Learned Monkeys: Emergent Properties of Deep Reinforcement Learning Generated Networks

Shosei Anegawa[1], Iris Ho[1], Khoa Ly[1], James Rounthwaite[1], and Theresa Migler[1][0000−0001−7857−1100]

California Polytechnic State University, San Luis Obispo, CA 93405, USA
tmigler@calpoly.edu

Abstract. Graph generation methods used in the study of animal social networks suffer from the inability to fully explain individual agent behaviors and values. However, recent advancements in complex networks and machine learning offer a novel way of artificially simulating network formations. We use deep reinforcement learning (DRL) to model the proximity network of white-faced capuchin monkeys. This process of constructing a graph provides insight into the unique, individual social strategies of a network's agents depending on the initial DRL parameters. We generated a network to closely match the characteristics of the proximity network constructed from an observational dataset. For example, our model-generated graph and the observed graph consistently showed a few members with significantly higher betweenness centrality than all other members despite each agent starting with the same parameters.

Keywords: Generative Network Models · Animal Social Networks · White-faced Capuchin Monkeys · Deep Reinforcement Learning

1 Introduction

Our understanding of animal behaviors and animal socialization is underpinned in large part by our knowledge and analyses of their social networks. The study of animal social networks has largely relied on graphs, represented as complex networks of nodes and edges, constructed from data collected by field researchers or experimental observations. Analyses of simulations leveraging those constructed model networks can often provide great insight into the governing rules of social networks, but can struggle to illuminate individual behavior and decision-making processes.

Recent novel work in the field of complex networks has sought to apply deep reinforcement learning (DRL) - a paradigm of machine learning algorithms interested in simulated individuals, called agents, learning and accruing experience through interactions with their environment - to the construction of networks. The application of DRL in this context, referred to as building a network based on reinforcement learning, has produced networks with promising similarity to real network distributions. [8]

The contribution of this paper is an analysis of a dataset of Panamanian large brained white-faced capuchin monkeys from a graph theoretical standpoint, a presentation of the application of the metrics on the network as constructed from observational field data, and the application of a similar process of constructing a network based on reinforcement learning [6]. With continuous experimentation of our reinforcement learning model parameters, we matched the characteristics of our model-generated graph to that of the graph constructed from the observational data.

This paper is organized as follows: first, we offer a brief, non-exhaustive overview of reinforcement learning to provide adequate background for understanding our application of reinforcement learning in constructing the network. Next, we discuss relevant work in order to highlight several recent applications of agent based modeling for studying animal social networks. Subsequently, we present the graph metrics of our white-faced capuchin monkey dataset and our methods for building our network. Finally, we summarize our comparisons and findings between our generated network and the observed network.

2 Related Works

White-faced capuchins are not new to the concept of being analyzed via social networks. Crofoot et al. built a multitude of social networks from extensive observed data to learn more about white-face capuchins from a general social network theory perspective. This analysis has noted patterns of both aggressive as well as cooperative behaviors [1]. Perry et al. has also done extensive work with white-faced capuchins using social networks as a lens to learn more about their unique personalities and group structure [7], culminated as the Lomas Barbudal capuchin Monkey database and project [5]. More recently Perry and Smolla analyzed why these monkeys sometimes perform a complex and awkward series of social interactions coined as 'rituals' [6]. We pull proximity data from this paper as a foundation for our research. Lastly, the only major paper to combine agent-based modeling (ABM) with capuchin monkeys is Kajokaite's publication set to model male monkey dispersal movement. This ABM focuses on predicting specific actions of capuchin monkeys, whereas our model aims to model a proximity network. [4]

There have been many studies using reinforcement learning to generate networks, although most have been used to characterize human social networks. Song et al. used a network generation method using reinforcement learning to represent a human social network. Their generated network was scale-free and small-world, characteristics seen in observed human networks. [8]. Also, work done which represents individual animals as their own agents to model group animal behavior has been found to be successful in studying grizzly bears by Hoegh et al. [2]. While these works used their representative models to predict group movement, the same underlying concept of representing each animal as its own autonomous intelligent individual has also been used to create a complete animal network [1]. Using models in this manner to generate networks is utilized in numerous other fields as well. Notably Jacobs et al. has presented a literature review of the field. Jacobs' work presents the numerous methods of generative networks and has provided sound techniques and optimizations to apply generative models from several perspectives [3]. Our paper uses both Song's and Jacob's work as a launch point for our model development.

The contributions of our work apply reinforcement learning as an agent-based network generation to analyze the unique and sociable white-faced capuchin monkeys.

3 Background

Reinforcement Learning In this paper, we utilize reinforcement learning to model the behavior of capuchin monkeys. Reinforcement learning aims to have an agent maximize reward in the given environment. We define agents as simulated individuals that interact with and learn from the environment and other agents through actions and reward. A reward is a function that maps states of the environment to a positive or negative value. An agent takes actions moving them from different states seeking maximal value. The value is then defined as a long-term reward, or the sum of all the rewards from the initial state to the terminating state. Given a state, an agent will determine the optimal action based on the policy π, which is a function that maps states to actions based on the reward. The goal is to then have an optimal policy, whereby given any state, the agent provides the action that maximizes reward. For the most part, the policy heavily relies on the value function, which is what the agents need to learn. The value function represents the long-term reward given the current policy, and thus desirability, of any given state.

We employ an algorithm known as *Q-learning* for our agents. Q-learning aims to produce a Q-Function which is an extended definition of a value function. Unlike the traditional value function that takes in a state and returns a value given the current policy, the Q-function takes in a state-action pair. $Q_\pi(S, A)$ is defined as the value of taking action A from state S, and then thereafter following the current policy π. In addition to this, Q-learning is known as an online reinforcement learning algorithm, meaning the Q-function is updated during the sampling process, resulting in a dynamic policy that improves throughout an episode. A single update step is defined as follows[9]:

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \underset{a}{max}\, Q(S_{t+1}, a) - Q(S_t, A_t)] \tag{1}$$

Where α is the learning rate, A represents the set of all actions, S is the set of all states, and R represents the immediate reward assigned. γ represents the discount factor, which decreases the value of a reward the farther it is in the future. A bigger α value means that the Q-function is updated in larger steps, which can help the value function converge more quickly, but can also harm convergence if set too high. Our model uses a neural network to approximate this function, with its loss defined as the mean squared error of[9]:

$$\Big(R_{t+1} + \gamma \max_a Q(S_{t+1}, a)\Big) - Q(S_t, A_t) \tag{2}$$

In this case, we aim to get the current prediction of $Q(S_t, A_t)$ closer to $R + \gamma * max_a(Q(S_{t+1}, a))$, which is at least as accurate as $Q(S_t, A_t)$, and serves to bring the current approximate Q-function closer to the correct Q-function.

This then leads us to the policy, which as mentioned previously is determined by the Q-function. In our case, we employ an ϵ-greedy policy. This policy normally chooses actions greedily by using the max_a over actions of $Q(S, A)$, but with a probability of ϵ takes a random action. This exists so that the agent can balance exploring new options and exploiting existing options. Tuning ϵ is an extremely important step in any Q-learning model.

Slowly the agent learns more accurate values in the Q-function and is able to take actions that yield higher rewards. However, the introduction of an ϵ-greedy policy forces the agent to take possibly worse actions, but occasionally this exploration results in new states with greater rewards. This tradeoff of exploring new paths and staying with what is known to work propels Q-learning forward.

4 Observed Dataset

Dataset and Proximity Network Model The data relevant to this paper comes from Perry et al. and the Lomas Barbudal Capuchin Monkey database [5]. More specifically, we will be concerned with the data detailing the number of interactions per dyad of monkeys from the 'Flakes' group observations. The data includes 47 individuals, 761 dyad pairs, and 26797 proximity observations over a multi-year period between 1 February 2004 and 11 October 2018. In this network vertices represent white-faced capuchin monkeys and two monkeys are connected if they were observed within 40 cm of each other. The edges are weighted with an SRI metric, a measure of total proximity observations between dyad pairs normalized by the total scans per individual monkey in the dyad. The resulting network had degrees ranging from 5 to 44 with the mean degree being 26.9. The network had an average path length of 3.615 and an average clustering coefficient of 0.831.

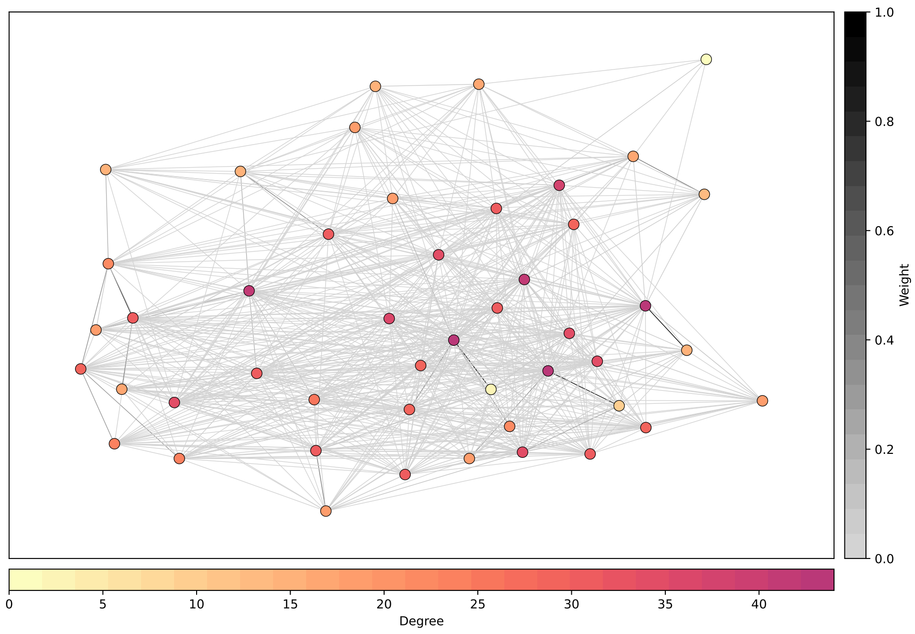
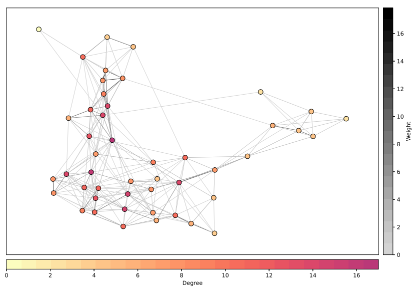


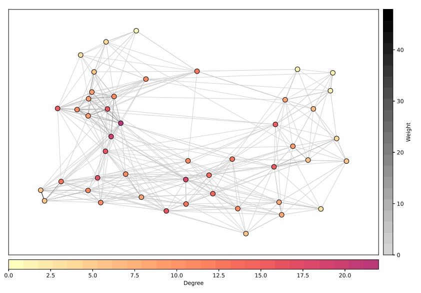
Fig. 1 The proximity graph (47 nodes) of the observed white-faced capuchin dataset with weight defined as SRI

5 Reinforcement Learning Model

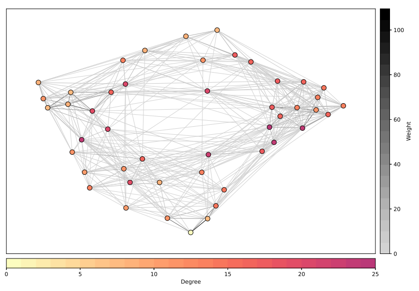
For our model, we follow the general architecture of Song et al. [8]. We define a space ranging from 7×7 to 10×10 states on a board, and place 47 agents in there in order to model our data. Each agent has their own independent Q-function approximator, which are each 3-layer neural networks with 128, 128, and 64 units respectively, eventually returning a number representing the value. During graph formation, we randomly initialize the weights to the Q-function and the positions of each individual on the board, and build an initial graph based on this. At each timestep, an individual can take one of five actions: move up, down, left, right or stay. If two or more individuals end up in the same state, the weights on the graph are incremented by a value ω. If two individuals have a weight greater than zero and are not in the same state in a given timestep, their weights are decremented by a value τ. If at this point, the weights for an edge reach zero, the edge is removed. After the weights are updated at the end of a timestep, rewards are distributed to agents equal to the total of their current weights. This pushes them to both maintain and form new social connections throughout an episode.



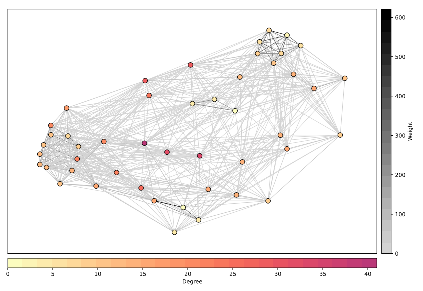
(a) $T = 50$



(b) $T = 100$



(c) $T = 250$



(d) $T = 1500$

Fig. 2 As time progresses the generated network more closely resembles the observed graph

6 Methods

Model Tuning — Episodes and Iterations The first thing we tuned for our model was the decay of weights on the graphs over time τ. We found that this had the greatest impact on the KS-tests, clustering coefficient, and average path length. We settled on a decay $\tau = -0.005$ per timestep, relative to $+1$ reward for forming or maintaining connections. We hypothesize this microscopic decay performed the best due to the observed graph being a proximity graph, where connections cannot be removed once formed. In addition, we tune ϵ which represents the inclination of a monkey to explore new connections or to focus on retaining current connections.

Finally, we tune γ, the discount factor, which represents to what extent monkeys take into account future reward, or how far they look ahead into the future to make decisions.

Generative Network Comparison To test how well our generated graphs matched characteristics of the original graph, we created a baseline model and compared iterations of the RL model and the baseline model to the observed graph. The baseline model has agents taking random actions at every timestep.

We utilized various metrics to show the validity of our generated network. Focusing on just the degree distribution of the generated and observed graph, we utilize a couple of methods to compare equality.

The first is the two-sample Kolmogorov–Smirnov (KS) test, which produces the KS statistic. The KS statistic represents how likely the two sets of degree distributions could have been drawn from the same unknown degree population distribution. The implementation we use generates a KS statistic and an associated p-value. If the p-value is high, then we cannot reject the null hypothesis which states the two samples are from the same distribution. A low KS statistic indicates a high p-value.

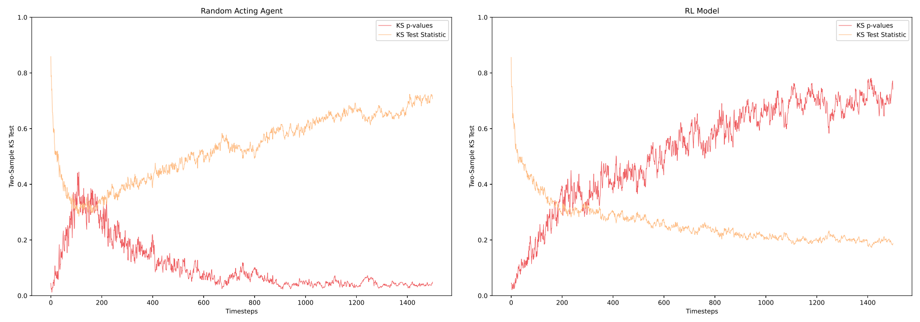
We also compared general graph metrics such as average path lengths and average clustering coefficients. Our goal was to tune the graph to better emulate the observed graph.

Lastly, one comparison we compare is the number of "keystone" individuals in all graphs. Krause et al. defines keystone individuals with a strong influence on the graph's structure and group. Keystone individuals are most commonly identified through centrality metrics [10]. This paper defines keystone individuals as nodes exhibiting a high betweenness centrality.

7 Results

Model Comparison For most of this section, we fixed ϵ at 0.3, the discount factor γ at 0.9, and the board size at 10×10. The graphs are flattened over 50 iterations of 1500 timesteps episodes. As a baseline for showing model improvement we compared the graphs produced by our agents to a second model where agents took random actions at every timestep. We found that most of the metrics that we used tended to converge after the model had been running for 1500 timesteps, so all graphs shown represent the progress of the model leading up to 1500 timesteps. In the following section we will present each metric used to compare our generated graphs to the real-world proximity graphs of the capuchin monkeys. We also experimented with board sizes smaller and larger than 10×10, but over 1500 timesteps, this made little difference.

Degree Distribution Comparison Degree distribution is an important metric for a social network as it gives deep insight into the structure of a network. In this section, by comparing the degree distributions of our generated graphs to that of the observed graphs, we show that our model generates graphs representative of the real world. With the above parameters, the random acting agent performed significantly worse in the KS test. The random acting agent model reached its best metric for the two-sample KS test at around 200 iterations, with a KS-test statistic of around 0.38 and a p-value of 0.36. On the other hand, the RL model reaches a KS-test statistic of 0.17 and a P-value of 0.77. It should also be noted that at low timesteps such as 200 iterations, the graphs tend to be more sparse which makes these tests less reliable. This shows that the degree distribution of our model was significantly more closely aligned to the proximity graph's degree distribution. With a significance level of 0.77, the RL generated a degree distribution similar enough to the observed degree distribution that the two could have come from the same population.

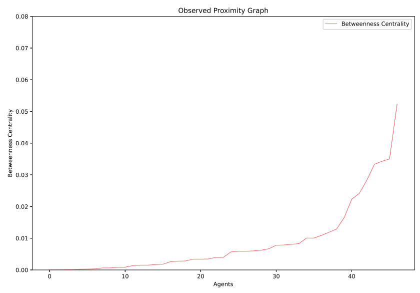


(a) Random Acting Agent **(b)** RL Model

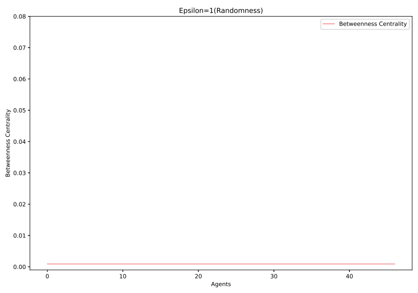
Fig. 3 Throughout an episode, Figure 3b has consistently decreasing KS test statistics and increasing p-values, highlighting similarity between our RL model and the observed proximity network.

It is also interesting to note that both the RL model and random acting agent have similar results up to around 200 timesteps. This makes sense, considering that our Q-function approximator's learning rate parameter is set to a fixed 0.01, so meaningful learning should begin being reflected around after 100–200 updates.

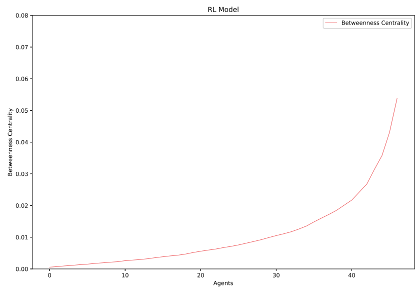
Keystone Individuals/Centrality Comparison The second metric we used was the betweenness centrality of each individual member of the graph. In the observed proximity graph, there are two "keystone" individuals, who had significantly higher betweenness centrality than all other members in the graph and were hubs for social interaction. Interestingly, the graph generated by the RL model often had 2-3 keystone individuals with significantly higher betweenness centrality over others. This means at each iteration some agent-policy optimizations always led towards having 1 or 2 keystone individuals. Shown in Figure 4 centrality over each individual agent/monkey. The similarities in the number of keystone individuals is striking, because it shows how agents learn to build connections in a similar fashion as monkeys in the observed data. Despite all agents being initialized with the same Q-function and parameters, we still observed consistent differences between individual agents in a similar manner to our observed proximity graphs. It was notable to see that even over 50 distinct episodes of 1500 timesteps, a few agents consistently became important within the group of monkeys.



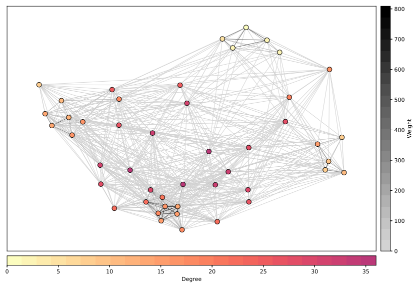
(a) Observed Graph



(b) Random Acting Agents



(c) RL Model

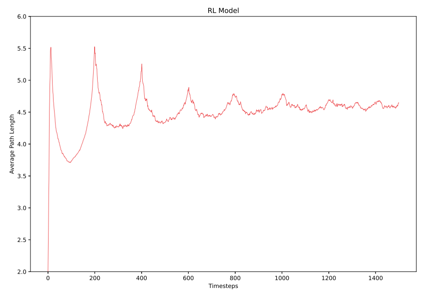


(d) Sample Graph

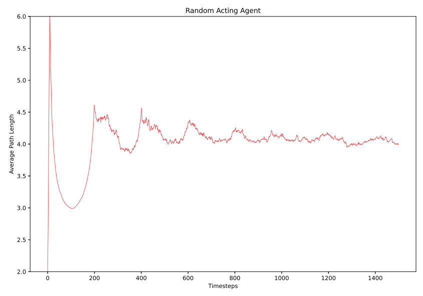
Fig. 4 Figure 4b had most nodes with extremely low betweenness centrality, with each node having the same connections around 1e-5, due to the graph having minimal connections between clusters. The smoothness of the Figure 4c graph results from it being the average over 50 episodes. In addition to this, Figure 4d clearly shows the individuals that are keystone, with very central positions on the graph. In this case, Agents 1 and 11 have the highest centralities

Clustering Coefficient & Average Path Length In general, the random agent graphs generated had a strong clustering coefficient, almost reaching 1.0 over 1500 timesteps, and a lower average path length, at around 3.8. This number might be disproportionately low because given that the graph is relatively sparse, nonexistent paths are not factored in the average path length metric. The RL generated network ended with a clustering coefficient of 0.73 and an average path length of 4.5. Recall that the observed proximity graph had an average clustering coefficient of 0.81, and an average path length of 3.7. From these we see that the RL graph better modeled the observed data on clustering coefficient, but not in average path length.

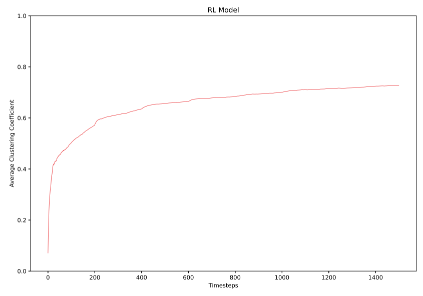
The RL graph generation method outperforms the random acting agent model at every metric aside from average path length when attempting to represent how monkeys act in nature.



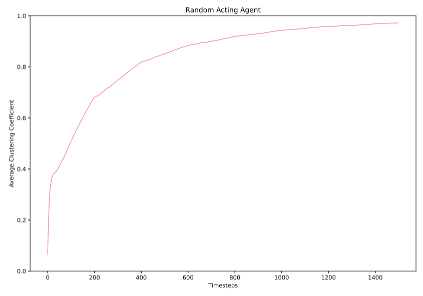
(a)



(b)

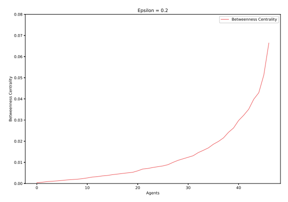


(c)

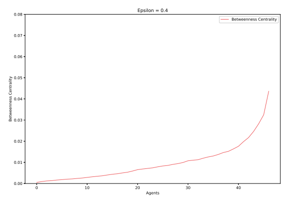


(d)

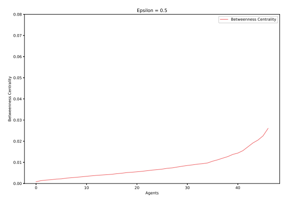
Fig. 5 The random acting agent has a more similar average path length to the observed dataset, while the RL model is more similar in clustering coefficient



(a) $\epsilon = 0.2$



(b) $\epsilon = 0.4$



(c) $\epsilon = 0.5$

Fig. 6 ϵ has a high impact on the existence of clear keystone individuals in the generated graphs. The relative roughness of these graphs is due to them being averaged over a smaller number of episodes

Epsilon Next we varied ϵ, the coefficient of exploration. The metric most correlated with changes in ϵ was the distribution of betweenness centrality, and thus keystone individuals throughout the network. Some significant increments are shown below in Figure 6. In general, as ϵ grows larger, the more prominent keystone individuals become. As it approaches 1, individuals begin to have very evenly distributed connections, and thus behave like a single large cluster rather than a collection of smaller cliques. ϵ is a metric of how exploratory our agents are. Due to the extreme similarity of the centrality of our graph generated with $\epsilon = 0.3$, we can hypothesize that out of every 10 interactions that these monkeys have on average, 3 will likely be with new or weaker connected monkeys, while 7 will be with closer connections.

8 Future Directions

Limitations of the Research Our current dataset only includes the total number of proximity interactions that occurred from a subgroup of white-faced capuchin monkeys. This limited our final networks to be flattened proximity representations of many timesteps. Knowing when each proximity interaction occurred would allow us to compare our model and the observed graphs at different timesteps. Seeing how the model performs as its agents learn could yield additional interesting results.

Increased Agent Uniqueness and Reward Changes To better simulate the behavior of white-faced capuchin monkeys in the wild, additional parameters for each RL agent could be added, such as placing a higher reward for connecting with agents who fall within proximity to higher matriline ranking monkeys. In addition, adding sibling and family relations between agents, or sex-specific characteristics, and tuning rewards could potentially better represent the social structure of the capuchin monkeys.

Social Interactions and Network Adding in social interactions opens a wide range of research opportunities to utilize game theoretics, treating social interactions as games of risk and reward. Furthermore, with enough complexity, we could even model Perry's and Smolla's observations of intricate ritual interactions and relationship quality index [6]. We could also apply our method of generating a network to other primates to see if the patterns persist beyond white faced capuchin monkeys.

9 Conclusion

We modeled white-faced capuchin monkeys as reinforcement learning agents using an ϵ-greedy policy to generate proximity networks. Running our model to generate flattened graphs over 50 iterations of 1500 timestep episodes, we found our graphs had many similar properties to a real-world proximity network. The most prominent result we observed was the consistent presence of a small number of keystone individuals in our generated graphs. We also observed how the parameter ϵ impacted the distribution of keystone individuals. Furthermore, in comparison to a simplistic graph generated by random acting agents, our network's degree distribution was more consistently and closely related to the observed graph.

10 Acknowledgements

We would like to express our deep gratitude for Dr. Susan Perry, for her enthusiasm in our work and dedication to studying white faced capuchin monkeys. We would also like to thank Colin Chun, Evan Diaz, Spencer Wong, and Allen Yu for their support and camaraderie during our initial exploration of animal networks. Lastly our greatest thanks for the support of Linda Anegawa and Robert Rounthwaite whose help propelled this paper to new heights.

References

1. Crofoot, M., Rubenstein, D., Maiya, A., Berger-Wolf, T.: Aggression, grooming and group-level cooperation in white-faced capuchins (cebus capucinus): Insights from social networks. American journal of primatology **73**, 821–33 (08 2011). https://doi.org/10.1002/ajp.20959
2. Hoegh, A.B., van Manen, F.T., Haroldson, M.A.: Agent-based models for collective animal movement: Proximity-induced state switching. Journal of Agricultural, Biological and Environmental Statistics **26** (2021). https://doi.org/10.1007/s13253-021-00456-0
3. Jacobs, A.Z., Clauset, A.: A unified view of generative models for networks: models, methods, opportunities, and challenges (2014). https://doi.org/10.48550/ARXIV.1411.4070, https://arxiv.org/abs/1411.4070
4. Kajokaite, K.: Male dispersal decisions: An agent-based general model and suggested refinements for white-faced capuchin monkeys (cebus capucinus) (2014)
5. Perry, S., Godoy, I., Lammers, W.: The Lomas Barbudal Monkey Project: Two Decades of Research on Cebus capucinus, pp. 141–163 (11 2012). https://doi.org/10.1007/978-3-642-22514-7_7
6. Perry, S., Smolla, M.: Capuchin monkey rituals: an interdisciplinary study of form and function (02 2020). https://doi.org/10.1101/2020.02.21.958223
7. Perry, S.E., Baker, M., Fedigan, L.M., Gros-Louis, J., Jack, K., MacKinnon, K.C., Manson, J.H., Panger, M.A., Pyle, K., Rose, L.M.: Social conventions in wild white-faced capuchin monkeys. Current Anthropology **44**, 241 – 268 (2003). https://doi.org/10.1086/345825
8. Song, W., Sheng, W., Li, D., Wu, C., Ma, J.: Modeling complex networks based on deep reinforcement learning. Frontiers in Physics **9** (2022). https://doi.org/10.3389/fphy.2021.822581, https://www.frontiersin.org/articles/10.3389/fphy.2021.822581
9. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction, pp. 135–181. The MIT Press, second edn. (11 2018)
10. Wilson, A.D.M., Krause, J.: Personality and social network analysis in animals. In: Animal Social Networks. Oxford University Press (01 2015). https://doi.org/10.1093/acprof:oso/9780199679041.003.0006, https://doi.org/10.1093/acprof:oso/9780199679041.003.0006