Text to image generation:

VQGAN tokenization + text encoder+base transformer+ highres transformer-> discrete token -> masked language modelling+ textembedding -> lowres -> highres -> generation via decoder -> non autoregressive iterative decoding in very less number of step using cosine schedule masking ratio, classifier free guidance with unconditional text logit and conditional text logit based on guidance t which is linearly increasing

Muse: Text-To-Image Generation via Masked Generative Transformers

Huiwen Chang * **Han Zhang** * **Jarred Barber** † **AJ Maschinot** † **José Lezama** **Lu Jiang** **Ming-Hsuan Yang**
Kevin Murphy **William T. Freeman** **Michael Rubinstein** † **Yuanzhen Li** † **Dilip Krishnan** †

Google Research

CLIP Score is a text-to-image similarity metric. It measures the similarity between generated image embedding and the text embedding. If they are close to each other in the same latent/embedding space, then CLIP score would be high for that generated image.(ranges from -1 to +1 bcz its just cosine similarity). It basically measure an image ' s proximity to a text

Like MaskGIT

Abstract

We present Muse, a text-to-image Transformer model that achieves state-of-the-art image generation performance while being significantly more efficient than diffusion or autoregressive models. Muse is trained on a masked modeling task in discrete token space: given the text embedding extracted from a pre-trained large language model (LLM), Muse is trained to predict randomly masked image tokens. Compared to pixel-space diffusion models, such as Imagen and DALL-E 2, Muse is significantly more efficient due to the use of discrete tokens and requiring fewer sampling iterations; compared to autoregressive models, such as Parti, Muse is more efficient due to the use of parallel decoding. The use of a pre-trained LLM enables fine-grained language understanding, translating to high-fidelity image generation and the understanding of visual concepts such as objects, their spatial relationships, pose, cardinality etc. Our 900M parameter model achieves a new SOTA on CC3M, with an FID score of 6.06. The Muse 3B parameter model achieves an FID of 7.88 on zero-shot COCO evaluation, along with a CLIP score of 0.32. Muse also directly enables a number of image editing applications without the need to fine-tune or invert the model: inpainting, outpainting, and mask-free editing. More results are available at http://muse-model.github.io.

specfiying the numbers in the text prompt -> generative that specified cardinality properly -> 3 bottles, 2 books

Image quality and diversity quantification - Firchlet inception distance

CC3M is a dataset of 3 million image and caption.(image - text pairs)

adding noise to the image in pixels and learns to generate images from noise

1. Introduction

Generative image models conditioned on text prompts have taken an enormous leap in quality and flexibility in the last few years (Ramesh et al., 2022; Nichol et al., 2021; Saharia et al., 2022; Yu et al., 2022; Rombach et al., 2022; Midjourney, 2022). This was enabled by a combination of deep learning architecture innovations (Van Den Oord et al., 2017; Vaswani et al., 2017); novel training paradigms such as masked modeling for both language (Devlin et al., 2018; Raffel et al., 2020) and vision tasks (He et al., 2022; Chang et al., 2022); new families of generative models such as diffusion (Ho et al., 2020; Rombach et al., 2022; Saharia et al., 2022) and masking-based generation (Chang et al., 2022); and finally, the availability of large scale image-text paired datasets (Schuhmann et al., 2021).

In this work, we present a new model for text-to-image synthesis using a masked image modeling approach (Chang et al., 2022). Our image decoder architecture is conditioned on embeddings from a pre-trained and frozen T5-XXL (Raffel et al., 2020) large language model (LLM) encoder. In agreement with Imagen (Saharia et al., 2022), we find that conditioning on a pre-trained LLM is crucial for photorealistic, high quality image generation. Our models (except for the VQGAN quantizer) are built on the Transformer (Vaswani et al., 2017) architecture.

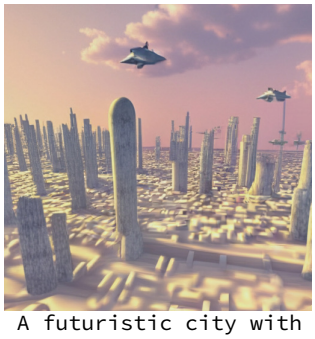
encoder nd decoder in vqgan is cnn based

We have trained a sequence of Muse models, ranging in size from 632M parameters to 3B parameters (for the image decoder; the T5-XXL model has an additional 4.6B parameters). Each model consists of several sub-models (Figure 3): First, we have a pair of VQGAN "tokenizer" models (Esser et al., 2021b), which can encode an input image to a sequence of discrete tokens as well as decode a token sequence back to an image. We use two VQGANs, one for 256x256 resolution ("low-res") and another for 512x512 resolution ("high-res"). Second, we have a base masked image model, which contains the bulk of our parameters. This model takes a sequence of partially masked low-res tokens and predicts the marginal distribution

*Equal contribution †Core contribution. Correspondence to: Huiwen Chang <huiwenchang@google.com>, Han Zhang <zhanghan@google.com>, Dilip Krishnan <dilipkay@google.com>.

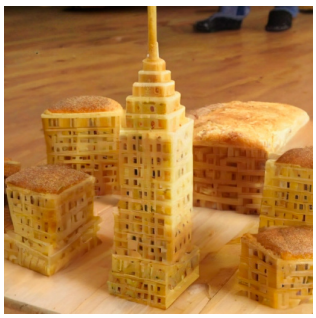
A fluffy baby sloth with a knitted hat trying to figure out a laptop, close up.

A sheep in a wine glass.



A futuristic city with flying cars.

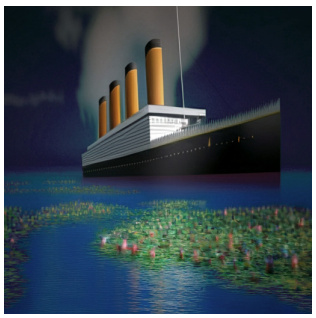
A large array of colorful cupcakes, arranged on a maple table to spell MUSE.



Manhattan skyline made of bread.

Astronauts kicking a football in front of Eiffel tower.

Two cats doing research.



3D mesh of Titanic floating on a water lily pond in the style of Monet.

A storefront with 'Muse' written on it, in front of Matterhorn Zermatt.

A surreal painting of a robot making coffee.

A cake made of macarons in a unicorn shape.

Three dogs celebrating Christmas with some champagne.

Figure 1. Muse text-to-image generation (512 × 512 resolution). Under each generated image, the corresponding caption is shown, exhibiting a variety of styles, captions and understanding. Each image was generated in 1.3s on a TPUv4 chip.

for each masked token, conditioned on the unmasked tokens and a T5XXL text embedding. Third, we have a "superres" transformer model which translates (unmasked) low-res tokens into high-res tokens, again conditioned on T5-XXL text embeddings. We explain our pipeline in detail in Section 2.

Compared to Imagen (Saharia et al., 2022) or Dall-E2 (Ramesh et al., 2022) which are built on cascaded pixel-space diffusion models, Muse is significantly more efficient due to the use of discrete tokens; it can be thought of as a discrete diffusion process with the absorbing state ([MASK]) (Austin et al., 2021). Compared to Parti (Yu et al., 2022), a state-of-the-art autoregressive model, Muse is more efficient due to the use of parallel decoding. Based on comparisons on similar hardware (TPU-v4 chips), we estimate that Muse is more than 10x faster at inference time than either Imagen-3B or Parti-3B models and 3x faster than Stable Diffusion v1.4 (Rombach et al., 2022) (see Section 3.2.2). All these comparisons are when images of the same size: either 256 × 256 or 512 × 512. Muse is also faster than Stable Diffusion (Rombach et al., 2022), in spite of both models working in the latent space of a VQGAN. We believe that this is due to the use of a diffusion model in Stable

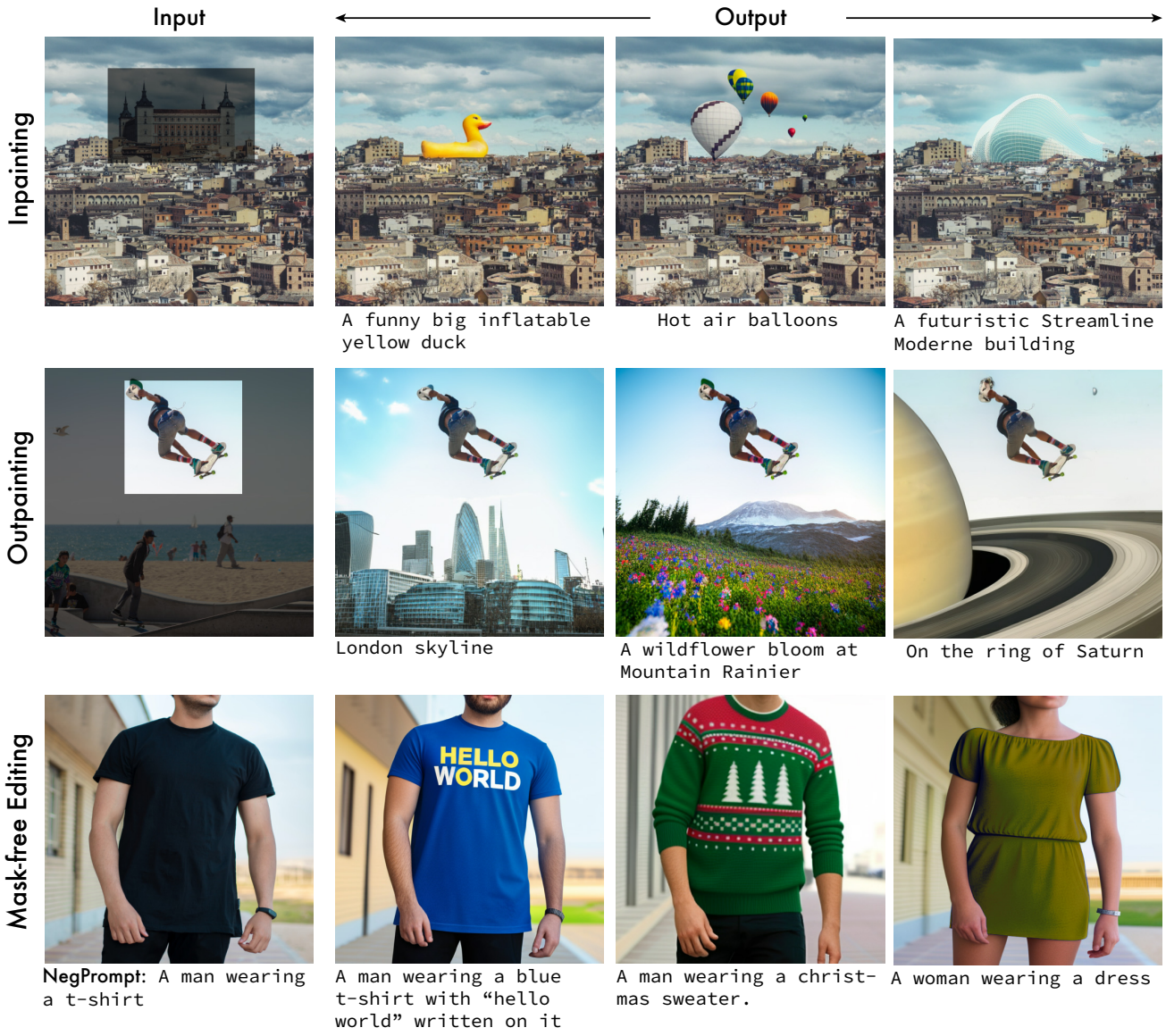


Figure 2. Examples of zero-shot text-guided image editing using Muse. We show examples of a number of editing applications using the Muse text-to-image generative model, on *real* input images, without fine-tuning. All edited images are generated at 512×512 resolution.

Diffusion v1.4 which requires a significantly higher number of iterations at inference time.

The efficiency improvement of Muse, however, does *not* come at a loss of generated image quality or semantic understanding of the input text prompt. We evaluate our output on multiple criteria, including CLIP score (Radford et al., 2021) and FID (Heusel et al., 2017). The former is a measure of image-text correspondence; and the latter a measure of image quality and diversity. Our 3B parameter model achieves a CLIP score of 0.32 and an FID score of 7.88 on the COCO (Lin et al., 2014) zero-shot validation benchmark, which compares favorably with that of other large-scale text-to-image models (see Table 2). Our 632M(base)+268M(super-res) parameter model achieves a state of the art FID score of 6.06 when trained and evaluated on the CC3M (Sharma et al., 2018) dataset, which is significantly lower than all other reported results in the literature (see Table 1). We also evaluate our generations on the PartiPrompts (Yu et al., 2022) evaluation suite with human raters, who find that Muse generates images better aligned with its text prompt 2.7x more often than Stable Diffusion v1.4 (Rombach et al., 2022).

Muse generates images that reflect different parts of speech in input captions, including nouns, verbs and adjectives. Furthermore, we present evidence of multi-object properties understanding, such as compositionality and cardinality, as

well image style understanding. See Figure 1 for a number of these examples and our website http://muse-model.github.io for more examples. The mask-based training of Muse lends itself to a number of zero-shot image editing capabilities. A number of these are shown in Figure 2, including zero-shot, text-guided inpainting, outpainting and mask-free editing. More details are in Section 3. Our contributions are:

1. We present a state-of-the-art model for text-to-image generation which achieves excellent FID and CLIP scores (quantitative measures of image generation quality, diversity and alignment with text prompts).
2. Our model is significantly faster than comparable models due to the use of quantized image tokens and parallel decoding.
3. Our architecture enables out-of-the-box, zero-shot editing capabilities including inpainting, outpainting, and mask-free editing.

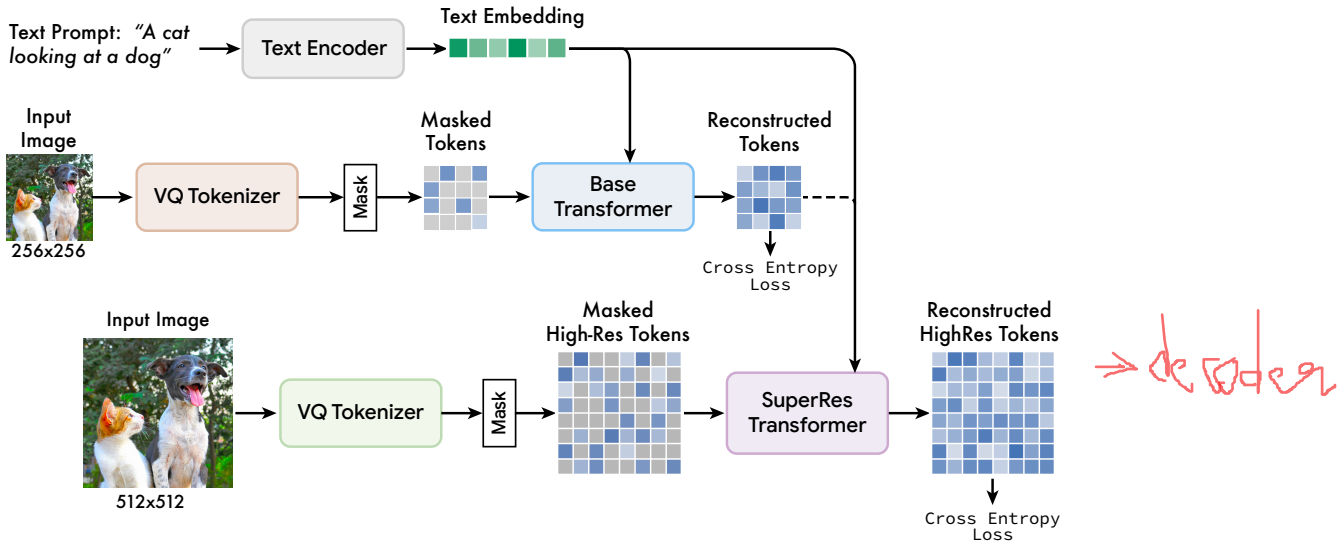


Figure 3. Muse Framework: We show the training pipeline for our model, with the T5-XXL pre-trained text encoder, base model and super-resolution model depicted on the three rows. The text encoder generates a text embedding that is used for cross-attention with image tokens for both base and super-res Transformer layers. The base model uses a VQ Tokenizer that is pre-trained on lower resolution (256×256) images and generates a 16×16 latent space of tokens. This sequence is masked at a variable rate per sample and then the cross-entropy loss learns to predict the masked image tokens. Once the base model is trained, the reconstructed lower-resolution tokens and text tokens are passed into the super-res model that then learns to predict masked tokens at a higher resolution.

2. Model

Our model is built on a number of components. Here, we provide an overview of each of those components in the order of their training, while relegating many details of the architecture and parameters to the Appendix. Figure 3 provides an overview of the model architecture.

2.1. Pre-trained Text Encoders

Similar to the findings in (Saharia et al., 2022), we find that leveraging a pre-trained large language model (LLM) is beneficial to high-quality image generation. The embeddings extracted from an LLM such as T5-XXL (Raffel et al., 2020) carry rich information about objects (nouns), actions (verbs), visual properties (adjectives), spatial relationships (prepositions), and other properties such as cardinality and composition. Our hypothesis is that the Muse model learns to map these rich visual and semantic concepts in the LLM embeddings to the generated images; it has been shown in recent work (Merullo et al., 2022) that the conceptual representations learned by LLM's are roughly linearly mappable to those learned by models trained on vision tasks. Given an input text caption, we pass it through the frozen T5-XXL encoder, resulting in a sequence of 4096 dimensional language embedding vectors. These embedding vectors are linearly projected to the hidden size of our Transformer models (base and super-res).

2.2. Semantic Tokenization using VQGAN

A core component of our model is the use of semantic tokens obtained from a VQGAN (Esser et al., 2021b) model. This model consists of an encoder and an decoder, with a quantization layer that maps an input image into a sequence of tokens from a learned codebook. We build our encoder and decoder entirely with convolutional layers to support encoding images from different resolutions. The encoder has several downsampling blocks to reduce the spatial dimension of the input, while the decoder has the corresponding number of upsampling blocks to map the latents back into original image size. Given an image of size $H \times W$, the encoded token is of size ${}^{H}/_{f} \times {}^{W}/_{f}$, with downsampling ratio f. We train two VQGAN models: one with downsampling ratio $f = 16$ and the other with downsampling ratio $f = 8$. We obtain tokens for our base model using the $f = 16$ VQGAN model on 256×256 pixel images, thus resulting in tokens with spatial size 16×16. We obtain the tokens for our super-resolution model using the $f = 8$ VQGAN model on 512×512 images, and the corresponding token has spatial size 64×64. As mentioned in previous work (Esser et al., 2021b), the resulting discrete tokens after encoding capture higher-level semantics of the image, while ignoring low level noise. Furthermore, the discrete nature of these tokens allows us to use a cross-entropy loss at the output to predict masked tokens in the next stage.

2.3. Base Model

Our base model is a masked transformer(Vaswani et al., 2017; Devlin et al., 2018), where the inputs are the projected T5 embeddings and image tokens. We leave all the text embeddings unmasked and randomly mask a varying fraction of image tokens (see Section 2.6) and replace them with a special `[MASK]` token (Chang et al., 2022). We then linearly map image tokens into image input embeddings of the required Transformer input/hidden size along with learned 2D positional embeddings. Following previous transformer architecture (Vaswani et al., 2017), we use several transformer layers including self-attention block, cross-attention block and MLP block to extract features. At the output layer, an MLP is used to convert each masked image embedding to a set of logits (corresponding to the VQGAN codebook size) and a cross-entropy loss is applied with the ground truth token label as the target. At training, the base model is trained to predict all masked tokens at each step. However, for inference, mask prediction is performed in an iterative manner which significantly increases quality. See Section 2.8 for details.

2.4. Super-Resolution Model

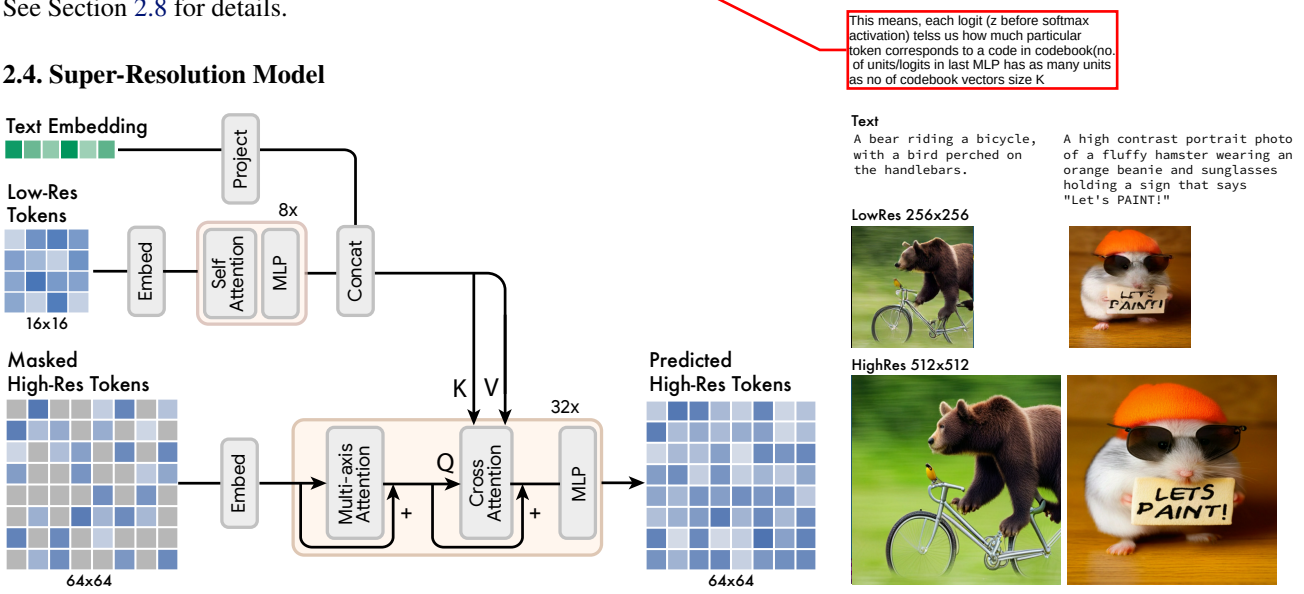


Figure 4. Super-resolution Model. On the left is shown the architecture of the super-resolution model. Low-resolution tokens are passed into a series of self-attention Transformer layers; and the resulting output embeddings are concatenated with text embeddings extracted from the conditioning text prompt. Following this, cross-attention is applied from these concatenated embeddings to the masked high-resolution tokens; the loss learns to predict these masked tokens conditioned on the low-resolution and text tokens. On the right are shown two examples of the improvement brought about by the super-resolution model.

We found that directly predicting 512×512 resolution leads the model to focus on low-level details over large-scale semantics. As a result we found it beneficial to use a cascade of models: first a base model that generates a 16×16 latent map (corresponding to a 256×256 image), followed by a super-resolution model that upsamples the base latent map to a 64×64 latent map (corresponding to a 512×512 image). The super-res model is trained after the base model has been

trained.

As mentioned in Section 2.2, we trained two VQGAN models, one at 16×16 latent resolution and 256×256 spatial resolution, and the second at 64×64 latent resolution and 512×512 spatial resolution. Since our base model outputs tokens corresponding to a 16×16 latent map, our super-resolution procedure learns to "translate" the lower-resolution latent map to the higher-resolution latent map, followed by decoding through the higher-resolution VQGAN to give the final high-resolution image. This latent map translation model is also trained with text conditioning and cross-attention in an analogous manner to the base model, as shown in Figure 4.

2.5. Decoder Finetuning

To further improve our model's ability to generate fine details, we increase the capacity of the VQGAN decoder by the addition of more residual layers and channels while keeping the encoder capacity fixed. We then finetune the new decoder layers while keeping the VQGAN encoder weights, codebook and transformers (i.e., base model and super resolution model) frozen. This allows us to improve our visual quality without re-training any of the other model components (because the visual token "language" stays fixed). This is shown in Figure 13 in the Appendix, where we see that the finetuned decoder can reconstruct more sharper details in the store front. We also give details of the finetuned decoder architecture in the Appendix.

2.6. Variable Masking Rate

As was done in (Chang et al., 2022), we train our model with a variable masking rate based on a Cosine scheduling: for each training example, we sample a masking rate $r \in [0, 1]$ from a truncated $\arccos$ distribution with density function $p(r) = \frac{2}{\pi}(1 - r^2)^{-\frac{1}{2}}$. This has an expected masking rate of 0.64, with a strong bias towards higher masking rates. The bias towards higher masking rates makes the prediction problem harder. In contrast with autoregressive approaches, which learn conditional distributions $P(x_i|x_{<i})$ for some fixed ordering of tokens, random masking with a variable masking ratio allows our models to learn $P(x_i|x_\Lambda)$ for arbitrary subsets of tokens Λ. This is not only critical for our parallel sampling scheme, but it also enables a number of zero-shot, out-of-the-box editing capabilities, such as shown in Figure 2 and Section 3.3.

this us just lambda(arcane lambda)

its just cosine dustribution

2.7. Classifier Free Guidance

We employ classifier-free guidance (CFG) (Ho & Salimans, 2022) to improve our generation quality and our text-image alignment. At training time, we remove text conditioning on 10% of samples chosen randomly (thus attention reduces to image token self-attention). At inference time, we compute a conditional logit ℓ_c and an unconditional logit ℓ_u for each masked token. We then form the final logits ℓ_g by moving away from the unconditional logits by an amount t, the *guidance scale*:

cuz cfg tries to push generation logits close to text embeddings nd hence diversity is compromised

$$\ell_g = (1 + t)\ell_c - t\ell_u \tag{1}$$

no cross attention btwn text embeddings and image embeddings fir these 10% tokens as the text embeddings are kept as 0 for them

Intuitively, CFG trades off diversity for fidelity. Different from previous approaches, we reduce the hit to diversity by linearly increasing the guidance scale t through the sampling procedure. This allows the early tokens to be sampled more freely, with low or no guidance, but increases the influence of the conditioning prompt for the later tokens.

We also exploit this mechanism to enable *negative prompting* (NegPrompt, 2022) by replacing the unconditional logit ℓ_u with a logit conditioned on a "negative prompt". This encourages the resulting image to have features associated with the positive prompt ℓ_c and remove features associated with the negative prompt ℓ_u.

Similar to MaskGit

2.8. Iterative Parallel Decoding at Inference

The critical component for our model's inference time efficiency is the use of parallel decoding to predict multiple output tokens in a single forward pass. The key assumption underlying the effectiveness of the parallel decoding is a Markovian property that many tokens are conditionally independent given other tokens. Decoding is performed based on a cosine schedule (Chang et al., 2022) that chooses a certain fixed fraction of the highest confidence masked tokens that are to be predicted at that step. These tokens are then set to unmasked for the remainder of the steps and the set of masked tokens is appropriately reduced. Using this procedure, we are able to perform inference of 256 tokens using only 24 decoding steps in our base model and 4096 tokens using 8 decoding steps in our super-resolution model, as compared to the 256 or 4096 steps required for autoregressive models (e.g. (Yu et al., 2022)) and hundreds of steps for diffusion models (e.g., (Rombach et al., 2022; Saharia et al., 2022)). We note that recent methods including progressive distillation (Salimans & Ho, 2022) and

Figure 5. Inference samples. We visualize the evolution of masked tokens over the sequence of steps for the base model (left) and the super-res model (right). The super-res model, being conditioned on the low-res tokens, requires significantly fewer sampling steps for convergence.

better ODE solvers (Lu et al., 2022) have greatly reduced the sampling steps of diffusion models, but they have not been widely validated in large scale text-to-image generation. We leave the comparison to these faster methods in the future work, while noting that similar distillation approaches are also a possibility for our model.

3. Results

We train a number of base Transformer models at different parameter sizes, ranging from 600M to 3B parameters. Each of these models is fed in the output embeddings from a T5-XXL model, which is pre-trained and frozen and consists of 4.6B parameters. Our largest base model of 3B parameters consists of 48 Transformer layers with cross-attention from text to image and self-attention among image tokens. All base models share the same image tokenizer. We use a CNN model with 19 ResNet blocks and a quantized codebook of size 8192 for the tokenization. Larger codebook sizes did not result in performance improvements. The super-resolution model consists of 32 multi-axis Transformer layers (Zhao et al., 2021) with cross-attention from concatenated text and image embedding to high resolution image and self-attention among high resolution image tokens. This model converts a sequence of tokens from one latent space to another: the first latent space being that of the base model tokenizer, a latent space of 16×16 tokens, to that of a higher resolution tokenizer with 64×64 tokens. After token conversion, the decoder for the higher resolution tokenizer is used to convert to the higher resolution image space. Further details of configurations are provided in the appendix.

We train on the Imagen dataset consisting of 460M text-image pairs (Saharia et al., 2022). Training is performed for 1M steps, with a batch size of 512 on 512-core TPU-v4 chips (Jouppi et al., 2020). This takes about 1 week of training time. We use the Adafactor optimizer (Shazeer & Stern, 2018) to save on memory consumption which allowed us to fit a 3B parameter model without model parallelization. We also avoid performing exponential moving averaging (EMA) of model weights during training, again to save on TPU memory. In order to reap the benefits of EMA, we checkpoint every 5000 steps, then perform EMA offline on the checkpointed weights with a decay factor of 0.7. These averaged weights form the final base model weights.

3.1. Qualitative Performance

Figure 6 qualitatively demonstrates the capabilities of Muse for text prompts with different properties. The top left of Figure 6 shows examples that demonstrate a basic understanding of cardinality. For objects with non-unity cardinality, instead of generating the same object pixels multiple times, Muse instead adds contextual variations to make the overall image more realistic, e.g., elephant size and orientation, wine bottle wrapper color, and tennis ball rotation. The top right of Fig, 6 demonstrates understanding of multi-object composition and relativeness. Instead of placing objects at random locations, Muse generates images that preserve prepositional object relations in the text, e.g., on vs under, left vs right, etc. The middle left of Figure 6 demonstrates its ability to generate images spanning many styles, both specific to a renowned artist (e.g., Rembrandt) as well as general to a style as a whole (e.g., pop art and Chinese ink and wash). The middle right of Figure 6 demonstrates the ability of Muse to render words and phrases. Text generation is fundamentally different than generating most other objects. Instead of the model learning a mapping between an object name and its characteristics (e.g., that "elephant" maps to "large", "gray", and "peanut eating"), the virtual continuum of possible words and phrases demands that the model learn differently. It must instead learn a hierarchical understanding between phrases, words, and letters. The bottom left of Figure 6 demonstrates that Muse uses the entirety of a text prompt when rendering instead of focusing

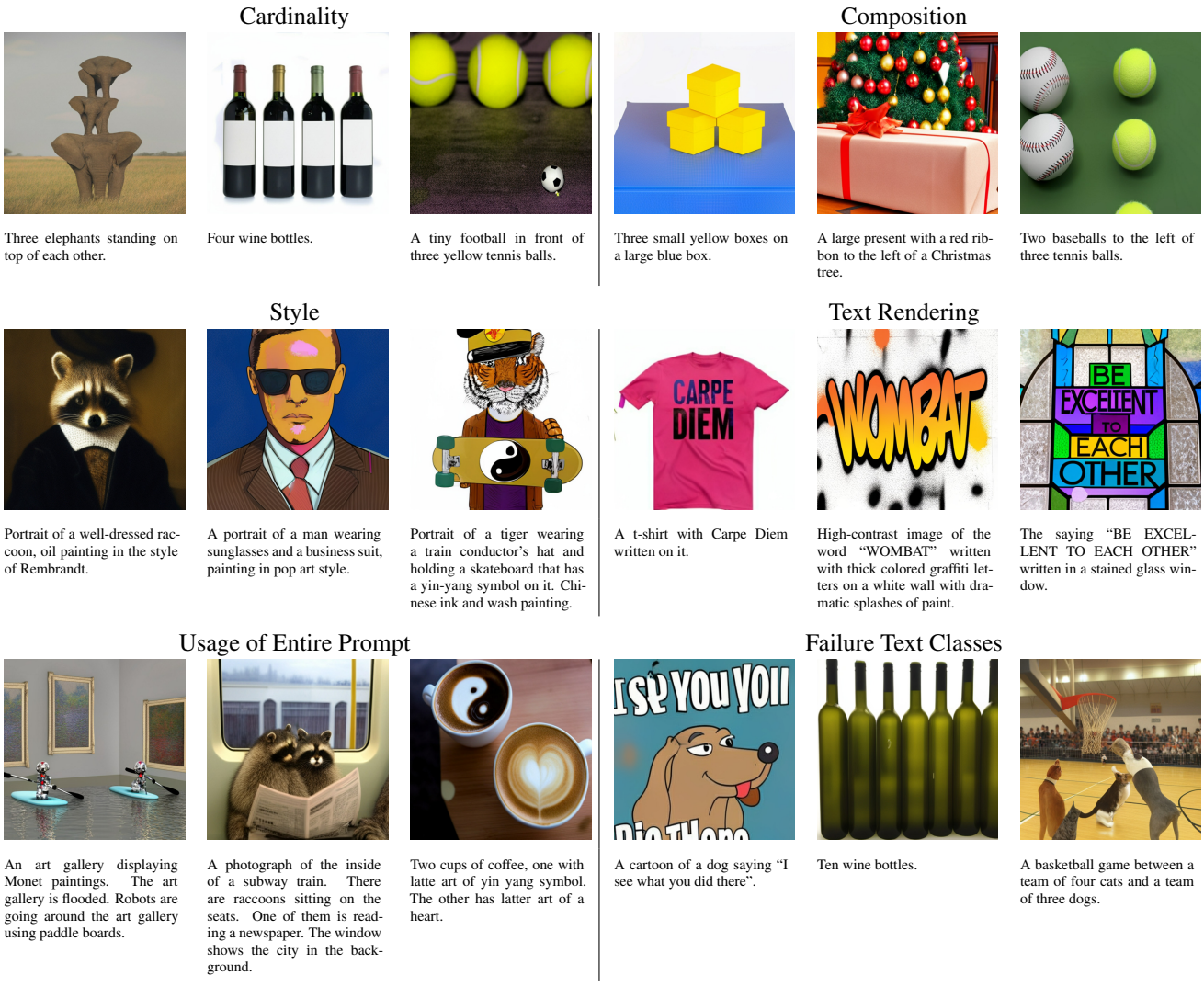


Figure 6. Examples demonstrating text-to-image capabilities of Muse for various text properties. Top left: cardinality; top right: composition; middle left: style; middle right: text rendering; and bottom left: usage of the entire prompt. For all examples, 16 instances per prompt were generated, and the one with the highest CLIP score (Radford et al., 2021) was chosen. Bottom right: examples of generated image failure in Muse for various text properties such as direct rendering of long phrases, high cardinalities, and multiple cardinalities.

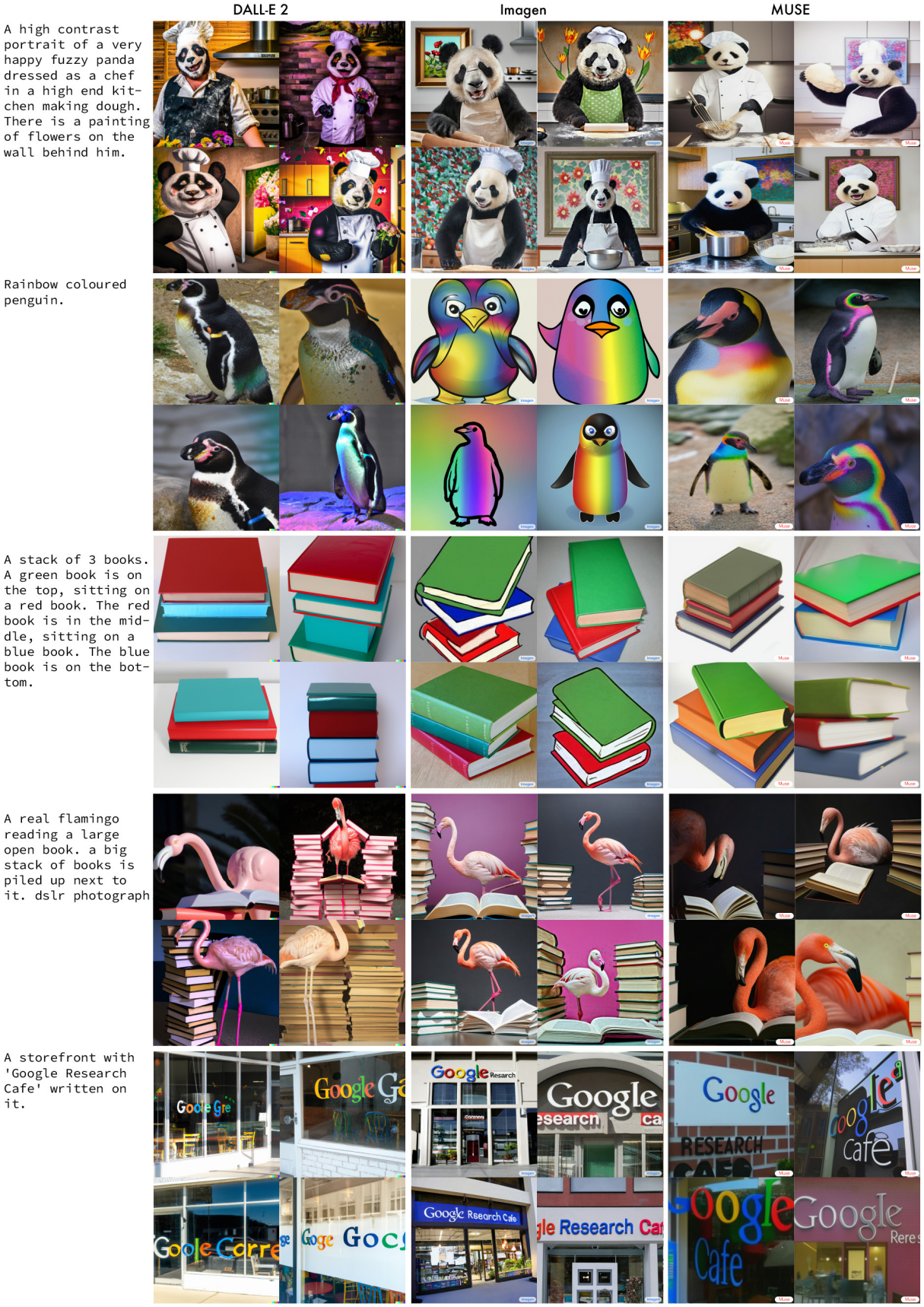


Figure 7. Comparing the same prompts across DALL-E2 (Ramesh et al., 2022) (left), Imagen (Saharia et al., 2022) (middle) and Muse (right).

Approach	Model Type	Params	FID	CLIP
VQGAN (Esser et al., 2021b)	Autoregressive	600M	28.86	0.20
ImageBART (Esser et al., 2021a)	Diffusion+Autogressive	2.8B	22.61	0.23
LDM-4 (Rombach et al., 2022)	Diffusion	645M	17.01	0.24
RQ-Transformer (Lee et al., 2022a)	Autoregressive	654M	12.33	0.26
Draft-and-revise (Lee et al., 2022b)	Non-autoregressive	654M	9.65	0.26
Muse(base model)	Non-autoregressive	632M	6.8	0.25
Muse(base + super-res)	Non-autoregressive	632M + 268M	6.06	0.26

Table 1. Quantitative evaluation on CC3M (Sharma et al., 2018); all models are trained and evaluated on CC3M.

Approach	Model Type	Params	FID-30K	Zero-shot FID-30K
AttnGAN (Xu et al., 2017)	GAN		35.49	-
DM-GAN (Zhu et al., 2019)	GAN		32.64	-
DF-GAN (Tao et al., 2020)	GAN		21.42	-
DM-GAN + CL (Ye et al., 2021)	GAN		20.79	-
XMC-GAN (Zhang et al., 2021)	GAN		9.33	-
LAFITE (Zhou et al., 2021)	GAN		8.12	-
Make-A-Scene (Gafni et al., 2022)	Autoregressive		7.55	-
DALL-E (Ramesh et al., 2021)	Autoregressive		-	17.89
LAFITE (Zhou et al., 2021)	GAN		-	26.94
LDM (Rombach et al., 2022)	Diffusion		-	12.63
GLIDE (Nichol et al., 2021)	Diffusion		-	12.24
DALL-E 2 (Ramesh et al., 2022)	Diffusion		-	10.39
Imagen-3.4B (Saharia et al., 2022)	Diffusion		-	7.27
Parti-3B (Yu et al., 2022)	Autoregressive		-	8.10
Parti-20B (Yu et al., 2022)	Autoregressive		3.22	7.23
Muse-3B	Non-Autoregressive		-	7.88

Table 2. Quantitative evaluation of FID and CLIP score (where available) on MS-COCO (Lin et al., 2014) for 256×256 image resolution. Muse achieves a CLIP score of 0.32, higher than the score of 0.27 reported in Imagen. Other papers in the table above did not report a CLIP score.

exclusively on only a few salient words. Finally, Figure 7 shows comparisons between Muse, Dall-E 2 (Ramesh et al., 2022), and Imagen (Saharia et al., 2022) for some select prompts, showing that Muse is at par with Imagen and qualitatively better than Dall-E2 for many prompts.

However, as demonstrated in the bottom right of Figure 6, Muse is limited in its ability to generate images well aligned with certain types of prompts. For prompts which indicate that long, multi-word phrases should be directly rendered, Muse has a tendency to render those phrases incorrectly, often resulting in (unwanted) duplicated rendered words or rendering of only a portion of the phrase. Additionally, prompts indicating high object cardinality tend to result in generated images which do not correctly reflect that desired cardinality (e.g., rendering only 7 wine bottles when the prompt specified 10). In general, the ability of Muse to render the correct cardinalities of objects decreases as the cardinality increases. Another difficult prompt type for Muse is ones with multiple cardinalities (e.g., "four cats and a team of three dogs"). For such cases, Muse has a tendency to get at least one cardinality incorrect in its rendering.

Muse limitation case

3.2. Quantitative Performance

In Table 1 and Table 2, we show our performance against other methods on the CC3M (Sharma et al., 2018) and COCO (Lin et al., 2014) datasets as measured by Fréchet Inception Distance (FID) (Heusel et al., 2017), which measures quality and diversity of samples, as well as CLIP (Radford et al., 2021) score, which measures image/text alignment. For the CC3M

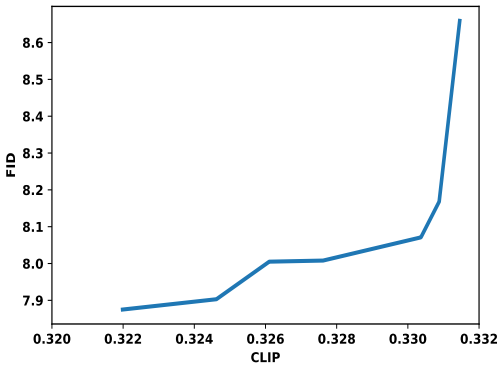


Figure 8. CLIP vs. FID tradeoff curve. We perform sweeps of sampling parameters for a fixed model, then plot the Pareto front.

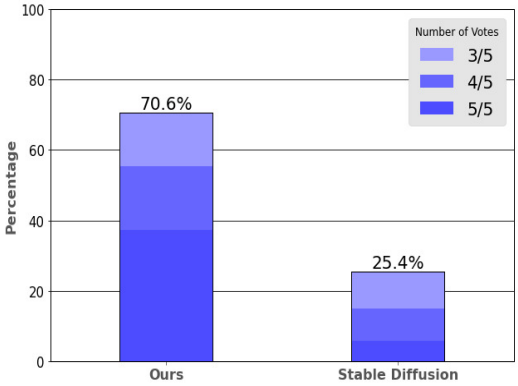


Figure 9. Percentage of prompts for which a human rater consensus chose a model alignment preference. Contributions from specific numbers of rater consensuses are shown in different colors, while marginals over consensuses ($= 5$, ≥ 4, and ≥ 3) are shown numerically.

3 rater consensus" refers to a scenario where three individuals or raters are involved in making judgments or assessments, and the term specifically emphasizes the level of agreement or consensus among these three raters. This concept is often encountered in situations where multiple opinions or evaluations are needed to provide a more comprehensive or reliable assessment.

For example, in the context of image annotation or data labeling, three raters might be asked to independently label or classify certain elements within an image. The level of consensus among these three raters could be measured to evaluate the reliability and consistency of the annotations.

results, both Muse models were trained on CC3M. The COCO results are zero-shot, using a model trained on the same dataset as Imagen (Saharia et al., 2022).

Our 632M model achieves SOTA results on CC3M, significantly improving upon the state of the art in FID score, and also achieving state of the art CLIP score. Our 3B model achieves an FID score of 7.88 which is slightly better than the score of 8.1 achieved by the Parti-3B model which has a similar number of parameters. Our CLIP score of 0.32 is higher than the CLIP score of 0.29 achieved by Imagen (which is achieved when the FID is significantly higher 20). For the FID of 7.27, Imagen achieves a CLIP score of around 0.27 (see Figure 4 in (Saharia et al., 2022)).

Our sampling algorithm (Section 2.8) has a number of hyperparameters, such as guidance scale, sampling temperature, whether or not to linearly increase guidance during sampling, etc. We perform evaluation sweeps over these parameters. We find subsets of sampling parameters that are Pareto efficient, in the sense that we cannot improve FID without hurting CLIP. This allows us to study the tradeoff between diversity and image/text alignment, which we show in Figure 8.

3.2.1. Human Evaluation

Similar to previous works (Yu et al., 2022; Saharia et al., 2022), we perform side-by-side evaluations in which human raters are presented with a text prompt and two images, each generated by a different text-to-image model using that prompt. The raters are asked to assess prompt-image alignment via the question, "Which image matches with the caption better?" Each image pair is anonymized and randomly ordered (left vs right). Raters have the option of choosing either image or that they are indifferent[1]. Each (prompt, image pair) triplet is assessed by five independent raters; the raters were provided through the Google internal crowd computing team and were completely anonymous to the Muse team. For the set of prompts presented to raters, we used PartiPrompts (Yu et al., 2022), a collection of 1650 text prompts curated to measure model capabilities across a variety of categories. For the two text-to-image models, we compared Muse (3B parameters) to that of Stable Diffusion v1.4 (Rombach et al., 2022), the text-to-image model most comparable to Muse in terms of inference speed. For each prompt, 16 image instances were generated, and the one with the highest CLIP score (Radford et al., 2021) was used. The stable diffusion images were generated via the CompVis Stable Diffusion v1.4 notebook (CompVis, 2022). We required at least a 3 rater consensus for results to be counted in favor of a particular model. From this analysis, we found that Muse was chosen as better aligned than Stable Diffusion for 70.6% of the prompts, Stable Diffusion was chosen as better aligned than Muse for 25.4%, and no rater consensus was chosen for 4%. These results are consistent with Muse having significantly better caption matching capability ($\sim$2.7x). Figure 9 shows a breakdown of the rater results for rater consensuses of 3, 4, and all 5 possible votes. Prompts for which all 5 raters said Muse had better alignment than Stable

[1]Choosing indifference makes sense when neither image is aligned with the text prompt and helps reduce statistical noise in the results.

Diffusion are the larger contributor.

In addition to measuring alignment, other works (Yu et al., 2022; Saharia et al., 2022) have also measured image realism, often via a rater question similar to, "Which image is more realistic?". However, we note that care must be taken with examination of such results. Though it is not the intent of the question, a model that is completely mode collapsed so that it generates the same sufficiently realistic image regardless of prompt will virtually always do better on this question than a model that *does* take the prompt into account during image generation. We propose this type of question is only applicable between models of similar alignment. Since Muse is significantly better aligned than Stable Diffusion, we did not assess realism via human raters. We consider this topic an area of open research.

3.2.2. Inference Speed

In Table 3, we compare the inference time of Muse to several other popular models. We benchmarked Parti-3B, Imagen, and Muse-3B internally on TPUv4 accelerators. For Stable Diffusion/LDM, we used the fastest reported benchmark (Lambda Labs, 2022), which was done on A100 GPUs. For Stable Diffusion, the TPU implementations we tested were not faster than the A100 implementation. We also report an inference time for LDM with 250 iterations, which is the configuration used to achieve the FID in Table 2. Muse is significantly faster than competing diffusion or autoregressive models, despite having comparable parameter counts (and around 3x more parameters than Stable Diffusion/LDM). The speed advantage of Muse over Imagen is due to the use of discrete tokens and requiring fewer sampling iterations. The speed advantage of Muse over Parti is due to the use of parallel decoding. The speed advantage of Muse over Stable Diffusion is primarily attributable to requiring fewer sampling iterations.

Model	Resolution	Time
Imagen	256×256	9.1s
Imagen	1024×1024	13.3s
LDM (50 steps)	512×512	3.7s
LDM (250 steps)	512×512	18.5s
Parti-3B	256×256	6.4s
Muse-3B	256×256	0.5s
Muse-3B	512×512	1.3s

Table 3. Per-batch inference time for several models. Muse, Imagen, and Parti were benchmarked internally on TPUv4 hardware. Stable Diffusion/LDM benchmark from (Lambda Labs, 2022), on A100 GPUs. The "LDM (250 steps)" time comes from scaling the 50-step time by 5; 250 steps were used to achieve the FID in Table 2.

3.3. Image Editing

By exploiting the fact that our model can condition on arbitrary subsets of image tokens, we can use the model out-of-the-box for a variety of image editing applications with no additional training or model fine-tuning.

predict, sample, mask schedule, mask

3.3.1. Text-guided Inpainting / outpainting

Our sampling procedure (Section 2.8) gives us text-guided inpainting and outpainting for free: we convert an input image into a set of tokens, mask out the tokens corresponding to a local region, and then sample the masked tokens conditioned on unmasked tokens and a text prompt. We integrate superresolution through a multi-scale approach: Given an image of size 512x512, we first decimate it to 256x256 and convert both images to high- and low-res tokens. Then, we mask out the appropriate regions for each set of tokens. Next, we inpaint the low-res tokens using the parallel sampling algorithm. Finally, we condition on these low-res tokens to inpaint the high-res tokens using the same sampling algorithm. We show examples of this in Figure 2 and Figure 10.

downscale 512x512 to 256x256 and then convert 512x512 to high res tokens and 256x256 to low res token

3.3.2. Zero-shot Mask-free editing

We use Muse in a zero-shot sense for mask-free image editing of real input images. This method works directly on the (tokenized) image and does not require "inverting" the full generative process, in contrast with recent zero-shot image editing techniques leveraging generative models (Gal et al., 2022b; Patashnik et al., 2021; Kim et al., 2022; Mokady et al., 2022).

We first convert an input image into visual tokens. Next, we iteratively mask and resample a random subset of tokens, conditioned on text prompts. We can think of this as being analogous to a Gibbs sampling procedure, where we fix some tokens and resample others conditioned on them. This has the effect of moving the tokenized image into the typical set of the conditional distribution of images given a text prompt.

We perform the editing using the low-resolution base model, then perform superres on the final output (conditioned on the

Gibbs sampling is a Markov Chain Monte Carlo (MCMC) technique used for generating a sequence of samples from a multivariate probability distribution. It is particularly useful when sampling from a joint distribution with several variables, and it involves iteratively sampling from the conditional distributions of each variable given the values of the other variables. The procedure can be summarized as follows:

Initialize:Start with an initial assignment of values to all variables in the system.
Iterative Sampling:For each variable in the system, sample a new value from its conditional distribution given the current values of all other variables. This involves treating all other variables as fixed and sampling from the distribution of the current variable.
Repeat this process for each variable in the system.
Update:After sampling from all the conditional distributions, update the values of all variables in the system.
Repeat:Repeat the process for a predefined number of iterations or until convergence.
The key idea behind Gibbs sampling is that, even though sampling from the joint distribution might be challenging, sampling from the conditional distributions of individual variables may be more tractable. Over multiple iterations, the samples converge to approximate samples from the joint distribution.

Iterative Masking and Resampling:

The process follows an iterative Gibbs sampling-like procedure. A random subset of tokens is masked and resampled in each iteration.
The sampling is conditioned on text prompts, meaning that the resampling is influenced by the given textual guidance. This conditioning allows the user to guide the editing process by providing relevant text prompts.

Top-k sampling is a technique commonly used in generative models, including image editing models, to control the diversity of generated outputs. In the context of image editing with top-k sampling:
Token Logits Calculation:During the editing process, each token in the tokenized representation of the image has associated logits. Logits represent the unnormalized probabilities assigned to each token for the next iteration.
Top-k Sampling:Top-k sampling involves selecting the top-k tokens with the highest logits for the next iteration. The value of k determines how many tokens are considered in the sampling process.
For example, if k = 3, only the three tokens with the highest logits will be considered for the next iteration.
Diversity Control:By limiting the sampling to the top-k tokens, the diversity of the generated outputs is controlled. This is particularly useful in scenarios where too much diversity could lead to outputs that deviate significantly from the original input.
Stability and Consistency:Top-k sampling adds a level of stability and consistency to the editing process. It ensures that the edits are influenced by the most probable tokens, preventing extreme or unpredictable changes.
Balancing Exploration and Exploitation:The choice of k allows for a balance between exploration and exploitation. A smaller k value tends to exploit the most probable tokens, producing more deterministic and focused edits. A larger k value introduces more exploration and randomness into the editing process.
User Control:In the context of image editing, top-k sampling provides a mechanism for users to influence the editing results. By adjusting the value of k, users can control the degree of variation in the edited images.

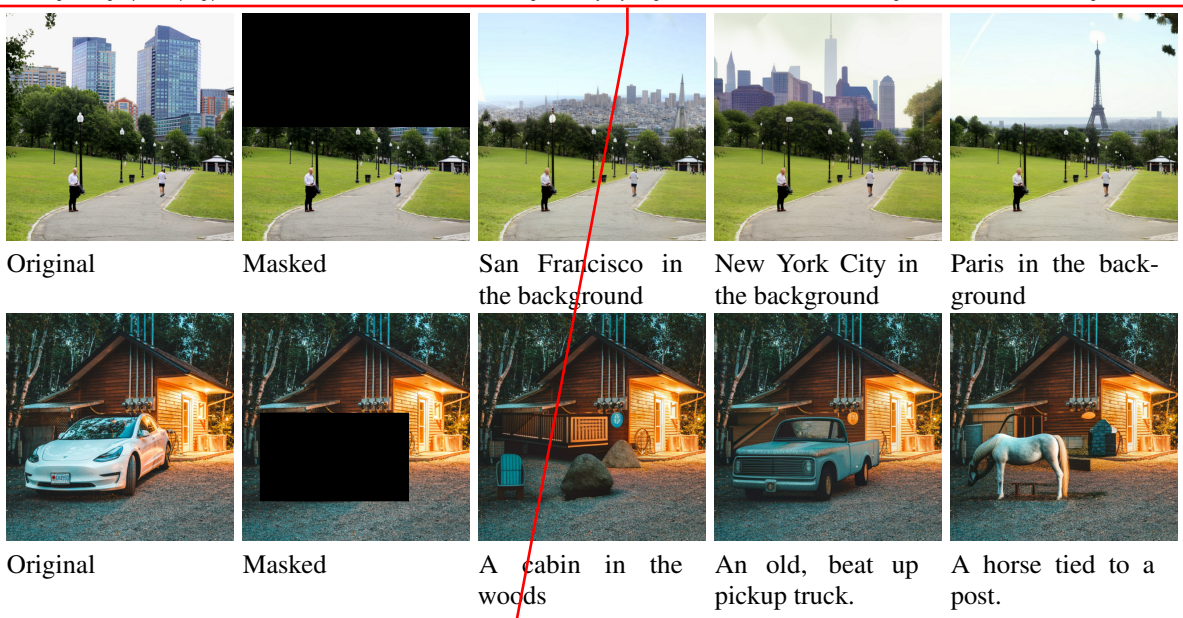


Figure 10. Examples of text-guided inpainting. The mask is shown in the second column of each row. This behavior arises directly from the model with no fine-tuning.

editing prompt). In the examples (Figure 2, Figure 11), we resample 8% of the tokens per iteration for 100 iterations, with a guidance scale of 4. We also perform top-k ($k = 3$) sampling on the token logits to prevent the process from diverging too much from the input. The iterative nature allows for control over the final output. Figure 12 shows a few intermediate edits (without superres); in this example, the user may prefer iteration 50 or 75 over the final output.

The iterative nature of the process allows for control over the final output. The user may choose to stop the iterations at a certain point if a desired result is achieved.
Top-k sampling on the token logits is employed to restrict the diversity of the editing process. This prevents the output from diverging too much from the input, ensuring a controlled and guided editing experience.

4. Related Work

4.1. Image Generation Models

Variational autoencoders (Van Den Oord et al., 2017) and Generative Adversarial Models (GANs) have shown excellent image generation performance with many variants proposed for both convolutional and Transformer architectures e.g. (Goodfellow et al., 2020; Esser et al., 2021b; Karras et al., 2019; Brock et al., 2018; Donahue & Simonyan, 2019). Until recently, GANs were considered state of the art. Diffusion models, based on progressive denoising principles, are now able to synthesize images and video at equal or higher fidelity (Ho et al., 2020; Kingma et al., 2021; Ho et al., 2022). Hybrid approaches that combine principles from multiple approaches have also shown excellent performance (Chang et al., 2022; Lezama et al., 2022), suggesting that there are more complementarities between approaches that can be exploited.

4.2. Image Tokenizers

Image tokenizers are proving to be useful for multiple generative models due to the ability to move the bulk of the computation from input (pixel) space to latents (Rombach et al., 2022), or to enabling more effective loss functions such as classification instead of regression (Chang et al., 2022; Lezama et al., 2022; Li et al., 2022). A number of tokenization approaches such as Discrete VAE's (Rolfe, 2016), VQVAE (Van Den Oord et al., 2017) and VQGAN (Esser et al., 2021b) have been developed, with the latter being the highest-performing as it combines perceptual and adversarial losses to achieve excellent reconstruction. ViT-VQGAN (Yu et al., 2021) extends VQGAN to the Transformer architecture. We use VQGAN rather than ViT-VQGAN as we found it to perform better for our model, noting that a better performing tokenization model does not always translate to a better performing text-to-image model.

4.3. Large Language Models

Our work leverages T5, a pre-trained large language model (LLM) that has been trained on multiple text-to-text tasks (Raffel et al., 2020). LLMs (including T5, BERT (Devlin et al., 2018), and GPT (Brown et al., 2020; Radford et al., 2019)) have

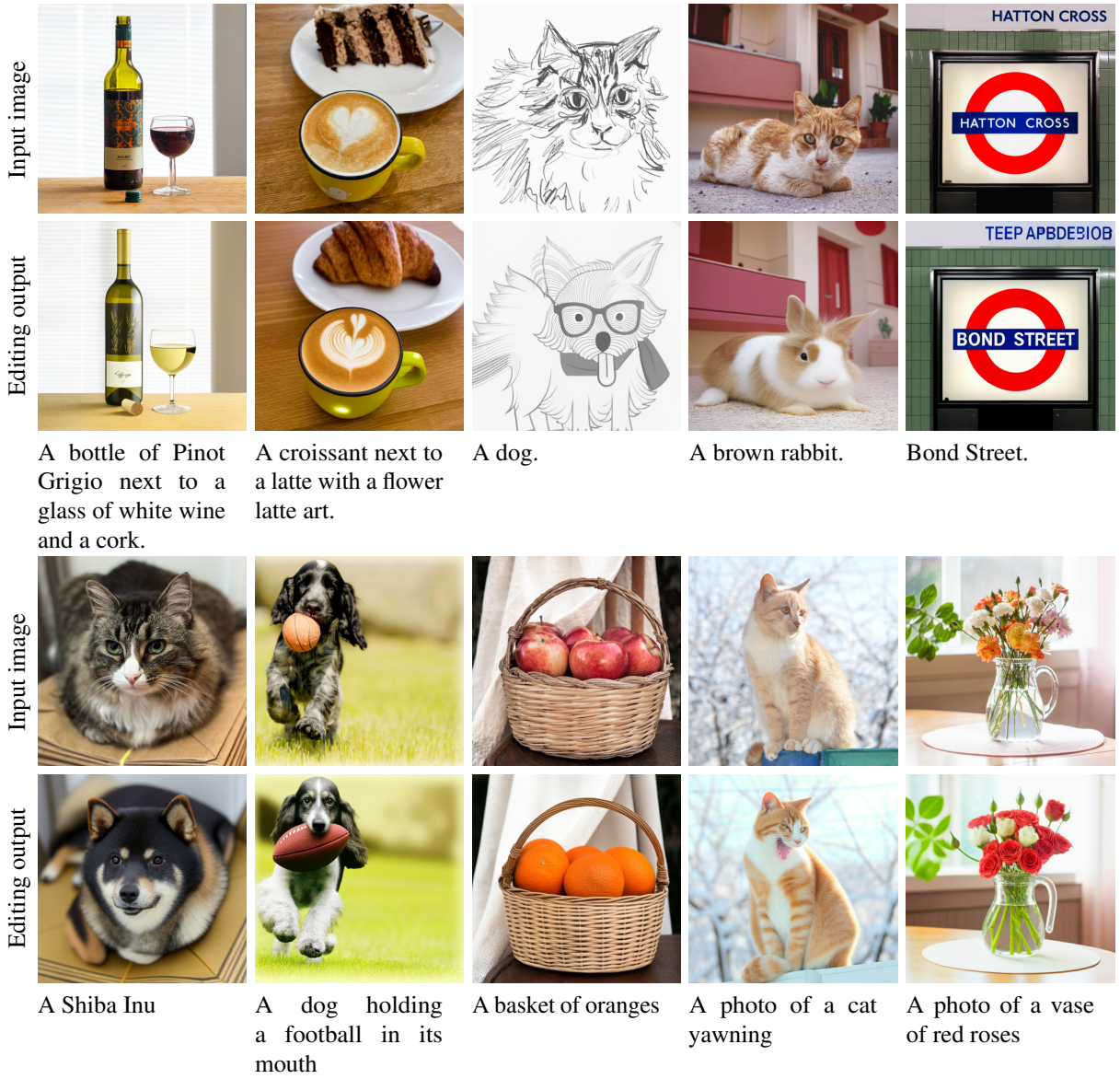


Figure 11. Examples of zero-shot mask-free image editing, post superres. We see that the pose and overall structure of the image is maintained while changing some specific aspects of the object based on the text prompt.

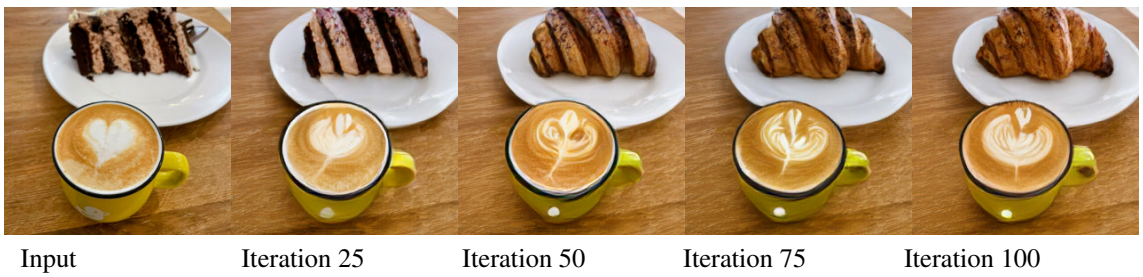


Figure 12. Intermediate iterations producing one of the edits in Figure 11 (pre-superres)

been shown to learn powerful embeddings which enable few-shot transfer learning. We leverage this capacity in our model. All of the modern LLMs are trained on token prediction tasks (either autoregressive or not). The insights regarding the power of token prediction is leveraged in this work, where we apply a transformer to predict *visual* tokens.

4.4. Text-Image Models

Leveraging paired text-image data is proving to be a powerful learning paradigm for representation learning and generative models. CLIP (Radford et al., 2021) and ALIGN (Jia et al., 2021) train models to align pairs of text and image embeddings, showing excellent transfer and few-shot capabilities. Imagen (Saharia et al., 2022) and Parti (Yu et al., 2022) use similar large scale text-image datasets (Schuhmann et al., 2021; 2022) to learn how to predict images from text inputs, achieving excellent results on FID and human evaluations. A key trick is the use of classifier-free guidance (Ho & Salimans, 2022; Dhariwal & Nichol, 2021) that trades off diversity and quality.

4.5. Image Editing with Generative Models

GANs have been extensively studied for image editing and manipulation capabilities (see (Xia et al., 2022) for a survey). A number of techniques have been developed on diffusion models to enable editing, personalization and inversion to token space (Gal et al., 2022a; Meng et al., 2021; Ruiz et al., 2022; Kawar et al., 2022; Brooks et al., 2022; Hertz et al., 2022; Mokady et al., 2022). Dreambooth (Ruiz et al., 2022) and Imagic (Kawar et al., 2022) involve fine-tuning of the generative models. ImagenEditor (Wang et al., 2022) frames the editing task as text-guided image inpainting, and involves user specified masks.

5. Discussion and Social Impact

The Muse model confirms the findings of (Saharia et al., 2022) that frozen large pretrained language models serve as powerful text encoders for text-to-image generation. We also tried in our initial experiments to learn a language model from scratch on the training data, but found that performance was significantly worse than using a pre-trained LLM, especially on long prompts and rare words. We also show that non-diffusion, non-autoregressive models based on the Transformer architecture can perform at par with diffusion models while being significantly more efficient at inference time. We achieve SOTA CLIP scores, showing an excellent alignment beteween image and text. We also show the flexibility of our approach with a number of image editing applications.

We recognize that generative models have a number of applications with varied potential for impact on human society. Generative models (Saharia et al., 2022; Yu et al., 2022; Rombach et al., 2022; Midjourney, 2022) hold significant potential to augment human creativity (Hughes et al., 2021). However, it is well known that they can also be leveraged for misinformation, harassment and various types of social and cultural biases (Franks & Waldman, 2018; Whittaker et al., 2020; Srinivasan & Uchino, 2021; Steed & Caliskan, 2021). Due to these important considerations, we opt to not release code or a public demo at this point in time.

Dataset biases are another important ethical consideration due to the requirement of large datasets that are mostly automatically curated. Such datasets have various potentially problematic issues such as consent and subject awareness (Paullada et al., 2021; Dulhanty, 2020; Scheuerman et al., 2021). Many of the commonly used datasets tend to reflect negative social stereotypes and viewpoints (Prabhu & Birhane, 2020). Thus, it is quite feasible that training on such datasets simply amplifies these biases and significant additional research is required on how to mitigate such biases, and generate datasets that are free of them: this is a very important topic (Buolamwini & Gebru, 2018; Hendricks et al., 2018) that is out of the scope of this paper.

Given the above considerations, we do not recommend the use of text-to-image generation models without attention to the various use cases and an understanding of the potential for harm. We especially caution against using such models for generation of people, humans and faces.

Acknowledgements

We thank William Chan, Chitwan Saharia, and Mohammad Norouzi for providing us training datasets, various evaluation codes and generous suggestions. Jay Yagnik, Rahul Sukthankar, Tom Duerig and David Salesin provided enthusiastic support of this project for which we are grateful. We thank Victor Gomes and Erica Moreira for infrastructure support, Jing

Yu Koh and Jason Baldridge for dataset, model and evaluation discussions and feedback on the paper, Mike Krainin for model speedup discussions, JD Velasquez for discussions and insights, Sarah Laszlo, Kathy Meier-Hellstern, and Rachel Stigler for assisting us with the publication process, Andrew Bunner, Jordi Pont-Tuset, and Shai Noy for help on internal demos, David Fleet, Saurabh Saxena, Jiahui Yu, and Jason Baldridge for sharing Imagen and Parti speed metrics.

References

Austin, J., Johnson, D. D., Ho, J., Tarlow, D., and van den Berg, R. Structured denoising diffusion models in discrete state-spaces. *Advances in Neural Information Processing Systems*, 34:17981–17993, 2021.

Brock, A., Donahue, J., and Simonyan, K. Large scale gan training for high fidelity natural image synthesis. *arXiv preprint arXiv:1809.11096*, 2018.

Brooks, T., Holynski, A., and Efros, A. A. Instructpix2pix: Learning to follow image editing instructions. *arXiv preprint arXiv:2211.09800*, 2022.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Buolamwini, J. and Gebru, T. Gender shades: Intersectional accuracy disparities in commercial gender classification. In *Conference on fairness, accountability and transparency*, pp. 77–91. PMLR, 2018.

Chang, H., Zhang, H., Jiang, L., Liu, C., and Freeman, W. T. Maskgit: Masked generative image transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11315–11325, 2022.

CompVis. Stable diffusion colab, 2022. URL https://colab.sandbox.google.com/github/huggingface/notebooks/blob/main/diffusers/stable_diffusion.ipynb#scrollTo=zHkHsdtnry57.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

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

Donahue, J. and Simonyan, K. Large scale adversarial representation learning. *Advances in neural information processing systems*, 32, 2019.

Dulhanty, C. Issues in computer vision data collection: Bias, consent, and label taxonomy. Master's thesis, University of Waterloo, 2020.

Esser, P., Rombach, R., Blattmann, A., and Ommer, B. Imagebart: Bidirectional context with multinomial diffusion for autoregressive image synthesis. *Advances in Neural Information Processing Systems*, 34:3518–3532, 2021a.

Esser, P., Rombach, R., and Ommer, B. Taming transformers for high-resolution image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 12873–12883, 2021b.

Franks, M. A. and Waldman, A. E. Sex, lies, and videotape: Deep fakes and free speech delusions. *Md. L. Rev.*, 78:892, 2018.

Gafni, O., Polyak, A., Ashual, O., Sheynin, S., Parikh, D., and Taigman, Y. Make-a-scene: Scene-based text-to-image generation with human priors, 2022. URL https://arxiv.org/abs/2203.13131.

Gal, R., Alaluf, Y., Atzmon, Y., Patashnik, O., Bermano, A. H., Chechik, G., and Cohen-Or, D. An image is worth one word: Personalizing text-to-image generation using textual inversion. *arXiv preprint arXiv:2208.01618*, 2022a.

Gal, R., Patashnik, O., Maron, H., Bermano, A. H., Chechik, G., and Cohen-Or, D. Stylegan-nada: Clip-guided domain adaptation of image generators. *ACM Transactions on Graphics (TOG)*, 41(4):1–13, 2022b.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

Goyal, P., Dollár, P., Girshick, R. B., Noordhuis, P., Wesolowski, L., Kyrola, A., Tulloch, A., Jia, Y., and He, K. Accurate, large minibatch SGD: Training ImageNet in 1 hour. *preprint arXiv:1706.0267*, 2017.

He, K., Chen, X., Xie, S., Li, Y., Dollár, P., and Girshick, R. Masked autoencoders are scalable vision learners. In *cvpr*, pp. 16000–16009, June 2022.

Hendricks, L. A., Burns, K., Saenko, K., Darrell, T., and Rohrbach, A. Women also snowboard: Overcoming bias in captioning models. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 771–787, 2018.

Hertz, A., Mokady, R., Tenenbaum, J., Aberman, K., Pritch, Y., and Cohen-Or, D. Prompt-to-prompt image editing with cross attention control. *arXiv preprint arXiv:2208.01626*, 2022.

Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., and Hochreiter, S. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Ho, J. and Salimans, T. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598*, 2022.

Ho, J., Jain, A., and Abbeel, P. Denoising diffusion probabilistic models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.

Ho, J., Salimans, T., Gritsenko, A., Chan, W., Norouzi, M., and Fleet, D. J. Video diffusion models. *arXiv preprint arXiv:2204.03458*, 2022.

Hughes, R. T., Zhu, L., and Bednarz, T. Generative adversarial networks–enabled human–artificial intelligence collaborative applications for creative and design industries: A systematic review of current approaches and trends. *Frontiers in artificial intelligence*, 4:604234, 2021.

Jia, C., Yang, Y., Xia, Y., Chen, Y.-T., Parekh, Z., Pham, H., Le, Q., Sung, Y.-H., Li, Z., and Duerig, T. Scaling up visual and vision-language representation learning with noisy text supervision. In *International Conference on Machine Learning*, pp. 4904–4916. PMLR, 2021.

Jouppi, N. P., Yoon, D. H., Kurian, G., Li, S., Patil, N., Laudon, J., Young, C., and Patterson, D. A domain-specific supercomputer for training deep neural networks. *Communications of the ACM*, 63(7):67–78, 2020.

Karras, T., Laine, S., and Aila, T. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 4401–4410, 2019.

Kawar, B., Zada, S., Lang, O., Tov, O., Chang, H., Dekel, T., Mosseri, I., and Irani, M. Imagic: Text-based real image editing with diffusion models. *arXiv preprint arXiv:2210.09276*, 2022.

Kim, G., Kwon, T., and Ye, J. C. Diffusionclip: Text-guided diffusion models for robust image manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2426–2435, 2022.

Kingma, D., Salimans, T., Poole, B., and Ho, J. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. In *ICLR*, 2015.

Lambda Labs. All you need is one gpu: Inference benchmark for stable diffusion, 2022. URL https://lambdalabs.com/blog/inference-benchmark-stable-diffusion.

Lee, D., Kim, C., Kim, S., Cho, M., and Han, W.-S. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 11523–11532, 2022a.

Lee, D., Kim, C., Kim, S., Cho, M., and Han, W.-S. Draft-and-revise: Effective image generation with contextual rq-transformer. *arXiv preprint arXiv:2206.04452*, 2022b.

Lezama, J., Chang, H., Jiang, L., and Essa, I. Improved masked image generation with token-critic. In *European Conference on Computer Vision*, pp. 70–86. Springer, 2022.

Li, T., Chang, H., Mishra, S. K., Zhang, H., Katabi, D., and Krishnan, D. Mage: Masked generative encoder to unify representation learning and image synthesis. *arXiv preprint arXiv:2211.09117*, 2022.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., and Zitnick, C. L. Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer, 2014.

Loshchilov, I. and Hutter, F. SGDR: Stochastic gradient descent with warm restarts. In *iclr*, 2017.

Lu, C., Zhou, Y., Bao, F., Chen, J., Li, C., and Zhu, J. Dpm-solver: A fast ODE solver for diffusion probabilistic model sampling in around 10 steps. *arXiv preprint arXiv:2206.00927*, 2022.

Meng, C., Song, Y., Song, J., Wu, J., Zhu, J.-Y., and Ermon, S. Sdedit: Image synthesis and editing with stochastic differential equations. *arXiv preprint arXiv:2108.01073*, 2021.

Merullo, J., Castricato, L., Eickhoff, C., and Pavlick, E. Linearly mapping from image to text space. *arXiv preprint arXiv:2209.15162*, 2022.

Midjourney. Midjourney, 2022. URL https://www.midjourney.com.

Mokady, R., Hertz, A., Aberman, K., Pritch, Y., and Cohen-Or, D. Null-text inversion for editing real images using guided diffusion models, 2022. URL https://arxiv.org/abs/2211.09794.

NegPrompt. Negative prompt, 2022. URL https://github.com/AUTOMATIC1111/stable-diffusion-webui/wiki/Negative-prompt.

Nichol, A., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., Sutskever, I., and Chen, M. Glide: Towards photorealistic image generation and editing with text-guided diffusion models. *arXiv preprint arXiv:2112.10741*, 2021.

Patashnik, O., Wu, Z., Shechtman, E., Cohen-Or, D., and Lischinski, D. Styleclip: Text-driven manipulation of stylegan imagery. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2085–2094, 2021.

Paullada, A., Raji, I. D., Bender, E. M., Denton, E., and Hanna, A. Data and its (dis) contents: A survey of dataset development and use in machine learning research. *Patterns*, 2(11):100336, 2021.

Prabhu, V. U. and Birhane, A. Large image datasets: A pyrrhic win for computer vision? *arXiv preprint arXiv:2006.16923*, 2020.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pp. 8748–8763. PMLR, 2021.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P. J., et al. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67, 2020.

Ramesh, A., Pavlov, M., Goh, G., Gray, S., Voss, C., Radford, A., Chen, M., and Sutskever, I. Zero-shot text-to-image generation, 2021. URL https://arxiv.org/abs/2102.12092.

Ramesh, A., Dhariwal, P., Nichol, A., Chu, C., and Chen, M. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*, 2022.

Rolfe, J. T. Discrete variational autoencoders. *arXiv preprint arXiv:1609.02200*, 2016.

Rombach, R., Blattmann, A., Lorenz, D., Esser, P., and Ommer, B. 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.

Ruiz, N., Li, Y., Jampani, V., Pritch, Y., Rubinstein, M., and Aberman, K. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. *arXiv preprint arXiv:2208.12242*, 2022.

Saharia, C., Chan, W., Saxena, S., Li, L., Whang, J., Denton, E., Ghasemipour, S. K. S., Ayan, B. K., Mahdavi, S. S., Lopes, R. G., et al. Photorealistic text-to-image diffusion models with deep language understanding. *arXiv preprint arXiv:2205.11487*, 2022.

Salimans, T. and Ho, J. Progressive distillation for fast sampling of diffusion models. In *ICLR*, 2022.

Scheuerman, M. K., Hanna, A., and Denton, E. Do datasets have politics? disciplinary values in computer vision dataset development. *Proceedings of the ACM on Human-Computer Interaction*, 5(CSCW2):1–37, 2021.

Schuhmann, C., Vencu, R., Beaumont, R., Kaczmarczyk, R., Mullis, C., Katta, A., Coombes, T., Jitsev, J., and Komatsuzaki, A. Laion-400m: Open dataset of clip-filtered 400 million image-text pairs. *arXiv preprint arXiv:2111.02114*, 2021.

Schuhmann, C., Beaumont, R., Vencu, R., Gordon, C., Wightman, R., Cherti, M., Coombes, T., Katta, A., Mullis, C., Wortsman, M., et al. Laion-5b: An open large-scale dataset for training next generation image-text models. *arXiv preprint arXiv:2210.08402*, 2022.

Sharma, P., Ding, N., Goodman, S., and Soricut, R. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 2556–2565, 2018.

Shazeer, N. and Stern, M. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pp. 4596–4604. PMLR, 2018.

Srinivasan, R. and Uchino, K. Biases in generative art: A causal look from the lens of art history. In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, pp. 41–51, 2021.

Steed, R. and Caliskan, A. Image representations learned with unsupervised pre-training contain human-like biases. In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pp. 701–713, 2021.

Tao, M., Tang, H., Wu, F., Jing, X.-Y., Bao, B.-K., and Xu, C. Df-gan: A simple and effective baseline for text-to-image synthesis, 2020. URL https://arxiv.org/abs/2008.05865.

Van Den Oord, A., Vinyals, O., et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Wang, S., Saharia, C., Montgomery, C., Pont-Tuset, J., Noy, S., Pellegrini, S., Onoe, Y., Laszlo, S., Fleet, D. J., Soricut, R., Baldridge, J., Norouzi, M., Anderson, P., and Chan, W. Imagen editor and editbench: Advancing and evaluating text-guided image inpainting, 2022. URL https://arxiv.org/abs/2212.06909.

Whittaker, L., Kietzmann, T. C., Kietzmann, J., and Dabirian, A. "all around me are synthetic faces": the mad world of ai-generated media. *IT Professional*, 22(5):90–99, 2020.

Xia, W., Zhang, Y., Yang, Y., Xue, J.-H., Zhou, B., and Yang, M.-H. Gan inversion: A survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022.

Xu, T., Zhang, P., Huang, Q., Zhang, H., Gan, Z., Huang, X., and He, X. Attngan: Fine-grained text to image generation with attentional generative adversarial networks. *CoRR*, abs/1711.10485, 2017. URL http://arxiv.org/abs/1711.10485.

Ye, H., Yang, X., Takac, M., Sunderraman, R., and Ji, S. Improving text-to-image synthesis using contrastive learning, 2021. URL https://arxiv.org/abs/2107.02423.

Yu, J., Li, X., Koh, J. Y., Zhang, H., Pang, R., Qin, J., Ku, A., Xu, Y., Baldridge, J., and Wu, Y. Vector-quantized image modeling with improved vqgan. *arXiv preprint arXiv:2110.04627*, 2021.

Yu, J., Xu, Y., Koh, J. Y., Luong, T., Baid, G., Wang, Z., Vasudevan, V., Ku, A., Yang, Y., Ayan, B. K., et al. Scaling autoregressive models for content-rich text-to-image generation. *arXiv preprint arXiv:2206.10789*, 2022.

Zhang, H., Koh, J. Y., Baldridge, J., Lee, H., and Yang, Y. Cross-modal contrastive learning for text-to-image generation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 833–842, 2021.

Zhao, L., Zhang, Z., Chen, T., Metaxas, D. N., and Zhang, H. Improved transformer for high-resolution gans, 2021. URL https://arxiv.org/abs/2106.07631.

Zhou, Y., Zhang, R., Chen, C., Li, C., Tensmeyer, C., Yu, T., Gu, J., Xu, J., and Sun, T. LAFITE: towards language-free training for text-to-image generation. *CoRR*, abs/2111.13792, 2021. URL https://arxiv.org/abs/2111.13792.

Zhu, M., Pan, P., Chen, W., and Yang, Y. Dm-gan: Dynamic memory generative adversarial networks for text-to-image synthesis. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5802–5810, 2019.

A. Appendix.

A.1. Base Model Configurations

Our base model configuration for our largest model of size 3B parameters is given in Table 4.

Configuration	Value
Number of Transformer layers	48
Transformer Hidden Dimension	2048
Transformer MLP Dimension	8192
Optimizer	AdaFactor (Shazeer & Stern, 2018)
Base learning rate	1e-4
Weight decay	0.045
Optimizer momentum	β_1=0.9, β_2=0.96
Batch size	512
Learning rate schedule	cosine decay (Loshchilov & Hutter, 2017)
Warmup steps	5000
Training steps	1.5M

Table 4. Configuration and training hyperparameters for base model.

A.2. VQGAN Configurations

Configuration	Value
Perceptual loss weight	0.05
Adversarial loss weight	0.1
Codebook size	8192
Optimizer	Adam (Kingma & Ba, 2015)
Discriminator learning rate	1e-4
Generator learning rate	1e-4
Weight decay	1e-4
Optimizer momentum	β_1=0.9, β_2=0.99
Batch size	256
Learning rate schedule	cosine decay (Loshchilov & Hutter, 2017)
Warmup steps (Goyal et al., 2017)	10000
Training steps	1M

Table 5. Configuration and training hyperparameters for VQGAN.

VQGAN Architecture: Our VQGAN architecture is similar to the previous work (Esser et al., 2021b). It consists of several residual blocks, downsample(encoder) and upsample (decoder) blocks. The main difference is that we remove the non-local block to make the encoder and decoder fully convolutional to support different image sizes. In the base VQGAN model, we apply 2 residual blocks in each resolution and the base channel dimension is 128. For the finetuned decoder, we apply 4 residual blocks in each resolution and we also make the base channel dimension to be 256.

Input Image VQGAN Reconstruction Finetuned Decoder

Figure 13. Visual example of the improvement from the fine-tuned decoder (Section 2.5). Please zoom in by at least 200% to see the difference between the VQGAN reconstruction and the reconstruction with a finetuned decoder. We can see especially that fine details such as the house number (bottom left), the storefront sign (middle) and the bars on the windows (right) are better preserved in the finetuned decoder.

A.3. Super Resolution Configurations

Configuration	Value
LowRes Encoder Transformer Layers	16
Number of Transformer layers	32
Transformer Hidden Dimension	1024
Transformer MLP Dimension	4096
Optimizer	AdaFactor (Shazeer & Stern, 2018)
Base learning rate	1e-4
Weight decay	0.045
Optimizer momentum	β_1=0.9, β_2=0.96
Batch size	512
Learning rate schedule	cosine decay (Loshchilov & Hutter, 2017)
Warmup steps	5000
Training steps	1M

Table 6. Configuration and training hyperparameters for the Super-Resolution Model.