We would like to thank all reviewers for their valuable feedback and comments. Please find our responses below.

Reviewer 1 - Use of mini-batches: in our experiments, we indeed use mini-batches of size B, by sampling B points independently according to q.

- Comparison to AdaBoost: we use the general strategy to solve minimax games used in the AdaBoost paper, as we explain in line 119. Note that AdaBoost trains and combines multiple models (weak learners), whereas we train a single model via stochastic optimization, with points sampled adaptively. AdaBoost optimizes the empirical risk, whereas we minimize the CVaR.

- Comparison to Combinatorial Bandits: we indeed use a particular combinatorial bandit algorithm for m-sets as discussed and cited in lines 153-155. We also provide a detailed comparison to other combinatorial bandit approaches in appendix F.1.

- Relation to GANs: Both approaches solve minimax games (as do many other ML approaches), but there is no deeper connection.

- Algorithm elegance: we find the algorithm quite simple and efficient as it is basically SGD with an adaptive sampling distribution which is efficiently updated.

- Number of runs: As explained in Appendix D, we do runs over 5 different random seeds.

- Large error bars: These are only in the trunc-cvar and soft-cvar algorithms, which arise from their high variance (which is one of their disadvantages).

- Interpretation of results in Fig. 2: The CVaR of Trunk-CVaR is the lowest, but the predictive accuracy is very low. This is because it predicts an almost uniform distribution. AdaCVaR instead has slightly worse CVaR, but obtains a very good predictive accuracy. AdaCVaR also has a lower CVaR than ERM (standard SGD). For detailed results please refer to Table 1 in Appendix E.

Reviewer 2 - Definition of $\hat{C}$. Thank you for observing that. It is expressed in (3), but not spelled out.

- Reference for the 'partial bandit feedback model': In Freund and Shapire 1999 they introduce such model. But maybe Lattimore and Szepesvári 2018 is a more modern introduction. We will include it when we introduce the model.

- $\hat{L}_t$ is in $\mathbb{R}^N$ and is all zeros expect in the index $i = i_t$. We will clarify the notation $[[i = i_t]]$.

- $\sim_{u.a.r.}$ means sampled uniformly at random.

- $W_{I,t}$ vs $W_{t,I}$: Yes you are correct, we will fix these nomenclature issues.

Reviewer 3 - Difference with Namkoong & Duchi (2016): The problem we address is different. Our goal is to efficiently solve the CVaR optimization problem due to its wide use as a criterion in many applications. To do so, we use a DRO formulation. The goal in N&D (2016) is to ensure the robustness of ERM w.r.t. a family of distributions. To do so they use a DR-formulation of ERM. Notice that the DR family of distributions considered by N&D (2016) *do not* contain the CVaR. Importantly, this family of distributions induces a convex structure whereas the CVaR induces a combinatorial structure, hence the algorithm we use is also different. We will clarify this difference when discussing the related work.

- Non-Convexity: We do not solve the problem of non-convexity. We propose an approach that reduces the CVaR problem to a (sequence of) Empirical Risk Minimization (ERM) problems on weighted data. Thus, as long as the resulting non-convex ERM problems can be solved, we are able to solve the (non-convex) CVaR. Empirically, SGD finds good solutions for non-convex ERM problems that arise in deep learning, hence such reduction is useful.

- High-variance: As discussed in line 120 and lines 291-299, the trunc-cvar has gradients with higher magnitude, thus stochastic gradients have higher variance. To address this, we propose adaptive sampling to reduce the variance of such gradients (avoiding the multiplication by 1/alpha). Such importance-sampling strategies are common to reduce variance. In our case, the importance sampling is adaptive because the model is changing through the optimization process.

- Other combinatorial bandits: We have a discussion in Appendix F.1. We will elaborate on the relationship with Combes et al., Combinatorial Bandits Revisited (2015) in the revised version. In particular, the work by Combes et al. considers general combinatorial sets and the regret incurred for the particular case of m-sets is higher than our algorithm by an extra $m^{1/2}$ term, which is important for $m = \alpha N$. Furthermore, the algorithm requires to compute a pseudo-inverse of a $N \times N$ matrix at each iteration which is prohibitive for large scale applications and it invalidates the benefits of stochastic optimization.