A Computational Model for Human Perception of Communicativeness

Annie Chen and Julian Jara-Ettinger

Yale University Computer Science Department

(Dated: May 2, 2018)

Abstract

Social reasoning, thinking about what other people are doing, requires an agent to make sense of their behaviors and predict future actions by inferring their mental states. Most prior research in this area has focused on inferring external goals, such as collecting objects. But communication is another important part of social interaction, so machines must learn to recognize behaviors as communicative or non-communicative.

In this project, we build a computational framework to extend a previous psychology study on how people infer communicative intent from actions. Participants watched videos of 23 paths and rated how communicative they were. We turn those paths into lists of actions in a 2-D Gridworld. We model an agent’s mental states as a Markov Decision Process (MDP) applied to that GridWorld. This model uses an observed set of actions and calculates a likelihood. Then, we use Bayesian inference to compute the posterior probability of a Goal, conditioned on observed actions. Iteratively for every state in grid, a reward is placed in that state, and then the policy is constructed. Then, we calculate the product of the likelihood of taking that action in that state for every action in the path, and sum that for each reward, resulting in a number between 0 and 1 for each of the 23 paths.

We find this model does not completely capture human behavior. Thus, we build a second model with a communicative prior for each path. This model will have a perfect fit because we can fit a prior to each data point. Thus, we look for patterns in the priors. We find that they correlate strongly with the efficiency of the path, supporting previous research that people may believe more efficient movements have a lower communicative prior.
I. INTRODUCTION

As a human, making sense of other people’s behavior involves reasoning about their unobservable mental states such as beliefs and desires. Because one cannot see other people’s minds, to make sense of their behaviors and predict future actions, one must infer their mental states.

In recent years, researchers have developed human inspired artificial intelligence systems which are capable of inferring underlying intentions and desires based on observable actions. These systems rely on frameworks developed for robotics: Markov Decision Processes. In the study of robotics, researchers know what they want robots to accomplish, and their task is finding a set of actions to consistently reach that goal. To do so, researchers formalize the robot’s task using a reward function, which, combined with the actions that the robot can take and a model of the world (represented as a state transition matrix), produces the optimal policy—a function that maps each state to the action the robot ought to take.

Computational models of mental-state inference reverse this process, observing a set of actions and attempting to find the underlying goal. This is done by applying Bayesian inference to invert a Markov Decision Process model. That is, the task of inferring other people’s goals can be formalized as the problem of seeing a trace of the agent’s policy (the actions that they take), and finding a latent reward function that generates the observed policy. The models are then run on situations, and their outputs compared to human behavior in the same situation. In this way, it is possible to test cognitive theories, and better understand how humans make sense of the social and physical world.

Computational models using these ideas have been successfully applied to reach human-level performance in relatively simple tasks that involve inferring other people’s beliefs (Baker, Jara-Ettinger, Saxe, & Tenenbaum, 2017), and their desires (Velez-Ginorio et al., 2017; using a probabilistic context free grammar to express more complex reward functions with temporal and logical structure).

To date, most research in this area is focused on how we infer other people’s actions in terms of external goals, such as collecting objects. But people often act not to seek an object, but to communicate. If machines are to interact with and understand human behaviors, they must understand how humans recognize behaviors as communicative or non-communicative. In this paper, we focus on human perception of communicativeness.
Previous work studying how people perceive communicativeness has shown that people may use a gesture’s efficiency, repetitiveness, and possibility of external goal to judge whether a gesture is communicative. Gestures that are less efficient, more repetitive, or less likely to have an external goal are judged to be more communicative (Royka, Aboody & Jara-Ettinger 2018).

We expand upon that study, taking the framework and expanding it to create a model to reason about communicative goals. We seek a computational model that matches human behavior.

Here, we present two models that move meaningfully toward that goal. The first uses a Markov Decision Process to calculate the probability of a path being goal directed. We find that this model does not fully match humans. Thus, the second model uses the probability of a path being goal directed, but also incorporates a communicative prior.

A. Background: The Initial Experiment

The ground truth we aim to model is data from the first experiment on a psychology study on gestures, in which participants watched videos of dots moving on two dimensional planes and rated how communicative they were.

23 paths’ were constructed by combining 4 of 16 possible primitive path segments, which were a set of horizontal, vertical, diagonal, and 90 degree arc segments. Each path is named using a 4 letter string. Each letter refers to a the primitive path segment (see FIG. 1). These paths were then converted to a list of float coordinate points, and plotted to create the video.

![FIG. 1. 16 possible primitive path segments](image)
Participants were told that the videos represented someone on an island, who on some days needed to communicate with a helicopter using movements, and some days was just walking around on the island. They were asked to rate how likely they thought it was that the person was communicating something to the helicopter, on a scale from one (definitely not communicating) to seven (definitely communicating). The results over 30 participants were then averaged to get one floating point number from 1 to 7 per path. This list of 23 numbers is what we compare our model output to.

II. MODELS

A. Goal Inference Model (Model G)

We use a framework provided by previous work on understanding actions as inverse planning (Baker, C. L., Saxe, R., & Tenenbaum, J. B., 2009). We model an agent’s mental states as a Markov Decision Process (MDP) applied to a GridWorld. This model uses an observed set of actions and calculates a likelihood. Then, we use Bayesian inference to compute the posterior probability of a Goal, conditioned on observed actions.

1. Markov Decision Process

A process has the Markov property if transitions to new states depend only on the current state. We use Markov Decision Processes here to model how an agent should behave for maximum reward in an a given environment with the Markov Property. MDPs have been successfully used to capture how people infer goals (Baker, C. L., Saxe, R., & Tenenbaum, J. B., 2009).

We implement an MDP class in mdp.py. An MDP consists of:

- **S**: a list of states
- **A**: a list of possible actions
- **T**: a transition matrix containing the probability of reaching state \( s' \) when taking action \( a \) from state \( s \)
- **R**: a reward matrix containing the reward for taking action \( a \) in state \( s \)
• Gamma: the future discount parameter, which manipulates how risky the agent will be. Higher gamma values make less risky agents

• Tau: the softmax parameter, which controls how much noise the agent adds to the policy

We discuss below how S, A, T, and R are created in the GridWorld. Gamma is held constant at 0.95, and tau, the softmax parameter, is discussed in Adjustments.

2. GridWorld

To apply an MDP to the paths, we implement in gridworld_agent.py a GridWorld, a 2-dimensional grid that agents move around in, which contains:

• S: a list of every square in the grid and a dead state. When an agent reaches a reward, any action from there takes it to the dead state, and once it’s in there, it stays there forever. This is necessary to model rewards that disappear once they have been collected - without a dead state, a reward on the edge of the grid behaves differently than a reward in the middle of the grid, because the agent can bump into walls on the edge and return to the award.

• T: a transition matrix where taking action a from state s moves the agent to the logical next state (for instance, moving left from state i leads to state i − 1) with probability 1. If the current state has the reward in it, then the agent moves to the dead state with probability 1. If the move isn’t possible (for instance, if the agent is on the top edge and trying to move up), then it remains in its current state.

• A: a list of 8 actions to best imitate the paths that humans watched. Up, Down, Left, Right, Up Left, Up Right, Down Left, and Down Right.

• R: a list of rewards and their locations. For this model, we only allowed a reward 10 in one square

The GridWorldAgent class runs Value Iteration, an algorithm that solves for the optimal value function for each state. Then, it builds a policy using the results from value iteration, by picking the action that will lead to the state with the highest value.
In the abstract formalization of a MDP, this policy is deterministic, because there is a right action or set of right actions. Thus, the agent would take an inoptimal action with probability 0. However, this doesn’t work when modeling human behavior, because people make mistakes. Thus, we add a softmax parameter, which provides some noise and gives each action a small nonzero probability of being taken at each state.

3. Bayesian Inference

The Gridworld Agent takes goals and creates an action policy. Because our goal is to take an observed set of actions and find a likelihood that those actions are the result of a reward function, we formalize it as a Bayesian inference problem.

The posterior probability that the agent is pursuing a goal given their actions is given by the following, where \( a \) is a list of actions, and \( G \) is the event where the action is goal directed (a reward function leads to this set of actions):

\[
p(G|a) = \frac{p(a|G)P(G)}{p(a)}
\]

To calculate \( p(a|G) \) we need to integrate over the space of possible goals. If each goal is
determined by a reward function, then

\[ p(a|G) = \int_{r \in \mathbb{R}} p(a|r)p(r|G) \]  

If we assume that each goal’s reward function must have reward 0 in all states except in a target state, then we have a finite number of reward functions, equal to the number of states \(s\), making

\[ p(a|G) = \sum_{r \in \mathcal{R}} p(a|r)p(r|G) \]  

\[ p(G|a) = \frac{\sum_{r \in \mathcal{R}} p(a|r)p(r|G)P(G)}{p(a)} \]  

Human participants were told that there was a 50% chance that the agent is pursuing a goal, so \(p(G) = 1/2\). If we assume a uniform distribution over possible reward functions, then \(p(r|G) = 1/|S|\).

\[ p(G|a) = \frac{1}{2|S|} \sum_{r \in \mathcal{R}} \frac{p(a|r)}{p(a)} \]

where \(p(a)\) is the normalizing constant. \(p(a|r)\) is then determined by a rational planner:

\[ p(a|r) = \prod_{i=1}^{n} p(a_i|s_i; r). \]

Thus, the probability that the agent is pursuing a goal is:

\[ p(G|a) = \frac{1}{p(a)} \sum_{r \in \mathcal{R}} \frac{\prod_{i=1}^{n} p(a_i|s_i; r)}{2|S|} \]

This calculation is implemented in reward_guesser.py, where for each path, each list of path coordinates is turned into a list of actions. Then, a GridWorldAgent is created, and iteratively for every state in grid, a reward is placed in that state, and then the policy is constructed. Then, we calculate the product of the likelihood of taking that action in that state for every action in the path. Then, the product of the likelihood for each reward is summed, resulting in a number between 0 and 1 for each of the 23 paths.

4. Adjustments

There were several adjustments considered and implemented when running this model.

First, we considered how to convert the path coordinates from the video to a set of actions for the MDP. Due to limits in computing power, we were constrained to a 41x41 GridWorld, and had to map floats between 0 and 500 to 41x41. We tried 4 methods:
1. Simply dividing everything by 10, mapping the lowest number to 0, and the highest number to 1.

2. The same as above, but converting a sequence of a horizontal motion and a vertical motion into a diagonal (ex: (Up, Left) becomes (Up Left))

3. Normalizing each of the four segments that make up a path such that each traveled the same Euclidean distance

4. Normalizing each of the four segments such that each had the same number of actions in it.

We ran the model on each of these sets of paths, and found that option 2 had the highest correlation with what people said.

We also tested 4 different softmax values: 0.28, 0.4, 0.5 and 0.6. These values were chosen by testing different values on a small GridWorld and trying to find a value such that the best action had probability around 0.5. Because this property varied based on whether the state was on an edge, we ran the whole model 4 times, and found that a softmax value of 0.4 had the highest correlation with what people said.

B. Results for Model G

If people determine how communicative a gesture is by saying "the less goal directed it looks, the more communicative it is," then we expect a strong negative correlation between the model’s predictions and participant judgments. Upon first analysis, the Pearson correlation between Model G and people’s results, -0.39 is quite low. However, the Spearman correlation is -0.79, suggesting an issue with the scale of our model. When we apply a log transform to the model’s prediction, we find promising results.

C. Goal Inference + Communicative Model (model GC)

The need for a log transform for the results of Model G suggests that the Goal Inference model alone is relatively successful at ranking the communicativeness of gestures, but the scaling is off. Thus, in the second model, we explore adding understanding of the probability that an agent is communicating given their actions.
FIG. 3. Log transformed model predictions vs participant judgments. Points are labeled with their path name.

The posterior probability that the agent is communicating given their actions is given by:

$$p(C|a) = \frac{p(a|C)P(C)}{p(a)}$$

To calculate $p(a|C)$ we need to integrate over the space of possible gestures:

$$p(a|C) = \int_{C\pi} p(a|C\pi)p(C\pi|C)$$

where $C\pi$ is a communicative policy (a list of actions).

If we assume that communicative policies are always communicated perfectly, then $p(a|C\pi) = 1$ whenever $C\pi = a$ and 0 otherwise. This reduces the integral to a single point and we get

$$p(a|C) = p(a|C\pi = a)p(C\pi = a|C)$$

which becomes

$$p(a|C) = p(C\pi = a|C)$$ \hspace{1cm} (5)

Using this in the posterior we get:
\[ p(G|a) = \frac{p(C \pi = a|C) P(C)}{p(a)} \]

Human participants were told that there was a 50% chance that the agent is communicating. Instead of assuming what the prior is, we can define it as a parameter \( p_a \) (prior belief that actions \( a \) are communicative). With this, we get:

\[ p(C|a) = \frac{1}{p(a)} \frac{p_a}{2} \] (6)

Then, we can combine the probability that an agent is communicating given an action and the probability that an agent is pursuing a goal given an action. Thus, our model predictions are calculated as follows:

\[ p(C|a) = \frac{p_a}{p_a + \sum_{r \in R} p(a|r)/|S|} = \frac{|S|p_a}{|S|p_a + \sum_{r \in R} p(a|r)} \] (7)

We want to fit these predictions to participant data. The problem is that participants answered on a different scale. So we’d need to fit the data using some unknown linear transformation:

\[ y = \beta_0 + \beta_1 \frac{|S|p_a}{|S|p_a + \sum_{r \in R} p(a|r)} \] (8)

We have a set of participant judgments \( y_a \) and a prior \( p_a \) that each respective action sequence \( a \) looks communicative. We want to find the prior \( p_A \) that maximizes correlation between Eq. 7 and participant answers \( y \). (We do not use a constant prior because we believe that people have different priors for different motions. For instance, they may think curves are more communicative than straight lines)

Thus, we find a prior that leads to a perfect fit using the following equation:

\[ y_s = \frac{|S|p_a}{|S|p_a + \sum_{r \in R} p(a|r)} \]

\[ p_a = \frac{y \sum_{r \in R} p(a|r)}{|S| \cdot (1 - y)} \]

We multiply \( y_a \) by a constant to ensure that \( p_a \) falls between 0 and 1.
D. Results for model GC

Because we can fit a prior to each data point, this model has a perfect fit. Thus, we look for a pattern in the priors. There is evidence that people may believe more efficient movements have a lower communicative prior (Royka, A., Aboody, R. & Jara-Ettinger, J. (2018). When we plot the priors against the efficiency of the path, we find a moderate fit, and a Spearman correlation of 0.68. However, there are 6 paths of efficiency 0 (they return to exactly where they started), which cause complications with the Spearman correlation. When they are removed from the data, the Spearman correlation jumps to 0.88.

III. FURTHER EXPLORATION

Although the Goal Inference model somewhat accurately predicts human judgments, there are further factors in human judgments that our model does not capture. For instance, the paths ‘aaaa’, ‘aeaa’, and ‘eaea’ have very similar model predictions and efficiency scores, but humans rate them 2.5, 3, and 3.8, respectively (see Figure XXXX). There seems to be some additional sense of communicativeness, perhaps related to changes in direction, that our model currently does not capture.

Further exploration of patterns in the priors is needed. One other direction I’m currently pursuing is feature engineering: creating new features such as how many times the path...
switches from curved to straight, and using stepwise selection and linear regression to try to find significant features. Hopefully, this will lead to a more nuanced understanding of how humans naturally perceive and judge communicativeness.

IV. ACKNOWLEDGEMENTS

I’d like to thank Professor Jara-Ettinger for his kindness, guidance, and support, and Amanda Royka for her data, paper, and feedback.

Also, thanks to the Yale High Performance Computing Clusters for making all of this modeling possible!

V. REFERENCES


