Генерация и обработка изображений с помощью нейросетей

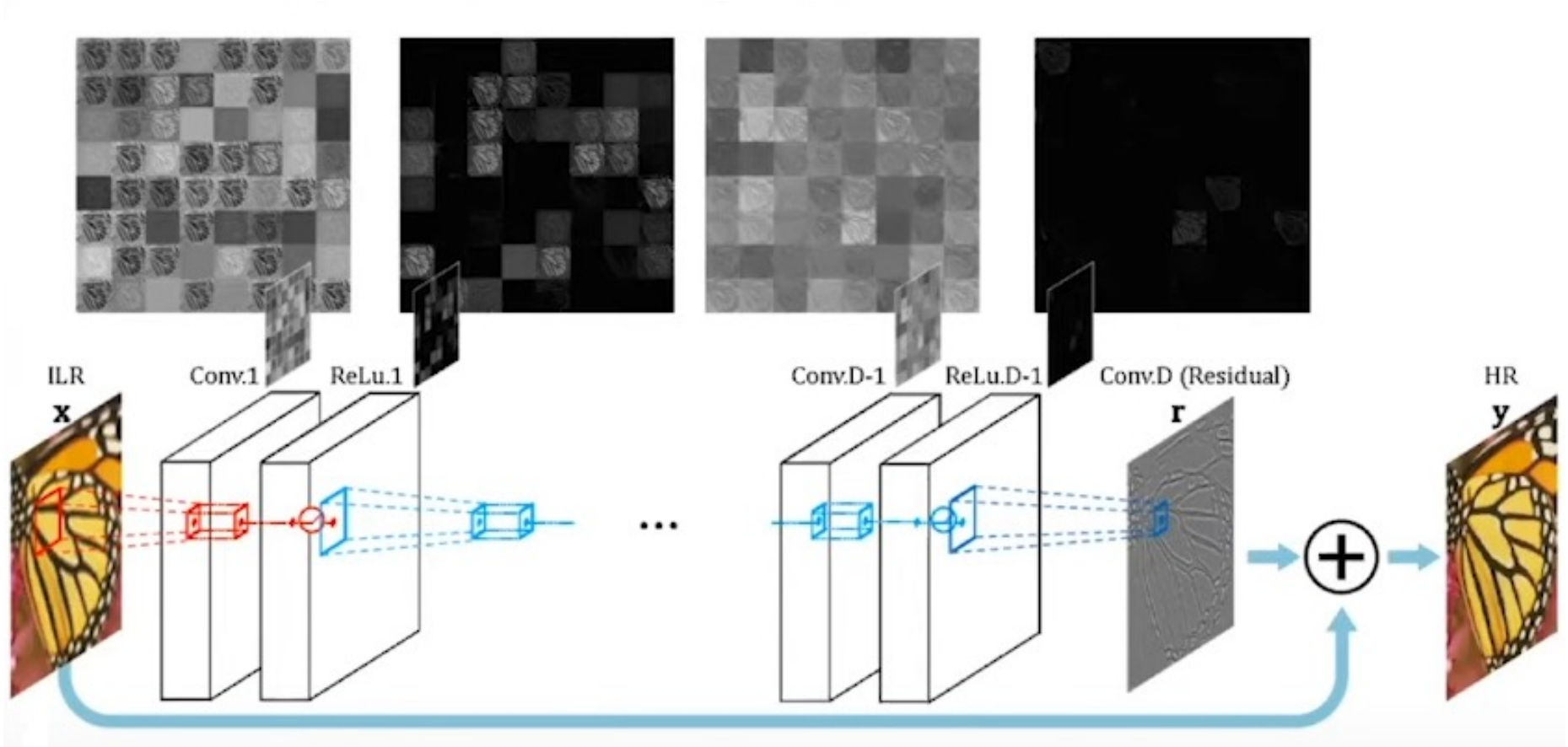
Кафедра технологий проектирования сложных технических систем

2023

Superresolution

- CovNet, у которого вход – x, выход – y
- Есть возможность легко получить данные для обучения

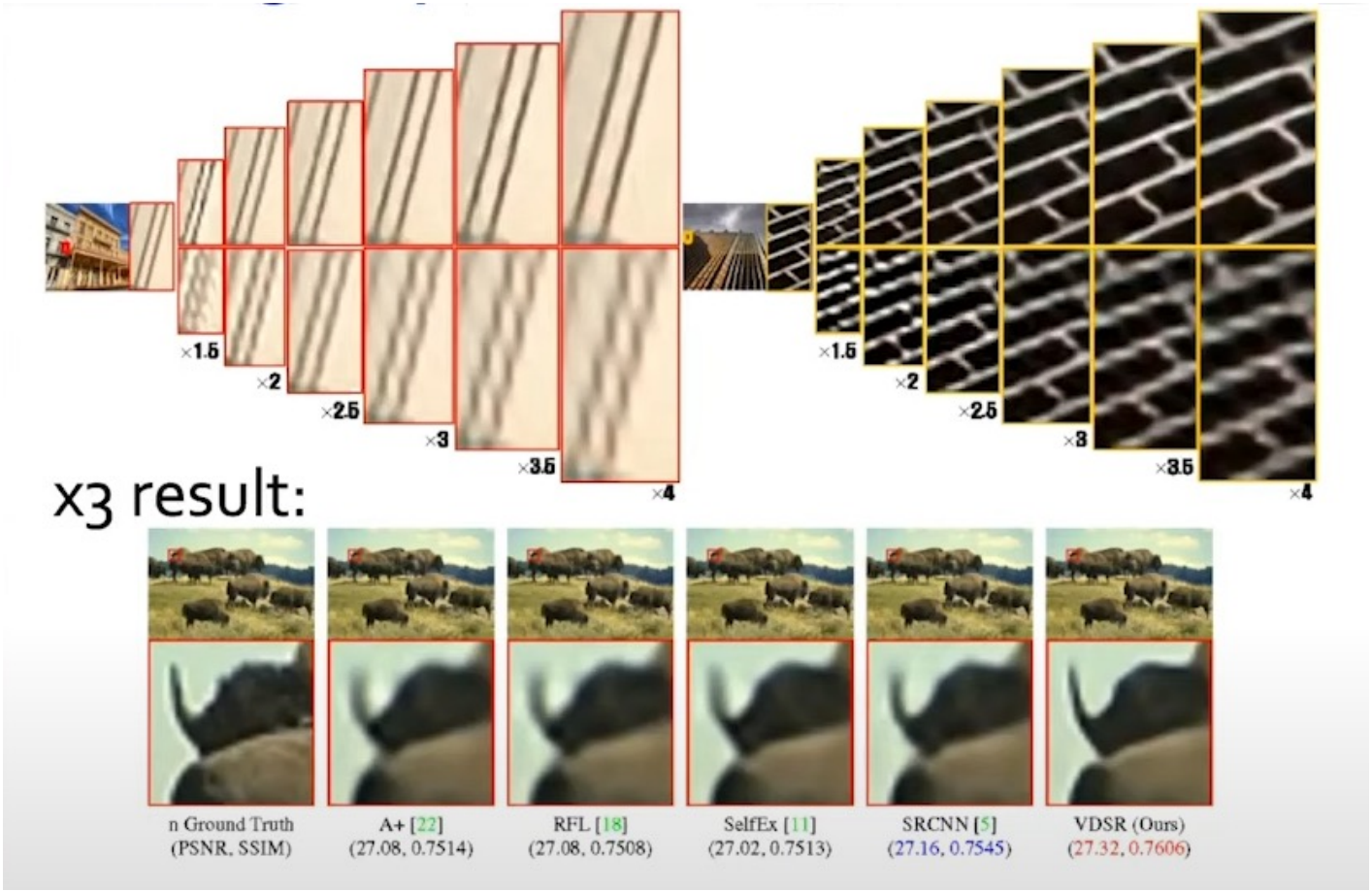
Image superresolution



$$\hat{\theta} = \arg\min_{\theta} \sum_{i} \|f(x_i;\theta) - y_i\|^2$$

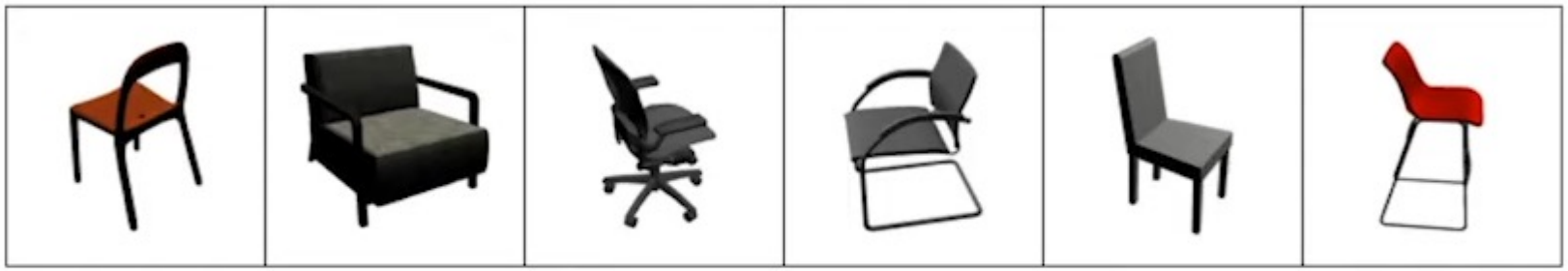
[Kim, Lee, CVPR16]

Image superresolution



[Kim, Lee, CVPR16]

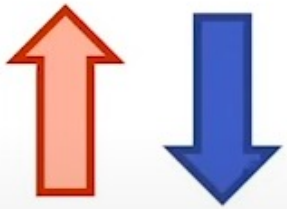
Neural rendering



x_i = chair ID and camera parameters

x_i = chair ID and camera parameters

Recognition
(*computer vision*)

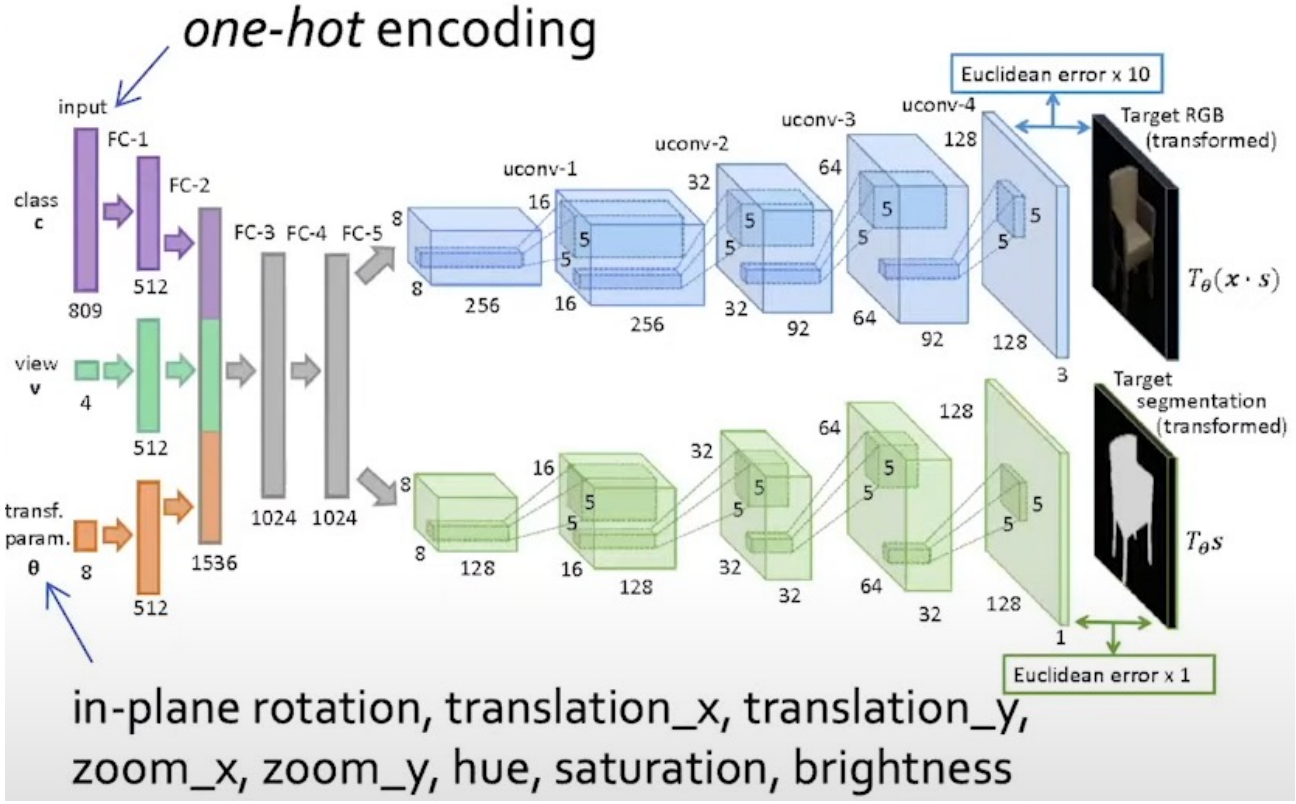


Rendering
(*computer graphics*)

y_i = RGB image + Binary foreground mask

[Dosovitskiy, Springerber, CVPR15]

Neural rendering



x_i = chair ID and camera parameters

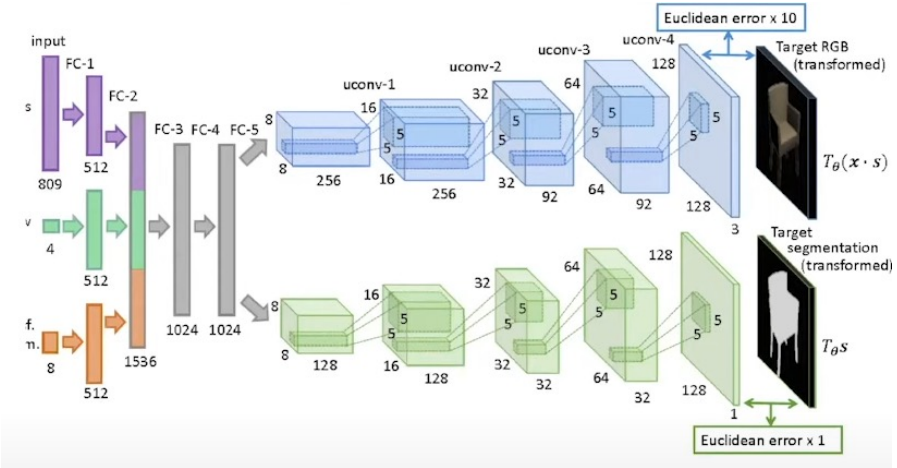
y_i = RGB image + Binary foreground mask

Training with MSE loss:

$$\hat{\theta} = \arg\min_{\theta} \sum \|f(x_i;\theta) - y_i\|^2$$

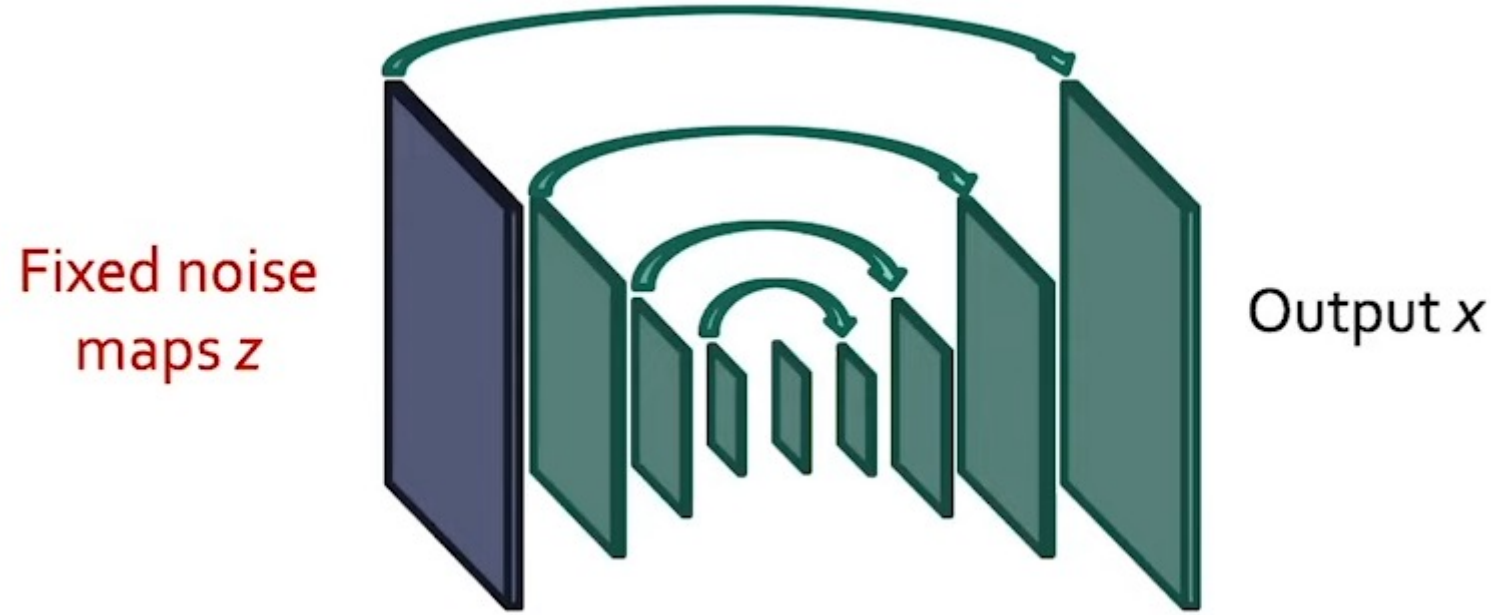
[Dosovitskiy, Springerber, CVPR15]

Neural rendering



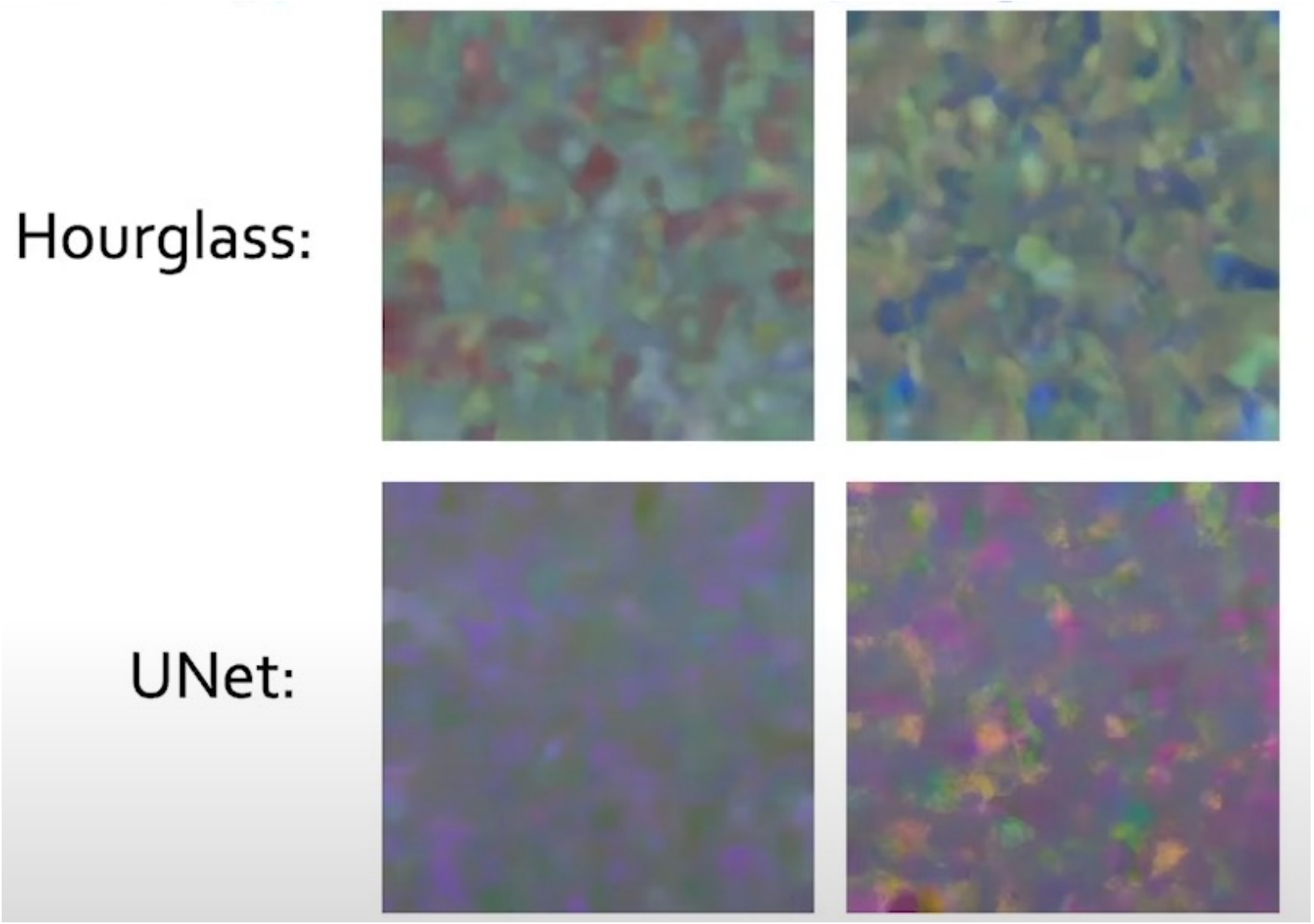
[Dosovitskiy, Springerber, CVPR15]

Почему это может работать?



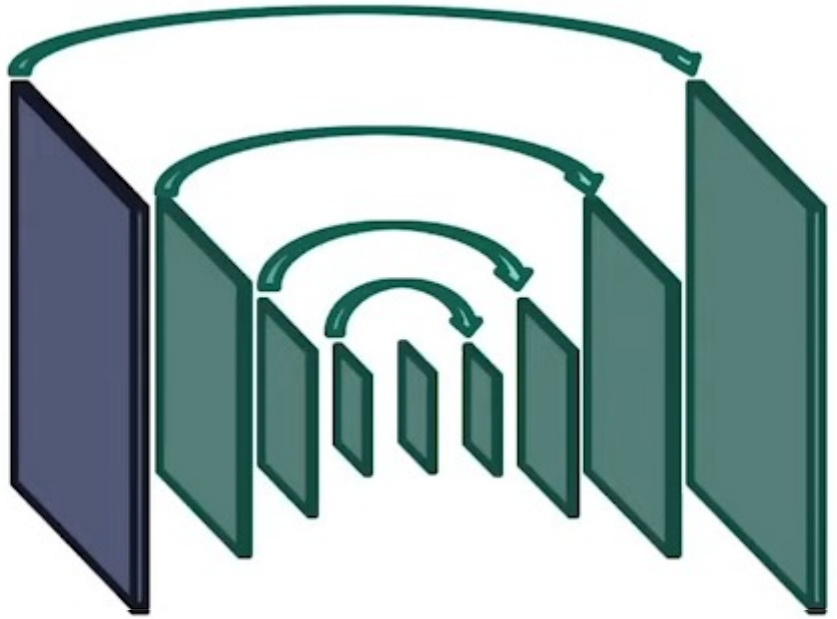
[ШАД, 2021]

Почему это может работать?



[ШАД, 2021]

Deep inpainting



- U-net with thin skip connections (containing convolution blocks)
- Convolution block: conv + batch norm + leaky ReLU + (up/down-sample)
- ~2 millions of parameters
- Fit to known pixel values

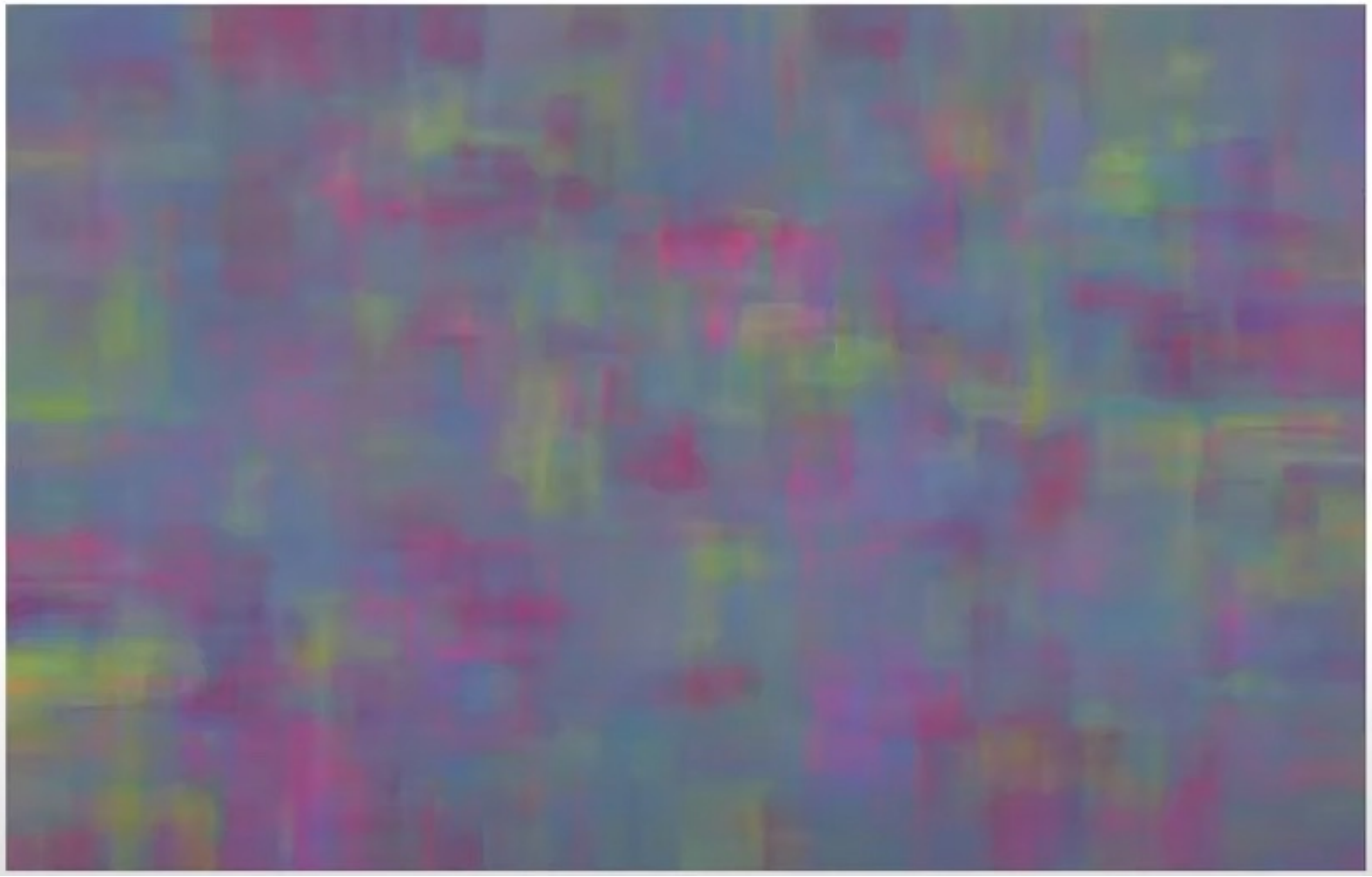
$$\min_{\theta} \|(f(z;\theta) - x_0) \odot m\|^2$$

[ШАД, 2021]

Deep inpainting

[Ulyanov, et.al CVPR2018]

Deep inpainting



[Ulyanov, et.al CVPR2018]

Deep inpainting

[Ulyanov, et.al CVPR2018]

Deep inpainting

[Ulyanov, et.al CVPR2018]

Deep inpainting

[Ulyanov, et.al CVPR2018]

Deep inpainting

[Ulyanov, et.al CVPR2018]

Deep inpainting

[Ulyanov, et.al CVPR2018]

Deep inpainting

[Ulyanov, et.al CVPR2018]

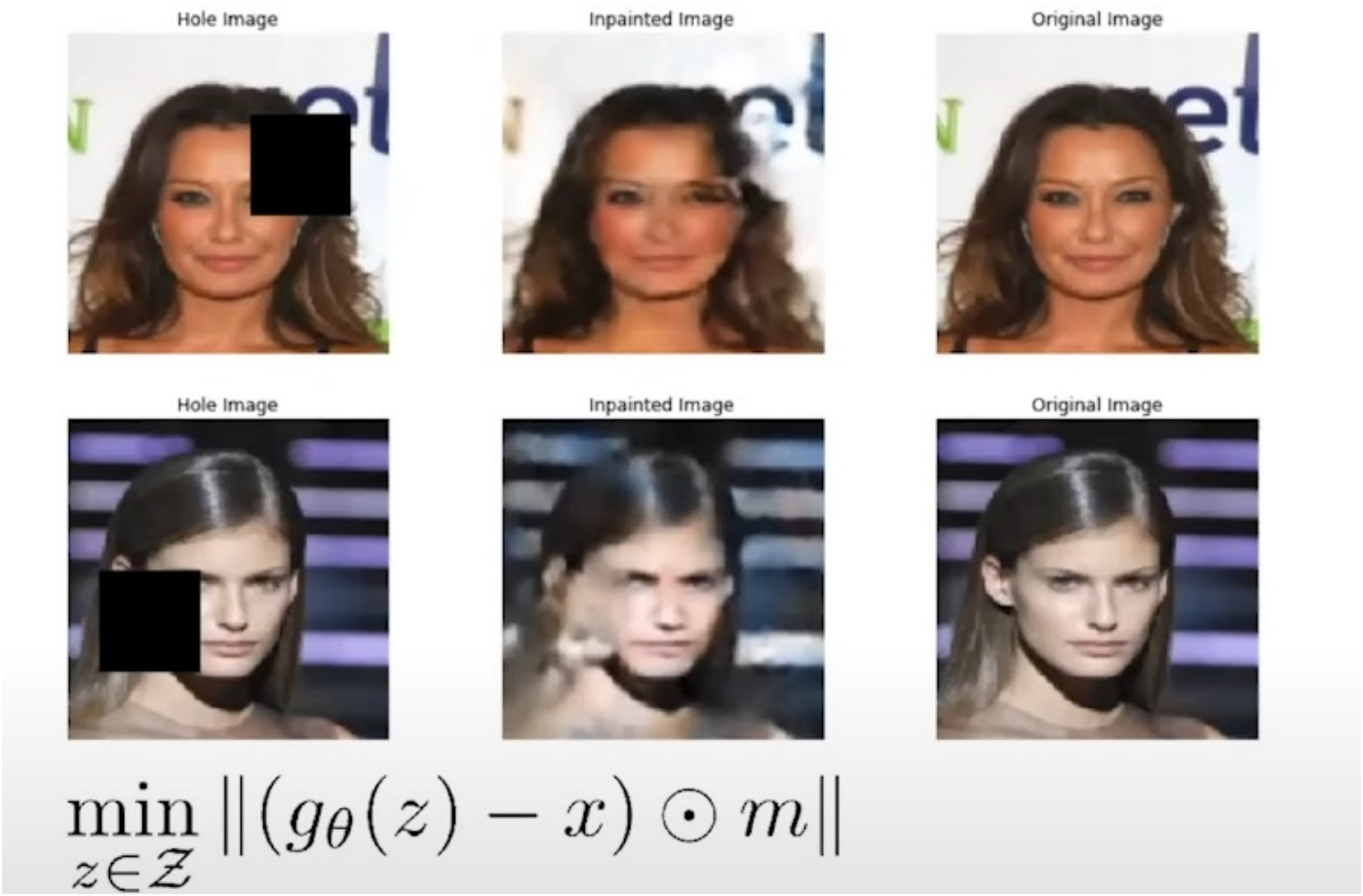
Deep inpainting

[Ulyanov, et.al CVPR2018]

Deep inpainting

[Ulyanov, et.al CVPR2018]

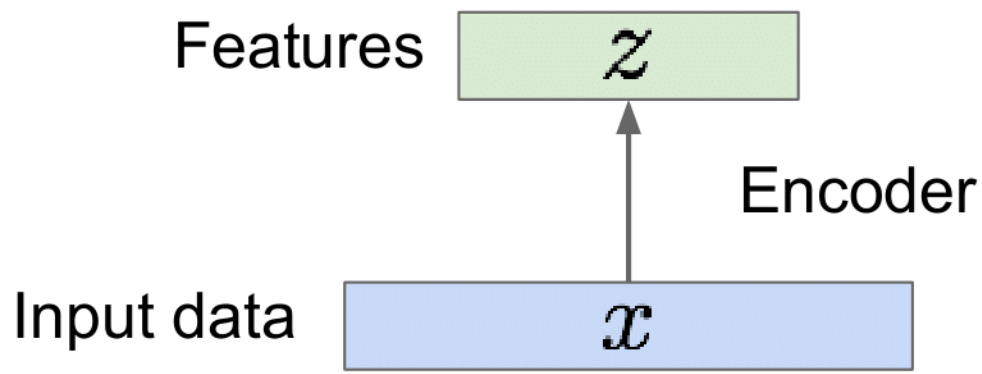
Deep inpainting

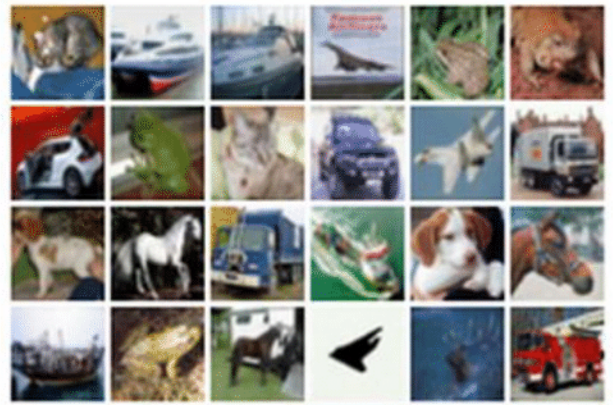


[ШАДБ 2021]

Some background first: Autoencoders

Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data



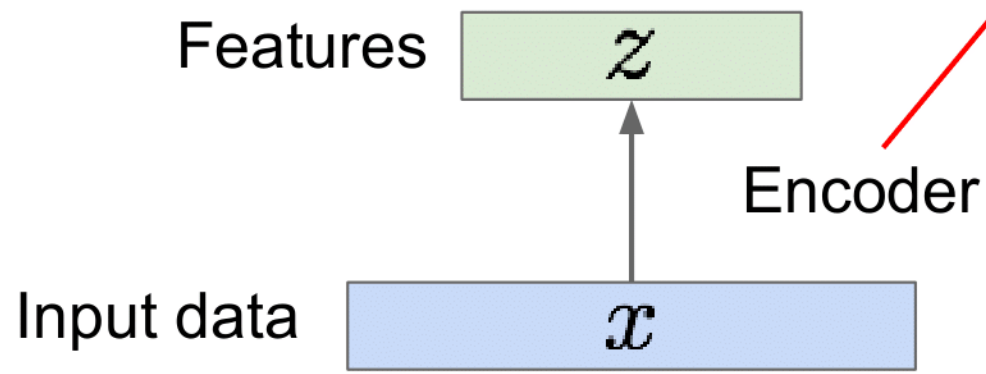


Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

Some background first: Autoencoders

Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data

Originally: Linear + nonlinearity (sigmoid)
Later: Deep, fully-connected
Later: ReLU CNN



Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

Some background first: Autoencoders

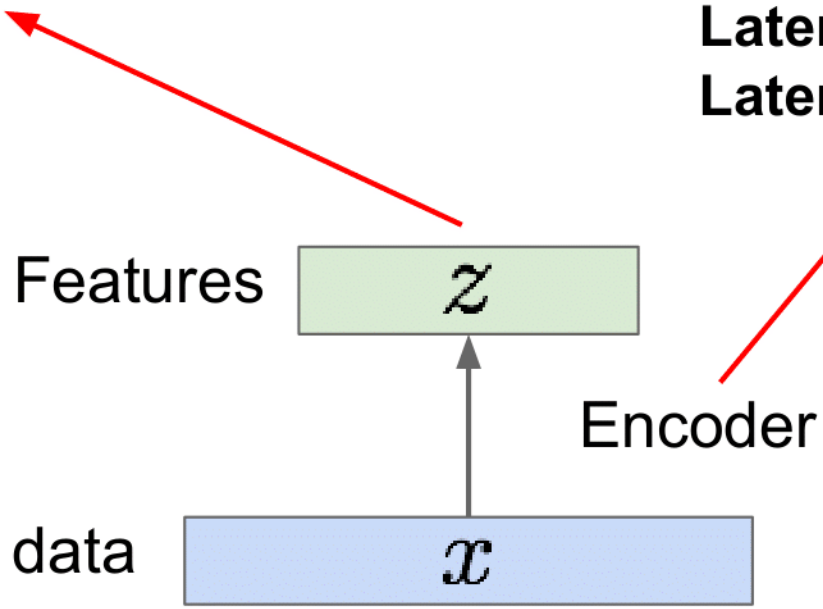
Unsupervised approach for learning a lower-dimensional feature representation from unlabeled training data

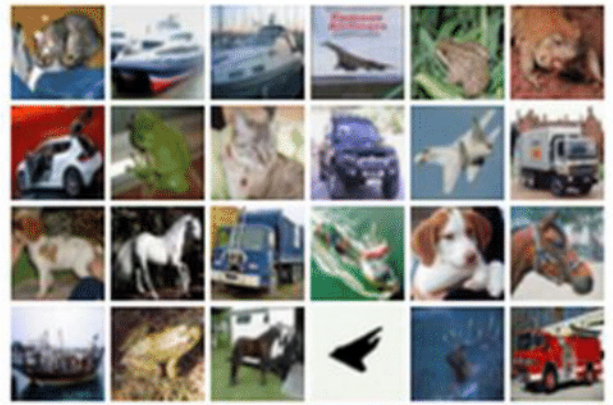
z usually smaller than **x** (dimensionality reduction)

Q: Why dimensionality reduction?

A: Want features to capture meaningful factors of variation in data

Originally: Linear + nonlinearity (sigmoid)
Later: Deep, fully-connected
Later: ReLU CNN

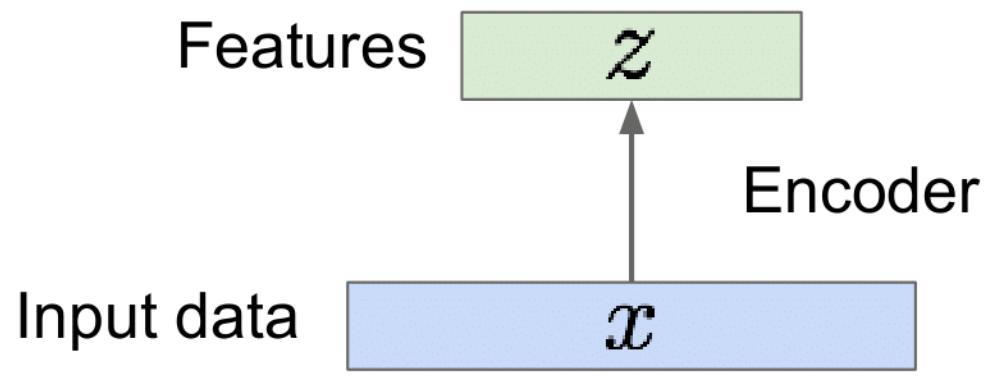




Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

Some background first: Autoencoders

How to learn this feature representation?



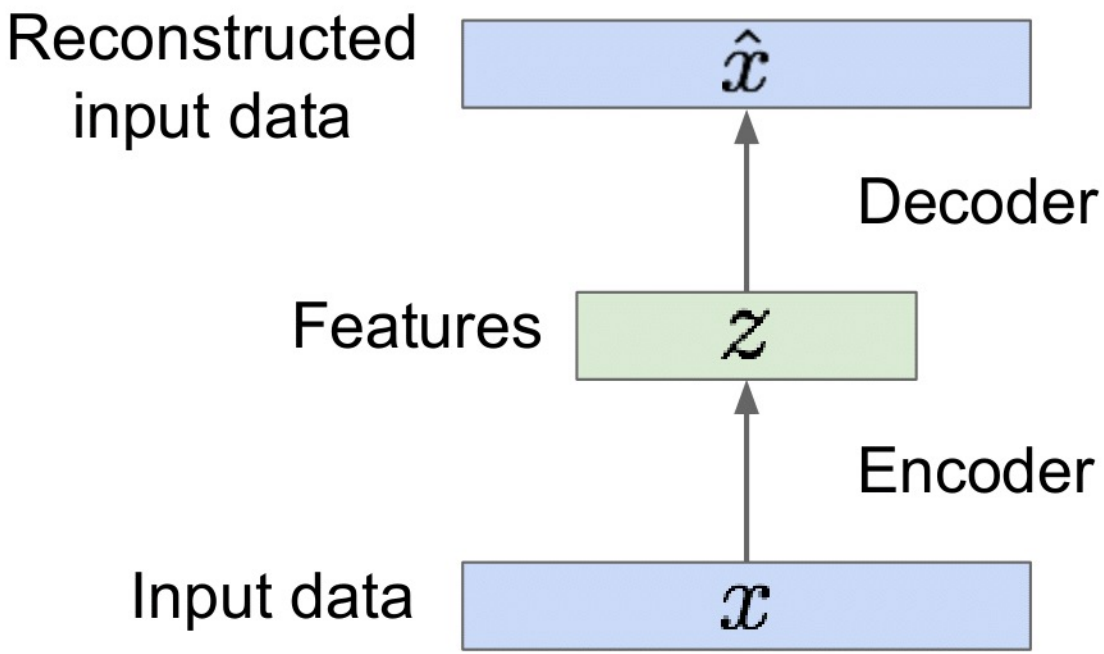
Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

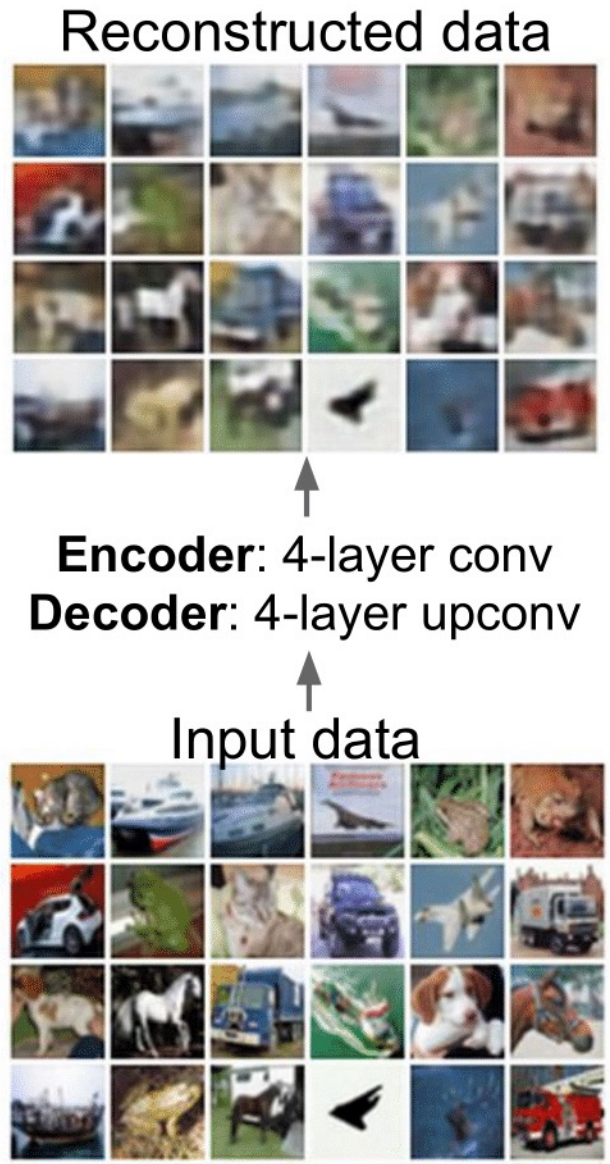
Some background first: Autoencoders

How to learn this feature representation?

Train such that features can be used to reconstruct original data

"Autoencoding" - encoding itself

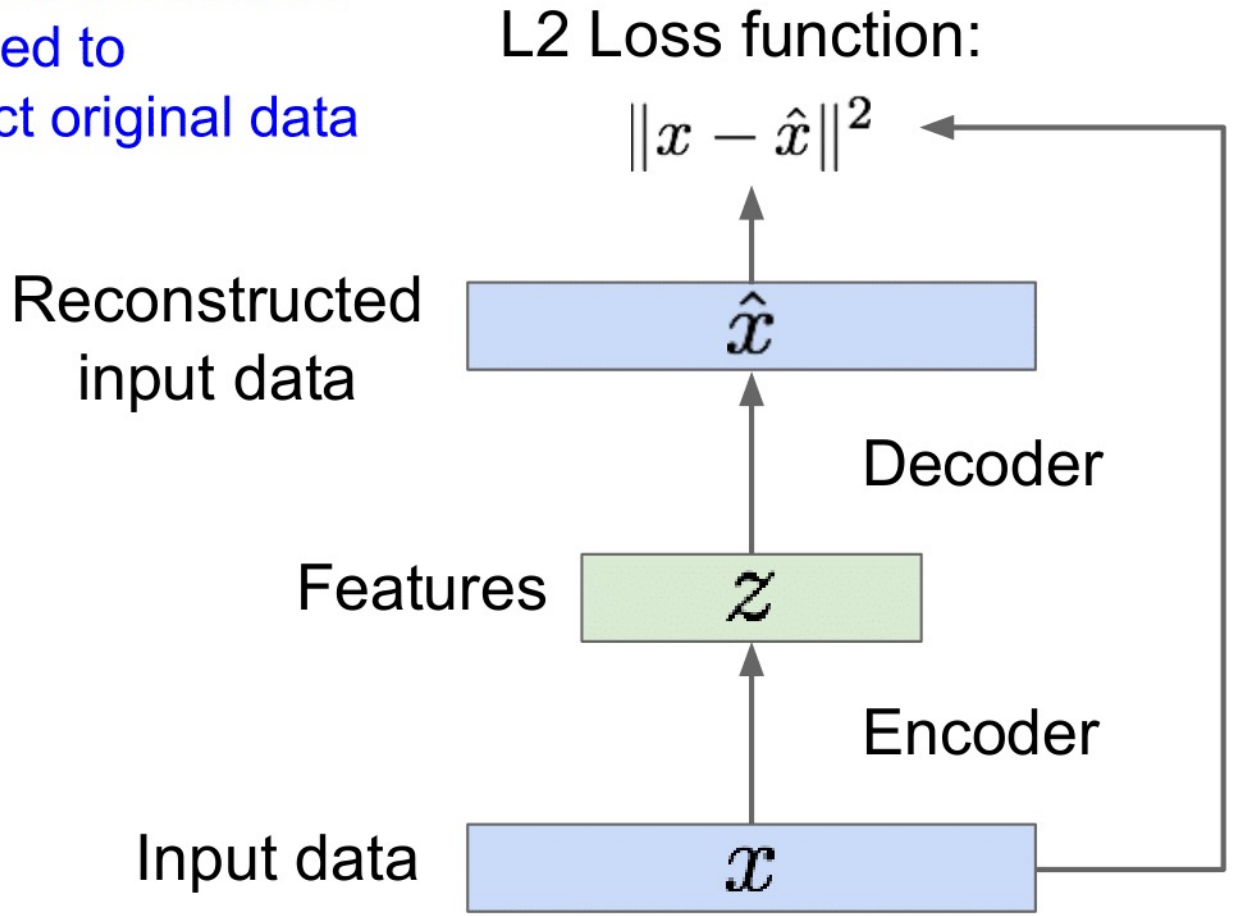


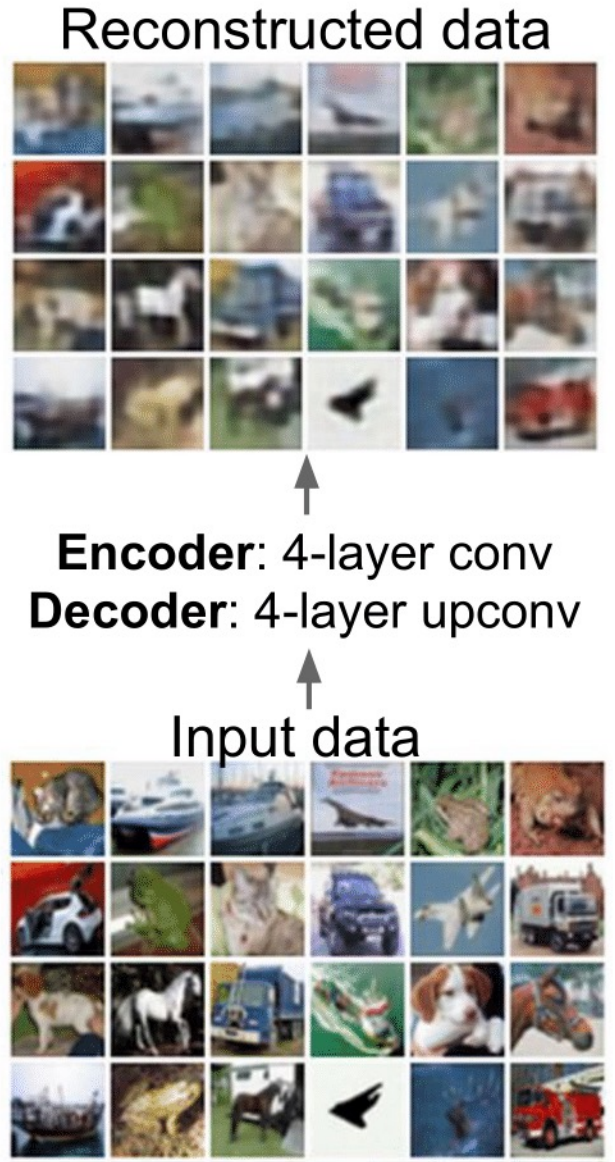


Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

Some background first: Autoencoders

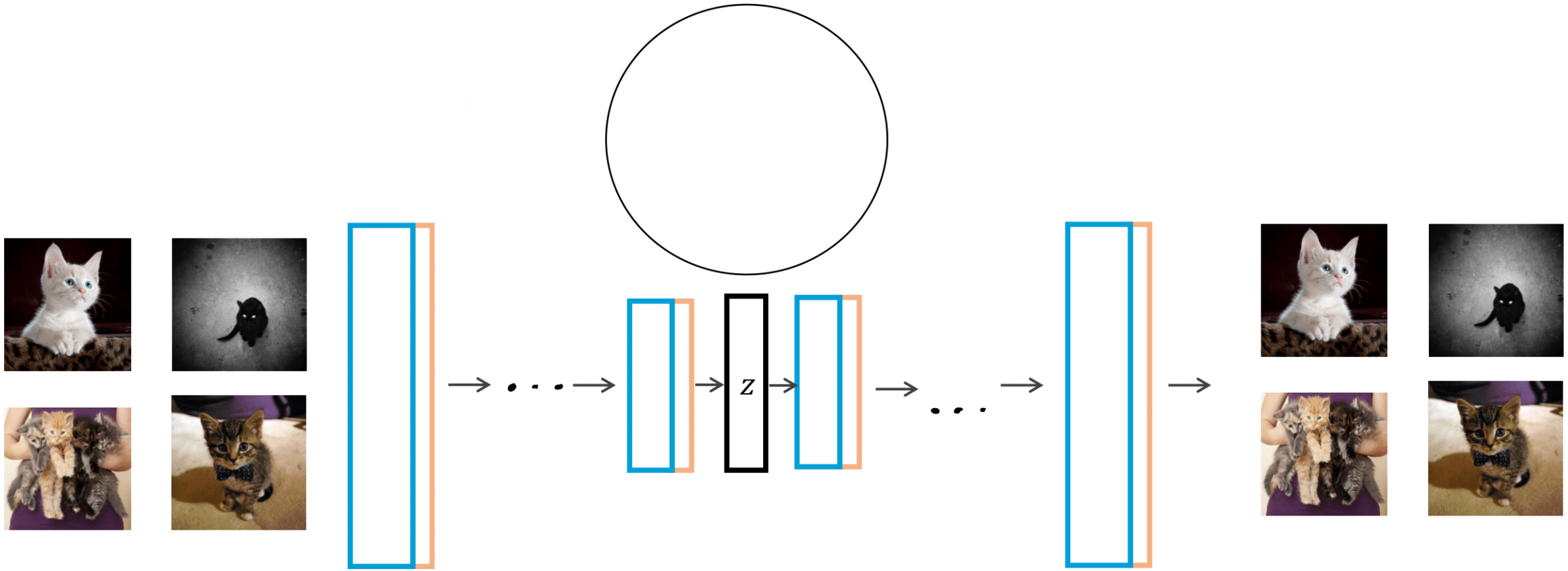
Train such that features can be used to reconstruct original data





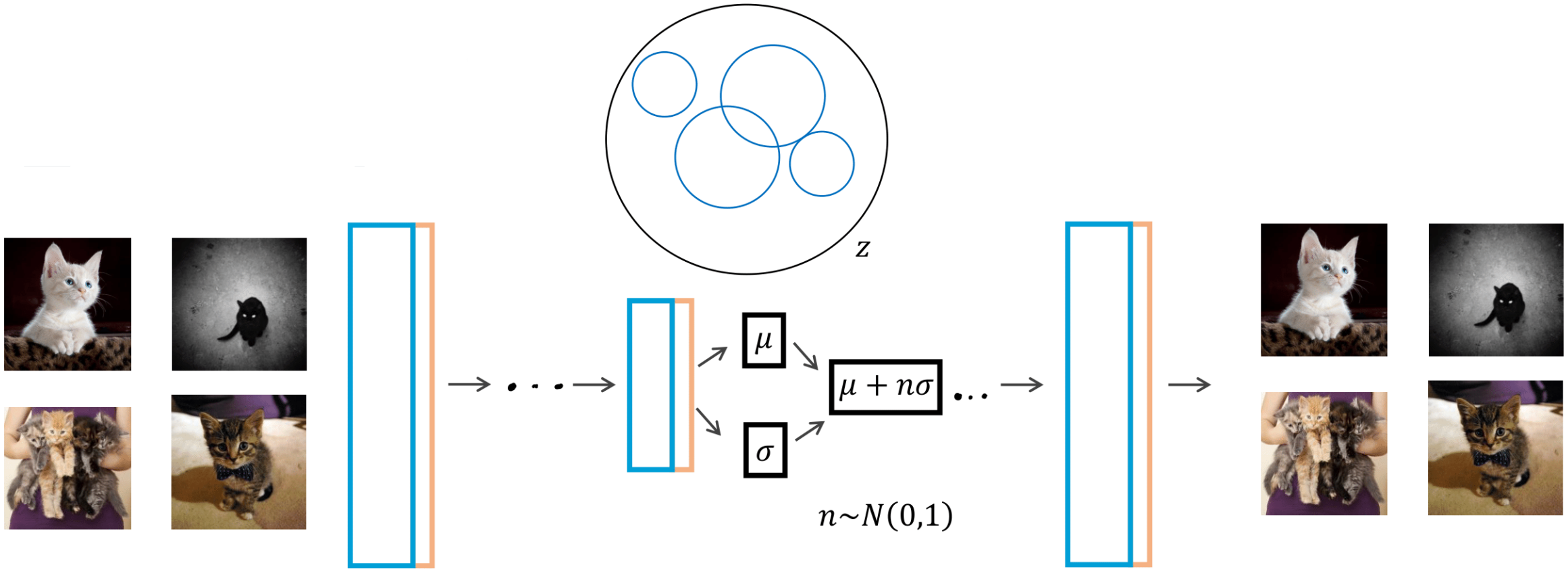
Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

Some background first: Autoencoders



Source: slides by @sim0nsays

Variational Autoencoder



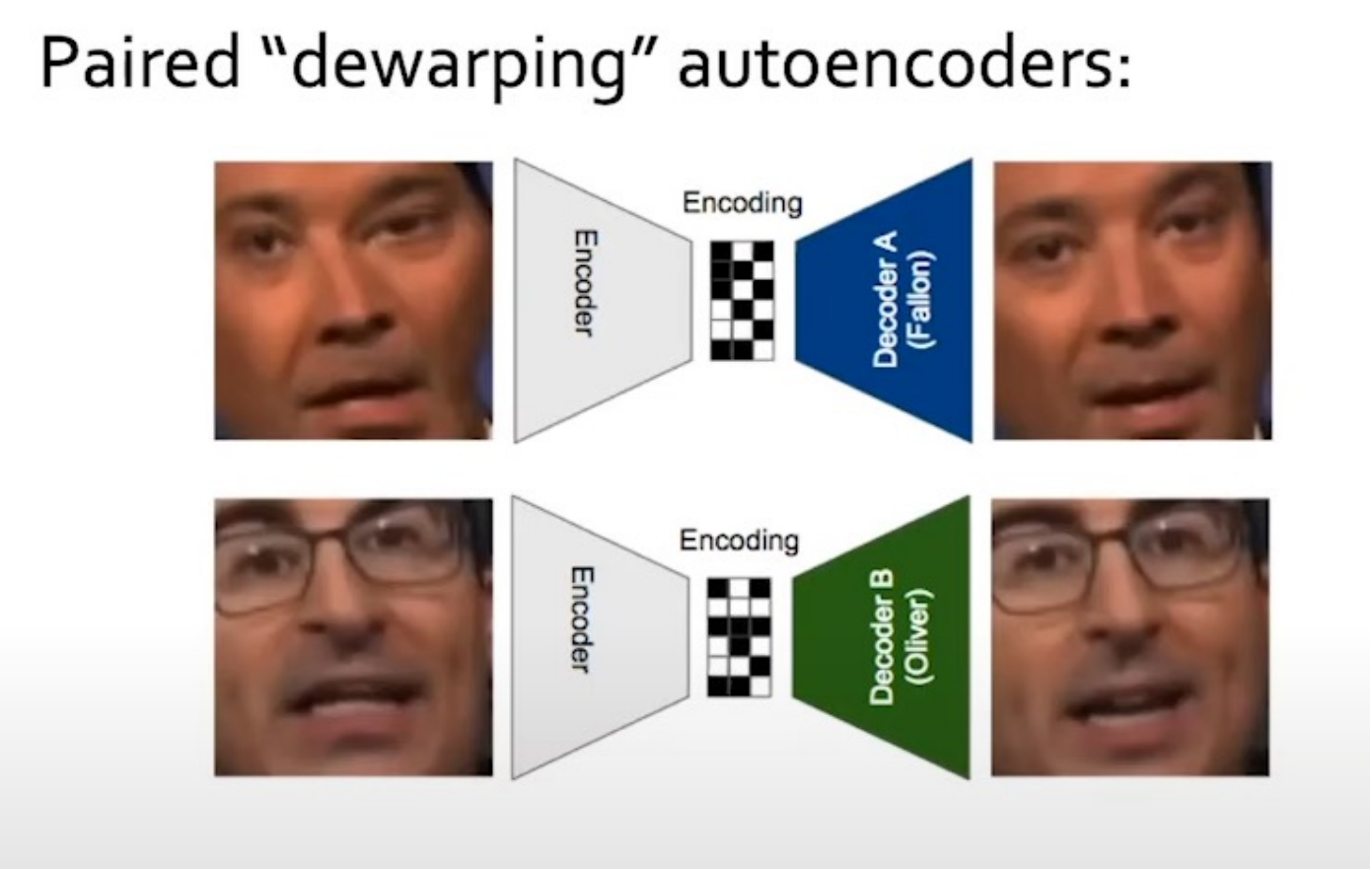
$$L_{\mathrm{KL}} = D_{KL}\left(\mu, \sigma \,||N(0,1)\right)$$

$$= \frac{1}{2}\sum_{j=0}^{N}\left(1 + \log \sigma_j^2 - \mu_j^2 - \sigma_j^2\right)$$

$$L_{\mathrm{r}} = \|x_{\mathrm{out}} - x_{in}\|^2$$

Source: slides by @sim0nsays

Deep Fake



[Gaurav Oberoi blog]

Generative Adversarial Networks

Ian Goodfellow et al., "Generative Adversarial Nets", NIPS 2014

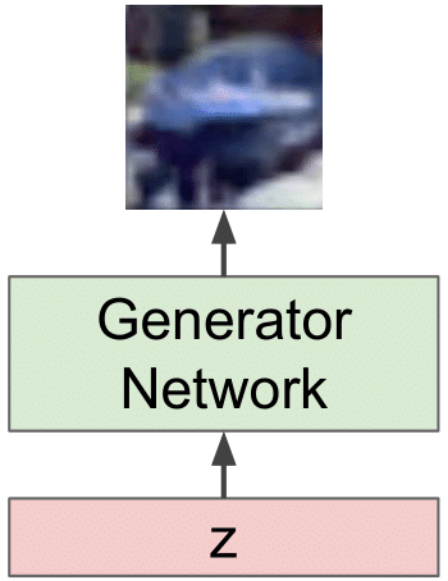
Problem: Want to sample from complex, high-dimensional training distribution. No direct way to do this!

Solution: Sample from a simple distribution, e.g. random noise. Learn transformation to training distribution.

Q: What can we use to represent this complex transformation?

A: A neural network!

Output: Sample from training distribution



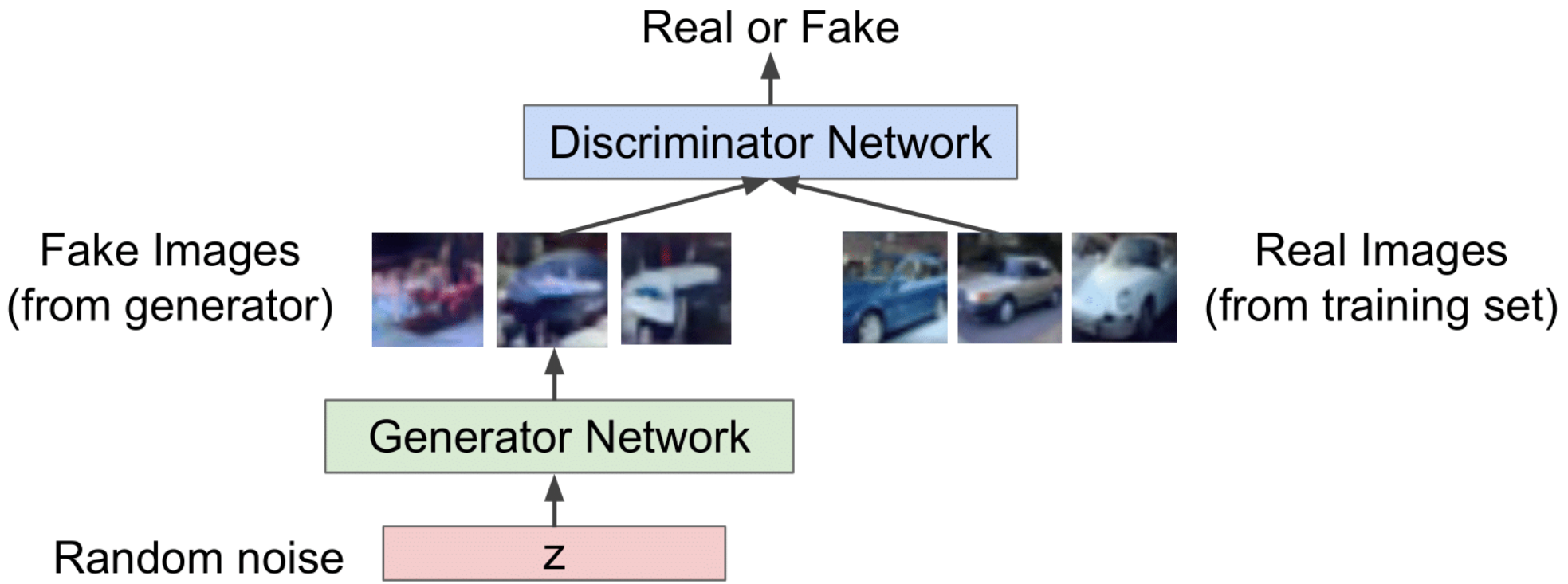
Input: Random noise

Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

Training GANs: Two-player game

Ian Goodfellow et al., "Generative Adversarial Nets", NIPS 2014

Generator network: try to fool the discriminator by generating real-looking images
Discriminator network: try to distinguish between real and fake images



Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

Training GANs: Two-player game

Ian Goodfellow et al., "Generative Adversarial Nets", NIPS 2014

Generator network: try to fool the discriminator by generating real-looking images
Discriminator network: try to distinguish between real and fake images

Train jointly in **minimax game**

Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

Training GANs: Two-player game

Ian Goodfellow et al., "Generative Adversarial Nets", NIPS 2014

Generator network: try to fool the discriminator by generating real-looking images
Discriminator network: try to distinguish between real and fake images

Train jointly in **minimax game**

Discriminator outputs likelihood in (0,1) of real image

Minimax objective function:

$$\min_{\theta_g} \max_{\theta_d} \left[\mathbb{E}_{x \sim p_{data}} \log D_{\theta_d}(x) + \mathbb{E}_{z \sim p(z)} \log(1 - D_{\theta_d}(G_{\theta_g}(z))) \right]$$

Discriminator output for real data x

Discriminator output for generated fake data G(z)

- Discriminator (θ_d) wants to **maximize objective** such that D(x) is close to 1 (real) and D(G(z)) is close to 0 (fake)
- Generator (θ_g) wants to **minimize objective** such that D(G(z)) is close to 1 (discriminator is fooled into thinking generated G(z) is real)

Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

Training GANs: Two-player game

Ian Goodfellow et al., "Generative Adversarial Nets", NIPS 2014

Putting it together: GAN training algorithm

for number of training iterations **do**

  for k steps **do**

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Sample minibatch of m examples $\{\boldsymbol{x}^{(1)}, \ldots, \boldsymbol{x}^{(m)}\}$ from data generating distribution $p_{\text{data}}(\boldsymbol{x})$.
- Update the discriminator by ascending its stochastic gradient:

$$\nabla_{\theta_d} \frac{1}{m} \sum_{i=1}^{m} \left[\log D_{\theta_d}(x^{(i)}) + \log(1 - D_{\theta_d}(G_{\theta_g}(z^{(i)}))) \right]$$

  end for

- Sample minibatch of m noise samples $\{\boldsymbol{z}^{(1)}, \ldots, \boldsymbol{z}^{(m)}\}$ from noise prior $p_g(\boldsymbol{z})$.
- Update the generator by ascending its stochastic gradient (improved objective):

$$\nabla_{\theta_g} \frac{1}{m} \sum_{i=1}^{m} \log(D_{\theta_d}(G_{\theta_g}(z^{(i)})))$$

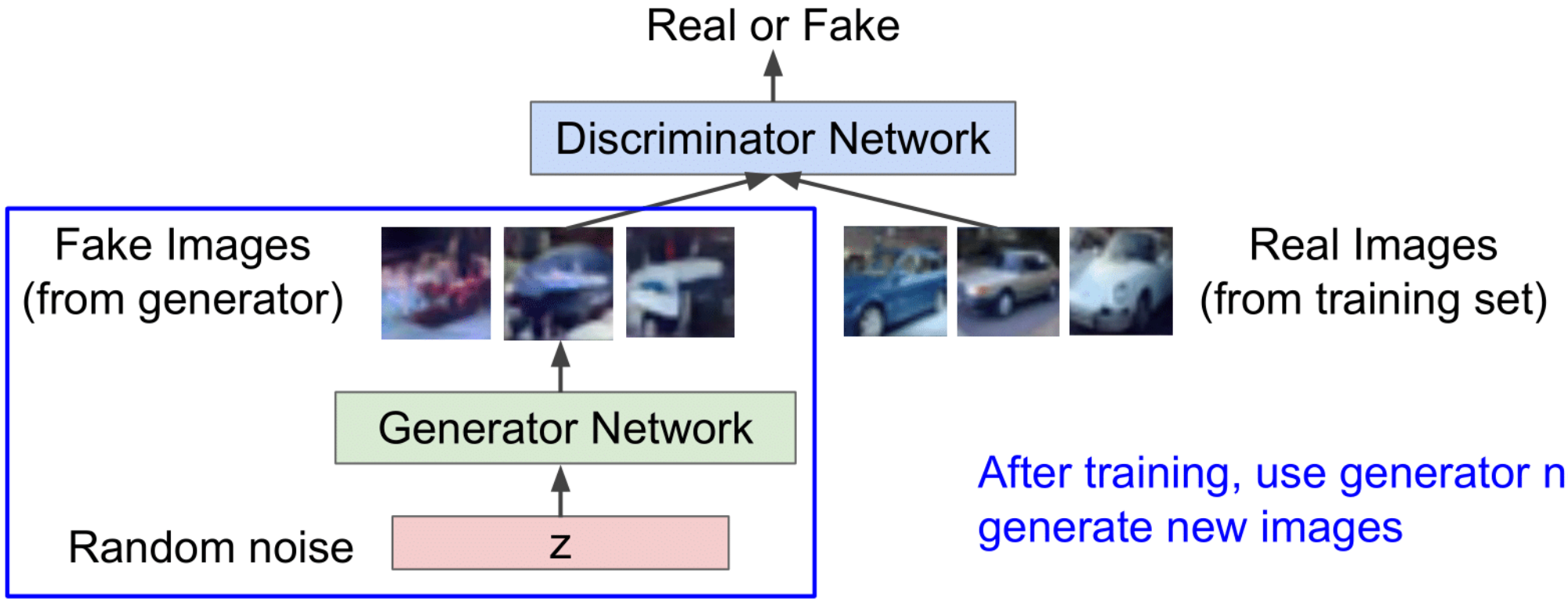
end for

Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

Training GANs: Two-player game

Ian Goodfellow et al., "Generative Adversarial Nets", NIPS 2014

Generator network: try to fool the discriminator by generating real-looking images
Discriminator network: try to distinguish between real and fake images



After training, use generator network to generate new images

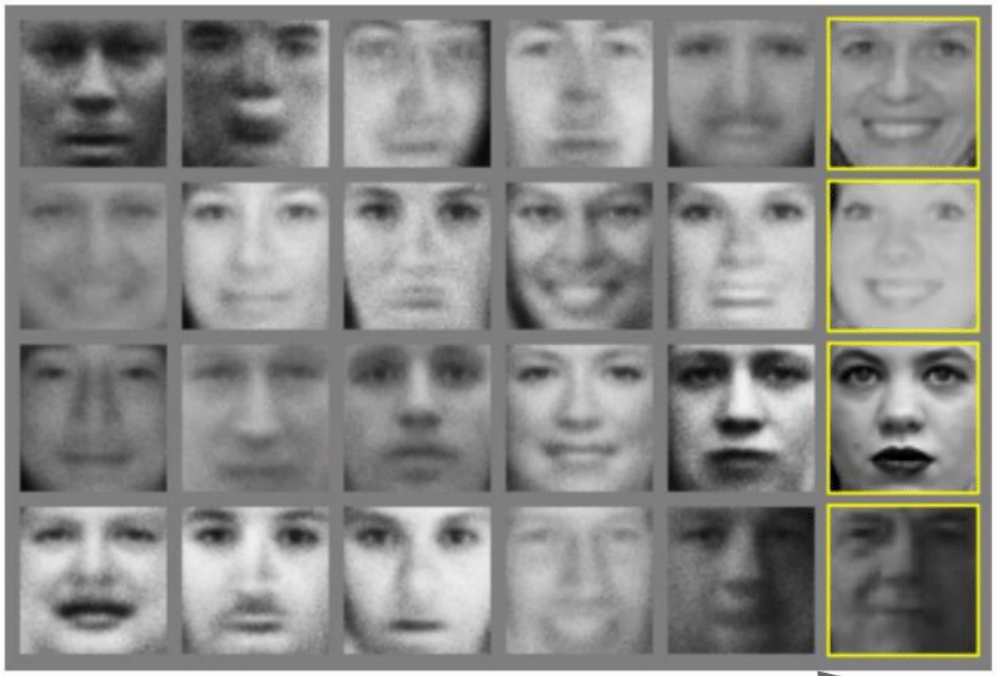
Fake and real images copyright Emily Denton et al. 2015. Reproduced with permission.

Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

Ian Goodfellow et al., “Generative Adversarial Nets”, NIPS 2014

Generative Adversarial Nets

Generated samples



Nearest neighbor from training set

Ian Goodfellow et al., “Generative Adversarial Nets”, NIPS 2014

Generative Adversarial Nets

Generated samples (CIFAR-10)

Nearest neighbor from training set

Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

Generative Adversarial Nets: Convolutional Architectures

Generator is an upsampling network with fractionally-strided convolutions
Discriminator is a convolutional network

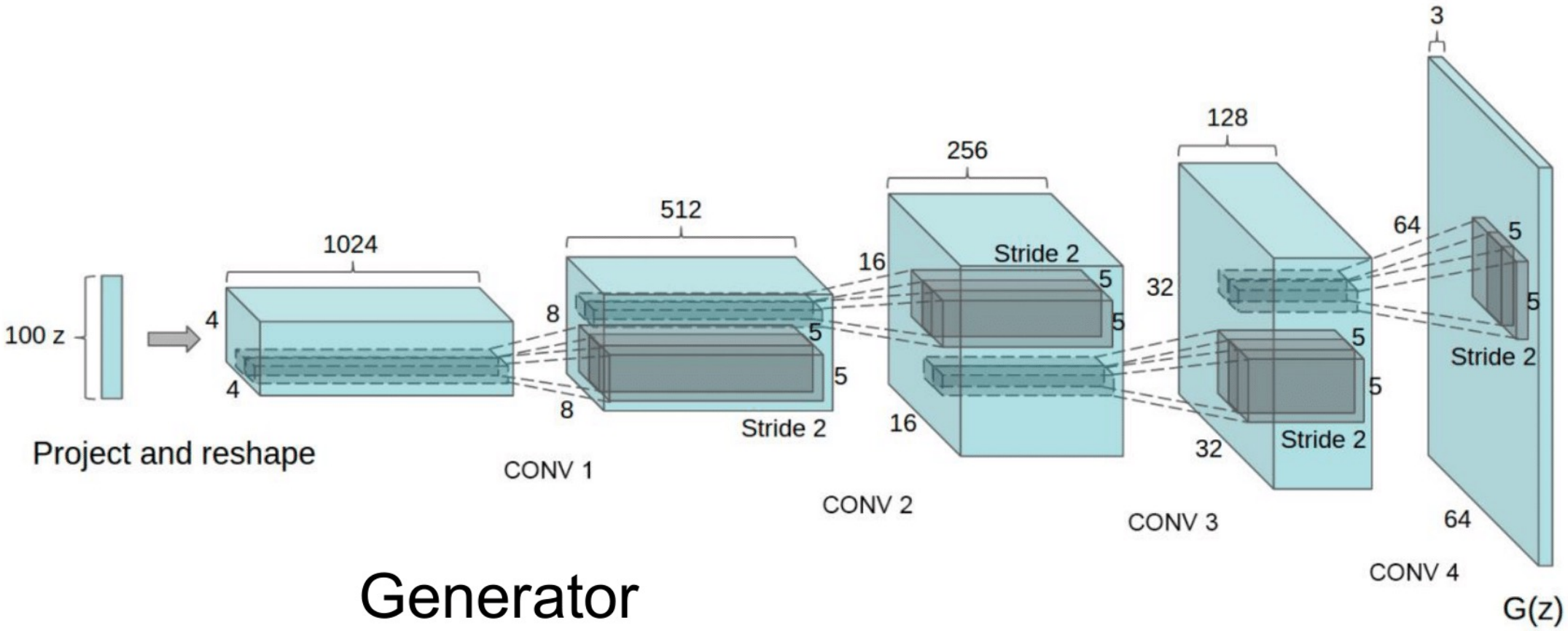
Architecture guidelines for stable Deep Convolutional GANs

- Replace any pooling layers with strided convolutions (discriminator) and fractional-strided convolutions (generator).
- Use batchnorm in both the generator and the discriminator.
- Remove fully connected hidden layers for deeper architectures.
- Use ReLU activation in generator for all layers except for the output, which uses Tanh.
- Use LeakyReLU activation in the discriminator for all layers.

Radford et al, “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”, ICLR 2016

Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

Generative Adversarial Nets: Convolutional Architectures



Generator

Radford et al, “Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks”, ICLR 2016

Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

Generative Adversarial Nets: Convolutional Architectures

Samples from the model look amazing!

Radford et al, ICLR 2016

Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

Generative Adversarial Nets: Convolutional Architectures

Interpolating between random points in laten space

Radford et al, ICLR 2016

Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

Generative Adversarial Nets: Interpretable Vector Math

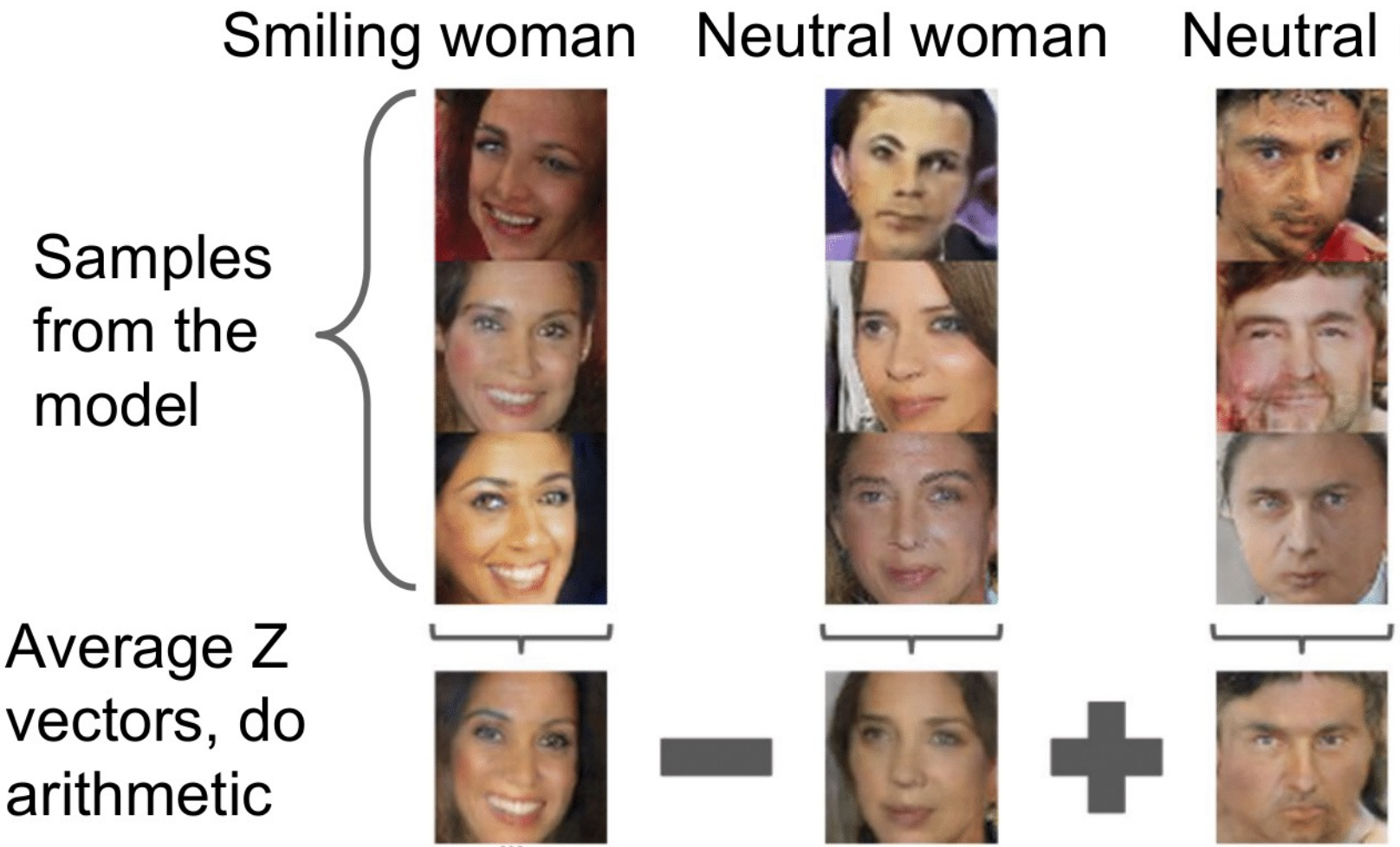
Radford et al, ICLR 2016

Smiling woman | Neutral woman | Neutral man

Samples from the model

Generative Adversarial Nets: Interpretable Vector Math

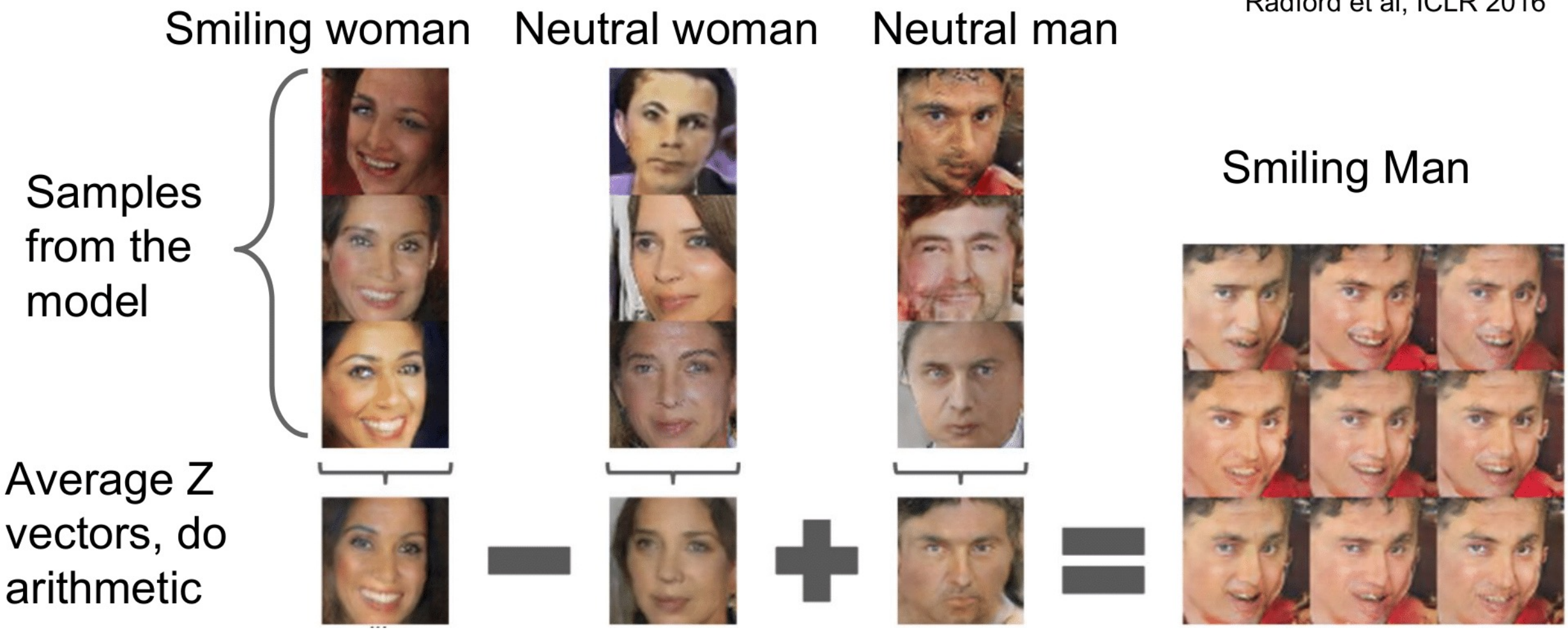
Radford et al, ICLR 2016



Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

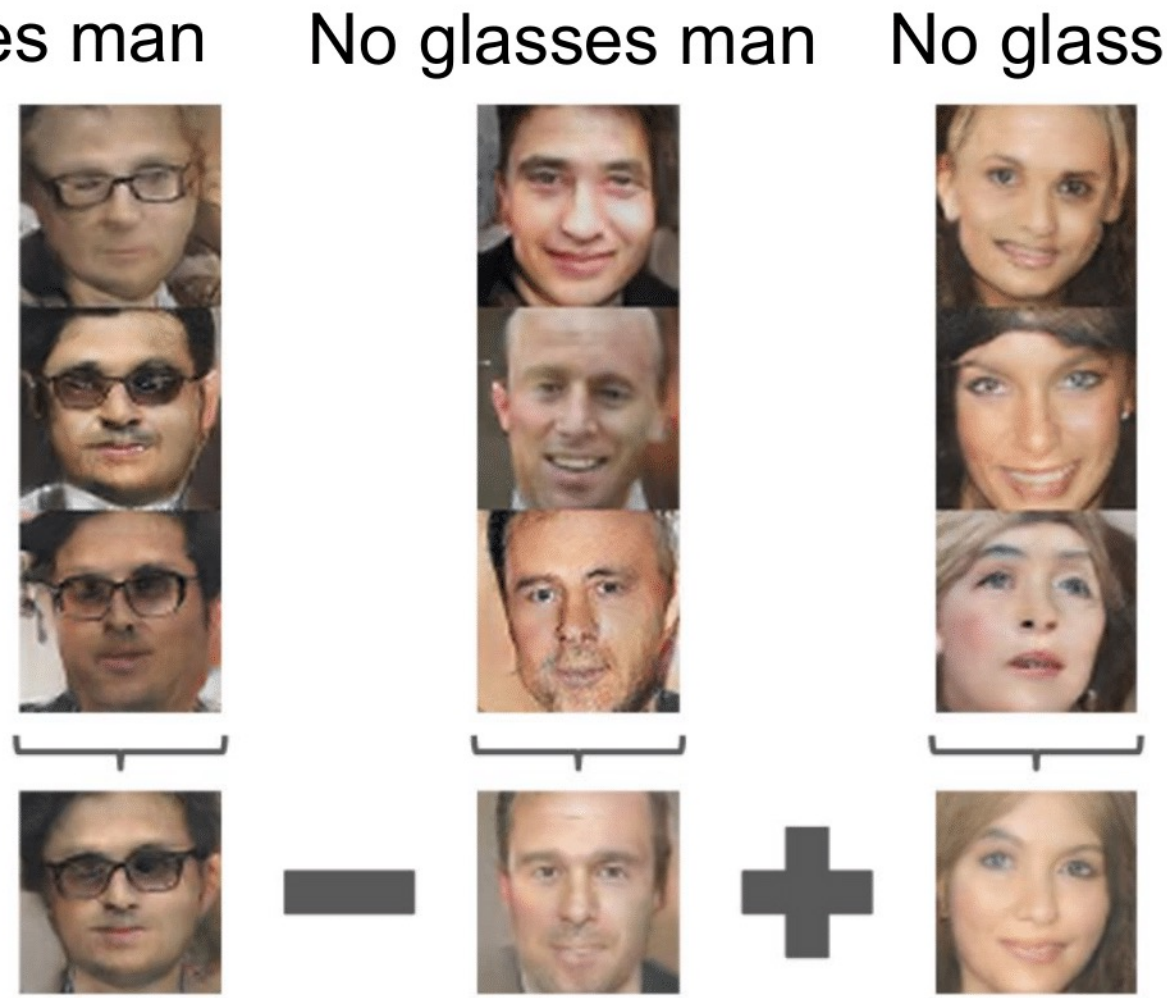
Generative Adversarial Nets: Interpretable Vector Math

Radford et al, ICLR 2016



Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

Generative Adversarial Nets: Interpretable Vector Math



Radford et al, ICLR 2016

Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

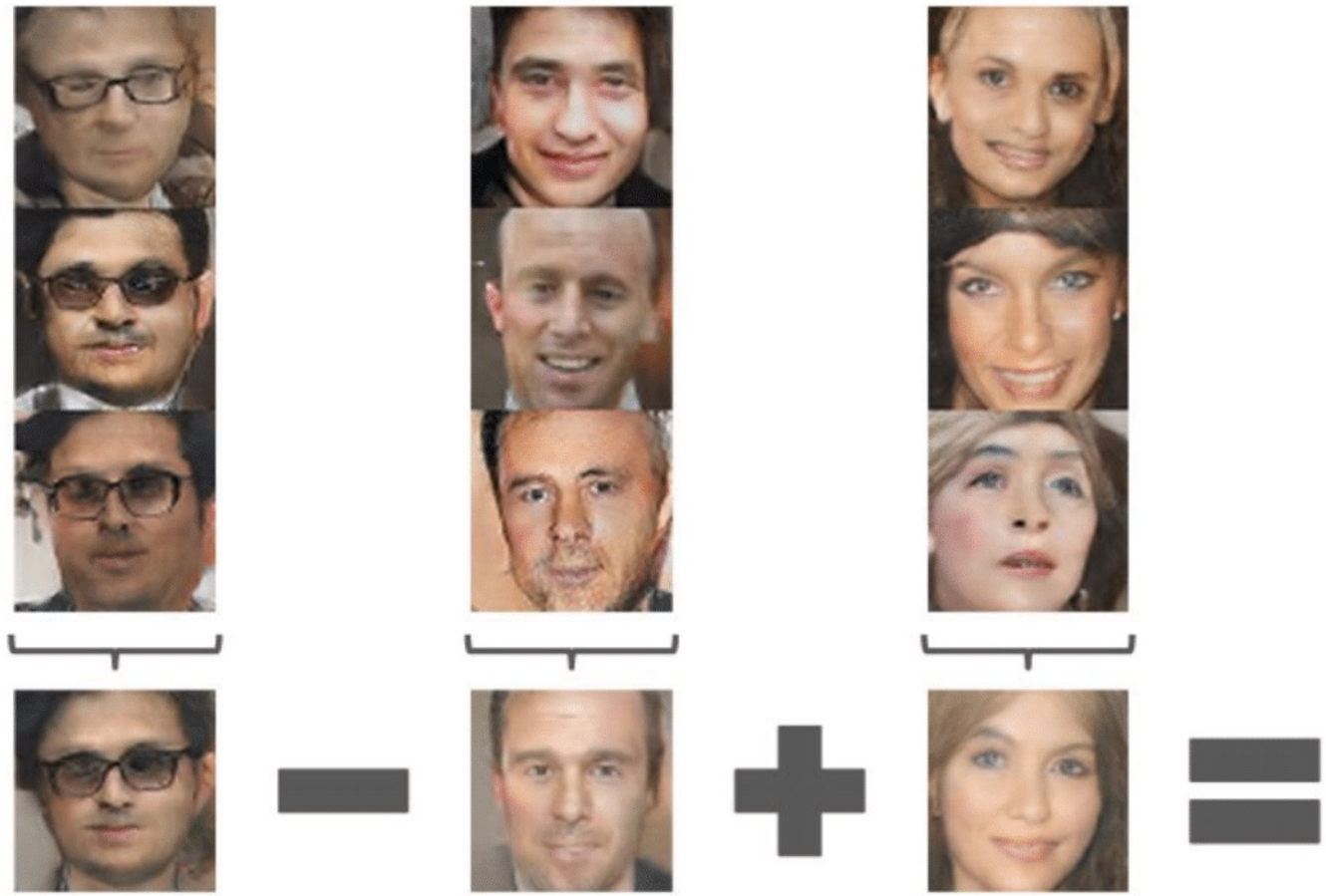
Generative Adversarial Nets: Interpretable Vector Math

Glasses man

No glasses man

No glasses woman

Radford et al, ICLR 2016

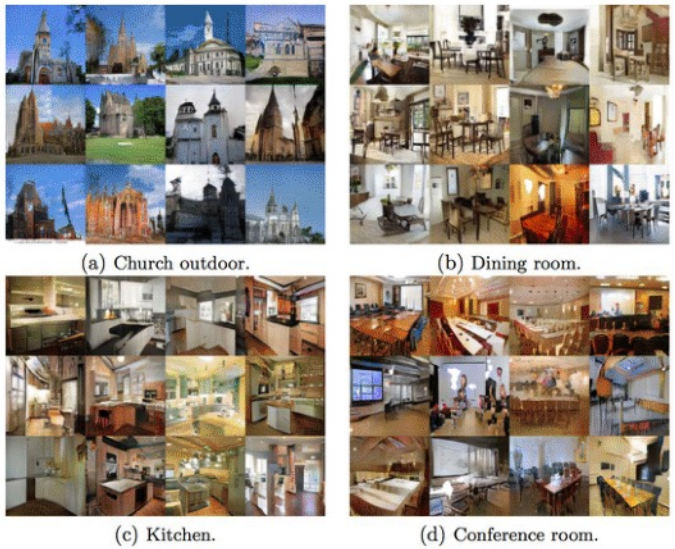


Woman with glasses

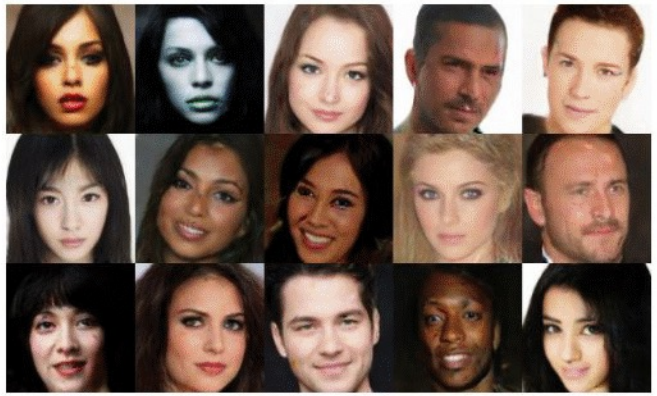
Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

2017: Year of the GAN

Better training and generation

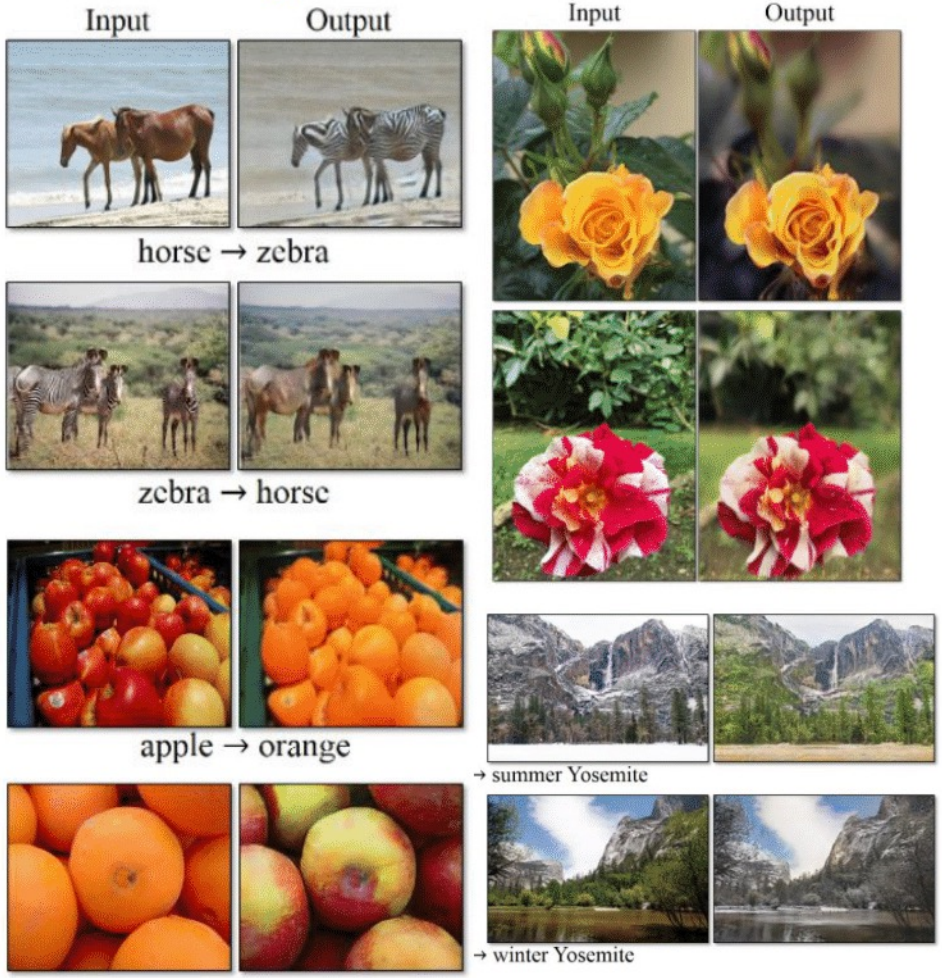


LSGAN. Mao et al. 2017.



BEGAN. Bertholet et al. 2017.

Source->Target domain transfer



CycleGAN. Zhu et al. 2017.

Text -> Image Synthesis

Reed et al. 2017.

Many GAN applications

Pix2pix. Isola 2017. Many examples at https://phillipi.github.io/pix2pix/

Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

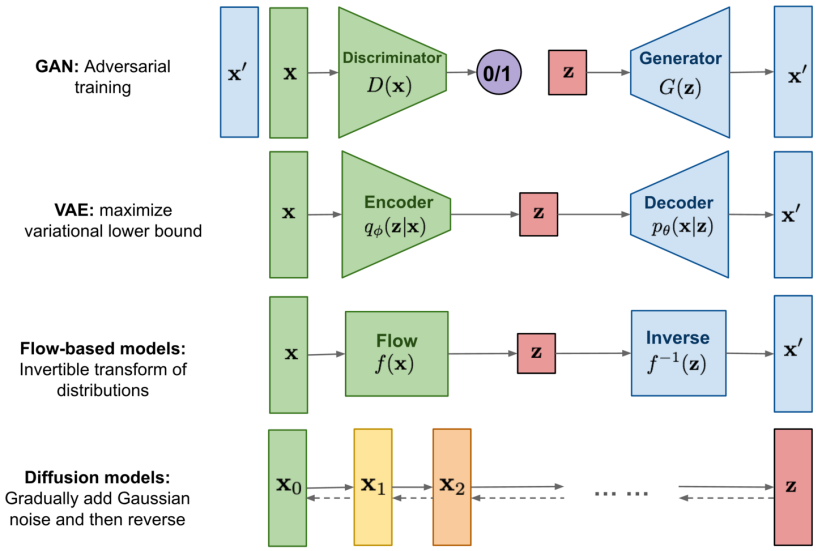
“The GAN Zoo”

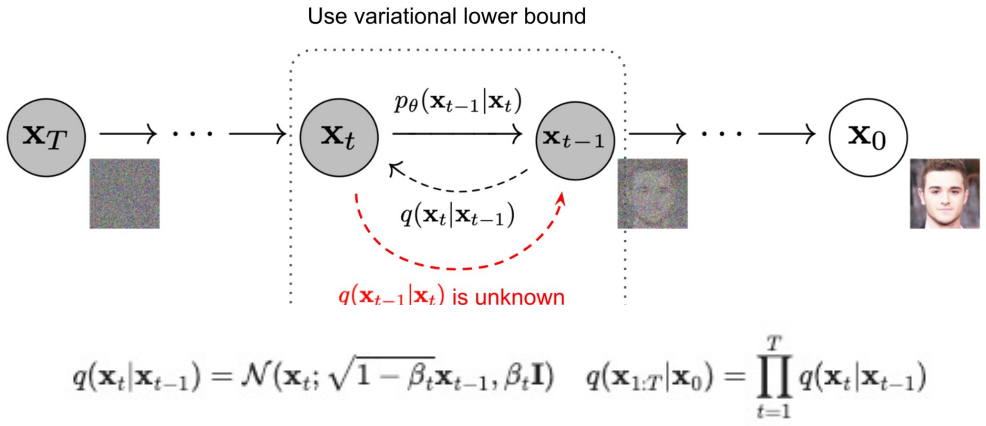
- GAN - Generative Adversarial Networks
- 3D-GAN - Learning a Probabilistic Latent Space of Object Shapes via 3D Generative-Adversarial Modeling
- acGAN - Face Aging With Conditional Generative Adversarial Networks
- AC-GAN - Conditional Image Synthesis With Auxiliary Classifier GANs
- AdaGAN - AdaGAN: Boosting Generative Models
- AEGAN - Learning Inverse Mapping by Autoencoder based Generative Adversarial Nets
- AffGAN - Amortised MAP Inference for Image Super-resolution
- AL-CGAN - Learning to Generate Images of Outdoor Scenes from Attributes and Semantic Layouts
- ALI - Adversarially Learned Inference
- AM-GAN - Generative Adversarial Nets with Labeled Data by Activation Maximization
- AnoGAN - Unsupervised Anomaly Detection with Generative Adversarial Networks to Guide Marker Discovery
- ArtGAN - ArtGAN: Artwork Synthesis with Conditional Categorial GANs
- b-GAN - b-GAN: Unified Framework of Generative Adversarial Networks
- Bayesian GAN - Deep and Hierarchical Implicit Models
- BEGAN - BEGAN: Boundary Equilibrium Generative Adversarial Networks
- BiGAN - Adversarial Feature Learning
- BS-GAN - Boundary-Seeking Generative Adversarial Networks
- CGAN - Conditional Generative Adversarial Nets
- CaloGAN - CaloGAN: Simulating 3D High Energy Particle Showers in Multi-Layer Electromagnetic Calorimeters with Generative Adversarial Networks
- CCGAN - Semi-Supervised Learning with Context-Conditional Generative Adversarial Networks
- CatGAN - Unsupervised and Semi-supervised Learning with Categorical Generative Adversarial Networks
- CoGAN - Coupled Generative Adversarial Networks
- Context-RNN-GAN - Contextual RNN-GANs for Abstract Reasoning Diagram Generation
- C-RNN-GAN - C-RNN-GAN: Continuous recurrent neural networks with adversarial training
- CS-GAN - Improving Neural Machine Translation with Conditional Sequence Generative Adversarial Nets
- CVAE-GAN - CVAE-GAN: Fine-Grained Image Generation through Asymmetric Training
- CycleGAN - Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks
- DTN - Unsupervised Cross-Domain Image Generation
- DCGAN - Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks
- DiscoGAN - Learning to Discover Cross-Domain Relations with Generative Adversarial Networks
- DR-GAN - Disentangled Representation Learning GAN for Pose-Invariant Face Recognition
- DualGAN - DualGAN: Unsupervised Dual Learning for Image-to-Image Translation
- EBGAN - Energy-based Generative Adversarial Network
- f-GAN - f-GAN: Training Generative Neural Samplers using Variational Divergence Minimization
- FF-GAN - Towards Large-Pose Face Frontalization in the Wild
- GAWWN - Learning What and Where to Draw
- GeneGAN - GeneGAN: Learning Object Transfiguration and Attribute Subspace from Unpaired Data
- Geometric GAN - Geometric GAN
- GoGAN - Gang of GANs: Generative Adversarial Networks with Maximum Margin Ranking
- GP-GAN - GP-GAN: Towards Realistic High-Resolution Image Blending
- IAN - Neural Photo Editing with Introspective Adversarial Networks
- iGAN - Generative Visual Manipulation on the Natural Image Manifold
- IcGAN - Invertible Conditional GANs for image editing
- ID-CGAN - Image De-raining Using a Conditional Generative Adversarial Network
- Improved GAN - Improved Techniques for Training GANs
- InfoGAN - InfoGAN: Interpretable Representation Learning by Information Maximizing Generative Adversarial Nets
- LAGAN - Learning Particle Physics by Example: Location-Aware Generative Adversarial Networks for Physics Synthesis
- LAPGAN - Deep Generative Image Models using a Laplacian Pyramid of Adversarial Networks

https://github.com/hindupuravinash/the-gan-zoo

Source: Stanford CS231n Lecture 13 2017 by Fei-Fei Li & Justin Johnson & Serena Yeung

Diffusion models





$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_t; \sqrt{1-\beta_t}\mathbf{x}_{t-1}, \beta_t\mathbf{I}) \quad q(\mathbf{x}_{1:T}|\mathbf{x}_0) = \prod_{t=1}^{T} q(\mathbf{x}_t|\mathbf{x}_{t-1})$$

Прямое распространение:

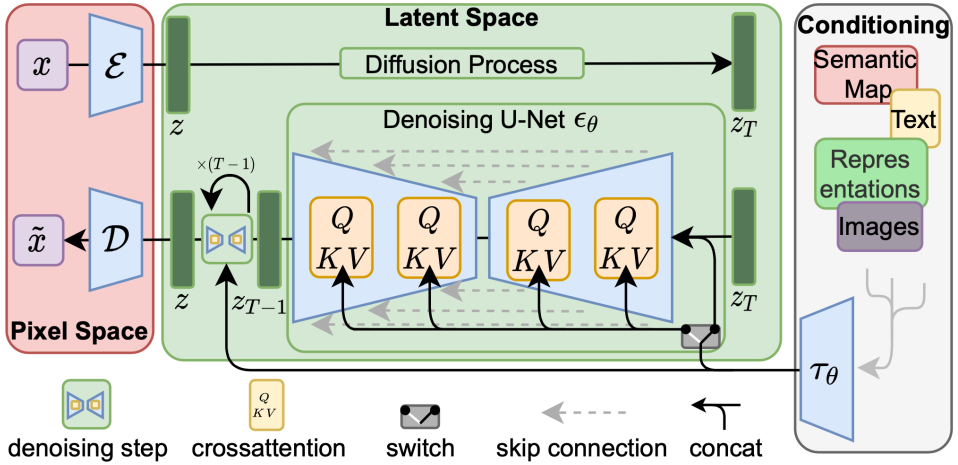
1. Исходное изображение ($\mathbf{x}_0$) медленно итеративно искажается (цепочка Маркова) путем добавления (масштабированного гауссова) шума.
2. Этот процесс выполняется для некоторых временных шагов **T**, т.е. $\mathbf{x}_T$.
3. Изображение на временном шаге **t** создается: $\mathbf{xt}_{-1} + \varepsilon_{t-1}$ **(шум) → xt**
4. На этом этапе модель не задействована.
5. В конце этапа прямой диффузии **Xt**, из-за итеративного добавления шума, мы остаемся с (чистым) зашумленным изображением, представляющим “**изотропный гауссовский**”. Это просто математический способ сказать, что у нас есть стандартное нормальное распределение, и дисперсия распределения одинакова по всем измерениям. Мы преобразовали распределение данных в гауссово распределение.

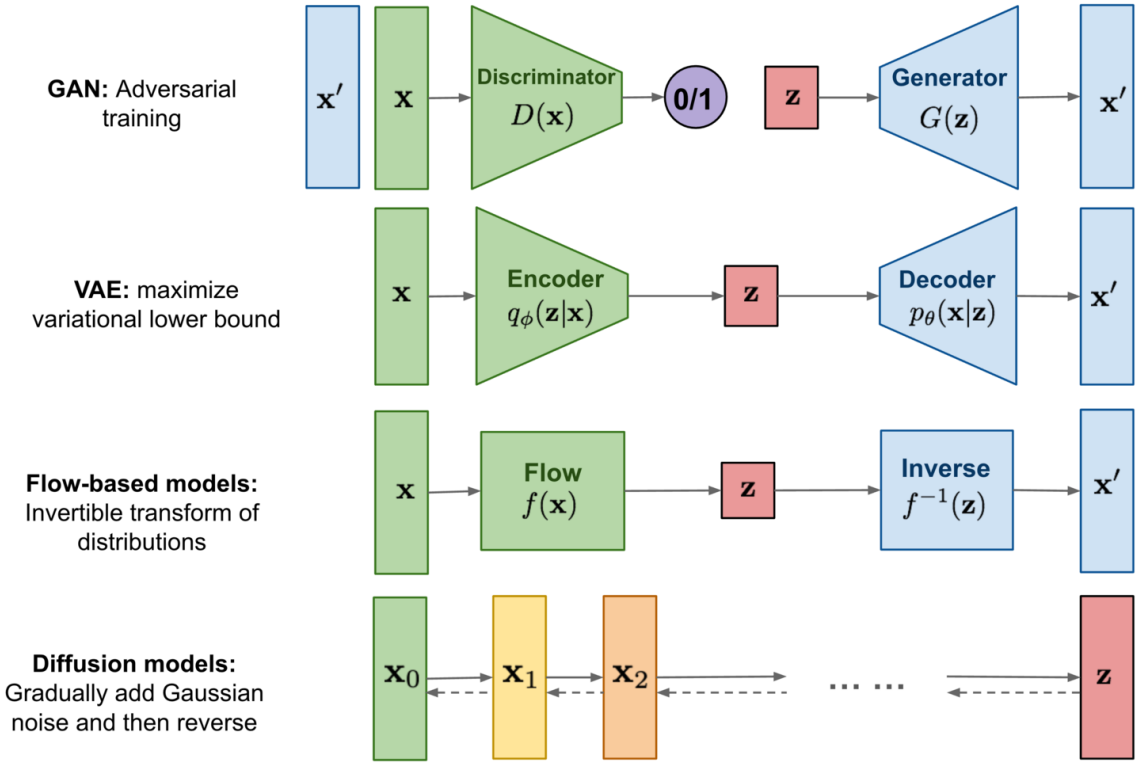
Обратная диффузия:

1. На этом этапе мы отменяем процесс пересылки. Задача состоит в том, чтобы удалить шум, добавленный в прямом процессе, снова итеративным способом (цепочка Маркова). Это делается с использованием модели нейронной сети.
2. Задача модели заключается в следующем: учитывая временной интервал t и зашумленное изображение **xt**, спрогнозируйте шум (ɛ́), добавленный к изображению на шаге $_t$**-1**.
3. **xt** → Модель → **ɛ́** (прогнозируемый шум). Модель предсказывает (аппроксимирует) шум, добавленный к $\mathbf{x}_{t-1}$ при прямом проходе.

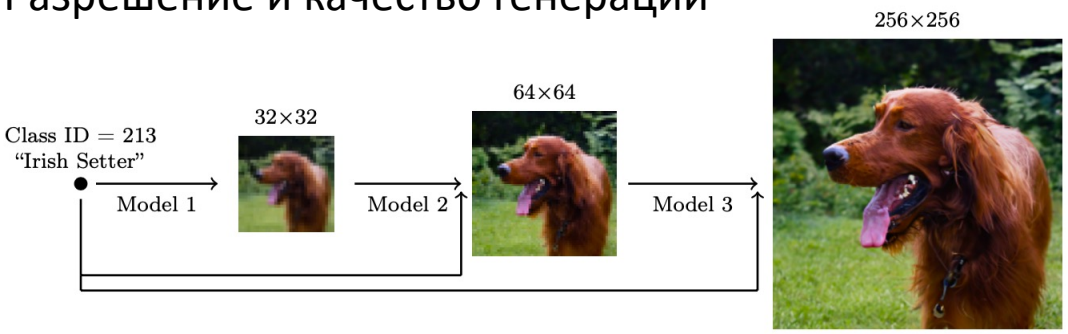
[Больше математики про diffusion]

Diffusion models





Разрешение и качество генерации



Задачи генерации

Text2Image

"Avocado chair" using Midjourney version v4 – By Jack Gallagher

Text2Video

Image Inpainting

Image Outpainting

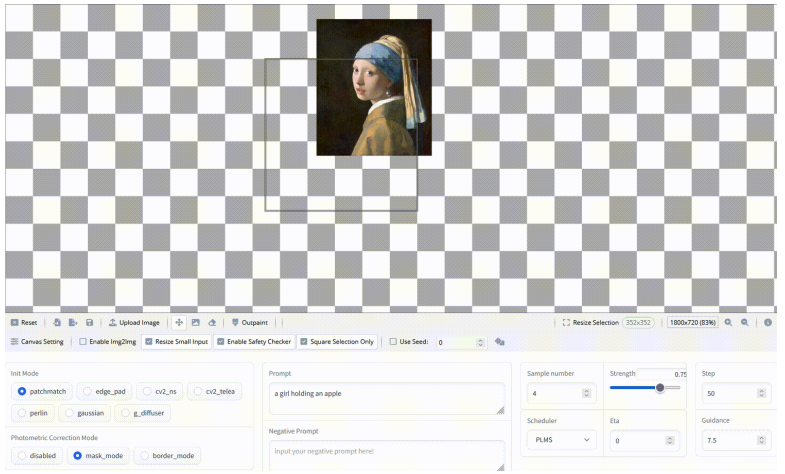


Image2Image

Задачи генерации

Infinite Zoom In & Zoom Out

Процесс, используемый для создания “Infine zoom in”

1.Создайте изображение, используя стабильную диффузионную модель.
2.Уменьшите изображение и скопируйте и вставьте его в центр.
3.Замаскируйте границу за пределами уменьшенной копии.
4.Используйте стабильную диффузионную раскраску для заполнения замаскированной части.