Learning Trade Negotiation Policies in Strategic Conversation

Simon Keizer, Heriberto Cuayáhuitl, Oliver Lemon
Interaction Lab
School of Mathematical and Computer Sciences
Heriot-Watt University, Edinburgh (UK)
{s.keizer|h.cuayahuitl|o.lemon}@hw.ac.uk

Abstract

This paper presents a new, data-driven method for learning trade negotiation policies in strategic, non-cooperative dialogue. The learned policies focus on selecting trade offers in the context of playing the game *Settlers of Catan*. First, a supervised learning approach is used to train a Random Forest model for ranking trade offers, using data from an annotated corpus of humans playing the game. Second, a Reinforcement Learning agent for trade offer selection is trained by playing games against three artificial players that use the human-like, data-driven Random Forest offer selection model. In a comparative evaluation our trained models significantly outperform an expert hand-crafted negotiation baseline as well as the supervised learning negotiator. We therefore show that rather than hand-crafting rule-based heuristics for trading, a more successful approach is to train policies from human trading dialogue data.

1 Introduction

Non-cooperative dialogues, where agents act to satisfy their own goals rather than those of other participants, are of practical and theoretical interest (Georgila and Traum, 2011; Efstathiou and Lemon, 2014a). The game-theoretic underpinnings of non-Gricean behaviour have also been investigated (Asher and Lascarides, 2008).

In practice, it may be useful for dialogue agents not to be fully cooperative when trying to gather information from humans, or when trying to persuade, or for believable characters in video games and educational AI (Georgila and Traum, 2011; Shim and Arkin, 2013). Negotiation, where hiding information (and even lying) can be advantageous, is also of interest (Traum, 2008).

Previous work on Reinforcement Learning (RL) in non-cooperative dialogue (Efstathiou and Lemon, 2014a) focused on a small 2-player trading problem with 3 resource types, and without using any real human dialogue data. This work showed that explicit manipulation moves (e.g. "I really need sheep") can be used to win when playing against adversaries who are gullible (i.e. they believe such statements) but also against adversaries who can detect manipulation and can punish the player for being manipulative (Efstathiou and Lemon, 2014b).

In this paper, we apply RL to a much larger trading problem in the context of the board game Settlers of Catan, involving 4 players and 5 resource types to trade with. Furthermore, the trading polices are optimised by playing against adversaries that have been trained on a corpus of trading conversations between humans playing the game.

2 Task domain and dialogue data

Settlers of Catan is a complex multi-player board game[1]; the board is a map consisting of hexes of different types: hills, mountains, meadows, fields and forests (see the central part of Fig. 1). The objective of the game is for the players to build roads, settlements and cities on the map, paid for by combinations of resources of different types: clay, ore, sheep, wheat and wood, which are obtained according to the numbers on the hexes adjacent to which a player has a settlement or city after the roll of a pair of dice at each player's turn. In addition, players can negotiate trades with each other in order to obtain the resources they desire. Players can also buy Development Cards, randomly drawn from a stack of different kinds of cards. Players earn Victory Points (VPs) for their settlements (1 VP each) and cities (2 VPs each), and for having the Longest Road (at least 5 consecutive

[1] www.catan.com

roads; 2 VPs) or the Largest Army (by playing at least 3 Knight development cards; 2 VPs). The first player to have 10 VPs wins the game.

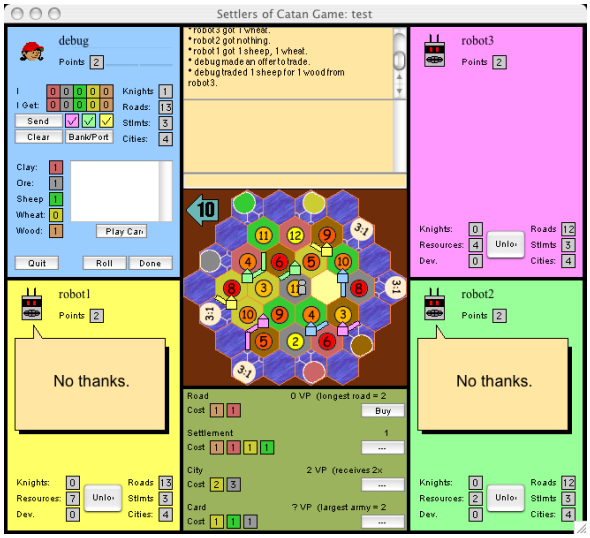

Figure 1: Graphical interface of the online game version of Settlers of Catan, showing in the central area the board itself, in each corner information about one of the four players, and in the top middle area a chat interface with the game history and a text box for chat negotiation.

In our work we are interested in strategic conversation and therefore focus on the trade negotiation aspect of the game. The negotiation models we build take as input a so-called Build Plan (BP), which can be 1) to build a settlement, 2) to build a city, or 3) to buy a development card. Based on the player's current resources and BP, s/he selects trade offers to the other players in order to obtain the resources to realise the BP.

2.1 The jSettlers implementation

For testing and evaluating our models for trade negotiation, we use the jSettlers[2] open source implementation of the game (Thomas, 2003). The environment is a client-server system supporting humans playing against each other via a graphical interface, but also has artificial players. These artificial players use complex heuristics for both the board play (including deciding when and where to build roads, settlements and cities) and negotiation with other players. Our models are integrated in the environment as replacements of the built-in negotiators.

[2] `jsettlers2.sourceforge.net`

2.2 Human data

With the aim of studying strategic conversations, a corpus of on-line trading chats between humans playing Settlers of Catan was collected (Afantenos et al., 2012b; Afantenos et al., 2012a). The jSettlers implementation of the game was modified to let players use a chat interface to engage in conversations with each other, involving the negotiation of trades in particular. Table 1 shows an annotated trade negotiation chat from the corpus between players W, T, and G; in this dialogue, a trade is agreed between W and G, where W gives G a clay in exchange for an ore.

For the supervised learning experiments, which will be described in Section 3, we used a set of 32 logged and annotated games, corresponding to 2512 trading negotiation events (training instances) denoted as $D=\{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$, where $\mathbf{x}_i$ are vectors of features and y_i are class labels (i.e. giveable resources). An example trading negotiation in the game of Settlers of Catan in natural language is *"I'll give anyone sheep for clay"*, which can be represented as follows, including the agent's available resources: $Give(Sheep, all) \wedge Receive(Clay, all) \wedge Resources(\, clay = 0,\, ore = 0,\, sheep = 4,\, wheat = 1,\, wood = 0\,) \wedge Buildups(\, roads = 2,\, settlements = 0,\, cities = 0\,)$.

From this example, we extract the training instance $y_i = sheep$ and $\mathbf{x}_i = \{\, clay = 0,\, ore = 0,\, sheep = 4,\, wheat = 1,\, wood = 0, roads = 2, settlements = 0, cities = 0\}$. During each training situation, there is a finite set of possible trading negotiation events. Choosing the best trading offer can be seen as a *ranking task*, where we focus on computing a score representing the importance of each trading offer (similar to the one above)—from which we choose the trade with the highest score. While the model in Section 3 computes the most human-like trading negotiation, the one in Section 4 computes the one with the highest cumulative reward. Other applicable learning approaches are discussed in (Cuayáhuitl et al., 2013).

Note that this corpus was not collected with expert or especially experienced players of the game. We could therefore expect different and more successful trading behaviour to be found in a corpus of expert games.

Speaker	Utterance	Game act	Surface act	Addressee	Resource
W	*can i get an ore?*	Offer	Request	all	Receivable(ore,1)
T	*nope*	Refusal	Assertion	W	
G	*what for.. :D*	Counteroffer	Question	W	
W	*a wheat?*	Offer	Question	G	Givable(wheat,1)
G	*i have a bounty crop*	Refusal	Assertion	W	
W	*how about a wood then?*	Counteroffer	Question	G	Givable(wood,1)
G	*clay or sheep are my primary desires*	Counteroffer	Request	W	Receivable((clay,?) OR (sheep,?))
W	*alright a clay*	Accept	Assertion	G	Givable(clay,1)
G	*ok!*	Accept	Assertion	W	

Table 1: Example trade negotiation chat.

3 Supervised learning

Here we cast trading in interactive board games as a statistical classification task, where we trained a Random Forest classifier using the features listed in Table 2. Our set of features includes the resources available (features 1-5), the built pieces ('buildups', features 6-8) with a default minimum of 0 and maximum value of 7, the receivable resources in binary form to reduce data sparsity (features 9-13), and the giveable resource contains the classes to predict (feature 14). This agent is trained using an ensemble of trees, which are used to vote for the class prediction at test time (Breiman, 2001; Hastie et al., 2009).

No.	Feature	Domain
1	hasClay	{0...7}
2	hasOre	{0...7}
3	hasSheep	{0...7}
4	hasWheat	{0...7}
5	hasWood	{0...7}
6	hasRoads	{0...7}
7	hasSettlements	{0...7}
8	hasCities	{0...7}
9	recClay	Binary
10	recOre	Binary
11	recSheep	Binary
12	recWheat	Binary
13	recWood	Binary
14	givable	{Clay, Ore, Sheep, Wheat, Wood}

Table 2: Feature set for predicting the offered resource in human-like trades.

We use probabilistic inference to compute the probability of a trade being generated by a human player, given their current resources. The probability distribution—also viewed as a ranking—of a set of trading negotiations is computed as:

$$P(givable|evidence) = \frac{1}{Z}\prod_{t\in T} P_t(givable|evidence),$$

where $givable$ refers to the predicted class, $evidence$ refers to observed features 1-13, $P_t(.|.)$ is the posterior distribution of the t-th tree, and Z is a normalisation constant—see (Criminisi et al., 2012) for further details. Assuming that Y is a set of trades at a particular point in time in the game, extracting the most human-like trade is defined as:

$$y^* = \arg\max_{y\in Y} Pr(\, givable = y \,|\, evidence\,).$$

Our evaluation metrics include classification accuracy and precision-recall. The former is computed as Accuracy $= \frac{t_p+t_n}{t_p+t_n+f_p+f_n}$ and the latter as F-measure $= \frac{2\times precision\times recall}{precision+recall}$, where $precision$=$\frac{t_p}{t_p+f_p}$, $recall$=$\frac{t_p}{t_p+f_n}$, t_p=true positives, t_n=true negatives, f_p=false positives, and f_n=false negatives.

Using the data described in Section 2.2 and an ensemble of 100 trees using the features listed in Table 2[3], we obtained a classification accuracy of 65.7% based on a 10-fold cross validation. A break-down of results per prediction class is shown in Table 3. It can be noted that predicting human trades is a difficult task, and that our Random Forest substantially outperforms a majority baseline. This result motivates future work on learning agents with improved predictive power. See (Cuayáhuitl et al., 2015) for further details.

[3]The feature set listed in Table 2 reported the best results when compared to other representations—in this paper we only report our best classifier. Other feature sets that we explored include smaller domains (only binary features), larger domains (only non-binary features) smaller and larger sets of features, multiple givables, among others.

Class	Precision	Recall	F-Measure
Clay	0.697	0.686	0.691
Ore	0.644	0.587	0.614
Sheep	0.684	0.751	0.716
Wheat	0.668	0.628	0.648
Wood	0.588	0.613	0.600
All	**0.657**	**0.657**	**0.656**
Majority	0.055	0.234	0.089

Table 3: Classification results of trades from human players in Settlers of Catan.

4 Reinforcement learning

Although some work has been done on using Reinforcement Learning (RL) in the context of Settlers of Catan (Pfeiffer, 2004), this work focused on learning high-level behaviours for playing the entire game. We however focus on the trade negotiation dialogue aspect of the game, which can be seen as strategic conversations in which the participants try to agree on an exchange of resources. In these conversations, each participant has their own personal goals, which inherently conflict with the other participants' goals. Strategies for trade negotiations in Settlers are very complex, given the large state and action space and the fact that the other players' resources are not observable and can only be partially inferred from the trade dialogues that take place. As a first attempt at using RL to optimise such strategies in the full context of the Settlers game, we have designed a Markov Decision Process (MDP) model in which only the given build plan (BP) and the player's own resources make up the observable state, and the actions correspond to 1-for-1 trade offers, addressed to all other players.

At every time step t, an MDP agent perceives the state of the environment $s_t \in S$, selects an action $a_t \in A$, receives a reward signal $r(s_t, a_t) \in \mathbb{R}$ and transitions to a new state $s_{t+1} \in S$. The goal is to find an optimal policy $\pi : S \rightarrow A$, that maximises the cumulative (discounted) reward over time (Sutton and Barto, 1998).

4.1 States and actions

The states of the MDP model are represented via 6 state features (see Table 4): the first feature is the build plan that the negotiator should target, the other features represent how many units of each resource the agent has. The set of possible values for the resource features is restricted by grouping together all quantities of over 3 units. This results in a state space of $3 * 4^5 = 3072$ states. The action space includes all 1-for-1 offers, plus the no-offer action, making $5 * 4 + 1 = 21$ actions (see Table 5). In contrast to the supervised learning negotiator described in Section 3, which re-ranks legal trade offers, the MDP negotiator can offer resources that it does not in fact have, which might be an effective strategy of misleading opponents.

Feature	Values
BuildPlan	{ settlement, city, dev-card }
NumClay	{ 0, 1, 2, at least 3 }
NumOre	{ 0, 1, 2, at least 3 }
NumSheep	{ 0, 1, 2, at least 3 }
NumWheat	{ 0, 1, 2, at least 3 }
NumWood	{ 0, 1, 2, at least 3 }

Table 4: State features for the MDP trade negotiation model.

Action index	Trade offer
0	No offer
1	1 clay for 1 ore
2	1 clay for 1 sheep
⋮	⋮
19	1 wood for 1 sheep
20	1 wood for 1 wheat

Table 5: MDP action space of all 1-for-1 trade offers.

4.2 Reward Function

The reward function that guides the optimisation consists of small penalties for each offer made, a slightly larger penalty if the offer is illegal, a penalty resp. reward for an offer that is rejected by all resp. accepted by at least one of the other players, and a bigger reward for achieving the build plan, e.g. placing a city on the board (see Table 6).

4.3 Adversaries

In order to optimise and test our MDP negotiator agent, we use four different adversaries to play against during training and evaluation. The four

Type of reward	Value
an *offer* was made	-1
an *illegal offer* was made	-3
the offered *trade succeeded*	5
the offered *trade failed*	-5
a *settlement* was built	25
a *city* was built	50
a *development card* was acquired	20

Table 6: Reward function for optimising the MDP trade negotiation strategy.

types (see Table 7) arise from two types of jSettlers 'bots' (i.e. automated players), and two types of negotiators (HEU and SUP). The bots (BOT1 and BOT2) only differ in their building strategy, not their negotiation strategy. The baseline negotiation strategy (HEU) uses heuristics to filter and rank the list of legal trades and then select the top-ranked trade, whereas the supervised learning negotiator (SUP, see Section 3) uses data-driven ranking for trade selection. For more details about game strategies in jSettlers, see (Guhe and Lascarides, 2014).

Adversary	Build strategy	Negotiation strategy
BOT1-HEU	original jSettlers	heuristics baseline
BOT1-SUP	original jSettlers	supervised learning
BOT2-HEU	advanced jSettlers	heuristics baseline
BOT2-SUP	advanced jSettlers	supervised learning

Table 7: Description of negotiation adversaries for training and evaluation.

4.4 Training

Two different MDP policies were trained, both obtained while playing against three copies of one of the two supervised learning adversaries, BOT1-SUP and BOT2-SUP, described above (Settlers is normally a 4-player game). The resulting trained policies are indicated by prefixing the corresponding adversaries with "TRA-". The MDP policies are optimised using Monte Carlo Control (MCC), a basic RL algorithm which processes recorded $(state, action, reward)$ trajectories, updating estimates of the long-term cumulative reward for each state-action pair, stored in the so-called Q-function $Q(s, a) \in \mathbb{R}$. During training, an ϵ-greedy policy is used, i.e., the agent selects a random action with probability $\epsilon = 0.2$ and an action $a' = argmax_a\, Q(s, a)$ otherwise. This setting is used to balance exploitation of the current policy with exploration of the state-action space (Sutton and Barto, 1998).

Figure 2 shows the learning curve for the optimisation of our MDP negotiator playing against 3 copies of the BOT2-SUP adversary negotiator. Each point on the curve represents the result of an evaluation of the MDP over 10k games, played against three copies of the same BOT2-SUP player it was trained on. Figure 3 shows another learning curve, this time in terms of win rates. The 15% win rate at the beginning of training represents a policy that selects random offers (including many illegal ones). After 11k games, the policy reaches the level of 25% that is expected in a 4-player game if the negotiators are identical (indicated by the green horizontal line); after 14k games the policy starts to be stronger than the BOT2-SUP adversaries. The learning (in terms of average reward) seems to converge after about 21k games, but after that still slowly improves. After about 31k games no further improvement is made for another 10k games, so training was ended at 40k games. At that stage, the policy achieves a win rate of 28%.

5 Evaluation

After training the two MDP policies, an evaluation was carried out by letting each of them play 10k games against three copies of each of the four adversaries. Table 8 gives the results in terms of win rates, where each of the columns represents one of the four evaluation conditions.

The same evaluation was also carried out with the adversaries themselves (last two rows of Table 8), making sure that in all evaluations, the build strategy of all four agents was the same and only the negotiation strategy was different for the player being evaluated. Note that in the last two rows, the 'BOT1/2' prefix refers to the building strategy that matches the evaluation condition, i.e. BOT1 when playing against BOT1-HEU or BOT1-SUP, and BOT2 when playing against BOT2-HEU or BOT2-SUP.

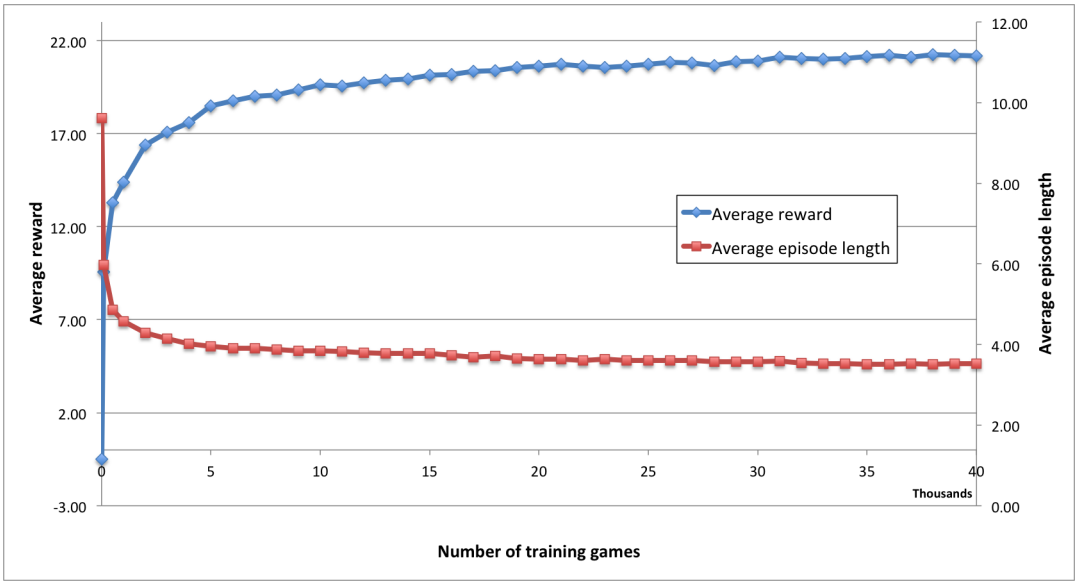

Figure 2: Learning curve in terms of average reward and episode length when training the MDP against the BOT2-SUP adversary.

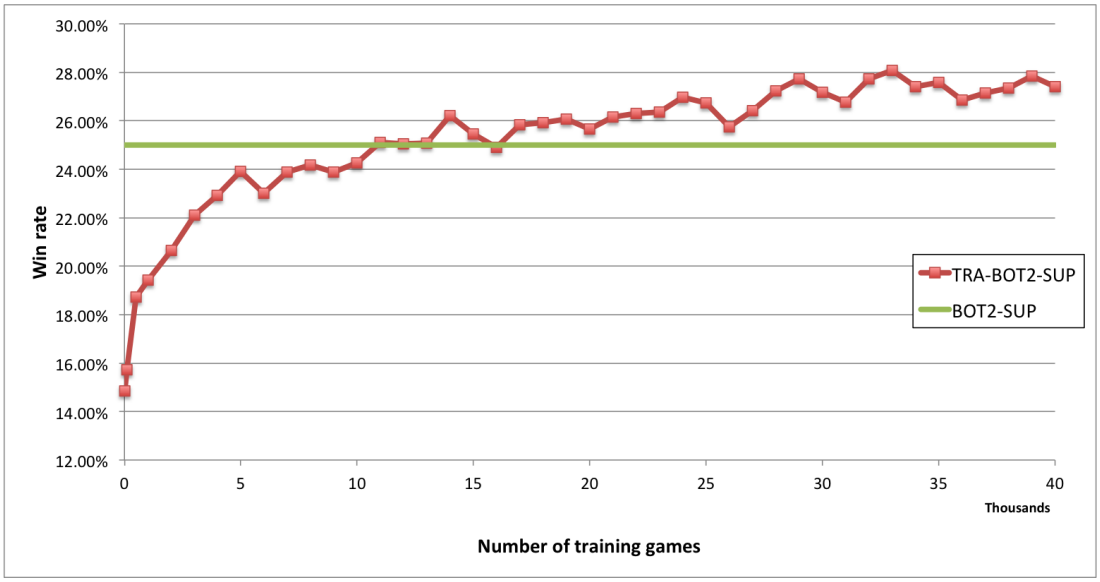

Figure 3: Learning curves in terms of win rates when training the MDP against the BOT2-SUP adversary.

	BOT1-HEU	**BOT2-HEU**	**BOT1-SUP**	**BOT2-SUP**
TRA-BOT1-SUP	**27.35%**	26.41%	**27.26%**	26.23%
TRA-BOT2-SUP	25.90%	**27.69%**	26.81%	**27.40%**
BOT1/2-SUP	24.07%	24.18%	(25%)	(25%)
BOT1/2-HEU	(25%)	(25%)	25.22%	24.99%

Table 8: Evaluation results in terms of win rates of the adversary and trained negotiators (each row corresponding to one negotiator) against the two baseline jSettlers bots (each column corresponding to three instances of an adversary negotiator). In the cases involving four identical players, the expected theoretical win rate of 25% is indicated between brackets.

Note that with 10k games, win rates between 24% and 26% are not statistically significantly different ($p < 0.01$) from the level of 25% that is expected when the four players are identical (Guhe and Lascarides, 2014).

Note also that in the four evaluations of the trained MDP policies, there can be a mismatch between train and test time in the build strategy used: e.g. when evaluating TRA-BOT2-SUP against BOT1-SUP, the trained negotiation strategy is run with the BOT1 build strategy (matching the three adversaries), but it was trained using the BOT2 build strategy. Finally, note that there can also be a mismatch between train and test time in the negotiation strategies of the adversaries.

In all four evaluations, the trained policies significantly outperform both the supervised learning and heuristic baselines. As expected, the policies where the build strategy was the same during training and evaluation obtain higher win rates, as do the ones where the adversary negotiation strategies were the same during training and evaluation. Even when either the build strategy or the adversary negotiation strategies were different between train and test time, the resulting win rates are significantly higher than those of the baselines, except for TRA-BOT2-SUP evaluated against BOT1-HEU, which gets a non-significant 25.9%.

5.1 Discussion

In our experiments we have shown that an MDP model for selecting trade offers can be optimised by interacting with artificial adversaries, and that the optimised model outperforms several baseline strategies in different conditions (i.e. when playing against opponents with different kinds of negotiation strategies).

A preliminary analysis of the trained policies showed that most of the time, trades are selected that one would intuitively expect, given the build plan and resources (e.g. the MDP agent will ask for a resource it needs to realise the given build plan). However, for a portion of the state space the agent seems to select trades that are less obvious, but must have a more indirect long-term game advantage in mind. This can be due to the fact that the negotiation dialogues are embedded in the context of complete games and are therefore not completely isolated from each other. Also, in a number of cases, the trained MDP agent offers resources it does not in fact have, see Table 9. For roughly a third of the states encountered during training, the policy selects an illegal trade offer (column 'PolicyIllegal'), suggesting that lying to the supervised learning adversaries can be advantageous, despite the penalty for illegal offers in the reward function (Table 6). However, many of these states occur quite rarely (making its Q-value estimates relatively inaccurate): in a test run of 1000 games, less than 2% of the turns where the policy is triggered, an illegal offer was selected (column 'IllegalOffers'). After re-evaluating the trained policies whilst restricting the output to legal offers only, no significant differences in terms of win rate were observed (27.75% resp. 27.94% for TRA-BOT1-SUP resp. TRA-BOT2-SUP, tested against adversaries matching the training conditions).

The current MDP model is relatively simple, in the sense that it only supports the selection of 1-for-1 trades, it only takes into account three of the possible build plans in jSettlers, and only knows about its own resources. We therefore plan extensions to both state and action spaces to further improve performance, e.g. by including the option to offer 2 units in the selection of trades, and including strategies for the additional build plans of obtaining the longest road and the largest army (each worth 2 VPs). Since the values of the different components of the reward function used in this paper are qualitatively intuitive but quantitatively somewhat arbitrary, further improvements might be achieved by using data from expert players to *learn* the reward function (Abbeel and Ng, 2004).

A more challenging future direction is to include information about the adversaries' resources and preferences, which are unobservable, but might be estimated based on the trades taking place and the adversaries' negotiation behaviour. This would require a POMDP-style approach (Kaelbling et al., 1998), which attempts to represent and exploit uncertain information. One step further is 'opponent modelling', in which the beliefs, goals, and preferences of the adversaries are modelled and exploited (Gmytrasiewicz and Doshi, 2005), which could be particularly important in non-cooperative dialogue modelling. We also aim to extend the range of actions with strategic conversational moves, for example to manipulate and exploit the adversaries' beliefs and behaviour, following (Efstathiou and Lemon, 2014a; Efstathiou and Lemon, 2014b).

	—POLICY—		—TEST 1000 GAMES—	
	StatesVisited	**PolicyIllegal**	**Turns**	**IllegalOffers**
TRA-BOT1-SUP	2593	29.1%	13,159	1.57%
TRA-BOT2-SUP	2288	32.5%	15,852	1.94%

Table 9: Illegal offer statistics for the learned policies. Column 'StatesVisited' gives the number of unique states encountered during training, column 'PolicyIllegal' gives the percentage of these states for which the policy outputs an illegal offer, and, based on a test run of 1000 games with one of the trained players against against three instances of the adversary matching the training conditions, columns 'Turns' and 'IllegalOffers' give the number of turns and in how many of them the policy selected an illegal offer.

6 Conclusion and future work

In this paper we have presented a new, data-driven method for learning trading negotiation policies in strategic, non-cooperative dialogue. We showed that rather than hand-crafting rule-based heuristics for trading, a more successful approach is to train policies from human trading dialogue data.

The experiments were carried out in the context of the board game *Settlers of Catan*, making use of an existing java implementation of the game, *jSettlers*. First, a supervised learning approach is used to build a Random Forest model for ranking legal trade offers, using an annotated corpus of data from non-expert humans playing the game. Evaluation results for the underlying classification model show significant improvements in terms of accuracy over a majority baseline (see Table 3). Second, an MDP model for trade offer selection is trained using Reinforcement Learning, by playing games against three instances of one of two possible adversaries, which only differ in their build strategy, but both use the Random Forest model for selecting trade offers. In contrast to the adversaries which always select legal trade offers, the MDP model is capable of offering resources that it does not in fact have (i.e. it can lie).

We evaluated our trained MDP policies, again by playing games against three instances of different adversaries. There are four types of evaluation conditions, arising from the adversaries' two possible build strategies and two possible negotiation strategies (one of them being the Random Forest negotiator). We assessed the performance of the policies by comparing their win rates with other player bots playing games under the same conditions. The results indicate that the trained MDPs achieved significantly higher win rates compared to the expected level of 25% that an agent would achieve if it was identical to the three opponents. More specifically, the policies outperform the two baseline negotiation strategies (an expert handcrafted jSettlers strategy and the human-like supervised learning strategy) in terms of win rate when playing against the same three adversaries. The trained MDP is robust in the sense that even when there was a mismatch between train and test time in the build strategy used or the adversary negotiation strategy, the MDP significantly outperforms the baselines in most cases (see Table 8).

Further improvements can be expected when extending both state and action spaces. For example, the current model only supports one-for-one trades, whereas the option to offer more units could make trades more likely to be successful. In addition, improvements in the supervised learning adversary models can also lead to better MDP performance. We also aim to include information about the opponents into the model, e.g. their resources, requiring a POMDP-style approach, since such information is not observable. Opponent modelling would also enable the selection of additional moves for strategic conversation (in addition to lying), such as manipulation moves. Furthermore, we plan to include the selection of other kinds of actions in trade negotiation dialogue that are not yet supported, such as responses to offers (accept, reject, or counter-offer) in our data-driven models. Finally, we aim to eventually test our trained negotiation strategies against humans, which requires modules for Natural Language Generation and Understanding (NLG and NLU), creating an end-to-end text-based dialogue system for playing Settlers of Catan.

Acknowledgments

The research leading to this work is supported by ERC grant 269427 (the STAC project).

References

Pieter Abbeel and Andrew Y. Ng. 2004. Apprenticeship learning via inverse reinforcement learning.

S. Afantenos, N. Asher, F. Benamara, A. Cadilhac, C. Dégremont, P. Denis, M. Guhe, S. Keizer, A. Lascarides, O. Lemon, P. Muller, S. Paul, V. Popescu, V. Rieser, and L. Vieu. 2012a. Developing a corpus of strategic conversation in the settlers of catan. In *Proc. 1st Workshop on Games and NLP*.

S. Afantenos, N. Asher, F. Benamara, A. Cadilhac, C Dégremont, P Denis, M Guhe, S. Keizer, A. Lascarides, O. Lemon, P. Muller, S. Paul, V. Popescu, V. Rieser, and L. Vieu. 2012b. Modelling strategic conversation: model, annotation design and corpus. In *Proc. SEMDIAL*.

N. Asher and A. Lascarides. 2008. Commitments, beliefs and intentions in dialogue. In *Proc. SEMDIAL*.

Leo Breiman. 2001. Random forests. *Machine Learning*, 45(1):5–32.

Antonio Criminisi, Jamie Shotton, and Ender Konukoglu. 2012. Decision forests: A unified framework for classification, regression, density estimation, manifold learning and semi-supervised learning. *Foundations and Trends in Computer Graphics and Vision*, 7(2-3):81–227.

Heriberto Cuayáhuitl, Martijn van Otterlo, Nina Dethlefs, and Lutz Frommberger. 2013. Machine learning for interactive systems and robots: A brief introduction. In *Proc. MLIS*.

Heriberto Cuayáhuitl, Simon Keizer, and Oliver Lemon. 2015. Learning to trade in strategic board games. In *IJCAI Workshop on Computer Games (IJCAI-CGW)*.

Ioannis Efstathiou and Oliver Lemon. 2014a. Learning non-cooperative dialogue behaviours. In *Proc. SIGDIAL*.

Ioannis Efstathiou and Oliver Lemon. 2014b. Learning to manage risk in non-cooperative dialogues. In *Proc. SEMDIAL*.

Kallirroi Georgila and David Traum. 2011. Reinforcement learning of argumentation dialogue policies in negotiation. In *Proc. of INTERSPEECH*.

Piotr J Gmytrasiewicz and Prashant Doshi. 2005. A framework for sequential planning in multi-agent settings. *J. Artif. Intell. Res.(JAIR)*, 24:49–79.

Markus Guhe and A Lascarides. 2014. Game strategies for The Settlers of Catan. In *Proc. IEEE Conference on Computational Intelligence and Games (CIG)*.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2009. *The elements of statistical learning: data mining, inference and prediction*. Springer, 2 edition.

Leslie Pack Kaelbling, Michael L Littman, and Anthony R Cassandra. 1998. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1):99–134.

M. Pfeiffer. 2004. Reinforcement learning of strategies for settlers of catan. In *Proc. Int. Conf. on Computer Games: Artificial Intelligence, Design and Education*.

J. Shim and R.C. Arkin. 2013. A Taxonomy of Robot Deception and its Benefits in HRI. In *Proc. IEEE Systems, Man, and Cybernetics Conference*.

R. Sutton and A. Barto. 1998. *Reinforcement Learning: An Introduction*. MIT Press.

Robert Shaun Thomas. 2003. *Real-time decision making for adversarial environments using a plan-based heuristic*. Ph.D. thesis, Northwestern University.

David Traum. 2008. Extended abstract: Computational models of non-cooperative dialogue. In *Proc. of SIGdial Workshop on Discourse and Dialogue*.