Stable Diffusion 3

Scaling Rectified Flow Transformers for High-Resolution Image Synthesis

Patrick Esser * Sumith Kulal Andreas Blattmann Rahim Entezari Jonas Müller Harry Saini Yam Levi
Dominik Lorenz Axel Sauer Frederic Boesel Dustin Podell Tim Dockhorn Zion English
Kyle Lacey Alex Goodwin Yannik Marek Robin Rombach *
Stability AI

Prompt: Epic anime artwork of a wizard atop a mountain at night casting a cosmic spell into the dark sky that says "Stable Diffusion 3" made out of colorful energy

https://stability.ai/news/stable-diffusion-3

https://arxiv.org/pdf/2403.03206.pdf

Stable Diffusion 3

- Released March 5, 2024
- 28 Pages of background, explanation, methods, results—maybe the definitive paper in the field? (assumes you understand probability flows and ODEs as flows)
- Lots of ablation studies on parameter choices
- Evaluated in the right way. I love this paper!

an old rusted robot wearing pants and a jacket riding skis in a supermarket.

smiling cartoon dog sits at a table, coffee mug on hand, as a room goes up in flames. "This is fine," the dog assures himself.

Vector Gradient Flow Scalers

- Define Loss as a vector field, after lots of computations:

$$\mathcal{L}_w(x_0) = -\frac{1}{2}\mathbb{E}_{t\sim\mathcal{U}(t),\epsilon\sim\mathcal{N}(0,I)}\left[w_t\lambda_t'\|\epsilon_\Theta(z_t,t)-\epsilon\|^2\right]$$

x0 is latent of original image

sampling at time step t

with noise added to image latent

~Predicted Noise at time t

Actual Noise

defines SNR and different scaling factors, depends on how noise added and time steps

- In paper, they compare lots of different scaling variations with different noise models:

Rectified flow: $z_t = (1-t)x_0 + t\epsilon \qquad w_t^{\mathrm{RF}} = \frac{t}{1-t}$

EDM: $z_t = x_0 + b_t\epsilon \qquad b_t = \exp F_{\mathcal{N}}^{-1}(t|P_m, P_s^2) \quad w_t^{\mathrm{EDM}} = \mathcal{N}(\lambda_t|-2P_m,(2P_s)^2)(e^{-\lambda_t}+0.5^2)$

Cosine: $z_t = \cos(\frac{\pi}{2}t)x_0 + \sin(\frac{\pi}{2}t)\epsilon \qquad w_t = e^{-\lambda_t/2}$

LDM Linear: $z_t = a_t x_0 + b_t\epsilon \qquad b_t = \sqrt{1-a_t^2}, \quad a_t = (\prod_{s=0}^{t}(1-\beta_s))^{\frac{1}{2}} \quad \beta_t = \left(\sqrt{\beta_0} + \frac{t}{T-1}(\sqrt{\beta_{T-1}} - \sqrt{\beta_0})\right)^2$

Sampling t

- In paper, look at lots of ways to sample the t across the various distributions:

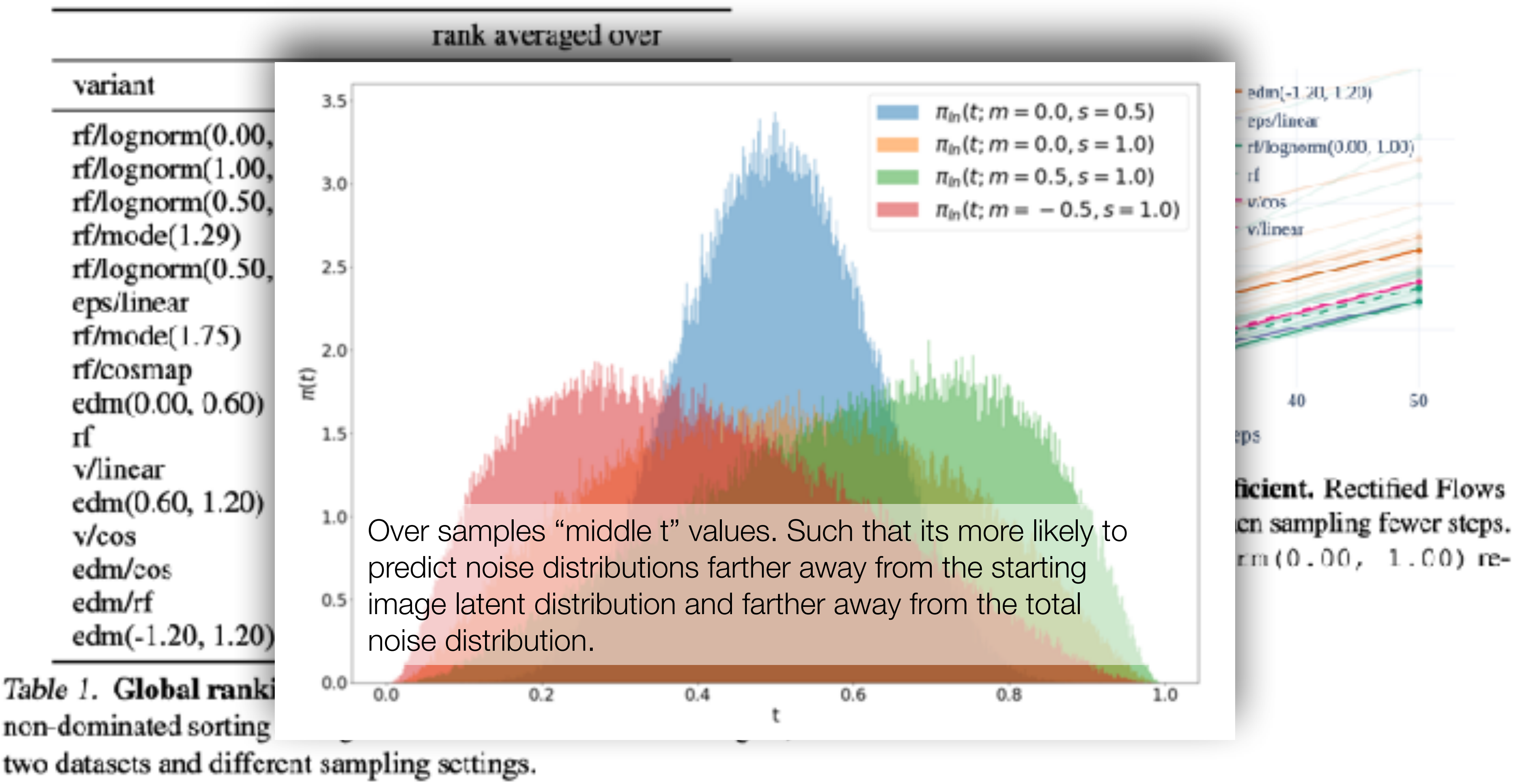
Uniform Distribution: $\mathcal{U}(t)$

Logit-normal: $$\pi_{\ln}(t; m, s) = \frac{1}{s\sqrt{2\pi}} \frac{1}{t(1-t)} \exp\left(-\frac{(\operatorname{logit}(t) - m)^2}{2s^2}\right),$$

Heavy Tailed Mode: $$f_{\text{mode}}(u; s) = 1 - u - s \cdot \left(\cos^2\left(\frac{\pi}{2}u\right) - 1 + u\right)$$

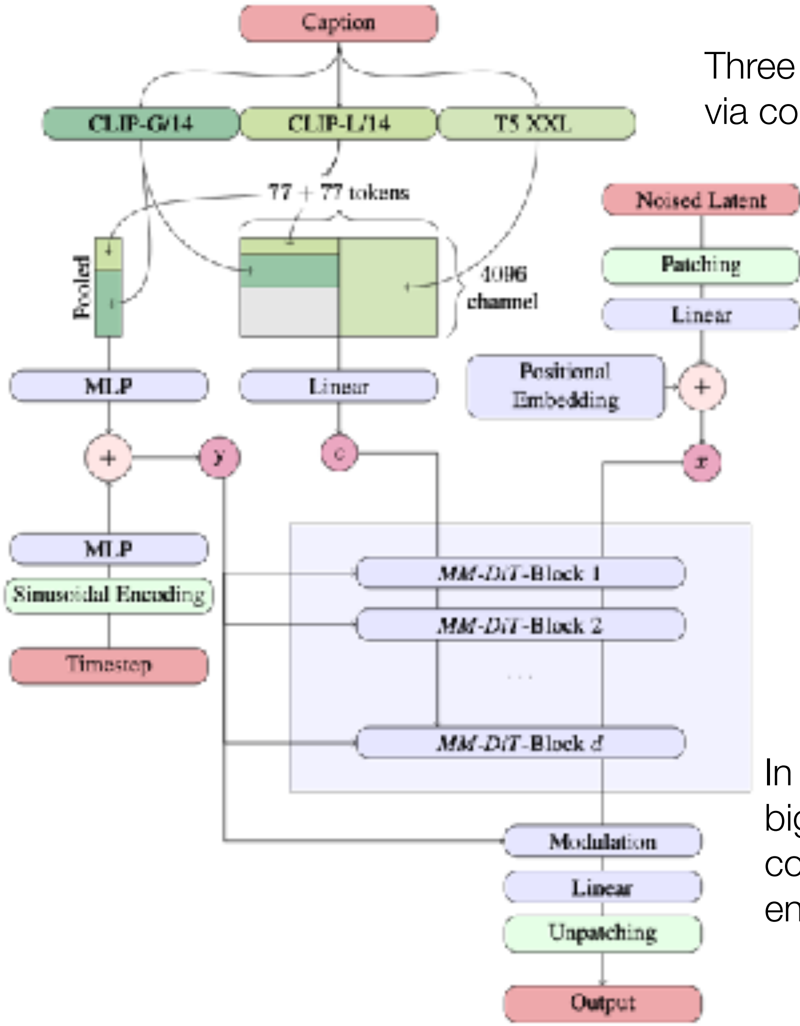
CosMap: $$\pi_{\text{CosMap}}(t) = \left|\frac{d}{dt} f^{-1}(t)\right| = \frac{2}{\pi - 2\pi t + 2\pi t^2}.$$

Ablation for Variant and Sampling



Rectified flow is always one of the better performers, especially using logit-normal sampling m=0, s=1

The Architecture: Overview



(a) Overview of all components.

Three Conditioning Language and Image encoders used, via concatenation

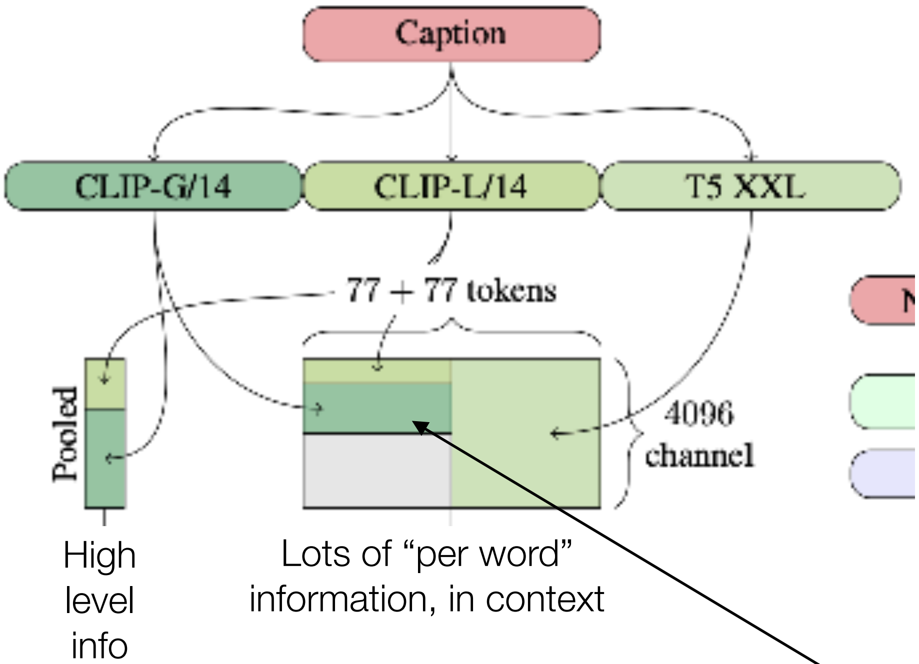
Input Modality Dropout used to ensure "good" results on any of the encoders at deployment time Drop out T5, or Dropout CLIP, etc.

Separate text modality and image modality before feeding into the noise prediction network.

Each MM-DiT is just a transformer working on the concatenated modalities

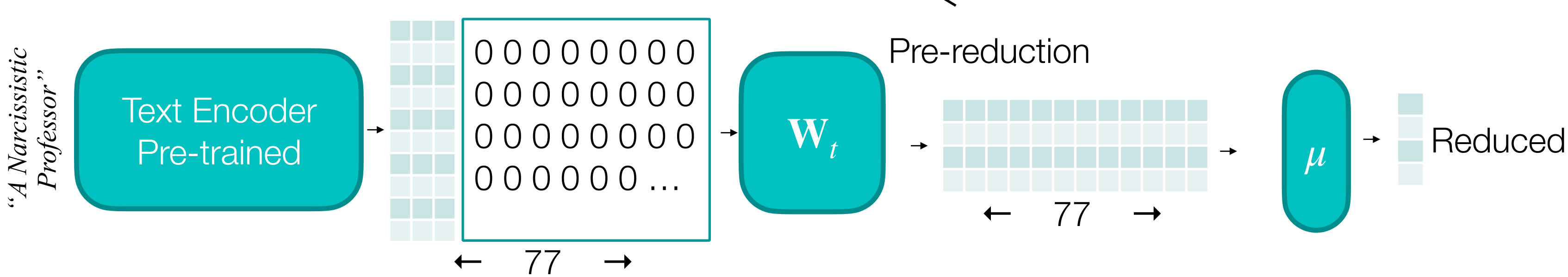
In paper, investigated many "depth scaling" techniques, found bigger is always better. An that there was no saturation, so could probably get even better results, but the GPUs are not big enough…

The Architecture: Text Input

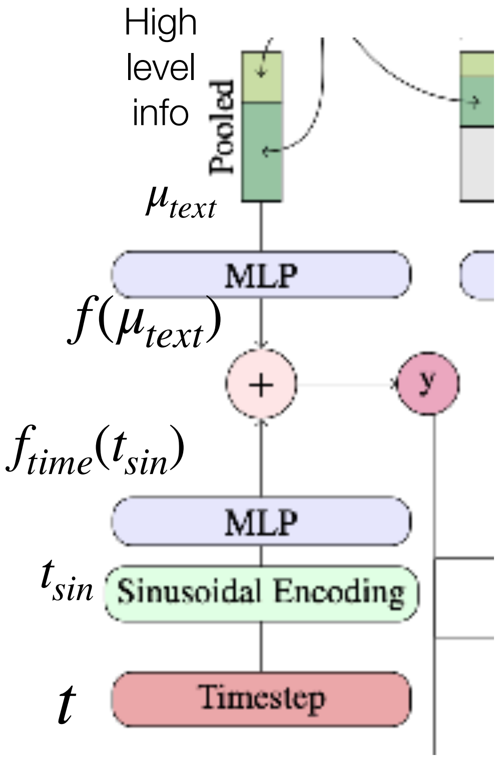


- Max caption length 77
- Feed pre-reduced CLIP embeddings
- Combine with T5 encoder for each token
- Add "modality dropout" to reduce dependence on each text encoder

CLIP



The Architecture: Time info

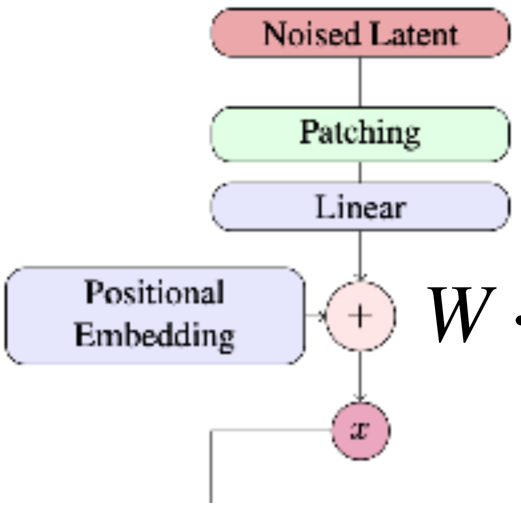


- Want this "de-noise" architecture to have information of how much noise: $z_t = (1-t) \cdot x_1 + t \cdot \epsilon$
- So tell it the value of t via sinusoidal position encoding of the same size as pooled text information
- Combine both through "addition" and will use throughout the network

$$y = f(\mu_{text}) + f_{time}(t_{sin})$$

y now encodes lots of information about the text we want and the amount of noise.

The Architecture: Noised Latent

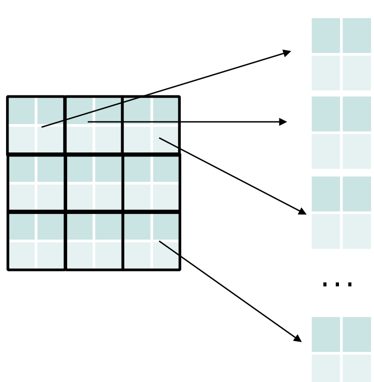


z_t

(z_{patch})

$W \cdot z_{patch} + b$

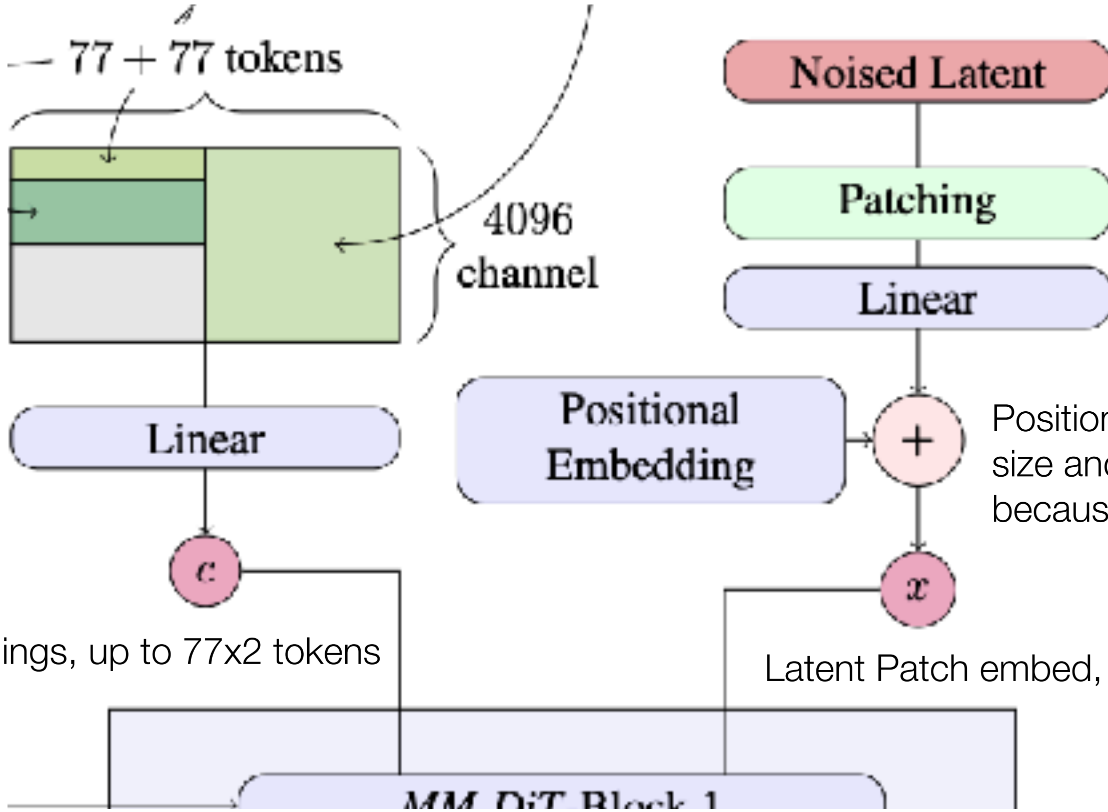
$$x = W \cdot z_{patch} + b + P_{embed}$$



z_t $(z_{patch})_1$ $(z_{patch})_N$

- $z_t = (1 - t) \cdot x_1 + t \cdot \epsilon$
- This is a 3D latent tensor, Shown here as 2D, but there are channels in each patch with rich features
- Want to process this like a regular image in a ViT
- So we break into patches spatially
- Flatten each patch for use in X-former

The Architecture: Multi-modal inputs



Position encoding controls the size and aspect ratio of image because its 2x2

Text embeddings, up to 77x2 tokens

Latent Patch embed, up to N=77x2

Each "latent patch" is 2x2xc and represents an encoding of a portion of the image. These patches can represent a large number of pixels.

Architecture: X-former block

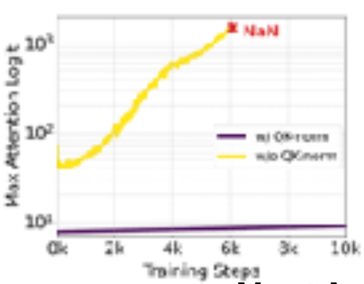
Other Things:
Do not use only human generated captions. Humans tend to not describe things like background, colors, etc.

Solution: Use trained models for generating captions of high quality. Then use a mix of "human captions" and "augmented captions" while training.

Make Encoder/Decoder large dimensionality.

Use as many x-formers as possible.

Normalize the QK attention matrix



time step encoding

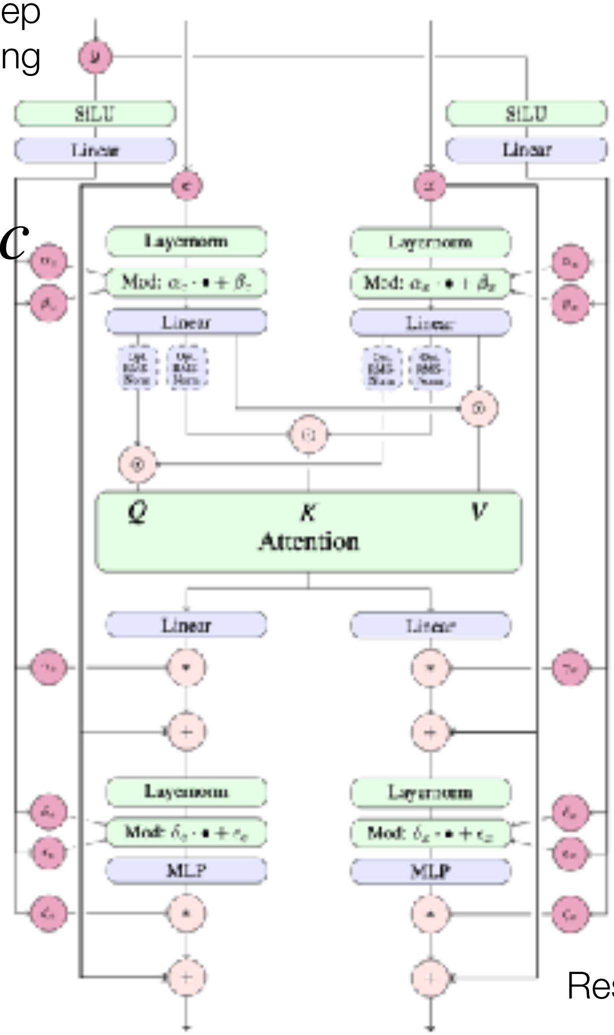
text embed, 77x2

latent image embed, 77x2

Previous Layer

$\hat{y}_c$

$\hat{y}_x$



Shift the feature distribution using the time/text embed
$= \alpha_x(\hat{y}) \cdot x_{patch}^{norm} + \beta_x(\hat{y})$

Concatenate each modality and run in standard x-former (multi-headed)

Branch into two MLPs to separate out into two modalities

These feed into next layer

Residual Connections

Next Layer: maybe text features maybe latent image features

Once trained, how to get images?

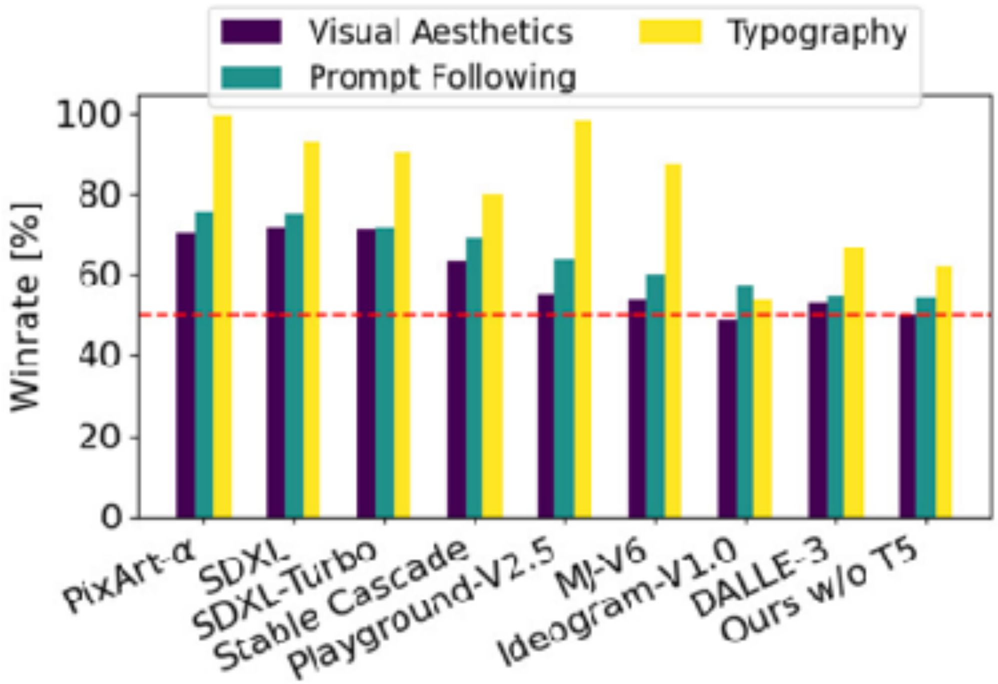
- We start with the base noise Probability, t=1 in
  $z_t = (1 - t) \cdot x_1 + t \cdot \epsilon$
- The network has a "t" awareness branch for predicting noise. After the first step, how do we update "t"?
- Should we update "t" for different resolutions?
- **Solution**: try different step sizes and see what people prefer:

Least Preferred:

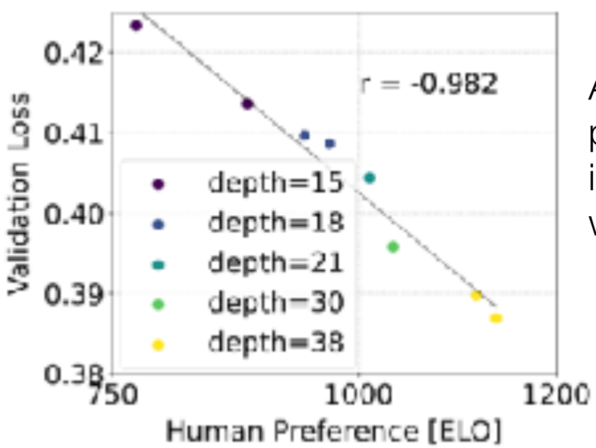
Most Preferred:

Evaluation: compare to state of the art

- Conducted large scale human subjects rating study
- Generate images from top models from same prompt
- Ask humans which version they prefer.
- What is the probability of their model winning?



Asked preference based on:
1. Simple Aesthetics (looks)
2. Following of the prompt
3. Accuracy of text and typography



Also: Human performance is correlated with loss!!

A whimsical and creative image depicting a hybrid creature that is a mix of a waffle and a hippopotamus. This imaginative creature features the distinctive, bulky body of a hippo, but with a texture and appearance resembling a golden-brown, crispy waffle. The creature might have elements like waffle squares across its skin and a syrup-like sheen. It's set in a surreal environment that playfully combines a natural water habitat of a hippo with elements of a breakfast table setting, possibly including oversized utensils or plates in the background. The image should evoke a sense of playful absurdity and culinary fantasy.

All text-encoders	*w/o T5 (Raffel et al., 2019)*

"A burger patty, with the bottom bun and lettuce and tomatoes. "COFFEE" written on it in mustard"

"A monkey holding a sign reading "Scaling transformer models is awesome!"

"A mischievous ferret with a playful grin squeezes itself into a large glass jar, surrounded by colorful candy. The jar sits on a wooden table in a cozy kitchen, and warm sunlight filters through a nearby window"

Detailed pen and ink drawing of a happy pig butcher selling meat in its shop.

a massive alien space ship that is shaped like a pretzel.

A kangaroo holding a beer, wearing ski goggles and passionately singing silly songs.

An entire universe inside a bottle sitting on the shelf at walmart on sale.

A cheeseburger surfing the vibe wave at night

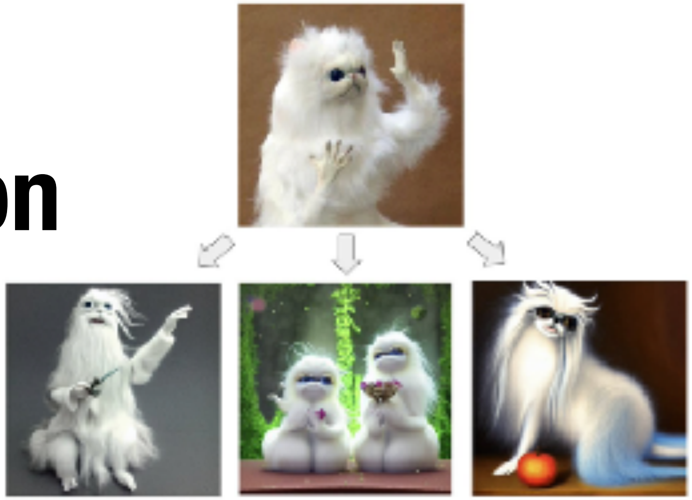
A swamp ogre with a pearl earring by Johannes Vermeer

A car made out of vegetables.

heat death of the universe, line art

if time

Stable Diffusion



High-performance image generation using Stable Diffusion in KerasCV

Authors: fchollet, lukewood, divamgupta
Date created: 2022/09/25
Last modified: 2022/09/25
Description: Generate new images using KerasCV's StableDiffusion model.

View in Colab • GitHub source

Sponsor Follow

LukeWood Luke Wood

KerasCV Author, Full Time Keras team member & Machine Learning researcher @ Google, Part Time UCSD Ph.D student

PRO

Student Paper Presentation

DreamFusion: Text-to-3D using 2D Diffusion

Ben Poole, Ajay Jain, Jonathan T. Barron, Ben Mildenhall

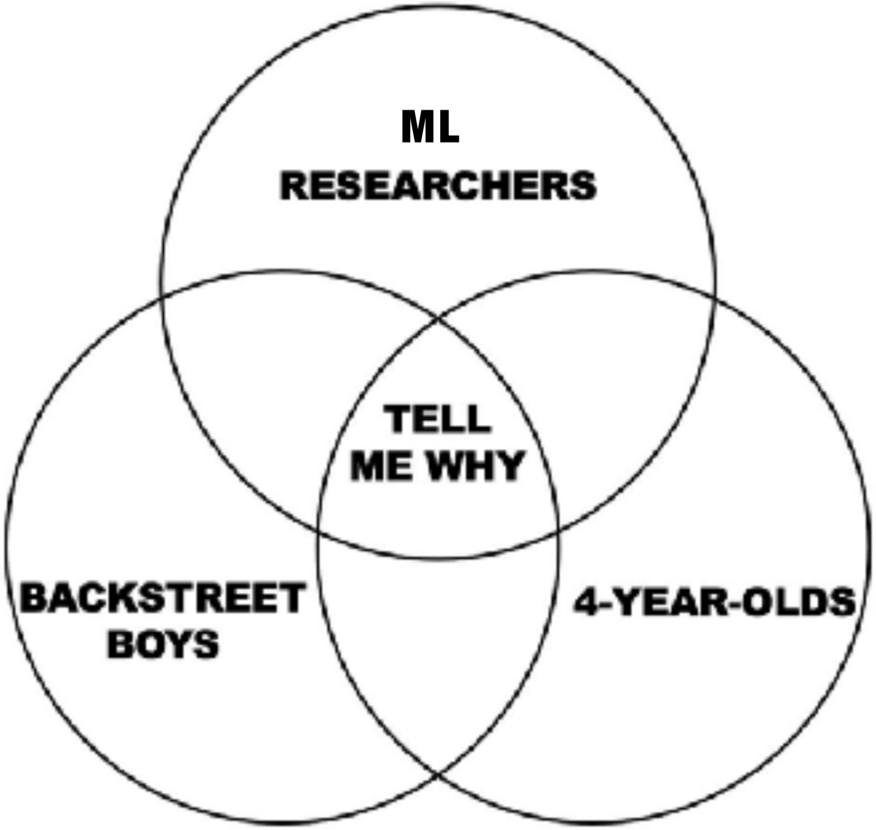
Published: 01 Feb 2023, Last Modified: 11 Mar 2024 ICLR 2023 notable top 5% Readers: Everyone Show Bibtex Show Revisions

Keywords: diffusion models, score-based generative models, NeRF, neural rendering, 3d synthesis

TL;DR: DeepDream on a pretrained 2D diffusion model enables text-to-3D synthesis

Abstract: Recent breakthroughs in text-to-image synthesis have been driven by diffusion models trained on billions of image-text pairs. Adapting this approach to 3D synthesis would require large-scale datasets of labeled 3D or multiview data and efficient architectures for denoising 3D data, neither of which currently exist. In this work, we circumvent these limitations by using a pretrained 2D text-to-image diffusion model to perform text-to-3D synthesis. We introduce a loss based on probability density distillation that enables the use of a 2D diffusion model as a prior for optimization of a parametric image generator. Using this loss in a DeepDream-like procedure, we optimize a randomly-initialized 3D model (a Neural Radiance Field, or NeRF) via gradient descent such that its 2D renderings from random angles achieve a low loss. The resulting 3D model of the given text can be viewed from any angle, relit by arbitrary illumination, or composited into any 3D environment. Our approach requires no 3D training data and no modifications to the image diffusion model, demonstrating the effectiveness of pretrained image diffusion models as priors.

Final Project Draft Town Hall



Lecture Notes for

Neural Networks and Machine Learning

Stable Diffusion

Next Time:
Reinforcement Learning

Reading: None