Reusing Computation in Text-to-Image Diffusion for Efficient Generation of Image Sets

Dale Decatur[1] Thibault Groueix[2] Wang Yifan[2] Rana Hanocka[1]
Vladimir Kim[2] Matheus Gadelha[2]

[1]University of Chicago [2]Adobe Research

Figure 1. When generating a collection of images from text prompts, our approach shares denoising steps and can produce images of comparable or better quality while using a fraction of the compute. Here, we show images generated by the standard approach vs our approach, for a fixed compute budget.

Abstract

Text-to-image diffusion models enable high-quality image generation but are computationally expensive. While prior work optimizes per-inference efficiency, we explore an orthogonal approach: reducing redundancy across correlated prompts. Our method leverages the coarse-to-fine nature of diffusion models, where early denoising steps capture shared structures among similar prompts. We propose a training-free approach that clusters prompts based on semantic similarity and shares computation in early diffusion steps. Experiments show that for models trained conditioned on image embeddings, our approach significantly reduces compute cost while improving image quality. By leveraging UnClip's text-to-image prior, we enhance diffusion step allocation for greater efficiency. Our method seamlessly integrates with existing pipelines, scales with prompt sets, and reduces the environmental and financial burden of large-scale text-to-image generation.

1. Introduction

Diffusion models have revolutionized creative workflows, profoundly reshaping the landscape of image generation since their introduction [7, 16, 27, 28, 30, 35]. Their impressive capability to generate realistic, high-quality images from textual descriptions has made them integral to both consumer applications and professional creative processes [1, 24, 25, 29]. However, the significant computational cost associated with diffusion models poses a substantial barrier to their widespread practical deployment—generating a single image can consume energy equivalent to charging half a smartphone battery [21].

In practical scenarios, users rarely generate just a single image. Instead, these models serve primarily as tools to visually explore ideas, leading users to produce and evaluate numerous image variations to refine and select the most satisfactory outputs. Community-driven methods, such as prompt templates and automatic prompt variation techniques [2, 3, 6], have emerged precisely to facilitate this extensive exploration of prompt variations. Consequently,

the typical workflow for text-to-image generation inherently involves creating extensive image collections to effectively explore the ideation space encoded in textual inputs. Yet, despite the ubiquity of this approach, little research has explicitly targeted optimization across multiple prompts.

In this paper, we address this critical gap by proposing a novel method specifically designed to reduce computational overhead when performing large-scale inference on diffusion-based text-to-image models. While existing literature has primarily focused on per-inference cost optimization through methods such as distillation [22, 36], we explore an orthogonal direction. Our core insight leverages the widely recognized property of diffusion models (see Fig. 2): they generate images in a coarse-to-fine manner [25, 28, 30], where early denoising steps primarily define low-frequency, structural content (overall composition and layout), and subsequent steps progressively refine high-frequency, fine-grained details. Given the inherent statistical correlation among natural images at low frequencies, significant redundancy arises during initial diffusion steps for prompts describing visually distinct but structurally similar images – for example, "a Labrador wearing a bow-tie" and "a Portuguese water dog wearing sunglasses".

Motivated by this observation, we explore the fundamental question: can we reuse early-stage computations across correlated prompts to increase efficiency? Through extensive experiments, we found that while all diffusion models generate images from coarse to fine, certain families exhibit a more evenly distributed progression, making them particularly suitable for early-stage computation sharing. For instance, as shown in Fig. 2, Stable Diffusion (SD) [28] and Stable UnCLIP [26] generate fine details relatively early, limiting opportunities for shared computation. In contrast, models such as Kandinsky [27, 35] and Karlo [14] exhibit a more gradual emergence of high-frequency details, allowing more efficient reuse of early-stage computations. While we hypothesize that training with a text-to-image prior may encourage this behavior, our method remains effective as long as the model exhibits a sufficiently gradual detail emergence, regardless of embedding strategy.

To put these findings into practice, we introduce a straightforward yet effective approach that utilizes off-the-shelf text encoders and agglomerative clustering to automatically construct hierarchical representations of prompts. Specifically, given a set of prompts, we construct a hierarchical embedding tree, where similar concepts share their averaged embedding, used to guide early steps of the denoising process. By exploiting this tree structure during diffusion, our method shares computations among prompts with structural similarities in early diffusion steps and progressively specializes the process as denoising advances.

Through extensive experimentation across diverse image collections and prompt variations – including systematically generated prompts using established templates – we demonstrate that our hierarchical diffusion procedure significantly reduces computational overhead, achieving comparable or even superior visual quality with as few as 26% of the total diffusion steps required by conventional methods.

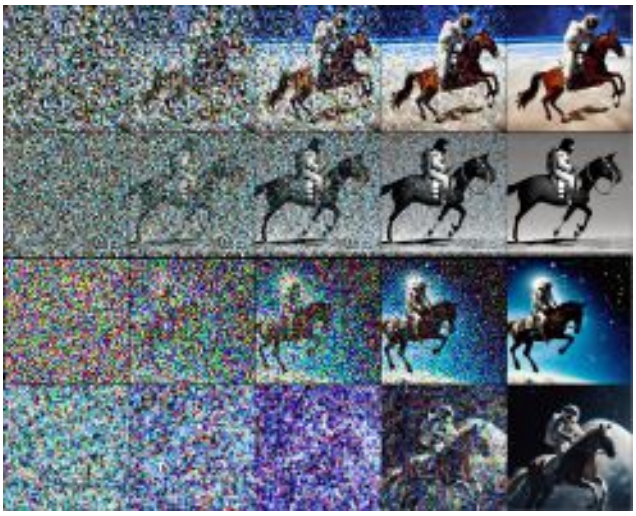

Figure 2. **Coarse-to-Fine Generation**. We show intermediate latents from the diffusion process for Stable Diffusion 1.5 [28] (top row), Stable UnCLIP [26] (second row), Karlo [14] (third row), and Kandinsky [27] (bottom row). The models trained without a text-to-image prior (Stable Diffusion and Stable UnCLIP) learn structural details and high frequency features earlier on in the diffusion process. In contrast, Karlo and Kandinsky which were trained with the text-to-image prior, learn structural details later in denoising and are able to quickly add high frequency details in few steps, making them ideal for our compute sharing approach.

Our approach is entirely automatic, training-free, easily integrated into existing denoising pipelines, and scales favorably with the number of prompts. Crucially, our findings not only demonstrate substantial computational savings but also highlight unique generation dynamics of different diffusion models, contributing valuable insights into diffusion behavior. We hope our research will inspire further exploration into efficient and sustainable multi-prompt optimization strategies, ultimately reducing the substantial carbon footprint associated with diffusion models.

2. Related Work

Fast Image Generation. Image generation has evolved through two major breakthroughs: Generative Adversarial Networks (GANs) [8] and diffusion models [9, 32]. GANs enable fast inference but suffer from training instability and poor generalization [4, 11]. Diffusion models, while more stable and expressive, require significantly more compute due to the iterative denoising at inference time. Methods such as latent-space diffusion [28] mitigate some inefficien-

cies but do not fundamentally reduce the sampling burden.

To accelerate diffusion sampling, prior works propose efficient numerical solvers [12, 18–20, 33, 38], reducing steps from 1000 steps [9, 32] to 10 [12, 33, 38], though in practice most models are sampled with about 40 steps to trade off between speed and quality. Distillation-based methods [13, 16, 22, 31, 36, 37] train models to generate images in fewer steps. While effective, these approaches require additional non-trivial training. Complementary to these methods, our work addresses the common scenario of multiple prompts by providing a training-free acceleration approach that leverages hierarchical prompting to enable shared computations across related prompts.

Prompt Expansion. Prompt expansion techniques help users explore diverse image variations by generating structured prompt sets. Large language models assist in expanding prompts while preserving semantic relevance [6]. Community-driven prompt templates provide predefined tag structures [5, 23], allowing systematic prompt variation.

Systems like DreamSheets [2] use spreadsheets to dynamically populate prompts, while Promptify [3] enhances user interactions with automated prompt suggestions. These tools enable efficient idea exploration. Our method aligns with this paradigm but focuses on efficient image generation when prompts are predefined, optimizing computation for structured workflows.

Coarse-to-Fine Generation in Diffusion Models. Diffusion models generate images progressively, with lower-frequency details emerging in early denoising steps and high-frequency details refined later [9]. While this property exists across architectures, models trained with UnCLIP-style image embeddings [26] exhibit it more distinctly.

UnCLIP models introduce a text-to-image embedding prior, mapping CLIP text embeddings to image embeddings. When trained with UnCLIP image embeddings, the model exhibits improved diversity [26] and, as shown by Karlo [14], generates fine-grained details with fewer steps, which consequently affects the allocation of diffusion steps, allowing fine details to emerge later in the process [27, 35]. Our method leverages this property effectively. It shares computation during early steps that generate low-frequency features, while dedicating fewer unique steps to produce the high-frequency details specific to each prompt.

3. Preliminaries: Text-to-Image Diffusion

Training. Given an input image $\mathbf{x}$ and text embedding y, a noisy image $\mathbf{x}_t$ is first obtained by adding a random amount of noise to $\mathbf{x}$:

$$\mathbf{x}_t = \sqrt{\alpha(t)}\,\mathbf{x} + \sqrt{1-\alpha(t)}\,\epsilon, \tag{1}$$

where $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ is Gaussian noise, and $t \in [0, T]$ parameterizes a schedule α corresponding to the amount of noise in $\mathbf{x}_t$; *i.e.* $\alpha(0) = 1$ (no noise) and $\alpha(T) = 0$ (pure noise). ϵ_θ is the denoiser – a model trained to invert this noising process by minimizing the following loss through gradient-descent:

$$\mathcal{L}_{\text{diff}} := w(t)\|\epsilon_\theta(\mathbf{x}_t; t, y) - \epsilon\|_2^2,$$

where $w(t)$ is a weighting scheme parametrized by t. In practice, all our experiments use latent diffusion models which means that the previous loss is computed in latent space and a VAE is used to decode it into pixels [28].

Inference. Once ϵ_θ is trained, a new image is generated using the following procedure. Given a text embedding y, we start from pure noise $\mathbf{x}_T$ and iteratively denoise it:

$$\begin{aligned} \mathbf{x}_T &\sim \mathcal{N}(\mathbf{0}, \mathbf{I}) \\ \mathbf{x}_{t-1} &\sim \mathcal{N}(a_t\mathbf{x}_t - b_t\epsilon_\theta(\mathbf{x}_t; t, y), \sigma_t^2\mathbf{I}) \end{aligned} \tag{2}$$

where a_t, b_t, the variance σ_t^2, and the timesteps are chosen according to a denoising schedule. The number of steps taken (K) is a parameter of the the denoising scheduler. K has a strong correlation with the quality of the generated image. After Eq. (2) is applied K times, $\mathbf{x}_t$ should yield a fully denoised result. The standard diffusion process generates a set of N images by running Eq. (2) independently for each text embedding, requiring KN evaluations of $\epsilon_\theta(\mathbf{x}_t; t, y)$.

4. Method

Our key insight is that two similar embeddings yield similar denoising updates during the early diffusion steps, introducing substantial redundancy. Our idea is to initially share computation using this shared embedding $\tilde{y}$ for the first few denoising steps before progressively diverging into specialized prompt-specific embeddings. For two prompts, assuming we share the initial $K/2$ steps, this simple strategy reduce the diffusion steps from $2K$ to $\frac{3K}{2}$.

For a large set of prompts, using a hierarchical tree structure maximizes computational reuse. This structure ensures that early diffusion computations – corresponding to the widely shared visual features – are performed only once at the root node, producing substantial computational savings. Subsequent computations then progressively branch out, specializing embeddings as the diffusion process advances toward fine-grained details. Notably, when structured as a balanced binary tree, as illustrated in Fig. 3, the computational complexity reduces from $\mathcal{O}(N)$ to $\mathcal{O}(\log N)$. As the number of prompts increases, opportunities for computational sharing grow significantly, substantially reducing the total number of denoising steps required.

Next, we introduce a fully automated method to generalize this strategy for arbitrary prompt sets.

Constructing Embedding Tree. Given a set of N text prompts encoded to latent codes $\mathcal{Y} = \{y_i\}_{i=1}^N$, our goal is to construct an embedding tree, exemplified by Fig. 3, where nodes near the root represent widely shared visual

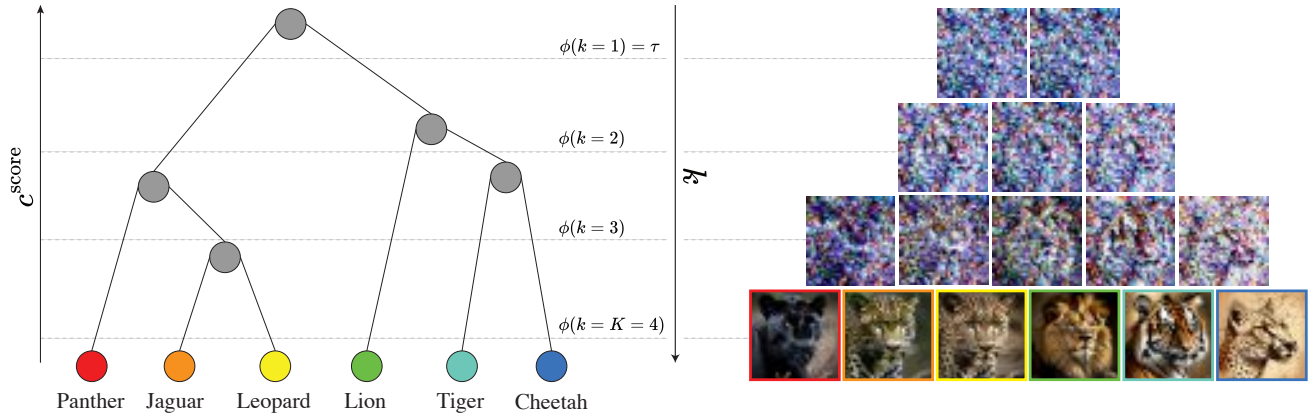

Figure 3. **Tree Traversal**. *Left*: our approach relies on a tree structure obtained by running agglomerative clustering on a set of prompt embeddings. Each node in the tree contains the average of the embeddings of its children, and a heterogeneity score c^{score} based on the distance between its two children. To connect the tree hierarchy to the denoising steps, we design a function ϕ taking as input the denoising step k, and controlling via its output and c^{score} which level to use in the tree for step k. *Right*: As a result, early diffusion steps are shared using the averaged embeddings, and the denoising steps gradually diverge to individual prompt embeddings, resulting in saved computation while maintaining high image generation quality.

concepts, while nodes at lower levels specialize on more specific concepts shared only by a few prompts, culminating with leaf-level nodes corresponding to input prompts y_i.

We construct the embedding tree using agglomerative clustering [10]:

1. Initialize leaf nodes with embeddings corresponding directly to input prompts: $c_1, \ldots, c_N = \{y_1\}, \ldots, \{y_N\}$.
2. Select two clusters with the smallest distance between them. Specifically, find clusters (c_a, c_b) such that:
$$(c_a, c_b) = \underset{c_a, c_b \in C, c_a \neq c_b}{\operatorname{argmin}} \; d(c_a, c_b), \tag{3}$$
where the distance $d(c_a, c_b)$ is measured by cosine similarity between the mean embeddings of clusters:
$$d(c_a, c_b) = 1 - \frac{\bar{e}_{c_a} \cdot \bar{e}_{c_b}}{\|\bar{e}_{c_a}\| \|\bar{e}_{c_b}\|}, \quad \text{where} \quad \bar{e}_c = \frac{1}{|c|} \sum_{y \in c} y. \tag{4}$$
3. Merge clusters (c_a, c_b) to form a new cluster $c_{ab} = c_a \cup c_b$ and assign it an embedding equal to the mean of its constituent embeddings $\bar{y}_{c_{ab}}$ computed as above.
4. Repeat the merging steps until a cluster containing all embeddings is added to the embedding tree.

The compute spent in constructing the tree is negligible compared to the cost of the denoising diffusion process.

Adaptive Embedding Selection. Once the embedding tree is constructed, we define a function $f(y, k)$ to select the cluster whose centroid embedding will be used at a step k to generate an image conditioned on y. At the final diffusion step ($k = K$), the diffusion step uses their leaf node embeddings. At earlier steps, embeddings from higher-level ancestor nodes are selected.

We introduce a cluster heterogeneity score based on the distance between merged child clusters:

$$c^{\text{score}} = d(c_a, c_b), \tag{5}$$

where c_a, c_b are child clusters merged to form node c. Lower heterogeneity indicate prompts within the cluster are semantically closer, suitable for guidance during earlier diffusion steps. Note that, by construction, the heterogeneity is always decreasing on any path from the root node to leaf.

During the diffusion process, we traverse the tree from the root to leaf nodes, using the heterogeneity score to decide how many diffusion steps should be spent at each level of the tree. More precisely, for each diffusion step k, we connect the k to an acceptable level of heterogeneity through a function ϕ, and select the child of the lowest node level that satisfies this constraint. Formally, given a prompt embedding y, for each diffusion step k, we select the embedding from the node satisfying:

$$\begin{aligned} f(y, k) = \underset{c}{\operatorname{argmin}} \, c^{\text{score}}, \text{ s.t.} \\ [\mathcal{P}\text{arent}(c)]^{\text{score}} \geq \phi(k) \wedge y \in c, \end{aligned} \tag{6}$$

where $\phi(k)$ decreases linearly as the diffusion progresses:

$$\phi(k) = \tau \cdot \left(1 - \frac{k}{K}\right). \tag{7}$$

Here, τ is a hyperparameter ranging from 0 to c^{max}, the score of the root node, that controls how quickly embeddings become specialized. A lower τ forces embeddings to specialize earlier, resulting in lower savings and higher image quality, while a higher τ allows prompts to share embeddings longer, resulting in higher savings and lower image quality. In our experiments, high-quality generation and high savings can be achieved with a value of $\tau = 1$.

We integrate this hierarchical embedding selection into

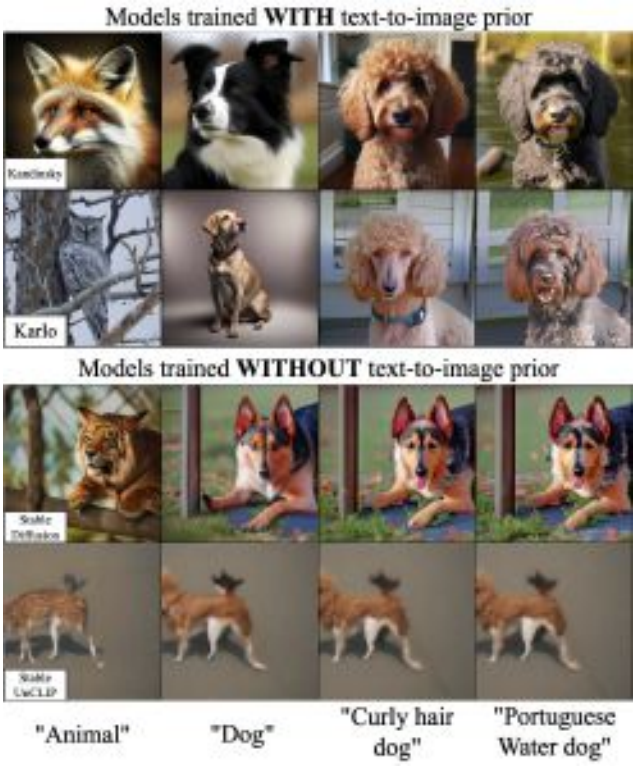

Figure 4. **Importance of text-to-image prior**. We share diffusion steps using a right-splitting binary tree as our hierarchy and the text prompts shown above for text-to-image diffusion models with various architectures. Kandinsky and Karlo are trained using an UnCLIP-style text-to-image prior. Stable diffusion does not use a text-to-image prior. While Stable UnCLIP does use a text-to-image prior during inference, the model was finetuned from Stable Diffusion and thus did not use it in training. Models trained with a text-to-image prior exhibit significantly higher quality and diversity with our approach than model trained without such prior.

standard diffusion inference. The denoising process can be written as

$$\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I}),$$
$$\mathbf{x}_{t-1} \sim \mathcal{N}\left(a_t \mathbf{x}_t - b_t \epsilon_\theta(\mathbf{x}_t; t, \bar{e}_{f(y,k)}), \sigma_t^2 \mathbf{I}\right). \quad (8)$$

Notice that while performing these steps to generate a set of images, we can keep track of evaluations of $\epsilon_\theta(\mathbf{x}_t; t, \bar{e}_{f(y,k)})$ and make sure that they are not done more than once, which significantly reduces computational cost without sacrificing image quality (see Sec. 5).

5. Experiments

We demonstrate our method's generation quality and compute savings on various benchmarks and applications. We present both quantitative and qualitative comparisons that show our method outperforms standard diffusion sampling, producing equal or higher quality generations using fewer

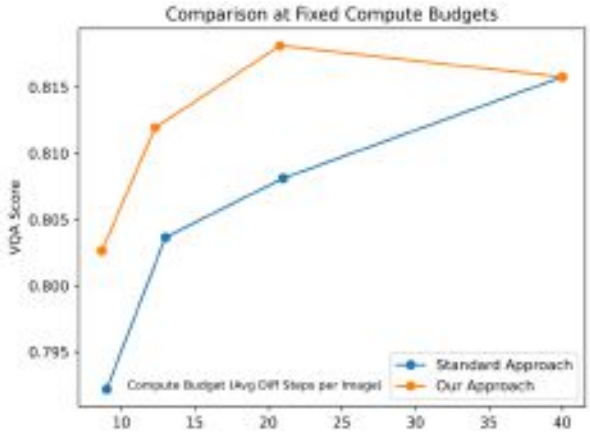

Figure 5. **Generation Quality at Fixed Compute Budgets**. We report VQA Score [17] on the Prompt Template Dataset [5] for both our method (orange) and the standard approach (blue) over various diffusion step budgets. Note that at a 40-step budget, our approach is identical to the standard approach. For all other fixed compute budgets, our method achieves higher VQA Scores (a measure of generation quality) than the standard approach.

Dataset	Num Images	Compute Saved ↑	Win % ↑
GenAI Bench	1600	50.43%	50.75%
Prompt Template	2000	69.25%	51.55%
Style Variations	100	73.93%	52.00%
Animals	500	62.80%	52.80%

Table 1. **Compute savings using our approach.** We run our method on four diverse datasets ranging from most general (GenAI Bench [15]) to more structured (Style Variations) and compute the VQA Score [17] for both our method and the standard approach. For each dataset, we report the number of images, the percentage of compute we save using our approach relative to standard denoising, and the percentage of images on which our approach achieves a higher VQA Score. From the win percentages, we can see that our method consistently produces comparable if not higher quality results to the standard approach, all while using significantly less compute (as evidenced by the compute saved percentages).

total denoising steps for sets of images. We ablate key components of our method such as our semantic clustering and the condition modality of the underlying diffusion model. Finally, we show multiple applications of our method for real world personal and commercial tasks.

5.1. Quantitative Evaluation

Experimental and Evaluation Setup. Throughout this section we look to evaluate both generation quality and compute savings of our method. Since our method focuses on text-conditioned image generation, we use VQA Score [17] as a measure of generation quality. VQA Score uses a visual-question-answering model to compute the probability of a "Yes" answer to the question "Does this

figure show $\{p\}$" for a given image and its corresponding text prompt p. VQA Score significantly outperforms CLIP score in terms of alignment to human preference on text-to-image generation [15]. For our fixed compute experiments, we report this VQA score outright, while in experiments looking to establish comparative generation quality, we report the percentage of examples for which a given method achieves a higher VQA score. To evaluate compute savings, we report the percentage of steps saved relative to the standard approach. Unless specified otherwise, all experiments are run on the Kandinsky 2.2 model [27].

Datasets. We evaluate our method on several diverse collections of prompts for text-driven image set generation varying in size, structure, and generality. GenAI Bench [15] is a text-to-image generative AI dataset containing 1600 text prompts created by artists for the purpose of AI text-driven image generation. These prompts are diverse, challenging, and often require higher order reasoning such as logic and comparison. We also use three custom datasets to understand how our approach performs on text prompts with more clear cut similarity due to their formulaic nature. The Prompt Template dataset uses the technique of prompt templates [5] to generate 2000 variations of the prompt "A <art style> painting of <subject>, <location>" by combinatorial replacement of each bracketed component. The Animals dataset is a set of 500 prompts describing animals generated with an LLM. The Style Variations dataset is a set of 100 prompts describing image variations to the prompt "A child sitting on a swing, looking up at the stars, <style>" (see Fig. 7).

Generation Quality Under Fixed Compute Budget. In Fig. 5, we compare our method to standard diffusion using the Prompt Template dataset over multiple fixed compute budgets. At the full 40 step budget, each generation is allowed its own 40 steps and thus there is no sharing of compute leading our approach to become identical to standard diffusion. However, for fixed compute budgets less than this maximum 40 step budget, our method obtains significantly higher VQA scores indicating superior generation quality. Notably, our approach with a compute budget of an average approximately 31 steps per image produces higher VQA scores than the standard non-shared approach with a full 40 step budget. This implies that our method is able to achiever higher quality generation by sharing compute than by using standard diffusion. We hypothesize that this is due to average text prompt conditions at early stages in the diffusion process producing a more stable denoising trajectory.

Compute savings under fixed generation quality. While our method outperforms standard diffusion at fixed compute budgets, it also enjoys significant compute savings for a fixed level of image quality. Specifically, we investigate how much compute can be saved with our approach to achieve similar quality as the 40-step standard diffusion. We report both the percentage of average diffusion steps saved relative to the standard approach at 40 steps, as well as the percentage of images on which our approach achieves a higher VQA Score. From Tab. 1, we can observe that our method achieves comparable if not better quality on all four datasets while saving anywhere from 50% to 74% of the computational cost.

	Compute Saved ↑			
Dataset	GenAI Bench	Prompt Template	Style Variations	Animals
Random Clustering	5.43%	5.48%	3.98%	4.89%
Ours	**50.43%**	**69.25%**	**73.93%**	**62.80%**

Table 2. **Clustering Ablation.** We ablate the our semantic clustering algorithm by comparing to a baseline approach that clusters based on random encodings. Similar to Table 1, we report the percentage of compute we save using our approach relative to standard denoising, both tuned to achieve quality similar to 40-step diffusion (measured by a VQA score of about 50%). For all datasets, our clustering algorithm saves massively more compute.

5.2. Qualitative Comparisons

In addition the quantitative metrics presented above, we show qualitative comparisons of our approach with standard diffusion sampling for both a fixed compute budget of 18 steps and a full 40 step budget. In Fig. 6, we run these methods on a set of 500 LLM-generated text prompts and display 9 randomly selected images. Enforcing a compute budget of 18 steps in standard diffusion produces oversmoothed images that lack fine-grained details. In contrast, our method is able to generate sharp features such as the wrinkles on the man's shirt (bottom row, middle column) or the sharp contours of the clouds (top row, middle column). When compared to the full 40 step generation with standard diffusion, our method uses significantly less compute, while producing images of comparable detail and quality.

5.3. Ablations

We run several ablations on key components of our method to demonstrate the effect of these design choices. To highlight the importance of our semantic clustering approach, we pass random embeddings to the hierarchical clustering algorithm and report the results in Tab. 2. Since every cluster is more heterogeneous, the random baseline diverges very early to the leaf nodes which leads to little compute being saved.

We also evaluate our approach across various backbone models. As seen in Fig. 2, models trained with a text-to-image prior generate structure more evenly during the dif-

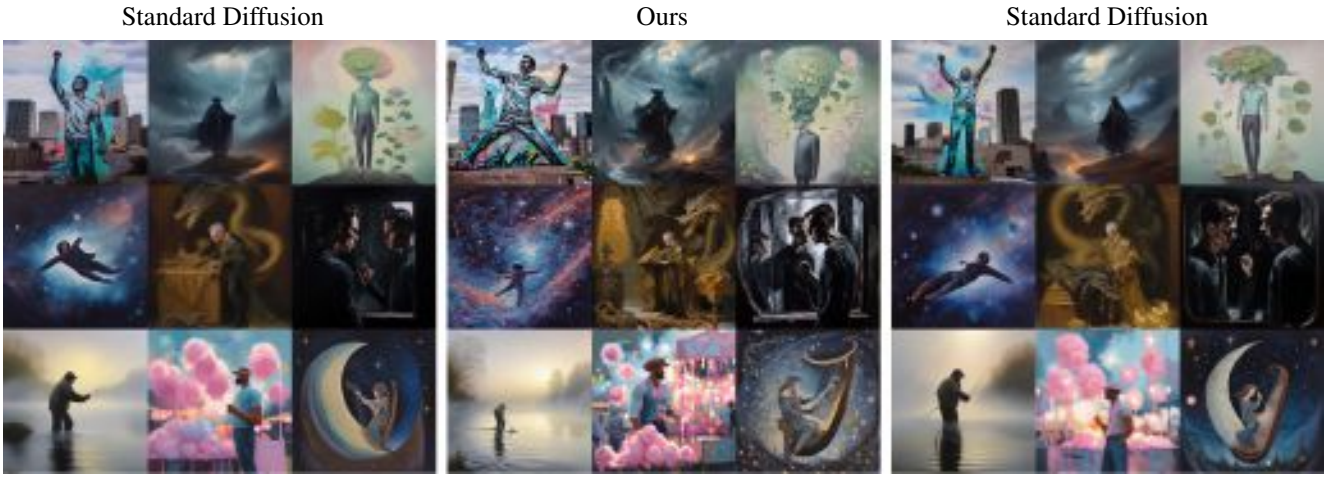

Figure 6. **Qualitative Comparison**. We compare our approach (middle) with standard diffusion for a fixed compute budget of 18 steps (left) and a full compute budget of 40 steps (right). For the fixed compute budget, our method produces higher quality images than the standard approach. When compared to the 40 step compute budget, our approach uses significantly less steps while generating images of comparable if not higher quality.

fusion process than models that are not. Empirically, we observe that our approach performs much better on these models. In Fig. 4, we run our approach on the set of four prompts "Animal," "Dog," "Curly hair dog," and "Portuguese water dog." Since the embeddings for "Curly hair dog" and "Portuguese water dog" are very similar, these two generations share most of the diffusion steps and only diverge on the final denoising stages to their respective prompts. In models trained with a text-to-image prior such as Kandinsky (first row) and Karlo (second row), this is not an issue as the structure is not fully set and fine-grained details can be generated in the remaining steps. Conversely, for models that were not trained with this prior such as Stable Diffusion (third row) and Stable UnCLIP (fourth row), there are not enough remaining steps to properly distinguish these two.

Finally, we investigate different values of τ for the mapping function ϕ of Eq. (7). Since the diffusion steps are evenly spaced out between $\phi(1) = \tau$ and $\phi(K) = 0$, by increasing or decreasing τ, we can control where on the hierarchy the diffusion steps are concentrated and thus the amount of steps shared by our approach. Using this, we can effectively control our compute budget. In Fig. 5, we show our approach (orange) for τ values $[0, 0.5, 1, 1.5]$. While we leave τ as a hyperparameter users can tune to achieve a desired trade-off between savings and quality, we found that a value of $\tau = 1.0$ worked well in most scenarios. We also investigated using non-linear mappings for ϕ such as a log scale, however in practice, the linear map worked best.

5.4. Applications

In Figs. 7 to 9, we present three applications of our method to efficiently generate image variations using prompt templates, including stylistic and structural variations. Our approach achieves a compute saving of 74% compared to the standard method when generating 100 style variations, 76% for 500 subject variations, and 65.3% for the 16 virtual try-on variations. Notably, the latter demonstrates significant

``A child sitting on a swing, looking up at the stars, <style>"

Figure 7. **Image Style Variations**. We show an application of our method for efficiently generating style variations on given input prompt. We show a subset of 100 generated images, saving 74% of the equivalent compute for the standard approach.

Figure 8. **Virtual try-on**. Our method generates a set of only 16 images, all containing the same subject, with various accessories, saving 65.3% of the equivalent compute for the standard approach.

``An image of a <animal> wearing a birthday hat with a cake and birthday candles"

Figure 9. **Subject Variation**. We show an application of our method for efficiently generating subject variations on given input prompt. We show a subset of 500 generated images, saving 76% of the equivalent compute for the standard approach.

computational efficiency even with relatively small image sets, provided they share common structural content. From a user perspective, efficiently generating image variations is helpful for iterating and co-creating with the diffusion model.

5.5. Limitations and Future Work

Our approach relies on factorizing denoising steps across similar samples, which introduces two primary limitations. First, our method is particularly suited to diffusion models where detail generation is evenly distributed throughout the diffusion process. As illustrated in Fig. 2, Stable Diffusion [28] tends to establish structural information very early in sampling, significantly reducing the opportunities for factorizing denoising steps effectively. Second, our technique excels in scenarios where the image set is either large (and possibly diverse) or small but homogeneous; however, its performance diminishes in situations where the image set is small yet highly diverse. In these cases, factorization of computational effort becomes less beneficial, leading to reduced efficiency.

While our method is primarily designed for diffusion, we are interested in extending it to autoregressive models (AR). In particular, we anticipate that recent autoregressive models such as the VAR approach [34], which generate images in a coarse-to-fine manner via next-scale prediction, could also be adapted to benefit from sharing computation.

6. Conclusion

We introduced a method to reuse computational resources when generating multiple images using diffusion models. Our key insight is that, if images contain similar content, they can share early stages of the denoising process. Leveraging this idea, we developed a technique that organizes target prompts into a hierarchical tree structure, where each node represents the average text embedding of its associated leaf prompts. This structure enables effective computation factorization during image generation. Through empirical analysis, we observed that diffusion models trained with text-to-image priors are particularly suitable for computation factorization. For this category of models, our approach consistently achieves at least 50% computational savings for diverse sets of prompts and even greater savings for more homogeneous sets, while preserving image quality. We demonstrated various practical applications, including generating image variations based on style and content.

We hope that this research will inspire further studies aimed at enhancing the efficiency of generative models, reducing environmental impact, and lowering carbon footprints.

Acknowledgments. We thank Richard Zhang for insights on evaluation metrics and more generally, the members of Adobe Research and 3DL for their insightful feedback. This work was supported by Adobe Research and NSF grant 2140001.

References

[1] Adobe. Adobe firefly, 2023. Accessed: 2024. 1

[2] Shm Garanganao Almeda, JD Zamfirescu-Pereira, Kyu Won Kim, Pradeep Mani Rathnam, and Bjoern Hartmann. Prompting for discovery: Flexible sense-making for ai art-making with dreamsheets. In *CHI Conference on Human Factors in Computing Systems*, 2024. 1, 3

[3] Stephen Brade, Bryan Wang, Mauricio Sousa, Sageev Oore, and Tovi Grossman. Promptify: Text-to-image generation through interactive prompt exploration with large language models. In *Proceedings of the 36th Annual ACM Symposium on User Interface Software and Technology*, 2023. 1, 3

[4] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018. 2

[5] Minsuk Chang, Stefania Druga, Alexander J Fiannaca, Pedro Vergani, Chinmay Kulkarni, Carrie J Cai, and Michael Terry. The prompt artists. In *Proceedings of the 15th Conference on Creativity and Cognition*, pages 75–87, 2023. 3, 5, 6

[6] Siddhartha Datta, Alexander Ku, Deepak Ramachandran, and Peter Anderson. Prompt expansion for adaptive text-to-image generation. *arXiv preprint arXiv:2312.16720*, 2023. 1, 3

[7] Patrick Esser, Sumith Kulal, Andreas Blattmann, Rahim Entezari, Jonas Müller, Harry Saini, Yam Levi, Dominik Lorenz, Axel Sauer, Frederic Boesel, et al. Scaling rectified flow transformers for high-resolution image synthesis. In *Forty-first international conference on machine learning*, 2024. 1

[8] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014. 2

[9] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020. 2, 3

[10] Joe H. Ward Jr. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):236–244, 1963. 4

[11] Minguk Kang, Jun-Yan Zhu, Richard Zhang, Jaesik Park, Eli Shechtman, Sylvain Paris, and Taesung Park. Scaling up gans for text-to-image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10124–10134, 2023. 2

[12] Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. *Advances in neural information processing systems*, 35:26565–26577, 2022. 3

[13] Dongjun Kim, Chieh-Hsin Lai, Wei-Hsiang Liao, Naoki Murata, Yuhta Takida, Toshimitsu Uesaka, Yutong He, Yuki Mitsufuji, and Stefano Ermon. Consistency trajectory models: Learning probability flow ode trajectory of diffusion. *ICLR*, 2024. 3

[14] Donghoon Lee, Jiseob Kim, Jisu Choi, Jongmin Kim, Minwoo Byeon, Woonhyuk Baek, and Saehoon Kim. Karlo-v1.0.alpha on coyo-100m and cc15m. https://github.com/kakaobrain/karlo, 2022. 2, 3

[15] Baiqi Li, Zhiqiu Lin, Deepak Pathak, Jiayao Li, Yixin Fei, Kewen Wu, Tiffany Ling, Xide Xia, Pengchuan Zhang, Graham Neubig, et al. Genai-bench: Evaluating and improving compositional text-to-visual generation. *arXiv preprint arXiv:2406.13743*, 2024. 5, 6

[16] Shanchuan Lin, Anran Wang, and Xiao Yang. Sdxl-lightning: Progressive adversarial diffusion distillation. *arXiv preprint arXiv:2402.13929*, 2024. 1, 3

[17] Zhiqiu Lin, Deepak Pathak, Baiqi Li, Jiayao Li, Xide Xia, Graham Neubig, Pengchuan Zhang, and Deva Ramanan. Evaluating text-to-visual generation with image-to-text generation. In *European Conference on Computer Vision*, pages 366–384. Springer, 2024. 5

[18] Luping Liu, Yi Ren, Zhijie Lin, and Zhou Zhao. Pseudo numerical methods for diffusion models on manifolds. *arXiv preprint arXiv:2202.09778*, 2022. 3

[19] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver: A fast ode solver for diffusion probabilistic model sampling in around 10 steps. *Advances in Neural Information Processing Systems*, 35:5775–5787, 2022.

[20] Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, and Jun Zhu. Dpm-solver++: Fast solver for guided sampling of diffusion probabilistic models. *arXiv preprint arXiv:2211.01095*, 2022. 3

[21] Sasha Luccioni, Yacine Jernite, and Emma Strubell. Power hungry processing: Watts driving the cost of ai deployment? In *Proceedings of the 2024 ACM conference on fairness, accountability, and transparency*, 2024. 1

[22] Simian Luo, Yiqin Tan, Longbo Huang, Jian Li, and Hang Zhao. Latent consistency models: Synthesizing high-resolution images with few-step inference. *ICLR*, 2024. 2, 3

[23] Atefeh Mahdavi Goloujeh, Anne Sullivan, and Brian Magerko. Is it ai or is it me? understanding users' prompt journey with text-to-image generative ai tools. In *CHI Conference on Human Factors in Computing Systems*, 2024. 3

[24] Midjourney, Inc. Midjourney, 2022. Accessed: 2024. 1

[25] OpenAI. Dall-e, 2021. Accessed: 2024. 1, 2

[26] Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 1 (2):3, 2022. 2, 3

[27] Anton Razzhigaev, Arseniy Shakhmatov, Anastasia Maltseva, Vladimir Arkhipkin, Igor Pavlov, Ilya Ryabov, Angelina Kuts, Alexander Panchenko, Andrey Kuznetsov, and Denis Dimitrov. Kandinsky: an improved text-to-image synthesis with image prior and latent diffusion. *EMNLP demoes*, 2023. 1, 2, 3, 6

[28] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022. 1, 2, 3, 8

[29] Runway AI. Runway: Ai magic tools, 2022. Accessed: 2024. 1

[30] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, et al. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in neural information processing systems*, 35:36479–36494, 2022. 1, 2

[31] Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. *ICLR*, 2022. 3

[32] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. pmlr, 2015. 2, 3

[33] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. *ICLR*, 2021. 3

[34] Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *Advances in neural information processing systems*, 37:84839–84865, 2025. 8

[35] Arkhipkin Vladimir, Viacheslav Vasilev, Andrei Filatov, Igor Pavlov, Julia Agafonova, Nikolai Gerasimenko, Anna Averchenkova, Evelina Mironova, Bukashkin Anton, Konstantin Kulikov, Andrey Kuznetsov, and Denis Dimitrov. Kandinsky 3: Text-to-image synthesis for multifunctional generative framework. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 475–485, Miami, Florida, USA, 2024. Association for Computational Linguistics. 1, 2, 3

[36] Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Fredo Durand, William T Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6613–6623, 2024. 2, 3

[37] Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Fredo Durand, and Bill Freeman. Improved distribution matching distillation for fast image synthesis. *Advances in Neural Information Processing Systems*, 37: 47455–47487, 2025. 3

[38] Wenliang Zhao, Lujia Bai, Yongming Rao, Jie Zhou, and Jiwen Lu. Unipc: A unified predictor-corrector framework for fast sampling of diffusion models. *Advances in Neural Information Processing Systems*, 36:49842–49869, 2023. 3

Reusing Computation in Text-to-Image Diffusion for Efficient Generation of Image Sets

Supplementary Material

A. Further Method Details

We provide pseudocode for our hierarchical diffusion algorithm. Our inputs are as follows: τ is a hyperparameter controlling the tradeoff between savings and quality, K is the total number of diffusion steps, C is our set of clusters obtained through agglomerative clustering, C^{scores} are the corresponding c^{score}s for each cluster in C, and Y is our set of all text prompts. The additional variables in line 18 are standard diffusion variables: a_k and b_k are scheduler coefficients, ϵ_θ is the diffusion model, and σ_{k-1} is a time-step-dependent standard deviation. Note, this algorithm assumes that $\tau \leq max_c(c^{score})$ otherwise a dummy parent node is required above the root node with a corresponding $c^{score} > \tau$. Our method returns the image from step K for each leaf cluster containing an individual prompt.

Algorithm 1 Hierarchical Diffusion.

1: Inputs: $\tau, K, C, C^{\text{scores}}, Y$
2: Let $\phi(k) = \tau \cdot \left(1 - \frac{k}{K}\right)$ assign scores to each diffusion step $k \in 1...K$ s.t. the scores of the K steps are evenly spaced over the interval $[\tau, 0]$.
3: **for** k in $1...K$ **do**:
4: $\quad C_{\text{k}} \leftarrow []$
5: $\quad$ **for** y in Y **do**:
6: $\quad\quad c' = \underset{c \in C}{\text{argmin}}\, c^{\text{score}}$ s.t. $[\mathcal{P}\text{arent}(c)]^{\text{score}} \geq \phi(k) \wedge y \in c$
7: $\quad\quad$ **if** c' not in C_{k} **then**
8: $\quad\quad\quad C_{\text{k}}$.append($c'$)
9: $\quad\quad$ **end if**
10: $\quad$ **end for**
11: $\quad$ **for** c in C_{k} **do**
12: $\quad\quad$ **if** k==1 **then**
13: $\quad\quad\quad \mathbf{x}_{k-1}^{c} \sim N(0, \mathbf{I})$
14: $\quad\quad$ **else**
15: $\quad\quad\quad \mathbf{x}_{k-1}^{c} \leftarrow \mathbf{x}_{k-1}^{\mathcal{P}\text{arent}(c)}$
16: $\quad\quad$ **end if**
17: $\quad\quad \bar{y} = \text{mean}(y : y \in c)$
18: $\quad\quad \mathbf{x}_{k}^{c} \sim \mathcal{N}\left(a_k \mathbf{x}_{k-1}^{c} - b_k \epsilon_\theta(\mathbf{x}_{k-1}^{c}; k-1, \bar{y}), \sigma_{k-1}^2 \mathbf{I}\right)$
19: $\quad$ **end for**
20: **end for**
21: **return** $\{\mathbf{x}_K^{\{y\}} : y \in Y\}$

B. Additional Experiments

Experiments on Additional Models. Due to the coarse-to-fine generation properties exhibited by models conditioned on a text-image prior, our approach works best on the Kandinsky and Karlo models which are trained in this manner (See Fig. 4). However, we can still achieve savings using more standard models such as SD 1.5 and models using the more modern DiT architecture and Flow Matching scheduler such as FLUX. While these models generate high frequency details earlier on in the diffusion process and thus leave less room for shared compute (see Fig. 2), we still achieve moderate savings over standard diffusion inference. For comparable quality results (better VQA scores in $\approx 50\%$ of the samples), we save up to **28.25%** and **23.73%** on compute when using SD 1.5 and FLUX, respectively. We report results for both models on all four datasets in the table below.

Model	Dataset	GenAI B.	Prompt T.	Style V.	Animals
SD 1.5	Savings ↑	10.4%	28.25%	23.55%	11.70%
	Win % ↑	48%	50%	49%	50%
FLUX	Savings ↑	2.33%	3.61%	23.73%	6.28%
	Win % ↑	45%	45%	51%	48%

Diversity of Generations. Examining sample generations from both our approach and standard diffusion in Figs. 1 and 6, we observe that our generations qualitatively display comparable diversity to the results from standard diffusion. Since our approach leverages similarities across prompts to save compute in the generation process, when used at small scales, our generations can exhibit high similarity between examples (see Fig. 8). However, as we increase the size of our datasets such as with the GenAI Bench or Prompt Templates datasets, our diversity significantly increases. To quantitatively evaluate generation diversity, we randomly sample 100 images from each generated dataset and compute the CLIP cosine similarity between all possible pairs. The reported result is the mean similarity. On larger datasets (GenAI Bench, Prompt Templates, and Animals which all have $\geq$ 500 examples), our method achieves comparable diversity to the standard approach. While our results on the Style Variations dataset are less diverse, this is expected since the dataset contains only 100 examples and the highly structured and similar prompts naturally lend themselves to less diverse generations.

Dataset	GenAI B.	Prompt T.	Style V.	Animals
Std. CLIP Diversity ↓	0.6082	0.6508	**0.8071**	0.6652
Our CLIP Diversity ↓	0.6027	0.6493	0.8374	0.6552

Figure 10. De-noised mean embedding (left) and its children using the Kandinsky model.

Visualization of Mean Embedding. We analyze the mean cluster embedding by using it to guide image generation. As we can see in Fig. 10, even though this embedding (left) is not associated with any particular prompt, it yields an image corresponding to a reasonable visual mixture of its children. In practice, this "mean image" is never generated (fully denoised) – we show it here as a tool to visualize the semantics of that embedding.