Transfer Learning for Bilateral Multi-Issue Negotiation

Siqi Chen [a] Haitham Bou Ammar [a] Karl Tuyls [a] Gerhard Weiss [a]

[a] *Department of Knowledge of Engineering, Maastricht University, The Netherlands*

Abstract

This paper proposes a novel strategy named Transfer between Negotiation Tasks (**TNT**) for automated bilateral negotiation with multiple issues. **TNT** is able to probabilistically transfer between different negotiation tasks in order to bias the target agent's learning behavior towards improved performance without unrealistic assumptions. We analyze the performance of our strategy and show that it substantially outperforms a powerful negotiation strategy across a variety of negotiation scenarios.

1 Introduction

Automated negotiation has been achieving steadily growing attention as a coordination mechanism for interaction among computational autonomous agents which are in a consumer-provider or buyer-seller relationship and thus typically have different interests over possible joint contracts. Automated negotiation would come in many shapes and forms, for instance, sequential versus concurrent negotiation (i.e., multiple negotiations occur one after the other or at the same time), bilateral versus multi-lateral negotiations (i.e., an agent negotiates with a single other agent or multiple agents are involved in a single negotiation at the same time), and single-issue versus multi-issue negotiation (i.e., a single or several issues are subject of a negotiation among agents) and so on.

Given the pervasive nature of automated negotiation, negotiating agents are required to obtain a high level of self-determination, whereby they decide for themselves what, when and under what conditions their actions should be performed to reach a satisfactory agreement. This objective is however difficult to achieve mainly due to the lack of sufficient knowledge about opponents. To address this problem, existing work concentrates on opponent modeling. For instance, Saha et al. [11] applies Chebychev polynomials to estimate the chance that the negotiation partner accepts an offer in repeated single-issue negotiations on the same domain. In this setting the opponent's response can only be acceptance or rejection to a certain offer. In [2], Brzostowski et al. investigate the online prediction of future counter-offers on the basis of previous negotiation history by using differentials, assuming that the opponent's strategy is known to be based on a mix of time- and behavior-dependent one. In [3] an artificial neural network (ANN) is used to compete against human negotiators in a specific domain, its training however requires a very large database of previous encounters. It is clear that the previously done work assumes certain additional structural assumptions to guarantee the effectiveness of the proposed approach. Apart from the structural assumptions, learning in different negotiation settings within the same domain is slow due to the lack of enough information about the opponent.

Transfer Learning (TL) [1, 7], is one approach that aims at improving learning times and/or performance by leveraging already acquired knowledge in similar tasks. In this work we propose the first, to the best of our knowledge, transfer learning negotiation framework titled "Transfer between Negotiation Tasks (**TNT**)". **TNT** is capable of probabilistically transferring between different negotiation tasks in order to bias the target agent's learning behavior towards better performance. **TNT** also aims at solving the structural assumption problem by relaxing the previous work, where it doesn't introduce any additional assumptions on the structure of the opponent's model and/or its behavior. It rather makes use of nonparametric regression techniques namely, Gaussian Processes (GPs) to learn the opponent's model. In this sense, **TNT** advances the state-of-the-art bilateral multi-issue negotiation by contributing in: (1) proposing a first-of-its-kind transfer opponent modeling framework, (2) outperforming tough target negotiation strategies using the probabilistic strategy transfer mixture, and (3) providing a problem independent negotiation transfer scheme, where the type of function approximators don't restrict the framework from any direction.

The remainder of this paper is structured as follows. Section 2 describes the background preliminaries of this work. Section 3 presents the **TNT** strategy. The performance evaluation is given in section 4. Section 5 discusses experimental results. Lastly, Section 6 concludes and identifies some important research lines induced by the work.

2 Background Preliminaries

2.1 Bilateral Negotiation

We adopt a basic bilateral multi-issue negotiation model which is widely used in the agents field (e.g., [5]) and the negotiation protocol we use is based on a variant of the alternating offers protocol proposed in [8]. Let $I = \{a, b\}$ be a pair of negotiating agents, and let i ($i \in I$) represent a specific agent. The goal of a and b is to establish a contract for a product or service. Thereby a contract consists of a vector of values, each assigned to a particular issue such as price, quality and delivery time. Agents a and b act in conflictive roles. To make this precise, let J be the set of issues under negotiation and j ($j \in \{1, ..., n\}$) be a particular issue. Each agent has a lowest expectation for the outcome of a negotiation; this expectation is called reserved utility u_{res}. During negotiation, each issue j gets assigned a value O_j. The tuple $O = (O_1, \ldots, O_n)$ is called a contract. A contract is said to be established if both agents agree on it.

Following Rubinstein's alternating bargaining model [10], each agent makes, in turn, an offer in form of a contract proposal. An agent receiving an offer needs to decide whether to accept or reject it and to propose a counter-offer. In the case an agent's deadline is reached, it has to withdraw from the negotiation. The agents decide as follows. Each agent has a weight vector (also called importance vector or preference vector) over the issues, representing the relative importance it assigns to each of them. The weight vector of agent i is written as $w^i = (w^i_1, \ldots, w^i_n)$, where w^i_j ($j \in \{1, \ldots, n\}$) is the weight (or preference) which agent i assigns to issue j. The weights of an agent are normalized (i.e., $\sum_{j=1}^{n}(w^i_j) = 1$ for agent i). The utility of an offer for agent i is obtained by the utility function, defined as:

$$U^i(O) = \sum_{j=1}^{n}(w^i_j \cdot V^i_j(O_j)) \tag{1}$$

where w^i_j and O are as defined above and V^i_j is the evaluation function for i, mapping every possible value of issue j (i.e., O_j) to a real number.

After receiving an offer from the opponent, O_{opp}, an agent decides on acceptance or rejection according to its interpretation $I(t, O_{opp})$ of the current negotiation situation. For instance, this decision can be made depending on a certain threshold or can be based on utility differences. Negotiation continues until one of the negotiating agents accepts or withdraws due to timeout[1].

2.2 Gaussian Processes

Gaussian Processes (GPs) are a form of nonparametric regression techniques that perform inference directly in the functional space. In other words, GPs define probability distributions over functions. Concretely, given a data set $\mathcal{D} = \{\mathbf{x}^{(i)}, y^{(i)}\}_{i=1}^{m}$ where $\mathbf{x} \in \mathbb{R}^d$ is the input vector, $y \in \mathbb{R}$ the output vector and m is the number of available data points, when a function is sampled from a GP, we write:

$$f(\mathbf{x}) \sim \mathcal{GP}(m(\mathbf{x}), k(\mathbf{x}, \mathbf{x}')),$$

[1] If the agents know each other's utility functions, they can compute the Pareto-optimal contract [8]. However, a negotiator will not make this information available to its opponent in general.

where $m(\mathbf{x})$ the mean function, and $k(\mathbf{x}, \mathbf{x}')$ the covariance function that both fully specify a GP.

Learning in a GP setting involves maximizing the marginal likelihood given by Equation 2.

$$\log p(\mathbf{y}|\mathbf{X}) = -\frac{1}{2}\mathbf{y}^T \left(\mathbf{K} + \sigma_n^2 \mathbf{I}\right)^{-1} \mathbf{y} - \frac{1}{2}\log|\mathbf{K} + \sigma_n^2\mathbf{I}| - \frac{n}{2}\log 2\pi, \quad (2)$$

where $\mathbf{y} \in \mathbb{R}^{m\times 1}$ is the vector of all collected outputs, $\mathbf{X} \in \mathbb{R}^{m\times d}$ is the matrix of the input data set, and $\mathbf{K} \in \mathbb{R}^{m\times m}$ is the covariance matrix with $|.|$ representing the determinant. It is interesting to see that GPs automatically avoid over-fitting. Maximizing Equation 2 is attained according to conjugate gradient descent; refer to [9] for a thorough discussion of the topic.

2.3 Transfer Learning

Learning in new negotiation tasks is expensive due to either the lack of enough information, or to the complexity of the task. Transfer Learning (TL) is a technique that leverages the usage of information in a source task to aid learning in a target. In TL settings, there typically exists a source and a target task. The agent is assumed to have learned a model of the source task and is now faced by a new target task where little or no information is present. Using the source task knowledge, the target task agent can bias its learning to increase and/or improve the learning speeds, the quality of the learned behavior and/or the overall performance. In this work we focus on transfer learning in supervised learning tasks, where we transfer source task models to aid the target task agent in learning against a new opponent strategy. Our main focus here is to transfer opponent models in the same negotiation domains. In other words, we re-use already learned utility modeling results from a source opponent to aid in learning the strategies against a different opponent in a target negotiation task. There are a lot of directions for this knowledge re-use, such as [7]. In this work we propose a new transfer in negotiation scheme where the source task knowledge is probabilistically re-used in a target negotiation one. The technicalities of the proposed framework are explained and discussed next.

3 Transfer between Negotiation Tasks (TNT)

3.1 Learning in the Source Task

The source negotiation task starts by the opponent agent presenting an offer describing values for the different negotiation issues. Our utility is calculated according to the proposed opponent's offer, which is either accepted or rejected. If the offer is accepted the negotiation session ends. On the other hand, if the offer is rejected our agent proposes a counter-offer to the opponent. Here the opponent can decide, according to his own utility function, whether to accept or reject our counter-offer. The opponent utility function is unknown to our agent rather it tries to learn it incrementally over time. This opponent utility is indirectly modeled from the utilities that our agent attains through the opponents counteroffers. To better clarify, once the opponent agent decides to propose a counteroffer to ours, we calculate the utility we get from its counteroffer and add it to the data set $\mathcal{D}_1 = \{t^{(i)}, u^{(i)}\}_{i=1}^{t_{\max}}$. It is worth noting that this is not a one shot learning. In other words, the data set grows dynamically as the negotiation session continues where the model is trained again with the addition of the new attained data points[2]. Once the $\mathcal{D}_1$ is collected the nonparametric GP approximators are used to learn the opponent utilities.

The negotiation starts by the opponent proposing an offer that is accepted or reject by our agent lines $1-4$ of Algorithm 1. If our agent accepts the proposed offer then the negotiation session ends, line 5. In case there was no agreement we propose a new offer that the opponent has to asses. We use the counteroffer produced by the opponent to approximate his utility indirectly using $\mathcal{GP}_1$, lines $7-15$.

3.2 Knowledge Transfer and Target Task Learning

After the source agent had learned to negotiate against the source task opponent, it is now faced with a new opponent within the same negotiation domain. This target opponent differs from the source one by having

[2] In this work we split the negotiation session in intervals of 3 sec.

Algorithm 1 Source Task Utility Learner

Require: Two negotiation agents, maximum time interval $t_{\max}$

while $t < t_{\max}$ **do**
  Opponent proposes offer
  Calculate utility
  if Accept **then**
    agreement reached
  else
    Propose newoffer
    if Opponent Accept **then**
      agreement reached
    else
      Opponent proposes counteroffer
      Collect time and utility and add to $\mathcal{D}_1$
  Use GPs to approximate the opponents utility
  Use approximated opponents utility function to present an offer
return Opponent's utility model $\mathcal{GP}_1$

a different negotiation strategy that could be more or less powerful. The idea is that the model learned against the source opponent will help in exploiting and learning against the target one. We make use of the source task's GP model by probabilistically proposing a combination of transfer and tough offers to the target opponent. In this context, tough refers to offers randomly produced adhering to a certain utility threshold value. To better clarify, the predicted outputs of the source task approximation model (i.e., $\mathcal{GP}_1$), are probabilistically used in combination with a tough offering strategy to aid the target task learner in approximating the target task opponent's model. This transfer approach is better described in Algorithm 2.

Algorithm 2 Transfer between Negotiation Tasks (TNT)

Require: Two negotiation agents, maximum time interval $t^{(2)}_{\max}$, $\mathcal{GP}_1$, ϵ

1: **while** $t^{(2)} < t^{(2)}_{\max}$ **do**
2:   Opponent proposes offer
3:   Calculate utility
4:   **if** Accept **then**
5:     agreement reached
6:   **else**
7:     Propose newoffer according to,

$$u_2 = \begin{cases} \text{tough offer,} & \text{with } p(\epsilon) \\ \text{transfer offer} & \text{with } 1 - p(\epsilon) \end{cases}$$

8:     **if** Opponent Accept **then**
9:       agreement reached
10:     **else**
11:       Opponent proposes counteroffer
12:       Collect time and utility and add to $\mathcal{D}_2$
13:   Use GPs to approximate the opponents utility
14:   Use approximated opponents utility function
15: **return** Opponent's utility model $\mathcal{GP}_2$

Algorithm 2 requires a target negotiation task, a maximum time interval $t^{(2)}_{\max}$ presenting the end of the target negotiation task, as well as the source task approximated model $\mathcal{GP}_1$. The negotiation procedure commences similarly to that in Algorithm 1, where the target opponent proposes an offer that our agent either accepts or rejects. In the acceptance case, the negotiation is terminated as an agreement has been reached, lines $1-3$. If our agent decides to present a newoffer it makes use of: (1) a tough strategy and (2) a transfer strategy, as shown in line 7 of Algorithm 2. This trade-off between a tough and a transfer strategy is probabilistic where the agent probabilistically decides which offer to propose according to what strategy. In case it decides to make use of the source task knowledge in order to explore the target opponents behavior, it uses the output of the approximated source task model (i.e., $\mathcal{GP}_1$) in order to propose an offer[3] according to, $u_2 \sim \mathcal{GP}_1(\mathbf{m}_1(t^{(2)}), \mathbf{k}_1(t^{(2)}, t'^{(2)}))$, where $\mathbf{m}_1(t^{(2)})$ presents the already learned mean function of $\mathcal{GP}_1$ in the source negotiation task evaluated at the target task's learning time, and $\mathbf{k}_1(t^{(2)}, t'^{(2)})$ represents the already fitted source task covariance function also evaluated at the target task's time[4].

Once these offers are proposed, the target opponent can either accept or reject. In the rejection case, it will propose a counteroffer, in which the utilities are collected from and added to the data set $\mathcal{D}_2$, line $8-12$. Then a GP is used to approximate the target opponent's utility, as shown in lines $14-15$ of Algorithm 2. In expectation this added knowledge is expected to improve the performance of our target agent in approximating the target's opponent utility. This view is solidified and shown applicable in eight experimental domains described next.

4 Experiments

The performance evaluation of **TNT** is done with GENIUS (General Environment for Negotiation with Intelligent multipurpose Usage Simulation [6]) which is also used as a competition platform for the international Automated Negotiating Agents Competition (ANAC). It allows to compare agents (representing different negotiation strategies) across a variety of application domains under real-time constraints, where the preference profiles of two negotiating agents are specified for the individual domains.

In the first set of experiments, we first carry out the source task, which is a negotiation session against a weak opponent according to the ANAC ranking. The source task is done separately in two domains, namely, *Amsterdam Party* and *Travel*. The knowledge gained from previous negotiations is then used by the **TNT-agent** (the implementation of **TNT**) to adjust/optimize its strategy in target tasks, where it competes with different opponents (i.e., the agents who use a weak or strong strategy, which is classified according the performance in ANAC). For the second set of experiments, the knowledge comes from a bargaining process with a strong agent in the same two domains (i.e., *Amsterdam Party* and *Travel*).

In target tasks, we build a simple but strong negotiation strategy named Tough as the basic benchmark. Tough will propose random offers whose utilities are above a certain threshold α over the negotiation course (in our experiments α was set to 0.75.) The weak agent in this task is Agent_K2, who is in the fifth place at the ANAC2011, meanwhile the strong agent is Hardheaded, who was the champion of ANAC2011.

4.1 Weak Source Task

In this set of experiments, we choose Iamhaggler2011 as the weak opponent and the OMACagent [4] as our agent. OMACagent finished in the third place at the ANAC2012, and is able to advance Iamhaggler2011 (the third of ANAC2011) by a significant margin. The utilities of counter-offers from Iamhaggler2011 are used by the Gaussian process to approximate the received opponent utility points, and the resulting predicted values are transferred to the target task.

Figure 1 shows the performance between transfer and no transfer strategies in the domain $Amsterdam\ Party$ playing against Agent_2K. The x-axis represents time percentage elapsed in negotiation and y-axis presents the utility. During the early phase of bargaining, no significant difference can be observed. **TNT-agent** however outperforms its counterpart on the late stage of negotiation. When the negotiations happen in $Travel$ given in Figure 2, the **TNT-agent**, like the Tough, does not obtain obvious concession from Agent_2K on the early stage, meanwhile the gap between the two strategies becomes larger and larger as the negotiation is approaching the end.

The performance difference when negotiating with the hard opponent, $Hardheaded$, is demonstrated in Figure 3 and 4. It is very similar to what we observed before, that is, there is no obvious difference between these two strategies in first 80% of the negotiation. But, the **TNT-agent** again performs much better than the Tough in the remaining time. Another interesting observation is that in Figure 4 the hard opponent

[3]Please note that as the offers' utilities change between the source and the target task, we force the target task agent to adhere to a certain upper-bound once transferring.

[4]Please note that the GP prediction produces a multi-dimensional gaussian distribution and therefore a mean vector and a covariance matrix.

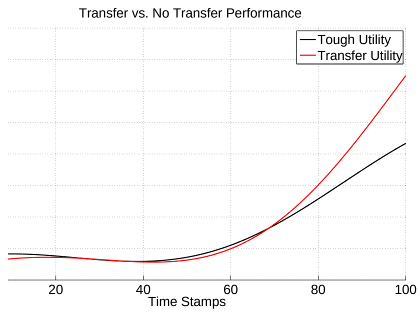

Figure 1: Negotiation with an easy opponent (Agent_2K) in *Amsterdam Party* domain based on easy source task.

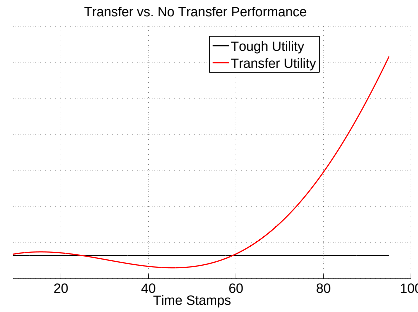

Figure 2: Negotiation with an easy opponent (Agent_2K) in *Travel* domain based on easy source task.

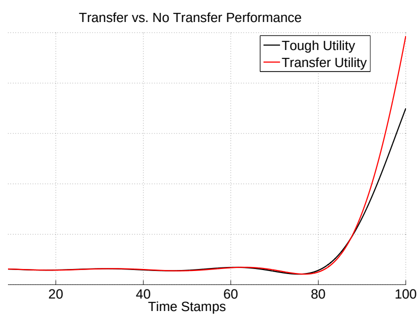

Figure 3: Negotiation with a hard opponent (Head-headed) in *Amsterdam Party* domain based on easy source task.

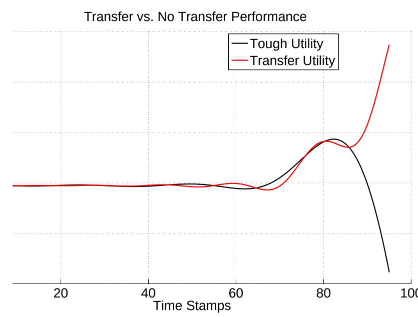

Figure 4: Negotiation with a hard opponent (Head-headed) in *Travel* domain based on easy source transferred to the target task.

made adverse concession with the purpose of achieving better agreement for itself when bargaining with Tough. **TNT-agent**, however, can avoid the negative behavior through combing the learned knowledge with its strategy.

Conclusion I: Transfer from a weak source negotiation task is capable of outperforming both an easy and hard target task opponent in expectation.

4.2 Hard Source Task

In the second set of experiments, the opponent in the source task switches to a strong one (i.e., Gahboninho, the second place of ANAC2011) while we keep using OMACagent as our agent. This is since Gahboninho is very strong in these domains and is able to achieve a very close score to OMACagent. According to Figure 5 and 6, when negotiating with the weak opponent (Agent_2K) in the domains, **TNT-agent** outperforms the no transfer learning approach except for a short period in $Travel$ (i.e., 20 time stamps). The transfer strategy outperforms the Tough one in the later phase of negotiation.

Then, in experiments where our agents play against the hard opponent using knowledge from hard source task, **TNT-agent** still performs well. For the two scenarios given by Figure 7 and 8, **TNT-agent** shows similar behavior. More specifically, its performance is close to the Tough for first 70%, and advances increasingly larger on the closing stage.

Conclusion II: Transfer from a hard source negotiation task is capable of outperforming an easy and hard target task opponents in expectation.

5 Discussions

In this section we will discuss the applicability scope of **TNT** as well provide an intuition on why transfer learning works in such a setting.

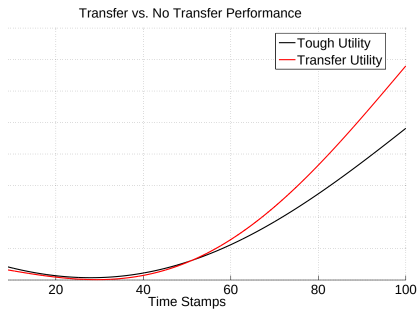

Figure 5: Negotiation with an easy opponent (Agent_2K) in *Amsterdam Party* domain based on hard source task.

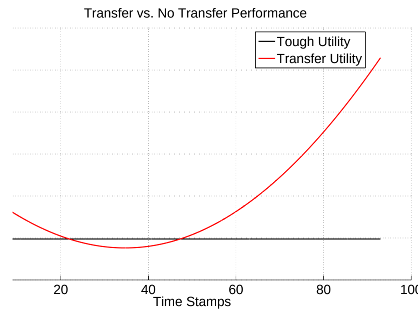

Figure 6: Negotiation with an easy opponent (Agent_2K) in *Travel* domain based on hard source task.

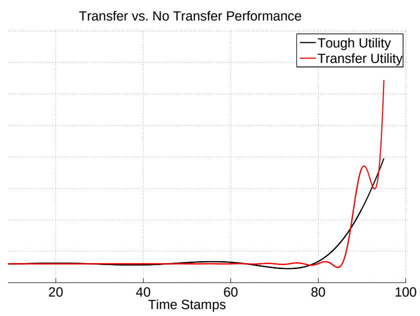

Figure 7: Negotiation with an easy opponent (HardHeaded) in *Amsterdam Party* domain based on hard source task.

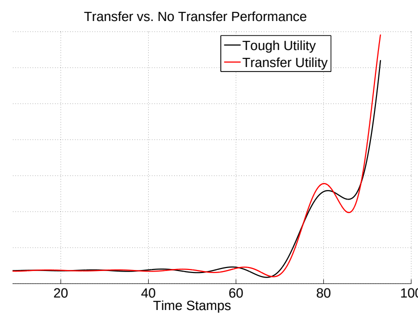

Figure 8: Negotiation with an easy opponent (HardHeaded) in *Travel* domain based on hard source transferred to the target task.

TNT is the first transfer learning agent in negotiation settings. It tries to use already learned behavior in a source task in order to bias and improve learning against a target agent. The first important point is that **TNT** is a function approximation independent platform. In this work we chose to focus on GPs as we believe that the sought function might be complex and hard to picture. For that reason, we have employed a nonparametric functional space prior that is capable of capturing such properties as well as avoiding overfitting automatically. Using any other function approximation scheme such as neural networks, among others, is equally applicable. This increases the scope of applicability of **TNT** to any opponent modeling negotiation setting.

Transfer Learning has been deployed in a lot of interesting fields in machine learning. In the field of negotiation such a fame is still far-sought. Our presented results clearly demonstrate the applicability, and efficiency of transfer learning for negotiation tasks. To better understand the eight experimental results, we speculate that a weak preference to strategy relation, played a role in the success of the transfer scheme. Typically, in negotiation literature the relation between the agent's preferences and the attained strategy is considered to be unimportant or even doesn't exist. In other words, the played strategy is considered to be independent of the agent's preferences. We disagree with this view and argue that such a relation is essential. To better clarify, no agent will produce an offer that is not to some extend influenced by its preferences as an upper bound in a successful negotiation session. We do acknowledge, however, that this relation is weak and that the preference-strategy is not a strong coupling. Having the goal of learning the opponents preferences in the future, and given the positive transfer results, we speculate that this weak preference-strategy relation was one of the main reasons for the success of **TNT**. To better clarify, in our experiments the source and target tasks had similar preferences on different issues but used different strategies (e.g., tough, weak, et cetera.). Since the source and the target opponent agents share this common similarity, and due to the existence of the weak preference-strategy correlation, we expect these different strategies are weakly influenced by the similar preferences and thus have a "common ground". From that point view, the positive results of the transfer algorithm are not surprising but are rather expected.

6 Conclusions and Future Work

In this paper we have presented **TNT**, the first transfer learning framework in negotiation settings. **TNT** makes use of gaussian processes and a probabilistic strategy mixture in order to improve learning against a target task opponents. A set of eight experiments in two different negotiation domains were conducted to proof the applicability of the proposed framework. We speculate that the weak preference to strategy relationship is one of the main reasons for the success of **TNT**. It is worth noting that **TNT** is not restrictive to what function approximation technique to be used.

In our future work we plan to target the following two issues. First, we will conduct a deeper analysis of the preference to strategy relation to better understand the transfer behavior among different opponents. Second, we plan to extend **TNT** to concurrent as well as to multi-issue dependent negotiation settings.

References

[1] Haitham Bou Ammar, Karl Tuyls, Matthew E. Taylor, Kurt Driessens, and Gerhard Weiss. Reinforcement learning transfer via space coding. In *Proceedings of 11th International Conference on Adaptive Agents and Multi-agent Systems (AAMAS 2012)*, Valencia, Spain, June 2012.

[2] Jakub Brzostowski and Ryszard Kowalczyk. Predicting partner's behaviour in agent negotiation. In *Proceedings of the Fifth Int. Joint Conf. on Autonomous Agents and Multiagent Systems*, AAMAS '06, pages 355–361, New York, NY, USA, 2006. ACM.

[3] Réal Carbonneau, Gregory E. Kersten, and Rustam Vahidov. Predicting opponent's moves in electronic negotiations using neural networks. *Expert Syst. Appl.*, 34:1266–1273, February 2008.

[4] Siqi Chen and Gerhard Weiss. An efficient and adaptive approach to negotiation in complex environments. In *Proceedings of the 20th European Conference on Artificial Intelligence (ECAI2012)*, Montpellier, France, 2012. IOS Press.

[5] Robert M. Coehoorn and Nicholas R. Jennings. Learning on opponent's preferences to make effective multi-issue negotiation trade-offs. In *Proceedings of the 6th Int. conf. on Electronic commerce*, ICEC '04, pages 59–68, New York, NY, USA, 2004. ACM.

[6] K. Hindriks, C. Jonker, S. Kraus, R. Lin, and D. Tykhonov. Genius: negotiation environment for heterogeneous agents. In *Proceedings of AAMAS 2009*, pages 1397–1398, 2009.

[7] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning.

[8] Howard Raiffa. *The art and science of negotiation*. Harvard University Press Cambridge, Mass, 1982.

[9] C. E. Rasmussen and C. K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 2006.

[10] Ariel Rubinstein. Perfect equilibrium in a bargaining model. *Econometrica*, 50(1):97–109, 1982.

[11] Sabyasachi Saha, Anish Biswas, and Sandip Sen. Modeling opponent decision in repeated one-shot negotiations. In *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, AAMAS '05, pages 397–403, New York, NY, USA, 2005. ACM.