Bias-Variance Tradeoffs in Single-Sample Binary Gradient Estimators

Alexander Shekhovtsov(✉)

Czech Technical University in Prague, Prague, Czech Republic
shekhole@fel.cvut.cz

Abstract. Discrete and especially binary random variables occur in many machine learning models, notably in variational autoencoders with binary latent states and in stochastic binary networks. When learning such models, a key tool is an estimator of the gradient of the expected loss with respect to the probabilities of binary variables. The straight-through (ST) estimator gained popularity due to its simplicity and efficiency, in particular in deep networks where unbiased estimators are impractical. Several techniques were proposed to improve over ST while keeping the same low computational complexity: Gumbel-Softmax, ST-Gumbel-Softmax, BayesBiNN, FouST. We conduct a theoretical analysis of bias and variance of these methods in order to understand tradeoffs and verify the originally claimed properties. The presented theoretical results allow for better understanding of these methods and in some cases reveal serious issues.

1 Introduction

Binary variables occur in many models of interest. Variational autoencoders (VAE) with binary latent states are used to learn generative models with compressed representations [10,11,22,33] and to learn binary hash codes for text and image retrieval [6,7,20,30]. Neural networks with binary activations and weights are extremely computationally efficient and attractive for embedded applications, in particular pushed forward in the vision research [1–5,8,12,15,17,25,31,34,36]. Training these discrete models is possible via the stochastic relaxation, equivalent to training a Stochastic Binary Networks (SBN) [23,24,26,27,29]. In this relaxation, each binary weight is replaced with a Bernoulli random variable and each binary activation is replaced with a conditional Bernoulli variable. The gradient of the expected loss in the weight probabilities is well defined and SGD optimization can be applied.

The author gratefully acknowledges support by Czech OP VVV project "Research Center for Informatics (CZ.02.1.01/0.0/0.0/16019/0000765)".

Supplementary Information The online version contains supplementary material available at https://doi.org/10.1007/978-3-030-92659-5_8.

C. Bauckhage et al. (Eds.): DAGM GCPR 2021, LNCS 13024, pp. 127–141, 2021.
https://doi.org/10.1007/978-3-030-92659-5_8

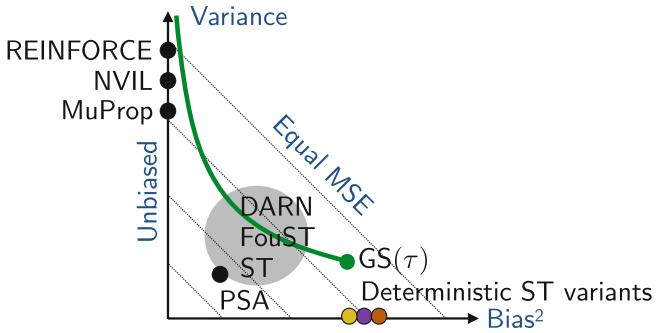

Fig. 1. Schematic illustration of bias-variance tradeoffs (we do not pretend on exactness, but see experimental evaluations in [28,29]; notice that the Mean Squared Error (MSE) is the sum of variance and squared bias). Unbiased methods have a prohibitively high variance for deep models. PSA achieves a significant reduction in variance at a price of a small bias, but has a limited applicability. According to [29], ST estimator can be as accurate as PSA in wide deep models. We analytically study methods in the gray area: GS, DARN and FouST in order to find out whether they can offer a sound improvement over ST. In particular, for GS estimator the figure illustrates its possible tradeoffs when varying the temperature parameter according to the asymptotes we prove.

For the problem of estimating gradient of expectation in probabilities of (conditional) Bernoulli variables, several unbiased estimators were proposed [9,11,19,32,35]. However, in the context of deep SBNs these methods become impractical: MuProp [11] and REINFORCE with baselines [19] have a prohibitively high variance in deep layers [28, Figs. C6, C7] while other methods' complexity grows quadratically with the number of Bernoulli layers. In these cases, biased estimators were more successful in practice: straight-through (ST) [28], Gumbel-Softmax (GS) [13,16] and their variants. In order to approximate the gradient of the expectation these methods use a single sample of all random entities and the derivative of the objective function extended to the real-valued domain. A more accurate PSA method was presented in [29], which has low computation complexity, but applies only to SBNs of classical structure[1] and requires specialized convolutions. Notably, it was experimentally reported [29, Fig. 4] that the baseline ST performs nearly identically to PSA in moderate size SBNs. Figure 1 schematically illustrates the bias-variance tradeoff with different approaches.

Contribution. In this work we analyze theoretical properties of several recent single-sample gradient based methods: GS, ST-GS [13], BayesBiNN [18] and FouST [22]. We focus on clarifying these techniques, studying their limitations and identifying incorrect and over-claimed results. We give a detailed analysis of bias and variance of GS and ST-GS estimators. Next we analyze the application

[1] Feed-forward, with no residual connections and only linear layers between Bernoulli activations.

of GS in BayesBiNN. We show that a correct implementation would result in an extremely high variance. However due to a hidden issue, the estimator in effect reduces to a deterministic straight-through (with zero variance). A long-range effect of this swap is that BayesBiNN fails to solve the variational Bayesian learning problem as claimed. FouST [22] proposed several techniques for *lowering* bias and variance of the baseline ST estimator. We show that the baseline ST estimator was applied incorrectly and that some of the proposed improvements may increase bias and or variance.

We believe these results are valuable for researchers interested in applying these methods, working on improved gradient estimators or developing Bayesian learning methods. Incorrect results with hidden issues in the area could mislead many researchers and slow down development of new methods.

Outline. The paper is organized as follows. In Sect. 2 we briefly review the baseline ST estimator. In the subsequent sections we analyze Gumbel-Softmax estimator (Sect. 3), BayesBiNN (Sect. 4) and FouST estimator (Sect. 5). Proofs are provided in the respective Appendices A to C. As most of our results are theoretical, simplifying derivation or identifying limitations and misspecifications of the preceding work, we do not propose extensive experiments. Instead, we refer to the literature for the experimental evidence that already exists and only conduct specific experimental tests as necessary. In Sect. 6 we summarize our findings and discuss how they can facilitate future research.

2 Background

We define a stochastic binary unit $x \sim \text{Bernoulli}(p)$ as $x = 1$ with probability p and $x = 0$ with probability $1 - p$. Let $f(x)$ be a loss function, which in general may depend on other parameters and may be stochastic aside from the dependence on x. This is particularly the case when f is a function of multiple binary stochastic variables and we study its dependence on one of them explicitly. The goal of binary gradient estimators is to estimate

$$g = \frac{\mathrm{d}}{\mathrm{d}p}\mathbb{E}[f(x)], \tag{1}$$

where $\mathbb{E}$ is the total expectation. Gradient estimators which we consider make a stochastic estimate of the total expectation by taking a single joint sample. We will study their properties with respect to x only given the rest of the sample fixed. In particular, we will confine the notion of bias and variance to the conditional expectation $\mathbb{E}_x$ and the conditional variance $\mathbb{V}_x$. We will assume that the function $f(x)$ is defined on the interval $[0, 1]$ and is differentiable on this interval. This is typically the case when f is defined as a composition of simple functions, such as in neural networks. While for discrete inputs x, the continuous definition of f is irrelevant, it will be utilized by approximations exploiting its derivatives.

The expectation $\mathbb{E}_x[f(x)]$ can be written as

$$(1 - p)f(0) + pf(1), \tag{2}$$

Its gradient in p is respectively

$$g = \frac{\mathrm{d}}{\mathrm{d}p}\mathbb{E}_x[f(x)] = f(1) - f(0). \tag{3}$$

While this is simple for one random variable x, it requires evaluating f at two points. With n binary units in the network, in order to estimate all gradients stochastically, we would need to evaluate the loss $2n$ times, which is prohibitive.

Of high practical interest are stochastic estimators that evaluate f only at a single joint sample (perform a single forward pass). Arguably, the most simple such estimator is the straight-through (ST) estimator:

$$\hat{g}_{\mathrm{ST}} = f'(x). \tag{4}$$

For an in-depth introduction and more detained study of its properties we refer to [28]. The mean and variance of this ST estimator are given by

$$\mathbb{E}_x[\hat{g}_{\mathrm{ST}}] = (1-p)f'(0) + pf'(1), \tag{5a}$$
$$\mathbb{V}_x[\hat{g}_{\mathrm{ST}}] = \mathbb{E}_x[\hat{g}^2_{\mathrm{ST}}] - (\mathbb{E}_x[\hat{g}_{\mathrm{ST}}])^2 = p(1-p)(f'(1) - f'(0))^2. \tag{5b}$$

If $f(x)$ is linear in x, *i.e.*, $f(x) = hx + c$, where h and c may depend on other variables, then $f'(0) = f'(1) = h$ and $f(1) - f(0) = h$. In this linear case we obtain

$$\mathbb{E}_x[\hat{g}_{\mathrm{ST}}] = h, \tag{6a}$$
$$\mathbb{V}_x[\hat{g}_{\mathrm{ST}}] = 0. \tag{6b}$$

From the first expression we see that the estimator is unbiased and from the second one we see that its variance (due to x) is zero. It is therefore a reasonable baseline: if f is close to linear, we may expect the estimator to behave well. Indeed, there is a theoretical and experimental evidence [28] that in typical neural networks the more units are used per layer, the closer we are to the linear regime (at least initially) and the better the utility of the estimate for optimization. Furthermore, [29] show that in SBNs of moderate size, the accuracy of ST estimator is on par with a more accurate PSA estimator.

We will study alternative single-sample approaches and improvements proposed to the basic ST. In order to analyze BayesBiNN and FouST we will switch to the ± 1 encoding. We will write $y \sim \mathrm{Bin}(p)$ to denote a random variable with values $\{-1, 1\}$ parametrized by $p = \mathbb{P}_y(y{=}1)$. Alternatively, we will parametrize the same distribution using the expectation $\mu = 2p - 1$ and denote this distribution as $\mathrm{Bin}(\mu)$ (the naming convention and the context should make it unambiguous). Note that the mean of Bernoulli(p) is p. The ST estimator of the gradient in the mean parameter μ in both $\{0, 1\}$ and $\{-1, 1\}$ valued cases is conveniently given by the same equation (4).

Proof. Indeed, $\mathbb{E}_y[f(y)]$ with $y \sim \mathrm{Bin}(\mu)$ can be equivalently expressed as $\mathbb{E}_x[\tilde{f}(x)]$ with $x \sim \mathrm{Bernoulli}(p)$, where $p = \frac{\mu+1}{2}$ and $\tilde{f}(x) = f(2x-1)$. The

ST estimator of gradient in the Bernoulli probability p for a sample x can then be written as

$$\hat{g}_{\text{ST}} = \tilde{f}'(x) = 2f'(y), \tag{7}$$

where $y = 2x - 1$ is a sample from $\text{Bin}(\mu)$. The gradient estimate in μ becomes $2f'(y)\frac{\partial p}{\partial \mu} = f'(y)$. □

3 Gumbel Softmax and ST Gumbel-Softmax

Gumbel Softmax [13] and Concrete relaxation [16] enable differentiability through discrete variables by relaxing them to real-valued variables that follow a distribution closely approximating the original discrete distribution. The two works [13,16] have contemporaneously introduced the same relaxation, but the name Gumbel Softmax (GS) became more popular in the literature.

A categorical discrete random variable x with K category probabilities π_k can be sampled as

$$x = \arg\max_k(\log \pi_k - \Gamma_k), \tag{8}$$

where Γ_k are independent Gumbel noises. This is known as Gumbel reparametrization. In the binary case with categories $k \in \{1, 0\}$ we can express it as

$$x = [\![\log \pi_1 - \Gamma_1 \geq \log \pi_0 - \Gamma_0]\!], \tag{9}$$

where $[\![\cdot]\!]$ is the Iverson bracket. More compactly, denoting $p = \pi_1$,

$$x = [\![\log \frac{p}{1-p} - (\Gamma_1 - \Gamma_0) \geq 0]\!]. \tag{10}$$

The difference of two Gumbel variables $z = \Gamma_1 - \Gamma_0$ follows the logistic distribution. Its cdf is $\sigma(z) = \frac{1}{1+e^{-z}}$. Denoting $\eta = \text{logit}(p)$, we obtain the well-known noisy step function representation:

$$x = [\![\eta - z \geq 0]\!]. \tag{11}$$

This reparametrization of binary variables is exact but does not yet allow for differentiation of a single sample because we cannot take the derivative under the expectation of this function in (1). The relaxation [13,16] replaces the threshold function by a continuously differentiable approximation $\sigma_\tau(\eta) := \sigma(\eta/\tau) = \frac{1}{1+e^{-\eta/\tau}}$. As the temperature parameter $\tau > 0$ decreases towards 0, the function $\sigma_\tau(\eta)$ approaches the step function. The GS estimator of the derivative in η is then defined as the total derivative of f at a random relaxed sample:

$$z \sim \text{Logistic}, \tag{12a}$$

$$\tilde{x} = \sigma_\tau(\eta - z), \tag{12b}$$

$$\frac{\hat{d}f}{d\eta} := \frac{\mathrm{d}f(\tilde{x})}{\mathrm{d}\eta} = f'(\tilde{x})\frac{\partial \tilde{x}}{\partial \eta}. \tag{12c}$$

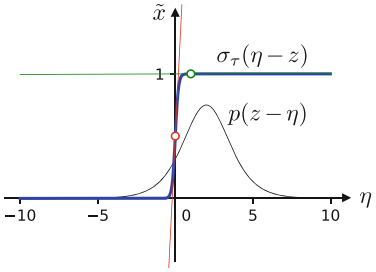

Fig. 2. GS Estimator: relaxed samples $\tilde{x}$ are obtained and differentiated as follows. Noisy inputs, following a shifted logistic distribution (black density), are passed through a smoothed step function σ_τ (blue). Observe that for a small τ, the derivative is often (in probability of $\eta - z$) close to zero (green) and, very rarely, when $|\eta - z|$ is small, it becomes $O(1/\tau)$ large (red). (Color figure online)

A possible delusion about GS gradient estimator is that it can be made arbitrary accurate by using a sufficiently small temperature τ. This is however not so simple and we will clarify theoretical reasons for why it is so. An intuitive explanation is proposed in Fig. 2. Formally, we show the following properties.

Proposition 1. *GS estimator is asymptotically unbiased as $\tau \to 0$ and the bias decreases at the rate $O(\tau)$ in general and at the rate $O(\tau^2)$ for linear functions.*

Proof in Appendix A. The decrease of the bias with $\tau \to 0$ is a desirable property, but this advantage is practically nullified by the fast increase of the variance:

Proposition 2. *The variance of GS estimator grows at the rate $O(\frac{1}{\tau})$.*

Proof in Appendix A. This fast growth of the variance prohibits the use of small temperatures in practice. In more detail the behavior of the gradient estimator is described by the following two propositions.

Proposition 3. *For any given realization $z \neq \eta$ the norm of GS estimator asymptotically vanishes at the exponential rate $O(\frac{1}{\tau}c^{1/\tau})$ with $c = e^{-|x|} < 1$.*

Proof in Appendix A. For small x, where c is close to one, the term $1/\tau$ dominates at first. In particular for $z = \eta$, the asymptote is $O(1/\tau)$. So while for the most of noise realizations the gradient magnitude vanishes exponentially quickly, it is compensated by a significant grows at rate $1/\tau$ around $z = \eta$. In practice it means that most of the time a value of gradient close to zero is measured and occasionally, very rarely, a value of $O(1/\tau)$ is obtained.

Proposition 4. *The probability to observe GS gradient of norm at least ε is asymptotically $O(\tau \log(\frac{1}{\varepsilon}))$, where the asymptote is $\tau \to 0$, $\varepsilon \to 0$.*

Proof in Appendix A.

Unlike ST, GS estimator with $\tau > 0$ is biased even for linear objectives. Even for a single neuron and a linear objective it has a non-zero variance. Propositions

3 and 4 apply also to the case of a layer with multiple units since they just analyze the factor $\frac{\partial}{\partial \eta}\sigma_\tau(\eta - z)$, which is present independently at all units. Proposition 4 can be extended to deep networks with L layers of Bernoulli variables, in which case the chain derivative will encounter L such factors and we obtain that the probability to observe a gradient with norm at least ε will vanish at the rate $O(\tau^L)$.

These facts should convince the reader of the following: it is not possible to use a very small τ, not even with an annealing schedule starting from $\tau = 1$. For a very small τ the most likely consequence would be to never encounter a numerically non-zero gradient during the whole training. For moderately small τ the variance would be prohibitively high. Indeed, Jang et al. [13] anneal τ only down to 0.5 in their experiments.

A major issue with this and other relaxation techniques (*i.e.* techniques using relaxed samples $\tilde{x} \in \mathbb{R}$) is that the relaxation biases all the expectations. There is only one forward pass and hence the relaxed samples $\tilde{x}$ are used for all purposes, not only for the purpose of estimating the gradient with respect to the given neuron. It biases all expectations for all other units in the same layer as well as in preceding and subsequent layers (in SBN). Let for example f depend on additional parameters θ in a differentiable way. More concretely, θ could be parameters of the decoder in VAE. With a Bernoulli sample x, an unbiased estimate of gradient in θ can be obtained simply as $\frac{\partial}{\partial \theta} f(x; \theta)$. However, if we replace the sample with a relaxed sample $\tilde{x}$, the estimate $\frac{\partial}{\partial \theta} f(\tilde{x}; \theta)$ becomes biased because the distribution of $\tilde{x}$ only approximates the distribution of x. If y were other binary variables relaxed in a similar way, the gradient estimate for x will become more biased because $E_{\tilde{y}}[\nabla_{\tilde{x}} f(\tilde{x}, \tilde{y})]$ is a biased estimate of $E_y[\nabla_{\tilde{x}} f(\tilde{x}, y)]$ desired. Similarly, in a deep SBN, the relaxation applied in one layer of the model additionally biases all expectations for units in layers below and above. In practice the accuracy for VAEs is relatively good [13], [28, Fig. 3] while for deep SBNs a bias higher than of ST is observed for $\tau = 1$ in a synthetic model with 2 or more layers and with $\tau = 0.1$ for a model with 7 (or more) layers [29, Fig. C.6]. When training moderate size SBNs on real data, it performs worse than ST [29, Fig. 4].

ST Gumbel-Softmax. Addressing the issue that relaxed variables deviate from binary samples on the forward pass, Jang et al. [13] proposed the following empirical modification. *ST Gumbel-Softmax* estimator [13] keeps the relaxed sample for the gradient but uses the correct Bernoulli sample on the forward pass:

$$z \sim \text{Logistic}, \tag{13a}$$

$$\tilde{x} = \sigma_\tau(\eta - z), \tag{13b}$$

$$x = [\![\eta - z \geq 0]\!], \tag{13c}$$

$$\hat{g}_{\text{ST-GS}(\tau)} = f'(x)\frac{\partial \tilde{x}}{\partial \eta}. \tag{13d}$$

Note that x is now distributed as Bernoulli(p) with $p = \sigma(\eta)$ so the forward pass is fixed. We show the following asymptotic properties.

Proposition 5. *ST Gumbel-Softmax estimator* [13] *is asymptotically unbiased for quadratic functions and the variance grows as* $O(1/\tau)$ *for* $\tau \to 0$.

Proof in Appendix A.

To summarize, ST-GS is more expensive than ST as it involves sampling from logistic distribution (and keeping samples), it is biased for $\tau > 0$. It becomes unbiased for quadratic functions as $\tau \to 0$, which would be an improvement over ST, but the variance grows as $\frac{1}{\tau}$.

4 BayesBiNN

Meng et al. [18], motivated by the need to reduce the variance of REINFORCE, apply GS estimator. However, in their large-scale experiments they use temperature $\tau = 10^{-10}$. According to the previous section, the variance of GS estimator should go through the roof as it grows as $O(\frac{1}{\tau})$. It is practically prohibitive as the learning would require an extremely small learning rate and a very long training time as well as high numerical accuracy. Nevertheless, good experimental results are demonstrated [18]. We identify a hidden implementation issue which completely changes the gradient estimator and enables learning.

First, we explain the issue. Meng et al. [18] model stochastic binary weights as $w \sim \mathrm{Bin}(\mu)$ and express GS estimator as follows.

Proposition 6. (Meng et al. [18] Lemma 1). *Let* $w \sim Bin(\mu)$ *and let* $f\colon \{-1,1\} \to \mathbb{R}$ *be a loss function. Using parametrization* $\mu = \tanh(\lambda)$, $\lambda \in \mathbb{R}$, *GS estimator of gradient* $\frac{\mathrm{d}\mathbb{E}_w[f]}{\mathrm{d}\mu}$ *can be expressed as*

$$\delta \sim \frac{1}{2}\mathrm{Logistic}, \tag{14a}$$

$$\tilde{w} = \tanh_\tau(\lambda - \delta) \equiv \tanh(\frac{\lambda - \delta}{\tau}), \tag{14b}$$

$$J = \frac{1 - \tilde{w}^2}{\tau(1 - \mu^2)}, \tag{14c}$$

$$\hat{g} = Jf'(\tilde{w}), \tag{14d}$$

which we verify in Appendix B. However, the actual implementation of the scaling factor J used in the experiments [18] according to the published code[2] introduces a technical $\epsilon = 10^{-10}$ as follows:

$$J := \frac{1 - \tilde{w}^2 + \epsilon}{\tau(1 - \mu^2 + \epsilon)}. \tag{15}$$

It turns out this changes the nature of the gradient estimator and of the learning algorithm. The BayesBiNN algorithm [18, Table 1middle] performs the update:

$$\lambda := (1 - \alpha)\lambda - \alpha s f'(\tilde{w}), \tag{16}$$

where $s = NJ$, N is the number of training samples and α is the learning rate.

[2] https://github.com/team-approx-bayes/BayesBiNN.

Proposition 7. *With the setting of the hyper-parameters* $\tau = O(10^{-10})$ [18, Table 7] *and* $\epsilon = 10^{-10}$ *(author's implementation) in large-scale experiments (MNIST, CIFAR10, CIFAR100), the BayesBiNN algorithm is practically equivalent to the following deterministic algorithm:*

$$w := sign(\bar{\lambda}); \tag{17a}$$

$$\bar{\lambda} := (1-\alpha)\bar{\lambda} - \alpha f'(w). \tag{17b}$$

In particular, it does not depend on the values of τ *and* N*.*

Proof in Appendix B. Experimentally, we have verified, using authors implementation, that indeed parameters λ (16) grow to the order 10^{10} during the first iterations, as predicted by our calculations in the proof.

Notice that the step made in (17b) consists of a decay term $-\alpha\bar{\lambda}$ and the gradient descent term $-\alpha f'(w)$, where the gradient $f'(w)$ is a straight-through estimate for the *deterministic* forward pass $w = \text{sign}(\bar{\lambda})$. Therefore the deterministic ST is effectively used. It is seen that the decay term is the only remaining difference to the deterministic STE algorithm [18, Table 1left], the method is contrasted to. From the point of view of our study, we should remark that the deterministic ST estimator used in effect indeed decreases the variance (down to zero) however it increases the bias compared to the baseline stochastic ST [28].

The issue has also downstream consequences for the intended Bayesian learning. The claim of Proposition 7 that the method does not depend on τ and N is perhaps somewhat unexpected, but it makes sense indeed. The initial BayesBiNN algorithm of course depends on τ and N. However due to the issue with the implementation of Gumbel Softmax estimator, for a sufficiently small value of τ it falls into a regime which is significantly different from the initial Bayesian learning rule and is instead more accurately described by (17). In this regime, the result it produces does not dependent on the particular values of τ and N. While we do not know what problem it is solving in the end, it is certainly not solving the intended variational Bayesian learning problem. This is so because the variational Bayesian learning problem and its solution do depend on N in a critical way. The algorithm (17) indeed does not solve any variational problem as there is no variational distribution involved (nothing sampled). Yet, the decay term $-\alpha\lambda$ stays effective: if the data gradient becomes small, the decay term implements some small "forgetting" of the learned information and may be responsible for an improved generalization observed in the experiments [18].

5 FouST

Pervez et al. [22] introduced several methods to improve ST estimators using Fourier analyzes of Boolean functions [21] and Taylor series. The proposed methods are guided by this analysis but lack formal guarantees. We study the effect of the proposed improvements analytically.

One issue with the experimental evaluation [22] is that the baseline ST estimator [22, Eq. 7] is *misspecified*: it is adopted from the works considering $\{0, 1\}$ Bernoulli variables without correcting for $\{-1, 1\}$ case as in (7), differing by a

coefficient 2. The reason for this misspecifications is that ST is know rather as a folklore, vaguely defined, method (see [28]). While in learning with a simple expected loss this coefficient can be compensated by the tuned learning rate, it can lead to a more serious issues, in particular in VAEs with Bernoulli latents and deep SBNs. VAE training objective [14] has the data evidence part, where binary gradient estimator is required and the prior KL divergence part, which is typically computed analytically and differentiated exactly. Rescaling the gradient of the evidence part only introduces a bias which cannot be compensated by tuning the learning rate. Indeed, it is equivalent to optimizing the objective with the evidence part rescaled. In [28, Fig. 2] we show that this effect is significant. In the reminder of the section we will assume that the correct ST estimator (7) is used as the starting point.

5.1 Lowering Bias by Importance Sampling

The method [22, Sec. 4.1]"Lowering Bias by Importance Sampling", as noted by authors, obtains DARN gradient estimator [10, Appendix A] who derived it by applying a (biased) control variate estimate in the REINFORCE method. Transformed to the encoding with ± 1 variables, it expresses as

$$\hat{g}_{\text{DARN}} = f'(x)/p(x). \tag{18}$$

By design [10], this method is unbiased for quadratic functions, which is straightforward to verify by inspecting its expectation

$$\mathbb{E}[\hat{g}_{\text{DARN}}] = f'(1) + f'(0). \tag{19}$$

While, this is in general an improvement over ST—we may expect that functions close to linear will have a lower bias, it is not difficult to construct an example when it can increase the bias compared to ST.

Example 1. The method [22, Sec. 4.1] "Lowering Bias by Importance Sampling", also denoted as Importance Reweighing (IR), can increase bias.

Let $p \in [0, 1]$ and $x \sim \text{Bin}(p)$. Let $f(x) = |x + a|$. The derivative of $\mathbb{E}[f(x)]$ in p is

$$\frac{\mathrm{d}}{\mathrm{d}p}((1-p)f(-1) + pf(1)) = f(1) - f(-1) = f(1) = 2a. \tag{20}$$

The expectation of $\hat{g}_{\text{ST}}$ is given by

$$(1-p)2f'(-1) + p2f'(1) = 2(2p-1). \tag{21}$$

The expectation of $\hat{g}_{\text{DARN}}$ is given by

$$f'(-1) + f'(1) = 0. \tag{22}$$

The bias of DARN is $2|a|$ while the bias of ST is $2|a+1-2p|$. Therefore for $a > 0$ and $p > 0.5$, the bias of DARN estimator is higher. In particular for $a = 0.9$ and $p = 0.95$ the bias of ST estimator equals 0 while the bias of DARN estimator equals 1.8.

Furthermore, we can straightforwardly express its variance.

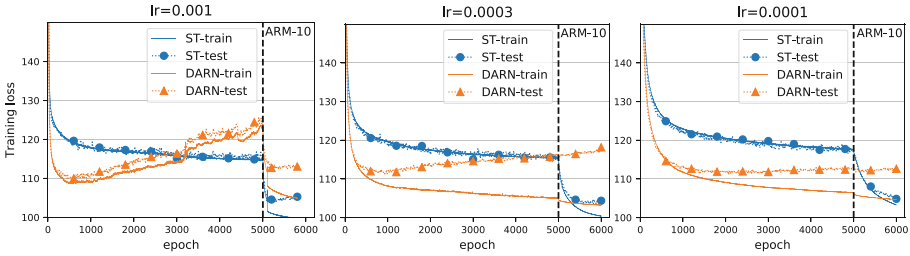

Fig. 3. Experimental comparison of DARN and ST estimators on MNIST VAE. The plots show training and test loss (negative ELBO) during training for different learning rates. After 5000 epochs, an unbiased ARM-10 estimator is applied in order to measure (and correct) the accumulated bias. At the smaller learning rates, where DARN does not diverge, it clearly has a much smaller accumulated bias but manages to overfit significantly.

Proposition 8. *The variance of* $\hat{g}_{\text{DARN}}$ *is expressed as*

$$\mathbb{V}_z[\hat{g}_{\text{DARN}}] = \frac{(f'(1) - p(f'(1) + f'(-1)))^2}{p(1-p)}. \tag{23}$$

It has asymptotes $O(\frac{f'(-1)^2}{1-p})$ *for* $p \to 1$ *and* $O(\frac{f'(1)^2}{p})$ *for* $p \to 0$.

The asymptotes indicate that the variance can grow unbounded for units approaching deterministic mode. If applied in a deep network with L layers, L expressions (18) are multiplied and the variance can grow respectively. Interestingly though, if the probability p is defined using the sigmoid function as $p = \sigma(\eta)$, then the gradient in η additionally multiplies by the Jacobian $\sigma'(\eta) = p(1-p)$, and the variance of the gradient in η becomes bounded. Moreover, a numerically stable implementation can simplify $p(1-p)/p(x)$ for both outcomes of x. We conjecture that this estimator can be particularly useful with this parametrization of the probability (which is commonly used in VAEs and SBNs).

Experimental evidence [11, Fig. 2.a], where DARN estimator is denoted as "$\frac{1}{2}$" shows that the plain ST performs similar for the structural output prediction problem. However, [11, Fig. 3.a] gives a stronger evidence in favor of DARN for VAE. In Fig. 3 we show experiment for the MNIST VAE problem, reproducing the experiment [11,22] (up to data binarization and implementation details). The exact specification is given in [28, Appendix D.1]. It is seen that DARN improves the training performance but needs an earlier stopping and or more regularization. Interestingly, with a correction of accumulated bias using unbiased ARM [35] method with 10 samples, ST leads to better final training and test performance.

5.2 Reducing Variance via the Fourier Noise Operator

The Fourier noise operator [22, Sec. 2] is defined as follows. For $\rho \in [0, 1]$, let $x' \sim N_\rho(x)$ denote that x' is set equal to x with probability ρ and chosen as

an independent sample from Bin(p) with probability $1-\rho$. The Fourier noise operator smooths the loss function and is defined as $T_\rho[f](x) = \mathbb{E}_{x' \sim N_\rho(x)}[f(x')]$. When applied to f before taking the gradient, it can indeed reduce both bias and variance, ultimately down to zero when $\rho = 0$. Indeed, in this case x' is independent of x and $T_\rho[f](x) = \mathbb{E}[f(x)]$, which is a constant function of x. However, the exact expectation in x' is intractable. The computational method proposed in [22, Sec. 4.2] approximates the gradient of this expectation using S samples $x^{(s)} \sim N_\rho(x)$ as

$$\hat{g}_\rho = \frac{1}{S} \sum\nolimits_s \hat{g}(x^{(s)}), \tag{24}$$

where $\hat{g}$ is the base ST or DARN estimator. We show the following.

Proposition 9. *The method* [22, Sec. 4.2] *"Reducing Variance via the Fourier Noise operator" does not reduce the bias (unlike T_ρ) and increases variance in comparison to the trivial baseline that averages independent samples.*

Proof in Appendix C.

This result is found in a sharp contradiction with the experiments [22, Figure 4], where independent samples perform worse than correlated. We do not have a satisfactory explanation for this discrepancy except for the misspecified ST. Since the author's implementation is not public, it is infeasible to reproduce this experiment in order to verify whether a similar improvement can be observed with the well-specified ST. Lastly, note, that unlike correlated sampling, uncorrelated sampling can be naturally applied with multiple stochastic layers.

5.3 Lowering Bias by Discounting Taylor Coefficients

For the technique [22, Sec. 4.3.1] "Lowering Bias by Discounting Taylor Coefficients" we present an alternative view, not requiring Taylor series expansion of f, thus simplifying the construction. Following [22, Sec. 4.3.1] we assume that the importance reweighing was applied. Since the technique samples f' at non-binary points, we refer to it as a *relaxed* DARN estimator. It can be defined as

$$\tilde{g}_{\text{DARN}}(x,u) = \frac{f'(xu)}{p(x)}, \text{where} \quad u \sim \mathcal{U}[0,1]. \tag{25}$$

In the total expectation, when we draw x and u multiple times, the gradient estimates are averaged out. The expectation over u alone effectively integrates the derivative to obtain:

$$\mathbb{E}_u\big[\tilde{g}_{\text{DARN}}(x,u)\big] = \begin{cases} \frac{1}{p}\int_0^1 f'(u)\mathrm{d}u = \frac{1}{p}(f(1)-f(0)), & \text{if} \quad x=1, \\ \frac{1}{1-p}\int_0^1 f'(-u)\mathrm{d}u = \frac{1}{1-p}(f(-1)-f(0)), & \text{if} \quad x=-1. \end{cases} \tag{26}$$

In the expectation over x we therefore obtain

$$\mathbb{E}_{x,u}[\tilde{g}_{\text{DARN}}(x,u)] = f(1) - f(-1), \tag{27}$$

which is the correct derivative. One issue, discussed by [22] is that variance increases (as there is more noise in the system). However, a major issue similar to GS estimator Sect. 3, reoccurs here, that all related expectations become biased. In particular (25) becomes biased in the presence of other variables Pervez et al. [22, Sec. 4.3.1] propose to use $u \in \mathcal{U}[a,1]$ with $a > 0$, corresponding to shorter integration intervals around ± 1 states, in order to find an optimal tradeoff.

5.4 Lowering Bias by Representation Rescaling

Consider the estimator $\hat{g}$ of the gradient of function $\mathbb{E}_x[f(x)]$ where $x \sim \text{Bin}(p)$. Representation rescaling is defined in [22, Algorithm 1] as drawing $\tilde{x} \sim \frac{1}{\tau}\text{Bin}(p)$ instead of x and then using FouST estimator based on the derivative $f'(\tilde{x})$. It is claimed that using a scaled representation can decrease the bias of the gradient estimate. However, the following issue occurs.

Proposition 10. *The method* [22, Sec. 4.3.2] *"Lowering Bias by Representation Rescaling" compares biases of gradient estimators of different functions.*

Proof. Sampling $\tilde{x}$ can be equivalently defined as $\tilde{x} = x/\tau$. Bypassing the analysis of Taylor coefficients [22], it is easy to see that for a smooth function f, as $\tau \to \infty$, $f(x/\tau)$ approaches a linear function of x and therefore the bias of the ST estimator of $\mathbb{E}_x[f(x/\tau)]$ approaches zero. However, clearly $\mathbb{E}_x[f(x/\tau)]$ is a different function from $\mathbb{E}_x[f(x)]$ which we wish to optimize. □

We explain, why this method nevertheless has effect. Choosing and fixing the scaling hyper-parameter τ is equivalent to staring from a different initial point, where (initially random) weights are scaled by $1/\tau$. At this initial point, the network is found to be closer to a linear regime, where the ST estimator is more accurate and possibly the vanishing gradient issue is mitigated. Thus the method can have a positive effect on the learning as observed in [22, Appendix Table 3].

6 Conclusion

We theoretically analyzed properties of several methods for estimation of binary gradients and gained interesting new insights.

- For GS and ST-GS estimator we proposed a simplified presentation for the binary case and explained detrimental effects of low and high temperatures. We showed that bias of ST-GS estimator approaches that of DARN, connecting these two techniques.
- For BayesBiNN we identified a hidden issue that completely changes the behavior of the method from the intended variational Bayesian learning with Gumbel-Softmax estimator, theoretically impossible due to the used temperature $\tau = 10^{-10}$, to non-Bayesian learning with deterministic ST estimator and latent weight decay. As this learning method shows improved experimental results, it becomes an open problem to clearly understand and advance the mechanism which facilitates this.

– In our analysis of techniques comprising FouST estimator, we provided additional insights and showed that some of these techniques are not well justified. It remains open, whether they are nevertheless efficient in practice in some cases for other unknown reasons, not taken into account in this analysis.

Overall we believe our analysis clarifies the surveyed methods and uncovers several issues which limit their applicability in practice. It provides tools and clears the ground for any future research which may propose new improvements and would need to compare with existing methods both theoretically and experimentally. We hope that this study will additionally motivate such research.

References

1. Alizadeh, M., Fernandez-Marques, J., Lane, N.D., Gal, Y.: An empirical study of binary neural networks' optimisation. In: ICLR (2019)
2. Bethge, J., Yang, H., Bornstein, M., Meinel, C.: Back to simplicity: how to train accurate BNNs from scratch? CoRR, abs/1906.08637 (2019)
3. Bulat, A., Tzimiropoulos, G.: Binarized convolutional landmark localizers for human pose estimation and face alignment with limited resources. In: ICCV (2017)
4. Bulat, A., Tzimiropoulos, G., Kossaifi, J., Pantic, M.: Improved training of binary networks for human pose estimation and image recognition. arXiv (2019)
5. Bulat, A., Martinez, B., Tzimiropoulos, G.: High-capacity expert binary networks. In: ICLR (2021)
6. Chaidaroon, S., Fang, Y.: Variational deep semantic hashing for text documents. In: SIGIR Conference on Research and Development in Information Retrieval, pp. 75–84 (2017)
7. Dadaneh, S. Z., Boluki, S., Yin, M., Zhou, M., Qian, X.: Pairwise supervised hashing with Bernoulli variational auto-encoder and self-control gradient estimator. ArXiv, abs/2005.10477 (2020)
8. Esser, S.K., et al.: Convolutional networks for fast, energy-efficient neuromorphic computing. Proc. Natl. Acad. Sci. **113**(41), 11441–11446 (2016)
9. Grathwohl, W., Choi, D., Wu, Y., Roeder, G., Duvenaud, D.: Backpropagation through the void: optimizing control variates for black-box gradient estimation. In: ICLR (2018)
10. Gregor, K., Danihelka, I., Mnih, A., Blundell, C., Wierstra, D.: Deep autoregressive networks. In: ICML (2014)
11. Gu, S., Levine, S., Sutskever, I., Mnih, A.: MuProp: unbiased backpropagation for stochastic neural networks. In: 4th International Conference on Learning Representations (ICLR), May 2016
12. Horowitz, M.: Computing's energy problem (and what we can do about it). In: International Solid-State Circuits Conference Digest of Technical Papers (ISSCC), pp. 10–14 (2014)
13. Jang, E., Gu, S., Poole, B.: Categorical reparameterization with gumbel-softmax. In: ICLR (2017)
14. Kingma, D.P., Welling, M.: Auto-encoding variational Bayes. CoRR, abs/1312.6114 (2013)
15. Liu, Z., Wu, B., Luo, W., Yang, X., Liu, W., Cheng, K.-T.: Bi-real net: enhancing the performance of 1-Bit CNNs with improved representational capability and advanced training algorithm. In: Ferrari, V., Hebert, M., Sminchisescu, C., Weiss,

Y. (eds.) ECCV 2018. LNCS, vol. 11219, pp. 747–763. Springer, Cham (2018). https://doi.org/10.1007/978-3-030-01267-0_44
16. Maddison, C.J., Mnih, A., Teh, Y.W.: The concrete distribution: a continuous relaxation of discrete random variables. In: ICLR (2017)
17. Martínez, B., Yang, J., Bulat, A., Tzimiropoulos, G.: Training binary neural networks with real-to-binary convolutions. In: ICLR (2020)
18. Meng, X., Bachmann, R., Khan, M.E.: Training binary neural networks using the Bayesian learning rule. In: ICML (2020)
19. Mnih, A., Gregor, K.: Neural variational inference and learning in belief networks. In: ICML of JMLR Proceedings, vol. 32, pp. 1791–1799 (2014)
20. Ñanculef, R., Mena, F.A., Macaluso, A., Lodi, S., Sartori, C.: Self-supervised Bernoulli autoencoders for semi-supervised hashing. CoRR, abs/2007.08799 (2020)
21. O'Donnell, R.: Analysis of Boolean Functions. Cambridge University Press, Cambridge (2014). ISBN 1107038324
22. Pervez, A., Cohen, T., Gavves, E.: Low bias low variance gradient estimates for Boolean stochastic networks. In: ICML, vol. 119, pp. 7632–7640 (2020)
23. Peters, J.W., Welling, M.: Probabilistic binary neural networks. arXiv preprint arXiv:1809.03368 (2018)
24. Raiko, T., Berglund, M., Alain, G., Dinh, L.: Techniques for learning binary stochastic feedforward neural networks. In: ICLR (2015)
25. Rastegari, M., Ordonez, V., Redmon, J., Farhadi, A.: XNOR-Net: ImageNet classification using binary convolutional neural networks. In: Leibe, B., Matas, J., Sebe, N., Welling, M. (eds.) ECCV 2016. LNCS, vol. 9908, pp. 525–542. Springer, Cham (2016). https://doi.org/10.1007/978-3-319-46493-0_32
26. Roth, W., Schindler, G., Fröning, H., Pernkopf, F.: Training discrete-valued neural networks with sign activations using weight distributions. In: Brefeld, U., Fromont, E., Hotho, A., Knobbe, A., Maathuis, M., Robardet, C. (eds.) ECML PKDD 2019. LNCS (LNAI), vol. 11907, pp. 382–398. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-46147-8_23
27. Shayer, O., Levi, D., Fetaya, E.: Learning discrete weights using the local reparameterization trick. In: ICLR (2018)
28. Shekhovtsov, A., Yanush, V.: Reintroducing straight-through estimators as principled methods for stochastic binary networks. In: GCPR (2021)
29. Shekhovtsov, A., Yanush, V., Flach, B.: Path sample-analytic gradient estimators for stochastic binary networks. In: NeurIPS (2020)
30. Shen, D., et al.: NASH: toward end-to-end neural architecture for generative semantic hashing. In: Annual Meeting of the Association for Computational Linguistics (2018)
31. Tang, W., Hua, G., Wang, L.: How to train a compact binary neural network with high accuracy? In: AAAI (2017)
32. Tucker, G., Mnih, A., Maddison, C.J., Lawson, J., Sohl-Dickstein, J.: REBAR: low-variance, unbiased gradient estimates for discrete latent variable models. In: NeurIPS (2017)
33. Vahdat, A., Andriyash, E., Macready, W.: Undirected graphical models as approximate posteriors. In: ICML, vol. 119, pp. 9680–9689 (2020)
34. Xiang, X., Qian, Y., Yu, K.: Binary deep neural networks for speech recognition. In: INTERSPEECH (2017)
35. Yin, M., Zhou, M.: ARM: augment-REINFORCE-merge gradient for stochastic binary networks. In: ICLR (2019)
36. Zhou, S., Wu, Y., Ni, Z., Zhou, X., Wen, H., Zou, Y.: DoReFa-Net: training low bitwidth convolutional neural networks with low bitwidth gradients. arXiv preprint arXiv:1606.06160 (2016)