Journal of Information Systems Engineering and Management
2025, 10(55s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Kinematics Prediction of a Robotic Arm Using Traditional Machine Learning and CNN-LSTM Techniques

Hector Mora [1 *], Dalila Pachajoa [1], Jesús Insuasti [2]
[1] *Tecnofilia Research Group, Faculty of Engineering, CESMAG University, Pasto 520001, Colombia*
[2] *Galeras.NET Research Group, Systems Engineering Department, University of Nariño, Pasto 520001, Colombia*
**Corresponding author: hamora@unicesmag.edu.co*

ARTICLE INFO

Received: 29 Dec 2024

Revised: 15 Feb 2025

Accepted: 24 Feb 2025

ABSTRACT

Learning inverse kinematics in 3 degrees of freedom (3-DOF) robots usually requires manual intervention, which limits their autonomy. This research proposes to optimize this process using Machine Learning techniques, evaluating the performance of models trained with tabular (organized in tables) and visual (videos and images) data. A robotic arm that could trace a given letter in space was simulated. Cartesian coordinates and angular positions of the robot's joints were extracted to structure a dataset organized in tables (tabular data) and the videos corresponding to each writing task. With these datasets, the performance was compared on the one hand using tabular data with traditional Machine Learning algorithms such as Support Vector Machines (SVMs), Artificial Neural Networks (ANNs), Decision Tree (DTs), and Random Forest (RFs); on the other hand, with the videos with contemporary Machine Learning algorithms with convolutional neural networks of short and long temporal memory CNN-LSTM. The results show good predictability for most of the algorithms under study, with coefficients of determination that are very close to one. However, the CNN-LSTM trained with videos has less error than traditional Machine Learning algorithms and performs automatic feature extraction.

Keywords: Machine Learning, direct kinematics, inverse kinematics, robotic arm, simulation, CNN, LSTM

INTRODUCTION

Learning inverse and direct kinematics in 3 degrees of freedom (3-DOF) robots is a key challenge in robotics[2], even more so if one wants to predict future robot movements, as it traditionally requires external intervention for motion programming. However, applying Machine Learning techniques raises the possibility of automating these processes, allowing robots to learn to perform tasks without requiring manual reconfigurations. One of the main issues is the lack of specific datasets to train machine learning models in this task and the uncertainty about what kind of data (tabular or visual) and which algorithms offer the best performance regarding error and computational efficiency.

This research seeks to find the degree of predictability of a set of Machine Learning models in the prediction of the movements of a 3-DOF robot. On the other hand, some research seeks to find the transformation function that most closely approximates the kinematic aspects of the robot. In addition, the present study contributes to a simulator that generates a set of tabular and video data that can be useful for future studies, providing a basis for improving simulation [5] and control of robots with machine learning.

The methodology used in this research follows a quantitative approach. It is based on the simulation [5] of a 3-DOF robotic arm [15] that traces letters in space. Subsequently, two data sets are created: a tabular one with Cartesian coordinates and a visual one with videos and images of the robot's movement. Then, the performance of different Machine Learning algorithms [3], including SVM, Artificial Neural Network [7] (ANN), Decision Tree [8] (DT) and Random Forest [6] (RF) were compared on the tabular data, while convolutional neural networks (CNN [9]) with LSTM [10] were applied on the visual data.

The results indicate that the visual dataset performed better than the tabular data in predicting the robot's inverse kinematics[2]. The CNN [9]-LSTM [10,2] neural network shows the lowest margin of error in the simulation [5].

Journal of Information Systems Engineering and Management
2025, 10(55s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

The most relevant theoretical aspects, methodology, results, and corresponding discussion are presented below.

METHODS

MACHINE LEARNING IN ROBOTICS

In robotics, Machine Learning solves problems such as motion control, computer vision, autonomous navigation, and path planning. Depending on the type of learning, ML in robotics can be classified into three main categories.

(1) **Maintaining the Integrity of the Specifications:**

Supervised learning: The model is trained with a labeled data set where the inputs and outputs are known. It is used in robot control, object recognition, and optimal trajectory learning.

No Supervised learning is used when the robot must find patterns in unlabeled data. It is helpful in anomaly detection, motion clustering, and strategy optimization.

Reinforcement Learning: A robot learns by trial and error, optimizing its actions based on a reward. It is applied in autonomous robots, manipulator control, and simulation [5] of intelligent behavior.

(2) **Supervised Machine Learning algorithms [3]:**

The present study focuses on supervised learning, which seeks to predict the kinematic movements of a 3-DOF robot within the classical algorithms used.

Support Vector Machine [5] (SVM): This looks for patterns in the data with the best-fit maximum margin hyperplane. They generally use the kernel trick to find nonlinear patterns.

Artificial Neurol Network (fully connect) (ANN): They iteratively search for patterns by adjusting their synaptic weights through Back Propagation.

Decision Tree [8] (DT): This method finds patterns by applying rules to the feature set's attributes. It is generally used because of its high degree of interpretability.

Random Forest [6] (RF): It is an algorithm that combines several Decision trees [8] trained with data sub-partitions to determine the output pattern jointly.

We also considered using a Conv-LSTM [10] model that combines CNN [9] convolutional neural networks and LSTM [10] short and long-memory neural networks.

Convolutional Neural Network (CNN)[9]: These networks can extract automatic features from data by learning convolutional filters. One-dimensional Conv1D, two-dimensional Conv2D, and three-dimensional Conv3D convolutions are used depending on the signal.

Long Short-Term Memory Neural Networks (LSTM [10]): They are neural networks with memory cells simulated by logic gates capable of retaining and discarding information. They are used for sequential data (texts, videos, time series [14]).

(3) **Most relevant ML applications in robotics:**

Optimization of inverse kinematics [2]: Models such as neural networks can learn to predict a robot's joint angles [11] from a desired position.

Simulation [5] of movements: Robots can be trained in virtual environments to generate data that will be used in the real world.

Correction of movement errors: Algorithms such as Artificial Neural Networks (ANN) [7] and Support Vector Machines (SVM) [5] can minimize errors in the execution of trajectories.

To develop this type of application, it is necessary to understand the formal aspects that allow replicating and studying the kinematics of robots related to their joint structure.

Journal of Information Systems Engineering and Management
2025, 10(55s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

DIRECT AND INVERSE KINEMATICS [2]

Direct kinematics: It is formally for a robotic arm [15] of n degrees of freedom to obtain the position and orientation of the end of a robot by angular changes in its joints, such that defined by the Denavit-Hartenberg [1] method, where are homogeneous transformation matrices see Equations (1) and (2).

$$fourT_z = \begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & \mathrm{dx_i} \\ \sin\theta_i & \cos\theta_i & 0 & dy_i \\ 0 & 0 & 1 & \mathrm{dz_i} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (1)$$

Equation (1). Homogeneous transformation matrix for the z-axis

$$T_x = \begin{bmatrix} 1 & 0 & 0 & \mathrm{dx_i} \\ 0 & \cos\alpha_i & -\sin\alpha_i & dy_i \\ 0 & \sin\alpha_i & \cos\alpha_i & \mathrm{dz_i} \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (2)$$

Equation (2). Homogeneous transformation matrix for the x-axis

Where $Ti = Ti(q, dx, dy, dz)$ q represents an angle and dx, dy, dz are displacements in each respective axis, therefore at a time i, θi is the rotation angle in the Z axis, αi is the rotation angle in the X axis and dxi, dyi, dzi the displacements. To determine the endpoint position, the corresponding homogeneous transformation matrices are multiplied with the origin of the coordinate system, as shown in Equation (3).

$$P_n = \begin{bmatrix} X_n \\ Y_n \\ Z_n \\ 1 \end{bmatrix} = \prod_{j=1}^{n} T_j \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \quad (3)$$

Equation (3). Calculating the position of the nth end of a robot

Inverse kinematics reference [2]: Inverse kinematics [2] is the process by which the angles of the joints of a robot are determined from a desired position in space. In robotics, it is fundamental for the planning of autonomous movements. In formal terms, what is sought is to find the inverse function of the inverse kinematics [2] as expressed in Equation (4).

$$\Theta = f^{-1}(X, Y, Z) \quad (4)$$

Equation 4. The inverse function of the direct kinematics of a robot

Where X, Y, Z are the end coordinates n of the robot, and Θ represents the vector of angles corresponding to the robot joints such that.

$\Theta = [\theta 1, \theta 2, \ldots, \theta n]$

METHODOLOGY

In this research, a robot arm simulator was composed to extract the datasets, finally train the Machine Learning algorithms [3], and choose the best configuration. The above can be seen schematized in Fig.1.

Journal of Information Systems Engineering and Management
2025, 10(55s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

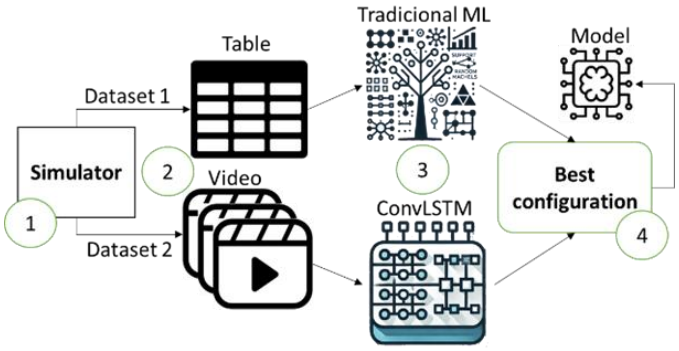

Fig.1. Diagram of the methodology used.

CONSTRUCTION OF 3-DOF ROBOTIC ARM [15] SIMULATOR

Construction of letters: For the construction of the robotic arm [15] simulator, the direct and inverse kinematics [2] were designed and programmed using the Python programming language, using the Denavit Hartemberg algorithm of the 3-DOF robot focused on letter writing. To do this, initially, the vertices of a letter were plotted in 3D Figure 2, then the letter's stroke is created as shown in the matrix of Equation (3) where the stroke follows the order A, B, C, D, E.

$$(x, y, z) = \begin{cases} \boldsymbol{A}(5.00, 0.00, 0.00) \\ \boldsymbol{B}(2.50, 5.00, 0.00) \\ \boldsymbol{C}(0.00, 0.00, 0.00) \\ \boldsymbol{D}(1.25, 2.50, 0.00) \\ \boldsymbol{E}(3.75, 2.50, 0.00) \end{cases} \quad (5)$$

Equation (5). Vertices letter A expressed in Figure 2

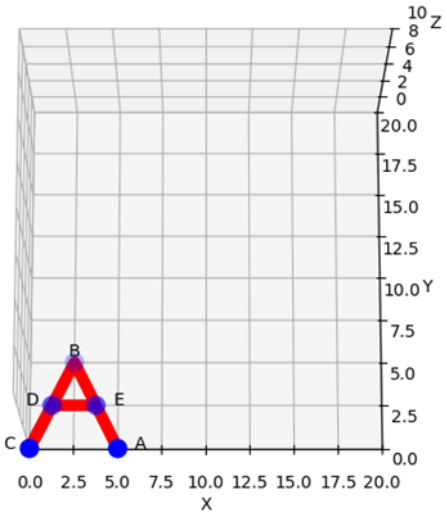

Fig.2. Vertices of the letter A

Once we have the line a letter should follow, we perform a linear interpolation between each segment, as shown in Equation (6).

$$P(t) = (1 - t)P_i + tP_{i+1} \quad (6)$$

Equation 6. Linear interpolation between two points in space.

Journal of Information Systems Engineering and Management
2025, 10(55s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Where PiPi+1 represents the vertices of a stroke of a letter; for example, stroke $\bar{A}\bar{B}$ and$t$ represents the interpolation parameter between 0 and 1. With the interpolation of each stroke, we have the set of points L that compose a letter, which is transformed to move or rotate the letter, as shown in Equation (7).

$$L_n = LT_i \tag{7}$$

Equation (7). Set of transformed letter points

Inverse Kinematic: With the set of points *Ln*, we proceed to apply Equation (4), specifically for the 3-DOF robotic arm [15] (see figure 3) of this study is expressed as shown in Equations (8),(9),(10),(11),(12) and (13), where l_1,l_2 y l_3 are the sizes of each link of the robot, q_1,q_2 y q_3 are the rotation angles of the joints, the tg_2^(-1) represents the arctangent of two parameters used to find a correct and unambiguous angle in the conversion of Cartesian coordinates.

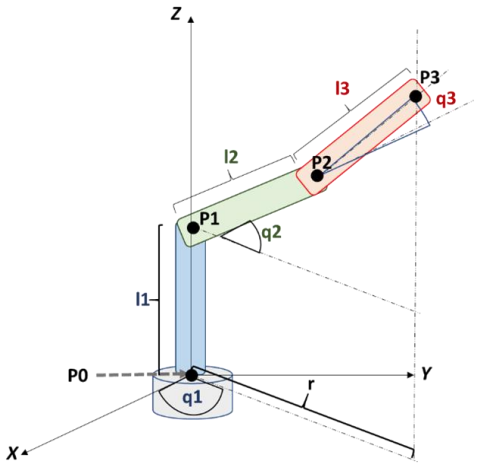

Fig.3. 3-DOF arm

$$\cos q_3 = \frac{x^2+y^2+(z-l_1)^2-l_2^2-l_3^2}{2l_2l_3} \tag{8}$$

Equation 8. Cosine of the end-effector joint

$$\sin q_3 = \sqrt{1-cs_3^2} \tag{9}$$

Equation 9. End effector joint sinus

$$r = \sqrt{x^2+y^2} \tag{10}$$

Equation 10. Distance from the projection of x and y

$$q_3 = tg_2^{-1}(\sin q_3, \cos q_3) \tag{11}$$

Equation 11. Angle of the end effector 3

$$q_2 = tg_2^{-1}(z-l_1, r) - tg_2^{-1}(l_3 \sin q_3, l_2 + l_3 \cos q_3) \tag{12}$$

Equation 12. The angle of the penultimate articulation

$$q_1 = tg_2^{-1}(y, x) \tag{13}$$

Journal of Information Systems Engineering and Management
2025, 10(55s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Equation 13. Angle of the ultimate articulation

Direct kinematics: Once the inverse kinematics [2] were modeled to find $q1$, $q2$ y $q3$, the transformations Ti were modeled as shown in equations (14), (15), (16), (17) and (18).

$$P_0 = T_z(0,10,10,0) \quad (14)$$

Equation 14. Positioning of the base

$$P_1 = T_z(0,10,10,l_1) \quad (15)$$

Equation 15. Positioning of the end of link 1

$$P_2 = P_0P_1T_x\left(\frac{\pi}{2},0,0,0\right)T_z(q_2,l_2\cos q_2\,,l_2\sin q_2\,,0) \quad (16)$$

Equation 16. Positioning of the end of link 2

$$P_3 = P_2T_z(q_3,l_3\cos q_3\,,l_3\sin q_3\,,0) \quad (17)$$

Equation 17. Positioning of the end of link 3

After constructing the letters and modeling the robot's direct and inverse kinematics [2], the respective routines and a 3-DOF writer robot simulator were coded in Python using the animation module of the Matplotlib library.

GENERATION OF TABULAR AND VISUAL DATASETS

Once the robotic arm [15] simulator was built, routines were integrated to extract the angles of the joints and the corresponding Cartesian coordinates of the links, as well as the recording of the writing of the robot. According to the above, 27 simulations [5] corresponding to the strokes of each of the letters of the alphabet were executed, and in coherence to the above, 27 CSV files were generated through the pandas library and 27 videos in mp4 format recorded from the animation object instantiated from the FuncAnimation class of the animation module of the matplotlib library.

COMPARISON OF MACHINE LEARNING ALGORITHMS

Following the acquisition of the data sets, training was continued using the tabular data where the target set y=q_3 and the set of features X=[q_1,q_2,x_0,x_1,x_2,x_3,y_0,y_1,y_2,y_3,z_0,z_1,z_2,z_3]. Subsequently, the feature set was transformed into a time series [14] format with window size n={1,2,4,8} and where the Machine Learning algorithms[3] SVM (Support Vector Machine[5]), ANN (Artificial Neural Network[7]), DT (Decision Tree[8]) and RF (Random Forest[6]) were used through the Sckit-Learn library, with a standard transformation over the entire dataset and using default hyperparameters.

Table 1. Model architecture

Layer	**hyperparameters**
Input	High: high Width: width Channels: 3
ConvLSTM2D	Filters 64 Kernel 5x5 Padding same Activation ReLU
BatchNormalization	No applied
ConvLSTM2D	Filters 64 Kernel 3x3 Padding same Activation ReLU
BatchNormalization	No applied
ConvLSTM2D	Filters 64 Kernel 1x1

Journal of Information Systems Engineering and Management
2025, 10(55s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Layer	hyperparameters
	Padding same Activation ReLU
Conv3D	Filters 3 Kernel 3x3x3 Activation sigmoid Padding same

Then, we experimented with the video data using a convolutional neural network of long and short memory (Conv-LSTM [10]), whose architecture is shown in Table 1, using the Tensor-Flow library. We also tried to predict the next frame from the previous frames. Binary cross entropy [5], Adam optimizer, and Early Stopping with Patience 5 were used as losses.

To guarantee the validity and reliability of the results, 80% of the total data were used for training and 20% for test data; in the case of the Conv-LSTM [10] model, the training data were subdivided into 20% for validation. The determination coefficient $R2$ and binary cross entropy [5] were used as a comparison metric.

RESULTS

CONSTRUCTION OF THE 3-DOF ROBOTIC ARM [15] SIMULATOR

The robotic arm [15] simulator allows for the simulation of the writing of fine strokes of alphabet letters from a given coordinate in the space reachable by the robot links. This simulator consists of:

Path module: In charge of generating the set of points that compose a letter in Cartesian coordinates, correctly interpolated and organized to compose a stroke.

Inverse kinematics [2] module: In charge of finding the rotation angles of the robot's joints for the given letter from a starting location.

Generator module: In charge of obtaining the start and end points through direct kinematics given the rotation angles of each joint and thus building the graphic artifacts that make up the robot.

Animation module: It is responsible for obtaining the video frames using the generated robot and composing the animation with them.

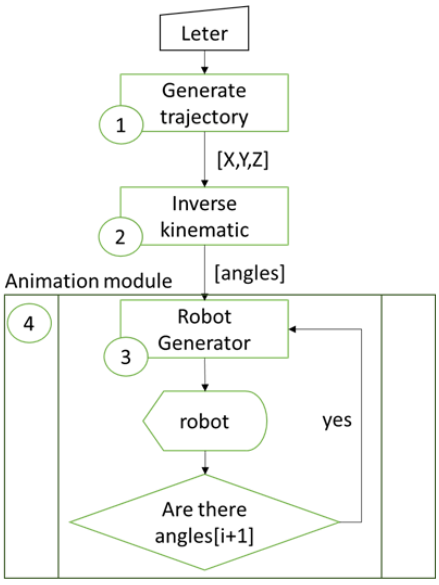

Fig.4. Modules that make up the robot simulator

Journal of Information Systems Engineering and Management
2025, 10(55s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Figure 4 shows the interaction between the simulator modules; it can be observed that when generating the trajectory of a letter, all the angles are calculated from the inverse kinematics [2]. Then, each set of articulated stroke angles is traversed to show the frame of the generated robot.

Figure 5 shows some examples of frames of the simulation [5] performed.

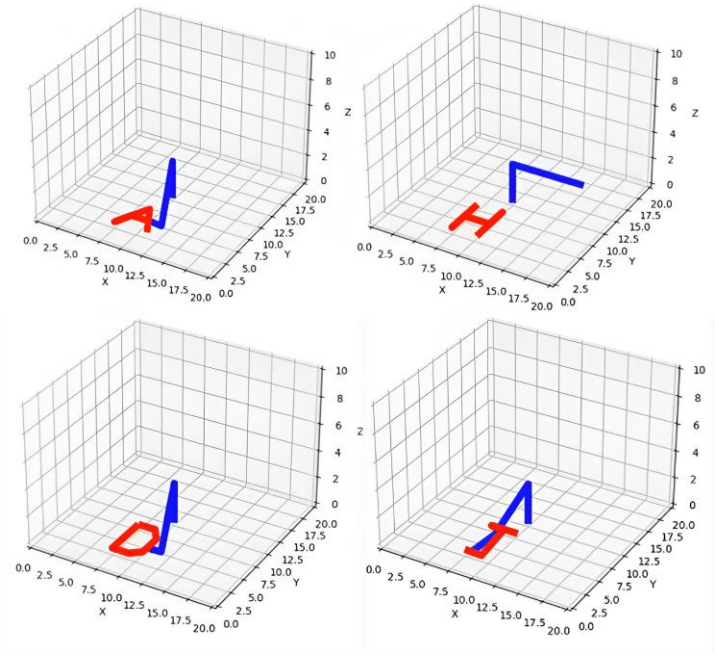

Fig.5. Frames of simulation [5] of writing letters A, H, D, and J.

GENERATION OF TABULAR AND VISUAL DATA SETS

Tabular dataset reference [4]: The tabular dataset [4] consists of 1332 records, which contain 16 attributes that can be seen in Figure 3 and are described below:

Angular attributes: $q1$, $q2$, $q3$ are the angles corresponding to the rotational movement of each joint. The range of which is between 0 and π degrees.

Cartesian attributes: $x0$, $y0$, $z0$ coordinates of the beginning of the first link $P0$, $x1$, $y1$, $z1$ coordinates of the end of the first link $P1$, $x2$, $y2$, $z2$ coordinates of the end of the second link $P2$, $x3$, $y3$, $z3$ coordinates of the end of the third and last link $P3$. The range of each point corresponds to the real ones bounded by Equations 14,15,16,17 and 18, respectively.

Finally, the last attribute corresponds to the plotted letter whose value is one of the 26 letters of the alphabet.

Dataset of videos: The video dataset corresponds to 26 videos corresponding to the simulation [5] of writing each letter of the alphabet, as shown in Figure 5. Each video has a total of $11NT$ frames where NT is the number of strokes of a letter, and 11 is the number of interpolated values in a stroke.

COMPARISON OF THE PERFORMANCE OF MACHINE LEARNING ALGORITHM

The resulting metrics for both the traditional Machine Learning algorithms [3] with the tabular dataset [4] and the Conv-LSTM [10] model trained with videos are shown below.

Performance of traditional Machine Learning algorithms [3]: Table 2 shows the coefficient of determination (Cai et al., 2019) (R2), the error measured by binary cross-entropy (BCE) per model and window size. The evaluation was performed with the tabular dataset [4].

Journal of Information Systems Engineering and Management
2025, 10(55s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

Table 2. Tabular time series [14] metrics

Model	Window size	R2	BCE
SVM	1	0.9677	0.6044
ANN		0.9970	0.6032
DT		0.9964	0.6031
RF		**0.9984**	**0.6030**
SVM	2	0.9690	0.6043
ANN		0.9973	0.6031
DT		0.9970	0.6031
RF		**0.9987**	**0.6030**
SVM	4	0.9737	0.6043
ANN		0.9970	0.6032
DT		0.9984	0.6030
RF		**0.9989**	**0.6030**
SVM	8	0.9563	0.6058
ANN		0.9840	0.6042
DT		0.9966	0.6036
RF		**0.9986**	**0.6036**

Conv-LSTM [10] model performance

Figure 6 shows the training curves. Measuring the loss with the test data yielded a binary cross entropy [5] of 0.3639.

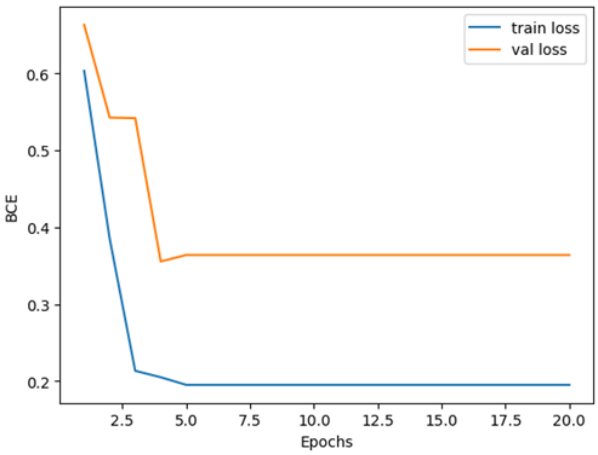

Fig.6. Conv-LSTM [10] model training curves

DISCUSSION AND ANALYSIS OF THE RESULTS

The research shows that the Conv-LSTM [10] model has a lower binary cross-entropy than all experiments with traditional Machine Learning algorithms [3]. However, the degree of predictability of these algorithms, as measured by their coefficient of determination (Cai et al., 2019), is very close to one, demonstrating a perfect fit of the models to the data.

When analyzing the training curves in Figure 6, a wide gap between the validation and training loss is observed, suggesting a possible adjustment in the training. This indicates that the model, despite using batch normalization, needs to experiment with other regularization techniques.

Good predictability metrics are obtained in the present study; however, these studies focus on predicting the inverse and direct kinematics of robots, while this study's task is to predict future robot motions.

Journal of Information Systems Engineering and Management
2025, 10(55s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

ACKNOWLEDGMENT

The preferred spelling of the word “acknowledgment” in America is without an “e” after the “g.” Avoid the stilted expression “one of us (R. B. G.) thanks ...”. Instead, try “R. B. G. thanks...”. Put sponsor acknowledgments in the unnumbered footnote on the first page.

DECLARATIONS

Author contribution. The contribution or credit of the author must be stated in this section.

Funding statement. The unding agency should be written in full, followed by the grant number in square brackets and year.

Conflict of interest. The authors declare no conflict of interest.

Additional information. No additional information is available for this paper.

Venenatis urna cursus eget nunc scelerisque viverra. Lectus magna fringilla urna porttitor rhoncus dolor. Proin libero nunc consequat interdum varius sit. Arcu felis bibendum ut tristique et egestas quis.

REFRENCES

[1] C. Bentley, Statistical optimization of CNN-LSTM network architectures: A case study in autonomous vehicle control, *University of Waterloo*, 2024. [Online]. Available: https://uwspace.uwaterloo.ca/items/b8987b74-abf5-42b9-a328-799038d16efa

[2] A. Calzada-Garcia, J. G. Victores, and F. J. Naranjo-Campos, "A review on inverse kinematics, control, and planning for robotic manipulators with and without obstacles via deep neural networks," *Algorithms*, vol. 18, no. 1, p. 23, 2025. [Online]. Available: https://doi.org/10.3390/a18010023

[3] H. Jahanshahi and Z. H. Zhu, "Review of machine learning in robotic grasping control in space application," *Acta Astronautica*, vol. 211, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S009457652400211X

[4] N. Wagaa, Deep learning based modeling and control of robotic systems, *ProQuest*, 2024. [Online]. Available: https://search.proquest.com/openview/519c90e9406be0b5591c8684416bd6c5/1

[5] M. Mansour, K. Serbest, M. Kutlu, and M. Cilli, "Estimation of lower limb joint moments based on the inverse dynamics approach: a comparison of machine learning algorithms," *Medical & Biological Engineering & Computing*, vol. 61, pp. 857–871, 2023. [Online]. Available: https://link.springer.com/article/10.1007/s11517-023-02890-3

[6] S. T. Mubarrat y S. Chowdhury, "Convolutional LSTM: a deep learning approach to predict shoulder joint reaction forces," *Comput. Methods Biomech. Biomed. Engin.*, vol. 26, no. 1, pp. 65–77, 2022, doi: 10.1080/10255842.2022.2045974.

[7] R. Rahmatizadeh, P. Abolghasemi, and A. Behal, "From virtual demonstration to real-world manipulation using LSTM and MDN," *in Proc. AAAI Conf. Artificial Intelligence*, vol. 32, no. 1, pp. 1015–1023, 2023. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/12099

[8] S. Funabashi, S. Ogasa, T. Isobe, and T. Ogata, "Variable in-hand manipulations for tactile-driven robot hand via CNN-LSTM," *in Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, 2020. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/9341484

[9] Y. Chen, S. Yu, K. Ma, S. Huang, G. Li, S. Cai, and L. Xie, "A continuous estimation model of upper limb joint angles using surface electromyography and deep learning method," *IEEE Access*, vol. 7, pp. 140496–140506, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8918182

[10] W. Deng, F. Ardiani, K. T. P. Nguyen, and M. Benoussaad, "Physics informed machine learning model for inverse dynamics in robotic manipulators," *Applied Soft Computing*, vol. 150, p. 110355, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1568494624006513

[11] N. Liu, L. Li, B. Hao, L. Yang, T. Hu, T. Xue, and S. Wang, "Modeling and simulation of robot inverse dynamics using LSTM-based deep learning algorithm," *IEEE Access*, vol. 7, pp. 145218–145229, 2019. [Online]. Available: https://ieeexplore.ieee.org/document/8918341

[12] R. Patel, Modeling and learning control system design for robots using deep learning techniques, *Laurentian University*, 2021. [Online]. Available: https://laurentian.scholaris.ca/items/013d6696-426c-417b-b102-6316f5616bb2

Journal of Information Systems Engineering and Management
2025, 10(55s)
e-ISSN: 2468-4376
https://www.jisem-journal.com/

Research Article

[13] Y. Zhang, Enhancing surgical gesture recognition using bidirectional LSTM and evolutionary computation, *McMaster University*, 2024. [Online]. Available: https://macsphere.mcmaster.ca/handle/11375/30421
[14] P. Sathiyanarayanan, Data driven feedforward control of a 2-DOF redundantly actuated manipulator, *University of Twente*, 2021. [Online]. Available: http://essay.utwente.nl/89139/
[15] F. S. Wibowo, R. Mandal, and H. I. Lin, "BN-LSTM-based energy consumption modeling for an industrial robot manipulator," *Robotics and Computer-Integrated Manufacturing*, vol. 86, p. 102668, 2024. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0736584523001047
[16] A. Sintov and N. D. Kahanowich, "Learning human-arm reaching motion using IMU in human-robot collaboration," *arXiv preprint* arXiv:2308.13936, 2023. [Online]. Available: https://arxiv.org/abs/2308.13936