We thank the reviewers for the constructive feedback. We will incorporate the valuable suggestions in the revised version. We included prior works suggested by **R1** and **R5** in Section 6 and moved SAC+DIAYN results to the main text (**R4** and **R5**).

R1: Worst case objective in Equation 1, $\min$ **over** $S_{\textbf{test}}$ **vs summation.** Thank you for catching this important typo. The summation should actually be a $\min$, and we will fix this in the revised paper. The motivation in Section 4.1 above Eq 1 and all references and theoretical results using the equation are consistent with a $\min$, rather than a summation.

R1: Complexity of $\bar{\Pi}$**.** As noted in Proposition 4's proof, SMERL finds an optimal policy *for each MDP in* S_{test}, if $|\bar{\Pi}| \geq |S_{\pi^*_{\mathcal{M}},\varepsilon}|$. The set $S_{\pi^*_{\mathcal{M}},\varepsilon}$ is generally going to be much smaller than Π, so we can choose a size of $\bar{\Pi}$ that is much smaller than $|\Pi|$. However, $|S_{\pi^*_{\mathcal{M}},\varepsilon}|$ is still large, and as we note in Remark 2, metric space smoothness assumptions on MDPs in S_{test} are needed to to obtain reasonable, non worst-case performance guarantees *for all* test MDPs when a smaller $\bar{\Pi}$ is used. A natural next step is to perform an analysis with smoothness assumptions, and we will include this analysis in the final version. In our practical SMERL algorithm that we use for our experiments, $\bar{\Pi}$ is restricted to only contain 5 policies, so the optimum $\bar{\Pi}$ is much smaller than Π.

R1: SMERL vs hyperpolicy (e.g., PGPE). A key consequence of this prior approach is that the discriminator in SMERL would need to predict the policy parameters, which is more difficult than predicting a low-dimensional latent variable in a latent-variable policy. Additionally, the PGPE paper does not consider the problem of few-shot robustness.

R1: Choice of latent distribution. $p(Z)$ is the uniform distribution over the space of latent variables. In our experiments, Z is a uniform random variable over a discrete set of values. We choose $p(Z)$ to be uniform to maximize the entropy $H(Z)$ over the latent variables, which is needed to maximize $I(s_t; z)$ in Equation 3.

R3: Environments where SMERL is expected to succeed. We will add a discussion of settings in which we expect SMERL to not succeed and add an experimental setup in which we expect SMERL to fail, for example by making a perturbation non-local (e.g. many obstacles rather than a single obstacle).

R3: Assumption: near-optimal policies on the train MDPs will be near-optimal for the test environments. We agree that this assumption will not hold in all real-world problem settings. We expect this to hold in settings where an environment changes locally (i.e. a few nearby states) and there is another path that is near optimal. This is often true in real robot navigation and manipulation problems when there are a small number of new obstacles or local terrain changes. We also expect it to be true when there is a large action space (e.g. recommender systems) and local perturbations (e.g. changes in the content of a small number of items).

R3: Report performance of all the SMERL policies...report which policy is selected. We added a brief summary of these results on the HalfCheetah-Goal + Obstacles perturbation environment in the following table. We will add a more detailed empirical analysis of policy performance and selection by SMERL to the revised paper.

Obstacle		Policy ID (z_i)		
Height	0	1	2	3
0.0	**-72.2**	-87.2	-86.1	-107.9
0.15	**-72.2**	-87.2	-86.1	-107.9
0.3	-653.2	-721.1	**-86.1**	-1004.3
0.45	-690.1	-761.6	**-86.1**	-1004.3

R4: Optimal reward computation. The optimal return is the expected discounted sum of rewards obtained over an entire episode by the optimal policy. In practice, we estimate the optimal return by measuring the return achieved by a trained SAC agent's policy over the whole trajectory averaged over different initial states. In SMERL, we compare the current policy's whole trajectory return with the optimal return.

R4: Difference between the optimal return and the current policy's return with different initial states. In the experiments, the variability in the initial state distribution was small, so the SMERL policy's trajectory return for each episode was individually compared with the optimal return estimate. In other environments, a separate optimal return estimate for different initial state regions may be required.

R4: Selection of ε**.** We experimented with different values of ε on the HalfCheetah-Goal + Obstacle task (see Fig. 5 in App. B). We computed the percentage of the optimal return that this ε value corresponds to, and used this percentage to select ε for all environments. We have added a description of this protocol to the revised paper.

R4: SAC+DIAYN. SAC+DIAYN uses a weighted sum of the environment and DIAYN reward. We experimented with 3 different weights on the DIAYN reward on the HalfCheetah-Goal + Obstacle test environment and found SMERL to consistently outperform SAC+DIAYN. We have included these results in the revised paper.

R5: RARL. For RARL, we performed a grid search on HalfCheetah-Goal with evaluation on the Obstacles perturbation to select the hyper-parameters of TRPO, the policy optimizer for the protagonist and adversarial policies. We then kept these hyper-parameters fixed for all environments. We have added a detailed description to the revised paper.

R5: How are latent variables sampled? During both the learning (see Alg 1) and the adaptation phases, the latent variable z is sampled at the beginning of an episode and is held fixed for the episode. We will clarify this Section 4.2.

R5: How is the hyper-parameter B used? The hyper-parameter B is associated with epsilon in Equations 2-5. It is the return value that SMERL must achieve prior to the unsupervised reward being added to the objective. Specifically, $B = R_{\mathcal{M}}(\pi^*_{\mathcal{M}}) - \varepsilon$. We have added this description to the revised paper.