Reviewer 1: *On the assumptions in Proposition 1.* Note that, although we require such a mixture to exist, we do *not* require this mixture to be known (hence the mixture search portion of our algorithm). In practice, one could begin with a set of distributions, run Mix&Match to find the mixture distribution with smallest validation loss, and if the model does not have high enough accuracy, simply add more distributions and re-run Mix&Match. Note also that, to our knowledge, there are no other known techniques that can provably correct for distribution shift when the shift is due in part to changes in latent variables (see Remark 1). Our framework also permits shifts in $p(y|x)$, which cannot be tolerated under the common covariate shift assumption. Therefore, while such an assumption need not always be true, it allows us to prove results in a nontrivial setting, and additionally seems to be quite an effective heuristic in practice, as we demonstrate in several experiments in Section 6 and Appendix H.

Regarding comments on experiments: Refer to the response for Reviewer 4.

Lower bound on regret: Assuming you mean Theorem 3 here – the theorem is correct as stated. Recall that we are solving a minimization problem.

Reviewer 2: *On typo in β-smooth definition*: Yes, this was a typo. We however use the correct defn. in all of our proofs.

Strong convexity assumption: While ideally we could relax this assumption, it allows us to prove theorems for a variety of distribution shifts that existing techniques cannot provably correct. Additionally, this assumption does not appear to limit the practical applicability. Indeed, our algorithm performs well in practice when training neural networks on a variety of problems, as we demonstrate in our experiments.

Theorem 2 scaling with κ. Larger κ will not imply a faster convergence rate, as there is a κ dependence in the third term in $\tilde{C}$. The emphasis in Theorem 2 is on the scaling with respect to d_0 and $\mathcal{G}_*$, since Mix&Match aims to reuse models to get an exponentially decaying term in d_0^2.

Smoothness of G. We mean Lipschitz continuity, as we want close-by models to imply the solution values are close.

$G(.,.)$ **in Theorem 2**. Yes, this is a clash in notation. The use of this term is meant to follow the notation in Bottou et. al., 2018. It is defined in the formal statement of Theorem 2 (Theorem 5 in the appendix).

L251+L266 comment. The key point is that, by reusing models from the parent node, by Corollary 1, the d_0^2 term decays exponentially with height. Thus, as long as this term is large relative to the noise of the stochastic gradient, it is sufficient to take a number of steps to reach the error guarantee required by our algorithm. Beyond this point, however, the SGD budget for a node must scale with tree height.

Reviewer 3: *Validation set size*: The constraint that the validation loss can only be queried after using ≥ 1 SGD step simply ensures that, in our model, an algorithm which queries the validation loss infinitely many times without using *any* computational budget is *disallowed*. We do not require that the validation loss can be obtained accurately uniformly over all models – we only need this guarantee for the models that our guarantees require, which is much fewer. Additionally, as the search tree grows deeper, models along a given path in the tree become increasingly similar, and have similar loss (Corollary 1). Thus, we can leverage results such as the recent work "Model Similarity Mitigates Test Set Overuse," Mania et. al. 2019.

$\boldsymbol{w}^*$ vs $\boldsymbol{w}_0$ in Theorem 2. Yes, this is a typo and should be $\|\boldsymbol{w}_{t+1} - \boldsymbol{w}^*(\boldsymbol{\alpha})\|^2$.

Strong convexity. We assume also that $f(.,z)$ is convex. We use strong convexity of the averaged distribution to obtain SGD concentration results on the ℓ_2 distance between the final iterate and optimal solution (Theorem 2), and then also to argue that close mixtures imply close models (Theorem 1).

Reviewer 4: *Environment scaling+partitioning*: For more insights in scaling with respect to number of environments K, please refer to Corollary 2 in the appendix. This also provides a reference for the simplex bisection strategy. We will add more details addressing these issues in the main body of the paper.

Experiments: In the Allstate experiment (Figure 1a), the mixture is mostly ($\sim 93\%$) CT data (see Table 2 in the appendix). Thus, it seems reasonable that OnlyCT outperforms the Genie during earlier iterations when features from minority classes are less likely to be useful. For each plot, we run *all algorithms with the same hyperparameters* (SGD step size, neural network architecture, etc.). Since all algorithms we compare against *use SGD with identical parameters running simply on different mixture distributions*, we view this as a fair comparison point. Additionally, after $60k$ SGD iterations, `Genie` *does* outperform all other algorithms. We chose to include the intermediate measurement points before $60k$ iterations to increase transparency of the performance of each algorithm over time.

The hyperparameters are listed in the shell scripts in the `experiment_running` folder. The Allstate parameter settings are in `allstate_aimk_alt_newfeats_alt2.sh`. For example, in this file, the variable `NU` configures the step size for every experiment, and `BATCH` configures the SGD batch size. Line 42 of this file runs the `Genie` experiment. This script sets up the necessary parameters to run the python script for the experiments, `run_single_experiment.py`.