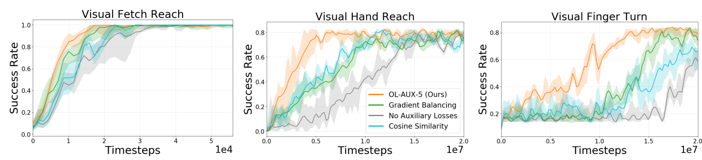
We thank the reviewers for the positive feedback on our motivation and algorithm. As most of the concerns are on our empirical study, we now address these concerns with additional experiments and some clarifications:

More random seeds: We originally used 2 random seeds as goal conditioned reinforcement learning has a relative low variance from our experience. We have re-run all the experiments with 5 random seeds and all our results still hold. The updated figure of the submitted paper's Figure 2 is shown on the right.



Comparison with more baselines: 1) **Hand tuned, fixed weights:** We compare OL-AUX with hand-tuned weights either on a single auxiliary task, where the best fixed weight is found with grid search (Figure 1), or on all the auxiliary tasks where the best fixed weights are the final weights learned by OL-AUX (Figure 2). It shows that our method can adaptively combine auxiliary tasks and outperforms the best fixed weight. 2) **No gradient balancing:** In our original experiments, we compare to the cosine similarity method [Yunshu et al. 2018] with gradient balancing added for fair comparison. We show in Figure 3 that cosine similarity performs worse when gradient balancing is removed.

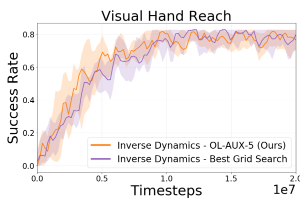


Figure 1: Comparing handtuned weight, single auxiliary task.

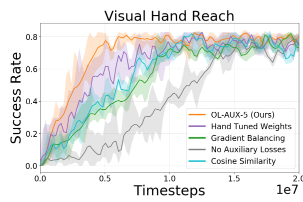


Figure 2: Comparing handtuned weights, all auxiliary tasks.

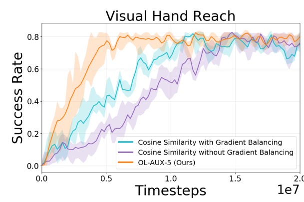


Figure 3: Effect of gradient balancing on cosine similarity.

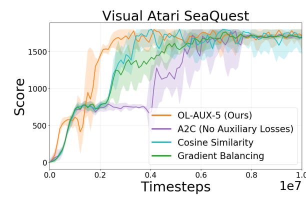


Figure 4: Atari seaquest

Empirical results on benchmark RL tasks: In Figure 4,5,6, we show that in three benchmark RL environments in Atari, OL-AUX also outperforms all the baselines. The base algorithm we use is A2C [Mnih et al. 2016]. All hyper-parameters are the same as the ones used in the paper or are default to A2C. The same set of auxiliary tasks are also used. This shows that OL-AUX gives significant improvement across different domains.

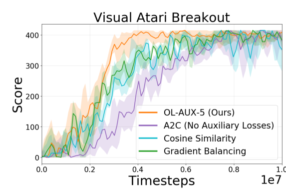


Figure 5: Atari breakout

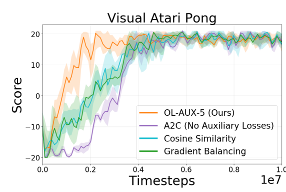


Figure 6: Atari pong

Ability to separate harmful auxiliary tasks: In Figure 3 of the orignal paper, we show that AutoEncoder is a harmful auxiliary task for Finger Turn environment. Here, a toy example in Figure 7 with one positive auxiliary task and one harmful auxiliary task shows that our algorithm is able to avoid adversarial auxiliary tasks without any prior knowledge. In this 2d example, the main task loss is $L(x, y) = x^2 + y^2$. There are two auxiliary tasks, $L_1(x, y) = (x - 0.5)^2 + (y - 0.5)^2$ and $L_2(x, y) = -L(x, y)$. Using a fixed weight for auxiliary tasks (Left), the agent converges to a sub-optimal point. Our method finds the optimum of the main task from different starting points (Middle). The auxiliary task weights during training for our method (Right) shows that our method is able to ignore L_2, which is a harmful auxiliary task.

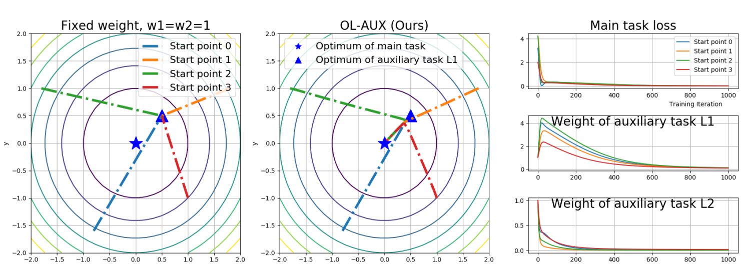


Figure 7: Ignoring adversarial auxiliary tasks

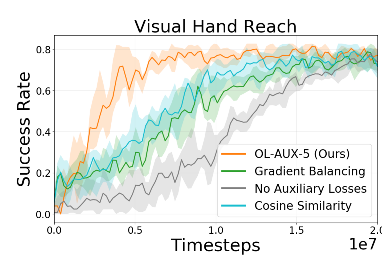


Figure 8: Learning with a binary reward of {0, 1} in goal-conditioned RL.

Response to **Reviewer 3**: The MuJoCo environments we tested effectively all have sparse rewards, since scaling and translating the rewards from {-1, 1} to {0, 1} does not change the ordering of the value function or the optimal policy. Nevertheless, in Figure 8 we show one experiment using a negative reward of 0 where our method performs just as well. Additionally, most of our hyper-parameters in the paper are taken from the defaults of Hindsight Experience Replay.