Construct and Optimize Low-variance Black-box Gradient Surrogate with Control Variates for Discrete Latent Variable Model

Abstract

Recently, deep reinforcement learning like actor-critic methods has gained fairly good performances in many tasks. Like all other discrete latent variable models, it's challenging to train due to high variance gradient estimators such as policy gradient. Control variates method is a way to produce low-variance, unbiased gradient estimates. In this paper, we use control variates method to combine Monte Carlo policy gradient and path derivative estimator via continuous relaxation to construct a unbiased black-box gradient surrogate, and optimize its variance continuously. Thus we can stabilize the learning progress and may improve results as well. We demonstrate the effects of gradient surrogate on two tasks: VAE on binary-mnist dataset and Seq-GAN for sequence generation, in which the latent variate obeys bernoulli distribution and multinomial distribution respectively.

1 Introduction

Recently, models with discrete latent variables have been applied to many tasks and achieved great performances. Deep reinforce learning, using Markov Decision Processes, has shown its capability to play games like Atari and locomotion tasks [Duan *et al.*, 2016]. Especially, A3C [Mnih *et al.*, 2016], which is an extension of actor-critic method [Sutton *et al.*, 2000], attains the best result. Besides, many methods and technologies in RL are adapted to Natural Language Processing as well. SeqGAN [Yu *et al.*, 2017], aiming for natural text generation, uses Monte Carlo policy gradient(REINFORCE-like) in GAN [Goodfellow *et al.*, 2014] phrase, reaching better results than only maximum likelihood estimation. [Bahdanau *et al.*, 2016] takes actor-critic method for sequence prediction, validating this approach on spelling correction and machine translation.

Generally, REINFORCE-like methods[Williams, 1992] are often adopted for these discrete latent variable model to do sequence prediction, such as SeqGAN, thus gradient-based optimization is essential due to high-variance gradient estimates obtained from sampling. One way to relieve is to jointly optimizing the policy parameters with an estimate of the value function, just like advantage actor-critic methods [Sutton *et al.*, 2000]. Another popular way is to develop clever control variates [Paisley *et al.*, 2012] to reduce variances [Mnih and Rezende, 2016; Tucker *et al.*, 2017; Grathwohl *et al.*, 2017]. By the way, path derivative estimator or reparameterization trick [Kingma and Welling, 2013] can provide low-variance, but biased estimate when it continuously relaxes discrete random variables , like Gumbel-Softmax distribution in [Jang *et al.*, 2016] and Concrete distribution [Maddison *et al.*, 2016], and it's directly through back-propagation algorithm [Rumelhart *et al.*, 1986].

In this paper, for discrete latent variable models, we use control variates method to combine Monte Carlo policy gradient and path derivative estimator via continuous relaxation to construct a unbiased black-box gradient surrogate, and optimize its variance continuously. In detail, the surrogate is composed of three parts: origin policy gradient, new policy gradient of the control variate which is parameterized by a neural network, and new path derivative gradient of the control variate. The parameters of control variate will continuously be optimized toward lower variance of constructed surrogate. Thus we can stabilize the learning progress and may improve results as well. Also, we introduce 2 different ways to do continuous relaxation for discrete variables obeying multinomial distribution, and the progress for conditional marginalization then. We demonstrate the effects of gradient surrogate on two tasks: VAE on binary-mnist dataset and Seq-GAN for sequence generation, in which the latent variate obeys bernoulli distribution and multinomial distribution respectively. At last , we discuss a lot about its inherent mechanism and conditions when it can works, such as replacing learning rate decay manually and comparing or combing with actor-critic method.

2 Background

2.1 Monte Carlo Policy Gradient

For clarity, consider f is a black-box function or neural network, which is parameterized by θ, and we set $b \sim p(\theta)$, then we need to optimize:

$$\mathcal{L}(\theta) = \mathbb{E}_{p(b|\theta)}[\,f(b)]\,. \tag{1}$$

One of the most generally-applicable gradient estimator is known as the score-function estimator, or Monte Carlo Policy Gradient(REINFORCE) [Williams, 1992]:

$$\hat{g}_{\text{REINFORCE}}[f] = f(b)\frac{\partial}{\partial\theta}\log p(b|\theta), \qquad b \sim p(b|\theta) \quad (2)$$

This is an unbiased gradient estimate of $\nabla_\theta f$ by backpropagating along a surrogate loss effectively. But in general, it has high variance with small Monte Carlo samples.

2.2 SeqGAN via Policy Gradient

Sequence Generative Adversarial Nets (SeqGAN) uses a discriminative model to guide the training of the generative model to gain further performance on text generation after period of maximum likelihood estimation(MLE) pre-training. It models the data generator as a stochastic policy, and trains by directly performing gradient policy update where rewards are from discriminator.

The generative model G_θ is a θ-parameterized function to produce a sequence $Y_{1:T} = (y_1, \ldots, y_t, \ldots, y_T), y_t \in \mathcal{V}$, where $\mathcal{V}$ is the vocabulary of all candidate tokens. To interpret this problem in reinforcement learning way: in timestep t, the state s is the current predicted tokens $(y_1, \ldots, y_{t-1})$ and the action a is the next token y_t to select. Thus $G_\theta(y_t|Y_{1:t-1})$ can be seen as a policy function, which is stochastic. And the state transition is deterministic after a particular action has been chosen.

Additionally, a ϕ-parameterized discriminative model D_ϕ is designed to provide a guidance for improving generator G_θ. $D_\phi(Y_{1:T})$ is the reward of a sequence $Y_{1:T}$, indicating its confidence degree to be a real sequence data.

We need to maximize the expectation reward:

$$L(\theta) = \mathbb{E}[R_T|s_0, \theta] = \sum_{y_1 \in \mathcal{V}} G_\theta(y_1|s_0) \cdot Q_{D_\phi}^{G_\theta}(s_0, y_1). \quad (3)$$

where R_T is the reward of a sentence, $Q_{D_\phi}^{G_\theta}(s, a)$ is action-value function, returning the expected accumulative reward starting from state s, taking action a, and then following policy G_θ. Usually, the starting is set a begin-of-word symbol, and the goal of generator is to generate a sequence to make discriminator consider it's from real data.

The discriminator $Q_{D_\phi}^{G_\theta}(s, a)$ is estimated using REINFORCE algorithm:

$$Q_{D_\phi}^{G_\theta}(a = y_t, s = Y_{1:t-1}) = D_\phi(Y_{1:T}). \quad (4)$$

The discriminator D_ϕ as a reward function is dynamically updated to further improve the generator iteratively. Once we have a set of more realistic generated sequences, we shall retrain the discriminator model as follows:

$$\min_\phi -\mathbb{E}_{Y\sim p_{\text{data}}}[\log D_\phi(Y)] - \mathbb{E}_{Y\sim G_\theta}[\log(1 - D_\phi(Y))]. \quad (5)$$

The gradient of the objective function $J(\theta)$ w.r.t. the generator's parameters θ can be derived as:

$$\nabla_\theta L(\theta) = \textstyle\sum_{t=1}^{T} \mathbb{E}_{Y_{1:t-1}\sim G_\theta}\big[\sum_{y_t \in \mathcal{Y}} \nabla_\theta G_\theta(y_t|Y_{1:t-1}) \cdot Q_{D_\phi}^{G_\theta}(Y_{1:t-1}, y_t)\big]. \quad (6)$$

On each episode, we then build an unbiased estimation using policy gradient method:

$$\begin{aligned}
\nabla_\theta L(\theta) &\simeq \sum_{t=1}^{T}\sum_{y_t\in\mathcal{Y}} \nabla_\theta G_\theta(y_t|Y_{1:t-1}) \cdot Q_{D_\phi}^{G_\theta}(Y_{1:t-1}, y_t) \qquad (7)\\
&= \sum_{t=1}^{T}\sum_{y_t\in\mathcal{Y}} G_\theta(y_t|Y_{1:t-1})\nabla_\theta \log G_\theta(y_t|Y_{1:t-1}) \cdot Q_{D_\phi}^{G_\theta}(Y_{1:t-1}, y_t)\\
&= \sum_{t=1}^{T} \mathbb{E}_{y_t\sim G_\theta(y_t|Y_{1:t-1})}[\nabla_\theta \log G_\theta(y_t|Y_{1:t-1}) \cdot Q_{D_\phi}^{G_\theta}(Y_{1:t-1}, y_t)].
\end{aligned}$$

At last, the whole framework is first to pre-train with MLE and then training generator and discriminator iteratively until converging.

3 Construct and Optimize Low-variance Black-box Gradient Surrogate

In this section, we will (1) use control variates method to construct an unbiased, low-variance gradient surrogate for Monte Carlo Policy Gradient. And optimize variance continuously through control variate, which is parameterized by a neural network (2) extend to discrete latent model, demonstrating the progress of derivation. (3) provide 2 alternative ways for continuous relaxations of discrete latent variable, which obeys multinomial distribution. (4) adapt this approach to SeqGAN, providing a whole training framework.

3.1 Use Control Variates Method to Construct and Optimize Gradient Surrogate

First, for distributions that are reparameterizable, we can compute sample z from a deterministic function T with parameters θ. That is path derivative gradient estimator or reparameterization trick:

$$\hat{g}_{\text{reparam}}[f] = \frac{\partial}{\partial\theta} f(b) = \frac{\partial f}{\partial T}\frac{\partial T}{\partial\theta}, \qquad \epsilon \sim p(\epsilon) \quad (8)$$

Then we construct a gradient surrogate with control variates method:

$$\hat{g}_{\text{S}}(b) = \hat{g}(b) - c(b) + \mathbb{E}_{p(b)}[c(b)] \quad (9)$$

where c is any function or neural network. And it's an unbiased estimator:

$$\begin{aligned}
\mathbb{E}_{p(b)}\left[\hat{g}_{\text{S}}(b)\right] &= \mathbb{E}_{p(b)}\left[\hat{g}(b) - c(b) + \mathbb{E}_{p(b)}\left[c(b)\right]\right]\\
&= \mathbb{E}_{p(b)}\left[\hat{g}(b)\right] \qquad (10)
\end{aligned}$$

Finally, a concrete form of an unbiased, low-variance gradient surrogate for Monte Carlo Policy Gradient is given as:

$$\begin{aligned}
\hat{g}_{\text{LAX}} &= g_{\text{REINFORCE}}[f] - g_{\text{REINFORCE}}[c_\phi] + g_{\text{reparam}}[c_\phi]\\
&= [f(b) - c_\phi(b)]\frac{\partial}{\partial\theta}\log p(b|\theta) + \frac{\partial}{\partial\theta}c_\phi(b),\\
b &= T(\theta, \epsilon), \epsilon \sim p(\epsilon). \qquad (11)
\end{aligned}$$

Algorithm 1 RELAX: Unbiased, low-variance black-box gradient surrogate with continuous variance optimization.

```
Require: $f(\cdot)$, $\log p(b|\theta)$, reparameterized samplers $b = H(z)$, $z = S(\epsilon, \theta)$ and $\tilde{z} = S(\epsilon, \theta|b)$, neural network $c_\phi(\cdot)$
  while not converged do
    $\epsilon_i, \widetilde{\epsilon_i} \sim p(\epsilon)$                                   ▷ Sample noise
    $z_i \leftarrow S(\epsilon_i, \theta)$                                  ▷ Compute unconditional relaxed input
    $b_i \leftarrow H(z_i)$                                       ▷ Compute input
    $\widetilde{z_i} \leftarrow S(\widetilde{\epsilon_i}, \theta|b_i)$                              ▷ Compute conditional relaxed input
    $g_\theta \leftarrow [f(b_i) - c_\phi(\widetilde{z_i})] \nabla_\theta \log p + \nabla_\theta c_\phi(z_i) - \nabla_\theta c_\phi(\widetilde{z_i})$     ▷ Estimate gradient
    $g_\phi \leftarrow 2 g_\theta \frac{\partial g_\theta}{\partial \phi}$                                ▷ Estimate gradient of variance of gradient
    $\theta \leftarrow \theta + \alpha_1 g_\theta$                                ▷ Update parameters
    $\phi \leftarrow \phi + \alpha_2 g_\phi$                                ▷ Update control variate
  end while
  return $\theta$
```

Optimize this gradient surrogate by continuously reducing its variances for θ, this is done by gradient backpropogation to control variate c, which is parameterized by parameters ϕ:

$$\frac{\partial}{\partial \phi}\text{Variance}(\hat{g}) = \frac{\partial}{\partial \phi}\mathbb{E}[\hat{g}^2] - \frac{\partial}{\partial \phi}\mathbb{E}[\hat{g}]^2 = \frac{\partial}{\partial \phi}\mathbb{E}[\hat{g}^2]$$
$$= \mathbb{E}\left[\frac{\partial}{\partial \phi}\hat{g}^2\right] = \mathbb{E}\left[2\hat{g}\frac{\partial \hat{g}}{\partial \phi}\right]. \tag{12}$$

3.2 Discrete Latent Variable and Conditional Reparameterization

Following [Maddison *et al.*, 2016], we can reparameterize b with a element-wise hard threshold function $H(z)$:

$$\hat{g}_{\text{DLAX}} = f(b)\frac{\partial}{\partial \theta}\log p(b|\theta) - c_\phi(z)\frac{\partial}{\partial \theta}\log p(z|\theta) + \frac{\partial}{\partial \theta}c_\phi(z),$$
$$b = H(z), z \sim p(z|\theta). \tag{13}$$

The form is different with Eq. (11), but we can incorporate a further refinement using another conditional variable, so as to improve effectiveness. First, review that:

$$\mathop{\mathbb{E}}_{p(b)}\left[f(b)\frac{\partial}{\partial \theta}\log p(b)\right] = \frac{\partial}{\partial \theta}\mathop{\mathbb{E}}_{p(b)}[f(b)]. \tag{14}$$

$$\frac{d}{d\theta}\mathop{\mathbb{E}}_{p(b)}[f(b,\theta)] = \mathop{\mathbb{E}}_{p(b)}\left[\frac{\partial f(b,\theta)}{\partial \theta} + f(b,\theta)\frac{\partial}{\partial \theta}\log p(b)\right]. \tag{15}$$

Then perform a conditional marginalization to gradient of $c(\tilde{z})$, $\tilde{z}$ is a conditional variable to b, θ:

$$\mathop{\mathbb{E}}_{p(z)}\left[c(z)\frac{\partial}{\partial \theta}\log p(z)\right] = \frac{\partial}{\partial \theta}\mathop{\mathbb{E}}_{p(z)}[c(z)]$$
$$= \mathop{\mathbb{E}}_{p(b)}\left[\frac{\partial}{\partial \theta}\mathop{\mathbb{E}}_{p(z|b)}c(z)\right] + \mathop{\mathbb{E}}_{p(b)}\left[\mathop{\mathbb{E}}_{p(z|b)}c(z)\frac{\partial}{\partial \theta}\log p(b)\right]. \tag{16}$$

where the first term can be efficiently estimated with reparameterization trick:

$$\mathop{\mathbb{E}}_{p(b)}\left[\frac{\partial}{\partial \theta}\mathop{\mathbb{E}}_{p(z|b)}c(z)\right] = \mathop{\mathbb{E}}_{p(b)}\left[\mathop{\mathbb{E}}_{p(v)}\frac{\partial}{\partial \theta}c(\tilde{z})\right]. \tag{17}$$

where v is a noise sampling from a distribution, usually $\text{Uniform}(0,1)$ and $\tilde{z}$ can be seen as $\tilde{g}(v,b,\theta)$ that is differentiable for $z|b$. Therefore,

$$\mathop{\mathbb{E}}_{p(z)}\left[c(z)\frac{\partial}{\partial \theta}\log p(z)\right] = \mathop{\mathbb{E}}_{p(b)}\left[\mathop{\mathbb{E}}_{p(v)}\frac{\partial}{\partial \theta}c(\tilde{z})\right]$$
$$+ \mathop{\mathbb{E}}_{p(b)}\left[\mathop{\mathbb{E}}_{p(v)}c(\tilde{z})\frac{\partial}{\partial \theta}\log p(b)\right]. \tag{18}$$

With Eq. (18), we then derive Eq. (13) to arrive at:

$$\hat{g}_{\text{RELAX}} = [f(b) - \eta c_\phi(\tilde{z})]\frac{\partial}{\partial \theta}\log p(b|\theta)$$
$$+ \eta\frac{\partial}{\partial \theta}c_\phi(z) - \eta\frac{\partial}{\partial \theta}c_\phi(\tilde{z}), \tag{19}$$
$$b = H(z), z \sim p(z|\theta), \tilde{z} \sim p(z|b,\theta)$$

where z and $\tilde{z}$ can all be reparameterization, η is an additional variable, either to be constant or learnable.

Thus, the form can be the same as unconditional one Eq. (11) and its effectiveness is improved as well. The whole algorithm is demonstrated in $Algorithm$ 1.

3.3 Continuous Relaxations and Conditional Marginalization

In this section, we talk about taking discrete latent variable to continuous relaxations and then further conditional marginalization. Here, we assume the discrete latent variable $b \sim Multinomial$.

When $p(b|\theta)$ is a Multinomial distribution where $\theta_i = p(b = i|\theta)$, we simply let $H(z) = \text{argmax}(z)$ and we do sampling from $p(z|\theta)$:

$$z = \log\theta - \log(-\log u), u \sim \text{uniform}[0,1]^k \tag{20}$$

where k is the number of possible outcomes.

Then the conditional case: to sample from $p(z|b,\theta)$, we can sample a value v' and compute $\tilde{z} = \log\theta - \log(-\log v')$. We note that in the unconditional case we

Algorithm 2 SeqGAN with Unbiased, Variance-optimized Gradient Surrogate

```
Require: generator policy $G_\theta$; roll-out policy $G_\beta$; discriminator
    $D_\gamma$; a sequence dataset $\mathcal{S} = \{X_{1:T}\}$; control variate $c_\phi$
 1: Initialize $G_\theta$, $D_\gamma$ with random weights $\theta, \gamma$.
 2: Pre-train $G_\theta$ using MLE on $\mathcal{S}$
 3: $\beta \leftarrow \theta$
 4: Generate negative samples using $G_\theta$ for training $D_\gamma$
 5: Pre-train $D_\gamma$ via minimizing the cross entropy
 6: Pre-train $c_\phi$ by Eq. (12)
 7: repeat
 8:     for g-steps do
 9:         Generate a sequence $Y_{1:T} = (y_1, \dots, y_T) \sim G_\theta$
10:         for $t$ in $1 : T$ do
11:             Compute $Q(a = y_t; s = Y_{1:t-1})$ by Eq. (4)
12:         end for
13:         Construct gradient surrogate $\hat{g}$ by Eq. (20)
14:         Update generator parameters via gradient surrogate
15:         Compute gradient to control variate $c_\phi$ to reduce
    Variance($\hat{g}$)
16:         Update control variate parameters by backpropogation
17:     end for
18:     for d-steps do
19:         Use current $G_\theta$ to generate negative examples and com-
    bine with given positive examples $\mathcal{S}$
20:         Train discriminator $D_\gamma$ for $k$ epochs by Eq. (5)
21:     end for
22:     $\beta \leftarrow \theta$
23: until SeqGAN converges
```

would have $v'_b \sim$ uniform$[0,1]$ but in the conditional case $v'_b \sim$ Beta $\left[1 + \frac{1-\theta_b}{\theta_b}, 1\right]$. We first sample v'_b in this way. Then we can sample $v'_{i\neq b}$ by finding the point in $[0,1]$ where $z_b = z_{i\neq b}$ and scaling a uniform random variable v_i to be below that value.

Formally,

$$v'_i = \begin{cases} v'_b & i = b \\ v_i \cdot (v'_b)^{\frac{\theta_i}{\theta_b}} & i \neq b \end{cases}$$

$$v'_b \sim \text{Beta}\left[1 + \frac{1-\theta_b}{\theta_b}, 1\right], \qquad v_{i\neq b} \sim \text{uniform}[0,1]$$

and then $\tilde{z} = \log\theta - \log(-\log v')$ which is our sample from $p(z|b,\theta)$.

In nature, we just need to make others below the one that sampled. So here to be an alternative way: Intuitively, to sample from $p(z|b,\theta)$ we can first sample $v \sim$ uniform$[0,1]^k$, then compute $g_b = \log\theta_b - \log(-\log(v_b))$. Then we must determine how to scale each $v_{i\neq b}$ such that $g_{i\neq b} < g_b$. We can define v' such that

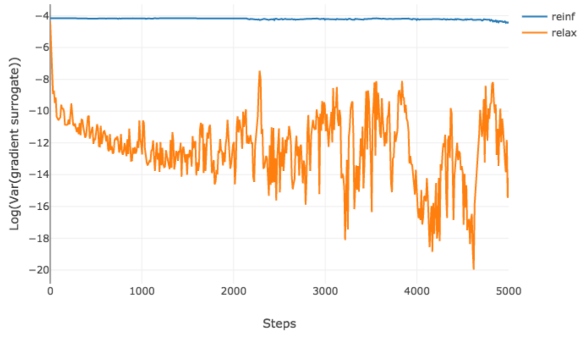
$$v'_i = \begin{cases} v_i & i = b \\ v_i \cdot (v_b)^{\frac{\theta_i}{\theta_b}} & i \neq b \end{cases}$$

Then $\tilde{z} = \log\theta - \log(-\log v')$, just same as the first way introduced before.

3.4 Adapt to SeqGAN: Sequential Multinomial

The gradient surrogate can be applied to SeqGAN via Monte Carlo Policy Gradient, the overall training progress is pointed out to be different as follows: (1) introduce a control variate c_ϕ, a neural network parameterized by ϕ; (2) pre-train c_ϕ as well as MLE G_θ training; (3) construct gradient surrogate to estimate gradient; (4) update c_ϕ towards lower variance of gradient of G_θ.

The whole algorithm for training SeqGAN with GS is demonstrated in $Algorithm$ 2.



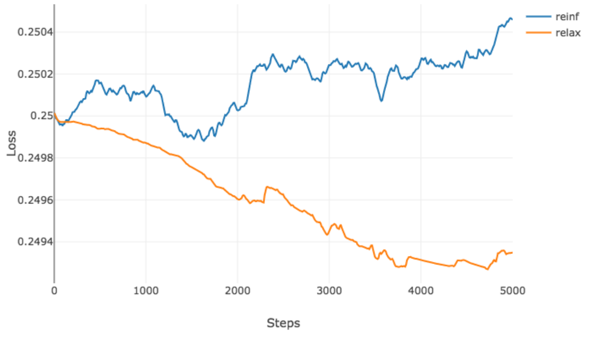


Figure 1: Learning curves of Log Variance and Loss on toy problem

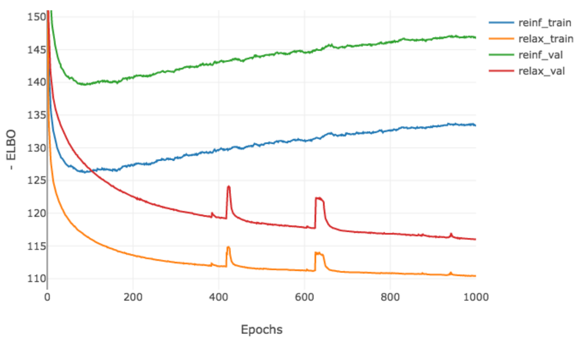
4 Experiments

4.1 Toy problem

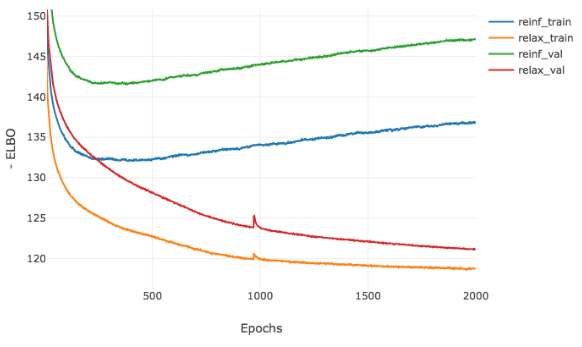
First, to illustrate the effectiveness of gradient surrogate, we do a simple test on a toy problem. Let b is a single parameter controlling the Bernoulli distribution, we want to minimize the loss $\mathbb{E}_{p(b|\theta)}[(b-t)^2]$, where t is a single continuous target value and $t \in (0,1)$, especially we set $t = 0.499$. Fig 1 shows the learning curve of log variance and loss.

4.2 Variational AutoEncoder on Binarized MNIST and Omniglot

Then, we evaluate gradient surrogate on binarized MNIST digits from [Salakhutdinov and Murray, 2008] and a fixed binarization of the Omniglot character dataset. Follow experiments from [Tucker *et al.*, 2017], we use a single-sample one linear-layer or nonlinear-layer encoder-decoder model with Bernoulli latent variables, and control variate model is simple MLE with relu activation. Random seeds are all fixed in both experiments so as to gain low variance.



(a) MNIST



(b) Omniglot

Figure 2: Learning curves of Negative ELBO on MINST and Omniglot

We optimize the variational evidence lower-bound (ELBO) as loss function:

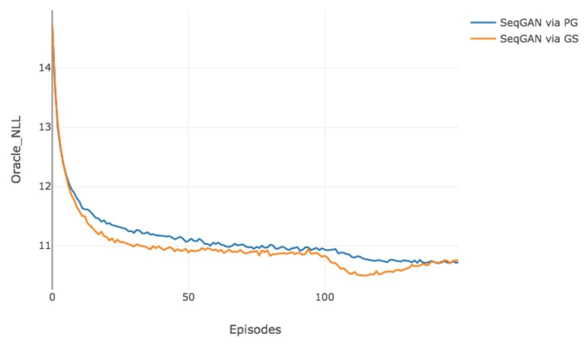
$$\log p(x|\theta) \geq \mathop{\mathbb{E}}_{q(b|x,\theta)} \left[\log p(x,b|\theta) - \log q(b|x,\theta)\right].$$

And Fig 2 shows that gradient surrogate can improve the performance on both datasets. Moreover, convergence speed is increased as well.

4.3 SeqGAN

Then, we adapt to Sequence Generation Task, which is sequential multinomial distribution: following [Yu *et al.*, 2017], we estimate on synthetic dataset. Fig 3 demonstrates the learning curves of NLL on Oracle LSTM, on full progress and adversarial phrase. To do this simply, the generator and discriminator are all GRU [Cho *et al.*, 2014], and c_ϕ is simple RNN as well. Similarly, not only the performance of training is improved, but also the convergence speed is increased. As discriminator changes, the stability may be worse after certain epochs and we will explore on this later.

Fig 5(a) shows that with less pre-training(MLE) epochs, our gradient surrogate can still get better performance while policy gradient has bad performance in this case. We think that it seems to be same effect like leaky information([Guo *et al.*, 2017]).



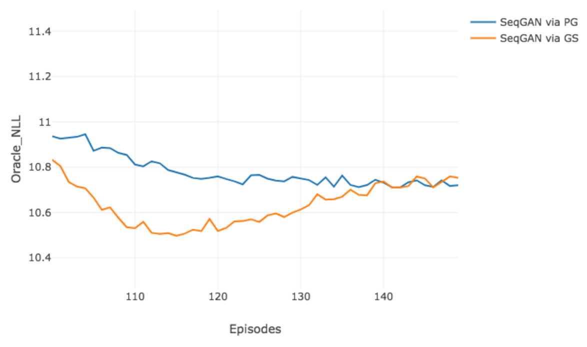


Figure 3: Learning curves of Oracle_NLL of SeqGAN without Roll-out (top is whole progress and bottom is adversarial training part)

And we extend with Roll-out technology which is an essential part for SeqGAN to get more useful and stable rewards in original paper. Fig 5(b) shows that RO can also make GS more stable, with better performance as well. We find that this experiment is sensitive to tune: with discriminator always changing and unstability of GAN itself.

4.4 Gradient Visualization

In addition, as shown in Fig. 4(a), we make an visualization with 5k samples on mnist task to show how Eq. (20) works: make unbiased, low-variance gradient surrogate with biased, high-variance (conditional) control variate and its biased, low-variance reparameterization.

5 Related works

REINFORCE is the most popular score function estimator, and there are its variants: NVIL([Mnih and Gregor, 2014]), MuProp([Gu *et al.*, 2015]), and DARN([Gregor *et al.*, 2013]). Also, path derivative estimator is an alternative way such as Gumbel-Softmax, Straight-Through([Bengio, 2013]) and Concrete. Combining these two methods, we can use control variate method to construct unbiased, low-variance gradient surrogate.

SeqGAN is made for sequential decision making, and actor-critic can be applied too([Bahdanau *et al.*, 2016]).

The most related works are Gumbel Softmax, which is biased, categorical distribution, REBAR, which makes a special

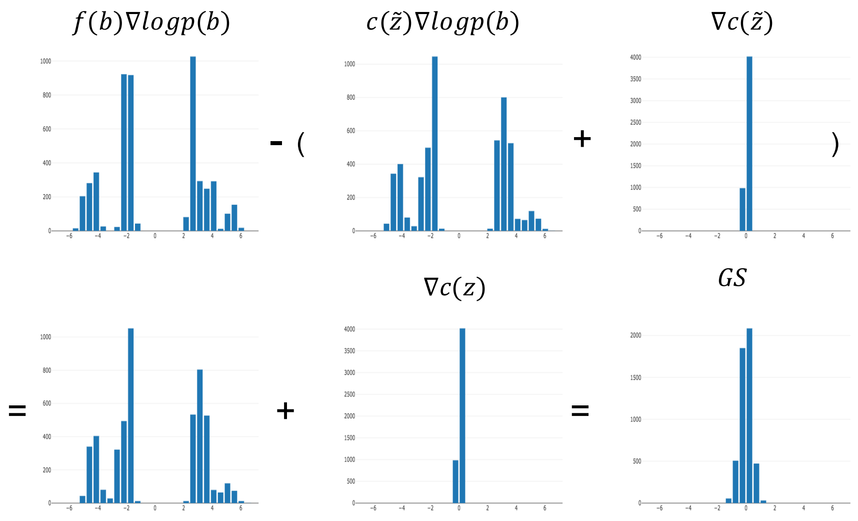
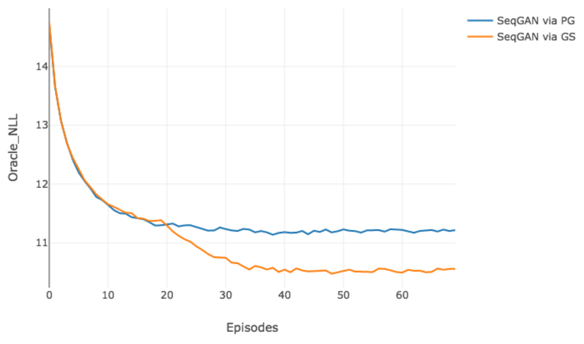
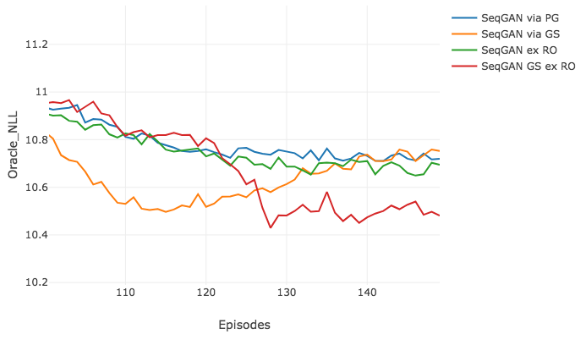


Figure 4: Gradient Visualization: construct unbiased low-variance gradient surrogate with unbiased, high-variance (conditional) control variate and biased low-variance reparameterization



(a) Pre-training MLE=20



(b) Extension with Roll-out

Figure 5: training curves with different training strategies: Less pre-training MLE epochs and Extension of Roll-out

form for unbiased, bernoulli distribution, RELAX, which has general form but applied on bernoulli distribution. Different with these, we adapt to SeqGAN: a sequential multinomial distribution and estimate and discuss gradient surrogate effect on this application which is more typical in practice. And we make reflection and discussion about gradient surrogate and for future work as follows.

6 Discussion

Besides making black-box function gradient estimate stable, there seems to be more potential on gradient surrogate. Firstly, it acts as replacing decaying learning rate manually by an automatic way, so as to do finetuning under som circumstances.

In addition, we may need to think deeply about the power of the reparametrization trick. Here are two cases: when the target function is correct and exact, such as combining with actor-critic: control variate may provide leaky information or modify exploration space.

The other case is that when target function is not exact, the whole gradient surrogate can be seen as a doubly robust estimation. Thus the function can be corrected by the difference among these (weighted) estimates.

7 Conclusion

In this paper, we use control variate method to construct an unbiased, low-variance gradient surrogate for discrete latent model, with its variance decreased continuously as well. And we provide ways of continuous relaxations and conditional marginalization on multinomial distribution. We apply this on two tasks: VAE on binary-mnist dataset and SeqGAN for sequence generation, in which the latent variate obeys bernoulli distribution and multinomial distribution respectively. Also, various training strategies on SeqGAN are investigated and discussed. In addition, we make reflection and discussion about gradient surrogate and for future work.

References

[Bahdanau *et al.*, 2016] Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. An actor-critic algorithm for sequence prediction. *arXiv preprint arXiv:1607.07086*, 2016.

[Bengio, 2013] Yoshua Bengio. Estimating or propagating gradients through stochastic neurons. *CoRR*, abs/1305.2982, 2013.

[Cho *et al.*, 2014] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[Duan *et al.*, 2016] Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning*, pages 1329–1338, 2016.

[Goodfellow *et al.*, 2014] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[Grathwohl *et al.*, 2017] Will Grathwohl, Dami Choi, Yuhuai Wu, Geoff Roeder, and David Duvenaud. Backpropagation through the void: Optimizing control variates for black-box gradient estimation. *arXiv preprint arXiv:1711.00123*, 2017.

[Gregor *et al.*, 2013] Karol Gregor, Andriy Mnih, and Daan Wierstra. Deep autoregressive networks. *CoRR*, abs/1310.8499, 2013.

[Gu *et al.*, 2015] Shixiang Gu, Sergey Levine, Ilya Sutskever, and Andriy Mnih. Muprop: Unbiased backpropagation for stochastic neural networks. *arXiv preprint arXiv:1511.05176*, 2015.

[Guo *et al.*, 2017] Jiaxian Guo, Sidi Lu, Han Cai, Weinan Zhang, Yong Yu, and Jun Wang. Long text generation via adversarial training with leaked information. *arXiv preprint arXiv:1709.08624*, 2017.

[Jang *et al.*, 2016] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[Kingma and Welling, 2013] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[Maddison *et al.*, 2016] Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016.

[Mnih and Gregor, 2014] Andriy Mnih and Karol Gregor. Neural variational inference and learning in belief networks. *arXiv preprint arXiv:1402.0030*, 2014.

[Mnih and Rezende, 2016] Andriy Mnih and Danilo Rezende. Variational inference for monte carlo objectives. In *International Conference on Machine Learning*, pages 2188–2196, 2016.

[Mnih *et al.*, 2016] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International Conference on Machine Learning*, pages 1928–1937, 2016.

[Paisley *et al.*, 2012] John Paisley, David Blei, and Michael Jordan. Variational bayesian inference with stochastic search. *Neuroimage*, 19(3):727–741, 2012.

[Rumelhart *et al.*, 1986] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *nature*, 323(6088):533, 1986.

[Salakhutdinov and Murray, 2008] Ruslan Salakhutdinov and Iain Murray. On the quantitative analysis of deep belief networks. In *Proceedings of the 25th international conference on Machine learning*, pages 872–879. ACM, 2008.

[Sutton *et al.*, 2000] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

[Tucker *et al.*, 2017] George Tucker, Andriy Mnih, Chris J Maddison, Dieterich Lawson, and Jascha Sohl-Dickstein. Rebar: Low-variance, unbiased gradient estimates for discrete latent variable models. 2017.

[Williams, 1992] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. In *Reinforcement Learning*, pages 5–32. Springer, 1992.

[Yu *et al.*, 2017] Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. Seqgan: Sequence generative adversarial nets with policy gradient. In *AAAI*, pages 2852–2858, 2017.

A Appendix

[?; ?]