ASHOKA UNIVERSITY

# StyleCLIP

Text-Driven Manipulation of StyleGAN Imagery

---

**Karnav**

ug24

# Confidentiality

I kindly request that pictures of figures and illustrations put on this presentation (marked "confidential") not be circulated owing to reasons of confidentiality.

# Statement of the Problem

How do you manipulate an image with a text prompt, without having to do math?

## Objectives

- Understand the task.
- Look at possible approaches to solving it.
- Use methods that have no ambiguity i.e. that are clear and easily understandable.

# Questions

How do you internalize what the (possibly complex, multi-layered) image is showing you?

How do you understand what the text prompt says?

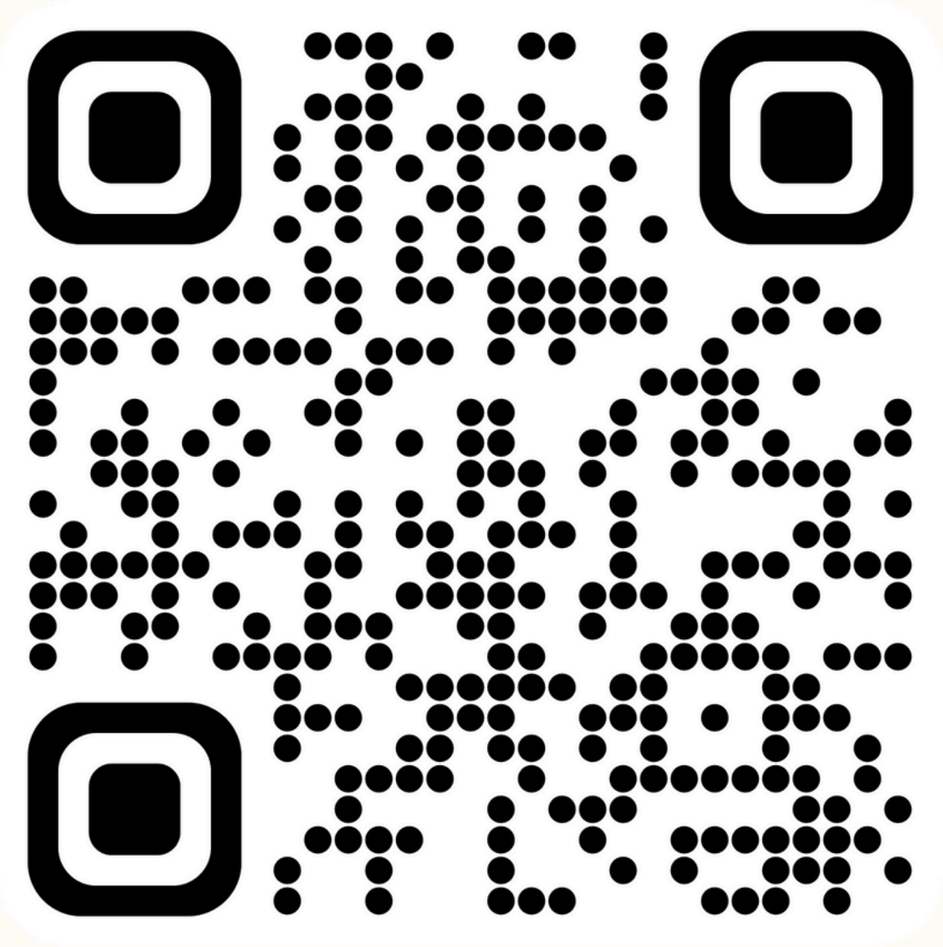
How do you translate a text prompt into a change in the image?

# Note on terminology

| | |
|---|---|
| **Layers** | • Background, foreground |
| **Features** | • Columns, variables |
| **Loss** | • The difference between actual and predicted that we use to improve the model |
| **Style** | • Everything except the image's content - Van Gogh-y, abstract, black-and-white.<br>• Also the y-vector space |
| **Levels** | • Coarse, middle and fine details |

# Q&A

Any Questions at this stage?

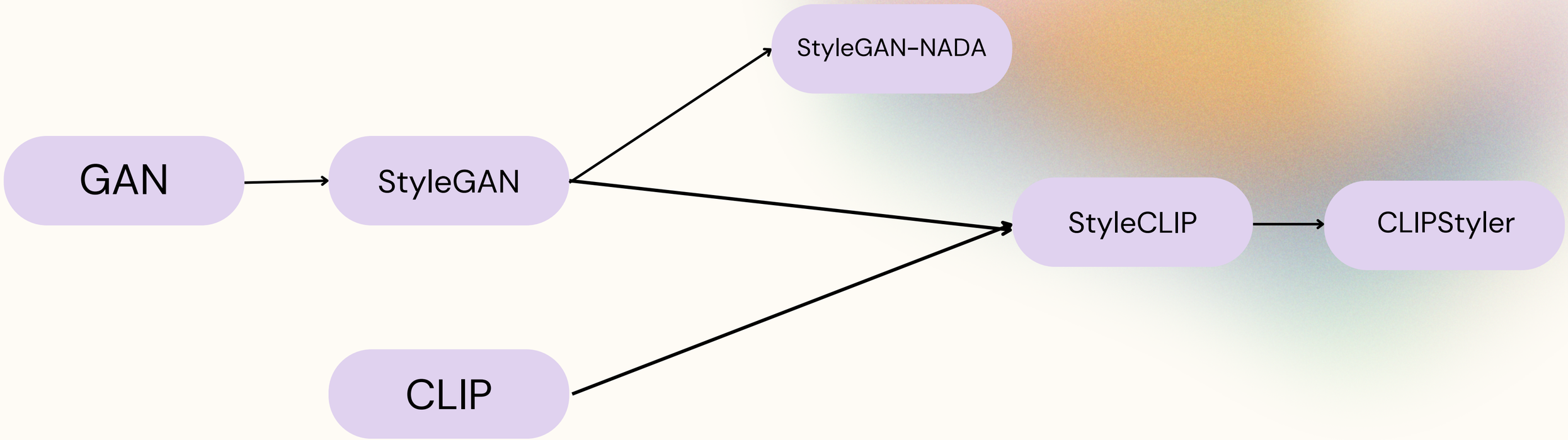


# The paper

Today we are going to be taking a look at **"StyleCLIP: Text-Driven Manipulation of StyleGAN Imagery"** a paper by Or Patashnik and co

*Scan the QR code on the side to view the paper*

# Gentlemen, a short view back to the past



# Under the trenchcoat

| | |
|---|---|
| **GAN** | • The framework of improving image generation by training the cop alongside the thief |
| **StyleGAN** | • Rework of the generator (thief) to disentangle high-level features into their own vectors/codes , and operating on them at progressively more detailed layers |
| **CLIP** | • Mapping text prompts and images to the same mathematical space |
| **StyleCLIP** | • Combining StyleGAN and CLIP |
| **CLIPStyler** | • Substituting the highest style layer of StyleGAN with a CLIP generated imagining of the style |

# GAN: Generative Adversarial Networks

Train two models: the generator G and the discriminant D

G trains on the image dataset to learn to generate new images that fit the criteria

D trains on the image dataset and on G's output to learn to distinguish between the two

$$\min_G \max_D V(D,G) = \min_G \max_D \left(E_{x\sim P_{data}(x)}[logD(x)] + E_{z\sim P_z(z)}[log(1-D(G(z)))]\right) \quad (6)$$

# GAN: Generative Adversarial Networks

$$L(\hat{y}, y) = [y.log\hat{y} + (1 - y).log(1 - \hat{y})]$$

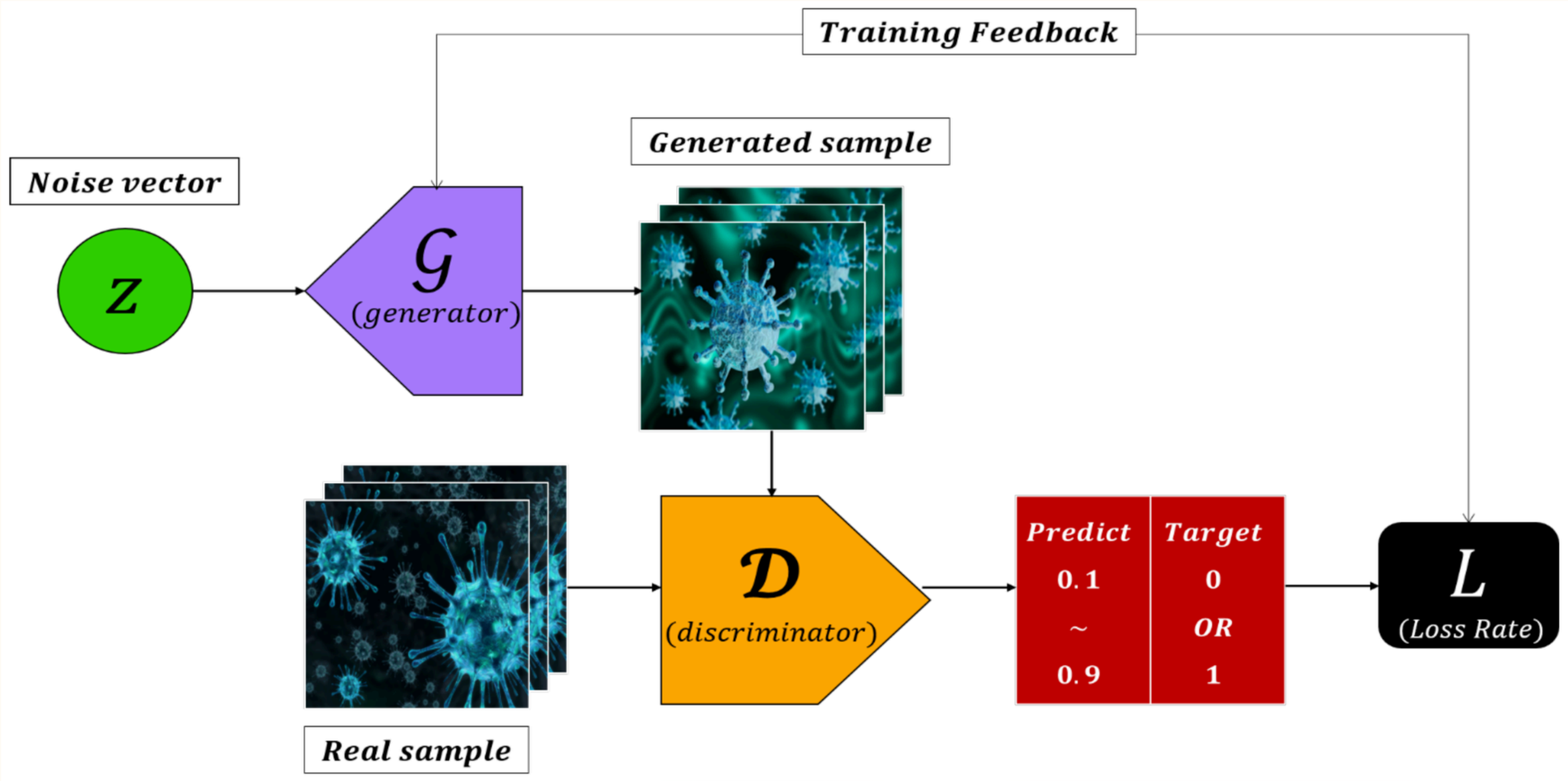
Original data  Reconstructed data

$$L^{(D)} = \max[log(D(x)) + log(1 - D(G(z)))] \quad (3)$$

$$L^{(G)} = \min[log(D(x)) + log(1 - D(G(z)))] \quad (4)$$

$$L = \min_{G} \max_{D}[log(D(x)) + log(1 - D(G(z)))] \quad (5)$$

# GAN: Generative Adversarial Networks

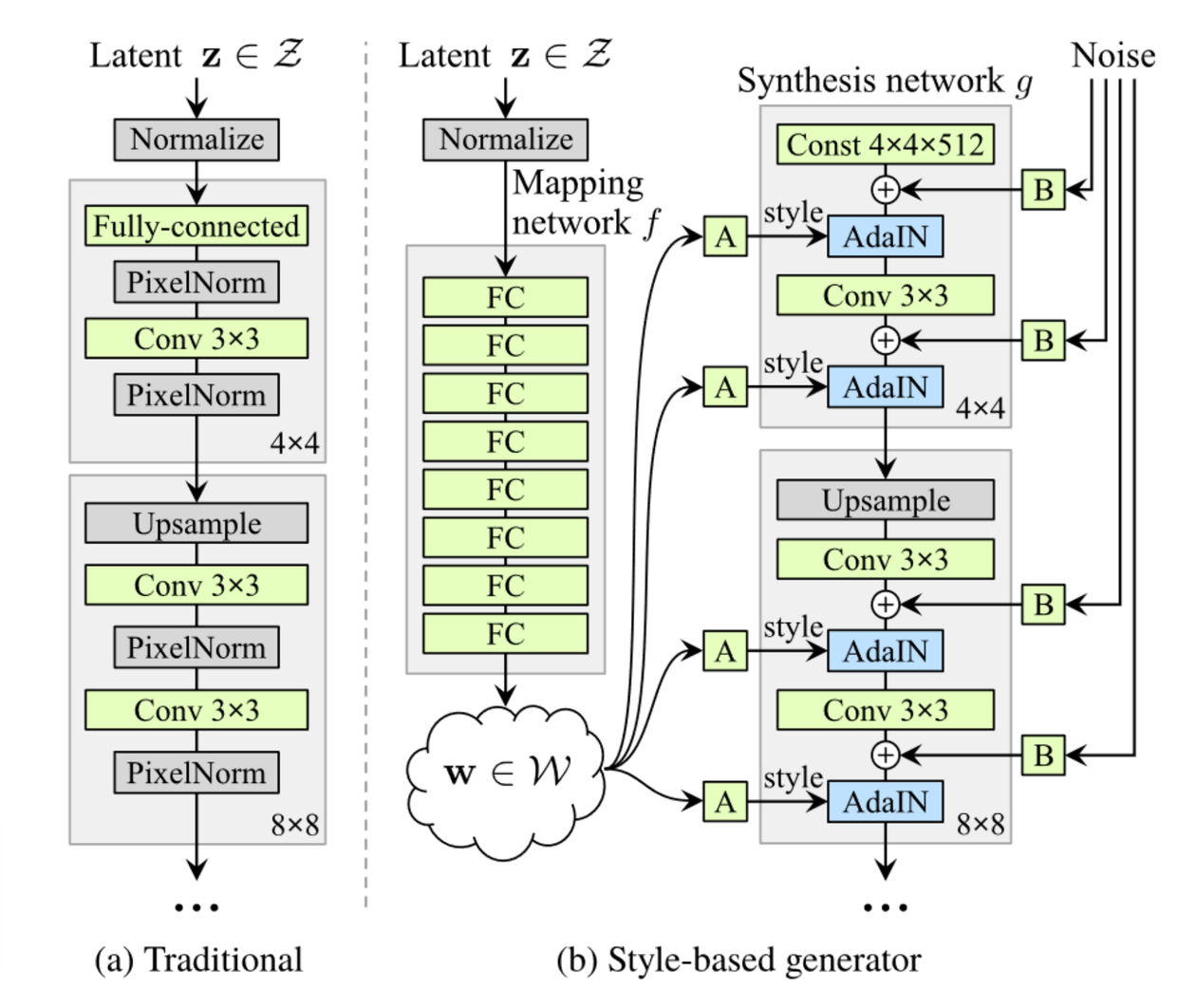


# Where are we?

| | |
|---|---|
| **GAN** | • The framework of improving image generation by training the cop alongside the thief |
| **StyleGAN** | • Rework of the generator (thief) to disentangle high-level features into their own vectors/codes , and operating on them at progressively more detailed layers |

# StyleGAN: A Style Based Generator Architecture

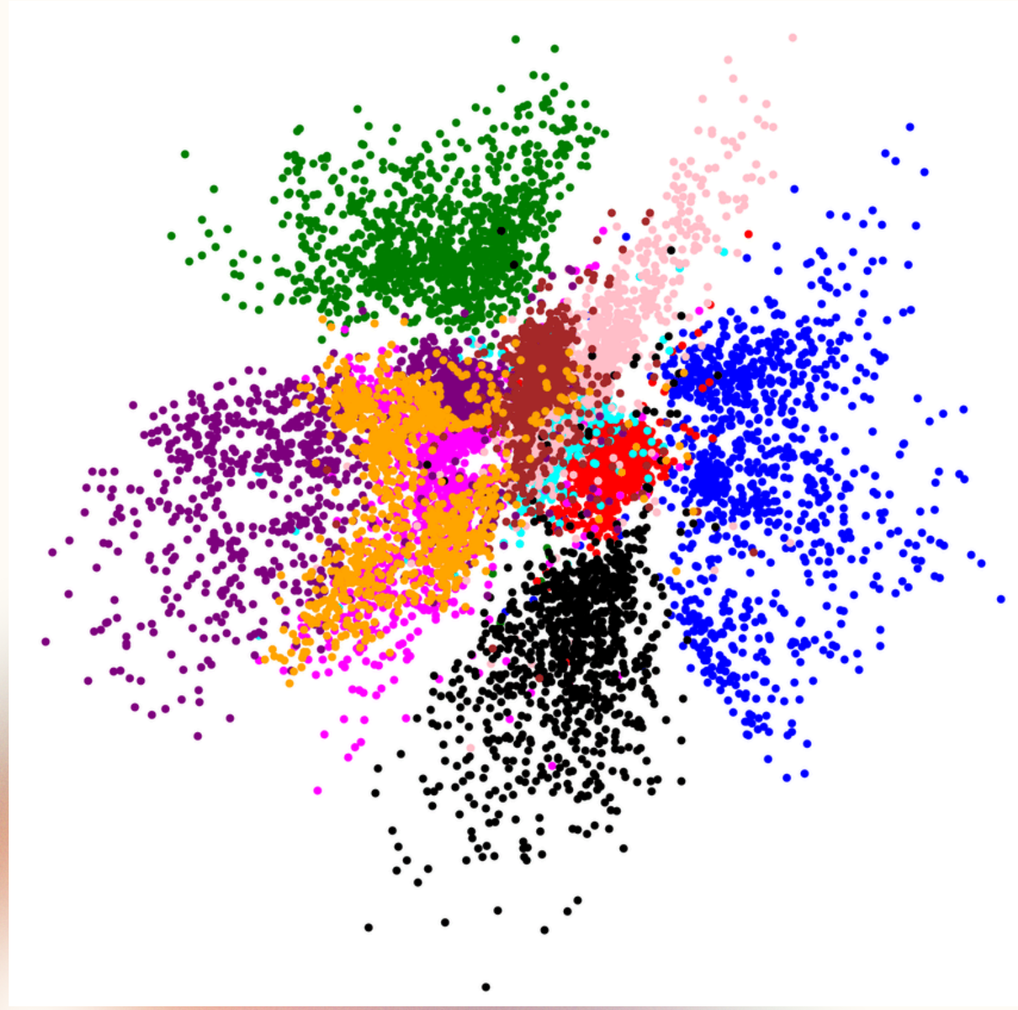
Aims to achieve the same thing as GAN but like, better



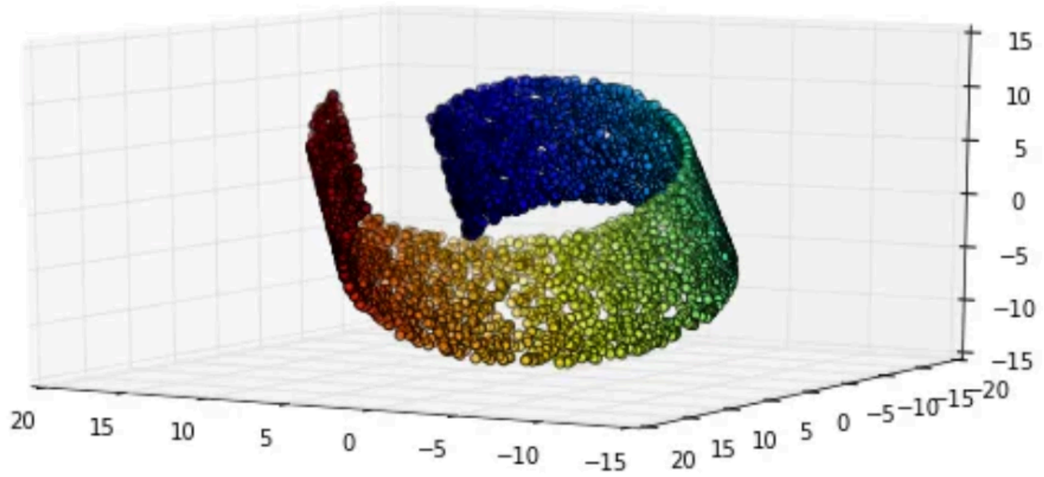
# Detour: Latent Space

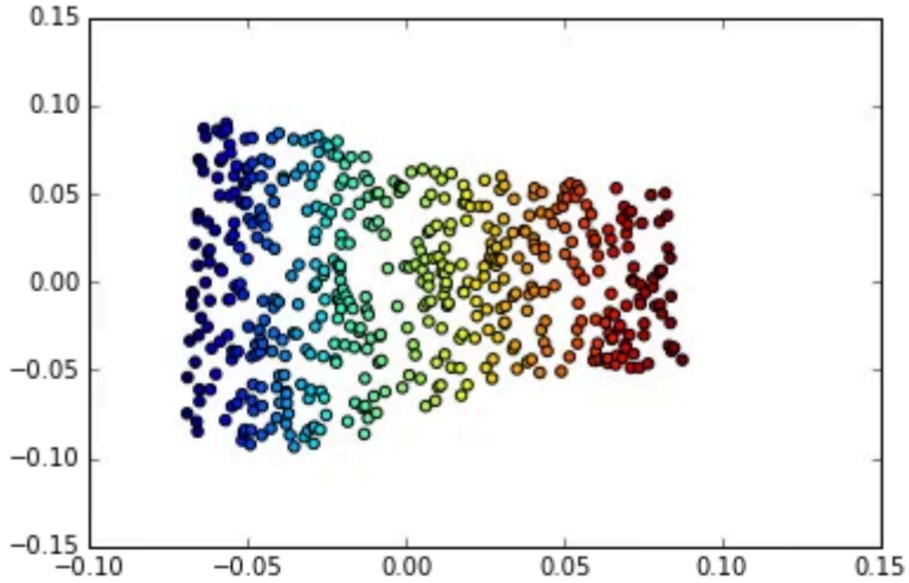
A latent code (latent vectors or image code or z-vectors) compresses the image from a height x width dimension vector to a more manageable level

Similar images have similar latent codes and the distance between vectors in the latent space represents the difference between images



# Detour: Latent Space




3D Representation of Swiss Roll vs. 2D Representation of same data. Example from https://datascience.stackexchange.com/a/5698

# Detour: W space

StyleGAN uses w-vectors instead of traditional z-vectors, where it passes the latent code through a mapping network and separates out each attribute into its own disentangled w-vector

P.S. The shape of both Z and W space in the StyleGAN architecture is 512-D. Also, the distribution of Z is Gaussian but W space does not follow any specific distribution.

Extended w-space is the w-space but with a separate w-code for each layer instead of just one w-vector, which enables more fine-tuned manipulation of each attribute

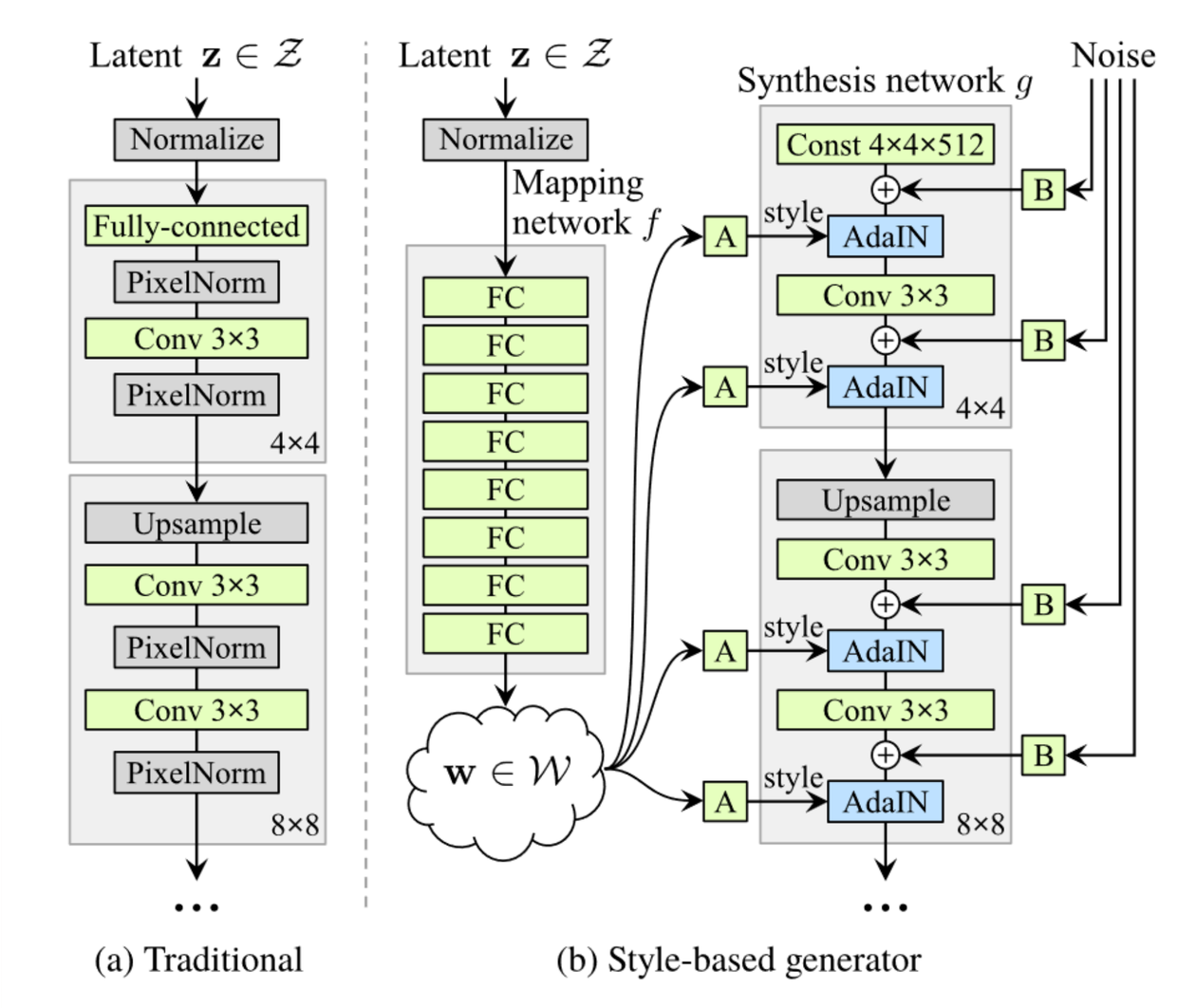
# StyleGAN: A Style Based Generator Architecture

Aims to achieve the same thing as GAN but like, better

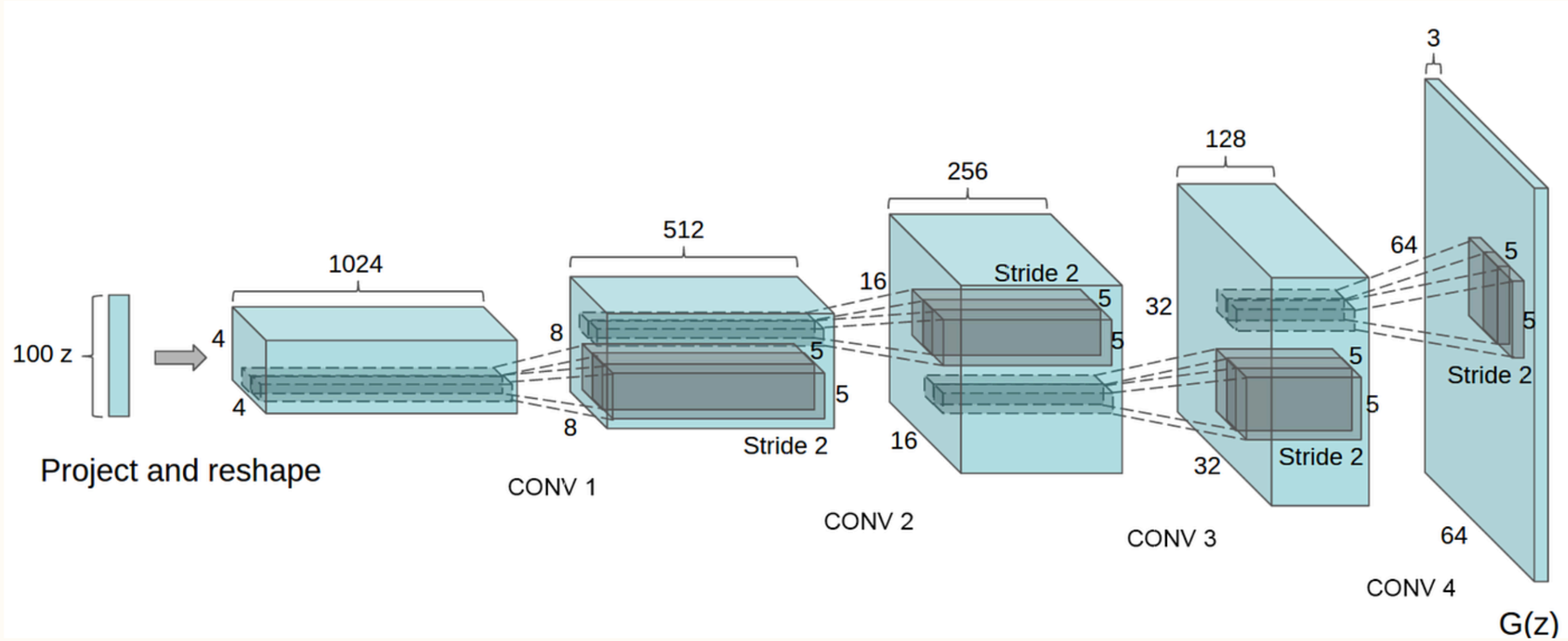
> Start with a constant random latent code in the z-space

> Convert it to a style code in the w-space

> AdaIN it to a "style" y-vector in the StyleSpace S, which maps the attributes of the w-space to specific local impacts in the final image, at all three levels progressively



# StyleGAN: A Style Based Generator Architecture



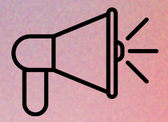
# Q&A

Any Questions at this stage?

# Consequences of latent space

GAN inversion encoders

Take a latent code and encode it into an image instead of compressing an image to a latent code



Semantic editing

Picking a direction in the latent space and moving the latent code in that generation to effect a change in the image

Latent space interpolation

FaceMash-esque generation of a latent code and image that lies "between" two images

# Where are we?

| | |
|---|---|
| **GAN** | • The framework of improving image generation by training the cop alongside the thief |
| **StyleGAN** | • Rework of the generator (thief) to disentangle high-level features into their own vectors/codes , and operating on them at progressively more detailed layers |
| **CLIP** | • Mapping text prompts to the image space |

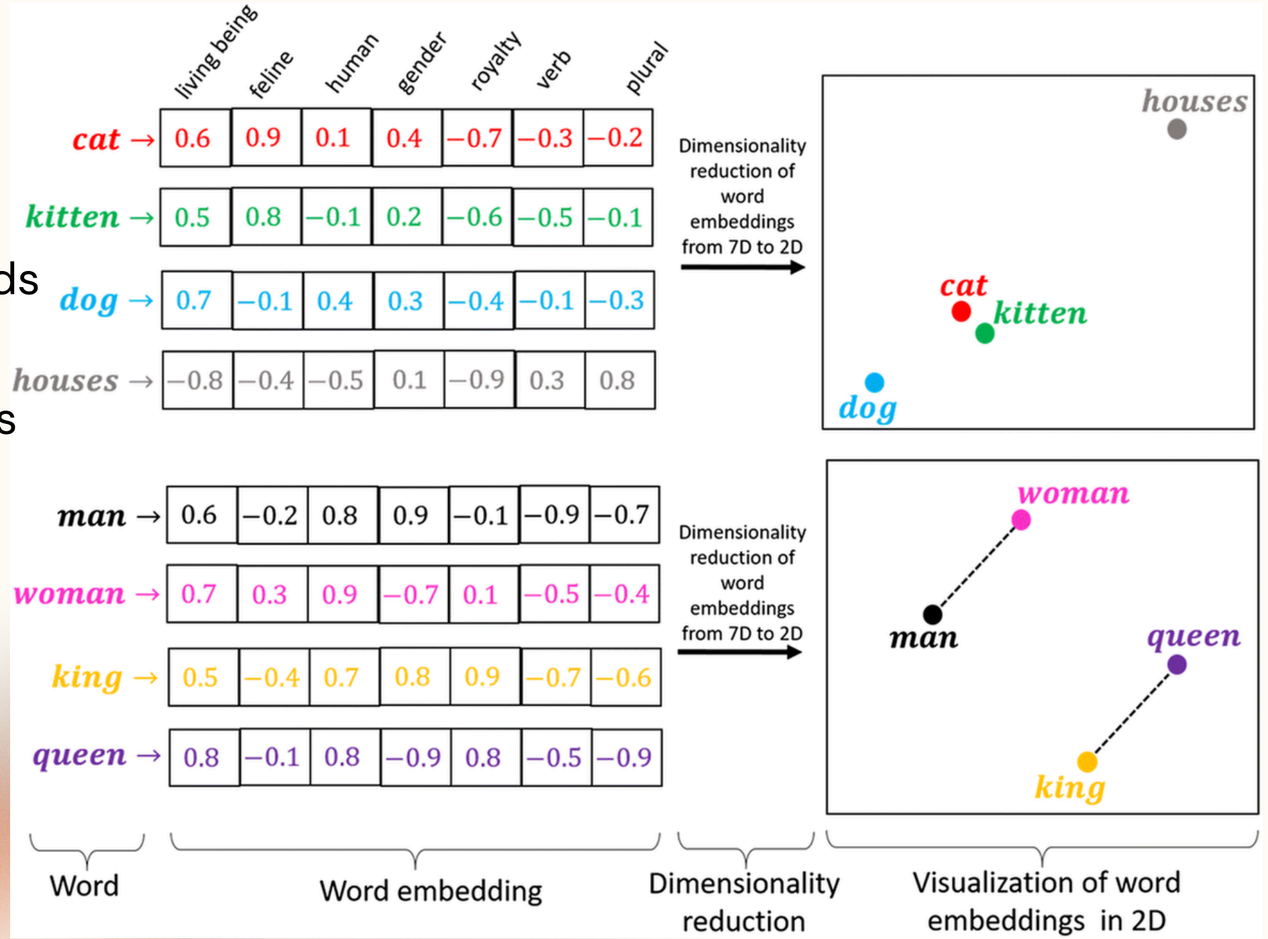
# CLIP: Contrastive Language-Image Pre-training

<- remember Semantic Editing?

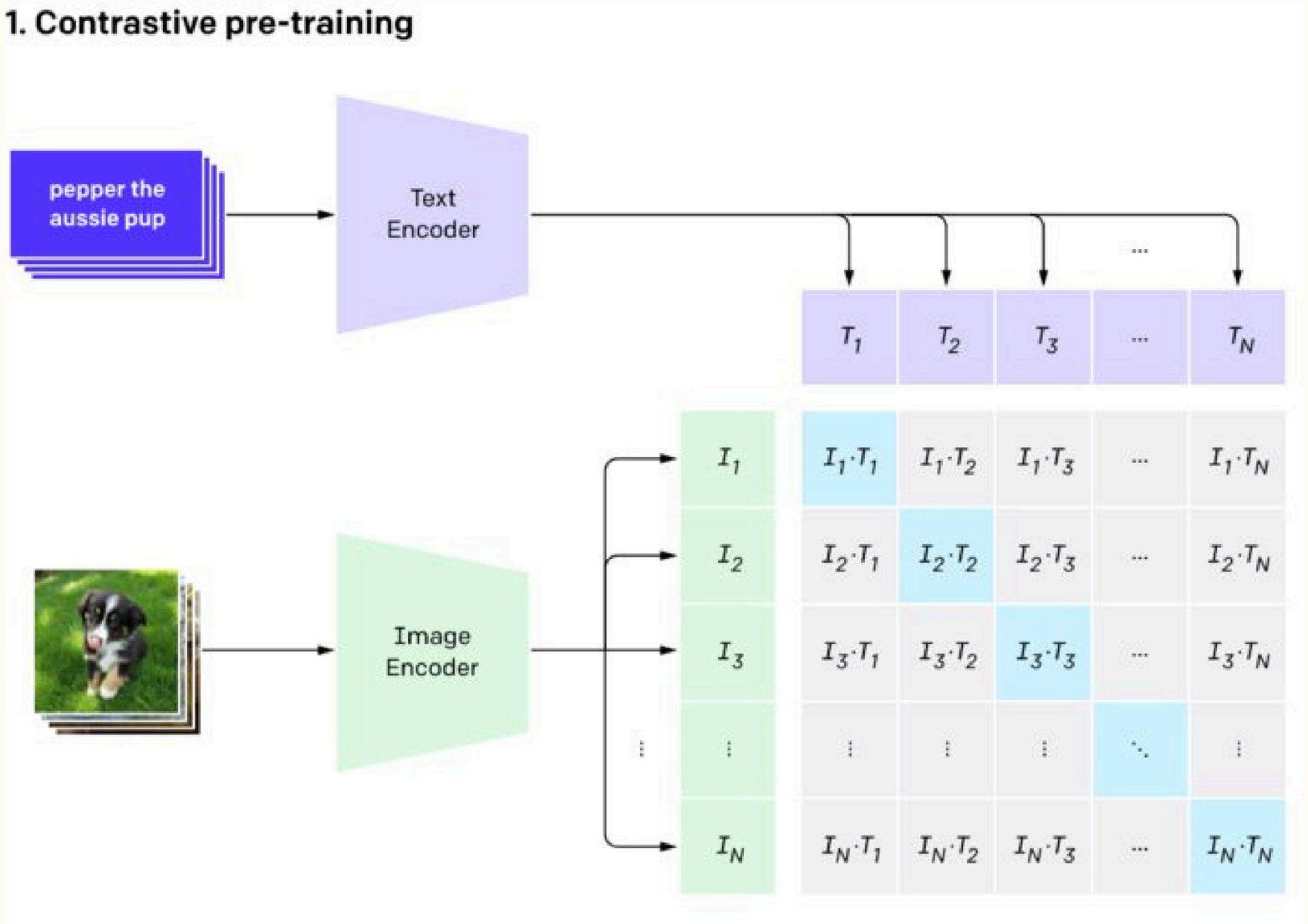
Takes a text prompt and and an image and embeds them into the same space

Uses cosine similarity to train itself to output texts and images of the same things as similar vectors

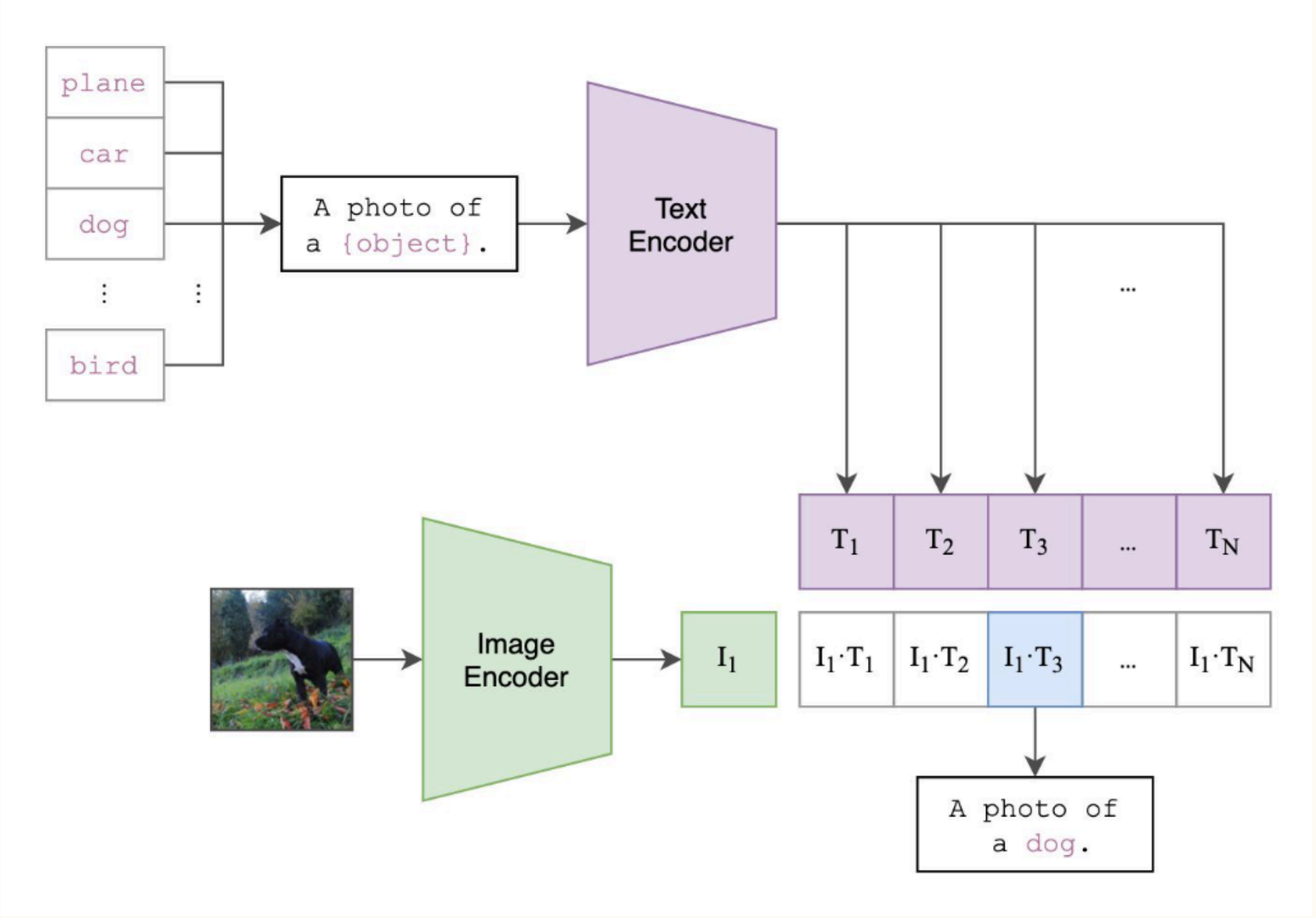
Can be used to train and output z-vectors that can then be used as inputs to GANs



# CLIP: Contrastive Language–Image Pre-training



# CLIP: Contrastive Language-Image Pre-training



# Q&A

Any Questions at this stage?

# Where are we?

| | |
|---|---|
| **GAN** | • The framework of improving image generation by training the cop alongside the thief |
| **StyleGAN** | • Rework of the generator (thief) to disentangle high-level features into their own vectors/codes , and operating on them at progressively more detailed layers |
| **CLIP** | • Mapping text prompts to the image space |
| **StyleCLIP** | • Combining StyleGAN and CLIP |

# StyleCLIP: Text-Driven Manipulation of StyleGAN Imagery

CLIP -> StyleGAN

1. Text-guided latent optimization, where a CLIP model is used as a loss network

2. A latent residual mapper, trained for a specific text prompt. Given a starting point in latent space (the input image to be manipulated), the mapper yields a local step in latent space.

3. A method for mapping a text prompt into an input-agnostic (global) direction in StyleGAN's style space, providing control over the manipulation strength as well as the degree of disentanglement.

# StyleCLIP: Text-Driven Manipulation of StyleGAN Imagery

1. Text-guided latent optimization, where a CLIP model is used as a loss network, and there's backpropagation over ever (Text prompt, image) pair

$$\arg\min_{w \in \mathcal{W}+} D_{\text{CLIP}}(G(w), t) + \lambda_{\text{L2}} \|w - w_s\|_2 + \lambda_{\text{ID}} \mathcal{L}_{\text{ID}}(w), \tag{1}$$

where $G$ is a pretrained StyleGAN[1] generator and $D_{\text{CLIP}}$ is the cosine distance between the CLIP embeddings of its two arguments. Similarity to the input image is controlled by the $L_2$ distance in latent space, and by the identity loss [46]:

$$\mathcal{L}_{\text{ID}}(w) = 1 - \langle R(G(w_s)), R(G(w)) \rangle, \tag{2}$$

where $R$ is a pretrained ArcFace [11] network for face recognition, and $\langle \cdot, \cdot \rangle$ computes the cosine similarity be-

# StyleCLIP: Text-Driven Manipulation of StyleGAN Imagery

2. A latent residual mapper, trained for a specific text prompt. Given a starting point in latent space (the input image to be manipulated), the mapper yields a local step in latent space.

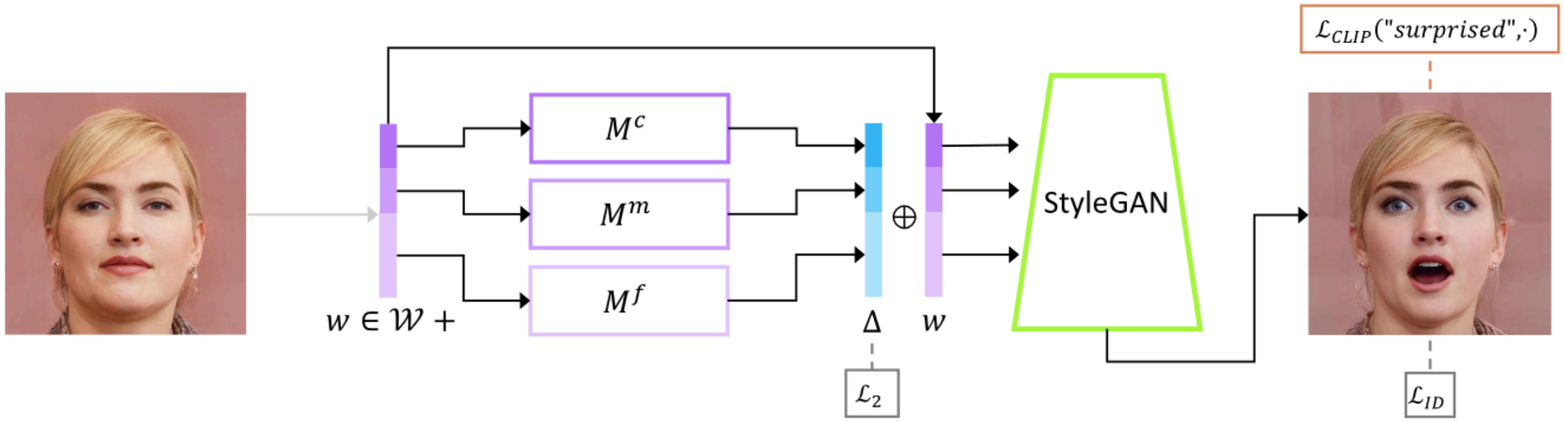



Figure 3. The architecture of our text-guided mapper (using the text prompt "grey hair", in this example). The source image (left) is inverted into a latent code $w$. Three separate mapping functions are trained to generate residuals (in blue) that are added to $w$ to yield the target code, from which a pretrained StyleGAN (in green) generates an image (right), assessed by the CLIP and identity losses.

| | Mohawk | Afro | Bob-cut | Curly | Beyonce | Taylor Swift | Surprised | Purple hair |
|---|---|---|---|---|---|---|---|---|
| **Mean** | 0.82 | 0.84 | 0.82 | 0.84 | 0.83 | 0.77 | 0.79 | 0.73 |
| **Std** | 0.096 | 0.085 | 0.095 | 0.088 | 0.081 | 0.107 | 0.893 | 0.145 |

Table 2. Average cosine similarity between manipulation directions obtained from mappers trained using differnt text prompts.

# StyleCLIP: Text-Driven Manipulation of StyleGAN Imagery

2. A latent residual mapper, trained for a specific text prompt. Given a starting point in latent space (the input image to be manipulated), the mapper yields a local step in latent space.

age. The CLIP loss, $\mathcal{L}_{\text{CLIP}}(w)$ guides the mapper to minimize the cosine distance in the CLIP latent space:

$$\mathcal{L}_{\text{CLIP}}(w) = D_{\text{CLIP}}(G(w + M_t(w)), t), \tag{4}$$

where $G$ denotes again the pretrained StyleGAN generator. To preserve the visual attributes of the original input image, we minimize the $L_2$ norm of the manipulation step in the latent space. Finally, for edits that require identity preservation, we use the identity loss defined in eq. (2). Our total loss function is a weighted combination of these losses:

$$\mathcal{L}(w) = \mathcal{L}_{\text{CLIP}}(w) + \lambda_{L2} \|M_t(w)\|_2 + \lambda_{\text{ID}}\mathcal{L}_{\text{ID}}(w). \tag{5}$$

# StyleCLIP: Text-Driven Manipulation of StyleGAN Imagery

3. A method for mapping a text prompt into an input-agnostic (global) direction in StyleGAN's style space, providing control over the manipulation strength as well as the degree of disentanglement.

Let $s \in \mathcal{S}$ denote a style code, and $G(s)$ the corresponding generated image. Given a text prompt indicating a desired attribute, we seek a manipulation direction $\Delta s$, such that $G(s + \alpha \Delta s)$ yields an image where that attribute is introduced or amplified, without significantly affecting other attributes. The manipulation strength is controlled by $\alpha$. Our high-level idea is to first use the CLIP text encoder to obtain a vector $\Delta t$ in CLIP's joint language-image embedding and then map this vector into a manipulation direction $\Delta s$ in $\mathcal{S}$. A stable $\Delta t$ is obtained from natural language, using prompt engineering, as described below. The corresponding direction $\Delta s$ is then determined by assessing the relevance of each style channel to the target attribute.

# StyleCLIP: Text-Driven Manipulation of StyleGAN Imagery

3. A method for mapping a text prompt into an input-agnostic (global) direction in StyleGAN's style space, providing control over the manipulation strength as well as the degree of disentanglement.

$$\Delta s = \begin{cases} \Delta i_c \cdot \Delta i & \text{if } |\Delta i_c \cdot \Delta i| \geq \beta \\ 0 & \text{otherwise} \end{cases} \qquad (7)$$

# Other advances

- CLIPStyler takes the style from one image and transplants it onto another image, "Mona Lisa in the style of Van Gogh"
- StyleGAN-NADA generates images like StyleGAN but is trained purely on CLIP's text-image embeddings, without any images
- Upcoming research

# Q&A Session

Any final thoughts or comments?

# Thank you!

---

Feel free to drop me an email at **karnav.popat_ug24@ashoka.edu.in** for any questions or concerns.