Benchmarking imputation methods on the implementation of GRU to forecast Mendota lake's epilimnetic phosphorus.

Gabriel Juri A.

Abstract

To manage phosphorus lake concentration, it is crucial to model and forecast it thought the time, being the gated recurrent neural networks (GRU) an outstanding approach. Nevertheless, one of the main challenges in time series forecasting is to deal with long and consecutive nan values. Consequently, this work aims to benchmark four imputation method, three naïve approaches and one advanced, to then compare the signal reconstruction and GRU performance. The linear and quadratic methods have the worst performance (r^2: 45 and 31%). KTS and the linear approach has the best forecast prediction capability, differing only by 1.46% between them.

1. Introduction:

Eutrophication is a gradual and natural process in which the primary production in freshwater ecosystem increases, and generally, it evolves over ages as a result of the accumulation of nutrients needed by primary producers (Qin et al., 2013). Since phosphorus is the main factor that led the algal growth in freshwater ecosystems, it is the principal cause of lake water detriment, arise of water health hazards, economic losses, and a decrease in ecosystem services (Carpenter., 1981; Noges., 2009; Sinha et al., 2017; Nazari-Sharabian et al., 2018; Kalsido & Berhanu., 2021).

Consequently, the monitoring and control of the main driver of the eutrophication has emerged as a fundamental objective in the management of lakes around the world. To achieve this goal, it is crucial to model and predict the phosphorus dynamic (Jeppesen et al., 2012; Katsev., 2016; Schindler et al., 2016). Giving the temporal nature of the features and the phosphorus, we are dealing with 'Time series forecasting' problem, in which Gated recurrent neural networks (GRU-RNN, or simply GRU) have had outstanding success in sequential modelling and have achieved high prediction accuracy in the water quality prediction of lakes (Yan et al.,2021; Yu et al., 2022).

Beyond the prediction capability of GRU models, one of the key issues in time series forecasting (E.g.: in clinical, and environmental data) is dealing with gaps of consecutive missing data (nan values) produced by machine failure, routine maintenance and/or human errors. If one encounter this situation, the first step before forecasting, or the application of any analysis method, is the imputation of those missing value followed by the feeding of the data into the Gated recurrent neural networks. There are several imputation methods that one could use to fill those missing instances, however the imputation quality (signal recreation), and the forecast performance can differ upon the selected method (Li & Xu., 2019; Samal et al., 2021).

Consequently, the present work aims to use different imputation method to fill induced missing values in data's features, to then feed the data into the GRU model and compare the performance, and finally choose the best imputation method.

2. Methods:

2.1 Data.

To benchmark the imputation methods on the GRU-forecasting of the Mendota's epilimnetic phosphorus, I use Mendota's Lake data provided by Hanson et all (2020), which contains 7200 instances between 1995 and 2015, 10 features, and the target variable, epilimnetic phosphorus (gL^{-1}). The features are: Epilimnetic and hypolimnetic temperature (°C) and volume (m^3), stratification (Boolean), thermocline depth (m), river discharge (m^3d^{-1}), air temperature (°C), Short Wave radiation (Wm^{-2}), precipitation (m), wind speed (ms^{-1}), and phosphorus load (gd^{-1}) (Hanson et al 2020).

2.2 Imputation methods:

Three common and naïve impute approaches are implemented from the in-build pandas functions [1]: linear, quadratic, and mean imputation. As a more advance approach, the filter Kalman smoother [2] (KS) is selected since it has showed the most effective imputation method in some non-environmental studies (Turicchi et al., 2020). To use and compare the different imputation methods, it is necessary to induce artificial missing values in each feature. The latter is done by selecting a random position in the time series of each feature and a random length (10 to 15% of the feature values) of consecutive nan values (nan window), then each imputation method is applied to fill the generate nan's window.

[1]: https://pandas.pydata.org/docs/reference/api/pandas.DataFrame.interpolate.html
[2]: https://github.com/cerlymarco/tsmoothie

To compare the reconstruction performance of each method the coefficient of determination (r^2) method is applied between the original and imputed data.

Finally, since the Kalman smoother (KS) method has several hyperparameters to tune in order to impute missing data, as previous step, KS will be trained in order to find the best combination of hyperparameter (**Table 1**) that can capture the nature of each feature time series. According to the latter, a random grid search is implemented on KS for each feature. The latter is applied by using, as a metric, the mean absolute error between the predicted and observed values, of each feature, towards the time series section where the missing values are located.

Table 1. Kalman smoother (KS) hyperparameters and range on value for each of them

Hyperparameters	Values range
Number of seasons	0 to 30
Number of long seasons	0 to 1000
Component noise	0.001 to 1.5
Component	'level', 'level_trend', 'level_season', 'level_trend_season', 'level_longseason','level_trend_longseason', 'level_season_longseason', 'level_trend_season_longseason'

2.3 Recurrent neural network (RNN):

The Recurrent neural network (RNN) is derived from the Artificial Neural Network (ANN) theory, and it is based on the memory of the experience. The RNN takes a previous computation or information as the input that will influence the current output, giving a network memory function. From the unfold structure of RNN (**Figure 1**), with one hidden layer s_t, we can see that the nonlinear transformations of the inputs done by the hidden layer s_t, at time t, will produce the corresponding wights (U) by combining it with the previously hidden layer output (s_{t-1},). The latter will have the output weights (**V**) and outcome of the current hidden layer at time t (O_t) (Figure 2) (Hanson et al., 2020; Ren et al., 2020).

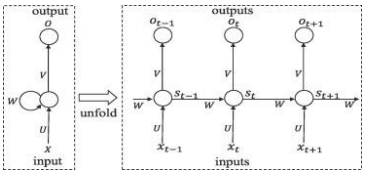


Figure 1. Unfold neural network of RNN with one hidden layer (From Ren et al., 2020).

The hidden representation and the output can be computed using the transition function and the output function. As mentioned above, the Gated Recurrent Unit (GRU) structure is considered because it memorizes short-term and long-term computations, produced by its two gate components, update Z_t, and Reset r_t^j gates, that determine the amount of information to be updated and the amount of last hidden information to be deleted, respectively. Then, the update and reset gate information are scaled and added to the activation function *tanh* (Huang et al., 2019; Ren et al., 2020).

Previous the GRU implementation, the variables in the data are scaled in a range from 0 to 1 using the MinMax function provided by Sklearn. After the data scaling, the features are reshaped by using data chunks with 100-time steps. In addition, to avoid over- or under-fitting towards a specific section of the timeseries, the hyperparameters are tuned by implementing a 6-fold cross validation, considering 85% of the data as train and 15% as test set, hence at each fold the training is done with 71% of the data and the validation with the 17%. The explored hyperparameters, and the used factor to tune them can be found in the following table:

Table 2. Explored GRU's hyperparameters, its range of values, and the increasing factor used to change the values.

Hyperparameters	Values range	Increasing factor
Number of hidden layers	up to 2	-
Number of hidden neurons (**NN**)	10 to 200	**NN** * 2
Dropout (**D**)	0 to 0.5	**D** + 0.1
Activation functions	Tanh, Sigmoid, Relu, Elu	-
Optimizer	SGD, ADAM, RMSprop	-

Learning rate (**LR**)	0.001 to 1e-7	**LR** * 0.1
Epochs (**E**)	0 to 100	Early stopping
Batch size (**B**)	16 to 512	**B** * 2

To optimize the GRU model towards an optimal solution, the implemented loss function $\boldsymbol{L_{RNN}}$ minimizes the root-mean-square error (RMSE) between the predicted and the observed values at each time step.

Finally, the chosen GRU architecture is derived from the reference dataset (**Ref**) and applied to the imputation method dataset. The latter simulate a transfer learning, assuming that GRU**ref** can be consider as the reference model of phosphorus prediction, and hence can be taken to improve the performance of the forecasting problem

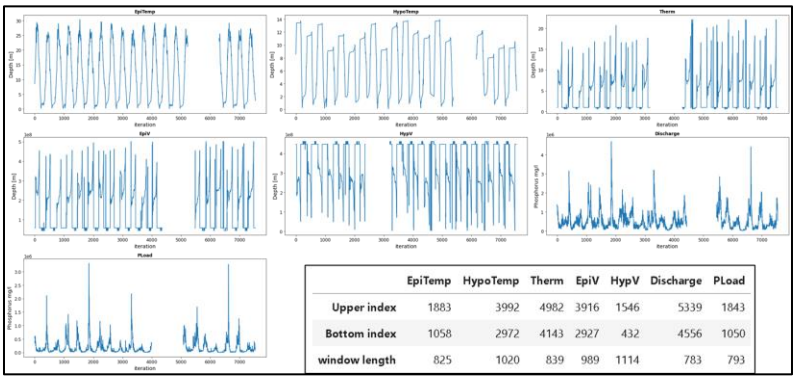
2.4 GRU Model evaluation:

To compare the GRU model performance for the different imputation methods the following metrics are considered: root mean squared error (RMSE) and mean absolute error (MAE), and the coefficient of determination (r^2) between the predicted and the observed data. The predicted phosphorus is the result of the average of the obtained GRU model for fold of the imputation method.

3 Results and discussions:

3.1 Missing data and imputation method:

The consecutive missing data values artificially insert in each feature can be found in **Figure 2**. In addition, the information of the index position and the window length of missing data it is also provided.



	EpiTemp	HypoTemp	Therm	EpiV	HypV	Discharge	PLoad
Upper index	1883	3992	4982	3916	1546	5339	1843
Bottom index	1058	2972	4143	2927	432	4556	1050
window length	825	1020	839	989	1114	783	793

Figure 2. Features' time series (Figures) induced with nan values generated by considering a random consecutive length and position (Table).

From previous results it is possible to observe that the insertion of missing data described before are achieved as intended, hence having random missing data position and length for each feature.

Regarding the imputation method output one can compare the two best feature data reconstruction as it is expressed in **Figure 3**. From these results Kalman smoother imputation method outperforms the data reconstruction compared to the naïve approaches.

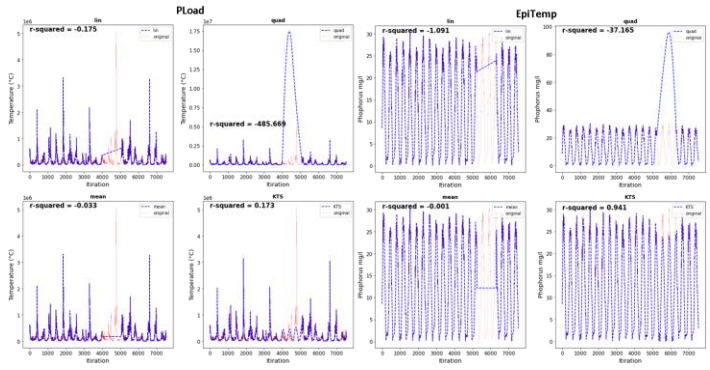


Figure 3. Imputation visualization and its corresponding R^2 of the imputed section.

3.2 GRU-model:

3.2.1 GRUref architecture:

Based on the best cross-validation result (**Figure 4**), the chosen GRU's structure has a simple architecture with three layers: the input layer with input shape of 100 x 12 (time steps and input variables, respectively), one hidden layer with 100 neurons, and a dense layer, as the output, with one output neuron. The hidden activation function is represented by RELU combined with a dropout of 0.15. The optimization of the GRU were done using Adam with a learning rate of 1e-5. The batch size was set at 64, and the epoch number at each fold determined by early stopping which monitor the validation loss at each fold.

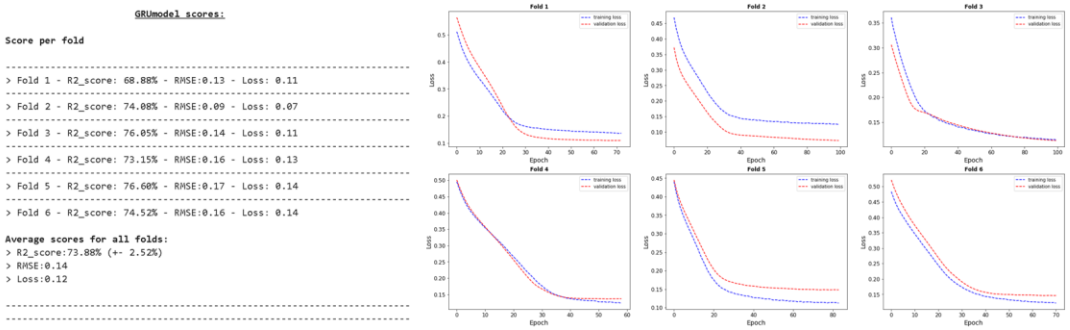
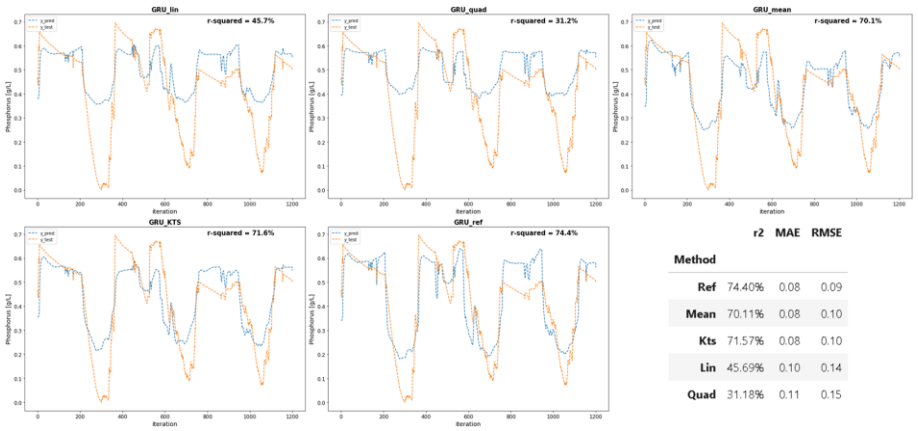


Figure 4. Cross-validation scores (R2, RMSE, Loss) for each fold (Left Table), and validation/training loss vs number of epochs (Right Figure).

The optimum epoch values that would give a model with the lowest over/underfitting correspond to 100 epochs for the first three folds, 35 epochs for the fold n°6 and fold n°4, and finally 25 epochs for the fold n°3.

3.2.2 GRU performance:

Compared to the reference GRU performance (Ref) (**Figure 5)**, the linear and quadratic imputation method have the worst performance, and KTS has the best forecast prediction capability. However, the linear approach only differed by 1.46% compared to KTS and 3.17% to Ref, in terms of r^2.



Method	r2	MAE	RMSE
Ref	74.40%	0.08	0.09
Mean	70.11%	0.08	0.10
Kts	71.57%	0.08	0.10
Lin	45.69%	0.10	0.14
Quad	31.18%	0.11	0.15

Figure 5. GRU performance on the test set for each method: Time series plots (Figures), and metrics performance (R2, RMSE, Loss) (Table).

The performance of KTS can be related to its good signal reconstruction, able to reproduce the general trend of the data. The competitive performance of the mean imputation approach (compare to KTS) could be associated with the good capability of GRU model to model the average behavior of timeseries data. The poor performance of the linear and quadratic model may be linked to the important disruption, that could produce this imputation method, between the epilimnetic phosphorus and its features. Finally, the higher explained variability obtained by KTS is desired, however the extra time/effort expended to compute the KTS imputation, and to build/train the KTS method is not a trivial variable to be consider, especially in machine learning task where, due to the complexity of the modeled problem, local but high-performance solutiond are the final goal (McLeay et al., 2021).

Project repository:

The code and data used to generate this report is available at:

https://github.com/Gjuri/2022_ML_Earth_Env_Sci/tree/main/Project

References

Carpenter, S. R. (1981). Submersed vegetation: an internal factor in lake ecosystem succession. The American Naturalist, 118(3), 372-383.

Hadjisolomou, E., Stefanidis, K., Papatheodorou, G., & Papastergiadou, E. (2016). Assessing the Contribution of the Environmental Parameters to Eutrophication with the Use of the "PaD" and "PaD2" Methods in a Hypereutrophic Lake. International journal of environmental research and public health, 13(8), 764.

Hanson, P. C., Stillman, A. B., Jia, X., Karpatne, A., Dugan, H. A., Carey, C. C., & Kumar, V. (2020). Predicting lake surface water phosphorus dynamics using process-guided machine learning. Ecological Modelling, 430, 109136.

Hirayama, A. 2003. Development of decision-making tools for eutrophic lakes. UNESCO–EOLSS. UNESCO.

Huang, Z., Yang, F., Xu, F., Song, X., & Tsui, K. L. (2019). Convolutional gated recurrent unit–recurrent neural network for state-of-charge estimation of lithium-ion batteries. IEEE Access, 7, 93139-93149.

Janse, J. H., Aldenberg, T., & Kramer, P. R. G. (1992). A mathematical model of the phosphorus cycle in Lake Loosdrecht and simulation of additional measures. Hydrobiologia, 233(1), 119-136.

Jensen, J. P., Pedersen, A. R., Jeppesen, E., & Søndergaard, M. (2006). An empirical model describing the seasonal dynamics of phosphorus in 16 shallow eutrophic lakes after external loading reduction. Limnology and Oceanography, 51(1part2), 791-800.

Jeppesen, E., Søndergaard, M., Lauridsen, T. L., Davidson, T. A., Liu, Z., Mazzeo, N., & Meerhoff, M. (2012). Biomanipulation as a restoration tool to combat eutrophication: recent advances and future challenges. Advances in ecological research, 47, 411-488.

McLeay, A. J., McGhie, A., Briscoe, D., Bi, Y., Xue, B., Vennell, R., & Zhang, M. (2021, December). Deep Convolutional Neural Networks with Transfer Learning for Waterline Detection in Mussel Farms. In 2021 IEEE Symposium Series on Computational Intelligence (SSCI) (pp. 01-08). IEEE.

Ngatia, L., & Taylor, R. (2018). Phosphorus eutrophication and mitigation strategies. In Phosphorus-Recovery and Recycling. IntechOpen.

Noges, T. (2009). Relationships between morphometry, geographic location and water quality parameters of European lakes. Hydrobiologia, 633(1), 33-43.

Paerl, H. W., Hall, N. S., & Calandrino, E. S. (2011). Controlling harmful cyanobacterial blooms in a world experiencing anthropogenic and climatic-induced change. Science of the total environment, 409(10), 1739-1745.

Peters, D. P., Havstad, K. M., Cushing, J., Tweedie, C., Fuentes, O., & Villanueva-Rosales, N. (2014). Harnessing the power of big data: infusing the scientific method with machine learning to transform ecology. Ecosphere, 5(6), 1-15.

Samal, K. K. R., Babu, K. S., & Das, S. K. (2021). Multi-directional temporal convolutional artificial neural network for PM2. 5 forecasting with missing values: A deep learning approach. Urban Climate, 36, 100800.

Smith, V. H. (2003). Eutrophication of freshwater and coastal marine ecosystems a global problem. Environmental Science and Pollution Research, 10(2), 126-139.

Smith, V. H., & Schindler, D. W. (2009). Eutrophication science: where do we go from here?. Trends in ecology & evolution, 24(4), 201-207.

Turicchi, J., O'Driscoll, R., Finlayson, G., Duarte, C., Palmeira, A. L., Larsen, S. C., ... & Stubbs, R. J. (2020). Data imputation and body weight variability calculation using linear and nonlinear methods in data collected from digital smart scales: simulation and validation study. JMIR mHealth and uHealth, 8(9), e17977.

Qin, B., Gao, G., Zhu, G., Zhang, Y., Song, Y., Tang, X., & Deng, J. (2013). Lake eutrophication and its ecosystem response. Chinese Science Bulletin, 58(9), 961-970**.**

Yan, J., Liu, J., Yu, Y., & Xu, H. (2021). Water quality prediction in the luan river based on 1-drcnn and bigru hybrid neural network model. Water, 13(9), 1273.

Yang, X. E., Wu, X., Hao, H. L., & He, Z. L. (2008). Mechanisms and assessment of water eutrophication. Journal of zhejiang university Science B, 9(3), 197-209.

Yates, K. L., Bouchet, P. J., Caley, M. J., Mengersen, K., Randin, C. F., Parnell, S., ... & Sequeira, A. M. (2018). Outstanding challenges in the transferability of ecological models. Trends in ecology & evolution, 33(10), 790-802.

Yu, J. W., Kim, J. S., Li, X., Jong, Y. C., Kim, K. H., & Ryang, G. I. (2022). Water quality forecasting based on data decomposition, fuzzy clustering and deep learning neural network. Environmental Pollution, 303, 119136.