Gradient Estimation Using Stochastic Computation Graphs[*]

Schulman et al.

2015

1 What

Very often we need to estimate the gradient of the loss function defined as an expectation over some random variables. The paper introduces the formalism of *stochastic computation graphs* (SCG) which unifies the prior work on the variety of estimators as well as variance reduction techniques. The work provides us with a modification of the backprop which works for the combination of both deterministic and stochastic nodes.

Contributions:

- formalism + an unbiased estimator of the gradient of the expected loss
- the authors show how the gradient can be computed via computing the gradient of the surrogate loss using auto-diff software
- description of variance reduction techniques for SCG generalising prior work from RL and variational inference
- brief description of generalising some other optimisation techniques to the setting[1]

2 Why

A lot of problems in machine learning comes to likelihood maximisation in probabilistic models with latent variables and policy gradient methods in RL. We need to compute gradients of the expectation over some random variables. We can't use just backprop for that. We have recently seen a few papers featuring the loss functions which can be easily described by the stochastic computation graphs formalism.

[*]Notes by Vitaly Kurin `https://yobibyte.github.io/`

[1]really brief

3 How

Let's assume that x is an r.v. There are different ways of parametrising the process of generating x.

For the case when $x \sim p(\cdot;\theta)$ we have the score function estimator (famous logtrick everywhere):

$$\frac{\partial}{\partial\theta}\mathbb{E}_x\left[f(x)\right] = \mathbb{E}_x\left[f(x)\frac{\partial}{\partial\theta}\log p(x;\theta)\right] \tag{1}$$

The equation is valid iff $p(x;\theta)$ is a continuous function of θ, continuity wrt x is not necessary here [2].

When x is deterministic, differentiable function of θ and a r.v. z, we can use a pathwise derivative estimator:

$$\frac{\partial}{\partial\theta}\mathbb{E}_z\left[f(x(z,\theta))\right] = \mathbb{E}_z\left[f(x(z,\theta))\right]. \tag{2}$$

The equation is valid iff $f(x(z,\theta))$ is a continuous function of θ for all z.

The final combo is when we combine the two approaches and have the following estimator:

$$\frac{\partial}{\partial\theta}\mathbb{E}_{z\sim p(\cdot;\theta)}\left[f(x(z,\theta))\right] = \mathbb{E}_{z\sim p(\cdot;\theta)}\left[\frac{\partial}{\partial\theta}f(x(z,\theta)) + (\frac{\partial}{\partial\theta}\log p(z;\theta))f(x(z,\theta))\right] \tag{3}$$

Properties of SF and PD:

- SF is less restricted in terms of x and f. It works if f is discontinuous or x is discrete r.v.
- SF requires only sample of $f(x)$, PD requires $f'(x)$. E.g. in RL when we don't have access to the dynamics, SF is able to learn only from sampled rollouts.
- SF has more variance. PD is usually preferrable when x is high dimensional. At the same time, PD has high variance when f is rough [3].
- PD allows for the deterministic limit, SF does not.

Okay, here we go, finally, what is SCG?

SCG is a directed, acyclic graph with three types of nodes: inputs (including the parameters we differentiate wrt), deterministic (functions of their parents) and stochastic (distributed conditionally on their parents).

I'll copypaste the notation directly from the paper below:

[2] Is it just simply because we have a derivative wrt θ here and not to x?

[3] What the hell is that?

Cost are scalar-valued deterministic nodes for the sum of which we want to compute the gradients wrt some input node θ.

We need the following requirements to hold in order the gradient exists:

- Given the input node θ and all edges (v, w) s.t. $\theta \prec^D v$ and $\theta \prec^D w$
- if w is deterministic, $\frac{\partial w}{\partial v}$ exists
- if w is stochastic, $\frac{\partial}{\partial v} p(w| \, PARENTS_w)$ exists.

It's interesting that the statements above don't restrict on all the functions to be differentiable. If the path from the input to some node is blocked by a stochastic node, the whole function does not need to be differentiable.

Okay, now if we satisfy these requirements, the following is true:

Notation Glossary

$\mathcal{X}$: Input nodes

$\mathcal{D}$: Deterministic nodes

$\mathcal{S}$: Stochastic nodes

$\mathcal{C}$: Cost nodes

$v \prec w$: v influences w, i.e. there is a path from v to w.

$v \prec^D w$: v deterministically influences w. Same as above, but all the nodes along the path are deterministic.

DEPS_v: "dependencies", $\{w \in \mathcal{X} \cup \mathcal{S} | w \prec^D v\}$

$\hat{Q}_v$: sum of cost nodes influenced by v. $\hat{Q}_v := \sum_{c \succ v, c \in \mathcal{C}} \hat{c}$

$\hat{v}$: denotes the sampled value of the node v.

$$\frac{\partial}{\partial \theta} \mathbb{E} \sum_{c \in \mathcal{C}} c = \mathbb{E} \sum_{\substack{w \in \mathcal{S}, \\ \theta \prec^D w}} \left(\frac{\partial}{\partial \theta} \log p(w | \text{DEPS}_w)\right) \hat{Q}_w + \sum_{\substack{c \in \mathcal{C} \\ \theta \prec^D c}} \frac{\partial}{\partial \theta} c(\text{DEPS}_c) \tag{4}$$

$$= \mathbb{E} \sum_{c \in \mathcal{C}} \hat{c} \sum_{\substack{w \prec c, \\ \theta \prec^D w}} \frac{\partial}{\partial \theta} \log p(w | \text{DEPS}_w) + \sum_{\substack{c \in \mathcal{C}, \\ \theta \prec^D c}} \frac{\partial}{\partial \theta} c(\text{DEPS}_c). \tag{5}$$

The first term of the equations above is here since θ influences probability distributions. The second term is here because θ influences the costs through a chain of differentiable functions. Note, that in the second equation, we sum over all the cost nodes, whereas in the first equation, we sum only the costs downstream of node v. There is also the change in the summation of grads.

So, we have the expression for the gradient of the expectation of the sum of the costs. The authors claim that $\frac{\partial}{\partial \theta} \mathbb{E} \sum_{c \in \mathcal{C}} c = \mathbb{E} \frac{\partial}{\partial \theta} L(\mathcal{X}, \mathcal{S})$ gives us an unbiased estimator of the grad if $L(\mathcal{X}, \mathcal{S}) := \sum_w \log p(w|\text{DEPS}_w) \hat{Q}_w + \sum_{c \in \mathcal{C}} c(\text{DEPS}_c)$. This is the place where magic begins, the result above enables us to do the usual `from tensorflow import *` and calculate the unbiased gradient estimation since the graph is not stochastic anymore.

The authors also write that the grad estimator of a stochastic graph is a stochastic graph and you can do the procedure again. Looks like you cannot simply do that [Foerster et al., 2018].

Finally, let's go to the variance reduction. The main point is that due to the following theorem, you can add a baseline to every stochastic node, which depends on all of the nodes it (the node, as I get it) doesn't influence $\textsc{NonInfluenced}(v) := \{w := v \not\prec w\}$:

$$\frac{\partial}{\partial\theta}\mathbb{E}\sum_{c\in\mathcal{C}} c = \mathbb{E}\,[\sum_{\substack{v\in\mathcal{S}\\ v\succ\theta}}(\frac{\partial}{\partial\theta}\log p(v|\textsc{parents}_v))(\hat{Q}_v - b(\textsc{NonInfluenced}(v)) + \sum_{c\in\mathcal{C}\succeq\theta}\frac{\partial}{\partial\theta}c]$$

I'll not copypaste the grad computation algorithm, you can find it on page 8 of the paper. It's an algorithm which recursively computes the gradients for all the nodes along the path.

4 Evaluation

There is no typical benchmark evaluation in this paper. But who cares about it when you provide such a cool general framework with theoretical analysis and bunch of relevant work? We need more papers like that!

5 Comments

- [Foerster et al., 2018] shows, that estimation of the second order gradients is much more sophisticated than just estimation of the first order. Always stay alert!
- The paper references [Glasserman, 2013] several times. It turns out that research in finance has a lot of in common with RL. Or is it just estimating the gradient of the expectation?

References

[Foerster et al., 2018] Foerster, J., Farquhar, G., Al-Shedivat, M., Rocktäschel, T., Xing, E. P., and Whiteson, S. (2018). Dice: The infinitely differentiable monte-carlo estimator. *arXiv preprint arXiv:1802.05098.*

[Glasserman, 2013] Glasserman, P. (2013). *Monte Carlo methods in financial engineering*, volume 53. Springer Science & Business Media.