

# Image synthesis

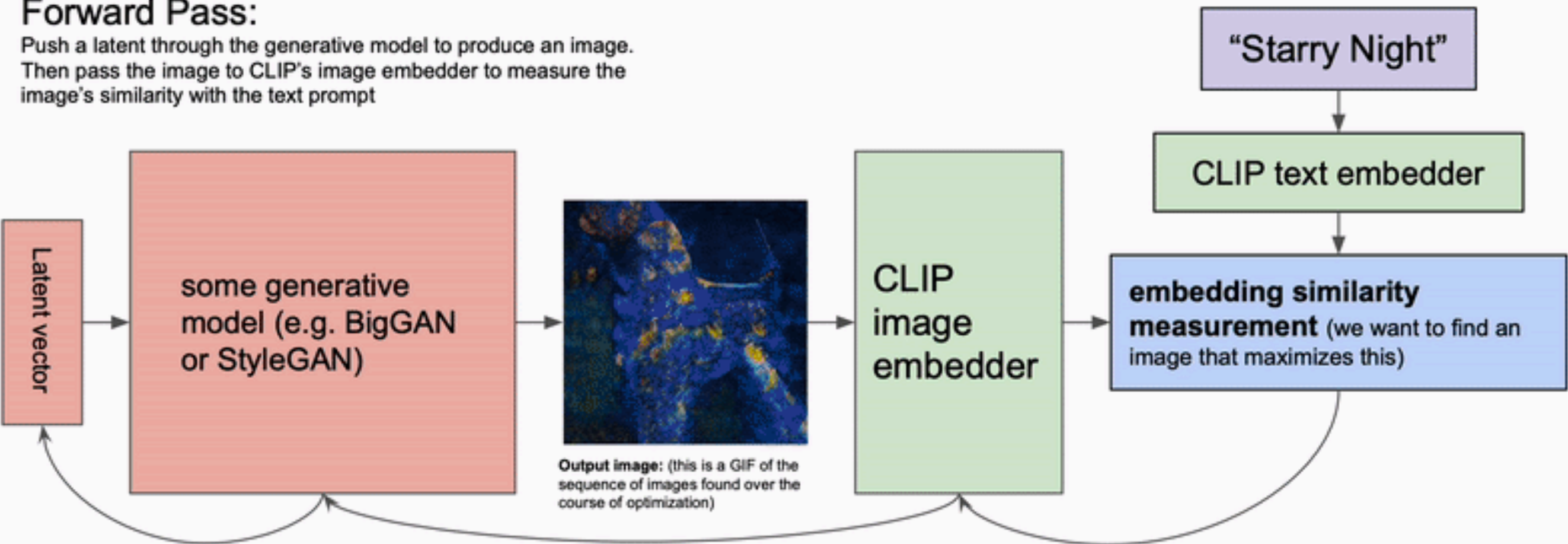
- Thanks to recent advances in multimodal (text+image) deep learning, we can now create compelling, original images by giving textual prompts to "AI artists".
- The main components of such systems are:
  - An artificial neural network capable of **generating images** by mapping real vectors to image output.
  - An artificial neural network **quantifying the relatedness** of a text and an image.
  - A gradient descent based optimization algorithm using the above two networks as "differentiable subroutines" to **gradually create an image** corresponding to the text prompt.



# How CLIP Generates Art

## Forward Pass:

Push a latent through the generative model to produce an image. Then pass the image to CLIP's image embedder to measure the image's similarity with the text prompt



**repeat forward and backward passes until convergence**

## Backward Pass:

Backpropagate through CLIP and the generative model, all the way back to the latent vector, and then use gradient ascent to update the latent, bringing the image slightly closer to matching with the text prompt.

**“Girl looking annoyed”**





“Interdimensional portal  
in the middle of the street”



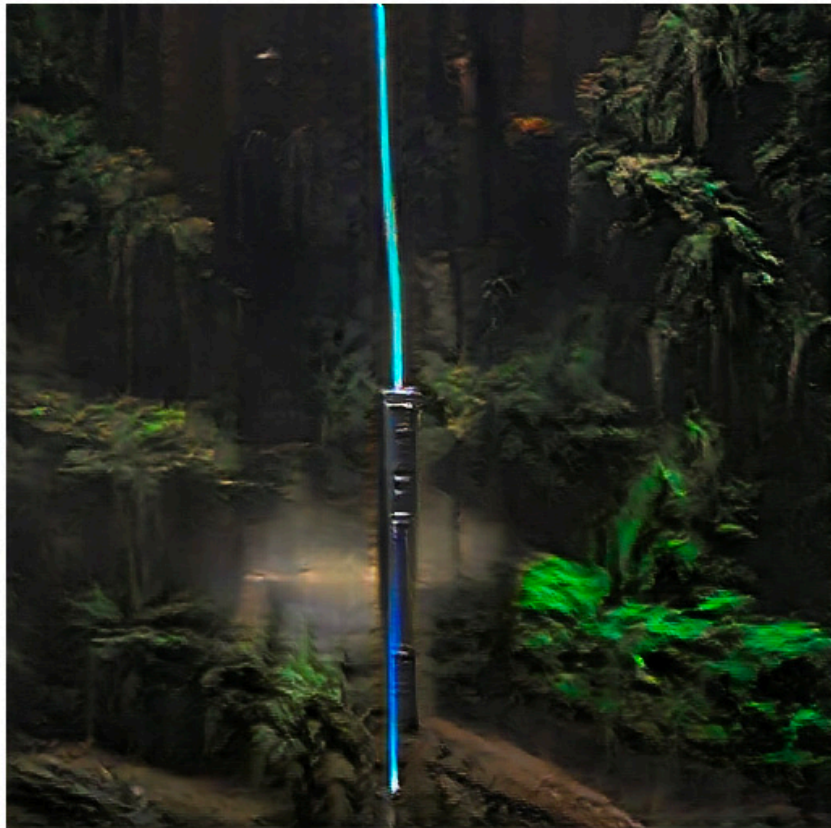
“Photograph of David Bowie in Helsinki”



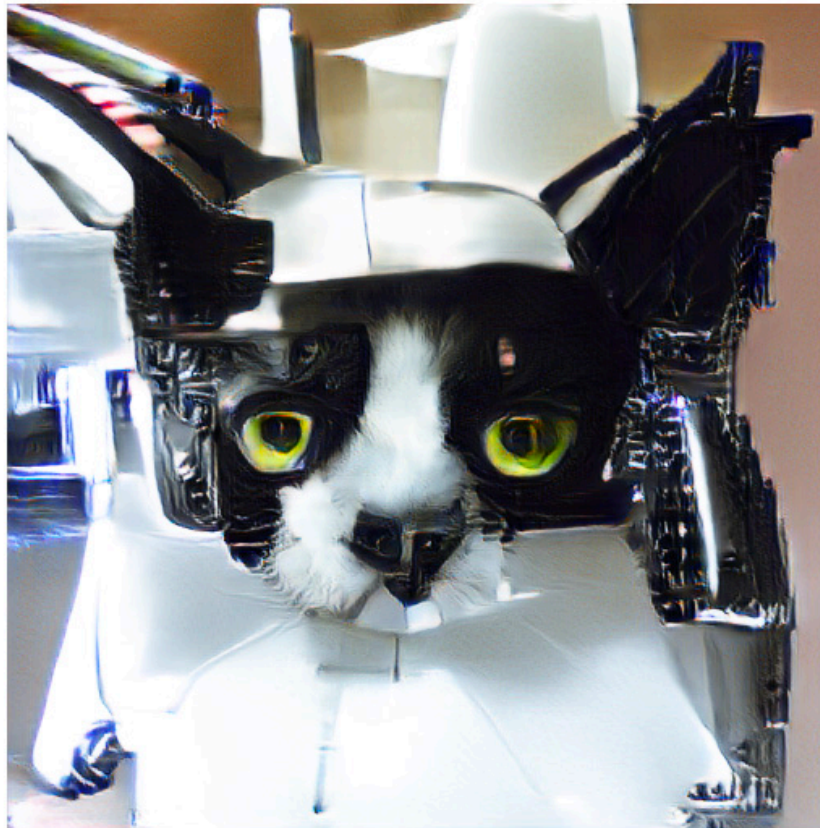
“Witches around a cauldron  
in the style of Synthwave”



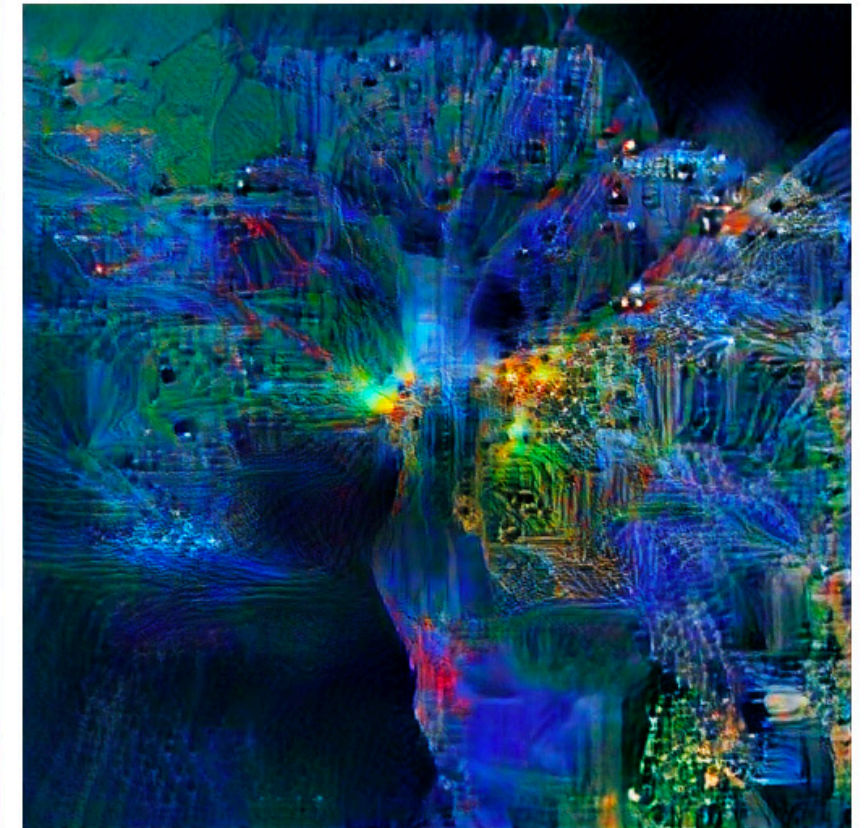
“A lightsaber in the jungle”



“Robot cat”



“Inside an AI's mind”











Inversion

Photo →  
Monet Portrait

Photo →  
Anime Painting

Photo →  
Sketch

Human →  
Tolkien Elf

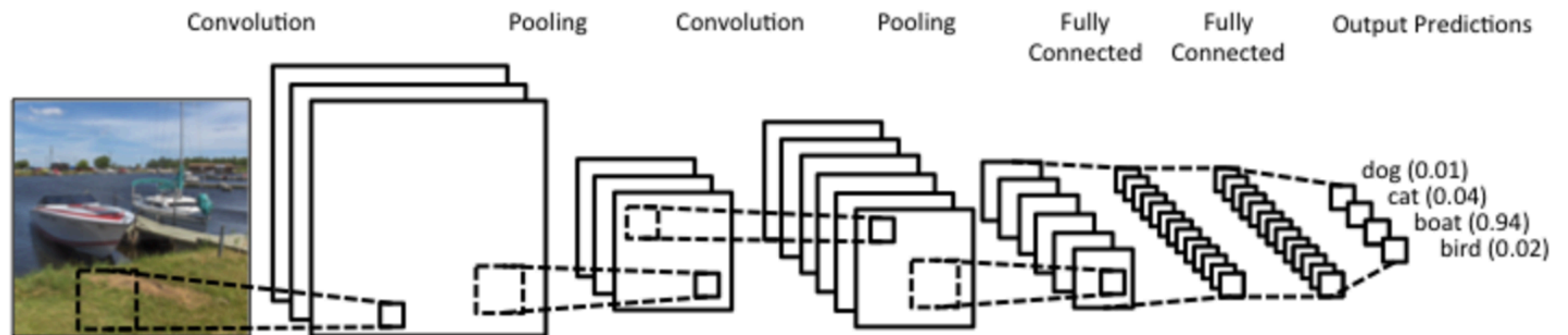
Goku →  
Super Saiyan Goku

Human →  
Zombie



# Preliminaries

How do we represent images and text?



**input:**  
**RGB pixel-**  
**intensities**  
**(224x224x3**  
**dim)**

**output:**  
**categorical**  
**(4 dim)**

# Character-based one-hot encoding

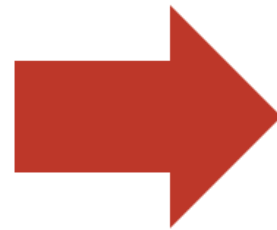
		A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
L	→	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
U	→	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
K	→	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
A	→	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
S	→	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0



# Token-based one-hot encoding

Vocabulary:

Man, woman, boy,  
girl, prince,  
princess, queen,  
king, monarch



	1	2	3	4	5	6	7	8	9
man	1	0	0	0	0	0	0	0	0
woman	0	1	0	0	0	0	0	0	0
boy	0	0	1	0	0	0	0	0	0
girl	0	0	0	1	0	0	0	0	0
prince	0	0	0	0	1	0	0	0	0
princess	0	0	0	0	0	1	0	0	0
queen	0	0	0	0	0	0	1	0	0
king	0	0	0	0	0	0	0	1	0
monarch	0	0	0	0	0	0	0	0	1

Each word gets  
a 1x9 vector  
representation

# Modern neural networks are huge

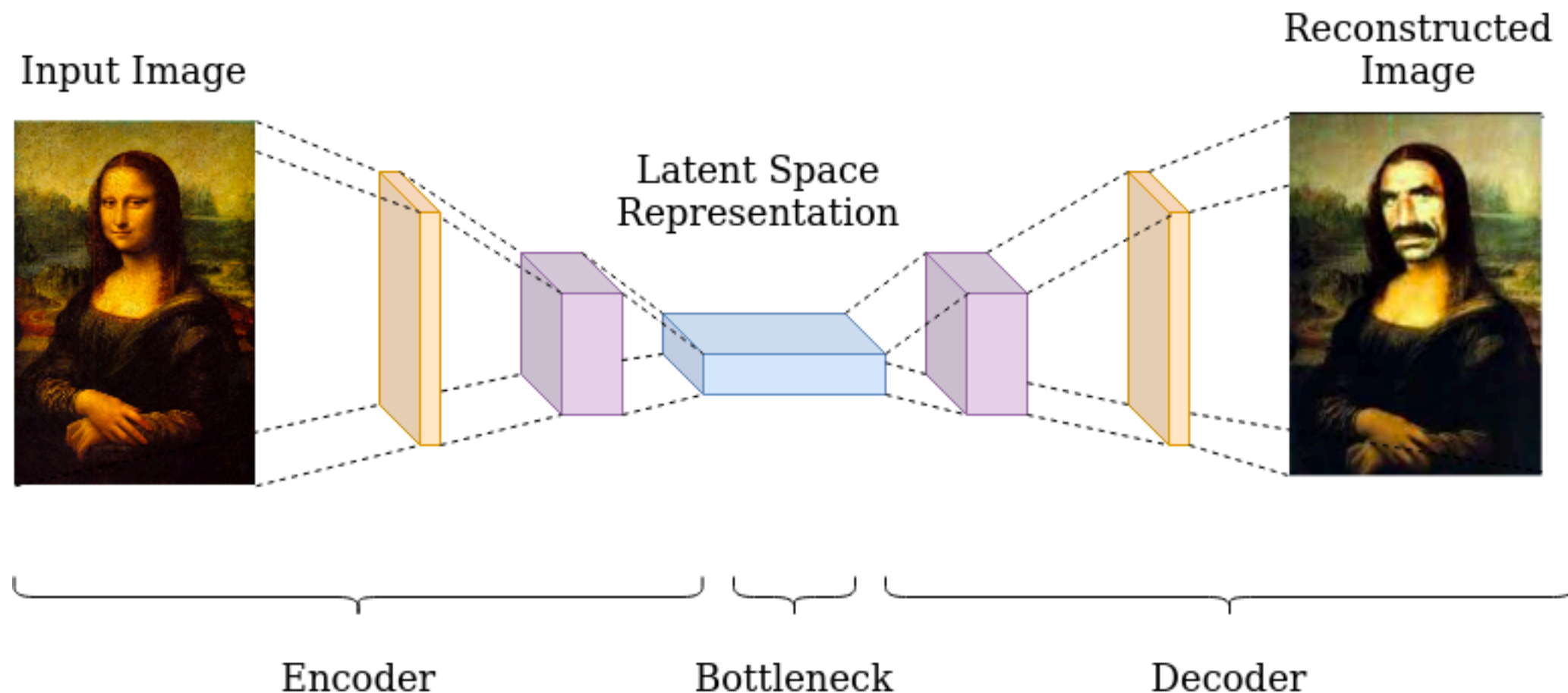
- Using simple building blocks, deep learning researchers have built network architectures that (with the right choice of parameters) can represent and **learn** extremely complex functions.
- These network architectures are HUGE, nowadays often with  $10^8$  parameters, sometimes even reaching  $10^{11}$ .
- Examples are: Residual networks, Transformers, Vision Transformers, MLP-Mixer.
- They can reach close to human performance on tasks like “what is on this image?”.

# Encoders, decoders

- We will call an *encoder* any neural network that takes some real-world data as input (image, text, sound, video), and maps it into an abstract  $d$ -dimensional vector space.
- We will call a *decoder* any neural network that maps elements of an abstract  $d$ -dimensional vector space to real-world data (image, text, sound, video).

# Autoencoders

- When we have an encoder architecture and a decoder architecture, we can plug them into each other. Their function composition maps from images to images. We can define a trivial loss function: the mean square error of the reconstruction:

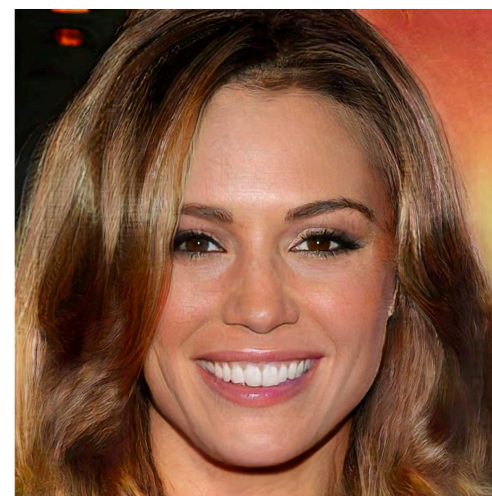
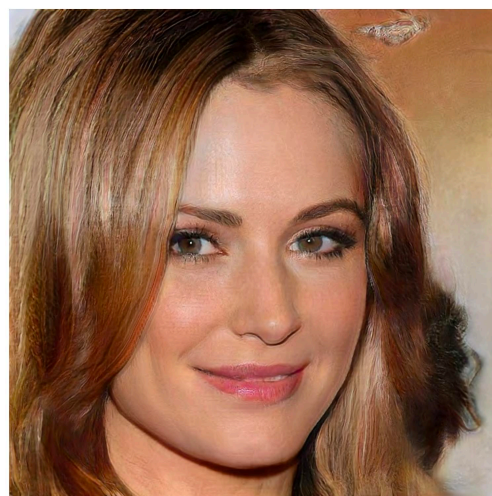
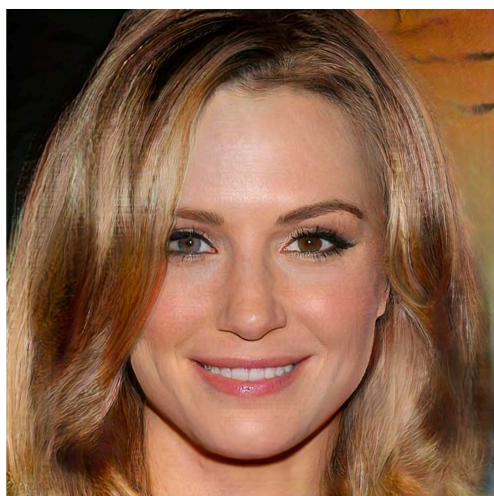
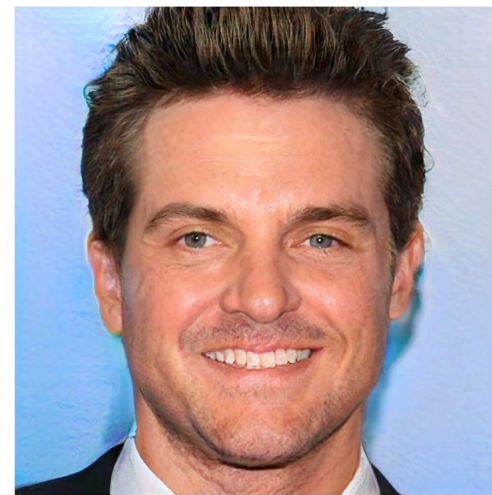
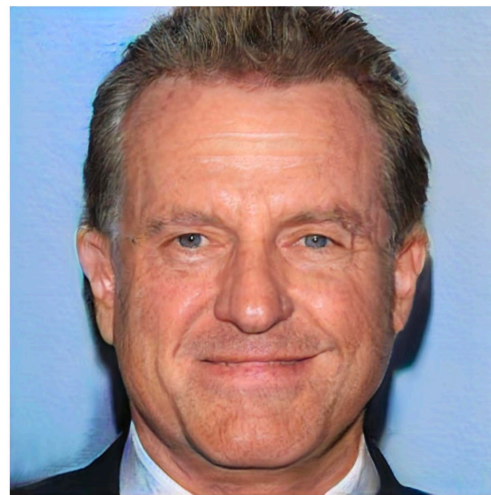
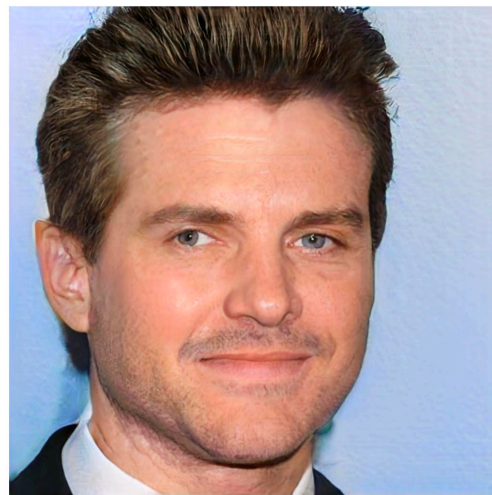


# Generators

- A *generator* is a kind of decoder, mapping latent vectors to data vectors.
- You train the generator on some data distribution  $X$ .
- You then give it a random vector as input (standard normal distribution, say).
- It is supposed to give you a random element of the data distribution  $X$ .
- Most famously GANs, but also Variational Autoencoders, Energy-based models, Autoregressive models etc.



# Latent arithmetic



Original

Pose

Age

Expression

Eyeglasses



# Homework:

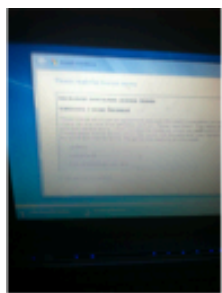
## latent space arithmetics

[https://colab.research.google.com/drive/1OxRHEfaZvqC\\_CkbSFctTzjrCLrh\\_JsHE](https://colab.research.google.com/drive/1OxRHEfaZvqC_CkbSFctTzjrCLrh_JsHE)



# Text to Image

- Let's plug an image decoder into a text encoder, and train the combination on image-caption pairs.
- Easier said than done.
- It was finally done in January 2021 by OpenAI.



A computer screen with a Windows message about Microsoft license terms.



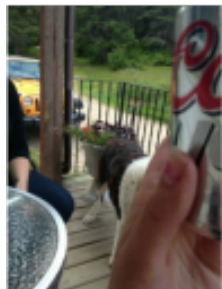
A can of green beans is sitting on a counter in a kitchen.



A photo taken from a residential street in front of some homes with a stormy sky above.



A blue sky with fluffy clouds, taken from a car while driving on the highway.



A hand holds up a can of Coors Light in front of an outdoor scene with a dog on a porch.



A digital thermometer resting on a wooden table, showing 38.5 degrees Celsius.



A Winnie The Pooh character high chair with a can of Yoohoo sitting on it in front of a white wall.



A cup holder in a car holding loose change from Canada.



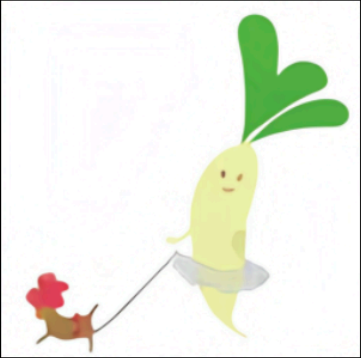




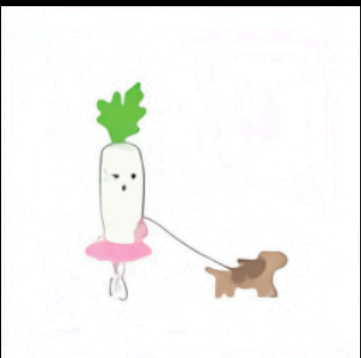


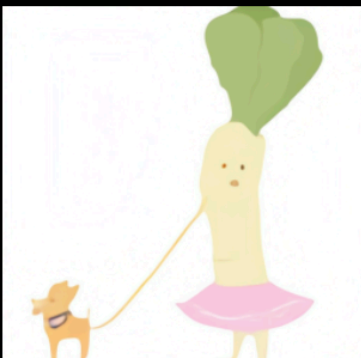
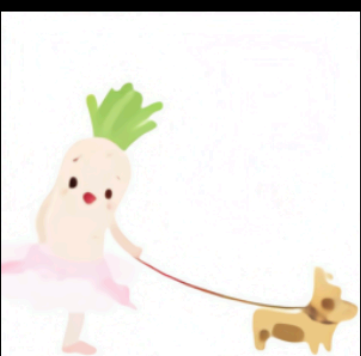
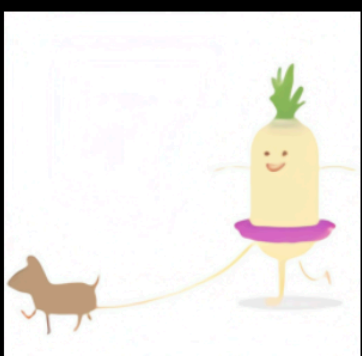

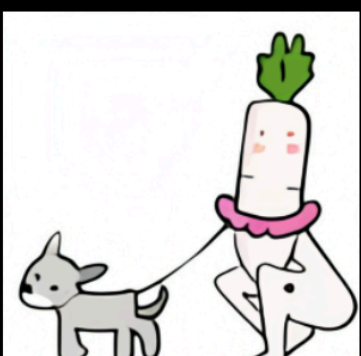
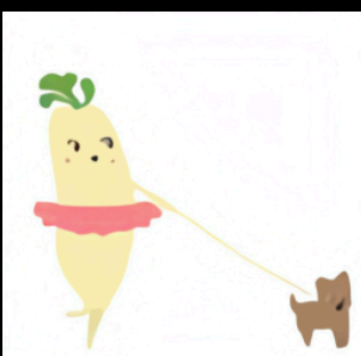
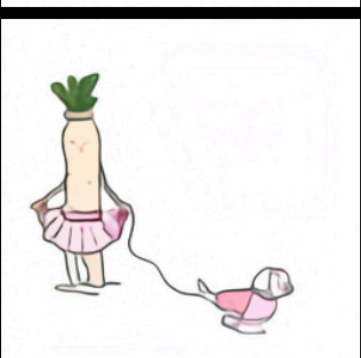
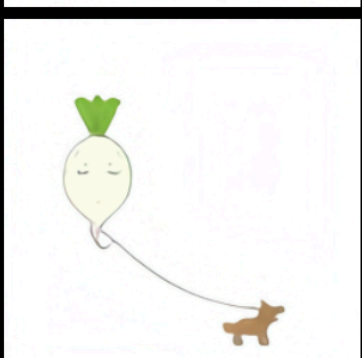
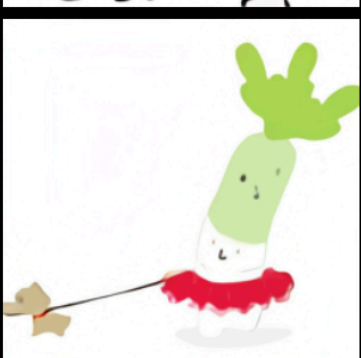
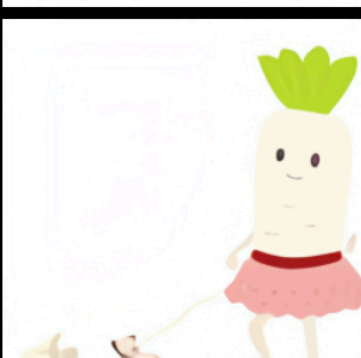
# DALL-E - text to image

TEXT PROMPT an armchair in the shape of an avocado. an armchair imitating an avocado.

AI-GENERATED  
IMAGES



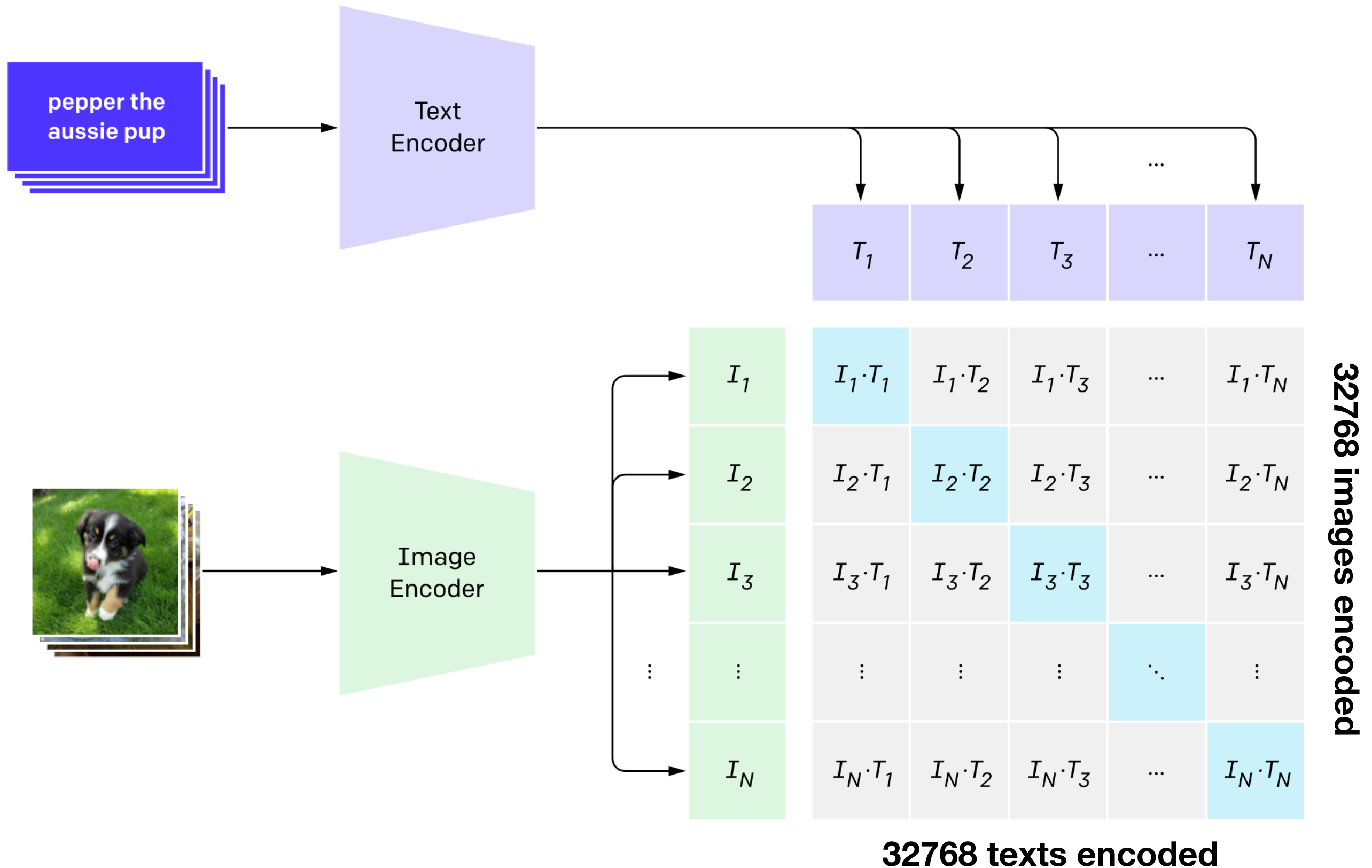
# DALL-E - text to image

TEXT PROMPT	an illustration of <u>a baby daikon radish</u> in a tutu <u>walking a dog</u>				
AI-GENERATED IMAGES					
					
					
					

# CLIP

- OpenAI have never published the text encoder + image decoder DALL-E model.
- But they have published another model based on a text encoder + image encoder architecture.
- That is CLIP.

# Contrastive training



# CLIP pseudocode

```
# image_encoder - ResNet or Vision Transformer
# text_encoder  - CBOW or Text Transformer
# I[n, h, w, c] - minibatch of aligned images
# T[n, l]       - minibatch of aligned texts
# W_i[d_i, d_e] - learned proj of image to embed
# W_t[d_t, d_e] - learned proj of text to embed
# t            - learned temperature parameter

# extract feature representations of each modality
I_f = image_encoder(I) #[n, d_i]
T_f = text_encoder(T)  #[n, d_t]

# joint multimodal embedding [n, d_e]
I_e = l2_normalize(np.dot(I_f, W_i), axis=1)
T_e = l2_normalize(np.dot(T_f, W_t), axis=1)

# scaled pairwise cosine similarities [n, n]
logits = np.dot(I_e, T_e.T) * np.exp(t)

# symmetric loss function
labels = np.arange(n)
loss_i = cross_entropy_loss(logits, labels, axis=0)
loss_t = cross_entropy_loss(logits, labels, axis=1)
loss    = (loss_i + loss_t)/2
```



# Categorical cross-entropy loss

This is how we turn network outputs into predictions:

$$p_i(x) = \frac{e^{y_i}}{\sum_j e^{y_j}}$$

(By construction,  $\forall i : 0 < p_i < 1$  and  $\sum_i p_i = 1$ .)

And this is how we turn predictions and a known correct label into a loss function:

$$L_{XENT}(p, i) = -\ln p_i$$

where  $i$  is the correct label, and  $p_j$  is the probability assigned to class  $j$  by the model.

By the way, this is just a special case of

$$L_{XENT}(p_{model}, p_{known}) = -\sum_i p_{known,i} \ln p_{model,i}$$

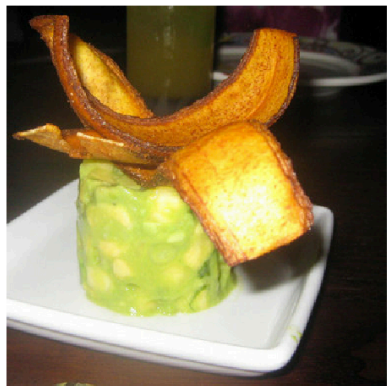
# Lots of data, lots of compute

- CLIP was trained on 400 million (image, text) pairs.
- Contrastive batch size is 32768.
- 32 epochs. (Each of the 400M pairs was shown this many times to the network during training.)
- Training took 18 days on 592 NVIDIA V100 GPUs.
- That amount of computation would cost a few 100 thousand dollars on the market. (OpenAI can thank Microsoft, though.)

# Zero-shot learning

## FOOD101

**guacamole** (90.1%) Ranked 1 out of 101 labels



✓ a photo of **guacamole**, a type of food.

✗ a photo of **ceviche**, a type of food.

✗ a photo of **edamame**, a type of food.

✗ a photo of **tuna tartare**, a type of food.

✗ a photo of **hummus**, a type of food.

## FACIAL EMOTION RECOGNITION 2013 (FER2013)

**angry** (8.2%) Ranked 5 out of 7



✗ a photo of a **happy** looking face.

✗ a photo of a **neutral** looking face.

✗ a photo of a **surprised** looking face.

✗ a photo of a **fearful** looking face.

✓ a photo of a **angry** looking face.

## SUN397

**television studio** (90.2%) Ranked 1 out of 397



✓ a photo of a **television studio**.

✗ a photo of a **podium indoor**.

✗ a photo of a **conference room**.

✗ a photo of a **lecture room**.

✗ a photo of a **control room**.

## UCF101

**Volleyball Spiking** (99.3%) Ranked 1 out of 101



✓ a photo of a person **volleyball spiking**.

✗ a photo of a person **jump rope**.

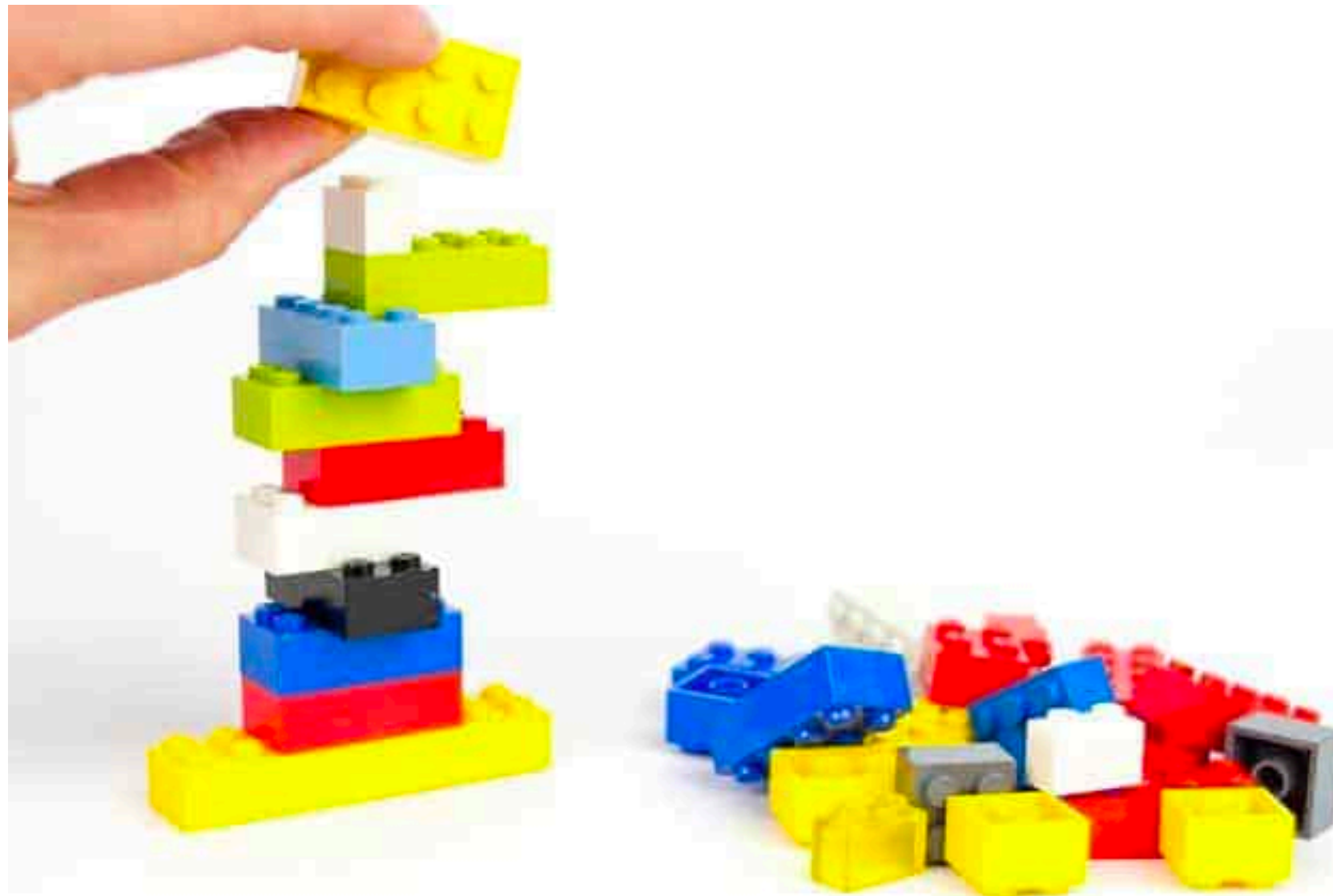
✗ a photo of a person **long jump**.

✗ a photo of a person **soccer penalty**.

✗ a photo of a person **table tennis shot**.



# Let's try to build something from this



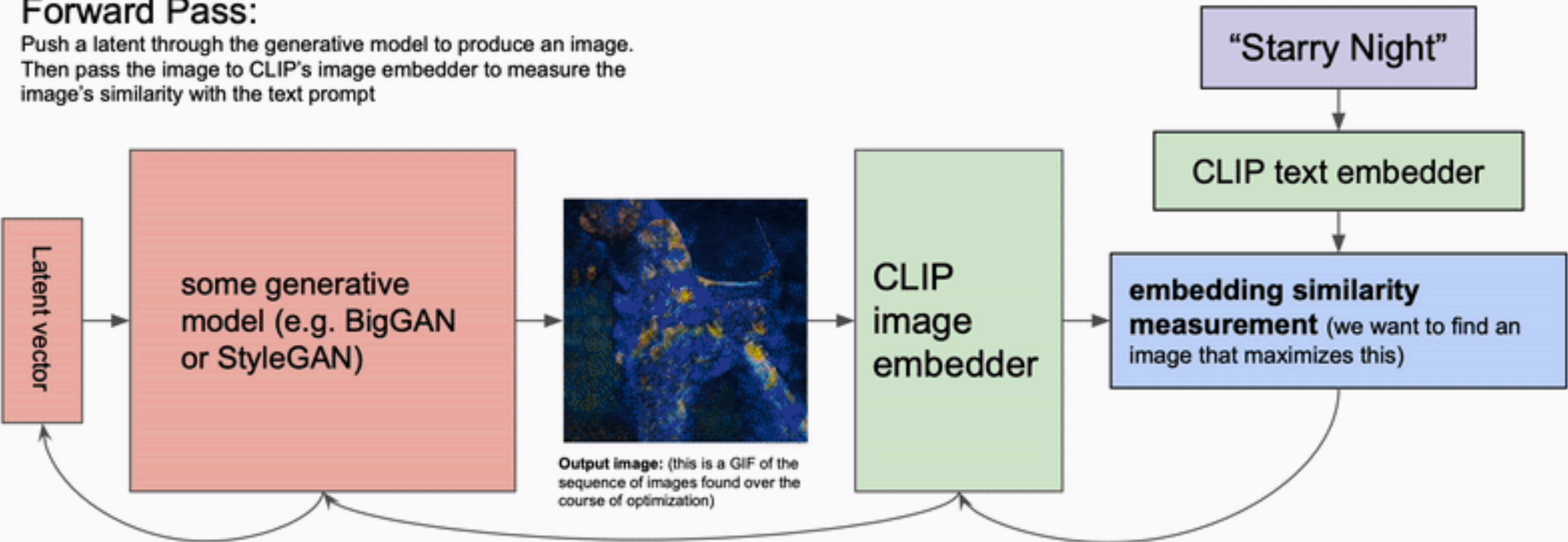
# CLIP as a poor man's DALL-E

- People soon figured out that they can combine CLIP with an image generator to do something like what DALL-E does.
- (Remember: a generator takes a random embedding vector, and outputs an image.)

# How CLIP Generates Art

## Forward Pass:

Push a latent through the generative model to produce an image. Then pass the image to CLIP's image embedder to measure the image's similarity with the text prompt



**repeat forward and backward passes until convergence**

## Backward Pass:

Backpropagate through CLIP and the generative model, all the way back to the latent vector, and then use gradient ascent to update the latent, bringing the image slightly closer to matching with the text prompt.



**“Man looking like Frankenstein’s monster”**





**“Man looking like Frankenstein’s monster”**





**“Geisha”**

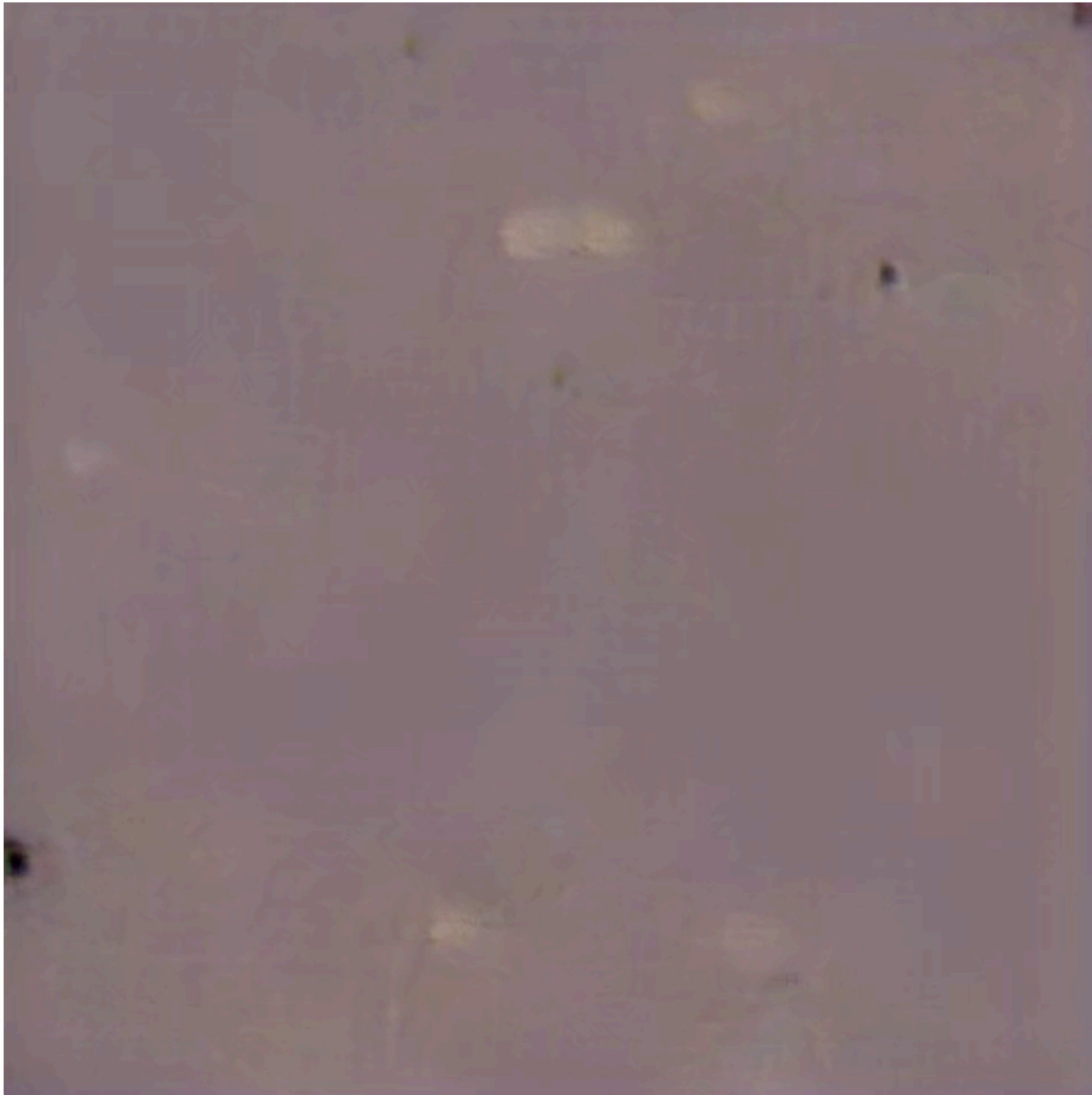




## **“Geisha”**



## **“The cactus man” (VQ-GAN generator)**

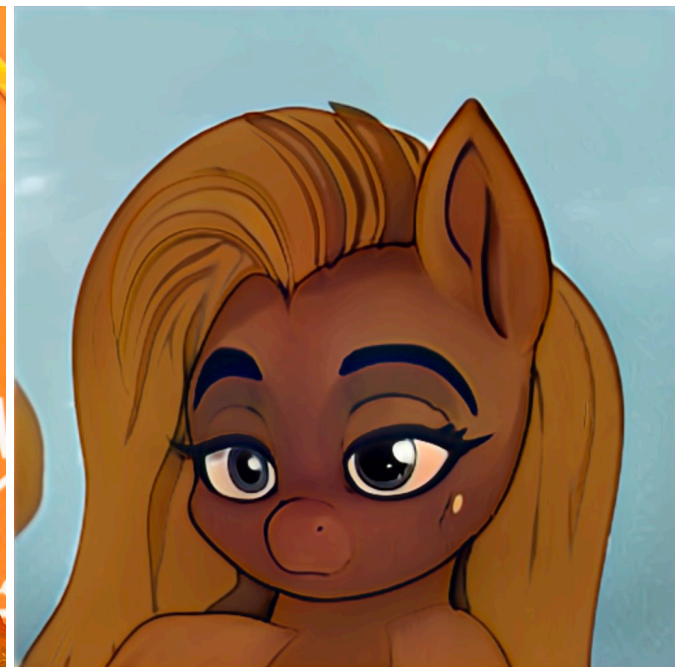




**“The cactus man” (VQ-GAN generator)**







**These are fully text directed. Left image is just illustration.  
CLIP knows how these people look, just from their names.**



“Interdimensional portal  
in the middle of the street”



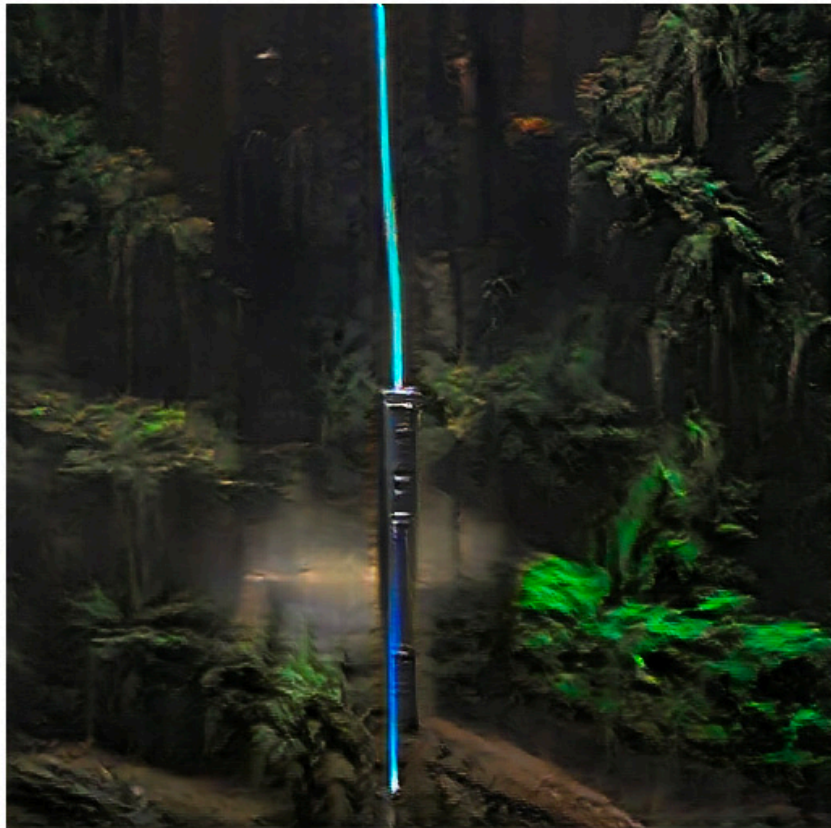
“Photograph of David Bowie in Helsinki”



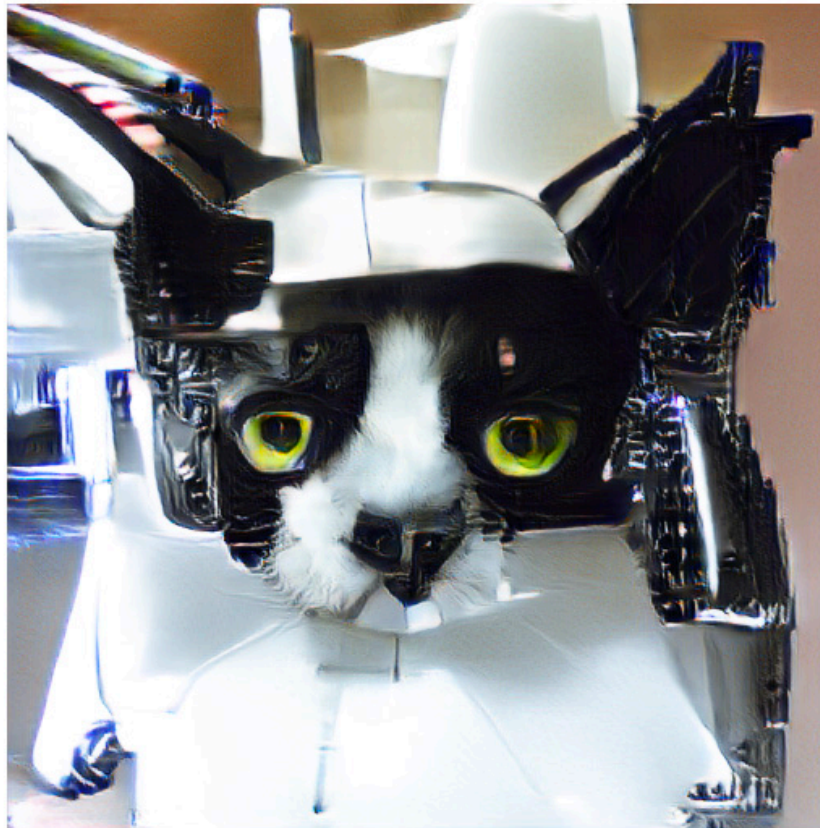
“Witches around a cauldron  
in the style of Synthwave”



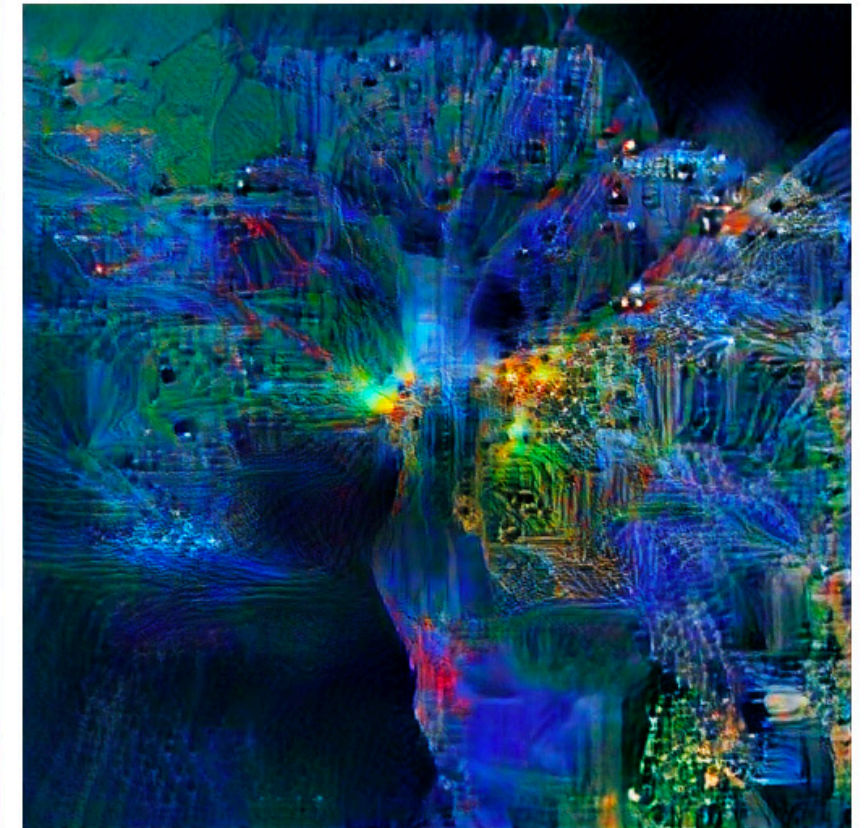
“A lightsaber in the jungle”



“Robot cat”

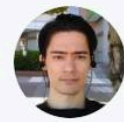


“Inside an AI's mind”





# The Unreal Engine trick



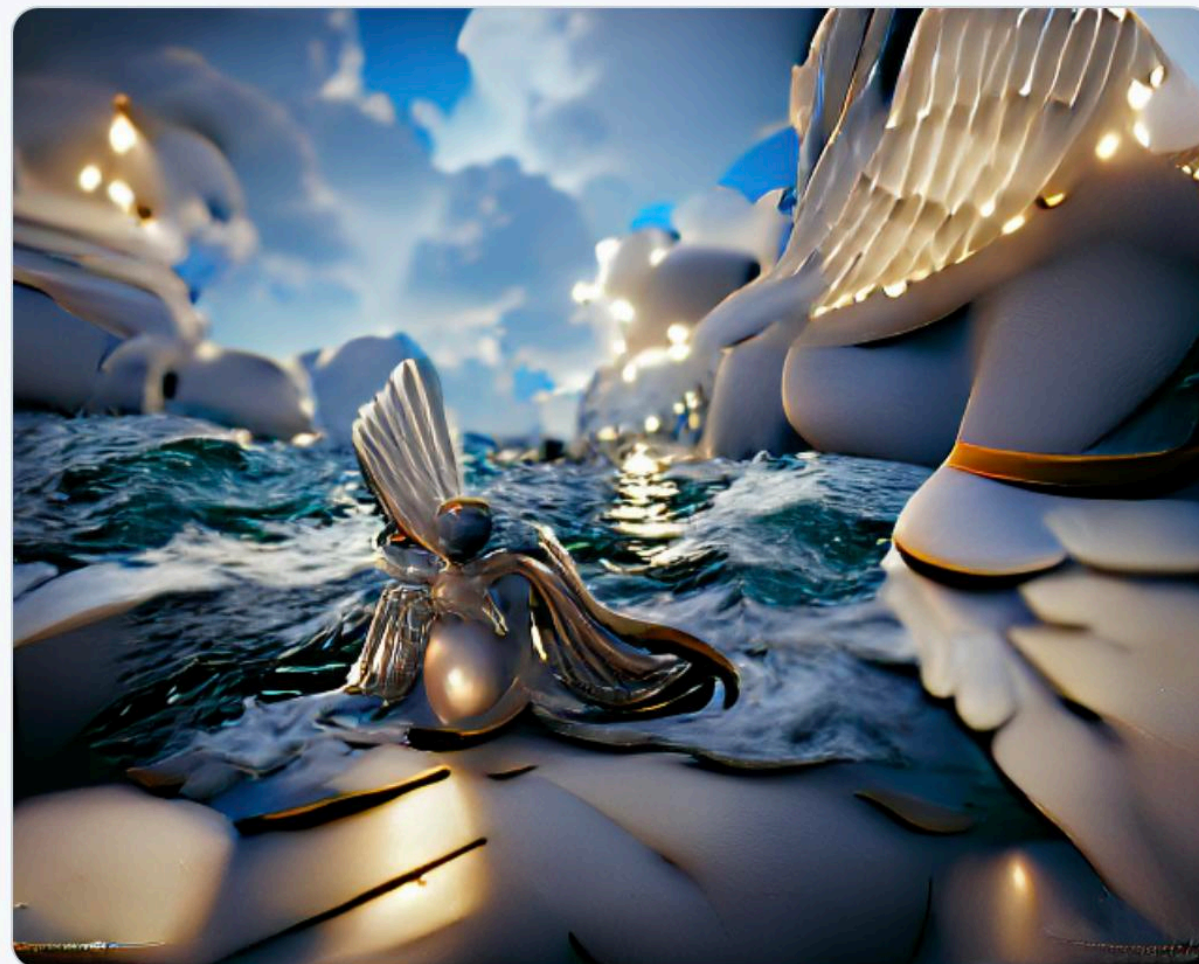
Aran Komatsuzaki  
@arankomatsuzaki



When you generate images with VQGAN + CLIP, the image quality dramatically improves if you add "unreal engine" to your prompt.

People are now calling this "unreal engine trick" lol

e.g. "the angel of air. unreal engine"



4:02 PM · May 31, 2021



♡ 2.4K    💬 40    ↗ Share this Tweet



# The Unreal Engine trick

- “trending on artstation”
- “painting by James Gurney”
- “in the style of Studio Ghibli”
- “charcoal”
- “hyperrealistic”
- “dramatic desktop wallpaper high definition”

**"matte painting of a house on a hilltop at midnight with small fireflies flying around in the style of studio ghibli | artstation | unreal engine"  
(VQ-GAN+CLIP)**



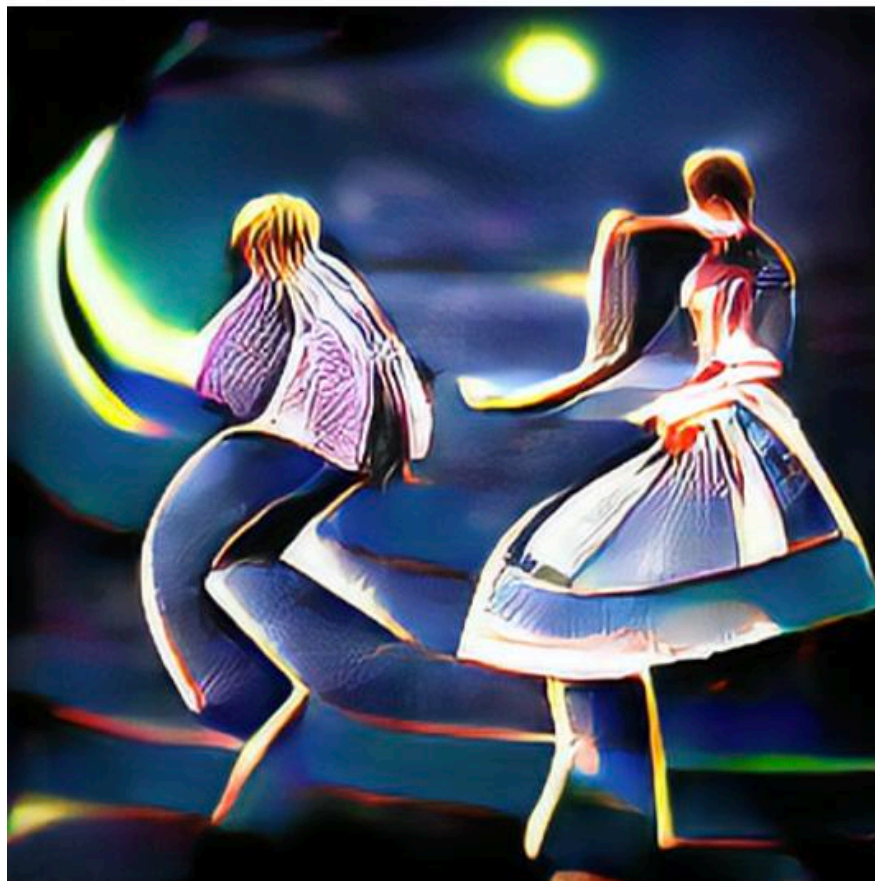
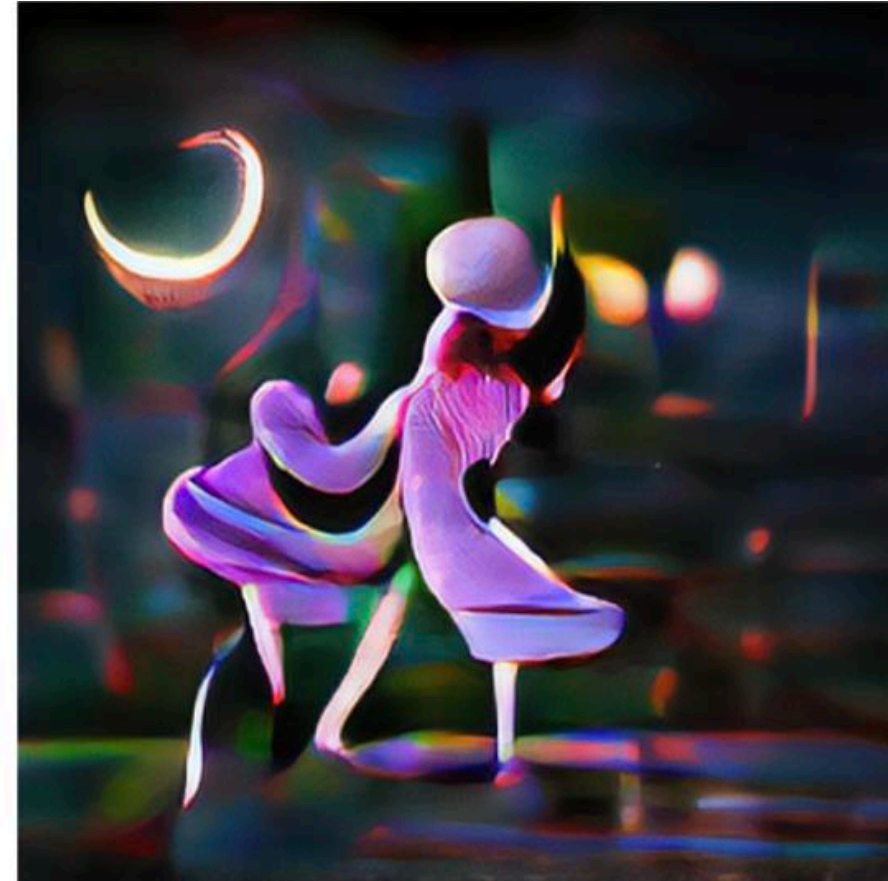


**“The Yellow Smoke That Rubs Its Muzzle On The Window-Panes” by @RiversHaveWings, (VQ-GAN+CLIP)**





**“Dancing in the moonlight” by @RiversHaveWings, (VQ-GAN+CLIP)**









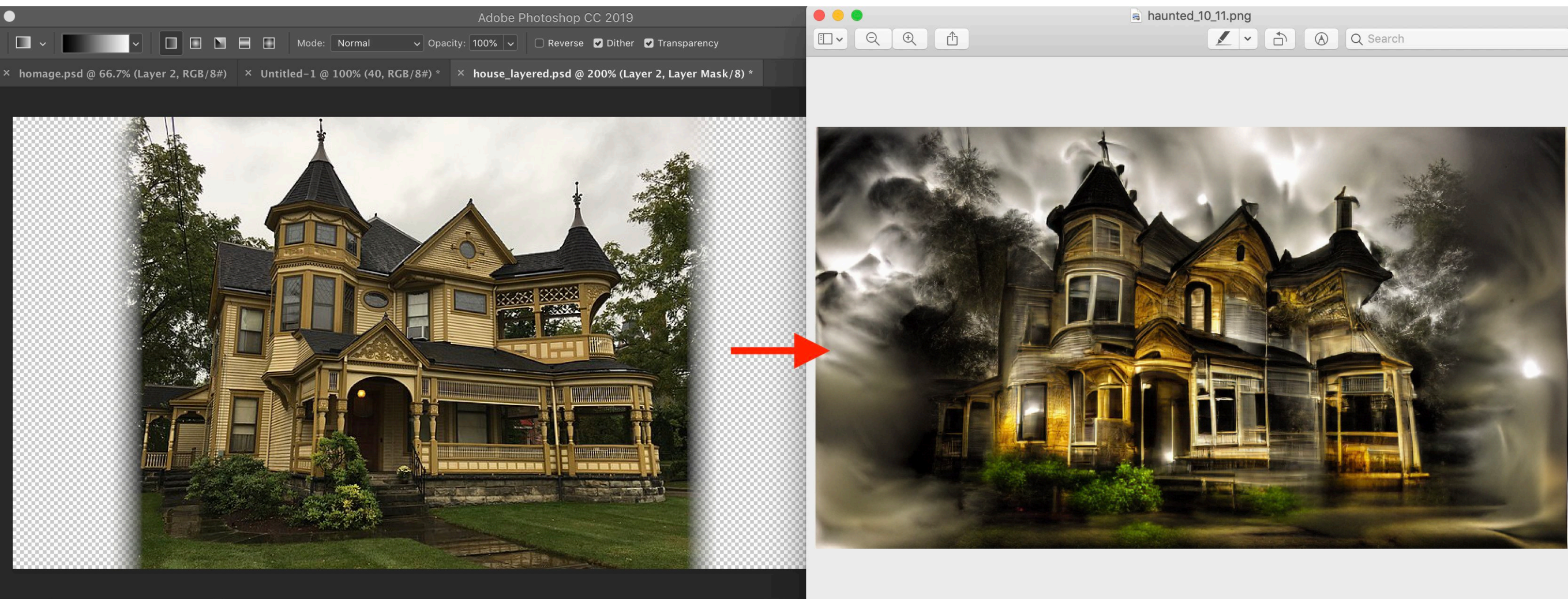




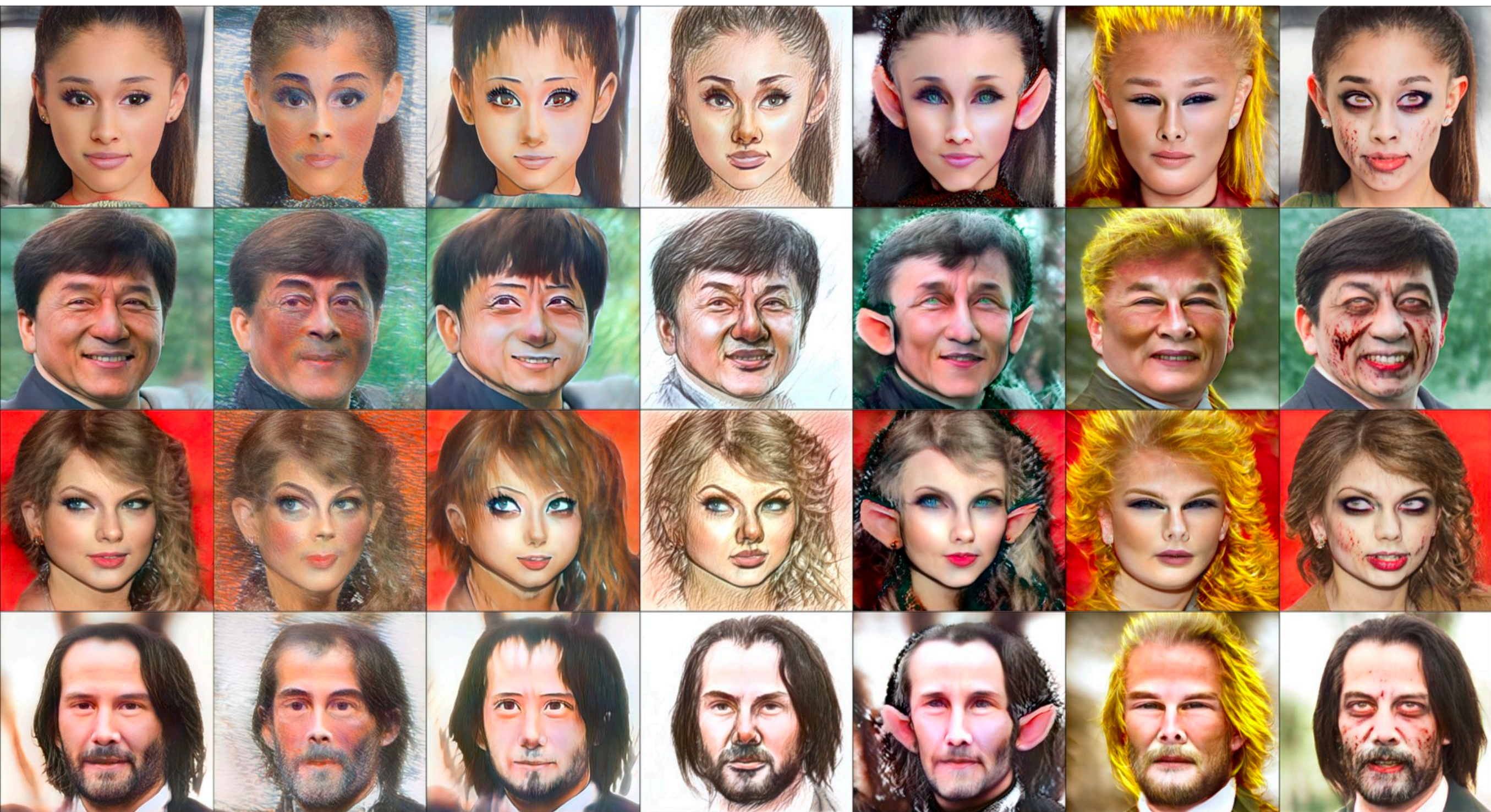




# Image inversion







Inversion

Photo →  
Monet Portrait

Photo →  
Anime Painting

Photo →  
Sketch

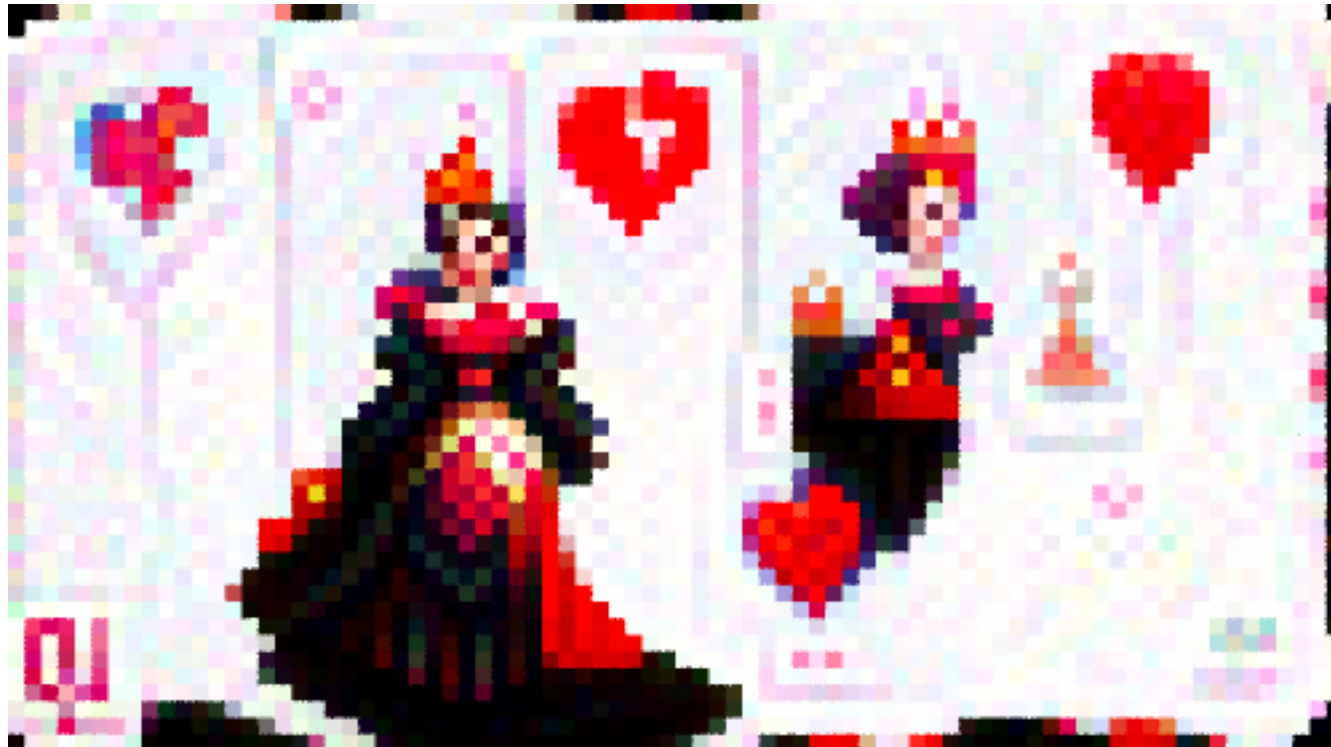
Human →  
Tolkien Elf

Goku →  
Super Saiyan Goku

Human →  
Zombie



# PixelDraw generator

















# Distraction, social commentary

- Right-clicked on the above three images and saved them.
- At the time of this writing, their NFTs trade for a few thousand USDs each.
- “To right-click is one thing, but to have a right-clicker mentality implies an ontological break between crypto-fans and critics. Indeed, it implies the person saving the JPEG to their hard drive isn’t just wrong, they’re broken in some way.” (Matthew Gault)







# Summary

- I believe the most important takeaway is: **colab**.
- There are very few people in the world who can train something like CLIP, or architect something like StyleGAN3.
- But there are millions of people who can assemble cool stuff from them. They put their creations on <https://colab.research.google.com/> and share them on blogs and Twitter.
- And anyone with an internet connection can try these creations, and inspect them, and learn from them.



# Colab notebooks

- [https://colab.research.google.com/drive/1NCceX2mbiKOSIAd\\_o7IU7nA9UskKN5WR](https://colab.research.google.com/drive/1NCceX2mbiKOSIAd_o7IU7nA9UskKN5WR) Big Sleep (BigGAN + CLIP)
- [https://colab.research.google.com/github/rinongal/stylegan-nada/blob/main/stylegan\\_nada.ipynb](https://colab.research.google.com/github/rinongal/stylegan-nada/blob/main/stylegan_nada.ipynb) StyleGAN-Nada (Photo transformation, StyleGAN2-ADA + CLIP)
- [https://colab.research.google.com/drive/1fWka\\_U56NhCegbbrQPt4PWpHPtNRdU49?usp=sharing#scrollTo=RWjzl82Nv7IG](https://colab.research.google.com/drive/1fWka_U56NhCegbbrQPt4PWpHPtNRdU49?usp=sharing#scrollTo=RWjzl82Nv7IG) CLIP-GLaSS (Various generators + CLIP)
- <https://colab.research.google.com/drive/1L8oL-vLJXVcRzCFbPwOoMkPKJ8-aYdPN> or [https://huggingface.co/spaces/akhaliq/VQGAN\\_CLIP](https://huggingface.co/spaces/akhaliq/VQGAN_CLIP) VQGAN+CLIP
- <https://colab.research.google.com/github/dribnet/clipit/blob/master/demos/PixelDrawer.ipynb> Pixray PixelArt (CLIPDraw / Pixel generator + CLIP)