First Steps in Chat-Based Negotiating Agents

Inon Zuckerman, Erel Segal-Halevi, Avi Rosenfeld and Sarit Kraus

1 Introduction

In multiagent environments, similarly to human societies, agents might have conflicting interests. Negotiation is often used as a protocol to bridge those differences and reach an agreement. Negotiations scenarios can take difference forms: haggling over a single issue versus several interdependent issues, how much does the parties know about each other? do they have time constraints?

To date, a variety of agents have been created to negotiate with people within a large spectrum of settings including: the number of parties, the number of interactions, and the number of issues to be negotiated. Katz and Kraus [15] proposed an agent for one-shot interactions in an environment where only one issue needed to be negotiated between two parties (bilateral negotiation). The *AutONA* agent was developed for repeated interactions between buyers and sellers over the price and quantity of a given product [4]. More complex agents have been created for multi-attribute negotiations involving several issues to be considered. For example, the *KBAgent* has been shown to be the most effective agent in achieving agreements with people in several domains involving multiple attributes [23].

Two elements are common to all these agents. First, these agents are all based on the assumption that the human negotiators use bounded rationality. People did not

I. Zuckerman (✉)
Department of Industrial Engineering and Management,
Ariel University, 4070000 Ariel, Israel
e-mail: inonzu@ariel.ac.il

E. Segal-Halevi · S. Kraus
Department of Computer Science, Bar-Ilan University,
5290002 Ramat-gan, Israel

A. Rosenfeld
Department of Industrial Engineering, Jerusalem
College of Technology, 9116001 Jerusalem, Israel

K. Fujita et al. (eds.), *Next Frontier in Agent-Based Complex Automated Negotiation*, Studies in Computational Intelligence 596,
DOI 10.1007/978-4-431-55525-4_6

successfully reach agreements with agents based on notions of equilibrium or optimal methods, and thus alternatives needed to be found for all agents [19]. Second, all agents needed mechanisms for dealing with incomplete information. This is typically done through reasoning about the negotiating partners by learning their preferences and strategies [11].

The key point this paper addresses is a study of how to extend current state of the art agents to use natural language processing. Unfortunately, this ability is lacking in current state of the art negotiation agents—something that has been previously noted [19]. This inherent limitation requires these agents to "force" their human counterparts to interact via menus or other non-natural interfaces.

Towards creating agents that use natural language, this paper addresses what extensions, if any, are needed to bridge this gap. As a first step towards creating negotiation agents with full NLP capabilities, we conducted extensive studies of interactions between the leading automated negotiation agent and people. We compared how people negotiated with this agent through its previous menu-based interface, and a new chat-based interface that allowed people to converse freely with the agent. This was done by using a *Wizard of Oz* approach [10, 16], in which a human is sitting behind the scenes translating the natural language sentences to objects that are recognizable by the agent.

Next, we developed a fully working prototype of a Natural Language Processing (NLP) system to replace the "wizard". This complex component is built of several components: A natural language understander, a dialog manager and a natural language generator. Following an extensive training period in which we let the system talk to Amazon Turk workers, we managed to attain a correct classification rate of 72 %. This is considered a very high classification rate in NLP solutions. We then evaluated the performances of the negotiation agent when using the NLP system versus using the *Wizard of Oz* approach.

This paper presents three important results based on this study. First, we discovered that the automated negotiation strategies did not transfer well to more natural forms of conversation. Simply adding a chat-based interface instead of a menu-based interface to the existing agent yielded agreements that were significantly *worse* for the agent, while the utility for the human player remained the same. In addition, we found that the human partners were significantly happier with the final agreement, and they perceived the final outcome to be more balanced if they were using the chat-based interface, despite the fact that they attained the *same* average utility in both interfaces.

Second, we managed to isolate the reason for the algorithm's inability to cope with partial agreements as the main cause for its decreased performance. One key issue that we study is the centrality of creating partial agreements within natural language based negotiation agents. It is known that bounded rational agents (such as humans) find that simultaneously negotiating a complete package might be too complex [2, 3], and therefore they prefer to negotiate issue-by-issue. As our next section details, this is an open issue within the general negotiation research community, but is evidently a key issue that must be addressed by agent designers.

Lastly, we show that the current state-of-the-art NLP solutions are still limited and that the agent who is using the *Wizard of Oz* approach performs as par with the

NLP based agent. This points out to the fact that further developments in the field of NLP are required in order to facilitate the construction of Chat based agents.

2 Related Work

This paper's main contribution lies in empirically analyzing how negotiation agents should be extended to support more natural interfaces, and specifically a chat interface. Extensive studies in the field of Human Computer Interactions (HCI) have noted that the goal of any system should be an intuitive interface with the stress being put on creating agents which operate in environments which are as real and natural as possible [7, 8]. Thus, following these approaches, it is critical to develop natural language support for negotiation agents to allow for these types of "normal" interactions [17]. This form of typing as natural interaction is referred to as *Natural-language interaction* (NLI) in the literature. There have been numerous informal tests of NLI systems, but few controlled experimental comparisons against some other design [25].

While automated negotiation agents have been developed for quite some time, unfortunately, even state of the art negotiation agents do not yet support natural language interactions. Over twenty years ago in [18] they developed an agent called *Diplomat*, that played the Diplomacy game with the goal to win. Byde et al. [4] developed *AutONA*, an automated negotiation agent. Their problem domain involves multiple negotiations between buyers and sellers over the price and quantity of a given product. Jonker et al. [13] created an agent to handle multi-attribute negotiations which involve incomplete information. The *QOAgent* [20] is a domain independent agent that can negotiate with people in environments of finite horizon bilateral negotiations with incomplete information. The negotiations consider a finite set of multi-attribute issues and time-constraints. Costs are assigned to each negotiator, such that during the negotiation process, the negotiator might gain or lose utility over time. The game involves negotiations in multi-attribute settings with incomplete information concerning the other agents' goals, and misleading information can be exchanged between the different agents.

We focus on the KBAgent, which like the *QOAgent* also considers negotiations with a finite set of multi-attribute issues and time-constraints, but has been shown to be the most effective agent in achieving agreements with people in several domains [23]. This area continues to be quite popular, with one active research avenue being the ANAC (Automated Negotiating Agents Competition) Workshop. Since 2010, this competition has focused on agents that use the GENIUS interface [1]. However, we note that even to date, this competition focuses on agent-agent interactions and the interface supports only menu-based interactions between agents and people.

To address this limitation, we study what logical extensions are needed, if any, to make existing negotiation agents suitable for natural language. Previous economic and behavior research into people's negotiation suggests that the current approach of attempting an agreement on all issues simultaneously might be ineffective. For example, Bac and Raff [2] found that simultaneously negotiating a complete package might be too complex for individual buyers. Furthermore they show that, in the context of incomplete information with time discount, the more informed player ("strong" in their terminology) will push towards issue-by-issue negotiation. Busch and Horstmann [3] found some people might like to decide all issues at once, while others prefer to decide one by one. Chen [6] studied issue-by-issue negotiation with opt-out factor, and argues that when the opt-out probability is low, agent prefer to negotiate a complete package because intuitively we know that the negotiations can last long enough so that agents can get to a "win-win" situation. However, with high opt-out probability, agents prefer issue-by-issue negotiation. Thus, one key contribution of this paper is its study as to how the negotiation strategy should be changed when agents cannot propose issue-by-issue agreements.

3 Methodology

The main goal of this research was to push the envelope of automated negotiation research by moving from menu-driven interfaces to chat based environments. As this work transitions from the fruitful work of previously developed agents [1, 23], we intentionally chose to base ourselves on these agents and the complex environments they had studied. Thus, we shied away from dealing with overly simplified settings, such as those with full information, single issues, or alternating turn based offers, and instead considered a complex problem with partial information, multi-attribute negotiations, and an unconstrained interaction protocol. In this section we detail the negotiation problem we considered, the state of the art KBAgent agent we based our study on, and the GENIUS environment used by the agent.

3.1 Problem Description

The negotiation environment we consider can be formally described as follows: We studied *bilateral* negotiation in which two agents negotiate to reach an agreement on conflicting issues. The negotiation can end either when (a) the negotiators reach a full or partial agreement, (b) one of the agents opts out (denoted as *OPT*), thus forcing the termination of the negotiation with a predefined opt-out outcome, or (c) a time limit is reached, that results in a predefined status-quo outcome (denoted as *SQ*).

The negotiations resolve around *multi-attribute* characteristics. There is a set of issues, denoted as I, and a finite set of values, O_i for each issue $i \in I$. *Partial* agreements are possible as subset of the issues contains $\perp \in O_i$. An offer is denoted as $(o) \in O$, and O is a finite set of values for all issues. The negotiations are sensitive to *time*. Time impacts the utilities of the negotiating parties, and is defined as $Time = \{0, \ldots, dl\}$, where dl is the deadline limit. Each agent is assigned a time cost which influences its utility as time passes. The time effect may be negative or positive with respect to the utility.

The negotiation *protocol* is fully flexible. As long as the negotiation has not terminated earlier, each side can propose a possible agreement, reject a previously offered agreement, opt-out of the negotiation, or communicate any general remark. In contrast to the model of alternating offers [22], each agent can perform up to $M > 0$ interactions with the opponent agent in each time period.

Last, we consider environments with *incomplete information*. That is, agents are not fully aware of the utility structure of their opponents. We assume that there is a finite set of utility structures which will be referred to as agent types. For example, one type might model a person that cares about the long-term implications of the agreement, while another type might model a person who cares only about the short term. Formally, we denote the possible types of the agents $Types = \{1, \ldots, k\}$. Given $l \in Types$, we refer to the utility of the agent of type l as u_l, and $u_l : \{O \cup \{SQ\} \cup \{OPT\}\} \times Time \rightarrow \mathscr{R}$. Each agent is fully aware of its own utility function, but it does not know the exact type of its negotiating partner.

3.2 The KBAgent

The state-of-the-art automated negotiator for the above environment is the *KBAgent* [23]. It has been shown that the *KBAgent* negotiates efficiently with people and achieves better utility values than other automated negotiators. Moreover, the *KBAgent* achieves significantly better agreements, in terms of individual utility, than the human counterparts playing the same role.

The main difference between the *KBAgent* and other agents is its inherent design, which builds a general opponent model. *KBAgent* utilizes past negotiation sessions of other agents as a knowledge base for the extraction of the likelihood of acceptance and offers which will be proposed by the other party. That data is used to determine which offers to propose and what offers to accept. One of its main advantages is that it can also work well with small databases of training data from previous negotiation sessions.

In order to generate an offer, the *KBAgent* creates a list ordered by the QOValue, which is an alternative to the Nash bargaining solution. The QOValue presents a qualitative evaluation of the possible offers in the domain based on the agent's utility and the probability of their acceptance by the other party.

$$\begin{aligned}
QOValue(\mathbf{o}) &= min\{\alpha_o, \beta_o\} \\
\alpha_o &= rank_a(\mathbf{o}) * lu_a(\mathbf{o}) \\
\beta_o &= [lu_a(\mathbf{o}) + lu_b(\mathbf{o})] * rank_b(\mathbf{o})
\end{aligned}$$

where *rank*() is the ranking of the offer based on the utility function (normalized to [0, 1]), and $lu()$ denotes the Luce number of an offer (see [20] for the exact definitions).

The first offer that is proposed is the one with the maximal QOValue. The other offers are picked from the ordered list based on the concession rate the *KBAgent* applies and are chosen with a decreasing QOValue for the agent and an increasing utility value for the other party. The concession rate is a list that is built offline based on previous experimental data in that domain. The list presents a single offer (or concession) per negotiation round. To decide which offers to accept, the *KBAgent* determines a time dependent threshold to decide whether to accept or reject an offer. In order to decide on the optimal threshold, the probabilities learned from the database of past negotiations are used.

3.3 The GENIUS Environment

When conducting computer based experiments with human participants, the interface design might have significant impact on the results as different design decisions might affect the subjects' behaviors [25]. For example, items placed at the top of a drop-down list have higher probability of being selected, default values might have framing effects, etc. Over the years, research on human-agent negotiation suffered from a comparative weakness when a new algorithm was compared to an old one but on a different interface.

To remedy this, an open source negotiation environment by the name of GENIUS was published in order to facilitate research on bilateral, multi-issue negotiations. The environment can be used both as an API for the development and testing of automated agents, and as a simulated environment to run tournaments and experiments in various negotiation scenarios (see Fig. 1). Moreover, since 2010 it has been used as the main tool of a novel annual Automated Negotiating Agent Competition (ANAC) [1].

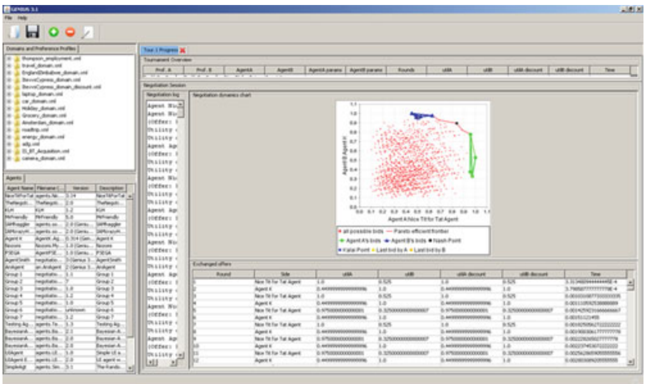

Fig. 1 The GENIUS environment

Fig. 2 GENIUS menu-based interface

The front-end interface for human based negotiation experiments is a dialog based graphical user interface. It contains various action buttons, pull-down boxes to select values for issues, and text areas to display information. See Fig. 2 for an example. We have used exactly the same interface in terms of its look&feel, but replaced the menus with a single text box for the chat area that will be used to pass messages between the negotiating parties (Fig. 3).

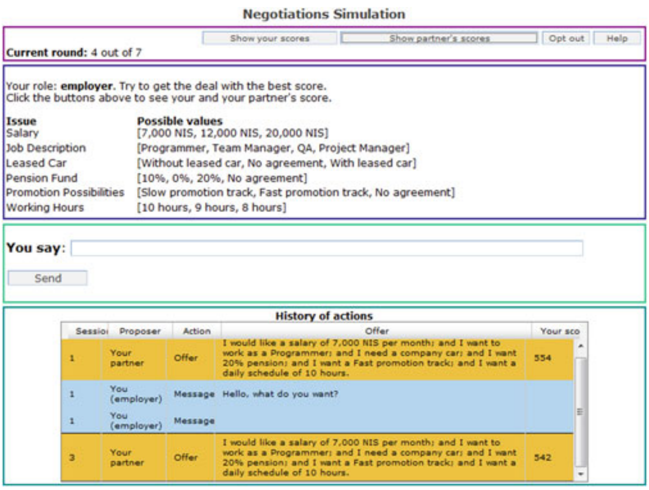

Fig. 3 GENIUS chat-based interface

3.4 Wizard of Oz

The main goal of our research is to understand whether the constraints of the menu-based interface affects the nature of agreements produced by a state-of-the-art negotiating agent. Stated differently, *we would like to check whether an automated agent developed in a menu-based negotiation environment, will be as effective in a chat-based environment.* Intuitively, it is not easy to say if there is any relationship between the negotiation interface and the negotiation algorithm that is used by the agent. But if such relationship does exist, it should be analyzed so that a new generation of negotiation strategies should incorporate these findings.

In order to study this point, we needed to translate each natural language sentence written in the chat box to an action object that can be accepted by agent. For example:

```
''I offer you a salary of 12,000''
```

should be translated to an object of the form:

```
Offer(Salary=12000)
```

Sentences in natural language might be *ambiguous*. For example, the sentence:

```
''Can you agree to work for 12,000?''
```

can be interpreted in at least two different ways. The first interpretation is a simple query to gather information regarding whether the candidate will agree to work for that salary. An answer to that query will reflect the willingness to accept such a value for that issue in a future agreement:

```
Query(Salary=12000)
```

A second possible interpretation is an offer of that salary. In this case, the person is in fact proposing an offer for this issue, expects a response to this offer, and in fact wishes to conclude a partial agreement:

```
Offer(Salary=12000)
```

The problem of *ambiguity* within natural language is a well known challenge within the field of Natural Language Processing (e.g., [24]). Unfortunately, even state of the art approaches cannot deal with such ambiguity with absolute certainty. As a first step we sidestepped this problem by having people manually decode ambiguity in other people's chat statements. To do so we used the *Wizard of Oz* (WOZ) approach [10, 16].

In WOZ experiments, the users believe that they are interacting with an automated agent directly, but behind the scenes there is a human being that translates their messages to the language that the agent understands. For instance, given the above sentence "Can you agree to work for 12,000?", a human "Wizard" decides which of the possible interpretations is more likely, and sends the correct interpretation to the agent.

An advantage of the WOZ approach is that it allows us to separate the NLP component of the agent from its strategy, allowing us to focus on the negotiation

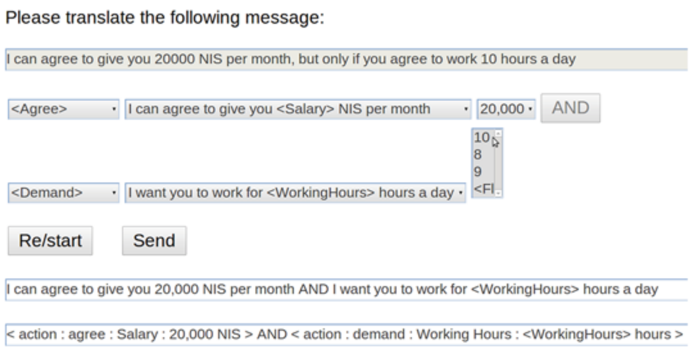

Fig. 4 The WOZ interface

algorithm and proceed to study the question at hand. A snapshot of the WOZ interface used in our experiments can be seen in Fig. 4.

4 Experiments

In order to properly evaluate the influence of natural language input on automated negotiation agents, we intentionally picked the *job candidate* domain used in previous research [20, 23]. In this domain, a negotiation takes place after a successful job interview between an employer and a job candidate. In the negotiation both the employer and the job candidate wish to formalize the hiring terms and conditions of the applicant. The issues are salary, job description, social benefits, promotion possibilities and working hours, for a total of 1296 possible agreements ($3 \times 4 \times 3 \times 4 \times 3 \times 3 = 1296$).

Salary This issue dictates the total net salary the applicant will receive per month. The possible values are {7000, 12000, 20000}.

Job description This issue describes the job description and responsibilities given to the job applicant. The possible values are {QA, programmer, team manager, project manager}.

Social benefits The social benefits are divided into two categories: company car and the percentage of the salary allocated, by the employer, to the candidate's pension funds. The possible values for a company car are {leased car, no leased car, no agreement}. The possible value for the percentage of the salary deposited in pension funds are {0 %, 10 %, 20 %, no agreement}.

Promotion possibilities This issue describes the commitment by the employer regarding the fast track for promotion for the job candidate. The possible values are {fast promotion track (2 years), slow promotion track (4 years), no agreement}

Working hours This issue describes the number of working hours required by the employee per day (not including over-time). The possible values are {8 h, 9 h, 10 h}.

The negotiation deadline is 30 min. If the sides do not reach an agreement by the end of the allocated time, the job interview ends with the candidate being hired with a standard contract, which cannot be renegotiated during the first year. This outcome is modeled for both agents as the status quo outcome. Each side can also opt-out of the negotiation if it feels that the prospects of reaching an agreement with the opponent are slim and it is impossible to negotiate anymore. Opting out by the employer entails the postponement of the project for which the candidate was interviewing, with the possible prospect of its cancellation and a considerable amount of expenses. Opting-out by the job candidate will make it very difficult for him to find another job, as the employer will spread his/her negative impression of the candidate to other CEOs of large companies. Time also has an impact on the negotiation. As time advances the candidate's utility decreases, as the employer's good impression has of the job candidate decreases. The employer's utility also decreases as the candidate becomes less motivated to work for the company.

To facilitate incomplete information there are 3 possible utility structures for each side, which model a long term candidate, short term candidate and compromising candidate. Here are their descriptions.[1]

Short-term orientation The candidate has a family to support and needs the job now. He puts a lot weight on working less hours and some weight on a higher salary. The candidate is indifferent regarding the exact job description, the social benefits and promotion track.

Long-term orientation The candidate currently has another job. He has experience in the field and believes in his ability to improve his status in the job market. The candidate puts more weight on a higher salary, a good job description and a fast promotion track.

Compromise orientation The candidate is willing to compromise in order to get a good job in a good company, believing that his contract can be improved after excelling in the job. The candidate wants a fast promotion track, while other social benefits are not as important.

4.1 Experiment Design

We extended the existing GENIUS negotiation system to include a newly developed chat interface for a WOZ based system using the previously described *KBAgent*. We then studied 32 human participants negotiate interactions with this agent playing the role of the employer (while the *KBAgent* played the job candidate), and additional

[1] The exact functions (and the complete domain) are available in the GENIUS framework that can be freely downloaded from the Internet.

30 human participants playing as the job candidate (while the *KBAgent* played the employer). All participants were students in three different academic institutions, and had different fields of studies. They were highly motivated to attain good scores as they received bonus points to their course grade which is a function of their final utility score in the session.

We then divided these students randomly so that half of each group played using the "old" menu based interface, and half used the newly developed chat interface. It is important to note that all other parts of the interfaces were **identical**; That is the only visible difference between them was the chat-box instead or pull-down boxes (see Fig. 3).

Prior to the start of the negotiation task, the people were given a full tutorial about the task at hand, the interface and the possible utility functions. A short test was issued to verify that the subjects understood the instructions and task at hand. The subjects did not know any details regarding the automated agent with which they were matched, or the fact that it was not a human player. The outcome of each negotiation task was either reaching a full agreement, opting out, or reaching the deadline.

In addition, following each session of the experiments (for both interfaces) we conducted a post-experiment questionnaire, in which the subjects had to score on a scale of 1 (lowest) to 5 (highest) the following questions:

- How happy are you with the negotiation's end result?
- Do you think that your partner was a computer program?
- Do you consider the end result to be fair?

4.2 Experiment Results

The main goal of the experiments was to check if there are differences in the agent's performance when playing against a human subject who is using a menu-based interface vs a chat-based interface.

Table 1 presents the average utility gained by the human players and the *KBAgent*. The standard deviation is written in parenthesis. We can see that the human players got on average similar utility scores when they were playing the employer, regardless of the interface that they were using. From the agent's perspective, we can see that the

Table 1 Results of menu versus chat negotiation experiments

	Agent as a job candidate		Agent as an employer	
	Menu-based	Chat-based	Menu-based	Chat-based
Avg. utility—human	398 ($\sigma = 44$)	385 ($\sigma = 41$)	332 ($\sigma = 81$)	397 ($\sigma = 78$)
Avg. utility—agent	**484** ($\sigma = 49$)	**438** ($\sigma = 65$)	**458** ($\sigma = 90$)	**351** ($\sigma = 78$)
Avg. nego. length (min)	10.06	14.30	14.07	23.12

Table 2 Results of post-experiment questionnaire

	Agent at job candidate		Agent as employer	
	Menu-based	Chat-based	Menu-based	Chat-based
Happiness	3.1	**3.95**	3.2	**3.61**
Computer program	3.89	3.95	3.06	3.76
Fairness	3.26	4	3.38	3.06

agent attained significantly higher scores when faced with partners who were using menu-based interfaces. When the agent was playing the job candidate, we can see a utility improvement from 438 when the human on the other side played using the chat interface to 484 when the human used a menu based interface ($p < 0.01$ on a *t-test*). When the agent was playing the employer, we can see similar improvement from 351 to 458 ($p < 0.05$ on *t-test*). That is, in both cases, the *KBAgent* performances decreased **significantly** when facing chat-based opponents. Similarly, we can see that the average session length was significantly longer using the chat interface—while the negotiation sessions using the menu interface were on average arounds 10 and 14 minutes long, chat interface sessions took on average 14 and 23 minutes long.

After studying the post-experiment questionnaire, the results of which are summarized in Table 2, we can see that with respect to the subjects' happiness level following the negotiation, users who were using the chat-based interface were significantly happier with the end results. Furthermore, we can see that subjects playing the job-candidate role with the chat-based interface tended to believe that their opponent is a computer, while the menu-based player did not hold such a strong belief. This can be easily explained as the chat-based seems a more natural interaction than the other. With respect to the fairness of the results, we cannot conclude anything as the different roles provide contradicting results.

4.3 Discussion

The above results were somewhat surprising to us as we would have expected exactly the opposite result. That is, we expected the negotiating agent to attain **lower** utility when playing against a user who is using the menu-based interface. This is because of the following reasons: first, forcing a person to use a preset number of choices in the menu requires her to focus on a limited number of possibilities making the task easier to compute. Second, within the menu-interface, drop-down lists existed for each of the limited choices, allowing the user to see the ordinal relationship of the values inside the list. This allows her to take smaller concession steps and greatly reduces the probability of errors. Last, when selecting the offer from a drop-down lists the utility of the offer is computed and presented automatically to the user, making the task even easier. Thus, even though the selected experimental domain is *not* a strict zero-sum scenario and some collaboration in the negotiation can be achieved, we assumed the person would do better in this case, and consequently, the agent would do worse as the person would achieve higher utility at the expense of the agent.

Thus, our results yielded two key implications: (1) automated negotiators developed for menu-based environment should be somehow **adapted** when migrated to chat-based environments. (2) Humans perceived the outcome of the negotiation session more **positively** when using chat even though their objective utility score remained the same.

Consequently, we focused on the following questions:

Why does the agent get significantly lower utility when playing against chat-based partners?

How should the next generation of negotiation agents be modified to address this shortcoming?

In addressing these questions, we studied various possible hypotheses for explaining our results. We first present two hypotheses which, while reasonable, do not adequately explain our results, and further develop a third hypothesis relating to the nature of people's offers which we believe will need to be addressed in the next generation of negotiation agents.

4.3.1 Rejected Hypotheses

When looking at the causes of the significant difference in utility, a first and intuitive conjuncture is that the discount factor in utility as the time progresses might be a prominent cause. This is an acceptable cause simply because inputting a natural language sentence takes more time than clicking on the dialog boxes. Specifically, Table 2 shows that chat-based sessions takes another 4 minutes on average for the job candidate, which amounts to ≈ -12 utility points, and 8 minutes on average for the employer which amounts to ≈ -24 utility points.

However, after analytically adding the utility lost due to time discount factor to both groups, the results remains significantly better for when playing against menu players. Specifically, when the agent was playing the job candidate, an average utility of 524.45 ($\sigma = 32$) against menu players, and 496.65 against chat players ($\sigma = 50$), $p < 0.05$ with the *two-tailed t-test*. Similar results were shown on the employer role.

Next, we looked at the time that was spent in the chat experiments due to WOZ translation. This represents the time it took the human behind the system to translate message from natural language to the agent actions model and vice-versa. It appears that on average there were ≈ 274 translation seconds in the chat-based experiments. This amounts to the additional 4 minutes from the previous hypothesis, and correcting them still did not resolve the significant utility advantage when playing against menu-based players.

4.3.2 Accepted Hypothesis—Percentage of Partial Offers

Another interesting observation from the above experiment was that chat-based users sent a higher number of partial agreements than the menu-based users. Specifically, on average the chat-based users sent approximately 2.4 partial offers per session, which amounts to around 40 % of their total offers. The menu-based users rarely offer partial agreements even though the interface does not constrain them from doing so, and the instructions explicitly discuss this possibility. In addition, our preliminary experiments agree with the literature as they show that when humans negotiate with humans, they tend to negotiate on one issue or a small group of issues at a time, agree on them, and then move to other issues. This is different than the usual mode of negotiation, where all issues are discussed at once.

In order to verify this claim we conducted an additional set of experiments in which we did not allow users to send partial offers (unless of course using the specific value of "no agreement" in the minor issues). We did so by issuing a message saying "I prefer to discuss offers with all 6 issues" whenever a partial offer had been sent. Besides that message, we followed exactly the same experiment design as before (the results refer to the agent playing the employer role).

The experiment was conducted in a similar manner and included 24 participants: 12 played with the menu interface and 12 with the chat interface. The results are depicted in Table 3, and they verify this hypothesis. We found that there is **no** statistically-significant difference between the average utility gained by the agent when playing against these two groups. In most cases following the presentation of a partial offer and the consequent system message, no further partial offers were issued in that session.

We continued with analyzing the post-experiment questionnaire (see Table 4) and now, to our surprise, we did not see any *significant* difference in the groups perception of fairness (3.5 vs. 3.9), or overall happiness with the outcome (3.3 vs. 3.4).

When negotiation is conducted using a chat interface, several additional problems arise, such as dialog manager and context resolution. For instance, the following sentence that was sent by a chat user:

```
''I suggest you work 9 h as a QA.''
```

Table 3 Results—negotiation without partial offers

	Menu-based	Chat-based
Avg. utility—human	397 ($\sigma = 39$)	373 ($\sigma = 51$)
Avg. utility—Agent	458 ($\sigma = 82$)	414 ($\sigma = 94$)
Avg. nego. length	6.6	9.4

Table 4 Results without partial offers post-experiment questionnaire

	Menu-based	Chat-based
Happiness	3.3	3.4
Computer program	4	3.8
Fairness	3.5	3.9

Can be interpreted in two ways: a partial offer of the following form {Salary=QA, Hours=9}, or an adaptation of these issues with respect to a previously discussed offer, thus a complete offer with these two new values. Regardless of the interpretation, an automated negotiator that was built around menu-based interface will not have to deal with many partial offers that exist in chat-based negotiation. Therefore, it might be the case that the *KBAgent*'s strategy with respect to partial offers, or specifically its lack of strategy, hindered its performance.

5 Moving from WOZ to NLP

In this section we want to address the problem of how to move from Wizard-of-Oz implementation (in which a person is translating the natural language sentences behind the scenes) to pure Natural Language Processing (NLP) solution.

We constructed a natural language system based on a standard dialog system architecture [14], as described in Fig. 5. We illustrate the system with a running example from our experimental domain, where the human is an employer and the agent is a job-candidate, and they negotiate over the candidate's job conditions. Nonetheless, the system itself is general and can be applied to support chat in any system.

The natural language system is composed of several components. The **Natural Language Understander** (NLU) translates the human sentences from natural language to a set of *dialog acts* that represents the user intentions. We represent our dialog acts in the standard JSON format.[2] For example, the human utterance "I accept your salary offer, but only if you work for 10 h", is translated to a set of two dialog acts: [[{Accept:Salary}, {Offer:{Hours:10}}]]. The NLU is described in detail in Sects. 5.1 and 5.2.

The **Dialog Manager** (DM) has several responsibilities:

(1) It interprets the human dialog acts based on the current dialog state. For example, it interprets the dialog act {Accept:Salary} based on the salary value in the most recent offer made by the agent, and converts it to an explicit Offer.
(2) It responds to human dialog acts that are not directly related to negotiation, such as greetings and questions.
(3) It notifies the agent when the human dialog acts are related to negotiation. For example, if one of the human's dialog acts in an offer, then the DM sends a "Received-Offer" notification, and if the human has accepted a full offer, the DM sends a "Received-Accept" notification.
(4) It controls the timing of conversation. For example, if the human hasn't done anything in a pre-specified time interval (e.g., 25 s), then the DM asks the agent to make an action, e.g., repeat the previous offer or make a new offer.

---

[2] www.json.org.

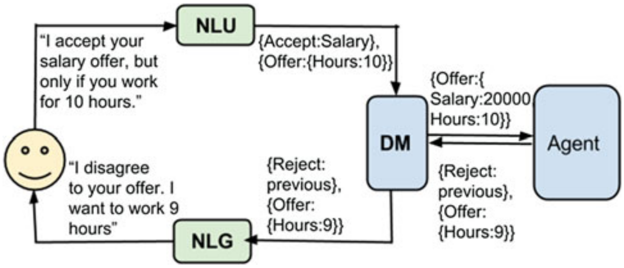

Fig. 5 Dialog system architecture. Example starts at the top-left corner

(5) It receives commands from the agent, and translates them to dialog acts. For example, if the agent issues a Send-Reject command, then the DM creates the dialog act {Reject:previous}.

The **Natural Language Generator** (NLG) translates the set of dialog acts, created by the DM, to a single natural language sentence, that is sent to the human. Our NLG works in cooperation with our NLU in order to create human-like sentences, as we describe in detail in Sect. 5.3.

5.1 Natural Language Understander (NLU)

Our NLU component is a **multi-label classifier** (MLC)—a classifier that returns a set of zero or more labels for each input sample. The set of possible labels is the set of dialog acts recognized by our DM, whose total number is 58. They have a hierarchical structure, for example: {Offer:{Salary:20000}} and {Offer:{Hours:9}} are two different dialog acts. The top level of the hierarchy contains 8 different labels: {Offer, Accept, Reject, Append, Insist, Query, Quit, Greet}. In order to take advantage of the hierarchical structure of the dialog acts, we used the HOMER approach (Hierarchy Of Multi-label classifiERs, [26]). In this approach, there is a different MLC for each level of the hierarchy. The input sentence is first sent to the top-level MLC, which returns a subset of the top-level labels, e.g., {Offer, Query}. Then the sentence is sent in parallel to all relevant second-level MLCs, e.g., the Offer MLC and the Query MLC. The Offer MLC returns a set of second-level labels from the set relevant to Offer (i.e., Salary, Hours, etc.), and the MLC for Query returns a set of second-level labels from the set relevant to Query. This process continues until the leaves of the hierarchy are reached. Then the replies of all MLCs are combined to produce the final set of dialog acts.

For the MLCs in each node of the HOMER, we used the One-versus-All approach: each MLC is a collection of binary classifiers, one for each label. For each input sentence, it runs each binary classifier in turn, and returns the set of labels whose classifier returned "true". As the base binary classifier, we used Modified Balanced Winnow [5]—a classifier that supports online training and real-time classification.[3]

An input sentence goes through several pre-processing components before it arrives at the MLC. The **normalizer** converts numbers and other common phrases in the input sentence to canonical format. The **splitter** splits the sentence around punctuation marks and creates several sub-sentences. We found out that this simple heuristic greatly improves the performance of the MLC. The **feature extractor** creates a feature vector from each sub-sentence. As features, we use unigrams and bigrams (pairs of adjacent words).[4] As feature values we use the standard TF/IDF metric. The resulting feature vectors are the inputs to the MLC.

5.2 *Development and Training*

As a first step in adding natural language capabilities, we manually wrote a single natural language sentence for each dialog act supported by the agent. This facilitated the coordination between the team working on the agent and the team tagging the training data, and made sure they both understand the negotiation acts in the same way. We also used these sentences as an initial training set for the multi-label classifier (MLC).

Using this initial NLU component, we let our agent speak with students and Amazon Mechanical Turk workers. During these preliminary experiments, one of the developers acted as a "wizard-of-oz": through a web-based GUI, he viewed each set of dialog acts produced by the NLU component, and could edit it before it is sent to the DM. He could also immediately train the classifier with each new sentence, thanks to its fast training abilities. During the online learning process, the sentence-level accuracy of the NLU component improved from 18 % (with only the initial 58 manually-written sentences) to 72 % (with 775 tagged sentences).[5]

The total time spent by the Wizard-of-Oz was about 5-10 hours. This means that it is relatively cheap to adapt the system to new negotiation domains. A possible line of future work is to use the manually-written natural language sentences, which

---

[3] The state-of-the-art in NLU for dialog systems is sequence classification [12]. We decided against this option because it requires too much labeling effort: while in multi-label classification you only need to label each sentence, in sequence classification you must label each **fraction** of a sentence. After deciding to use multilabel classifiers, we checked various approaches to multi-label classification [21] and various kinds of base binary classifiers. We found out that the combination of HOMER with Modified Balanced Winnow, described above, had the best performance in terms of both classification accuracy and run-time.

[4] We tried more sophisticated features, such as pairs of non-adjacent words, but this didn't improve performance.

[5] Sentence-level accuracy is the number of sentences whose classification was exactly correct (i.e., the set of dialog acts returned by the MLC is identical to the correct set), divided by the total number of sentences. The 72 % accuracy was calculated using 5-fold cross-validation on the set of 775 tagged sentences. Sentence-level accuracy is the strictest possible performance measure. In other measures, such as precision, recall or F1, the performance of our NLU was higher

are a natural part of the development process, to build an NLU component with no training at all. This can be done using a textual inference engine, that can tell, given two natural language sentences, whether one of the sentences can be inferred from another [9].

5.3 Natural Language Generator (NLG)

The NLG takes as input a set of dialog acts produced by the DM, and returns a natural language sentence that is sent to the human. Usually, NLGs are based on manually-written templates. In contrast, our NLG uses the training data of the NLU, in reverse direction. For each dialog act, the NLG asks the NLU for a sentence tagged with exactly this dialog act, and combines the received sentences to a single output sentence. This approach has several advantages, which we exemplify with several actual examples from our experiments:

(1) The agent's replies are versatile, even when the strategy demands that it repeats the same offer again and again. For example: the agent says "I would like to work at 20,000", and 25 s later, "I need to make 20,000".
(2) The agent's replies are human-like. They even contain spelling and grammar mistakes that occur naturally in chat conversations between humans.
(3) Some of the agent's replies contain reasoning and argumentation. For example: "i would like a 20000 salary. this is mandatory to me to have a good salary as i believe working conditions affect directly my effectiveness" (sic).
(4) The agent continuously learns new ways to express itself, during the online learning process of the NLU.

5.4 Evaluating the NLU Component

To analyze the impact of the NLU in general, we studied how the KBAgent's performance was impacted by its NLU. To aid in the collection of data, we hired 42 workers from Amazon Turk to participate in this experiment with 21 participants interacting with the KBAgent with the NLU, and 21 people using a "Wizard-of-Oz"

Table 5 The NLU impact on the KBAgent (playing the job candidate role)

Interface	AgentScore	Participants	Time to reach agreement (in Sec)	Fairness	Happiness
NLU	$468.71, \sigma = 62.68$	21	$767.62, \sigma = 326.62$	$3, \sigma = 1.21$	$2.7, \sigma = 1.08$
Wizard of Oz	$481.47, \sigma = 59.21$	21	$669.76, \sigma = 343.614$	$3.3, \sigma = 1.28$	$3.3, \sigma = 0.86$

approach where a person manually translated the chat messages to a language that the KBAgent could understand without the NLU. The results, presented in Table 5, show no significant difference in the performance of the agent between the NLU and the WOZ setting. This indicates that the main problem that needs to be addressed in the future is improving the strategy of the agent, since the effect of the strategy on the agent utility is much more significant than the effect of NLU mistakes.

6 Conclusions

This paper takes the first step towards automated negotiation in natural language interfaces. Before tackling the complex problems of NLP and Dialog management, we studied how the current state-of-the-art automated negotiator would perform when paired against chat-based interface. We discovered that the automated negotiation algorithm did not transfer well to more natural forms of conversation. Simply adding a chat-based interface to the existing agent yielded agreements that were significantly *worse* than agreements based on the menu-based interface. In an additional experiment we isolated the reason for the algorithm's inability to cope with partial agreements as the main cause for its decreased performance.

Next, we developed a fully working prototype of an Natural Language Processing (NLP) system, that achieved a 72 % classification rate following an online training session. Nevertheless, even with the state-of-the-art NLP system, the negotiation agent still underperformed with respect to an agent whose sentences were translated by a human.

We conclude that future negotiation algorithms for chat environments and other natural interfaces will need to take different strategies from those used by current negotiation agents [4, 13, 18, 20, 23]. While these state of the art agents attempt to find successful agreements on all issues simultaneously, our findings strongly suggest that future agents will instead need to take an issue-by-issue algorithm towards negotiations, or explicitly form partial agreements with people. We are currently studying how this finding can be implemented, and encourage other researchers to do the same.

References

1. Baarslag, T., Fujita, K., Gerding, E.H., Hindriks, K., Ito, T., Jennings, N.R., Jonker, C., Kraus, S., Lin, R., Robu, V., Williams, C.R.: Evaluating practical negotiating agents: results and analysis of the 2011 international competition. Artif. Intell. **198**, 73–103 (2013)
2. Bac, M., Raff, H.: Issue-by-issue negotiations: the role of information and time preference. Games Econ. Behav. **13**(1), 125–134 (1996)
3. Busch, L.-A., Horstmann, I.: A comment on issue-by-issue negotiations. Games Econ. Behav. **19**(1), 144–148 (1997)

4. Byde, A., Yearworth, M., Chen, K.-Y., Bartolini, C.: Aut ONA: a system for automated multiple 1–1 negotiation. In: Proceedings of the 2003 IEEE International Conference on Electronic Commerce (CEC), pp. 59–67 (2003)
5. Carvalho, V.R., Cohen, W.W.: Single-pass online learning: performance, voting schemes and online feature selection. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD'06, pp. 548–553. ACM, New York (2006)
6. Chen, MK.: Agendas in multi-issue bargaining: when to sweat the small stuff. Technical report, Harvard Department of Economics, Cambridge, November 2002
7. Coen, M.H.: Design principles for intelligent environments. In: AAAI/IAAI, pp. 547–554 (1998)
8. Cohen, P.R.: The role of natural language in a multimodal interface. In: Proceedings of the 5th Annual ACM Symposium on User Interface Software and Technology, UIST'92, pp. 143–149. ACM, New York (1992)
9. Dagan, I., Roth, D., Sammons, M., Zanzotto, F.M.: Recognizing textual entailment: models and applications. Synth. Lect. Hum. Lang. Technol. **6**(4), 1–220 (2013)
10. Dahlbäck, N., Jönsson, A., Ahrenberg, L.: Wizard of Oz studies: why and how. In: Proceedings of the 1st International Conference on Intelligent User Interfaces, IUI'93, pp. 193–200. ACM, New York (1993)
11. Gal, Y., Kraus, S., Gelfand, M., Khashan, H., Salmon, E.: An adaptive agent for negotiating with people in different cultures. ACM TIST **3**(1), 8 (2011)
12. Hahn, S., Dinarelli, M., Raymond, C., Lefevre, F., Lehnen, P., de Mori, Renato, Moschitti, A., Ney, H., Riccardi, G.: Comparing Stochastic Approaches to Spoken Language Understanding in Multiple Languages. IEEE Transactions on Audio, Speech, and Language Processing, vol.19, no.6, pp.1569–1583 (2011). http://dx.doi.org/10.1109/tasl.2010.2093520
13. Jonker, C.M., Robu, V., Treur, J.: An agent architecture for multi-attribute negotiation using incomplete preference information. Auton. Agents Multi-Agent Syst. **15**(2), 221–252 (2007)
14. Jurafsky, D., Martin, James H.: Speech and Language Processing, 2nd edn, Prentice Hall (2008). ISBN:0131873210
15. Katz, R., Kraus, S.: Efficient agents for cliff edge environments with a large set of decision options. In: Proceedings of the 5th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS), pp. 697–704 (2006)
16. Kelley, J.F.: An empirical methodology for writing user-friendly natural language computer applications. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI'83, pp. 193–196. ACM, New York (1983)
17. Kenny, P., Hartholt, A., Gratch, J., Swartout, W., Traum, D., Marsella, S., Piepol, D.: Building interactive virtual humans for training environments. In: Proceedings of Interservice/Industry Training, Simulation and Education Conference (I/ITSEC) (2007)
18. Kraus, S., Lehmann, D.: Designing and building a negotiating automated agent. Comput. Intell. **11**(1), 132–171 (1995)
19. Lin, R., Kraus, S.: Can automated agents proficiently negotiate with humans? CACM **53**(1), 78–88 (2010)
20. Lin, R., Kraus, S., Wilkenfeld, J., Barry, J.: Negotiating with bounded rational agents in environments with incomplete information using an automated agent. Artif. Intell. **172**(6–7), 823–851 (2008)
21. Madjarov, G., Kocev, D., Gjorgjevikj, D., Džeroski, S.: An extensive experimental comparison of methods for multi-label learning. Pattern Recogn. **45**(9), 3084–3104 (2012). http://dx.doi.org/10.1016/j.patcog.2012.03.004
22. Osborne, M.J., Rubinstein, A.: A Course In Game Theory. MIT Press, Cambridge (1994)
23. Oshrat, Y., Lin, R., Kraus, S.: Facing the challenge of human-agent negotiations via effective general opponent modeling. In: Proceedings of the 8th International Conference on Autonomous Agents and Multiagent Systems (AAMAS) (2009)
24. Pease, A., Colton, S., Smaill, A., Lee, J.: Semantic negotiation: Modelling ambiguity in dialogue. In: Proceedings of Edilog 2002, the 6th Workshop on the Semantics and Pragmatics of Dialogue, Edinburgh, UK (2002)

25. Shneiderman, B., Plaisant, C.: Designing the User Interface: Strategies for Effective Human-Computer Interaction, 4th edn. Pearson Addison Wesley, Boston (2004)
26. Tsoumakas, G., Katakis, I., Vlahavas, I.: Effective and efficient multilabel classification in domains with large number of labels. In: Proceedings ECML/PKDD 2008 Workshop on Mining Multidimensional Data (MMD'08) (2008)