Graph Neural Network-Based Representation Learning for Medical Time Series

Zhuzi Zheng[1], Changchun Guo[2], Jianyong Chen[1(✉)], and Jianqiang Li[1]

[1] College of Computer Science and Software Engineering, Shenzhen University, Shenzhen 518000, China
jychen@szu.edu.cn
[2] Shenzhen Pingshan People's Hospital, Shenzhen 518000, China

Abstract. The ability to analyze and predict medical time series data is crucial for enhancing healthcare decision-making and improving patient outcomes. Currently, the algorithms used for classification and prediction of medical time series data are limited in their capabilities and may not be reliable enough to meet the demands of practical applications. The purpose of this paper is to promote the representation learning of complex data primarily comprised of medical time series, in order to facilitate various downstream tasks. Under the framework of graph neural networks (GNN), we present indegree regularized neural message passing to reflect the dependencies between different sequences. Our approach also leverages representation learning to convert multivariate time series (MTS) and static features into nodes of GNN. Moreover, we propose a dynamic loss function to encourage the consistent learning of sensor dependency graphs across models. Based on these proposals, our method can effectively capture not only the temporal dependencies among variables, but also the multidimensional dependencies among MTS and static features. We classify time series on two medical challenge and a human activity datasets. The results show that our approach can significantly improve downstream task performance across various metrics. Code is available at https://github.com/Zzzoptimus/GICG.

Keywords: Graph neural network · Medical time series · Regularized neural message passing · Time series classification

1 Introduction

The significance of classification and prediction related to multiple medical time series lies in their ability to assist medical professionals in making informed decisions about patient care. By analyzing and predicting medical time series data, healthcare providers can identify patterns and trends that may be indicative of certain conditions or diseases, and take appropriate action to mitigate or treat those conditions. For example, the early detection of certain diseases

L. Iliadis et al. (Eds.): ICANN 2023, LNCS 14259, pp. 194–205, 2023.
https://doi.org/10.1007/978-3-031-44223-0_16

can allow for timely intervention and improve treatment outcomes. Overall, the ability to classify and predict medical time series data can lead to more accurate diagnoses, earlier detection of health issues, and better treatment outcomes for patients. However, the current state of algorithms utilized for classifying and predicting medical time series data is limited in terms of its abilities and may not be dependable enough to fulfill practical application requirements.

The dataset we use mainly consists of medical time series. In medical scenarios, the various physiological indicators of the human body, such as heart rate, blood pressure, and oxygen saturation, are interdependent time series. Changes in heart rate can impact blood pressure, while high blood pressure can cause harm to heart function, elevating the risk of cardiovascular events like myocardial infarction and stroke in individuals with rapid heartbeat. Additionally, the static characteristics of the human body, including age and weight, can also influence these indicators. Moreover, if the influence between two physiological indicators in the human body changes, it can also indicate that there are problems in certain parts of the body. The changes in the influence between these sensors can be represented using message passing in graph neural networks (GNN).

The message passing neural network [5] was proposed in 2017. Each node and edge in the graph is associated with a learnable feature vector. The network iteratively passes messages between the nodes in a graph structure, updating their feature vectors based on the information received from neighboring nodes. This iterative process allows the information to be shared and aggregated across the graph, enabling GNN to model complex relationships and dependencies between nodes. It have been successfully applied to a wide range of tasks, including molecule property prediction, social network analysis, and protein structure prediction. They are particularly useful for problems where the input data has an inherent graph structure and traditional neural networks cannot be directly applied. We improve the message passing in GNN to make it more suitable for our task. In summary, the main contributions of this paper are as follows:

- We propose a regularized message passing strategies. A technique is proposed to apply indegree regularized constraint on the message passing of nodes.
- We propose global graph node representation learning. The static and the time series features can be kept in a consistent feature space after representation.
- A dynamic loss function is proposed that can promote consistent learning of sensor dependency graphs by establishing the similarity between two self-connections graphs.
- We conduct experimental analysis on three public large-scale datasets. The performance of our proposed model is significantly improved compared to the other baselines.

2 Related Works

In this chapter, we will demonstrate the relevant work from two perspectives: the data analysis about multivariate time series (MTS) and the development of GNN.

2.1 Multivariate Time Series

MTS are a crucial and ubiquitous type of data that represent multiple time-varying variables or signals recorded over time. They are widely used in a diverse range of domains, such as finance to analyze stock prices, meteorology to forecast weather patterns, medicine to monitor vital signs of patients, and many others. Irregularity in a multivariate case can create challenges for models that expect well-aligned and fixed-size inputs. Observations can be misaligned across different sensors and the number of observations can also vary considerably across samples due to a multitude of sampling frequencies and varying time intervals [24]. These characteristics can further complicate the analysis. An intuitive way to deal with irregular time series is to impute missing values and process them as regular time series [4]. However, imputation methods can distort the underlying distribution and lead to unwanted distribution shifts.

Historically, linear regression [6], random forest [9], and support vector machines [25] have been widely adopted for modeling and predicting MTS in academic research. Thanks to the advancement of computing power, state-of-the-art deep learning architectures have been developed to analyze MTS in various domains. In particular, these models have shown remarkable performance in analyzing and forecasting medical MTS. Recurrent neural networks (RNNs) [2], auto-encoders (AE) [11], and generative adversarial networks (GANs) [23] have demonstrated remarkable performance in medical data imputation and prediction, leveraging their powerful learning and generalization capabilities achieved through complex nonlinear transformations. Recent advancements in the field have led to the development of models that can directly learn from irregularly sampled time series. For instance, Che et al. developed a decay mechanism based on gated recurrent units (GRU-D) [1] and binary masking to capture long-range temporal dependencies. SeFT [8] takes a set-based approach and transforms irregularly sampled time series datasets into sets of observations modeled by set functions insensitive to misalignment. mTAND [16] leverages a multi-time attention mechanism to learn temporal similarity from non-uniformly collected measurements and produce continuous-time embeddings. IP-Net [14] adopt imputation to interpolate irregular time series against a set of reference points using a kernel-based approach.

2.2 Graph Neural Network

GNN are a type of deep learning model that can capture the structural and relational information of graphs [22,27]. It is an optimizable transformation on all attributes of the graph (nodes, edges, global-context) that preserves graph symmetries (permutation invariances) [20]. The graph's description is in a matrix format that is permutation invariant. All nodes, edges and global-context can be represented by vectors. This type of representation can be used for various applications, such as predicting molecular properties, reasoning with graphs and relations, and many others.

Recently, GNN have emerged as a successful approach to model complex patterns in structured graph data and a variety of improved GNNs have been reported. Graph convolutional networks (GCN) model node feature representations by aggregating representations of their one-step neighbors. Building upon this, graph attention networks (GAT) [19] use attention functions to compute different weights for different neighbors during aggregation. GNN-based models have shown success in time-related tasks, such as traffic prediction [21]. Applications of GNN in recommendation systems [10] and related domains have demonstrated their effectiveness in modeling large-scale, multi-relational data. Raindrop [26] is introduced as a graph-guided network for irregularly sampled time series and cases where sensor data is missing.

3 Method

In order to efficiently capture the relationships between various types of data and provide reference for medical decision-making, we propose a novel architecture named global indegree consistency GNN(GICG), which takes each sample (In clinical data, a patient's health status is recorded at irregular time intervals using different sensors) as input, where each sample is represented as a weighted directed graph (see Fig. 1). In each graph, nodes are formed by the irregularly recorded observations of each sensor, while the edge weights of the message passing between these nodes are learned during training. GICG aims to learn a fixed-dimensional embedding vector for a given sample and predict relevant labels based on this vector. Previous research [3,26] has represented MTS as feature vectors of nodes, and then through message passing, resulted in a global feature vector for the graph. Then it is concatenated with the static features of each sample to form a combined feature vector for downstream tasks. However, this concatenation approach is not a good choice because the processed time series feature vectors and the static features have undergone different mappings and transformations, and they reside in different high-dimensional feature spaces with potentially different data scales. In fact, the static features are similar with the time series data with little or no temporal variation. Therefore, we have adopted a consistent approach to process both types of data as time series data and represent them as different nodes (see Fig. 1 "Static" and "Active") in GNN.

3.1 Global Graph Node Representation

In order to construct sensor dependency graphs, consider a dataset $D = \{(S_i, y_i) \mid i = 1, \ldots, N\}$ consisting of N labeled samples, where each sample S_i is an irregular MTS with a corresponding label $y_i \in \{1, \ldots, C\}$. The label y_i denotes the class membership of the sample S_i among C istinct classes. Each sample contains M_a uniformly sampled sensors and M_s static features, resulting in M nodes ($M = M_a + M_s$) denoted as u, v, and so on. Each node is represented by a time-sorted observation sequence, with static features having a constant observation value at every timestamp. And d_k is denoted as cardinality

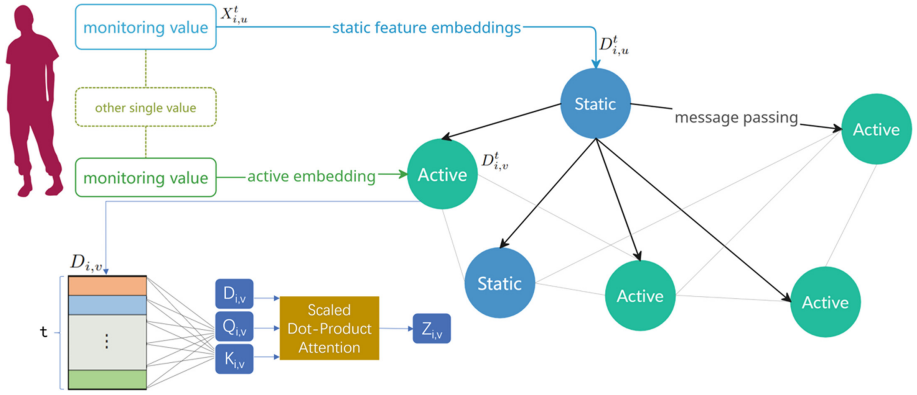

Fig. 1. The overview architecture of our model. Based on observed single value $X^t_{i,u}$, we generates the sensor embedding $Z_{i,u}$. The "Static" node represents the static feature representation, while the "Active" nodes represent the original time series data. The graph is updated through message passing to obtain the feature matrix $D_{i,v}$ of a node at all timestamps. Finally, the sensor embedding $Z_{i,v}$ is obtained through the attention mechanism. For simplicity, we have omitted the time information and the layer index of multi-layer message passing in the graph.

of the set comprising all timestamps included in the time series. For a node u in sample S_i, a single observation is represented as a tuple $(t, X^t_{i,u})$, where the value $X^t_{i,u} \in \mathbb{R}$ is recorded for the node u at timestamp $t \in \mathbb{R}^+$. For nodes representing time series, the observation values are irregularly recorded, meaning that the time intervals between consecutive observation values may vary across nodes.

In order to map observations from both active sensors and static features to a high-dimensional space, we apply a nonlinear transformation. Due to the possibility of different distributions being followed by recorded values at different sensors, this is achieved through a trainable weight vector W_u, which depends on different nodes but is shared across samples and time dimensions. W_u transforms observations X from different nodes into a fixed-length vector of d, which is then mapped to a high-dimensional space through a nonlinear transformation using the following formula:

$$D^t_{i,u} = Sigmoid(X^t_{i,u} W_u) \tag{1}$$

3.2 Regularized Message Passing

Regularized message passing strategies are used to improve the consistent learning of sensor dependency graphs across different samples. They generate embeddings for inactive neighboring sensors, enhancing the quality and robustness of the learned embeddings. (see Fig. 1 "message passing"). These techniques aim to promote similarity between two self-connection graphs and ensure that the learning process remains stable and accurate. Following the previous step shown

in Eq. (1), we now have fixed-length embeddings for each sensor at every timestamp. To update the weights of node v, we consider its dependencies with other nodes. This approach enables us to provide reasonable values for nodes with missing data at a particular timestamp, leveraging the inter dependency among nodes. Let us consider nodes u and v as an example to calculate the proportion of the information propagated from u to v with the following equation:

$$P_{i,uv}^t = Sigmoid(D_{i,u}^t W[r_u||p_i^t]^T), \tag{2}$$

where r_u refers to the outdegree vector of node v. It enables the model to learn different attention weights for distinct edges from various sensors. Moreover, $P_i^t \in \mathbb{R}^+$ represents time by mapping the 1-dimensional timestamp t to a multi-dimensional vector using trigonometric functions. We use P_i^t to compute attention weights that are sensitive to time. And W is a trainable weight matrix that is used for dimension mapping. The calculation result $P_{i,uv}^t \in [0, 1]$ is attention weight from u to v. Combining all these components, we can compute the embedding $D_{i,v}^t$ with the neighbor v of node u as follows:

$$D_{i,v}^t = Gelu(D_{i,u}^t W_u W_v^T P_{i,uv}^t c_{i,uv}) \tag{3}$$

The edge weight vector $c_{i,uv}$ is shared across all timestamps. Moreover, it will be updated in the following sections to achieve more efficiency in regularized message passing. We utilize regularized message passing by updating the edge weight vector $c_{i,uv}$. Firstly, we calculate the exponential sum of in-degrees $c_{i,v}$ of node v: $c_{i,v} = \sum_{u=1}^{M}(e^{c_{i,uv}/T})$. Then we use an activation function with distillation temperature to obtain the weight propagated to node v. The formula is shown as follows:

$$c_{i,uv} = \frac{e^{c_{i,uv}/T}}{\sum_{u=1}^{M}(e^{c_{i,uv}/T})}, \qquad \text{for} \quad u = 1, 2, \ldots, M \tag{4}$$

In the above equation, T is the distillation temperature [7], which controls the smoothness of the connections to node v. A higher T leads to a smoother result. For each outgoing node u and incoming node v, we traverse all M nodes in the graph and update the weights of all edges connecting them.

3.3 Sensor Embedding

After performing the previous operations, we obtained the high-dimensional vector $D_{i,v}$ for node v across all timestamps. Utilizing the time attention weights introduced in this section, we compressed $D_{i,v}$ along its dimensions to obtain the sensor embedding $S_{i,v}$(see Fig. 1). It is noteworthy that the impact of different timestamps on vector $S_{i,v}$ varies. To be specific, we compute in parallel the significance of node v at each timestamp using the following formula:

$$A_{i,v} = softmax(\frac{Q_{i,v}K_{i,v}^T}{\sqrt{d_k}}W_s)D_{i,v}, \tag{5}$$

where $Q_{i,v}, K_{i,v}$ are mapped by the formula: $Q_{i,v} = D_{i,v}W_Q, K_{i,v} = D_{i,v}W_K$. d_k is the dimension of unique timestamps in all time series. The equation is divided by $\sqrt{d_k}$ in order to maintain the dimensionality of attention weights within a more favorable range of numerical values, where $W_s \in \mathbb{R}^{d\times 1}$ is introduced to facilitate the dimension reduction operation. We will generate a temporal attention weight vector instead of self-attention matrix. Subsequently, we perform residual connections and transformations on the temporal embedding with the following formula:

$$S_{i,u} = \sum_{i=1}^{t}(A_i^t||[W_p p_i^t]W_a), \tag{6}$$

where W_p is a linearly mapped matrix. After the mapping process, we concatenate the resulting matrices and A_i^t to form a single matrix, where W_p and W_a are two linear projector shared by all samples and sensors.

3.4 Graph Embedding

The values of node embeddings are highly informative to some extent, and the concatenation of all node vectors results in a vector of only thousands of dimensions. Therefore, there is no need to further compress information, and we can directly use the concatenated vector S_i as the feature vector for the sample i. Good performance can be achieved by adding a simple fully connected layer. This approach works perfectly fine when the number of nodes is small, but for cases with a large number of nodes, it is necessary to use some feature extraction methods.

3.5 Dynamic Loss Function

In order to promote consistent learning of sensor dependency graphs, we employ the following loss function: $L = L_{CE} + pL_s$, where L_{CE} is cross entropy loss. The dissimilarity L_s is computed for identical sensors across different graphs (samples). L_s is calculated with the following formula:

$$L_s = \frac{\sum_{i=1}^{N}\sum_{j=1}^{N}\sum_{v=1}^{M}||S_{i,v} - S_{j,v}||_2}{(N-1)^2 M}, \tag{7}$$

where N denotes the total number of samples and M denotes the number of nodes in each sample. p is calculated with formula: $p = \sqrt{M/T}/2$, where T is the length of sequences. As the number of samples (N) can be large, L_s is typically computed only for samples within a batch to improve efficiency.

4 Experiments

4.1 Datasets

These datasets are all related to healthcare and human activity. The purpose of them is to enable early disease screening or advance prediction of ICU mortality in patients. Here, we provide a brief overview of the datasets used in the following experiments.

(1) P19: PhysioNet Sepsis Early Prediction Challenge 2019. P19 dataset includes 40,336 patients and each patient contains MTS by 34 irregularly sampled sensors. Each patient has a binary label representing occurrence of sepsis in the next 6 h. The dataset is imbalanced with 4% positive samples [13].

(2) P12: PhysioNet Predicting Mortality Challenge 2012. P12 dataset contains 12, 000 patients. Each patient contains MTS with 36 sensors. Each sample has a static vector with 9 elements including age, gender, etc. Each patient is associated with a binary label indicating length of stay in ICU. P12 is imbalanced with only 7% negative samples [17].

(3) PAM: PAMAP2 Physical Activity Monitoring. PAM dataset measures daily living activities of 9 subjects with 3 inertial measurement units. PAM dataset contains 5, 333 samples. Each sample is measured by 17 sensors and contains 600 continuous observations. PAM does not include static attributes and the samples are approximately balanced across all 8 categories [12].

We divide the dataset randomly into training (80%), validation (10%), and test (10%) sets. We split the dataset in five random ways according to the above ratio, then we performed five independent experiments.

4.2 Baselines

For comparison, we consider several models including decay mechanism based on GRU-D and binary masking, SeFT's set-based modeling, and the transformer-based method trans-mean. We will also compare our approach to mTAND's multi-time attention mechanism, as well as graph-guided network Raindrop. Additionally, we will evaluate IP-Net imputation-based methods, which interpolate irregular time series using kernel-based techniques against a set of reference points.

4.3 Evaluation Metrics

In binary classification (P19 and P12), we will use classification Accuracy, AUROC (Area Under the Receiver Operating Characteristic Curve), and AUPRC (Area Under the Precision-Recall Curve) as evaluation index. AUROC measures the ability of a model to distinguish between positive and negative classes by calculating the area under the curve of the receiver operating characteristic (ROC) curve, which plots the true positive rate (sensitivity) against the false positive rate (1-specificity) at various threshold settings. AUPRC, on the other hand, measures the trade-off between precision and recall of a model by calculating the area under the curve of the precision-recall curve, which plots the precision against the recall at various threshold settings.

We use both AUROC and AUPRC metrics simultaneously because while AUROC is useful for binary classification with balanced or imbalanced class distribution, it may not be optimal for imbalanced datasets. On the other hand, AUPRC is particularly useful for imbalanced classification problems, but can be less sensitive to false negatives and challenging to optimize precision and recall simultaneously.

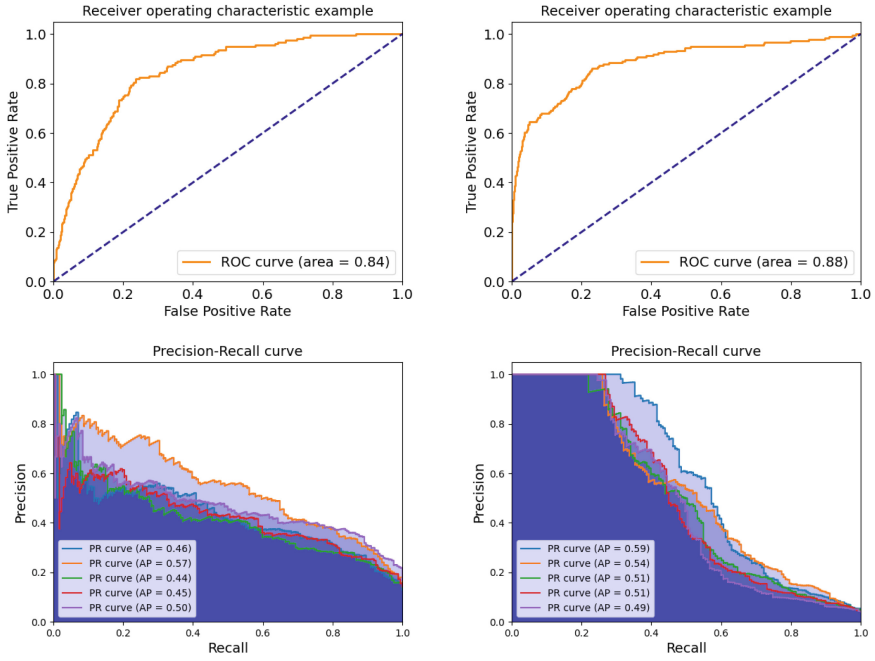

Fig. 2. The ROC curves and PRC curves of GICG tests on binary classification datasets, with the P12 dataset on the left and the P19 dataset on the right. On the PRC curves, average precision (AP) is the area under the curve, and the five curves represent five independent tests.

As shown in Fig. 2, the AUROC of GICG on the P12 and the P19 datasets reached 0.84 and 0.88, respectively. The average AUPRC of five experiments on the P12 and the P19 datasets reached 0.48 and 0.53, respectively.

4.4 Experimental Comparison Results

We conducted experiments on the aforementioned three datasets and identified optimal experimental settings and hyperparameter optimization for each dataset. Our experimental results are compared with multiple baselines on all the three datasets, demonstrating the superiority of our proposed approach.

Table 1. Comparison with the state-of-the-art methods on the datasets P12 and P19. It is the average of 5 experiments. Values with bold fonts indicate the best performance of all results.

Models	P12			P19		
	ACC	AUROC	AUPRC	ACC	AUROC	AUPRC
GRU-D [1]	0.763	0.672	0.460	0.785	0.839	0.469
Transformer [18]	0.752	0.651	0.429	0.778	0.832	0.476
Trans-mean [15]	0.764	0.668	0.449	0.802	0.841	0.474
SeFT [8]	0.719	0.668	0.361	0.786	0.787	0.311
mTAND [16]	0.731	0.653	0.403	0.780	0.804	0.324
Raindrop [26]	0.759	0.722	0.453	0.831	0.862	0.516
IP-Net [14]	0.765	0.725	0.441	0.813	0.846	0.381
GICG	**0.795**	**0.836**	**0.485**	**0.868**	**0.876**	**0.525**

In accordance with Table 1, our method GICG has achieved the highest performance on the binary classification datasets P12 and P19. This indicates that GICG can better capture the interrelationships between different variables and has stronger capabilities to aggregate information over the time dimension. To further substantiate our idea, We conducted further experiments on the eight-class classification dataset PAM.

Table 2. Comparison with the state-of-the-art methods on the PAM dataset. For every value, its left-hand side displays the average of 5 experiments, while its right-hand side represents the standard deviation.

PAM	Accuracy	Precision	Recall	F1 score
GRU-D [1]	0.833 ± 0.02	0.846 ± 0.01	0.852 ± 0.02	0.848 ± 0.01
Transformer [18]	0.835 ± 0.02	0.848 ± 0.02	0.860 ± 0.01	0.850 ± 0.01
Trans-mean [15]	0.837 ± 0.02	0.849 ± 0.03	0.864 ± 0.02	0.851 ± 0.02
SeFT [8]	0.671 ± 0.02	0.700 ± 0.02	0.682 ± 0.02	0.685 ± 0.02
mTAND [16]	0.746 ± 0.04	0.743 ± 0.04	0.795 ± 0.03	0.768 ± 0.03
Raindrop [26]	0.870 ± 0.01	0.889 ± 0.02	0.886 ± 0.02	0.886 ± 0.01
IP-Net [14]	0.843 ± 0.04	0.756 ± 0.02	0.779 ± 0.02	0.766 ± 0.03
GICG	**0.889 ± 0.01**	**0.902 ± 0.01**	**0.906 ± 0.01**	**0.903 ± 0.01**

As show in Table 2, We compare the performance of different models for the classification of irregularly sampled time series, and provide a summary of the average accuracy, precision, recall, and F1 score across five experiments. GICG has demonstrated the best performance on all evaluation measures. The performance has the lowest standard deviation from the average value, as implies

high stability of the performance. Therefore, our approach can be regarded as the better choice for irregularly sampled time series classification tasks.

5 Conclusion

In this paper, we propose GICG, an improved approach based on GNN for handling medical MTS data. GICG utilizes representation learning to capture the temporal and multidimensional dependencies in medical MTS data and static features with time-invariant properties. This is achieved through the integration of global regularized message passing. GICG provides a flexible and robust solution for various downstream tasks of complex time series data. We demonstrate the effectiveness of our approach through experiments on two healthcare and a human activity datasets, showing significant improvements in downstream task performance across various metrics. This can lead to better performance and more reliable results in a wide range of applications.

Acknowledgements. This work was supported in part by the National Key R&D Program of China under Grant 2020YFA0908700, in part by the National Nature Science Foundation of China under Grant U2013201 and in part by the Pearl River Talent Plan of Guangdong Province under Grant 2019ZT08X603.

References

1. Che, Z., Purushotham, S., Cho, K., Sontag, D., Liu, Y.: Recurrent neural networks for multivariate time series with missing values. Sci. Rep. **8**(1), 6085 (2018)
2. Cui, R., Liu, M., Initiative, A.D.N., et al.: RNN-based longitudinal analysis for diagnosis of Alzheimer's disease. Comput. Med. Imaging Graph. **73**, 1–10 (2019)
3. Deng, A., Hooi, B.: Graph neural network-based anomaly detection in multivariate time series. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 35, pp. 4027–4035 (2021)
4. Du, W., Côté, D., Liu, Y.: SAITS: self-attention-based imputation for time series. Expert Syst. Appl. **219**, 119619 (2023)
5. Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E.: Neural message passing for quantum chemistry. In: International Conference on Machine Learning, pp. 1263–1272. PMLR (2017)
6. Godfrey, K.: Simple linear regression in medical research. N. Engl. J. Med. **313**(26), 1629–1636 (1985)
7. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network. arXiv preprint 1503.02531 (2015)
8. Horn, M., Moor, M., Bock, C., Rieck, B., Borgwardt, K.: Set functions for time series. In: International Conference on Machine Learning, pp. 4353–4363. PMLR (2020)
9. Khalilia, M., Chakraborty, S., Popescu, M.: Predicting disease risks from highly imbalanced data using random forest. BMC Med. Inform. Decis. Mak. **11**, 1–13 (2011)

10. Kumar, I., Hu, Y., Zhang, Y.: EFLEC: efficient feature-leakage correction in GNN based recommendation systems. In: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 1885–1889 (2022)
11. Lee, J., Sun, S., Yang, S.M., Sohn, J.J., Park, J., Lee, S., Kim, H.C.: Bidirectional recurrent auto-encoder for photoplethysmogram denoising. IEEE J. Biomed. Health Inform. **23**(6), 2375–2385 (2018)
12. Reiss, A., Stricker, D.: Introducing a new benchmarked dataset for activity monitoring. In: 2012 16th International Symposium on Wearable Computers (2012). https://doi.org/10.1109/iswc.2012.13
13. Reyna, M., et al.: Early prediction of sepsis from clinical data: the physionet/computing in cardiology challenge 2019. In: 2019 Computing in Cardiology Conference (CinC), Computing in Cardiology Conference (CinC) (2020). https://doi.org/10.22489/cinc.2019.412
14. Shukla, S.N., Marlin, B.M.: Interpolation-prediction networks for irregularly sampled time series. arXiv preprint arXiv:1909.07782 (2019)
15. Shukla, S.N., Marlin, B.M.: A survey on principles, models and methods for learning from irregularly sampled time series. arXiv preprint arXiv:2012.00168 (2020)
16. Shukla, S.N., Marlin, B.M.: Multi-time attention networks for irregularly sampled time series. arXiv preprint arXiv:2101.10318 (2021)
17. Silva, I., Moody, G., Scott, D.J., Celi, L.A., Mark, R.G.: Predicting in-hospital mortality of ICU patients: the physionet/computing in cardiology challenge 2012. In: 2012 Computing in Cardiology, pp. 245–248. IEEE (2012)
18. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
19. Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Lio, P., Bengio, Y., et al.: Graph attention networks. Stat **1050**(20), 10–48550 (2017)
20. Wang, C., Qiu, Y., Gao, D., Scherer, S.: Lifelong graph learning. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 13719–13728 (2022)
21. Wang, X., et al.: Traffic flow prediction via spatial temporal graph neural network. In: Proceedings of the Web Conference 2020, pp. 1082–1092 (2020)
22. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Philip, S.Y.: A comprehensive survey on graph neural networks. IEEE Trans. Neural Netw. Learn. Syst. **32**(1), 4–24 (2020)
23. Yan, B., Wang, H., Wang, X., Zhang, Y.: An accurate saliency prediction method based on generative adversarial networks. In: 2017 IEEE International Conference on Image Processing (ICIP), pp. 2339–2343. IEEE (2017)
24. Zerveas, G., Jayaraman, S., Patel, D., Bhamidipaty, A., Eickhoff, C.: A transformer-based framework for multivariate time series representation learning. In: Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, pp. 2114–2124 (2021)
25. Zhang, G.: A modified SVM classifier based on RS in medical disease prediction. In: 2009 Second International Symposium on Computational Intelligence and Design, vol. 1, pp. 144–147. IEEE (2009)
26. Zhang, X., Zeman, M., Tsiligkaridis, T., Zitnik, M.: Graph-guided network for irregularly sampled multivariate time series. arXiv preprint arXiv:2110.05357 (2021)
27. Zhou, J., et al.: Graph neural networks: a review of methods and applications. AI Open **1**, 57–81 (2020)