ESTIMATING GRADIENTS FOR DISCRETE RANDOM VARIABLES BY SAMPLING WITHOUT REPLACEMENT

Anonymous authors
Paper under double-blind review

ABSTRACT

We derive an unbiased estimator for expectations over discrete random variables based on sampling *without replacement*, which reduces variance as it avoids duplicate samples. We show that our estimator can be derived as the Rao-Blackwellization of three different estimators. Combining our estimator with REINFORCE, we obtain a policy gradient estimator and we reduce its variance using a built-in control variate which is obtained without additional model evaluations. The resulting estimator is closely related to other gradient estimators. Experiments with a toy problem, a categorical Variational Auto-Encoder and a structured prediction problem show that our estimator is the only estimator that is consistently among the best estimators in both high and low entropy settings.

1 INTRODUCTION

Put replacement in your basement! We derive the *unordered set estimator*: an unbiased (gradient) estimator for expectations over discrete random variables based on (unordered sets of) samples *without replacement*. In particular, we consider the problem of estimating the expectation of $f(x)$ where x has a categorical distribution p over the domain D, i.e.

$$\mathbb{E}_{x\sim p(x)}[f(x)] = \sum_{x\in D} p(x)f(x). \tag{1}$$

This problem is relevant for reinforcement learning, discrete latent variable modelling (e.g. for compression), structured prediction (e.g. for translation), hard attention and many other tasks that use models with discrete operations in their computational graphs (see e.g. Jang et al. (2016)). If f is deterministic, then sampling without replacement reduces variance by avoiding duplicate samples.

Related work. Many algorithms for estimating gradients for discrete distributions have been proposed. A general and widely used estimator is REINFORCE (Williams, 1992). Biased gradients based on a continuous relaxations of the discrete distribution (known as Gumbel-Softmax or Concrete) were jointly introduced by Jang et al. (2016) and Maddison et al. (2016). These can be combined with the straight through estimator (Bengio et al., 2013) if the model requires discrete samples or be used to construct control variates for REINFORCE, as in REBAR (Tucker et al., 2017) or RELAX (Grathwohl et al., 2018). Many other methods use control variates and other techniques to reduce the variance of REINFORCE (Paisley et al., 2012; Ranganath et al., 2014; Gregor et al., 2014; Mnih & Gregor, 2014; Gu et al., 2016; Mnih & Rezende, 2016).

Some works rely on explicit summation of the expectation, either for the marginal distribution (Titsias & Lázaro-Gredilla, 2015) or globally summing some categories while sampling from the remainder (Liang et al., 2018; Liu et al., 2019). Other approaches use a finite difference approximation to the gradient (Lorberbom et al., 2018; 2019). Yin et al. (2019) introduced ARSM, which uses multiple model evaluations where the number adapts automatically to the uncertainty.

In the structured prediction setting, there are many algorithms for optimizing a quantity under a sequence of discrete decisions, using (weak) supervision, multiple samples (or deterministic model evaluations), or a combination both (Ranzato et al., 2016; Shen et al., 2016; He et al., 2016; Norouzi et al., 2016; Bahdanau et al., 2017; Edunov et al., 2018; Leblond et al., 2018; Negrinho et al., 2018). Most of these algorithms are biased and rely on pretraining using maximum likelihood or gradually transitioning from supervised to reinforcement learning. Using Gumbel-Softmax based approaches in a sequential setting is difficult as the bias accumulates because of mixing errors (Gu et al., 2018).

2 PRELIMINARIES

Throughout this paper, we will denote with B^k an *ordered* sample without replacement of size k and with S^k an *unordered* sample (of size k) from the categorical distribution p.

Restricted distribution. When sampling without replacement, we remove the set $C \subset D$ already sampled from the domain and we denote with $p^{D \setminus C}$ the distribution *restricted to the domain* $D \setminus C$:

$$p^{D \setminus C}(x) = \frac{p(x)}{1 - \sum_{c \in C} p(c)}, \quad x \in D \setminus C. \tag{2}$$

Ordered sample without replacement B^k. Let $B^k = (b_1, ..., b_k), b_i \in D$ be an *ordered sample without replacement*, which is generated from the distribution p as follows: first, sample $b_1 \sim p$, then sample $b_2 \sim p^{D \setminus \{b_1\}}$, $b_3 \sim p^{D \setminus \{b_1, b_2\}}$, etc. i.e. elements are sampled one by one without replacement. Using this procedure, B^k can be seen as a (partial) ranking according to the Plackett-Luce model (Plackett, 1975; Luce, 1959) and the probability of obtaining the vector B^k is

$$p(B^k) = \prod_{i=1}^{k} p^{D \setminus B^{i-1}}(b_i) = \prod_{i=1}^{k} \frac{p(b_i)}{1 - \sum_{j<i} p(b_j)}. \tag{3}$$

We can also restrict B^k to the domain $D \setminus C$, which means that $b_i \notin C$ for $i = 1, ..., k$:

$$p^{D \setminus C}(B^k) = \prod_{i=1}^{k} \frac{p^{D \setminus C}(b_i)}{1 - \sum_{j<i} p^{D \setminus C}(b_j)} = \prod_{i=1}^{k} \frac{p(b_i)}{1 - \sum_{c \in C} p(c) - \sum_{j<i} p(b_j)}. \tag{4}$$

Unordered sample without replacement. Let $S^k \subseteq D$ be an *unordered* sample without replacement from the distribution p, which can be generated simply by generating an ordered sample and discarding the order. We denote elements in the sample with $s \in S^k$ (so without index) and we write $\mathcal{B}(S^k)$ as the set of all $k!$ permutations (orderings) B^k that correspond to (could have generated) S^k. It follows that the probability for sampling S^k is given by:

$$p(S^k) = \sum_{B^k \in \mathcal{B}(S^k)} p(B^k) = \sum_{B^k \in \mathcal{B}(S^k)} \prod_{i=1}^{k} \frac{p(b_i)}{1 - \sum_{j<i} p(b_j)} = \left(\prod_{s \in S^k} p(s) \right) \cdot \sum_{B^k \in \mathcal{B}(S^k)} \prod_{i=1}^{k} \frac{1}{1 - \sum_{j<i} p(b_j)}. \tag{5}$$

The last step follows since $B^k \in \mathcal{B}(S^k)$ is an ordering of S^k, such that $\prod_{i=1}^{k} p(b_i) = \prod_{s \in S} p(s)$. Naive computation of $p(S^k)$ is $O(k!)$, but in Appendix B we show how to compute it efficiently.

When sampling from the distribution restricted to $D \setminus C$, we sample $S^k \subseteq D \setminus C$ with probability:

$$p^{D \setminus C}(S^k) = \left(\prod_{s \in S^k} p(s) \right) \cdot \sum_{B^k \in \mathcal{B}(S^k)} \prod_{i=1}^{k} \frac{1}{1 - \sum_{c \in C} p(c) - \sum_{j<i} p(b_j)}. \tag{6}$$

The Gumbel-Top-k trick. As an alternative to sequential sampling, we can also sample B^k and S^k by taking the top k of Gumbel variables (Yellott, 1977; Vieira, 2014; Kim et al., 2016). Following notation from Kool et al. (2019b), we define the *perturbed log-probability* $g_{\phi_i} = \phi_i + g_i$, where $\phi_i = \log p(i)$ and $g_i \sim \text{Gumbel}(0)$. Then let $b_1 = \arg\max_{i \in D} g_{\phi_i}$, $b_2 = \arg\max_{i \in D \setminus \{b_1\}} g_{\phi_i}$, etc., so B^k is the top k of the perturbed log-probabilities *in decreasing order*. The probability of obtaining B_k using this procedure is given by equation 3, so this provides an alternative sampling method which is effectively a (non-differentiable) reparameterization of sampling without replacement. For a differentiable reparameterization, see Grover et al. (2019).

It follows that taking the top k perturbed log-probabilities *without order*, we obtain the unordered sample set S^k. This way of sampling underlies the efficient computation of $p(S^k)$ in Appendix B.

3 METHODOLOGY

In this section, we derive the *unordered set policy gradient estimator*: a low-variance, unbiased estimator of $\nabla_{\boldsymbol{\theta}} \mathbb{E}_{p_{\boldsymbol{\theta}}(x)}[f(x)]$ based on an unordered sample without replacement S^k. First, we derive the generic (non-gradient) estimator for $\mathbb{E}[f(x)]$ as the Rao-Blackwellized version of a single sample Monte Carlo estimator (and two other estimators!). Then we combine this estimator with REINFORCE (Williams, 1992) and we show how to reduce its variance using a built-in baseline.

3.1 RAO-BLACKWELLIZATION OF THE SINGLE SAMPLE ESTIMATOR

A very crude but simple estimator for $\mathbb{E}[f(x)]$ based on the *ordered* sample B^k is to *only* use the first element b_1, which by definition is a sample from the distribution p. We define this estimator as the *single sample estimator*, which is unbiased, since

$$\mathbb{E}_{B^k \sim p(B^k)}[f(b_1)] = \mathbb{E}_{b_1 \sim p(b_1)}[f(b_1)] = \mathbb{E}_{x \sim p(x)}[f(x)]. \tag{7}$$

Discarding all but one sample, the single sample estimator is inefficient, but we can use Rao-Blackwellization (Casella & Robert, 1996) to signficantly improve it. To this end, we consider the distribution $B^k|S^k$, which is, knowing the unordered sample S^k, the conditional distribution over ordered samples $B^k \in \mathcal{B}(S^k)$ that could have generated S^k.[1] Using $B^k|S^k$, we rewrite $\mathbb{E}[f(b_1)]$ as

$$\mathbb{E}_{B^k \sim p(B^k)}[f(b_1)] = \mathbb{E}_{S^k \sim p(S^k)}\left[\mathbb{E}_{B^k \sim p(B^k|S^k)}\left[f(b_1)\right]\right] = \mathbb{E}_{S^k \sim p(S^k)}\left[\mathbb{E}_{b_1 \sim p(b_1|S^k)}\left[f(b_1)\right]\right].$$

The Rao-Blackwellized version of the single sample estimator computes the inner conditional expectation exactly. Since B^k is an ordering of S^k, we have $b_1 \in S^k$ and we can compute this as

$$\mathbb{E}_{b_1 \sim p(b_1|S^k)}\left[f(b_1)\right] = \sum_{s \in S^k} P(b_1 = s|S^k) f(s) \tag{8}$$

where, in a slight abuse of notation, $P(b_1 = s|S^k)$ is the probability that the first sampled element b_1 takes the value s, given that the complete set of k samples is S^k. Using Bayes' Theorem we find

$$P(b_1 = s|S^k) = \frac{p(S^k|b_1 = s)P(b_1 = s)}{p(S^k)} = \frac{p^{D\setminus\{s\}}(S^k \setminus \{s\})p(s)}{p(S^k)}. \tag{9}$$

The step $p(S^k|b_1 = s) = p^{D\setminus\{s\}}(S^k \setminus \{s\})$ comes from analyzing sequential sampling without replacement: given that the first element sampled is s, the remaining elements have a distribution restricted to $D \setminus \{s\}$, so sampling S^k (including s) given the first element s is equivalent to sampling the remainder $S^k \setminus \{s\}$ from the restricted distribution, which has probability $p^{D\setminus\{s\}}(S^k \setminus \{s\})$ (see equation 6).

The unordered set estimator. For notational convenience, we introduce the *leave-one-out ratio*.

Definition 1. The *leave-one-out ratio* of s w.r.t. the set S is given by $R(S^k, s) = \frac{p^{D\setminus\{s\}}(S^k \setminus \{s\})}{p(S^k)}$.

Rewriting equation 9 as $P(b_1 = s|S^k) = p(s)R(S^k, s)$ shows that the probability of sampling s first, given S^k, is simply the unconditional probability multiplied by the leave-one-out ratio. We now define the unordered set estimator as the Rao-Blackwellized version of the single-sample estimator.

Theorem 1. *The* unordered set estimator*, given by*

$$e^{US}(S^k) = \sum_{s \in S^k} p(s)R(S^k, s)f(s) \tag{10}$$

is the Rao-Blackwellized version of the (unbiased!) single sample estimator.

Proof. Using $P(b_1 = s|S^k) = p(s)R(S^k, s)$ in equation 8 we have

$$\mathbb{E}_{b_1 \sim p(b_1|S^k)}\left[f(b_1)\right] = \sum_{s \in S^k} P(b_1 = s|S^k) f(s) = \sum_{s \in S^k} p(s)R(S^k, s)f(s). \tag{11}$$

□

As a result of Theorem 1, the unordered set estimator is unbiased and has variance equal or lower than the single sample estimator by the Rao-Blackwell Theorem (Lehmann & Scheffé, 1950).

[1] Note that $B^k|S^k$ is *not* a Plackett-Luce distribution restricted to S^k!

3.2 RAO-BLACKWELLIZATION OF OTHER ESTIMATORS

The unordered set estimator is also the result of Rao-Blackwellizing two other unbiased estimators: the *stochastic sum-and-sample* estimator and the *importance-weighted estimator*.

The sum-and-sample estimator. We define as *sum-and-sample estimator* any estimator that relies on the identity that for any $C \subset D$

$$\mathbb{E}_{x\sim p(x)}[f(x)] = \mathbb{E}_{x\sim p^{D\setminus C}(x)}\left[\sum_{c\in C} p(c)f(c) + \left(1 - \sum_{c\in C} p(c)\right) f(x)\right]. \tag{12}$$

For the derivation, see Appendix C.1 or Liang et al. (2018); Liu et al. (2019). In general, a sum-and-sample estimator sums expectation terms for a number of categories explicitly (for example selected by their value f (Liang et al., 2018) or probability p (Liu et al., 2019)), and uses a (down-weighted) sample from $D \setminus C$ to estimate the remaining terms. As equation 12 holds for any C, we choose $C = B^{k-1}$ stochastically to define the *stochastic sum-and-sample* estimator:

$$e^{\text{SSAS}}(B^k) = \sum_{j=1}^{k-1} p(b_j)f(b_j) + \left(1 - \sum_{j=1}^{k-1} p(b_j)\right) f(b_k). \tag{13}$$

Sampling without replacement, it holds that $b_k|B^{k-1} \sim p^{D\setminus B^{k-1}}$, so the unbiasedness follows from equation 12 by separating the expectation over B^k into expectations over B^{k-1} and $b_k|B^{k-1}$:

$$\mathbb{E}_{B^{k-1}\sim p(B^{k-1})}\left[\mathbb{E}_{b_k\sim p(b_k|B^{k-1})}\left[e^{\text{SSAS}}(B^k)\right]\right] = \mathbb{E}_{B^{k-1}\sim p(B^{k-1})}\left[\mathbb{E}[f(x)]\right] = \mathbb{E}[f(x)].$$

In general, a sum-and-sample estimator reduces variance if the probability mass is concentrated on the summed categories. As typically high probability categories are sampled first, the stochastic sum-and-sample estimator sums high probability categories, similar to the estimator by Liu et al. (2019) which we refer to as the *deterministic sum-and-sample estimator*. As we show in Appendix C.2, Rao-Blackwellizing the stochastic sum-and-sample estimator also results in the unordered set estimator. This means that the unordered set estimator has equal or lower variance.

The importance-weighted estimator. The importance-weighted estimator (Vieira, 2017) is

$$e^{\text{IW}}(S^k, \kappa) = \sum_{s\in S^k} \frac{p(s)}{q(s,\kappa)} f(s). \tag{14}$$

This estimator does *not* use the order of the sample, but assumes sampling using the Gumbel-Top-k trick and requires access to κ, the $(k+1)$-th largest perturbed log-probability, which can be seen as the 'threshold' since $g_{\phi_s} > \kappa \ \forall s \in S^k$. $q(s,a) = P(g_{\phi_s} > a)$ can be interpreted as the *inclusion probability* of $s \in S^k$ (assuming a fixed threshold a instead of a fixed sample size k). For details and a proof of unbiasedness, see Vieira (2017) or Kool et al. (2019b). As the estimator has high variance, Kool et al. (2019b) resort to *normalizing* the importance weights, resulting in biased estimates. Instead, we use Rao-Blackwellization to eliminate stochasticity by κ. Again, the result is the unordered set estimator (see Appendix D.1), which thus has equal or lower variance.

3.3 THE UNORDERED SET POLICY GRADIENT ESTIMATOR

Writing $p_{\boldsymbol{\theta}}$ to indicate the dependency on the model parameters $\boldsymbol{\theta}$, we can combine the unordered set estimator with REINFORCE (Williams, 1992) to obtain the *unordered set policy gradient* estimator.

Corollary 1. *The* unordered set policy gradient estimator*, given by*

$$e^{USPG}(S^k) = \sum_{s\in S^k} p_{\boldsymbol{\theta}}(s)R(S^k, s)\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(s)f(s) = \sum_{s\in S^k} \nabla_{\boldsymbol{\theta}} p_{\boldsymbol{\theta}}(s)R(S^k, s)f(s), \tag{15}$$

is an unbiased estimate of the policy gradient.

Proof. Using REINFORCE (Williams, 1992) combined with the unordered set estimator we find:

$$\nabla_{\boldsymbol{\theta}}\mathbb{E}_{p_{\boldsymbol{\theta}}(x)}[f(x)] = \mathbb{E}_{p_{\boldsymbol{\theta}}(x)}[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(x)f(x)] = \mathbb{E}_{S^k\sim p_{\boldsymbol{\theta}}(S^k)}\left[\sum_{s\in S^k} p_{\boldsymbol{\theta}}(s)R(S^k, s)\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(s)f(s)\right].$$

□

Variance reduction using a built-in control variate. The variance of REINFORCE can be reduced by subtracting a baseline from f. When taking multiple samples (with replacement), a simple and effective baseline is to take the mean of other (independent!) samples (Mnih & Rezende, 2016). Sampling without replacement, we can use the same idea to construct a baseline based on the other samples, but we have to correct for the fact that the samples are *not* independent.

Theorem 2. *The* unordered set policy gradient estimator with baseline*, given by*

$$e^{\textit{USPGBL}}(S^k) = \sum_{s \in S^k} \nabla_{\boldsymbol{\theta}} p_{\boldsymbol{\theta}}(s) R(S^k, s) \left(f(s) - \sum_{s' \in S^k} p_{\boldsymbol{\theta}}(s') R^{D \setminus \{s\}}(S^k, s') f(s') \right), \tag{16}$$

where

$$R^{D \setminus \{s\}}(S^k, s') = \frac{p_{\boldsymbol{\theta}}^{D \setminus \{s, s'\}}(S^k \setminus \{s, s'\})}{p_{\boldsymbol{\theta}}^{D \setminus \{s\}}(S^k \setminus \{s\})} \tag{17}$$

is the second order leave-one-out ratio*, is an unbiased estimate of the policy gradient.*

Proof. See Appendix E.1. □

Including the pathwise derivative. So far, we have only considered the scenario where f does not depend on $\boldsymbol{\theta}$. If f does depend on $\boldsymbol{\theta}$, for example in a VAE (Kingma & Welling, 2014; Rezende et al., 2014), then we use the notation $f_{\boldsymbol{\theta}}$ and we can write the gradient (Schulman et al., 2015) as

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{x \sim p_{\boldsymbol{\theta}}}[f_{\boldsymbol{\theta}}(x)] = \mathbb{E}_{x \sim p_{\boldsymbol{\theta}}}[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(x) f_{\boldsymbol{\theta}}(x) + \nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}}(x)]. \tag{18}$$

The additional second ('pathwise') term can be estimated (using the same samples) with the standard unordered set estimator. This results in the *full* unordered set policy gradient estimator:

$$\begin{aligned} e^{\text{FUSPG}}(S^k) &= \sum_{s \in S^k} \nabla_{\boldsymbol{\theta}} p_{\boldsymbol{\theta}}(s) R(S^k, s) f_{\boldsymbol{\theta}}(s) + \sum_{s \in S^k} p_{\boldsymbol{\theta}}(s) R(S^k, s) \nabla_{\boldsymbol{\theta}} f_{\boldsymbol{\theta}}(s) \\ &= \sum_{s \in S^k} R(S^k, s) \nabla_{\boldsymbol{\theta}} \left(p_{\boldsymbol{\theta}}(s) f_{\boldsymbol{\theta}}(s) \right) \end{aligned} \tag{19}$$

Equation 19 is straightforward to implement using an automatic differentiation library. We can also include the baseline (as in equation 16) but we must make sure to call STOP_GRADIENT (DETACH in PyTorch) on the baseline (but not on $f_{\boldsymbol{\theta}}(s)$!). Importantly, we should *never* track gradients through the leave-one-out ratio $R(S^k, s)$ which means it can be efficiently computed in pure inference mode.

Scope & limitations. We can use the unordered set estimator for any discrete distribution from which we can sample without replacement, by treating it as a univariate categorical distribution over its domain. This includes sequence models, from which we can sample using Stochastic Beam Search (Kool et al., 2019b), as well as multivariate categorical distributions which can also be treated as sequence models (see Section 4.2). In the presence of continuous variables or a stochastic function f, we may separate this stochasticity from the stochasticity over the discrete distribution, as in Lorberbom et al. (2019). The computation of the leave-one-out ratios adds some overhead, although they can be computed efficiently, even for large k (see Appendix B). For a moderately sized model, the costs of model evaluation and backpropagation dominate the cost of computing the estimator.

3.4 Relation to other multi-sample estimators

Relation to the empirical risk estimator. The empirical risk loss (Edunov et al., 2018) estimates the expectation in equation 1 by summing only a subset S of the domain, using *normalized* probabilities $\hat{p}_{\boldsymbol{\theta}}(s) = \frac{p_{\boldsymbol{\theta}}(s)}{\sum_{s' \in S} p_{\boldsymbol{\theta}}(s)}$. Using this loss, the (biased) estimate of the gradient is given by

$$e^{\text{RISK}}(S^k) = \sum_{s \in S^k} \nabla_{\boldsymbol{\theta}} \left(\frac{p_{\boldsymbol{\theta}}(s)}{\sum_{s' \in S^k} p_{\boldsymbol{\theta}}(s')} \right) f(s). \tag{20}$$

The risk estimator is similar to the unordered set policy gradient estimator, with two important differences: 1) the individual terms are normalized by the total probability mass rather than the leave-one-out ratio and 2) the gradient is computed through the normalization factor.

Theorem 3. *By taking the gradient w.r.t. the normalization factor into account, the risk estimator has a built-in baseline, which means it can be written as*

$$e^{RISK}(S^k) = \sum_{s\in S^k} \nabla_{\boldsymbol{\theta}} p_{\boldsymbol{\theta}}(s) \frac{1}{\sum_{s''\in S^k} p_{\boldsymbol{\theta}}(s'')} \left(f(s) - \sum_{s'\in S^k} p_{\boldsymbol{\theta}}(s') \frac{1}{\sum_{s''\in S^k} p_{\boldsymbol{\theta}}(s'')} f(s') \right). \quad (21)$$

Proof. See Appendix F.1 □

This theorem highlights the similarity between the biased risk estimator and our unbiased estimator (equation 16), and suggests that their only difference is the weighting of terms. Unfortunately, the implementation by Edunov et al. (2018) has more sources of bias (e.g. length normalization), which are not compatible with our estimator. However, we believe that our analysis helps analyze the bias of the risk estimator and is a step towards developing unbiased estimators for structured prediction.

Relation to VIMCO. VIMCO (Mnih & Rezende, 2016) is an unbiased estimator that uses multiple samples *with replacement* and has a built-in baseline based on the other $k-1$ samples. Denoting the samples (with replacement) with $X^k = (x_1, ..., x_k)$, VIMCO computes the gradient estimate as:

$$e^{\text{VIMCO}}(X^k) = \frac{1}{k} \sum_{i=1}^{k} \nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(x_i) \left(f(x_i) - \frac{1}{k-1} \sum_{j\neq i} f(x_j) \right). \quad (22)$$

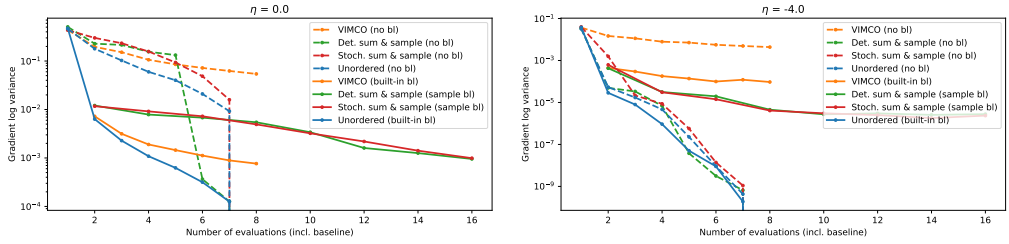
We think of our estimator as the without-replacement version of VIMCO, which weights terms by $p_{\boldsymbol{\theta}}(s)R(S^k, s)$ instead of $\frac{1}{k}$. This puts more weight on higher probability elements to compensate sampling without replacement. If probabilities are small and (close to) uniform, there are (almost) no duplicate samples and the weights will be close to $\frac{1}{k}$, so the gradient estimate is similar to VIMCO.

Relation to ARSM. The ARSM (Yin et al., 2019) estimator also uses multiple evaluations of $p_{\boldsymbol{\theta}}$ and f. It determines a number of 'pseudo-samples', from which duplicates should be removed for efficient implementation. This can be seen as similar to sampling without replacement, and the estimator also has a built-in control variate. Compared to ARSM, our estimator allows direct control over the computational cost (through the sample size k) and has wider applicability, for example it also applies to multivariate categorical variables with different numbers of categories per dimension.

4 Experiments

4.1 Bernoulli toy experiment

We use the code by Liu et al. (2019) to reproduce their Bernoulli toy experiment. Given a vector $\mathbf{p} = (0.6, 0.51, 0.48)$ the goal is to minimize the loss $\mathcal{L}(\eta) = \mathbb{E}_{x_1,x_2,x_3\sim\text{Bern}(\sigma(\eta))}\left[\sum_{i=1}^{3}(x_i - p_i)^2\right]$. Here x_1, x_2, x_3 are i.i.d. from the Bernoulli$(\sigma(\eta))$ distribution, parameterized by a scalar $\eta \in \mathbb{R}$, where $\sigma(\eta) = (1+\exp(-\eta))^{-1}$ is the sigmoid function. We compare different estimators, with and without baseline (either 'built-in' or using additional samples, referred to as REINFORCE+ in Liu et al. (2019)). We report the (log-)variance of the scalar gradient $\frac{\partial \mathcal{L}}{\partial \eta}$ as a function of the number of model evaluations, which is twice as high when using a sampled baseline (for each term).



(a) High entropy ($\eta = 0$) (b) Low entropy ($\eta = -4$)

Figure 1: Gradient log variance as a function of the number of model evaluations (including baseline evaluations). Note that for some estimators, the variance is 0 (log variance $-\infty$) for $k = 8$.

As can be seen in Figure 1, the unordered set estimator is the only estimator that has consistently the lowest (or comparable) variance in both the high ($\eta = 0$) and low entropy ($\eta = -4$) regimes and for different number of samples/model evaluations. This suggests that it combines the advantages of the other estimators. We also ran the actual optimization experiment, where with as few as $k = 3$ samples the trajectory was indistinguishable from using the exact gradient (see Liu et al. (2019)).

4.2 Categorical Variational Auto-Encoder

We use the code[2] from Yin et al. (2019) to train a *categorical* Variational Auto-Encoder (VAE) with 20 dimensional latent space, with 10 categories per dimension. To use our estimator, we consider the latent distribution as a single factorized distribution with 10^{20} categories from which we can sample without replacement using Stochastic Beam Search (Kool et al., 2019b), sequentially sampling each dimension as if it where a sequence model. We also perform experiments with 10^2 latent space, which provides a lower entropy setting, to highlight the advantage of our estimator.

Measuring the variance. The most direct way to compare unbiased gradient estimators is to compare their variance. We measure the variance of different estimators with $k = 4$ samples during training with VIMCO (Mnih & Rezende, 2016), such that all estimators are computed for the same model parameters. In Figure 2 we see that the unordered set estimator has the lowest variance in both the small domain (low entropy) and large domain (high entropy) setting, being on-par with the best of the (stochastic[3]) sum-and-sample estimator and VIMCO. This confirms the toy experiment in a real scenario, suggesting that the unordered set estimator provides the best of both estimators.

ELBO optimization. We made changes in the code which caused our results to differ from Yin et al. (2019) (see Appendix G.1). Additionally we compare against VIMCO and the stochastic sum-and-sample estimator. In Figure 3 we observe that our estimator performs on par with VIMCO and outperforms other estimators. There are a lot of other factors, e.g. exploration that may explain why we do not get a strictly better result despite the lower variance. The low -ELBO scores are the result of overfitting and binarization by a fixed threshold. In Appendix G.2 we report validation curves and results using the standard binarized MNIST dataset from Salakhutdinov & Murray (2008).

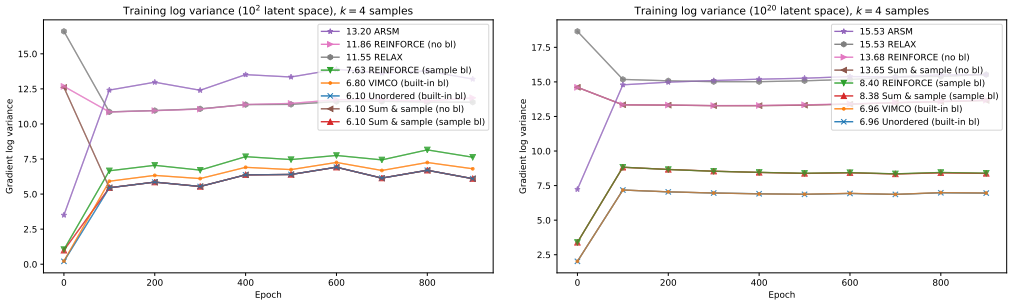
4.3 Structured Prediction for the Travelling Salesman Problem

To show the wide applicability of our estimator, we consider the structured prediction task of predicting routes (sequences) for the Travelling Salesman Problem (TSP) (Vinyals et al., 2015; Bello et al., 2016; Kool et al., 2019a). We use the code by Kool et al. (2019a)[4] to reproduce their TSP experiment with 20 nodes. We implement VIMCO (sampling with replacement) as well as the

[2] https://github.com/ARM-gradient/ARSM/tree/master/vae

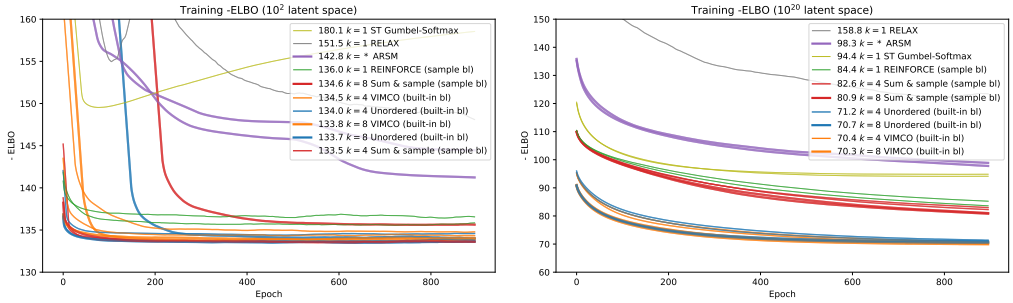
[3] We cannot use the deterministic version by Liu et al. (2019) since we cannot select the top k categories.

[4] https://github.com/wouterkool/attention-learn-to-route



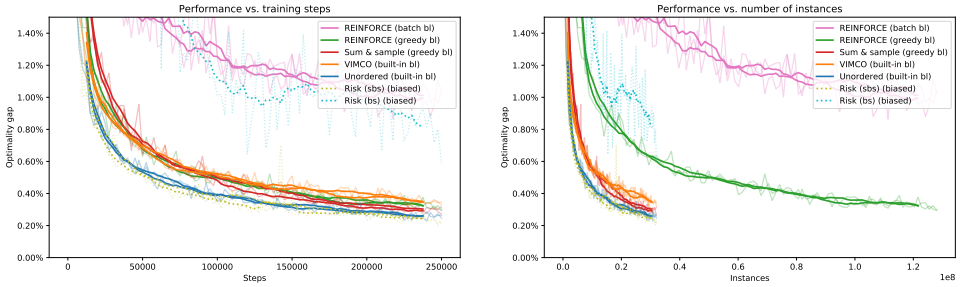
(a) Small domain (latent space size 10^2) (b) Large domain (latent space size 10^{20})

Figure 2: Gradient log variance of different unbiased estimators with $k = 4$ samples, estimated every 100 (out of 1000) epochs while training using VIMCO. Each estimator is computed 1000 times with different latent samples for a fixed minibatch (the first 100 records of training data). We report (the logarithm of) the sum of the variances per parameter (trace of the covariance matrix). Some lines coincide, so we sort the legend by the last measurement and report its value.



(a) Small domain (latent space size 10^2) (b) Large domain (latent space size 10^{20})

Figure 3: Smoothed training curves (-ELBO) of two independent runs when training with different estimators with $k = 1$, 4 or 8 (thicker lines) samples (ARSM has a variable number). Some lines coincide, so we sort the legend by the lowest -ELBO achieved and report this value.



(a) Performance vs. training steps (b) Performance vs. number of instances

Figure 4: TSP validation set optimality gap measured during training. Raw results are light, smoothed results are darker (2 random seeds per setting). We compare our estimator against VIMCO, the sum-and-sample estimator and the (biased!) risk estimator with $k = 4$ samples and against single-sample REINFORCE, with batch-average or greedy rollout baseline.

stochastic sum-and-sample estimator and our estimator, using Stochastic Beam Search (Kool et al., 2019b) for sampling. Additionally, we compare against REINFORCE with greedy rollout baseline (Rennie et al., 2017) used by Kool et al. (2019b) and a batch-average baseline. For reference, we also include the *biased* risk estimator, either 'sampling' using stochastic or deterministic beam search (as in Edunov et al. (2018)). In Figure 4a, we compare training progress (measured on the validation set) as a function of the number of training steps, where we divide the batch size by k to keep the total number of samples equal. Our estimator outperforms VIMCO, the stochastic sum-and-sample estimator and the strong greedy rollout baseline (which uses additional baseline model evaluations) and performs on-par with the biased risk estimator. In Figure 4b, we plot the same results against the number of instances, which shows that, compared to the single sample estimators, we can train with less data and less computational cost (as we only need to run the encoder once for each instance).

5 Discussion

We introduced the unordered set estimator, a low-variance, unbiased (gradient) estimator based on sampling without replacement. Our estimator is the result of Rao-Blackwellizing three existing estimators, which guarantees equal or lower variance, and is closely related to a number of other estimators. It has wide applicability, is parameter free (except for the sample size k) and has competitive performance to the best of alternatives in both high and low entropy regimes.

In our experiments, we found that VIMCO (Mnih & Rezende, 2016), closely related to our estimator, is a simple yet strong baseline which has performance similar to our estimator in the high entropy setting. We want to stress that many recent works on gradient estimators for discrete distributions have omitted this strong baseline, which may be often preferred given its simplicity. In future work, we want to investigate if we can apply our estimator to estimate gradients 'locally' (Titsias & Lázaro-Gredilla, 2015), as locally we have a smaller domain and expect more duplicate samples.

REFERENCES

Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. An actor-critic algorithm for sequence prediction. In *International Conference on Learning Representations*, 2017.

Irwan Bello, Hieu Pham, Quoc V Le, Mohammad Norouzi, and Samy Bengio. Neural combinatorial optimization with reinforcement learning. *arXiv preprint arXiv:1611.09940*, 2016.

Yoshua Bengio, Nicholas Léonard, and Aaron Courville. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*, 2013.

George Casella and Christian P Robert. Rao-Blackwellisation of sampling schemes. *Biometrika*, 83 (1):81–94, 1996.

Sergey Edunov, Myle Ott, Michael Auli, David Grangier, et al. Classical structured prediction losses for sequence to sequence learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pp. 355–364, 2018.

Will Grathwohl, Dami Choi, Yuhuai Wu, Geoffrey Roeder, and David Duvenaud. Backpropagation through the void: Optimizing control variates for black-box gradient estimation. In *International Conference on Learning Representations*, 2018.

Karol Gregor, Ivo Danihelka, Andriy Mnih, Charles Blundell, and Daan Wierstra. Deep autoregressive networks. In *International Conference on Machine Learning*, pp. 1242–1250, 2014.

Aditya Grover, Eric Wang, Aaron Zweig, and Stefano Ermon. Stochastic optimization of sorting networks via continuous relaxations. In *International Conference on Learning Representations*, 2019.

Jiatao Gu, Daniel Jiwoong Im, and Victor OK Li. Neural machine translation with Gumbel-greedy decoding. In *Thirty-Second AAAI Conference on Artificial Intelligence (AAAI)*, 2018.

Shixiang Gu, Sergey Levine, Ilya Sutskever, and Andriy Mnih. Muprop: Unbiased backpropagation for stochastic neural networks. In *International Conference on Learning Representations*, 2016.

Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. Dual learning for machine translation. In *Advances in Neural Information Processing Systems*, pp. 820–828, 2016.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2016.

Carolyn Kim, Ashish Sabharwal, and Stefano Ermon. Exact sampling with integer linear programs and random perturbations. In *Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

Diederik P Kingma and Max Welling. Auto-encoding variational Bayes. In *International Conference on Learning Representations*, 2014.

Wouter Kool, Herke van Hoof, and Max Welling. Attention, learn to solve routing problems! In *International Conference on Learning Representations*, 2019a.

Wouter Kool, Herke Van Hoof, and Max Welling. Stochastic beams and where to find them: The gumbel-top-k trick for sampling sequences without replacement. In *International Conference on Machine Learning*, pp. 3499–3508, 2019b.

Hugo Larochelle and Iain Murray. The neural autoregressive distribution estimator. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, pp. 29–37, 2011.

Rémi Leblond, Jean-Baptiste Alayrac, Anton Osokin, and Simon Lacoste-Julien. Searnn: Training RNNs with global-local losses. In *6th International Conference on Learning Representations*, 2018.

EL Lehmann and Henry Scheffé. Completeness, similar regions, and unbiased estimation: Part i. *Sankhyā: The Indian Journal of Statistics*, pp. 305–340, 1950.

Chen Liang, Mohammad Norouzi, Jonathan Berant, Quoc V Le, and Ni Lao. Memory augmented policy optimization for program synthesis and semantic parsing. In *Advances in Neural Information Processing Systems*, pp. 9994–10006, 2018.

Runjing Liu, Jeffrey Regier, Nilesh Tripuraneni, Michael Jordan, and Jon Mcauliffe. Rao-Blackwellized stochastic gradients for discrete distributions. In *International Conference on Machine Learning*, pp. 4023–4031, 2019.

Guy Lorberbom, Andreea Gane, Tommi Jaakkola, and Tamir Hazan. Direct optimization through argmax for discrete variational auto-encoder. *arXiv preprint arXiv:1806.02867*, 2018.

Guy Lorberbom, Chris J Maddison, Nicolas Heess, Tamir Hazan, and Daniel Tarlow. Direct policy gradients: Direct optimization of policies in discrete action spaces. *arXiv preprint arXiv:1906.06062*, 2019.

R Duncan Luce. *Individual choice behavior: A theoretical analysis*. John Wiley, 1959.

Chris J Maddison, Daniel Tarlow, and Tom Minka. A* sampling. In *Advances in Neural Information Processing Systems*, pp. 3086–3094, 2014.

Chris J Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2016.

Andriy Mnih and Karol Gregor. Neural variational inference and learning in belief networks. In *International Conference on Machine Learning*, pp. 1791–1799, 2014.

Andriy Mnih and Danilo Rezende. Variational inference for Monte Carlo objectives. In *International Conference on Machine Learning*, pp. 2188–2196, 2016.

Renato Negrinho, Matthew Gormley, and Geoffrey J Gordon. Learning beam search policies via imitation learning. In *Advances in Neural Information Processing Systems*, pp. 10673–10682, 2018.

Mohammad Norouzi, Samy Bengio, Navdeep Jaitly, Mike Schuster, Yonghui Wu, Dale Schuurmans, et al. Reward augmented maximum likelihood for neural structured prediction. In *Advances In Neural Information Processing Systems*, pp. 1723–1731, 2016.

John Paisley, David M Blei, and Michael I Jordan. Variational Bayesian inference with stochastic search. In *International Conference on Machine Learning*, pp. 1363–1370, 2012.

Robin L Plackett. The analysis of permutations. *Journal of the Royal Statistical Society: Series C (Applied Statistics)*, 24(2):193–202, 1975.

Rajesh Ranganath, Sean Gerrish, and David Blei. Black box variational inference. In *Artificial Intelligence and Statistics*, pp. 814–822, 2014.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. In *International Conference on Learning Representations*, 2016.

Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 7008–7024, 2017.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, pp. 1278–1286, 2014.

Ruslan Salakhutdinov and Iain Murray. On the quantitative analysis of deep belief networks. In *International Conference on Machine Learning*, pp. 872–879, 2008.

John Schulman, Nicolas Heess, Theophane Weber, and Pieter Abbeel. Gradient estimation using stochastic computation graphs. In *Advances in Neural Information Processing Systems*, pp. 3528–3536, 2015.

Shiqi Shen, Yong Cheng, Zhongjun He, Wei He, Hua Wu, Maosong Sun, and Yang Liu. Minimum risk training for neural machine translation. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pp. 1683–1692, 2016.

Michalis K Titsias and Miguel Lázaro-Gredilla. Local expectation gradients for black box variational inference. In *Advances in Neural Information Processing Systems-Volume 2*, pp. 2638–2646, 2015.

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

Tim Vieira. Gumbel-max trick and weighted reservoir sampling, 2014. URL https://timvieira.github.io/blog/post/2014/08/01/gumbel-max-trick-and-weighted-reservoir-sampling/.

Tim Vieira. Estimating means in a finite universe, 2017. URL https://timvieira.github.io/blog/post/2017/07/03/estimating-means-in-a-finite-universe/.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *Advances in Neural Information Processing Systems*, pp. 2692–2700, 2015.

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

John I Yellott. The relationship between Luce's choice axiom, Thurstone's theory of comparative judgment, and the double exponential distribution. *Journal of Mathematical Psychology*, 15(2): 109–144, 1977.

Mingzhang Yin, Yuguang Yue, and Mingyuan Zhou. Arsm: Augment-reinforce-swap-merge estimator for gradient backpropagation through categorical variables. In *International Conference on Machine Learning*, pp. 7095–7104, 2019.

A NOTATION

Throughout this appendix we will use the following notation from Maddison et al. (2014):

$$\begin{aligned} e_\phi(g) &= \exp(-g+\phi) \\ F_\phi(g) &= \exp(-\exp(-g+\phi)) \\ f_\phi(g) &= e_\phi(g) F_\phi(g). \end{aligned}$$

This means that $F_\phi(g)$ is the CDF and $f_\phi(g)$ the PDF of the Gumbel(ϕ) distribution. Additionally we will use the identities by Maddison et al. (2014):

$$F_\phi(g) F_\gamma(g) = F_{\log(\exp(\phi)+\exp(\gamma))}(g) \tag{23}$$

$$\int_{g=a}^{b} e_\gamma(g) F_\phi(g) \partial g = (F_\phi(b) - F_\phi(a)) \frac{\exp(\gamma)}{\exp(\phi)}. \tag{24}$$

Also, we will use the following notation, definitions and identities (see Kool et al. (2019b)):

$$\phi_i = \log p(i) \tag{25}$$

$$\phi_S = \log \sum_{i \in S} p(i) = \log \sum_{i \in S} \exp \phi_i \tag{26}$$

$$\phi_{D \setminus S} = \log \sum_{i \in D \setminus S} p(i) = \log \left(1 - \sum_{i \in S} p(i)\right) = \log(1 - \exp(\phi_S)) \tag{27}$$

$$G_{\phi_i} \sim \text{Gumbel}(\phi_i) \tag{28}$$

$$G_{\phi_S} = \max_{i \in S} G_{\phi_i} \sim \text{Gumbel}(\phi_S) \tag{29}$$

For a proof of equation 29, see Maddison et al. (2014).

B COMPUTATION OF $p(S^k)$, $p^{D \setminus C}(S \setminus C)$ AND $R(S^k, s)$

We can sample the set S^k from the Plackett-Luce distribution using the Gumbel-Top-k trick by drawing Gumbel variables $G_{\phi_i} \sim \text{Gumbel}(\phi_i)$ for each element and returning the indices of the k largest Gumbels. If we ignore the ordering, this means we will obtain the set S^k if $\min_{i \in S^k} G_{\phi_i} > \max_{i \in D \setminus S^k} G_{\phi_i}$. Omitting the superscript k for clarity, we can use the Gumbel-Max trick, i.e. that $G_{\phi_{D \setminus S}} = \max_{i \notin S} G_{\phi_i} \sim \text{Gumbel}(\phi_{D \setminus S})$ (equation 29) and marginalize over $G_{\phi_{D \setminus S}}$:

$$\begin{aligned} p(S) &= P(\min_{i \in S} G_{\phi_i} > G_{\phi_{D \setminus S}}) \\ &= P(G_{\phi_i} > G_{\phi_{D \setminus S}}, i \in S) \\ &= \int_{g_{\phi_{D \setminus S}}=-\infty}^{\infty} f_{\phi_{D \setminus S}}(g_{\phi_{D \setminus S}}) P(G_{\phi_i} > g_{\phi_{D \setminus S}}, i \in S) \partial g_{\phi_{D \setminus S}} \\ &= \int_{g_{\phi_{D \setminus S}}=-\infty}^{\infty} f_{\phi_{D \setminus S}}(g_{\phi_{D \setminus S}}) \prod_{i \in S} \left(1 - F_{\phi_i}(g_{\phi_{D \setminus S}})\right) \partial g_{\phi_{D \setminus S}} \qquad (30) \\ &= \int_{u=0}^{1} \prod_{i \in S} \left(1 - F_{\phi_i}\left(F^{-1}_{\phi_{D \setminus S}}(u)\right)\right) \partial u \qquad (31) \end{aligned}$$

Here we have used a change of variables $u = F_{\phi_{D \setminus S}}(g_{\phi_{D \setminus S}})$. This expression can be efficiently numerically integrated (although another change of variables may be required for numerical stability depending on the values of $\boldsymbol{\phi}$).

Exact computation in $O(2^k)$. The integral in equation 30 can be computed exactly using the identity

$$\prod_{i \in S} (a_i - b_i) = \sum_{C \subseteq S} (-1)^{|C|} \prod_{i \in C} b_i \prod_{i \in S \setminus C} a_i$$

which gives

$$
\begin{aligned}
p(S) &= \int_{g_{\phi_{D\setminus S}}=-\infty}^{\infty} f_{\phi_{D\setminus S}}(g_{\phi_{D\setminus S}}) \prod_{i\in S} \left(1 - F_{\phi_i}(g_{\phi_{D\setminus S}})\right) \partial g_{\phi_{D\setminus S}} \\
&= \sum_{C\subseteq S} (-1)^{|C|} \int_{g_{\phi_{D\setminus S}}=-\infty}^{\infty} f_{\phi_{D\setminus S}}(g_{\phi_{D\setminus S}}) \prod_{i\in C} F_{\phi_i}(g_{\phi_{D\setminus S}}) \prod_{i\in S\setminus C} 1 \partial g_{\phi_{D\setminus S}} \\
&= \sum_{C\subseteq S} (-1)^{|C|} \int_{g_{\phi_{D\setminus S}}=-\infty}^{\infty} e_{\phi_{D\setminus S}}(g_{\phi_{D\setminus S}}) F_{\phi_{D\setminus S}}(g_{\phi_{D\setminus S}}) F_{\phi_C}(g_{\phi_{D\setminus S}}) \partial g_{\phi_{D\setminus S}} \\
&= \sum_{C\subseteq S} (-1)^{|C|} \int_{g_{\phi_{D\setminus S}}=-\infty}^{\infty} e_{\phi_{D\setminus S}}(g_{\phi_{D\setminus S}}) F_{\phi_{(D\setminus S)\cup C}}(g_{\phi_{D\setminus S}}) \partial g_{\phi_{D\setminus S}} \\
&= \sum_{C\subseteq S} (-1)^{|C|} (1-0) \frac{\exp(\phi_{D\setminus S})}{\exp(\phi_{(D\setminus S)\cup C})} \\
&= \sum_{C\subseteq S} (-1)^{|C|} \frac{1 - \sum_{i\in S} p(i)}{1 - \sum_{i\in S\setminus C} p(i)}.
\end{aligned} \tag{32}
$$

Computation of $p^{D\setminus C}(S \setminus C)$. When using the Gumbel-Top-k trick over the restricted domain $D \setminus C$, we do *not* need to renormalize the log-probabilities $\phi_s, s \in D \setminus C$ since the Gumbel-Top-k trick applies to unnormalized log-probabilities. Also, assuming $C \subseteq S^k$, it holds that $(D \setminus C) \setminus (S \setminus C) = D \setminus S$. This means that we can compute $p^{D\setminus C}(S \setminus C)$ similar to equation 30:

$$
\begin{aligned}
p^{D\setminus C}(S \setminus C) &= P(\min_{i\in S\setminus C} G_{\phi_i} > G_{\phi_{(D\setminus C)\setminus(S\setminus C)}}) \\
&= P(\min_{i\in S\setminus C} G_{\phi_i} > G_{\phi_{D\setminus S}}) \\
&= \int_{g_{\phi_{D\setminus S}}=-\infty}^{\infty} f_{\phi_{D\setminus S}}(g_{\phi_{D\setminus S}}) \prod_{i\in S\setminus C} \left(1 - F_{\phi_i}(g_{\phi_{D\setminus S}})\right) \partial g_{\phi_{D\setminus S}}.
\end{aligned} \tag{33}
$$

Computation of $R(S^k, s)$. Note that, using equation 9, it holds that

$$
\sum_{s\in S^k} \frac{p^{D\setminus\{s\}}(S^k \setminus \{s\}) p(s)}{p(S^k)} = \sum_{s\in S^k} P(b_1 = s | S^k) = 1
$$

from which it follows that

$$
p(S^k) = \sum_{s\in S^k} p^{D\setminus\{s\}}(S^k \setminus \{s\}) p(s)
$$

such that

$$
R(S^k, s) = \frac{p^{D\setminus\{s\}}(S^k \setminus \{s\})}{p(S^k)} = \frac{p^{D\setminus\{s\}}(S^k \setminus \{s\})}{\sum_{s'\in S^k} p^{D\setminus\{s'\}}(S^k \setminus \{s'\}) p(s')}. \tag{34}
$$

This means that, to compute the leave-one-out ratio for all $s \in S^k$, we only need to compute $p^{D\setminus\{s\}}(S^k \setminus \{s\})$ for $s \in S^k$. When using the numerical integration or summation in $O(2^k)$, we can reuse computation, whereas using the naive method, the cost is $O(k \cdot (k-1)!) = O(k!)$, making the total computational cost comparable to computing just $p(S^k)$, and the same holds when computing the 'second-order' leave one out ratios for the built-in baseline (equation 16).

C THE SUM-AND-SAMPLE ESTIMATOR

C.1 UNBIASEDNESS OF THE SUM-AND-SAMPLE ESTIMATOR

We show that the sum-and-sample estimator is unbiased for any set $C \subset D$ (see also Liang et al. (2018); Liu et al. (2019)):

$$
\begin{aligned}
&\mathbb{E}_{x\sim p^{D\setminus C}(x)}\left[\sum_{c\in C} p(c)f(c) + \left(1-\sum_{x\in C} p(c)\right) f(x)\right] \\
&= \sum_{c\in C} p(c)f(c) + \left(1-\sum_{c\in C} p(c)\right) \mathbb{E}_{x\sim p^{D\setminus C}(x)}[f(x)] \\
&= \sum_{c\in C} p(c)f(c) + \left(1-\sum_{c\in C} p(c)\right) \sum_{x\in D\setminus C} \frac{p(x)}{1-\sum_{c\in C} p(c)} f(x) \\
&= \sum_{c\in C} p(c)f(c) + \sum_{x\in D\setminus C} p(x)f(x) \\
&= \sum_{x\in D} p(x)f(x) \\
&= \mathbb{E}_{x\sim p(x)}[f(x)]
\end{aligned}
$$

C.2 RAO-BLACKWELLIZATION OF THE STOCHASTIC SUM-AND-SAMPLE ESTIMATOR

In this section we give the proof that Rao-Blackwellizing the stochastic sum-and-sample estimator results in the unordered set estimator.

Theorem 4. *Rao-Blackwellizing the stochastic sum-and-sample estimator results in the unordered set estimator, i.e.*

$$
\mathbb{E}_{B^k\sim p(B^k|S^k)}\left[\sum_{j=1}^{k-1} p(b_j)f(b_j) + \left(1-\sum_{j=1}^{k-1} p(b_j)\right) f(b_k)\right] = \sum_{s\in S^k} p(s)R(S^k,s)f(s). \tag{35}
$$

Proof. To give the proof, we first prove three Lemmas.

Lemma 1.

$$
P(b_k = s|S^k) = \frac{p(S^k\setminus\{s\})}{p(S^k)} \frac{p(s)}{1-\sum_{s'\in S^k\setminus\{s\}} p(s')} \tag{36}
$$

Proof. Similar to the derivation of $P(b_1 = s|S^k)$ (equation 9 in the main paper), we can write:

$$
\begin{aligned}
P(b_k = s|S^k) &= \frac{P(S^k \cap b_k = s)}{p(S^k)} \\
&= \frac{p(S^k\setminus\{s\})p^{D\setminus(S^k\setminus\{s\})}(s)}{p(S^k)} \\
&= \frac{p(S^k\setminus\{s\})}{p(S^k)} \frac{p(s)}{1-\sum_{s'\in S^k\setminus\{s\}} p(s')}.
\end{aligned}
$$

The step from the first to the second row comes from analyzing the event $S^k \cap b_k = s$ using sequential sampling: to sample S^k (including s) with s being the k-th element means that we should first sample $S^k \setminus \{s\}$ (in any order), and then sample s from the distribution restricted to $D \setminus (S^k \setminus \{s\})$. □

Lemma 2.

$$
p(S) + p(S\setminus\{s\}) \frac{1-\sum_{s'\in S} p(s')}{1-\sum_{s'\in S\setminus\{s\}} p(s')} = p^{D\setminus\{s\}}(S\setminus\{s\}) \tag{37}
$$

Dividing equation 32 by $1 - \sum_{s' \in S} p(s')$ on both sides, we obtain

Proof.

$$
\begin{aligned}
&\frac{p(S)}{1 - \sum_{s' \in S} p(s')} \\
=&\sum_{C \subseteq S} (-1)^{|C|} \frac{1}{1 - \sum_{s' \in S \setminus C} p(s')} \\
=&\sum_{C \subseteq S \setminus \{s\}} \left((-1)^{|C|} \frac{1}{1 - \sum_{s' \in S \setminus C} p(s')} + (-1)^{|C \cup \{s\}|} \frac{1}{1 - \sum_{s' \in S \setminus (C \cup \{s\})} p(s')} \right) \\
=&\sum_{C \subseteq S \setminus \{s\}} (-1)^{|C|} \frac{1}{1 - \sum_{s' \in S \setminus C} p(s')} + \sum_{C \subseteq S \setminus \{s\}} (-1)^{|C \cup \{s\}|} \frac{1}{1 - \sum_{s' \in S \setminus (C \cup \{s\})} p(s')} \\
=&\sum_{C \subseteq S \setminus \{s\}} (-1)^{|C|} \frac{1}{1 - p(s) - \sum_{s' \in (S \setminus \{s\}) \setminus C} p(s')} - \sum_{C \subseteq S \setminus \{s\}} (-1)^{|C|} \frac{1}{1 - \sum_{s' \in (S \setminus \{s\}) \setminus C} p(s')} \\
=&\frac{1}{1 - p(s)} \sum_{C \subseteq S \setminus \{s\}} (-1)^{|C|} \frac{1}{1 - \sum_{s' \in (S \setminus \{s\}) \setminus C} \frac{p(s')}{1 - p(s)}} - \frac{p(S \setminus \{s\})}{1 - \sum_{s' \in S \setminus \{s\}} p(s')} \\
=&\frac{1}{1 - p(s)} \frac{p^{D \setminus \{s\}}(S \setminus \{s\})}{1 - \sum_{s' \in S \setminus \{s\}} \frac{p(s')}{1 - p(s)}} - \frac{p(S \setminus \{s\})}{1 - \sum_{s' \in S \setminus \{s\}} p(s')} \\
=&\frac{p^{D \setminus \{s\}}(S \setminus \{s\})}{1 - p(s) - \sum_{s' \in S \setminus \{s\}} p(s')} - \frac{p(S \setminus \{s\})}{1 - \sum_{s' \in S \setminus \{s\}} p(s')} \\
=&\frac{p^{D \setminus \{s\}}(S \setminus \{s\})}{1 - \sum_{s' \in S} p(s')} - \frac{p(S \setminus \{s\})}{1 - \sum_{s' \in S \setminus \{s\}} p(s')}.
\end{aligned}
$$

Multiplying by $1 - \sum_{s' \in S} p(s')$ and rearranging terms proves Lemma 2. □

Lemma 3.

$$
p(s) + \left(1 - \sum_{s' \in S^k} p(s') \right) P(b_k = s | S^k) = p(s) R(S^k, s) \tag{38}
$$

Proof. First using Lemma 1 and then Lemma 2 we find

$$
\begin{aligned}
&p(s) + \left(1 - \sum_{s' \in S^k} p(s') \right) P(b_k = s | S^k) \\
=&p(s) + \left(1 - \sum_{s' \in S^k} p(s') \right) \frac{p(S^k \setminus \{s\})}{p(S^k)} \frac{p(s)}{1 - \sum_{s' \in S^k \setminus \{s\}} p(s')} \\
=&\frac{p(s)}{p(S^k)} \left(p(S^k) + \frac{1 - \sum_{s' \in S^k} p(s')}{1 - \sum_{s' \in S^k \setminus \{s\}} p(s')} p(S^k \setminus \{s\}) \right) \\
=&\frac{p(s)}{p(S^k)} p^{D \setminus \{s\}}(S^k \setminus \{s\}) \\
=&p(s) R(S^k, s).
\end{aligned}
$$

□

Now we can complete the proof of Theorem 4 by adding $p(b_k)f(b_k) - p(b_k)f(b_k) = 0$ to the estimator, moving the terms independent of B^k outside the expectation and using Lemma 3:

$$
\begin{aligned}
&\mathbb{E}_{B^k \sim p(B^k|S^k)} \left[\sum_{j=1}^{k-1} p(b_j) f(b_j) + \left(1 - \sum_{j=1}^{k-1} p(b_j) \right) f(b_k) \right] \\
=&\mathbb{E}_{B^k \sim p(B^k|S^k)} \left[\sum_{j=1}^{k} p(b_j) f(b_j) + \left(1 - \sum_{j=1}^{k} p(b_j) \right) f(b_k) \right] \\
=& \sum_{s \in S^k} p(s) f(s) + \mathbb{E}_{B^k \sim p(B^k|S^k)} \left[\left(1 - \sum_{s' \in S^k} p(s') \right) f(b_k) \right] \\
=& \sum_{s \in S^k} p(s) f(s) + \sum_{s \in S^k} \left(1 - \sum_{s' \in S^k} p(s') \right) P(b_k = s|S^k) f(s) \\
=& \sum_{s \in S^k} \left(p(s) + \left(1 - \sum_{s' \in S^k} p(s') \right) P(b_k = s|S^k) \right) f(s) \\
=& \sum_{s \in S^k} p(s) R(S^k, s) f(s).
\end{aligned}
$$

□

D THE IMPORTANCE-WEIGHTED ESTIMATOR

D.1 RAO-BLACKWELLIZATION OF THE IMPORTANCE-WEIGHTED ESTIMATOR

In this section we give the proof that Rao-Blackwellizing the importance-weighted estimator results in the unordered set estimator.

Theorem 5. *Rao-Blackwellizing the importance-weighted estimator results in the unordered set estimator, i.e.:*

$$
\mathbb{E}_{\kappa \sim p(\kappa|S^k)} \left[\sum_{s \in S^k} \frac{p(s)}{1 - F_{\phi_s}(\kappa)} f(s) \right] = \sum_{s \in S^k} p(s) R(S^k, s) f(s). \tag{39}
$$

Here we have slightly rewritten the definition of the importance-weighted estimator, using that $q(s, a) = P(g_{\phi_s} > a) = 1 - F_{\phi_s}(a)$, where F_{ϕ_s} is the CDF of the Gumbel distribution (see Appendix A).

Proof. We first prove the following Lemma:

Lemma 4.

$$
\mathbb{E}_{\kappa \sim p(\kappa|S^k)} \left[\frac{1}{1 - F_{\phi_s}(\kappa)} \right] = R(S^k, s) \tag{40}
$$

Proof. Conditioning on S^k, we know that the elements in S^k have the k largest perturbed log-probabilities, so κ, the $(k+1)$-th largest perturbed log-probability is the largest perturbed log-probability in $D \setminus S^k$, and satisfies $\kappa = \max_{s \in D \setminus S^k} g_{\phi_s} = g_{\phi_{D \setminus S^k}} \sim \text{Gumbel}(\phi_{D \setminus S^k})$. Computing $p(\kappa|S^k)$ using Bayes' Theorem, we have

$$
p(\kappa|S^k) = \frac{p(S^k|\kappa)p(\kappa)}{p(S^k)} = \frac{\prod_{s \in S^k}(1 - F_{\phi_s}(\kappa)) f_{\phi_{D \setminus S^k}}(\kappa)}{p(S^k)} \tag{41}
$$

which allows us to compute (using equation 33 with $C = \{s\}$ and $g_{\phi_{D\setminus S}} = \kappa$)

$$\begin{aligned}
&\mathbb{E}_{\kappa\sim p(\kappa|S^k)}\left[\frac{1}{1-F_{\phi_s}(\kappa)}\right] \\
=&\int_{\kappa=-\infty}^{\infty} p(\kappa|S^k)\frac{1}{1-F_{\phi_s}(\kappa)}\partial\kappa \\
=&\int_{\kappa=-\infty}^{\infty} \frac{\prod_{s\in S^k}(1-F_{\phi_s}(\kappa))f_{\phi_{D\setminus S^k}}(\kappa)}{p(S^k)}\frac{1}{1-F_{\phi_s}(\kappa)}\partial\kappa \\
=&\frac{1}{p(S^k)}\int_{\kappa=-\infty}^{\infty}\prod_{s\in S^k\setminus\{s\}}(1-F_{\phi_s}(\kappa))f_{\phi_{D\setminus S^k}}(\kappa)\partial\kappa \\
=&\frac{1}{p(S^k)}p^{D\setminus\{s\}}(S\setminus\{s\}) \\
=&R(S^k,s).
\end{aligned}$$

□

Using Lemma 4 we find

$$\begin{aligned}
&\mathbb{E}_{\kappa\sim p(\kappa|S^k)}\left[\sum_{s\in S^k}\frac{p(s)}{1-F_{\phi_s}(\kappa)}f(s)\right] \\
=&\sum_{s\in S^k}p(s)\mathbb{E}_{\kappa\sim p(\kappa|S^k)}\left[\frac{1}{1-F_{\phi_s}(\kappa)}\right]f(s) \\
=&\sum_{s\in S^k}p(s)R(S^k,s)f(s).
\end{aligned}$$

□

E THE UNORDERED SET POLICY GRADIENT ESTIMATOR

E.1 PROOF OF UNBIASEDNESS OF THE UNORDERED SET POLICY GRADIENT ESTIMATOR WITH BASELINE

To prove the unbiasedness of result we need to prove that the control variate has expectation 0:

Lemma 5.

$$\mathbb{E}_{S^k\sim p_{\boldsymbol{\theta}}(S^k)}\left[\sum_{s\in S^k}\nabla_{\boldsymbol{\theta}}p_{\boldsymbol{\theta}}(s)R(S^k,s)\sum_{s'\in S^k}p_{\boldsymbol{\theta}}(s')R^{D\setminus\{s\}}(S^k,s')f(s')\right]=0. \tag{42}$$

Proof. Similar to equation 9, we apply Bayes' Theorem conditionally on $b_1 = s$ to derive for $s' \neq s$

$$\begin{aligned}
P(b_2=s'|S^k,b_1=s) &= \frac{P(S^k|b_2=s',b_1=s)P(b_2=s'|b_1=s')}{P(S^k|b_1=s)} \\
&= \frac{p_{\boldsymbol{\theta}}^{D\setminus\{s,s'\}}(S^k\setminus\{s,s'\})p_{\boldsymbol{\theta}}^{D\setminus\{s\}}(s')}{p_{\boldsymbol{\theta}}^{D\setminus\{s\}}(S^k\setminus\{s\})} \\
&= \frac{p_{\boldsymbol{\theta}}(s')}{1-p_{\boldsymbol{\theta}}(s)}R^{D\setminus\{s\}}(S^k,s').
\end{aligned} \tag{43}$$

For $s' = s$ we have $R^{D\setminus\{s\}}(S^k, s') = 1$ by definition, so using equation 43 we can show that

$$\begin{aligned}
&\sum_{s' \in S^k} p_{\boldsymbol{\theta}}(s') R^{D\setminus\{s\}}(S^k, s') f(s') \\
&= p_{\boldsymbol{\theta}}(s) f(s) + \sum_{s' \in S^k \setminus \{s\}} p_{\boldsymbol{\theta}}(s') R^{D\setminus\{s\}}(S^k, s') f(s') \\
&= p_{\boldsymbol{\theta}}(s) f(s) + (1 - p_{\boldsymbol{\theta}}(s)) \sum_{s' \in S^k \setminus \{s\}} \frac{p_{\boldsymbol{\theta}}(s')}{1 - p_{\boldsymbol{\theta}}(s)} R^{D\setminus\{s\}}(S^k, s') f(s') \\
&= p_{\boldsymbol{\theta}}(s) f(s) + (1 - p_{\boldsymbol{\theta}}(s)) \sum_{s' \in S^k \setminus \{s\}} P(b_2 = s' | S^k, b_1 = s) f(s') \\
&= p_{\boldsymbol{\theta}}(s) f(s) + (1 - p_{\boldsymbol{\theta}}(s)) \mathbb{E}_{b_2 \sim p_{\boldsymbol{\theta}}(b_2 | S^k, b_1 = s)} \left[f(b_2) \right] \\
&= \mathbb{E}_{b_2 \sim p_{\boldsymbol{\theta}}(b_2 | S^k, b_1 = s)} \left[p_{\boldsymbol{\theta}}(b_1) f(b_1) + (1 - p_{\boldsymbol{\theta}}(b_1)) f(b_2) \right].
\end{aligned}$$

Now we can show that the control variate is actually the result of Rao-Blackwellization:

$$\begin{aligned}
&\mathbb{E}_{S^k \sim p_{\boldsymbol{\theta}}(S^k)} \left[\sum_{s \in S^k} \nabla_{\boldsymbol{\theta}} p_{\boldsymbol{\theta}}(s) R(S^k, s) \sum_{s' \in S^k} p_{\boldsymbol{\theta}}(s') R^{D\setminus\{s\}}(S^k, s') f(s') \right] \\
&= \mathbb{E}_{S^k \sim p_{\boldsymbol{\theta}}(S^k)} \left[\sum_{s \in S^k} p_{\boldsymbol{\theta}}(s) R(S^k, s) \nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(s) \sum_{s' \in S^k} p_{\boldsymbol{\theta}}(s') R^{D\setminus\{s\}}(S^k, s') f(s') \right] \\
&= \mathbb{E}_{S^k \sim p_{\boldsymbol{\theta}}(S^k)} \left[\sum_{s \in S^k} P(b_1 = s | S^k) \nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(s) \sum_{s' \in S^k} p_{\boldsymbol{\theta}}(s') R^{D\setminus\{s\}}(S^k, s') f(s') \right] \\
&= \mathbb{E}_{S^k \sim p_{\boldsymbol{\theta}}(S^k)} \left[\mathbb{E}_{b_1 \sim p_{\boldsymbol{\theta}}(b_1 | S^k)} \left[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(b_1) \sum_{s' \in S^k} p_{\boldsymbol{\theta}}(s') R^{D\setminus\{b_1\}}(S^k, s') f(s') \right] \right] \\
&= \mathbb{E}_{S^k \sim p_{\boldsymbol{\theta}}(S^k)} \left[\mathbb{E}_{b_1 \sim p_{\boldsymbol{\theta}}(b_1 | S^k)} \left[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(b_1) \mathbb{E}_{b_2 \sim p_{\boldsymbol{\theta}}(b_2 | S^k, b_1)} \left[p_{\boldsymbol{\theta}}(b_1) f(b_1) + (1 - p_{\boldsymbol{\theta}}(b_1)) f(b_2) \right] \right] \right] \\
&= \mathbb{E}_{S^k \sim p_{\boldsymbol{\theta}}(S^k)} \left[\mathbb{E}_{B^k \sim p_{\boldsymbol{\theta}}(B^k | S^k)} \left[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(b_1) \left(p_{\boldsymbol{\theta}}(b_1) f(b_1) + (1 - p_{\boldsymbol{\theta}}(b_1)) f(b_2) \right) \right] \right] \\
&= \mathbb{E}_{B^k \sim p_{\boldsymbol{\theta}}(B^k)} \left[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(b_1) \left(p_{\boldsymbol{\theta}}(b_1) f(b_1) + (1 - p_{\boldsymbol{\theta}}(b_1)) f(b_2) \right) \right]
\end{aligned}$$

This expression depends only on b_1 and b_2 and we recognize the stochastic sum-and-sample estimator for $k = 2$ used as 'baseline'. As a special case of equation 12 for $C = \{b_1\}$, we have

$$\mathbb{E}_{b_2 \sim p_{\boldsymbol{\theta}}(b_2 | b_1)} \left[(p_{\boldsymbol{\theta}}(b_1) f(b_1) + (1 - p_{\boldsymbol{\theta}}(b_1)) f(b_2)) \right] = \mathbb{E}_{i \sim p_{\boldsymbol{\theta}}(i)} \left[f(i) \right]. \tag{44}$$

Using this, and the fact that $\mathbb{E}_{b_1 \sim p_{\boldsymbol{\theta}}(b_1)} \left[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(b_1) \right] = \nabla_{\boldsymbol{\theta}} \mathbb{E}_{b_1 \sim p_{\boldsymbol{\theta}}(b_1)} \left[1 \right] = \nabla_{\boldsymbol{\theta}} 1 = 0$ we find

$$\begin{aligned}
&\mathbb{E}_{S^k \sim p_{\boldsymbol{\theta}}(S^k)} \left[\sum_{s \in S^k} \nabla_{\boldsymbol{\theta}} p_{\boldsymbol{\theta}}(s) R(S^k, s) \sum_{s' \in S^k} p_{\boldsymbol{\theta}}(s') R^{D\setminus\{s\}}(S^k, s') f(s') \right] \\
&= \mathbb{E}_{B^k \sim p_{\boldsymbol{\theta}}(B^k)} \left[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(b_1) \left(p_{\boldsymbol{\theta}}(b_1) f(b_1) + (1 - p_{\boldsymbol{\theta}}(b_1)) f(b_2) \right) \right] \\
&= \mathbb{E}_{b_1 \sim p_{\boldsymbol{\theta}}(b_1)} \left[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(b_1) \mathbb{E}_{b_2 \sim p_{\boldsymbol{\theta}}(b_2 | b_1)} \left[(p_{\boldsymbol{\theta}}(b_1) f(b_1) + (1 - p_{\boldsymbol{\theta}}(b_1)) f(b_2)) \right] \right] \\
&= \mathbb{E}_{b_1 \sim p_{\boldsymbol{\theta}}(b_1)} \left[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(b_1) \mathbb{E}_{x \sim p_{\boldsymbol{\theta}}(x)} \left[f(x) \right] \right] \\
&= \mathbb{E}_{b_1 \sim p_{\boldsymbol{\theta}}(b_1)} \left[\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(b_1) \right] \mathbb{E}_{x \sim p_{\boldsymbol{\theta}}(x)} \left[f(x) \right] \\
&= 0 \cdot \mathbb{E}_{x \sim p_{\boldsymbol{\theta}}(x)} \left[f(x) \right] \\
&= 0
\end{aligned}$$

□

F THE RISK ESTIMATOR

F.1 PROOF OF BUILT-IN BASELINE

We show that the RISK estimator, taking gradients through the normalization factor actually has a built-in baseline. We first use the log-derivative trick to rewrite the gradient of the ratio as the ratio times the logarithm of the gradient, and then swap the summation variables in the double sum that arises:

$$
\begin{aligned}
e^{\text{RISK}}(S) &= \sum_{s\in S} \nabla_{\boldsymbol{\theta}} \left(\frac{p_{\boldsymbol{\theta}}(s)}{\sum_{s'\in S} p_{\boldsymbol{\theta}}(s')} \right) f(s) \\
&= \sum_{s\in S} \frac{p_{\boldsymbol{\theta}}(s)}{\sum_{s'\in S} p_{\boldsymbol{\theta}}(s')} \nabla_{\boldsymbol{\theta}} \log \left(\frac{p_{\boldsymbol{\theta}}(s)}{\sum_{s'\in S} p_{\boldsymbol{\theta}}(s')} \right) f(s) \\
&= \sum_{s\in S} \frac{p_{\boldsymbol{\theta}}(s)}{\sum_{s'\in S} p_{\boldsymbol{\theta}}(s')} \left(\nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta}}(s) - \nabla_{\boldsymbol{\theta}} \log \sum_{s'\in S} p_{\boldsymbol{\theta}}(s') \right) f(s) \\
&= \sum_{s\in S} \frac{p_{\boldsymbol{\theta}}(s)}{\sum_{s'\in S} p_{\boldsymbol{\theta}}(s')} \left(\frac{\nabla_{\boldsymbol{\theta}} p_{\boldsymbol{\theta}}(s)}{p_{\boldsymbol{\theta}}(s)} - \frac{\sum_{s'\in S} \nabla_{\boldsymbol{\theta}} p_{\boldsymbol{\theta}}(s')}{\sum_{s'\in S} p_{\boldsymbol{\theta}}(s')} \right) f(s) \\
&= \sum_{s\in S} \frac{\nabla_{\boldsymbol{\theta}} p_{\boldsymbol{\theta}}(s) f(s)}{\sum_{s'\in S} p_{\boldsymbol{\theta}}(s')} - \frac{\sum_{s,s'\in S} p_{\boldsymbol{\theta}}(s) \nabla_{\boldsymbol{\theta}} p_{\boldsymbol{\theta}}(s') f(s)}{\left(\sum_{s'\in S} p_{\boldsymbol{\theta}}(s')\right)^2} \\
&= \sum_{s\in S} \frac{\nabla_{\boldsymbol{\theta}} p_{\boldsymbol{\theta}}(s) f(s)}{\sum_{s'\in S} p_{\boldsymbol{\theta}}(s')} - \frac{\sum_{s,s'\in S} p_{\boldsymbol{\theta}}(s') \nabla_{\boldsymbol{\theta}} p_{\boldsymbol{\theta}}(s) f(s')}{\left(\sum_{s'\in S} p_{\boldsymbol{\theta}}(s')\right)^2} \\
&= \sum_{s\in S} \frac{\nabla_{\boldsymbol{\theta}} p_{\boldsymbol{\theta}}(s)}{\sum_{s'\in S} p_{\boldsymbol{\theta}}(s')} \left(f(s) - \frac{\sum_{s'\in S} p_{\boldsymbol{\theta}}(s') f(s')}{\sum_{s'\in S} p_{\boldsymbol{\theta}}(s')} \right) \\
&= \sum_{s\in S} \frac{\nabla_{\boldsymbol{\theta}} p_{\boldsymbol{\theta}}(s)}{\sum_{s''\in S} p_{\boldsymbol{\theta}}(s'')} \left(f(s) - \sum_{s'\in S} \frac{p_{\boldsymbol{\theta}}(s')}{\sum_{s''\in S} p_{\boldsymbol{\theta}}(s'')} f(s') \right).
\end{aligned}
$$

G CATEGORICAL VARIATIONAL AUTO-ENCODER

G.1 CHANGES MADE TO CODE BY YIN ET AL. (2019)

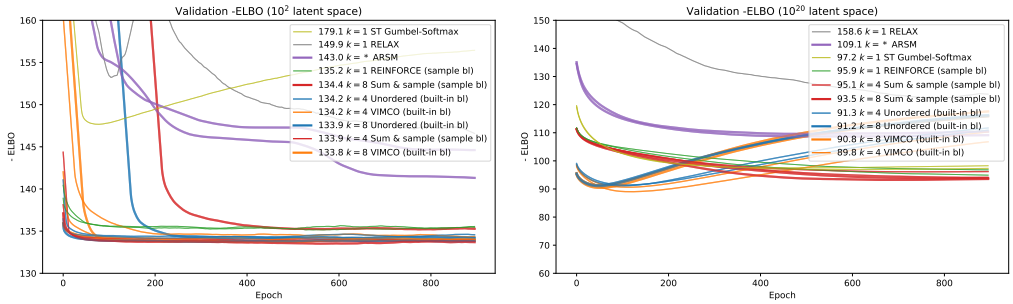
We made a number of changes to the code[5] by Yin et al. (2019), which contained some inconsistencies. This causes our results to be different from reported in Yin et al. (2019).

1. Not all estimators used the same model architecture. We used 512 and 256 hidden neurons for the encoder and 256 and 512 for the decoder and Leaky ReLu with $\alpha = 0.1$ for all estimators.
2. A bug in the code caused some estimators to use non-binarized data. This was fixed after correspondence with the authors[6], from which we learned that they actually used the standard binarized dataset[7] (Salakhutdinov & Murray, 2008; Larochelle & Murray, 2011). Our results using this dataset are in this section below.
3. The ELBO has a direct dependency on the encoder/inference model parameters, but this 'pathwise term' of the gradient (see Section 3.3) was not implemented. Adding this term improved results for REINFORCE based estimators (including ours), while we were unable to get a similar improvement with ARSM.
4. We implemented all estimators in the same file to make sure that results are computed in the same manner and using the same model architecture for all estimators.

[5] `https://github.com/ARM-gradient/ARSM/`

[6] Commit E35A954

[7] `http://www.dmi.usherb.ca/~larocheh/mlpython/_modules/datasets/binarized_mnist.html`



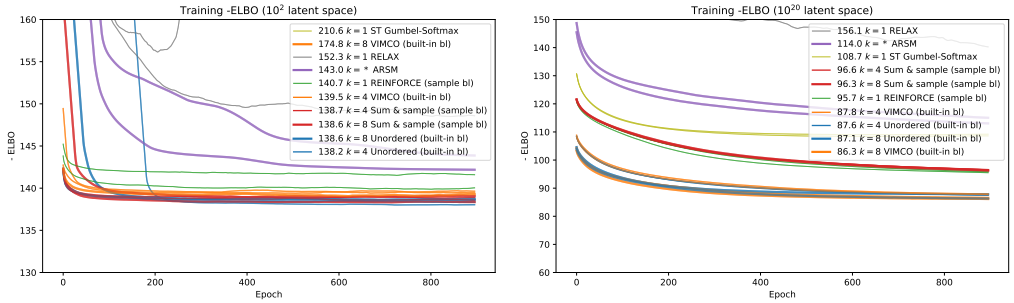
(a) Small domain (latent space size 10^2) (b) Large domain (latent space size 10^{20})

Figure 5: Smoothed validation -ELBO curves during training of two independent runs when with different estimators with $k = 1$, 4 or 8 (thicker lines) samples (ARSM has a variable number). Some lines coincide, so we sort the legend by the lowest -ELBO achieved and report this value.

G.2 Additional results

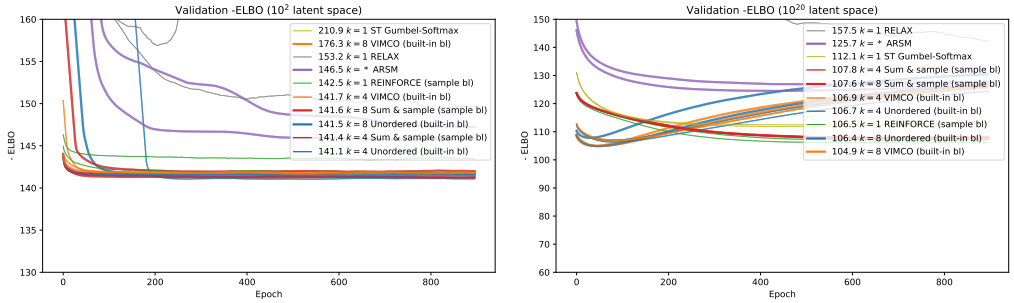
Negative ELBO on validation set. Figure 5 shows the -ELBO evaluated during training on the validation set. For the large latent space, we see validation error quickly increase (after reaching a minimum) which is likely because of overfitting (due to improved optimization), a phenomenon observed before (Tucker et al., 2017; Grathwohl et al., 2018). Note that before the overfitting starts, both VIMCO and the unordered set estimator achieve a lower validation error than the other estimators (which show less overfitting), such that in a practical setting, one can use early stopping.

Results using standard binarized MNIST dataset. Instead of using the MNIST dataset binarized by thresholding values at 0.5 (as in the code and paper by Yin et al. (2019)) we also experiment with the standard (fixed) binarized dataset by Salakhutdinov & Murray (2008); Larochelle & Murray (2011), for which we plot train and validation curves for two runs on the small and large domain in Figure 6. This gives more realistic (higher) -ELBO scores, although we still observe the effect of overfitting. As this is a bit more unstable setting, one of the runs using VIMCO diverged, but in general the relative performance of estimators is similar to using the dataset with 0.5 threshold.



(a) Training -ELBO, small domain (10^2)

(b) Training -ELBO, large domain (10^{20})



(c) Validation -ELBO, small domain (10^2)

(d) Validation -ELBO, large domain (10^{20})

Figure 6: Smoothed training and validation -ELBO curves during training on the standard binarized MNIST dataset (Salakhutdinov & Murray, 2008; Larochelle & Murray, 2011) of two independent runs when with different estimators with $k = 1$, 4 or 8 (thicker lines) samples (ARSM has a variable number). Some lines coincide, so we sort the legend by the lowest -ELBO achieved and report this value.