Chapter 12

CODA Algorithm: An Immune Algorithm for Reinforcement Learning Tasks

Daniel R. Ramirez Rebollo, Pedro Ponce Cruz and Arturo Molina

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/63570

Abstract

This document presents the design of an algorithm that takes on its basis: reinforcement learning, learning from demonstration and most importantly Artificial Immune Systems. The main advantage of this algorithm named CODA (Cognition from Data). Is; it can learn from limited data samples- that is given a single example and the algorithm will create its own knowledge. The algorithm imitates from the Natural Immune System the clonal procedure for obtaining a repertoire of antibodies from a single antigen. It also uses the self-organised memory in order to reduce searching time in the whole action-state space by searching in specific clusters. CODA algorithm is presented and explained in detail in order to understand how these three principles are used. The algorithm is explained with pseudocode, flowcharts and block diagrams. The clonal/mutation results are presented with a simple example. It can be seen graphically how new data that has a completely new probability distribution. Finally, the first application where CODA is used, a humanoid hand is presented. In this application the algorithm created affordable grasping postures from limited examples, creates its own knowledge and stores data in memory data in memory in order to recognise whether it has been on a similar situation.

Keywords: artificial immune system, artificial intelligence, neuroscience, biologically inspired algorithm, cognition

1. Introduction

Learning in humans involves approximately 100 billion of neurons (brain cells). Neurons in humans are useless individually but extremely useful and powerful when working together.

Each neuron is formed by a body or soma, and axon that sends information and thousands of dendrites for receiving data. The more dendrite connections there are, the more learning there would be. It can be seen than at its basic functional level, learning is without a doubt a complex activity. It involves not only the brain cells and their connections but also certain factors such as attention, memory, motivation and stress. There are even different ways in which learning can occur, such as empiricism, innatism and constructivism [1].

In humans, learning is not only directly engaged with the amount of connection in the neurons but is also influenced by external factors relative to the subject state. There is a necessity to learn in an entity free of unnecessary passions and no underpinnings that could hinder or limit its learning ability. Machine learning is the response to this paradigm, several algorithms have arisen having as main objective; making computers learn as can be seen in [2], even they may have different particular objectives according to their taks. Machine learning algorithms have already demonstrated their competence at learning in different engineering and science problems [3–5]. However, there is still no algorithm that is more versatile and generalised, i.e., a "do-it-all" algorithm that can be used in several situations no matter the nature of the task itself.

In machine learning, there is no single algorithm that could solve all the problems [6]. To address this, a machine learning algorithm from three principles is created: Learning from Demonstration, Reinforcement Learning and Artificial Immune System. This aims to obtain an algorithm with several advantages from the mixture of those techniques. The resulting algorithm would keep all the advantages from the techniques used plus some of the Artificial Immune System characteristics from **Table 2**. **Table 1** shows advantages of CODA.

CODA'S advantages
Low training samples
Short training time
Produces new knowledge
Unnecessary/unused data is deleted
Reward function assures search for maximum reward
Reward function is task specific
Self-organised memory mechanism reduce searching time in repertoire

Table 1. Advantages that characterise CODA algorithm.

This document is organised as follows. Section 1 contains a discussion on the theory on how learning from demonstration, reinforcement learning and Artificial Immune Systems has been used to develop CODA algorithm in order to help the reader understand how useful these methods are for the algorithm presented in Section 3.

http://dx.doi.org/10.5772/63570

Section 5 explains the CODA algorithm by presenting the pseudocode and simulations. Section 6, on the other hand, explains the application where CODA will be used. Sections 7 and 8 give further discussion and conclusion, respectively, on this chapter.

2. Learning from demonstration

Defining Learning from Demonstration (LfD) should not be difficult. In [7], a simple yet complete sentence clarifies the concept: Learning from Demonstration is learning from watching a demonstration of the task to be performed. Embedded in this sentence is the main goal, which is to learn a skill from demonstrative examples. What is needed is for the computer to learn from demonstrations and this is what it is all about. Learning from demonstration is also known as "programming by demonstration", "imitation learning" or "teaching by showing". As the second name exposes, another goal is to replace certain programming that would be time-consuming and that would require a specialised person in order to modify the programs within the robot. Programming by demonstration promises an automatic self-programming process by showing a robot the task to perform in itself.

Learning from demonstration is not a new topic. It is a well-known discipline in robotics and has been studied in [8–12], where their approaches needed direct teaching to the entity in order for the entity to imitate certain human actions. Recent studies focus on developing the learning from demonstration theory in order to produce better systems that would lead to better demonstration and feedback, resulting in better teaching and learning [13]. Other studies propose a system using a Bayesian nonparametric reward in order to assign rewards to subgoals (more than one reward function) instead of a unique task [14]. And there are still other studies focusing on enhancing the quality of the demonstrations in order to decrease the number of demonstrations and learn more effectively [15].

Learning from demonstration is commonly applied in robotics, the robot assumes the student role and the human is the expert teacher. Thus the goal is to demonstrate the task to the robot, so it should learn from watching the demonstration, so the robot could use the skill when necessary. Once a robot has acquired certain skill it could also teach other robots.

2.1. Learning

Learning supposes the generalisation of a task. Let's suppose you have to learn how a bird looks like. The first step would probably be to list down the characteristics of a bird. You should have a list describing the wings, eyes, feathers, tail and beak. Without focusing on specifics like the species, geography or colours, we can say that animals that fit the description can be called birds. The list is a guideline for you to be able to distinguish a bird from any other animal or object.

Once we have learned the characteristics that are uniquely assigned to certain things, it is now easy to distinguish objects and animals around us. They can be clustered in simple term such as, in this particular case, a bird. Our brain learns to generalise tasks in order to obtain the

desired results every time the action is performed regardless of the variability of the environment at each try.

Once a robot is able to reproduce certain skills with the least error possible or even better, non-error, it can be said that the robot has learnt a skill correctly. Therefore, it can be said that it has certain intelligence embedded that let it learn.

Robots can be controlled by different methods and techniques that have been used for several years and produce excellent results in certain conditions and applications [16, 17]. But it is important to note that robots are also being used outside of factories in applications and environments that require high adaptability, reliability and constant learning. These may demand robots that are capable of handling uncertainty and variability in fast and dynamic environments.

It can be assured now that learning is a valuable and almost necessary skill for robots, almost since Alan Turing wrote *Computing Machinery and Intelligence,* concluding that "*We can only see a short distance ahead, but we can see plenty there that needs to be done*" [18].

2.2. Hard programming vs learning

A simple but effective definition of learning from demonstration has been presented previously. In this section, a more formal and structured definition is presented. Learning from demonstration is a mapping built by examples between environment states and actions to perform. This is called policy, and it gives the robot the skill of selecting which action to select if found in a certain environment configuration. The policy is built by all the demonstrations the robot obtains.

There are two paths in learning from demonstration. The first one is very simple. It basically let the robot mimic the motion of the instructor; in other words, the computer must learn how the instructor acts, reacts and handle errors in different situations, which is a process called learning the policy.

The second path, and is more important in this work, is where the robot may not be doing exactly the same task as the demonstrator and there is a small number of task demonstrations available. This means the robot should learn under uncertainty and with low loads of data. In these cases, algorithms should extract the most out of the information in order to be able to process all the information and then create correct cognitive responses topic to be studied in the CODA section.

2.2.1. Pendant programming and kinetic teaching

Pendant programming and kinetic teaching are techniques that could be considered the simplest form of learning from demonstration. Both techniques have their advantages and disadvantages, but they are outside the aim of this book. It is of most importance to mention how they work.

Figure 1 shows an example of pendant programming where the human uses a programming pendant in order to move the robot thru desired points, this technique can produce two paths,

http://dx.doi.org/10.5772/63570

a polynomic or linear path, but learning procedure never takes place during this action, the robot will always move on the chosen path, this limits the generalization capacity of the robot. This is why this technique belongs to a hard programming techniques to control a robot even it uses demonstration to hel the user program the robot.

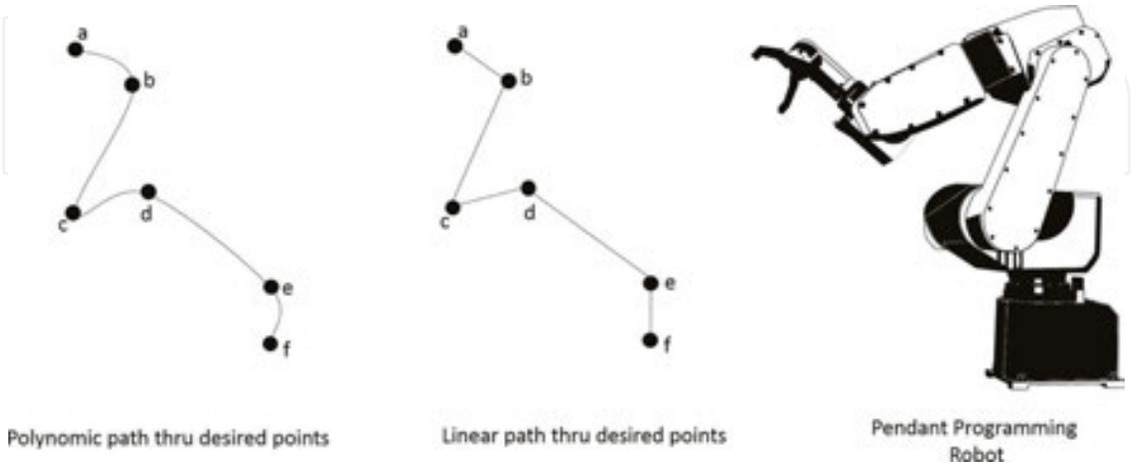

Figure 1. Pendant programming example, same trajectory with two interpolation options.

2.2.2. Learning from data

The previous section exemplified how a robot can be programmed by a demonstration and still lacks intelligence. Intelligence is formed by several different bits such as reasoning and logical deduction but for our purpose, the most basic animal intelligence characteristics that matter are remembering, adapting and generalising.

Remembering, adapting and generalising are part of learning, and these characteristics are the most important to embed in machines. Machines should be able to learn with examples, data and experience. First the machine should obtain an example which contains data, and at each try, the machine should acquire experience. Using the concepts presented earlier, the machine should be able to recognise when was the last time it was in this situation (encountered the same data), tried a particular action (produce this output) and it worked (the output action(s) was/were correct), so it can be tried again. But if it does not work properly, then something different should be tried.

Learning is a skill that gives us the ability to be flexible in our day to day activities. We adapt and adjust our self to new circumstances, recognise similarity between different situations in order to use knowledge in one place or another, and therefore use acquired knowledge in different places and situations.

Learning by demonstration is a powerful tool that has already shown its competence in the robotics field. However, it may be impossible to complete any desired task if it will be the only tool to be used in the entire system. To address this, a hybrid system was built where it will use a non-model based algorithm such as Q-learning or SARSA (explained in the next section)

where the policy will be represented by a Q matrix and adjusted directly according to the reward obtained during each try. These consequences of the adjustment in the policy are measured by the reward function to guide future adjustments in the policy.

The reward function does not actually tell the algorithm if the output is correct or not; instead it tells how much correct it is, and the stored values of the Q matrix should serve as a repertoire of knowledge in order to recognise certain "*encountered data–output action*" pairs for future reference.

3. Reinforcement learning

Taking this extract from [19]:

> *"Of several responses made to the same situation, those which are accompanied or closely followed by satisfaction to the animal will, other things being equal, be more firmly connected with the situation, so that, when it recurs, they will be more likely to recur; those which are accompanied or closely followed by discomfort to the animal will, other things being equal, have their connections with that situation weakened, so that when it recurs, they will be less likely to occur. The greater the satisfaction or discomfort, the greater the strengthening or weakening of the bond".*

As it can be read from the extract, desired behaviours will prevail over the undesired ones. This is due to the reward or satisfaction feeling, and this is precisely part of what CODA is willing to imitate in a computational environment. And since no machine can feel any kind of emotion, the reward will be numerically represented with a function.

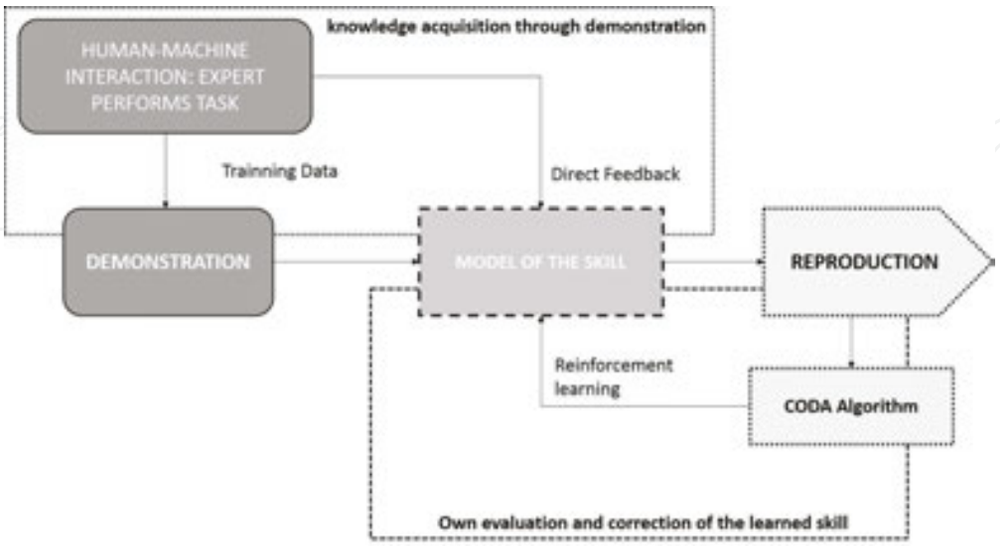

Figure 2. Block diagram showing the cognition model proposed by the authors with the CODA algorithm.

http://dx.doi.org/10.5772/63570

Figure 2 shows a block diagram of the learning by demonstration iterative process taken and modified from [11]. The diagram shows the process of learning a task with the proposed approach and the CODA algorithm.

The interaction between the entities is initiated by the human–machine interaction where the human expert performs the task or gives an example of the task in order for the system to record the information as a training example. In this manner, it gives direct feedback on how the skill is being modelled. The CODA algorithm will have its own evaluation and correction process in order to acquire the knowledge and reproduce the task.

During this process, a reinforcement learning algorithm is the most important tool. SARSA, which is a modification of Q-learning, was chosen. The main difference between the two algorithms is that Q-learning always attempts to search for the optimal path, which will always coincide in that it is the shortest, but this does not necessarily mean it is the best one on each task. In contrast the SARSA algorithm will not always take the shortest path but the safest, because it will try to avoid large negative rewards taking in consideration those "dangerous" situations.

Mathematically, Q-learning and SARSA differ in their expressions as the following two equations state:

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \mu[r_{t+1} + \gamma_{a_{t+1}} max\, Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)], \tag{1}$$

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \mu[r_{t+1} + \gamma\, Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)], \tag{2}$$

Eq. (1) shows the Q-learning algorithm equation for the actualisation of the Q matrix and Eq. (2) shows that of the SARSA algorithm. It can be inferred that mathematically the main difference between them is that Q-learning is off-policy since it takes the maximum value of $Q(s',a')$ instead of computing a'. SARSA on the other hand makes this possible by the simple modification made where it takes care not only of the actual state and action but also the next possible state, action and reward, making SARSA an on-policy algorithm, something explained previously.

Since the inputs and outputs of the reinforcement learning algorithms are discretised if the size of the action space and state space can be reduced, it is always a good idea since the reinforcement learner algorithm is basically a search method. Therefore it is good to search for certain techniques that would let us handle these spaces in a proper manner, and it is upon the application and the expertise of the reader how this could be approached.

Figure 3 describes how the algorithms search for the goal state given a state space, action space and a reward function. It also explains how the learning agent, in our case the robotic hand, performs action in the state s_t and receives a reward $r_{(t+1)}$ from the environment, after this the agent will move to state $s_{(t+1)}$ because of the reward given. Basically the sensors on the hand define the state, because they are the representation of the environment around the robot, telling us if the hand is touching any surface or not. The actions are how the robot moves its

actuators. The reward could be the similarity between the learnt (desired) data and the produced data on the present state, and could be bigger if similarity is higher and reward should be lower if similarity is not likely.

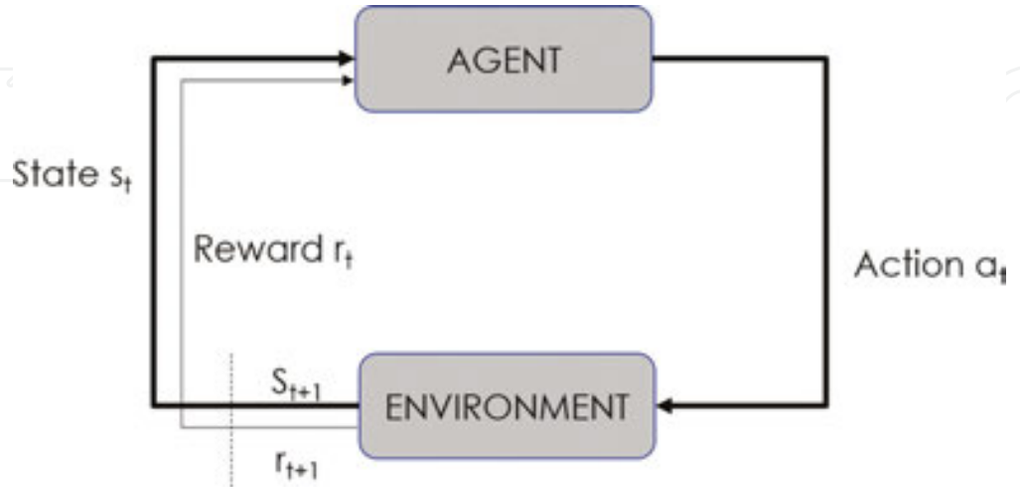

Figure 3. Reinforcement learning cycle, agent performs action, receives reward and ends up on a new state.

4. Artificial immune systems

In the human body, the immune system is the collection of cells, tissues and molecules that defend the body against infections [20]. Its basic function is to eradicate and prevent infections.

Naturally in our body there are two types of immunity: innate immunity and adaptive immunity, both share the same objective, but have different tasks and time reactions. Innate immunity acts in the first hours whereas adaptive immunity operates through days.

Adaptive immunity can further be classified as humeral immunity and cell-mediated immunity, containing different responding, effectors cells and functions.

A detailed biologically inclined overview of the Artificial Immune System can be found in [20]; while a computationally inclined overview can be found at [21].

Simulating active immunity is of great interest for the CODA algorithm. The most common example is vaccination of an individual, in this case the "naive individual" is exposed to antigens in order to mount an active response and be able to eradicate the infection. After this process the individual will be immune to that microbe and that is because it has already built resistance for a later infection. In contrast, passive immunity is shown in new-borns that do not have an immune system mature enough to fight against pathogens, but are protected by their mother's antibodies through the placenta and milk.

http://dx.doi.org/10.5772/63570

Artificial Immune Systems where developed with active immunity in mind since at the very first they were implemented for intrusion detection in computer servers. The main goal of the system was to detect insiders and external intruders that were committing abuse or misuse of the computer systems.

There are several crucial characteristics that are important for the CODA algorithm to inherit the robustness, adaptability and specificity among other characteristics that are desirable for a real problem-solving algorithm. The characteristics that are considered important and should be applied in the computational version of the immune system are shown in **Table 2**.

Feature	Functional significance
Specificity	Ensures that distinct antigens elicit specific responses
Diversity	Enables immune system to respond to a large variety of antigens
Memory	Leads to enhanced responses to repeated exposures to the same antigens
Clonal expansion	Increases number of antigen-specific lymphocytes to keep pace with microbes
Specialisation	Generates responses that are optimal for defence against different types of microbes
Contraction and homeostasis	Allows immune system to respond to newly encountered antigens
No reactivity to self	Prevents injury to the host during responses to foreign antigens

Table 2. Natural Immune System characteristics that make it such a potent tool in the human body. This features are imitated by the computer counterparts of the NIS the AIS.

Specificity and diversity: This is an important feature that lets the immune system distinguish from similar antigens, helping us specify the response for a certain antigen and no response to any other even if they are quite similar. Therefore lymphocyte repertoire could be over millions of different units, all of them cloned, built specifically for different antigens, leading to a much extended antigen distinguishing.

Memory: When the system is exposed to an antigen for the first time, the response is called primary immune response and is mediated by lymphocytes, called naive lymphocytes. The second encounter is called secondary immune response, and it is supposed to be rapid, larger and more effective.

In order to implement and use an Artificial Immune System, it is quite important to understand deeply how the Artificial Immune systems works for the several processes that occur within the system and manage several tasks. In [22], a very explicit description of the AIS based upon the natural model from the human body of the Natural Immune System (NIS) can be found. According to the explanation of the model, the immune system is an example of a mechanism that is capable of learning and remembering. This memory is capable not only of string previous interactions but also forgetting information with little use.

The NIS is an example of a yet adaptive, decentralised and effective system. The B and T cells are just examples of how the NIS has a working structure that delegates all tasks. All the previous characteristics are desirable for a problem-solving algorithm that can offer novel

methods and thus testing is required to compare this paradigm across all the machine learning methods.

In [22] the authors test the algorithm with a simple pattern recognition problem as a first test. In order to have a better overview of the performance of the AIS, it was applied to a real-world problem such as the recognition of the promoters in DNA sequences. According to the results presented, all were consistent with other approaches such as ANN and Quinlan ID3 obtaining similar results. The performance was better than Nearest Neighbour Algorithm.

The diversification of the AIS is an important characteristic since it does not focus on a global optima, instead the antibodies evolve in order to handle all the variety that the antigens can represent. **Figure 6** shows how the antigen is taken and mutated to produce all the antibodies represented by the blue circles. It is also quickly adaptable to changing situations, naturally the system can handle event-response situations. This is one of the most important features of the NIS as well, and the response must be active in a matter of minutes or hours in order to protect the human body.

According to [22] a remarkable characteristic of the AIS is the genetic mechanism that can mutate and reproduce the antibodies, memory and its self-organising properties. In CODA algorithm these concepts are implemented [22], for example, the self-organising characteristic implemented with a clustering step running over the antibodies in order to organise all data.

It is important to notice that the computational model presented in this document or in any of the articles that were revised does not aim to model the human NIS perfectly, nor it is an attempt to provide explanations of how the system works within the body. Rather, the stated features are emulated in order to solve problems that need such characteristics presented in **Table 1** and **Table 2**, such as specificity, diversity and memory, among others.

Farmer et al. presented Eq. (3) as a mathematical model of the immune system, in which N represents the antibodies, n represents the antigens, c is a rate constant that depends on the number of comparisons how antibodies are being stimulated, a represents the current B cell, xe_j represents the j_{th} B cell's epitope, xp_j represents the j_{th} B cell object paratope and y represents the current antigen. Finally the first term in the sum is the affinity between antibodies and neighbours; the second term is the enmity between the previous named objects; the third represents how well the antibodies are capable of binding with the antigen; and the last term models the tendency of cells to die if no interactions are present.

$$Stimulation = c\left[\sum_{j=1}^{N} m(a, xe_j) - k_1 \sum_{j=1}^{N} m(a, xp_j) + k_2 \sum_{j=1}^{N} m(a, y)\right] - k_3, \tag{3}$$

The model used by Farmer et al. presented three types of mutation presented as follows.

Multi-point mutation: Each element in the antibody is processed in turn. If a randomly generated number is above the mutation threshold, then the element is mutated.

http://dx.doi.org/10.5772/63570

Substring regeneration: Two points are selected at random in the antibody's paratope. Then all the elements between these two points are replaced by randomly generated elements, resulting in a partial regeneration of the antibody.

Simple substitution: Operator uses the roulette wheel [23] algorithm to select another B cell object from which elements will be substituted into the current B cell object.

The mutation procedure is an important procedure since it has been designed to introduce diversity into the system, as antibodies are created, the previous examples are just a few techniques that can be used during the mutation process, but there are several ways in which the mutation can be implemented.

The AIS systems is compared with other several machine learning approaches such as ANN, LCS, CBR and others showing that AIS offers certain advantages over some of them specifically the self-organising features and the unsupervised nature of the algorithm [22]. It also let us know that it can be noise tolerant and that the algorithms inherently generalises such a problem with ANN that can over fit. Also the self-organising feature makes it easier to handle rather than ANN that can be tedious and time consuming in order to tune them for a certain application checking the bias and variance of a certain configuration of the ANN.

The AIS is a paradigm that yet seems to have certain characteristics that attract engineers and scientist as well. It is a powerful example of a learning system that not only adapts rapidly but it also is a non-linear network, has a content addressable memory and it is self-organised. With this in mind, it is important to stop and analyse about the contributions that the paradigm has brought to different areas where it has been applied. CODA wants to use these features and apply them to the grasping problem in robotics, this is the main reason why the algorithm uses previous released machine learning tools plus it emulates the AIS in order to solve real problems and use AIS in important applications.

Precisely the study [24] explores how the AIS has been applied in "out of the lab" applications, where real world problems state new challenges, this question of how have the AIS has perform in the real world applications or industry leads us to the importance of focusing the efforts on developing theory that will serve as hard evidence of this paradigm.

Since according to the no-free lunch theorem [6], there is no one "do-it-all algorithm" that can outperform all the machine learning methods available so, it is important for any paradigm to contain features that may not be present in any of the previous techniques. In this manner the new algorithm can be described as a truly novel method.

Authors of [22] disagree with the definition proposed in [25] in which effectiveness is measured in terms of how an algorithm performs better or not, compared to other in benchmark test. For example they measured the time it took to complete the task. The same work exposes certain problems where AIS has done a magnificent job, these applications are presented in **Table 3**.

Major	Minor
Clustering/classification	Bio-informatics
Anomaly detection	Image processing
Computer security	Control
Numeric function optimisation	Robotics
Combinatoric optimisation	Virus detection
Learning	Web mining

Table 3. Major and minor areas where AIS have been used.

Table 3 reflects how the articles can be grouped according to the main areas where AIS has been applied. The studies in [3, 26–28] talk about Anomaly Detection, [29–31] expose works on optimisation, image processing [5], robot control [32–35] and web mining [36]. It is important to note that this table does not represent all publications related to AIS. It is a general picture based on information contained in ICARIS 2004 [37], ICARIS 2005 [38] and in the bibliography produced by de Castro [39].

Something important that must be noticed about most of the applications in several papers is that the AIS is tried in benchmark problems and robotic applications tend to be simulated rather than in real environments and almost all of them are small and simplified.

The article [22] concludes that there is a necessity to construct a robust framework that will allow the AIS to have a more biologically grounded theory, but it also remarks the need of multidisciplinary work between computer scientist, engineers, biologists and mathematicians.

Another important aspect discussed is that AIS algorithms are not so generic and most of them are quite specific to the task on duty. More generalised algorithms would be develop if the theoretical aspects of the AIS are improved in such a manner that generalisation would be easier. The authors close their work [22] with a phrase that is important for the development of the AIS algorithms:

> *"However, all this futuristic discussion is interesting, but what is needed is well-grounded immune inspired techniques that are applied in a logical and coherent matter"*.

CODA algorithm pretends to handle data that could contain noise or that may be a small amount. Therefore the AIS should be an algorithm that can handle data in those circumstances, from the several bibliography revised it was found that in [40], the use of the AIS helps us in visualising how the AIS can be an unsupervised machine learning method but also how it can handle an analysis on the data and help visualise information.

The article describes the procedures that are internal to the AIS such as the primary and secondary response, how the antibody/antigen binding occurs, the B cell stimulation, the immune network among others that can serve as a really good introductory text for the reader. But it is not until the description of the B cell cloning were interest in such concept was really awaken.

http://dx.doi.org/10.5772/63570

The text is very explicit on how the AIS clones and mutates the B cells to build up the memory mechanism where all the information will be stored and will help on the recognition tasks where similar patterns must be identified. With all this data it is necessary for the memory to have an organising mechanism to form clusters that later will identify patterns.

As it is stated before in this document, there is a need for the AIS to be implemented in a simple yet logical manner, and the authors of [40] build up a really simple method to measure the affinity between the B cells with the Euclidean distance shown in Eq. (4).

$$Affinity(a,b) = \sqrt{\sum_{n=1}^{ND}(a(n) - b(n))^2}, \tag{4}$$

This implementation was tested with the "Fisher iris data set", which consist of 150 instances from three different classes, specifically plants. The attributes taken in consideration for the three classes are: sepal length, sepal width, petal length and petal width. The dataset cannot be separated linearly in two groups (Iris Virginica and Iris Versicolor) but it is possible to segregate the third group linearly corresponding to the Iris Setosa class. First principal component analysis was used to represent the data in a lower dimensional space in order to make possible a two dimensional graph. The AIS was able to recognise three classes independently from the distribution of the data set that may facilitate to classes and have more problems in the recognition of the third class.

The created network was capable of receiving unseen data and still able to recognise between classes, concluding that the networks were effective as a simple classification, one of the main objectives for using this AIS. The second valuable characteristic is that they seem to generalise in a wider region of the input space making it quite interesting since this feature could be from great importance in environments where little data can be collected, where the data could be corrupted with noise or demonstrations of the task are limited.

It has been seen all over the bibliography that a common point is the development of a more reliable theoretical point of view for the AIS, this was one of the first statements when approaching the AIS. The main reason the ABBAS book of Basic Immunology was revised was most of the documents found are a general overview of the NIS and difficult to understand if you have not had any previous experience. One interesting work is [40] and "Where" should be "where". Where the clonal selection mechanism was treated in a really interesting and straight forward manner. The clonal selection theory (CST) [41] is basically used to explain how the NIS response to any antigen stimulus. The main idea of the CST is that those cloned cells are in charge of recognising antigens.

In [20], a recognition region is presented but it is defined as spherical, a characteristic that could limit how the recognition ball could work. In [40] the author does not limit this region with a spherical geometry, leading to advancement in the theoretical framework for the AIS.

In [40] the B cell Algorithm (BCA) is defined as an iterative process that improves candidate solutions in a specific problem, with the use of tools such as cloning, mutation and selection.

Again the Euclidean distance is used as a measure for affinity, which suggests it could be a standard for this process. But how the BCA can be differentiated from one of the most popular evolutionary algorithm such as genetic algorithms? The reader should notice that no cross over is employed in the cloning process, there is no necessity for this method to be applied in the BCA in order to increase diversity.

Specifically talking about AIS algorithms and more precisely about clonal selection ones, the authors consider it really important to make the population variation according to probabilistic rules and follow the nature of the model so this probabilities for the transitions to a new state depend only on the current state so the Markov chain could be satisfied. This also means that the algorithm should converge and find at least one global optimum solution with probability equal to one as $t \to \infty$. One of the very first papers that introduced this was [42] and should be revised for a detailed lecture on the theme.

5. CODA

Cognition from Data is what names the proposed algorithm, as its name can tell, the objective of the algorithm is to aim for cognition in a computational manner – analysing data, extracting more information if possible and acting in an environment to achieve a task with the most accurate action even with few data as training examples. This is done by the NIS when a "naive individual" for the first time recognises an antigen. In order to comply to these characteristics and objectives, an algorithm should be flexible yet specific, should learn from previous experiences and store the knowledge in an organised manner. In doing so, it would respond faster the next time the situation occurs, just as the NIS in the secondary immune response responds to a previous known antigen.

In a search for a system that could have these characteristics, the NIS is one of the most impressive natural systems that not only adapts itself to new and unknown situations. It also learns from every experience and creates its own knowledge by an awesome mechanism where antibodies are created and stored in order to create a fastest response if a second interaction with the antigen occurs. **Table 1** in the previous section talked about certain desirable characteristics that CODA emulates from its biological counterpart – the NIS. This mechanism lets the NIS keep safe our body and it has been doing a great job for millions of years. Therefore CODA should provide a great solution to the grasping problem shown in Section 6, using AIS exceptional features.

The main reason why the NIS was studied in [20] is to have a more detailed panorama from this system that was taken as spinal vertebrae to design an algorithm. The non-free lunch theorem presented in [6] was also taken into consideration since it mentions that there is not one tool that could do all the possible tasks available in the problem space. It is important to create machine learning pipelines that could handle a complex problem and solve it with different tools not just one that would try to be the "do it all".

http://dx.doi.org/10.5772/63570

Figure 4. Block diagram showing the cognition model proposed by the authors with the CODA algorithm.

Algorithm 1 in **Figure 9** (Appendix) shows the pseudocode for the CODA algorithm, where it can be seen it has three elements that were presented previously on this document, the Reinforcement learner and the AIS. The learning from demonstration is an external part of the algorithm that acquires the training information and then presents this data to the CODA algorithm. **Figure 4** shows a flow chart that presents in a simpler way the pseudo code steps that may take place and adds the part of the algorithm that takes place outside of the learning system which is the demonstration of the skill.

The sequence of steps shown in **Figure 2** are of great importance since any change in the order will modify how the entity will learn and measure its performance according to the desired state. Also it is important to notice that at the beginning, there is a human–machine interaction that would recover the data from the example, and that it is necessary to be careful on which platform is chosen for this interaction in order to be positive and recover the most of the data possible without causing any discomfort in humans or any stress. The most positive the interaction the easier will be for the humans to share data and for the algorithm to have more and better examples of the task to be done.

In our case a data glove is the hardware chosen in order to acquire data (antigen). The glove is a well-known tool and almost any human has wear one, in this manner the acceptance will be faster, easier and will lead to a better interaction. This is really important because as mentioned before, there will be scenarios where data could be hard to obtain and may be corrupted with noise without mentioning that it may also be limited.

Figure 5. Probability densities from 11 neurons showing their preferred firing rates to certain stimulus, notice the true value stimulus is at $s = 0$.

The flow chart is specific for the presented application but it lets the reader follow the algorithm forward on its run. On the first stage, the knowledge should be given in certain manner. It could be demonstrated (in our case with the data glove) or it could be set with some previous recorded data. CODA will obtain this antigen from the data glove and produce clones from this single example. A simulation result can be found in **Figure 7**, where red squares are the training example and the blue circles are the cloned antibodies produced by CODA's clonal/mutation procedure. Then the entity should measure its state in order to define this information

http://dx.doi.org/10.5772/63570

as the initial state, of course a task to be performed by the sensors embedded in the hand as explained in **Figure 8** in Section 6. After the cloning procedure has created the antibodies, the affinity should be measured as explained in Eq. (4). If the affinity criteria are not met the antibodies will be discarded and deleted.

The clonal procedure was inspired from the response of certain neurons given a set of stimulus. The outputs and the probability densities are shown in **Figure 5**. The graph represents the conditional probability presented by certain neurons producing certain firing rates r given a specific stimulus s [43]. This information is presented in [44] where several experiments are presented using the cercal system of the cricket in order to study neural decoding.

Using a clonal/mutation procedure the algorithm was able to obtain completely different probability distributions maintaining a normal density, this can be shown in **Figure 6** where the original values are compared with the mutated values after the clonal/mutation procedure.

It can be seen that we obtained different values that give much more data where the algorithm can explore and evaluate with the next stages of CODA Algorithm in order to use them if they are suitable for solving the problem.

Figure 6. Probability densities comparison from original data and cloned/mutated data using CODA clonal/mutation procedure.

In fact, the clonal/mutation procedure will be given limited data, and it must produce new data samples even if it is given a simple example. **Figure 7** shows precisely that the clonal/mutation procedure is capable of producing completely new data from a simple example and

Figure 6 shows how this new data has a completely different probability density, providing new opportunities for unseen data. The examples were obtained from the data glove as a demonstration example from the first stage of the block diagram shown in **Figure 8**.

The algorithm was given a five-line two-column array that contains force and angle values for each of the five fingers. With this example the algorithm was able to produce completely new data that before its use should be evaluated (Eq. (4)) in order to be used.

Figure 7. Red squares are antigens that were cloned and mutated producing the blue circles (antibodies) with the CO-DA clonal/mutation procedure. Mutated under a normal distribution set by the user.

Once clonal and mutation procedure is completed the algorithm will take the antibodies information and evaluate their affinity to the original value and with the reward function, this will produce an expected reward before actually using this configuration in the hardware. If values of affinity and reward are both below a threshold, the antibodies are considered to be maintained in the memory.

After the previous procedure is completed, the reinforcement learning algorithm uses the antibodies and defines the most rewarded as the goal state in order to complete the task. Finally from the *M* taken antibodies the algorithm will lead the system to the goal state. It is important to store the antibody-reward pair that corresponds to the one that obtain the highest value by the reward function doing the action, since this was the one that complete the task as desired. Finally a matrix should be constructed in order to store the Q matrix and the antibody-reward pair, for future reference in the memory. The last step will run the clustering in order to segregate the different classes that can be found and traduced as different tasks for the system.

http://dx.doi.org/10.5772/63570

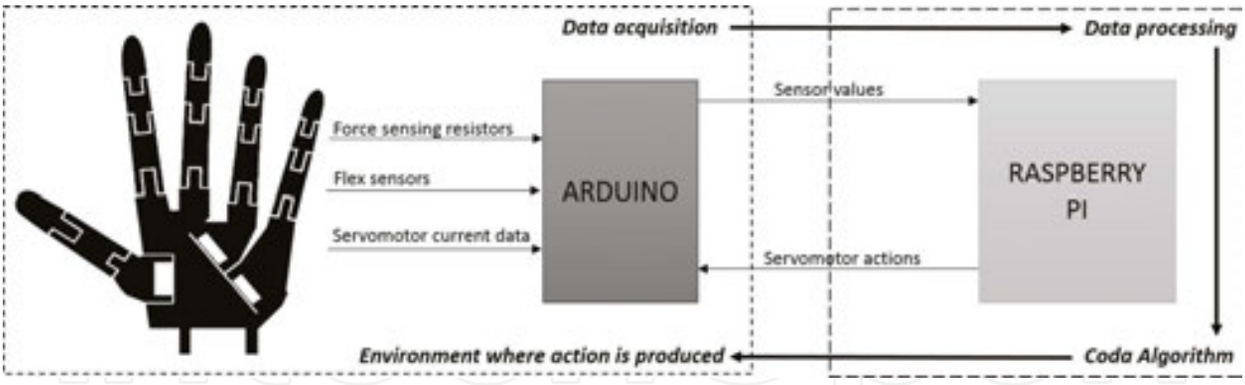

Figure 8. Hand system with its elements, where arrows show how data are managed between hardware components.

6. Application

The CODA algorithm in our laboratory will be implemented in a humanoid hand which is part of the Inmoov open source project. The application is one of the most difficult tasks in robotics – grasping objects. The system components are as follows and it is shown in the picture below.

The humanoid hand, all 3D printed in the laboratory consist of six standard servomotors, flex sensors for position information and force-sensing resistors for obtaining information about grasping forces.

The main brain for the entire system is a Raspberry pi where the algorithm will be saved and implemented. The Raspberry pi is connected with an Arduino which acquires the data from the sensors and the servomotors in order to send it to the Raspberry pi for processing.

Figure 3 will help the reader understand how the algorithm will be used in the application. The main idea is to produce an affordable grasping for an object. To exploit the advantages of CODA, the training examples will be limited, and CODA should produce more data as the one showed in **Figure 6**.

The first stage will demonstrate how to grasp the object (training example or antigen), where the human will use the data glove in order to grasp the object and record this training data. Once the training data is obtained, CODA will then define it as an antigen, since it is emulating the NIS, the information from the antigen will be taken for the cloning procedure.

The clonal procedure, affinity measuring, reward measuring and reinforcement learning will all take place in the Raspberry pi, which is the dedicated hardware for processing all data and store the knowledge or repertoire of antibodies.

The previous mentioned steps shown in **Figure 3** and in Algorithm 1 will then produce an affordable grasp that can be used by the hand in order to grasp the same object grasped by the human hand previously.

In this manner CODA will let the hand grasp objects with minimum data examples and with no knowledge of the physical characteristics of the human or robot hand. Instead it will use a reward function specifically designed for the application that will give numerical incentives in order to reach certain finger angles and force values.

Angle	Force	Mutated angle	Mutated force	Affinity
80	500	82.6499	498.8566	2.886058483
100	534	100.1784	534.696	0.7185500216
73	480	74.0844	481.2094	1.624368099
60	767	60.427	767.4406	0.613561211
100	680	100.8786	680.0866	0.882857588

Table 4. Single example given to the CODA algorithm, showing a simple mutated pair of angle and force values for each finger.

Table 4 shows a simple example of the original pair obtained from the data glove including angle and force. It can be seen how the clonal/mutation procedure in CODA modifies the values producing new data. Within the same table in columns three and four contain a simple set of cloned values that are also shown in **Figure 7**. And finally, the last column exposes the measured affinity computed with Eq. (4). Now the reward should be measured before the antibodies can be stored and used by the SARSA algorithm as goal states.

The Discussion section talked about the reward function that needs to be designed taking in consideration several factors in the system in order to make it possible to measure how suitable are the final antibodies for reproducing the task.

From this point on, the algorithm is straightforward. SARSA is used in order to reproduce the grasping posture which was set as a goal by the antibodies that meet the criteria. And finally, the Q-Matrix will be stored with all the related values such as antigen, antibodies, reward and affinity.

This application was selected because it happens to have low amounts of examples available about the task and a really difficult task that most of the time needs the use of kinematics and dynamics in order to produce grasping postures. Thus grasping is a great application area, since it is desired to create new knowledge about the grasping postures and remember those executed grasping in order to have faster responses for further interactions with the same object just as the AIS does with the secondary response with an antigen that has already been in the body.

The more objects the hand grasp, the more knowledge it will acquire. To simulate the self-organised memory, a clustering algorithm should run through the knowledge database and cluster similar postures, suggesting which of those postures correspond to similar shaped objects but with different dimensions.

http://dx.doi.org/10.5772/63570

Figure 9. Grasping a cube with data-glove, simple example of grasping posture obtained is on the graph below.

To demonstrate the functionality of the algorithm, a simple test was done. From a set of geometric figures, one was chosen for the test. The test protocol is simple: first the object is grasped by any user wearing the the data-glove system, once the sensor reading remain stable during grasping the data is saved to MATLAB workspace in order to be handle by CODA as antigen. As explained during previous chapters the algorithm will generate diversity from this given simple example (limited data) and produce certain amount of antigens for later evaluation by the emulated clonal selection and negative selection procedures embedded in the algorithm using the reward and affinity functions. For this specific test the cube figure was selected, and a simple example was recorded. **Figure 9** shows the cube grasped by the human teacher wearing the data-glove and the data obtained.

Finally the obtained information shown in **Figure 9** is feed into CODA. From this, a set of 500 antibodies was created that were evaluated by the reinforcement learning reward function and filtered in order to leave the most rewarded antibodies in the repertoire for the hand to grasp the object. **Figure 10** (left) shows the total antibodies created after the learning from demonstration procedure and on the right, the remaining repertoire data result of the reward and affinity evaluation procedures. This repertoire is considered the most capable data to produce affordable grasping postures, therefore, all of them were delivered to the hand system to confirm the functionality. Resulting to 100% of positive results means all the repertoire pairs were able to grasp the object.

With this simple test, the hand, data-glove and CODA algorithm have proven to function correctly and to produce useful data from limited samples (one simple example). They were able to produce new information that after being filtered and evaluated could reproduce the task without any parametric information regarding the hand size or link dimensions. The algorithm prove to be able to grasp the cube 100% of the time but when tested with an hexagonal prism it worked with 83% of the cases only, but still is a good result. **Figure 11** shows the hand grasping the cube and the hexagonal prism during the tests under the same protocol.

Figure 10. CODA algorithm cube test data obtained. Left, green antibodies created on red the unique example provided. Right, red the unique example, in green the repertoire of the most rewarded antibodies thus the most capable of producing grasping postures.

Figure 11. Hand correctly grasping objects from the data obtained from the CODA algorithm.

http://dx.doi.org/10.5772/63570

7. Discussion

CODA algorithm has been described and all the theories that involve this new design have been presented. Also the pseudocode and the flowchart have been exposed in order to explain how this algorithm will first learn from a demonstration, clone and mutate this data in order to create new knowledge (data never ever introduced to the algorithm before, created by the clonal/mutation process over the input data), evaluate its possible success within the system and finally use this new information as a goal.

The algorithm uses NIS as a basis in order to emulate several tools. The NIS is one of the most important and reliable systems within the human body. The developed algorithm uses a machine learning pipeline in order to create an algorithm, since certainly there is no one "do-it-all" method in machine learning, but several tools that need to be used together in order to exploit the different characteristics of each method.

CODA algorithm needs to be evaluated first in a simple environment with a simple task such as a 2R robot, and then with the main applicationexplained in the previous section. The hand grasping test would let usoptimise the algorithm and consider which methods fit better for measuring, affinity, stimulation and if the cloning and mutation processes are suitable for the tasks.

Affinity and stimulation techniques presented would be used in order to evaluate if for the presented algorithm they are suitable or not. If not, they should be modified or even design new ones that would let the algorithm work with much more personalised functions, increasing the algorithm's reliability and positive results.

The reward function is under experimentation. It involves two aspects: the forces and the angles. The algorithm should be given a reward as soon as the hand starts closing that means when angles are bigger and the force starts to increase in the finger tips. But if the hand just closes, there is the possibility that the grasp will not be affordable since the object could be damaged. Therefore the function should give high reward when the force starts increasing but reduce the reward if the force is beyond a certain threshold. Something similar happens with the angles since they may not be bigger and completely close the hand. This procedure needs to be done experimentally in order to obtain values from the sensors and set the thresholds for the angles and forces.

The three methods that conform CODA algorithm were chosen to be part of CODA in order to increase their efficacy at solving problems rather than using them individually, in order to exploit their advantages. They complement each other and are used for a specific objective for the proposed learning process, each one specialises on the one task it can perform better.

The very first contact with information is provided by demonstrations, which is a crucial aspect for the algorithm. Providing these examples simplifies the learning of the task, if no examples were given, an extremely well-designed algorithm was to be designed and programmed in the agent in order for it to reproduce the desired task, and the algorithm would need to be really descriptive providing every possible detail in order to obtain correct results. This

procedure would require specially trained personnel and it would be time consuming. Compared to the demonstration process explained in this document that requires the sensors on the robot or in the person and almost anyone would be able to teach numerous skills just by reproducing them, without specialised studies or any coding knowledge.

In the previous paragraph it was said that if programming with a complete, well-designed and correct algorithm, an entity (in this case a robot) could reproduce a task. This is completely true if conditions and the environment remain constant. The truth is that for dynamic environments and uncertainty situations this would not be the case. Today's robots are being more and more embedded in our society. In these dynamic environments, sensor noise is more frequent. For these reasons a robot programmed as mentioned before would not be able to commit to its objective easily and would probably fail. This is due to a lack of learning capabilities, which means, robot would not be able to learn from past mistakes and acquire new abilities or perform the same task at different scenarios. This is where reinforcement learning used by CODA becomes an important method.

As explained in previous sections, reinforcement learning would produce a change in an entity's behaviour in order to obtain the highest reward, therefore since the reward function is designed upon the task's most crucial and important aspects, the algorithm will always search for the states that will produce the highest reward, if not, actions would be needed in order to achieve the task. These actions have the advantage that they can be low-level (i.e., voltages applied to motors) or high-level decisions as complex concepts or complex decisions (i.e. don't move).

In the CODA Algorithm the reinforcement learning helps the Artificial Immune System (AIS) to produce the best antibodies possible given a task-specific reward function, meaning the new antibodies would produce higher rewards. If bad antibodies are being produced, actions would alter the clonal/mutation procedure in order to modify the diversity generation.

Finally, the third method is the AIS which produces new data from single example – from the pathogen's antigen reducing the necessity of big data set, that may result expensive, difficult to obtain and construct thus time consuming. It also handles all the information and evaluates the produced data in order to find and produce the best possible antibodies. Finally it constructs a self-repertoire of state-action arrays in order to produce a faster response the next time the agent is encountered in the same state.

As an example of the functionality of CODA the system was tested, the test methodology is as follows: First wearing the data-glove a person grasped one of the geometric shapes in **Figure 12**. For this specific example the grasp was done with the cube. Once grasped, the sensor values were stored.

With these stored values, CODA produces 100 new data samples (antibodies). They are evaluated in order to select the most rewarded. Once filtered, the data set was reduced to maintain the most capable antibodies to reproduce the grasping task, then these antibodies were used to reproduce the task.

http://dx.doi.org/10.5772/63570

Figure 12. Test result from grasping cube, top left figures show the data-glove system used to acquire grasping data. Top right graph shows the single example acquired and provided to CODA. Bottom left show the example and the antibodies produced by CODA, finally bottom right show a close up of the thumb antibodies and sample distribution.

Figure 13. Geometric figures for testing the system, and the cube used for this very first test.

It is important to notice that the only example provided was the one obtained with the data-glove. No information was given to the algorithm about the hand size or any parametric information. It only searched for the best possibility in the antibody population based on the reward function (**Figure 13**).

The CODA algorithm has proved to be able to use one simple example and produce new and useful data from this unique information and from there produce a grasping posture. The requirements are less than some other grasping algorithms that can correctly produce grasping postures as well but need much more information such as 3D objects [52], orientation of the hand and tracking with tags [45], tons of pre-processing hand pose (examples) needed for the algorithm to learn with expensive computational procedures for reducing dimensions [46, 47] or partial object geometry information [48]. These approaches are way more expensive not only computationally but also in the hardware required and rely on the use of big databases and more requirements than CODA does.

8. Conclusion

CODA Algorithm is a Machine learning approach that can solve the grasping problem in robotic systems. In fact, the main algorithm is proposed as general solution when the problem can be expressed in Markov Chains. The algorithm is designed with the Computational model of the Natural Immune System, which is an Artificial Immune System.

The Artificial Immune System is a really broad system that has several task and subsystems as it is decentralised; therefore, numerous algorithms have been modelled in order to solve problems such as classification, fault detection among others. In particular, the AIS memory, the self-organised model, the genetic procedure in which antibodies are created and selected to complete the tasks, are all features that will lead to a robust yet versatile algorithm.

The algorithm presented in the previous sections was designed in order to produce new knowledge from limited data samples in order to reduce the need of big databases of examples of an specific task, that are difficult to build and most of the time have to be paid. In the bibliography it can be often see how there is a trend to reduce the training data sets [53], or studies to select correct data sets that would be representative enough [49], reduce the use of examples in order to learn [50] or reduce the population in order to optimise the algorithm [51]. From the previous articles it can be seen that the advantage of CODA over some mentioned algorithms is that CODA can use a simple example provided by demonstration and use this information to produce more knowledge. For example in [53] training data sizes range from 47 to 17861 samples. This would mean to ask for 17861 samples of grasping postures to people which is not impossible of course but it would be expensive, time consuming and tedious. In contrast CODA will need one demonstration of the task and it would do the hard job of producing new knowledge evaluate it and then use the most capable antibodies in order to reproduce the task.

http://dx.doi.org/10.5772/63570

According to **Table 1** CODA has several advantages. The first one being the quantity of examples provided in order to learn. Some algorithms need a big number of examples in order to learn a task, just as presented in the previous paragraph. CODA lets the algorithm produce new example possibilities distributed in a Gaussian space from a single example, user configurable. This is achieved thanks to the clonal/mutation procedure of the algorithm obtained from the Natural Immune systems. This can be seen in **Figure 7** were CODA was given a single example and proved to produce new data in seconds. Additionally **Figure 13**; shows how the algorithm produced 100 new data samples from a single demonstration of grasping a sphere.

The clonal/mutation procedure produced *N* different data points that can be used in order to solve the task. They have to be evaluated in order to corroborate the affinity to the original example. It is important to notice the importance of the evaluation in the algorithm, since the clonal/mutation can produce as much data as the user configures the process. Therefore, the evaluation process filters the data that may be harmful for the system or non-useful, reducing risks. The algorithm is capable of reducing learning time compared to algorithms that have to adjust weights or parameters in order to learn from a database. It is well known that those algorithms require certain minimum amount of data to effectively model the task, and it is also known that the more data samples, the more learning of the task. With CODA and the way it was designed, the task can be modelled from the very beginning since the learning is acquired by a demonstration of an expert. The clonal/mutation procedure reduces the amount of learning examples needed to be provided to the algorithm. The affinity function evaluates if the data obtained is coherent with the example provided in order to eliminate data that was created by the clonal/mutation procedure that can be harmful or that has no sense.

The next procedures complete the task with a reinforcement learning algorithm that assume as a goal state the data produced by the previous steps. This advantage of CODA creates a new generation of algorithms that should be capable of learning but reduce the necessity of data and training time, in order to model a task. Another advantage of these algorithms should be to produce own knowledge and the capability of evaluating their own, in order to produce reliable data.

CODA is an algorithm that pretends to complete the gap between data and cognition developing trust worthy responses, creating its own knowledge, measure its reliability, memorise and organise the data in order to re-use it when needed based on the immune system. These are procedures that CODA was able to emulate numerically.

Appendices

Appendix

See **Figure 14**.

Algorithm 1 The CODA algorithm

```
INITIALISATION
Set μ ← a value between 0 and 1
Set γ ← a value between 0 and 1
Set ϵ ← a value between 0 and 1
for all classes do
  if antigen belongs to any stored repertoire then
    Obtain repertoire class
    Obtain information for repertoire
    Extract Q-Matrix
    Do action
  else
    Create new class
    Store antigen
  end if
  Clone N antibodies from the antigen information
  if antibody affinity is greater than parameter then
    Keep antibody
  else
    Delete Antibody
  end if
  for all kept antibodies do
    Measure reward
  end for
  Obtain Maximmum Reward-Antibody pair
  Set Q(s,a) ← small random values
  while training scenarios < m do
    s ← random state of all possible states
    a ← according to ϵ-greedy policy
    while s =! antibody do
      Obtain r ← according to s and a
      Replace s' ← a
      Obtain a' ← according to ϵ-greedy using s'
      Update Q(s,a) ← Q(s,a)+ μ (r+ γQ(s',a')-Q(s,a))
      s ← s'
      a ← a'
    end while
    m ← m+1
  end while
  Print Q(s,a)
  Print s
  Store Q matrix, maximum Reward-Antibody pair and reward in repertoire
  Do action
  Run Clustering Algorithm
end for
```

1

Figure 14. CODA algorithm pseudocode.

Author details

Daniel R. Ramirez Rebollo*, Pedro Ponce Cruz and Arturo Molina

*Address all correspondence to: dan.ramirez@itesm.mx

Tecnologico de Monterrey, Mexico, DF, Mexico

References

[1] Lawson, Anton E. The Neurological Basis of Learning, Development and Discovery: Implications for Science and Mathematics Instruction. 18th ed. Springer Science & Business Media; 2006.

[2] Marsland, Stephen. Machine Learning: An Algorithmic Perspective. CRC Press; Boca Raton, FL. 2014.

[3] Pinto, José Carlos L and Von Zuben, Fernando J. Fault detection algorithm for telephone systems based on the danger theory. In: Artificial Immune Systems. Springer; Berlin, Heidelberg 2005. pp. 418–431.

[4] Luh, Guan-Chun and Liu, Wei-Wen. An Immunological Approach to Mobile Robot Navigation. INTECH Open Access Publisher; Taiwan, Republic of China 2008.

[5] Su, Mu-Chun, Yang, Yuan-Shao, Lee, Jonathan and Chen, Gwo-Dong. A neuro-fuzzy approach for compensating color backlight images. Neural Processing Letters. 2006;23(3):273–287.

[6] Wolpert, David H and Macready, William G. No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation. 1997;1(1):67–82.

[7] Atkeson, Christopher G and Schaal, Stefan. Robot learning from demonstration. In: ICML; 1997. pp. 12–20.

[8] Bakker, Paul and Kuniyoshi, Yasuo. Robot see, robot do: An overview of robot imitation. In: AISB96 Workshop on Learning in Robots and Animals; 1996. pp. 3–11.

[9] Dillmann, Rüdiger, Kaiser, M and Ude, Ales. Acquisition of elementary robot skills from human demonstration. In: International Symposium on Intelligent Robotics Systems; 1995. pp. 185–192.

[10] Ikeuchi, Katsushi and Suehiro, Takashi, Tanguy, Peter and Wheeler, Mark. Assembly plan from observation. Symbolic Visual Learning. 1990;193–224.

[11] Billard, Aude, Calinon, Sylvain, Dillmann, Rdiger and Schaal, Stefan. Robot programming by demonstration. In: Springer Handbook of Robotics. Springer; Berlin, Heidelberg 2008. pp. 1371–1394.

[12] Schaal, Stefan and others. Learning from demonstration. Advances in Neural Information Processing Systems. 1997;1040–1046.

[13] aus der Wieschen, Maria Vanessa, Fischer, Kerstin and Krüger, Norbert. Error feedback for robust learning from demonstration. In: Proceedings of the Tenth Annual ACM/IEEE International Conference on Human-Robot Interaction Extended Abstracts; Portland, Oregon, USA: 2015. pp. 225–226. DOI: 978-1-4503-3318-4

[14] Michini, Bernard, Walsh, Thomas J, Agha-Mohammadi, Ali-Akbar and How, Jonathan P. Bayesian nonparametric reward learning from demonstration. IEEE Transactions on Robotics. 2015;31(2):369–386.

[15] Brys, Tim, Harutyunyan, Anna, Suay, Halit Bener, Chernova, Sonia, Taylor, Matthew E, and Nowé, Ann. Reinforcement learning from demonstration through shaping. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI); 2015.

[16] Spong, Mark W, Hutchinson, Seth and Vidyasagar, Mathukumalli. Robot Modeling and Control. 3rd ed. Wiley, New York; 2006.

[17] de Wit, Carlos Canudas, Siciliano, Bruno and Bastin, Georges. Theory of Robot Control. Springer Science & Business Media; London, UK 2012.

[18] Turing, Alan M. Computing machinery and intelligence. Mind. 1950: 433–460.

[19] Thorndike, Edward Lee. Animal Intelligence: Experimental Studies. Macmillan; New York, USA 1911.

[20] Abbas, Abul K, Lichtman, Andrew HH, and Pillai, Shiv. Basic Immunology: Functions and Disorders of the Immune System. Elsevier Health Sciences; Philadelphia, PA 2012.

[21] Dipankar Dasgupta. Artificial Immune Systems and their Applications. Springer; Berlin, Heidelberg 1998.

[22] Hunt, John E and Cooke, Denise E. Learning using an artificial immune system. Journal of Network and Computer Applications. 1996;19(2):189–212.

[23] Golberg, David E. Genetic Algorithms in Search, Optimization, and Machine Learning. Addion Wesley. Massachusetts, USA 1989;

[24] Hart, Emma and Timmis, Jon. Application areas of AIS: The past, the present and the future. Applied Soft Computing. 2008;8(1):191–201.

[25] Garrett, Simon M. How do we evaluate artificial immune systems? Evolutionary Computation. 2005;13(2):145–177.

[26] Ayara, Modupe, Timmis, Jon, De Lemos, Rogério and Forrest, Simon. Immunising automated teller machines. In: Artificial Immune Systems. Springer; Berlin, Heidelberg 2005. pp. 404–417.

[27] Dasgupta, Dipankar, KrishnaKumar, Kalmanje, Wong, D and Berry, Misty. Negative selection algorithm for aircraft fault detection. In: Artificial Immune Systems. Springer; Sicily, Italy 2004. pp. 1–13.

[28] Taylor, Dan W, and Corne, David W. An investigation of the negative selection algorithm for fault detection in refrigeration systems. In: Artificial Immune Systems. Springer; 2003. pp. 34–45.

[29] Campelo, Felipe, Guimaraes, Frederico G, Igarashi, Hajime, Ramírez, Jaime, Noguchi, So and others. A modified immune network algorithm for multimodal electromagnetic problems. IEEE Transactions on Magnetics. 2006;42(4):1111–1114.

[30] Kalinli, Adem and Karaboga, Nurhan. Artificial immune algorithm for IIR filter design. Engineering Applications of Artificial Intelligence. 2005;18(8):919–929.

[31] Lee, Zne-Jung, Lee, Chou-Yuan and Su, Shun-Feng. An immunity-based ant colony optimization algorithm for solving weapon–target assignment problem. Applied Soft Computing. 2002;2(1):39–47.

[32] Hart, Emma, Ross, Peter, Webb, Andrew and Lawson, Alistair. A role for immunology in "next generation" robot controllers. In: Artificial Immune Systems. Springer; 2003. pp. 46–56.

[33] Lau, Henry YK, Wong, Vicky WK, and Lee, Ivan SK. Immunity-based autonomous guided vehicles control. Applied Soft Computing. 2007;7(1):41–57.

[34] Philip, Ninan Sajeeth. Optimal selection of training data for the difference boosting neural networks. In: Proc. of iAstro., Guan-Chun and Liu, Wei-Wen (editors). An Immunological Approach to Mobile Robot Navigation. INTECH Open Access Publisher; 2008.

[35] Zhong, Yanfei and Zhang, Liangpei, Huang, Bo, and Li, Pingxiang. An unsupervised artificial immune classifier for multi/hyperspectral remote sensing imagery. IEEE Transactions on Geoscience and Remote Sensing. 2006;44(2):420–431.

[36] Nasraoui, Olfa, Rojas, Carlos and Cardona, Cesar. A framework for mining evolving trends in web data streams using dynamic learning and retrospective validation. Computer Networks. 2006;50(10):1488–1512.

[37] Artificial immune systems. In: Proceedings of ICARIS; Springer; 2004. DOI: LNCS3239

[38] Jacob, Christian, Pilat, Marcin, Bentley, Peter and Timmis, Jonathan. Artificial Immune Systems: 4th International Conference Proceedings. In: ICARIS; August 14-17; Alberta Canada. Springer; 2005.

[39] de Castro, L.. Available from: http://www.dca.fee.unicamp.br/lnunes/AIS.html

[40] Timmis, Jon, Neal, Mark and Hunt, John. An artificial immune system for data analysis. Biosystems. 2000;55(1):143–150.

[41] Timmis, Jonathan, Hone, Andrew, Stibor, Thomas and Clark, Edward. Theoretical advances in artificial immune systems. Theoretical Computer Science. 2008;403(1):11–32.

[42] Burnet, Sir Frank Macfarlane and others. The Clonal Selection Theory of Acquired Immunity. University Press Cambridge; 1959.

[43] Villalobos-Arias, Mario, Coello, Carlos A Coello, and Hernández-Lerma, Onésimo. Convergence analysis of a multiobjective artificial immune system algorithm. In: Artificial Immune Systems. Springer; 2004. pp. 226–235.

[44] Peter Dayan and Laurence F. Abbott, Theoretical Neuroscience; Computational and Mathematical Modeling of Neural Systems. MIT Press; 2001.

[45] Cordella, Francesca. Grasping algorithms for anthropomorphic robotic hands inspired to human behavior [thesis]. Faculty of Engineering, University of Naples Federico II. 2011.

[46] Romero, Javier, Kjellstrom, Hedvig, and Kragic, Danica. Modeling and evaluation of human-to-robot mapping of grasps. In: IEEE International Conference on Advanced Robotics ICAR; 2009. pp. 1–6.

[47] Romero, Javier. From human to robot grasping [thesis]. KTH Royal Institute of Technology: 2011.

[48] Dang, Hao and Allen, Peter K. Semantic grasping: planning task-specific stable robotic grasps. Autonomous Robots. 2014;37(3):301–316. DOI: Springer

[49] Borovicka, Tomas, Jirina Jr, Marcel, Jirina, Marcel and Kordik, Pavel. Selecting Representative Data Sets. INTECH Open Access Publisher; 2012.

[50] Tommasi, Tatiana, Orabona, Francesco and Caputo, Barbara. Safety in numbers: Learning categories from few examples with multi model knowledge transfer. In: IEEE Conference on Computer Vision and Pattern Recognition; 2010. pp. 3081–3088.

[51] Brest, Janez and Maucec, Mirjam Sepesy. Population size reduction for the differential evolution algorithm. Applied Intelligence. 2008;29:228–247. DOI: Springer

[52] Goldfeder, Corey and Allen, Peter K. Data-driven grasping. Autonomous Robots. 2011;1:1–20. DOI: Springer

[53] Philip, Ninan Sajeeth. Optimal selection of training data for the difference boosting neural networks. Proc. of iAstro, Nice, France, October; 16–17.