Graph-Conditioned Diffusion for Offline Multi-Agent Reinforcement Learning

Luis Pimentel*
Georgia Institute of Technology
Atlanta, GA, United States
lpimentel3@gatech.edu

Minwoo Cho*†
Georgia Institute of Technology
Atlanta, GA, United States
mcho318@gatech.edu

Sean Ye*
Zoox
Foster City, CA, United States
sye@zoox.com

James Ellis Grant Pagan
Sandia National Laboratories
Albuquerque, NM, United States
jepagan@sandia.gov

Matthew Gombolay
Georgia Institute of Technology
Atlanta, GA, United States
matthew.gombolay@cc.gatech.edu

ABSTRACT

Multi-agent reinforcement learning struggles with scalability and real-world applicability, as the high interaction variability across team compositions limits the effectiveness of online adaptive methods. Alternatively, offline RL can address these limitations by leveraging diverse offline data to facilitate learning across teams. However, existing offline RL methods fail to produce multi-agent policies that can both adapt using only offline data and coordinate effectively under decentralized execution. To address these challenges, we present Graph Conditioned Diffusion (GCD), a multi-agent diffusion framework that uses graph-based communication to learn generalizable offline policies and maintain decentralization during execution. Our framework leverages the conditional generative modeling ability of diffusion models to learn multi-modal distributions of trajectories across team compositions by conditioning on team communication embeddings. We then adapt coordination online through classifier-free guidance, which steers the generative process toward behaviors that generalize across team compositions. We evaluate our method on the StarCraft II Multi-Agent Challenge v2 (SMACv2) domain, demonstrating superior generalization with an average win-rate improvement of 7.4% to 221.4% in unseen team compositions compared to decentralized baselines.

KEYWORDS

Offline MARL; Graph-Conditioned Diffusion; Multi-Agent Communication; Multi-Agent Generalization

ACM Reference Format:
Luis Pimentel, Minwoo Cho, Sean Ye, James Ellis Grant Pagan, and Matthew Gombolay. 2026. Graph-Conditioned Diffusion for Offline Multi-Agent Reinforcement Learning. In *Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), Paphos, Cyprus, May 25 – 29, 2026*, IFAAMAS, 9 pages. https://doi.org/10.65109/BMST1644

*Equal contribution. Our code and appendix are available at CORE-Robotics-Lab/GCD.
†Corresponding author.

Proc. of the 25th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2026), C. Amato, L. Dennis, V. Mascardi, J. Thangarajah (eds.), May 25 – 29, 2026, Paphos, Cyprus. © 2026 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). https://doi.org/10.65109/BMST1644

1 INTRODUCTION

Learning generalizable policies across diverse multi-agent teams is a central challenge in multi-agent reinforcement learning (MARL). This challenge often requires policies to be effective in coordinating teams with heterogeneous agents of diverse capabilities. While heterogeneous teams are well-suited to address complex real-world problems [9, 30], learned policies also need to be adaptive to the varying capabilities across team structures, particularly when team compositions change. MARL policies trained with conventional approaches under fixed team structures can struggle to adapt when those structures change, especially in decentralized settings where disrupted communication links demand flexible coordination.

In critical settings like search and rescue, directly interacting with the environment in an online setting poses safety risks [19, 33], making the offline MARL paradigm particularly valuable as it enables agents to learn from pre-existing datasets [39]. However, offline MARL faces substantial challenges, including extrapolation errors when agents encounter state-action pairs absent from the training dataset. Furthermore, the necessity of multi-agent coordination adds complexity, as suboptimal coordination by any agent can result in widespread failures [4]. These challenges become more pronounced in heterogeneous systems, where scalability depends on data distributions that sufficiently capture diverse interactions across varying team compositions [10]. When such diversity is limited, learning becomes significantly more challenging. Recently, diffusion models have excelled in offline reinforcement learning (RL) tasks [12] through their capacity to model complex multi-modal data distributions [42] and their conditional generative modeling ability, which shapes policy behaviors towards desired outcomes [1]. Building on these strengths, we leverage diffusion models as the foundation of our framework.

In this paper, we address the difficulties of learning generalizable offline policies for diverse multi-agent systems by leveraging diffusion models to learn multi-modal distributions of trajectories. We formulate this problem under an offline meta-MARL framework, where different team compositions are treated as distinct tasks. Moreover, we address the challenge of decentralization by leveraging graph-based multi-agent communication, allowing distributed information sharing during the diffusion process. We evaluate our approach on the widely used StarCraft II Multi-Agent Challenge v2 (SMACv2) domain [3], which incorporates substantial stochasticity,

meaningful partial observability, and highly variable team compositions. These characteristics make SMACv2 a challenging and comprehensive testbed for assessing generalization in multi-agent systems. **Our contributions are as follows:**

- We propose a novel diffusion-based offline MARL architecture with graph-based communication designed for generalizable multi-agent teaming.
- We introduce a novel classifier-free guidance method that conditions on multi-agent communication embeddings to enable generalization across team compositions.
- We perform experiments characterized by the availability of team composition data during training. Our method consistently outperforms baselines on unseen teams, achieving average win-rate improvements from 7.4% to 221.4%.
- We show that our method is flexible to varied communication protocols and that heterogeneous communication protocols result in increased robustness in limited observability, with an average win-rate improvement of 35.8% to 73.3% on unseen tasks.

2 RELATED WORK

Offline Reinforcement Learning: Offline RL has recently received significant attention due to its potential to leverage static datasets without the need for additional environment interactions [17, 19]. However, offline RL often suffers from a distributional shift problem where the data used to train the policy differs from the learned behavior. To address this, recent work leverages value-based learning techniques, including methods that reduce Q-value overestimation [5] by learning policy constraints [17] or by using regularization [16]. Alternatively, non-value based methods have used generative methods like diffusion models to learn reward and trajectory distributions from offline datasets and generate trajectories through guided sampling [1, 12]. We build on this work, leveraging the conditional generative modeling ability of diffusion models to learn multi-modal distributions of trajectories across team compositions.

Communication in Multi-Agent Systems: Multi-agent systems that learn communication protocols show improved coordination performance as agents learn how to effectively share information, leading to more effective collaboration [34, 38]. Recent work employs Graph Neural Networks (GNN) models within multi-agent communication, showing superior performance in modeling the communication structure of multi-agent teams [25]. Building on this, prior work has also leveraged heterogeneous graphs to model class-specific interactions among agents [30]. Motivated by this, we incorporate graph-based communication in our method and extend it to the offline setting. To the best of our knowledge, we are the first to integrate graph-based multi-agent communication within diffusion-based offline MARL.

Multi-Agent Generalization: Generalization of MARL algorithms has focused on learning transferable policies across tasks, where increasing environment and team sizes lead to scalability challenges. Prior work in online MARL addresses these challenges through specialized architectures like GNNs, Transformers, and meta-learning paradigms [11, 24, 40, 41, 47]. However, how to effectively extract task information to enable generalization in multi-agent settings remains an open question [45]. Methods such as MATE [28] and M3 [23] learn to condition policies on task representations, but require additional online learning for effective adaptation. Other methods, such as ODIS [46] and HiSSD [21], also extract common and task-specific skills as conditioning mechanisms for value-based methods, but this approach is inflexible as it biases models toward pre-defined skill partitions, and value-based algorithms that they build on may not be ideal for conditioning.

In contrast, we tackle this problem from an offline MARL perspective by using graph-based multi-agent communication to capture task-relevant information. Our method leverages the diffusion model's inherent conditional generative capabilities to enable policy conditioning, using task embeddings derived from graph-based multi-agent communication as the conditioning signal. A key advantage of our design is its ability to adaptively capture team-level representations that emerge through communication, enabling more robust and scalable generalization.

3 PRELIMINARIES

Multi-Agent Heterogeneous POMDP: Our problem is modeled as a multi-agent heterogeneous partially observable Markov decision process (MAH-POMDP) [30], an extension of the decentralized POMDP (Dec-POMDP) formulation [26], defined by $\mathcal{T} = \langle C, \mathcal{N}, \{\mathcal{S}^{(j)}\}_{j\in C}, \{\mathcal{A}^{(j)}\}_{j\in C}, \{\mathcal{O}^{(j)}\}_{j\in C}, \mathcal{P}, \mathcal{R}, \gamma\rangle$. Here, C denotes the set of available agent classes, and $j \in C$ denotes an agent's class. $\mathcal{N} = \sum_{j\in C} N^{(j)}$ is the total number of agents in the team, where $N^{(j)}$ denotes the number of agents in class j. We define the team composition as the tuple $\mathfrak{m} = \{N^{(j)}\}_{j\in C}$, specifying the distribution of classes within the team. The joint set of state-spaces is denoted as $\mathcal{S} = \{\mathcal{S}^{(j)}\}_{j\in C}$ where each $\mathcal{S}^{(j)} = \left[s_t^{1_j}, s_t^{2_j}, \cdots, s_t^{N_j^{(j)}}\right]$, with $s_t^{i_j}$ representing the state-vector of agent i of the j-th class, at time t. In the same manner, $\mathcal{A} = \{\mathcal{A}^{(j)}\}_{j\in C}$ denotes the joint set of action-spaces where each $\mathcal{A}^{(j)} = \left[a_t^{1_j}, a_t^{2_j}, \cdots, a_t^{N_j^{(j)}}\right]$, with $a_t^{i_j}$ representing the action-vector of agent i of the j-th class, at time t. Similarly, $\mathcal{O} = \{\mathcal{O}^{(j)}\}_{j\in C}$ denotes the joint set of observation-spaces where each $\mathcal{O}^{(j)} = \left[o_t^{1_j}, o_t^{2_j}, \cdots, o_t^{N_j^{(j)}}\right]$, with $o_t^{i_j}$ representing the observation-vector of agent i of the j-th class, at time t. The state transition is denoted as $\mathcal{S}' \sim \mathcal{P}(\mathcal{S}'|\mathcal{S}, \mathcal{A})$ where $\mathcal{P}$ is the state-transition function of the domain. We consider a fully-cooperative domain where a team's shared reward is represented by $\mathcal{R} = R(\mathcal{S}, \mathcal{A})$, and $\gamma \in [0, 1)$ is the temporal discount factor.

Offline Meta-Reinforcement Learning: We consider learning cooperative MARL policies in different team compositions as distinct tasks, each modeled as a separate MAH-POMDP. We formulate our approach within an offline Meta-RL framework [2], where an *inner-loop* process learns task-specific representations and an *outer-loop* learns a generalizable policy capable of adapting across tasks. We learn over distributions of datasets, collected by behavior policies, denoted as $\mathcal{D} = \{\mathcal{D}_\mathfrak{m}\}_{\mathfrak{m}=1}^{M}$ with each $\mathcal{D}_\mathfrak{m}$ containing a fixed number of trajectory samples from the $\mathfrak{m}$-th task in the set of training tasks $\mathcal{T}_{train} = \{\mathcal{T}_\mathfrak{m}\}_{\mathfrak{m}=1}^{M}$. Our learning objective is denoted as $\max_\pi \mathbb{E}_{\mathcal{T}_\mathfrak{m}\sim\mathcal{P}(\mathcal{T}_{train})}\left[\mathcal{R}_\mathfrak{m}(\pi)\right]$, where we learn policies π maximizing the cumulative discounted reward, $\mathcal{R}_\mathfrak{m}$, of each training task, $\mathcal{T}_\mathfrak{m}$, under the training task distribution, $\mathcal{P}(\mathcal{T}_{train})$.

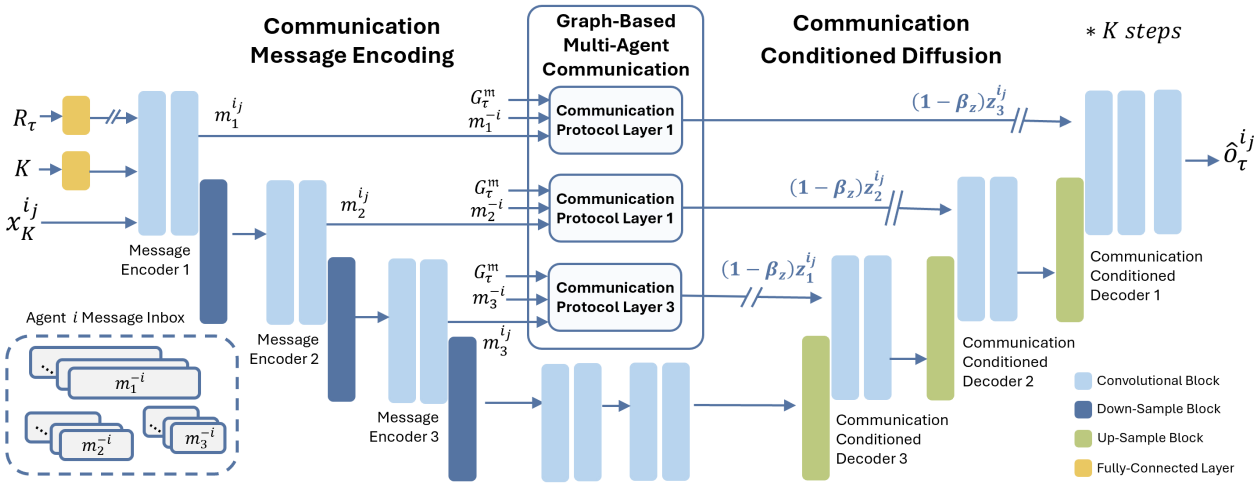

Figure 1: GCD consists of two phases: Communication Message Encoding and Communication Conditioned Diffusion. First communication layers generate embeddings, z_r^{ij}, from observations, messages, and the graph structure $\mathcal{G}_\tau^{\mathfrak{m}}$. Second, these embeddings condition generation through dropout sampling, $(1-\beta_z)$, applied during training to enable classifier-free guidance.

Offline RL with Diffusion Models. Diffusion models [7, 35, 37] are a class of generative models defined by two primary phases: a forward noising process $q(x_k|x_{k-1})$ and a trainable reverse denoising process $p_\theta(x_{k-1}|x_k)$. We use the Denoising Diffusion Probabilistic Model (DDPM) formulation, and create noisy trajectories through $q(x_k|x_0) := \mathcal{N}(x_k; \sqrt{\bar{\alpha}^k}x_0, (1-\bar{\alpha}^k)\boldsymbol{I})$, where $\bar{\alpha}^k$ is a predefined noise scheduler dependent on the diffusion timestep k. A deep neural network $\epsilon_\theta(x_k, k)$ is trained to predict the noise added to x_k at each diffusion step through $\mathcal{L}(\theta) := \mathbb{E}_{x_0\sim q, k\sim[1,K], \epsilon\sim\mathcal{N}(\mathbf{0},\boldsymbol{I})}\left[\|\epsilon - \epsilon_\theta(x_k, k)\|^2\right]$. Samples are then generated via a series of reverse denoising steps, gradually removing noise from a starting Gaussian noise sample $x_K \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$. In the context of offline RL, diffusion models can learn a policy distribution by treating trajectories as data points. We extend diffusion models to model interactions among agents, following prior work [44, 48].

4 METHOD

We introduce **G**raph **C**onditioned **D**iffusion (**GCD**), a novel multi-agent diffusion framework that leverages graph-based multi-agent communication to learn decentralized and generalizable policies. The overall GCD architecture is outlined in Figure 1, and detailed configurations are provided in Appendix A.1.

4.1 Diffusion with Graph Communication

Incorporating communication in MARL naturally aligns with GNN architectures, where effectively modeling the graph topology of agent interactions has shown to be essential for effective coordination [25, 30, 34, 38]. Motivated by this, we adapt graph communication to model information sharing between agents during diffusion process, thereby learning to extract implicit task information from multi-agent communication.

4.1.1 Communication Message Encoding. We first describe how agents generate message encodings for communication within our diffusion architecture, as shown in Figure 1. During offline training, each team-specific data sample is defined as $d_{\mathfrak{m}} = \left[\mathcal{G}_\tau^{\mathfrak{m}}, \mathcal{R}_\tau^{\mathfrak{m}}, \bar{\mathbf{a}}_\tau^{\mathfrak{m}}, \bar{\mathbf{o}}_\tau^{\mathfrak{m}}\right]$. For a given trajectory τ and team composition $\mathfrak{m}$, this sample consists of the graph structure $\mathcal{G}_\tau^{\mathfrak{m}}$, encoding agent connectivity and class-type information, the team reward $\mathcal{R}_\tau^{\mathfrak{m}}$, joint actions $\bar{\mathbf{a}}_\tau^{\mathfrak{m}} = \{a_\tau^{i_{C(i)}}\}_{i=1}^N$, and joint observations $\bar{\mathbf{o}}_\tau^{\mathfrak{m}} = \{o_\tau^{i_{C(i)}}\}_{i=1}^N$. Here, $C(i)$ denotes the class of agent i. For the rest of the paper, we adopt heterogeneous graph notation aligned with the MAH-POMDP setting, which generalizes to other graph-based communication protocols detailed in Appendix A.2.

The initial noisy trajectory of each agent at diffusion step K consists of two parts. First is the history conditioning up to the h-th timestep, denoted as $o_{1:h}^{ij}$, where each $o_t^{ij} \in \mathbb{R}^{d_0}$ is an observation feature at timestep t in the trajectory with a feature dimension of d_0. Second is the diffusion horizon, extending H steps beyond the last history step, denoted as $\tilde{x}_{K,h+1:h+H}^{ij}$, where each $\tilde{x}_{K,t}^{ij} \in \mathbb{R}^{d_0}$ is sampled from a standard normal Gaussian distribution $\mathcal{N}(\mathbf{0}, \boldsymbol{I})$. At each diffusion step k, each agent encodes this information through a set of message encoders, generating messages $\{m_r^{ij}\}_{r=1}^R$, where R is the total number of layers and r is the encoder layer index. Extending the temporal U-Net architecture [12], the message encoders use down-convolutional layers that progressively compress both the time horizon and the feature dimensions such that each message $m_r^{ij} \in \mathbb{R}^{T_r \times d_r}$ has a time dimension of T_r and feature dimension of d_r at each layer r. Formally, we define the message computation as: $m_r^{ij} = \text{MessageEncoder}_r^i(x_k^{ij}) : \mathbb{R}^{T_0\times d_0} \to \mathbb{R}^{T_r\times d_r}$ where $x_k^{ij} = [o_{1:h}^{ij} \,\|\, \tilde{x}_{k,h+1:h+H}^{ij}] \in \mathbb{R}^{T_0\times d_0}$ concatenates the agent's history and diffusion horizon, with total length $T_0 = h + H$.

We note that message encoder parameters are homogeneous and shared across agents as the observation space is consistent across team compositions. However, the heterogeneity of agents is captured by the node-type information in the graph data structure and can be leveraged by heterogeneous communication protocols in the message aggregation phase, as detailed in Section 4.1.2. During each diffusion timestep, each agent enters the message-passing

phase, sending messages to neighboring agents as determined by the topology of $\mathcal{G}_\tau^{\mathfrak{m}}$. Importantly, agents compute their messages via individual encoders, crucial for maintaining decentralization, as the encoders can be distributed during execution.

4.1.2 Communication Embeddings. Following the message-passing phase at each layer, we apply graph-based aggregation to the encoder messages to facilitate inter-agent communication. To take advantage of the heterogeneous structure of our problem, we describe this process using the HetGAT [31] mechanism that incorporates type-specific relational information. Since the message at layer r has shape $\mathbb{R}^{T_r \times d_r}$, we denote its temporal slices as $m_{r,t}^{ij} \in \mathbb{R}^{d_r}$ and apply HetGAT across each agent's time dimension.

First, normalized attention coefficients are computed, $\alpha_{ik}^{l\to j} = \mathrm{softmax}_k\big(\sigma'\left(W_{l\to j}^{att}[W_j m_{r,t}^{ij} \parallel W_{l\to j} m_{r,t}^{k_l}]\right)\big)$, to weigh neighbor messages along the communication edges. Here $l \to j$ denotes an edge-type meaning "from class l to class j", and the respective communicating agents' indexes are k and i. At each layer, the class-based node parameters are denoted as $W_j \in \mathbb{R}^{d_r}$, the class-to-class edge parameters as $W_{l\to j} \in \mathbb{R}^{d_r}$, and the class-to-class attention parameters as $W_{l\to j}^{\text{att}} \in \mathbb{R}^{2d_r}$. Note that this formulation naturally accounts for intra-class communication when $l = j$, allowing agents of the same class to communicate along distinct channels. Next, using these coefficients, we compute the *communication embedding* $z_{r,t}^{ij}$ for agent i of class j by applying a nonlinear activation $\sigma(\cdot)$ to the sum of its transformed embedding $W_j m_{r,t}^{ij}$ and the attention-weighted neighborhood messages:

$$z_{r,t}^{ij} = \sigma\Big(W_j m_{r,t}^{ij} + \sum_{l\in C}\sum_{k\in N_l(i)} \alpha_{ik}^{l\to j} m_{r,t}^{k_l}\Big) \tag{1}$$

These embeddings are stacked along the temporal dimension to form $z_r^{ij} \in \mathbb{R}^{T_r \times d_r}$ per layer, yielding the agent's full set of communication embeddings $z_\tau^{ij} = \{z_r^{ij}\}_{r=1}^{R}$.

The learned communication embeddings in Eq. 1 serve as task representations, based on our hypothesis that multi-agent communication captures underlying coordination patterns and task-relevant information. By leveraging decentralized graph-based message aggregation, these embeddings flexibly represent team-level interactions that adapt to changing structures without relying on predefined task features or skill partitions. We use them to ground trajectory generation across team compositions by *conditioning* the generative diffusion process, as described in Section 4.2.1.

4.2 Communication Conditioned Diffusion

We extend diffusion-based RL planning [1] by conditioning trajectory generation on both high-returns and coordinated team behaviors, by using graph-based communication to steer the generative process through classifier-free guidance.

4.2.1 Guided Sampling with Communication. For effective guided sampling across teams, the conditioning signal should represent the team composition, enabling the model to capture correlations between trajectories and team structures. Moreover, it must be flexible and expressive enough to support a meta-learning framework, ensuring that generalizable representations enable adaptation to new team compositions. With these motivations, we use the learned *communication embeddings* z_τ^{ij} to condition each agent's trajectory generation, as they provide a suitable basis of task-relevant information across tasks, as described in Section 4.1.2.

First, the conditioning constraint for each agent i of class j, incorporating both the communication embeddings z_τ^{ij} and the team's return throughout a trajectory R_τ, can be denoted as $y(\tau) := \left(z_\tau^{ij}, R_\tau\right)$. Assuming that, during training, the model has learned the conditional distributions of trajectories under these constraints, $p_\theta(x_0^{ij} | (z_\tau^{ij}, R_\tau))$, each agent can effectively sample from the conditional distribution by guiding the diffusion process. To achieve this, we modify the classifier-free guidance sampling [8] process of each agent. Starting with a Gaussian noise sample $x_K^{ij} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I})$ for each agent, this sample undergoes a series of "denoising" steps [20]. At each step k, the transformation moves x_k^{ij} to x_{k-1}^{ij} by removing the perturbed noise $\hat{\epsilon}_\theta$ denoted in Eq. 2. Here, the noise sample from the unconditional model is denoted as $\epsilon_\theta\left(x_k^{ij}, \emptyset, k\right)$, while the noise sample from the conditional model is denoted as $\epsilon_\theta\left(x_k^{ij}, \left(z_\tau^{ij}, R_\tau\right), k\right)$, ω is the guidance scale coefficient, and θ parametrizes both unconditional and conditional diffusion models. This guides the diffusion process of each agent towards state sequences that satisfy the constraint of achieving high-returns under their team composition.

$$\hat{\epsilon}_\theta := \epsilon_\theta\left(x_k^{ij}, \emptyset, k\right) + \omega\left(\epsilon_\theta\left(x_k^{ij}, \left(z_\tau^{ij}, R_\tau\right), k\right) - \epsilon_\theta\left(x_k^{ij}, \emptyset, k\right)\right) \tag{2}$$

4.2.2 Inverse Dynamics Model. The process in Eq. 2 continues until the final diffusion state $x_0^{ij} := [o_{1:h}^{ij} \| \hat{o}_{h+1:h+H}^{ij}]$, containing both the agent's observation history up to timestep h and its estimated future observations up to the diffusion horizon H. This final state constitutes each agent's *plan*, which is used by a controller to generate actions. Formally, decision-making is performed by inferring a policy through an inverse dynamics model [1] with parameters ϕ that estimates each agent's action $\hat{a}_t^{ij} := f_\phi\left(o_t^{ij}, o_{t+1}^{ij}\right)$ resulting in the transition from o_t^{ij} to o_{t+1}^{ij}. Our key motivation for using an inverse dynamics model, rather than directly diffusing actions, is our focus on multi-agent domains with discrete actions.

4.3 Offline Meta-MARL

We formulate our method as an offline meta-RL framework [2], where an *inner-loop* handles task inference and an *outer-loop* drives policy generalization. The *inner-loop* consists of graph-based communication layers that extract task-relevant representations by aggregating multi-agent messages according to the graph structure of each team composition. These embeddings capture team-specific context and serve as conditioning signals for downstream trajectory generation. The *outer-loop* consists of the shared diffusion architecture and inverse dynamics model, which are meta-trained across datasets containing multiple team compositions. Together, the outer-loop's capacity to generate high-return trajectories across tasks and the inner-loop's ability to produce adaptive, task-relevant conditioning signals enable generalization beyond the specific team compositions encountered during training.

Building on this structure, we define our training objective by adapting the standard diffusion loss [7] into an offline meta-MARL objective. This loss is optimized across data samples $d_{\mathfrak{m}}$ for all team compositions in the dataset $\mathcal{D}_{\mathfrak{m}}$, as shown in Eq. 3. At diffusion timestep k for composition $\mathfrak{m}$, $\bar{\mathbf{x}}_k^{\mathfrak{m}}$ is the noised joint state, $\bar{\mathbf{z}}_\tau^{\mathfrak{m}}$ the

joint communication embedding, and $R_\tau^{\mathfrak{m}}$ the return. $\beta := [\beta_z, \beta_r]$ gives the dropout probabilities for the communication embedding and return conditioning, respectively.

$$\mathcal{L}_\theta^D = \mathop{\mathbb{E}}_{d_{\mathfrak{m}} \sim \mathcal{D}_{\mathfrak{m}}} \left[\mathop{\mathbb{E}}_{\epsilon,k,\beta} \left[\left\| \epsilon - \epsilon_\theta\left(\bar{\mathbf{x}}_k^{\mathfrak{m}}, (1-\beta)(\bar{\mathbf{z}}_\tau^{\mathfrak{m}}, R_\tau^{\mathfrak{m}}) + \beta\emptyset, k\right) \right\|^2 \right] \right] \quad (3)$$

Note that $\bar{\mathbf{z}}_\tau^{\mathfrak{m}}$ is an online representation derived directly from raw observations, rather than a pre-computed or frozen feature. It serves as a conditioning signal at the decoder level, enabling end-to-end optimization in which gradients from the diffusion loss $\mathcal{L}_\theta^D$ propagate through $\bar{\mathbf{z}}_\tau^{\mathfrak{m}}$ back into the encoder. As a result, the communication protocol and the diffusion model co-adapt during training, allowing task inference to evolve jointly with trajectory generation and improving the agent's conditional planning capability.

Alongside the diffusion loss, each agent's action is inferred by an inverse dynamics model, trained jointly with the diffusion model. The loss is given in Eq. 4, where $(\bar{\mathbf{o}}_t^{\mathfrak{m}}, \bar{\mathbf{a}}_t^{\mathfrak{m}}, \bar{\mathbf{o}}_{t+1}^{\mathfrak{m}})$ denotes a transition tuple sampled from trajectory τ under team composition $\mathfrak{m}$.

$$\mathcal{L}_\phi^I = \mathop{\mathbb{E}}_{d_{\mathfrak{m}} \sim \mathcal{D}_{\mathfrak{m}}} \left[\mathbb{E}_{t\sim\tau} \left[\| \bar{\mathbf{a}}_t^{\mathfrak{m}} - f_\phi(\bar{\mathbf{o}}_t^{\mathfrak{m}}, \bar{\mathbf{o}}_{t+1}^{\mathfrak{m}}) \|^2 \right] \right] \quad (4)$$

We combine the training objectives in Eq. 3–4 into the total offline meta-RL objective $\mathcal{L}_{\theta,\phi}^T = \mathcal{L}_\theta^D + \mathcal{L}_\phi^I$, applied across data samples $d_{\mathfrak{m}}$ for all team compositions in the dataset. Detailed training and testing procedures are provided in Appendix B.

5 EVALUATION

5.1 Domain

We evaluate our method in the ubiquitous SMACv2 domain [3], which incorporates a high level of stochasticity, meaningful partial observability, and heterogeneity, presenting challenging multi-agent scenarios that motivate the development of robust, decentralized, and generalizable algorithms. Our main experiments are conducted on the `terran_5_vs_5` and `protoss_5_vs_5` scenarios, where each team consists of five agents drawn from three distinct classes. In `terran_5_vs_5`, the classes are Marine (ma), Marauder (mr), and Medivac (md); in `protoss_5_vs_5`, they are Stalker (st), Zealot (ze), and Colossus (co). Throughout the paper, we will denote team compositions by the number of agents per class, e.g., a team with two Marines, two Marauders, and one Medivac is written as `2ma_2mr_1md`. We also evaluate the `zerg_5_vs_5` scenario, with results reported in Appendix D. Additional details on the SMACv2 domain are provided in Appendix C.1.

5.2 Dataset

Each SMACv2 scenario includes twenty possible team compositions, reflecting the variability of agents and roles in this domain. To construct an expert dataset, we train a distinct expert policy for each team composition in the set of training tasks, to ensure that the collected trajectories capture coordination behaviors specific to each team structure. Expert policies are trained using HATRPO [15], a state-of-the-art method that leverages trust-region gradient updates and decomposed joint advantages to achieve more stable and reliable optimization.

For the `terran_5_vs_5` and `protoss_5_vs_5` scenarios, we select 6 out of the 20 possible team compositions as training tasks. Each selected composition consists of exactly two unit types, arranged in a fixed 2–3 or 3–2 split (e.g., `2ma_3md`, `3ma_2mr`). This setup captures a subset of inter-agent coordination strategies while leaving generalization to unseen team compositions with different unit types as a core challenge. Based on the dataset win rates, we collected 600 episodes per training task without selecting only the best demonstrations. Further details are provided in Appendix C.2.

5.3 Baseline Methods

We evaluate GCD against four categories of baselines. First, we compare with standard decentralized offline MARL methods, including **MA-CQL** [32], **MA-ICQ** [43], **MA-IQL** [14], and **OMAR** [27]. These methods represent competitive baselines for offline policy learning in multi-agent systems without explicit mechanisms for generalization. Second, we include offline MARL methods explicitly designed for generalization across tasks, such as **ODIS** [46] and **HiSSD** [21], to provide a relevant comparison for evaluating our framework's ability to adapt across team compositions. Third, we compare against Multi Agent Diffusion (**MADiff**) [48], which is closest to our architecture and allows for direct comparison between our graph-conditioned diffusion framework and prior generative modeling techniques utilized in multi-agent offline RL. Here, on top of the decentralized version (**MADiff-D**), a centralized variant (**MADiff-C**) is further tested to investigate whether full centralization can match the graph-based generalization achieved by GCD. Finally, we include two communicative offline MARL methods, **MASIA** [6] and **MHCI** [22], to assess whether explicit inter-agent communication can yield comparable generalization performance.

5.4 Experiments

To assess the effectiveness and generalization capabilities of GCD, our experiments address the following research questions:

- **RQ1:** Can Graph Conditioned Diffusion (GCD) leverage decentralized multi-agent communication to achieve stronger generalization across seen and unseen team compositions?
- **RQ2:** How does the choice of underlying graph-based communication protocol affect GCD's performance and adaptability?

To address **RQ1**, we train all methods on the same datasets, comprising a subset of team compositions used as training tasks, and perform a thorough evaluation of performance across all team compositions. Specifically, we run 100 online evaluation episodes per seed across all compositions, using three seeds, and report the average (± standard deviation) win-rate percentage.

To address **RQ2**, we conduct ablation experiments on the graph-based communication protocol used in GCD. We train three variants of our method: **GCD-Homo**, which employs a Graph Convolutional Network (GCN) [13] with homogeneous layers, **GCD-HetGCN**, which applies a heterogeneous GCN [29], and **GCD-HetGAT**, which extends the heterogeneous GCN with a type-specific attention mechanism [30]. The communication protocols for GCD-Homo and GCD-HetGCN are detailed in Appendix A.2, while GCD-HetGAT follows the formulation in Section 4.1.2. Moreover, to assess the flexibility of graph-based communication, we evaluate all three variants under settings where agent type information is removed from the observation space in the dataset.

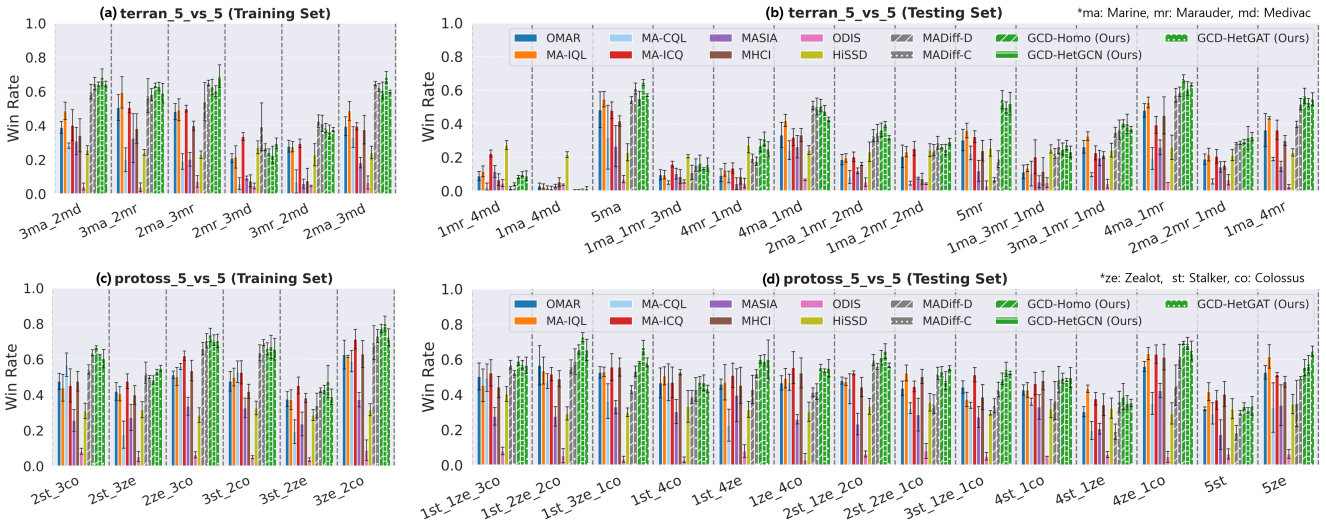

Figure 2: Results of evaluation experiments for the `terran_5_vs_5` scenario (a)-(b) and the `protoss_5_vs_5` scenario (c)-(d). We show the average win-rates (± std) across all methods, grouped by team composition labels. Our method, GCD, is shown in green, with distinct dashed patterns representing distinct graph-based communication protocols.

All experiments are conducted under decentralized execution, except for MADiff-C. Standard offline MARL and meta-MARL baselines operate in a decentralized manner by default. MADiff-D preserves decentralization through teammate and opponent modeling, while MADiff-C serves as a centralized variant with access to full agent information. Communicative offline MARL baselines maintain decentralized execution but allow information sharing through communication channels; following their original design, agents are granted full communicability with all other agents. Our method also adheres to decentralization, with agents communicating via graph-based messaging defined by sight range (i.e., limited to visually observable agents). Note that our work focuses on offline generalization, and therefore online baselines are not included in our experiments. Please refer to Appendix A.3 and Appendix A.4 for detailed experimental setups.

6 RESULTS AND DISCUSSION

In this section, we present the results of our experiments and discuss the performance of our proposed framework relative to the baseline methods. We begin with a task-level analysis and examine performance across individual team compositions within each domain. Then, we provide an algorithm-level analysis and quantify the overall performance improvement relative to each baseline across all team compositions. Finally, we analyze the results of the ablation experiments on graph-based communication protocols by comparing the task-level performance of different GCD variants. Full experimental results are provided in Appendix D.

6.1 Generalization Across Team Compositions

6.1.1 Task-level Analysis of Generalization. As shown in Figure 2, we present the generalization performance of all methods across training and unseen testing compositions in `terran_5_vs_5` and `protoss_5_vs_5` scenarios. In `terran_5_vs_5`, GCD outperforms all baselines in 4 out of the 6 team compositions in the training set as seen in Figure 2a, and in 10 out of the 14 unseen team compositions from the testing set as seen in Figure 2b. Similarly, in `protoss_5_vs_5`, GCD outperforms all baselines in 5 out of the 6 team compositions in the training set, as seen in Figure 2c, and in 10 out of the 14 unseen team compositions in the testing set, as seen in Figure 2d. We note that in `terran_5_vs_5`, some compositions have a high-number of Medivac (md) units, which serve support rather than attack roles. While our method may correctly capture this unit type's role in inter-agent interactions, this does not contribute to the win-rate itself. Notably, this accounts for 3 out of the 4 unseen test team compositions where our method underperforms baselines. Similarly, in `protoss_5_vs_5`, some team compositions contain a high number of special Colossus (co) units, which are very powerful and can solely determine the outcome of battles. This makes inter-agent cooperation less critical, a factor that may contribute to our method's underperformance.

Figure 2 indicates that GCD's ability to condition on task-relevant information from communication embeddings leads to improved coordination performance across team compositions in both training and testing tasks. Focusing on results in seen training compositions, we show evidence of intra-task adaptability, where our conditioning mechanism enables the policy to adapt its coordination behavior to the specific dynamics of each team composition. Beyond training compositions, GCD consistently achieves higher performance in unseen team compositions. We attribute this improvement to the model's ability to leverage communication embeddings as transferable representations of team structure. This enables the conditional generative process to adapt coordination behaviors to new combinations of agents. This capability underscores the capacity and flexibility of graph-based multi-agent communication to produce task-relevant embeddings that generalize beyond the specific team compositions encountered during training.

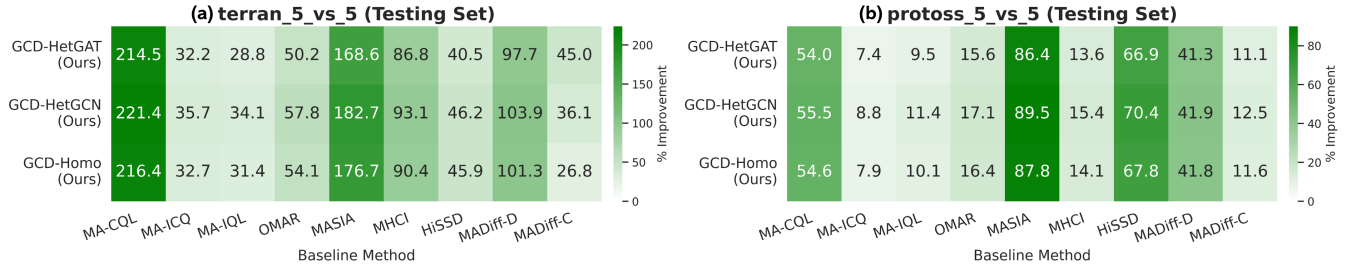

Figure 3: GCD's improvement over baselines across unseen test compositions for `terran_5_vs_5` (a) and `protoss_5_vs_5` (b).

Finally, across team compositions where GCD achieves superior performance, all GCD models outperform the baselines. However, we do not observe substantial performance differences among different graph-based communication protocols. This indicates that while GCD is robust to the choice of communication mechanism, the benefits of heterogeneous graph modeling are under-realized in the current setting. This is further examined in Section 6.2.

Key Takeaway: Through decentralized graph-based communication, GCD demonstrates strong generalization across team compositions, exhibiting improved adaptability on seen training compositions and greater generalization to unseen testing compositions.

6.1.2 Algorithm-level Analysis of Generalization. We compute the average performance improvement of each GCD variant over every baseline across all team compositions in the testing set. These results are compactly visualized in Figure 3. For all methods, every variant of GCD demonstrates a positive average percent performance improvement. This ranges from 26.8% to 221.4% improvement in the `terran_5_vs_5` scenario and 7.4% to 89.5% improvement in the `protoss_5_vs_5` scenario.

Offline MARL. The results in Figure 3 show that the smallest performance gap to our method occurs in standard offline-RL algorithms such as MA-ICQ, MA-IQL, and OMAR. Although these algorithms lack an explicit adaptation mechanism like those used in ODIS, HiSSD, and our framework, they mitigate out-of-distribution (OOD) data shift problems through techniques that regularize the learned value functions and stabilize the learning process. Their relative competitiveness underscores the importance of addressing OOD shift in offline-RL, whether the challenge stems from limited data in the single-task setting for which these algorithms were designed, or from adapting to new tasks in the multi-task setting explored in this work. However, while these mechanisms enable moderate generalization due to the similarity in dynamics across teams, our results show that this implicit robustness is insufficient for adapting to specific team compositions.

Offline Meta-MARL. Our results also show that GCD outperforms methods designed with explicit adaptation tools such as ODIS and HiSSD. While these methods share conceptual similarities with our approach in that they also condition policies on task representations, they underperform due to the lack of flexibility of their representation mechanisms. ODIS relies on predefined skill partitions to separate task behaviors. This limits its effectiveness in our problem setting, which is characterized by high variability across tasks. Meanwhile, HiSSD employs a hierarchical architecture that decomposes behavior into high-level and low-level skills, each optimized through additional surrogate objectives. Although this design can be effective in well-structured domains, the added complexity makes learning more difficult in settings with high task variability, limiting its generalizability in our experiments. In contrast, our method improves coordination performance by conditioning on task-relevant communication embeddings learned directly from agent interactions, enabling flexible adaptation to varying team compositions.

Offline Multi-Agent Diffusion. Our performance improvements relative to MADiff, which also employs generative modeling through diffusion, highlight generalization advantages of our approach over the most closely related method. In contrast to MADiff, which relies on teammate and opponent modeling to maintain decentralization, our framework enables structured information exchange via graph-based communication while preserving decentralized execution. Furthermore, whereas MADiff incorporates shared information as additional input features, our method leverages it to learn conditional generative policies, providing a more principled mechanism for generating adaptive behavior across tasks. Notably, communication alone does not account for these improvements; GCD consistently outperforms MADiff-C despite operating under decentralized execution with access only to local information.

Offline Communicative MARL. Finally, our comparison with MASIA and MHCI further supports that GCD's performance gains arise not only from communication itself but also from its graph-based generalization. Moreover, we observe that communication does not always guarantee success in offline training, as poorly learned communication protocols may introduce noise that hinders effective policy learning.

Key Takeaway: The conditional generative modeling abilities of diffusion models, combined with the flexibility of graph-based communication, enable improved generalization over standard offline MARL, meta-MARL, communicative MARL, and diffusion-based generative modeling baselines.

6.2 Ablation on Communication Protocols

As previously discussed, we observe no significant performance difference across GCD models with different graph-based communication protocols. Although GCD-HetGCN and GCD-HetGAT are designed to exploit the heterogeneity of our problem, they do not

Table 1: Ablation of GCD communication protocols. Average win rates are reported on training and test sets, highlighting improvements of heterogeneous modules over the homogeneous baseline (Homo), as well as HetGAT's gains over HetGCN.

Method	terran 5 vs 5				protoss 5 vs 5			
	Training Set	Δ vs Homo	Test Set	Δ vs Homo	Training Set	Δ vs Homo	Test Set	Δ vs Homo
GCD-Homo	0.318	–	0.179	–	0.227	–	0.150	–
GCD-HetGCN	0.359	+12.89%	0.243	+35.75%	0.311	+37.00%	0.223	+48.67%
GCD-HetGAT	**0.424**	+18.11%	**0.253**	+41.34%	**0.337**	+48.46%	**0.260**	+73.33%
HetGAT vs HetGCN	+18.11%	–	+4.11%	–	+8.36%	–	+16.59%	–

outperform GCD-Homo in the standard setting. We hypothesize that access to agent-type information in the observation space allows GCD-Homo to construct task-relevant representations that support conditional policy generation.

To validate this hypothesis, we conduct ablation experiments by masking type information and comparing GCD-Homo, GCD-HetGCN, and GCD-HetGAT. As shown in Table 1, both GCD-HetGCN and GCD-HetGAT consistently outperform GCD-Homo, with respective average gains of 35.75% and 41.34% on unseen tasks in `terran_5_vs_5` and 48.67% and 73.33% in `protoss_5_vs_5`. This suggests that heterogeneous graph structures are more effective in capturing emergent inter-agent interactions when explicit type information is unavailable. Moreover, GCD-HetGAT surpasses GCD-HetGCN by 4.11% and 16.59% in the unseen tasks from `terran_5_vs_5` and `protoss_5_vs_5`, respectively, suggesting that its attentional communication mechanism further enhances generalization to novel team compositions. Overall, this underscores the flexibility of GCD to operate effectively across different communication protocols, while also demonstrating its ability to exploit structural information when the problem setting requires it. Compared to homogeneous models, modeling communication heterogeneously increases the network's expressivity and enables straightforward modeling of complex team distributions.

Key Takeaway: While GCD demonstrates robustness across multiple graph-based communication protocols, heterogeneous methods provide superior performance in settings that demand stronger reasoning over agent heterogeneity.

6.3 Computational Complexity

Our approach's main computational cost per decision step stems from the GNN communication and the K-step denoising process. For N agents, each denoising step has complexity $O(N^2)$ due to the fully-connected agent message passing in the GNN architecture, defined by the number of edges. Additionally, our diffusion-based method requires K denoising steps, leading to a complexity of $O(KN^2)$. This is computationally similar to the baseline MADiff method, which uses a standard attention mechanism.

In contrast to MADiff, our approach supports decentralized information sharing among agents via graph-based communication, where $O(KN^2)$ represents the worst-case complexity, where each agent communicates with every other agent. If agents communicate with only $d \ll N$ agents, where d is the average number of edges of each agent, then our total inference complexity can decrease to $O(dKN)$. Moreover, we take advantage of DDIM sampling in our approach, which can require exponentially fewer diffusion steps than standard diffusion sampling methods [36]. As such, our approach advances the scalability and robustness of diffusion models for multi-agent decision-making, especially for scenarios with real-world constraints such as communication bandwidth and communication network failures.

7 LIMITATIONS AND FUTURE WORK

One limitation of our work is the reduced accuracy of the inverse dynamics model, which may be the bottleneck for learning higher-performing policies. Future work could reduce the reliance on the inverse dynamics model by directly generating actions using a discrete diffusion model. Another promising direction for improving policies involves applying RL for fine-tuning the diffusion model. While RL fine-tuning has been explored in single-agent contexts, its effectiveness is to be determined for MARL. The results presented in this work also yield relevant future work. SMACv2 has well-defined dynamics and sufficiently covers the problem of generalizing across team compositions through its heterogeneous teaming scenarios. However, future work can explore additional settings and domains where more advanced graph structures lead to better generalization performance, guided by insights from our ablation on graph-based communication protocols. Additionally, incorporating graph-based multi-agent communication enables future exploration of generalization to continuous and variable team size domains. Such extensions could include larger 10 and 20-agent settings available in SMACv2 or Google Research Football [18].

8 CONCLUSION

We presented Graph Conditioned Diffusion (GCD), a novel method that combines graph-based communication with diffusion models to address the challenges of generalization in offline MARL. By modeling agent interactions through graphs and leveraging classifier-free guidance in diffusion models, GCD effectively learns decentralized policies that generalize across team compositions. Our experiments on SMACv2 micromanagement tasks showed that GCD outperforms state-of-the-art offline MARL methods in limited data settings by generalizing across seen and unseen team compositions. These results highlight the potential of integrating graph-based communication within diffusion models to enhance coordination and generalization across multi-agent systems.

ACKNOWLEDGMENTS

This work was supported by the Naval Research Laboratory under Grants N00173-21-1-G009 and N00173-25-1-0050 and by the Laboratory Directed Research and Development program at Sandia National Laboratories.

REFERENCES

[1] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua Tenenbaum, Tommi Jaakkola, and Pulkit Agrawal. 2022. Is conditional generative modeling all you need for decision-making? *arXiv preprint arXiv:2211.15657* (2022).

[2] Jacob Beck, Risto Vuorio, Evan Zheran Liu, Zheng Xiong, Luisa Zintgraf, Chelsea Finn, and Shimon Whiteson. 2023. A survey of meta-reinforcement learning. *arXiv e-prints* (2023), arXiv–2301.

[3] Benjamin Ellis, Jonathan Cook, Skander Moalla, Mikayel Samvelyan, Mingfei Sun, Anuj Mahajan, Jakob Foerster, and Shimon Whiteson. 2024. Smacv2: An improved benchmark for cooperative multi-agent reinforcement learning. *Advances in Neural Information Processing Systems* 36 (2024).

[4] Claude Formanek, Asad Jeewa, Jonathan Shock, and Arnu Pretorius. 2023. Off-the-Grid MARL: Datasets with Baselines for Offline Multi-Agent Reinforcement Learning. arXiv:2302.00521 [cs.LG] https://arxiv.org/abs/2302.00521

[5] Scott Fujimoto, David Meger, and Doina Precup. 2019. Off-policy deep reinforcement learning without exploration. In *International conference on machine learning*. PMLR, 2052–2062.

[6] Cong Guan, Feng Chen, Lei Yuan, Zongzhang Zhang, and Yang Yu. 2024. Efficient Communication via Self-Supervised Information Aggregation for Online and Offline Multiagent Reinforcement Learning. *IEEE Transactions on Neural Networks and Learning Systems* (2024).

[7] Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems* 33 (2020), 6840–6851.

[8] Jonathan Ho and Tim Salimans. 2022. Classifier-free diffusion guidance. *arXiv preprint arXiv:2207.12598* (2022).

[9] Lu Hong and Scott E Page. 2001. Problem solving by heterogeneous agents. *Journal of economic theory* 97, 1 (2001), 123–163.

[10] Christopher D Hsu, Heejin Jeong, George J Pappas, and Pratik Chaudhari. 2021. Scalable reinforcement learning policies for multi-agent control. In *2021 IEEE/RSJ international conference on intelligent robots and systems (IROS)*. IEEE, 4785–4791.

[11] Siyi Hu, Fengda Zhu, Xiaojun Chang, and Xiaodan Liang. 2021. Updet: Universal multi-agent reinforcement learning via policy decoupling with transformers. *arXiv preprint arXiv:2101.08001* (2021).

[12] Michael Janner, Yilun Du, Joshua B Tenenbaum, and Sergey Levine. 2022. Planning with diffusion for flexible behavior synthesis. *arXiv preprint arXiv:2205.09991* (2022).

[13] Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907* (2016).

[14] Ilya Kostrikov, Rob Fergus, Jonathan Tompson, and Ofir Nachum. 2021. Offline reinforcement learning with fisher divergence critic regularization. In *International Conference on Machine Learning*. PMLR, 5774–5783.

[15] Jakub Grudzien Kuba, Ruiqing Chen, Muning Wen, Ying Wen, Fanglei Sun, Jun Wang, and Yaodong Yang. 2022. Trust Region Policy Optimisation in Multi-Agent Reinforcement Learning. In *International Conference on Learning Representations*.

[16] Aviral Kumar, Justin Fu, George Tucker, and Sergey Levine. 2019. Stabilizing Off-Policy Q-Learning via Bootstrapping Error Reduction. arXiv:1906.00949 [cs.LG] https://arxiv.org/abs/1906.00949

[17] Aviral Kumar, Aurick Zhou, George Tucker, and Sergey Levine. 2020. Conservative q-learning for offline reinforcement learning. *Advances in Neural Information Processing Systems* 33 (2020), 1179–1191.

[18] Karol Kurach, Anton Raichuk, Piotr Stańczyk, Michał Zając, Olivier Bachem, Lasse Espeholt, Carlos Riquelme, Damien Vincent, Marcin Michalski, Olivier Bousquet, et al. 2020. Google research football: A novel reinforcement learning environment. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 4501–4510.

[19] Sergey Levine, Aviral Kumar, George Tucker, and Justin Fu. 2020. Offline reinforcement learning: Tutorial, review, and perspectives on open problems. *arXiv preprint arXiv:2005.01643* (2020).

[20] Nan Liu, Shuang Li, Yilun Du, Antonio Torralba, and Joshua B Tenenbaum. 2022. Compositional visual generation with composable diffusion models. In *European Conference on Computer Vision*. Springer, 423–439.

[21] Sicong Liu, Yang Shu, Chenjuan Guo, and Bin Yang. 2025. Learning generalizable skills from offline multi-task data for multi-agent cooperation. *arXiv preprint arXiv:2503.21200* (2025).

[22] Yihuan Mao, Rui Hu, Lulu Zheng, Jianhao Wang, and Chongjie Zhang. [n.d.]. Offline Communication Learning with Multi-source Datasets. ([n. d.]).

[23] Linghui Meng, Jingqing Ruan, Xuantang Xiong, Xiyun Li, Xi Zhang, Dengpeng Xing, and Bo Xu. 2023. M3: Modularization for Multi-task and Multi-agent Offline Pre-training. In *Proceedings of the 2023 International Conference on Autonomous Agents and Multiagent Systems*. 1624–1633.

[24] Linghui Meng, Muning Wen, Yaodong Yang, Chenyang Le, Xiyun Li, Weinan Zhang, Ying Wen, Haifeng Zhang, Jun Wang, and Bo Xu. 2021. Offline pre-trained multi-agent decision transformer: One big sequence model tackles all smac tasks. *arXiv preprint arXiv:2112.02845* (2021).

[25] Yaru Niu, Rohan R Paleja, and Matthew C Gombolay. 2021. Multi-Agent Graph-Attention Communication and Teaming.. In *AAMAS*. 964–973.

[26] Frans A Oliehoek and Christopher Amato. 2016. *A concise introduction to decentralized POMDPs*. Springer.

[27] Ling Pan, Longbo Huang, Tengyu Ma, and Huazhe Xu. 2022. Plan better amid conservatism: Offline multi-agent reinforcement learning with actor rectification. In *International conference on machine learning*. PMLR, 17221–17237.

[28] Lukas Schäfer, Filippos Christianos, Amos Storkey, and Stefano V Albrecht. 2022. Learning task embeddings for teamwork adaptation in multi-agent reinforcement learning. *arXiv preprint arXiv:2207.02249* (2022).

[29] Michael Schlichtkrull, Thomas N Kipf, Peter Bloem, Rianne Van Den Berg, Ivan Titov, and Max Welling. 2018. Modeling relational data with graph convolutional networks. In *European semantic web conference*. Springer, 593–607.

[30] Esmaeil Seraj, Rohan Paleja, Luis Pimentel, Kin Man Lee, Zheyuan Wang, Daniel Martin, Matthew Sklar, John Zhang, Zahi Kakish, and Matthew Gombolay. 2024. Heterogeneous policy networks for composite robot team communication and coordination. *IEEE Transactions on Robotics* (2024).

[31] Esmaeil Seraj, Zheyuan Wang, Rohan Paleja, Daniel Martin, Matthew Sklar, Anirudh Patel, and Matthew Gombolay. 2022. Learning efficient diverse communication for cooperative heterogeneous teaming. In *Proceedings of the 21st international conference on autonomous agents and multiagent systems*. 1173–1182.

[32] Jianzhun Shao, Yun Qu, Chen Chen, Hongchang Zhang, and Xiangyang Ji. 2023. Counterfactual conservative q learning for offline multi-agent reinforcement learning. *Advances in Neural Information Processing Systems* 36 (2023), 77290–77312.

[33] Tianyu Shi, Dong Chen, Kaian Chen, and Zhaojian Li. 2021. Offline reinforcement learning for autonomous driving with safety and exploration enhancement. *arXiv preprint arXiv:2110.07067* (2021).

[34] Amanpreet Singh, Tushar Jain, and Sainbayar Sukhbaatar. 2018. Learning when to communicate at scale in multiagent cooperative and competitive tasks. *arXiv preprint arXiv:1812.09755* (2018).

[35] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. 2015. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*. PMLR, 2256–2265.

[36] Jiaming Song, Chenlin Meng, and Stefano Ermon. 2020. Denoising diffusion implicit models. *arXiv preprint arXiv:2010.02502* (2020).

[37] Yang Song and Stefano Ermon. 2019. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems* 32 (2019).

[38] Sainbayar Sukhbaatar, Rob Fergus, et al. 2016. Learning multiagent communication with backpropagation. *Advances in neural information processing systems* 29 (2016).

[39] Wei-Cheng Tseng, Tsun-Hsuan Johnson Wang, Yen-Chen Lin, and Phillip Isola. 2022. Offline Multi-Agent Reinforcement Learning with Knowledge Distillation. In *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (Eds.), Vol. 35. Curran Associates, Inc., 226–237. https://proceedings.neurips.cc/paper_files/paper/2022/file/01d78b294d80491fecddea897cf03642-Paper-Conference.pdf

[40] Wei-Cheng Tseng, Wei Wei, Da-Cheng Juan, and Min Sun. 2021. Meta-cpr: Generalize to unseen large number of agents with communication pattern recognition module. *arXiv preprint arXiv:2112.07222* (2021).

[41] Weixun Wang, Tianpei Yang, Yong Liu, Jianye Hao, Xiaotian Hao, Yujing Hu, Yingfeng Chen, Changjie Fan, and Yang Gao. 2020. From few to more: Large-scale dynamic multiagent curriculum learning. In *Proceedings of the AAAI conference on artificial intelligence*, Vol. 34. 7293–7300.

[42] Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. 2022. Diffusion policies as an expressive policy class for offline reinforcement learning. *arXiv preprint arXiv:2208.06193* (2022).

[43] Yiqin Yang, Xiaoteng Ma, Chenghao Li, Zewu Zheng, Qiyuan Zhang, Gao Huang, Jun Yang, and Qianchuan Zhao. 2021. Believe what you see: Implicit constraint approach for offline multi-agent reinforcement learning. *Advances in Neural Information Processing Systems* 34 (2021), 10299–10312.

[44] Sean Ye, Manisha Natarajan, Zixuan Wu, and Matthew C. Gombolay. 2023. Diffusion Models for Multi-target Adversarial Tracking. In *2023 International Symposium on Multi-Robot and Multi-Agent Systems (MRS)*. 142–148. https://doi.org/10.1109/MRS60187.2023.10416775

[45] Lei Yuan, Ziqian Zhang, Lihe Li, Cong Guan, and Yang Yu. 2023. A survey of progress on cooperative multi-agent reinforcement learning in open environment. *arXiv preprint arXiv:2312.01058* (2023).

[46] Fuxiang Zhang, Chengxing Jia, Yi-Chen Li, Lei Yuan, Yang Yu, and Zongzhang Zhang. 2022. Discovering generalizable multi-agent coordination skills from multi-task offline data. In *The Eleventh International Conference on Learning Representations*.

[47] Tianze Zhou, Fubiao Zhang, Kun Shao, Kai Li, Wenhan Huang, Jun Luo, Weixun Wang, Yaodong Yang, Hangyu Mao, Bin Wang, et al. 2021. Cooperative multi-agent transfer learning with level-adaptive credit assignment. *arXiv preprint arXiv:2106.00517* (2021).

[48] Zhengbang Zhu, Minghuan Liu, Liyuan Mao, Bingyi Kang, Minkai Xu, Yong Yu, Stefano Ermon, and Weinan Zhang. 2023. Madiff: Offline multi-agent learning with diffusion models. *arXiv preprint arXiv:2305.17330* (2023).