Transformer-Based Scalable Multi-Agent Reinforcement Learning for Networked Systems with Long-Range Interactions

Vidur Sinha[1], Muhammed Ustaomeroglu[1], Guannan Qu[1]

Abstract— **Multi-agent reinforcement learning (MARL) has shown promise for large-scale network control, yet existing methods face two major limitations. First, they typically rely on assumptions leading to decay properties of local agent interactions, limiting their ability to capture long-range dependencies such as cascading power failures or epidemic outbreaks. Second, most approaches lack generalizability across network topologies, requiring retraining when applied to new graphs. We introduce STACCA (Shared Transformer Actor–Critic with Counterfactual Advantage), a unified transformer-based MARL framework that addresses both challenges. STACCA employs a centralized Graph Transformer Critic to model long-range dependencies and provide system-level feedback, while its shared Graph Transformer Actor learns a generalizable policy capable of adapting across diverse network structures. Further, to improve credit assignment during training, STACCA integrates a novel counterfactual advantage estimator that is compatible with state-value critic estimates. We evaluate STACCA on epidemic containment and rumor-spreading network control tasks, demonstrating improved performance, network generalization, and scalability. These results highlight the potential of transformer-based MARL architectures to achieve scalable and generalizable control in large-scale networked systems.**

I. Introduction

Multi-agent networked systems are ubiquitous in modern infrastructure, spanning power grids, transportation networks, social platforms, and beyond [4], [2]. These systems involve large numbers of interacting agents whose states and actions are coupled by the underlying network topology and dynamics. Although most interactions are local (e.g. power flowing between connected nodes in an electrical grid or diseases spreading through contact), they can sometimes trigger long-range effects, such as cascading power failures or rapid epidemic outbreaks. Designing local decision policies that capture complex long-range dependencies among agents is essential for effective control in large networked systems.

Multi-agent reinforcement learning (MARL) has emerged as a powerful paradigm for such networked control problems. There are many general MARL algorithms (e.g. MADDPG [6], MAPPO [14]); however, these are known to suffer from scalability issues, as the factorized state space in networked systems grows exponentially. More recently, Qu et al. proved decay properties under the local interaction structure of networked systems and exploited them to develop the Scalable Actor Critic framework [7] and applied it to decentralized voltage control in power grids [13]. Despite the progress, two key challenges remain in networked multi-agent systems:

(1) Long-Range Interactions. Existing MARL frameworks for networked systems rely on assumptions that show long-range network interactions have negligible impact on the system [7]. However, in many cases, these assumptions do not hold, and there can be long-range interactions (e.g. a few infected travelers can spark outbreaks in geographically distant regions). Thus, capturing long-range interactions becomes crucial for developing successful policies.

(2) Network Generalizability. Current solutions are typically trained and evaluated on fixed networks, with limited ability to generalize to unseen topologies. Achieving network generalizability under decentralized execution has tremendous implications for scalability, but it is especially difficult in real-world networks where local neighborhoods are heterogeneous. Moreover, the large number of agents exacerbates the infamous credit-assignment problem in MARL, making learning such policies incredibly challenging.

These challenges motivate the following question: *Can we create a MARL solution for networked systems that handles long-range interactions and is network generalizable?*

Contribution. To address this question, we introduce STACCA (Shared Transformer Actor–Critic with Counterfactual Advantage), a unified transformer-based MARL framework. Transformers are known for their ability to model long-range dependencies, and their core self-attention mechanism has been adapted to many domains, including graphs [10][11]. STACCA employs a centralized Graph Transformer Critic that learns global, long-range dependencies among agents, providing system-level context to aid policy learning and enhance local decisions. Complementing this, a shared Graph Transformer Actor learns a policy that effectively *stacca* ("detaches" in Italian) from the specific network topology seen during training, enabling zero-shot transfer to unseen network topologies. However, to both encourage the critic to learn long-range dependencies and expose the actor to diverse local network structures, we must train on relatively large networks which amplifies the challenge of *multi-agent credit assignment*. To address this, STACCA integrates a novel counterfactual advantage that leverages the global critic's state-value estimates to isolate each agent's contribution. Overall, our proposed method mitigates key drawbacks of prior MARL approaches in networked systems by jointly addressing long-range interaction modeling, network generalization, and credit-assignment.

Our experiments on epidemic containment and rumor-spreading network control tasks demonstrate that STACCA substantially outperforms the baseline MAPPO algorithm and showcase its scalability and generalizability benefits.

[1]Vidur Sinha, Muhammed Ustaomeroglu, and Guannan Qu are with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, USA. Correspondence to: Vidur Sinha <vidursin@andrew.cmu.edu>.

These results highlight the potential of transformer-based MARL architectures to achieve scalable, transferable, and effective control in complex, large-scale networked systems.

II. PROBLEM FORMULATION AND BACKGROUND

In this section, we first formulate the problem as a Networked Decentralized Partially Observable Markov Decision Process (Dec-POMDP), and use it to introduce two network control examples: epidemic containment and rumor spreading. We then review key ideas regarding Multi-Agent Proximal Policy Optimization (MAPPO), the Transformer architecture, and Graph Attention Networks (GATs).

A. Problem Formulation

We adopt a Dec-POMDP framework for networked systems (based on [7]). Consider a graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where nodes $\mathcal{N} = \{1, \ldots, N\}$ are agents and $\mathcal{E}$ are the edges. The global state at time t is the collection of individual agent states, $s_t = (s_{1,t}, \ldots, s_{N,t}) \in \mathcal{S} = \mathcal{S}_1 \times \cdots \times \mathcal{S}_N$, with $s_{i,t} \in \mathcal{S}_i$. Similarly, the global action is $a_t = (a_{1,t}, \ldots, a_{N,t}) \in \mathcal{A} = \mathcal{A}_1 \times \cdots \times \mathcal{A}_N$, and the global observation is $o_t = (o_{1,t}, \ldots, o_{N,t})$. Each agent's local observation is given by the states of the agents in its closed (includes i) k-hop neighborhood $\mathcal{N}_i^k$ and the edges of the corresponding subgraph: $o_{i,t} = (s_{\mathcal{N}_i^k, t}, \mathcal{E}_{\mathcal{N}_i^k})$.

A key property of our model is that state transitions factorize across the closed 1-hop neighborhood of each agent.

$$P(s_{t+1} | s_t, a_t) = \prod_{i=1}^{N} P_i(s_{i,t+1} | s_{\mathcal{N}_i^1, t}, a_{\mathcal{N}_i^1, t}) \qquad (1)$$

Further, we denote $r(s, a)$ as a team reward and γ is the discount factor. In MARL, typically each agent acts according to a local/decentralized policy $\pi_i(a_i | o_i)$, using only its local observation o_i under a memoryless policy π_i. The objective is to maximize the expected team return

$$J(\boldsymbol{\pi}) = \mathbb{E}_{\boldsymbol{\pi}}\left[\sum_{t=0}^{\infty} \gamma^t r(s_t, a_t) \right], \qquad (2)$$

where $\boldsymbol{\pi} = (\pi_1, \ldots, \pi_N)$. Correspondingly, the global state-value function under policy $\boldsymbol{\pi}$ is defined as

$$V_{\boldsymbol{\pi}}(s_t) = \mathbb{E}_{\boldsymbol{\pi}}\left[\sum_{k=0}^{\infty} \gamma^k r(s_{t+k}, a_{t+k}) \,\Big|\, s_t \right], \qquad (3)$$

and the advantage of taking joint action a_t in state s_t is

$$A_{\boldsymbol{\pi}}(s_t, a_t) = Q_{\boldsymbol{\pi}}(s_t, a_t) - V_{\boldsymbol{\pi}}(s_t), \qquad (4)$$

with $Q_{\boldsymbol{\pi}}(s_t, a_t) = \mathbb{E}_{\boldsymbol{\pi}}[\sum_{k=0}^{\infty} \gamma^k r(s_{t+k}, a_{t+k}) | s_t, a_t]$.

This general formulation applies to diverse networked systems, including power grid management, social network dynamics, and epidemic control. In this work, we focus on two representative and contrasting control tasks to demonstrate its utility.

Example 1 (Epidemic Containment). In this task, the goal is to minimize the spread of a disease by managing some control level (e.g. quarantine/vaccination intensity). The state of agent i is $s_{i,t} = (h_{i,t}, c_{i,t})$, where $h_{i,t} \in \{0(\text{susceptible}), 1(\text{infected})\}$ is its health status and $c_{i,t} \in [0, 1]$ is its control value. The action $a_{i,t} \in \{-\Delta c, 0, \Delta c\}$ is to decrease, maintain, or increase the control value. The transition of the state $s_{i,t} \to s_{i,t+1}$ happens in two parts. Firstly, the health status $h_{i,t}$ transitions probabilistically based on the *current* state. Let $I_{i,t} = |\{j \in \mathcal{N}_i^1 \setminus \{i\} : h_{j,t} = 1\}|$ denote the number of infected neighbors. The probability of an agent being susceptible at the next timestep is given by:

$$P(h_{i,t+1} = 0 \mid s_t) = \begin{cases} \delta & \text{if } h_{i,t} = 1, \\ (1 - \beta(c_{i,t}))^{I_{i,t}} & \text{if } h_{i,t} = 0, \end{cases} \quad (5)$$

where δ is the fixed probability of recovery. The effective transmission rate $\beta(c_{i,t}) = (1 - \eta c_{i,t})\beta_0$ is a function of a base transmission rate β_0, the agent's current control state $c_{i,t}$, and the control effectiveness η (e.g. for $\eta = 0.9$, full-control provides at most 90% reduction in infection probability for 1 infected neighbor). Secondly, the control level updates deterministically based on the action: $c_{i,t+1} = \text{clip}(c_{i,t} + a_{i,t}, 0, 1)$. The reward function is structured to penalize the global infection rate and control costs.

Example 2 (Rumor Spreading). In this task, the goal is to maximize the spread of information by managing advertising efforts. The state of agent i is $s_{i,t} = (h_{i,t}, c_{i,t})$, where $h_{i,t} \in \{0(\text{unaware}), 1(\text{aware})\}$ is its awareness status and $c_{i,t} \in [0, 1]$ is its advertising level, which we call the "boosting-factor." The action $a_{i,t} \in \{-\Delta c, 0, \Delta c\}$ is to decrease, maintain, or increase the boosting-factor. The awareness status transitions probabilistically, where the dynamics follow a "viral marketing" scenario with market saturation. Let $I_{i,t} = |\{j \in \mathcal{N}_i^1 \setminus \{i\} : h_{j,t} = 1\}|$ be the number of aware neighbors. The probability of an agent becoming unaware is then:

$$P(h_{i,t+1} = 0 \mid s_t) = \begin{cases} 0 & \text{if } h_{i,t} = 1 \\ (1 - \beta(s_t, c_{i,t}))^{I_{i,t}} & \text{if } h_{i,t} = 0 \end{cases} \quad (6)$$

The transmission probability $\beta(s_t, c_{i,t}) = c_{i,t}(1 - \bar{h}_t)^\kappa \beta_0$ now depends on the base spreading rate β_0, the boosting-factor state $c_{i,t}$, the fraction of aware nodes $\bar{h}_t = \frac{1}{N} \sum_{j=1}^{N} h_{j,t}$, and a saturation exponent $\kappa > 0$. Note that the saturation component in this example explicitly incorporates long-range interactions into the transition dynamics, which violates eq. (1), but provides a contrasting variation to the purely local dynamics of the epidemic example. Finally, the boosting-factor updates deterministically: $c_{i,t+1} = \text{clip}(c_{i,t} + a_{i,t}, 0, 1)$. Here, a reward function is designed to encourage the aware state while penalizing the cost of the global boosting-factor usage.

These examples highlight two core challenges we aim to address. First, the problem of *long-range interactions*, where local agent states and actions have complex, non-local consequences. Second, the need for *network-generalizability*, as an ideal control policy should be transferable and effective

both across local observations within the Dec-POMDP and across entire network topologies without requiring retraining.

Our proposed STACCA framework is designed to address these challenges by drawing from several methods in the literature. We now review the relevant background on Multi-Agent Proximal Policy Optimization (MAPPO), the Transformer architecture, and Graph Attention Networks (GATs).

B. Background: Multi-Agent Proximal Policy Optimization

Multi-Agent Proximal Policy Optimization (MAPPO)[14] is an on-policy actor–critic algorithm that adopts the Centralized Training with Decentralized Execution (CTDE) paradigm [5]. Under CTDE, a centralized critic has access to global information (e.g., the joint state of the entire graph $\mathcal{G}$) during training, providing a more stable and informative learning signal. During execution, however, each agent i employs a decentralized actor policy $\pi_\theta(a_i|o_i)$ to select actions using only its local observation o_i. Here, we have used parameter sharing – all agents use the same policy neural network parameterized by θ.

At each training round, suppose the current actor parameter is θ_{old}. We collect a batch of trajectories $\mathcal{D}$ by executing this policy in the environment, where at each timestep t in trajectory τ, agent i samples an action $a_{i,t}^\tau \sim \pi_{\theta_{\text{old}}}(\cdot|o_{i,t}^\tau)$ based on its local observation. The next actor parameter is obtained by optimizing the clipped surrogate objective over all agents, trajectories, and timesteps:

$$L_\pi^{CLIP}(\theta) = \hat{\mathbb{E}}_{i,\tau,t}\Big[\min\big(\rho_{i,t}^\tau(\theta)\,\hat{A}_t^\tau, \\ \text{clip}\big(\rho_{i,t}^\tau(\theta), 1-\epsilon, 1+\epsilon\big)\,\hat{A}_t^\tau\big)\Big], \quad (7)$$

where $\hat{\mathbb{E}}_{i,\tau,t}[\dots]$ denotes the empirical average over the samples $(s_{i,t}^\tau, a_{i,t}^\tau, o_{i,t}^\tau)$, $\rho_{i,t}^\tau(\theta)$ is the probability ratio $\frac{\pi_\theta(a_{i,t}^\tau|o_{i,t}^\tau)}{\pi_{\theta_{\text{old}}}(a_{i,t}^\tau|o_{i,t}^\tau)}$, ϵ is the clipping hyperparameter, and $\hat{A}_t^\tau$ is an estimate of the advantage function $A_{\pi_{\text{old}}}(s_t^\tau, a_t^\tau)$ (see eq. (4)) which is usually shared among agents at each timestep. This estimate is typically computed via Generalized Advantage Estimation (GAE) [8], which uses the critic $V_\phi(s_t^\tau)$ as a baseline value function. The critic parameters ϕ are updated by minimizing the loss (e.g. mean squared error) between predicted and bootstrapped returns R_t^τ (see [8] for details):

$$L_V(\phi) = \hat{\mathbb{E}}_{\tau,t}\left[\left(V_\phi(s_t^\tau) - R_t^\tau\right)^2\right]. \quad (8)$$

This process of collecting trajectories and updating the actor and critic is repeated for multiple training iterations.

C. Background: Transformers and the Attention Mechanism

The Transformer architecture has been shown to be especially powerful in capturing long-range dependencies through the self-attention mechanism [10]. Given a set of input embeddings $X = [x_1, \dots, x_n]$, they are first projected into queries (Q), keys (K), and values (V) using learned weight matrices:

$$Q = XW_Q, \quad K = XW_K, \quad V = XW_V. \quad (9)$$

The core of the mechanism is computing a pairwise score between each query and all keys, which is then scaled and normalized via softmax to produce attention weights. The final output is a weighted sum of the values.

$$\text{Attention}(Q, K, V) = \underbrace{\text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)}_{\text{Attention Weights}} V. \quad (10)$$

where d_k denotes the dimensionality of the key (and query) vectors. This mechanism's global receptive field is critical for a centralized MARL critic to model system-wide, long-range interactions. Multi-head attention performs this computation in parallel, capturing diverse interaction patterns that are especially useful for a shared actor learning a generalizable policy. Furthermore, recent theoretical work investigating self-attention through the lens of interacting entities has provided a strong motivation for its application in modeling the complex dynamics of multi-agent systems[9].

D. Background: Graph Attention Networks

While the global self-attention mechanism is powerful, most networked systems exhibit a known graph structure, such as physical or communication networks. To leverage this topology as an inductive bias, we employ the Graph Attention Network (GAT) architecture, which adapts self-attention to graph domains[11].

In a GAT, instead of attending to all other agents in the system, each agent i computes attention coefficients α_{ij} exclusively over its connected neighbors $j \in \mathcal{N}_i$.

$$\alpha_{ij} = \frac{\exp\big(\text{LeakyReLU}\left(\mathbf{a}^\top[\mathbf{W}h_i \,\|\, \mathbf{W}h_j]\right)\big)}{\sum_{k \in \mathcal{N}_i} \exp(\text{LeakyReLU}(\mathbf{a}^\top[\mathbf{W}h_i \,\|\, \mathbf{W}h_k]))}. \quad (11)$$

This approach elegantly combines the flexibility of learned, attention-based weighting with the strong structural prior of the graph. For our shared actor, this allows agents to learn a generalizable, structure-aware policy by dynamically weighting information from relevant neighbors. For the centralized critic, stacking GAT layers enables information to propagate across the graph, allowing it to build a global value estimate that still respects the underlying network topology.

III. METHODS

Shared Transformer Actor–Critic with Counterfactual Advantage (STACCA) is designed to address the two key challenges: modeling *long-range interactions* and achieving *network-generalizability* in network control problems. In Section III-A, we introduce (1) a graph-transformer critic architecture designed to learn global, long-range dependencies among agents to guide local policy learning and (2) a graph-transformer actor architecture designed to adapt to heterogeneous local neighborhoods for network-generalizability. Furthermore, in addressing the above challenges, we require training on relatively large graphs to both capture long-range interactions and expose the actor to diverse local network topologies. This, in turn, exacerbates the significant challenge of *multi-agent credit assignment*. To address this, we introduce a novel counterfactual advantage which we

cover in Section III-B. Finally, we present the complete STACCA framework in Section III-C.

A. Graph Transformer Actor and Critic Architectures

To address the challenges of modeling long-range interactions and learning a network-generalizable shared policy, we introduce critic and actor networks, respectively, based on a hybrid Graph Transformer architecture. Both networks combine the strengths of Graph Attention Networks (GATs) for leveraging the explicit graph structure with the power of Transformer self-attention for learning (potentially) long-range global dependencies. However, while sharing this foundational structure, the actor and critic differ in their scope and final aggregation mechanisms, tailored to their distinct roles (See fig. 1).

The *centralized Graph Transformer Critic* processes the entire graph to learn the global state-value function $V_\phi(s)$. Each node i begins with raw features s_i, which are first projected through an MLP embedding transformation $f(\cdot)$ to obtain initial node embeddings $e_i = f(s_i)$. These embeddings are then refined by a stack of GAT layers, producing structure-aware representations $h_i = g(e_i)$. The resulting collection $H = [h_1, \ldots, h_N]^\top$ serves as the Transformer Encoder input (X in eq. (9)). This two-stage process of extracting a structural inductive bias through the GAT layers then refining the embeddings with Transformer Encoder to model long-range interactions serves as the backbone for STACCA's critic (and actor) architectures. Following the GAT and Transformer layers, it employs an attentional aggregation mechanism to produce a single embedding vector representing the entire graph state. This learned, weighted average of all node embeddings creates a stable and holistic representation, which is then passed to a feed-forward network to produce the final scalar value estimate.

The *decentralized Graph Transformer Actor* is a shared policy network $\pi_\theta(a_i|o_i)$ applied independently at each node to produce an action distribution. The structure of the network is similar to the critic, except the input is agent/node i's local observation o_i. The attention-based architecture allows the shared policy to adapt naturally to heterogeneous local topologies. After neighborhood processing, the actor selects the ego-node embedding, which is now infused with topological context from the GAT and Transformer layers, as input to the policy head, enabling specialized, context-sensitive behaviors. In contrast, an averaging mechanism–while increasing stability in the critic–can potentially push agents into more uniform, less specialized behaviors in the actor by "smoothing out" the local context.

B. Counterfactual Advantage Calculation

Counterfactual reasoning [3] is a powerful technique used to address the multi-agent credit assignment problem by isolating each agent's contribution to the team rewards. It does so through an advantage formulation that compares an agent's action to an individualized counterfactual baseline – what would have happened if the agent had acted differently – as opposed to the global value function baseline shown in
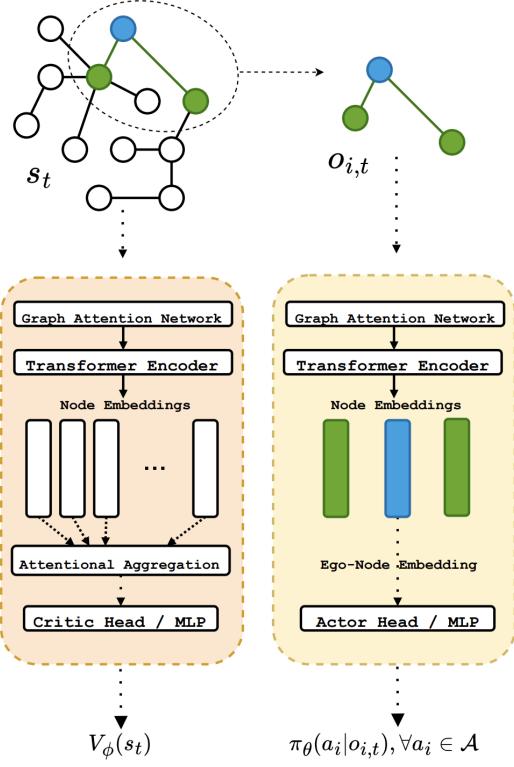

Fig. 1: Graph Transformer Critic (left) and Actor (right) Architectures. Node i is represented by the blue node. Node i's 1-hop neighborhood (green nodes) is its local observation at time t, $o_{i,t}$.

eq. (4). However, existing methods that use this idea typically rely on learning a joint action-value critic $Q(s,a)$. These methods scale poorly with the number of agents and are incompatible with the more stable state-value critic $V_\phi(s)$ used in MAPPO.

We introduce a novel calculation procedure and baseline for counterfactual advantages that is compatible with MAPPO's state-value critic and is effective even with a large number of agents. While a state-value critic $V_\phi(s)$ enhances stability compared to a joint action-value critic $Q_\phi(s,a)$, it does not directly provide the action-dependent values needed for counterfactual reasoning. Our method extracts this information using an efficient, three-step procedure:

1. One-step Trajectory Branching. For each agent i at each timestep t, we generate one-step counterfactuals. We iterate through every possible action $a_i' \in \mathcal{A}_i$ and compute the counterfactual next state $s_{t+1}'(a_i')$ that would have occurred if agent i had taken action a_i' while all other agents' actions $a_{-i,t}$ remained the same (See fig. 2). For continuous actions, this can still be accomplished by discretizing or sampling from the action space.

2. Compute Advantages. We use the centralized critic V_ϕ to compute the counterfactual advantage. This is the difference between the one-step return of the agent's taken action and the policy-weighted average of the one-step returns of

all its possible actions:

$$\hat{A}_{i,t}^{CF} = [r_t + \gamma V_\phi(s_{t+1})]$$
$$- \sum_{a_i' \in \mathcal{A}_i} \pi_\theta(a_i'|o_{i,t})[r_t'(a_i') + \gamma V_\phi(s_{t+1}'(a_i'))] \quad (12)$$

where r_t and $r_t'(a_i')$ are the rewards from the actual and counterfactual actions, respectively. This provides each agent with a personalized baseline, isolating its individual contribution.
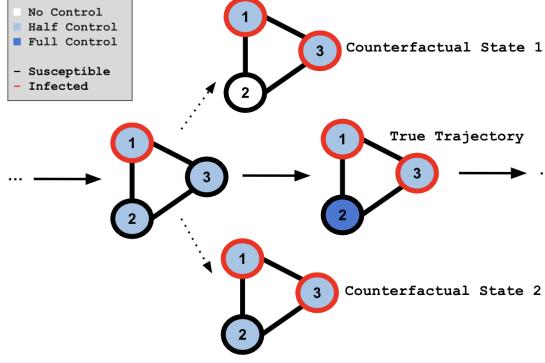

Fig. 2: One-step trajectory branching for Node 2 in a 3 node epidemic containment environment.

3. Amplify credit-assignment signal. A stable, low-variance critic may produce only small differences in value between these counterfactual branches. To amplify these crucial learning signals, we introduce an *timestep-level normalization* step. At each timestep, we normalize the computed advantages across all N agents:

$$\tilde{A}_{i,t}^{CF} = \frac{\hat{A}_{i,t}^{CF} - \mu_{A_t}}{\sigma_{A_t} + \epsilon} \quad (13)$$

where μ_{A_t} and σ_{A_t} are the mean and standard deviation of the advantages of all agents at timestep t. This procedure rescales each agent's contribution relative to the team's average performance at that specific moment and is particularly effective in systems with many agents, where these cross-agent statistics are meaningful.

Finally, to integrate these agent-specific normalized advantages into the clipped surrogate objective (7), we use the following updated objective:

$$L_\pi^{CLIP,CF}(\theta) = \hat{\mathbb{E}}_{i,\tau,t}\Big[\min\big(\rho_{i,t}^\tau(\theta) \, \tilde{A}_{i,t}^{\tau,CF},$$
$$\text{clip}\big(\rho_{i,t}^\tau(\theta), 1-\epsilon, 1+\epsilon\big) \, \tilde{A}_{i,t}^{\tau,CF}\big)\Big]. \quad (14)$$

This procedure is computationally efficient. In our networked environments, an agent's action has a localized effect, so constructing each counterfactual state is an $O(1)$ operation. The total complexity per timestep is thus $O(AN)$ for all agents (where $A = |\mathcal{A}_i|$), which does not increase the asymptotic complexity of the MAPPO training loop.

Our approach builds on the principles of counterfactual reasoning, most notably introduced by COMA [3]. The one-step trajectory branching with timestep-level normalization

efficiently extracts action-dependent values from a state-value critic $V_\phi(s)$ and processes them into effective learning signals. This enables compatibility with a broader variety of MARL algorithms and scales well to many agents.

C. Proposed Method: STACCA

We now present the complete STACCA framework integrated into the base MAPPO training loop; however, STACCA is designed as a modular framework, and its core components can be integrated into various actor-critic MARL algorithms.

Specifically, the Graph Transformer Actor serves as the policy network π_θ, and the Graph Transformer Critic serves as the value network V_ϕ. The standard GAE advantage calculation used in baseline MAPPO is replaced by our counterfactual advantage, $\tilde{A}^{CF}$. This agent-specific advantage is then used to update the shared actor network π_θ via the clipped surrogate objective (14). The centralized critic V_ϕ is trained concurrently by minimizing the huber loss or mean-squared error between its value predictions and the bootstrapped returns R_t over a set of trajectories $\mathcal{D}$. The high-level training loop is presented in Algorithm 1.

Algorithm 1 STACCA w/ MAPPO Training Loop

1: Initialize Graph Transformer actor π_θ and critic V_ϕ.
2: **for** each training iteration **do**
3:     Collect trajectories $\mathcal{D}$ and one-step counterfactual states $\mathcal{C}$ with policy π_θ for all agents.
4:     Compute state values for $\mathcal{D}$ and $\mathcal{C}$ with V_ϕ.
5:     Compute bootstrapped returns using Generalized Advantage Estimation, $R_t^\tau, \forall \tau, t$.
6:     Compute counterfactual advantages $\tilde{A}_{i,t}^{\tau,CF}, \forall \tau, i, t$ using (12) (13).
7:     **for** K_π epochs **do**
8:         Update π_θ by minimizing $L_\pi^{CLIP,CF}(\theta)$ on $\mathcal{D}$.
9:     **end for**
10:     **for** K_V epochs **do**
11:         Update V_ϕ by minimizing $L_V(\phi)$ on $\mathcal{D}$.
12:     **end for**
13: **end for**

IV. EXPERIMENTS AND ANALYSES

Our experiments are designed to answer several key questions regarding the framework's actor and critic architectures, credit assignment mechanism, and overall effectiveness. **(1) Long-Range Interactions:** Do the long-range interactions captured by the critic's global self-attention and the structural inductive biases learned by GAT layers lead to better policies? **(2) Network-Generalizability:** Does STACCA's shared actor architecture improve policy learning and successfully generalize to network topologies and scales unseen during training? **(3) Credit Assignment:** Does our proposed counterfactual advantage effectively address the multi-agent credit assignment problem, leading to improved and more stable learning for decentralized policies?

To answer these questions, we begin by describing the training environments. We then present a comprehensive set

of ablation studies to validate the individual contributions of STACCA's core components. Finally, we conduct extensive generalization and scalability experiments to evaluate STACCA's ability to transfer policies to networks of varying structures and sizes. Our results demonstrate that STACCA not only outperforms baseline architectures but also learns robust, generalizable policies that are effective even under significant changes in the network environment.

A. Training Environments

For all training experiments, we use a 50-node Barabási-Albert (BA) [1] graph with the attachment parameter $m = 1$. This topology was chosen for its diverse degree distribution, which features a few high-degree hubs and many low-degree nodes. This heterogeneity exposes the actor to a wide variety of local neighborhood structures during training. Furthermore, the BA ($m = 1$) graph maintains a relatively high diameter, creating scenarios where the critic must learn long-range credit assignment to inform the actor's policy.

In both environments, each agent observes only its 1-hop local neighborhood. The action space is discrete, allowing agents to either increase, decrease, or maintain their current control (or boosting) factor by increments of 0.1, with values clipped to the range $[0, 1]$. In the actor input, each node/agent comprises of five features: (1) a binary indicator identifying the ego node, (2) the node's state—infected/susceptible in the epidemic setting or aware/unaware in the rumor-spreading setting, (3) the current control or boosting-factor level, (4) the node degree, and (5) the shortest-path distance from the ego node. The critic input includes the corresponding node-level features for all nodes, excluding the ego-node indicator and distance features, since these are defined relative to each agent's local perspective.

1) Epidemic Containment Environment: In the Epidemic Containment environment, agents apply control to limit the spread of an infection. Each training episode begins with three randomly selected seed nodes, a base transmission rate of 0.15, and a control effectiveness of 0.9. The reward function is designed to balance infection mitigation with cost-efficiency. An exponential penalty is applied to control usage, encouraging initial control usage, but discouraging excessive intervention. A softened catastrophe penalty, implemented as a flipped sigmoid function, penalizes the system when the number of infected nodes approaches an epidemic threshold. This is supplemented by a smaller linear penalty on the total number of infected nodes. This reward structure incentivizes agents to use control judiciously while preventing widespread outbreaks (see fig. 3). Additionally, a bonus of 3 per timestep is awarded when the infection is *eradicated* (0 infected nodes), incentivizing rapid and complete elimination.

2) Rumor Spreading Environment: In this environment, the goal is to maximize the spread of information from a set of seed nodes (3 in this case). The dynamics are governed by a maximum transmission rate of 0.25 (at full boosting factor) and a market saturation factor of $\kappa = 3$, which makes spreading more difficult as more nodes become aware. For the reward function, an exponential penalty is applied to boosting-factor usage and a linear reward is applied to the aware nodes, directly incentivizing propagation.

B. STACCA Ablation Experiments

We conduct a series of ablation studies to isolate and quantify the contribution of each key component of STACCA (see fig. 4). To evaluate the critic architecture, we compare the STACCA critic to a version with the global attention mechanism removed (GAT-only) and a version with both graph-based and global attention removed (a standard MLP architecture). To evaluate the actor Architecture, we compare the graph-based STACCA actor to an MLP-based actor. Finally, we compare STACCA's counterfactual advantage to the standard Generalized Advantage Estimation (GAE) with trajectory-level normalization, as is conventionally performed in MAPPO.

1) Critic Comparison: Incorporating the graph-based and global attention mechanisms demonstrated a clear performance benefit. The MLP critic performed sub-optimally, highlighting the value of the structural inductive bias from GAT layers. Adding the global transformer on top of the GAT layers yielded further improvements, with a more pronounced effect in the rumor-spreading environment. This suggests that the global attention mechanism is particularly vital when task dynamics, such as the saturation component in rumor spreading, depend on global state information.

2) Actor Comparison: The STACCA actor outperformed the MLP Actor in both environments, with a more significant improvement in the rumor-spreading task. However, the primary benefit of the STACCA actor architecture lies in its generalizability, a crucial feature evaluated in Section IV-C.

3) Advantage Comparison: The counterfactual advantage had a significant impact on performance in both scenarios. The conventional GAE, applied uniformly to all agents, provides a weak and noisy learning signal in a system with 50 agents. This result underscores the importance of agent-specific credit assignment in large-scale MARL settings.

C. Network Generalization and Scalability Experiments

To evaluate the ability of STACCA's learned policies to generalize, we test policies trained on the 50-node BA ($m = 1$) graph on four unseen network topologies at two different scales (100 and 1000 nodes). The test graphs include two Watts-Strogatz (WS) graphs ($k = 4$) [12] with rewiring probabilities $p = 0.1$ and $p = 0.5$, and two Barabási-Albert (BA) graphs with $m = 1$ and $m = 2$.
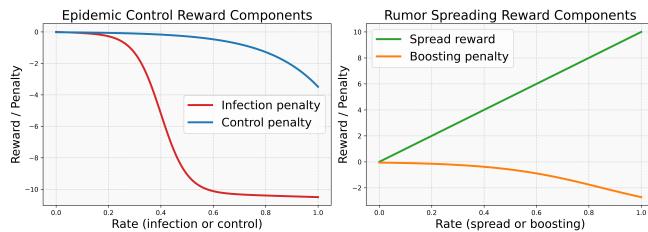

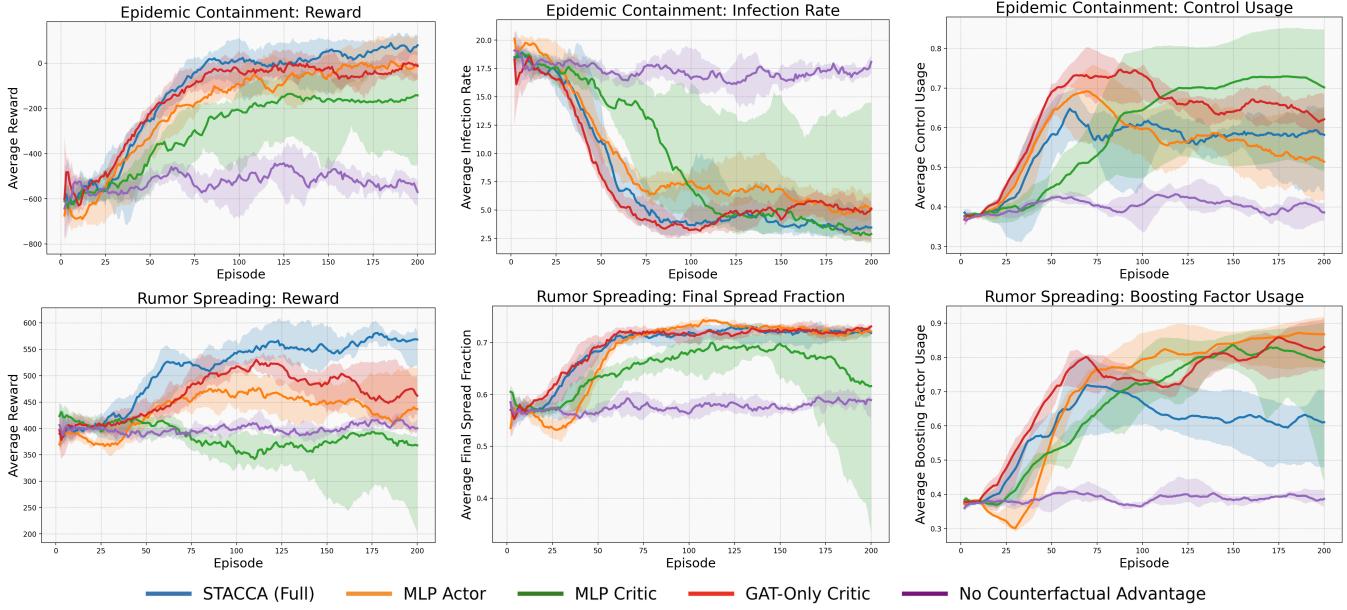Fig. 3: Reward Shaping Visualization.

Fig. 4: Ablation Experiments for STACCA: Comparing STACCA, to STACCA w/ MLP Actor, STACCA w/ MLP Critic, STACCA w/ GAT-Only Critic, and STACCA w/ no counterfactual advantage for the epidemic containment environment (top row) and rumor-spreading environment (bottom-row).
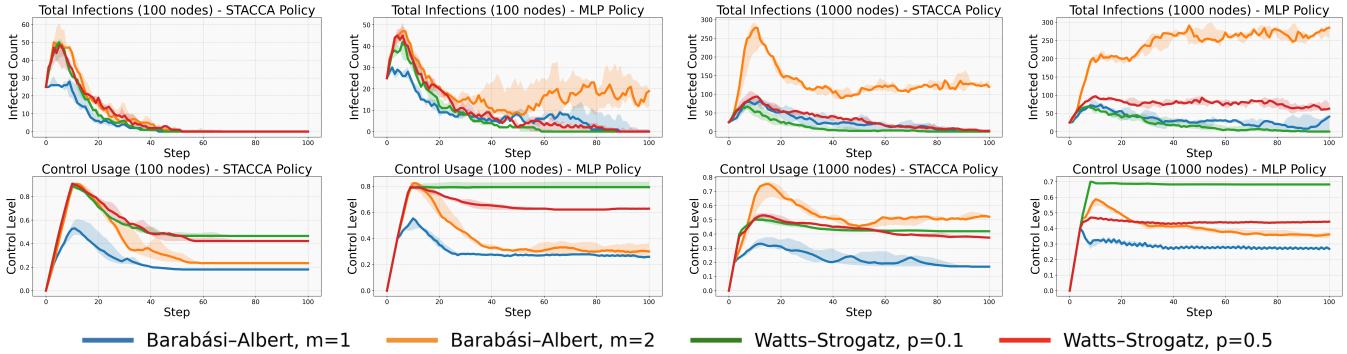

Fig. 5: Epidemic Containment Environment: Comparison of MLP Policy and STACCA Graph-Transformer Policy on 100-node and 1000-node graphs of each of the 4 graph types. All examples use 25 infected seed nodes.

We compare the performance of the STACCA actor architecture against an identically trained MLP actor. For each actor type, we use the highest-performing model based on rewards from 10 independent training runs.

1) Epidemic Containment: As shown in fig. 5, the STACCA policy demonstrates strong generalization. For the 100-node graphs, the STACCA actor successfully eradicates the infection on all four graph types within 50 timesteps using moderate control. In contrast, the MLP actor takes nearly twice as long on three of the graphs with higher control usage, and fails to eradicate the infection on the BA ($m = 2$) graph, where ~20% of nodes remain infected. At a larger scale (1000 nodes), the STACCA actor still eradicates the infection on three of the four graphs. On the more challenging BA ($m = 2$) graph, it contains the infection to 10-15% of the population. The MLP actor's performance degrades substantially at this scale, failing to eradicate the infection on three of the four graphs and allowing severe

outbreaks (~30% infected) on the BA ($m = 2$) graph.

The reduced performance on the BA ($m = 2$) graph, especially at scale, can be attributed to its degree distribution. As BA graphs grow, the degree of hub nodes increases significantly, creating dense local neighborhoods unseen during training. This effect is more pronounced for $m = 2$, where total edge density is approximately double that of $m = 1$. The more uniform local structure of WS graphs likely contributes to the more consistent performance.

We also observe an emergent behavior where the STACCA policy maintains a low level of precautionary control even after eradication. This behavior reduces the peak number of infections in subsequent outbreaks. As shown in fig. 6, this precautionary state is more effective than a naive, uniform control initialization of equivalent global control usage. However, the difference is minor, so we leave this for further investigation.
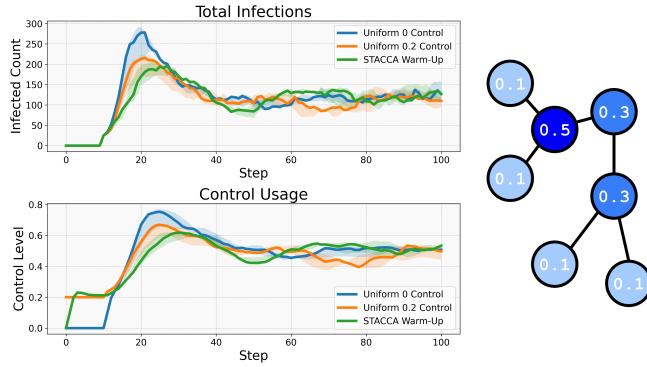
Fig. 6: Node initialization comparison on BA ($m = 2$) 1000 node graph. 25 infected seeds are introduced at $t = 10$ (left). Visual toy example of a precautionary baseline state (right).

2) Rumor Spreading: As shown in fig. 7, the STACCA actor generalizes effectively across different network structures. It achieves higher final spread on denser networks (BA ($m = 2$), WS ($p = 0.5$)), effectively exploiting the network structure. The MLP actor achieves a comparable final spread but employs a significantly less efficient strategy with excessive boosting-factor usage. At 1000 nodes, STACCA maintains its efficient, high-performance strategy. While the final spread fraction is slightly lower, this is likely an artifact of the fixed 100-timestep evaluation window. The MLP policy continues to exhibit high-cost, inefficient behavior at scale.
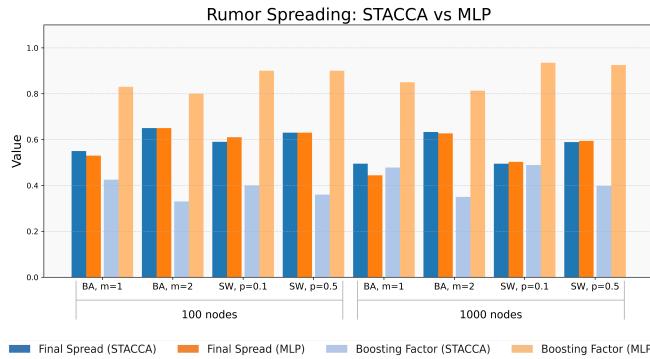

Fig. 7: Rumor Spreading Environment: Comparing STACCA Policy and MLP Policy on final spread proportion and average boosting-factor usage over 100 timesteps. 5 seed nodes are used in all cases.

V. CONCLUSION

In this work, we addressed two key challenges in the application of MARL solutions to networked systems: capturing long-range dependencies and generalizing across network topologies. We introduced STACCA, a framework whose Graph Transformer Critic learns global, long-range system dynamics, while its shared Graph Transformer Actor learns a network-generalizable policy, enabling zero-shot transfer to new environments. By integrating a novel counterfactual baseline in the advantage formulation, STACCA addresses

the credit assignment problem at scale. Our experiments show that STACCA successfully generalizes to networks of diverse structures and significantly larger sizes than those seen during training. These results highlight the potential of transformer-based architectures to create scalable, adaptable, and effective control policies for complex, real-world multi-agent systems.

REFERENCES

[1] Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *science*, 286(5439):509–512, 1999.

[2] Xin Chen, Guannan Qu, Yujie Tang, Steven Low, and Na Li. Reinforcement learning for selective key applications in power systems: Recent advances and future challenges. *IEEE Transactions on Smart Grid*, 13(4):2935–2958, 2022.

[3] Jakob N. Foerster, Gregory Farquhar, Triantafyllos Afouras, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence*, pages 2798–2805. AAAI, 2018.

[4] Manuel Herrera, Marco Pérez-Hernández, Ajith Kumar Parlikad, and Joaquín Izquierdo. Multi-agent systems and complex networks: Review and applications in systems engineering. *Processes*, 8(3):312, 2020.

[5] Landon Kraemer and Bikramjit Banerjee. Multi-agent reinforcement learning as a rehearsal for decentralized planning. *Neurocomputing*, 190:82–94, 2016.

[6] Ryan Lowe, 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.

[7] Guannan Qu, Adam Wierman, and Na Li. Scalable reinforcement learning of localized policies for multi-agent networked systems. In Alexandre M. Bayen, Ali Jadbabaie, George Pappas, Pablo A. Parrilo, Benjamin Recht, Claire Tomlin, and Melanie Zeilinger, editors, *Proceedings of the 2nd Conference on Learning for Dynamics and Control*, volume 120 of *Proceedings of Machine Learning Research*, pages 256–266. PMLR, 10–11 Jun 2020.

[8] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 6, 2015.

[9] Muhammed Ustaomeroglu and Guannan Qu. A theoretical study of (hyper) self-attention through the lens of interactions: Representation, training, generalization. In *Forty-second International Conference on Machine Learning*, 2025.

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 5998–6008, 2017.

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

[12] Duncan J Watts and Steven H Strogatz. Collective dynamics of 'small-world' networks. *nature*, 393(6684):440–442, 1998.

[13] Han Xu, Jialin Zheng, and Guannan Qu. A scalable network-aware multi-agent reinforcement learning framework for decentralized inverter-based voltage control. *CoRR*, abs/2312.04371, 2023.

[14] 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. In *Advances in Neural Information Processing Systems (NeurIPS) 2022, Datasets and Benchmarks Track*, volume 35, pages 24611–24624, 2022.