CMPT 983

Grounded Natural Language Understanding

February 09, 2022

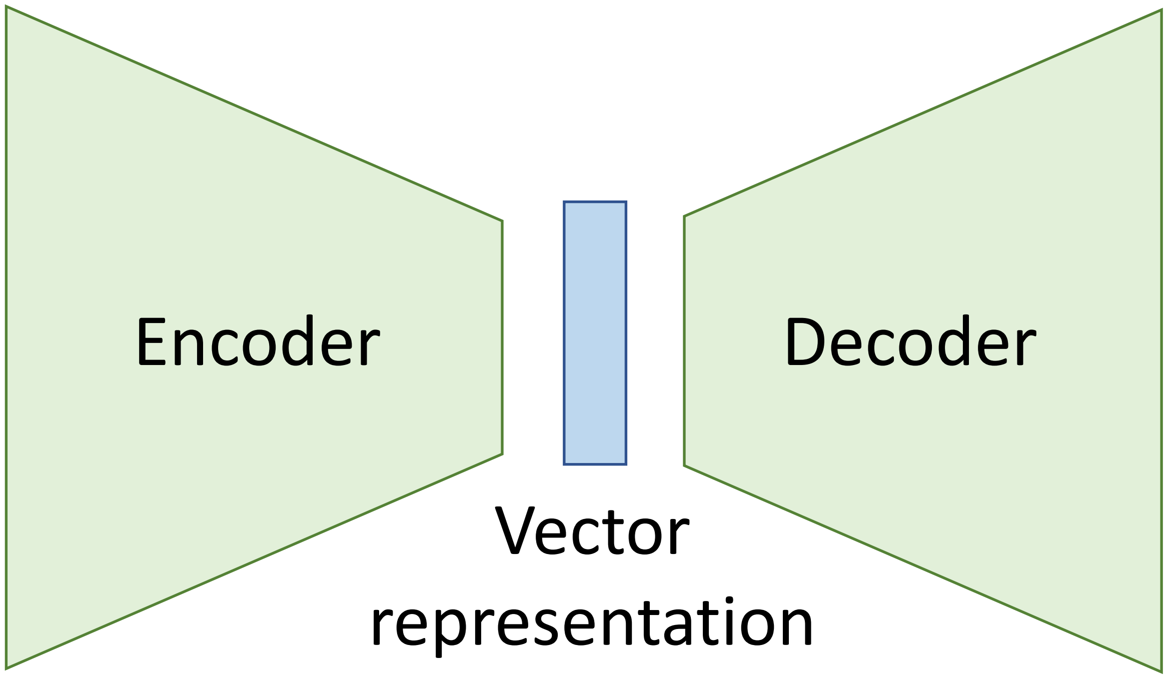
Content generation from language

Content generation from language

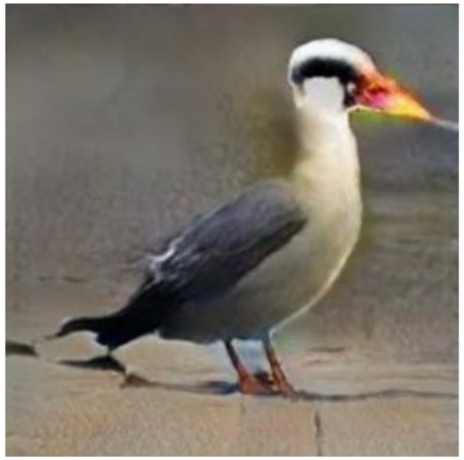
Translating across modalities

Text

A white bird with
a black crown
and yellow beak

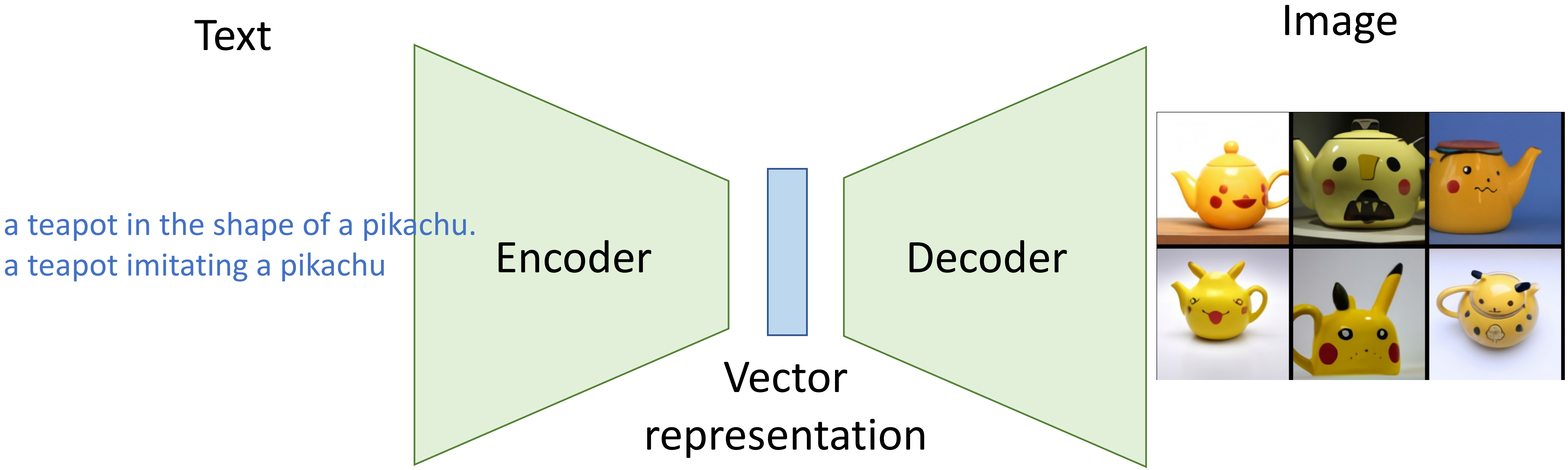


Image



"StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks"
[Zhang et al, ICCV 2017]

Translating across modalities

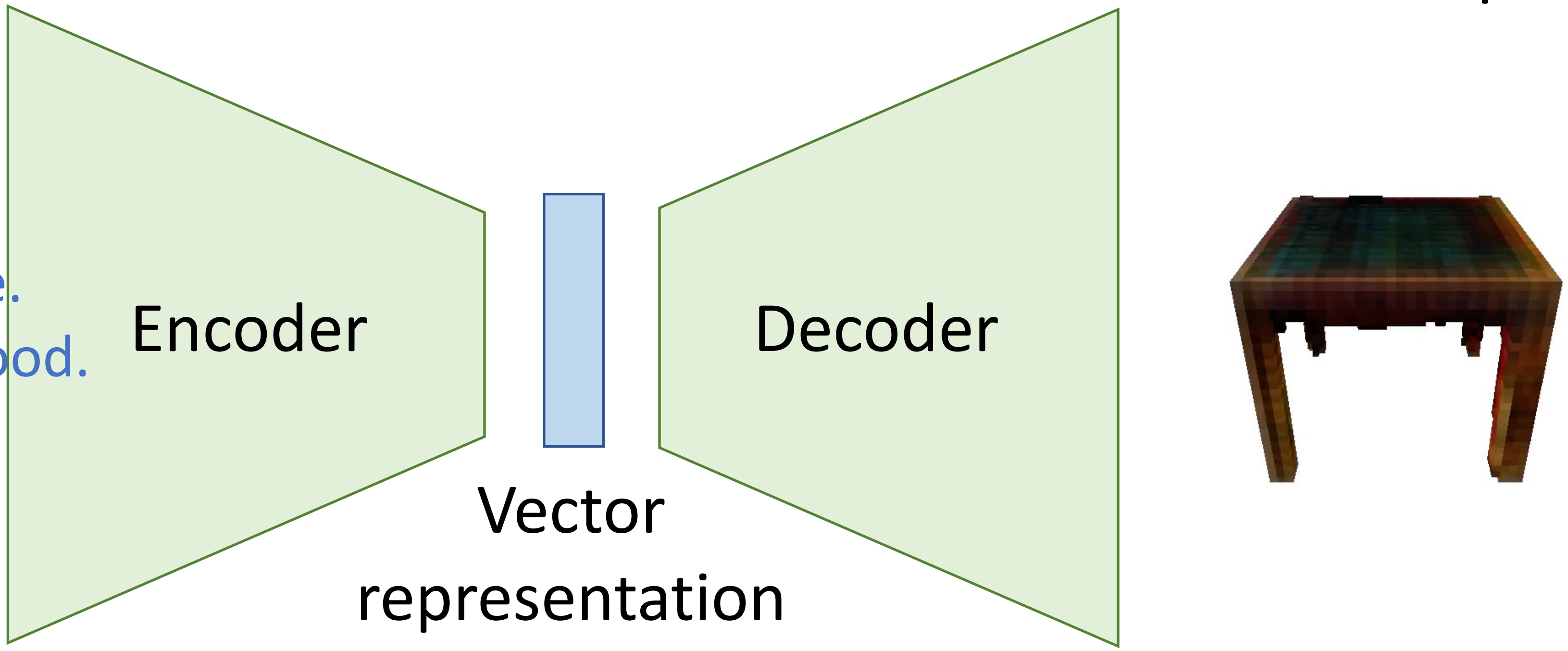


"Dall-e"
[Ramesh et al, https://openai.com/blog/dall-e/]

Translating across modalities

Text

Brown colored dining table.
It has four legs made of wood.



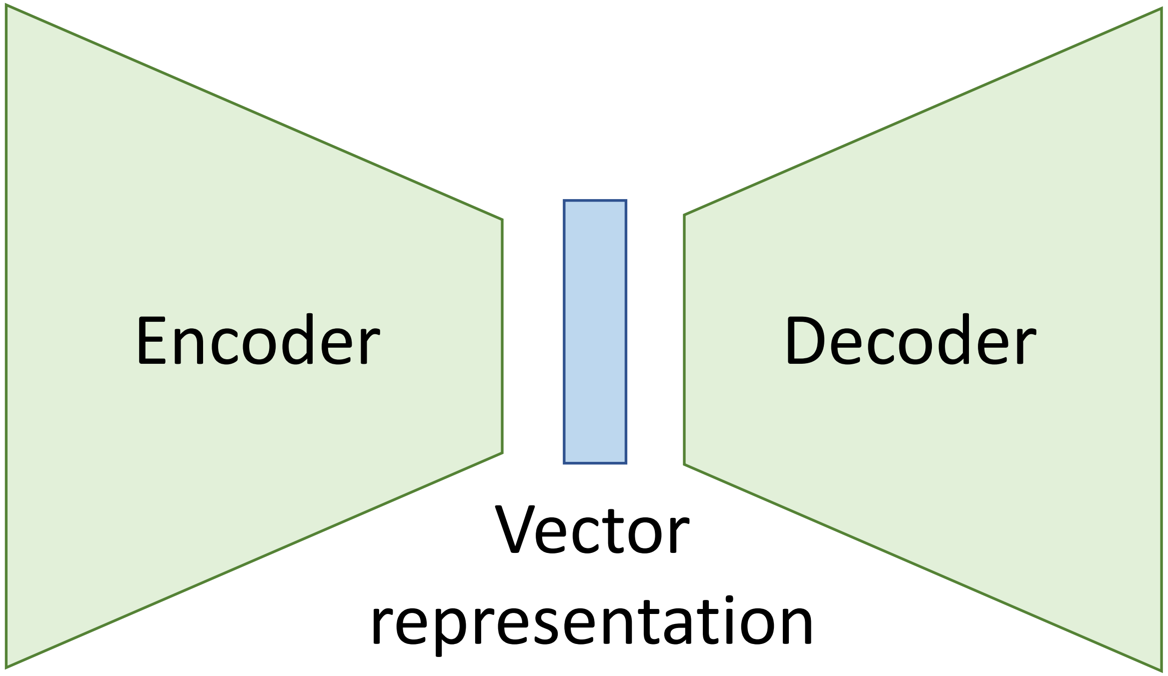
3D Shape

“Text2Shape: Generating Shapes from Natural Language by Learning Joint Embeddings”
[Chen et al, ACCV 2018]

How is generating images and shapes different from generating text?

Translating across modalities

Image



Text

man in black shirt
is playing guitar

Image captioning

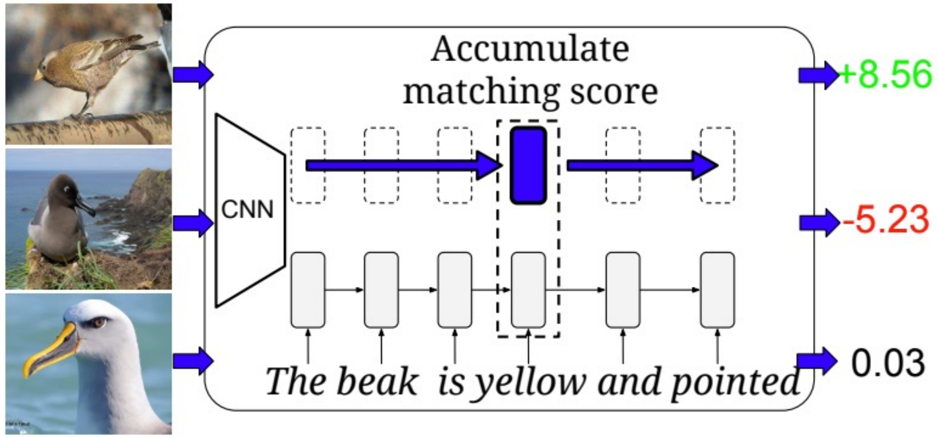
"Deep Visual-Semantic Alignments for Generating Image Descriptions"
[Karpathy and Fei-Fei CVPR 2015]

Generating Content

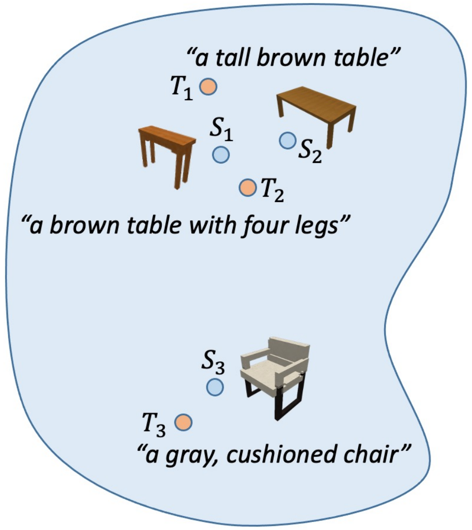
- Note: retrieval as most basic form of generation

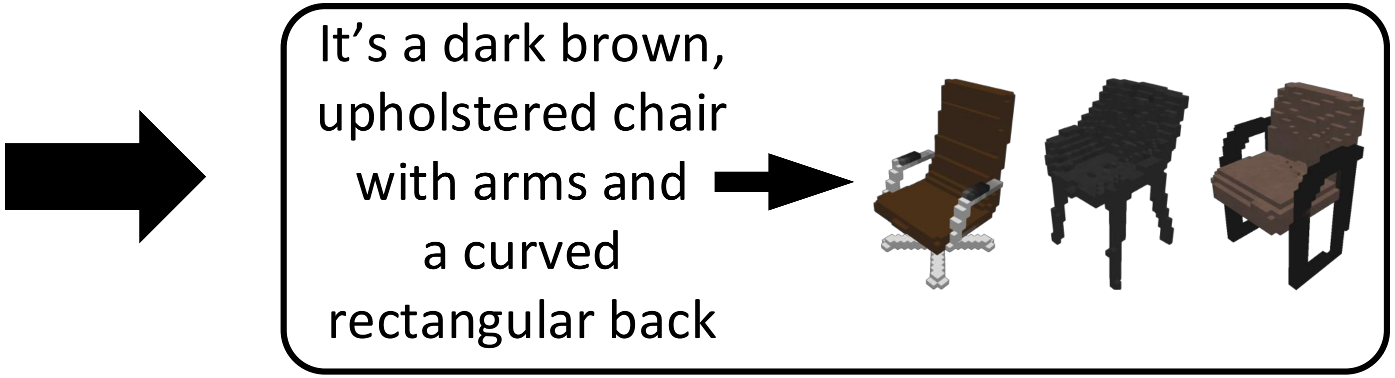
Generation as retrieval

Learn joint embedding → Embed and retrieve



"Learning Deep Representations of Fine-Grained Visual Descriptions" (Reed et al, CVPR 2016)



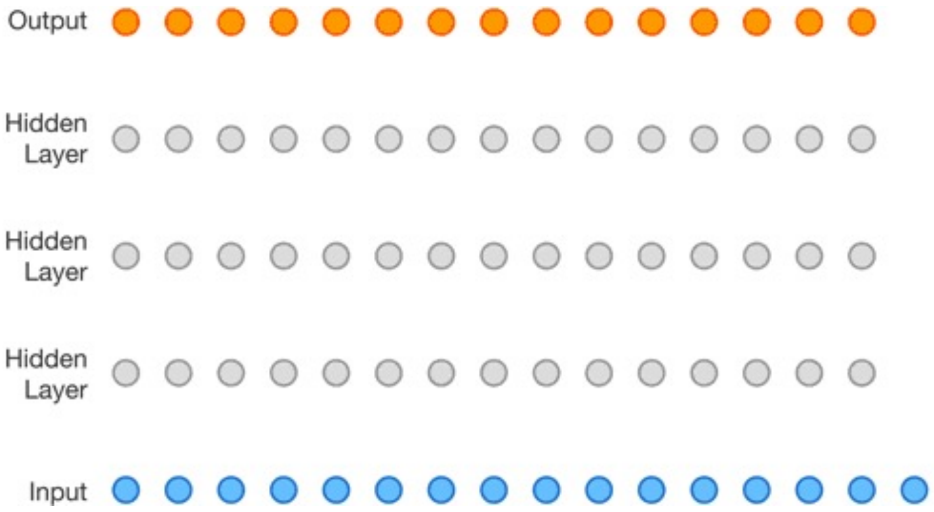


"Text2Shape: Generating Shapes from Natural Language by Learning Joint Embeddings" (Chen et al, ACCV 2018)

Generating Content

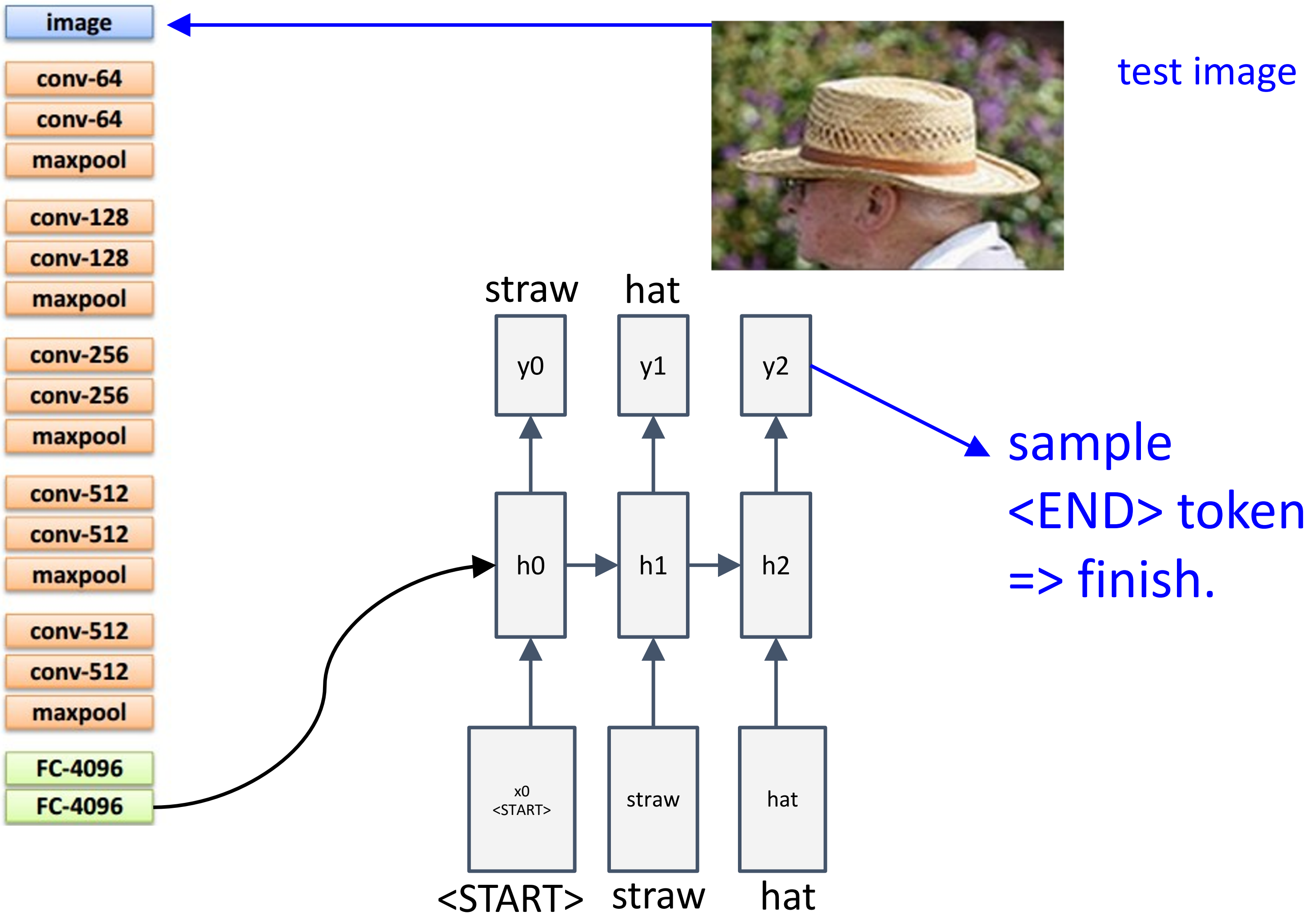
- Note: retrieval as most basic form of generation
  - Can also retrieve + edit

- Note : can model as output as a sequence and generate autoregressively



https://ml.berkeley.edu/blog/posts/AR_intro/

Autoregressive captioning



Output from previous step is fed as input into next

How to get different ouputs?

Decoding strategies:

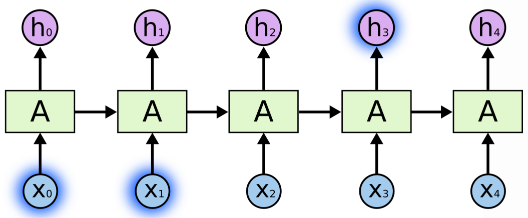
- Greedy decoding
  - Take $\text{argmax}\, P_t(w)$
- Beam search
- Sampling
  - Basic sampling: sample from $P_t(w)$
  - Top-*n* sampling: restrict to top *n* words
  - Top-*p* sampling: restrict to top *p* proportion of words
- Temperature scaling (make distribution less spiky)

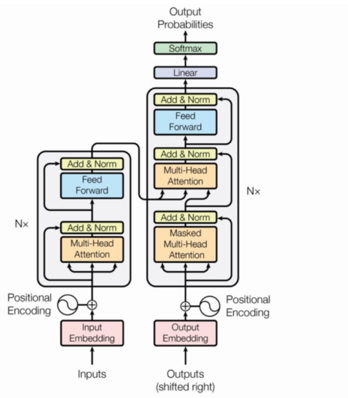
$$P_t(w) = \frac{\exp(s_w/\tau)}{\sum_{w' \in V} \exp(s_{w'}/\tau)}$$

Generating Content

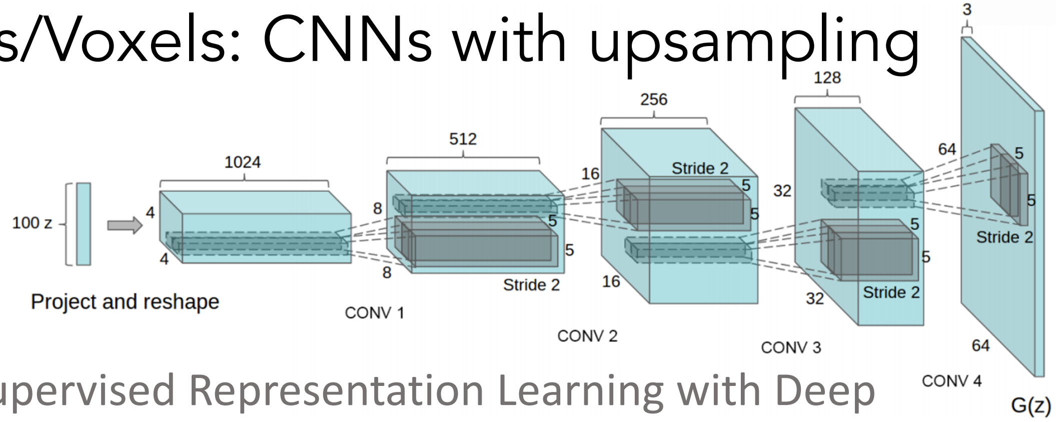
- Note: retrieval as most basic form of generation

- Note : can model as output as a sequence and generate autoregressively





- Decoders:
  - Language: RNNs/Transformers
  - Images/Voxels: CNNs with upsampling



“wrongly called deconvolutions”

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

Taxonomy of machine learning models

Models different probability distributions

Discriminative models:
Learn p(y|x) → Assign labels to data; Feature learning (with labels)

Generative Model:
Learn p(x) → Detect outliers; Feature learning (without labels); Sample to generate new data

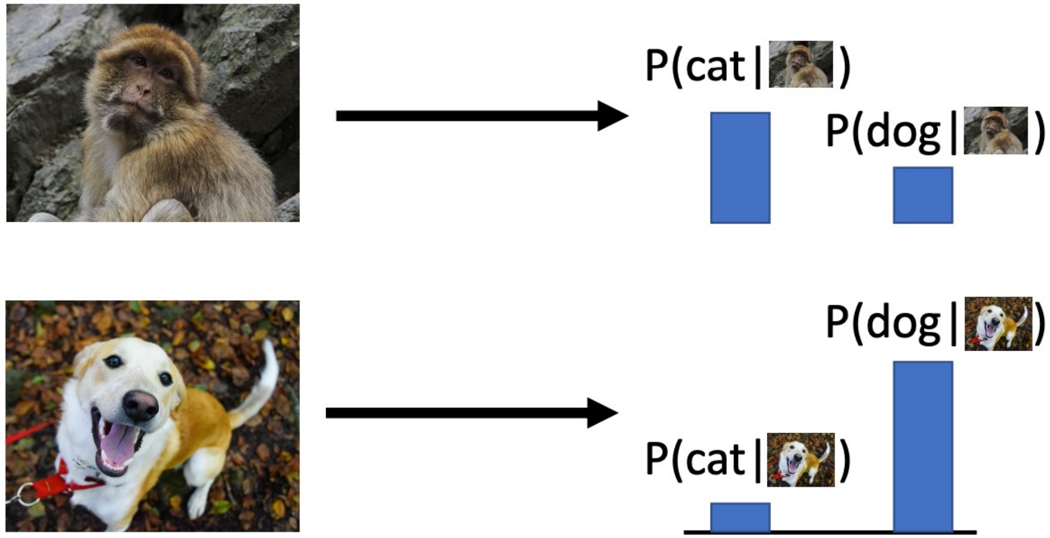
Conditional Generative Model:
Learn p(x|y) → Assign labels, while rejecting outliers!; Generate new data conditioned on input labels

Slide credit: Justin Johnson (https://web.eecs.umich.edu/~justincj/teaching/eecs498/FA2020/schedule.html, L19,20)

Taxonomy of machine learning models

Models different probability distributions

Discriminative models:
Learn p(y|x)



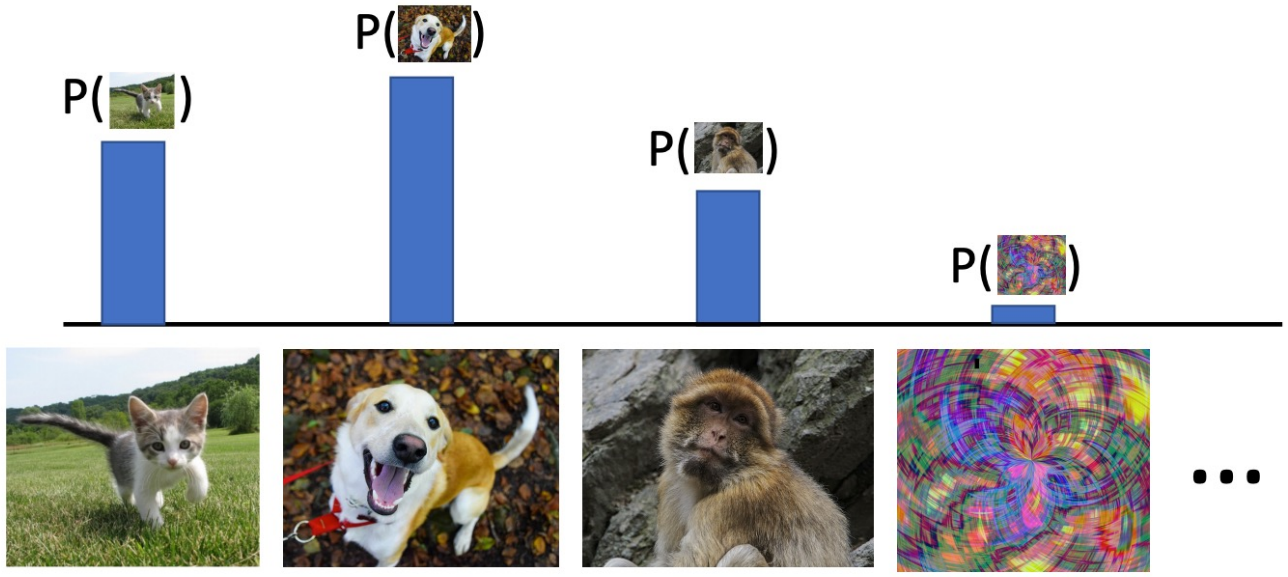
Discriminative model: No way for the model to handle unreasonable inputs; it must give label distributions for all images

Taxonomy of machine learning models

Models different probability distributions

Generative Model:

Learn p(x)



Generative model: All possible images compete with each other for probability mass

Model can “reject” unreasonable inputs by assigning them small values

Taxonomy of machine learning models

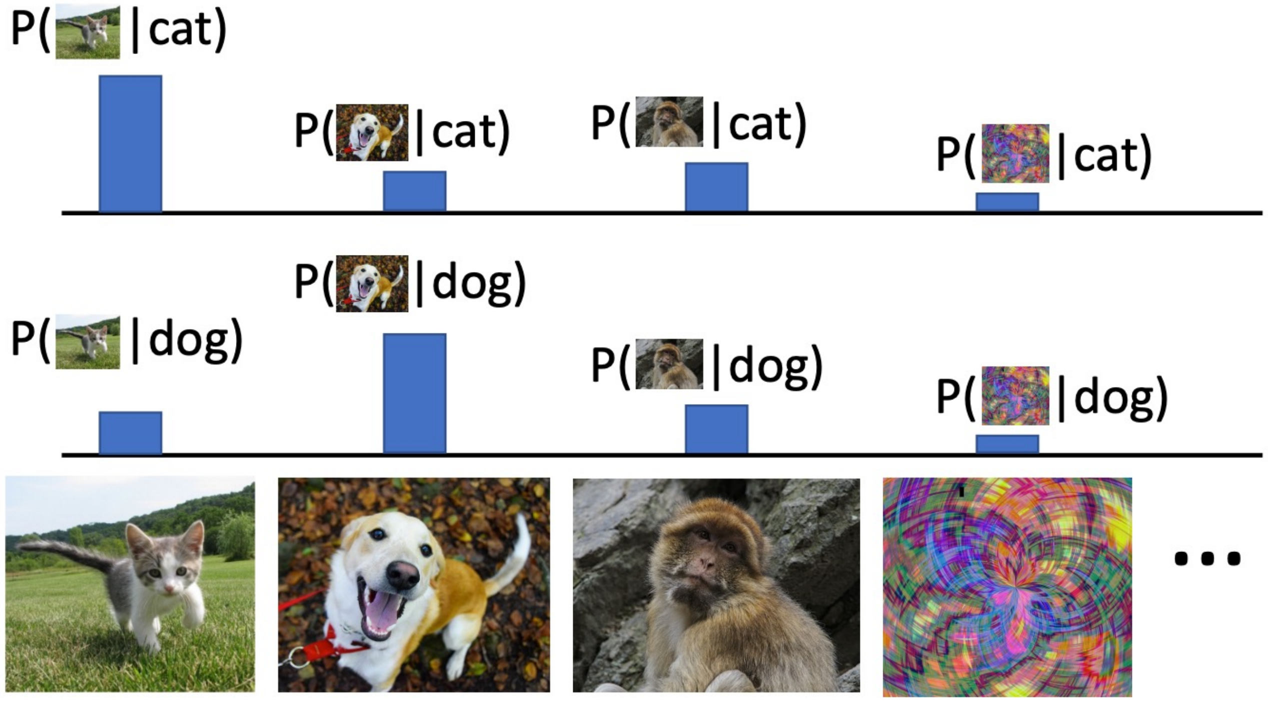
Models different probability distributions

Recall **Bayes' Rule:**

$$\underbrace{\boxed{P(x \mid y)}}_{\text{Conditional Generative Model}} = \frac{\overbrace{\boxed{P(y \mid x)}}^{\text{Discriminative Model}}}{\underbrace{\boxed{P(y)}}_{\text{Prior over labels}}} \overbrace{\boxed{P(x)}}^{\text{(Unconditional) Generative Model}}$$

We can build a conditional generative model from other components!

Conditional Generative Model:
Learn p(x|y)



Conditional Generative Model: Each possible label induces a competition among all images

Taxonomy of machine learning models

Models different probability distributions

Discriminative models:
Learn p(y|x) → Assign labels to data; Feature learning (with labels)

Generative Model:
Learn p(x) → Detect outliers; Feature learning (without labels); Sample to generate new data

Conditional Generative Model:
Learn p(x|y) → Assign labels, while rejecting outliers!; Generate new data conditioned on input labels

Slide credit: Justin Johnson (https://web.eecs.umich.edu/~justincj/teaching/eecs498/FA2020/schedule.html, L19,20)

Taxonomy of generative models

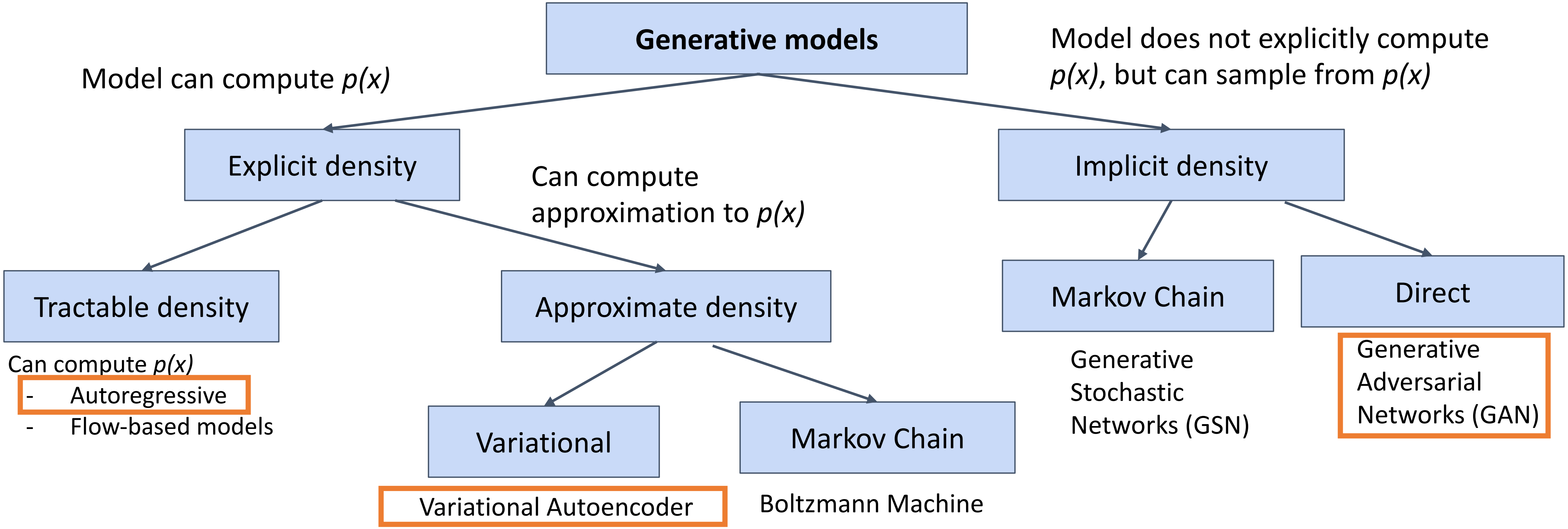


Figure copyright and adapted from Ian Goodfellow, Tutorial on Generative Adversarial Networks, 2017.

Different types of generative models

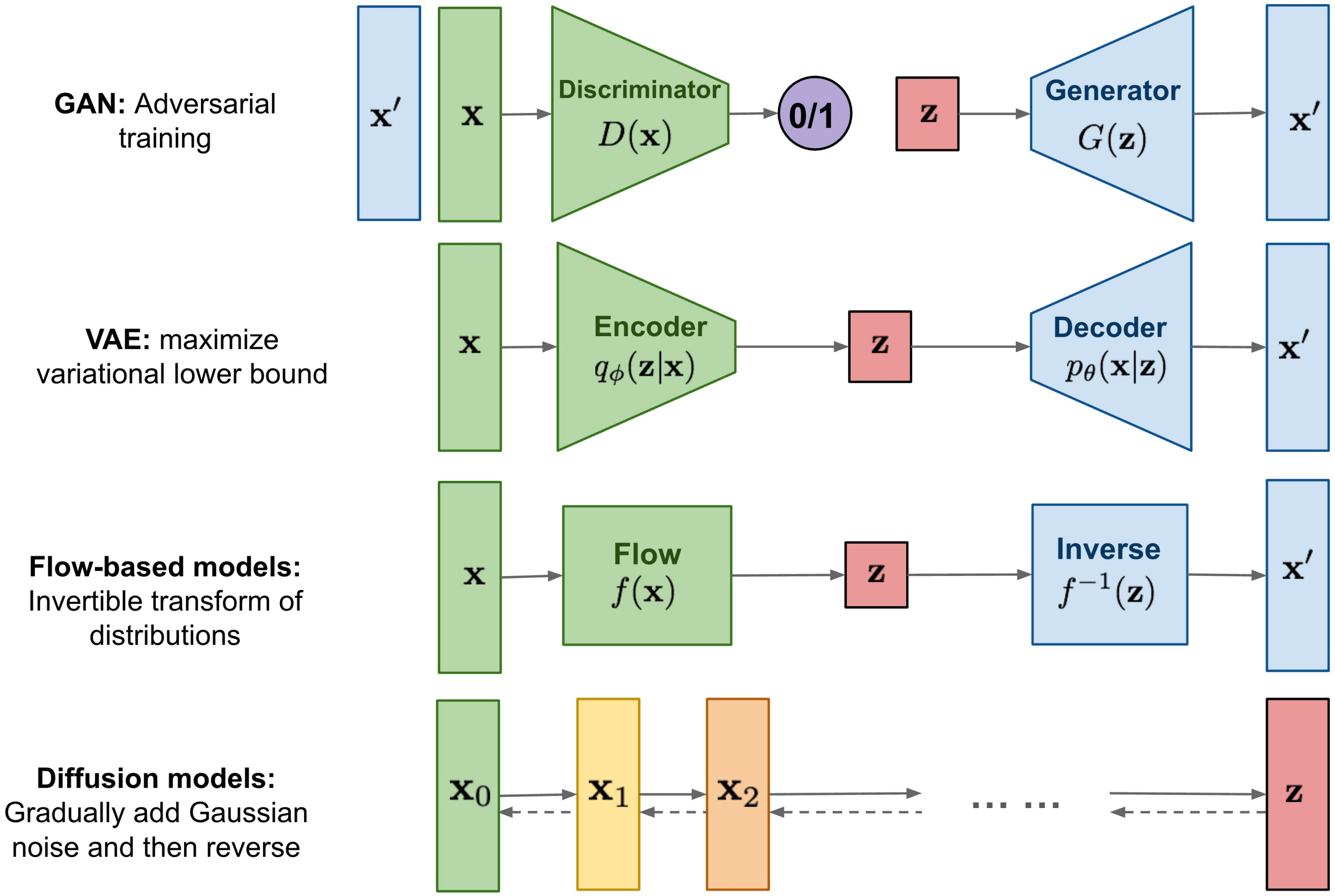


Figure credit: https://lilianweng.github.io/lil-log/2021/07/11/diffusion-models.html

Taxonomy of generative models

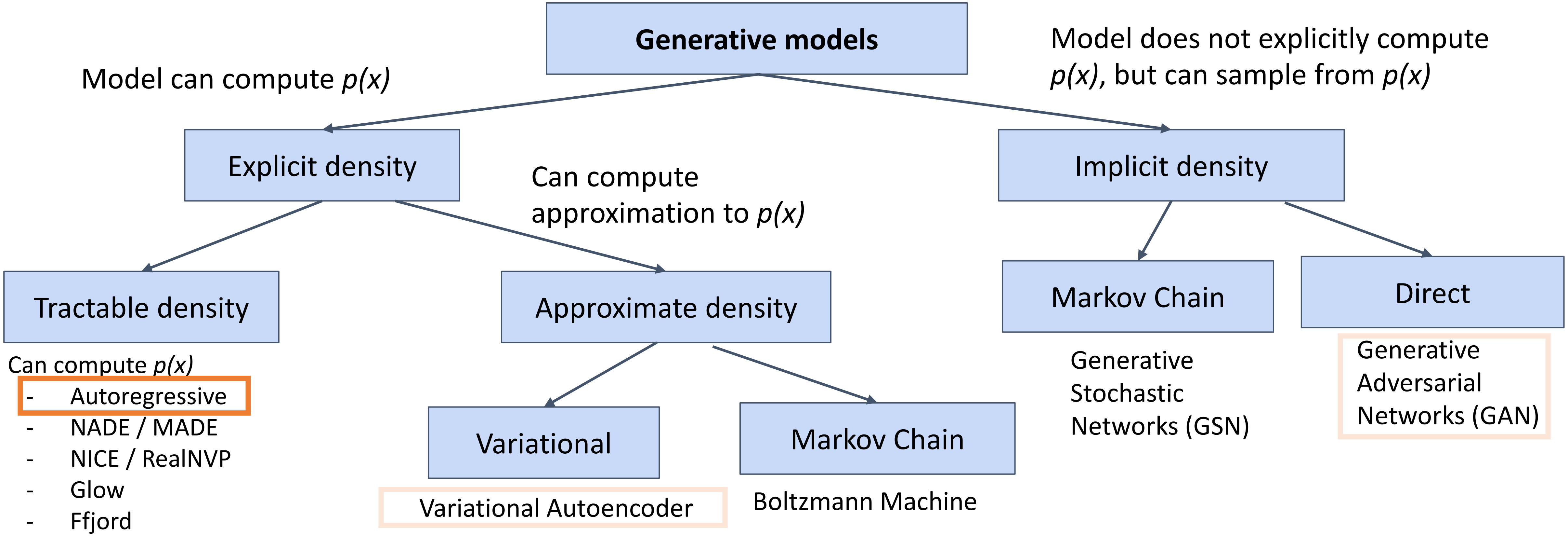


Figure copyright and adapted from Ian Goodfellow, Tutorial on Generative Adversarial Networks, 2017.

Explicit Density Estimation

Goal: Write down an explicit function for $p(x) = f(x, W)$

Given dataset $x^{(1)}, x^{(2)}, \ldots x^{(N)}$, train the model by solving:

$$W^* = \arg\max_W \prod_i p(x^{(i)})$$

Maximize probability of training data
(Maximum likelihood estimation)

$$= \arg\max_W \sum_i \log p(x^{(i)})$$

Log trick to exchange product for sum

$$= \arg\max_W \sum_i \log f(x^{(i)}, W)$$

This will be our loss function!
Train with gradient descent

Slide credit: Justin Johnson (https://web.eecs.umich.edu/~justincj/teaching/eecs498/FA2020/schedule.html, L19,20)

Explicit Density: Autoregressive models

Goal: Write down an explicit function for $p(x) = f(x, W)$

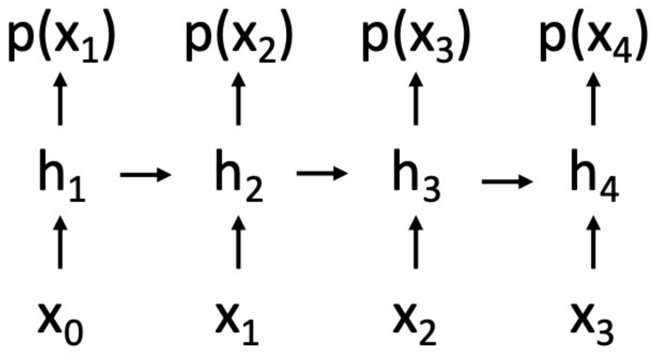
Assume x consists of multiple subparts:

$$x = (x_1, x_2, x_3, \ldots, x_T)$$

Break down probability using the chain rule:

$$\begin{aligned} p(x) &= p(x_1, x_2, x_3, \ldots, x_T) \\ &= p(x_1)p(x_2 \mid x_1)p(x_3 \mid x_1, x_2) \ldots \\ &= \prod_{t=1}^{T} p(x_t \mid x_1, \ldots, x_{t-1}) \end{aligned}$$

Probability of the next subpart given all the previous subparts



This is exactly what we had with the language modeling with RNNs and Transformers for captioning

PixelRNN

Generate image pixels one at a time, starting at the upper left corner

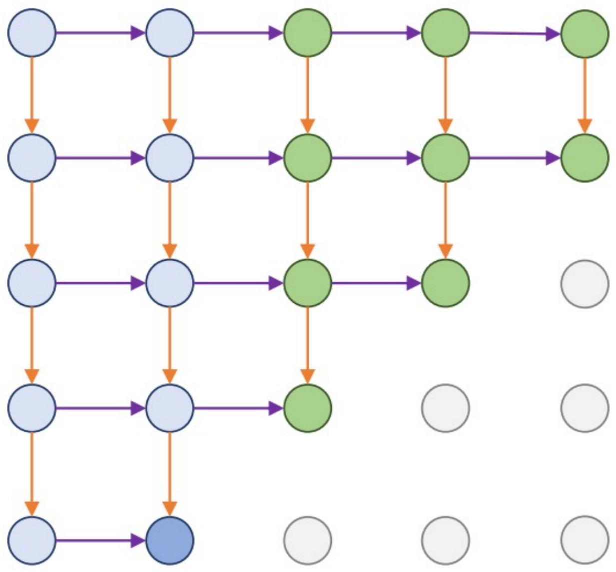
Compute a hidden state for each pixel that depends on hidden states and RGB values from the left and from above (LSTM recurrence)

$$h_{x,y} = f(h_{x-1,y}, h_{x,y-1}, W)$$

At each pixel, predict red, then blue, then green: softmax over [0, 1, ..., 255]

Each pixel depends **implicity** on all pixels above and to the left:

Problem: Very slow during both training and testing; N x N image requires 2N-1 sequential steps



Van den Oord et al, "Pixel Recurrent Neural Networks", ICML 2016

Slide credit: Justin Johnson (https://web.eecs.umich.edu/~justincj/teaching/eecs498/FA2020/schedule.html, L19,20)

PixelCNN

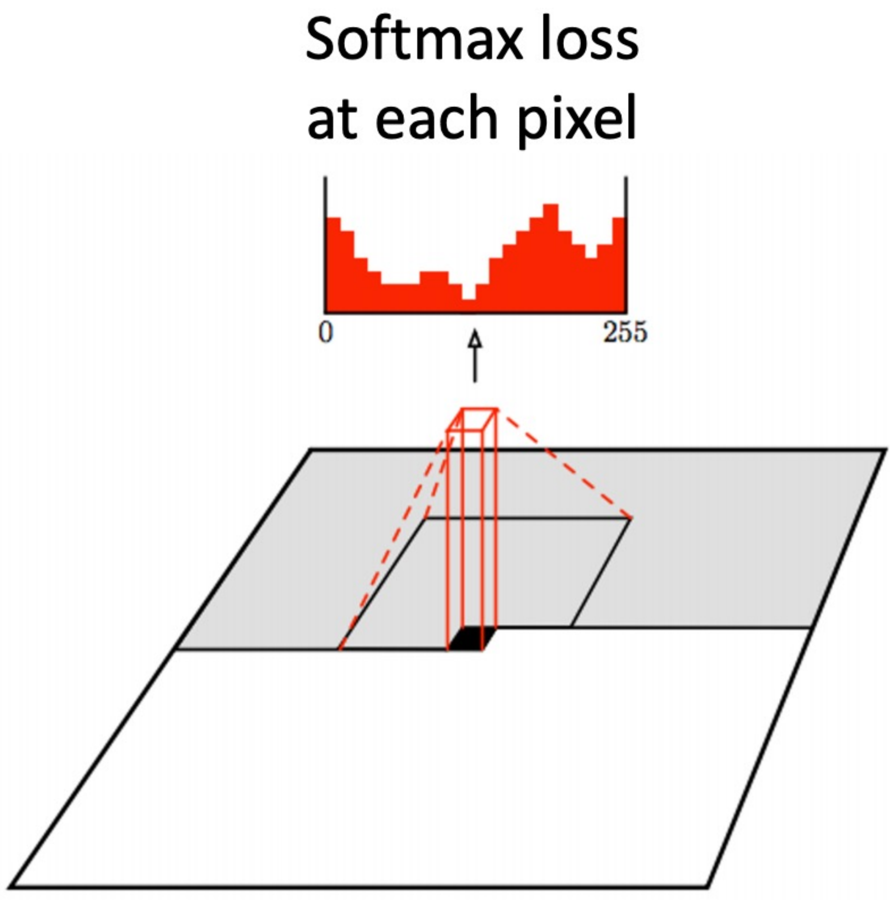
Still generate image pixels starting from corner

Dependency on previous pixels now modeled using a CNN over context region

Training: maximize likelihood of training images

Training is faster than PixelRNN (can parallelize convolutions since context region values known from training images)

Generation must still proceed sequentially => still slow



Van den Oord et al, "Conditional Image Generation with PixelCNN Decoders", NeurIPS 2016

Slide credit: Justin Johnson (https://web.eecs.umich.edu/~justincj/teaching/eecs498/FA2020/schedule.html, L19,20)

Autoregressive models: PixelRNN and PixelCNN

Pros:

- Can explicitly compute likelihood p(x)
- Explicit likelihood of training data gives good evaluation metric
- Good samples

Con:

- Sequential generation => slow

Improving PixelCNN performance

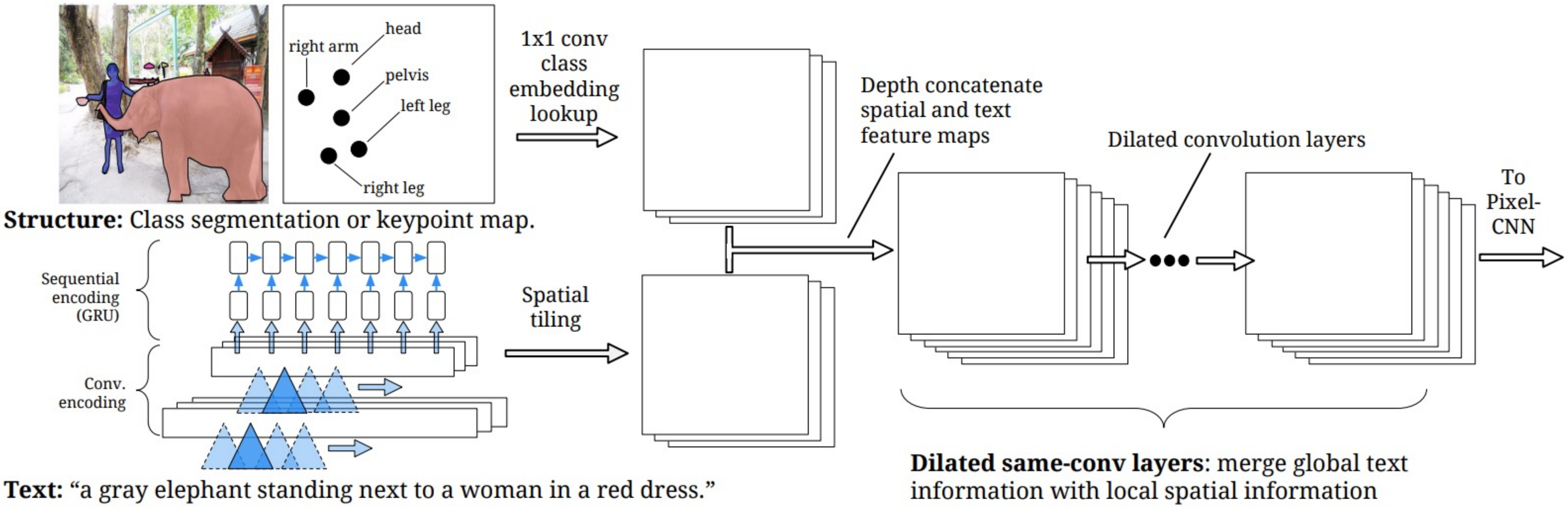
- Gated convolutional layers
- Short-cut connections
- Discretized logistic loss
- Multi-scale
- Training tricks
- Etc…

See

- Van der Oord et al. NIPS 2016
- Salimans et al. 2017 (PixelCNN++)

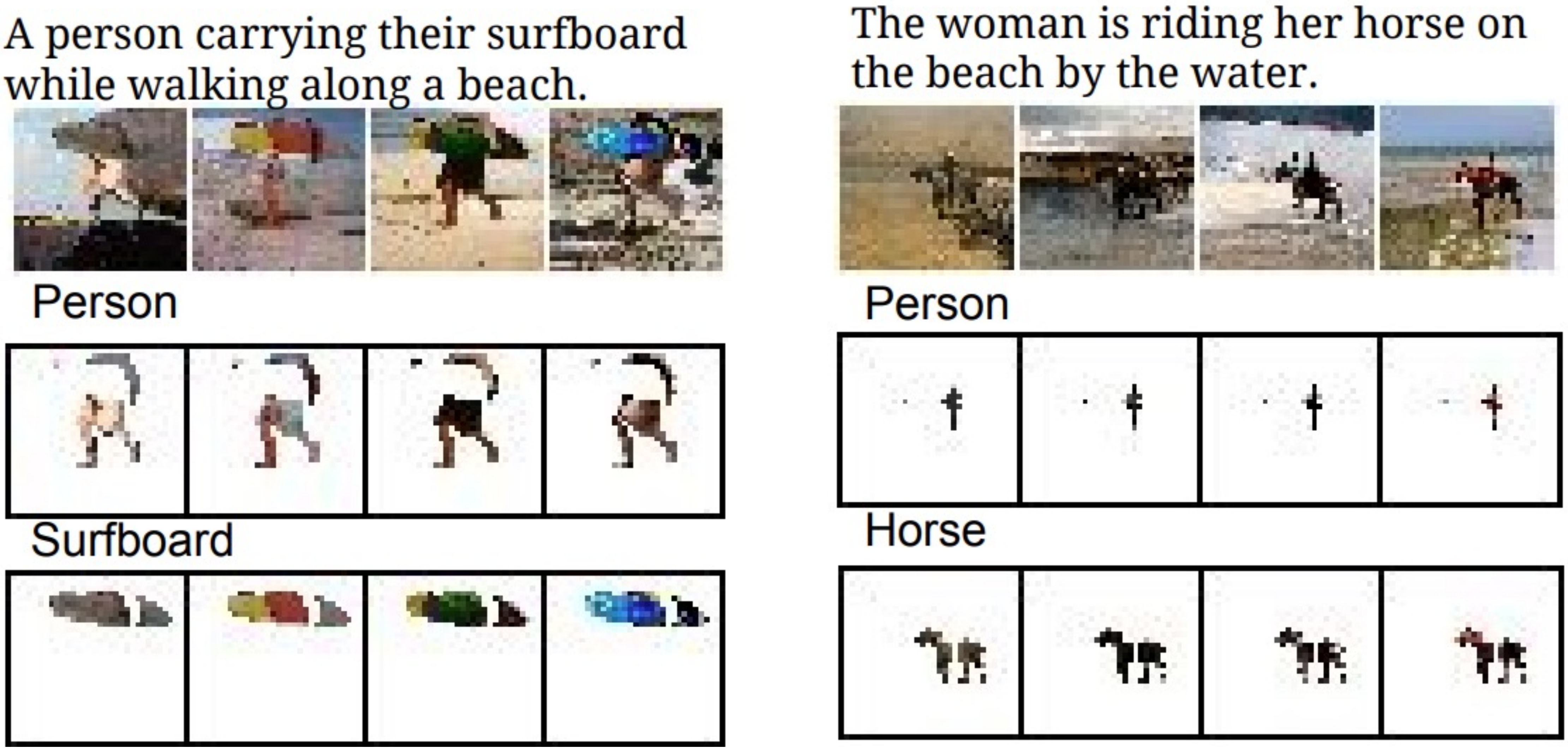
Slide credit: Justin Johnson (https://web.eecs.umich.edu/~justincj/teaching/eecs498/FA2020/schedule.html, L19,20)

Text-based image generation with PixelCNN



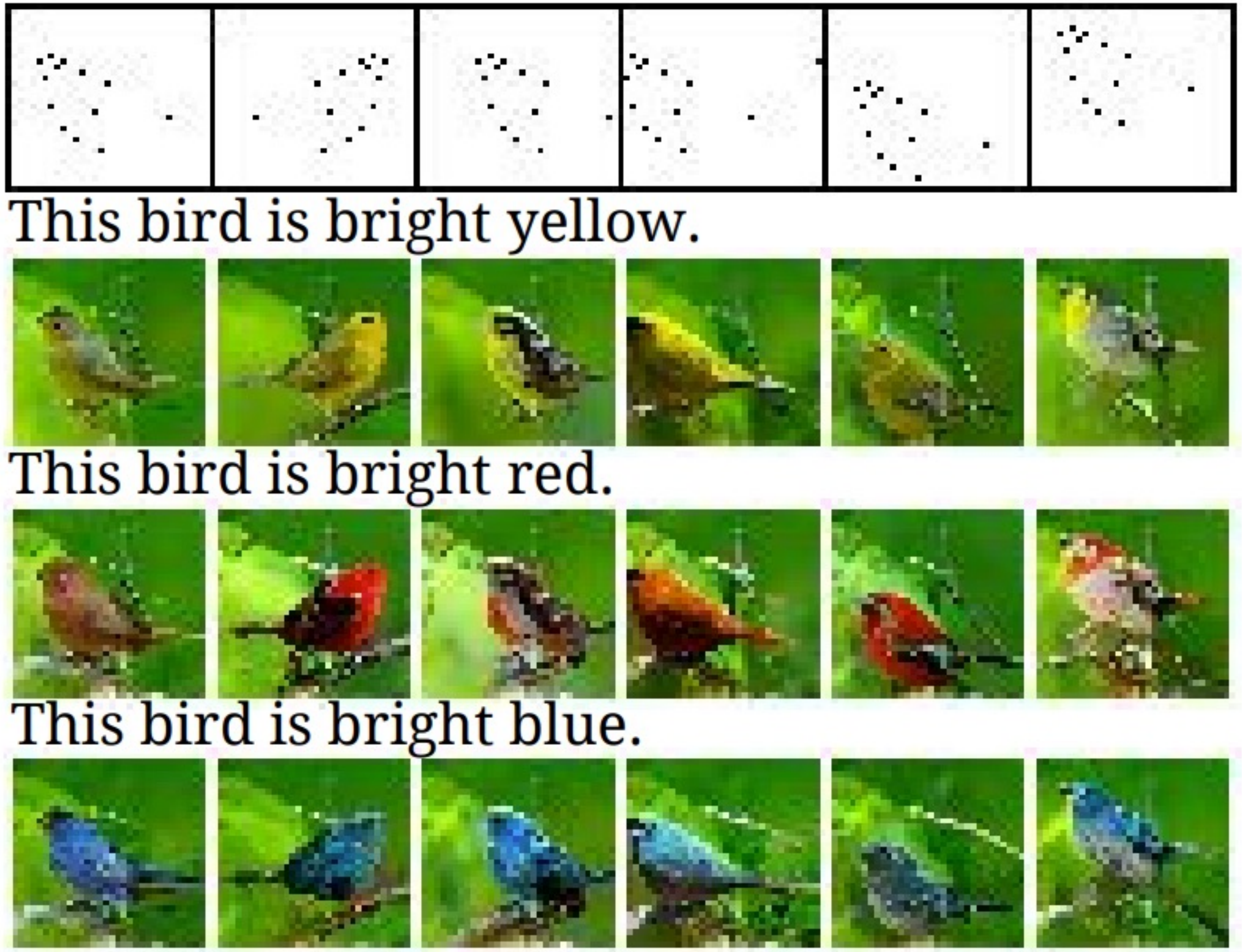
Text- and Structure-conditional PixelCNN, http://www.scottreed.info/files/txtstruct2pixel.pdf, Reed et al, 2016

Text + segmentations



Text- and Structure-conditional PixelCNN, http://www.scottreed.info/files/txtstruct2pixel.pdf, Reed et al, 2016

Text + keypoints



Text- and Structure-conditional PixelCNN, http://www.scottreed.info/files/txtstruct2pixel.pdf, Reed et al, 2016

Taxonomy of generative models

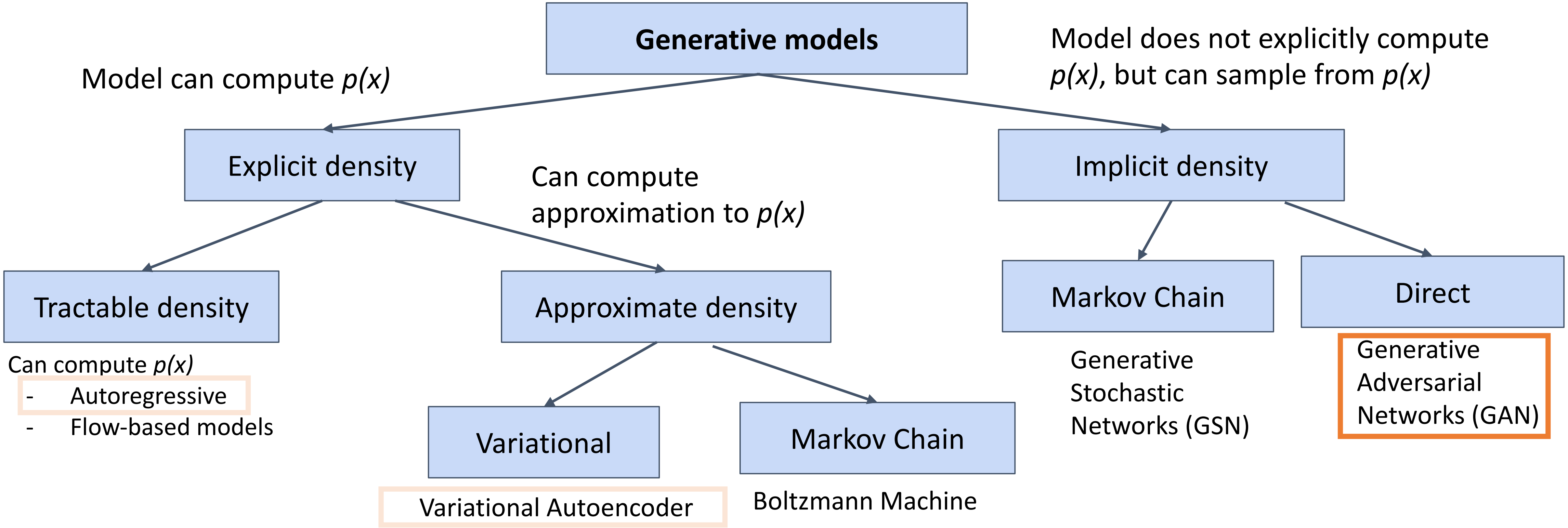
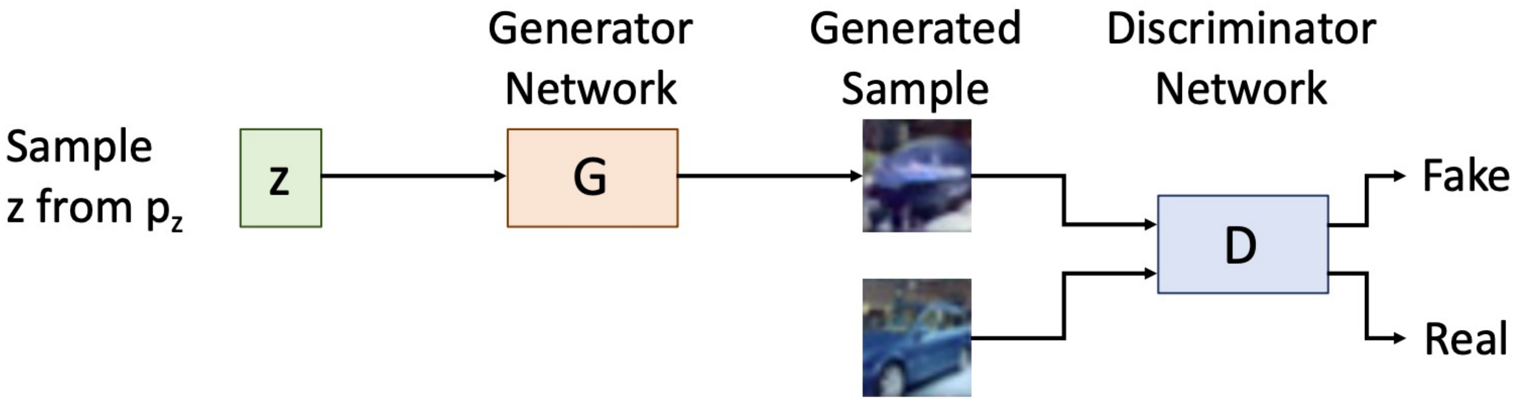


Figure copyright and adapted from Ian Goodfellow, Tutorial on Generative Adversarial Networks, 2017.

Generative Adversarial Networks (GAN)

Jointly train generator G and discriminator D with a **minimax game**

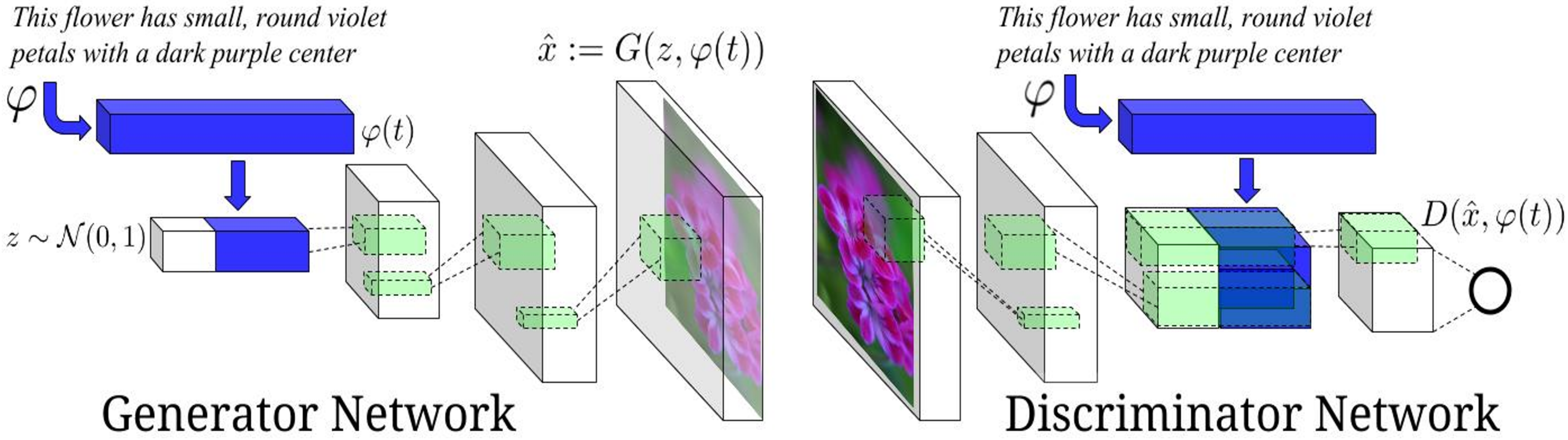
$$\min_{G} \max_{D} \left(\overbrace{E_{x\sim p_{data}}[\log D(x)]}^{\text{Discriminator wants } D(x) = 1 \text{ for real data}} + E_{z\sim p(z)} \overbrace{\underbrace{\left[\log\left(1 - D(G(z))\right)\right]}_{\text{Generator wants } D(x) = 1 \text{ for fake data}}}^{\text{Discriminator wants } D(x) = 0 \text{ for fake data}} \right)$$



Goodfellow et al, "Generative Adversarial Nets", NeurIPS 2014

Text to image with GANs

- Generator and Discriminator are alternately trained



Generative Adversarial Text to Image Synthesis, https://arxiv.org/pdf/1605.05396v2.pdf, Reed et al, ICML 2016

Text to image with GANs

- Image encoder (CNN ϕ) and text encoder (char-CNN-RNN φ) are pre-trained to produce a joint embedding where the embedded representations can be used to predict the class label of the image

$$\frac{1}{N}\sum_{n=1}^{N}\Delta(y_n, f_v(v_n)) + \Delta(y_n, f_t(t_n))$$

$$f_v(v) = \underset{y\in\mathcal{Y}}{\arg\max}\ \mathbb{E}_{t\sim\mathcal{T}(y)}[\phi(v)^T\varphi(t))]$$

$$f_t(t) = \underset{y\in\mathcal{Y}}{\arg\max}\ \mathbb{E}_{v\sim\mathcal{V}(y)}[\phi(v)^T\varphi(t))]$$

Generative Adversarial Text to Image Synthesis, https://arxiv.org/pdf/1605.05396v2.pdf, Reed et al, ICML 2016

Datasets

- CUB-200 (Birds)
  - 11,788 images of birds from 200 categories
- Oxford-102 (Flowers)
  - 8,189 images of flowers from 102 categories
- MSCOCO
  - 330K images
- 5 captions per image

Caltech-UCSD-Birds (CUB) 200

an all black bird with a distinct thick, rounded bill.

this small bird has a yellow breast, brown crown, and black superciliary

this flower is white and pink in color, with petals that have veins.

bright droopy yellow petals with burgundy streaks, and a yellow stigma.

The man at bat readies to swing at the pitch while the umpire looks on.

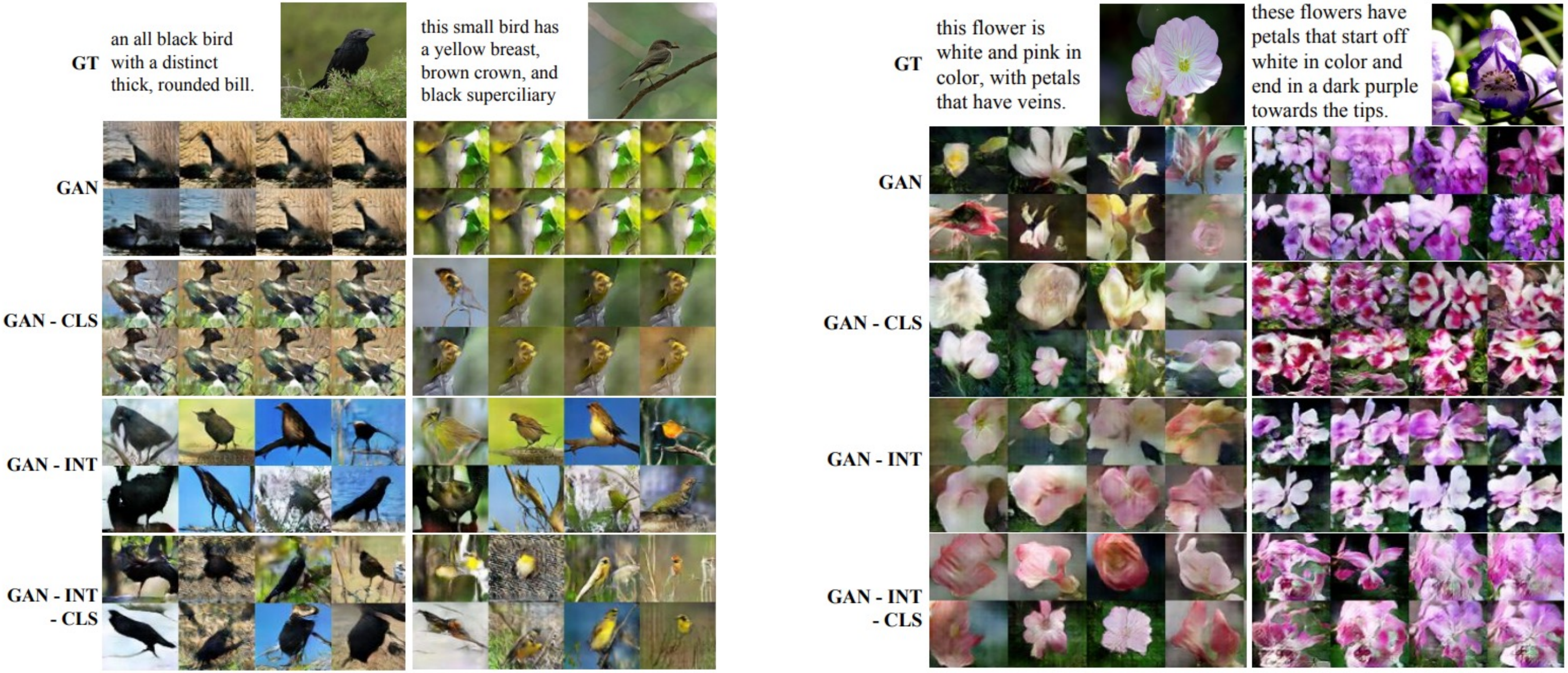
Bunk bed with a narrow shelf sitting underneath it.

Text to image with GANS: Results

- CLS: Add discriminator to distinguish if (image,text) match or not

  (real image, right text), (real image, wrong text), (fake image, right text)

- INT: Add interpolated text embeddings (fake additional text embeddings)



Generative Adversarial Text to Image Synthesis, https://arxiv.org/pdf/1605.05396v2.pdf, Reed et al, ICML 2016

Text to image with GANS: Results

Very low res!
64 x 64

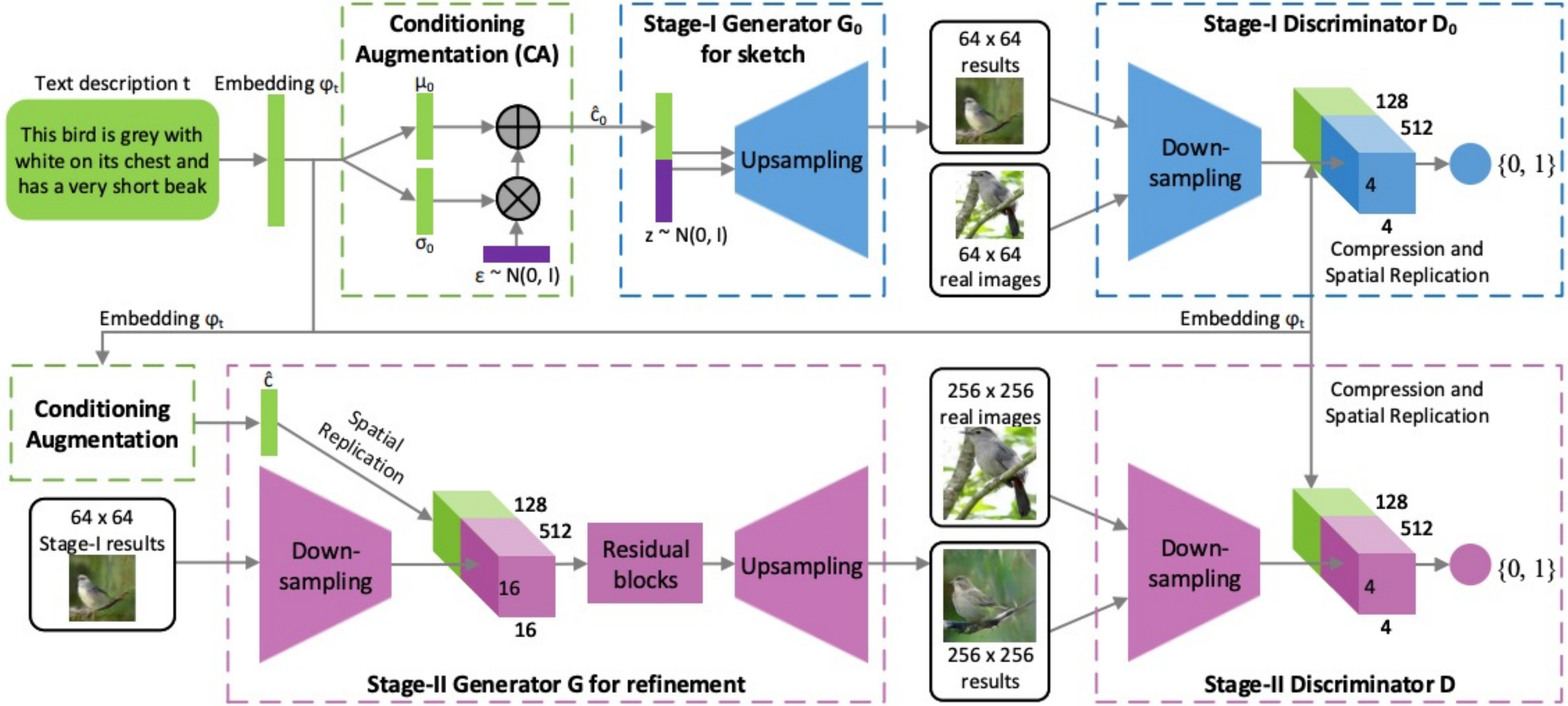
Follow up work:
128 x 128

Still low res!

Generative Adversarial Text to Image Synthesis, https://arxiv.org/pdf/1605.05396v2.pdf, Reed et al, ICML 2016

StackGAN

Generate low resolution, and then pass through another GAN for improved resolution

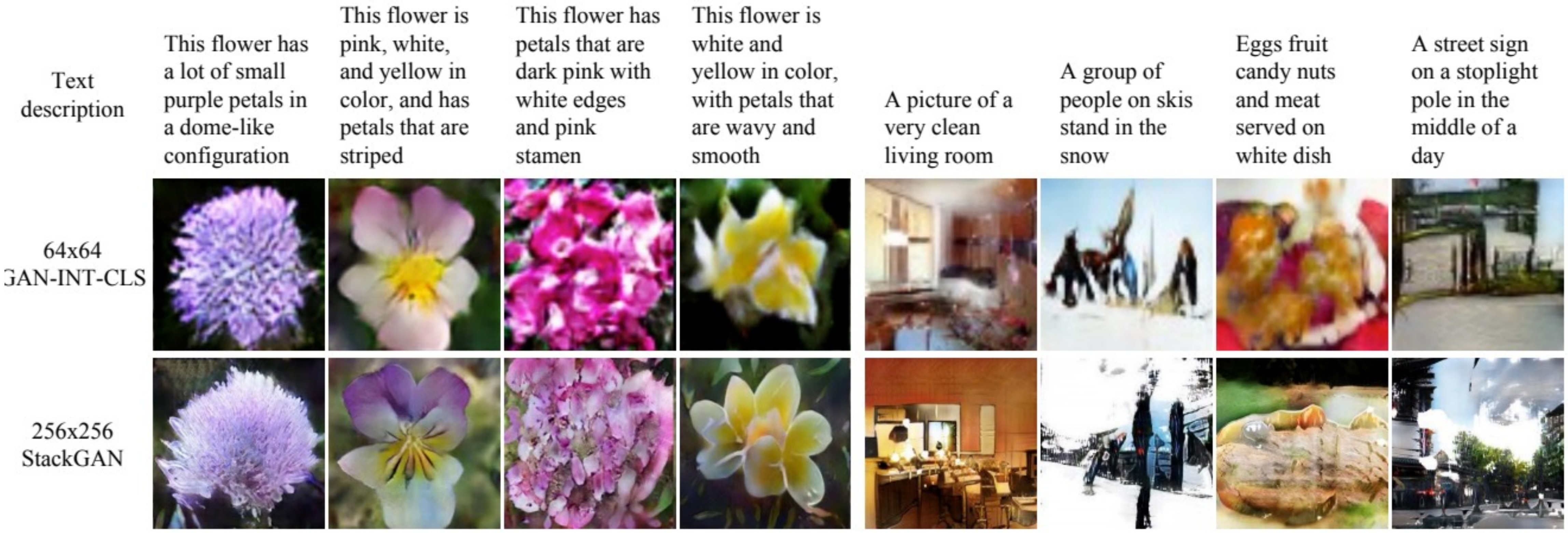


StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks
https://arxiv.org/pdf/1612.03242.pdf, Zhang et al, ICCV 2017

StackGAN: Results

StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks
https://arxiv.org/pdf/1612.03242.pdf, Zhang et al, ICCV 2017

StackGAN: Results



StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks
https://arxiv.org/pdf/1612.03242.pdf, Zhang et al, ICCV 2017

StackGAN: Evaluation

- Inception Score: $I = \exp(\mathbb{E}_{\boldsymbol{x}} D_{KL}(p(y|\boldsymbol{x}) \,||\, p(y)))$
  - Use inception model to predict class y
  - Want good models to generate diverse but meaningful images
  - Large distance between marginal prior (of labels) and conditional prior
- Human rank images generated by models

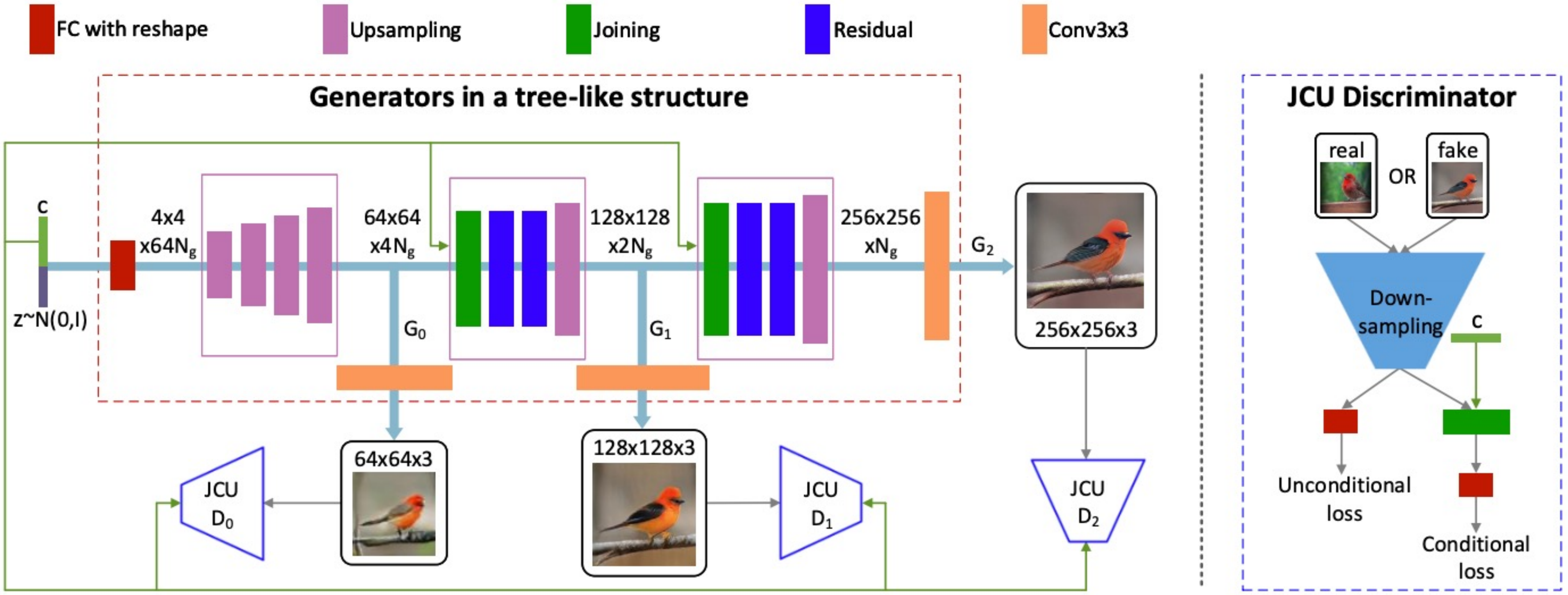
Metric	Dataset	GAN-INT-CLS	GAWWN	Our StackGAN
Inception score	CUB	$2.88 \pm .04$	$3.62 \pm .07$	**$3.70 \pm .04$**
	Oxford	$2.66 \pm .03$	/	**$3.20 \pm .01$**
	COCO	$7.88 \pm .07$	/	**$8.45 \pm .03$**
Human rank	CUB	$2.81 \pm .03$	$1.99 \pm .04$	**$1.37 \pm .02$**
	Oxford	$1.87 \pm .03$	/	**$1.13 \pm .03$**
	COCO	$1.89 \pm .04$	/	**$1.11 \pm .03$**

StackGAN: Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks
https://arxiv.org/pdf/1612.03242.pdf, Zhang et al, ICCV 2017

StackGAN++

Joint Discriminator
- if image is real/fake (unconditional loss)
- if text+image match (conditional loss)

Generalization of StackGAN (Multiscale)



Color constancy regularization

StackGAN++: Realistic Image Synthesis with Stacked Generative Adversarial Networks
https://arxiv.org/pdf/1710.10916.pdf, Zhang et al, TPAMI 2018

StackGAN++

- Generalization of StackGAN (arbitrary number of Generators/Discriminators)
- Color constancy regularization
- Joint Discriminator (similar to +CLS from Reed et al)
  - if image is real/fake (unconditional loss)
  - if text+image match (conditional loss)
- Alternately train generator and discriminator

$$\mathcal{L}_{G_i} = \underbrace{-\frac{1}{2}\mathbb{E}_{\hat{x}_i \sim p_{G_i}}[\log(D_i(\hat{x}_i)]}_{\text{unconditional loss}} \underbrace{-\frac{1}{2}\mathbb{E}_{\hat{x}_i \sim p_{G_i}}[\log(D_i(\hat{x}_i, \overline{e})]}_{\text{conditional loss}},$$

$$\mathcal{L}_{D_i} = \underbrace{-\frac{1}{2}\mathbb{E}_{x_i \sim p_{data_i}}[\log D_i(x_i)] - \frac{1}{2}\mathbb{E}_{\hat{x}_i \sim p_{G_i}}[\log(1 - D_i(\hat{x}_i)]}_{\text{unconditional loss}} + \underbrace{-\frac{1}{2}\mathbb{E}_{x_i \sim p_{data_i}}[\log D_i(x_i, \overline{e})] - \frac{1}{2}\mathbb{E}_{\hat{x}_i \sim p_{G_i}}[\log(1 - D_i(\hat{x}_i, \overline{e})]}_{\text{conditional loss}},$$

StackGAN++: Realistic Image Synthesis with Stacked Generative Adversarial Networks
https://arxiv.org/pdf/1710.10916.pdf, Zhang et al, TPAMI 2018

StackGAN++: Results

- FID (Frechet Inception distance): measures distance between generated and real distribution

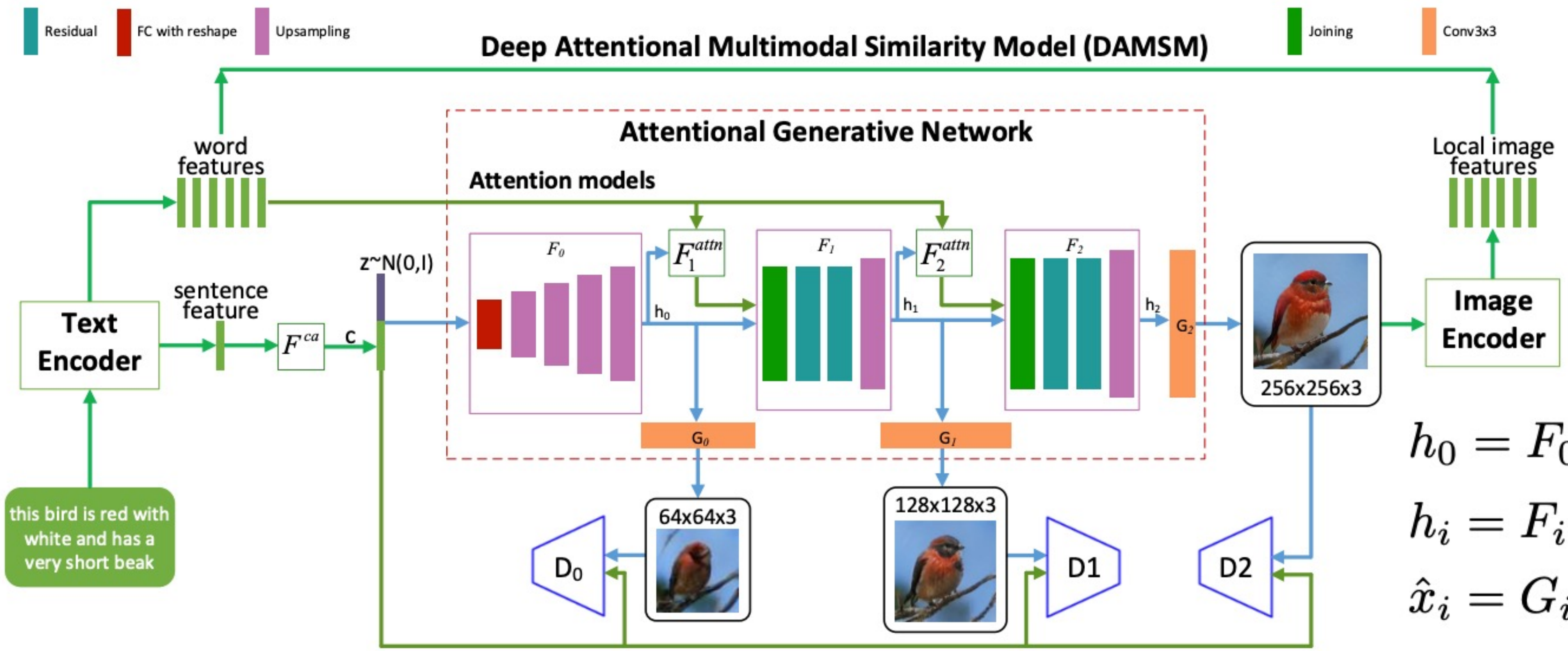
Metric	CUB			Oxford		COCO	
	GAN-INT-CLS	GAWWN	Our StackGAN-v1	GAN-INT-CLS	Our StackGAN-v1	GAN-INT-CLS	Our StackGAN-v1
FID ↓	68.79	67.22	**51.89**	79.55	**55.28**	**60.62**	74.05
FID* ↓	68.79	53.51	**35.11**	79.55	**43.02**	60.62	**33.88**
IS ↑	2.88 ± .04	3.62 ± .07	**3.70 ± .04**	2.66 ± .03	**3.20 ± .01**	7.88 ± .07	**8.45 ± .03**
IS* ↑	2.88 ± .04	**3.10 ± .03**	3.02 ± .03	2.66 ± .03	**2.73 ± .03**	7.88 ± .07	**8.35 ± .11**
HR ↓	2.76 ± .01	1.95 ± .02	**1.29 ± .02**	1.84 ± .02	**1.16 ± .02**	1.82 ± .03	**1.18 ± .03**

Dataset		CUB	Oxford-102	COCO	LSUN-bedroom	LSUN-church	ImageNet-dog	ImageNet-cat
FID ↓	StackGAN-v1	51.89	55.28	**74.05**	91.94	57.20	89.21	58.73
	StackGAN-v2	**15.30**	**48.68**	81.59	**35.61**	**25.36**	**44.54**	**28.59**
IS ↑	StackGAN-v1	3.70 ± .04	3.20 ± .01	**8.45 ± .03**	**3.59 ± .05**	**2.87 ± .05**	8.84 ± .08	**4.77 ± .06**
	StackGAN-v2	**4.04 ± .05**	**3.26 ± .01**	8.30 ± .10	3.02 ± .04	2.38 ± .03	**9.55 ± .11**	4.23 ± .05
HR ↓	StackGAN-v1	1.81 ± .02	1.70 ± .03	**1.45 ± .04**	1.95 ± .01	1.86 ± .02	1.90 ± .01	1.88 ± .02
	StackGAN-v2	**1.19 ± .02**	**1.30 ± .03**	1.55 ± .05	**1.05 ± .01**	**1.14 ± .02**	**1.10 ± .01**	**1.12 ± .02**

StackGAN++: Realistic Image Synthesis with Stacked Generative Adversarial Networks
https://arxiv.org/pdf/1710.10916.pdf, Zhang et al, TPAMI 2018

AttnGAN

- Attention based similarity matching of image and text that tries to align regions of the image to words in the text
- m generators (G_i), each taking hidden state h_i to produce image $\hat{x}_i$



$$h_0 = F_0(z, F^{ca}(\bar{e}));$$

$$h_i = F_i(h_{i-1}, F_i^{attn}(e, h_{i-1}))$$

$$\hat{x}_i = G_i(h_i).$$

AttnGAN: Fine-Grained Text to Image Generation with Attentional Generative Adversarial Networks
https://arxiv.org/pdf/1711.10485.pdf, Xu et al, CVPR 2018

AttnGAN:

- Attention based similarity matching of image and text that tries to align regions of the image to words in the text
- m generators (G_i), each taking hidden state h_i to produce image $\hat{x}_i$
- Total Loss:

$$\mathcal{L} = \mathcal{L}_G + \lambda \mathcal{L}_{DAMSM}, \quad \text{where } \mathcal{L}_G = \sum_{i=0}^{m-1} \mathcal{L}_{G_i}.$$

- Main contribution:
  - Semi-supervised training to match image regions to text
  - Attention-based match score $R(Q, D)$of image (Q) to text (D) based on attention-based match of words to regions in the image
  - Train to optimize match based on words (w) and sentences (s)
  - Estimate probability of text given image and vice versa

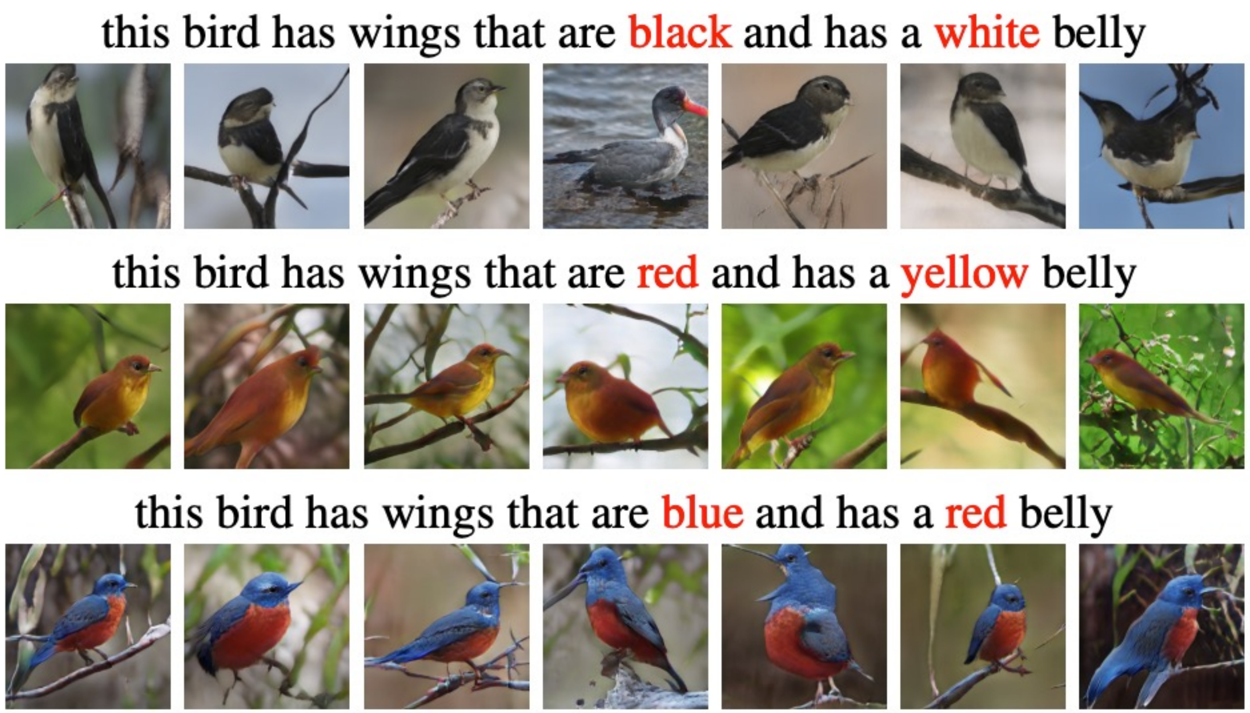
$$\mathcal{L}_{DAMSM} = \mathcal{L}_1^w + \mathcal{L}_2^w + \mathcal{L}_1^s + \mathcal{L}_2^s.$$

$$P(D_i|Q_i) = \frac{\exp(\gamma_3 R(Q_i, D_i))}{\sum_{j=1}^{M} \exp(\gamma_3 R(Q_i, D_j))}$$

$$\mathcal{L}_1^w = -\sum_{i=1}^{M} \log P(D_i|Q_i), \quad \mathcal{L}_2^w = -\sum_{i=1}^{M} \log P(Q_i|D_i),$$

AttnGAN: Fine-Grained Text to Image Generation with Attentional Generative Adversarial Networks
https://arxiv.org/pdf/1711.10485.pdf, Xu et al, CVPR 2018

AttnGAN: Results



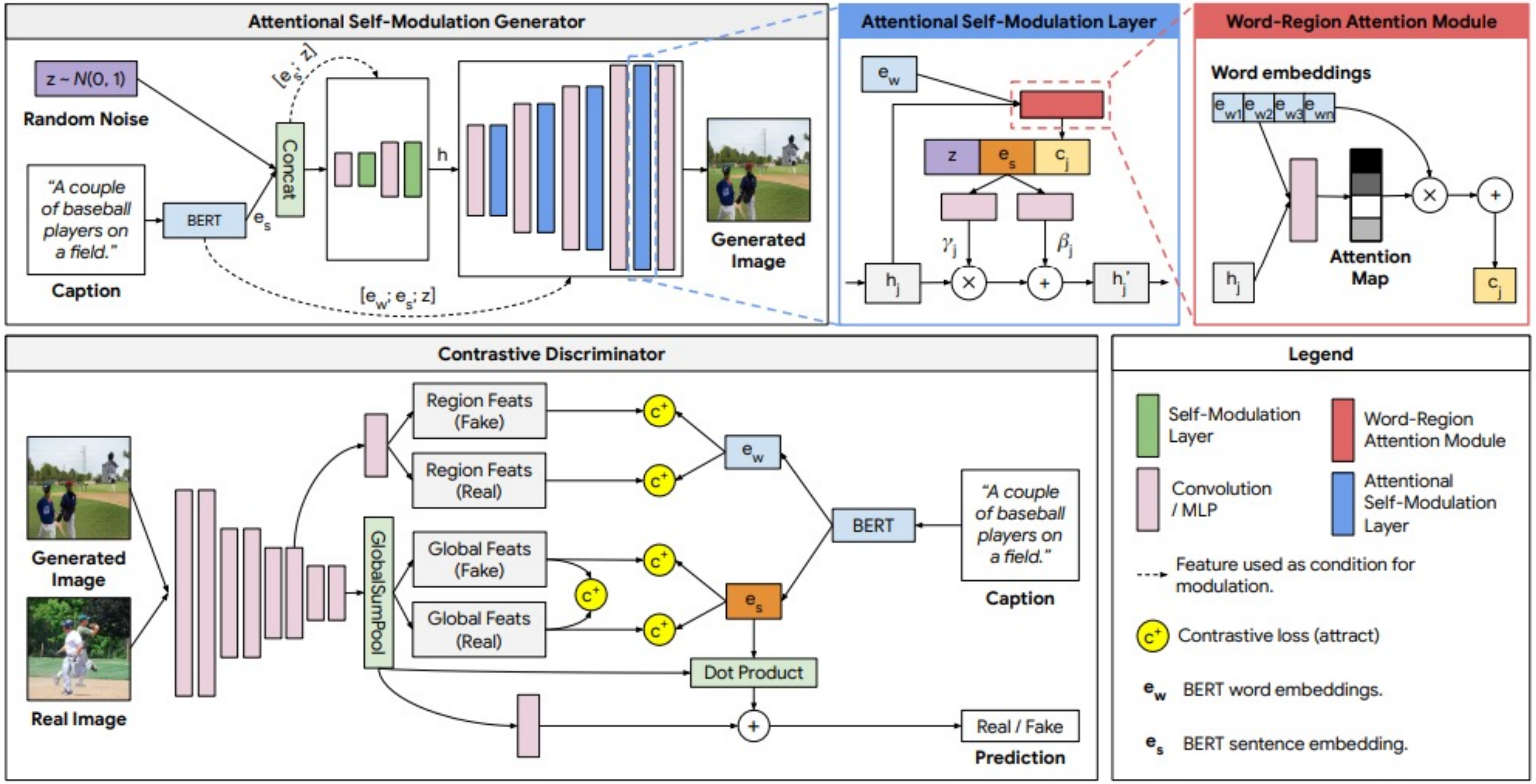
Inception Scores

Dataset	GAN-INT-CLS [20]	GAWWN [18]	StackGAN [31]	StackGAN-v2 [32]	PPGN [16]	Our AttnGAN
CUB	$2.88 \pm .04$	$3.62 \pm .07$	$3.70 \pm .04$	$3.82 \pm .06$	/	$\mathbf{4.36 \pm .03}$
COCO	$7.88 \pm .07$	/	$8.45 \pm .03$	/	$9.58 \pm .21$	$\mathbf{25.89 \pm .47}$

AttnGAN: Fine-Grained Text to Image Generation with Attentional Generative Adversarial Networks
https://arxiv.org/pdf/1711.10485.pdf, Xu et al, CVPR 2018

Cross-Modal Contrastive Learning

- GAN with contrastive losses



Cross-Modal Contrastive Learning for Text-to-Image Generation
https://arxiv.org/pdf/2101.04702.pdf, Zhang et al, CVPR 2021

Taxonomy of generative models

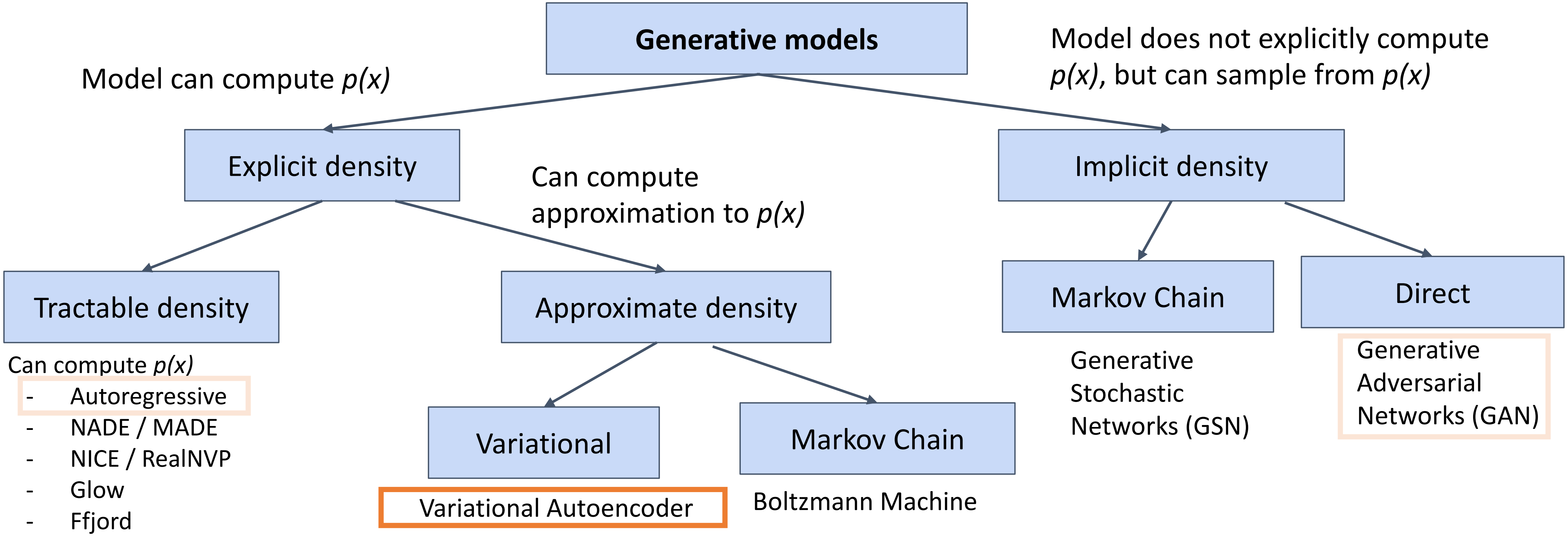


Figure copyright and adapted from Ian Goodfellow, Tutorial on Generative Adversarial Networks, 2017.

Variational Autoencoders

- PixelRNN/PixelCNN explicitly parameterizes density function with a neural network, so we can train to maximize likelihood of training data

$$p_\theta(x) = \prod_{t=1}^{T} p_\theta(x_t | x_1, \ldots, x_{t-1})$$

Assume data can be broken into subparts!

What if we don't make this assumption?

- Variational Autoencoders (VAE) use an intractable density that we cannot explicitly compute or optimize

- But we will be able to directly optimize a lower bound on the density

Slide credit: Justin Johnson (https://web.eecs.umich.edu/~justincj/teaching/eecs498/FA2020/schedule.html, L19,20)

(Regular, non-variational) Autoencoders

Loss: L2 distance between input and reconstructed data.

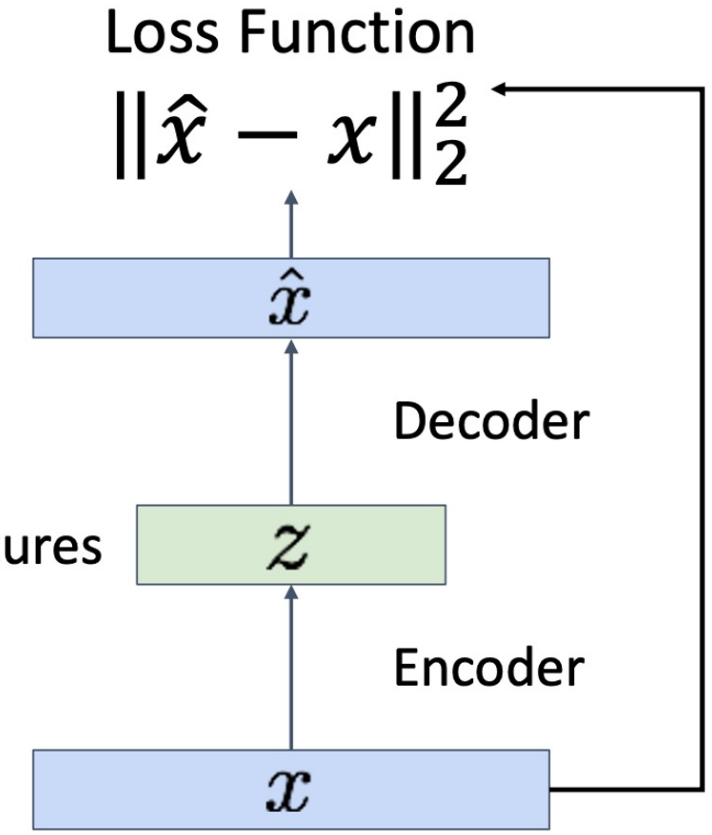
Does not use any labels! Just raw data!

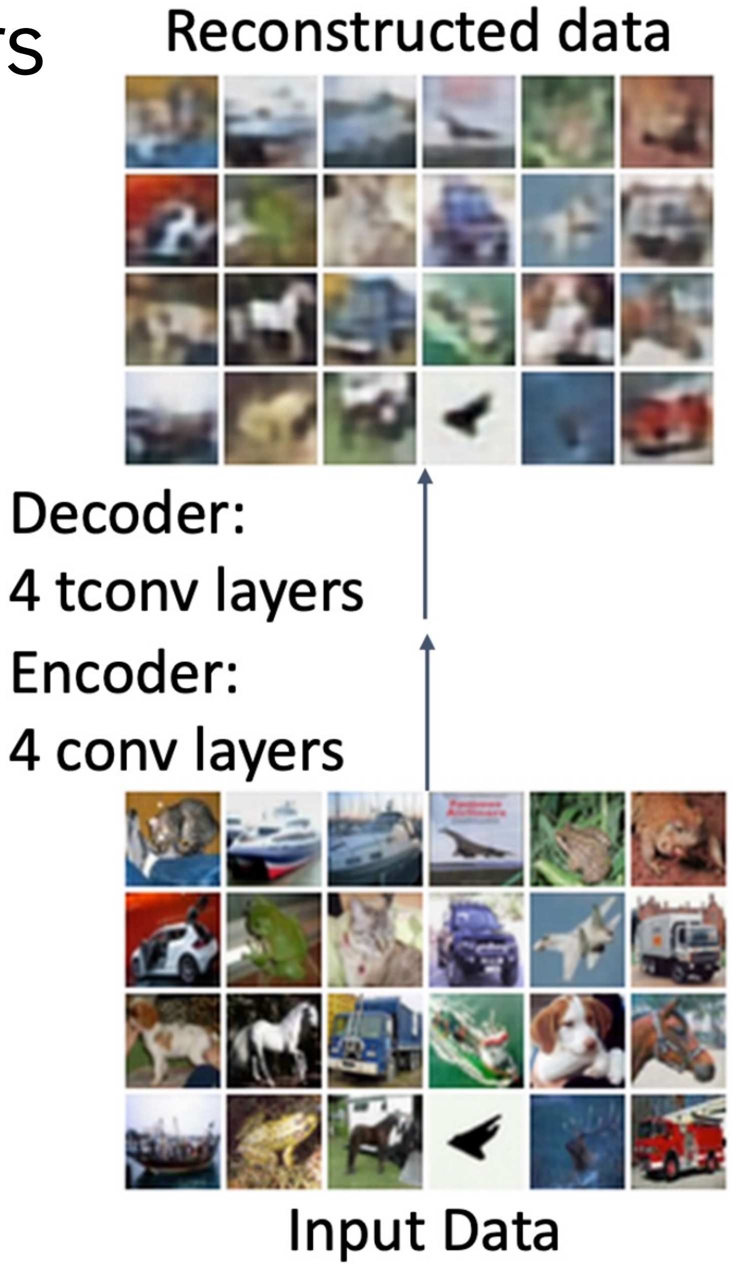
Reconstructed input data

Features need to be **lower dimensional** than the data

Features

Input data

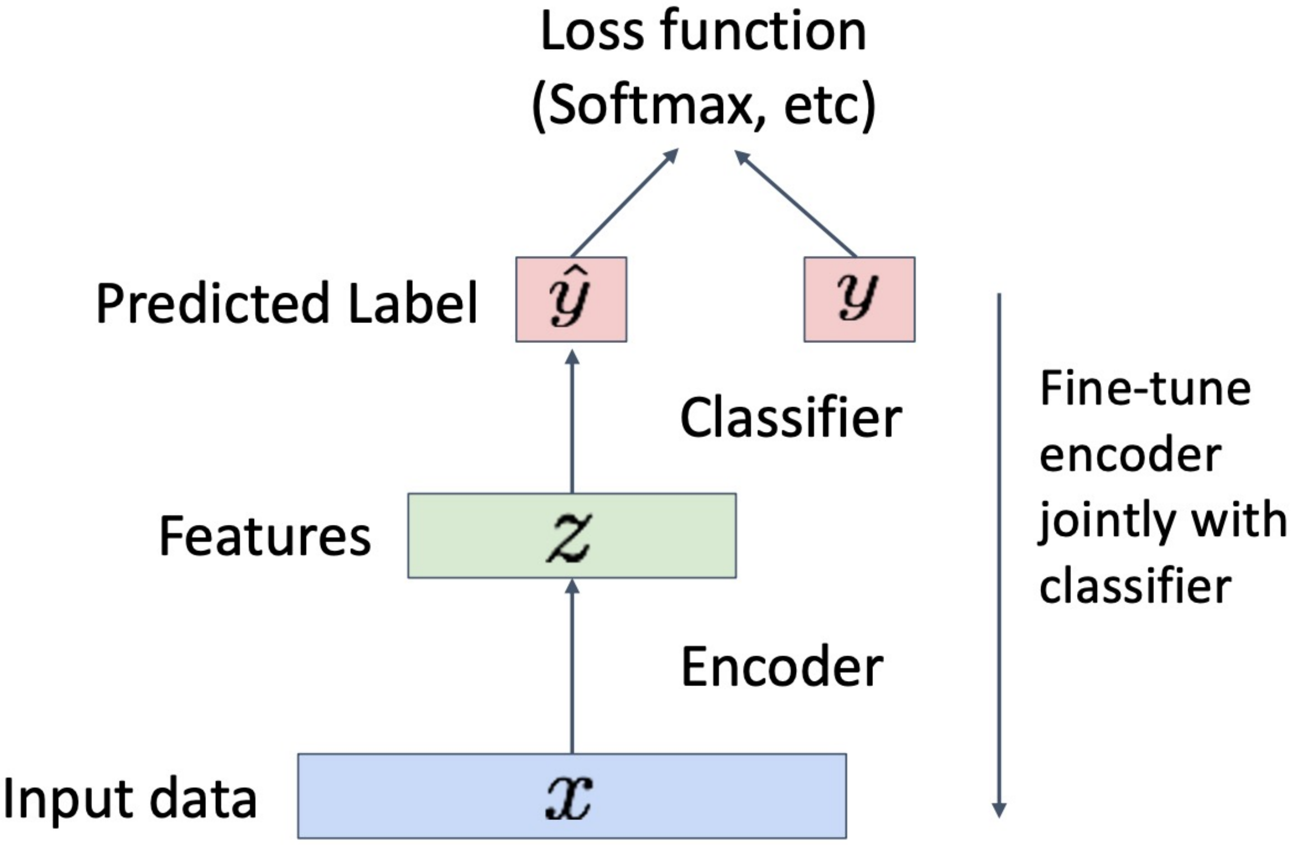




Slide credit: Justin Johnson (https://web.eecs.umich.edu/~justincj/teaching/eecs498/FA2020/schedule.html, L19,20)

(Regular, non-variational) Autoencoders

After training, **throw away decoder** and use encoder for a downstream task

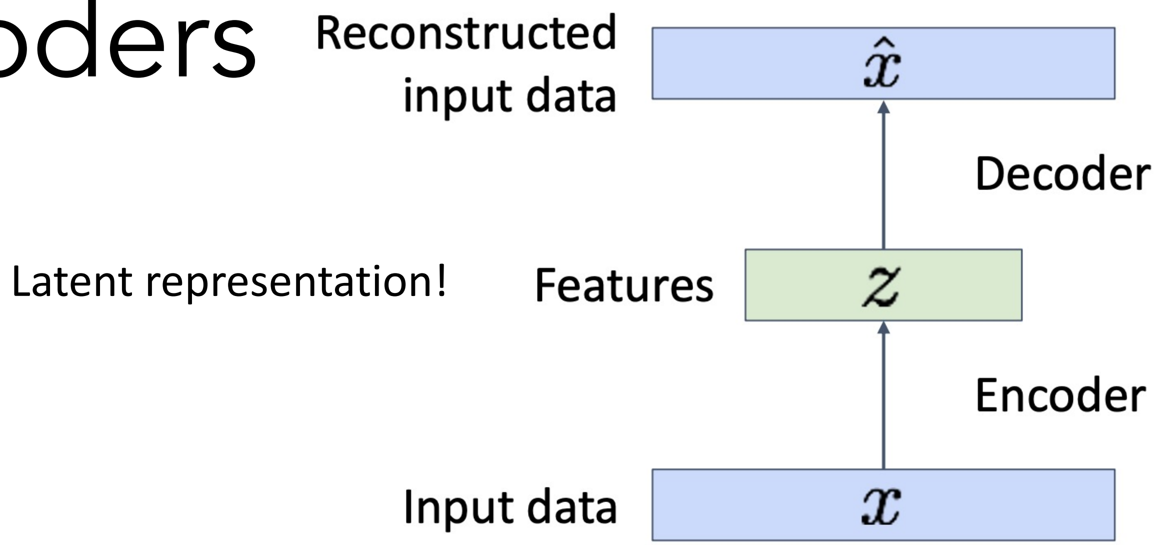


Encoder can be used to initialize a **supervised** model

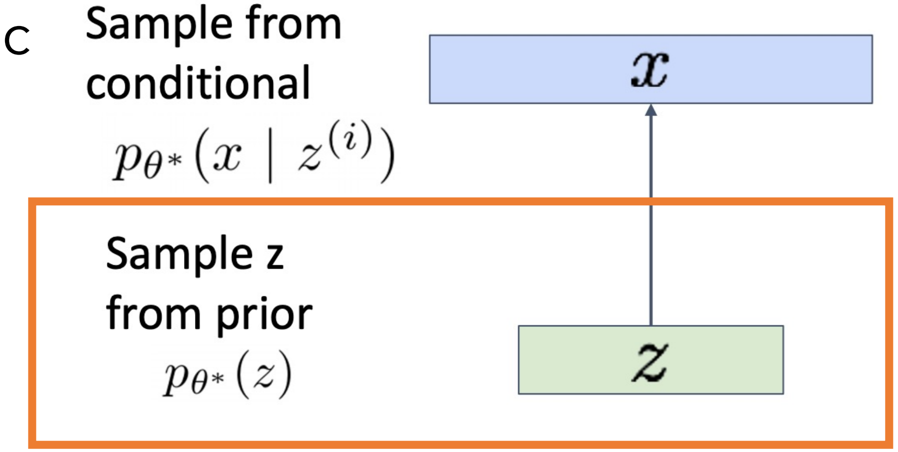
Train for final task (sometimes with small data)

Variational Autoencoders

- Autoencoders
  - Not probabilistic
  - No sampling



- Variational
  - Probabilistic



How to sample?

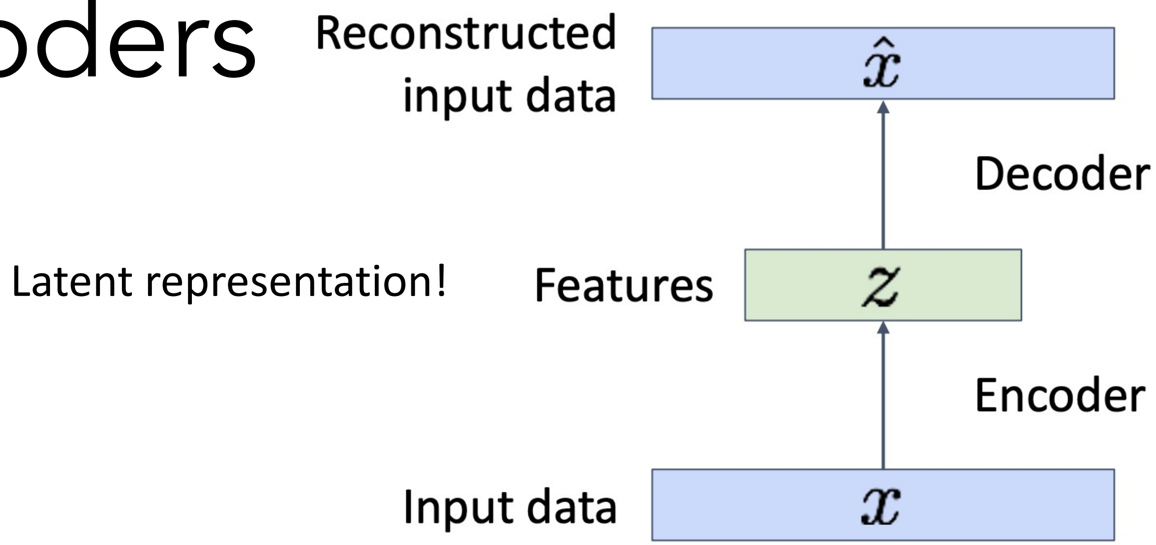
Assume simple prior p(z), e.g. Gaussian with mean

Assume **z** is latent representation that we can sample from to generate image **x**.

1. Learn latent representation
2. Sample to generate images

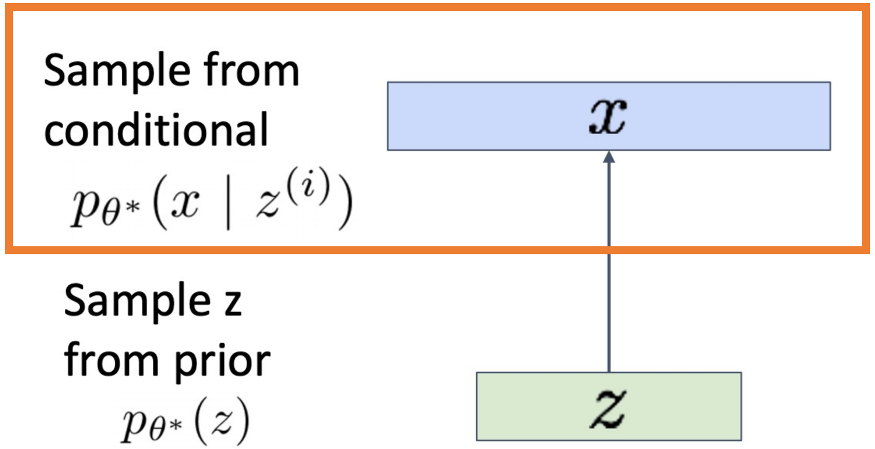
Adapted from slides by Justin Johnson

Variational Autoencoders



- Autoencoders
  - Not probabilistic
  - No sampling

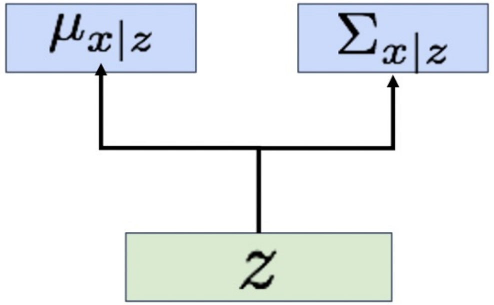
- Variational
  - Probabilistic



Sample x from Gaussian with mean $\mu_{x|z}$ and (diagonal) covariance $\Sigma_{x|z}$

Decoder Network

$$p_\theta(x \mid z) = N(\mu_{x|z}, \Sigma_{x|z})$$



How to sample?

Assume simple prior p(z), e.g. Gaussian with mean

Variational Autoencoders

- Let's maximize the likelihood of data! Need to compute $p_\theta(x)$

Marginalize?

$$p_\theta(x) = \int p_\theta(x, z)dz = \boxed{\int} p_\theta(x|z)p_\theta(z)\boxed{dz}$$

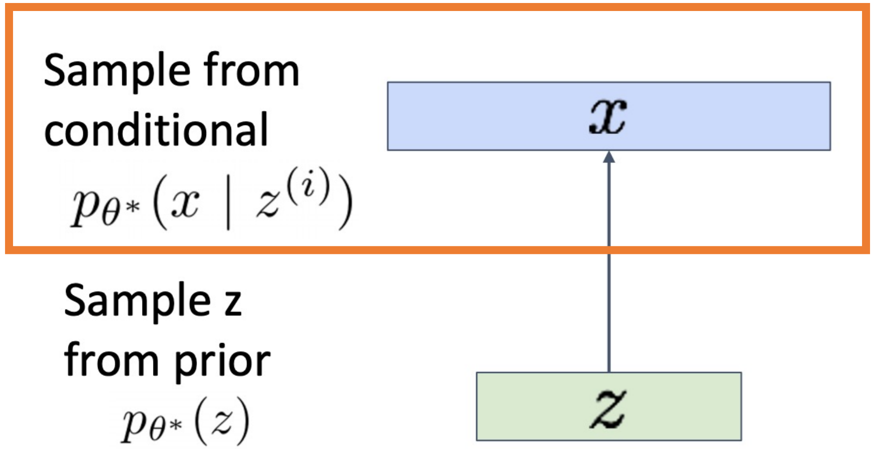
Problem: Impossible to integrate over all z!

Bayes Rule?

$$p_\theta(x) = \frac{p_\theta(x \mid z)p_\theta(z)}{\boxed{p_\theta(z \mid x)}}$$

Problem: No way to compute this!

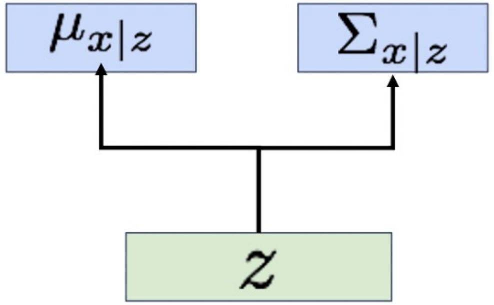
- Variational
  - Probabilistic



Sample x from Gaussian with mean $\mu_{x|z}$ and (diagonal) covariance $\Sigma_{x|z}$

Decoder Network

$$p_\theta(x \mid z) = N(\mu_{x|z}, \Sigma_{x|z})$$



How to sample?

Assume simple prior p(z), e.g. Gaussian with mean

Adapted from slides by Justin Johnson

Variational Autoencoders

- Let's maximize the likelihood of data! Need to compute $p_\theta(x)$

Let's train encoder and decoder jointly!

Solution: Train another network **(encoder)** that learns $q_\phi(z \mid x) \approx p_\theta(z \mid x)$

Bayes Rule?

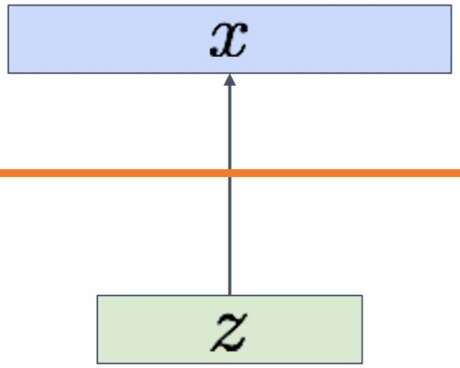
$$p_\theta(x) = \frac{p_\theta(x \mid z) p_\theta(z)}{\boxed{p_\theta(z \mid x)}}$$

Problem: No way to compute this!

- Variational
  - Probabilistic

Sample from conditional $p_{\theta^*}(x \mid z^{(i)})$

Sample z from prior $p_{\theta^*}(z)$



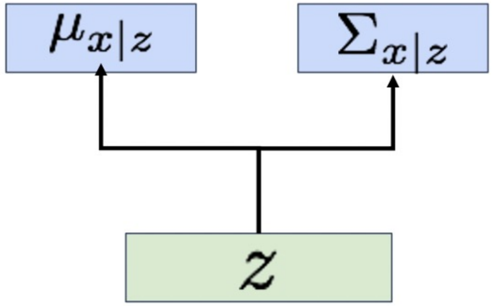
How to sample?

Assume simple prior p(z), e.g. Gaussian with mean

Sample x from Gaussian with mean $\mu_{x|z}$ and (diagonal) covariance $\Sigma_{x|z}$

Decoder Network

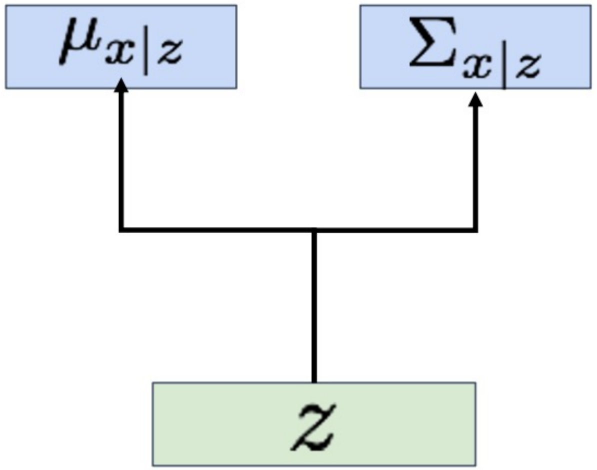
$$p_\theta(x \mid z) = N(\mu_{x|z}, \Sigma_{x|z})$$



Variational Autoencoders (VAE)

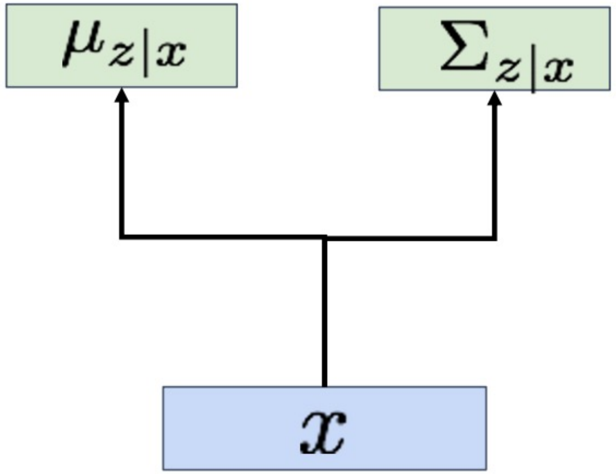
Decoder network inputs latent code z, gives distribution over data x

$$p_\theta(x \mid z) = N(\mu_{x|z}, \Sigma_{x|z})$$



Encoder network inputs data x, gives distribution over latent codes z

$$q_\phi(z \mid x) = N(\mu_{z|x}, \Sigma_{z|x})$$



If we can ensure that $q_\phi(z \mid x) \approx p_\theta(z \mid x)$,

then we can approximate

$$p_\theta(x) \approx \frac{p_\theta(x \mid z)p(z)}{q_\phi(z \mid x)}$$

Idea: Jointly train both encoder and decoder

Adapted from slides by Justin Johnson

Variational AutoEncoders (VAE)

Bunch of math to get a lower bound that we can optimize for!

$$\log p_\theta(x) = \log \frac{p_\theta(x \mid z)p(z)}{p_\theta(z \mid x)} = \log \frac{p_\theta(x|z)p(z)q_\phi(z|x)}{p_\theta(z|x)q_\phi(z|x)}$$

Variational AutoEncoders (VAE)

Bunch of math to get a lower bound that we can optimize for!

$$\log p_\theta(x) = \log \frac{p_\theta(x \mid z)p(z)}{p_\theta(z \mid x)} = \log \frac{\boxed{p_\theta(x|z)}\boxed{p(z)}\boxed{q_\phi(z|x)}}{\boxed{p_\theta(z|x)}\boxed{q_\phi(z|x)}}$$

$$= \log \boxed{p_\theta(x|z)} - \log \frac{\boxed{q_\phi(z|x)}}{\boxed{p(z)}} + \log \frac{\boxed{q_\phi(z|x)}}{\boxed{p_\theta(z|x)}}$$

Apply expectation (safely because x doesn't depend on z)

$$\log p_\theta(x) = E_{z \sim q_\phi(z|x)}[\log p_\theta(x)]$$

$$= E_z[\log p_\theta(x|z)] - E_z\left[\log \frac{q_\phi(z|x)}{p(z)}\right] + E_z\left[\log \frac{q_\phi(z|x)}{p_\theta(z|x)}\right]$$

Adapted from slides by Justin Johnson

Variational AutoEncoders (VAE)

Bunch of math to get a lower bound that we can optimize for!

$$\log p_\theta(x) = \log \frac{p_\theta(x \mid z) p(z)}{p_\theta(z \mid x)} = \log \frac{p_\theta(x|z) p(z) q_\phi(z|x)}{p_\theta(z|x) q_\phi(z|x)}$$

$$= E_z[\log p_\theta(x|z)] - E_z\left[\log \frac{q_\phi(z|x)}{p(z)}\right] + E_z\left[\log \frac{q_\phi(z|x)}{p_\theta(z|x)}\right]$$

Data reconstruction

KL divergence between prior, and samples from the encoder network

KL divergence between encoder and posterior of decoder

$$= E_{z \sim q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL}\left(q_\phi(z|x), p(z)\right) + D_{KL}(q_\phi(z|x), \boxed{p_\theta(z|x)})$$

KL is >= 0, so dropping this term gives a **lower bound** on the data likelihood:

Adapted from slides by Justin Johnson

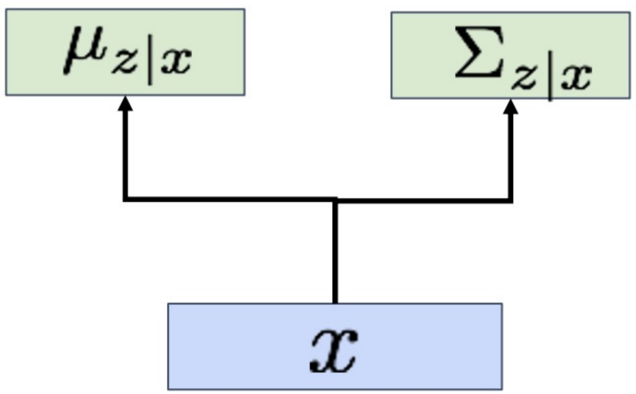
Variational Autoencoders (VAE)

Jointly train **encoder** q and **decoder** p to maximize the **variational lower bound** on the data likelihood

$$\log p_\theta(x) \geq E_{z \sim q_\phi(z|x)}[\log p_\theta(x|z)] - D_{KL}\left(q_\phi(z|x), p(z)\right)$$

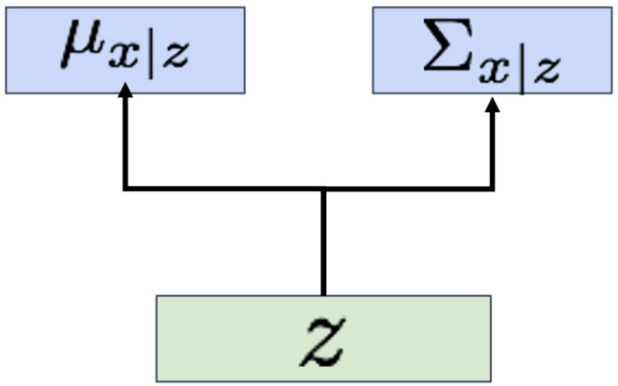
Encoder Network

$$q_\phi(z \mid x) = N(\mu_{z|x}, \Sigma_{z|x})$$



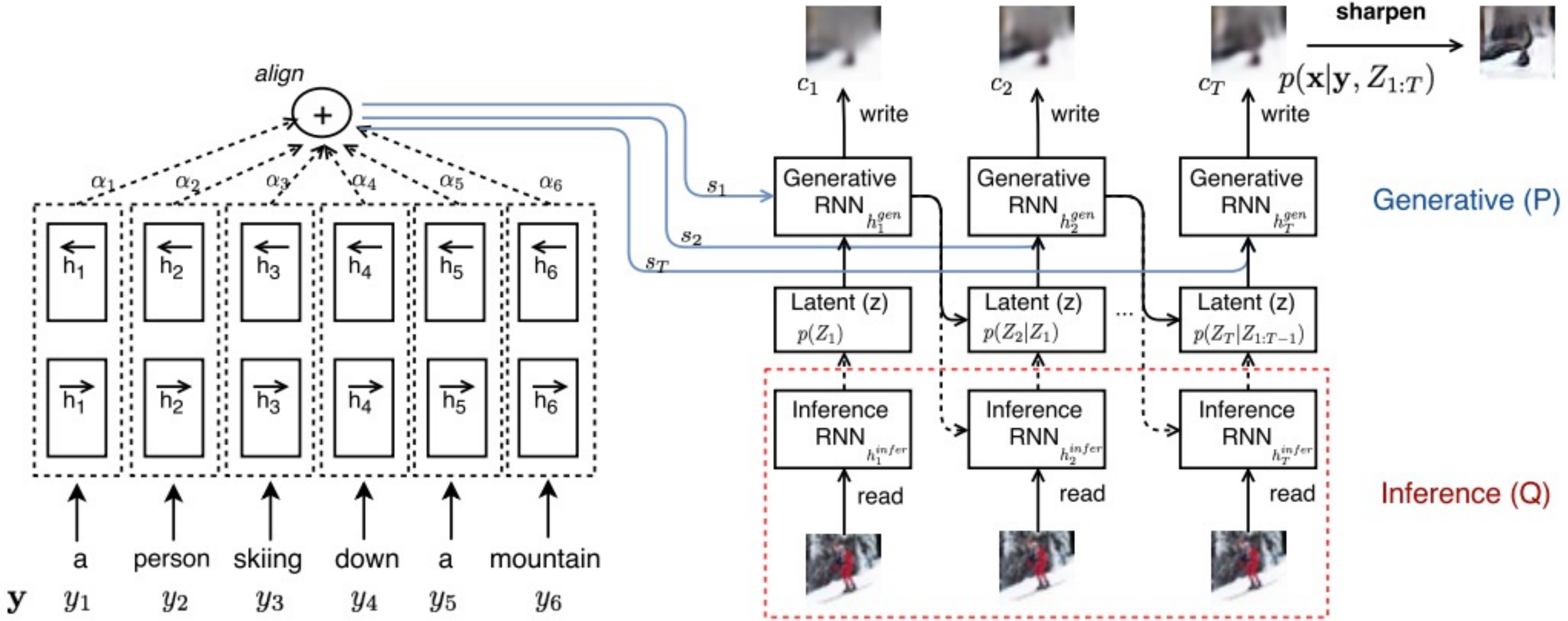
Decoder Network

$$p_\theta(x \mid z) = N(\mu_{x|z}, \Sigma_{x|z})$$



Adapted from slides by Justin Johnson

Text-based image generation with VAE



Generating Images from Captions with Attention
https://arxiv.org/pdf/1511.02793.pdf, Mansimov et al, ICLR 2016

Text-based image generation with VAE

Generating Images from Captions with Attention
https://arxiv.org/pdf/1511.02793.pdf, Mansimov et al, ICLR 2016

Compare AR and VAE models

Autoregressive models
- Directly maximize p(data)
- High-quality generated images
- Slow to generate images
- No explicit latent codes

Variational models
- Maximize lower-bound on p(data)
- Generated images often blurry
- Very fast to generate images
- Learn rich latent codes

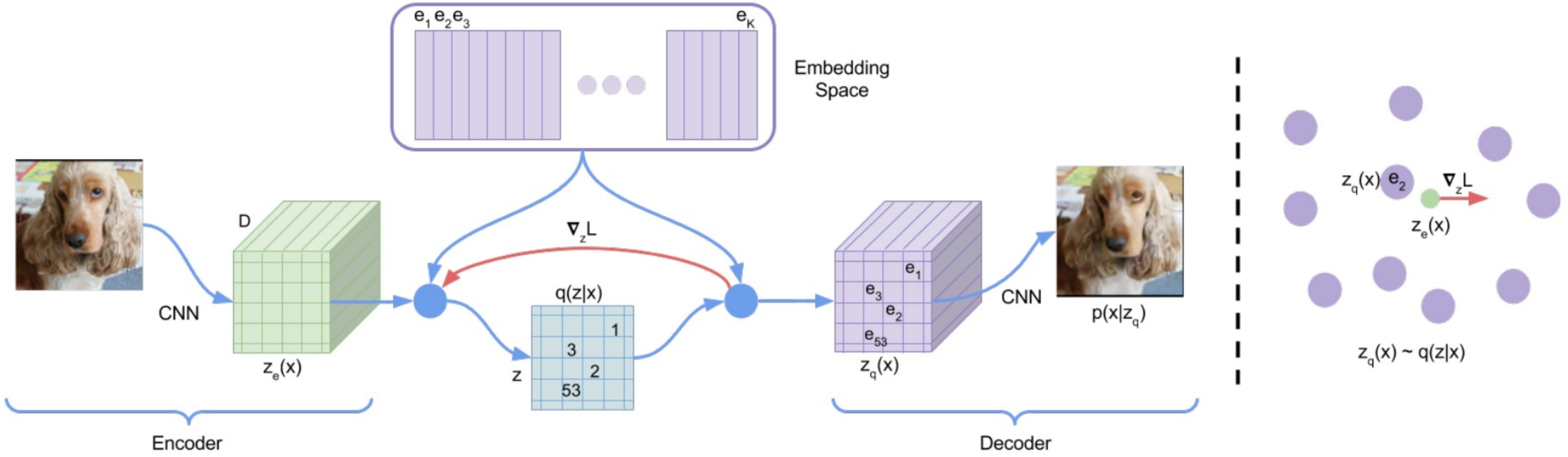
Can we combine them and get the best of both worlds?

Slide credit: Justin Johnson (https://web.eecs.umich.edu/~justincj/teaching/eecs498/FA2020/schedule.html, L19,20)

Combine VAE + Autoregressive

Vector-Quantized Variational Autoencoder (VQ-VAE)

- Autoregressively model images
- But instead of directly on pixels, on image patches compressed into image ``tokens'' using VAE



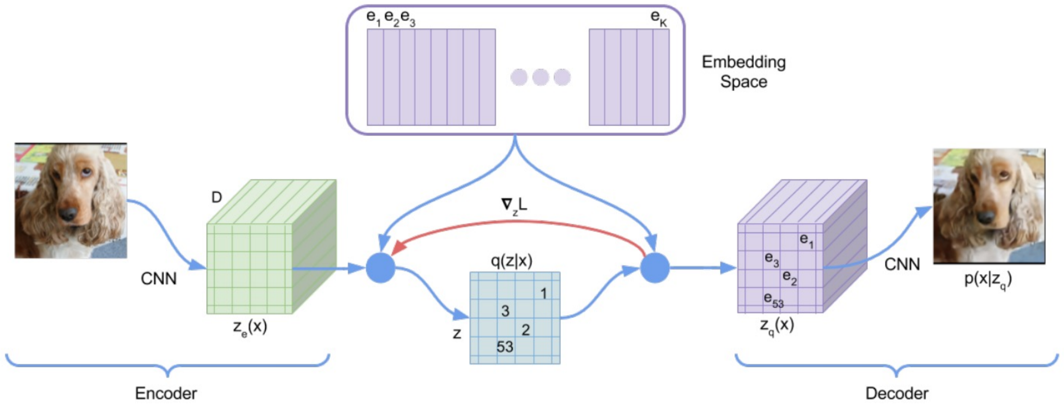
- Two-stage training process

Neural Discrete Representation Learning
https://arxiv.org/pdf/1711.00937.pdf, Oord et al, NIPS 2017

Combine VAE + Autoregressive

Vector-Quantized Variational Autoencoder (VQ-VAE)

- Two-stage training process
- Use VAE to create a code book to encode image patch into latent quantized discrete vector



128x128 class-conditional results trained on ImageNet

kit fox | gray whale | brown bear | admiral butterfly

- Use autoregressive model (PixelCNN) to model latent prior p(z)

Neural Discrete Representation Learning
https://arxiv.org/pdf/1711.00937.pdf, Oord et al, NIPS 2017

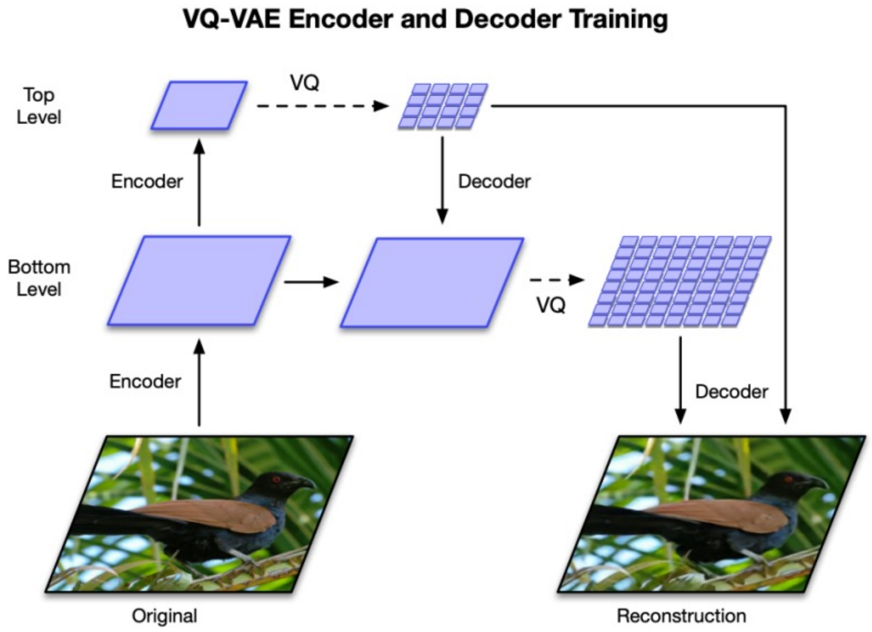
Combine VAE + Autoregressive

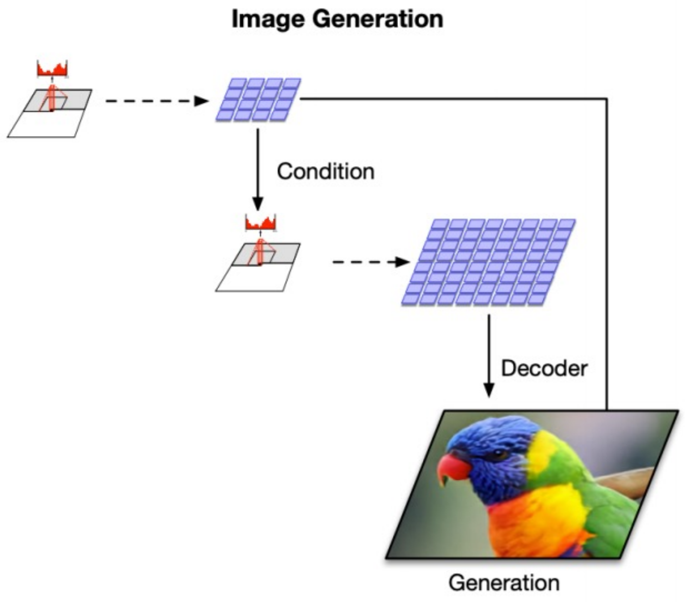
Vector-Quantized Variational Autoencoder (VQ-VAE2)

- Hierarchical VQ-VAE

Train a VAE-like model to generate multiscale grids of latent codes

Use a multiscale PixelCNN to sample in latent code space





Generating Diverse High-Fidelity Images with VQ-VAE-2
https://arxiv.org/pdf/1906.00446.pdf, Razavi et al, NeurIPS 2019

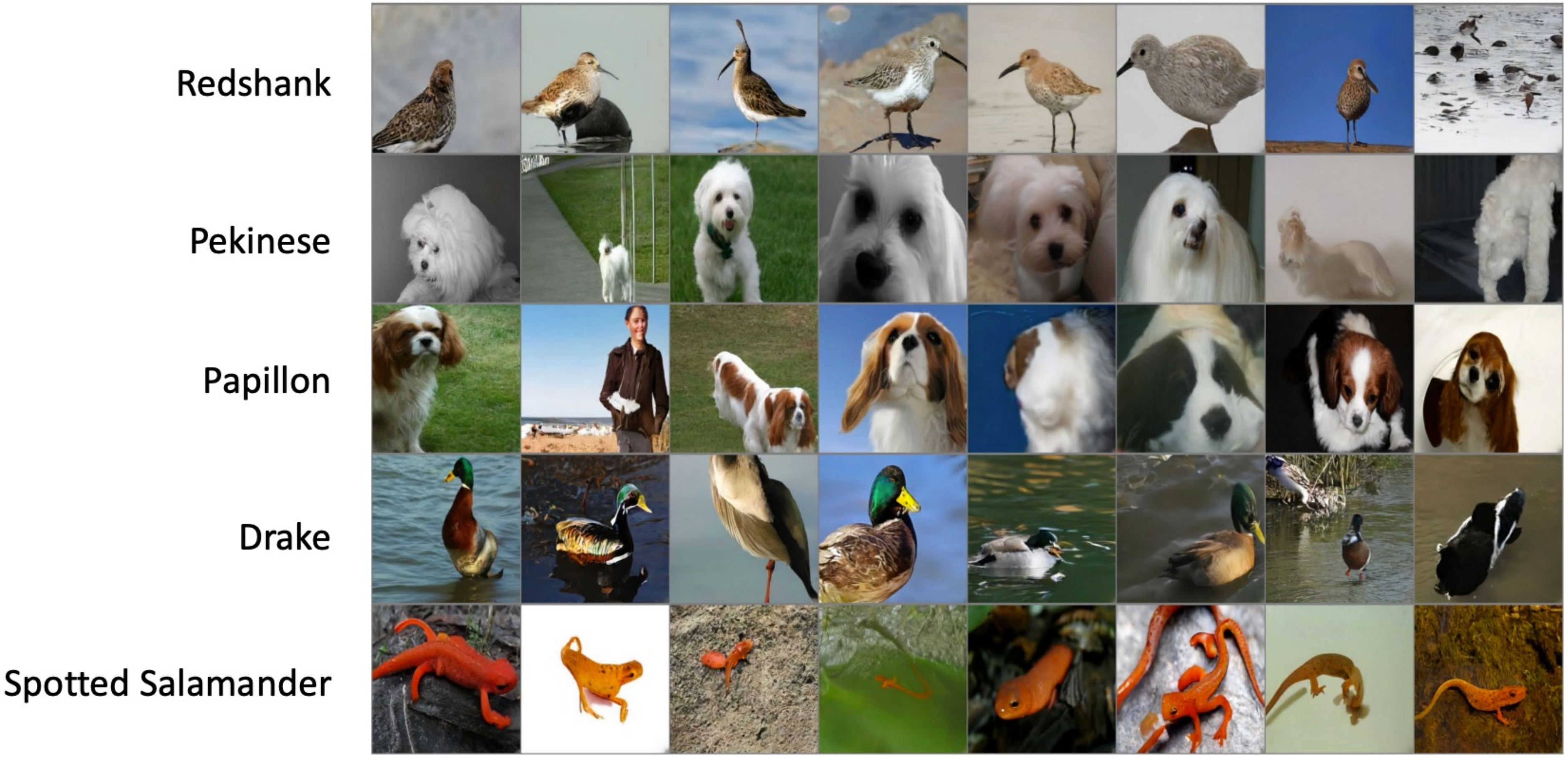
VQ-VAE2 Results

256 x 256 class-conditional samples, trained on ImageNet

Generating Diverse High-Fidelity Images with VQ-VAE-2
https://arxiv.org/pdf/1906.00446.pdf, Razavi et al, NeurIPS 2019

VQ-VAE2 Results

256 x 256 class-conditional samples, trained on ImageNet



Generating Diverse High-Fidelity Images with VQ-VAE-2
https://arxiv.org/pdf/1906.00446.pdf, Razavi et al, NeurIPS 2019

VQ-VAE2 Results

1024 x 1024 generated faces, trained on FFHQ

Generating Diverse High-Fidelity Images with VQ-VAE-2
https://arxiv.org/pdf/1906.00446.pdf, Razavi et al, NeurIPS 2019

DALL-E

- Like VQ-VAE2 but
  - Conditioned on text
  - Large network trained with tons of data
    - Used 3.3M text/image pairs (Conceptual Captions) for 1.2B parameter model
    - Used 120 text/image pairs (collected from Internet) for 12B parameter model
  - Uses autoregressive transformer vs PixelCNN
  - Uses CLIP to rerank generated images (vs classifier network trained on ImageNet)

"Dall-e"
[Ramesh et al, https://openai.com/blog/dall-e/]

DALL-E: Results

this gray bird has a pointed beak black wings with small white bars long thigh and tarsus and a long tail relative to its size

this rotund bird has a black tipped beak a black tail with a yellow tip and a black cheek patch

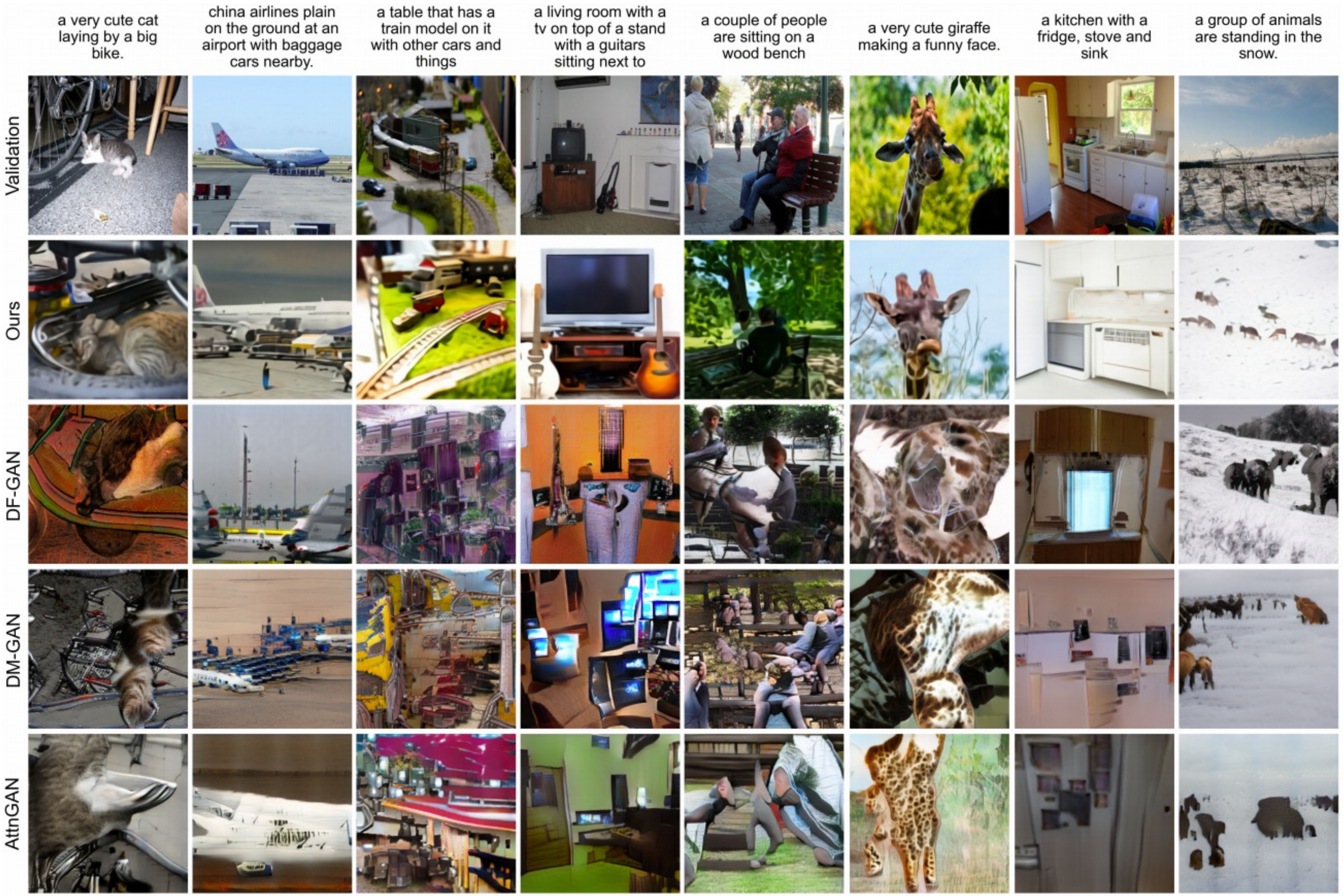
this is a small white bird with a yellow crown and a black eye ring and cheek patch and throat

the small bird has a dark brown head and light brown body

small bird with a pale yellow underside light brown crown and back gray tail and wing tips tip of tail feather bright yellow black eyes and black stripe over eyes

a small bird with a grey head and grey nape with grey black and white covering the rest of the body

DALL-E: Results



Diffusion models

- Define Markov chain of transitions from input to series of latent variables.

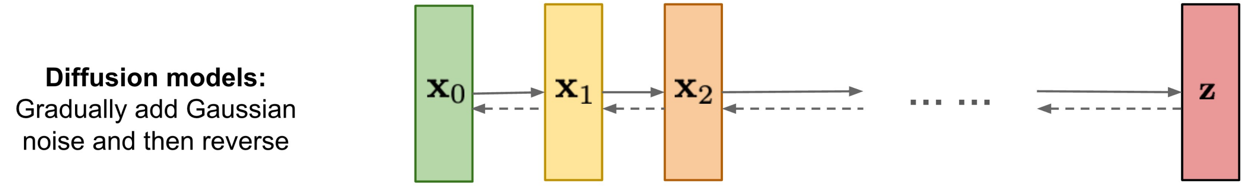
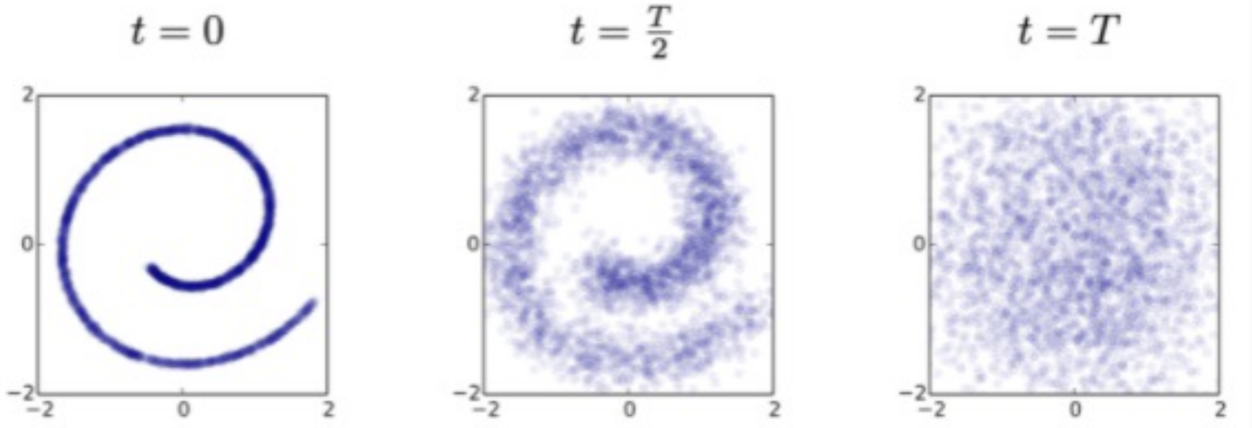


Figure credit: https://lilianweng.github.io/lil-log/2021/07/11/diffusion-models.html

- Forward process (diffusion process)

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

- Reverse diffusion: recreate sample (image) from latent (gaussian noise)



(Image source: [Sohl-Dickstein et al., 2015](#))

GLIDE: Diffusion Models

- Large diffusion model

"a hedgehog using a calculator"

"a corgi wearing a red bowtie and a purple party hat"

"robots meditating in a vipassana retreat"

"a fall landscape with a small cottage next to a lake"

"a surrealist dream-like oil painting by salvador dalí of a cat playing checkers"

"a professional photo of a sunset behind the grand canyon"

"a high-quality oil painting of a psychedelic hamster dragon"

"an illustration of albert einstein wearing a superhero costume"

GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models
https://arxiv.org/pdf/2112.10741.pdf, Nichol et al, arXiv 2021

Evaluating generated content

Evaluation

- Evaluation of these models are tricky!

- What makes for a good generation?

- General
  - Is the generated content high quality?
  - Does it match the distribution?
  - Is it diverse?

- For language conditioned generation:
  - Does the generated content match the language?
  - Are salient aspects of the language captured in the objects, appearance, and relationships?

GAN evaluation

- Inception Score: $I = \exp(\mathbb{E}_{\boldsymbol{x}} D_{KL}(p(y|\boldsymbol{x}) \,||\, p(y)))$
    - Use inception model to predict class y
    - Want good models to generate diverse but meaningful images
    - Large distance between marginal prior (of labels) and conditional prior
- FID (Frechet Inception distance): measures distance between generated and real distribution
- Human rank images generated by models

Metrics

- R-Precision (retrieval)
  - Randomly sample 99 other captions, where is the input caption ranked (using cosine similarity) compared to the rest (is it in the top r)?
- Visual similarity (VS)
  - how well does the encoded text and image match)
  - High variance, dependency on the specific encoders used

$$\mathrm{VS} = \frac{f_t(t) \cdot f_x(x)}{||f_t(t)||_2 \cdot ||f_x(x)||_2}$$

- Semantic Object Accuracy (SOA)
  - Use pretrained object detector to match words in text
- Captioning – generate caption and evaluate with original caption using standard captioning metrics

Adversarial Text-to-Image Synthesis: A Review
https://arxiv.org/pdf/2101.09983.pdf, Frolov et al, 2021

Metrics

Metric	Image Quality	Image Diversity	Object Fidelity	Text Relevance	Mentioned Objects	Numerical Alignment	Positional Alignment	Paraphrase Robustness	Explainable	Automatic
IS [130]	✓									✓
FID [131]	✓	✓								✓
SceneFID [103]			✓							✓
R-prec. [35]				✓						✓
VS [42]				✓						✓
SOA [108]				✓	✓					✓
Captioning				(✓)						✓
User Studies	✓	✓	✓	✓	✓	✓	✓	✓	✓	

Adversarial Text-to-Image Synthesis: A Review
https://arxiv.org/pdf/2101.09983.pdf, Frolov et al, 2021

Next time

- Monday: Paper presentations and discussions
  - (Tristan) Cross-Modal Contrastive Learning for Text-to-Image Generation
  - (Han-Hung) GLIDE: Toward Photorealist Image Generation
- Wednesday: Compositionality and structured representations