Continuous-Time Attention for Sequential Learning

Jen-Tzung Chien, Yi-Hsiang Chen
Department of Electrical and Computer Engineering
National Chaio Tung University, Hsinchu, Taiwan
{jtchien, ethernet420.eed08g}@nctu.edu.tw

Abstract

Attention mechanism is crucial for sequential learning where a wide range of applications have been successfully developed. This mechanism is basically trained to spotlight on the region of interest in hidden states of sequence data. Most of the attention methods compute the attention score through relating between a query and a sequence where the discrete-time state trajectory is represented. Such a discrete-time attention could not directly attend the continuous-time trajectory which is represented via neural differential equation (NDE) combined with recurrent neural network. This paper presents a new continuous-time attention method for sequential learning which is tightly integrated with NDE to construct an attentive continuous-time state machine. The continuous-time attention is performed at all times over the hidden states for different kinds of irregular time signals. The missing information in sequence data due to sampling loss, especially in presence of long sequence, can be seamlessly compensated and attended in learning representation. The experiments on irregular sequence samples from human activities, dialogue sentences and medical features show the merits of the proposed continuous-time attention for activity recognition, sentiment classification and mortality prediction, respectively.

Introduction

Learning representation of sequence data in spatial or temporal domain is crucial (Chien 2019; Tseng et al. 2017). The popular examples of sequence data include natural sentences, video streams and medical signals. Most of them are seen as time-series data although the sequences of spatial samples in images are also recognized as the sequence data. An essential solution to sequential learning is based on the recurrent neural network (RNN) where the hidden states of previous samples are continuously updated by a recurrent machine, and seamlessly applied for prediction of next sample. One critical issue in sequential learning is to characterize the dynamics of sampling resolution in sequence data. Generally, RNNs are feasible to learning representation for regularly-sampled time-series data, while they are an awkward fit to irregularly-sampled time series. However, irregular time-series data are very common in real world. For example, in medical areas, we usually predict the health of a patient by using several biomedical signals acquired by different sensors or diagnosis facilities where the sampling resolutions in different signals are varied (Lipton, Kale, and Wetzel 2016; Che et al. 2018; El Naqa et al. 2018; Chuang et al. 2020). Some other practical applications in presence of irregular-sampled data (Elman 1990) include financial marketing (Bauer, Schölkopf, and Peters 2016), weather forecasting (Shi et al. 2015) and traffic engineering (Wang et al. 2017), to name a few. A key issue in these applications is the missing data problem which is possibly caused due to the facility cost or affected by machine anomaly. A naive way to deal with this issue is to adopt the time difference of sequence samples as a new feature input for RNN training. Alternatively, an attractive approach is to construct a continuous-time machine based on neural differential equation (NDE) (Chen et al. 2018) where the continuous-time hidden state space is constructed for learning representation similarly over the sequence data with unlimited length. The discrete-time state transitions in RNN are generalized to the continuous-time state dynamics by combining NDE with RNN (Rubanova, Chen, and Duvenaud 2019). NDE is then built as a strong model with rich sequence information for prediction. NDE is seen as an RNN model for extremely long sequence. NDE can tackle the weakness of RNN which is deteriorated when the length of input sequence increases (Bahdanau, Cho, and Bengio 2015). Nevertheless, the performance of sequential learning is bounded because the relevance or the importance of individual samples to target task is neglected. This paper presents a continuous-time attention scheme to strengthen the learning of irregular sampled data.

In general, attention mechanism can be powerful to capture the relevance information between a query and a document or sequence which has been successfully developed for a wide range of applications based on standard RNN (Chien and Lin 2018; Chien and Wang 2019; Chien and Lin 2020). The document is formed as a matrix where each row is a feature vector extracted from sequence data at different time points. Then, the attention score is computed by using a query vector and a document matrix. This paper presents a novel continuous-time attention method for sequential learning where the attention is performed in continuous-time state space based on NDE. Accordingly, the document is represented by a continuous function rather than a matrix. This function represents the features at any desired time moments

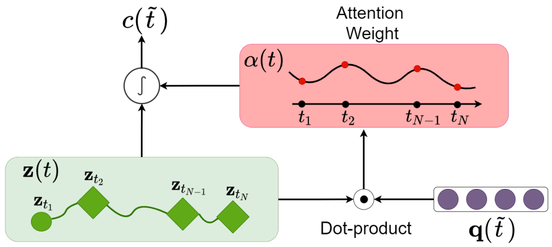


Figure 1: Illustration for continuous-time attention. The difficulty is to calculate the dot-product of two continuous functions and integrate them to obtain context vector $\mathbf{c}(\tilde{t})$. $\tilde{t}$ is time index for query vector $\mathbf{q}(\tilde{t})$. $\{t_n\}_{n=1}^{N}$ denote the time points of sequence data $\{\mathbf{x}_{t_n}\}$.

via continuous-time hidden states. The concept of calculating the continuous-time attention is depicted in Figure 1. The difficulty in this calculation is that there is no previous attention method suitable to find attention weights by using a continuous function for document. This paper tackles such a dilemma by representing both attention score and context vector as continuous-time functions. Attention score function is exploited to carry out the attention $\boldsymbol{\alpha}(t)$ for the whole state trajectory while context vector function reflects the weighted sum of whole continuous-time hidden states $\mathbf{z}(t)$. The merits of the proposed continuous-time attention are illustrated by the experiments on action recognition and medical data analysis as well as emotion recognition. We report the results of the so-called attentive neural differential equation (also denoted by Att-NDE) under different experimental settings by comparing with RNN and NDE.

Related Works

The sequential learning methods with continuous-time state machine and attention mechanism are first introduced.

Continuous-Time State Machine

Neural differential equation (Chen et al. 2018; Zhang et al. 2021) was proposed to build a continuous-time state machine for learning representation of sequence data $\{\mathbf{x}_{t_n}\}$ where the time points $\{t_n\}_{n=1}^{N}$ are *irregularly* sampled. NDE was implemented to learn the dynamics of transformation so as to characterize the state transition $\mathbf{z}(t)$ at continuous-time t between input samples and output targets based on an ordinary differential equation (ODE). This problem was solved by handling an ODE with initial value

$$\frac{d\mathbf{z}(t)}{dt} = f(\mathbf{z}(t), t; \theta), \quad \mathbf{z}(t_i) = \mathbf{z}(t_0) + \int_{t_0}^{t_i} \frac{d\mathbf{z}(t)}{dt} dt \quad (1)$$

where f is the function represented by neural network with parameter θ, $\mathbf{z}(t_0)$ denotes the initial state, and $\mathbf{z}(t_i)$ denotes the hidden state at a desirable time point t_i. ODE solver is introduced to deal with the integration as illustrated in Figure 2. The ODE solver, which is a tool to solve ODE initial value problem, is continuously applied to carry out the continuous-time state $\mathbf{z}(t)$ at continuous time t in a range between t_1 and t_N. A neural network is introduced to represent the derivative f for an unknown differential equation. Figure 3 illustrates the dynamic from $\mathbf{z}(t)$ to $\mathbf{z}(t + \triangle_t)$ in a continuous-time state machine by using neural ODE or simply denoted by NDE. ODE solver is adopted to resolve the latent dynamic system in time-series data (Rubanova, Chen, and Duvenaud 2019). This state machine calculates the continuous-time hidden states $\mathbf{z}(t)$ between two discrete observations $\mathbf{x}_{t_{n-1}}$ and $\mathbf{x}_{t_n}$ by applying an ODE solver

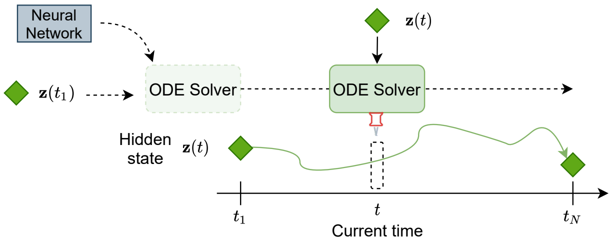


Figure 2: ODE solver for continuous-time state machine.

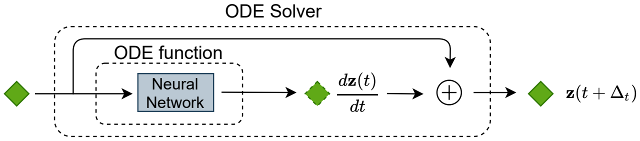


Figure 3: Dynamic of hidden state $\mathbf{z}(t)$ in neural ODE.

$$\hat{\mathbf{z}}(t_n) = \text{ODESolver}\left(f, \mathbf{z}(t_{n-1}), t_{n-1}, t_n, \theta\right) \quad (2)$$

which is a function of neural network f with parameter θ, start state $\mathbf{z}(t_{n-1})$, start time t_{n-1}, and end time t_n. NDE is used to update hidden state by using an RNN cell

$$\mathbf{z}(t_n) = \text{RNNCell}(\hat{\mathbf{z}}(t_n), \mathbf{x}_{t_n}). \quad (3)$$

Here, the time index t_n in brackets in $\mathbf{z}(t_n)$ means continuous time while that in subscript in $\mathbf{x}_{t_n}$ means discrete time.

Discrete-Time Attention Mechanism

Traditional attention mechanism was developed to elevate the performance of sequential learning based on recurrent neural network (Bahdanau, Cho, and Bengio 2015; Luong, Pham, and Manning 2015; Su et al. 2018; Cao et al. 2018) where the discrete-time hidden states $\{\mathbf{z}_j\}$ (or $\{\mathbf{z}_{t_n}\}$) from the time-series observations $\{\mathbf{x}_j\}$ were represented by a recurrent machine. The discrete-time attention is then implemented by calculating the context vector $\mathbf{c}_i$ corresponding to a query vector $\mathbf{q}_i$ at each discrete time i by using the attention weights $\alpha_{i,j}$, which is yielded by a softmax function

$$\mathbf{c}_i = \sum_{j=1}^{N} \alpha_{i,j}\mathbf{z}_j, \text{ where } \alpha_{i,j} = \frac{\exp(\text{score}(\mathbf{q}_i, \mathbf{z}_j))}{\sum_{k=1}^{N} \exp(\text{score}(\mathbf{q}_i, \mathbf{z}_k))}. \quad (4)$$

The inner product can be used to compute the matching score between query $\mathbf{q}_i$ and state $\mathbf{z}_j$, i.e. $\text{score}(\mathbf{q}_i, \mathbf{z}_j) = \mathbf{q}_i^{\top}\mathbf{z}_j$, at discrete times i and j, respectively, where $1 \leq j \leq N$. Once the context vector $\mathbf{c}_i$ is calculated, the attended feature is usually obtained by the addition $\mathbf{q}_i + \mathbf{c}_i$. However, discrete-time attention scheme is infeasible to merge with the continuous-time state machine.

Continuous-Time Attention

This study presents a new sequential learning strategy where the continuous-time attention mechanism is seamlessly employed in continuous-time state machine. Figure 4 illustrates the conceptual difference between discrete-time attention and continuous-time attention. The curves in the bottom reflect the state trajectories $\mathbf{z}(t_n)$ and $\mathbf{z}_{t_n}$ of observation sample $\mathbf{x}_{t_n}$ at different time points $\{t_n\}_{n=1}^N$ in discrete-time and continuous-time state machines based on RNN and NDE, respectively. NDE predicts the continuous-time hidden states $\{\mathbf{z}(t_{n-1}), \mathbf{z}(t_n)\}$ between two observations $\{\mathbf{x}_{t_{n-1}}, \mathbf{x}_{t_n}\}$ which are more meaningful than those of RNN where only the hidden states for specific time points $\{t_{n-1}, t_n\}$ are represented. Therefore, discrete-time attention is defined for finite number of hidden states $\mathbf{z}_j$ while continuous-time attention is performed for continuous state function $\mathbf{z}(t)$. In what follows, we generalize the discrete-time attention to the continuous-time attention. Finding the context vector $\mathbf{c}(\tilde{t})$ corresponding to a query vector $\mathbf{q}(\tilde{t})$ using the summation is now extended to that using the integral.

Continuous-Time Generalization

In (Ramachandran et al. 2019; Cordonnier, Loukas, and Jaggi 2020), self attention was interpreted as a kind of convolution calculation in convolutional neural network. In contrast, this work calculates the attention weights based on the convolution operation which has been well defined in discrete-time and continuous-time signal processing. First, in discrete-time processing, the context vector $\mathbf{c}[\tilde{n}]$ is calculated by the convolution with attention weight $\alpha_{\tilde{n}}[n]$

$$\mathbf{c}[\tilde{n}] = \sum_{n=t_1}^{t_N} \alpha_{\tilde{n}}[n]\mathbf{z}[n], \text{ where } \alpha_{\tilde{n}}[n] = \exp(\mathbf{q}[\tilde{n}]^\top \mathbf{z}[n]). \tag{5}$$

where $\tilde{n}$ is the time index for query vector. Note that the context vector is denoted as a time series $\mathbf{c}[\tilde{n}]$ with the same time index as query vector $\mathbf{q}[\tilde{n}]$ which is defined as $\mathbf{q}[\tilde{n}] \triangleq \mathbf{q}_{\tilde{n}} \cdot \delta[n']$ where $\delta[n']$ denotes the delta function. The score function for finding attention weight is then obtained by

$$\mathbf{q}[\tilde{n}]^\top \mathbf{z}[n] = \sum_{n'=t_1}^{t_N} (\mathbf{q}_{\tilde{n}} \cdot \delta[n'])^\top \mathbf{z}[n+n'] = \mathbf{q}_{\tilde{n}}^\top \mathbf{z}[n]. \tag{6}$$

Next step is to generalize Eq. (5) to calculate the context vector using continuous-time convolution

$$\mathbf{c}(\tilde{t}) = \int_{t_1}^{t_N} \alpha_{\tilde{t}}(t)\mathbf{z}(t)dt \tag{7}$$

where the summation is replaced by the integral over continuous-time state trajectory $\mathbf{z}(t)$. $\tilde{t}$ indicates the time index of query vector. Attention weight is then generalized to

$$\alpha_{\tilde{t}}(t) = \exp\left(\int_{t_1}^{t_N} (\mathbf{q}_{\tilde{t}} \cdot \delta(t'))^\top \mathbf{z}(t+t')dt'\right). \tag{8}$$

Similar to Eq. (6), the score function is written by

$$\int_{t_1}^{t_N} (\mathbf{q}_{\tilde{t}} \cdot \delta(t'))^\top \mathbf{z}(t+t')dt' = \mathbf{q}_{\tilde{t}}^\top \mathbf{z}(t). \tag{9}$$

Model Implementation

In the implementation, the integral operation in continuous-time attention can be handled through ODE solver. Considering the ODE property, the solution is implemented by modeling the dynamics of hidden state, context vector as well as attention weight using neural networks. ODE solver is introduced to find the solution to multiple dynamics at the same time. To do so, we first rewrite Eq. (7) by meeting the format of ODE solver. Time variable t is required. We first compute the context vector $\mathbf{c}(\tilde{t})$. An additional context vector function $\mathcal{C}(t)$ is defined with the value in time point t_N such that $\mathcal{C}(t_N) = \mathbf{c}(\tilde{t})$. t is time variable while $\tilde{t}$ is a fixed time point of a query. This continuous-time function is particularly calculated by $\mathcal{C}(t) = \int_{t_1}^{t} \alpha_{\tilde{t}}(\tau)\mathbf{z}(\tau)d\tau$ which is reduced to Eq. (7) when time variable t equals to t_N. The meaning of time variable t is the continuous time that query vector attends. Next, the context vector function is expressed to fit the setting of ODE solver by using the start time $t_1 = 0$ and following the Leibniz integral rule in a form of

$$\frac{d\mathcal{C}(t)}{dt} = \alpha_{\tilde{t}}(t)\mathbf{z}(t) + \int_0^t \frac{\partial \alpha_{\tilde{t}}(\tau)\mathbf{z}(\tau)}{\partial t} = \alpha_{\tilde{t}}(t)\mathbf{z}(t) \tag{10}$$

Note that the attention weight $\alpha_{\tilde{t}}(t)$ defined in Eq. (8) is calculated without performing normalization similar to softmax function in Eq. (4) for discrete-time attention. However, the context vector should be normalized by using the summation of all attention scores along various times. We therefore define another continuous function $\mathcal{A}(t)$ for attention score function, which is used to represent the attention score summing up to the current time t. To apply ODE solver, the attention score function and its dynamic are expressed by

$$\mathcal{A}(t) = \int_0^t \alpha_{\tilde{t}}(\tau)d\tau, \quad \frac{d\mathcal{A}(t)}{dt} = \alpha_{\tilde{t}}(t). \tag{11}$$

Then, the normalization of context vector is performed via the division $\mathcal{C}(t)/\mathcal{A}(t)$. Finally, the solutions to differential equations of hidden state, context vector and attention score are simultaneously yielded by

$$\underbrace{\begin{bmatrix} \mathbf{z}(t_n) \\ \mathcal{C}(t_n) \\ \mathcal{A}(t_n) \end{bmatrix}}_{\text{solutions}} = \begin{bmatrix} \mathbf{z}(t_{n-1}) \\ \mathcal{C}(t_{n-1}) \\ \mathcal{A}(t_{n-1}) \end{bmatrix} + \int_{t_{n-1}}^{t_n} \underbrace{\begin{bmatrix} f(\mathbf{z}(t), t; \theta) \\ \alpha_{\tilde{t}}(t)\mathbf{z}(t) \\ \alpha_{\tilde{t}}(t) \end{bmatrix}}_{\text{dynamics}} dt \tag{12}$$

where the initial conditions are given by $\mathbf{z}(t_1) = \mathbf{x}_{t_1}$, $\mathcal{C}(t_1) = \mathbf{0}$ and $\mathcal{A}(t_1) = 0$. Figure 5 illustrates how ODE solver is applied to implement continuous-time attention. The dynamic functions of $\mathbf{z}(t)$, $\mathcal{C}(t)$ and $\mathcal{A}(t)$ are used to represent the hidden state trajectory, context vector function and attention score function, respectively, which are various

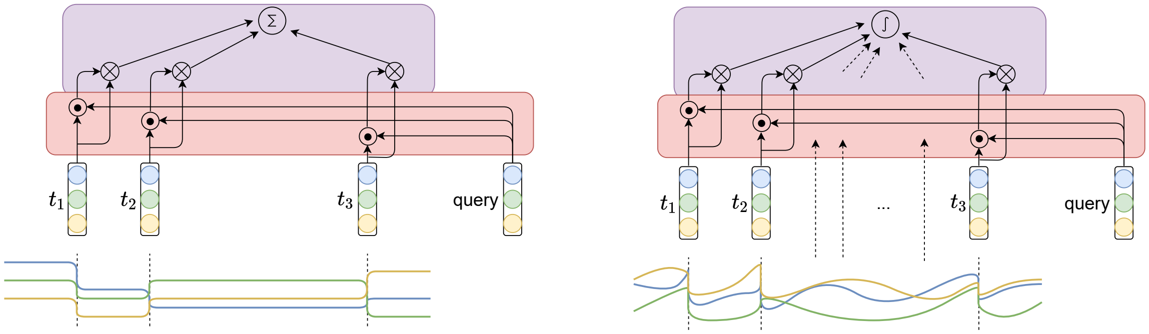
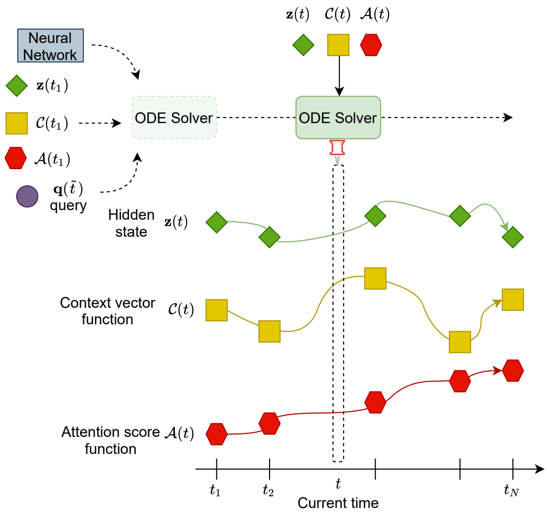
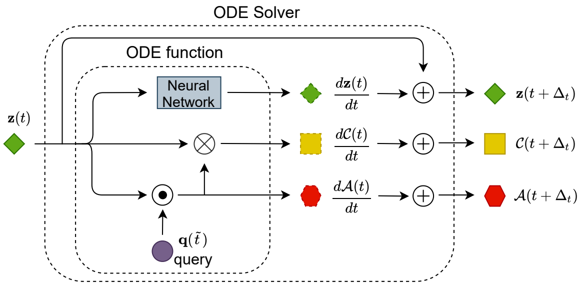


Figure 4: Comparison between discrete-time attention (left) and continuous-time attention (right). The discrete-time attention score is calculated by summing the dot-products between query and documents at time points $\{t_1, t_2, t_3\}$. But, the continuous-time attention score is computed by integrating and interpolating via ordinary differential equation over continuous time t.



Neural Network
$\mathbf{z}(t_1)$
$\mathcal{C}(t_1)$
$\mathcal{A}(t_1)$
$\mathbf{q}(\tilde{t})$ query
ODE Solver
$\mathbf{z}(t)$ $\mathcal{C}(t)$ $\mathcal{A}(t)$
ODE Solver
Hidden state $\mathbf{z}(t)$
Context vector function $\mathcal{C}(t)$
Attention score function $\mathcal{A}(t)$
t_1
t_2
t
Current time
t_N

Figure 5: ODE solver for continuous-time attention.



ODE Solver
ODE function
Neural Network
$\mathbf{z}(t)$
$\frac{d\mathbf{z}(t)}{dt}$
$\frac{d\mathcal{C}(t)}{dt}$
$\frac{d\mathcal{A}(t)}{dt}$
$\mathbf{z}(t+\Delta_t)$
$\mathcal{C}(t+\Delta_t)$
$\mathcal{A}(t+\Delta_t)$
$\mathbf{q}(\tilde{t})$ query

Figure 6: Dynamics of $\mathbf{z}(t)$, $\mathcal{C}(t)$ and $\mathcal{A}(t)$ in the attentive neural differential equation.

continuous-time functions with different markers and colors. ODE solver is seen as a black box with inputs consisting of neural network f, initial values $\mathbf{z}(t_1)$, $\mathcal{C}(t_1)$, $\mathcal{A}(t_1)$ as well as query $\mathbf{q}(\tilde{t})$. This solver is implemented from start time t_1 to end time t_N. The current time t is spotlighted. Notably, RNNs are continuously applied to update $\mathbf{z}(t)$ once a new sample $\mathbf{x}_{t_n}$ is observed at time t_n. Figure 6 shows the computation of derivatives or dynamics of various continuous-time functions inside the proposed attentive neural differential equation (Att-NDE). The ODE functions $f(\mathbf{z}(t), t; \theta)$, $\alpha_{\tilde{t}}(t)\mathbf{z}(t)$ and $\alpha_{\tilde{t}}(t)$ are calculated to solve the continuous-time functions $\mathbf{z}(t)$, $\mathcal{C}(t)$ and $\mathcal{A}(t)$, respectively. The dot-product of query and hidden state is used to update $\mathcal{A}(t)$ while the element-wise multiplication of current hidden state and derivative of attention score function is used to update $\mathcal{C}(t)$. After finding three derivatives $\frac{d\mathbf{z}(t)}{dt}$, $\frac{\mathcal{C}(t)}{dt}$ and $\frac{\mathcal{A}(t)}{dt}$, the values of next time point $t + \triangle_t$ are obtained by adding the current values with first-order derivatives. Algorithm 1 shows the overall procedure of Att-NDE where the continuous-time attention is performed by Algorithm 2. Basically, the continuous-time attention is obtained to attend hidden states by computing the normalized context vector between observations at time steps t_{n-1} and t_n. The augmented dynamic function f_{aug} is calculated as the gradients of $\mathbf{z}(t)$, $\mathcal{C}(t)$ and $\mathcal{A}(t)$ as $\mathbf{g}_z$, $\mathbf{g}_c$ and g_a, respectively, which are incorporated into ODE solver to find the corresponding continuous-time functions between t_{n-1} and t_n. ODE solver is a kind of integrator to find the integrated dynamics at any desired time instant t. RNN cell (Dupont, Doucet, and Teh 2019; Chien and Chen 2021; Chien and Ku 2015; Kuo and Chien 2018) is then used to update hidden state from $\hat{\mathbf{z}}(t_n)$ to $\mathbf{z}(t_n)$ when query point $\mathbf{x}_{t_n}$ (or $\mathbf{q}$) is observed. The attended feature $\mathbf{z}(t_n) + \mathcal{C}(t_n)/\mathcal{A}(t_n)$ is then computed and used to find classification output $\mathbf{y}_n$ via the classifier layer OutputNN$(\cdot)$. The classification loss is finally calculated and optimized to train Att-NDE.

Algorithm 1: Attentive neural differential equation

```
Input the parameter $\theta$, data points, time stamps
  $\{(\mathbf{x}_{t_n}, t_n)\}_{n=1}^{N}$, query $\mathbf{x}_{\tilde{t}}$
for each sample $\mathbf{x}_{t_n}$ at time $t_n$ do
    $\{\hat{\mathbf{z}}(t_n), \mathcal{C}(t_n), \mathcal{A}(t_n)\} =$
      $\text{CTA}(\theta, \mathbf{z}(t_{n-1}), \mathcal{C}(t_{n-1}), \mathcal{A}(t_{n-1}), t_{n-1}, t_n, \mathbf{x}_{\tilde{t}})$
    $\mathbf{z}(t_n) = \text{RNNCell}(\hat{\mathbf{z}}(t_n), \mathbf{x}_{t_n})$
end
$\mathbf{y}_{t_N} = \text{OutputNN}(\mathbf{z}(t_N) + \mathcal{C}(t_N)/\mathcal{A}(t_N))$
return $\mathbf{y}_{t_N}$
```

Algorithm 2: Continuous-time attention (CTA)

```
Input the parameter $\theta$, initial values $\mathbf{z}(t_{t_{n-1}}), \mathcal{C}(t_{n-1})$,
  $\mathcal{A}(t_{n-1})$, start time $t_{n-1}$, end time $t_n$, query $\mathbf{q}$
function $f_{\text{aug}}(\mathbf{z}(t), \mathbf{c}(t)), \mathbf{a}(t)$:
    $\mathbf{g}_z = f(\mathbf{z}(t), t, \theta)$
    $g_a = \mathbf{q}^\top \mathbf{z}(t)$
    $\mathbf{g}_c = g_a \mathbf{z}(t)$
    return $\{\mathbf{g}_z, \mathbf{g}_c, g_a\}$
end function
$\{\hat{\mathbf{z}}(t_n), \mathcal{C}(t_n), \mathcal{A}(t_n)\} =$
    ODESolver$(f_{\text{aug}}, \mathbf{z}(t_{n-1}), \mathcal{C}(t_{t_{n-1}}), \mathcal{A}(t_{n-1}), t_{n-1}, t_n, \theta)$
return $\hat{\mathbf{z}}(t_n), \mathcal{C}(t_n), \mathcal{A}(t_n)$
```

Extension to Self Attention

Self attention has been popular in sequential learning tasks (Vaswani et al. 2017). This paper presents a new self attention scheme based on the continuous-time state machine. Attention is performed by treating all of data samples of a sequence as query and working with the other samples of the same sequence as key and value. The same sample is transformed to find query, key and value using *individual* parameters. A general context vector $\mathbf{c}_i$ based on discrete-time attention is extended from Eq. (4) which is calculated by dot-product (or matching score) between query $\mathbf{W}_q \mathbf{z}_i = \mathbf{q}_i$ and key $\mathbf{W}_k \mathbf{z}_j = \mathbf{k}_j$, softmax, and then multiplication with value $\mathbf{W}_v \mathbf{z}_j = \mathbf{v}_j$ as

$$\mathbf{c}_i = \sum_j \frac{\exp(\mathbf{q}_i^\top \mathbf{k}_j)}{\sum_k \exp(\mathbf{q}_i^\top \mathbf{k}_k)} \cdot \mathbf{v}_j. \tag{13}$$

The continuous-time context vector function and the corresponding dynamic using self attention are modified as

$$\begin{aligned} \mathcal{C}(t) &= \int_0^t \exp(\mathbf{q}(\tilde{t})^\top \mathbf{k}(\tau)) \cdot \mathbf{v}(\tau) d\tau \\ \frac{d\mathcal{C}(t)}{dt} &= \exp(\mathbf{q}(\tilde{t})^\top \mathbf{k}(t)) \cdot \mathbf{v}(t). \end{aligned} \tag{14}$$

The attention score function and its dynamic are extended as

$$\mathcal{A}(t) = \int_0^t \exp(\mathbf{q}(\tilde{t})^\top \mathbf{k}(\tau)) d\tau, \ \frac{d\mathcal{A}(t)}{dt} = \exp(\mathbf{q}(\tilde{t})^\top \mathbf{k}(t)) \tag{15}$$

The continuous-time functions in numerator and denominator of Eq. (13) are calculated. Notably, self attention employs individual transformations to obtain query $\mathbf{q}_i$, key $\mathbf{k}_j$ and value $\mathbf{v}_j$. However, using standard attention, input sample $\mathbf{x}_i$ is used as query and state variable $\mathbf{z}_j$ is shared as key and value. Discrete-time attention is correspondingly extended to the continuous-time attention based on Eqs. (14)-(15).

Experiments

A set of experiments are conducted to evaluate the performance of continuous-time attention in sequential learning.

Evaluation on Action Recognition

Human activity dataset (Kaluza et al. 2010) was used as an action recognition task which contained *irregular* time series from five individuals with 3D positions of tags to their belt, chest and ankles. There were 12 observation features and eleven different actions including walking, sitting, lying, etc. For consistent comparison, we used the same preprocessing method as (Rubanova, Chen, and Duvenaud 2019) and combined similar activities like "lying" and "lying down", "sitting" and "sitting down". Different methods adopted the same hyperparameter setting as (Rubanova, Chen, and Duvenaud 2019). Number of training epoch was 200. Learning rate was initialized by 0.01 and decayed after each iteration by multiplying 0.999. Adamax (Kingma and Ba 2014) was used. Hidden state size was 15. Relative and absolute tolerances were 1e-3 and 1e-4 for solver, respectively. A six-layer fully-connected network was configured as ODE function. One-layer GRU was used as RNN cell. Classifier was built by three-layer fully-connected network.

It is important to investigate how the continuous-time attention in Att-NDE is working. Figure 7 shows the attention score function $\mathcal{A}$ calculated by ODE solver in four conditions where different settings of dropping samples from original sequence are considered. For the setting of dropping three important points which are near to query, it is found that Att-NDE can still pay attention in that missing region. High attention region is also extended. Att-NDE can compensate the missing region via continuous-time attention through the dynamic function. When dropping the unimportant time points far from the query location, Att-NDE simply ignores that region and obtains almost the same attention scores compared with the scores by using full sequence. The last setting is the case of dropping a wide range of samples. Interestingly, Att-NDE even attends the unimportant region. This case happens partially because the missing region is too large to ignore. Attention is needed in this situation. Next, the continuous-time attention is evaluated by comparing the predictions using NDE and Att-NDE. Table 1 shows that Att-NDE is robust to obtain comparable results even when a wide range of samples are missing while NDE could not preserve the predictions. Such a phenomenon still happens in case of random dropping. This is because that the attention mechanism in Att-NDE can capture the history information to learn a reasonable state trajectory.

Table 2 compares the accuracy and parameter size of different methods. The work in (Rubanova, Chen, and Duvenaud 2019) was trained by using time-invariant dynamics $\frac{d\mathbf{z}(t)}{dt} = f(\mathbf{z}(t), \theta)$. Att-NDE carries out the time-variant dynamics in Eq. (1). The results of our implementation and that work are both reported. Δ_t implies the implementation by treating time information as a new augmented feature. Basically, the discrete-time state machine with attention (RNN + Att relative to RNN) was degraded. Next, the time-variant and invariant dynamics with NDE (w/ and w/o time) are compared. Time-variant dynamics work well. In addition, we combine NDE with *discrete-time* attention mechanism to examine the other two implementations NDE + Att (w/ and w/o time). Interestingly, adding discrete-time attention does not help, even degrades the performance of NDE (w/time) setting. This is because that discrete-time attention could not property characterize temporal information from irregular time series. Att-NDE achieves the highest accuracy

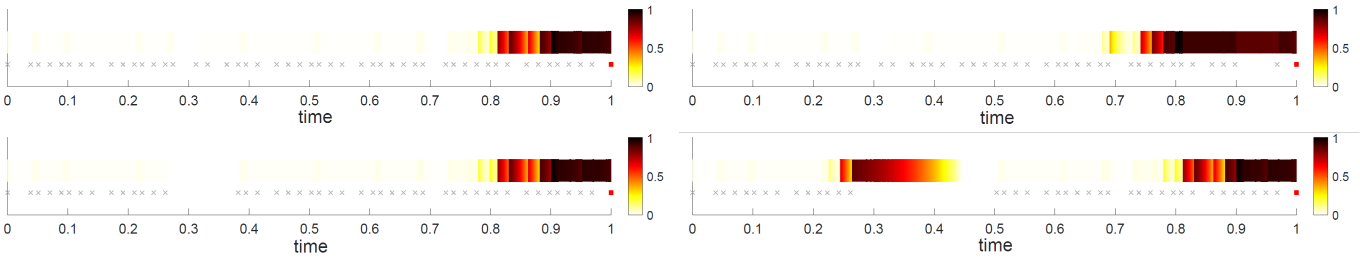


Figure 7: Illustration of how continuous-time attention values (via the darkness of red) are affected by four cases. × denotes the data samples at the corresponding time points. Red rectangular in time end is the query. The score functions are shown with different settings containing full sequence (top left), sequence dropped by a slice of time points which are important (top right) and unimportant (bottom left), and dropped by a wide range of samples (bottom right). Human activity dataset is used.

Model	Prediction for two rows of sequence data
NDE	0 0 0 3 3 3 3 3 3 3 3 3 3 1 1 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 6 6 6 6 6 6 6 6 6 6 6 6 6 6
Att-NDE	1 1 1 1 1 1 1 1 1 1 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6
labels	3 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6
NDE	0 0 0 3 3 3 3 3 3 3 3 – – – – – – – – – – 3 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2
Att-NDE	1 1 1 1 1 1 1 1 1 1 6 – – – – – – – – – – 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6
labels	3 3 1 1 1 1 1 1 1 1 1 – – – – – – – – – – 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6
NDE	0 0 – 0 – – 0 0 0 0 0 0 – 0 0 – – 0 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 – 0 1 1 – 6 6 – 6 6 6 – 6 6 6
Att-NDE	1 1 – 1 – – 1 1 1 1 1 1 – 1 1 – – 1 1 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 – 6 6 6 – 6 6 – 6 6 6 – 6 6 1
labels	3 3 – 1 – – 1 1 1 1 1 1 – 1 1 – – 1 1 1 1 6 6 6 6 6 6 6 6 6 6 6 6 6 6 – 6 6 6 – 6 6 – 6 6 6 – 6 6 6

Table 1: Comparison of predictions using different models and settings. From top to bottom are the predictions for full sequence, dropping wide range of data, and dropping 10 time points randomly. 0:*walking*. 1:*falling*. 2:*lying*. 3:*sitting*. 4:*standing up*. 5:*on all fours*. 6:*sitting on the ground*.

even higher than latent ODE (Rubanova, Chen, and Duvenaud 2019) which was the best among the previous methods. Number of parameters is comparable for different NDEs.

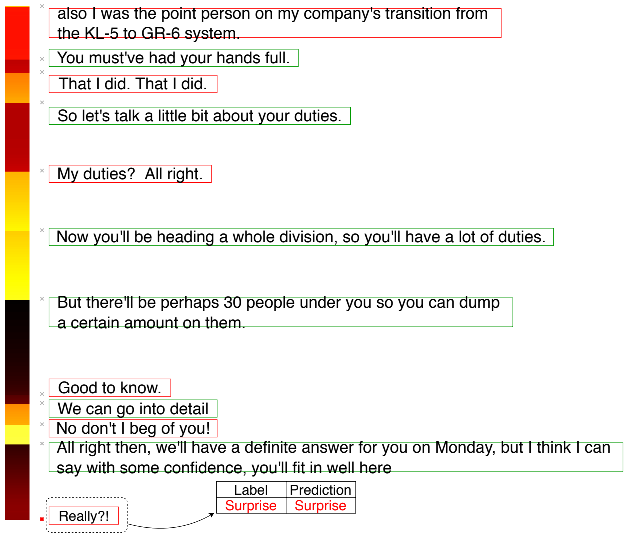
Evaluation on Emotion Recognition

Multimodal EmotionLines Dataset (MELD) (Poria et al. 2019) contained the dialogue instances collected from Friends TV series. Conventionally, the dialogue sequences were treated as regular time series. However, each utterance had different sequence lengths. Representing irregular time series in a spoken dialogue (Chien and Lieow 2019; Chien and Hsu 2020) is desirable for dialogue modeling. MELD had not only text information but also audio and visual modalities. There were 1433 dialogues and 13708 utterances. Each utterance in a dialogue was labeled by one of seven emotions including anger, joy, neutral, etc. For preprocessing procedure, the Glove embedding was employed to embed all tokens of an utterance, and then compute the average of those embeddings to represent the utterance. This study ignored the audio and visual clips, and stamped the start time of each utterance as time index. Then, the utterances in dialogue could be seen as irregular time series. The number of epochs was 300 and the learning rate 0.001 was used. Hidden state size was 50. ODE function was represented by a five-layer fully-connected network. Other settings were the same as those in the first task. Att-NDE with self-attention was evaluated.

Models	Accuracy (%)	Param
RNN Δ_t	78.7 (79.7*)	–
RNN Δ_t + att	77.6	–
NDE (w/o time)	83.3 (82.9*)	1.13M
NDE (w/ time)	84.3	1.14M
NDE + att (w/o time)	83.5	1.13M
NDE + att (w/ time)	83.3	1.14M
Latent ODE	84.6 (84.6*)	1.70M
Att-NDE	**86.8**	1.13M

Table 2: Comparison of accuracy and number of parameters on action recognition using different methods. * means the result from (Rubanova, Chen, and Duvenaud 2019).

Figure 8 shows an example of attention score function $\mathcal{A}(t)$. In this example, “Really?!” is served as query and Att-NDE is used to predict the emotion of this utterance. Those utterances marked by red were said by the interviewee and green were said by the interviewer. Att-NDE pays most of the attention on three utterances, which is “So let’s talk about ...”, “But there’ll be perhaps ...” and “All right then, we’ll have ...”. All of them were said by interviewer. This is reasonable that the emotion of the interviewee would be affected by the interviewer. Att-NDE also attends on “Good to know.”, which may not be well attended. It is because that this one is used to answer two utterances, “So let’s talk about ...” and “But there’ll be perhaps ...”. Namely, Att-NDE predicts how the interviewee will reply to those two utterances, which are related to the query. Another example is provided in Figure 9 where five persons were chatting. In this situation, person B couldn’t distinguish two girls and made some mistake. Att-NDE pays the highest attention on utterances


also I was the point person on my company's transition from the KL-5 to GR-6 system.

You must've had your hands full.

That I did. That I did.

So let's talk a little bit about your duties.

My duties? All right.

Now you'll be heading a whole division, so you'll have a lot of duties.

But there'll be perhaps 30 people under you so you can dump a certain amount on them.

Good to know.

We can go into detail

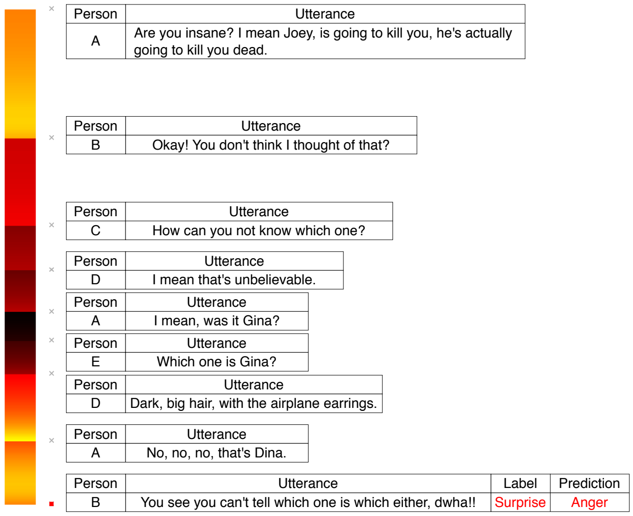
No don't I beg of you!

All right then, we'll have a definite answer for you on Monday, but I think I can say with some confidence, you'll fit in well here

Label	Prediction
Surprise	Surprise

Really?!

Figure 8: An example of dialogue with two persons.


Person	Utterance
A	Are you insane? I mean Joey, is going to kill you, he's actually going to kill you dead.

Person	Utterance
B	Okay! You don't think I thought of that?

Person	Utterance
C	How can you not know which one?

Person	Utterance
D	I mean that's unbelievable.

Person	Utterance
A	I mean, was it Gina?

Person	Utterance
E	Which one is Gina?

Person	Utterance
D	Dark, big hair, with the airplane earrings.

Person	Utterance
A	No, no, no, that's Dina.

Person	Utterance	Label	Prediction
B	You see you can't tell which one is which either, dwha!!	Surprise	Anger

Figure 9: An example of dialogue with five persons.

“I mean, was it Gina?” and “Which one is Gina?”. This example shows that the other persons still couldn't distinguish these two girls. Another two utterances “How can you not know which one?” and “I mean that's unbelievable.” are also attended by our model, which blamed person B. Although the label is “surprise”, the prediction “anger” is also acceptable. From the results in two tasks, it is obvious that the behavior of sequence data is substantially reflected by the attention scores. For human activity, which is irregular time series, the attention score function is smoother and more like continuous function. While MELD, which is seen as regular time series in literature, is more like discrete function. Stamping start time as time index may be too naive. Table 3 shows the weighted average of precision and recall (i.e. F1 score) using different methods. Att-NDE with self attention achieves the best performance in sentiment classification.

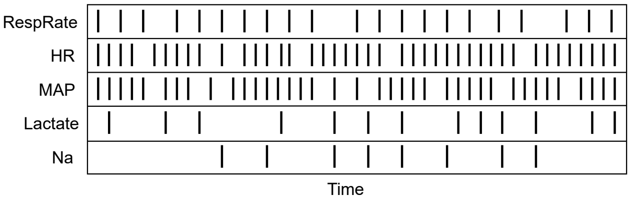


Figure 10: Different features from irregular samples in PhysioNet. Bold bars indicate the observation time points.

Models	F1-score	att type
RNN Δ_t	0.539	-
NDE	0.551	-
Att-NDE	0.560	att
Att-NDE	**0.565**	self-att

Table 3: F1-score on MELD.

Model	AUC
RNN Δ_t	0.783
NDE	0.826
Att-NDE	**0.833**

Table 4: AUC on PhysioNet.

Evaluation on Mortality Prediction

PhysioNet (Silva et al. 2012) was collected from the intensive care unit (ICU) containing the first forty eight hours of patients' physiological signals like respiration rate, heart rate (HR), etc. There were four time-invariant features including age, gender, height and ICU type. Figure 10 shows the scenarion of irregular samples. This task is to predict in-hospital mortality rate. Hyperparameter setting was similar to the previous tasks. Number of epochs was 40 and hidden state size was 20. ODE function was built by a five-layer fully-connected network. Because positive samples only have 13.75%, area under the curve (AUC) is used to evaluate model performance as shown in Table 4. Att-NDE performs better than RNN and NDE in terms of AUC.

Conclusions

This paper presented the continuous-time attention for sequential learning over irregular sequence data. This attention scheme was derived by merging with neural differential equation to build continuous-time state machine. This Att-NDE represented the mapping from observations to targets where the continuous-time functions of attention score and context vector were computed. The experimental results showed that adding continuous-time attention did improve the robustness to missing time samples.The property in continuous-time attention was investigated. In future works, the limitation for Att-NED will be handled. In self attention setting, we basically feed all of query vectors to ODE solver and solve them individually, which is memory inefficient. In addition, we will extend our methods to other NDE methods such as latent ODE (Chen et al. 2018; Rubanova, Chen, and Duvenaud 2019) or neural stochastic differential equation (Liu et al. 2019) where stochastic property is preserved. The proposed attention is also feasible to other types of time series (Yildiz, Heinonen, and Lahdesmaki 2019; Jia and Benson 2019). The time information of each word rather than each utterance will be used for emotion recognition.

Acknowledgments

This work is supported by the Ministry of Science and Technology, Taiwan, under MOST 109-2634-F-009-024.

References

Bahdanau, D.; Cho, K.; and Bengio, Y. 2015. Neural machine translation by jointly learning to align and translate. In *Proc. of International Conference on Learning Representations*.

Bauer, S.; Schölkopf, B.; and Peters, J. 2016. The arrow of time in multivariate time series. In *Proc. of International Conference on Machine Learning*, 2043–2051.

Cao, W.; Wang, D.; L., J.; Zhou, H.; Li, L.; and Li, Y. 2018. BRITS: bidirectional recurrent imputation for time series. In *Advances in Neural Information Processing Systems*, 6775–6785.

Che, Z.; Purushotham, S.; Cho, K.; Sontag, D. A.; and Liu, Y. 2018. Recurrent reural networks for multivariate time series with missing values. *Scientific Reports* 8.

Chen, R. T. Q.; Rubanova, Y.; Bettencourt, J.; and Duvenaud, D. K. 2018. Neural ordinary differential equations. In *Advances in Neural Information Processing Systems*, 6571–6583.

Chien, J.-T. 2019. Deep Bayesian natural language processing. In *Proc. of Annual Meeting of the Association for Computational Linguistics : Tutorial Abstracts*, 25–30.

Chien, J.-T.; and Chen, Y.-H. 2021. Continuous-time self-attention in neural differential equation. In *Proc. of International Conference on Acoustics, Speech, and Signal Processing*.

Chien, J.-T.; and Hsu, P.-C. 2020. Stochastic curiosity exploration for dialogue systems. In *Proc. of Annual Conference of International Speech Communication Association*, 3885–3889.

Chien, J.-T.; and Ku, Y.-C. 2015. Bayesian recurrent neural network for language modeling. *IEEE Transactions on Neural Networks and Learning Systems* 27(2): 361–374.

Chien, J.-T.; and Lieow, W. X. 2019. Meta learning for hyperparameter optimization in dialogue system. In *Proc. of Annual Conference of International Speech Communication Association*, 839–843.

Chien, J.-T.; and Lin, T.-A. 2018. Supportive attention in end-to-end memory networks. In *Proc. of IEEE International Workshop on Machine Learning for Signal Processing*, 1–6.

Chien, J.-T.; and Lin, T.-A. 2020. Supportive and Self Attentions for Image Caption. In *Proc. of Asia-Pacific Signal and Information Processing Association Annual Summit and Conference*, 1713–1718.

Chien, J.-T.; and Wang, C.-W. 2019. Self attention in variational sequential learning for summarization. In *Proc. of Annual Conference of International Speech Communication Association*, 1318–1322.

Chuang, Y.-H.; Huang, C.-L.; Chang, W.-W.; and Chien, J.-T. 2020. Automatic Classification of Myocardial Infarction Using Spline Representation of Single-Lead Derived Vectorcardiography. *Sensors* 20(24): 7246.

Cordonnier, J.-B.; Loukas, A.; and Jaggi, M. 2020. On the relationship between self-attention and convolutional layers. In *Proc. of International Conference on Learning Representations*.

Dupont, E.; Doucet, A.; and Teh, Y. W. 2019. Augmented neural ODEs. In *Advances in Neural Information Processing Systems*, 3140–3150.

El Naqa, I.; Pandey, G.; Aerts, H.; Chien, J.-T.; Andreassen, C. N.; Niemierko, A.; and Ten Haken, R. K. 2018. Radiation therapy outcomes models in the era of radiomics and radiogenomics: uncertainties and validation. *International Journal of Radiation Oncology• Biology• Physics* 102(4): 1070–1073.

Elman, J. L. 1990. Finding structure in time. *Cognitive Science* 14(2): 179–211.

Jia, J.; and Benson, A. R. 2019. Neural jump stochastic differential equations. In *Advances in Neural Information Processing Systems*, 9847–9858.

Kaluza, B.; Mirchevska, V.; Dovgan, E.; Lustrek, M.; and Gams, M. 2010. An agent-based approach to care in independent living. In *Proc. of International Joint Conference on Ambient Intelligence*, 177–186.

Kingma, D. P.; and Ba, J. 2014. Adam: a method for stochastic optimization. *arXiv preprint arXiv:1412.6980* abs/1412.6980.

Kuo, C.-Y.; and Chien, J.-T. 2018. Markov recurrent neural networks. In *Proc. of IEEE International Workshop on Machine Learning for Signal Processing (MLSP)*, 1–6.

Lipton, Z. C.; Kale, D.; and Wetzel, R. 2016. Directly modeling missing data in sequences with RNNs: improved classification of clinical time series. In *Proc. of Machine Learning for Healthcare Conference*, 253–270.

Liu, X.; Xiao, T.; Si, S.; Cao, Q.; Kumar, S. P. S.; and Hsieh, C.-J. 2019. Neural SDE: stabilizing neural ODE networks with stochastic noise. *arXiv preprint arXiv:1906.02355* .

Luong, T.; Pham, H.; and Manning, C. D. 2015. Effective approaches to attention-based neural machine translation. In *Proc. of Conference on Empirical Methods in Natural Language Processing*, 1412–1421.

Poria, S.; Hazarika, D.; Majumder, N.; Naik, G.; Cambria, E.; and Mihalcea, R. 2019. MELD: a multimodal multi-party dataset for emotion recognition in conversations. In *Proc. of Annual Meeting of the Association for Computational Linguistics*.

Ramachandran, P.; Parmar, N.; Vaswani, A.; Bello, I.; Levskaya, A.; and Shlens, J. 2019. Stand-alone self-attention in vision models. In *Advances in Neural Information Processing Systems*, 68–80.

Rubanova, Y.; Chen, R. T. Q.; and Duvenaud, D. K. 2019. Latent ordinary differential equations for irregularly-sampled time series. In *Advances in Neural Information Processing Systems*, 5320–5330.

Shi, X.; Chen, Z.; Wang, H.; Yeung, D.-Y.; Wong, W.; and Woo, W. 2015. Convolutional LSTM network: a machine learning approach for precipitation nowcasting. In *Advances in Neural Information Processing Systems*, 802–810.

Silva, I.; Moody, G. B.; Scott, D. J.; Celi, L.; and Mark, R. G. 2012. Predicting in-hospital mortality of ICU patients: The PhysioNet/Computing in cardiology challenge 2012. *Computing in Cardiology* 245–248.

Su, J.; Wu, S.; Xiong, D.; Lu, Y.; Han, X.; and Zhang, B. 2018. Variational recurrent neural machine translation. In *Proc. of AAAI Conference on Artificial Intelligence*, 5488–5495.

Tseng, H.-H.; Luo, Y.; Cui, S.; Chien, J.-T.; Ten Haken, R. K.; and El Naqa, I. 2017. Deep reinforcement learning for automated radiation adaptation in lung cancer. *Medical physics* 44(12): 6690–6705.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, 5998–6008.

Wang, D.; Cao, W.; Li, J.; and Ye, J. 2017. DeepSD: supply-demand prediction for online car-hailing services using deep neural networks. In *Proc. of International Conference on Data Engineering*, 243–254.

Yildiz, C.; Heinonen, M.; and Lahdesmaki, H. 2019. ODE2VAE: deep generative second order ODEs with Bayesian neural networks. In *Advances in Neural Information Processing Systems*, 13412–13421.

Zhang, J.; Zhang, P.; Kong, B.; Wei, J.; and Jiang, X. 2021. Continuous Self-Attention Models with Neural ODE Networks. In *Proc. of AAAI Conference on Aritificial Intelligence*.