Deep Generative Models

Lecture 6

Roman Isachenko

AI Masters

2024, Summer

Recap of previous lecture

$$\mathcal{L}(\phi, \theta) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x},\phi)} \left[\log p(\mathbf{x}|\mathbf{z}, \theta) - \log \frac{q(\mathbf{z}|\mathbf{x}, \phi)}{p(\mathbf{z})} \right] \to \max_{\phi,\theta}.$$

M-step: $\nabla_\theta \mathcal{L}(\phi, \theta)$, Monte Carlo estimation

$$\nabla_\theta \mathcal{L}(\phi, \theta) = \int q(\mathbf{z}|\mathbf{x}, \phi) \nabla_\theta \log p(\mathbf{x}|\mathbf{z}, \theta) d\mathbf{z} \approx$$

$$\approx \nabla_\theta \log p(\mathbf{x}|\mathbf{z}^*, \theta), \quad \mathbf{z}^* \sim q(\mathbf{z}|\mathbf{x}, \phi).$$

E-step: $\nabla_\phi \mathcal{L}(\phi, \theta)$, reparametrization trick

$$\nabla_\phi \mathcal{L}(\phi, \theta) = \int r(\epsilon) \nabla_\phi \log p(\mathbf{x}|\mathbf{g}_\phi(\mathbf{x}, \epsilon), \theta) d\epsilon - \nabla_\phi \mathrm{KL}$$

$$\approx \nabla_\phi \log p(\mathbf{x}|\mathbf{g}_\phi(\mathbf{x}, \epsilon^*), \theta) - \nabla_\phi \mathrm{KL}$$

Variational assumption

$$r(\epsilon) = \mathcal{N}(0, \mathbf{I}); \quad q(\mathbf{z}|\mathbf{x}, \phi) = \mathcal{N}(\boldsymbol{\mu}_\phi(\mathbf{x}), \boldsymbol{\sigma}^2_\phi(\mathbf{x})).$$

$$\mathbf{z} = \mathbf{g}_\phi(\mathbf{x}, \epsilon) = \boldsymbol{\sigma}_\phi(\mathbf{x}) \odot \epsilon + \boldsymbol{\mu}_\phi(\mathbf{x}).$$

Recap of previous lecture

Final EM-algorithm

- pick random sample $\mathbf{x}_i, i \sim U[1, n]$.
- compute the objective:

$$\epsilon^* \sim r(\epsilon); \quad \mathbf{z}^* = \mathbf{g}_\phi(\mathbf{x}, \epsilon^*);$$

$$\mathcal{L}(\phi, \theta) \approx \log p(\mathbf{x}|\mathbf{z}^*, \theta) - KL(q(\mathbf{z}^*|\mathbf{x}, \phi)||p(\mathbf{z}^*)).$$

- compute a stochastic gradients w.r.t. ϕ and θ

$$\nabla_\phi \mathcal{L}(\phi, \theta) \approx \nabla_\phi \log p(\mathbf{x}|\mathbf{g}_\phi(\mathbf{x}, \epsilon^*), \theta) - \nabla_\phi \mathrm{KL}(q(\mathbf{z}|\mathbf{x}, \phi)||p(\mathbf{z}));$$
$$\nabla_\theta \mathcal{L}(\phi, \theta) \approx \nabla_\theta \log p(\mathbf{x}|\mathbf{z}^*, \theta).$$

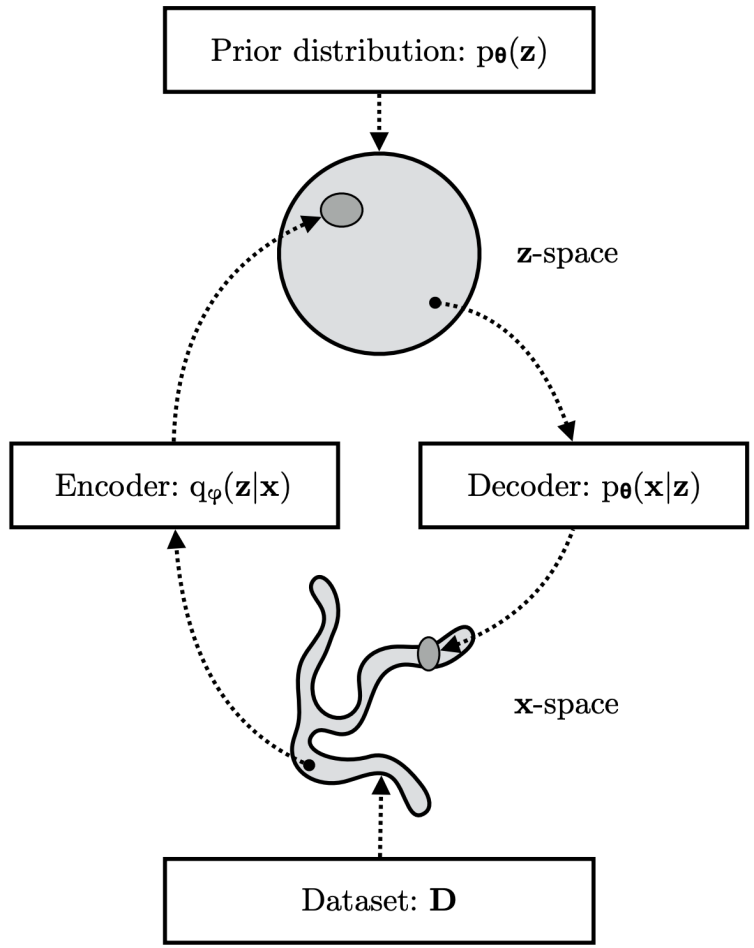
- update θ, ϕ according to the selected optimization method (SGD, Adam):

$$\phi := \phi + \eta \cdot \nabla_\phi \mathcal{L}(\phi, \theta),$$
$$\theta := \theta + \eta \cdot \nabla_\theta \mathcal{L}(\phi, \theta).$$

Recap of previous lecture

Variational autoencoder (VAE)

- VAE learns stochastic mapping between $\mathbf{x}$-space, from $\pi(\mathbf{x})$, and a latent $\mathbf{z}$-space, with simple distribution.
- The generative model learns distribution $p(\mathbf{x}, \mathbf{z}|\boldsymbol{\theta}) = p(\mathbf{z})p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})$, with a prior distribution $p(\mathbf{z})$, and a stochastic decoder $p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta})$.
- The stochastic encoder $q(\mathbf{z}|\mathbf{x}, \boldsymbol{\phi})$ (inference model), approximates the true but intractable posterior $p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})$.



Kingma D. P., Welling M. An introduction to variational autoencoders, 2019

Recap of previous lecture

	VAE	**NF**
Objective	ELBO $\mathcal{L}$	Forward KL/MLE
Encoder	stochastic $\mathbf{z} \sim q(\mathbf{z}\|\mathbf{x}, \boldsymbol{\phi})$	deterministic $\mathbf{z} = \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x})$ $q(\mathbf{z}\|\mathbf{x}, \boldsymbol{\theta}) = \delta(\mathbf{z} - \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x}))$
Decoder	stochastic $\mathbf{x} \sim p(\mathbf{x}\|\mathbf{z}, \boldsymbol{\theta})$	deterministic $\mathbf{x} = \mathbf{g}_{\boldsymbol{\theta}}(\mathbf{z})$ $p(\mathbf{x}\|\mathbf{z}, \boldsymbol{\theta}) = \delta(\mathbf{x} - \mathbf{g}_{\boldsymbol{\theta}}(\mathbf{z}))$
Parameters	$\boldsymbol{\phi}, \boldsymbol{\theta}$	$\boldsymbol{\theta} \equiv \boldsymbol{\phi}$

Theorem

MLE for normalizing flow is equivalent to maximization of ELBO for VAE model with deterministic encoder and decoder:

$$p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) = \delta(\mathbf{x} - \mathbf{f}_{\boldsymbol{\theta}}^{-1}(\mathbf{z})) = \delta(\mathbf{x} - \mathbf{g}_{\boldsymbol{\theta}}(\mathbf{z}));$$

$$q(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta}) = p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta}) = \delta(\mathbf{z} - \mathbf{f}_{\boldsymbol{\theta}}(\mathbf{x})).$$

Nielsen D., et al. SurVAE Flows: Surjections to Bridge the Gap between VAEs and Flows, 2020

Recap of previous lecture

Theorem

$$\frac{1}{n}\sum_{i=1}^{n} KL(q(\mathbf{z}|\mathbf{x}_i,\phi)||p(\mathbf{z})) = KL(q_{\text{agg}}(\mathbf{z}|\phi)||p(\mathbf{z})) + \mathbb{I}_q[\mathbf{x},\mathbf{z}].$$

ELBO surgery

$$\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}_i(\phi,\theta) = \underbrace{\frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{q(\mathbf{z}|\mathbf{x}_i,\phi)}\log p(\mathbf{x}_i|\mathbf{z},\theta)}_{\text{Reconstruction loss}} - \underbrace{\mathbb{I}_q[\mathbf{x},\mathbf{z}]}_{\text{MI}} - \underbrace{KL(q_{\text{agg}}(\mathbf{z}|\phi)||p(\mathbf{z}))}_{\text{Marginal KL}}$$

Optimal prior

$$KL(q_{\text{agg}}(\mathbf{z}|\phi)||p(\mathbf{z})) = 0 \quad \Leftrightarrow \quad p(\mathbf{z}) = q_{\text{agg}}(\mathbf{z}) = \frac{1}{n}\sum_{i=1}^{n} q(\mathbf{z}|\mathbf{x}_i,\phi).$$

The optimal prior distribution $p(\mathbf{z})$ is the aggregated variational posterior distribution $q_{\text{agg}}(\mathbf{z}|\phi)$.

Hoffman M. D., Johnson M. J. ELBO surgery: yet another way to carve up the variational evidence lower bound, 2016

Outline

1. Learnable VAE prior

2. Discrete VAE latent representations
   - Vector quantization
   - Gumbel-softmax for discrete VAE latents

Outline

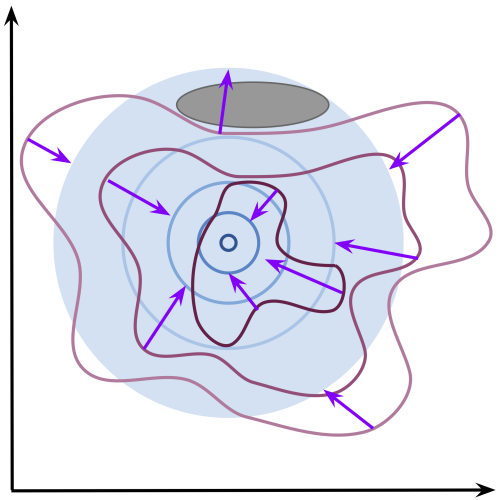
1. Learnable VAE prior

2. Discrete VAE latent representations
   - Vector quantization
   - Gumbel-softmax for discrete VAE latents

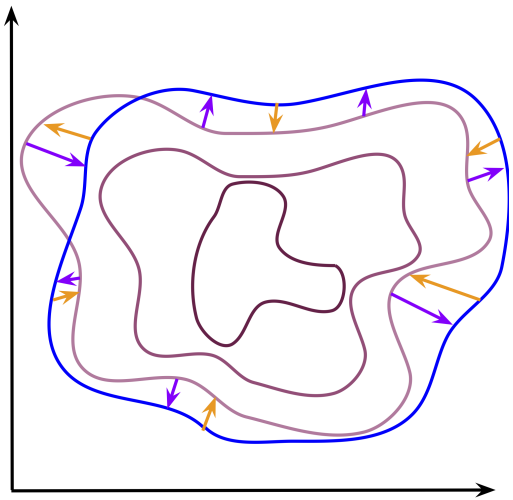
Optimal VAE prior

- Standard Gaussian $p(\mathbf{z}) = \mathcal{N}(0, \mathbf{I}) \Rightarrow$ over-regularization;
- $p(\mathbf{z}) = q_{\text{agg}}(\mathbf{z}|\phi) = \frac{1}{n}\sum_{i=1}^{n} q(\mathbf{z}|\mathbf{x}_i, \phi) \Rightarrow$ overfitting and highly expensive.

Non learnable prior $p(\mathbf{z})$



Learnable prior $p(\mathbf{z}|\boldsymbol{\lambda})$



ELBO revisiting

$$\frac{1}{n}\sum_{i=1}^{n}\mathcal{L}_i(\phi, \theta) = \text{RL} - \text{MI} - KL(q_{\text{agg}}(\mathbf{z}|\phi)||p(\mathbf{z}|\boldsymbol{\lambda}))$$

It is Forward KL with respect to $p(\mathbf{z}|\boldsymbol{\lambda})$.

---

image credit: https://jmtomczak.github.io/blog/7/7_priors.html

NF-based VAE prior

NF model in latent space

$$\log p(\mathbf{z}|\boldsymbol{\lambda}) = \log p(\mathbf{z}^*) + \log \left| \det \left(\frac{d\mathbf{z}^*}{d\mathbf{z}} \right) \right| = \log p(\mathbf{f}_{\boldsymbol{\lambda}}(\mathbf{z})) + \log \left| \det(\mathbf{J}_{\mathbf{f}}) \right|$$

$$\mathbf{z} = \mathbf{g}_{\boldsymbol{\lambda}}(\mathbf{z}^*) = \mathbf{f}_{\boldsymbol{\lambda}}^{-1}(\mathbf{z}^*)$$

- RealNVP with coupling layers.
- Autoregressive NF (fast $\mathbf{f}_{\boldsymbol{\lambda}}(\mathbf{z})$, slow $\mathbf{g}_{\boldsymbol{\lambda}}(\mathbf{z}^*)$).

ELBO with NF-based VAE prior

$$\mathcal{L}(\boldsymbol{\phi}, \boldsymbol{\theta}) = \mathbb{E}_{q(\mathbf{z}|\mathbf{x},\boldsymbol{\phi})} \left[\log p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) + \log p(\mathbf{z}|\boldsymbol{\lambda}) - \log q(\mathbf{z}|\mathbf{x}, \boldsymbol{\phi}) \right]$$

$$= \mathbb{E}_{q(\mathbf{z}|\mathbf{x},\boldsymbol{\phi})} \Big[\log p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) + \underbrace{\Big(\log p(\mathbf{f}_{\boldsymbol{\lambda}}(\mathbf{z})) + \log \left| \det(\mathbf{J}_{\mathbf{f}}) \right| \Big)}_{\text{NF-based prior}} - \log q(\mathbf{z}|\mathbf{x}, \boldsymbol{\phi}) \Big]$$

Chen X. et al. Variational Lossy Autoencoder, 2016

Outline

1. Learnable VAE prior

2. Discrete VAE latent representations
   - Vector quantization
   - Gumbel-softmax for discrete VAE latents

Discrete VAE latents

Motivation

- Previous VAE models had **continuous** latent variables $\mathbf{z}$.
- **Discrete** representations $\mathbf{z}$ are potentially a more natural fit for many of the modalities.
- Powerful autoregressive models (like PixelCNN) have been developed for modelling distributions over discrete variables.
- All cool transformer-like models work with discrete tokens.

ELBO

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

- Reparametrization trick to get unbiased gradients.
- Normal assumptions for $q(\mathbf{z}|\mathbf{x}, \phi)$ and $p(\mathbf{z})$ to compute KL analytically.

Discrete VAE latents

Assumptions

- Let $c \sim \text{Categorical}(\boldsymbol{\pi})$, where
$$\boldsymbol{\pi} = (\pi_1, \dots, \pi_K), \quad \pi_k = P(c = k), \quad \sum_{k=1}^{K} \pi_k = 1.$$
- Let VAE model has discrete latent representation c with prior $p(c) = \text{Uniform}\{1, \dots, K\}$.

ELBO

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

$$KL(q(c|\mathbf{x}, \boldsymbol{\phi})||p(c)) = \sum_{k=1}^{K} q(k|\mathbf{x}, \boldsymbol{\phi}) \log \frac{q(k|\mathbf{x}, \boldsymbol{\phi})}{p(k)} =$$

$$= \sum_{k=1}^{K} q(k|\mathbf{x}, \boldsymbol{\phi}) \log q(k|\mathbf{x}, \boldsymbol{\phi}) - \sum_{k=1}^{K} q(k|\mathbf{x}, \boldsymbol{\phi}) \log p(k) =$$

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

Discrete VAE latents

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

- Our encoder should output discrete distribution $q(c|\mathbf{x}, \phi)$.
- We need the analogue of the reparametrization trick for the discrete distribution $q(c|\mathbf{x}, \phi)$.
- Our decoder $p(\mathbf{x}|c, \theta)$ should input discrete random variable c.

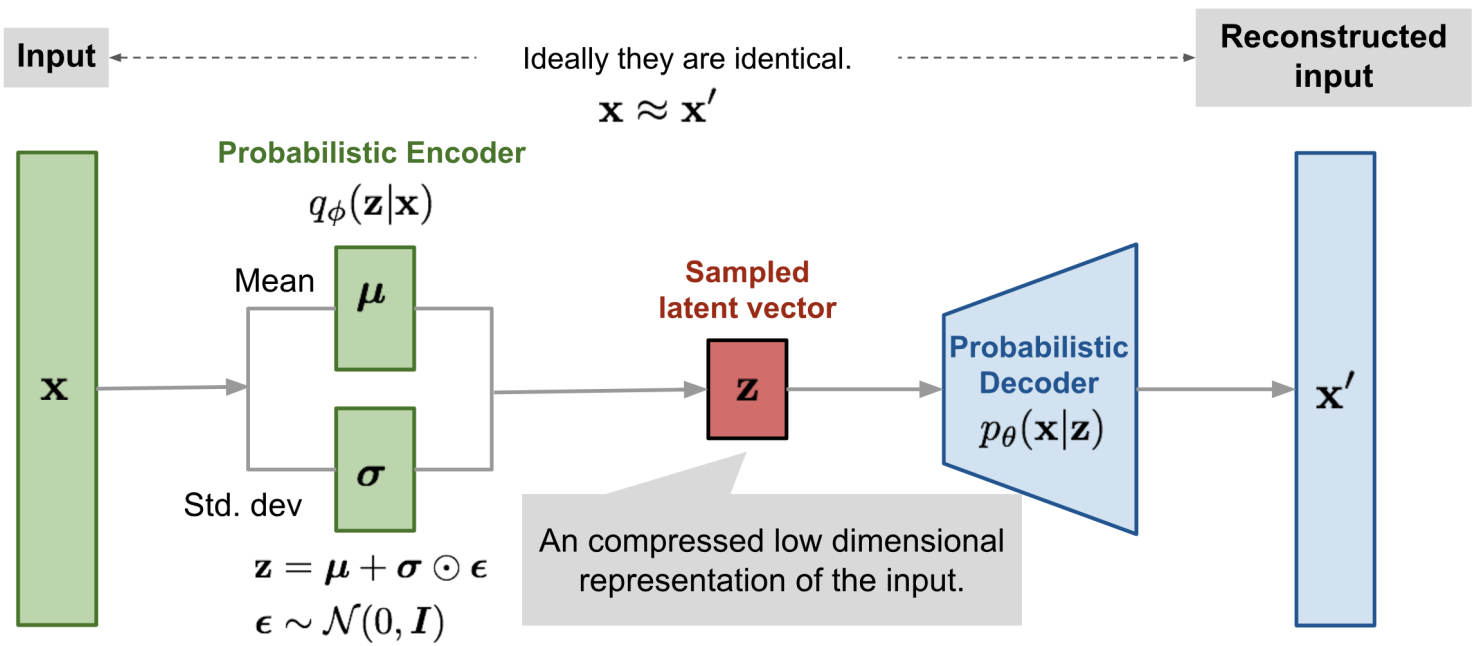


image credit:
https://lilianweng.github.io/lil-log/2018/08/12/from-autoencoder-to-beta-vae.html

Outline

1. Learnable VAE prior

2. Discrete VAE latent representations
   - Vector quantization
   - Gumbel-softmax for discrete VAE latents

Vector quantization

Define the dictionary space $\{\mathbf{e}_k\}_{k=1}^K$, where $\mathbf{e}_k \in \mathbb{R}^C$, K is the size of the dictionary.

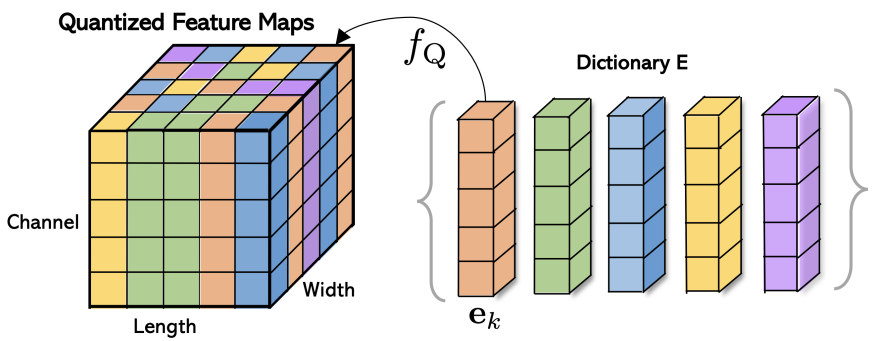
Quantized representation

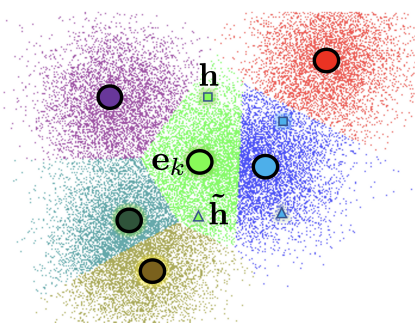
$\mathbf{z}_q \in \mathbb{R}^C$ for $\mathbf{z} \in \mathbb{R}^C$ is defined by a nearest neighbor look-up using the dictionary space

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

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 our encoder outputs continuous representation $\mathbf{z}_e = \text{NN}_{e,\phi}(\mathbf{x}) \in \mathbb{R}^C$.
- Quantization will give us the deterministic mapping from the encoder output $\mathbf{z}_e$ to its quantized representation $\mathbf{z}_q$.
- Let use the dictionary elements $\mathbf{e}_c$ in the generative distribution $p(\mathbf{x}|\mathbf{e}_c, \boldsymbol{\theta})$ (decoder).

Deterministic variational posterior

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

$$KL(q(c|\mathbf{x}, \phi)||p(c)) = -\underbrace{H(q(c|\mathbf{x}, \phi))}_{=0} + \log K = \log K.$$

Generalization to the spatial dimension: $\mathbf{c} \in \{1, \dots, K\}^{W \times H}$

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

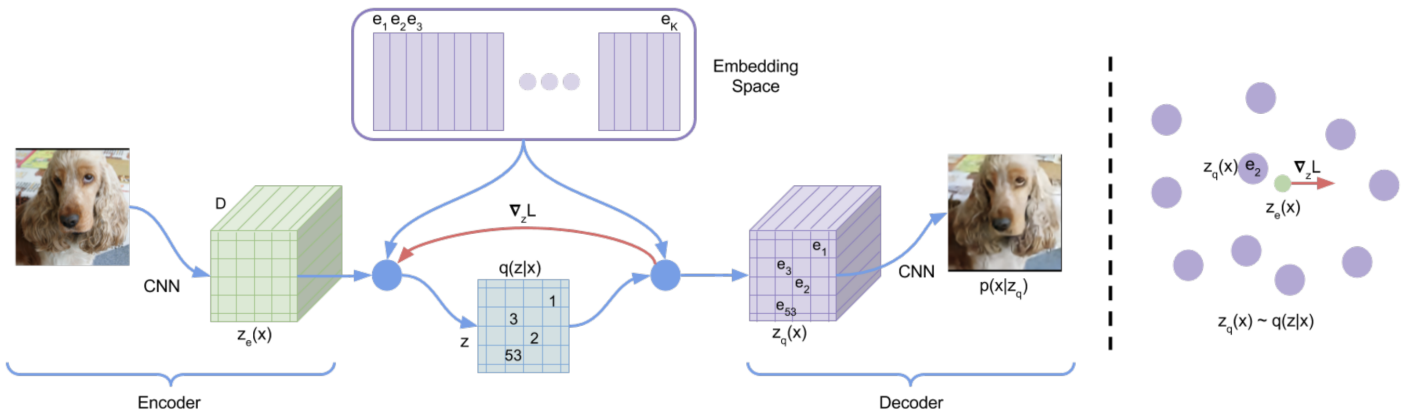
Oord A., Vinyals O., Kavukcuoglu K. Neural Discrete Representation Learning, 2017

Vector Quantized VAE (VQ-VAE)

ELBO

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

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

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

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. Learnable VAE prior

2. Discrete VAE latent representations
   - Vector quantization
   - Gumbel-softmax for discrete VAE latents

Discrete probabilistic VAE encoder

- 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.

How to make the model with probabilistic encoder?

- Variational posterior $q(c|\mathbf{x}, \phi) = \text{Categorical}(\boldsymbol{\pi}_\phi(\mathbf{x}))$ (encoder) outputs discrete probabilities vector $\boldsymbol{\pi}_\phi(\mathbf{x}) = \text{Softmax}(\text{NN}_{e,\phi}(\mathbf{x}))$.
- We sample c^* from $q(c|\mathbf{x}, \phi)$ (reparametrization trick analogue).
- We use $\mathbf{e}_{c^*}$ in the generative distribution $p(\mathbf{x}|\mathbf{e}_{c^*}, \boldsymbol{\theta})$ (decoder).

Problem: Reparametrization trick does not work in this case. Non-differentiable sampling operation depends on the parameters $\boldsymbol{\pi}_\phi(\mathbf{x})$.

Gumbel-max trick

Gumbel distribution

$$g \sim \text{Gumbel}(0, 1) \quad \Leftrightarrow \quad g = -\log(-\log u),\ u \sim \text{Uniform}[0, 1]$$

Theorem

Let $g_k \sim \text{Gumbel}(0, 1)$ for $k = 1, \ldots, K$. Then a discrete random variable

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

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

- We could sample from the discrete distribution using Gumbel-max reparametrization.
- Here parameters and random variable sampling are separated (reparametrization trick). We could apply LOTUS trick.

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**

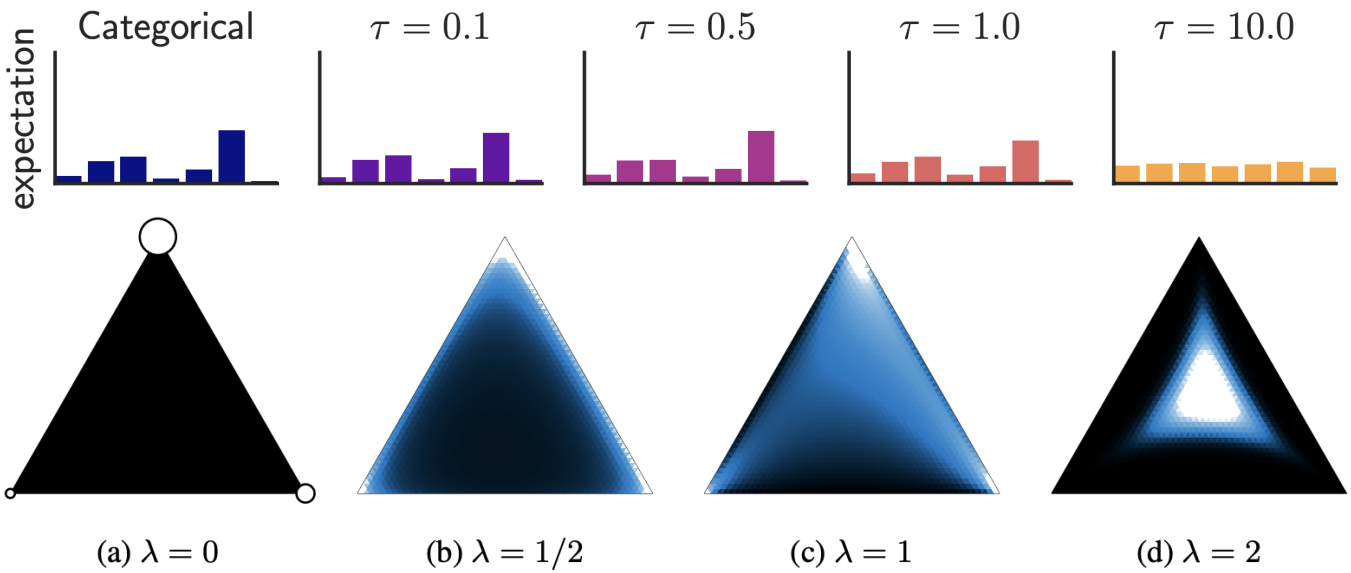
$$\hat{\mathbf{c}} = \text{Softmax}\left(\frac{\log q(\mathbf{c}|\mathbf{x}, \phi) + \mathbf{g}}{\tau}\right)$$

Here τ is a temperature parameter. Now we have differentiable operation, but the gradient estimator 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

$$\hat{\mathbf{c}} = \text{Softmax}\left(\frac{\log q(\mathbf{c}|\mathbf{x}, \phi) + \mathbf{g}}{\tau}\right)$$



(a) $\lambda = 0$ (b) $\lambda = 1/2$ (c) $\lambda = 1$ (d) $\lambda = 2$

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).

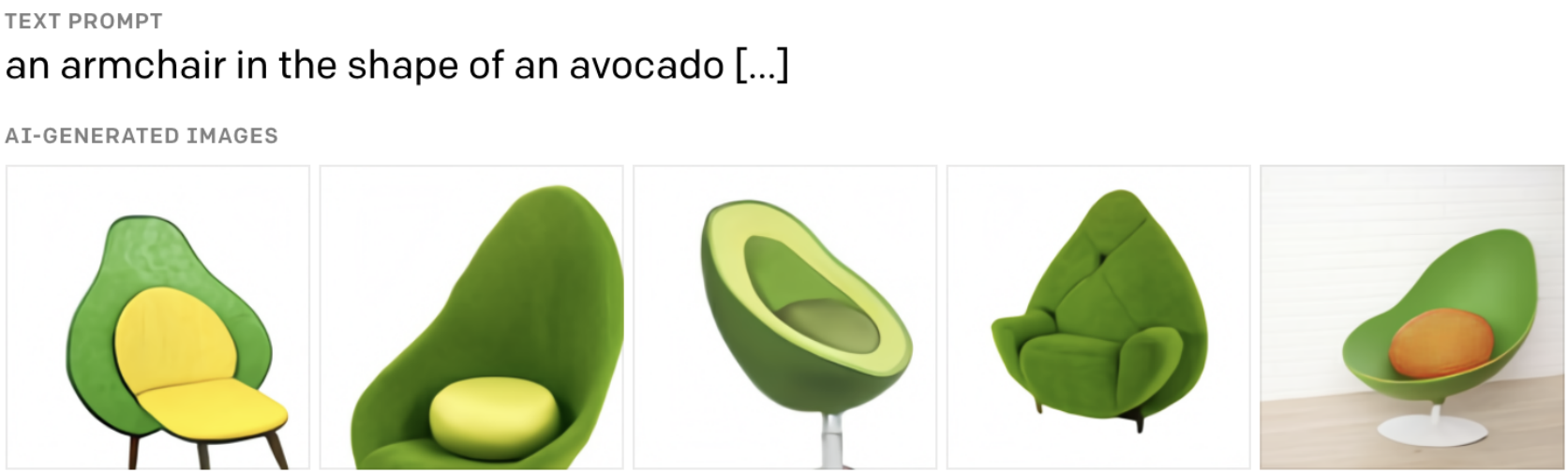
Maddison C. J., Mnih A., Teh Y. W. The Concrete distribution: A continuous relaxation of discrete random variables, 2016
Jang E., Gu S., Poole B. Categorical reparameterization with Gumbel-Softmax, 2016

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

Summary

- We could use NF-based prior in VAE (even autoregressive).
- 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.