Multi-Objective Reinforcement Learning–based Deep Neural Networks for Cognitive Space Communications

Paulo Victor R. Ferreira*, Randy Paffenroth†, Alexander M. Wyglinski*, Timothy M. Hackett‡, Sven G. Bilén‡, Richard C. Reinhart§, and Dale J. Mortensen§
* Department of Electrical and Computer Engineering
Worcester Polytechnic Institute, Worcester, MA, USA
prferreira@wpi.edu, alexw@wpi.edu
† Department of Mathematical Sciences, Department of Computer Science and Data Science Program
Worcester Polytechnic Institute, Worcester, MA, USA
rcpaffenroth@wpi.edu
‡ School of Electrical Engineering and Computer Science
The Pennsylvania State University, University Park, PA, USA
tmh5344@psu.edu, sbilen@engr.psu.edu
§ Space Communications and Navigation
NASA John H. Glenn Research Center, Cleveland, OH, USA
richard.c.reinhart@nasa.gov, dale.mortensen@nasa.gov

***Abstract*—Future communication subsystems of space exploration missions can potentially benefit from software-defined radios (SDRs)controlled by machine learning algorithms. In this paper, we propose a novel hybrid radio resource allocation management control algorithm that integrates multi-objective reinforcement learning and deep artificial neural networks. The objective is to efficiently manage communications system resources by monitoring performance functions with common dependent variables that result in conflicting goals. The uncertainty in the performance of thousands of different possible combinations of radio parameters makes the trade-off between exploration and exploitation in reinforcement learning (RL) much more challenging for future critical space-based missions. Thus, the system should spend as little time as possible on exploring actions, and whenever it explores an action, it should perform at acceptable levels most of the time. The proposed approach enables on-line learning by interactions with the environment and restricts poor resource allocation performance through 'virtual environment exploration'. Improvements in the multi-objective performance can be achieved via transmitter parameter adaptation on a packet-basis, with poorly predicted performance promptly resulting in rejected decisions. Simulations presented in this work considered the DVB-S2 standard adaptive transmitter parameters and additional ones expected to be present in future adaptive radio systems. Performance results are provided by analysis of the proposed hybrid algorithm when operating across a satellite communication channel from Earth to GEO orbit during clear sky conditions. The proposed approach constitutes part of the core cognitive engine proof-of-concept to be delivered to the NASA Glenn Research Center SCaN Testbed located onboard the International Space Station.**

***Index Terms*—Satellite communication, machine learning, artificial intelligence, reinforcement learning, neural networks, cognitive radio, space communication, SCaN Testbed, NASA GRC**

This work was partially supported by: NASA John H. Glenn Research Center, grant number NNC14AA01A; NASA Space Technology Research Fellowship, grant number NNX15AQ41H; and CAPES Science without Borders scholarship, grant number BEX 18701/12-4.

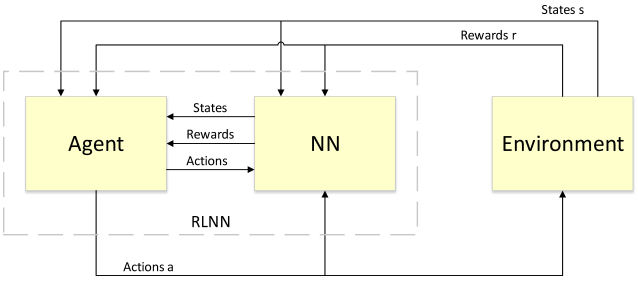

Fig. 1: Hybrid multi-objective reinforcement learning and deep neural network (RLNN) block diagram. 'Virtual exploration', driven by 'action rejection' probability, prevents time expenditure exploring "bad actions" using radio resources over-the-air.

I. Introduction

Software-defined radio (SDR) [1], a communications technology that is frequently used in terrestrial applications, has only started to appear in space as part of a testbed located on-board the International Space Station (ISS) in 2012 through a research project initiative proposed, developed, and led by the Space Communications and Navigation group at the NASA John H. Glenn Research Center [2–5]. Known as SCaN Testbed, it is comprised of three SDRs operating at S-band and Ka-band.

As discussed in [6, 7], the flexibility enabled by SDRs also bring complexity to operations. The radios have a large

number of parameters to modify link configurations and to improve performance, but these changes could add risk to a mission through operator burden or misconfiguration To reduce the complexity of SDRs, optimize the performance and operation of SDR communications, and investigate autonomous communications operations these on-orbit flexible radios are now being used to enable research on cognitive space communications, the next frontier on communications systems.

These on-orbit flexible radios are now being used to enable research on cognitive space communications, which is the next frontier in communications systems. A cognitive radio [8], which possesses environmental awareness across different network layers [9], is capable of autonomously performing perception, learning, and reasoning activities in order to optimize resource allocations on individual nodes or distributed throughout the network based on its current hardware and software capabilities, channel conditions, mission phases, and network conditions.

For instance, several technologies have been developed in order to achieve a full cognition level. At a very basic level, currently they enable: (i) radio parameter adaptation such as conventional adaptive coding and modulation (ACM) [10] employed by standard such as DVB-S2 in order to mitigate fading [11], and (ii) spectrum sensing for dynamic channel access by optimizing spectrum utilization of temporarily vacant frequencies [12–14].

In order to enable full cognition in radios, machine learning [15–17] has become an important technology in wireless communications, especially with respect to leveraging online learning methods. Several techniques for performing resource allocation optimization have been proposed in the open literature, including genetic algorithms (GA) [18]. The main drawback of these techniques is the convergence time, including no convergence guarantee for online operations over dynamic channels. Additionally, learning is not conveyed over time and each different channel condition requires the algorithm be executed. Knowing when to exploit a certain action, recognizing a different channel condition, and running the GA again could potentially result in spending the majority of time on exploring radio parameter sets possessing poor performance, herein referred to as actions. Techniques such as these, in their purest form, seem unfeasible for critical space-based missions due to the challenging dynamic environments affecting the channel between the spacecraft and the ground station [19–21]. Recently, research on machine-learning techniques with cognitive radios via a case study have focused on the learning problem [15, 16], with the majority of the research being focused on terrestrial cognitive radios [17] or on space communications resource allocation [22] and spectrum sensing [12, 13].

Seeking to tackle the learning problem for space communication systems, a solution was proposed in [22] for optimizing conflicting multi-objectives when selecting multi-dimensional radio resource allocations for space communications. In this case, a pure Reinforcement Learning (RL) [23, 24] solution was considered. In this work, we build upon that solution in order to solve the problem of the RL spending too much time on exploring actions that result in low performance scores. We also propose a hybrid algorithm comprising of a RL and neural network [25]. Known as RLNN, this proposed algorithm enables the radio to predict the effects of multi-dimensional radio parameters on multi-dimensional conflicting performance goals before allowing the radio to actually try these parameters over the air, avoiding the cost of spending time and resources on learning action–performance mapping that will not be useful in the near future.

This paper is organized as follows: Section II provides a brief overview of the machine learning techniques used to build the proposed hybrid solution, Section III describes the proposed RLNN algorithm, Section IV presents simulation results, and Section V provides concluding remarks.

II. Machine Learning Overview

Machine learning (ML) defines a set of techniques that allows computer systems to learn certain tasks after being presented with several examples. There are three main ML categories: (i) supervised learning, (ii) unsupervised learning, and (iii) reinforcement learning [23, 24]. The basic difference between (i) and (ii) is that in (i) the examples are labeled and in (ii) the examples do not have labels. These algorithms are trained for classification, such as pattern recognition or regression such as in function approximation. RL is a special case in which the algorithm learns how to achieve goals by interacting with an uncertain environment. Below we briefly describe (i) and (iii) in more detail.

A. Neural Networks Overview

An artificial neural network (NN) is a method for mapping inputs to outputs and usually is used for classification, such as in pattern-recognition problem, or for use in non-linear function approximation such as in function fitting [25]. For instance, in this paper, a NN is used to approximate the non-linear environmental effects by mapping actions into rewards. Several improvements have been made to the NN algorithms, which is composed of two main steps: training and prediction. Initially, examples containing input and output data are preprocessed and provided to the NN for training. After meeting some minimum performance requirements, the trained NN, which consists basically of the NN architecture and its weights, can be used as a predictor.

For a detailed description of NN basics, derivations, and algorithm details, the interested reader is referred to Chapter 6 in [26]. When using NNs, each different problem requires a specific NN architecture, comprised of a training function; performance metrics, a number and size of hidden layers such that its usage becomes feasible for the desired application in terms of required processing capabilities and processing time, and generalization of error performance. Currently, there are no general guidelines in the literature on how to pick these items. Reference [27] provides some useful comments and advice on what to consider when making these decisions.

In this paper, we consider the standard multi-layer fully connected NN trained by a backpropagation-based algorithm. More details on the chosen NN architecture are provided in Section III.

B. Reinforcement Learning Overview

The contents of this section provide a brief summary of relevant concepts underpinning RL found in [23] for the proposed algorithms presented in the following sections. RL is an algorithm designed to learn through interactions with the environment in a trial-and-error fashion, shown in Fig. 1. Based on predefined goals, RL looks for actions that optimize its performance.

The Multi-armed bandit (MAB) [28–31] models RL problems in which an action set results in rewards, which represents a measurement of how well a certain task was executed. Thus, it can be seen as an optimization problem to find the action set that results in the maximum reward.

Instead of using MAB, these problems could be modeled as state-transition problems. The state-transition itself is modeled as a Markov decision process (MDP) [32]. State-transitions can be deterministic, *i.e.*, executing a certain action will always lead the system to that same state, as assumed in this paper, or it will make the next state to behave as a random variable.

Usually control problems require the computation of an optimal policy that maps observed states into actions that will be taken when the system is in one of those states. Thus, the work presented in this paper is concerned about controlling radio parameters such that optimal performance is achieved based on the current environmental conditions and kept there for the entire time, such as regulator. The environment is comprised of the satellite communications channel through which propagating signals are affected by the dynamic geometry of the line-of-sight between the transmitter and the receiver and its surroundings (buildings in the vicinity of the ground stations or structures in the vicinity of the antennas on-board the spacecraft), as well as the dynamics of the atmospheric and space weather. Therefore, a state-transition model and the action-state mapping takes all these variables into account and it is assumed to be unknown due to its high level of complexity.

Fortunately, there are several techniques to compute policies, for which most of the time the environment model, *i.e.*, state-transition model, is unknown due to being too complex or difficult to obtain. In this case, the agent must interact with the environment in an efficient way to find the best policy possible while balancing exploration of new actions and exploitation of known actions. In these cases, an action-value function $Q_\pi(s,a)$ representing the value of a certain action a taken when in state s while following policy π, should be evaluated for all actions possible from state s through a greedy policy given by:

$$\pi(s) = \arg\max_a Q(s,a), \tag{1}$$

where for every state $s \in S$ an action $a \in A$ with maximal action-value is chosen given the state space S and action space A. For several problems with either a continuous or discrete A containing thousands of actions a, it may not be feasible to evaluate all action-values when in a certain state s. This is the case of radio communications, for which exploring each action a over the air may take time and force the radio receiver to experience a certain performance degradation. The practical alternative is to ensure that the agent keeps exploring them using either on-policy or off-policy approaches. On-policy approaches evaluate or improve policies used to make decisions, whereas off-policy methods evaluate or improve a policy that is learned about, known as a target policy, that is different from a policy used to generate behavior, known as a behavior policy [23].

A common model-free method to find these policies is Temporal-Difference (TD), which updates the action-value function $Q(s,a)$ using past experiences at each time step, suitable for on-line, *i.e.*, time-sensitive applications. The on-policy TD control is known as State-Action-Reward-State-Action (SARSA) and updates Q by computing:

$$\begin{aligned} Q_{k+1}(s_k,a_k) \;=\; & Q_k(s_k,a_k) + \\ & \alpha[r + \gamma Q(s_{k+1},a_{k+1}) - Q(s_k,a_k)], \end{aligned} \tag{2}$$

where α is the learning rate, r is the reward, γ is the discount factor, s_{k+1} and a_{k+1} are the state and action chosen the current target policy, before the Q update. The difference within the brackets in Eq. (2) is known as the TD error, and it computes the difference between the estimated value of $Q(s_k,a_k)$ and a better estimate, $r+\gamma Q_k(s_{k+1},a_{k+1})$. The off-policy TD control algorithm is known as Q-learning and is computed by:

$$\begin{aligned} Q_{k+1}(s_k,a_k) \;=\; & Q_k(s_k,a_k) + \\ & \alpha[r + \gamma \max_a Q_k(s_{k+1},a) - Q_k(s_k,a_k)], \end{aligned} \tag{3}$$

where the TD error uses the Q-value with the highest value independent of the action. Eqs. (2) and (3) are derived from the well-known Bellman equations [23, 32].

As mentioned in [22], within the context of decision-making in radio communications, the discounted factor does not have a practical meaning since the cognitive radio is interested in the immediate reward ($\gamma = 0$) and any action can be taken from any state without the need for planning. These assumptions result in a modified version of the Q-value functions for both on- and off-policy, which turn out to be the same, given by:

$$Q_{k+1}(s_k,a_k) = Q_k(s_k,a_k) + \alpha[r_k - Q_k(s_k,a_k)]. \tag{4}$$

Even though the state-transition model is unknown, several functions need to be defined, for instance, the action exploration function, the state-action policy function h used in $a_k = h(s_k)$, and the reward function ρ, used in $r = \rho(s_k,a_k)$.

III. Proposed Solution

The proposed hybrid method consists of the RL mentioned in Section II-B, which as proposed in [22], and the standard multi-layer NN mentioned in Section II-A, herein

referred to as as NN-based RL, or RLNN. It considers a multi-dimensional parameter optimization of radio configurations seeking to achieve the best multi-objective performance possible given the current satellite communication channel conditions.

The NN is used as an approximation of the environment in terms of the RL experience, the chosen actions, and its respective rewards achieved so far. By having the luxury of approximating the mapping of actions into states and rewards, the NN allows for actions to be explored all at once, or as many as one would like to, without having to actually spend time trying those actions in the real environment. We call this new approach 'virtual exploration.' It improves the exploration performance by removing the time the RL agent would spend exploring actions that are predicted to yield poor performance. This additional feature is called 'action rejection'.

Poor actions are defined by a 'rejection performance threshold' value defined by the user such that actions resulting in performance below that threshold are classified as poor. The action rejection probability defines the time percentage that bad actions will be rejected during exploration, *i.e.*, prevented from being used over the air by the radio. Whenever the RL agent is exploring, it predicts the actions' performance using the trained NN and classifies them into either good or bad. Then, according to the rejection probability it randomly chooses one action from either a good or bad set.

Regarding the NN architecture, a feedforward with Levenberg-Marquardt backproapgation training algorithm described in [33, 34] was used. The NN has three fully-connected layers without bias: two hidden layers that contain 7 and 50 neurons each, both using a log-sigmoid transfer function, and the output layer with one neuron using the standard linear transfer function. With respect to the performance function, the mean-squared error was used with two different training stop conditions: minimum error gradient of 10^{-12} and maximum validation checks equal to 20. During training, the data was randomly split into 70% for training, 15% for testing, and 15% for validation, all scaled across a $[-1, 1]$ range.

In order to improve the NN prediction error, an ensemble of 20 NNs were used during both training and prediction, as shown in Fig. 2, and the output was simply the average among all these NNs. The reason for the choice of this amount of NNs constituting the ensemble is also made using the mean-square error as a performance metric, similar to the way the NN architecture itself is chosen, due to the lack of a more formal theoretical method.

This hybrid approach is depicted in Fig. 1 where the RL interacts with the ensemble of NNs, receiving the same actions sent to the environment and the same rewards and state information from the environment during training. When used for prediction, this information is exchanged only with the RL agent, avoiding the cost of executing such actions in the real-world environment.

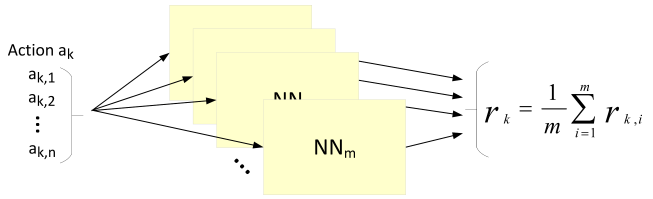

Fig. 2: Deep neural networks used for ensemble learning of radio transmitter parameters, represented by multi-dimensional actions, and their mapping into multi-objective reinforcement learning performance, represented by rewards.

IV. Simulation Results

In order to comply with the DVB-S2 standard, the radio-adaptable parameters are the same as defined in [11], considering all four modulation schemes (QPSK, 8-PSK, 16-APSK, and 32-APSK) and their respective encoding schemes. In addition to all roll-off factors, the following were considered: bandwidth range of $[0.5 - 5$ MHz$]$, additional variable transmission symbol power range of $[0 - 10$ dB$]$ in steps of 1 dB, and long-frame with frame length equal to $64{,}800$ bits. The action space is comprised of more than $30{,}400$ possible actions. Each action vector $\bar{a}$ is composed of six parameters a_n, where $n = 1, \cdots, 6$: symbol rate (Rs), energy per symbol (Es), roll-off factor (rof), modulation order (M), number of bits per symbol (k), and encoding rate (er).

The GEO satellite channel is assumed to be an AWGN during clear sky conditions, with adaptation taking place on the downlink to a fixed ground station only, similar to the channel presented by the authors in [22]. In these simulations, for proof-of-concept purposes it was assumed that the satellite's transmitter amplifier operates in the close-to-linear region.

Regarding performance, the conflicting multi-objective target considered is comprised of six parameters: bit error rate (BER) estimated at the receiver, throughput (Thrp), bandwidth (BW), spectral efficiency (Spc_eff), consumed power (Pwr_con), and power efficiency (Pwr_eff) measured at the transmitter and sent over to the receiver, all of which are scaled to the range of $[0, 1]$. The reward function ρ is computed by the fitness function f_{obs} given by the weighted sum computed by:

$$\begin{aligned} f_{obs}(x) =& w_1 f_{\text{Thrp}} + w_2 f_{\text{BER}} + w_3 f_{\text{BW}} + \\ & w_4 f_{\text{Spc\_eff}} + w_5 f_{\text{Pwr\_eff}} + w_6 f_{\text{Pwr\_con}}, \end{aligned} \tag{5}$$

where x is a vector containing the performance parameters, described above, and the weights w_i for each performance parameter, specified according to each different communications mission and defined by the user. The following simulation results considered all $w_i = 1/6$.

As mentioned in Section II-B, the action exploration functions used in this paper are: (i) constant exploration probability $\epsilon = 0.5$ and the well-known ϵ-greedy exploration algorithm [35, 36] with variable exploration probability $\epsilon = 1/k$, where k is the step size between resets of ϵ back to 1 whenever it reaches a minimum, in this case assumed to be equal to

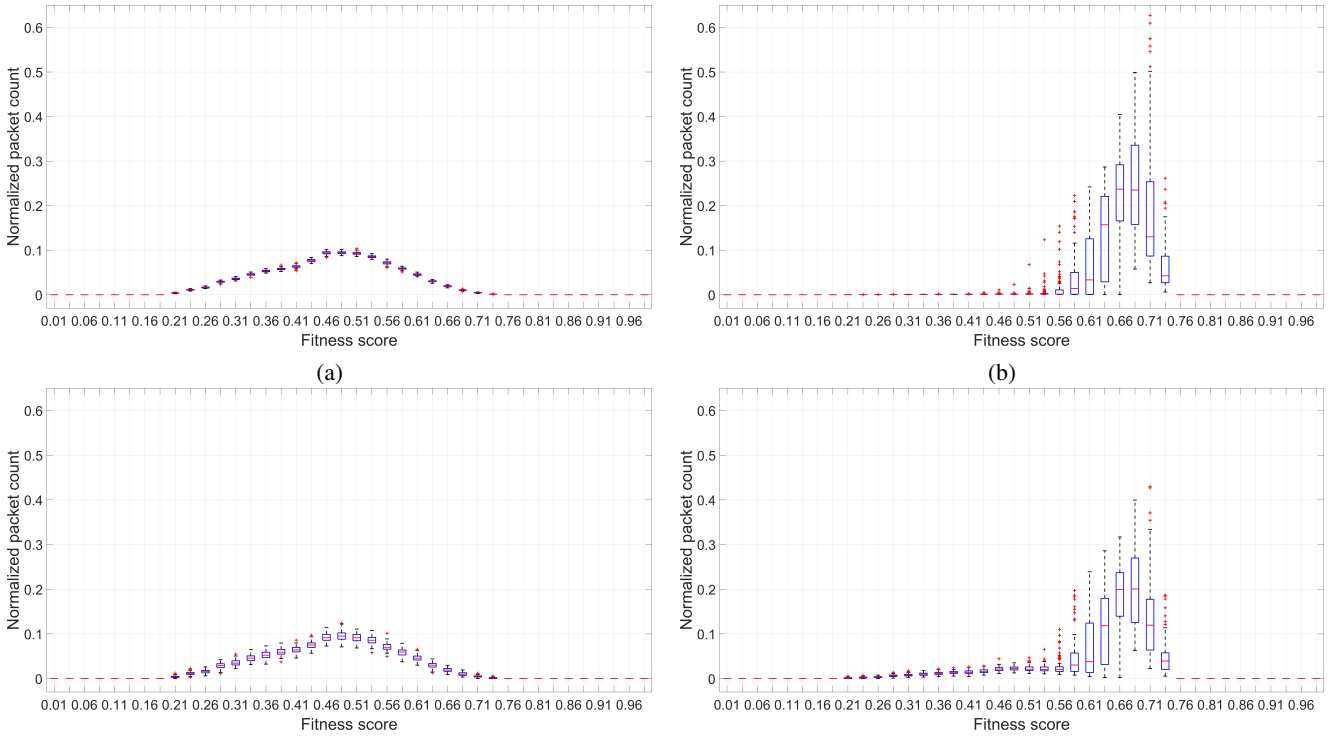

Fig. 3: Boxplots of proposed RLNN algorithm multi-objective performances. On the left panels virtual exploration was turned off (likewise the authors' algorithm proposed in [22] considering the DVB-S2 standard), and on the right panels it was turned on with rejection probability equal to 1. Top panels used fixed exploration probability value equal to 0.5, and bottom panels used the variable exploration probability function. High number of packets concentrated around larger fitness scores is better.

4×10^{-3}. In this proposed hybrid solution, the state-action policy function h is approximated by the NN during virtual exploration followed by uniform random sampling between the bad and good action sets, based on the rejection probability value. During exploration, h is greedy and chooses the action with the maximum Q-value, computed by Eq. (4), with $\alpha_k = \alpha_{k-1}/2$, another user-defined parameter that decreases from 1 until it meets a threshold of 10^{-3}, when it gets reset.

Simulations were run for the hybrid RLNN algorithm proposed in this paper and for a modified version of the RL algorithm proposed by the authors in [22], this time considering DVB-S2 and additional adaptable parameters as mentioned above. Fig. 3 presents the average distribution of the amount of network packets according to their fitness score for both algorithms while using exploration probabilities equal to $\epsilon = 0.5$ and $\epsilon = 1/k$. This distribution accounts for performance during exploration only. A total of 100 simulations were run for each of the four different configurations (combinations between exploration functions and virtual exploration set on/off), with the same simulation duration of 512 seconds. Even though the channel considered in these simulations is assumed constant over time (no slow or fast fading), this time duration represents the average duration of a LEO orbit and may allow performance comparisons to be done in future research by the authors.

As expected, the introduction of the NN for virtual exploration allows the radio to drastically decrease the time spent, and consequently the number of packets, on exploring actions that resulted in poor performance when compared to the maximum performance achieved while rejecting all those actions predicted to perform below a threshold. This improvement can be seen as a shift to the left on the distributions shown on the right-hand side panels in Fig. 3. For these results the rejection performance threshold was considered to be equal to 95% of the current maximum performance predicted by the NN. The virtual exploration feature was disabled and enabled by setting the rejection probability to 0 and 1, respectively, meaning that 0 no action is to be rejected and 1 all actions with performance below the selected performance threshold will be rejected.

In terms of resultant numerical performance, in scenarios with virtual exploration disabled, the average number of packets experiencing multi-objective performance values above 0.56 when using a fixed and variable exploration probability values, was 33% and 25% respectively, as shown by Fig. 3 panels (a) and (c). In both scenarios, the majority of packets experienced a performance score value of 0.485.

However, in scenarios with virtual exploration enabled, the average number of packets experiencing multi-objective performance values above 0.56 when using a fixed and variable exploration probability values, was 82% and 98%,

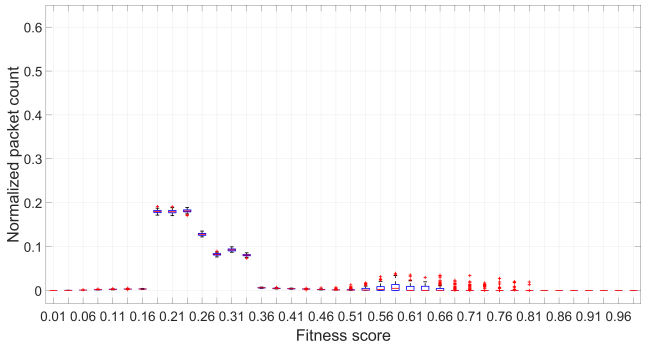

Fig. 4: Boxplot of the genetic algorithm multi-objective performance during the same time duration used by the proposed RLNN simulations. The majority of the time is spent in very low performance levels.

respectively, as shown by Fig. 3 panels (b) and (d). In both scenarios, the majority of packets experienced a performance score value of 0.685. This represents an improvement on number of packets experiencing performance values above 0.56 of 2.48 times and 3.92 times for fixed and variable exploration probabilities, respectively.

In terms of the integral values of the average of histograms, both scenarios with virtual exploration disabled have an integral equal to 0.472, while the scenarios with that feature enabled have an integral equal to 0.67 and 0.62, for fixed and variable exploration values, respectively. Thus, improvements of 1.32 times and 1.42 times on the integral values were achieved by the proposed RLNN for fixed and variable exploration probability values, respectively.

It should be noted that using a fixed exploration probability value, the amount of packets used during exploration represented 50.58% and 50.7% of the total for scenarios with and without virtual exploration, respectively. When using the variable exploration function these percentage values were 3.17% and 3.12%, respectively. These values combined with the distributions shown in Fig. 3 and with the improvements on the integral values demonstrates the effectiveness of the proposed virtual exploration in increasing the number of packets experiencing high multi-objective performance values independently of the exploration probability function chosen.

For comparison, a GA simulation was run for 100 times, each with the same time duration of the RLNN simulations mentioned above. Its average performance distribution is shown in Fig. 4. Although it was able to achieve higher performance scores than the proposed RLNN, the cost to achieve that was to spend 66% of the time exploring actions that resulted in very low performance values, scored between 0.18 and 0.26. Only 0.8% of the time was spent on performance values between 0.69 and 0.81.

Batch methods, such as GA approaches, might have an advantage over the standard RL and our proposed RLNN for the cases when the environment remains constant and/or the system can spend a long time exploring a large number of different actions, resultant from different GA generations. However, if the environment changes, a reset may be required, which will result in the system having to spend a considerable amount of time experiencing low performance values after restarting the search again. If the system can not wait until the GA convergence, it might stick with using an action that can have any performance level.

Even though evolutionary methods might be good as searching methods, they do not guarantee a minimum performance. Our proposed RLNN method does not guarantee a specific performance level either. However, through virtual exploration it provides guidance to which actions to explore, giving control over the performance levels experienced during exploration by performing action rejection. In addition to that, through the rejection probability value, the RLNN provides control over the amount of time spent on actions that may result in a certain performance level.

V. Conclusions

In this paper, a hybrid learning architecture for multi-objective radio resource allocation was proposed using reinforcement learning and neural networks, resulting in the RLNN algorithm. The main goal of this architecture is to provide control over which actions to be explored based on their predicted multi-objective performance, as well as to control the time spent on exploring actions with performance values above a threshold defined by the user.

Simulations were run for scenarios with fixed and time-varying exploration probabilities, considering a satellite equipped with DVB-S2-compliant adaptable radios and an AWGN channel, assuming reconfiguration on the return link only. In scenarios with virtual exploration enabled, the the majority of packets experienced a performance score value very close to the maximum score values achieved throughout the simulation, independently of the exploration function being used.

Numerical simulation results showed that the proposed RLNN algorithm improves of 2.48 times and 3.92 times the number of packets experiencing performance values above 0.56 for fixed and variable exploration probabilities, respectively, when the assumptions made are considered. An overall improvement of 1.32 times and 1.42 times on the integral values of the average performance distribution curves for these two scenarios also demonstrate that a larger amount of packets have their performance values concentrated around higher values during exploration.

Since there was no difference in performance regarding the exploration probability functions used, future research or applications might consider using the proposed RLNN algorithm with different exploration strategies and still take advantage of the improvements shown in this paper.

References

[1] S. G. Bilén, A. M. Wyglinski, C. R. Anderson, T. Cooklev, C. Dietrich, B. Farhang-Boroujeny, J. V. Urbina, S. H. Edwards, and J. H. Reed, “Software-defined radio:

A new paradigm for integrated curriculum delivery," *IEEE Communications Magazine*, 2014.

[2] NASA Glenn Research Center, *SCaN Testbed*, NASA Glenn Research Center. Available at http://spaceflightsystems.grc.nasa.gov/SOPO/SCO/SCaNTestbed/.

[3] S. K. Johnson, R. C. Reinhart, and T. J. Kacpura, "CoNNeCTs approach for the development of three software-defined radios for space application," *Proceedings of IEEE Aerospace Conference*, pp. 1–13, 2012.

[4] R. Reinhart, T. J. Kacpura, S. K. Johnson, and J. P. Lux, "NASA's Space Communications and Navigation Testbed aboard the International Space Station," *IEEE Aerospace and Electronic Systems Magazine*, vol. 28, no. 4, 2013.

[5] R. C. Reinhart, "Using International Space Station for cognitive system research and technology with space-based reconfigurable software-defined radios," *66th International Astronautical Congress*, 2015.

[6] D. Chelmins, J. Downey, S. K. Johnson, and J. Nappier, "Unique challenges testing sdrs for space," *Proceedings of the 2013 IEEE Aerospace Conference, Big Sky, Montana*, 2013.

[7] S. Johnson, D. Chelmins, D. Mortensen, M. Shalkhauser, and R. Reinhart, "Lessons learned in the first year operating software defined radios in space," *Proceedings of the 2014 IEEE Aerospace Conference, Big Sky, Montana*, 2014.

[8] E. Hossain, D. Niyato, and D. I. Kim, "Evolution and future trends of research in cognitive radio: a contemporary survey," *Wiley Wireless Communications and Mobile Computing*, vol. 15, pp. 1530–1564, 2013.

[9] J. F. Kurose and K. W. Ross, *Computer Networking A Top-Down Approach*. Pearson, 2013.

[10] A. Morello and V. Mignone, "DVB-S2: The second generation standard for satellite broadband services," *Proceedings of the IEEE*, vol. 94, no. 1, pp. 210–227, 2006.

[11] "DVB-S2 Standard," available at https://www.dvb.org/standards/dvb-s2.

[12] S. Chatzinotas, B. Ottersten, and R. D. Gaudenzi, *Cooperative and Cognitive Satellite Systems*. Elsevier, Academic Press, 2015.

[13] S. K. Sharma, S. Maleki, S. Chatzinotas, J. Grotz, and B. Ottersten, "Implementation issues of cognitive radio techniques for Ka-band (17.7-19.7 GHz) SatComs," *7th Advanced Satellite Multimedia Systems Conference and the 13th Signal Processing for Space Communications Workshop*, 2014.

[14] T. Yucek and H. Arslan, "A survey of spectrum sensing algorithms for cognitive radio applications," *IEEE Communications Surveys and Tutorials*, vol. 11, no. 1, pp. 116–130, 2009.

[15] A. He, K. K. Bae, T. R. Newman, J. Gaeddert, K. Kim, R. Menon, L. Morales-Tirado, J. Neel, Y. Zhao, J. H. Reed, and W. H. Tranter, "A survey of artificial intelligence for cognitive radios," *IEEE Transactions on Vehicular Technology*, vol. 59, no. 4, 2010.

[16] N. Abbas, Y. Nasser, and K. E. Ahmad, "Recent advances on artificial intelligence and learning techniques in cognitive radio networks," *EURASIP Journal on Wireless Communications and Networking*, 2015.

[17] M. Bkassiny, Y. Li, and S. K. Jayaweera, "A survey on machine-learning techniques in cognitive radios," *IEEE Communications Survey and Tutorials*, vol. 15, no. 3, pp. 1136–1159, 2013.

[18] S. Chen, T. R. Newman, J. B. Evans, and A. M. Wyglinski, "Genetic algorithm-based optimization for cognitive radio networks," *IEEE Sarnoff Symposium*, 2010.

[19] G. Maral and M. Bousquet, *Satellite Communications Systems, Techniques and Technology*. John Wiley and Sons, 2009.

[20] M. Richharia, *Mobile Satellite Communications Principles and Trends*. Wiley, 2014.

[21] G. E. Corazza, *Digital Satellite Communications*. Springer, 2007.

[22] P. V. R. Ferreira, R. Paffenroth, A. M. Wyglinski, T. M. Hackett, S. G. Bilén, R. C. Reinhart, and D. J. Mortensen, "Multi-objective reinforcement learning for cognitive radio-based satellite communications," *34th AIAA International Communications Satellite Systems Conference*, 2016.

[23] A. Barto and R. S. Sutton, *Reinforcement Learning: An Introduction*. MIT Press, 1988.

[24] D. M. Roijers, P. Vamplew, S. Whiteson, and R. Dazeley, "A survey of multi-objective sequential decision-making," *Journal of Artificial Intelligence Research*, vol. 48, pp. 67–113, 2013.

[25] S. S. Haykin, *Neural Networks : A Comprehensive Foundation*. Prentice Hall, 1999.

[26] Y. B. Ian Goodfellow and A. Courville, *Deep Learning*. The MIT Press, 2016.

[27] "Neural Networks FAQ," available at http://www.faqs.org/faqs/ai-faq/neural-nets/part2/.

[28] D. A. Berry and B. Fristedt, *Bandit Problems: Sequential Allocation of Experiments*. Springer, 1985.

[29] T. Lai and H. Robbins, "Asymptotically efficient adaptive allocation rules," *Advances in Applied Mathematics*, vol. 6, no. 1, 1985.

[30] S. L. Scott, "A modern bayesian look at the multi-armed bandit," *Applied Stochastic Models in Business and Industry*, vol. 26, no. 6, pp. 639–658, 2010.

[31] J. Langford and T. Zhang, "The epoch-greedy algorithm for multi-armed bandits with side information," *Advances in Neural Information Processing Systems*, vol. 20, 2007.

[32] L. Busoniu, R. Babuska, B. D. Schutter, and D. Ernst, *Reinforcement Learning and Dynamic Programming Using Function Approximators*. CRC Press, 2010.

[33] M. Hagan and M. Menhaj, "Training feed-forward networks with the marquardt algorithm," *IEEE Transactions on Neural Networks*, vol. 5, p. 989993, 1994.

[34] H. Yu and B. M. Wilamowski, *Levenberg-Marquardt Training*. The Industrial Electronics Handbook, 2nd

Edition, 2012.

[35] M. Tokic and G. Palm, "Value-difference based exploration: Adaptive control between epsilon-greedy and softmax," *KI 2011: Advances in Artificial Intelligence: 34th Annual German Conference on AI*, pp. 335–346, 2011.

[36] M. Tokic, "Adaptive ϵ-greedy exploration in reinforcement learning based on value differences," *KI 2010: Advances in Artificial Intelligence: 33rd Annual German Conference on AI*, pp. 203–210, 2010.