Low Bias Low Variance Gradient Estimates for Boolean Stochastic Networks

Adeel Pervez [1] **Taco Cohen** [2] **Efstratios Gavves** [1]

Abstract

Stochastic neural networks with discrete random variables are an important class of models for their expressiveness and interpretability. Since direct differentiation and backpropagation is not possible, Monte Carlo gradient estimation techniques are a popular alternative. Efficient stochastic gradient estimators, such Straight-Through and Gumbel-Softmax, work well for shallow stochastic models. Their performance, however, suffers with hierarchical, more complex models. We focus on stochastic networks with Boolean latent variables. To analyze such networks, we introduce the framework of harmonic analysis for Boolean functions to derive an analytic formulation for the bias and variance in the Straight-Through estimator. Exploiting these formulations, we propose *FouST*, a low-bias and low-variance gradient estimation algorithm that is just as efficient. Extensive experiments show that FouST performs favorably compared to state-of-the-art biased estimators and is much faster than unbiased ones.

1. Introduction

Stochastic neural networks with discrete latent variables have been an alluring class of models for their expressivity and interpretability, dating back to foundational work on Helmholtz machines (Dayan et al., 1995) and sigmoid belief nets (Neal, 1992). Since they are not directly differentiable, discrete random variables do not mesh well with the workhorse of modern Deep Learning, that is the backpropagation algorithm. Monte Carlo gradient estimation is an effective solution where, instead of computing the true gradients, one can sample gradients from some distribution. The sample estimates can be either biased or unbiased. Unbiased gradient estimates like score function estimators (Williams, 1992) come typically at the cost of high variance leading to slow learning. In contrast, biased gradient estimates such Straight-Through (Bengio et al., 2013), while efficient, run the risk of convergence to poor minima and unstable training. To this end several solutions have recently been proposed that either reduce variance in unbiased estimators (Mnih & Gregor, 2014; Gu et al., 2016; Tucker et al., 2017; Rezende et al., 2014; Grathwohl et al.) or control bias in biased estimators (Jang et al., 2017; Maddison et al., 2017). These methods, however, have difficulty scaling up to hierarchical neural networks with multiple stochastic layers: low-variance unbiased estimators are too expensive [1] , while the compounded bias from the continuous relaxations on multiple stochastic layers leads to poor minima. In this work we focus on biased estimators.

Our goal in this paper is a gradient estimator for Boolean random variables that works for any complex hierarchical neural networks also. We resort to the term *Boolean* instead of binary to emphasize that we work directly on the Boolean space $\{-1, +1\}$, without any continuous relaxations or quantizations. With this in mind we re-purpose the framework of harmonic analysis of Boolean functions, widely used in computational learning and computational complexity theory (O'Donnell, 2014; Linial et al., 1993; Mossel et al., 2003; Mansour, 1994). We cast stochastic neural networks as Boolean functions $f(z)$ over Boolean latent variables z sampled from probability distributions $p(z)$. We then use harmonic analysis to determine that the bias in the Straight-Through gradient estimates corresponds to the weighted sum of higher-order Taylor coefficients of $f(z)$. Moreover, we show that variance in stochastic neural networks is caused by higher order Fourier coefficients. The direct consequence is that one can *control the bias and the variance in the Straight-Through estimator by manipulating the higher-order Fourier and Taylor coefficients of* $f(z)$. Building upon the harmonic analysis of existing gradient estimators, we present an algorithm, *FouST*, that admits low bias and low variance gradient estimates for Boolean latent variable models. In experiments, we are able to train complex hierarchical neural networks with no difficulty,

[1]QUVA Lab, Informatics Institute, University of Amsterdam, The Netherlands [2]Qualcomm AI Research, Qualcomm Technologies Netherlands B.V., The Netherlands. Correspondence to: Adeel Pervez <a.a.pervez@uva.nl>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

[1]Training a nonlinear sigmoid belief network model on GPU with two stochastic layers on MNIST with REBAR took 1.5 days.

Algorithm 1 FouST Gradient Estimator

Require: Parameters $\theta \in \mathbf{R}^K$, Bernoulli Representation $\{-1/\tau, +1/\tau\}$, Interval Parameter $b \in [0, a]$, Constant scaling parameter γ, $NoiseOp$ flag, T NoiseOp samples, ρ

1: Sample $z_i \sim p_{\theta_i}(z_i), i = 1, ..., K$
2: Sample $y_i \sim \text{Unif}(b, a)$
3: Set $y_i := z_i * y_i$
4: Compute ST gradient $\partial_{\theta_i} := \partial_{y_i} f(y)$
5: Importance reweighing $\partial_{\theta_i} := 0.5/p(z_i) \cdot \partial_{\theta_i}$
6: **if** not $NoiseOp$ **then**
7: **return** partial gradient $\gamma \partial_{\theta_i}, i = 1, ..., K$
8: **else**
9: Sample z^k from $N_\rho(z)$, $k = 1, \dots T$
10: Run steps 1-5 for each z^k
11: **return** average of T partial gradients.
12: **end if**

while scoring high accuracies compared to state-of-the-art baselines. We summarize FouST in Algorithm 1.

We make the following three contributions.

1. We introduce the framework of harmonic analysis of Boolean functions to analyze discrete stochastic neural networks and gradients. Under this framework, we quantify the bias and variance in stochastic gradients, associating them with higher order Taylor and Fourier coefficients respectively.

2. Based on the harmonic analysis and quantification of bias and variance, we present *FouST*. FouST is a gradient estimator that admits low bias and low variance by manipulating higher order Taylor and Fourier coefficients, as guided by the Harmonic Analysis.

3. As a side contribution, based on our theoretical analysis we show that the DARN (Gregor et al., 2014) gradient estimator – originally motivated in terms for unbiased control variates for quadratic functions – can be interpreted as a lower-bias version of Straight-Through.

We continue with briefly introducing the framework of harmonic analysis for Boolean functions in section 2. We use this framework in section 3 to interpret true stochastic gradients as Fourier coefficients. In section 4 we describe the proposed low-bias, low-variance estimator. We close with related work in section 5, experimental validation in section 6 and a conclusion in section 7.

2. Harmonic Analysis of Boolean Functions

We consider Boolean functions on the n-dimensional Boolean cube, $f : \{-1, +1\}^n \to \mathbb{R}$. The setting of Harmonic Analysis for Boolean functions is the space of Boolean functions f with a *product probability distribution* on the Boolean input, that is $p(z) = \prod_{i=1}^n p_i(z)$. We denote p_i as the probability of the i-th dimension being one, *i.e.*, $p_i := p_i(z_i = +1)$. We denote the mean and variance of z_i by μ_i and σ_i, respectively.

An example of a Boolean function in this setting is a generative neural network $f : z \mapsto y$ with a factorized latent distribution, as commonly done in representation learning (Kingma & Welling, 2014; Higgins et al., 2017). In this example, z is the stochastic Boolean input taking only two possible values, $+1$ or -1. The stochastic input is also known as the latent code in stochastic neural networks. y is the output of the neural network, like the cross entropy loss. Often, the goal of a generative neural network is to learn or approximate the latent distribution given input data x, *i.e.*, $p(z|x)$, which we explore in the experiments.

Next, we introduce a few basic operations in the context of Harmonic Analysis of Boolean functions, which we need later on. To improve readability, we collect all important notations in table 1. We include further details in the appendix. For a more comprehensive introduction, we refer the reader to O'Donnell (2014).

Inner product. The inner product of two Boolean functions f and g is: $\langle f, g \rangle = \mathbb{E}_{p(z)}[f(z)g(z)]$. To have cleaner notation, from here on we drop $p(z)$ from the subscript of the expectation $\mathbb{E}[\cdot] = \mathbb{E}_{p(z)}[\cdot]$ whenever an expectation is taken with respect to the whole vector z.

Orthonormal basis. Let S be any subset of dimensions of the n-dimensional Boolean cube, $S \subseteq [n] = \{1, ..., n\}$. Per S we define a basis function, $\phi_S(z) := \prod_{i \in S} \phi_i(z_i)$, where for the empty set $\phi_\varnothing(z) = 1$ and for the i-th dimension $\phi_i := \frac{z_i - \mu_i}{\sigma_i}$, *i.e.*, the z-score normalized dimension. As an example, we take the case of the uniform Bernoulli distribution, namely $p_i(z_i = +1) = 1/2$ and $p_i(z_i = -1) = 1/2$. In that case, the basis for the i-th dimension is $\phi_i(z_i) = z_i$, as $\mu_i = 0$ and $\sigma_i = 1$.

As z is n-dimensional, the total number of basis functions ϕ_S is 2^n for all possible subsets in $[n]$. In expectation, the inner product between the basis functions forms the identity matrix

$$\begin{aligned} \mathbb{E}_{p(z)}[\phi_i(z_i)] &= 0, \\ \mathbb{E}_{p(z)}[\phi_i(z_i)^2] &= 1, \\ \mathbb{E}_{p(z)}[\phi_i(z_i)\phi_j(z_j)] &= \mathbb{E}_{p(z)}[\phi_i(z_i)]\ \mathbb{E}_{p(z)}[\phi_j(z_j)] = 0, i \neq j. \end{aligned} \tag{1}$$

where the last identity derives from the independence of any dimensions $i \neq j$. The ϕ_S functions, therefore, form an orthonormal basis for the space of Boolean functions f.

Fourier expansion of Boolean functions. We expand the boolean function f on the set of 2^n orthonormal basis functions,

$$f(z) = \sum_{S \subset [n]} \hat{f}^{(p)}(S)\phi_S(z). \tag{2}$$

The Fourier coefficients $\hat{f}^{(p)}(S)$ are computed by the inverse Fourier expansion,

$$\hat{f}^{(p)}(S) = \mathbb{E}[f(z)\phi_S(z)], \tag{3}$$

namely the inner product of the Boolean function $f(z)$ with the basis functions under distribution $p(z)$. We write $\hat{f}(S)$ for $f^{(p)}(S)$, when there is no confusion regarding $p(z)$. The degree of a coefficient $\hat{f}^{(p)}(S)$ is the cardinality of S.

- There is only a single degree-0 coefficient for the empty set, $\hat{f}^{(p)}(\varnothing)$. Recalling that $\phi_\varnothing = 1$, the degree-0 coefficient equals to the expected value of f under $p(z)$, $\hat{f}^{(p)}(\varnothing) = \mathbb{E}[f(z)]$.
- The variance of a function can be given in terms of Fourier coefficients as

$$\sigma^2(f) = \mathbb{E}[(f - \mathbb{E}[f])^2] = \sum_{S \subseteq [n], S \neq \varnothing} \hat{f}(S)^2, \tag{4}$$

  where the last equation follows by Plancherel's theorem: $\langle f, f \rangle = \sum_{S \subseteq [n]} \hat{f}(S)^2$.

As the coefficients depend on the distribution $p(z)$, the expansion in (2) is known as the p-biased Fourier expansion of the Boolean function $f(z)$.

Fourier Noise Operator. Given $z, z' \in \{-1, 1\}^n$ and $\rho \in [0, 1]$, we say that z, z' are *ρ-correlated* if z' is generated by independently setting each z'_i to z_i with probability ρ and a sample from p_i with probability $1 - \rho$. We denote this by $z' \sim N_\rho(z)$. We use this to define the *noise operator* T_ρ acting on f as the expectation over ρ-correlated inputs as follows:

$$T_\rho[f](z) = \mathbb{E}_{z' \sim N_\rho(z)}[f(z')]$$

The noise operator has a special action on the functions ϕ_i, multiplying them by ρ. For the basis function ϕ_S this translates to multiplication by $\rho^{|S|}$:

$$T_\rho[\phi_S](z) = T_\rho\left(\prod_{i \in S} \phi_i\right)(z) = \prod_{i \in S} T_\rho[\phi_i](z) = \rho^{|S|}\phi_S(z).$$

By linearity of expectation it follows that

$$T_\rho[f](z) = \sum_{S \subseteq [n]} \hat{f}(S) T_\rho[\phi_S](z) = \sum_{S \subseteq [n]} \rho^{|S|}\hat{f}(S)\phi_S(z).$$

Table 1. A summary of notation.

z	Boolean variable $\in \{-1 + 1\}$
$f(z)$	Boolean function, VAE decoder loss
$\phi_S(z)$	Basis for subset S
$\hat{f}^{(p)}(S)$	Fourier coefficient of f for S under $p(z)$
$T_\rho[f](z)$	Noise operator with parameter ρ that averages inputs z' correlated with z
$\partial_{p_i} \mathbb{E}[f(z)]$	True gradient for the loss $\mathbb{E}_{p(z)}[f(z)]$
$\mathbb{E}[\partial_{z_i} f(z)]$	Straight-Through gradient
$bias^{(p)}(g^i_{\text{ST}})$	Bias of the gradient estimate for i-th dimension under distribution $p(z)$
$p^{(i \to 1/2)}$	Uniform Bernoulli distribution for $p(z_i)$, *i.e.*, $p(z_i = +1) = 1/2$

Following equation (4), the variance of the new function $T_\rho[f](z)$ is now given by

$$\sum_{S \subseteq [n], S \neq \varnothing} \rho^{2|S|}\hat{f}(S)^2. \tag{5}$$

3. Harmonic Analysis of Straight-Through

We start from a stochastic neural network and optimize the model parameters with respect to a loss function $\mathcal{L} := \mathbb{E}_{p(z)}[f(z)]$ by taking the gradients

$$\partial_{p_i} \mathcal{L} = \partial_{p_i} \mathbb{E}_{p(z)}[f(z)] \tag{6}$$

per dimension. As the latent codes z are obtained by randomly sampling from $p(z)$ –a non-differentiable operation– backpropagation is inapplicable. To this end, Bengio et al. (2013) propose the Straight-Through estimator that approximates the gradient with

$$\partial_{p_i} \mathcal{L} = \mathbb{E}_{p(z)}[\partial_{z_i} f(z)] \tag{7}$$

Due to the mismatch in the forms of the true and the Straight through gradients, the approximation is biased. Our goal is to understand and quantify the source of this bias.

In our setting, the Boolean function $f(z)$ is a neural network model operating on the Boolean latent input. The strategy, therefore, is to study the Boolean function $f(z)$ under the framework of Harmonic Analysis of Boolean functions.

We first derive a relation between the true gradient in (6) and the Fourier coefficients for $f(z)$. Specifically, we show that the true gradient for the i-th dimension is equal to the degree-1 Fourier coefficient for the i-th dimension. By taking a Taylor expansion of *(i)* the true gradient -or its Fourier coefficient to be precise, and *(ii)* the Straight-Through gradient, we can compare the two. The term-by-term comparison quantifies the source of the bias in the Straight-Through gradient, yielding the following lemma, where $bias^{(p)}(g^i_{\text{ST}})$ denotes the bias of the i-th gradient estimate, g^i_{ST}, under

the general distribution $p(z)$; $bias^{(p^{i\to 1/2})}\left(g^i_{\text{ST}}\right)$ is the bias under the uniform Bernoulli distribution for the ith variable, while the other dimensions remain unchanged; and, c_k are the Taylor coefficients for the i-th dimension on $f(z)$ around 0, where we drop i from the superscript for clarity.

Lemma 1. *The bias in the Straight-Through estimator, g^i_{ST}, is caused by the mismatch of higher-order odd terms in the Taylor expansion of g^i and g^i_{ST} and is equal to*

$$bias^{(p)}\left(g^i_{ST}\right) = bias^{(p^{i\to 1/2})}\left(g^i_{ST}\right) + \mathbb{E}\left[\sum_{k=2j,j>0} kc_k\right]\mu_i. \tag{8}$$

Proof. For compactness, we provide only a sketch for the proof, keeping the steps needed later for the final estimator. For the detailed proof please refer to the appendix.

True gradient as degree-1 Fourier coefficient. Starting from the true gradient in (6), we can show that it is equivalent to the degree-1 Fourier coefficients under the uniform Bernoulli distribution.

$$\partial_{p_i}\mathbb{E}_{p(z)}[f(z)] = 2\frac{\hat{f}^{(p)}(i)}{\sigma_i} = 2\hat{f}^{(p\to 1/2)}(i) \tag{9}$$

$$= 2\,\mathbb{E}_{p^{(i\to 1/2)}(z)}[f(z)z_i] \tag{10}$$

We give the detailed proof for obtaining this identity in the appendix.

Taylor expansion of true gradient. $f(z)$ can be further analyzed in terms of Taylor expansion centered around 0,

$$f(z) = c_0 + c_1 z_i + c_2 z_i^2 + c_3 z_i^3 + \cdots \tag{11}$$

where $c_k k! = \partial^k_{z_i} f(z)|_{z_i=0}$ are the Taylor coefficients. Note that all c_k are a function of $z_j, j \neq i$. Replacing $f(z)$ in (10) with (11) , we have that

$$\partial_{p_i}[\mathbb{E}_{p(z)}[f(z)]] = 2\,\mathbb{E}_{p^{i\to 1/2}(z)}[c_0 z_i + c_1 z_i^2 + c_2 z_i^3 + \cdots] \tag{12}$$

$$= 2\,\mathbb{E}_{p(z_{\setminus i})}[c_1 + c_3 + \cdots], \tag{13}$$

using $z_i^{2k} = 1$ and $z_i^{2k+1} = z_i$ to go from (12) to (13). According to (13), the true gradient of $f(z)$ with respect to p_i is equal to the expected sum of the odd Taylor coefficients of $f(z)$.

Taylor expansion of Straight-Through gradient. We take the derivative of the Taylor expansion in (11) with respect to z_i,

$$\partial_{z_i} f(z) = c_1 + 2c_2 z_i + 3c_3 z_i^2 + 4c_4 z_i^3 + \ldots \tag{14}$$

The Straight-Through estimator is the expectation of (14), that is

$$\mathbb{E}_{p(z)}[\partial_{z_i} f(z)] = \mathbb{E}_{p(z)}[c_1 + 2c_2 z_i + 3c_3 z_i^2 + \cdots] \tag{15}$$

$$= \mathbb{E}_{p(z_{\setminus i})}[c_1 + 3c_3 + 5c_5 + \cdots] + \mathbb{E}_{p(z_{\setminus i})}[2c_2 + 4c_4 + \cdots]\mu_i \tag{16}$$

where $\mathbb{E}_{p(z_i)}[z_i] = \mu_i$.

Comparison of Taylor terms of true and Straight-Through gradients. To determine the bias of the Straight-Through gradient, we subtract the Taylor expansion of the true gradient (13) from the Taylor expansion of the Straight-Through gradient (16),

$$bias^{(p)}(g^i_{\text{ST}}) = \mathbb{E}_{p(z)}[\partial_{z_i} f(z)] - \partial_{p_i}[\mathbb{E}_{p(z)}[f(z)]] \tag{17}$$

$$= \mathbb{E}_{p(z_{\setminus i})}\left[\sum_{k=2j+1,j>0}(k-2)c_k\right] + \mathbb{E}_{p(z_{\setminus i})}\left[\sum_{k=2j,j>0} kc_k\right]\mu_i, \tag{18}$$

$$= bias^{(p\to 1/2)}(g^i_{\text{ST}}) + \mathbb{E}_{p(z_{\setminus i})}\left[\sum_{k=2j,j>0} kc_k\right]\mu_i. \tag{19}$$

To obtain the last identity, note that the expectation in (16) under the uniform Bernoulli distribution sets $\mu_i = 0$ and vanishes the second term. □

Lemma 1 introduces two perspectives on the source of bias for the Straight-Through estimator. According to (18), the bias is a series of Taylor terms. The higher order coefficients contribute more to the bias because of the multiplying factors $k-1$ and k for odd and even terms respectively. We refer to this as the *Taylor* formulation. According to (19), part of the bias is due to the mismatch of the generative distribution $p(z_i)$ from the uniform Bernoulli distribution $p^{(i\to 1/2)}(z_i)$. We refer to this as the *distribution mismatch* formulation.

4. Low-bias & Variance Gradient Estimates

We start from the two formulations for the source of bias in the Straight-Through estimator to design a better gradient estimator. The method yields gradient estimation that is just as efficient as the Straight-Through but with low bias with a *single sample* evaluation of the decoder. We also show that we can couple the estimator with the Fourier Noise operator to reduce variance, which in turn increases training stability with a small extra cost.

We coin the final algorithm *FouST*, for Fourier Straight-Through estimator, and summarize the steps in Algorithm 1.

4.1. Lowering Bias by Importance Sampling

Starting from the distribution mismatch formulation in (19), the total bias under any (unknown) distribution $p(z)$ decomposes to the bias under the uniform Bernoulli $p^{(i\to 1/2)}(z)$ and bias from the non-zero higher-order harmonics in expectation under $p(z)$. This gives a straightforward way to

lower bias: encourage the gradient estimator under $p(z)$ to approach the bias under the uniform Bernoulli, thus setting the bias from higher order harmonics to 0:

$$bias^{(p)}(g^i_{\text{ST}}) \to bias^{(p\to 1/2)}(g^i_{\text{ST}}) \Rightarrow$$
$$\mathbb{E}_{p(z_{\setminus i})}\left[\sum_{k=2j,j>0} kc_k\right]\mu_i \to 0$$

That is, we can lower the bias for the Straight-Through estimator if we sample from the uniform Bernoulli $p^{(i\to 1/2)}(z)$ distribution rather than $p(z)$.

One way to satisfy this condition is to resort to importance sampling from $p^{(i\to 1/2)}(z)$ instead of $p(z)$. To this end, we first get samples from $p(z)$ and compute the gradients $\partial_{z_i} f(z)$ as we would normally do with the Straight-Through. Then, we correct the gradient by estimating the expectation under $p^{(i\to 1/2)}(z)$, namely we multiply and divide by $p(z)$ per dimension

$$\mathbb{E}_{p^{(i\to 1/2)}(z)}[\partial_{z_i} f(z)] = \mathbb{E}_{p(z)}\left[\frac{1}{2p(z_i)}\partial_{z_i} f(z)\right], \quad (20)$$

recalling that $p^{(i\to 1/2)}(z_i) = 1/2$. Interestingly, Gregor et al. (2014) arrive at a similar proposition, albeit in the context of *unbiased control variates* for quadratic functions. Our derivation provides the additional insight that importance sampling with respect to the uniform Bernoulli distribution, as in (20), reduces the bias of the Straight-Through estimator.

4.2. Reducing Variance via the Fourier Noise operator

A common observation when using biased gradient estimators such as straight-through is that training tends to become unstable. This observation is usually attributed to the bias in the gradient estimator. On the other hand, we observe in our experiments that it is a combination of bias and variance that leads to instability and that training stability can be improved by lowering variance even when biased estimators are employed.

In this section we propose the application of the Fourier noise operator to construct lower variance functions to improve optimization. The Fourier noise operator takes as input a Boolean function f to produce another function $T[f](z)$ which is similar to f but with significantly smaller dependence on higher degree terms leading to reduced variance. The noise operator can be thought of as a smoothing operator where the output function is produced by taking inputs z' that are correlated with z and averaging the given function f evaluated on the correlated inputs. The Fourier expansion of this function is reproduced in the following, where $\rho \in [0, 1]$ is a parameter.

$$T_\rho[f](z) = \sum_{S\subseteq[n]} \rho^{|S|}\hat{f}(S)\phi_S(z).$$

Such a function has a number of properties that make it attractive, all of which can be seen from the Fourier expansion. The first is that expected values of the two functions remain the same, since the degree 0 Fourier coefficients are the same for both $f(z)$ and $T[f](z)$. The second property is that the variance of the resulting function is much lower than that of the original function, since the new Fourier coefficients are given by $\rho^{|S|}\hat{f}(S)$ leading to smaller coefficients for higher order terms (see equation (5) for the variance expression).

Given a VAE decoder function f that has high variance, we can replace the decoder with its noise operated version $T_\rho[f]$ to reduce variance without affecting the output value in expectation. The choice of the parameter ρ determines the extent to which variance is reduced, with $\rho = 1$ corresponding to no reduction and $\rho = 0$ being a simple expectation of f over independent samples and hence 0 variance. In our experiments we observe a significant improvement when using correlated samples (with ρ around 0.5) over independent samples (corresponding to $\rho = 0$).

4.3. Lowering Bias by Discounting Taylor Terms

Continuing to the Taylor formulation in (18), the total bias can also be seen as the sum of rescaled, higher order Taylor terms. We can lower the bias by discounting the contribution of these higher order Taylor terms. This can be achieved by modifying the function $f(z)$ to yield Taylor terms with discounted coefficients c_{k+1}. It can also be achieved by rescaling the representation of the Boolean variables z_i^k to discount the higher order Taylor terms.

4.3.1. Discounting Taylor Coefficients c_{k+1}

We recall that the moments of the uniform distribution over an interval $[a, b]$:

$$\int_a^b \frac{1}{b-a}u^k du = \frac{b^{k+1}-a^{k+1}}{(k+1)(b-a)} \quad (21)$$
$$= \frac{b^k}{(k+1)} \qquad \text{(when } a = 0\text{)} \quad (22)$$

We make two observations. First, the numerator contains b^k, which has the same form as z_i^k. Second, the denominator contains $k + 1$, which discounts the multiplying factors in (18). This suggests that we can feed uniformly sampled inputs from $[0, b]^n$ to the function to obtain similar Taylor terms z_i^k with discounted c_{k+1}. Next, we describe the precise steps to use the uniformed samples to lower bias.

First, we sample z from the latent Bernoulli distribution. We assume that we want the gradient relative to the variable z_1 and that $\mu_1 = 0$ or the importance sampling correction has been applied. Next, we sample $u = (u_1, \ldots, u_n)$ such that u_i is sampled from the uniform distribution over $[0, z_i]$ if

$z_i = 1$ or $[z_i, 0]$ if $z_i = -1$. We then estimate the gradient as $\partial_{u_1} f(u)$. To express the expectation of the gradient under such random sampling we look at the multivariate Taylor expansion of f and the respective gradients.

$$f(z) = \sum_{\alpha} c_\alpha \prod_i z_i^{\alpha_i}. \tag{23}$$

where $\alpha \in \mathbf{N}^n$ a multi-index. Each component α_i is the exponent of z_i (similar to k above). The true gradient $\partial_{p_1} E[f(z)]$ depends on the terms where α_1 is odd and for all $i \neq 1$, α_i is even and is expressed as follows.

$$\partial_{p_1} \mathbb{E}[f(z)] = \sum_{\substack{\alpha:\alpha_1 odd \\ \alpha_i even, i\neq 1}} c_\alpha \prod_{i\neq 1} \mathbb{E}[z_i^{\alpha_i}] \tag{24}$$

$$= \sum_{\substack{\alpha:\alpha_1 odd \\ \alpha_i even, i\neq 1}} c_\alpha \tag{25}$$

The expected gradient under our random sampling is given as follows,

$$\mathbb{E}_{z,u}[\partial_{u_1} f(u)] = \sum_{\substack{\alpha:\alpha_1 odd \\ \alpha_i even, i\neq 1}} \alpha_1 c_\alpha \, \mathbb{E}_{z,u}[z_1^{\alpha_1 - 1}] \prod_{i\neq 1} \mathbb{E}_{z,u}[z_i^{\alpha_i}] \tag{26}$$

$$= \sum_{\substack{\alpha:\alpha_1 odd \\ \alpha_i even, i\neq 1}} \alpha_1 c_\alpha \frac{z_1^{\alpha_1 - 1}}{\alpha_1} \prod_{i\neq 1} \mathbb{E}_z[\frac{z_i^{\alpha_i}}{\alpha_i + 1}] \tag{27}$$

$$= \sum_{\substack{\alpha:\alpha_1 odd \\ \alpha_i even, i\neq 1}} c_\alpha \prod_{i\neq 1} \mathbb{E}_z[\frac{z_i^{\alpha_i}}{\alpha_i + 1}], \tag{28}$$

where we now use z, u in the subscript of the expectations for clarity. A comparison between the true gradient expression (24) and the expected gradient in (28) shows that

1. This gives the correct coefficient for all terms where $\alpha_i = 0$ for $i \neq 1$. That is, for all terms in which only z_1 appears (for any degree).

2. The estimate becomes progressively worse as the total degree of other terms (except for dimension 1 in our use case) becomes larger.

Sampling from the uniform distribution can lead to bias reduction for functions with small coefficients. For functions with large coefficients we can use smaller interval $[a, b]$ by increasing a in (21). In that case, if we repeat the above analysis we discount low degree terms less and high degree terms more. In practice, we find that setting $a = 0$ and focusing on the low degree terms suffices across datasets and setups.

Last, we emphasize that u serves only as an *auxiliary variable*; its purpose is to help compute a gradient of lower bias and has no learned parameters. In the end, the only latent variable is the Boolean z. A potential problem from the proposed uniform sampling is increased variance. We can control the variance by incorporating the Fourier noise operator as described in section 4.2.

4.3.2. Discounting by Representation Rescaling

We can further discount higher order Taylor terms and lower bias by rescaling the representation of z_i. Specifically, we transform z_i by $h(z_i) = z_i/\tau$. $h(z_i)$ is still a valid Boolean representation, as $h(z) \in \{-1/\tau, +1/\tau\}$ is a two valued set. After the transformation $h(z)$, the gradient of the Taylor expansion of $f(z)$ in (14) becomes

$$\partial_{z_i} f(z) = c_1 + 2c_2 z_i/\tau + 3c_3 z_i^2/\tau^2 + 4c_4 z_i^3/\tau^3 + ... \tag{29}$$

In expectation, the Straight-Through gradient then becomes

$$\mathbb{E}_{p(z)}[\partial_{z_i} f(z)] = \mathbb{E}_{p(z)}[c_1 + 2c_2 z_i/\tau + ...] \tag{30}$$

$$= \mathbb{E}_{p(z_{\setminus i})}[c_1/\tau + 3c_3/\tau^3 + \cdots] + \mathbb{E}_{p(z_{\setminus i})}[2c_2/\tau^2 + \cdots]\mu_i \tag{31}$$

In comparison, the true gradient is

$$\partial_{p_i}[\mathbb{E}_{p(z)}[f(z)]] = 2\,\mathbb{E}_{p(z_{\setminus i})}[c_1/\tau + c_3/\tau^3 + \cdots]. \tag{32}$$

By comparing (31) and (32), the gradient estimate has lower bias by damping higher order Taylor terms by inverse powers of τ. The Taylor expansion after scaling the input in (31) implies that larger values for τ would lead to lower bias. However, as $1/\tau \to 0$ the sampling is confined to a smaller interval and the gradient estimate becomes more deterministic, thus causing worse performance. In practice, we treat τ as a hyperparameter tuned with cross-validation.

5. Related Work

Monte Carlo gradient estimators for training models with stochastic variables can be biased or unbiased. Perhaps the best known example of an unbiased gradient estimator is the REINFORCE algorithm (Williams, 1992). Unfortunately, REINFORCE gives gradients of high variance. For continuous stochastic variables Kingma & Welling (2014) propose the reparameterization trick, which transforms the random variable into a function of deterministic ones perturbed by a fixed noise source, yielding much lower variance gradient estimates. For discrete stochastic variables, REINFORCE is augmented with *control variates* for variance reduction. A number of control variate schemes have been proposed: NVIL (Mnih & Gregor, 2014) subtracts two baselines (one constant and one input-dependent) from the objective to reduce variance. MuProp (Gu et al., 2016) uses the first-order

Taylor approximation of the function as a baseline. REBAR (Tucker et al., 2017) uses the Gumbel-Softmax trick to form a control variate for unbiased gradient estimates. RELAX (Grathwohl et al.) generalizes REBAR to include an auxiliary network in the gradient expression and uses continuous relaxations and the reparameterization trick to give unbiased gradients.

Regarding biased estimators, a simple choice is the Straight-Through estimator (Bengio et al., 2013) which uses the gradient relative to the sample as that relative to the probability parameter. Another recent approach is to use continuous relaxations of discrete random variables so that the reparameterization trick becomes applicable. The most common example of this being the Gumbel-Softmax estimator (Maddison et al., 2017; Jang et al., 2017). Although this is a continuous relaxation, it has been used to define the Gumbel Straight-Through estimator with hard samples. This uses $\arg\max$ in the forward pass and the Gumbel-Softmax gradient is used as an approximation during in the backward pass. DARN (Gregor et al., 2014), like MuProp, also uses the first-order Taylor expansion as a baseline but does not add the analytical expectation, making the estimator biased for non-quadratic functions.

The Fourier expansion itself is widely used in computational learning theory with applications to learning low-degree functions (Kushilevitz & Mansour, 1993), decision trees (Mansour, 1994), constant-depth circuits (Linial et al., 1993) and juntas (Mossel et al., 2003). To the best of our knowledge we are the first to explore Fourier expansions for lowering the bias and variance in stochastic gradient estimators.

6. Experiments

Experimental Setup. We first validate FouST on a toy setup with a known analytic expression for $f(z)$. Next, we validate FouST by training generative models using the variational autoencoder framework of Kingma & Welling (2014). We optimize the single sample variational lower bound (ELBO) of the log-likelihood. We train variational autoencoders *exclusively* with Boolean latent variables on OMNIGLOT, CIFAR10, mini-ImageNet (Vinyals et al., 2016) and MNIST (see the appendix). We compare against Straight-Through, Gumbel-Softmax, and DARN. The results were consistent over multiple runs. Details regarding architectures and hyperparameters are in the appendix.

6.1. Biased Estimators on a Toy Problem

To validate the excessive bias in the Straight-Through estimator, as well as to explore the benefits of FouST, we first explore a toy problem. Similar to Tucker et al. (2017), we minimize $\mathbb{E}_{p(z)}[(z-t)^2]$, where $t \in (0,1)$ is a continuous target value and z is a sample from the Bernoulli distribution $p(z)$. The optimum is obtained for $p(z=+1) \in \{0,1\}$. Figure 1, shows a case with $t = 0.45$, where the minimizing solution is $p(z=+1) = 0$. Unlike the Straight-Through, FouST converges to the minimizing deterministic solution.

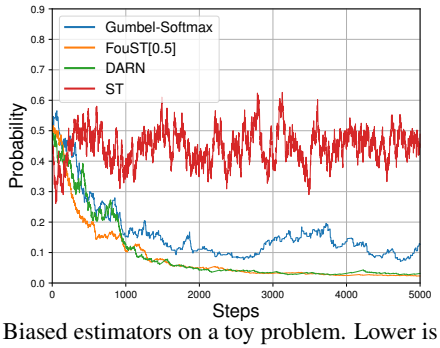

Figure 1. Biased estimators on a toy problem. Lower is better.

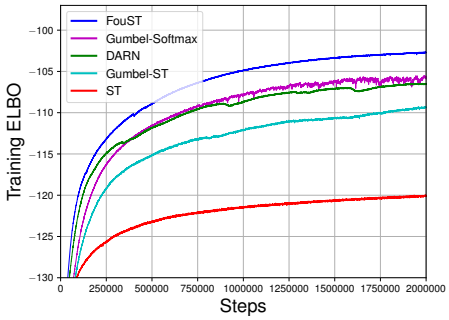

Figure 2. Training ELBO for two stochastic layer nonlinear models on OMNIGLOT

6.2. FouST vs. State-of-the-Art Gradient Estimators

Training Stochastic MLPs. We train MLPs with one and two stochastic layers on OMNIGLOT with the non-linear architecture of Tucker et al. (2017). Each stochastic Boolean layer is preceded by two deterministic layers of 200 $\tanh$ units. All hyperparameters remain fixed throughout the training. All estimators in this section use *one sample per example and a single decoder evaluation*.

We present results in figure 2. FouST outperforms other biased gradient estimators in both datasets and architectures. FouST is clearly better than the Straight-Through estimator. Despite the complicated nature of the optimized neural network function $f(z)$ the bias reduction appears fruitful.

With one or two stochastic layers we can also use the unbiased REBAR. REBAR is not directly comparable to the estimators we study, since it uses multiple decoder evaluations and for models with multiple stochastic layers, multiple passes through later layers. Nevertheless, for MNIST with two stochastic layers REBAR reaches a worse test ELBO of -94.43 v. -91.94 for FouST (see appendix for test ELBO and training curves with REBAR). A possibility of worse test than training ELBOs for REBAR was also suggested in the original work (Tucker et al., 2017).

Table 2. Validation bits per dimension for various models on CIFAR-10. ($*$) Discrete VAE++ uses a sophisticated RBM-based prior to obtain these results.

MODEL	BPD
VIMCO (VAN DEN OORD ET AL., 2017)	5.14
DISCRETE VAE++* (VAHDAT ET AL., 2018)	3.59
VQ-VAE (VAN DEN OORD ET AL., 2017)	4.67
FOUST (L=4)	4.16
FOUST+NOISEOP (L=1)	4.02

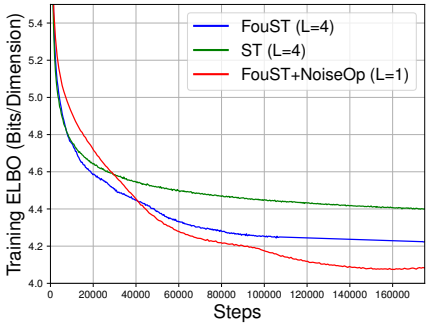

Figure 3. ELBO comparison on CIFAR-10 with stochastic ResNet.

Training Stochastic ResNets. We further validate FouST in a setting where the encoder and decoder are stochastic ResNets (S-ResNets), that is standard ResNets with stochastic layers inserted between ResNet blocks. Similar to MLPs, FouST outperforms other biased estimators also on CIFAR-10 (Figure 3). Note that despite the hyperparameter sweep, we were unable to train S-ResNet's with Gumbel-Softmax, so we compare against Straight-Through. With an S-ResNet with 12 ResNet blocks and 4 stochastic layers FouST yields a validation score of 4.16 bits per dimension (bpd). We used the discretized logistic mixture output model (Salimans et al., 2017). A comparison with other reported numbers is presented in table 2. We repeat the same experiment on mini-ImageNet with deeper hierarchical models with up to 10 stochastic layers but with a simpler Gaussian output model. We arrive at the same conclusions (see appendix).

Efficiency. We compare the efficiency of different estimators in the appendix. Like other biased estimators, FouST requires a single sample for estimating the gradients and has similar wallclock times. On MNIST, the unbiased REBAR is 15x and 40x slower than the biased estimators for two and five stochastic layer MLP's respectively. From the above experiments we conclude that FouST allows for efficient and effective training of fully connected and convolutional neural networks with Boolean stochastic variables.

6.3. FouST with Fourier Noise Operator

We test FouST with a reduced variance VAE decoder function obtained by applying the Fourier noise operator, $T_\rho[f](z)$. To extend the procedure to hierarchical networks we apply the noise operator in the lowest stochastic layer.

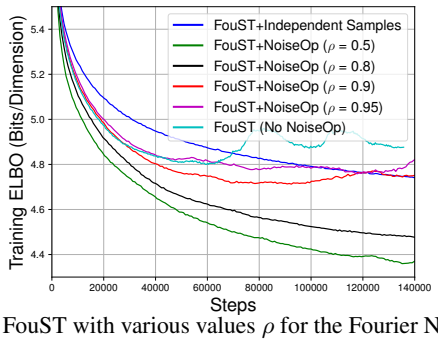

Figure 4. FouST with various values ρ for the Fourier Noise Operator versus independent samples on CIFAR-10.

We test whether applying the Fourier noise operator gives an advantage over multiple independent samples. In each case we used 5 samples to estimate $T_\rho[f](z)$. We a use single 64 channel-wide stochastic layer with an S-ResNet VAE on CIFAR-10 but with increased deterministic depth. The training ELBO curves are in figure 4, which show a significant improvement when correlated samples with $\rho = 0.5$ are used as opposed to independent samples.

A comparison with the same architecture trained without the Fourier noise operator shows a significant worsening of training stability, which reinforces our assumption that training instability is due to a combination of bias and variance. As evident from figure 4, training stability improves as ρ is decreased. Nevertheless, correlated samples lead to better optimization over independent ones.

Next we trained stochastic ResNets on CIFAR-10 with a wider 256 channel stochastic layer using the reduced variance version. We improve upon the deeper 4 layer version described in the previous section, achieving a validation score of 4.02 bits per dimension, as shown in figure 3.

7. Conclusion

In this work we introduce the framework of harmonic analysis for Boolean functions. We use harmonic analysis to derive the source of bias in the Straight-Through estimator for Boolean random variables. Based on the analysis we propose the FouST gradient estimate algorithm to train neural networks with Boolean random variables in hierarchical stochastic layers. FouST outperforms state-of-the-art biased gradient estimators, while maintaining efficiency that is orders of magnitude higher than unbiased estimators. Importantly, to the best of our knowledge FouST is the first gradient estimate algorithm, biased or unbiased, to be able to train very deep and wide neural networks with Boolean random variables. We conclude that harmonic analysis for Boolean function is a useful methodological tool for hierarchical Boolean stochastic neural networks and FouST is a practical gradient estimate algorithm to train them.

References

Bengio, Y., Léonard, N., and Courville, A. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

Dayan, P., Hinton, G. E., Neal, R. M., and Zemel, R. S. The helmholtz machine. *Neural computation*, 7(5):889–904, 1995.

Grathwohl, W., Choi, D., Wu, Y., Roeder, G., and Duvenaud, D. Backpropagation through the void: Optimizing control variates for black-box gradient estimation. In *ICLR 2018 : International Conference on Learning Representations 2018*.

Gregor, K., Danihelka, I., Mnih, A., Blundell, C., and Wierstra, D. Deep AutoRegressive networks. In *ICML*, 2014.

Gu, S., Levine, S., Sutskever, I., and Mnih, A. MuProp: Unbiased backpropagation for stochastic neural networks. In *ICLR (Poster)*, 2016.

Higgins, I., Matthey, L., Pal, A., Burgess, C., Glorot, X., Botvinick, M., Mohamed, S., and Lerchner, A. beta-vae: Learning basic visual concepts with a constrained variational framework. *ICLR*, 2(5):6, 2017.

Jang, E., Gu, S., and Poole, B. Categorical reparameterization with gumbel-softmax. In *ICLR 2017 : International Conference on Learning Representations 2017*, 2017.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. In *ICLR 2014 : International Conference on Learning Representations (ICLR) 2014*, 2014.

Kushilevitz, E. and Mansour, Y. Learning decision trees using the fourier spectrum. *SIAM Journal on Computing*, 22(6):1331–1348, 1993.

Linial, N., Mansour, Y., and Nisan, N. Constant depth circuits, fourier transform, and learnability. *Journal of the ACM (JACM)*, 40(3):607–620, 1993.

Maddison, C. J., Mnih, A., and Teh, Y. W. The concrete distribution: A continuous relaxation of discrete random variables. In *ICLR 2017 : International Conference on Learning Representations 2017*, 2017.

Mansour, Y. Learning boolean functions via the fourier transform. In *Theoretical advances in neural computation and learning*, pp. 391–424. Springer, 1994.

Mnih, A. and Gregor, K. Neural variational inference and learning in belief networks. In *Proceedings of The 31st International Conference on Machine Learning*, pp. 1791–1799, 2014.

Mossel, E., O'Donnell, R., and Servedio, R. P. Learning juntas. In *Proceedings of the thirty-fifth annual ACM symposium on Theory of computing*, pp. 206–212. ACM, 2003.

Neal, R. M. Connectionist learning of belief networks. *Artificial intelligence*, 56(1):71–113, 1992.

O'Donnell, R. *Analysis of boolean functions*. Cambridge University Press, 2014.

Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of The 31st International Conference on Machine Learning*, pp. 1278–1286, 2014.

Salimans, T., Karpathy, A., Chen, X., and Kingma, D. P. Pixelcnn++: Improving the pixelcnn with discretized logistic mixture likelihood and other modifications. In *ICLR 2017 : International Conference on Learning Representations 2017*, 2017.

Tucker, G., Mnih, A., Maddison, C. J., Lawson, J., and Sohl-Dickstein, J. Rebar: Low-variance, unbiased gradient estimates for discrete latent variable models. In *Advances in Neural Information Processing Systems*, pp. 2627–2636, 2017.

Vahdat, A., Macready, W., Bian, Z., and Khoshaman, A. Dvae++: Discrete variational autoencoders with overlapping transformations. In *ICML 2018: Thirty-fifth International Conference on Machine Learning*, pp. 5035–5044, 2018.

van den Oord, A., Vinyals, O., et al. Neural discrete representation learning. In *Advances in Neural Information Processing Systems*, pp. 6306–6315, 2017.

Vinyals, O., Blundell, C., Lillicrap, T., Wierstra, D., et al. Matching networks for one shot learning. In *Advances in neural information processing systems*, pp. 3630–3638, 2016.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.