Spatial-Temporal Online Learning with Sliding Window in Gaussian Processes for Tactile Object Classification

Tasbolat Taunyazov*
School of Computing
National University of Singapore
Singapore, 21 Lower Kent Ridge Rd
e0348851@u.nus.edu
A0192157B

Abstract

In this work, we propose fast and simply interpretable online learning with Gaussian Processes for tactile object classification for 9 objects. The primary concern of the proposed work is ability of the robot to learn the environment incrementally by touch. Gaussian Process with different kernels with fixed window size is computed at each iteration and used to predict the next sample by discriminative model. It achieves accuracy of 98% on tactile dataset, which is comparable with state of the art results.

1 Introduction

The main advantage of the robots are ability to operate continuously over long time. That's why it's essential for the robot to learn the environment incrementally as time passes. This is similar as small kid learns about matters time by time, either with touch, vision or listening. In this project we concentrate on online learning by touch through tactile sensor. Tactile sensors measures information which comes with interaction with environment such as touch and slide.

In this paper, we address issue of online learning with incoming multi-dimensional tactile feedback data. Tactile data are represented using temporal signal with spatial correlation. The aim of the online learning is to understand the objects in the environment better over time. The main issue on analyzing tactile data is that it's troublesome to get enough tactile data from operating robot. Thus, we are limited to small amount of data and state of the art deep learning techniques cannot be used in our case.

In order to address online learning through tactile signal, we propose Gaussian Process (GP) with two different kernels. Automatic Relevance Kernel (ARD) is used for better representation of the spatial relationship of the data, while recursive kernel handles temporal structure of data. Advantages of the GP over other machine learning methods are that it can predict not only actual estimate but also confidence level for the specific data point. This is crucial, since we can monitor how the online learning is behaving over time.

The major contributions of the papers are as following:

1. Object classification through spatial-temporal tactile data is feasible with high accuracy in online learning.

*The author is affiliated with Infocomm Research Department of A-STAR.

33rd Conference on Neural Information Processing Systems (NeurIPS 2019), Vancouver, Canada.

2. Simple and fast sliding window temporal kernel GP (SWT-GP) framework with ARD and recursive kernels integrated into *sci-kit* package. It can be used for other tasks too.

The paper is structured as following: the section explains the motivation behind SWT-GP in terms of state of the art methods. Then, the Gaussian Process and different kernels are explained with mathematical background. It follows with implementation details on tactile dataset with proposed algorithm. Finally, we discuss the results and conclude the paper.

2 Related Work

This work is based on *Spatio-Temporal Online Recursive Kernel Gaussian Process* (STORK-GP) proposed by Soh et al. [1] and *Online Infinite Echo State Gaussian Process* (OIES-GP) Soh and Demiris [2]. In former case, authors use sparse gaussian process with spatial-temporal recursive kernel. Even though sparsity helps with generalization of the hypothesis and promises fast learning, in real experiment it doesn't perform that fast (around 80 ms for given object classification task). We expect the learner to be fast, since robot shall continue to operate to learn more over time and have smooth movement. In the latter authors utilize Echo State Network over Gaussian Process to handle temporal structure. The accuracy score of OIES-GP is higher than STORK-GP. It also learns faster, since echo state can control *capacity* (a.k.a depth of the network). However, implementation of Infinite Echo State Network is more complicated and cannot be easily interpreted. Taking into account all advantages of STORK-GP and OIES-GP, our model, SWT-GP, utilizes both and simplifies to be interpretable. The proposed method is comparable with state of the art Online Learning Gaussian Processes.

3 Gaussian Processes

Gaussian Process (GP) is defined as a collection of random variables, any finite number of which have a joint Gaussian distribution (Matthews et al. [3]). Given the training samples $\mathbf{x} \in \mathcal{X}$, the mean and covariance functions of the Gaussian distribution is defined as (Rasmussen [4]):

$$m = \mathbb{E}[f(\mathbf{x})] \tag{1}$$

$$k(\mathbf{x_i}, \mathbf{x_j}) = \mathbb{E}[(f(\mathbf{x_i}) - m(\mathbf{x_i}))(f(\mathbf{x_j}) - m(\mathbf{x_j}))] \tag{2}$$

where $\mathbf{x_i}, \mathbf{x_j} \in \mathcal{X}$. Most of the cases, we assume that function $f(\cdot)$ is sampled from zero mean Gaussian distribution (Lugosi et al. [5]):

$$f(\mathbf{x}) \sim \mathcal{N}(0, k(\mathbf{x_i}, \mathbf{x_j})) \tag{3}$$

which leads to have $m(\mathbf{x}) = 0$. Furthermore we assume that samples are drawn from independent and identically distributed (i.i.d.) from a distribution $\mathcal{D}$, $(\mathbf{x_i}, y_i) \sim \mathcal{D}$. y_i is a corresponding target value for given observations $\mathbf{x_i}$. Covariance matrix, a.k.a Gram matrix K, is defined between N observed points and defined as following (de Mello and Ponti [6]):

$$K = \begin{bmatrix} k(\mathbf{x_1}, \mathbf{x_1}) & k(\mathbf{x_1}, \mathbf{x_2}) & \dots & k(\mathbf{x_1}, \mathbf{x_N}) \\ k(\mathbf{x_2}, \mathbf{x_1}) & k(\mathbf{x_2}, \mathbf{x_2}) & \dots & k(\mathbf{x_2}, \mathbf{x_N}) \\ \dots & \dots & \dots & \dots \\ k(\mathbf{x_N}, \mathbf{x_1}) & k(\mathbf{x_N}, \mathbf{x_2}) & \dots & k(\mathbf{x_N}, \mathbf{x_N}) \end{bmatrix} \tag{4}$$

K is known to be symmetric and positive semi-definite with diagonal values as $K(\mathbf{x_i}, \mathbf{x_i}) = 1$ for $\forall i$. Let's define $\mathbf{k}(\mathbf{x_i})$ as a row corresponding to $\mathbf{x_i}$. Hence, given new test point $\mathbf{x}_*$,, GP predicts a distribution as:

$$p(f_*|\mathbf{x}_*, \mathcal{D}) = \mathcal{N}(f_*|\mu_*, \sigma_*^2) \tag{5}$$

where

$$\mu_* = \mathbf{k}(\mathbf{x}_*)^T (K + \sigma^2 I_N)^{-1} \mathbf{y} \tag{6}$$

$$\sigma_*^2 = \mathbf{k}(\mathbf{x}_*, \mathbf{x}_*) - \mathbf{k}(\mathbf{x}_*)^T (K + \sigma^2 I_N)^{-1} \mathbf{k}(\mathbf{x}_*) \tag{7}$$

The difference from most of the classification algorithms existing in Machine Learning, GP predicts a distribution over test point. This can be helpful to design stochastic algorithms (Lawler [7].

4 Kernels

Kernel matrix, a.k.a Gram matrix K, depends on kernel function which defines similarity between two samples in the training set. In this project, we use squared exponential kernel, or radial basis function kernel (RBF), defined as:

$$k_{\mathrm{E}}(\mathbf{x_i}, \mathbf{x_j}) = \exp\Big(- \frac{||\mathbf{x_i} - \mathbf{x_j}||^2}{2l^2} \Big) \tag{8}$$

where l is a length scale or hyperparameter of RBF model. This kernel is widely used in many machine learning techniques including Support Vector Machines (SVM) Jiang et al. [8]. The gradient of kernel is simply calculated as following:

$$\frac{\partial k_{\mathrm{E}}(\mathbf{x_i}, \mathbf{x_j})}{\partial l} = \frac{||\mathbf{x_i} - \mathbf{x_j}||^2}{l^3} \exp\Big(- \frac{||\mathbf{x_i} - \mathbf{x_j}||^2}{2l^2} \Big) \tag{9}$$

4.1 ARD Kernels

In order to capture spatial data, we would like to use Automatic Relevance Detection (ARD) Kernel (Zhao et al. [9]):

$$k_{\mathrm{EA}}(\mathbf{x_i}, \mathbf{x_j}) = \exp\Big(- \sum_{k=1}^{D} \frac{||\mathbf{x_i} - \mathbf{x_j}||^2}{2l_k^2} \Big) \tag{10}$$

where any $\mathbf{x}$ is D dimensional feature vector. Only corresponding length scale l_k to the dimension k is $l_k \geq 0$, while in all other cases its value is 0. Note that, the optimization of hyperparameters for ARD kernel is much more expensive than RBF kernel. This should be used with caution. For optimization purposes, we calculate the gradient of the ARD kernel is calculated as:

$$\frac{\partial k_{\mathrm{REA}}(\mathbf{x_i}, \mathbf{x_j})}{\partial l_k} = \frac{||\mathbf{x_i} - \mathbf{x_j}||^2}{l_k^3} \exp\Big(- \sum_{k=1}^{D} \frac{||\mathbf{x_i} - \mathbf{x_j}||^2}{2l_k^2} \Big) \tag{11}$$

4.2 Recursive Kernels

First proposed to be used in Bayesian Network, recursive kernels are well-known in measure of temporal similarity of data as shown in Neal [10].

$$k_{\mathrm{REA}}^{(t)}(\mathbf{x_i}, \mathbf{x_j}) = \exp\Big(- \sum_{k=1}^{D} \frac{||\mathbf{x_i} - \mathbf{x_j}||^2}{2l_k^2} \Big) \exp\Big(- \frac{k_{\mathrm{REA}}^{(t-1)}(\mathbf{x_i}, \mathbf{x_j}) - 1}{2\rho^2} \Big) \tag{12}$$

where ρ is referred as temporal relevance and second exponential term is used similar to hidden state in Recurrent Neural Networks (RNN) as proposed in Lin et al. [11].

$$\frac{\partial k_{\mathrm{REA}}^{(t)}(\mathbf{x_i}, \mathbf{x_j})}{\partial l_k} = k_{\mathrm{REA}}^{(t)} \Big(\frac{||\mathbf{x_i} - \mathbf{x_j}||^2}{l_k^3} + \frac{1}{\rho^2} \frac{\partial k_{\mathrm{REA}}^{(t-1)}}{\partial l_k} \Big) \tag{13}$$

Again, If used with ARD, the recursive kernel has many hyperparameters to be learnt. However, we can easily calculate the gradient of the recursive kernel to speed up the optimization problem.

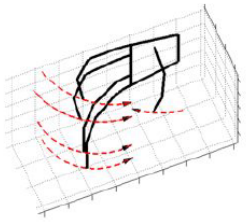
$$\frac{\partial k_{\mathrm{REA}}^{(t)}(\mathbf{x_i}, \mathbf{x_j})}{\partial \rho} = k_{\mathrm{REA}}^{(t)} \Big(\frac{-2(k_{\mathrm{REA}}^{(t-1)} - 1)}{\rho^3} + \frac{1}{\rho^2} \frac{\partial k_{\mathrm{REA}}^{(t-1)}}{\partial \rho} \Big) \tag{14}$$

Notice that the gradient is also recursive in its nature.

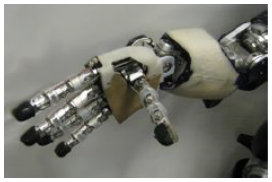
5 Implementation

In this paper, we implement GP process for object classification by tactile data. The tactile dataset is given by Soh et al. [1] and contains 9 different objects as shown in figure 1a. In this dataset, humanoid iCub robot (Metta et al. [12]) grasps with objects 20 times (trials). First, its hand poses in pre-grasping position as shown in figure 1c. Then, it fingers slowly occupies the object with specific trajectory as given in figure 1b. Every trial is spatial-temporal data sensed tactile sensor collected at

(a) 9 objects to be grasped by iCub robot: full and empty coke cans, full, half full and empty bottles, book, lotion, monkey toy and Teddy-bear.



(b) Robot hand behaviour to grasp an object.



(c) Initial robot hand pose for grasping.

Figure 1: Tactile dataset for object classification.

50 Hz. In this project, we aim to classify those objects online, meaning our algorithm incrementally learns object properties by touch.

Unlike many other GP implementations, we adopt sliding window based GP (SWT-GP) training to keep the training time as fast as possible. Window size, τ, determines how deep the GP can look to be able to predict the next incoming data. Any data, beyond τ is disregarded. The predicted output of GP is a function described as in equation 5. In this GP classification, we choose binary classification with targets being $\{\pm 1\}$. The One-vs-Rest algorithm used to handle multi-class classification task. In general, GP is not designed for online learning. For this reason, we propose an algorithm which fits to our purpose. In this algorithm, we set kernel value calculated at previous state to current state for recursive kernels (algorithm 1). This have no effect for standard non-recursive RBF and ARD kernels. In this way, it's possible to monitor kernel values and make diagnostics on hidden state of the kernel.

Algorithm 1: Recursive Update of the GP

input : $X \in \{\mathbf{x_1}, ..., \mathbf{x_T}\}$, $Y \in \{y_1, ..., y_T\}$, τ, K_{t-1}, $\dot{K}_{t-1}$
output : $\bar{y}_t$

initialization;
for $t \leftarrow \tau$ **to** T **do**
  $\tilde{X} \leftarrow X_{t:(t+\tau)}$;
  $\tilde{Y} \leftarrow Y_{t:(t+\tau)}$;
  `// update GP with previous kernel value and its gradient`
  $K_{t-1}, \dot{K}_{t-1} \rightarrow$ to GP;
  `// maximize marginal log likelihood, fit GP`
  $K_t, \dot{K}_t = \text{GP}(\tilde{X}, \tilde{Y})$;
  `// predict next test point`
  $\bar{y}_t = \text{GP}(X_{t+1})$
end

In order to see all capability of GPs with different configurations, we implement 4 different scenarios. These scenarios are described below.

1. *Scenario 1.* Standard full GP with squared exponential kernel (RBF) given in equation 8 is used to fit the tactile data. Hyperparameter (length scale) for this scenario is l, which is found by maximizing pseudo-maximum likelihood.

2. *Scenario 2.* Automatic Relevance Kernel (ARD) is used in RBF form as given in equation 10. ARD helps to impose spatial correlation on tactile data as it handles each dimension separately. In this case, we have D hyperparameters (length scale): $l_1, l_2, ..., l_D$.

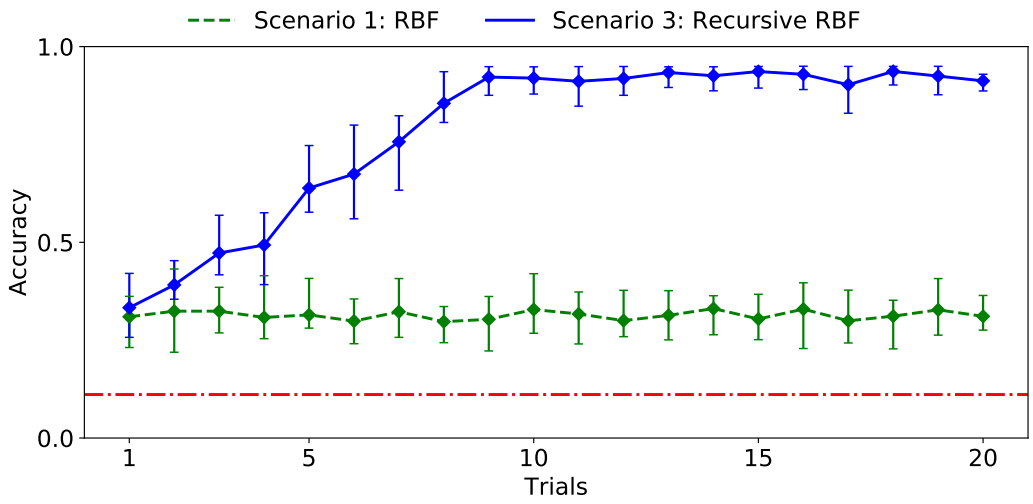


Figure 2: Accuracy score for Isotropic kernels in SWT-GP.

3. *Scenario 3.* In this scenario, we use recursive kernel given in equation 12 in full GP with two hyperparameter l (scale lenght) and ρ (hidden state). Recursive kernel memorizes information through hidden state to teh next window. In this way, one can learn the temporal information given in data.
4. *Scenario 4.* The last case includes both spatial and temporal scenarios. We combine recursive kernel with ARD as given in equation 12. However, the number of hyperparameters rises to $D + 1$.

Scenario 1 and 3 are referred as *isotropic kernels* as it doesn't handle spatial structure of the data. Scenario 2 and 4 are refered as *anisotropic kernels*. Line 5 in algorithm 1 has no effect in scenario 1 and 2, since hidden state is not stored and passed. We implement GP online learning in python using scipy and numpy libraries. Implementation of recursive RBF and ARD in scipy and numpy can be found in this link: `https://github.com/tasbolat1/GaussianProcess`.

6 Results and Discussions

In this section we discuss results of online learning for 4 scenarios. First, we measure accuracy score for each trial where it's simply total number of correct classified samples divided by total samples. Second, we analyse running time of the algorithm 1.

6.1 Accuracy on Isotropic Kernel

Figure 2 demonstrates accuracy score for online GP learning with RBF and Recursive kernels. At each trial, accuracy measure is given as probability density function with confidence interval on measured accuracy set on that interval. In the figure, we can see that GP with standard RBF kernel doesn't improve over each trial. Since we are not passing history data, this result is very well expected. However, GP with recursive RBF converges to 96% in 9-th trial. We can also notice that in recursive kernel case, accuracy steadily rises. This means that kernel successfully transfers history information to the current state.

6.2 Accuracy on Anisotropic Kernel

The accuracy score for online GP learning with ARD and Recursive ARD kernels are given in figure 3. Results are similar to isotropic cases, except in this case GP with recursive kernel converges faster (around 7-th trial). In addition, accuracy increases to 98%. This shows that recursive recursive ARD can take care of spatial and temporal structure much better than other cases.

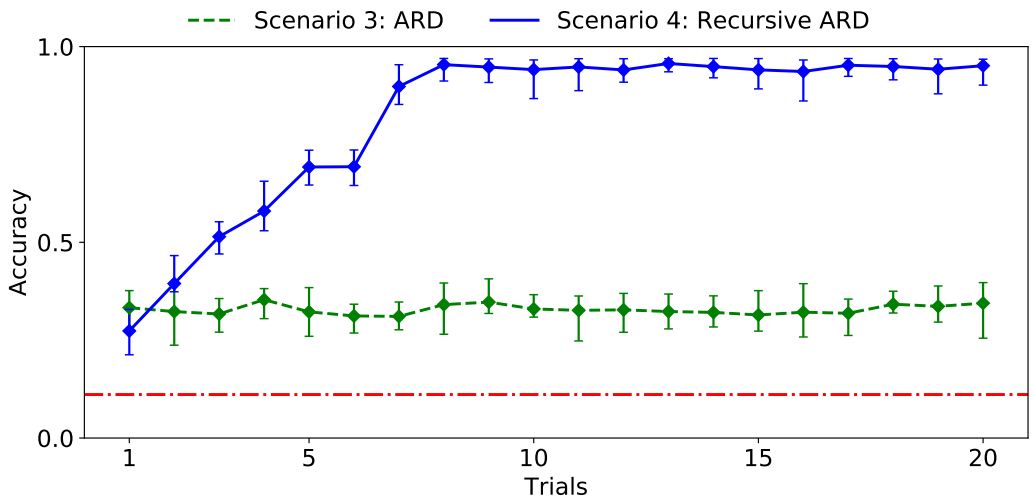


Figure 3: Accuracy score for Anisotropic kernels in SWT-GP.

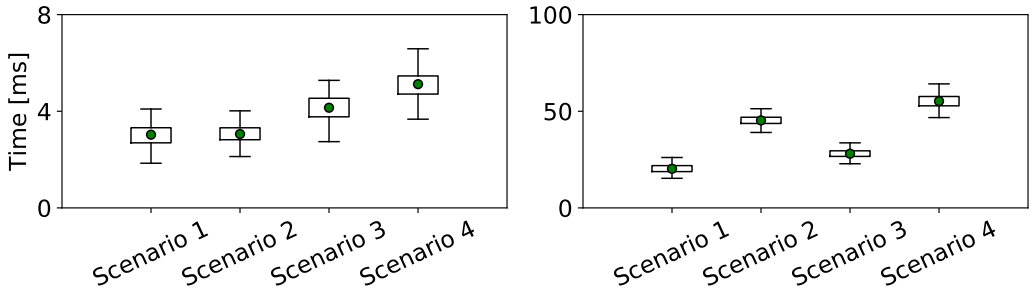


Figure 4: Execution time for predict (left) and update (right) parts of algorithm 1.

We also note that non-recursive kernels perform worse than recursive kernel in online learning, however GP non-recursive kernel gives higher accuracy than random chance, which is around 8.1%. Selecting appropriate depth value τ we can have better results for non-recursive case. However, the speed of update of GP will be much more slower.

6.3 Running speed

Algorithm 1 consists of two main parts which are computationally expensive to estimate. These are update (line 4, 5) and prediction (line 6) parts given in algorithm 1. In both cases, we shall be fast enough so that the robot will move smoothly.

Figure 4 demonstrates time computation in two cases. All of the scenarios perform predict part within 7 ms. The GP with recursive ARD takes more time than others. However, the differences are not that big. It's a bit surprising that computation for ARD kernel takes as much as time for RBF kernel, since the number of parameters in latter case is much lower than former one.

Unlike predict case, the difference in update part is very clear. Since computation of maximization log-likelihood in GP with ARD kernel requires more time due to large number of hyperparameters, in both scenarios 2 and 3 we can notice that computation takes around 50 ms. In addition, in recursive kernel seems slightly slower. In isotropic sceanarios, the computation time doesn't exceed 30 ms, which is more than 50 Hz.

Finally, we compare proposed SWT-GP with state of the art online GP learning frameworks such as STORK-GP and OIES-GP as shown in table 1. Due to the simple structure, SWT-GP cannot reach accuracy level of other methods, however it's very close enough. GP with recursive kernels

Table 1: Comparasion of SWT-GP with other Online GP algorithms

Algorithm	Accuracy	Update time, ms	Convergence, trial
SWT-GP RBF	0.332	**19**	-
ARD	0.354	52	-
Recursive RBF	0.961	26	9
Recursive ARD	0.982	57	7
STORK-GP	0.997	40	**4**
OIES-GP	**0.999**	20	**4**

in SWT-GP cannot update within 50 Hz, mainly it's naively implemented in python, while the rest algorithms are implemented in C++. We can also increase convergence to the highest accuracy score faster selection larger window size. However, it will add compuational cost and can be done if implementation is revisited. In general, as preliminary work, the SWT-GP with recursive kernels are promising in terms of accuracy and learning time.

7 Conclusion

In this paper, we proposed Sliding Window Online Learning with Gaussian Process (SWT-GP) for object classification with tactile data. Even it's not comparable with state of the art methods, SWT-GP provides very high accuracy and fast learning of the hyperparameters. In the future work, SWT-GP shall be re-written with more sophisticated way in C or C++ to have faster computation. In addition the effect of τ (window size) must be analyzed in detail. The presentation for this paper can be found in `https://github.com/tasbolat1/GaussianProcess/tree/master/presentation`.

References

[1] Harold Soh, Yanyu Su, and Yiannis Demiris. Online spatio-temporal gaussian process experts with application to tactile classification. In *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 4489–4496. IEEE, 2012.

[2] Harold Soh and Yiannis Demiris. Incrementally learning objects by touch: Online discriminative and generative models for tactile-based recognition. *IEEE transactions on haptics*, 7(4):512–525, 2014.

[3] Alexander G de G Matthews, Mark Rowland, Jiri Hron, Richard E Turner, and Zoubin Ghahramani. Gaussian process behaviour in wide deep neural networks. *arXiv preprint arXiv:1804.11271*, 2018.

[4] Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer School on Machine Learning*, pages 63–71. Springer, 2003.

[5] Gábor Lugosi, Shahar Mendelson, et al. Sub-gaussian estimators of the mean of a random vector. *The Annals of Statistics*, 47(2):783–794, 2019.

[6] Rodrigo Fernandes de Mello and Moacir Antonelli Ponti. A brief introduction on kernels. In *Machine Learning*, pages 325–362. Springer, 2018.

[7] Gregory F Lawler. *Introduction to stochastic processes*. Chapman and Hall/CRC, 2018.

[8] Shihao Jiang, Richard Hartley, and Basura Fernando. Kernel support vector machines and convolutional neural networks. In *2018 Digital Image Computing: Techniques and Applications (DICTA)*, pages 1–7. IEEE, 2018.

[9] Jun Zhao, Long Chen, Witold Pedrycz, and Wei Wang. Variational inference-based automatic relevance determination kernel for embedded feature selection of noisy industrial data. *IEEE Transactions on Industrial Electronics*, 66(1):416–428, 2019.

[10] Radford M Neal. Priors for infinite networks. In *Bayesian Learning for Neural Networks*, pages 29–53. Springer, 1996.

[11] Tsungnan Lin, Bill G Horne, Peter Tino, and C Lee Giles. Learning long-term dependencies in narx recurrent neural networks. *IEEE Transactions on Neural Networks*, 7(6):1329–1338, 1996.

[12] Giorgio Metta, Giulio Sandini, David Vernon, Lorenzo Natale, and Francesco Nori. The icub humanoid robot: an open platform for research in embodied cognition. In *Proceedings of the 8th workshop on performance metrics for intelligent systems*, pages 50–56. ACM, 2008.