Trust in Motivated Learning Agents

J. A. Starzyk and J. Graham
School of EECS
Ohio University,
Athens, OH, USA
{jg193404, starzykj}@ohio.edu

A. Horzyk
Dept. Automatics and Biomedical Engineering
AGH University of Science and Technology
Krakow, Poland
horzyk@agh.edu.pl

***Abstract*—This paper generalizes motivated learning to include selected personality traits of autonomous learning agents. Specifically, we show how a motivated learning agent can develop trust and use it in interaction with other agents in a dynamic unknown environment where it operates and learns. Trust is built over time through interactions with other agents, while considering the interactions' effects on the needs of the motivated learning agent. A positive effect increases trust while a negative one reduces it. A neural network model used in the motivated learning scheme is utilized to compute the trust level for each non-agent character. Simulation of the motivated learning agent is performed to demonstrate the benefits that result from using trust in interaction with other agents. We imply that trust may influence other traits of character like shyness, friendliness, bravery, etc.**

Keywords—trusting agents, motivated learning, organizing principles, personality traits.

I. Introduction

In human relations trust indicates willingness to rely on someone else's future actions. It contains elements of accepting risk and believing that the trustee will act as expected by the trustor. Humans are naturally inclined to trust and to judge trustworthiness starting from the early development of the brain as part of the nourishment and care a child receives.

Trust plays a useful role in relationships between individuals and groups of people [1][2] and is a factor in developing complex relationship between individuals. Without trust, all contingencies have to be carefully considered, leading to a slow und uncertain decision making process. Trust reduces the complexity of human relations, allowing actions to be taken that otherwise would be too complicated or too risky. Trust also indirectly facilitating cooperation. Believing in the future outcomes of actions taken creates a mutual dependency between participants [3] leading to the creation of more and more complex social structures. Working in a trusting environment increases the chances for cooperation and success [4]. Trust and mutual benefit were the main reasons the behind development of villages, cities, nations and international organizations.

The tendency to trust others is an important personality trait [5][6]. We can differentiate many personality traits defined from various points of views [2][7][8][9][10]. People usually place more trust in other people that have similar traits to their own because their reactions are better understandable or look favorable and professional for them in general. Compatibility between people's traits makes them more comparable to each other, which leads to trust. For example, a systematic person will usually perceive another systematic person as more professional and trustworthy than other non-systematic people because he or she has order in their plans, actions, and general behavior [10].

Trust is also related to human character traits. People with some kinds of personality traits do not usually trust anyone or trusting is very difficult for them in general, e.g. people who like to verify, criticize, protect, or control everything and everybody [10]. They are always on the lookout for something that is beyond their control, not sufficiently secured or protected, damaged or not fully functional. People with harmonious traits can hardly be fully trusted about what they say because their tendency to avoid quarrels, disputes and disagreements stops them from telling the whole truth [7],[10]. Likewise, empathic people avoid hurting others so they often cannot tell the difficult truth. Other traits like to trust only people that are open-minded, empathic, tolerant, or accommodate people's needs. In general, people's traits are very important motivation factors associated with their needs, choices, decisions, and behavior. The ability to recognize people's traits is necessary for intelligent agents to correctly predict human reactions, be able to better cooperate with people and be perceived as intelligent and trustworthy [7], [11]. Understanding of human personality traits [2][9]allows for better understanding and cooperation between people [10], and can help autonomous motivated learning (ML) agents to better respond to various human needs according to the recognized traits. Furthermore, cybernetic modeling of human traits in ML agents makes their behavior more humanlike, positive, and trustworthy because they can act as if they really understand people and their needs.

Motivated learning was developed to create abstract needs in autonomous intelligent agents [12]. It is a framework in which motivations are created out of a few primitive needs, as the agent experiences the environment [13]. It provides a vehicle by which an agent can become more knowledgeable, more sophisticated, and more adapted to the environments in which it must act.

In the motivated learning framework, no prior knowledge or understanding of the environment is used. Thus, all desired features must be derived from a few organizing principles in the motivated learning agent. Prior work resulted in such organizing principles that provided an agent with learning mechanisms for how to use resources in the environment to its own advantage

Manuscript received 30 Mar. 2016.
This research was supported by The National Science Centre, Poland, grant No. 2011/03/B/ST7/02518 and by AGH 11.11.120.612.
J. A. Starzyk and J. Graham are with the School of Electrical Engineering and Computer Science, Stocker Center, Ohio University, Athens, OH 45701.
J. A. Starzyk is also with University of Information Technology and Management, Rzeszow, Poland.
A. Horzyk is with Department of Automatics and Biomedical Engineering, AGH University of Science and Technology, 30-059 Krakow, Poland.

and how to react to actions (either desirable or harmful) by other agents [14].

We believe that any desired feature of an intelligent agent's character, like trust, emotions, etc. must be derived from a few organizing principles that may be compared to genetically defined modifiers of the agent's learning and behavior. These modifiers should be embedded in the agent in a generic form without specificity of culture, language, objects or tasks that the agent will face in its environment. Following this strategy, we developed organizing principles in this paper that lead to trust or lack of it towards other agents. We also demonstrated how developed trust is beneficial to the agent, how it can change over time, and how it can lead to the development of emotions in the ML agent.

In this paper we describe a schema that can be used to introduce personality traits in autonomous learning agents. Specifically we focus on trust as directly related to learning agent experience in interaction with other agents. We show how trust towards other agents can be defined and computed based on past experiences. We demonstrate that the learning agent may benefit from using trust in its interactions in an environment populated with other agents.

The rest of this paper is organized as follows. Section II presents the behavioral foundations of developing trust and describes a function that yields a numerical trust value. This function shows how to use the bias weights that were obtained in the motivated learning process in relation to other agent actions in order to compute trust for another agent. This section also shows how to parameterize the trustworthiness of a learning agent. Section III presents an algorithm for trust development in a motivated learning agent. This section also shows simulation results that demonstrate the learning advantage that a trusting agent has over one that does not use trust in its decision making process. Section IV presents a general discussion of character traits and how they can be developed and used in motivated learning agents. Finally, section V presents conclusions.

II. Behavioral Fundations for Trust

When developing learning agents a challenge that one must face is to provide them with proper mechanisms to help with things like learning, generating knowledge, developing character and yielding higher cognitive capabilities that will be useful throughout their lifetimes. We call them *organizing principles* for learning.

The motivated learning (ML) approach that we are developing [15] is a generalization of reinforcement learning (RL), where instead of fixed value functions for specific rewards, we consider a dynamic system of values created according to new needs that a system may generate. In [16] we demonstrated how this ability to generate and manage various needs improves performance even in the relatively impoverished environments of virtual worlds. Trust, emotions, and understanding extend this even further towards modifiers of behavior and cooperation, leading in a natural way to a social agent with expectations, predictions and a character.

Trust is needed if we want to extend our system to include a teacher. A ML agent must trust another agent before it accepts it as its teacher. Trust is also a necessary ingredient if we want to build social agents. Because trust has clear social values it is worth developing and maintaining.

Trust enriches our ML approach and makes it more effective in the development of intelligent agents. Further extensions of the organizing principles for ML may include emotions. However, developing emotions in ML agents is out of the scope of this paper and is planned for future work.

A. Computational Fundations of Trust

In our ML scheme an agent develops a bias for or against specific actions by other agents. To distinguish these other agents from the ML agent we name them non-agent characters (NACs). NAC actions will either be desirable, undesirable or neutral. The desirability of NAC actions is closely related to trust. We develop trust after experiencing helpful actions by others. Likewise, we do not trust those who hurt us. The degree of trust varies based on the traits of character and experience.

Thus, in our approach trust is not given but must be earned (and learned), which makes it a perfect extension and a consequence of action motivations. Once trust is lost it is hard to regain it. Since trust is related to predicting the results of future actions by others, we may use the mechanism we introduced to learn how to respond to NAC actions [14].

In general, if the ML agent likes the results of a NAC action it learns how to encourage the NAC to repeat the action, and if it does not like the action, it tries to discourage the NAC from repeating this action. In ML this is done by introducing biases for or against the NAC action and abstract pains related to such bias signals. The agent generates an internal pain signal if it cannot perform a desired action or cannot stop an undesired one. Such simple rules for generating intrinsic pain signals are general enough for the agent to develop many abstract goals from those that are needed for its survival - fight its enemies, make friends, or obey a teacher. The ML agent is motivated to reduce its internal pains, regardless of what triggered these pains. Such a simple rule is a foundation for goal creation and goal management as discussed in [13].

There are two kinds of action related pains – the agent's own action, or the action of a NAC. When the pain increases as a result of the agent's own action, such action must be avoided. When the pain increases as a result of a NAC's action – the agent must learn how to avert such an action. The ML agent can learn how to do this by introducing biases related to NAC actions. The bias signal is activated whenever the environment conditions are similar to those that caused the pain increase or pain reduction in the past.

'NAC Action Pain' is first learned when the NAC's action is observed in correlation with an increased pain. Subsequently the NAC action pain is triggered by the 'NAC Action Bias'. Likewise, the agent would like to encourage an action if the observed result of a NAC's action is beneficial (reduced pain or reward received). Such biases are created as the result of pain reduction.

As discussed in [15] the 'NAC Action Bias' is calculated from:

$$B(s_i) = -(1+\delta_i) * (lnA(s_i) + 1) + 2 * A(s_i) \qquad (1)$$

where $\delta_i = 1$ when the NAC action is *desired*, $\delta_i = -1$ when it is *not desired*, and $\delta_i = 0$ otherwise. $A(s_i)$ represents action availability, computed from:

$$A(s_i) = \frac{1}{\frac{d_c}{d_d} + \frac{1+\delta_i}{2}} \quad (2)$$

where d_c is current and d_d is a desired (a comfortable) distance to another agent.

Setting the 'NAC Action Pain' is important since this motivates the agent to properly respond to a NAC action. This motivates the agent to learn the skills how to beneficially interact with other agents. While distance based NAC availability or a function chosen to compute the bias signals are set arbitrarily, other forms of implementing these functions are acceptable as long as they reflect relative changes in the strength of bias signals. For instance, the distance based availability $A(s_i)$ used in (2) may be replaced with the likelihood that the NAC agent will perform its action. In any case, the increase in the bias signal values when the agent is not able to either encourage a desired action or stop an undesired one must be preserved. This means that if the likelihood that NAC may be able to perform an undesired action goes up, then the bias signal (against this action) must increase. Similarly, if likelihood to perform a desired action goes down, the bias signal (for the action) must go up. This triggers an abstract pain for ML agent to encourage such action.

B. Bias Weight Changes

Increasing strength of bias signals triggers abstract pain signals related to specific NAC actions. The agent learns the importance of the observed events by adjusting bias weights w_{bp} (shown in Fig. 1).

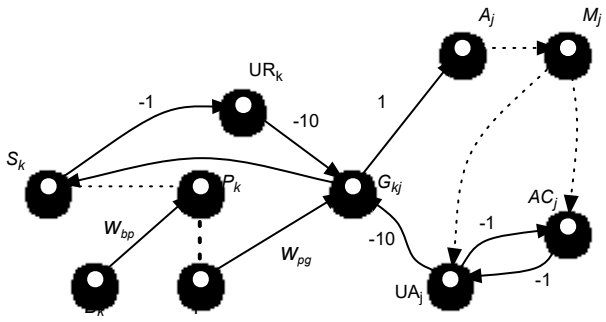

Fig. 1. Synaptic connections between pain, bias, and goal neurons.

Fig. 1 shows trainable connections between bias B, pain P, and goal G neurons as well as additional inhibitory neurons (*unavailable resource* (UR) and *unsuccessful action* (UA) neurons) that represent the environment conditions. A UR neuron inhibits goal selection if a resource required to perform the action is not observed on the sensory input, while a UA neuron inhibits goal selection when a desired action could not be completed (due to motor malfunction or adverse environment conditions). A UA neuron is normally inhibited by an inhibitory link from an *action completed* (AC) neuron, indicating that a motor function can be performed if needed. When a desired action (A) cannot be completed a UA neuron is activated. In Fig. 1, S represents sensory input neuron and M is a motor neuron that represents an action taken by the agent.

Initially, all bias weights w_{bp} are set to 0. In this initial stage, the machine responds only to the primitive pain signals P directly triggered by a sensory input from the environment. Each time a specific pain P is reduced, the weight w_{bp} of the B_k-P_k bias link increases. However, if the goal activated by the pain center P was completed and did not result in a reduction of pain P, then the B_k-P_k weights w_{bp} are reduced.

When a specific goal is not invoked for a long period of time its importance in satisfying a lower level pain is gradually reduced. This requires a reduction of the w_{pg} weight to this goal from all the pain centers. A similar reduction of the w_{bp} weights indicates a gradual decline in importance of an abstract pain. Without such a decay mechanism, the machine can set higher level goals even if better ways were discovered to support its lower level needs.

To implement these concepts, the bias weight w_{bp} is computed incrementally based on pain change signals that resulted from the action taken as follows:

$$w_{bp} = \begin{cases} w_{bp} + \Delta_{b+} * (\alpha_b - w_{bp}) & if\ associated\ pain\ changed \\ w_{bp} * (1 - \Delta_{b+}) & if\ there\ was\ no\ change\ in\ pain \\ w_{bp} * (1 - \Delta_{b-}) & if associated\ percept\ was\ not\ used \end{cases} \quad (3)$$

where $\alpha_b = 0.5$, sets the ceiling for w_{bp}; $\Delta_{b-} = 0.0001$, sets the rate of decline for w_{bp} weights; $\Delta_{b+} = 0.08$, sets the rate of increase for w_{bp} weights. These parameters are set experimentally and while they affect the learning speed, the agent behavior is not critically sensitive to these parameters. The bias weights are limited to the (0, α_b) interval, α_b sets an upper limit for each weight (typically set to 1), Δ_{b+} should be smaller than 1 (typically 0.1 or less) and should be significantly larger than Δ_{b-} since a positive experience (of using a specific sensory input) is more telling about its effect on pain than a negative experience where this sensory input was not used at all.

Since the bias weight w_{bp} indicates how useful it is to perform a selected motor function M, a bias weight adjustment parameter Δ_{b+} must be properly selected to reflect the rate of stimuli applied to handle a higher order pain center. This rate reflects how often a given abstract pain center P_k was used to reduce the lower order pain signal P.

Using the bias signal, the pain value is estimated from:

$$P(s_i) = B(s_i) * w_{bp}(s_i) \quad (4)$$

where w_{bp} is a bias to pain weight for a given pain center.

The ML agent chooses its goals depending on the relative strength of pain signals and environment conditions allowing it to control these pains. In a simple neural network implementation of ML, the goal is selected based on the strength of interconnection weights w_{pg} and activation of pain signals P shown in Fig. 1. A given pain signal P is multiplied

by w_{pg} weights and a goal neuron with the strongest activation is selected to chose and intended action A. Then, if the environment conditions are right, the ML agent initiates a motor action to reach its goal.

C. Action Selection

If the NAC acts and as a result the ML agent's pain increases, the agent must protect itself or its resources by performing the proper action. The correct response depends on the agent's ability to observe the NAC's action and prevent it from harming the agent. Thus, detecting an action by the NAC and determining if it is a desired or undesired action is critical. A NAC action is desired if, as a result of this action, a primitive or an abstract pain of ML agent is reduced, and it is undesired if the pain increases.

When the agent detects a NAC action, it activates a potential pain based on the bias signal (1) for the NAC action. This potential pain competes for the agent's attention and motivates it to act defensively. At the same time, the potential pain is not registered as a pain increase, since it is only the motivating factor and not the real pain. Likewise, if the agent acts defensively and stops the attack, it learns the proper action by observing that the real pain did not increase. This lack of pain increase is sufficient reason to reinforce such behavior. However, if the real pain increased, this indicates that the agent did not use a proper response to the attack, and the corresponding w_{pg} weight must go down.

This action related potential pain signal is removed as the observed NAC action stops. If it stopped as a result of the agent action this response was correct one and reinforced learning takes place. If the pain stopped without the agent's action (for instance when the attacker walks away), the learning agent gradually reduces the potential pain signal without learning.

If the agent is attacked, it can resolve to either defend itself by attacking the enemy or running away from it. The agent can learn which action is a better choice, depending on its past experiences. The cognitive agent will be able to estimate its chance of running away from the enemy based on its understanding of its own and the enemy's embodiment and limits on its ability to prevent the enemy's action.

D. Developing Trust

In motivated learning, trust is associated with NACs and their actions. While learning the desirability of NAC actions and how to respond to them, the ML agent also develops trust towards the NAC. If most of the NAC's actions are desirable (as when a parent nourishes and takes care of their child, providing it with warmth and comfort) the ML agent's trust towards the NAC's actions increases. If, on the other hand, the NAC's actions are undesirable (and increase the ML agent's pain), trust towards the NAC decreases.

Thus, ML is generalized from a learning system that learns how to use resources in the environment to its own advantage and how to react to actions (either desirable or harmful) by other agents, to one that can develop trust, friendship, and cooperation. This is also a first step towards accepting another agent as a teacher – a trusted teacher can accelerate the learning speed of an ML agent by showing it how to act in various situations. Hence, the ability of an ML agent to learn whom to trust is very important.

Consequently, a mechanism to develop trust is considered as one of the few organizing principles that a ML agent can use to develop its skills. At the same time, it is general enough that we do not directly describe agents that can be trusted, we do not specify what actions they must perform to be trusted, or we do not use any initial knowledge about the characterization of what type of behaviors are indicative of a trusting relationship.

Since trust is one of character features that modifies an agent's behavior, we may differentiate agents by how easy they trust. More trusting agents will be more tolerant to accidental events that cause them harm, while more paranoid or untrusting agents will remember any harm caused and their trust will be hard to gain.

In this work we use the bias weights w_{bp} described in (3) and desirability factor δ of a NAC action to characterize trust towards a NAC. Trust is developed independently toward each NAC agent and its actions.

Other features like shyness may be related to overall experience with NAC agents. If most NACs interactions hurt the ML agent it may develop mistrust to all NACs, and become shy, while on the other hand if most of the NACs that a ML agent encounters are good to it, the agent may become more trusting towards strangers.

Trust is defined separately for each NAC based on prior experiences the ML agent with the NACs, and is characterized by a value between -1 and 1:

$$T_{NAC} = \frac{\sum_{k=1}^{n_k} \delta_k \left(\frac{\sum_{i=1}^{m_i} w_{b_k p_i}}{m_i} \right)}{n_k} \tag{5}$$

where n_k is the number of different types of actions that a NAC performed that resulted in an increase or decrease in pain for the ML agent, δ_k is desirability of action k performed by the NAC, m_i the number of pain signals affected by this action, $w_{b_k p_i}$ is a bias to pain weight specific to action k and affecting the pain signal p_i. In most of the cases $m_i = 1$ and (5) computes a weighted average of the bias to pain weights for all the actions performed by the NAC that modified the ML agent's pain.

E. Qualifying Trust

Computed by (5), the trust value puts equal weight on both desirable and undesirable actions by the agent. If we want to differentiate more between good and bad actions and provide a means to introduce variability in the ML agent's trust we will use the following:

$$T_{NAC} = \gamma_d \frac{\sum_{k=1}^{n_d} \left(\frac{\sum_{i=1}^{m_i} w_{b_k p_i}}{m_i} \right)}{n_d} - \gamma_u \frac{\sum_{k=1}^{n_u} \left(\frac{\sum_{i=1}^{m_i} w_{b_k p_i}}{m_i} \right)}{n_u} \tag{6}$$

where n_d is a number of different desired actions by the NAC, n_u is a number of different undesired actions, $0 < \gamma_d < 1$ is the trust factor level, $0 < \gamma_u < 1$is the level of distrust factor.

While in (5), if we have the same number of desirable and undesirable actions they balance to result in a neutral trust towards the NAC, in (6) this is not the case. The ML agent with $\gamma_u < \gamma_d$ tends to brush off any negative experience and may reward the NAC with greater trust even with a smaller number of desirable NAC actions. We may characterize such an agent as friendly towards others. And the opposite is true if $\gamma_d < \gamma_u$. Such an agent puts a higher weight toward negative experiences, is less trusting, and as a result may lead to a more cautious behavior.

While other ways to characterize trust are possible, we chose this one for its simplicity, as we focus on presenting the developmental advantage of the ML agent that uses trust over the one without it.

III. Using Trust in Motivated Learning

A. Design of Trust in Motivated Learning

For the experiments described in this section we developed an algorithm that uses an environment in which the ML agent operates together with one or two NACs (although more are possible). Using this approach, we are able to test the effects of NACs with differing levels of "friendliness", where NAC friendliness is defined as a prespecified percentage of NAC actions that are beneficial to an ML agent (while the remaining actions are harmful). When presented with an action by a NAC the ML agent has the option of performing one of three actions. It can "kick" the NAC to stop it from performing the action, do nothing, or encourage the NAC to perform the action.

As in our previous work on ML agents [12]-[16], the agent will learn the desirability of the actions of each NAC it is exposed to based on a pain reduction and adjust its w_{bp} (bias to pain) weights accordingly. The w_{bp} weights start at 0 and are adjusted using (3). Since actions may be repeated a few times (they are randomly generated in these tests), the weights will change gradually, providing a basis to compute trust values for each NAC. During learning, the ML agent will adjust its w_{bp} weights and update the trust value for each NAC agent and apply it in its decision making process for the next action by the same NAC.

When one of a NAC's actions is first observed its desirability is unknown and the "trusting" ML agent has to rely on its computed trust value to proactively determine what to do. An ML agent without a trust feature will have to allow the NAC action to determine its utility regardless of whether the associated NAC is trustworthy or not. Thus, the real difference between the two agents is only during the first use of a new action by the NAC.

Each time a NAC action occurs, the agent's pain either increases or decreases as a result of the action. First, a NAC announces which action it wants to perform. If the ML agent does nothing, NAC will perform its action, either delivering pain or reward to the agent. And if the desirability of the action was unknown, the agent will learn it based on the action's result. Alternatively, if the ML agent interrupts the NAC and stops its action (good or bad) and the action desirability is unknown, it will remain unknown.

The next subsection outlines the pseudo-code of the algorithm and shows where and how equation (6) is used to compute the trust value.

B. Algorithm for Trust Development in ML Agent

A brief outline of trust development algorithm's operation is as follows:

1) Initialize the algorithm.
   a) Set parameters such as number of runs, number of time steps/events each run, ratios of good vs. bad actions, etc.
   b) Initialize NAC actions and randomly determine which actions are good or bad based on defined parameters.
   c) Initialize history data for actions taken, trust levels, rewards, etc.
   d) Initialize the ML agent by zeroing trust values, w_{bp} values, desirability settings, etc.
2) Execute a run of either a trusting or untrusting agent.
   a) Randomly select an intended NAC action (only one action can be executed each time step.)
   b) The ML agent determines what to do with the NAC action based on trust values.
      i) If the action desirability is unknown and the ML agent trusts the NAC, the action is performed.
      ii) If the action desirability is unknown and the ML agent distrusts the NAC, the action is blocked.
      iii) If the action desirability is known, the ML agent decides what to do based on the action's desirability.
   c) If the ML agent allows the NAC to execute the action, it receives reward or punishment based on the action's result.
   d) w_{bp} values are updated based on the change in the agent's pain levels (3).
   e) If the action's desirability was unknown it will now be determined based on the received reward and is recorded by the ML agent.
   f) The ML agent updates its internal trust levels according to equation (6).
   g) History values are updated (action number, trust level, reward, etc.)
   h) Repeat step 2 until the number of time steps specified at initialization have been completed.
3) Collect and plot results.

The preceding algorithm indicates how the experiments presented in the next subsection were executed. As mentioned, the algorithm was designed to determine the effect of trust on an agent's pain levels over time and the potential impact trust will have on an agent's behavior and learning.

C. Simulation Experiment

Experiment 1:

First, we tested how trust values change over time as the ML agent experiences good and bad actions from a NAC. Each

NAC has a predetermined level of *friendliness* that has values between 0 and 1.

In simulation, we fixed γ_u at 1 and changed the γ_d value. With $\gamma_d = 0.7$ the results are shown in Fig. 2, where the NAC friendliness changes from 0.1 to 1. Changing the trust factor permits us to build more or less trusting agents, which will influence their overall behavior.

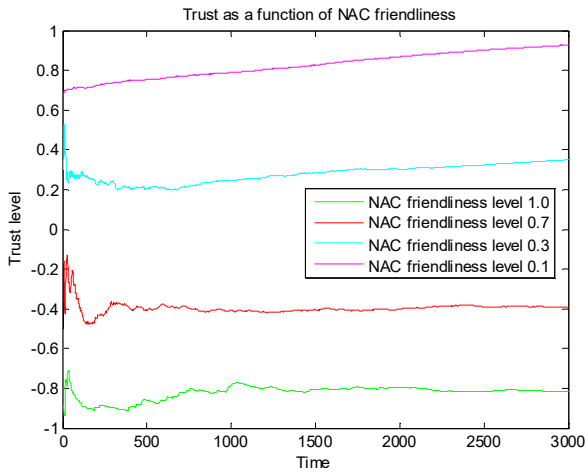

Fig. 2. Trust values computed by the ML agent for the various NACs using (6) $\gamma_d = 0.7$.

The trusting agent benefited from estimating trust for various NACs and this resulted in **a higher level of reward received** as its trust tended to correctly reflect the friendliness of the NAC.

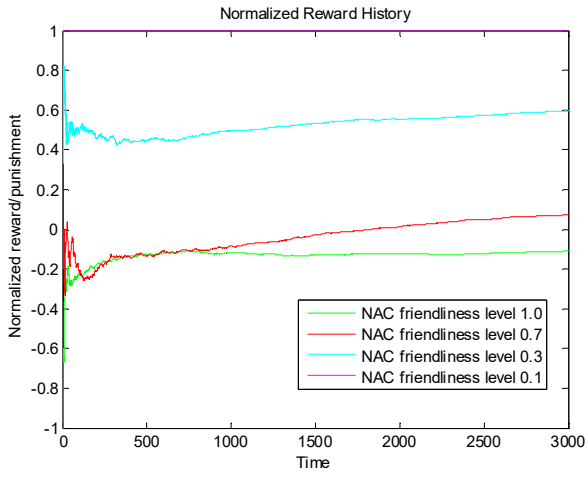

Fig. 3. Normalized reward received by the ML agent from various NACs using $\gamma_d = 0.7$.

Fig. 3 shows the normalized reward/punishment that the agent received after it used the NAC's trust measure (6) with $\gamma_u = 1$ and $\gamma_d = 0.7$. The maximum reward that the agent could get is normalized to 1 (and punishment is normalized to -1). The ML agent significantly benefitted from using trust as it did not allow actions from unfriendly agents to hurt it, limiting the punishment to stay above -0.2 for most of the time and for all NACs (even those with friendliness close to 0).

Experiment 2:

In the second experiment the motivated learning agent observed actions by two NACs working in the same environment. Eighty percent of the first NAC's actions were undesired, while eighty percent of the second NAC's actions were desired. So their friendliness was set at 0.2 and 0.8 respectively. We set $\gamma_d = 0.6$, $\gamma_u = 1$. The trusting agent would learn to trust the second NAC while distrusting the first one. Therefore, it would block new actions from the first NAC while allowing new actions by the second NAC, even though 20% of these actions were undesirable and could hurt the agent. This simulation was repeated 100 times and average performance was calculated. Fig. 4 shows average changes in the trust level for the two NACs over time.

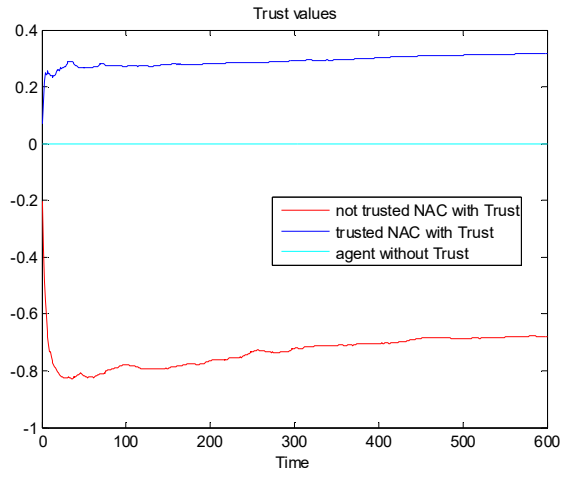

Fig. 4. Average trust values computed by the ML agent for the two NACs using (6).

Fig. 5 shows the average of the normalized reward that the ML agent received together with 2 standard deviation bands that estimate the confidence intervals of the result. We see that the ML agent that blocked actions by the NAC it did not trust while allowing actions by those it did, received higher average reward than ML agent without the trust feature.

However, a very interesting, if somehow unexpected, result was observed when the simulation was extended for several thousand iterations. First, we observed that the trust value for the friendly NAC gradually increased as it continued to deliver desirable actions, while the mistrust for the first agent was maintained or eased a little, as we can see in Fig. 6.

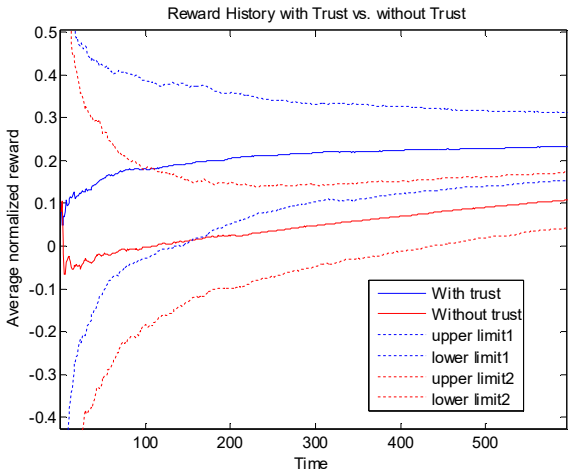

Fig. 5. Average normalized reward over time.

This is in agreement with (6) and (3) as w_{bp} weights gradually increase for trusted agent increasing the trust value.

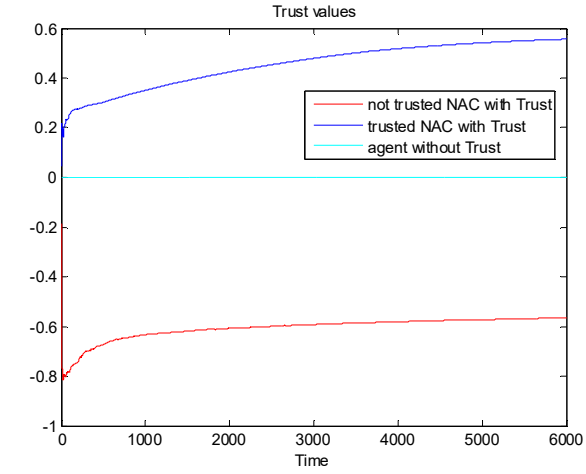

Fig. 6. Trust values over longer period of time.

But an unexpected result is observed in Fig. 7, where, after 2000 iterations, the average reward received by the ML agent without the trust measure started to exceed the reward obtained by the ML agent with the trust measure. The difference was statistically significant. This was the result of the ML agent learning which actions by the first NAC that had 80% undesirable actions were good and which were bad. Recognizing good actions, the ML agent would not interfere in the first NAC's operations, and as a result, its reward was higher then the reward obtained by the ML agent using the trust option. The ML agent with the trust option would not trust the NAC, and block all of its new actions, therefore, it never learned which of the new actions that NAC performed were beneficial to it. This purely numerical result may teach us something about human nature, as it are in line with the proverb "know your enemy" and also the related one "trust but verify". It teaches us to that it may be better to observe and learn about an enemy than to always fight it off.

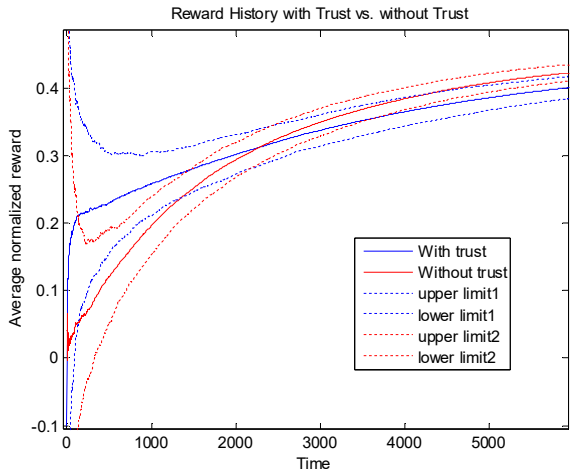

Fig. 7. Reward history over longer period of time.

However, if the friendliness of the first NAC was reduced to 0.03 then such a crossover did not happen as demonstrated in Fig. 8, where using trust was always more beneficial than not using it.

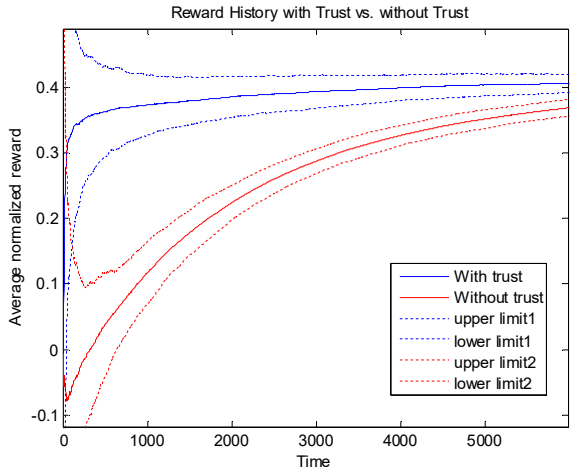

Fig. 8. Reward history with friendly and unfriendly NACs over longer period of time.

In this paper, we investigated how trust can benefit the ML agent. Other characteristic features (like shyness, bravery, etc) may be developed in a similar way. An extension of this work will yield a social agent whose interaction with other agents is well regulated through rules that are understood and accepted as useful by the agents.

IV. Development of Character Traits

It is noticeable that people behave differently in various situations. There are a few reasons for this. One of them is that a person's character that can be described by various traits and associated needs, which also determine one's will to trust.

During previous research [7], [10] twenty traits of human character were distinguished and described. All of these traits can be easily recognized by text corpora or body language

analysis as well as by observing other behaviors, choices or decisions.

Regarding ML agents, we can take into account a few traits, most affecting trust and its development. The ML agent should follow some important rules that affect humans and make them more inclined to trust.

Trusting Rules regarding personality traits for intelligent ML agents:

1. Make proposals but do not decide without consultation with your partner (people do not fully trust ideas that they have not participated in).
2. Make sure that all important aspects of all adversaries have been taken into account (people do like and do not trust decisions that do not take into account their important needs).
3. Try to understand the most important personality traits and needs of your adversaries and adapt your proposals and decisions to them. Try to behave as having similar traits as your adversaries (people treat behaviors that correspond to their traits as professional and trustworthy).
4. Try to take into account all the needs characteristic for personality traits of your adversaries (people subconsciously trust and strive to cooperate with others of the same of similar traits because it satisfies their needs and make the cooperation easier).
5. Do not try to dominate your adversaries or make them to do something if you want to cooperate with them (people do not trust anybody that act with force or push others).
6. Appreciate people you work with for their capabilities of their nature and personality traits (people trust more the others when they are appreciated).

Intelligent ML agents could be even more capable of adapting their behavior to human adversaries than humans can, because ML agents can be programmed to automatically recognize and reflect human traits and make the cooperation more easygoing and trustworthy.

Human traits can develop throughout life, making some needs or behaviors more significant and important. Such development is associated with individual intelligence that can strengthen or weaken some expressions of human traits. People can also try to understand and learn other trait behaviors to adapt better to other people in order to cooperate with them more efficiently. Intelligent ML agents should be able not only to adapt to human traits but also have an impact on motivating people as well as themselves. Motivation signals are important factors for developing skills and intelligence in people as well as automatic intelligent agents and systems. Thus, ML agents should be able to associate adversaries with their traits and needs in order to choose a proper set of behaviors and algorithms to cooperate with them. Intelligent agents should also develop their knowledge about the positive and negative effects of the recognized traits on people. This allows for generalization and drawing conclusions about future ways of acting with people.

V. Conclusions

This paper introduced a concept of trust in motivated learning agents. We defined trust based on experience with a non-agent character and proposed a simple measure to evaluate it. Using our trust measure, the ML agent can benefit by blocking new actions performed by an untrusted NAC. We demonstrated that such a strategy pays off at the initial stage of interaction with NACs. If interaction is lifelong, and the trust measure is higher than 0, a better strategy may be to learn more about the unfriendly NAC and use this knowledge to benefit the ML agent. However, all actions by very unfriendly NACs with a very low trust measure should always be blocked. We consider trust to be a natural step towards introducing a trusted teacher that would accelerate learning in an ML agent. Trust is also related to other features of an agent's character including emotions and various traits of character that modify its behavior.

References

[1] D. Gambetta, Can We Trust Trust? In: Gambetta, D. (ed.) Trust: Making and Breaking Cooperative Relations, Department of Sociology, University of Oxford, chapter 13, 2000, 213-237.

[2] G. Matthews, I.J. Deary, M.C. Whiteman, Personality Traits ,2nd ed., Cambridge University Press, 2003.

[3] A. Baier, Trust and antitrust. Ethics, vol. 96, pp. 231-260. Reprinted in: Moral Prejudices. Cambridge University Press, 1986.

[4] Kurt T. Dirks, Donald L. Ferrin, The Role of Trust in Organizational Settings, 2001.

[5] H. Buss, R. Plomin, Temperament (PLE: Emotion): Early Developing Personality Traits, Psychology Press, 2014.

[6] B. W. Roberts, N. R Kuncel, R. Shiner, A. Caspi, L. R. Goldberg, "The power of personality: The comparative validity of personality traits", Perspectives on Psychological Science, vol.2 (4), pp.313-345, 2007.

[7] R. Tadeusiewicz, A. Horzyk, Man-Machine Interaction Improvement by Means of Automatic Human Personality Identification, Springer-Verlag, LNCS 8838, 2014, pp. 278-289.

[8] C.G. Jung, Psychological Types, The collected works of C.G.Jung, vol. 6, eds. R.F.C. Hull, Princeton University Press, 1971.

[9] I.B. Myers, P.B. Myers, Gifts differing: understanding personality type, CA: Davies-Black Publishing, Mountain View, 1995.

[10] A. Horzyk, R. Tadeusiewicz, A Psycholinguistic Model of Man-Machine Interactions Based on Needs of Human Personality, Man-Machine Interactions, Proc. of ICMMI 2009, Springer, Advances in Intelligent and Soft Computing 59, 2009, pp. 55-67.

[11] S. Sac-Tang, V. Esichaikul, "Web Personalization Techniques for E-commerce", Active Media Technology, Springer, 2011, pp. 36-44.

[12] J. A. Starzyk, "Motivation in Embodied Intelligence" in Frontiers in Robotics, Automation and Control, I-Tech Education and Publishing, Austria, 2008, pp. 83-110.

[13] J. A. Starzyk, "Motivated Learning for Computational Intelligence, in *Computational Modeling and Simulation of Intellect: Current State and Future Perspectives*, B. Igelnik, Ed., IGI, ch.11, pp. 265-292, 2011.

[14] J. A. Starzyk, J. Graham, L. Puzio, "Needs, Pains, and Motivations in a Simulated Learning Agent," accepted in IEEE Trans on Neural networks and Learning Systems, 2016.

[15] J. A. Starzyk, J. Graham, "MLECOG - Motivated Learning Embodied Cognitive Architecture" IEEE Systems Journal, (Vol. PP , Issue 99), 2015, pp. 1-12. DOI:10.1109/JSYST.2015.2442995.

[16] J. Graham, J. A. Starzyk, Z. Ni, H. He, T-H. Teng, and A-H. Tan, "A Comparative Study between Motivated Learning and Reinforcement Learning," The Int. Joint Conf. on Neural Networks, Killarney, Ireland, July 12 – 17, 2015.