META Q-LEARNING ON SHEAF-REGULARIZED DQNS FOR MULTI-AGENT PATH FINDING

AVANIKO ASOKKUMAR, KEVIN SONG, AND ZACH SWIDEY

ABSTRACT. We study cross-task generalization in grid-based multi-agent path finding (MAPF) with decentralized deep Q-learning. Starting from SIGMA, a sheaf-regularized DQN that encourages local-to-global consensus among agents, we add a Meta-Q Learning (MQL) style training and adaptation procedure. We evaluate four training paradigms across multiple MAPF tasks (map generators), reporting success rate, makespan, and few-shot adaptation behavior. In 0-shot transfer, the meta-learned model is roughly on par with multi-task averaged training; with 1000 episodes of test-time adaptation, it improves average success by 10.5%. Overall, our results suggest that combining explicit coordination structure (sheaf regularization) with off-policy meta-RL adaptation improves robustness to distribution shift in cooperative planning.

Code: `https://github.com/kevins4202/SIGMA-MQL#`.

1. INTRODUCTION

Multi-Agent Path Finding (MAPF) seeks collision-free paths for multiple agents from start to goal on a shared discrete environment. Classical optimal planners (e.g., ODrM*) can be highly effective but scale poorly as agent count or environment complexity increases [3]. Learning-based decentralized planners amortize computation into a policy, enabling fast inference at test time, but often struggle to generalize across environment distributions (e.g., from "random" clutter to structured "warehouse" layouts) [1, 4].

This project asks: can we improve cross-task generalization of decentralized DQN-based MAPF policies by meta-learning an initialization + adaptation rule, rather than learning a single averaged policy? We build on SIGMA, which injects a sheaf-theoretic consensus loss to improve coordination under partial observability [1]. We then incorporate Meta-Q-Learning (MQL), an off-policy meta-RL method that (i) conditions Q-learning on a learned context and (ii) adapts with a small number of off-policy TD updates using limited new-task data [2].

1.1. Contributions.

- We implement and evaluate four training paradigms for decentralized MAPF: (1) task-specific training, (2) multi-task averaged training, (3) transfer via finetuning, and (4) meta-Q learning with few-shot adaptation.
- We provide a cross-task generalization matrix over `random/warehouse/house/maze` map generators, reporting success rate and makespan.
- We analyze when meta-learning helps vs. when straightforward finetuning can match or exceed it, and connect these outcomes to distribution shift severity and adaptation budget.

2. BACKGROUND

2.1. MAPF as a Decentralized RL Problem. We consider a grid world with n homogeneous agents. Each episode assigns start/goal pairs and terminates on success (all agents reach goals) or timeout. Each agent has a partial observation (field-of-view) and selects from a discrete action set (move NSEW or wait), with penalties for time and collisions and a terminal reward for finishing [1]. We treat each map generator as a task: $\{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_4\} = \{\texttt{random}, \texttt{warehouse}, \texttt{house}, \texttt{maze}\}$.

2.2. SIGMA: Sheaf-Regularized DQN. SIGMA augments a DQN with a learned restriction map $M(\cdot)$ and a global section loss that encourages neighboring agents' latent representations to agree (local-to-global consistency), improving coordination and reducing deadlocks in structured environments [1]. Concretely, SIGMA adds a consensus loss l_{sec} to the TD loss and injects $M(s)$ into the Q-function computation (see Figure 1).

2.3. Meta-Q Learning (MQL). Meta-Q-Learning is an off-policy meta-RL approach that learns a Q-function that can adapt quickly to a new task with few additional updates [2]. MQL emphasizes: (i) a learned context variable summarizing recent trajectory history, (ii) a simple multi-task meta-training objective (optimize average performance across tasks), and (iii) an adaptation phase that performs a small number of off-policy TD updates using limited new-task data (and optionally reuses meta-training replay data via importance weighting) [2].

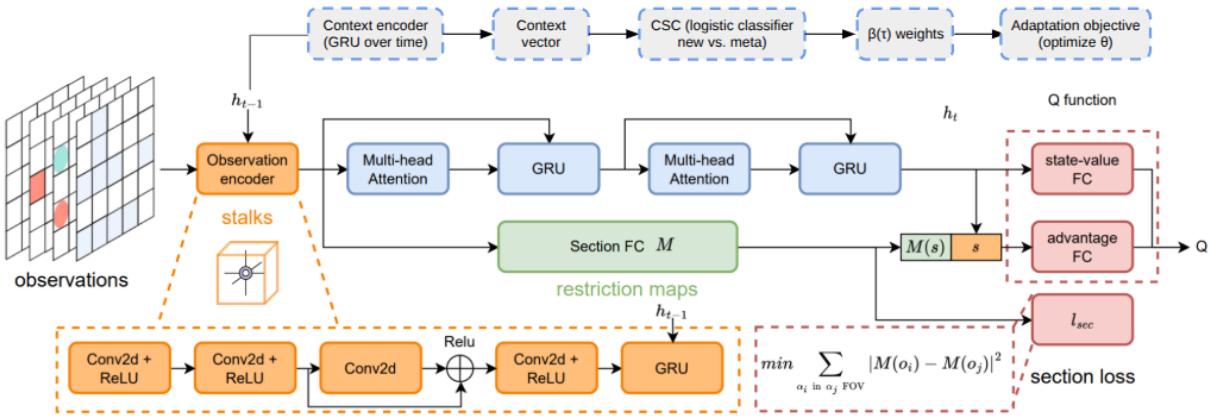


FIGURE 1. SIGMA with an MQL side branch: context encoding and CSC-derived β weights reweight TD losses for test-time adaptation, while the main Q-value computation stays unchanged.

3. RELATED WORK

Classical MAPF planners include coupled search and CBS-style methods; ODrM* is a strong deterministic planner that improves scalability via subdimensional expansion and operator decomposition [3]. Learning-based decentralized MAPF methods include PRIMAL, DHC, and transformer/communication-based approaches such as SCRIMP, which improves scalability via learned communication under small FOVs [4]. SIGMA differs by explicitly learning a consensus structure using sheaf-inspired losses [1]. In meta-RL, MQL is a representative off-policy method focused on data reuse and quick adaptation through TD updates, contrasting with gradient-based meta-learning like MAML and latent-context methods like PEARL [2].

4. APPROACH

4.1. **Tasks and Problem Setup.** We define each environment generator as a task $\mathcal{T} \in \{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \mathcal{T}_4\}$. All experiments share the same base configuration unless otherwise stated: grid size - 15x15 unit grid, 3 agents, 0.3 obstacle density, 512 episode horizon, and the same reward function as SIGMA. We evaluate generalization by training on one (or many) tasks and testing on all tasks.

4.2. **Four Training Paradigms.** We compare four paradigms under a comparable compute budget (total gradient updates and environment steps):

- **Task-Specific Training (Single-Task SIGMA):**
  Train separate SIGMA models on each task $\mathcal{T}_i$ and test each trained model on all tasks. This isolates cross-task transfer without any multi-task exposure.
- **Multi-Task Averaged Training (MT-SIGMA):**
  Train a single SIGMA model while sampling a task uniformly at random each episode/update. This produces an "averaged" policy that may generalize but can underfit any one task.
- **Transfer via Finetuning (SIGMA + FT):**
  Pretrain on a source task $\mathcal{T}_s$ and finetune on a target task $\mathcal{T}_t$ for n epochs/updates, then test. We set n to match the *adaptation budget* used in the meta method for fairness.
- **Meta-Q Learning (SIGMA-MQL):**
  Meta-train a single model on all tasks and evaluate on each test task with *few-shot adaptation*. Concretely, we collect k episodes (or m transitions) on the test task, perform n off-policy TD updates, and then evaluate success/makespan/collisions. This follows the MQL idea of adapting with limited new-task data via off-policy updates [2].

4.3. **Fairness and Compute Accounting.** We standardize: (i) total gradient updates (or total env steps), (ii) network capacity, (iii) replay buffer size and sampling, (iv) evaluation protocol (same number of episodes/seeds).

5. Experimental Results

5.1. **Experimental Protocol.** We evaluate on four tasks: `random`, `warehouse`, `house`, `maze`. Each run trains for 50,000 updates and is evaluated on 200 test episodes per task with random seeds. We grid tested on map size (15x15, 30x30, 45x45) and number of agents (3, 6, 12, 24, 48). Metrics: success rate (%), average makespan (steps on successful episodes), arrival rate (average ratio of agents that are on target by episode termination), and few-shot adaptation curves at $\{0, 100, 200, 1000\}$ episodes of adaptation data.

5.2. **Single-Task SIGMA.**

TABLE 1. Single-Task SIGMA: average success rate (%) after 50,000 updates on 45x45 size map and 48 agents. Rows: train task. Columns: test task.

Train \ Test	Random	Warehouse	House	Maze	Avg
Random	**95**	55	60	57	66.8
Warehouse	83	**89**	80	80	83.0
House	90	83	**88**	76	84.3
Maze	81	82	78	**82**	80.8
Avg	87.3	77.3	76.5	73.8	78.7

Observations: Structured tasks generalize better to unstructured tasks, likely due to more bottlenecks, and learning of communication as mentioned in the SIGMA paper. Points like doorways and corridors where many robots have to have to fit through require planning, especially in high-agent density environments. Despite warehouse being structured, the many straight line movement patterns seem simpler to learn than those in the maze and house. The diagonal suggests tasks' inherent difficulty go from Random (Easiest), Warehouse, House, to Maze (Hardest).

5.3. **Multi-Task Averaged Training.**

TABLE 2. Multi-task averaged / MT-SIGMA: success rate (%) after 50,000 updates total on 45x45 size map and 48 agents. One model trained on all tasks (task sampled uniformly during training), tested on each task without adaptation.

Model	Random	Warehouse	House	Maze	Avg
MT-SIGMA	82	82	81	71	79.0

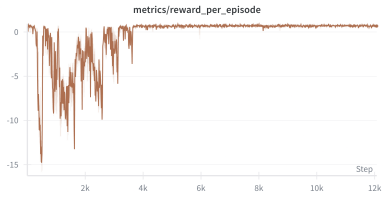
Observations: The Multi-task average training is generally worse overall compared to the Single-Task SIGMA in-distribution evaluation metrics. The relative performance seems worse on structured environments Maze and House. However, this does have some generalization capabilities. MT-SIGMA is not absolutely better than ST-SIGMA (ex. All ST-SIGMAs other than random perform better than MT-SIGMA). Interestingly, MT-SIGMA performs marginally better on Warehouse than ST-SIGMA-Warehouse on Warehouse, but we are unsure if this is significant.

5.4. **Finetuning Transfer.**

TABLE 3. PT→FT-SIGMA: success rate (%) after 50,000 finetuning updates on the target task on 45x45 size map and 48 agents (column). Rows: pretraining task. Columns: finetuning+test task.

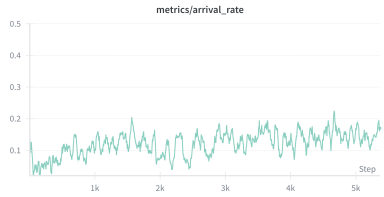
Pretrain \ FT+Test	Random	Warehouse	House	Maze	Avg
Random	**96**	88	87	70	85.3
Warehouse	94	**91**	89	78	88.0
House	95	88	**92**	79	88.5
Maze	N/A	N/A	N/A	**N/A**	N/A
Avg	94.0	89.0	89.3	75.7	87.3

Observations: Finetuning (FT) from House onto all tasks has the highest absolute success rate. The greatest difference between FT and Single Task is for Random onto all. We did not observe overfitting when finetuning was ran, but there was some training instability for the House finetuned on Maze example, which might have been fixed with better hyperparameter (learning rate) selection. We ran into compute/environment issues while doing the Maze finetuning section, so we could not run that train/test sequence.



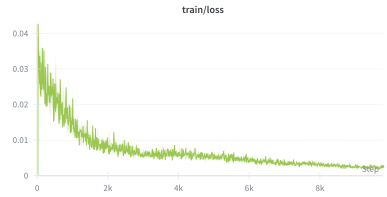
(A) Reward of House-FT on Warehouse

We chose finetuning to be 50k updates arbitrarily, but you can see rewards often converged before that (14k updates in this case).



(B) AR of House Single Task Training

Arrival rates are far more unstable at start of Single Task training then on finetuning examples.



(C) Loss of House-FT on Warehouse

Loss is not particularly informative, even on the finetuned models, given the dueling Q Network architecture.

FIGURE 2. WANDB logging examples and insights

5.5. **SIGMA-MQL.**

TABLE 4. SIGMA-MQL, 100-shot: success rate (%) after 50,000 meta-training updates total (12,500 per task) and 100 adaptation updates on 45x45 size map and 48 agents on test task.

Model	**Random**	**Warehouse**	**House**	**Maze**	**Avg**
SIGMA-MQL	90	89	86	78	85.8

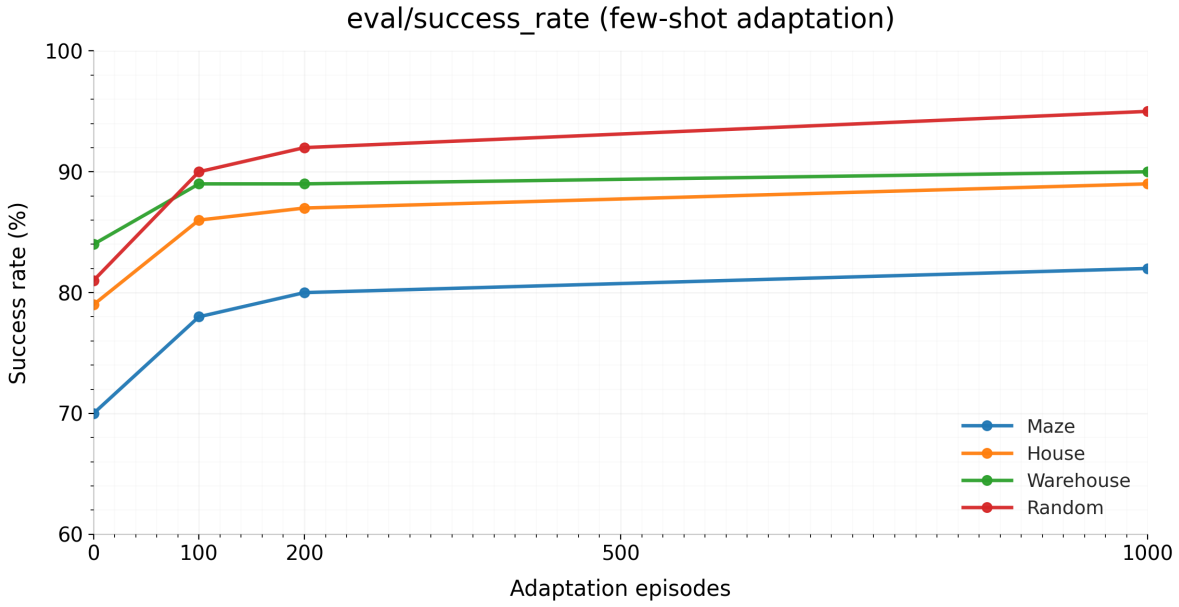


FIGURE 3. Few-shot adaptation chart. MQL-style adaptation uses a small number of off-policy TD updates on new-task data [2].

Observations: The 0-shot metric does not seem to be significantly better than the MT-SIGMA performance, which is expected because ascertaining task from a single timestep in the observation encoder is difficult, given the limited FOV of each agent. While tasks might be different on a global scale, the local field and states fed into the DQN are similar from task to task. On hard tasks (maze/warehouse), adaptation seems to close the gap with ST-SIGMA.

5.6. **Comparative Summary.**

TABLE 5. Summary across paradigms.

Metric	**ST-SIGMA**	**MT-SIGMA**	**SIGMA + FT**	**SIGMA-MQL**
Avg in-distribution success (%)	88.5	79.0	93	85.8
Avg out-of-distribution success (%)	75.4	79.0	85.3	85.8
Generalization gap (ID - OOD)	13.1	0.0	7.7	0.0
# models required	4	1	4	1
Test-time updates needed	0	0	$50000/0$	100

Key Findings:

(1) **Generalization:** SIGMA-MQL beats ST-SIGMA, MT-SIGMA, and SIGMA + FT on average OOD success.
(2) **Adaptation:** SIGMA-MQL improves from 0-shot to 1,000-shot by an average of 10.5% across the four tasks.
(3) **When finetuning wins:** FT outperforms SIGMA-MQL, but total FT network has more training time overall.

6. DISCUSSION

Our goal was robustness under task shift: train on one map family and deploy on another without retraining from scratch. Across paradigms, we see a consistent tradeoff between specialization (best in-task metrics) and portability (good performance across generators with minimal per-task overhead).

Single-Task SIGMA can achieve strong performance on its training generator, but cross-task transfer degrades when obstacle morphology and bottleneck structure shift. MT-SIGMA reduces this brittleness by training on all generators, but the averaged objective can underfit task-specific conventions, especially on harder structured layouts like `maze`. Finetuning (PT→FT) is a powerful baseline: with enough target updates, it can approach task-specialist performance. However, it requires heavy target-side training (and typically a separate finetuned model per target), which is costly when tasks are not known until runtime or when the number of deployment tasks is large.

SIGMA-MQL targets the complementary regime: a single shared prior plus a small, bounded test-time adaptation budget. Figure 3 shows consistent improvements as we increase adaptation from 0 to 1000 episodes, with diminishing returns. Averaged across tasks, success improves from 78.5% (0-shot) to 89.0% (1000-shot), a 10.5% relative gain. This suggests that even when local observations are ambiguous across generators (limited FOV), off-policy TD updates can tune the policy to the target's congestion patterns and coordination constraints.

Structured layouts impose long-range interaction constraints: narrow corridors and repeated bottlenecks amplify the cost of local mistakes, leading to deadlocks and cascading failures. SIGMA's consensus loss biases representations toward coordination, and meta-learning concentrates capacity on what varies across task families while using adaptation to specialize those decisions at deployment.

Our evaluation is limited to 15–45 length square grids and up to 48 agents in some cases; scaling to larger teams and denser obstacles may change the relative ranking. We also did not run a full ablation suite (e.g., removing the sheaf loss, sweeping the section-loss weight, or comparing against a non-SIGMA MQL baseline), so we cannot yet isolate which component drives which gains. Finally, while we logged makespan and arrival rate, we did not perform a fully scaled analysis across these metrics; in regimes where success saturates, makespan/collisions become the more informative discriminators.

7. CONCLUSION

We compared single-task, multi-task averaged, finetuned, and meta-Q-trained SIGMA-style DQNs on multiple MAPF tasks. Overall, SIGMA-MQL provides the best tradeoff between generalization and adaptation cost, supporting the thesis that explicit coordination structure plus meta-RL improves robustness under distribution shift.

REFERENCES

[1] S. Liao, W. Xia, Y. Cao, W. Dai, C. He, W. Wu, and G. Sartoretti, *SIGMA: Sheaf-Informed Geometric Multi-Agent Pathfinding*, arXiv:2502.06440, 2025.
[2] R. Fakoor, P. Chaudhari, S. Soatto, and A. Smola, *Meta-Q-Learning*, ICLR, 2020. (arXiv:1910.00125)
[3] C. Ferner, G. Wagner, and H. Choset, *ODrM*: Optimal Multirobot Path Planning in Low Dimensional Search Spaces*, ICRA, 2013.
[4] Y. Wang, B. Xiang, S. Huang, and G. Sartoretti, *SCRIMP: Scalable Communication for Reinforcement- and Imitation-Learning-Based Multi-Agent Pathfinding*, AAMAS, 2023.