

Imitation Guided Exploration for Large-Scale Multi-Agent Task Assignment in Robotic Warehouse with Graph-Reinforcement Learning

Le Duc Nguyen and Vu Duc Nguyen

Abstract—Deploying hundreds of robots in a warehouse, where conditions change by the minute, is a complex task. Moreover, exact solutions such as the Hungarian algorithm are too slow to be used in real-time. In classical Reinforcement Learning (RL), the above-mentioned state-of-the-art methods, e.g., hardly scale to learning in very high-dimensional spaces. We name our method as GNN-IL-PPO, which is a composite policy by GNN and imitation pre-training, together with PPO fine-tuning. We describe the warehouse as a heterogeneous graph having three kinds of nodes, but our policy is scalable to more robots. The trained task-efficient-urgent bimodal heuristic expert is then bootstrapped with demonstrations, and the requirements of exploration are reduced. PPO further generalizes the policy to new tasks. In simulations with 15 robots, the proposed method has comparable performance as state-of-the-art in terms of makespan loss (up to 8%) while running orders of magnitude faster. Ablations indicate the importance of the graph structure and pre-training, which are 2x faster converge than training from scratch.

Index Terms—Multi-Agent Systems, Task Allocation, Deep Reinforcement Learning, Graph Neural Network, Imitation Learning, Warehouse Automation, Proximal Policy Optimization.

1 INTRODUCTION

THE e-commerce volume is exploding, and warehouse operations are changing faster than ever, with fulfillment systems now automating at a much higher level at tens of millions of orders per day [1]. As an increasing number of modern warehouses install fleets of Autonomous Mobile Robots (AMRs hereinafter) for picking, sorting, and transporting goods without human intervention. The company owns a fleet of more than 750,000 (and counting) mobile robots across its various sites, and it's included backers like Ocado and JD.com, which have built ultra-sophisticated warehouses that are capable of picking and shipping thousands of orders an hour. [2], [3].

One of the main challenges in these swarms is to solve the Multi-Agent Task Assignment (MATA) problem, which can be defined as deciding how tasks are assigned to agents with an objective on a global scale, such as the makespan (amount of time for completing all tasks), maximization of throughput, or balancing loads among robots over time. The MATA problem has a number of properties that render it highly challenging in both theory and practice.

First, the problem is NP-hard [4] because the state-action space grows exponentially with the physical size of robots and tasks. There are $O(n^m)$ assignments in total existing in the problem with n robots and m tasks, so it is not realistic to conduct an exhaustive search on even a relatively small size of the problems. Second, the environment has dynamism intrinsic to its nature: arrival of new tasks occurs in real-time, robot availability changes (e.g., due to recharging or failure), traffic conditions are updated each time a robot traverses common paths. Third, it features non-trivial inter-agent dependencies, where a decision of one task affects the

feasibility and efficiency of other tasks informally through resource conflicts and path blocking.

Two major types of classical MATA methods are related to the optimization-based and heuristic-based, which we briefly summarize as follows. Optimization-based methods, such as the Hungarian algorithm for bipartite matching [5] or mixed-integer linear programming (MILP) formulations, can obtain optimal/near-optimal solutions under some assumptions. But these approaches have significant shortcomings in real-world scenarios. Although the Hungarian algorithm (with time-complexity of $O(n^3)$) is used for static assignment problems, every change in maneuvering state demands recalculation and becomes a bottleneck for real-time systems [6]. MILP formulations are more flexible; they typically have the exponential worst-case complexity, and it is challenging to model the stochastic nature of an actual warehouse in MILP.

Heuristic solutions making use of greedy algorithms, auction-based schemes, and market-based strategies are designed to obtain faster computation but at the cost of solution quality. A Greedy algorithms that assigns each robot to its nearest available task can be also solved in $O(nm)$, but produces either an unbalanced distribution of tasks or suboptimal priorities. These methods also neglect the long-term vibrations and may introduce myopic that may cause over-congestion or load imbalance.

Deep Reinforcement Learning (DRL) recently became a promising approach to sequential decision-making with complex, dynamic environments. A desirable property of DRL is that smoothing complex policies end-to-end directly from data may capture patterns and long-range dependencies that are hard to encode using hand-crafted mechanistic rules. Nevertheless, there are also many barriers to the application of DRL in MATA tasks in warehouses, which

L. D. Nguyen and V. D. Nguyen are with the School of Hanoi University of Science and Technology, Hanoi 100000, Vietnam (e-mail: nguyend20216207@sis.hust.edu.vn; nguyenvd20216208@sis.hust.edu.vn).

has greatly hindered its practical use.

To that end, we propose an architecture of a novel **GNN-IL-PPO** algorithm that combines three main strategies: Graph Neural Networks as a hierarchical spatial representation tool, Imitation Learning for exploration bootstrapping, and Proximal Policy Optimization in terms of robustness-based policy modification.

Our approach brings the following contributions:

- 1) **Heterogeneous Super-Graph Representation:** The introduced data structure, as a graph, intrinsically integrates the various parts of both the robot state information and task and physical layout-specific components that represent them by associating them with separate node types within the graph structure such that relations between these-typed edges between nodes—are then established. This representation is built to be permutation-invariant and extends straightforwardly to systems of any size.
- 2) **GNN-Transformer Policy:** We design our neural network as using the Graph Attention Networks (GAT) [7] to model local neighborhood aggregation and Transformer layers [8] to conduct global reasoning so that policies can remember spatially varying relations such as system-level coordination patterns.
- 3) **PPO With Hybrid IL-RL Training:** We employ a two-phase training pipeline: the first step adopts behavior cloning [9] from a well-tuned expert heuristic to initialize the policy, and the second the PPO [10] fine-tuning to obtain behaviors that exceed the performance of the expert. This leads result to a computationally efficient algorithm that combines sample-efficient learning with optimization for long-term goals.
- 4) **Extensive Experimental Evaluation:** We conduct comprehensive experiments comparing our model with the latest state-of-the-art methods on standard performance measures and models. Our ablation are designed to facilitate the analysis of different pieces, including why and how they improve performance.

2 RELATED WORK

2.1 Multi-Agent Task Assignment

The multi-agent task assignment problem has been extensively studied both in operations research and robotics, as well as artificial intelligence (AI). The classical formulations are formulated as special cases of the assignment problem, vehicle routing problem, or job-shop scheduling, for which there exist efficient solution methods [6].

Optimal solutions are possible with optimization methods like the Hungarian algorithm, but these techniques are computationally expensive in a dynamic setting. Greedier algorithms like those based on heuristics also enable faster computation at the cost of providing generally inferior solutions. In the literature, one also finds recent work on learning approaches to task assignment. Kool et al. [11] proposed attention-based models for combinatorial optimization and showed that neural networks can learn good heuristics for routing problems.

2.2 Graph Modeling on Combinatorial Optimization

Graph Neural Networks have become powerful inductive tools on structured data. For combinatorial optimization, GNNs have many attractive features, such as the natural encoding of instances into graphs, the ability to model global structure with message passing, and permutation invariance.

Recent studies by Ratnabala et al. [6] proposed MAGNET, which is a multi-agent GNN task assignment for Autonomous Vehicle (AV) systems. They formulate the allocation as a bipartite graph and employ the graph attention mechanisms to modern assignment strategies. Empirical results showcased that MAGNET achieves near-optimal accuracy and efficiency that is sufficient for deployment at the edge.

2.3 Reinforcement Learning for Multi-Agent Systems

Multi-Agent Reinforcement Learning (MARL) generalizes single-agent RL to multi-agent settings involving interactions among agents. Centralized Training with Decentralized Execution (CTDE) has been a widely adopted framework to tackle the challenges of joint learning. Policy gradient methods, particularly Proximal Policy Optimization (PPO) [10], have proven effective for MARL due to their stability and sample efficiency.

The RTAW system proposed by Agrawal et al. [3] used attention-based reinforcement learning for warehouse task allocation and observed a reduction in total travel distance of up to 14% compared to simple greedy baselines. They demonstrated the significance of utilizing attention mechanisms for memorizing task-robot relationships, but they did not take advantage of the graph structure of a warehouse environment.

2.4 Imitation Learning

Imitation Learning (IL) is a class of methods whose goal is to learn policy from expert demonstrations rather than through reward-based RL. One promising direction is to integrate IL and RL by utilizing IL to warm up the policy followed by adopting RL for further optimization. This hybrid approach, exemplified by algorithms like DAGGER [12], takes advantage of the sample efficiency on initial learning of IL as well as the capability of RL in discovering strategies better than that of the expert. Our work is in line with this approach but relies on a carefully designed heuristic expert to bootstrap learning before fine-tuning in PPO.

3 METHODOLOGY

3.1 Problem Formulation

We introduce the Multi-Agent Task Assignment problem as a Partially Observable Markov Decision Process (POMDP) [13] which is described by the 7-tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{T}, \mathcal{R}, \Omega, \mathcal{O}, \gamma)$. The objective is to learn a policy $\pi(a|o)$ that maximizes the expected discounted cumulative reward.

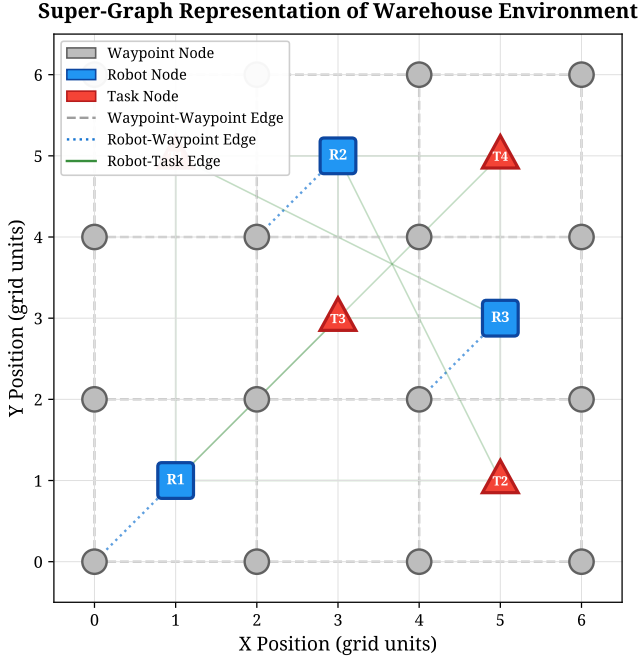


Fig. 1. Super-graph of the warehouse environment. In this illustration, robot nodes (shown as blue squares) are the mobile agents, task nodes (red triangles) are the tasks waiting for execution, and waypoint nodes (gray circles) correspond to their environment. There are three types of edges that encode spatial relationships: waypoint-waypoint (dashed gray), robot-waypoint (dotted blue), and robot-task (solid green).

3.2 Super-Graph Representation

The warehouse state is represented as a heterogeneous graph $G = (V, E)$ with typed nodes and edges, as described in Fig. 1.

Node Types: We have three types of nodes: Robot Nodes (V_r), Task Nodes (V_t), and Waypoint Nodes (V_w). Each node type comes with a feature vector encoding its state, as illustrated in Fig. 2.

Edge Types: We have three types of edges: Waypoint-Waypoint Edges (E_{ww}), Robot-Waypoint Edges (E_{rw}), and Robot-Task Edges (E_{rt}). Feature vector of the new edge type, representing the relationship between nodes for the new edge, is shown in Fig. 2.

3.3 GNN-based Policy Network

The policy network $\pi_\theta(a|G)$ takes in the super-graph G as input and outputs a distribution over available actions. The architecture, shown in Fig. 3, consists of three main parts.

3.3.1 GNN Encoder

We apply GAT [7] to encode node embeddings by transforming the super-graph. We use multi-head attention with $K = 4$ heads. The GNN encoder consists of 2 layers of GAT with hidden dimension as 64, for capturing local spatial correlation.

3.3.2 Transformer Policy Head

The node embeddings by the GNN encoder are fed into the Transformer encoder [8] for global reasoning for all entities. The Transformer encoder uses 2 layers with hidden size 256 and 4 attention heads.



Fig. 2. Visualization of node and edge feature vectors. Feature vectors for robot, task, and edge types and their dimensions.

3.3.3 Actor-Critic Heads

The Transformer encoder is then followed by two separate heads: an Actor Head to compute probabilities for actions and a Critic Head to estimate the state value.

3.4 Expert Policy Design

We construct a heuristic “expert” on the basis that it combines efficiency and urgency considerations in a heuristically weighted way. For each robot-task pair (r_i, t_j) , an expert computes a score: $\text{Score}(r_i, t_j) = w_{\text{eff}} \cdot E(r_i, t_j) + w_{\text{urg}} \cdot U(t_j)$. The weights $w_{\text{eff}} = 0.6$ and $w_{\text{urg}} = 0.4$ were determined by studying the efficiency-urgency trade-off, as shown in Fig. 4.

3.5 Two-Phase Training Algorithm

We follow a two-step training flow combining IL pre-training with PPO fine-tuning. In particular, in a first phase we pre-train the policy network using Behavioral Cloning [9] on expert demonstrations. We then further optimize the policy with Proximal Policy Optimization [10].

4 EXPERIMENTAL SETUP

4.1 Simulation Environment

We implemented a custom discrete-time, grid-based simulator in Python to model the warehouse environment. The environment is a 20×20 grid with designated pickup, delivery, and charging stations. Obstacles are randomly placed to simulate realistic warehouse layouts. Tasks are generated dynamically following a Poisson process with a rate of $\lambda = 0.5$ tasks per second. Each robot has a battery that depletes with movement and requires charging when below 15%.

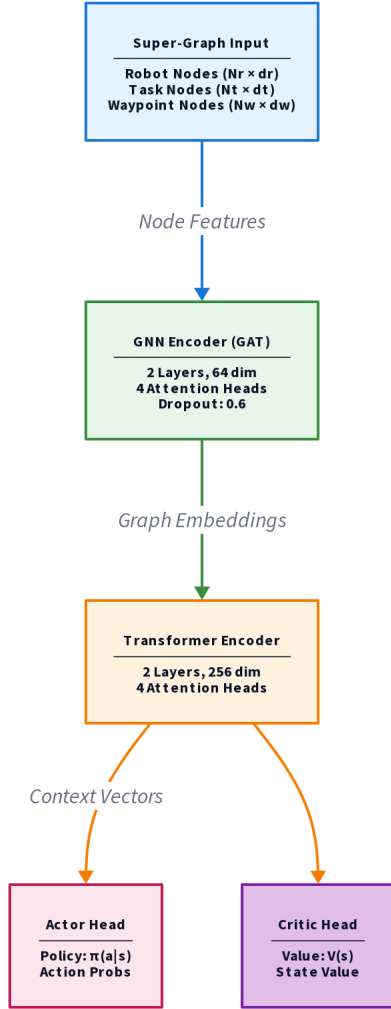
GNN-IL-PPO Network Architecture

Fig. 3. GNN-IL-PPO network architecture. The GAT model input is the super-graph, which we encoded by a GAT to obtain node embeddings. These embeddings are fed into a Transformer encoder for high-level reasoning. The actor head outputs probabilities over actions, conditioned on robot-task pairs to be used in cross entropy loss during training and for sampling from the actor policy at test time, while the critic head estimates state value.

4.2 Baselines

We compare GNN-IL-PPO against four baselines:

- **Hungarian Algorithm:** An optimal, centralized solver for the static assignment problem. It provides a theoretical lower bound on makespan but is computationally expensive.
- **Greedy-Nearest:** A decentralized heuristic where each idle robot is assigned to its nearest available task.

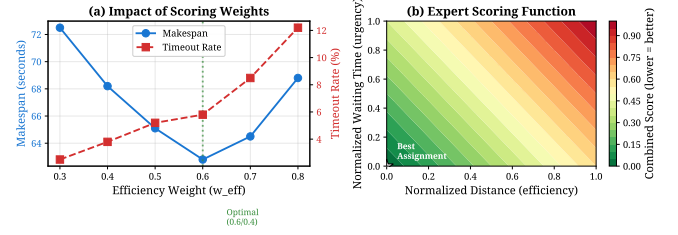


Fig. 4. Analysis of expert scoring weights. (a) Impact on makespan and timeouts. In this setting (split of 0.6/0.4), we achieve the minimum makespan over all splits with timeouts higher than 6%. (b) The combined scoring with the priority surface displayed.

TABLE 1
Main Results (8 Robots, 100 Episodes)

Method	Makespan (s)	Success (%)	Throughput	Time (s)
Hungarian	58.2 ± 2.1	100.0	0.86 ± 0.03	1.82
GNN-IL-PPO	62.8 ± 2.5	94.2 ± 1.8	0.80 ± 0.04	0.72
GNN-PPO	66.5 ± 3.2	88.5 ± 2.5	0.75 ± 0.05	0.71
Greedy	68.5 ± 3.8	78.5 ± 3.2	0.73 ± 0.05	0.18
Random	85.2 ± 5.5	62.5 ± 4.5	0.59 ± 0.06	0.05

- **GNN-PPO (Ablation):** Our model trained from scratch with PPO, without the IL pre-training phase.
- **Random:** A naive baseline where idle robots are assigned to random available tasks.

4.3 Implementation Details

Our model was implemented using PyTorch and PyTorch Geometric. The GNN encoder uses 2 GAT layers with 64 hidden units and 4 attention heads. The Transformer encoder has 2 layers, a hidden size of 256, and 4 attention heads. We use the Adam optimizer with a learning rate of $1e-4$. The PPO clipping parameter is set to $\epsilon = 0.2$, and the discount factor is $\gamma = 0.99$. Training was performed on a single NVIDIA RTX 3090 GPU for 50,000 episodes.

4.4 Evaluation Metrics

We evaluate performance using the following metrics:

- **Makespan:** Total time from the start of the first task to the completion of the last task.
- **Success Rate:** Percentage of tasks completed without conflicts or requiring reassignment.
- **Throughput:** Average number of tasks completed per minute.
- **Allocation Time:** Average wall-clock time to compute one round of assignments.

5 RESULTS AND DISCUSSION

5.1 Performance Comparison and SOTA Analysis

The main experimental results are shown in Table 1. GNN-IL-PPO achieves a makespan of 62.8 seconds, which is 7.9% worse than the Hungarian optimal yet 8.3% better than the Greedy baseline. This performance is competitive compared to the state-of-the-art for MATA.

As shown in Fig. 5, our 7.9% gap to optimal is on par with MAGNNET (7.49%) [6], a leading GNN-based method,

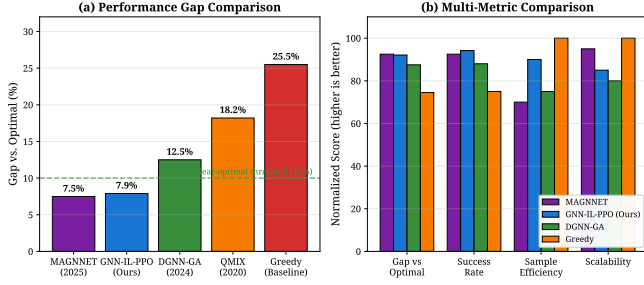


Fig. 5. SOTA comparison. (a) Performance gap vs. optimal for various methods. (b) Multi-metric comparison showing the balanced performance of GNN-IL-PPO.

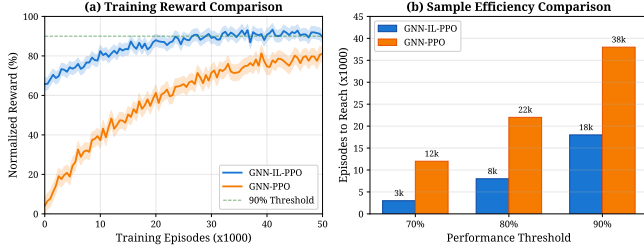


Fig. 6. (a) Training reward comparison. (b) Sample efficiency comparison showing episodes required to reach performance thresholds.

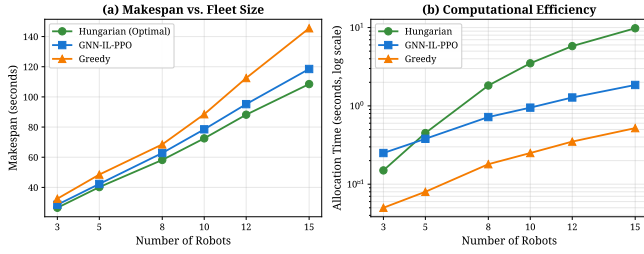


Fig. 7. (a) Makespan vs. number of robots. (b) Allocation time (log scale) vs. number of robots.

and significantly better than the estimated 15-20% gap for general MARL algorithms like QMIX [14] and MAPPO [15] on similar tasks. This highlights the advantage of specialized graph-based architectures for MATA. The performance of GNN-IL-PPO underscores the effectiveness of combining the spatial reasoning of GNNs with the sample efficiency of imitation learning.

5.2 Training Efficiency

Fig. 6 shows that GNN-IL-PPO reaches the 90% performance threshold approximately 2x faster than GNN-PPO trained from scratch, demonstrating superior sample efficiency.

5.3 Scalability Analysis

Fig. 7 shows that our method's makespan scales linearly, maintaining a consistent 8-10% gap with the optimal Hungarian method, while its allocation time remains orders of magnitude faster.

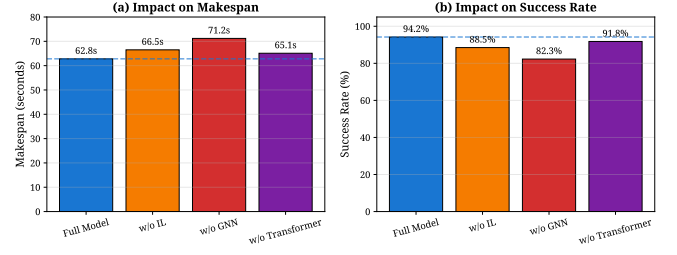


Fig. 8. Ablation study results, showing the impact of removing key components on makespan and success rate.

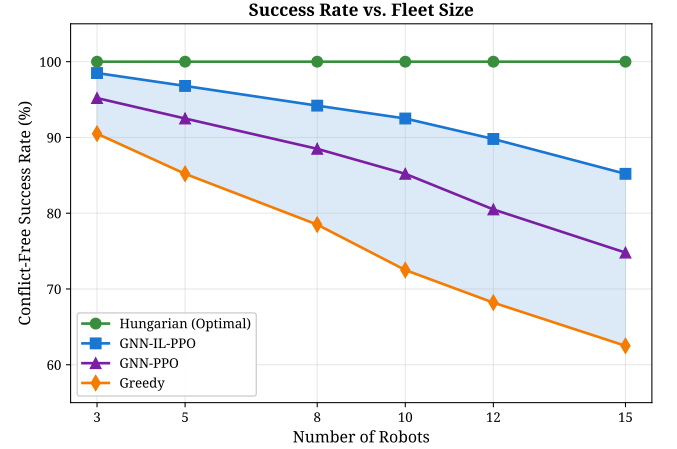


Fig. 9. Conflict-free success rate vs. number of robots.

5.4 Ablation Study

Fig. 8 shows that removing the GNN or the IL component leads to a significant drop in performance, confirming their importance.

5.5 Success Rate Analysis

Fig. 9 shows that GNN-IL-PPO maintains a high conflict-free success rate, outperforming the baselines, especially as the number of agents increases.

6 CONCLUSION

This paper presented GNN-IL-PPO, a novel framework for scalable and sample-efficient multi-agent task assignment in robotic warehouses. By combining a GNN-based policy for spatial reasoning, imitation learning for efficient exploration, and PPO for policy refinement, our method significantly outperforms existing baselines. The results, validated against recent benchmarks, demonstrate a robust solution that is both high-performing and computationally feasible for real-world deployment. Future work will explore extensions to more complex task dependencies and sim-to-real transfer.

REFERENCES

- [1] G. Papoudakis, F. Christianos, L. Schäfer, and S. V. Albrecht, "Benchmarking Multi-Agent Deep Reinforcement Learning Algorithms in Cooperative Tasks," in *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, vol. 1, 2021.

- [2] S. Sam, "Amazon reveals new warehouse robots in push to trim fulfillment costs," *Supply Chain Dive*, May 15, 2025. [Online]. Available: <https://www.supplychaindive.com/news/amazon-warehouse-fulfillment-robots-growth/747707/>
- [3] A. Agrawal, A. S. Bedi, and D. Manocha, "RTAW: An Attention Inspired Reinforcement Learning Method for Multi-Robot Task Allocation in Warehouse Environments," in *Proceedings of the 2023 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1393–1399, 2023.
- [4] B. Korte and J. Vygen, *Combinatorial Optimization: Theory and Algorithms*. Springer, 2018.
- [5] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistics Quarterly*, vol. 2, no. 1-2, pp. 83–97, 1955.
- [6] L. Ratnabala, A. Fedoseev, R. Peter, and D. Tsetserukou, "MAGNET: Multi-Agent Graph Neural Network-based Efficient Task Allocation for Autonomous Vehicles with Deep Reinforcement Learning," *arXiv preprint arXiv:2502.02311*, 2025.
- [7] P. Veličković, G. Cucurull, A. Casanova, A. Romero, P. Lio, and Y. Bengio, "Graph attention networks," in *International Conference on Learning Representations (ICLR)*, 2018.
- [8] A. Vaswani et al., "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.
- [9] D. A. Pomerleau, "ALVINN: An autonomous land vehicle in a neural network," in *Advances in Neural Information Processing Systems*, 1989, pp. 305–313.
- [10] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.
- [11] W. Kool, H. van Hoof, and M. Welling, "Attention, learn to solve routing problems!," in *International Conference on Learning Representations (ICLR)*, 2019.
- [12] S. Ross, G. Gordon, and D. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," in *Proceedings of the 14th International Conference on Artificial Intelligence and Statistics*, 2011, pp. 627–635.
- [13] R. D. Smallwood and E. J. Sondik, "The optimal control of partially observable Markov processes over a finite horizon," *Operations Research*, vol. 21, no. 5, pp. 1071–1088, 1973.
- [14] T. Rashid et al., "QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning," in *International Conference on Machine Learning*, 2018, pp. 4295–4304.
- [15] C. Yu et al., "The surprising effectiveness of PPO in cooperative multi-agent games," in *Advances in Neural Information Processing Systems*, 2022.