A Deep Probabilistic Framework for Heterogeneous Self-Supervised Learning of Affordances

Atabak Dehban[1,2], Lorenzo Jamone[3,1], Adam R. Kampff[4] and José Santos-Victor[1]

***Abstract*— The perception of affordances provides an action-centered parametric representation of the environment. By perceiving an object's visual features in terms of what actions they afford, novel behavior opportunities can be inferred about previously unseen objects. In this paper, a flexible deep probabilistic framework is proposed which allows an explorative agent to learn tool-object affordances in continuous space. To this end, we use a deep variational auto-encoder with heterogeneous probabilistic distributions to infer the most probable action that achieves a desired effect or to predict a parametric probability distribution over action consequences i.e. effects. Our experiments show the generalization of the method to unseen objects and tools and we have analyzed the influence of different design choices. Our framework goes beyond other proposals by incorporating various probability distributions tailored for each individual modality and by eliminating the need for any pre-processing of the data.**

I. INTRODUCTION

Goal-driven agents are required to continuously select actions that bring their perceptual stimuli closer to a desired state [1]. However, predicting the consequences of actions remains a non-trivial task for different agents with varying degrees of intelligence. In addition, actions and their outcomes are tightly linked to the contextual environment, meaning that if specific prerequisites are not satisfied, applying the action would be impossible, useless or even harmful [2].

The complexity of the problem notwithstanding, humans and other animals seem to flexibly select and execute proper actions to handle everyday tasks and they can generalize their knowledge to new situations. Affordances, introduced in the seminal work of J. J. Gibson [3], provide a partial solution by connecting the essential properties of objects and surroundings with actions an agent can perform.

Affordances are dynamic environment–agent relations which are learned through embodied interaction processes by "discovering distinctive features and invariant properties of things and events" [4]. In robotics, affordances are often represented as the triplet (*entity, action, effect*) [5], [6].

In this work, an iCub robot [7] performs simple actions with tools on a set of objects and observes the resulting effect. By repeating the experiments several times, the robot is able to learn a probabilistic interpretation of affordance variables. This is done with a Variational Auto-Encoder (VAE) [8], [9] which allows approximating intractable posterior distributions by minimizing a cost function.

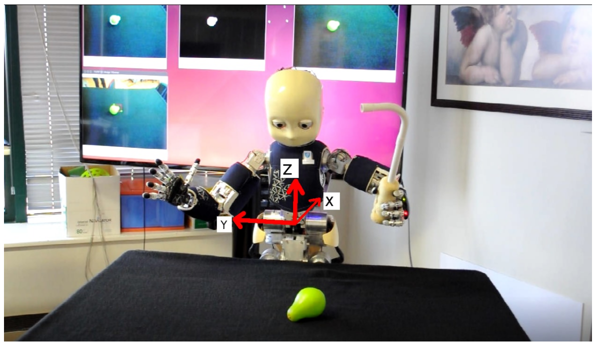


Fig. 1: Experimental setup, showing the iCub humanoid robot at the beginning of a robot–object interaction trial, and the reference frame annotation. In the background screen we show some intermediate results of the visual perception routines.

VAEs were selected because they allow black box approximations of multiple target probability distributions, thus providing two desirable properties: (1) no pre-processing is required to transform the data into a target shape, enabling one to fuse multiple heterogeneous distributions which naturally fit to the data and (2) it is possible to increase the amount of training data incrementally until no further performance improvements are observed..

Our main contribution in this paper is to propose a probabilistic framework to learn objects and tools affordances in a continuous space with flexible probability distributions in a self-supervised manner and suitable for incremental learning. To the best knowledge of authors, training VAEs with multiple target probability distributions is unprecedented in the literature as it poses many challenges during training like domination of a specific distribution over others. In addition, the way to combine different distributions of various modalities is non-trivial. Thus, as our second contribution, we propose several techniques and assumptions to mitigate the severity of these difficulties which allow us to train a VAE and use it in a dataset collected by the robot. By remaining faithful to the probability distribution of each modality in the dataset, we were able to train the model without any pre-processing of the dataset, thus achieving a fully self-supervised learning of affordances in the sense that the knowledge of each modality contributes to the knowledge

[1]Institute for Systems and Robotics, Instituto Superior Técnico, Universidade de Lisboa, Lisbon, Portugal.
{adehban, jasv}@isr.tecnico.ulisboa.pt
[2]Champalimaud Research, Champalimaud Centre for the Unknown, Lisbon, Portugal.
[3]ARQ (Advanced Robotics at Queen Mary), School of Electronic Engineering and Computer Science, Queen Mary University of London, UK.
l.jamone@qmul.ac.uk
[4]Sainsbury Wellcome Centre for Neural Circuits and Behaviour (SWC), London, UK. a.kampff@ucl.ac.uk

of other ones [10].

The paper is organized as follows: in Sec. II we briefly review the recent literature on tool use in affordances. Sec. III will be devoted to develop the theory of using VAEs to learn objects and tools affordances. The results of our experiments, detailing multiple views on different aspects of the problem will be presented in Sec. IV. Finally we draw our conclusions and discuss possible future directions of research in Sec. V.

II. PREVIOUS WORK

In this section, we provide a succinct review on affordance learning and modeling relevant to our approach. This section only serves to highlight the differences of our work with regard to state-of-the-art advances in the field and we direct interested reader to [11] for a recent and comprehensive review of affordances across several fields and [12] for a survey of affordance research in developmental robotics.

Sun et al. [13] have simplified learning of a probabilistic affordance model by using object categories as hidden variables. They have shown such new formulation is scalable to various affordance categories and enables an incremental learning of affordances while being more data efficient. Even though using latent variables may significantly simplify the learning of a probabilistic model and, as such, we also use a similar formulation, there are significant conceptual and computational differences between the two approaches. First, using object categories to determine their usability is orthogonal to the Gibsonian view of direct perception (a graspable object with a sharp end is a knife). On the other hand, we introduce latent variables only as a means to simplify learning of relations in the probabilistic model, overcoming the limitation of providing supervised category labels to learn affordances. Moreover, because these latent variables summarize the relation between observable random variables, they can later be used in a classifier similar to [14] to infer object or tools categories. Second, [13] have used discrete affordance labels. In contrast, we define affordances as learned relations between object and tools' visual features with actions and effects.

Osório et al. [15] have used Gaussian Mixture Models to account for continuous observation of object features and effects. As suggested by their results, considering the continuous nature of the variables reduces the estimated parameter errors with respect to the parameters that generated the simulated dataset. However, they assumed a known distribution of the sensor noise model, which is not always feasible when dealing with real data.

Gonçalves et al. [16], [17] have extended the Bayesian network formulation of [5] to incorporate the notion of tools. This model was further developed in [18] to allow the robot to generalize the learned affordances of its own hands to unseen tools. In a scenario similar to [16] we proposed the usage of denoising auto-encoders to learn the affordances of objects and tools in continuous space [19]. Our model could use the continuous nature of features and effects while improving on the accuracy of previous works over different measures of performance. However, this model needs pre-processing of the data, which is undesirable, as different choices of this pre-processing considerably influence the performance.

Complementary to our approach, Mar et al. [20] have used the iCub robot to learn the effects of applying different actions with different tools in a self-supervised manner. They have first reduced the dimensions of tool features and action–effect vectors using two separate Self Organizing Maps and learned a mapping from one space to another. This mapping does not take the object's shape into considerations and thus, the influence of object on the measured effects is not studied.

Recently, Chavez-Garcia et al. [21] proposed the usage of Gaussian Bayesian networks to model the relations between objects' features in continuous space with effects of different actions. The network provides means and covariances of a query variable given the observed variables, and the structure is learned by maximizing Bayesian or Akaike information criterion score.

Advances in deep learning research combined with the ability of variational methods in approximating intractable posterior densities, make VAEs an interesting area of research for robotics. E.g. Sung et al. [22] have used VAEs to learn latent variables over haptic data and later used those variables to control a robot in a Partially Observable Markov Decision Process setting. Similar to our approach, their VAE is also responsible to learn representations over different modalities but they have assumed Gaussian distributions for all the modalities. In contrast, in this paper we are investigating the usage of VAEs over non-equal observation distributions.

This paper differs from all previous approaches by proposing a computational model which is suitable for desired heterogeneous probability distributions. In our experiments, the features of objects and tools are *defined* as bounded continuous variables between zero and one, and thus, they cannot be properly modeled by a distribution with unbounded domain like multivariate Gaussians. The nature of our applied actions are discrete and requires a categorical distribution. On the other hand, the effects generated by actions are correlated and cannot be constrained by an optimal upper or lower bound. Using a VAE, all the above desiderata can be satisfied in one framework as explained in the next section. Since the nature of each data modality is explicitly considered, no pre-processing is applied to the training set.

III. METHODS

We start this section by explaining the theory of VAEs in Sec. III-A. Afterwards, in Sec. III-B we introduce the changes we have applied to the standard VAE to make it suitable for learning of affordances.

A. *Variational auto-encoder*

A variational auto-encoder is a generative model which aims to find latent factors of variation which can explain the observations while being as independent as possible. Consider a dataset of observations $\mathcal{D} = \{\boldsymbol{x}^i \mid i = 1, \ldots, N\}$, where each $\boldsymbol{x}^i \in \mathcal{R}^n$ are i.i.d. observations. The goal is to find a distribution $q_{\boldsymbol{\phi}}\left(\boldsymbol{z}^i \mid \boldsymbol{x}^i\right)$, $\boldsymbol{z}^i \in \mathcal{R}^m$ from a family of known distributions (usually independent Gaussians) parameterized by $\boldsymbol{\phi} = \{\boldsymbol{\mu}_{\boldsymbol{z}}(\boldsymbol{x}), \boldsymbol{\sigma}_{\boldsymbol{z}}(\boldsymbol{x})\}$ that minimizes the Kullback–Leibler (KL) divergence

$$\mathbb{D}_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}\left(\boldsymbol{z}^i \mid \boldsymbol{x}^i\right) \parallel p_{\boldsymbol{\theta}}\left(\boldsymbol{z}^i \mid \boldsymbol{x}^i\right)\right) \tag{1}$$

over all training samples, where $\boldsymbol{z}^i$ are the latent variables and $p_{\boldsymbol{\theta}}\left(\boldsymbol{z}^i \mid \boldsymbol{x}^i\right)$ is the true posterior density we are trying to approximate with q. By using the definition of KL divergence, (1) can be written as:

$$\mathbb{E}\left[\log q_{\boldsymbol{\phi}}\left(\boldsymbol{z}^i \mid \boldsymbol{x}^i\right)\right] - \mathbb{E}\left[\log p_{\boldsymbol{\theta}}\left(\boldsymbol{z}^i, \boldsymbol{x}^i\right)\right] + \log p_{\boldsymbol{\theta}}\left(\boldsymbol{x}^i\right), \tag{2}$$

where all expectations are taken with respect of q and we have used the Bayes identity $p_{\boldsymbol{\theta}}\left(\boldsymbol{z}^i \mid \boldsymbol{x}^i\right) = p_{\boldsymbol{\theta}}\left(\boldsymbol{z}^i, \boldsymbol{x}^i\right)/p_{\boldsymbol{\theta}}\left(\boldsymbol{x}^i\right)$. Notice that calculating the log likelihood of complete data (last term in (2)) is intractable [23]. As a result, true posterior distribution cannot be recovered and exact inference wouldn't be possible. Hence, in variational inference the following loss function will be optimized:

$$\mathcal{L}^i = \mathbb{E}\left[\log q_{\boldsymbol{\phi}}\left(\boldsymbol{z}^i \mid \boldsymbol{x}^i\right)\right] - \mathbb{E}\left[\log p_{\boldsymbol{\theta}}\left(\boldsymbol{z}^i, \boldsymbol{x}^i\right)\right]. \tag{3}$$

Equation (3) is written for a single training sample. The quantity $\mathcal{L} = \sum_{i=1}^{N} \mathcal{L}^i$, also known as Evidence Lower BOund (ELBO), can be re-written as

$$\mathcal{L} = \mathbb{D}_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{z})\right) - \mathbb{E}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z})\right], \tag{4}$$

where $p_{\boldsymbol{\theta}}(\boldsymbol{z})$ is the prior over latent variables and, to simplify computations and provide a closed form solution, a popular choice is to set $p_{\boldsymbol{\theta}}(\boldsymbol{z}) = \mathcal{N}(0,1)$. $\boldsymbol{\theta}$ is the parameter vector of the generative distribution which determines how latent variables combine to generate an observation $\boldsymbol{x}$ and it will be explained further in the next subsection.

B. Affordance learning and modeling

The first term in (4) is the KL divergence between the approximate posterior and prior distributions. It is trying to make the latent variables come from an uninformative standard Gaussian distribution and acts as a regularizer to ensure that independent latent variables are learned, potentially making the network less prone to overfit. The second term, however, tries to maximize the log-likelihood of the observation, hence the name *auto-encoder*. It is the interplay between the two which results in learning independent latent factors that generate the data. In this paper, the vector of observations $\boldsymbol{x}$ is a concatenation of $\{\boldsymbol{o}; \boldsymbol{t}; \boldsymbol{a}; \boldsymbol{e}\}$ where $\boldsymbol{o} \in (0,1)^{n_o}$ are the object features, $\boldsymbol{t} \in (0,1)^{n_t}$ are the tool features, $\boldsymbol{a} = \{a_i \mid \sum_i^{n_a} a_i = 1\}$ is the one-hot representation of the applied motor action and $\boldsymbol{e} \in \mathcal{R}^{n_e}$ are the effects. n_o, n_t, n_a and n_e are the dimensions of their respective variables. In order to take the different nature of each variable into account, we represent each of them with their own probability distribution that factorizes conditioned on the latent variable, i.e.

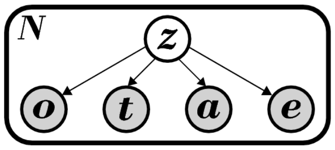


Fig. 2: Graphical model of affordance network. Grey nodes are observed and white node is hidden. o: object, t: tool, a: action, e: effect, z: latent variable, and N: number of samples.

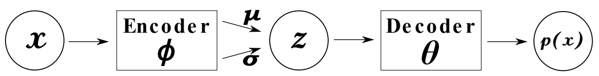


Fig. 3: Schematic of the variational auto-encoder. Circles represent random variables and rectangles represent fully connected neural networks.

$$p_{\boldsymbol{\theta}}(\boldsymbol{x}) = p_{\boldsymbol{\theta}}(\boldsymbol{o}) \cdot p_{\boldsymbol{\theta}}(\boldsymbol{t}) \cdot p_{\boldsymbol{\theta}}(\boldsymbol{a}) \cdot p_{\boldsymbol{\theta}}(\boldsymbol{e}),$$

where conditioning on $\boldsymbol{z}$ is not explicitly written to simplify the notation (Fig. 2). By considering this assumption, the log-likelihood part of (4) can be written as:

$$\begin{aligned}\log p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z}) = &\log p_{\boldsymbol{\theta}}(\boldsymbol{o} \mid \boldsymbol{z}) + \log p_{\boldsymbol{\theta}}(\boldsymbol{t} \mid \boldsymbol{z}) + \\ &\log p_{\boldsymbol{\theta}}(\boldsymbol{a} \mid \boldsymbol{z}) + \log p_{\boldsymbol{\theta}}(\boldsymbol{e} \mid \boldsymbol{z}).\end{aligned} \tag{5}$$

With respect to the regularization term in (4), gradient based optimization techniques tend to start reducing it very early in training. As a result, it becomes difficult to recover from this local minima solution. To overcome this problem Sønderby et al. [24] have modified (4) to the following:

$$\mathbb{L} = \beta \cdot \mathbb{D}_{\mathrm{KL}}\left(q_{\boldsymbol{\phi}}(\boldsymbol{z} \mid \boldsymbol{x}) \parallel p_{\boldsymbol{\theta}}(\boldsymbol{z})\right) - \mathbb{E}\left[\log p_{\boldsymbol{\theta}}(\boldsymbol{x} \mid \boldsymbol{z})\right], \tag{6}$$

where β increases from zero to one during training linearly. Our experiments suggest that this simple modification is crucial in training a the network.

Fig. 3 shows the schematic of the whole architecture which is composed of an encoder and a decoder. The parameters $\boldsymbol{\theta}$ and $\boldsymbol{\phi}$ are jointly trained to minimize $\mathbb{L}$.

IV. EXPERIMENTS AND RESULTS

The results of our experiments and implementation details are described in this section. We start by briefly introducing the dataset[1] and the experimental procedure of obtaining the data in Sec. IV-A. Afterwards, we describe the data-augmentation procedure and train, test, and validation split, and finally we introduce the values of the hyper-parameters and usage of the network at test time. Sec. IV-B will introduce the first use case of VAE in determining the correct action and associated metrics will be discussed. Sec. IV-C will explain how well the network can predict the effects of

[1] http://vislab.isr.tecnico.ulisboa.pt/datasets/

applying an action with a tool on an object. It is challenging to quantitatively assess the quality of predictions because unlike the actions, the effects are continuous random variables. Regardless, we provide several measures to investigate how well effects are predicted. Afterwards, in Sec. IV-D we quantify the influence of different design choices on some of the performance metrics and how it was possible to avoid the dominance of one modality over others. Finally, in Sec. IV-E we report the performance of the network on previously unseen objects and tools.

A. Dataset and experiment procedure

The dataset and experiments are fully explained in [25] and a video of the robot during data collection phase is accessible from https://youtu.be/pKa6GNeBfjk. Briefly, the robot holds one of the 3 tools in hand and applies one of the 4 cardinal actions {*pull*, *push*, *tap from left* and *tap from right*} to one of the 11 objects, placed in front of it on a table (Fig. 1). Each experiment is repeated at least 10 times, resulting in more than $3 \times 11 \times 4 \times 10 = 1320$ unique experiments.

To gather the effects information, the robot takes note of the initial position of the object on the table and after executing the action, records the 3D location of the object's geometric center, if the experimenter decides the action was successful i.e. the action was carried out correctly and initial placement of the tool with respect to action was appropriate. Effects will be defined as 2D displacement of the object on the table in front of the robot in meters.

The visual features used in this work are **Convexity**, **Eccentricity**, **Compactness**, **Circleness** and **Squareness** [17] which are collected by placing the objects and tools on different locations and orientations in front of the robot after all the experiments are done. In the case of tools, these features are calculated only for the distant half of the tool to represent its tip. For a detailed description of the dataset and trials please refer to [25].

By having ten independent views of objects and tools, one can augment the data by considering that different views do not change affordances associated with tools and objects if they are reasonably visible. For example, considering the rake, one can posit each of the 10 real trials were performed with the first view of the rake, then the same set of 10 trials were performed with the second view of the rake, effectively increasing the size of the dataset. The same argument goes for the objects. This way, it is possible to reach to a dataset of size (tool views$\times$object views$\times$trials) which results in around 132 000 data points.

After all the points in the dataset are collected, 80% are selected for training and 20% are used only to report the performance. We added a zero mean uniform noise of maximum 1 centimeter to the effects of the training data and selected 10% of this set as validation for hyper-parameter tuning and stopping criterion.

The parameters $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ are the weights and biases of fully connected neural networks with sigmoid activation units. The feature vector $\boldsymbol{x}$ is composed of $n_o = 5$ visual features of the target object and $n_t = 5$ visual features of the used tool, $n_a = 4$ one hot encoded actions and $n_e = 2$ motion vectors of effects in meters, resulting in a vector of size 16. In accordance with the guidelines in [26], each of these four modalities are followed by a separate layer of size 12, resulting in a vector of size 48 which is then, densely connected to another layer of size 50. We also have selected $m = 50$ dimensions for the latent variable $\boldsymbol{z}$ which results in estimating 50 means and 50 variances for the latent variables. The means are passed through a linear activation function, while the variances have softplus, to ensure that they remain positive.

As mentioned at the end of Sec. II, the visual features are defined as bounded values between zero and one. Beta distributions were selected to represent them as they are defined in the desired [0,1] interval and it is flexible in the possible shapes it can get. Furthermore, we posit that individual features become conditionally independent of each other given the values of latent variables. This is justified according to the definition of the features, as no clear relation between them can be observed and even if it appears out of the limitation in the number of available objects and tools, those relations are expected to be captured via the latent variables. In other words:

$$\log p_{\boldsymbol{\theta}}\left(\boldsymbol{o}\right) = \sum_{i=1}^{n_o=5} \log p_{\boldsymbol{\theta}}\left(o_i\right) = \sum_{i=1}^{5} \log \mathrm{Beta}\left(o_i; a_i, b_i\right), \quad (7)$$

where $a_i > 0$ and $b_i > 0$ are shape parameters of their respective distributions and conditioning on $\boldsymbol{z}$ is implicit. Tool features are treated in a similar way. By estimating two parameters for each distribution and having 5 visual features, 10 parameters for objects and 10 separate parameters for the tools will be estimated. These parameters will be the output of softplus nonlinearity to ensure their positiveness.

Actions, as mentioned, are categorical and we have used categorical cross-entropy to estimate the probability of each action, i.e.:

$$\log p_{\boldsymbol{\theta}}\left(\boldsymbol{a}\right) = \sum_{i=1}^{n_a=4} a_i^* \log a_i, \quad (8)$$

where a_i is the output of a sotfmax function, representing the belief that action i was executed and $\boldsymbol{a}^*$ is a one-hot representation of the executed action.

In this work, the effects are considered as 2D displacement of the object's geometric center as being subject to different actions with different tools. Even though the actions are along the cardinal directions, small errors in initial tool tip placement and non-uniformity in weight distribution of objects makes the displacement on x and y axis correlated. To account for this behavior, we model the effects as bi-variate Gaussians, estimating $\boldsymbol{\mu}$, $\boldsymbol{\sigma}$ and ρ which are the vector of means, standard deviations and correlation coefficients

$$p_{\boldsymbol{\theta}}\left(\boldsymbol{e}\right) = \mathcal{N}\left(\boldsymbol{e}; \boldsymbol{\mu}, \boldsymbol{\Sigma}\right) = \frac{1}{2\pi\sigma_x\sigma_y\sqrt{1-\rho^2}} \exp\left[-\frac{m}{2(1-\rho^2)}\right] \quad (9)$$

$$m \triangleq \frac{(e_x-\mu_x)^2}{\sigma_x^2}+\frac{(e_y-\mu_y)^2}{\sigma_y^2}-\frac{2\rho(e_x-\mu_x)(e_y-\mu_y)}{\sigma_x\sigma_y},$$

in which e_x and e_y are the observed effect, μ_x and μ_y are the predicted mean, σ_x and σ_y are the predicted standard deviations on x and y directions, respectively, and

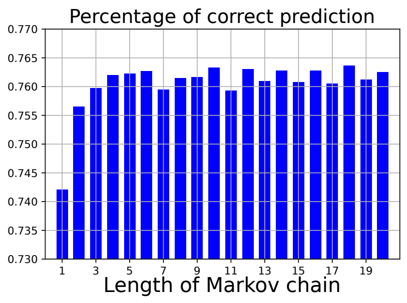
$$\mathbf{\Sigma}=\begin{bmatrix}\sigma_x^2 & \rho\sigma_x\sigma_y\\ \rho\sigma_x\sigma_y & \sigma_y^2\end{bmatrix}$$

is the covariance matrix. Means are estimated with linear activation functions, softplus was used to estimate the standard deviations and, since the correlation coefficient is $\rho \in [-1,1]$ it is estimated using hyperbolic tangent, resulting in estimating 5 parameters for modeling the probability distributions over effects.

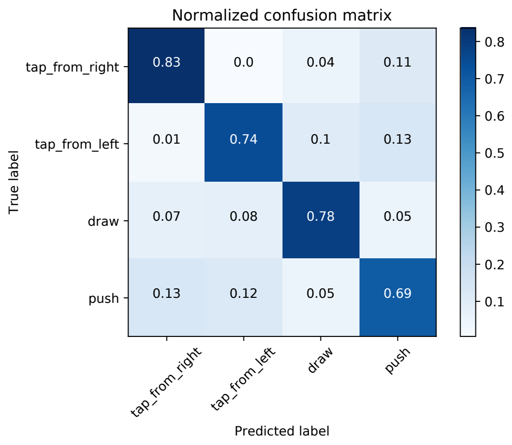
After predicting $\boldsymbol{\mu_z}(\boldsymbol{x})$ and $\boldsymbol{\sigma_z}(\boldsymbol{x})$ from the encoder, we get a sample from $\boldsymbol{z}$ distribution and that sample will be passed through two layers of densely connected neural networks with 50 units in each layer. This representation will be used to estimate the already mentioned $10+10+4+5=29$ parameters.

In all our experiments, we used batch sizes of 3500 samples and Adam [27] training algorithm. Even though Adam uses an adaptive learning rate, we found significant performance differences with various learning rates. This might be attributed to the different natures of individual cost elements $\mathbb{D}_{\mathrm{KL}}(q_{\boldsymbol{\phi}}(\boldsymbol{z}\mid\boldsymbol{x})\parallel p_{\boldsymbol{\theta}}(\boldsymbol{z}))$ of (6) with (7), (8) and (9) (tool cost is omitted), which are ultimately summed together. As each cost would require a bigger or smaller learning rate, we use the Cyclical Learning Rate (CLR) [28] which has shown to improve performance (more details in Sec. IV-D). In all experiments, β is increased from 0.01 to 1 during the first 2000 iterations. We did not see any difference in linear or exponential scheduling for β. Unless stated otherwise, the training is done for 5300 iterations in which the first 100 epochs have a learning rate of 0.01 and, afterwards, the learning rate changes between 1e-3 and 1e-4 with stepsize $=100$.

After the network is trained, in order to query the network with a missing modality (e.g., predicting most probable action given the values of object and tool features and the observed effect), we fix the values of objects, tools and effects and put random numbers between zero and one in place of the action as the input of the auto-encoder and obtain a distribution over actions. Next we sample from this distribution and put the sample in place of the random variables in the previous step. Repeating this procedure many times creates a Markov chain which gets closer and closer to the true distribution [8]. In the case of testing for the original test-set, increasing the length of the Markov chain never significantly worsens the accuracy of the prediction, however, it will reach a saturation which is caused by the approximation on the true posterior density in intractability of data log-likelihood, as we will see subsequently.



(a) Correct action prediction.



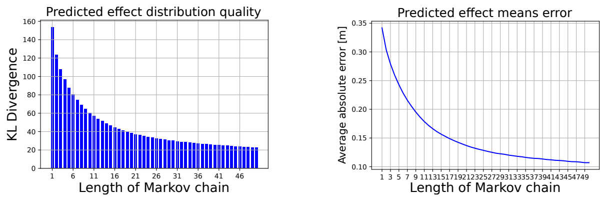
(b) Confusion Matrix.

Fig. 4: Evaluating the performance of the learned model in predicting the correct action, given the values of object and tool features and the desired effect. The confusion matrix is calculated from the 20th iteration of the Markov chain.

B. Action prediction

In this section, we investigate the performance of the network from the action's point of view. Note that by having 4 actions, chance prediction accuracy will be 25%. Fig. 4a shows how many times the most probable action will coincide with the true action. According to this analysis, the prediction accuracy varies between 74 to 76 percent of accuracy after only 3 iteration of the Markov chain. As evident from Fig. 4a, no clear improvement is visible after a Markov chain of length 10. Fig. 4b shows the confusion matrix after 20 iterations of the Markov chain.

In order to investigate the most informative feature for predicting actions, we have taken the derivative of each action parameter a_i with respect to observation vector $\boldsymbol{x}$. This provides us with a measure of sensitivity of action outputs to the input. However, this analysis alone cannot reliably provide us with the most informative feature, as this sensitivity might be caused because of low variations in a specific feature and the network could have learned to rely heavily on that feature. To account for this, we multiplied the derivative by the observed standard deviation of each feature across the whole training set, and we call this quantity as *input-influence*. This means the variables with the same absolute magnitude of derivative will have lower



(a) KL divergence over estimated effect probabilities.

(b) Mean absolute error over test set.

Fig. 5: Performance evaluation of VAE in predicting effects.

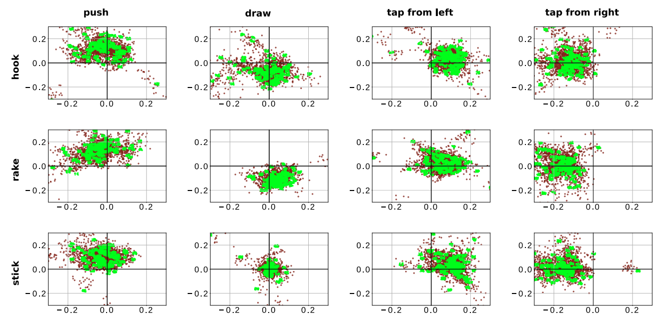


Fig. 6: Distribution of effects, divided by different actions and tools. Green points are observations and red points are predicted means for each of the green points. Note that corresponding to each mean, a covariance matrix is also provided by the network. The axis units are in meters.

input-influence if they have lower standard deviations in the training set.

It was observed that motion on x axis e_x had the highest input-influence on drawing and pushing, where as the motion on y axis e_y had the highest input–influence on taping from left and right (Fig. 1). This can explain why the correct action was predicted 74% of the times, even at the first iteration of the Markov chain. The network has learned to be greatly influenced by the effect observations, and with only having access to the correct effect, it makes correct action predictions most of the time.

C. *Effect prediction*

It is more challenging to provide a quantitative measure of how well the network performs in predicting the effect distributions. This is due to the fact that, in the current formulation, the effect of each trial is a sample from an unknown distribution and because it is impossible to repeat each trial *perfectly* for multiple times, no empirical distribution can be evaluated. One way to overcome this problem is to use the VAE itself to provide us with probabilities.

In the first experiment, initially we put the whole vector of observations $\boldsymbol{x}$ into the VAE and get the probability distributions over $\boldsymbol{e}$ i.e. $p^*_{\boldsymbol{\theta}}(\boldsymbol{e} \mid \boldsymbol{z})$. This distribution is superscripted by $*$ because we treat it as the true distribution. Then we remove the observation over effects and fill its place with random numbers and continue with the Markov chain, this time analyzing the KL divergence between $p^*_{\boldsymbol{\theta}}(\boldsymbol{e} \mid \boldsymbol{z})$ and $p_{\boldsymbol{\theta}}(\boldsymbol{e} \mid \boldsymbol{z})$, which are the obtained distributions: Fig. 5a shows the result of this process.

Fig. 5b shows the mean absolute error $\left|\boldsymbol{e} - \boldsymbol{\mu}\right|$ in meters averaged over all samples of the test set. The error converges to a value around 8 cm after a Markov chain of length 150 (see Fig. 7b). In contrast, the true mean of $\left|\boldsymbol{e} - \boldsymbol{\mu}^*\right|$ is around 4 cm. It should be noted that this result is only a crude estimate, and the distribution of effects and predicted means, separated by actions and tools is shown in Fig. 6. An interesting observation is the higher variance in mean predictions, associated with outliers of the test set.

It turns out the most influential tool feature in predicting effect means is either tool's convexity or eccentricity which are the features that best distinguish between stick-like and rake-like tools.

D. *Reducing the effect of modality dominance*

In order to quantify how the different design choices have contributed to the final performance of the model, here we have compared the result of two variants of the same network with the proposed architecture. In one of the models, we have kept β of (6) constant and equal to 1 during the training. In the second model, after epoch 100 we kept the learning rate constant and equal to 1e-4 to analyse the effect of CLR. The result of this analysis is depicted in Fig. 7.

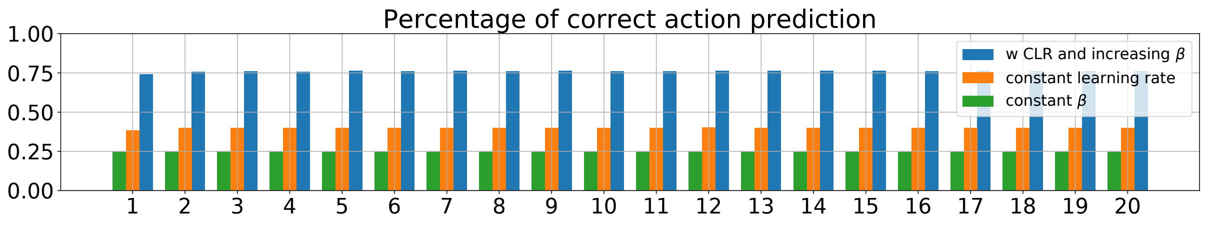
By looking at Fig. 7a, it is clear that keeping a constant β causes the network to simply converge to spaces where the latent variables are very similar to their priors and fails to reduce the cost of other data modalities. On the other hand, with a constant learning rate, the percentage of correct action predictions remains low.

To see that this is an example of one modality dominating another, we can look at Fig. 7b. According to this figure, keeping a constant learning rate has marginally improved the effect prediction, at the cost of a much lower correct action selection. This can be explained by the observation that each component of (5) would require a different learning rate and CLR manages to smoothly change this quantity, thus finding a balanced solution.

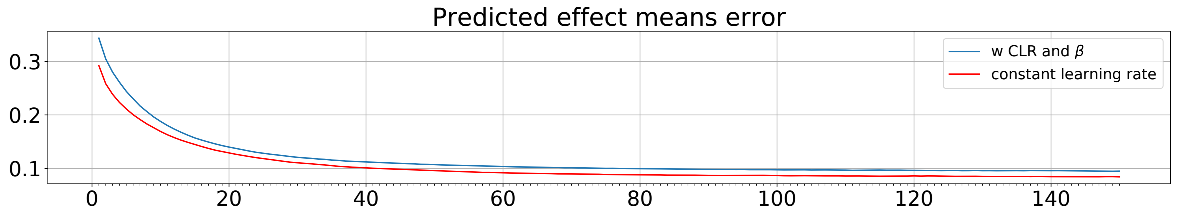
E. *Generalization to novel objects and tools*

One of the benefits of using visual features instead of object categories or labels in learning of affordances is the ability to generalize the already learned knowledge to novel objects and tools where no previous interaction was conducted. To assess the extent to which this generalization is possible, we report the performance of the network on the dataset collected using iCub in [19]. In those experiments, the tools and objects are completely different than the ones used to train the VAE. A video of the experiment is accessible at https://youtu.be/yUzlrHFx8MM. The metrics introduced in Sec. IV-B are demonstrared in Fig. 8 and 9 for the case of new objects and tools.

Similar to Fig. 4a increasing the length of the Markov chain after the first few iterations does not significantly



(a) Correct action prediction.



(b) Mean absolute error of predicted effects.

Fig. 7: Ablation study of the influence of different design choices on two of the performance measures. The mean absolute error in effect prediction of the network with constant β was *off the charts* and is not drawn here. x axis on both plots is the length of Markov chain.

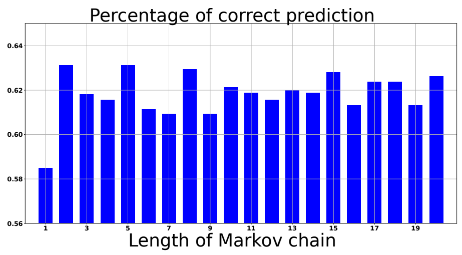
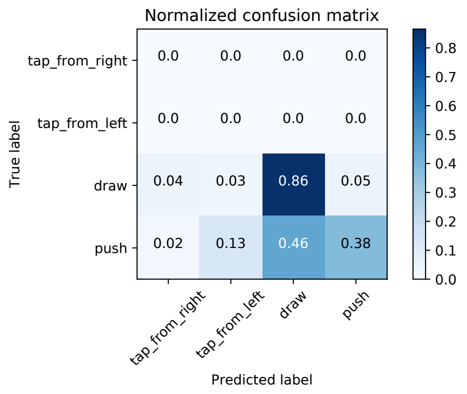


Fig. 8: Correct action prediction over the novel dataset.



	tap_from_right	tap_from_left	draw	push
tap_from_right	0.0	0.0	0.0	0.0
tap_from_left	0.0	0.0	0.0	0.0
draw	0.04	0.03	0.86	0.05
push	0.02	0.13	0.46	0.38

Fig. 9: Confusion Matrix for the novel dataset with a Markov chain of length 20.

increase the accuracy of action predictions and according to Fig. 8, this accuracy hovers over 60%. Looking at the confusion matrix (Fig. 9) reveals that the majority of mistakes correspond to the case where the correct action was pushing but the network has predicted a pulling action. It is evident from Fig. 10 that the stick has failed to generate an effect of pushing the objects away, this has caused the network to mistake it many times for pulling as usually it is pulling with a stick which results in no motion. As explained before, the most influential feature to predict a push is movement along the y axis which did not occur during pushing with the novel stick. According to our expectation, the VAE has predicted more forward motion for pushing with the novel stick (see the red dots for pushing with the stick in Fig. 10).

Another interesting observation is that in the novel dataset, we only performed experiments with pushing and drawing, so the true lable cannot be tap from left or right (two first rows of Fig. 9), however, since we have used the exact same network, sometimes we get tapping predictions which are always classified as incorrect.

V. CONCLUSIONS AND FUTURE WORK

This paper proposes a novel probabilistic framework based on VAEs which uses deep neural networks to predict heterogeneous distributions of multi-modal observations. This model was used to endow a robot with the capability to perceive and use objects and tools affordances. In order to cope with different distributions, we have assumed conditional factorizations among disparate modalities. CLR was used to prevent some modalities dominating over other ones by continuously increasing and decreasing the learning rate during training. This flexibility enabled us to achieve *complete self-supervision* as a result of no pre-processing of the data. This property is unique to the presented work.

Actions that were used in this work were categorical and pre-defined. As a result, the robot cannot learn new actions

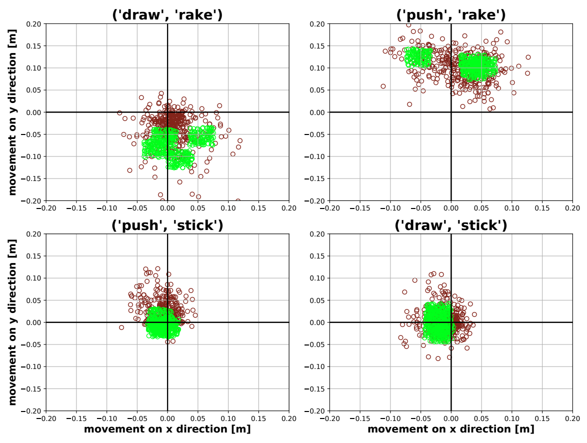


Fig. 10: Distribution of effects, divided by different actions and tools for the novel dataset; similar to Fig. 6, the length of the Markov chain is 150.

by interacting with the environment. A possible extension to this work would be to incorporate parametric actions in the formulation of affordances.

Variational Auto-Encoders were originally tested on a dataset of images. An exciting area of research would be to learn the affordances directly from images through interactions without the need to extract predefined visual features. We hope to facilitate further research in this direction by providing a relatively large dataset of robot's tool and object manipulation trials which can easily be extended by external research groups.

Finally, one can argue that specific forces and tool trajectories with respect to objects are among important variables which ultimately determine the effects of actions. Having a flexible probabilistic framework, it is possible to incorporate these variables for a more complete modeling of affordance effects.

ACKNOWLEDGMENT

This work is partially supported by the Portuguese Government (Fundação para a Ciência e a Tecnologia, doctoral grant PD/BD/105776/2014 Robotics, Brain and Cognition programme PD-FCT and project UID/EEA/50009/2013).

References

[1] G. Pezzulo and C. Castelfranchi, "Thinking as the control of imagination: a conceptual framework for goal-directed systems." *Psychological research*, vol. 73, no. 4, pp. 559–577, Jul 2009.

[2] I. Awaad, G. K. Kraetzschmar, and J. Hertzberg, "Affordance-based reasoning in robot task planning," *Planning and Robotics*, p. 2, 2013.

[3] J. J. Gibson, *The Ecological Approach to Visual Perception*. Boston, MA: Houghton Mifflin, 1979.

[4] E. J. Gibson, "Perceptual learning in development: Some basic concepts," *Ecological Psychology*, vol. 12, no. 4, pp. 295–302, 2000.

[5] L. Montesano, M. Lopes, A. Bernardino, and J. Santos-Victor, "Learning object affordances: From sensory–motor coordination to imitation," *IEEE Transactions on Robotics*, vol. 24, no. 1, pp. 15–26, 2008.

[6] 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," *Adaptive Behavior*, vol. 15, no. 4, pp. 447–472, 2007.

[7] G. Metta, L. Natale, F. Nori, G. Sandini, D. Vernon, L. Fadiga, C. von Hofsten, K. Rosander, M. Lopes, J. Santos-Victor, A. Bernardino, and L. Montesano, "The iCub humanoid robot: An open-systems platform for research in cognitive development," *Neural Networks*, vol. 23, no. 8, pp. 1125–1134, 2010.

[8] D. J. Rezende, S. Mohamed, and D. Wierstra, "Stochastic Backpropagation and Approximate Inference in Deep Generative Models," in *International Conference on Machine Learning*, 2014.

[9] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," in *International Conference on Learning Representations*, 2014.

[10] C. Doersch and A. Zisserman, "Multi-task self-supervised visual learning," *arXiv preprint arXiv:1708.07860*, 2017.

[11] L. Jamone, E. Uğur, A. Cangelosi, L. Fadiga, A. Bernardino, J. Piater, and J. Santos-Victor, "Affordances in psychology, neuroscience and robotics: a survey," *IEEE Transactions on Cognitive and Developmental Systems*, 2016.

[12] H. Min, R. Luo, J. Zhu, S. Bi, *et al.*, "Affordance research in developmental robotics: A survey," *IEEE Transactions on Cognitive and Developmental Systems*, 2016.

[13] J. Sun, J. L. Moore, A. Bobick, and J. M. Rehg, "Learning visual object categories for robot affordance prediction," *The International Journal of Robotics Research*, vol. 29, no. 2-3, pp. 174–197, 2010.

[14] I. Higgins, L. Matthey, X. Glorot, A. Pal, B. Uria, C. Blundell, S. Mohamed, and A. Lerchner, "Early Visual Concept Learning with Unsupervised Deep Learning," *arXiv preprint arXiv:1606.05579*, 2016.

[15] P. Osório, A. Bernardino, R. Martinez-Cantin, and J. Santos-Victor, "Gaussian mixture models for affordance learning using bayesian networks," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2010, pp. 4432–4437.

[16] A. Gonçalves, G. Saponaro, L. Jamone, and A. Bernardino, "Learning Visual Affordances of Objects and Tools through Autonomous Robot Exploration," in *IEEE International Conference on Autonomous Robot Systems and Competitions*, 2014, pp. 128–133.

[17] A. Gonçalves, J. Abrantes, G. Saponaro, L. Jamone, and A. Bernardino, "Learning Intermediate Object Affordances: Towards the Development of a Tool Concept," in *IEEE International Conference on Developmental and Learning and on Epigenetic Robotics*, 2014, pp. 482–488.

[18] G. Saponaro, P. Vicente, A. Dehban, L. Jamone, A. Bernardino, and J. Santos-Victor, "Learning at the ends: From hand to tool affordances in humanoid robots," in *IEEE International Conference on Developmental and Learning and on Epigenetic Robotics*, 2017.

[19] A. Dehban, L. Jamone, A. R. Kampff, and J. Santos-Victor, "Denoising auto-encoders for learning of objects and tools affordances in continuous space," in *IEEE International Conference on Robotics and Automation*, May 2016, pp. 4866–4871.

[20] T. Mar, V. Tikhanoff, and L. Natale, "What can I do with this tool? Self-supervised learning of tool affordances from their 3D geometry," *IEEE Transactions on Cognitive and Developmental Systems*, 2017.

[21] O. Chavez-Garcia, M. Andries, P. Luce-Vayrac, and R. Chatila, "Discovering and manipulating affordances," in *International Symposium on Experimental Robotics (ISER)*, 2016.

[22] J. Sung, J. K. Salisbury, and A. Saxena, "Learning to represent haptic feedback for partially-observable tasks," in *IEEE International Conference on Robotics and Automation*, May 2017, pp. 2802–2809.

[23] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational inference: A review for statisticians," *Journal of the American Statistical Association*, vol. 112, no. 518, pp. 859–877, 2017.

[24] C. K. Sønderby, T. Raiko, L. Maaløe, S. K. Sønderby, and O. Winther, "Ladder variational autoencoders," in *Advances in Neural Information Processing Systems*, 2016, pp. 3738–3746.

[25] A. Dehban, L. Jamone, A. R. Kampff, and J. Santos-Victor, "A Moderately Large Size Dataset to Learn Visual Affordances of Objects and Tools Using iCub Humanoid Robot," in *ECCV Workshop on Action and Anticipation for Visual Learning*, 2016.

[26] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, "Multimodal deep learning," in *International Conference on Machine Learning*, 2011, pp. 689–696.

[27] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *International Conference on Learning Representations*, 2015.

[28] L. N. Smith, "Cyclical learning rates for training neural networks," in *IEEE Winter Conference on Applications of Computer Vision*, 2017, pp. 464–472.