On Choosing Structure for a Machine Learning-based Reaction Force Predictor for Walking Robots

E Zalyaev\textsuperscript{1}, S Savin\textsuperscript{1}, A Salikhzyanov\textsuperscript{2} and S Golousov\textsuperscript{3}

\textsuperscript{1} Center for Technologies in Robotics and Mechatronics Components, Innopolis University, Innopolis, 420500, Russia
\textsuperscript{2} Edge Vision Ltd., Innopolis, 420500, Russia
\textsuperscript{3} Technology Arrival Ltd, Banbury, London, UK

E-mail: s.savin@innopolis.ru

Abstract. This paper focuses on the topic of contact reaction prediction for walking robots, namely on the analysis of performances on different structures of the machine-learning-based predictors. Predicting reaction forces is important due to the fact that it can allow us to retrieve a simplified model of the contact scenario, as was proposed in the literature previously. This allows to turn the problem of contact model identification into a data collection and processing problem. In order to do it effectively, both a data compression (feature extraction) and regression strategies might be used. This research provides an analysis of both with respect to the discussed problem.

1. Introduction

Among the variety of walking robots, bipedal robots (and anthropomorphic robots in particular) present a special interest from both the theoretical and practical perspectives. The later is due to their potential flexibility in the kind of tasks they can perform, which is inherited from their biological prototypes. They should be able to function for work in industrial [1], office, home, and natural environments, and they could be used in exploration missions [2], improvised explosive device response and disaster response [3]. This, however, requires building a control system that take advantage of their potential flexibility.

Control of bipedal robots is a challenging task for a number of reasons. One of them is the lack of exact dynamic model for the robot in all its operation modes. While dynamics of the mechanism itself can be modelled precisely [4, 5] and the parameters of the model can be identified [6], the contact model in general cannot be inferred directly from a typical sensor suit of a bipedal robot [2, 7].

One of the typical approaches to dealing with contact interaction in bipedal robotics is to make assumptions about the environment. For example, it is often assumed that the supporting surface, on which the motion takes place, is horizontal and flat [8, 9]. With additional assumptions about the rate of change of the angular momentum and about the constancy of the height of the center of mass of the robot, a zero-moment point (ZMP)-based approach to trajectory generation can be used [10–12]. This approach can be extended to uneven terrain, as was shown in [13], with additional limitations, producing conservative trajectories. However, modern trajectory planning methods for bipedal robots, such as contact wrench cones (CWC) seek to avoid those assumptions [14, 15].
Some of the methods in control pipeline for bipedal robots, such as foothold planning, allow taking into account uneven terrain, obtaining the actual information about the geometry of the supporting surface directly from RGB-D cameras or LIDAR systems [2, 16]. Such methods can be based on mixed-integer programming, as was shown in [17–19]. However, such approaches require that lower level control system is able to execute the found foothold plan; that sets a limit on how dynamic and aggressive the foothold plan can be. In practice, estimating the robustness of the lower-level control system is a non-trivial task; during DARPA Robotics Challenge, some of the teams used manual choice of motion plans in order to ensure safety of the operation [20,21].

Here, we address one of the issues directly related to the design and stability of the low-level control system for robots with contact interaction with terrain, when the contact model can’t be easily obtained. One way to deal with such situation was proposed in [22, 23], where a dense neural network was used to predict ground reaction forces in contact points between the robot and the environment. The motivation of the method is as follows. If the robot has force sensors, they can be used to collect real-world data which accurately (up to the sensor’s precision) represents the contact model. Such data can be collected for all contact modes of the robot (robot contacts the ground only with left or right leg, with both legs, only with hill of one leg, etc.). Then the data, together with the state of the robot and proprioceptive sensors, can be used to train a predictor, which can be based on a neural network. The predictor can then be used as a supplement for the correct contact model in a model-based control framework.

In the paper [22], the predictor is based on a network with 5 fully-connected layers with ReLU (rectified linear unit) activations. This can be seen as one of the simplest predictor models. However, this model has more parameters than a lot of the other machine learning models suitable for the regression task. Other structures include the use of support vector machines and others to reduce dimension of the parameter space. Those can be used to improve robustness of the predictor, make it less computationally expensive and improve the training process. In this paper, we present a comprehensive analysis of the predictor structures, based on main machine learning models supporting regression tasks. In particular, we studied the use of auto-encoders, non-negative matrix factorization and principle component analysis for data compression, and support vector regression, linear regression and dense neural networks (DNN) for the regression on the compressed data. Our main contribution is the demonstration of the advantage of the use of auto-encoders in conjunction with DNN or with linear regression.

The rest of the paper is organized as follows. Section II discusses mathematical models of the robot and the contact. Section III gives a description to the data compression strategies used in this work, and gives an analysis of the effects of using different autoencoder designs. Section IV discusses regression models and approaches used here and gives an analysis of the comparative performances of these approaches.

2. Robot and Contact Interaction Models

In general, dynamics of a robot with mechanical constraints can be written as follows:

\[
\begin{align*}
\{ H\ddot{q} + c &= Bu + F^T\lambda \\
F\dot{q} + \dot{F}\dot{q} &= 0,
\end{align*}
\]

where \( q \) is a vector of generalized coordinates (joint space coordinates and floating base coordinates \([24]\)), \( u \) is the vector of motor torques (control inputs), \( \lambda \) is the vector of Lagrange multipliers (reaction forces related to the constraints), \( H \) is generalized inertia matrix, \( c \) is a dynamics bias vector, \( B \) is the matrix that describes the actuator placement, and \( F \) is the jacobian matrix of the constraints.

If jacobian matrix \( F \) is known exactly and (1) always holds, both the generalized accelerations \( \ddot{q} \) and reaction forces \( \lambda \) can be calculated analytically. However, actual contact interactions are
often subject to friction cone constraints, the mechanical constraints can be overdetermined, and jacobian matrix $F$ can be unknown. Then, neither $\ddot{q}$ nor $\lambda$ can be calculated, which makes it difficult to design control schemes.

In [**] it was proposed to collect data describing the current state of the robot, augmented with control actions $[q, \dot{q}, u]$ and current values of reaction forces $\lambda$, and use it to train a predictor $\bar{\lambda} = \lambda(q, \dot{q}, u)$. In simulation, the data can be collected by solving (1). In physical experiments, this can be achieved with force sensors.

Particular robot, studied in [**] was a 7-link planar bipedal robot. It is described by 9 generalized coordinates $q$ and 9 generalized velocities $\dot{q}$, has 6 control inputs $u$. We consider the case when 3 mechanical constraints $\lambda$ are applied, whose reaction forces are subject to friction cone constraints.

3. Data compression strategies

In input data for the predictor includes generalized coordinates, generalized velocities and the control actions. For the simplicity, we will introduce the input vector $s$ as follows:

$$s = [q, \dot{q}, u]^T$$

The training dataset includes $N$ rows $s^T$, and corresponding rows $\lambda^T$. We denote the input (attributes) dataset matrix with rows $s^T$ as $S$, and output (labels) dataset matrix with rows $\lambda^T$ as $\Lambda$.

Data compression is an important part of learning, as it allows discovering relevant representations without analysing the output, while usually being a robust and reliable procedure. Overall, all the data compression strategies discussed here can be seen as a task to find function $g(s) : \mathbb{R}^n \rightarrow \mathbb{R}^k$, where $n$ is the dimensions of the input vector $s$ and the $k$ is the dimensions of the compressed input $z \in \mathbb{R}^k$:

$$z = g(s) \quad (2)$$

In this section we briefly describe the data compression strategies that were studied in this paper.

3.1. Non-negative matrix factorization

One of the well known strategies for data compression is the non-negative matrix factorization. Re-centering the training data $S$ we get a new attributes dataset $S^* > 0$, with the same corresponding labels $\Lambda$: here and for the rest of this subsection, the inequality is taken as an element-wise operation. Then we can perform a decomposition, finding such $W > 0$ and $V > 0$, that:

$$S^* = WV \quad (3)$$

where $W \in \mathbb{R}^{N \times k}$ and $V \in \mathbb{R}^{k \times n}$.

With (3), we get the desired compressed dataset (2) as follows:

$$Z = W \quad (4)$$

where $Z$ is the new compressed dataset, in which $z$ from (2) represent rows.

3.2. Principle component analysis (PCA)-based factorization

The original training data $S$ can be decomposed with the following factorization:

$$S = U\Sigma V^T \quad (5)$$
Table 1. Parameters of autoencoders and corresponding scores

| #  | Number of elements in each hidden layer | Score  |
|----|----------------------------------------|--------|
| 1  | 40, 37, 35                             | 0.041  |
| 2  | 40, 39, 37, 35                         | 0.036  |
| 3  | 40, 30, 25                             | 0.037  |
| 4  | 40, 39                                 | 0.033  |
| 5  | 29, 14                                 | 0.075  |
| 6  | 24, 9                                  | 0.153  |
| 7  | 39                                     | 0.037  |
| 8  | 19                                     | 0.074  |
| 9  | 9                                      | 0.208  |

where \( U \) and \( V \) are matrices of the left and right singular vectors and \( \Sigma \) is a diagonal matrix of singular values. Discarding all singular values smaller than a given threshold, we obtain a new matrix of singular values \( \Sigma^* \) and new corresponding matrices of singular vectors \( U^* \) and \( V^* \). Corresponding compression algorithm is given below:

\[
Z = U^* \Sigma^* = SV^*,
\]  

where \( Z \) is the compressed dataset, as previously.

3.3. Autoencoders
In machine learning, there is a specific design of neural networks for data compression called autoencoders. The basis of the concept is to teach the network to predict the exact same value it is given as an input, while having less variables to store the information than there are input channels. If the input is \( n \)-dimensional, the output of an autoencoder is also \( n \)-dimensional, but one of the layers is only \( k \)-dimensional, where \( k < n \).

After training the network on the input dataset only, we can use the layers up to the \( k \)-dimensional layer to compress the data, producing the sought function \( z = g(s) \).

Autoencoders are a family of neural networks and the choice of a particular structure plays a role in the quality of the result. Here we considered a family of autoencoders. The varied parameters are the number of layers and the number of elements in each layer. The activation functions were maintained through all experiments (hyperbolic tangent was used as an activation). Table 1 provides the scores of different structures studied here. Due to insignificance of the error differences between two and three layer autoencoders we used two layer architecture for testing.

It is important to note that the scores cited in the Table 1 do not represent the fitness of the resulting predictor, but only the quality of the encoder itself.

4. Regression strategies
Contact reaction prediction constitutes a regression task. There are a number of machine learning tools designed for that type of problems. In this work, three regression strategies were considered: linear regression (LR), support vector regression (SVR) and dense (fully connected) neural networks (DNN). It is important to note that support vector regression depends on the choice of the kernel function, and DNN depend on activation function, number of layers and number of elements in each layer.
Table 2. Regression analysis comparison

| Feature Extraction | Regressor     | MSE        | $R^2$     |
|-------------------|--------------|------------|-----------|
|                   | baseline     | 2.7e-05    | 0.9987    |
| PCA               | baseline     | 2.8e-04    | 0.9867    |
| NMF               | baseline     | 5.9e-04    | 0.7715    |
|                   | SVR          | 1.0e-05    | 0.9868    |
| PCA               | SVR          | 9.5e-06    | 0.9880    |
|                   | LR           | 5.2e-07    | 0.9994    |
| PCA               | LR           | 2.5e-01    | negative  |
|                   | 1-layer NN   | 7.4e-07    | 0.9995    |
|                   | 2-layer NN   | 7.4e-07    | 0.9993    |
| AE(20, 18)        | 1-layer NN   | 5.6e-07    | 0.9995    |
| AE(20, 18)        | 2-layer NN   | 1.1e-06    | 0.9994    |
| AE(20, 18)        | LR           | 7.0e-05    | 0.9808    |
| AE(30, 36)        | 1-layer NN   | 6.9e-07    | 0.9992    |
| AE(30, 36)        | 2-layer NN   | 7.0e-07    | 0.9996    |
| AE(30, 36)        | LR           | 3.6e-06    | 0.9971    |

Since there are many possible combinations of types of SVR kernels and parameters of those kernels, a cross validation-based optimal choice was made. For each kernel parameter range were discretized, the learning performed for all combinations of parameters and best choice was made based on the verification dataset.

Mean squared error function was chosen to be objective function during training. Coefficient of determination ($R^2$) was chosen as an accuracy metric. It can be formulated as follows:

$$S_{reg} = \sum_i (y_i - f_i)^2$$

$$S_{tot} = \sum_i (y_i - \bar{y})^2$$

$$R^2 = 1 - \frac{S_{reg}}{S_{tot}}$$

where $y_i$ is the $i$th target vector, $f_i$ is corresponding prediction and $\bar{y}$ is the mean of observed data: $\bar{y} = \frac{1}{N} \sum_{i=1}^{N} y_i$, where $N$ is the number of samples.

5. Results

Table 2 contains data collected over experiments run on different configuration of the proposed regression model.

As the experiments have shown, linear models successfully predict the samples that follow the dynamics model. However, the real world applications of these methods suggest that the inputs will include noise and statistical outliers. Increase in data complexity or addition of the noise will hinder the performance of Linear Regression significantly due to its tendency to overfit the data. On the other hand, both undercomplete and overcomplete autoencoders introduced nonlinear latent spaces that reduce the accuracy of the linear methods but at the same time helped neural networks to achieve better results. Overall, neural networks coupled
with autoencoders should be considered as a primary method for the regression task due to their complexity and noise robustness which yet has to be tested in further research.

6. Conclusions
In thus paper the problem of simultaneous choice of a data compression technique and a regression model for a contact model parameter estimation for a walking robot was tackled. The diversity of the available approaches present a challenge on their own, as it was outlined in the work. It was shown that autoencoders are a viable option for data compression, however similar results can be achieved without the use of data compression in the absence of noise. Neural networks showed a better performance over all, compared with support vector machines and linear regression models.

Acknowledgments
The reported study was funded by RFBR, project number 20-38-70155. Software for the machine learning experiments used here was developed with the support of the Presidential grant MK-1537.2019.8

References
[1] Kheddar A, Caron S, Gergondet P, Comport A, Tanguy A, Ott C, Henze B, Mesesan G, Englsberger J, Roa M A et al. 2019 IEEE Robotics & Automation Magazine 26 30–45
[2] Wang M, Wonsick M, Long X and Padr T 2020 2020 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM) (IEEE) pp 765–770
[3] Krotkov E, Hackett D, Jackel L, Perschbacher M, Pippine J, Strauss J, Pratt G and Orlowski C 2017 Journal of Field Robotics 34 229–240
[4] Westervelt E R, Grizzle J W, Chevallereau C, Choi J H and Morris B 2018 Feedback control of dynamic bipedal robot locomotion (CRC press)
[5] Martin W C, Wu A and Geyer H 2015 2015 IEEE International Conference on Robotics and Automation (ICRA) (IEEE) pp 6307–6312
[6] Jovic J, Escande A, Ayusawa K, Yoshida E, Kheddar A and Venture G 2016 IEEE Transactions on Robotics 32 728–735
[7] Popov D and Klimchik A 2019 2019 IEEE International Conference on Mechatronics (ICM) vol 1 (IEEE) pp 646–651
[8] Shigemi S, Goswami A and Vadakkepat P 2019 Humanoid robotics: A reference (Springer) pp 55–90
[9] Jatsun S, Savin S and Yatsun A 2017 International Conference on Interactive Collaborative Robotics (Springer) pp 75–82
[10] Kajita S, Kanehiro F, Kaneko K, Fujiwara K, Harada K, Yokoi K and Hirukawa H 2003 2003 IEEE International Conference on Robotics and Automation (Cat. No. 03CH37422) vol 2 (IEEE) pp 1620–1626
[11] Kuindersma S, Permenter F and Tedrake R 2014 2014 IEEE International Conference on Robotics and Automation (ICRA) (IEEE) pp 2589–2594
[12] Savin S 2020 Control and Signal Processing Applications for Mobile and Aerial Robotic Systems (IGI Global) pp 266–285
[13] Savin S, Khussainov R and Klimchik A 2020 Proceedings of 14th International Conference on Electromechanics and Robotics “Zavalskii’s Readings” (Springer) pp 125–136
[14] Caron S, Pham Q C and Nakamura Y 2015 2015 IEEE International Conference on Robotics and Automation (ICRA) (IEEE) pp 5107–5112
[15] Hirukawa H, Hattori S, Harada K, Kajita S, Kaneko K, Kanehiro F, Fujiwara K and Morisawa M 2006 Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006. (IEEE) pp 1976–1983
[16] Kuindersma S, Deits R, Fallon M, Valenzuela A, Dai H, Permenter F, Kooen T, Marion P and Tedrake R 2016 Autonomous robots 40 429–455
[17] Deits R and Tedrake R 2014 2014 IEEE-RAS international conference on humanoid robots (IEEE) pp 279–286
[18] Deits R L H 2014 Convex segmentation and mixed-integer footstep planning for a walking robot Ph.D. thesis Massachusetts Institute of Technology
[19] Aceituno-Cabezas B, Dai H, Cappelletto J, Grieco J C and Fernández-López G 2017 2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE) pp 4467–4472
[20] Atkeson C G, Babu B, Banerjee N, Berenson D, Bove C, Cui X, DeDonato M, Du R, Feng S, Franklin P et al. 2016 submitted to the DRC Finals Special Issue of the Journal of Field Robotics 1
[21] Yi S J, McGill S G, Vadakedathu L, He Q, Ha I, Han J, Song H, Rouleau M, Zhang B T, Hong D et al. 2015 Journal of Field Robotics 32 315–335
[22] Savin S 2018 2018 International Russian Automation Conference (RusAutoCon) (IEEE) pp 1–6
[23] Savin S 2019 2019 3rd School on Dynamics of Complex Networks and their Application in Intellectual Robotics (DCNAIR) (IEEE) pp 161–162
[24] Featherstone R 2014 Rigid body dynamics algorithms (Springer)