# Deep Generative Models

## Lecture 12

Roman Isachenko

Moscow Institute of Physics and Technology

2022 – 2023

# Recap of previous lecture

Let take some pretrained image classification model to get the conditional label distribution $p(y|\mathbf{x})$ (e.g. ImageNet classifier).

## Evaluation of likelihood-free models

- Sharpness $\Rightarrow$ low $H(y|\mathbf{x}) = -\sum_y \int_{\mathbf{x}} p(y, \mathbf{x}) \log p(y|\mathbf{x}) d\mathbf{x}$.
- Diversity $\Rightarrow$ high $H(y) = -\sum_y p(y) \log p(y)$.

## Inception Score

$$IS = \exp(H(y) - H(y|\mathbf{x})) = \exp\left(\mathbb{E}_{\mathbf{x}} KL(p(y|\mathbf{x}) || p(y))\right)$$

## Frechet Inception Distance

$$D^2(\pi, p) = \|\mathbf{m}_\pi - \mathbf{m}_p\|_2^2 + \mathrm{Tr}\left(\boldsymbol{\Sigma}_\pi + \boldsymbol{\Sigma}_p - 2\sqrt{\boldsymbol{\Sigma}_\pi \boldsymbol{\Sigma}_p}\right).$$

FID is related to moment matching.

*Salimans T. et al. Improved Techniques for Training GANs, 2016*
*Heusel M. et al. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium, 2017*

# Recap of previous lecture

- $\mathcal{S}_\pi = \{\mathbf{x}_i\}_{i=1}^n \sim \pi(\mathbf{x})$ – real samples;
- $\mathcal{S}_p = \{\mathbf{x}_i\}_{i=1}^n \sim p(\mathbf{x}|\boldsymbol{\theta})$ – generated samples.

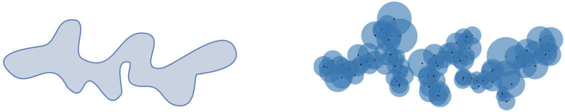
Embed samples using pretrained classifier network (as previously):

$$\mathcal{G}_\pi = \{\mathbf{g}_i\}_{i=1}^n, \quad \mathcal{G}_p = \{\mathbf{g}_i\}_{i=1}^n.$$

Define binary function:

$$f(\mathbf{g}, \mathcal{G}) = \begin{cases} 1, \text{if exists } \mathbf{g}' \in \mathcal{G} : \|\mathbf{g} - \mathbf{g}'\|_2 \leq \|\mathbf{g}' - \mathrm{NN}_k(\mathbf{g}', \mathcal{G})\|_2; \\ 0, \text{otherwise.} \end{cases}$$

$$\mathrm{Precision}(\mathcal{G}_\pi, \mathcal{G}_p) = \frac{1}{n} \sum_{\mathbf{g} \in \mathcal{G}_p} f(\mathbf{g}, \mathcal{G}_\pi); \quad \mathrm{Recall}(\mathcal{G}_\pi, \mathcal{G}_p) = \frac{1}{n} \sum_{\mathbf{g} \in \mathcal{G}_\pi} f(\mathbf{g}, \mathcal{G}_p).$$



(a) True manifold (b) Approx. manifold

*Kynkäänniemi T. et al. Improved precision and recall metric for assessing generative models, 2019*

# Recap of previous lecture

## Discrete VAE latents

- Define dictionary (word book) space $\{\mathbf{e}_k\}_{k=1}^K$, where $\mathbf{e}_k \in \mathbb{R}^C$, $K$ is the size of the dictionary.
- Our variational posterior $q(c|\mathbf{x}, \phi) = \text{Categorical}(\boldsymbol{\pi}(\mathbf{x}, \phi))$ (encoder) outputs discrete probabilities vector.
- We sample $c^*$ from $q(c|\mathbf{x}, \phi)$ (reparametrization trick analogue).
- Our generative distribution $p(\mathbf{x}|\mathbf{e}_{c^*}, \boldsymbol{\theta})$ (decoder).

## ELBO

$$\mathcal{L}(\phi, \boldsymbol{\theta}) = \mathbb{E}_{q(c|\mathbf{x},\phi)} \log p(\mathbf{x}|c, \boldsymbol{\theta}) - KL(q(c|\mathbf{x}, \phi)||p(c)) \to \max_{\phi, \boldsymbol{\theta}}.$$

## KL term

$$KL(q(c|\mathbf{x}, \phi)||p(c)) = -H(q(c|\mathbf{x}, \phi)) + \log K.$$

Is it possible to make reparametrization trick? (we sample from discrete distribution now!).

# Outline

1. Discrete VAE latent representations
   - Vector quantization
   - Gumbel-softmax

2. Neural ODE

# Outline

1. Discrete VAE latent representations
   - Vector quantization
   - Gumbel-softmax

2. Neural ODE

# Outline

# Vector quantization

## Quantized representation

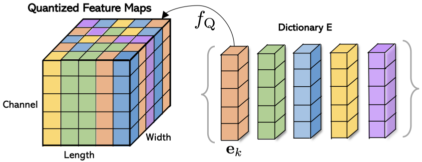
$\mathbf{z}_q \in \mathbb{R}^C$ for $\mathbf{z} \in \mathbb{R}^C$ is defined by a nearest neighbor look-up using the shared dictionary space

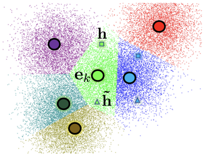
$$\mathbf{z}_q = \mathbf{e}_{k^*}, \quad \text{where } k^* = \arg\min_k \|\mathbf{z} - \mathbf{e}_k\|.$$

- Let our encoder outputs continuous representation $\mathbf{z}$.
- Quantization will give us the discrete distribution $q(c|\mathbf{x}, \phi)$.

## Quantization procedure

If we have tensor with the spatial dimensions we apply the quantization for each of $W \times H$ locations.




*Zhao Y. et al. Feature Quantization Improves GAN Training, 2020*

# Vector Quantized VAE (VQ-VAE)

Let VAE latent variable $\mathbf{c} \in \{1, \dots, K\}^{W \times H}$ is the discrete with spatial-independent variational posterior and prior distributions

$$q(\mathbf{c}|\mathbf{x}, \phi) = \prod_{i=1}^{W} \prod_{j=1}^{H} q(c_{ij}|\mathbf{x}, \phi); \quad p(\mathbf{c}) = \prod_{i=1}^{W} \prod_{j=1}^{H} \text{Uniform}\{1, \dots, K\}.$$

Let $\mathbf{z}_e = \text{NN}_e(\mathbf{x}, \phi) \in \mathbb{R}^{W \times H \times C}$ is the encoder output.

## Deterministic variational posterior

$$q(c_{ij} = k^*|\mathbf{x}, \phi) = \begin{cases} 1, & \text{for } k^* = \arg\min_k \|[\mathbf{z}_e]_{ij} - \mathbf{e}_k\|; \\ 0, & \text{otherwise.} \end{cases}$$

$KL(q(c|\mathbf{x}, \phi)||p(c))$ term in ELBO is constant, entropy of the posterior is zero.

$$KL(q(c|\mathbf{x}, \phi)||p(c)) = -H(q(c|\mathbf{x}, \phi)) + \log K = \log K.$$

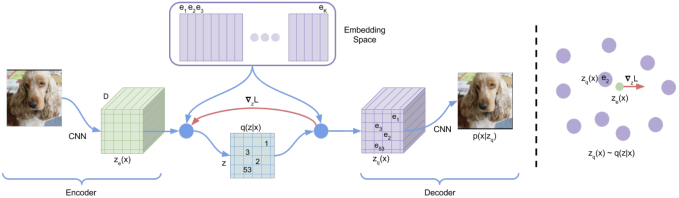
*Oord A., Vinyals O., Kavukcuoglu K. Neural Discrete Representation Learning, 2017*

# Vector Quantized VAE (VQ-VAE)

## ELBO

$$\mathcal{L}(\phi, \boldsymbol{\theta}) = \mathbb{E}_{q(c|\mathbf{x},\phi)} \log p(\mathbf{x}|\mathbf{e}_c, \boldsymbol{\theta}) - \log K = \log p(\mathbf{x}|\mathbf{z}_q, \boldsymbol{\theta}) - \log K,$$

where $\mathbf{z}_q = \mathbf{e}_{k^*}, \; k^* = \arg\min_k \|\mathbf{z}_e - \mathbf{e}_k\|$.




**Problem:** arg min is not differentiable.

## Straight-through gradient estimation

$$\frac{\partial \log p(\mathbf{x}|\mathbf{z}_q, \boldsymbol{\theta})}{\partial \phi} = \frac{\partial \log p(\mathbf{x}|\mathbf{z}_q, \boldsymbol{\theta})}{\partial \mathbf{z}_q} \cdot \frac{\partial \mathbf{z}_q}{\partial \phi} \approx \frac{\partial \log p(\mathbf{x}|\mathbf{z}_q, \boldsymbol{\theta})}{\partial \mathbf{z}_q} \cdot \frac{\partial \mathbf{z}_e}{\partial \phi}$$

*Oord A., Vinyals O., Kavukcuoglu K. Neural Discrete Representation Learning, 2017*

# Vector Quantized VAE-2 (VQ-VAE-2)

## Samples 1024x1024

## Samples diversity

VQ-VAE (Proposed) | BigGAN deep

*Razavi A., Oord A., Vinyals O. Generating Diverse High-Fidelity Images with VQ-VAE-2, 2019*

# Outline

1. Discrete VAE latent representations
   Vector quantization
   Gumbel-softmax

2. Neural ODE

# Gumbel-softmax trick

- VQ-VAE has deterministic variational posterior (it allows to get rid of discrete sampling and reparametrization trick).
- There is no uncertainty in the encoder output.

## Gumbel-max trick

Let $g_k \sim \text{Gumbel}(0, 1)$ for $k = 1, \dots, K$, i.e. $g = -\log(-\log u)$, $u \sim \text{Uniform}[0, 1]$. Then a discrete random variable

$$c = \arg\max_k [\log \pi_k + g_k],$$

has a categorical distribution $c \sim \text{Categorical}(\boldsymbol{\pi})$.

- Let our encoder $q(c|\mathbf{x}, \phi) = \text{Categorical}(\boldsymbol{\pi}(\mathbf{x}, \phi))$ outputs logits of $\boldsymbol{\pi}(\mathbf{x}, \phi)$.
- We could sample from the discrete distribution using Gumbel-max reparametrization.

---

*Maddison C. J., Mnih A., Teh Y. W. The Concrete distribution: A continuous relaxation of discrete random variables, 2016*
*Jang E., Gu S., Poole B. Categorical reparameterization with Gumbel-Softmax, 2016*

# Gumbel-softmax trick

## Reparametrization trick (LOTUS)

$$\nabla_\phi \mathbb{E}_{q(c|\mathbf{x},\phi)} \log p(\mathbf{x}|\mathbf{e}_c, \boldsymbol{\theta}) = \mathbb{E}_{\text{Gumbel}(0,1)} \nabla_\phi \log p(\mathbf{x}|\mathbf{e}_{k^*}, \boldsymbol{\theta}),$$

where $k^* = \arg\max_k [\log q(k|\mathbf{x}, \phi) + g_k]$.

**Problem:** We still have non-differentiable $\arg\max$ operation.

## Gumbel-softmax relaxation

Concrete distribution = **con**tinuous + dis**crete**

$$\hat{c}_k = \frac{\exp\left(\frac{\log q(k|\mathbf{x},\phi) + g_k}{\tau}\right)}{\sum_{j=1}^K \exp\left(\frac{\log q(j|\mathbf{x},\phi) + g_j}{\tau}\right)}, \quad k = 1, \dots, K.$$

Here $\tau$ is a temperature parameter. Now we have differentiable operation, but the gradient estimate is biased now.

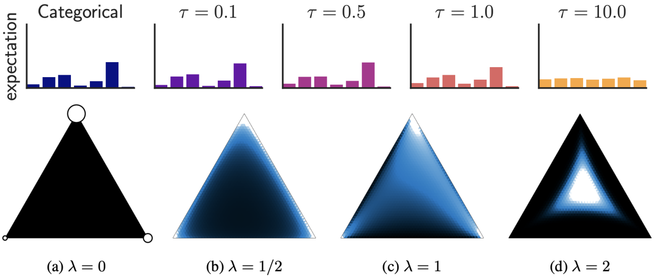
---

*Maddison C. J., Mnih A., Teh Y. W. The Concrete distribution: A continuous relaxation of discrete random variables, 2016*

*Jang E., Gu S., Poole B. Categorical reparameterization with Gumbel-Softmax, 2016*

# Gumbel-softmax trick

## Concrete distribution




## Reparametrization trick

$$\nabla_\phi \mathbb{E}_{q(c|\mathbf{x},\phi)} \log p(\mathbf{x}|\mathbf{e}_c, \boldsymbol{\theta}) = \mathbb{E}_{\text{Gumbel}(0,1)} \nabla_\phi \log p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}),$$

where $\mathbf{z} = \sum_{k=1}^{K} \hat{c}_k \mathbf{e}_k$ (all operations are differentiable now).

---

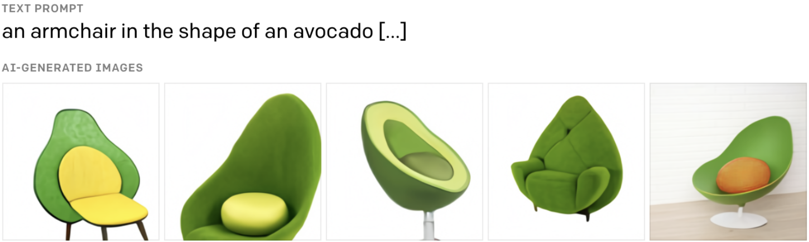
*Maddison C. J., Mnih A., Teh Y. W. The Concrete distribution: A continuous relaxation of discrete random variables, 2016*
*Jang E., Gu S., Poole B. Categorical reparameterization with Gumbel-Softmax, 2016*

# DALL-E/dVAE

## Deterministic VQ-VAE posterior

$$q(\hat{z}_{ij} = k^*|\mathbf{x}) = \begin{cases} 1, & \text{for } k^* = \arg\min_k \|[\mathbf{z}_e]_{ij} - \mathbf{e}_k\| \\ 0, & \text{otherwise.} \end{cases}$$

- Gumbel-Softmax trick allows to make true categorical distribution and sample from it.
- Since latent space is discrete we could train autoregressive transformers in it.
- It is a natural way to incorporate text and image token spaces.




*Ramesh A. et al. Zero-shot text-to-image generation, 2021*

# Outline

# Neural ODE

Consider Ordinary Differential Equation (ODE)

$$\frac{d\mathbf{z}(t)}{dt} = f(\mathbf{z}(t), t, \boldsymbol{\theta}); \quad \text{with initial condition } \mathbf{z}(t_0) = \mathbf{z}_0.$$

$$\mathbf{z}(t_1) = \int_{t_0}^{t_1} f(\mathbf{z}(t), t, \boldsymbol{\theta})dt + \mathbf{z}_0 = \text{ODESolve}(\mathbf{z}(t_0), f, t_0, t_1, \boldsymbol{\theta}).$$

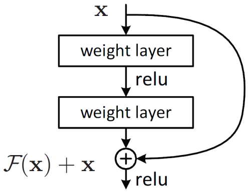
## Euler update step

$$\frac{\mathbf{z}(t+\Delta t) - \mathbf{z}(t)}{\Delta t} = f(\mathbf{z}(t), t, \boldsymbol{\theta}) \;\Rightarrow\; \mathbf{z}(t+\Delta t) = \mathbf{z}(t) + \Delta t \cdot f(\mathbf{z}(t), t, \boldsymbol{\theta})$$

## Residual block

$$\mathbf{z}_{t+1} = \mathbf{z}_t + f(\mathbf{z}_t, \boldsymbol{\theta})$$

- It is equavalent to Euler update step for solving ODE with $\Delta t = 1$!
- Euler update step is unstable and trivial. There are more sophisticated methods.




*Chen R. T. Q. et al. Neural Ordinary Differential Equations, 2018*

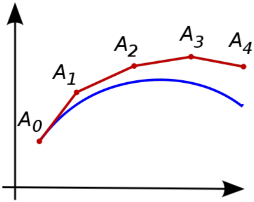
# Neural ODE

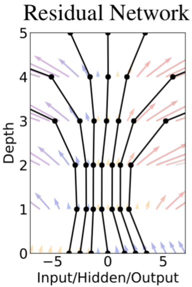
## Residual block

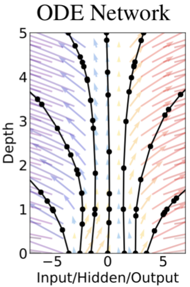
$$\mathbf{z}_{t+1} = \mathbf{z}_t + f(\mathbf{z}_t, \boldsymbol{\theta}).$$

In the limit of adding more layers and taking smaller steps, we parameterize the continuous dynamics of hidden units using an ODE specified by a neural network:

$$\frac{d\mathbf{z}(t)}{dt} = f(\mathbf{z}(t), t, \boldsymbol{\theta}); \quad \mathbf{z}(t_0) = \mathbf{x}; \quad \mathbf{z}(t_1) = \mathbf{y}.$$




*Chen R. T. Q. et al. Neural Ordinary Differential Equations, 2018*

# Neural ODE

## Forward pass (loss function)

$$L(\mathbf{y}) = L(\mathbf{z}(t_1)) = L\left(\mathbf{z}(t_0) + \int_{t_0}^{t_1} f(\mathbf{z}(t), t, \boldsymbol{\theta})dt\right)$$
$$= L\big(\text{ODESolve}(\mathbf{z}(t_0), f, t_0, t_1, \boldsymbol{\theta})\big)$$

**Note:** ODESolve could be any method (Euler step, Runge-Kutta methods).

## Backward pass (gradients computation)

For fitting parameters we need gradients:

$$\mathbf{a}_{\mathbf{z}}(t) = \frac{\partial L(\mathbf{y})}{\partial \mathbf{z}(t)}; \quad \mathbf{a}_{\boldsymbol{\theta}}(t) = \frac{\partial L(\mathbf{y})}{\partial \boldsymbol{\theta}(t)}.$$

In theory of optimal control these functions called **adjoint** functions. They show how the gradient of the loss depends on the hidden state $\mathbf{z}(t)$ and parameters $\boldsymbol{\theta}$.

*Chen R. T. Q. et al. Neural Ordinary Differential Equations, 2018*

# Summary

- Vector Quantization is the way to create VAE with discrete latent space and deterministic variational posterior.
- Straight-through gradient ignores quantize operation in backprop.
- Gumbel-softmax trick relaxes discrete problem to continuous one using Gumbel-max reparametrization trick.
- It becomes more and more popular to use discrete latent spaces in the fields of image/video/music generation.
- Residual networks could be interpreted as solution of ODE with Euler method.