We thank all reviewers for their comments. A common suggestion, which we will follow, was comparing against more baselines. In §5.1, we compare against Sum&Sample, which was shown in prior work [Fig. 4 of Liu et al., 2019] to outperform RELAX (of which REBAR is a special case [Grathwohl et al., 2018]). During the rebuttal, we ran preliminary experiments with VIMCO which does not seem to outperform moving average baseline on the bit-vector experiment (even after searching across learning rates and number of K samples; though more seeds are needed to confirm this). Upon acceptance we will normalize these baselines for all experiments and include suggested ones.

Reviewer #1

> "The main weakness is the issue of scaling (...) to a setting with a large number of categories or binary variables"

Thanks for bringing this up, scalability is an important point that we want to make sure is clear in the final version. Our goal in experimenting with latent sizes $K \in \{10, 256, 2^{32}, 2^{128}\}$ is precisely to analyze how our methods scale with K.

For the VAE with binary latents, top-k sparsemax is indeed not the best choice when $D = 128$, since it always requires k decoder calls, as shown in Fig. 3. Note, however, that our other proposed method (SparseMAP) scales well, reaching very sparse solutions from the beginning of training. This is expected, as SparseMAP is better tailored for combinatorial problems, as discussed in the *Results and discussion* paragraph of §5.3. We will make this clearer.

In the categorical case, where the number of classes K can be large but not combinatorial, sparsemax appears to scale well: Fig 1 (right) and Table 1 shows that sparsemax has much fewer decoder calls than K; for extremely large K, top-k sparsemax with $k \leq K$ can be used to upper bound this number in the beginning of training.

> "The paper will be much stronger if the authors could include analysis of their method in cases with large number of categories/binaries (...) Plotting training objectives as a function of computational resources would be useful"

We will include this analysis and plots as suggested. Thank you!

Reviewer #2

> "Implementation details of the active set algorithm is a bit unclear to me. I'd appreciate a more detailed discussion"

Good suggestion, we will add a detailed discussion and pseudo-code.

> " How sparse are the optimal sparseMAP solutions (...)? Are they the same [solutions of] the active set algorithm? "

The number of decoder calls on the y-axis of Fig.3 corresponds to the SparseMAP support. Upon convergence, the support is 1 for $> 50\%$ of the examples. In general, SparseMAP is guaranteed to have a solution spanned by $D - 1$ configurations. With enough active set iterations, this guarantees the exact solution. We set the maximum number of iterations to 300 ($\gg D - 1$), and most of the time an exact solution was found in much fewer iterations.

Thank you for sharing the missing related work, we'll include it in the final version of the paper!

Reviewer #3

> "For the 2nd set of experiments (...) a temperature of 1 was used [for Gumbel-Softmax]. Were other values tested?"

Thank you for noticing this omission. Following Jang et al. [2017], we decayed the temperature throughout training. We tuned this decay rate for both experiments §5.1 and §5.2. We'll clear this up in the final version.

We will also include ELBO vs. epoch plots in the camera-ready. Thank you for your suggestions!

Reviewer #4

> "In section 5.1 (...) Do you expect these results to generalize to more complex architectures?"

Modifications to Kingma's original semi-supervised VAE include changes to the probabilistic model (e.g. more variables, different conditional independent assumptions), architecture (e.g. increasing capacity), and objective (e.g. promoting representation orthogonality). These changes seem motivated more from the point of view of disentanglement than from a potential limitation imposed by noisy gradients. While assessing our techniques in those settings is surely interesting, we feel it lies a bit beyond our current submission.

> "How does this method behave with challenging tasks that may contain many ambiguous data points?"

Good question. In Figure 3 of our paper, you can catch a glimpse of this—when the bit-vector dimensionality is $D = 128$, top-k sparsemax has full support throughout training, and SparseMAP is a better choice (see answer to R1). In the extreme case where all data points are genuinely ambiguous, the sparsity assumption may not be suitable.

> "It would be useful to add some details on [sparsemax] forward/backward passes and their computational complexity"

We will include these details, as suggested. Thank you!