Methods for Evolution of Agent Language

Sivaraman Rajaganapathy, Nabil Khan
Wednesday 23rd November, 2022

I. Motivation and Goal

It is widely understood that communication and language have a direct correlation. In the natural environment animals with complex social behavior exhibiting intelligence also show forms of communication. As an example, dolphins have been known to coordinate their hunt for fish and chimpanzees can alert others of predators in the vicinity. Both contribute to the survival of the pack involved in the communication and the species at large. Humans would likely have evolved from a species that had no coherent vocabulary, syntax, or structure to one with a complex and rich language system to communicate complex and abstract ideas to each other. Understanding the environment conditions and rules that promote language development could help in:

- Understanding the evolution of natural languages
- Understanding the correlation between language complexity and intelligence
- Developing techniques for automatic generation of agent to agent languages
- Improving the efficiency of existing agent to agent languages

The main goal of this project is to find and create the environmental conditions that promote the evolution of a language between agents *ab initio*. We hope that our simulations will prove that a set of objectives in a suitable environment can induce the evolution of communication between agents. We also attempt to understand if the complexity of the environment contributes to the acquisition of language. Our initial intuition is given the right environment and a basic framework for communication, we would see communication evolve between agents in a multi-agent environment.

II. State of the Art

Cangelosi describes in [1], [2], different types of models to evaluate how language and communication evolves. The book poses the question of how language originated with the assumption that there was no language in the beginning. Evolutionary biology and neuroscience establish that there is both a genetic and a neural basis for the evolution of language. The simulations described in the book are way to establish and understand the origins of language using empirical evidence. This experimental approach is used in [3], where a 1 bit language is evolved by agents when trying to meet a global objective.

A recent and relevant research paper on the topic of learning a communication protocol is [4]. Two approaches for learning covered in the paper are Reinforced Inter-agent learning (RIAL) and Differentiable Inter-agent Learning (DIAL) and the goal is to maximize a shared utility. The paper is motivated by some of the fundamental questions of language learning and the contributing role of the environment for language to evolve. Communication between agents is limited to a narrow bandwidth

during execution, whereas learning is centralized, with no restrictions on communication bandwidth. RIAL uses deep Q-Reinforcement learning with two variations. One is an independent approach where each agent learns the neural network parameters independently while treating other agents as part of the environment. In the second variant, training occurs over a single network with the same neural network parameters shared among all agents. DIAL allows agents to send real valued messages to each other during the centralized learning phase making the agents fully trainable among themselves.

Another paper that contributed directly on the topic is [5] which applies reinforcement learning on simulations of a predator/prey model. Agents learn communication codes or signals (bit strings) to more efficiently hunt prey on a simulated grid world. The concept of reinforcement learning is briefly introduced in the paper and the Q-function is used to choose the appropriate rule from a set of rules. That is, given a state and an action, how effective is the action towards reaching its goal from the given state. A set of the Q-functions essentially define policy. The agent eventually learns the policy that maximizes its utility and this is accomplished by updating its set of rules to be more optimal (improve Q-value). A team of two hunters have the goal of surrounding a prey and simulations are run with variable lengths of the bit strings including zero length (no communication). The paper effectively shows that communication invariably improves performance of the hunters.

In [6] a genetic algorithm approach is used to evolve communication in the predator-prey model. A few benchmark fixed strategies are developed such as follower and herder, where no communication takes place between the predator agents. The genetic algorithm is used to evolve predator strategy and then compared against the fixed strategies. A binary chromosome represents behavior of an individual predator. Fitness is evaluated for each of the evolved strategies. A multi-point crossover increases likelihood of behavior modification in successive generation. Fitness is evaluated on the predator strategy of each successive generation (starting with a randomly selected generation). Then a new population is created by selecting and pairing the fittest individuals. Communication takes place simultaneously over a message board where symbols are written to and read from. The length of the communication string is varied and the genetic algorithm is used here again to evolve communication.

III. METHODOLOGY

A. *Simulation Environments*

We have have used python for our project. The python Mesa package [7] provides simulation functionality in a multi-agent environment and has predefined grids, scheduling, and agent models. We also added obstacles and targets to the existing Mesa framework for our simulation and implemented visualization tools using python packages instead of using Mesa's web based visualization package to use with our reinforcement learning algorithm.

We have created two grid based simulation worlds - the Bee World, an infinite horizon game and the Bridge World, a finite horizon game. These are described in the following sub-sections.

*1) **Bee World**:* The Bee World is a $n \times n$ toroidal grid, and consists of agents that can move up, down, left and right. This is shown in Figure 1. The agents are initially placed in a random location on the grid and have a target they have to reach and an obstacle (a hornet) just next to the target that they must avoid. Each step in any direction costs them a penalty of -5. Reaching the target results in a

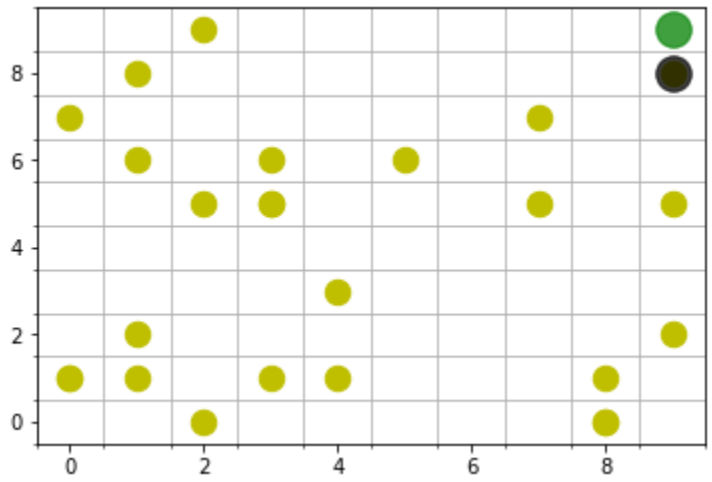


Figure 1: The Bee World. The yellow dots represent the bees who must continuously move with colliding with each other and while avoiding the hornet agents shown in black.

positive reward of 100. If they fall into the cell with the hornet they incur a penalty of -100. The agents are free to move in a any of the four directions but they are not allowed to stay in their current location. Multiple agents are also allowed to occupy the same cell. All agents are activated simultaneously, that is, at each time step they all move in a chosen direction. If they simultaneously move to the same cell, collisions occurs and they incur a penalty of -10. However, they are allowed to continue occupying the same space. When all the agents reach the target the simulation comes to an end. The simulation can also end by reaching the hornet. The simulation is however terminated after 100 time steps to avoid infinite loops particularly in random simulations where the probability of all agents reaching the target is nearly zero. All the agents are given the same objective to move in the direction of the target while avoiding collisions with neighbouring bee agents. This is implemented by the way of a utility function given by

$$U(t) = \mathbb{1}_m u_m + \mathbb{1}_c u_c + \mathbb{1}_h u_h + \mathbb{1}_t u_t,$$

where $\mathbb{1}_m$, $\mathbb{1}_c$, $\mathbb{1}_h$, and $\mathbb{1}_t$ are the indicator functions for movement, collision with other bees, with the hornets, and reaching the target at time t respectively. Further, u_m, u_c, u_h, and u_t are the utilities for movement, collisions, and reaching the target and are set to -5, -10, -100, and 100 respectively.

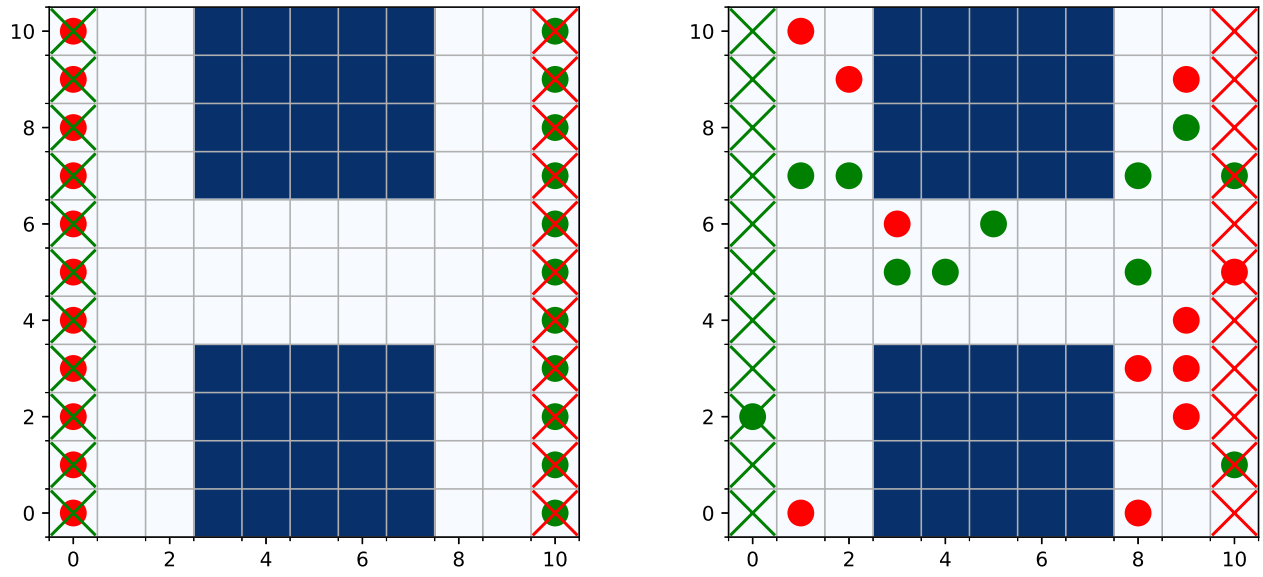
The bee world emulates a swarm or a cluster where each member has to exhibit some group behaviour while avoiding collisions. In figure 1 the yellow dots represent the bees and the black dots represent hornets (obstacles) that the bees should avoid. The actions are deterministic and any desired action is achieved with certainty. The environment is fully observable since each agent has information about the entire grid.

2) ***Bridge World****:* The Bridge World is an $n \times n$ planar grid, and consists of agents that also move up, down, left, right, or stay put. The agents in the bridge world are not allowed to occupy the same cell. The world further consists of two regions connected by a narrow bridge as shown in Figure 2. The agents are initially placed in each of the two regions and they have the objective to travel to the opposite region, via the narrow bridge. If the agents collide with any of the walls they stay put and incur a penalty. They also incur a penalty with each step. All agents step simultaneously with each time

step. Collisions with neighbours occur when two agents attempt to step into the same cell. Both agents stay put but incur a penalty. This is a finite horizon game where the game ends and restarts if all the agents reach their respective zones (unless the agents get stuck in an infinite loop, but this is avoided by using a watchdog timer). The utility function is designed as

$$U(t) = \mathbb{1}_m u_m + \mathbb{1}_c u_c + \mathbb{1}_t u_t,$$

where the symbols retain the same meaning as before. The utilities u_m, u_c, and u_t have been set to -5, -5, and 100 respectively. The Bridge World has been created to emulate a swarm with differing objectives with resource constraints (in this case the space on the bridge). The actions in the bridge world are deterministic in the sense that when an action is chosen and executed, there is no uncertainty in moving to the desired location except in case of the presence of obstacles in the direction of the desired movement. The environment is partially observable to the agents, that is, they only have information about the cells that are in their immediate vicinity of radius equal to 1 cell width (a total of 8 cells around them).



(a) (b)

Figure 2: The Bridge World. (a) Two sets of agents are started off in two sides of the world. They all need to reach the opposite end, via the 3 lane bridge connection. The dark region is inaccessible. (b) The bridge world after some random steps.

B. Multi-Agent Training

We started with training a single agent for each of the simulated environments, Bee World and Bridge World. To simplify our approach we started with no communication built in, so as to tune the training parameters that will suffice for our environments. We started with training with the entire grid and established that convergence on the training data was achievable using partial observability. We established that a densely connected neural network with 2 hidden layers of roughly 30 relu function neurons provides a good balance of complexity and performance for our environments. The neural model takes the observed states as an input along with the target location and the location of the agent being

controlled. The output consists of the action space. The reinforcement learning algorithm we used for our training is a Q-Learning approach with linear annealing of the randomness. This method initially explores the space in random directions, and gradually the probability of a random action is reduced to near 0 as the exploration progresses. After the initial exploration phase, it starts the training phase in batches and continuously updates the neural network weights to satisfy the Bellman equation

$$Q_t(s_t, a_t) = r_t + \gamma \arg\max_{a_{t+1}} Q_{t+1}(s_{t+1}, a_{t+1}).$$

Here $Q_t(s_t, a_t)$ is the discounted expected future reward, with a discount factor of γ, given the state s_t and action a_t at time t. We used the Q-values and loss to establish convergence through the training process.

For multi-agent learning, we used the centralized learning and decentralized execution approach suggested in [4]. This meant that all the agents shared the same network weights, however they were deployed to reach the target independently receiving independent rewards and states. For training, the number of agents were increased from 2, 3, 5 to 25 agents to achieve multi-agent training. In the Bee World training was performed with random initial positions for the agents to maximize initial exploration. Although the single agent successfully found the most optimal path to the target on every test trial after initial training, in the multi-agent world we noticed that the training became inadequate as we increased the number of agents. The targets were reached with 3 agents and 5 agents, however, for 25 agents the last 3 agents were still exploring the space. This is because the state space grew larger with the number of agents and the training was not sufficient for exploring a sufficiently large portion of the state space. The initial positions of the Bridge world are fixed and training with single as well as multiple agents successfully accomplished the task of reach the opposite. However, the network did not extrapolate well on scenarios with an unseen number of agents.

In a third phase we added communication functionality in the Bridge World where agents had the ability to share their intended direction of movement with agents in the vicinity (agents who they are likely to collide in the next time step). Two identical networks were used for the actions and communication actions. The input space of the neural networks were expanded for the additional information now available in the form of communication received from neighboring agents. The training involved the same reinforcement algorithm with training in batch from observations we received from each agent with each movement. The only difference was that the communication actions were not simultaneous, but sequential with each agent assigned a randomized priority order for communication in each round. Thus, in each round, each of the agents takes a decision on where to move and what to say, in a random sequence, based on not just the states, but also on what their neighbours are saying. During the training phase of the communication enabled Bridge agents the Q-values we observed were much lower than anticipated and at times negative. Q-values during the training phase have been observed to be a low number which increases as the training progresses. The Bee World also exhibited similarly low values during the training phase. With the additional states and agents it is possible that the exploration phase does not necessarily lead to discovery of the target which ultimately results in the correct Q-value.

IV. Results & Discussion

A. *Evaluation Method*

Our evaluation criteria for convergence during the training of the data were the Q-values and the loss which generally showed a reverse trend during the training phase. That is Q-values increased and the loss from the batch trained decreased. However, when observing this trend in larger grids with an increasing number of agents (25+) we noticed that the Q-values and losses were less helpful confirming convergence because the values fluctuated more sporadically. During training of the Bee World we ensured that the initial placement of the agents were randomized so the RL algorithm would not just trace out a fixed path for all the agents. This was particularly necessary in large grids where even random walks do not guarantee coverage of the entire grid space.

The primary metric we used to evaluate our results during the test phase (following training) is to look at the overall reward. Since the goal of the agent is to maximize expected utility and this is achieved by the reinforcement learning algorithm with convergence on the Bellman equation, this is the most direct way to prove successful training. We ran simulations using the trained weights and compared it against simulation runs with random actions. In order to compare the effect of communication we also ran simulations with and without communication. Besides the reward, we also looked at the number of collisions because communication is expected to reduce this number. Since the actual communication process do not incur a cost but collision has a significant penalty, our RL algorithm would minimize the number of collisions. We also performed several simulation runs to observe the behavior of the agents particularly with fewer agents. In addition to agent to agent collisions in the bridge world we also looked at the collisions that occur with obstacles and the wall. Since there are no wall in the Bee World this was not relevant in the Bee World.

B. *Results*

For the Bridge World, it was found that training with up to 2-4 agents lead to successful convergence of the neural network weights in reasonable time. Extrapolation of the network to more agents during test performed poorly, but the degradation was graceful and not drastic. Figure 3 shows the average rewards obtained by an agent with and without communication compared with a random agent. The baselines have been shifted to move the mean of the random agent rewards to 0. Both the neural network models (with and without communication) were trained with 4 agents in the training phase. As can be seen from the graphs, the performance reduces with increasing number agents, because it takes longer for each to them to reach the destination through the crowd. Furthermore, the Q-learning approach seems to memorize some of the routes and thus shows further degradation in the performance. The agents with communication enabled seem to do marginally better. The improvement in non-existent for a small number of agents due to the fact that there are fewer chances of two agents encountering each other. For larger number of agents (>10), the training with communication does substantially poorer. This is likely due to the agents not knowing what to do when surrounded by multiple agents.

It was difficult to discern if the agents had actually learned to communicate in a meaningful way. Therefore, we have shown the average number of collisions of each of the agent in Figure 4. It can be seen that with communication, the number of agent-agent collisions is lesser than random and nearly the same as the network without communication. However, the collisions with walls and obstacles

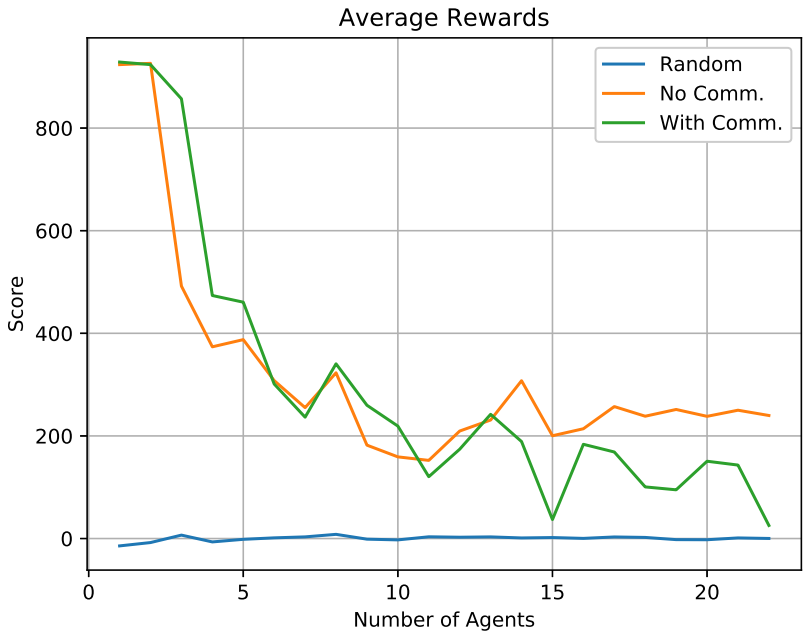


Figure 3: Average Rewards in the Bridge world over 5 runs of 100 frames each.

is considerably increased with communication. Our belief is that this is an indication that complex communication is likely not evolved. Based on the marginal improvement in performance, we could at best conclude that some primitive communication has evolved.

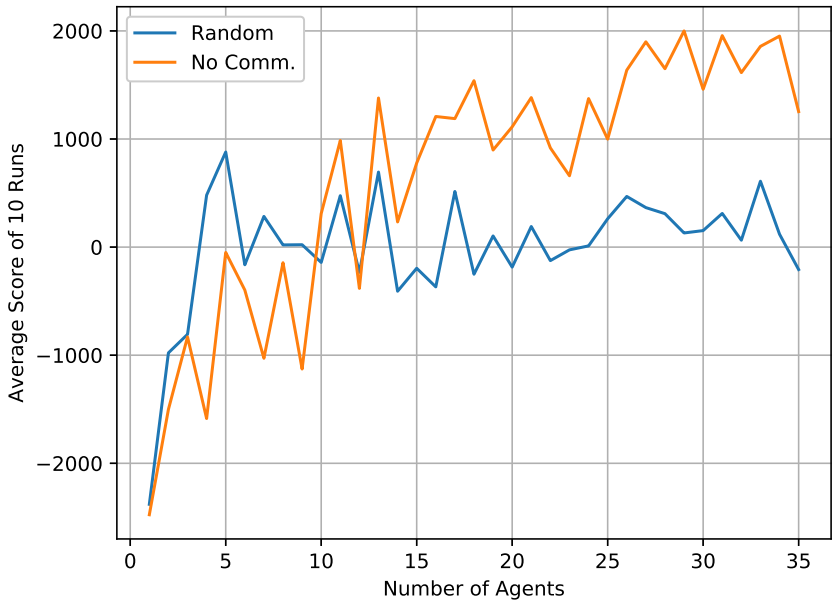
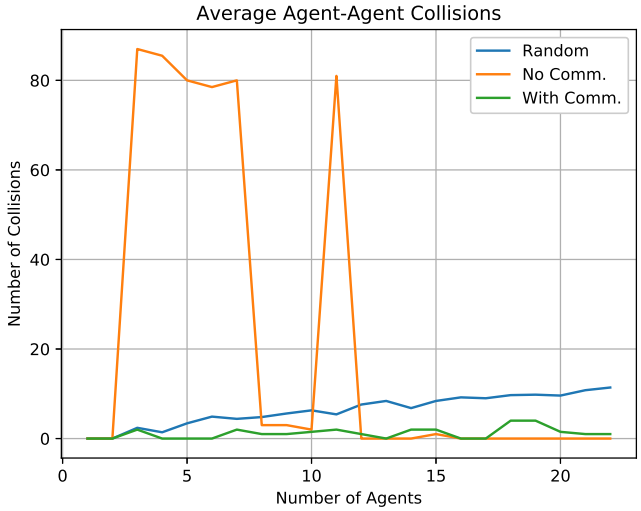
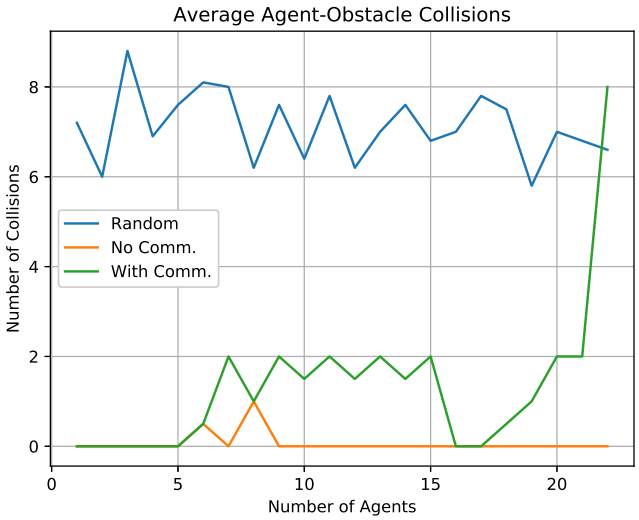


Figure 5: Average Rewards in the Bee world over 5 runs of 100 frames each.

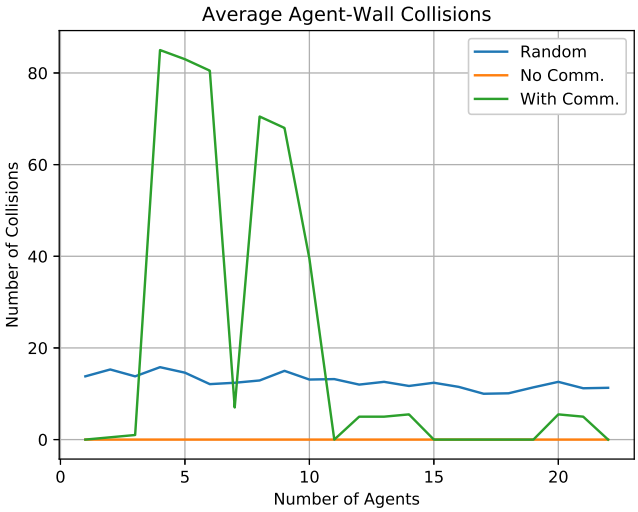
Figure 5 shows the test results for the Bee World, compared with random actions. The model used



(a) Agent to Agent Collisions



(b) Agent to Obstacle Collisions



(c) Agent to Wall Collisions

Figure 4: Average number of collisions in the Bridge World over 5 runs of 100 frames (a) Agent to Agent. (b) Agent to Obstacle. (c) Agent to Wall.

was trained with 25 agents, so as expected we see a higher performance around the 25 agent mark and above. Since we are using a model-free RL approach a large variation on the number of agents leads to sub-optimal performance and the data needs to be retrained at approximate intervals of about 5 agent margins. The communication option in the Bee world was left out to limit the scope of the project because training the Bee World even without proved to be extremely challenging. Due to the random nature of the path planning in the Bee world we had to rely on expanding the state space to the full extent of the grid already requiring a complex neural network. Training such a network demands additional processing power and training time that can be better handled as part of future work.

V. Conclusions & Future Work

Both the Bee World and Bridge World provided a non-trivial multi-agent environment where the agent had to learn an optimal path and avoid colliding with other agents. The size of the grid and the number of agents provided introduced additional complexity in the environment to explore the role communication played for the agents to accomplish their task. However, communication seems to have a detrimental effect on agent performance. In the first phase of the communication attempt we had simultaneous action and communication occurring at each time step. This model performance much worse than agents with no communication because it complicated the model without adding any usable information for the agents to use. With modification to this model we a communication action was followed by the move action we observed slightly better performance but still below what we achieved with no communication. We can only theorize that a more complex neural network, along with longer training periods is needed for handling the expanded state space. Since the neural network and our reinforcement algorithm successfully trained our agents to avoid collision but failed to achieve this objective when more agents were introduced, we can conclude that the network is not being adequately trained. The same phenomenon is probably occurring with communication. However, this will be left as future work to establish the cause for the diminished performance with communication.

Since communication is a two way street where both sending and receiving communication is a critical phase of the overall process, there is value in exploring each phase separately where we send the intended direction of movement and observe if agents can decide an optimal path with the information available. The reverse direction is also worth investigating where we establish that agents can already utilize the information from agents but agents learn to send relevant information. We also need to explore more efficient reinforcement learning strategies compared to the one employed in this project, so that more complex models can be explored and trained in reasonable time. We also wish to explore model based reinforcement learning, since the we did not see good performance when extrapolating to varying numbers of agents.

REFERENCES

[1] A. Cangelosi and D. Parisi, *Simulating the evolution of language*. Springer Science & Business Media, 2012.

[2] A. Cangelosi, "Evolution of communication and language using signals, symbols, and words," *IEEE Transactions on Evolutionary Computation*, vol. 5, no. 2, pp. 93–101, 2001.

[3] D. Floreano, S. Mitri, S. Magnenat, and L. Keller, "Evolutionary conditions for the emergence of communication in robots," *Current biology*, vol. 17, no. 6, pp. 514–519, 2007.

[4] J. Foerster, Y. M. Assael, N. de Freitas, and S. Whiteson, "Learning to communicate with deep multi-agent reinforcement learning," in *Advances in Neural Information Processing Systems*, 2016, pp. 2137–2145.

[5] T. Kasai, H. Tenmoto, and A. Kamiya, "Learning of communication codes in multi-agent reinforcement learning problem," in *Soft Computing in Industrial Applications, 2008. SMCia'08. IEEE Conference on*. IEEE, 2008, pp. 1–6.

[6] C. L. Giles and K.-C. Jim, "Learning communication for multi-agent systems," in *WRAC*. Springer, 2002, pp. 377–392.

[7] "Mesa," http://mesa.readthedocs.io/en/latest/tutorials/intro_tutorial.html.

APPENDIX

Our working codes can be found at:

- https://github.umn.edu/khan0096/IntelligentAgentsProject
- https://github.umn.edu/rajag038/IAProject

The model used for the data shown in this report is given in Figure 6.

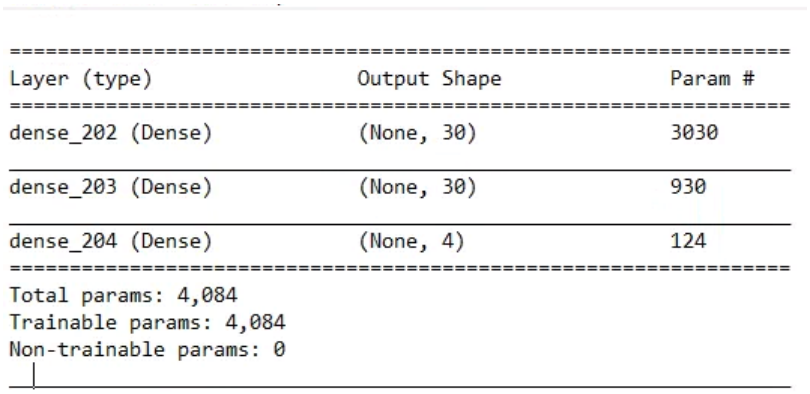


Figure 6: Neural Network Model Summary