An Agent for Deception Detection in Discussion Based Environments

Amos Azaria
Dept. of Machine Learning
Carnegie Mellon University,
Pittsburgh, PA

Ariella Richardson
Dept. of Industrial Engineering
Lev Academic Center, Israel

Sarit Kraus
Dept. of Computer Science
Bar Ilan University, Israel

ABSTRACT

Extensive use of computerized forums and chat-rooms provides a modern venue for deception. We propose introducing an agent to assist in detecting and incriminating a deceptive participant. We designed a game, where deception in a text based discussion environment occurs. In this game several participants attempt to collectively detect a deceptive member. We compose an automated agent which participates in this game as a regular player. The goal of the agent is to detect the deceptive participant and alert other members, without raising suspicion itself. We use machine learning on the data collected from human players to design this agent. Extensive evaluation of our agent shows that it succeeds in raising the players collective success rate in catching the deceptive player.

Author Keywords

Deception Detection; Human Agent Interaction; Suspicion Evasion; Machine Learning

ACM Classification Keywords

I.2.m Computing Methodologies: ARTIFICIAL INTELLIGENCE—*Miscellaneous*

INTRODUCTION

Many activities in our everyday lives involve sharing opinions with peers through computer-mediated communication. People participate in forum discussions on important topics such as: how to raise their children, what medication they should use and how to improve their business. It would be nice to assume that all of the people participating in these discussions have a common, honest goal and that malicious participants are spotted by moderators. However, this is often not true. Pedophiles manage to infiltrate kids' chat rooms, commercial products are pushed in forums by dealers posing as regular users, and business forums are probably full of advice that actually assists their competitors. Computer-mediated communication has provided a modern venue for deception [30], and encouraged research on the extent of deception online [21]. For example, consider a salesperson that participates in a chat discussion on buying a specific car. The salesperson may try to promote the car, and may be confronted by other participants as being dishonest. The salesperson would likely deny being a salesperson, and may accuse other participants of being a salesperson trying to promote another car. It can be difficult to determine which of the participants is presenting an honest opinion, and which is motivated by an ulterior motive. Many chat environments are not monitored by an administrator, or loosely monitored. Sometimes the moderator does not wish to intervene (this could be for various reasons, for example if the administrator is only responsible for the infrastructure and not for the content). Furthermore, even in monitored environments a user (or a group of users) may have personal preferences to the type of moderation they require. We suggest introducing an agent to the chat environment, that participates in the chat as a regular user. We expect this agent to have three capabilities. First, the agent must be able to find the salesman. Second, the agent should convince other users that this participant is a salesman. Third, we expect our agent not to raise suspicion as being a salesman himself. Such an agent could decrease deception in this environment.

In his book "Telling Lies" [12], Ekman states that "it is not a simple matter to catch lies" (pg. 80). Ekman explains (pg. 82) that when people are dishonest, they choose their words with care, however they tend to give away the lie through body language and tone. This knowledge from Psychology has motivated studies on the automatic detection of lies based on video and audio data [8, 25, 15, 4]. However, in many web-based environments, in which a group forms a discussion, such as chat rooms and forums, there is no audio or visual information. Such environments are based solely on short text messages.

Inspired by this, we investigate scenarios where several participants attempt to collectively detect a deceptive member. We also study how and if they may be assisted by an autonomous agent. As a first step towards reducing deception by human subjects, in discussion based environments, we designed an agent. This agent participates in a discussion based game that involves discussion and deception. The agent joins a group of human players as a regular member, and has no enforcement capabilities on other players. The agent must detect the deceptive participant and alert other members of suspicious behavior while not raising suspicion itself.

We designed a game to deploy and evaluate our agent in a text-based discussion environment. In this game there are several credible participants and one dishonest participant (a pirate). In the first phase of the game the participants con-

CSCW '15, March 14 - 18 2015, Vancouver, BC, Canada

ACM 978-1-4503-2922-4/15/03$15.00
http://dx.doi.org/10.1145/2675133.2675137

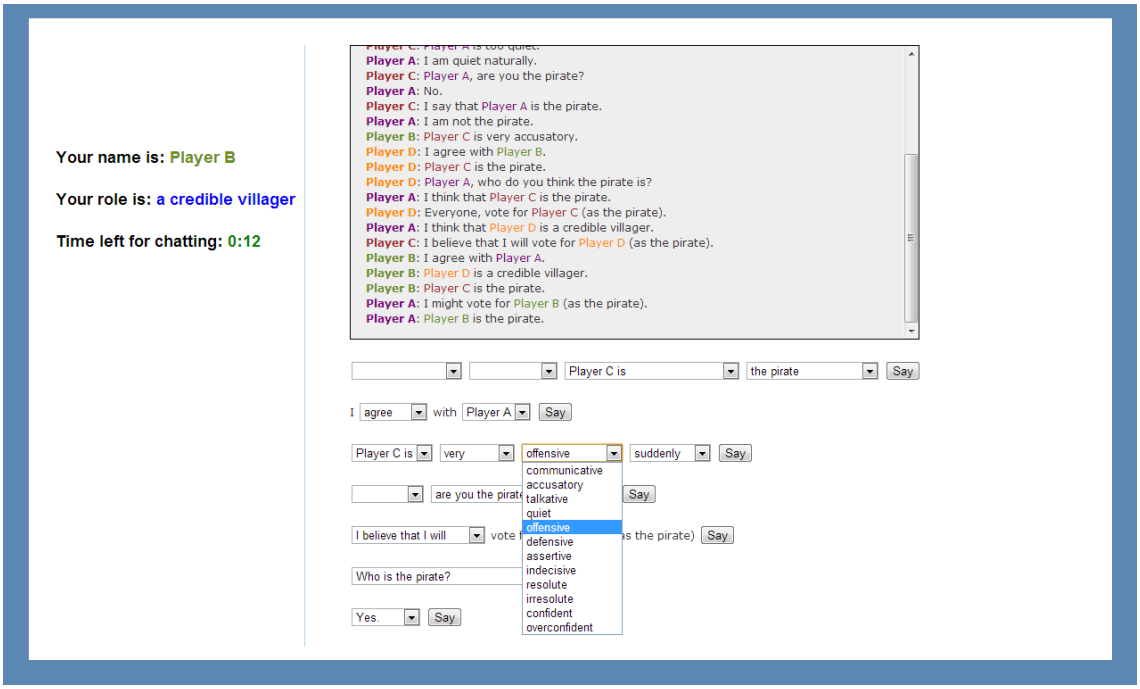

Figure 1. A screen-shot of the pirate game in progress.

duct a textual discussion in an attempt to uncover the liar, and later they cast their votes as to whom they think the liar is. The game is played either with a human set of participants, or with a mixture of human participants and an autonomous agent that plays as a regular participant. We evaluate the contribution of the agent to the credible participants in the following two different game versions: the first version of our game allows the pirate to be as active or inactive as he chooses and the second version encourages the pirate to be more active. The second version is similar to one where a salesman is pushing a commercial product. The credible participants performed no better than chance at spotting the pirate in both versions of the game. However, once one of the credible participants was replaced by the agent that we built, the group had significantly greater success in finding the pirate.

We use machine learning on the data collected from human participants to determine whether a player is honest or not. The agent uses this information to catch the pirate. We also apply machine learning methods in order to learn when players fall under suspicion. This information is also used by our agent in order to minimize the suspicion that it raises. We focus on the discussion dynamics such as tendencies towards accusation, denial or agreement.

In this paper we provide a platform for studying deception detection within a text based game that supports group discussion. In this game we deploy an agent that is required not only to detect deception and lead other participants to recognize the deceiver, but also to refrain from raising suspicion itself. Previous approaches use corpora to perform off-line deception detection, whereas we have perform on-line detection during multi-player interaction. To the best of our knowledge we are the first to deploy an autonomous agent in any such environment. We provide extensive evaluation of our agent in both game settings.

THE PIRATE GAME

In order to simulate the environment which we are interested in studying (deception in chat-rooms and forums), we need a game which will provide us with the following properties:

1. The game is played by a group of people.
2. The game uses text-based communication.
3. The game is based on a discussion, using short messages and in which players refer to one another.
4. The deceiver has some motivation to deceive.
5. The other players have some motivation to catch the deceiver.

To satisfy these properties we introduce the Pirate Game; a game, with four players and two roles. Three players are the honest players, the “credible villagers”, and the fourth player plays the deceptive participant, the “pirate”. All players are informed of their own role but not of anyone else’s. The participants are told that they are a group of villagers who went on a journey to find a treasure. They have found a treasure of coins and can split it. However, one of the participants is a pirate and can steal the coins unless he is detected. In order to detect the pirate a discussion phase is held. After the discussion, all credible villagers cast votes as to whom they think is the pirate (or an empty vote if they wish). In our game we do not allow the pirate to vote, as allowing the pirate to vote would make the game harder and less fun for the players and demotivate them. The votes are concealed until all players

cast their votes. If there is a majority of votes for the pirate he is "caught" and the money is split between the credible players. Otherwise the pirate receives all the money. At the beginning of the game each player is told his role and the discussion phase begins. The discussion is composed of structured sentences (examples are presented in Figure 1). Although we do not use free chat, our frame-work allows the composition of approximately $4,000$ sentences (which is, for instance, much richer than the options available in [11] or what can be achieved using icons as in [2]). A video demonstrating the user interface can be found at: http://azariaa.com/pirate.avi.

We use structured sentences rather than free text for several reasons. First of all, structured sentences, as opposed to open text, enable focusing on the dynamics of the discussion, and less on the syntax. Furthermore, the structured sentences encourage participants to use meaningful sentences, and they also encourage players to speak and be active because the player merely selects a sentence rather than phrasing his own. Since the players have limited time, slow typists may not have time to respond. Structured sentences limit the effect of typing speed (since this environment is new to everyone). Finally, the controlled environment also makes it easy to add an agent that plays similarly to other players and keeps the agent inconspicuous. This does not mean that an agent cannot be deployed into an unstructured environment, but adding "social credentials" to the agent were not the focus of this work. An open chat environment requires solving NLP challenges in order to automatically tag sentences. These are difficult challenges, but recent progress in NLP seems promising [28]. Once these challenges are solved it would be straightforward to adapt our agent to incorporate the NLP tagging.

We also implemented a second variation of the game. This variation is different only for the scenario where the pirate manages to escape (does not receive a majority of the votes). In this variation the pirate is told that one of the credible villagers will turn him over to the village ruler, if he escapes with the money. This results in neither the pirate nor the other players receiving any money, unless the pirate manages to convince the other players to cast at least one vote against the villager. In this case the villager will be considered unreliable (to the village ruler) and the pirate will gain all of the coins, that he escaped with. This setting encourages the pirate to be active in the game. We call this version of the game the "informer version" and differentiate it from the "basic version". A description of the formal model for the Pirate game can be found in the Appendix.

RELATED WORK

Of all existing games, the "Pirate Game" we introduce is most similar to the Mafia games (also known as "werewolf"), however pirate game is a simpler game which is closer to actual situations which take place in real life. In the mafia game once a player is accused to be a malicious player (or the werewolf") by a majority of votes, this player is eliminated from the game and the game continues. The mafia game also includes a phase in which the malicious player (or the werewolf") eliminates a credible player. Furthermore, every time a player is eliminated his real role is revealed. This elimination process receives considerable attention in studies which examine this game. However, in real-life situations there is very seldom a situation where some users are eliminated and the discussion continues without them. We therefore introduce the pirate game which we believe resembles real-life situations much better. The mafia game has been used as a platform for studying communication, coordination and functionality in situations where participants have different amounts of information, and have also raised much interest in theoretical analysis [18, 6]. Dias et al. [11] introduce MIXER (Moderating Interactions for Cross-Cultural Empathic Relationships), which is a platform incorporating intelligent and interactive characters and is based on the mafia game. They study the level of human reasoning (e.g. if someone wanted to eliminate a player which turned out to be a credible villager he may be the werewolf), and the level of reasoning which is required for an agent in order to play well. Aylett et al. [2] use the MIXER platform for studying the attitude children adopt towards cultural differences. They had children play the mafia game using two different sets of rules and observed how a child playing in one set of rules reacts when exposed to a group playing the game under a different set of rules. The mafia game has also been used for detecting deceptive participants using audio and video data [8, 25]. Chittaranjan and Hung [8] use non-verbal audio cues such as: the length of speaking intervals, the number of speaking turns, interruptions and pitch. They achieve an F-Measure of 0.62 for deception detection. These cues are used to detect suspicious behavior. They also show that the degree of speaker activity influences the group decision. In a later work, Raiman, Hung and Englebienne [25] combine these audio features with low quality visual data, in which facial expressions cannot be recognized, but gestures can. Combining the audio data with the video results in an improved detection rate with an F-measure of 0.76.

A similar study was performed by [15] who use digital audio tape to detect deceptive speech. They use both prosodic features (which do not use the actual words) such as pitch, loudness, duration, etc. and lexical features such as denials and flags for positive and negative emotion words (a total of 20 features). The best combination resulted with an accuracy of 64.4%. Our experiment is run under the assumption that the conversation is performed over the web where visual and audio data are unavailable.

Zhou et al. [30], use a text-based environment in which deceptive senders have to persuade receivers to make decisions which they know to be incorrect. This experiment tests whether the automatic extraction of linguistic cues contributes to differentiation between deceptive and non-deceptive texts. In their work, though subjects were motivated to play the game, there was no special motivation for the deceptive players to be deceptive. In the game we design we make a point of motivating the deceptive player to act accordingly. The deceptive senders were found to be more expressive than honest senders. This is contrary to previous work that shows that liars tend to be less expressive. This result was explained by the fact that previous studies were conducted in an interview type scenario where a liar had to

answer in real time without planning or editing, as opposed to this experiment where liars can think about and edit the message before they send it. This work was later extended [31] to a chat-based environment where the effects of the type of environment on deception detection by human subjects were studied. They observed that humans felt that viewing the messages assisted detection more than viewing video of the actual typing of the liars. However, the accuracy of the human deception detection in all settings was lower than 50%. Although the text-based domain that we use is similar to the domains used by Zhou et al., our research differs substantially from these studies. We use machine learning to differentiate between deceptive and non-deceptive texts as opposed to these studies that either collected statistics on the existence of automatically extracted phrases or used humans for detecting deception.

Newman et al. [20] and Toma and Hancock [29] find that certain words are chosen more often by liars. Zhou et al. use text-based computer mediated environments to study what linguistic-based cues are used more in deceptive texts than in non-deceptive texts [31] and how modality affects human deception detection in chat-based environments [30]. However they do not build a model for deception detection with machine learning.

Another study on deceptive language is performed by [19]. Their data consists of paragraphs written by subjects who were asked to write their opinion on a topic and then write the opposite opinion. Naive Bayes and Support Vector Machines were used to classify words in this data set. Words that are connected to the self such as "I, friends, self" are often used in the honest text. In contrast, detached words such as "you, other, human" appear more often in the deceptive text. When topics were evaluated separately, an average detection rate of 70.8% was found. When testing the portability of the classifiers across topics, accuracy dropped to 59.8% on average.

Toma and Hancock analyze text which is presented in online dating profiles [29]. This is a setting where the subjects have time to revise their descriptions, and make sure they sound honest. Yet again deceptive participants use less self-references, and use more negations. 63% of the profiles were correctly classified using logistic regression. Ott et al. conduct a study [23] in which they try to differentiate honest hotel reviews from fake ones. They asked Mechanical Turk workers to write fake *positive* reviews for a set of hotels and fetched (presumable) truthful *positive* reviews from TripAdvisor. One of their proposed methods reaches a success rate of nearly 90%. Ott et al. also conduct a recent similar study [22] for *negative* reviews. These types of studies are inherently different to the discussion environment that we study, since in a discussion environment people are part of a group and refer to each other, while in writing a review etc. each person merely posts the review, doesn't refer to anyone else and isn't required to later post any additional response. While writing a review, description or opinion is composed of a large bulk of text; in a discussion environment the messages are short and fast. As we chose to focus on the dynamics of the discussion and therefore use structure sentences, most of the features which heavily rely on syntax features are not applicable to our work.

Mancilla-Caceres et al. [17] developed a game in order to identify children who act as bullies within a social network. The game involves a restricted set of resources and two tasks. One is collaborative and the other competitive. The communication between participants is through text messages. Mancilla et al. manually classify the messages exchanged and, using machine learning, are able to detect the bullies with an accuracy of slightly above 60%. This game is reminiscent of our game as it uses text messages and players are given a set of resources. However our game is aimed at assessing dishonest behavior and not aggressive behavior.

Similar work dealing with cyberbullying was performed by Bosse and Stam [5]. They deploy an agent in a game in order to reduce cyberbullying in it. However, their agent is given special capabilities over the ordinary players, such as using rewards and punishments. Additionally, since they do not deal with deception, their agent is not required to appear trustworthy to the other players.

Human-agent interaction has taken a major role in the negotiation environment [27, 14, 3]. Occasionally, in such environments, agreements aren't enforceable and thus an agent must reason about the probability or the likelihood that the opponent might be deceiving and will not keep his promise. However, to the best of our knowledge, in this realm, predictions are based merely on the user's actions and monetary promises and not on verbal or textual context.

Our work should not be confused with the following works on deception and deception detection in agents, [26, 16, 7, 9], where agents must determine whether other agents are lying, or agents use deception themselves. Santos an Li [26] study deception detection in a multi-agent environment. They focused on the agent's reasoning process in a medical inference system. In this system the agents infer medical information in an intensive care unit. Some of these agents are simulated to be deceptive. Correlation between the various agents in the system are used to make predictions on the agent reasoning process and are then used to detect deception. Kaluža, Kaminka and Tambe [16] propose a two step approach for detecting suspicious behavior in agents. In their study they simulate an airport domain, where agents simulate passenger behavior. Some of the agents simulate suspicious behavior. The detection of suspicious agents is based on sequences of events performed by the agents. In the first stage trigger events are detected, and in the second multiple events are combined to determine suspiciousness. An interesting study is performed by Bridewell and Isaac [7]. They developed a framework for socially aware inference, using deception as an example of social behavior. They model the mental states of other agents using beliefs, goals and intentions. Christian and Young [9] use strategic deception in a planner in order to achieve certain goals. The deception is used in order to improve planning. This is an example to the agent being deceptive itself rather than detecting deception in others.

THE DIG AGENT

We present the Deception In Group detector and catcher Agent (DIG). The DIG agent is capable of participating in the pirate game while posing as a regular player. During an ongoing session of the pirate game, DIG uses input from multiple previous games, along with input from the current game, and outputs the next sentence it wants to contribute to the discussion. DIG has four components. The first is used to detect the deceiver and is called the *'Suspector'*. The second component, the *'Output Selector'*, aims to avoid raising suspicion (as being the pirate). The third component is the *'Accuser'*; it directs the other credible players towards DIG's suspect. The fourth component, the *'Responder'*, responds to sentences other players may direct to DIG.

The structure of DIG is presented in Figure 2. All of the sentences from the chat in the current game are provided as input to the *'Suspector'* component which (using machine learning) outputs an ordered list of suspects. This list of suspects is provided as input to the *'Accuser'* and is used to output a time-dependent list of possible sentences for the DIG agent to use. In addition, the last sentence in the current chat is input to the *'Responder'*. In the case this sentence is referred to DIG, the *'Responder'* (Using training data) outputs another optional sentence set. Based on the sentences in the chat and the optional sentence set, the *'Output Selector'* (using machine learning) selects a sentence for DIG to output. We describe the components in detail.

The Suspector Component

The 'Suspector' relies on a machine learning model built offline that classifies various users' texts and whether or not they played the role of a deceiver. We use data from previous games. For each player in each game we build a feature vector. We use the feature set mentioned below and label each player vector as being the pirate or a credible villager. We used an SVM classifier with a linear kernel [10] which is built on these player vectors. The *'Suspector'* is interested in the SVM-score (the signed distance from the separating hyperplane) and not in the actual SVM classification. The online part of the component works as follows: it extracts feature vectors from all of the sentences that appeared in the game. The feature vectors are fed to the model (built offline), and the model outputs the degree of suspiciousness for each of the players. The component then sorts the players according to their degree of suspiciousness. The *'Suspector'* is used mainly by the *'Accuser'* component.

The Output Selector Component

The 'Output Selector' gets a set of sentences from the *'Accuser'* and the *'Responder'* components and using another machine learning model outputs the least suspicious sentence from the perspective of human players (the sentence that would raise the least suspicion if said at the current time). Similarly to the model used in the *'Suspector'* component, the model we build here (offline) uses player feature vectors and uses SVM as described above. However, this model uses different labeling. The player vectors are labeled as being suspicious or not, based on whether they received votes or did not (rather than whether the player was a pirate or not). Note: just as the *'Suspector'* component, this model is built on a group of sentences and not on single sentences. The model is fed with features derived from all of the sentences that appeared in the game, together with each option provided as input by the *'Accuser'* or *'Responder'*. Based on the SVM-score obtained from this model, for each of the options, the *'Output Selector'* chooses the sentence which would raise the least suspicion, towards DIG, if said at the current time. In addition to the above, a random delay was added to each sentence said by DIG, in order to appear human.

The Accuser Component

The 'Accuser' is awoken by a clock at given times[1] and selects possible sentences for DIG to say. The list of sentences output by this component is based on a manual tagging of all sentences. For example, some sentences were tagged as opening sentences, such as "Who is the pirate", "I am not the pirate" and "I am a credible villager". Other sentences were tagged as accusing sentences, as alter justification (standing up for a different player) and as questions to other players. The *'Accuser'* component could also output an empty sentence as part of the set of sentences. Depending on the time left until the end of the game and according to a manually constructed table which determines which sentence tags are appropriate for which time, and based upon the *'Suspector'*s suspects, the *'Accuser'* provided the *'Output Selector'* a set of possible sentences to output. For example, when 100 seconds remain till the end of the game, the *'Accuser'* component provides all sentences which were either tagged as accusing sentences (towards the *'Suspector'*s top suspect) or tagged as alter justification sentences (towards the *'Suspector'*s least suspected player). The sentence tags are determined under the assumption that as time progresses, more confident sentences should be chosen. The exact sentence to say is chosen by the *'Output Selector'*. The *'Accuser'*s main goal is to encourage the other players to vote for DIG's suspect. This component is also in charge of casting a vote (on the *'Suspector'*'s leading suspect) in the voting phase.

The Responder Component

The 'Responder' is activated when someone referrers to DIG in the previous sentence. The *'Responder'* then searches the data set for sentences used by other players when posed with the same question or statement. The *'Responder'* provides these sentences to the *'Output Selector'* which determines which sentence to output. Similarly to the *'Accuser'* component, if any of these sentences require reference to another player (if the sentence is tagged as accusing or alter justification), the player which the *'Responder'* refers to depends on the list of suspects obtained from the *'Suspector'*.

DIG Feature Selection

For the machine learning used in DIG's *'Suspector'* and *'Output Selector'* components we need to define the feature set obtained from the structured chat. We used 41 features, some of which are mentioned in [30] and the rest were dedicated

[1]We used fixed time values, however, these clock values may be learned online and updated according to performance.

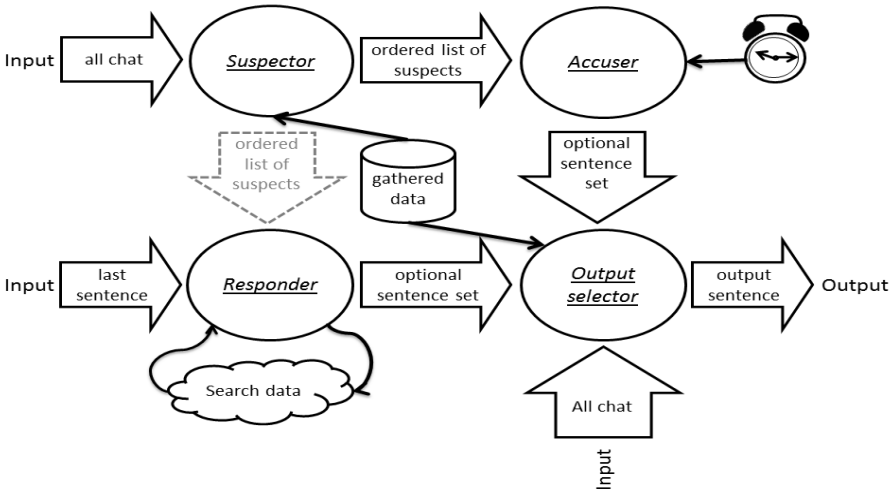

Figure 2. The structure of the DIG agent. Circles represent components, arrows represent input and output.

to our problem. Since we wanted our approach to be scalable and suitable also for a free chat environment, we used more general features such that other sentences (which do not appear in the structured sentences) may be mapped to these features as well. Some of the features also depend on the order of the sentences and not solely on whether a sentence was said (or how many times it was said). For the learning, each player is treated as an instance which must be classified. All of the 41 features that we used had a positive impact. The top 10 features are:

- **Fraction of talking:** the fraction of sentences said by the player out of the total number of sentences said (in current game).
- **Accusations:** the number of accusing sentences used by the player. And the fraction of accusations from all sentences said by that player.
- **Consistency:** indicates the level of accusation consistency towards the other players. A player who always accused the same other player will have a high value, while a player who accused all three other players equally will have a low value. This feature calculates the maximum among the accusations towards each of the other players and divides it by the total number of accusations.
- **Characteristics:** indicates how many times a player referred to a different player's characteristics such as being too quiet, talkative, accusatory etc. This was easily identified since we used structured sentences (see the third form of sentences in Figure 1).
- **First sentence:** indicates whether the player was first, second, third or fourth to say his first sentence.
- **First Accusation:** indicates whether the player was first, second, third or fourth to accuse a different player of being the pirate.
- **Self-justification:** indicates how many times the player stood up for himself (saying he is not a pirate etc.).
- **Alter justification:** indicates how many times the player stood up for a different player (saying that a different player is not a pirate etc.).
- **Agreeing:** indicates whether a player agreed with other players (and how many times).
- **Agree to accusation:** indicates whether a player agreed with another player when that other player accused someone (and how many times).

EXPERIMENTS

All of our experiments were performed using Amazon's Mechanical Turk service (AMT) [1][2]. Participation in all experiments consisted of a total of 320 subjects from the USA, of which 47.8% were females and 52.2% were males. The subjects' ages ranged from 18 to 67, with a mean of 32 and median of 30. All subjects had to pass a short quiz to assure that they understood the rules and had to practice the usage of the structured sentences before they could play. We ran experiments with the two versions of the game ("informer" and "basic"), each with two different setups. We ran the game with only human players and then with an agent playing the role of one of the credible villagers (the agent is never the pirate). The subjects weren't told about the agent and therefore assumed all players were humans. According to comments we collected, no players suspected a nonhuman player.

Participants had to play 5 games each. Each game was played with different participants, so no deductions based on participants' behavior can be made between games. The discussion phase was limited to 240 seconds and the voting phase was limited to 30 seconds. The average number of sentences per

[2]For a comparison between AMT and other recruitment methods see [24].

game was 36.6. The subjects were paid 62 cents for participating in the study. They gained 12 cents for every time they were a credible villager in a group that managed to catch the pirate and 36 cents if they were a pirate who managed to escape with the gold. A stake of 36 cents for a single game is relatively high in AMT (58% of total payoff for 5 games). This is in order to increase the players' incentive to play seriously.

The players enjoyed the game very much. When asked whether they enjoyed the game, on a scale between 1 to 5 they gave it on average a score of 4.01 (standard deviation of: 1.06). The players also provided very positive feedback such as: "It was the best survey or game I have done...", "This was really fun... Thanks for the good time!" and "That was so much fun! ... I could play it all day!". . Interestingly, the subjects found the "informer version" of the game more enjoyable than the "basic version": 4.22 vs. 3.79 ($p < 0.01$). This can be explained by the fact that in the "informer version" both the pirate (needs to incriminate a certain player) and the credible villagers (need to find who is trying to incriminate a different player) have a clearer task. On average the subjects did not seem to believe that they played very well. The subjects were asked how well they believed that they played (on a 1 to 5 scale) and the average response was 2.89 (standard deviation of: 1.05). Interestingly, the subjects seemed to believe that they played slightly better in the "informer version": 2.94 vs. the "basic version": 2.84, however, these differences are not statistically significant and have no support in the actual results (as presented later).

DIG Construction

Recall that two of DIG's components (the 'Suspector' and the 'Output Selector') are based on machine learning. We now describe their construction.

Constructing the 'Suspector' component

The *'Suspector'* component builds a model that identifies whether a player is a pirate or a credible villager. We used the results obtained from the all-human games (with no agent) as our learning data for this component. For the "basic version" we used 204 examples and for the "informer version" we used 272 examples. We obtained an accuracy of 72.1% and an F-Measure of 0.69 for the "basic version" of the game, and similarly an accuracy of 72.8% and an F-Measure of 0.7 for the "informer version". Some interesting insights into the pirate's behavior are: The pirate is an average talker (not too much, not too little), doesn't talk much about himself, avoids direct accusation but hints that someone else is the pirate, is often the third to talk (rarely the first). In the "informer version" the pirate talks mostly about the same player, and in "basic version" the pirate talks about others quite equally.

Constructing the 'Output Selector' Component

We use this component to supply DIG with information regarding which sentences to use and which to avoid. The model of this component was less accurate (using the same feature-set above), with only 56.4% accuracy and an F-Measure of 0.56 in the "basic version". This result implies that we aren't very successful at predicting which players will receive votes and which will not (possibly because the subjects themselves might have voted almost randomly). The accuracy in the "informer version" was slightly higher at 61.0% and an F-Measure of 0.61, implying that in the "informer version" we were more successful at identifying whom the other players suspect. We found that people think that the pirate is quiet, is the first to talk, and that the pirate tends to agree with others, especially when making accusations.

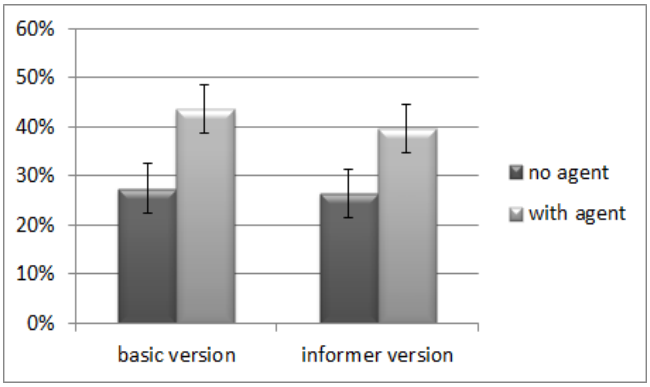

Figure 3. Success rate in catching the pirate. Compares both versions of the game, with and without an agent.

Based on this model, the *'Output Selector'* component chose to say "I am a credible villager" at the beginning of the game, and decided to use "Alter justification" (stood up for other players) with the least suspicious player (when such sentences were optional according to the *'Accuser'* component's output). The *'Output Selector'* component tried to avoid any accusations, and therefore started to accuse a player only towards the end of the game (when the *'Accuser'* component only provided sentences tagged as accusing). It always avoided asking other players whether they were pirates, as this was also shown to raise suspicion. The *'Output Selector'* preferred sentences which clearly said for whom it would vote "I will vote for..." rather than using a phrase such as "I believe that I will vote for..." as, once again, while the first is shown to lower the other players' suspicion, the latter is shown to raise it.

Experimental Results

Figure 3 presents the success rate of the credible villagers at catching the pirate in both versions of the game, with and without the agent. In both versions of the game the groups including the DIG agent (basic: 43.7%, informer: 39.7%) significantly outperform (using chi square test, with $\alpha = 0.05$) the groups that didn't include the DIG agent (basic: 27.5%, informer: 26.5%). The performance of the human players without the agent is very close to the expected utility of random voting which is 0.26 (see random voting analysis in the appendix).

Table 1 summarizes the voting results. These should not be confused with the success rate. The number of correct votes in the setting that includes DIG were higher than those without DIG. This is true both in the "basic version" of the game (41.9% with DIG vs. 35.9% without DIG) and in the "informer version" (42.4% with DIG vs. 33.8% without DIG).

game version	agent participation	correct votes	agent correct votes	human correct votes
basic	no agent	35.1%	-	35.1%
basic	with agent	41.9%	48.3%	38.6%
informer	no agent	33.8%	-	33.8%
informer	with agent	42.4%	46.6%	41.6%

Table 1. Voting Accuracy (random vote = 33.3%)

Even more interesting are the number of correct human votes. Humans cast more correct votes when DIG participated than when humans played alone (see Table 1 for details). This is true both for the "basic version" and for the "informer version". This implies that not only did DIG help the group by casting more accurate votes, but DIG also seems to improve the number of correct votes cast by the humans in its group. We would also like to mention that very rarely did subjects cast an empty vote (only in 4% of the cases).

We end this section by testing the ability of DIG to avoid suspicion (as being the pirate). Recall that one of DIG's properties is to choose sentences that reduce suspicion. We measure suspicion by counting the number of votes cast by the other credible villagers against DIG. Note, that as DIG was never the pirate, a low vote percentage is better than a high one. In the "basic version", 32.2% of the votes were cast against DIG (meaning they falsely thought DIG was the pirate), which is still a little below average (which is 33.3%) and therefore may be reasonable. However, unfortunately, 37.6% of the votes in the "informer version" were cast against DIG (which is higher than the average). We explain this by the fact that DIG tried to encourage the other humans to vote for the player which DIG detected as the pirate, as required by the *'Accuser'* component. This act in the "informer version" probably raised suspicion (as players might have incorrectly assumed that DIG is the pirate and therefore trying to incriminate the informer). Another reason that could have caused suspicion in both versions is that people might have noticed that DIG doesn't play as a completely normal player, as the *'Responder'* component was not the focus of this study. We believe that had we not selected DIG's sentences with care using the *'Output Selector'* component, the suspicion would have been greater. We plan to investigate this in future work. An example of one of the games played by DIG (in the basic version) can be found at: http://azariaa.com/GameExample.htm.

DISCUSSION

One might wonder how the use of open chat rather than structured sentences would affect the relationship between the success rates of human players and the agent. Previous work [31] has shown that people are roughly as good as chance at detecting liars. We believe that since we succeeded while using structured sentences, using free text, and extracting features from it should be even more successful, but requires additional research.

We designed our agent to remain unsuspicious, as we did not want it to be falsely accused as being the pirate, and since we wanted it to be able to influence other players. We believe that if the agent seemed human these goals would be accomplished. However, behaving like a human was not a goal by itself. It is unclear how well the agent would perform, if the context was one not restricted to structured sentences, and the agent could identify it-self as non-human. This would require additional research. On the other hand, it would be easier for DIG to function in such an environment.

In our game people had little experience at deceiving. It is interesting to know how well DIG would perform against experienced deceivers. Note that this problem is not specific to our work, but a known challenge in deception detection in general. People, who would be experienced in deceiving in these types of environments, would be experts at deceiving people, rather than agents. We have shown people use different features for deception detection than DIG. Experienced deceivers would probably avoid the features picked up by humans. However they may still display other features, which DIG would pick up. Therefore, DIG may be able to succeed in detection of experienced deceivers as well. This has not been determined.

If DIG does not have enough data collected yet, it could use data from a similar domain. Alternatively DIG can collect data from Mechanical Turk, or ask humans to tag discussions.

In our game, we have a single deceptive participant. We would expect that in a scenario where there may also be zero or multiple deceptive participants DIG will not perform as well, but will still perform much better than humans (who will also not perform as well). An extension to this work would need to design a new game to account for multiple deceptive participants.

CONCLUSIONS AND FUTURE WORK

In this paper we presented "the pirate game" as a platform that enables the study of deception in a computerized discussion text-based environment. We presented two versions of the game, the "basic version" where the deceiving participant could hide and an "informer version" where the deceptive player was encouraged to be active. In both versions we found that humans' success rate in catching the deceptive player was similar to the success rate of random votes achieved when all players cast a random vote. This result is consistent with previous studies which show that people do not perform much better than chance when asked to detect deception.

We introduced DIG, an agent that uses machine learning to build a successful strategy for deception detection. DIG was successful at detecting the deceptive player in both versions of the game. DIG provided two contributions to the group of players. The first is that as we look at a group task, the ability of DIG to cast a correct vote increases the group ability. The second contribution of DIG is the indication that other players have higher detection rates when the agent participates.

The platform we introduce is a first step towards decreasing deception in textual discussion based environments. The input method relies on the use of structured sentences. Although, this method has many benefits, as we described, many natural discussion based environments use free text. As

NLP continues to develop and improve, it is natural to extend our work to include NLP tagging and integrate it into our agent.

In future work we also intend to pursue a method for the agent to select its sentences so that it increases its probability of detecting the pirate. The agent will need to find a question for each situation that, when answered, may either increase or decrease the probability that the agent's current suspect is the real pirate. This is a very challenging issue since each sentence may change the behavior of the rest of the game.

ACKNOWLEDGMENT

This work was supported in part by ERC grant #267523, and ARO grants W911NF0910206 and W911NF1110344.

REFERENCES

1. Amazon. Mechanical Turk services. http://www.mturk.com/, 2012.
2. Aylett, R., Hall, L., Tazzyman, S., Endrass, B., André, E., Ritter, C., Nazir, A., Paiva, A., Höfstede, G., and Kappas, A. Werewolves, cheats, and cultural sensitivity. In *AAMAS'14* (2014), 1085–1092.
3. Azaria, A., Aumann, Y., and Kraus, S. Automated strategies for determining rewards for humanwork. In *AAAI'12* (2012).
4. Bhaskaran, N., Nwogu, I., Frank, M., and Govindaraju, V. Lie to me: Deceit detection via online behavioral learning. In *FG'11* (march 2011), 24 –29.
5. Bosse, T., and Stam, S. A normative agent system to prevent cyberbullying. In *WI-IAT'11*, vol. 2, IEEE (2011), 425–430.
6. Braverman, M., Etesami, O., and Mossel, E. Mafia: A theoretical study of players and coalitions in a partial information environment. *Annals of Applied Probability 18*, 3 (2008), 825–846.
7. Bridewell, W., and Isaac, A. Recognizing deception: A model of dynamic belief attribution. In *Advances in Cognitive Systems: Papers from the 2011 AAAI Fall Symposium* (2011), 50–57.
8. Chittaranjan, G., and Hung, H. Are you a werewolf? detecting deceptive roles and outcomes in a conversational role-playing game. In *ICASSP* (2010), 5334–5337.
9. Christian, D., and Young, R. M. Strategic deception in agents. In *AAMAS'04*, IEEE Computer Society (2004), 218–226.
10. Cortes, C., and Vapnik, V. Support-vector networks. *Machine learning 20*, 3 (1995), 273–297.
11. Dias, J., Aylett, R., Reis, H., and Paiva, A. The great deceivers: Virtual agents and believable lies. In *CogSci 2013* (2013), 2189–2194.
12. Ekman, P. *Telling Lies: Clues to Deceit in the Marketplace, Politics, and Marriage*. W. W. Norton & Company, 1991.
13. Farrell, J., and Rabin, M. Cheap talk. *The Journal of Economic Perspectives 10*, 3 (1996), 103–118.
14. Gal, Y., Kraus, S., Gelfand, M. J., Khashan, H., and Salmon, E. Negotiating with people across cultures using an adaptive agent. *ACM TIST 3*, 1 (2012).
15. Graciarena, M., Shriberg, E., andFrank Enos, A. S., Hirschberg, J., and Kajarekar, S. Combining prosodic, lexical and cepstral systems for deceptive speech detection. In *Proc. IEEE ICASSP* (2006).
16. Kaluža, B., Kaminka, G. A., and Tambe, M. Detection of suspicious behavior from a sparse set of multiagent interactions. In *AAMAS'12* (2012), 955–964.
17. Mancilla-Caceres, J. F., Pu, W., Amir, E., and Espelage, D. Identifying bullies with a computer game. In *AAAI* (2012).
18. Migdal, P. A mathematical model of the mafia game. *CoRR abs/1009.1031* (2010).
19. Mihalcea, R., and Strapparava, C. The lie detector: Explorations in the automatic recognition of deceptive language. In *ACL/AFNLP (Short Papers)* (2009), 309–312.
20. Newman, M. L., Pennebaker, J. W., Berry, D. S., and Richards, J. M. Lying words: predicting deception from linguistic styles. *Pers Soc Psychol Bull 29(5)* (2003), 665–75.
21. Ott, M., Cardie, C., and Hancock, J. Estimating the prevalence of deception in online review communities. In *WWW '12*, ACM (2012), 201–210.
22. Ott, M., Cardie, C., and Hancock, J. T. Negative deceptive opinion spam. In *HLT-NAACL 2013* (2013).
23. Ott, M., Choi, Y., Cardie, C., and Hancock, J. T. Finding deceptive opinion spam by any stretch of the imagination. In *HLT '11 - Volume 1* (2011), 309–319.
24. Paolacci, G., Chandler, J., and Ipeirotis, P. G. Running experiments on Amazon Mechanical Turk. *Judgment and Decision Making 5*, 5 (2010).
25. Raiman, N., HayleyHung, and Englebienne, G. Move, and i will tell you who you are: detecting deceptive roles in low-quality data. In *ICMI '11*, ACM (2011), 201–204.
26. Santos, E., and Li, D. On deception detection in multiagent systems. *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on 40*, 2 (march 2010), 224 –235.
27. Simari, G. I., Broecheler, M., Subrahmanian, V., and Kraus, S. Promises kept, promises broken: An axiomatic and quantitative treatment of fulfillment. In *KR* (2008), 59–69.
28. Toledo, A., Katrenko, S., Alexandropoulou, S., Klockmann, H., Stern, A., Dagan, I., and Winter, Y. Semantic annotation for textual entailment recognition. In *Advances in Computational Intelligence*. Springer, 2013, 12–25.

29. Toma, C. L., and Hancock, J. T. Reading between the lines: linguistic cues to deception in online dating profiles. In *CSCW '10*, ACM (2010), 5–8.

30. Zhou, L., Burgoon, J. K., Nunamaker, J. F., and Twitchell, D. Automating linguistics-based cues for detecting deception in text-based asynchronous computer-mediated. *Group Decision and Negotiation 13* (2004), 81–106.

31. Zhou, L., and Zhang, D. Typing or messaging? modality effect on deception detection in computer-mediated communication. *Decision Support Systems 44*, 1 (2007), 188 – 201.

APPENDIX

FORMAL MODEL

In the formal model, the Pirate Game consists of k players $P = \{p_1, p_2, ..., p_k\}$, one of which is a pirate. We assume $k \geq 4$. The identity of the pirate player is known only to the pirate himself. The game includes two phases: the first phase consists of a communication phase where all players can discuss their strategy. The communication in this phase bears no cost and is not binding ("cheap talk" [13]). The second phase is a voting phase where all players except the pirate may simultaneously cast their vote $v(p)$, where $v : P \rightarrow \{\{p_1\}, \{p_2\}, ..., \{p_k\}, \phi\}$ and ϕ indicates an empty vote. It is assumed that the pirate always casts an empty vote ϕ. If the majority of the votes are cast against the pirate, all players but the pirate receive a point and the pirate receives zero points. Otherwise the pirate receives a point and the other players receive zero points. More formally, we define a function for a player's role:

$$r(p) = \begin{cases} 0 & \text{if p is the pirate} \\ 1 & \text{else} \end{cases} \quad (1)$$

We will use $\overline{r(p)} = 1 - r(p)$. The reward function for each player $u(p)$ is given by:

$$u(p) = \overline{r(p)} + (\text{-}1)^{\overline{r(p)}} \cdot \mathbb{1}\Big\{\Big(\sum_{i=1}^{k} \sum_{p' \in v(p_i)} \overline{r(p')}\Big) \geq \Big\lceil \frac{1 + \sum_{i=1}^{k} |v(p_k)|}{2} \Big\rceil \Big\} \quad (2)$$

where $\mathbb{1}\{\}$ is the indicator function.

Equilibrium Strategies

Assuming all players are perfectly rational, the communication phase doesn't reveal any information regarding the pirate's identity, since the pirate may act as if he were a credible player. We provide the equilibrium strategies analysis in order to compare human performance to that of perfectly rational players.

Pure strategy equilibrium

Given a player p', the strategies $\forall p | r(p) = 1, v(p) = \{p'\}$ are in equilibrium, since any deviation from the equilibrium by a single player will not change the final result[3]. Assuming p' is random, this equilibrium assures an expected utility of $\frac{1}{k}$ for the credible players (and $1 - \frac{1}{k}$ for the pirate). Although it is in equilibrium, agreeing upon the player to vote for (p') requires the communication phase to allow simultaneous messaging. Assuming simultaneous messaging, all players must choose a random number $x(p)$ between 1 and k, simultaneously publish it, and then vote for player p_l where $l = \sum_{p \in P} x(p) \pmod k$. Although the pirate may choose a non-random number, since he has no knowledge of the numbers chosen by the other players, the result remains random. This method was also proposed in [6].

Mixed Equilibrium

Following is a mixed equilibrium for the game:
$\forall p | r(p) = 1, \forall p' | p' \neq p, v(p) = \{p'\}$ with probability $\frac{1}{k-1}$.
The expected utility for the credible players using this equilibrium is given by the following binomial distribution mass function: $\sum_{j=\lceil \frac{k}{2} \rceil}^{k-1} \binom{j}{k-1} (\frac{1}{k-1})^j \cdot (1 - \frac{1}{k-1})^{k-1-j}$

When $k = 4$, the above mixed equilibrium yields a slightly greater expected utility than the pure equilibrium mentioned before (0.26 vs. 0.25). Being symmetric towards all players, the mixed equilibrium doesn't require any prior communication. However, as k increases, the mixed equilibrium yields a very low expected utility for the credible players. For example, when $k = 7$ the expected utility of the credible players goes down to 0.01.

Alternative Models

Informer version

We also consider a model with a slight modification to the pirate's utility function, where there exists a player $\bar{p}$ whose identity is known only to the pirate. The pirate's utility function is identical to Equation 2, except if $\bar{p}$ doesn't receive even a single vote, in which case the pirate's utility is 0 regardless of the other votes. Formally, if $\sum_{i=1}^{k} \sum_{p' \in v(p_i)} r(p') = 0$, then the pirate's utility is 0. This change doesn't affect the equilibria, as they do not depend on the pirate's actions.

Voting pirate

Allowing also the pirate to vote significantly reduces the credible villagers' probability of catching him. In addition to allowing the pirate to vote, one might also consider requiring that the pirate need only receive the plurality of votes (rather than the majority), where in case of a tie, the chosen player is defined by a toss of a coin. This drastically changes the equilibria for all games. As long as all players vote and no player votes for himself, almost any mixed strategy is in equilibrium with an expected utility of $\frac{1}{k}$ for the credible players, since, in every game, one and only one player is chosen. For symmetric reasons, there is a probability of exactly $\frac{1}{k}$ that this player is the pirate.

[3] if the majority of players plus one have a strategy of $v(p) = \{p'\}$, all other players may have any other strategy and still be in equilibrium. Although additional equilibria exist, in this appendix we give only a few examples of equilibria and do not try to list them all.