Stable Diffusion

Binxu Wang, John Vastola

Machine Learning from Scratch

Nov.1st 2022




What’s the deal with all these pictures?




These pictures were generated by **Stable Diffusion,**
a recent diffusion generative model.

It can turn text prompts (e.g. “an astronaut riding a horse”) into images.
It can also do a variety of other things!

You may have also heard of DALL·E 2, which works in a similar way.

Why should we care?

"a lovely cat running in the desert in Van Gogh style, trending art."

Could be a model of imagination.

Similar techniques could be used to generate any number of things (e.g. neural data).

It's *cool!*

“Batman eating pizza in a diner"

How does it work?

It’s complicated…
but here’s the high-level idea.

What do we need?

"bad stick figure drawing"

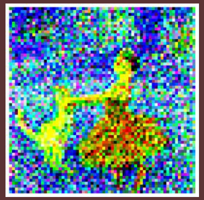
Example pictures of people

1. Method of learning to generate new stuff given many examples

What do we need?

2. Way to link text and images

"cool professor person" ⟷  

 ⟷ 

$z[0\!:3, :, :]$

3. Way to compress images
(for speed in training and generation)

What do we need?

4. Way to add in good image-related inductive biases…

…since when you’re generating something new, you need a way to safely go beyond the images you’ve seen before.

What do we need?

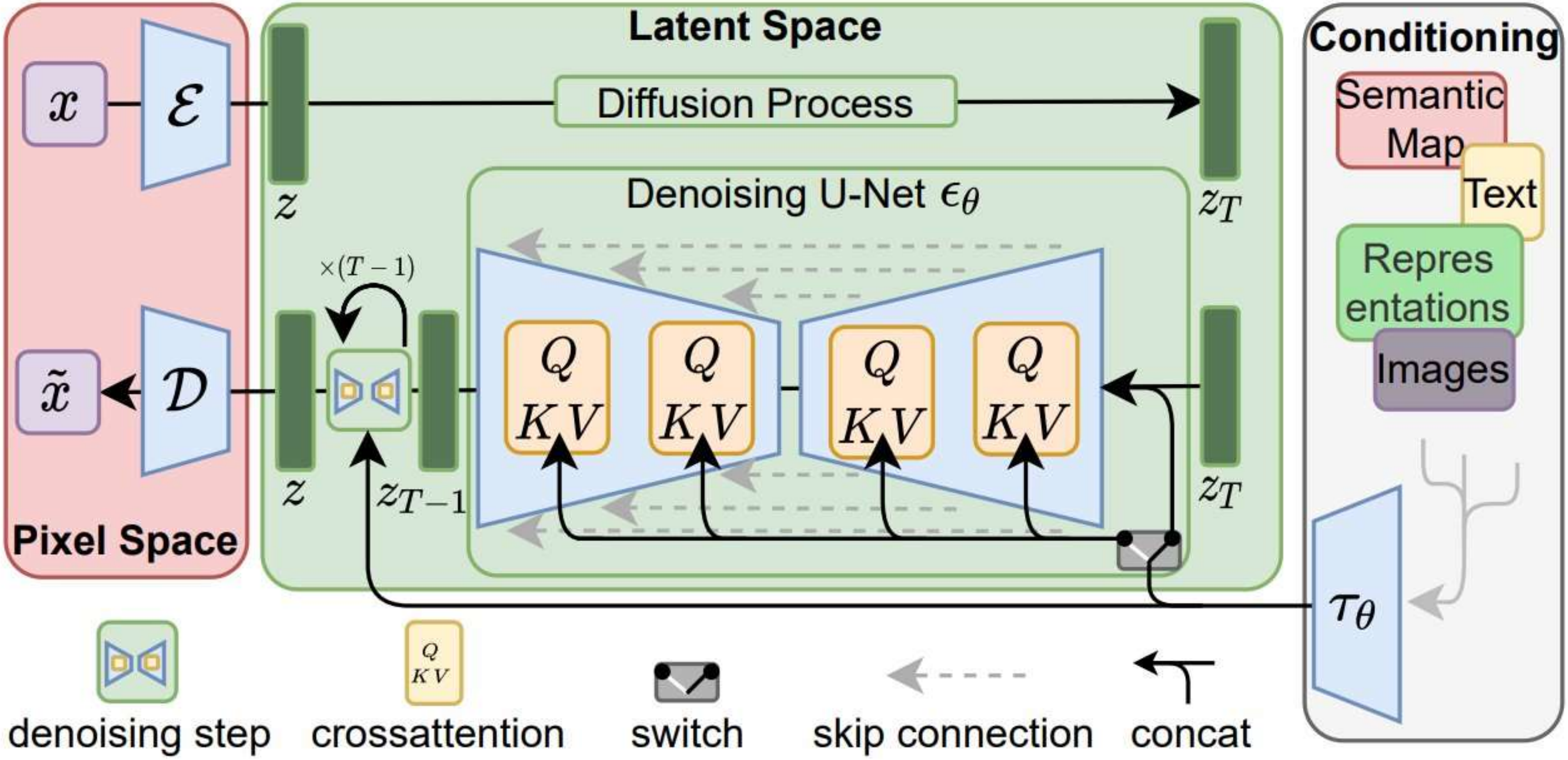
1. Method of learning to generate new stuff — **Forward/reverse dffusion**

2. Way to link text and images — **Text-image representation model**

3. Way to compress images — **Autoencoder**

4. Way to add in good inductive biases — **U-net architecture** + **'attention'**

Making a 'good' generative model is about making all these parts work together well!



Latent Space
Conditioning
Semantic Map
Text
Repres entations
Images
Diffusion Process
Denoising U-Net ϵ_θ
x
$\mathcal{E}$
z
z_T
$\times(T-1)$
$\tilde{x}$
$\mathcal{D}$
z_{T-1}
Q
$K\,V$
τ_θ
Pixel Space
denoising step
crossattention
switch
skip connection
concat

Stable Diffusion in Action

Cartoon with StableDiffusion + Cartoon

https://www.reddit.com/r/StableDiffusion/comments/xcjj7u/sd_img2img_after_effects_i_generated_2_images_and/

Some Resources

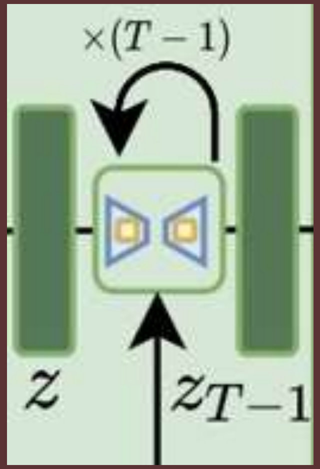
- Diffusion model in general
  - What are Diffusion Models? | Lil'Log
  - Generative Modeling by Estimating Gradients of the Data Distribution | Yang Song
- Stable diffusion
  - Annotated & simplified code: U-Net for Stable Diffusion (labml.ai)
  - Illustrations: The Illustrated Stable Diffusion – Jay Alammar
- Attention & Transformers
  - The Illustrated Transformer

Outline

- Stable Diffusion is cool!
- Build Stable Diffusion “from Scratch”
  - Principle of Diffusion models (sampling, learning)
  - Diffusion for Images – UNet architecture
  - Understanding prompts – Word as vectors, CLIP
  - Let words modulate diffusion – Conditional Diffusion, Cross Attention
  - Diffusion in latent space – AutoEncoderKL
  - Training on Massive Dataset. – LAION 5Billion
- Let’s try ourselves.

Principle of Diffusion Models

Learning to generate by iterative denoising.



“Creating noise from data is easy;
Creating data from noise is generative modeling.”

-- Song, Yang

Diffusion models

- Forward diffusion (noising)
  - $x_0 \rightarrow x_1 \rightarrow \cdots x_T$
  - Take a data distribution $x_0 \sim p(x)$, turn it into noise by diffusion $x_T \sim \mathcal{N}\left(0, \sigma^2 I\right)$

- Reverse diffusion (denoising)
  - $x_T \rightarrow x_{T-1} \rightarrow \cdots x_0$
  - Sample from the noise distribution $x_T \sim \mathcal{N}(0, \sigma^2 I)$, reverse the diffusion process to generate data $x_0 \sim p(x)$

Math Formalism

- For a forward diffusion process

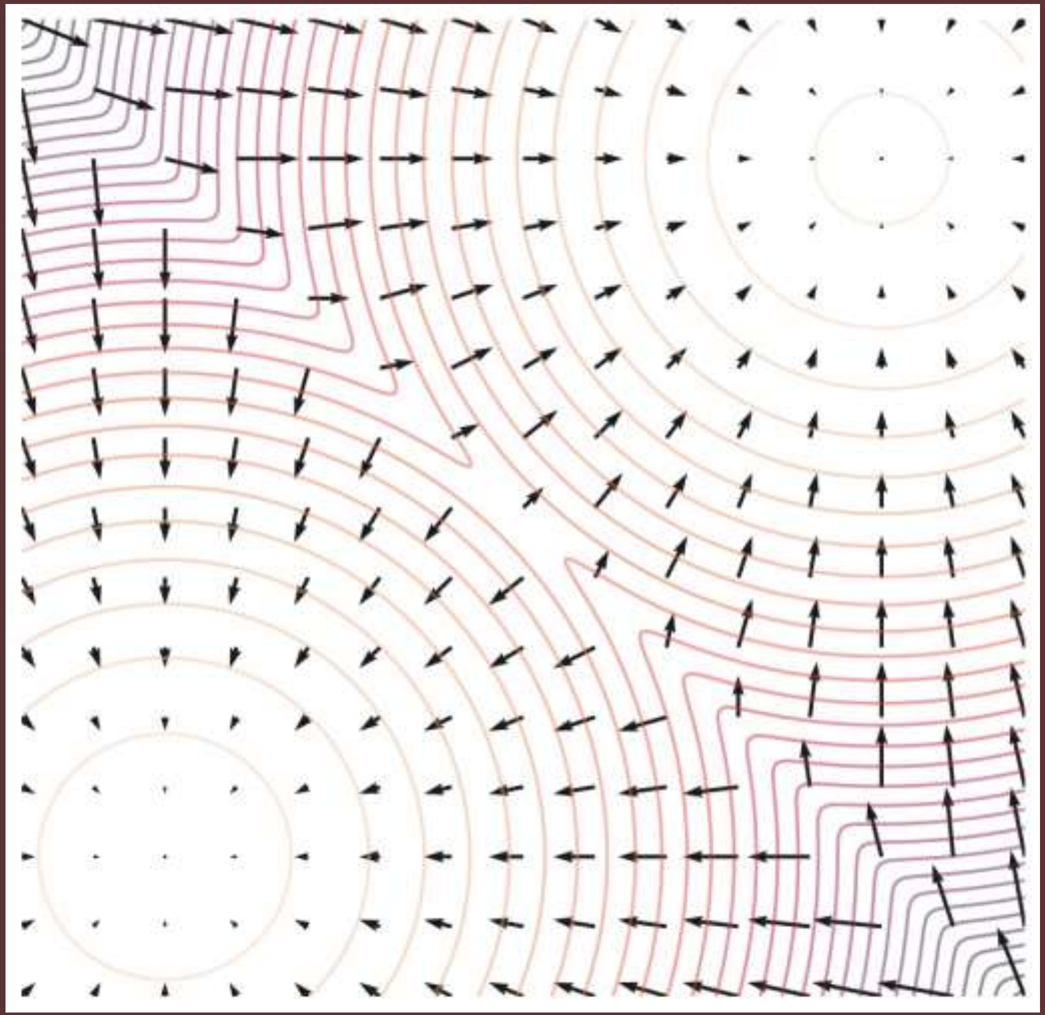
$$d\boldsymbol{x} = f(\boldsymbol{x}, t)dt + g(t)d\boldsymbol{w}$$

- There is a backward diffusion process that reverse the time

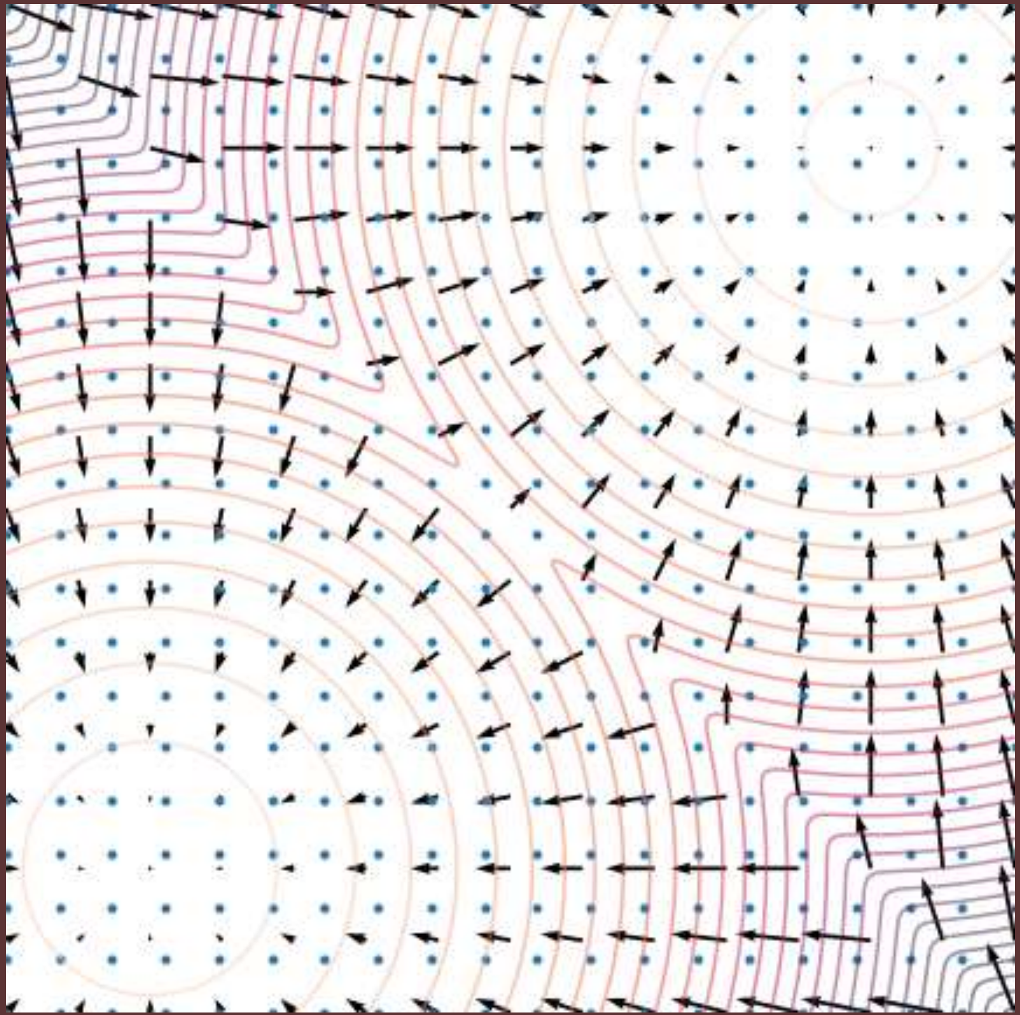
$$d\boldsymbol{x} = \left[f(x, t) - g(t)^2 \nabla_x \log p(\boldsymbol{x}, t)\right]dt + g(t)d\boldsymbol{w}$$

  - If we know the *time-dependent* score function $\nabla_x \log p(\boldsymbol{x}, t)$
  - Then we can reverse the diffusion process.

Animation for the Reverse Diffusion



Score Vector Field



Reverse Diffusion guided by the score vector field

https://yang-song.net/blog/2021/score/

Training diffusion model = Learning to denoise

- If we can learn a score model

$$f_\theta(x,t) \approx \nabla \log p(x,t)$$

  - Then we can denoise samples, by running the reverse diffusion equation. $x_t \rightarrow x_{t-1}$

- Score model $f_\theta: \mathcal{X} \times [0,1] \rightarrow \mathcal{X}$
  - A time dependent vector field over x space.

- Training objective: Infer noise from a noised sample

$$x \sim p(x), \epsilon \sim \mathcal{N}(0, I), t \in [0,1]$$

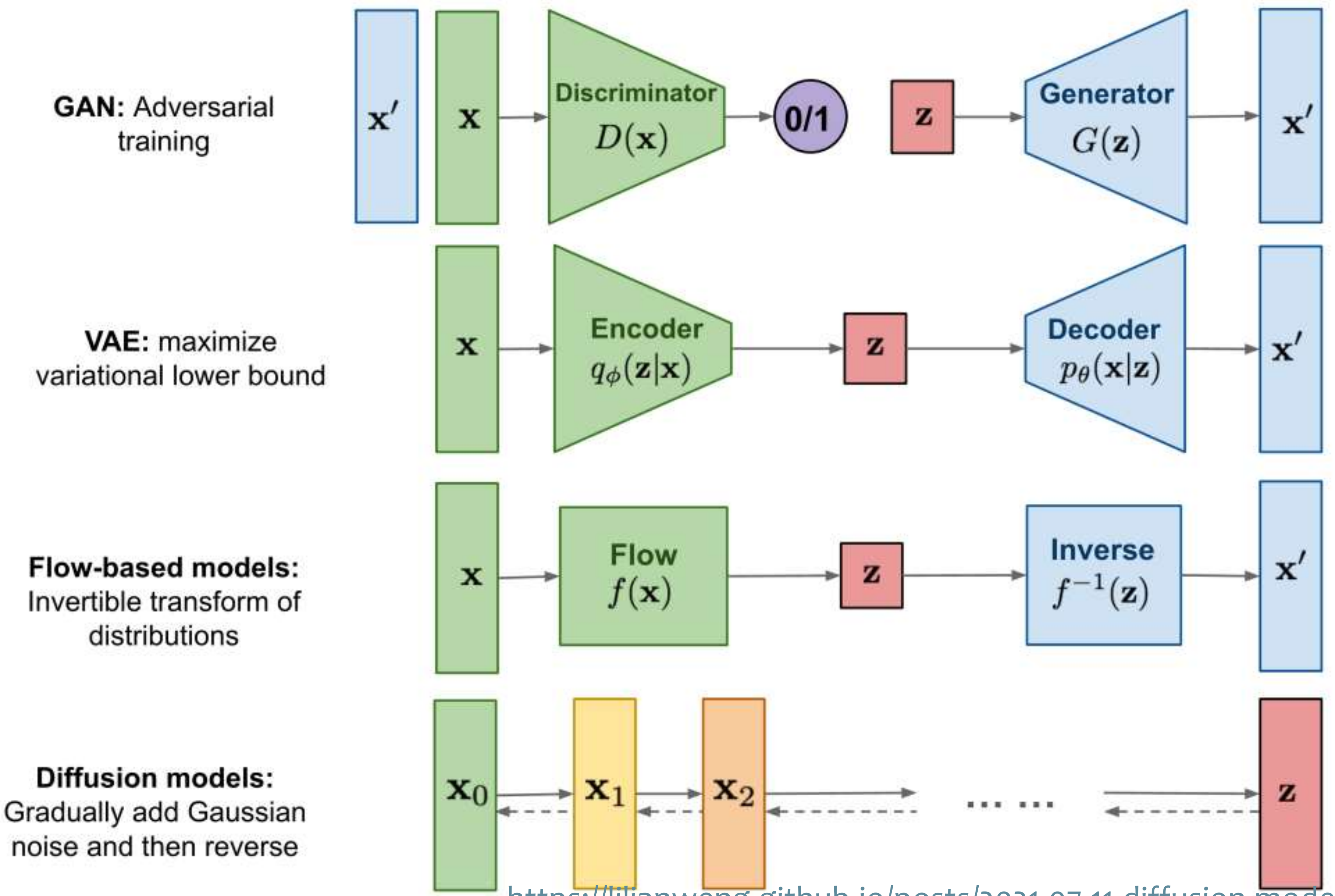
$$\min \left\| \epsilon + f_\theta(x + \sigma^t \epsilon, t) \right\|_2^2$$

  - Add Gaussian noise ϵ to an image x with scale σ^t, learn to infer the noise σ.

Conditional denoising

- Infer noise from a noised sample, based on a condition y
  - $x, y \sim p(x, y), \epsilon \sim \mathcal{N}(0, I), t \in [0,1]$
  - $\min \left\| \epsilon - f_\theta(x + \sigma^t \epsilon, y, t) \right\|_2^2$
- Conditional score model $f_\theta : \mathcal{X} \times \mathcal{Y} \times [0,1] \rightarrow \mathcal{X}$
  - Use Unet as to model image to image mapping
  - Modulate the Unet with condition (text prompt).

Comparing Generative Models



https://lilianweng.github.io/posts/2021-07-11-diffusion-models/

Diffusion vs GAN / VAE

GAN

- One shot generation. Fast.
- Harder to control in one pass.
- Adversarial min-max objective. Can collapse.

Diffusion

- Multi-iteration generation. Slow.
- Easier to control during generation.
- Simple objective, no adversary in training.

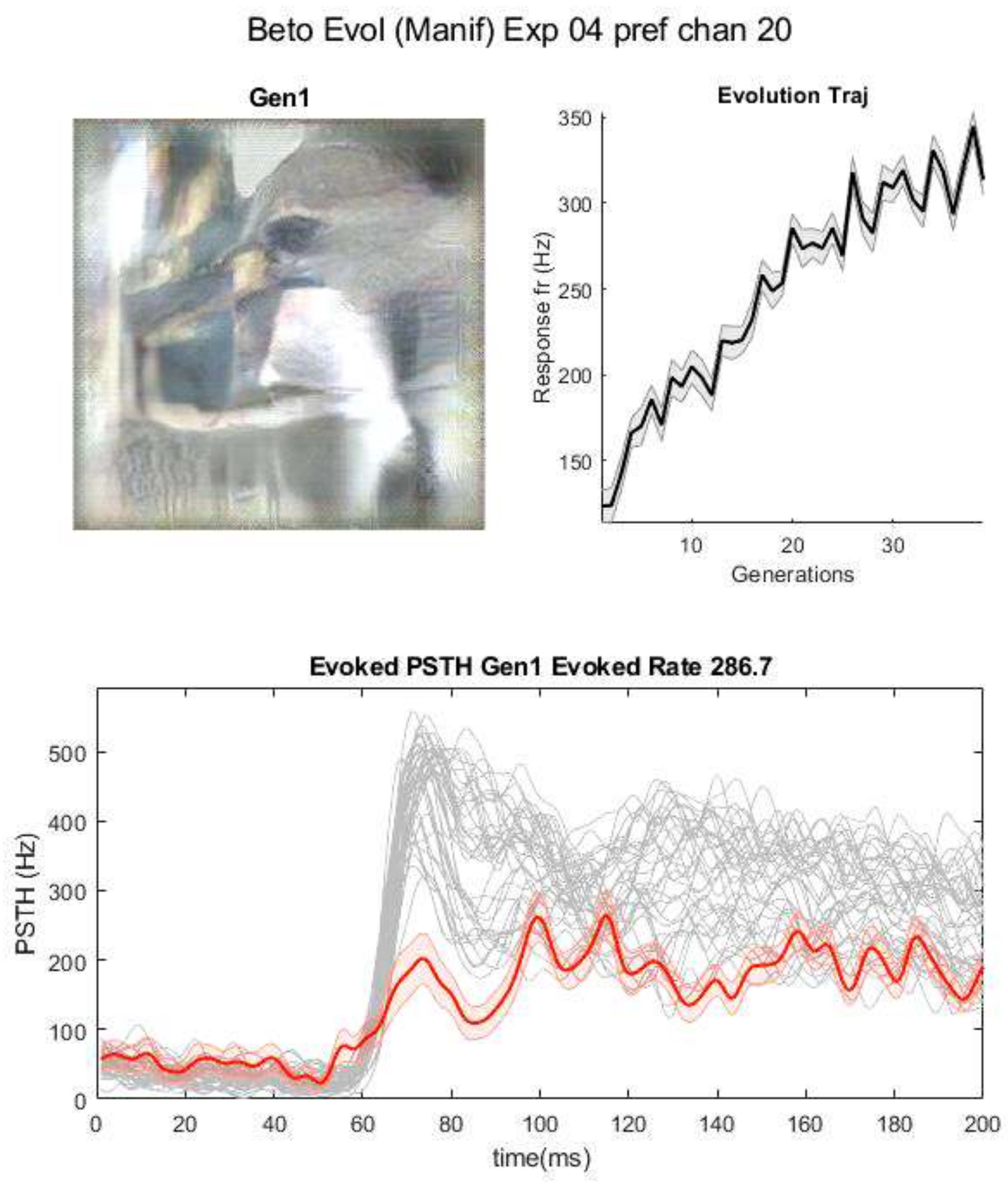
Activation maximization ~ Reverse Diffusion

- For a neuron, activation maximization can be realized by gradient ascent

$$z_{t+1} \leftarrow z_t + \nabla f\big(G(z_t)\big) + \epsilon$$

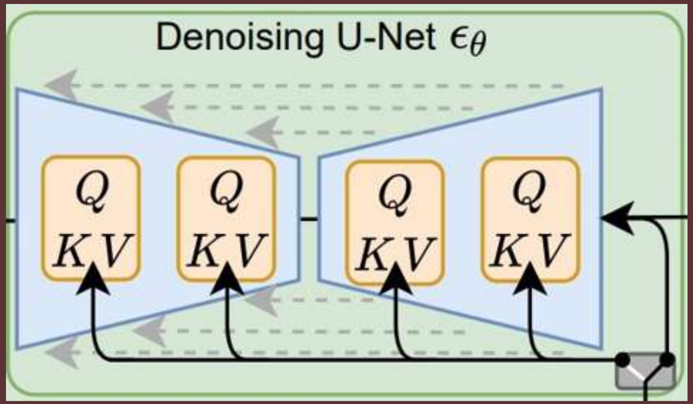
  - Homologous to the reverse diffusion equation.

- *Idea*: Neuron activation defines a Generative model on image space.

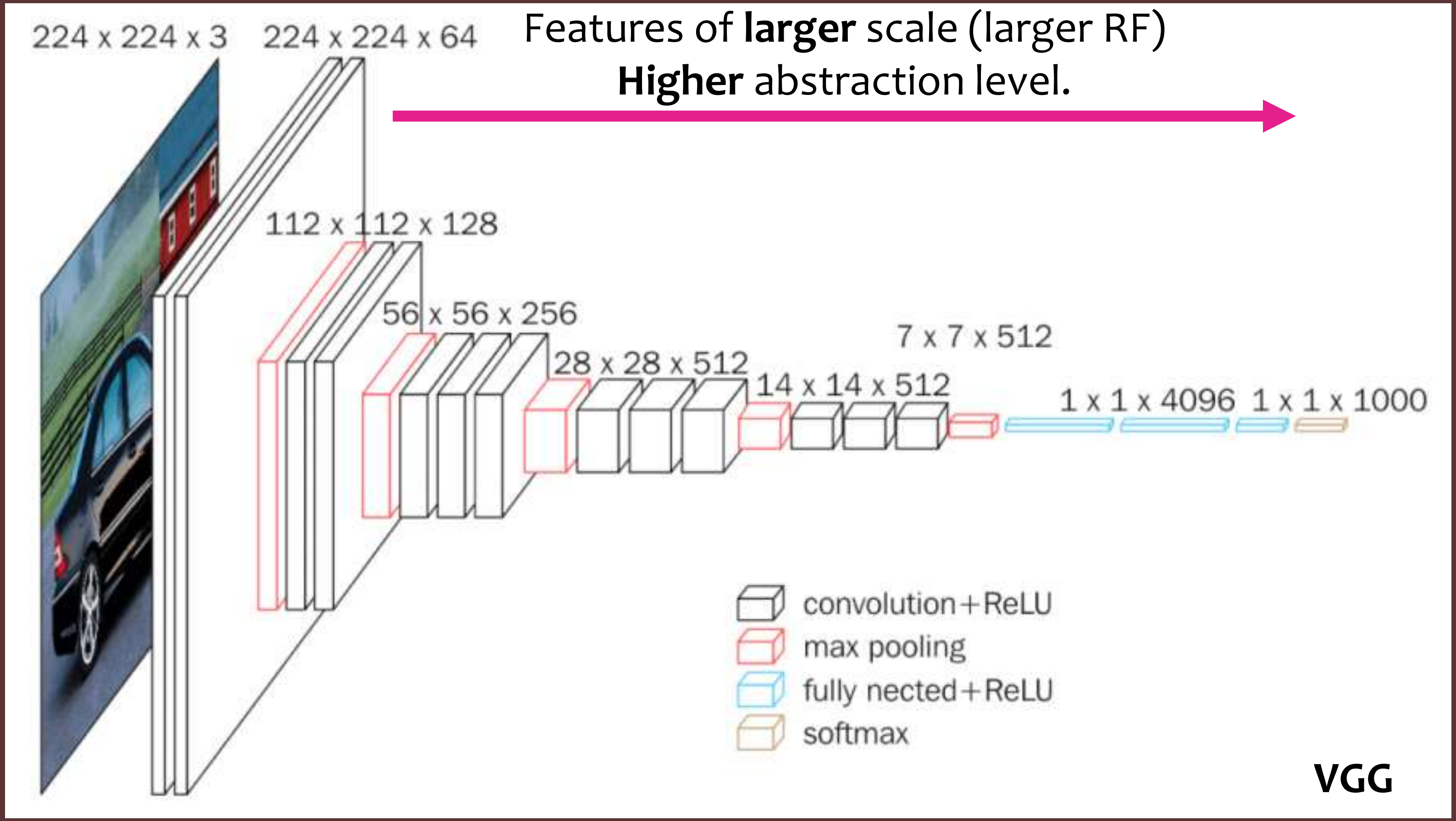


Modelling Score function over Image Domain

Introducing UNet



Convolutional Neural Network



- CNN parametrizes function over images
- Motivation
  - Features are translational invariant
  - Extract feature at different scale / abstraction level
- Key modules
  - Convolution
  - Downsamping (Max-pool)

CNN + inverted CNN ⇒ UNet

Down Sampling

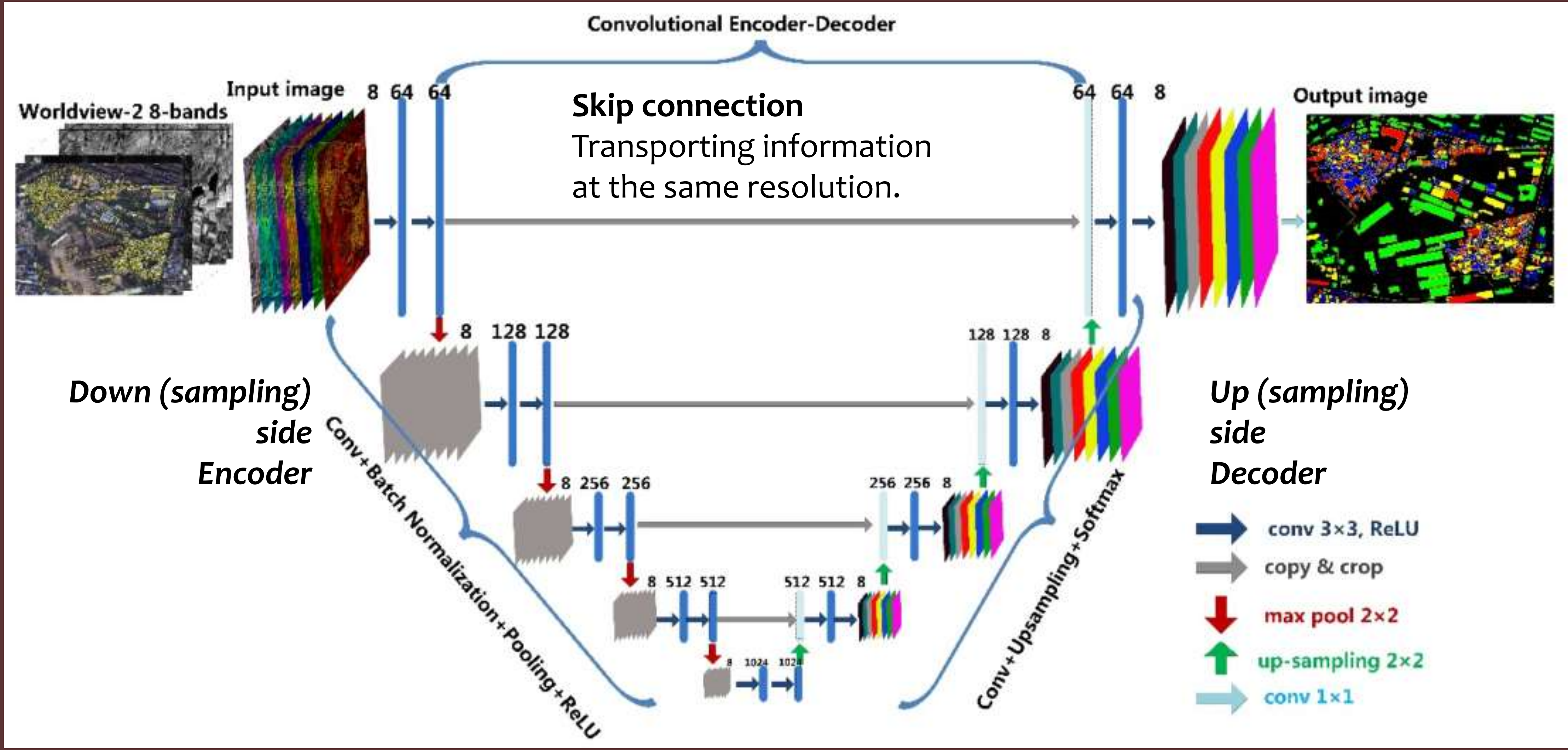
Convolution

Up Sampling

TransposedConvolution

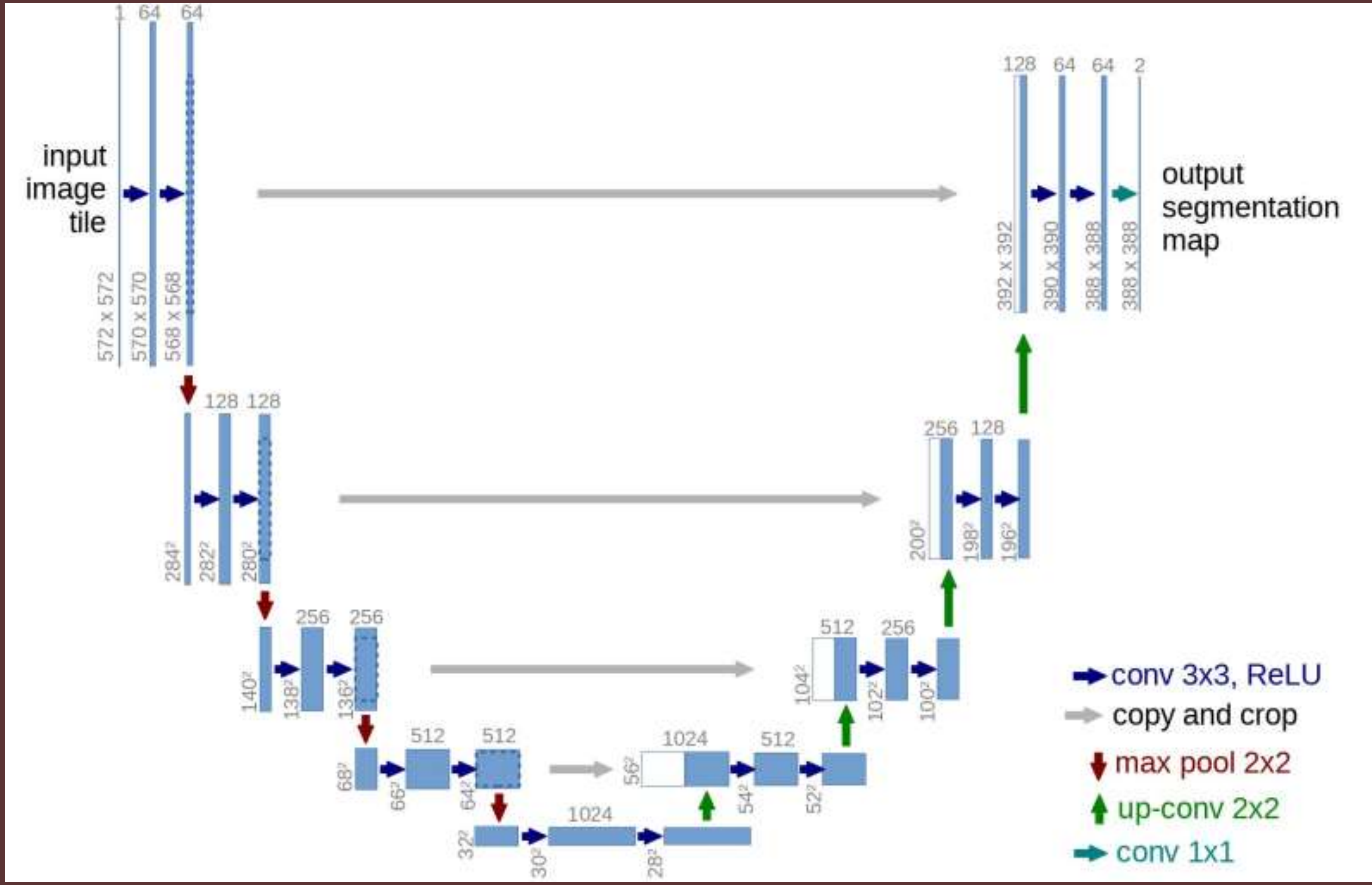
- Inverted CNN (generator) can generate images.
- CNN + inverted CNN could model Image → Image function.

UNet: a natural architecture for image-to-image function



Key Ingredients of UNet

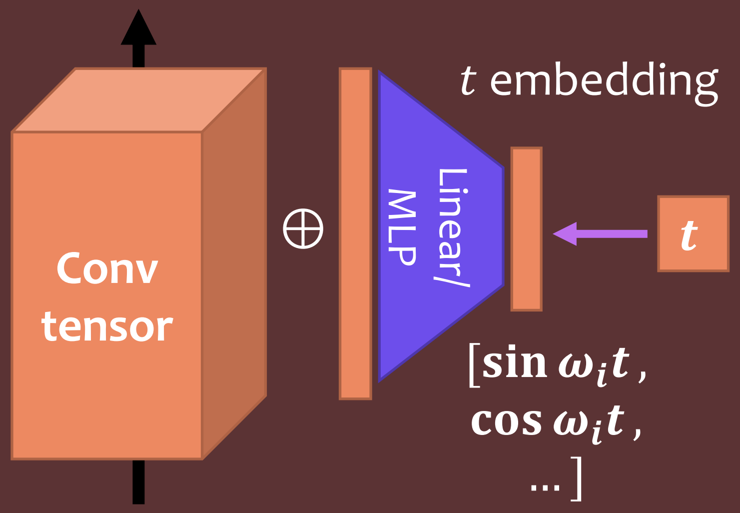
- Convolution operation
  - Save parameter, spatial invariant
- Down/Up sampling
  - Multiscale / Hierarchy
  - Learn modulation at multi scale and multi-abstraction levels.
- Skip connection
  - ***No bottleneck***
  - Route feature of the same scaledirectly.
  - Cf. AutoEncoder has bottleneck



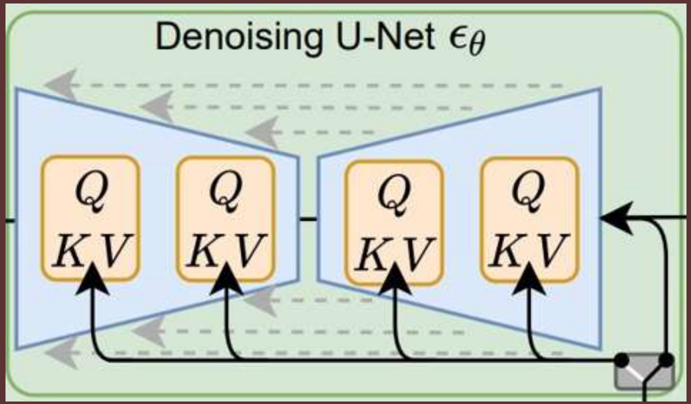
Note: Add Time Dependency

- The score function is *time-dependent*.
  - Target: $s(x,t) = \nabla_x \log p(x,t)$

- Add time dependency
  - Assume time dependency is spatially homogeneous.
  - Add one scalar value per channel $f(t)$
  - Parametrize $f(t)$ by MLP / linear of Fourier basis.



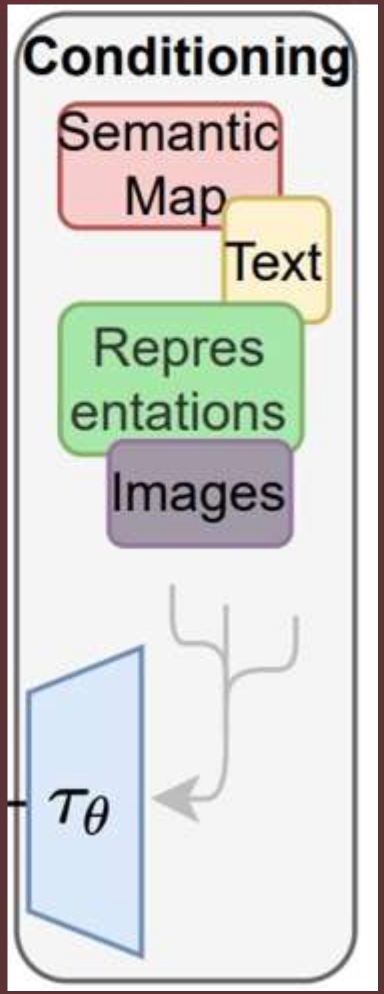
Unet in Stable Diffusion




```
(conv_in): Conv2d(4, 320, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
(time_proj): Timesteps()
(time_embedding): TimestepEmbedding
        (linear_1): Linear(in_features=320, out_features=1280, bias=True)
        (act): SiLU()
        (linear_2): Linear(in_features=1280, out_features=1280, bias=True)
(down_blocks):
        (0): CrossAttnDownBlock2D
        (1): CrossAttnDownBlock2D
        (2): CrossAttnDownBlock2D
        (3): DownBlock2D
(up_blocks):
        (0): UpBlock2D
        (1): CrossAttnUpBlock2D
        (2): CrossAttnUpBlock2D
        (3): CrossAttnUpBlock2D
(mid_block): UNetMidBlock2DCrossAttn
        (attentions):
        (resnets):
(conv_norm_out): GroupNorm(32, 320, eps=1e-05, affine=True)
(conv_act): SiLU()
(conv_out): Conv2d(320, 4, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
```

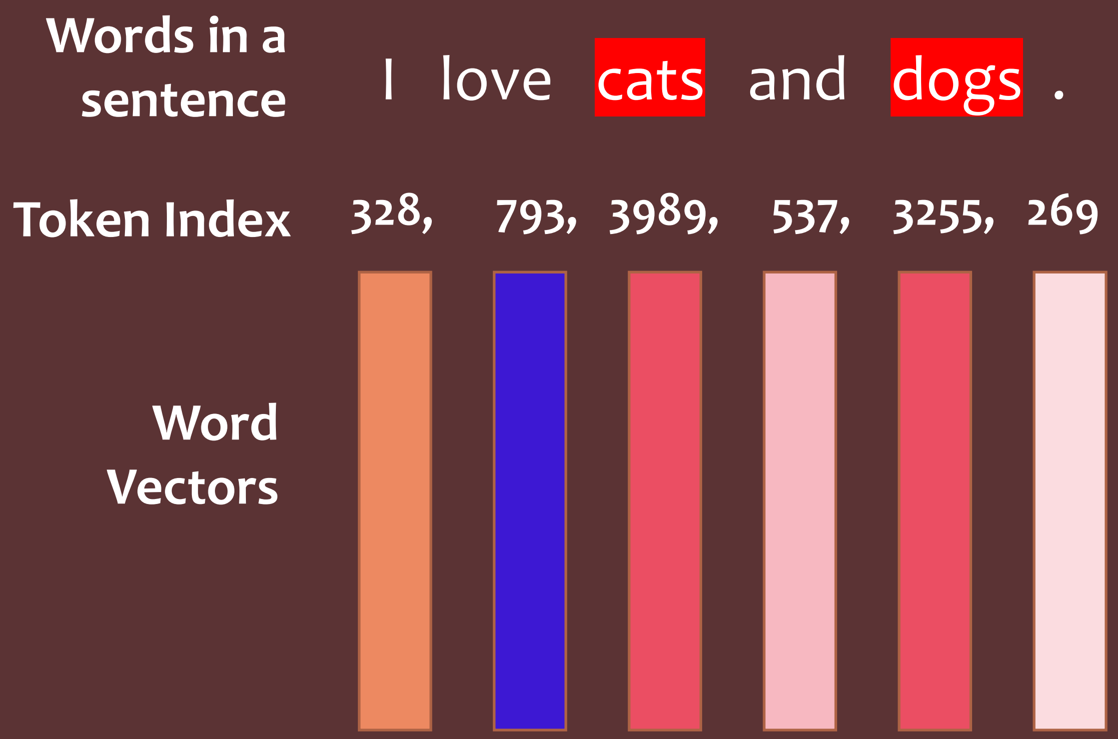
How to understand prompts?

Language / Multimodal Transformer, CLIP!



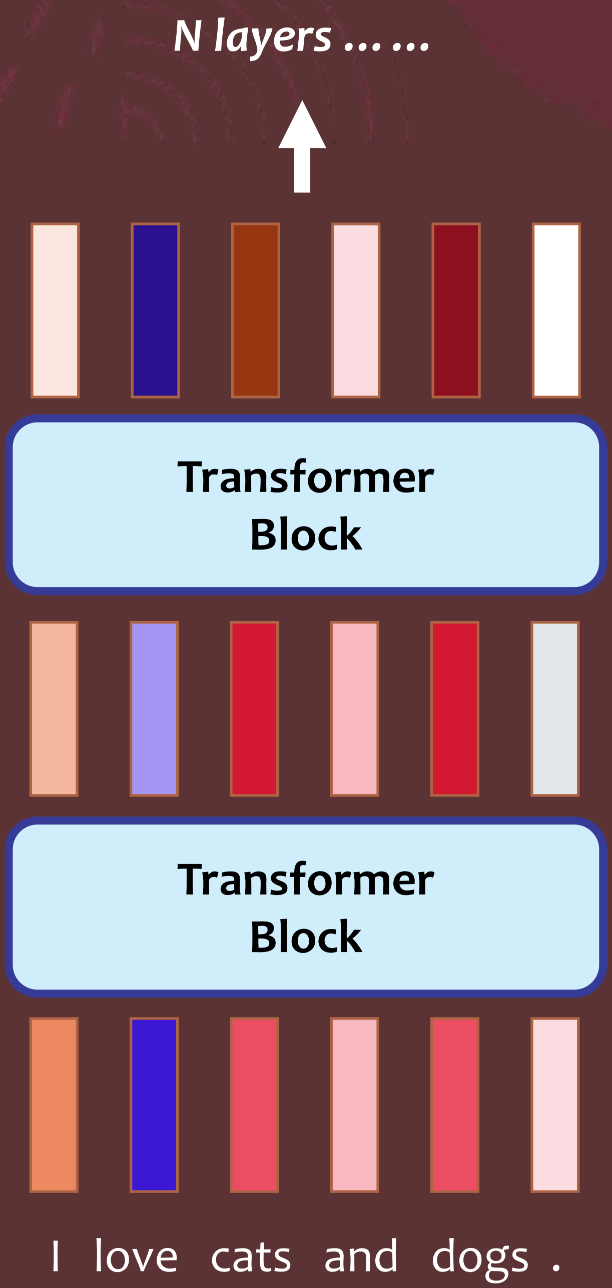
Word as Vectors: Language Model 101

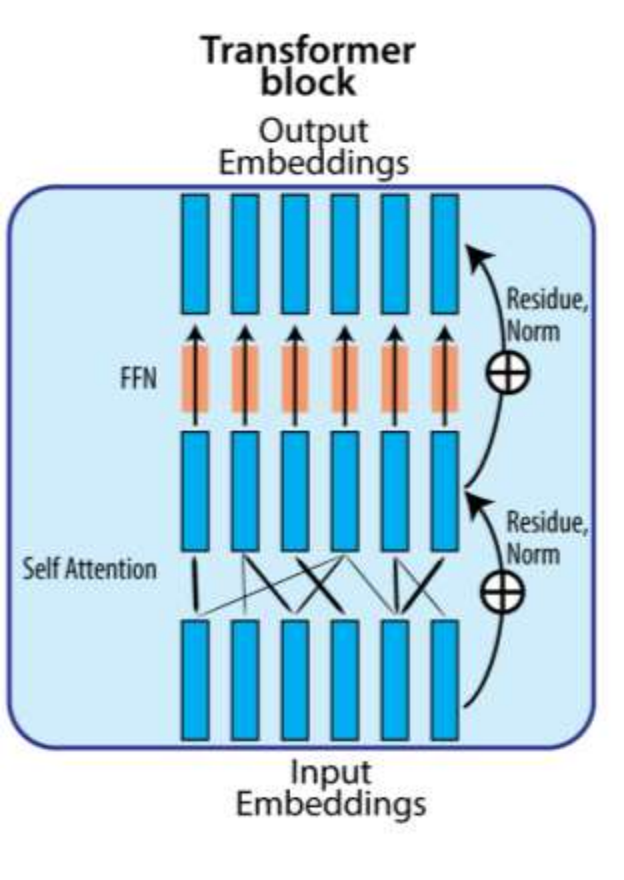
- Unlike pixel, meaning of word are not explicitly in the characters.
- Word can be represented as index in dictionary
  - But index is also meaning less.
- Represent words in a vector space
  - Vector geometry => semantic relation.



Word Vector in Context: RNN / Transformers

- Meaning of word depends on context, not always the same.
  - "I book a ticket to buy that book."
  - Word vectors should depend on context.
- Transformers let each word "absorb" influence from other words to be "contextualized"



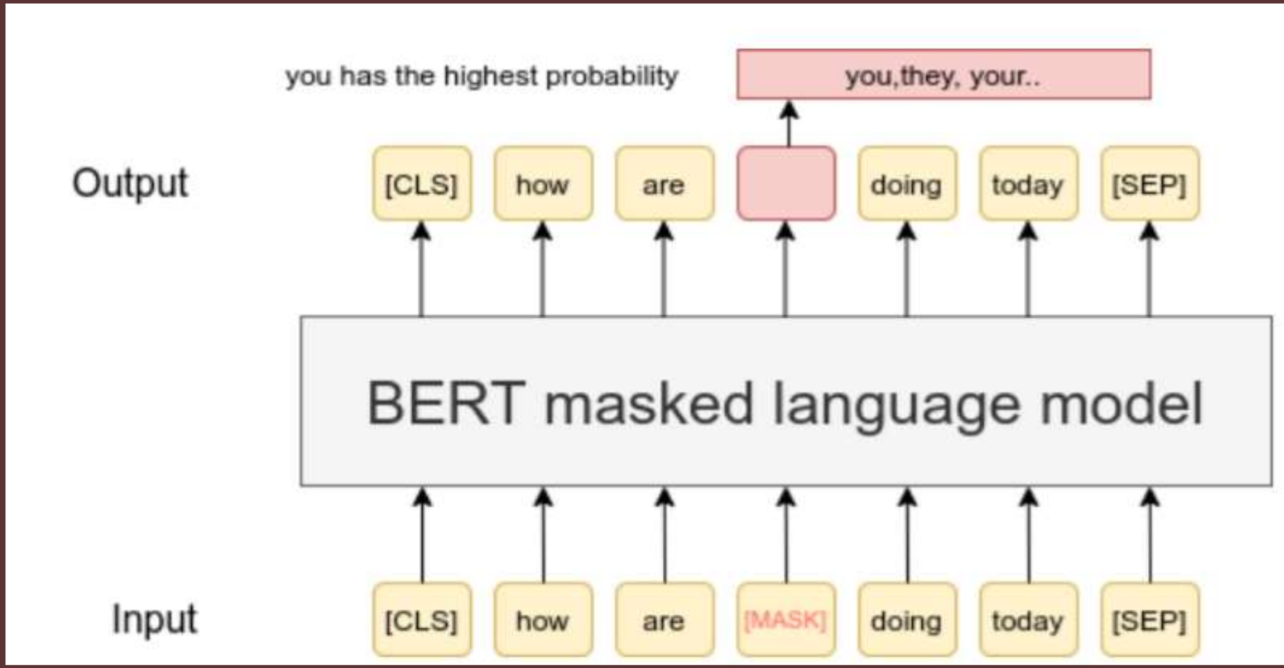


More on attention later…

Learning Word Vectors: GPT & BERT & CLIP

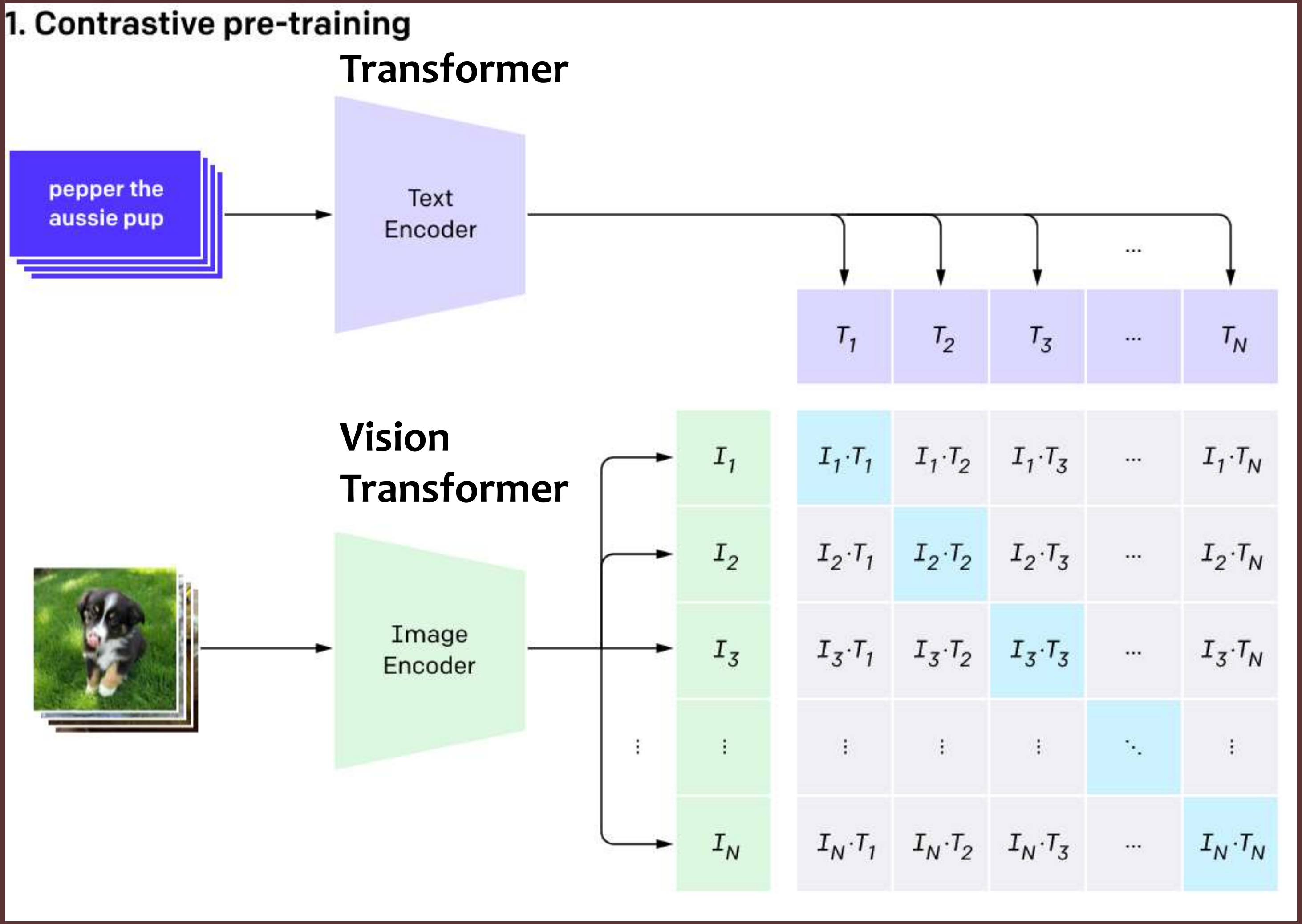
- Self-supervised learning of word representation
  - Predicting missing / next words in a sentence. (BERT, GPT)
  - Contrastive Learning, matching image and text. (CLIP)

Downstream Classifier can decode:
Part of speech, Sentiment, …



MLM — Sentence-Transformers documentation (sbert.net)

Joint Representation for Vision and Language : CLIP



- Learn a joint encoding space for text caption and image
- Maximize representation similarity between an image and its caption.
- Minimize other pairs

CLIP paper 2021

Choice of text encoding

- Encoder in Stable Diffusion: pre-trained CLIP ViT-L/14 text encoder

- Word vector can be randomly initialized and learned online.

- Representing other conditional signals

  - Object categories (e.g. Shark, Trout, etc.):
    - 1 vector per class

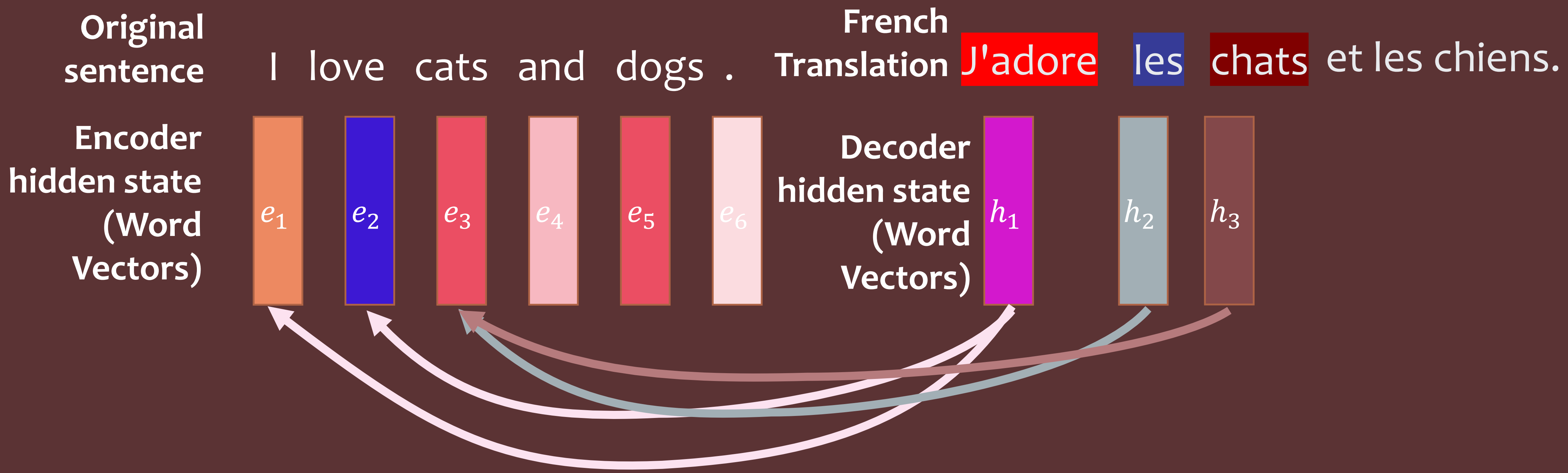
  - Face attributes (e.g. {female, blonde hair, with glasses, … }, {male, short hair, dark skin}):
    - set of vectors, 1 vector per attributes

- Time to be creative!!

How does text affect diffusion?

Incoming **Cross Attention**

Origin of Attention: Machine Translation (Seq2Seq)

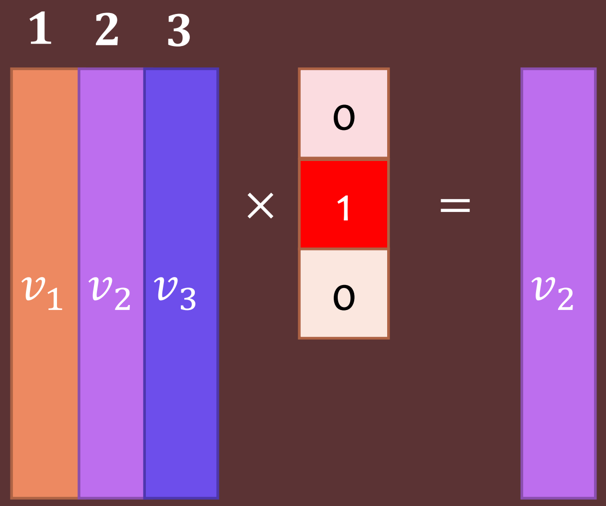


- Use **Attention** to retrieve useful info from a batch of vectors.

From Dictionary to Attention

Dictionary: Hard-indexing

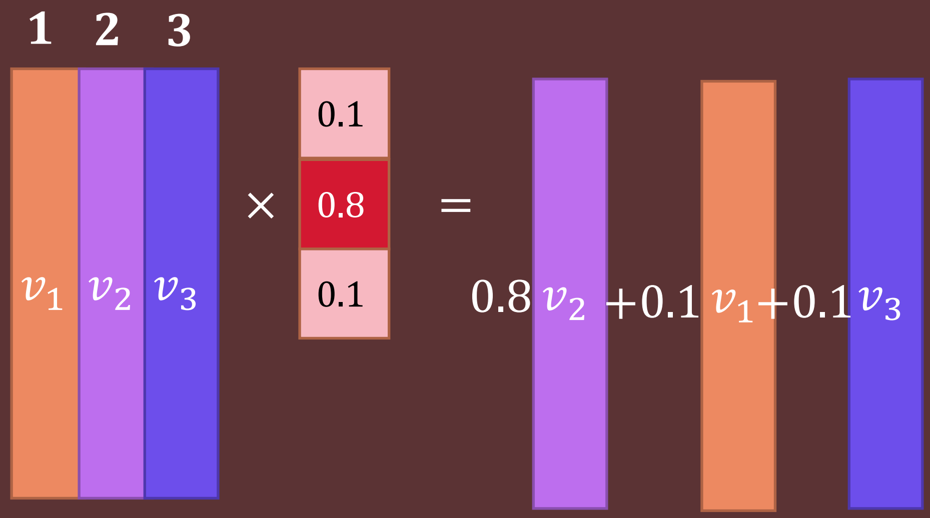
- `dic = $\{1 : v_1, 2 : v_2, 3 : v_3\}$`
  - Keys 1,2,3
  - Values v_1, v_2, v_3
- `dic[2]`
  - Query 2
  - Find 2 in keys
  - Get corresponding value.
- Retrieving values as matrix vector product
  - One hot vector over the keys
  - Matrix vector product



From Dictionary to Attention

Attention: Soft-indexing

- Soft indexing
  - Define an attention distribution a over the keys
  - Matrix vector product.
  - Distribution based on similarity of query and key.



QKV attention

- **Q**uery : what I need (*J'adore* : "I want subject pronoun & verb")
- **K**ey : what the target provide (*I* : "Here is the subject")
- **V**alue : the information to be retrieved (latent related to *Je* or *J'*)

- Linear projection of "word vector"
  - Query $q_i = W_q h_i$
  - Key $k_j = W_k e_j$
  - Value $v_j = W_v e_j$

  - e_j *hidden state of encoder (English, source)*
  - h_i *hidden state of decoder (French, target)*

Attention mechanism

- Compute the inner product (similarity) of key k and query q
- SoftMax the normalized score as attention distribution.

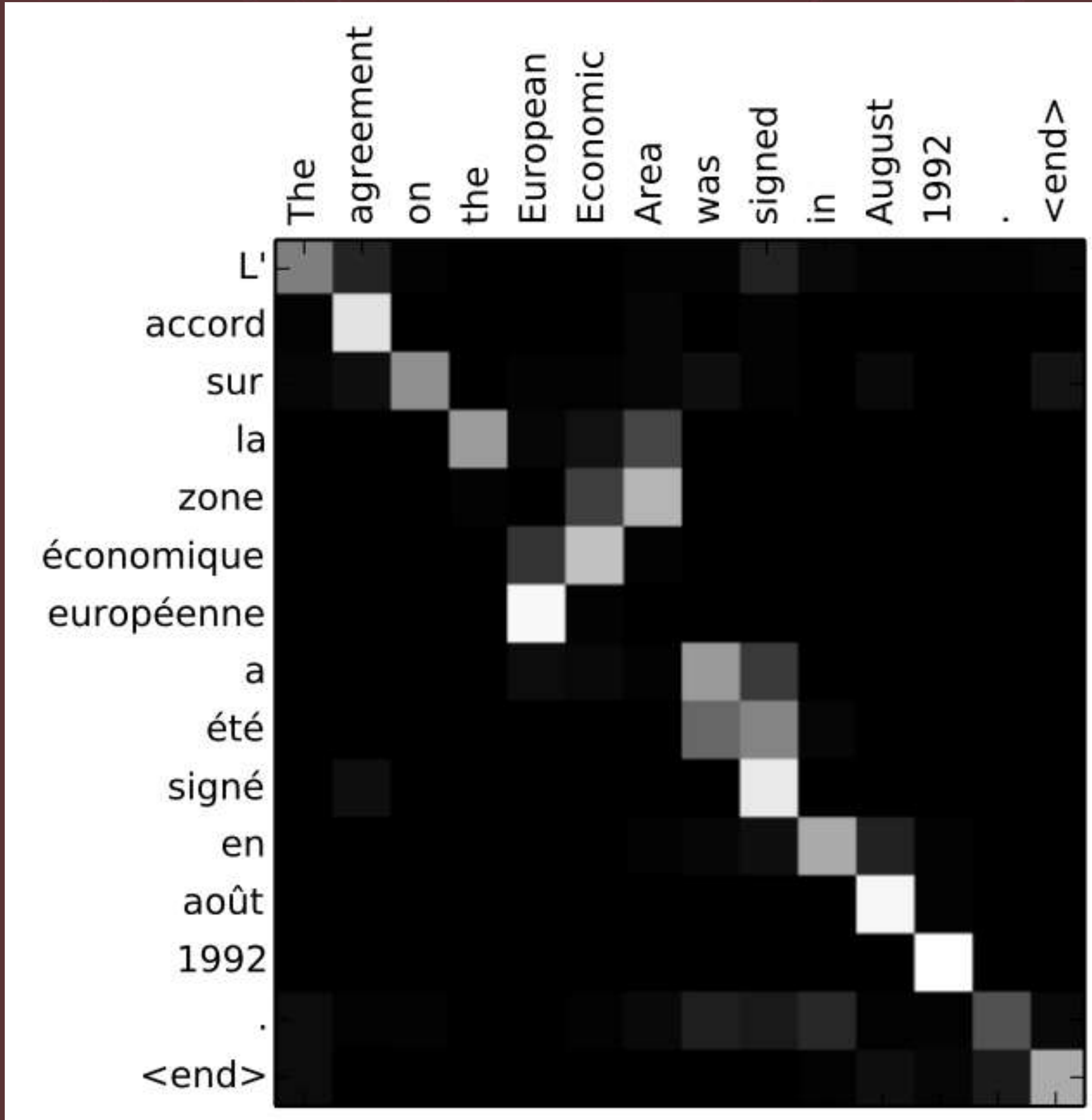
$$a_{ij} = \mathrm{SoftMax}\left(\frac{k_j^T q_i}{\sqrt{len(q)}}\right), \sum_j a_{ij} = 1$$

- Use attention distribution to weighted average values v.

$$c_i = \sum_j a_{ij} v_j$$

Visualizing Attention matrix a_{ij}

- *French 2 English*
- *“Learnt to pay Attention”*
  - *“la zone economique europeenne” -> “the European Economic Area”*
  - *“a ete signe” -> “was signed”*

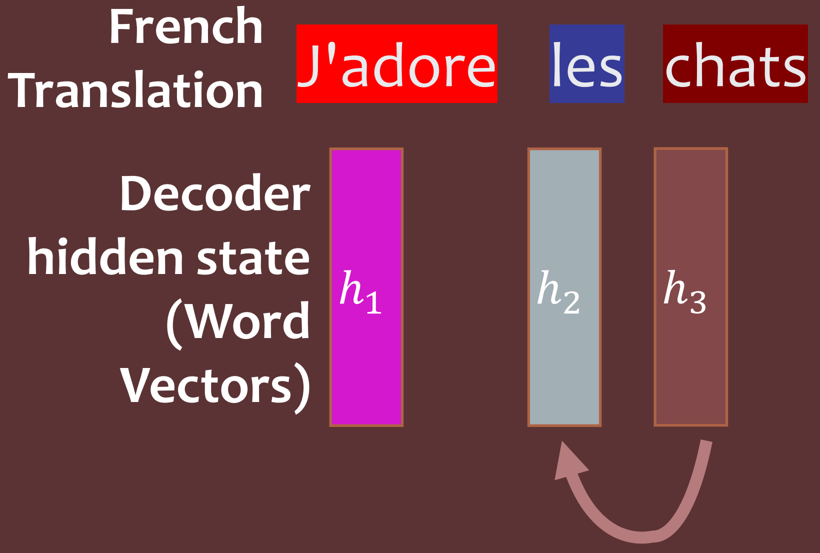


Attention + RNN

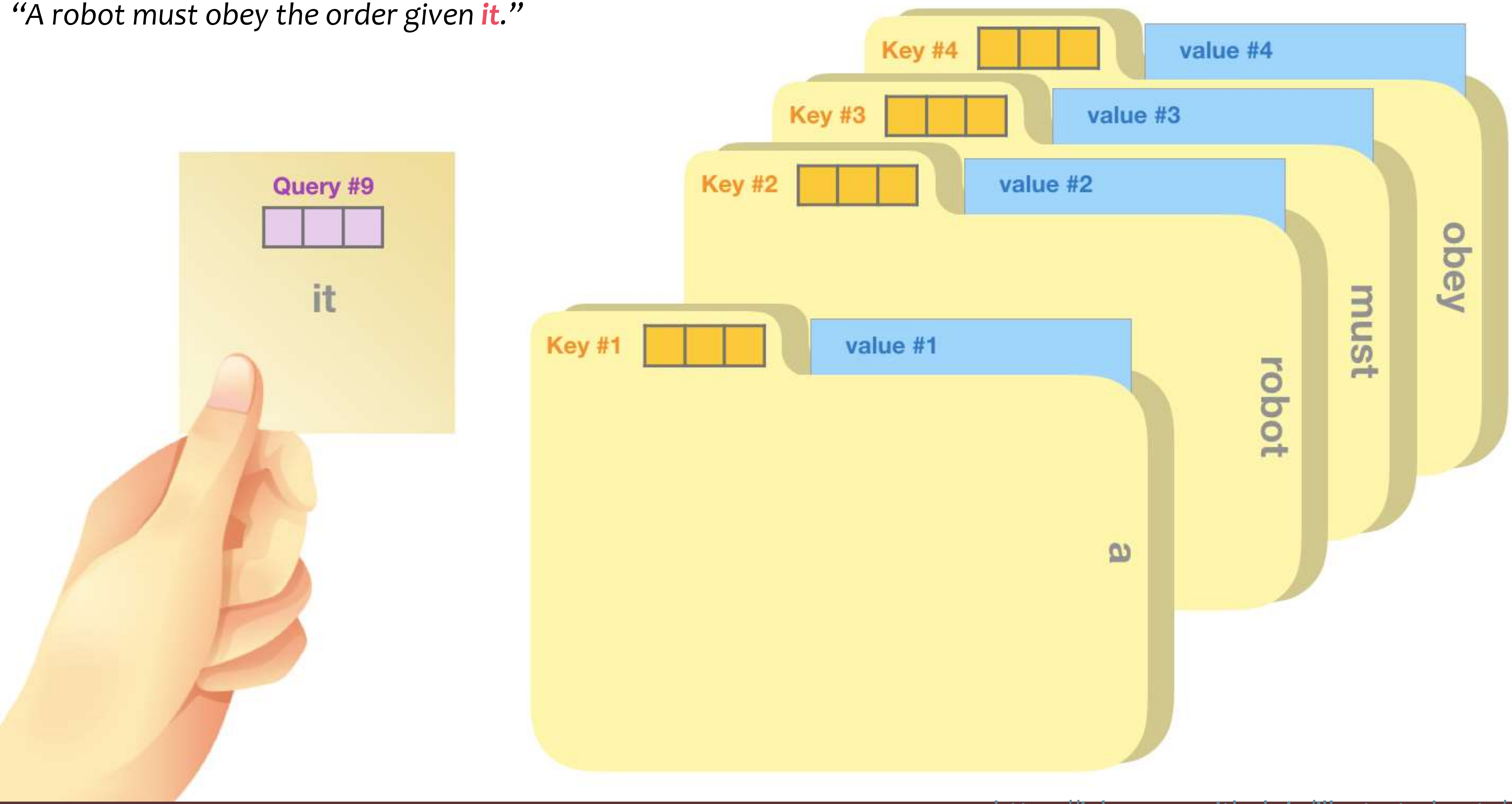
https://jalammar.github.io/visualizing-neural-machine-translation-mechanics-of-seq2seq-models-with-attention/

Cross & Self Attention

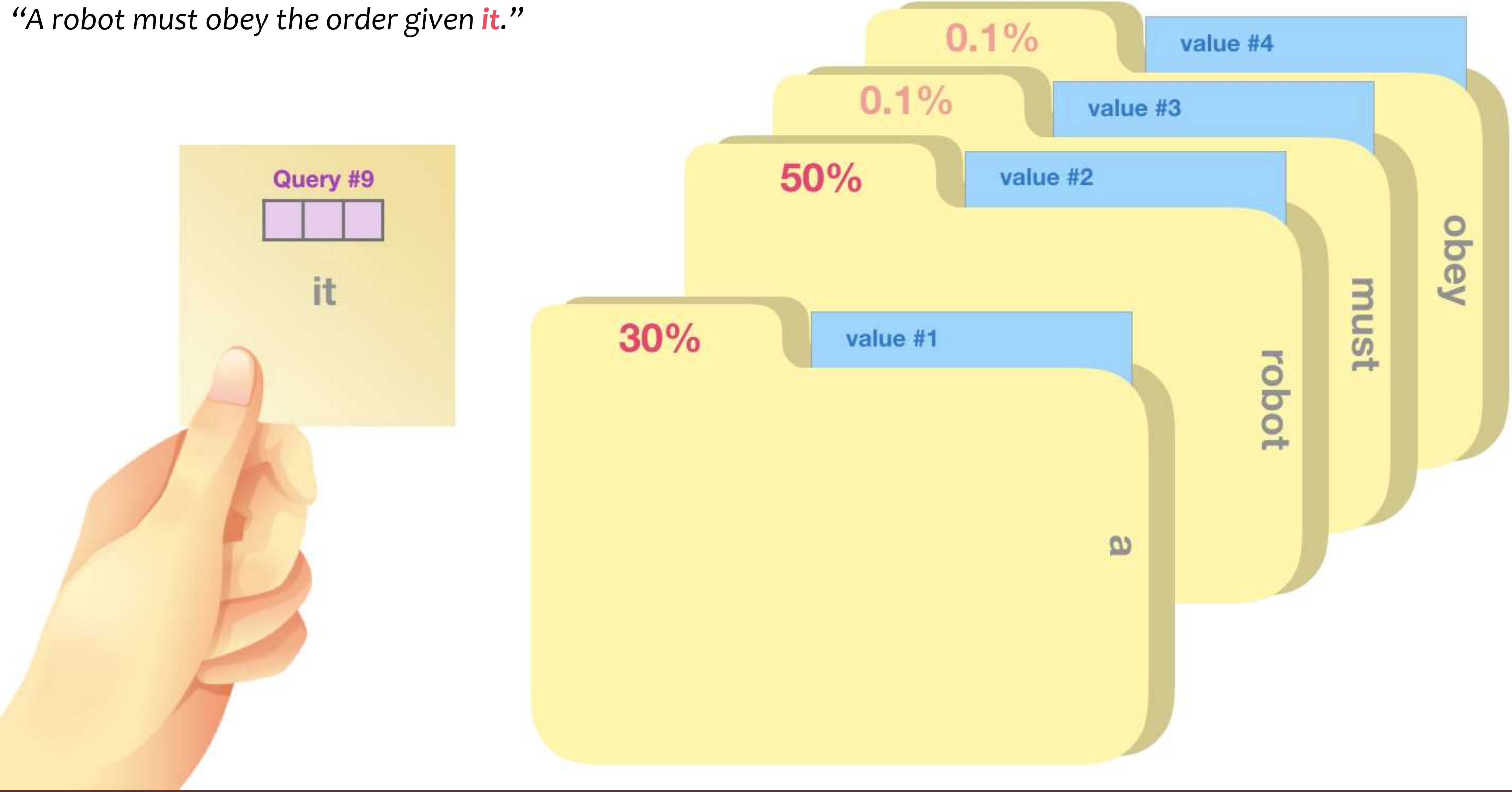
- Cross Attention
  - Tokens in one language pay attention to tokens in **another**.
- Self Attention ($e_i = h_i$)
  - Tokens in a language pay attention to **each other**.



*"A robot must obey the order given **it**."*



“A robot must obey the order given ***it****.”*

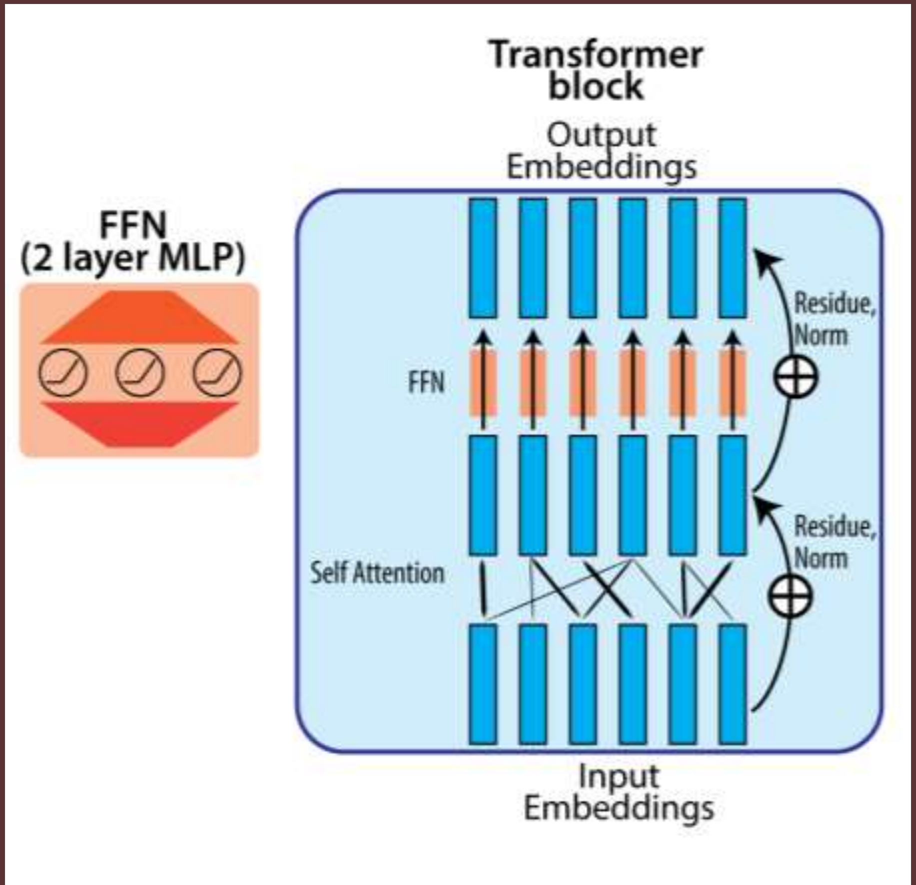


Word	Value vector	Score	Value X Score
<s>		0.001	
a		0.3	
robot		0.5	
must		0.002	
obey		0.001	
the		0.0003	
orders		0.005	
given		0.002	
it		0.19	
		Sum:	

https://jalammar.github.io/illustrated-gpt2/

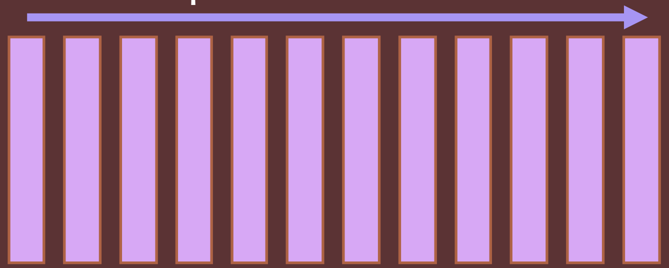
Note: Feed Forward network

- Attention is usually followed by a 2-layer MLP and Normalization
  - Learn nonlinear transform.



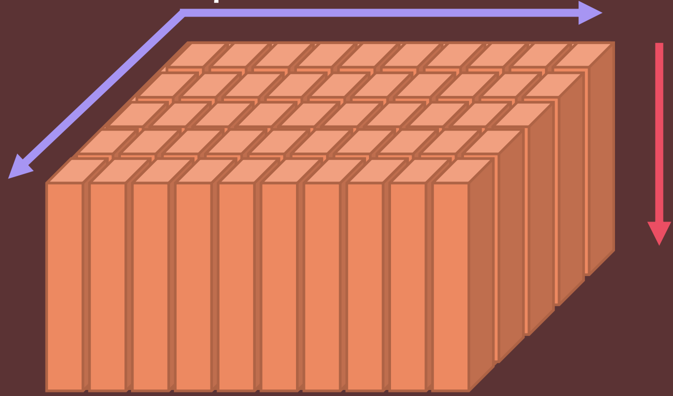
Text2Image as translation

Source language: Words

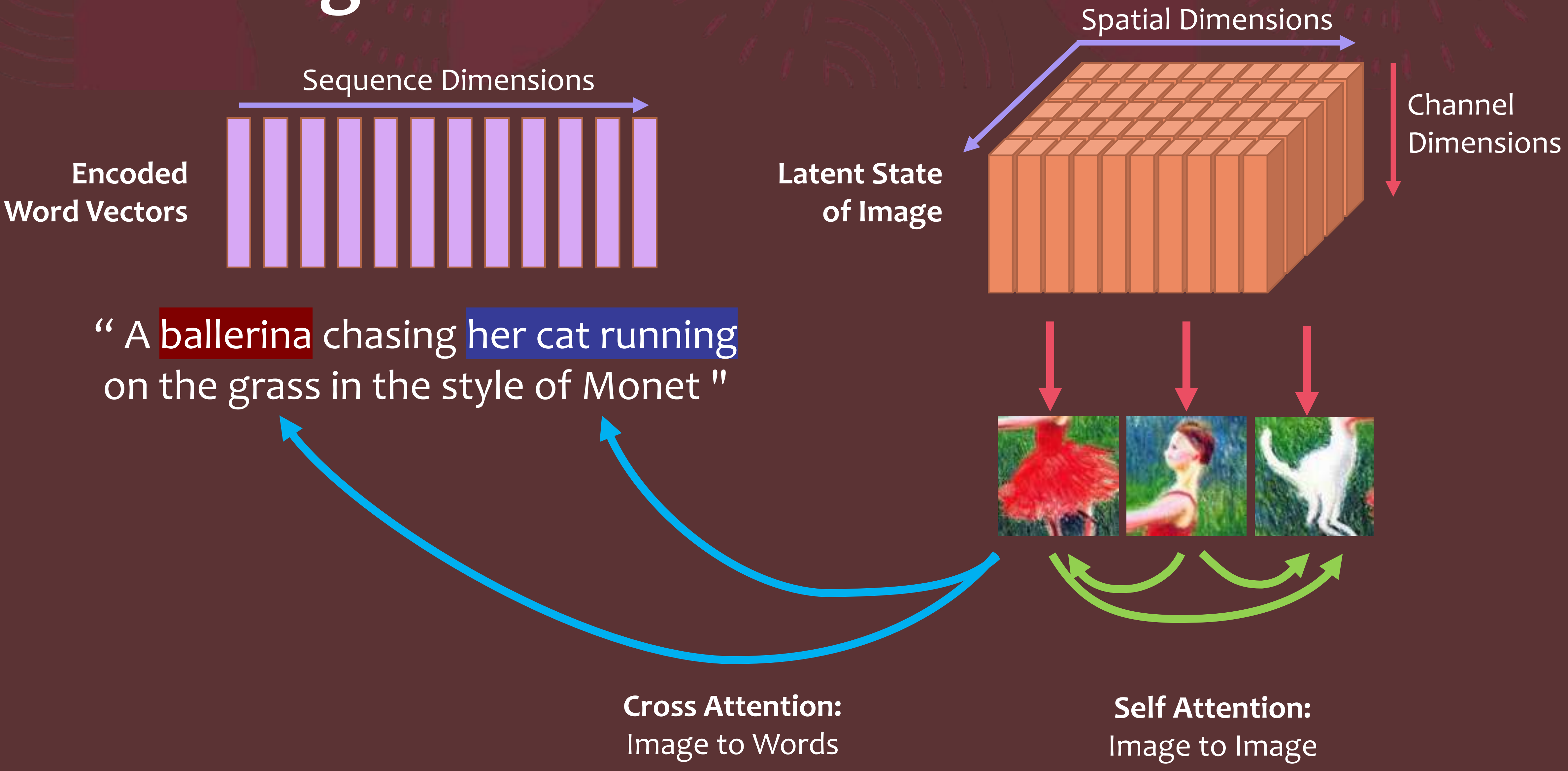


“ A ballerina chasing her cat running on the grass in the style of Monet "

Target language: Images

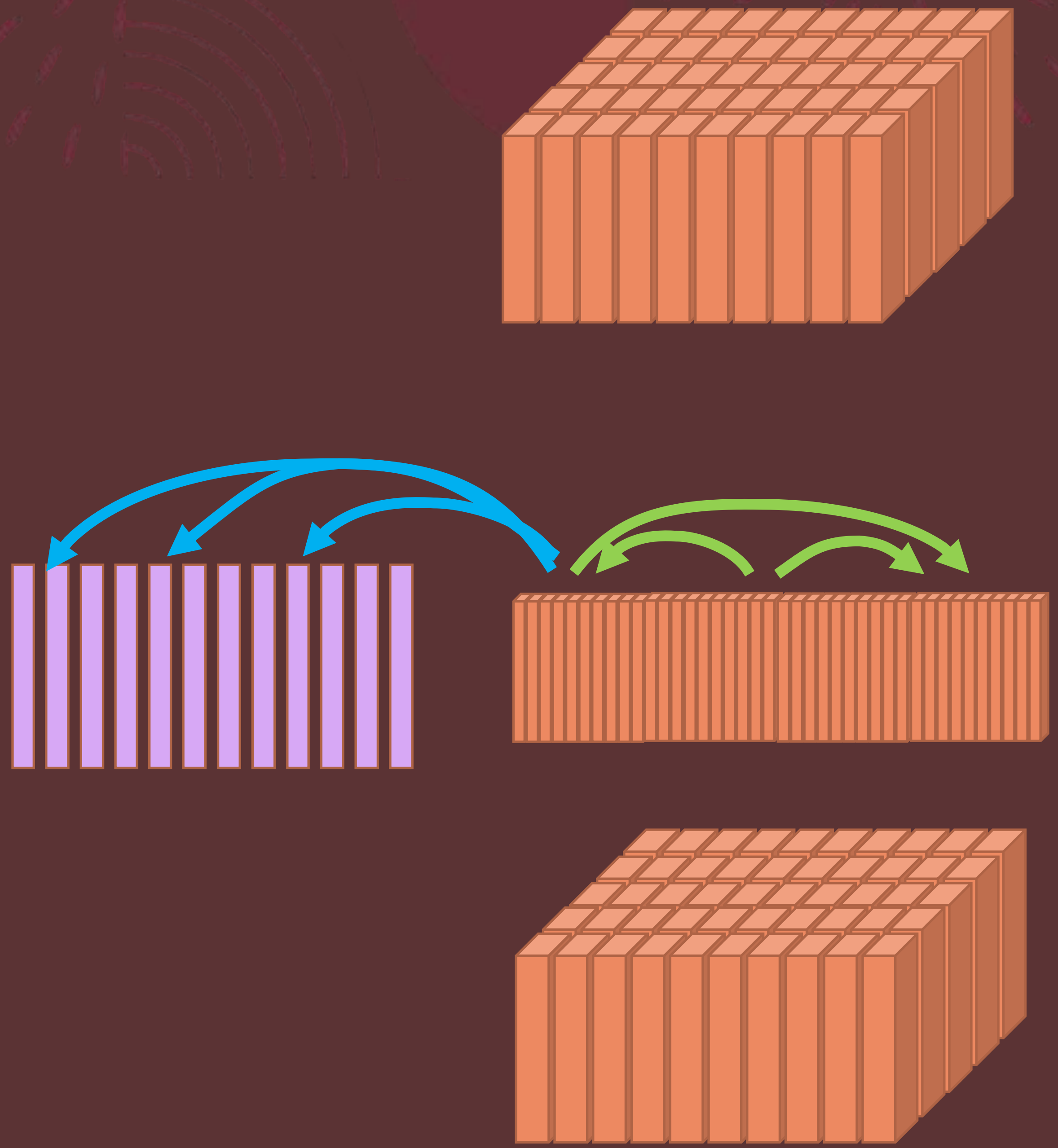


Text2Image as translation



Spatial Transformer

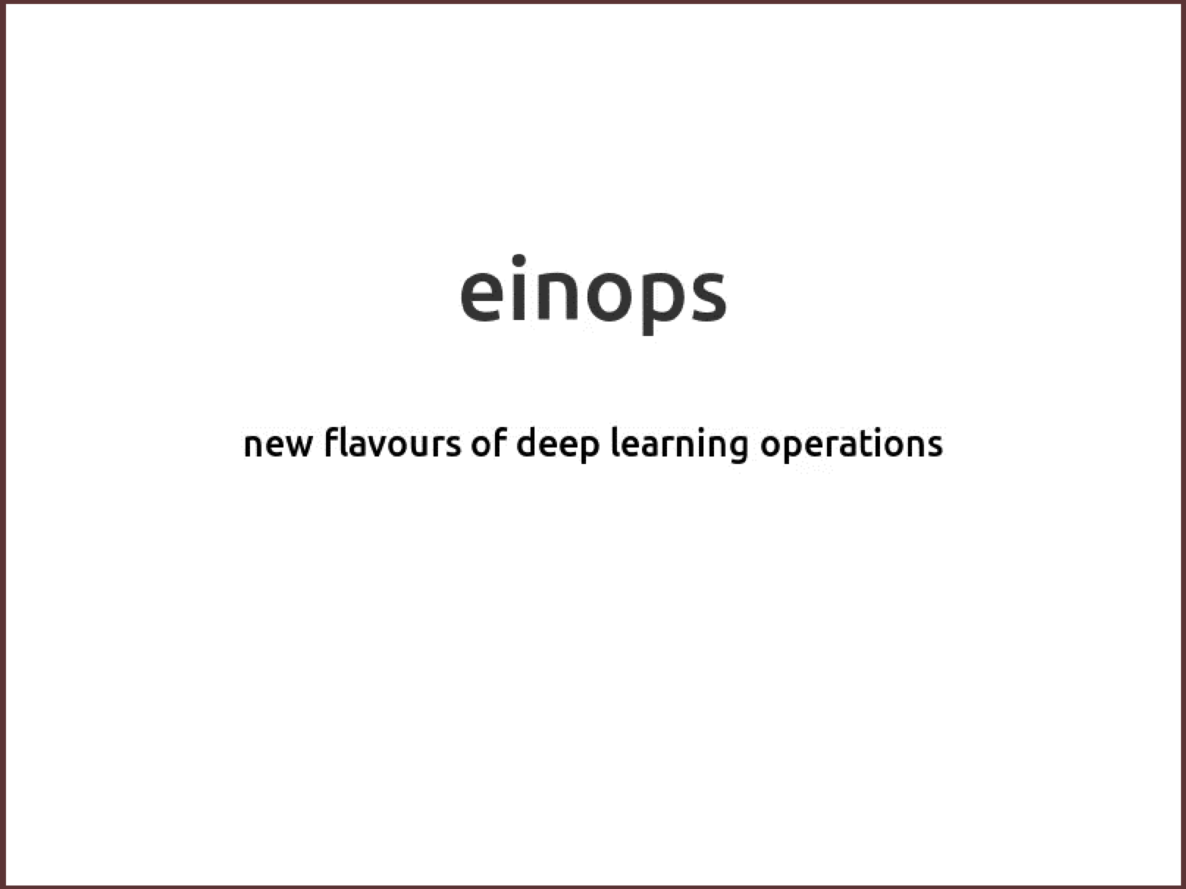
- Rearrange spatial tensor to sequence.
- Cross Attention
- Self Attention
- FFN
- Rearrange back to spatial tensor (same shape)



Tips: Implementing attention `einops` lib

- `einops.rearrange` function
  - Shift order of axes
  - Split / combine dimension.

- `torch.einsum` function
  - Multiply & sum tensors along axes.



UNet = Giant Sandwich of Spatial transformer + ResBlock (Conv layer)

- Down blocks
  - ResBlock
  - SpatialTransformer
  - ResBlock
  - SpatialTransformer
  - DownSample

  - ResBlock
  - SpatialTransformer
  - ResBlock
  - SpatialTransformer
  - DownSample

  - ResBlock
  - SpatialTransformer
  - ResBlock
  - SpatialTransformer
  - DownSample

  - ResBlock
  - ResBlock

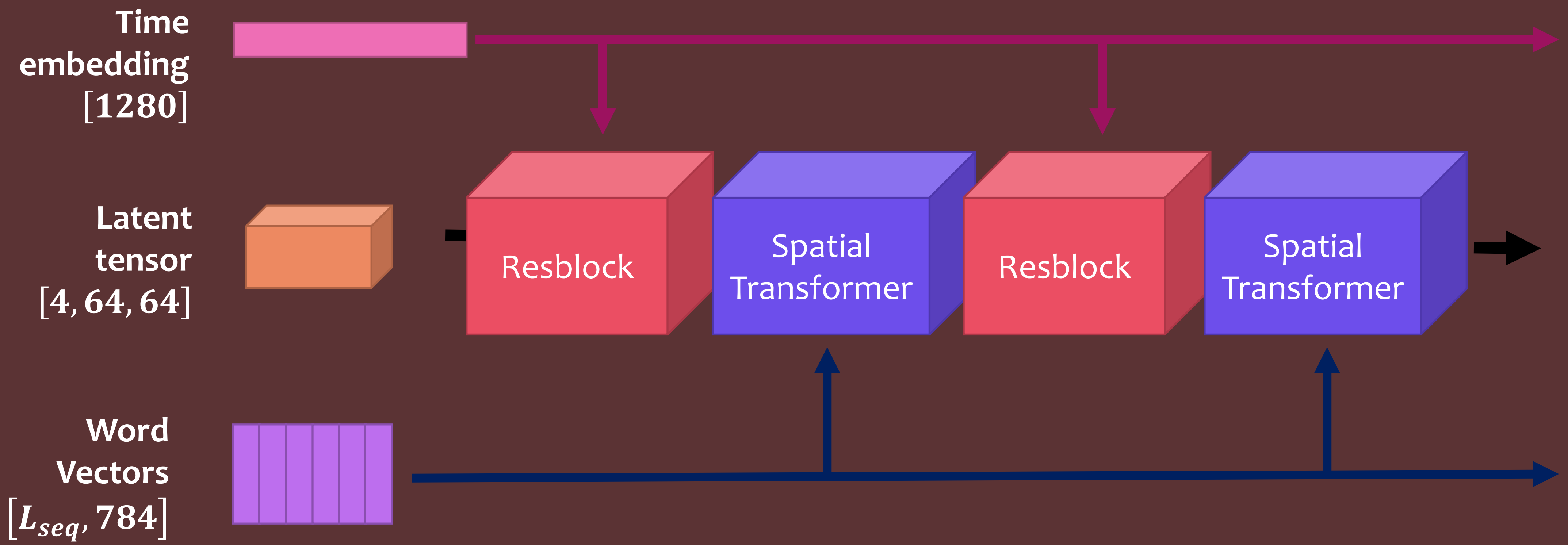
- Up blocks
  - ResBlock
  - ResBlock
  - ResBlock
  - UpSample

  - ResBlock
  - SpatialTransformer
  - ResBlock
  - SpatialTransformer
  - ResBlock
  - SpatialTransformer
  - UpSample

  - ResBlock
  - SpatialTransformer
  - ResBlock
  - SpatialTransformer
  - ResBlock
  - SpatialTransformer
  - UpSample

  - ResBlock
  - SpatialTransformer
  - ResBlock
  - SpatialTransformer
  - ResBlock
  - SpatialTransformer

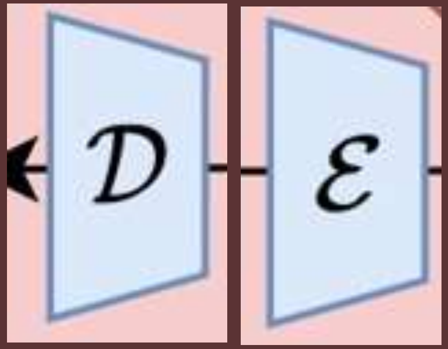
Spatial transformer + ResBlock (Conv layer)



- Alternating Time and Word Modulation
- Alternating Local and Nonlocal operation

Diffusion in Latent Space

Adding in AutoEncoder



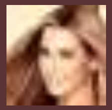
Rombach, R., Blattmann, A., Lorenz, D., Esser, P., & Ommer, B. (2022). High-resolution image synthesis with latent diffusion models, *CVPR*

Diffusion in latent space

- Motivation:
  - Natural images are high dimensional
  - but have many redundant details that could be compressed / statistically filled out
- Division of labor
  - Diffusion model -> Generate low resolution sketch
  - AutoEncoder -> Fill out high resolution details
- Train a VAE model to compress images into latent space.
  - $x \to z \to x$
- Train diffusion models in latent space of z.

DownSampling

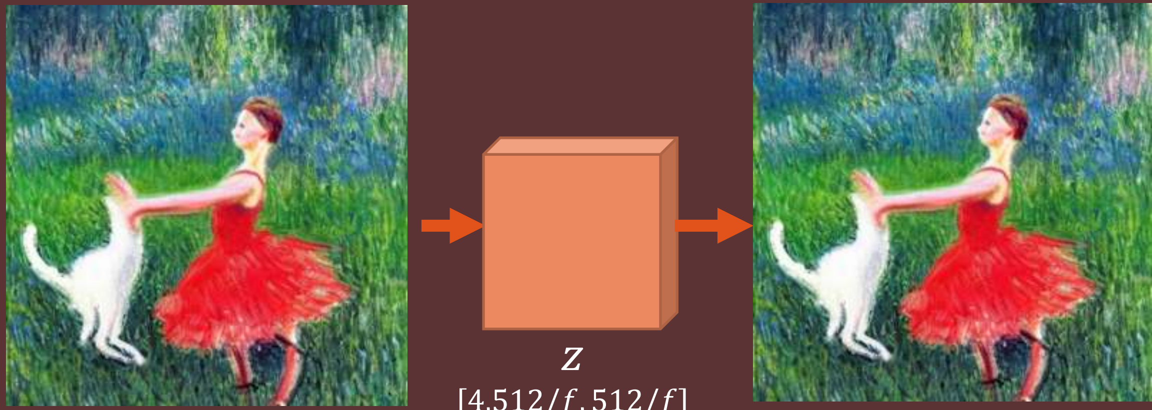
32 pix



$d = 2352$

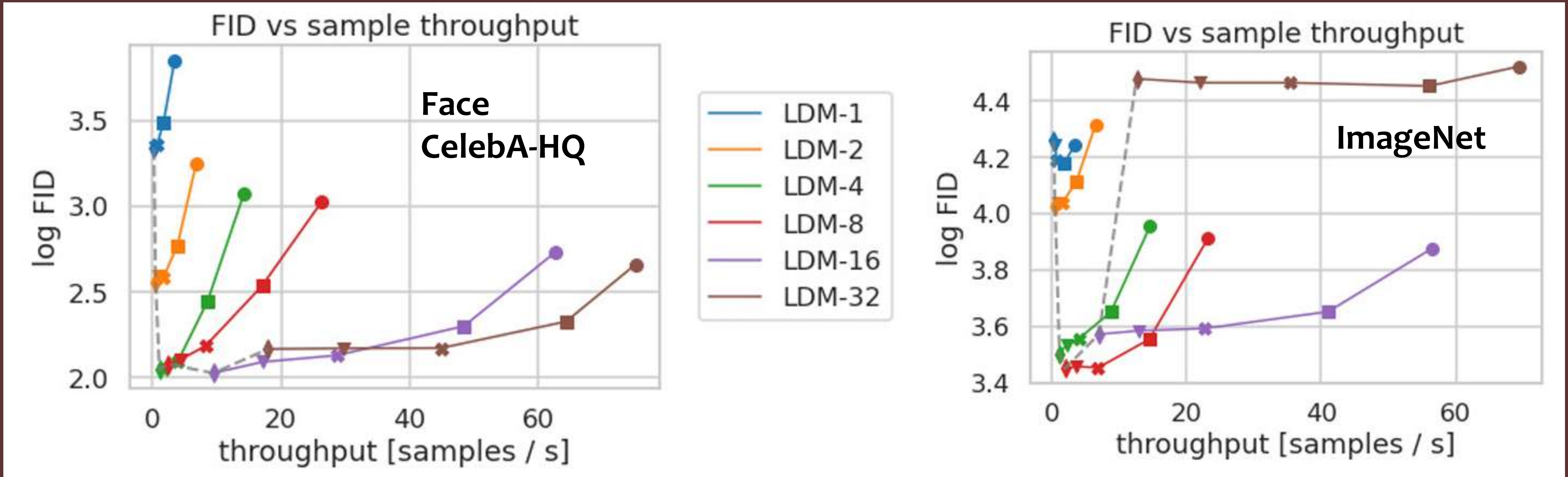
180 pix

$d = 97200$



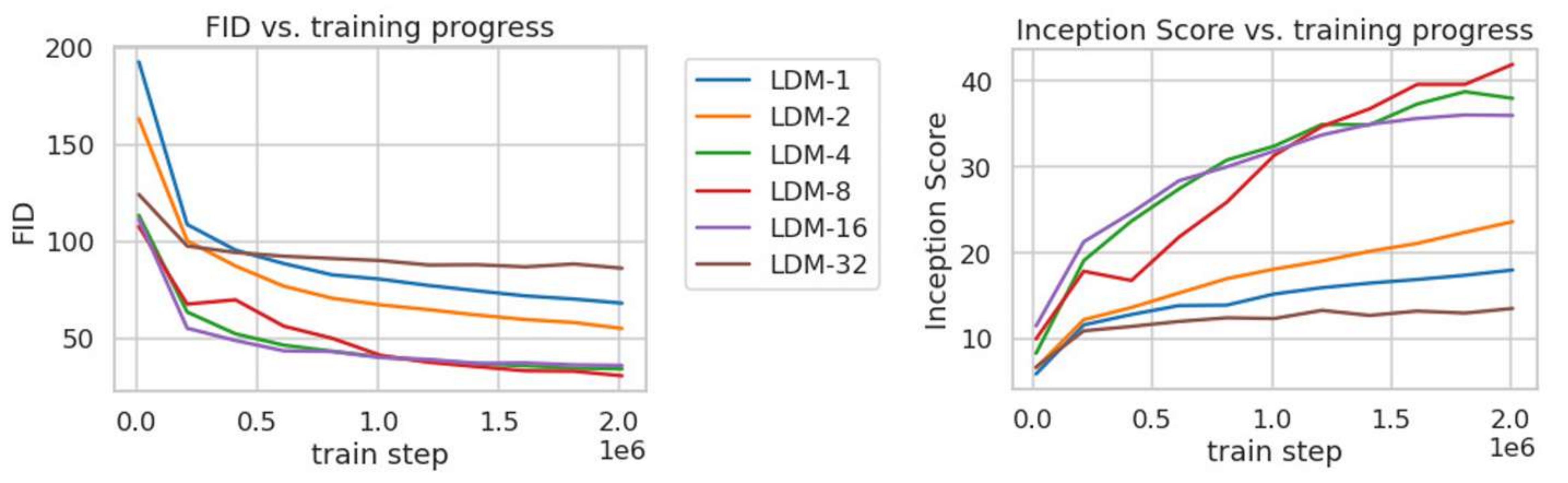
Spatial Compression Tradeoff

- LDM-$\{f\}$. f = Spatial downsampling factor
  - Higher f leads to faster sampling, with degraded image quality (FID ↑)
  - Fewer sampling steps leads to faster sampling, with lower quality (FID ↑)



Spatial Compression Tradeoff

- LDM-$\{f\}$. f = Spatial downsampling factor
  - Too little compression $f = 1{,}2$ or too much compression $f = 32$, makes diffusion hard to train.



Details in Stable Diffusion

- In stable diffusion, spatial downsampling $f = 8$
  - x is $(3, 512, 512)$ image tensor
  - z is $(4, 64, 64)$ latent tensor

Regularizing the Latent Space

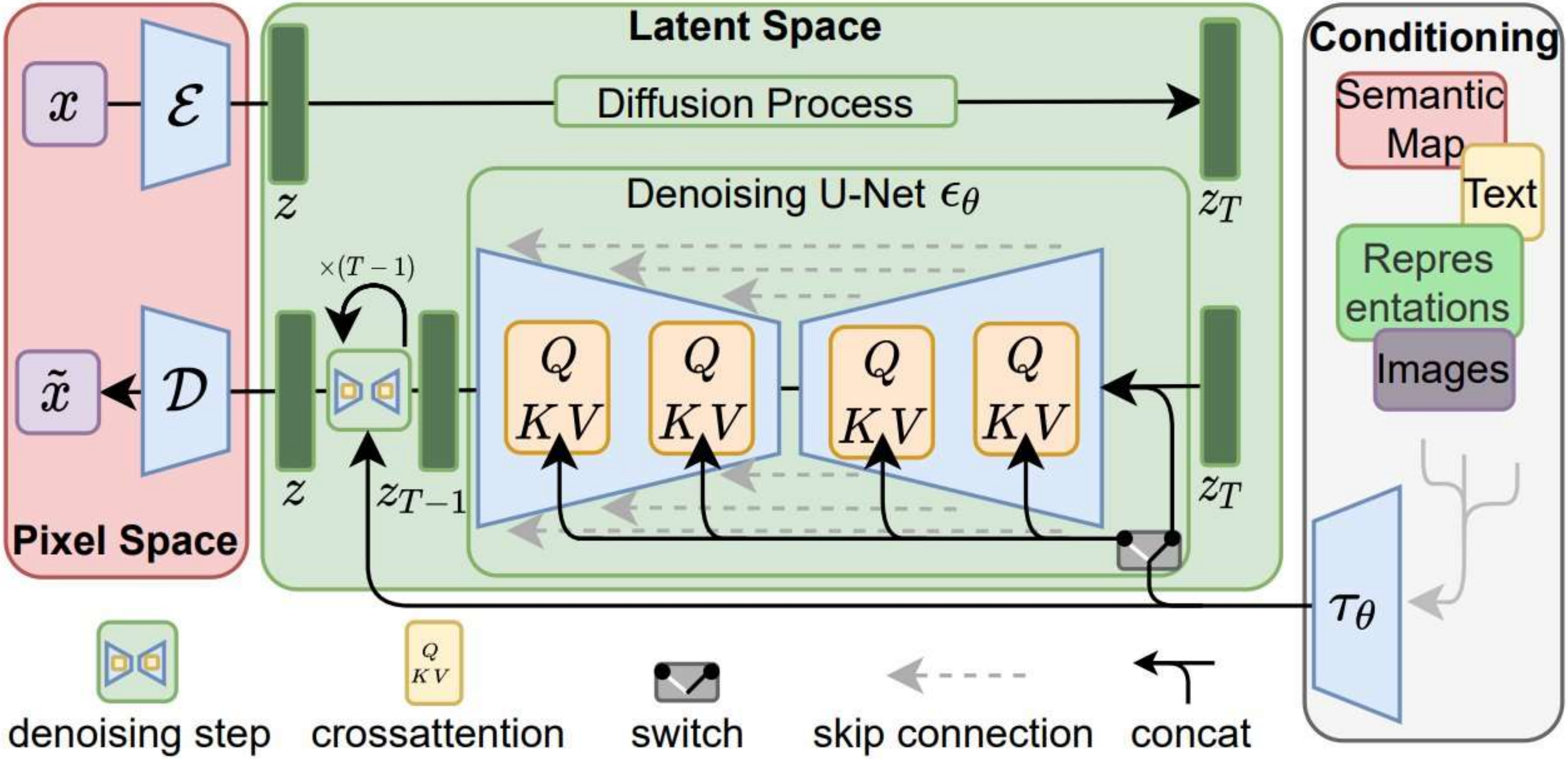
- KL regularizer
  - Similar to VAE, make latent distribution like Gaussian distribution.
- VQ regularizer
  - Make the latent representation quantized to be a set of discrete tokens.

Let the GPUs roar!

Training data & details.

Large Data Training

- SD is trained on ~ 2 Billion image – caption (English) pairs.
  - Scraped from web, filtered by CLIP.
  - https://laion.ai/blog/laion-5b/

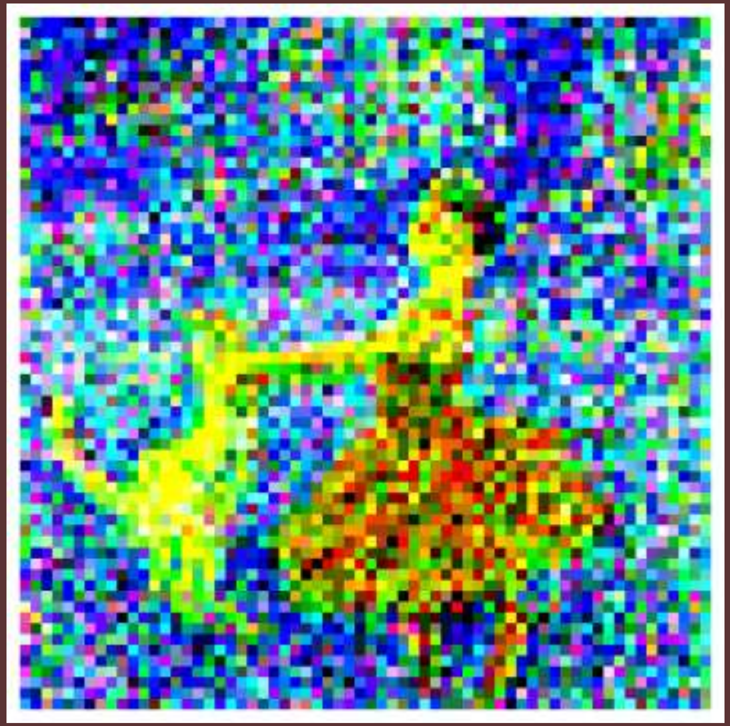


Latent Space
Conditioning
Semantic Map
Text
Repres entations
Images
Diffusion Process
x
$\mathcal{E}$
z
z_T
Denoising U-Net ϵ_θ
$\times(T-1)$
$\tilde{x}$
$\mathcal{D}$
z_{T-1}
Q
KV
τ_θ
Pixel Space
denoising step
crossattention
switch
skip connection
concat

Diffusion Process Visualized

Meaning of latent space

- Latent state contains a “sketch version” of the image.



$z[0{:}3,:,:]$