Learning Social Preferences in Games

Ya'akov Gal
Division of Engineering and Applied Sciences
Harvard University
Cambridge, MA 02138
gal@eecs.harvard.edu

Avi Pfeffer
Division of Engineering and Applied Sciences
Harvard University
Cambridge, MA 02138
avi@eecs.harvard.edu

Francesca Marzo
Cognitive Science Department
University of Siena
Siena, 53100 Italy
marzo@eecs.harvard.edu

Barbara J. Grosz
Division of Engineering and Applied Sciences
Harvard University
Cambridge, MA 02138
grosz@eecs.harvard.edu

Abstract

This paper presents a machine-learning approach to modeling human behavior in one-shot games. It provides a framework for representing and reasoning about the social factors that affect people's play. The model predicts how a human player is likely to react to different actions of another player, and these predictions are used to determine the best possible strategy for that player. Data collection and evaluation of the model were performed on a negotiation game in which humans played against each other and against computer models playing various strategies. A computer player trained on human data outplayed Nash equilibrium and Nash bargaining computer players as well as humans. It also generalized to play people and game situations it had not seen before.

Introduction

Technology is opening up vast opportunities for computer agents to interact with people. In many of these settings, agents need to make decisions that affect the other participants. Many human-computer interactions may be modeled as games, in which human and computer players each have their own choices of actions and their own goals. In AI, game theory has often been used as a paradigm for modeling interaction in multi-agent systems; it prescribes rules-of-behavior for all agents in the game.

However, traditional game theory cannot naturally capture the diversity of human behavior. People play quite differently from game theoretic predictions and their play varies considerably (Kagel & Roth 1995). A multitude of social factors, such as altruism, selfishness and reciprocity have been shown to mediate people's play in games (Camerer 2003). It has proven difficult to derive analytically the extent to which these factors affect the play of different people. To design computer agents that interact successfully with humans and adapt to new situations, we must represent these factors explicitly in our model and be able to learn them.

In this paper, we use machine learning techniques to learn models of how people play games. The model learns a utility function that players use, in which social factors are explicitly represented and defined in terms of players' outcomes.

We explicitly model several types of social factors, ranging from pure self-interest to altruism and including a notion of fairness. The model assumes that people reason about the same types of social factors, but that individuals weigh them differently. The model learns the utility functions deployed by different types of people; it also allows individuals to reason about their preferences with some degree of error. Since each utility function provides a guideline for behavior, learning the utility functions is key to developing a model that can predict which action a player will take.

The learned models were incorporated into a computer agent that interacted with humans and used to predict how a human player would likely react to different actions of the computer player. These predictions were then used to determine the best possible strategy for the computer player.

We implemented our model in a two-player negotiation game that uses the Colored Trails framework, developed by Grosz and Kraus (2004), an environment in which each player has a goal and certain resources are needed to reach it. The players can trade resources, leading to interesting negotiation scenarios. We study a scenario in which one player proposes a trade to the other, who then has the opportunity to accept or reject. Our model learns to predict whether the second player accepts or rejects the offer. This model is then used by an agent taking the role of the first player to help it decide which offer to make.

We tested our model in experiments involving human subjects. In the first phase of the experiments, data consisting of human play was collected. This data was used to learn a model of human play. In the second phase, a "social" computer player using the learned model was compared against two other computer players and against the performance of human players themselves. The social computer player was able to perform better than all the other players. It exhibited a variety of interesting behavior, depending on the resources players started out with and their positions in the game. The model generalized well to play people and game situations it had not seen before.

Most of the work on learning in games (Fudenberg & Levine 1998) has been on learning the strategy of a particular opponent from repeated play against that opponent. For example, opponent modeling has been applied successfully

to poker (Davidson *et al.* 2000).

Common to the techniques in this approach is that they focus on learning and evolution within a single repeated game, in which the agent keeps playing against the same opponents. In contrast, our approach to learning has been to generalize from the behavior of some human players to the behavior of other players. We aim to develop agents that interact well with human agents, even those they have never seen before.

Work in AI with regard to learning utility functions within MDPs (Ng & Russell 2000) and probabilistic frameworks (Chajewska, Koller, & Ormoneit 2001) differs from ours in fundamental ways. First, they learn a utility function directly in terms of outcomes of the game. No external factors enter into the utility function, and there is no consideration of behavioral aspects in their model. In contrast, we learn utility functions that are influenced by the social preferences that hold between players. Second, they assume that players choose the pure strategy that maximizes their utility function. We allow for the possibility that players' actions will sometimes contradict their supposed utility function, enabling us to handle noise in people's decisions.

Behavioral Decision-Making

Social factors play a crucial part in players' reasoning, both in repeated and in one-shot interactions. For example, reciprocal behavior has been shown to appear in the ultimatum negotiation game (Guth, Schmittberger, & Schwarze 1982), contradicting the predictions of traditional game theoretic models. In this game, one player proposes a division of some amount of good to another player. If the second player accepts, the good is divided as proposed, otherwise both players get nothing. The unique sub-game perfect Nash equilibrium of the game is to offer the smallest amount possible to the other player, and for the other player to agree to the proposal. However, numerous experiments have shown that individuals behave differently; most offers consist of half of the goods, and most rejections occur for offers consisting of less than a quarter of the goods.

A growing body of research in behavioral economics is concerned with *social preference* models. These models assume that players consider others' outcomes, as well as their own, when they reason about the game. These models allow players to follow a "social" utility function, exogenous to the game description, in which factors such as the following are explicitly represented.

Self Interest A key motivation for players is to maximize their individual utility, as specified by the rules of the game. This is the sole motivation considered by classical game theory.

Social Welfare Players concerned with social welfare are interested in the welfare of the group as a whole, as well as their own utility. Such players are willing to sacrifice some individual utility to make others better off.

Inequity Aversion A player who cares about fairness is concerned that the outcome be as equal as possible for all players. Such a player is willing to sacrifice individual utility, or to decrease others' payoffs, in order to ensure a more balanced outcome.

Social preference models that depend on these factors have been formalized. Bolton (1991) provides a model that assumes people care about inequity-aversion, in addition to self-interest. Charness and Rabin (2002) propose a model in which players also care about reciprocity; one player is able to punish the other. Bolton and Ockenfels (2002) compare models that measure fairness in terms of relative payoff comparisons with models that use reciprocal measures. Lowenstein et.al (1989) compared several social utility forms and found that a utility function including terms for self-interest, as well as a separate term for positive and negative discrepancies between the parties' payoffs, matched data corresponding to one-shot dispute type negotiation.

Common to these approaches is that models were learned and evaluated in simple scenarios; subjects in the laboratory were allowed to choose between two options, each of them corresponding to a different social feature. These approaches did not attempt to model social preferences in a more complex setting, such as a negotiation game which includes many possible deals. Moreover, they did not explicitly model different types of people interacting together.

There has been no work to date on building a computer agent that learns social preferences through repeated observation of human play. If human behavior presents particular regularities such as described above, and if a model can capture different types of players, then it should be possible to identify the factors influencing people's behavior through the use of computational modeling. An agent based on such a model would be able to generalize to playing new situations and examples of interaction it had not seen before.

Colored Trails

Our study used the game Colored Trails (CT), designed by Grosz and Kraus. CT is played on a board of colored squares with a set of tiles in colors chosen from the same palette as the squares. One square is designated as the "goal square" and each player has a piece on the board, initially located in one of the non-goal squares. The players have a set of colored tiles. To move a piece into an adjacent square a player must turn in a chip of the same color as the square. Tiles may be exchanged by the players, and the conditions of exchange may be varied to model different decision-making situations.

A player's performance in CT is determined by a scoring function. This function may depend on many factors, such as the player's distance from the goal-square, the number of moves made, and the number of tiles the player possesses at the end of the game. In addition, a player's performance in the game can be made to depend on the performance of other players, by including the score of other players in her own scoring function.

For our study, we use a version of CT in which two players played on 4x4 boards with a palette consisting of 4 colors. Each player has full view of the board as well as the other player's tiles. At the beginning of the game, the two players are randomly placed at two locations on the CT board and

allocated four tiles at random, which could include any color in the palette. The distribution of tiles is designed such that it is likely that the game is "interesting". A game is considered to be interesting if (1) at least one of the players can reach the goal after trading with the other player; (2) it is not the case that both players can reach the goal without trading.

The scoring function for the players was set as follows.

- 150 points bonus for reaching the goal; otherwise, 50 points bonus.
- 10 points for each tile left in a player's possession.
- 15 points deducted for any square in the path between the player's final position and the goal-square. This path is computed by the Manhattan distance.

Note that 50 points were rewarded for players that did not reach the goal square. The idea here was to keep the scores from being negative: the maximum Manhattan distance of a player from the goal square on a 4×4 board game is 6. In the worst case, the player cannot move from this position because it does not possess the right chips. In this case, the player's score is $50 + (4 \times 10) - (6 \times 15) = 0$.

The parameters were chosen so that while getting to the goal is by far the most important component, if a player cannot get to the goal it is preferable to get as close to the goal as possible. Furthermore, a player's outcome is determined solely by her own performance.

In each game, each player is designated one of two roles, which determines the possible actions that are available during the game. One player is the *allocator*, the other player is the *deliberator*. The allocator is allowed to propose an offer for exchange of tiles to the deliberator. The deliberator can either accept or reject the allocator's offer. If the allocator does not make an offer, then both players are left with their initial allocation of tiles. The deliberator is not allowed to counter the allocator's offer with another proposal. The score that each player receives if no offer is made is identical to the score each player receives if the offer is rejected by the deliberator. We refer to this event as the *no negotiation alternative*. The score that each player receives if the offer is accepted by the deliberator is referred to as the *proposed outcome* score.

Under the conditions specified above, each game consists of a one-shot negotiation deal between the two players, and a deliberator's reply to the exchange proposed by the allocator completely determines the final outcome of the game.

Model Construction

This work aims to model human deliberators in the CT game. Our task is to predict whether a deliberator will accept a given proposal. The inputs to the model are NN_A and NN_D, the no-negotiation alternative scores for the allocator and deliberator, and PO_A and PO_D, the proposed outcome scores for the allocator and deliberator.

To develop the model, we introduce the following features, which represent possible social factors that might affect the deliberator for a given deal:

- Individual-benefit (IB) $\qquad PO_D - NN_D$
- Aggregate-utility (AU)

$$(PO_D + PO_A) - (NN_D + NN_A)$$

- Advantage-of-outcome (AO) $\qquad PO_D - PO_A$
- Advantage-of-trade (AT)

$$(PO_D - NN_D) - (PO_A - NN_A)$$

The features Individual-benefit and Aggregate-utility match the social preferences self-interest and social-welfare that have been shown to contribute to players' reasoning in the behavioral economics literature. We add features to distinguish between two types of inequality. Advantage-of-outcome measures inequality in the final outcomes. Advantage-of-trade measures inequality in the gains received from a trade. In both cases this means that the deliberator has a strict preferences to do better than the allocator.

Given any proposed exchange x, a particular deliberator's utility u is a weighted sum of features. The utility function is determined by the weights $\mathbf{w} =< w_{IB}, w_{AU}, w_{AO}, w_{AT} >$. The weights measure the relative importance of each of the social preferences to the deliberator. The utility function is normalized around 0, so a utility of 0 corresponds to indifference between accepting the proposal and rejecting it. We interpret the utility to be not only an indication of which decision to make, but also the degree to which one decision is preferred. Thus, accepting a proposal is more strongly preferred when the utility is a large positive number than a small positive number.

We could try to learn a single utility function representing the preferences of all people, but this would be unlikely to fit many deliberators well; it is not likely that all deliberators behave the same way. On the other hand, if we try to learn a separate utility function for each deliberator, we would need to play against the same deliberator for many rounds before we could learn about that person, and we would not be able to generalize from one deliberator to another. Therefore we assume that there are several *types* of deliberators, where each type has its own utility function. We use a mixture model over types, with a probability distribution $P(t)$ over the set of types. Each type t is associated with its own set of social preference weights $\mathbf{w}^t$, defining a utility function u^t.

In addition, while a utility function corresponds to a decision rule, we do not assume that deliberators implement the decision rules perfectly, for two reasons. First, the deliberators' play may be noisy; they may make mistakes and act capriciously. Therefore, we expect noise in the data that could not be explained if the deliberators played perfectly. Second, even if a deliberator falls under a certain type, it is unlikely that her utility function will exactly match that of the type. Rather, it may differ in small ways, which cause her to take decisions that are contrary to the type's utility function.

To capture the fact that a decision rule might be implemented noisily, we use a sigmoid function. We define the probability of acceptance for a particular exchange x by deliberator of type t to be

$$P(accept|x,t) = \frac{1}{1 + e^{-u^t(x)}}$$

In particular, the probability of acceptance converges to 1 as the utility becomes large and positive, and to 0 as the utility becomes large and negative. When the decision is less clear-cut, i.e. the utility is close to zero, the probability of acceptance is close to $\frac{1}{2}$, meaning "mistakes", or decisions that are contrary to the utility function, are more likely to happen.

Given that we have a model describing the deliberator's behavior, the next step is to incorporate this model into a computer agent that plays with humans. In our framework, the computer agent plays the allocator and a human is playing the deliberator. The goal of this step is to maximize the score of the allocator, by exploiting the model of how the deliberator acts. The strategy is to propose the deal that maximizes the expected utility to the allocator. The expected utility is the sum of the allocator's utility of the proposal times the probability the proposal is accepted, and the allocator's no-negotiation alternative score times the probability the proposal is rejected.

Given that we have a model describing the deliberator's behavior, the next step is to incorporate this model into a computer agent that plays with humans. The goal of this step is to maximize the score of the allocator, by exploiting the model of how the deliberator acts. The strategy is to propose the deal that maximizes the expected utility to the allocator. The expected utility is the sum of the allocator's utility of the proposal times the probability the proposal is accepted, and the allocator's no-negotiation alternative score times the probability the proposal is rejected.

We take the expectation of this sum with respect to all of the deliberator utility functions. Formally, let E be the set of all possible exchanges that are available to the allocator. Let T be the set of deliberator types. The computer model will choose exchange e s.t.

$$e = \operatorname*{argmax}_{e \in E} \sum_{t \in T} P(t) \cdot [P(accept|x,t) \cdot PO_A(x) + (1 - P(accept|x,t)) \cdot NN_A]$$

Learning

The goal of the learning task is to complete the model of the deliberators by estimating the parameters from collected data. Since we are learning a mixture model, the task is two-fold; we must learn the distribution $P(T)$ over deliberator types, and for each type $t \in T$, we must learn the feature weights $\mathbf{w} =< w_{IB}, w_{AU}, w_{AO}, w_{AT} >$, corresponding to the contribution of each social preference. To solve this problem, we interleaved two optimization procedures, a version of the EM algorithm (Dempster, Laird, & Rubin 1977) and the gradient descent technique, as described in the Results section. We began by placing an arbitrary distribution over deliberator types and setting the feature weights with particular parameter values. We varied the number of types and the initial feature weights for each type.

Each observation d consists of inputs $\mathbf{x}^d = (NN_A^d, NN_D^d, PO_A^d, PO_D^d)$, and response y^d, which equals 1 or 0 for "accept" or "reject". To speed up learning, the inputs are scaled to lie in the interval [-1,1], by setting -1 to be the smallest value of the feature and 1 to be the largest.

The deliberator type t^d is unobserved, but we can compute the probability of each type for each data case using the current parameter settings:

$$P(t^d|d) = \frac{1}{Z^d} P(y^d|t^d, x^d) \cdot P(t^d)$$

where Z^d is a normalizing factor. Note that $P(y^d|t^d, x^d)$ is the likelihood of the deliberator's response in the game at data point d, according to type t^d. This can be computed by plugging the social utility function u^t for the observed game into the sigmoid function.

Computing the new values of the $P(t)$ parameters is straightforward. We compute $E(N_t|D) = \sum_{d \in D} P(t^d \mid d)$ and normalize. The maximization for the feature weights for each type is more interesting. The model for each type is a simple sigmoid belief network (Neal 1992). (In fact, it is a particularly simple sigmoid belief network, because it has no hidden layers.) The maximum likelihood problem for such a network can be solved by gradient descent, using a delta rule.

Since these networks participate in a mixture model, each one makes a contribution to the final output of $P(t|d)$. Therefore the gradient for the model associated with type t is proportional to $P(t|d)$. In other words, the degree to which a training example can be used to learn the weights in a network is proportional to the probability that the network actually generated the data. In the delta rules for these networks, therefore, we multiply the learning rate by $P(t|d)$. We obtain the following delta rule for each feature j:

$$w_j^t = w_j^t + \alpha P(t|d) \sum_{d \in D} x_j^d (1 - P(y^d|t)).$$

The $1 - P(y^d|t)$ error term comes from assuming that if the network was a perfect predictor, it would have predicted the outcome y^d with probability 1. The difference between 1 and $P(y^d|t)$ is the degree to which the network is not a perfect predictor.

Experimental Setup

A total of 42 subjects participated in the experiment, 32 in the data-collection phase and 10 in the evaluation phase. Participants were given a 20 minute tutorial of the game, consisting of an explanation of the rules, the scoring function and a practice game.

Each subject was identified by a serial number, and was seated in front of the same terminal for the entire length of the experiment, which was composed of a number of rounds of Colored Trails. A central server was responsible for matching up the participants at each round and for keeping the total score for each subject in all of the rounds of the experiment. No subject was paired up with any other subject more than once in the same role capacity. Subjects could not observe the terminals of other subjects, and they were not told about the identity of their partner.

Participants were paid in a manner consistent with the scoring function in the game. For example, a score of 130

points gained in a round earned a $1.30 payment. We kept a running score for each subject, revealed at the end of the experiment.

The experiment consisted of two separate phases: data-collection and evaluation. In the data-collection study, 16 subjects played consecutive CT games against each other. Each subject played 12 CT rounds, making for a total of 96 games played. The initial settings (board layout, tile distribution, goal and starting point positions) were different in each game. For each round of the game, we recorded the board and tile settings, as well as the proposal made by the allocator, and the response of the deliberator. We ran two instances of the data-collection phase, each one with different subjects, collecting 192 games. The data obtained from the data-collection phase was then used to learn a model of human play.

The evaluation study consisted of two groups, each involving 5 human subjects and 3 computer players. The computer players, only playing allocators, were automatic agents capable of mapping any CT game position to some proposed exchange. Agent SP proposed the exchange with the highest expected utility, according to our learned social preferences model. Agent NE proposed the exchange corresponding to the Nash equilibrium strategy for the allocator. Agent NB proposed the exchange corresponding to the Nash bargaining strategy for the allocator. Nash (1971), who addressed the topic of bargaining, gave several axioms which every reasonable bargaining solution must follow. He then showed that the deal which maximized the product of the agents' utilities is the sole solution to the set of axioms. An interesting point is that the Nash bargaining solution is always Pareto optimal. In the context of CT, this means that if there is no deal in which both parties do better than not negotiating, this player will not propose an exchange.

At each round, eight concurrent games of CT were played in which members of the same group played each other. The set-up was as follows. One of the human subjects, designated as an allocator, played another human subject, designated as deliberator; each computer player, designated as an allocator, played another human subject, designated as deliberator. The game settings, including board layout, start and goal positions, and initial tile distributions, were the same for all of the games played by members of the same group. Therefore, at each round there were 4 matching CT games being played by the eight members of each group.

Participants were given the same instructions and tutorial as in the data-collection experiment. We expected that telling the participants they would be playing computer agents in some of the rounds would alter their behavior. Therefore, participants were led to assume that they were playing a human at each round.

As before, for each round of CT, we recorded the settings of each game, the proposals being offered by the allocators, and the deliberators' response. The first group played 7 games, and the second group played 14 games, for a total of 21 games, where each game was replicated 4 times with different allocators.

Results and discussion

We learned separate models for one, two and three possible types of deliberators, henceforth referred to as Model1, Model2 and Model3 respectively. We set prior parameter values for the models as follows. For all models, we used random initial values for the distribution over deliberator types. For Model1 we also set random values for the feature weights. For Model2 and Model3, we assigned each deliberator type with initial feature values that corresponded to different points in feature space, by "highlighting" some features and giving them significantly higher initial value than others. In Model2, one of the deliberator types highlighted advantage-of-outcome and advantage-of-trade, while the other highlighted aggregate-utility. In Model3, deliberator types highlighted advantage-of-outcome, aggregate-utility, and Advantage-of-trade separately.

We ran each model on the data from the data-collection phase. We obtained the following posterior parameter values for each model. Model1, which had a single deliberator type, learned feature weights $(7.00, 5.42, 0.40, 4.00)$ for features individual-benefit, aggregate-utility, advantage-of-outcome, advantage-of-trade, respectively. Model1 described a highly competitive deliberator who only cares about her own outcome and would hurt the other to do better. In Model2, the distribution over deliberator types was $(0.36, 0.63)$ and feature weights were $(3.00, 5.13, 4.61, 0.46)$ and $(3.13, 4.95, 0.47, 3.30)$ for each type respectively. Model2 described two partially altruistic deliberators. They both have high weights for social welfare, while still being competitive; one of the types cares more about advantage-of-outcome, and the other type cares more about advantage-of-trade. In Model3, the distribution over deliberator types assigned miniscule probability for the third type, and resembled Model2 in all other parameter values. We decided to use Model2 in the validation study.

The following table presents the results of the evaluation phase for each of the models used in the experiment.

Model	Total Reward	Proposals Accepted	Proposals Declined	No Offers
SP	2880	16	5	0
NE	2100	13	8	0
NB	2400	14	2	5
HU	2440	16	1	4

It lists the total monetary reward, the number of proposals accepted, the number of proposals rejected, and the number of times no offer was proposed. The computer allocator labeled NE always proposed the exchange that corresponded to the allocator's strategy in the (unique) sub-game perfect Nash equilibrium of each CT game. In essence, this resulted to offering the best exchange for the allocator, out of the set of all of the exchanges that are not worse off to the deliberator. As a consequence, many of the exchanges proposed by this agent were declined. We hypothesize this is because they were not judged as fair by the deliberator. This result closely follows the findings of behavioral game theory. The performance of NE was the worst of the four. The computer allocator labeled NB always proposed the exchange that corresponded to the allocator's strategy in the Nash Bargaining profile. This exchange consistently offered more to

the deliberator than the NE player did for the same game, when the board and tile distribution enabled it. Because NB tended to offer quite favorable deals to the deliberator, they were accepted more than the other computer players, provided that an offer was made.

The allocator labeled HU combines the monetary rewards for all of the human players. Human offers were almost always accepted, when they were made. The computer allocator that followed our expected utility model, labeled SP, achieved a significantly higher reward than NE and HU (T-test comparison for mean reward was $p < .05, p < .1$, respectively). It also had the highest number of accepted proposals, along with the allocations proposed by humans. Interestingly, our model proposed the same offer as the human proposal in 4 of the games, whereas the Nash equilibrium player did not match a human proposal in any game, and the Nash bargaining player matched human proposals in 2 games. This suggests that our computational model would be perceived to be relatively more reasonable, or "human like", by other people.

Next, we describe some interesting behavior displayed by our program. The NNA heading in the following tables represents the no-negotiation alternative situation. First, we show a round in which SP proposed an exchange which was accepted by an altruistic human deliberator. Interestingly, there was only one observation in which an altruistic deliberator agreed to such an exchange, yet it was used 4 times by the allocator in the evaluation phase. This behavior, of asking for a favor when the other player is much better off, was consistently accepted by the human player.

Model	Allocator Score	Deliberator Score
NNA	45	170
SP	70	150

A second example, in which the proposed outcome of the exchange proposed by NE, while beneficial for the deliberator, was lower than the exchange proposed by SP. The NE exchange was rejected, while the SP exchange was accepted. This seems to indicate that the responders in this game cared about the equality of outcomes. Note that in this exchange, the SP exchange and the exchange proposed by the human were equal.

Model	Allocator Score	Deliberator Score
NNA	75	150
SP	170	170
NE	180	160
NB	150	190
HU	170	170

Conclusion and Future Work

We have presented a computational framework for representing and learning a social preference model of people in one-shot games. The model successfully learns the factors that affect human play and can generalize to people and game situations that were not seen before. We plan to model additional social features, such as the regret that a player might feel from accepting one trade when a more preferable trade was available.

Our current model assumed that observations of play are independent of each-other. We plan to constrain multiple observations from the same subject to originate from the same type.

The CT framework provides a natural test-bed for learning in more complex scenarios, such as repeated games. In repeated games, additional factors are at work, including dynamics of reciprocity and punishment. We also plan to study games involving more than two allocators, where issues of competition between the allocators arise.

While we have focused on one particular game for practical reasons, the learned models we use are cast in terms of general social preferences and do not depend on the specific features of the game. Therefore the learned models should be immediately applicable to other games in which the same social preferences play a role. It will be interesting to see if the model learned for one type of negotiation game will generalize to another type of game.

For our performance measure, we have used the score obtained by our computational agent while playing against humans. One can also imagine setting the goal of trying to play as much like humans as possible. Our learned models would now be used to generate computer play, rather than simply as predictors of human reaction. With the approach described in this paper, it is conceivable that we could build a program to pass a limited Turing test in a negotiation game, in the spirit of the Caltech Turing Test competition[1].

Acknowledgments

This work was supported by NSF grant IIS-0222892 and NSF Career Award IIS-0091815. We thank Jacomo Corbo and the Harvard AI research group for valuable discussions and programming assistance.

References

Bolton, G., and Ockenfels, A. 2002. A stress test of fainess measures in models of social utility. Technical report, Max Plank Institute.

Bolton, G. 1991. A comparative model of bargaining. *American Economic Review* (81):1096–1136.

Camerer, C. 2003. *Behavioral Game Theory: Experiments in Strategic Interaction*. Princeton University Press.

Chajewska, U.; Koller, D.; and Ormoneit, D. 2001. Learning an agent's utility function by observing behavior. In *ICML'01*.

Charness, G., and Rabin, M. 2002. Understanding social preferences with simple tests. *Quarterly Journal of Economics* (117):817–869.

Davidson, A.; Billings, D.; Schaeffer, J.; and Szafron, D. 2000. Improved opponent modeling in poker. In *International Conference on Artificial Intelligence(ICAI'00)*.

Dempster, A.; Laird, N.; and Rubin, D. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society* 39(1).

[1] http://turing.ssel.caltech.edu/

Fudenberg, D., and Levine, D. K. 1998. *The Theory of Learning in Games*. MIT Press.

Grosz, B.; Kraus, S.; Talman, S.; and Stossel, B. 2004. The influence of social dependencies on decision-making. Initial investigations with a new game. In *AAMAS'04*.

Guth, W.; Schmittberger, R.; and Schwarze, B. 1982. An experimental analysis of ultimatum bargaining. *Journal of Economic Behavior and Organization* (3):367–388.

J.Nash. 1971. The bargaining problem. *Econometrica* 18:155–162.

Kagel, J., and Roth, A., eds. 1995. *The hanbook of experimental economics*. Princeton University Press.

Loewenstein, G.; Bazerman, M.; and Thompson, L. 1989. Social utility and decision making in interpersonal contexts. *Journal of Personality and Social psychology* (57)(3):426–441.

Neal, R. 1992. Connectionist learning of belief networks. *Artificial Intelligence* 56:71–113.

Ng, A., and Russell, S. 2000. Algorithms for inverse reinforcement learning. In *ICML'00*.