WÜRSTCHEN: AN EFFICIENT ARCHITECTURE FOR LARGE-SCALE TEXT-TO-IMAGE DIFFUSION MODELS

Pablo Pertinas[*]
Indpendent researcher, Sant Joan d'Alacant, Spain

Dominic Rampas[*]
Technische Hochschule Ingolstadt, Ingolstadt, Germany
Wand Technologies Inc., New York, USA

Mats L. Richter[*]
Université de Montréal, Montreal, Canada
Mila, Quebec AI Institute, Montreal, Canada

Christopher J. Pal
Polytechnique Montréal, Montreal, Canada
Mila, Quebec AI Institute, Quebec, Canada
Canada CIFAR AI Chair

Marc Aubreville
Technische Hochschule Ingolstadt, Ingolstadt, Germany

ABSTRACT

We introduce Würstchen, a novel architecture for text-to-image synthesis that combines competitive performance with unprecedented cost-effectiveness for large-scale text-to-image diffusion models. A key contribution of our work is to develop a latent diffusion technique in which we learn a detailed but extremely compact semantic image representation used to guide the diffusion process. This highly compressed representation of an image provides much more detailed guidance compared to latent representations of language and this significantly reduces the computational requirements to achieve state-of-the-art results. Our approach also improves the quality of text-conditioned image generation based on our user preference study. The training requirements of our approach consists of 24,602 A100-GPU hours – compared to Stable Diffusion 2.1's 200,000 GPU hours. Our approach also requires less training data to achieve these results. Furthermore, our compact latent representations allows us to perform inference over twice as fast, slashing the usual costs and carbon footprint of a state-of-the-art (SOTA) diffusion model significantly, without compromising the end performance. In a broader comparison against SOTA models our approach is substantially more efficient and compares favorably in terms of image quality. We believe that this work motivates more emphasis on the prioritization of both performance and computational accessibility.

1 INTRODUCTION

State-of-the-art diffusion models (Ho et al., 2020; Saharia et al., 2022; Ramesh et al., 2022) have advanced the field of image synthesis considerably, achieving remarkable results that closely approxi-

[*]equal contribution

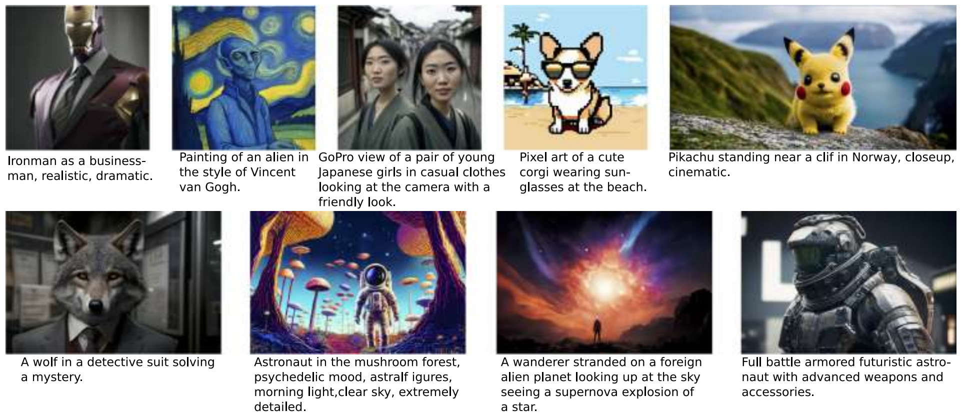

Ironman as a business-man, realistic, dramatic.
Painting of an alien in the style of Vincent van Gogh.
GoPro view of a pair of young Japanese girls in casual clothes looking at the camera with a friendly look.
Pixel art of a cute corgi wearing sun-glasses at the beach.
Pikachu standing near a clif in Norway, closeup, cinematic.
A wolf in a detective suit solving a mystery.
Astronaut in the mushroom forest, psychedelic mood, astralf igures, morning light,clear sky, extremely detailed.
A wanderer stranded on a foreign alien planet looking up at the sky seeing a supernova explosion of a star.
Full battle armored futuristic astro-naut with advanced weapons and accessories.

Figure 1: Text-conditional generations using Würstchen. Note the various art styles and aspect ratios.

mate photorealism. However, these foundation models, while impressive in their capabilities, carry a significant drawback: they are computationally demanding. For instance, Stable Diffusion (SD) 1.4, one of the most notable models in the field, used 150,000 GPU hours for training (Rombach & Esser, 2022). While more economical text-to-image models do exist (Ding et al., 2021; 2022; Tao et al., 2023; 2022), the image quality of these models can be considered inferior in terms of lower resolution and overall aesthetic features.

The core dilemma for this discrepancy is that increasing the resolution also increases visual complexity and computational cost, making image synthesis more expensive and data-intensive to train. Encoder-based Latent Diffusion Models (LDMs) partially address this by operating on a compressed latent space instead of directly on the pixel-space (Rombach et al., 2022), but are ultimately limited by how much the encoder-decoder model can compress the image without degradation (Richter et al., 2021a).

Against this backdrop, we propose a novel three-stage architecture named ”Würstchen”, which drastically reduces the computational demands while maintaining competitive performance. We achieve this by training a diffusion model on a very low dimensional latent space with a high compression ratio of 42:1. This very low dimensional latent-space is used to condition the second generative latent model, effectively helping it to navigate a higher dimensional latent space of a Vector-quantized Generative Adversarial Network (VQGAN), which operates at a compression ratio of 4:1. More concretely, the approach uses three distinct stages for image synthesis (see Figure 2): initially, a text-conditional LDM is used to create a low dimensional latent representation of the image (Stage C). This latent representation is used to condition another LDM (Stage B), producing a latent image in a latent space of higher dimensionality. Finally, the latent image is decoded by a VQGAN-decoder to yield the full-resolution output image (Stage A).

Training is performed in reverse order to the inference (Figure 3): The initial training is carried out on Stage A and employs a VQGAN to create a latent space. This compact representation facilitates learning and inference speed (Rombach et al., 2022; Chang et al., 2023; Rampas et al., 2023). The next phase (Stage B) involves a first latent diffusion process (Rombach et al., 2022), conditioned on the outputs of a Semantic Compressor (an encoder operating at a very high spatial compression rate) and on text embeddings. This diffusion process is tasked to reconstruct the latent space established by the training of Stage A, which is strongly guided by the detailed semantic information provided by the Semantic Compressor. Finally, for the construction of Stage C, the strongly compressed latents of the Semantic Compressor from Stage B are used to project images into the condensed latent space where a text-conditional LDM (Rombach et al., 2022) is trained. The significant reduction in space dimensions in Stage C allows for more efficient training and inference of the diffusion model, considerably reducing both the computational resources required and the time taken for the process.

Our proposed Würstchen model thus introduces a thoughtfully designed approach to address the high computational burden of current state-of-the-art models, providing a significant leap forward in text-to-image synthesis. With this approach we are able to train a 1B parameter Stage C text-conditional diffusion model within approximately 24,602 GPU hours, resembling a 8x reduction in computation compared to the amount SD 2.1 used for training (200,000 GPU hours), while showing similar fidelity both visually and numerically. Throughout this paper, we provide a comprehensive evaluation of Würstchen’s efficacy, demonstrating its potential to democratize the deployment & training of high-quality image synthesis models.

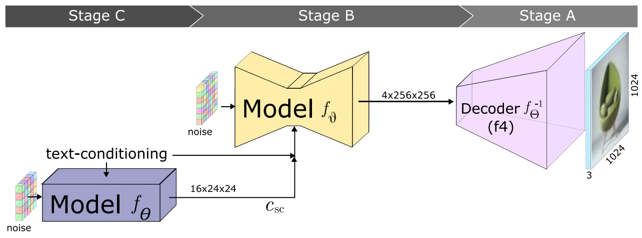


Figure 2: Inference architecture for text-conditional image generation.

Our main contributions are the following:

1. We propose a novel three-stage architecture for text-to-image synthesis at strong compression ratio, consisting of two conditional latent diffusion stages and a latent image decoder.
2. We show that by using a text-conditional diffusion model in a strongly compressed latent space we can achieve state-of-the-art model performance at a significantly reduced training cost and inference speed.
3. We provide comprehensive experimental validation of the model's efficacy based on automated metrics and human feedback.
4. We are publicly releasing the source code and the entire suite of model weights.

2 Related Work

2.1 Conditional Image Generation

The field of image generation guided by text prompts has undergone significant progression in recent years. Initial approaches predominantly leveraged Generative Adversarial Networks (GANs) (Reed et al., 2016; Zhang et al., 2017). More recently, however, a paradigm shift in the field of image generation towards diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020) has occurred. These approaches, in some cases, have not only met but even exceeded the performance of GANs in both conditional and unconditional image generation (Dhariwal & Nichol, 2021). Diffusion models put forth a score-based scheme that gradually eliminates perturbations (e.g., noise) from a target image, with the training objective framed as a reweighted variational lower-bound. Next to diffusion models, another dominant choice for training text-to-image models is transformers. In their early stages, transformer-based models utilized an autoregressive approach, leading to a significant slowdown in inference due to the requirement for each token to be sampled individually. Current strategies, however, employ a bidirectional transformer (Ding et al., 2022; Chang et al., 2022; 2023) to address the challenges that traditional autoregressive models present. As a result, image generation can be executed using fewer steps, while also benefiting from a global context during the generative phase. Other recent work has shown that convolution-based approaches for image generation can yield similar results (Rampas et al., 2023).

2.2 Compressed Latent Spaces

The majority of approaches in the visual modality of generative models use some way to train at a smaller space, followed by upscaling to high resolutions, as training at large pixel resolutions can become exponentially more expensive with the size of images. For text-conditional image generation, there are two established categories of approaches: encoder-based and upsampler-based. LDMs (Rombach et al., 2022), DALL-E (Ramesh et al., 2021), CogView (Ding et al., 2021; 2022), MUSE (Chang et al., 2023) belong to the first category and employ a two-stage training process. Initially, an autoencoder (Rumelhart et al., 1985) is trained to provide a lower-dimensional, yet perceptually equivalent, representation of the data. This representation forms the basis for the subsequent training of a diffusion or transformer model. Eventually, generated latent representations can be decoded with the decoder branch of the autoencoder to the pixel space. The result is a significant reduction in computational complexity for the diffusion/sampling process and efficient image decoding from the latent space using a single network pass. On the contrary, upsampler-based methods generate images at low resolution in the pixel space and use subsequent models for upscaling the images to higher

How Encoder based and upsample based text-conditional-image generation model work resolution. UnClip (Ramesh et al., 2022) and Imagen (Saharia et al., 2022) both generate images at 64x64 and upscale using two models to 256 and 1024 pixels. The former model is the largest in terms of parameter count, while the latter models are smaller due to working at higher resolution and only being responsible for upscaling.

2.3 Conditional Guidance

The conditional guidance of models in text-based scenarios is typically facilitated through the encoding of textual prompts via a pretrained language model. Two major categories of text encoders are employed: contrastive text encoders and uni-modal text encoders. Contrastive Language-Image Pretraining (CLIP) (Radford et al., 2021) is a representative of the contrastive multimodal models that strives to align text descriptions and images bearing semantic resemblance within a common latent space. A host of image generation methodologies have adopted a frozen CLIP model as their exclusive conditioning method in recent literature. The hierarchical DALL-E 2 by Ramesh et al. (2022) specifically harnesses CLIP image embeddings as input for their diffusion model, while a 'prior' performs the conversion of CLIP text embeddings to image embeddings. SD (Rombach et al., 2022), on the other hand, makes use of un-pooled CLIP text embeddings to condition its LDM. In contrast, the works of Saharia et al. (2022), Liu et al. (2022a) and Chang et al. (2023) leverage a large, uni-modal language model such as T5 (Raffel et al., 2020) or ByT5 (Xue et al., 2022) that can encode textual prompts with notable accuracy, leading to image generations of superior precision in terms of composition, style, and layout.

3 Method

Our method comprises three stages, all implemented as deep neural networks. For image generation, we first generate a latent image at a strong compression ratio using a text-conditional LDM (Stage C). Subsequently, this representation is transformed to a less-compressed latent space by the means of a secondary model which is tasked for this reconstruction (Stage B). Finally, the tokens that comprise the latent image in this intermediate resolution are decoded to yield the output image (Stage A). The training of this architecture is performed in reverse order, starting with Stage A, then following up with Stage B and finally Stage C (see Figure 3). Text conditioning is applied on Stage C using CLIP-H (Ilharco et al., 2021). Details on the training procedure can be found in Appendix E.

3.1 Stage A and B

It is a known and well-studied technique to reduce the computational burden by compressing data into a smaller representation(Richter et al., 2021a;b; Chang et al., 2022). Our approach follows this paradigm, too, and makes use of Stages A & B to achieve a notably higher compression than usual. Let $H \times W \times C$ be the dimensions of images. A spatial compression maps images to a latent representation with a resolution of $h \times w \times z$ with $h = H/f, w = W/f$, where f defines the compression rate. Common approaches for modeling image synthesis use a one-stage compression between $f4$ and $f16$ (Esser et al., 2021; Chang et al., 2023; Rombach et al., 2022), with higher factors usually resulting in worse reconstructions. Our Stage A consists of a $f4$ VQGAN (Esser et al., 2021) with parameters Θ and initially encodes images $\boldsymbol{X} \in \mathbb{R}^{3\times1024\times1024}$ into 256×256 discrete tokens from a learned codebook of size 8,192.

$$\boldsymbol{X}_q = f_\Theta(\boldsymbol{X})$$

The network is trained as described by Esser *et al.* and tries to reconstruct the image based on the quantized latents, so that:

$$f_\Theta^{-1}\left(f_\Theta\left(\boldsymbol{X}\right)\right) = f_\Theta^{-1}\left(\boldsymbol{X}_q\right) \approx \boldsymbol{X}$$

where f_Θ^{-1} resembles the decoder part of the VQGAN.

Afterward, the quantization is dropped from Stage A, and Stage B is trained in the unquantized latent space of the Stage A-encoder as a conditioned LDM. In stage B, we utilize a Semantic Compressor, i.e., an encoder-type network that is tasked to create latent representations at a strong spatial compression rate that can be used to create a latent representation to guide the diffusion process. The unquantized image embeddings are noised following an LDM training procedure. The

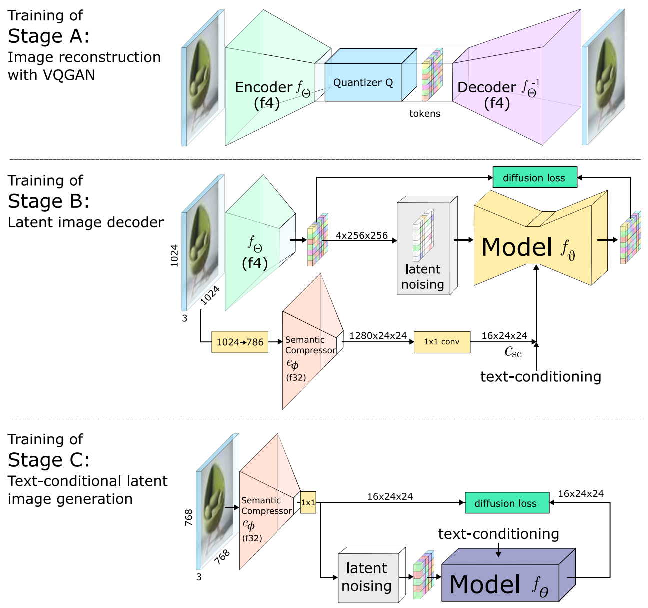


Figure 3: Training objectives of our model. Initially, a VQGAN is trained. Secondly, Stage B is trained as a diffusion model inside Stage A's latent space. Stage B is conditioned on text-embeddings and the output of the Semantic Compressor, which produces strongly downsampled latent representations of the same image. Finally, Stage C is trained on the latents of the Semantic Compressor as a text-conditional LDM, effectively operating on a compression ratio of $42:1$.

noised representation $\tilde{\boldsymbol{X}}_t$, together with the visual embeddings from the Semantic Compressor, $\boldsymbol{C}_{\text{sc}}$, text conditioning $\boldsymbol{C}_{\text{text}}$ and the timestep t are given to the model.

The highly compressed visual embeddings extracted by the Semantic Compressor will act as an interface for Stage C, which will be trained to generate them. The embeddings will have a shape of $\mathbb{R}^{1280\times24\times24}$ obtained by encoding images with shape $\boldsymbol{X}\in\mathbb{R}^{3\times786\times786}$. We use simple bicubic interpolation for the resizing of the images from 1024×1024 to 786×786, which is a sufficiently high resolution to fully utilize the parameters of the Semantic Compressor (Richter et al., 2023; Richter & Pal, 2022), while also reducing the latent representation size. Moreover, we further compress the latents with a 1×1 convolution that normalizes and projects the embeddings to $\boldsymbol{C}_{\text{sc}}\in\mathbb{R}^{16\times24\times24}$. This compressed representation of the images is given to the Stage B decoder as conditioning to guide the decoding process.

$$\bar{\boldsymbol{X}}_0 = f_\vartheta(\tilde{\boldsymbol{X}}_t, \boldsymbol{C}_{\text{sc}}, \boldsymbol{C}_{\text{text}}, t)$$

By conditioning Stage B on low-dimensional latent representations, we can effectively decode images from a $\mathbb{R}^{16\times24\times24}$ latent space to a resolution of $\boldsymbol{X}\in\mathbb{R}^{3\times1024\times1024}$, resulting in a total spatial compression of **42:1**.

We initialized the Semantic Compressor with weights pre-trained on ImageNet, which, however, does not capture the broad distribution of images present in large text-image datasets and is not well-suited for semantic image projection, since it was trained with an objective to discriminate the ImageNet categories. Hence we updated the weights of the Semantic Compressor during training, establishing a latent space with high-precision semantic information. We use Cross-Attention (Vaswani et al., 2017) for conditioning and project $\boldsymbol{C}_{\text{sc}}$ (flattened) to the same dimension in each block of the model and concatenate them. Furthermore, during training Stage B, we intermittently add noise to the Semantic Compressor's embeddings, to teach the model to understand non-perfect embeddings, which is likely to be the case when generating these embeddings with Stage C. Lastly, we also randomly drop $\boldsymbol{C}_{\text{sc}}$ to be able to sample with classifier-free-guidance (Ho & Salimans, 2022) during sampling.

3.2 Stage C

After Stage A and Stage B were trained, training of the text-conditional last stage started. In our implementation, Stage C consists of 16 ConvNeXt-block (Liu et al., 2022b) without downsampling, text and time step conditionings are applied after each block via cross-attention. We follow a standard diffusion process, applied in the latent space of the finetuned Semantic Compressor. Images are encoded into their latent representation $\boldsymbol{X}_{\text{sc}} = \boldsymbol{C}_{\text{sc}}$, representing the target. The latents are noised by using the following forward diffusion formula:

$$\boldsymbol{X}_{\text{sc},t} = \sqrt{\bar{\alpha}_t} \cdot \boldsymbol{X}_{\text{sc}} + \sqrt{1 - \bar{\alpha}_t} \cdot \epsilon$$

where ϵ represents noise from a zero mean unit variance normal distribution. We use a cosine schedule (Nichol & Dhariwal, 2021) to generate $\bar{\alpha}_t$ and use continuous timesteps. The diffusion model takes in the noised embeddings $\boldsymbol{X}_{\text{sc},t}$, the text conditioning $\boldsymbol{C}_{\text{text}}$ and the timestep t. The model returns the prediction for the noise in the following form:

$$\bar{\epsilon} = \frac{\boldsymbol{X}_{\text{sc},t} - \boldsymbol{A}}{\mid 1 - \boldsymbol{B} \mid + 1e^{-5}}$$

with

$$\boldsymbol{A}, \boldsymbol{B} = f_\theta(\boldsymbol{X}_{\text{sc},t}, \boldsymbol{C}_{\text{text}}, t)$$

We decided to formulate the objective as such, since it made the training more stable. We hypothesize this occurs because the model parameters are initialized to predict **0** at the beginning, enlarging the difference to timesteps with a lot of noise. By reformulating to the $\boldsymbol{A}$ & $\boldsymbol{B}$ objective, the model initially returns the input, making the loss small for very noised inputs. We use the standard mean-squared-error loss between the predicted noise and the ground truth noise. Additionally, we employ the p2 loss weighting (Choi et al., 2022):

$$p_2(t) \cdot \mid\mid \epsilon - \bar{\epsilon} \mid\mid^2$$

where $p_2(t)$ is defined as $\frac{1-\bar{\alpha}_t}{1+\bar{\alpha}_t}$, making higher noise levels contribute more to the loss. Text conditioning $\boldsymbol{C}_{\text{text}}$ are dropped randomly for 5% of the time and replaced with a null-label in order to use classifier-free-guidance (Ho & Salimans, 2022)

3.3 Image Generation (Sampling)

A depiction of the sampling pipeline can be seen in Figure 2. Sampling starts at Stage C, which is primarily responsible for image-synthesis (see Appendix D), from initial random noise $\boldsymbol{X}_{\text{sc},\tau_C} = \mathcal{N}(0, \mathbf{I})$. We use the DDPM (Ho et al., 2020) algorithm to sample the Semantic Compressor latents conditioned on text-embeddings. To do so, we run the following operation for τ_C steps:

$$\hat{\boldsymbol{X}}_{\text{sc},t-1} = \frac{1}{\sqrt{\alpha_t}} \cdot (\hat{\boldsymbol{X}}_{\text{sc},t} - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \bar{\epsilon}) + \sqrt{(1 - \alpha_t) \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t}} \epsilon$$

We denote the outcome as $\bar{\boldsymbol{X}}_{\text{sc}}$ which is of shape $16 \times 24 \times 24$. This output is flattened to a shape of 576×16 and given as conditioning, along with the same text embeddings used to sample $\bar{\boldsymbol{X}}_{\text{sc}}$, to Stage B. This stage operates at $4 \times 256 \times 256$ unquantized VQGAN latent space. We initialize $\boldsymbol{X}_{q,\tau_B}$ to random tokens drawn from the VQGAN codebook. We sample $\tilde{\boldsymbol{X}}$ for τ_B steps using the standard LDM scheme.

$$\tilde{\boldsymbol{X}}_{t-1} = f_\vartheta(\tilde{\boldsymbol{X}}_t, \boldsymbol{C}_{\text{sc}}, \boldsymbol{C}_{\text{text}}, t)$$

Finally $\tilde{\boldsymbol{X}}$ is projected back to the pixel space using the decoder f_Θ^{-1} of the VQGAN (Stage A):

$$\bar{\boldsymbol{X}} = f_\Theta^{-1}(\tilde{\boldsymbol{X}})$$

3.4 Model Decisions

Theoretically, any feature extractor could be used as backbone for the Semantic Compressor. However, we hypothesize that it is beneficial to use a backbone that already has a good feature representation of a wide variety of images. Furthermore, having a small Semantic Compressor makes training of Stage B & C faster. Finally, the feature dimension is vital. If it is excessively small, it may fail to capture sufficient image details or will underutilize parameters (Richter & Pal, 2022); conversely, if it is overly large, it may unnecessarily increase computational requirements and extend training duration (Richter et al., 2021a). For this reason, we decided to use an ImageNet1k pre-trained EfficientV2 (S) as the backbone for our Semantic Compressor, as it combines high compression with well generalizing feature representations and computational efficiency.

Furthermore, we deviate in Stage C from the U-Net standard architecture. As the image is already compressed by a factor of 42, and we find further compression harmful to the model quality. Instead, the model is a simple sequence of 16 ConvNeXt blocks (Liu et al., 2022b) without downsampling. Time and text conditioning is applied after each block.

4 Experiments and Evaluation

To demonstrate Würstchen's capabilities on text-to-image generation, we trained an 18M parameter Stage A, a 1B parameter Stage B and a 1B parameter Stage C. We employed an EfficientNet2-Small as Semantic Compressor (Tan & Le, 2020) during training. Stage B and C are conditioned on un-pooled CLIP-H (Ilharco et al., 2021) text-embeddings. The setup is designed to produce images of variable aspect ratio with up to 1538 pixels per side. All stages were trained on subsets of the improved-aesthetic LAION-5B (Schuhmann et al., 2022) dataset.

All the experiments use the standard DDPM (Ho et al., 2020) algorithm to sample latents in Stage B and C. Both stages also make use of classifier-free-guidance (Ho & Salimans, 2022) with guidance scale w. We fix the hyperparameters for Stage B sampling to $\tau_B = 12$ and $w = 4$, Stage C uses $\tau_C = 60$ for sampling. Images are generated using a 1024×1024 resolution.

Baselines To better assess the efficacy of our architecture, we additionally train a U-Net-based 1B parameter LDM on SD 2.1 first stage and text-conditioning model. We refer to this model as Baseline LDM, it is trained for $\approx$ 25,000 GPU-hours (same as Stage C) using an 512×512 input resolution.

Additionally, we evaluate our model against various state-of-the-art models that were publicly available at the time of writing (see Tables 1 and Table 2). All these models were used in their respective default configuration for text-to-image synthesis. Whenever possible, the evaluation metrics published by the original authors were used.

Evaluation Metrics We used the Fréchet Inception Distance (FID) (Heusel et al., 2018) and Inception Score (IS) to evaluate all our models on COCO-30K, similar to (Tao et al., 2023; Ding et al., 2021; 2022). For evaluating the FID score, all images were downsampled to 256×256 pixels to allow for a fair comparison between other models in the literature. However, both metrics suffer from inconsistencies and are known to be not necessarily well correlated with the aesthetic quality perceived by humans (Podell et al. (2023); Ding et al. (2021; 2022), see also Appendix B). For this reason, we chose PickScore (Kirstain et al., 2023) as our primary automated metric. PickScore is designed to imitate human preferences, when selecting from a set of images given the same prompt. We applied PickScore to compare Würstchen to various other models on various datasets. We provide the percentage of images, where PickScore preferred the image of Würstchen over the image of the other model. To also evaluate the environmental impact of our model we estimated the carbon emitted during training based on the work of (Lacoste et al., 2019).

Finally, we also conducted a study with human participants, where the participants chose between two images from two different models given the prompt.

Datasets To assess the zero-shot text-to-image capabilities of our model, we use three distinct sets of captions alongside their corresponding images. The COCO-validation is the de-facto standard dataset to evaluate the zero-shot performance for text-to-image models. For MS COCO we generate 30,000 images based on prompts randomly chosen from the validation set. We refer to this set of

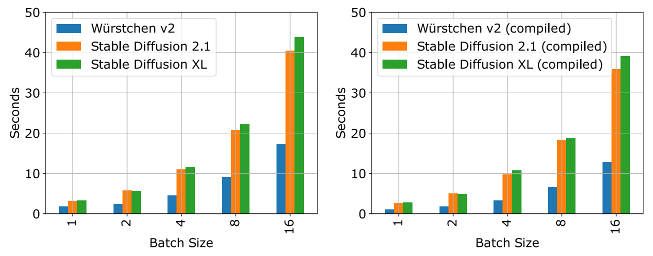


Figure 4: Inference time for 1024 × 1024 images on an A100-GPUs. Left plot shows performance without specific optimization, right plot shows performance using `torch.compile()`.

Table 1: Evaluation of Image Quality on MS-COCO and Localized Narratives (Pont-Tuset et al., 2020) using the PickScore (Kirstain et al., 2023) to binary select images generated from the same captions by two different models. **Würstchen outperforms all models of equal and smaller size, despite Stable Diffusion models using a significantly higher compute budget.**

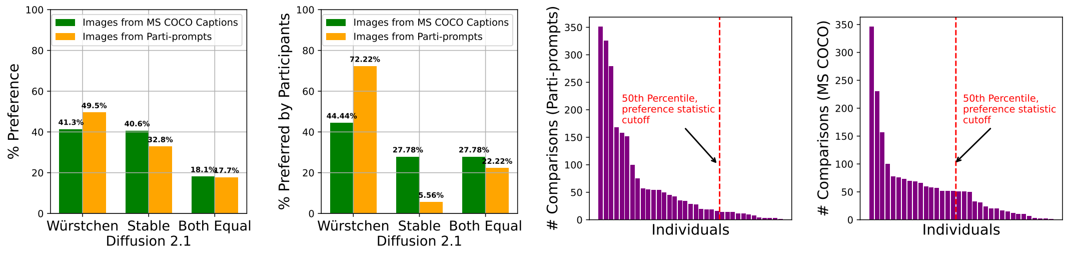
	PickScore(COCO-30k) ↑					
Model (train cost)	Baseline LDM (ours) (≈25,000 gpu-h)	DF-GAN -	GALIP -	SD 1.4 (150.000 gpu-h)	SD 2.1 (200.000 gpu-h)	SD XL -
Würstchen (24,602 gpu-h)	**96.5%**	**99.8%**	**98.1%**	**78.1%**	**64.4%**	39.4%
	PickScore (Localized Narratives-COCO-5K) ↑					
Model (train cost)	Baseline LDM (ours) (≈25,000 gpu-h)	DF-GAN -	GALIP -	SD 1.4 (150.000 gpu-h)	SD 2.1 (200.000 gpu-h)	SD XL -
Würstchen (24,602 gpu-h)	**96.6%**	**98.0%**	**95.5%**	**79.9%**	**70.0%**	39.1%
	PickScore (Parti-prompts) ↑					
Model (train cost)	Baseline LDM (ours) (≈25,000 gpu-h)	DF-GAN -	GALIP -	SD 1.4 (150.000 gpu-h)	SD 2.1 (200.000 gpu-h)	SD XL -
Würstchen (24,602 gpu-h)	**98.6%**	**99.6%**	**97.9%**	**82.1%**	**74.6%**	39.0%

images as COCO30K. Since the prompts of MS COCO are quite short and frequently lack detail, we also generate 5,000 images from the Localized Narrative MS COCO subset, we refer to his dataset as Localized Narratives-COCO-5K. Finally, we also use Parti-prompts (Yu et al., 2022b), a highly diverse set of 1633 captions, which closely reflects the usage scenario we intend for our model.

4.1 Automated Text-to-Image Evaluation

We evaluate the quality of the generated images using automated metrics in comparison to other, publicly available models (see Appendix A for random examples). The PickScores in Table 1 paint a consistent picture over the three datasets the models were evaluated on. Würstchen is preferred very significantly over smaller models like DF-GAN and GALIP, which is expected. The LDM is outperformed dramatically in all cases, highlighting that the architecture had a significant impact on the model's computational training efficiency. **Würstchen is also preferred in all three scenarios over SD 1.4 and 2.1, despite their significantly higher compute-budget at a similar model-capacity.** While SD XL is still superior in image quality, our inference speed is significantly faster (see Figure 4). This comparison is not entirely fair, as it's a higher capacity model and its data and compute budget is unknown. For this reason, we are omitting SD XL from the following experiments.

While we achieve a higher Inception Score (IC) on COCO30K compared to all other models in our broader comparison in Table 2 also shows a relatively high FID on the same dataset. While still outperforming larger models like CogView2 (Ding et al., 2022) and our Baseline LDM, the FID is substantially lower compared to other state-of-the-art models. We attribute this discrepancy to high-frequency features in the images. During visual inspections we find that images generates by Würstchen tend smoother than in other text-to-image models. This difference is most noticeable in real-world images like COCO, on which we compute the FID-metric.



(a) Overall Preference (b) Individual Preference (c) Histogram (MS COCO) (d) Histogram (Parti)

Figure 5: Overall human preferences (left) and by users (middle). The preference by users considered only users with a large number of comparisons (right).

4.2 Human Preference Evaluation

While most metrics evaluated in the previous section are correlated with human preference (Kirstain et al., 2023; Heusel et al., 2018; Salimans et al., 2016), we follow the example of other works and also conducted two brief studies on human preference. To simplify the evaluation, we solely compared Würstchen against SD 2.1, its closest capacity and performance competitor, and evaluated the human preference between these two models following the setup of Kirstain et al. (2023). In total, we conducted two studies using the generated images from Parti-prompts and COCO30K images. Participants were presented randomly chosen pairs of images in randomized order. For each pair the participants selected a preference for one or neither of the images (see Appendix C for details). In total, 3343 (Parti-prompts) and 2262 (COCO Captions) comparisons by 90 participants were made.

We evaluate results in two distinct ways. First, by counting the total number of preferences independent of user-identity. In Figure 5 (a) we can see that images generated by our model on Parti-prompts were clearly preferred. This is important to us, since Parti-prompt closely reflects the intended use case of the model. However, for MS-COCO this statistic is inconclusive. We hypothesize that this is due to the vague prompts generating a more diverse set of images, making the preference more subject to personal taste, biasing this statistics towards users that completed more comparisons (Figure 5 (c, d)). For this reason, we conducted a second analysis, where we evaluated the personally preferred model for each individual. In an effort to only include participants that completed a representative number of comparisons, we only include users from the upper 50th percentile and above. By doing so, we include only individuals with at least 30 (MS-COCO) and 51 (Parti-prompts) comparisons in the statistic. Under these circumstances, we observed a light preference for MS-COCO in favor of Würstchen and a strong preference for our model on Parti-prompts (Figure 16 (b)). In summary, the human preference experiments confirm the observation made in the PickScore experiments. While the real-world results were in-part less decisive, **the image generation quality of Würstchen was overall preferred by the participants of both studies over SD 2.1**.

Table 2: Comparison to other architectures. * computed from own evaluation. † based on official model cards (Rombach & Esser, 2022; Rombach et al., 2023).

Model	Params	Sampling Steps	FID ↓ @256^2	IS ↑ @299^2	Open Source	GPU Hours @ A100 ↓	Train ↓ Samples	Est. Carbon Em. [kg CO_2 eq.]
GLIDE (Nichol et al., 2021)	3.5B	250	12.24		–	–	–	–
Make-A-Scene (Gafni et al., 2022)	4B	1024	11.84		–	–	–	–
Parti (Yu et al., 2022a)	20B	1024	**7.23**		–	–	–	–
CogView (Ramesh et al., 2021)	4B	1024	27.1	22.4	✓	–	–	–
CogView2 (Ding et al., 2022)	6B	-	24.0	25.2	-	–	–	–
DF-GAN (Tao et al., 2022)	19M	-	19.3	18.6	✓	–	–	–
GALIP (Tao et al., 2023)	240M	-	12.5	26.3*	✓	–	–	–
DALL-E (Ramesh et al., 2021)	12B	256	17.89	17.9	–	–	–	–
LDM (Rombach et al., 2022)	1.45B	250	12.63	30.3	✓	–	–	–
Baseline LDM (ours)	0.99B	60	43.5*	20.1*	-	≈25,000		≈2,300
Würstchen (ours)	0.99B	60	23.6*	**40.9***	✓	**24,602**	**1.42B**	**2,276**
SD 1.4 (Rombach et al., 2022)	0.8B	50	16.2*	40.6*	✓	150,000 †	4.8B †	11,250 †
SD 2.1 (Rombach et al., 2022)	0.8B	50	15.1*	40.1*	✓	200,000 †	3.9B †	15,000 †
SD XL (Podell et al., 2023)	2.6B	50	> 18	–	✓	–	–	–

4.3 Efficiency

Table 2 shows the computational costs for training Würstchen compared to the original SD 1.4 and 2.1. Based on the evaluations in Section 4.1, it can be seen that the proposed setup of decoupling high-resolution image projection from the actual text-conditional generation can be leveraged even more as done in the past (Esser et al., 2021; Saharia et al., 2022; Ramesh et al., 2022), while still staying on-par or outperforming in terms of quality, fidelity and alignment. Stage C, being the most expensive stage to train from scratch, required only 24,602 GPU hours, compared to 200,000 GPU hours (Rombach et al., 2023) for SD 2.1, making it a 8x improvement. Additionally, SD 1.4 and 2.1 processed significantly more image samples. The latter metric is based on the total number of steps of all trainings and finetunings and multiplied with the respective batch sizes. Even when accounting for 11,000 GPU hours and 318M train samples used for training Stage B, Würstchen is significantly more efficient to train than the SD models. Moreover, although needing to sample with both Stage A & B to generate the VQGAN latents $\bar{\mathbf{x}}_q$, the total inference is still significantly faster than SD 2.1 and XL (see Figure 4).

5 Conclusion

In this work, we presented our text-conditional image generation model Würstchen, which employs a three stage process of decoupling text-conditional image generation from high-resolution spaces. The proposed process enables to train large-scale models efficiently, substantially reducing computational requirements, while at the same time providing high-fidelity images. Our trained model achieved comparable performance to models trained using significantly more computational resources, illustrating the viability of this approach and suggesting potential efficient scalability to even larger model parameters. We hope our work can serve as a starting point for further research into a more sustainable and computationally more efficient domain of generative AI and open up more possibilities into training, finetuning & deploying large-scale models on consumer hardware. We will provide all of our source code, including training-, and inference scripts and trained models on GitHub.

Acknowledgements

The authors wish to express their thanks to Stability AI Inc. for providing generous computational resources for our experiments and LAION gemeinnütziger e.V. for dataset access and support. This work was supported by a fellowship within the IFI program of the German Academic Exchange Service (DAAD).

Author contributions

The model architecture was designed by PP and DR. The model training was carried out by PP and DR. The baseline model was trained and implemented by MR. The evaluation was carried out by MR and MA. The manuscript was written by PP, DR, MR, CP and MA.

Reproducibility Statement

We release the entire source code of our pipeline, together with the model weights used to generate these results in our GitHub repository. We also include instructions on how to train the model and an inference notebook. As described in Appendix E, we only used dedublicated publicly available data to train the model. The methodology that was used to conduct the study on human preference can be found in Appendix C. The setup of the comparison between other open-source baselines is described in Section 4. We exclusively used open-soruce models with their official repositories and weights, when computing metrics for other models.

References

Huiwen Chang, Han Zhang, Lu Jiang, Ce Liu, and William T Freeman. MaskGIT: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern*

Recognition, pp. 11315–11325, 2022.

Huiwen Chang et al. Muse: Text-to-image generation via masked generative transformers. *arXiv:2301.00704*, 2023.

Jooyoung Choi, Jungbeom Lee, Chaehun Shin, Sungwon Kim, Hyunwoo Kim, and Sungroh Yoon. Perception prioritized training of diffusion models, 2022.

Prafulla Dhariwal and Alexander Nichol. Diffusion models beat gans on image synthesis. *Advances in Neural Information Processing Systems*, 34:8780–8794, 2021.

Ming Ding, Zhuoyi Yang, Wenyi Hong, Wendi Zheng, Chang Zhou, Da Yin, Junyang Lin, Xu Zou, Zhou Shao, Hongxia Yang, et al. Cogview: Mastering text-to-image generation via transformers. *Advances in Neural Information Processing Systems*, 34:19822–19835, 2021.

Ming Ding, Wendi Zheng, Wenyi Hong, and Jie Tang. Cogview2: Faster and better text-to-image generation via hierarchical transformers. *arXiv:2204.14217*, 2022.

Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12873–12883, 2021.

Oran Gafni, Adam Polyak, Oron Ashual, Shelly Sheynin, Devi Parikh, and Yaniv Taigman. Make-a-scene: Scene-based text-to-image generation with human priors. *arXiv:2203.13131*, 2022.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium, 2018.

Jonathan Ho and Tim Salimans. Classifier-free diffusion guidance. *arXiv:2207.12598*, 2022.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Gabriel Ilharco, Mitchell Wortsman, Nicholas Carlini, Rohan Taori, Achal Dave, Vaishaal Shankar, Hongseok Namkoong, John Miller, Hannaneh Hajishirzi, Ali Farhadi, and Ludwig Schmidt. *OpenCLIP*. Zenodo, July 2021. doi: 10.5281/zenodo.5143773. URL `https://doi.org/10.5281/zenodo.5143773`.

Yuval Kirstain, Adam Polyak, Uriel Singer, Shahbuland Matiana, Joe Penna, and Omer Levy. Pick-a-pic: An open dataset of user preferences for text-to-image generation, 2023.

Alexandre Lacoste, Alexandra Luccioni, Victor Schmidt, and Thomas Dandres. Quantifying the carbon emissions of machine learning, 2019.

Rosanne Liu, Dan Garrette, Chitwan Saharia, William Chan, Adam Roberts, Sharan Narang, Irina Blok, RJ Mical, Mohammad Norouzi, and Noah Constant. Character-aware models improve visual text rendering. *arXiv:2212.10562*, 2022a.

Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11976–11986, 2022b.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *International Conference on Learning Representations (ICLR)*, 2019. doi: 10.48550/ARXIV.1711.05101. URL `https://arxiv.org/abs/1711.05101`.

Alex Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models, 2021.

Alex Nichol, Prafulla Dhariwal, Aditya Ramesh, Pranav Shyam, Pamela Mishkin, Bob McGrew, Ilya Sutskever, and Mark Chen. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv:2112.10741*, 2021.

Dustin Podell, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. Sdxl: Improving latent diffusion models for high-resolution image synthesis, 2023.

Jordi Pont-Tuset, Jasper Uijlings, Soravit Changpinyo, Radu Soricut, and Vittorio Ferrari. Connecting vision and language with localized narratives. In *ECCV*, 2020.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pp. 8748–8763. PMLR, 2021.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67, 2020.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pp. 8821–8831. PMLR, 2021.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. Hierarchical text-conditional image generation with CLIP latents. *arXiv:2204.06125*, 2022.

Dominic Rampas, Pablo Pernias, and Marc Aubreville. A novel sampling scheme for text- and image-conditional image synthesis in quantized latent spaces. *arXiv:2211.07292*, 2023.

Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. In *International conference on machine learning*, pp. 1060–1069. PMLR, 2016.

Mats L. Richter and Christopher Pal. Receptive field refinement for convolutional neural networks reliably improves predictive performance, 2022.

Mats L. Richter, Wolf Byttner, Ulf Krumnack, Anna Wiedenroth, Ludwig Schallner, and Justin Shenk. (input) size matters for cnn classifiers. In Igor Farkaš, Paolo Masulli, Sebastian Otte, and Stefan Wermter (eds.), *Artificial Neural Networks and Machine Learning – ICANN 2021*, pp. 133–144, Cham, 2021a. Springer International Publishing. ISBN 978-3-030-86340-1.

Mats L. Richter, Julius Schöning, and Ulf Krumnack. Should you go deeper? optimizing convolutional neural network architectures without training. *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 964–971, 2021b. URL `https://api.semanticscholar.org/CorpusID:235606454`.

Mats L. Richter, Julius Schöning, Anna Wiedenroth, and Ulf Krumnack. *Receptive Field Analysis for Optimizing Convolutional Neural Network Architectures Without Training*, pp. 235–261. Springer Nature Singapore, Singapore, 2023. ISBN 978-981-19-6153-3. doi: 10.1007/978-981-19-6153-3_10. URL `https://doi.org/10.1007/978-981-19-6153-3_10`.

Robin Rombach and Patrick Esser. Stable diffusion 1.4 model card, 2022. URL `https://huggingface.co/CompVis/stable-diffusion-v-1-4-original`.

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*, pp. 10684–10695, 2022.

Robin Rombach, Patrick Esser, and David Ha. Stable diffusion 2.1 model card, 2023. URL `https://huggingface.co/stabilityai/stable-diffusion-2-1`.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily Denton, Seyed Kamyar Seyed Ghasemipour, Burcu Karagol Ayan, S Sara Mahdavi, Rapha Gontijo Lopes, et al. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv:2205.11487*, 2022.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. Improved techniques for training gans. *Advances in neural information processing systems*, 29, 2016.

Christoph Schuhmann, Romain Beaumont, Richard Vencu, Cade Gordon, Ross Wightman, Mehdi Cherti, Theo Coombes, Aarush Katta, Clayton Mullis, Mitchell Wortsman, et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *arXiv:2210.08402*, 2022.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pp. 2256–2265. PMLR, 2015.

Mingxing Tan and Quoc Le. Efficientnetv2: Smaller models and faster training. In *International Conference on Machine Learning*, pp. 10096–10106. PMLR, 2021.

Mingxing Tan and Quoc V. Le. Efficientnet: Rethinking model scaling for convolutional neural networks, 2020.

Ming Tao, Hao Tang, Fei Wu, Xiao-Yuan Jing, Bing-Kun Bao, and Changsheng Xu. Df-gan: A simple and effective baseline for text-to-image synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 16515–16525, 2022.

Ming Tao, Bing-Kun Bao, Hao Tang, and Changsheng Xu. Galip: Generative adversarial clips for text-to-image synthesis. *arXiv preprint arXiv:2301.12959*, 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. Byt5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306, 2022.

Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, Ben Hutchinson, Wei Han, Zarana Parekh, Xin Li, Han Zhang, Jason Baldridge, and Yonghui Wu. Scaling autoregressive models for content-rich text-to-image generation. *arXiv:2206.10789*, 2022a. doi: 10.48550/ARXIV.2206.10789. URL `https://arxiv.org/abs/2206.10789`.

Jiahui Yu et al. Scaling autoregressive models for content-rich text-to-image generation. *arXiv:2206.10789*, 2022b.

Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris N Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 5907–5915, 2017.

A COLLAGES (RANDOMLY CHOSEN FROM PARTI-PROMPTS)

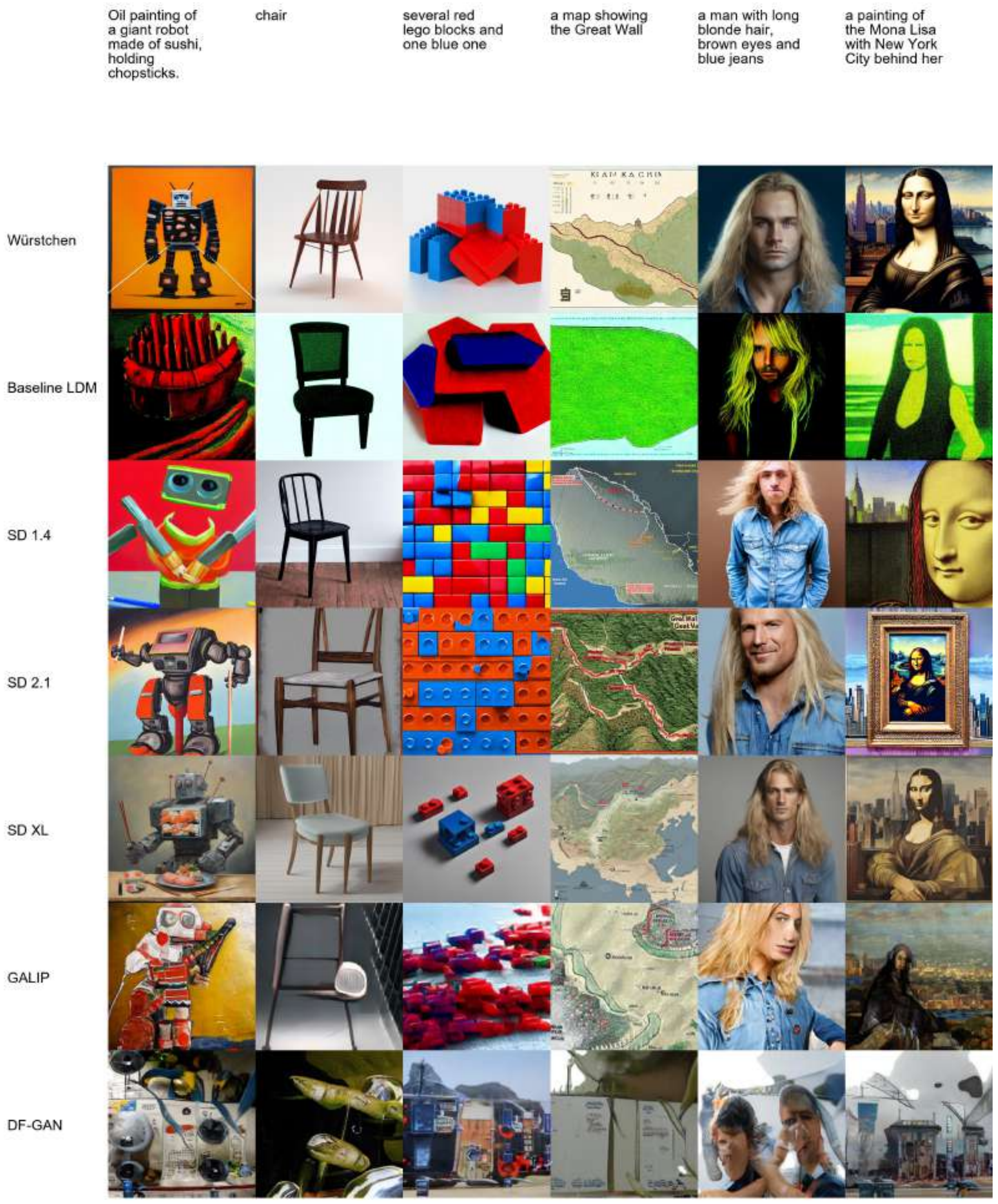

Oil painting of a giant robot made of sushi, holding chopsticks.
chair
several red lego blocks and one blue one
a map showing the Great Wall
a man with long blonde hair, brown eyes and blue jeans
a painting of the Mona Lisa with New York City behind her
Würstchen
Baseline LDM
SD 1.4
SD 2.1
SD XL
GALIP
DF-GAN

Figure 6: Collage # 1

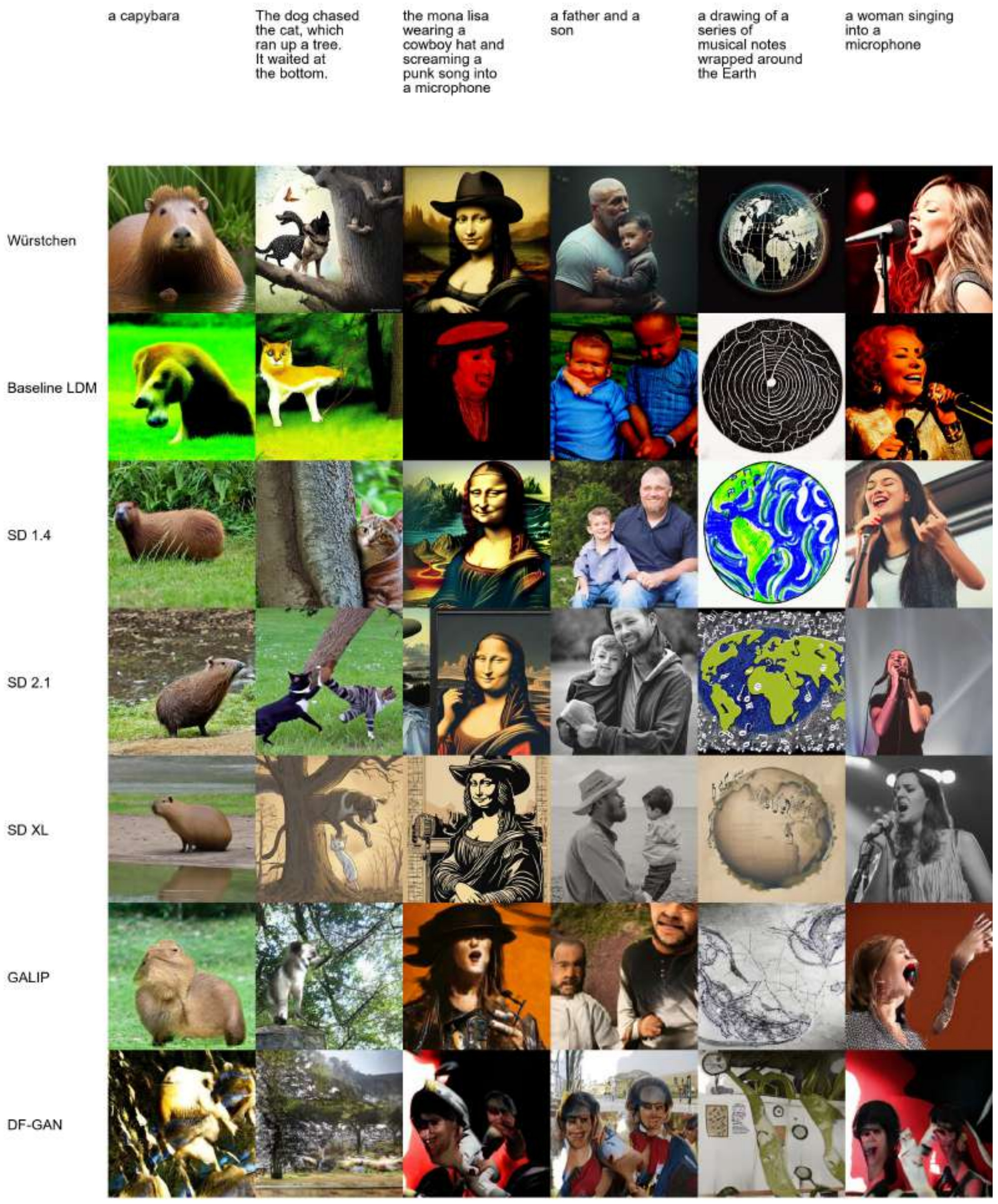

a capybara
The dog chased the cat, which ran up a tree. It waited at the bottom.
the mona lisa wearing a cowboy hat and screaming a punk song into a microphone
a father and a son
a drawing of a series of musical notes wrapped around the Earth
a woman singing into a microphone
Würstchen
Baseline LDM
SD 1.4
SD 2.1
SD XL
GALIP
DF-GAN

Figure 7: Collage # 2

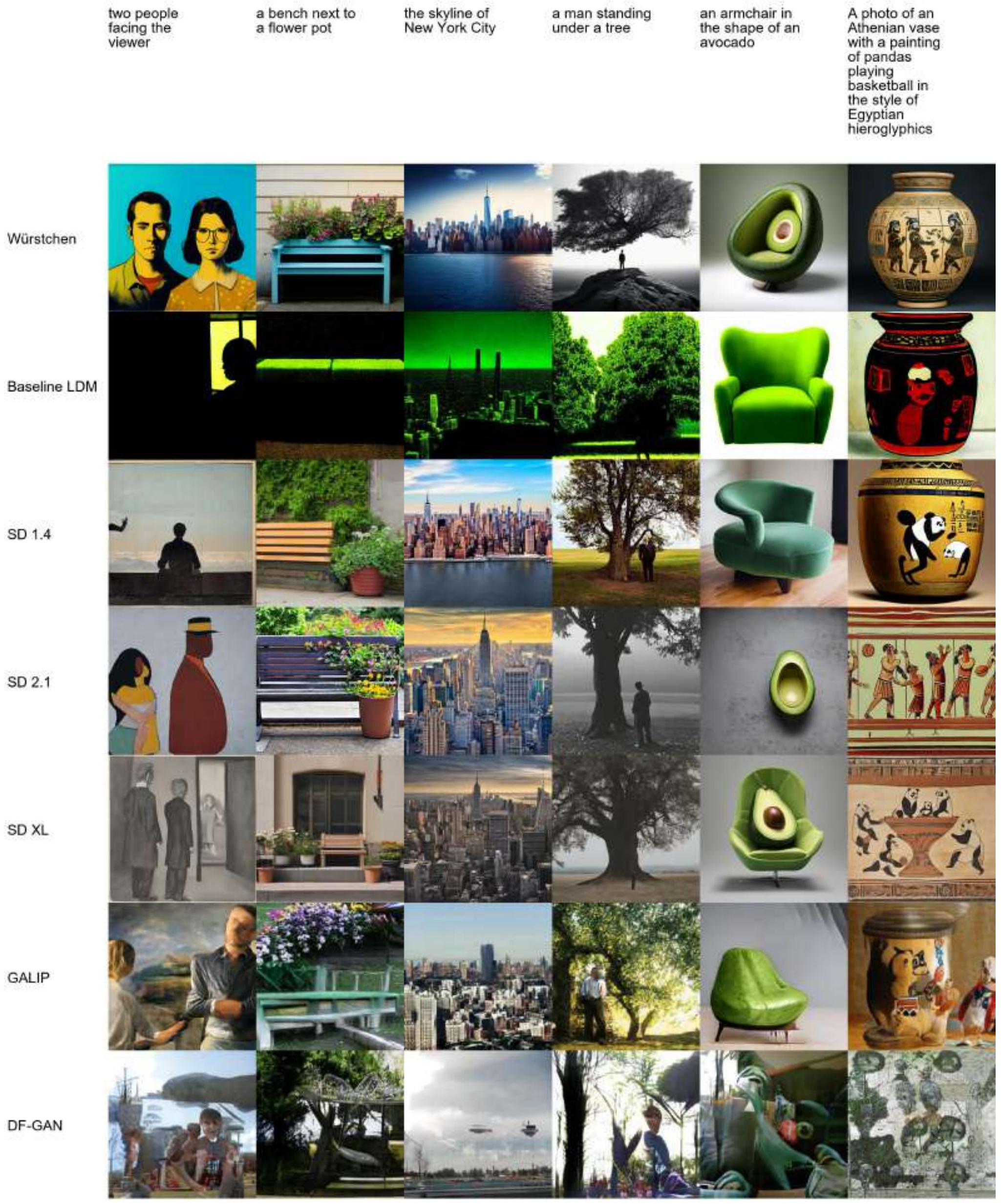

two people facing the viewer
a bench next to a flower pot
the skyline of New York City
a man standing under a tree
an armchair in the shape of an avocado
A photo of an Athenian vase with a painting of pandas playing basketball in the style of Egyptian hieroglyphics
Würstchen
Baseline LDM
SD 1.4
SD 2.1
SD XL
GALIP
DF-GAN

Figure 8: Collage # 3

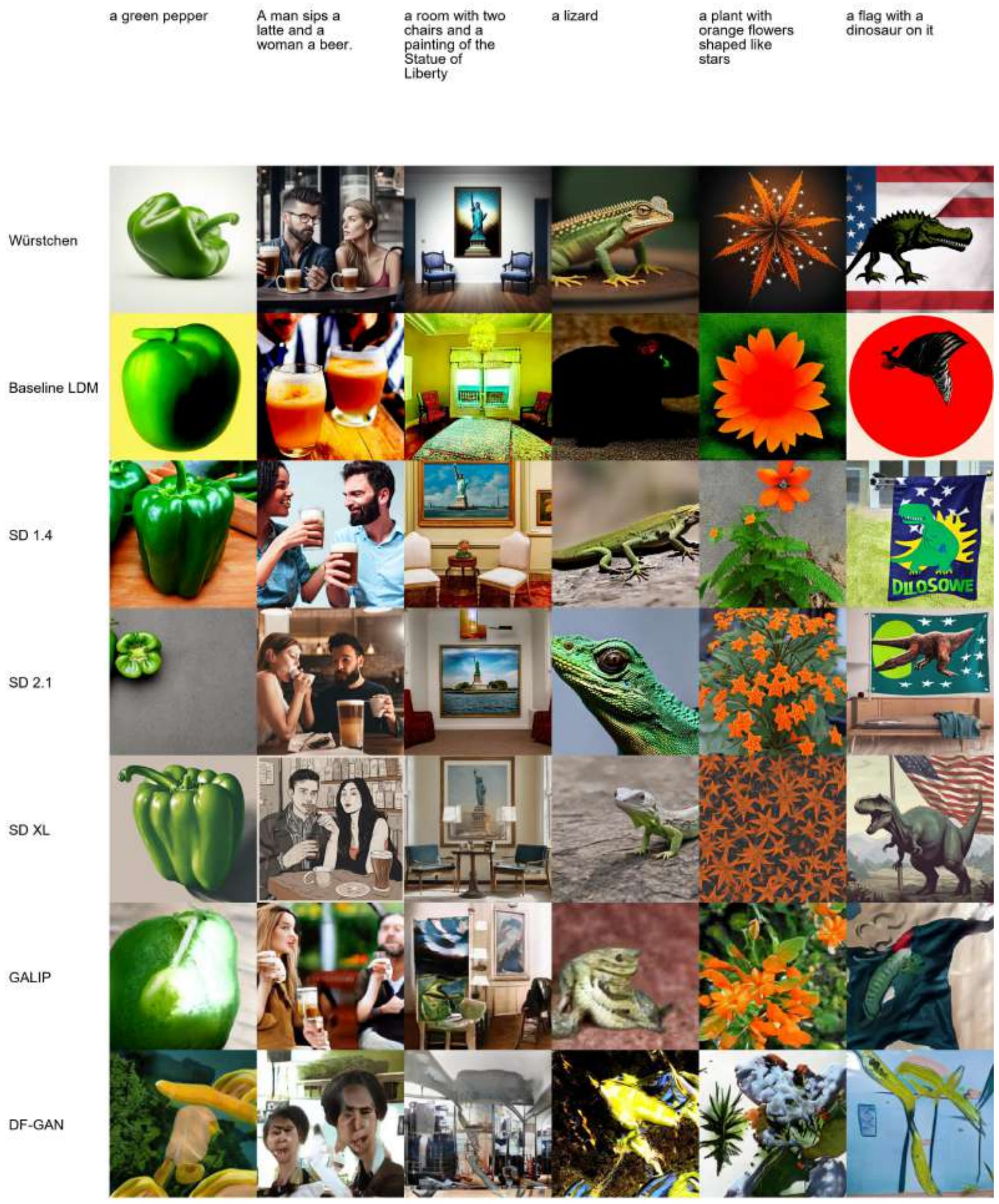
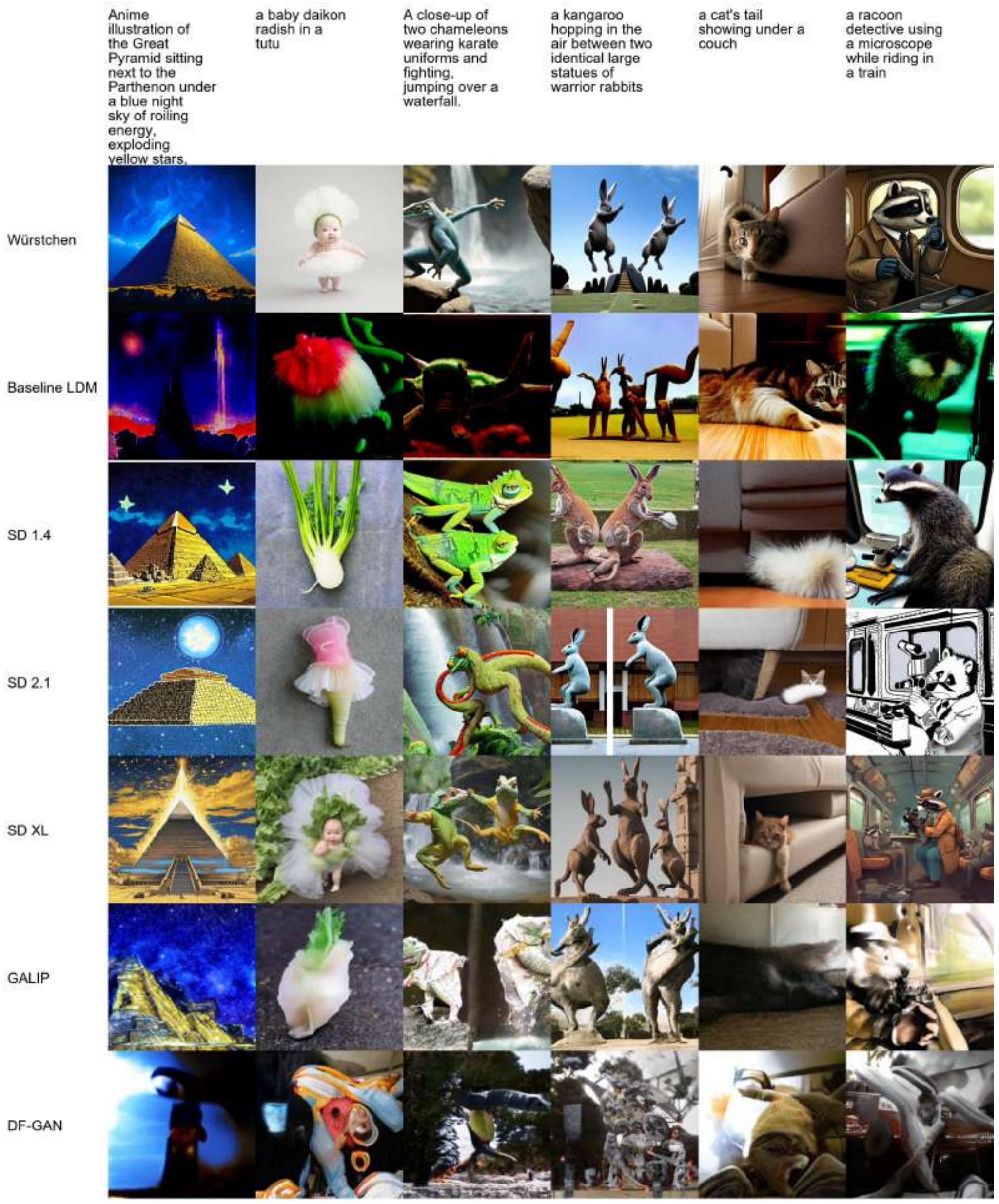

a green pepper
A man sips a latte and a woman a beer.
a room with two chairs and a painting of the Statue of Liberty
a lizard
a plant with orange flowers shaped like stars
a flag with a dinosaur on it
Würstchen
Baseline LDM
SD 1.4
SD 2.1
SD XL
GALIP
DF-GAN
DILOSOWE

Figure 9: Collage # 4



Anime illustration of the Great Pyramid sitting next to the Parthenon under a blue night sky of roiling energy, exploding yellow stars.
a baby daikon radish in a tutu
A close-up of two chameleons wearing karate uniforms and fighting, jumping over a waterfall.
a kangaroo hopping in the air between two identical large statues of warrior rabbits
a cat's tail showing under a couch
a racoon detective using a microscope while riding in a train
Würstchen
Baseline LDM
SD 1.4
SD 2.1
SD XL
GALIP
DF-GAN

Figure 10: Collage # 5

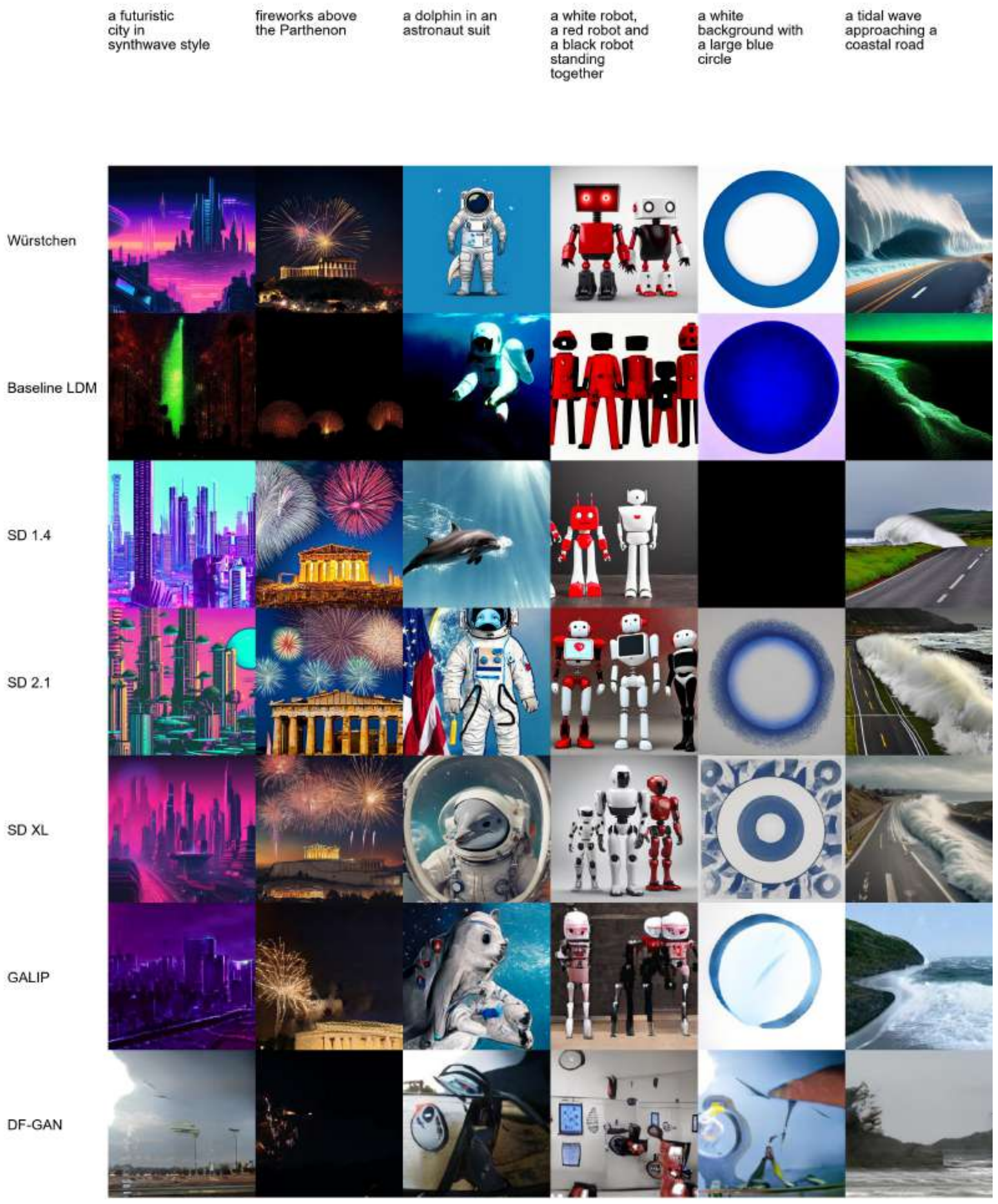

a futuristic city in synthwave style
fireworks above the Parthenon
a dolphin in an astronaut suit
a white robot, a red robot and a black robot standing together
a white background with a large blue circle
a tidal wave approaching a coastal road
Würstchen
Baseline LDM
SD 1.4
SD 2.1
SD XL
GALIP
DF-GAN

Figure 11: Collage # 6

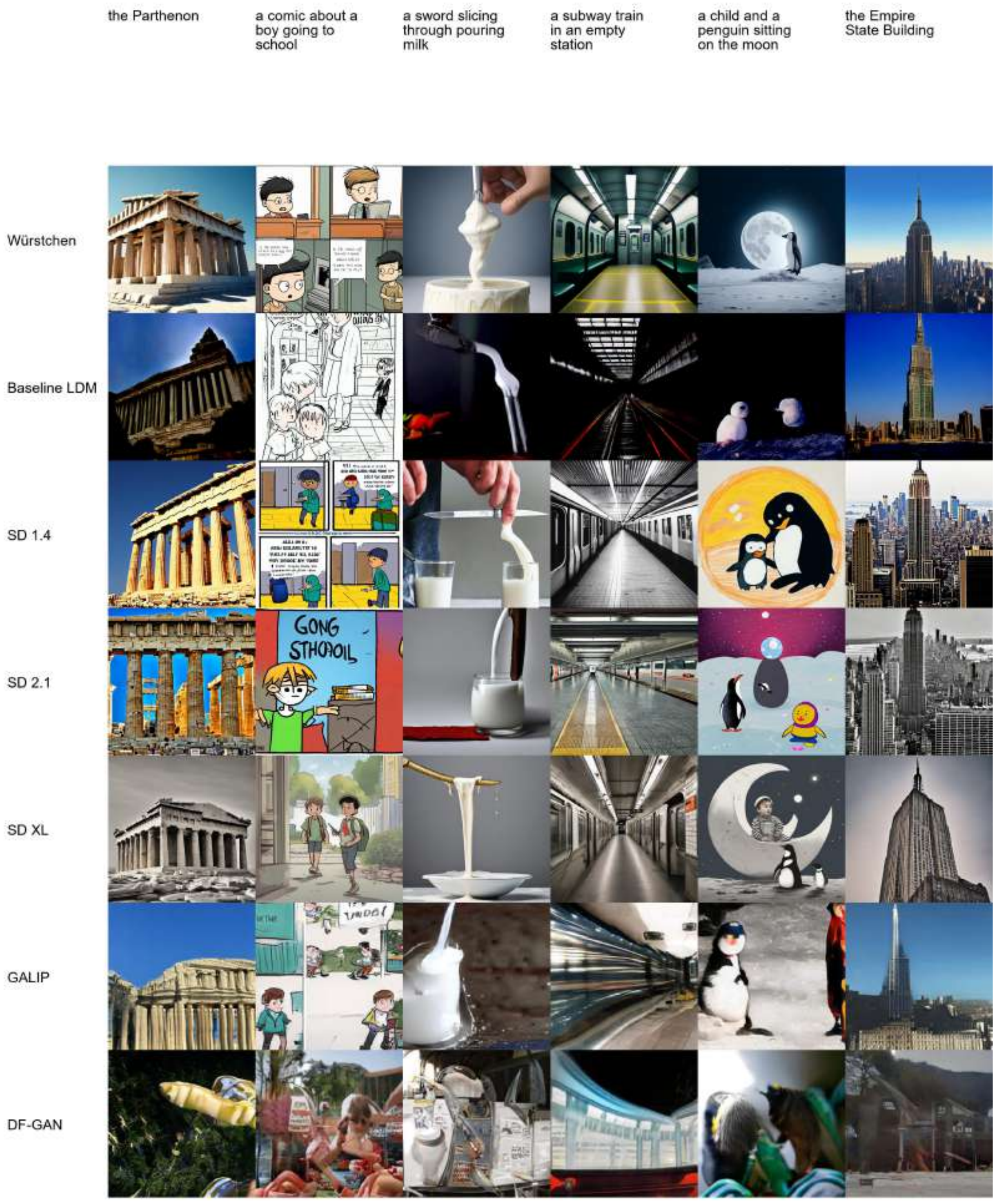

the Parthenon
a comic about a boy going to school
a sword slicing through pouring milk
a subway train in an empty station
a child and a penguin sitting on the moon
the Empire State Building
Würstchen
Baseline LDM
SD 1.4
SD 2.1
SD XL
GALIP
DF-GAN

Figure 12: Collage # 7

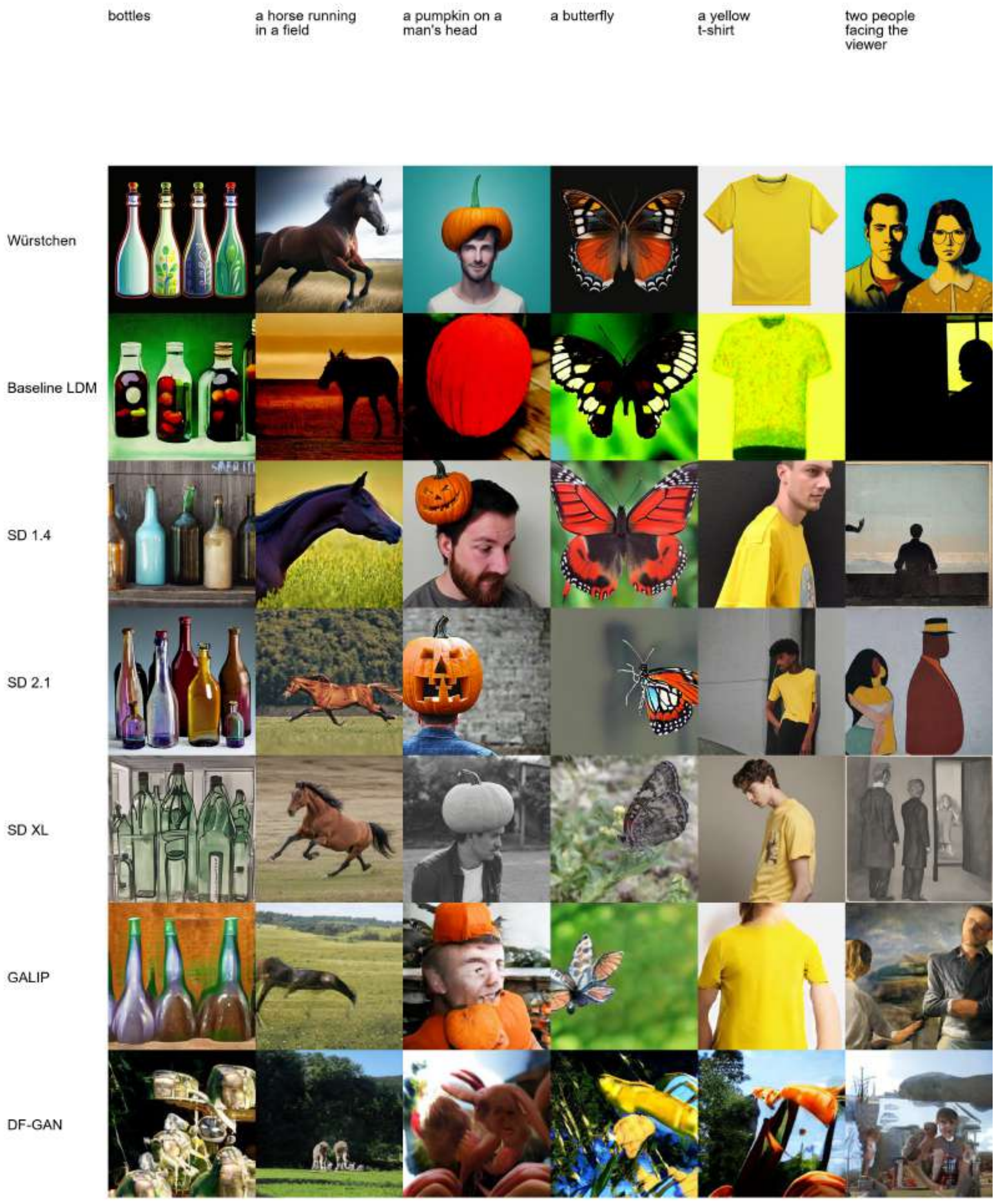


Figure 13: Collage # 8

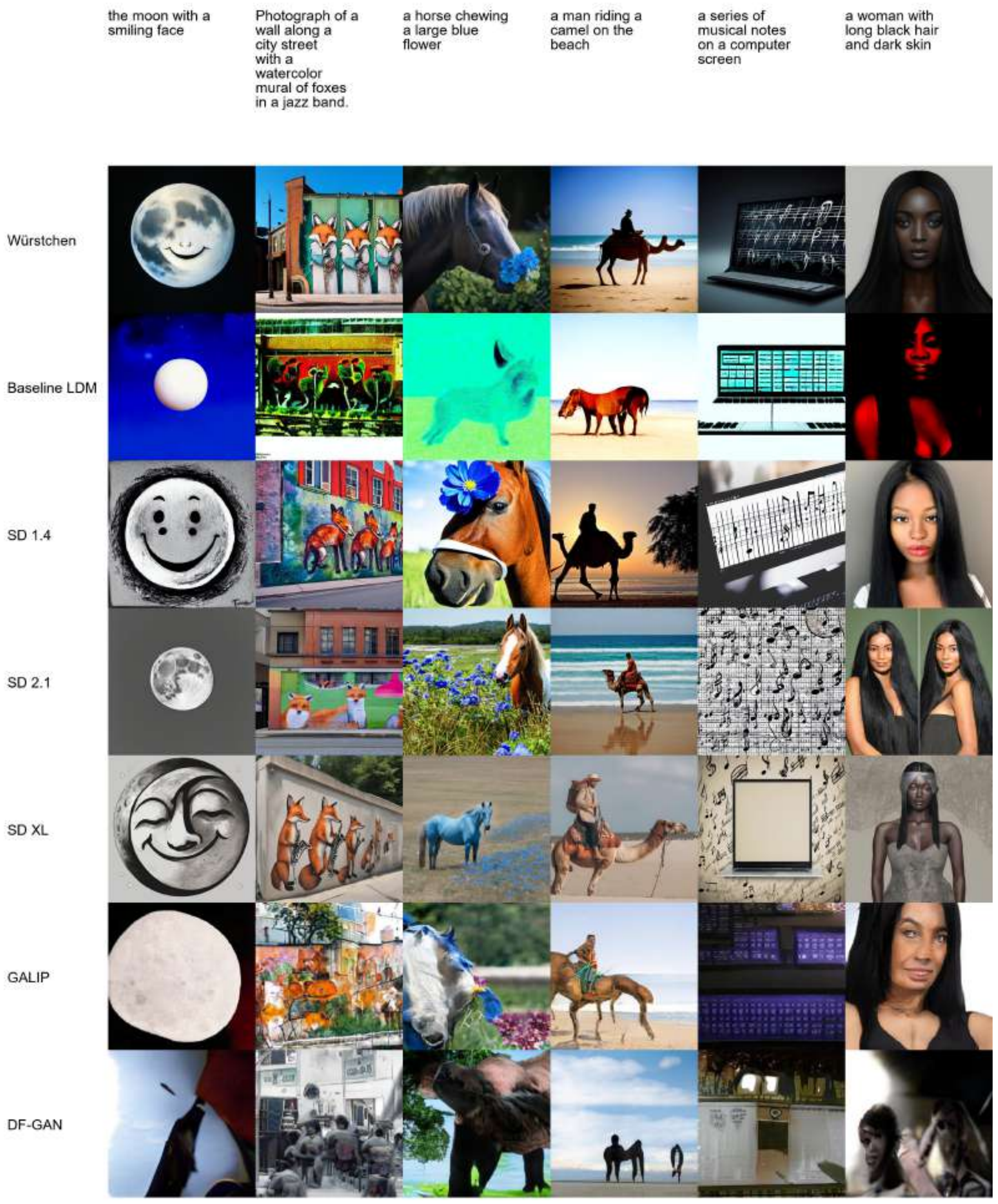
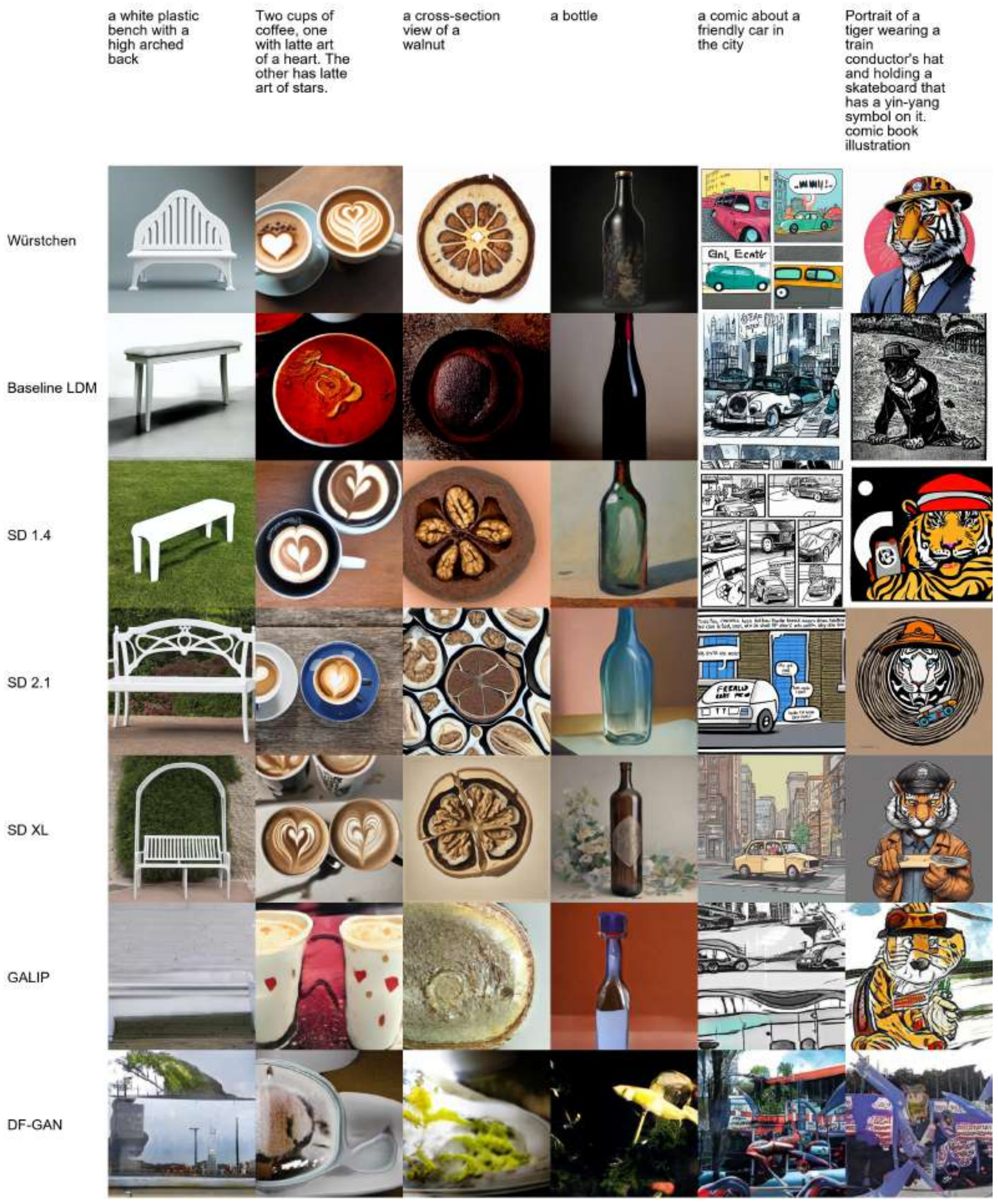

the moon with a smiling face
Photograph of a wall along a city street with a watercolor mural of foxes in a jazz band.
a horse chewing a large blue flower
a man riding a camel on the beach
a series of musical notes on a computer screen
a woman with long black hair and dark skin
Würstchen
Baseline LDM
SD 1.4
SD 2.1
SD XL
GALIP
DF-GAN

Figure 14: Collage # 9



a white plastic bench with a high arched back
Two cups of coffee, one with latte art of a heart. The other has latte art of stars.
a cross-section view of a walnut
a bottle
a comic about a friendly car in the city
Portrait of a tiger wearing a train conductor's hat and holding a skateboard that has a yin-yang symbol on it. comic book illustration
Würstchen
Baseline LDM
SD 1.4
SD 2.1
SD XL
GALIP
DF-GAN

Figure 15: Collage # 10

B Robustness Assessment of Fréchet Inception Distance

The Fréchet Inception Distance (FID) is commonly used to evaluate the fidelity of text-conditional image generators. For this, a large quantity (e.g., 30k) of prompts are retrieved from a dataset that was not used for the training of the system under test. The original images corresponding to those prompts as well as the generated images are then fed to an independent model (typically Inception V3) that was trained on another independent dataset (typically ImageNet). As this model was typically trained on a reduced input resolution (e.g., 299x299 in the case of Inception v3), resampling of the input images is necessary. The FID is calculated from the first and second order statistics of the features found as output of the feature extractor of that model.

Usage of a metric, however, also implies knowledge of eventual misbehaviors of said metric, hence we compare in this section the reaction of the FID towards manipulations that we assumed could be assumed not to be linked towards strong changes in image fidelity. In particular, we stored the images using various quality settings for JPEG compression, used different methods for resampling of the images and performed mild changes in image color, brightness and contrast.

As shown in Table 3, while changes that are part of standard image augmentation like mild changes in brightness and contrast do not impact the FID significantly, a moderate JPEG compression as well as just a change of resampling do impact the metric strongly. In particular, a JPEG compression with 70% quality, which impedes the image quality only to a small degree, yields an FID score in the range of our results, without involving any image generation.

Manipulation	Configuration	FID @ COCO 30k	FID @ CelebA-HQ
JPEG compression	quality=95%	0.268	0.560
	quality=90%	1.713	2.381
	quality=80%	6.658	6.291
	quality=70%	10.469	9.156
	quality=60%	13.274	11.617
	quality=50%	15.129	13.519
Resampling	NN interpolation	5.239	3.705
	bilinear interpolation	0.330	0.569
Color change	8-bit color palette	27.989	31.289
	brightness +10%	0.085	0.112
	brightness -10%	0.054	0.101
	contrast +10%	0.051	0.077
	contrast -10%	0.072	0.098

Table 3: Assessment of Fréchet Inception Distance (FID) following minor image manipulations that are not expected to significantly alter the fidelity and composition of the images.

C METHDOLOGY OF THE HUMAN PREFERENCE EXPERIMENTS

Used Models: The studies were conducted with images generated with SD 2.1 and Würstchen.

Data Displayed to User: We generated 30,000 images based on the COCO-validation set prompts for each model for the first study and 1,633 images each based on Partiprompts for the second study. All images were scanned manually for harmful and graphic and pornographic content.

Setup: Both studies are conducted online. Participants are presented an image generated from both models using the same prompt. The prompt is also displayed. Neither the model that generated the images nor the number of models used for the image generation as a whole is known and never displayed to the participants. The displayed images are randomly chosen every time and displayed in a random order.

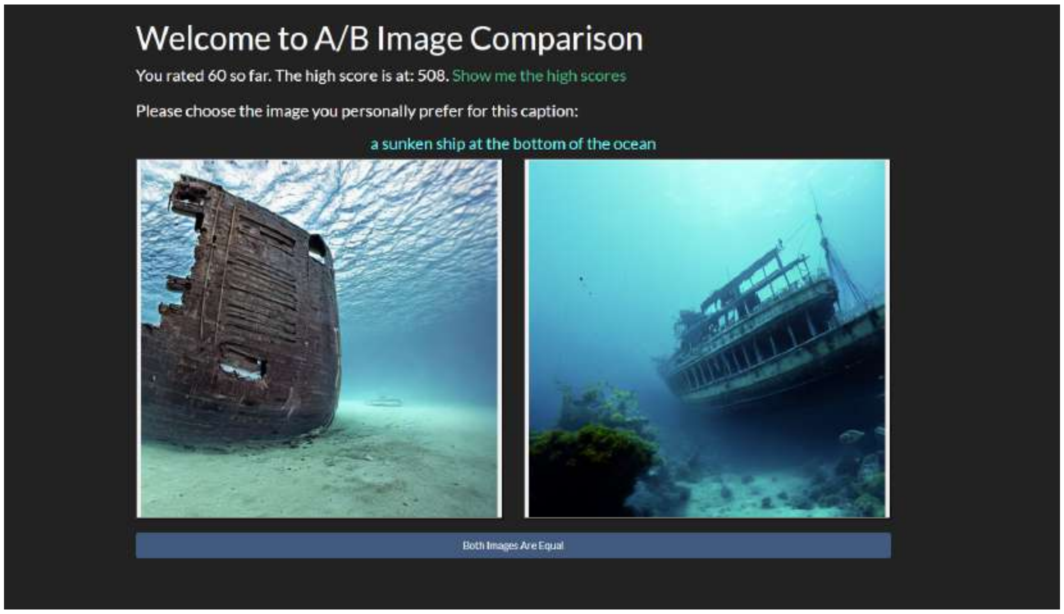

Welcome to A/B Image Comparison
You rated 60 so far. The high score is at: 508. Show me the high scores
Please choose the image you personally prefer for this caption:
a sunken ship at the bottom of the ocean
Both Images Are Equal

Figure 16: A screenshot from the human preference study. Users can click on either image or the button below to select a preference.

Participants can answer that they prefer the left, right image or perceive both as equal. participants are not paid and are only given the instruction to select images based on the prompt and their personal taste. The second study, which was conducted on Parti-prompts, also urged the participants to annotate 50 pairs to avoid a long-tailed turnout, which happened in the first study. For gamification reasons, a leaderboard was added which allows users which achieved a high number of votes to enter a personal alias.

Participation: Based on the randomly generated pseudonyms, in total of 90 unique users participated in both studies combined, 33 of which participated exclusively in the first study using images generated from COCO-prompts, 58 participated exclusively in the second study, which used images generated from Parti-prompts, 3 users participated in both studies.

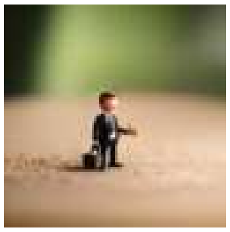
Over both datasets, a total of 4026 comparisons were evaluated. 2490 comparison were done on MS-COCO generated images and 1604 on images generated from Parti-prompt captions.

Participants received no compensation of any form. Participation was pseudonymous.

D How are Stage B and C sharing their workload?

In our work we view Stage C as the primary working part of the model, when it cames to generating images from text. However, this is not immediately clear from the architecture as Stage B and C are both generative models and have similar capacities. In this section, we are going to briefly explore how Stage B and Stage C share the workload of image generation. By doing so, we demonstrate that Stage C is responsible for the content of the image, while Stage B acts as a refining model, adding details and increasing the resolution, but ultimately not changing the image in a semantically meaningful way. To investigate, we trained a small (3.9M parameter) decoder to reconstruct the images from the latents produced by Stage C and compared the reconstructions with reconstructions from Stage B conditioned on Stage C. The results in Figures 17, 18, 19 and 20 show that the images generated by Stage C are very similar to the images generated from Stage B and C combined. From the visual inspection we can observe that the main difference are minor details as well as a reduction in blurriness. These changes can be viewed as the main contributions of Stage B. From this we conclude that Stage C is the primary factor when it comes to transforming text into images. This is further supported by the fact that short experiments conducted on alternative training regimes suggest that the text conditioning on Stage B does not enhance the quality of the images and could be dropped in future generations our model.

The Decoder Architecture: This decoder is very simple, to mitigate the influence on the latents as little as possible, consisting of 4 stages, composed of 2 convolutional layers. the first downsampling layer is 2×2 convolution with stride size 2. The second convolution is a 3×3 convolution with stride size 1 and GELU-activation function and batch norm. The first stage has 512 channels, consecutive stages half the channel width. A final 1×1 convolution squeezes the channels to 3 color channels.


(a) Stage C

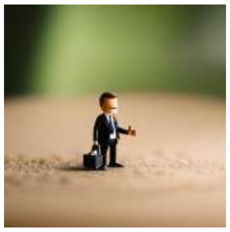
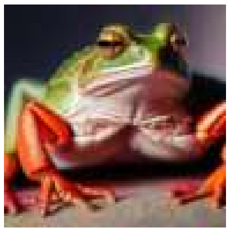

(b) Stage B and C

Figure 17: Caption: Macro photography of a tiny businessman.


(a) Stage C

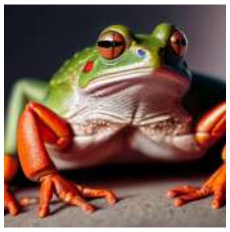
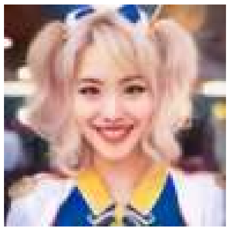

(b) Stage B and C

Figure 18: Caption: Dramatic photography of a frog evolving into a crab, crab legs, macro photography.


(a) Stage C

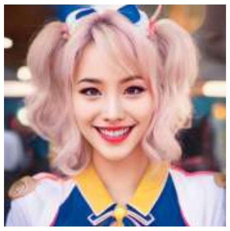
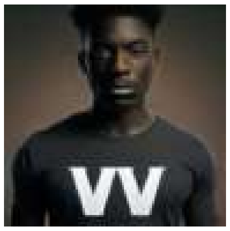

(b) Stage B and C

Figure 19: Caption: Cute woman smiling wearing a Sailor Moon cosplay.


(a) Stage C

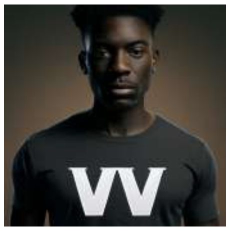
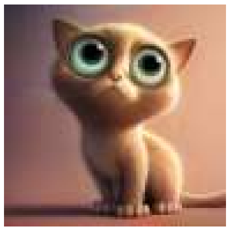

(b) Stage B and C

Figure 20: Caption: a black man with a t-shirt with the letter W


(a) Stage C

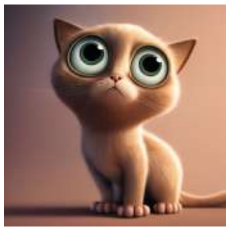

(b) Stage B and C

Figure 21: Caption: Cute cat, big eyes pixar style.

E A Detailed Look at the Neural Architectures

E.1 Stage A

Neural Architecture: The VQGAN is composed of an encoder and decoder with 2 stages each, separated with a downsampling or upsampling layer with a 4×4-kernel and stride size 2. The encoder starts, and the decoder ends with a pixel shuffle operation with a scale factor of 2. In the encoder, each stage consists of a single ConvNeXt block (Liu et al., 2022b) with 384 input channels and 1536 embedding channels. the final layer of the encoders is a 1×1 convolution followed by a BatchNorm-layer, reducing the number of channels to the dimensionality of the encoding layer (4). The decoder reverses this operation with a similar combination of layers. The decoding layer uses 16 blocks in the first stage and 1 block in the second stage. All blocks have 384 input channels and 1536 embedding channels.

Training Details: The model is trained in a single run with 500,000 iterations using the AdamW optimizer with a learning rate of $1e-4$ and a batch size of 256. The model is fed with 128×128 pixel crops from images taken from the deduplicated subsets of the improved-aesthetic LAION-5B (Schuhmann et al., 2022) dataset, which were previously resized to 256×256 images. As we anticipate the removal of the quantization, we randomly drop the quantization during training with a drop-chance 10%. For training we use three distinct losses. Mean Squared Error (MSE), Adverserial Loss (AL) and Perceptual Loss (PL). for the first 10.000 iterations we use the loss weights of (1.0, 0.0, 0.1) ofr MSE, AL and PL respectively. We activate the AL after 10,000 training steps by increasing the weight to 0.01.

E.2 Stage B

Neural Architecture: Stage B is a U-Net architecture with 4 stages in the encoder and decoder on the latents of the (unquantized) latent of the Stage A VQGAN. The stages have a channel width of 320, 640, 1280 and 1280 respectively. Each stage starts with a single convolutional layer with a 2×2 kernel and a stride size of 2 acting as downsampling layer. The stages consist of a number of building blocks. Except for the first stage, a building block consists of a ConvNeXt block (Liu et al., 2022b), a time-step block, applying a linear conditioning (Rombach et al., 2022) on the latents and a cross-attention block for conditioning on text-embeddings and image-embeddings. The first stage omits the cross-attention for conditioning on text and images. GlobalResponseNorm and GELU activation functions are used for normalization and as activation functions respectively. The cross-attention mechanisms of each stage have a different number of attention-heads: -, 10, 20, 20 (as the first stage is only conditioned on time). For all stages with cross-attention, the latents of Semantic Compressor are also concatenated to each residual block before the channelwise-convolution is applied. To fit the respective feature-map size, bicubic interpolation is used to resize the latent of the Semantic Compressor. The encoder and decoder stages have 4, 4, 14 and 4 blocks respectively. The clip-embeddings have a dimensionality of 1024. The Semantic Compressor produces latents with 16 channels.

The Semantic Compressor is composed of an EfficientNetV2 S (Tan & Le, 2021) backbone. The final global-pooling and classification-head is replaced by a 1×1-convolution with stride size 1, compressing the channels down to 16. It is worth noting that EfficientNetV2 S is dropped after the training of Stage B and C is complete, as it is replaced by Stage C during inference time.

Training Details: The model is trained using the AdamW optimizer (Loshchilov & Hutter, 2019) with a learning rate of $1e^{-4}$ using a linear warm-up schedule for 10k steps. In total, the model is trained for 457,000 iterations with an input resolution of 512×512 and a batch size of 512. The model is trained for an additional 300,000 iterations with an input resolution of 1024×1024 and a batch size of 128. The resolutions described in this work are the image sizes fed into the VQGAN for encoding. The Semantic Compressor is fed an input resolution of 384×384 pixels and 768×768 during these two phases of training respectively. The training time and compute budget listed in the paper reflects this entire training process. All images fed into the semantic compressor are normalized using $\mu = (0.485, 0.456, 0.406)$ and $\sigma = (0.229, 0.224, 0.225)$. The model is trained on a deduplicated subsets of the improved-aesthetic LAION-5B (Schuhmann et al., 2022) dataset.

E.3 Stage C

Neural Architecture: Stage C consists of a sequence of 16 building blocks. Each building block is composed of ConvNeXt block (Liu et al., 2022b) a time-conditioning block and a cross-attention block for text-conditioning, similar to Stage B. Text-Conditioning is applied from an unpooled CLIP-H model. Text-embedding has a dimensionality of 1024, each cross-attention block has 16 heads. The width of the network is 1280 channels.

The Semantic Compressor is composed of an EfficientNetV2 S (Tan & Le, 2021) backbone trained during Stage B training and otherwise unchanged. It is worth noting that EfficientNetV2 S is dropped after the training of Stage B and C is complete. During inference time, the output of Stage C instead used to condition Stage B, replacing the Semantic Compressor entirely. The Semantic Compressor is not trained during Stage B training.

Training Details: The model is trained a total of 4 consecutive times using the AdamW optimizer with a learning rate of $1e-4$. The first three consecutive trainings are conducted on a deduplicated subsets of the improved-aesthetic LAION-5B (Schuhmann et al., 2022) dataset. The final training is conducted on the dataset but further filtered by aesthetical artworks. The images are fed into the frozen Semantic Compressor to produce latents of a specific resolution. We provide the resolution of the latents alongside the resolution fed into the Semantic Compressor. The preprocessing done on images of the compressor is identical to Stage B.

The first training is conducted for 500,000 iterations on 12×12 latents, which corresponds to an 384×384 input resolution for the semantic compressor using a batch size of 1536. The second training is conducted for an additional 364,000 iterations on 24×24 latents, corresponding to 768×768 images being fed into the Semantic Compressor, using a batch size of 1536. The third training is run for only 4,000 steps and is done to adapt the model to various aspect ratios. The aspect ratio is randomized uniformly for each batch of 768 images to one of the three following values: 768×1280, 1280×768 and 768×768. The fourth and final training is designed to improve the aesthetical quality of images and is conducted for another 50,000 iterations using a batch size of 384 and a resolution of 768×768 (24×24 latents).

The final model is a 50:50 interpolation between the weights after the 3rd training and the final training run. This allows the model to generate a blend of aesthetic/artistic and realistic images. However, we open source the two models this interpolation is based on besides this final model.

E.4 Baseline LDM

Neural Architecture: The LDM is a U-Net architecture with 4 stages in the encoder and decoder on the latents of the VAE used by Stable-Diffusion 1.4. The stages have a channel width of 320, 640, 1280 and 1280 respectively. Each stage starts with a single convolutional layer with a 2×2 kernel and a stride size of 2 acting as downsampling layer. The stages consist of a number of building blocks. A building block consists of a ConvNeXt block (He et al., 2016) and two cross attention blocks for conditioning on time and text-embeddings, using GlobalResponseNorm and GELU activation functions. The first cross-attention in a building block conditions on the time step t, while the second one on the text embeddings c_{text}. The cross-attention mechanisms of each stage have a different number of attention-heads: 5, 10, 20, 20. The encoder stages have 2, 4, 14 and 4 blocks respectively, while the corresponding decoder stages have 5, 15, 5 and 3 blocks. Like for the other models, the clip-embeddings have a dimensionality of 768. Dropout is applied with a probability of 10% on a features of the text and image embeddings as well as the 3×3-convolution in the ConvNeXt-block.

Training Setup: The model is trained using the AdamW optimizer (Loshchilov & Hutter, 2019) with a learning rate of $1e^{-4}$ using a linear warm-up schedule for 10k steps and a batch size of 1280. The training is conducted for approximatly 25,000 GPU hours, which roughly corresponds to 1.5 million training steps.

The model is trained on subsets of the improved-aesthetic LAION-5B (Schuhmann et al., 2022) dataset. We use a dropout of 5% on the CLIP-H-text embeddings.