Summary. We thank the reviewers for their detailed and insightful comments. We are pleased that the reviewers highlighted the importance of our motivation and relevance to the NeurIPS community (R2,R3,R4), our simple and easy-to-implement solution (R2,R4), and our strong empirical results (R2,R3,R4). We are also glad that they unanimously appreciated our clarity of writing, the soundness of our claims, reproducibility of our results, and correctness of our methodology (R1,R2,R3,R4). We are happy that, as a result, the overall sentiment was positive, with three reviewers (R2,R3,R4) recommending acceptance. R1 was mainly concerned with the novelty of our work, and suggested that we discuss additional literature and consider more baselines. We address the reviewers' concerns and questions below.

Novelty of contributions. R1 was concerned that weighted retraining and rank-based weighting functions are not novel. We agree, but *do not claim these ideas to be novel contributions of our paper*. We view our core contributions to be: 1) identifying critical failure modes of LSO, 2) addressing those issues by weighted retraining, and 3) showing that this "simple, almost trivial (in the good way), solution" (R2) yields *substantial* improvements on important practical problems (l. 44–50 of our paper). We like R3's comment: "While retraining and importance weighting aren't novel ideas, I consider the main contributions of this work to come from identifying and isolating issues with how LSO is currently used. These types of contributions are relevant of the NeurIPS community and can have considerable impact."

Comparison to cross-entropy (CE) method. R1 has pointed out that CE has similarities to our proposed method, but is not discussed in our paper. We acknowledge this, and want to highlight two main differences between the methods. Firstly, *standard CE produces only binary weights* (Boer et al), which amounts to simply adding or removing points from the training set. This is sub-optimal for reasons discussed in our paper (lines 123–9, 162–4), and consequently *we consider a strictly more general form of weighting*. Secondly, *CE has no intrinsic optimization component.* High-performing points are found only by repeatedly sampling from the generative model and evaluating target function f. By contrast, our method *explicitly selects high-performing points* using Bayesian optimization. The necessity of repeated sampling in CE makes it only suitable in cases where evaluating f is cheap, which is *not* the problem scenario that we are considering, and therefore we did not consider it to be a useful baseline. However, we agree that the similarities to CE are enough to warrant discussion in the camera-ready version. For completeness, at the request of R1, we ran some CE methods used in Angermueller et al. and Brookes et al. on our easiest task (2D shapes), where they *significantly underperform* our method (Fig. 1). Note that we did extensive hyperparameter tuning of the baselines, and plot only the best result across three random seeds.

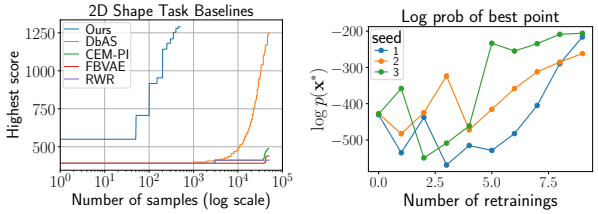

Figure 1: More baselines (left); $\log p$ over time (right)

Comparison to model-based/Bayesian optimization for chemistry (R1, R4). Model-based optimization methods in input space (e.g. ChemBO) require highly engineered search strategies infused with a lot of domain knowledge (e.g. molecular distance measures and synthesis graphs) to perform well. They are thus designed for a specific problem class (e.g. chemical design) and not applicable to other problems (e.g. the shape and expression tasks we consider). In contrast, our method can be applied to any problem without domain knowledge. Despite this generality, even for chemical design, our method is competitive with ChemBO (a SOTA model-based method), achieving a final penalized logP score of 22.55, which is even higher than the final score of 18.39 reported in the ChemBO paper (see their Table 3).

Other (mostly minor) questions and concerns raised by the reviewers.

- R2 & R4 suggested that we examine the trajectory of high-scoring points in latent space. Fig. 1 shows $\log p(\mathbf{x})$ for the overall best point $\mathbf{x}$ (with score 22.55) with 3 seeds. $\log p(\mathbf{x})$ was estimated with importance sampling using 1000 samples from the encoder. The increase supports our conjecture that such points move into the feasible region.
- *Comparison with RL* (R1). Table 1 in Appendix B.5 favourably compares our method with GCPN and MolDQN.
- *"Original" line in Fig. 4 (right)* (R1). These are the original results from Jin et al, highlighting our improvement.
- *Sudden increase of 'weight; retrain' in Fig 4* (R1). The retraining every 50 epochs incorporates new information into the latent space, causing performance to increase (l. 284-87 in our paper).
- *Why is 'weight; retrain' much better than 'weight; no retrain' on all tasks except in Fig. 4 (left)?* (R1,R4). 'weight; no retrain' already comes close to the optimal score of 0, leaving less room for further improvements by retraining. In contrast, in Fig. 3 (left) the best score is much harder to achieve, and in Fig. 4 (right) it is even unbounded.
- *Error Bars*. Error bars represent standard error over 3 seeds (R1). This accounts for the randomness in the optimization procedure (R3). Negligible variation was observed for the shapes task (R3). We agree that standard error is not the best metric in this case, and will replace it with empirical standard deviation (R3).
- *What predictive model do you use?* (R2). Our method does not require a specific implementation for h. In our experiments we used Gaussian processes, which are universal function approximators with principled uncertainty estimates. Their use is standard practice in the Bayesian optimization literature. We will clarify this in the main text.
- *"How do you find new latent space samples?"* (R2). We optimize the expected improvement function using L-BFGS.