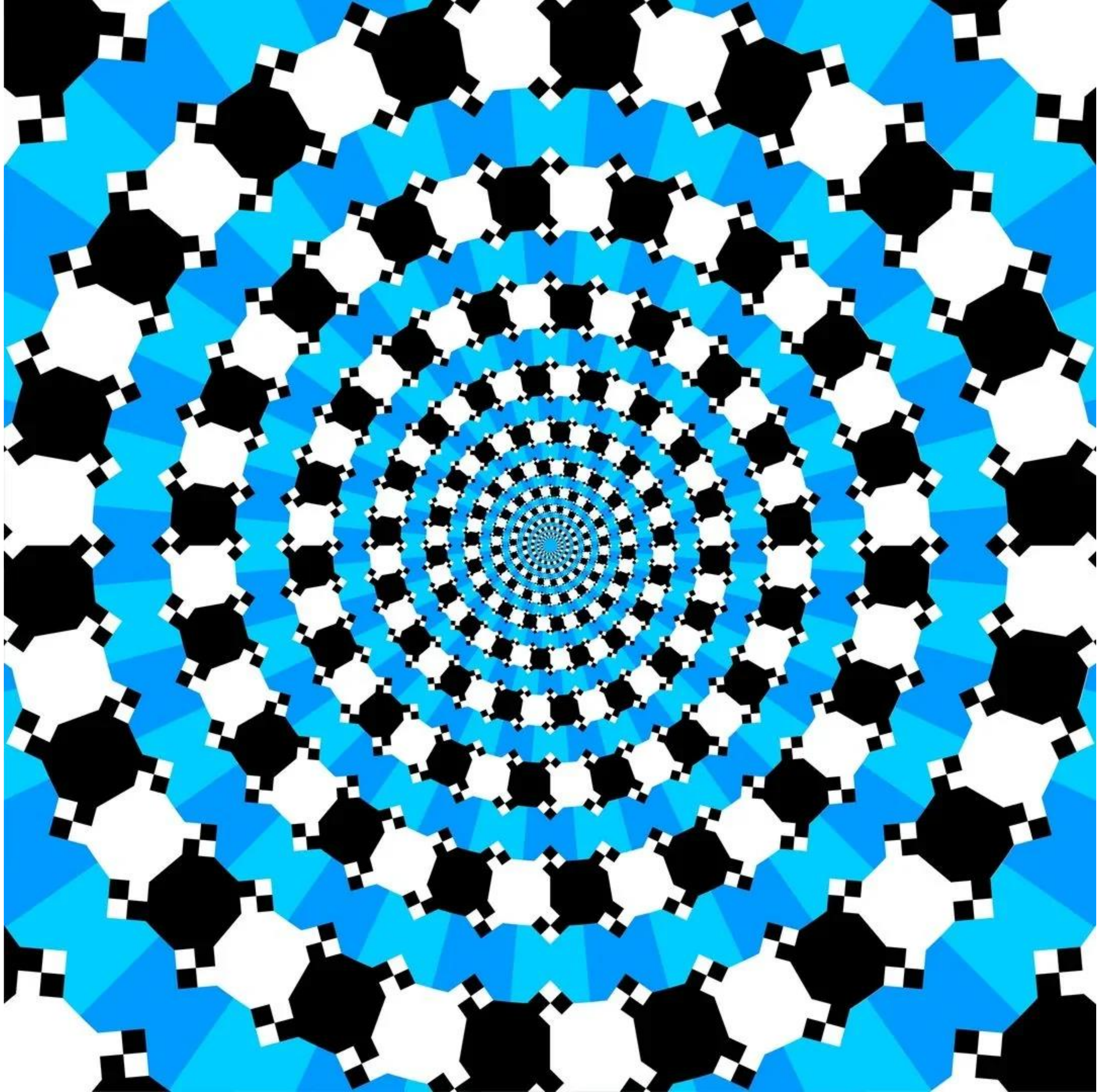
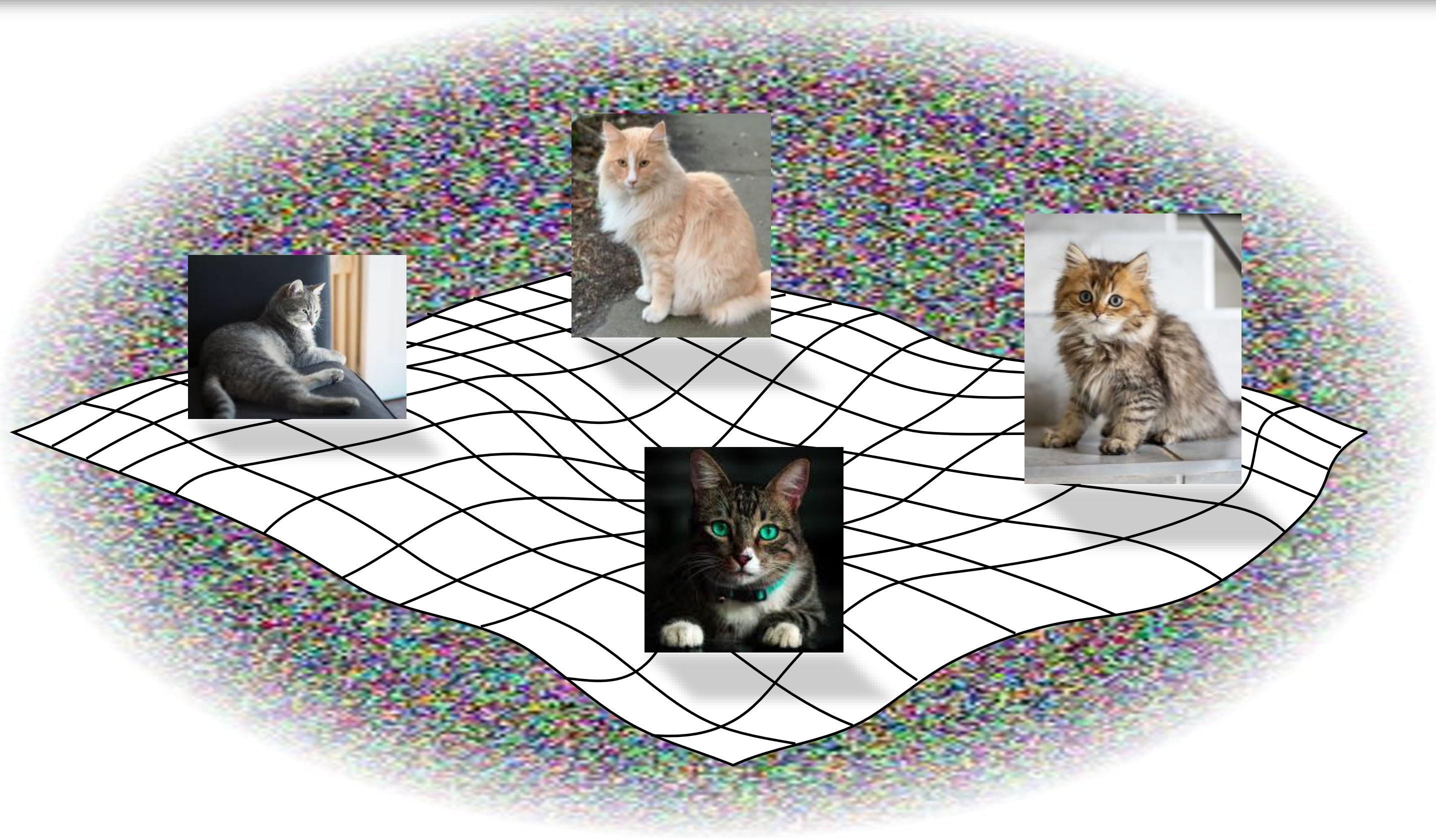


Image Manifolds & Image Generation



Slides from Noah Snavely, Abe Davis, Jin Sun, and Phillip Isola

Agenda

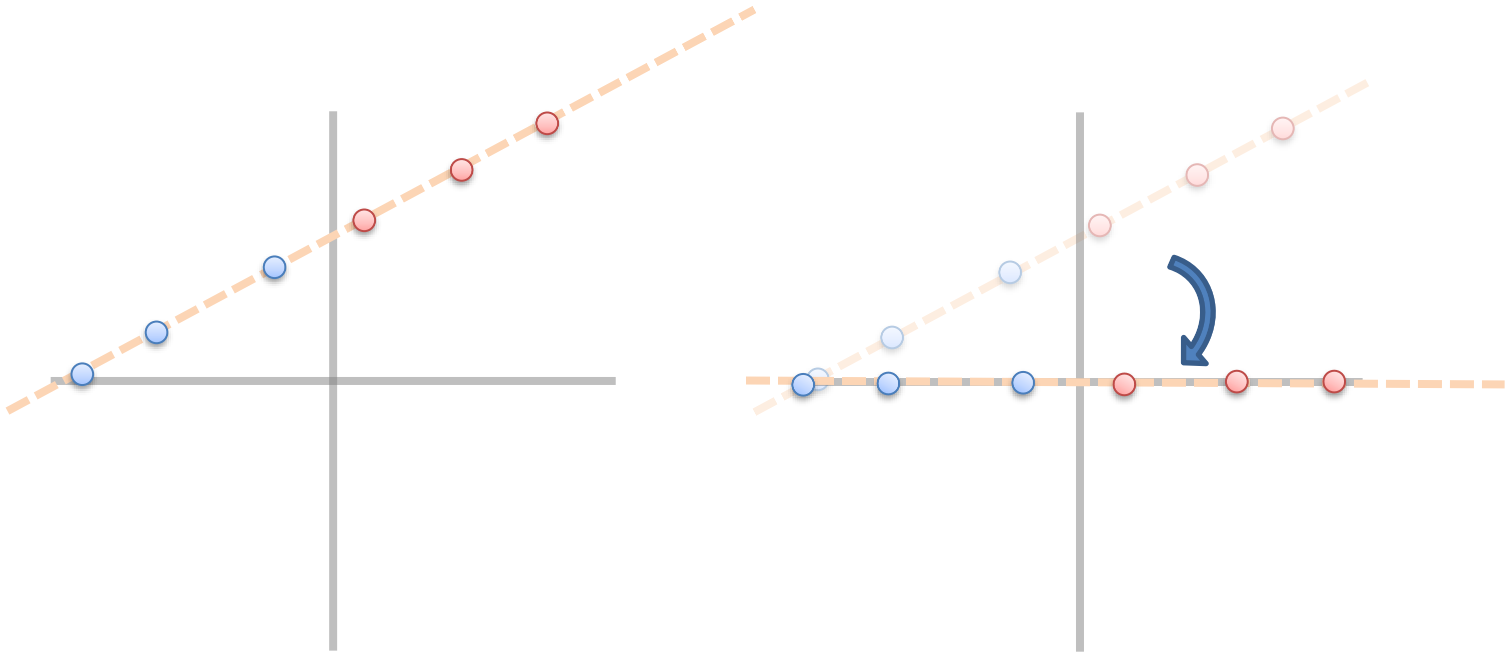
- The manifold of natural images
- Image-to-image methods and GANs
- Image synthesis methods
- diffusion models

By Abe Davis

DIMENSIONALITY REDUCTION

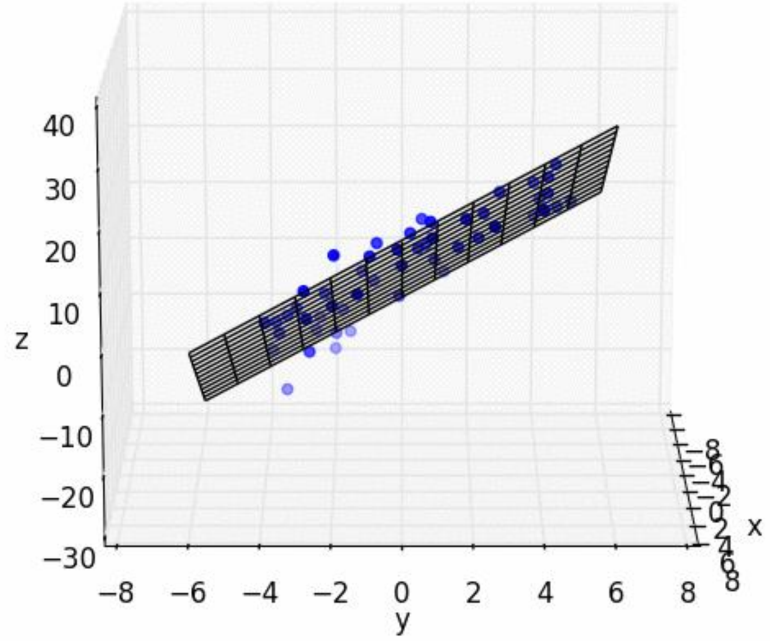
Linear Dimensionality Reduction: 2D->1D

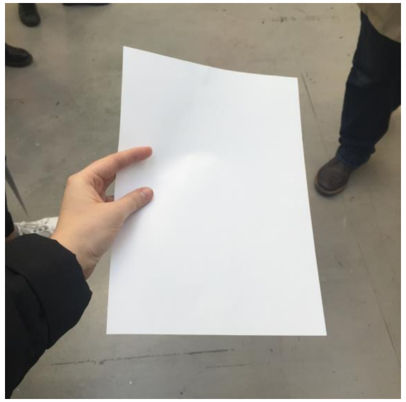
- Consider a bunch of data points in 2D
- Let’s say these points lie along a line
- If so, we can translate and rotate our data so that it is 1D



Linear Dimensionality Reduction: 3D->2D

- Similar to 1D case, we can fit a plane to the data, and transform our coordinate system so that plane becomes the x-y plane
- “Plane fitting”
- Now we only need to store two numbers for each point (and the plane parameters)
- More generally: look for the 2D subspace that best fits the data, and ignore the remaining dimensions

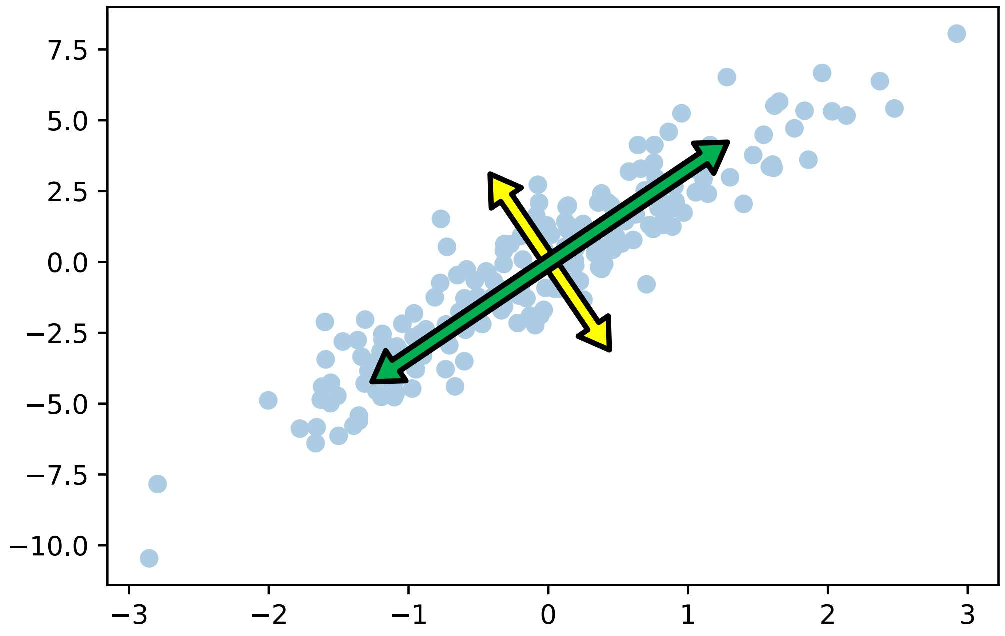




Think of this as data that sits on a flat sheet of paper, suspended in 3D space. We will come back to this analogy in a couple slides…

Generalizing Linear Dimensionality Reduction

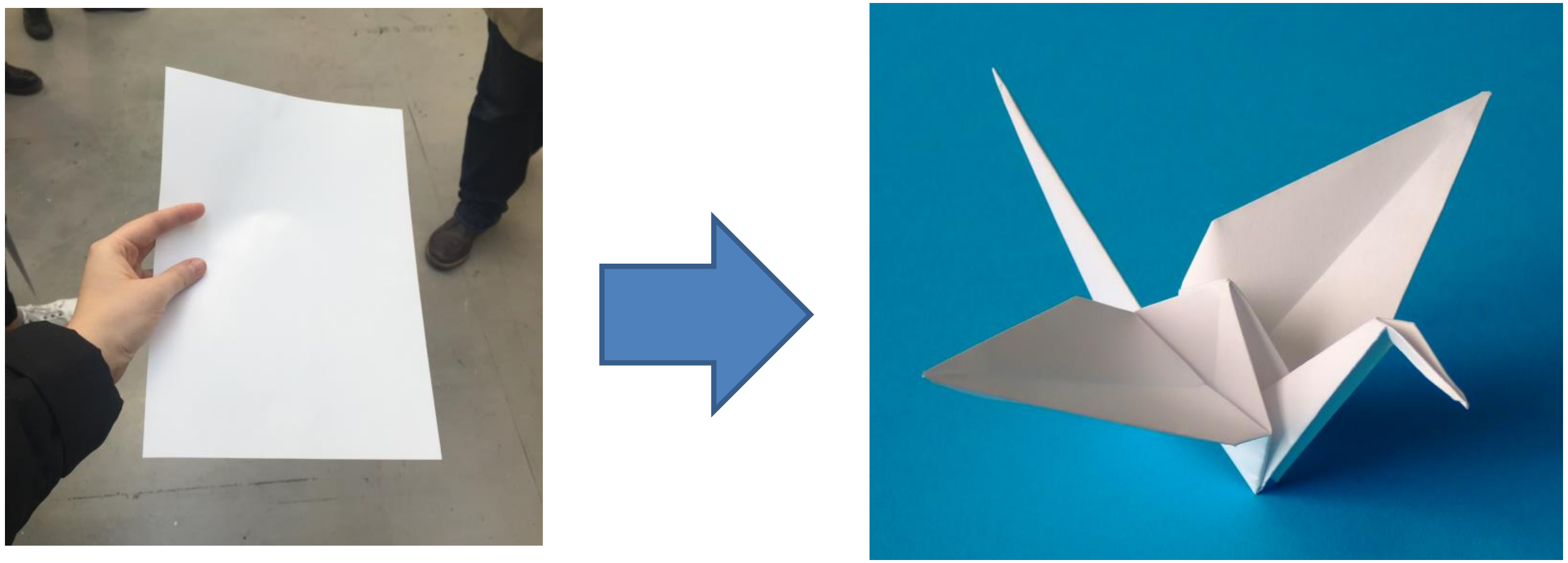
- ***Principal Components Analysis (PCA)***: find and order orthogonal axes by how much the data varies along each axis.
- The axes we find (ordered by variance of our data) are called ***principal components***.
- Dimensionality reduction can be done by using only the first *k* principal components



Side Note: principal components are closely related to the eigenvectors of the covariance matrix for our data

Manifolds

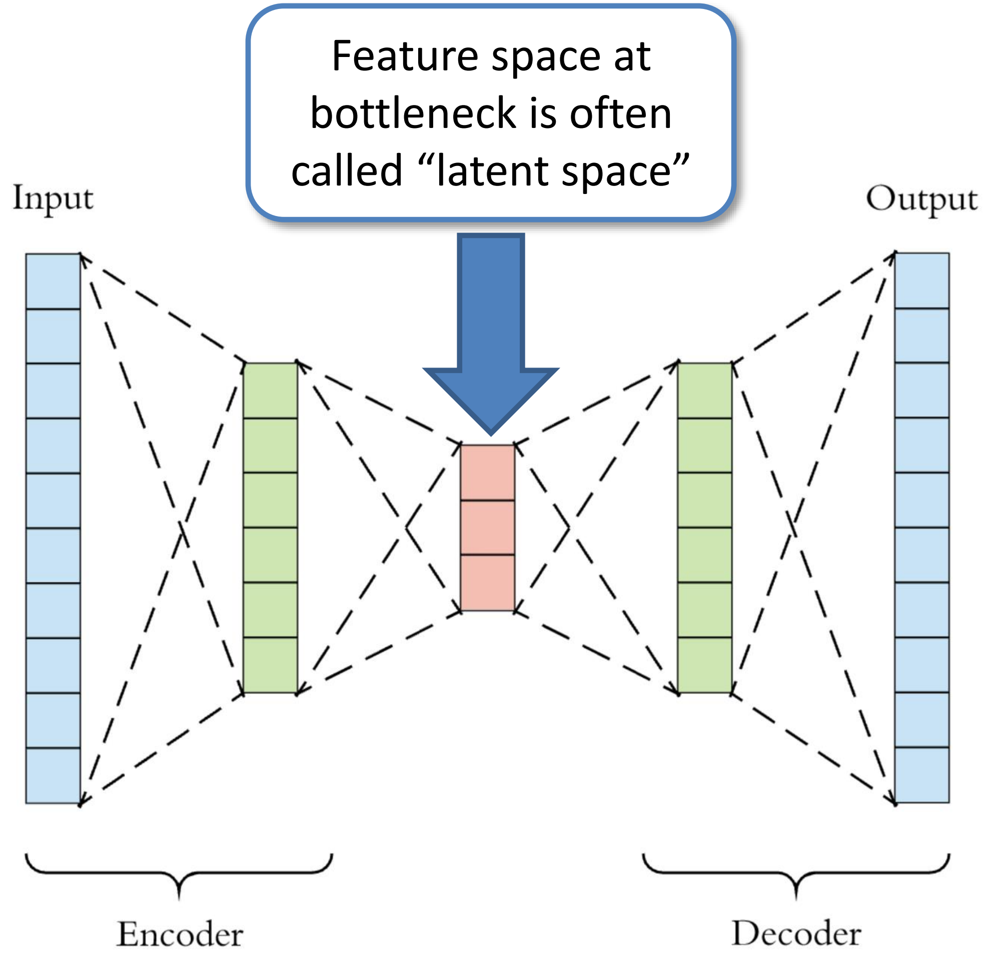
- Think of a piece of paper as a 2D subspace
- If we bend & fold it, it's still locally a 2D subspace…
- A "manifold" is the generalization of this concept to higher dimensions…



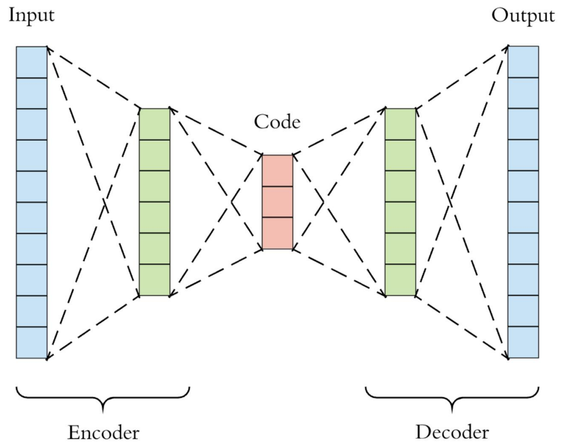
Autoencoders: Dimensionality Reduction for Manifolds

- Learn a non-linear (deep network) transformation into some lower-dimensional space (encoder)
- Learn a transformation from lower-dimensional space back to original content (decoder)
- Loss function measures difference between input & output

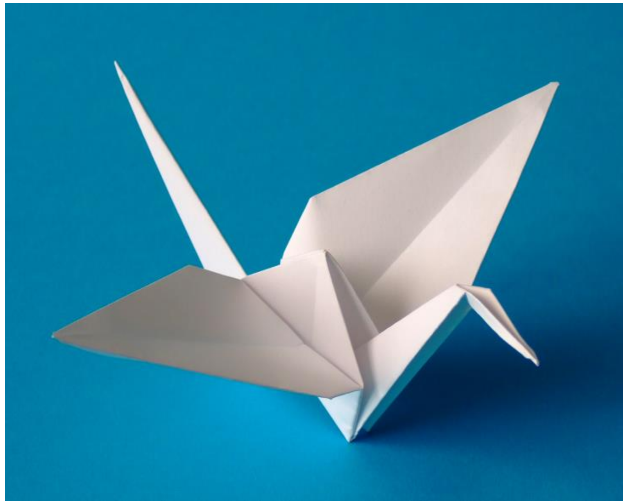
- **Unsupervised**
  - No labels required! Signal is just from learning to compress data



Autoencoders: Dimensionality Reduction for Manifolds



- Transformations that reduce dimensionality **cannot be invertible** in general



- An autoencoder tries to learn a transformation that is **invertible for points on some manifold**

By Abe Davis

IMAGE MANIFOLDS

The Space of All Images

- Lets consider the space of all 100x100 images

- Now lets randomly sample that space…

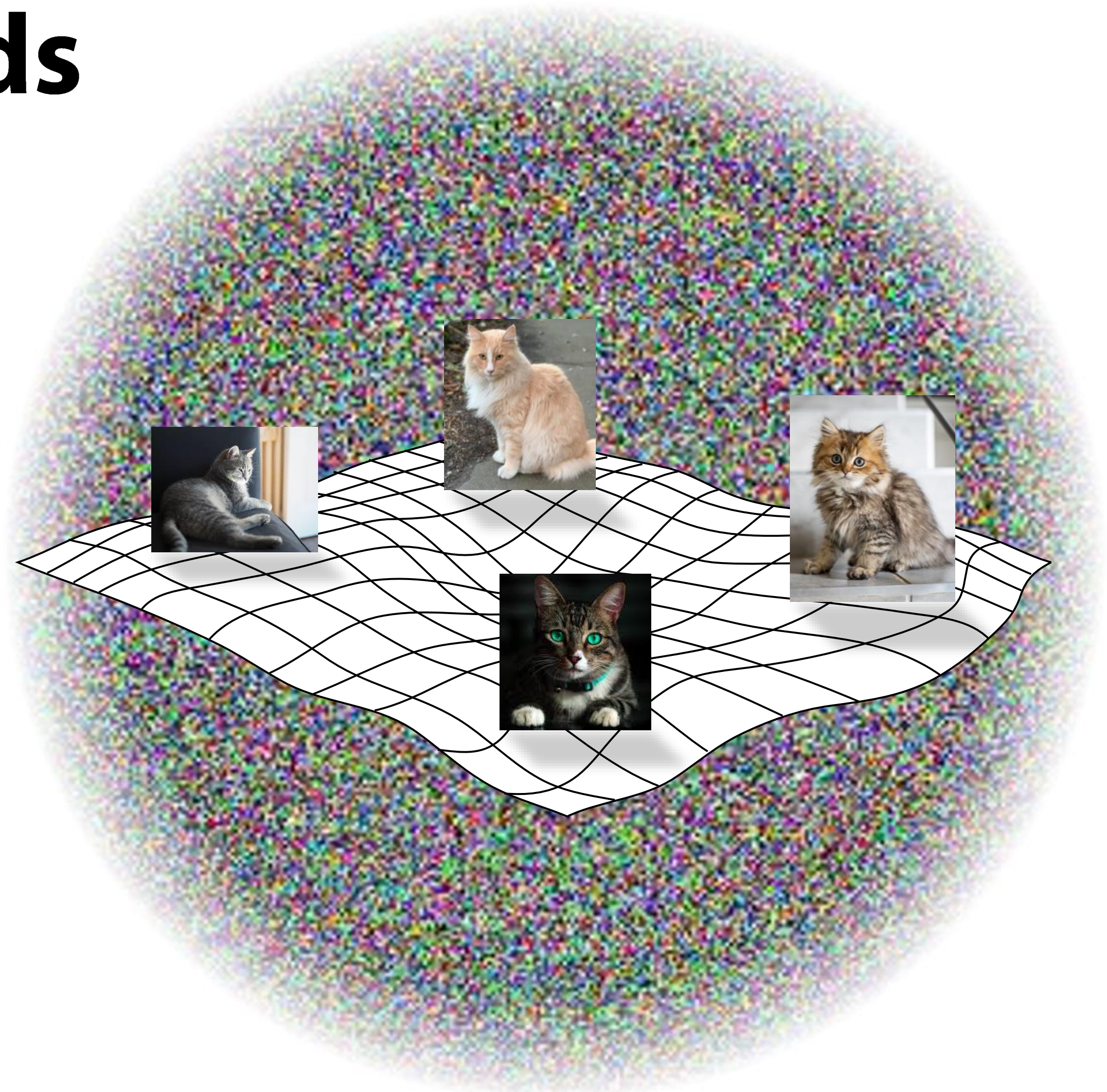
- Conclusion: Most images are noise

Question:
What do we expect a random uniform sample of all images to look like?

```
pixels = np.random.rand(100,100,3)
```

Natural Image Manifolds

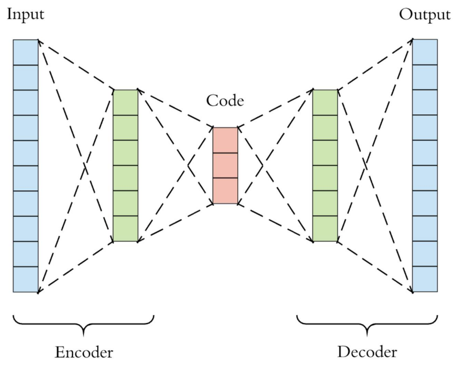
- Most images are "noise"
- "Meaningful" images tend to form some manifold within the space of all images
- Images of a particular class fall on manifolds within that manifold…

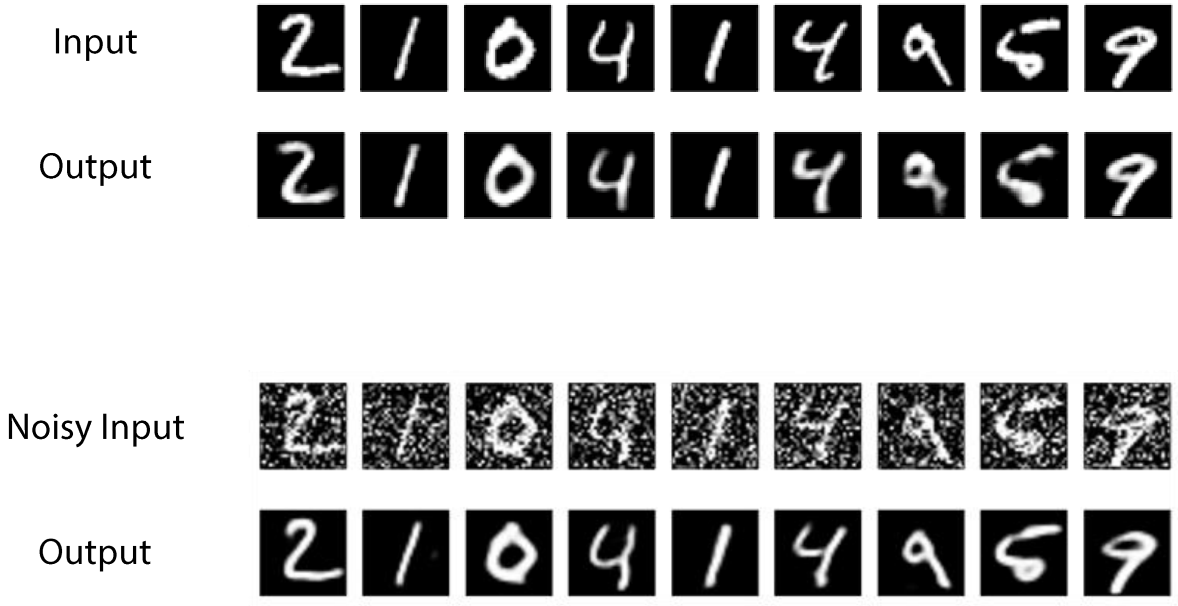


The Space of All Images

Denoising & the "Nullspace" of Autoencoders

- The autoencoder tries to learn a dimensionality reduction that is invertible for our data (data on some manifold)
- Most noise will be in the non-invertible part of image space (off the manifold)
- If we feed noisy data in, we will often get denoised data out

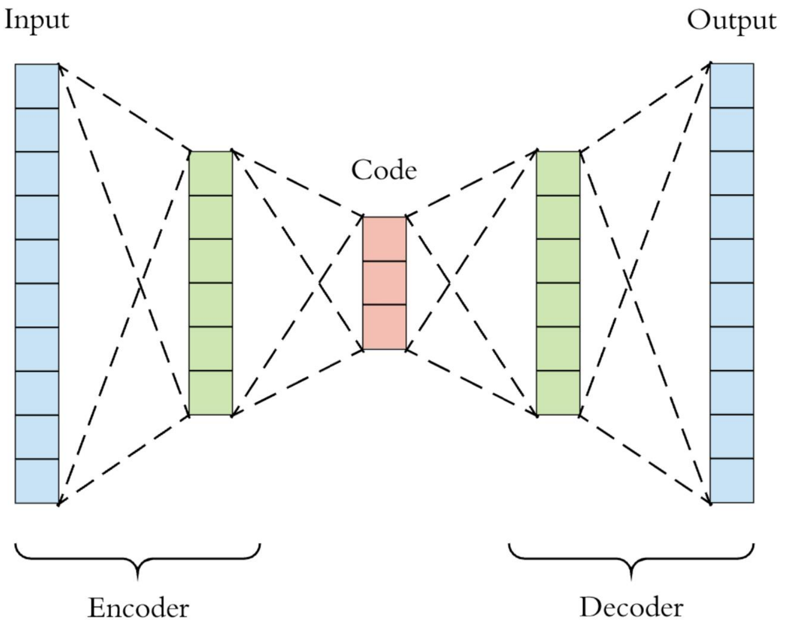


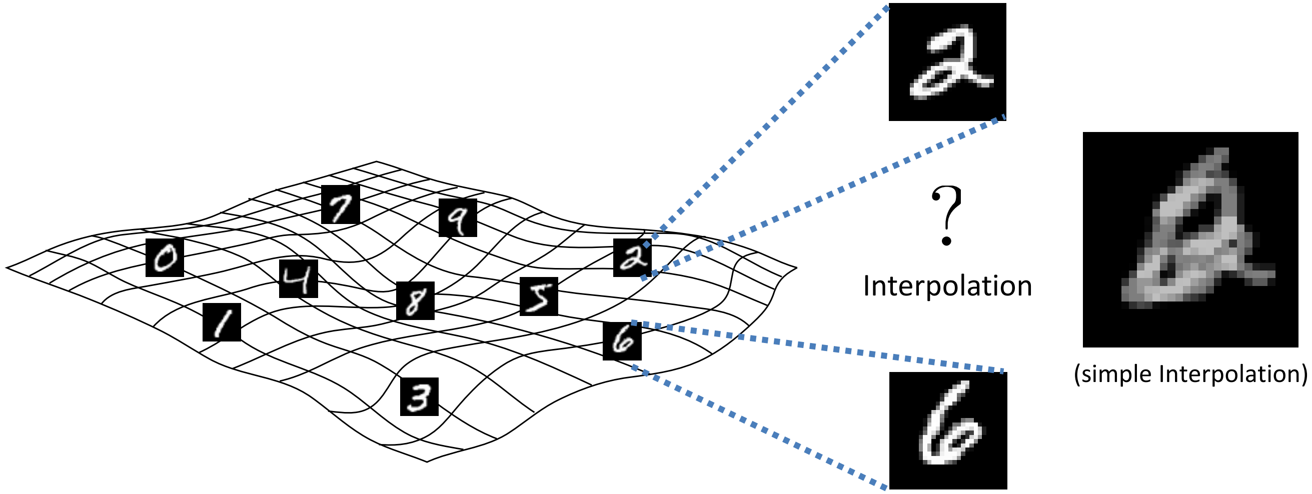


Examples from: https://blog.keras.io/building-autoencoders-in-keras.html

Problem

- Autoencoders can compress because data sits on a manifold
- This doesn't mean that every point in the latent space will be on the manifold…
- GANs (later this lecture) will learn a loss function that helps with this…



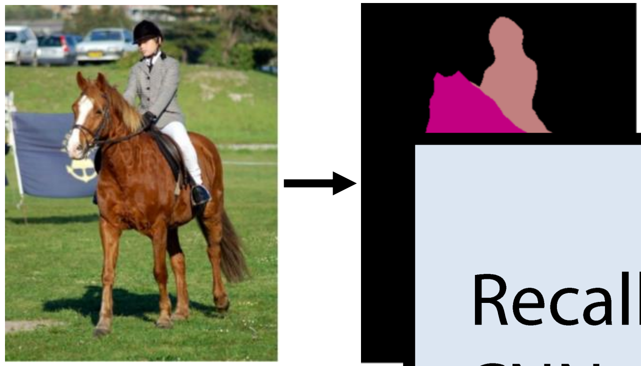


Abe Davis, with slides from Jin Sun, Phillip Isola, and Richard Zhang

IMAGE-TO-IMAGE APPLICATIONS

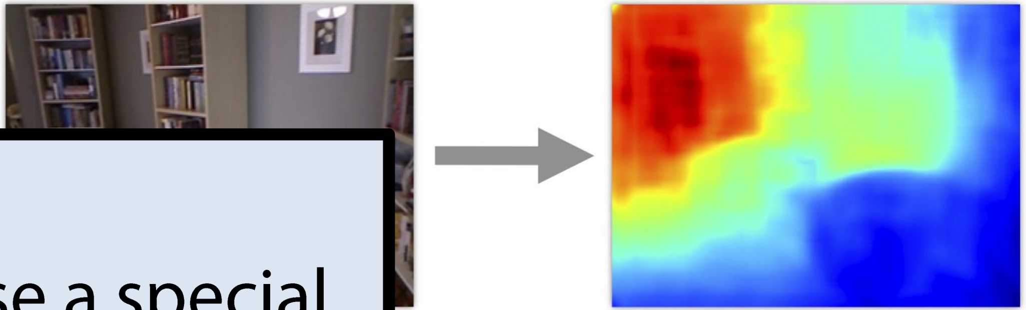
Image prediction (“structured prediction”)

Object labeling



[Long et al. 201

Depth prediction

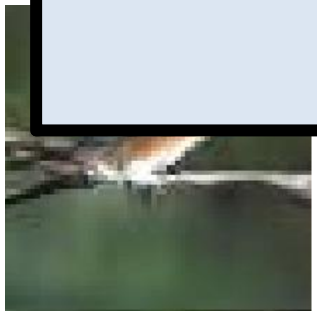


Eigen et al. 2014, …]

Recall: we often use a special CNN architecture like a U-Net for such image-to-image mappings

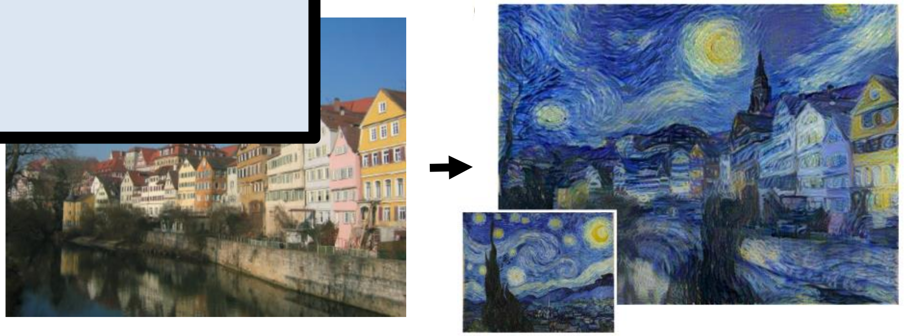
Text-to-ph

“this small bird has a pink breast and crown…”



[Reed et al. 2016, …]

yle transfer



[Gatys et al. 2016, …]

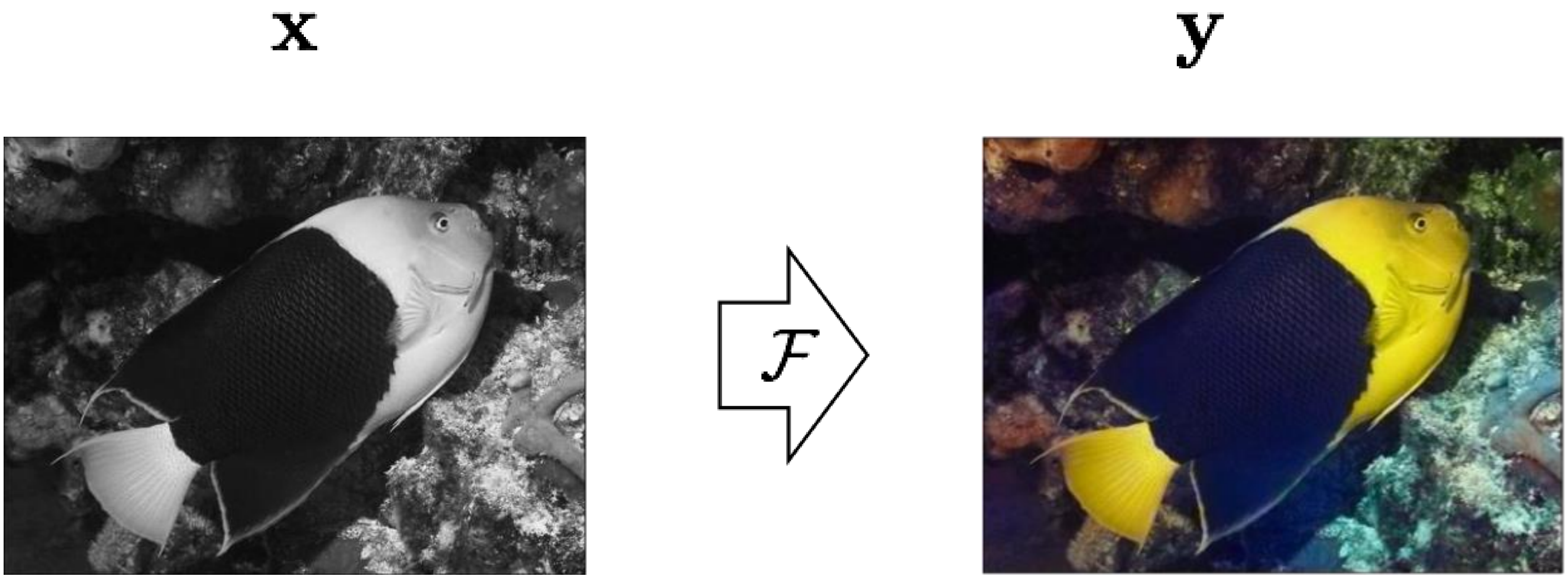


Image Colorization

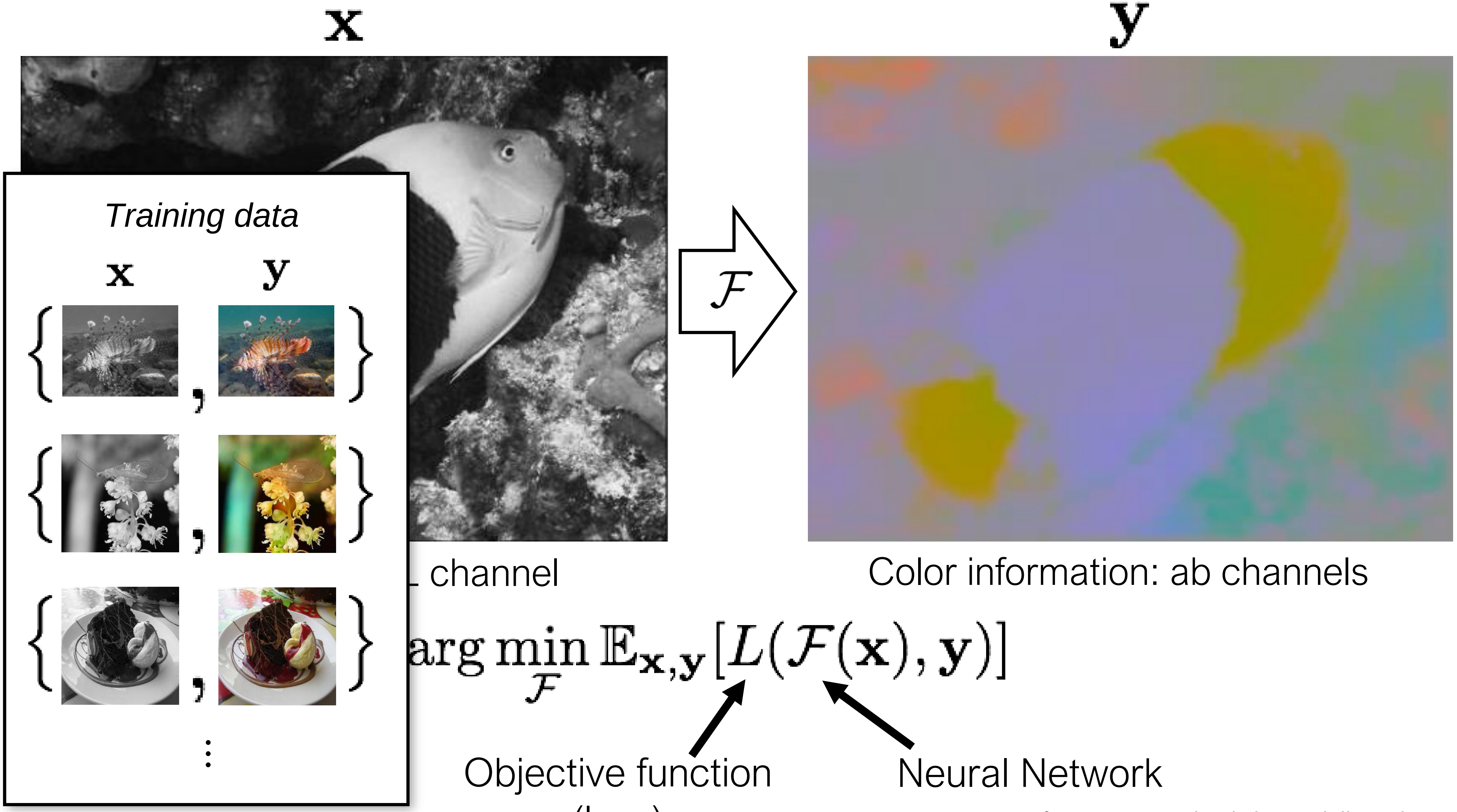
from Jin Sun, Richard Zhang, Phillip Isola

$\mathbf{x}$ $\mathbf{y}$

$$\arg\min_{\mathcal{F}} \mathbb{E}_{\mathbf{x},\mathbf{y}}[L(\mathcal{F}(\mathbf{x}), \mathbf{y})]$$

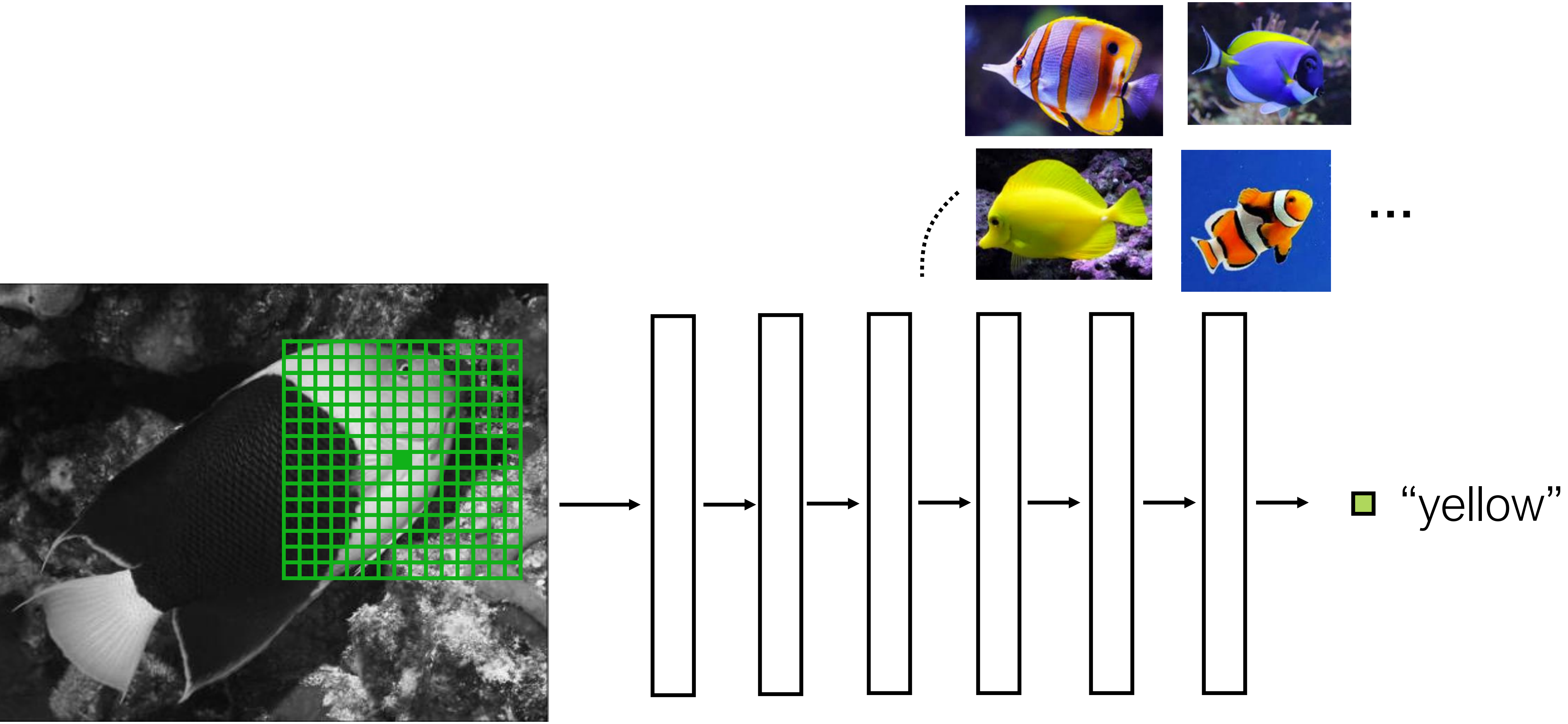
“**What** should I do” “**How** should I do it?”

from Jin Sun, Richard Zhang, Phillip Isola

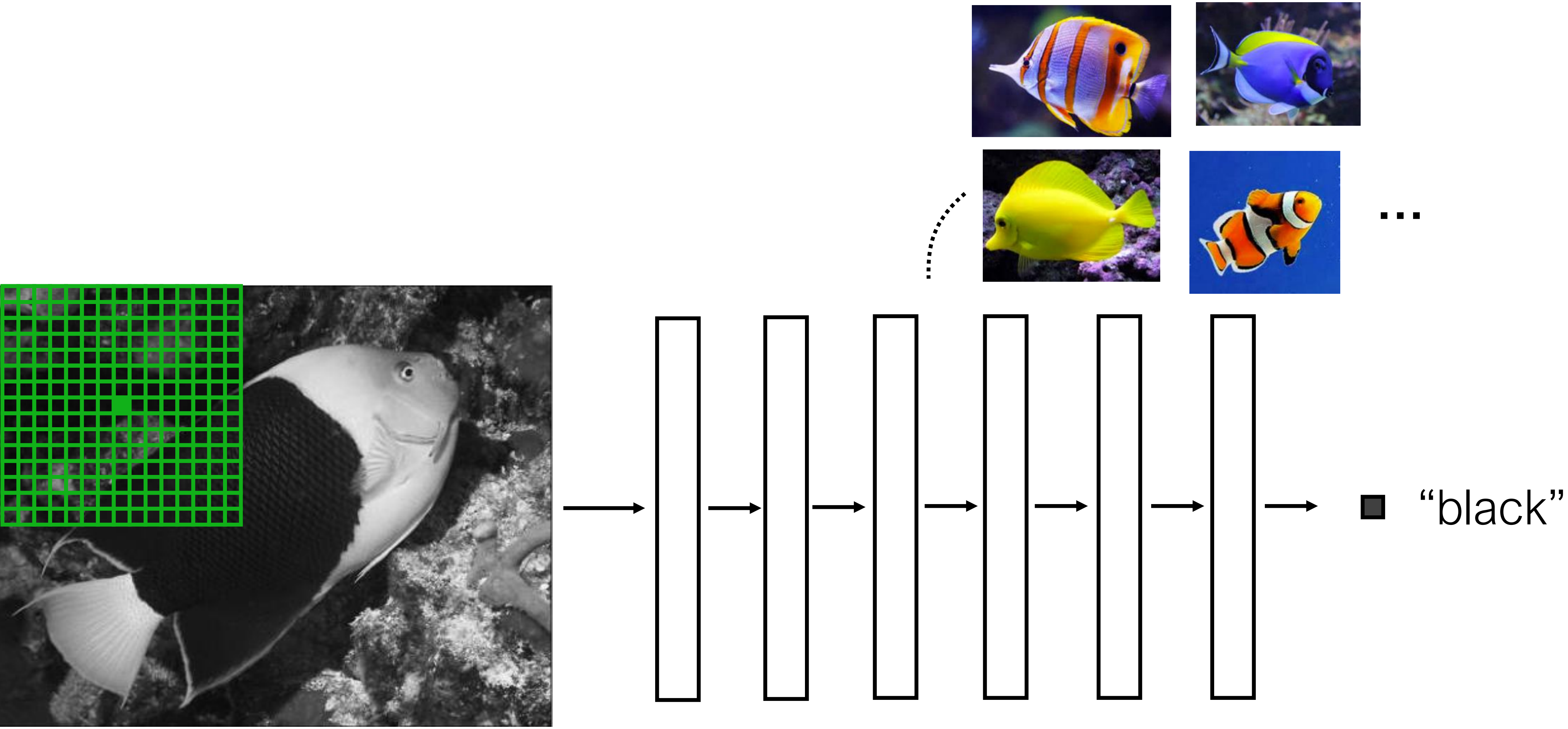


$$\arg\min_{\mathcal{F}} \mathbb{E}_{\mathbf{x},\mathbf{y}}[L(\mathcal{F}(\mathbf{x}), \mathbf{y})]$$

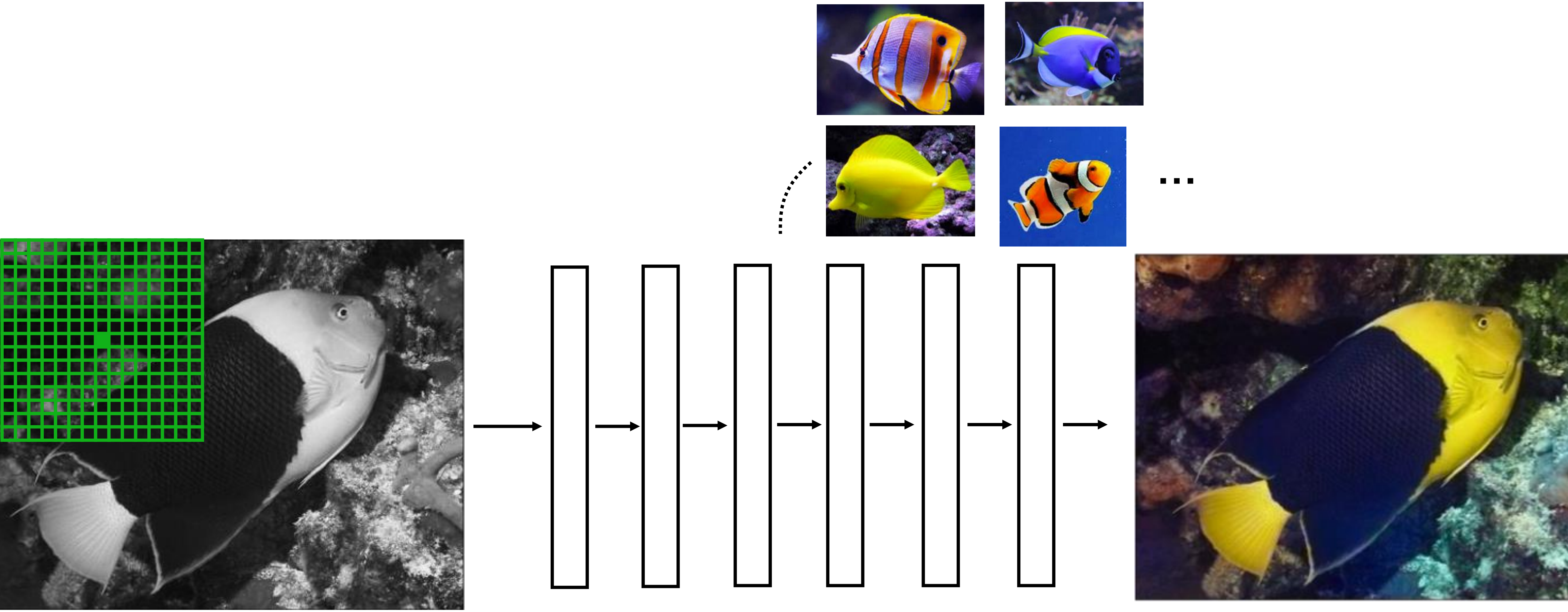
from Jin Sun, Richard Zhang, Phillip Isola



from Jin Sun, Richard Zhang, Phillip Isola



from Jin Sun, Richard Zhang, Phillip Isola



from Jin Sun, Richard Zhang, Phillip Isola

Recap: basic loss functions

Prediction: $\hat{\mathbf{y}} = \mathcal{F}(\mathbf{x})$

Truth: $\mathbf{y}$

Classification (cross-entropy):

$$L(\hat{\mathbf{y}}, \mathbf{y}) = -\sum_i \hat{\mathbf{y}}_i \log \mathbf{y}_i$$

← How many extra bits it takes to correct the predictions

from Jin Sun, Richard Zhang, Phillip Isola

Recap: basic loss functions

Prediction: $\hat{\mathbf{y}} = \mathcal{F}(\mathbf{x})$

Truth: $\mathbf{y}$

Classification (cross-entropy):

$$L(\hat{\mathbf{y}}, \mathbf{y}) = -\sum_i \hat{\mathbf{y}}_i \log \mathbf{y}_i$$

← How many extra bits it takes to correct the predictions

Least-squares regression:

$$L(\hat{\mathbf{y}}, \mathbf{y}) = \|\hat{\mathbf{y}} - \mathbf{y}\|_2$$

← How far off we are in Euclidean distance

from Jin Sun, Richard Zhang, Phillip Isola

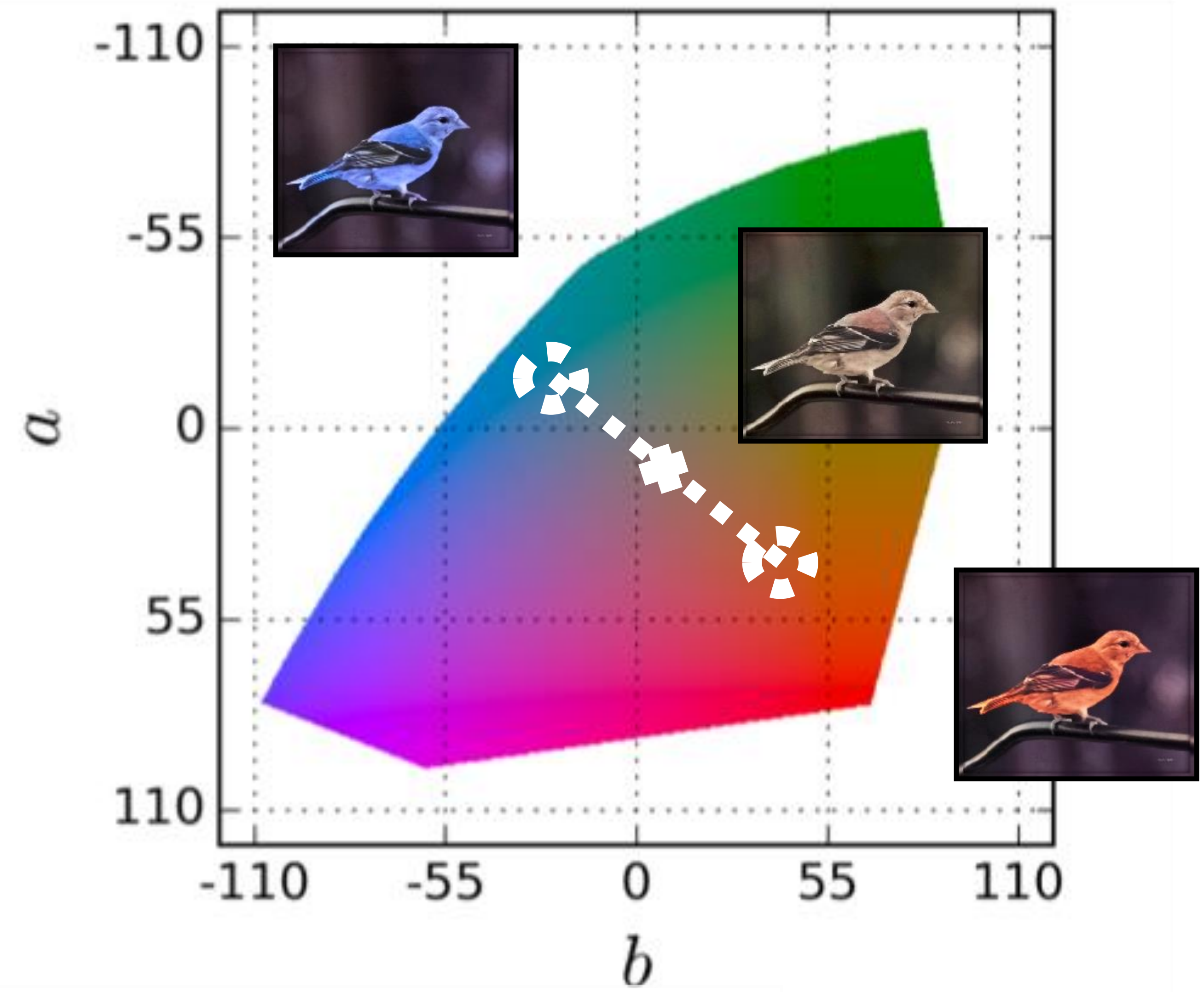
Designing loss functions

Input

Output (with L2 loss)

Ground truth

$$L_2(\hat{\mathbf{Y}}, \mathbf{Y}) = \frac{1}{2} \sum_{h,w} \|\mathbf{Y}_{h,w} - \hat{\mathbf{Y}}_{h,w}\|_2^2$$ (L2 loss)



With L2 loss, predictions “regress to the mean”, and lack vivid colors

$$\mathrm{L}_2(\widehat{\mathbf{Y}},\mathbf{Y}) = \frac{1}{2}\sum_{h,w}\|\mathbf{Y}_{h,w} - \widehat{\mathbf{Y}}_{h,w}\|_2^2$$

Designing loss functions

Input

Zhang et al. 2016

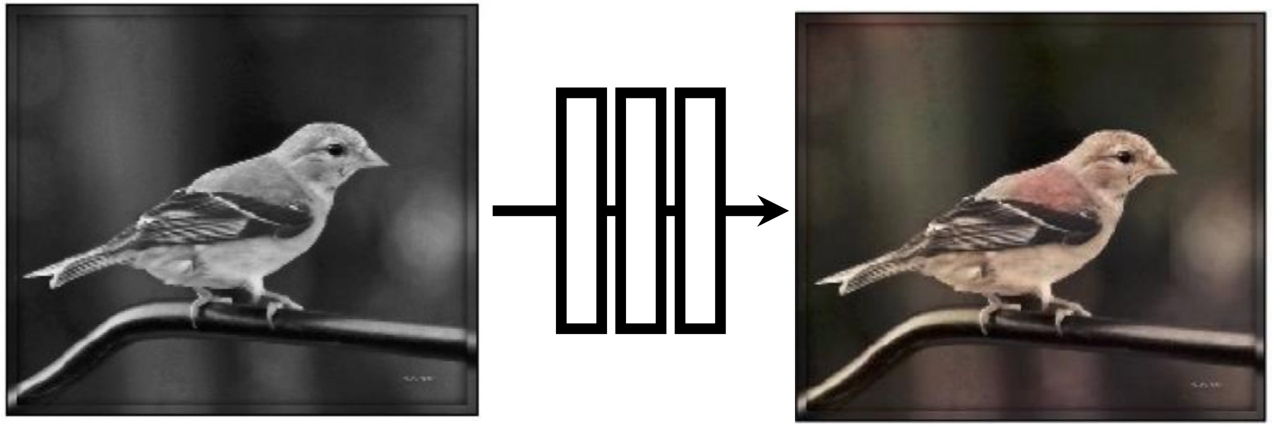
Ground truth

Color distribution cross-entropy loss with colorfulness enhancing term.

[Zhang, Isola, Efros, ECCV 2016]

Designing loss functions

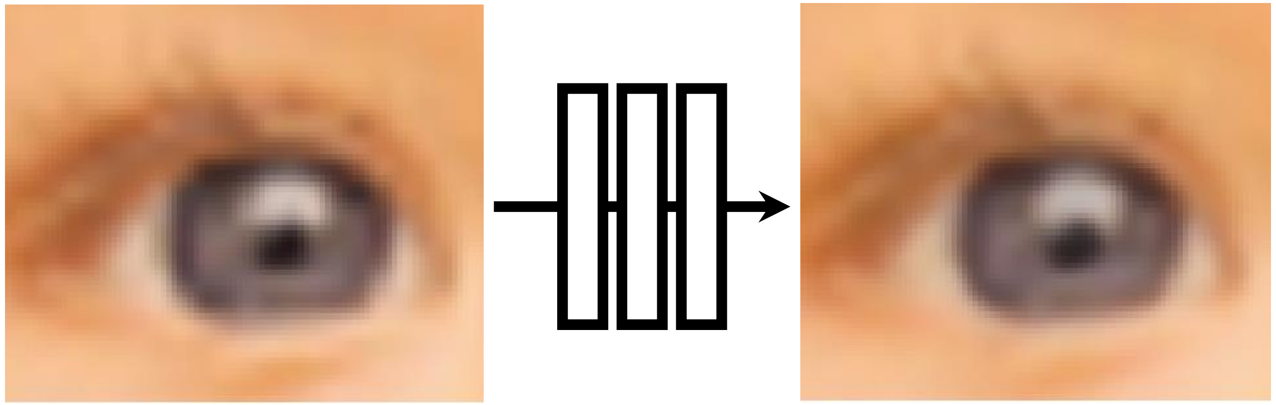
Image colorization



[Zhang, Isola, Efros, ECCV 2016]

L2 regression

Super-resolution

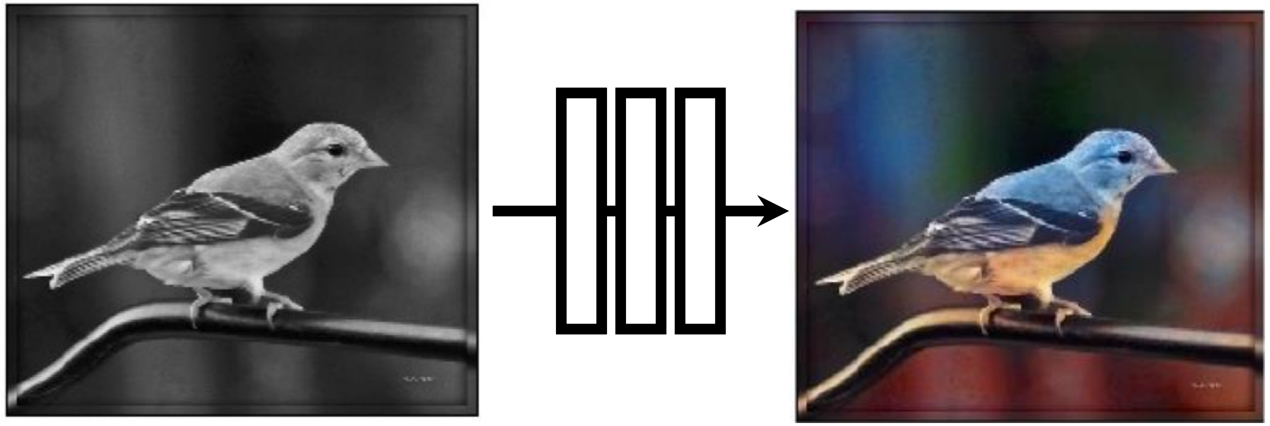


[Johnson, Alahi, Li, ECCV 2016]

L2 regression

Designing loss functions

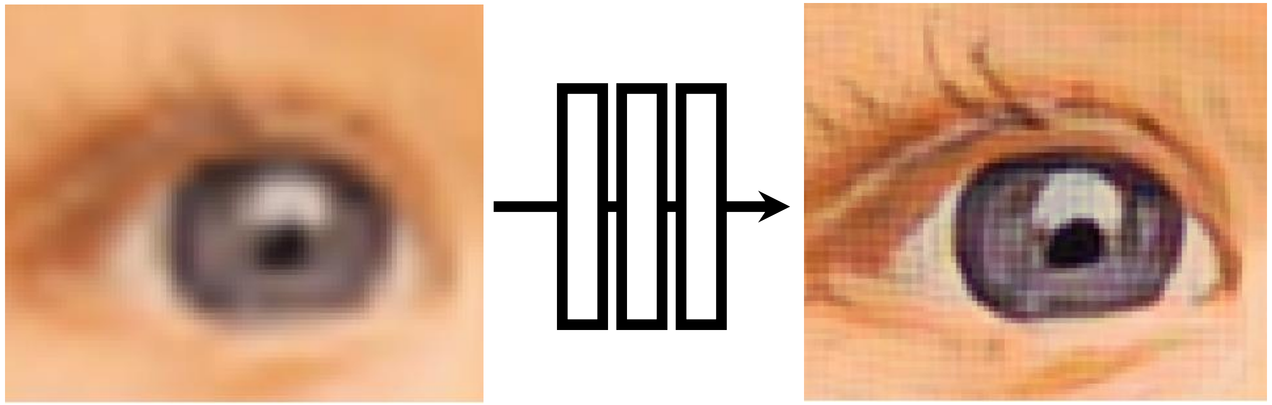
Image colorization



Cross entropy objective, with colorfulness term

[Zhang, Isola, Efros, ECCV 2016]

Super-resolution



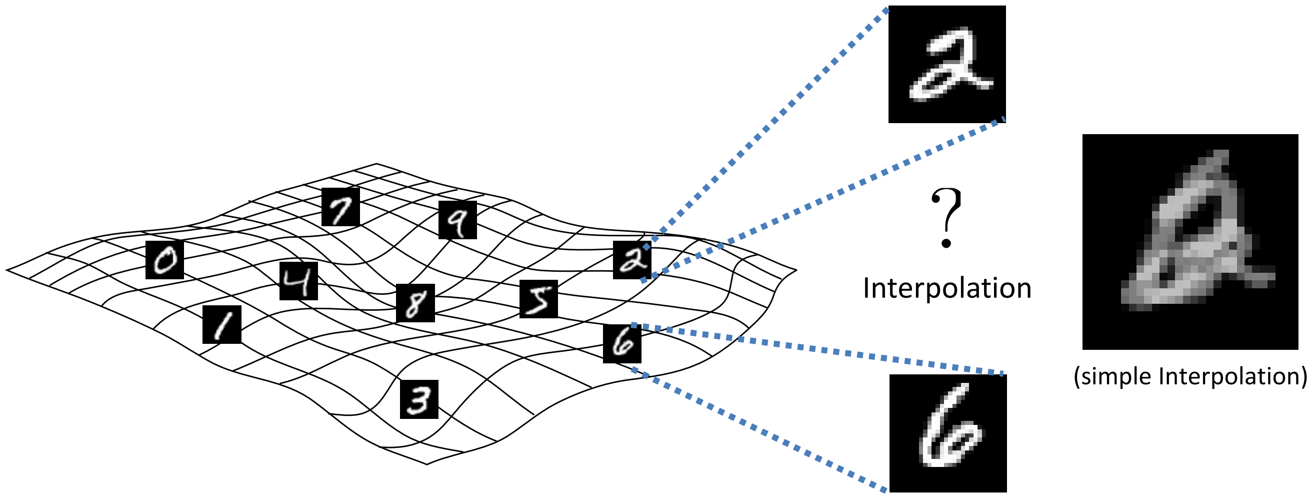
Deep feature covariance matching objective

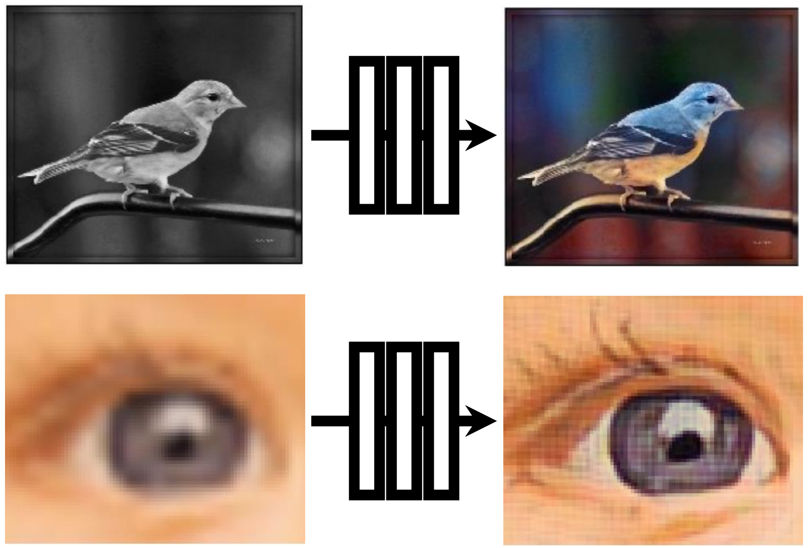
[Johnson, Alahi, Li, ECCV 2016]

Better Loss Function: Sticking to the Manifold

- How do we design a loss function that penalizes images that aren't on the image manifold?

- Key insight: we will *learn* our loss function by training a network to discriminate between images that are on the manifold and images that aren't



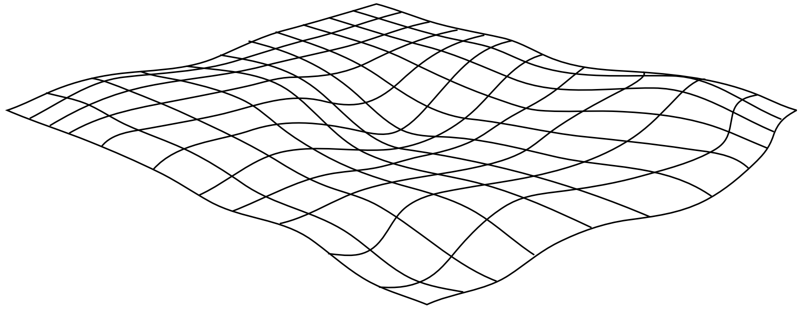


Abe Davis, with slides from Jin Sun and Phillip Isola

PART 3: GENERATIVE ADVERSARIAL NETWORKS (GANS)

Generative Adversarial Networks (GANs)

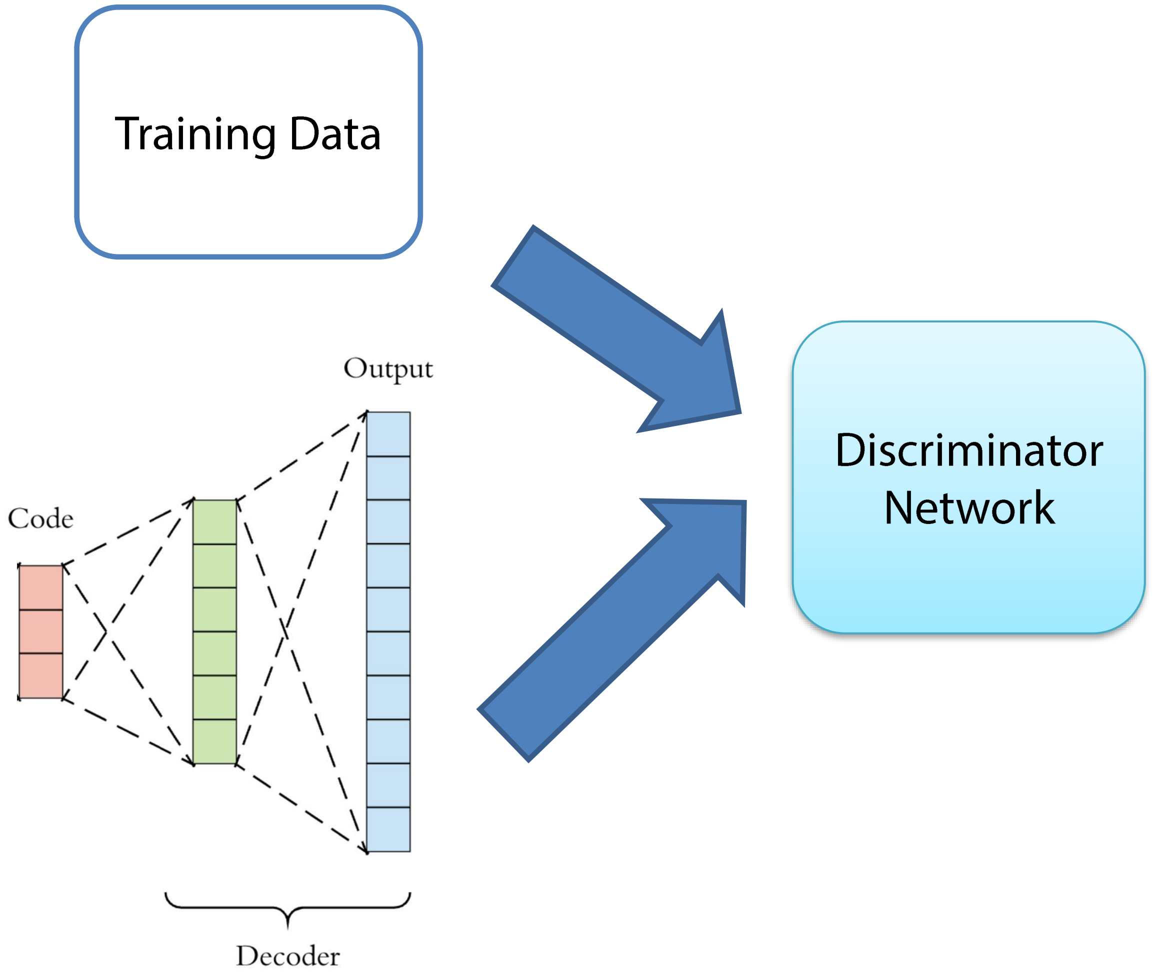
- Basic idea: Learn a mapping from some latent space to images on a particular manifold



- Example of a ***Generative Model:***
  - We can think of classification as a way to compute some P(x) that tells us the probability that image x is a member of a class.
  - Rather than simply evaluating this distribution, a generative model tries to learn a way to sample from it

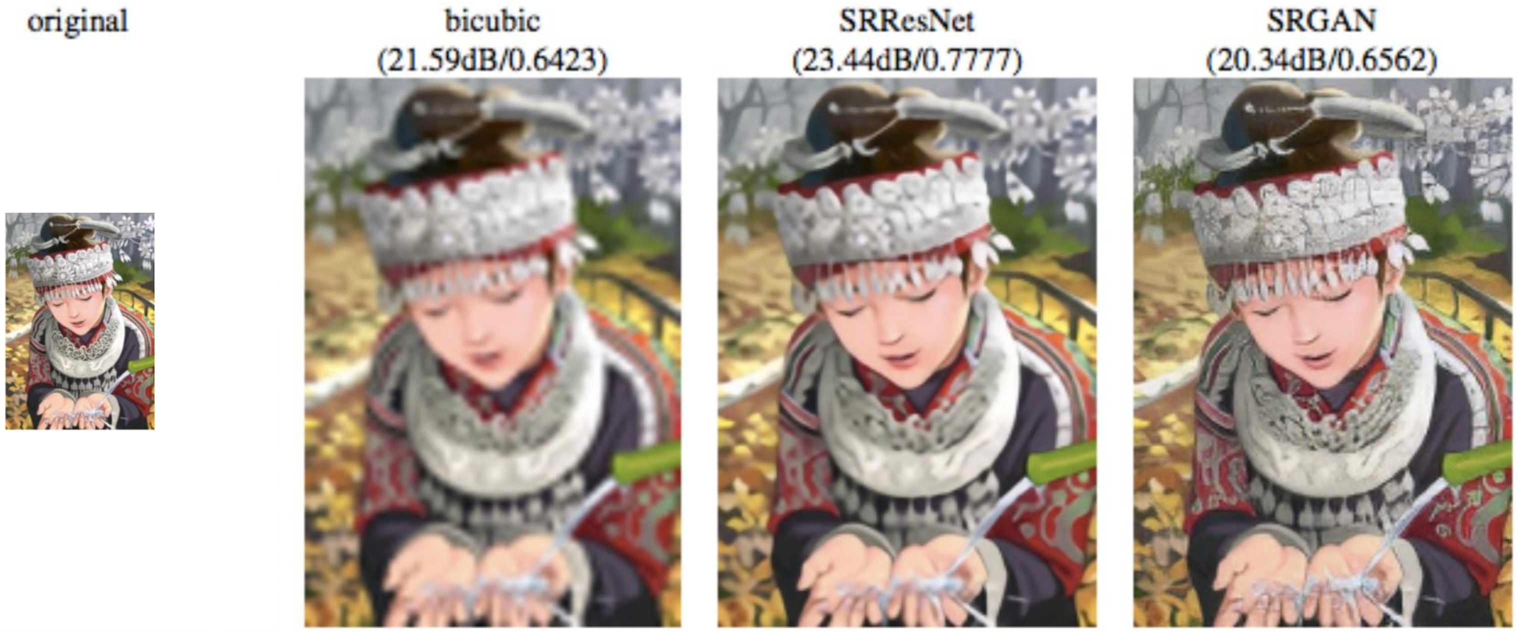
Generative Adversarial Networks (GANs)

- Generator network has similar structure to the decoder of our autoencoder
  - Maps from some latent space to images
- We train it in an adversarial manner against a discriminator network
  - Generator takes image noise, and tries to create output indistinguishable from training data
  - Discriminator tries to distinguish between generator output and training data

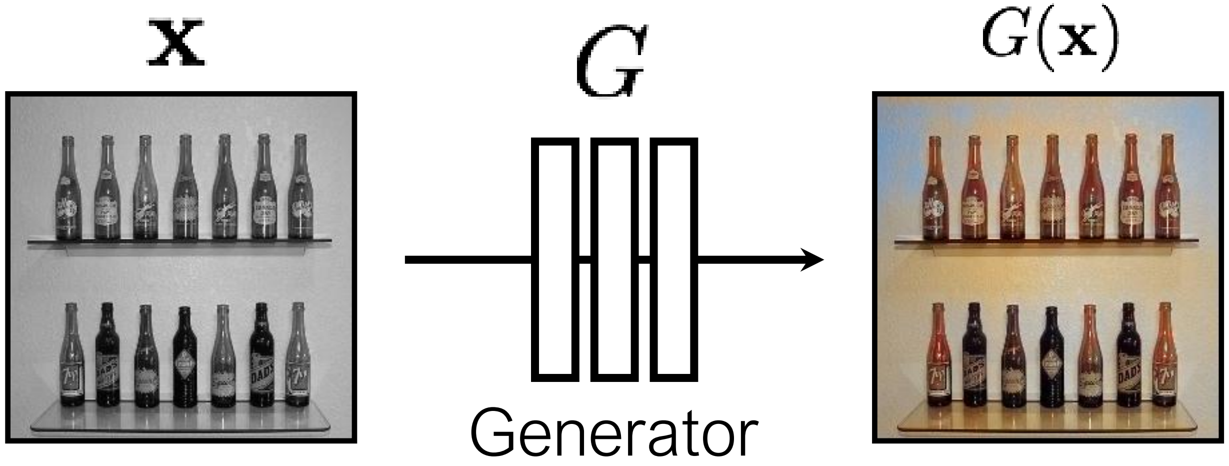


First: Conditional GANs

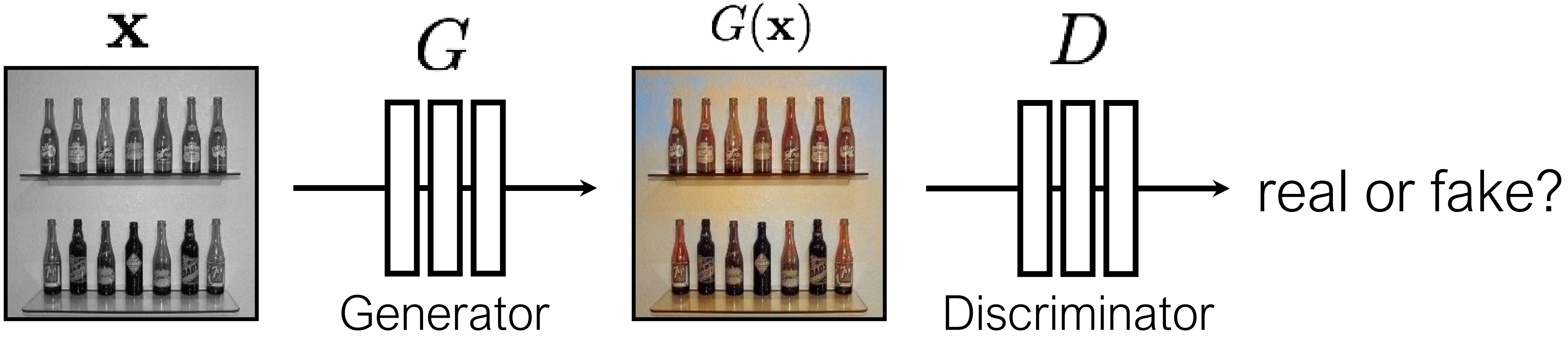
- Generate samples from a *conditional distribution* (conditioned on some other input)
- Example: generate high-resolution image conditioned on low resolution input



[Ledig et al 2016]



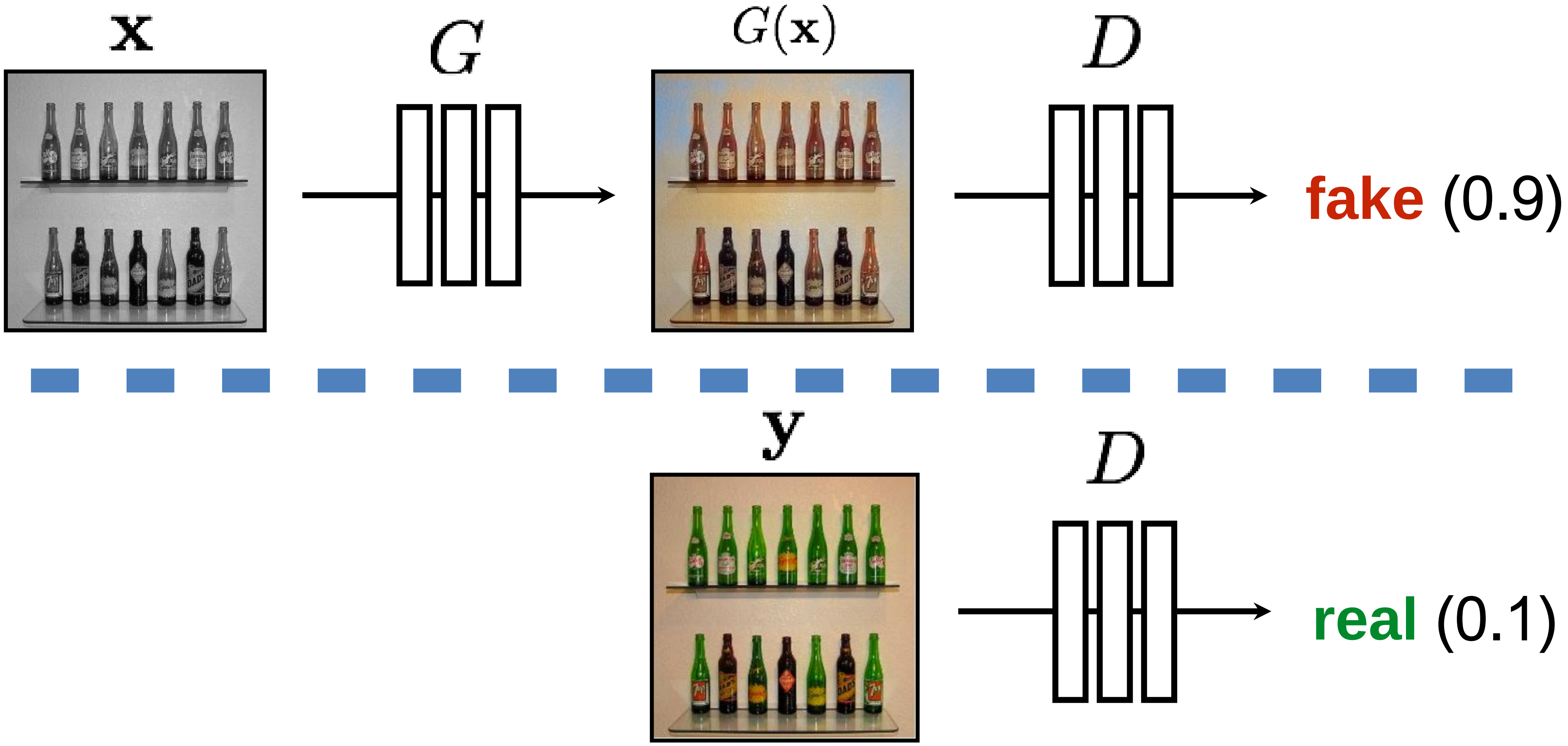
[Goodfellow et al., 2014]



G tries to synthesize fake images that fool **D**

D tries to identify the fakes

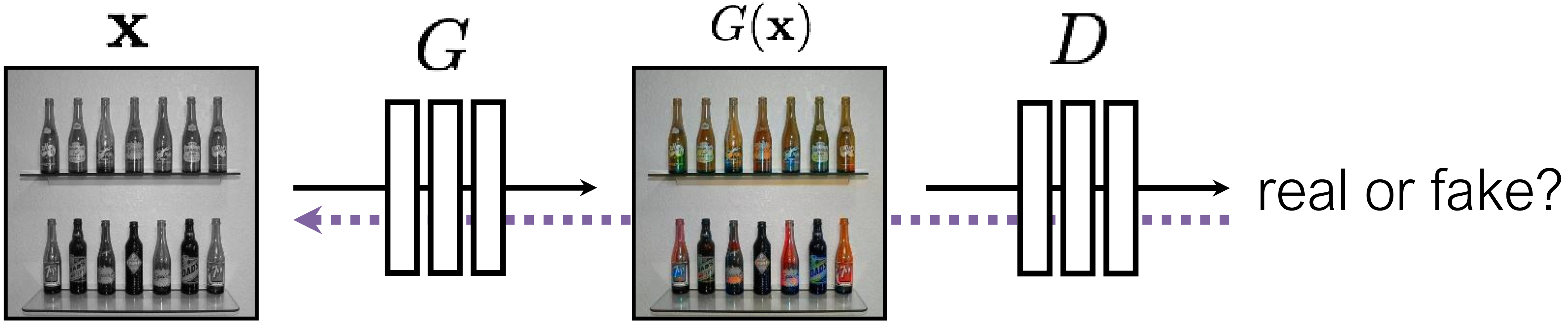
[Goodfellow et al., 2014]



(Identify generated images as fake) (Identify training images as real)

$$\arg\max_{D} \; \mathbb{E}_{\mathbf{x},\mathbf{y}}\left[\boxed{\log D(G(\mathbf{x}))} + \boxed{\log(1 - D(\mathbf{y}))} \right]$$

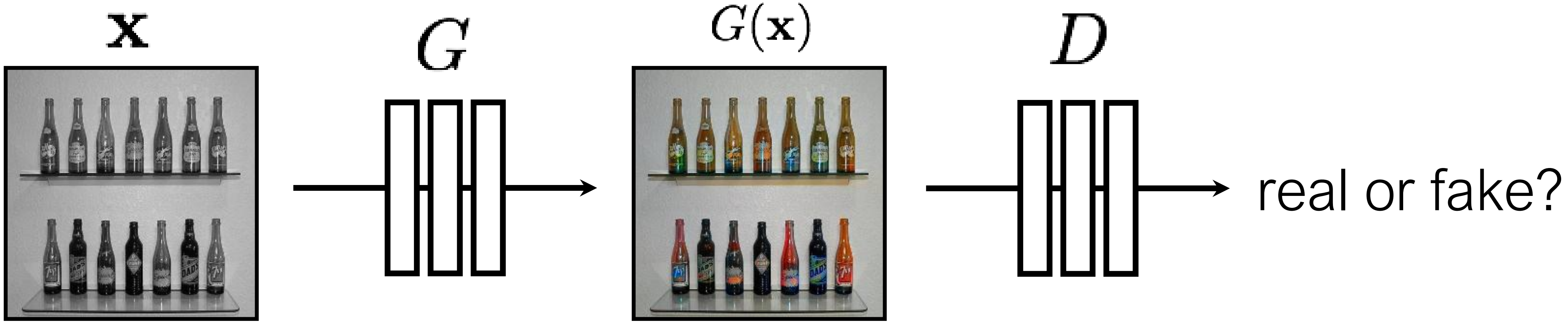
[Goodfellow et al., 2014]



G tries to synthesize fake images that ***fool*** **D**:

$$\arg\min_{G} \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ \log D(G(\mathbf{x})) + \log(1 - D(\mathbf{y}))\ ]$$

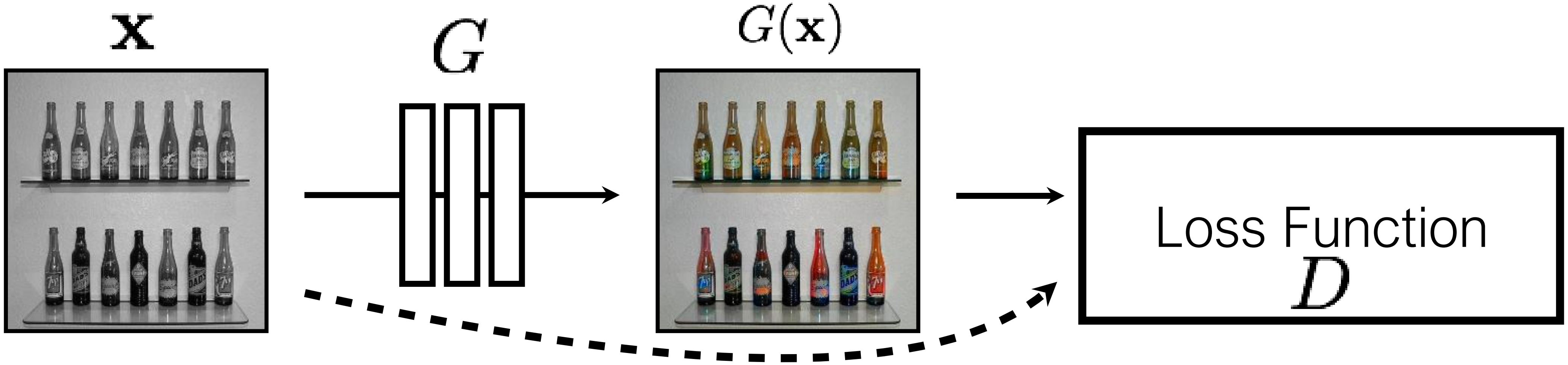
[Goodfellow et al., 2014]



G tries to synthesize fake images that ***fool*** the ***best*** **D**:

$$\arg\min_{G}\max_{D}\mathbb{E}_{\mathbf{x},\mathbf{y}}[\ \log D(G(\mathbf{x})) \ + \ \log(1 - D(\mathbf{y}))\ ]$$

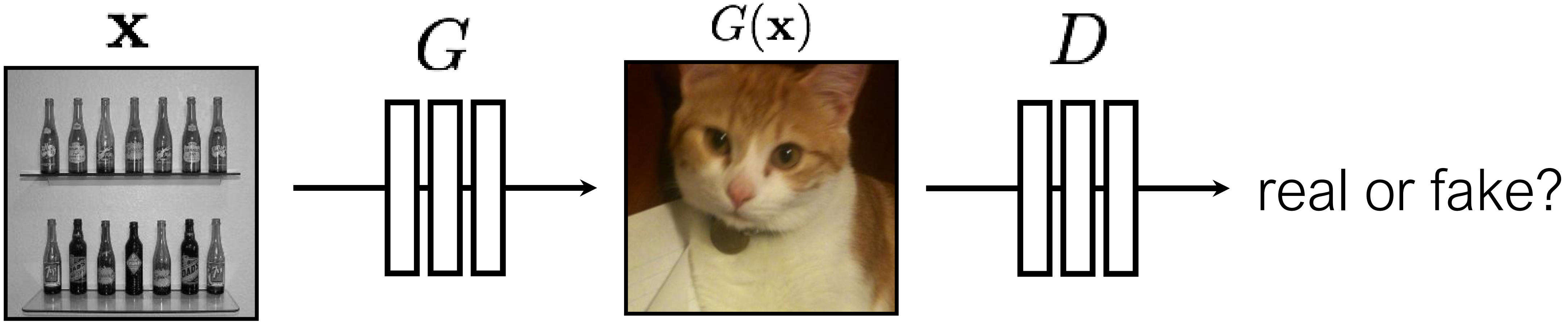
[Goodfellow et al., 2014]



G's perspective: **D** is a loss function.

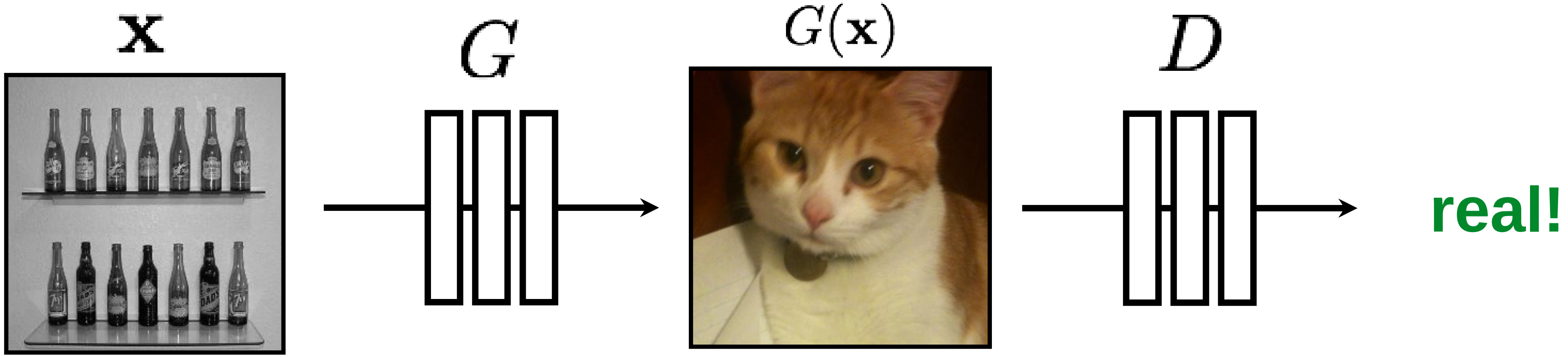
Rather than being hand-designed, it is *learned*.

[Goodfellow et al., 2014]
[Isola et al., 2017]



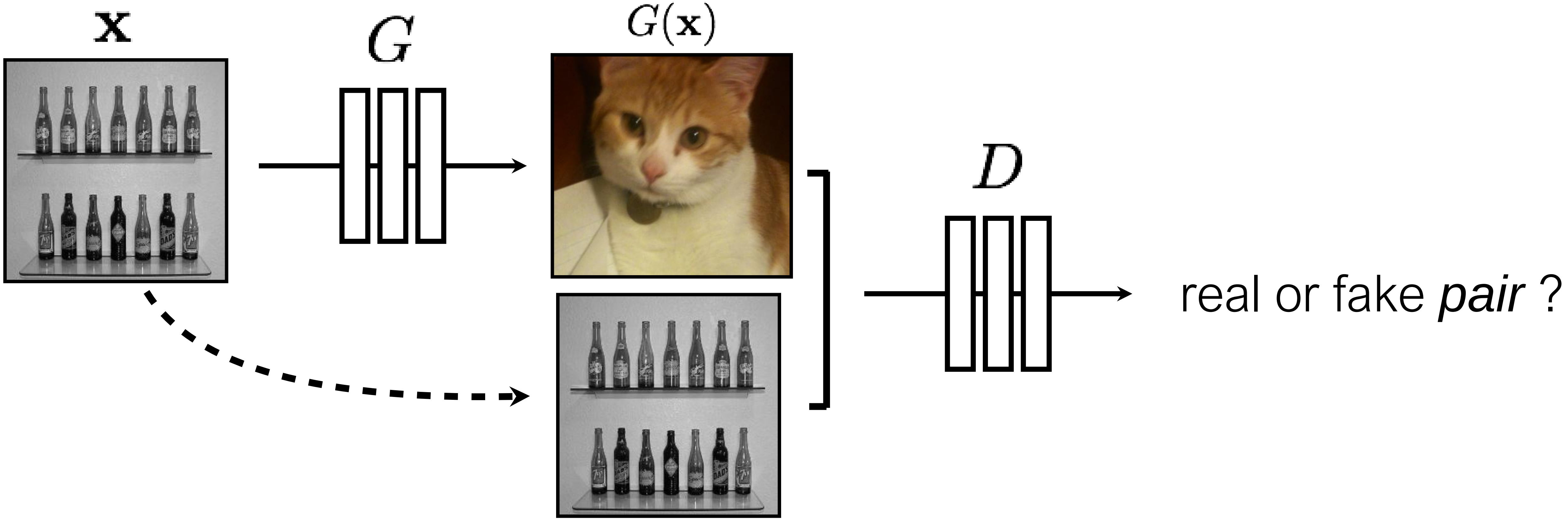
$$\arg\min_{G}\max_{D}\ \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ \log D(G(\mathbf{x})) \ + \ \log(1 - D(\mathbf{y}))\ ]$$

[Goodfellow et al., 2014]



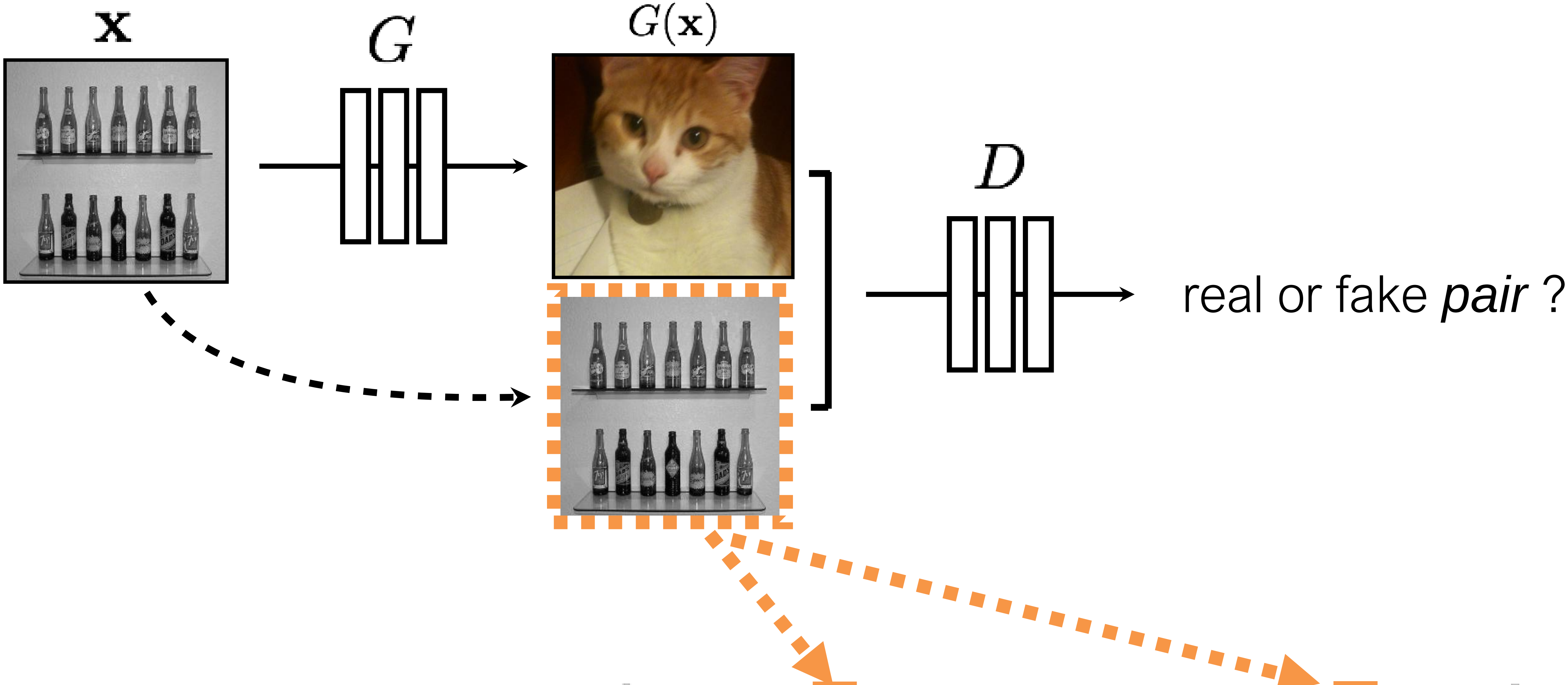
$$\arg\min_G \max_D \; \mathbb{E}_{\mathbf{x},\mathbf{y}}[\; \log D(G(\mathbf{x})) \;+\; \log(1 - D(\mathbf{y})) \;]$$

[Goodfellow et al., 2014]



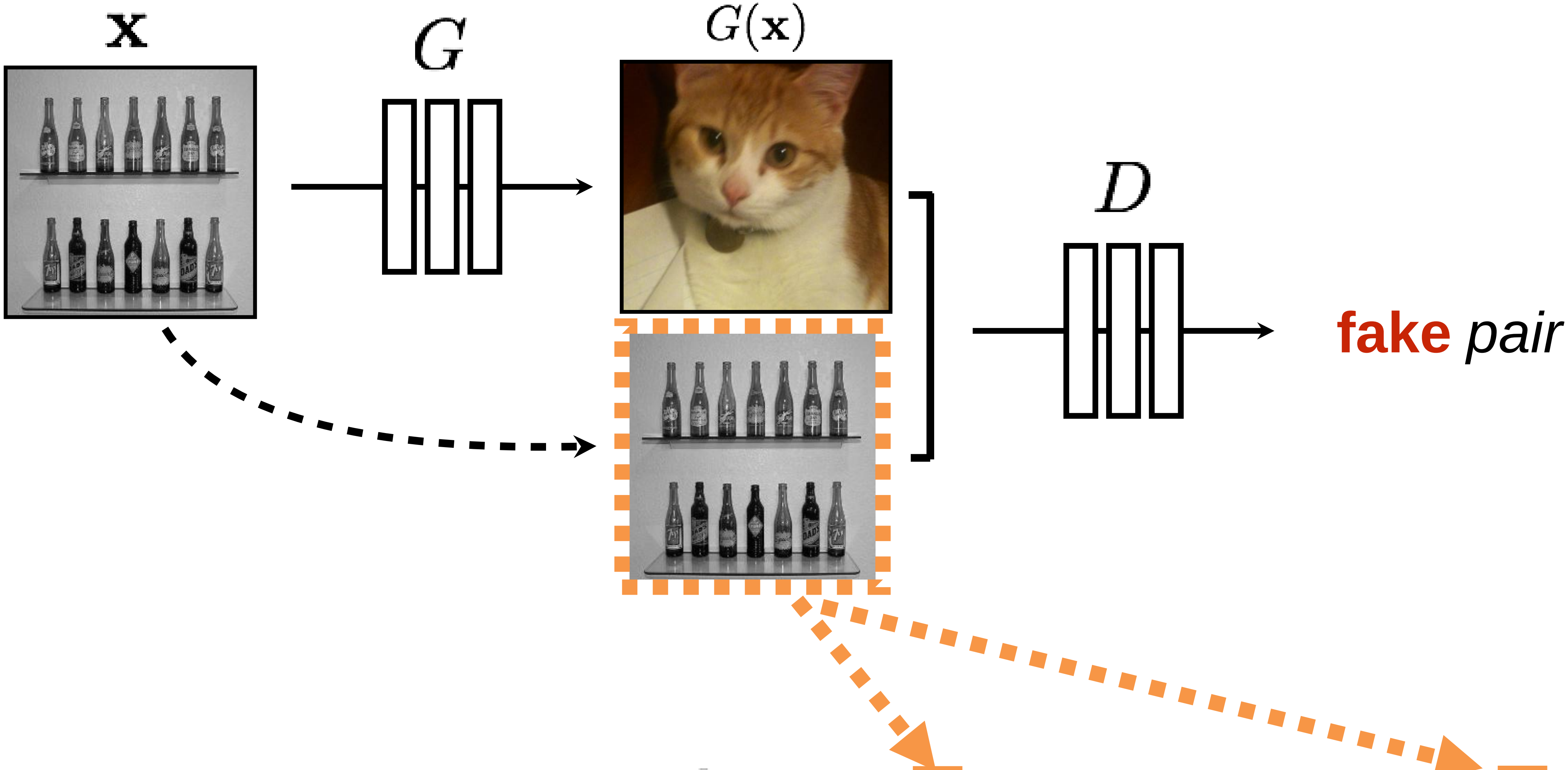
$$\arg\min_G \max_D \; \mathbb{E}_{\mathbf{x},\mathbf{y}}[\; \log D(G(\mathbf{x})) \;\; + \;\; \log(1 - D(\mathbf{y})) \;]$$

[Goodfellow et al., 2014]
[Isola et al., 2017]



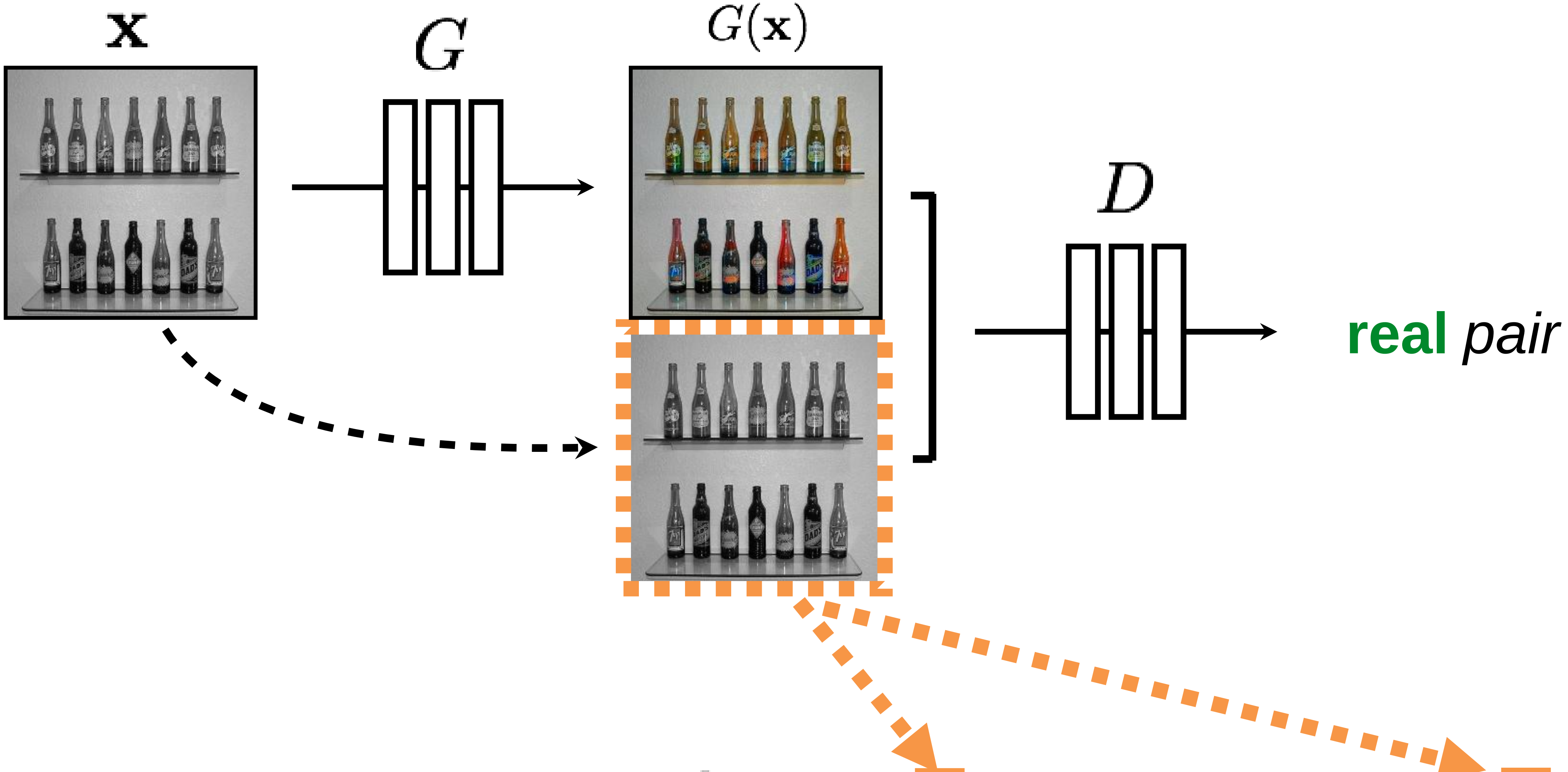
$$\arg\min_G \max_D \; \mathbb{E}_{\mathbf{x},\mathbf{y}}[\; \log D(\mathbf{x}, G(\mathbf{x})) + \log(1 - D(\mathbf{x}, \mathbf{y})) \;]$$

[Goodfellow et al., 2014]
[Isola et al., 2017]



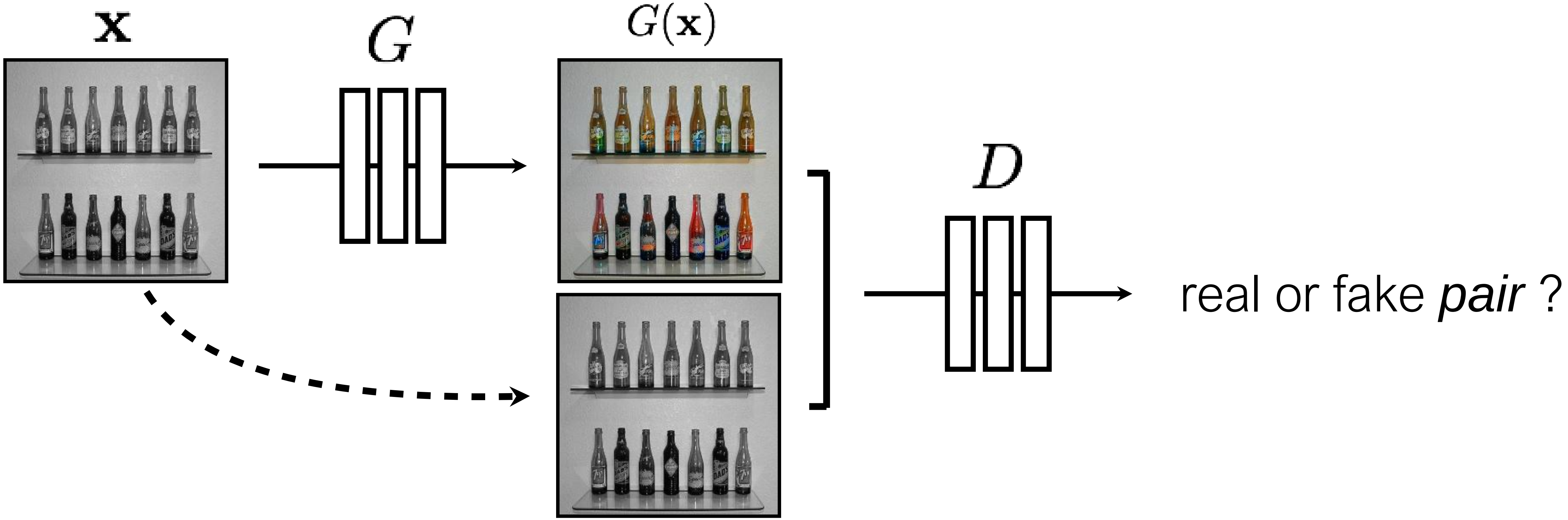
$$\arg\min_G \max_D \ \mathbb{E}_{\mathbf{x},\mathbf{y}}[\ \log D(\mathbf{x}, G(\mathbf{x})) + \log(1 - D(\mathbf{x}, \mathbf{y}))\ ]$$

[Goodfellow et al., 2014]
[Isola et al., 2017]



$$\arg\min_G \max_D \ \mathbb{E}_{\mathbf{x},\mathbf{y}}\left[\ \log D(\mathbf{x}, G(\mathbf{x})) + \log(1 - D(\mathbf{x}, \mathbf{y}))\ \right]$$

[Goodfellow et al., 2014]
[Isola et al., 2017]



$$\arg\min_G \max_D \; \mathbb{E}_{\mathbf{x},\mathbf{y}}[\; \log D(\mathbf{x}, G(\mathbf{x})) + \log(1 - D(\mathbf{x}, \mathbf{y})) \;]$$

[Goodfellow et al., 2014]
[Isola et al., 2017]

More Examples of Image-to-Image Translation with GANs

- We have pairs of corresponding training images
- Conditioned on one of the images, sample from the distribution of likely corresponding images

Edges to Image

Aerial Photo To Map

Segmentation to Street Image

BW → Color

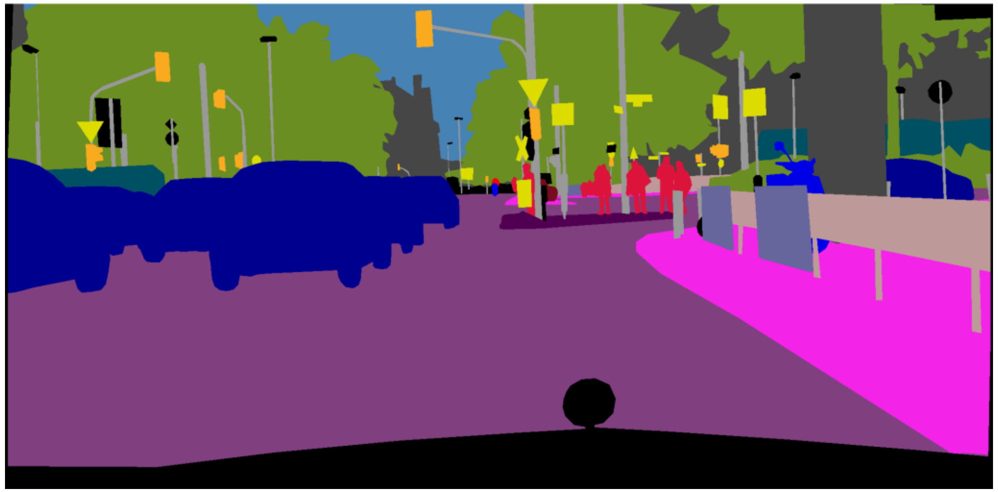
Input Output Input Output Input Output

Data from [Russakovsky et al. 2015]

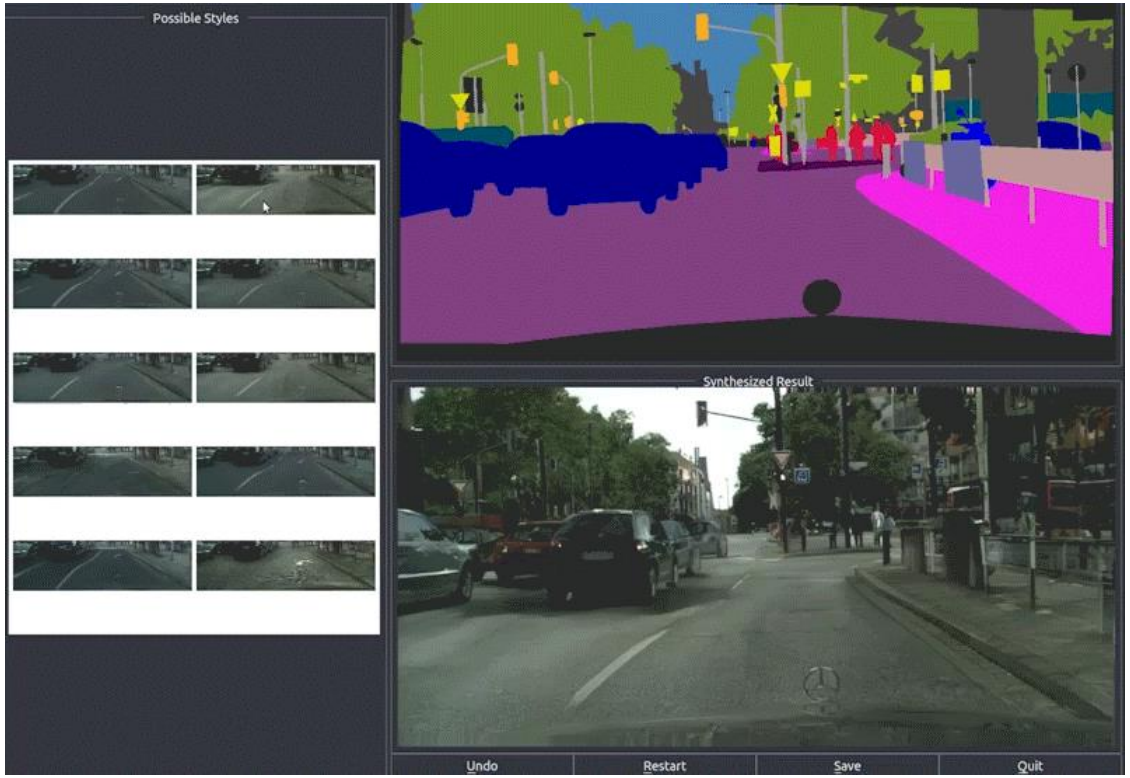
Data from
[maps.google.com]

Labels → Street Views

Input labels



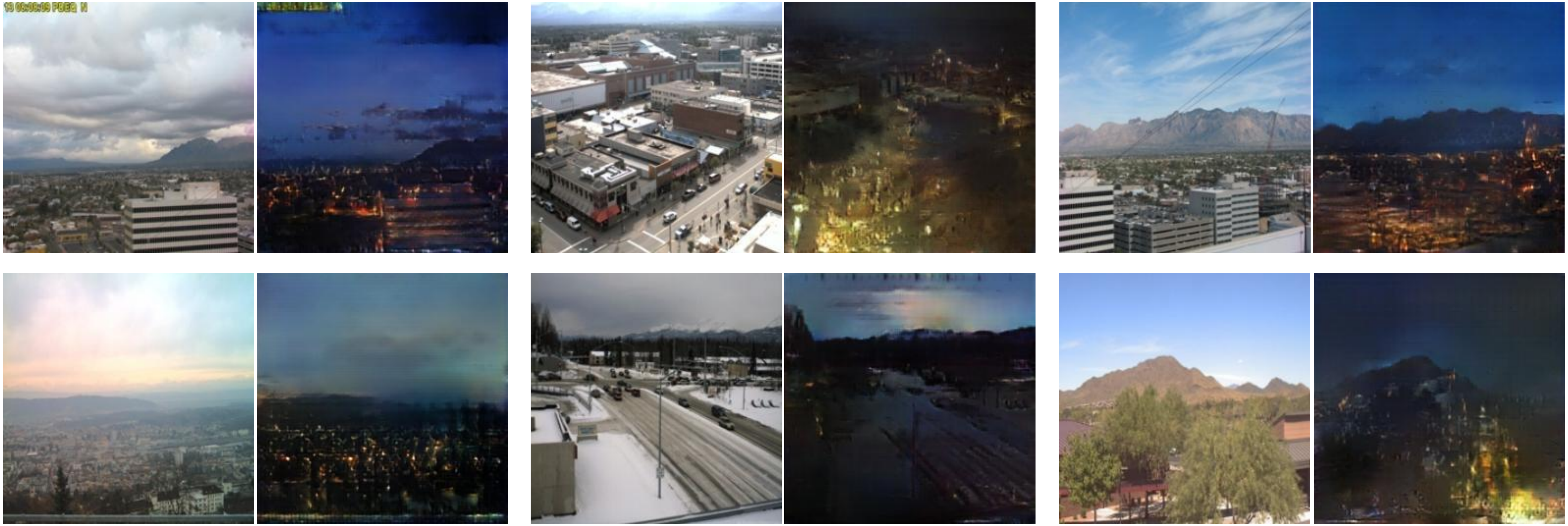
Synthesized image



Data from [Wang et al, 2018]

Day → Night

Input Output Input Output Input Output



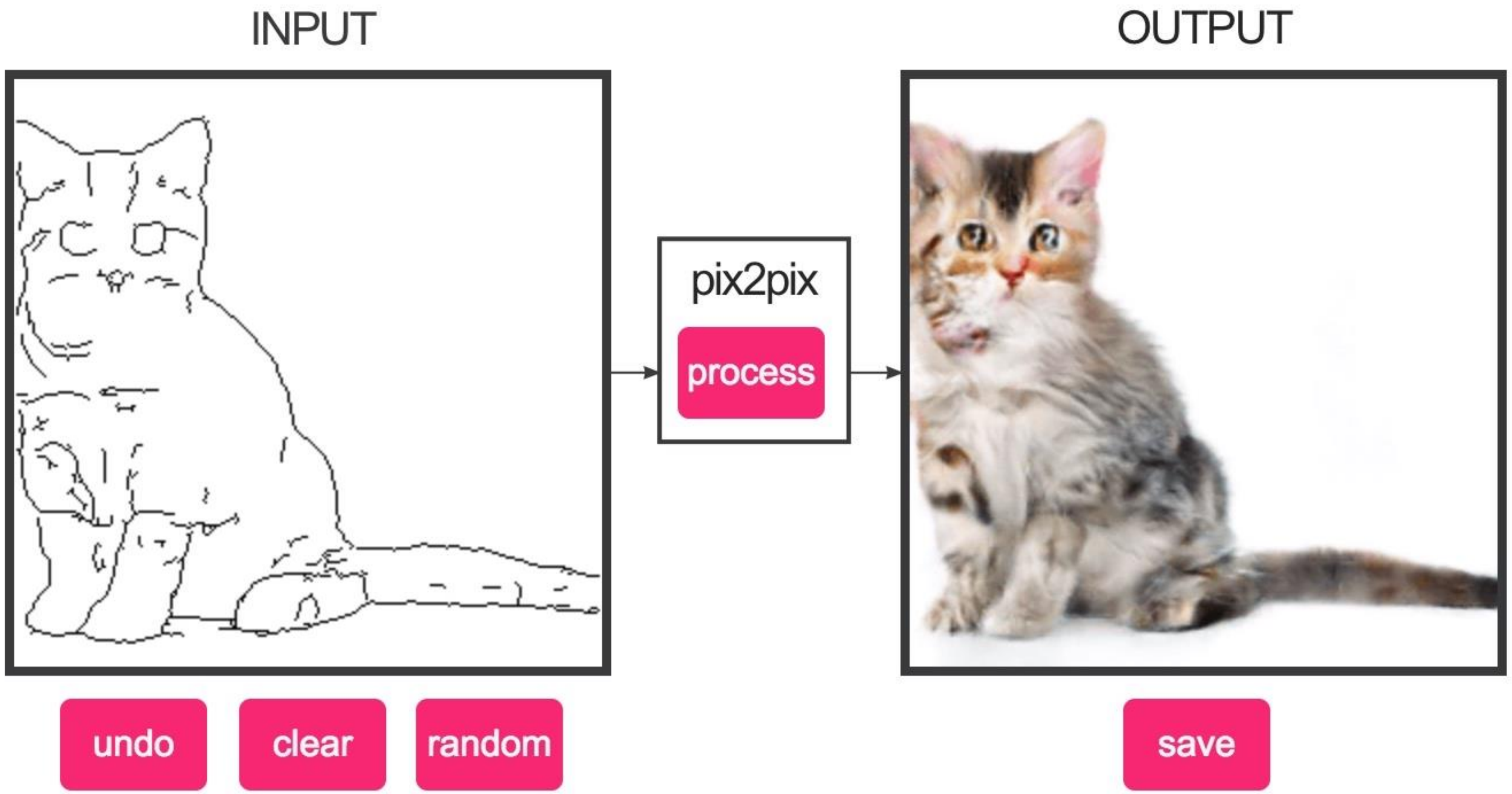
Data from [Laffont et al., 2014]

Edges → Images

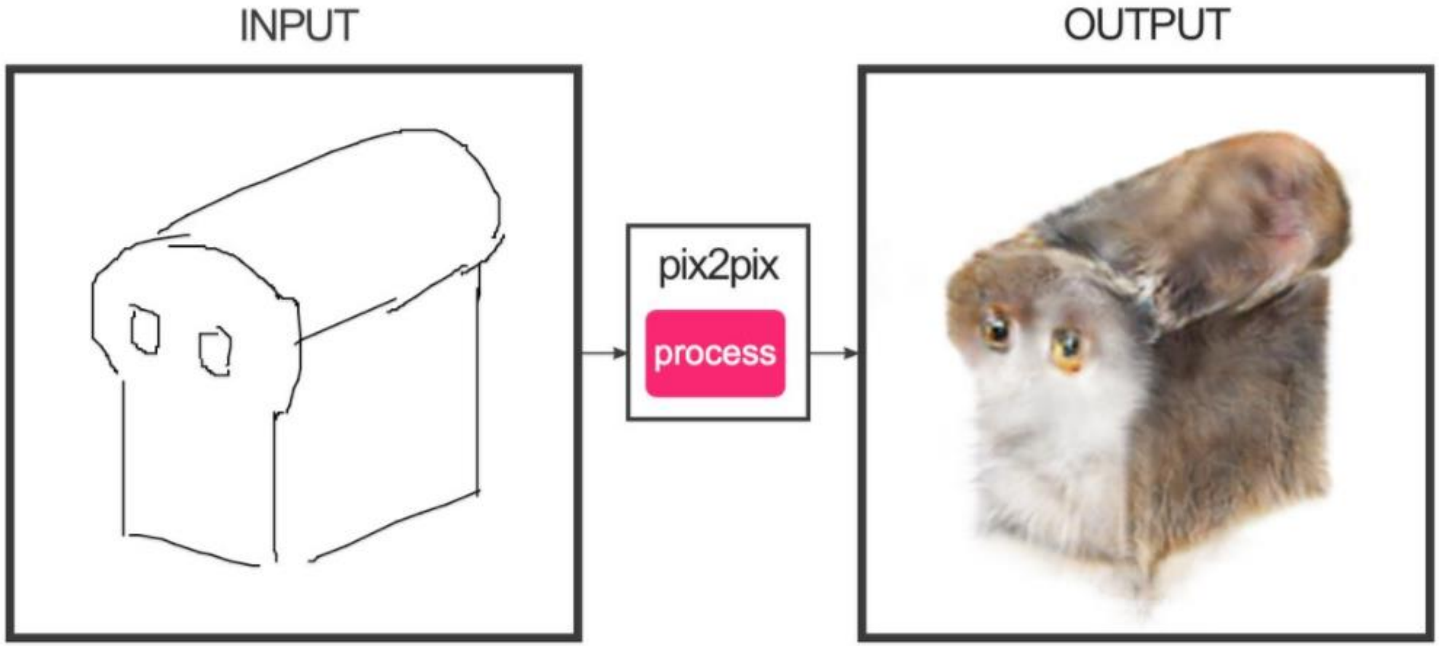
Input Output Input Output Input Output

Edges from [Xie & Tu, 2015]

Demo



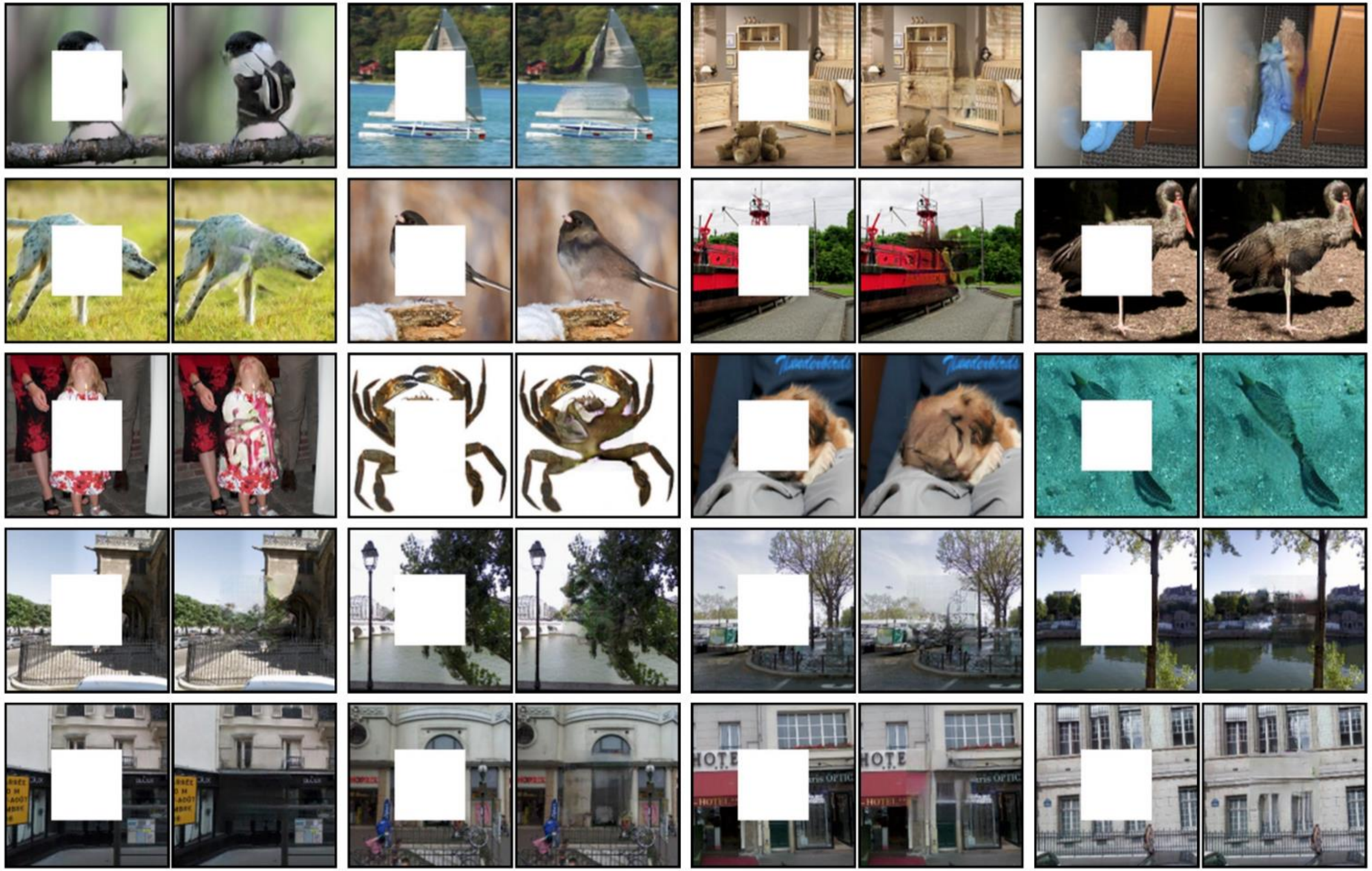
https://affinelayer.com/pixsrv/



Ivy Tasi @ivymyt

Vitaly Vidmirov @vvid

Image Inpainting



Data from [Pathak et al., 2016]

Pose-guided Generation

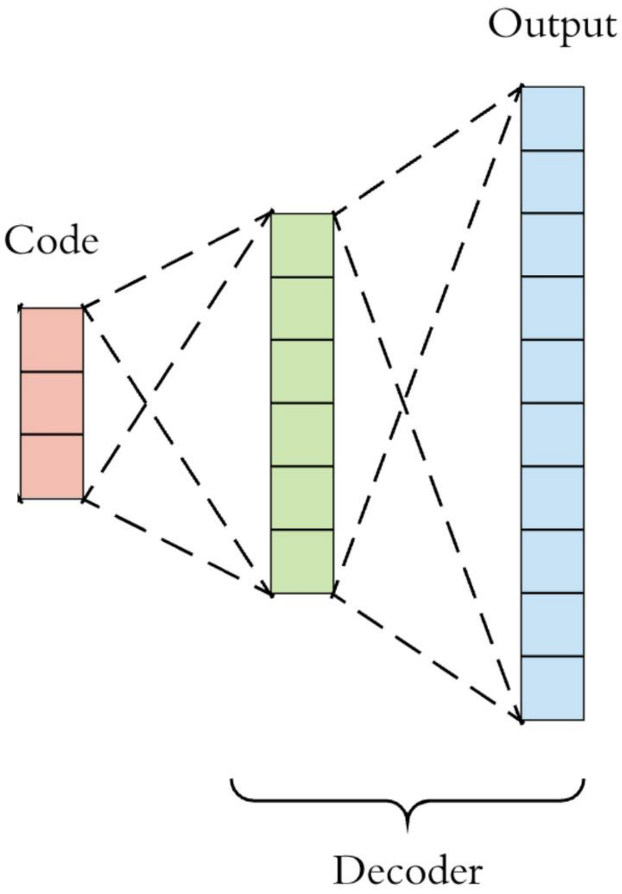
Data from [Ma et al., 2018]

Challenges —> Solutions

- Output is high-dimensional, structured object
  - Approach: Use a deep net, D, to analyze output!
- Uncertainty in mapping; many plausible outputs
  - Approach: D only cares about "plausibility", doesn't hedge

Unconditional GANs:
Learning an image manifold for a category

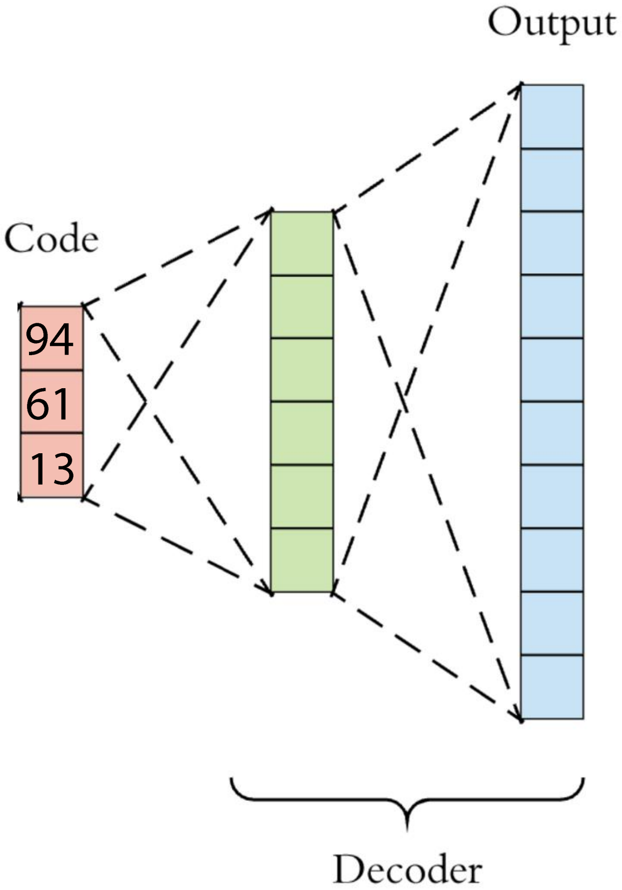
Category-specific
image dataset (FFHQ)



Latent code ("noise")-to-image decoder network

Unconditional GANs:
Learning an image manifold for a category

Category-specific
image dataset (FFHQ)

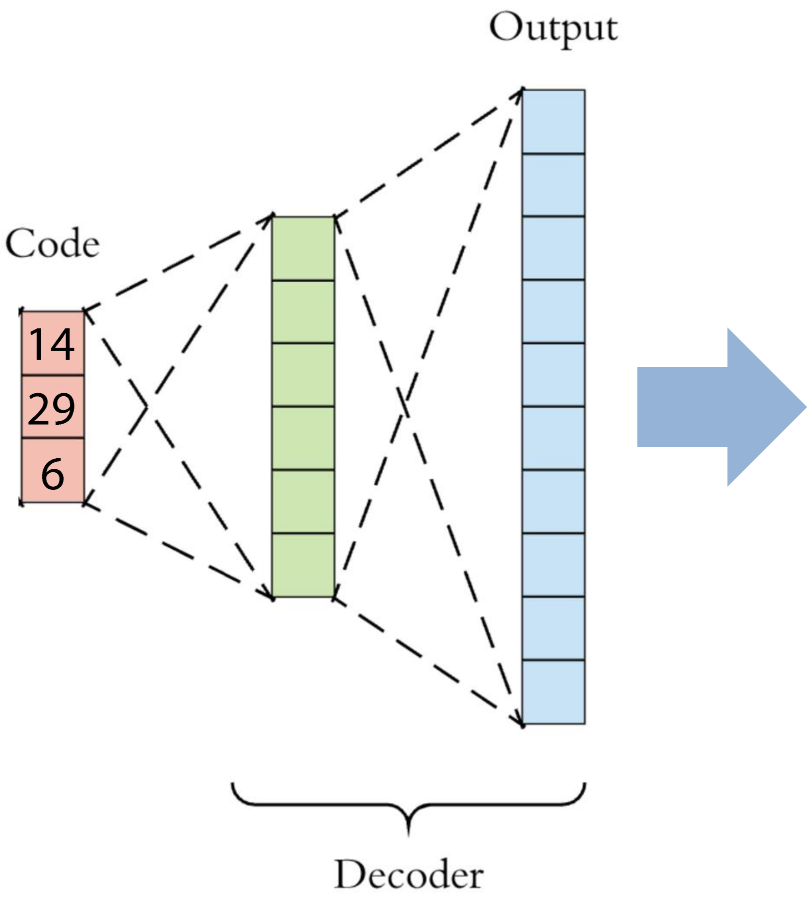


Latent code ("noise")-to-image decoder network

Output image

Unconditional GANs:
Learning an image manifold for a category

Category-specific
image dataset (FFHQ)

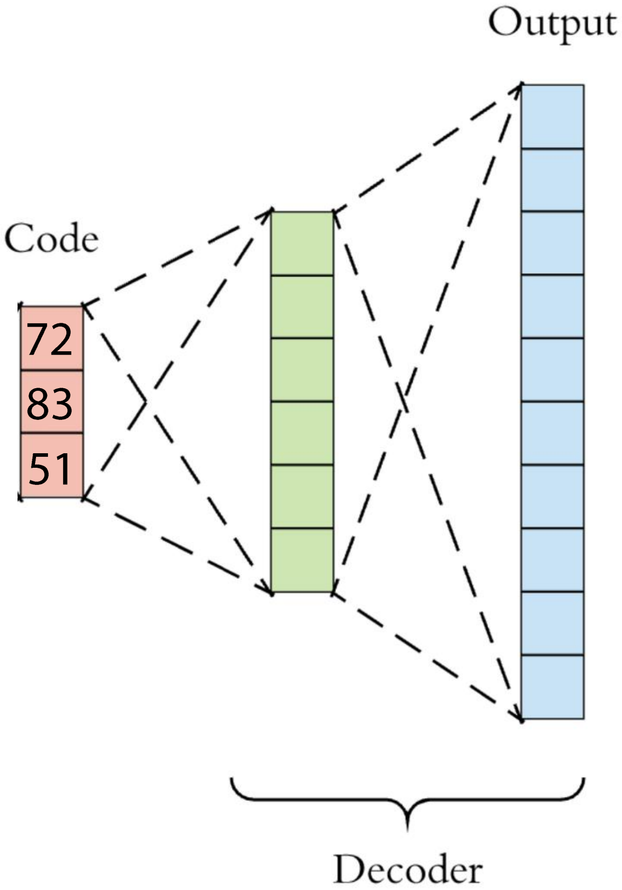


Latent code (“noise”)-to-image decoder network

Output image

Unconditional GANs:
Learning an image manifold for a category

Category-specific
image dataset (FFHQ)



Output image

Latent code (“noise”)-to-image decoder network

Example: Randomly Sampling the Space of Face Images

A

B

Which face is real?

Example: Randomly Sampling the Space of Face Images

A

B

[Which face is real](#)?

StyleGAN

A Style-Based Generator Architecture for Generative Adversarial Networks

Tero Karras, Samuli Laine, Timo Aila

https://github.com/NVlabs/stylegan

StyleGAN2 [2020]

Analyzing and Improving the Image Quality of StyleGAN
Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, Timo Aila

https://github.com/NVlabs/stylegan2

StyleGAN3 [2021]

Alias-Free Generative Adversarial Networks (StyleGAN3)

Tero Karras, Miika Aittala, Samuli Laine, Erik Härkönen, Janne Hellsten, Jaakko Lehtinen, Timo Aila

GAN models trained on animal faces: interpolating between latent codes

GAN models trained on MetFaces: interpolating between latent codes

GANs for 3D

EG3D: Efficient Geometry-aware 3D Generative Adversarial Networks

Eric Ryan Chan [*1,2] Connor Zhizhen Lin [*1] Matthew Aaron Chan [*1]
Koki Nagano [*2] Boxiao Pan [1] Shalini De Mello [2] Orazio Gallo [2]
Leonidas Guibas [1] Jonathan Tremblay [2] Sameh Khamis [2] Tero Karras [2]
Gordon Wetzstein [1]

[1] Stanford University [2] NVIDIA

[*] Equal contribution.

https://nvlabs.github.io/eg3d

Limitations

- The unconditional models above must be trained per-category:
  - We have a separate model for every category – an animal face model, broccoli model, horse model, etc…
- What if we want to generate an image from **any** description?

- -> diffusion and text-to-image models

Recall: The Space of All Images

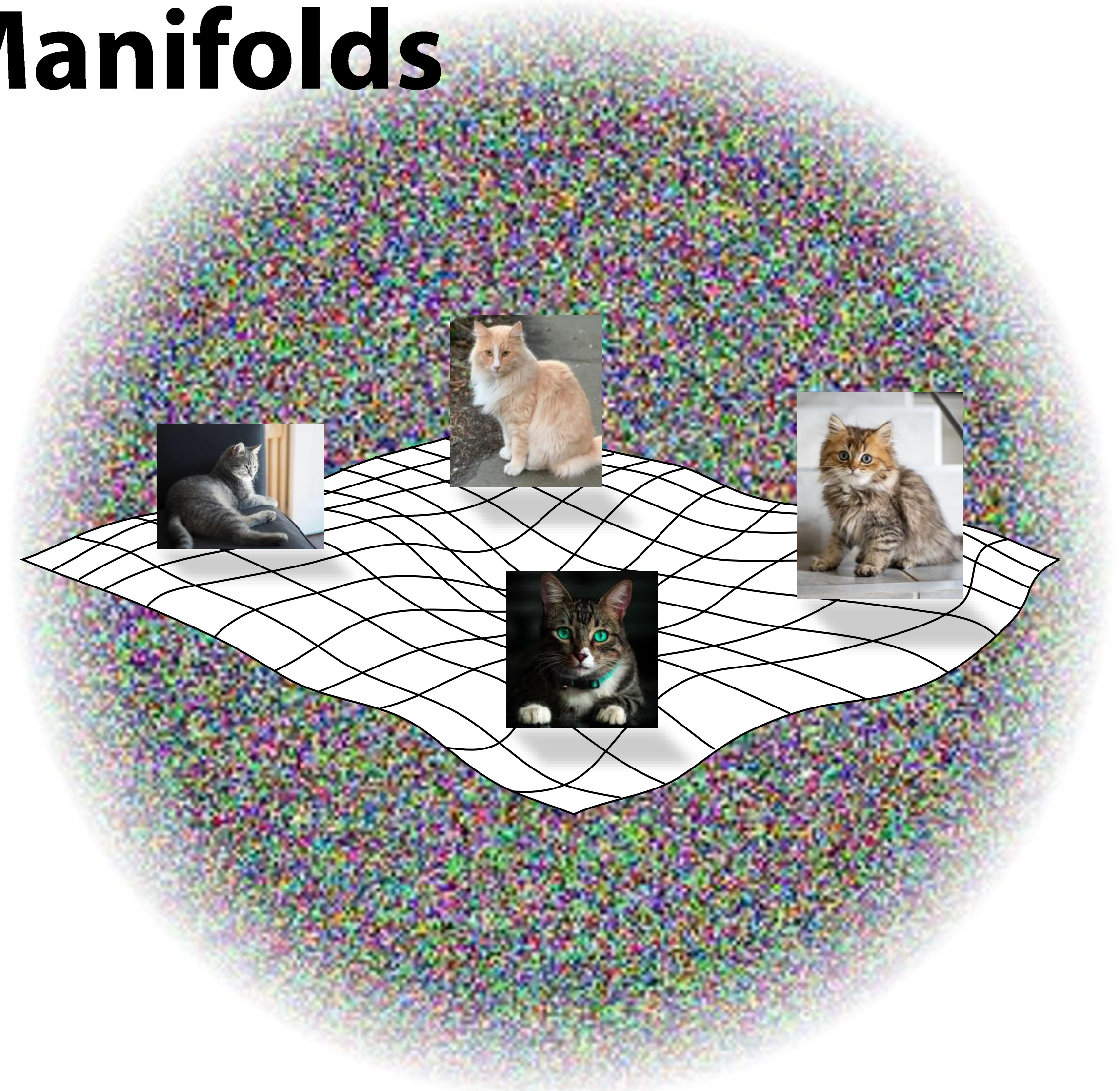
- Lets consider the space of all 100x100 images
- Now lets randomly sample that space…
- Conclusion: Most images are noise

Question:
What do we expect a random uniform sample of all images to look like?

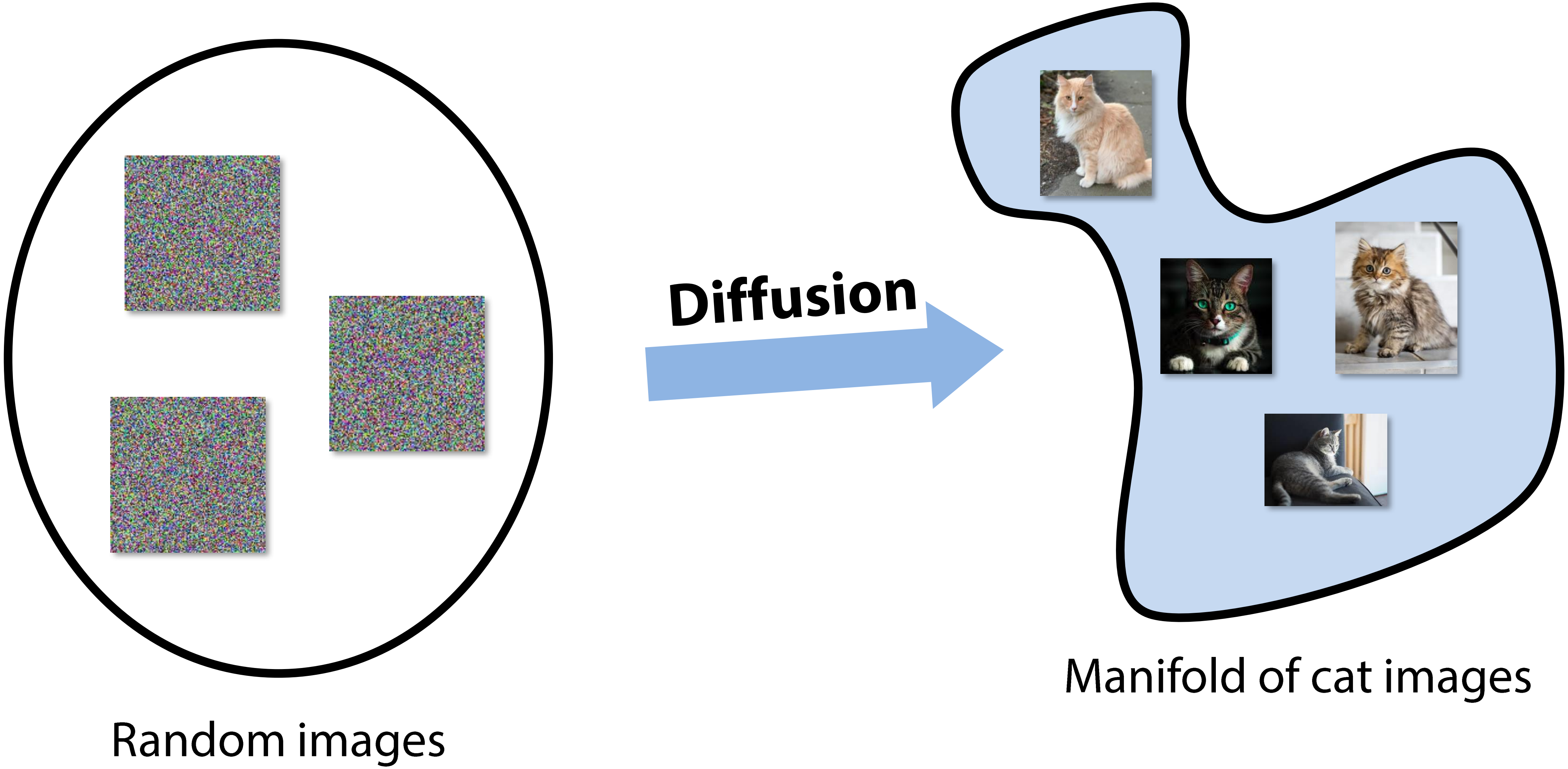
```
pixels = np.random.rand(100,100,3)
```

Recall: Natural Image Manifolds

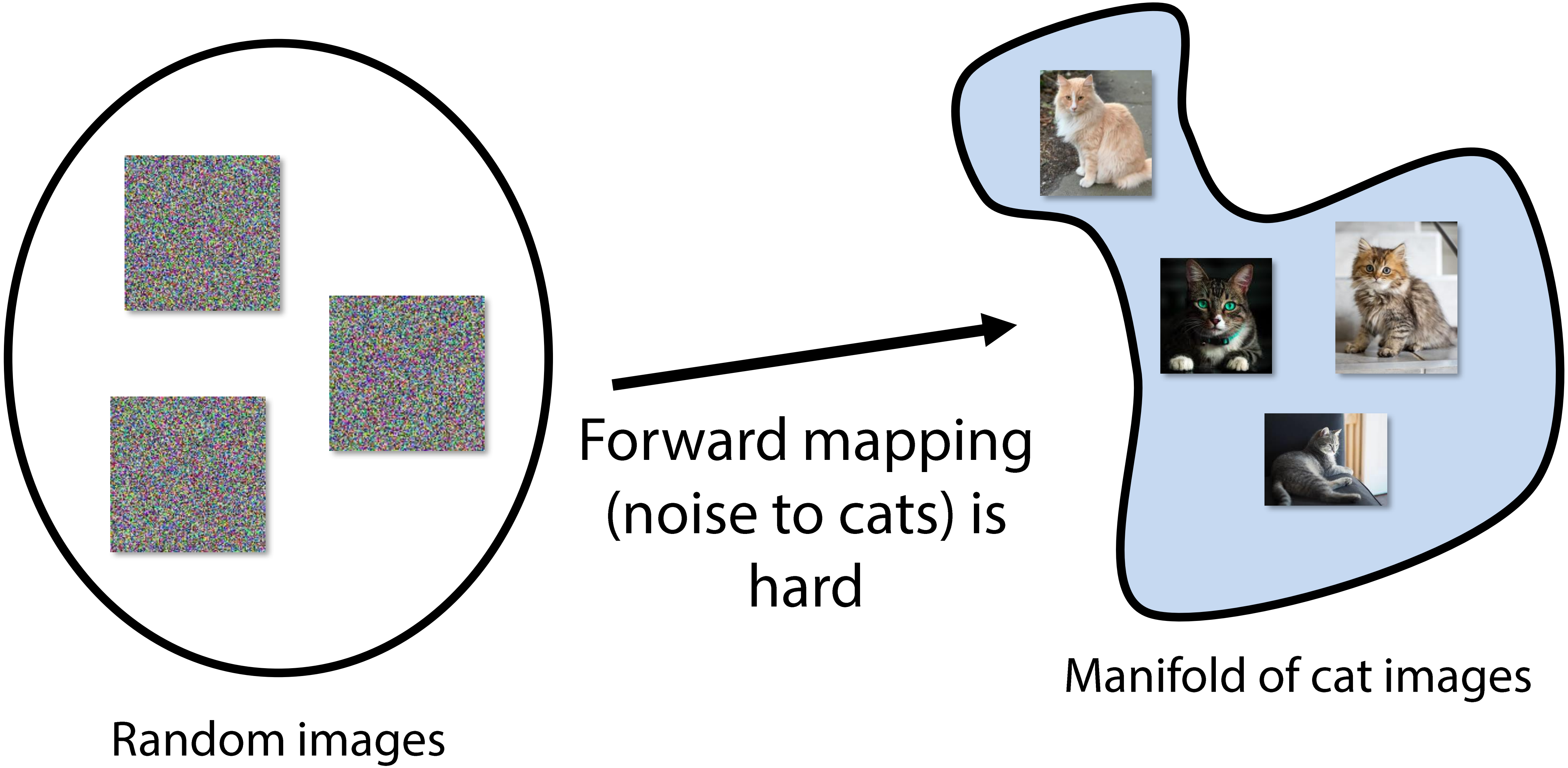
- Most images are "noise"
- "Meaningful" images tend to form some manifold within the space of all images
- Images of a particular class fall on manifolds within that manifold…



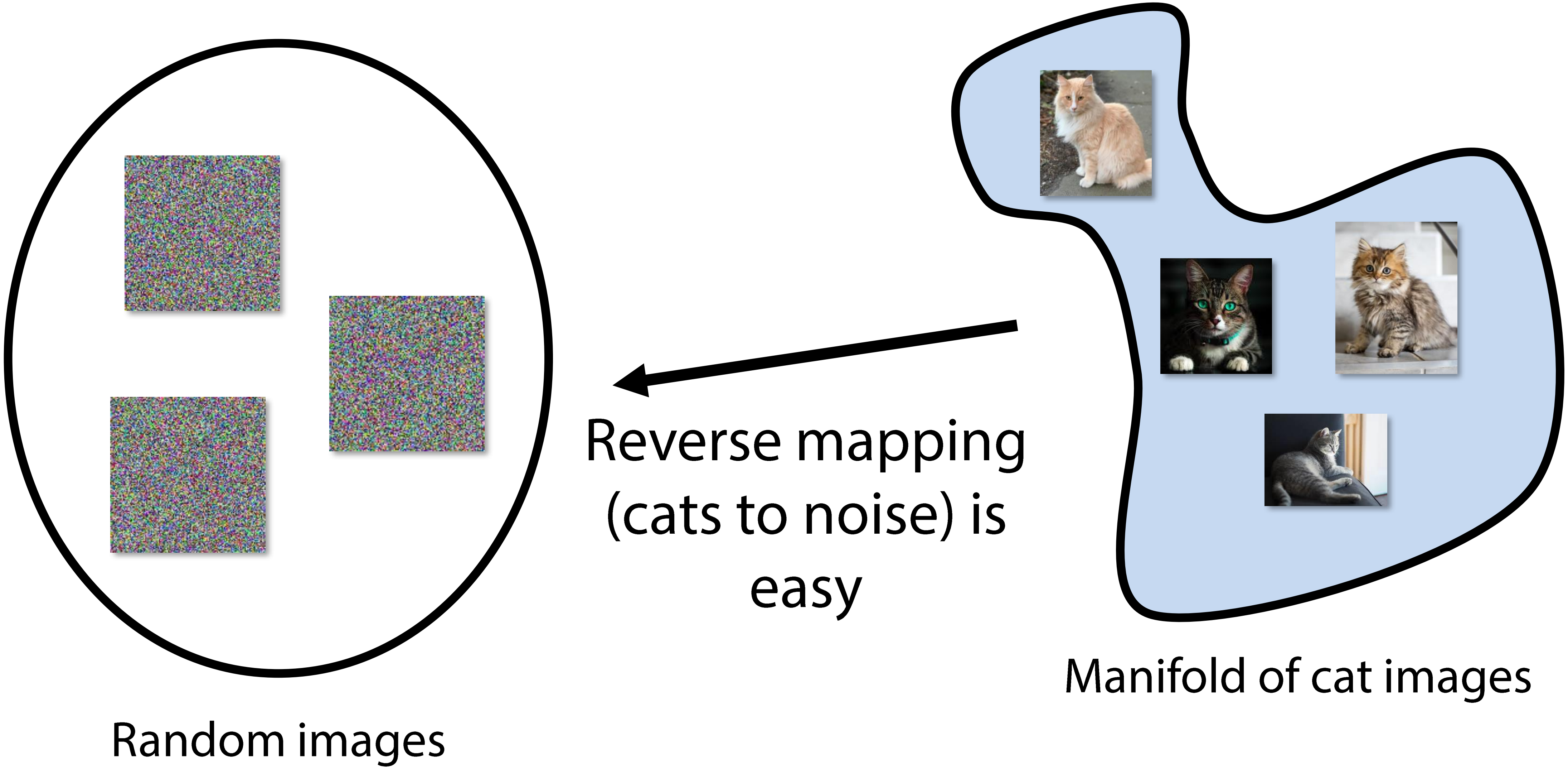
The Space of All Images



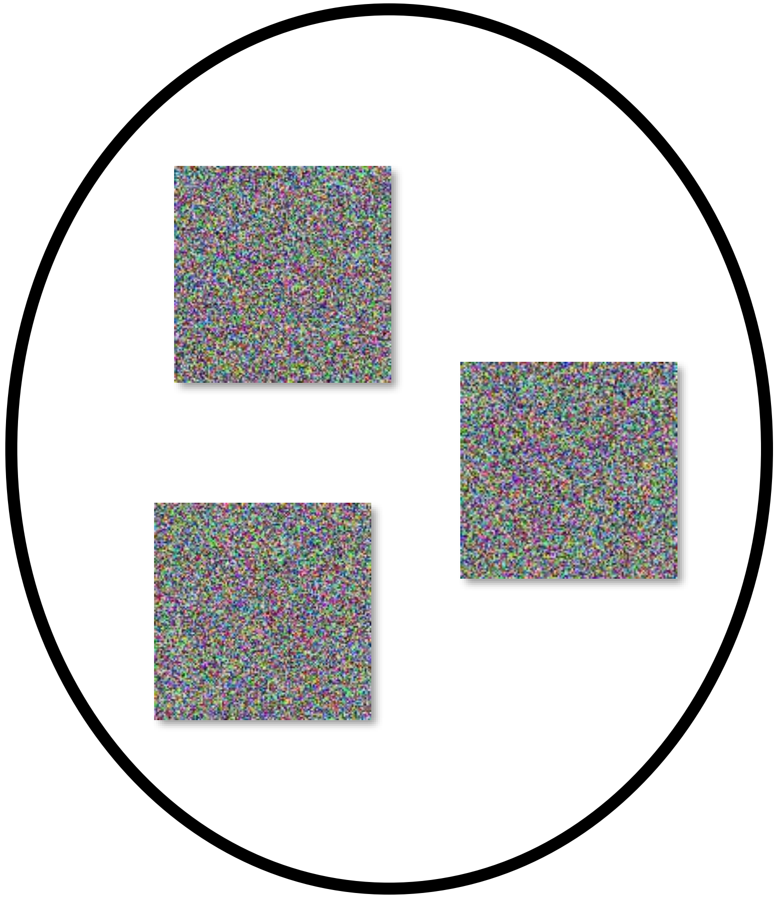
Slide concept: Steve Seitz



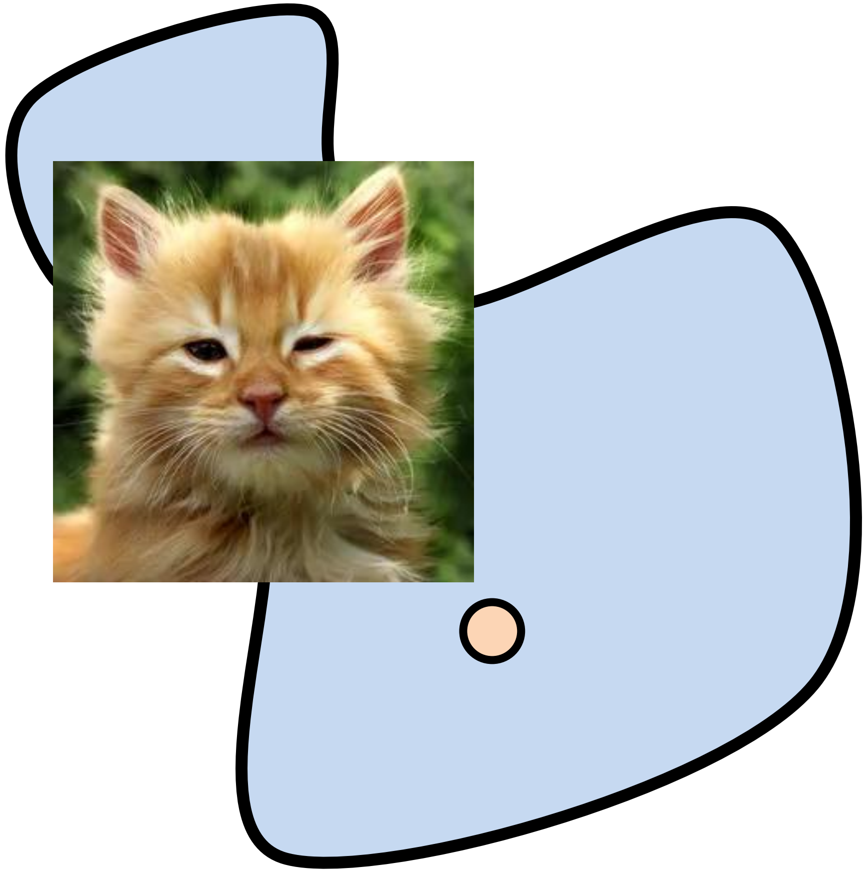
Slide concept: Steve Seitz



Slide concept: Steve Seitz

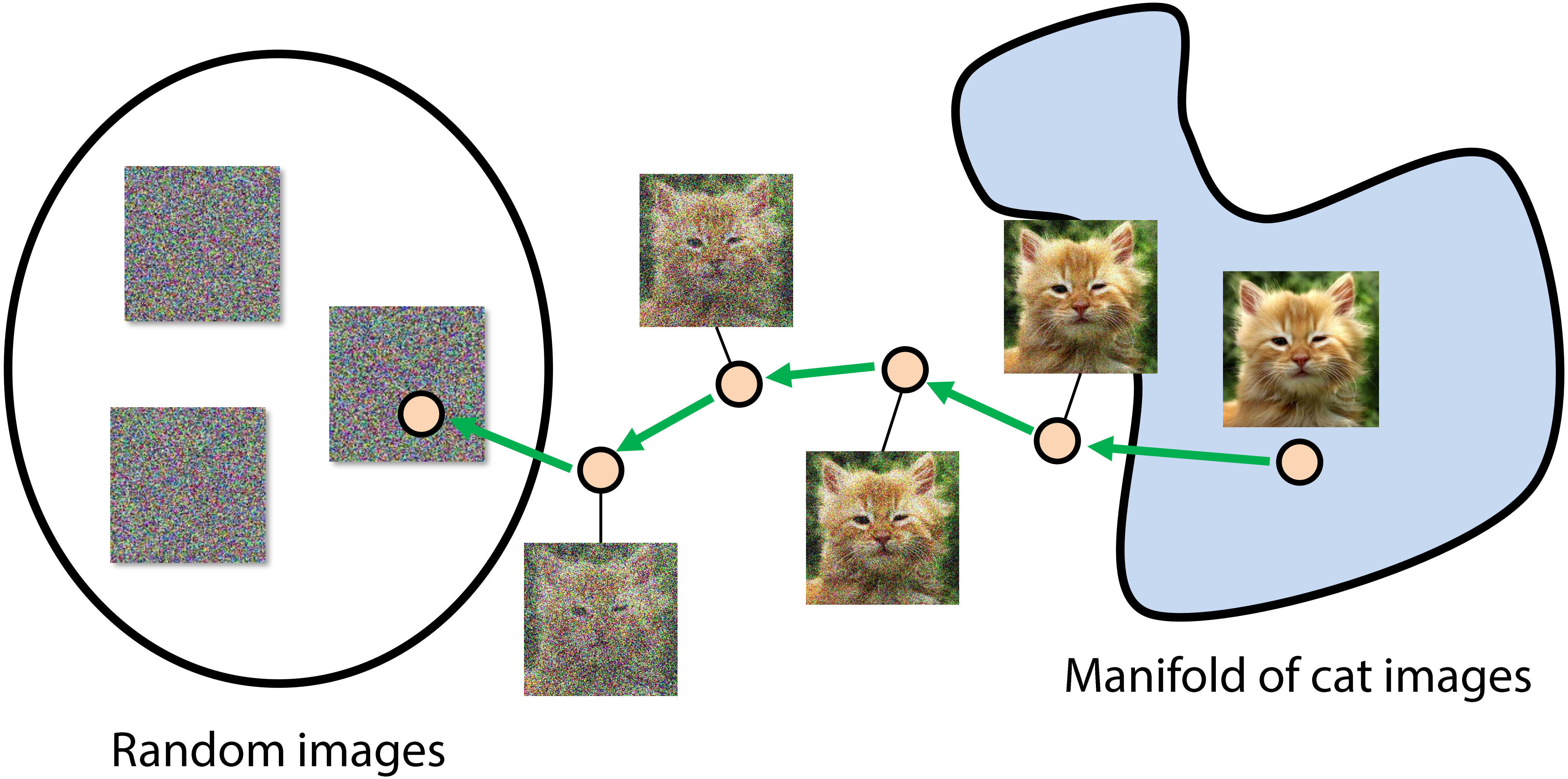


Random images

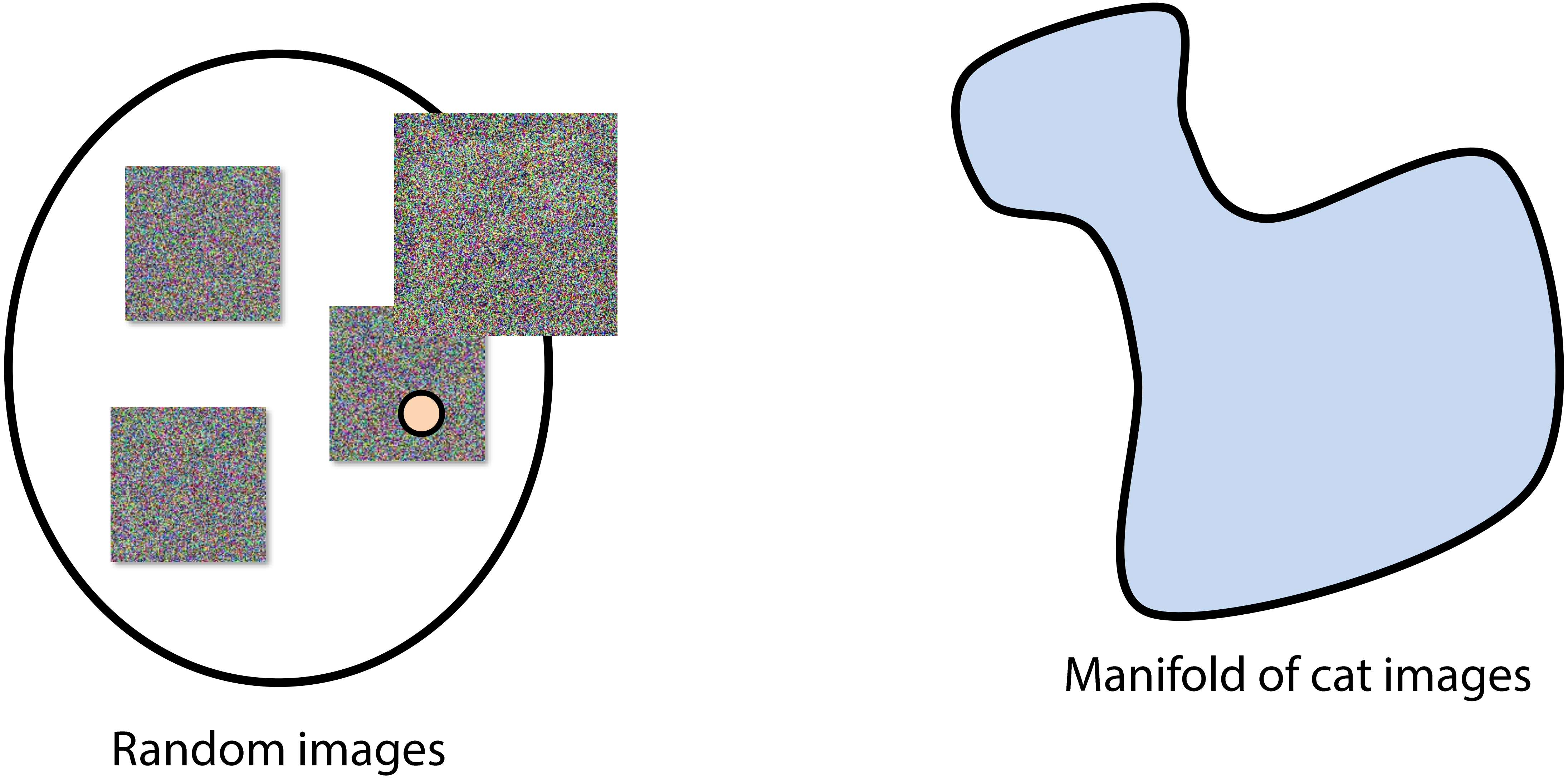


Manifold of cat images

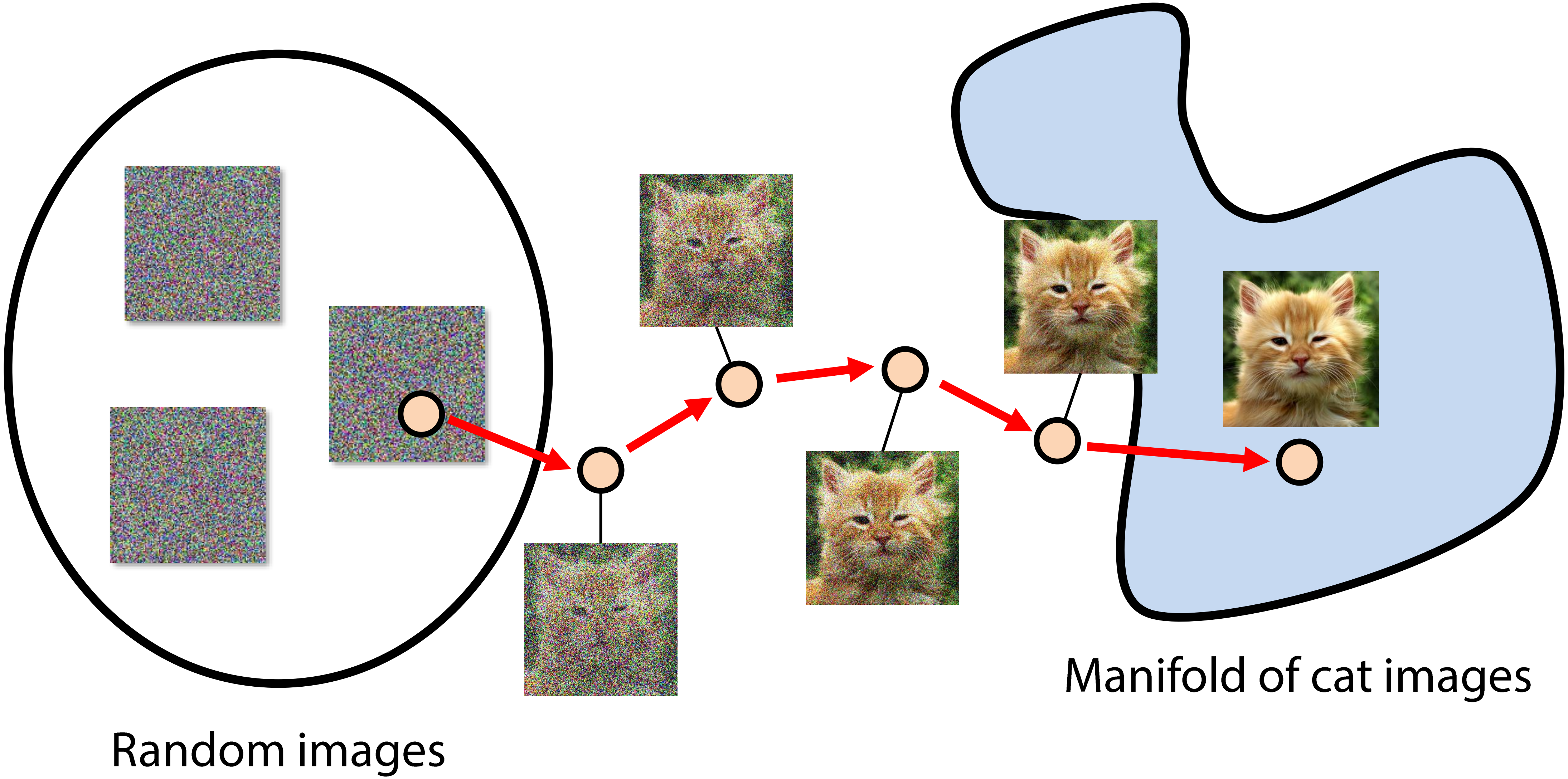
Slide concept: Steve Seitz



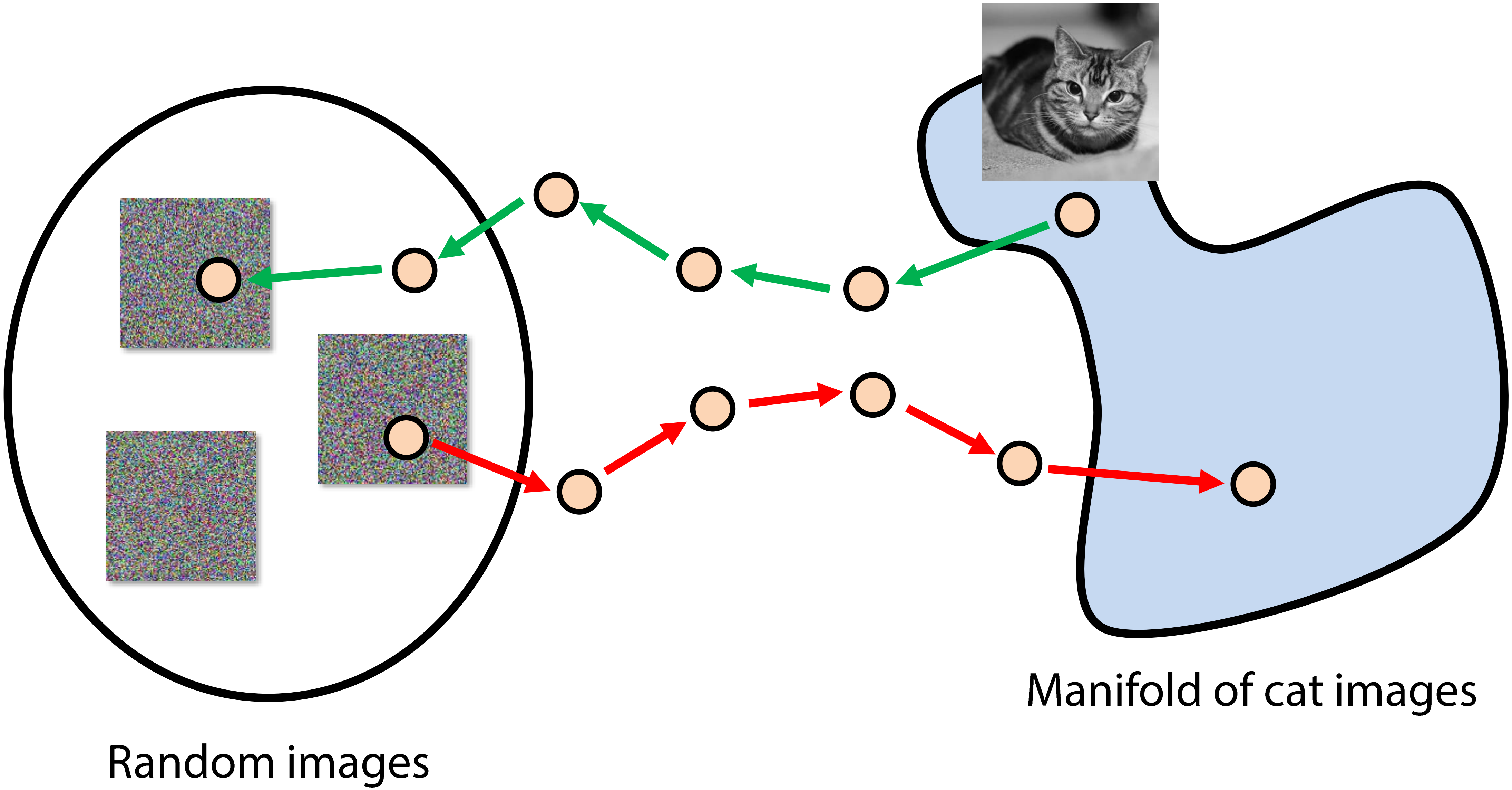
Slide concept: Steve Seitz



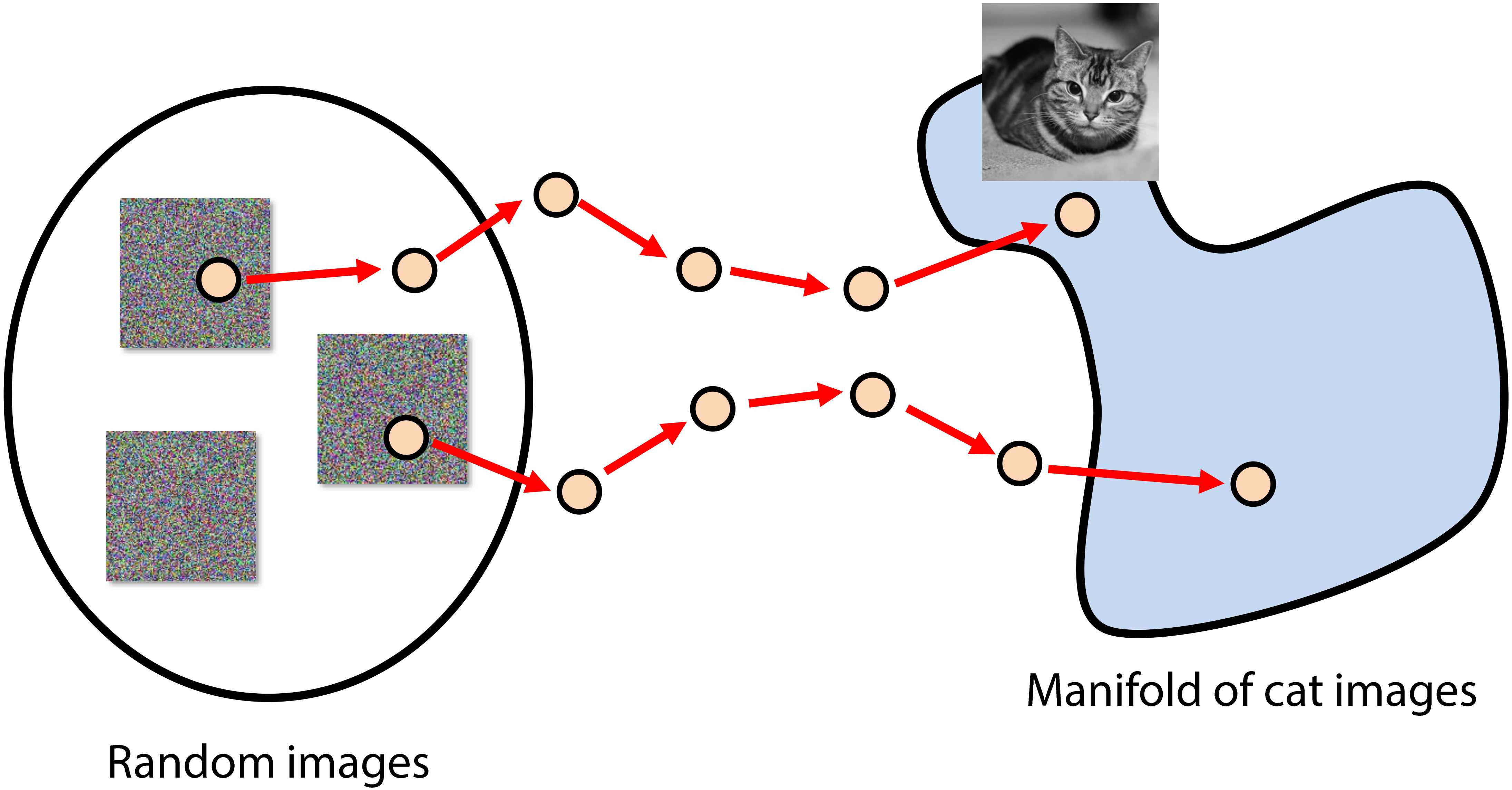
Slide concept: Steve Seitz



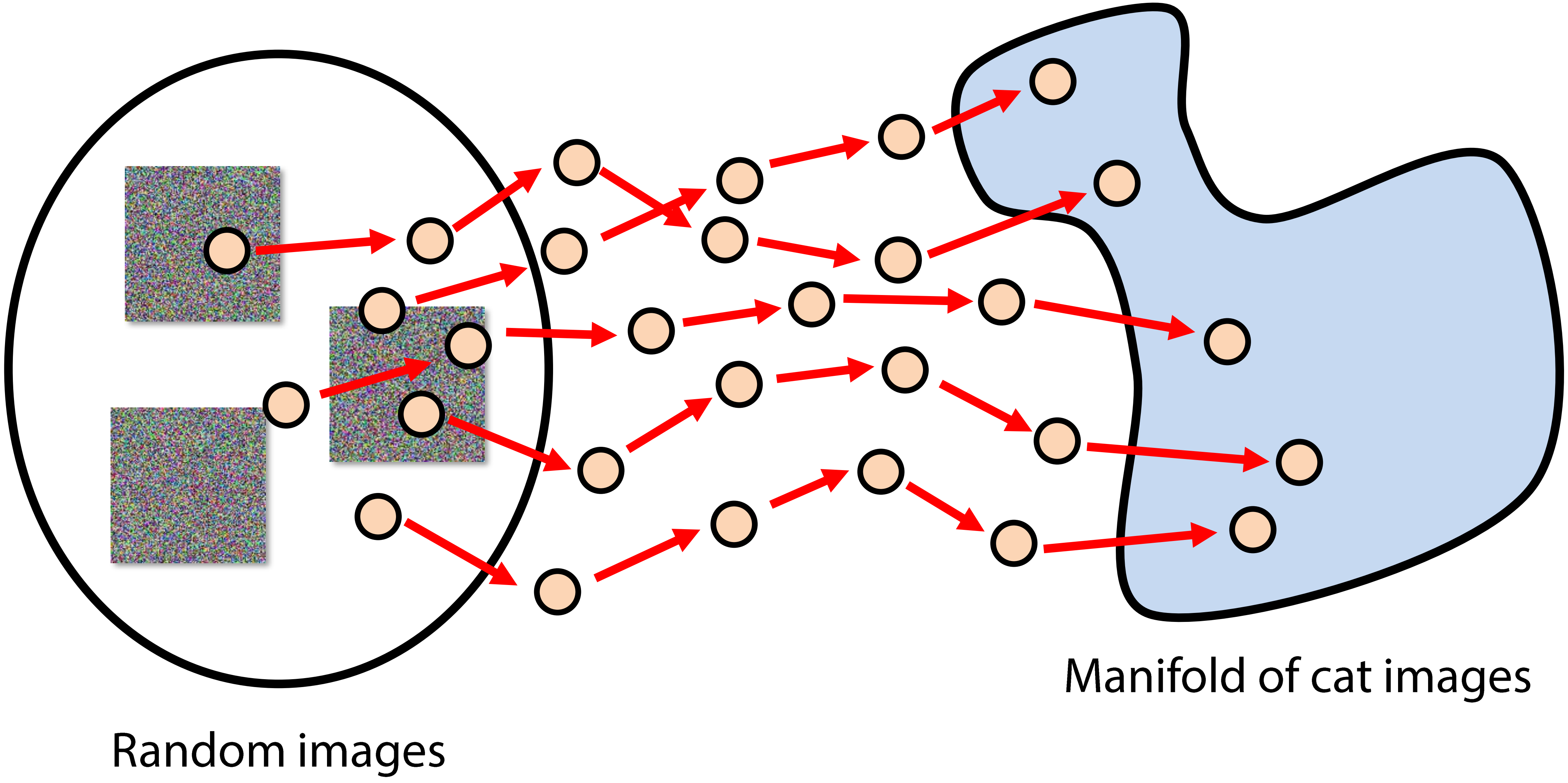
Slide concept: Steve Seitz



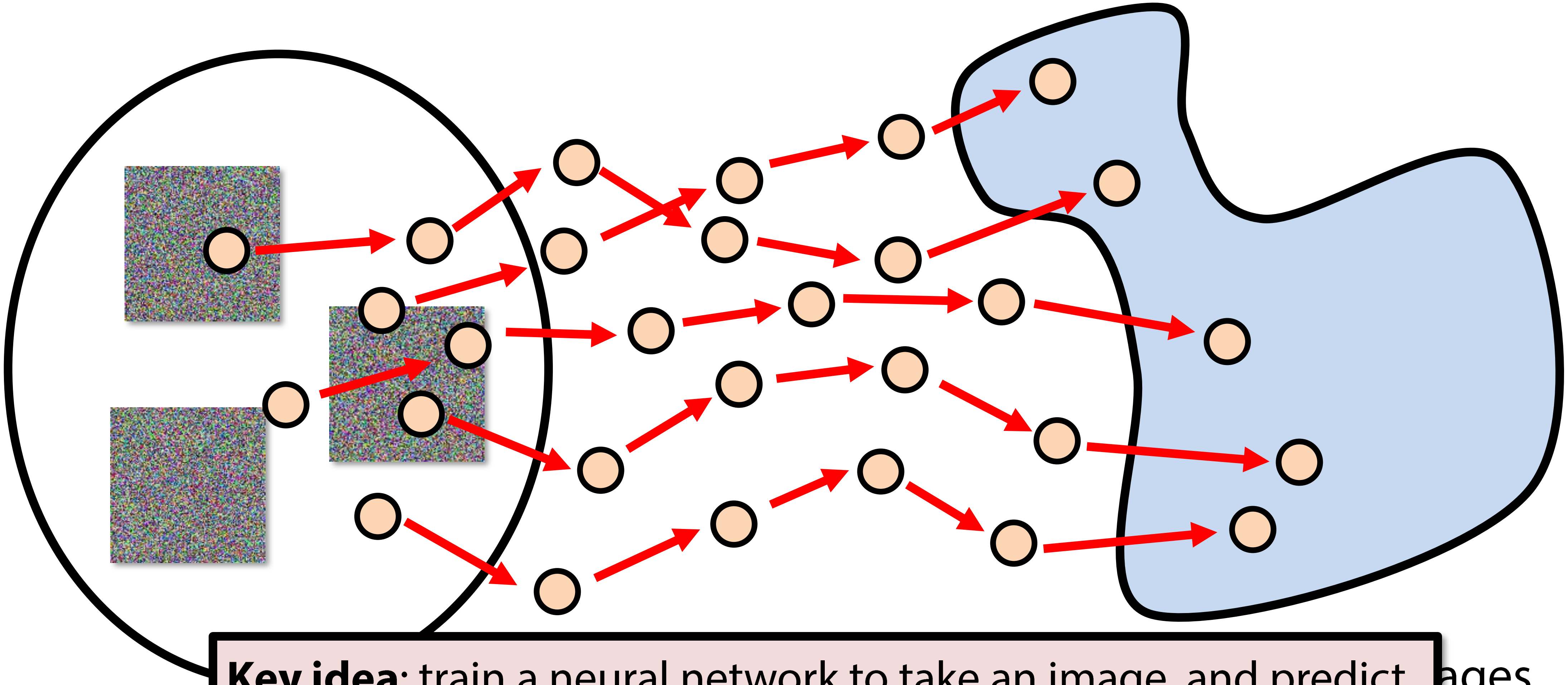
Slide concept: Steve Seitz



Slide concept: Steve Seitz



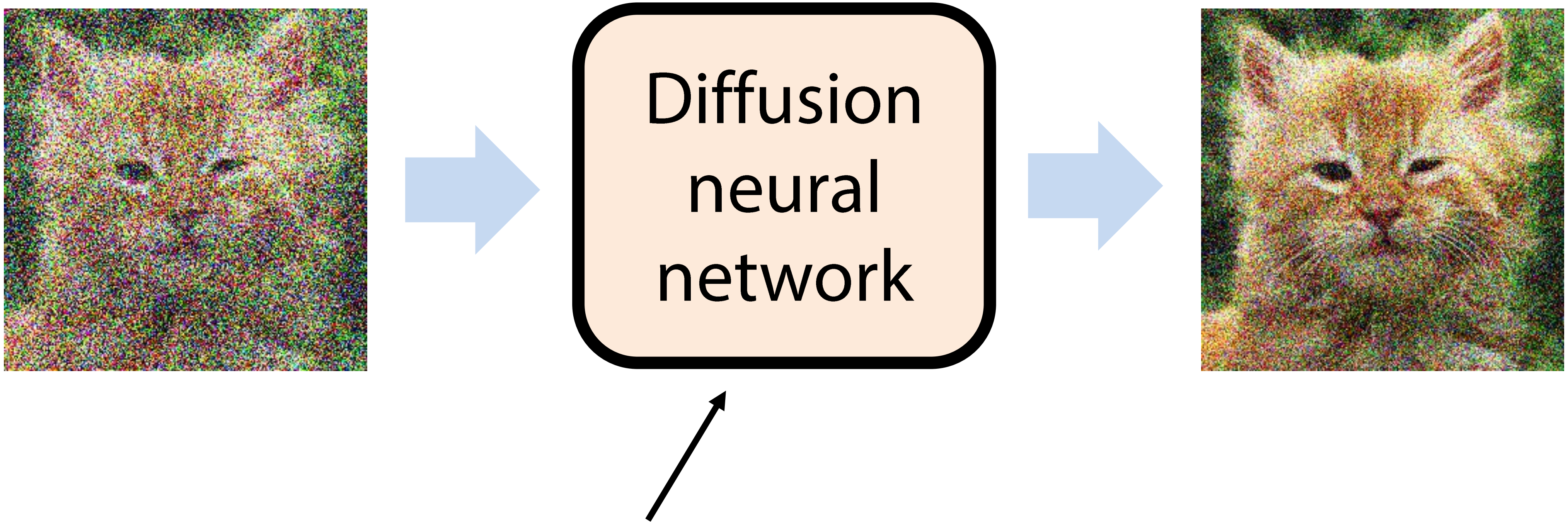
Slide concept: Steve Seitz



Key idea: train a neural network to take an image, and predict the corresponding arrow above; that is, predict to convert a noisy image to a slightly less noisy image that is closer to the desired image manifold, using the examples above to train.

Slide concept: Steve Seitz

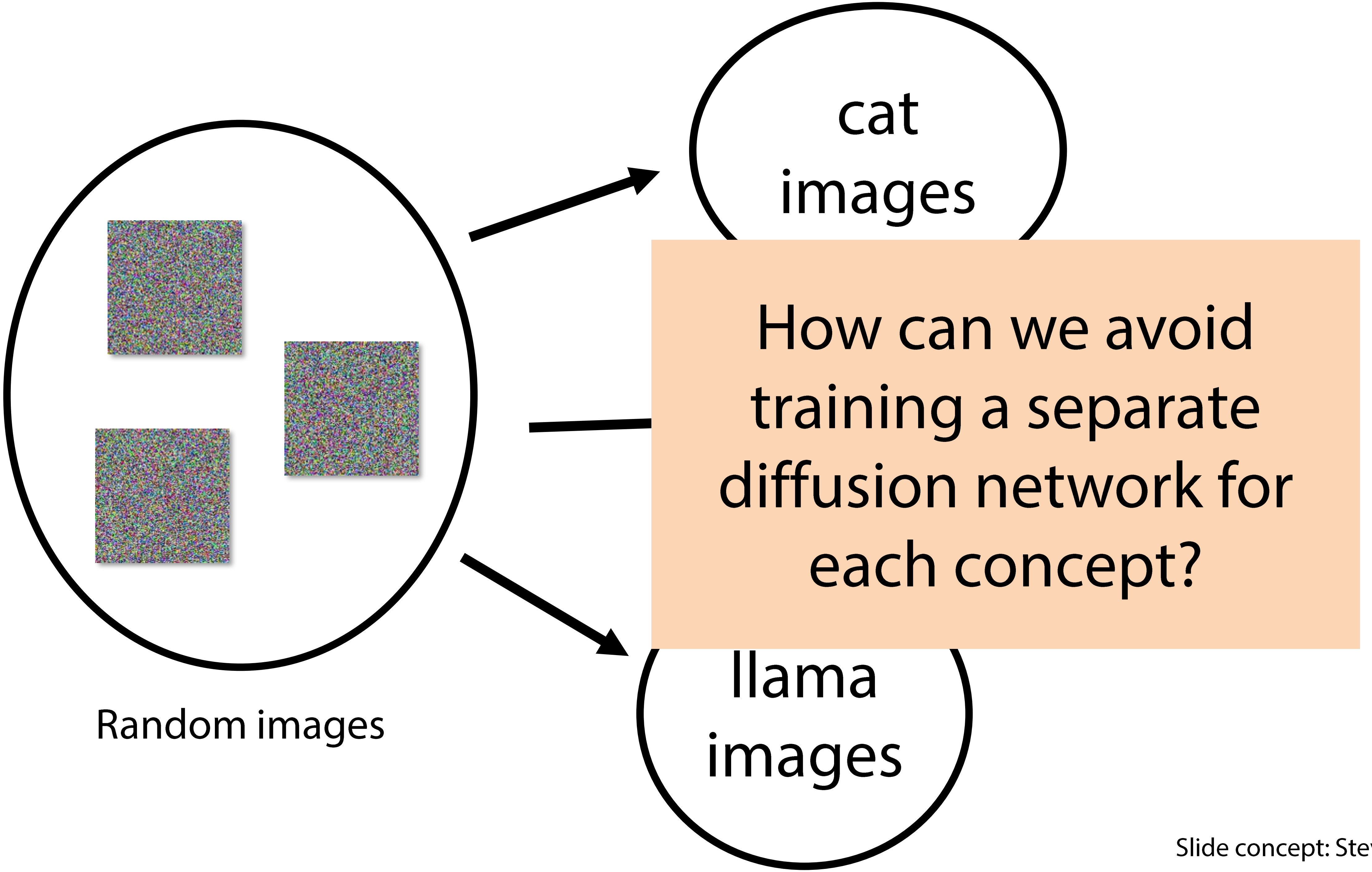
Denoising diffusion neural network



This network can be a U-Net or other suitable image-to-image network

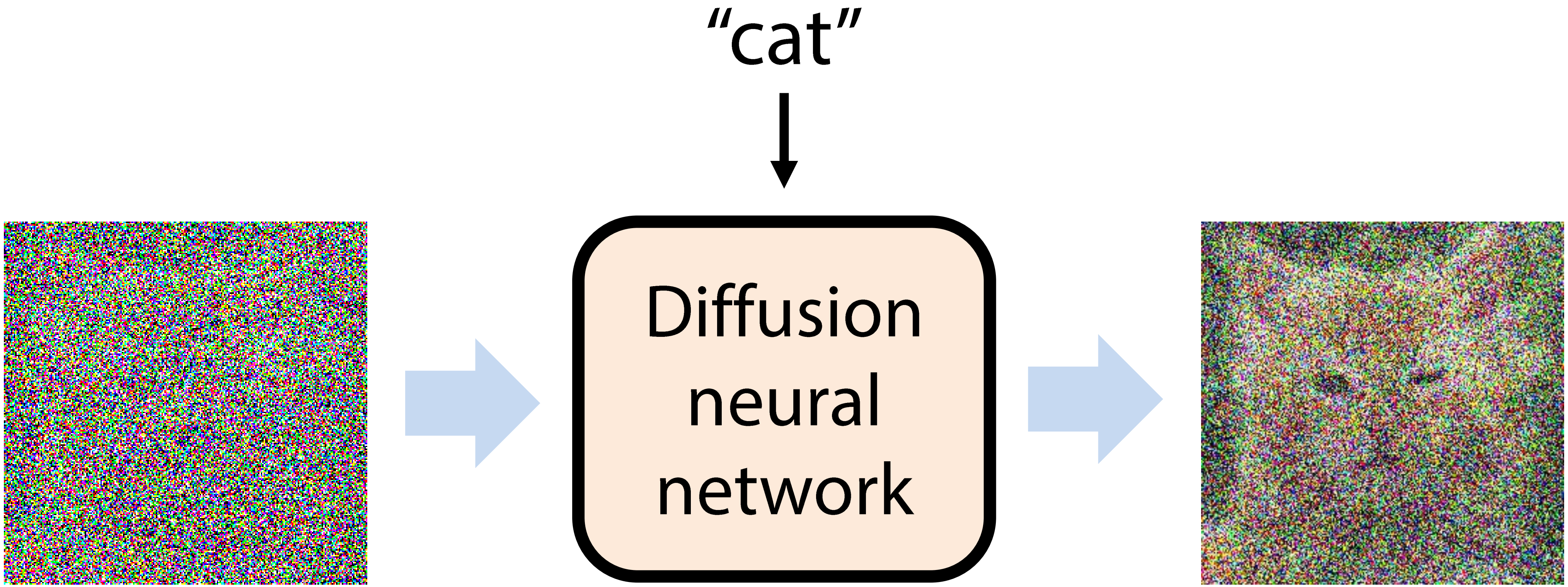
Generating new images

- Once diffusion network has been trained, generate new images by starting with a random noise image, and iteratively applying the network to slowly remove noise, for some number of steps (e.g., 1,000 for DALL-E 2)
- "Walking from random images towards the manifold of natural images"

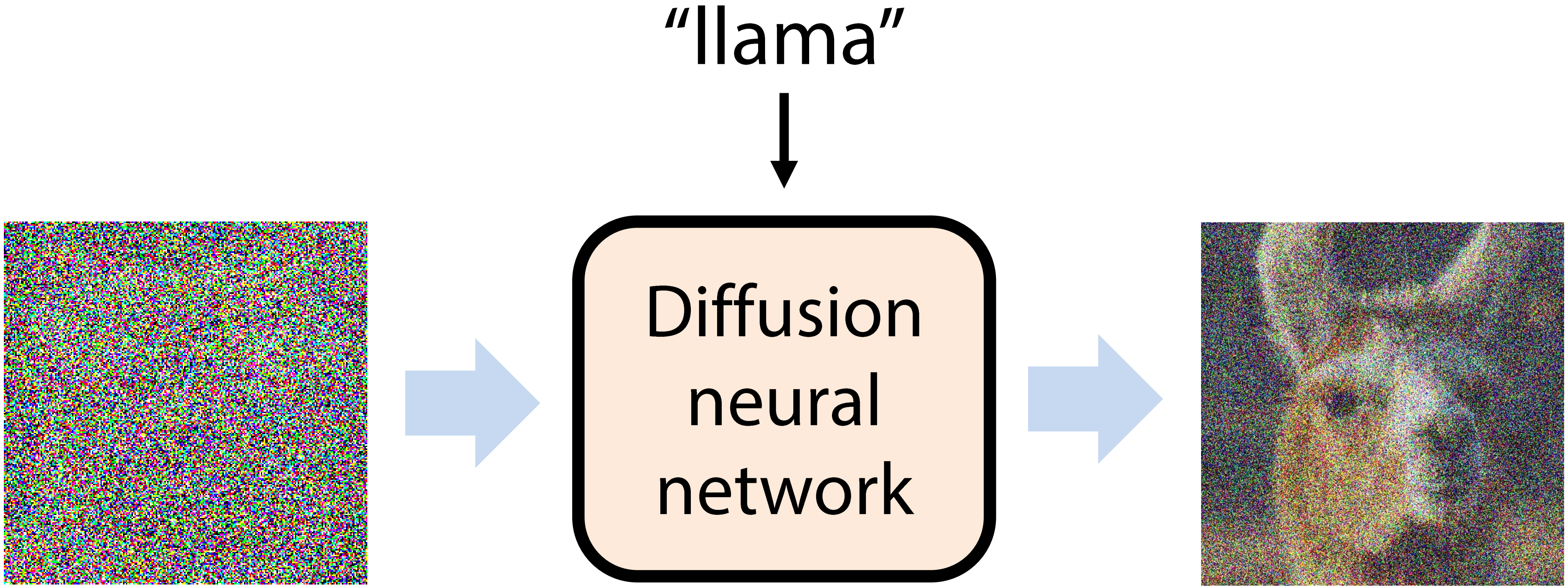


Slide concept: Steve Seitz

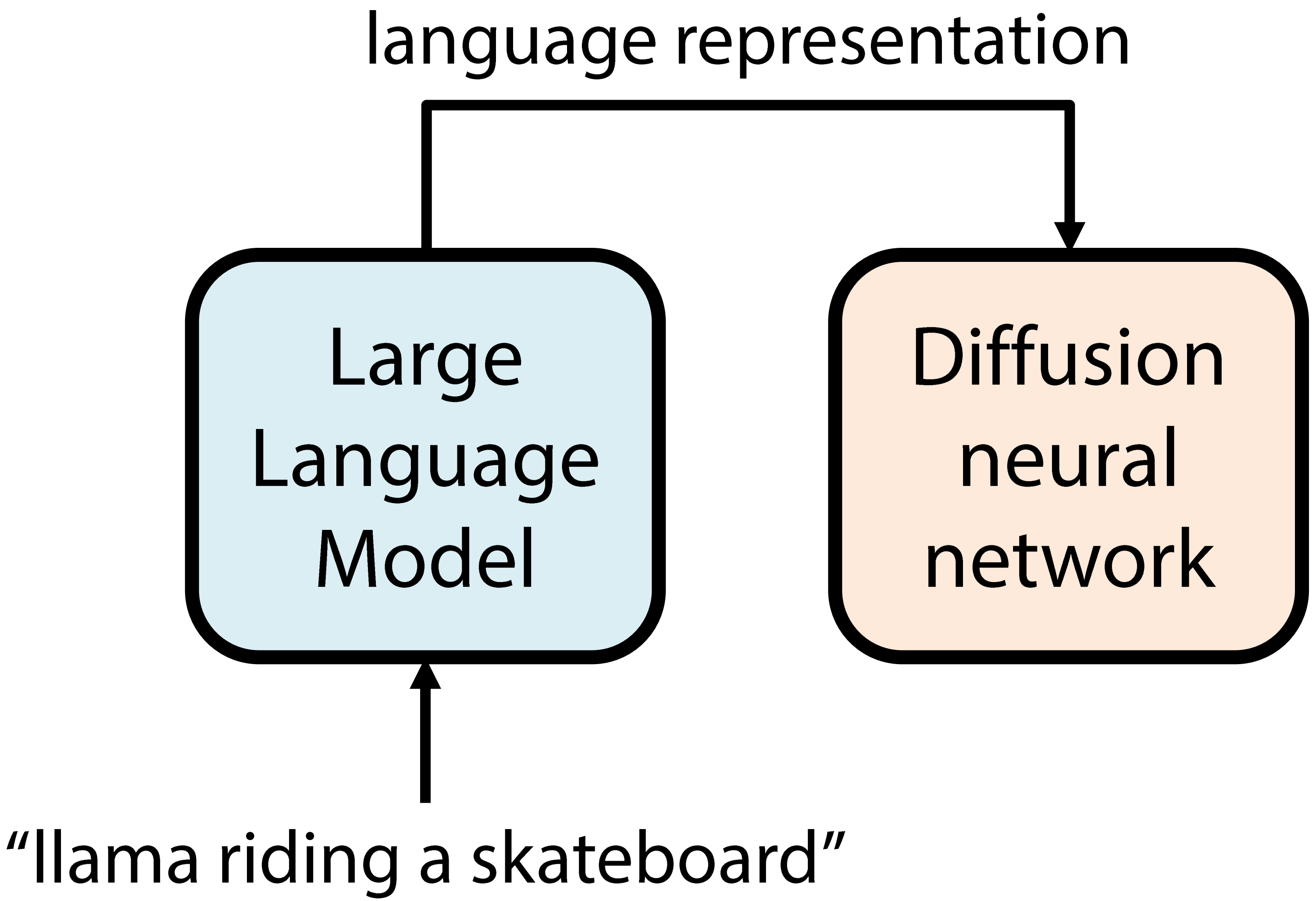
Idea 1: add a text label as conditioning



Idea 1: add a text label as conditioning



Idea 2: condition using large language model



Training on images + captions

A pack llama in the Rocky Mountain National Park

https://en.wikipedia.org/wiki/Llama

DALL-E 2

“A llama riding a skateboard”

“A llama riding a skateboard captured with a DSLR”

Imagen

"Sprouts in the shape of text 'Imagen' coming out of a fairytale book."

"A dragon fruit wearing karate belt in the snow."

Other applications of diffusion models

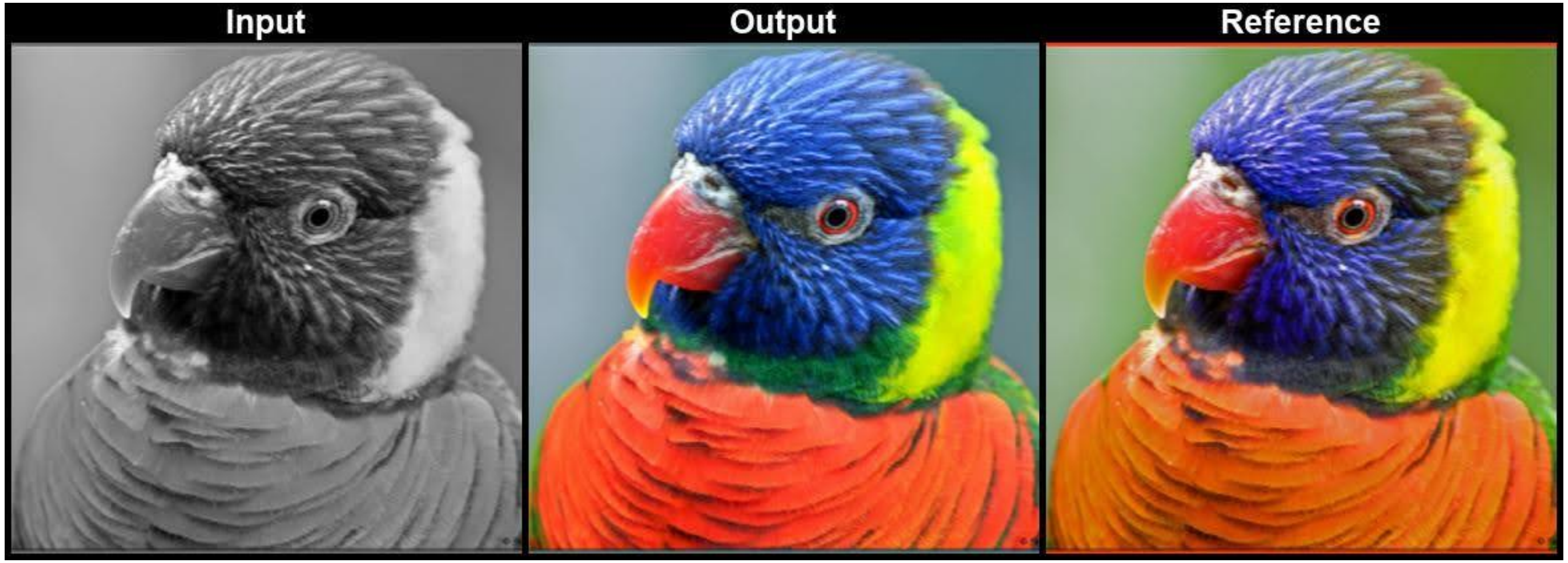
- Uncropping

Progressively zooming out. The most zoomed-in image is the input

[Palette: Image-to-Image Diffusion Models](#)
Saharia et al. arXiv 2022.

Other applications of diffusion models

- Colorization



[Palette: Image-to-Image Diffusion Models](#)
Saharia et al. arXiv 2022.

Other applications of diffusion models

- Inpainting

[Palette: Image-to-Image Diffusion Models](#)
Saharia et al. arXiv 2022.

DreamFusion: Text-to-3D using 2D Diffusion

“a DSLR photo of a squirrel”

https://dreamfusion3d.github.io/

DreamBooth: Fine Tuning Text-to-Image Diffusion Models for Subject-Driven Generation

Nataniel Ruiz Yuanzhen Li Varun Jampani Yael Pritch Michael Rubinstein Kfir Aberman

Google Research

It's like a photo booth, but once the subject is captured, it can be synthesized wherever your dreams take you…

[Paper] (new!) [Dataset] [BibTeX]

Personalized Residuals for Concept-Driven Text-to-Image Generation

Cusuh Ham, Matthew Fisher, James Hays,

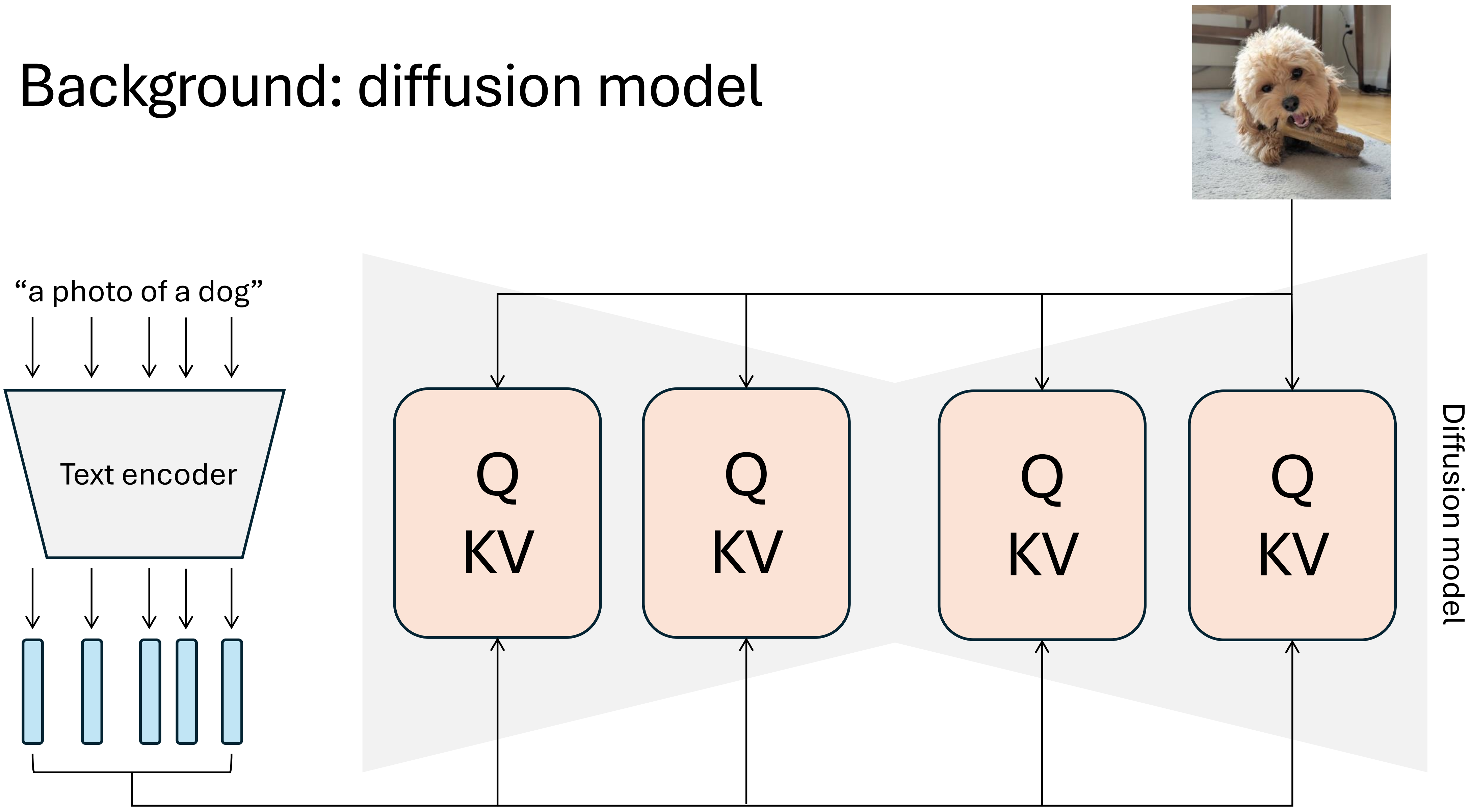
Nicholas Kolkin, Yuchen Liu, Richard Zhang, Tobias Hinz

CVPR 2024

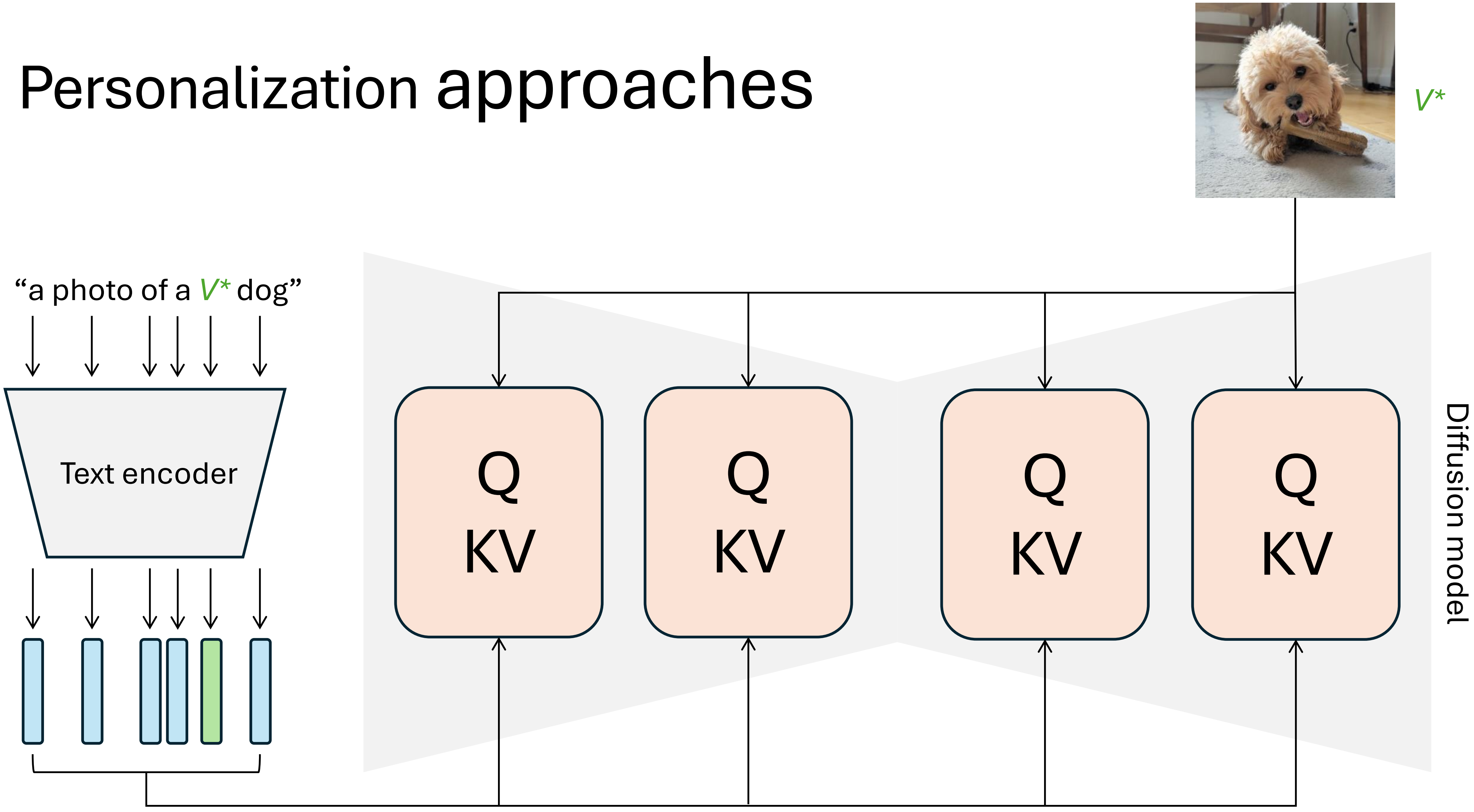
Motivation

DreamBooth: fine tuning text-to-image diffusion models for subject-driven generation. N. Ruiz, Y. Li, V. Jampani, Y. Pritch, M. Rubinstein, K. Aberman. CVPR 2023.

Background: diffusion model

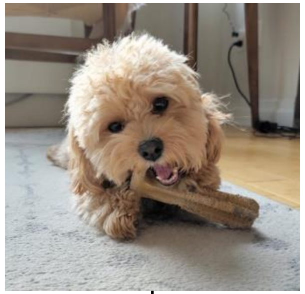


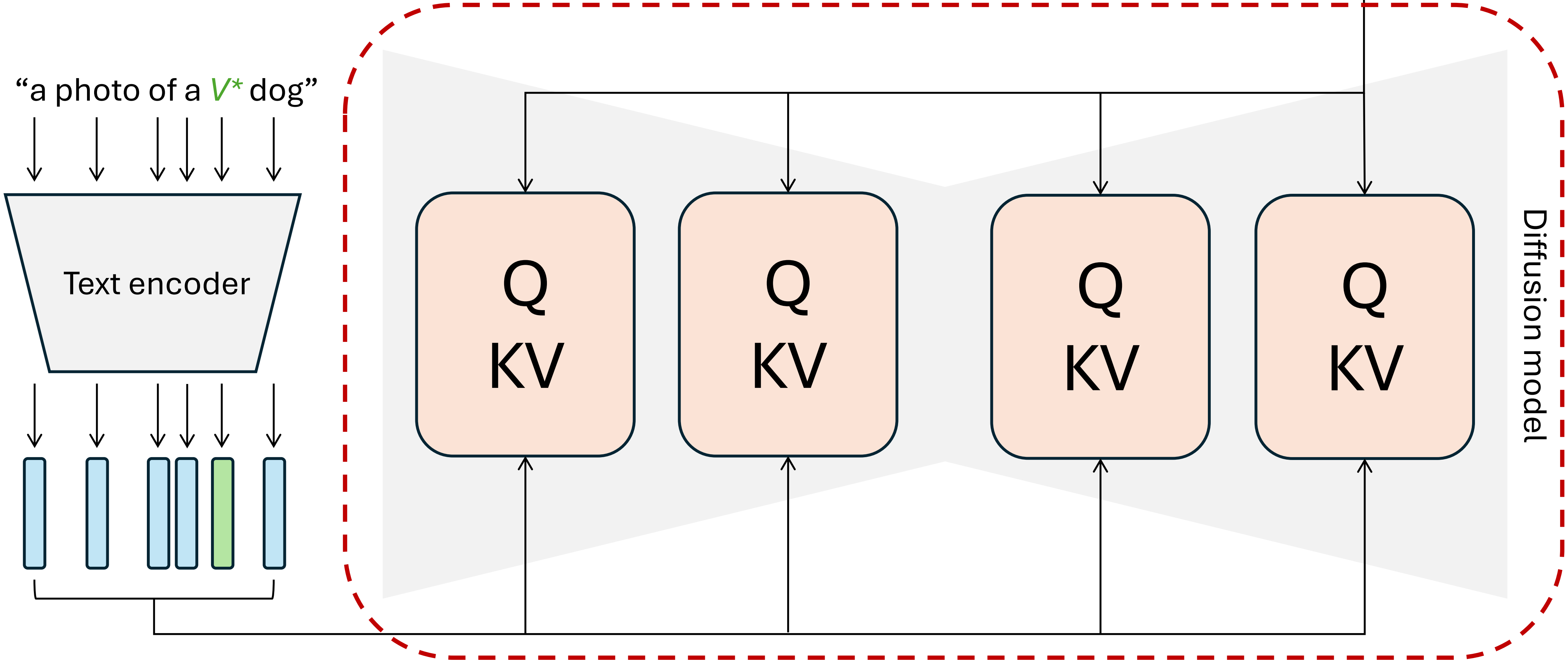
Personalization approaches



DreamBooth

- Large # parameters
- Requires regularization images to preserve learned prior

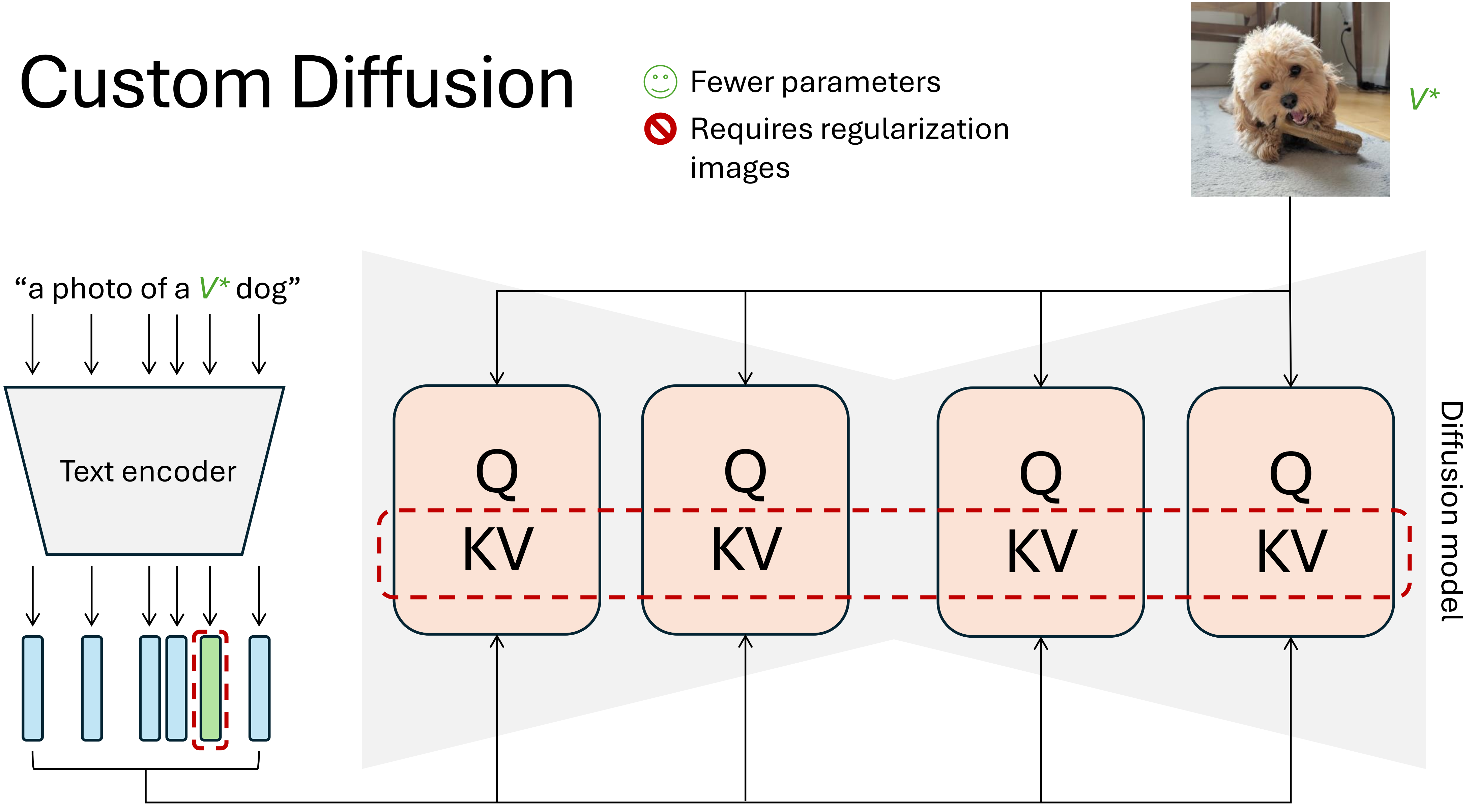




DreamBooth: fine tuning text-to-image diffusion models for subject-driven generation. N. Ruiz, Y. Li, V. Jampani, Y. Pritch, M. Rubinstein, K. Aberman. CVPR 2023.

Custom Diffusion

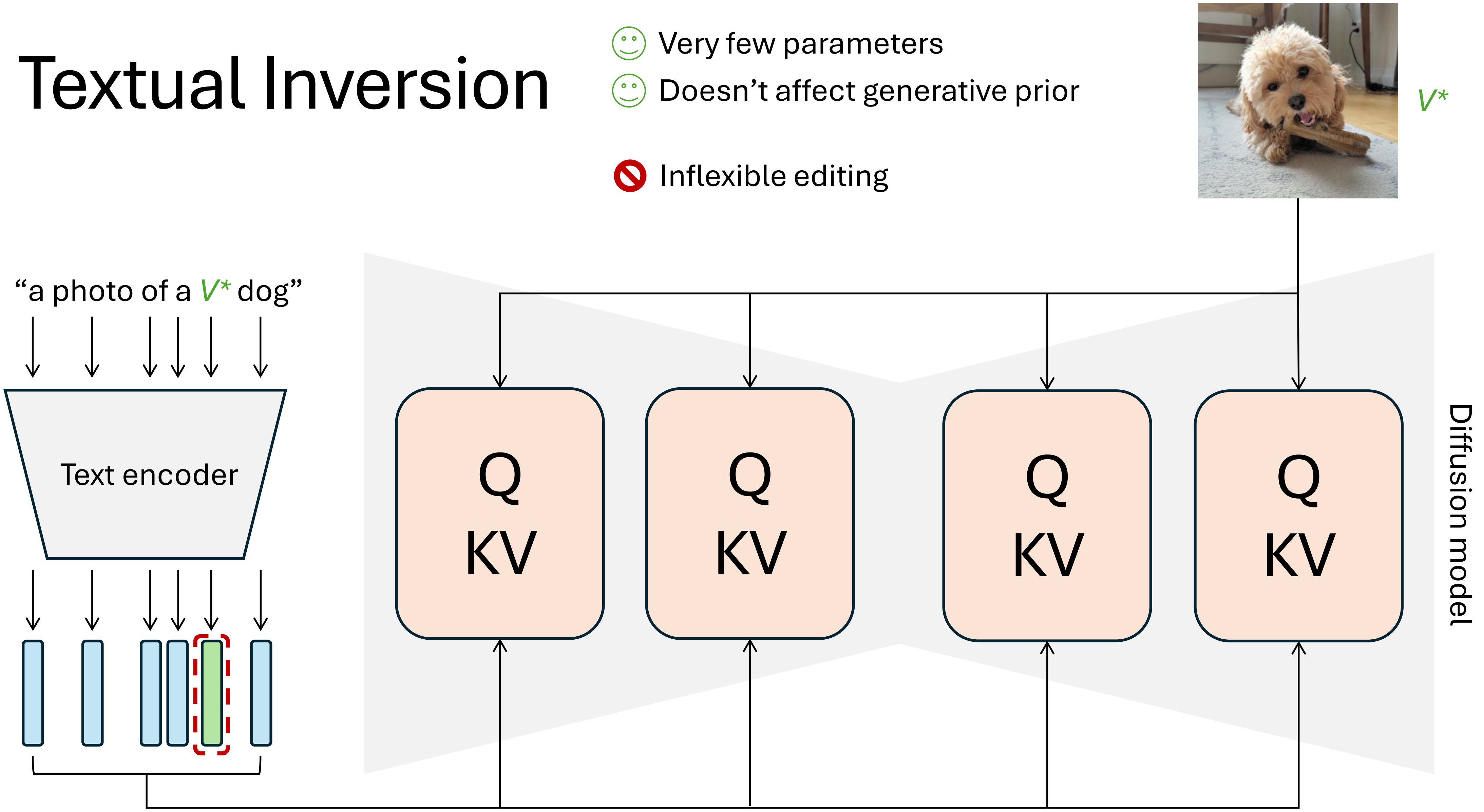
- 🙂 Fewer parameters
- 🚫 Requires regularization images



Multi-concept customization of text-to-image diffusion. N. Kumari, B. Zhang, R. Zhang, E. Shechtman, J. Zhu. CVPR 2023.

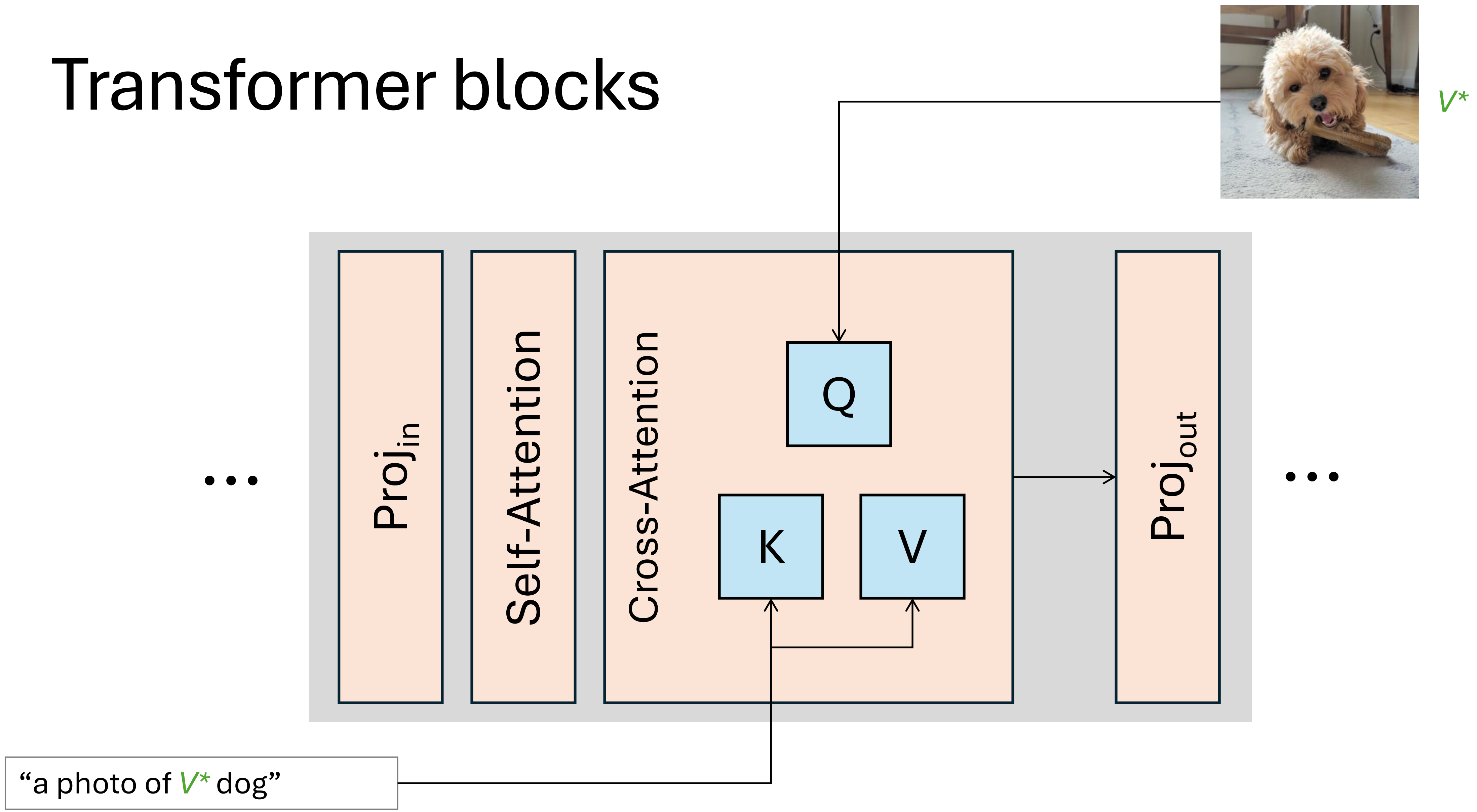
Textual Inversion

- Very few parameters
- Doesn't affect generative prior
- Inflexible editing

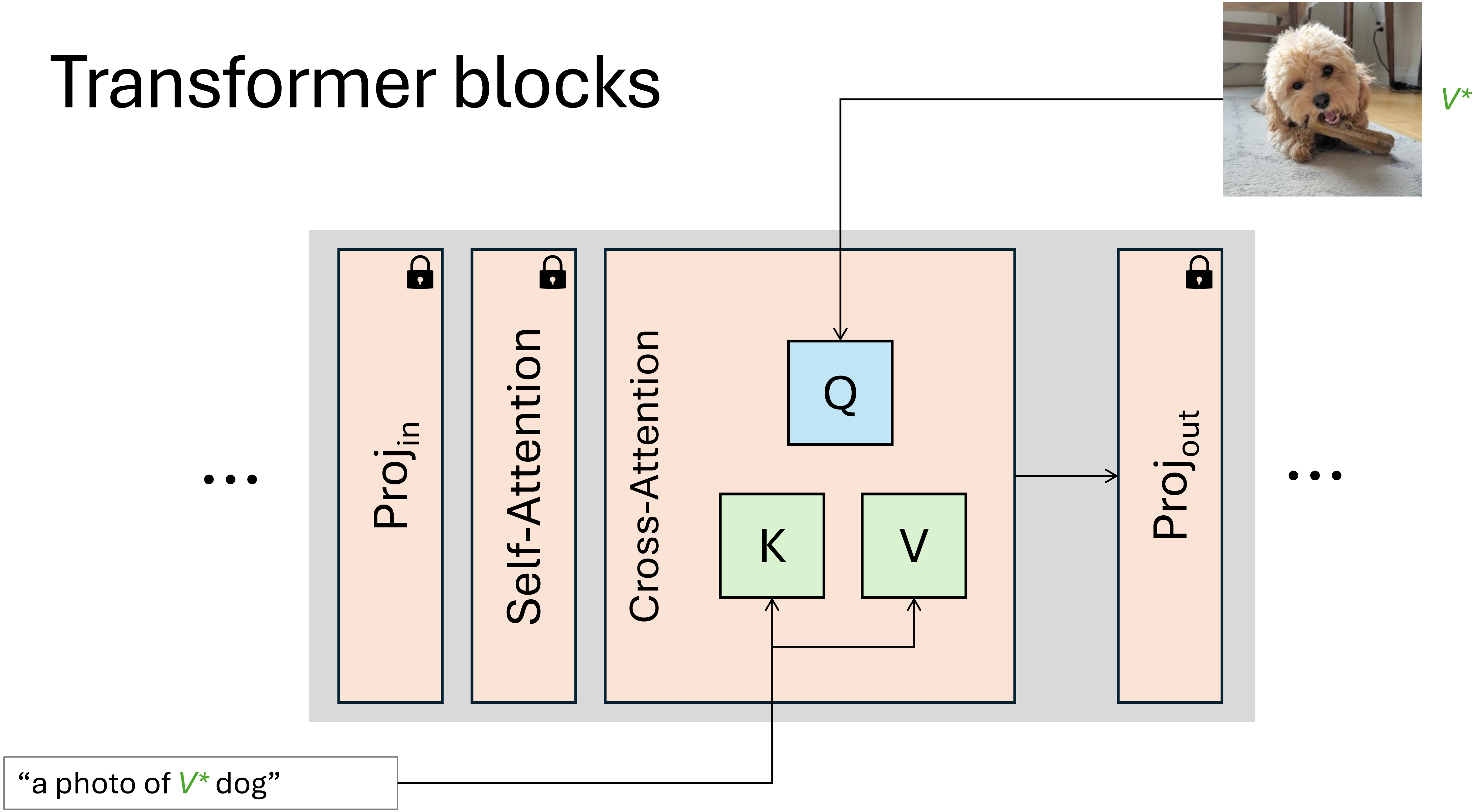


An image is worth one word: personalizing text-to-image generation using textual inversion. R. Gal, Y. Alaluf, Y. Atzmon, O. Patashnik, A. H. Bermano, G. Chechik, D. Cohen-Or. arXiv preprint 2022.

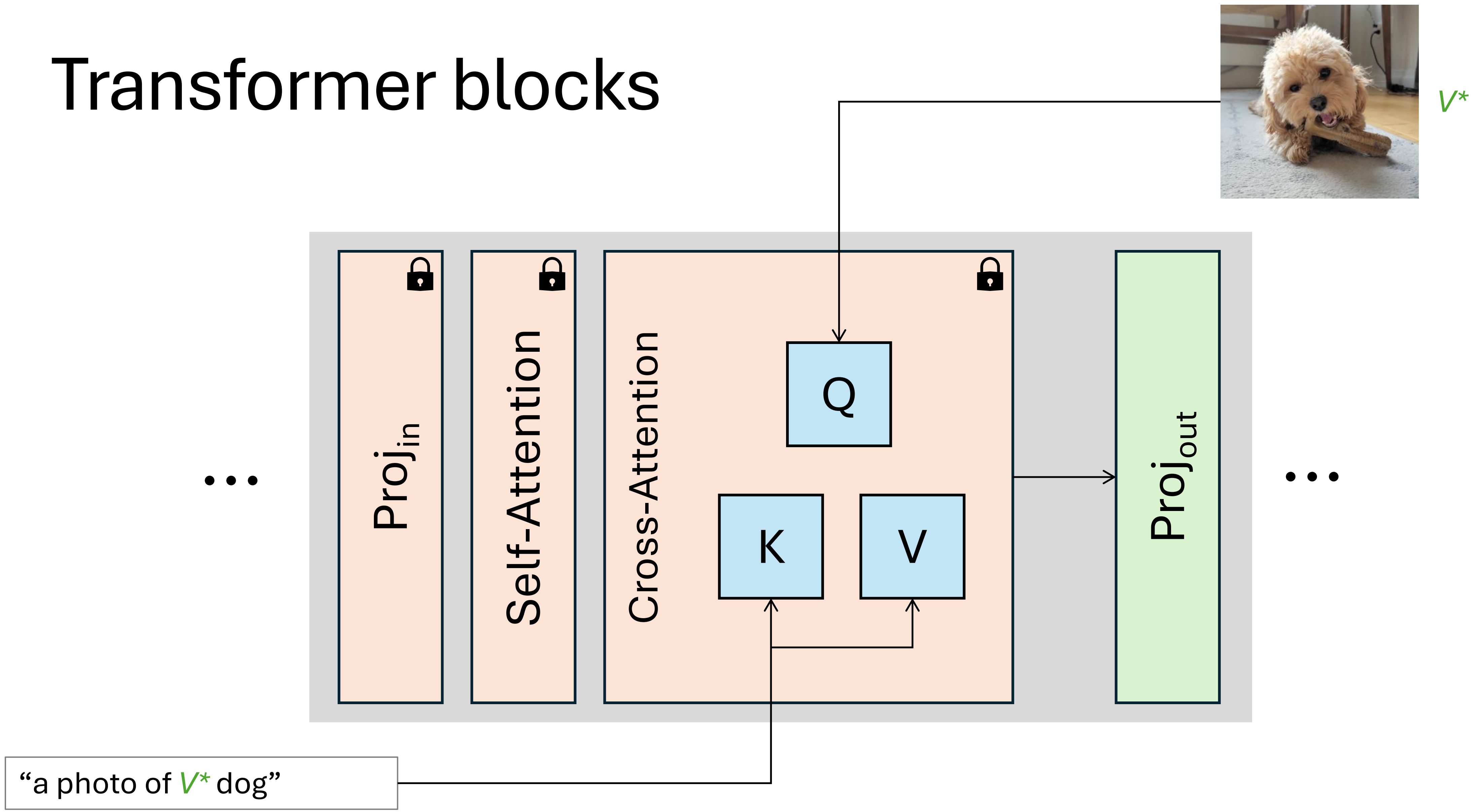
Transformer blocks



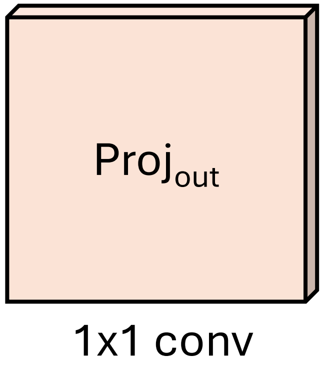
Transformer blocks

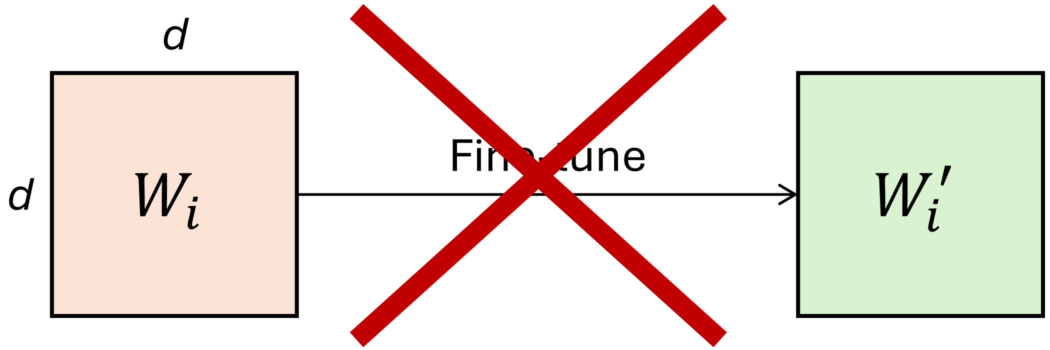


Transformer blocks



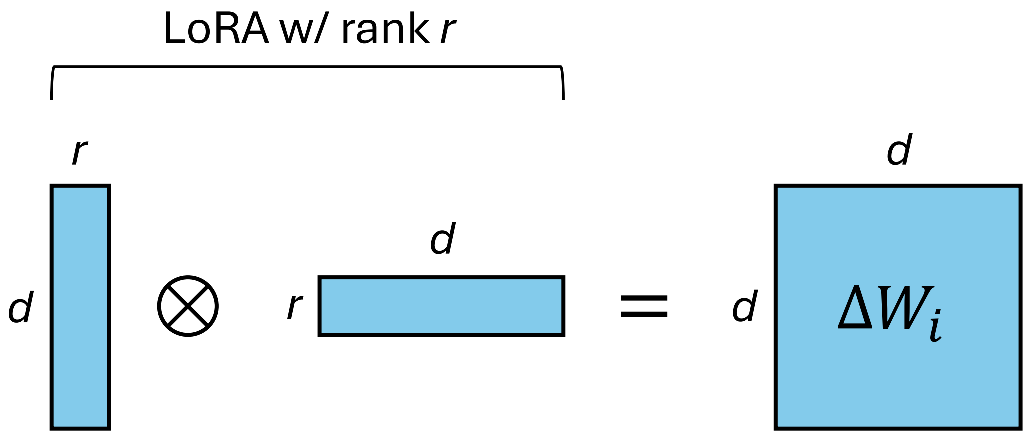
Our approach: personalized residuals





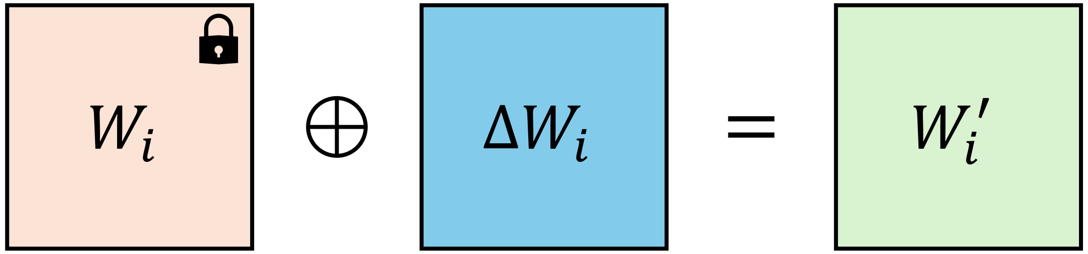
Overwrites learned prior

Our approach



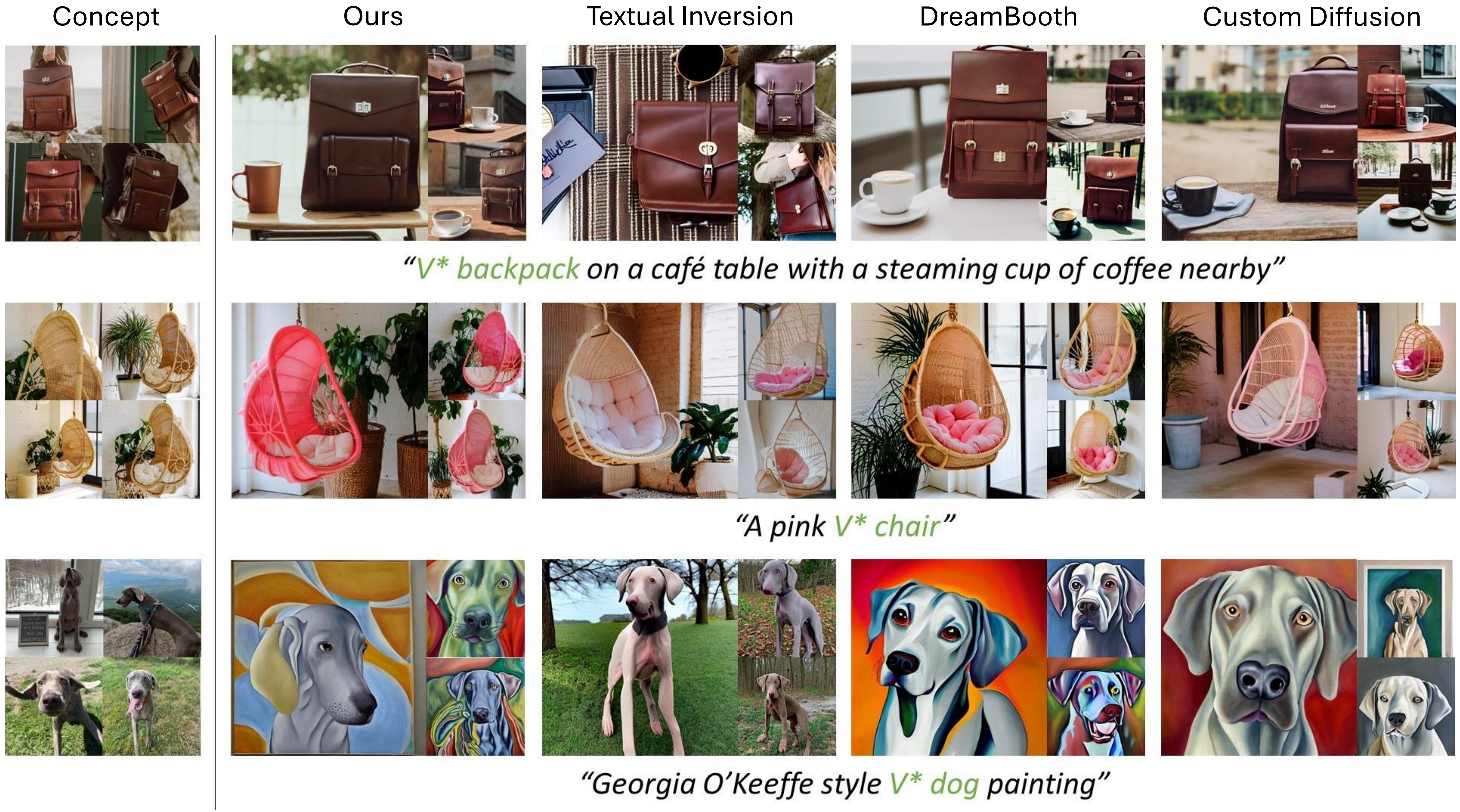
Method	Regularization images?	# parameters
Textual inversion	✗	768
DreamBooth	✓	983M
Custom Diffusion	✓	19M
Ours	✗	1.2M

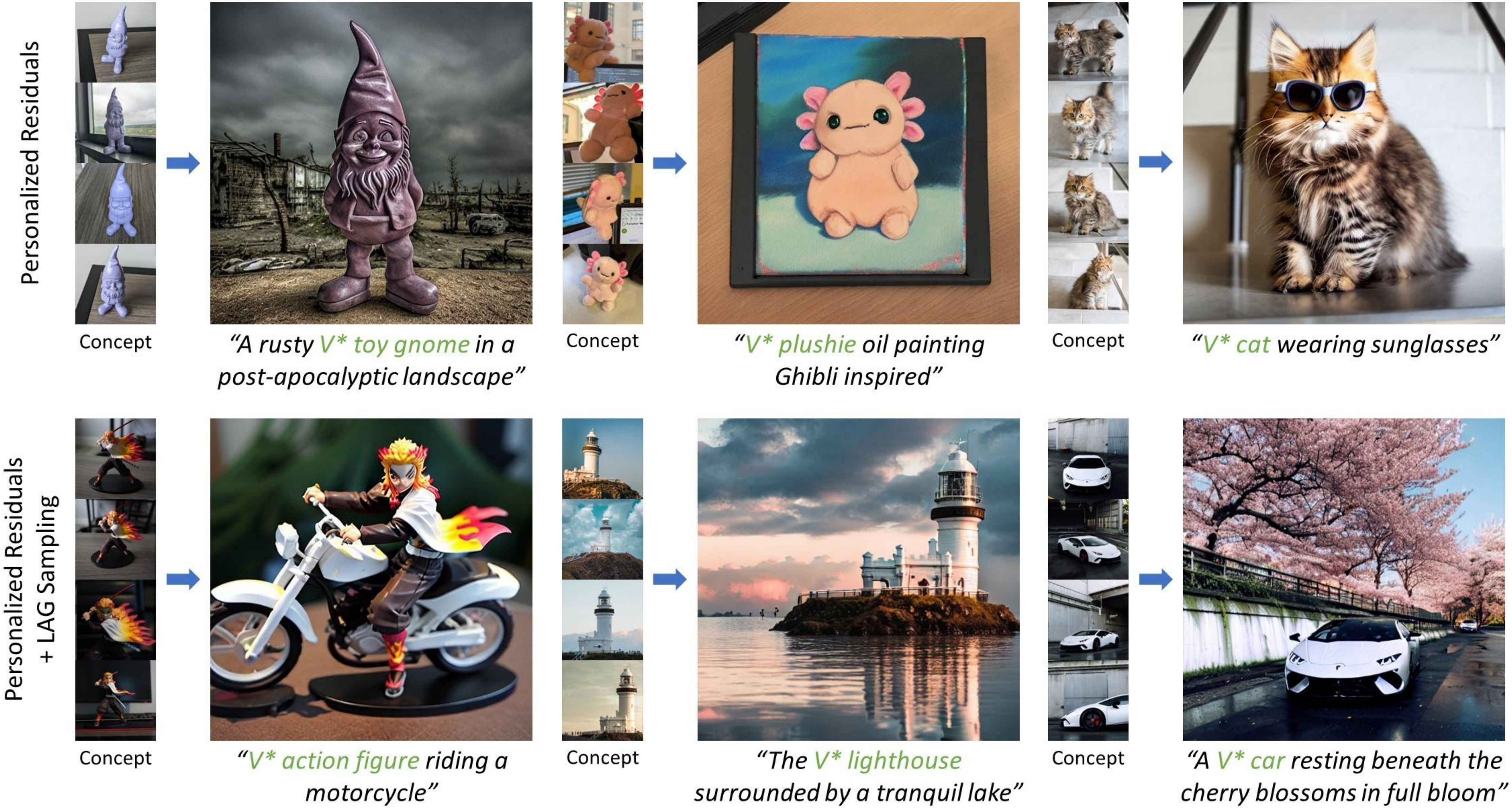
150 iterations
~3 min on 1 A100



Personalized residual

LoRA: low-rank adaptation of large language models. E. J. Hu, Y. Shen, P. Wallis, Z. Allen-Zhu, Y. Li, S. Wang, L. Wang, W. Chen. ICLR 2022.


Concept
Ours
Textual Inversion
DreamBooth
Custom Diffusion
“V* backpack on a café table with a steaming cup of coffee nearby”
“A pink V* chair”
“Georgia O’Keeffe style V* dog painting”



Visit the poster tomorrow night 5-6:30pm in Arch 4A-E poster #329

Comparison with GANs

- Diffusion models tend to be easier to train and more scalable
- Diffusion models tend to be slower – often many iterations of denoising are required
- However, recent work is mitigating some of these issues (with both GANs and diffusion models)