Building Automated Negotiation Strategies Enhanced by MLP and GR Neural Networks for Opponent Agent Behaviour Prognosis

Ioanna Roussaki, Ioannis Papaioannou, and Miltiades Anangostou

National Technical University of Athens, School of Electrical and Computer Engineering,
9 Heroon Polytechneiou Str, 157-73 Athens, Greece
{nanario,jpapai,miltos}@telecom.ntua.gr

Abstract. A quite challenging research field in the artificial intelligence domain is the design and evaluation of agents handling automated negotiations on behalf of their human or corporate owners. This paper aims to enhance such agents with techniques enabling them to predict their opponents' negotiation behaviour and thus achieve more profitable results and better resource utilization. The proposed learning techniques are based on MLP and GR neural networks (NNs) that are used mainly to detect at an early stage the cases where agreements are not achievable, supporting the decision of the agents to withdraw or not from the specific negotiation thread. The designed NN-assisted negotiation strategies have been evaluated via extensive experiments and are proven to be very useful.

Keywords: negotiating agents, MLP & GR neural networks, NN-assisted negotiation strategies, opponent behaviour prediction.

1 Introduction

Automated negotiations constitute an emerging research field in the artificial intelligence domain [1]. In this framework, building intelligent agents adequate for participating in negotiations and acting autonomously on behalf of their owners is a very complex and demanding task [2]. In automated negotiations three main aspects need to be considered [3][4][5]: (i) negotiation protocol and model, (ii) negotiation issues, and (iii) negotiation strategies that the agents will employ.

Negotiating agents aim to address their requirements of their human or corporate owners as efficiently as possible. As defined in [5], "*Negotiation is a form of interaction in which a group of agents, with conflicting interests and a desire to cooperate try to come to a mutually acceptable agreement on the division of scarce resources*". These resources do not only refer to money, but also include other parameters, over which the agents' owners are willing to negotiate, such as product quality features, delivery conditions, guarantee, etc. [6]. In this framework, agents operate following predefined rules and procedures specified by the employed negotiation protocol [4]. Furthermore, the negotiating agents use a reasoning model based on which their response to their opponent's offers are formulated [7]. This policy is widely known as the negotiation strategy of the agent [8].

F. Sandoval et al. (Eds.): IWANN 2007, LNCS 4507, pp. 152–161, 2007.

This paper[1] is mainly concerned with the design of negotiation strategies for autonomous agents. The proposed strategies are adequate for single-issue bilateral negotiations, where agents have strict deadlines. Learning techniques based on MLP and GR Neural Networks (NNs) are employed by the client agents, in order to predict their opponents' behaviour and achieve a timely detection of unsuccessful negotiations. The proposed NN-assisted strategies have been empirically evaluated and turn out to be highly effective with regards to the duration reduction of the threads that cannot lead to agreements.

The rest of the paper is structured as follows. In Section 2, the basic negotiation framework is presented and the formal problem statement is provided. Section 3 describes the NN-assisted strategies proposed as well as the main aspects of the NNs employed. Section 4 presents the experiments conducted, while Section 5 summarizes and evaluates the results of these experiments. Finally, in Section 6 conclusions are drawn and future research plans are described.

2 The Automated Negotiation Framework Basics

This paper studies a single issue, bilateral automated negotiation framework. Thus, there are two negotiating parties (Client and Provider) that are represented by mobile intelligent agents. The agents negotiate over a single issue based on an alternating offers protocol [9][10] aiming to maximize the utilities of the parties they represent.

We hereafter consider the case where the negotiation process is initiated by the Client Agent (CA) that sends to the Provider Agent (PA) an initial Request for Proposal (RFP) specifying the features of the service/product its owner is interested to obtain. Without loss of generality, it is assumed that the issue under negotiation is the price of the product or service. Thus, the PA negotiates aiming to agree on the maximum possible price, while the CA aims to reduce the agreement price as much as possible. Once the PA receives the RFP of the CA, it either accepts to be engaged in the specific negotiation thread and formulates an initial price offer, or rejects the RFP and terminates the negotiation without a proposal. At each round, the PA sends to the CA a price offer, which is subsequently evaluated by the CA against its constraints and reservation values. Then, the CA generates a counter-offer and sends it to the PA that evaluates it and sends another counter-offer to the CA. This process continues until a mutually acceptable offer is proposed by one of the negotiating agents, or one of the negotiators withdraws from the negotiation, in case for example its time deadline is reached without an agreement being in place. Thus, at each negotiation round, the agents may: (i) accept the previous offer, in case their constraints are addressed, (ii) generate a counter-offer, or (iii) withdraw from the negotiation.

Quantity p_l^a denotes the price offer proposed by negotiating agent a during negotiation round l. A price proposal p_l^b is always rejected by agent a if $p_l^b \notin \left[p_m^a, p_M^a\right]$, where $\left[p_m^a, p_M^a\right]$ denotes agent-a's acceptable price interval. In case an agreement is reached, we call the negotiation successful, while in case one of the

[1] This work has in part been supported by the project "Amigo - Ambient intelligence for the networked home environment" (www.amigo-project.org), funded by the European Commission in the 6th Framework Programme under the contract number IST 004182.

negotiating parties quits, it is called unsuccessful. In any other case, we say that the negotiation thread is active. The objective of our problem is to predict the PA's behaviour in the future negotiation rounds until the CA's deadline expires. More specifically, the negotiation problem studied can formally be stated as follows:

Given: (i) two negotiating parties: a Provider that offers a specific good and a Client that is interested in this good's acquisition, (ii) the acceptable price interval $\left[p_m^C, p_M^C\right]$ for the Client, (iii) a deadline T_C up to which the Client must have completed the negotiation with the Provider, (iv) the final negotiation round index L_C for the Client, (v) a round threshold L_C^d until which the Client must decide whether to continue being engaged in the negotiation thread or not, and (vi) the vector $P_l^P = \{p_l^P\}$, where $l = 2k-1$ and $k = 1,\ldots,\left\lfloor \frac{L_C^d}{2} \right\rfloor$, of the prices that were proposed by the Provider during the initial $L_C^d - 1$ negotiation rounds, *find* (i) the vector $P_{l'}^P = \{p_{l'}^P\}$, where $l' = 2k' - 1$ and $k' = \left\lfloor \frac{L_C^d}{2} \right\rfloor + 1,\ldots, L_C$, of the prices that will be proposed by the Provider during the last $L_C - L_C^d$ rounds, and (ii) decide on whether the Client should continue being engaged in the specific negotiation thread or not.

3 A Negotiation Strategy Based on Neural Networks

The policy employed by negotiating agents in order to generate a new offer is called *negotiation strategy*. In principle, three main families of automated negotiation strategies can be distinguished: time-dependent, resource-dependent and behaviour-dependent strategies [3]. These strategies are well defined functions that may use various input parameters in order to produce the value of the issue under negotiation to be proposed at the current negotiation round. The mechanism proposed in this paper enhances any of the legacy strategies with learning techniques based on Neural Networks (NNs). In the studied framework, the NN-assisted strategies are used by the CA in order to estimate the future behaviour of the PA. This section presents the proposed NN-assisted strategy and describes the specifics of the NNs employed.

3.1 Enabling PA Behaviour Prediction

As already mentioned, the research presented in this paper aims to estimate the parameters governing the PA's strategy enabling the CA to predict the PA's future price offers. The objective is to decide at an early round whether to aim for an agreement with the specific PA, or withdraw from the negotiation thread as early as possible, if no agreement is achievable. For this purpose, two different Neural Networks (NNs) have been employed. These NNs are trained off-line with proper training sets and are then used during the on-line negotiation procedure whenever the CA requires so. The procedure starts normally, and as long as there are enough proposals made by the PA, the CA uses the NNs to make a reliable prediction of its opponent's strategy. This only requires a few negotiation rounds (compared to the CA's deadline expiration round) and this is the main reason why this technique turns out to be extremely useful.

Most negotiation strategies are based on an offer generation procedure that gradually gives ground on the value of the issue under negotiation towards a mutual agreement [3]. In addition to the $[p_m^a, p_M^a]$ interval that represents the range of potential price offers of agent a, there are mainly 3 other parameters that determine the agent's negotiation strategy: k^a, L_p and β. Parameter $k^a \in [0,1]$ determines the initial offer made by the agent at $t = 0$, while $\beta > 0$ is the concession rate. In the study presented in this paper, k^a does not lie among the parameters for prediction as it is safely assumed that it is equal to the PA's initial price offer. L_p represents the PA's deadline. Depending on the value of β, three strategy types are distinguished [3]: *Boulware* ($\beta < 1$) where the agent sticks to its initial offer until the deadline is close to expiring, *Conceder* ($\beta > 1$) where the agent starts conceding to its reservation value fairly quickly, and *Linear* ($\beta = 1$) where the agent concedes by the same amount at each negotiation round. Without loss of generality, we focus on the case where the PA follows a polynomial strategy of arbitrary concession rate and timeout.

The CA negotiates based on a legacy strategy until round L_C^d. Then, the CA makes use of the NNs to obtain estimations $\overline{\beta}$ and $\overline{L_p}$. Round L_C^d will be hereafter called the *prediction round*. In the experiments conducted we have $L_C^d = 30$ and $L_C = 100$. Based on the history of the PA's price offers, NNs attempt to produce a valid estimation of the PA's offer generation function. Then, the CA may determine whether the current negotiation thread can lead to an agreement or this is not feasible given the CA's deadline. Thus, the NN-assisted strategy enables the CA to save time and withdraw early from negotiation threads that will not result in agreements.

3.2 The Neural Networks Employed

Lately, Neural Networks (NNs) have been extensively used in real world applications, as they can be trained to approximate the responses originating from most of the physical or not systems. This behaviour can be modelled so that output estimation with similar inputs is feasible and accurate. In practice, there are two main kinds of NN architectures, the feedforward NNs and the feedback or recurrent ones applied in totally different problem domains [11]. In our framework, where the prediction of a continuous function is required, we selected to study two types of NNs with no feedback loops: the multilayer perceptron (MLP) NN and the Generalized Regression (GR) NN. The latter is a special case of a Radial Basis Function (RBF) NN that is more appropriate for on-line function approximation [11].

A MLP is a common NN architecture applied in various domains, where solutions to diverse and difficult problems are required [12]. Critical parameters affecting the NN's performance are: the number of hidden layers, their corresponding neurons, the NN's weights and the hidden layers' transfer functions. The former are decided by the complexity of the problem and most of the times require extensive experiments to identify an adequate solution [13]. Regarding the network weights, the MLPs use the error back-propagation algorithm [14] to train their values on the supervised learning phase. For the transfer functions we can select among various different species [15].

On the other hand, the RBF NNs [11] have been used mainly for interpolation in multidimensional spaces. This method requires a network architecture that is strict and rather impractical for real world applications, as it supposes a NN that is as large in nodes as the number of the different data points. Thus, we should seek for ways to reduce this size. In this paper we are using a GR NN that is suitable for function approximation with arbitrary accuracy [16].

As the NNs will be used by usually resource limited autonomous agents, the NNs' sizes need to be reduced as much as possible. Furthermore, for the same reason, the time required for prediction, and the storage resources required by the NNs need to be very limited, while the NNs' estimation accuracy needs to be significantly high. Driven by the design principles above, in the remainder of this section we focus on reasoning over the specific characteristics demonstrated by the NNs employed.

For the MLP, we used a training function based on the Levenberg-Marquardt algorithm [13] as it is the most convenient for such problems. Each training vector forms the history of PA's offers until round 30 (as $L_C^d = 30$). Thus, the MLP can be used after round 30 to provide predictions for the future PA offers. The set of training vectors derives from the application of different values for parameters β and L_p to the polynomial function f. The input vectors are generated for the following values of the specified output parameters: β=[0:0.1:0.9 1:1:10], k^a=[0] and L_p=[30:30:300]. From the above values we can see that 19x1x10=190 different vectors have been applied, each for 200 epochs. The 190 output vectors above are the target of the MLP's training. The size of the MLP is 23 neurons on the single hidden layer (log-sigmoid transfer function) and 3 output neurons (linear). This architecture was proven to be the best solution when different networks were tested for estimation efficiency after exhaustive experiments for the MLP architecture. Similarly to the MLP training, we used for the GR training input vectors of 30 values for polynomial function f, with all the possible combinations of the following targets: β=[0.1:0.1:0.9 1:0.5:10], k^a=[0] and L_p=[30:30:300]. Note here that the required vectors (28x1x10=280) are much more than the MLP and the required neurons are thus 280 (1 for each pattern). This was expected, as RBF (and thus GR) NNs tend to have bigger sizes compared to MLP NNs for the same problem. After exhaustive experiments, we selected the spread parameter to be equal to 0.075 instead of the default (1.0), in order to fit data precisely instead of a smoother and less precise fit [13].

Both NNs are employed by CAs and can provide reliable prediction of the PA's behaviour, once sufficient input samples (proposals) are available. The experiments conducted and the NNs performance evaluation are presented in Sections 4 and 5.

4 Experiments

In this section, the experiments conducted to evaluate the performance of the designed MLP and the GR NNs concerning the estimation of the future behaviour of the negotiating PA are presented. The first experiments' family aims to compare the actual behaviour of the PA with the one predicted by the MLP and the GR NNs, when

$\left[p_m^P, p_M^P\right] = [0{,}100]$, $L_P = 200$ and $\beta \in [0.1{,}10]$. The sample values for β are derived from a uniformly distributed random vector of 100 values in the aforementioned area: 50 $\beta < 1$ (Boulware) and 50 $\beta > 1$ (Conceder). The estimated parameters include: the future PA offers until the 100th negotiation round, the minimum PA price offer until then and the PA's concession rate (β). The second experiment family investigates the case where $\left[p_m^P, p_M^P\right] = [0{,}100]$, $\beta = 1$ and $L_P \in [150{,}250]$. The sample values for L_P are: 150:1:250. The estimated parameters include: the future PA price offers until the 100th negotiation round and the minimum PA price offer until then.

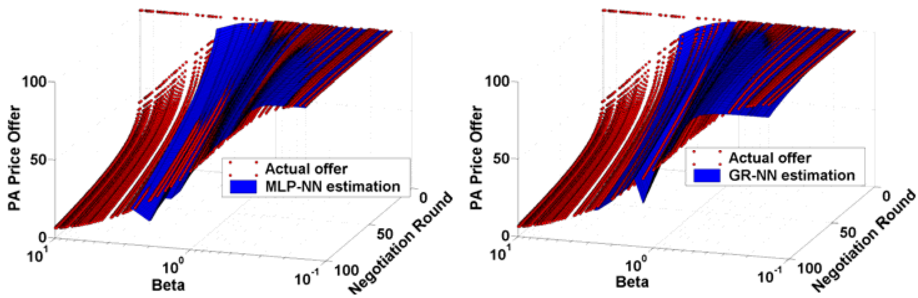

Fig. 1. Actual PA price offer and PA price offer predicted by (a) a MLP-NN and (b) a GR-NN, for 100 negotiation rounds when $L_P = 200$, $p_m^P = 0$, $p_M^P = 100$ and $\beta \in [0.1{,}10]$

The results of the two families of experiments are depicted in Figures 1 and 3a (1st experiment set), Figure 2 (2nd experiment set), and Figure 3b (both experiment sets). In Figures 1a and 2a, the MLP NN estimation for the PA's price offer is depicted (as a blue surface) against the actual PA offer (represented by the red sphere marks) for the 1st and the 2nd experiment family respectively. In Figures 1b and 2b, the same parameters are illustrated but there the GR NN is employed instead of the MLP NN.

As illustrated in Figure 1, the MLP- and the GR-NN perform very similarly managing to accurately predict the PA's price offer in principle. In the same Figure, one may observe that both NNs are used until the 58th experiment (i.e. for $\beta \le 2.8$). For higher concession rates, an agreement is reached before the 30th round and the NN is not necessary for opponent behaviour prediction. As depicted in Figure 2, the MLP- and the GR-NN perform almost identically estimating the PA's price offer with low error margin. However, the deviation between the actual and the estimated PA offers increases as the round index increases and the PA timeout decreases. This is due to the fact that both NNs have a tendency to slightly underestimate PA's concession rate, especially when $\beta \ge 0.5$ (Figure 3a). Finally, as depicted in Figure 3b, with regards to the estimation of the PA's concession rate, the MLP slightly outperforms the GR. A brief analysis of all above findings is presented in Section 5.

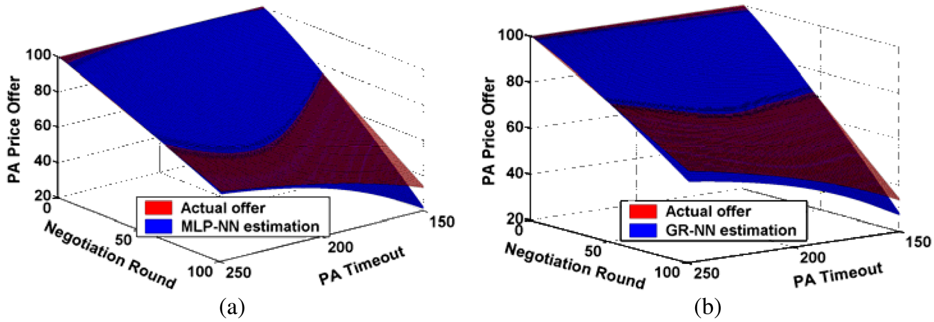

Fig. 2. Actual PA price offer and PA price offer predicted by (a) an MLP-NN and (b) a GR-NN, for 100 negotiation rounds when $\beta = 1$, $p_m^P = 0$, $p_M^P = 100$ and $L_P \in [150,250]$

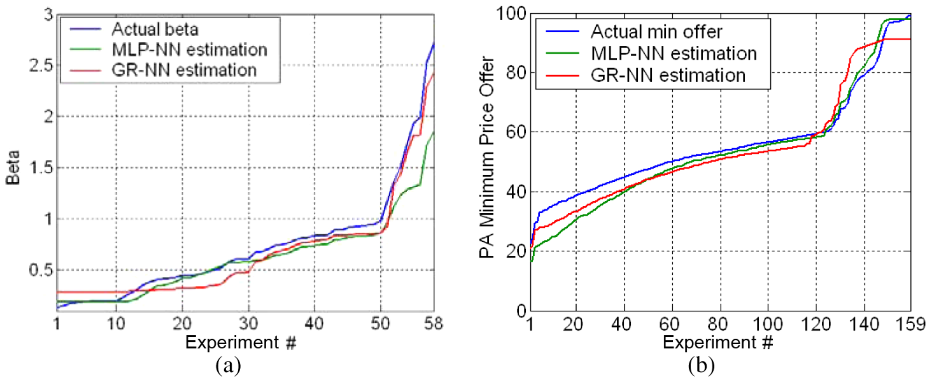

Fig. 3. (a) Actual and estimated (by MLP and GR NNs) concession rate values when $L_P = 200$, $p_m^P = 0$, $p_M^P = 100$ and $\beta \in [0.1,10]$. (b) Actual and estimated (by MLP and GR NNs) PA minimum price offer for all the experiments conducted in both families.

5 Evaluation

In Table 1 comparative results for two experiment families are illustrated with regards to the mean estimation errors of the MLP and the GR NNs concerning the PA price offer, the PA minimum price offer and the PA's concession rate. For all experiment families we have $[p_m^P, p_M^P] = [0,100]$. The rest of the parameter settings are presented in the table's first column, while at the second column the number of the experiments where the NN estimation was used is depicted. The results presented in the rest of the table indicate that the MLP NN slightly outperforms the GR NN with regards to the PA (minimum) price offer estimation demonstrating 0.5% - 2.6% higher accuracy in average. However, the opposite stands concerning the PA beta estimation, as the GR NN provides more accurate estimations by more than 3% in average.

Table 1. Comparative results concerning the mean estimation error of the two NN-assisted negotiation strategies for the PA price offers, for the PA min offer and the PA concession rate

Experiment Settings	Times NN-estimation was used	Mean [price-offer estimation error]		Mean [min-price-offer estim. error]		Mean [beta estimation error]	
		MLP	GR	MLP	GR	MLP	GR
$\beta \in [0.1,10]$, $L_P = 200$	4118	0.97%	2.12%	0.41%	2.80%	15.65%	8.26%
$L_P \in [150,250]$, $\beta = 1$	7171	1.21%	1.71%	8.26%	8.91%	12.51%	12.73%
OVERALL	11289	1.12%	1.86%	5.40%	6.68%	13.92%	10.72%

As already stated, the enhanced strategies use the NN estimation for the minimum acceptable price of the PA to decide whether they should continue being engaged in the specific negotiation thread or not. In case $\overline{p_m^P} > p_M^C$, where p_m^P is the price offer made by the PA to the CA upon the CA's deadline expiration (in our study at round 100), the CA terminates the negotiation at round 30. In Table 2, evaluation results for the two NN-assisted negotiation strategies are illustrated for both experiment families assuming that $p_M^C = 50$ [2]. The experiment settings are presented in the table's first column, while at the second column the number unsuccessful negotiation threads (UNTs) is depicted. These unsuccessful negotiations are due to the fact that $p_m^P > p_M^C$. The third column indicates that the duration of the UNTs is always equal to $L_C = 100$ [3] in case no opponent behaviour prediction mechanism is used. The next pair of columns illustrates the number of UNTs that were detected by the NNs at round 30, while the subsequent pair of columns presents the UNTs' elimination ratio, i.e. the ratio of UNTs that were correctly identified by the NNs as unsuccessful and terminated before the expiration of the CAs deadline. It should be mentioned that the MLP NN manages to identify ~91% of the UNTs in average, while the GR NN detects ~83% of the UNTs in average. The last two pairs of columns illustrate the mean duration of the UNTs and the mean UNT duration decrease with regards to the case where no opponent behaviour prediction mechanism is used. It should be highlighted that the MLP-NN assisted negotiation strategy achieves ~64% reduction of the UNTs' duration in average, while the GR-NN assisted strategy manages to reduce the UNTs' duration by ~58%. This is highly significant as the CA has the time to get engaged in approximately another two negotiation threads that may lead to agreements. Of course, as expected and as one may also observe in Figure 3b, the lower (higher) quantity p_M^C is, the more (less) cases of UNTs occur and the higher (lower) mean UNTs' duration decrease is achieved by the NN-assisted strategies.

With regards to the elimination of the UNTs, the MLP-assisted strategy clearly outperforms the GR-assisted negotiation strategy. However, with regards to the processing/time resources required, the GR NN outperforms by far the MLP NN. As presented in Table 3, the mean training time required by the MLP NN (i.e. 203 sec) is approximately 1450 times higher in average than the time required by the GR NN

[2] We selected the p_M^C to be equal to the median value in the PA's acceptable price interval.

[3] To be more accurate, the duration of UNTs is equal to: $\min(L_C, L_P)$. However, in this paper's study, we always have $L_C < L_P$, and thus the duration of UNTs is equal to L_C.

(i.e. 0.14 sec). This happens due to the fact that the MLP NN training is a highly complex procedure requiring forward and backward passes of weight updates in order to render them stable. And although the training vectors required for the MLP are far less compared to the GR, the former requires 200 passes (epochs) of these vectors in order to be adequately trained. But as the NNs are trained only one time (off-line), these time resources required are not that significant. However, the NN simulation time required (on-line) is a more suitable measure of comparison. This is comparable for the two NNs as shown in Table 3, as the simulation time required by the MLP NN (i.e. 0.024 sec) is just ~26% higher in average than the one required by the GR NN (i.e. 0.019 sec). Nevertheless, the mean storage resources required by the GR NN (i.e. 172 KB) are approximately 4 times higher in average than the storage resources required by the MLP NN (i.e. 172 KB), as the overall number of neurons used by the MLP NN is just 26, while the GR NN requires 283 neurons in total. For the reasons above, it is estimated that a MLP NN is more appropriate for assisting negotiating intelligent agents to estimate their opponent's behaviour at an early negotiation round in case the agent values a timely detection of unsuccessful negotiation threads.

Table 2. Comparative results concerning the unsuccessful negotiation thread detection by the two NN-assisted negotiation strategies

Experiment Settings	# Unsuc. Negot. Threads (UNTs)	Mean duration of UNTs (no NN)	# UNTs detected at round 30		UNTs' elimination ratio		Mean UNTs' duration		Mean UNTs' duration decrease	
			MLP	GR	MLP	GR	MLP	GR	MLP	GR
$\beta \in [0.1,10]$, $L_P = 200$, $p_M^C = 50$	50	100	49	49	98.0%	98.0%	31.4	31.4	68.6%	68.6%
$L_P \in [150,250]$, $\beta = 1$, $p_M^C = 50$	51	100	43	35	84.3%	68.6%	41.0	52.0	59.0%	48.0%
OVERALL	101	100	92	84	91.1%	83.2%	36.2	41.8	63.8%	58.2%

Table 3. Comparative results for the time and storage resources required by the NNs used

NN Type	Experiment set size	Times NN estim. used	Mean training time	Mean simul. time	# neurons required	Mean storage requirements
MLP	20301	11289	203.00 sec	0.024 sec	30-23-3	45 KB
GR	20301	11289	0.14 sec	0.019 sec	30-280-3	172 KB

6 Conclusions and Future Plans

Using Neural Networks to enhance intelligent agents that negotiate over a single issue, turns out to be extremely useful, leading to substantial duration reduction of unsuccessful negotiation threads. When the CA uses the NN-assisted strategies it is capable of predicting its opponent's behaviour with significant accuracy, thus getting aware of the potential outcome of the negotiation. Both the MLP and the GR NNs demonstrate average opponent price offer estimation error lower than 2% and PA min acceptable price estimation error ~6%. Additionally, the unsuccessful negotiations are detected by the MLP NN in more than 90% of the cases in average, demonstrating

~8% better overall performance than the GR NN. In a nutshell, the CA is enhanced with the ability to avoid a possible unprofitable or even unachievable agreement. This leads to minimization of the required resources and maximization of the CAs overall profit from a series of threads for a single commodity. After these promising results, we are now working on alternative NN architectures and on the design of a hybrid CA strategy that coupling the NN estimations with legacy strategies from the very first round. Finally, we aim to study scalability aspects and lead autonomous agents deliberate over negotiation, as well as to apply our techniques on PAs following arbitrary strategies, a highly challenging task in the automated negotiation field.

References

1. Kraus, S.: Automated Negotiation and Decision Making in Multiagent Environments. Lecture Notes in Artificial Intelligence 2086, 150–172 (2001)
2. Lomuscio, A., Wooldridge, M., Jennings, N.: A Classification Scheme for Negotiation in Electronic Commerce. In: Dignum, F, Sierra, C. (eds.) Agent Mediated Electronic Commerce. Lecture Notes In Computer Science, vol. 1991, pp. 19–33. Springer, Heidelberg (2001)
3. Faratin, P., Sierra, C., Jennings, N.J.: Negotiation Decision Functions for Autonomous Agents. Int. J. of Robotics and Autonomous Systems 24(3-4), 159–182 (1998)
4. Rosenschein, J., Zlotkin, G.: Rules of Encounter: Designing Conventions for Automated Negotiation among Computers. MIT Press, MA, USA (1994)
5. Jennings, N., Faratin, P., Lomuscio, A., Parsons, S., Sierra, C., Wooldridge, M.: Automated Negotiation: Prospects, Methods, and Challenges. Int. J. of Group Decision and Negotiation 10(2), 199–215 (2001)
6. Lai, G., Li, C., Sycara, K., Giampapa, K.: Literature Review on Multi-attribute Negotiations. Technical Report CMU-RI-TR-04-66, Robotics Institute, Carnegie Mellon University Pittsburgh USA (2004)
7. Muller, H.: Negotiation principles. In: O'Hare, G., Jennings, N. (eds.) Foundations of Distributed Artificial Intelligence, pp. 211–229. John Wiley and Sons, New York, USA (1996)
8. Skylogiannis, T., Antoniou, G., Skylogiannis, N., Governatori, G.: DR-NEGOTIATE - A System for Automated Agent Negotiation with Defeasible Logic-Based Strategies. IEEE Int. Conf. on e-Technology, e-Commerce and e-Service (EEE'05), Hong Kong China (2005)
9. Osborne, M., Rubinstein, A.: A course in game theory. MIT Press, MA, USA (1994)
10. Kraus, S.: Strategic Negotiation in Multiagent Environments. MIT Press, MA, USA (2001)
11. Haykin, S.: Neural Networks: A Comprehensive Foundation, 2nd edn. Prentice Hall, London, UK (1999)
12. Caudill, M., Butler, C.: Understanding Neural Networks: Computer Explorations, 1st edn., vols. 1, 2. MIT Press, MA USA (1992)
13. Hagan, M., Demuth, H., Beale, M.: Neural Network Design. PWS Publishing Company, Boston, MA, USA (1996)
14. Rumelhart, D., Hinton, G., Williams, R.: Learning internal representations by error propagation. In: Rumelhart, D., McClelland, J. (eds.) Parallel Distributed Processing, vol. 1, MIT Press, MA, USA (1986)
15. Duch, W., Jankowski, N.: Survey of Neural Transfer Functions. Neural Computing Surveys 2, 163–212 (1999)
16. Wasserman, P.D.: Advanced Methods in Neural Computing. John Wiley and Sons, New York, USA (1993)