Gradients for Discrete Latent Variables

Archit Sharma, Devansh Arpit, Sherjil Ozair, Yoshua Bengio

Work done at Montreal Institute of Algorithms (MILA), University of Montreal

Problem Description

- Given a generic directed acyclic computation graph with discrete latent variables, we need gradient estimators for loss function.
- Naive backpropagation (with or without reparametrization) is not possible.

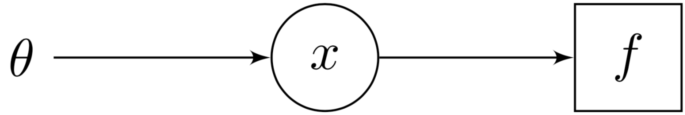
Motivation

Discrete latent variables are very commonly encountered:

- Reinforcement Learning (Policy Gradients!)
- Conditional Computation (Mixture of Experts)
- GANs for structured outputs (especially Natural Language)

A very general and widely research problem. But, even today it is practically an open problem (with potentially wide reaching impact).

A Generic Setup



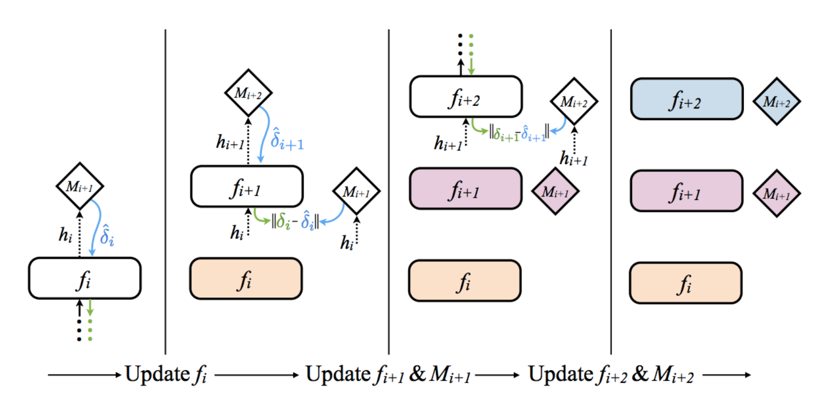
Current Set of Solutions

A fair amount of research has gone into the solution of this problem:

- REINFORCE: The staple gradient estimator, which is the basis of "Policy Gradients" in RL. Usually a very high variance estimator, but, variance can be controlled using "baselines" (NVIL, MuProp etc.)
- Straight Through: A biased estimator, but works well in practice for small scale problems.
- Gumbel-Softmax/Concrete: A recently proposed estimator which uses samples a *reparametrizable continuous distribution* to represent samples from a discrete distribution (in limit). Usually has trouble scaling up as the discrete latent variables become higher dimensional.
- REBAR: A recently proposed unbiased estimator with low variance.

A Detour: Synthetic Gradients

Recently proposed framework to compute gradients, which tries to decouple feedforward layers using “subnetworks”. A classic example of deep learning going meta!



Another Detour: REINFORCE

Classic log-gradient trick

Given latent variable $x \sim p(x|\theta)$, we require:

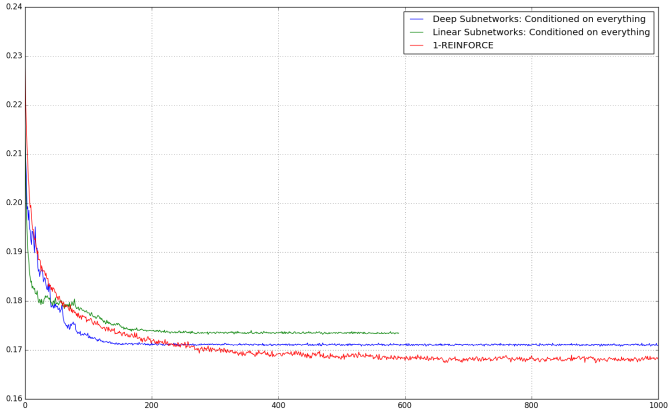
$$\frac{\partial \mathbb{E}_x[f(x)]}{\partial \theta} = \mathbb{E}_x[f(x)\frac{\partial \log p}{\partial \theta}] \approx \frac{1}{L}\sum_{i=1}^{L} f(x_i)\frac{\partial \log p(x_i|\theta)}{\partial \theta}$$

This is unbiased and with large number of samples, is a good aproximation of the "true" gradient. Practically though, one sample is considered in which the high variance kicks in. Although, many solutions using "baselines" are used to reduce variance.

Synthetic Gradients for Expectation Modelling?

- Neural networks have a natural tendency to average, when using the MSE loss.
- Idea: Use the unbiased REINFORCE as the training signal for a subnetwork.
- Hope is that the neural network would cut through the noise, and model the expectation if fed with enough examples.
- Skipping a lot of details!

Results



Deep Subnetworks: Conditioned on everything
Linear Subnetworks: Conditioned on everything
1-REINFORCE
0.24
0.23
0.22
0.21
0.20
0.19
0.18
0.17
0.16
0
200
400
600
800
1000

Questions?