Personalizing Image Generation : Fine-Tuning Diffusion Models

Thesis project

June 2023

Metyis

Content

1 Introduction & background

2 Objective

3 Usecase & challenges

4 Technical details – Model architecture

5 Approach

6 Results

7 Q&A

8 Appendix

1 Introduction & background

What is Generative AI?

Generative AI refers to a category of artificial intelligence (AI) algorithms that generates outcomes similar to their training data, from which they can interpolate according to the user input.

It describes algorithms (such as ChatGPT) that can be used to create new content, including audio, code, images, text, simulations, and videos:

Images: Generative AI can create new images text descriptions

Text: Generative AI can be to answer user questions, write code and generate summaries and articles.

Audio: Generative AI can generate new music tracks, sound effects, and even voice acting.

An astronaut riding a horse in photorealistic style.

Midjourney

STABLE DIFFUSION

OpenAI DALL·E 2

What are Diffusion Models?

Diffusion Models are generative models inspired by the **physical Diffusion process***.

They work by destroying training data through the successive addition of random noise, and then learning to recover the data by reversing this noising process.

After training, the generator can transform random noise in the picture you described!

**gradual movement/dispersion of concentration, like a drop of paint dissolving in water*

2 Objective

Standardization of diffusion models with proper experimentation on various fine-tuning methods

3 Usecase & challenges

Usecases

From Text to Image | Additional applications

Marketing
Generate effective dynamic content or Ad creatives for campaigns

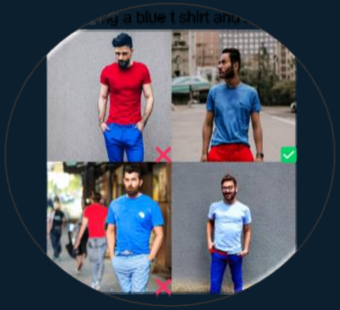
Ecommerce/Retail
Generate designs for new products, catalogue & alternate angle generation

Inspirational Designs
Generate inspirational designs for product design team e.g., mood board creator

In/out Image painting
Extend the creativity by editing visual elements in the same style, or taking a story in new directions

Video Generation
Generate coherent and higher quality videos from text

Challenges



Failure to process the text input

Poor performance for specific entities (e.g., text)

Faces and people may not be generated properly

‘een nijlpaard’

May not work well with non-English prompts

Model can be lossy and takes relatively long time

4

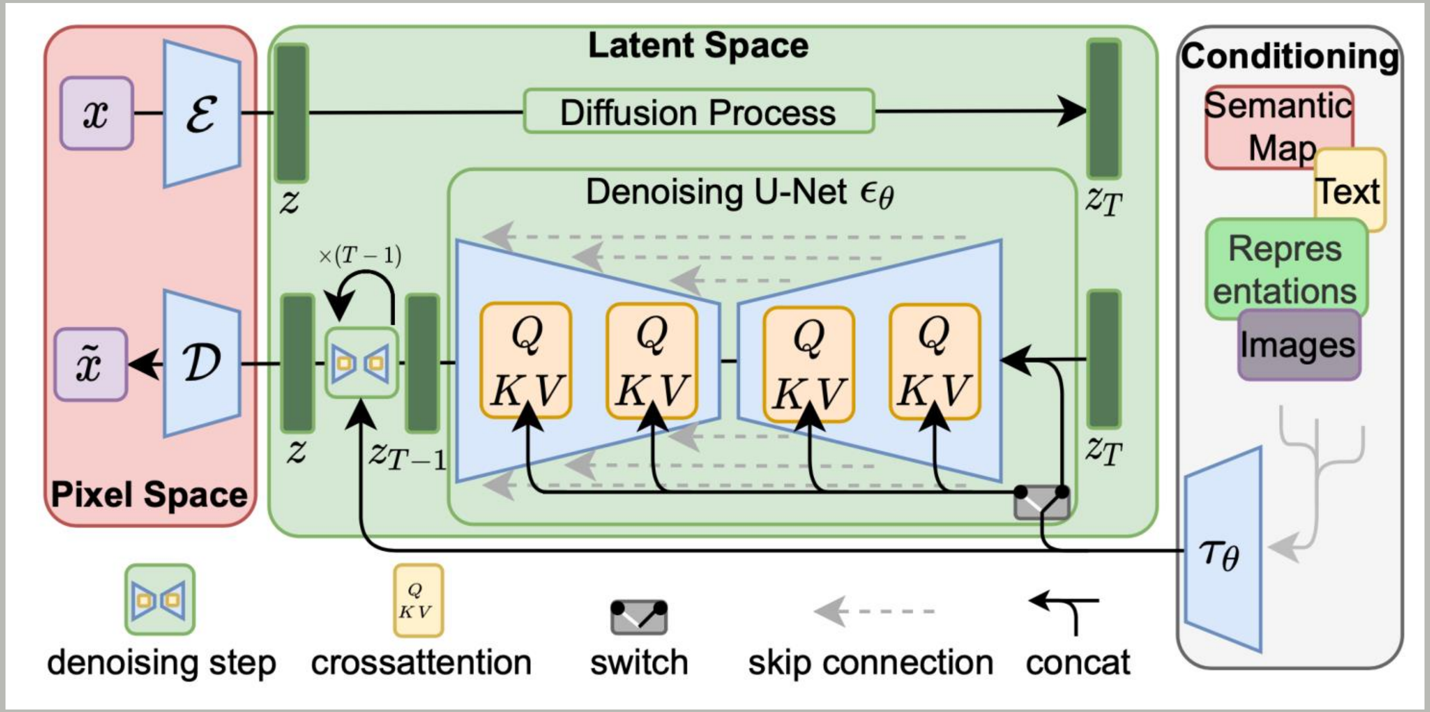
Technical details – Model architecture

Stable Diffusion

Open-source Latent Diffusion model

Input →

Output ←



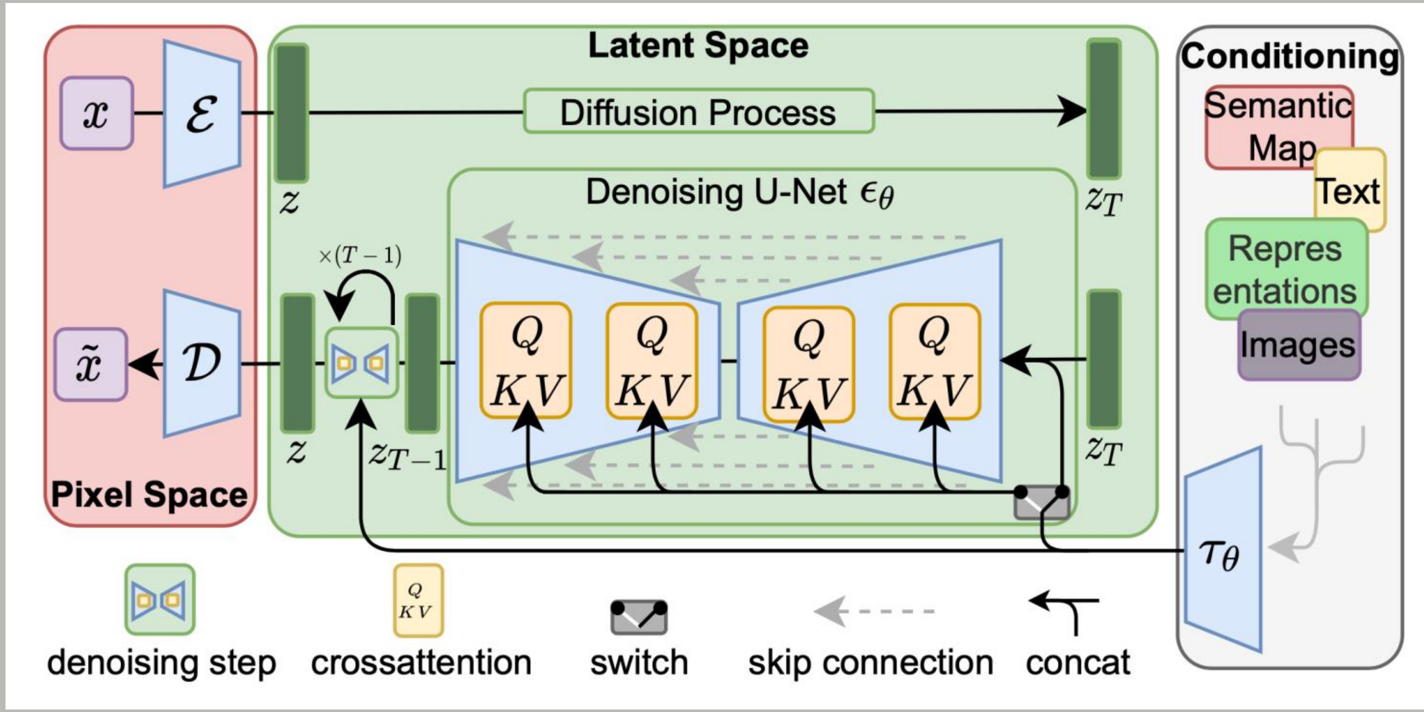
Metyis

Stable Diffusion

Open-source Latent Diffusion model

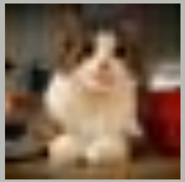
Input →

Output ←



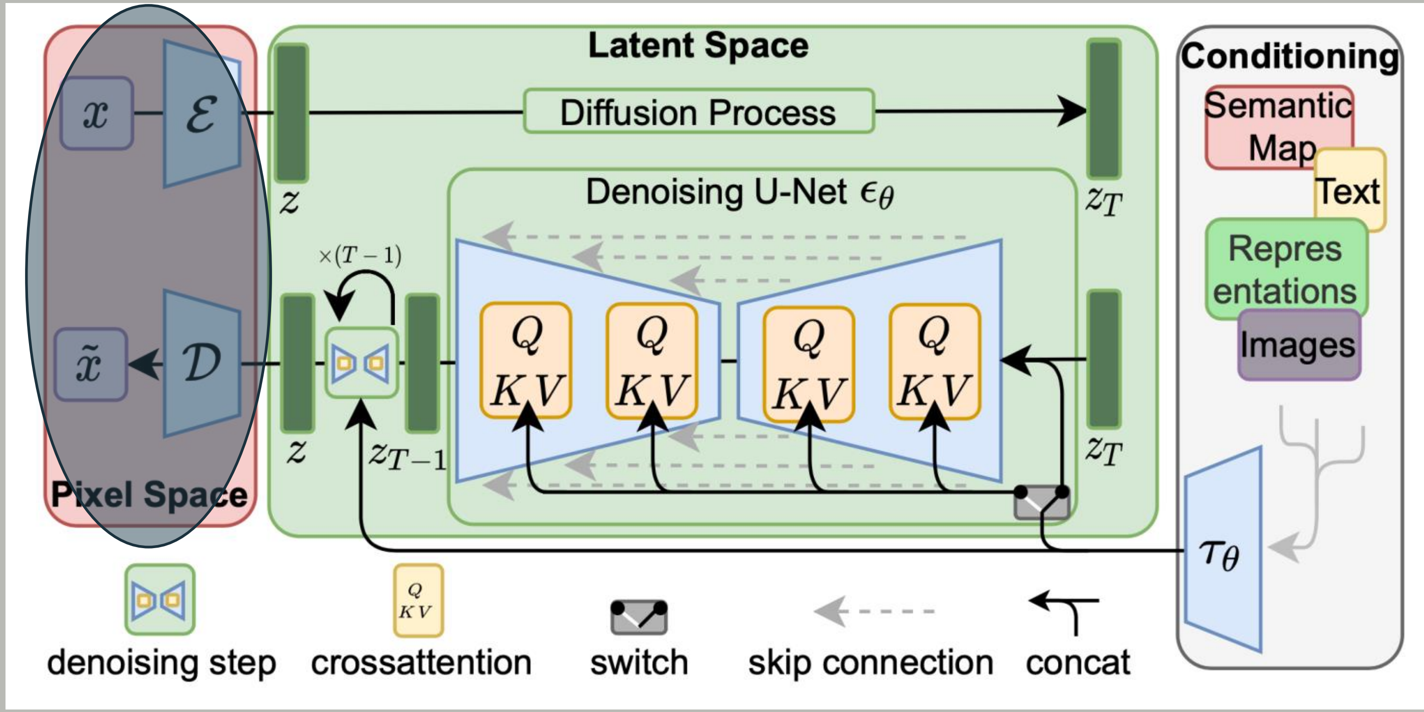
Metyis

Encoder / Decoder



Input

Output



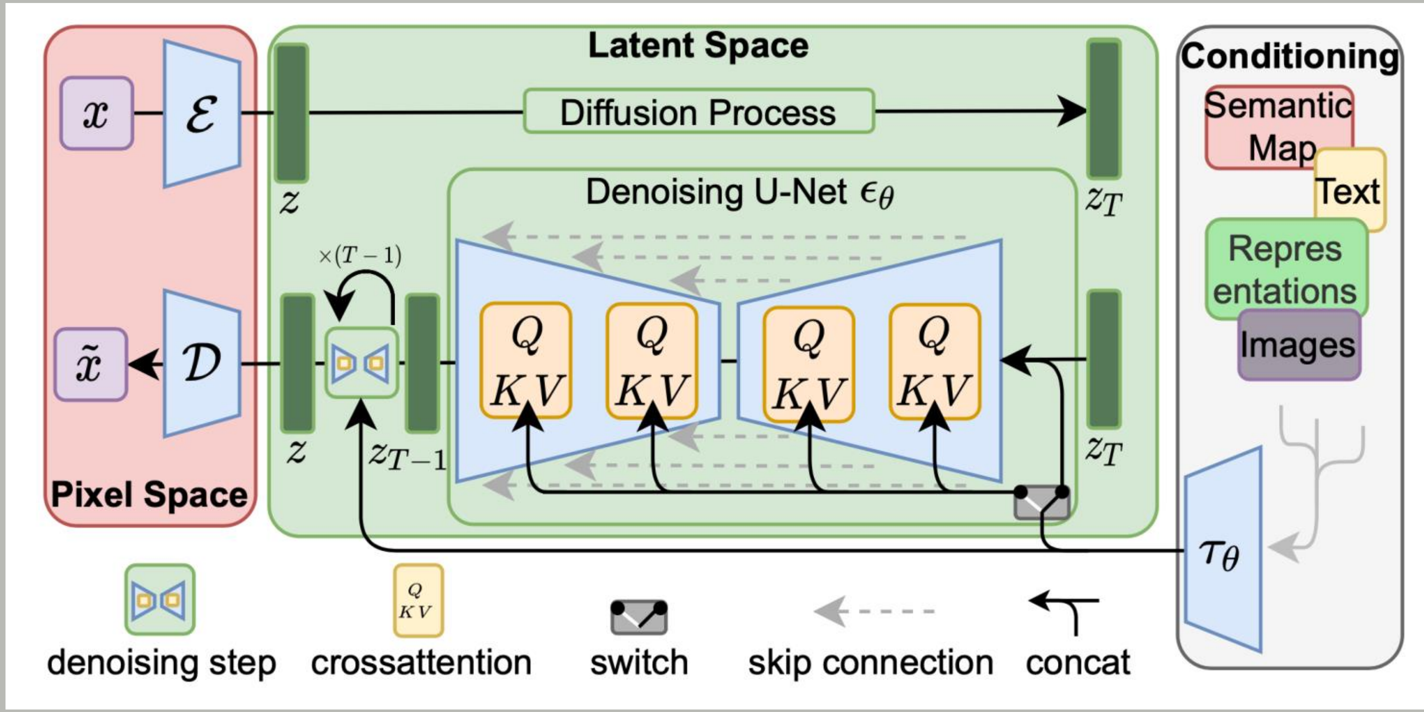
The encoder compresses the image into a lower dimensional latent space to allow faster computing and better image processing

Metyis

Noising process

Input

Output

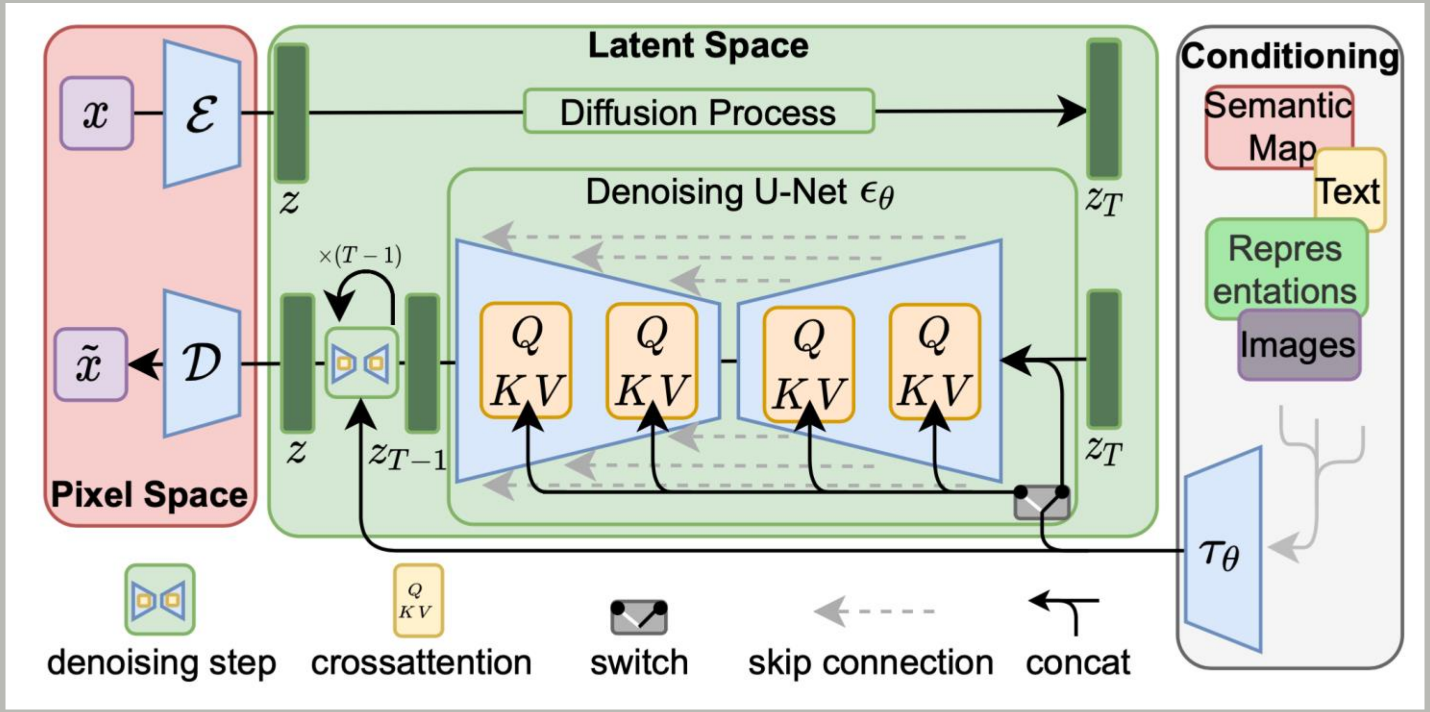


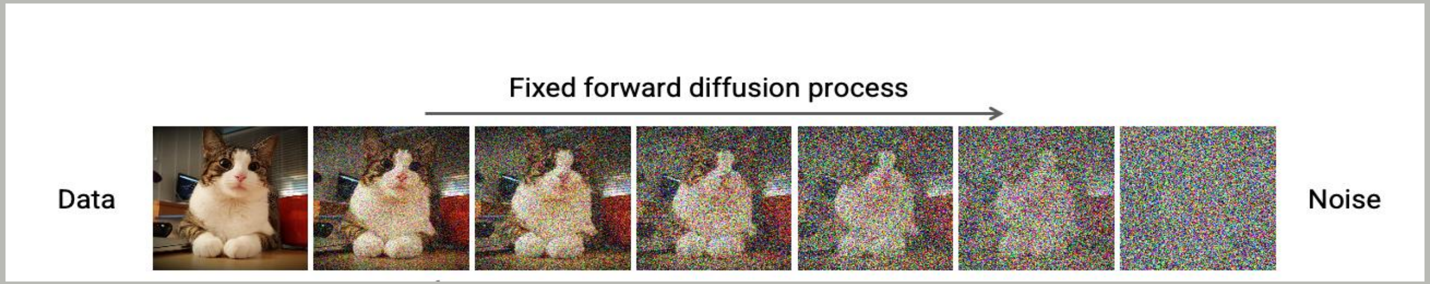
For 50 steps: Gaussian noise is drawn for every pixel and added to the pixel values

Noising process

Input

Output

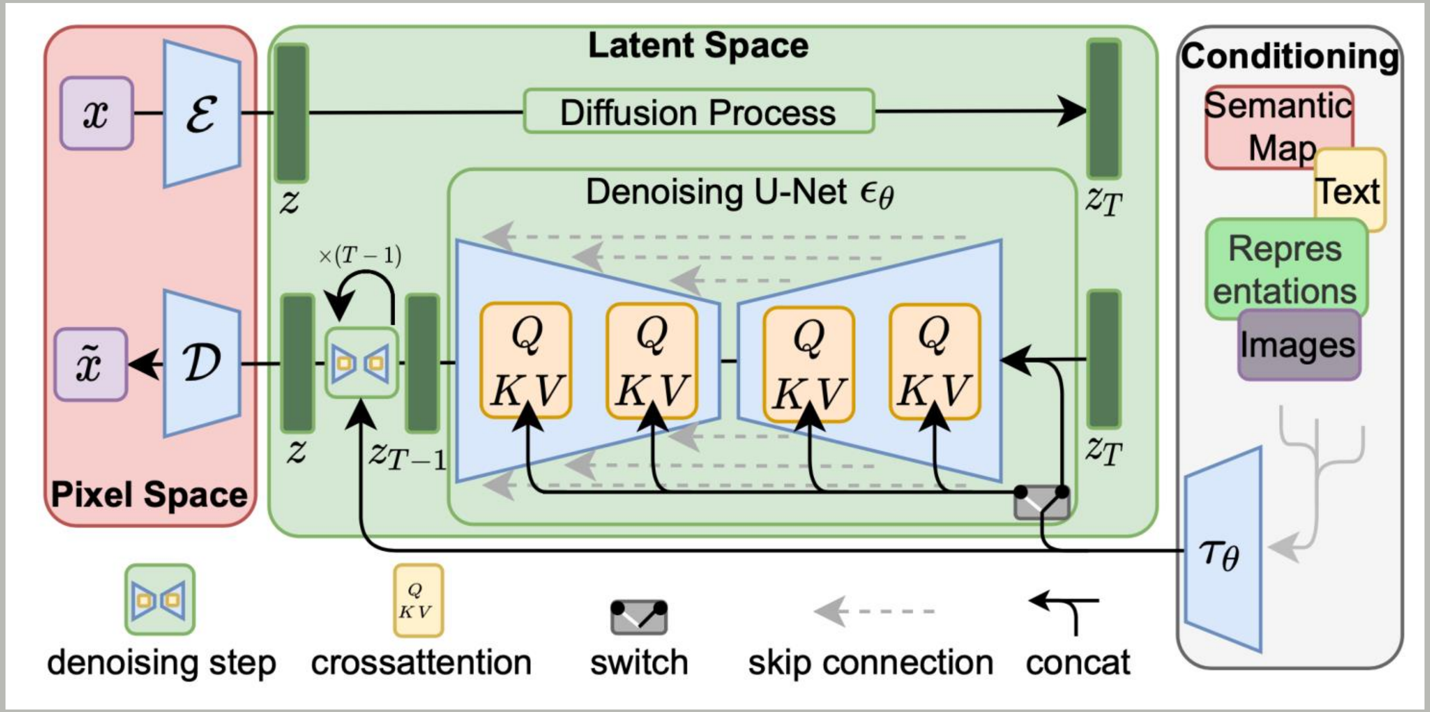




Noising process

Input

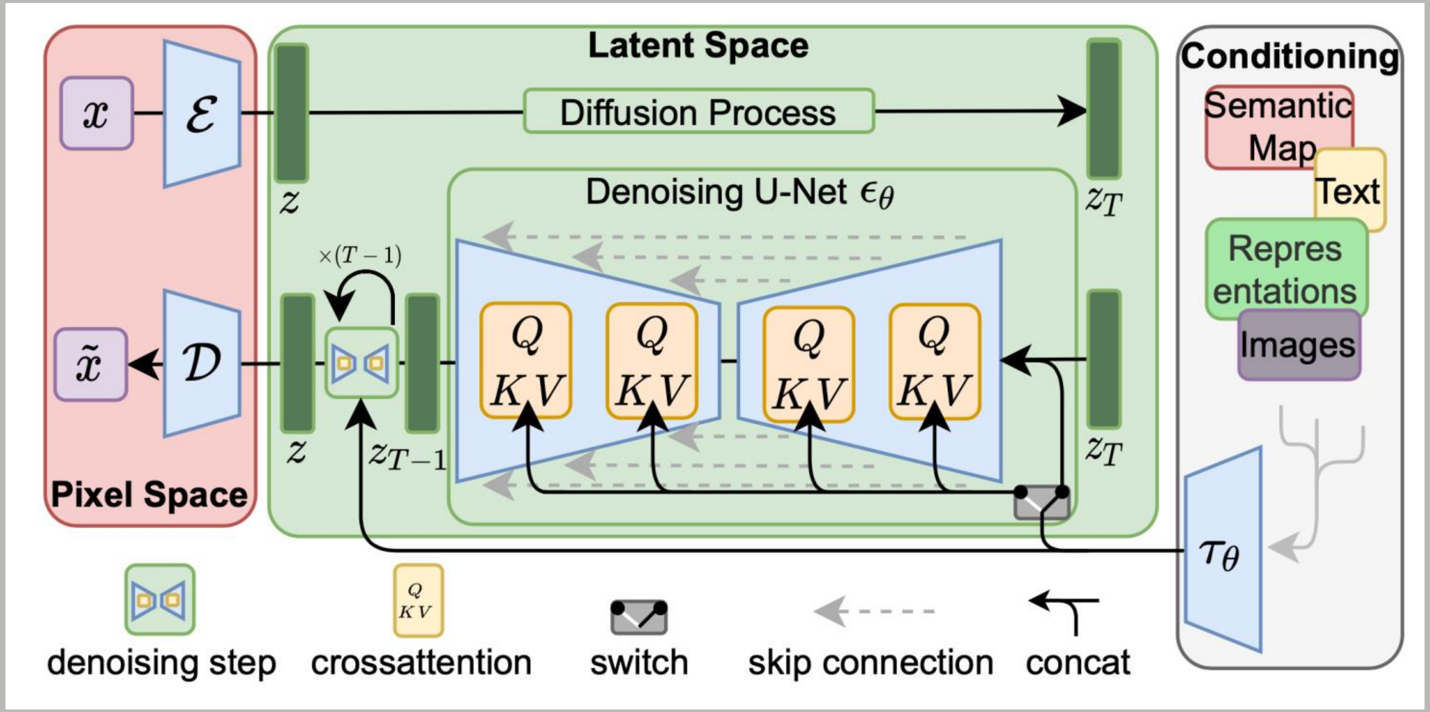
Output



For 50 steps: Gaussian noise is drawn for every pixel and added to the pixel values,
resulting in a fully noised picture

Input

Output

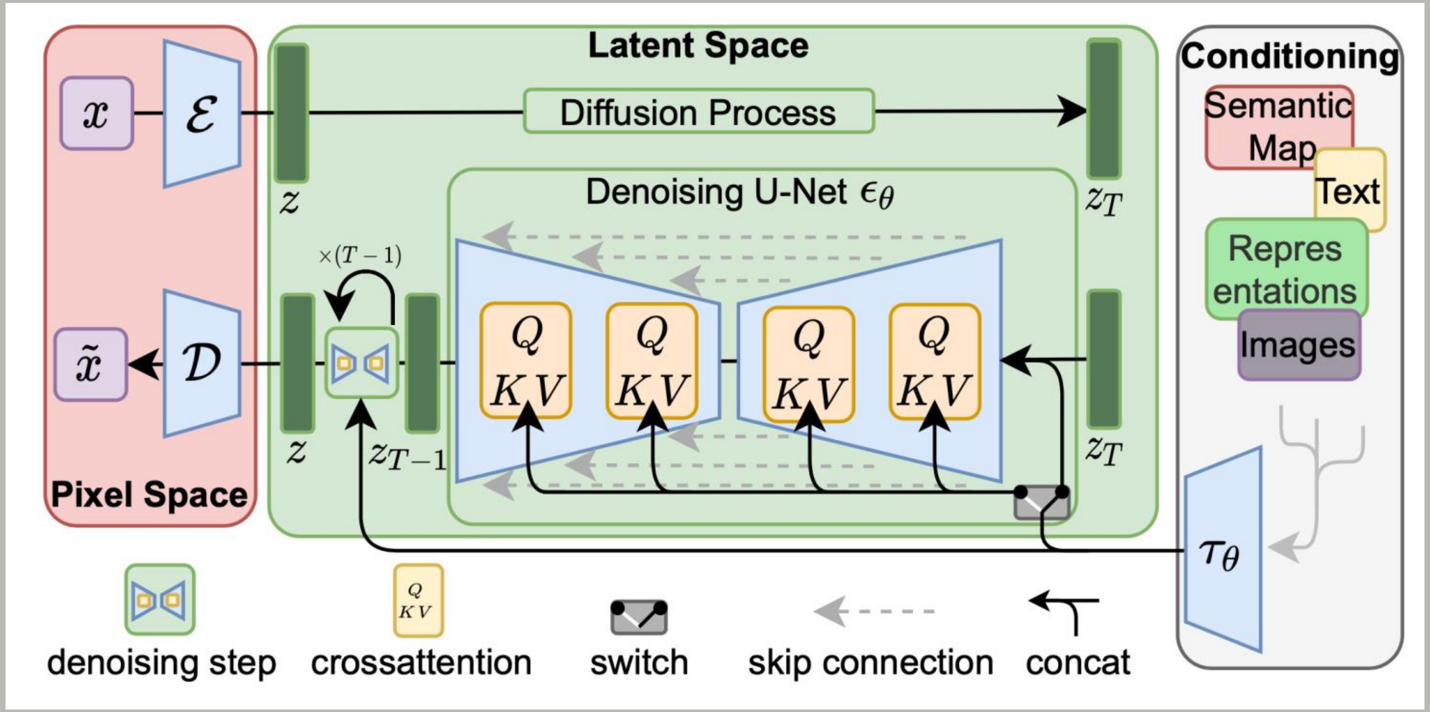


A picture of a cat

Now the noise and the text prompt serve as input for the generator

Input

Output



+

1.2	-0.1	4.3	3.2
0.4	2.5	-0.9	0.5
2.1	0.3	0.1	0.4

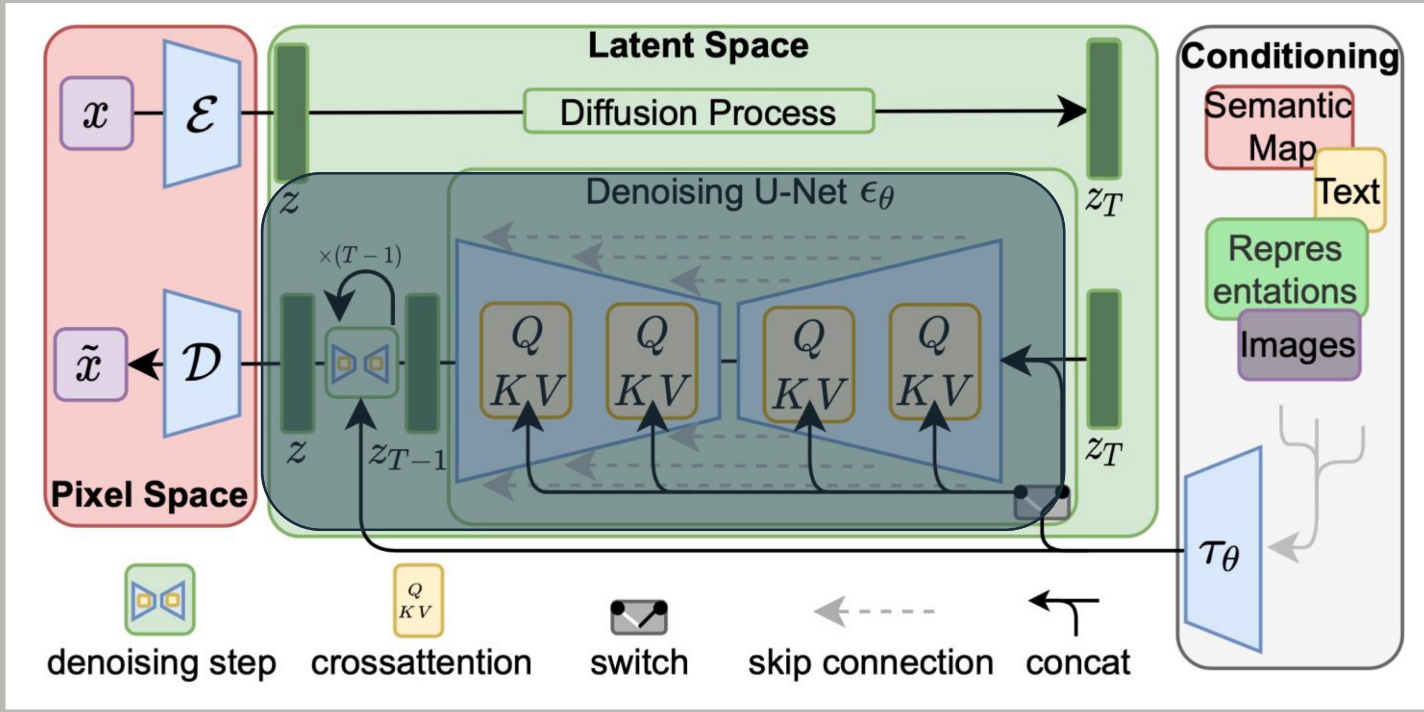
OpenAI CLIP

Now the noise and the text prompt serve as input for the generator, or rather an embedding of the prompt

Metyis

Input

Output



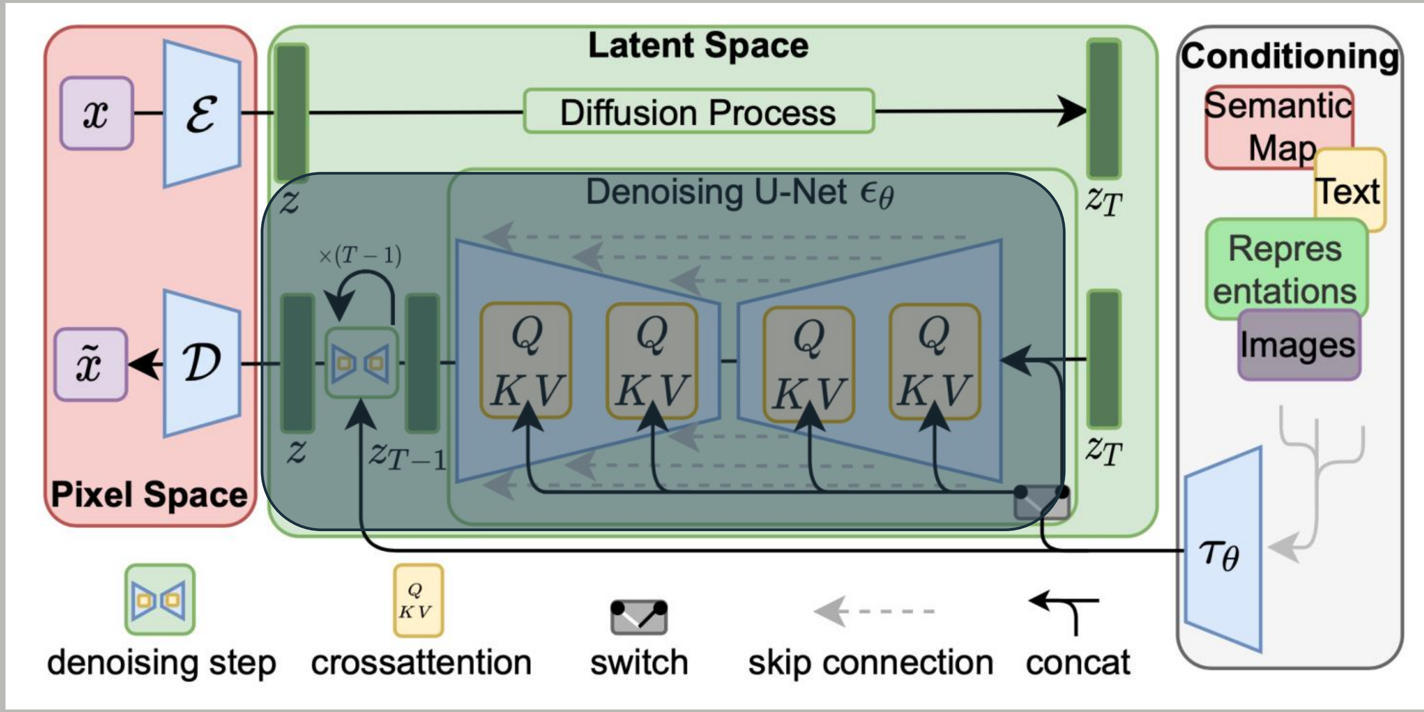
+

A picture of a cat

For 50 steps: the denoiser module tries to predict which noise was added to the picture. The embedded text guides the model in this process.

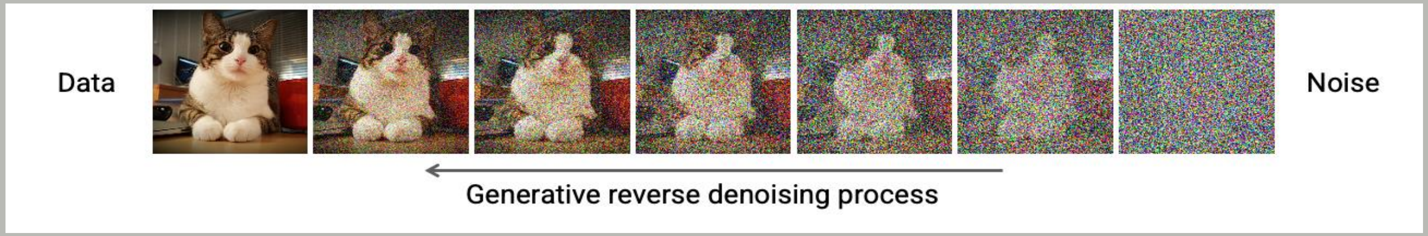
Input

Output



+

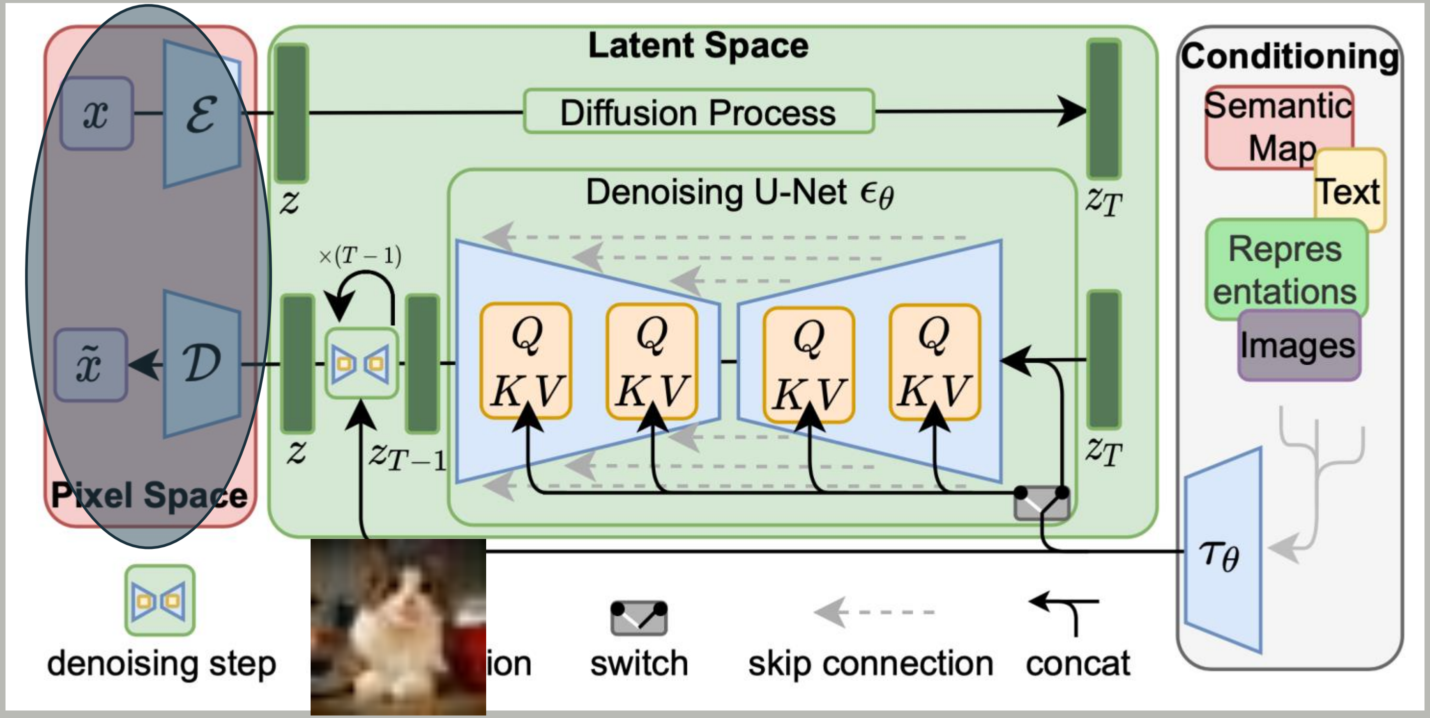
A picture of a cat



Encoder / Decoder (de)compression

Input

Output



+

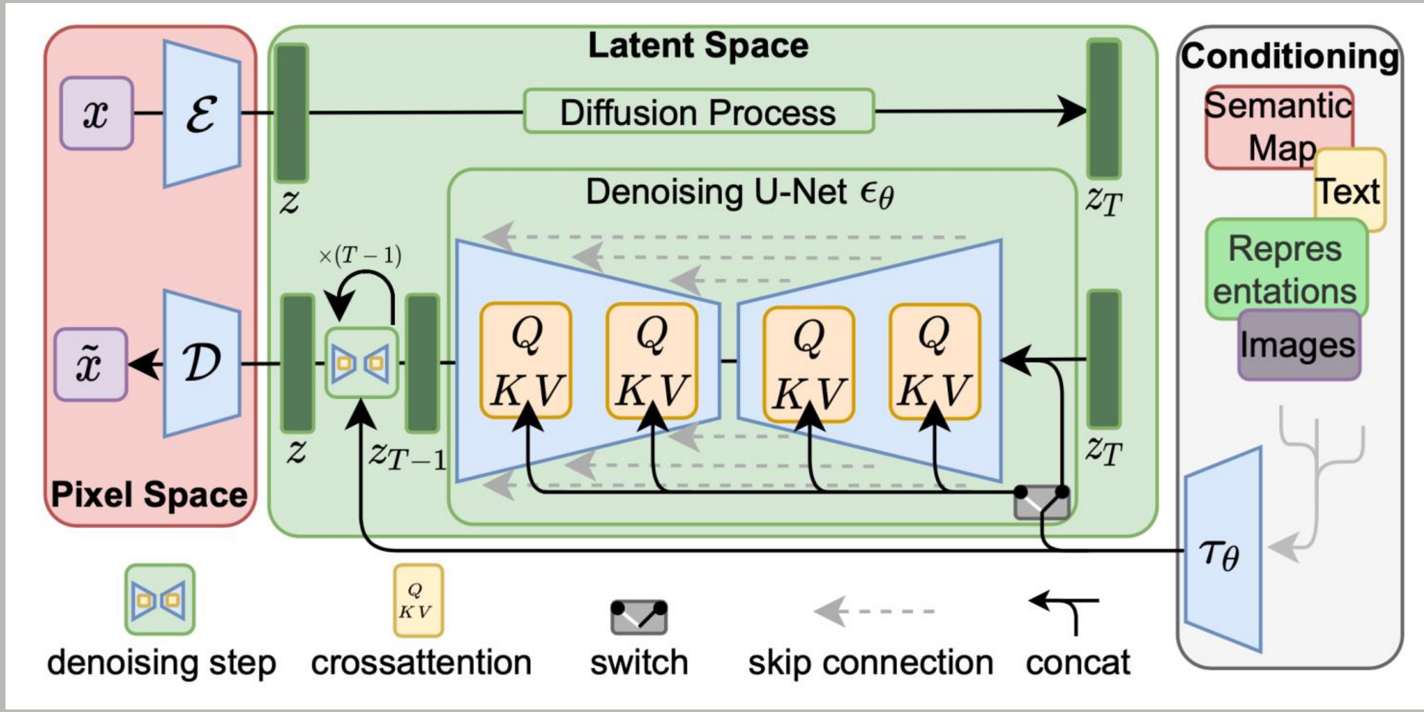
A picture of a cat

The decoder decompresses the image into its original size

Metyis

Input

Output



+

A picture of a cat

Calculate MSE loss between input and output

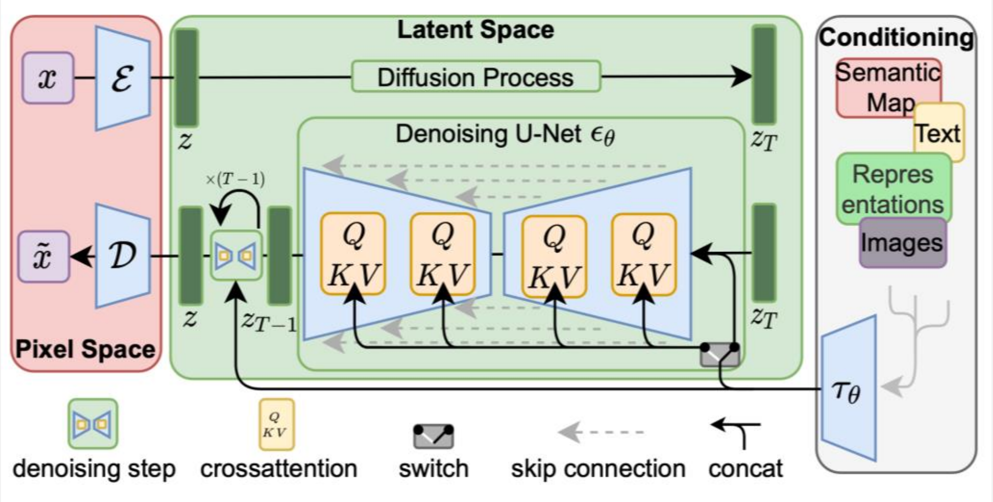
5 Approach

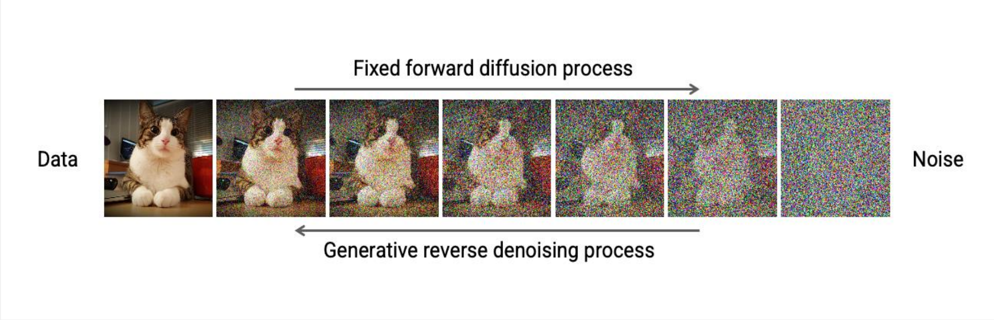
Goal:

- Add custom items
- Generate consistent output

1. Identify base model
2. Select appropriate KPI
3. Experiment with fine-tuning methods
4. Finalize pipeline

Base model: Stable Diffusion
-> open-source latent diffusion model





Metyis

Prompt failures ⟶ inference methods

Catastrophic neglect

parts of the prompt gets ignored

"A blue cat and a yellow bowl"

Incorrect attribute binding

characteristics getting linked to the wrong subject

"A man wearing a blue t shirt and red pants"

Personalization ⟶ fine-tuning

No brand characteristics captured

Pepe Jeans sweater

Malformed logos

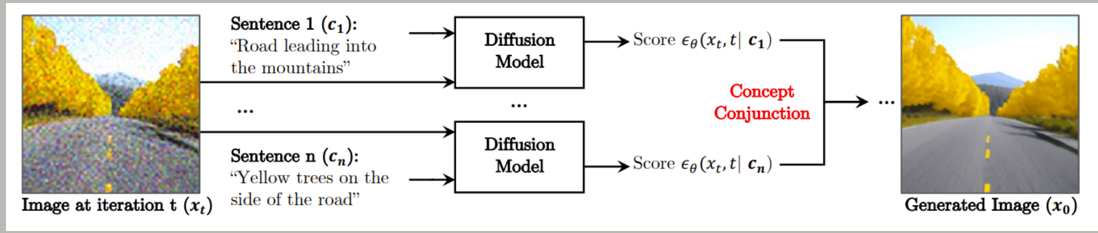
a man wearing a Hugo Boss polo

Inference methods

To improve prompt comprehension

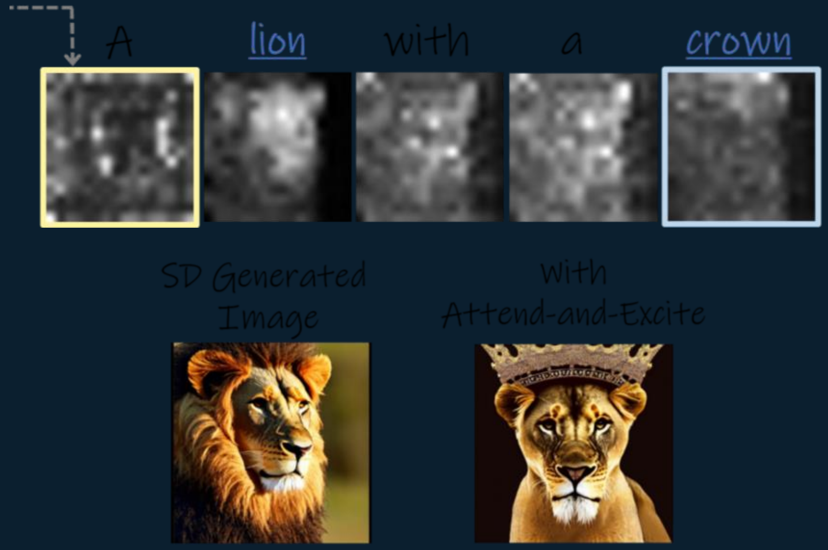
Composable Diffusion

- Divide the prompt into components using AND statements
- Let separate denoisers solve for the component
- Join their outputs



Attend and Excite

- Select keywords to "excite" in the prompt
- During the generation process, attention maps for the keywords are analyzed
- If attention for keyword is lower than the threshold, iteratively increase attention on this token



Metyis

Fine-tuning methods

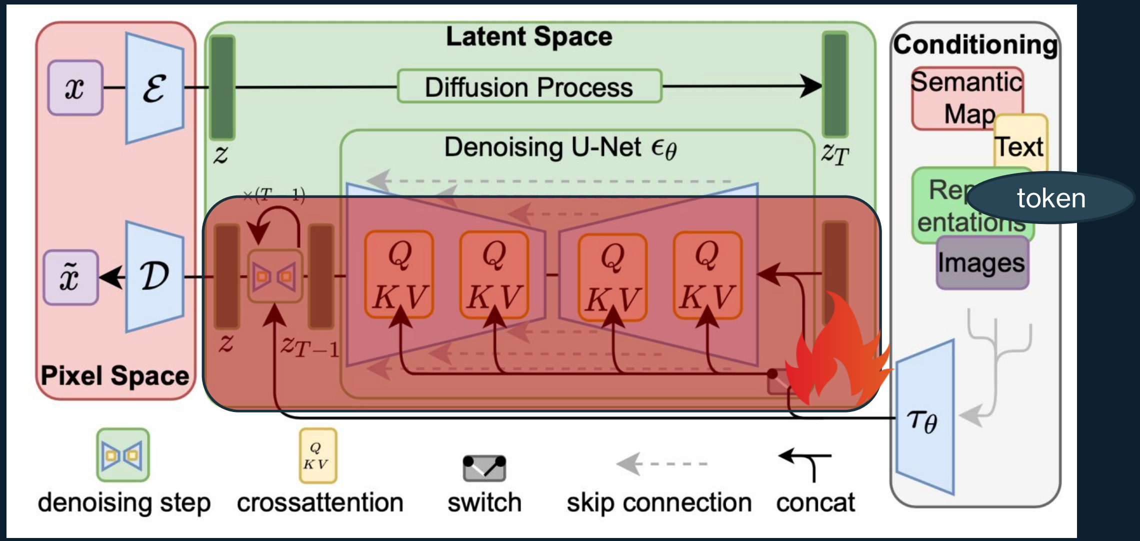
To personalize the model

Train the model to include new tokens.

Tokens can include specific items, the style of a brand, a person, a logo...

Dreambooth

Train optimal weights for specified concept



Metyis

Fine-tuning methods

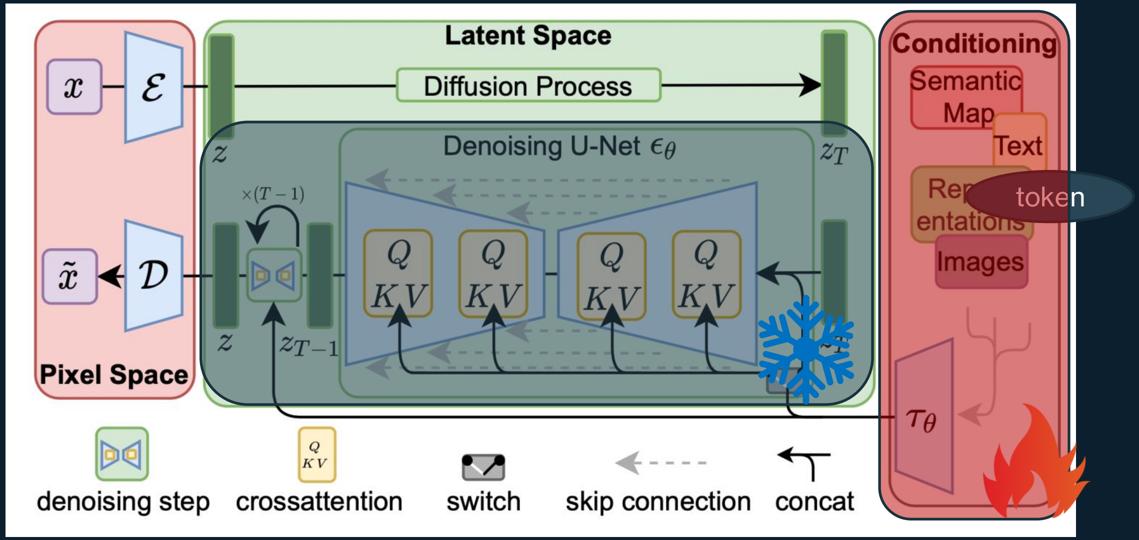
To personalize the model

Train the model to include new tokens.

Tokens can include specific items, the style of a brand, a person, a logo...

Textual Inversion

Train optimal *embedding* for specified concept



Metyis

Fine-tuning methods

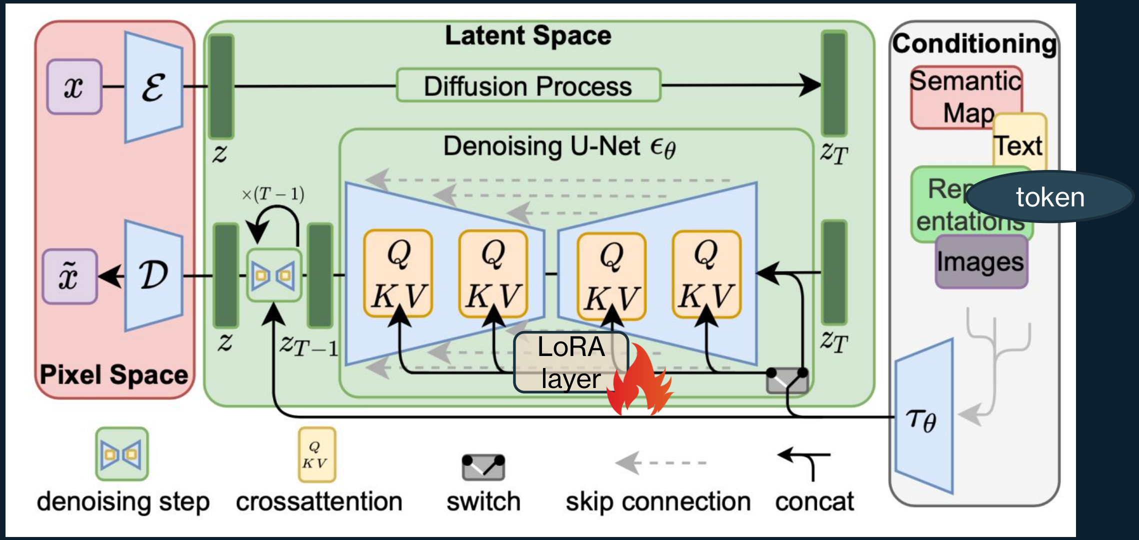
To personalize the model

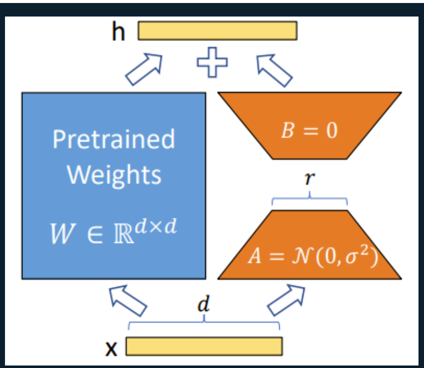
Train the model to include new tokens.

Tokens can include specific items, the style of a brand, a person, a logo...

Low Rank Adaptation

Train low rank intermediate layers





Metyis

Dataset

A clothing line of HUGO was chosen as dataset for POC

All training images contain the ‘red_hugo_logo’

black shorts with red_hugo_logo

black sweater with red_hugo_logo

black T-shirt with red_hugo_logo

black hat with red_hugo_logo

6 Evaluation

Object Detection

Does the logo look like the logo?

YOLOv8
150 training images

+ Image augmentation methods
+ Regularizing images

Optical Character Recognition

Is the logo spelled correctly?

OCR models output text displayed on image

0: HUCO

1: HUGo

2: Hug Hudo

7 Results

Dreambooth best

LoRA okay

Textual Inversion not great, especially OCR

method	parameters	average(ocr, yolo)	mean_confidence_score	mean_ocr_score
db	lr0_00002	0.724898001	0.949796001	0.5
lora	UNet5e-06TE0_0001dim16	0.594578506	0.709990345	0.479166667
ti	lr_0_001	0.545065025	0.756796718	0.333333333

Dreambooth best

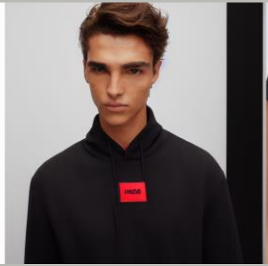
LoRA okay

Textual Inversion not great, especially OCR

method	parameters	average(ocr, yolo)	mean_confidence_score	mean_ocr_score
db	lr0_00002	0.724898001	0.949796001	0.5
lora	UNet5e-06TE0_0001dim16	0.594578506	0.709990345	0.479166667
ti	lr_0_001	0.545065025	0.756796718	0.333333333

lr0_00002 epoch 21

A red_hugo_logo


A male model wearing a blue red_hugo_logo sweater


A female model wearing a green red_hugo_logo t-shirt

The latest red_hugo_logo products

A billboard with the red_hugo_logo

The new red_hugo_logo fragrance perfume

Metyis

Dreambooth best

LoRA okay

Textual Inversion not great, especially OCR

method	parameters	average(ocr, yolo)	mean_confidence_score	mean_ocr_score
db	lr0_00002	0.724898001	0.949796001	0.5
lora	UNet5e-06TE0_0001dim16	0.594578506	0.709990345	0.479166667
ti	lr_0_001	0.545065025	0.756796718	0.333333333

lr0_00001 epoch 12

A red_hugo_logo

A male model wearing a blue red_hugo_logo sweater

A female model wearing a green red_hugo_logo t-shirt

The latest red_hugo_logo products

A billboard with the red_hugo_logo

The new red_hugo_logo fragrance perfume

Dreambooth

+ Captures details
+ High quality
+ Consistent

- Overfits easily
- Huge output size

The new red_hugo_logo perfume

Image from training data

Textual Inversion

+ Captures concept
+ Converges well over high LR
+ Small output

- Does not capture details well (e.g., spelling)
- Inconsistent

LoRA

+ Captures concept
+ Captures details
+ Small output

- Hard to get right
- Inconsistent

8 Conclusion

Take home:

- For better prompt guidance: Attend-and-Excite
- If compute and memory does not matter, and desired output similar to dataset: Dreambooth
- If details not important, more about aesthetics: Textual Inversion
- Details important, but need scalable solution: LoRA
    - Training parameters matter
    - Training does not require a lot of data
    - Evaluation and testing can be tricky


HUGO


HUGO


HUGO


HUGO


HUGO


HUGO


HUGO


HUGO


HUGO

Special thanks to:

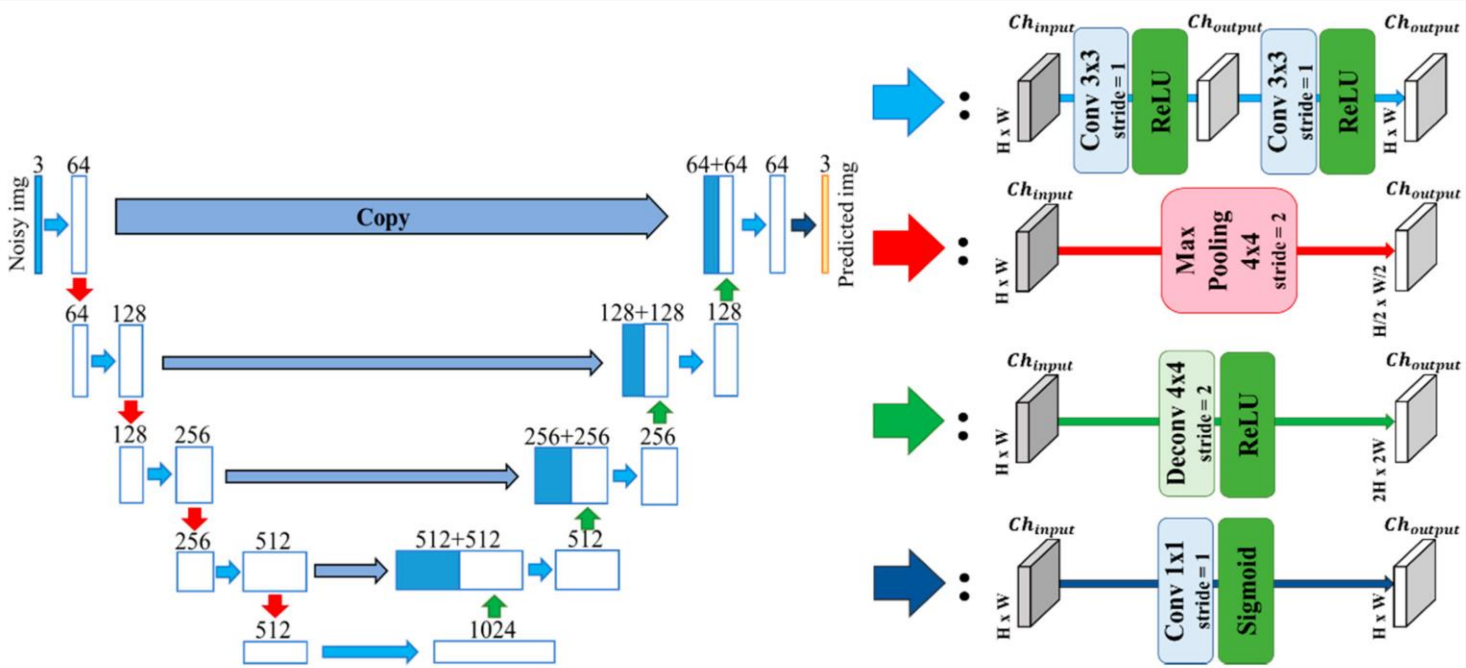
- Praneetha Yekkaluru
- Tomás Costa
- Anil Yaman (Vrije Universiteit Amsterdam)
- Akshay Singh

6 Questions

A

Appendix

Denoising U Net



U Net Architecture

The U Net architecture was originally used for (medical) image segmentation.

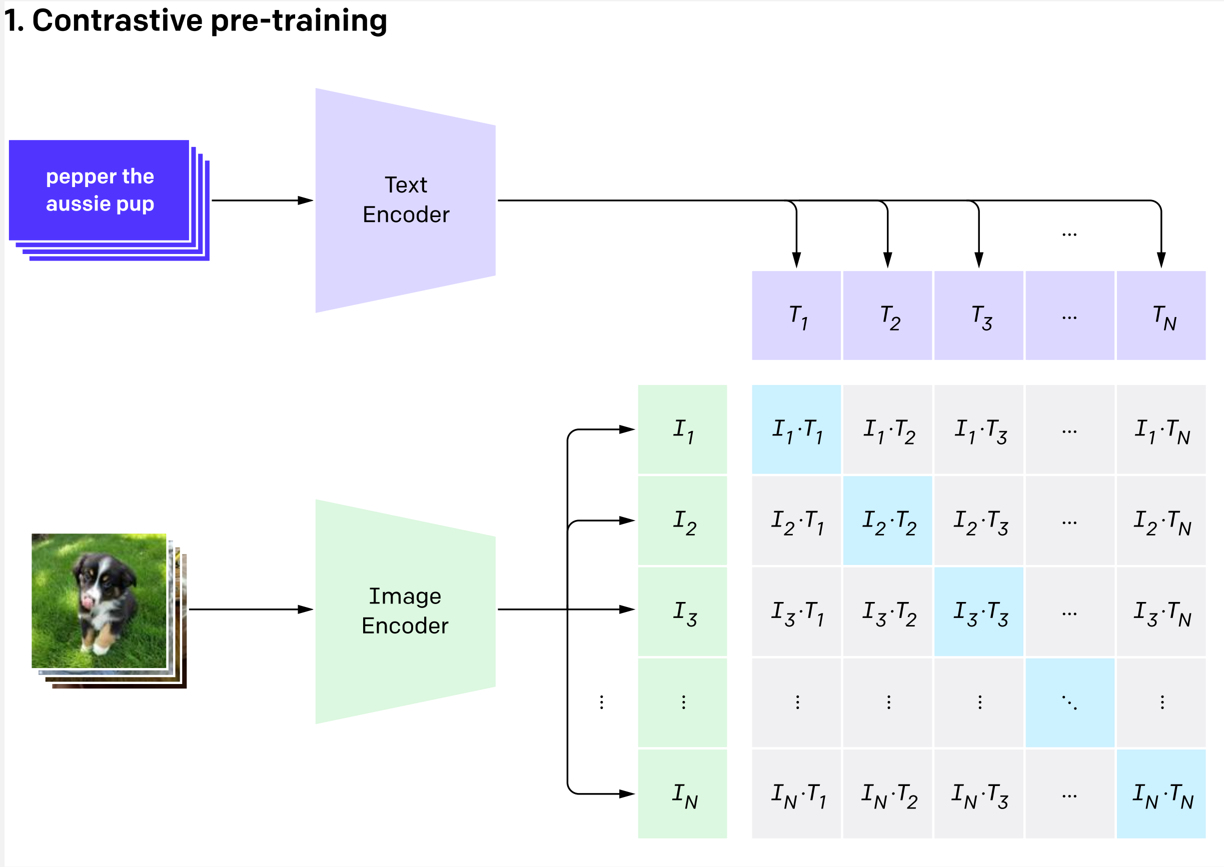
In Diffusion Models, it functions as the noise predictor.

It segments the image, through dimensionality reduction and guided by the text embedding.

Per segments it tries to remove noise in a stepwise fashion.

Metyis

CLIP text embedding



CLIP is OpenAI's zero-shot image classifier.

It's a multi-modal network that embeds any image or text input, allowing it to classify for unknown labels.

CLIP similarity score can be used in the same fashion to evaluate generated images.

Prompt failures

Catastrophic neglect

parts of the prompt do not get generated

“A blue cat and a yellow bowl”

[Composable Diffusion]

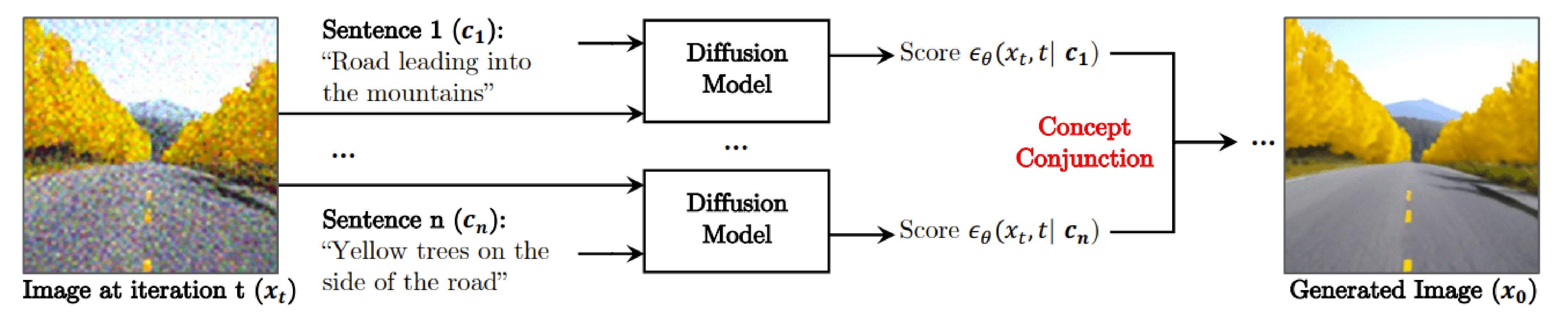
[Attend and Excite]

Incorrect attribute binding

characteristics getting linked to the wrong subject

“A man wearing a blue t shirt and red pants”

Composable Diffusion



Diffusion models capable of generating simple prompts, can we stack diffusion models using AND or NOT statements?

By combining the score-functions of multiple diffusion models, we can guide the diffusion process with multiple conjunctions

Prompt failures

Catastrophic neglect

parts of the prompt do not get generated

"A blue cat and a yellow bowl"

[Composable Diffusion]

[Attend and Excite]

Incorrect attribute binding

characteristics getting linked to the wrong subject

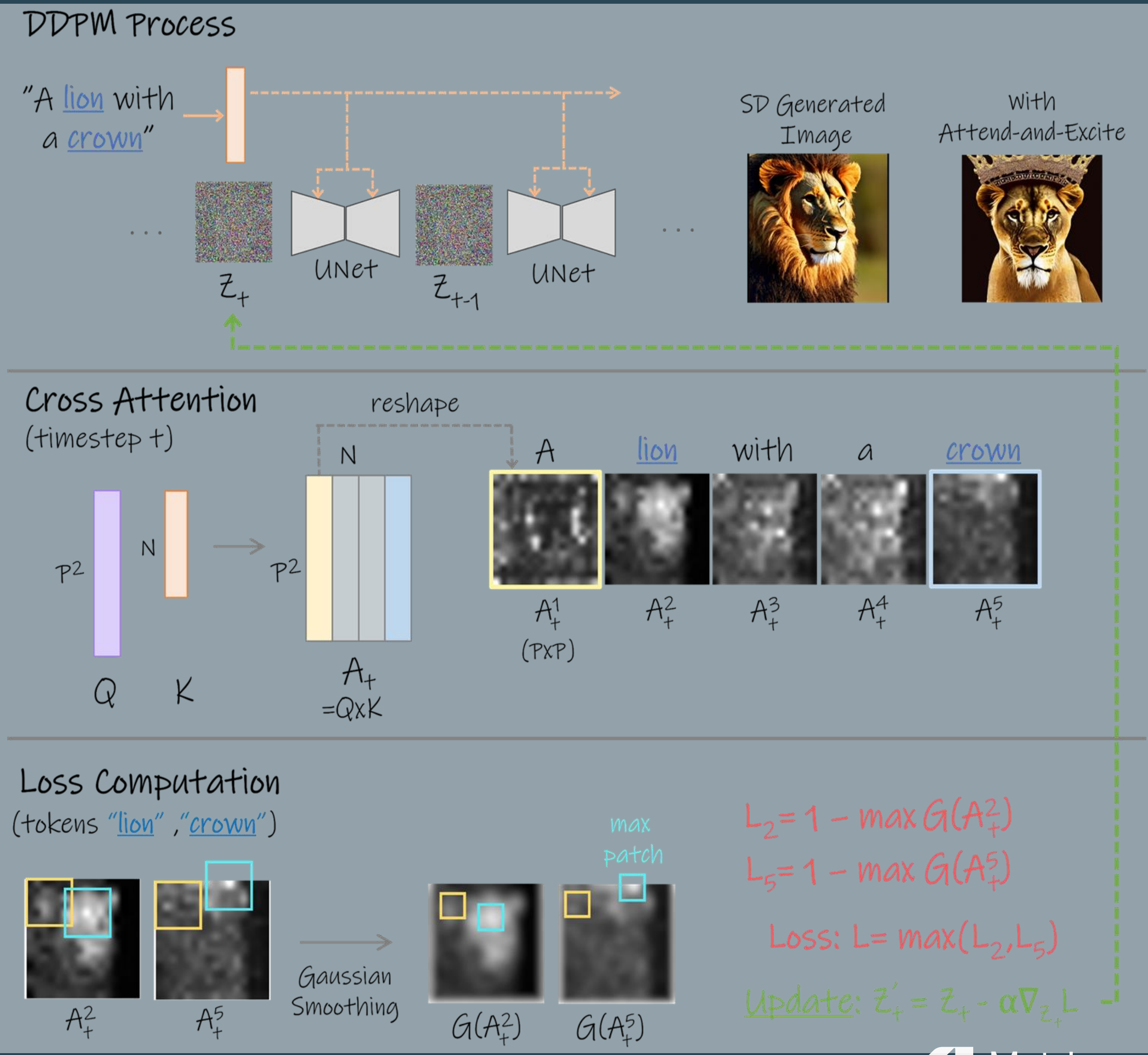
"A man wearing a blue t shirt and red pants"

Attend and Excite

Embedding method overrules attention blocks

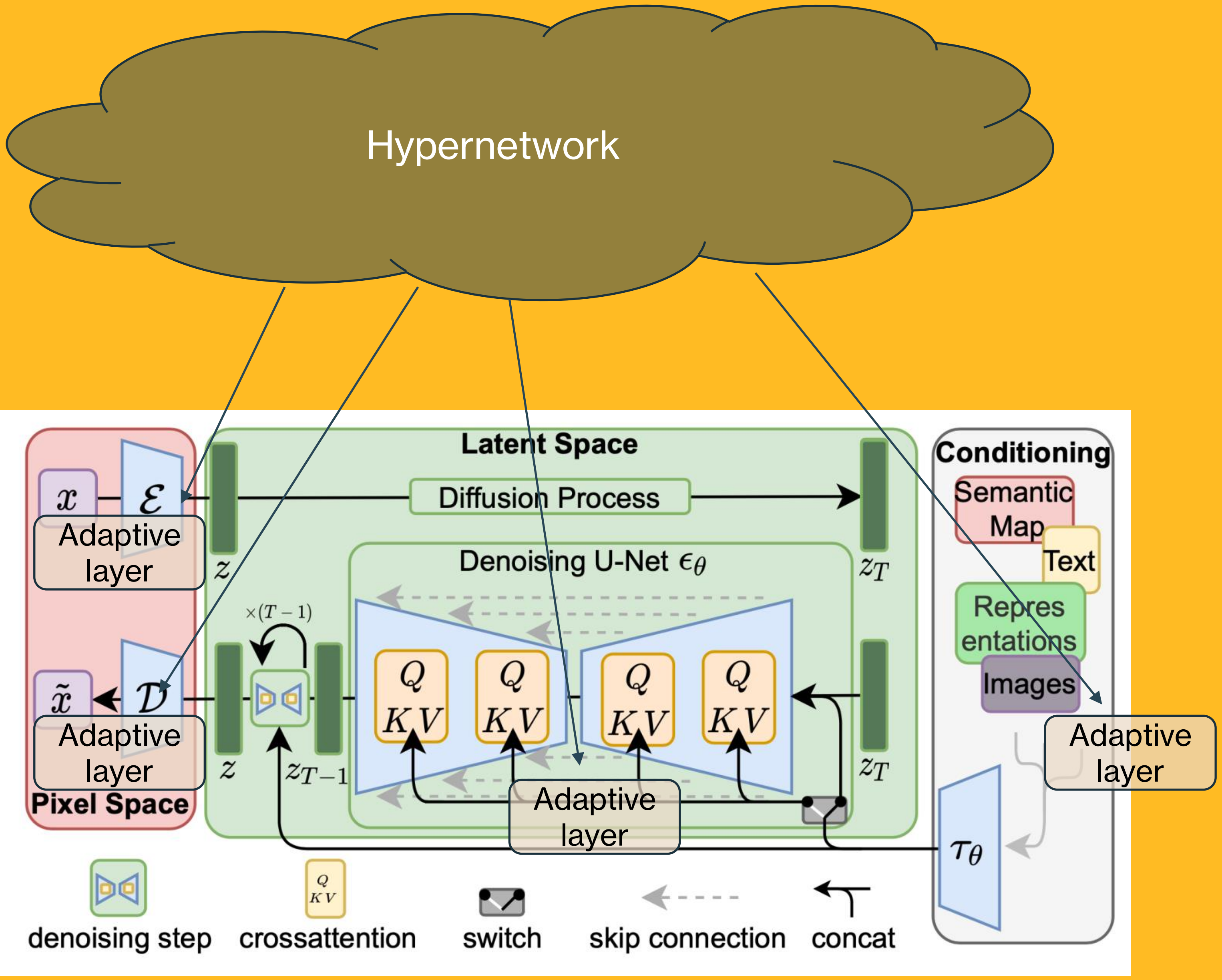
Can we force words to be included?

- Reweigh attention over excited words
- Increase attention for most neglected subject token



Hypernetwork

Maybe we should not do this one? People report very bad results, and its functionality has been replaced by LoRA



Model Architecture

Metyis

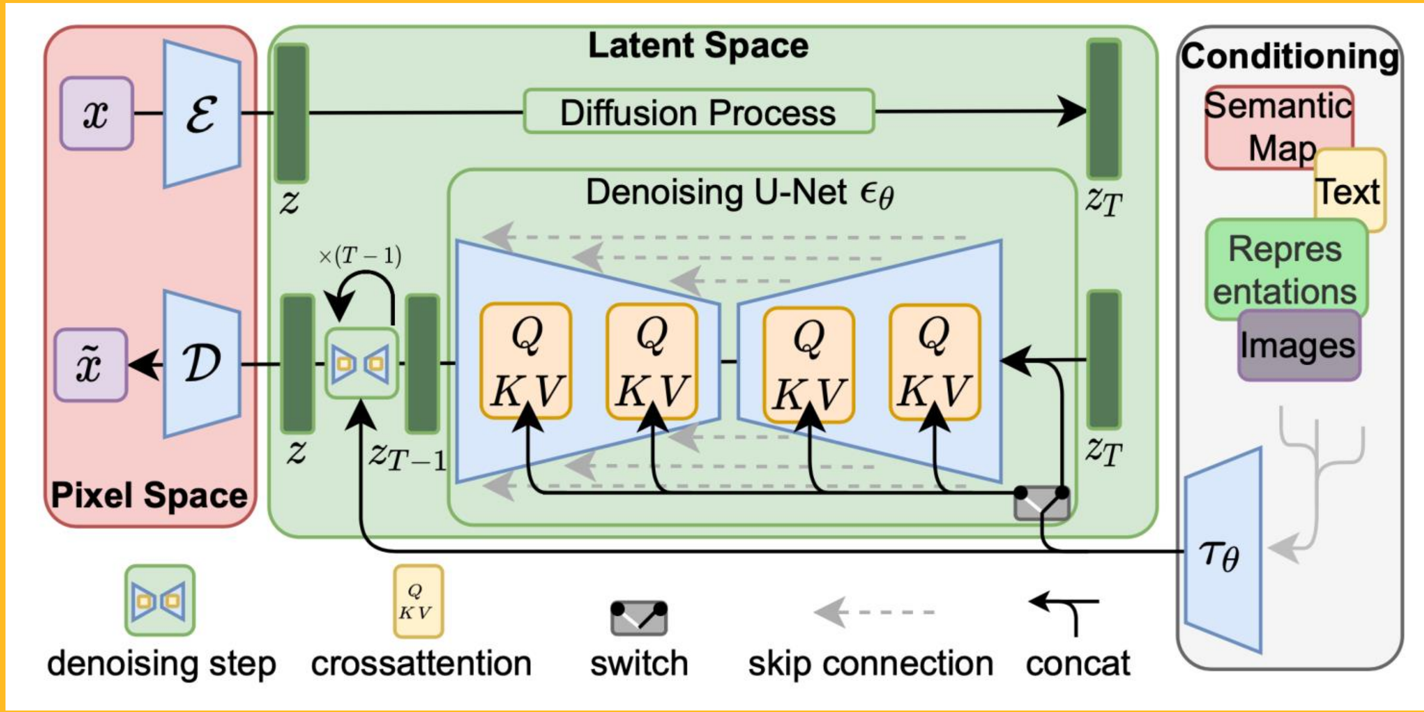
Use-cases

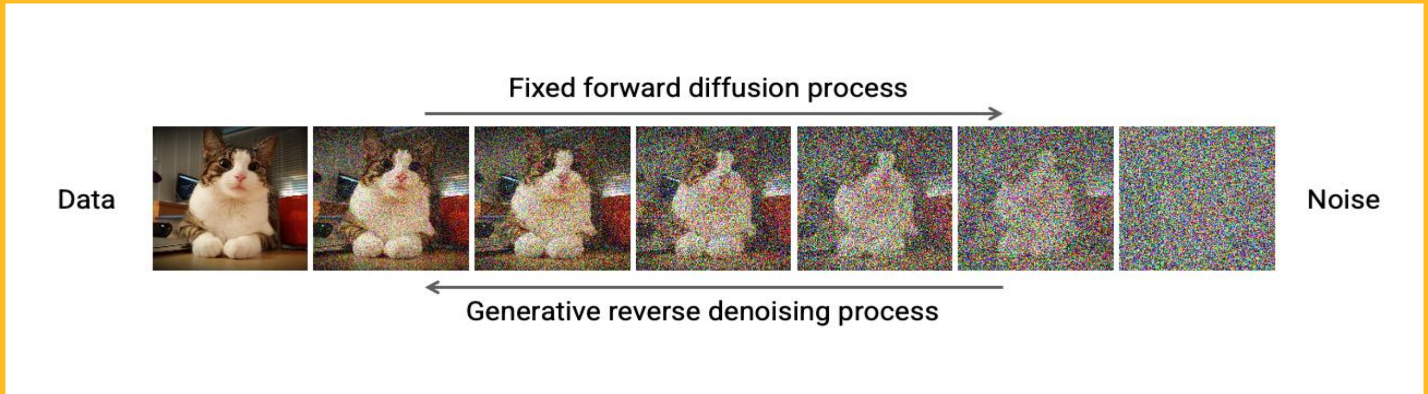
Generate ads for marketing campaigns

Generate catalog pictures

Input

Output

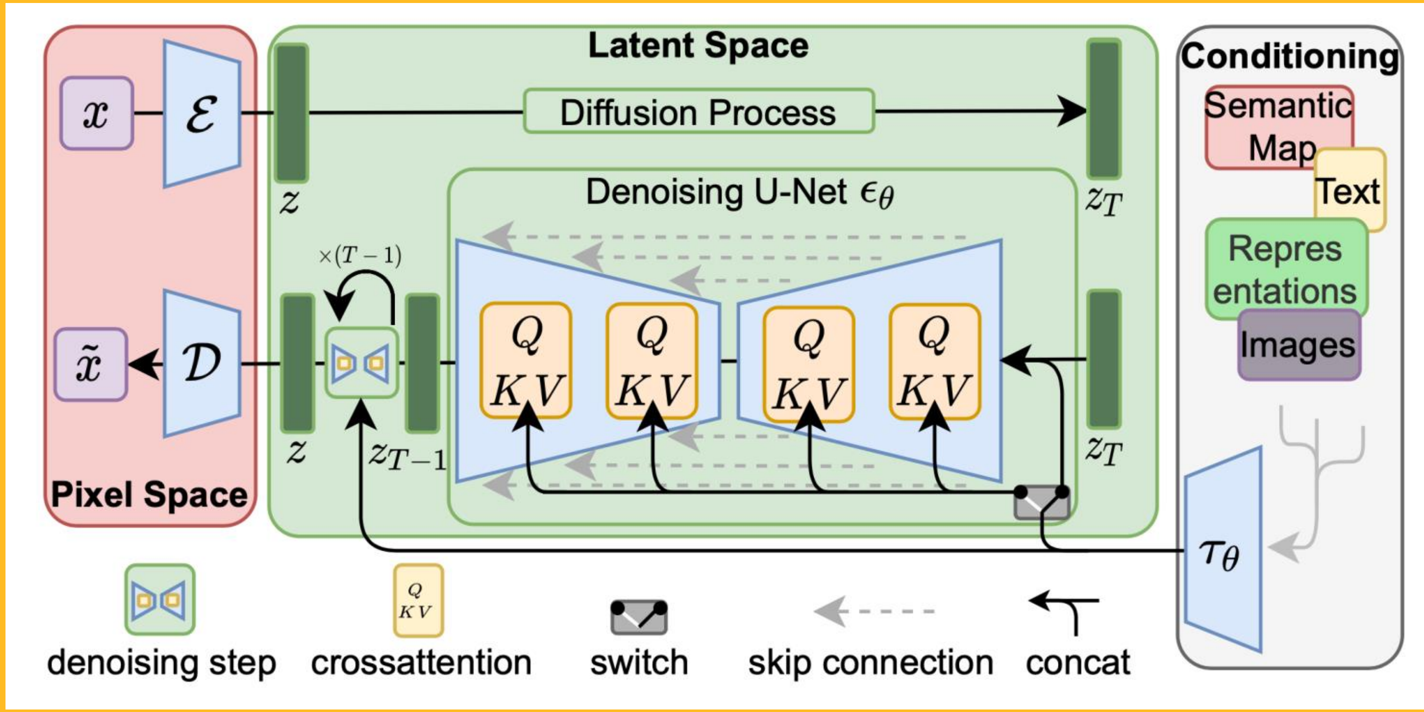




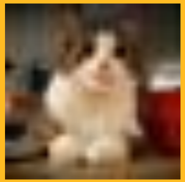
Metyis

Input

Output

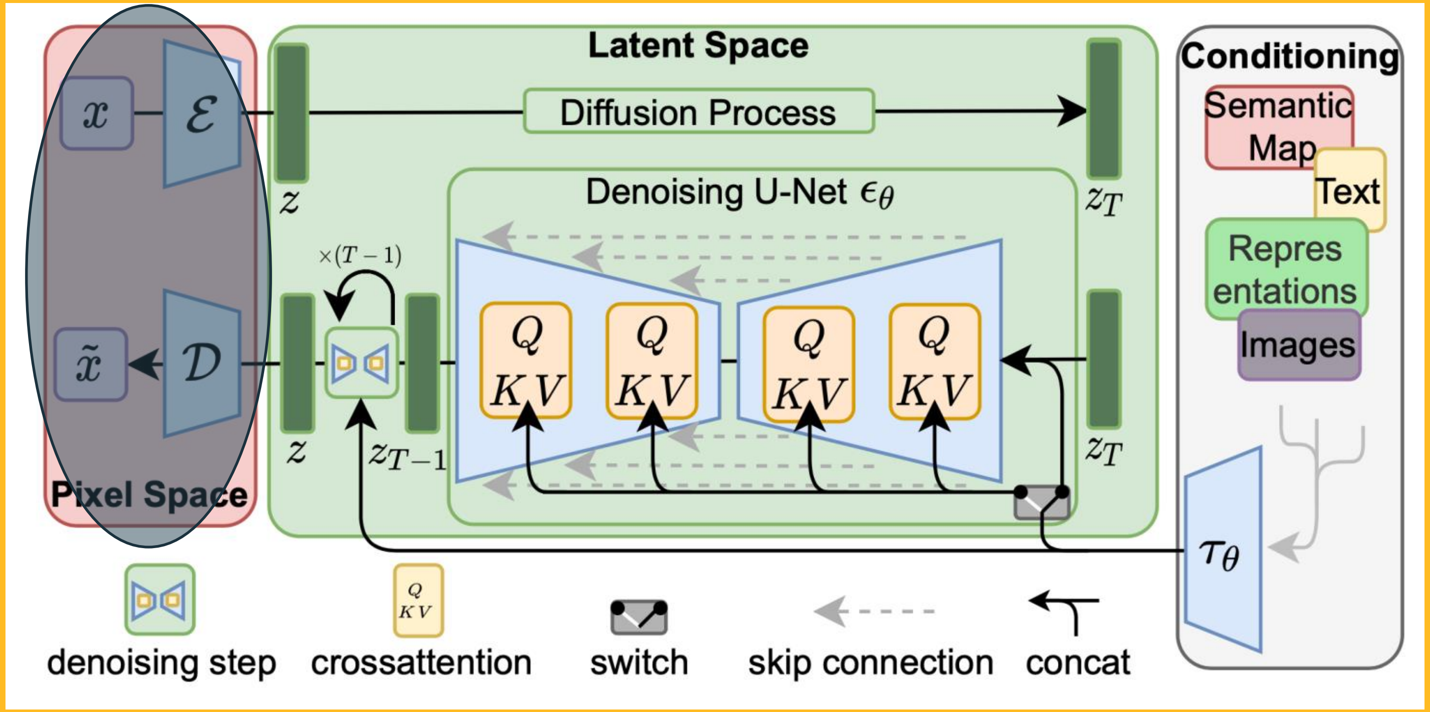


Encoder / Decoder



Input

Output

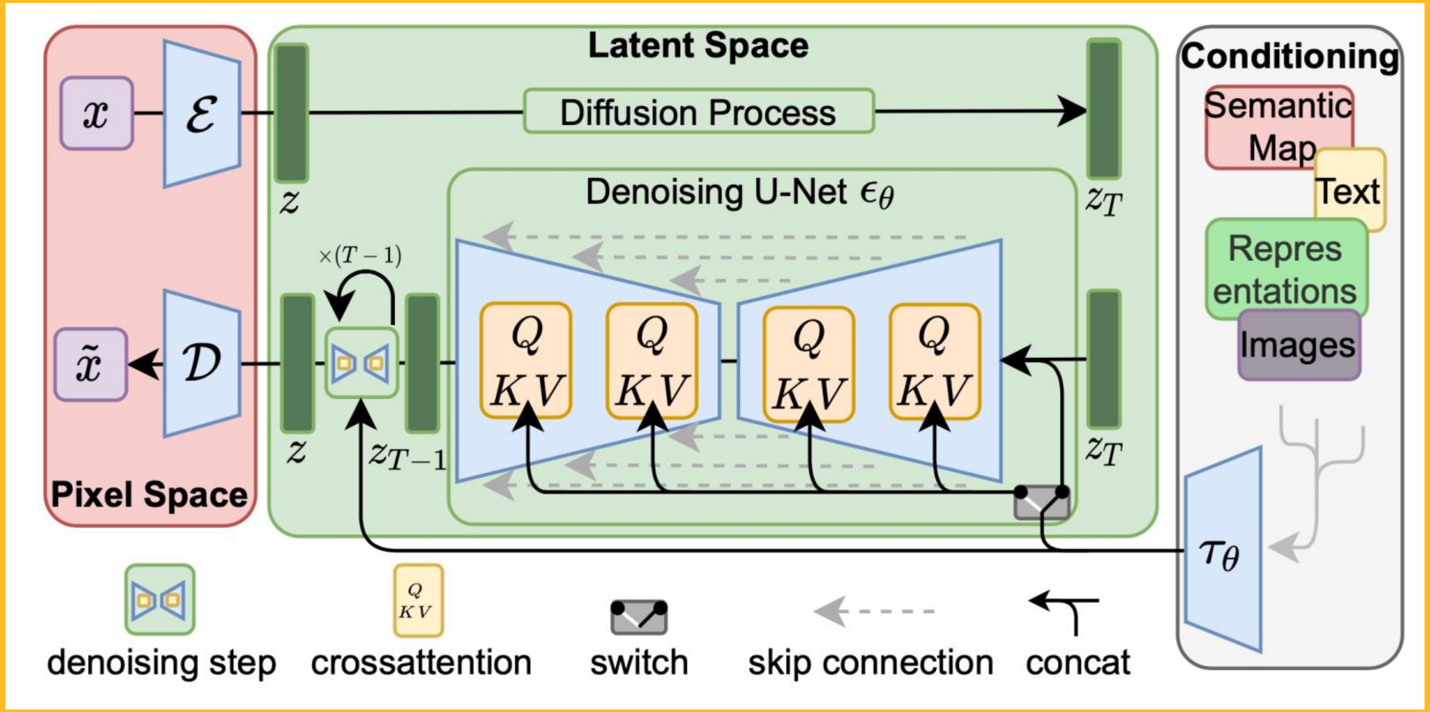


The encoder compresses the image into a lower dimensional latent space to allow faster computing and better image processing

Noising process

Input

Output

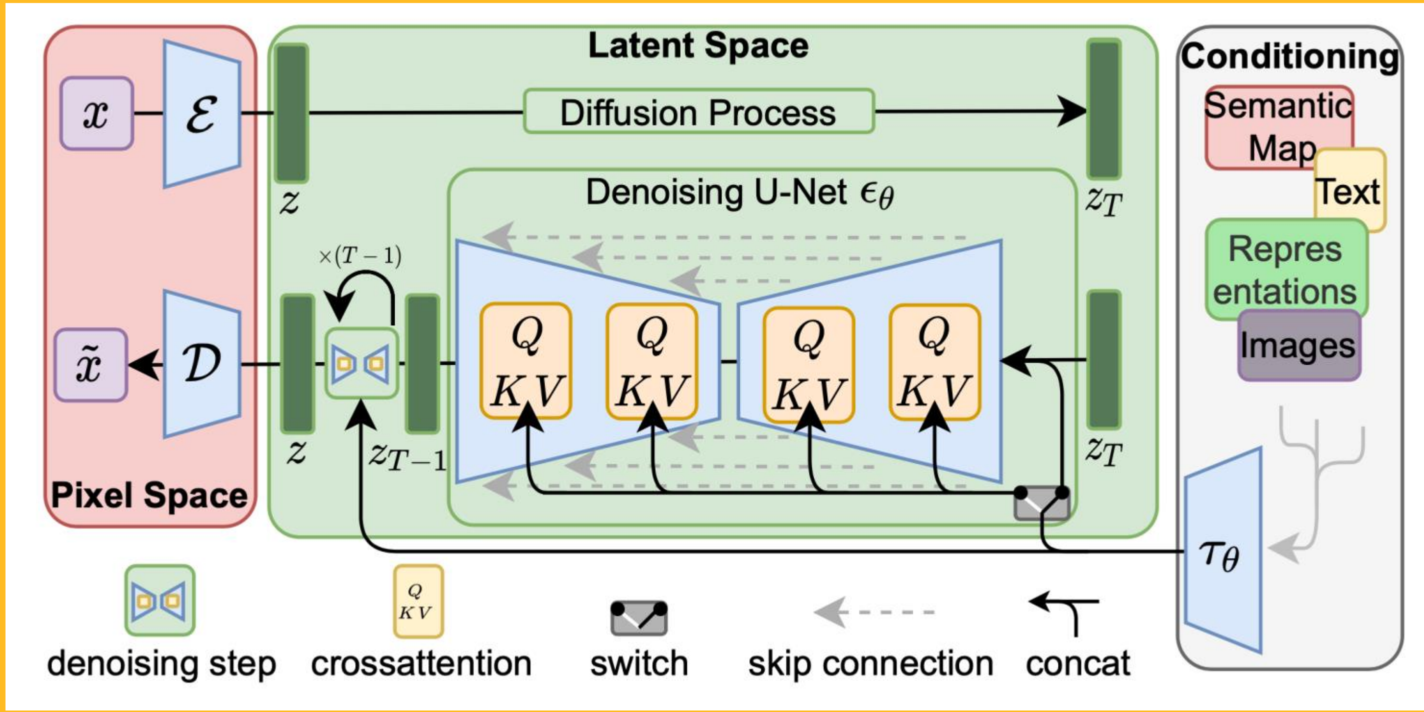


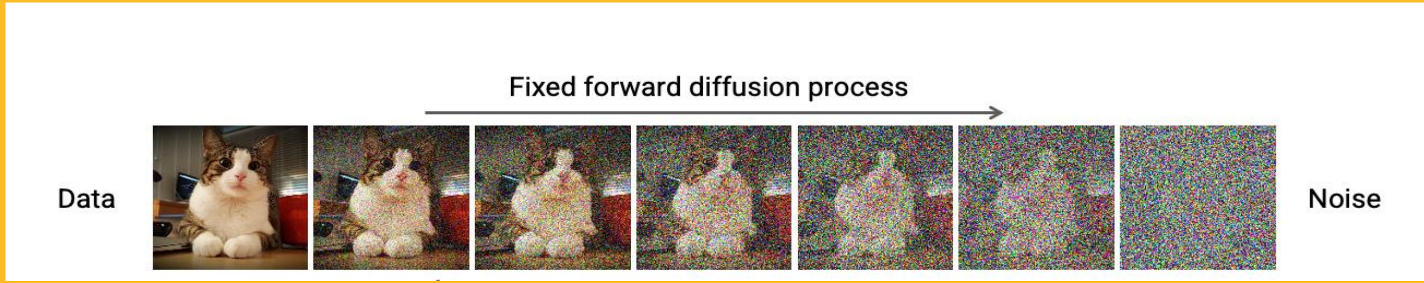
For 50 steps: Gaussian noise is drawn for every pixel and added to the pixel values

Noising process

Input

Output

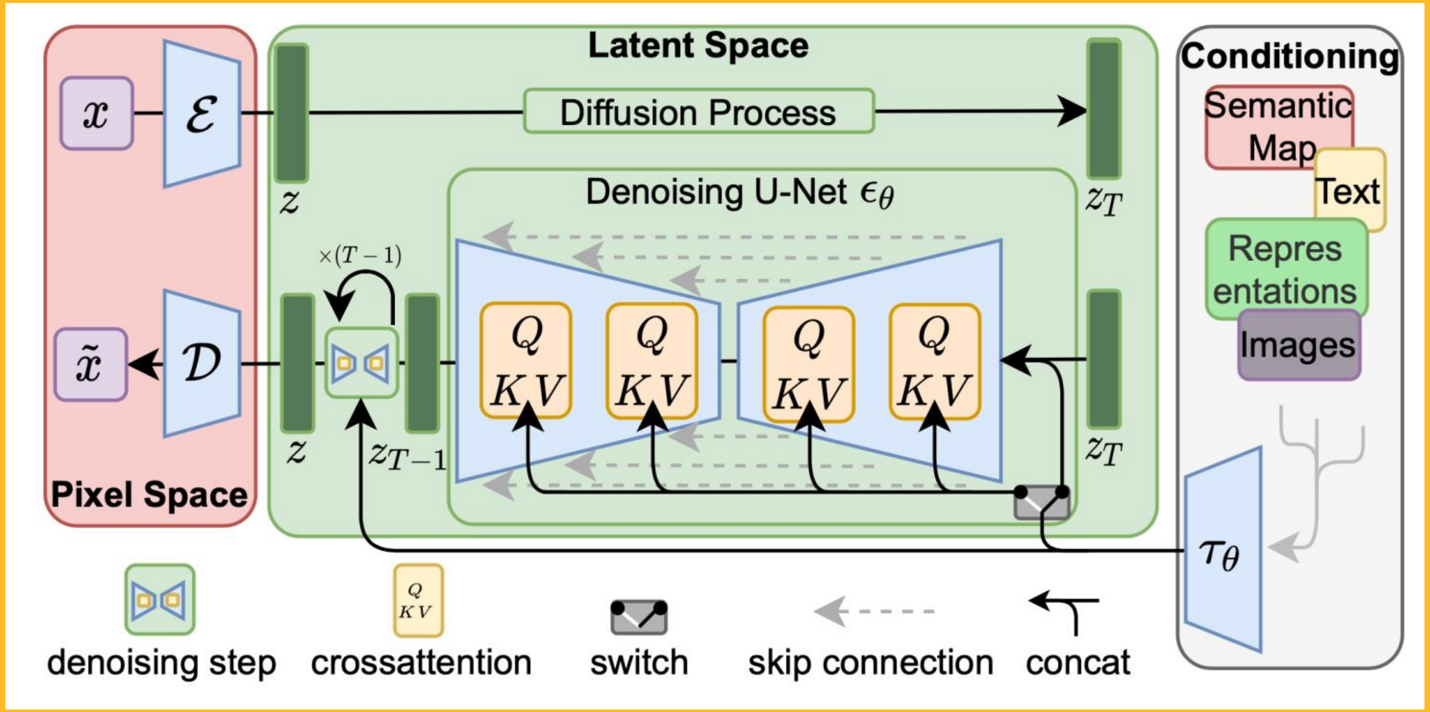




Noising process

Input

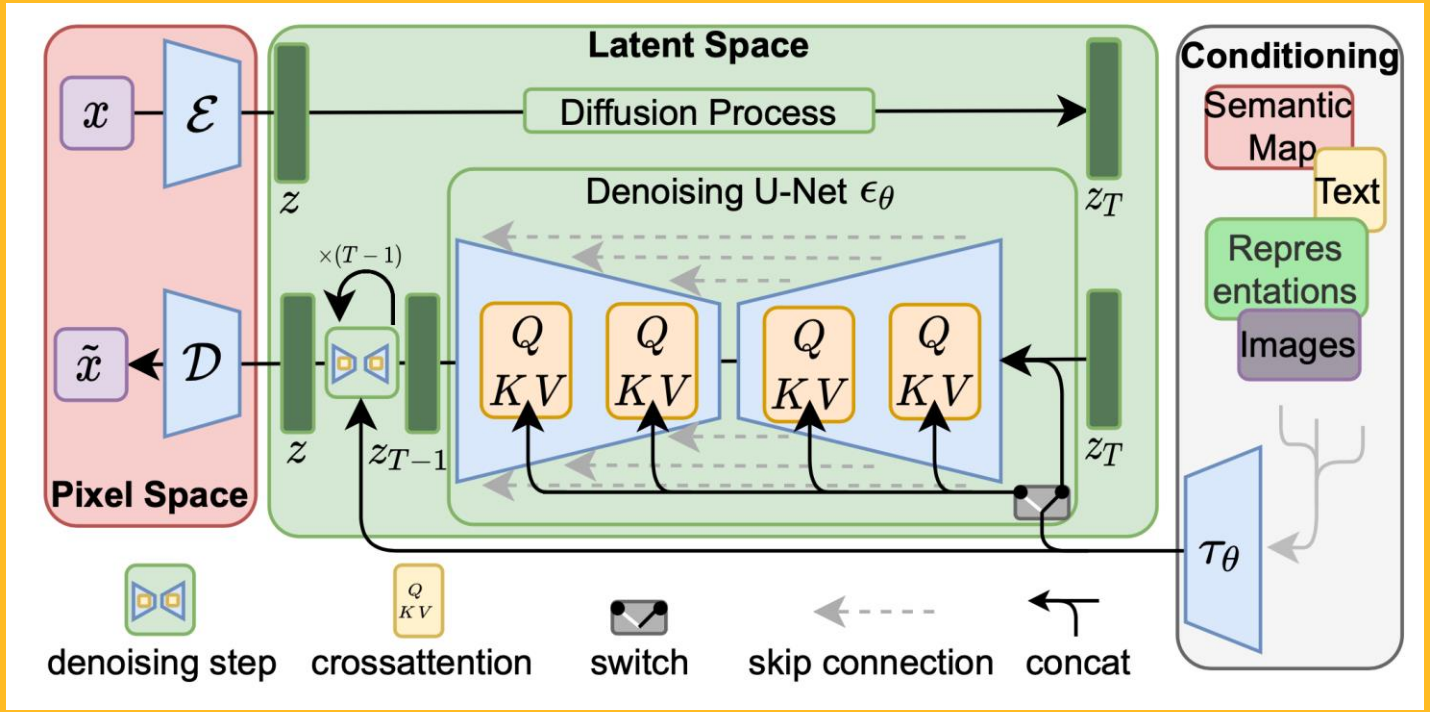
Output



For 50 steps: Gaussian noise is drawn for every pixel and added to the pixel values,
resulting in a fully noised picture

Input

Output



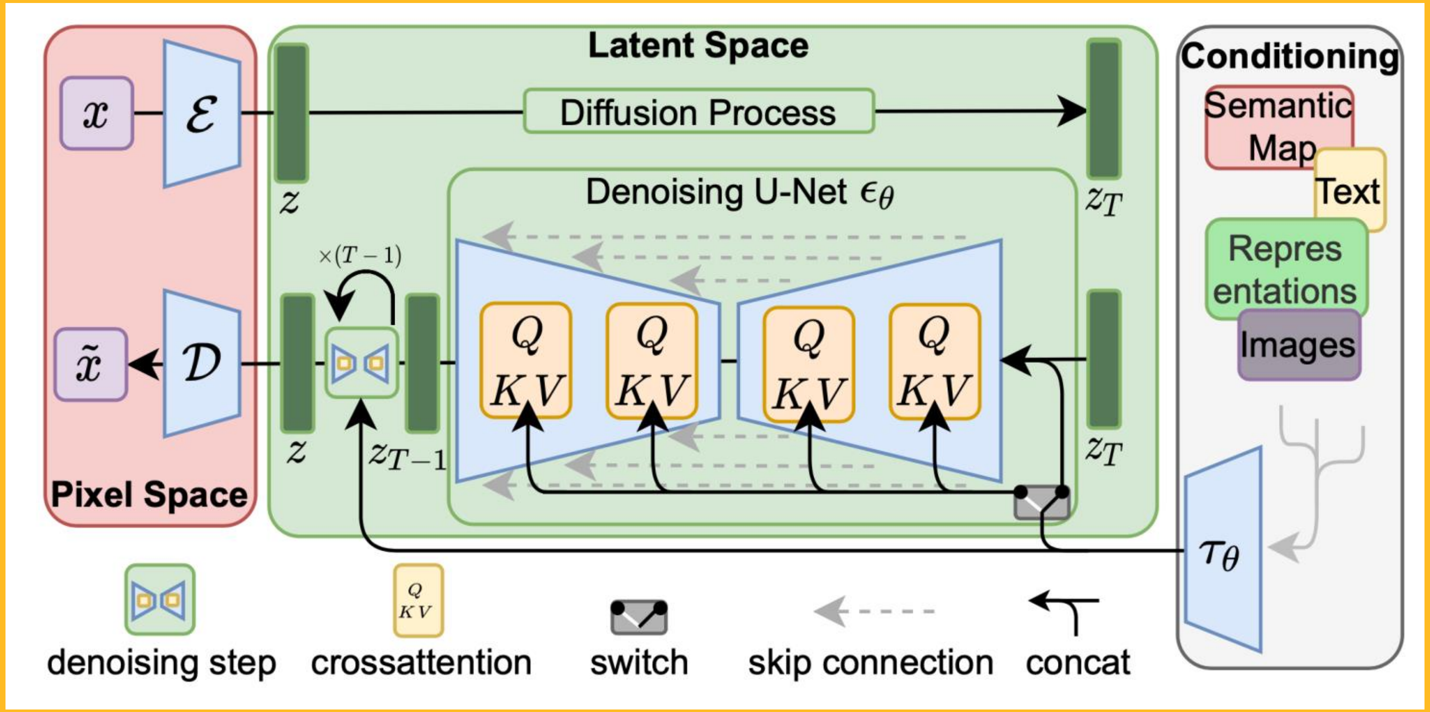
+

A picture of a cat

Now the noise and the text prompt serve as input for the generator

Input

Output



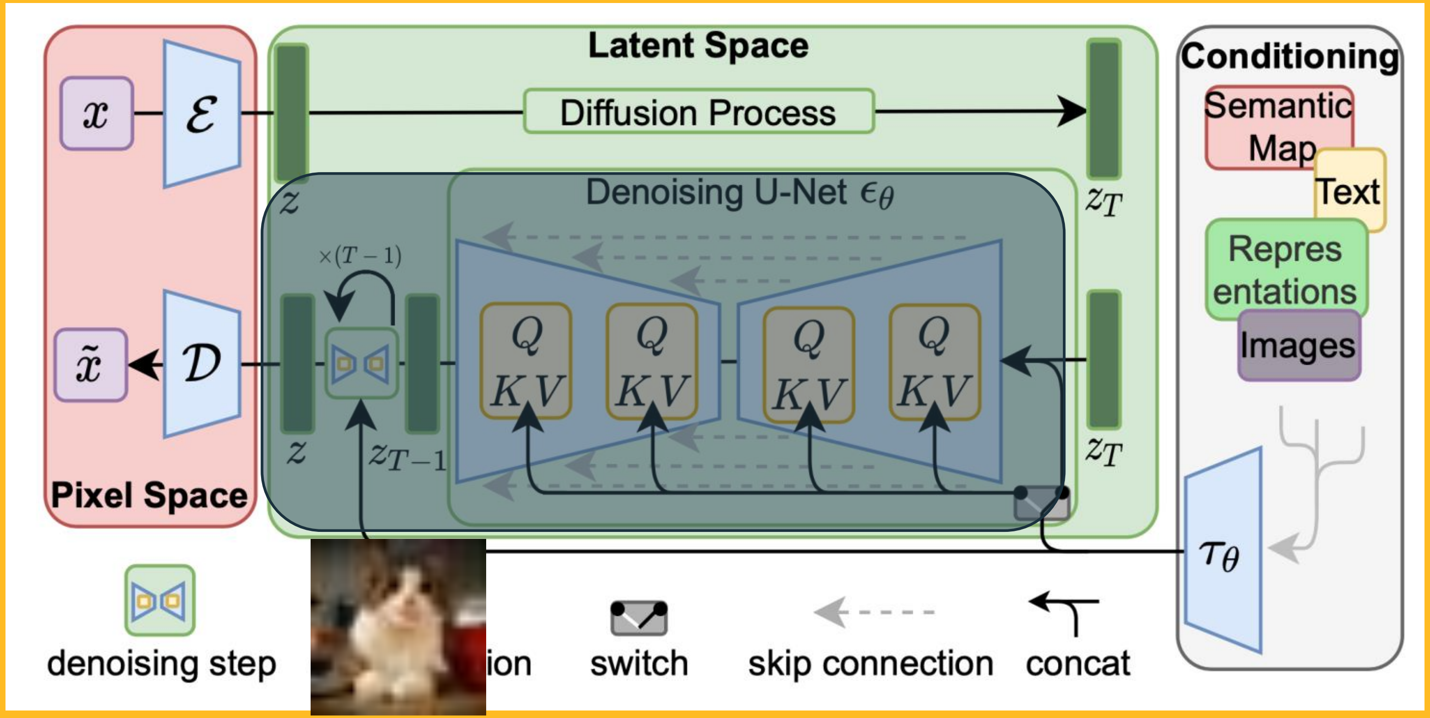
+

1.2	-0.1	4.3	3.2
0.4	2.5	-0.9	0.5
2.1	0.3	0.1	0.4

Now the noise and the text prompt serve as input for the generator, or rather an embedding of the prompt

Input

Output



+

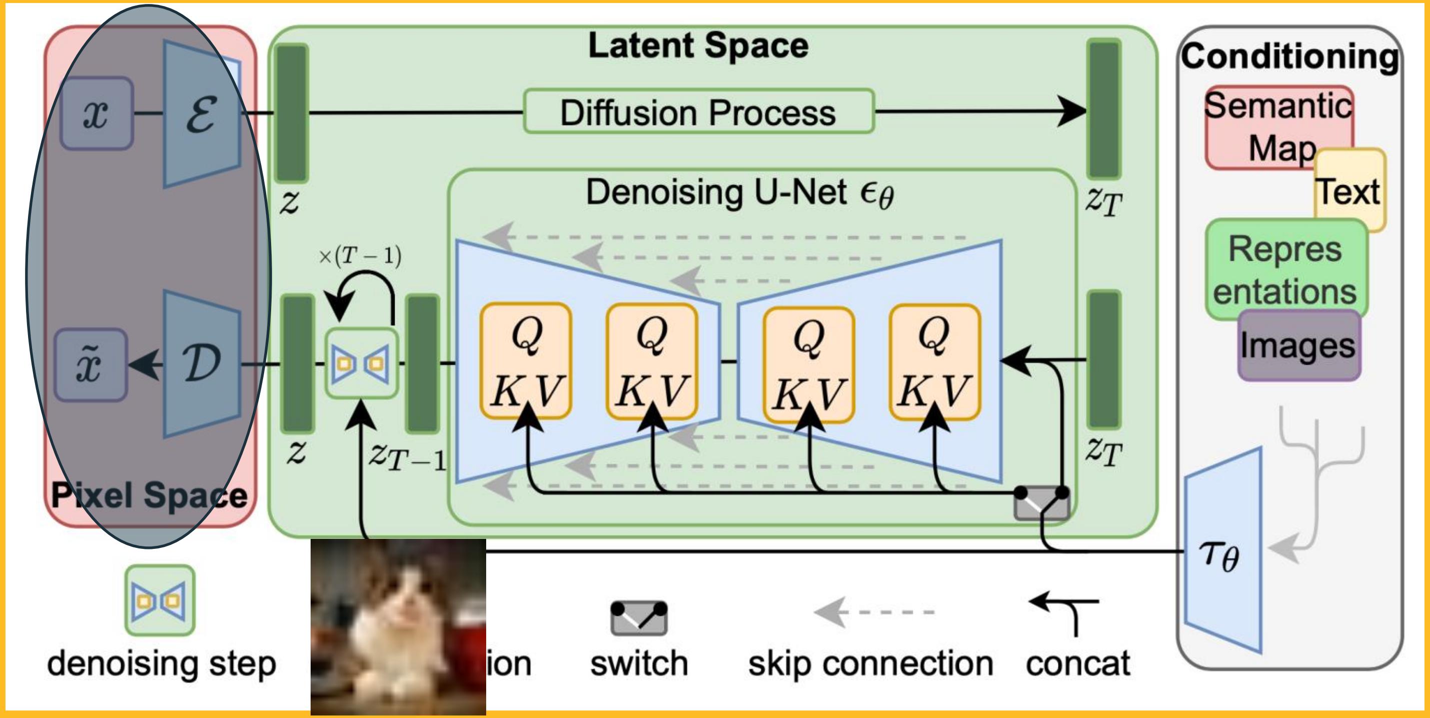
A picture of a cat

For 50 steps: the denoiser module tries to predict which noise was added to the picture. The embedded text guides the model in this process.

Encoder / Decoder (de)compression

Input →

Output←



+

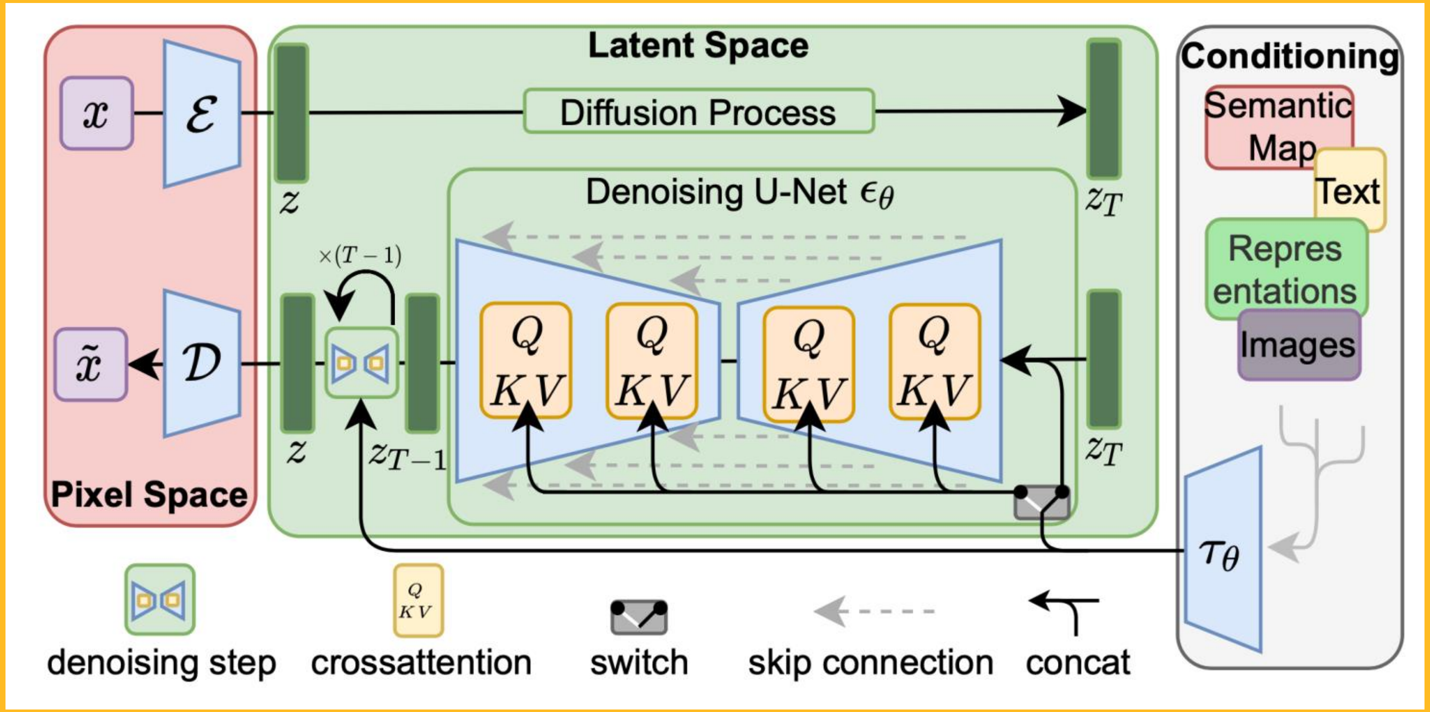
A picture of a cat

The decoder decompresses the image into its original size

Model Architecture

Input

Output



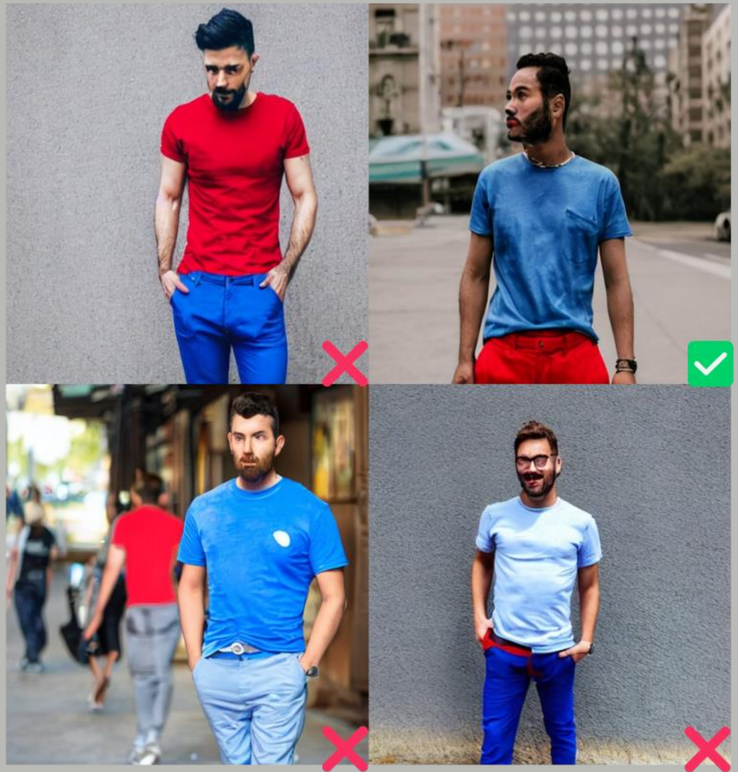
+

A picture of a cat

Shortcomings

Prompt failure

a man wearing a blue t shirt and red pants



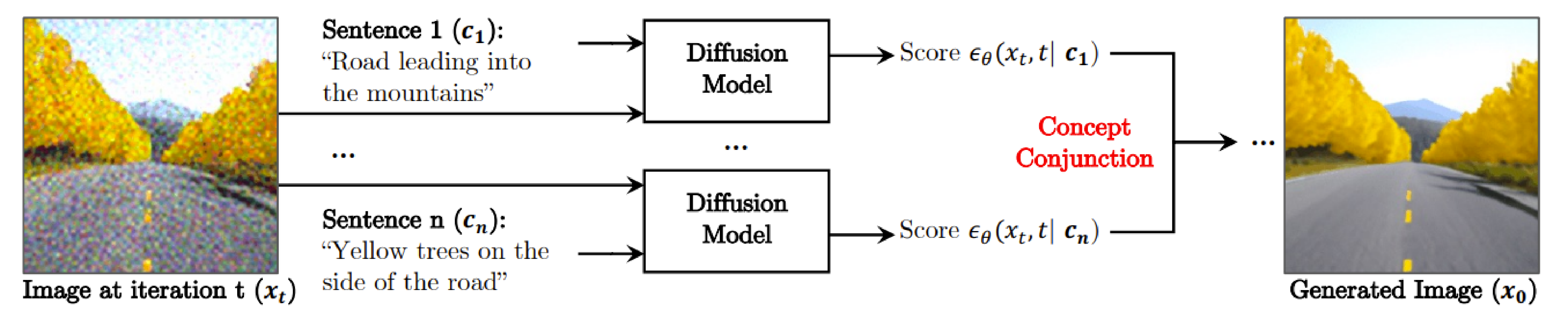
Brand failure

a man wearing a Hugo Boss polo

Approach

Metyis

Composable Diffusion



Diffusion models capable of generating simple prompts, can we stack diffusion models using AND or NOT statements?

By combining the score-functions of multiple diffusion models, we can guide the diffusion process with multiple conjunctions

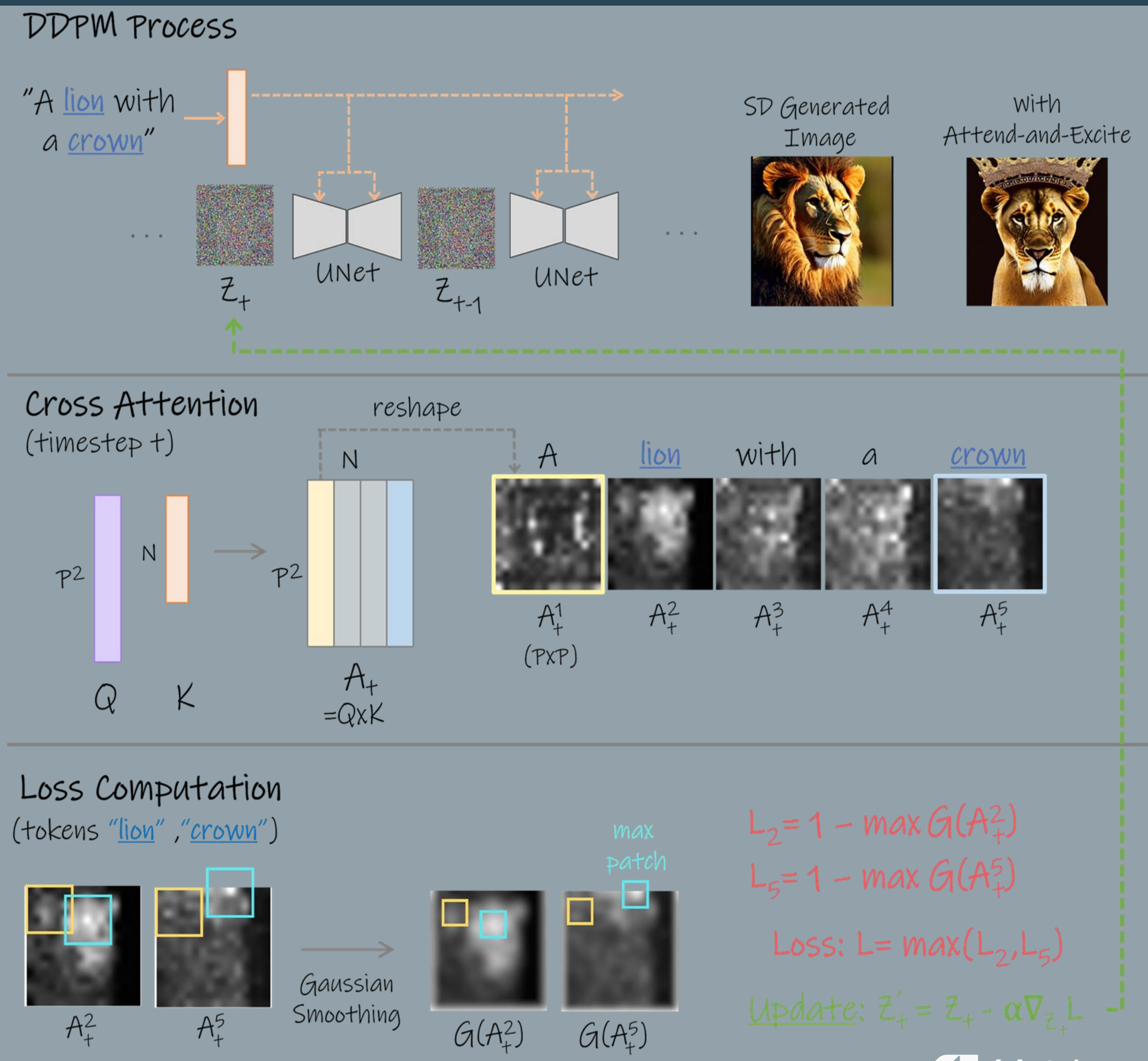
Metyis

Attend and Excite

Embedding method overrules attention blocks

Can we force words to be included?

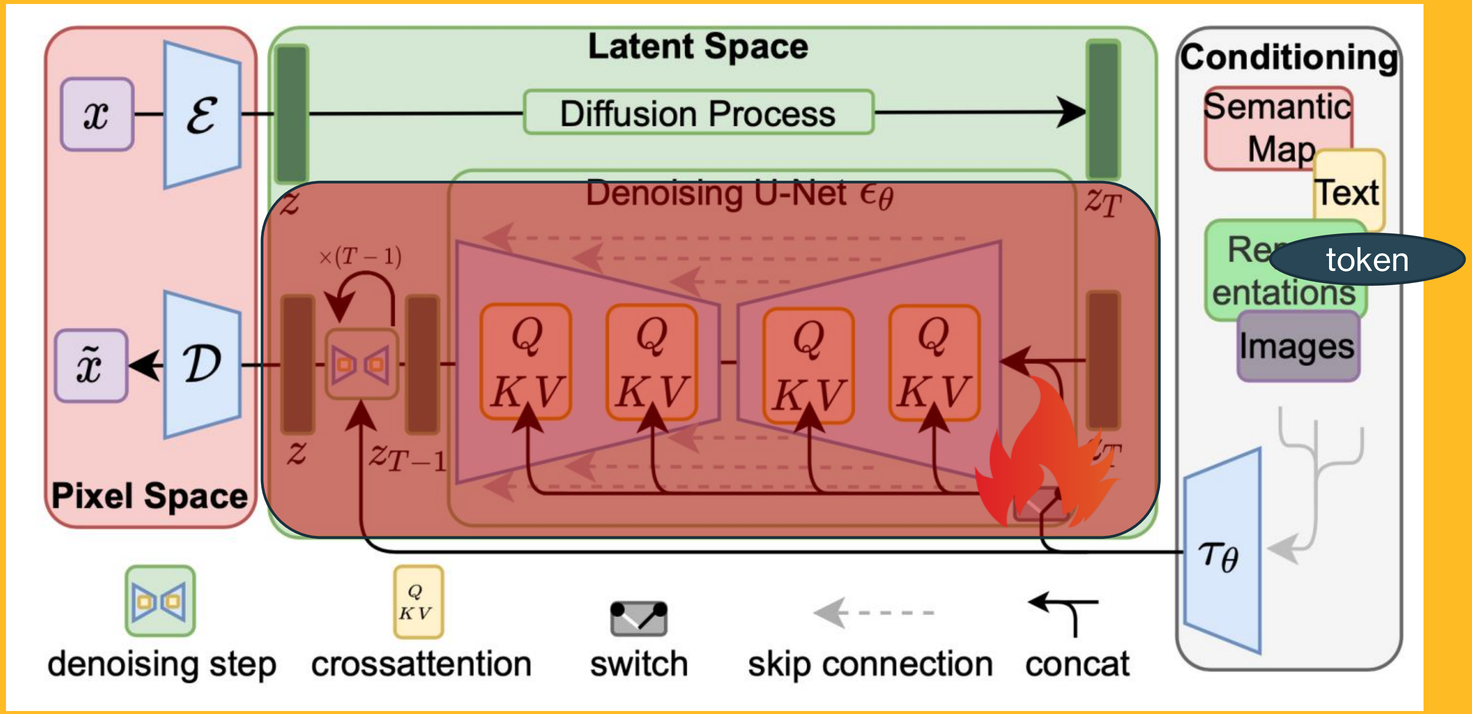
- Reweigh attention over excited words
- Increase attention for most neglected subject token



Metyis

Personalizing Diffusion models

Dreambooth

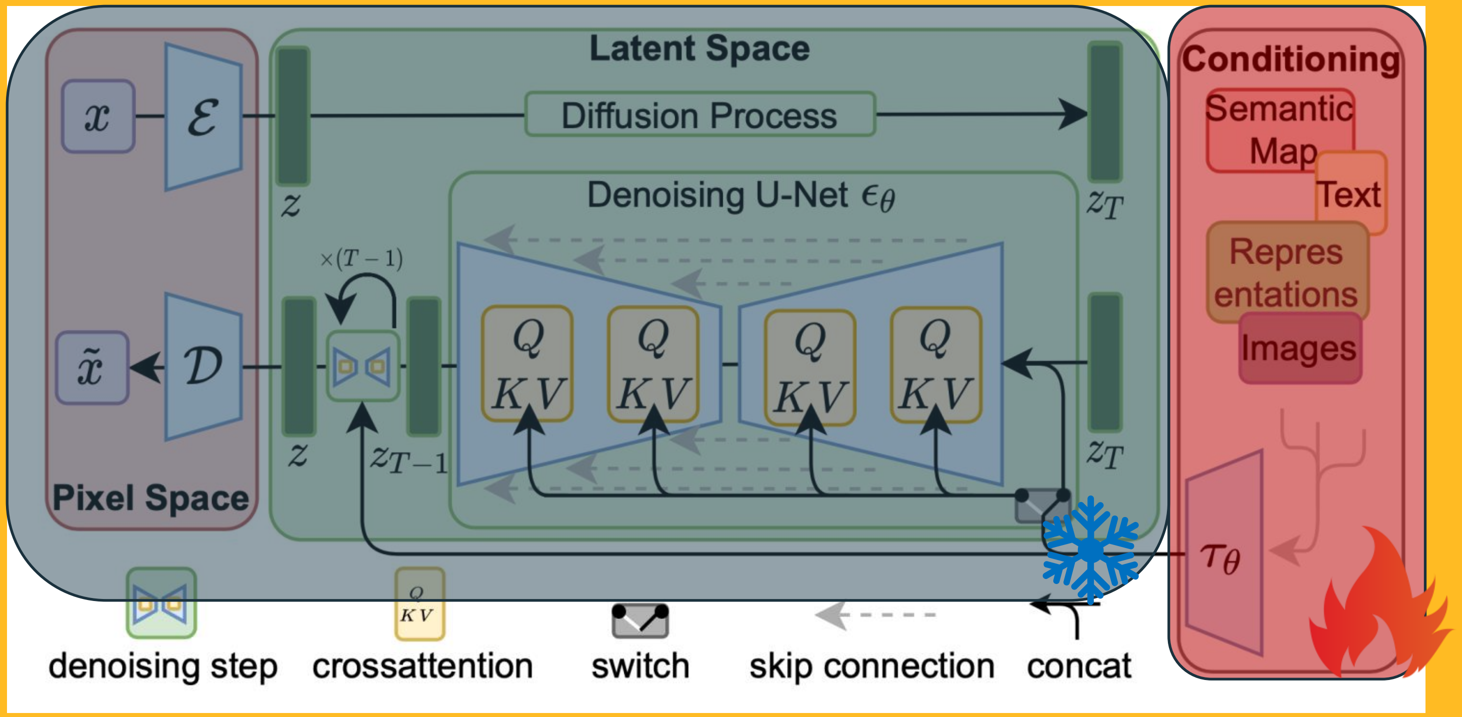


Train optimal weights for specified concept

\+ Best quality

− Very expensive
Breaks the model (overfitting)

Textual Inversion



Insert new token

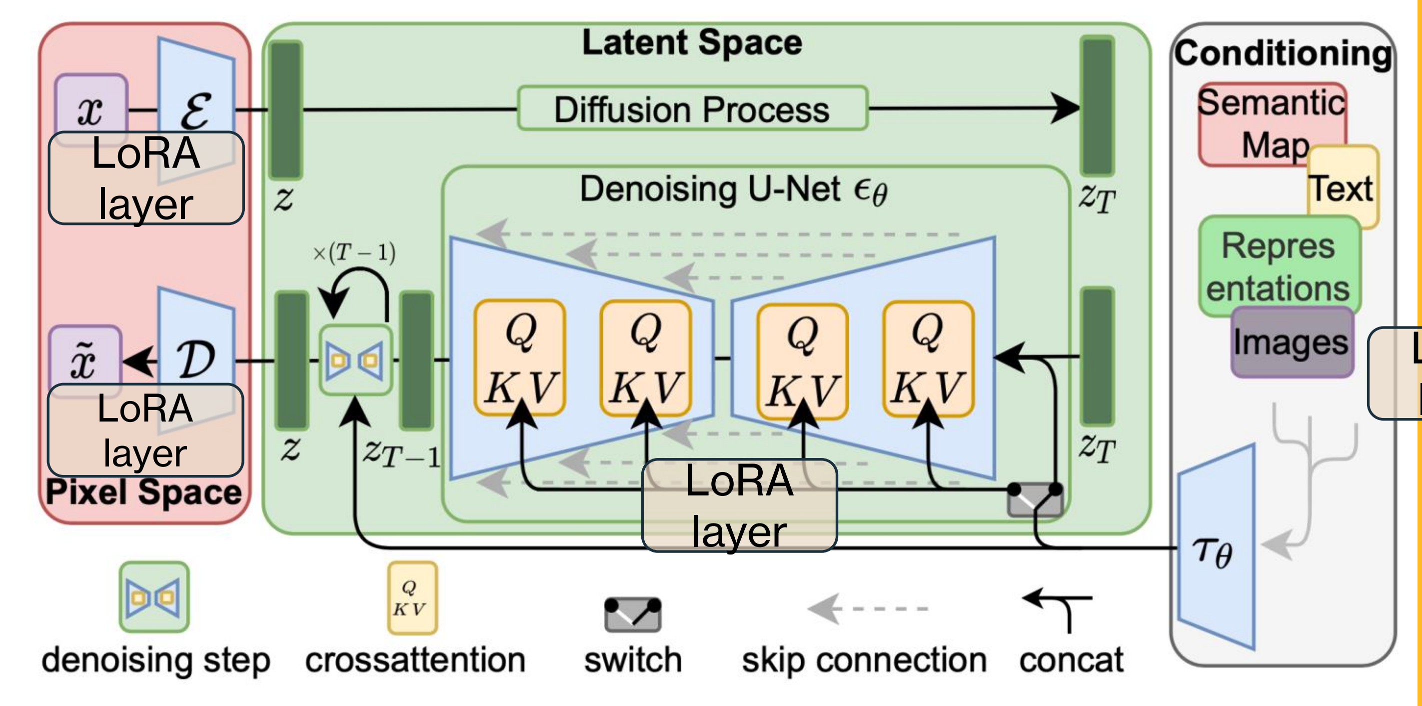
Train optimal embeddings for this token

\+ Less expensive
Scalable

− Lower quality

Metyis

Low Rank Adaptation



Inject low-rank layers

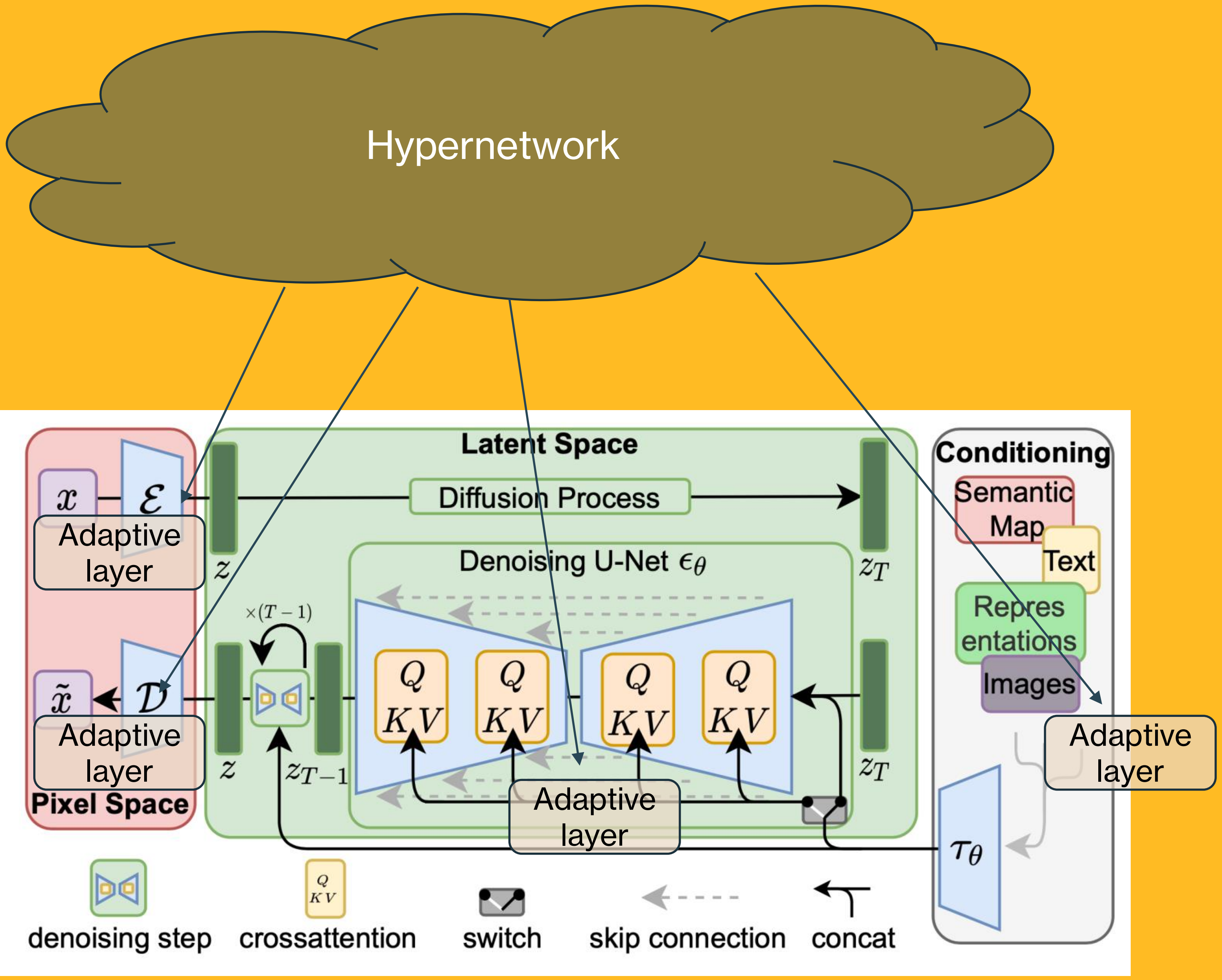
Train optimal embeddings for these layers

+ Better quality
  Scalable

− Higher chance of failure

Hypernetwork

Maybe we should not do this one? People report very bad results, and its functionality has been replaced by LoRA



Q & A