ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

Learning Contextual Affordances with an Associative Neural Architecture

Francisco Cruz, German I. Parisi, and Stefan Wermter

University of Hamburg - Department of Informatics
Vogt-Koelln-Strasse 30, 22527 Hamburg - Germany
http://www.informatik.uni-hamburg.de/WTM/

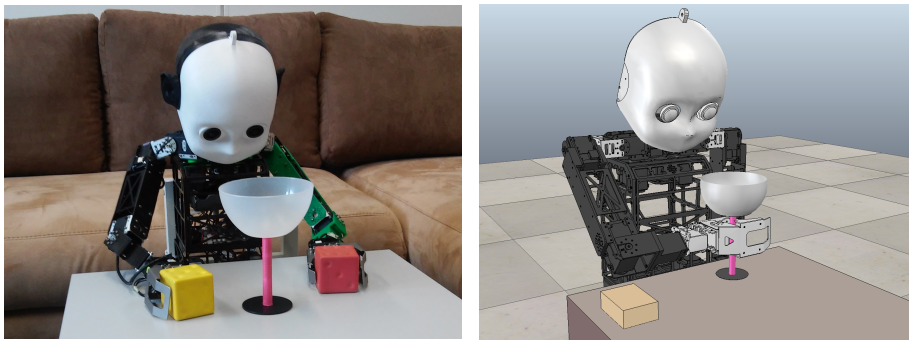
Abstract. Affordances are an effective method to anticipate the effect of actions performed by an agent interacting with objects. In this work, we present a robotic cleaning task using contextual affordances, i.e. an extension of affordances which takes into account the current state. We implement an associative neural architecture for predicting the effect of performed actions with different objects to avoid failed states. Experimental results on a simulated robot environment show that our associative memory is able to learn in short time and predict future states with high accuracy.

1 Introduction

Robots are increasingly being used in diverse fields of application and it is expected that they will carry out dexterous tasks in real time. Therefore, the anticipation and resolution of conflict situations that may lead to mistakes or incomplete tasks is a desired property for robots aiming to successfully operate in real-world environments. In this work, we extend a reinforcement learning scenario that consists of a robot in front of a table with the aim to clean it [1]. During the execution of this task, the robot will transit different states by performing actions and using objects until a desired final state is achieved. However, there are actions that cannot be performed in certain states since they may lead to a failed state, thereby preventing the robot to successfully finish its task. To deal with this issue we use affordances [2], which are a learning model that allows to predict the effect of performing an action utilizing an object. Nevertheless, this scheme does not take into account the current state of the agent and hence, the information needed as input to anticipate the effect is incomplete. In this regard, we propose an extension to this model called *contextual affordances* that considers the current state as an additional input variable in order to accurately predict the effect of an action using an object.

We implement an architecture containing a layer with a quadratic complex neuron [3] to learn and associate the contextual affordances. The associative architecture shapes a virtual grid in a complex plane to map inputs into the output space. This architecture allows us to train our model with few iterations obtaining accurate results in a simulated environment with a humanoid robot that must clean a table interacting with different objects.

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.



(a) In this scenario, the affordance of graspability is temporally unavailable.

(b) Our simulated scenario with two objects: a cup and a sponge.

Fig. 1: A real (a) and a simulated (b) robotic scenario.

2 Contextual Affordances

Affordances are available action possibilities for an agent in its environment [2]. They represent characteristics of the relation between an agent and an object in terms of opportunities the object offers to the agent[4]. In robotics, they have been used as a triplet:

$$\mathit{affordance} := < \mathit{action}, \mathit{object}, \mathit{effect} >, \tag{1}$$

which encodes relationships between its components [5][6]. Therefore, it is possible to predict the effect using actions and objects as domain variables, i.e. $\mathit{effect} = f(\mathit{action}, \mathit{object})$.

Nevertheless, although this model has been shown to be suitable for many scenarios, it does not include context information which allows to properly anticipate the effects in all situations [7]. We would like to point out that the fact of being able to use or not an affordance in a given state does not determine the existence of the affordance itself. Conversely, the affordance is still present but cannot be applied at this state, or it can imply a different effect using a certain action with a given object. Let us consider the scenario shown in Fig. 1a: a cup affords grasping, as does a die, but in the case that an agent has both hands occupied with two dice, then it will not be able to also grasp the cup, i.e. the affordance is temporarily unavailable.

To overcome this issue, it is possible to use contextual affordances where an additional variable is considered to introduce information about the current state [7]. In this case, the previous triplet is now extended to:

$$\mathit{contextualAffordance} := < \mathit{state}, \mathit{action}, \mathit{object}, \mathit{effect} >. \tag{2}$$

Using this tuple, we then can predict the effect by considering the function $\mathit{effect} = f(\mathit{state}, \mathit{action}, \mathit{object})$. For instance, given two affordances using the

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

same action a and the same object o, but at different states $s_1 \neq s_2$, they may generate different effects $e_1 \neq e_2$. It is unfeasible to establish differences between these affordances without state information, given that $e_1 = f(a, o)$ and $e_2 = f(a, o)$ would suggest $e_1 = e_2$. Therefore, dealing with the current states $s_1 \neq s_2$, an agent will distinguish each case and learn at the same time by using contextual affordances to predict the effects $e_1 \neq e_2$ by $e_1 = f(s_1, a, o)$ and $e_2 = f(s_2, a, o)$ establishing clear differences between them [1].

In some cases, the object can also be a location, e.g., a hill affords climbing if the action is to climb and the object, or rather the location, is the hill. In general, we use the term object to refer to both objects and locations.

3 Associative Neural Architecture

We develop an associative neural architecture with a complex-valued quadratic neuron [8] to define a new two-dimensional grid on the output space as presented in [3]. For an input vector $X \in \mathbb{C}^n$, the scalar complex output is $y = X^* A X$, where $A \in \mathbb{C}^{n \times n}$ is the weight matrix and X^* denotes the conjugate transpose. The output can be written as the summation of the individual terms that involve the components of X and A:

$$y = \sum_{j=1}^{n} \sum_{k=1}^{n} \bar{x}_j x_k a_{jk}. \tag{3}$$

The gradient descent learning rule that minimizes the mean-square error is:

$$\triangle A = \alpha \varepsilon \bar{X} X^T, \tag{4}$$

where α is a small real-valued learning rate. For a given input vector X, the desired output Y to be used in the learning algorithm is defined as the nearest intersection point of the grid lines of the complex plane. In practice, a function Ψ is defined that rounds to the nearest integer for grid lines spaced at a fixed distance δ in both directions:

$$\Psi(Y) = \frac{round(\delta Re(Y))}{\delta} + i \frac{round(\delta Im(Y))}{\delta}. \tag{5}$$

This function creates a virtual grid where the output snaps onto the nearest grid corner. The training algorithm is as follows: (i) initialize the weights of the neuron with random values, (ii) compute Y, (iii) compute $d = \Psi(Y)$, and (iv) update the weights of the neuron according to Eq. 4.

At each iteration, the steps (ii) to (iv) are carried out for all the input vectors, so that a cluster in the input space will map to a similar region in the output space due to the continuity of the activation function. The stop criterion can be a fixed number of iterations, a decreasing learning rate, or a given minimum mean-square error over all inputs.

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

Data Representation																
Side conditions					**Locations**				**Actions**					**Objects**		
dd	1	0	0	0	home	1	0	0	get	1	0	0	0	sponge	1	0
dc	0	1	0	0	left	0	1	0	drop	0	1	0	0	cup	0	1
cd	0	0	1	0	right	0	0	1	goto	0	0	1	0	free	0	0
cc	0	0	0	1	none	0	0	0	clean	0	0	0	1			

Table 1: Representation of training data used for neural classification.

4 Robotic Scenario

The task consists of a robot standing in front of a table to clean it. The robot can use one arm and its gripper to manipulate a set of objects in order to complete the cleaning task. For this task, we define *objects*, *locations*, and *actions*. The scenario includes two objects: a *sponge* and a *cup*. The table is divided in three zones, the *left* and *right* table sides and an additional position called *home* where the robot can place the sponge during the execution of the task. We allow the robot to perform four actions: *get* <*object*>, *drop* <*object*>, *goto* <*location*>, and *clean* the table section where the robot arm is placed at that moment. The robotic-cleaning task in a simulated environment is depicted in Fig. 1b.

Each robot state in the scenario takes into account four variables: (i) the robot's hand position, (ii) the object held in its hand (if any), (iii) the position of the cup, and (iv) the condition of each side of the table, i.e. whether the surface is clean or dirty. The vector state is represented as:

$$s_t =< handPosition, handObject, cupPosition, sideCondition > . \quad (6)$$

Nevertheless, from a given state the robot could perform actions that lead to a failed state, i.e. a state from where it is not possible to complete the task. For instance, let us assume the current state $s_t =< right, sponge, right, (dirty, dirty) >$, i.e. the cup is placed on the right side of the table and the robot's hand is above it holding the sponge. If the robot then cleans the *right* section of the table, it may shatter the cup, therefore, it is not feasible to finish the cleaning task from the next state s_{t+1}.

We encode all the variables as presented in Table 1, where we show the data representation for side conditions, locations, actions, and objects. In side conditions, letters d and c represent the fact of being dirty or clean respectively.

5 Experimental Results

Our approach uses contextual affordances to predict the effect of an action after it has been performed by the robot. We use the representation shown in Table 1 to represent the training data. As input, we use vectors with 21 variables containing information about the current state, the action, the object and/or the location, whereas each state is contained in the first 12 components of the vector

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

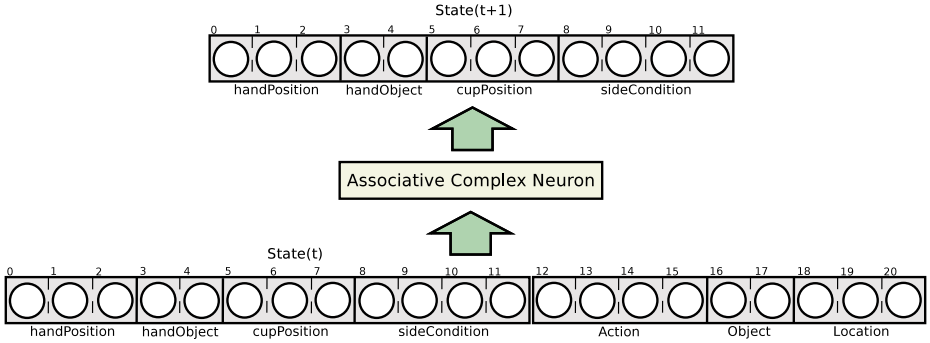
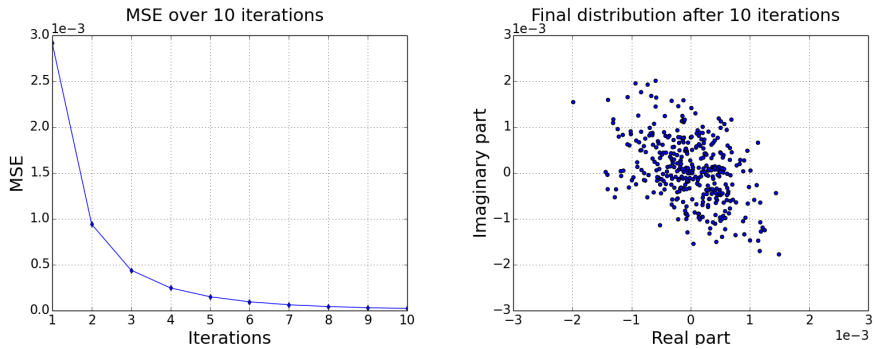


Fig. 2: Associative neural architecture for next state prediction. In our scenario, the state reached by the robot represents the affordance effect.



(a) Mean squared error over 10 training iterations.

(b) Final distribution of the output projected into the complex domain.

Fig. 3: Training error (a) and final distribution (b) of the associative layer.

considering the four variables that define a state (see Fig. 2). Our architecture comprises an associative neural layer that maps the current state of the system into the expected effect, that corresponds to the effect from contextual affordances encoded as 12 variables representing the next state. When a performed action leads to a failed state, all components of the output vector are equal to zero. The data were created considering all possible states together with actions and objects (or locations). The total number of data samples was 368 instances for the training of the associative layer.

During the training, we associate the desired output state label $l(\Psi(Y))$ for classification purposes. After the training phase, when a new sample is presented to the neuron, we compute y' and return the state label that minimizes $\|\Psi(y') - \Psi(Y)\|$. For our implementation, we set $\delta = 0.001$ and used the decaying learning rate:

$$\alpha_t = \alpha_0 * e^{\frac{-t(t+3)}{k}}, \tag{7}$$

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

where t is the iteration number, $\alpha_0 = 0.01$ and $k = 5000$.

Experiments show that our architecture with an associative layer is able to classify all the instances correctly after training. The mean square error decreased from 2.92e-3 to 2.37e-5 after 10 iterations as shown in Fig. 3a. The final distribution of the output after 10 iterations is shown in Fig. 3b, where the x and y axes are the real and imaginary parts respectively of the complex plane.

6 Conclusions and Future Work

Our proposed architecture is able to successfully predict the effect of performing an action using an object by using contextual affordances. We use additional state information to distinguish different situations in a robotic cleaning scenario and avoid failed states to effectively finish the task. The associative complex architecture allows to map the input vectors into valid states with few training iterations, which represents an advantage for online learning applications where the response time plays a crucial role.

As future work, we will extend the simulated scenario to a real robot platform obtaining the input vector using a vision sensor and the output vector from the real state of the robot after performing the action in the cleaning scenario.

Acknowledgment

The authors gratefully acknowledge partial support by the Universidad Central de Chile, CONICYT scholarship 5043, the DAAD German Academic Exchange Service (Kz:A/13/94748) under CASY project, the German Research Foundation DFG under project CML (TRR 169), and the Hamburg Landesforschungsförderungsproject.

References

[1] F. Cruz, J. Twiefel, S. Magg, C. Weber, and S. Wermter, *Interactive reinforcement learning through speech guidance in a domestic scenario*, in The International Joint Conference on Neural Networks (IJCNN), pp. 1341–1348, 2015.

[2] J. J. Gibson, *The Ecological Approach to the Visual Perception of Pictures*, Boston: Houghton Mifflin, 1979.

[3] G. Georgiou and K. Voigt, *Self-organizing maps with a single neuron*, in The International Joint Conference on Neural Networks (IJCNN), pp. 1–6, 2013.

[4] T.E. Horton, A. Chakraborty, and R. St. Amant, *Affordances for robots: a brief survey*, in AVANT: Journal of Philosophical-Interdisciplinary Vanguard, Vol (2), pp. 70–84. 2012

[5] E. Şahin, M. Çakmak, M. R. Doğar, E. Uğur, and G. Üçoluk, *To afford or not to afford: A new formalization of affordances toward Affordance-Based robot control*, in Adaptive Behavior, Vol. 15, no. 4, pp. 447–472, 2007.

[6] L. Montesano, M. Lopes, A. Bernardino, and J. Santos-Victor, *Learning Object Affordances: From Sensory-Motor Coordination to Imitation*, in IEEE Transactions on Robotics, Vol. 24, No. 1, pp. 15–26, 2008.

[7] M. Kammer, T. Schack, M. Tscherepanow, and N. Yukie, *From Affordances to Situated Affordances in Robotics - Why Context is Important*, in Frontiers in Computational Neuroscience, Conference Abstract IEEE ICDL-EpiRob, Vol. 5(30), 2011.

[8] G. Georgiou, *Exact interpolation and learning in quadratic neural networks*, in The International Joint Conference on Neural Networks (IJCNN), pp. 230–234, 2006.