PATH-NM: Learning Orthogonal N:M Sparse Patterns via Differentiable Transformation for Efficient Heterogeneous MARL

Zhenfeng Su[1,2], Zhen Liu[1,2,*], and Zhiming Zhou[1]
[1]The Key Laboratory of Cognition and Decision Intelligence for Complex Systems,
Institute of Automation, Chinese Academy of Sciences
[2]School of Artificial Intelligence, University of Chinese Academy of Sciences, China
[*]Corresponding author
{suzhenfeng2023, liuzhen, zhiming.zhou}@ia.ac.cn

***Abstract*—Semi-structured $N:M$ sparsity enables hardware acceleration on modern GPUs, but its use in heterogeneous multi-agent reinforcement learning (MARL) remains largely unexplored. Existing parameter-sharing methods either ignore hardware constraints or encourage policy homogenization, while strict $N:M$ sparsity introduces additional challenges due to non-stationary activations, discrete constraints, and agent heterogeneity. We propose PATH-NM (Pattern-Aware Transformation for Hardware-Efficient $N:M$ Sparse MARL), a sparsity-aware framework that learns agent-specific $N:M$ sparse subnetworks on top of a shared backbone. PATH-NM reformulates pruning as pattern distribution learning and combines dynamic activation-aware scoring, differentiable pattern transformation, and orthogonal diversity regularization. This design supports end-to-end optimization, adapts to non-stationary RL dynamics, and explicitly promotes structural diversity across agents. Experiments on MPE and SMACv2 show that PATH-NM consistently outperforms representative parameter-sharing baselines in both sample efficiency and final performance, while remaining fully compatible with hardware-accelerated $N:M$ sparse inference.**

***Index Terms*—Multi-Agent Reinforcement Learning, Heterogeneous Agents, N:M Sparsity, Parameter Sharing, Hardware-Efficient Learning**

I. Introduction

Multi-agent reinforcement learning (MARL) has achieved strong performance in cooperative decision-making problems such as game playing, autonomous driving, and network control [1]–[3]. In particular, the centralized training with decentralized execution (CTDE) paradigm has become a standard framework for improving training stability and sample efficiency [4]. Representative methods include COMA, MADDPG, QMIX [5]–[7]. However, most CTDE-based methods assume homogeneous agents and rely on fully shared policy architectures, which is often unrealistic in practical settings.

In many real-world tasks, agents differ in capabilities, roles, or objectives. For example, units in StarCraft micromanagement have different attack ranges, attributes, and tactical functions [8]. In such cases, full parameter sharing can lead to policy homogenization and limit specialization. Heterogeneous MARL therefore requires a better balance between shared representations and agent-specific adaptation.

A promising direction is to derive agent-specific subnetworks from a shared model. Existing parameter-sharing optimization methods, such as SePS [9], SNP [10], and Kaleidoscope [11], relax full sharing through clustering, pruning, or learnable masks. Although effective, these approaches largely overlook hardware efficiency, and the induced sparsity patterns often do not translate into practical acceleration on modern accelerators.

Recent NVIDIA GPUs provide native support for semi-structured $N:M$ sparsity (e.g., 2:4 sparsity) through Sparse Tensor Cores. This creates an opportunity to obtain both policy specialization and efficient inference. However, directly applying $N:M$ sparsity to MARL is nontrivial for three reasons: (1) MARL exhibits highly non-stationary activation distributions due to continual policy updates and inter-agent interactions; (2) strict $N:M$ constraints are discrete and are usually imposed by non-differentiable Top-K operators; and (3) enforcing similar sparse structures across agents can further aggravate policy homogenization.

To address these issues, we propose **PATH-NM**, a framework for learning hardware-efficient $N:M$ sparse policies in heterogeneous MARL. Instead of treating pruning as static weight removal, PATH-NM formulates it as a *pattern distribution learning* problem. The framework consists of three components: (i) *dynamic heterogeneous scoring*, which tracks online activation statistics and modulates them in an agent-specific manner; (ii) *differentiable pattern transformation*, which uses Gumbel-Softmax to learn valid $N:M$ sparse patterns while preserving differentiability; and (iii) *orthogonal diversity learning*, which explicitly discourages structural overlap among agents.

Our main contributions are as follows:

- We identify the key obstacles to applying semi-structured $N:M$ sparsity in heterogeneous MARL, namely non-stationarity, non-differentiability, and policy homogenization.

This work was supported by the National Key Research and Development Program of China under Grant 2018AAA0102402.

- We propose PATH-NM, a sparsity-aware MARL framework that jointly enables hardware-efficient inference, end-to-end optimization, and agent specialization.
- We show that learning sparse *patterns*, rather than sparse *weights*, provides an effective mechanism for heterogeneous policy learning under strict hardware constraints.

II. Related Work

A. Pruning and Semi-structured Sparsity

Model pruning can be broadly divided into structured, unstructured, and semi-structured sparsity. Structured pruning removes entire channels, heads, or layers and is generally hardware friendly, but often sacrifices flexibility. Unstructured pruning removes individual weights and typically preserves accuracy better, but its irregular sparsity is difficult to accelerate on commodity GPUs. Semi-structured sparsity, such as $N:M$ sparsity, provides a practical compromise by enforcing regular local patterns that are directly supported by modern accelerators [12], [13]. This makes it especially attractive for deployment-oriented learning systems. Recent pruning methods such as SparseGPT and WANDA further highlight the effectiveness of data-aware sparsification, although they are developed primarily for supervised or language-model settings [14], [15]. Unlike prior sparsification methods developed for supervised learning, our setting additionally requires robustness to non-stationary activations and explicit support for agent-level specialization.

B. Heterogeneous MARL

A central challenge in heterogeneous MARL is balancing parameter sharing with policy diversity. Non-sharing methods preserve agent specialization but scale poorly. Role-based methods improve this trade-off by introducing latent roles or subtasks. More closely related to our work are parameter-sharing optimization approaches, such as SePS [9], SNP [10], and Kaleidoscope [11], which derive agent-specific subnetworks or masks from a shared backbone. However, these methods do not explicitly target hardware-aligned semi-structured sparsity. PATH-NM builds on this line of work and introduces agent-specific $N:M$ sparse pattern learning with explicit diversity regularization.

III. Background

A. $N:M$ Sparsity

In $N:M$ sparsity, at most N values are kept in every contiguous group of M weights. Modern NVIDIA GPUs natively support 2:4 sparsity through Sparse Tensor Cores, making this pattern particularly attractive for efficient deployment. For a 4-dimensional weight block, exactly two entries remain non-zero, yielding $\binom{4}{2} = 6$ valid binary patterns. In practice, this blockwise structure is far more deployment-friendly than unstructured sparsity because it can be executed by vendor-provided sparse matrix kernels without requiring custom hardware or irregular memory access.

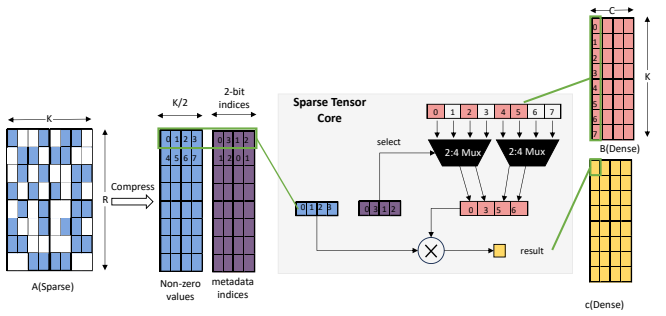

Fig. 1. Illustration of 2:4 sparsity and its mapping to Sparse Tensor Cores.

B. Problem Setting

We consider cooperative MARL under the CTDE paradigm. A fully cooperative task can be modeled as a Dec-POMDP [16], in which each agent receives a local observation and selects an action based on a decentralized policy, while training may access global information. Our goal is to learn hardware-efficient agent-specific subnetworks from a shared backbone so that execution remains decentralized, specialization is preserved, and the learned policies satisfy strict $N:M$ sparsity constraints.

IV. Method

PATH-NM learns hardware-efficient agent-specific subnetworks from a shared backbone. Given shared parameters W, the framework first estimates agent-dependent importance scores under non-stationary RL dynamics, then maps these scores to valid $N:M$ sparse patterns in a differentiable manner, and finally regularizes the resulting pattern distributions to promote structural diversity across agents. The three stages are tightly coupled: the scoring stage provides adaptive blockwise saliency, the transformation stage enforces hardware-compliant sparse masks, and the diversity stage prevents different agents from collapsing to the same subnetwork.

During training, the learned masks are applied to both actor and critic backbones, and gradients from the RL objective jointly update the shared weights and the agent-specific scoring parameters. In this way, PATH-NM does not treat pruning as a separate post-processing step; instead, sparse structure learning is integrated into policy optimization itself. During execution, each agent uses its own deterministic sparse subnetwork, which preserves decentralized inference while remaining compatible with hardware-supported sparse kernels.

A. Dynamic Heterogeneous Scoring

Because MARL training is highly non-stationary, importance estimation based on fixed calibration data is unreliable. We therefore extend the weight-and-activation criterion with online activation tracking and agent-specific modulation.

For agent k, the importance score tensor is defined as

$$\mathbf{S}^{(k)} = |W| \cdot \|\mathbf{X}\|_2, \tag{1}$$

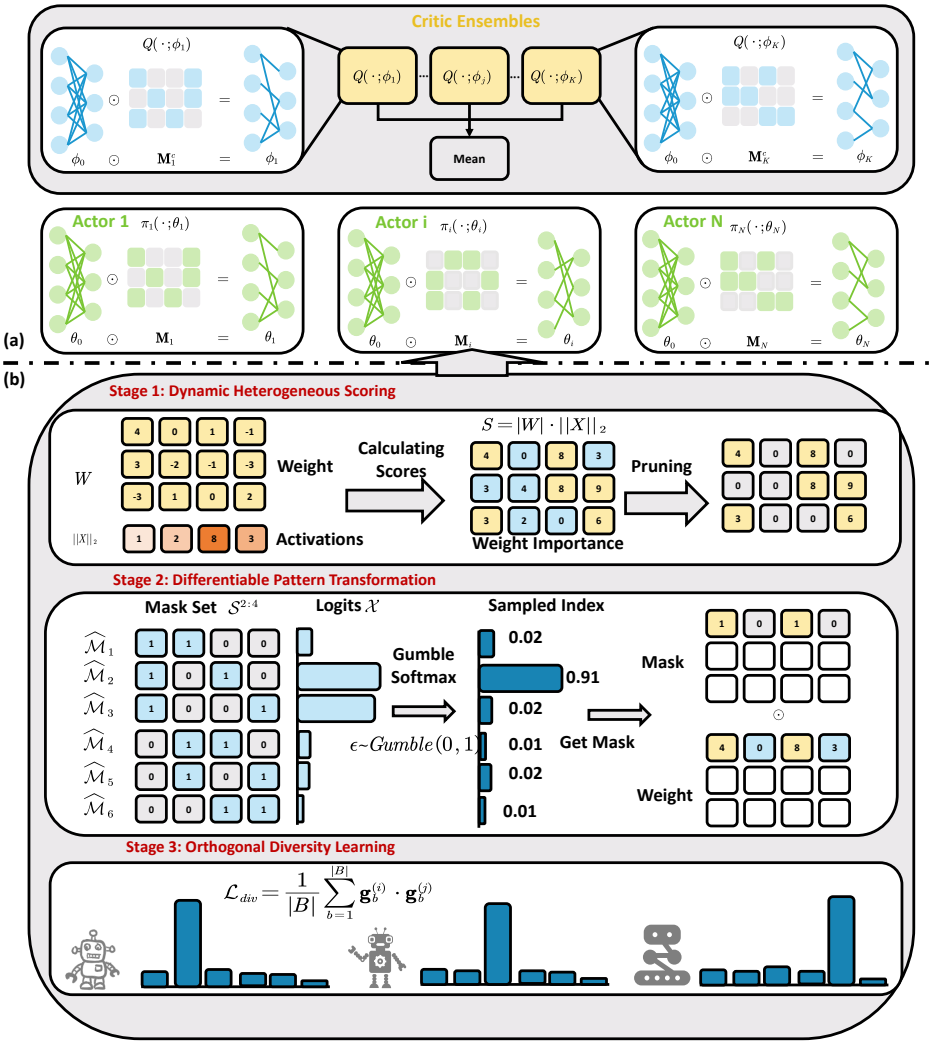

Fig. 2. Overview of **PATH-NM**. (a) Agents share a common backbone while learning agent-specific $N:M$ sparse subnetworks for both actors and critics. (b) PATH-NM proceeds in three connected stages: dynamic heterogeneous scoring produces block importance scores, differentiable pattern transformation maps these scores to valid sparse patterns, and orthogonal diversity learning discourages overlap across agents.

where $\|\mathbf{X}\|_2$ denotes the agent-specific activation magnitude. We compute

$$\|\mathbf{X}\|_2 = \boldsymbol{\mu}_{act}^{(k)} \odot \sigma(\boldsymbol{\alpha}^{(k)}), \tag{2}$$

where $\boldsymbol{\mu}_{act}^{(k)}$ is a smoothed activation statistic and $\boldsymbol{\alpha}^{(k)}$ is a learnable heterogeneity coefficient. Intuitively, $\boldsymbol{\mu}_{act}^{(k)}$ captures which features are currently useful for a given agent, while $\boldsymbol{\alpha}^{(k)}$ lets different agents rescale the same shared backbone in different ways.

Since offline calibration is unavailable in RL, activation statistics are updated online using exponential moving average (EMA):

$$\boldsymbol{\mu}_{act}^{(k)} \leftarrow \beta \boldsymbol{\mu}_{act}^{(k)} + (1-\beta)\,|\mathbf{X}_{net}^{(k)}|, \tag{3}$$

where $\mathbf{X}_{net}^{(k)}$ denotes intermediate activations and β is a momentum coefficient. This mechanism stabilizes importance estimation while allowing different agents to emphasize different subsets of shared parameters.

Compared with static pruning criteria, this design is better suited to RL because feature relevance can change substantially over the course of training as policies co-adapt. Online activation tracking therefore reduces the mismatch between current policy behavior and the sparse structures being learned, while the heterogeneity coefficients allow different agents to exploit the same shared backbone in distinct ways.

B. Differentiable Pattern Transformation

For 2:4 sparsity, each 4-element block must keep exactly two non-zero entries. Let $\mathbf{s} \in \mathbb{R}^4$ denote the importance scores of a weight block. We enumerate all valid patterns in a binary matrix $\mathbf{P} \in \{0,1\}^{6\times 4}$ and compute pattern logits as

$$\chi = \mathbf{P}\mathbf{s}^\top. \tag{4}$$

Each logit therefore measures the compatibility between a valid hardware-supported pattern and the current blockwise importance scores.

To preserve end-to-end differentiability, we apply the Gumbel-Softmax relaxation:

$$\mathbf{g} = \mathrm{Softmax}\left(\frac{\chi + \epsilon}{\tau}\right), \tag{5}$$

where $\epsilon \sim \mathrm{Gumbel}(0, 1)$ and τ is the temperature. In the forward pass, the most likely pattern is selected to obtain a hard one-hot vector $\mathbf{g}_{hard}$, and the corresponding sparse mask is reconstructed as

$$\mathbf{m}^{(k)} = \mathbf{g}_{hard}^{\top}\mathbf{P}. \tag{6}$$

This guarantees strict hardware-compliant $N : M$ sparsity. In the backward pass, gradients are propagated through the soft distribution $\mathbf{g}$ using the straight-through estimator. As a result, PATH-NM can jointly optimize the shared weights and the agent-specific scoring parameters without breaking the discrete sparsity constraints required by hardware.

An important advantage of this formulation is that optimization is performed over the discrete set of valid hardware-supported patterns rather than over unconstrained binary masks. This reduces the gap between training and deployment: every forward-pass mask is already executable by sparse hardware, and the Gumbel-Softmax relaxation provides a smooth surrogate for selecting among these valid candidates.

C. Orthogonal Diversity Learning

Agent-specific pattern learning alone does not prevent different agents from converging to similar sparse structures. To explicitly encourage specialization, we regularize the pattern distributions to be dissimilar.

Let $\mathbf{g}_b^{(i)}$ and $\mathbf{g}_b^{(j)}$ denote the soft pattern distributions of agents i and j at block b. We define the diversity loss as

$$\mathcal{L}_{div} = \frac{1}{|B|}\sum_{b=1}^{|B|}\mathbf{g}_b^{(i)} \cdot \mathbf{g}_b^{(j)}, \tag{7}$$

where B is the set of all blocks. Minimizing $\mathcal{L}_{div}$ discourages overlapping pattern selections and promotes structurally distinct subnetworks across agents. This term is especially important in heterogeneous MARL, where agents may otherwise collapse to similar solutions even when different behaviors are desirable.

Regularizing the soft pattern distributions is more stable than directly penalizing overlap between hard binary masks, since the latter would introduce additional discontinuities into optimization. The proposed formulation therefore encourages specialization before the final sparse structure becomes fully deterministic, which is especially beneficial under non-stationary multi-agent learning dynamics.

D. Sparse Critic Ensembles

PATH-NM applies the same masked parameterization to critics. Instead of maintaining fully independent critics, we use a shared critic backbone $\boldsymbol{\phi}_0$ together with K learnable sparse masks $\{\mathbf{M}_c^{(j)}\}_{j=1}^{K}$, so that the j-th critic is

$$Q^{(j)}(\cdot) = Q(\cdot; \boldsymbol{\phi}_j), \qquad \boldsymbol{\phi}_j = \boldsymbol{\phi}_0 \odot \mathbf{M}_c^{(j)}. \tag{8}$$

Each critic minimizes its own temporal-difference objective,

$$\mathcal{L}_c^{(j)} = \mathbb{E}_{(s_t,a_t,r_t,s_{t+1})\sim\mathcal{D}}\left[\left(Q^{(j)}(s_t, a_t) - y_t\right)^2\right], \tag{9}$$

where y_t is the target value computed from the target networks. Policy updates use the mean estimate of the ensemble. This design retains ensemble diversity while keeping the parameter count close to that of a single critic.

We further apply the same diversity regularization to critic masks to avoid structural collapse. During decentralized execution, each agent stores the shared backbone once together with a compact mask representation. For 2:4 sparsity, masks can be encoded as pattern indices, so the additional storage overhead is much smaller than maintaining independent dense policies.

E. Training Stabilization

To stabilize optimization, we anneal the Gumbel-Softmax temperature from 5.0 to 0.1, enabling a smooth transition from exploratory pattern sampling to near-deterministic selection. In addition, when consecutive policy updates exhibit excessive KL divergence, we softly reset the activation EMA using current batch statistics to avoid stale importance estimates. In all experiments, the total training objective combines the standard RL loss with diversity regularization:

$$\mathcal{L} = \mathcal{L}_{RL} + \lambda\mathcal{L}_{div}, \tag{10}$$

where λ controls the diversity–performance trade-off.

V. Experiments

A. Experimental Setup

We evaluate PATH-NM on two widely used MARL benchmarks: Multi-Agent Particle Environments (MPE) [6] and SMACv2 [17]. MPE provides cooperative continuous-control tasks with low-dimensional observations, while SMACv2 contains partially observable StarCraft II scenarios with heterogeneous unit types. These two benchmarks are complementary: MPE is useful for controlled analysis, whereas SMACv2 better reflects large-scale heterogeneous coordination.

We compare PATH-NM with four representative baselines: **NoPS** (no parameter sharing), **FuPS** (full sharing), **SNP** [10], and **Kaleidoscope** [11]. All methods are implemented within the same QMIX-based framework [7] for fair comparison. For MPE, we assign one mask per agent; for SMACv2, we assign one mask per agent type. We report mean performance with 95% confidence intervals over 5 random seeds on MPE and 3 random seeds on SMACv2, following common practice in prior MARL work.

To ensure fair comparison, all methods use the same backbone architecture, optimizer settings, replay configuration, and training budget unless a method requires a task-specific modification. This unified implementation isolates the effect

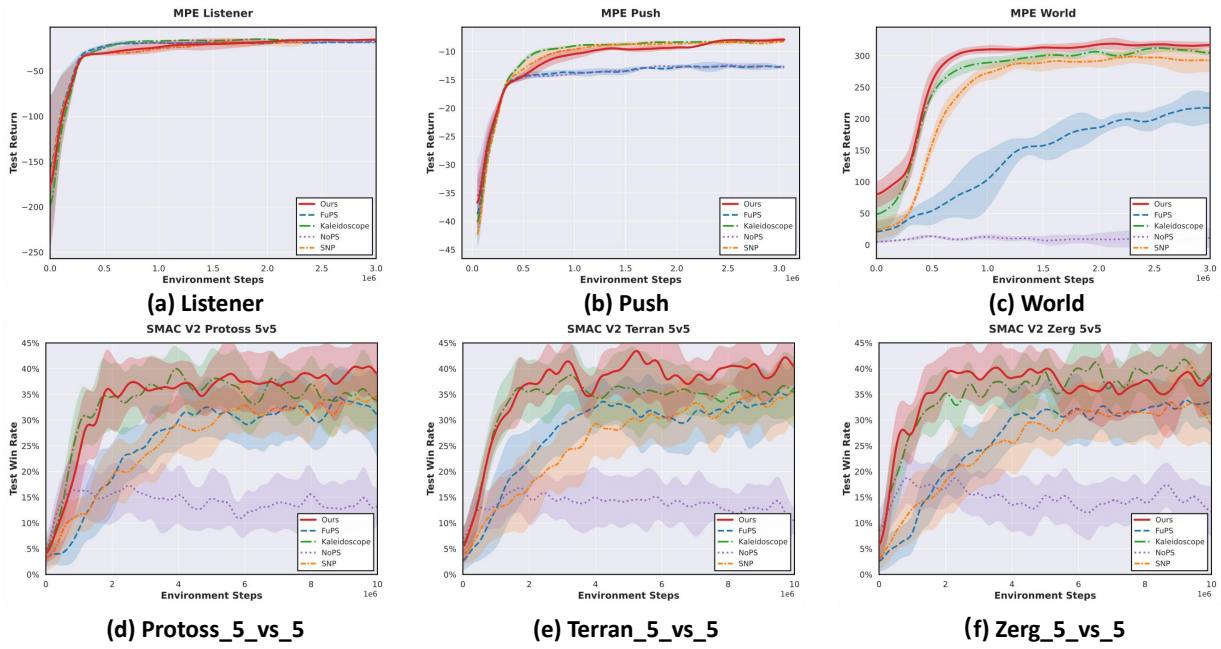

Fig. 3. Performance comparison on MPE and SMACv2 benchmarks.

TABLE I
METHODS COMPARED IN THE EXPERIMENTS.

Method	Sharing	Description
NoPS	None	Independent networks
FuPS	Full	Fully shared network
SNP	Partial	Randomly pruned subnetworks
Kaleidoscope	Partial	Learnable unstructured masks

of the sharing mechanism itself, rather than confounding it with differences in network capacity or training protocol.

B. Main Results

Fig. 3 compares PATH-NM with the baselines on MPE and SMACv2. Overall, PATH-NM consistently achieves the best or near-best performance while showing faster convergence.

On MPE, PATH-NM matches or slightly outperforms FuPS on simpler tasks such as *Listener* and *Push*, and yields a clear advantage on the more challenging *World* task. This indicates that hardware-aligned sparse specialization improves both learning speed and final coordination quality. Compared with SNP and NoPS, PATH-NM is more sample efficient because it retains a shared representation while still allowing agent-specific adaptation.

On SMACv2, PATH-NM achieves the highest win rates across Protoss, Terran, and Zerg settings. Compared with FuPS, it benefits from agent-type-specific sparse subnetworks and therefore avoids policy homogenization. NoPS remains clearly less sample efficient, while SNP struggles under non-stationary MARL dynamics. Kaleidoscope is competitive, but its unstructured sparsity does not provide direct hardware acceleration and yields less stable gains than PATH-NM.

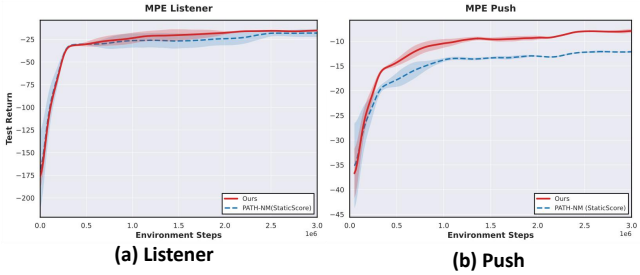

Fig. 4. Ablation of PATH-NM (StaticScore).

We focus on standard 5v5 SMACv2 scenarios to enable controlled comparison with prior parameter-sharing methods. Extending PATH-NM to larger-scale settings, such as 10v10 and 20v25, is an important direction for future work. Taken together, these results suggest that hardware-aware sparse specialization improves not only deployment efficiency but also representation quality in heterogeneous coordination tasks, likely because it preserves useful shared structure while preventing excessive policy coupling across agents.

C. Ablation Study

We evaluate two ablated variants: **PATH-NM (StaticScore)**, which removes online activation-aware scoring, and **PATH-NM (NoDiv)**, which removes diversity regularization. As shown in Fig. 4 and Fig. 5, removing either component consistently slows convergence on *Listener* and *Push*, and leads to lower final returns on the more challenging task. These results confirm that both dynamic scoring and explicit diversity regularization are important for effective sparse specialization in MARL. Removing dynamic scoring weakens the

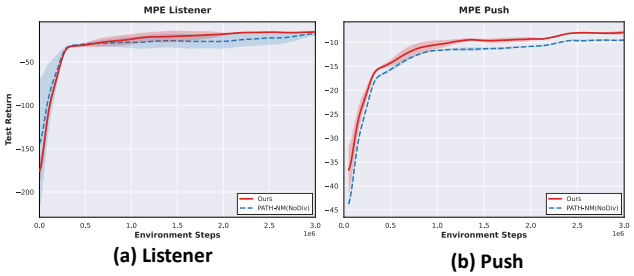

Fig. 5. Ablation of PATH-NM (NoDiv).

TABLE II
INFERENCE-TIME SPEEDUP OVER KALEIDOSCOPE.

Scenario	Kaleidoscope	PATH-NM
Push	1.0×	1.23×
World	1.0×	1.25×
Zerg	1.0×	1.57×
Protoss	1.0×	1.59×
Average	1.0×	1.41×

model's ability to track evolving activation statistics during RL training, while removing diversity regularization increases structural overlap across agents and reduces specialization.

D. Acceleration Evaluation

We measure inference speed on an NVIDIA RTX 4090 GPU. PATH-NM directly leverages Sparse Tensor Cores through hardware-aligned $N:M$ sparsity, and we use CUDA Graph to reduce kernel launch overhead for the small matrix operations common in MARL policies.

As shown in Table II, PATH-NM achieves speedups of $1.23\times$, $1.25\times$, $1.57\times$, and $1.59\times$ over Kaleidoscope on Push, World, Zerg, and Protoss, respectively, with an average speedup of $1.41\times$. In contrast, Kaleidoscope relies on unstructured sparsity and does not provide practical hardware acceleration on standard GPUs.

E. Discussion

PATH-NM is designed for settings where parameter sharing remains desirable but agent specialization is also necessary. Its main practical advantage is that specialization is achieved through hardware-aligned sparse patterns rather than through fully independent networks or unstructured masks. Because 2:4 masks can be represented compactly as pattern indices, the additional per-agent overhead is small relative to maintaining separate dense policies, while inference remains directly compatible with sparse GPU kernels.

VI. CONCLUSION

We presented PATH-NM, a hardware-aware framework for learning semi-structured $N:M$ sparse policies in heterogeneous MARL. By reformulating sparsification as pattern distribution learning, PATH-NM addresses non-stationary activation dynamics, discrete hardware constraints, and policy homogenization under parameter sharing. The proposed dynamic scoring, differentiable pattern transformation, and orthogonal diversity learning enable efficient specialization while preserving end-to-end optimization. Experiments on MPE and SMACv2 demonstrate consistent gains over representative parameter-sharing baselines, together with practical inference acceleration on modern GPUs. In future work, we will evaluate PATH-NM on larger-scale heterogeneous scenarios and study its sensitivity to sparsity and regularization hyperparameters.

REFERENCES

[1] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev *et al.*, "Grandmaster level in starcraft ii using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.

[2] Y. Cao, W. Yu, W. Ren, and G. Chen, "An overview of recent progress in the study of distributed multi-agent coordination," *IEEE Transactions on Industrial Informatics*, vol. 9, no. 1, pp. 427–438, 2013.

[3] Y. Sinan Nasir and D. Guo, "Multi-agent deep reinforcement learning for dynamic power allocation in wireless networks," *arXiv e-prints*, pp. arXiv–1808, 2018.

[4] F. A. Oliehoek, M. T. Spaan, and N. Vlassis, "Optimal and approximate q-value functions for decentralized pomdps," *Journal of Artificial Intelligence Research*, vol. 32, pp. 289–353, 2008.

[5] J. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32, no. 1, 2018.

[6] R. Lowe, Y. I. Wu, A. Tamar, J. Harb, O. Pieter Abbeel, and I. Mordatch, "Multi-agent actor-critic for mixed cooperative-competitive environments," *Advances in neural information processing systems*, vol. 30, 2017.

[7] T. Rashid, M. Samvelyan, C. S. De Witt, G. Farquhar, J. Foerster, and S. Whiteson, "Monotonic value function factorisation for deep multi-agent reinforcement learning," *The Journal of Machine Learning Research*, vol. 21, no. 1, pp. 7234–7284, 2020.

[8] M. Samvelyan, T. Rashid, C. S. De Witt, G. Farquhar, N. Nardelli, T. G. Rudner, C.-M. Hung, P. H. Torr, J. Foerster, and S. Whiteson, "The starcraft multi-agent challenge," *arXiv preprint arXiv:1902.04043*, 2019.

[9] F. Christianos, G. Papoudakis, M. A. Rahman, and S. V. Albrecht, "Scaling multi-agent reinforcement learning with selective parameter sharing," in *International Conference on Machine Learning*. PMLR, 2021, pp. 1989–1998.

[10] W. Kim and Y. Sung, "Parameter sharing with network pruning for scalable multi-agent deep reinforcement learning," *arXiv preprint arXiv:2303.00912*, 2023.

[11] X. Li, L. Pan, and J. Zhang, "Kaleidoscope: Learnable masks for heterogeneous multi-agent reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 37, pp. 22 081–22 106, 2024.

[12] R. L. Castro, A. Ivanov, D. Andrade, T. Ben-Nun, B. B. Fraguela, and T. Hoefler, "Venom: A vectorized n: M format for unleashing the power of sparse tensor cores," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, 2023, pp. 1–14.

[13] K. Zhao, T. Yuan, H. Bao, Z. Su, C. Gao, Z. Sun, Z. Liang, L. Jing, and J. Chen, "Beyond 2: 4: exploring v: N: M sparsity for efficient transformer inference on gpus," *arXiv preprint arXiv:2410.16135*, 2024.

[14] E. Frantar and D. Alistarh, "Sparsegpt: Massive language models can be accurately pruned in one-shot," in *International conference on machine learning*. PMLR, 2023, pp. 10 323–10 337.

[15] M. Sun, Z. Liu, A. Bair, and J. Z. Kolter, "A simple and effective pruning approach for large language models," *arXiv preprint arXiv:2306.11695*, 2023.

[16] F. A. Oliehoek, C. Amato *et al.*, *A concise introduction to decentralized POMDPs*. Springer, 2016, vol. 1.

[17] B. Ellis, J. Cook, S. Moalla, M. Samvelyan, M. Sun, A. Mahajan, J. Foerster, and S. Whiteson, "Smacv2: An improved benchmark for cooperative multi-agent reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 36, 2024.