Deep Generative Models

Lecture 7

Roman Isachenko

Moscow Institute of Physics and Technology

2023, Autumn

Recap of previous lecture

Theorem

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

ELBO surgery

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

Optimal prior

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

The optimal prior distribution $p(\mathbf{z})$ is aggregated posterior $q(\mathbf{z})$.

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

Recap of previous lecture

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

ELBO revisiting

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

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

ELBO with flow-based VAE prior

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

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

$$\mathbf{z} = f^{-1}(\mathbf{z}^*, \boldsymbol{\lambda}) = g(\mathbf{z}^*, \boldsymbol{\lambda}), \quad \mathbf{z}^* \sim p(\mathbf{z}^*) = \mathcal{N}(0, 1)$$

Chen X. et al. Variational Lossy Autoencoder, 2016

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

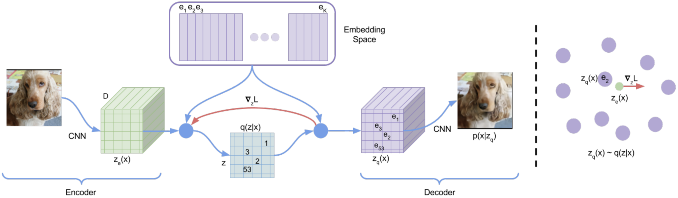
$$\mathcal{L}(\phi, \boldsymbol{\theta}) = \mathbb{E}_{q(c|\mathbf{x},\phi)} \log p(\mathbf{x}|c, \boldsymbol{\theta}) - KL(q(c|\mathbf{x}, \phi)||p(c)) \rightarrow \max_{\phi,\boldsymbol{\theta}}.$$

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!).

Recap of previous lecture



Deterministic variational posterior

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

ELBO

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

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

Outline

Gumbel-softmax for discrete VAE latents

1. Likelihood-free learning

2. Generative adversarial networks (GAN)

3. Adversatial variational Bayes

Outline

Gumbel-softmax for discrete VAE latents

1. Likelihood-free learning
2. Generative adversarial networks (GAN)
3. Adversatial variational Bayes

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**

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

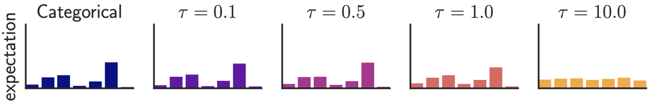
Here τ is a temperature parameter. Now we have differentiable operation, but the gradient estimate is biased now.

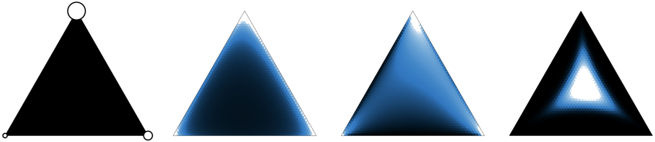
Maddison C. J., Mnih A., Teh Y. W. The Concrete distribution: A continuous relaxation of discrete random variables, 2016

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

Gumbel-softmax trick

Concrete distribution





(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).

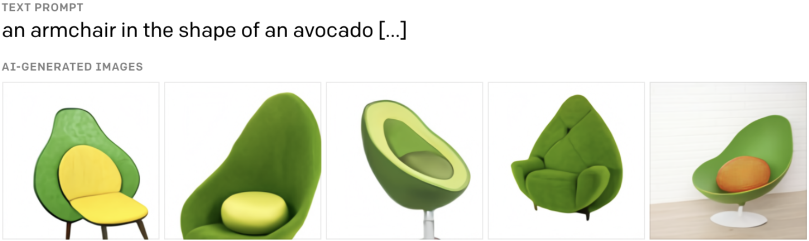
Maddison C. J., Mnih A., Teh Y. W. The Concrete distribution: A continuous relaxation of discrete random variables, 2016
Jang E., Gu S., Poole B. Categorical reparameterization with Gumbel-Softmax, 2016

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.



TEXT PROMPT

an armchair in the shape of an avocado [...]

AI-GENERATED IMAGES

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

Outline

Gumbel-softmax for discrete VAE latents

1. Likelihood-free learning

2. Generative adversarial networks (GAN)

3. Adversatial variational Bayes

Likelihood based models

Poor likelihood
Great samples

$$p_1(\mathbf{x}) = \frac{1}{n} \sum_{i=1}^{n} \mathcal{N}(\mathbf{x}|\mathbf{x}_i, \epsilon \mathbf{I})$$

For small ϵ this model will generate samples with great quality, but likelihood will be very poor.

Great likelihood
Poor samples

$$p_2(\mathbf{x}) = 0.01 p(\mathbf{x}) + 0.99 p_{\text{noise}}(\mathbf{x})$$

$$\log \left[0.01 p(\mathbf{x}) + 0.99 p_{\text{noise}}(\mathbf{x})\right] \geq$$
$$\geq \log \left[0.01 p(\mathbf{x})\right] = \log p(\mathbf{x}) - \log 100$$

Noisy irrelevant samples, but for high dimensions $\log p(\mathbf{x})$ becomes proportional to m.

- Likelihood is not a perfect quality measure for generative model.
- Likelihood could be intractable.

Theis L., Oord A., Bethge M. A note on the evaluation of generative models, 2015

Likelihood-free learning

Where did we start

We would like to approximate true data distribution $\pi(\mathbf{x})$. Instead of searching true $\pi(\mathbf{x})$ over all probability distributions, learn function approximation $p(\mathbf{x}|\boldsymbol{\theta}) \approx \pi(\mathbf{x})$.

Imagine we have two sets of samples

- $\mathcal{S}_1 = \{\mathbf{x}_i\}_{i=1}^{n_1} \sim \pi(\mathbf{x})$ – real samples;
- $\mathcal{S}_2 = \{\mathbf{x}_i\}_{i=1}^{n_2} \sim p(\mathbf{x}|\boldsymbol{\theta})$ – generated (or fake) samples.

Let define discriminative model (classifier):

$$p(y = 1|\mathbf{x}) = P\bigl(\{\mathbf{x} \sim \pi(\mathbf{x})\}\bigr); \quad p(y = 0|\mathbf{x}) = P\bigl(\{\mathbf{x} \sim p(\mathbf{x}|\boldsymbol{\theta})\}\bigr)$$

Assumption

Generative distribution $p(\mathbf{x}|\boldsymbol{\theta})$ equals to the true distribution $\pi(\mathbf{x})$ if we can not distinguish them using discriminative model $p(y|\mathbf{x})$. It means that $p(y = 1|\mathbf{x}) = 0.5$ for each sample $\mathbf{x}$.

Outline

Gumbel-softmax for discrete VAE latents

1. Likelihood-free learning

2. Generative adversarial networks (GAN)

3. Adversatial variational Bayes

Likelihood-free learning

The more powerful discriminative model we will have, the more likely we got the "best" generative distribution $p(\mathbf{x}|\boldsymbol{\theta})$.

The most common way to learn a classifier is to minimize cross entropy loss.

- **Generator:** generative model $\mathbf{x} = G(\mathbf{z})$, which makes generated sample more realistic. Here $\mathbf{z}$ comes from the base (known) distribution $p(\mathbf{z})$ and $\mathbf{x} \sim p(\mathbf{x}|\boldsymbol{\theta})$. Generator tries to **maximize** cross entropy.
- **Discriminator:** a classifier $p(y = 1|\mathbf{x}) = D(\mathbf{x}) \in [0, 1]$, which distinguishes real samples from generated samples. Discriminator tries to **minimize** cross entropy (tries to enhance discriminative model).

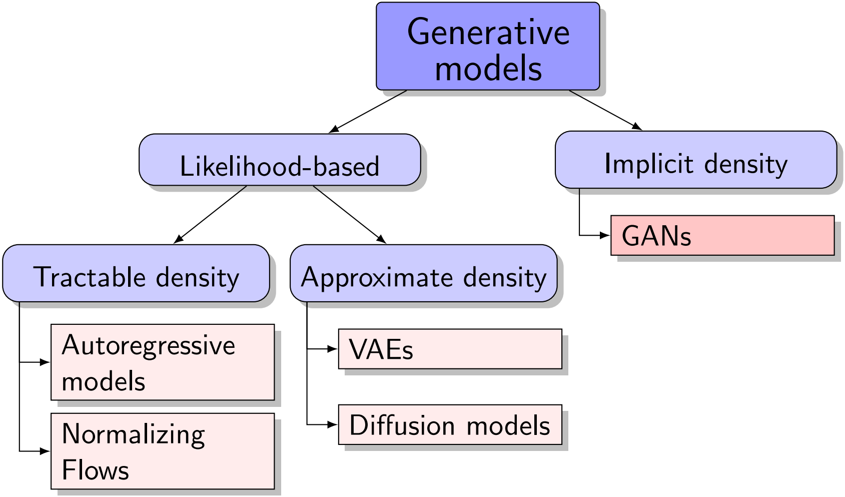
Generative adversarial network (GAN) objective

$$\min_G \max_D \left[\mathbb{E}_{\pi(\mathbf{x})} \log D(\mathbf{x}) + \mathbb{E}_{p(\mathbf{x}|\boldsymbol{\theta})} \log(1 - D(\mathbf{x}))\right]$$

$$\min_G \max_D \left[\mathbb{E}_{\pi(\mathbf{x})} \log D(\mathbf{x}) + \mathbb{E}_{p(\mathbf{z})} \log(1 - D(G(\mathbf{z})))\right]$$

Goodfellow I. J. et al. Generative Adversarial Networks, 2014

Generative models zoo



Generative models

Likelihood-based

Implicit density

GANs

Tractable density

Approximate density

Autoregressive models

Normalizing Flows

VAEs

Diffusion models

GAN optimality

Theorem

The minimax game

$$\min_G \max_D \Bigl[\underbrace{\mathbb{E}_{\pi(\mathbf{x})} \log D(\mathbf{x}) + \mathbb{E}_{p(\mathbf{z})} \log(1 - D(G(\mathbf{z})))}_{V(G,D)}\Bigr]$$

has the global optimum $\pi(\mathbf{x}) = p(\mathbf{x}|\boldsymbol{\theta})$, in this case $D^*(\mathbf{x}) = 0.5$.

Proof (fixed G)

$$\begin{aligned} V(G, D) &= \mathbb{E}_{\pi(\mathbf{x})} \log D(\mathbf{x}) + \mathbb{E}_{p(\mathbf{x}|\boldsymbol{\theta})} \log(1 - D(\mathbf{x})) \\ &= \int \underbrace{[\pi(\mathbf{x}) \log D(\mathbf{x}) + p(\mathbf{x}|\boldsymbol{\theta}) \log(1 - D(\mathbf{x})]}_{y(D)} d\mathbf{x} \end{aligned}$$

$$\frac{dy(D)}{dD} = \frac{\pi(\mathbf{x})}{D(\mathbf{x})} - \frac{p(\mathbf{x}|\boldsymbol{\theta})}{1 - D(\mathbf{x})} = 0 \quad \Rightarrow \quad D^*(\mathbf{x}) = \frac{\pi(\mathbf{x})}{\pi(\mathbf{x}) + p(\mathbf{x}|\boldsymbol{\theta})}$$

Goodfellow I. J. et al. Generative Adversarial Networks, 2014

GAN optimality

Proof continued (fixed $D = D^*$)

$$V(G, D^*) = \mathbb{E}_{\pi(\mathbf{x})} \log \frac{\pi(\mathbf{x})}{\pi(\mathbf{x}) + p(\mathbf{x}|\boldsymbol{\theta})} + \mathbb{E}_{p(\mathbf{x}|\boldsymbol{\theta})} \log \frac{p(\mathbf{x}|\boldsymbol{\theta})}{\pi(\mathbf{x}) + p(\mathbf{x}|\boldsymbol{\theta})}$$

$$= KL\left(\pi(\mathbf{x}) || \frac{\pi(\mathbf{x}) + p(\mathbf{x}|\boldsymbol{\theta})}{2}\right) + KL\left(p(\mathbf{x}|\boldsymbol{\theta}) || \frac{\pi(\mathbf{x}) + p(\mathbf{x}|\boldsymbol{\theta})}{2}\right) - 2\log 2$$

$$= 2JSD(\pi(\mathbf{x}) || p(\mathbf{x}|\boldsymbol{\theta})) - 2\log 2.$$

Jensen-Shannon divergence (symmetric KL divergence)

$$JSD(\pi(\mathbf{x}) || p(\mathbf{x}|\boldsymbol{\theta})) = \frac{1}{2}\left[KL\left(\pi(\mathbf{x}) || \frac{\pi(\mathbf{x}) + p(\mathbf{x}|\boldsymbol{\theta})}{2}\right) + KL\left(p(\mathbf{x}|\boldsymbol{\theta}) || \frac{\pi(\mathbf{x}) + p(\mathbf{x}|\boldsymbol{\theta})}{2}\right)\right]$$

Could be used as a distance measure!

$$V(G^*, D^*) = -2\log 2, \quad \pi(\mathbf{x}) = p(\mathbf{x}|\boldsymbol{\theta}), \quad D^*(\mathbf{x}) = 0.5.$$

Goodfellow I. J. et al. Generative Adversarial Networks, 2014

GAN optimality

Theorem

The minimax game

$$\min_{G} \max_{D} \Big[\underbrace{\mathbb{E}_{\pi(\mathbf{x})} \log D(\mathbf{x}) + \mathbb{E}_{p(\mathbf{z})} \log(1 - D(G(\mathbf{z})))}_{V(G,D)} \Big]$$

has the global optimum $\pi(\mathbf{x}) = p(\mathbf{x}|\boldsymbol{\theta})$, in this case $D^*(\mathbf{x}) = 0.5$.

Expectations

If the generator could be **any** function and the discriminator is **optimal** at every step, then the generator is **guaranteed to converge** to the data distribution.

Reality

- Generator updates are made in parameter space, discriminator is not optimal at every step.
- Generator and discriminator loss keeps oscillating during GAN training.

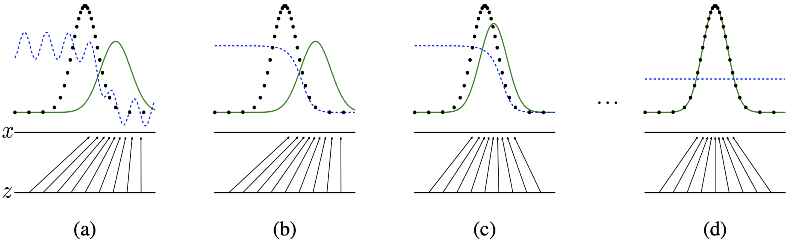
Goodfellow I. J. et al. Generative Adversarial Networks, 2014

GAN training

Let futher assume that generator and discriminator are parametric models: $D(\mathbf{x}, \phi)$ and $G(\mathbf{z}, \boldsymbol{\theta})$.

Objective

$$\min_{\boldsymbol{\theta}} \max_{\boldsymbol{\phi}} \left[\mathbb{E}_{\pi(\mathbf{x})} \log D(\mathbf{x}, \phi) + \mathbb{E}_{p(\mathbf{z})} \log(1 - D(G(\mathbf{z}, \boldsymbol{\theta}), \phi))\right]$$

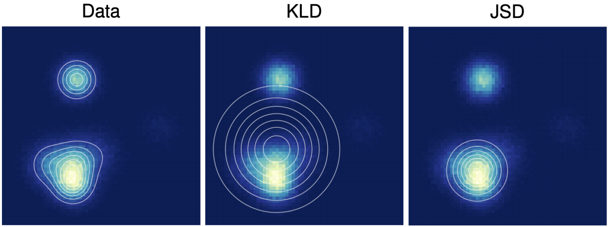


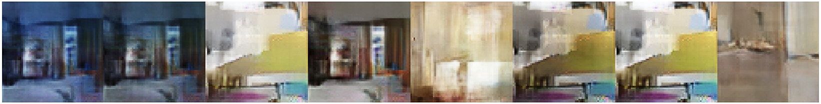
- $\mathbf{z} \sim p(\mathbf{z})$ is a latent variable.
- $p(\mathbf{x}|\mathbf{z}, \boldsymbol{\theta}) = \delta(\mathbf{x} - G(\mathbf{z}, \boldsymbol{\theta}))$ is deterministic decoder (like NF).
- We do not have encoder at all.

Goodfellow I. J. et al. Generative Adversarial Networks, 2014

Mode collapse

The phenomena where the generator of a GAN collapses to one or few distribution modes.





Alternate architectures, adding regularization terms, injecting small noise perturbations and other millions bags and tricks are used to avoid the mode collapse.

Goodfellow I. J. et al. Generative Adversarial Networks, 2014
Metz L. et al. Unrolled Generative Adversarial Networks, 2016

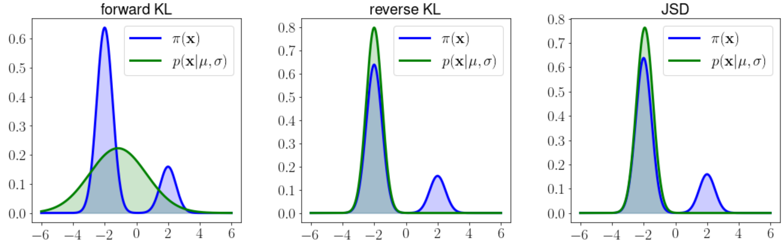
Jensen-Shannon vs Kullback-Leibler

- $\pi(\mathbf{x})$ is a fixed mixture of 2 gaussians.
- $p(\mathbf{x}|\mu,\sigma) - \mathcal{N}(\mu,\sigma^2)$.

Mode covering vs mode seeking

$$KL(\pi||p) = \int \pi(\mathbf{x}) \log \frac{\pi(\mathbf{x})}{p(\mathbf{x})} d\mathbf{x}, \quad KL(p||\pi) = \int p(\mathbf{x}) \log \frac{p(\mathbf{x})}{\pi(\mathbf{x})} d\mathbf{x}$$

$$JSD(\pi||p) = \frac{1}{2}\left[KL\left(\pi(\mathbf{x})||\frac{\pi(\mathbf{x})+p(\mathbf{x})}{2}\right) + KL\left(p(\mathbf{x})||\frac{\pi(\mathbf{x})+p(\mathbf{x})}{2}\right)\right]$$



Outline

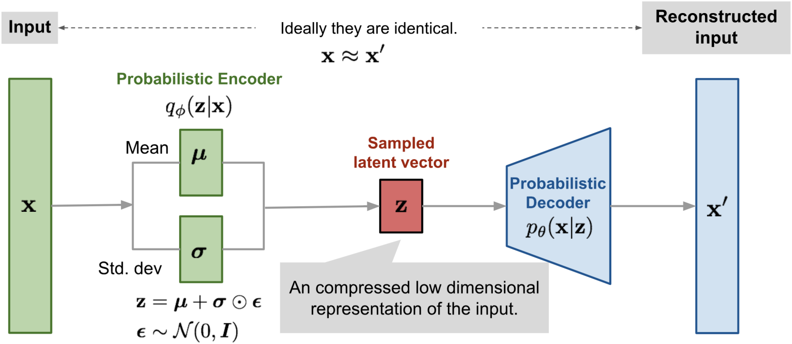
Gumbel-softmax for discrete VAE latents

1. Likelihood-free learning

2. Generative adversarial networks (GAN)

3. Adversatial variational Bayes

VAE recap


Input
Ideally they are identical.
$\mathbf{x} \approx \mathbf{x}'$
Reconstructed input
Probabilistic Encoder
$q_\phi(\mathbf{z}|\mathbf{x})$
$\mathbf{x}$
Mean
$\boldsymbol{\mu}$
Std. dev
$\boldsymbol{\sigma}$
$\mathbf{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$
$\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I})$
Sampled latent vector
$\mathbf{z}$
An compressed low dimensional representation of the input.
Probabilistic Decoder
$p_\theta(\mathbf{x}|\mathbf{z})$
$\mathbf{x}'$

- Encoder $q(\mathbf{z}|\mathbf{x}, \phi) = \mathcal{N}(\mathbf{z}|\boldsymbol{\mu}_\phi(\mathbf{x}), \boldsymbol{\sigma}_\phi(\mathbf{x}))$.
- Variational posterior $q(\mathbf{z}|\mathbf{x}, \phi)$ originally approximates the true posterior $p(\mathbf{z}|\mathbf{x}, \boldsymbol{\theta})$.
- Which methods are you already familiar with to make the posterior is more flexible?

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

Adversarial Variational Bayes

ELBO objective

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

What is the problem to make the variational posterior model an implicit model?

- The first term is the reconstruction loss that needs only samples from $q(\mathbf{z}|\mathbf{x}, \phi)$ to evaluate.
- Reparametrization trick allows to get gradients of the reconstruction loss

$$\nabla_\phi \int q(\mathbf{z}|\mathbf{x}, \phi) f(\mathbf{z}) d\mathbf{z} = \nabla_\phi \int r(\boldsymbol{\epsilon}) f(\mathbf{z}) d\boldsymbol{\epsilon}$$
$$= \int r(\boldsymbol{\epsilon}) \nabla_\phi f(g(\mathbf{x}, \boldsymbol{\epsilon}, \phi)) d\boldsymbol{\epsilon} \approx \nabla_\phi f(g(\mathbf{x}, \boldsymbol{\epsilon}^*, \phi)),$$

where $\boldsymbol{\epsilon}^* \sim r(\boldsymbol{\epsilon}), \quad \mathbf{z} = g(\mathbf{x}, \boldsymbol{\epsilon}, \phi), \quad \mathbf{z} \sim q(\mathbf{z}|\mathbf{x}, \phi).$

Mescheder L., Nowozin S., Geiger A. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks, 2017

Adversarial Variational Bayes

ELBO objective

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

What is the problem to make the variational posterior model an implicit model?

- The second term requires the explicit value of $q(\mathbf{z}|\mathbf{x}, \phi)$.
- We could join second and third terms:
$$KL = \mathbb{E}_{q(\mathbf{z}|\mathbf{x},\phi)} \log \frac{q(\mathbf{z}|\mathbf{x}, \phi)}{p(\mathbf{z})} = \mathbb{E}_{q(\mathbf{z}|\mathbf{x},\phi)} \log \frac{q(\mathbf{z}|\mathbf{x}, \phi)\pi(\mathbf{x})}{p(\mathbf{z})\pi(\mathbf{x})}.$$
- We have to estimate density ratio
$$r(\mathbf{x}, \mathbf{z}) = \frac{q_1(\mathbf{x}, \mathbf{z})}{q_2(\mathbf{x}, \mathbf{z})} = \frac{q(\mathbf{z}|\mathbf{x}, \phi)\pi(\mathbf{x})}{p(\mathbf{z})\pi(\mathbf{x})}.$$

Mescheder L., Nowozin S., Geiger A. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks, 2017

Density ratio trick

Consider two distributions $q_1(\mathbf{x})$, $q_2(\mathbf{x})$ and probabilistic model

$$p(\mathbf{x}|y) = \begin{cases} q_1(\mathbf{x}), & \text{if } y = 1, \\ q_2(\mathbf{x}), & \text{if } y = 0, \end{cases} \qquad y \sim \text{Bern}(0.5).$$

Density ratio

$$\frac{q_1(\mathbf{x})}{q_2(\mathbf{x})} = \frac{p(\mathbf{x}|y=1)}{p(\mathbf{x}|y=0)} = \frac{p(y=1|\mathbf{x})p(\mathbf{x})}{p(y=1)} \Bigg/ \frac{p(y=0|\mathbf{x})p(\mathbf{x})}{p(y=0)} =$$

$$= \frac{p(y=1|\mathbf{x})}{p(y=0|\mathbf{x})} = \frac{p(y=1|\mathbf{x})}{1 - p(y=1|\mathbf{x})} = \frac{D(\mathbf{x})}{1 - D(\mathbf{x})}$$

Here $D(\mathbf{x})$ is a discriminator model the output of which is a probability that $\mathbf{x}$ is a sample from $q_1(\mathbf{x})$ rather than from $q_2(\mathbf{x})$.

$$\max_D \left[\mathbb{E}_{q_1(\mathbf{x})} \log D(\mathbf{x}) + \mathbb{E}_{q_2(\mathbf{x})} \log(1 - D(\mathbf{x}))\right]$$

Mescheder L., Nowozin S., Geiger A. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks, 2017

Density ratio trick

$$r(\mathbf{x}, \mathbf{z}) = \frac{q_1(\mathbf{x}, \mathbf{z})}{q_2(\mathbf{x}, \mathbf{z})} = \frac{q(\mathbf{z}|\mathbf{x}, \phi)\pi(\mathbf{x})}{p(\mathbf{z})\pi(\mathbf{x})}.$$

Density ratio

$$\frac{q_1(\mathbf{x}, \mathbf{z})}{q_2(\mathbf{x}, \mathbf{z})} = \frac{p(y = 1|\mathbf{x}, \mathbf{z})}{1 - p(y = 1|\mathbf{x}, \mathbf{z})} = \frac{D(\mathbf{x}, \mathbf{z})}{1 - D(\mathbf{x}, \mathbf{z})}$$

Adversarial Variational Bayes

$$\max_D \left[\mathbb{E}_{\pi(\mathbf{x})}\mathbb{E}_{q(\mathbf{z}|\mathbf{x},\phi)} \log D(\mathbf{x}, \mathbf{z}) + \mathbb{E}_{\pi(\mathbf{x})}\mathbb{E}_{p(\mathbf{z})} \log(1 - D(\mathbf{x}, \mathbf{z}))\right]$$

Monte-Carlo estimation for KL divergence:

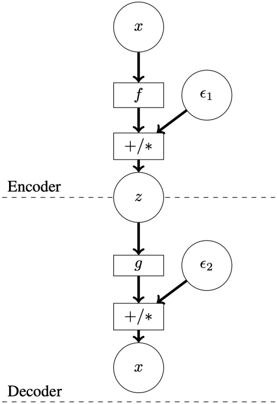
$$KL(q(\mathbf{z}|\mathbf{x}, \phi)||p(\mathbf{z})) \approx \frac{D(\mathbf{x}, \mathbf{z})}{1 - D(\mathbf{x}, \mathbf{z})}.$$

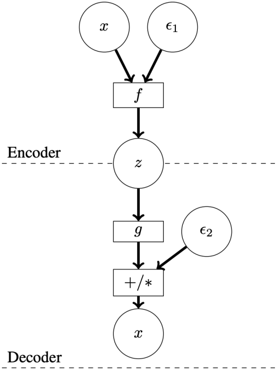
Mescheder L., Nowozin S., Geiger A. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks, 2017

Adversarial Variational Bayes

ELBO objective

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





Mescheder L., Nowozin S., Geiger A. Adversarial variational bayes: Unifying variational autoencoders and generative adversarial networks, 2017

Summary

- 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.
- Likelihood is not a perfect criteria to measure quality of generative model.
- Adversarial learning suggests to solve minimax problem to match the distributions.
- GAN tries to optimize Jensen-Shannon divergence (in theory).
- Mode collapse and vanishing gradients are the two main problems of vanilla GAN. Lots of tips and tricks has to be used to make the GAN training is stable and scalable.
- Adversarial Variational Bayes uses density ratio trick to get more powerful variational posterior.