(2/4
Generator data
Discriminatory data
Over time it will always succeed. The generating images can fool a person.

You use a neural network to distinguish the two for the past few years.

VQ-GAN.

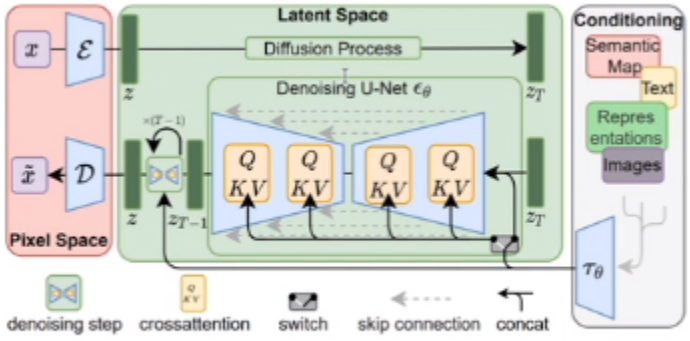


Figure 3. We condition LDMs either via concatenation or by a more general cross-attention mechanism. See Sec. 3.3

Vector quantification. Latent diffusion.
Instead of a big distribution. We are not trying to learn how to model the distribution accurately.
Gradient - change between each of the images. With respect to the variable input data. Change - how to calculate the change.

They did it by a neuronetwork.

Stats differential equation - where they want to reverse that noise.

Math to formalize and do it for images instead of the toy example.
Generalize bigger images.

Here - shown that the formulation in paper by openAI

Diffusion models can beat GANS.

The way diffusion is done in - Core - the way diffusion is done is through apprixmiate with a U.Net.

U.Net is learning about data at a pixel level. Segment. U.Net most common method. Labeling each pixel. Benefits.

U.Net consist of convolution. Pairs down the input. And deconvolution. And builds the image backup. Connections between each stage. And each stage of the deconvolution.

They use the U.Net on the loss.
Formulated: The difference between noise added.

U.Net denoising.
Trivial if you know which part of the image - denoise - subtract it from () Subtract the output of the u.net.

Z and X conversion. (diffusion process)
Trains a u.net to determine the difference.

Hierarchical - refers to spatial hierarchy
Convolution on a certain path - features at the resolution that is max image scale. Latent space.

Denoising. At the core of the model.

Latent space: describe input pixel space and latent space.

GANS - vector politician works.

You have an encoder. Encode an image. Using a series of convolutions.
- Converts it to a series of vectors.
  - Snaps vectors into code vault.
  - Snap into a single point in space.
- Combination: Predetermined outputs. 502 points it snaps to.

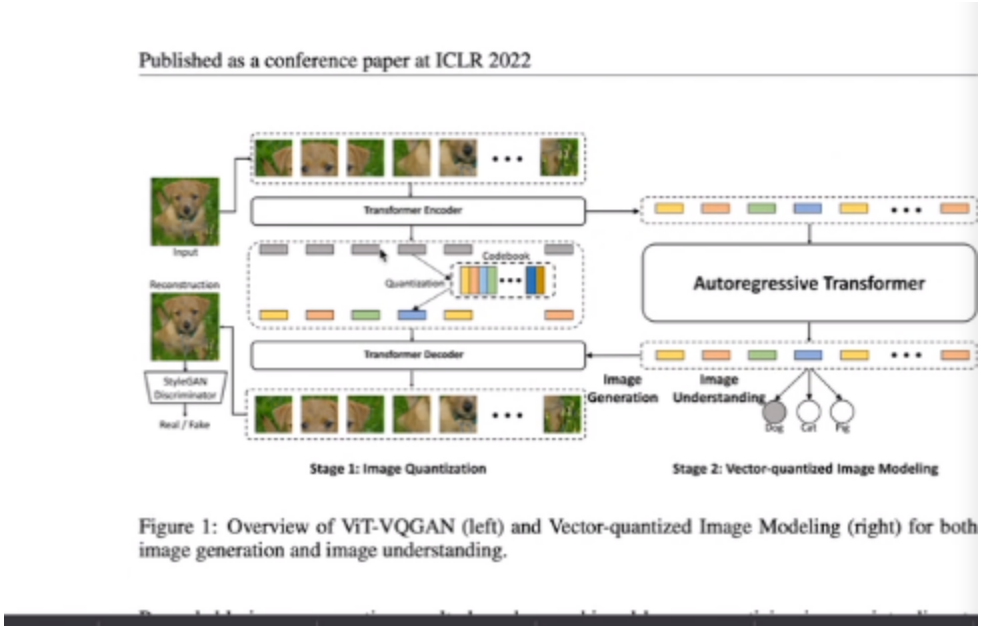


Figure 1: Overview of ViT-VQGAN (left) and Vector-quantized Image Modeling (right) for both image generation and image understanding.

Latent space: The point:
Couple Reasons:

1. Encoding: images are superfollus data (way)

2. J.peg. Already figured out how to compress an image. While preserving the image itself. Perceptual compression. It compresses the space without losing the image.

3. Conversion to latent space - semantic compression. You start losing details - high level details of the image but preserve the idea. Press the image of a dog - corgi - by saying that you remove a lot of details in the image itself. Compressing - Turning that into a denser form of the image. Images taking the meaning make it less information dense.

Motivating compression: to make things faster. That's why we squish images. Un.et is a full neuronetwork. Not computational. Run several times. U.net is run several times in a roll in a series/parallel. Making it as image dense, save as much compute time as possible.

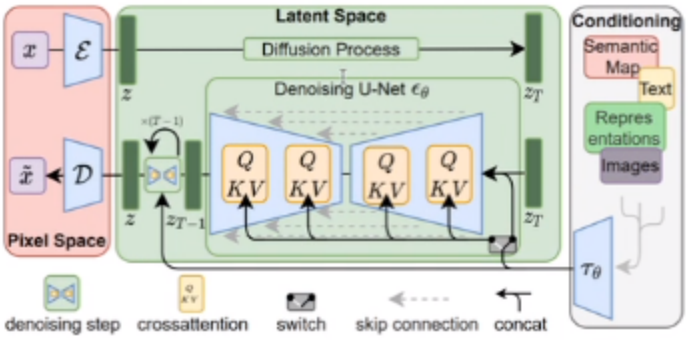


Figure 3. We condition LDMs either via concatenation or by a more general cross-attention mechanism. See Sec. 3.3

Say that we are doing latent space and compression. Why are we using this U.net in the first place. Take the images as separate vectors perform autoregression and the series of vectors corresponds to an idea. Why not treat each vector in the latent space. Big network and regression and auto-regression. Treat each vector separately. It's a lot more generalizable. Exact word was: **Inductive Bias.**

In the end, we are working with spatial features.
Spaces that are related to each other.
Details : table needs to legs.
You need a head attached to a body.
Core features need to be kept.

Keeping and doing analysis and doing training models in that space: spatial area. Latent version is a lot more accurate. Vectors and fitting a model to a pool of vectors. Latent space how do you decode it? One way to do it is through vector quantification. Output of all the denoising and fits it into a certain set of vectors. Then generates an image.

Using divergence (*KALE DIVERGENCE?????) - modeling the accuracy of the image. Difference in the total generator. Oddly enough - after all the denoising - dealing with the inverse of a distribution is still a distribution this output is not a fixed value.

Denoise could be a distribution of value depending on the noise.
Sample - distribution and choose a vector from the distribution to decode.

Vector quantification. Or through (IS IT KALE OR KL????)
Distance between two points. Or two blobs. Or two clouds. The mathematical way to figure out the distribution. You can't just choose two points (two unlikely events in two different distrbutions_) It's a measure of getting the overall likelihood or distance. It's the expectation of when you have a certain input distribution. And another distribution yourself. What is the difference between distributions? Use Kale (IS IT KL OR KALE???) convergence.

Metric for distributions.
So that one isn't the other.

Final thing: Over interesting curveball - next question. Reconstruct images from Noise. Generate accurate looking images of dogs. You could randomly train images of dogs well. How do I specify what is it I want from this diffusion model?

We conversed how to reconstruct an image.
How to get a certain set of structure
How to recover
How to put it in latent space to make it faster.
How do you do this caption?

Stable diffusion _ how do you factor in the caption?

How do you put in your crude image? To make it a more complicated image? You do it by **unconditioning mechanism.**

Another giant paper: "**Attention is all you Need"**

Attention for use in models. QKEV - each of these are attention.

Attention: Mathematical - you multiply the two layers

Idea: you have a high level query: three different inputs to a transformer.

K, Q, and V (value) .

High level - it takes "Q" and maps it over in "K"s.
Closest matching result in V.

Matches the similarity between the Q and K. And it outputs that. And it maps it.
The Q: Output of special encoder
Captions: They use CLIP.
Encoder uses this model they condition it on. It turns it into a Q input.
K and V.
Q is the input of noise input. (latent space)
K and V is the encoded condition.
Take the input as input. Noise model. And outputs the mapping of the noise model. Onto the encoding of the conditioning input. Mapped onto the space of the conditioning input.

Each convolution: Cross Attention

Text encoding.
Image encoding
They are different spaces. Cross Attention is attention between different spaces.

This is not part of the original U.Net.

U.Net is just the convolutions itself.
Added to U.net to take into account the conditioning.

K and V aren't directly the encoder.
Defined by the W.
Its a linear transformer matrix transformer.

Q is not the exact output.

It's a conditioning that goes into a linear transformer first.
The attention model maps it to the best space.

K and V areas.

U.Net is a sandwich between the U.net.

Apply it to the intermediate representation. Do it in parallel. Cross attention output of U.net and flatten it into a vector and use it as a cross attention.

Plans:

Type of data: concatenates to the cross attention.
Augmenting the U.net backbone: Modalities.

Generative power of conditioning _ underscore area.

If the condition is an image: attach it to noise input and run it through U.net.
Not of the image space. Text encoding is not the same space.

Cartoony picture. Scroll down to boxes with short prompts in it or classes.

The idea: draw out something to be a line of trees, grass. Train.

Figure 8. Layout-to-image synthesis with an *LDM* on COCO [4], see Sec. 4.3.1. Quantitative evaluation in the supplement D.3.

4.3. Conditional Latent Diffusion

4.3.1 Transformer Encoders for LDMs

By introducing cross-attention based conditioning into LDMs we open them up for various conditioning modali-

Method	FID↓	IS↑	Precision↑	Recall↑	N_{params}	
BigGan-deep [3]	6.95	203.6$_{\pm 2.6}$	**0.87**	0.28	340M	-
ADM [15]	10.94	100.98	0.69	**0.63**	554M	250 DDIM steps
ADM-G [15]	4.59	186.7	0.82	0.52	608M	250 DDIM steps
LDM-4 (ours)	10.56	103.49$_{\pm 1.24}$	0.71	0.62	400M	250 DDIM steps
LDM-4-G (ours)	**3.60**	**247.67**$_{\pm 5.59}$	**0.87**	0.48	400M	250 steps, *c.f.g* [32], $s = 1.5$

Table 3. Comparison of a class-conditional ImageNet *LDM* with recent state-of-the-art methods for class-conditional image generation on ImageNet [12]. A more detailed comparison with additional baselines can be found in D.4, Tab. 10 and F. *c.f.g.* denotes classifier-free guidance with a scale s as proposed in [32].

purpose image-to-image translation models. We use this to train models for semantic synthesis, super-resolution (Sec. 4.4) and inpainting (Sec. 4.5). For semantic synthesis, we use images of landscapes paired with semantic maps [23, 61] and concatenate downsampled versions of the semantic maps with the latent image representation of a $f = 4$ model (VQ-reg., see Tab. 8). We train on an input resolution of 256^2 (crops from 384^2) but find that our model generalizes to larger resolutions and can generate images up to the

Works in a similar way to the cartoony image (cartoon-y images are called classes single token). Simple image. Further. Semantic mapping - specify a class.

Converts - photoshop people out of its stable diffusion.
Null class.

LDM used to decompress images.
In section D-3 details about how that semantic map is implemented.

Quantitative and train a model

Pg. 26 - Transformer Block - U.net Architecture. Attention mechanism works.

Just above that images: embedding layers. 512B vectors. Class condition stuff. Same trek of embedding program that classes - interpreted.

Table 16. Architecture of a transformer block as described in Sec. E.2.1, replacing the self-attention layer of the standard "ablated UNet" architecture [15]. Here, n_h denotes the number of attention heads and d the dimensionality per head.

	Text-to-Image	Layout-to-Image
seq-length	77	92
depth N	32	16
dim	1280	512

Table 17. Hyperparameters for the experiments with transformer encoders in Sec. 4.3.

E.2.2 Inpainting

For our experiments on image-inpainting in Sec. 4.5, we used the code of [88] to generate synthetic masks. We use a fixed set of 2k validation and 30k testing samples from Places [108]. During training, we use random crops of size 256 × 256 and evaluate on crops of size 512 × 512. This follows the training and testing protocol in [88] and reproduces their reported metrics (see † in Tab. 7). We include additional qualitative results of *LDM-4, w/ attn* in Fig. 21 and of *LDM-4, w/o attn, big, w/ ft* in Fig. 22.

E.3. Evaluation Details

This section provides additional details on evaluation for the experiments shown in Sec. 4.

E.3.1 Quantitative Results in Unconditional and Class-Conditional Image Synthesis

We follow common practice and estimate the statistics for calculating the FID, Precision, and Recall scores [29, 50] shown in

Test accuracy: generalizability: increasing the image size.
Self-attention -

Z-shaped layout image:

Depth	2	2	2	2	2	2
Channel Multiplier	1,1,2,2,4,4	1,2,2,4,4	1,2,3,4	1,2,4	1,2,4	1,2,4
Attention resolutions	32, 16, 8	32, 16, 8	32, 16, 8	32, 16, 8	16, 8, 4	8, 4, 2
Head Channels	32	32	32	32	32	32
Batch Size	9	11	48	96	128	128
Iterations*	500k	500k	500k	500k	500k	500k
Learning Rate	9e-5	1.1e-4	9.6e-5	9.6e-5	1.3e-4	1.3e-4

Table 14. Hyperparameters for the unconditional *LDMs* trained on the CelebA dataset for the analysis in Fig. 7. All models trained on a single NVIDIA A100. *: All models are trained for 500k iterations. If converging earlier, we used the best checkpoint for assessing the provided FID scores.

Task	Text-to-Image	Layout-to-Image		Class-Label-to-Image	Super Resolution	Inpainting	Semantic-Map-to-Image
Dataset	LAION	OpenImages	COCO	ImageNet	ImageNet	Places	Landscapes
f	8	4	8	4	4	4	8
z-shape	32 × 32 × 4	64 × 64 × 3	32 × 32 × 4	64 × 64 × 3	64 × 64 × 3	64 × 64 × 3	32 × 32 × 4
$\|\mathcal{Z}\|$	-	8192	16384	8192	8192	8192	16384
Diffusion steps	1000	1000	1000	1000	1000	1000	1000
Noise Schedule	linear	linear	linear	linear	linear	linear	linear
Model Size	1.45B	306M	345M	395M	169M	215M	215M
Channels	320	128	192	192	160	128	128
Depth	2	2	2	2	2	2	2
Channel Multiplier	1,2,4,4	1,2,3,4	1,2,4	1,2,3,5	1,2,2,4	1,4,8	1,4,8
Number of Heads	8	1	1	1	1	1	1
Dropout	-	-	0.1	-	-	-	-
Batch Size	680	24	48	1200	64	128	48
Iterations	390K	4.4M	170K	178K	860K	360K	360K
Learning Rate	1.0e-4	4.8e-5	4.8e-5	1.0e-4	6.4e-5	1.0e-6	4.8e-5
Conditioning	CA	CA	CA	CA	concat	concat	concat
(C)A-resolutions	32, 16, 8	32, 16, 8	32, 16, 8	32, 16, 8	-	-	-
Embedding Dimension	1280	512	512	512	-	-	-
Transformer Depth	1	3	2	1	-	-	-

To work on: Implementation of T for conditional LDMs.
Hyper-parameters - this is the current shape "Z"
Synthesis of a lot of different stuff.

What “F” is defined as: .

OpenAI looks at diffusion for some time.
Before they came across stable diffusion.
“Attention is all you need” is also OpenAI

The paper is in-house work.

OpenAI generative model is made up of: ?????
Nominal contributions for latent space. Lower dimensional vector.

It is something of latent space.
Diffusion: is already there. (that is the assumption).

Expectation:

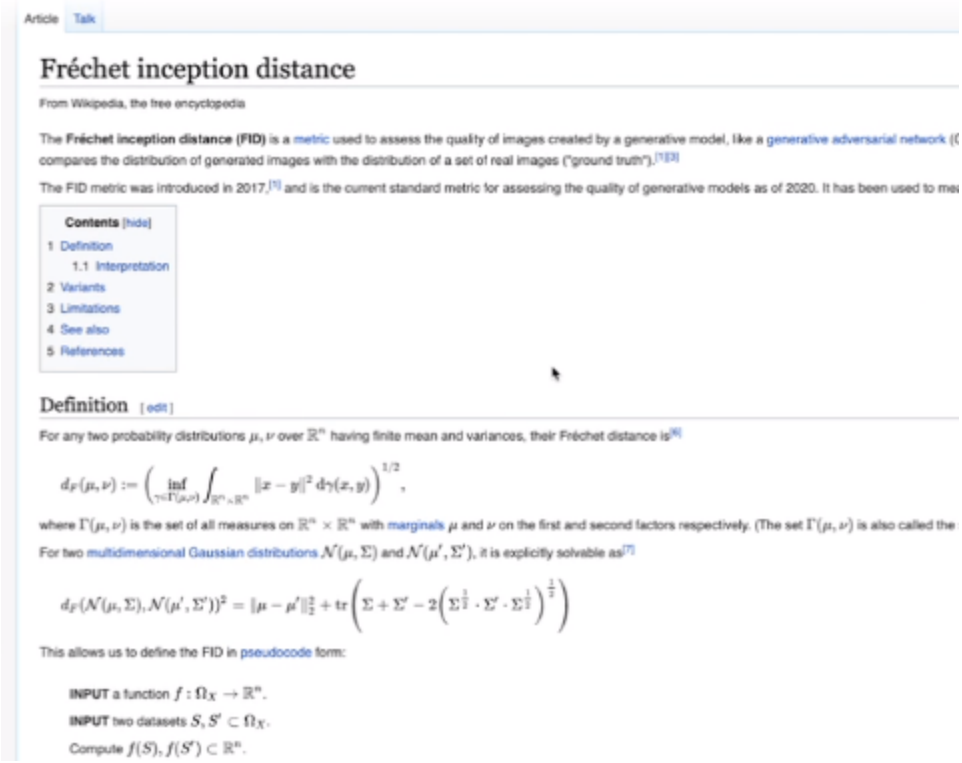
$$L_{DM} = \mathbb{E}_{x,\epsilon\sim\mathcal{N}(0,1),t}\left[\|\epsilon - \epsilon_\theta(x_t, t)\|_2^2\right], \qquad (1)$$

Loss: Objective the expectation overall what is the average value of all expected outcomes? What is the random noise here? All of the outcomes that comes from this additive noise. Noise value that is approximated as a standard gaussian distribution stage T. The difference between L-2 Norm and Calculated noise from U-Net. Expanded. This is not working with image space and not latent space. Its Z and not X. They add the conditioning here and use output to say this is the output of the conditioning.

Data-sets used for image recreation.

FID - KALE DIVERGENCE????

FID - Inception distance - Frechet Inception Distance
Accuracy measure for generative models.



Article Talk

Fréchet inception distance

From Wikipedia, the free encyclopedia

The **Fréchet inception distance (FID)** is a metric used to assess the quality of images created by a generative model, like a generative adversarial network (G
compares the distribution of generated images with the distribution of a set of real images ("ground truth").[1][2][3]

The FID metric was introduced in 2017,[1] and is the current standard metric for assessing the quality of generative models as of 2020. It has been used to me

Contents [hide]

1 Definition
1.1 Interpretation
2 Variants
3 Limitations
4 See also
5 References

Definition [edit]

For any two probability distributions μ, ν over $\mathbb{R}^n$ having finite mean and variances, their Fréchet distance is[4]

$$d_F(\mu,\nu) := \left(\inf_{\gamma\in\Gamma(\mu,\nu)}\int_{\mathbb{R}^n\times\mathbb{R}^n}\|x-y\|^2\,\mathrm{d}\gamma(x,y)\right)^{1/2},$$

where $\Gamma(\mu,\nu)$ is the set of all measures on $\mathbb{R}^n \times \mathbb{R}^n$ with marginals μ and ν on the first and second factors respectively. (The set $\Gamma(\mu,\nu)$ is also called the

For two multidimensional Gaussian distributions $\mathcal{N}(\mu,\Sigma)$ and $\mathcal{N}(\mu',\Sigma')$, it is explicitly solvable as[7]

$$d_F(\mathcal{N}(\mu,\Sigma),\mathcal{N}(\mu',\Sigma'))^2 = \|\mu-\mu'\|_2^2 + \mathrm{tr}\left(\Sigma+\Sigma'-2\left(\Sigma^{\frac{1}{2}}\cdot\Sigma'\cdot\Sigma^{\frac{1}{2}}\right)^{\frac{1}{2}}\right)$$

This allows us to define the FID in pseudocode form:

INPUT a function $f : \Omega_X \to \mathbb{R}^n$.
INPUT two datasets $S, S' \subset \Omega_X$.
Compute $f(S), f(S') \subset \mathbb{R}^n$.

Input distribution and output distribution of images.
Then you determine the distance between those.

You will find pseudocode in wikipedia.

The difference between the two distributions.
Taking the representation of these images. Comparison.
Convolving it down. Semantic difference between the two.
Distribution with a set of (reel?)

Inception score: only determines quality of images. FID is trying to compare the images to ground truth. It is how most people **evaluate generative models. Recently.**

F is the inception v3 model.
Images put through the model then compare it between the two.

Then compare the distribution - determine the images - unsupervised losses. For accuracy. Nothing to compare it against. Compare it against distribution as a whole. Autoencoders. Distribution of entire set of generative images_compare it and output distribution images. And take the real images: put it through inception. To put it through encoded vectors. Then compare the distance between the two distributions. Standard deviation of that layer. The distance between two points (normal distribution is defined by standard deviation).

Inception V3 network.
You don't want to use pixel level comparison. Getting the exact image. The semantic of the image correct (its a dog!) the thing generated is a dog. To do that - you do it to a network. Pairs down the image into a latent vector. Classification on that latent vector. You take a set of images and put it through a set of inception v3. ANd compare the latent space. Inception v3 is the encoder.

Fancy convolutional neuro-net.
Res-net. Computationally extremely expensive.

Different variants (res net) inception uses digital layers with its effectors.
They say because neural networks and object classification is proven by better networks is almost human level by a certain definition. They say its human level because of object classification 95% accuracy. Load to mid 90s. Inception. The conclusion: more or less getting classification and understanding an object. Models for understanding is a solved problem. Bootstrapping by using a model to determine the accuracy of a model. Using something (predictive model) to determine another predictive model. They are so good that they have some ground truth to what they do.

Lower computation cost but not talk about the semantic cost of the images?

Was there anything special about the latent space aside from it being cheaper?
Generalizability? Doing U.net on a latent space. Images inductive biases U.net. More suitable for likelihood base generative models. Its more information dense. You are better off using the latent space for doing probabilistic and diffusion models.

Metrics are image quality.

The metric (KALE DIVERGENCE?)

Better than some other network produces - produces test data set. (celebrity face, church beds) generate images and compare it.

Look at figure 5 (because its according to Toby)

They can't spell.

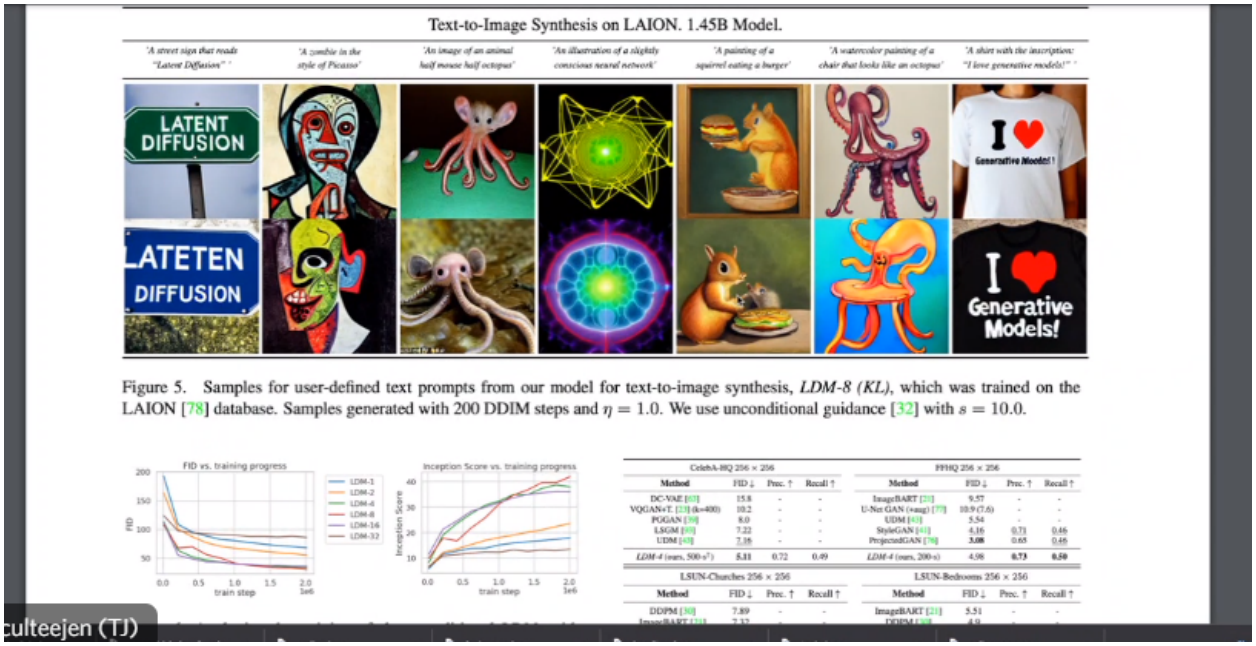


Figure 5. Samples for user-defined text prompts from our model for text-to-image synthesis, *LDM-8 (KL)*, which was trained on the LAION [78] database. Samples generated with 200 DDIM steps and $\eta = 1.0$. We use unconditional guidance [32] with $s = 10.0$.

Fine tuned a diffusion model - but can't spell.

You could reddit karma _ signs like these.
Spelling on signs. There is no post.

High fidelity -fullyness methods.
Really clever things are happening.

The quality (the semantics is a subjective things) code synthesis the code you get out of it doesnt' work then you know something has gone wrong.

Perhaps the model are getting a free pass from us because we interpret them as good images.
The image quality is going up.

LDM - Very good at recreating and understanding the image from voids.
Harder time generating text if its upside down (dataset there is not as much as horizontal text) not seem that much text overall.

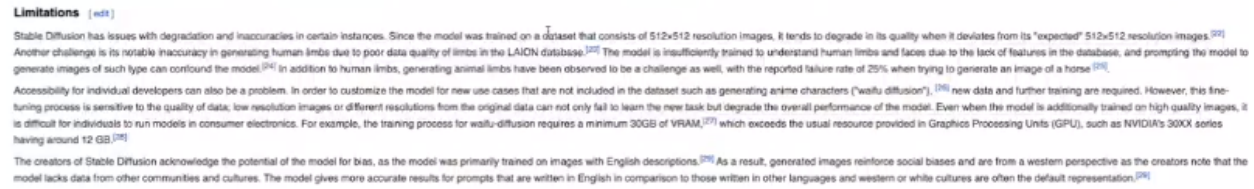
Augmentations that were used:

Text is a weak point of these models.

We don't know if there is a bit of cherry picking going on in those images.

The wikipedia page has a good summary on latent diffusion.

Markov Kernel.



Limitations [edit]

Stable Diffusion has issues with degradation and inaccuracies in certain instances. Since the model was trained on a dataset that consists of 512×512 resolution images, it tends to degrade in its quality when it deviates from its "expected" 512×512 resolution images.[22] Another challenge is its notable inaccuracy in generating human limbs due to poor data quality of limbs in the LAION database.[22] The model is insufficiently trained to understand human limbs and faces due to the lack of features in the database, and prompting the model to generate images of such type can confound the model.[24] In addition to human limbs, generating animal limbs have been observed to be a challenge as well, with the reported failure rate of 25% when trying to generate an image of a horse.[25]

Accessibility for individual developers can also be a problem. In order to customize the model for new use cases that are not included in the dataset such as generating anime characters ("waifu diffusion"),[26] new data and further training are required. However, this fine-tuning process is sensitive to the quality of data; low resolution images or different resolutions from the original data can not only fail to learn the new task but degrade the overall performance of the model. Even when the model is additionally trained on high quality images, it is difficult for individuals to run models in consumer electronics. For example, the training process for waifu-diffusion requires a minimum 30GB of VRAM,[27] which exceeds the usual resource provided in Graphics Processing Units (GPU), such as NVIDIA's 30XX series having around 12 GB.[28]

The creators of Stable Diffusion acknowledge the potential of the model for bias, as the model was primarily trained on images with English descriptions.[29] As a result, generated images reinforce social biases and are from a western perspective as the creators note that the model lacks data from other communities and cultures. The model gives more accurate results for prompts that are written in English in comparison to those written in other languages and western or white cultures are often the default representation.[29]

We look at the limitations on the wiki page of stable diffusion.

Mini Dall-E model.
Context for two thumbs up for rock and roll.

Fever dream thumb in the image.

Matt's stable diffusion.
Surprised it has that much data - animals and giraffe.
Terridactal.
Make a picture of puppy or kitty in highly quality outputs in training data (it tries).

You stick the things with lots of traction.
Mouse octopus: divide the image space between partial and partial mouse. How does it decide the pixels? *it is determined by the neuronetwork*

TV static: working backwards - every image guides how that image gets worked backwards based off structure - representation of that text. Work that noise backwards into an image that has a semantic embedding that matches that text embedding. ½ the other. When it talks about half stuff (it gives these things up abrupting in half) why is the mouse head - and the octopus the body?

If you did a bunch of C.

Mice are easier to recognize for cherry picking. Mice are easier to recognize by their ears.
Both methods of cherry picking it is easier to recognize whatever discriminative.

Orientation: of the squirrel.
Its goal is to semantically show that there is a squirrel there. Painting of squirrel eating a burger.
There is some bias in the AI.

One shows left and one shows right.
It is trying to determine - is there bias in the image? Things like that.

Two different AI - probabilistic AI matching a single prompt.
It generates from random noise - mold it to an image.
It tries to match the distribution - of images training with - a lot of the diversity is due to noise.

Some bias in the data. Have a bunch of images from advertising. Hero imaging would face left to right in the west. Whereas Right to left in other places. The noise gives you an additional input. Neither of the squirrels is eating a burger. Many more people with their mouths wide open if you google this. Penguins chucking wood. Never get the act of the penguin chopping the wood. These networks don't understand what is happening in the image. Peop;e try to use it for storyboarding. It is tough.

It may has to do with the data itself and not a lot of action images.
Image-net.
They don't contain a lot of action images.
Its trying to take these images and trying to recreate semantically a distribution semantically similar to these images. If there are a lot of still-images.

Artist that is trying to draw a possum.
You don't mathematically know what makes it possible. If you don't have an image for reference - you may not be drawing an exact possum. But you know enough possums to know what they look like. Its the same thing with a set of images (the wheel house) the data set images is sort of wheel house is what it could do. Lots of cats and lots of dogs. It will do really well at generating cats and dogs. But if you are trying to generate something fancy - it doesn't have a lot of references.

Images with not a lot of action poses.
Mangled limbs - eating something, biting down something. It doesn't have a lot of limb. It doesn't have a good understanding of limbs and how limbs work.

The labeling is a weak form of supervision.
You are able to learn about the content of actions without putting names to them. And you get some narrative about what is happening in the video. And put names to its sync.

Figure 5. Samples for user-defined text prompts from our model for text-to-image synthesis, *LDM-8 (KL)*, which was trained on the