Interpretable Multi-Agent Path Finding via Decision Tree Extraction from Neural Policies

Thibault Lahire
Dassault Aviation
Saint-Cloud, France
thibault.lahire@dassault-aviation.com

Abstract

Multi-Agent Path Finding (MAPF) is a fundamental problem in robotics and logistics, where multiple agents must reach their goals without collisions. While deep Multi-Agent Reinforcement Learning methods have recently shown impressive scalability and adaptability, their black-box nature hinders interpretability and trust—crucial aspects for deployment in real-world systems. In this work, we propose an interpretable policy distillation framework for MAPF. We first formulate MAPF as a stochastic game and execute a trained neural policy across diverse environments to build a large dataset of state–action pairs. We then distill this neural policy into a decision tree model that captures its underlying decision rules while maintaining strong performance. Through extensive evaluation, we analyze the trade-off between interpretability and performance, demonstrating that our distilled models achieve high fidelity to the original policy while providing transparent, human-understandable reasoning about agent behavior.

Introduction

A wide range of real-world applications such as warehouse logistics, autonomous vehicle coordination, and drone fleet management can be formulated as Multi-Agent Path Finding (MAPF) problems. In MAPF, several agents must reach their respective goal locations while avoiding collisions with obstacles and with each other. Finding optimal solutions with respect to flowtime or makespan is known to be NP-hard (Stern et al. 2019). Despite this complexity, numerous solvers have been developed that can produce optimal (Sharon et al. 2012), bounded-suboptimal (Boyarski et al. 2015), or feasible solutions (Standley 2010). However, these methods are usually centralized, requiring global information and costly replanning whenever the environment changes. This makes them unsuitable for large-scale or partially observable real-time domains.

In recent years, Multi-Agent Reinforcement Learning (MARL) has emerged as a promising alternative (Zhang, Yang, and Başar 2021). By learning decentralized policies that can operate under partial observability, MARL approaches allow agents to react autonomously to local observations. Centralized Training with Decentralized Execution (CTDE) has become a dominant paradigm, enabling coordinated yet scalable behaviors (Foerster et al. 2018; Rashid et al. 2020). MARL-based MAPF methods have demonstrated strong generalization across various environments and robustness to dynamic changes such as new obstacles or agent failures.

Yet, a major drawback of neural network-based policies remains their lack of interpretability. Deep networks encode decisions in millions of parameters, making it difficult to understand why an agent behaves in a certain way, to trust its behavior in safety-critical scenarios, or to debug undesired outcomes. For real-world multi-agent systems—such as autonomous fleets or human-robot collaboration—interpretability is essential to ensure transparency, accountability, and human oversight.

In this work, we aim to bridge the gap between performance and interpretability in MARL for MAPF. Specifically, we propose to distill the policy of a trained neural network into an interpretable model, such as a decision tree (Breiman, Friedman, and Stone 2017). Our goal is to retain as much of the performance as possible while providing a model that can be inspected, explained, and potentially verified.

To achieve this, we first construct a comprehensive dataset by running the neural network policy in a variety of MAPF environments. Each agent's local observation and corresponding action are recorded, forming a large collection of state-action pairs. This dataset serves as the supervision source for training interpretable models. Our first contribution is a detailed methodology for building this dataset, ensuring it captures the diversity of situations encountered by agents during navigation.

We perform a qualitative analysis of the decision rules learned by the interpretable models, examining whether these rules correspond to intuitive human reasoning for navigation and coordination. Our second contribution is an extensive evaluation of interpretable policies derived from this dataset. We assess their fidelity to the original neural network and their effectiveness in MAPF environments of varying complexity.

Related Work

Multi-Agent Path Finding

We focus on maps represented as undirected, unweighted graphs $G = \langle V, E \rangle$, where the vertex set V contains all possible locations and the edge set E contains all possible transitions between adjacent locations. A MAPF instance $\mathcal{I}$ consists of a map G and a set of agents $\mathcal{D} = \{1, \ldots, N\}$, where each agent $i \in \mathcal{D}$ has a start location $v_i^{\text{start}} \in V$ and a goal location $v_i^{\text{goal}} \in V$. We assume that start and goal vertices are unique for all agents, i.e., $v_i^{\text{start}} \neq v_j^{\text{start}}$ and $v_i^{\text{goal}} \neq v_j^{\text{goal}}$ for all $i \neq j$.

The objective of MAPF is to find collision-free plans for all agents. A plan $P = \{p_1, \ldots, p_N\}$ consists of individual paths $p_i = \langle p_{i,0}, \ldots, p_{i,\ell(p_i)} \rangle$ for each agent $i \in \mathcal{D}$, where $\langle p_{i,t}, p_{i,t+1} \rangle \in E$, $p_{i,0} = v_i^{\text{start}}$, and $p_{i,\ell(p_i)} = v_i^{\text{goal}}$. We consider two types of conflicts: vertex conflicts $\langle i, j, v, t \rangle$, when two agents occupy the same vertex $v \in V$ at time t, and edge conflicts $\langle i, j, u, v, t \rangle$, when two agents traverse the same edge $\langle u, v \rangle = \langle v, u \rangle \in E$ in opposite directions at time t (Silver 2005). A plan P is feasible if it contains no vertex or edge conflicts. The goal is to find a feasible plan P^* that minimizes the global objective, such as the total flowtime $\sum_{p \in P} \ell(p)$.

Despite its NP-hardness (Yu and LaValle 2013), a variety of MAPF solvers exist that can find optimal (Sharon et al. 2012), bounded-suboptimal (Cohen and Koenig 2016), or fast feasible solutions (Li et al. 2021). However, most of these solvers rely on centralized computation and global information, which limits scalability and flexibility in dynamic or partially observable domains where costly replanning would otherwise be required.

Multi-Agent Reinforcement Learning

Multi-Agent Reinforcement Learning (MARL) offers an alternative decentralized approach to MAPF. A MARL problem can be formalized as a partially observable stochastic game

$$\mathcal{M} = \langle \mathcal{D}, S, A, P, R, Z, \Omega \rangle,$$

where $\mathcal{D} = \{1, \ldots, N\}$ is the set of agents, S is the state space, $A = A_1 \times \cdots \times A_N$ the joint action space, $P(s_{t+1}|s_t, a_t)$ the transition probability, $R(s_t, a_t) = \langle r_{t,1}, \ldots, r_{t,N} \rangle \in \mathbb{R}^N$ the joint reward, Z the set of local observations, and $\Omega(s_{t+1}) = z_{t+1} = \langle z_{t+1,1}, \ldots, z_{t+1,N} \rangle \in Z^N$ the joint observation function.

Each agent i maintains an action-observation history $\tau_{t,i} \in (Z \times A_i)^t$ and follows a local policy $\pi_i(a_{t,i}|\tau_{t,i})$, forming the joint policy $\pi = \langle \pi_1, \ldots, \pi_N \rangle$. The local policy can be evaluated by a value function $Q_i^\pi(s_t, a_t) = \mathbb{E}_\pi[R_{t,i}|s_t, a_t]$, where $R_{t,i} = \sum_{k=0}^{T-1} \gamma^k r_{t+k,i}$ is the return of agent i, $\gamma \in [0, 1]$ is the discount factor, and T the time horizon.

In cooperative MARL, the goal is to find an optimal joint policy $\pi^* = \langle \pi_1^*, \ldots, \pi_N^* \rangle$ that maximizes a shared performance metric, typically the global return:

$$Q_{\text{tot}}^\pi(s_t, a_t) = \sum_{i \in \mathcal{D}} Q_i^\pi(s_t, a_t).$$

Recent progress in MARL, such as value factorization (Sunehag et al. 2017; Rashid et al. 2020) and communication-based coordination (Sukhbaatar, Fergus et al. 2016), has significantly improved coordination among agents. Combined with centralized training and decentralized execution (CTDE) (Foerster et al. 2018), exemplified by the COMA algorithm, these methods allow agents to learn coordinated, robust behaviors while acting independently during execution.

Interpretable Policies and Distillation Methods

While deep MARL has shown strong performance in complex environments, neural policies remain largely black-box models. Their internal representations are difficult to interpret, which hinders trust, safety verification, and transferability to real-world systems. In safety-critical domains such as autonomous navigation, interpretability is essential for human oversight and debugging.

Model distillation (Hinton, Vinyals, and Dean 2015) provides a principled way to transfer knowledge from a complex model (the *teacher*) to a simpler one (the *student*). When applied to reinforcement learning, this approach—known as policy distillation (Rusu et al. 2015)—aims to train a new policy that mimics the behavior of a pretrained agent while using a smaller or more interpretable architecture.

In the context of interpretability, decision trees and rule-based models have gained attention for their transparency and their ability to express decision boundaries in human-understandable terms (Craven 1996; Bastani, Kim, and Bastani 2017). Recent works have explored using decision trees as interpretable surrogates for reinforcement learning policies (Coppens et al. 2019; Verma et al. 2018), balancing explainability and performance.

In this work, we follow this direction and propose to distill a MARL policy trained for MAPF into a decision tree. This allows us to analyze the learned decision rules, assess their faithfulness to the original neural policy, and evaluate their ability to generalize across various MAPF scenarios.

Background

MAPF as a Stochastic Game

To study interpretable policies in the context of Multi-Agent Path Finding (MAPF), we adopt the stochastic game formulation introduced in prior works (Silver 2005; Phan et al. 2024). We consider discrete gridworld environments in which each cell can be free, occupied by an obstacle, or occupied by an agent. Formally, each environment is represented as a tuple $\mathcal{G} = \langle V, E \rangle$, where V denotes the set of grid cells and E the set of edges connecting adjacent cells under 4-neighborhood connectivity.

At each time step t, the joint state is defined as $s_t = \langle v_{t,1}, \ldots, v_{t,N} \rangle \in S \subseteq V^N$, where $v_{t,i}$ is the position of agent i and all positions are distinct, i.e., $v_{t,i} \neq v_{t,j}$ for $i \neq j$. Each agent i selects an action $a_{t,i}$ from a discrete action space $\mathcal{A}_i = \{\text{wait}, \text{north}, \text{south}, \text{west}, \text{east}\}$. State transitions are deterministic: valid moves update the agent's position, while invalid moves (e.g., collisions, out-of-bound actions, or simultaneous traversal of the same edge by two agents) are converted into a *wait* action.

The individual reward function is defined as:

$$r_{t,i} = \begin{cases} +1 & \text{if agent } i \text{ reaches its goal } v_i^{\text{goal}}, \\ 0 & \text{if agent } i \text{ remains at its goal,} \\ -1 & \text{otherwise.} \end{cases}$$

Each agent i observes a local neighborhood around its position $v_{t,i}$ through a fixed-size field of view (FOV). Following prior work (Sartoretti et al. 2019), a 7×7 FOV is adopted. The observation $z_{t,i}$ is encoded as a multi-channel tensor including: (1) obstacle locations, (2) nearby agents, (3) visible goals, (4) the agent's own goal if present in the FOV, and (5) a channel encoding the Manhattan distance and direction to the goal. This is illustrated in Figure 1, from (Phan et al. 2024).

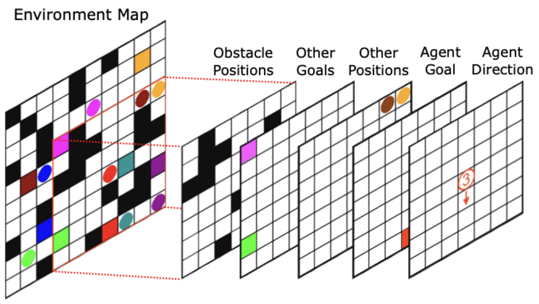

Figure 1: Example for an individual observation of the red agent in a gridworld domain. Agents are represented as colored circles, their goals as similarly-colored squares, and obstacles as black squares. Each agent i has a limited field of view (FOV) of the environment map, which is centered around its location encoded by five channels: locations of obstacles, location of other agents' goals, locations of nearby agents, and location of the goal v_i^{goal} if within the FOV, and the Manhattan distance and direction of agent i to its goal.

With a discount factor $\gamma = 1$, the negated return $-R_{t,i}$ equals the travel distance $\ell(p_i)$ if the goal is reached, or the episode horizon T otherwise. Thus, maximizing the total value

$$Q_{\text{tot}} = \sum_{i \in \mathcal{D}} Q_i$$

in MARL is equivalent to minimizing the expected flowtime in MAPF. This simple reward structure discourages unnecessary delays and implicitly penalizes collisions, enabling a general black-box formulation without handcrafted penalties or heuristics.

Neural Network Training

The neural policy used in this work is trained using the CACTUS framework (Phan et al. 2024), which introduces a confidence-based curriculum learning strategy for MARL in MAPF domains. The key idea is to progressively increase the complexity of training environments in accordance with the confidence of the agent's policy.

At the beginning of training, agents are placed in small, obstacle-free grids with short start-goal distances. As the training progresses and the actor network gains confidence—measured through the stability of its performance—the curriculum gradually introduces more challenging settings. These include larger grid sizes, higher obstacle densities, and longer goal distances.

The actor network, denoted as π_{NN}, is a fully-connected feedforward neural network that operates on flattened local observations of size $d = 245$ (corresponding to a 5-channel 7×7 field of view). The network consists of two hidden layers, each containing 64 neurons with ELU activation functions, which introduce non-linearities while maintaining smooth gradient flow:

$$\text{Input (245)} \rightarrow \text{Linear(64)} \rightarrow \text{ELU} \rightarrow \text{Linear(64)}$$
$$\rightarrow \text{ELU} \rightarrow \text{Linear(5)} \rightarrow \text{Softmax}.$$

The output layer produces a probability distribution over the discrete action space $\{\text{wait, north, south, west, east}\}$ using a softmax activation. This architecture is lightweight yet expressive enough to capture complex policies based on partial observations, enabling agents to effectively coordinate and avoid collisions in multi-agent scenarios.

This incremental training approach helps stabilize learning, mitigates sparse reward issues, and encourages the emergence of cooperative behaviors among agents. The resulting neural policy π_{NN} achieves high task performance and generalization across diverse gridworld configurations, providing a strong foundation for subsequent distillation into an interpretable model.

Contribution

Dataset Construction for Policy Distillation

To distill the learned neural policy π_{NN} into an interpretable surrogate, we first construct a large-scale dataset that captures its behavior across varied environments. Each data sample corresponds to an individual agent's decision and is represented as a pair (x, y) where:

$$x = z_{t,i} \in \mathbb{R}^d, \qquad y = \pi_{\text{NN}}(x),$$

with $\pi_{\text{NN}}(x)$ denoting the action probability distribution output by the trained policy. We systematically generate environments by varying three parameters:

- grid size $G \in \{10, 40, 80\}$,
- obstacle density $\rho \in \{0, 0.1, 0.2, 0.3\}$,
- number of agents $N \in \{4, 8, 16\}$.

For each combination (G, ρ, N), 1000 random environments are generated. During simulation, the neural policy acts autonomously, producing tuples (x, y) at each decision step. The resulting dataset therefore approximates the empirical distribution of agent behaviors under the trained MARL policy.

Because exhaustive enumeration of all configurations is infeasible, this sampling strategy emphasizes diversity over completeness, ensuring the dataset contains representative examples of both typical and rare scenarios (e.g., congestion, deadlocks, narrow passages). Note also that the CACTUS training is done for a given multi-agent system (i.e. a given number of agents). As the number of agents is varying in our study, the performance of the neural network are not as good as those presented in the original research article (Phan et al. 2024).

Decision Tree Distillation and Analysis

Using the collected dataset, we train a Decision Tree (DT) classifier to approximate the neural policy. The DT learns a mapping $\hat{\pi}_{\text{DT}} : \mathbb{R}^d \to \mathcal{A}$ by minimizing the cross-entropy loss between the neural policy outputs and the DT predictions:

$$\mathcal{L}_{\text{distill}} = - \sum_{(x,y)\in\mathcal{D}} \sum_{a\in\mathcal{A}} y_a \log \hat{\pi}_{\text{DT}}(a|x).$$

We evaluate the performance and interpretability of the distilled Decision Tree (DT) policy. We select a tree of depth 4, which offers a good balance between expressiveness and interpretability. The dataset is split into a training set (80%) and a test set (20%) to assess generalization performance.

Table 1 reports the quantitative evaluation of the distilled model. The DT achieves an overall test accuracy of 0.81, demonstrating that a relatively shallow interpretable model can faithfully reproduce most behaviors of the original neural policy.

Action	Precision	Recall	F1-score	Support
NORTH (1)	0.87	0.78	0.82	17076
SOUTH (2)	0.76	0.78	0.77	16789
WEST (3)	0.81	0.86	0.83	18594
EAST (4)	0.81	0.83	0.82	14741

Table 1: Classification report for the distilled DT policy.

The results show that the model performs particularly well on the dominant movement classes (*SOUTH*, *EAST*, and *WEST*), with F1-scores above 0.77. The *WAIT* (0) action remains difficult to predict due to its rarity and contextual dependence (e.g., temporary congestion or deadlocks), which are underrepresented in the dataset.

Beyond classification fidelity, we also assess the distilled DT policy on the original MAPF task metrics to evaluate its practical effectiveness. Specifically, we report the *completion rate* and the *total reward*. The completion rate measures the proportion of agents that successfully reach their goal positions at the end of each episode, while the total reward corresponds to the cumulative sum of rewards collected by all agents throughout the episode. These metrics provide a more direct evaluation of how well the distilled policy preserves the functional behavior of the neural policy in a multi-agent navigation setting.

Table 2 summarizes the results obtained on a 40×40 grid with an obstacle density of 0.0. The DT achieves a completion rate and total reward comparable to those of the original neural policy, indicating that the distilled model retains most of the operational performance despite its simplicity.

Model	Completion Rate	Total Reward
Neural Policy	0.627 ± 0.227	-1194 ± 1099
DT (depth 4)	0.627 ± 0.239	-1212 ± 1154

Table 2: MAPF performance metrics comparing the neural policy and its distilled DT counterpart on a 40×40 grid.

The results demonstrate that the distilled Decision Tree policy closely matches the neural policy in terms of both completion rate and total reward. This suggests that the distilled model not only captures the decision boundaries of the neural policy but also generalizes effectively in the original MAPF environment, achieving near-equivalent task-level performance while remaining interpretable.

From Decision Tree to Human-Readable Rules

Rather than visualizing the full decision tree, which can be difficult to read, we converted it into a compact set of human-readable decision rules of the form `IF condition THEN action`. Each feature index corresponds to a specific spatial position and channel in the $5 \times 7 \times 7$ observation tensor, following the flattening order (channel-first). Here is one of the rule that has been extracted:

```
IF feat17 > 0.55 AND feat23 <= 0.59
    AND feat25 <= 0.67 THEN action = 1.
```

`feat17`, `feat23`, `feat25`, and `feat31` belong to Channel 0, encoding the normalized Euclidean distances from the goal to the agent's neighboring cells (respectively North, West, East, and South). These values implicitly indicate the goal direction — the smaller the value, the closer the goal lies in that direction. The interpretation of this rule is the following. Since `feat23` and `feat25` are small, and `feat17` is large, the agent is going north. This type of interpretation can be done for each extracted rule.

From these rules, we observe that the Decision Tree primarily relies on interpretable spatial cues: the relative direction and distance to the goal (features from Channel 0), and the presence of obstacles or other agents (Channels 2 and 3). This is fully consistent with intuitive navigation behavior — the policy learns to move toward the smallest-distance direction while avoiding obstacles and collisions.

Visual Interpretation of Feature Importance

To further assess interpretability, we visualize the relative importance of each feature channel in the Decision Tree. For each channel, we aggregate the normalized feature importance scores across the entire tree and project them back onto the 7×7 observation grid.

The resulting activation maps (Fig. 2) highlight which spatial regions of the FOV most strongly influence the agent's decisions. For instance, goal-related features show strong activations along the ”Distance to goal” axis. Similarly, high activations appear near the frontal and lateral cells

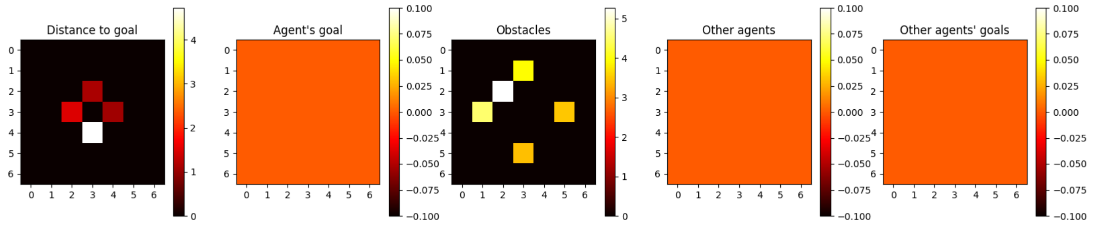

Figure 2: Spatial importance maps for each observation channel. White areas indicate high feature importance in decision-making.

of the obstacle channel, confirming that obstacle avoidance plays a dominant role in policy decisions.

These visualizations provide a clear and intuitive understanding of the decision process, bridging the gap between black-box deep reinforcement learning and interpretable symbolic reasoning.

Sensitivity Analysis

In this section, we analyze the robustness of both the neural and distilled Decision Tree (DT) policies with respect to various factors: the tree depth, obstacle density, and grid size. All experiments use the same neural policy trained with 8 agents. While absolute performance values are lower than those reported in the original study (Phan et al. 2024), the following analysis provides useful insights into the relative behaviors of the two models under different environmental conditions.

All experiments were executed on a x86_64 GNU/Linux (Ubuntu 18.04.5 LTS) machine equipped with an Intel i7-8700 CPU (8 cores at 3.2GHz) and 64 GB RAM.

Effect of Decision Tree Depth

Table 3 reports the influence of tree depth on fidelity and task performance, under fixed grid size (40×40) and zero obstacle density. Accuracy measures the DT's agreement with the neural policy on the test set, while completion rate and total reward quantify task-level performance on MAPF.

Depth	Accuracy	Completion rate	Total reward
NN	–	0.921 ± 0.050	-473 ± 364
4	0.81	0.885 ± 0.064	-551 ± 374
5	0.82	0.904 ± 0.052	-507 ± 351
6	0.84	0.894 ± 0.060	-537 ± 376
7	0.85	0.898 ± 0.064	-534 ± 401
8	0.85	0.898 ± 0.064	-534 ± 401

Table 3: Sensitivity of Decision Tree performance to model depth (Grid size = 40, Obstacle density = 0).

Deeper trees yield slightly higher fidelity, as indicated by improved accuracy. However, beyond depth 5, performance on the MAPF task saturates, with completion rates and rewards plateauing near those of the neural policy. This suggests that while additional depth enhances representational capacity, a moderate depth (4–6) is sufficient to capture most of the neural policy's structure without significant loss in task-level performance.

Effect of Obstacle Density

Table 4 presents the sensitivity of both models to increasing obstacle density, under a fixed 40×40 grid and DT depth 4.

Density	Model	Completion rate	Total reward
0.0	NN	0.921 ± 0.050	-473 ± 364
	DT	0.885 ± 0.064	-551 ± 374
0.1	NN	0.267 ± 0.077	-1866 ± 1100
	DT	0.221 ± 0.093	-1956 ± 1118
0.2	NN	0.073 ± 0.037	-2257 ± 1254
	DT	0.081 ± 0.033	-2241 ± 1259
0.3	NN	0.033 ± 0.031	-2331 ± 1261
	DT	0.040 ± 0.025	-2322 ± 1276

Table 4: Sensitivity to obstacle density (Grid size = 40, Decision Tree depth = 4).

As obstacle density increases, both models experience a sharp decline in completion rate and total reward, reflecting the growing difficulty of coordination and path planning. The DT closely follows the neural policy across all densities, with similar degradation trends and overlapping confidence intervals. This consistency suggests that the symbolic surrogate generalizes adequately to more constrained environments, despite its reduced expressiveness.

Effect of Grid Size

Finally, Table 5 investigates how scaling the environment affects performance, with obstacle density fixed at zero and DT depth set to 4.

Both models demonstrate strong scalability across grid sizes. The neural policy maintains high completion rates as the environment grows, and the DT achieves nearly identical results. These findings indicate that the distilled policy preserves the neural model's generalization ability across spatial scales, supporting the viability of interpretable surrogates for large-scale MAPF tasks.

Grid size	Model	Completion rate	Total reward
10	NN	0.627 ± 0.227	-1194 ± 1099
	DT	0.627 ± 0.239	-1212 ± 1154
40	NN	0.921 ± 0.050	-473 ± 364
	DT	0.885 ± 0.064	-551 ± 374
80	NN	0.940 ± 0.024	-623 ± 390
	DT	0.952 ± 0.051	-624 ± 425

Table 5: Sensitivity to grid size (Obstacle density = 0, Decision Tree depth = 4).

Discussion and Conclusion

This study investigated the faithfulness and robustness of a symbolic surrogate model for neural policies in multi-agent pathfinding (MAPF). The proposed decision tree achieved high fidelity to the neural policy (accuracy up to 0.89), yet its task-level performance remained consistently below that of the original network. Sensitivity analyses across tree depth, obstacle density, and grid size revealed that deeper trees partially recover the neural model's efficiency, while generalization to unseen environmental scales remains limited.

These findings underline a trade-off between interpretability and behavioral expressiveness in symbolic approximations of learned policies. In safety-critical or explainability-oriented control systems, such surrogates may provide valuable transparency while preserving part of the original functionality. Future work should focus on standardizing the evaluation of symbolic proxies for neural policies, particularly in dynamic and multi-agent control settings, to better quantify the trustworthiness and reproducibility of interpretable AI controllers.

References

Bastani, O.; Kim, C.; and Bastani, H. 2017. Interpreting blackbox models via model extraction. *arXiv preprint arXiv:1705.08504*.

Boyarski, E.; Felner, A.; Stern, R.; Sharon, G.; Betzalel, O.; Tolpin, D.; and Shimony, E. 2015. Icbs: The improved conflict-based search algorithm for multi-agent pathfinding. In *Proceedings of the International Symposium on Combinatorial Search*, volume 6, 223–225.

Breiman, L.; Friedman, J.; and Stone, C. J. 2017. *Classification and regression trees*. Chapman and Hall/CRC.

Cohen, L.; and Koenig, S. 2016. Bounded suboptimal multi-agent path finding using highways. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence*, 3978–3979.

Coppens, Y.; Efthymiadis, K.; Lenaerts, T.; and Nowe, A. 2019. Distilling Deep Reinforcement Learning Policies in Soft Decision Trees. In *IJCAI 2019 Workshop on Explainable Artificial Intelligence*, 1–6.

Craven, M. W. 1996. *Extracting comprehensible models from trained neural networks*. The University of Wisconsin-Madison.

Foerster, J. N.; Farquhar, G.; Afouras, T.; Nardelli, N.; and Whiteson, S. 2018. Counterfactual Multi-Agent Policy Gradients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2974–2982.

Hinton, G.; Vinyals, O.; and Dean, J. 2015. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*.

Li, J.; Chen, Z.; Harabor, D.; Stuckey, P. J.; and Koenig, S. 2021. Anytime multi-agent path finding via large neighborhood search. In *International Joint Conference on Artificial Intelligence 2021*, 4127–4135. Association for the Advancement of Artificial Intelligence (AAAI).

Phan, T.; Driscoll, J.; Romberg, J.; and Koenig, S. 2024. Confidence-Based Curriculum Learning for Multi-Agent Path Finding. In *Proceedings of the 23rd International Conference on Autonomous Agents and Multiagent Systems*, 1558–1566.

Rashid, T.; Samvelyan, M.; De Witt, C. S.; Farquhar, G.; Foerster, J.; and Whiteson, S. 2020. Monotonic value function factorisation for deep multi-agent reinforcement learning. *Journal of Machine Learning Research*, 21(178): 1–51.

Rusu, A. A.; Colmenarejo, S. G.; Gulcehre, C.; Desjardins, G.; Kirkpatrick, J.; Pascanu, R.; Mnih, V.; Kavukcuoglu, K.; and Hadsell, R. 2015. Policy distillation. *arXiv preprint arXiv:1511.06295*.

Sartoretti, G.; Kerr, J.; Shi, Y.; Wagner, G.; Kumar, T. S.; Koenig, S.; and Choset, H. 2019. Primal: Pathfinding via reinforcement and imitation multi-agent learning. *IEEE Robotics and Automation Letters*, 4(3): 2378–2385.

Sharon, G.; Stern, R.; Felner, A.; and Sturtevant, N. 2012. Conflict-Based Search For Optimal Multi-Agent Path Finding. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26, 563–569.

Silver, D. 2005. Cooperative pathfinding. In *Proceedings of the aaai conference on artificial intelligence and interactive digital entertainment*, volume 1, 117–122.

Standley, T. 2010. Finding optimal solutions to cooperative pathfinding problems. In *Proceedings of the AAAI conference on artificial intelligence*, volume 24, 173–178.

Stern, R.; Sturtevant, N.; Felner, A.; Koenig, S.; Ma, H.; Walker, T.; Li, J.; Atzmon, D.; Cohen, L.; Kumar, T.; et al. 2019. Multi-agent pathfinding: Definitions, variants, and benchmarks. In *Proceedings of the International Symposium on Combinatorial Search*, volume 10, 151–158.

Sukhbaatar, S.; Fergus, R.; et al. 2016. Learning multiagent communication with backpropagation. *Advances in neural information processing systems*, 29.

Sunehag, P.; Lever, G.; Gruslys, A.; Czarnecki, W. M.; Zambaldi, V.; Jaderberg, M.; Lanctot, M.; Sonnerat, N.; Leibo, J. Z.; Tuyls, K.; et al. 2017. Value-decomposition networks for cooperative multi-agent learning. *arXiv preprint arXiv:1706.05296*.

Verma, A.; Murali, V.; Singh, R.; Kohli, P.; and Chaudhuri, S. 2018. Programmatically interpretable reinforcement learning. In *International conference on machine learning*, 5045–5054. PMLR.

Yu, J.; and LaValle, S. M. 2013. Planning optimal paths for multiple robots on graphs. In *2013 IEEE International Conference on Robotics and Automation*, 3612–3617. IEEE.

Zhang, K.; Yang, Z.; and Başar, T. 2021. Multi-agent reinforcement learning: A selective overview of theories and algorithms. *Handbook of reinforcement learning and control*, 321–384.