Alignment-driven neural network variant for irregular multivariate time series

Anonymous Author(s)

ABSTRACT

Time series analysis is an increasingly important topic in data science. This is a consequence of the massive adoption, in many fields such as medicine and engineering, of electronic sensors dedicated to the continuous collection of data from heterogeneous sources. However, due to the heterogeneous configuration of these sensors, the processing of time series requires considering sparsity, irregularity and missing values issues. Added to this is the problem of alignment when several irregular univariate time series, also known as irregular multivariate time series (IMTS), are involved in the processing. Various methods have been designed to fully or partially solve the above-mentioned issues. Although some of them have obtained functional results, they still have to align the values manually, which can increase the rate of missing values and therefore lead to a drop in model performance. To address this issue, we propose a variant of the Alignment-driven Neural Network model that incorporates a data-driven imputation method and a novel loss function dedicated to explicitly preserving univariate time series similarity scores in latent space. We have conducted extensive empirical experiments on two real-world datasets, MIMIC-3 and Physionet, to show that our model improves the accuracy of multivariate time series classification tasks.

KEYWORDS

Alignment, Data-driven imputation, Deep learning, Irregular Multivariate time series.

ACM Reference Format:

Anonymous Author(s). 2018. Alignment-driven neural network variant for irregular multivariate time series. In . ACM, New York, NY, USA, 9 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

The adoption of electronic sensors designed to collect heterogeneous data from heterogeneous sources is undoubtedly the factor that has contributed the most to the advancement of research in the field of time series analysis. Indeed, through these, a massive amount of data can be collected continuously over time and used for various tasks such as climate forecast [3], person activity recognition [13], traffic congestion [9] to name a few. While quantity can be guaranteed, quality is often lacking.

In many real-world applications, it is common to find sensors that vary in their data collection frequency. This variation leads to temporal irregularity and the sparsity of collected time series data. Additionally, we may also face the problem of missing values due to sensor failure and corrupted data. All these aforementioned issues lead to irregular univariate time series data when a single sensor is used for time series analysis and irregular multivariate time series (IMTS) when there are multiple sensors.

Due to the deep learning architecture's ability to learn from hidden patterns and their superior performance compared to classical machine learning or statistical models on various tasks, several based-deep learning models such as [4, 19, 21, 24, 30, 31] have been implemented to partially or fully solve problems related to irregular time series. Although these models have shown functional results, they still perform manual alignment during the pre-processing phase. Alignment refers to the representation of irregular multivariate time series data in matrix form, where each row represents the value of each sensor at a specific timestamp. The problem with this approach is that it can increase missing value rates. Figure 1 illustrates this. It is observed that when a manual alignment is performed with three different sensors (a, b and c) whose time trend is different, the rate of missing values increases. For instance, it went from 2 to 14.

Unlike most existing methods, we do not perform manual alignment, which could increase the rate of missing values and affect model performance. Instead, we rely on the alignment-driven neural network (ALNN), as proposed in [1], to perform this alignment while overcoming one of its main limitations, namely the imputation of missing values during the pre-processing stage. We propose a data-driven method based on graph neural network [8, 10] and message-passing to fill in initial missing values, i.e. those that are not caused by a temporal irregularity but rather by other factors such as sensor failure. Subsequentially, we feed the imputed irregular multivariate time series into the ALNN to produce a pseudo-aligned latent multivariate time series (PLMTS). PLMTS is a matrix in which each row contains the latent values of each sensor at each user-defined reference time point (see box at the bottom right of Figure 1).

In addition to the data-driven method we used to get around the with the ALNN limitation, we changed its core and added a new loss function that helps it keep the underlying similarity between sensors existing in the observed space in the latent space. The changes we made are the following:

- As we perform feedforwardfill (or padding) to deal with time series of variable length, we introduce a padded mask that will help the model adapt its decision more on unpadded values than padded ones, which may be a noise factor.
- We removed the temporal variation matrix because we noted that the temporal variation was implicitly encoded

Conference'17, July 2017, Washington, DC, USA
© 2018 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-x-xxxx-xxxx-x/YY/MM
https://doi.org/XXXXXXX.XXXXXXX

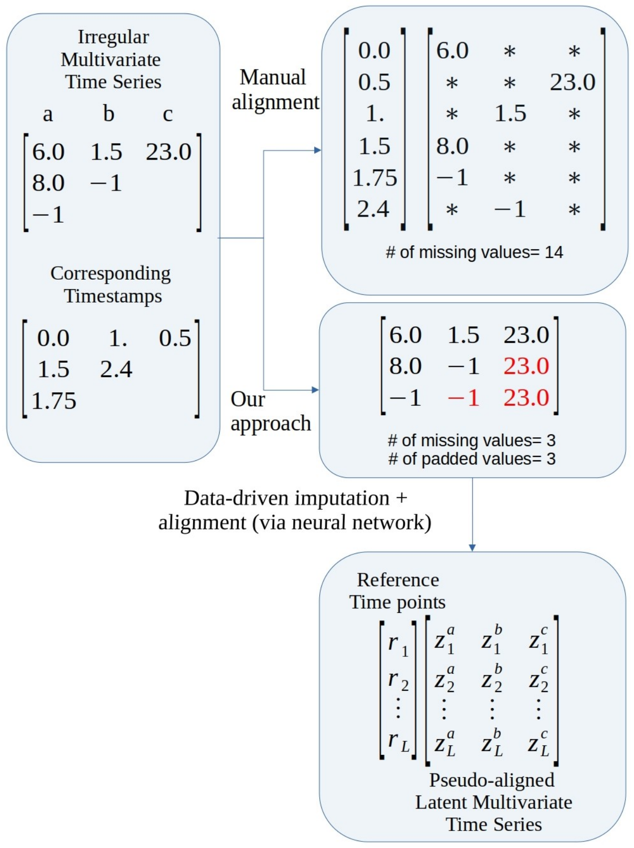

Figure 1: Illustration manual alignment versus our approach. a, b and c are three different sensors. -1s represent missing initial values (e.g., due to sensor failure). $*$ represents missing values caused by temporal irregularity. z_l^a represents the latent value of the sensor a at the reference time point r_l. Red values are padded values.

in the time lag intensity matrix, which is dedicated to calculating the amount of information that must be extracted from each value at each reference time.

In summary, the main contributions of this paper are as follows:

- Proposing a data-driven imputation method based on a graph neural network and message passing to impute missing values in IMTS;
- Introducing a padded mask to allow the model to drive its decision more on the unpadded values rather than on the padded ones;
- Feeding the imputed IMTS into a modified ALNN to obtain its corresponding pseudo-aligned latent multivariate time series (PLMTS);
- Introducing a new loss function that helps the model maintain the underlying similarity between sensors existing in the observed space in the latent space;
- Training the model in an end-to-end fashion with a GRU (which works better with regular and aligned data) that is responsible for learning the sequential pattern of the multivariate time series and a set of stacked feed-forward neural networks that act as a classifier.
- We use two real-world databases, MIMIC-3 and Physionet, to evaluate our model versus the state-of-the-art models designed for irregular multivariate time series. We also carried out various ablation studies. The results obtained showed that the different strategies we implemented improved the accuracy of downstream tasks.

2 RELATED WORK

Temporal irregularity and data sparsity are the main issues we may face when processing data collected from sensors whose data collection frequencies may vary over time. Additionally, when multiple asynchronous sensors are involved in our study, we will also face the alignment issue. In the case of data collected by a single sensor with the above characteristics, we speak of irregular univariate time series and, in the case of several sensors, irregular multivariate time series (IMTS). A common, simple, and acceptable approach to dealing with irregular time series is to discretize into equidistant time intervals the period over which the data is collected and to fill the empty time intervals with zeros, the empirical mean, or the empirical median [17, 29]. The disadvantage of this method is that the aggregation carried out when many values are observed over the same time bin may remove fine-grained information. Furthermore, as stated in [24], this discretization into equidistant time intervals of the data collection period removes the underlying information carried by the temporal irregularity. To mitigate the impact of imputed values, [15] introduces a binary mask modelling the missing information pattern during the learning phase of their RNN model. [11] proposed to calculate the level of uncertainty of imputed values and propagate it over time so that imputed values with a low level of uncertainty cannot introduce noise in the model calculations. Since imputation during pre-processing is often based on strong assumptions, some methods have adopted a data-driven approach that allows imputation to be driven by the objective function of the downstream task [5, 23]. [6] modelled with learnable GRU parameters, a medical assumption that drives the imputation process. In addition, they applied an exponential decay mechanism to the hidden states of the GRU to model temporal irregularity. With this trick, they did not need to discretize time. Other methods based on a semi-parametric interpolation, Spline and time gate control, have been implemented by [2, 16, 20] to directly process irregular multivariate time series without performing any pre-processing.

The similarity between all aforementioned models is that they are almost RNN variants models. Indeed, much of the research on irregular time series processing consists of adding components to the RNN or modifying its core so that it can better handle irregular time series. Inspired by [7], [18] and [13] proposed to combine RNNs with ordinary differential equations (ODE) so that irregular time series can be processed in a continuous space and thus overcome the problem of temporal irregularity. Similar work using ODE to overcome the irregular time series issues has also been carried out by [12]. More recently, due to the effectiveness of Transformers in natural language processing, several models based on transformer architecture have been implemented to handle irregular time series.

[19, 21, 22, 24, 25, 28]. In these models, irregular timestamps are often mapped into embedded vectors with periodic functions and combined with the embedded representation of their respective values.

Although the above models have achieved remarkable performance, most of them, still require manual alignment during the pre-processing phase, which increases the rate of missing values and makes them less accurate than they likely would have been with fewer missing values. To overcome this limitation, the authors in [1] , proposed a data-driven alignment model called ALNN. However, the imputation they perform to deal with missing values (not caused by irregular timestamps) relies on strong assumptions and undoubtedly introduces noise into the model calculation. So we leverage this ALNN to provide a more efficient way to align data. We introduce a data-driven method that relies on a graph neural network and message passing to impute missing values. Indeed, graph neural networks have proved to be an effective model for filling in missing values in multivariate time series on the basis of surrounding values, also known as 'neighbours' [27, 32]. As our method requires padding to handle univariate time series of variable length, we introduce a padding mask to mitigate their negative impact. In addition to this trick, we implement a new loss function responsible for preserving the underlying similarity score between univariate time series existing in the observed space in the latent space.

3 PROPOSED MODEL

In this section, we formally present our model, which is an ALNN variant. Indeed, ALNN was designed to overcome the limitations of RNN when dealing with irregular multivariate time series. Although it showed functional results, the different techniques used to fill in missing values rely on strong assumptions and undoubtedly introduce noise during the calculations. To overcome this limitation, we use a graph neural network structure and messaging to fill in missing values. The result of this imputation, which is the imputed irregular multivariate time series (IMTS), is fed into a modified ALNN responsible for transforming the IMTS into pseudo- aligned latent multivariate time series (PLMTS). Therefore, the GRU that performs better with regular time series can be further used for sequence modelling of the PLMTS. Finally, we pass the GRU's output to the classifier or regressor (depending on the downstream task). Furthermore, to preserve the similarity score of univariate time series (or sensors) that exist in IMTS in PLMTS, we introduce a new loss function during the training phase. **The model architecture is shown in Figure 2. Each main component, including data-driven imputation, alignment, and classifier, is detailed and supported with illustrations in the following subsections.**

In the next subsection, we first provide the formal description of the data.

3.1 Data notation

We let $\mathcal{D} = \{X_n, S_n, M_n, P_n, y_n\}_{n=1}^{N}$ represent the dataset, where N is the number of samples. $X_n = [\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_T]_n^{\intercal} \in \mathbb{R}^{T\times K}$ is the *n-th* multivariate time series with T observations. $\mathbf{x}_t = [x_t^1, x_t^2, \cdots, x_t^K]$ is the *t-th* observation of K features. T represents the number of observations per univariate time series. As univariate

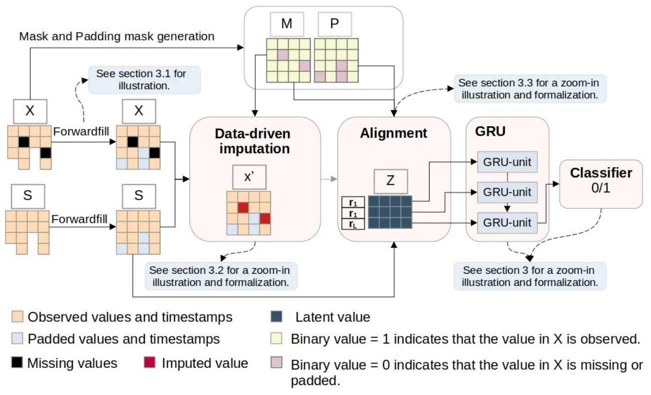
Mask and Padding mask generation
M
P
See section 3.1 for illustration.
See section 3.3 for a zoom-in illustration and formalization.
X
Forwardfill
X
Data-driven imputation
x'
Alignment
Z
GRU
GRU-unit
GRU-unit
GRU-unit
Classifier
0/1
S
Forwardfill
S
r₁
r₁
r_L
See section 3.2 for a zoom-in illustration and formalization.
See section 3 for a zoom-in illustration and formalization.
Observed values and timestamps
Padded values and timestamps
Missing values
Imputed value
Latent value
Binary value = 1 indicates that the value in X is observed.
Binary value = 0 indicates that the value in X is missing or padded.

Figure 2: Model architecture.

time series may have different numbers of observations, T was defined as follows: We set its value to that of the univariate time series with the maximum number of observations in the entire dataset. For the univariate that does not have T observations, we apply a forwardfill (see Figure 3). $S_n = [\mathbf{s}_1, \mathbf{s}_2, \cdots, \mathbf{s}_T]_n^{\intercal} \in \mathbb{R}^{T\times K}$, $P_n = [\mathbf{p}_1, \mathbf{p}_2, \cdots, \mathbf{p}_T]_n^{\intercal} \in \{0,1\}^{T\times K}$ and $M_n = [\mathbf{m}_1, \mathbf{m}_2, \cdots, \mathbf{m}_T]_n^{\intercal} \in \{0,1\}^{T\times K}$, are the timestamp matrix, the padding matrix and the mask matrix associated with the *n-th* multivariate time series, respectively. $p_t^k \in \mathbf{p}_t$ is a binary value that indicates whether x_t^k is a padded value or not. $p_t^k = 0$ if x_t^k is a padded value; 1 otherwise. $m_t^k \in \mathbf{m}_t$ is a binary value that indicates whether x_t^k is missing. $m_t^k = 1$ if x_t^k is observed; 0 otherwise. y_n is the target value of the *n-th* multivariate time series. Depending on the downstream task (classification or regression), y can be a categorical target or a real number. Figure 3 illustrates how samples are constructed.

In the following subsections, we omit the subscript n to simplify the formulas.

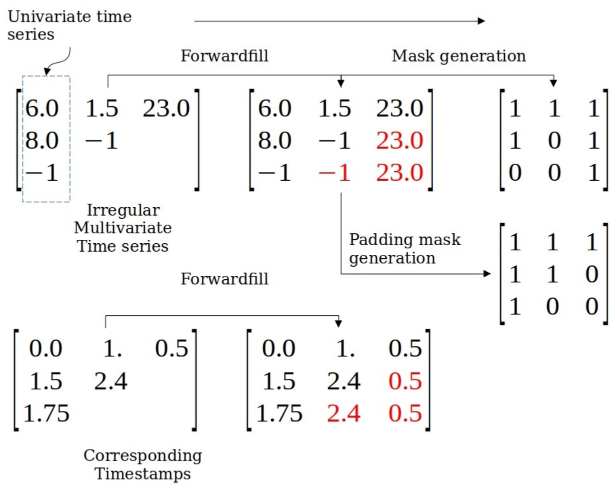

Figure 3: Illustration of sample construction. The −1s represent missing values. The red values are the padded values.

3.2 Data-driven imputation

To handle missing values in IMTS, we first represent them in the form of a structured null graph (a graph without edges). We then construct the oriented edges on the basis of the missing values. The principle of this construction is to impute a value by exploiting adjacent horizontal and vertical values. It is, in fact, a message-passing process. Missing values are imputed based on the surrounding observed values (also called *neighbours*). The imputation strategy is shown in Figure 4. We carry out two types of imputation: intra and extra.

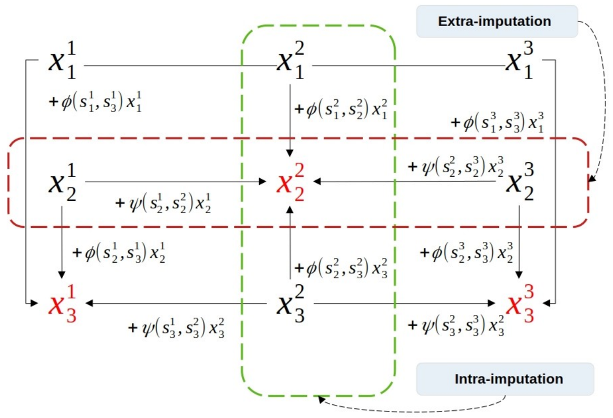
Extra-imputation
x_1^1
x_1^2
x_1^3
$+\phi(s_1^1,s_3^1)x_1^1$
$+\phi(s_1^2,s_2^2)x_1^2$
$+\phi(s_1^3,s_3^3)x_1^3$
x_2^1
x_2^2
x_2^3
$+\psi(s_2^2,s_2^3)x_2^3$
$+\psi(s_2^1,s_2^2)x_2^1$
$+\phi(s_2^1,s_3^1)x_2^1$
$+\phi(s_2^2,s_3^2)x_3^2$
$+\phi(s_2^3,s_3^3)x_2^3$
x_3^1
x_3^2
x_3^3
$+\psi(s_3^1,s_3^2)x_3^2$
$+\psi(s_3^2,s_3^3)x_3^2$
Intra-imputation

Figure 4: Imputation strategy. Red values are the missing values that we seek to fill in. $\phi(.)$ and $\psi(.)$ are the interpolation kernels. The values obtained with $\phi(s_t^k, s_{t'}^k)x_{t'}^k$ are summed up (hence the + symbol) to obtain the corresponding intra-imputed value of x_t^k. The values obtained with $\psi(s_t^k, s_t^{k'})x_t^{k'}$ are summed up to obtain the corresponding extra-imputed value of x_t^k.

Intra-imputation. That consists of imputing a missing value x_t^k based exclusively on the observed values of the univariate time series k to which it belongs. **The observed values used in intra-imputation are those that are vertically adjacent to the one we seek to impute. For example, in Figure 4, the observed values used to impute x_2^2 are x_2^1 and x_2^3.** The intra-imputed value $\hat{x}_t^k$ of x_t^k is obtained as follows:

$$\hat{x}_t^k = \sum_{t'=1}^{T} \phi(s_t^k, s_{t'}^k) x_{t'}^k \tag{1}$$

$$\phi(s_t^k, s_{t'}^k) = \frac{\hat{w}_{t'}^k m_{t'}^k exp(- \mid s_t^k - s_{t'}^k \mid)}{\sum_{l=1}^{T} (\hat{w}_l^k m_l^k exp(- \mid s_t^k - s_l^k \mid)) + \epsilon} \tag{2}$$

where $x_{t'}^k$ is a value of the univariate time series k and $m_{t'}^k$ and $s_{t'}^k$ are its corresponding mask value and timestamp. $\phi(s_t^k, s_{t'}^k)$ is the interpolation time kernel. Its numerator calculates the absolute temporal distance $\mid s_t^k - s_{t'}^k \mid$ between the value x_t^k that we wish to impute and its neighbour $x_{t'}^k$. This absolute distance is then: i) passed to an exponential decay function $exp(-(.))$ to keep the value between $[0, 1]$ (the intuition is to give more weight to data points close in time to the one we are seeking to impute); ii) then we perform a product between $exp(- \mid s_t^k - s_{t'}^k \mid)$ and $m_{t'}^k$ to take into account only the observed values $s_{t'}^k$; and iii) finally, we multiply $m_{t'}^k exp(- \mid s_t^k - s_{t'}^k \mid)$ by a weight $\hat{w}_{t'}^k$ which is a learning parameter associated with $x_{t'}^k$. The denominator is used for normalization. We add an epsilon value to prevent division by zero.

Extra-imputation. As the values of different sensors may be correlated, we impute the missing values in a univariate time series k by also considering the observed values of other univariate time series k'. It should be noted that only the values horizontally adjacent to the one we are seeking to impute are taken into account in the calculation. **For example, in Figure 4, the observed values used to impute x_2^2 are x_2^1 and x_2^3.** The extra-imputed value $\tilde{x}_t^k$ of x_t^k is obtained as follows:

$$\tilde{x}_t^k = \sum_{k'=1}^{K} \psi(s_t^k, s_t^{k'}) x_t^{k'} \tag{3}$$

$$\psi(s_t^k, s_t^{k'}) = \frac{m_t^{k'} \sigma(\tilde{w}_t^{k'} H_{k,k'} + \tilde{u}_t^{k'} e^{(-|s_t^k - s_t^{k'}|)})}{\sum_{l=1}^{K} m_t^l \{\sigma(\tilde{w}_t^l H_{k,l} + \tilde{u}_t^l e^{(-|s_t^k - s_t^l|)})\} + \epsilon} \tag{4}$$

$$H = \{\underbrace{softmax(\frac{Q^1 (K^1)^\top}{\sqrt{d}})}_{h^1} \cdots \| \cdots \underbrace{softmax(\frac{Q^j (K^j)^\top}{\sqrt{d}})}_{h^j}\} W_H \tag{5}$$

$$Q^j = (\mathbf{X} \odot \mathbf{M})^\top W_q^j;\ K^j = (\mathbf{X} \odot \mathbf{M})^\top W_k^j; \tag{6}$$

where $\odot$ is the Hadamard product. J is the number of self-attention [26] h. $H \in \mathbb{R}^{K \times K}$ is the mixture of similarity score matrices between univariate time series pairs. It is obtained by a concatenation of $J \times h$ self-attentions combined linearly by $W_H \in \mathbb{R}^{JK \times K}$ which is a learning parameter **(see Equation 5)**. Each coefficient of H that we represent by $H_{k,k'}$ is the similarity score between two univariate time series k and k'. $(. \| .)$ is the concatenation symbol and Q^j and K^j are the j-th query and key matrices obtained by linear transformation of the observed values $X \odot M$. d is the dimension of the query vector. σ is the relu function. $\psi(s_t^k, s_t^{k'})$ is the interpolation kernel, which is calculated on the one hand as a function of the temporal distance $\mid s_t^k - s_t^{k'} \mid$ of x_t^k and its adjacent horizontal neighbour $x_t^{k'}$ and, on the other hand, according to the similarity score $H_{k,k'}$ between the univariate time series k and k'. Its numerator is a non-linear combination of the similarity score $H_{k,k'}$ and the exponential decay time distance $e^{(-|s_t k - s_t k'|)}$ multiplied by a mask value $m_t^{k'}$. The denominator is used for normalization. $\tilde{w}_t^{k'}$ **and $\tilde{u}_t^{k'}$ are learnable parameters.** We add an epsilon value to prevent division by zero.

Instead of simply averaging the intra-imputed $\hat{x}_t^k$ value a and the extra-imputed value $\tilde{x}_t^k$ to obtain the final imputed value $\bar{x}_t^k$, we calculate the latter as follows:

$$\bar{x}_t^k = a_t^k \hat{x}_t^k + (1 - a_t^k) \tilde{x}_t^k;\ a_t^k \in [0, 1] \tag{7}$$

where a_t^k is an element of $A \in \mathbb{R}^{T\times K}$ which is the balancing matrix that we calculate from the matrix mask M as follows:

$$A = softmax(MW_A) \tag{8}$$

$W_A \in \mathbb{R}^{K\times K}$ is a learnable parameter. The intuition behind this approach is to tailor the coefficient a_t^k, which we call the balancer, according to the underlying missing pattern of the actual input rather than that of the entire dataset. The imputed IMTS $X'(x'^k_t)$ is then:

$$X' = M \odot X + (1 - M) \odot \bar{X} \tag{9}$$

where $\bar{X}(\bar{x}_t^k) \in \mathbb{R}^{T\times K}$ is the matrix of imputed values.

In the next subsection, we present how the imputed IMTS is processed through a modified ALNN to obtain a pseudo-aligned latent multivariate time series (PLMTS).

3.3 Alignment

The IMTS we have processed so far are not aligned and regular, meaning that the value of each sensor in each row may have been observed at different irregular timestamps. Since this misalignment can affect the accuracy of the downstream task, we use ALNN, a neural network designed to transform IMTS into PLMTS. **In ALNN, a set of regularly spaced, user-defined reference time points $\mathbf{r} = [r_1, r_2, \cdots, r_L]$ are used to calculate the latent values of each sensor at these reference time points**. These latent values are obtained by performing value-level extraction and feature-level aggregation (the reader may refer to [1] for more details).

Value-level extraction. **This step consists of non-linearly combining the observed/imputed values x'^k_t which is an element of X' (see Equation 9) and their corresponding mask value m_t^k, temporal variation δ_t^k (time interval between two successive timestamps s_t^k and s_{t+1}^k) and time lag penalty value $(i_t^k)_l$:**

$$(v_t^k)_l = \sigma(x'^k_t(\check{w}_1^k)_t^l + m_t^k(\check{w}_2^k)_t^l + \delta_t^k(\hat{w}_3^k)_t^l + (i_t^k)_l(\hat{w}_4^k)_t^l + (\check{b}^k)_t^l) \tag{10}$$

where $(v_t^k)_l$ can be view as the new value of x'^k_t at the reference time point r_l. $(\check{w}_{1:4}^k)_t^l$ and $(\check{b}^k)_t^l$ are learnable parameters. The time lag penalty value $(i_t^k)_p$ encodes the amount of information to consider from x'^k_t given a reference time point r_l. It is obtained as follows:

$$(i_t^k)_l = x'^k_t exp(-\gamma_l |r_l - s_t^k|) \tag{11}$$

where γ_l is the transition value associated with r_l. It is a learnable parameter.

As we noticed that the temporal variation is implicitly encoded via $(i_t^k)_l$, we have not considered δ_t^k in the equation 10 as in the original ALNN. However, we introduce a new binary indicator, p_t^k, which indicates whether x'^k_t is a padded value or not. Therefore, unlike the original ALNN, we redefined Equation 10 as follows:

$$(v_t^k)_l = \sigma(x'^k_t(\check{w}_1^k)_t^l + m_t^k(\check{w}_2^k)_t^l + p_t^k(\hat{w}_3^k)_t^l + (i_t^k)_l(\hat{w}_4^k)_t^l + (\check{b}^k)_t^l) \tag{12}$$

Feature-level aggregation. . This step aims to aggregate all $(v_t^k)_l$ values according to each sensor k. Indeed, if we have L reference time points, then we will have L representations of each univariate time series, i.e.,

$$\mathbf{v}_1^k = [(v_1^k)_1, \cdots, (v_T^k)_1]^\top; \cdots; \mathbf{v}_L^k = [(v_1^k)_L, \cdots, (v_T^k)_L]^\top \tag{13}$$

This feature-level aggregation, which we keep unchanged (i.e., as in the original ALNN), in our model is obtained as follows:

$$z_l^k = \sigma(\sum_{i=1}^{T} (v_i^k)_l(\dot{w}_i^k)^l + (\dot{b}^k)^l);\ z_l^k \in Z \tag{14}$$

where z_l^k is the final latent representation of the sensor k at the reference time point r_l. $Z(z_l^k)\mathbb{R}^{L\times K}$ is the PLMTS. $(\dot{w}_{:T}^k)^l$ and $(\dot{b}^k)^l$ are learnable parameters. Figure 5 illustrates value-level extraction and feature-level aggregation with 3 sensors a, b, c and 2 reference time points.

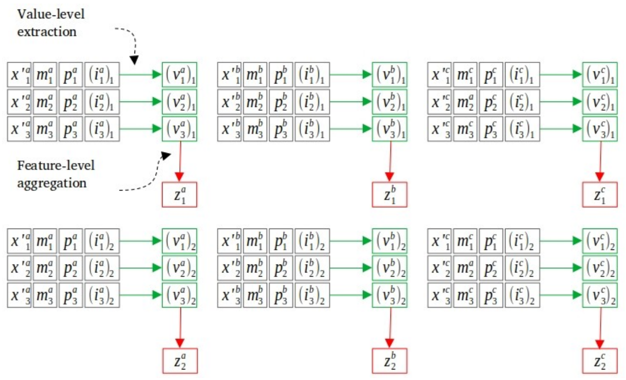

Figure 5: Illustration of value-level extraction and feature-level aggregation. a, b and c are sensor indicators. We use only two reference time points for better visualization.

In the next subsection, we present how the PLMTS Z is then sequentially transmitted to the GRU for sequential modelling and then to the classifier to obtain the final model's output.

3.4 GRU + Classifier

As in the original ALNN [1], we also use a gated recurrent unit (GRU), which works best with regular multivariate time series for sequential modelling, and a set of feedforward neural networks f stacked as a classifier (or regressor depending on the downstream task). This step, which is illustrated in Figure 6, is formulated as follows:

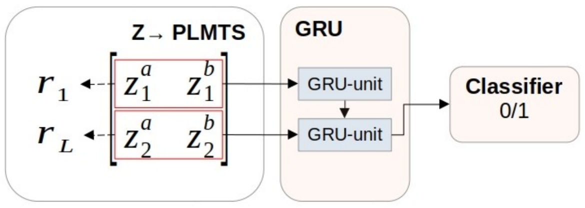

Figure 6: GRU+Classifier: The pseudo-aligned multivariate time series Z is fed into the GRU for sequential modding and substantially to the classifier.

$$\hat{y} = f_{\beta_N}^{N} \circ f_{\beta_{N-1}}^{N-1} \circ \cdots f_{\beta_0}^{0} \circ GRU_\theta(Z) \tag{15}$$

where $\circ$ is the composition symbol. $\mathcal{N}$ is the number of stacked feedforward neural networks. $\beta_{:\mathcal{N}}$ and θ are the learning parameters of feedforward neural networks and GRU, respectively. $\hat{y} \in [0, 1]$ is the likelihood value/vector or regression value/vector.

3.5 Loss functions

The proposed model incorporates three loss functions, namely: a binary cross-entropy loss $\mathcal{L}_{bc}$, that calculates the likelihood error relative to the targeted event; a mean squared error loss $\mathcal{L}_{mse1}$ that calculates the error between the imputed value and their corresponding actual values; and finally, another mean squared error loss $\mathcal{L}_{mse2}$ that calculates the error between the mixture of similarity scores H obtained from the IMTS and another H' obtained from the PLMTS. The way to get H' is the same as getting H (see Equation 5). The only different thing is that the query matrices $Q'^{j'} = Z^{\top} W'^{j'}_{q}$ and the key matrices $K'^{j'} = Z^{\top} W'^{j'}_{q}$ of H' are obtained by linearly transforming the PLMTS Z. It is thought that using $\mathcal{L}_{mse2}$ will keep the similarity score of univariate time series pairs that exist in the observed space in the latent space. We show experimentally that this loss makes it possible to improve the model's accuracy. The global loss is, then:

$$\mathcal{L} = \sum_{n=1}^{N} \underbrace{[y_n \ln(\hat{y}_n) + (1 - y_n) \ln(1 - \hat{y}_n)]}_{\mathcal{L}_{bc}} + \underbrace{\varsigma((H_n - H'_n)^2)}_{\mathcal{L}_{mse2}} + \underbrace{\varsigma((M_n \odot X_n - (1 - M_n) \odot X'_n)^2)}_{\mathcal{L}_{mse1}}$$

where $\varsigma((H_n - H'_n)^2)$ is the function that performs the element-wise addition of the resulting matrix $(H_n - H'_n)^2$ and divides this sum by the number of elements $K \times K$. It is the same for $\varsigma((M_n \odot X_n - (1 - M_n) \odot X'_n)^2)$ except that $(M_n \odot X_n - (1 - M_n) \odot X'_n)^2$ has $T \times K$ elements.

4 EXPERIMENT

We devote this section to evaluating the proposed model and comparing it to state-of-the-art models designed to handle irregular time series data. We perform this evaluation on the mortality prediction task with two different datasets. In addition, we carry out different ablation studies to show the effectiveness of the different strategies implemented.

In what follows, prior to the evaluation, we describe the different datasets used in the study, then present the environment and model setting, and finally the competing models.

4.1 Datasets

The empirical assessment is performed using data extracted from the MIMIC-3 and the PhysioNet/Computing in Cardiology Challenge 2012 databases. In both databases, the classes 1 and 0 are assigned to patients who will die and those who will stay alive, respectively.

MIMIC-3. is the anonymized health-related data of more than forty thousand patients who stayed in intensive care units at Beth Israel Deaconess Medical Center between 2001 and 2012. From its chartevents and outputevents tables, we extracted the data of patients who spent at least 48 hours in the ICU. We obtained 25, 755 patients fulfilling this condition. As a patient may have several admissions, we obtained 30, 415 admissions distributed as follows: 26, 577 with state 0 and 3, 838 (12%) with state 1. 12 time series variables describing the patient's condition are used as features. The average number of observations per feature is set to 120.

PhysioNet. is a database developed as part of a mortality prediction challenge. It includes data on patients who have been hospitalized in intensive care units for cardiac diseases. Only data from the first 48 hours after admission is available. We extracted data from 4, 000 patients. The distribution of the data is as follows: 3, 446 with state 0 and 554(13.85%) with state 1. 37 time series variables describing the patient's condition are used features. The number of observations per feature is set to 189.

5 ENVIRONMENT AND MODEL SETTING

We coded the model with Python (3.0) and its machine-learning libraries, Tensorflow and Keras. Pandas, NumPy and Scikit-learn have also been used for related tasks such as data loading, sample preparation, and metrics implementation, to name a few.

As for the model, after extensive manual search, here is how its hyperparameters were defined: The reference time point vector was set to $\mathbf{r} = [0, 1, 2, \cdots, 48]$ (see subsection 3.3). We use 15 self-attentions to obtain the mixture of similarity score matrices (see Equation 5). A dropout of 0.8 was applied after calculating the latter to prevent the overfitting problem. For the mixture of similarity score matrices of the PLMTS (see subsection 3.5), we used 10 self-attentions and applied a dropout of 0.2. We applied a dropout of 0.8 after the extra imputation. The number of GRU units was set to 168 and the classifier encompasses two Feedforward neural networks with 20 and 1 units, respectively (see Equation 15). Their activation functions are respectively *relu* and *sigmoid*. A dropout of 0.2 is applied after the one with 20 units. We set the batch size to 200 and train the model over 50 epochs. We use *Adam* as the optimizer and set the learning rate to 0.001. 10% of the observed value is used for missing values imputation.

6 COMPETING MODELS

The competing models are: DATA-GRU [24]: It is a Dual-Attention Time-Aware Gated Recurrent Unit that integrates a time aware mechanism to preserve the underlying information carry by irregular time interval and a dual-attention mechanism dedicated to missing value imputation based on data-quality and medical knowledge; Interp-net[20]: Interpolation neural network that uses a set of Radial Basis Functions to interpolate missing values against a set of reference time points; GRU-D[6]: GRU-based model with a decay mechanism applied in its core which aims to impute missing values according to the duration of their absence; mTAND [21]: It is a self-attention-based model that learns an embedding of timestamps vales through periodic functions and uses an attention mechanism to produce a fixed-length representation of a time series containing a variable number of observations; ODE-LSTM[14]: LSTM-based model whose hidden states are calculated with ordinary differential equations; SAnD [22]: It is an attention-based model that incorporates a masked self-attention mechanism and uses positional encoding and dense interpolation strategies to model the temporal order contained in clinical time series. Phased-LSTM[16]: LSTM-based model with an additional gate called time gate. This gate is dedicated to periodically updating the hidden cell and the memory cell of the LSTM according to a set of signals calculated in its different units. The ALNN model has been described in subsection 3.3.

7 RESULTS

We devote this section to comparing and interpreting the performance of our model with those of competing models. As we are dealing with unbalanced classes (alive and dead), we used the Area Under the ROC Curve (AUC) and the Area Under the Precision-Recall Curve (AUPRC) to evaluate the performance of our model against competing models over 5 cross-validation.

From Table 1, we can see that on both datasets, our model performs better than those of the competitors. This confirms the effectiveness of the different strategies we implemented. We find that ALNN is the best-performing model after ours. This shows how crucial it is to perform data-driven alignment rather than manual alignment which can be a noise driver.

Even though ALNN is the only model that uses simple imputation techniques such as mean imputation, backwardfill, etc., it is still more efficient than other competitors using a data-driven approach to fill in missing values. We can then hypothesize that these simple imputation techniques are acceptable even though they rely on strong assumptions.

We suspect that Interp-net's lower score is likely due to the interpolation it performs in its core. Indeed, this might introduce noise and change the underlying structure of the data. The superior performance of ALNN and our model over that of Interp-net and mTANd, which also used a set of reference time points, highlights the importance of considering these reference time points with respect to the period during which the data is collected. This trick further allows us to preserve temporal information.

Although we were able to provide better accuracy on the mortality prediction task than our competitors, our model is still limited by the fixed set of reference time points. With this approach, we force all data to have the same temporal trend when transformed into PLMTS. This should not be the case, as the trend varies across samples and univariate time series. Future work will therefore focus on how to make these temporal reference points different.

7.1 Ablation studies

To demonstrate the effectiveness of the different strategies implemented in our model, we carried out the following ablation studies: We use the average of the intra and extra-imputed values (see Equation 7) to obtain the final imputed value (mode = Intra and extra average); We evaluated the model without the padding mask introduced in Equation 12 (mode = w/o padding mask); and finally, we did not consider the loss function $\mathcal{L}_{mse2}$ dedicated to maintaining univariate correlations in the latent space (mode=w/o $\mathcal{L}_{mse2}$).

In Table 2, we can see that when one of the tricks we implemented is not integrated into the model, its performance drops. We see that the drop rate is more considerable with the Pysionet dataset than with MIMIC-3.

When we use the intra- and extra-imputed mean values to calculate the final imputed value, the $\mathcal{L}_{mse1}$ loss increases and can subsequently affect the performance of the model. This is why the AUC and AUPRC scores drop when the average is used rather than a balancer with a value between 0 and 1. Concerning the padding mask, the results obtained show its relevant role in attenuating the noise introduced by the padded values. Finally, we see that when the $\mathcal{L}_{mse2}$ loss is not used, the model performance also drops. Indeed, during the model calculation, certain information may be lost, in particular the information on the similarity of two univariate time series. $\mathcal{L}_{mse2}$ allows this information to be preserved so that it can flow from the IMTS to the PLMTS.

8 CONCLUSION

Throughout this paper, we have presented an ALNN variant model designed to overcome the limitations of the original ALNN model in order to improve the accuracy of downstream tasks that depend on irregular multivariate time series. In this variant, we introduced a data-driven imputation approach based on a graph neural network structure and message passing. We also incorporated a padding mask into the model calculation that mitigates the negative impact of the padded value on the model accuracy. Finally, we have implemented a loss function aimed at preserving the similarity score of univariate series that exist in the observed space in the latent space. We have shown through an extensive empirical evaluation that all these different strategies improve the accuracy of the model.

Although the performance obtained is superior to that of our competitors, an in-depth analysis of the imputed values is missing. Furthermore, even if we evaluate the model on two databases, the pilot cases are the same. We cannot therefore say whether the proposed model is generalizable. We plan to study these aspects gradually in future work.

Table 1: Models performance on the mortality prediction task (mean ± standard deviation from 5-cross validation).

	MIMIC-3		Physionet	
Model	AUC	AUPRC	AUC	AUPRC
DATA-GRU [24]	0.839 ± 0.015	0.391 ± 0.015	0.800 ± 0.016	0.399 ± 0.030
Interp-net [20]	0.754 ± 0.005	0.313 ± 0.005	0.718 ± 0.015	0.310 ± 0.005
GRU-D [6]	0.826 ± 0.004	0.463 ± 0.019	0.804 ± 0.018	0.427 ± 0.033
mTAND [21]	0.807 ± 0.004	0.442 ± 0.017	0.795 ± 0.014	0.430 ± 0.004
ODE-LSTM	0.806 ± 0.003	0.431 ± 0.016	0.718 ± 0.037	0.331 ± 0.037
Phased-LSTM [16]	0.831 ± 0.004	0.483 ± 0.021	0.813 ± 0.013	0.444 ± 0.027
SAnD [22]	0.826 ± 0.002	0.411 ± 0.010	0.791 ± 0.037	0.433 ± 0.054
STraTS [25]	0.754 ± 0.001	0.337 ± 0.002	0.726 ± 0.015	0.367 ± 0.010
ALNN-GRU [1]	0.846 ± 0.004	0.507 ± 0.013	0.833 ± 0.007	0.475 ± 0.017
Ours	**0.850 ± 0.003**	**0.517 ± 0.007**	**0.838 ± 0.008**	**0.488 ± 0.033**

Table 2: Model performance after various ablations

		MIMIC-3		Physionet	
Mode	Formula	AUC	AUPRC	AUC	AUPRC
Intra and extra average	$\bar{x}_t^k = (\hat{x}_t^k + \tilde{x}_t^k)/2$	0.846 ± 0.005	0.503 ± 0.014	0.820 ± 0.007	0.451 ± 0.032
w/o padding mask	$(v_t^k)_l = \sigma(x'^k_t(\check{w}_1^k)_t^l + m_t^k(\check{w}_2^k)_t^l + (i_t^k)_l(\hat{w}_4^k)_t^l + (\check{b}^k)_t^l)$	0.841 ± 0.06	0.492 ± 0.011	0.832 ± 0.010	0.470 ± 0.036
w/o $\mathcal{L}_{mse2}$	$\mathcal{L} = \mathcal{L}_{bc} + \mathcal{L}_{mse1}$	0.847 ± 0.033	0.506 ± 0.011	0.829 ± 0.011	0.468 ± 0.047
Full		**0.850 ± 0.003**	**0.517 ± 0.007**	**0.838 ± 0.008**	**0.488 ± 0.033**

REFERENCES

[1] Nzamba Bignoumba, Nedra Mellouli, and Sadok Ben Yahia. 2024. A new efficient alignment-driven neural network for mortality prediction from irregular multivariate time series data. *Expert Syst. Appl.*, 238, Part E, 122148. DOI: 10.1016/J.ESWA.2023.122148.

[2] Marin Bilos, Emanuel Ramneantu, and Stephan Günnemann. 2022. Irregularly-sampled time series modeling with spline networks. *CoRR*, abs/2210.10630. arXiv: 2210.10630. DOI: 10.48550/ARXIV.2210.10630.

[3] Edward De Brouwer, Jaak Simm, Adam Arany, and Yves Moreau. 2019. Gru-ode-bayes: continuous modeling of sporadically-observed time series. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*. Hanna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d'Alché-Buc, Emily B. Fox, and Roman Garnett, (Eds.), 7377–7388. https://proceedings.neurips.cc/paper/2019/hash/455cb2657aaa59e32fad80cb0b65b9dc-Abstract.html.

[4] Wei Cao, Dong Wang, Jian Li, Hao Zhou, Lei Li, and Yitan Li. 2018. BRITS: bidirectional recurrent imputation for time series. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, December 3-8, 2018, Montréal, Canada*. Samy Bengio, Hanna M. Wallach, Hugo Larochelle, Kristen Grauman, Nicolò Cesa-Bianchi, and Roman Garnett, (Eds.), 6776–6786. https://proceedings.neurips.cc/paper/2018/hash/734e6bfcd358e25ac1db0a4241b95651-Abstract.html.

[5] Wei Cao, Dong Wang, Jian Li, Hao Zhou, Lei Li, and Yitan Li. 2018. Brits: bidirectional recurrent imputation for time series. *Advances in neural information processing systems*, 31.

[6] Zhengping Che, Sanjay Purushotham, Kyunghyun Cho, David Sontag, and Yan Liu. 2018. Recurrent neural networks for multivariate time series with missing values. *Scientific reports*, 8, 1, 1–12.

[7] Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. 2018. Neural ordinary differential equations. *Advances in neural information processing systems*, 31.

[8] Ailin Deng and Bryan Hooi. 2021. Graph neural network-based anomaly detection in multivariate time series. In *Thirty-Fifth AAAI Conference on Artificial Intelligence, AAAI 2021, Thirty-Third Conference on Innovative Applications of Artificial Intelligence, IAAI 2021, The Eleventh Symposium on Educational Advances in Artificial Intelligence, EAAI 2021, Virtual Event, February 2-9, 2021*. AAAI Press, 4027–4035. DOI: 10.1609/AAAI.V35I5.16523.

[9] Eloi Garcia, Carles Serrat, and Fatos Xhafa. 2023. Breaking through the traffic congestion: asynchronous time series data integration and xgboost for accurate traffic density prediction. In *2023 Winter Simulation Conference (WSC)*. IEEE, 1747–1758.

[10] Yupu Guo, Yanxiang Ling, and Honghui Chen. 2020. A time-aware graph neural network for session-based recommendation. *IEEE Access*, 8, 167371–167382. DOI: 10.1109/ACCESS.2020.3023685.

[11] Eunji Jun, Ahmad Wisnu Mulyadi, Jaehun Choi, and Heung-Il Suk. 2020. Uncertainty-gated stochastic sequential model for ehr mortality prediction. *IEEE Transactions on Neural Networks and Learning Systems*, 32, 9, 4052–4062.

[12] Patrick Kidger, James Morrill, James Foster, and Terry Lyons. 2020. Neural controlled differential equations for irregular time series. *Advances in Neural Information Processing Systems*, 33, 6696–6707.

[13] Mathias Lechner and Ramin Hasani. 2020. Learning long-term dependencies in irregularly-sampled time series. *arXiv preprint arXiv:2006.04418*.

[14] Mathias Lechner and Ramin M. Hasani. 2020. Learning long-term dependencies in irregularly-sampled time series. *CoRR*, abs/2006.04418. https://arxiv.org/abs/2006.04418 arXiv: 2006.04418.

[15] Zachary C Lipton, David Kale, and Randall Wetzel. 2016. Directly modeling missing data in sequences with rnns: improved classification of clinical time series. In *Machine learning for healthcare conference*. PMLR, 253–270.

[16] Daniel Neil, Michael Pfeiffer, and Shih-Chii Liu. 2016. Phased lstm: accelerating recurrent network training for long or event-based sequences. *Advances in neural information processing systems*, 29.

[17] Nora El-Rashidy, Shaker El-Sappagh, Tamer Abuhmed, Samir Abdelrazek, and Hazem M El-Bakry. 2020. Intensive care unit mortality prediction: an improved patient-specific stacking ensemble model. *IEEE Access*, 8, 133541–133564.

[18] Yulia Rubanova, Ricky TQ Chen, and David Duvenaud. 2019. Latent odes for irregularly-sampled time series. arxiv. *Search in.*

[19] Siyuan Shan, Yang Li, and Junier B. Oliva. 2023. NRTSI: non-recurrent time series imputation. In *IEEE International Conference on Acoustics, Speech and Signal Processing ICASSP 2023, Rhodes Island, Greece, June 4-10, 2023*. IEEE, 1–5. DOI: 10.1109/ICASSP49357.2023.10095054.

[20] Satya Narayan Shukla and Benjamin M. Marlin. 2019. Interpolation-prediction networks for irregularly sampled time series. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. https://openreview.net/forum?id=r1efr3C9Ym.

[21] Satya Narayan Shukla and Benjamin M. Marlin. 2021. Multi-time attention networks for irregularly sampled time series. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net. https://openreview.net/forum?id=4c0J6lwQ4%5C_.

[22] Huan Song, Deepta Rajan, Jayaraman Thiagarajan, and Andreas Spanias. 2018. Attend and diagnose: clinical time series analysis using attention models. In *Proceedings of the AAAI conference on artificial intelligence* number 1. Vol. 32.

[23] Qiuling Suo, Weida Zhong, Guangxu Xun, Jianhui Sun, Changyou Chen, and Aidong Zhang. 2020. Glima: global and local time series imputation with multi-directional attention learning. In *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 798–807.

[24] Qingxiong Tan, Mang Ye, Baoyao Yang, Siqi Liu, Andy Jinhua Ma, Terry Cheuk-Fung Yip, Grace Lai-Hung Wong, and PongChi Yuen. 2020. Data-gru: dual-attention time-aware gated recurrent unit for irregular multivariate time series. In *Proceedings of the AAAI Conference on Artificial Intelligence* number 01. Vol. 34, 930–937.

[25] Sindhu Tipirneni and Chandan K Reddy. 2022. Self-supervised transformer for sparse and irregularly sampled multivariate clinical time-series. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 16, 6, 1–17.

[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

[27] Yifan Wang, Fanliang Bu, Xiaojun Lv, Zhiwen Hou, Lingbin Bu, Fanxu Meng, and Zhongqing Wang. 2023. Attention-based message passing and dynamic graph convolution for spatiotemporal data imputation. *Scientific Reports*, 13, 1, 6887.

[28] Zhen Wang, Yang Zhang, Ai Jiang, Ji Zhang, Zhao Li, Jun Gao, Ke Li, Chenhao Lu, and Zujie Ren. 2021. Improving irregularly sampled time series learning with time-aware dual-attention memory-augmented networks. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management*, 3523–3527.

[29] Tingyi Wanyan, Hossein Honarvar, Ariful Azad, Ying Ding, and Benjamin S Glicksberg. 2021. Deep learning with heterogeneous graph embeddings for mortality prediction from electronic health records. *Data Intelligence*, 3, 3, 329–339.

[30] Bin Yang, Le Qin, Jianqiang Liu, and Xinxin Liu. 2022. IRCNN: an irregular-time-distanced recurrent convolutional neural network for change detection in satellite time series. *IEEE Geosci. Remote. Sens. Lett.*, 19, 1–5. DOI: 10.1109/LGRS.2022.3154894.

[31] Jinsung Yoon, William R. Zame, and Mihaela van der Schaar. 2019. Estimating missing data in temporal data streams using multi-directional recurrent neural networks. *IEEE Trans. Biomed. Eng.*, 66, 5, 1477–1490. DOI: 10.1109/TBME.2018.2874712.

[32] Xiang Zhang, Marko Zeman, Theodoros Tsiligkaridis, and Marinka Zitnik. 2022. Graph-guided network for irregularly sampled multivariate time series. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net. https://openreview.net/forum?id=Kwm8I7dU-l5.