PATMASK: PLUGGABLE ADVERSARIAL TOPOLOGY MASKING FOR MULTI-AGENT REINFORCEMENT LEARNING UNDER COMMUNICATION DISCONNECTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Cooperative Multi-Agent Reinforcement Learning (MARL) relies on communication to mitigate limited local observations. However, existing methods are vulnerable to communication disconnection. To address this, we propose PatMask, a pluggable adversarial topology masking mechanism to enhance the robustness of existing MARL methods under diverse communication disconnections. Specifically, PatMask introduces an adversarial mask generator that adaptively disrupts critical communication links during training. We establish a contraction guarantee for robust policy evaluation under topology perturbations and develop an alternating adversarial training procedure. Extensive evaluations on various cooperative benchmarks show that PatMask can maintain high win rate and high coordination efficiency across various environments under controlled communication disconnection. Compared with baselines trained without additional disconnection augmentation, PatMask achieves 15.7-41.6 percentage-point higher win rates with only 41% of the training steps and less than 77% of the communication cost. Moreover, trained PatMask policies can generalize without retraining across the evaluated disconnection types and probabilities. [1]

1 INTRODUCTION

Cooperative Multi-Agent Reinforcement Learning (MARL) has emerged as a key technology for complex decision-making tasks, demonstrating immense potential in fields such as resource allocation (Wang et al., 2024), autonomous driving (Zhou et al., 2021), and robotic swarm control (Swamy et al., 2020). Under the prevailing Centralized Training with Decentralized Execution (CTDE) paradigm (Lowe et al., 2017; Yu et al., 2022), agents leverage global information during training but rely solely on local observations during execution. To mitigate the information loss caused by partial observability, incorporating communication mechanisms has become essential for enhancing inter-agent coordination and improving overall system performance (Li et al., 2025; Ding et al., 2020; Yuan et al., 2022; Sun et al., 2024).

Research on multi-agent communication has developed mechanisms for selecting message recipients and controlling message transmission to improve coordination and communication efficiency (Zhang et al., 2025). Communication availability introduces an additional challenge that physical obstruction, hardware failures, or interference can prevent messages from reaching their intended recipients (Yang et al., 2026). Failures may affect individual links or isolate agents, changing the information available for coordinated decisions. Policies that rely on particular communication patterns may consequently encounter an information mismatch at execution. This motivates studying how communication-dependent policies can be trained to tolerate different patterns of missing messages.

Existing robustness methods provide important foundations. For example, TMC (Zhang et al., 2020) uses temporal message control to improve robustness to transmission loss, and MessageDropout (Kim et al., 2019b) randomly removes received message blocks during training. Building

[1] The source code is available in the attachment for blind review.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

on this line of work, we investigate how training perturbations should be selected. Fixed-probability random masking provides diverse failure exposure, but its sampling probabilities do not adapt to the current learner. A learned generator can instead update topology-mask probabilities using feedback from the learner. The resulting research question is whether combining such adaptive masks with random masking improves robustness over random failure exposure alone.

In this study, we propose PatMask, a Pluggable adversarial topology Masking mechanism to enhance the robustness of CTDE-based MARL under communication disconnection. We address the communication problem by a zero-sum adversarial defense game (Reddi et al., 2024). PatMask combines a learned topology-mask generator with random masking during centralized training. The generator predicts link-retention probabilities and is updated to increase the learner's Temporal Difference (TD) loss, while the MARL learner is updated to reduce that loss. The learned mask distribution therefore responds to the current learner, while random masking provides additional topology diversity. We formulate a topological adversarial operator for evaluating fixed decentralized policies and prove its contraction property. The generator is removed after training, preserving decentralized execution. Extensive evaluations demonstrate that PatMask significantly outperforms SOTA baselines in terms of win rate, sample efficiency, and coordination efficiency in various environments under communication disconnection. It achieves a win rate of 99% (15.7 to 41.6 percentage-point improvement) in hard cooperative scenarios with only 41% of training steps and less than 77% of communication costs. Moreover, PatMask improves robustness across the evaluated disconnection settings and topology-aware backbones without test-time retraining. The major contributions are as follows:

- We formulate the MARL communication problem under dynamic disconnections using a topological adversarial operator and prove its contraction property.
- We introduce PatMask, which combines learned adversarial topology masks with random masking through alternating updates of a mask generator and a topology-aware MARL learner.
- Experiments across multiple environments and topology-aware backbones demonstrate improved performance under the evaluated communication disconnection modes and probabilities.

2 Related Work

2.1 Communication Optimization in MARL

Communication methods in cooperative MARL mainly reduce what agents send and whom they contact under bandwidth and transmission costs (Jiang & Lu, 2018). One line controls message content and timing. For example, NDQ (Wang et al., 2020) compresses messages by mutual-information maximization, I2C (Ding et al., 2020) forwards only causally relevant information, and TMC (Zhang et al., 2020) triggers transmission inside a time window. A second line decides the communication structure. SchedNet (Kim et al., 2019a), G2ANet (Jiang & Lu, 2018), and MAIC (Yuan et al., 2022) restrict access to important agents or links. More recent topology-aware methods make that structure part of coordination. For example, T2MAC (Sun et al., 2024) sparsifies communication through selective engagement, GTDE (Li et al., 2025) and GACG (Duan et al., 2024) organize agents into groups, and TGCNET (Zhang et al., 2025) learns a dynamic directed graph shared between training and execution. QMIX (Rashid et al., 2020) is the value-decomposition reference without a learned communication topology. These methods train the policy for a preferred communication pattern. PatMask keeps the underlying learner and, during centralized training, exposes it to missing links so coordination remains effective when that pattern is unavailable at execution.

2.2 Communication Constraints in Multi-Agent Systems

To address time delays, DATS (Chen et al., 2021), DAMARL (Chen et al., 2020), and DAMAVL (Zhang et al., 2023) target latencies in action execution, state perception, and reward feedback, respectively. For communication channels, DACOM (Yuan et al., 2023) optimizes decision-making during waiting periods, while CoDe (Song et al., 2025) uses dual alignment for asynchronous messaging. A more severe challenge arises from physical interferences or hardware failures, which can lead to topology disconnections. In the broader field of multi-agent control (MAC), researchers have developed protocols to handle such faults. Studies on link disconnection (Parlan-

geli & Valcher, 2021; Yuan et al., 2016) and agent disconnection (Qian et al., 2024) have proposed adaptive and federated control mechanisms for system recovery. However, these control-theoretic approaches often rely on explicit dynamics models or strict topological assumptions, such as strong connectivity. PatMask trains communication-based policies under learned topology masks and evaluates robustness to specified execution-time disconnection processes.

2.3 Adversarial Games in MARL

Adversarial training has been used to improve robustness to perceptual perturbations (Huang et al., 2017; Gleave et al., 2020) and physical parameter variation (Pinto et al., 2017; Peng et al., 2018). Robust communication has also received attention. Message-Dropout (Kim et al., 2019b) randomly drops received message blocks during training to improve robustness to communication errors. PatMask builds on training with unavailable messages by learning topology-mask probabilities through adversarial updates against the current learner, combined with random masking. Other approaches include TMC (Zhang et al., 2020) for transmission loss, MAGI (Ding et al., 2024) for robust message representations, and ADMAC (Yu et al., 2024) for adjusting message influence according to estimated reliability.

3 Preliminaries

3.1 Problem Formulation

We formulate MARL under execution-time communication disconnection as a Communication-Disrupted Decentralized Partially Observable Markov Decision Process (CD-Dec-POMDP). Built upon the standard Dec-POMDP (Bernstein et al., 2002; Oliehoek & Amato, 2016), CD-Dec-POMDP is defined by the tuple $\langle \mathcal{N}, \mathcal{S}, \Omega, \mathcal{O}, \mathcal{A}, R, \mathcal{P}, \gamma, \mathcal{G}, \mathcal{D} \rangle$. Specifically, $\mathcal{N}$ represents the set of agents; $\mathcal{S}$ is the global state space; $\mathcal{O}$ is the set of local observations, with Ω being the observation function; $\mathcal{A}$ denotes the joint action space; R is the global reward function; $\mathcal{P}$ represents the environmental dynamics; and $\gamma \in [0, 1)$ is the discount factor.

Distinct from the Dec-POMDP formulation, $\mathcal{G}$ characterizes the dynamic communication topology space and determines the feasible peer-to-peer information exchange at each step. $\mathcal{D} = \{D_{link}, D_{node}, D_{mixed}, D_{\text{obs-dist}}, D_{\text{reg}}\}$ denotes the set of disconnection modes, characterizing the topological perturbations within the system.

Independent Disconnection Modes: We initially model baseline disconnections as independent Bernoulli processes. We define indicators for link $\langle i, j \rangle$ failure as $I_t^{(i,j)} \sim \text{Bernoulli}(p_l)$ and node k failure as $J_t^k \sim \text{Bernoulli}(p_n)$:

$$I_t^{(i,j)} = \begin{cases} 0, & \text{link failure} \\ 1, & \text{normal} \end{cases}, \quad J_t^k = \begin{cases} 0, & \text{node failure} \\ 1, & \text{normal} \end{cases}. \tag{1}$$

Mixed Disconnection (D_{mixed}) occurs when both types of failures coexist.

Structured Disconnection Modes: We also evaluate two synthetic disconnection modes beyond independent failures: **Observation-Conditioned Link Disconnection ($D_{\text{obs-dist}}$)**, where link reliability decreases with observation dissimilarity, and **Regional Disconnection (D_{reg})**, where agents grouped by observation similarity experience correlated outages. Definitions and results appear in Appendices A.1 and C.2, respectively.

3.2 Multi-agent Cooperation Paradigms

CTDE permits joint information during training while retaining decentralized policies at execution. It supports both actor-critic methods such as MAPPO (Yu et al., 2022) and value-decomposition methods (Rashid et al., 2020; Wang et al.). We adopt the latter formulation to train a centralized value estimator and adversary while agents execute using local histories and available messages. Specifically, let τ^i denote the local action-observation history and a^i be the individual action for agent i. For each agent, an individual utility function $Q_i(\tau^i, a^i)$ is first estimated. A mixing network then aggregates these utilities into a joint action-value function Q_{tot}. To ensure consistency between

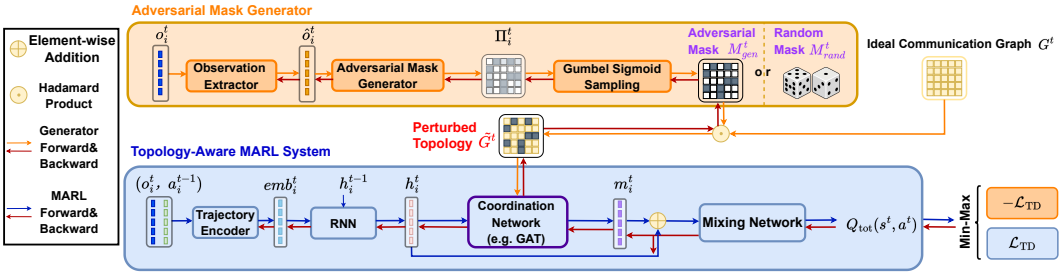


Figure 1: The overall architecture of PatMask. During training, the adversarial mask generator identifies critical links and plays a minimax game against the MARL system.

local and global greedy actions, the Individual-Global-Max (IGM) principle (Son et al., 2019) is maintained:

$$\arg\max_{a} Q_{tot}(\tau, a) = \left\{ \arg\max_{a^i} Q_i(\tau^i, a^i) \right\}_{i=1}^{N}. \tag{2}$$

To optimize the joint value approximation, the system parameters θ are trained end-to-end by minimizing the TD loss $\mathcal{L}_{TD}$ over mini-batches sampled from the experience replay buffer $\mathcal{B}$:

$$\mathcal{L}_{TD}(\theta) = \mathbb{E}_{\mathcal{B}} \left[\left(y_{tot} - Q_{tot}(\tau, a; \theta) \right)^2 \right], \tag{3}$$

where $y_{tot} = r + \gamma \max_{a'} \hat{Q}_{tot}(\tau', a')$ is the regression target computed using a target network $\hat{Q}_{tot}$. Here, the prime superscript (e.g., τ', a') denotes the variables at the next time step. During decentralized execution, agents select actions greedily based on their local Q_i, conditioned on available communication messages.

4 Methodology

4.1 Overview

To address the fragility of multi-agent coordination under passive, execution-time communication disconnection, we propose integrating PatMask into the cooperative MARL framework, as illustrated in Fig. 1. PatMask is pluggable as the mask generator substitutes a masked adjacency with a perturbed topology for the existing communication topology only during centralized training (Sec. 4.2). It is removed before decentralized execution. Message contents, the coordination network, and the host's decentralized policy remain those of the original MARL system. We formulate robustness to communication disconnection through the Topological Adversarial Bellman Operator $\mathcal{T}_{adv}$ (Sec. 4.3). Although exhaustive topology search involves exponentially many edge-retention combinations, the generator predicts an $N \times N$ mask-probability matrix, whose size grows quadratically with the number of agents. Motivated by this robust objective, PatMask combines learned mask generation (Sec. 4.4) with alternating adversarial updates (Sec. 4.6) to train the topology-aware MARL system (Sec. 4.5).

4.2 Communication Topology Modeling

Communication among agents is modeled as a time-varying directed graph $G^t = (V^t, E^t)$. Here, V^t represents the set of agents as nodes at time step t, and E^t denotes the set of edges representing communication links. To formalize topological attacks, we define the Perturbed Topology $\tilde{G}$ as the Hadamard product of the original adjacency matrix G and an adversarial mask matrix $\tilde{M}$:

$$\tilde{G}^t = G^t \odot \tilde{M}^t, \tag{4}$$

where $\tilde{M}^t \in \{0, 1\}^{N \times N}$ indicates link availability. Masking removes communication links without modifying the contents of delivered messages. Extensions to message-content attacks are discussed in Appendix D. For notational simplicity, the superscript of time t may be omitted when the context is not time-specific.

4.3 Topological Adversarial Bellman Operator

We characterize robustness through evaluation of a fixed decentralized policy π. Let x contain the environment state and agents' local histories before communication, so that the resulting process is Markov. This augmented state is used for analysis, and each agent still acts from its local history and delivered messages. For topology $\tilde{G}$, let $r^\pi(x, \tilde{G})$ and $P^\pi(\cdot \mid x, \tilde{G})$ denote the expected reward and next augmented-state distribution induced by π. Both account for the effect of message availability on actions and subsequent histories. We define the Topological Adversarial Bellman Operator:

$$(\mathcal{T}^\pi_{adv} V)(x) = \min_{\tilde{G} \in \mathcal{N}(G(x))} \Big\{ r^\pi(x, \tilde{G}) + \gamma \, \mathbb{E}_{x' \sim P^\pi(\cdot|x,\tilde{G})}[V(x')] \Big\}. \tag{5}$$

Here, $\mathcal{N}(G(x))$ is the nonempty finite set of admissible communication subgraphs. The adversary observes x and selects a topology before the agents act, with no additional constraints coupling its choices across time.

Theorem 4.1. *For fixed π, bounded rewards, and $0 \le \gamma < 1$, $\mathcal{T}^\pi_{adv}$ is a γ-contraction under the supremum norm. Its unique fixed point V^π_{rob} is the worst-case discounted value of π under the admissible topology adversary. Exact iteration of this operator converges to V^π_{rob}.*

The proof appears in Appendix A.2. This result characterizes robust policy evaluation. PatMask's alternating TD-loss updates (Sec. 4.6) provide a practical surrogate training procedure, whose convergence is not established by this theorem.

4.4 Adversarial Mask Generator

During centralized training, the adversarial mask generator constructs fixed-dimensional pairwise features from global state s and applies a shared 3-layer MLP to predict each link survival probability $\pi_{i,j}$. Collecting these predictions forms $\Pi \in [0,1]^{N \times N}$, which parameterizes the training-time communication masks. The generator searches for disruptive training topologies and is removed at execution, when each agent acts using only its local observation-action history and received messages. To enable differentiable training, we obtain a soft mask M_{gen} using the Gumbel-Softmax technique:

$$(M_{gen})_{i,j} = \mathrm{sigmoid}((\log(\pi_{i,j}) + g_{i,j})/\tau), \tag{6}$$

where $g_{i,j}$ is Gumbel noise and τ (default to 1.0) is the temperature parameter (Jang et al., 2017). To further enhance generalization, we introduce a hybrid training strategy that explicitly switches between the generated mask and a random mask based on a fixed probability ρ. The final adversarial mask $\tilde{M}$ is defined as:

$$\tilde{M} = \begin{cases} M_{gen}, & \text{with probability } \rho, \\ M_{rand}, & \text{with probability } 1-\rho, \end{cases} \tag{7}$$

where M_{rand} is a random Bernoulli mask. The fixed hyperparameter ρ controls the probability of selecting an adversarial mask during training, rather than the fraction of links removed. Random masking introduces additional topology diversity and PatMask does not automatically adapt ρ.

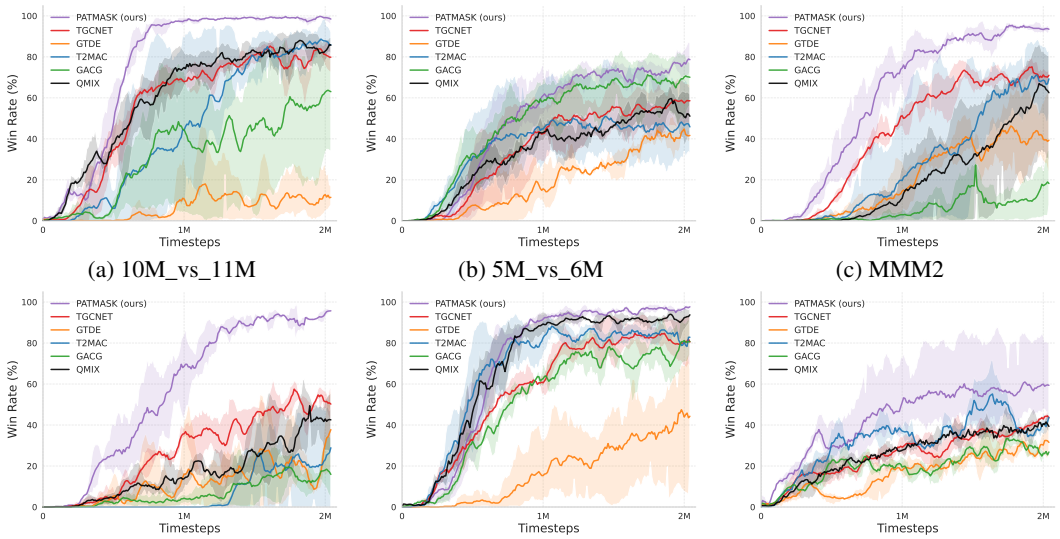
4.5 Topology-Aware MARL System

The MARL system utilizes a coordination network, such as Graph Attention Networks (GAT) (Veličković et al., 2018), to aggregate information under the perturbed topology $\tilde{G}$. Based on hidden states h_i from a recurrent neural network (RNN) (Cho et al., 2014), the system first computes raw attention coefficients $e_{i,j}$ for surviving neighbors $j \in \mathcal{N}_i(\tilde{G})$:

$$e_{i,j} = \mathrm{LeakyReLU}\left(\mathbf{a}^T [\mathbf{W}h_i \parallel \mathbf{W}h_j]\right), \tag{8}$$

where $\mathbf{a}$ is a learnable attention weight vector used to compute the attention coefficients, and $\mathbf{W}$ is a shared linear transformation weight matrix. These coefficients are normalized to determine the attention weights:

$$\alpha_{i,j} = \frac{\exp(e_{i,j})}{\sum_{k \in \mathcal{N}_i(\tilde{G})} \exp(e_{i,k})}. \tag{9}$$



(a) 10M_vs_11M (b) 5M_vs_6M (c) MMM2

(d) 27M_vs_30M (e) 8M_vs_9M (f) 1o_10b_vs_1r

Figure 2: Test win rates of PatMask compared to baselines across six SMAC scenarios under communication disconnection. Shaded regions represent 95% confidence intervals.

Agents then aggregate these messages into context vectors $m_i = \sum_{j \in \mathcal{N}_i(\tilde{G})} \alpha_{i,j} \mathbf{W} h_j$, with local utilities $Q_i(h_i, m_i)$ integrated by a centralized mixer to estimate Q_{tot}. PatMask integrates training-time masking with compatible topology-aware value-decomposition backbones. Extension to actor-critic learning requires adapting the training objectives and is discussed in Appendix D.

4.6 Adversarial Training Objective

We train the mask generator and MARL learner through alternating adversarial updates. The generator seeks masks that increase the learner's temporal-difference (TD) error, while the learner minimizes this error under the generated masks. We define

$$\mathcal{L}_{TD}(\theta, \phi) = \mathbb{E}_{\mathcal{B}} \left[\Big(\mathrm{sg}(y_{tot}) - Q_{tot}(s, \tilde{G}_\phi, a; \theta) \Big)^2 \right], \tag{10}$$

where y_{tot} is the bootstrapped TD target computed using target networks, and sg denotes the stop-gradient operation. With the learner fixed, the generator maximizes $\mathcal{L}_{TD}$ by minimizing

$$\mathcal{L}_{adv}(\phi; \theta) = -\mathcal{L}_{TD}(\theta, \phi). \tag{11}$$

With the generator fixed, the learner minimizes $\mathcal{L}_{TD}(\theta, \phi)$, updating its agent networks and centralized mixer. The implementation uses Adam for both updates.

TD-error maximization serves as a proxy for selecting difficult training masks. It is not an exact solution of the topology minimization in Eq. 5. Theorem 4.1 concerns evaluation of a fixed policy, whereas training changes both the policy and generator. The alternating procedure and its empirical dynamics are discussed in Appendix A.3.

5 Experiments

We evaluate PatMask by addressing the following research questions: **RQ1:** Learning progress and win-rate performance across various environments (Sec. 5.2). **RQ2:** Zero-shot generalization capability under diverse communication-disconnection scenarios (Sec. 5.3). **RQ3:** Effectiveness of PatMask as a modular component in enhancing the robustness of existing MARL algorithms (Sec. 5.4). **RQ4:** Impact of key design choices and hyperparameters on overall efficacy (Sec. 5.5).

5.1 EXPERIMENTAL SETUP

Environments. We evaluate our method on three benchmarks: the StarCraft II Multi-Agent Challenge (SMAC) (Whiteson et al., 2019), SMACv2 (Ellis et al., 2023), and Level-Based Foraging (LBF) (Christianos et al., 2020). For SMAC, we select six Hard and Super Hard scenarios (5M_vs_6M, MMM2, 10M_vs_11M, 8M_vs_9M, 27M_vs_30M, and 1o_10b_vs_1r). We also utilize SMACv2 (Protoss and Terran) to evaluate generalization under stochastic spawn positions and unit types, and the 19×19 LBF grid-world, where communication is essential to satisfy level-sum constraints. At evaluation, the simulator imposes synthetic communication failures using the modes in Sec. 3.1 and Appendix A.1, without the adversarial generator. These controlled tests examine transfer across specified communication-availability patterns. Environment descriptions and deployment-related limitations are introduced in Appendices B.1 and D, respectively.

Baselines. We integrate PatMask into the CTDE algorithm, using GAT as the coordination network by default. The baseline methods include QMIX (Rashid et al., 2020), a typical method for MARL, and four other SOTA communication topology-aware algorithms: TGCNET (Zhang et al., 2025) integrates topology-aware Q-learning; GTDE (Li et al., 2025) utilizes GNN-based decoupled exploration; T2MAC (Sun et al., 2024) leverages Transformer-based attention; GACG (Duan et al., 2024) incorporates dynamic grouping mechanisms. PatMask receives adversarial and random communication masks during training. Additional comparisons on MMM2 include random-dropout training for the topology-aware baselines and robustness-oriented communication methods (Appendix C.1). All experiments are trained three times with different random seeds. All training and testing are performed on NVIDIA GeForce RTX 4090 GPUs. Detailed hyperparameter settings and evaluation metrics are provided in Appendix B.2.

5.2 PERFORMANCE ACROSS VARIOUS ENVIRONMENTS

PatMask demonstrates significant improvements in win rate, sample efficiency, and communication efficiency compared to baseline methods across various environments.

Performance on SMAC. As illustrated in Fig. 2, PatMask demonstrates strong learning progress and high win rates across the tested methods. The performance improvement is pronounced in more challenging SMAC maps such as 10M_vs_11M, MMM2, and 27M_vs_30M, which are characterized by large agent populations and heterogeneous units where precise coordination is essential for success. For example, while other methods can achieve average win rates at the highest of 83.3%, 70.0%, and 52.9% in these maps under communication disconnection, PatMask achieves 99.0%, 93.5%, and 94.5%, respectively, corresponding to improvements of 15.7 to 41.6 percentage points over baselines without additional disconnection augmentation. PatMask reaches a 99.2% win rate after 0.64M training steps on 10M_vs_11M and 94.5% after 1.08M steps on 27M_vs_30M. These learning curves measure progress in environment steps. The additional training cost of alternating adversarial updates is quantified against a matched GAT baseline in Appendix C.4. For a detailed visualization of the behavioral differences in coordination, refer to the case study in Appendix C.6.

Performance on SMACv2 and LBF. SMAC has constrained environmental randomness and local observability. We extend the evaluation on SMACv2, where starting positions change in every episode, preventing agents from simply memorizing fixed routines or specific locations on the map to achieve coordination. As shown in Table 1, PatMask consistently outperforms all baselines across both SMACv2 scenarios. This performance gap is particularly pronounced in the Protoss 10_vs_10 task, where PatMask maintains a significant lead over representative baselines such as T2MAC. These findings confirm that adversarial training enables the system to maintain effective coordination even as unit layouts and starting positions change dynamically across stochastic scenarios. In the LBF benchmark, which requires precise timing synchronization for level-sum constraints, PatMask achieves the highest win rate. This confirms the framework's versatility and broad applicability across diverse task types.

Table 1: Win rates (%) in SMACv2 and LBF environments.

Method	Protoss 10 vs 10	Terran 10 vs 10	LBF 19×19-9p-4f
GTDE	11.5 ± 3.5	45.9 ± 5.5	42.0 ± 0.6
GACG	20.2 ± 5.6	42.5 ± 3.1	13.0 ± 18.3
QMIX	22.1 ± 3.1	43.8 ± 2.9	12.0 ± 16.5
TGCNET	27.4 ± 3.8	$\underline{46.1 \pm 2.1}$	$\underline{42.5 \pm 3.1}$
T2MAC	$\underline{32.4 \pm 5.8}$	29.4 ± 13.9	29.1 ± 18.9
PatMask (Ours)	$\mathbf{51.8 \pm 1.2}$	$\mathbf{59.5 \pm 2.4}$	$\mathbf{46.3 \pm 2.2}$

Table 3: Test win rates (%) under **link disconnection**. p indicates the link disconnection probability.

Map	10M_vs_11M			1o_10b_vs_1r			27M_vs_30M		
Method	$p = 0.3$	$p = 0.5$	$p = 0.7$	$p = 0.3$	$p = 0.5$	$p = 0.7$	$p = 0.3$	$p = 0.5$	$p = 0.7$
GTDE	68.2 ± 49.1	63.9 ± 46.3	47.3 ± 38.7	$\mathbf{80.6 \pm 6.0}$	$\mathbf{75.0 \pm 13.2}$	$\mathbf{59.3 \pm 15.9}$	93.2 ± 2.1	92.8 ± 2.8	89.0 ± 5.2
GACG	68.6 ± 6.9	67.3 ± 5.1	63.3 ± 6.7	34.8 ± 19.3	29.8 ± 17.7	26.4 ± 14.5	35.7 ± 18.6	32.8 ± 17.3	16.3 ± 23.6
T2MAC	86.0 ± 11.8	88.0 ± 9.8	84.6 ± 10.5	36.8 ± 34.5	31.2 ± 27.2	27.5 ± 23.9	14.5 ± 25.0	13.2 ± 22.9	15.6 ± 27.1
TGCNET	97.0 ± 1.9	91.5 ± 0.5	84.2 ± 2.9	56.0 ± 4.1	48.4 ± 8.1	43.0 ± 5.1	80.8 ± 12.1	74.2 ± 11.7	27.8 ± 20.0
PatMask (Ours)	$\mathbf{99.3 \pm 0.4}$	$\mathbf{99.0 \pm 0.9}$	$\mathbf{99.4 \pm 0.5}$	57.7 ± 21.5	59.3 ± 22.4	58.0 ± 20.5	$\mathbf{95.3 \pm 1.1}$	$\mathbf{96.9 \pm 0.9}$	$\mathbf{96.4 \pm 0.2}$
Map	**5M_vs_6M**			**8M_vs_9M**			**MMM2**		
Method	$p = 0.3$	$p = 0.5$	$p = 0.7$	$p = 0.3$	$p = 0.5$	$p = 0.7$	$p = 0.3$	$p = 0.5$	$p = 0.7$
GTDE	$\mathbf{75.3 \pm 8.1}$	69.3 ± 5.7	58.4 ± 4.5	93.3 ± 4.5	91.7 ± 3.6	78.3 ± 11.9	86.3 ± 4.4	81.9 ± 5.8	65.0 ± 10.7
GACG	39.4 ± 16.2	39.5 ± 17.7	40.2 ± 18.2	55.6 ± 25.2	54.4 ± 26.5	51.1 ± 29.1	55.4 ± 9.7	51.8 ± 12.9	46.5 ± 16.8
T2MAC	48.0 ± 18.8	47.0 ± 20.5	50.0 ± 20.0	81.5 ± 1.5	82.2 ± 3.0	80.5 ± 1.5	70.2 ± 10.2	64.8 ± 16.6	58.7 ± 22.2
TGCNET	56.9 ± 13.7	58.3 ± 14.9	57.2 ± 16.4	93.8 ± 3.3	90.3 ± 5.3	82.7 ± 11.8	77.5 ± 9.2	73.2 ± 9.3	65.3 ± 11.4
PatMask (Ours)	74.4 ± 9.2	$\mathbf{73.4 \pm 8.0}$	$\mathbf{74.7 \pm 10.1}$	$\mathbf{96.9 \pm 2.0}$	$\mathbf{97.3 \pm 1.5}$	$\mathbf{95.3 \pm 1.6}$	$\mathbf{90.5 \pm 1.3}$	$\mathbf{94.3 \pm 1.3}$	$\mathbf{94.7 \pm 1.4}$

Table 4: Test win rates (%) under **node disconnection**. p indicates the node failure probability.

Map	10M_vs_11M			1o_10b_vs_1r			27M_vs_30M		
Method	$p = 0.3$	$p = 0.5$	$p = 0.7$	$p = 0.3$	$p = 0.5$	$p = 0.7$	$p = 0.3$	$p = 0.5$	$p = 0.7$
GTDE	28.7 ± 34.2	8.3 ± 12.3	1.3 ± 2.2	56.1 ± 7.5	33.4 ± 5.1	17.3 ± 11.6	69.4 ± 7.9	31.3 ± 15.5	8.8 ± 12.9
GACG	67.4 ± 6.4	64.9 ± 6.8	61.2 ± 7.7	29.9 ± 18.5	27.6 ± 16.1	24.8 ± 14.1	33.4 ± 19.8	26.6 ± 19.4	13.0 ± 17.4
T2MAC	86.1 ± 12.0	87.2 ± 10.2	85.2 ± 11.3	34.8 ± 32.2	31.2 ± 27.5	26.7 ± 23.2	14.7 ± 25.5	14.1 ± 24.5	13.8 ± 23.9
TGCNET	93.8 ± 0.7	85.8 ± 3.5	78.0 ± 4.1	44.8 ± 4.8	42.5 ± 4.6	39.3 ± 5.0	74.3 ± 14.1	56.8 ± 14.4	20.8 ± 10.9
PatMask (Ours)	$\mathbf{99.5 \pm 0.5}$	$\mathbf{99.4 \pm 0.4}$	$\mathbf{99.0 \pm 0.5}$	$\mathbf{56.7 \pm 19.0}$	$\mathbf{54.5 \pm 13.0}$	$\mathbf{43.9 \pm 8.1}$	$\mathbf{98.0 \pm 0.4}$	$\mathbf{96.6 \pm 0.9}$	$\mathbf{95.3 \pm 0.4}$
Map	**5M_vs_6M**			**8M_vs_9M**			**MMM2**		
Method	$p = 0.3$	$p = 0.5$	$p = 0.7$	$p = 0.3$	$p = 0.5$	$p = 0.7$	$p = 0.3$	$p = 0.5$	$p = 0.7$
GTDE	56.6 ± 4.0	46.9 ± 6.8	34.6 ± 11.8	74.8 ± 20.2	55.8 ± 38.1	43.3 ± 37.4	70.9 ± 1.7	42.4 ± 6.6	24.8 ± 8.9
GACG	40.2 ± 18.7	39.9 ± 16.6	39.9 ± 17.6	54.2 ± 25.7	53.4 ± 29.1	49.5 ± 30.4	53.4 ± 12.0	52.3 ± 13.8	44.1 ± 18.0
T2MAC	47.0 ± 22.5	48.1 ± 21.2	47.5 ± 20.7	82.2 ± 1.0	85.6 ± 1.1	80.7 ± 2.3	68.9 ± 14.3	66.8 ± 17.5	57.4 ± 22.1
TGCNET	60.2 ± 14.8	58.3 ± 12.0	58.9 ± 14.4	89.6 ± 3.8	85.3 ± 8.8	78.3 ± 14.2	73.5 ± 9.7	72.5 ± 10.1	60.1 ± 14.0
PatMask (Ours)	$\mathbf{71.4 \pm 7.1}$	$\mathbf{76.0 \pm 7.1}$	$\mathbf{75.3 \pm 8.5}$	$\mathbf{96.9 \pm 0.9}$	$\mathbf{96.3 \pm 3.0}$	$\mathbf{96.3 \pm 3.0}$	$\mathbf{93.8 \pm 1.5}$	$\mathbf{94.6 \pm 1.4}$	$\mathbf{96.3 \pm 2.2}$

Table 5: Test win rates (%) under **mixed disconnection**. The disconnection probability p is enforced independently on links and nodes.

Map	10M_vs_11M		1o_10b_vs_1r		27M_vs_30M		5M_vs_6M		8M_vs_9M		MMM2	
Method	$p = 0.3$	$p = 0.5$	$p = 0.3$	$p = 0.5$	$p = 0.3$	$p = 0.5$	$p = 0.3$	$p = 0.5$	$p = 0.3$	$p = 0.5$	$p = 0.3$	$p = 0.5$
GTDE	25.3 ± 31.3	4.0 ± 6.3	49.7 ± 4.3	25.8 ± 8.9	68.7 ± 5.3	28.2 ± 13.9	56.5 ± 6.1	38.6 ± 6.4	69.4 ± 25.6	50.9 ± 39.6	62.3 ± 4.2	36.9 ± 8.1
GACG	68.9 ± 2.2	63.5 ± 5.7	28.9 ± 16.1	25.3 ± 13.7	35.5 ± 23.2	14.0 ± 23.2	41.2 ± 18.9	39.5 ± 17.2	48.6 ± 33.1	45.2 ± 39.7	51.4 ± 16.1	42.7 ± 17.8
T2MAC	93.7 ± 3.2	90.7 ± 7.8	30.9 ± 27.0	27.8 ± 24.2	23.4 ± 33.1	22.2 ± 31.4	49.5 ± 24.3	46.5 ± 19.9	83.8 ± 0.1	84.6 ± 2.9	63.7 ± 19.7	55.8 ± 22.6
TGCNET	90.0 ± 1.3	78.6 ± 3.5	46.3 ± 5.2	39.8 ± 3.1	71.2 ± 13.2	15.2 ± 15.1	60.3 ± 13.4	58.8 ± 14.6	87.5 ± 6.9	76.8 ± 17.6	73.3 ± 12.5	58.1 ± 13.0
PatMask (Ours)	$\mathbf{99.5 \pm 0.4}$	$\mathbf{99.1 \pm 0.9}$	$\mathbf{58.9 \pm 19.6}$	$\mathbf{53.8 \pm 13.4}$	$\mathbf{96.1 \pm 0.5}$	$\mathbf{95.5 \pm 1.1}$	$\mathbf{74.3 \pm 7.8}$	$\mathbf{76.3 \pm 6.7}$	$\mathbf{96.3 \pm 0.5}$	$\mathbf{97.1 \pm 2.3}$	$\mathbf{94.5 \pm 2.4}$	$\mathbf{95.2 \pm 1.6}$

Learned mask sparsity. Table 2 characterizes the communication topologies produced by the generator during training. Mean masked-link ratios range from 57% to 77%, leaving 23–43% of links available during adversarial training. Link removal does not increase monotonically with agent population size. The generator masks 76% of links on 8M_vs_9M but 63% on 27M_vs_30M. Thus, larger population do not necessarily receive a higher fraction of masked links in the evaluated tasks.

Table 2: Adversarial topology mask sparsity across different environments. Values represent the ratios of masked communication links.

Scenario	Sparsity	Scenario	Sparsity
5M_vs_6M	0.70 ± 0.11	27M_vs_30M	0.63 ± 0.09
8M_vs_9M	0.76 ± 0.15	Protoss 10vs10	0.64 ± 0.07
10M_vs_11M	0.57 ± 0.28	Terran 10vs10	0.70 ± 0.11
1o_10b_vs_1r	0.77 ± 0.10	LBF 19×19-9p	0.63 ± 0.06
MMM2	0.67 ± 0.15		

5.3 Zero-Shot Generalization across Various Communication Disconnections

We evaluate trained policies without retraining under link and node disconnection with $p \in \{0.3, 0.5, 0.7\}$, and mixed disconnection with $p \in \{0.3, 0.5\}$. In the mixed setting, p applies independently to link and node failures. Tables 3, 4, and 5 show that PatMask generally improves win rates over the evaluated baselines, although performance varies across tasks. For example, under link disconnection at $p = 0.7$, PatMask achieves $99.4 \pm 0.5\%$ on 10M_vs_11M and $58.0 \pm 20.5\%$

on 1o_10b_vs_1r. These results support transfer to the tested disconnection conditions. Evaluation scope is discussed in Appendix D.

5.4 Compatibility for Various MARL Algorithms

To evaluate PatMask's compatibility, we integrate it into various communication topology-aware MARL algorithms. These algorithms implement the coordination network as GAT (default), TGCNET, or GCN (Jiang et al., 2019), while other workflows remain unchanged. The results are shown in Fig. 3. After integrating the PatMask mechanism, all methods exhibit substantial performance gains in different environments. Notably, while the vanilla GCN baseline suffers from a complete coordination collapse with near-zero win rates under severe disconnection, the integrated PatMask-GCN version recovers significantly, achieving win rates exceeding 70% in the environments of MMM2 and 10M_vs_11M in SMAC. Similarly, PatMask-TGCNET shows a steady improvement of approximately 20% over its vanilla implementation, reaching an asymptotic win rate exceeding 90%. The synergy of dynamic aggregation and adversarial training of PatMask helps identify inter-agent coordination vulnerabilities, preventing the overfitting to ideal topologies common in vanilla TGCNET and GCN. The pluggable and general property of PatMask allows it to inherit the inherent advantages of existing topology-aware MARL algorithms.

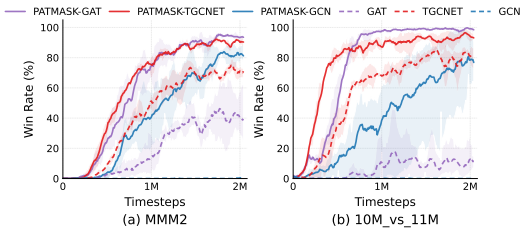


Figure 3: Results of integrating PatMask with different topology-aware algorithms in SMAC maps.

5.5 Ablation Study

Table 6 compares training with no masking, random-only, adversarial-only, and hybrid masks. Random masking substantially improves performance on both tasks. Adversarial-only training exceeds random-only training by 6.3 and 7.2 percentage points on MMM2 and 27M_vs_30M, respectively; hybrid training increases these gains to 10.4 and 11.8 points. These results support an additional benefit from learned adversarial masking in the tested configurations. Appendix C.1 reports comparisons with dropout-trained baselines, while Appendix C.3 examines sensitivity to the mask-sampling temperature and adversarial-mask probability.

Table 6: Masking-strategy ablation. Test win rates (%) on two SMAC tasks under mixed disconnection with $p = 0.5$. ρ is the probability of selecting an adversarial rather than a random training mask. No masking disables both perturbations.

Training masks	MMM2	27M_vs_30M
No masking	38.5	28.2
Random-only ($\rho = 0$)	87.5	83.7
Adversarial-only ($\rho = 1$)	93.8	90.9
Hybrid ($\rho = 0.8$)	**97.9**	**95.5**

6 Conclusion and Limitation

In this work, we address the communication disconnection problem in MARL frameworks. we propose PatMask as a pluggable, versatile mechanism to enhance the robustness and zero-shot generalization of existing topology-aware MARL algorithms across diverse communication disconnection scenarios. Experiments on SMAC, SMACv2, and LBF demonstrate improved win rates under the evaluated disconnections, with transfer across failure types and probabilities without retraining. Integrations with GAT, TGCNET, and GCN support compatibility with the tested coordination networks. Component ablations show that hybrid masking exceeds random-only training by 10.4 and 11.8 percentage points on MMM2 and 27M_vs_30M, respectively. Limitations include unassessed robustness against independently optimized test-time topology attacks (Appendix D).

AI USE STATEMENT

In this work, we used generative AI tools for formulating mathematical claims and assisting in the writing of proofs. Additionally, we used generative AI tools for correcting grammar errors and improving readability. We have *NOT* used generative AI tools to implement methods, design or provide feedback on research methodology or experiments, help develop theoretical models or conceptual frameworks, assist with translation, clean and reformat datasets, support qualitative and thematic data analysis, or interpret results. Generating synthetic datasets, providing critical ingredients for proving mathematical claims, and proposing or refining hypotheses are *NOT applicable* to this work. We have reviewed all AI-assisted work. For example, LLM-assisted proof was verified for correctness. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REPRODUCIBILITY STATEMENT

We have taken steps to improve the reproducibility of our experiments. The method is described in Section 4. The experimental protocol, including benchmark selection, baselines, training and evaluation setup, evaluation metrics, and the communication-disconnection conditions used at test time, is described in Section 5.1. The synthetic disconnection modes are provided in Appendix A.1. The detailed benchmark descriptions are provided in Appendix B.1, and the main hyperparameters, network settings, and training budgets are reported in Appendix B.2. The source code, scripts, and archived logs of major experiments are provided in the supplementary materials for anonymous review, as indicated by the footnote on page 1. The accompanied document introduces required environment setup, software installation, benchmark configuration, and the unified training interface used to run the reported experiments.

REFERENCES

Daniel S Bernstein, Robert Givan, Neil Immerman, and Shlomo Zilberstein. The complexity of decentralized control of markov decision processes. *Mathematics of operations research*, 2002.

Baiming Chen, Mengdi Xu, Zuxin Liu, Liang Li, and Ding Zhao. Delay-aware multi-agent reinforcement learning for cooperative and competitive environments. *arXiv preprint arXiv:2005.05441*, 2020.

Baiming Chen, Mengdi Xu, Liang Li, and Ding Zhao. Delay-aware model-based reinforcement learning for continuous control. *Neurocomputing*, 450:119–128, 2021.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In Alessandro Moschitti, Bo Pang, and Walter Daelemans (eds.), *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724–1734, Doha, Qatar, October 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1179. URL https://aclanthology.org/D14-1179/.

Filippos Christianos, Lukas Schäfer, and Stefano Albrecht. Shared experience actor-critic for multi-agent reinforcement learning. *Advances in neural information processing systems*, 33:10707–10717, 2020.

Shifei Ding, Wei Du, Ling Ding, Lili Guo, and Jian Zhang. Learning efficient and robust multi-agent communication via graph information bottleneck. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 17346–17353, 2024.

Ziluo Ding, Tiejun Huang, and Zongqing Lu. Learning individually inferred communication for multi-agent cooperation. *Advances in neural information processing systems*, 33:22069–22079, 2020.

Wei Duan, Jie Lu, and Junyu Xuan. Group-aware coordination graph for multi-agent reinforcement learning. In Kate Larson (ed.), *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, IJCAI-24*, pp. 3926–3934. International Joint Conferences on Artificial Intelligence Organization, 8 2024. doi: 10.24963/ijcai.2024/434. URL https://doi.org/10.24963/ijcai.2024/434. Main Track.

Benjamin Ellis, Jonathan Cook, Skander Moalla, Mikayel Samvelyan, Mingfei Sun, Anuj Mahajan, Jakob Foerster, and Shimon Whiteson. Smacv2: An improved benchmark for cooperative multi-agent reinforcement learning. *Advances in Neural Information Processing Systems*, 36:37567–37593, 2023.

A Gleave, M Dennis, N Kant, C Wild, S Levine, and S Russsell. Adversarial policies: Attacking deep reinforcement learning. *In Proc. ICLR-20*, 2020.

Sandy Huang, Nicolas Papernot, Ian Goodfellow, Yan Duan, and Pieter Abbeel. Adversarial attacks on neural network policies. In *International Conference on Learning Representations (ICLR)*, 2017.

Garud N Iyengar. Robust dynamic programming. *Mathematics of Operations Research*, 30(2): 257–280, 2005.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations (ICLR)*, 2017.

Bo Jiang, Ziyan Zhang, Doudou Lin, Jin Tang, and Bin Luo. Semi-supervised learning with graph learning-convolutional networks. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11313–11320, 2019.

Jiechuan Jiang and Zongqing Lu. Learning attentional communication for multi-agent cooperation. *Advances in neural information processing systems*, 31, 2018.

Daewoo Kim, Sangwoo Moon, David Hostallero, Wan Ju Kang, Taeyoung Lee, Kyunghwan Son, and Yung Yi. Learning to schedule communication in multi-agent reinforcement learning. In *ICLR 2019: International Conference on Representation Learning*. International Conference on Representation Learning, 2019a.

Woojun Kim, Myungsik Cho, and Youngchul Sung. Message-dropout: An efficient training method for multi-agent deep reinforcement learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pp. 6079–6086, 2019b.

Mengxian Li, Qi Wang, and Yongjun Xu. Gtde: Grouped training with decentralized execution for multi-agent actor-critic. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 18368–18376, 2025.

Ryan Lowe, Yi I Wu, Aviv Tamar, Jean Harb, OpenAI Pieter Abbeel, and Igor Mordatch. Multi-agent actor-critic for mixed cooperative-competitive environments. *Advances in neural information processing systems*, 30, 2017.

Frans A Oliehoek and Christopher Amato. *A concise introduction to decentralized POMDPs*. Springer, 2016.

Gianfranco Parlangeli and Maria Elena Valcher. On the detection and identification of edge disconnections in a multi-agent consensus network. *arXiv preprint arXiv:2101.06728*, 2021.

Xue Bin Peng, Marcin Andrychowicz, Wojciech Zaremba, and Pieter Abbeel. Sim-to-real transfer of robotic control with dynamics randomization. In *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 3803–3810. IEEE, 2018.

Lerrel Pinto, James Davidson, Rahul Sukthankar, and Abhinav Gupta. Robust adversarial reinforcement learning. In *International Conference on Machine Learning (ICML)*, pp. 2817–2826. PMLR, 2017.

Rongrong Qian, Zhisheng Duan, Yuan Qi, Tao Peng, and Wenbo Wang. Connecting disconnected agents in multiagent systems via federated control. In *2024 European Control Conference (ECC)*, pp. 2786–2793. IEEE, 2024.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. Monotonic value function factorisation for deep multi-agent reinforcement learning. *Journal of Machine Learning Research*, 21(178):1–51, 2020.

Aryaman Reddi, Maximilian Tölle, Jan Peters, Georgia Chalvatzaki, and Carlo D'Eramo. Robust adversarial reinforcement learning via bounded rationality curricula. In *International Conference on Learning Representations*, volume 2024, pp. 3920–3944, 2024.

Kyunghwan Son, Daewoo Kim, Wan Ju Kang, David Earl Hostallero, and Yung Yi. Qtran: Learning to factorize with transformation for cooperative multi-agent reinforcement learning. In *International conference on machine learning*, pp. 5887–5896. PMLR, 2019.

Shoucheng Song, Youfang Lin, Sheng Han, Chang Yao, Hao Wu, Shuo Wang, and Kai Lv. Code: Communication delay-tolerant multi-agent collaboration via dual alignment of intent and timeliness. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 23304–23312, 2025.

Chuxiong Sun, Zehua Zang, Jiabao Li, Jiangmeng Li, Xiao Xu, Rui Wang, and Changwen Zheng. T2mac: Targeted and trusted multi-agent communication through selective engagement and evidence-driven integration. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 15154–15163, 2024.

Gokul Swamy, Siddharth Reddy, Sergey Levine, and Anca D Dragan. Scaled autonomy: Enabling human operators to control robot fleets. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5942–5948. IEEE, 2020.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

Jianhao Wang, Zhizhou Ren, Terry Liu, Yang Yu, and Chongjie Zhang. Qplex: Duplex dueling multi-agent q-learning.

Jingwei Wang, Qianyue Hao, Wenzhen Huang, Xiaochen Fan, Zhentao Tang, Bin Wang, Jianye Hao, and Yong Li. Dyps: Dynamic parameter sharing in multi-agent reinforcement learning for spatio-temporal resource allocation. In *Proceedings of the 30th ACM SIGKDD conference on knowledge discovery and data mining*, pp. 3128–3139, 2024.

Tonghan Wang, Jianhao Wang, Chongyi Zheng, and Chongjie Zhang. Learning nearly decomposable value functions via communication minimization. In *8th International Conference on Learning Representations, ICLR 2020*, 2020.

S Whiteson, M Samvelyan, T Rashid, CS De Witt, G Farquhar, N Nardelli, TGJ Rudner, CM Hung, PHS Torr, and J Foerster. The starcraft multi-agent challenge. In *Proceedings of the International Joint Conference on Autonomous Agents and Multiagent Systems, AAMAS*, pp. 2186–2188, 2019.

Guang Yang, Tianpei Yang, Jingwen Qiao, Yanqing Wu, Jing Huo, Xingguo Chen, and Yang Gao. Multi-agent reinforcement learning with communication-constrained priors. volume 38, pp. 154040–154060, 2026.

Donghao Ying, Yunkai Zhang, Yuhao Ding, Alec Koppel, and Javad Lavaei. Scalable primal-dual actor-critic method for safe multi-agent rl with general utilities. volume 36, pp. 36524–36539, 2023.

Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of ppo in cooperative multi-agent games. volume 35, pp. 24611–24624, 2022.

Lebin Yu, Yunbo Qiu, Quanming Yao, Yuan Shen, Xudong Zhang, and Jian Wang. Robust communicative multi-agent reinforcement learning with active defense. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 17575–17582, 2024.

Lei Yuan, Jianhao Wang, Fuxiang Zhang, Chenghe Wang, Zongzhang Zhang, Yang Yu, and Chongjie Zhang. Multi-agent incentive communication via decentralized teammate modeling. In *Proceedings of the AAAI conference on artificial intelligence*, volume 36, pp. 9466–9474, 2022.

Tingting Yuan, Hwei-Ming Chung, Jie Yuan, and Xiaoming Fu. Dacom: Learning delay-aware communication for multi-agent reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 11763–11771, 2023.

Yuan Yuan, Huanhuan Yuan, Lei Guo, Hongjiu Yang, and Shanlin Sun. Resilient control of networked control system under dos attacks: A unified game approach. *IEEE transactions on Industrial Informatics*, 12(5):1786–1794, 2016.

Lijun Zhang, Lin Li, Wei Wei, Huizhong Song, Yaodong Yang, and Jiye Liang. Scalable constrained policy optimization for safe multi-agent reinforcement learning. volume 37, pp. 138698–138730, 2024.

Sai Qian Zhang, Qi Zhang, and Jieyu Lin. Succinct and robust multi-agent communication with temporal message control. *Advances in neural information processing systems*, 33:17271–17282, 2020.

Yuyang Zhang, Runyu Zhang, Yuantao Gu, and Na Li. Multi-agent reinforcement learning with reward delays. In *Learning for Dynamics and Control Conference*, pp. 692–704. PMLR, 2023.

Zhuohui Zhang, Bin He, Bin Cheng, and Gang Li. Bridging training and execution via dynamic directed graph-based communication in cooperative multi-agent systems. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pp. 23395–23403, 2025.

Ming Zhou, Jun Luo, Julian Villella, Yaodong Yang, David Rusu, Jiayu Miao, Weinan Zhang, Montgomery Alban, Iman Fadakar, Zheng Chen, et al. Smarts: An open-source scalable multi-agent rl training school for autonomous driving. In *Conference on robot learning*, pp. 264–285. PMLR, 2021.

APPENDIX CONTENTS

A Supplementary Methodology and Analysis **15**

A.1 Structured Communication Disconnection Modes 15

A.2 Robust Policy Evaluation under Topology Perturbations 15

A.3 Training Procedure and Stability 16

B Experimental Setup **16**

B.1 Benchmark Descriptions 16

B.2 Hyperparameter Settings and Evaluation Metrics 18

C Additional Experiments and Analysis **19**

C.1 Additional Baseline Comparisons 19

C.2 Results under Structured Disconnections 20

C.3 Hyperparameter Sensitivity 21

C.4 Generator Complexity and Training Cost 21

C.5 Performance under Perfect Communication 22

C.6 Case Study 22

D Scope and Limitations **22**

A Supplementary Methodology and Analysis

A.1 Structured Communication Disconnection Modes

We define two synthetic disconnection modes that introduce observation-dependent link reliability and correlated group outages. The simulator applies these failures without using the trained adversarial generator. Agents receive only the messages delivered through surviving links. These modes test robustness beyond independent failures, without assuming that deployment failures follow the training adversary's distribution.

A.1.1 Observation-Conditioned Link Disconnection ($D_{\text{obs-dist}}$)

Beyond i.i.d. Bernoulli link failures, we consider a synthetic disconnection mode in which edge reliability depends on agents' observation dissimilarity at time t. Let $\mathbf{o}_t = (o_t^1, \dots, o_t^N)$ denote the joint observation profile. For $i \neq j$, we define a pairwise dissimilarity $D_{t,ij}^{\text{obs}} \in \mathbb{R}_{\geq 0}$, such as $D_{t,ij}^{\text{obs}} = \|o_t^i - o_t^j\|_2$.

We normalize off-diagonal dissimilarities within each decision step by the maximum pair:

$$\tilde{D}_{t,ij} = \begin{cases} 0, & i = j, \\ \dfrac{D_{t,ij}^{\text{obs}}}{\max_{k \neq \ell} D_{t,k\ell}^{\text{obs}} + \varepsilon}, & i \neq j. \end{cases} \tag{12}$$

We define the link-retention probability to decrease exponentially with normalized observation dissimilarity:

$$P_{t,ij}^{\text{keep}}(\mathbf{o}_t) = (1 - p_\ell) \exp\big(-\beta\, \tilde{D}_{t,ij}\big), \qquad i \neq j, \tag{13}$$

with hyperparameters $p_\ell \in [0,1]$ (baseline edge-drop probability) and $\beta > 0$ (severity). Conditional on $\mathbf{o}_t$, peer-to-peer link indicators are sampled as $I_t^{(i,j)} \sim \text{Bernoulli}\Big(P_{t,ij}^{\text{keep}}(\mathbf{o}_t)\Big)$ for $i < j$, and symmetrized for undirected channels.

Optionally, node survival may be biased by deviation from the team observation centroid $\bar{o}_t = \frac{1}{N}\sum_i o_t^i$:

$$\delta_t^i = \|o_t^i - \bar{o}_t\|_2, \quad \hat{\delta}_t^i = \frac{\delta_t^i}{\max_k \delta_t^k + \varepsilon}, \quad p_{t,i}^{(n)} = \text{clip}\Big(p_n + \lambda \hat{\delta}_t^i,\, [0,1]\Big), \tag{14}$$

and $J_t^i \sim \text{Bernoulli}(1 - p_{t,i}^{(n)})$, where $\lambda \geq 0$ controls observation-induced node fragility.

A.1.2 Regional Disconnection (D_{reg})

To construct correlated group outages, at each step we sample a seed agent $S_t \sim \text{Uniform}(\mathcal{N})$ and define its neighborhood in observation space. Define distances $c_t^i = \|o_t^i - o_t^{S_t}\|_2$ and a radius $R_t = \rho \max_{j \in \mathcal{N}} c_t^j$ with $\rho \in (0,1]$. Let $\mathcal{N}_t^{\text{near}} = \{i \in \mathcal{N} : c_t^i \leq R_t\}$. Two Bernoulli gates trigger a regional incident:

$$U_t^{\text{app}} \sim \text{Bernoulli}(p_{\text{app}}), \qquad U_t^{\text{kill}} \sim \text{Bernoulli}(p_{\text{kill}}). \tag{15}$$

A regional kill is applied iff $U_t^{\text{app}} U_t^{\text{kill}} = 1$, upon which all agents in $\mathcal{N}_t^{\text{near}}$ suffer node failures ($J_t^i = 0$). Edge indicators may additionally follow an independent Bernoulli edge process at rate p_ℓ to reflect residual link noise. This induces correlated failures among agents with similar observations, while not explicitly modeling physical propagation or interference.

A.2 Robust Policy Evaluation under Topology Perturbations

The construction specializes robust dynamic programming (Iyengar, 2005) to the process induced by a fixed decentralized policy and an adversary controlling communication availability. The policy, including its message-generation and memory-update rules, remains fixed throughout this evaluation.

Proof. For bounded functions V and W, define

$$F_V(x, \tilde{G}) = r^\pi(x, \tilde{G}) + \gamma \, \mathbb{E}_{P^\pi(\cdot|x,\tilde{G})}[V(x')].$$

Since the reward and transition distribution are fixed for each $(x, \tilde{G})$,

$$|F_V(x, \tilde{G}) - F_W(x, \tilde{G})| \le \gamma \|V - W\|_\infty.$$

The non-expansiveness of minimization gives

$$\begin{aligned} |(\mathcal{T}^\pi_{adv} V)(x) - (\mathcal{T}^\pi_{adv} W)(x)| &\le \max_{\tilde{G} \in \mathcal{N}(G(x))} |F_V(x, \tilde{G}) - F_W(x, \tilde{G})| \\ &\le \gamma \|V - W\|_\infty. \end{aligned}$$

Taking the supremum over x proves contraction. Bounded rewards ensure that the operator maps bounded functions to bounded functions. The Banach fixed-point theorem therefore gives a unique fixed point and convergence of exact operator iteration.

To identify this fixed point, start from $V_0 = 0$. Because the adversary selects an admissible topology at each step before the policy acts, with no additional constraints coupling choices across time, backward induction shows that $V_k = (\mathcal{T}^\pi_{adv})^k V_0$ is the worst-case k-step discounted value. If $|r| \le R_{\max}$, the remaining discounted return is bounded uniformly by $\gamma^k R_{\max}/(1-\gamma)$. Thus V_k converges to the infinite-horizon worst-case value V^π_{rob}. □

Connection to training. For an initial-state distribution μ, the ideal robust learning objective is to maximize $\mathbb{E}_{x_0 \sim \mu}[V^{\pi_\theta}_{\mathrm{rob}}(x_0)]$. The theorem characterizes the evaluation of each fixed policy; it does not establish that PatMask optimizes this objective exactly. The neural generator uses TD-error maximization to select training masks, while the learner updates its value estimates and policy. Replay sampling, Gumbel relaxation, and alternating updates do not by themselves provide an approximation error bound or a convergence guarantee for this coupled training procedure.

A.3 Training Procedure and Stability

Algorithm 1 summarizes the alternating updates of the generator and MARL learner.

Interpreting training dynamics. The generator maximizes TD loss, while the learner minimizes it under the generated masks. Each player's updates change the other player's optimization problem, and replay samples and TD targets also evolve during training. Neither loss is therefore guaranteed to decrease monotonically. Note that oscillations alone establish neither convergence nor divergence. Their magnitude and evolution require empirical examination.

Figure 2 reports policy performance throughout training under externally imposed communication failures. These curves provide evidence about learned policy quality, but do not establish convergence of the generator or the joint optimization dynamics. The contraction theorem concerns exact robust evaluation of a fixed policy, rather than the coupled neural training updates.

B Experimental Setup

B.1 Benchmark Descriptions

We evaluate PatMask on three standard cooperative benchmarks: SMAC, SMACv2, and LBF. These environments cover different coordination challenges under partial observability and communication disruption, and together they test both tactical robustness and generalization.

B.1.1 SMAC

SMAC consists of micromanagement combat tasks in which decentralized agents control units to defeat enemy teams. The environments are partially observable, with discrete action spaces for movement, attack, and support behaviors. Tasks are grouped by difficulty as Easy, Hard, and Super Hard, and span both homogeneous and heterogeneous unit compositions. Representative scenarios include 10M_vs_11M, 5M_vs_6M, MMM2, 27M_vs_30M, 8M_vs_9M, and 1o_10b_vs_1r. These maps capture different coordination demands, including focus firing, kiting, role specialization, and large-scale battle management.

Algorithm 1 Adversarial Training for PatMask

Require: Parameters θ (Agent), ϕ (Generator); Adversarial ratio ρ; Replay buffer $\mathcal{B}$.

Initialize θ, ϕ and target network θ^-.
for each episode $= 1, \ldots, K$ **do**
  Observe initial state s_0 and initialize hidden states h_0.
  for time step $t = 0, \ldots, T$ **do**
    // Topological Perturbation (Hybrid Strategy):
    Sample soft mask M_{gen} via Gumbel sampling (Eq. 6).
    Determine mask source based on adversarial ratio ρ (Eq. 7):
    if Uniform$(0, 1) < \rho$ **then**
      Set active mask $\tilde{M}_t \leftarrow M_{gen}$.
    else
      Sample random mask $\tilde{M}_t \sim$ Bernoulli(p_{rand}).
    end if
    Obtain perturbed topology $\tilde{G}_t = G \odot \tilde{M}_t$ as defined in Eq. 4.
    Extract features using GAT with residual connections based on $\tilde{G}_t$.
    Select action $a_t = \arg\max_a Q_\theta(s_t, \tilde{G}_t, a)$ (with ε-greedy exploration).
    Execute a_t, observe r_t and s_{t+1}, store (s_t, a_t, r_t, s_{t+1}) in $\mathcal{B}$.
  end for
  // Alternating Adversarial Optimization:
  Sample mini-batch from $\mathcal{B}$.
  // Stage 1: Update Generator
  for each generator update **do**
    Compute detached TD targets and $\mathcal{L}_{TD}(\theta, \phi)$ using Eq. 10.
    Hold θ fixed and update ϕ using Adam to minimize $\mathcal{L}_{adv} = -\mathcal{L}_{TD}$.
  end for
  // Stage 2: Update MARL Learner
  Recompute detached TD targets and $\mathcal{L}_{TD}(\theta, \phi)$ with the updated generator.
  Hold ϕ fixed and update θ using Adam to minimize $\mathcal{L}_{TD}(\theta, \phi)$.
  Update target network $\theta^- \leftarrow \eta_{\text{target}}\theta + (1 - \eta_{\text{target}})\theta^-$.
end for

B.1.2 SMACv2

SMACv2 extends SMAC with randomized team compositions and spawn configurations, increasing distribution shift and testing zero-shot robustness. We use two 10-vs-10 settings: Terran Gen and Protoss Gen. Both scenarios preserve the same core combat mechanics but vary unit types and initial conditions, forcing agents to adapt to unfamiliar layouts and heterogeneous role assignments rather than memorize fixed opening strategies.

B.1.3 LBF

The LBF benchmark is a grid-world coordination task in which agents must move and collect food items whose level requirements depend on the combined levels of nearby agents. We use the 19×19-9p-4f setting, which creates a sparse-reward cooperation problem: agents must explore the map, coordinate movement, and gather food efficiently under partial observability. This benchmark complements SMAC by testing coordination under positional constraints and group-level credit assignment rather than unit-based combat tactics.

These benchmark choices are intended to evaluate whether PatMask improves robustness under communication disconnection across both combat-style coordination and more general cooperative planning tasks.

B.2 Hyperparameter Settings and Evaluation Metrics

Metrics. An episode consists of a sequence of environment steps from initialization to termination. The test win rate is evaluated periodically every $10,000$ steps via 32 independent episodes. It is defined as the percentage of episodes where agents eliminate all enemy units within the maximum time limit.

Hyperparameters. Detailed hyperparameters are listed in Tables 7, 8 and 9.

Table 7: Hyperparameters used for PyMARL under SMAC, SMACv2 and LBF.

Hyperparameter	**SMAC**	**SMACv2**	**LBF** (19×19-**9p-4f**)
Buffer size	5000	5000	5000
Batch size	32	32	32
Learning rate	5×10^{-4}	5×10^{-4}	5×10^{-4}
Start epsilon ε_s	1.0	1.0	1.0
Finish epsilon ε_f	0.05	0.05	0.05
ε decay steps	50000	50000	50000
Evaluate interval	10000	10000	10000
Evaluate times	32	100	100
Target update interval	200	200	200
Discount factor γ	0.99	0.99	0.99
$t_{\max}$	2050000	2050000	2050000

Table 8: Network Architecture Hyperparameters.

Parameter	**Value**
RNN Hidden Dimension	64
GAT Dimension	64
GAT Attention Heads	1
Mask Generator Hidden Dimension	64
Activation Function	ReLU

Table 9: Adversarial mask probability ρ settings across different scenarios.

Scenario / Map	**Value** (ρ)
1o_10b_vs_1r	0.2
3m, 8m, 2s3z, MMM2, SMACv2, LBF	0.5

C Additional Experiments and Analysis

C.1 Additional Baseline Comparisons

Random-dropout training. Table 10 reports additional comparisons with GTDE, TGCNET, T2MAC, and GACG trained using random communication dropout at rate 0.3. Evaluation uses MMM2 under mixed disconnection with $p = 0.5$, following Table 5. The comparison examines whether the evaluated baselines benefit from communication-failure exposure during training.

Table 10: Test win rates on MMM2 under mixed disconnection with $p = 0.5$. Dropout-augmented baselines use a training dropout rate of 0.3.

Method	Training configuration	Win rate (%)
GTDE	Original	36.9
GTDE	Random dropout	89.7
TGCNET	Original	58.1
TGCNET	Random dropout	91.2
T2MAC	Original	55.8
T2MAC	Random dropout	72.3
GACG	Original	42.7
GACG	Random dropout	62.7
PatMask	Adversarial and random masking	95.2

All four baselines obtain higher reported win rates with random-dropout training. The strongest dropout-trained baseline in this comparison, TGCNET, achieves 91.2%, compared with 95.2% for PatMask. These results show that random failure exposure is an effective baseline and that PatMask achieves a higher reported win rate in the tested setting.

Robustness-focused communication methods. Table 11 reports additional comparisons on MMM2 under mixed disconnection with $p = 0.5$, following the evaluation setting of Table 5.

Table 11: Test win rates on MMM2 under mixed disconnection with $p = 0.5$ for robustness-focused communication methods.

Method	Test win rate (%)
Message-Dropout (Kim et al., 2019b)	18.6
TMC (Zhang et al., 2020)	12.4
MAGI (Ding et al., 2024)	9.7
ADMAC (Yu et al., 2024)	10.6
PatMask	95.2

PatMask achieves the highest reported win rate in this comparison. These results concern the tested mixed link/node-disconnection setting and do not establish superiority under other communication perturbations, such as message-content corruption.

C.2 Results under Structured Disconnections

We evaluate observation-conditioned link disconnection ($D_{\text{obs-dist}}$) and regional disconnection (D_{reg}) on MMM2. Table 12 specifies the two parameter configurations, labeled *Sparse* and *Dense*, for each mode. Both modes use observation-space relationships to construct communication failures; their parameters are not calibrated to physical distances or measured interference.

Table 12: Hyperparameter configurations for advanced disconnection scenarios.

Hyperparameter	**Obs-Distance** ($D_{\text{obs-dist}}$)		**Regional** (D_{reg})	
	Sparse	**Dense**	**Sparse**	**Dense**
Node Failure Prob. (p_{node})	0.20	0.60	0.20	0.60
Edge Failure Prob. (p_{edge})	0.05	0.20	0.05	0.20
Distance Penalty (β)	2.0	4.0	-	-
Regional Apply Prob. (p_{apply})	-	-	1.0	1.0
Regional Radius Frac. (r_{frac})	-	-	0.3	0.5
Regional Kill Prob. (p_{kill})	-	-	0.35	0.80

Table 13 reports performance under observation-conditioned link failures and correlated group outages on MMM2. PatMask achieves the highest reported mean win rate in all four settings. Under the dense observation-conditioned and regional settings, its win rates are 96.48% and 94.97%, respectively, compared with 56.50% and 80.00% for TGCNET, the strongest evaluated baseline in these settings. These results extend the evaluation beyond independent failures to the specified observation-dependent and correlated availability patterns.

Table 13: Test win rates (%) under advanced communication disconnections in the MMM2 map. Evaluated over 200 environment steps per mode.

Method	**Obs-Distance** ($D_{\text{obs-dist}}$)		**Regional** (D_{reg})	
	Sparse	**Dense**	**Sparse**	**Dense**
GTDE	64.82 ± 8.34	42.21 ± 11.20	79.90 ± 6.55	66.33 ± 9.42
GACG	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00	0.00 ± 0.00
T2MAC	48.20 ± 18.45	31.50 ± 22.30	44.10 ± 19.80	36.80 ± 24.15
TGCNET	$\underline{85.00 \pm 4.20}$	$\underline{56.50 \pm 7.85}$	$\underline{89.00 \pm 3.50}$	$\underline{80.00 \pm 6.10}$
PatMask (Ours)	$\mathbf{94.97 \pm 0.85}$	$\mathbf{96.48 \pm 0.62}$	$\mathbf{93.97 \pm 1.12}$	$\mathbf{94.97 \pm 0.94}$

C.3 HYPERPARAMETER SENSITIVITY

Mask-sampling temperature. Table 14 reports test win rates on MMM2 for four Gumbel-Softmax temperatures. The temperature controls the training-time mask relaxation, and the mask generator is removed during execution.

Table 14: Sensitivity to the Gumbel-Softmax temperature τ on MMM2.

Temperature τ	Test win rate (%)
0.1	86.8 ± 3.6
0.5	93.0 ± 1.8
1.0	94.7 ± 1.4
2.0	92.8 ± 1.9

Among the tested values, $\tau = 1.0$ achieves the highest reported mean win rate. The means at $\tau = 0.5$ and $\tau = 2.0$ are within 1.9 percentage points of this value, whereas $\tau = 0.1$ yields a lower mean. These results characterize temperature sensitivity on MMM2.

Adversarial-mask probability. Table 15 compares two values of ρ on MMM2 for PatMask-GAT and PatMask-TGCNET.

Table 15: Sensitivity to the adversarial-mask probability ρ on MMM2. Test win rates (%) are reported as mean $\pm$ standard deviation.

ρ	PatMask-GAT	PatMask-TGCNET
0.2	91.2 ± 2.6	93.7 ± 1.2
0.5	93.8 ± 2.1	88.4 ± 4.6

Increasing ρ from 0.2 to 0.5 raises the reported mean win rate of PatMask-GAT by 2.6 percentage points, while lowering that of PatMask-TGCNET by 5.3 points. These results demonstrate backbone-dependent sensitivity that a larger adversarial-mask probability does not uniformly improve performance. Architectural compatibility therefore does not imply that a single training mixture is preferable across backbones.

C.4 GENERATOR COMPLEXITY AND TRAINING COST

Generator complexity. The generator predicts an $N \times N$ mask-probability matrix, containing N^2 entries (729 for $N = 27$). The exponential complexity concerns exhaustive enumeration of binary topologies, which PatMask does not perform. Output size alone does not determine total training cost, which also includes policy learning and alternating adversarial updates. With fixed feature and hidden dimensions, the shared edge predictor contains 18,817 parameters independent of N, and evaluating it over all ordered agent pairs requires $O(N^2)$ work.

Training cost. Table 16 reports end-to-end training time on 27M_vs_30M with $t_{\max} = 2{,}050{,}000$ environment steps for both methods. PatMask-GAT requires 25.22 h compared with 15.84 h for GAT, an additional 9.38 h or 59.2%, while the reported final win rate increases from 88% to 96%. These timings measure the total cost of the training procedure rather than isolated generator execution. The generator is removed after training and therefore adds no generator computation during execution.

Table 16: End-to-end training cost on 27M_vs_30M with $t_{\max} = 2{,}050{,}000$ environment steps for both methods.

Method	Training time (h)	Relative time	Final win rate (%)
GAT	15.84	$1.00\times$	88
PatMask-GAT	25.22	$1.59\times$	96

This comparison quantifies the computational trade-off on the 27-agent task. It does not establish a runtime scaling law across team sizes. The learning curves in Fig. 2 provide empirical training trajectories, while convergence of the practical alternating optimization remains unproven.

C.5 Performance under Perfect Communication

Adversarial training may trade performance under perfect communication for robustness to failures. Table 17 examines this trade-off for GAT and PatMask-GAT on MMM2. With no test-time disconnection, their reported win rates are 96.8% and 96.1%, respectively, a difference of 0.7 percentage points. At 30% link disconnection, PatMask-GAT achieves 94.7% compared with 65.3% for GAT.

These results show a small observed performance difference under perfect communication alongside improved performance under the tested failures. They do not establish the absence of over-conservatism across tasks or failure distributions, nor do they isolate the effect of the generator's access to global state. The zero-disconnection results are reported without uncertainty estimates, so the statistical significance of the 0.7-point difference is undetermined.

Table 17: Experimental Validation (Zero Disconnection Regime). Test win rates (%) on MMM2 across different test-time link disconnection rates.

Method	**0%**	**10%**	**20%**	**30%**
GAT Baseline	$\mathbf{96.8}$	88.4	72.1	65.3 ± 11.4
PatMask-GAT	96.1	$\mathbf{95.8}$	$\mathbf{95.5}$	$\mathbf{94.7 \pm 1.4}$

C.6 Case Study

To explicitly analyze the impact of communication disconnection on agent behavior, we visualize the decision-making processes in MMM2, 5m_vs_6m, and 8m_vs_9m scenarios (see Fig. 4). In the heterogeneous MMM2 scenario, the GACG baseline suffers from coordination collapse where the Medivac remains idle due to missing health updates (Fig. 4a), whereas PatMask successfully recovers the healing action by reconstructing semantic context (Fig. 4b). This limitation extends to homogeneous scenarios where formation control is vital: GACG exhibits a "straggler phenomenon" in 5m_vs_6m (Fig. 4c) that worsens in the denser 8m_vs_9m map (Fig. 4e), leading to unit isolation and rapid elimination. Conversely, PatMask maintains high-level tactical cooperation under the same constraints; as observed in Fig. 4d and Fig. 4f, it enables advanced "cover-and-retreat" maneuvers where low-health soldiers retreat while teammates provide cover fire, thereby preserving group cohesion and combat sustainability.

D Scope and Limitations

Synthetic communication disconnection modes. The evaluation isolates the effect of missing communication through controlled masks on simulated MARL benchmarks. Observation-conditioned and regional masks introduce observation dependence and correlated outages, but are not calibrated wireless propagation models. The results support robustness within the tested availability patterns; transfer to physical communication systems or measured deployment failure traces remains unvalidated.

Training paradigm. PatMask assumes access to centralized information during training, while execution uses local histories and available messages. Decentralized training methods, including scalable primal-dual actor-critic (Ying et al., 2023) and Scal-MAPPO-L (Zhang et al., 2024), address learning with local interactions and safety constraints. Our choice of CTDE preserves centralized value estimation for the present implementation; it does not imply that decentralized training is unsuitable or generally inferior. Extending PatMask to settings without centralized training information remains outside the present evaluation.

Actor-critic applicability. The current implementation trains a value-decomposition learner using TD loss. Applying adversarial topology masking to communicating actor-critic policies would require specifying how masks affect actor and critic inputs, adapting the adversarial objective, and



(a) GACG (MMM2): Coordination collapse (Idle Medivac).



(b) PatMask (MMM2): Robust coordination (Active Healing).



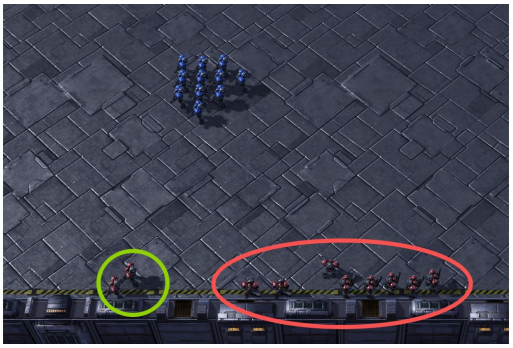
(c) GACG (5m_vs_6m): Movement desynchronization (1 Straggler).



(d) PatMask (5m_vs_6m): Cover & Retreat tactics.



(e) GACG (8m_vs_9m): Scaled failure (2 Stragglers).



(f) PatMask (8m_vs_9m): Cohesive formation.

Figure 4: Visualization of agent behaviors under communication disconnection across different scenarios. **Top Row (MMM2):** The Baseline fails to heal, while PatMask recovers the specific healing action. **Middle Row (5m_vs_6m):** The Baseline suffers from unit isolation, whereas PatMask enables low-health units to retreat under cover. **Bottom Row (8m_vs_9m):** The Baseline's coordination issues scale up with more units detaching, contrasting with PatMask's maintained group cohesion.

evaluating the resulting optimization. Comparisons against actor-critic baselines do not establish this extension.

Communication attack scope. PatMask addresses communication availability loss. It does not detect or correct corrupted, injected, or deceptive messages. If a separate defense discards suspicious messages, PatMask could potentially improve tolerance to the resulting missing communication, but this combination has not been evaluated. A direct extension would require an adversary that can modify message contents, explicit constraints on its capabilities, and evaluation under the corresponding attacks. The present masking experiments do not establish robustness to such attacks.

Adversarial evaluation. Our experiments evaluate robustness to specified disconnection processes, rather than independently optimized attacks against the final policy. Training performance under an evolving generator is not a held-out adversarial evaluation. Moreover, the generator maximizes a TD-loss surrogate, which does not certify that its masks minimize task return. The reported results therefore do not establish worst-case robustness. Evaluating frozen policies against separately optimized topology attacks remains future work.

Evaluation coverage. Our empirical conclusions concern the evaluated tasks, failure modes, and disconnection probabilities. They do not establish performance under near-total or complete communication loss. Learned mask sparsity describes the training perturbations and does not establish a performance ordering between partial and full disconnection.

Perfect communication. The results in Appendix C.5 characterize the robustness–performance trade-off on MMM2; they do not establish its magnitude across tasks.