Dynamic Gaussian Mixture based Deep Generative Model For Robust Forecasting on Sparse Multivariate Time Series

Yinjun Wu[1*], **Jingchao Ni**[2‡], **Wei Cheng**[2], **Bo Zong**[2], **Dongjin Song**[3], **Zhengzhang Chen**[2], **Yanchi Liu**[2], **Xuchao Zhang**[2], **Haifeng Chen**[2], **Susan B. Davidson**[1]

[1]University of Pennsylvania
[2]NEC Laboratories America
[3]University of Connecticut
[1]{wuyinjun@seas.upenn.edu, susan@cis.upenn.edu}
[2]{jni, weicheng, bzong, zchen, yanchi, xuczhang, haifeng}@nec-labs.com, [3]dongjin.song@uconn.edu

Abstract

Forecasting on sparse multivariate time series (MTS) aims to model the predictors of future values of time series given their incomplete past, which is important for many emerging applications. However, most existing methods process MTS's individually, and do not leverage the dynamic distributions underlying the MTS's, leading to sub-optimal results when the sparsity is high. To address this challenge, we propose a novel generative model, which tracks the transition of latent clusters, instead of isolated feature representations, to achieve robust modeling. It is characterized by a newly designed dynamic Gaussian mixture distribution, which captures the dynamics of clustering structures, and is used for emitting time series. The generative model is parameterized by neural networks. A structured inference network is also designed for enabling inductive analysis. A gating mechanism is further introduced to dynamically tune the Gaussian mixture distributions. Extensive experimental results on a variety of real-life datasets demonstrate the effectiveness of our method.

Introduction

Multivariate time series (MTS) analysis is heavily used in a variety of applications, such as cyber-physical system monitoring (Zhang et al. 2019), financial forecasting (Binkowski, Marti, and Donnat 2018), traffic analysis (Li et al. 2018), and clinical diagnosis (Che et al. 2018a). Recent advances in deep learning have spurred on many innovative machine learning models on MTS data, which have shown remarkable results on a number of fundamental tasks, including forecasting (Qin et al. 2017), event prediction (Choi et al. 2016), and anomaly detection (Zhang et al. 2019). Despite these successes, most existing models treat the input MTS as homogeneous and complete sequences. In many emerging applications, however, MTS signals are integrated from heterogeneous sources and are very sparse.

For example, consider MTS signals collected for dialysis patients. Dialysis is an important renal replacement therapy for purifying the blood of patients whose kidneys are not working normally (Inaguma et al. 2019). Dialysis patients have routines of dialysis, blood tests, chest X-ray, etc., which record data such as venous pressure, glucose level, and cardiothoracic ratio (CTR). These signal sources may have different frequencies. For instance, blood tests and CTR are often evaluated less frequently than dialysis. Different sources may not be aligned in time and what makes things worse is that some sources may be irregularly sampled, and missing entries may present. Despite such discrepancies, different signals give complementary views on a patient's physical condition, and therefore are all important to the diagnostic analysis. However, simply combining the signals will induce highly sparse MTS data. Similar scenarios are also found in other domains: In finance, time series from financial news, stock markets, and investment banks are collected at asynchronous time steps, but are correlated strongly (Binkowski, Marti, and Donnat 2018). In large-scale complex monitoring systems, sensors of multiple sub-components may have different running environments, thus continuously emitting asynchronous time series that may still be related (Safari, Shabani, and Simon 2014).

The sparsity of MTS signals when integrated from heterogeneous sources presents several challenges. In particular, it complicates temporal dependencies and prevents popular models, such as recurrent neural networks (RNNs), from being directly used. The most common way to handle sparsity is to first impute missing values, and then make predictions on the imputed MTS. However, as discussed in (Che et al. 2018a), this two-step approach fails to account for the relationship between missing patterns and predictive tasks, lead-

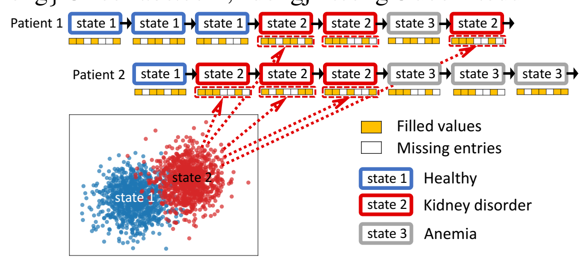

Figure 1: An illustration of latent structures underlying the sparse MTS of two dialysis patients. The vector below each state is a temporal feature generated from some distribution.

[*]This work was done when the first author was an intern at NEC Laboratories America. [‡]Corresponding author.

ing to sub-optimal results when the sparsity is severe.

Recently, some end-to-end models have been proposed. One approach is to consider missing time steps as intervals, and design RNNs with continuous dynamics via functional decays between observed time steps (Cao et al. 2018; Rubanova, Chen, and Duvenaud 2019). Another approach is to parameterize all missed entries and jointly train the parameters with predictive models, so that the missing patterns are learned to fit downstream tasks (Che et al. 2018a; Shukla and Marlin 2019; Tang et al. 2020). However, these methods have the drawback that MTS samples are assessed individually. Latent relational structures that are shared by different MTS samples are seldom explored for robust modeling.

In many applications, MTS's are not independent, but are related by hidden structures. Fig. 1 shows an example of two dialysis patients. Throughout the course of treatments, each patient may experience different latent states, such as kidney disorder and anemia, which are externalized by time series, such as glucose, albumin, and platelet levels. If two patients have similar pathological conditions, some of their data may be generated from similar state patterns, and could form clustering structures. Thus, inferring latent states and modeling their dynamics are promising for leveraging the complementary information in clusters, which can alleviate the issue of sparsity. This idea is not limited to the medical domain. For example, in meteorology, nearby observing stations that monitor climate may experience similar weather conditions (*i.e.*, latent states), which govern the generation of metrics, such as temperature and precipitation, over time. Although promising, inferring the latent clustering structures while modeling the dynamics underlying sparse MTS data is a challenging problem.

To address this problem, we propose a novel Dynamic Gaussian Mixture based Deep Generative Model (DGM^2). DGM^2 has a state space model under a non-linear transition-emission framework. For each MTS, it models the transition of latent cluster variables, instead of isolated feature representations, where all transition distributions are parameterized by neural networks. DGM^2 is characterized by its emission step, where a dynamic Gaussian mixture distribution is proposed to capture the dynamics of clustering structures. For inductive analysis, we resort to variational inferences, and develop structured inference networks to approximate posterior distributions. To ensure reliable inferences, we also adopt the paradigm of parametric pre-imputation, and link a pre-imputation layer ahead of the inference networks. The DGM^2 model is carefully designed to handle discrete variables and is constructed to be end-to-end trainable. Our contributions are summarized as follows:

- We investigate the problem of sparse MTS forecasting by modeling the latent dynamic clustering structures.
- We propose DGM^2, a novel deep generative model that leverages the transition of latent clusters and the emission from dynamic Gaussian mixture for robust forecasting.
- We perform extensive experiments on real-life datasets to validate the effectiveness of our proposed method.

Related Work

To the best of our knowledge, this is the first work to exploit latent clustering structures via dynamic Gaussian mixture distributions for robust forecasting on sparse MTS.

Traditional forecasting methods are mainly developed for homogeneously complete MTS data, such as autoregression, ARIMA, and boosting trees (Chen and Guestrin 2016). Recently, to tackle non-linear temporal dynamics, various deep learning models have been proposed (Shi et al. 2015; Qin et al. 2017). These methods, however, are not designed to handle the challenges of highly sparse MTS. Their applicability relies heavily on pre-processing steps such as statistical imputation (e.g., mean imputation) (Che et al. 2018a), kernel based methods (Rehfeld et al. 2011), matrix completion (Koren, Bell, and Volinsky 2009), multivariate imputation by chained equations (Azur et al. 2011), and recent GAN based methods (Luo et al. 2018). Such a two-step approach neglects the sparsity patterns that could be aligned with the downstream tasks, thus often leading to sub-optimal solutions on highly sparse MTS (Che et al. 2018a).

A more reasonable way is to apply end-to-end training methods on sparse MTS, which can be divided into two categories. The first is to transform the sparsity to time gaps between observations, which are integrated into the predictive models via (1) being explicit parts of the input features (Lipton, Kale, and Wetzel 2016; Binkowski, Marti, and Donnat 2018), and (2) decaying the hidden states by exponential functions (Baytas et al. 2017; Che et al. 2018a), or solving ordinary differential equations (ODEs) (Rubanova, Chen, and Duvenaud 2019; De Brouwer et al. 2019). The second category trains a joint model for concurrent imputation and forecasting, so that task-aware missing patterns can be learned from the back-propagated errors. For example, Che et al. (2018a) and Tang et al. (2020) used a trainable decay mechanism for approximating missing values, Cao et al. (2018) regarded unobserved entries as variables of a bidirectional RNN graph, Shukla and Marlin (2019) exerted several kernel-based intensity functions to parameterize missing variables. In addition to these methods, Che et al. (2018b) also studied a similar problem of modeling multi-rate MTS. However, none of the above methods explores the dynamic clustering structures underlying a batch of MTS samples for robust forecasting.

Vanilla Gaussian mixture (GM) model does not suit dynamic scenario. There are some works applying it on speech recognition (Tüske et al. 2015a,b; Variani, McDermott, and Heigold 2015; Zhang and Woodland 2017), which model the transition of words, but keep conditional GM distributions independent and static. Diaz-Rozo, Bielza, and Larrañaga (2018) studied data streams of IoT systems, where a static GM model was continuously retrained to fit new data. In contrast to these methods, our model explicitly defines dynamic GM distributions with temporal dependencies, and is inductive and end-to-end trainable. Some dynamic topic models (Wei, Sun, and Wang 2007; Zaheer, Ahmed, and Smola 2017) aim to unveil the flow of topics in documents. These methods, however, are neither Gaussian nor inductive, thus unable to be applied to solve the investigated problem.

Problem Statement

As suggested by the joint imputation-prediction framework (Che et al. 2018a; Shukla and Marlin 2019), a sparse MTS sample can be represented with missing entries against a set of evenly spaced reference time points $t = 1, ..., w$.

Let $\mathbf{x}_{1:w} = (\mathbf{x}_1, ..., \mathbf{x}_w) \in \mathbb{R}^{d \times w}$ be a length-w MTS recorded from time steps 1 to w, where $\mathbf{x}_t = (x_t^1, ..., x_t^d)^\top \in \mathbb{R}^d$ is a *temporal feature* vector at the t-th time step, x_t^i is the i-th variable of $\mathbf{x}_t$, and d is the total number of variables. To mark observation times, we employ a binary mask $\mathbf{m}_{1:w} = (\mathbf{m}_1, \mathbf{m}_2, ..., \mathbf{m}_w) \in \{0, 1\}^{d \times w}$, where $m_t^i = 1$ indicates x_t^i is an observed entry; $m_t^i = 0$ otherwise, with a corresponding placeholder $x_t^i = \texttt{NaN}$.

In this work, we are interested in a sparse MTS forecasting problem, which is to estimate the most likely length-r sequence in the future given the *incomplete* observations in past w time steps, *i.e.*, we aim to obtain

$$\tilde{\mathbf{x}}_{w+1:w+r} = \arg\max_{\mathbf{x}_{w+1:w+r}} p(\mathbf{x}_{w+1:w+r} | \mathbf{x}_{1:w}, \mathbf{m}_{1:w}) \quad (1)$$

where $\tilde{\mathbf{x}}_{w+1:w+r} = (\tilde{\mathbf{x}}_{w+1}, ..., \tilde{\mathbf{x}}_{w+r})$ are predicted estimates, and $p(\cdot|\cdot)$ is a forecasting function to be learned.

Our Proposed Model

In this section, we introduce our DGM^2 model. Inspired by the successful paradigm of joint imputation and prediction, we design DGM^2 to have a *pre-imputation layer* for capturing (1) the temporal intensity, and (2) the multi-dimensional correlations present in every MTS, for parameterizing missing entries. The parameterized MTS is fed to a *forecasting component*, which has a deep generative model that estimates the latent dynamic distributions for robust forecasting.

Pre-Imputation Layer

This layer aims to estimate the missing entries by leveraging the smooth trends and temporal intensities of the observed parts, which can help alleviate the impacts of sparsity in the downstream predictive tasks.

Similar to (Shukla and Marlin 2019), for the i-th variable at the t^*-th reference time point, we use a Gaussian kernel $\kappa(t^*, t; \alpha_i) = e^{-\alpha_i (t^* - t)^2}$ to evaluate the temporal influence of any time step t ($1 \le t \le w$) on t^*, where α_i is a parameter to be learned. Based on the kernel, we then employ a weighted aggregation for estimating $x_{t^*}^i$ by

$$\bar{x}_{t^*}^i = \frac{1}{\lambda(t^*, \mathbf{m}^i; \alpha_i)} \sum\nolimits_{t=1}^{w} \kappa(t^*, t; \alpha_i) m_t^i x_t^i \quad (2)$$

where $\mathbf{m}^i = (m_1^i, ..., m_w^i)^\top \in \mathbb{R}^w$ is the mask of the i-th variable, and $\lambda(t^*, \mathbf{m}^i; \alpha_i) = \sum_{t=1}^{w} m_t^i \kappa(t^*, t; \alpha_i)$ is an intensity function that evaluates the observation density at t^*, in which m_t^i is used to zero out unobserved time steps.

To account for the correlations of different variables, we also merge the information across d variables by introducing learnable correlation coefficients ρ_{ij} for $i, j = 1, ..., d$, and formulating a parameterized output if $x_{t^*}^i$ is unobserved.

$$\hat{x}_{t^*}^i = \left[\sum\nolimits_{j=1}^{d} \rho_{ij} \lambda(t^*, \mathbf{m}^i; \alpha_j) \bar{x}_{t^*}^j \right] / \sum_{j'=1}^{d} \lambda(t^*, \mathbf{m}^i; \alpha_{j'}) \quad (3)$$

where ρ_{ij} is set as 1 for $i = j$, and $\lambda(t^*, \mathbf{m}^i; \alpha_j)$ is introduced to indicate the reliability of $\bar{x}_{t^*}^j$, because larger $\lambda(t^*, \mathbf{m}^i; \alpha_j)$ implies more observations near $\bar{x}_{t^*}^j$.

In this layer, the set of parameters are $\boldsymbol{\alpha} = [\alpha_1, ..., \alpha_d]$, and $\boldsymbol{\rho} = [\rho_{ij}]_{i,j=1}^{d} \in \mathbb{R}^{d \times d}$. DGM^2 trains them jointly with its generative model for aligning missing patterns with the forecasting tasks.

Forecasting Component

Next, we design a generative model that captures the latent dynamic clustering structures for robust forecasting.

Suppose there are k latent clusters underlying all temporal features $\mathbf{x}_t$'s in a batch of MTS samples. For every time step t, we associate $\mathbf{x}_t$ with a *latent cluster variable* z_t to indicate to which cluster $\mathbf{x}_t$ belongs. Instead of the transition of $\mathbf{x}_t \to \mathbf{x}_{t+1}$, in this work, we propose to model the transition of the cluster variables $z_t \to z_{t+1}$. Since the clusters integrate the complementary information of similar features across MTS samples at different time steps, leveraging them is more robust than using individual sparse feature $\mathbf{x}_t$'s.

Generative Model. The generative process of our DGM^2 follows the *transition* and *emission* framework of state space models (Krishnan, Shalit, and Sontag 2017).

First, the transition process of DGM^2 employs a recurrent structure due to its effectiveness on modeling long-term temporal dependencies of sequential variables. Each time, the probability of a new state z_{t+1} is updated upon its previous states $z_{1:t} = (z_1, ..., z_t)$. We use a learnable function to define the transition probability, *i.e.*, $p(z_{t+1}|z_{1:t}) = f_{\boldsymbol{\theta}}(z_{1:t})$, where the function $f_{\boldsymbol{\theta}}(\cdot)$ is parameterized by $\boldsymbol{\theta}$, which can be variants of RNNs, for encoding non-linear dynamics that may be established between the latent variables.

For the emission process, we propose a *dynamic Gaussian mixture* distribution, which is defined by dynamically tuning a static basis mixture distribution. Let $\boldsymbol{\mu}_i$ ($i = 1, ..., k$) be the mean of the i-th mixture component of the basis distribution, and $p(\boldsymbol{\mu}_i)$ be its corresponding mixture probability. The emission (or forecasting) of a new feature $\mathbf{x}_{t+1}$ at time step $t + 1$ involves two steps: (1) draw a latent cluster variable z_{t+1} from a categorical distribution on all mixture components, and (2) draw $\mathbf{x}_{t+1}$ from the Gaussian distribution $\mathcal{N}(\boldsymbol{\mu}_{z_{t+1}}, \sigma^{-1}\mathbf{I})$, where σ is a hyperparameter, and $\mathbf{I}$ is an identity matrix. Here, we use isotropic Gaussian because of its efficiency and effectiveness in our experiments.

In step (1), the categorical distribution is usually defined on $p(\boldsymbol{\mu}) = [p(\boldsymbol{\mu}_1), ..., p(\boldsymbol{\mu}_k)] \in \mathbb{R}^k$, *i.e.*, the static mixture probabilities, which cannot reflect the dynamics in MTS. In light of this, and considering the fact that transition probability $p(z_{t+1}|z_{1:t})$ indicates to which cluster $\mathbf{x}_{t+1}$ belongs, we propose to dynamically adjust the mixture probability at each time step using $p(z_{t+1}|z_{1:t})$ by

$$\boldsymbol{\psi}_{t+1} = \underbrace{(1 - \gamma) p(z_{t+1}|z_{1:t})}_{\text{dynamic adjustment}} + \underbrace{\gamma p(\boldsymbol{\mu})}_{\text{basis mixture}} \quad (4)$$

where $\boldsymbol{\psi}_{t+1}$ is the dynamic mixture distribution at time step $t + 1$, and γ is a hyperparameter within $[0, 1]$ that controls the relative degree of change that deviates from the basis mixture distribution.

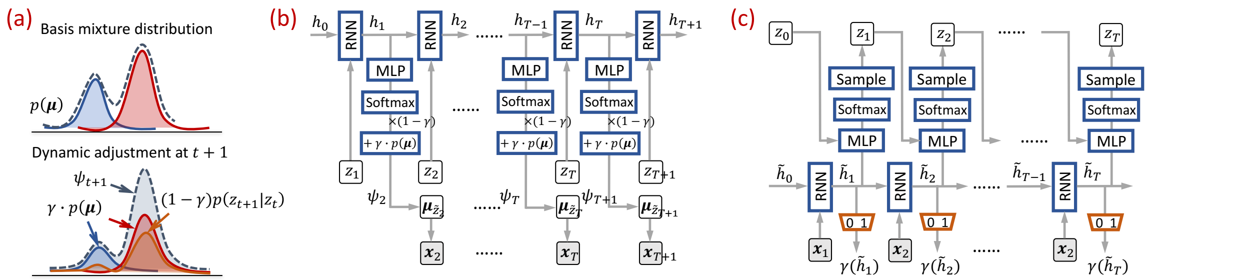

Figure 2: Illustrative examples of (a) the dynamic adjustment of the Gaussian mixture according to Eq. (4), with two mixture components, (b) the generative network, and (c) the inference network, where $\gamma(\cdot)$ is a gate function used in Eq. (4).

Fig. 2(a) illustrates the dynamic adjustment process of Eq. (4) on a Gaussian mixture with two components, where $p(z_{t+1}|z_{1:t})$ adjusts the mixture towards the component (*i.e.*, cluster) that $\mathbf{x}_{t+1}$ belongs to. It is noteworthy that adding the basis mixture in Eq. (4) is indispensable because it determines the relationships between different components, which regularizes the learning of the means $\boldsymbol{\mu} = [\boldsymbol{\mu}_1, ..., \boldsymbol{\mu}_k]$ during model training.

As such, our generative process can be summarized as

1. for each MTS sample:
   (a) draw $z_1 \sim \mathrm{Uniform}(k)$
   (b) for time step $t = 1, ..., w$:
      i. compute transition probability by $p(z_{t+1}|z_{1:t}) = f_{\boldsymbol{\theta}}(z_{1:t})$
      ii. draw $z_{t+1} \sim \mathrm{Categorial}(p(z_{t+1}|z_{1:t}))$ for transition
      iii. draw $\tilde{z}_{t+1} \sim \mathrm{Categorial}(\boldsymbol{\psi}_{t+1})$ using Eq. (4) for emission
      iv. draw a feature vector $\tilde{\mathbf{x}}_{t+1} \sim \mathcal{N}(\boldsymbol{\mu}_{\tilde{z}_{t+1}}, \sigma^{-1}\mathbf{I})$

where z_{t+1} (step ii) and $\tilde{z}_{t+1}$ (step iii) are different: z_{t+1} is used in transition (step i) for maintaining recurrent property; $\tilde{z}_{t+1}$ is used in emission from updated mixture distribution.

In the above process, the parameters in $\boldsymbol{\mu}_i$ are shared by samples in the same cluster, whereby consolidating complementary information for robust forecasting.

Parameterization of Generative Model. The key parametric function in the generative process is $f_{\boldsymbol{\theta}}(\cdot)$, for which we choose a recurrent neural network architecture as

$$
\begin{aligned}
&p(z_{t+1}|z_{1:t}) = \mathrm{softmax}(\mathrm{MLP}(\mathbf{h}_t)),\\
&\text{where } \ \mathbf{h}_t = \mathrm{RNN}(z_t, \mathbf{h}_{t-1}),
\end{aligned}
\tag{5}
$$

and $\mathbf{h}_t$ is the t-th hidden state, MLP represents a multilayer perceptron, RNN can be instantiated by either an LSTM or a GRU. Moreover, to accommodate the applications where the reference time steps of MTS's could be unevenly spaced, we can also incorporate the recently proposed neural ordinary differential equations (ODE) based RNNs (Rubanova, Chen, and Duvenaud 2019) to handle the time intervals. In our experiments, we demonstrate the flexibility of our framework in Eq. (5) by evaluating several choices of RNNs.

Fig. 2(b) illustrates our generative network. In summary, the set of trainable parameters of the generative model is $\boldsymbol{\vartheta} = \{\boldsymbol{\theta}, \boldsymbol{\mu}\}$. Given this, we aim at maximizing the log marginal likelihood of observing each MTS sample, *i.e.*,

$$
\mathcal{L}(\boldsymbol{\vartheta}) = \log\Big(\sum\nolimits_{z_{1:w}} p_{\boldsymbol{\vartheta}}(\mathbf{x}_{1:w}, z_{1:w})\Big) \tag{6}
$$

where the joint probability in Eq. (6) can be factorized w.r.t. the dynamic mixture distribution in Eq. (4) after the Jensen's inequality is applied on $\mathcal{L}(\boldsymbol{\vartheta})$ by

$$
\begin{aligned}
\mathcal{L}(\boldsymbol{\vartheta}) \geq \sum\nolimits_{t=0}^{w-1} \sum\nolimits_{z_{1:t+1}} \Big[&\log\big(p_{\boldsymbol{\vartheta}}(\mathbf{x}_{t+1}|z_{t+1})\big) p_{\boldsymbol{\theta}}(z_{1:t})\\
&\big[(1-\gamma)p_{\boldsymbol{\theta}}(z_{t+1}|z_{1:t}) + \gamma p(\boldsymbol{\mu}_{z_{t+1}})\big]\Big]
\end{aligned}
\tag{7}
$$

in which the above lower bound will serve as our objective to be maximized. The detailed derivation of Eq. (7) is deferred to the supplementary materials.

In order to estimate the parameters $\boldsymbol{\vartheta}$, our goal is to maximize Eq. (7). However, summing out $z_{1:t+1}$ over all possible sequences is computationally difficult. Therefore, evaluating the true posterior density $p(\mathbf{z}|\mathbf{x}_{1:w})$ is intractable. To circumvent this problem, meanwhile enabling inductive analysis, we resort to variational inference (Hoffman et al. 2013) and introduce an inference network in the following.

Inference Network. We introduce an approximated posterior $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}_{1:w})$, which is parameterized by neural networks with parameters $\boldsymbol{\phi}$. We design our inference network to be structural, and employ the idea of deep Markov processes to maintain the temporal dependencies between latent variables, which leads to the following factorization.

$$
q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}_{1:w}) = q_{\boldsymbol{\phi}}(z_1|\mathbf{x}_1) \prod\nolimits_{t=1}^{w-1} q_{\boldsymbol{\phi}}(z_{t+1}|\mathbf{x}_{1:t+1}, z_t) \tag{8}
$$

With the introduction of $q_{\boldsymbol{\phi}}(\mathbf{z}|\mathbf{x}_{1:w})$, instead of maximizing the log marginal likelihood $\mathcal{L}(\boldsymbol{\vartheta})$, we are interested in maximizing the variational *evidence lower bound* (ELBO) $\ell(\boldsymbol{\vartheta}, \boldsymbol{\phi}) \leq \mathcal{L}(\boldsymbol{\vartheta})$ with respect to both $\boldsymbol{\vartheta}$ and $\boldsymbol{\phi}$. By incorporating the bounding step in Eq. (7), we can derive the EBLO of our problem, which is written by

$$
\begin{aligned}
&\ell(\boldsymbol{\vartheta}, \boldsymbol{\phi}) = (1-\gamma) \sum\nolimits_{t=1}^{w} \mathbb{E}_{q_{\boldsymbol{\phi}}(z_t|\mathbf{x}_{1:t})}[\log\big(p_{\boldsymbol{\vartheta}}(\mathbf{x}_t|z_t)\big)]\\
&- \sum_{t=1}^{w-1} \mathbb{E}_{q_{\boldsymbol{\phi}}(z_{1:t}|\mathbf{x}_{1:t})}[\mathcal{D}_{KL}\big(q_{\boldsymbol{\phi}}(z_{t+1}|\mathbf{x}_{1:t+1}, z_t)||p_{\boldsymbol{\vartheta}}(z_{t+1}|z_{1:t})\big)]\\
&- \mathcal{D}_{KL}\big(q_{\boldsymbol{\phi}}(z_1|\mathbf{x}_1)||p_{\boldsymbol{\vartheta}}(z_1)\big)\\
&+ \gamma \sum\nolimits_{t=1}^{w} \sum\nolimits_{z_t=1}^{k} p_{\boldsymbol{\vartheta}}(\boldsymbol{\mu}_{z_t}) \log\big(p_{\boldsymbol{\vartheta}}(\mathbf{x}_t|z_t)\big)
\end{aligned}
\tag{9}
$$

where $\mathcal{D}_{KL}(\cdot||\cdot)$ is the KL-divergence. $p_{\boldsymbol{\vartheta}}(z_1)$ is a uniform prior as described in the generative process. Similar to VAE

(Kingma and Welling 2013), it helps prevent overfitting and improve the generalization capability of our model. The detailed derivation of Eq. (9) can be found in the supplementary materials.

Eq. (9) also sheds some insights on how our dynamic mixture distribution in Eq. (4) works: the first three terms encapsulate the learning criteria for dynamic adjustments; the last term after γ regularizes the relationships between different basis mixture components.

Fig. 2(c) illustrates the architecture of the inference network, in which $q_{\boldsymbol{\phi}}(z_{t+1}|\mathbf{x}_{1:t+1}, z_t)$ is a recurrent structure

$$\begin{aligned} &q_{\boldsymbol{\phi}}(z_{t+1}|\mathbf{x}_{1:t+1}, z_t) = \text{softmax}(\text{MLP}(\tilde{\mathbf{h}}_{t+1})), \\ &\text{where } \tilde{\mathbf{h}}_{t+1} = \text{RNN}(\mathbf{x}_t, \tilde{\mathbf{h}}_t), \end{aligned} \tag{10}$$

$\tilde{\mathbf{h}}_t$ is the t-th latent state of the RNNs, and z_0 is set to $\mathbf{0}$ so that it has no impact in the iteration.

Since sampling discrete variable z_t from the categorical distributions in Fig. 2(c) is not differentiable, it is difficult to optimize the model parameters. To get rid of it, we employ the Gumbel-softmax reparameterization trick (Jang, Gu, and Poole 2017) to generate differetiable discrete samples, which is illustrated by the "sample" steps in Fig. 2(c). In this way, our DGM^2 model is end-to-end trainable.

Gated Dynamic Distributions. In Eq. (4), the dynamics of the Gaussian mixture distribution is tuned by a hyperparameter γ, which may require some tuning efforts on validation datasets. To circumvent it, we introduce a gate function $\gamma(\tilde{\mathbf{h}}_t) = \text{sigmoid}(\text{MLP}(\tilde{\mathbf{h}}_t))$ using the information extracted by the inference network, as shown in Fig. 2(c), to substitute γ in Eq. (4). As such, $\boldsymbol{\psi}_t$ becomes a gated distribution that can be dynamically tuned at each time step.

Model Training

We jointly learn the parameters $\{\boldsymbol{\alpha}, \boldsymbol{\rho}, \boldsymbol{\vartheta}, \boldsymbol{\phi}\}$ of the pre-imputation layer, the generative network $p_{\boldsymbol{\vartheta}}$, and the inference network $q_{\boldsymbol{\phi}}$ by maximizing the ELBO in Eq. (9).

The main challenge to evaluate Eq. (9) is to obtain the gradients of all terms under the expectation $\mathbb{E}_{q_{\boldsymbol{\phi}}}$. Because z_t is categorical, the first term can be analytically calculated with the probability $q_{\boldsymbol{\phi}}(z_t|\mathbf{x}_{1:t})$. However, $q_{\boldsymbol{\phi}}(z_t|\mathbf{x}_{1:t})$ is not an output of the inference network, so we derive a subroutine to compute $q_{\boldsymbol{\phi}}(z_t|\mathbf{x}_{1:t})$ from $q_{\boldsymbol{\phi}}(z_t|\mathbf{x}_{1:t}, z_{t-1})$. In the second term, since the KL divergence is sequentially evaluated, we employ ancestral sampling techniques to sample z_t from time step 1 to w to approximate the distribution $q_{\boldsymbol{\phi}}$. It is also noteworthy that in Eq. (9), we only evaluate observed values in $\mathbf{x}_t$ by using masks $\mathbf{m}_t$ to mask out the unobserved parts. The subroutine to compute $q_{\boldsymbol{\phi}}(z_t|\mathbf{x}_{1:t})$ can be found in the supplementary materials.

As such, the entire DGM^2 is differentiable, and we use stochastic gradient descents to optimize Eq. (9). In the last term of Eq. (9), we also need to perform density estimation of the basis mixture distribution, *i.e.*, to estimate $p(\boldsymbol{\mu})$. Given a batch of MTS samples, suppose there are n temporal features $\mathbf{x}_t$ in this batch, and their collection is denoted by a set $\mathcal{X}$, we can estimate the mixture probability by

$$p(\boldsymbol{\mu}_i) = \sum_{\mathbf{x}_t \in \mathcal{X}} q_{\boldsymbol{\phi}}(z_t = i|\mathbf{x}_{1:t}, z_{t-1})/n, \text{ for } i = 1, ..., k \tag{11}$$

where $q_{\boldsymbol{\phi}}(z_t = i|\mathbf{x}_{1:t}, z_{t-1})$ is the inferred membership probability of $\mathbf{x}_t$ to the i-th latent cluster by Eq. (10).

Experiments

In this section, we evaluate the performance of our DGM^2 model on real-life datasets from different domains and compare it with state-of-the-art approaches.

Datasets and Task Description

USHCN[1] This dataset consists of daily meteorological records collected from 1219 stations across the contiguous states from 1887 to 2009. It has five climate features, such as mean temperature and total daily precipitation. From the dataset, we randomly extracted 5000 segments recorded at all stations in NY state from 1900 to 2000. Each segment is an MTS sample with 100 consecutive daily records. The dataset has a missing ratio of 10.4%. Our task is to use the past 80 days' records to forecast the future 20 days' weather.

KDD-CUP[2] This is an air quality dataset from KDD CUP challenge 2018, which consists of PM2.5 measurements from 35 monitoring stations in Beijing. The values were recorded hourly from 01/2017 to 12/2017, and the overall missing ratio is 16.5%. On this dataset, our task is to forecast the PM2.5 values for the 35 stations in future 12 hours using the historical records in the past 24 hours.

MIMIC-III This is a public clinical dataset (Johnson et al. 2016), with over 58,000 hospital admission records. We collected 31,332 adults' ICU stay data, and extracted 17 clinical features, such as glucose and heart rate, from the first 72 hours using the benchmark tool (Harutyunyan et al. 2019). Because of the heterogeneous sources (e.g., lab tests, routine signals, etc.), the MTS's are highly sparse, with a missing ratio of 72.7%. The task is to forecast the last 24 hour signals using the recorded data in the first 48 hours.

Compared Methods

We compare our model with both conventional approaches and state-of-the-art approaches, including Vector Autoregression (VAR), LSTM, Deep Markov Model (DMM) (Krishnan, Shalit, and Sontag 2017), XGBoost (Chen and Guestrin 2016), GRU-I (Luo et al. 2018), GRU-D (Che et al. 2018a), Interpolation-Prediction Networks (IPN) (Shukla and Marlin 2019), LGNet (Tang et al. 2020), and Latent ODE (L-ODE) (Rubanova, Chen, and Duvenaud 2019).

Among these methods, VAR, XGBoost, LSTM and DMM cannot handle missing values. Therefore, we follow (Tang et al. 2020) to concatenate each MTS with its mask matrix as their input features. GRU-I is a two-step approach, which first imputes missing values with GAN-based networks, and then uses GRU for forecasting. GRU-D, IPN, and LGNet deal with sparsity by jointly training parametic imputation functions and RNNs. GRU-D and LGNet use exponential decay based imputation, IPN uses kernel intensity based functions. In addition, LGNet has a memory module

[1] https://www.ncdc.noaa.gov/ushcn/introduction
[2] https://www.kdd.org/kdd2018/kdd-cup

Method	MIMIC-III		USHCN		KDD-CUP	
	RMSE	MAE	RMSE	MAE	RMSE	MAE
VAR	6.7154±0.0360	3.7420±0.0246	0.9811±0.0183	0.7927±0.0195	0.8164±0.0377	0.5052±0.0256
LSTM	1.0587±0.0091	0.8203±0.0106	0.6340±0.0098	0.4456±0.0098	0.7465±0.0370	0.5082±0.0358
DMM	0.9852±0.0025	0.7510±0.0057	0.6068±0.0092	0.4058±0.0112	0.7067±0.0220	0.5052±0.0303
XGBoost	0.9900±0.0002	0.7209±0.0003	0.8664±0.0011	0.7816±0.0010	0.8841±0.0106	0.7580±0.0102
GRU-I	1.0322±0.0069	0.8256±0.0074	0.9491±0.0046	0.7764±0.0049	0.8233±0.0333	0.5925±0.0364
GRU-D	1.0495±0.0068	0.8502±0.0069	0.9695±0.0089	0.7912±0.0087	0.7268±0.0254	0.5051±0.0228
IPN	0.9888±0.0025	0.7856±0.0039	0.6097±0.0066	0.4204±0.0089	0.7207±0.0313	0.4834±0.0229
LGNet	0.9590±0.0033	0.7093±0.0033	0.5883±0.0071	0.3841±0.0063	0.7346±0.0272	0.5181±0.0218
L-ODE	0.9315±0.0034	0.7325±0.0035	0.6171±0.0056	0.4216±0.0038	0.8226±0.0387	0.5834±0.0331
DGM^2-L	0.9143±0.0025	0.7089±0.0037	0.5426±0.0066	0.3848±0.0047	0.6975±0.0224	0.4748±0.0162
DGM^2-O	**0.9003±0.0015**	**0.6876±0.0027**	**0.4983±0.0053**	**0.3367±0.0059**	**0.6835±0.0276**	**0.4646±0.0213**

Table 1: Forecasting results (RMSE and MAE) of the compared methods on different datasets

Setting		MIMIC-III	USHCN	KDD-CUP
-L	before	1.4111	0.7780	5.2363
	after	**0.9052**	**0.5250**	**0.5506**
-O	before	1.4186	0.4761	4.2868
	after	**0.8979**	**0.4663**	**0.5362**

Table 2: Imputation results (RMSE) before/after the forecasting component of DGM^2-L (-L) and DGM^2-O (-O)

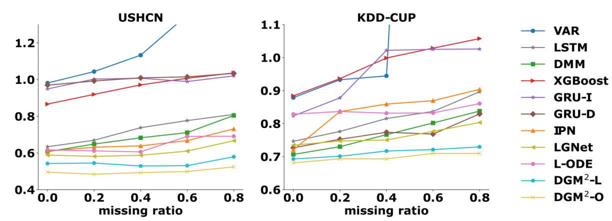

Figure 3: Forecasting results (RMSE) of the compared methods w.r.t. varying missing ratios on USHCN and KDD-CUP

Model	MIMIC-III	USHCN	KDD-CUP
(a) $\gamma = 1.0$	0.9832	0.9913	0.9998
(b) $\gamma = 0.0$	0.9191	0.5151	0.7533
(c) $\gamma = 1e^{-2}$	0.9033	**0.4958**	0.6878
(d) Gate $\gamma(\cdot)$	**0.9003**	0.4983	**0.6835**

Table 3: Ablation analysis (RMSE)

to leverage global MTS patterns in its RNNs. Different from the above methods, L-ODE addresses sparsity by modeling uneven time intervals via learnable ordinary differential equations. If a temporal feature $\mathbf{x}_t$ is too sparse (*e.g.*, the sparsity is above a threshold), L-ODE will replace it by an interval, thus generating unevenly spaced MTS's. It is worth noting that none of these methods exploits the clustering structures underlying the MTS set.

For our method, since it is a flexible framework, we use LSTM and ODE as the RNNs, and denote them by DGM^2-L and DGM^2-O, where the latter has the capability to handle uneven time intervals. To gain further insights on some of the design choices, we also compare DGM^2 with its variants, which will be discussed in the ablation analysis.

Experimental Setup

For each dataset, train/valid/test sets were split as 70/10/20. All compared methods were trained by Adam optimizer with hyperparameters selected on the validation set. The configurations of the compared methods are described in the supplementary materials. For DGM^2, we grid-searched k, *i.e.*, the number of mixture components (or $\boldsymbol{\mu}_i$'s), from 10 to 200. The γ in Eq. (4) was searched within $\{1e^{-5}, 1e^{-4}, 1e^{-3}, 1e^{-2}, 1e^{-1}\}$. The gate function $\gamma(\cdot)$ was also tested to automatically tune Eq. (4). The variance σ of Gaussian distributions was selected from $\{1e^{-5}, 1e^{-4}, 1e^{-3}, 1e^{-2}, 1e^{-1}\}$.

Similar to (Krishnan, Shalit, and Sontag 2017), we configured DGM^2-L with one-layer LSTMs, and searched the hidden dimensionality within $\{10, 20, 30, 40, 50\}$. For DGM^2-O, we follow (Rubanova, Chen, and Duvenaud 2019) to use an MLP for instantiating the neural networks, with dimensionality selected from $\{10, 20, 30, 40, 50\}$. The parameters in $\boldsymbol{\mu}$ were randomly initialized. Early stopping was applied to avoid overfitting. To evaluate the performance of the compared methods, we use the root mean square error (RMSE) and mean absolute error (MAE) (Che et al. 2018b). For both metrics, smaller values indicate better performance.

Experimental Results

Forecasting Table 1 summarizes the average results of the compared methods over 5 runs, from which we have several observations. First, methods under joint imputation-prediction framework, such as IPN and LGNet, often outperform traditional methods. This validates the usefulness of learning task-aware missing patterns. Second, DMM performs well even without special designs for handling sparsity, which suggests the superiority of generative models on the forecasting tasks. We also observe good results of L-ODE on USHCN and MIMIC-III. This may indicate the effectiveness of modeling time intervals as another way toward handling sparse MTS's. Finally, the proposed DGM^2-L outperforms other methods in most cases, which validates its design as a generative model with a joint imputation-prediction paradigm. More importantly, this justifies its dynamic modeling of latent clustering structures. Furthermore, our flexible framework enables DGM^2-O, which incorpo-

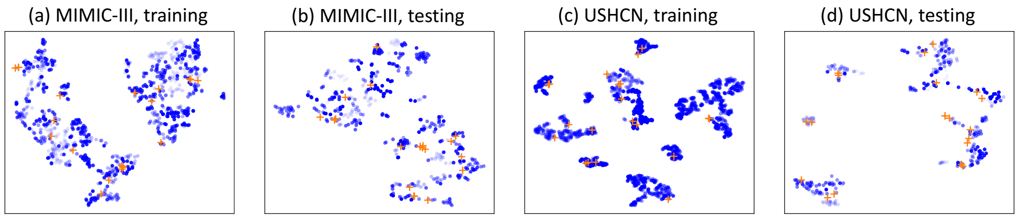

Figure 4: The tSNE visualization on MIMIC-III and USHCN datasets. Circles are temporal features. Different transparencies indicate different MTS samples from which the features are extracted. "+" markers are the learned Gaussian means by DGM^2.

rates the advantages of ODE. It obtains further improvements, with relative improvements on RMSE of at least 3.5%, 18.1% and 3.4% on MIMIC-III, USHCN, and KDD-CUP datasets, respectively. The statistics in Table 1 also demonstrate that the improvements are significant.

To gain further insights about the usefulness of modeling clustering structures, we compare the outputs of the pre-impulation layer and the forecasting component of DGM^2. Since the forecasting component generates values at every time step, the generated values at those missing entries, *i.e.*, $x_t^i = \texttt{NaN}$, can be regarded as new imputations. If the newly imputed values are more accurate than the pre-imputed values, then it implies the learned clustering structures can facilitate generating true MTS's effectively. To this end, we randomly removed 10% of the observations in each dataset, and evaluated the imputation error. Table 2 summarizes the average results of 5 runs. From Table 2, we can observe both variants of DGM^2 significantly reduce RMSE via the generative model, which hence validates the usefulness of the learned clustering structures.

Robustness Next, we evaluate the robustness of DGM^2 w.r.t. varying missing ratios using USHCN and KDD-CUP datasets, which have moderate sparsity. Specifically, we randomly dropped δ ($0 \leq \delta \leq 1$) fraction of the observed values, and tuned δ from 0 to 0.8. The compared methods were trained on the corrupted datasets, and evaluated on the same forecasting tasks as before. Fig. 3 reports the results in terms of RMSE. First of all, Fig. 3 shows that the forecasting errors of the methods without handling sparsity, such as VAR, LSTM, DMM, go up quickly as δ increases. In comparison, methods that explicitly address sparsity, *e.g.*, GRU-D, LGNet, and L-ODE, can maintain relatively stabler results to varying extents. However, many of them still suffer when the missing ratio is very high, *e.g.*, $\delta \geq 0.6$. In such scenarios, both DGM^2-L and DGM^2-O obtain the best forecasting accuracy, which is attributed to the modeling of robust clustering structures via dynamic Gaussian mixtures.

Ablation Analysis In this section, we focus on the analysis of the gating mechanism $\gamma(\cdot)$ introduced in our inference network. Recall in Eq. (4), γ is a hyperparameter controlling the dynamics in the Gaussian mixture. Table 3 compares the results of different choices on γ, as well as the use of $\gamma(\cdot)$, in terms of RMSE. $\gamma = 0$ and $\gamma = 1$ correspond to the extreme cases when there is no basis mixture and no dynamic adjustment, respectively. From Table 3, both cases lead to sub-optimal performance, especially when there is no modeling of dynamics, *i.e.*, $\gamma = 1$. $\gamma = 1e^{-2}$ is the optimal choice from grid search, which trades off the two cases for improved results. The choice of a small γ also indicates a few introduction of basis mixture is sufficient. Moreover, the comparable results of cases (c) and (d) in Table 3 validates the effective design of the gating function $\gamma(\cdot)$, which can help save a lot of tuning efforts.

Visualization As discussed before, DGM^2 explores the latent clustering structures by learning a basis mixture distribution and tuning it over time. To understand how DGM^2 uncovers the clustering structures, we randomly sampled a batch of training (testing) samples for visualization (the full set is too large to be visualized). We visualized every temporal feature $\mathbf{x}_t$ using tSNE (Maaten and Hinton 2008) in a 2D space. The missing values in $\mathbf{x}_t$'s were imputed using the outputs of the forecasting component.

Fig. 4 presents the visualization results and the Gaussian means $\boldsymbol{\mu}$ learned by DGM^2-O (with gate function $\gamma(\cdot)$) on MIMIC-III and USHCN datasets. For the training set, we visualized all fitted data. For the testing set, we investigated the forecasting parts, *i.e.*, last 24 hours (20 days) on MIMIC-III (USHCN). In the figure, circles represent temporal features, "+" markers represent the learned Gaussian means. Different transparencies indicate different MTS samples. From the figure, we can clearly observe clusters. In particular, features $\mathbf{x}_t$ from different MTS's may stay in the same cluster, which implies different samples (*e.g.*, patients) may share the same state (*e.g.*, pathological condition) at different periods. From Fig. 4(a)(c), DGM^2 can effectively learn Gaussian means in dense areas from the training set, so that the mixture distributions are well fitted. Moreover, Fig. 4(b)(d) demonstrate the learned means fit the testing data as well, which explains how the new time series are forecasted meanwhile.

Conclusions

In this paper, we proposed a new method, Dynamic Gaussian Mixture based Deep Generative Model (DGM^2), for robust forecasting on sparse multivariate time series (MTS). DGM^2 achieves robustness by modeling the transition of latent clusters of temporal features, and emitting MTS's from dynamic Gaussian mixture distributions. We parameterized the generative model by neural networks and developed an inference network for enabling inductive analysis. The extensive experimental results on a variety of real-life datasets demonstrated the effectiveness of our proposed method.

Acknowledgments

This material is based upon work that is in part supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. HR001117C0047.

Ethical Impact Statement

Multivariate time series forecasting has high impacts in wide domains, such as medicine, finance, and meteorology. In many emerging applications, MTS's collected from different sources are often interrelated and demands collective analysis for fully understanding the monitored targets, which necessitates solutions to handle sparsity. Such solutions do not only save engineering efforts for combining data, but also enhances predictive performance in important scenarios including clinical diagnosis, traffic surveillance, and large-scale system debugging. Our proposed model was tested on datasets from a variety of scenarios, which showcases some of its societal impacts.

Moreover, the theoretical design and analysis of the dynamic Gaussian mixture distribution itself may have broader impacts in applications other than MTS forecasting. Static Gaussian mixture has been extensively used in many applications, while its dynamic counterpart is less developed. Our proposed method provides a novel and general solution that explicitly defines temporal dependency between Gaussian mixture distributions at different time steps. It has a potential to be used on different types of sequential data such as sentences, dynamic graphs, and videos for modeling the flows of clustering structures which can be topics, social communities, and concepts, thus generates values in the correspondingly various areas.

References

Azur, M. J.; Stuart, E. A.; Frangakis, C.; and Leaf, P. J. 2011. Multiple imputation by chained equations: what is it and how does it work? *INT J METH PSYCH RES* 20(1): 40–49.

Baytas, I. M.; Xiao, C.; Zhang, X.; Wang, F.; Jain, A. K.; and Zhou, J. 2017. Patient subtyping via time-aware lstm networks. In *KDD*, 65–74.

Binkowski, M.; Marti, G.; and Donnat, P. 2018. Autoregressive convolutional neural networks for asynchronous time series. In *ICML*, 580–589.

Cao, W.; Wang, D.; Li, J.; Zhou, H.; Li, L.; and Li, Y. 2018. Brits: Bidirectional recurrent imputation for time series. In *NeurIPS*, 6775–6785.

Che, Z.; Purushotham, S.; Cho, K.; Sontag, D.; and Liu, Y. 2018a. Recurrent neural networks for multivariate time series with missing values. *Scientific reports* 8(1): 1–12.

Che, Z.; Purushotham, S.; Li, G.; Jiang, B.; and Liu, Y. 2018b. Hierarchical deep generative models for multi-rate multivariate time series. In *ICML*, 784–793.

Chen, T.; and Guestrin, C. 2016. Xgboost: A scalable tree boosting system. In *KDD*, 785–794.

Choi, E.; Bahadori, M. T.; Sun, J.; Kulas, J.; Schuetz, A.; and Stewart, W. 2016. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. In *NeurIPS*, 3504–3512.

De Brouwer, E.; Simm, J.; Arany, A.; and Moreau, Y. 2019. GRU-ODE-Bayes: Continuous modeling of sporadically-observed time series. In *NeurIPS*, 7379–7390.

Diaz-Rozo, J.; Bielza, C.; and Larrañaga, P. 2018. Clustering of data streams with dynamic gaussian mixture models: an IoT application in industrial processes. *IEEE Internet Things J.* 5(5): 3533–3547.

Harutyunyan, H.; Khachatrian, H.; Kale, D. C.; Ver Steeg, G.; and Galstyan, A. 2019. Multitask learning and benchmarking with clinical time series data. *Sci. Data* 6(1): 96. ISSN 2052-4463. doi:10.1038/s41597-019-0103-9. URL https://doi.org/10.1038/s41597-019-0103-9.

Hoffman, M. D.; Blei, D. M.; Wang, C.; and Paisley, J. 2013. Stochastic variational inference. *J MACH LEARN RES* 14(1): 1303–1347.

Inaguma, D.; Morii, D.; Kabata, D.; Yoshida, H.; Tanaka, A.; Koshi-Ito, E.; Takahashi, K.; Hayashi, H.; Koide, S.; Tsuboi, N.; et al. 2019. Prediction model for cardiovascular events or all-cause mortality in incident dialysis patients. *PloS one* 14(8): e0221352.

Jang, E.; Gu, S.; and Poole, B. 2017. Categorical reparameterization with gumbel-softmax. In *ICLR*.

Johnson, A. E.; Pollard, T. J.; Shen, L.; Li-Wei, H. L.; Feng, M.; Ghassemi, M.; Moody, B.; Szolovits, P.; Celi, L. A.; and Mark, R. G. 2016. MIMIC-III, a freely accessible critical care database. *Sci. Data* 3(1): 1–9.

Kingma, D. P.; and Welling, M. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114* .

Koren, Y.; Bell, R.; and Volinsky, C. 2009. Matrix factorization techniques for recommender systems. *Computer* 42(8): 30–37.

Krishnan, R. G.; Shalit, U.; and Sontag, D. 2017. Structured inference networks for nonlinear state space models. In *AAAI*, 2101–2109.

Li, Y.; Yu, R.; Shahabi, C.; and Liu, Y. 2018. Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting. In *ICLR*.

Lipton, Z. C.; Kale, D. C.; and Wetzel, R. 2016. Modeling missing data in clinical time series with rnns. *arXiv preprint arXiv:1606.04130* .

Luo, Y.; Cai, X.; Zhang, Y.; Xu, J.; et al. 2018. Multivariate time series imputation with generative adversarial networks. In *NeurIPS*, 1596–1607.

Maaten, L. v. d.; and Hinton, G. 2008. Visualizing data using t-SNE. *J MACH LEARN RES* 9(Nov): 2579–2605.

Qin, Y.; Song, D.; Chen, H.; Cheng, W.; Jiang, G.; and Cottrell, G. W. 2017. A Dual-Stage Attention-Based Recurrent Neural Network for Time Series Prediction. In *IJCAI*.

Rehfeld, K.; Marwan, N.; Heitzig, J.; and Kurths, J. 2011. Comparison of correlation analysis techniques for irregularly sampled time series. *NONLINEAR PROC GEOPH* 18(3): 389–404.

Rubanova, Y.; Chen, R. T.; and Duvenaud, D. 2019. Latent ODEs for Irregularly-Sampled Time Series. In *NeurIPS*.

Safari, S.; Shabani, F.; and Simon, D. 2014. Multirate multisensor data fusion for linear systems using Kalman filters and a neural network. *Aerosp. Sci. Technol.* 39: 465–471.

Shi, X.; Chen, Z.; Wang, H.; Yeung, D.-Y.; Wong, W.-K.; and Woo, W.-c. 2015. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *NeurIPS*, 802–810.

Shukla, S. N.; and Marlin, B. 2019. Interpolation-Prediction Networks for Irregularly Sampled Time Series. In *ICLR*.

Tang, X.; Yao, H.; Sun, Y.; Aggarwal, C. C.; Mitra, P.; and Wang, S. 2020. Joint Modeling of Local and Global Temporal Dynamics for Multivariate Time Series Forecasting with Missing Values. In *AAAI*, 5956–5963.

Tüske, Z.; Golik, P.; Schlüter, R.; and Ney, H. 2015a. Speaker adaptive joint training of gaussian mixture models and bottleneck features. In *ASRU*, 596–603. IEEE.

Tüske, Z.; Tahir, M. A.; Schlüter, R.; and Ney, H. 2015b. Integrating Gaussian mixtures into deep neural networks: Softmax layer with hidden variables. In *ICASSP*, 4285–4289. IEEE.

Variani, E.; McDermott, E.; and Heigold, G. 2015. A Gaussian mixture model layer jointly optimized with discriminative features within a deep neural network architecture. In *ICASSP*, 4270–4274. IEEE.

Wei, X.; Sun, J.; and Wang, X. 2007. Dynamic Mixture Models for Multiple Time-Series. In *IJCAI*, volume 7, 2909–2914.

Zaheer, M.; Ahmed, A.; and Smola, A. J. 2017. Latent LSTM allocation: Joint clustering and non-linear dynamic modeling of sequence data. In *ICML*, 3967–3976.

Zhang, C.; Song, D.; Chen, Y.; Feng, X.; Lumezanu, C.; Cheng, W.; Ni, J.; Zong, B.; Chen, H.; and Chawla, N. V. 2019. A deep neural network for unsupervised anomaly detection and diagnosis in multivariate time series data. In *AAAI*, volume 33, 1409–1416.

Zhang, C.; and Woodland, P. C. 2017. Joint optimisation of tandem systems using Gaussian mixture density neural network discriminative sequence training. In *ICASSP*, 5015–5019. IEEE.