The Implicit Reparameterization Trick in Action: Python Library for Gradients Computation

Matvei Kreinin Iryna Zabarianska Petr Babkin
Maria Nikitina

1 Introduction

The explicit reparameterization trick (ERT) is often used to train various latent variable models due to the ease of calculating gradients of continuous random variables. By making it possible to backpropagate error in computation graphs with certain types of continuous random variables (e.g., Normal and Logistic distributions), ERT serves as a powerful tool for learning. However, due to its peculiarities, ERT is not applicable to several important continuous standard distributions, such as mixture, Gamma, Beta and Dirichlet.

An alternative method for calculating reparameterization gradients relies on implicit differentiation of cumulative distribution functions (CDFs). The implicit reparameterization trick (IRT), being a modification of ERT, is much more expressive and applicable to a wider class of distributions

This article provides an overview of various reparameterization tricks and announces a new Python library, **torch.distributions.implicit**, for sampling from various distributions using the IRT.

2 Explicit reparameterization gradients

If we would like to optimize the expectation $\mathbb{E}_{q_\phi(z)}[f(z)]$ of some continuously differentiable function $f(z)$ w.r.t. the parameters ϕ of the distribution, we are faced with the difficulty of doing this directly. The idea behind the reparameterization trick is to replace a probability distribution with an equivalent parameterization of it using a deterministic and differentiable transformation of some fixed distribution.

We will assume that there exists a continuously differentiable (w.r.t. its argument and parameters) and invertible standardization function $S_\phi(z)$ that, when applied to a sample from the distribution $q_\phi(z)$, eliminates its dependence on the distribution's parameters. This standardization function should be continuously differentiable with respect to both its argument and parameters, and it must be invertible:

$$S_\phi(z) = \varepsilon \sim q(\varepsilon), \quad z = S_\phi^{-1}(\varepsilon).$$

Example 1 *For a Gaussian distribution $\mathcal{N}(\mu, \sigma)$ we can use $S_{\mu,\sigma}(z) = (z - \mu)/\sigma \sim \mathcal{N}(0, 1)$, as a standardization function.*

Under the assumptions above, we can then represent the objective as an expectation w.r.t. ε, shifting the dependence on ϕ into f:

$$\mathbb{E}_{q_\phi(z)}[f(z)] = \mathbb{E}_{q(\varepsilon)}[f(S_\phi^{-1}(\varepsilon))].$$

This enables us to calculate the gradient of the expectation as the expectation of the gradients:

$$\nabla_\phi \mathbb{E}_{q_\phi(z)}[f(z)] = \mathbb{E}_{q(\varepsilon)}[\nabla_\phi f(S_\phi^{-1}(\varepsilon))] = \mathbb{E}_{q(\varepsilon)}[\nabla_z f(S_\phi^{-1}(\varepsilon)) \nabla_\phi S_\phi^{-1}(\varepsilon)].$$

A standardization function $S_\phi(z)$ satisfying the requirements exists for a wide range of continuous distributions. However, inverting the CDF is often complex and computationally intensive, and calculating its derivative poses even greater challenges.

3 Implicit reparametrization gradients

The IRT avoids the need to invert the standardization function. To accomplish this, we perform a change of variables $z = S_\phi^{-1}(\varepsilon)$:

$$\nabla_\phi \mathbb{E}_{q_\phi(z)}[f(z)] = \mathbb{E}_{q_\phi(z)}[\nabla_z f(z) \nabla_\phi z], \quad \nabla_\phi z = \nabla_\phi S_\phi^{-1}(\varepsilon)|_{\varepsilon = S_\phi(z)}.$$

By applying the total gradient ∇^{TG} to the equality $S_\phi(z) = \varepsilon$ and expressing the result in terms of partial gradients using the chain rule, we derive:

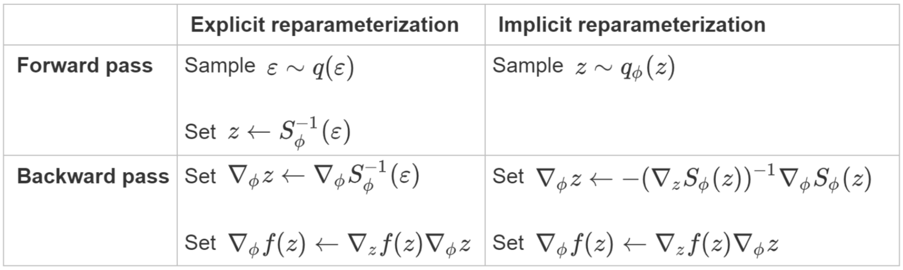
$$\nabla_z S_\phi(z) \nabla_\phi z + \nabla_\phi S_\phi(z) = 0.$$

Now, let's solve the latter equation for $\nabla_\phi z$:

$$\nabla_\phi z = -(\nabla_z S_\phi(z))^{-1} \nabla_\phi S_\phi(z).$$

It is important to note that this expression for the gradient $\nabla_\phi z$, calculated by implicit differentiation, only requires differentiation of the standardization function rather than its inversion.

The following table compares two types of reparameterization: ERT and IRT. Samples of z in the case of IRT can be obtained, for instance, by rejection sampling, and the gradients of the standardization function can be calculated either analytically or using automatic differentiation.



	Explicit reparameterization	Implicit reparameterization
Forward pass	Sample $\varepsilon \sim q(\varepsilon)$ Set $z \leftarrow S_\phi^{-1}(\varepsilon)$	Sample $z \sim q_\phi(z)$
Backward pass	Set $\nabla_\phi z \leftarrow \nabla_\phi S_\phi^{-1}(\varepsilon)$ Set $\nabla_\phi f(z) \leftarrow \nabla_z f(z) \nabla_\phi z$	Set $\nabla_\phi z \leftarrow -(\nabla_z S_\phi(z))^{-1} \nabla_\phi S_\phi(z)$ Set $\nabla_\phi f(z) \leftarrow \nabla_z f(z) \nabla_\phi z$

4 New Python Library: torch.distributions.implicit

We plan to implement the following distributions in **torch.distributions.implicit**:

1. Gaussian normal distribution;
2. Dirichlet distribution (Beta distribution);
3. Sampling from a mixture of distributions;
4. Sampling from the Student's t-distribution;
5. Sampling from an arbitrary factorized distribution.

Our focus will be on providing reliable, efficient, and user-friendly tools that will benefit both researchers and practitioners. We hope that the implementation of the library will contribute to the further development of this field of science. Stay tuned for updates!

Git: https://github.com/intsystems/implicit-reparameterization-trick.git