TIME SERIES ARE IMAGES: VISION TRANSFORMER FOR IRREGULARLY SAMPLED TIME SERIES

Anonymous authors
Paper under double-blind review

ABSTRACT

Irregularly sampled time series are often observed in medical applications. Highly customized models have been developed to tackle the irregularity. In this work, we propose a simple yet effective approach that transforms irregularly sampled time series into line graph images and adapts vision transformers to perform time series classification in a way similar to image classification. Our approach simplifies the model design without assuming prior knowledge. Despite its simplicity, we show that it is able to outperform state-of-the-art specialized algorithms on several popular healthcare and human activity datasets, especially in the challenging leave-sensors-out setting where a subset of variables are masked during testing. We hope this work could provide beneficial insight into leveraging fast-evolving computer vision techniques in the time series analysis domain.

1 INTRODUCTION

Time series data are ubiquitous in a wide range of domains, including healthcare, finance, traffic, and climate science. With the advances in deep learning architectures such as LSTM (Graves, 2012), Temporal Convolutional Network (TCN) (Lea et al., 2017), and Transformer (Vaswani et al., 2017), numerous algorithms have been developed for time series analysis. However, these methods typically assume fully observed data points at regular intervals. They cannot deal with irregularly sampled ones, a sequence of samples with irregular intervals between their observation times. To respond to this challenge, highly specialized models were developed, which require a considerable amount of prior knowledge in the model architecture choice and design (Marlin et al., 2012; Lipton et al., 2016; Che et al., 2018; Horn et al., 2020; Shukla & Marlin, 2020; Zhang et al., 2022).

The recently emerging transformer-based vision models, most notably Vision Transformers (Dosovitskiy et al., 2020)[1], have demonstrated strong performance on various vision tasks such as image classification and object detection. In this paper, we raise a simple question: Since vision transformers have exceeded humans in various image recognition tasks, are they able to "visually" capture temporal patterns in irregularly sampled time series data? To answer this question, we take the following minimalist approach: Transform the irregularly sampled multivariate time series into line graphs (Fig. 1), arrange these line graphs into an image, and train a vision transformer to perceive the image and perform the classification task. We dub this approach **ViTTs**, short for **Vi**sion **T**ransformer for **T**ime **s**eries. Note that although some prior studies share a similar idea of transforming time series into images for feature extraction (Wang & Oates, 2015a), they are not domain agnostic and require domain knowledge in designing specialized imaging methods such as Gramian fields (Wang & Oates, 2015a), recurring plots (Hatami et al., 2018; Tripathy & Acharya, 2018), and Markov transition fields (Wang & Oates, 2015b). By contrast, we transform time series into line graphs without assuming prior knowledge.

The line graph image encodes informative patterns in multivariate time series: (1) Temporal dynamics of each variable in its corresponding line graph; and (2) the correlation among variables across different line graphs. To help the vision transformers better capture these patterns, we introduce ***temporal position embedding*** to provide fine-grained information about the local and global contexts. Experimental results on three popular public healthcare and human activity datasets demonstrate that

[1] In this paper, we refer to vision transformers as a type of vision models based on Transformer, including ViT (Dosovitskiy et al., 2020), DeiT (Touvron et al., 2021), Swin Transformer (Liu et al., 2021), to name a few.

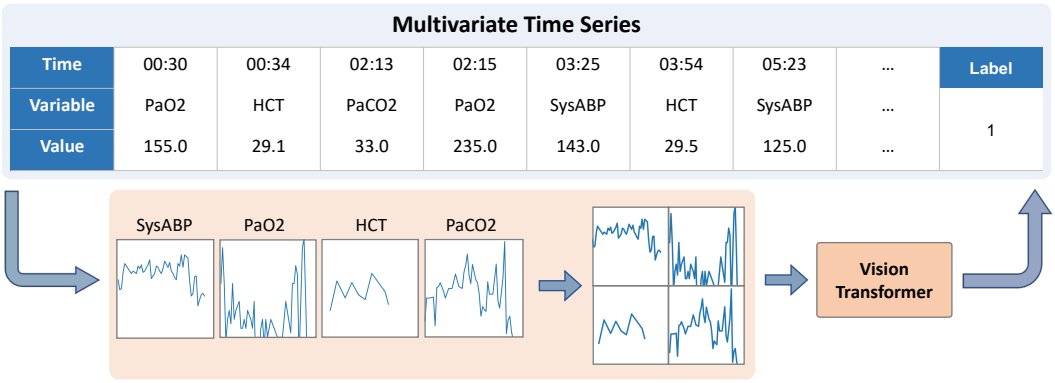

Figure 1: An illustration of our approach **ViTTs**. Shown in the figure is a sample from a healthcare dataset P12 (Goldberger et al., 2000), which provides the irregularly sampled observations of 36 variables for patients (we only show 4 variables here for simplicity). Each column in the table is an observation of a variable, with the observed time and value. We plot separate line graphs for each variable and arrange them into an image, which is then fed into the Vision Transformer to perform the classification task.

ViTTs outperforms baselines by up to 8.2 accuracy points. It is also robust to missing observations. It improves the prior work by up to 54.0% in absolute F1-score points in the leave-sensors-out setting where a part of the sensors (variables) in the test set are masked. We further evaluate on regular time series. Our approach achieves competitive results, demonstrating its generality as regularly sampled time series algorithms usually do not work well for irregularly sampled data, and vice versa.

In summary, the contributions of this work are three-fold: (1) We propose a straightforward vision transformer-based approach for multivariate irregularly sampled time series classification. The simplicity contrasts with the state-of-the-art performance it has achieved. (2) The proposed approach can be potentially extended as a general-purpose method to deal with other kinds of time series analysis tasks. It adds to the collection of works exploring artificial general intelligence (AGI) to deal with tasks in various domains instead of dedicated designs for each task. (3) It provides a new perspective for time series modeling, which might encourage the utilization of fast-evolving and well-studied computer vision techniques in the time series domain, such as better model architecture (Liu et al., 2022), data augmentation (Shorten & Khoshgoftaar, 2019), interpretability (Chefer et al., 2021), self-supervised learning (Chen et al., 2021), to name a few.

2 Related Work

Irregularly Sampled Time Series. An irregularly sampled time series is a sequence of observations with irregular intervals between observation times. Different variables can be misaligned within the same time series in a multivariate setting. Such characteristics have posed a significant challenge to the standard time series modeling methods, which typically assume fully observed and regularly sampled data points.

In recent years, there has been a surge of work that develops highly specialized neural networks to deal with irregularly sampled time series. Some methods discretize continuous-time observations into fixed time intervals and utilize probabilistic clustering models (Marlin et al., 2012) or Recurrent Neural Networks (RNN) (Lipton et al., 2016). To incorporate the dynamics between observations, GRU-D Che et al. (2018) decays the hidden states based on gated recurrent units (GRU) (Chung et al., 2014), which takes as input the observations' values and also times. Pham et al. (2017) modified the forget gate of LSTM (Graves, 2012) to capture the irregularity. Similarly, Yoon et al. (2017) proposed an approach based on multi-directional RNN, which can capture the inter- and intra-steam patterns. IP-Net (Shukla & Marlin, 2018) utilizes semi-parametric interpolation layers for interpolation and GRU for prediction. Neural ordinary differential equations (neural ODEs) (Chen et al., 2018) are used to model the continuous dynamics of a hidden state. For example, ODE-RNN (Rubanova et al., 2019) uses RNN gates to update the hidden states when there is an observation.

Besides the recurrent and differential equation-based architectures, recent work has explored attention mechanisms. Transformer (Vaswani et al., 2017) is able to handle arbitrary sequences of observations. ATTAIN (Zhang, 2019) incorporates an attention mechanism with LSTM to model the time irregularity between observations. SeFT (Horn et al., 2020) maps the irregular time series into a set of observations based on differentiable set functions and thus applies an attention mechanism for classification. mTAND (Shukla & Marlin, 2020) presented a multi-time attention network, which learns continuous-time embeddings coupled with a multi-time attention mechanism to deal with the continuous-time inputs. RAINDROP (Zhang et al., 2022) instead considers the irregularly sampled time series as separate sensor graphs and utilizes graph neural networks to learn the dependencies between different sensors (variables). Most of these existing methods are highly customized, requiring considerable prior knowledge and effort to design and modify model architectures. In this work, we propose a simple and general approach without assuming any prior knowledge or specific engineering.

Imaging Time Series. With the success of computer vision techniques, there are prior studies transforming time series into images for feature extraction. Wang & Oates (2015a) use Gramian Angular Fields (GAF) and Markov Transition Fields (MTF) to encode time series into texture images and utilize Tiled Convolutional Neural Network for classification. Similarly, Hatami et al. (2018) represent time series with recurrence plots and use CNN to classify the time series. Silva et al. (2013) generate time series images according to the compression distance. However, these imaging methods require a considerable amount of domain knowledge and fail to serve as a general solution to different kinds of time series, especially irregular ones. In contrast, we transform time series into a general and simple visual representation, *i.e.,* time series line graphs and perform image classification.

3 APPROACH

As illustrated in Fig. 1, **ViTTs** consists of two steps: (Step 1) Transform multivariate time series into a concatenated line graph image; (Step 2) Utilize the Vision Transformer as an image encoder to learn effective representation for the classification task. To begin with, we present some basic notations and problem formulation.

Notation. Let $\mathcal{D} = \{(\mathcal{S}_i, y_i)|i = 1, \cdots, N\}$ denote a time series dataset containing N samples. Every data sample is associated with a label $y_i \in \{1, \cdots, C\}$, where C is the number of classes. Each multivariate time series $\mathcal{S}_i$ consists of observations of D variables at most (some might have no observations). The observations for each variable d is given by a sequence of tuples with observed time and value $[(t_1^d, v_1^d), (t_2^d, v_2^d), \cdots, (t_{n_d}^d, v_{n_d}^d)]$, where n_d is the number of observations for variable d. When the set of observation times $[t_1^d, t_2^d, \cdots, t_{n_d}^d]$ is different across the variables and samples, $\mathcal{S}_i$ is an irregularly sampled time series, otherwise regular time series.

Problem Formulation. *We aim to learn a function $f : \mathcal{S}_i \rightarrow \boldsymbol{z}_i$ to map the multivariate time series $\mathcal{S}_i$ to its representation $\boldsymbol{z}_i$, which can be used to predict the corresponding label $\hat{y}_i$. In our framework, the function is decomposed into two parts: (1) a function $f_1 : \mathcal{S}_i \rightarrow \mathrm{x}_i$ that transforms the time series $\mathcal{S}_i$ into an image x_i; (2) an image encoder $f_2 : \mathrm{x}_i \rightarrow \mathbf{z}_i$ that learns the representation of input time series image.*

3.1 TIME SERIES LINE GRAPH

Time series line graph is a widely-used data visualization method to illustrate temporal data points at successive intervals. Each point on the line graph corresponds to an observation with an observed time and value. The horizontal axis is used to plot timestamps, and the vertical axis is used to plot values. Straight lines connect the points on the graph in the order of time, where the missing value interpolation is done automatically. We use markers to distinguish observations from the "interpolated" dots in the line. As the scale of different variables varies greatly, we plot the observations of each variable in an individual line graph, as shown in Fig. 1. The scales of each line graph $\mathrm{g}_{i,d}$ are kept the same across different time series s_i. Different color is used for each line graph to distinguish them. We experimentally found that the tick labels of line graphs are unnecessary, as the position of an observation in a line graph indicates the relative magnitude of observed time and value.

Given a set of time series line graphs $\mathcal{G}_i = \{\mathrm{g}_1, \mathrm{g}_2, \cdots, \mathrm{g}_D\}$ for time series $\mathcal{S}_i$, we place them in a super image using a pre-defined layout. Similar with (Fan et al., 2021), we experimentally found that

a compact layout (*i.e.*, square grid) leads to better performance. Specifically, given the D time series line graph for a time series, we place them in a grid whose size is $l \times l$ when $l \times (l-1) < D <= l \times l$, and $l \times (l+1)$ when $l \times l < D <= l \times (l+1)$. The order of line graphs does not affect the performance. The size of the created image, *i.e.,* image resolution, is determined by the grid layout and the size of each time series line graph, which can be seen as a sub-image.

3.2 Vision Transformers for Time Series Modeling

Given the image x_i transformed from time series $\mathcal{S}_i$, we leverage an image classifier to learn its effective representation and perform the classification task. Unlike natural images that a computer vision model usually sees, the time series patterns in a line graph image involve both local (*i.e.*, the temporal dynamics of a single variable in a line graph) and global (the correlation among variables across different line graphs) contexts, which poses a challenge to the model to capture them. To this end, we choose the recently developed transformer-based vision models, notably Vision Transformers (Dosovitskiy et al., 2020). Unlike the predominant CNNs, vision transformers are proven to have much less image-specific inductive bias and stronger abilities to capture local and global dependencies (Dosovitskiy et al., 2020; Liu et al., 2021).

Vision Transformers. In the vanilla Vision Transformer ViT (Dosovitskiy et al., 2020), an image is split into fix-sized patches. Each patch is then linearly embedded and added with position embeddings, which indicate the absolute position of this patch in the image. The resulting sequence of vectors is fed to a standard Transformer encoder to obtain the token/patch representation. An extra classification token to the sequence is used to perform classification or other tasks. However, as the computational complexity of its self-attention is quadratic to image size, it has trouble modeling high-resolution images. To solve the issue, Swin transformer (Liu et al., 2021) constructs hierarchical feature maps and achieves linear complexity to the image size. It computes self-attention locally with non-overlapping windows. The small window size in the shallow layers makes Swin transformer capture the local context. The neighboring patches are gradually merged in the deeper layers, and thus, the global information will be learned. We implement our approach using Swin Transformer in this work, as it is more efficient.

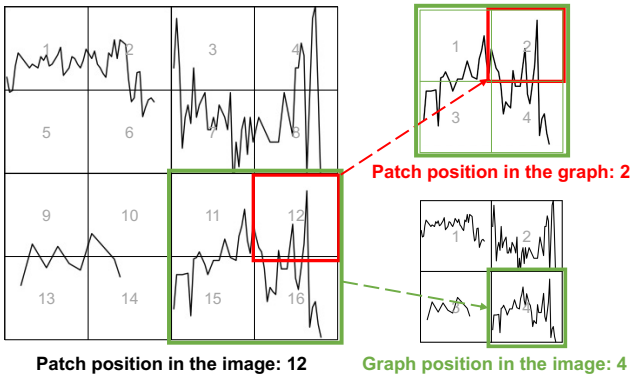

Figure 2: Positions of a patch (red box) based on different contexts in a time series image with 4 time series line graphs (green box) and 16 patches. Left is the position used in the vanilla Vision Transformer, *i.e.*, the absolute position in the image. Right shows the patch's position in a time series line graph (top) and the graph's position in the whole image (bottom). Best viewed in color.

Temporal Position Embedding. There are different regions in a line graph image encoding the temporal dynamics and relations of multivariate time series data. Specifically, the patch interactions within a single time series line graph reflect the temporal dynamics of the corresponding variable. In contrast, the interactions of patches across different line graphs reveal the variable correlations. To inject this knowledge into Vision Transformer and help it better capture these temporal patterns, we introduce two kinds of positions in the time series line graph images, as seen in Fig. 2:

- ***local temporal position***: the position of a patch in the local time series line graph, which reflects the relative value and time range of observations in a patch region.
- ***global temporal position***: the position of a time series line graph in the whole image, which indicates the time series line graph (variable) that a patch belongs to.

Being aware of these two kinds of position information, the model can learn to distinguish between the local patch interactions within a line graph and the global patch interactions across different line graphs. Consequently, the ***intra-variable temporal dynamics*** and ***inter-variable correlations*** can

be better captured. We thus combine these two kinds of temporal position information to constitute the ***temporal position embedding***. Specifically, we assign an embedding for each of these two kinds of positions, with the size half of patch embeddings. We then concatenate them to form the final temporal position embedding, which is added to the patch embedding as the input of the transformer encoder. The state of the classification token at the output serves as the image representation $\mathbf{z}_i$. As this method does not change the size of input embeddings and the model architecture, it can be applied in any vision transformer model, such as ViT (Dosovitskiy et al., 2020) and Swin Transformer (Liu et al., 2021).

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

Datasets	# Samples	# Variables	# Avg. obs.	# Classes	Static info	Imbalanced
P19	38,803	34	401	2	True	True
P12	11,988	36	233	2	True	True
PAM	5,333	17	4,048	8	False	False

Table 1: Statistics of the irregularly sampled time series datasets. "# Avg. obs." denotes the average number of observations for each sample. "Static info" indicates if the time series sample is associated with static attributes (*e.g.*, patients' demographics).

Datasets and Metrics. We experiment with three widely-used public healthcare and human activity datasets whose statistics are presented in Table 1. The P19 dataset records 34 irregularly sampled sensor variables for 38,803 patients. Each patient is associated with a binary label indicating the occurrence of sepsis. The P12 dataset contains the observations of 36 irregularly sampled sensor variables for 11,988 patients and a binary label representing whether the patient survives his hospitalization. The PAM (Reiss & Stricker, 2012) dataset contains 5,333 samples from 8 classes of human activities given observations of 17 sensor variables. We used the processed data provided by RAINDROP[2]. The evaluation metrics are also kept the same. Specifically, we report the Area Under a ROC Curve (AUROC) and Area Under Precision-Recall Curve (AUPRC) for the highly imbalanced datasets P12 and P19. As for the nearly balanced PAM dataset, we report Accuracy, Precision, Recall, and F1 score.

Implementation. The time-series-to-image transformation can be implemented using the Matplotlib package[3] within a few lines of Python code, as shown in Fig. 3. We use a representative vision transformer: Swin Transformer (Liu et al., 2021) to implement our approach using the pre-trained weights[4] provided in HuggingFace Wolf et al. (2020). The original ViT has a slightly worse performance in our setting.

The default patch size of Swin Transformer is 4×4. We set the size of each time series line graph as 64×64 for all datasets by default, meaning there will be 256 patches in each line graph. One can also change this size as long as it is divisible by the patch size of used models. The sizes of created images for each dataset are listed in Table 2. We also investigate the influence of different image resolutions on the model performance, which is presented in Sec. 4.3.

Datasets	# Variables	Grid Layout	Image Size
P19	34	6×6	384×384
P12	36	6×6	384×384
PAM	17	4×5	256×320

Table 2: The sizes of created images for each dataset, with the time series line graph's size being 64×64 by default.

Training. We apply the cutout (DeVries & Taylor, 2017) augmentation method to the input images during training. Specifically, 16 square regions with 16×16 size are randomly masked in each image. The models are trained using A6000 GPUs with 48G memory. Depending on the size of the input

[2]https://github.com/mims-harvard/Raindrop

[3]https://matplotlib.org/

[4]https://huggingface.co/microsoft/swin-base-patch4-window7-224-in22k

```python
def TS2Image(t, v, D, image_height, image_width, grid_height, grid_width):
    import matplotlib.pyplot as plt
    plt.figure(figsize=(image_height/100, image_width/100), dpi=100)
    for d in range(D): # enumerate the multiple variables
        plt.subplot(grid_height, grid_width, d+1) # grid layout
        plt.plot(t[d], v[d]) # plot the observations of variable d
```

Figure 3: The implementation of time-series-to-image transformation with Matplotlib in Python.

image, we use a batch size of 48 for P19 and P12 and 72 for PAM. We train all the models for 20 epochs on each dataset with 2 warming-up epochs. The learning rate is 2e-5.

Incorporating static features. The P12 and P19 datasets provide patients' demographics, such as their weight, height, and ICU type. This static information will not change over time and can be well described by the natural language. To incorporate them into our framework, we transform them into natural language sentences via a template and utilize a text encoder RoBERTa Liu et al. (2019) to encode them. The obtained text embedding is concatenated with the image embeddings obtained from the vision transformer to perform classification. The static feature is also applied to all the baselines we compare.

4.2 Main Results

Table 3: Comparison with the baseline methods on irregularly sampled time series classification task.

Methods	P19		P12		PAM			
	AUROC	AUPRC	AUROC	AUPRC	Accuracy	Precision	Recall	F1 score
Transformer	83.2 ± 1.3	47.6 ± 3.8	83.3 ± 1.7	47.9 ± 3.6	83.5 ± 1.5	84.8 ± 1.5	86.0 ± 1.2	85.0 ± 1.3
Trans-mean	84.1 ± 1.7	47.4 ± 1.4	82.6 ± 2.0	46.3 ± 4.0	83.7 ± 2.3	84.9 ± 2.6	86.4 ± 2.1	85.1 ± 2.4
GRU-D	83.9 ±1.7	46.9 ± 2.1	81.9 ± 2.1	46.1 ± 4.7	83.3 ± 1.6	84.6 ± 1.2	85.2 ± 1.6	84.8 ± 1.2
SeFT	78.7 ± 2.4	31.1 ± 2.8	73.9 ± 2.5	31.1 ± 4.1	67.1 ± 2.2	70.0 ± 2.4	68.2 ± 1.5	68.5 ± 1.8
mTAND	80.4 ± 1.3	32.4 ± 1.8	84.2 ± 0.8	48.2 ± 3.4	74.6 ± 4.3	74.3 ± 4.0	79.5 ± 2.8	76.8 ± 3.4
IP-Net	84.6 ± 1.3	38.1 ± 3.7	82.6 ± 1.4	47.6 ± 3.1	74.3 ± 3.8	75.6 ± 2.1	77.9 ± 2.2	76.6 ± 2.8
DGM^2-O	86.7 ± 3.4	44.7 ± 11.7	84.4 ± 1.6	47.3 ± 3.6	82.4 ± 2.3	85.2 ± 1.2	83.9 ± 2.3	84.3 ± 1.8
MTGNN	81.9 ± 6.2	39.9 ± 8.9	74.4 ± 6.7	35.5 ± 6.0	83.4 ± 1.9	85.2 ± 1.7	86.1 ± 1.9	85.9 ± 2.4
RAINDROP	87.0 ± 2.3	51.8 ± 5.5	82.8 ± 1.7	44.0 ± 3.0	88.5 ± 1.5	89.9 ± 1.5	89.9 ± 1.5	89.8 ± 1.0
ViTTs	**88.6±1.9**	**52.0±2.1**	**85.3±2.1**	**49.4±2.9**	**96.7±0.9**	**97.1±0.8**	**96.8±0.9**	**96.9±0.9**

Comparison with Baselines. We compare our approach with several state-of-the-art methods specialized for irregularly sampled time series: Transformer (Vaswani et al., 2017), Trans-mean, GRU-D (Che et al., 2018), SeFT (Horn et al., 2020), mTAND (Shukla & Marlin, 2020), IP-Net (Shukla & Marlin, 2018), and Raindrop (Zhang et al., 2022). Trans-mean denotes the Transformer with an imputation method that replaces the missing value with the average observed value. Raindrop is the best-performing method before this work. Besides, two methods initially designed for forecasting tasks are also compared, including DGM^2-O (Wu et al., 2021) and MTGNN (Wu et al., 2020). The implementations and hyperparameters setting of these baselines follow RAINDROP (Zhang et al., 2022). The batch size is 128, and all the models are trained for 20 epochs. As the P12 and P19 datasets are highly imbalanced, we upsample the minority class to ensure that each batch is balanced with half negative and half positive samples, which is kept the same across all the compared methods. The checkpoint that achieves the best AUROC performance on the validation set is used to make predictions on the test set. We rerun implementations of these baselines provided by (Zhang et al., 2022). Our obtained performances on the P19 and PAM are consistent with those reported in the original paper. However, the difference in the P12 dataset is not negligible. We thus report our experimental results on P12, and the reported performance on P19 and PAM are from (Zhang et al., 2022). The performances are averaged on 5 different data splits, which are fixed across all the compared baselines.

As seen from Table 3, our approach substantially outperforms the specialized state-of-the-art algorithms on all the datasets. On the P19 and P12 datasets, ViTTs improved 1.6 and 0.9 AUROC points compared with the best baseline. As for the PAM dataset, the improvement achieved by our approach is much more significant: 8.2 points in Accuracy, 7.2 points in Precision, 6.9 points in Recall, and 7.1

points in the F1 score. Considering the characteristics of these three datasets as presented in Table 1, the difference in improvement scales might be understandable: PAM has more observations and is not as sparse as P19 and P12. The interpolated line graph is thus closer to the true distribution.

Leaving-sensors-out Settings. We further evaluate models' performance in more challenging leave-sensors-out settings, where the observations of a subset of sensors (variables) are masked during testing. This setting simulates real-world scenarios when some sensors fail or become unreachable. Following (Zhang et al., 2022), we experiment with two setups to mask the sensors: (1) *leave-**fixed**-sensors-out* which drops a fixed set of sensors across all the samples and compared methods; (2) *leave-**random**-sensors-out* which drops the sensors randomly. Only the observations in the validation and test set are dropped. The training set is kept unchanged. For a fair comparison, we dropped the same set of sensors as in (Zhang et al., 2022) in the *leave-**fixed**-sensors-out* setting. We report the performance on the PAM dataset as it only contains the dynamic time series data without the intervention of static information.

Missing ratio	Methods	PAM (Leave-**fixed**-sensors-out)				PAM (Leave-**random**-sensors-out)			
		Accuracy	Precision	Recall	F1 score	Accuracy	Precision	Recall	F1 score
10%	Transformer	60.3 ± 2.4	57.8 ± 9.3	59.8 ± 5.4	57.2 ± 8.0	60.9 ± 12.8	58.4 ± 18.4	59.1 ± 16.2	56.9 ± 18.9
	Trans-mean	60.4 ± 11.2	61.8 ± 14.9	60.2 ± 13.8	58.0 ± 15.2	62.4 ± 3.5	59.6 ± 7.2	63.7 ± 8.1	62.7 ± 6.4
	GRU-D	65.4 ± 1.7	72.6 ± 2.6	64.3 ± 5.3	63.6 ± 0.4	68.4 ± 3.7	74.2 ± 3.0	70.8 ± 4.2	72.0 ± 3.7
	SeFT	58.9 ± 2.3	62.5 ± 1.8	59.6 ± 2.6	59.6 ± 2.6	40.0 ± 1.9	40.8 ± 3.2	41.0 ± 0.7	39.9 ± 1.5
	mTAND	58.8 ± 2.7	59.5 ± 5.3	64.4 ± 2.9	61.8 ± 4.1	53.4 ± 2.0	54.8 ± 2.7	57.0 ± 1.9	55.9 ± 2.2
	RAINDROP	77.2 ± 2.1	82.3 ± 1.1	78.4 ± 1.9	75.2 ± 3.1	76.7 ± 1.8	79.9 ± 1.7	77.9 ± 2.3	78.6 ± 1.8
	ViTTs	**92.7 ± 0.9**	**94.2 ± 0.9**	**93.2 ± 0.4**	**93.6 ± 0.6**	**88.4 ± 1.4**	**92.3 ± 0.5**	**88.6 ± 1.9**	**89.8 ± 1.5**
20%	Transformer	63.1 ± 7.6	71.1 ± 7.1	62.2 ± 8.2	63.2 ± 8.7	62.3 ± 11.5	65.9 ± 12.7	61.4 ± 13.9	61.8 ± 15.6
	Trans-mean	61.2 ± 3.0	74.2 ± 1.8	63.5 ± 4.4	64.1 ± 4.1	56.8 ± 4.1	59.4 ± 3.4	53.2 ± 3.9	55.3 ± 3.5
	GRU-D	64.6 ± 1.8	73.3 ± 3.6	63.5 ± 4.6	64.8 ± 3.6	64.8 ± 0.4	69.8 ± 0.8	65.8 ± 0.5	67.2 ± 0.0
	SeFT	35.7 ± 0.5	42.1 ± 4.8	38.1 ± 1.3	35.0 ± 2.2	34.2 ± 2.8	34.9 ± 5.2	34.6 ± 2.1	33.3 ± 2.7
	mTAND	33.2 ± 5.0	36.9 ± 3.7	37.7 ± 3.7	37.3 ± 3.4	45.6 ± 1.6	49.2 ± 2.1	49.0 ± 1.6	49.0 ± 1.0
	RAINDROP	66.5 ± 4.0	72.0 ± 3.9	67.9 ± 5.8	65.1 ± 7.0	71.3 ± 2.5	75.8 ± 2.2	72.5 ± 2.0	73.4 ± 2.1
	ViTTs	**88.4 ± 1.0**	**90.4 ± 1.4**	**89.3 ± 0.8**	**89.7 ± 1.0**	**85.1 ± 1.2**	**91.1 ± 1.0**	**85.6 ± 1.0**	**87.0 ± 1.0**
30%	Transformer	31.6 ± 10.0	26.4 ± 9.7	24.0 ± 10.0	19.0 ± 12.8	52.0 ± 11.9	55.2 ± 15.3	50.1 ± 13.3	48.4 ± 18.2
	Trans-mean	42.5 ± 8.6	45.3 ± 9.6	37.0 ± 7.9	33.9 ± 8.2	65.1 ± 1.9	63.8 ± 1.2	67.9 ± 1.8	64.9 ± 1.7
	GRU-D	45.1 ± 2.9	51.7 ± 6.2	42.1 ± 6.6	47.2 ± 3.9	58.0 ± 2.0	63.2 ± 1.7	58.2 ± 3.1	59.3 ± 3.5
	SeFT	32.7 ± 2.3	27.9 ± 2.4	34.5 ± 3.0	28.0 ± 1.4	31.7 ± 1.5	31.0 ± 2.7	32.0 ± 1.2	28.0 ± 1.6
	mTAND	27.5 ± 4.5	31.2 ± 7.3	30.6 ± 4.0	30.8 ± 5.6	34.7 ± 5.5	43.4 ± 4.0	36.3 ± 4.7	39.5 ± 4.4
	RAINDROP	52.4 ± 2.8	60.9 ± 3.8	51.3 ± 7.1	48.4 ± 1.8	60.3 ± 3.5	68.1 ± 3.1	60.3 ± 3.6	61.9 ± 3.9
	ViTTs	**84.1 ± 1.3**	**86.5 ± 0.4**	**83.1 ± 0.8**	**84.9 ± 1.0**	**80.6 ± 1.2**	**89.5 ± 1.3**	**80.9 ± 1.1**	**82.6 ± 1.1**
40%	Transformer	23.0 ± 3.5	7.4 ± 6.0	14.5 ± 2.6	6.9 ± 2.6	43.8 ± 14.0	44.6 ± 23.0	40.5 ± 15.9	40.2 ± 20.1
	Trans-mean	25.7 ± 2.5	9.1 ± 2.3	18.5 ± 1.4	9.9 ± 1.1	48.7 ± 2.7	55.8 ± 2.6	54.2 ± 3.0	55.1 ± 2.9
	GRU-D	46.4 ± 2.5	64.5 ± 6.8	42.6 ± 7.4	44.3 ± 7.9	47.7 ± 1.4	63.4 ± 1.6	44.5 ± 0.5	47.5 ± 0.0
	SeFT	26.3 ± 0.9	29.9 ± 4.5	27.3 ± 1.6	22.3 ± 1.9	26.8 ± 2.6	24.1 ± 3.4	28.0 ± 1.2	23.3 ± 3.0
	mTAND	19.4 ± 4.5	15.1 ± 4.4	20.2 ± 3.8	17.0 ± 3.4	23.7 ± 1.0	33.9 ± 6.5	26.4 ± 1.6	29.3 ± 1.9
	RAINDROP	52.5 ± 3.7	53.4 ± 5.6	48.6 ± 1.9	44.7 ± 3.4	57.0 ± 3.1	65.4 ± 2.7	56.7 ± 3.1	58.9 ± 2.5
	ViTTs	**76.5 ± 1.9**	**83.5 ± 0.9**	**76.7 ± 2.4**	**78.3 ± 2.1**	**73.7 ± 2.2**	**86.4 ± 1.1**	**74.0 ± 2.2**	**75.8 ± 1.8**
50%	Transformer	21.4 ± 1.8	2.7 ± 0.2	12.5 ± 0.4	4.4 ± 0.3	43.2 ± 2.5	52.0 ± 2.5	36.9 ± 3.1	41.9 ± 3.2
	Trans-mean	21.3 ± 1.6	2.8 ± 0.4	12.5 ± 0.7	4.6 ± 0.2	46.4 ± 1.4	59.1 ± 3.2	43.1 ± 2.2	46.5 ± 3.1
	GRU-D	37.3 ± 2.7	29.6 ± 5.9	32.8 ± 4.6	26.6 ± 5.9	49.7 ± 1.2	52.4 ± 0.3	42.5 ± 1.7	47.5 ± 1.2
	SeFT	24.7 ± 1.7	15.9 ± 2.7	25.3 ± 2.6	18.2 ± 2.4	26.4 ± 1.4	23.0 ± 2.9	27.5 ± 0.4	23.5 ± 1.8
	mTAND	16.9 ± 3.1	12.6 ± 5.5	17.0 ± 1.6	13.9 ± 4.0	20.9 ± 3.1	35.1 ± 6.1	23.0 ± 3.2	27.7 ± 3.9
	RAINDROP	46.6 ± 2.6	44.5 ± 2.6	42.4 ± 3.9	38.0 ± 4.0	47.2 ± 4.4	59.4 ± 3.9	44.8 ± 5.3	47.6 ± 5.2
	ViTTs	**70.0± 2.7**	**79.9 ± 2.2**	**70.5 ± 3.1**	**72.2 ± 3.0**	**70.9 ± 1.2**	**83.6 ± 2.4**	**71.5 ± 1.4**	**73.3 ± 2.1**

Table 4: Performance comparison in the leave-sensors-out settings on the PAM dataset. The "missing ratio" denotes the ratio of masked variables.

The results are listed in Table 4, from which we can see that our approach consistently achieves the best performance and outperforms the second best by a large margin across all the settings. With the missing ratio ranging from 10% to 50%, the performance improvement over the previously best model becomes increasingly significant. When half of the variables are dropped, our approach can still achieve acceptable performance, exceeding the best-performed baseline by up to 50.2% in Accuracy, 40.7% in Precision, 59.6% in Recall, and 54.0% in the F1 score. These experimental results suggest the robustness of our approach to the missing values or variables, which can be attributed to our image framework, enabling the use of image augmentation methods such as cutout (DeVries & Taylor, 2017).

Table 5: Ablation studies on the effects of different components and strategies on the model performance. The reported numbers are averaged on 5 data splits. We omit variances for simplicity. The numbers in brackets denotes the performance change *w.r.t.* the full model.

Methods	P19		P12		PAM			
	AUROC	AUPRC	AUROC	AUPRC	Accuracy	Precision	Recall	F1 score
ViTTs	**88.6**	**52.0**	**85.3**	**49.4**	**96.7**	**97.1**	**96.8**	**96.9**
w/o pos. emb.	87.5(-1.1)	50.7(-1.3)	84.4(-0.9)	48.0(-1.4)	94.7(-2.0)	95.5(-1.6)	95.3(-1.5)	95.2(-1.7)
w/o diff. colors	85.1(-3.5)	44.8(-7.2)	83.3(-2.0)	38.5(-10.9)	92.9(-3.8)	94.9(-2.2)	93.6(-3.2)	94.1(-2.8)
w/o cutout	87.1(-1.5)	48.6(-3.4)	84.2(-1.1)	47.3(-2.1)	95.4(-1.3)	96.1(-1.0)	95.9(-0.9)	96.1(-0.8)
w/o pre-training	73.6(-15.0)	28.7(-23.3)	69.1(-16.2)	24.3(-25.1)	84.5(-12.2)	86.6(-10.5)	87.1(-9.7)	86.6(-10.3)

4.3 Ablation Studies

To evaluate the effectiveness of our proposed design, we conduct ablation studies to answer the following questions.

Does temporal position embedding help? Temporal position embedding is introduced to help the vision transformer capture the *intra-variable* and *inter-variable* temporal correlations in the created time series line graph images. To investigate its effectiveness, we report performance of ViTTs without the *temporal position embedding* (*w/o pos. emb.*) in Table 5. We can observe the performance decrease without the introduced temporal position embedding on all the datasets, which suggests its importance in helping the model capture complex time series patterns. We witnessed a more significant performance reduction when trained on images that do not have different colors for each line graph (*w/o diff. colors*). This finding verifies the effectiveness of such a simple approach in distinguishing between different time series line graphs (variables), owing to the time series line graph image framework of our approach.

Does image augmentation improve the model performance? Our approach casts the time series classification task as an image classification task, enabling the use of image augmentation methods. However, in our case, some widely-used position augmentation methods, such as flipping, rotations, and color augmentation methods, are not good choices, as they will mix up the temporal patterns in the well-organized line graph images. Our approach thus utilizes the cutout (DeVries & Taylor, 2017) augmentation method, which will not change the position and color of the original image. From Table 5, we can see that it has a considerable impact on the model's performance, especially on the P19 and P12 datasets where we upsample the minority class to deal with class imbalance. The performance drop demonstrates that the cutout augmentation can mitigate overfitting and improve the generalization ability of our approach.

Does the model benefit from pre-training? To explore whether the model can benefit from pre-training, we disable pre-trained weights and train the Swin Transformer from scratch. The results are shown in the last row of Table 5. We found that the model performance decreases significantly without pre-training on each dataset, which suggests that the knowledge obtained from pre-training on natural images can be transferred to the created time series line graph images to some extent. Another interesting finding is that despite the performance reduction, the performance of our approach is still comparable with the baselines on the PAM dataset.

Grid Layout	Graph Size	Image Size	PAM			
			Accuracy	Precision	Recall	F1 score
2×9	64×64	128×576	95.9±1.4	96.5±1.0	95.9±1.2	96.0±0.5
3×6	64×64	192×384	96.1±0.8	96.7±0.5	95.9±0.9	96.2±0.7
4×5	64×64	256×256	**96.7 ± 0.9**	**97.1 ± 0.8**	**96.8 ± 0.9**	**96.9 ± 0.9**
4×5	48×48	192×240	96.1 ± 1.3	96.7 ± 0.5	96.0 ± 1.1	96.3 ± 0.6
4×5	64×64	256×256	96.7 ± 0.9	97.1 ± 0.8	96.8 ± 0.9	96.9 ± 0.9
4×5	80×80	320×400	**97.1 ± 0.6**	**97.8 ± 0.6**	**97.4 ± 0.7**	**97.1 ± 0.9**

Table 6: Ablation studies on the effects of different grid layouts and image sizes on the model performance on the PAM dataset. "Graph size" represents the size of each time series line graph which is a sub-image. "Image size" denotes the resolution of the whole image.

How do the grid layout and image size affect the performance? The grid layout and image size determine the granularity of the time series line graphs in the image. We trained ViTTs on images with different grid layouts and image sizes from the PAM dataset to investigate their effects on the model performance. The results are listed in Table 6. As seen from the first three rows, the square grid layout leads to the best performance despite limited improvement. This might be because the square images are more similar to the natural images the model is pre-trained on in terms of the layout. Without surprise, the results in the last three rows demonstrate that a higher image resolution can lead to better performance.

4.4 Regular Time Series Classification

As our approach can be directly applied to regular time series without any change, we are thus interested in how it performs on the regular time series classification task. We tested four representative regular multivariate time series datasets from the UEA Time Series Classification Archive (Bagnall et al., 2018): EthanolConcentration (EC), SelfRegulationSCP1 (SCP), SpokenArabicDigits (SAD), and Heartbeat (HB).

Datasets	# Variables	# Classes	Length
EC	3	4	1,751
SCP1	6	2	896
SAD	13	10	93
HB	61	2	405

Table 7: The statistics of regular multivariate time series datasets.

As shown in Table 7, these four datasets have diverse characteristics in terms of numbers of classes, variables, and time series length. We follow (Zerveas et al., 2021) to use these baselines for comparison: DTW_D which stands for dimension-Dependent DTW combined with dilation-CNN Franceschi et al., 2019, LSTM, XGBoost, ROCKET (Dempster et al., 2020), and a transformer-based TST (Zerveas et al., 2021). Most of algorithms on regular and irregular time series are studied separately. They often perform worse on the type of time series they are not designed for. We thus use the methods designed for regular time series as baselines.

The results are shown in Table 8, from which we can see that our approach performs consistently well on these datasets with different characteristics. The average accuracy even exceeds the best-performed baseline method, TST. These results indicate that vision transformers are a promising approach for both regular and irregular time series analysis. Given its simplicity, this approach has the potential to be extended as a general-purpose method for various time series tasks.

Methods	EC	SCP	SAD	HB	Avg. Acc.
DTW_D	0.323	0.775	0.963	0.717	0.695
LSTM	0.323	0.689	0.319	0.722	0.512
XGBoost	0.437	0.846	0.696	0.732	0.678
Rocket	<u>0.452</u>	<u>0.908</u>	0.712	0.756	0.707
TST	0.337	**0.925**	**0.993**	**0.776**	<u>0.758</u>
ViTTs	**0.456**	0.894	<u>0.985</u>	<u>0.766</u>	**0.775**

Table 8: Performance comparison on regular time series datasets. "Avg. Acc. denotes the average accuracy on all the datasets. Bold indicates the best performer while underline represents the second best.

5 Conclusion

In this paper, we introduce **ViTTs**, a simple and effective approach for irregularly sampled multivariate time series modeling. By transforming the multivariate time series into images, we can adapt the powerful vision transformers to time series classification tasks. This approach largely simplifies the irregular time series analysis pipeline, removing most of the specialization in algorithm design. Despite its simplicity, our approach demonstrates strong performance against the highly specialized state-of-the-art methods on several popular public datasets and shows robustness to missing observations. We also evaluate our approach on regular time series and witness promising results. Our approach can be potentially extended as a general-purpose framework for various time series tasks, which is our future direction. We believe this paper could provide a new perspective for time series analysis and encourage the reuse of fast-evolving computer vision techniques in the time series domain.

REFERENCES

Anthony Bagnall, Hoang Anh Dau, Jason Lines, Michael Flynn, James Large, Aaron Bostrom, Paul Southam, and Eamonn Keogh. The uea multivariate time series classification archive, 2018. *arXiv preprint arXiv:1811.00075*, 2018.

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

Hila Chefer, Shir Gur, and Lior Wolf. Transformer interpretability beyond attention visualization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 782–791, 2021.

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

Xinlei Chen, Saining Xie, and Kaiming He. An empirical study of training self-supervised vision transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9640–9649, 2021.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

Angus Dempster, François Petitjean, and Geoffrey I Webb. Rocket: exceptionally fast and accurate time series classification using random convolutional kernels. *Data Mining and Knowledge Discovery*, 34(5):1454–1495, 2020.

Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2020.

Quanfu Fan, Chun-Fu Chen, and Rameswar Panda. Can an image classifier suffice for action recognition? In *International Conference on Learning Representations*, 2021.

Jean-Yves Franceschi, Aymeric Dieuleveut, and Martin Jaggi. Unsupervised scalable representation learning for multivariate time series. *Advances in neural information processing systems*, 32, 2019.

Ary L Goldberger, Luis AN Amaral, Leon Glass, Jeffrey M Hausdorff, Plamen Ch Ivanov, Roger G Mark, Joseph E Mietus, George B Moody, Chung-Kang Peng, and H Eugene Stanley. Physiobank, physiotoolkit, and physionet: components of a new research resource for complex physiologic signals. *circulation*, 101(23):e215–e220, 2000.

Alex Graves. Long short-term memory. *Supervised sequence labelling with recurrent neural networks*, pp. 37–45, 2012.

Nima Hatami, Yann Gavet, and Johan Debayle. Classification of time-series images using deep convolutional neural networks. In *Tenth international conference on machine vision (ICMV 2017)*, volume 10696, pp. 242–249. SPIE, 2018.

Max Horn, Michael Moor, Christian Bock, Bastian Rieck, and Karsten Borgwardt. Set functions for time series. In *International Conference on Machine Learning*, pp. 4353–4363. PMLR, 2020.

Colin Lea, Michael D Flynn, Rene Vidal, Austin Reiter, and Gregory D Hager. Temporal convolutional networks for action segmentation and detection. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 156–165, 2017.

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

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10012–10022, 2021.

Ze Liu, Han Hu, Yutong Lin, Zhuliang Yao, Zhenda Xie, Yixuan Wei, Jia Ning, Yue Cao, Zheng Zhang, Li Dong, et al. Swin transformer v2: Scaling up capacity and resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12009–12019, 2022.

Benjamin M Marlin, David C Kale, Robinder G Khemani, and Randall C Wetzel. Unsupervised pattern discovery in electronic health care data using probabilistic clustering models. In *Proceedings of the 2nd ACM SIGHIT international health informatics symposium*, pp. 389–398, 2012.

Trang Pham, Truyen Tran, Dinh Phung, and Svetha Venkatesh. Predicting healthcare trajectories from medical records: A deep learning approach. *Journal of biomedical informatics*, 69:218–229, 2017.

Attila Reiss and Didier Stricker. Introducing a new benchmarked dataset for activity monitoring. In *2012 16th international symposium on wearable computers*, pp. 108–109. IEEE, 2012.

Yulia Rubanova, Ricky TQ Chen, and David K Duvenaud. Latent ordinary differential equations for irregularly-sampled time series. *Advances in neural information processing systems*, 32, 2019.

Connor Shorten and Taghi M Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of big data*, 6(1):1–48, 2019.

Satya Narayan Shukla and Benjamin Marlin. Interpolation-prediction networks for irregularly sampled time series. In *International Conference on Learning Representations*, 2018.

Satya Narayan Shukla and Benjamin Marlin. Multi-time attention networks for irregularly sampled time series. In *International Conference on Learning Representations*, 2020.

Diego F Silva, Vinícius MA De Souza, and Gustavo EAPA Batista. Time series classification using compression distance of recurrence plots. In *2013 IEEE 13th International Conference on Data Mining*, pp. 687–696. IEEE, 2013.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. Training data-efficient image transformers & distillation through attention. In *International Conference on Machine Learning*, pp. 10347–10357. PMLR, 2021.

RK Tripathy and U Rajendra Acharya. Use of features from rr-time series and eeg signals for automated classification of sleep stages in deep neural network framework. *Biocybernetics and Biomedical Engineering*, 38(4):890–902, 2018.

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

Zhiguang Wang and Tim Oates. Imaging time-series to improve classification and imputation. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015a.

Zhiguang Wang and Tim Oates. Spatially encoding temporal correlations to classify temporal data using convolutional neural networks. *arXiv preprint arXiv:1509.07481*, 2015b.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, pp. 38–45, 2020.

Yinjun Wu, Jingchao Ni, Wei Cheng, Bo Zong, Dongjin Song, Zhengzhang Chen, Yanchi Liu, Xuchao Zhang, Haifeng Chen, and Susan B Davidson. Dynamic gaussian mixture based deep generative model for robust forecasting on sparse multivariate time series. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pp. 651–659, 2021.

Zonghan Wu, Shirui Pan, Guodong Long, Jing Jiang, Xiaojun Chang, and Chengqi Zhang. Connecting the dots: Multivariate time series forecasting with graph neural networks. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 753–763, 2020.

Jinsung Yoon, William R Zame, and Mihaela van der Schaar. Multi-directional recurrent neural networks: A novel method for estimating missing data. In *Time series workshop in international conference on machine learning*, 2017.

George Zerveas, Srideepika Jayaraman, Dhaval Patel, Anuradha Bhamidipaty, and Carsten Eickhoff. 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.

Xiang Zhang, Marko Zeman, Theodoros Tsiligkaridis, and Marinka Zitnik. Graph-guided network for irregularly sampled multivariate time series. In *International Conference on Learning Representations*, 2022.

Yuan Zhang. Attain: Attention-based time-aware lstm networks for disease progression modeling. In *In Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI-2019), pp. 4369-4375, Macao, China.*, 2019.