Revisiting Score Function Estimators for k-Subset Sampling

Klas Wijk, Ricardo Vinuesa, Hossein Azizpour

Are score function estimators an underestimated approach to learning with k-subset sampling?

We revisit score function estimators for k-subset sampling and find results competitive with popular approaches based on pathwise gradient estimation and relaxed sampling, despite weaker assumptions. Furthermore, our method produces exact samples, unbiased gradients, and introduces no hyperparameters in need of tuning.

Keywords

- Score function estimators
- k-Subset sampling
- Variance reduction

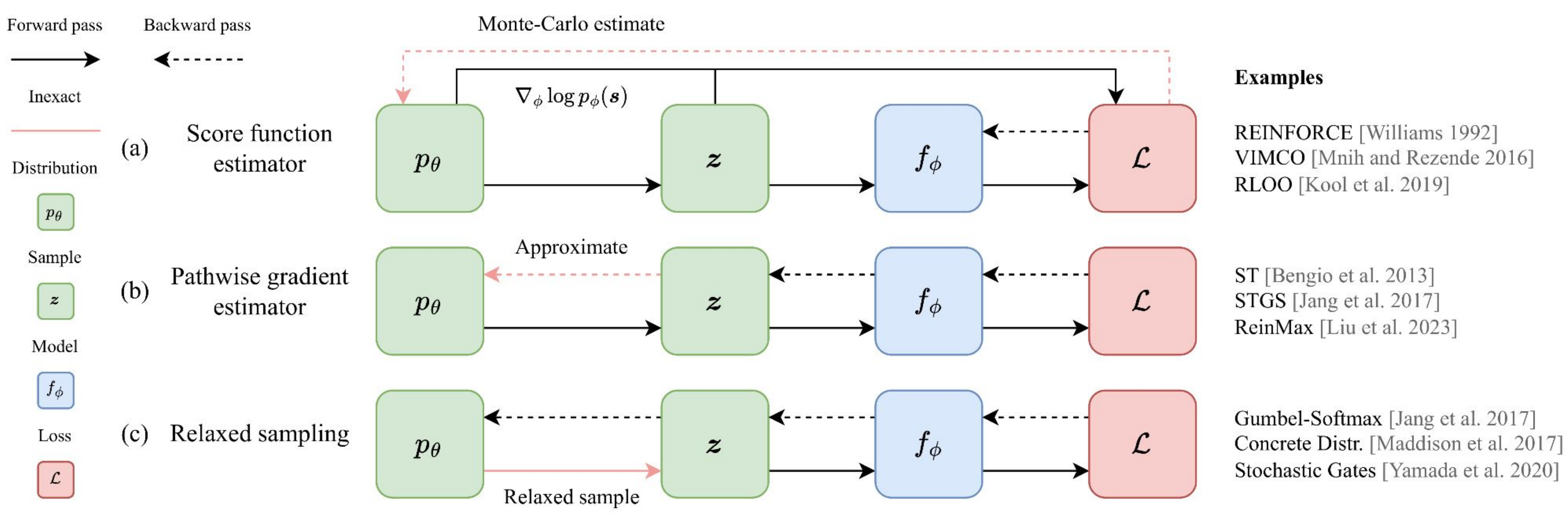


Figure 1: Three prominent approaches to learning by sampling.

The k-Subset Distribution

We model a subset using n parameters as the following conditional distribution:

$$p_{\boldsymbol{\theta},k}(\boldsymbol{z}) = p_{\boldsymbol{\theta}}\left(\boldsymbol{b} \,\middle|\, \textstyle\sum_{i=1}^n b_i = k\right)$$
$$= \frac{\prod_{i=1}^n p_{\boldsymbol{\theta}}(b_i)}{p_{\boldsymbol{\theta}}\left(\sum_{i=1}^n b_i = k\right)} \mathbb{1}\left[\textstyle\sum_{i=1}^n b_i = k\right]$$

The distribution has a combinatorially large support: all subsets.

Computing the Score Function

To construct our estimator, we need the score function, which involves computing the density of a Poisson binomial distribution. This can be efficiently done using a DFT:

$$p_{\boldsymbol{\theta}}\left(\textstyle\sum_{i=1}^n b_i = k\right)$$
$$= \frac{1}{n+1} \mathrm{DFT}\left(\prod_{i=1}^n p_{\boldsymbol{\theta}}(b_i) e^C + (1 - p_{\boldsymbol{\theta}}(b_i))\right)$$

With this, the score function is easily obtained using automatic differentiation.

Reducing the Variance with Control Variates

The vanilla score function estimator suffers from high variance. Control variates of different sorts offer a simple remedy. We opt for a multi-sample control variate:

$$\nabla_{\boldsymbol{\theta}} \mathbb{E}_{p(\boldsymbol{x})} \mathbb{E}_{p_{\boldsymbol{\theta},k}(\boldsymbol{z})}[f_{\boldsymbol{\phi}}(\boldsymbol{z}, \boldsymbol{x})]$$
$$\approx \frac{1}{NM} \sum_{i=1}^{N} \sum_{j=1}^{M} \nabla_{\boldsymbol{\theta}} \log p_{\boldsymbol{\theta},k}(\boldsymbol{z}^{(j)})$$
$$\cdot \left(f_{\boldsymbol{\phi}}(\boldsymbol{z}^{(j)}, \boldsymbol{x}^{(i)}) - \frac{1}{M-1} \sum_{k \neq j} f_{\boldsymbol{\phi}}(\boldsymbol{z}^{(k)}, \boldsymbol{x}^{(i)}) \right)$$

Results

We evaluate our method in a feature selection setting where a subset of features is sampled, passed through a downstream network, and optimized end-to-end. We use k = 30 selections.

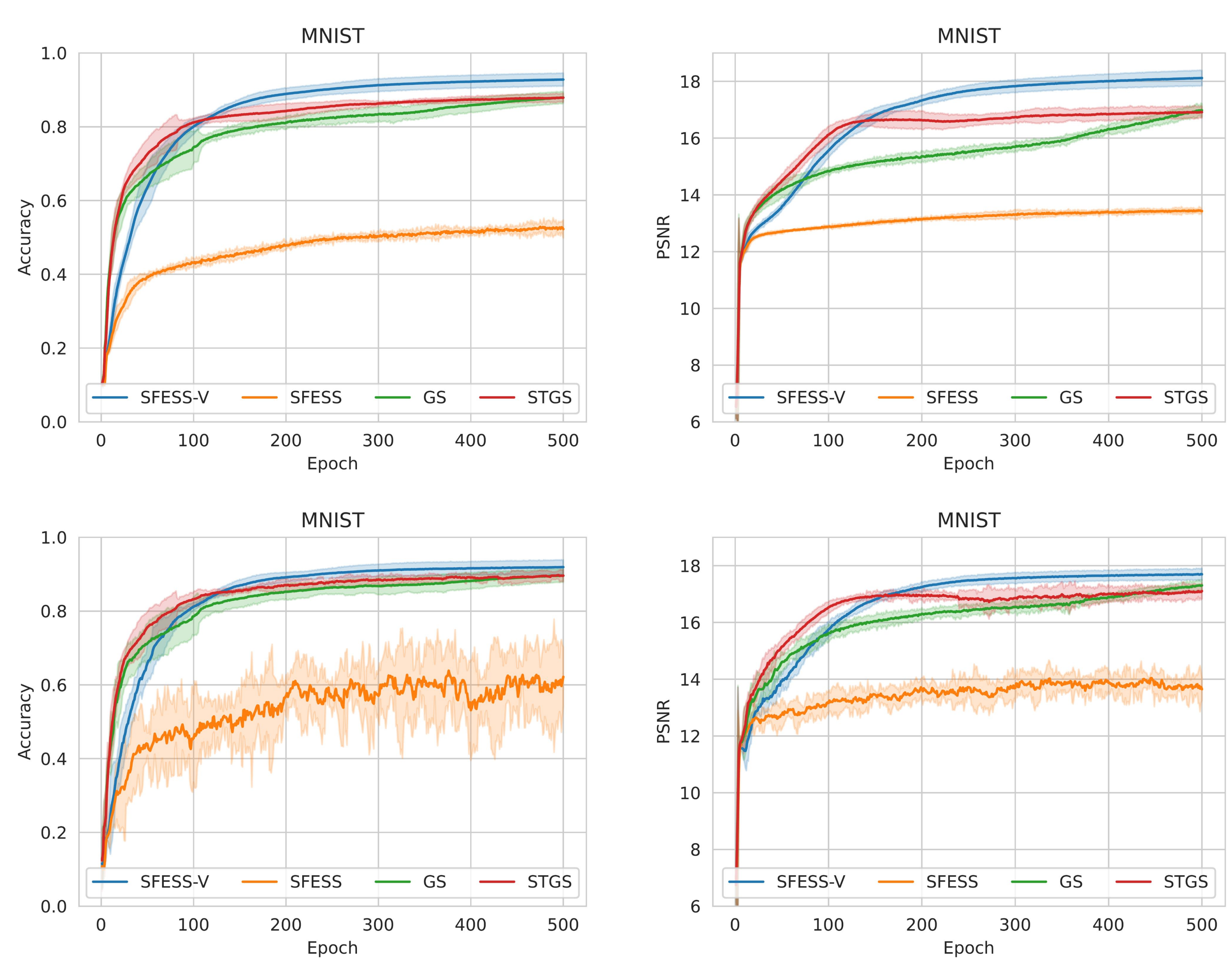
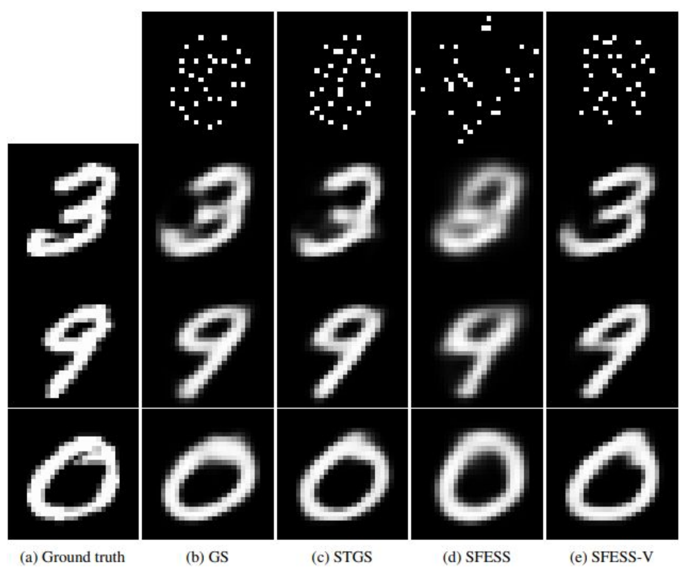


Figure 2: Training (top) and validation (bottom) metrics for classification (left) and reconstruction (right) on MNIST.

Table: Feature selection results.

Metric	Dataset	GS	STGS	SFESS	SFESS-V
PSNR ↑	MNIST	17.402 ± 0.194	17.186 ± 0.319	13.686 ± 0.412	$\mathbf{17.775 \pm 0.209}$
	Fashion-MNIST	16.922 ± 0.289	16.642 ± 0.358	15.308 ± 0.303	$\mathbf{17.805 \pm 0.075}$
	KMNIST	12.641 ± 0.083	12.561 ± 0.140	11.300 ± 0.281	$\mathbf{12.696 \pm 0.120}$
SSIM ↑	MNIST	0.771 ± 0.011	0.759 ± 0.319	0.416 ± 0.412	$\mathbf{0.796 \pm 0.209}$
	Fashion-MNIST	0.586 ± 0.017	0.578 ± 0.031	0.456 ± 0.016	$\mathbf{0.642 \pm 0.011}$
	KMNIST	0.428 ± 0.021	$\mathbf{0.464 \pm 0.048}$	0.230 ± 0.026	0.460 ± 0.022
Accuracy ↑	MNIST	0.898 ± 0.014	0.898 ± 0.019	0.627 ± 0.077	$\mathbf{0.921 \pm 0.015}$
	Fashion-MNIST	0.777 ± 0.012	0.774 ± 0.027	0.643 ± 0.112	$\mathbf{0.809 \pm 0.009}$
	KMNIST	0.604 ± 0.029	0.591 ± 0.032	0.425 ± 0.023	$\mathbf{0.634 \pm 0.059}$

Figure 3: Learned selections and reconstructions on the MNIST test set.



Paper