

APPENDIX

In Appendix [A](#) we discuss the broader impacts of our work in downstream uses. In Appendix [B](#) we note our use of LLMs as assistant. Appendix [C](#) details the algorithm for pre-inference alignment of the Latent Modifier Network (LMN). Appendix [D](#) includes further implementation details, including the LMN architecture (Appendix [D.1](#)), hyperparameters used for the core *D2D* formulation (Appendix [D.2](#)) and optimization (Appendix [D.3](#)), and extension of *D2D* to multi-object scenarios (Appendix [D.4](#)). Appendix [E](#) shows the performance improvements that result from using *D2D* to complement existing baselines. Appendix [F](#) shows a breakdown of numeracy by total density on D2D-Multi. Appendix [G](#) shows CountGD’s baseline counting performance, relative to other regression/detector-based counters. Additional qualitative results are illustrated in Appendix [H](#). Appendix [I](#) includes results from additional ablation experiments, including studies on the value of mixing weight w used to mix the LMN output with the original noise (Appendix [I.1](#)); the necessity of inference-time calibration of the LMN (Appendix [I.2](#)); regularization term variants (Appendix [I.3](#)); variants of the LMN architecture (Appendix [I.4](#)); and the necessity of pre-inference alignment of the M_ϕ (Appendix [I.5](#)). Appendix [J](#) details methods to preprocess prompts for selective activation of the *D2D* method. Appendix [K](#) compares alternative formulations of the *D2D* loss. Appendix [L](#) shows a sample numeracy optimization trajectory for a single prompt, compare the raw values of the *D2D* loss \mathcal{L}_{D2D} and f over iterations. Appendix [M](#) demonstrates an extension of the *D2D* critic for multi-step diffusion models, using an existing multi-step-oriented initial noise optimization framework. Appendix [N](#) presents analysis on the impact of *D2D* on the diversity of generated images. Appendix [O](#) presents extended evaluation of numeracy on newer/larger distilled models.

A BROADER IMPACTS

As a text-to-image pipeline, our method has many practical downstream uses, so it is necessary to exercise caution in deployment. Our method offers the advantage of stronger numeracy, which may be desirable in applications where users need to generate specific counts of objects. Our model may inherit biases of pre-trained diffusion base models and detectors. We suggest using strict NSFW filters and building more robust detectors.

B USE OF LARGE LANGUAGE MODELS (LLMs)

We used LLMs to help with word choice for clarity and debugging.

C ALGORITHMS

Algorithm 2 Pre-inference alignment stage (done once per model)

Input: Latent Modifier Network M_ϕ , initialized with random weights; latent dimension d ; weight w ; learning rate η , loss weight λ .

Output: Pre-trained M_ϕ with output aligned to Gaussian distribution.

Set seed = 1.

repeat 100 **times**

 Sample $\mathbf{x}_T \in \mathbb{R}^d \sim \mathcal{N}(0, \mathbf{I})$

for $1 \leq epoch \leq 200$ **do**

$\mathbf{x}'_T = w \cdot \mathbf{x}_T + (1 - w) \cdot M_\phi(\mathbf{x}_T)$

 Compute $\mathcal{L} = \lambda \mathcal{L}'_{\text{reg}}$

$\phi \leftarrow \phi - \eta \nabla \mathcal{L}$

end for

D IMPLEMENTATION DETAILS

D.1 ARCHITECTURE OF THE LATENT MODIFIER NETWORK

The LMN is a three-layer perceptron, with input/output layers of size d (dimension of initial latent, $\mathbf{x}_T \in \mathbb{R}^d$), and two hidden layers of size 100 each. On base models SDXL-Turbo, SD-Turbo, and Pixart-DMD, the LMN has 3,303,384 tunable parameters.

D.2 CORE $D2D$ HYPERPARAMETERS

The core $D2D$ hyperparameters are the detector threshold τ and steepness coefficient β , which determine the transition threshold and curvature of the sigmoid underlying \mathcal{L}_{D2D} , and the mixing weight w , which determines the ratio with which to combine the LMN output and original noise to obtain the optimal noise. We use $\tau = 0.2$, $\beta = 300$, and $w = 0.2$. Studies on hyperparameters τ and β are reported in Tab. 3 in Sec. 4.3 of the main text. Experiments on w are reported in Tab. 8 in Appendix I.1.

D.3 OPTIMIZATION

During pre-inference alignment and inference-time calibration, when we optimize only the regularization term, we use $\mathcal{L}'_{\text{reg}}$. At inference-time, during numeracy optimization, we use \mathcal{L}_{reg} in conjunction with \mathcal{L}_{D2D} . Appendix I.3 reports ablation studies on using $\mathcal{L}'_{\text{reg}}$ vs. \mathcal{L}_{reg} during numeracy optimization.

In the pre-inference alignment stage, we use learning rate $\eta = 10^{-4}$ and loss weight $\lambda = 0.01$. At inference-time (stage 1: calibration), we use minimum number of iterations $t_{\min} = 70$; “good enough” threshold for determining when calibration is done $\tau_{\text{reg}} = -712.8$, learning rate $\eta_{\text{calib}} = 10^{-3}$, and loss weight $\lambda_{\text{calib}} = 0.01$. During stage 2 (numeracy optimization), we use learning rate $\eta = 5 \times 10^{-4}$ and loss weights $\alpha = 5$ and $\lambda = 10^{-4}$. During this stage, we use loss-based, adaptive learning rate scheduling of η to ease steady convergence when the generated count approximates the requested count; we also adaptively rescale (i.e., increase) λ to counteract larger deviations from Gaussian (i.e., divergence).

For $D2D$ experimental results with Pixart-DMD on benchmark $D2D$ -Large, we use the base detector OWLv2, instead of f_{β, τ_x} , to perform early-stopping. For single-object scenarios, we set $K = 200$, except for $D2D$ with Pixart-DMD on $D2D$ -Large, for which we use $K = 400$. For the multi-object scenario, we use $K = 400$.

D.4 MULTI-OBJECT \mathcal{L}_{D2D}

In extending $D2D$ to prompts with $m > 1$ object classes $\{C_j, 1 \leq j \leq m\}$, we have to note that every predicted bbox B_i now comes with m corresponding scores $z_i^{(1)}, z_i^{(2)}, \dots, z_i^{(m)}$, the max of which, z_i^{\max} , indicates its corresponding label. Only the bboxes where $z_i^{\max} > \tau_z$ are counted. To correct over/under-generation, our approach then is to focus on appropriately adjusting the largest score z_i^{\max} of each bbox, while minimizing all other (i.e. non-max) scores. This means minimizing the z_i^{\max} of bboxes that correspond to over-generated classes and maximizing the z_i^{\max} of those that correspond to under/correctly generated classes (to increase/maintain the count). We use roughly the same formulation as Equation 2, with z_i replaced by the specific per-class, per-bbox logits z_i^j that need to be minimized or maximized (Equation 6).

If \mathbb{C}_{\geq} is the set of classes that are under-generated or correctly generated, then

$$\mathbb{S}_{\text{bbox}} = \{(i, j) | 1 \leq i \leq n \wedge j = (\arg \max_{j'} z_i^{j'}) \in \mathbb{C}_{\geq}\}$$

refers to the set of the bboxes corresponding to the under/correctly generated classes, along with their max logits. We compute the loss \mathcal{L}_{D2D} like so:

$$\mathcal{L}_{D2D}^{\text{multi}} = \sum_{(i,j) \in S} -\sigma(-\beta \cdot (z_i^j - \tau_z)) \cdot (z_i^j - \tau_z) + \sum_{(i,j) \notin S} \sigma(\beta \cdot (z_i^j - \tau_z)) \cdot (z_i^j - \tau_z) \quad (6)$$

E D2D YIELDS BOOSTS IN NUMERACY, IN COMPLEMENT WITH BASELINES

Table 6: **D2D additionally yields improvements, in combination with T2I counting/enhancement baselines.** Standard deviations indicate the significance of our findings. Avg. over four seeds.

Base Model	Method	CoCoCount	D2D-Small	D2D-Multi	D2D-Large
SDXL-Turbo	TokenOpt (Zafar et al., 2024)	35.12±.75	23.31±1.66	—	03.94±.72
	TokenOpt (Zafar et al., 2024) + $D2D$	48.75 ±2.40	34.00 ±1.70	—	08.56 ±1.28
	ReNO (Eyring et al., 2024)	41.88±1.03	27.50±.68	5.31±.38	4.69±1.25
	ReNO (Eyring et al., 2024) + $D2D$	54.38 ±2.17	41.25 ±2.45	9.69 ±.38	10.38 ±.72
SD-Turbo	ReNO (Eyring et al., 2024)	43.38±3.47	32.06±.99	8.94±1.76	4.25±1.14
	ReNO (Eyring et al., 2024) + $D2D$	53.62 ±2.02	42.44 ±2.82	11.62 ±1.30	10.19 ±1.77
Pixart-DMD	ReNO (Eyring et al., 2024)	44.75±1.44	37.25±1.70	9.44±.75	4.75±.74
	ReNO (Eyring et al., 2024) + $D2D$	52.50 ±2.27	40.62 ±2.37	12.00 ±.68	8.12 ±.92

F DENSITY-BASED DIFFERENTIAL PERFORMANCE PERSISTS IN D2D-MULTI

Table 7: **Quantitative results.** *D2D* yields higher performance on low total-density prompts than high total-density ones. Avg. over four seeds.

Method	D2D-Multi	
	Low Total Density	High Total Density
SDXL (Podell et al., 2024)	3.08	0.50
Make It Count (Binyamin et al., 2025)	—	—
SDXL-Turbo (Sauer et al., 2025)	2.75	0.25
ReNO (Eyring et al., 2024)	6.67	1.25
TokenOpt	—	—
<i>D2D</i> w/ OWLv2 (Ours)	12.08	3.00
<i>D2D</i> w/ YOLOv9 (Ours)	7.83	1.50
SD2.1 (Rombach et al., 2022)	6.17	0.75
SD1.4 (Rombach et al., 2022)	3.75	0.00
Counting Guidance (Kang et al., 2025)	4.42	0.25
SD-Turbo (Rombach et al., 2022)	3.00	1.25
ReNO (Eyring et al., 2024)	11.58	1.00
<i>D2D</i> w/ OWLv2 (Ours)	13.42	2.75
Pixart- α (Rombach et al., 2022)	1.67	0.25
Pixart-DMD (Chen et al., 2025b)	8.00	1.00
ReNO (Eyring et al., 2024)	12.25	1.00
<i>D2D</i> w/ OWLv2 (Ours)	17.17	1.75

G COUNTGD IN COMPARISON WITH OTHER COUNTERS

CountGD performs well in low and high-density settings.

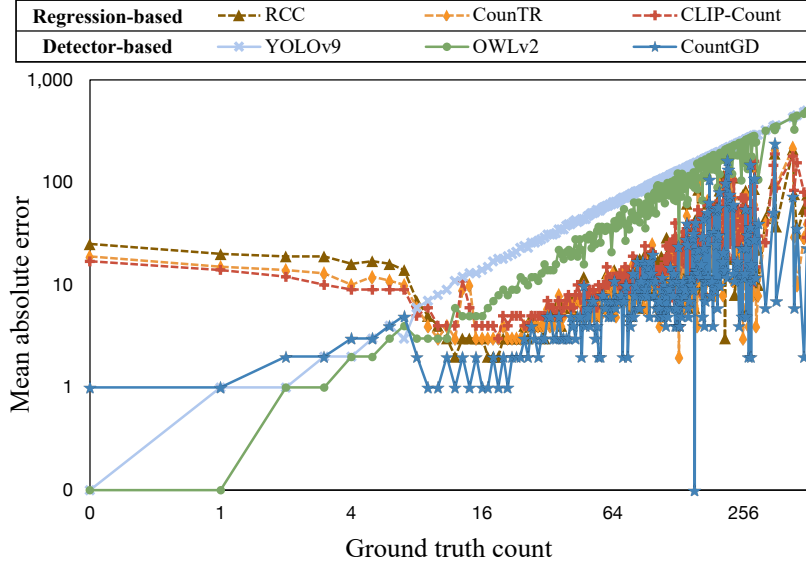


Figure 7: **CountGD** is a state-of-the-art counter, based on GroundingDINO.

H ADDITIONAL QUALITATIVE RESULTS

Additional qualitative results on prompts from benchmarks CoCoCount (Binyamin et al., 2025) and LMD (Lian et al., 2024) shown.



Figure 8: Qualitative examples from count correction methods on base model SDXL-Turbo, along with SDXL.

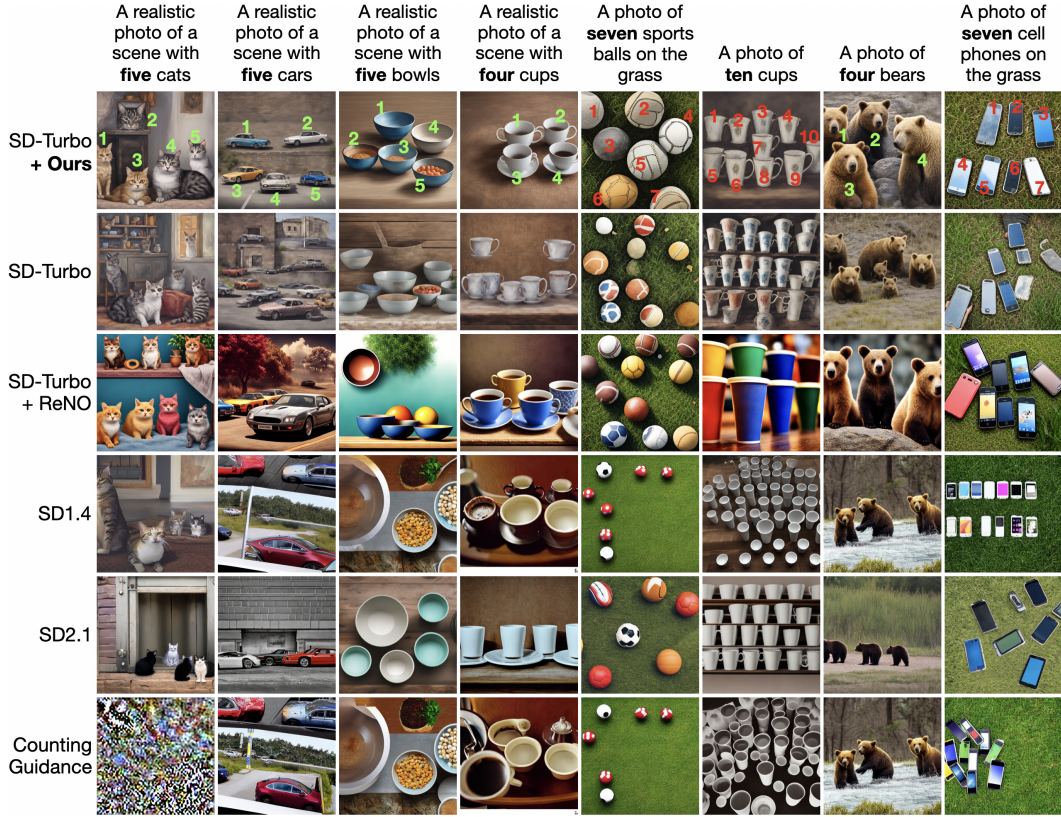


Figure 9: Qualitative examples from count correction methods on base model SD-Turbo, along with SD1.4, SD2.1, and Counting Guidance.

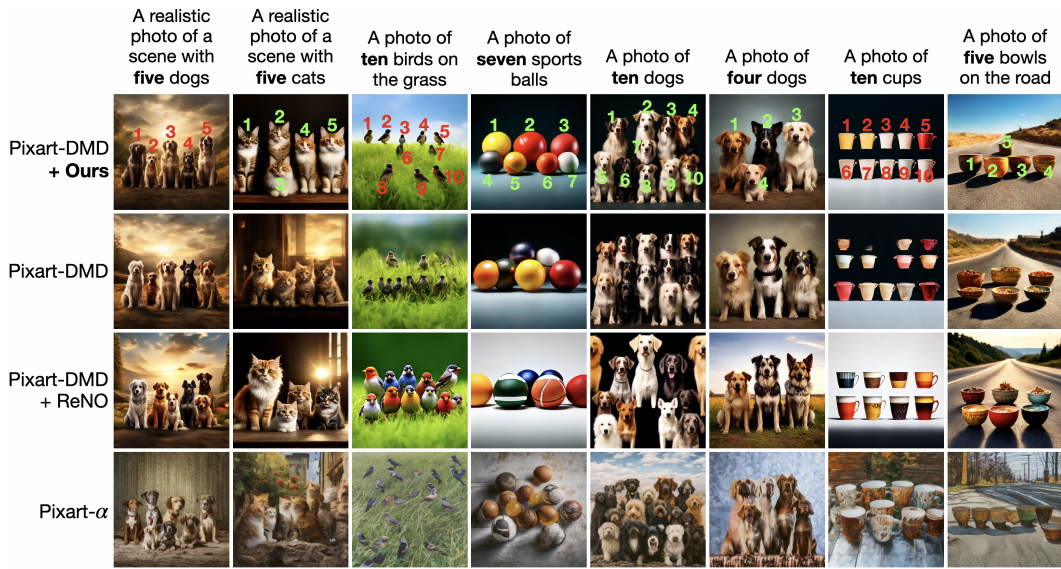


Figure 10: Qualitative examples from count correction methods on base model Pixart-DMD, along with Pixart- α .

I ADDITIONAL ABLATIONS

I.1 $w = 0.2$ IS OPTIMAL FOR NUMERACY AND QUALITY

w is the weight used to determine how to mix the LMN output with the original noise. $w = 0$ is numeracy-wise the best, followed by $w = 0.2$, but results in patchy visual artifacts, relative to $w = 0.2$ (Fig. 11). Hence, the hyperparameter value we use in our main experiments is $w = 0.2$.

Table 8: In terms of just numeracy, $w = 0$ is optimal, followed by $w = 0.2$. Base model SDXL-Turbo, on CoCoCount. Seed = 0.

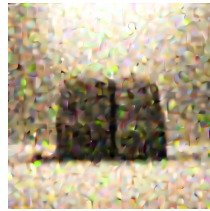
	w			
	0.0	0.2	0.5	0.8
CountGD	62.50	55.50	37.50	48.00



Figure 11: $w = 0$ results in patchy visual artifacts, relative to $w = 0.2$. So, we use $w = 0.2$.

I.2 INFERENCE-TIME CALIBRATION OF M_ϕ

At inference-time, given a new \mathbf{x}_T , we allow a few iterations to calibrate M_ϕ 's output to the Gaussian distribution. Fig. 12 shows alignment is crucial to maintaining high image quality.



(a) Without alignment.



(b) With alignment.

Figure 12: Inference-time alignment of P_ϕ is necessary. Prompt: "A realistic photo of a scene with one backpack."

I.3 REGULARIZATION

Our regularization term (Eq. 8) is a variant of the one ReNO uses (Eq. 7) that penalizes larger deviations from Gaussian more and smaller deviations less, to allow enough flexibility in the vicinity of the initial distribution to accommodate updates specified by the count critic. We scale $\mathcal{L}'_{\text{reg}}$ by coefficient a and shift by constant c , which we take to the power of 10.

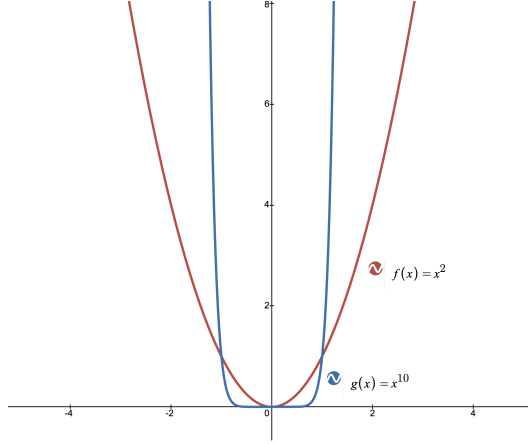


Figure 13: Exponentiating by the power of 10 sharpens the gradients far away from the minimum while flattening the gradient closer to the minimum, effectively allowing the optimization process to focus more on numeracy optimization when the latent is already well-regularized.

$$\mathcal{L}'_{\text{reg}} = \frac{\|\mathbf{x}'_T\|^2}{2} - (d-1) \cdot \log(\|\mathbf{x}'_T\|), \quad (7)$$

$$\mathcal{L}_{\text{reg}} = (a\mathcal{L}'_{\text{reg}} + c)^{10}. \quad (8)$$

We set coefficient a to 0.03. Shift constant c is set to 2139 to offset the optimal value of $a\mathcal{L}'_{\text{reg}}$, which we find to be -2139 in practice.

Our ablation analysis confirms that this variant of the regularization leads to overall numeracy improvements across both CoCoCount and D2D-Small (Tab. 9), indicating our regularization objective allows for effective numeracy correction, even while enforcing a heavier penalty on non-Gaussian initial noises.

Table 9: **Our variant of regularization is beneficial to numeracy optimization.** \mathcal{L}_{reg} , which penalizes larger deviations from Gaussian more and smaller deviations less, yields higher numeracy on CoCoCount and D2D-Small benchmarks. On SDXL-Turbo. Avg. over four seeds.

Method	CoCoCount	D2D-Small
<i>D2D</i> w/ $\mathcal{L}'_{\text{reg}}$	53.88	42.44
<i>D2D</i> w/ \mathcal{L}_{reg}	55.62	43.69

Table 10: Our variant of regularization also improves diversity, as measure by Vendi score (Friedman & Dieng, 2023), on CoCoCount and D2D-Small. On the SDXL-Turbo. Across four seeds.

Method	Vendi
<i>D2D</i> w/ $\mathcal{L}'_{\text{reg}}$	1.8198230
<i>D2D</i> w/ \mathcal{L}_{reg}	1.8231177

I.4 LMN ARCHITECTURE

For the choice of 3-layer MLP, we perform additional ablation studies and present the results in Tab. 11. Between MLP architectures, the 3-layer, 100-width and 200-width variants perform the best, but the former is more parameter-efficient. We also try using a 6-layer CNN (with adjusted kernels and channels to approximate the parameter count of the 3-layer, 100-width MLP), which yielded a slight 1% boost in numeracy. Our results demonstrate that even a small and simple MLP works quite well as an LMN architecture.

Table 11: **Even a small and simple 3-layer, 100-width MLP works effectively for boosting numeracy.** On CoCoCount. On SDXL-Turbo. Seed=0.

LMN Architecture	Numeracy
3-layers, width 100	55.5
2-layers, width 100	52.5
4-layers, width 100	52.0
3-layers, width 50	54.5
3-layer, width 200	55.5
CNN (approximate parameter-count match)	56.5

I.5 PRE-INFERENCE ALIGNMENT OF M_ϕ

We conduct experiments without the pre-inference alignment stage and report below (Tab. 12). We find that without pre-inference alignment, it generally takes an additional 20-30 steps during inference time to align the network to Gaussian outputs, which effectively reduce the number of steps available to perform numeracy optimization, because we cap the max number of steps.

Table 12: **Pre-inference alignment frees up more inference-time steps to be dedicated to numeracy optimization,** yielding higher numeracy. On CoCoCount, using SDXL-Turbo. Seed=0.

Method	Numeracy
w/ pre-inference alignment	55.5
w/o pre-inference alignment	47.0

J SELECTIVE *D2D* ACTIVATION

D2D focuses on numerical prompts, but not all prompts are numeric in nature. Furthermore prompts may lack concrete numerical targets, as in the prompt "A few cups." From the perspective of user intention, using words, like "few" or "many," that lack clear numerical grounding also indicates lack of strict numerical target in mind. So, it is reasonable to allow some flexibility in mapping those words to numbers. At test-time, we can leverage an LLM to determine whether a prompt is a numerical one and, if so, to extract the (object, requested count) pairs. Below is a sample context for this task.

In-context prompting for *D2D* selective activation

You are an expert prompt analyzer. Given a text-to-image prompt, you extract two things:

1. Boolean indicator of whether the prompt specifies a count of objects. Examples include "Six dogs," "Some cats." It is not a countable prompt if all the objects are uncountable nouns. For the prompt to be a counting prompt, at least one object must be countable.
2. For each ****countable**** object and requested count, extract this information (object in singular form, object in form seen in the prompt, requested count in number form). So, for prompts with multiple objects and corresponding requested counts, you would extract a list like this [(object in singular form, object in form seen in the prompt, requested count), (object in singular form, object in form seen in the prompt, requested count), ...].

#####

Examples:

Prompt: A photo of seven dogs.

Think: Prompt requests seven dogs, and dog is countable, so it is a counting prompt. Seven is 7 in number form. The object form seen in the prompt is dogs. The corresponding singular form is dog.

Response: True, [("dog", "dogs", 7)]

Prompt: A photo of seven dogs and two cats.

Think: Prompt requests seven dogs and two cats. In this case, both dog and cat are countable, so it is a counting prompt. Seven is 7 in number form, and two is 2. The object form seen in the prompt is dogs and cats. The corresponding singular forms are dog and cat.

Response: True, [("dog", "dogs", 7), ("cat", "cats", 2)]

Prompt: A photo of some cups.

Think: Prompt requests some cups; "some" indicates a small number, so it is countable. Let's just choose a reasonable number between 1 and 5, feel free to randomize within this range. Let's choose 3 this time. The object form seen in the prompt is cups. The corresponding singular form is cup.

Response: True, [("cup", "cups", 3)]

Transcript of selected LLM responses

Prompt: A realistic photo of a scene with nine pineapples.

ChatGPT: Think: The prompt specifies "nine pineapples." Pineapple is a countable noun, so this is a counting prompt. The number "nine" corresponds to 9. The object form seen in the prompt is "pineapples," and the singular form is "pineapple."

Response:

True, [{"pineapple", "pineapples", 9}]

Prompt: A photo of some water.

ChatGPT: Think: The prompt specifies "some water." Water is an uncountable noun, so it cannot be treated as a countable object. There is no countable object requested here.

Response:

False, []

Prompt: A few apples.

ChatGPT: Think: The prompt specifies "a few apples." Apple is a countable noun, so this is a counting prompt. "A few" typically corresponds to 3. The object form seen in the prompt is "apples," and the singular form is "apple."

Response:

True, [{"apple", "apples", 3}]

Prompt: A magnificent landscape with lots of sunshine, animals everywhere, and plenty of greenery.

ChatGPT: Think: The prompt mentions "animals," but it does not specify a number or quantity for any countable object. All quantities are vague ("lots," "plenty"), so there is no explicit count requested.

Response:

False, []

Prompt: A bright outdoor farmers' market captured in a natural documentary style: show 4 wooden tables arranged in two rows, each table displaying 11 neatly stacked pumpkins. Behind the tables stand 5 vendors, each with 3 small hanging lanterns attached to their stall. On the left, include 3 crates, each filled with 6 fresh apples, all clearly visible. In front of the crates, show 2 children crouched down examining 9 colorful marbles scattered on the ground. On the right side of the scene, place 2 bicycles, each with 2 woven baskets, and inside each basket include 4 lemons. Add 5 small fabric flags strung across the walkway above. Ensure all objects are visible, not overlapping, and easy to count. (*prompt generated separately by ChatGPT*).

ChatGPT: True, [("table", "tables", 4), ("pumpkin", "pumpkins", 11), ("vendor", "vendors", 5), ("lantern", "lanterns", 3), ("crate", "crates", 3), ("apple", "apples", 6), ("child", "children", 2), ("marble", "marbles", 9), ("bicycle", "bicycles", 2), ("basket", "baskets", 2), ("lemon", "lemons", 4), ("flag", "flags", 5)]

K $D2D$ SCALING FORMULATION

Table 13: **Our $D2D$ loss formulation is effective.** We compare our formulation against several alternatives. Ours yields the higher numeracy with fastest inference. On CoCoCount. On SDXL-Turbo. Seed=0.

Scaling formulation	Numeracy	Inference time (s)
$\frac{1}{1+e^{-\beta(z-\tau)}} \cdot (z - \tau)$ (ours)	55.50	17.10
$e^{z-\tau}$	52.00	56.28
$e^{\sigma(\beta(z-\tau))}$	31.50	32.24
$z - \tau$	50.5	53.23
$\ln(1 + e^{z-\tau})$	55.0	20.09
$(z - \tau) \cdot e^{z-\tau}$	49.0	72.72
$(z - \tau) \cdot \ln(1 + e^{z-\tau})$	54.0	22.31
$\sigma(\beta(z - \tau)) \cdot e^{z-\tau}$	50.5	63.96
$\sigma(\beta(z - \tau)) \cdot \ln(1 + e^{z-\tau})$	55.0	18.65

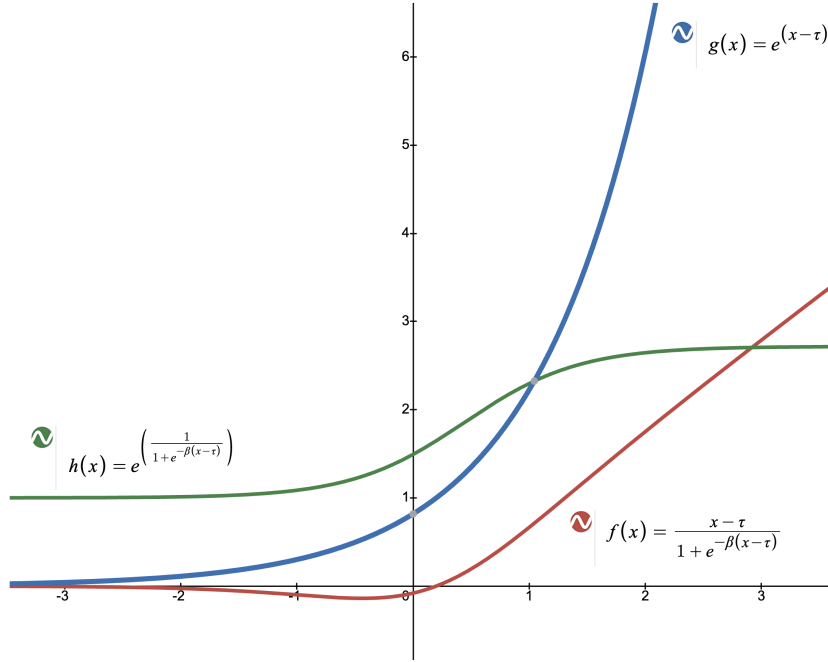


Figure 14: We want to design a fully-smooth loss that will effectively update each bbox’s logit in order to bring the total count closer to the requested count. To illustrate, if the total generated count $>$ requested count, then the overall objective is to bring the logits down. However, practically speaking, on the level of individual logits, this means focusing on decreasing the logits that are above the τ and allowing some flexibility for logits that are already below τ so that the optimization can focus on the $> \tau$ logits. In other words, we want a strong gradient signal above τ and little signal below. This plots the curves of the first three scaling formulations from the table above, illustrating how they behave above and below τ . Our formulation (in red) has a strong signal above and weak signal below τ . $e^{z-\tau}$ still maintains a significant gradient below τ . $e^{\frac{1}{1+e^{-\beta(z-\tau)}}}$ still shows plateauing above τ .

L SAMPLE NUMERACY OPTIMIZATION TRAJECTORY

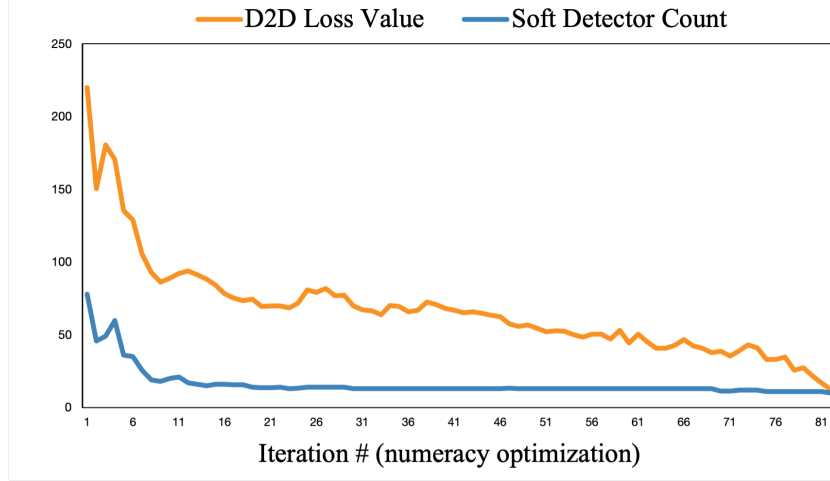


Figure 15: Sample trajectory of \mathcal{L}_{D2D} and the soft detector count f over numeracy optimization steps for the prompt “A photo of ten sheep on the grass” from CoCoCount. Though the two curves do not have the same raw value, their coarse trends do correlate.

M EXTENSION TO MULTI-STEP MODELS

We show that the *D2D* critic can be extended to multi-step diffusion models as well. We demonstrate this by applying *D2D* to the multi-step initial noise framework proposed in Direct Noise Optimization (DNO) (Tang et al., 2025) (see Fig. 16). However, there is a significant tradeoff in latency, as convergence took approximately 7 minutes for the prompt “A realistic photo of a scene with five cups.”

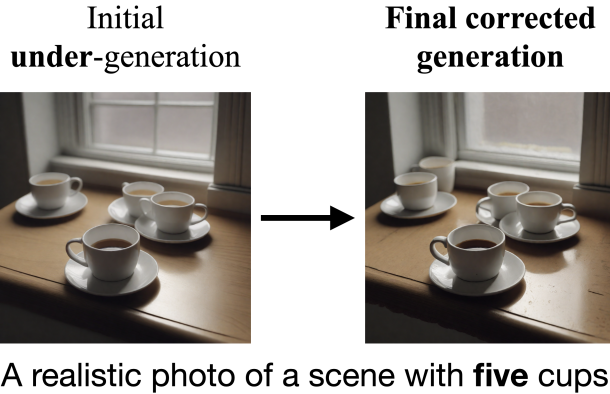


Figure 16: Example of numeracy correction using DNO (Tang et al., 2025) with our *D2D* critic. This took 12 optimization steps (approximately 7 minutes).

N DIVERSITY ANALYSIS OF GENERATED IMAGES

Table 14: *D2D* improves diversity, compared to corresponding base models. Diversity measured using Vendi score (Friedman & Dieng, 2023). On CoCoCount and D2D-Small, using base models SDXL-Turbo, SD-Turbo, and Pixart-DMD. Across four seeds.

Method	Vendi
SDXL-Turbo	1.7561059
SDXL-Turbo + <i>D2D</i>	1.8231177
SD-Turbo	1.8688096
SD-Turbo + <i>D2D</i>	1.9801648
Pixart-DMD	1.7600942
Pixart-DMD + <i>D2D</i>	1.8248777

O NUMERACY IN LARGER MODELS

Table 15: Counting remains a problem. Even newer/larger models like Flux-Schnell still struggle with accurate counting (46.0%), confirming the continued relevance of this problem. By integrating *D2D* with smaller models like SDXL-Turbo, we can already achieve better numeracy accuracy than larger models such as SD3.5 Large Turbo and FLUX-Schnell.

Method	Numeracy
FLUX.1 [schnell] (1-step)	46.0
FLUX.1 [schnell] (4 steps)	47.5
SD3.5 Large Turbo (4-steps)	48.0
SDXL-Turbo	30.0
SDXL-Turbo + <i>D2D</i>	55.5