Applied Computer Systems
ISSN 2255-8691 (online)
ISSN 2255-8683 (print)
December 2021, vol. 26, no. 2, pp. 150–157
https://doi.org/10.2478/acss-2021-0018
https://content.sciendo.com

Time Series Forecasting of Mobile Robot Motion Sensors Using LSTM Networks

Anete Vagale[1*], Luīze Šteina[2], Valters Vēciņš[3]
[1]*Norwegian University of Science and Technology, Aalesund, Norway*
[2]*LLC Robotic Solutions, Riga, Latvia*
[3]*Riga Technical University, Riga, Latvia*

Abstract – **Deep neural networks are a tool for acquiring an approximation of the robot mathematical model without available information about its parameters. This paper compares the LSTM, stacked LSTM and phased LSTM architectures for time series forecasting. In this paper, motion sensor data from mobile robot driving episodes are used as the experimental data. From the experiment, the models show better results for short-term prediction, where the LSTM stacked model slightly outperforms the other two models. Finally, the predicted and actual trajectories of the robot are compared.**

Keywords **– Deep neural networks, long short-term memory (LSTM), mobile robot, time series forecasting.**

I. Introduction

Time series forecasting is a powerful tool whose importance and topicality have been proven throughout the years by a high number of publications [1], [2] and a wide range of applications. One of the promising applications is predicting the future states of a wheeled mobile robot using historic time series by acquiring its imitation model. One of the motivators of this study is the little amount of research performed on time series forecasting of the mobile robot motion sensor data. Another reason is that algorithm testing for robots can be very time-consuming, and an accurate imitation model could decrease the time spent on testing these algorithms.

When it comes to mobile robot system modelling, it is possible to develop the mathematical models of the system based on the dynamics and kinematics of the robot [3], [4]. However, these models are often idealised, and, in reality, they do not correspond to the actual system of the robot. Additionally, acquiring data of all the different factors that are necessary for developing the imitation model is a time-consuming and complex task. In order to acquire a more realistic robot imitation model for mapping and future state prediction, different forecasting methods can be used. This allows considering the specific parameters of different robots, such as sensor calibration errors, internal production errors, imprecisions in the mechanics development, friction coefficients, inertia, etc.

There are two types of forecasting methods depending on whether the historic data are available [5]: (i) qualitative forecasting and (ii) quantitative forecasting. Quantitative forecasting, in turn, can be based either on *time series* data or *cross-sectional* data. In this paper, we focus on methods and models for quantitative time series forecasting. The methods for quantitative time series forecasting differ in their properties, accuracies, and costs.

Fig. 1 presents the authors' view on classification of different time series forecasting models and methods. Some well-developed, traditional time series forecasting methods are the autoregressive integrated moving average (ARIMA) model [6] and its variations (ARMA, SARIMA, SARIMAX, etc. [7]), exponential smoothing, decomposition models, average method and the naïve method. A disadvantage of some of these methods is that their implementation requires complete knowledge about the use of the methods, and they expect linear data as the system input [8]. In the case of mobile robots, the motion sensor data are non-linear, therefore requiring a different approach.

Few articles have proposed a quality overview of various time series forecasting methods for different applications. A comprehensive review of machine learning methods for time series forecasting is given in [9]. The paper presents different time series forecasting methods, such as Gaussian processes, convolutional neural networks (CNN), recurrent neural networks (RNN), deep neural networks (DNN) and multi-horizon forecasting models for weather and climate modelling. In another research [10], the authors compare several machine learning (ML) models for time series forecasting, concluding that the multilayer perceptron and the Gaussian process regression outperform others and show the best results in the comparison. The authors in the paper [11], in turn, review techniques for building energy consumption forecasting, providing advantages and disadvantages of the most commonly used time series forecasting methods.

One way of forecasting time series for mobile robot navigating indoors is proposed in [12] with an aim of enhancing navigation safety. The use of deep conditional generative

[*] Corresponding author's e-mail: anete.vagale@ntnu.no

models for trajectory forecasting of table tennis robot is proposed in [13].

In this paper, we choose three different variations of the LSTM architecture to train a mobile robot imitation model using motion sensor data. First, sensor data from the mobile robot are acquired and sorted. Secondly, the deep neural network models are implemented for forecasting the motion sensor data. Then, the performance of these models is examined with different hyperparameter setups.

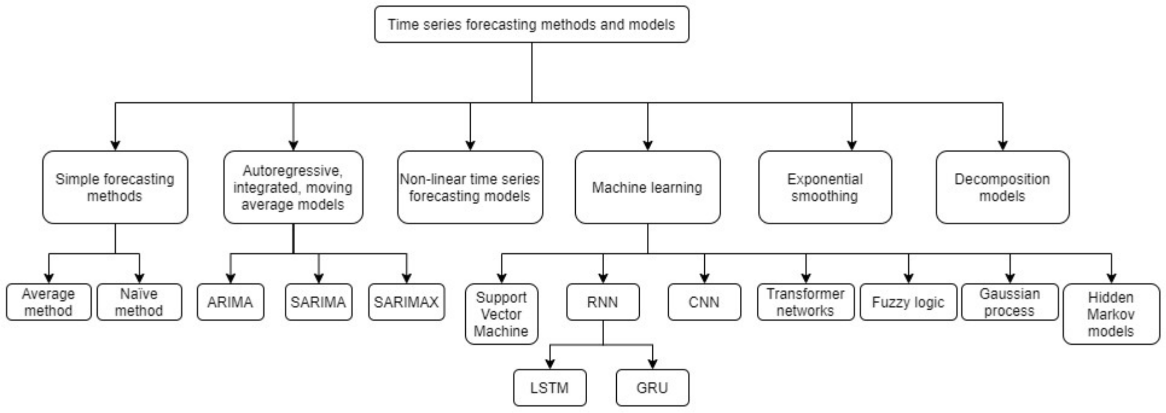

Fig. 1. Classification of time series forecasting methods and models.

Finally, the results of the obtained neural network models are compared and evaluated. The training of the LSTM models was carried out using the high-performance computing (HPC) infrastructure of Riga Technical University.

The rest of the paper is organised as follows: Section II presents the models used for motion sensor time series forecasting. Section III explains the experimental setup, used dataset, infrastructure, training process and the performed grid search. Section IV presents results and discussion. Finally, some concluding remarks are drawn in Section V.

II. Time Series Forecasting Architectures

A recurrent neural network (RNN) [14] is a network consisting of a linear input layer, a hidden layer and a linear output layer. In this paper, a modified version of recurrent neural networks called long short-term memory (LSTM) model [14] is used for time series forecasting. LSTM consists of three gates: *forget gate*, *input gate* and *output gate*. With its improved memory over recurrent neural network, a LSTM model can handle longer time series. Three variations of LSTM architectures used in this study are as follows: (i) *original LSTM*, (ii) *stacked LSTM*, and (iii) *phased LSTM*. These architectures are presented in the section below.

A. Original LSTM

The first model used in this paper is based on the original LSTM. This model consists of a linear input layer, $n = 2$ LSTM layers with a dropout $d = 0.2$, and a linear output layer (see Fig. 2). The input of the model consists of five features that are as follows: (i) signals from the right encoder, (ii) left encoder, (iii) gyroscope, (iv) right motor control value and (v) left motor control value. The three model outputs are the predicted right encoder, left encoder, and gyroscope sensor values. Two activation functions that are chosen for this model are *rectified linear activation unit* (ReLU) and *hyperbolic tangent function* (tanh). The ReLU function is applied before the input layer. While the hyperbolic tangent is applied to the output layer.

B. Stacked LSTM

Stacked LSTM [15] is a variation of the original LSTM architecture, with an exception of introducing hidden state connections between layers. In the case of the stacked LSTM, the output vectors are passed from one layer to the next, where the resulting output vector is the average of the output values of all layers.

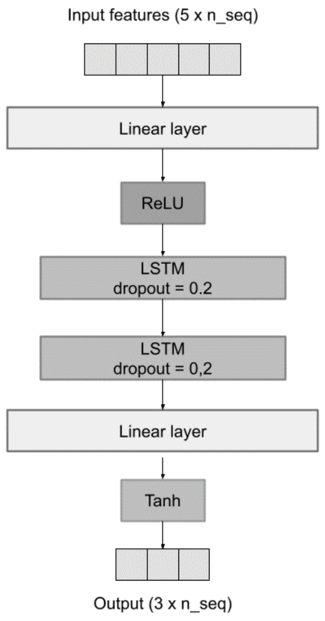

Fig. 2. The original LSTM architecture.

An additional learning parameter added to this network is the initialisation value of the hidden state. This network also consists of two linear layers, one before and one after the two LSTM layers (see Fig. 3). This model uses the same activation functions — *ReLU* and *tanh*.

C. Phased LSTM with Peepholes

Phased LSTM model [16] specialises in processing long time series. In this model (see Fig. 4), a time gate k_t is added to the LSTM model that can take one of three phases: (i) opening phase, (ii) closing phase or (iii) closed phase. The change of phase is controlled by three parameters: the oscillation frequency τ, the ratio of the opened time r_{on} and the phase shift s between each unit of the phased LSTM.

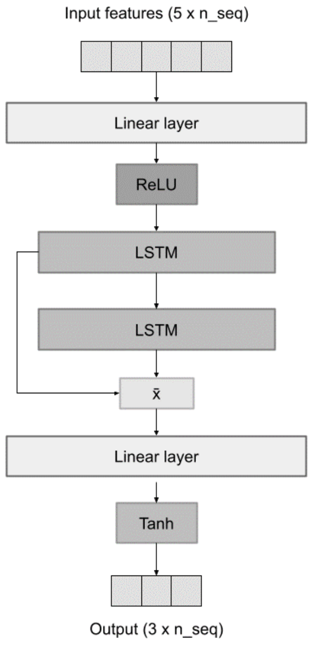

Fig. 3. The stacked LSTM architecture.

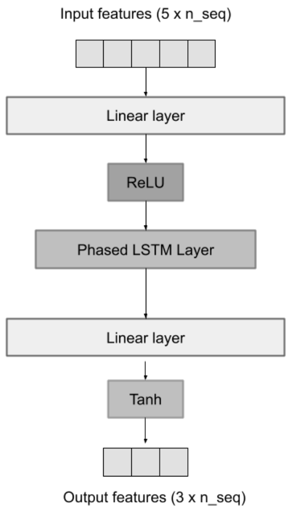

Fig. 4. The phased LSTM with peephole architecture.

A special part about this model is introduction of peepholes [17] that prevent a situation when the output gate might exclude important information. A few architecture specific parameters are introduced here: $\alpha = 1e^{-3}$, $\tau_{\max} = 3.0$ and $r_{\text{on}} = 5e^{-2}$. In this case, α is the leakage rate [16]. Peephole connections have also been introduced in this model. Similarly, like in previous models, it uses linear layers and two activation functions (*ReLU* and *tanh*). However, this model has only a single phased LSTM unit, and a dropout was not used in this network.

III. Experimental Setup

In this section, the time series forecasting models are applied to the motion sensor data of a mobile robot to test whether the model can accurately forecast the values of wheel encoders and gyroscope. The models are evaluated using mean absolute error (MAE), mean square error (MSE) and dynamic time warping (DTW) methods.

A. Software

In the experiment, software for model training, result acquisition and comparison has been developed. The *PyCharm* environment and *Anaconda* package manager have been used for the algorithm implementation in *Python* language. The functionality of the software consists of several modules:

1) data preprocessing;
2) data uploading – preparing data for the neural network;
3) original LSTM model;
4) stacked LSTM model;
5) phased LSTM with peephole model;
6) neural network training – training models, primary evaluation based on metrics and saving the trained models;
7) inference – evaluating the trained models; using the validation set to acquire sensor predictions and errors.

The PyTorch library [18] has been used for developing neural network models. The dynamic time warping algorithm for forecasting error calculation has been implemented using the fastdtw library. The training of the models has been performed using high-performance computing (HPC) infrastructure of Riga Technical University.

B. Dataset

In this experiment, the dataset has been acquired from the particular modelled robot. The mobile robot used in the experiment is equipped with a differential two-wheel drive, a gyroscope, and two encoders. The independent variables are the motor control signals for the right- and left-wheel drive. Sensor data have been obtained during driving segments, hereinafter referred to as episodes. Each episode is stored in a separate file where each row consists of sensor readings obtained at regular intervals. In the experiment, a total of 391 files have been retrieved with a different number of rows in each file. The shortest file has 80 sensor readings, while the longest has 143 802 readings. The average file length is ≈ 3932.40 rows. Most files are shorter; 75.96 % of the file length is within the range [80, 4480].

The original dataset is stored in *.csv* format. The data preprocessing consists of identifying and handling the erroneous encoder values that are replaced with a mean value of the previous and the following actual encoder readings. Some data characteristics can be seen in Table I.

Afterwards, the dataset is split into (i) a training set (80 % of files), (ii) a test set (18 %) and (iii) a validation set (2%).

TABLE I

The Structure and Characteristics of the Data after Preprocessing

Characteristic	Left motor control value	Right motor control value	Gyroscope	Right wheel encoder	Left wheel encoder
Max value	439.00	439.00	131 594.00	15 405.00	15 417.0
Min value	–439.00	–439.00	–182 121.00	–11 329.00	–9010.00
Median	253.00	338.00	480.00	4588.00	5282.00
Mean	226.84	262.85	6760.57	3976.88	4578.56
Mode	0	0	0	0	0
Mode value count	181 250	180 339	37 554	141 508	142 910

The split of training and test sets applied in this article is 80:20. This is a commonly accepted split ratio, and based on the relatively small dataset size, the authors consider it to be an appropriate ratio. The chosen validation set size is 10 % of the test set, and it is considered to be sufficient for this task.

When importing data, the input parameters are batch size *B*, sequence length *seq*, and expected number of files n_{pred}. When uploading data for the model training, the limit values of all data are obtained and scaled to the range [−1;1]. The control values are shifted backwards to obtain a sample where the sensor values are taken at time $t-1$ with control values from time *t*. Samples for prediction are obtained after each epoch according to the above-mentioned technique.

C. Neural Network Workflow

The hyperparameters chosen for the neural network tuning are (i) sequence length, (ii) learning rate, (iii) dropout, and (iv) number of layers. In the training phase, prediction is performed after each epoch. During the prediction, several driving episodes are selected from the validation dataset, leaving only the first sample of size [*seq*,*f*], where $f = 5$ is the number of features. Neural network models predict a sequence of the same length, only shifted by one timestep. The original input data are shifted, and the last element of the model predicted sequence is placed in the last position. For example, if the first input sequence consists of the input vectors x_{t-1}, x_t ,..., $x_{t+seq-1}$ and the last predicted vector is y_{t+seq}, then the next input string is x_t, x_{t+1},..., $x_{t+seq-1}$, y_{t+seq}. Thus, the model begins the rollout – predicting based on its own previous predictions.

D. Training and Grid Search

The following section describes the process of neural network training and hyperparameter tuning. The grid search consists of two phases. Firstly, neural network models are trained using all possible hyperparameter combinations. In the second phase, forecasting using all models is performed and the best model of each architecture is chosen based on the calculated prediction errors. The selected hyperparameters for the models in this study are as follows: (i) sequence length, (ii) learning rate and (iii) batch size. The possible values of these hyperparameters are presented in Table II.

TABLE II

Possible Values for Hyperparameters

Sequence length	Learning rate	Batch size
16, 32, 64, 128	$3e^{-3}$, $1e^{-3}$, $3e^{-4}$, $1e^{-4}$, $3e^{-5}$, $1e^{-5}$	64, 128

The model evaluation is performed in both grid search steps; however, in each phase, the comparison is within different scopes. In the first phase, all models are trained with a unique hyperparameter combination. Forecasting on the validation set and calculation of forecasting errors are performed after each epoch. After training each model, the weights of the epoch with the lowest error values are saved. Then, in the second phase, each saved model is used for forecasting on each driving episode from the validation dataset.

E. Testing Process

There are different methods for calculating the forecasting error [1]. In this study, the performance of all time series forecasting methods are evaluated based on three error measures: (i) mean absolute error (MAE) (1), (ii) mean square error (MSE) (2) and (iii) dynamic time warping (DTW). Additional to MAE and MSE, the authors in this study consider DTW to be a promising tool for comparing real and forecasted values.

$$MAE = \frac{1}{N}\sum_{t=1}^{N}\left|y_t - \hat{y}_t\right| . \quad (1)$$

$$MSE = \frac{1}{N}\sum_{t=1}^{N}\left(y_t - \hat{y}_t\right)^2 . \quad (2)$$

Here, y_t and $\hat{y}_t$ are the actual and predicted observation values at time *t*, and *N* is the size of the dataset. Dynamic time warping (DTW) is a method for comparing dynamic time series that are not always synchronous. This method is more described in [19].

IV. Results and Discussion

The models have been tested in two experiments: (i) with a relatively low prediction length of 600, and (ii) with a higher prediction length to test durability. Afterwards, the predicted robot positions have been compared with the actual positions.

The best models have been chosen based on the forecasting error measures. The obtained ranges of forecasting errors throughout the whole experiment are the following:

- MAE: [0.2813325;0.0266407];
- MSE: [0.1517630;0.0022978];
- DTW: [128.2911085;7.4350598].

A. Forecasting with the Sequence Length 600

From each architecture, one model with the best forecasting results has been chosen at a prediction length of 600 (see Table III). It can be seen that all of these models gained better results with the same batch size but different sequence length and learning rate values.

The table above shows that the best model in the experiment with the lowest forecasting errors is the stacked LSTM model with sequence length of 128, learning rate of 1e^{-3} and batch size of 128. Although its MSE measure is slightly higher than the one of the phased LSTM model, its MAE and DTW measures are lower than for other models. It is worth mentioning that this evaluation highly depends on the prediction length that in this case was 600. The authors speculate that with a higher number of considered timesteps the results would differ. An example of sensor predictions of the stacked LSTM model for one driving episode is given in Fig. 5 for the left encoder, in Fig. 6 for the right encoder, and in Fig. 7 for the gyroscope.

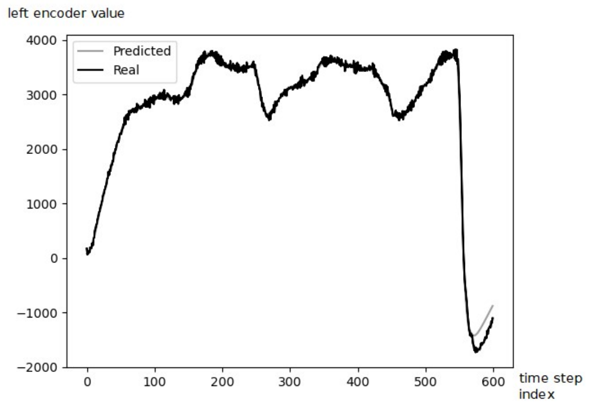

Fig. 5. An example of the left encoder readings – forecasted (grey) and real (black) values applying the stacked LSTM model.

In these figures, it can be seen that the forecasted gyroscope readings have higher error throughout the time series and especially at the peaks, in comparison with the forecasts of the encoder data. An important difference is also the noise that is noticeable in the actual value and is missing in the forecasts.

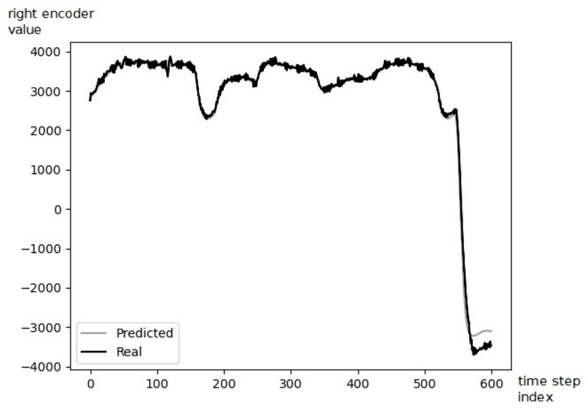

Fig. 6. An example of the right encoder readings – forecasted (grey) and real (black) values applying the stacked LSTM model.

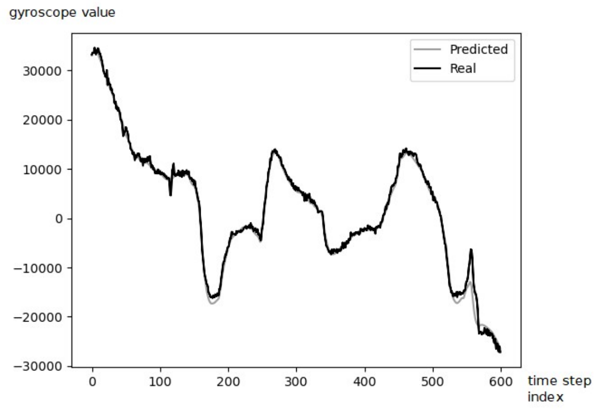

Fig. 7. An example of the gyroscope readings – forecasted (grey) and real (black) values applying the stacked LSTM model.

B. Forecasting with Higher Sequence Length

In real-life situations, when a mobile robot is navigating, there are several thousands of sensor readings per driving episode. Therefore, it is important to test the models with a higher prediction length to see if they are able to keep the low forecasting error for a longer period of time. For this experiment, the chosen prediction lengths were 1200, 3000 and 10 000. The forecasting errors of all the models with various prediction length are given in Table IV.

TABLE III
THE BEST HYPERPARAMETERS AND FORECASTING ERRORS FOR EACH MODEL WITH PREDICTION LENGTH OF 600

Model	Batch size	Sequence length	Learning rate	MAE	MSE	DTW
Original LSTM	128	64	3e^{-5}	0.0413085	0.0047809	11.0936524
Stacked LSTM	128	128	1e^{-3}	0.0266407	0.0026518	7.4350598
Phased LSTM	128	64	1e^{-3}	0.0276842	0.0022978	7.6637153

TABLE IV
THE FORECASTING ERRORS OF ALL MODELS WITH VARIOUS PREDICTION LENGTH

Model	Prognosis length	MAE	MSE	DTW
Original LSTM	1200	0.0237068	0.0014218	14.4852884
Original LSTM	3000	0.0220942	0.0010399	31.1709128
Original LSTM	10000	0.0212946	0.0008784	145.6757562
Stacked LSTM	1200	0.0212026	0.0011665	13.8564136
Stacked LSTM	3000	0.0207157	0.0009960	31.9183395
Stacked LSTM	10000	0.0218922	0.0011361	167.2592420
Phased LSTM	1200	0.0216834	0.0011402	13.2876328
Phased LSTM	3000	0.0208007	0.0010084	30.8149600
Phased LSTM	10000	0.0211377	0.0011128	139.9588102

The comparison of Table III and Table IV shows that in many cases the increased prediction length leads to an increased forecasting error. Secondly, the increase in the forecasting error of the phased LSTM is lower than the ones of the original or the stacked LSTM models, especially for the MSE and DTW measures. This result can also be observed in the sensor reading forecast at different prediction lengths. However, it has been observed that with the increase in the prediction length, the quality of the forecast does not decrease visibly.

C. Actual vs Predicted Trajectories

To test the suitability of the trained models for the imitation model task, trajectories of the robot navigation using actual and predicted sensor data have been calculated. Two ways of calculating the mobile robot heading (angle) are based on (i) gyroscope and (ii) encoder values. The robot used in the experiment calculates its heading using data from the gyroscope. Therefore, this technique has been applied first for the forecasted trajectory calculation. The results of one example show that the curvatures of both trajectories are similar; however, they are slowly diverging in time (the angle error increases) (see Fig. 8). These trajectories have been calculated based on the sensor data shown in Figs. 6–8.

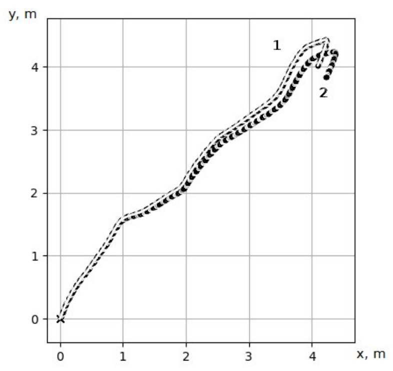

Fig. 8. The actual (1) vs predicted (2) trajectory of the robot, using the predicted sensor data from the stacked LSTM model (prediction length of 600).

The initial experiment results have shown better forecasting of the encoder data instead of the gyroscope. Therefore, another method for trajectory calculation – calculating heading based on the encoder data – has been tested. Fig. 9 compares the predicted and actual robot trajectories where the angle is calculated based on a) the gyroscope and b) encoder values. The method of calculating the angle based on the encoder values shows the improved results with a lower angle error.

D. Discussion

There are a few factors that could have led to imprecise sensor reading predictions. Firstly, the forecasting errors for the same model (the same prediction length) on different data varied a lot. That could be explained by the insufficient size of the dataset and the possible inability of the model to generalise.

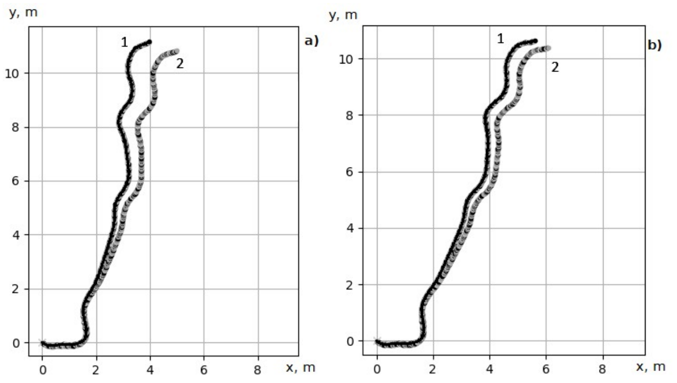

Fig. 9. Trajectories calculating heading based on the a) gyroscope and b) encoder values (prediction length of 1200), where 1 – actual trajectory and 2 – forecasted trajectory from the stacked LSTM data.

However, it is an inevitable problem that there might occur some situations when the models would not be able to predict sensor values precisely. One reason could be that the dataset was acquired during robot testing, following specific testing protocols.

Secondly, the choice of hyperparameters could be discussed. The most successful model in this experiment was acquired for prediction length of 600. However, the phased LSTM model with higher prediction length showed more resistance against the increase in the error. Although forecasting for longer driving episodes meets better real-life situations, the models used in this study are not suitable for long-term imitation of the mobile robot. The authors of this paper are of the opinion that the increased size of the dataset and the improved deep neural network architecture would aid in successful use of the models for long-term episodes. Since the data are of high aleatoric uncertainty, performing the same forecasting *n* times, the predicted sensor values will differ. A possible enhancement could be achieved by applying Bayesian neural networks that would also predict standard deviation.

Finally, the accumulation of the forecasting error for longer time series explains why the trajectory calculation based on the encoder readings produces lower error when forecasting for a smaller number of timesteps. Since the encoder data are predicted with a higher accuracy, the rate of displacement error accumulation is lower when compared to the gyroscope data.

V. Conclusion

In this paper, three variations of different LSTM time series forecasting architectures (original LSTM, stacked LSTM and phased LSTM with peepholes) have been implemented and their performance compared. Sensor readings of a gyroscope and two encoders of a wheeled mobile robot driving episodes have been used as the experimental data.

The experiment has shown improved results when applying the stacked LSTM architecture for short-term predictions. It has also been observed that the forecasted encoder values are closer to the actual data than the gyroscope forecasted values. When comparing the actual and forecasted trajectories, the heading calculation method based on the encoder values shows superior results over the gyroscope calculation method. The chosen prediction sequence length is one of the factors that could affect the resulting outcome. Another aspect to consider is a variety of data collected in the dataset. In this research, the data have been collected from a wheeled robot navigating in a limited number of rooms several times. Having possibly similar trajectories in the dataset could lead to less credible trained models.

Future research should consider collecting a larger and more diverse dataset to improve the credibility of the trained models. Also, other neural network architectures could be applied to the existing dataset to possibly improve the forecasting results. Another idea for future research is implementing Bayesian neural networks for improving the prediction outcome. Finally, a possible advancement would be predicting the future robot coordinates instead of the sensor readings.

Acknowledgements

The research has been supported by the European Regional Development Fund within the project “Robotic Solutions — True Autonomy for Indoor Autonomous Robots” 1.1.1.1/19/A/141.

References

[1] J. G. De Gooijer and R. J. Hyndman, “25 years of time series forecasting,” *International Journal of Forecasting*, vol. 22, no. 3, pp. 443–473, 2006. https://doi.org/10.1016/j.ijforecast.2006.01.001

[2] A. Tealab, “Time series forecasting using artificial neural networks methodologies: A systematic review,” *Future Computing and Informatics Journal*, vol. 3, no. 2, pp. 334–340, Dec. 2018. https://doi.org/10.1016/j.fcij.2018.10.003

[3] F. Chenavier and J. L. Crowley, “Position estimation for a mobile robot using vision and odometry,” in IEEE *International Conference on Robotics and Automation*, Nice, France, May 1992. https://doi.org/10.1109/robot.1992.220052

[4] F. Azizi and N. Houshangi, “Mobile robot position determination using data from gyro and odometry,” in *Canadian Conference on Electrical and Computer Engineering*, vol. 2, Niagara Falls, ON, Canada, May 2004, pp. 719–722. https://doi.org/10.1109/CCECE.2004.1345215

[5] R. Hyndman and G. Athanasopoulos, *Forecasting: Principles and Practice*, 3rd ed., Melbourne, Australia: OTexts, 2021.

[6] G. E. P. Box, G. M. Jenkins, G. C. Reinsel, and G. M. Ljung, *Time Series Analysis: Forecasting and Control*, 5th ed. Wiley, 2016.

[7] D. Mwanga, J. Ong’ala, and G. Orwa, “Modeling sugarcane yields in the Kenya sugar industry: A SARIMA model forecasting approach,” *International Journal of Statistics and Applications*, vol. 7, no. 6, pp. 280–288, 2017.

[8] Z. Wang and Y. Lou, “Hydrological time series forecast model based on wavelet de-noising and ARIMA-LSTM,” in *Proceedings of 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference, (ITNEC)*, Chengdu, China, Mar. 2019, pp. 1697–1701. https://doi.org/10.1109/ITNEC.2019.8729441

[9] B. Lim and S. Zohren, “Time-series forecasting with deep learning: A survey,” *Philosophical Transactions of the Royal Society A*, vol. 379, Feb. 2021. https://doi.org/10.1098/rsta.2020.0209

[10] N. K. Ahmed, A. F. Atiya, N. El Gayar, and H. El-Shishiny, “An empirical comparison of machine learning models for time series forecasting,” *Econometric Reviews*, vol. 29, no. 5, pp. 594–621, Sep. 2010. https://doi.org/10.1080/07474938.2010.481556

[11] C. Deb, F. Zhang, J. Yang, S. E. Lee, and K. W. Shah, “A review on time series forecasting techniques for building energy consumption,” *Renewable and Sustainable Energy Reviews*, vol. 74, pp. 902–924, Jul. 2017. https://doi.org/10.1016/j.rser.2017.02.085

[12] Z. Duan, H. Liu, X. Lv, Z. Ren, and S. Junginger, “Hybrid position forecasting method for mobile robot transportation in smart indoor environment,” in *Proceedings of the 2019 4th International Conference on Big Data and Computing (ICBDC)*, Association for Computing Machinery, May 2019, pp. 333–338. https://doi.org/10.1145/3335484.3335508

[13] S. Gomez-Gonzalez, S. Prokudin, B. Scholkopf, and J. Peters, “Real time trajectory prediction using deep conditional generative models,” *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 970–976, Apr. 2020. https://doi.org/10.1109/LRA.2020.2966390

[14] C. C. Aggarwal, *Neural Networks and Deep Learning*. Springer, 2018. https://doi.org/10.1007/978-3-319-94463-0

[15] Z. Karevan and J. A. K. Suykens, “Spatio-temporal stacked LSTM for temperature prediction in weather forecasting,” *ArXiv*, no. 1811.06341, Nov. 2018.

[16] D. Neil, M. Pfeiffer, and S.-C. Liu, “Phased LSTM: Accelerating recurrent network training for long or event-based sequences,” *ArXiv*, no. 1610.09513, Oct. 2016.

[17] F. A. Gers, N. N. Schraudolph, and J. Schmidhuber, “Learning precise timing with LSTM recurrent networks,” *Journal of Machine Learning Research*, vol. 3, pp. 115–143, Aug. 2002.

[18] A. Paszke *et al.*, “PyTorch: An imperative style, high-performance deep learning library,” *arXiv*, no. 1912.01703, Dec. 2019.

[19] J. Y. Hong, S. H. Park, and J. G. Baek, “Segmented dynamic time warping based signal pattern classification,” in *22nd IEEE International Conference on Computational Science and Engineering and 17th IEEE International Conference on Embedded and Ubiquitous Computing, CSE/EUC 2019*, New York, NY, USA, Aug. 2019, pp. 263–265. https://doi.org/10.1109/CSE/EUC.2019.00058

Anete Vagale received an M. Sc. in Intelligent Robotic Systems from Riga Technical University, Latvia, in 2017.
She is currently a PhD candidate in autonomous ship technology, working at the Cyber-Physical Systems Laboratory at NTNU – Norwegian University of Science and Technology in Aalesund, Norway. In her research, she is focusing on safety evaluation of autonomous surface vehicle algorithms for path planning and collision avoidance in congested waters. Her research interests include autonomous shipping, artificial intelligence and robotics.
E-mail: anete.vagale@ntnu.no
ORCID iD: https://orcid.org/0000-0003-1620-2742

Luīze Šteina received a BSc in Intelligent Robotic Systems from Riga Technical University, Latvia, in 2021. Now, she is in her first year of MSc studies in the same program.
Currently, she is also working at Robotic Solutions in Riga, Latvia. As a researcher, her work involves the development of autonomous mobile robots, working on robot functionality and simulation. Her research interests include robotics and machine learning.
E-mail: luize.steina@edu.rtu.lv

Valters Vēciņš received an MSc in Intelligent Robotic Systems from Riga Technical University, Latvia, in 2019.
He is currently working as a Research Assistant at RTU - Riga Technical University, a researcher at Robotic Solutions and as a programmer at asya.ai. His research interests include mobile robots and artificial intelligence.
E-mail: valters.vecins@rtu.lv