LSTM Rollout Curriculum Using Double Pendulum

Reinis Freibergs[1], Ēvalds Urtāns[1,*], Ansis Ēcis[2], and Henrik Gabrielyan[2]

[1]Riga Technical University, Riga, Latvia
[2]University of Latvia, Riga, Latvia
Email: reinis.freibergs@rtu.lv(R.F.), evalds.urtans@rtu.lv (E.U.), ansis.ecis@lu.lv(A.E.), henriquegabrielyan@gmail.com (H.G.)
[*]Corresponding author
Manuscript received May 30, 2023; revised June 20, 2023; accepted July 5 2023; published February 4, 2024

***Abstract*—In this work, we model a double pendulum system with deep neural networks based on a data set generated from video recordings. For comparison, a similar model is made by describing the system with differential equations. Actually compared are the capabilities of both models in predicting the next 2s of double pendulum motion using information about the previous second. In addition, both models are compared by their ability to make predictions in specific error margins. Results show that deep learning-based approaches give much better predictions, where the best deep learning-based model could predict the next 1.5s in a specified error margin, while the best differential equation-based one only 0.12s, all other metrics agree with this result as well.**

***Keywords*—curriculum learning, deep learning, differential equations, double pendulum, LSTM, teacher forcing**

I. Introduction

Differential equation modelling of mechanical systems is a typical way to approach problems in mechanical engineering. But modern engineering solutions allow one to create and manipulate large amounts of data because of the fact that data-driven methods, including neural networks, can be a valuable alternative to classical approaches. It is backed by statistics, which shows that in the last five years, the popularity of both deep neural networks altogether as well as specifically in the field of engineering has risen significantly. It also provides a way to switch the analysis domain in the system - while differential equation-based solutions generally use the physical parameters of the system, such as weight and material properties, as well as geometrical parameters, data-driven methods introduce more flexibility in this matter, such as using only visual information.

For the actual analysis and comparisons in this work, we use a well-known mechanical system, the double pendulum, because its states in time are simple to describe, but it has a well-known chaotic behaviour [1]. Our goal is to study the double pendulum system with experimental data and to make comparisons between various data-driven methods, more specifically deep learning, and methods based on differential equations and how well they can predict the pendulum motion.

For the deep learning part, we apply long- and short-term memory (LSTM) models, together with an implementation of Curriculum Learning [2] to modify how they are trained to achieve better long-term predictions. On the other hand, we describe the system with differential equations with various degrees of complexity, starting with a simple mathematical model and continuing with a physical one. To counteract the problem of lack of knowledge of the physical parameters of the pendulum, we use parameter grid search. The comparison aims to find the longest time a model can predict the motion of pendulum blobs within a specific error margin, measured with various metrics

II. Related Work

Previously, work has been done to find uses for deep learning considering problems in mechanics. A method to use deep neural networks has been proposed to solve ordinary and partial differential equations [3], providing an alternative to classical numerical methods. Raissia *et al.* have proposed incorporating both experimental data and knowledge about the systems that govern the equations into the deep learning method, known as PINNs (physics informed neural networks) [4]. Research has also been done considering uses for recurrent neural networks in the modelling of mechanical systems, by considering their data as a time series problem. An often used system is the double pendulum, because its states in time are simple to describe, yet it has a well-known chaotic behaviour [1]. Klinkachorn et al. have compared different machine learning methods to model a double pendulum system [5], and have found the solution involving recurrent neural networks to give the best predictions. A similar study has been done by D. Gannon and also found that an LSTM based architecture could successfully model the differential equations governing the double pendulum system [6]. Both aforementioned works look at differential equations that describe the system and compare how well they can be modelled, but an important aspect to consider is not only how well neural networks can learn the differential equation, but also how accurate the differential equation itself is in representing the real-life system as a mathematical model. This problem was identified, and a dataset was created by Asseman *et al.* [7], which contains information about a real-life double pendulum system, on which the further work here has been built. This dataset also provides a challenge considering the information about its physical properties - while the dataset contains information about the kinematics of the pendulum's motion, the only available physical properties are of the geometry, but not the masses of the system's parts, materials, weight distribution etc. But such a situation is possible where an already made system would need to be analyzed, for example, for diagnostical purposes, without access/ability to measure all physical properties necessary for mathematical modelling.

Another approach is model-free estimation of Lyapunov exponents of chaotic systems using reservoir computing, which utilizes high-dimensional dynamical systems to learn output weights from a limited time series, allowing the approximation of the ergodic properties of the original

system, as demonstrated through successful application to the Lorenz system and the Kuramoto-Sivashinsky equation [8]. In related work, researchers have proposed the use of recurrent neural networks to generate particle trajectories in classical molecular dynamics simulations, achieving energy-conserving dynamics with significantly longer time steps compared to traditional numerical integrators such as Verlet [9]. Methods have even been extended to work that explores the development of a neural network architecture inspired by human physical reasoning, enabling machine-assisted scientific discovery by leveraging representation learning and making predictions based on relevant parameters and conservation laws [10]. Not all research uses RNN or LSTM, and some also propose the use of a Convolutional Neural Network as a surrogate model to approximate the steady-state diffusion equation, offering significant computational speed-ups for simulations involving fast-diffusing chemical species, such as oxygen gradients in the retina, and discuss various loss functions and accuracy estimators for selecting the most suitable network for different applications [11]. Practical applications in related research have been studied for the logistics of ships [12] and router networking stabilization [13].

III. Methodology

A. Dataset

In 2019 a work was published in which IBM researchers released a dataset containing data on the kinematics of a double pendulum system, whose geometry is shown in Fig. 1 [7]. Their premise was that for a reliable test for various prediction methods considering chaotic systems a real-life benchmark was needed, but most works were using data from simulated origin, for example extracted from differential equations. The dataset contains 21 videos with oscillations of the double pendulum, each of which is about 40s long and contains around 17 500 frames. Each of the frames is processed with OpenCV and the coordinates of each of the pendulum blobs x_1, y_1, x_2, y_2 as also shown in Fig. 2, are extracted.

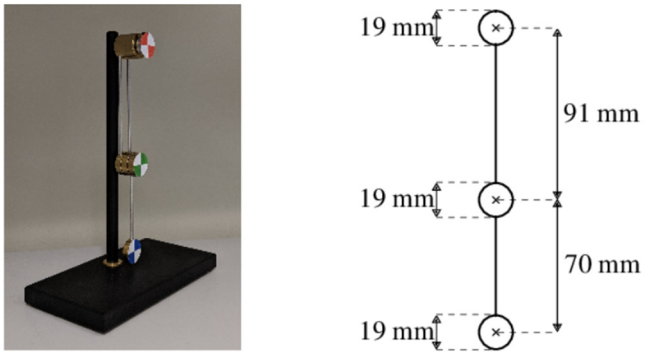

Fig. 1. The pendulum used for the experiment and its geometric parameters [7].

The parameters on the right-hand side in Fig. 1, describing the geometry of the system, are the only available ones, thus the rest of them, such as mass and moments of inertia, had to be approximated, which would provide a problem in any method based on these parameters. The coordinates of the pendulum bobs were converted into angles between its arms and the vertical axis, as shown in Fig. 2. A benefit of that is the possibility to use the gradients in further research involving the system's governing differential equations in the learning process as well, previously mentioned as physics-informed neural networks [4].

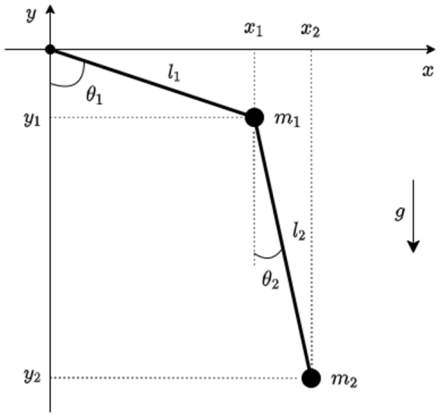

Fig. 2. Schematics describing the double pendulum system for creating the differential equations.

B. Limitations

To smooth out the noise, a Savitzky-Golay digital filter was used [14]. Then the dataset was split into train and test parts, corresponding to a ratio of 80:20 accordingly, from which the time series with a length of 400 steps were further extracted.

C. Metrics

Various metrics were implemented to objectively evaluate the results. The first is the mean absolute error, which is calculated for each time step of every sequence between the prediction and the ground truth.

$$MAE = \sum_{i=1}^{t=2s} |y_i - \hat{y}_i| \quad (1)$$

But in case of a bad prediction, the mean absolute error may be misleading due to a large accumulation of errors over each time step, because of that two additional metrics were used - the mean step count (MSC) and mean step sum (MSS).

$$MSC = i, \ when \ |y_i - \hat{y}_i| > \delta \quad (2)$$

The mean step count MSC measures the number of time steps it takes for the difference between the prediction and ground truth to reach a certain threshold δ. This metric helps to determine how close the prediction is to the initial period and whether it predicts the right direction and acceleration of the oscillations.

$$MSS = i, when \sum |y_i - \hat{y}_i| > \gamma \quad (3)$$

The sum of the mean steps MSS works similarly to the sum of the mean steps, the difference being that it measures the discrepancy in each time step and sums it up, measuring the time steps until a threshold is reached for the sum γ. It helps to distinguish how the accumulation of errors impacts

the predictions. For the evaluation in the experiments described further, the exact δ and γ values are $20\,rad/s$ and $8\,rad/s$ accordingly.

D. ODE Method

To model the system with differential equations an important feature of the dataset needs to be considered - the information about its dimensions and properties is scarce. Thus, many of the properties used in differential equations need to be approximated, potentially reducing the accuracy of their solutions. We model the system in two ways, the simplest one considers the system as a mathematical pendulum with point masses and no moments of inertia, air resistance, or other real-life factors. The other, the physical model, takes these factors into consideration, as far as possible with the limited knowledge about their properties.

The differential equations are written using Lagrange's equations for a system with $i = 1, 2, \dots, N$ degrees of freedom.

$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}_i}\right) - \frac{\partial K}{\partial q_i} + \frac{\partial P}{\partial q_i} = 0 \tag{4}$$

where: K – kinetic energy, P – potential energy, q – generalized coordinate.

Coordinates for the upper pendulum's part:

$$x_1 = l_1 sin\theta_1 \qquad y_1 = -l_1 cos\theta_1 \tag{5}$$

Coordinates for the lower pendulum's part:

$$\begin{aligned} x_2 &= l_1 sin\theta_1 + l_2 sin\theta_2 \\ y_2 &= -l_1 cos\theta_1 - l_2 cos\theta_2 \end{aligned} \tag{6}$$

Velocities can be attained by taking the derivatives of the coordinates with respect to time:

$$\begin{aligned} \dot{x}_1 &= l_1\dot{\theta}_1 cos\theta_1 \\ \dot{y}_1 &= l_1\dot{\theta}_1 sin\theta_1 \\ \dot{x}_2 &= l_1\dot{\theta}_1 cos\theta_1 + l_2\dot{\theta}_2 cos\theta_2 \\ \dot{y}_2 &= l_1\dot{\theta}_1 sin\theta_1 + l_2\dot{\theta}_2 sin\theta_2 \end{aligned} \tag{7}$$

Potential energy:

$$\begin{aligned} P &= m_1 g y_1 + m_2 g y_2 \\ &= -(m_1 + m_2) l_1 g\, cos\theta_1 - m_2 l_2 g\, cos\theta_2 \end{aligned} \tag{8}$$

Kinetic energy:

$$\begin{aligned} K &= \frac{m_1 v_1^2}{2} + \frac{m_2 v_2^2}{2} \\ &= \frac{m_1(\dot{x}_1^2 + \dot{y}_1^2)}{2} + \frac{m_2(\dot{x}_2^2 + \dot{y}_2^2)}{2} \end{aligned} \tag{9}$$

Then plugging both equations and their respective derivatives in the Lagrange's equation results in two differential equations describing the angle between the vertical axis for each of the pendulum's parts θ_1 and θ_2. The second differential equation based method takes into consideration the moments of inertia and air resistance of the pendulum. To accommodate air resistance, the Lagrange's equations can be supplemented with a dissipative term D in the form:

$$\frac{d}{dt}\left(\frac{\partial K}{\partial \dot{q}_i}\right) - \frac{\partial K}{\partial q_i} + \frac{\partial D}{\partial \dot{q}_i} + \frac{\partial P}{\partial q_i} = 0 \tag{10}$$

Often the dissipative term D is modelled with the Rayleigh's dissipation function, but that assumes a linear, velocity dependant friction. By calculating the average speed of the pendulum blobs and the respective Reynold's numbers in those speeds, it was concluded that air resistance in the observable velocity ranges in the experiment is not completely linear. Thus a more suitable option is the generalized dissipation function, which can model non-linear dissipation processes.

$$D = \frac{1}{n+1}\sum_i c_i v_i^{n+1} \tag{11}$$

where n depicts the order of the velocity dependence of the air resistance, c is the coefficient of friction, and v is the velocity. As was indicated by the calculated Reynolds numbers, we used a linearly dependent function for the first blob and a quadratic one for the second, giving the final form of the dissipation function:

$$D = \frac{c_1 v_1^2}{2} + \frac{c_2 v_2^2}{3} \tag{12}$$

The coefficients of friction c for both blobs were found with the Stoke's and drag equation accordingly.

E. LSTM Method

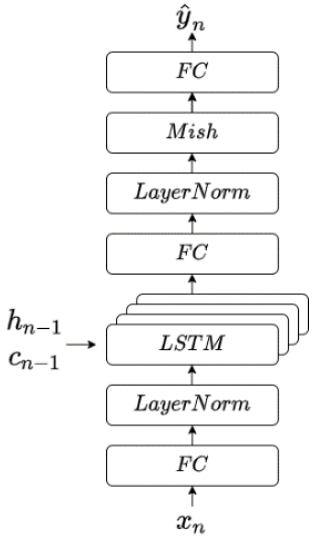

Fig. 3. A schematics describing the structure of the deep learning based model.

The other approach for the prediction of the pendulum's movement is based on deep learning methods. The models consist of stacked LSTM cells, whose outputs are put through linear layers. Research also shows that the data should be preprocessed with feed-forward layers before the

LSTM cells to simplify the temporal dynamics [15]. To reduce the loss at the initial time step learnable initial and hidden states c_0 and h_0 for the LSTM cells are used. LayerNorm is used after each feed-forward layer together with the Mish activation function [16, 17]. The structure of this model is illustrated in Fig. 3.

F. Training Methods

A typical training method for recurrent neural networks is teacher forcing, where the ground truth values are fed as model inputs in each time step [18]. But this method has problems, because in a real-life scenario the ground truth values are not available and the model has to work in a recurrent regime - only receiving the first step as input from the dataset and further using its own prediction from the previous step, which obviously differs from the training process if pure teacher forcing is used, resulting in accumulating errors.

So a potential workaround is training the model while incorporating its own previous predictions as inputs. To avoid large loss accumulation and thus unstable learning, a method called *Curriculum Learning* can be used, where, rather than starting with just the model's previous predictions, the input is gradually changed from ground truth to previously predicted values [2]. For the actual training, we combine both methods for some amount of time steps, which itself is a hyperparameter, as shown below in Fig. 4. So the training is started in teacher forcing mode and then switches to the recurrent mode with Curriculum Learning.

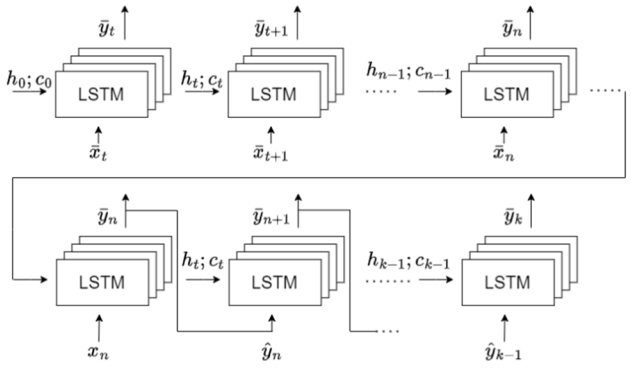

Fig. 4. A step scheme showing the transfer from teacher forcing initially to a generative part using the previous predictions as inputs.

G. Loss Function

The total loss function $\mathcal{L}$ used is the mean absolute error MAE, which then combines losses from both previously mentioned parts.

$$\mathcal{L} = \frac{1}{n(t_1)} \sum_{i=1}^{t_1=(2-\alpha)s} |y_i - \hat{y}_i| + \frac{1}{n(t_2)} \sum_{j=1}^{t_2=\alpha\, s} |y_j - \hat{y}_j| \tag{13}$$

where α is the length of the recurrent learning part and then $2-\alpha$ is the length of the part for teacher forcing, considering that the total prediction length is a step count corresponding to $2s$ of pendulum's motion.

IV. Results

Table I contains the best results from each type of model. The LSTM based models are grouped by the usage of the recurrent part, its length and whether the curriculum learning was applied to the recurrent part. The ODE models shown are the simplified mathematical model and the more complex physical model. Results from the LSTM based models were obtained with hyper-parameter grid-search, while a similar strategy was used to determine the best results from the differential equation-based models, with the difference being that ODE grid-searched parameters were the physical attributes m_1 and m_2.

Table 1. Comparison of ODE methods and LSTM methods

Method	Recurrent part	Length of rec. Part	Curriculum	MSC, steps	MSS, steps	MAE, rad/s
LSTM	+	50	+	600	79	0.0126
LSTM	+	50	-	444	56	0.0147
LSTM	-			190	41	0.0262
ODE physical	-			46.69	11.77	0.0341
ODE mathema-tical	-			47.63	11.73	0.0385

Fig. 5. illustrates the total loss in each epoch, where a clear convergence of both the test and train loss values can be observed with no noticeable overfitting.

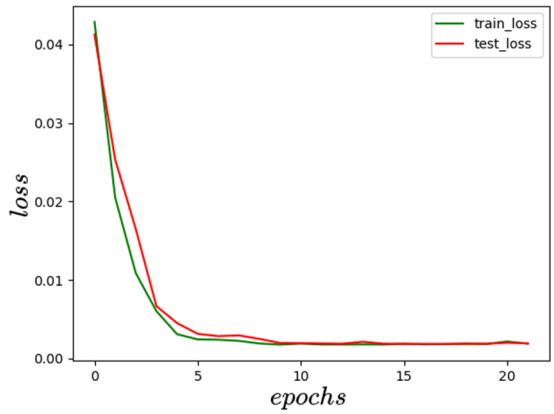

Fig. 5. Loss value for train and test set.

In the images below a comparison is illustrated for the prediction abilities of the best ODE model in Fig. 6 and the best deep neural network model in Fig. 7. Both predictions have used the same 1s long input sequence and predicted the next 2s or 800 steps. The upper and lower graphs in both images correspond to the first and second blob accordingly.

It can be visually seen that the predictions based on the LSTM model, shown in Fig. 7, are closer to the ground truth values and close to them throughout the $2s$ prediction period, whereas the ODE-based model, whose predictions are illustrated in Fig. 6, is capable of predicting the real values only in the early steps, while later diverging from the ground truth values. Data in the results Table 1 show the same conclusion. The best LSTM based model could predict 600 steps or $1.5s$ in the specified error margin $\delta = 20\, rad/s$, while the best ODE based model only 48 steps or $0.12s$. The MSS and MAE values also show a similar superiority of the LSTM based model, with the MSS values being

79.00 and 11.77 steps on average accordingly and the MAE being $0.0126\,rad/s$ and $0.0341\,rad/s$ accordingly. This means that LSTM based model's predictions could stay closer to the ground truth values for more steps and provide a lower average error overall.

Another comparison can be made between the prediction capabilities between the same types of models. Major improvements in the LSTM model predictions can be gained by adding the recurrent learning part with a curriculum-based approach, indicated by all metrics.

Downsampling of the signal provided worse results in all test cases, but could be considered if a solution with more limited computational resources would be of interest.

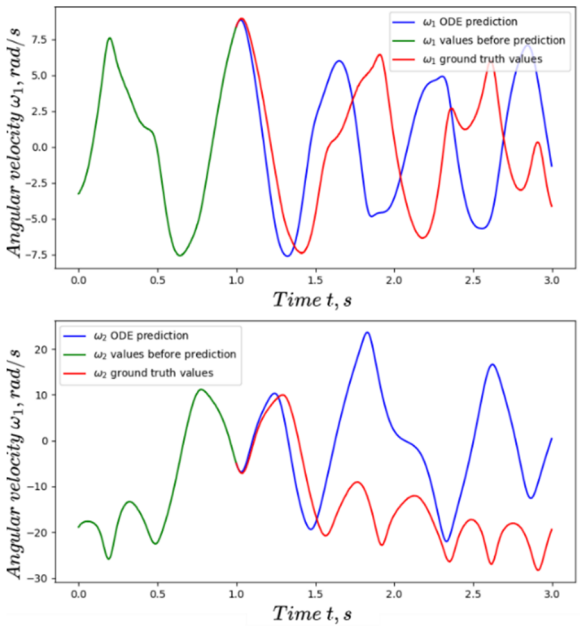

Fig. 6. Comparison between the ODE predictions and the ground truth values for the angular velocities ω_1 and ω_2 in a prediction span of $2s$.

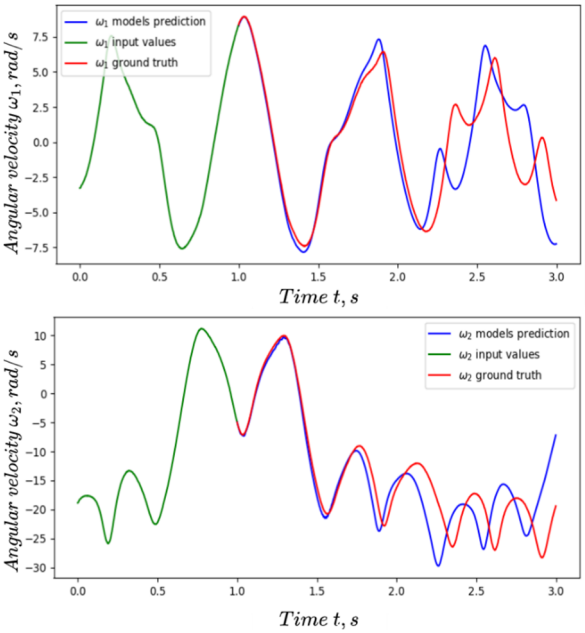

Fig. 7. Comparison between the LSTM predictions and the ground truth values for the angular velocities ω_1 and ω_2 in a prediction span of $2s$.

V. Further Research

In our work only experimental data was used for the training of the deep learning-based model, but, as outlined previously, a secondary part could be provided by using the differential equations themselves, similarly as in physics-informed neural networks [3]. In that case, more physical meaning of the system could be learned by the model, potentially improving long-term prediction abilities. As the deep learning based model showed good prediction capabilities, it could be adapted for some systems for diagnostic purposes, as the loss values could be used for determining a change in the system. That would also involve creating a procedure for data gathering. Additionally, further research could be done considering alternatives to the LSTM cell with the goal of improving the long-term prediction abilities. Some of these alternatives could include transformer-based architectures [19] or Phased-LSTM [20].

VI. Conclusion

In this study, we rigorously compare the prediction performance of LSTM and ODE-based models, with a specific focus on an experimentally extracted data set from a double pendulum system. Our research reveals that despite the scarce description of the system's physical parameters, a deep learning-based approach with LSTM consistently produces predictions closer to the ground-truth values throughout a 2 second prediction period. On the contrary, the ODE-based model tends to diverge from these ground truth values. This emphasis on the LSTM model's capacity to maintain closer proximity to the ground truth values over an extended duration translates to lower overall average errors compared to its ODE counterpart. Further comparative analysis among the deep learning models unveils a significant boost in prediction ability across all evaluated metrics when curriculum learning is incorporated into the model's training. This improvement is markedly apparent in contrast to other modes such as autoregression mode or a combination of auto-regression mode with a recurrent part without curriculum learning, underscoring the advantages of incorporating such a learning strategy in our model's training

Conflict of Interest

The authors declare no conflict of interest.

Author Contributions

Author R. Freibergs did the initial research, coding and evaluation of the models as well as writing the manuscript. Author Ē. Urtāns provided valuable guidance and ideas as the advisor. A. Ēcis contributed in describing the system with differential equations and H. Gabrielyan contributed to the literature analysis.

Funding

This research was funded by Riga Technical University and supported by High Performance Cluster of Riga Technical University.

References

[1] R. Levien and S. Tan, "Double pendulum: An experiment in chaos," *American Journal of Physics*, vol. 61, pp. 1038-1044, Nov. 1993.

[2] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, "Scheduled sampling for sequence prediction with recurrent neural networks," *NIPS*, pp. 1171-1179, Dec. 2015.

[3] I. Lagaris, A. Likas, and D. Fotiadis, "Artificial neural networks for solving ordinary and partial differential equations," *IEEE Transactions on Neural Networks*, vol. 9, pp. 987-1000, Sep. 1998.
[4] M. Raissi, P. Perdikaris, and G. Karniadakis, "Physics informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational Physics*, vol. 378, pp. 686-707, Feb. 2019.
[5] S. Klinkachorn and J. Parmar. (2019). Evaluating current machine learning techniques on predicting chaotic systems cs. [Online] Available: https://cs229.stanford.edu/proj2019spr/report/38.pdf
[6] D. Gannon. (Feb. 2021). Deep learning with real data and the chaotic double pendulum. [Online]. Available: https://www.researchgate.net/publication/349681865_Deep_Learning_with_Real_Data_and_the_Chaotic_Double_Pendulum
[7] A. Asseman, T. Kornuta and A. Ozcan. (Sep. 2018). Learning beyond simulated physics. [Online] Available: https://openreview.net/pdf?id=HylajWsRF7
[8] J. Pathak, Z. Lu, B. R. Hunt, M. Girvan, and E. Ott, "Using machine learning to replicate chaotic attractors and calculate lyapunov exponents from data," *Chaos*, vol. 27, pp. 102-121, 2017.
[9] J. C. S. Kadupitiya, G. Fox, and V. Jadhao, "Solving newton's equations of motion with large timesteps using recurrent neural networks based operators," *Machine Learning: Science and Technology*, vol. 3, No. 2, Apr. 2020.
[10] R. Iten, T. Metger, H. Wilming, L. del Rio, and R. Renner, "Discovering physical concepts with neural networks," *Physical Review Letters*, vol. 124, pp. 10-508, 2018.
[11] J. Q. Toledo-Mar'in, G. C. Fox, J. P. Sluka, and J. A. Glazier, "Deep learning approaches to surrogates for solving the diffusion equation for mechanistic realworld simulations," *Frontiers in Physiology*, vol. 12, Jun. 2021.
[12] N. Sun, Y. Wu, X. Liang, and Y. Fang, "Nonlinear stable transportation control for double-pendulum shipboard cranes with ship-motion-induced disturbances," *IEEE Transactions on Industrial Electronics*, vol. 66, pp. 9467-9479, 2019.
[13] J. K. Adamu, M. F. Hamza, and A. I. Isa, "Performance comparisons of hybrid fuzzy-lqr and hybrid pid-lqr controllers on stabilizing double rotary inverted pendulum," *Journal of Applied Materials and Technology*, vol. 1, pp. 71-80, Mar. 2020.
[14] A. Savitzky and M. J. E. Golay, "Smoothing and differentiation of data by simplified least squares procedures," *Analytical Chemistry*, vol. 36, pp. 1627–1639, 1964.
[15] D. Hafner. (2017). Tips for training recurrent neural networks, Blog post. [Online]. Available: https://danijar.com/tips-for-training-recurrent-neural-networks/
[16] J. Ba, J. Kiros, and G. E. Hinton, "Layer normalization," ArXiv, vol. abs/1607.06450, Jul. 2016.
[17] D. Misra, "Mish: A self regularized non-monotonic neural activation function," *CoRR*, vol. abs/1908.08681, 2019.
[18] I. Goodfellow, Y. Bengio, and A. Courville. (2016). *Deep Learning*. MIT Press. [Online]. Available: http://www.deeplearningbook.org.
[19] A. Vaswani, N. M. Shazeer, N. Parmar *et al.*, "Attention is all you need," *NIPS*, pp. 600-610, Dec. 2017.
[20] D. Neil, M. Pfeiffer, and S.-C. Liu, "Phased LSTM: Accelerating recurrent network training for long or event-based sequences," *NIPS*, pp. 3889-3898, Dec. 2016.