Imitation Learning for Social Simulation

Justin Downes(✉) and Hamdi Kavak

George Mason University, Fairfax, VA 22030, USA
jdownes4@gmu.edu

Abstract. Modeling the behavior of complex agents is challenging. In social systems, agents have different motivating factors and goals that drive each decision. In some situations, though, we can observe the agent behavior and the outcomes while being at a loss on how to quantify the decision-making process. Imitation learning is a powerful tool to learn behavior without understanding reasoning. In this paper, we explore modern machine learning techniques to train models that imitate agents' behavior in social environments. This work continues and builds off of emerging work that merges social simulation and modern machine learning. We have shown that such surrogate models can learn heuristically-driven agent behavior. We note that, however, these models do show fragility to changes in environmental dynamics.

Keywords: Agent-based modeling · Imitation learning · Computer vision · Social science · Surrogate models · Direct policy learning

1 Introduction

As social simulations get increasingly complex, it becomes more and more challenging to enumerate all of the parameters and dynamics we want to explore. With agent-based models, this includes how the environment is structured, the agent's interactions with that environment, and the heuristics that drive agent interactions. This leads to a couple of challenges for social scientists to overcome. First, when modeling agent behavior from scratch, how do we know we are crafting the correct parameters to simulate the target of interest? And second, how do we prevent our biases from being introduced to guide the agent's behavior towards the target behavior? These complications are not new, and they are not unique to the social sciences.

In this paper, machine learning is used to learn the underlying behaviors of social agents in order to replicate an observed social dynamic. The use of learning models that can operate in complex environments is a well-worn path in the areas of self-driving cars [16], robotics [17], and flying [5], among others. While many of these tasks replicate human behavior, the goal, regardless of the human's actions, is measurable. For instance, safely navigating from point A to point B is a measurable task even if a human does not make the decisions. It is a bit different for the social sciences since we are seeking to replicate human behavior as much as possible to model and predict actions they will take. Therefore, this line of research aims not to achieve some optimal performance for a task that humans currently do but to replicate how humans do a particular task.

Much of the current work in this area focuses on replicating the model's overall behavior [3,6], but not necessarily with an emphasis on an individual agent's specific behavior. Efforts replicating agent behaviors tend to focus on data mining techniques or hand-crafted methods to identify features to learn on [4,12]. Additional work in this area has used machine learning models as a surrogate in order to perform sensitivity analysis [9,20]. The contribution of this work is that it is meant to explore how to learn agent behavior solely through observation while evaluating the model through the imitation agents' collective behavior's ability to replicate the observed model.

Agent based models are ideal for solutions based in Reinforcement Learning (RL) techniques and its method of learning through long term rewards seems perfect for the social sciences. Yet the lack of an a priori reward function and the limited ability to observe social dynamics often precludes this method. Imitation learning techniques such as behavior cloning offer similar solutions that work nicely with these constraints and there is ongoing debate as to when each method should be used [14]. Behavior cloning's method of learning from historical examples that have been recorded allows the model to directly learn the policy that drives the agent's actions [5,8] and is the method we have chosen for this set of experiments.

2 Case Study: Schelling's Model of Segregation

We aim to choose a case study to test out mechanisms and complications that may arise from learning social dynamics. To this end, a model was chosen such that the underlying driver of agent behavior was known a priori. In practical use, if the underlying behavior were already known, then one would not need to construct a model to learn it. This type of solution would be more suitable for situations when one could observe complex behavior in dynamic environments. Situations such as evacuation [15] and traffic [21] simulations would be ideal for ABMs that have learned agent behavior through imitation. Since this experiment is about understanding the techniques that can be useful in this type of problem, a predefined model of agent behavior as the target is sufficient.

The specific social simulation chosen for this paper to imitate was Schelling's model of segregation [18]. The behavior of self-segregating agents is easy to measure both in individual agent decision-making as well as a final measure of outcome. The Schelling model, as originally specified, had agents moving randomly around a grid world until some segregation threshold was achieved in that agent's field of view and its type. This model demonstrated how even slight desires for segregation, i.e. the agent's threshold, can lead to systemic segregation in that world. A complication of the Schelling model is that the agent's moves are random, this means that the decision space for an agent is the entire grid world and the determination of that decision is random. This can introduce unnecessary complications when training models and does not reflect the intended goal of learning underlying decision making motivations.

The Sert *et al.* [19] variation of the Schelling model modifies the agent behavior such that it is modeled as a reinforcement learning problem that has been trained to behave in a way that maximizes future segregation of the system. The individual decisions that an agent makes are no longer driven by rules as the Schelling model but instead learned through this goal maximization process. This reinforcement learning modification provides two advantages for our problem. First, it makes an agent's decisions tractable and based on the environment instead of randomly moving. And second, it defines the goal-seeking that drives the agent's behavior, which can be used to define those behaviors in this experiment explicitly. With these modifications, an environment can be constructed where a model can learn to imitate an agent's actions in a dynamic environment.

3 Methods

In order to conduct this experiment we created two separate models. The heuristic model which sought to replicate the results of Schelling's model [18] by utilizing some of the advancement's made in Sert *et al.* [19], and the imitation model, which is a machine learning model that learns to imitate the actions that those agents take in different segregation environments.

3.1 The Grid World

The grid environment used in this paper is a replica of the Sert *et al.* [19] environment. It is a 50×50 grid with wrapping in both horizontal and vertical directions. There are two types of agents that are randomly placed in the environment targeting a desired population ratio of each type to number of cells (default of 5%). Each agent has an observable window size of 11×11 cells, with the agent at the center of that window, as illustrated in Fig. 1.

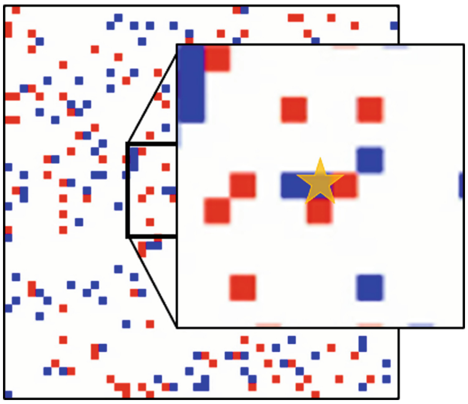


Fig. 1. The grid world with focus on the visible window for an agent.

The metric of how much an environment is segregated is calculated through a multi-scale entropy formulation [19], which calculates Shannon's entropy at multiple scales for 2 agent types in a window. It works by first calculating the entropy (see eq. (1)), for each patch E_n of size n, in this experiment 6×6, 12×12, and 25×25, where A is the number of type A agents and B is the number of type B agents in that patch. The final segregation score for an environment S_e is 1 minus the average of all of the patch entropy scores (see eq.(2)).

$$E_n = -\left(\frac{A}{A+B} log_2 \frac{A}{A+B} + \frac{B}{A+B} log_2 \frac{B}{A+B}\right) \tag{1}$$

$$S_e = 1 - \frac{1}{|N|} \sum_{n \in N} E_n \tag{2}$$

3.2 The Heuristic Model

This model acts as the baseline agent behavior for which the imitation model will learn from. Agents in this model move in a direction towards the maximum segregation patch in their visual window, with the expectation that the emerging segregated state of agent positions is in line with the original Schelling and Sert models.

A model can choose to move in direction d from a choice of directions D which is defined as a vector of x and y step sizes from a choice of three possible steps $\{-1, 0, 1\}$, or simply as the permutation of all 2 choices from that set of 3 movement steps $\{-1, 0, 1\}$, $D = P^r(\{-1, 0, 1\}, 2)$. This gives us 8 directions of movement and 1 choice to stay still, for a total decision space of 9 for each agent. Given an agent a_i, its optimal movement decision d_{opt} is defined as the maximum segregation score $s(w_i, d)$ for a window $w_{i,d}$ in a given direction $d \in D$ from an agent a_i's coordinates a_{xi}, a_{yi}, given compactly here.

$$d_{opt} = \underset{d \in D}{max}\ s(w(a_i, d)),\ \text{where } D = P^r(\{-1, 0, 1\}, 2) \tag{3}$$

The window $w_{i,d}$ for an agent a_i in direction d is simply the x and y coordinate vector of the agent's location added to the direction d vector multiplied by a look distance l_{dist}, and then getting the bounds of that window by adding and subtracting the observation window's dimensions, $[o_{width/2}, o_{height/2}]$, to get the upper left and lower right coordinates of the window (4). While the movement vector d has a magnitude of 1 the look distance l_{dist} magnitude is 3, this is done so that the segregation values for patches is more cleanly separated and, by moving 1 cell at a time, the agent can move more smoothly. In this experiment the look distance is set to 3 and the observation window is 5.

$$w(a_i, d) = w_{i,d} = [a_{xi}, a_{yi}] + l_{dist}[d_{xi}, d_{yi}] \pm [o_{width/2}, o_{height/2}] \tag{4}$$

The score for a window $w_{i,d}$ is simply $1 -$ *the entropy score*, which is described in (1). Since the entropy score of a patch is defined as the negative entropy it is simplified here in (5) as $1 +$ *the entropy score*.

$$s(w_{i,d}) = 1 + \left[\frac{A}{A+B} log_2 \frac{A}{A+B} + \frac{B}{A+B} log_2 \frac{B}{A+B} \right] \tag{5}$$

For each step this model calculates the segregation score for all agents in all directions. The simulation may stop at some fixed step count or until some threshold ratio of non-moving agents to moving agents is reached. In practice, both methods were used depending on whether the imitation model was being trained (threshold stopping was used) or evaluation was being conducted (fixed length run time was used).

3.3 The Imitation Model

The imitation model seeks to replicate the heuristic agent's decisions by observing an agent's action given its current state (observation window). The selected model architecture was a simple convolutional neural network (CNN) trained in a supervised fashion, shown in Fig. 2. It consists of multiple convolutional layers, topped with dense, fully connected layers and a classification layer that determines the direction for the agent to move. Trainable layers utilized Rectified Linear Unit (ReLU) activations with the final classification using a Softmax activation. The model used Adam optimization [13] and categorical cross-entropy as its loss.

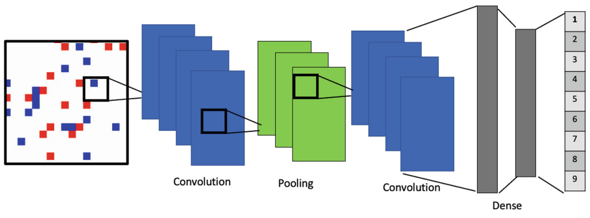
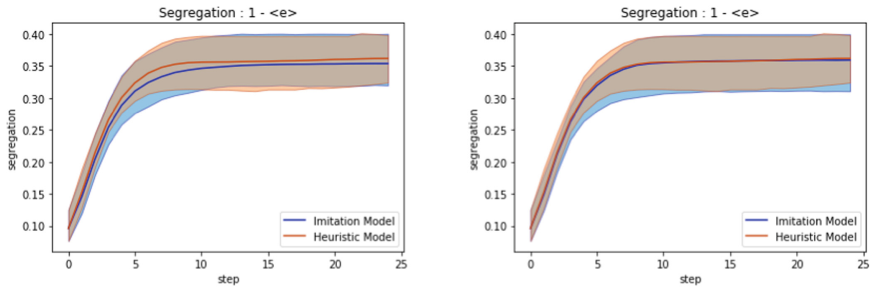


Fig. 2. The CNN model architecture.

We trained the model using two different methods. A static dataset of 100 simulations was generated from which a model was trained from in different batches. A separate model was trained off of simulations that were created on demand. This dynamically trained model was trained for a total of 50k simulation steps. Each input into the model is a single agent's viewable window, where the label is the direction of movement as determined by the heuristic model. Batches are sampled by shuffling multiple agent's windows and actions from across the available scenarios. All simulations had an early stopping mechanism once an entropy threshold was met (default .15). This early stopping mechanism was implemented to yield better agent action distribution, as at higher entropy levels most agents remain still. The static dataset of 100 simulations equates to roughly 6k simulations steps, yeilding approximately 10x difference between the static and dynamic dataset sizes. These two different methods of training were developed to simulate the different real world scenarios where you may have limited observed data due to safety constraints or other complications of generating observations and scenarios where you could realistically have an abundance of observed data, e.g. the stock market. The goal is to identify performance degradations between these different scenarios.

4 Results

The imitation model was evaluated on how well it was able to replicate the segregation score of the heuristic model at each step in a simulation run. Simulation runs were capped at 25 steps since at this point all observed simulations had ceased to have any significant movement. For evaluation, 10 simulations were run with both the statically and dynamically trained model, where each of these simulations had the same starting state. They were each compared to the heuristic model which also started in the same environment state.

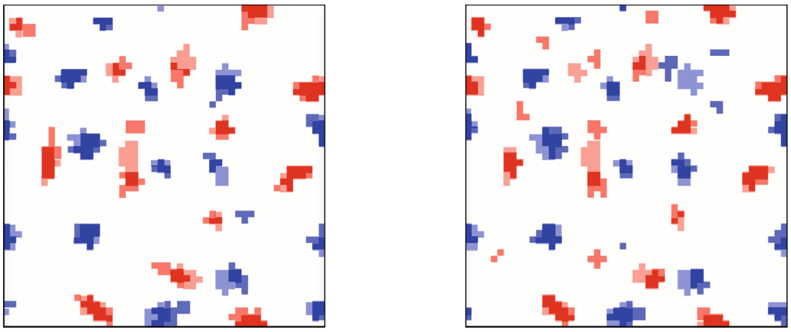


(a) Model trained on static dataset. (b) Model trained on dynamic dataset.

Fig. 3. Mean segregation score for imitation models and heuristic model, with min and max bands. Run on the same 10 starting environments for 25 steps. The segregation score is 1 - the hierarchical entropy score $<e>$, described in Eq. 2.

As can be seen in Fig. 3, both models were able to achieve roughly the same mean growth of segregation scores with similar min/max bands for the same simulations. This means that the imitation model is segregating generally at the same pace as the heuristic says it should for a given starting environment and ends with approximately the same metric. The displayed bands are not error bands but show the entire range of segregation scores for a given step during the 10 runs. The comparison should be in how well the mean line and border of the bands line up.

Another question to ask though is whether the end states look the same. If one expects for each agent's observable world the imitation model can make the same decision most of the time, then the end state should look approximately similar between the heuristic and imitation models. Figure 4 shows the end state of each imitation model overlaid with the end state of the heuristic model from the same starting state, broken down into the end state for the model trained on a static dataset in Fig. 4a and the dynamically trained model in Fig. 4b.



(a) Model trained on static dataset. (b) Model trained on dynamic dataset.

Fig. 4. Overlay of end states from same starting states compared between imitation models and heuristic model. Darker areas are where both models agreed.

The stochastic nature of the evolution of segregation states makes quantitative analysis a constant battle of sample size. Ten simulations were chosen due to computation timeliness constraints but did capture consistent metrics during the multiple runs.

4.1 Impact of Agent Density

One reason to train an agent to imitate another agent is to run simulations where the environment is too novel, or agent behavior is too difficult to enumerate. For instance, in evacuation scenarios where one wants to test new floor plans and would like to predict behavior from a set of real observations. To set up that type of situation with these models, new environments were constructed that had higher densities of agents than what had been observed and therefore trained on. For the heuristic model, this change in density is not a problem, as decisions are deterministic and can be anticipated and corrected. For the learned agents, though, this may demonstrate a weakness in the model where it is not robust to the environment changes one would expect and desire from a social model. In this experiment, fragility in the learned model is determined by deviation from the expected behavior of the heuristic model.

These experiments were run at increasing agent densities starting at 5%, which was the agent density for the preceding experiments, going up to 45% with steps of 2.5%. Of note is that this density percentage is for each agent type, and since there are two types of agents, the actual agent density is two times the displayed percentage. These experiments also followed the format of the other experiments in that the same ten starting states were used for all models, but each set of 10 starting states would be different for each density setting.

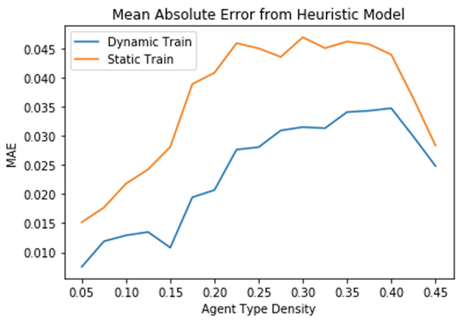


Fig. 5. Plot of Mean Absolute Error in segregation scores between dynamic and static model compared to the heuristic model at different agent densities. Since there are two agent types the overall density is approximately twice the displayed agent density.

As can be seen in Fig. 5, we start to see an increase in error of segregation scores as agent density increases. This may be an artifact of the stochastic states, but the trend is fairly clear that the model starts to deviate from expected behavior. We do also see a reversion to expected behavior as the density approaches maximum. Intuitively this appears to show a weakness in the model to replicate decisions where the environment starts to have many similar options, i.e. it easy when it's not dense because the attractive options are obvious and its easy when it's very dense as it can be best to stay still. This implies that even though creating agent models through machine learning can seem to simplify the features we want to model, one still must be aware of the biases they are introducing. In future simulations with learning based agents, holding out certain environmental dynamics may be warranted.

5 Discussion and Conclusion

This research investigated a method of learning an agent's behavior through observation. While we knew the correct choice an agent should make at each step, as defined by the target model, the model's behavior was evaluated on its holistic ability to replicate model metrics instead of individual agent decisions. The target agent in this scenario was driven heuristically, but in practical applications would be agents behaving in real-world situations under observation. These situations would be such that manually defining their behavior would be prohibitive to developing accurate models. We can unlock simulations of highly complex behaviors and environments by developing systems that can learn agent behavior. Also, by utilizing machine learning models as surrogates for heuristic agents, we can simplify computation, easily expand model sensitivity analysis, and create agents that are reusable across different environments.

The method implemented here utilized a computer vision model that was trained in a supervised fashion. It was able to know the agent's observable state and which action was taken. This simplified the model architecture that was required and the training mechanism utilized. There are other methods of imitation learning that can be explored for this exact type of problem [11,17], which may be more robust to more complex environments. Many problems exist where the state is not entirely observable, or the action taken at each step may not be known. Methods such as Reinforcement Learning (RL) [1] and Generative Adversarial Networks (GANs) [10] have been developed to work in just these types of scenarios. When the action taken at each step is noisy or missing, the end goal or a periodic reward can be used as a substitute for supervision. Future research in using RL to learn social agent behavior looks promising as it is already widely used to learn how to take actions in many environments [7], but still generally requires hand crafted rewards which this method we present operates through observation alone. Along with traditional RL, Inverse Reinforcement Learning could be especially useful for these types of experiments as it may allow for more ease in transportability for agent behavior to new environments [2] as the reward function can be more abstract than the visual features.

References

1. Abbeel, P., Ng, A.Y.: Apprenticeship learning via inverse reinforcement learning. In: Proceedings of the Twenty-first International Conference on Machine Learning. ACM Press (2004)
2. Adams, S., Cody, T., Beling, P.A.: A survey of inverse reinforcement learning. Artif. Intell. Rev. **55**(6), 4307–4346 (2022). https://doi.org/10.1007/s10462-021-10108-x
3. Angione, C., Silverman, E., Yaneske, E.: Using machine learning as a surrogate model for agent-based simulations. PLoS ONE **17**(2), 1–24 (2022). https://doi.org/10.1371/journal.pone.0263150, https://doi.org/10.1371/journal.pone.0263150, publisher: Public Library of Science
4. Bell, D., Mgbemena, C.: Data-driven agent-based exploration of customer behavior. SIMULATION **94**(3), 195–212 (2018). https://doi.org/10.1177/0037549717743106
5. Bratko, I., Urbančič, T., Sammut, C.: Behavioural cloning: phenomena, results and problems. IFAC Proc. Vol. **28**(21), 143–149 (1995). https://doi.org/10.1016/S1474-6670(17)46716-4, https://www.sciencedirect.com/science/article/pii/S1474667017467164, 5th IFAC Symposium on Automated Systems Based on Human Skill (Joint Design of Technology and Organisation), Berlin, Germany, 26-28 September
6. ten Broeke, G., van Voorn, G., Ligtenberg, A., Molenaar, J.: The use of surrogate models to analyse agent-based models. J. Artif. Soc. Soc. Simul. **24**(2), 3 (2021). https://doi.org/10.18564/jasss.4530, http://jasss.soc.surrey.ac.uk/24/2/3.html
7. Charpentier, A., Élie, R., Remlinger, C.: Reinforcement learning in economics and finance. Comput. Econ. (2021). https://doi.org/10.1007/s10614-021-10119-4
8. Chemali, J., Lazaric, A.: Direct policy iteration with demonstrations. In: Twenty-Fourth International Joint Conference on Artificial Intelligence (2015)

9. Edali, M., Yücel, G.: Automated analysis of regularities between model parameters and output using support vector regression in conjunction with decision trees. J. Artif. Soc. Soc. Simul. **21**(4), 1 (2018). https://doi.org/10.18564/jasss.3786. http://jasss.soc.surrey.ac.uk/21/4/1.html
10. Ho, J., Ermon, S.: Generative adversarial imitation learning. In: NeurIPS (2016)
11. Hussein, A., Gaber, M.M., Elyan, E., Jayne, C.: Imitation learning: a survey of learning methods. ACM Comput. Surv. **50**(2) (2017). https://doi.org/10.1145/3054912
12. Kavak, H., Padilla, J.J., Lynch, C.J., Diallo, S.Y.: Big data, agents, and machine learning: towards a data-driven agent-based modeling approach. In: Proceedings of the Annual Simulation Symposium. ANSS 2018, San Diego, CA, USA. Society for Computer Simulation International (2018)
13. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2014). https://doi.org/10.48550/ARXIV.1412.6980, https://arxiv.org/abs/1412.6980
14. Kumar, A., Hong, J., Singh, A., Levine, S.: When should we prefer offline reinforcement learning over behavioral cloning? In: Proceedings of the International Conference Learning Representations (2022)
15. Liu, Q., Lu, L., Zhang, Y., Hu, M.: Modeling the dynamics of pedestrian evacuation in a complex environment. Phys. A: Stat. Mech. Appl. **585**, 126426 (2022). https://doi.org/10.1016/j.physa.2021.126426, https://www.sciencedirect.com/science/article/pii/S0378437121006993
16. Muller, U., Ben, J., Cosatto, E., Flepp, B., Cun, Y.: Off-road obstacle avoidance through end-to-end learning. In: Weiss, Y., Schölkopf, B., Platt, J. (eds.) Advances in Neural Information Processing Systems, vol. 18. MIT Press (2005). https://proceedings.neurips.cc/paper/2005/file/fdf1bc5669e8ff5ba45d02fded729feb-Paper.pdf
17. Osa, T., Pajarinen, J., Neumann, G., Bagnell, J.A., Abbeel, P., Peters, J.: An algorithmic perspective on imitation learning. Found. Trends Robot. **7**(1-2), 1–179 (2018). https://doi.org/10.1561/2300000053, https://ieeexplore.ieee.org/document/8620668
18. Schelling, T.C.: Dynamic models of segregation. J. Math. Sociol. **1**(2), 143–186 (1971)
19. Sert, E., Bar-Yam, Y., Morales, A.J.: Segregation dynamics with reinforcement learning and agent based modeling. Sci. Rep. **10**, 11771 (2020)
20. van Strien, M.J., Huber, S.H., Anderies, J.M., Grêt-Regamey, A.: Resilience in social-ecological systems: identifying stable and unstable equilibria with agent-based models. Ecol. Soc. **24**(2) (2019). https://doi.org/10.5751/ES-10899-240208, https://www.ecologyandsociety.org/vol24/iss2/art8/, publisher: The Resilience Alliance
21. Zhao, B., Kumar, K., Casey, G., Soga, K.: Agent-Based Model (ABM) for city-scale traffic simulation: a case study on San Francisco, pp. 203–212 (2019). https://doi.org/10.1680/icsic.64669.203, https://www.icevirtuallibrary.com/doi/abs/10.1680/icsic.64669.203