Discrete Deep Generative Models

Volodymyr Kuleshov

Cornell Tech

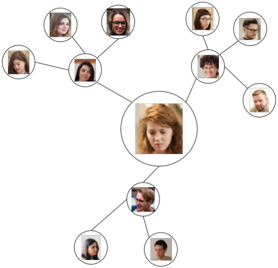
Lecture 16

Announcements

- Assignment 3 is due tonight
- Project progress report is due at the end of the month.
- Wednesday lecture will be on Zoom.
- Guest lecture by Hugging Face next Monday (also on Zoom).

Why should we care about discreteness?

- Many data modalities are inherently discrete
  - Graphs



  - Text, DNA Sequences, Program Source Code, Molecules, and lots more
- Many *latent* representations are inherently discrete: binary concepts (man or woman), syntactic parse trees, compressive codes, etc.

Lecture Outline

1. Discrete Stochastic Optimization
2. REINFORCE Gradient Estimation
   - The REINFORCE Estimator
   - Control Variates
   - Neural Variational Inference & Learning in Belief Networks
3. Relaxed Reparameterization Tricks
   - The Gumbel-Max Trick
   - The Gumbel-Softmax Trick
   - Extensions of Gumbel-Softmax to Combinatorial Objects

Key Challenge: Discrete Stochastic Optimization

A key challenge of discrete modeling is solving optimization problems of the form

$$\max_{\phi} E_{q_\phi(\mathbf{z})}[f(\mathbf{z})]$$

- An example: think of $q(\cdot)$ as the inference distribution for a VAE

$$\max_{\theta,\phi} E_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log \frac{p_\theta(\mathbf{x},\mathbf{z})}{q(\mathbf{z}|\mathbf{x})}\right].$$

- Gradients w.r.t. θ are easy & can be derived via linearity of expectation

$$\begin{aligned}\nabla_\theta E_{q(\mathbf{z};\phi)}[\log p(\mathbf{z},\mathbf{x};\theta) - \log q(\mathbf{z};\phi)] &= E_{q(\mathbf{z};\phi)}[\nabla_\theta \log p(\mathbf{z},\mathbf{x};\theta)] \\ &\approx \frac{1}{k}\sum_k \nabla_\theta \log p(\mathbf{z}^k,\mathbf{x};\theta)\end{aligned}$$

- If $\mathbf{z}$ is continuous, $q(\cdot)$ is reparameterizable, and $f(\cdot)$ is differentiable in ϕ, then we can use reparameterization to compute gradients w.r.t. ϕ
- What if some assumptions fail? Then obtaining gradients w.r.t. ϕ is hard.

Discrete Stochastic Optimization with REINFORCE

Consider the discrete optimization problem

$$\max_{\phi} E_{q_\phi(\mathbf{z})}[f(\mathbf{z})]$$

- For many class of problem scenarios, reparameterization trick is inapplicable
- **Scenario 1:** $f(\cdot)$ is non-differentiable in $\mathbf{z}$ e.g., optimizing a black box reward function in reinforcement learning
- **Scenario 2:** $q_\phi(\mathbf{z})$ cannot be reparameterized as a differentiable function of ϕ with respect to a fixed base distribution e.g., discrete distributions
- REINFORCE is a general-purpose solution to both these scenarios

The REINFORCE Gradient Trick

We want the gradient with respect to ϕ of the expectation of f:

$$E_{q_\phi(\mathbf{z})}[f(\mathbf{z})] = \sum_{\mathbf{z}} q_\phi(\mathbf{z}) f(\mathbf{z})$$

We can use the following trick to obtain an estimator:

$$\begin{aligned}\frac{\partial}{\partial \phi_i} E_{q_\phi(\mathbf{z})}[f(\mathbf{z})] \quad &= \textstyle\sum_{\mathbf{z}} \frac{\partial q_\phi(\mathbf{z})}{\partial \phi_i} f(\mathbf{z}) = \sum_{\mathbf{z}} q_\phi(\mathbf{z}) \frac{1}{q_\phi(\mathbf{z})} \frac{\partial q_\phi(\mathbf{z})}{\partial \phi_i} f(\mathbf{z}) \\ &= \textstyle\sum_{\mathbf{z}} q_\phi(\mathbf{z}) \frac{\partial \log q_\phi(\mathbf{z})}{\partial \phi_i} f(\mathbf{z}) = E_{q_\phi(\mathbf{z})}\left[\frac{\partial \log q_\phi(\mathbf{z})}{\partial \phi_i} f(\mathbf{z})\right]\end{aligned}$$

REINFORCE Gradient Estimation

We want to compute a gradient with respect to ϕ of

$$E_{q_\phi(\mathbf{z})}[f(\mathbf{z})] = \sum_{\mathbf{z}} q_\phi(\mathbf{z}) f(\mathbf{z})$$

The REINFORCE formula for the entire gradient is

$$\nabla_\phi E_{q_\phi(\mathbf{z})}[f(\mathbf{z})] = E_{q_\phi(\mathbf{z})}\left[f(\mathbf{z}) \nabla_\phi \log q_\phi(\mathbf{z})\right]$$

We sample $\mathbf{z}^1, \cdots, \mathbf{z}^K$ from $q_\phi(\mathbf{z})$ and estimate using Monte Carlo:

$$\nabla_\phi E_{q_\phi(\mathbf{z})}[f(\mathbf{z})] \approx \frac{1}{K} \sum_k f(\mathbf{z}^k) \nabla_\phi \log q_\phi(\mathbf{z}^k)$$

- Assumption: The distribution $q(\cdot)$ is easy to sample from and evaluate probabilities
- Works for both discrete and continuous distributions

Discrete-Variable GANs

The training objective for a generator is:

$$\min_G E_{\mathbf{x}\sim p_{\text{data}}}[\log D(\mathbf{x})] + E_{\mathbf{x}\sim p_G}[\log(1 - D(\mathbf{x}))]$$

When $\mathbf{x}$ is continuous, we can usually differentiate through x:

$$\min_G E_{\mathbf{z}\sim p(\mathbf{z})}[\log(1 - D(G(z)))]$$

When $\mathbf{x}$ is discrete, this no longer works, our objective is

$$\min_G E_{\mathbf{x}\sim p_G}[\log(1 - D(\mathbf{x}))]$$

But this is of the same form as $E_{q_\phi(\mathbf{z})}[f(\mathbf{z})]$ and we can deal with that:

- We may use REINFORCE to estimate gradients
- This can be viewed as training an RL agent whose actions are components of $\mathbf{x}$ and states are previously generated components
- See the SeqGAN (Lu et al., 2017) paper for details

REINFORCE Gradient Estimates have High Variance

- Want to compute a gradient with respect to ϕ of

$$E_{q_\phi(\mathbf{z})}[f(\mathbf{z})] = \sum_{\mathbf{z}} q_\phi(\mathbf{z}) f(\mathbf{z})$$

- Monte Carlo estimate: Sample $\mathbf{z}^1, \cdots, \mathbf{z}^K$ from $q_\phi(\mathbf{z})$

$$\nabla_\phi E_{q_\phi(\mathbf{z})}[f(\mathbf{z})] \approx \frac{1}{K} \sum_k f(\mathbf{z}^k) \nabla_\phi \log q_\phi(\mathbf{z}^k) := f_{\mathrm{MC}}(\mathbf{z}^1, \cdots, \mathbf{z}^K)$$

- Monte Carlo estimates of gradients are unbiased

$$E_{\mathbf{z}^1, \cdots, \mathbf{z}^K \sim q_\phi(\mathbf{z})} \left[f_{\mathrm{MC}}(\mathbf{z}^1, \cdots, \mathbf{z}^K) \right] = \nabla_\phi E_{q_\phi(\mathbf{z})}[f(\mathbf{z})]$$

- However, its variance is high! The value $\log q(z)$ can greatly vary in magnitude for small z. In practice, there are better estimates that one can use.

Control Variates

- The REINFORCE rule is

$$\nabla_\phi E_{q_\phi(\mathbf{z})}[f(\mathbf{z})] = E_{q_\phi(\mathbf{z})}\left[f(\mathbf{z})\nabla_\phi \log q_\phi(\mathbf{z})\right]$$

- Given any constant B (a control variate)

$$\nabla_\phi E_{q_\phi(\mathbf{z})}[f(\mathbf{z})] = E_{q_\phi(\mathbf{z})}\left[(f(\mathbf{z}) - B)\nabla_\phi \log q_\phi(\mathbf{z})\right]$$

- To see why,

$$\begin{aligned} E_{q_\phi(\mathbf{z})}\left[B\nabla_\phi \log q_\phi(\mathbf{z})\right] &= B\sum_{\mathbf{z}} q_\phi(\mathbf{z})\nabla_\phi \log q_\phi(\mathbf{z}) = B\sum_{\mathbf{z}} \nabla_\phi q_\phi(\mathbf{z}) \\ &= B\nabla_\phi \sum_{\mathbf{z}} q_\phi(\mathbf{z}) = B\nabla_\phi 1 = 0 \end{aligned}$$

- Monte Carlo gradient estimates of both $f(\mathbf{z})$ and $f(\mathbf{z}) - B$ have same expectation
- These estimates can however have different variances

Control Variates

- Suppose we want to compute

$$E_{q_\phi(\mathbf{z})}[f(\mathbf{z})] = \sum_{\mathbf{z}} q_\phi(\mathbf{z}) f(\mathbf{z})$$

- Define

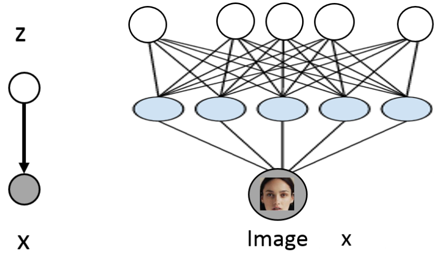
$$\widehat{f}(\mathbf{z}) = f(\mathbf{z}) + a\left(h(\mathbf{z}) - E_{q_\phi(\mathbf{z})}[h(\mathbf{z})]\right)$$

- $h(\mathbf{z})$ is referred to as a control variate
- Assumption: $E_{q_\phi(\mathbf{z})}[h(\mathbf{z})]$ is known
- Monte Carlo gradient estimates of $f(\mathbf{z})$ and $\widehat{f}(\mathbf{z})$ have the same expectation

$$E_{\mathbf{z}^1,\cdots,\mathbf{z}^K \sim q_\phi(\mathbf{z})}[\widehat{f}_{\text{MC}}(\mathbf{z}^1, \cdots, \mathbf{z}^K)] = E_{\mathbf{z}^1,\cdots,\mathbf{z}^K \sim q_\phi(\mathbf{z})}[f_{\text{MC}}(\mathbf{z}^1, \cdots, \mathbf{z}^K)]$$

  but different variances
- Can try to learn and update the control variate during training

Belief Networks



Latent variable models with discrete latent variables are often referred to as belief networks.

1. $\mathbf{z} \sim \text{UnifCat}(K)$
2. Conditional is $p(\mathbf{x} \mid \mathbf{z}; \theta_\psi(z))$ where $\theta_\psi(z)$ parametrize the probability as a function of z via a neural network with parameters ψ.

- These are discrete-variable extensions of the variational autoencoders.
- Can be seen as an exponentially large mixture of distributions.

Neural Variational Inference and Learning in Belief Nets

NVIL (Mnih&Gregor, 2014) learns belief networks via REINFORCE + control variates It optimizes the following learning objective:

$$\mathcal{L}(\mathbf{x};\theta,\phi,\psi,B) = E_{q_\phi(\mathbf{z}|\mathbf{x})}[f(\phi,\theta,\mathbf{z},\mathbf{x}) - h_\psi(\mathbf{x}) - B]$$

where $f(\phi,\theta,\mathbf{z},\mathbf{x}) = \log p(\mathbf{z},\mathbf{x};\theta) - \log q_\phi(\mathbf{z}|\mathbf{x})$ in the ELBO integrand.

- **Control Variate 1:** Constant baseline B
- **Control Variate 2:** Input dependent baseline $h_\psi(\mathbf{x})$
- Both B and ψ are learned via SGD to minimize $(f - h - B)^2$
- Gradient estimates w.r.t. ϕ

$$\begin{aligned} &\nabla_\phi \mathcal{L}(\mathbf{x};\theta,\phi,\psi,B) \\ &= E_{q_\phi(\mathbf{z}|\mathbf{x})}\left[(f(\phi,\theta,\mathbf{z},\mathbf{x}) - h_\psi(\mathbf{x}) - B)\nabla_\phi \log q_\phi(\mathbf{z}|\mathbf{x}) + \nabla_\phi f(\phi,\theta,\mathbf{z},\mathbf{x})\right] \end{aligned}$$

Lecture Outline

1. Discrete Stochastic Optimization
2. REINFORCE Gradient Estimation
   - The REINFORCE Estimator
   - Control Variates
   - Neural Variational Inference & Learning in Belief Networks
3. Relaxed Reparameterization Tricks
   - The Gumbel-Max Trick
   - The Gumbel-Softmax Trick
   - Extensions of Gumbel-Softmax to Combinatorial Objects

Towards reparameterized, continuous relaxations

Consider the discrete optimization problem

$$\max_{\phi} E_{q_\phi(\mathbf{z})}[f(\mathbf{z})]$$

- What if $\mathbf{z}$ is a discrete random variable?
  - Categories
  - Permutations
- Reparameterization trick is not directly applicable
- REINFORCE is a general-purpose solution, but needs careful design of control variates
- **Idea:** Relax $\mathbf{z}$ to a continuous random variable with a reparameterizable distribution

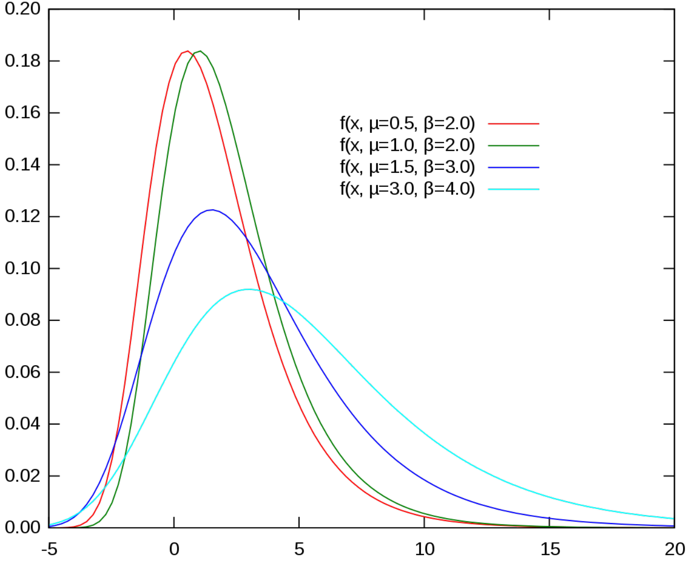
Gumbel Distribution

- The **Gumbel distribution** is very useful for modeling extreme, rare events, e.g., natural disasters, finance
- CDF for a Gumbel random variable g is parameterized by a location parameter μ and a scale parameter β

$$F(g; \mu, \beta) = \exp\left(-\exp\left(-\frac{g-\mu}{\beta}\right)\right)$$

- Note: If g is a Gumbel(μ, β) r.v., $-\log g$ is an Exponential(μ, β) r.v. Often, Gumbel r.v. are referred to as doubly exponential r.v.

Gumbel Distribution



Gumbel-Max Trick

- Let $\mathbf{z}$ denote a k-dimensional categorical random variable with distribution q parameterized by class probabilities $\boldsymbol{\pi} = \{\pi_1, \pi_2, \ldots, \pi_k\}$. We will represent $\mathbf{z}$ as a one-hot vector
- **Gumbel-Max reparameterization trick** for sampling from categorical random variables

$$\mathbf{z} = \text{one\_hot}\left(\arg\max_i (g_i + \log \pi_i)\right)$$

  where $g_1, g_2, \ldots, g_k$ are i.i.d. samples drawn from Gumbel$(0, 1)$
- Reparametrizable since randomness is transferred to a fixed Gumbel$(0, 1)$ distribution!
- Problem: $\arg\max$ is non-differentiable w.r.t. $\boldsymbol{\pi}$

Relaxing Categorical Distributions to Gumbel-Softmax

- Gumbel-Max Sampler (non-differentiable w.r.t. $\boldsymbol{\pi}$):

$$\mathbf{z} = \text{one\_hot}\left(\arg\max_i (g_i + \log \pi_i)\right)$$

- **Key idea:** Replace $\arg\max$ with $\text{soft}\max$ to get a Gumbel-Softmax random variable $\hat{\mathbf{z}}$
- Ouput of softmax is differentiable w.r.t. $\boldsymbol{\pi}$
- Gumbel-Softmax Sampler (differentiable w.r.t. $\boldsymbol{\pi}$):

$$\hat{\mathbf{z}}_i = \text{soft}\max_i \left(\frac{g_i + \log \boldsymbol{\pi}}{\tau}\right) = \frac{\exp(g_i + log \pi_i)/\tau}{\sum_{j=1}^k \exp(g_j + log \pi_j)/\tau}$$

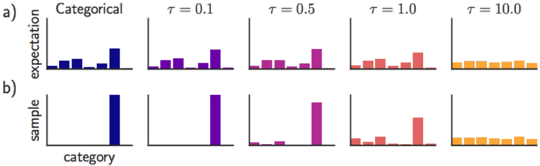
where $\tau > 0$ is a tunable parameter referred to as the temperature

Bias-variance tradeoff via temperature control

- Gumbel-Softmax distribution is parameterized by both class probabilities π and the temperature $\tau > 0$

$$\hat{\mathbf{z}} = \operatorname{soft\,max}_i \left(\frac{g_i + \log \pi_i}{\tau} \right)$$

- Temperature τ controls the degree of the relaxation via a bias-variance tradeoff

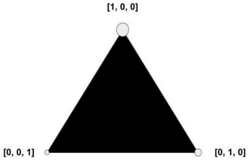


Source: Jang et al., 2017

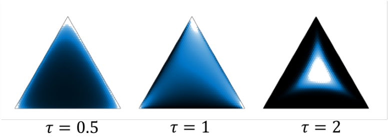
- As $\tau \to 0$, samples from Gumbel-Softmax$(\boldsymbol{\pi}, \tau)$ are similar to samples from Categorical$(\boldsymbol{\pi})$
  Pro: low bias in approximation **Con:** High variance in gradients
- As $\tau \to \infty$, samples from Gumbel-Softmax$(\boldsymbol{\pi}, \tau)$ are similar to samples from Categorical $\left(\left[\frac{1}{k}, \frac{1}{k}, \ldots, \frac{1}{k}\right]\right)$ (i.e., uniform over k categories)

Geometric Interpretation

- Consider a categorical distibution with class probability vector $\boldsymbol{\pi} = [0.60, 0.25, 0.15]$
- Define a probability simplex with the one-hot vectors as vertices



- For a categorical distribution, all probability mass is concentrated at the vertices of the probability simplex
- Gumbel-Softmax samples points within the simplex (lighter color intensity implies higher probability)



Source: Maddison et al., 2018

Gumbel-Softmax in Action

- Original optimization problem

$$\max_{\phi} E_{q_{\phi}(\mathbf{z})}[f(\mathbf{z})]$$

where $q_{\phi}(\mathbf{z})$ is a categorical distribution and $\phi = \boldsymbol{\pi}$

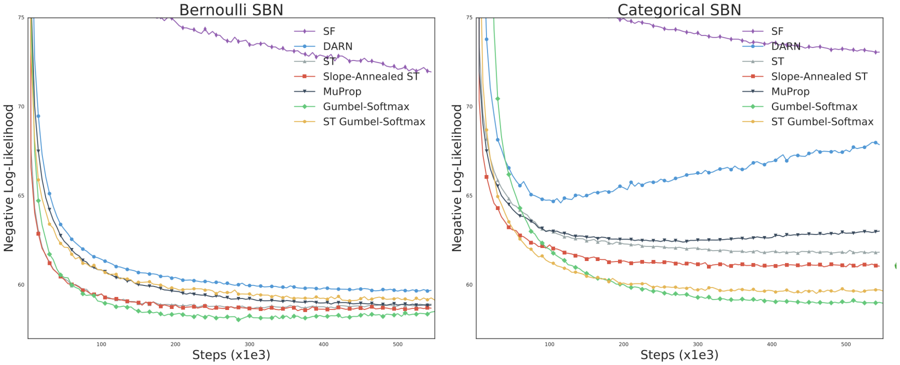
- Relaxed optimization problem

$$\max_{\phi} E_{q_{\phi}(\hat{\mathbf{z}})}[f(\hat{\mathbf{z}})]$$

where $q_{\phi}(\hat{\mathbf{z}})$ is a Gumbel-Softmax distribution and $\phi = \{\boldsymbol{\pi}, \tau\}$

- Usually, temperature τ is explicitly annealed. Start high for low variance gradients and gradually reduce to tighten approximation
- Note that $\hat{\mathbf{z}}$ is not a discrete category. If the function $f(\cdot)$ explicitly requires a discrete $\mathbf{z}$, then we estimate **straight-through gradients**:
  - Use hard $\mathbf{z} \sim \text{Categorical}(\mathbf{z})$ for evaluating objective in forward pass
  - Use soft $\hat{\mathbf{z}} \sim \text{GumbelSoftmax}(\hat{\mathbf{z}}, \tau)$ for evaluating gradients in backward pass

Gumbel-Softmax in action



(a) (b)

Combinatorial, Discrete Objects: Permutations

- For discovering rankings and matchings in an unsupervised manner, $\mathbf{z}$ is represented as a permutation
- A k-dimensional permutation $\mathbf{z}$ is a ranked list of k indices $\{1, 2, \ldots, k\}$
- Stochastic optimization problem

$$\max_{\phi} E_{q_\phi(\mathbf{z})}[f(\mathbf{z})]$$

  where $q_\phi(\mathbf{z})$ is a distribution over k-dimensional permutations
- First attempt: Each permutation can be viewed as a distinct category. Relax categorical distribution to Gumbel-Softmax
- Infeasible because number of possible k-dimensional permutations is $k!$. Gumbel-softmax does not scale for combinatorially large number of categories

Plackett-Luce (PL) Distribution

- In many fields such as information retrieval and social choice theory, we often want to rank our preferences over k items. The **Plackett-Luce (PL) distribution** is a common modeling assumption for such rankings
- A k-dimensional PL distribution is defined over the set of permutations $\mathcal{S}_k$ and parameterized by k positive scores $\mathbf{s}$
- **Sequential sampler** for PL distribution
  - Sample z_1 without replacement with probability proportional to the scores of all k items

$$p(z_1 = i) \propto s_i$$

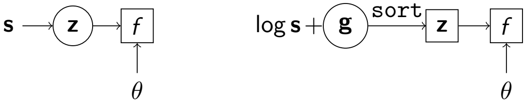
  - Repeat for $z_2, z_3, \ldots, z_k$
- **PDF** for PL distribution

$$q_{\mathbf{s}}(\mathbf{z}) = \frac{s_{z_1}}{Z} \frac{s_{z_2}}{Z - s_{z_1}} \frac{s_{z_3}}{Z - \sum_{i=1}^{2} s_{z_i}} \cdots \frac{s_{z_k}}{Z - \sum_{i=1}^{k-1} s_{z_i}}$$

where $Z = \sum_{i=1}^{k} s_i$ is the normalizing constant

Relaxing PL Distribution to Gumbel-PL

- **Gumbel-PL reparameterized sampler**
  - Add i.i.d. standard Gumbel noise $g_1, g_2, \dots, g_k$ to the log scores $\log s_1, \log s_2, \dots, \log s_k$
  $$\tilde{s}_i = g_i + \log s_i$$
  - Set $\mathbf{z}$ to be the permutation that sorts the Gumbel perturbed log-scores, $\tilde{s}_1, \tilde{s}_2, \dots, \tilde{s}_k$



(a) Sequential Sampler (b) Reparameterized Sampler

Figure: Squares and circles denote deterministic and stochastic nodes.

- **Challenge:** the sorting operation is non-differentiable in the inputs
- **Solution:** Use a differentiable relaxation. See the paper "Stochastic Optimization for Sorting Networks via Continuous Relaxations" for more details

Summary

- Discovering discrete latent structure e.g., categories, rankings, matchings etc. has several applications
- Stochastic Optimization w.r.t. parameterized discrete distributions is challenging
- REINFORCE is the general purpose technique for gradient estimation, but suffers from high variance
- Control variates can help in controlling the variance
- Continuous relaxations to discrete distributions offer a biased, reparameterizable alternative with the trade-off in significantly lower variance