Multi-Decoder RNN Autoencoder Based on Variational Bayes Method

Daisuke Kaji
AI R & D Division
Denso Corporation
Tokyo, Japan
daisuke.kaji.j3a@jp.denso.com

Kazuho Watanabe
Dept. of CSE
Toyohashi University of Technology
Aichi, Japan
wkazuho@cs.tut.ac.jp

Masahiro Kobayashi
Dept. of CSE
Toyohashi University of Technology
Aichi, Japan
kobayashi@lisl.cs.tut.ac.jp

***Abstract*—Clustering algorithms have wide applications and play an important role in data analysis fields including time series data analysis. However, in time series analysis, most of the algorithms used signal shape features or the initial value of hidden variable of a neural network. Little has been discussed on the methods based on the generative model of the time series. In this paper, we propose a new clustering algorithm focusing on the generative process of the signal with a recurrent neural network and the variational Bayes method. Our experiments show that the proposed algorithm not only has a robustness against for phase shift, amplitude and signal length variations but also provide a flexible clustering based on the property of the variational Bayes method.**

***Index Terms*—Time series analysis, Clustering, Recurrent neural network, Variational Bayes**

I. Introduction

The rapid progress of IoT technology has brought huge data in wide fields such as traffic, industries, medical research and so on. Most of these data are gathered continuously and accumulated as time series data, and the extraction of features from a time series have been studied intensively in recent years. The difficulty of time series analysis is the variation of the signal in time which gives rise to phase shift, compress/stretch and length variation. Many methods have been proposed to solve these problems. Dynamic Time Warping (DTW) was designed to measure the distance between warping signals [1]. This method solved the compress/stretch problem by applying a dynamic planning method. Fourier transfer or wavelet transfer can extract the features based on the frequency components of signals. The phase shift independent features are obtained by calculating the power spectrum of the transform result.

In recent years, the recurrent neural network (RNN), which has recursive neural network structure, has been widely used in time series analysis [2], [3]. This recursive network structure makes it possible to retain the past information of time series. Furthermore, this architecture enables us to apply this algorithm to signals with different lengths. Although the methods mentioned above are effective solutions for the compress/stretch, phase shift and signal length variation issues, little has been studied about these problems comprehensively.

Let us turn our attention to feature extraction again. Unsupervised learning using a neural network architecture autoencoder (AE) has been studied as a feature extraction method [4]–[6]. AE using RNN structure (RNN-AE) has also been proposed [7] and it has been applied to real data such as driving data [8] and others. RNN-AE can be also interpreted as the discrete dynamical system: chaotic behavior and the deterrent method have been studied from this point of view [9], [10].

In this paper, we propose a new clustering algorithm for feature extraction focusing on the dynamical system aspect of RNN-AE. In order to achieve this, we employed a multi-decoder AE to describe different dynamical systems as a generative model. We also applied the variational Bayes method [11]–[13] as the clustering algorithm.

This paper is composed as follows: in Section III, we explain AE from a dynamical system view, then we define our model and from this, derive its learning algorithm. In Section V, we describe the application of our algorithm to an actual time series to show its robustness, including experiments using periodic data, complex periodic data and driving data. Finally we summarize our study and describe our future work in Section VII.

II. Related work

A lot of excellent clustering/representation algorithms of data using AE have been studied so far [14]. Song et al. [15] integrated the distance between data and centroids into an objective function to obtain a cluster structure in the encoded data space. Pineau and Lelarge [16] proposed a generative model based on the variational autoencoder (VAE) [17] with a clustering structure as a prior distribution, VAE was also applied to the hierarchical clustering method of time series data [18]. Wang et al. [19] achieved a high separability clustering result by adding a regularization term for the orthogonality and balanced clusters of the encoded data. These, however, are regularization methods of the objective function, and focused on only the distribution of the encoded data as the initial value of decoder.

They did not give the clustering policy based on the decoder structure, namely, the reconstruction process of the data. From dynamical system point of view, one decoder of RNN-AE corresponds to a single dynamics in the space of latent representation. Hence, it is natural to equip RNN-AE with multiple decoders to implement multiple dynamics. Such an

978-1-7281-6926-2/20/$31.00 ©2020 IEEE

extension of RNN-AE, however, has yet to be proposed in related works to the best of our knowledge. It can also possibly be incorporated into the framework of VAE by treating the output of the RNN encoder as a latent random variable [17].

III. Recurrent Neural Network and Dynamical System

A. Recurrent Neural Network Using Unitary Matrix

RNN is a neural network designed for time series data. The architecture of the main unit is called cell, and mathematical expressions are shown in Fig. 1 and Eq. (1).

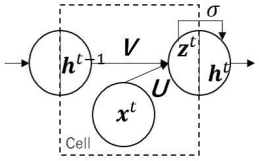

Fig. 1. RNN Cell

Suppose we are given a time series,

$$\mathbf{X} = (\boldsymbol{X}^1, \cdots, \boldsymbol{X}^n, \cdots, \boldsymbol{X}^N),\ \boldsymbol{X}^n = (\boldsymbol{x}_n^1, \cdots, \boldsymbol{x}_n^t, \cdots, \boldsymbol{x}_n^T),$$
$$\boldsymbol{x}_n^t \in \mathbb{R}^D,\ n = 1, \cdots, N,\ t = 1, \cdots, T,$$

where D denotes data dimension. RNN, unlike the usual feed-forward neural network, operates the same transform matrix to the hidden valuable recursively,

$$\boldsymbol{z}^t = \boldsymbol{V}\boldsymbol{h}^{t-1} + \boldsymbol{U}\boldsymbol{x}^t + \boldsymbol{b},\ \ \boldsymbol{h}^t = \sigma(\boldsymbol{z}^t), \tag{1}$$

where $\sigma(\cdot)$ is an activation function and $\boldsymbol{z}^t,\ \boldsymbol{h}^t,\ \boldsymbol{b} \in \mathbb{R}^L$. This recursive architecture makes it possible to handle signals with different lengths, although it is vulnerable to the vanishing gradient problem as with the deep neural network (DNN) [2], [3]. Long short-term memory (LSTM) and gated recurrent unit (GRU) are widely known solutions to this problem [20]–[22]. These methods have the extra mechanism called a gate structure to control output scaling and retaining/forgetting of the signal information. Though this mechanism works effectively in many application fields [23], [24], the architecture of network is relatively complicated. As an alternative simpler method to solve this problem, the algorithm using a unitary matrix as the transfer matrix $\boldsymbol{V}$ was proposed in recent years [25]–[29]. Since the unitary matrix does not change the norm of the variable vector, we can avoid the vanishing gradient problem. In addition, the network architecture remains unchanged from the original RNN.

In this paper, we focus on the dynamical system aspect of the original RNN. We employ the unitary matrix type RNN to take advantage of this dynamical system structure. However, to implement the above method, we need to find the transform matrix $\boldsymbol{V}$ in the space of unitary matrices $\mathbb{U} = \{\boldsymbol{U}(L) \in \mathrm{GL}(L) | \boldsymbol{U}(L)^* \boldsymbol{U}(L) = \boldsymbol{I}\}$, where $\mathrm{GL}(L)$ is the set of complex-valued general linear matrices with size $L \times L$ and $*$ means the adjoint matrix. Several methods to find the transform matrix from $\mathbb{U}$ has been reported so far [25]–[29]. Here, we adopt the method proposed by [26].

B. RNN Autoencoder and Dynamical System

The architecture of AE using RNN is shown in Fig. 2. AE is composed of an encoder unit and a decoder unit. The parameters $(\boldsymbol{V}_{enc}, \boldsymbol{U}_{enc}, \boldsymbol{V}_{dec}, \boldsymbol{U}_{dec})$ are trained by minimizing $\|\boldsymbol{X} - \boldsymbol{X}_{dec}\|_F^2 = \sum_{t=1}^T \|\boldsymbol{x}^t - \boldsymbol{x}_{dec}^t\|^2$, where $\boldsymbol{X}$ is the input data and $\boldsymbol{X}_{dec}$ is the decoded data.

The input data is recovered from only the encoded signal $\boldsymbol{h}$ using the matrix $(\boldsymbol{V}_{dec}, \boldsymbol{U}_{dec})$, therefore $\boldsymbol{h}$ is considered as the essential information of the input signal. When focusing

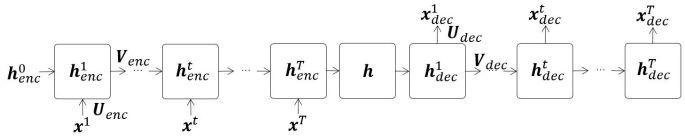

Fig. 2. Architecture of RNN Autoencoder

on the transformation of the hidden variable, this recursive operation has the same structure of a discrete dynamical system expression as described in the following equation:

$$\boldsymbol{h}^t = f(\boldsymbol{h}^{t-1}), \tag{2}$$

where f is given by Eq. (1). From this point of view, we can understand that RNN describes the universal dynamical system structure which is common to the all input signals by the reconstruction process in Fig. 2.

IV. Derivation of Multi-Decoder RNN AE Algorithm

In this section, we will give the architecture of the Multi-Decoder RNN AE (MDRA) and its learning algorithm. As we discussed in the previous section, RNN can extract the dynamical system characteristics of the time series. In the case of the original RNN, the model expresses just one dynamical system, hence all input data are recovered from the encoded result $\boldsymbol{h}$ by the same recovery rule. Therefore $\boldsymbol{h}$ is usually used as the feature value of the input data. In contrast, in this paper, we focus on the transformation rule itself. For this purpose, we propose MDRA which has multiple decoders to extract various dynamical system features. The architecture of MDRA is shown in Fig. 3. Let us put $\mathbf{W}_{dec}^k = (\boldsymbol{V}_{dec}^k, \boldsymbol{U}_{dec}^k)$ for $k = 1, \cdots, K$, $\mathbf{W}_{enc} = (\boldsymbol{V}_{enc}, \boldsymbol{U}_{enc})$, and $\mathbf{W} = (\mathbf{W}_{enc}, \mathbf{W}_{dec}^1, \cdots, \mathbf{W}_{dec}^K)$. We will derive the learning algorithm to optimize the whole set of parameters $\mathbf{W}$ in the following section.

A. Decomposition of Free Energy

We applied a clustering method to derive the learning algorithm of MDRA. Many clustering algorithms have been proposed: here we employ the variational Bayes (VB) method, because the VB method enables us to adjust the number of clusters by tuning the hyperparameters of a prior distribution

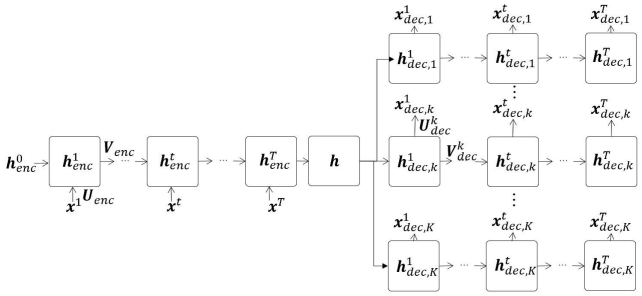

Fig. 3. Architecture of MDRA

[13], [30]. We first define free energy, which is negative log-marginal-likelihood, by the following equation,

$$F_{\mathbf{X}}(\mathbf{W}) = -\log \int\int \left\{ \prod_{n=1}^{N} \sum_{\boldsymbol{y}_n} p_{\mathbf{W}}(\boldsymbol{X}^n|\boldsymbol{y}_n, \boldsymbol{h}_n, \beta) p(\boldsymbol{y}_n|\boldsymbol{\alpha}) \right\} \cdot p(\boldsymbol{\alpha}) p(\beta) d\boldsymbol{\alpha} d\beta, \tag{3}$$

where $\mathbf{X}$ is data tensor defined in Section III and $\mathbf{W}$ is parameter tensor of MDRA defined above. $\boldsymbol{Y} = (\boldsymbol{y}_1, \boldsymbol{y}_2, \cdots, \boldsymbol{y}_N)$ is the set of latent variables each of which means an allocation for a decoder. That is, $\boldsymbol{y}_n = (y_{n1}, \cdots, y_{nK})^{\mathrm{T}} \in \mathbb{R}^K$, where $y_{nk} = 1$ if $\boldsymbol{X}^n$ is allocated to the k-th decoder and otherwise $y_{nk} = 0$. $p_{\mathbf{W}}(\boldsymbol{X}^n|\boldsymbol{y}_n, \boldsymbol{h}_n, \beta)$ is the probability density function representation of MDRA parametrized by tensor $\mathbf{W}$, $p(\boldsymbol{\alpha})$ and $p(\beta)$ are its prior distributions for a probability vector $\boldsymbol{\alpha} = (\alpha_1, \cdots, \alpha_K)$ and a precision parameter $\beta > 0$. We applied the Gaussian mixture model as our probabilistic model. Hence $p(\boldsymbol{\alpha})$ and $p(\beta)$ were given by Dirichlet and gamma distributions respectively which are the conjugate prior distributions of multinomial and Gaussian distributions. These specific distributions are given as follows:

$$p_{\mathbf{W}}(\mathbf{X}, \boldsymbol{Y}, \boldsymbol{\alpha}, \beta|\boldsymbol{H}) = p_{\mathbf{W}}(\mathbf{X}|\boldsymbol{Y}, \boldsymbol{H}, \beta) p(\boldsymbol{Y}|\boldsymbol{\alpha}) p(\boldsymbol{\alpha}) p(\beta),$$

$$p(\boldsymbol{\alpha}) = \frac{\Gamma(\theta_0 K)}{\Gamma(\theta_0)^K} \prod_{k=1}^{K} \alpha_k^{\theta_0 - 1}, p(\beta) = \frac{\lambda_0^{\nu_0}}{\Gamma(\nu_0)} \beta^{\nu_0 - 1} \exp(-\lambda_0 \beta),$$

$$p_{\mathbf{W}}(\mathbf{X}|\boldsymbol{Y}, \boldsymbol{H}, \beta) = \prod_{n=1}^{N} p_{\mathbf{W}}(\boldsymbol{X}^n|\boldsymbol{y}_n, \boldsymbol{h}_n, \beta),$$

$$p_{\mathbf{W}}(\boldsymbol{X}^n|\boldsymbol{y}_n, \boldsymbol{h}_n, \beta) = \prod_{k=1}^{K} \left\{ \left(\frac{\beta}{\pi} \right)^{\frac{T_n D}{2}} \exp(-\beta \|\boldsymbol{X}^n - g(\boldsymbol{h}_n|\mathbf{W}_{dec}^k)\|_F^2) \right\}^{y_{nk}},$$

$$p(\boldsymbol{Y}|\boldsymbol{\alpha}) = \prod_{n=1}^{N} p(\boldsymbol{y}_n|\boldsymbol{\alpha}), \;\; p(\boldsymbol{y}_n|\boldsymbol{\alpha}) = \prod_{k=1}^{K} \alpha_k^{y_{nk}}.$$

Here, $\theta_0 > 0, \nu_0 > 0$ and $\lambda_0 > 0$ are hyperparameters and $g(\boldsymbol{h}_n|\mathbf{W}_{dec}^k) = \boldsymbol{X}_{dec,k}^n$ denotes decoder mapping of RNN from the encoded n-th data $\boldsymbol{h}_n$, $\boldsymbol{H} = (\boldsymbol{h}_1, \cdots, \boldsymbol{h}_N)$ and $T_n D$ is the total signal dimension of input signal $\boldsymbol{X}^n$ including dimension of input data. To apply the variational Bayes algorithm, we then derive the upper bound of the free energy by applying Jensen's inequality,

$$\begin{aligned} F_{\mathbf{X}}(\mathbf{W}) &= -\log \mathbb{E}_{\bar{q}} \left[\frac{p_{\mathbf{W}}(\mathbf{X}|\boldsymbol{Y}, \boldsymbol{H}, \beta) p(\boldsymbol{Y}|\boldsymbol{\alpha}) p(\boldsymbol{\alpha}) p(\beta)}{q(\boldsymbol{Y}) q(\boldsymbol{\alpha}) q(\beta)} \right] \\ &\leq D_{\mathrm{KL}}(q(\boldsymbol{Y}) q(\boldsymbol{\alpha}) q(\beta) \| p(\boldsymbol{Y}, \boldsymbol{\alpha}, \beta|\mathbf{X})) + F_{\mathbf{X}}(\mathbf{W}) \\ &= D_{\mathrm{KL}}(q(\boldsymbol{Y}) q(\boldsymbol{\alpha}) q(\beta) \| p(\boldsymbol{Y}|\boldsymbol{\alpha}) p(\boldsymbol{\alpha}) p(\beta)) \\ &\quad - \sum_{n=1}^{N} \mathbb{E}_{\bar{q}'} \left[\log p_{\mathbf{W}}(\boldsymbol{X}^n|\boldsymbol{y}_n, \boldsymbol{h}_n, \beta) \right] \\ &\equiv \bar{F}_{\mathbf{X}}(q, \mathbf{W}), \end{aligned} \tag{4}$$

where $D_{\mathrm{KL}}(\cdot\|\cdot)$ is the Kullback−Leibler divergence and $\mathbb{E}_{\bar{q}}[\cdot] = \mathbb{E}_{q(\boldsymbol{Y})q(\boldsymbol{\alpha})q(\beta)}[\cdot], \mathbb{E}_{\bar{q}'}[\cdot] = \mathbb{E}_{q(\boldsymbol{y}_n)q(\beta)}[\cdot]$. The upper bound $\bar{F}_{\mathbf{X}}(q, \mathbf{W})$ is called the variational free energy or (negated) evidence lower bound (ELBO). The variational free energy is minimized with respect to the variational posterior $q(\boldsymbol{Y}, \boldsymbol{\alpha}, \beta) = q(\boldsymbol{Y}) q(\boldsymbol{\alpha}) q(\beta)$ using the variational Bayes method under the fixed parameters $\mathbf{W}$. Furthermore, it is also minimized with respect to the parameters $\mathbf{W}$ by applying the RNN learning algorithm to the second term of $\bar{F}_{\mathbf{X}}(q, \mathbf{W})$,

$$\begin{aligned} &- \sum_{n=1}^{N} \mathbb{E}_{q(\boldsymbol{y}_n)q(\beta)} \left[\log p_{\mathbf{W}}(\boldsymbol{X}^n|\boldsymbol{y}_n, \boldsymbol{h}_n, \beta) \right] \propto \\ &\quad \sum_{n=1}^{N} \mathbb{E}_{q(\boldsymbol{y}_n)} \left[\sum_{k=1}^{K} y_{nk} \|\boldsymbol{X}^n - g(\boldsymbol{h}_n|\mathbf{W}_{dec}^k)\|_F^2 \right] + const.. \end{aligned} \tag{5}$$

B. Minimization of the Variational Free Energy

In this section, we derive the variational Bayes algorithm for MDRA to minimize the variational free energy. We show the outline of the derivation below (for a detailed derivation, see Appendix A and B). The general formula of the variational Bayes algorithm is given by

$$\log q(\boldsymbol{Y}) = \mathbb{E}_{q(\boldsymbol{\alpha}, \beta)}[\log p_{\mathbf{W}}(\mathbf{X}, \boldsymbol{Y}, \boldsymbol{H}, \boldsymbol{\alpha}, \beta)] + const.,$$

$$\log q(\boldsymbol{\alpha}, \beta) = \mathbb{E}_{q(\boldsymbol{Y})}[\log p_{\mathbf{W}}(\mathbf{X}, \boldsymbol{Y}, \boldsymbol{H}, \boldsymbol{\alpha}, \beta)] + const..$$

By applying the above equations to the above probabilistic models (see Appendix A), we obtained the specific algorithm shown in Algorithm 1. Then we minimize the following weighted reconstruction error using RNN algorithm:

$$\sum_{n=1}^{N} \sum_{k=1}^{K} \left\{ r_{nk} \|\boldsymbol{X}^n - g(\boldsymbol{h}_n|\mathbf{W}_{dec}^k)\|_F^2 \right\}, \tag{6}$$

where $r_{nk} = \mathbb{E}_{q(\boldsymbol{y}_n)}[y_{nk}]$ as detailed in Appendix B. We denote $\boldsymbol{R} = (\boldsymbol{r}_1, \cdots, \boldsymbol{r}_N)$, where $\boldsymbol{r}_n = (r_{n1}, \cdots, r_{nK})^{\mathrm{T}} \in \mathbb{R}^K$. From the above discussion, we finally obtained the following Algorithm 2. We apply these two algorithms iteratively to minimize $\bar{F}_{\mathbf{X}}(q, \mathbf{W})$. Fig. 4 describes the relation of the VB and RNN steps of MDRA algorithm.

Algorithm 1 VB part of MDRA

Input: $\mathbf{X}$: set of input signals
Output: $\boldsymbol{R}$: allocation weights
for $i \leftarrow 0$ to I **do**
VB E-step:

$$\log \rho_{nk} = \psi(\bar{\theta}_k) - \psi\left(\sum_{k=1}^{K} \bar{\theta}_k\right) - \|\boldsymbol{X}^n - g(\boldsymbol{h}_n|\mathbf{W}_{dec}^k)\|_F^2 \bar{\nu}\bar{\lambda}^{-1} + \frac{T_n D}{2}(\psi(\bar{\nu}) - \log \bar{\lambda}) - \frac{T_n D}{2}\log \pi, \quad r_{nk} = \frac{\rho_{nk}}{\sum_{k=1}^{K} \rho_{nk}}$$

VB M-step:

$$N_k = \sum_{n=1}^{N} r_{nk}, \quad \bar{\theta}_k = \theta_0 + N_k, \quad \bar{\nu} = \nu_0 + \frac{1}{2}\sum_{n=1}^{N} T_n D$$

$$\bar{\lambda} = \lambda_0 + \sum_{k=1}^{K}\sum_{n=1}^{N} r_{nk}\|\boldsymbol{X}^n - g(\boldsymbol{h}_n|\mathbf{W}_{dec}^k)\|_F^2$$

end for

Algorithm 2 MDRA

Input: $\mathbf{X}$: set of input signals
Output: $\mathbf{W}$: weight tensors, $\boldsymbol{R}$: allocation weights, $\boldsymbol{H}$: encoded signals
Set hyperparameters $\theta_0, \nu_0, \lambda_0$ and the initial value of $\mathbf{W}$ randomly.
repeat
Calculate $\mathbf{W}$ that minimizes the following value by RNN algorithm:

$$\sum_{n=1}^{N}\sum_{k=1}^{K}\left\{r_{nk}\|\boldsymbol{X}^n - g(\boldsymbol{h}_n|\mathbf{W}_{dec}^k)\|_F^2\right\}.$$

Calculate $\boldsymbol{R} = (r_{nk})$ by the algorithm *VB part of MDRA* (Algorithm 1).
until the difference of variational free energy $\bar{F}_{\mathbf{X}}(q, \mathbf{W}) <$ Threshold

V. Experiments

A. Periodic Signals

We first examined the basic performance of our algorithm using periodic signals. Periodic signals are typical time series signals expressed by dynamical systems. Input signals have 2, 4, and 8 periods respectively in 64 steps. Each signal is added a phase shift (maximum one period), amplitude variation (from 50% to 100% of the maximum amplitude), additional noise (maximum 2% of maximum amplitude) and signal length variation (maximum 80% of the maximum signal length). Examples of input data are illustrated in Fig. 5.

We compared LSTM-AE and RNN-AE to MDRA on its feature extraction performance using the above periodic signals. Fig. 6 and Fig. 7 show the results of LSTM-AE, RNN-AE and MDRA, respectively. We set the same dimension of

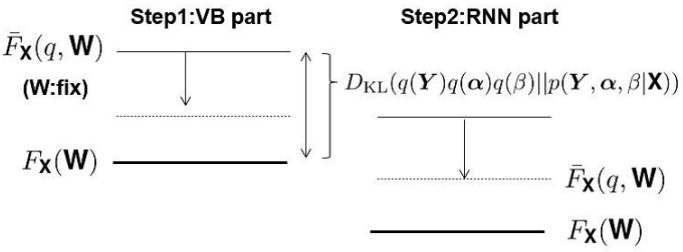

Fig. 4. MDRA algorithm

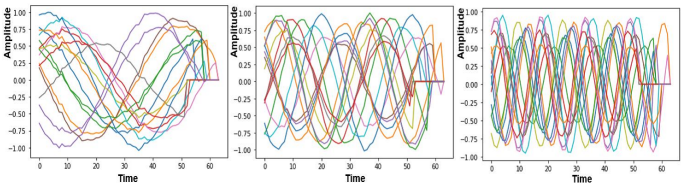

Fig. 5. Examples of periodic signals

hidden variable $\boldsymbol{h}_n$ in all algorithms. Note here that RNN-AE and MDRA use a complex-valued hidden variable while LSTM-AE uses real-valued one. Therefore LSTM-AE has twice the hidden variable dimension of RNN-AE and MDRA. The parameter setting is listed in Table II in Appendix D.

We used multi-dimensional scaling (MDS) as the dimension reduction method to visualize the distributions of features in Fig. 6 and Fig. 7.

Fig. 6 shows the distribution of the encoded data $\boldsymbol{h}_n$ which is the initial value of the decoder unit in Fig. 2.

We found that RNN-AE can separate the input data into three regions corresponding to each frequency (Fig. 6:right). However distribution on the hidden variable of LSTM-AE has complicated shape, each frequency overlapped each other. We guess this result was caused by the complex architecture of LSTM cell.

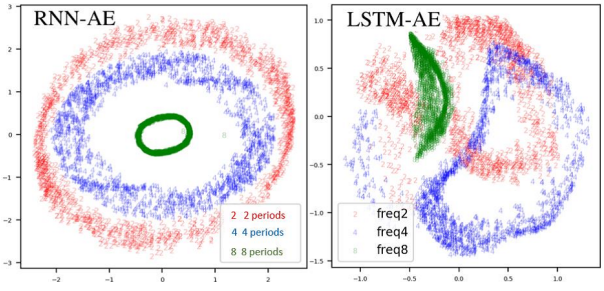

Fig. 6. Visualization of features extracted by RNN-AE: left, LSTM-AE: right

Fig. 7 shows the distributions of the encoded data $\boldsymbol{h}_n$ and the clustering allocation weight $\boldsymbol{r}_n$ extracted by MDRA. The distribution of $\boldsymbol{r}_n$ shown in the left figure of Fig. 7 is completely separated into each frequency component without overlap. The distribution of $\boldsymbol{h}_n$ was given as the initial value

of the corresponding decoder. This result shows that the distribution of $\boldsymbol{r}_n$ as the feature extraction has robustness for phase shift, amplitude and signal length variation.

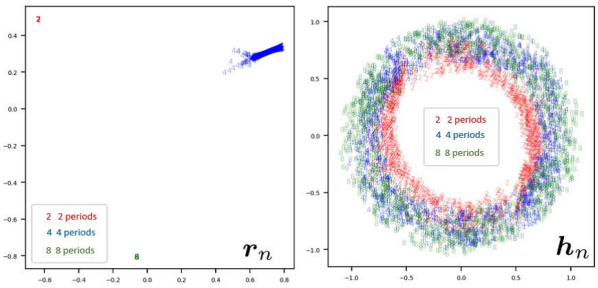

Fig. 7. Visualization of features extracted by MDRA (left: $\boldsymbol{r}_n$, right: $\boldsymbol{h}_n$)

B. Complex Periodic Signals

Next we applied our algorithm to more complicated signals. The input signals were all length 32 steps and created by the following steps.

1) Give $\theta_i \in [0, 2\pi], i = 1, 2$ randomly.
2) Set $\boldsymbol{h}_n^0 = (e^{i\theta_1}, e^{i\theta_2})$.
3) Create $\boldsymbol{h}_n^t \in \mathbb{C}^2$ by the rule $\boldsymbol{h}_n^{t+1} = \begin{pmatrix} e^{i\omega_1} & 0 \\ 0 & e^{i\omega_2} \end{pmatrix} \boldsymbol{h}_n^t, \quad (t = 1, 2, \cdots, 31)$.
4) Obtain the signal x_n^t by projecting $\boldsymbol{h}_n^t$ to the vector $(1, 1, 1, 1)$ as the real value vector $\boldsymbol{h}_n^t \in \mathbb{R}^4$

We created two types of signals (5000 for each type) with A:$(\omega_1, \omega_2) = (55.0, 20.0)$ and B:$(\omega_1, \omega_2) = (50.0, 25.0)$, respectively. We found that it is not very easy to separate the two types of signals from Fig. 8 visually.

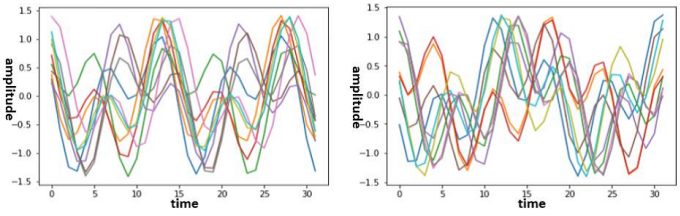

Fig. 8. Examples of complex periodic signals

Fig. 9 shows the result of each algorithm applied to the complex periodic signals. We used the same hidden variable dimension for all algorithms. Further information on the parameters are listed in Table III in Appendix D. Unlike the experiment V-A, although RNN-AE could not separate the two types of signals completely, LSTM-AE was able to separate them. Furthermore $\boldsymbol{r}_n$ of MDRA classified the signals based on the periodicity without any influence from the phase shift. The phase shift was expressed by $\boldsymbol{h}_n$ similarly to the experiment V-A.

In this experiment, the MDRA estimated the number of clusters and data ratios correctly in spite of the setting of the

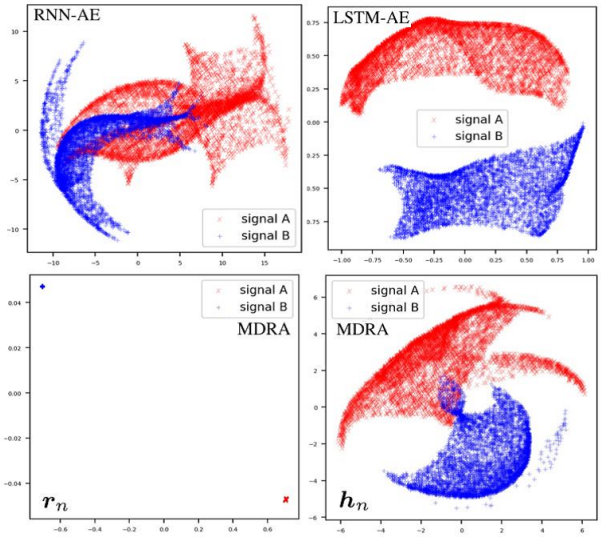

Fig. 9. Visualization of features extracted by RNN-AE: top left, LSTM-AE: top right and MDRA: bottom left and right (complex periodic signals)

number of decoders $K = 5$. The distribution ratios calculated from $\boldsymbol{r}_n$ for 2 major clusters were 49.8% and 49.0%. Fig. 10:left is the MDS expression of hidden variable trajectories. Fig. 10:right shows the first eight signals with successive data connected by lines. We found that these two types of signals were completely expressed as different periodic signals in the hidden space.

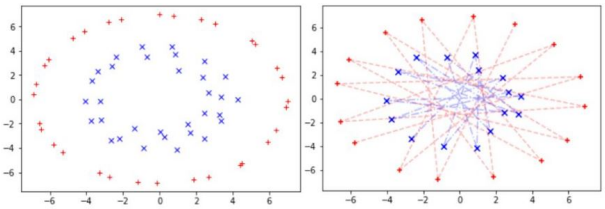

Fig. 10. Trajectory of hidden variable of MDRA (signal A: red, signal B: blue)

C. Experiment of Real Driving Data

We applied our algorithm to a real driving data clustering problem. We use the driving data consisting of speed, acceleration, braking and steering angle signals.[1] The input signal was about 1 minute differential data, which was cut out from the original data by a sliding window.[2] The detailed information of the input data is shown in Table I.

The feature extraction results by MDRA are shown in Fig. 11. The parameter setting of this experiment is listed in Table IV in Appendix D. The left figure is the route clustering result based on the driving behavior by the MDRA ($K = 10$).

[1]This data was created by HQL (Research Institute of Human Engineering for Quality Life: https:// www.hql.jp/ howhql/ spirit.html).

[2]We use only the data of which the maximum acceleration difference is more than a certain threshold.

TABLE I
DRIVING DATA CLUSTERING

#Training	Signal length	Sampling pitch	Slide
4644	512	0.1 sec.	8

This figure shows the actual trajectory of a driven car, each point of which is colored by RGB based on 3 dimensional representation of $\boldsymbol{r}_n$ given by the MDS. The right figures (No.1-No.4) show the typical driving behavior extracted from the major clusters. Blue, green, orange and red lines are speed, acceleration, brake and steering angle, respectively. From these results, the interpreted driving feature of each cluster and its ratio are as follows:

- No.1: moderate acceleration 14.0%
- No.2: stable travel (high speed) 7.5%
- No.3: moderate deceleration 3.5%
- No.4: stable travel (middle speed) 13.7%

In addition, we can extract the complicated driving operation such as No.5 by choosing the low ratio data point which is significantly different from the surrounding data points.

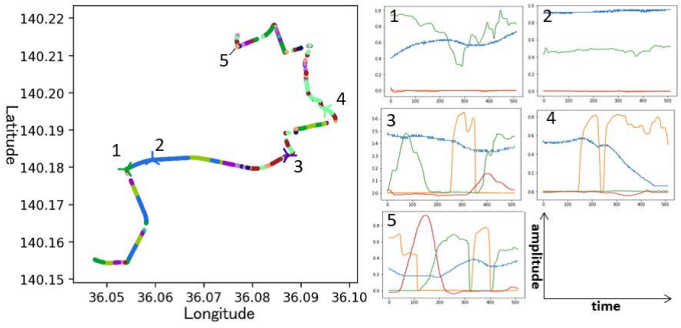

Fig. 11. Clustering result of driving data ($K = 10$)

Although we showed the result in the case of $K = 10$ here, we can adjust the clustering size by changing K and the hyparparameters.

VI. DISCUSSION

We verified the feature extraction performance of the MDRA using actual time series data. In Sections V-A and V-B, we saw that MDRA algorithm can achieve more stable clustering than LSTM-AE and RNN-AE by using decoder weight $\boldsymbol{r}_n$ for periodic and complex periodic data. In addition, we also showed that MDRA has the function to reduce the unnecessary clusters using the property of the variational Bayes method. In Section V-C, we confirmed that above variational Bayes property provides the flexible clustering and uncommon data extraction using the actual driving data. There are a lot of research on the variational Bayes method [31], therefore we can apply these algorithms and knowledges to improve the performance of MDRA. Especially the phase transition phenomenon of the variational Bayes learning method, depending on the hyperparameters, has been reported in [32]. The hyperparameter setting of the prior distribution has a great effect on the clustering result.

VII. CONCLUSION

In this paper, we proposed a new clustering algorithm, MDRA, which can extract features of time series data based on the data generating process expressed by decoders. We conducted experiments using periodic signals and actual driving data to verify the advantages of MDRA. The results show that our algorithm has not only robustness for the phase shift, amplitude, signal length variation, and signal synthesis but also flexibility on the clustering performance. We intend to undertake a detailed study of the relation between the feature extraction performance and hyperparameter setting of the prior distributions in the future.

REFERENCES

[1] L. Rabiner and B.-H. Juang, *Fundamentals of Speech Recognition*, Prentice Hall, Upper Saddle River, NJ, USA, 1993.

[2] J. L. Elman, "Finding structure in time," *Cognitive Science*, vol. 14, no. 2, pp. 179–211, 1990.

[3] J. L. Elman, "Distributed representations, simple recurrent networks, and grammatical structure," *Machine Learning*, vol. 7, no. 2, pp. 195–225, 1991.

[4] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[5] P. Vincent, H. Larochelle, Y. Bengio, and P.-A. Manzagol, "Extracting and composing robust features with denoising autoencoders," in *Proceedings of the 25th International Conference on Machine Learning (ICML)*, 2008, pp. 1096–1103.

[6] S. Rifai, P. Vincent, X. Muller, X. Glorot, and Y. Bengio, "Contractive auto-encoders: Explicit invariance during feature extraction," in *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 2011, pp. 833–840.

[7] N. Srivastava, E. Mansimov, and R. Salakhudinov, "Unsupervised learning of video representations using lstms," in *Proceedings of the 32nd International Conference on Machine Learning (ICML)*, 2015, pp. 843–852.

[8] W. Dong, T. Yuan, K. Yang, C. Li, and S. Zhang, "Autoencoder regularized network for driving style representation learning," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI)*, 2017, pp. 1603–1609.

[9] A. Zerroug, L. S. Terrissa, and A. Faure, "Chaotic dynamical behavior of recurrent neural network," *Annual Review of Chaos Theory, Bifurcations and Dynamical Systems*, vol. 4, pp. 55–56, 2013.

[10] T. Laurent and J. H. von Brecht, "A recurrent neural network without chaos," *CoRR*, vol. abs/1612.06212, 2016.

[11] H. Attias, "Inferring parameters and structure of latent variable models by variational Bayes," in *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence (UAI)*, 1999, pp. 21–30.

[12] Z. Ghahramani and M. J. Beal, "Graphical models and variational methods," in *Advanced Mean Field Methods Theory and Practice*, pp. 161–177. MIT Press, 2001.

[13] D. Kaji and S. Watanabe, "Two design methods of hyperparameters in variational Bayes learning for Bernoulli mixtures," *Neurocomputing*, vol. 74, no. 11, pp. 2002–2007, 2011.

[14] M. Tschannen, M. Lucic, and O. Bachem, "Recent advances in autoencoder-based representation learning," in *Proceedings of Workshop on Bayesian Deep Learning (NeurIPS)*, 2018.

[15] C. Song, F. Liu, Y. Huang, L. Wang, and T. Tan, "Auto-encoder based data clustering," in *Iberoamerican Congress on Pattern Recognition (CIARP)*, 2013, pp. 117–124.

[16] E. Pineau and M. Lelarge, "Infocatvae: Representation learning with categorical variational autoencoders," *CoRR*, vol. abs/1806.08240, 2018.

[17] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2014.

[18] P. Wilhelmssion, "Hierarchical clustering of times series using Gaussian mixture models and variational autoencoder," M.S. thesis, Lund Institute of technology, Sweden, 2019.
[19] W. Wang, D. Yang, F. Chen, Y. Pang, S. Huang, and Y. Ge, "Clustering with orthogonal autoencoder," *IEEE Access*, vol. 7, pp. 62421–62432, 2019.
[20] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," *Neural Computation*, vol. 12, no. 10, pp. 2451–2471, 2000.
[21] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.
[22] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using RNN encoder–decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2014, pp. 1724–1734.
[23] P. Malhotra, L. Vig, G. Shroff, and P. Agarwal, "Long short term memory networks for anomaly detection in time series," in *23rd European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*, 2015, pp. 89–94.
[24] R. Rana, "Gated recurrent unit (GRU) for emotion classification from noisy speech," *CoRR*, vol. abs/1612.07778, 2016.
[25] R. Pascanu, T. Mikolov, and Y. Bengio, "On the difficulty of training recurrent neural networks," in *Proceedings of the 30th International Conference on Machine Learning (ICML)*, 2013, pp. 1310–1318.
[26] L. Jing, Y. Shen, T. Dubcek, J. Peurifoy, S. Skirlo, Y. LeCun, M. Tegmark, and M. Soljačić, "Tunable efficient unitary neural networks (EUNN) and their application to RNNs," in *Proceedings of the 34th International Conference on Machine Learning (ICML)*, 2017, pp. 1733–1741.
[27] S. Wisdom, T. Powers, J. Hershey, J. Le Roux, and L. Atlas, "Full-capacity unitary recurrent neural networks," in *Advances in Neural Information Processing Systems 29 (NIPS)*, pp. 4880–4888. 2016.
[28] M. Arjovsky, A. Shah, and Y. Bengio, "Unitary evolution recurrent neural networks," in *Proceedings of The 33rd International Conference on Machine Learning (ICML)*, 2016, pp. 1120–1128.
[29] L. Jing, C. Gulcehre, J. Peurifoy, Y. Shen, M. Tegmark, M. Soljacic, and Y. Bengio, "Gated orthogonal recurrent units: On learning to forget," *Neural Computation*, vol. 31, no. 4, pp. 765–783, 2019.
[30] A. Corduneanu and C. Bishop, "Variational Bayesian model selection for mixture distributions," in *Proceedings of the Eighth International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2001.
[31] S. Nakajima, K. Watanabe, and M. Sugiyama, *Variational Bayesian Learning Theory*, Cambridge University Press, 2019.
[32] K. Watanabe and S. Watanabe, "Stochastic complexities of Gaussian mixtures in variational Bayesian approximation," *Journal of Machine Learning Research*, vol. 7, no. Apr, pp. 625–645, 2006.

APPENDIX

A. Minimization of Variational Free Energy with Respect to the Variational Posterior for the Fixed RNN Parameter

Initially, we suppose that the posterior is expressed by $q(\boldsymbol{Y}, \boldsymbol{\alpha}, \beta) = q(\boldsymbol{Y})q(\boldsymbol{\alpha}, \beta)$. Then

$$\begin{aligned}\log q(\boldsymbol{Y}) =& \mathbb{E}_{q(\boldsymbol{\alpha},\beta)}[\log p_{\mathsf{W}}(\mathbf{X}, \boldsymbol{Y}, \boldsymbol{H}, \boldsymbol{\alpha}, \beta)] + const.\\ =& \mathbb{E}_{q(\boldsymbol{\alpha},\beta)}[\log p_{\mathsf{W}}(\mathbf{X}|\boldsymbol{Y}, \boldsymbol{H}, \beta)] + \mathbb{E}_{q(\boldsymbol{\alpha},\beta)}[\log p(\boldsymbol{Y}|\boldsymbol{\alpha})]\\ &+ \mathbb{E}_{q(\boldsymbol{\alpha},\beta)}[\log p(\boldsymbol{\alpha})] + \mathbb{E}_{q(\boldsymbol{\alpha},\beta)}[\log p(\beta)] + const.\\ =& \mathbb{E}_{q(\boldsymbol{\alpha})}[\log p(\boldsymbol{Y}|\boldsymbol{\alpha})] + \mathbb{E}_{q(\boldsymbol{\alpha})}[\log p(\boldsymbol{\alpha})]\\ &+ \mathbb{E}_{q(\beta)}[\log p_{\mathsf{W}}(\mathbf{X}|\boldsymbol{Y}, \boldsymbol{H}, \beta)] + \mathbb{E}_{q(\beta)}[\log p(\beta)] + const..\end{aligned}$$

In addition,

$$\begin{aligned}\mathbb{E}_{q(\boldsymbol{\alpha})}[\log p(\boldsymbol{Y}|\boldsymbol{\alpha})] &= \mathbb{E}_{q(\boldsymbol{\alpha})}\left[\log \prod_{n=1}^{N}\prod_{k=1}^{K} \alpha_k^{y_{nk}}\right]\\ &= \sum_{n=1}^{N}\sum_{k=1}^{K} y_{nk}\mathbb{E}_{q(\boldsymbol{\alpha})}\left[\log \alpha_k\right],\end{aligned}$$

$$\begin{aligned}&\mathbb{E}_{q(\beta)}\left[\log p_{\mathsf{W}}(\mathbf{X}|\boldsymbol{Y}, \boldsymbol{H}, \beta)\right]\\ &= \sum_{n=1}^{N}\sum_{k=1}^{K} y_{nk}\mathbb{E}_{q(\beta)}\left[-\beta\|\boldsymbol{X}^n - g(\boldsymbol{h}_n|\mathbf{W}_{dec}^k)\|_F^2 + \frac{T_n D}{2}\log\frac{\beta}{\pi}\right],\end{aligned}$$

where $T_n D$ means total signal dimension. Therefore, we obtain

$$\log q(\boldsymbol{Y}) = \sum_{n=1}^{N}\sum_{k=1}^{K} y_{nk}\log\rho_{nk} + const..$$

We here put

$$\log\rho_{nk} = \mathbb{E}_{q(\boldsymbol{\alpha})}\left[\log\alpha_k\right] + \mathbb{E}_{q(\beta)}\left[G\right], \tag{7}$$

where $G = G' + \frac{T_n D}{2}(\log\beta - \log\pi), G' = -\beta\|\boldsymbol{X}^n - g(\boldsymbol{h}_n|\mathbf{W}_{dec}^k)\|_F^2$. Hence $q(\boldsymbol{Y}) \propto \prod_{n=1}^{N}\prod_{k=1}^{K}\rho_{nk}^{y_{nk}}$, by putting $r_{nk} = \frac{\rho_{nk}}{\sum_{k=1}^{K}\rho_{nk}}$, we obtain

$$q(\boldsymbol{Y}) = \prod_{n=1}^{N}\prod_{k=1}^{K} r_{nk}^{y_{nk}}.$$

Next we calculate $\log q(\boldsymbol{\alpha}, \beta)$,

$$\begin{aligned}\log q(\boldsymbol{\alpha}, \beta) &= \mathbb{E}_{q(\boldsymbol{Y})}[\log p_{\mathsf{W}}(\mathbf{X}, \boldsymbol{Y}, \boldsymbol{H}, \boldsymbol{\alpha}, \beta)] + const.\\ &= \mathbb{E}_{q(\boldsymbol{Y})}[\log p(\boldsymbol{Y}|\boldsymbol{\alpha})] + \log p(\boldsymbol{\alpha})\\ &\quad + \mathbb{E}_{q(\boldsymbol{Y})}[\log p_{\mathsf{W}}(\mathbf{X}|\boldsymbol{Y}, \boldsymbol{H}, \beta)] + \log p(\beta) + const..\end{aligned}$$

Above equation can be divided into the two terms including $\boldsymbol{\alpha}$ and β respectively,

$$\begin{aligned}&\log q(\boldsymbol{\alpha}) \propto \mathbb{E}_{q(\boldsymbol{Y})}[\log p(\boldsymbol{Y}|\boldsymbol{\alpha})] + \log p(\boldsymbol{\alpha}) + const.\\ &= \sum_{n=1}^{N}\sum_{k=1}^{K}\log\alpha_k\mathbb{E}_{q(\boldsymbol{y}_n)}\left[y_{nk}\right] + (\theta_0 - 1)\sum_{k=1}^{K}\log\alpha_k + const..\end{aligned}$$

Substituting $\mathbb{E}_{q(\boldsymbol{y}_n)}[y_{nk}] = 1\cdot q(y_{nk}=1) + 0\cdot q(y_{nk}=0) = q(y_{nk}=1) = r_{nk}$ to the above equation, we obtain

$$\log q(\boldsymbol{\alpha}) = \sum_{n=1}^{N}\sum_{k=1}^{K}\log\alpha_k r_{nk} + (\theta_0 - 1)\sum_{k=1}^{K}\log\alpha_k + const..$$

On the other hand,

$$\begin{aligned}&\log q(\beta) = \mathbb{E}_{q(\boldsymbol{Y})}[\log p_{\mathsf{W}}(\mathbf{X}|\boldsymbol{Y}, \boldsymbol{H}, \beta)] + \log p(\beta) + const.\\ &= \sum_{n=1}^{N}\sum_{k=1}^{K}\left[E_{q(\boldsymbol{y}_n)}\left[y_{nk}\right]\cdot G\right] + (\nu_0 - 1)\log\beta + \lambda\beta + const..\end{aligned}$$

By applying $\mathbb{E}_{q(\boldsymbol{y}_n)}[y_{nk}] = r_{nk}$, we obtain

$$\begin{aligned}&\log q(\beta)\\ &= \sum_{n=1}^{N}\sum_{k=1}^{K}\left[r_{nk}\cdot G\right] + (\nu_0 - 1)\log\beta + \lambda\beta + const..\end{aligned}$$

We finally calculate $\log\rho_{nk}$ in Eq. (7). We first calculate $\mathbb{E}_{q(\beta)}\left[G\right]$,

$$\log q(\beta) = \beta f + \left(\nu_0 + \frac{1}{2}\sum_{n=1}^{N} T_n D - 1\right)\log\beta - \lambda\beta + const.,$$

where we put $f = \sum_{k=1}^{K}\sum_{n=1}^{N} -r_{nk}\|\boldsymbol{X}^n - g(\boldsymbol{h}_n|\mathbf{W}_{dec}^k)\|_F^2$. In addition, putting $\bar{\lambda} = \lambda_0 - f$, $\bar{\nu} = \nu_0 + \frac{1}{2}\sum_{n=1}^{N} T_n D$,

$$q(\beta) = e^{\beta f}\beta^{\nu_0+\frac{1}{2}\sum_{n=1}^{N}T_nD-1}e^{-\lambda_0\beta}\cdot const. = e^{-\bar{\lambda}\beta}\beta^{\bar{\nu}-1}\cdot const. = \frac{\bar{\lambda}^{\bar{\nu}}}{\Gamma(\bar{\nu})}\beta^{\bar{\nu}-1}e^{-\bar{\lambda}\beta} = Gamma(\beta|\bar{\nu},\bar{\lambda}).$$

By using the expectations of β and $\log\beta$ by gamma distribution $\mathbb{E}_{q(\beta)}[\beta] = \nu\lambda^{-1}, \mathbb{E}_{q(\beta)}[\log\beta] = \psi(\nu) - \log\lambda$ (Appendix C), we obtain

$$\mathbb{E}_{q(\beta)}[G] = -\|\boldsymbol{X}^n - g(\boldsymbol{h}_n|\mathbf{W}_{dec}^k)\|_F^2\bar{\nu}\bar{\lambda}^{-1} + \frac{T_nD}{2}(\psi(\bar{\nu}) - \log\bar{\lambda}) - \frac{T_nD}{2}\log\pi.$$

Similarly, $q(\boldsymbol{\alpha})$ turns out to be the Dirichlet distribution with parameters $(\bar{\theta}_1, \cdots, \bar{\theta}_K)$, and $\mathbb{E}_{q(\boldsymbol{\alpha})}[\log\alpha_k] = \psi(\bar{\theta}_k) - \psi\left(\sum_{k=1}^{K}\bar{\theta}_k\right)$ is calculated by the same way in the general mixture model [11]–[13]. Therefore we finally obtain

$$\log\rho_{nk} = \psi(\bar{\theta}_k) - \psi\left(\sum_{k=1}^{K}\bar{\theta}_k\right) - \|\boldsymbol{X}^n - g(\boldsymbol{h}_n|\mathbf{W}_{dec}^k)\|_F^2\bar{\nu}\bar{\lambda}^{-1} + \frac{T_nD}{2}(\psi(\bar{\nu}) - \log\bar{\lambda}) - \frac{T_nD}{2}\log\pi.$$

From the above results, the following variational Bayes algorithm is derived.

B. Minimization of Variational Free Energy with Respect to the RNN Parameter for the Fixed Variational Posterior

We minimize

$$-\mathbb{E}_{q(\boldsymbol{Y})q(\beta)}\left[\sum_{n=1}^{N}\log p_{\mathbf{W}}(\boldsymbol{X}^n|\boldsymbol{y}_n, \boldsymbol{h}_n, \beta)\right]$$

to minimize the free energy Eq. (3) with respect to $\mathbf{W}$. More specifically, we minimize

$$\begin{aligned}
&-\sum_{n=1}^{N}\mathbb{E}_{q(\boldsymbol{y}_n)q(\beta)}\left[\log\left\{\prod_{k=1}^{K}\left\{\left(\frac{\beta}{\pi}\right)^{\frac{T_nD}{2}}e^{G'}\right\}^{y_{nk}}\right\}\right]\\
&= -\sum_{n=1}^{N}\mathbb{E}_{q(\boldsymbol{y}_n)q(\beta)}\left[\sum_{k=1}^{K}y_{nk}\left\{\frac{T_nD}{2}(\log\beta - \log\pi)G'\right\}\right]\\
&= \mathbb{E}_{q(\beta)}[\beta]\sum_{n=1}^{N}\sum_{k=1}^{K}r_{nk}\|\boldsymbol{X}^n - g(\boldsymbol{h}_n|\mathbf{W}_{dec}^k)\|_F^2\\
&\quad -\sum_{n=1}^{N}\sum_{k=1}^{K}r_{nk}\frac{T_nD}{2}(\mathbb{E}_{q(\beta)}[\log\beta] - \log\pi)\\
&\propto \sum_{n=1}^{N}\sum_{k=1}^{K}r_{nk}\|\boldsymbol{X}^n - g(\boldsymbol{h}_n|\mathbf{W}_{dec}^k)\|_F^2 + const.
\end{aligned}$$

where we used $r_{nk} = \mathbb{E}_{q(\boldsymbol{y}_n)}[y_{nk}]$.

We achieve this by applying RNN algorithm. From the above discussion including Appendix A, we obtain the MDRA algorithm.

C. Derivation of $\mathbb{E}_{Gamma(\beta|\nu,\lambda)}[\log\beta]$

By putting $\beta = e^x$, we obtain $x = \log\beta, d\beta = e^x dx$,

$$\begin{aligned}
\mathbb{E}_{Gamma(\beta|\nu,\lambda)}[\log\beta] &= \int_0^\infty \log\beta\frac{\lambda^\nu}{\Gamma(\nu)}\beta^{\nu-1}e^{-\lambda x}d\beta\\
&= \int x\frac{\lambda^\nu}{\Gamma(\nu)}(e^x)^{\nu-1}e^{-\lambda e^x}e^x dx\\
&= \int x\frac{\lambda^\nu}{\Gamma(\nu)}e^{x(\nu-1)}e^{-\lambda e^x}e^x dx\\
&= \int x\frac{\lambda^\nu}{\Gamma(\nu)}e^{x\nu-\lambda e^x}dx.
\end{aligned}$$

We here use

$$\frac{d}{d\nu}e^{x\nu-\lambda e^x} = xe^{x\nu-\lambda e^x},$$

then the above equation is

$$\begin{aligned}
\mathbb{E}_{Gamma(\beta|\nu,\lambda)}[\log\beta] &= \int\frac{\lambda^\nu}{\Gamma(\nu)}\frac{d}{d\nu}e^{x\nu-\lambda e^x}dx\\
&= \frac{\lambda^\nu}{\Gamma(\nu)}\frac{d}{d\nu}\int e^{x\nu-\lambda e^x}dx.
\end{aligned}$$

In addition, $\int_0^\infty x^{\nu-1}e^{-\lambda e^x}dx$ is the normalization constant of gamma distribution, therefore it equals to $\Gamma(\nu)/\lambda^\nu$. Hence we finally obtain

$$\begin{aligned}
\mathbb{E}_{Gamma(\beta|\nu,\lambda)}[\log\beta] &= \frac{\lambda^\nu}{\Gamma(\nu)}\frac{d}{d\nu}\frac{\Gamma(\nu)}{\lambda^\nu}\\
&= \frac{\lambda^\nu}{\Gamma(\nu)}\frac{\Gamma'(\nu)\lambda^\nu - \Gamma(\nu)\lambda^\nu\log\lambda}{\lambda^{2\nu}}\\
&= \psi(\nu) - \log\lambda.
\end{aligned}$$

D. Parameter Setting

In this section, we show the parameter setting of the experiments in Section V.

TABLE II
PARAMETER SETTING (PERIODIC SIGNALS)

	L	EUNN			VB			
		cap.	fft	cpx	K	θ_0	ν_0	λ_0
RNN-AE	4	8	T	F	-	-	-	-
MDRA	4				5	0.5	1.0	0.01
LSTM-AE	8	-	-	-	-	-	-	-

TABLE III
PARAMETER SETTING (COMPLEX PERIODIC SIGNALS)

	L	EUNN			VB			
		cap.	fft	cpx	K	θ_0	ν_0	λ_0
MDRA	4	8	T	F	5	1.0	1.0	0.01

TABLE IV
PARAMETER SETTING (ROUTE CLUSTERING)

	L	EUNN			VB			
		cap.	fft	cpx	K	θ_0	ν_0	λ_0
MDRA	4	8	T	F	10	10.0	1.0	5.0

Here L is the dimension of hidden variable $\boldsymbol{h}$, capacity, fft and cpx are parameters of EUNN [26], K is the number of the decoders, $\theta_0, \nu_0, \lambda_0$ are hyperparameters of prior distributions.