A Appendix

A.1 Additional Results/Analysis

Additional results and analysis on Multistage Reacher are shown in Figure 1. QMP outperforms all the baselines in this task set.

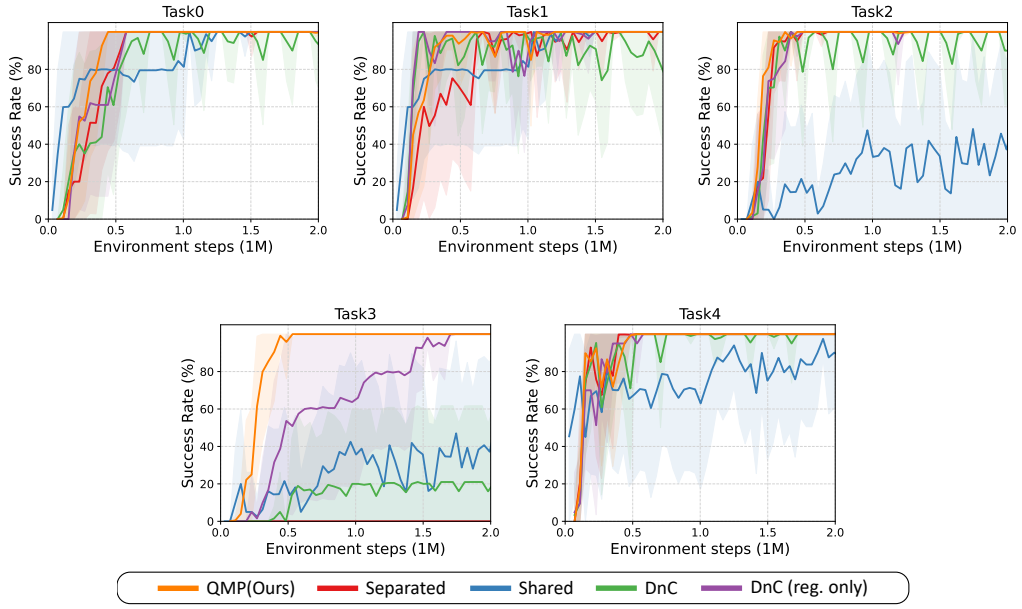

Figure 1: Success rates for individual tasks on Multistage Reacher. Our method especially helps in learning Task 3, which requires extra exploration because it only receives sparse reward.

A.2 Environment Details

Multistage Reacher We implement our multistage reacher tasks on top of the Open AI Gym (**?**) Reacher environment by defining a sequence of 3 subgoals per task. The reward function is the default gym reward function based on the distance to the goal plus an additional bonus for every subgoal completed. Task 3 recieves only a sparse reward of 1 for every subgoal reached. Task 4 has one fixed goal set at its initial position.

Maze Navigation The layout and dynamics of the maze follow **?**, but since their original design aims to train a single agent reaching a fixed goal from multiple start locations, we modified it to have both start and goal locations fixed in each task, as in **?**. The start location is still perturbed with a small noise to avoid memorizing the task. The layout we used is `LARGE_MAZE` which is an 8×11 maze with paths blocked by walls. The complete set of 10 tasks are visualized in 2, where green and red spots correspond to the start and goal locations, respectively. The environment provides an agent a dense reward of $\exp(-dist)$ where $dist$ is a linear distance between the agent's current position and the goal location. It also gives a penalty of 1 at each time step, in order to prevent the agent from exploiting the reward by staying near the goal. The episode terminates as soon as the goal is reached, by having $dist < 0.5$, or 600 time steps are passed.

Meta-World Manipulation We reproduce the Meta-world environment proposed by **?** using the Meta-world codebase (**?**). We additionally remove the previous state from the observation space so the policies cannot easily infer the current task. We use this modified environment instead of the Meta-world benchmark because our method assumes a consistent environment across tasks, thus the door and drawer should both be on the tabletop for all tasks.

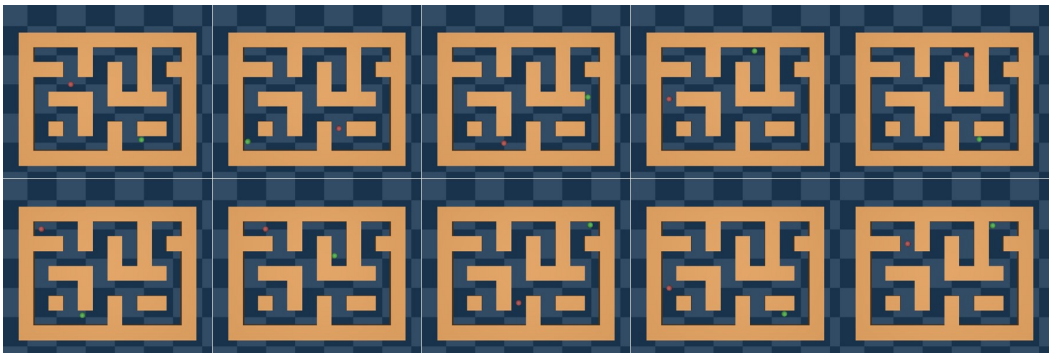

Figure 2: Ten tasks defined for the Maze Navigation. The start and goal locations in each task are shown in green and red spots, respectively. Best viewed in color.

A.3 Implementation Details

The SAC we used in our experiments is based on the open-source implementation from Garage (**?**). We used fully connected layers for the policies and Q-functions with the default hyperparameters listed in Table 1. For DnC baselines, we reproduced the method in Garage to the best of our ability with minimal modifications.

A.3.1 Hyperparameters

Table 1 details the list of important hyperparameters on all the 3 environments.

Table 1: QMP hyperparameters.

Hyperparameter	Multistage Reacher	Maze Navigation	Meta-World Manipulation
# Layers in π and Q	2	2	2
Activation function	tanh	tanh	tanh
Hidden dimension	256	256	256
Minimum buffer size (per task)	10000	3000	10000
# Environment steps per update (per task)	1000	600	500
# Gradient steps per update (per task)	100	100	50
Batch size	32	256	256
Learning rates for π, Q and α	0.0003	0.0003	0.0015
Target update frequency	1	1	1
Target update tau (τ)	0.995	0.995	0.995
Discount factor (γ)	0.99	0.99	0.99

A.3.2 Separated

All T networks have the same architecture with the hyperparameters presented in Table 1.

A.3.3 Shared

Since it is the only model with a single policy, we increased the number of parameters in the network to match others and tuned the learning rate. The hidden dimension of each layer is 600 in Multistage Reacher, 834 in Maze Navigation, and 512 in Meta-World Manipulation, and we kept the number of layers at 2. The number of environment steps as well as the number of gradient steps per update were increased by T times so that the total number of steps could match those in other models. For the learning rate, we tried 4 different values (0.0003, 0.0005, 0.001, 0.0015) and chose the most performant one. The actual learning rate used for each experiment is 0.0003 in Multistage Reacher and Maze Navigation, and 0.001 in Meta-World Manipulation.

This modification also applies to the Shared Multihead baseline, but with separate tuning for the network size and learning rates. In Multistage Reacher, we used layers with hidden dimensions of 512 and 0.001 as the final learning rate. In Maze Navigation, we used 834 for hidden dimensions and 0.0003 for the learning rate.

A.3.4 DNC

We used the same hyperparameters as in Separated, while the policy distillation parameters and the regularization coefficients were manually tuned. Following the settings in the original DnC (**?**), we adjusted the period of policy distillation to have 10 distillations over the course of training. The number of distillation epochs was set to 500 to ensure that the distillation is completed. The regularization coefficients were searched among 5 values (0.0001, 0.001, 0.01, 0.1, 1), and we chose the best one. Note that this search was done separately for DnC and DnC with regularization only. For DnC, the coefficients we used are: 0.001 in Multistage Reacher and Maze Navigation, and 0.001 in Meta-World Manipulation. For Dnc with regularization only, the values are: 0.001 in Multistage Reacher, 0.0001 in Maze Navigation, and 0.001 in Meta-World Manipulation.

A.3.5 QMP

Our method also uses the default hyperparameters. It requires one additional hyperparameter to decide when to start using the mixture of policies in exploration. Before that, each agent will collect the data using its own policy as an exploration policy. This 'mixture warmup period' was searched over 3 values (0, 50, or 100 iterations), and the best option was chosen. In all environments, we found the period of 0 works the best.

As in Shared Multihead, the QMP Multihead also required a separate tuning. Since QMP Multihead has effectively one network, we increased the network size in accordance with Shared Multihead, and tuned the learning rate in addition to the mixture warmup period. The best performing combinations of these parameters we found are 0 and 0.001 in Multistage Reacher, and 100 and 0.0003 in Maze Navigation.