Just Relax It

Discrete variables relaxation

Daniil Dorin, Igor Ignashin, Nikita Kiselev, Andrey Veprikov

Intelligent Systems, MIPT, 2024

Project description

Motivation

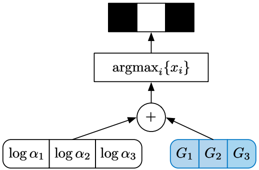
For lots of mathematical problems we need an ability to sample discrete random variables. But the usage of truely discrete random variables is infeasible. Thus we use different relaxation methods.

Used algorithms

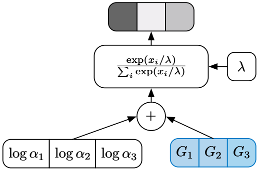
1. Relaxed Bernoulli
2. Correlated relaxed Bernoulli
3. Gumbel-softmax TOP-K
4. Straight-Through Bernoulli
5. Invertible Gaussian
6. Hard concrete
7. REINFORCE
8. Logit-Normal

Solution

`Just Relax It` is a flexible, scalable deep probabilistic programming library built on PyTorch and Pyro.



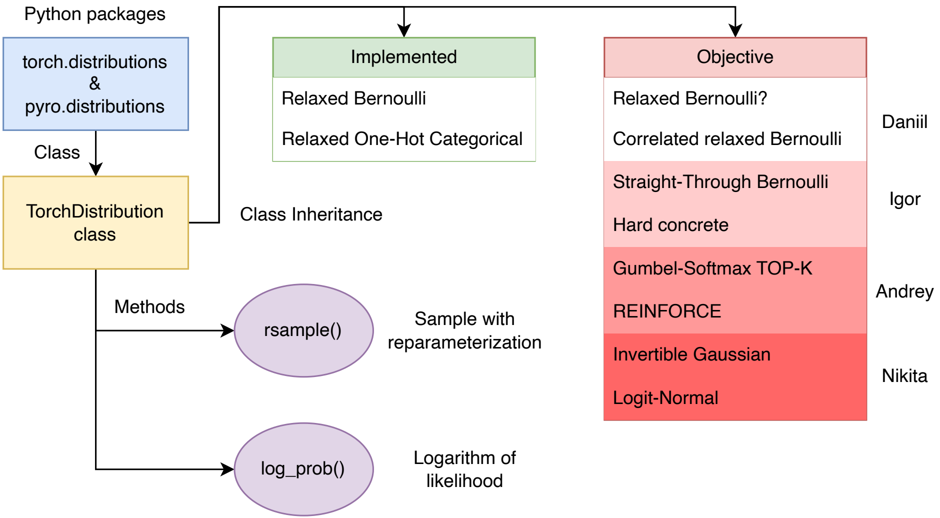
Discrete



Gumbel-Softmax

Scheme of the project

Project scheme

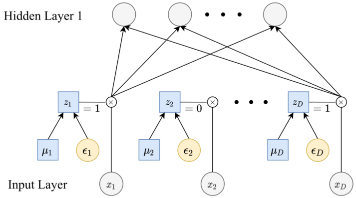


Brief algorithms description

Relaxed Bernoulli, Correlated relaxed Bernoulli (Daniil)

Relaxed Bernoulli

Feature Selection using Stochastic Gates



STG relaxation for a Bernoulli random variable is defined as:

$$z_d = \max(0, \min(1, \mu_d + \epsilon_d)),$$

where $\epsilon_d \sim N(0, \sigma^2)$, and σ is fixed.

Correlated relaxed Bernoulli

The relaxed gate vector $\tilde{m}_k$ is generated using the reparameterization trick:

$$\tilde{m}_k = \sigma\left(\frac{1}{\tau}(\log \pi_k - \log(1 - \pi_k) + \right.$$

$$\left. + \log U_k - \log(1 - U_k))\right),$$

U_k is a uniform random variable, and τ is a temperature hyperparameter.

Straight-Through Bernoulli, Hard concrete (Igor)

Straight-Through Bernoulli

$$u \sim \mathcal{U}[0,1]$$

$$s = \sigma(\frac{log u + log(1-u) + log \alpha}{\beta})$$

$$\bar{s} = s(\zeta - \gamma) + \gamma$$

$$z = \min(1, \max(0, \bar{s}))$$

Hard concrete

$$p_i = \sigma(a_i)$$

$$b_i \sim Binomial(\sqrt{p_i})$$

$$h_i = b_i \sqrt{p_i}$$

$$\mathbb{E} h_i = p_i$$

$$\mathbb{E} f(h_i) = f(p_i) + o(\sqrt{p_i})$$

Invertible Gaussian, Logit-Normal (Nikita)

Invertible Gaussian Reparameterization

Remove interpretability in Gumbel-Softmax

Objective: relax one-hot $\mathbf{z} \sim \mathrm{Cat}(\boldsymbol{\pi})$

GS: $\tau \to 0$ concentrates mass on vertices:
$\tilde{\mathbf{z}} = \mathrm{softmax}(\frac{\log \boldsymbol{\pi} + \mathbf{G}}{\tau})$, $G_i \sim \mathrm{Gumbel}(0, 1)$

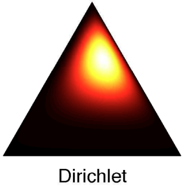
IGR: map $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ to simplex, using invertible $g(\cdot, \tau)$ with temperature τ:

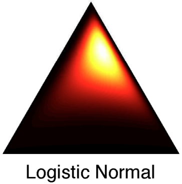
$$\mathbf{y} = \boldsymbol{\mu} + \mathrm{diag}(\boldsymbol{\sigma})\boldsymbol{\epsilon},$$

$$\tilde{\mathbf{z}} = g(\mathbf{y}, \tau) = \mathrm{softmax}_{++}(\mathbf{y}/\tau)$$

Logit-Normal distribution

$\mathrm{softmax}(\mathbf{x})$ of $\mathbf{x} \sim \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$

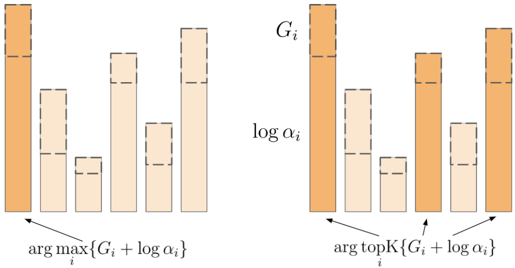




- $\text{Logit-Normal}(\boldsymbol{\mu}, \boldsymbol{\Sigma}) \to \mathrm{Dir}(\boldsymbol{\alpha})$
- $\mathrm{Dir}(\boldsymbol{\alpha}) \to \text{Logit-Normal}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$
- KL divergence?

Gumbel-softmax TOP-K, REINFORCE (Andrey)

Gumbel-softmax TOP-K



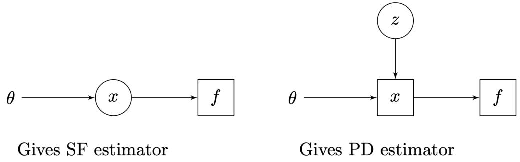
GS: one sample from $\mathrm{Cat}(\boldsymbol{\alpha})$

GS TOP-K: sample K elements without replacement via

$$i_1, \dots, i_K = \arg \operatorname{topK}_i \{G_i + \log \alpha_i\}$$

REINFORCE

- Score function (SF) vs pathwise derivative (PD) estimator of $\frac{\partial}{\partial \theta}\mathbb{E}_x[f(x)]$

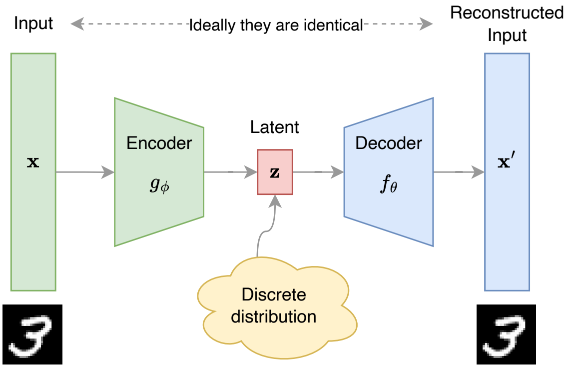


- Monte-Carlo policy gradient estimation of $\nabla_\theta J(\theta) = \mathbb{E}_\pi[G_t \nabla_\theta \ln \pi_\theta(A_t|S_t)]$:
  1. $\pi_\theta \to S_1, A_1, R_2, S_2, A_2, \dots, S_t$
  2. $\theta \leftarrow \theta + \alpha\gamma^t G_t \nabla_\theta \ln \pi_\theta(A_t|S_t)$

Idea for demo/basic code

Demo/basic code

- Consider a VAE architecture
- Classical latent space is Gaussian
- Discrete latent space needs gradient computation
- Implement[a] different relaxation methods for these discrete latents



[a] See our first attempts here