h, of the importance score distribution for each head. Only heads whose variance falls within a predefined threshold range $[v_{\min}, v_{\max}]$ are retained; the rest are discarded from subsequent calculations.

Algorithm 1 PageRank for Token Importance Scoring

Require: Attention matrix A, token length n
Require: Convergence threshold ϵ, maximum iterations T
Ensure: Token importance scores s

$A = \text{row\_norm}(A)$
Initialize uniform importance scores $s^{(0)} = \frac{1}{n}\mathbf{1}_n$
for iteration $t = 1$ to T **do**
  $s^{prev} = s^{(t-1)}$
  $s^{(t)} = A^T s^{prev}$
  $s^{(t)} = s^{(t)} / \|s^{(t)}\|_1$
  if $\|s^{(t)} - s^{prev}\|_1 < \epsilon$ **then**
    break
  end if
end for
return $s^{(t)}$

2. **Emphasizing Informative Region (EIR):** For the valid heads filtered by VHF, this method uses a "root-mean of sum of squares" approach for score aggregation. Compared to a direct average, the squaring operation amplifies high scores from any single head. This ensures that a token's high importance is not diluted in the final score, even if it is deemed critical by only a few "specialist" heads.

Final Score Calculation

Combining the VHF and EIR strategies, the final importance score $s^{(l)}(x_i)$ for the i-th token in the l-th layer is calculated as follows:

$$s^{(l)}\left(x_i\right) = \sqrt{\frac{\sum_{h=1}^{N_h} s^{(h,l)}\left(x_i\right)^2 \cdot \eta\left(v_{\min} \leq \mathrm{Var}_h \leq v_{\max}\right)}{\sum_{h=1}^{N_h} \eta\left(v_{\min} \leq \mathrm{Var}_h \leq v_{\max}\right)}}$$

Where:

- $s^{(h,l)}(x_i)$ is the importance score of token x_i from head h in layer l.
- Var_h is the variance of the score distribution for head h.
- $\eta(\cdot)$ is an indicator function that implements the VHF. The function returns 1 if the condition $v_{\min} \leq \mathrm{Var}_h \leq v_{\max}$ is met, and 0 otherwise.
- N_h is the total number of heads in the current layer.

A.3 Cross attention

A natural baseline for computing token importance is to leverage the cross-attention mechanism between text and image tokens. In diffusion models, the cross-attention matrix $C \in \mathbb{R}^{N_{\text{img}} \times N_{\text{text}}}$ is computed where image tokens serve as queries and text tokens serve as keys. For each text token i, its attention weights

across all image positions form the i-th column of C. These weights are typically reshaped to the spatial dimensions of the latent to produce saliency maps.

Following this intuition, one might compute the importance score of text token i by simply summing its corresponding column:

$$s_i^{\text{cross}} = \sum_{j=1}^{N_{\text{img}}} C_{j,i} \tag{1}$$

However, this approach yields counterintuitive results. Consider the prompt "A man is cooking, Minecraft Style." As shown in Figure 1, the cross-attention-based importance scores assign the highest values predominantly to padding tokens, rather than to semantically meaningful tokens like "man", "cooking", or "Minecraft". This paradoxical behavior contradicts our intuition and renders cross-attention unsuitable for token importance estimation.

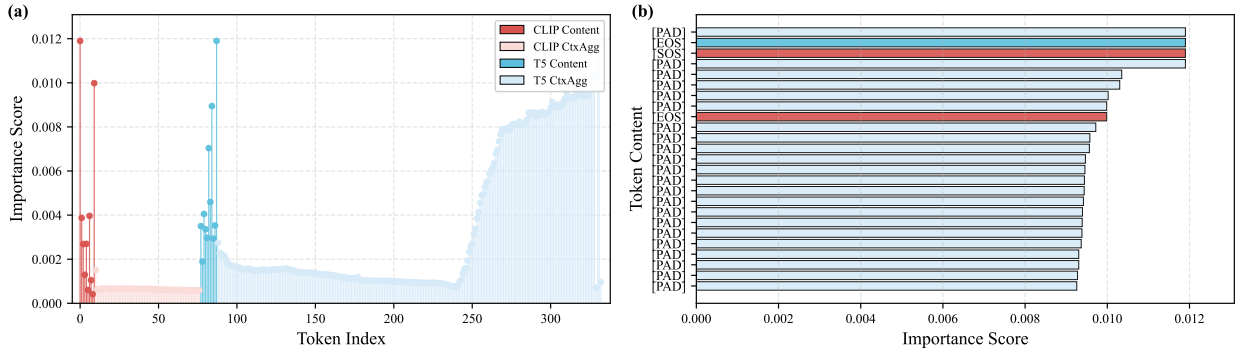


Figure 1: Token importance scores computed via cross-attention summation for the prompt "A man is cooking, Minecraft Style." The method incorrectly assigns highest importance to padding tokens rather than semantic content tokens.

This failure motivates our graph-based approach using self-attention and Weighted PageRank, which properly captures token relationships and avoids the padding token bias inherent in cross-attention aggregation.

B Implementation Details

This section provides the complete algorithmic and implementation details of our Condition-Degradation Guidance (CDG). We present: (1) the full sampling algorithm that integrates CDG into the diffusion denoising loop, and (2) the core PyTorch implementation for token degradation.

B.1 Algorithm: CDG Sampling

Algorithm 2 details the complete CDG sampling procedure, which augments the standard diffusion sampling loop with our semantic degradation mechanism. The algorithm consists of three key components:

CalculateImportance (lines 2–7) computes token importance scores using the Weighted PageRank (WPR) algorithm on the self-attention graph. It extracts the attention affinity matrix $\boldsymbol{A}$ from transformer block

λ_{block}, applies WPR and multi-head fusion, and returns sorted token indices $\boldsymbol{i}_{\text{sorted}}$ in descending order of importance. For efficiency, this is computed only once at $t = T$ and cached for all subsequent steps.

ProcessToken (lines 9–11) constructs the degraded condition $\boldsymbol{c}_{\text{deg}}$ at each timestep. Using the cached importance ranking and degradation ratios $r_{\text{deg}} = \{r_{\text{content}}, r_{\text{CtxAgg}}\}$, it builds a binary mask $\boldsymbol{m}$ that marks less important tokens for replacement, then performs masked interpolation: $\boldsymbol{c}_{\text{deg}} \leftarrow \boldsymbol{m} \odot \boldsymbol{c} + (1 - \boldsymbol{m}) \odot \boldsymbol{c}_{\emptyset}$.

Main Sampling Loop (lines 13–22) performs guided denoising for T steps. At each timestep t, it computes two noise predictions—$\boldsymbol{\epsilon}_{\text{cond}}$ with full condition $\boldsymbol{c}$ and $\boldsymbol{\epsilon}_{\text{deg}}$ with degraded condition $\boldsymbol{c}_{\text{deg}}$—then combines them via the CDG formula: $\hat{\boldsymbol{\epsilon}} \leftarrow \boldsymbol{\epsilon}_{\text{cond}} + (w - 1) \cdot (\boldsymbol{\epsilon}_{\text{cond}} - \boldsymbol{\epsilon}_{\text{deg}})$, where w is the guidance scale.

Algorithm 2 Sampling with Condition-Degradation Guidance (CDG)

Require: Initial noise $\boldsymbol{x}_T \sim \mathcal{N}(0, \boldsymbol{I})$, text embedding $\boldsymbol{c}$, empty embedding $\boldsymbol{c}_{\emptyset}$, guidance scale w, denoising steps T, denoiser D_θ, transformer block index λ_{block}, degradation ratio $r_{\text{deg}} = \{r_{\text{content}}, r_{\text{CtxAgg}}\}$.
Ensure: Generated image $\boldsymbol{x}_0$.

1: **CalculateImportance**($\boldsymbol{c}, D_\theta, \lambda_{\text{block}}$):
2: $\quad \boldsymbol{Q}, \boldsymbol{K} \leftarrow \text{EXTRACTQK}(D_\theta, \lambda_{\text{block}}, \boldsymbol{c})$
3: $\quad \boldsymbol{A} \leftarrow \boldsymbol{Q}\boldsymbol{K}^T$
4: $\quad \boldsymbol{s}_{\text{heads}} \leftarrow \text{WPR}(\boldsymbol{A})$
5: $\quad \boldsymbol{s} \leftarrow \text{SCOREFUSION}(\boldsymbol{s}_{\text{heads}})$
6: $\quad \boldsymbol{i}_{\text{sorted}} \leftarrow \text{ARGSORT}(\boldsymbol{s})$
7: **return** $\boldsymbol{i}_{\text{sorted}}$
8: **ProcessToken**($\boldsymbol{c}, \boldsymbol{c}_{\emptyset}, \boldsymbol{i}_{\text{sorted}}, r_{\text{deg}}$):
9: $\quad \boldsymbol{m} \leftarrow \text{BuildMask}(\boldsymbol{i}_{\text{sorted}}, r_{\text{deg}})$
10: $\quad \boldsymbol{c}_{\text{deg}} \leftarrow \boldsymbol{m} \odot \boldsymbol{c} + (1 - \boldsymbol{m}) \odot \boldsymbol{c}_{\emptyset}$
11: **return** $\boldsymbol{c}_{\text{deg}}$
12: **// Main sampling process**
13: **for** $t = T, T - 1, \ldots, 1$ **do**
14: $\quad$ **if** $t = T$ **then**
15: $\quad\quad \boldsymbol{i}_{\text{sorted}} \leftarrow \text{CALCULATEIMPORTANCE}(\boldsymbol{c}, D_\theta, \lambda_{\text{block}})$
16: $\quad$ **end if**
17: $\quad \boldsymbol{c}_{\text{deg}} \leftarrow \text{PROCESSTOKEN}(\boldsymbol{c}, \boldsymbol{c}_{\emptyset}, \boldsymbol{i}_{\text{sorted}}, r_{\text{deg}})$
18: $\quad \boldsymbol{\epsilon}_{\text{cond}} \leftarrow D_\theta(\boldsymbol{x}_t, t, \boldsymbol{c})$
19: $\quad \boldsymbol{\epsilon}_{\text{deg}} \leftarrow D_\theta(\boldsymbol{x}_t, t, \boldsymbol{c}_{\text{deg}})$
20: $\quad \hat{\boldsymbol{\epsilon}} \leftarrow \boldsymbol{\epsilon}_{\text{cond}} + (w - 1) \cdot (\boldsymbol{\epsilon}_{\text{cond}} - \boldsymbol{\epsilon}_{\text{deg}})$
21: $\quad \boldsymbol{x}_{t-1} \leftarrow \text{SCHEDULERSTEP}(\boldsymbol{x}_t, \hat{\boldsymbol{\epsilon}}, t)$
22: **end for**
23: **return** $\boldsymbol{x}_0$

B.2 Core Implementation

The code listing below presents the PyTorch implementation of the token degradation logic (corresponding to the `ProcessToken` function in Algorithm 2). The implementation includes three key optimizations: (1) early exit when no degradation is needed (lines 107–108), (2) importance caching controlled by

all_use_first_step_importance flag (lines 111–119), and (3) efficient masked interpolation via broadcasting (lines 128–130). This compact implementation demonstrates how CDG integrates seamlessly into existing diffusion pipelines with minimal code overhead.

```
def __call__(self, positive_encoder_hidden_states, negative_encoder_hidden_states)
    :
    """Process encoder hidden states based on importance and degrade ratios"""

    degrade_ratio = self.process_params["degrade_ratio"]

    # Optimization: Skip importance calculation if not needed
    if degrade_ratio in [{"content": 1, "CtxAgg": 0}]:
        sorted_indices = [i for i in range(len(positive_encoder_hidden_states))
    ]

    # Use cached importance from first step (if enabled)
    elif (self.process_params.get("all_use_first_step_importance")
          and self.first_step_importance is not None):
        sorted_indices = self.first_step_importance

    # Calculate importance (first step or every step depending on config)
    else:
        sorted_indices, scores = self.importance_calculator(positive_encoder_
    hidden_states, negative_encoder_hidden_states)
        if self.process_params.get("all_use_first_step_importance"):
            self.first_step_importance = sorted_indices  # Cache for later

    # Generate keep mask based on importance and ratios
    degrade_mask, degraded_indices = self.get_degrade_mask(
        sorted_indices,
        ratio=degrade_ratio
    )

    # Interpolate: degraded = (1 - degrade_mask) * positive + degrade_mask *
    negative
    degrade_mask = degrade_mask.unsqueeze(0).unsqueeze(-1)  # Expand dimensions
    result = ((1 - degrade_mask) * positive_encoder_hidden_states +
              degrade_mask * negative_encoder_hidden_states)

    return result
```

C Experimental Details and Supplementary Results

This section provides comprehensive details for all experimental components referenced in the main paper, including evaluation metrics implementation, optimal hyperparameters for different models, and additional experimental results.

C.1 Evaluation Metrics Implementation

This section provides detailed implementation specifications for all evaluation metrics used in our experiments to ensure full reproducibility.

C.1.1 Fréchet Inception Distance (FID).

FID is a standard metric for assessing the quality of generated images by measuring the distributional distance to real images. To eliminate ambiguity, we specify that all our FID calculations are performed using the official implementation within the **`torchmetrics`** library. Following standard practice, we extract the 2048-dimensional features from the final pooling layer of the pre-trained Inception-v3 network.

C.1.2 CLIP Score.

The CLIP Score evaluates the semantic consistency between a generated image and its corresponding text prompt. Different CLIP model variants can yield varying scores. To ensure fair and reproducible comparisons, we consistently use the `openai/clip-vit-base-patch32` model as the backbone for all CLIP Score calculations.

C.1.3 Aesthetic Score.

The Aesthetic Score is a metric designed to computationally estimate the visual appeal of an image, simulating human perception of beauty. The score is generated by a dedicated predictor model, which typically consists of a regressor built upon a powerful, pre-trained vision foundation model like a CLIP image encoder. This predictor is trained on a large-scale dataset where images are paired with aesthetic ratings collected from human volunteers. This process allows the model to learn the visual features that correlate with positive human aesthetic judgment. In our work, we employ the model provided within the `aesthetic_predictor_v2_5` library. A higher score from this model indicates a higher predicted visual appeal.

C.1.4 VQA Score.

The VQA (Visual Question Answering) Score assesses the factual alignment between a prompt and a generated image by framing the evaluation as a comprehension task. The process involves three steps: First, the descriptive prompt is programmatically converted into a closed-form (yes/no) question. Second, a pre-trained VQA model is presented with both the generated image and this question. Finally, the VQA Score is determined by the model's output probability for the affirmative answer, "Yes". A high score signifies high confidence from the VQA model that the image accurately depicts the content specified in the original prompt. For this evaluation, our implementation is based on the `t2v_metrics` framework, and we utilize the `clip-flant5-xxl` model as the VQA engine. The question template is fixed as "Does this figure show '[prompt]'? Please answer yes or no."

C.2 Dataset

In this study, we employed two core benchmark datasets for model evaluation: the COCO 2017 validation set and GenAI Bench.

C.2.1 COCO 2017 Validation Set

This is an authoritative dataset widely used in the field of computer vision. It contains 5,000 images, each associated with five textual descriptions, totaling 25,000 captions. For consistency in our experiments, we uniformly selected the first caption for each image as the prompt for the generation task.

C.2.2 GenAI Bench

This is a novel dataset focused on evaluating the compositional reasoning capabilities of models. Its core consists of 1,600 high-quality text prompts sourced from the real-world workflows of professional designers. The dataset is specifically designed to test the performance of AI models in understanding and executing text-to-vision tasks that involve complex logic and compositional relationships, such as multiple objects, fine-grained attributes, spatial relations, and advanced reasoning.

C.2.3 Rationale for Selection

The rationale for selecting these two datasets is that they collectively form a **complementary and comprehensive evaluation framework**. On one hand, the descriptive prompts from COCO 2017, which primarily depict common objects and general scenes, are ideal for measuring a model's **foundational capabilities and generalizability** in generating **common objects and conventional scenes**. On the other hand, GenAI Bench presents a more rigorous challenge; its complex, compositional prompts allow for an in-depth probe of a model's **upper limits in advanced semantic understanding, logical reasoning, and precise detail control**. By combining these datasets, our study can not only assess the model's baseline performance on conventional tasks but also conduct a more nuanced analysis of its strengths and weaknesses in handling complex, fine-grained instructions, thereby enabling a more holistic and multi-faceted evaluation of its overall capabilities.

C.3 Optimal Hyperparameters for Different Models

This section details the parameters used for the CFG, CADS, ICG, and CDG methods across different models, as summarized in Table 1. The guidance scale (w) and the number of steps (T) were set based on default values or official examples. The parameter s for CADS, which controls the intensity of guidance noise, was adopted from the value used for DiT models as specified in its paper's appendix. The seed for ICG (set to 42) initializes a random number generator used to select a random token ID for each prompt in the COCO dataset, ensuring reproducibility. For our CDG method, λ_{block} denotes the layer index where degradation is applied, while r_{content} and r_{CtxAgg} represent the degradation ratios for content and context-aggregating tokens, respectively.

It is important to note that for the FLUX.1-dev model, the guidance scale w corresponds to the `true_cfg_scale` parameter in its codebase, and we used an empty string for the negative prompt to implement the CFG method. Furthermore, while FLUX.1-dev is capable of generating high-quality, high-resolution images, its

Model	Approach	w	T	s	seed	σ	λ_{block}	$r_{\text{content}}\|r_{\text{CtxAgg}}$
SD3	CFG	7	28	-	-	-	-	-
	CADS	7	28	0.07	-	-	-	-
	ICG	7	28	-	42	-	-	-
	PAG	3	28	-	-	-	13	-
	SEG	3	28	-	-	5	13	-
	CDG	7	28	-	-	-	1	1.0|0.0
SD3.5	CFG	3.5	28	-	-	-	-	-
	CADS	3.5	28	0.07	-	-	-	-
	ICG	3.5	28	-	42	-	-	-
	PAG	3	28	-	-	-	13	-
	SEG	3	28	-	-	5	13	-
	CDG	3.5	28	-	-	-	2	1.0|0.0
FLUX.1	CFG	1.5	28	-	-	-	-	-
	CADS	1.5	28	0.07	-	-	-	-
	ICG	1.5	28	-	42	-	-	-
	CDG	1.5	28	-	-	-	1	1.0|0.0

Table 1: Optimal hyperparameters for different models. ("-" denotes not used)

inference time is considerable. Due to computational constraints, we reduced the generation resolution for the FLUX model from 1024×1024 to 512×512 and decreased the number of inference steps from 50 to 28.

Additionally, it is important to clarify the interpretation of the guidance scale parameter w in Table 1. For traditional guidance methods (CFG, CADS, ICG, CDG), w directly represents the guidance strength. However, for PAG and SEG methods, which support a combined `cfg+pag` mode, the reported w value corresponds to the PAG/SEG guidance scale, while the CFG scale is set to 1.0 (i.e., CFG is effectively disabled). This configuration allows PAG and SEG to operate independently without interference from CFG, which is the recommended setting in their respective implementations.

C.4 Hyperparameters analysis

While Section 6.3.2 of the main text presents a comprehensive hyperparameter analysis on SD3, establishing $R_{\text{deg}} = 1.0$ as the optimal default, this section provides complementary ablation studies on SD3.5 and FLUX.1 to validate the cross-model generality of this choice. Our goal is to demonstrate that $R_{\text{deg}} = 1.0$ serves as a robust initialization across diverse architectures, not merely a model-specific optimum.

Experimental Setup. For each model, we first conducted a preliminary search over intervention block indices (λ_{block}) to identify the most promising configuration. Based on these initial experiments, we selected $\lambda_{\text{block}} = 2$ for SD3.5 and $\lambda_{\text{block}} = 1$ for FLUX.1. We then performed ratio ablation at these fixed indices, systematically varying R_{deg} to observe metric responses.

Due to computational constraints, these supplementary ablations were conducted on a subset of 500 images from the MS-COCO validation set, rather than the full 5,000 images used in the main SD3 analysis. This design choice is justified by two considerations: (1) the primary conclusion regarding $R_{\text{deg}} = 1.0$ was already established through the comprehensive SD3 analysis on 5,000 images, and (2) the 500-image subset provides sufficient statistical power to observe trend patterns and validate consistency across models. The convergence of results across different sample sizes further supports the robustness of our findings.

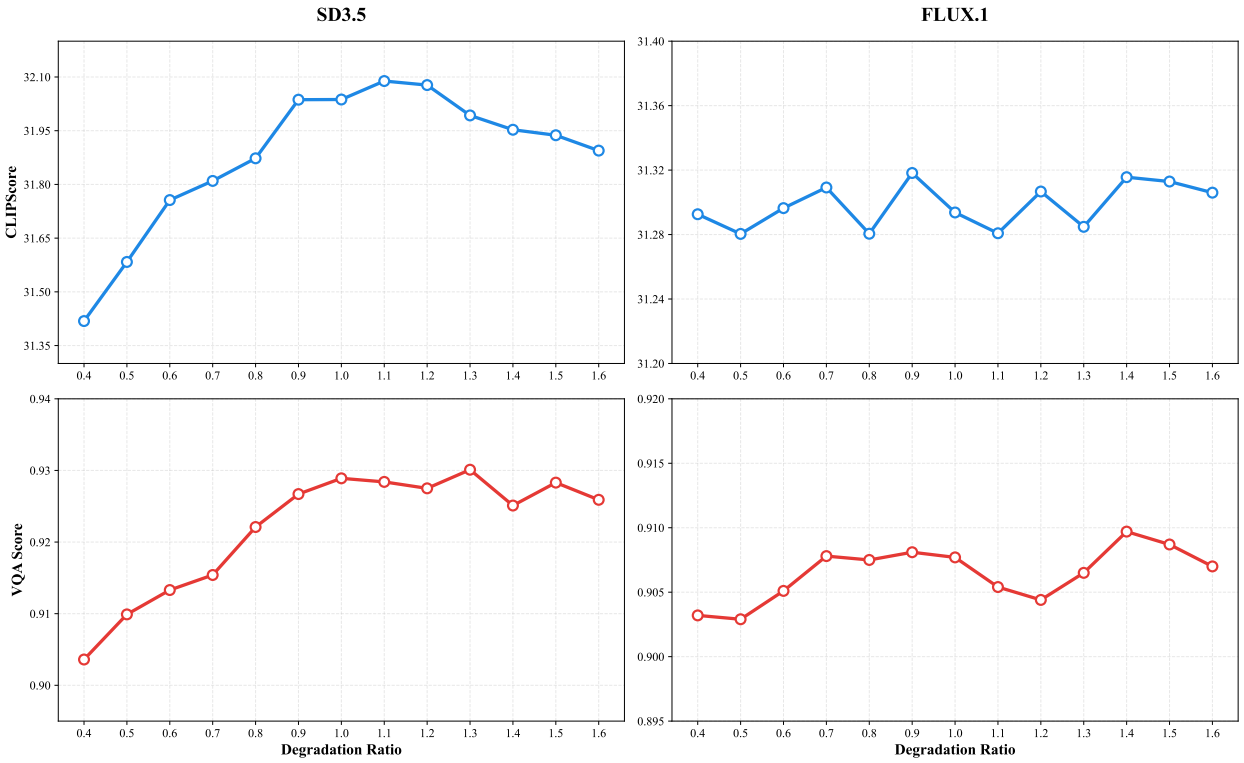


Figure 2: **Hyperparameter ablation on SD3.5 and FLUX.1:** Performance metrics (CLIP Score and VQA Score) as a function of Degradation Ratio R_{deg}. Top row: CLIP Score; bottom row: VQA Score. Left column: SD3.5 results; right column: FLUX.1 results. For SD3.5, both metrics achieve their best performance around $R_{\text{deg}} = 1.0$-1.3, exhibiting a stable plateau that is consistent with the SD3 analysis in the main text. For FLUX.1, metrics exhibit relative stability across the tested range, with $R_{\text{deg}} = 1.0$ providing a reliable initialization point.

Results and Analysis. Figure 2 presents the ablation results for both models across CLIP Score and VQA Score metrics. For SD3.5 (left column), the patterns closely mirror those observed for SD3 in the main text: both metrics achieve their best performance in the region around $R_{\text{deg}} = 1.0$, with CLIP Score reaching its maximum near $R_{\text{deg}} = 1.1$ and VQA Score achieving optimal values around $R_{\text{deg}} = 1.0$-1.3. Notably, the performance plateau in the $[1.0, 1.3]$ range demonstrates robustness to small variations in context-aggregating token degradation. The asymmetric response pattern—steeper sensitivity in $[0, 1.0]$ (content token removal) transitioning to gentler slopes in $[1.0, 2.0]$ (context-aggregating token removal)—confirms the content/context-aggregating dichotomy holds across SD3 variants.

For FLUX.1 (right column), the curves exhibit notably flatter profiles with reduced sensitivity to R_{deg} variations. This behavior aligns with FLUX.1's training paradigm: as discussed in Section 6.2, FLUX.1 employs *Guidance Distillation*, making it inherently less dependent on inference-time guidance mechanisms. Consequently, the model shows more stable performance across different degradation ratios. Despite this relative flatness, $R_{\text{deg}} = 1.0$ remains a sensible initialization choice, as it falls within the stable, high-performing region for both metrics and maintains consistency with the SD3/SD3.5 configuration.

Conclusion. These cross-model ablations validate the generality of $R_{\text{deg}} = 1.0$ as a default initialization. While SD3.5 demonstrates the same optimal behavior as SD3, FLUX.1's relative insensitivity to ratio variations further supports the robustness of this choice: even when the model is less guidance-dependent,

$R_{\text{deg}} = 1.0$ provides reliable performance without requiring extensive per-model tuning. This cross-architectural consistency, combined with the computational efficiency benefits at $R_{\text{deg}} = 1.0$ (as discussed in Section 6.3.4), establishes $R_{\text{deg}} = 1.0$ as a principled default for CDG across diverse diffusion model architectures.

C.5 Visual Examples of Varying Degradation Ratios

Figure 3 visually demonstrates the impact of the unified degradation ratio R_{deg} on the generated results. It can be observed that when fine-tuning around the default configuration of $R_{\text{deg}} = 1.0$, the model consistently generates images that adhere to the core elements of the prompt, yet they exhibit diverse variations in visual features such as composition and perspective. This indicates that R_{deg} not only ensures the alignment of key content but also serves as an effective control for users to flexibly adjust according to specific aesthetic preferences, allowing them to explore different generative styles.

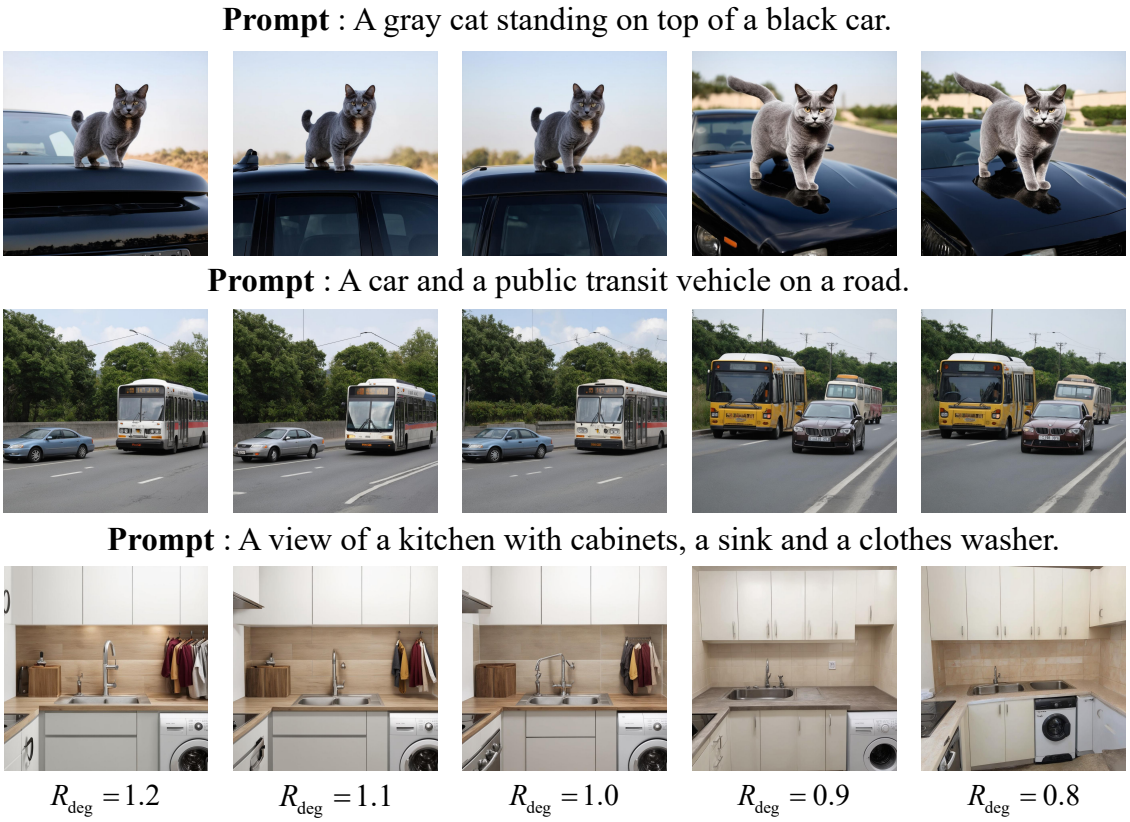


Figure 3: Visual comparison of results generated by varying the degradation ratio R_{deg}.

C.6 Additional Details for CFG* Analysis

This section provides detailed experimental setup and additional qualitative results for the CFG* analysis presented in Section 6.3.3 of the main text.

C.6.1 Detailed Experimental Setup.

As discussed in the main text, CFG* replaces the positive prompt $\boldsymbol{c}$ in standard CFG (Eq. (3)) with the degraded condition $\boldsymbol{c}_{\text{deg}}$. For completeness, we provide the explicit formulation and implementation details below.

Standard CFG formulation:

$$\hat{D}_\theta(\boldsymbol{x}_t, t) = D_\theta(\boldsymbol{x}_t, t, \boldsymbol{c}) + (w-1)(D_\theta(\boldsymbol{x}_t, t, \boldsymbol{c}) - D_\theta(\boldsymbol{x}_t, t, \varnothing))$$

CFG* formulation (for semantic probing):

$$\hat{D}_\theta(\boldsymbol{x}_t, t) = D_\theta(\boldsymbol{x}_t, t, \boldsymbol{c}_{\text{deg}}) + (w-1)(D_\theta(\boldsymbol{x}_t, t, \boldsymbol{c}_{\text{deg}}) - D_\theta(\boldsymbol{x}_t, t, \varnothing))$$

where $\boldsymbol{c}_{\text{deg}}$ is constructed according to Eq. (11) in the main text with varying degradation ratios R_{deg}.

C.6.2 Additional Qualitative Results.

To provide additional visual evidence for the semantic properties of degraded conditions, Figure 4 shows generation results guided purely by context-aggregating token information (i.e., $r_{\text{content}} = 1, r_{\text{CtxAgg}} = 0$). We compare unconditional generation (left) with context-aggregating-token-only guidance (right) across four diverse prompts. These results illustrate that context-aggregating tokens, after contextual encoding by the text encoder, carry global semantic information related to the original prompt.

C.7 Full Benchmark Results on GenAI-Bench

This section provides comprehensive quantitative results of our CDG method compared to baselines (CFG, CADS, ICG, SEG, and PAG) on the GenAI-Bench dataset, covering both advanced reasoning skills and basic skills. All metrics are reported with 2 decimal places. For each skill dimension, the best-performing method is marked in **bold**, while the second-best is underlined.

C.7.1 Advanced Reasoning Skills

Table 2 presents detailed evaluation results for advanced reasoning skills across three state-of-the-art text-to-image diffusion models: SD3, SD3.5, and FLUX.1. These skills encompass five key reasoning capabilities: Counting, Comparison, Differentiation, Negation, and Universal reasoning. Our CDG method demonstrates consistent improvements over baseline methods across most skill dimensions.

C.7.2 Basic Skills

Table 3 presents detailed evaluation results for basic compositional skills across SD3, SD3.5, and FLUX.1. These fundamental skills include: Action Relation (capturing action-object interactions), Attribute (object properties), Scene (environmental context), Spatial Relation (object positioning), and Part Relation (part-whole relationships). Our method shows consistent performance gains across all models.

Model	Method	Counting	Comparison	Differentiation	Negation	Universal	Avg
SD3	CFG	76.00	74.08	77.22	46.58	70.74	68.92
	CADS	75.59	74.05	76.98	46.74	70.11	68.70
	ICG	75.37	73.10	75.89	47.46	69.39	68.24
	SEG	70.87	69.43	70.87	**47.90**	64.96	64.81
	PAG	67.76	66.40	67.09	47.84	64.80	62.78
	CDG (Ours)	**76.50**	**74.86**	**78.26**	46.24	**71.53**	**69.48**
SD3.5	CFG	76.45	73.70	75.10	46.36	72.21	68.76
	CADS	76.27	73.54	75.08	46.63	71.94	68.69
	ICG	76.41	74.13	75.63	46.85	70.23	68.65
	SEG	71.98	71.43	72.80	46.53	67.73	66.09
	PAG	72.79	71.02	72.38	**48.68**	66.17	66.21
	CDG (Ours)	**77.92**	**76.06**	**78.74**	46.65	**73.13**	**70.50**
FLUX.1	CFG	**75.18**	72.97	75.39	**45.51**	71.13	68.04
	CADS	74.84	72.58	74.87	45.42	70.81	67.70
	ICG	74.57	72.83	74.50	44.65	71.47	67.60
	CDG (Ours)	74.98	**73.47**	**76.17**	44.97	**71.55**	**68.23**

Table 2: Comprehensive GenAI-Bench evaluation results for advanced reasoning skills. For each skill dimension within each model, the **best** method is shown in bold and the second-best is underlined. CDG achieves the best performance in most dimensions, demonstrating its superior reasoning capabilities.

C.8 Encoder Ablation and WPR Analysis

This section provides two supplementary ablations referenced in the main text: encoder-specific degradation analysis and WPR vs. random ranking comparison.

C.8.1 Encoder-Specific Ablation

In SD3, CLIP-L and CLIP-G are concatenated along the feature dimension (fused and inseparable), then concatenated with T5 along the sequence length dimension. Our method operates on this combined embedding, degrading CLIP and T5 portions independently. Table 4 shows that degrading only CLIP tokens (CDG-C) yields the best FID and CLIP Score, while degrading only T5 tokens (CDG-T) provides competitive results. The full CDG (both encoders) achieves the best balance across all metrics, confirming that joint degradation is optimal.

C.8.2 WPR vs. Random Ranking

At the default $R_{\text{deg}} = 1.0$, all content tokens are degraded regardless of ranking, making WPR unnecessary. For $R_{\text{deg}} \neq 1.0$, where only a subset of tokens is degraded, we compare WPR-based ranking against random ranking. Table 5 shows results at $R_{\text{deg}} = 0.9$ and $R_{\text{deg}} = 1.1$.

* Means computed on COCO-5K. Std ($\pm$): variance across 20 seeds on a 400-prompt subset (FID std omitted).

Mean performance is comparable, but WPR provides two key benefits: (1) **Stability**: random ranking introduces variance (e.g., VQA ± 1.57 at $R_{\text{deg}} = 0.9$), while WPR ensures deterministic behavior. (2) **Theoretical grounding**: the content/context-aggregating dichotomy revealed by WPR validates $R_{\text{deg}} = 1.0$ as a principled semantic boundary, providing the analytical foundation for stratified degradation.

Model	Method	Action Relation	Attribute	Scene	Spatial Relation	Part Relation	Avg
SD3	CFG	79.06	77.23	76.67	78.66	76.35	77.59
	CADS	78.84	77.05	76.51	78.55	**76.36**	77.46
	ICG	77.95	76.42	75.77	78.09	74.96	76.64
	SEG	72.57	71.98	71.28	73.50	70.99	72.06
	PAG	68.58	68.35	66.95	69.65	68.75	68.46
	CDG (Ours)	**79.29**	**77.56**	**77.18**	**79.84**	76.19	**78.01**
SD3.5	CFG	79.48	78.00	77.45	79.66	76.31	78.18
	CADS	79.39	77.91	77.25	79.58	76.30	78.09
	ICG	79.08	77.83	76.87	78.90	76.28	77.79
	SEG	76.02	74.42	74.09	76.34	73.21	74.82
	PAG	75.53	74.10	73.47	75.64	73.04	74.35
	CDG (Ours)	**81.18**	**79.19**	**78.18**	**80.69**	**78.08**	**79.46**
FLUX.1	CFG	77.33	76.03	76.09	77.47	74.78	76.34
	CADS	77.05	75.71	75.92	77.42	74.33	76.09
	ICG	77.10	75.52	75.87	77.07	74.65	76.04
	CDG (Ours)	**77.70**	**76.27**	**76.37**	**77.56**	**74.93**	**76.57**

Table 3: Comprehensive GenAI-Bench evaluation results for basic compositional skills. For each skill dimension within each model, the **best** method is shown in bold and the second-best is underlined. CDG consistently outperforms baseline methods across different models and skill types.

Table 4: Encoder-specific ablation on SD3. -C/-T: CLIP/T5 only degradation.

Method	FID↓	CLIP↑	Aes↑	VQA↑
CFG	35.69	31.73	5.66	91.44
CDG-C (CLIP only)	**32.68**	**32.01**	5.65	92.32
CDG-T (T5 only)	34.81	31.94	5.67	92.11
CDG (Both)	34.05	32.00	**5.70**	**92.40**

C.9 More Visual Results

This section presents additional visual results demonstrating the effectiveness and versatility of our CDG method. We first showcase CDG's compatibility with orthogonal techniques, followed by comprehensive qualitative comparisons across different models.

C.9.1 Combination with Orthogonal Methods.

The synergy of CDG with orthogonal methods like PAG is shown in Figure 5. In the sculpture example, PAG alone fails to correctly render the word "Freedom" on the base, producing a garbled text "FRE(nn)". With CDG, the semantic constraint ensures accurate text generation, displaying the complete and correct "FREEDOM" inscription. Similarly, for the food photography prompt, PAG alone struggles to capture the style specification, producing an image with distracting background clutter. The addition of CDG provides the necessary semantic precision, removing this clutter and results in a cleaner, more professional composition.

Table 5: WPR vs. Random ranking ablation on SD3.

R_{deg}	FID↓		CLIP↑		Aes↑		VQA↑	
	WPR	Rand*	WPR	Rand*	WPR	Rand*	WPR	Rand*
1.1	33.80	34.17	31.98	$32.02_{\pm 0.52}$	5.68	$5.68_{\pm 0.16}$	92.27	$92.27_{\pm 0.88}$
0.9	35.08	33.73	31.72	$31.90_{\pm 0.95}$	5.67	$5.67_{\pm 0.28}$	91.05	$91.65_{\pm 1.57}$

C.9.2 Application to Text-Guided Image Editing.

CDG's semantic precision is particularly beneficial for text-guided image editing tasks. As shown in Figure 6, in the convertible car example, the baseline struggles with the color attribute, generating a red or pink car instead of the specified "blue". CDG corrects this semantic error, producing the accurate retro blue convertible as described in the prompt. Similarly, in the golden retriever scene, while the baseline method preserves the overall image layout, it fails to interpret a key numerical constraint in the prompt ("three children"). In contrast, CDG successfully enforces this semantic detail, rendering the correct number of subjects without disrupting the composition.

C.9.3 Application to Controllable Generation.

As shown in Figure 7, in the furniture scene, ControlNet alone fails to follow both the structural layout and color attributes. The baseline incorrectly places the coffee mug on top of the books, whereas the canny figure shows it should be positioned to the right of the books. Additionally, the colors are wrong ("blue book, green book, red coffee mug" are not rendered correctly). With CDG, the model correctly positions all objects according to the structural constraint and renders them with their specified colors. Similarly, in the racing car example, while ControlNet successfully enforces structural constraints (from a canny figure), it struggles with complex combinatorial semantics (e.g., "green front, black rear" and "77"). By providing a more precise semantic signal, CDG enables the model to faithfully render these details, demonstrating a powerful synergy between structural and semantic control.

C.9.4 More Visual Results on SD3, SD3.5, and FLUX.1

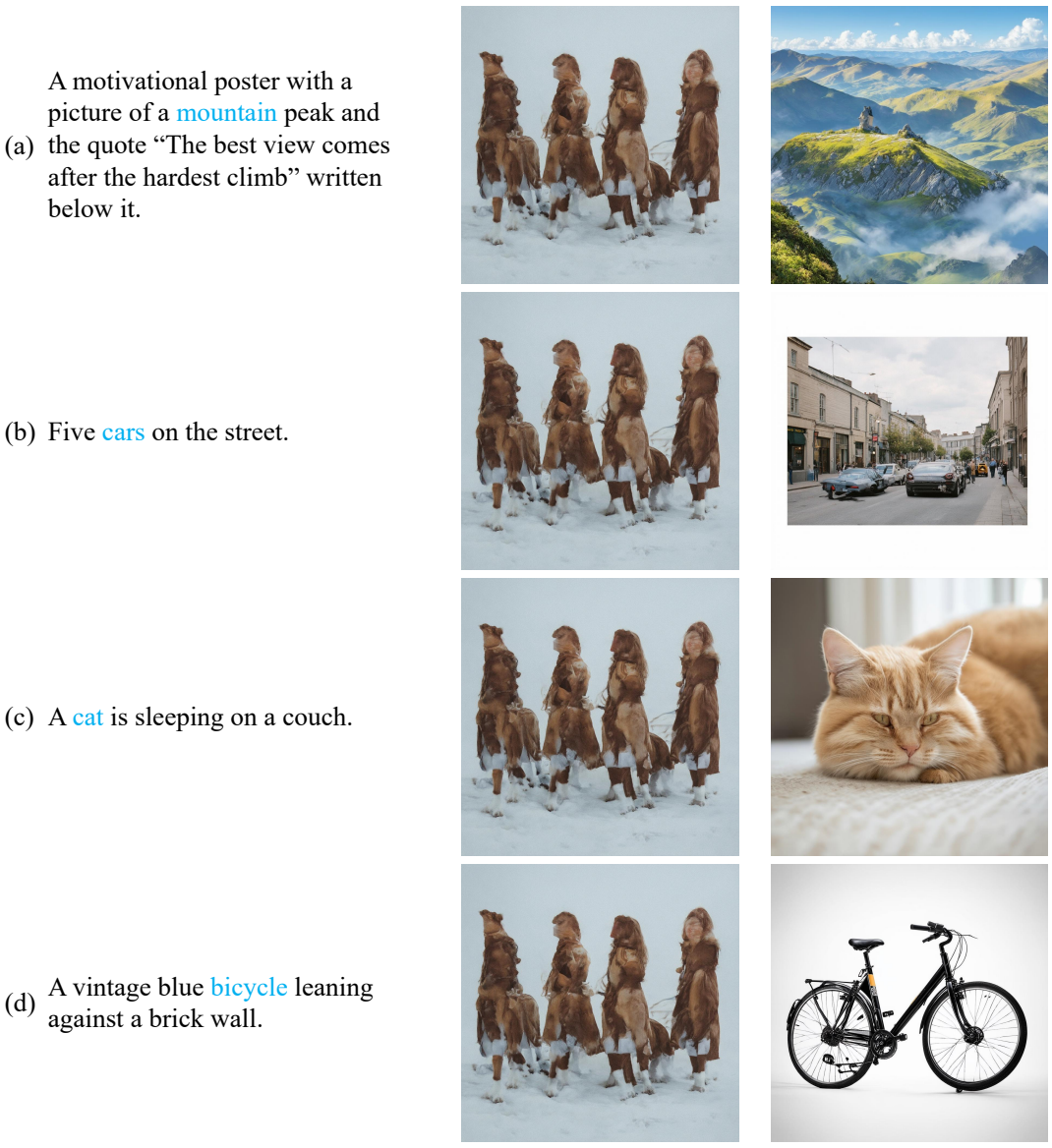
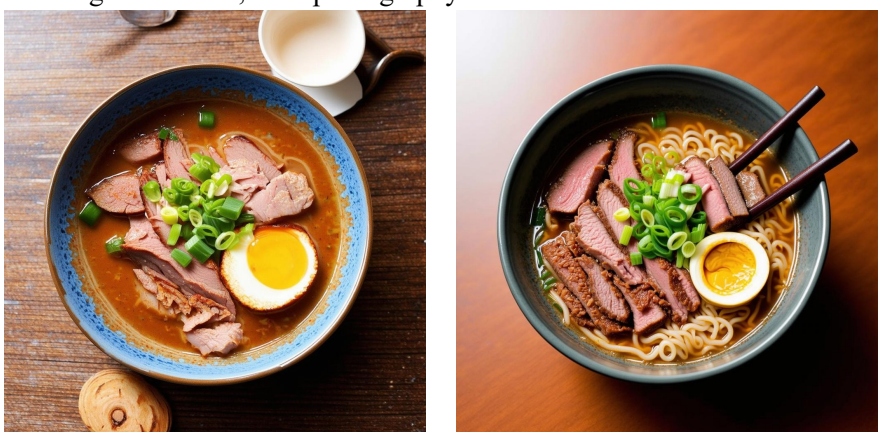


Figure 4: Comparison between unconditional generation (left) and context-aggregating-token-only guidance (right) for four prompts. The right column uses CFG* with $\boldsymbol{c}_{\text{deg}}$ constructed by $r_{\text{content}} = 1, r_{\text{CtxAgg}} = 0$.

Prompt : A sculpture with the words 'Freedom' engraved on the base is near a group of children playing.

Prompt : A bowl of steaming hot ramen with pork, egg, and green onions, food photography.



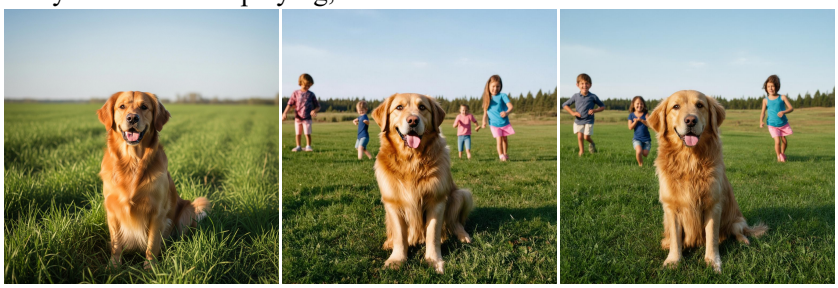
(a) PAG (b) PAG + CDG

Figure 5: Synergy with PAG: CDG provides the core semantic foundation, while PAG refines local details.

Prompt : A retro blue convertible drives on an asphalt road.

(a) 

Prompt : A golden retriever was sitting on the grass, surrounded by three children playing, and in the distance was a forest.

(b) 

(a) Initial Figure (b) Baseline (c) Baseline + CDG

Figure 6: CDG for Text-Guided Editing: Correcting semantic errors where the baseline fails.

Prompt : The scene features a wooden armchair and a small round metal table. The top is made of glass, with a blue book, a green book, and a red coffee mug on it.

(a) 

Prompt : A two-color racing car drives through a damp city street at night, its front is green and the rear is black, and the number "77" is written on the door. A blue neon sign on the side of the road reads "DREAM".

(b) 

(a) Canny Figure (b) Baseline (c) Baseline + CDG

Figure 7: CDG for Controllable Generation: Providing semantic precision to render complex details under structural constraints.

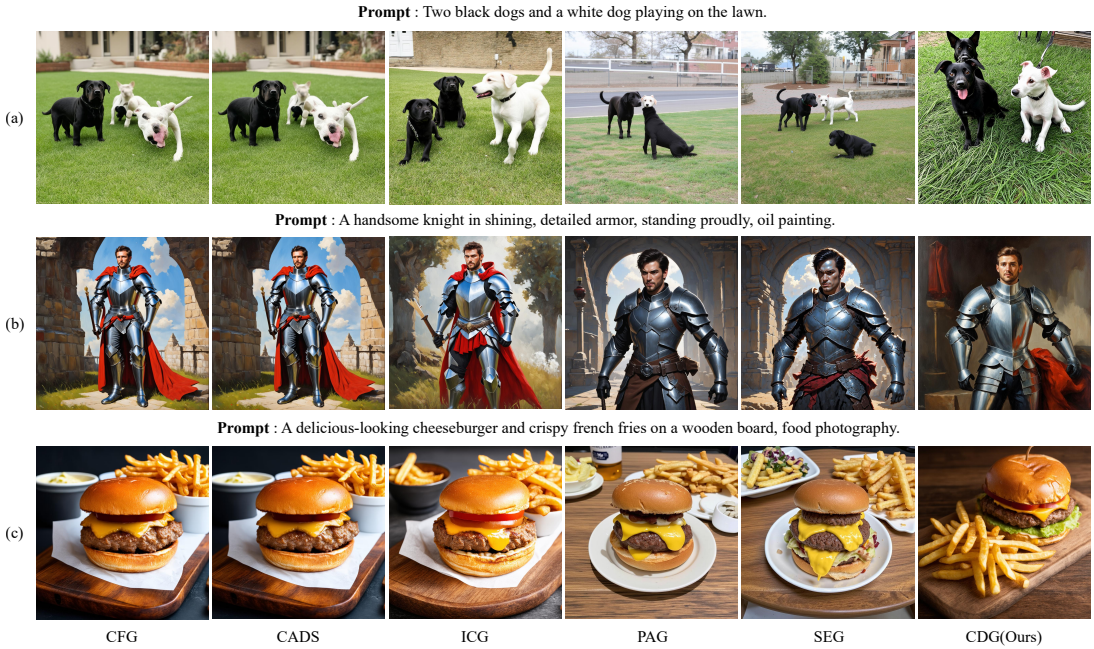


Figure 8: More visual results on the SD 3 model.

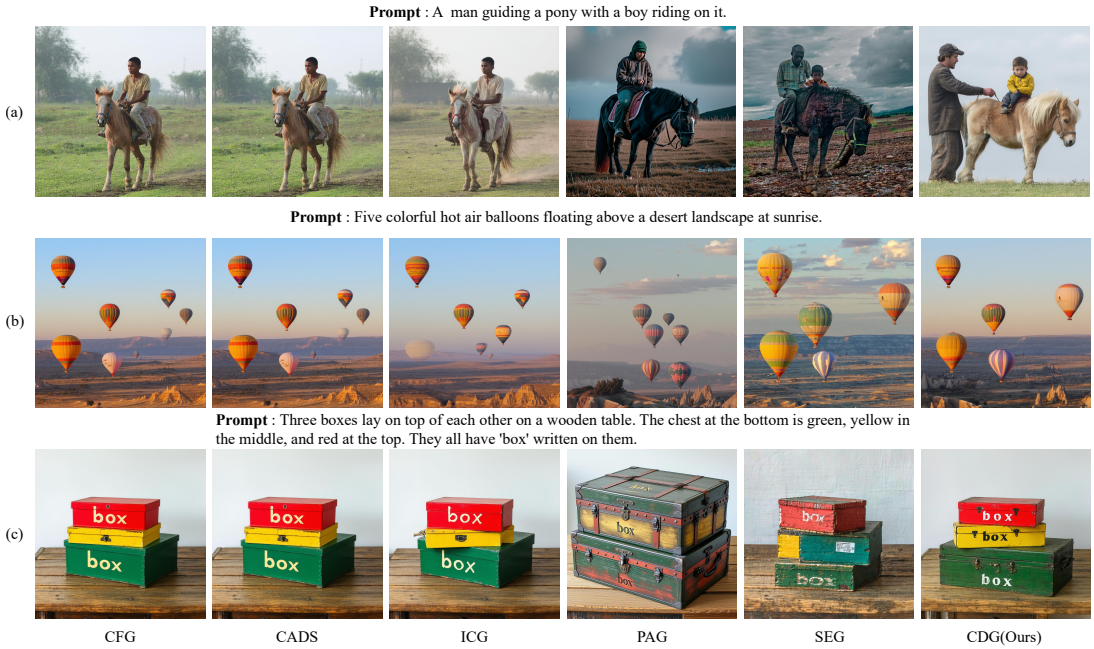


Figure 9: More visual results on the SD 3.5 model.

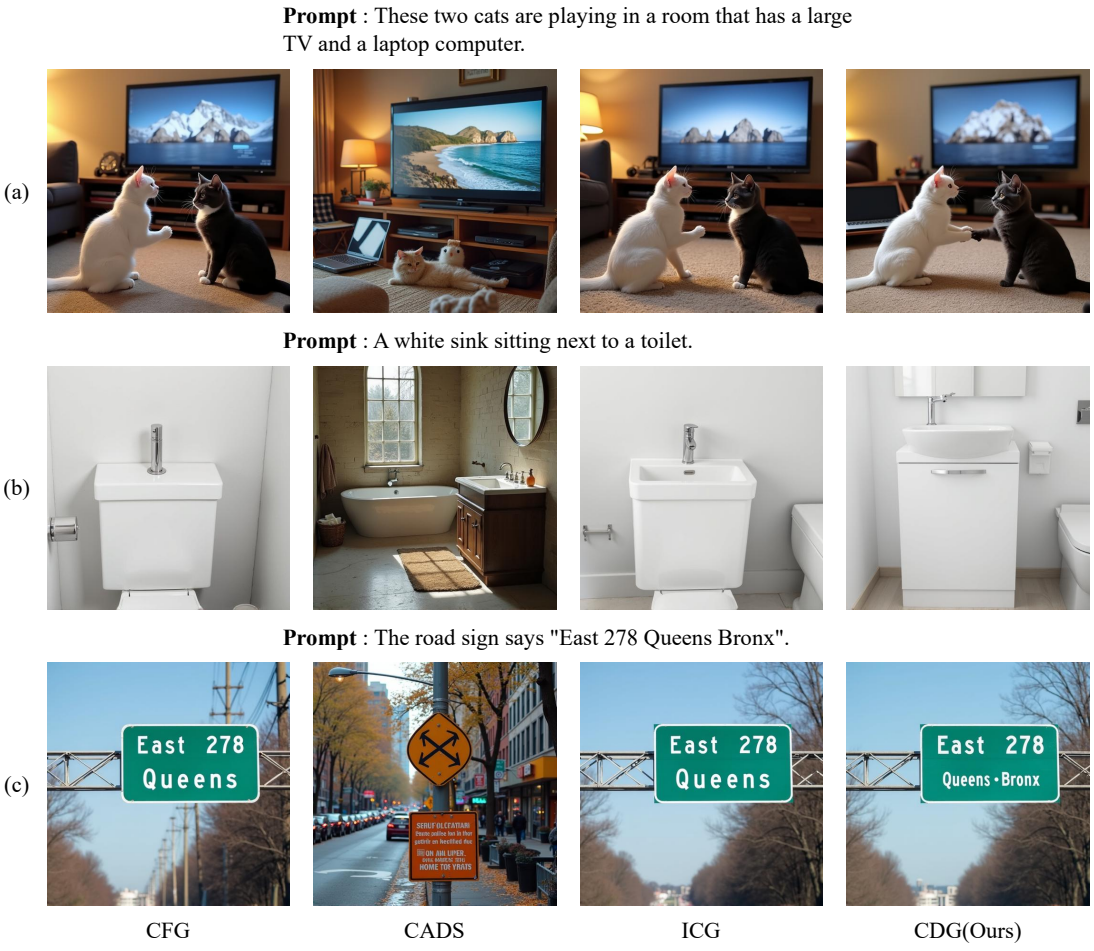


Figure 10: More visual results on the Flux model.