# Towards Decentralized Social Reinforcement Learning via Ego-Network Extrapolation

Extended Abstract

Mahak Goindani
Department of Computer Science
Purdue University
West Lafayette, IN
mgoindan@purdue.edu

Jennifer Neville
Departments of Computer Science and Statistics
Purdue University
West Lafayette, IN
neville@purdue.edu

## ABSTRACT

In this work, we consider the problem of multi-agent reinforcement learning in directed social networks with a large number of agents. Network dependencies among user activities impact the reward for individual actions and need to be incorporated into policy learning, however, directed interactions entail that the network is partially observable to each user. When estimating policies locally, the insufficient state information makes it challenging for users to effectively learn network dependencies. To address this, we use parameter sharing and *ego-network extrapolation* in a decentralized policy learning and execution framework. This is in contrast to previous work on social RL that assumes a centralized controller to capture inter-agent dependencies for joint policy learning. We evaluate our proposed approach on Twitter datasets and show that our decentralized learning approach achieves performance nearly equivalent to that of centralized learning approach and superior performance to other baselines.

**ACM Reference Format:**
Mahak Goindani and Jennifer Neville. 2021. Towards Decentralized Social Reinforcement Learning via Ego-Network Extrapolation: Extended Abstract. In *Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), Online, May 3–7, 2021*, IFAAMAS, 3 pages.

## 1 INTRODUCTION

Recent work on multi-agent reinforcement learning (MARL) for social network settings [1, 3, 4] has assumed a fully observable environment where each agent can observe the activities of all other agents in the network, with a common shared reward between all agents to be optimized. They employ a joint model (centralized training) to capture inter-agent dependencies by learning the collective action of all users conditioned on the complete network state across all users (centralized execution). In this work we consider the social network setting where each user receives a different reward and has a different partial observation of the environment—thus, we cannot employ centralized learning and centralized execution.

In contrast to centralized MARL, decentralized learning and decentralized execution focuses on learning a separate policy for every agent individually, and actions are conditioned solely on the local state and observation of the agent. Learning policies independently allows to preserve the users' privacy as there is no sharing of information [7]. However, due to sparse interaction data in social networks, the number of samples available for each user in decentralized learning is quite sparse. As such, it is challenging to learn accurate policies—since each agent does not have sufficient information available to capture inter-agent dependencies, decentralized learning is likely to lead to in large errors due to variance. To address this, we propose to perform *partially centralized learning* with decentralized execution. Specifically, we consider a single policy function that maps the local state and observation of an agent to an action. We use parameter sharing to learn this function across users, which offsets the data sparsity issues that would arise in fully decentralized learning. By only sharing model parameters sequentially, and not the user trajectories, we aim to provide more autonomy to agents and safeguard their privacy.

Given the partial observations of each user, to learn accurate policies we propose to use *ego-network extrapolation* to estimate the hidden state information, and exploit the local network structure and relations to learn inter-agent dependencies. In a social network with directed Followee-Follower relationships, a user can observe the state of her Followees, but not those of her Followers. Since the reward a user obtains depends on her Followers' states, the user needs to estimate her Followers' states, in order to improve her policy. We utilize the observation that each user has two roles in the network—a Followee to some users, and a Follower of other users—to locally learn the dependency between Followees and Followers. Our key idea is to perform ego-network extrapolation over the local neighborhood of a user, by first learning the dependency from the activities of her Followees, and then extrapolating those to estimate the activities of her Followers.

To our knowledge, this is the first MARL approach to exploit user relations in a *partially observable* social network, by transferring the knowledge learnt from one set of users to another and estimate the hidden environment information for policy learning. Note that the mapping between a user's Followees and the user (as a Follower) is learnt locally by the user, and is a many-to-one mapping. Thus, the challenge is to extrapolate/project this mapping from the user (as a Followee) to her Followers, which is one-to-many mapping. To overcome this challenge, we capture reciprocity in user interactions, which helps us learn a mapping over the set of users that can be easily projected to a many-to-many mapping (producing better estimates). Additionally, different Followees have different impact on the activities of their Followers, and this impact changes over time based on the dynamic user activities

*Proc. of the 20th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2021), U. Endriss, A. Nowé, F. Dignum, A. Lomuscio (eds.), May 3–7, 2021, Online.*

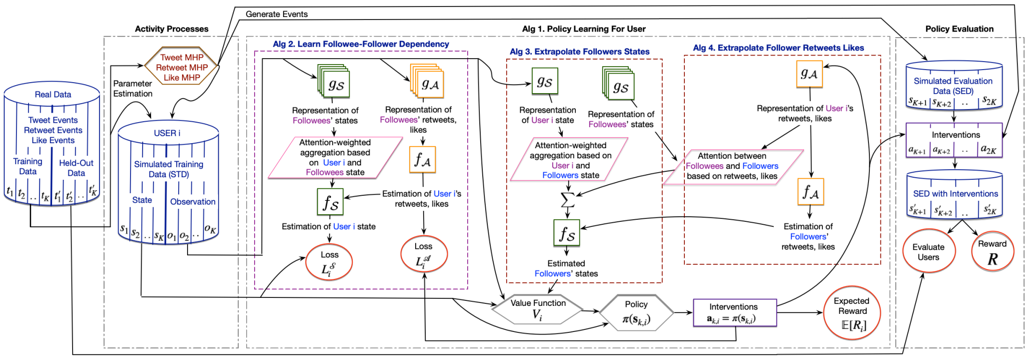
Generate Events
Activity Processes
Alg 1. Policy Learning For User
Policy Evaluation
Real Data
Tweet Events
Retweet Events
Like Events
Training Data
Held-Out Data
$t_1$ $t_2$ .. $t_K$ $t'_1$ $t'_2$ .. $t'_K$
Parameter Estimation
Tweet MHP
Retweet MHP
Like MHP
USER i
Simulated Training Data (STD)
State
Observation
$s_1$ $s_2$ .. $s_K$ $o_1$ $o_2$ .. $o_K$
Alg 2. Learn Followee-Follower Dependency
$g_{\mathcal{S}}$
Representation of Followees' states
$g_{\mathcal{A}}$
Representation of Followees' retweets, likes
Attention-weighted aggregation based on User i and Followees state
$f_{\mathcal{A}}$
$f_{\mathcal{S}}$
Estimation of User i's retweets, likes
Estimation of User i state
Loss $L_i^{\mathcal{S}}$
Loss $L_i^{\mathcal{A}}$
Alg 3. Extrapolate Followers States
$g_{\mathcal{S}}$
Representation of User i state
$g_{\mathcal{S}}$
Representation of Followees' states
Attention-weighted aggregation based on User i and Followers state
$\sum$
$f_{\mathcal{S}}$
Estimated Followers' states
Alg 4. Extrapolate Follower Retweets Likes
$g_{\mathcal{A}}$
Representation of User i's retweets, likes
Attention between Followees and Followers based on retweets, likes
$f_{\mathcal{A}}$
Estimation of Followers' retweets, likes
Value Function $V_i$
Policy $\pi(\mathbf{s}_{k,i})$
Interventions $\mathbf{a}_{k,i} = \pi(\mathbf{s}_{k,i})$
Expected Reward $\mathbb{E}[R_i]$
Simulated Evaluation Data (SED)
$s_{K+1}$ $s_{K+2}$ .. $s_{2K}$
Interventions
$a_{K+1}$ $a_{K+2}$ .. $a_{2K}$
SED with Interventions
$s'_{K+1}$ $s'_{K+2}$ .. $s'_{2K}$
Evaluate Users
Reward $R$

**Figure 1: Overview of DENPL Policy Learning for a user via Ego-Network Extrapolation**

and interactions. We incorporate the dynamic peer-influence by learning *attention* between activities of each Followee-Follower pair, to further improve estimates of hidden state, and thus, the policy for each user.

## 2 APPROACH

We consider a social network setting with $N$ users, where each user $i \in \{1, ..., N\}$ is an agent. Users have a Followee-Follower relationship, and interact via $d$ different network activities (e.g. tweet, retweet, like). We represent the followers adjacency matrix for the social network graph using $\mathbf{G}$, where $G_{ji} = 1$ if $i$ follows $j$, and 0 otherwise. We consider a directed social network graph where the information flow is only in one direction. Due to this, an agent can observe the activities of her Followees $\mathbf{G}_{.i}$, but she can't observe the activities of her Followers $\mathbf{G}_{i.}$. Thus, the environment is partially observable to an agent, and we refer to the user's view as a partially observable ego-network. Let the *state* of a user $i$, $\mathbf{s}_i \in \mathbb{R}^d$ corresponds to $d$ network activities that she performs, and her local *observation* $\mathbf{o}_i$ is the partial view of the environment that corresponds to the activities of her Followees. Given the individual state and observation, each user learns a policy $\pi_i : \mathbf{s}_i \rightarrow \mathbf{a}_i$ to obtain *actions* that maximize her *local reward* $R_i$.

Fig. 1 illustrates the different components of our system. We learn a set of Multivariate Hawkes Processes (MHP) for different network activities, from the training data. We use the MHP models to simulate additional training data, i.e. events, for learning the policy. We map the excitation events to states and observations in a Partially Observable Markov Decision Process, where each user learns the amount by which she needs to increment her *intensity function* for true news diffusion. For learning the policy for a user $i$, our idea is to learn the dependency between the activities of Followees and Followers, and utilize it to estimate the hidden state of $i$'s Followers, using the state of $i$'s Followees. Specifically, we learn,

**Model $g_{\mathcal{S}}$** *Input*: user $i$'s Followees' states; *Output*: generalized representation for the Followees' states, using a stacked auto-encoder.

**Model $f_{\mathcal{S}}$** Mapping from the generalized representation of user $i$'s Followee's states to user $i$'s (i.e., Follower's) state. Note this a many-to-one mapping.

We learn a many-to-one mapping from the state of Followees to a user's state. However, while extrapolating the states of Followers from the user's state, it is a one-to-many mapping. This results in less accurate estimates of Followers' states, since there is only one input signal to estimate the states of multiple Followers. To address this, we utilize pairwise user interactions, i.e., the retweets and likes that a user provides to her Followees, to learn a many-to-many mapping that can be used to effectively extrapolate the Followers' states, described next.

**Model $g_{\mathcal{A}}$** *Input*: retweets/likes of user $i$'s Followees; *Output*: generalized representation for the Followees' retweets/likes, using a stacked auto-encoder.

**Model $f_{\mathcal{A}}$** Mapping from the generalized representation of retweets/likes of user $i$'s Followees to the retweets/likes of user $i$ (i.e., Follower). Note this is a many-to-many mapping.

After learning $g_{\mathcal{S}}, g_{\mathcal{A}}, f_{\mathcal{S}}, f_{\mathcal{A}}$ using the activities of user $i$ and her Followees, we apply these to estimate the states of user $i$'s Followers from user $i$'s state and observation. Then, the state of user $i$, along with her Followees' states, and her Followers' estimated states, is used to approximate the value $V_i$, that is the total expected discounted reward of user $i$'s activities, and learn policy $\pi$. We approximate $\pi$ and $V$ using neural networks. Thus, overall we use six neural networks and three MHPs for learning.

We refer to our approach as Decentralized Ego-Network Policy Learning (DENPL). To apply the estimated policy, we simulate data again from the MHPs. Using the learned policy $\hat{\pi}$, we obtain actions that are added to the MHP intensity functions to generate evaluation data, which is then used to measure empirical reward.

## 3 EXPERIMENTS AND RESULTS

We consider an application of Fake News Mitigation [1, 3, 4] to demonstrate the utility of DENPL. We evaluate performance compared to different centralized and decentralized learning approaches, using two real-world Twitter datasets [5, 6]. Compared to other baselines, DENPL achieves performance nearly equivalent to that of the centralized learning method and other approaches that assume full observability of the complete network state. Experiments show that sequential update of parameters by each user individually improves sample efficiency per user and results in more accurate policy estimates. We also observed that the users with policies that increment their intensities to promote true news, receive a greater number of retweets in held out data, compared to other baselines. This illustrates the potential for social RL methods to learn in a decentralized fashion, limiting the need for data sharing across agents. See [2] for a complete description of algorithms, methodology, and experiments.

## REFERENCES

[1] M. Farajtabar, J. Yang, X. Ye, H. Xu, R. Trivedi, E. Khalil, S. Li, L. Song, and H. Zha. Fake news mitigation via point process based intervention. *International Conference on Machine Learning*, 2017.

[2] M. Goindani. *Social reinforcement learning*. PhD thesis, Purdue University Graduate School, 2020.

[3] M. Goindani and J. Neville. Social reinforcement learning to combat fake news spread. *Proceedings of the 35th Conference on Uncertainty in Artificial Intelligence*, 2019.

[4] M. Goindani and J. Neville. Cluster-based social reinforcement learning. *arXiv preprint arXiv:2003.00627*, 2020.

[5] Y. Liu and Y.-F. B. Wu. Early detection of fake news on social media through propagation path classification with recurrent and convolutional networks. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[6] J. Ma, W. Gao, and K.-F. Wong. Detect rumors in microblog posts using propagation structure via kernel learning. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 708–717, 2017.

[7] K. Zhang, Z. Yang, H. Liu, T. Zhang, and T. Başar. Fully decentralized multi-agent reinforcement learning with networked agents. *ICML*, 2018.