# Random Conditioning with Distillation for Data-Efficient Diffusion Model Compression

Supplementary Material

## A. Data-Efficient Distillation

In our experiments in Tab. 3, we highlight our method's effectiveness under a fully data-free scenario. To address this challenging setup, we develop a systematic workflow for generating a large-scale caption dataset, relying entirely on LLM-generated captions for both image synthesis and random conditioning.

The general workflow consists of two main stages. The first stage involves identifying a comprehensive set of conceptually meaningful *visual* nouns from a structured linguistic database. To exclude nouns that are not visual for text-to-image tasks (*e.g.*, abstract concepts such as "justice" or "freedom"), LLM-based filtering step is applied, ensuring only clearly visualizable concepts are selected. In the second stage, diverse and extensive textual prompts are systematically generated for each selected noun through LLM inference, resulting in a rich synthetic caption dataset suitable for image synthesis under fully data-free conditions.

Specifically, in our experimental setup, we first extract nouns from WordNet [41], restricting the selection to those with a hierarchical depth of 14 to avoid overly specific nouns (*e.g.*, detailed species or subspecies). Subsequently, we prompt GPT-3.5-turbo [45] to filter only visual nouns, ultimately selecting approximately 17K nouns. For each selected noun, GPT-3.5-turbo is prompted to generate 128 unique textual prompts, resulting in approximately 2.2M synthetic captions. From this pool, we randomly sample 212K prompts, ensuring a balanced distribution across nouns, and utilize a teacher model to generate corresponding image data. Below are the specific GPT prompts used for noun filtering and prompt generation.

### Noun Filtering Prompt

- System Prompt

> *You are a helpful assistant.*

- User Prompt

> *Can a text-to-image model clearly visualize {'noun'}? It must be a noun. Answer 'yes' or 'no' and give a brief explanation.*
>
> **Input**: *{'noun'}*

### Prompt Generation

- System Prompt

> *You are a helpful assistant.*

- User Prompt

> *Generate 128 visually rich prompts focused on the noun {'noun'}, ensuring each prompt is strictly distinct and cannot overlap in style, setting, or composition. Please return only the prompts on separate lines with numbering.*
>
> **Input**: *{'noun'}*

| Models | #Params | MACs | FID↓ | IS↑ | CLIP↑ |
|---|---|---|---|---|---|
| Teacher | 860M(+00.0%) | 339G(+00.0%) | 13.05 | 36.76 | 0.2958 |
| B-Base | 580M(-32.6%) | 224G(-33.9%) | 14.47 | 36.50 | 0.2932 |
| B-Small | 483M(-43.9%) | 218G(-35.7%) | 16.22 | 35.99 | 0.2804 |
| B-Tiny | 324M(-62.4%) | 205G(-39.5%) | 16.71 | 35.46 | 0.2782 |
| C-Base | 554M(-35.8%) | 217G(-36.0%) | 14.45 | 34.92 | 0.2904 |
| C-Small | 426M(-50.5%) | 166G(-51.0%) | 14.43 | 34.58 | 0.2888 |
| C-Tiny | 315M(-63.5%) | 122G(-64.0%) | 13.90 | 33.18 | 0.2860 |
| C-Micro | 220M(-74.4%) | 85G(-74.9%) | 13.42 | 32.64 | 0.2813 |

Table A. **Comparison of Model Size and MACs.** We measure UNet parameter count and the MACs for a single step in the UNet. for the teacher model and our models. THOP [91] is used to measure the MACs, following the approach of BK-SDM [24].

## B. Model Compression

We evaluate the efficiency of our block- and channel-compressed models in terms of UNet parameter counts, multiply-accumulate operations (MACs), and metric scores, as summarized in Tab. A. The results highlight the efficiency gains of our compressed models with random conditioning over the teacher model while maintaining performances. Notably, channel-compressed models show lower MACs than block-compressed models with similar parameter counts; C-Micro requiring only 25% of the MACs compared to the teacher model. Despite being trained from scratch, these channel-compressed models demonstrate competitive performance, even outperforming their block-compressed counterparts in several metrics. This is due to random conditioning, which expands the exploration of the condition space during distillation, offsetting the lack of teacher weight initialization.

| # | Rand Cond | Additional Text | FID↓ | IS↑ | CLIP↑ |
|---|---|---|---|---|---|
| 1 | ✗ | - | 18.31 | 26.83 | 0.2579 |
| 2 | ✓ | 0 | 16.70 | 26.53 | 0.2613 |
| 3 | ✓ | 1M | 16.13 | 28.89 | 0.2677 |
| 4 | ✓ | 10M | 15.67 | **28.90** | 0.2674 |
| 5 | ✓ | 20M | **15.22** | 28.89 | **0.2680** |

Table B. **Impact of Random Conditioning by Additional Text Size.** We evaluated models trained with additional text sizes of 0, 1M, 10M, and 20M, using the C-Micro architecture for 125K training iterations.

## C. Impact of Additional Text Data Size

Tab. B presents the performance results based on the amount of additional text dataset used for random conditioning. Row 1 shows the results of a naïve distillation approach without random conditioning, exhibiting lower performance compared to Rows 3, 4, and 5, which incorporate additional text datasets through random conditioning. Notably, Row 2, which applies random conditioning using only the training image-text pairs (212K) without any additional text data (0 additional text), achieves comparable or higher performance compared to Row 1. This indicates that random conditioning effectively utilizes unpaired text-image data during distillation without any performance degradation and can even enhance training. When comparing Rows 3, 4, and 5, which use additional text datasets of 1M, 10M, and 20M respectively, FID shows a slight improvement as the amount of text data increases, while IS and CLIP scores remain similar. This demonstrates that increasing the amount of text data can enhance training, but even a limited amount, such as 1M, is sufficient to significantly improve model performance. Furthermore, 1M text data requires substantially less memory compared to image datasets and is easier to obtain, highlighting the practical effectiveness of random conditioning in real-world scenarios.

## D. Random Conditioning Probability

Tab. C shows the scores for different random conditioning probabilities in two different setups: student models with random and teacher initialization. The different $p(t)$ functions are plotted in Fig. A. Row 2 corresponds to the exponential function used in the main experiments. Row 3 modifies this by symmetrically mirroring the function around $t = T/2$. Row 4 represents a linear function that increases from 0 at $t = 0$ to 1 at $t = T$. Row 5 uses a sigmoid function, shifted horizontally to ensure $p(t)$ approaches 1 for large time steps. Rows 6 and 7 employ constant functions, with probabilities fixed at 0.5 and 1, respectively, for all $t$. When the student is randomly initialized, all $p(t)$ functions except for $p(t) = 1$ outperform the baseline without random conditioning (Row 1). In particular, the sigmoid

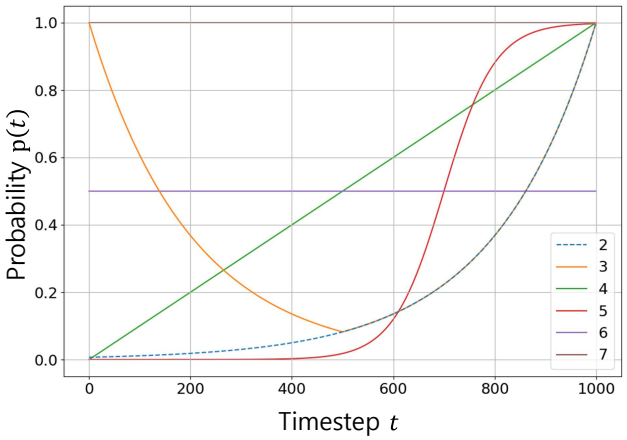


Figure A. **Plots of Different $p(t)$ used for Random Conditioning.** Each plot corresponds to a row in Tab. C.

function (Row 5) yields the strongest performance. When the model is initialized with the teacher weights, the improvements are less pronounced compared to Random Init, but most $p(t)$ functions still lead to better performance than the baseline, with the exponential function (Row 2) showing the best results.

Based on these observations, the final models reported in Tab. 4 and Tab. A use the sigmoid function (Row 5) for C-compressed models (Random Initialization) and the exponential function (Row 2) for B-compressed models (Teacher Initialization), as these $p(t)$ choices led to strong CLIP and IS scores in each setting. For all other experiments, we use the exponential function (Row 2) as the default $p(t)$. While this choice may not be optimal, it demonstrates a significant performance improvement over the baseline, validating its effectiveness.

## E. Discussions on Other Image-Free Methods.

There exist several image-free distillation approaches for building one-step diffusion models [10, 42, 89, 90], which do not require learning from intermediate noisy samples $\mathbf{x}_t$ since one-step models directly generate outputs without denoising steps. These methods receive signals from a pretrained teacher model during training, making them independent of explicit dataset requirements. However, in our case, the objective is to compress the teacher model into a smaller model while preserving its characteristics, which necessitates training on intermediate noisy samples $\mathbf{x}_t$. Similarly, DKDM [79] applies Dynamic Iterative Distillation for efficient compression, and although it also generates $\mathbf{x}_t$ through sampling from the teacher model, it specifically targets unconditional models. As a result, these methods are not directly applicable to our target task—distilling a conditional diffusion model with a large conditioning space using only a limited number of generated images.

| # | $p(t)$ | Random Initialization FID↓ | IS↑ | CLIP↑ | Teacher Initialization FID↓ | IS↑ | CLIP↑ |
|---|---|---|---|---|---|---|---|
| 1 | - | 19.69 | 28.75 | 0.2618 | 15.38 | 34.59 | 0.2905 |
| 2 | $p(t) = e^{-\lambda \cdot \left(1-\frac{t}{T}\right)}$ | 16.19 | 31.81 | 0.2727 | 14.47 | 36.50 | **0.2932** |
| 3 | $p(t) = \begin{cases} e^{-\lambda \cdot \left(1-\frac{t}{T}\right)}, & \text{if } t > \frac{T}{2} \\ e^{-\lambda \cdot \frac{t}{T}}, & \text{otherwise} \end{cases}$ | 15.39 | 31.22 | 0.2730 | 14.53 | **36.52** | 0.2917 |
| 4 | $p(t) = \frac{t}{T}$ | 15.30 | 31.82 | 0.2728 | 14.06 | 36.35 | 0.2898 |
| 5 | $p(t) = \frac{1}{1+e^{-20\left(\frac{t}{T}-0.7\right)}}$ | 15.87 | **33.34** | **0.2751** | 15.46 | 35.85 | 0.2916 |
| 6 | $p(t) = 0.5$ | 14.68 | 30.70 | 0.2692 | 14.56 | 35.86 | 0.2901 |
| 7 | $p(t) = 1.0$ | **14.43** | 28.55 | 0.2586 | **12.63** | 35.96 | 0.2909 |

Table C. **Effect of $p(t)$ in Random Conditioning.** We evaluate the models trained with varying random conditioning probabilities, $p(t)$. The first row represents the baseline without random conditioning. "Random Initialization" and "Teacher Initialization" refer to student models trained from scratch and from teacher-initialized weights, respectively. All models with random initialization are trained for 125K iterations, while those with teacher-initialized are trained for 75K iterations using the B-Base architecture.

| # | Rand Cond | Real image | FID↓ | IS↑ | CLIP↑ | Attribute Binding Color↑ | Shape↑ | Texture↑ | Object Relationship Spatial↑ | Non-Spatial↑ | Complex↑ | Average↑ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| (Teacher) | | | 13.05 | 36.76 | 0.2958 | 0.3599 | 0.3542 | 0.4004 | 0.1055 | 0.3095 | 0.3095 | 0.3065 |
| 1 | ✗ | ✗ | 18.15 | 33.81 | 0.2864 | 0.3474 | 0.3448 | 0.3786 | 0.0966 | **0.3092** | 0.3118 | 0.2985 |
| 2 | ✗ | ✓ | 15.76 | 33.79 | 0.2878 | 0.3585 | 0.3397 | 0.3838 | 0.0981 | 0.3080 | 0.3124 | 0.3001 |
| 3 | ✓ | ✗ | 15.76 | 36.03 | 0.2896 | 0.3593 | 0.3551 | 0.4053 | 0.0889 | 0.3086 | 0.3148 | 0.3053 |
| 4 | ✓ | ✓ | **15.00** | **36.14** | **0.2933** | **0.3789** | **0.3576** | **0.4207** | **0.1112** | 0.3075 | **0.3196** | **0.3159** |

Table D. **Results on T2I-CompBench [21] with Additional Metrics.** The row with a gray background shows the performance of the teacher model [55] for reference. "Rand Cond" indicates whether random conditioning is applied, and "Real image" specifies the use of real images during training. All models are based on the B-Base architecture.

| # | Rand Conditioning | Feature loss | FID↓ | IS↑ | CLIP↑ |
|---|---|---|---|---|---|
| 1 | ✗ | ✗ | 15.96 | 33.30 | 0.2786 |
| 2 | ✗ | ✓ | 18.15 | 33.81 | 0.2864 |
| 3 | ✓ | ✗ | **13.71** | 34.10 | 0.2847 |
| 4 | ✓ | ✓ | 15.76 | **36.03** | **0.2896** |

Table E. **Impact of Feature Loss.** We evaluate the effect of random conditioning with and without feature loss. All models are based on the B-Base architecture.

## F. Inconsistent Trends on FID

Although random conditioning generally improves performance across FID, IS, and CLIP scores, the trend is not always consistent for FID. As noted in [21], FID is known to exhibit substantial fluctuations, making it less reliable for fine-grained comparison. To better validate our findings, we additionally evaluate models on T2I-CompBench [21]. The results, presented in Tab. D, show improvements across most reported metrics, aligning well with IS and CLIP scores. These findings support our hypothesis that random conditioning helps the student model better explore the condition space, leading to performance improvements.

## G. Significance of Feature Losses

In Table E, we report additional experiments comparing the performance of our method with and without feature losses. By comparing Row 1 and Row 3, we observe that even in the absence of feature losses, random conditioning remains effective. Moreover, the comparisons between Row 1 and Row 2 as well as between row3 and Row 4 indicate that incorporating feature losses leads to improvements in both the IS and CLIP scores. These results are consistent with the findings in [24]. Note that our feature loss implementation follows those in [24], and we also observed a tendency for lower FID scores when feature losses are omitted.

## H. SDXL Compression with Koala

We apply random conditioning while compressing SDXL using the KOALA-700M [27] architecture as the student model, distilling from the SDXL-Base model [51]. Due to resource constraints, the number of training iterations is

| # | Random Conditioning | #Params | FID↓ | IS↑ | CLIP↑ |
|---|---|---|---|---|---|
| | (Teacher) | 2.56B | 13.04 | 35.83 | 0.3257 |
| 1 | ✗ | 0.78B | 23.28 | 27.93 | 0.2855 |
| 2 | ✓ | 0.78B | **21.45** | **28.53** | **0.2905** |

Table F. **Effect of Random Conditioning in SDXL.** The row with a gray background corresponds to the Teacher model (SDXL-Base [51]), while Rows 1 and 2 represent student models trained using the KOALA-700M architecture.

| Method | Random Conditioning | FID↓ | IS↑ | CLIP↑ |
|---|---|---|---|---|
| SLIM | ✗ | 27.79 | 16.76 | 0.2063 |
| SLIM | ✓ | 23.97 | 18.83 | 0.2174 |
| BK-SDM | ✗ | 15.76 | 33.79 | 0.2878 |
| BK-SDM | ✓ | **15.00** | **36.14** | **0.2933** |

Table G. **Effect of Random Conditioning in SLIM.** We evaluate the impact of random conditioning within SLIM's loss function and architecture. The "Method" column indicates the model configuration. All models are based on the B-Base architecture.

lower than what is reported for KOALA, resulting in relatively lower scores. The results are presented in Tab. F. Following the main experiments, we use 212K training images generated by the teacher model (SDXL-Base) using captions from LAION-Aesthetics V2 (L-Aes) 6.5+. We also incorporate an additional 20M text dataset through random conditioning.

## I. SLIM-Based Distillation

We also apply random conditioning in a distillation setup following SLIM's [82] loss function and architecture. Specifically, we use the authors' code to compress Stable Diffusion 1.4 on the B-Base architecture, incorporating the Dynamic Wavelet Gating module and using frequency loss. Across all tested configurations, models trained with Random Conditioning consistently achieve higher performance. However, in the SLIM setting, factors such as the smaller dataset size (212K vs. 400M in the original paper), fewer feature losses, and SLIM's unique model structure make it less competitive in our configuration. The results are shown in Tab. G. We adopt the same dataset setup as the main experiments.

## J. Details of Animal-Related Data Filtering

**Training set** To exclude animal images from training, we apply a filtering process to the original 212K LAION [65] dataset. First, each caption is checked for animal-related terms using a curated list, expanded from the 10 MS-COCO animal category names via GPT-4o [46] prompting and manual review. Next, the remaining captions are assessed by GPT-3.5-turbo [45], prompted to determine whether they are in any way related to animals. Finally, to catch cases where original captions miss animal presence, we generate new captions using BLIP [29] and apply the same filtering process.

**Evaluation set** For the analysis presented in Tab. 2, we use the MS-COCO validation split, which consists of 41K (40,504) image-text pairs. To compare performance across different dataset compositions, we construct two subsets: one containing 8K (8,265) samples categorized under the "animal" supercategory and the other comprising the remaining 33K (32,239) samples.

- System Prompt

> *You are an assistant that identifies if each sentence in the provided JSON contains references to animals. Do not consider plants, objects, places, or any word that could be confused for an animal. If you find a reference to an animal, provide the exact word that led you to identify it.*

- User Prompt

> *Read each of the following sentences and determine if it contains a reference to an animal. The sentences are given in JSON format. Provide an answer with 'yes' or 'no' for each sentence. If the answer is 'yes', provide the specific animal name that led you to this conclusion. Provide the answers in JSON format as a list of dictionaries, where each dictionary contains 'contains_animals' with either 'yes' or 'no', and 'reason', which is the animal name if 'yes', or an empty string if 'no'.*
>
> **Input**: *{JSON}*

## K. Further Implementation Details

During inference, including caching $\mathbf{x}_0$ images for training, evaluation, and generating qualitative results, we consistently employ the DDIM sampler [70] with a total of $T$=25 sampling steps, and the classifier-free guidance [16] is set to its default value of 7.5. In our experiments, we evaluate the models every 25K iterations. Unless specified otherwise, we report the best scores achieved within 125K iterations for experiments initialized with the teacher model and within 400K iterations for those with random initialization; notably, Tab. 2 presents scores at 500K iterations. For evaluation, we generate images at a resolution of 512×512 and resize them to 256×256, following [24]. In Tab. 4, we evaluate [51], [3], and [72] using the default settings from the Diffusers library [75]. Images for these models are generated at 1024×1024 resolution and then resized to 256×256 to adhere to our evaluation protocol.

## L. Prompts Used for Qualitative Samples

The first two prompts correspond to (a), while the last three prompts correspond to (b) in Fig. 1. Notably, the prompts in (b) are related to animals. We use the following prompts for Fig. 1 from left to right:

- My favorite landscape I've visited Mount Assinboine Provincial Park Canada.
- Storm Over The Black Sea Poster by Ivan Aivazovsky.
- Dogs on Girder Poster.
- David Shepherd, Stag, oil on canvas.
- Fotorolgordijn Schildpad Beautiful Green sea turtle swimming in tropical island reef in hawaii, split over/underwater picture.

We use the following prompts for Fig. 6 from top to bottom:

- Anthropomorphic jackal wearing steampunk armor, beautiful natural rim light, intricate, fantasy, anubis, elegant, hyper realistic, photo realistic, ultra detailed, concept art, octane render, beautirul natural soft rim light, silver details, elegant, ultra detaied, dustin panzino, giger, mucha.
- Medium shot side profile portrait photo of a warrior chief, sharp facial features, with tribal panther makeup in blue on red, looking away, werious but clear eyes, 50mm portrait, photography, hard rim lighting photography.

## M. Additional Qualitative Results

To further illustrate our experiments, we present additional qualitative results generated using the text dataset from DiffusionDB [78]. Fig. B provides additional results for the animal-related images shown in Fig. 1, while Fig. C is corresponding to Fig. 6.

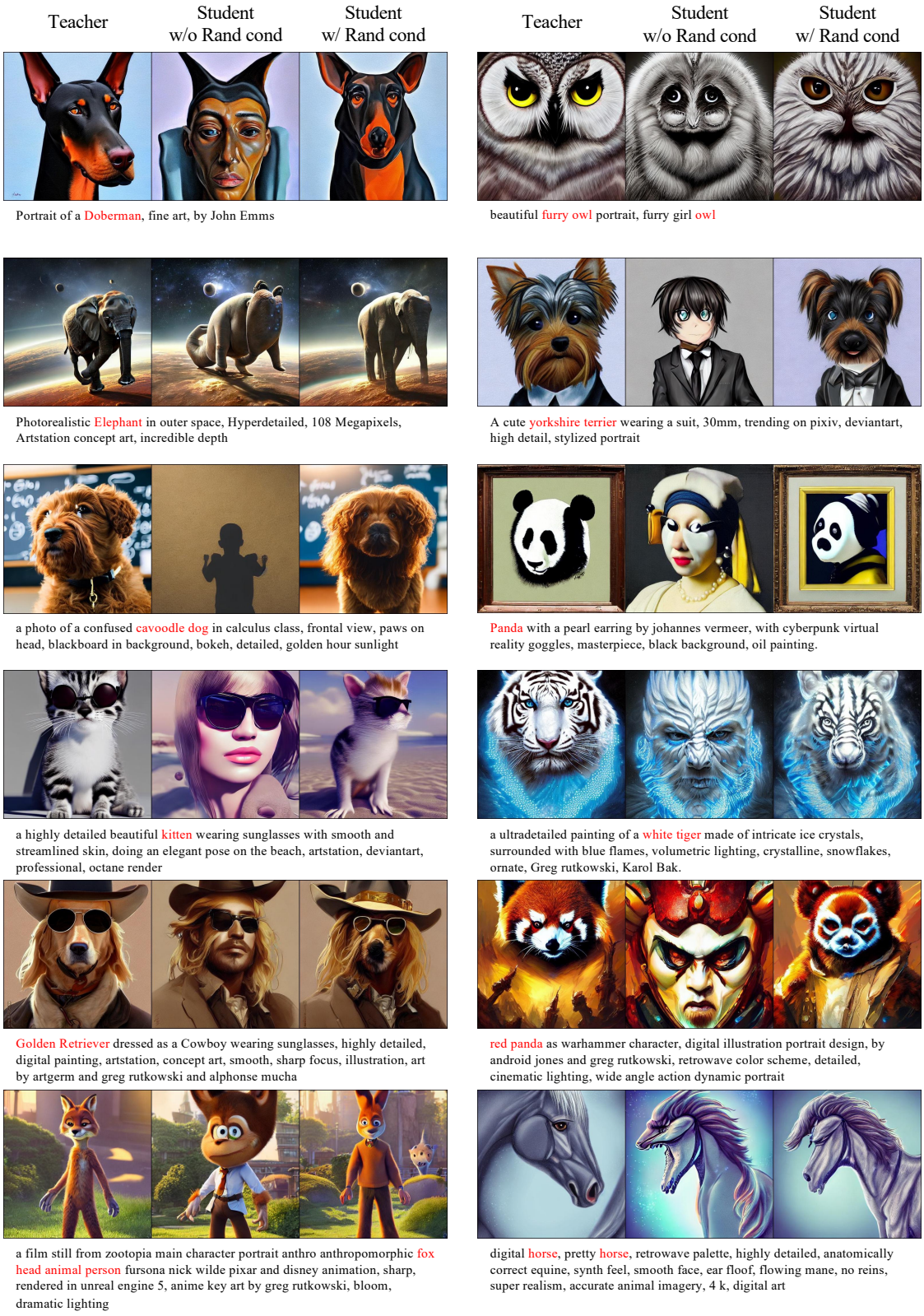


Figure B. **Additional Qualitative Results on DiffusionDB [78] datasets of Baseline and Our Method Trained Without Animal Image.** All samples are generated conditioned on captions related to animals, with animal-related terms highlighted in red for each image's caption

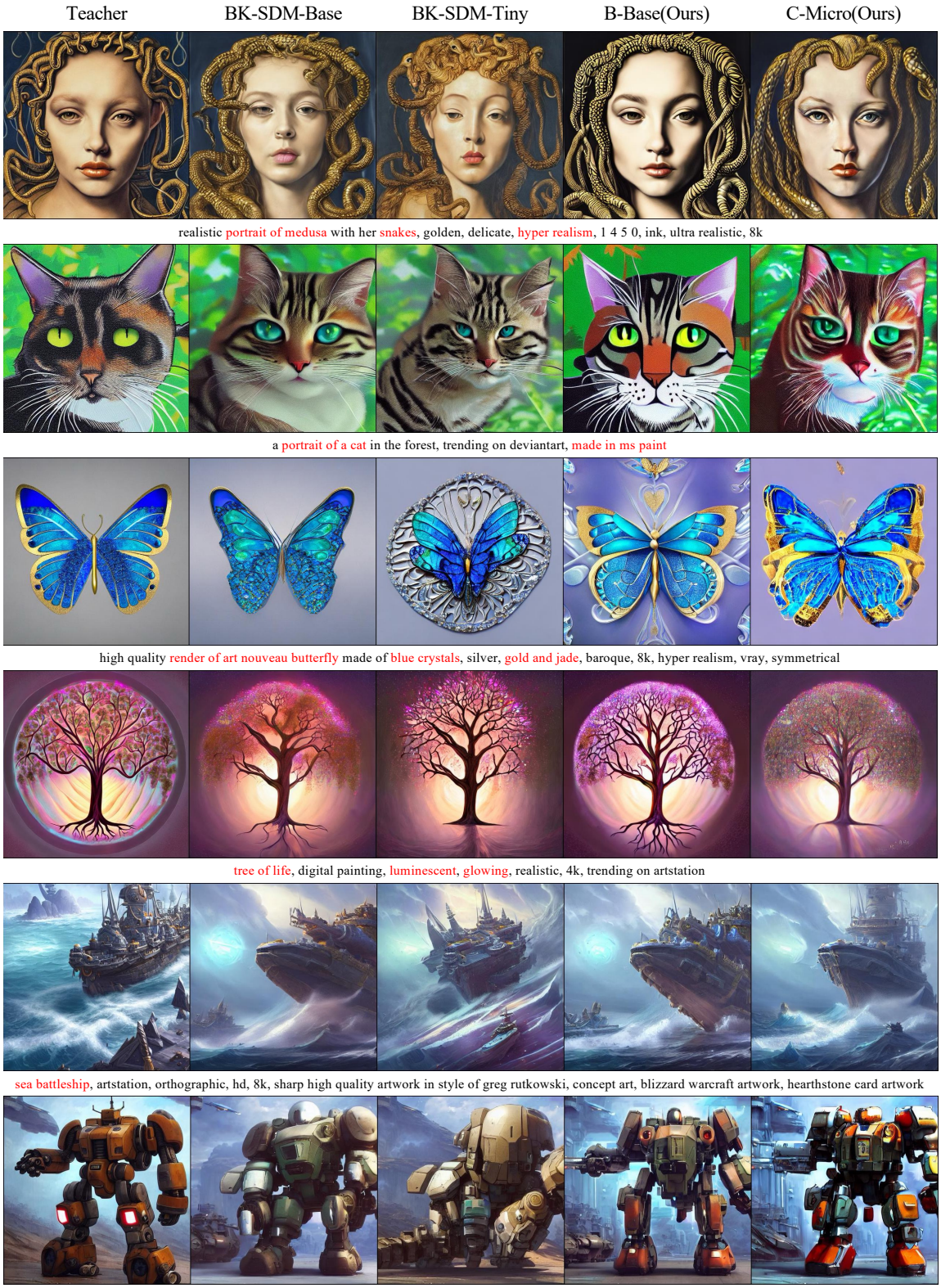


Figure C. **Additional Qualitative Results on DiffusionDB [78] datasets of Our Models and Baseline Models.**