Tackling Imputation Across Time Series Models Using Deep Learning and Ensemble Learning

Muhammad Saad
Department of Electrical and Computer Engineering
University of Waterloo
Ontario, Canada
m32saad@uwaterloo.ca

Lobna Nassar
Department of Electrical and Computer Engineering
University of Waterloo
Ontario, Canada
lnassar@uwaterloo.ca

Fakhri Karray
Department of Electrical and Computer Engineering
University of Waterloo
Ontario, Canada
karray@uwaterloo.ca

Vincent Gaudet
Department of Electrical and Computer Engineering
University of Waterloo
Ontario, Canada
vcgaudet@uwaterloo.ca

***Abstract*—Missing data are commonly found in time series datasets. These missing elements are usually a hurdle in utilizing the datasets in prediction or forecasting, making imputation of those missing values imperative. Due to the non-linear dependencies between the current and previous values, imputation remains a challenging task. Conventional methods such as averaging, deletion or filling with the last observed value add bias to the data and are therefore inefficient. Since different time series showcase varying characteristics, figuring out which imputation method works best for the respective time series is essential. In this work, seven different deep learning (DL) imputation methods are examined along with three machine learning (ML) ensembles. To enable a recommendation of the best imputation method for each time series type, the imputation models are tested using the four main types of time series: trend (T), seasonal (S), combined trend and seasonal time series (T&S) and random (R) time series. Results indicate that the Gated Recurrent Unit (GRU) neural networks are, in general, the best for missing values imputation with varying complexity based on the time series type. For example, it is found that the residual GRU is recommended for the trend and seasonal time series while the GRU is recommended for the combined type. Conversely, all tested DL imputation models can be used with the random time series type. In addition, the considered ML ensembles do not perform as high as the DL models with all tested types of times series.**

***Index Terms*—Missing value Imputation, Deep Neural Networks, Gate Recurrent Unit (GRU) Neural Networks, Machine Learning Ensemble.**

I. Introduction

Time series monitoring plays an essential role across domains, whether it be manufacturing [1], biology [2] or even social sciences [3]. Even though these time series vary from one another, one common problem across different disciplines is the missing values. These missing values may be caused by various factors ranging from human mistakes to device or equipment noise/malfunctioning [4], [5].

Since missing values hinder the utilization of a dataset for prediction or forecasting, researchers have developed multiple methods for imputation in order to mitigate the issue [6]–[8]. These methods range from the most basic ones, forward and backward filling [9] or averaging [10], to more complex ones such as using ANN to predict the missing values [11]–[13].

The work is supported by Grant's from NSERC CRD Funding and Loblaws Research Chair in Artificial Intelligence.

Each method has its own merits and demerits. While basic models provide quick and easy results at the expense of reliable forecasting, complex models, such as neural networks, provide better results but require higher computational resources and longer execution time.

In this work, four types of time series are considered: (1) Trend (T) time series, (2) Seasonal (S) time series, (3) Combined Trend and Seasonal (T&S) time series, and (4) Random (R) time series. Seven different DL models and three ML ensembles are applied to fill the missing values in each of these time series. The utilized imputation models range from the most basic to the highly complex ones, thereby allowing coverage of the spectrum of complexity. The performance analysis of the imputation methods helps in finding the best model for each time series type. The utilized imputation models are: (1) Simple DL models including the Long Short-Term Memory (LSTM) and the Gated Recurrent Unit (GRU). (2) Compound DL models which involve the Residual GRU, LSTM-GRU, Deep GRU, LSTM-Deep GRU, and Deep LSTM-GRU. (3) ML ensemble models including the Linear Regression (LR), Support Vector Regressor (SVR), and Gradient Boosting (GB) ensembles.

The experimentation results show that the GRU based neural networks provide promising results in terms of imputation. However, the complexity of the model varies across different types of time series. The residual GRU is recommended for the trend and seasonal time series, simple GRU is recommended for the combined type while the random time series can be imputed by any of the tested DL models. It is also found that the ML ensembles do not perform better than any of the tested DL models with any of the considered time series types.

The remainder of this paper is organized as follows: Section II provides a literature review on the time series imputation. In Section III the datasets and imputation framework are discussed in detail. Section IV presents the utilized imputation models and their configuration. In Section V the experiments and results are discussed. Finally, Section VI reports the conclusion.

II. Literature Review

Since missing values imputation is an active research area, various methods exist for dealing with it. The most direct way to tackle this issue is to discard incomplete or empty records.

Common examples are listwise deletion, which removes all instances with at least one missing value, or pairwise deletion, which removes an instance if the used variables contain missing values [14]. Unfortunately, these methodologies don't only introduce bias in the dataset but also reduce the amount of data available, thereby providing incorrect forecasting [15]–[18]. Additionally, a method like linear interpolation is considered primitive. Despite the fact that this method fits a curve to the given dataset and imputes the missing values using local interpolations, this method fails to take into consideration the dependencies of features over time [19].

On the other hand, methods like SARIMA for seasonal [20] and ARIMA for non-seasonal time series [21] are auto-regressive methods yet their performance deteriorates when the dataset contains consecutive missing values. Nevertheless, better results might be attained by combining ARIMA and Kalman filter in the state space model [22].

Furthermore, imputation methods using K-nearest neighbors are also applicable for time series. In this approach, the method identifies k instances which are most similar to the missing value. Then a predefined rule (e.g. weighted average [23]) or kernel function (e.g. exponential kernel [24]) is applied to impute the values. Besides that, there are certain pattern matching approaches as in [25], but these approaches work best only in certain cases. Moreover, research is also conducted in the tensor latent factorization as proposed in [26]–[29].

Many researchers have also utilized Recurrent Neural Networks (RNN) to impute the missing values [13], [30]–[32]. Che et al. in [13] proposed a modified GRU, GRU-D, that fills the missing values in a health care dataset. The underlying assumption of the model is that the missing variable can be represented as a combination of the last available value and the global mean. The GRU-D model has aided in harnessing the power of the RNNs [33] and the informativeness of the missing patterns in the data. The model uses GRU [34] and two representations of missing pattern, which are masking and the time interval. Masking informs the model whether the input is present or not whereas the time interval illustrates the input observation patterns. The model is not only capable of capturing long term temporal dependencies of time series but also utilizing the missing patterns to improve the prediction results. GRU-D performed well on the health care dataset but it might exhibit limitations on other datasets.

In this work, the imputation results of various models are compared and a recommendation framework is provided to decide the most optimal strategy for each type of time series under consideration. Since the models use RNN and its variants, it is able to cater the long-term dependencies. Furthermore, in this work the missing rate of 50% is considered which implies that there are consecutive missing values in the dataset.

III. Methodology

In order to gauge the imputation method, an aggregated average error metric (AGE) is calculated. The aggregated measure is the simple average of the Root Mean Squared Error (RMSE) and the Mean Absolute Error (MAE).

The input to the RMSE and MAE measures is the actual original values and the imputed ones. Imputation using the linear function is considered as the baseline for each of the tested time series. This setup requires a complete time series dataset to allow for generating the missing values and have the actual values of those missing ones. Since the availability of a complete time series is limited, synthetic data is used for all time series under consideration and noise of 10 Standard Deviations (SD) is added to create randomness in the dataset. 50% missing values are generated randomly in each dataset prior using it for imputation using the seven DL models. For the three ML ensemble models, an additional 10% values are masked to train the ML models. Overall, the entire methodology is encapsulated in the block diagram in Fig. 1.

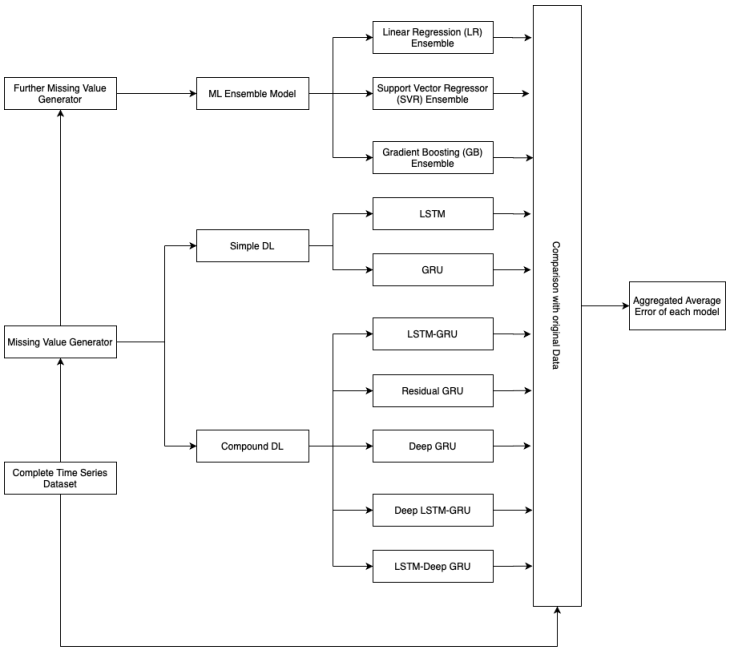

Fig. 1. Overall roadmap of the methodology. The complete dataset is passed through a missing value generator, which generates 50% missing values randomly in the dataset. Once the missing values are generated, the new dataset is fed into different DL models for imputation. For the ensemble model 10% further missing values are generated. The imputed results are compared with the original dataset to find the best model for that dataset.

A. Datasets

Each time series exhibits different characteristics, hence it is difficult to generalize one imputation method to fit all types of time series. Therefore, the four commonly used time series: trend, seasonal, combined and random time series are examined to find the best imputation method for each.

1) Trend Time Series (T): A trend time series moves in a simple linear fashion. The trend shows the general tendency of the data to increase or decrease during a long period of time. The increase or decrease does not need to be in the same direction throughout the given period of time [35]. Left side of Fig. 2 shows the created trend time series: without noise, with noise and with 50% missing values as mentioned in Section II.

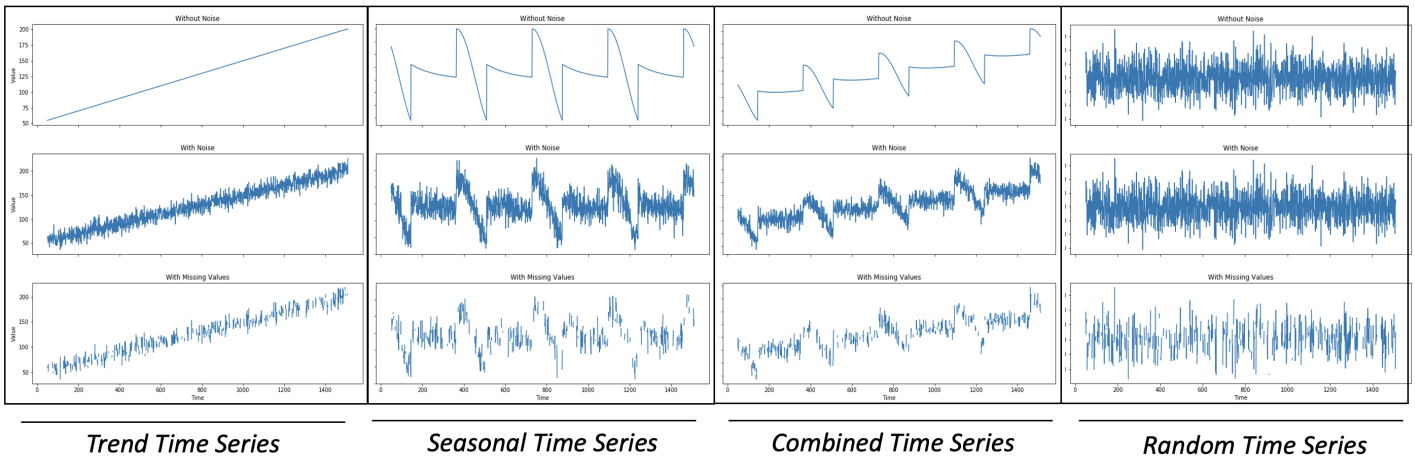

Fig. 2. Time series and its variation when noise is added and when missing values are generated.

2) Seasonal Time Series (S): A seasonal pattern exists when a series is influenced by seasonal factors (e.g., the quarter of the year, the month, or day of the week). Seasonality is always of a fixed and known period. Hence, seasonal time series are sometimes called periodic time series [36]. Middle portion of Fig. 2 shows the seasonal time series with and without the noise along with the missing values.

3) Combined Time Series (T&S): Many time series data contain the trend and seasonal patterns as its basic components. Trend and seasonality are very commonly encountered in economic and business time series [37]. Middle portion of Fig. 2 shows the combined time series utilized in this work.

4) Random Time Series: A random time series is a time series that does not fall in any of the aforementioned categories; as in right side of Fig. 2. Since the time series is random, two of its graphs, the with and without noise ones, look identical.

B. Imputation Framework

Our model is inspired by the work of Ma et al. [6]. The model consist of any of the module of RNN such as LSTM, Residual GRU and LSTM-Deep GRU etc, which has a learned linear combination of the previous states. The neural network is trained on the data and a vector, called Residual Sum Vector (RSV), is added to it. It is the weighted sum of the model's previous hidden states and the learned memory weights are then used to impute the missing values in the dataset.

The network operates in such a way that it learns and predicts simultaneously. Whenever the network finds the input variable present, it learns to regress it, and when the values are missing, they are filled based on this regression. This framework imputes randomly missing values in the dataset.

The considered time series are of T length as $X = \{x_1, x_2, ...x_T\}$ where $x_i \in \mathrm{R}^n$. An indicator vector of missing values is introduced, since the considered time series are incomplete, $M = \{m_1, m_2, ...m_T\}$ where $m_t \in \{0,1\}^n$. The m_{it} indicates whether the value is present at the ith position or not. $m_{it} = 0$ indicates that the value is present, while $m_{it} = 1$ portrays otherwise.

The RSV is deployed in the neural network in order to incorporate the past information from the hidden states. The output of RSV at time t is denoted by $r_t \in \mathrm{R}^m$ and its dimension is same as h_t. r_t is defined as:

$$r_t = \begin{cases} f(h_t), & \text{if } t = 1. \\ f(h_t + g(W_r, r_{t-1})), & \text{if } t = 2, 3, ..., T \end{cases} \quad (1)$$

here, g and f are the vector valued functions, $W_r \in \mathrm{R}^{m \times m}$, and $h_t \in \mathrm{R}^m$ represents the output of the network hidden layer at time t. The next input is approximated using:

$$z_{t+1} = W_{imp} r_t \quad (2)$$

In case x_{t+1} is present, z_{t+1} is trained to approximate it whereas if it is missing z_{t+1} is used to impute x_{t+1}.

The training of the model is done using back propagation of error. The process runs in such a way that if the next input x_{t+1} is present in the dataset then the output z_{t+1} is trained to approximate the input x_{t+1} and in the case where x_{t+1} is a missing value then z_{t+1} is directly copied to x_{t+1}. The approximating loss, L_{t_approx}, at each time step t, is the squared error loss between the original and approximated value, according to the existence of input x_t.

$$L_{t\_approx}(z_t, x_t) = \| \ (z_t - x_t \circ \neg m_t \ \|_2^2 \quad (3)$$

here, the $\neg m_t$ is masking off the missing data from the approximation loss.

The overall training loss has two components, the total approximating loss term (4) and the task related loss term (5), which are as follows:

$$L_{total\_approx} = \sum_{k=1}^{N} \sum_{t=1}^{T-1} L_{t\_approx}(z_{t+1}^{(k)}, x_{t+1}^{(k)}) \quad (4)$$

$$L_{total\_target} = \sum_{k=1}^{N} L_{target}(d^{(k)}, y_t^{(k)}) \quad (5)$$

The superscript k here denotes the kth sample of the time series. The $d^{(k)}$ and $y_t^{(k)}$ denote the task related target and

the output of the kth value. The total training loss, L_{total}, is obtained by combining (4) and (5).

$$L_{total} = L_{total\_approx} + \lambda_{target} L_{total\_target} \tag{6}$$

where λ_{target} is a coefficient weighing the importance of the task loss. This loss function is optimized by the Back Propagation Through Time algorithm. The overall working of the framework is illustrated in Fig. 3.

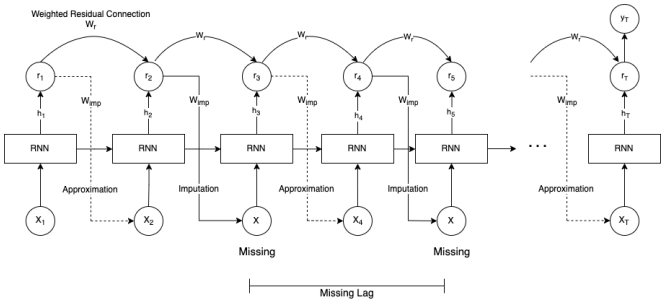

Fig. 3. Illustration of the imputation framework. The framework approximates and imputes at the same time. The "X" represents the missing value. The framework employs prediction by using neural networks to impute missing values.

IV. Utilized Imputation Models

For all the deployed DL models in this work a total of 100 nodes are added in the hidden layers except for deep GRU and deep LSTM-GRU models where 50 nodes are utilized.

Simple DL models have a single layer while compound DL models have multiple layers. The epochs are set to 150 and a batch size of 23 is used. The learning rate is annealed during training. Initial weights are uniformly distributed in the range [−0.1, 0.1]. For training, the Adam optimizer is used [38]. The experiments are conducted on the TensorFlow platform.

A. Simple DL Models

1) LSTM: One of the drawbacks of RNN is that it faces the vanishing gradient problem. Furthermore, whenever a long sequence of data is fed to RNNs, they have a problem of carrying the information from the earlier time steps to a later one causing them to miss important information. In order to mitigate these issues, LSTMs [39] are developed. They are called LSTMs as they have the ability to keep the short term memory for a longer period of time. The LSTMs are similar to the RNNs except that they have memory blocks instead of the neurons. The memory blocks consist of gates through which the information is propagated. The gates have a sigmoid neural network layer and point-wise multiplication operation. The sigmoid function assigns a value between zero and one which determines how much information should be allowed to pass through the gates.

LSTMs comprise three gates: forget gate, input gate and output gate. The forget gate, which is a sigmoid layer, decides what information needs to be discarded. It looks at the previous hidden state h_{t-1} and the current input x_t and outputs a number between 0 and 1 for the cell state c_{t-1} which decides how much information needs to be forgotten. It is denoted as f_t and is given as:

$$f_t = \sigma(W_f.[h_{t-1}, x_t] + b_f) \tag{7}$$

The input gate, i_t conditionally decides to update the memory state on the basis of input. It is composed of sigmoids which help decide what information has to be updated. The next tanh layer in the input gate creates a vector, $\vec{C_t}$, of the new candidate values which can be added to the state.

$$i_t = \sigma(W_i.[h_{t-1}, x_t] + b_i) \tag{8}$$

$$\vec{C_t} = \tanh(W_c.[h_{t-1}, x_t] + b_c) \tag{9}$$

In the next update layer the new vector $\vec{C_t}$ is combined with the previous cell state C_{t-1}. The old state is multiplied by f_t to forget the values we decided to forget initially.

$$C_t = f_t * C_{t-1} + i_t * \vec{C_t} \tag{10}$$

Then comes the output gate which decides what to output and has weights which are learned during the training.

$$\sigma_t = \sigma(W_o[h_{t-1}, x] + b_o) \tag{11}$$

$$h_t = \sigma_t * \tanh(C_t) \tag{12}$$

The sigmoid layer decides which part of the cell state is generated. Then, the cell state C_t is passed through tanh function and multiplied by the output of sigmoid layer, σ_t, in order to get the output. LSTMs are described in Fig. 4.

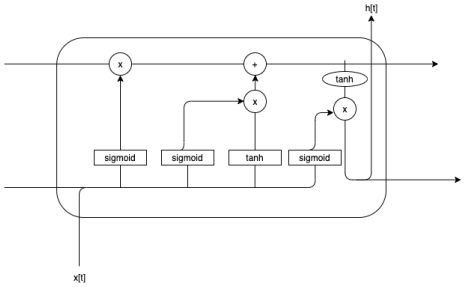

Fig. 4. Illustration of the LSTM structure. Since RNNs face vanishing or exploding gradient issues, LSTMs are introduced to mitigate those issues. The structure consists of gates that ensure that the issue of gradient is not encountered.

2) GRU: GRU is a modified version of LSTMs [40]. GRUs use hidden states instead of the cell state or memory to transfer the information. They have two gates: a reset gate and an update gate. The update gate acts similar to the forget and input gates in LSTMs. The reset gate, on the contrary, is used to decide how much past information has to be forgotten. GRUs are comparatively faster than the LSTMs since they have fewer tensor operations [34]. Both, LSTMs and GRUs, are designed to overcome the short term memory issues faced by the RNNs. Fig. 5 represents the GRU structure.

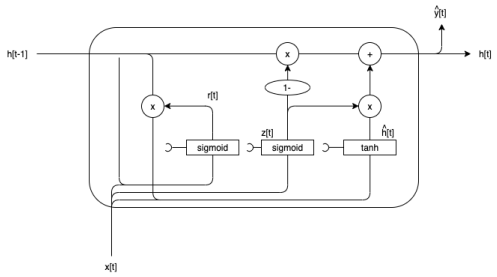

Fig. 5. Illustration of the GRU structure. Since RNNs face vanishing or exploding gradient issues, GRUs are introduced to mitigate those issues. The structure is similar to those of LSTMs and consist of gates that ensure that the issue of gradient is not encountered.

B. Compound DL Models

1) Residual GRU: Single layer of GRU with residual unit attached to it is used in this model. Residual network connects input in front of the layer directly to the output layer as illustrated Fig. 6.

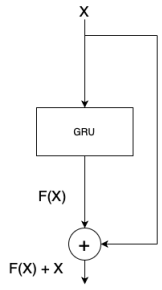

Fig. 6. Residual Network in which the input is directly connected to the output layer.

2) LSTM-GRU: This is a special type of neural network. It consists of two hidden layers where one layer is LSTM followed by GRU. This type of model produces promising results in stock price prediction as demonstrated in [41].

3) Deep GRU: The model is a deeper version of simple GRU. It consists of 2 layers of GRU.

4) LSTM-Deep GRU: The model is composed of an additional GRU layer with the LSTM-GRU configuration.

5) Deep LSTM-GRU: The model is a deeper version of LSTM-GRU model. It consists of 4 layers under the configuration LSTM - GRU - LSTM - GRU.

C. Ensemble Models

Ensemble pertains to combining results of two or more models and using them to perform the original task. This eliminates a lot of the model tuning that would otherwise be required to achieve good results.

Although neural networks offer flexibility and scalability to the amount of training data due to their non-linear behavior, a downside of this is that since they learn via stochastic training algorithms they are sensitive to the specifics of training data. This implies that a different set of weights can be found each time the model is trained, which in turn produces different predictions. Generally, these neural networks are referred as networks having high variance which makes it difficult to develop a final model to make predictions. An approach for reducing the variance in the network is to train multiple models instead of a single one and then combine predictions from those models [42]. This is called ensemble learning and it doesn't only reduce the variance of predictions but also results in better predictions.

There are multiple types of ensemble learning, the Stacked ML ensemble is examined in this work. For this ensemble model, the imputation results of the top 2 imputation models of each time series type are ensembled; the top 2 models are the ones with the least AGE values.

1) LR Ensemble: To apply the LR ensemble, an additional 10% of the values from the original dataset, which had 50% missing values, have to be masked. The top 2 DL imputation methods are applied on the resulting dataset. From imputation results of these DL models, the 10% masked values which were imputed are stacked along with their original values. The imputed values are considered as the input, while the actual values are considered as output for training the LR ensemble model. Masking of the values is done because ML models are supervised models that require both output along with its corresponding input to train itself. After the model is trained, it is used to impute the 50% missing values from the original dataset.

2) SVR Ensemble: The methodology is similar to the LR ensemble except that instead of LR we use SVR. Linear kernel is used in the model.

3) GB Ensemble: In GB ensemble the same methodology is used except that instead of LR the GB is used. The huber function is used for calculating the loss [43].

V. Experiments and Result Analysis

A. Simple vs. Compound DL Models

The considered DL imputation models are:

1) Simple DL Models
   a) Long Short Term Memory (LSTM)
   b) Gated Recurrent Unit (GRU)
2) Compound DL Models
   a) Residual GRU
   b) LSTM-GRU
   c) Deep LSTM-GRU
   d) LSTM-Deep GRU
   e) Deep GRU

These DL models are also compared against a non-DL model that uses linear function for interpolation. In order to gauge the effectiveness of the model with one value, the average error measure (AGE) is used to determine the top two imputation models for each tested time series type; AGE is calculated by comparing imputed values with actual ones. The results are summarized in Table I and are depicted in Fig. 7.

It can be seen that in both Trend and Seasonal time series the Residual GRU Network provides the best imputation results. This is because the network introduces the non-linearity in the system which enables it to predict and impute values close to the actual ones.

TABLE I
AGE OF EACH IMPUTATION METHOD AGAINST EACH TIME SERIES. THE BOLD FIGURES ILLUSTRATE THE TOP 2 PERFORMING MODELS FOR EACH TYPE.

Time Series Type	Measure	Imputation Model							
		Function	*Simple DL*		*Compund DL*				
		Linear	LSTM	GRU	Deep GRU	LSTM-GRU	Deep LSTM-GRU	Residual GRU	LSTM - Deep GRU
Trend	AGE (RMSE + MAE)/2	*11.71*	*10.60*	*9.92*	*9.94*	***9.89***	*10.05*	***9.86***	*10.44*
Seasonal		*11.55*	*10.60*	*10.82*	*11.20*	*10.75*	*11.45*	***10.17***	***10.54***
Combined		*11.50*	*10.94*	***10.73***	*11.64*	*10.83*	*12.45*	***10.75***	*10.98*
Random		*11.15*	*8.91*	*8.91*	*8.91*	***8.90***	*8.91*	*8.91*	***8.91***

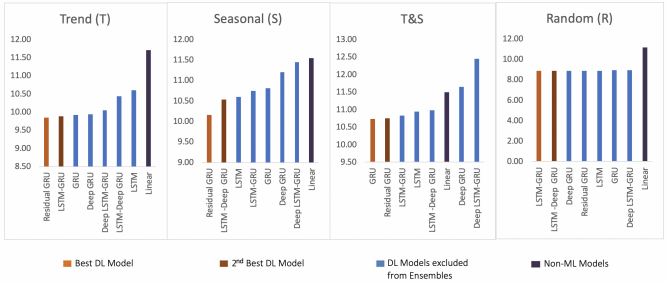

Fig. 7. Performance comparison of the utilized imputation methods highlighting the top 2 models for each time series type.

TABLE II
ML STACKING ENSEMBLE MODELS AGE FOR EACH TIME SERIES TYPE.

Time Series Type	Measure	*Stacking ML Ensemble*		
		LR Ensemble	SVR Ensemble	GB Ensemble
Trend	AGE (RMSE+MAE)/2	*9.95*	*10.08*	*11.27*
Seasonal		*10.56*	*10.53*	*11.71*
Combined		*10.86*	*10.80*	*14.64*
Random		*8.96*	*8.94*	*10.48*

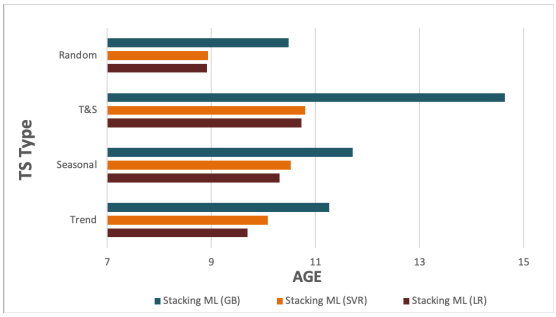

Fig. 8. ML ensembles performance comparison with each of the four time series types based on the AGE metric.

In the combined (T&S) time series, it is evident that GRU based neural networks are in general the most suitable choice for imputation. Since the combined time series incorporate the characteristics of both Trend and Seasonal time series, GRU based neural networks are found to be optimal in both.

It is also observed that in this case the simple GRU network is the best performing model compared to the other models. The optimal results obtained by the GRU are due to the simpler structure that enables it to take into account the non-linearity. Furthermore, the GRU performs better than LSTM because GRU allows the errors to back-propagate without vanishing too quickly as it bypasses multiple temporal steps [34].

In the Random time series, almost all the models are found to be suitable for imputation. Each of the models has almost the same AGE value and thus, any of them can be used. Such results are found due to the randomness in data.

B. Ensemble Models

Three ensemble models are compared in this work. The ensemble is implemented for the top two models that have the least two AGE values. Once the best performing models for each time series type are decided as in the previous experiment, the ensemble of the top two models is built. The LR, SVR and GB ML Stacking ensembles are used. The imputed values of the 10% masked values from the two models are stacked as inputs and the output is the actual values which were masked. These values are used to train the model and then the originally missing values present in the dataset are predicted using this trained model. The results based on the AGE metric are illustrated in Fig. 8 and Table II for all four tested types of time series. As evident from Table II and Fig. 8, the ML ensemble did not provide satisfactory results. This is due to the masking of additional 10% of the input data to the ML ensemble which is needed to train the ML model. This in turn means that the DL models have a greater portion of missing values to impute, which results in higher AGE.

VI. CONCLUSION

Missing values are commonly observed in time series datasets, hence data imputation is imperative in the data preprocessing stage prior to using the dataset for prediction or forecasting. A recommendation framework is provided in this work for selecting the most optimal imputation strategy for four types of time series. Since each time series exhibit different characteristics, it is difficult to generalize one method to all types. From experiments results, it is observed that the GRU based neural networks perform better in general, however, the complexity of the model varies depending on the type of time series. The residual GRU is recommended for the trend and seasonal time series, the GRU is recommended for the combined (T&S) type while for the random time series any of the tested DL models can be used for imputation. In addition, the Stacking ML ensembles fail in further enhancing the performance of the already top two performing DL imputation models for all tested time series types.

For future work, more types of time series should be considered. In addition, the consecutive missing values issue should be tackled as the results in that case might differ. This is mainly because each time series carries information within itself and the absence of a chunk of values might deteriorate the performance of the neural network. Moreover, improved recurrent neural network architectures such as dual attention [44], [45] or Convolutional Neural Networks- ConvLSTM [46], [47] in addition to other types of ensembles can be tried to determine their effectiveness in the imputation of missing values.

ACKNOWLEDGMENT

We acknowledge the generous contribution of Loblaws which funded the Research Chair in AI at the University of Waterloo. This work is also partially supported by NSERC CRD grant.

REFERENCES

[1] R. Billinton, H. Chen, and R. Ghajar, "Time-series models for reliability evaluation of power systems including wind energy," *Microelectron. Rel.*, vol. 36, no. 9, pp. 1253 – 1261, 1996.

[2] Z. Bar-Joseph, G. K. Gerber, D. K. Gifford, T. S. Jaakkola, and I. Simon, "Continuous representations of time-series gene expression data," *J. Comput. Biol.*, vol. 10, no. 3-4, pp. 341–356, 2003.

[3] W. D. Ray, "Time series analysis: A comprehensive introduction for social scientists," *J. Royal Stat. Soc.*, vol. 145, no. 4, pp. 511–511, 1982.

[4] C. Yozgatligil, S. Aslan, C. Iyigun, and I. Batmaz, "Comparison of missing value imputation methods in time series: the case of turkish meteorological data," *Theor. Appl. Climatol.*, vol. 112, no. 1-2, pp. 143–167, 2013.

[5] S. Moritz, A. Sardá, T. Bartz-Beielstein, M. Zaefferer, and J. Stork, "Comparison of different methods for univariate time series imputation in R," 2015.

[6] Q. Ma, S. Li, L. Shen, J. Wang, J. Wei, Z. Yu, and G. W. Cottrell, "End-to-end incomplete time-series modeling from linear memory of latent variables," *IEEE Trans Cybern*, pp. 1–13, 2019.

[7] A. R. T. Donders, G. J. [van der Heijden], T. Stijnen, and K. G. Moons, "Review: A gentle introduction to imputation of missing values," *J. Clin. Epidemiol.*, vol. 59, no. 10, pp. 1087 – 1091, 2006.

[8] P. Royston, "Multiple imputation of missing values," *Stata J*, vol. 4, no. 3, pp. 227–241, 2004.

[9] T. A. Moahmed, N. El Gayar, and A. F. Atiya, "Forward and backward forecasting ensembles for the estimation of time series missing data," in *Artificial Neural Networks in Pattern Recognition*, N. El Gayar, F. Schwenker, and C. Suen, Eds. Cham: Springer International Publishing, 2014, pp. 93–104.

[10] W. F. Velicer and S. M. Colby, "A comparison of missing-data procedures for arima time-series analysis," *Educ Psychol Meas*, vol. 65, no. 4, pp. 596–615, 2005.

[11] F. V. Nelwamondo, S. Mohamed, and T. Marwala, "Missing data: A comparison of neural network and expectation maximization techniques," *Curr. Sci.*, vol. 93, no. 11, pp. 1514–1521, 2007. [Online]. Available: http://www.jstor.org/stable/24099079

[12] E.-L. Silva-Ramírez, R. Pino-Mejías, M. López-Coello, and M.-D. C. de-la Vega, "Missing value imputation on missing completely at random data using multilayer perceptrons," *Neural Netw*, vol. 24, no. 1, pp. 121 – 129, 2011.

[13] Z. Che, S. Purushotham, K. Cho, D. Sontag, and Y. Liu, "Recurrent neural networks for multivariate time series with missing values," *Sci. Rep.*, vol. 8, no. 1, p. 6085, 2018.

[14] M. Hua and J. Pei, "Cleaning disguised missing data: A heuristic approach," in *KDD*, ser. KDD '07, New York, NY, USA, 2007, p. 950–958.

[15] X.-H. Zhou, G. J. Eckert, and W. M. Tierney, "Multiple imputation in public health research," *Stat Med*, vol. 20, no. 9-10, pp. 1541–1549, 2001.

[16] S. Nakagawa and R. P. Freckleton, "Missing inaction: the dangers of ignoring missing data," *Trends Ecol. Evol.*, vol. 23, no. 11, pp. 592 – 596, 2008.

[17] G. L. Schlomer, S. Bauman, and N. A. Card, "Best practices for missing data management in counseling psychology." *J. Couns. Psychol.*, vol. 57, no. 1, p. 1, 2010.

[18] P. D. Allison, *Missing data*. Sage Publications, 2001, vol. 136.

[19] D. S. Fung, "Methods for the estimation of missing values in time series," Master's thesis, Edith Cowan University Perth, 2006.

[20] K. Sutiene, G. Vilutis, and D. Sandonavicius, "Forecasting of GRID job waiting time from imputed time series," *Elektron Elektrotech*, vol. 114, no. 8, pp. 101–106, Oct. 2011.

[21] W. Junger and A. P. de Leon], "Imputation of missing data in time series for air pollutants," *Atmos. Environ.*, vol. 102, pp. 96 – 104, 2015.

[22] A. C. Harvey, *Forecasting, structural time series models and the Kalman filter*. Cambridge University Press, 1990.

[23] O. Troyanskaya, M. Cantor, G. Sherlock, P. Brown, T. Hastie, R. Tibshirani, D. Botstein, and R. B. Altman, "Missing value estimation methods for DNA microarrays ," *Bioinformatics*, vol. 17, no. 6, pp. 520–525, 06 2001.

[24] T. Yu, H. Peng, and W. Sun, "Incorporating nonlinear relationships in microarray missing value imputation," *EEE/ACM Trans. Comput. Biol. Bioinform.*, vol. 8, no. 3, pp. 723–731, 2011.

[25] S. Chiewchanwattana, C. Lursinsap, and C.-H. H. Chu], "Imputing incomplete time-series data based on varied-window similarity measure of data sequences," *Pattern Recognit Lett*, vol. 28, no. 9, pp. 1091 – 1103, 2007.

[26] D. Wu, X. Luo, M. Shang, Y. He, G. Wang, and M. Zhou, "A deep latent factor model for high-dimensional and sparse matrices in recommender systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, pp. 1–12, 2019.

[27] J. Chen and X. Luo, "Randomized latent factor model for high-dimensional and sparse matrices from industrial applications," in *2018 IEEE 15th International Conference on Networking, Sensing and Control (ICNSC)*, 2018, pp. 1–7.

[28] H. Wu, X. Luo, and M. Zhou, "Advancing non-negative latent factorization of tensors with diversified regularizations," *IEEE Transactions on Services Computing*, pp. 1–1, 2020.

[29] X. Luo, H. Wu, H. Yuan, and M. Zhou, "Temporal pattern-aware qos prediction via biased non-negative latent factorization of tensors," *IEEE Transactions on Cybernetics*, vol. 50, no. 5, pp. 1798–1809, 2020.

[30] E. Choi, M. T. Bahadori, A. Schuetz, W. F. Stewart, and J. Sun, "Doctor ai: Predicting clinical events via recurrent neural networks," in *Proc. Mach. Learn. Healthcare Conf.*, 2016, pp. 301–318.

[31] Z. C. Lipton, D. Kale, and R. Wetzel, "Directly modeling missing data in sequences with rnns: Improved classification of clinical time series," in *Proc. Mach. Learn. Healthcare Conf.*, 2016, pp. 253–270.

[32] J. Yoon, W. R. Zame, and M. van der Schaar, "Multi-directional recurrent neural networks : A novel method for estimating missing data," 2017.

[33] K. Cho, B. van Merrienboer, Ç. Gülçehre, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder-decoder for statistical machine translation," *Proc. Conf. Empirical Methods Natural Lang. Process. (EMNLP)*, vol. abs/1406.1078, 2014. [Online]. Available: http://arxiv.org/abs/1406.1078

[34] J. Chung, Ç. Gülçehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *CoRR*, vol. abs/1412.3555, 2014. [Online]. Available: http://arxiv.org/abs/1412.3555

[35] G. Zhang and M. Qi, "Neural network forecasting for seasonal and trend time series," *Eur. J. Oper. Res.*, vol. 160, no. 2, pp. 501 – 514, 2005, decision Support Systems in the Internet Age.

[36] E. Ghysels and D. R. Osborn, *The econometric analysis of seasonal time series*. Cambridge University Press, 2001.

[37] M. Gan, Y. Cheng, K. Liu, and G. lin Zhang, "Seasonal and trend time series forecasting based on a quasi-linear autoregressive model," *Appl. Soft Comput.*, vol. 24, pp. 13 – 18, 2014. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1568494614003214

[38] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[39] S. Hochreiter, "The vanishing gradient problem during learning recurrent neural nets and problem solutions," *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, vol. 06, no. 02, pp. 107–116, 1998.

[40] R. Fu, Z. Zhang, and L. Li, "Using LSTM and GRU neural network methods for traffic flow prediction," in *Proc. 31st Youth Academic Annu. Conf. Chin. Assoc. Automat. (YAC)*, 2016, pp. 324–328.

[41] A. M. Rather, A. Agarwal, and V. Sastry, "Recurrent neural network and a hybrid model for prediction of stock returns," *Expert Syst. Appl.*, vol. 42, no. 6, pp. 3234 – 3241, 2015.

[42] Z.-H. Zhou, J. Wu, and W. Tang, "Ensembling neural networks: Many could be better than all," *Artif. Intell.*, vol. 137, no. 1, pp. 239 – 263, 2002.

[43] A. Natekin and A. Knoll, "Gradient boosting machines, a tutorial," *Front. Neurorobot.*, vol. 7, p. 21, 2013.

[44] Y. Qin, D. Song, H. Chen, W. Cheng, G. Jiang, and G. W. Cottrell, "A dual-stage attention-based recurrent neural network for time series prediction," *arXiv preprint arXiv:1704.02971*, vol. abs/1704.02971, 2017.

[45] T. Chen, H. Yin, H. Chen, L. Wu, H. Wang, X. Zhou, and X. Li, "Tada: Trend alignment with dual-attention multi-task recurrent neural networks for sales prediction," in *ICDM*, 2018, pp. 49–58.

[46] T.-Y. Kim and S.-B. Cho, "Predicting residential energy consumption using cnn-lstm neural networks," *Energy*, vol. 182, pp. 72 – 81, 2019.

[47] Z. Tan and P. Pan, "Network fault prediction based on cnn-lstm hybrid neural network," in *CISCE*, 2019, pp. 486–490.