Multi-Time Attention Networks for Irregularly Sampled Time Series

Satya Narayan Shukla & Benjamin M. Marlin

Elliot Hill

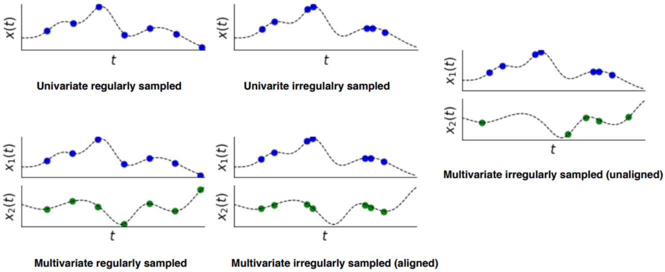
Duke University

March, 2023

Overview

- Background and motivation
- Related work
- Multi-time attention networks
- Results and discussion

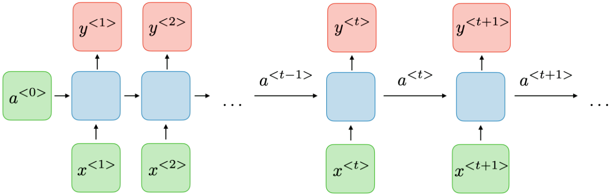
Time-series data poses significant challenges



- Irregular sampling: samples are not spaced at even time intervals
- Multivariate: outcome is condition on multiple features
- Misaligned sampling: features are measured at different time points
- Missing values: features may be only partially observed
- Sparsity: intervals between time-points may be long

Irregularly sampled data does not work out of the box for most deep learning architectures

RNNs, LSTMs, and transformers can use time-series data as their input, but they assume that the input is discrete and has a fixed length

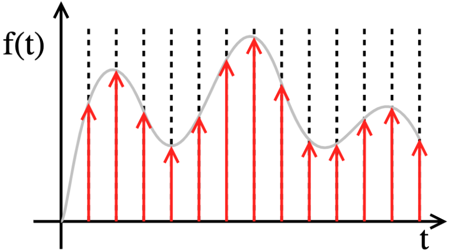


A lot of approaches that have been developed to address irregularly sampled time-series for deep learning feel ad hoc

(Amidi, RNN cheatsheet)

Binning/discretization is easy, but brittle

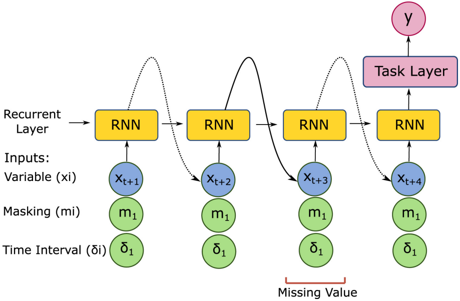
A common solution is to bin continuous values into fixed discrete values, but this loses valuable information and proliferates missing values



(Wikipedia, Discrete time)

Recurrent architecture + masking + time-interval

Another idea is to use architectures that handle sequences naturally, like RNN, GRU, or LSTM and add masking (missing/non-missing) and time-intervals between observations to their input



(Zhang & Thorburn, 2022)

ODE + RNN

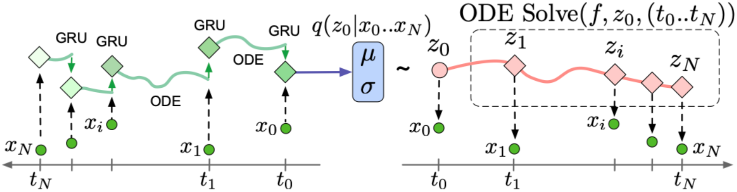


Figure 2: The Latent ODE model with an ODE-RNN encoder. To make predictions in this model, the ODE-RNN encoder is run backwards in time to produce an approximate posterior over the initial state: $q(z_0|\{x_i, t_i\}_{i=0}^{N})$. Given a sample of z_0, we can find the latent state at any point of interest by solving an ODE initial-value problem. Figure adapted from Chen et al. [2018].

(Chen & Duvenaud 2019)

Attention base solutions have been proposed previously

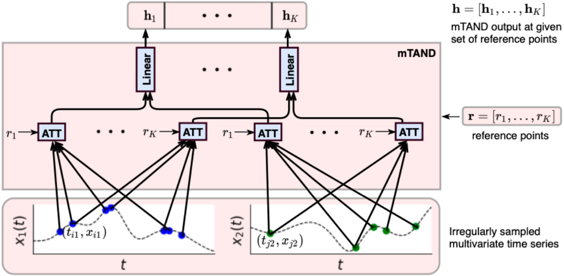
Often, the positional encoding in transformers (Vaswani et al. 2017) will be replace with a encoding of time and model sequences using attention (e.g. Zhang et al., 2019)

For example, Xu et al. (2019) learn a time representation and concatenates it with the input event embedding to model time-event interactions

In Shukla & Marlin (2021), instead of concatenating the time representation with the input embedding, the model learns to attend to observations at different time points by computing a similarity weighting

mTAND: Multi-time attention networks

mTAND re-represents an irregularly sampled time-series as a fixed set of reference points that are used as queries in the attention mechanism and the observed values are used as the keys



A learned continuous-time embedding mechanism coupled with a time attention mechanism replaces the use of a fixed similarity kernel

Notation

$\mathcal{D} = \{(s_n, y_n) | n = 1, ..., N\}$ represents a dataset containing N cases Where y_n is a target value and s_n is a D-dimensional, sparse and irregularly sampled multivariate time series

Time-series d for case n is $s_{dn} = (t_{dn}, x_{dn})$ where $t_{dn} = [t_{1dn}, ..., t_{L_{dn}dn}]$ is a list of time points, $x_{dn} = [x_{1dn}, ..., x_{L_{dn}dn}]$ is the corresponding observations, and L_{dn} is the total number of observations for a given time-series

The goal of the time attention module is to embed continuous time points into a fixed-length vector space

The time embeddings replaces the transformer's positional encoding

$$\phi_h(t)[i] = \begin{cases} \omega_{0h} \cdot t + \alpha_{0h}, & \text{if } i = 0 \\ \sin(\omega_{ih} \cdot t + \alpha_{ih}), & \text{if } 0 < i < d_r \end{cases}$$

where ω_{ih} and α_{ih} are learnable parameters. This time embedding component takes a continuous time point and embeds it into H different d_r-dimensional spaces. r is a reference point (described later),

The first term captures linear trends and the second term captures nonlinear seasonality

mTAN module

The multi-time attention module, mTAN(t, s), takes as input a query time point, t, and a time series, s, and outputs a J-dimensional embedding at time t

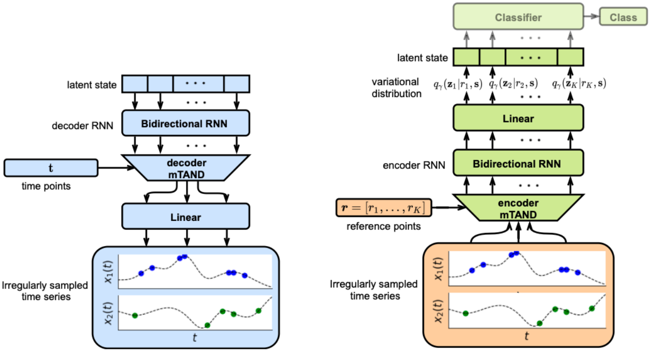
$$mTAN(t,s)[j] = \sum_{h=1}^{H}\sum_{d=1}^{D} \hat{x}_{hd}(t,s) \cdot U_{hdj}$$

$$\hat{x}_{hd}(t,s) = \sum_{i=1}^{L_d} \kappa_h(t, t_{id}) x_{id}$$

$$\kappa_h(t, t_{id}) = \frac{exp(\phi_h(t)\boldsymbol{w}\boldsymbol{v}^T\phi_h(t_{id})^T/\sqrt{d_k}}{\sum_{i'=1}^{L_d} exp(\phi_h(t)\boldsymbol{w}\boldsymbol{v}^T\phi_h(t_{i'd})^T/\sqrt{d_k}}$$

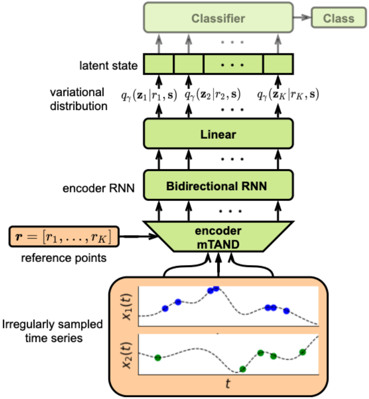
The parameters $\boldsymbol{w}$ and $\boldsymbol{v}$ are each $d_r \times d_k$ matrices where $d_k \leq d_r$, $\kappa_h(t, t_{id})$ are the interpolation weights for the kernel smoother $\hat{x}_{hd}(t,s)$, and parameters U_{hdj} are learnable weights

Encoder-decoder framework



(a) Generative Model (Decoder) (b) Inference Network (Encoder)

The encoder takes time-series as input and outputs a fixed-length latent representation for each reference point

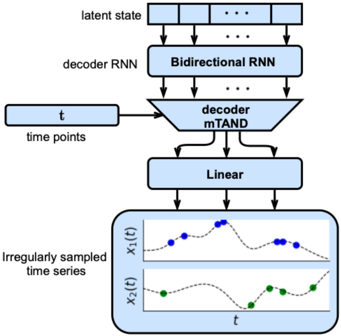


(b) Inference Network (Encoder)

$$\boldsymbol{h}^{enc}_{TAN} = mTAND^{enc}(\boldsymbol{r}, \boldsymbol{s})$$
$$\boldsymbol{h}^{enc}_{RNN} = RNN^{enc}(\boldsymbol{h}^{enc}_{RNN})$$
$$\boldsymbol{z}_k \sim q_\gamma(\boldsymbol{z}_k | \boldsymbol{\mu}_k, \boldsymbol{\sigma}^2_k)$$
$$\boldsymbol{\mu}_k = f^{enc}_\mu(\boldsymbol{h}^{enc}_{k,RNN})$$
$$\boldsymbol{\sigma}^2_k = exp(f^{enc}_\sigma(\boldsymbol{h}^{enc}_{k,RNN}))$$

where $z_k = [z_1, ..., z_K]$ is a set of latent states at K reference points

The decoder uses the latent representations to produce reconstructions conditioned on the observed time points



(a) Generative Model (Decoder)

$$
\begin{aligned}
\boldsymbol{z}_k &\sim p(\boldsymbol{z}_k) \\
\boldsymbol{h}^{dec}_{RNN} &= RNN^{dec}(\boldsymbol{z}) \\
\boldsymbol{h}^{dec}_{TAN} &= mTAND^{dec}(\boldsymbol{t}, \boldsymbol{h}^{dec}_{RNN}) \\
x_{id} &\sim \mathcal{N}(x_{id}; f^{dec}(\boldsymbol{h}^{dec}_{i,TAN})[d], \sigma^2 \boldsymbol{I})
\end{aligned}
$$

This generates a time-series $\hat{s} = (\boldsymbol{t}, \boldsymbol{x})$ with all data dimensions observed

mTAN uses a modified VAE loss

Unsupervised loss

$$\mathcal{L}_{\text{NVAE}}(\theta,\gamma) = \sum_{n=1}^{N} \frac{1}{\sum_d L_{dn}} \Big(\mathbb{E}_{q_\gamma(\mathbf{z}|\mathbf{r},\mathbf{s}_n)}[\log p_\theta(\mathbf{x}_n|\mathbf{z},\mathbf{t}_n)] - D_{\text{KL}}(q_\gamma(\mathbf{z}|\mathbf{r},\mathbf{s}_n)||p(\mathbf{z}))\Big)$$

$$D_{\text{KL}}(q_\gamma(\mathbf{z}|\mathbf{r},\mathbf{s}_n)||p(\mathbf{z})) = \sum_{i=1}^{K} D_{\text{KL}}(q_\gamma(\mathbf{z}_i|\mathbf{r},\mathbf{s}_n)||p(\mathbf{z}_i))$$

$$\log p_\theta(\mathbf{x}_n|\mathbf{z},\mathbf{t}_n) = \sum_{d=1}^{D}\sum_{j=1}^{L_{dn}} \log p_\theta(x_{jdn}|\mathbf{z},t_{jdn})$$

Supervised loss

$$\mathcal{L}_{\text{supervised}}(\theta,\gamma,\delta) = \mathcal{L}_{\text{NVAE}}(\theta,\gamma) + \lambda\mathbb{E}_{q_\gamma(\mathbf{z}|\mathbf{r},\mathbf{s}_n)} \log p_\delta(y_n|\mathbf{z})$$

mTAND beats SOTA methods for some interpolation tasks

Table 1: Interpolation performance versus percent observed time points on PhysioNet

Model	**Mean Squared Error** ($\times 10^{-3}$)				
RNN-VAE	13.418 ± 0.008	12.594 ± 0.004	11.887 ± 0.005	11.133 ± 0.007	11.470 ± 0.006
L-ODE-RNN	8.132 ± 0.020	8.140 ± 0.018	8.171 ± 0.030	8.143 ± 0.025	8.402 ± 0.022
L-ODE-ODE	6.721 ± 0.109	6.816 ± 0.045	6.798 ± 0.143	6.850 ± 0.066	7.142 ± 0.066
mTAND-Full	$\mathbf{4.139 \pm 0.029}$	$\mathbf{4.018 \pm 0.048}$	$\mathbf{4.157 \pm 0.053}$	$\mathbf{4.410 \pm 0.149}$	$\mathbf{4.798 \pm 0.036}$
Observed %	50%	60%	70%	80%	90%

mTAND matches or exceeds the performance of other SOTA methods and is much faster

Table 2: Classification Performance on PhysioNet, MIMIC-III and Human Activity dataset

Model	AUC Score		Accuracy	time
	PhysioNet	**MIMIC-III**	**Human Activity**	**per epoch**
RNN-Impute	0.764 ± 0.016	0.8249 ± 0.0010	0.859 ± 0.004	0.5
RNN-Δ_t	0.787 ± 0.014	0.8364 ± 0.0011	0.857 ± 0.002	0.5
RNN-Decay	0.807 ± 0.003	0.8392 ± 0.0012	0.860 ± 0.005	0.7
RNN GRU-D	0.818 ± 0.008	0.8270 ± 0.0010	0.862 ± 0.005	0.7
Phased-LSTM	0.836 ± 0.003	0.8429 ± 0.0035	0.855 ± 0.005	0.3
IP-Nets	0.819 ± 0.006	0.8390 ± 0.0011	0.869 ± 0.007	1.3
SeFT	0.795 ± 0.015	0.8485 ± 0.0022	0.815 ± 0.002	0.5
RNN-VAE	0.515 ± 0.040	0.5175 ± 0.0312	0.343 ± 0.040	2.0
ODE-RNN	0.833 ± 0.009	$\mathbf{0.8561 \pm 0.0051}$	0.885 ± 0.008	16.5
L-ODE-RNN	0.781 ± 0.018	0.7734 ± 0.0030	0.838 ± 0.004	6.7
L-ODE-ODE	0.829 ± 0.004	$\mathbf{0.8559 \pm 0.0041}$	0.870 ± 0.028	22.0
mTAND-Enc	0.854 ± 0.001	0.8419 ± 0.0017	$\mathbf{0.907 \pm 0.002}$	**0.1**
mTAND-Full	$\mathbf{0.858 \pm 0.004}$	$\mathbf{0.8544 \pm 0.0024}$	$\mathbf{0.910 \pm 0.002}$	0.2

mTAND has a lot of benefits

- Can handle sparse, irregularly sampled time-series with partially observed features
- Leverages a time attention mechanism to learn temporal similarity from data instead of using fixed kernels
- Meets or exceeds the performance of other SOTA methods on some time-series tasks
- Faster than other SOTA methods
- Could swap the VAE approach used here for any generative model

References

Afshine Amidi. Recurrent neural networks cheatsheet. URL `https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks`.

Yulia Rubanova, Ricky T. Q. Chen, and David Duvenaud. Latent odes for irregularly-sampled time series. *CoRR*, abs/1907.03907, 2019. URL `http://arxiv.org/abs/1907.03907`.

Satya Narayan Shukla and Benjamin M. Marlin. A survey on principles, models and methods for learning from irregularly sampled time series: From discretization to attention and invariance. *CoRR*, abs/2012.00168, 2020. URL `https://arxiv.org/abs/2012.00168`.

Satya Narayan Shukla and Benjamin M. Marlin. Multi-time attention networks for irregularly sampled time series. *CoRR*, abs/2101.10318, 2021. URL `https://arxiv.org/abs/2101.10318`.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017. URL `http://arxiv.org/abs/1706.03762`.

Da Xu, Chuanwei Ruan, Sushant Kumar, Evren Körpeoglu, and Kannan Achan. Self-attention with functional time representation learning. *CoRR*, abs/1911.12864, 2019. URL `http://arxiv.org/abs/1911.12864`.

Yifan Zhang and Peter J. Thorburn. Handling missing data in near real-time environmental monitoring: A system and a review of selected methods. *Future Generation Computer Systems*, 128:63–72, 2022. ISSN 0167-739X. doi: https://doi.org/10.1016/j.future.2021.09.033. URL `https://www.sciencedirect.com/science/article/pii/S0167739X21003794`.

Yuan Zhang, Xi Yang, Julie Ivy, and Min Chi. Attain: Attention-based time-aware lstm networks for disease progression modeling. In *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pages 4369–4375. International Joint Conferences on Artificial Intelligence Organization, 7 2019. doi: 10.24963/ijcai.2019/607. URL `https://doi.org/10.24963/ijcai.2019/607`.