Comparative Study of Neural Networks Techniques in the Context of Cooperative Observations

Matheus S. Araújo[1], Leonardo F. da Costa[1], Thayanne F. da Silva[1], Raimundo C. F. Junior[1], João P. B. Andrade[1] and Gustavo A. L. de Campos[1]

[1]State University of Ceará
Silas Munguba Avenue – 60.714.903 – Fortaleza – CE – Brazil

{math.araujo,leonardo.costa,thayanne.silva,junior.ferro,

joao.bernardino}@aluno.uece.br,gustavo@larces.uece.br

Abstract. *In the Cooperative Target Observation (CTO) problem, a group of moving observers should monitor a group of moving targets to maximize the average number of observed targets. The majority of computational approaches to the CTO considers that the observers are rational agents and that the targets are only naive agents. This work incorporates a model of the observers' behavior in the targets' decision-making system, considering four basic models of neural networks trained, to improve their performance. The results showed that the target team's performance increased when they were modeled as rational agents, mainly when the model incorporates basic models of recurrent neural networks compared to classic feed-forward approaches.*

1. Introduction

The phenomenon of globalization brought with it several advantages in people's lives, but also disadvantages and their impact on security issues. The tasks of recognition, monitoring and surveillance have become an important point in ensuring security, which is increasingly threatened by globalized societies. In the context of surveillance, some new technologies have been used to support the task of Cooperative Target Observation (CTO). This problem is a variant of the Cooperative Multi-Robot Observation of Multiple Moving Targets (CMOMMT) problem, a NP-Complete problem that is in the core of surveillance problems in environments where there are more targets than observers [Parker 1999].

In the CMOMMT, although the number of observers is smaller than the number of targets, the speed of observer agents is greater than the maximum speed of targets. The targets do not have sensor and the observers sense a small range of the total environment, i.e., a two dimensional lattice with entrances and exits. In the CMOMMT problem the observers must maximize the observation time of the robots, minimizing the total time in which the targets escape the observation of the group of observers. They must work collectively over time so that each target is observed by at least one of them. Differently from the original CMOMMT problem, in the CTO problem the environment is totally observable.

Currently, observer agents are implement by different technologies as, for example, Unmanned Ground Vehicles (UGV) [S. Andrew Gadsden Stephanie Bonadies 2016],

Unmanned Aerial Vehicles (UAV) [K. Kanistras and Valavanis 2013] and Unmanned Underwater Vehicles (UUV) [Sullivan and Luke 2004]. These technologies carry the resources for detection, information processing, and communication. Although powerful technologies to operate in a variety of environment tasks, these kinds of vehicles have restrictions to perform targets observation, such as: low observation range, difficulty in communication between involved entities, the complexity to perform the task when there are more targets than observers, among others.

The first approach to the CTO problem was based on the comparison and combination of two algorithms, the k-means (KM) clustering algorithm, and the hill-climbing (HC) search algorithm [Luke et al. 2005]. The focus was on the rationality of the movement of the observers. The targets were considered to move randomly by the environment. Considering the actual scenario for surveillance problems, others approaches tho the problem assume that targets can be as rational as the observers are. They focused on the targets movement, e.g., considering the straight-line movement and one kind of controlled randomization movement [Rashi Aswani and Paruchuri 2017].

Our work is a new approach trying to improve the rationality of the targets.The work incorporates a model of the observers' behavior in the targets' decision-making system trying to improve their performance. The model tries to predict the observers' new positions based on the local information that the targets have about the current positions of their neighbors. Although our initial motivation was to generate new research challenges for the observers' side, the approach can be adapted to case to different situation as, for example, in the case a military personnel in missions where it is necessary to obtain information about a person, a group or an area of interest, but avoiding being caught by predicting the actions of an opposing team.

The work proposes to evaluate four basic models of neural networks, trained considering data obtained from simulations of the interactions of the agents in a lot of different configurations of the task environment. The approach proposes four strategies for the target team, two inspired by classic feed-foward neural networks, i.e., the Multilaye Perceptron and the Radial Basis Function Network, and two inspired by partially recurrent neural networks, i.e., the Elman and Jordan networks. The results showed that modeling targets as intelligent agents has a negative impact on the performance of observers, especially when targets employ strategies based on the recurrent networks.

This paper is structured in four more sections. Section II presents the main theoretical references taken as a basis for the design of the approach. Section III describes the proposed approach. Section IV describes the experiments to measure the performance of targets and observers in the CTO problem. Section IV presents the results. Finally, Section V concludes the paper with the final remarks, limitations, and future researches.

2. Theoretical Reference

2.1. Cooperative Target Observation Problem

The CMOMMT is the core of sensor positioning problems in an environment where there are more targets than observers, in order to maximize the observation coverage of the targets given a region of observation [Parker 1999]. The CMOMMT problem considers a team of m robots with 360° of view observation sensors and a team of n targets in a two-dimensional, bounded, enclosed spatial region, with input and output. Observers work

as a cooperative team, seeking to observe the maximum of targets under a limited radial sensor range, while the targets move randomly.

The CTO is one of the reformulations of the CMOMMT, allowing all observers to know the location of other observers and all targets [Luke et al. 2005]. As in the CMOMMT, the environment task is a continuous, 2D, non-toroidal, obstacle-free rectangular field containing N observers and M targets, with N ¡ M [Luke et al. 2005]. Just as in the original problem, the observers try collectively to remain within a "range of observation" of as many targets as possible, and the targets move at random and are slower than the observers.

Formally, the objective of the observers is to maximize the average of observed targets during the observation time interval. So, at each time step t, the performance of each observer is computed by the following function [Luke et al. 2005]:

$$\theta(\omega) = \begin{cases} 1, if\ \omega\ is\ monitored\ at\ t, \\ 0, \qquad otherwise. \end{cases} \tag{1}$$

where ω represents a target. Then, the number of targets observed by the observers at each time-step is computed by:

$$\mu(t) = \sum_{\omega \in \Omega} \theta(\omega) \tag{2}$$

in which Ω represents the set of all targets being observed. So the average value considering all time-steps in the interval of observation, the function to be maximized by the observers, can be computed by:

$$\bar{\mu}_{ANOT} = \frac{1}{T} \sum_{t \in T} \mu(t) \tag{3}$$

where T represents the interval of observation. experiment. We call the μ_{ANOT} value of Average Number of Observed Targets (ANOT), and this value was considered as the performance measure of the observers, as used by [Luke et al. 2005]. For the purpose of this work, the metric used will be the Average Number of Target Evasion (*ANTE*), which consists in subtracting the Total Amount of Targets (*TAT*) with $\bar{\mu}_{ANOT}$ to evaluate the success of target evasion, i.e.:

$$\bar{\mu}_{ANTE} = TAT - \bar{\mu}_{ANOT} \tag{4}$$

2.2. Approaches to Target Movement

One of the first improvement in the strategy of the targets was the straight line and controlled randomization movements [Rashi Aswani and Paruchuri 2017]. In the straight line movement, the target tries to escape the observer's sensor range by following a straight-line path in the direction that maximizes the distance between them. The direction is calculated by joining a line segment between the current position of the observer and the target. If there is more than one agent observing the target calculates the average position

of the observers and uses it to identify the direction that leads to the maximum distance from them [Rashi Aswani and Paruchuri 2017].

The adjusted randomization motion was created to avoid, in real situations, that the observers suspect from the target movement, i.e., it proposes that the targets must follow the straight line movement for 0.7 time steps and the random movement for the other time steps. As much as the targets are observed in this movement, in real-world situations, it minimizes suspicion about the target agent [Rashi Aswani and Paruchuri 2017].

[Symeonidis and Mitkas 2005] proposed a framework to model the actions that the agents can execute and a mechanism to monitor their actions. This monitoring mechanism occurs through a central agent, in which he notes the actions of the other agents. Then, a pre-processing of the collected data is performed so that patterns are recognized in the common actions of the agents. Two mechanisms were developed, the fist for the storage of these profiles, and the second for searching profiles in previously in the storage. Finally, they proposed a interface for recommendation of actions for the agents. The objective of this work was to use a predictive approach to improve the behavior of the intelligent agents.

The work of [França et al. 2019] examined two approaches to the target team in the CTO problem, one based on organizational paradigms and clustering algorithms, and another one based on Multilayer Perceptron network decentralized algorithm. Both allowed a better performance for the targets, with emphasis on organizational approaches. However, with a simple neural network approach only with MLP and without a statistical analysis of the model or comparation with other networks.

2.3. Neural Networks

Neural networks can be defined as computer systems that learns and infers through data [Russell and Norvig 2009]. For instance, neural networks can make predictions in time series by using a dataset of historical data, e.g., financial transactions, used as input by the network and then is processed by a learning algorithm. At the end of the training, classification or predictions can be done on new data that comes in. In this paper, we model a neural networks to predict the behavior of observer agents. The main goal of this neural networks is to establish a decision-making strategy for target agents. By doing that, the targets will have a better rationale when trying to move or escape from observers. As said in last section, the approach proposes to evaluate two models inspired by classic feed-foward neural networks, i.e., the Multilayer Perceptron and the Radial Basis Function Network, and two models inspired by partially recurrent neural networks, i.e., the Elman and Jordan networks.

The MLP is a perceptron-like neural network, but with more than one layer of neurons in direct feeding. Such a network is composed of layers of neurons and connections between neurons with weights. Learning in the network type is realized by the error back-propagation algorithm, i.e., the resilient back-propagation (Rprop) [Riedmiller and Braun 1992]. The Rprop takes into account only the partial derivative signal over all patterns, optimizing the number of steps compared to other methods. The topological update function was employed, i.e., the neurons calculate their new activation in a topological order. This means that the first processed layer is the input layer. The next processed layer is the first hidden layer and the last layer is the output layer.

The Radial Basis Function (RBF) networks are feedfoward networks quite similar to MLP. The main difference being that the RBF has only a single hidden layer, whose neurons have a gaussian activation function. The RBF network can approximate any continuous function through the linear combination of Gaussian functions with centers at different positions of the input space. The weights are initialized with a specific procedure: it first selects evenly distributed centers from the loaded training patterns and assigns them to the connections between input and hidden layer. The update function used is topological such as MLP.

The recurrent neural network (RNN) is a class of neural networks that includes weighted connections within a layer. Because RNNs include loops, they can store information when processing new entries. This memory makes them ideal for processing tasks where the previous entries are to be considered as spatial temporal data and the record of the previous movement of the observing agents taken into account to predict their next move.

The Elman neural network (EN), in addition to the input, intermediate and output units there are also context units as, in general, occurs in the recurrent networks. The input and output units interact with the external environment, while the intermediate and context units do not. The input units are just storage units that pass the signals without modifying them. The output units are linear units that add up to the signals they receive. In this work the intermediate units have non-linear activation functions. Context units are used only to memorize the previous activation of the intermediate units and can be considered as time delay in one step. The feed-forward connections are adaptable, and the recurrent connections are fixed [Braga et al. 2000].

In the Jordan Network (JN), the network output is copied to the context unit. In addition, the context units are locally recurrent. The big difference in terms of topology between the two networks is that the recurrence in the EN is made from the hidden layer to the inputs, whereas in the JN the recurrence is made from the outputs to the inputs.

3. Approach Description

Our approach is a reformulation of others approaches to CTO, i.e., those that allow targets to act as rational agents during the whole task. In our approach, from the observers point of view, there is a coordinator agent that computes and sends the new destination position to each observer every γ time steps. The observers' next position is computed employing the k-means clustering algorithm [Luke et al. 2005] [Rashi Aswani and Paruchuri 2017]. Each observer agent moves toward the computed destination position and continuously wait until a new destination point is sent. If one observer reaches its destination point in less than γ time steps, then it waits until a new destination is received.

From the point of view of the target agents, they have sensors similar to the observers [Rashi Aswani and Paruchuri 2017], and can move in the same manner as them. In order to refine the target's decision-making system, this paper proposes to employ basic models of neural networks to predict the observers' rationality related to its next movement in CTO problems. More specifically, the approach is a comparative study between these basic models. We decomposed the approach in two phases that must be implemented in sequence. The goal of the first phase is to learn how to predict the observers' next position in simulated environments. The goal of the second phase is to validate the

targets' learned predictions in similar environments, but in different scenarios. Fig. 1 illustrates our approach.

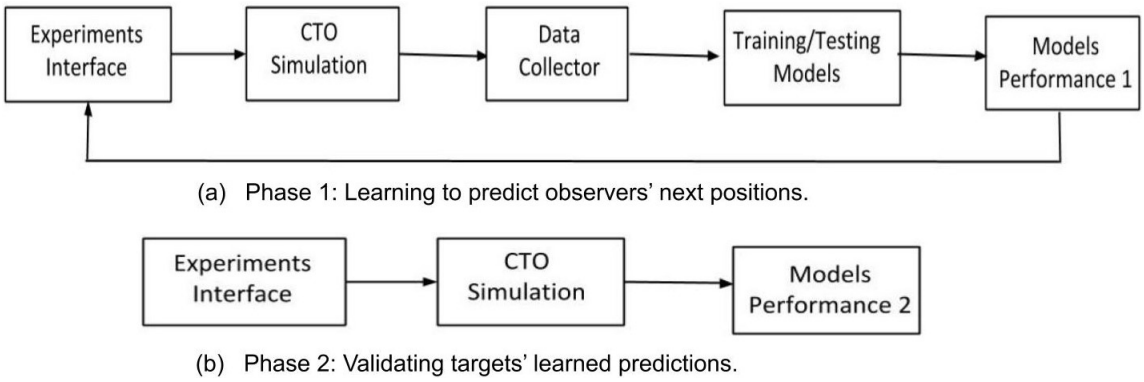


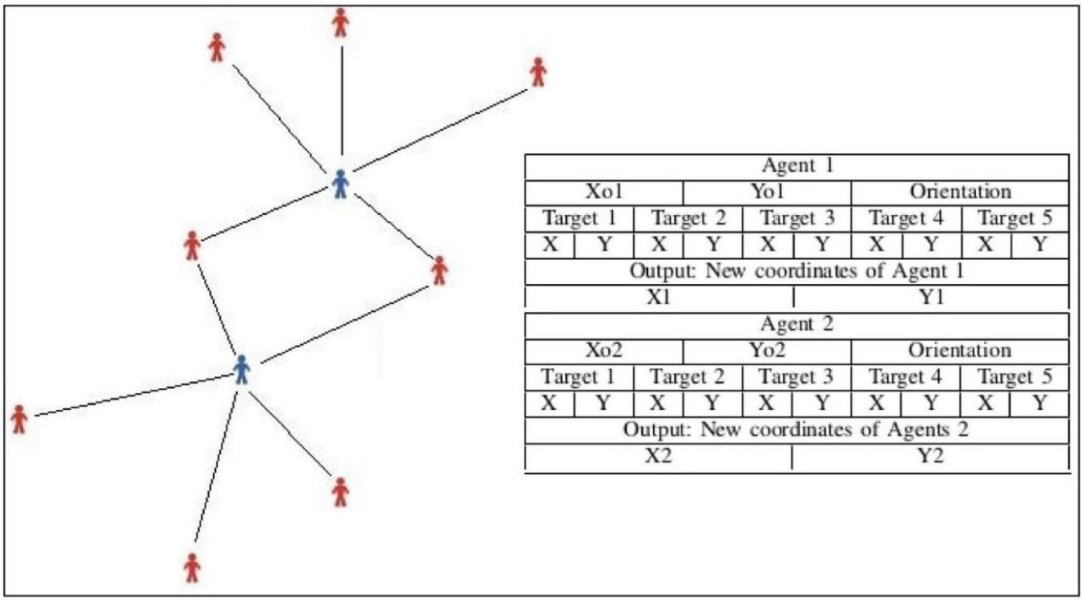
Figure 1. Approach to evaluate basic models of neural networks

In the first phase, the approach focus on the training process of neural networks and with their validation by means of the performance measure called Performance 1. It considers that there is an interface to elaborate experiments employing techniques of agent-based simulation, to collect data generated by simulations of the interactions between observers and targets in a lot of virtual scenarios. The data collected is normalized and divided into the training set and the testing set.

We collect data by monitoring all the agents when a lot of simulations were being configured in the experiments. This data was collected by a central agent that has access to all information of the given simulation, e.g., coordinate positions, direction, speed, etc. By having access to all these data, what remains are to define what features are going to be chosen. The data collection approach is represented in Fig. 2. The blue agents are observers and the red ones are targets. By trial and error, we determine that a better approach to predict observer behavior is to see which targets agents are neighbors to each one of them. In this example, we pick the five targets closest to each observer agent. For instance, we see that the agent 1 has five targets pointed out by the arrows as well as the agent 2.

Thus, we first annotate the coordinate position x and y of the current observer and its orientation, i.e., the direction the agent is pointing. Moreover, we collect the coordinate position of the n closest targets as well as their current speed. From 17 (seventeen) columns defined for the collection of data, 15 (fifteen) of them constitute the set of attributes, whose values will be feed to the input layers of ours neural networks. Considering the input attributes, the output attributes, whose values will be feed by the output layers of our networks, is defined in the two remaining columns, i.e., the coordinate of the observer's next position observing the scene. For each time in the simulation we fill a number of rows (cases) equal to the number of observers, as the data is collected for every observer agent. The table in Fig 2 shows the required input and its output information formatting the collected data, for instance, when time is 100 and the number of observers is 10, we have approximately 35000 rows.

The paper presents the approach considering the evaluation of four basic neural network models, i.e., two feed-forward networks and two recurrent networks: Multi-layer



Agent 1									
Xo1		Yo1		Orientation					
Target 1		Target 2		Target 3		Target 4		Target 5	
X	Y	X	Y	X	Y	X	Y	X	Y
Output: New coordinates of Agent 1									
X1				Y1					

Agent 2									
Xo2		Yo2		Orientation					
Target 1		Target 2		Target 3		Target 4		Target 5	
X	Y	X	Y	X	Y	X	Y	X	Y
Output: New coordinates of Agents 2									
X2				Y2					

Figure 2. Schema for collecting data to generate our datasets

Perceptron, Radial Basis Function, Elman and Jordan. Before being read by the networks the data are normalized with mean zero and variance one. Then split into a cross validation of 80% for training and 20% for test to fit the models and overfit verification. In relation to the performance measure in the first phase, at the end of the simulations we propose to perform the non-parametric Wilcoxon test and the comparison of root-mean-square deviation (RMSE) to measure the differences between the result values and verify the prediction ability of the networks. The Wilcoxon Signed-Rank Test is a non-parametric hypothesis test used when comparing two related samples and verifies whether or not the networks are significantly different in their predictions.

More specifically, for the statistically reliable comparison test we used the model described in [Veras 2013] which compares different configurations of three types of neural networks for behavioral prediction of the winds speed: the Multilayer Perceptron and the recurrent networks of Elman and Jordan. The Wilcoxon nonparametric test was used to perform the comparative tests. The Wilcoxon test verifies the magnitude of the difference between two data by comparing the medians of the samples to see if they are significantly different. This test assumes that the distribution of differences is symmetric. If the networks are similar in the test, it is performed the model of persistence with the comparison of performance of the RNA architectures for the prediction of winds speed, employing the mean squared error (EMQ) as metric, in our work we use the RMSE.

In relation to the second phase, after the prediction modelling realized in the first phase, it focuses on the validation of the neural networks predicting the observers' next positions online, on new scenarios configured in the experiments interface and employing a second performance measures, similar to the first one, but extending it by a new performance value, i.e., that considers expression of $\bar{\mu}_{ANTE}$ defined in the theoretical references, that we call Average Number of Target Evasion (ANTE). So, for the purpose of this work, the greater the targets evasion, the more efficient the will be the strategy adopted by the targets and greater the value of performance measure Performance 2.

4. Experiments and Results

The approaches to compute the target's next position were evaluated by means of conducting experiments in the NetLogo platform [Wilensky 1999], i.e., employing agent-based simulation to measure the performance of the targets in a virtual environment that can be programmed to represent different task complexity. The NetLogo platform was chosen because it allows running parallel experiments and easy integration with R [R Development Core Team 2008] software, More specifically the open source library RSNNS for the implementation of neural networks [Bergmeir 2018]. The simulations were configured as follows:

- The targets and observers are in a rectangular field with dimensions 150x150 units;
- Number of time steps per simulation equal to 400;
- The velocity of the observer is 1 step by time interval;
- Target velocity varies between 0.10, 0.25, 0.50, 0.75, 0.90 step by time interval;
- Sensor-range varies between 5, 10, 15, 20, 25 units;
- k-means parameter γ fixed in 15;
- Number of targets equal to 24;
- Number of observers equal to 12.

As said previously, in all experiments, we considered the strategy adopted in [Luke et al. 2005] to control the observers' movement, i.e., employing the k-means algorithm as the main source of information to compute the observers' next position. Thus, four different strategies, based on the four basic neural networks models presented in this paper, were programmed to generate the results about the targets' movement. To ensure reliability, simulations are repeated ten times, for all networks.

In relation to the k-means parameter γ, we fixed its value 15, since this value produced better results in the tests, comparing with the results obtained with the values 5 and 10. We considered five different values for setting the velocities, varying the sensor-range considering a set of five values. Before measuring the ANTE (Performance 2) in the second phase of the approach, at the end of the simulations we performed the non-parametric Wilcoxon test and the comparison of root-mean-square deviation (RMSE) to measure (Performance 1) the prediction ability of the networks in the first phase of the approach.

For the case of MLP, the hidden layer with 35 neurons performed better than others. The logistic function was employed as activation function of the neuron in the hidden layer. Almost 3000 iterations were required for the model to be trained. In our study the RFB is based on a heuristic proper to the Gaussian functions and 17 neurons or centers to be grouped, since from successive tests we arrived at the conclusion that a K=17 reduced the training errors and the errors of adjustments of the centers. In RBF it was necessary to include the constant momentum with value 0.3 to increase the rate of learning without oscillation, only 2000 iterations were necessary in training.

For modeling the CTO problem, the Elman and Jordan networks were designed with 35 neurons in the hidden layer as well as the Multi-layer Perceptron. However, with 35 context neurons in Elman regarding the storage of the hidden layer units, and 2 context neurons in the Jordan network regarding the storage of the output units. The definition of the number of neurons in the hidden layer was based on performance of the

MLP network. In our recurrent models we used the Rprop or resilient backpropagation adapted for partial recurrent networks, similar to MLP approach, because as detailed in the Theoretical Reference the Rprop has a good performance and a fast convergence. The update function used for Elman and Jordan networks propagates a pattern from the input layer to the hidden layer and finally to the output layer. After this follows a synchronous update of all context units. Both recurrent networks were trained in 3000 iterations as well as MLP.

Employed in the first phase, the Wilcoxon Signed-Rank Test a non-parametric hypothesis test used when comparing two prediction samples and verifies whether or not the networks are significantly different through a pairwise comparison, i.e .:

- $\mathbf{H_0}$: there is no difference between the medians of the sample predictions.
- $\mathbf{H_1}$: the medians of the two samples are different.

We considered the significance level equal to 0,05 (5%) for the tests. As shown in Table 1, using the dataset test sample, we conclude that all networks are statistically similar according to the hypothesis $\mathbf{H_0}$, since all paired tests were higher than 0.05.

Table 1. Results of Wilcoxon Test

Networks	***MLP***	***RBF***	***ELMAN***	***JORDAN***
MLP	X	0.117	0.673	0.668
RBF	X	X	0.160	0.109
ELMAN	X	X	X	0.550

Soon after, we compared the RMSE of the predictions of the networks to better measure forecasts accuracy and finalize performance 1. The RMSE were employed to measure the quality of the models. This value is computed by the square root of the mean difference between predicted values and the actual position of the observers according to:

$$RMSE = \sqrt{\frac{1}{n}\Sigma_{i=1}^{n}\Big(d_i - f_i\Big)^2} \tag{5}$$

where *n* represents the number of predictions, and d and f represent the predicted position and the actual position values respectively.

The effect of each error on RMSE is proportional to the size of the squared error. RMSE is always non-negative, and a value of 0 (almost never achieved in practice) would indicate the perfect fit to the data. In our study, the RMSE of all predictions of all the target agents during the simulations varies from 6 to 11. Considering the mean RMSE in all simulations, we concluded that the recurrent networks obtained the better performance, i.e., they have smaller RMSE. According to Table 2, the Jordan Network performed better than the others and RBF the worst performance.

Table 2. Mean RMSE of the predictions of all simulations

Network	RMSE
JORDAN	6.768
ELMAN	7.701
MLP	9.790
RBF	10.291

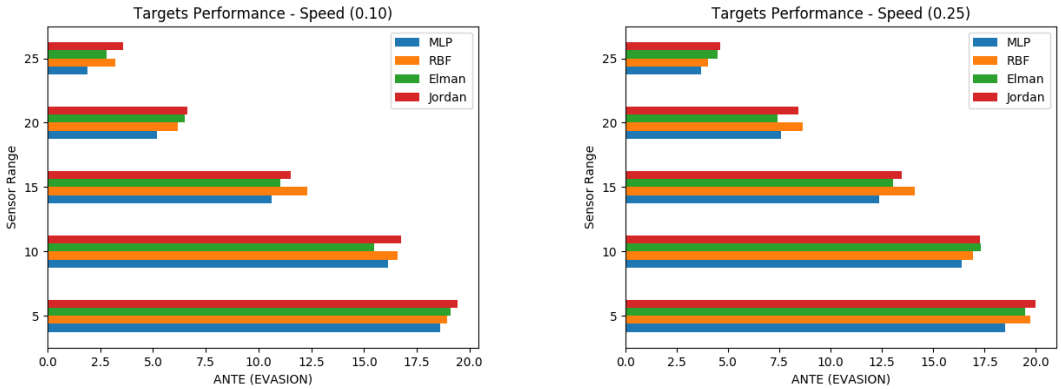


Figure 3. Performance to 0.1 and 0.25 speed-targets and different sensor ranges.

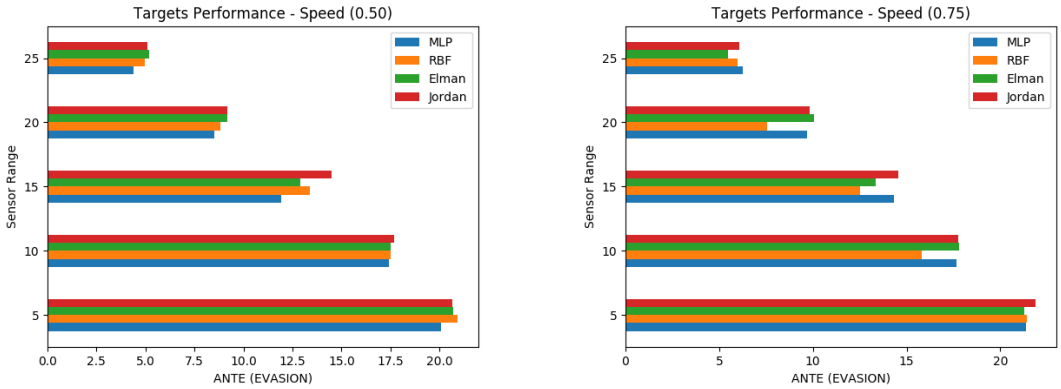


Figure 4. Performance to 0.5 and 0.75 speed-targets and different sensor ranges.

In the second phase we tried to observe the effect of the results, measured by the RMSE metric in the first phase, have in the results measured with the *ANTE* metric or Average Number Evasion Of Targets, i.e., if the neural networks improved the taget's performance, that is, increasing the values of ANTE, maximizing the evasion of targets.

The best network will be the one that obtains the greater *ANTE*, therefore obtains lowest reach of the targets. As seen in Fig. 3 in the lower speeds RBF outperformed MLP performance, while the Jordan Network achieved the best performance in almost all sensor ranges. The Elman Network in turn has fluctuated a lot in the performance, being sometimes worse than the RBF or MLP.

Figure 5. Performance to 0.9 speed-targets and different sensor ranges.

At higher speeds, as shown in Figures 4 and 5, sometimes the MLP obtained higher *ANTE* in relation to RBF, but both were quite similar in most configurations. In general, the Jordan Network obtained the best avoidance of targets which is a reflection of the lowest prediction error (RMSE) of the network as described in Table 2.

Elman's network in the majority of cases was the second highest in average evasion of the targets, although in some configurations RBF or MLP managed to overcome it. The MLP and the RBF were the networks with lower performance, having the RBF exceeded the MLP in most of the configurations, but with a small difference.

5. Conclusions

This paper examined four approaches to the target team in the CTO problem involving different neural network techniques. All networks improved the targets' performance, hindering the task of observers because of the evasion of targets, decreasing the *ANOT* of the observers. Although by the Wilcoxon test seems to be similar, the prediction errors (RMSE) of the recurrent networks were smaller, being the best approach to predict the behavior of the observers, especially the Jordan network the best network with the lowest error, and a higher average target avoidance *(ANTE)*.

Although classical neural network methods have achieved satisfactory results in some situations, they are limited in dealing with continuous geographical distances between sites over time intervals, close to sequential data modeling. This work presents a better modeling of spatial and temporal information in an approach in which, instead of considering only the current position of the agent, the target also considers previous positions in the definition of the next move. To do this, we used the class of partially recurrent neural networks to predict the movement of the observers with better quality given the ability of these networks to store in the context units the previous activation of the intermediate units, and can be regarded as a time delay in one step. A recurring feature yields better forecast result.

The limitation found was the long training time the neural networks takes. Another important point is that the networks are fixed, so they do not update the training in real time to improve performance. We also intend to implement deep recurrent neural networks such as LSTM (Long Short Term Memory) to predict observer movements in

scenarios with obstacles.

References

Bergmeir, C. (2018). Neural networks using the stuttgart neural network simulator (snns).

Braga, A. P., Ludermir, T. B., and Carvalho, A. C. L. (2000). *Redes Neurais Artificiais – Teoria e Aplicações*.

França, T., Leite, J. L. A., Junior, R. J. C. F., da Costa, L. F., de Souza, R. P., Andrade, J. P. B., and de Campos, G. A. L. (2019). Smart targets to avoid observation in cto problem. *Proc. of the 18th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2019)*.

K. Kanistras, G. Martins, M. J. R. and Valavanis, K. P. (2013). A survey of unmanned aerial vehicles (uavs) for traffic monitoring. *In 2013 International Conference on Unmanned Aircraft Systems (ICUAS)*.

Luke, S., Sullivan, K., Panait, L., and Balan, G. (2005). Tunably decentralized algorithms for cooperative target observation. pages 911–917.

Parker, L. E. (1999). Cooperative robotics for multi-target observation. intelligent automation soft computing.

R Development Core Team (2008). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.

Rashi Aswani, S. K. M. and Paruchuri, P. (2017). Improving surveillance using cooperative target observation. *Thirty-First AAAI Conference on Artificial Intelligence*.

Riedmiller, M. and Braun, H. (1992). Rprop-a fast adaptive learning algorithm. *ISCIS*.

Russell, S. and Norvig, P. (2009). *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, Upper Saddle River, NJ, USA, 3rd edition.

S. Andrew Gadsden Stephanie Bonadies, A. L. (2016). A survey of unmanned ground vehicles with applications to agricultural and environmental sensing.

Sullivan, K. M. and Luke, S. (2004). Autonomous uuv control via tunably decentralized algorithms. *2004 IEEE/OES Autonomous Underwater Vehicles*, IEEE Cat:47–53.

Symeonidis, A. and Mitkas, P. (2005). A methodology for predicting agent behavior by the use of data mining techniques. *International Workshop on Autonomous Intelligent Systems: Agents and Data Mining*.

Veras, C. V. A. (2013). Estudo comparativo de técnicas de redes neurais artificiais na previsão da velocidade do vento em curto prazo.

Wilensky, U. (1999). *Netlogo*. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston, IL.