Synergetic Learning of Heterogeneous Temporal Sequences for Multi-Horizon Probabilistic Forecasting

Longyuan Li[1,2*]**, Jihai Zhang**[3*]**, Junchi Yan**[1,3†]**, Yaohui Jin**[1,2 †]**,**
Yunhao Zhang[3]**, Yanjie Duan**[4]**, Guangjian Tian**[4]

[1] MoE Key Lab of Artificial Intelligence, AI Institute, Shanghai Jiao Tong University
[2] State Key Lab of Advanced Optical Communication System and Network, Shanghai Jiao Tong University
[3] Department of Computer Science and Engineering, Shanghai Jiao Tong University
[4] Huawei Noah's Ark Lab
{jeffli,yunfan243332345,yanjunchi,jinyh, zhangyunhao}@sjtu.edu.cn, {duanyanjie1,tian.guangjian}@huawei.com

Abstract

Time-series is ubiquitous across applications, such as transportation, finance and healthcare. Time-series is often influenced by external factors, especially in the form of asynchronous events, making forecasting difficult. However, existing models are mainly designated for either synchronous time-series or asynchronous event sequence, and can hardly provide a synthetic way to capture the relation between them. We propose Variational Synergetic Multi-Horizon Network (VSMHN), a novel deep conditional generative model. To learn complex correlations across heterogeneous sequences, a tailored encoder is devised to combine the advances in deep point processes models and variational recurrent neural networks. In addition, an aligned time coding and an auxiliary transition scheme are carefully devised for batched training on unaligned sequences. Our model can be trained effectively using stochastic variational inference and generates probabilistic predictions with Monte-Carlo simulation. Furthermore, our model produces accurate, sharp and more realistic probabilistic forecasts. We also show that modeling asynchronous event sequences is crucial for multi-horizon time-series forecasting.

Introduction

Temporal data streams are ubiquitous in areas including transportation, electronic health, economics, environment monitoring. One major type of temporal data is time-series (also known as synchronous sequence, e.g. temperature, economic index, ECG records), which consists of successive evenly-sampled discrete-time data points. Another type of temporal data is event sequence (also known as asynchronous sequence, e.g. e-commerce transactions, social interactions, extreme weathers), which consists of data points irregularly dispersed in the continuous-time domain (Xiao et al. 2017). Both types carry rich information about the evolution of complex systems, based on which effective predictions are crucial to subsequent decision making, especially for those time-sensitive cases. Multi-horizon time-series forecasting has wide application scenarios that can be integrated into

_____

*The first two authors have made equal contribution.

†Corresponding authors are Junchi Yan and Yaohui Jin.

many business processes. One particular setting is to forecast long term multi-step future time-series as a whole. The hope is to avoid error accumulation that is common in single-step autoregressive models (Benidis et al. 2020).

While there have been recent works on modeling time-series (Chandola, Banerjee, and Kumar 2012; Makridakis, Spiliotis, and Assimakopoulos 2018; Mariet and Kuznetsov 2019; Li et al. 2019b; Rangapuram et al. 2018; Salinas et al. 2019b) and event sequence (Du et al. 2016; Bacry, Mastromatteo, and Muzy 2015; Xiao et al. 2016, 2017; Wu et al. 2018). Most of them are focused on one of these two types and the research in these two directions are conducted separately and independently. However, the two types of temporal data are usually strongly correlated and are likely to have lead-lag relationships. In other words, time-series is influenced by temporal events, and temporal events may also be caused or influenced by certain fluctuations in time-series. For example, traffic accidents may lead to high road occupancy rate, and constant high road occupancy rate over time can cause traffic accidents or traffic jams.

Since the two types of data are interrelated, using only one can possibly lead to deviation in prediction. We believe there is a real need to jointly model both types of data. There are related efforts in linking the synchronous time-series and asynchronous event sequences to each other. One way is to aggregate event sequences to fixed-interval count sequences to extract aligned time-series data. The other is to convert time-series to event sequences by extracting events from time-series based on human-defined rules (Bacry, Mastromatteo, and Muzy 2015). However, such coarse treatments can lead to loss of key information about the actual dynamics of either two processes (Bińkowski, Marti, and Donnat 2017).

This paper addresses the problem of jointly modeling time-series and event sequences for multi-horizon time-series forecasting. We propose Variational Synergetic Multi-Horizon Network (VSMHN). The basic idea is to learn the probability distribution of future values conditioned on heterogeneous past sequences (i.e. time-series and event sequence). Our model is based on conditional variational autoencoder (CVAE) (Sohn, Lee, and Yan 2015). We devise a hybrid recognition model which combines advances in deep point processes models and variational recurrent neural networks. The

experimental results show that our model is able to untwine factors of asynchronous event sequence from time-series, and provides accurate and sharp multi-horizon probabilistic forecasting fulfilled by Monte-Carlo sampling.

In summary, the main highlights of the paper are:

1) We argue the importance of capturing the interaction between event sequence and time-series in prediction. To our best knowledge, this is the first work that considers the problem of multi-horizon time-series forecasting with synchronous and asynchronous past temporal sequences.

3) Our model combines neural networks and probabilistic models. It can be trained by stochastic gradient descent and scales to large datasets. As the embodiment of both recognition model and prior model, the Time-Aware Hybrid RNN is carefully devised to preserve valuable timing information.

3) We conduct experiments on public real-world datasets, showing its superiority against state-of-the-arts. We find event sequence is crucial to accurate multi-horizon forecasting, even when events themselves are indirectly extracted from the time-series. This also indicates the utility of our approach for multi-dimensional time-series data.

Related Works

Variational Autoencoder (VAE). VAE (Kingma and Welling 2013) is an efficient way to handle latent variables in neural networks. The VAE learns a generative model of the marginal probability $p_\theta(\mathbf{x})$ of the observed data $\mathbf{x}$. The generative process of the VAE is as follows: A set of latent variable $\mathbf{z}$ is generated from the prior distribution $p_\theta(\mathbf{z})$ and the data $\mathbf{x}$ is generated by the generative model $p_\theta(\mathbf{x}|\mathbf{z})$ conditioned on $\mathbf{z}$. A recognition model $q_\phi(\mathbf{z}|\mathbf{x})$ is introduced to approximate the true posterior $p_\theta(\mathbf{z}|\mathbf{x})$. Further, the conditional variational autoencoder (CVAE) (Sohn, Lee, and Yan 2015) extends the VAE to model conditional generative process $p_\theta(\mathbf{y}|\mathbf{x})$ of target $\mathbf{y}$ given features $\mathbf{x}$.

Time-series forecasting. Traditional statistical forecasting models include linear autoregressive models, such as ARIMA (Box et al. 2015), Exponential Smoothing (Hyndman et al. 2008) and VAR (Lütkepohl 2005). They are well-understood and still competitive in many forecasting competitions (Makridakis, Spiliotis, and Assimakopoulos 2018). Deep neural networks have been proposed to learn from multiple related time-series by fusing traditional models, such as DeepAR (Salinas et al. 2019b), RNN and Gaussian copula process model (Salinas et al. 2019a), Deep State Space models (Rangapuram et al. 2018) and its interpretable version (Li et al. 2019a), Deep Factor models (Wang et al. 2019).

Another line of network architectures for multi-horizon forecasting involves sequence-to-sequence models (Sutskever, Vinyals, and Le 2014; Fan et al. 2019), which are powerful tools in the domain of Natural Language Processing (NLP), and are often used in machine translation tasks. Rather than modeling each time point within a sequence in an autoregressive manner, the sequence-to-sequence models use an 'encoder-deocoder' framework, to learn a mapping between arbitrarily long sequences through an intermediate encoded state. Its loss (Vincent and Thome 2019), evaluation metrics (Taieb and Atiya 2015) and general-
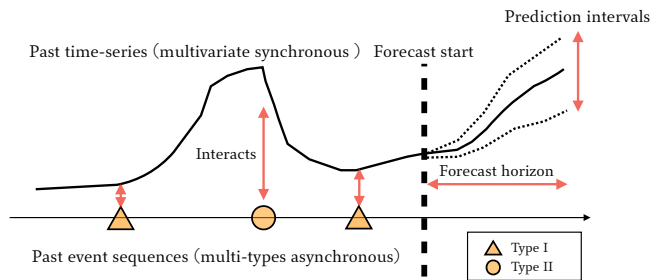

Figure 1: Multi-horizon probabilistic forecasting for heterogeneous sequences. As past time-series is influenced by events, a joint model is used to untwine the exogenous factors. Event sequence is unknown in forecasting horizon, and we are interested in confidence intervals of predictions.

ization bounds (Mariet and Kuznetsov 2019) are well-studied in the context of multi-horizon time-series forecasting.

The methods mentioned above are for evenly-sampled synchronous time-series. Models for other types of time-series such as multi-rate time-series (Che et al. 2018) and irregularly-sampled time-series (Bińkowski, Marti, and Donnat 2017; Baytas et al. 2017) are developed, however, they cannot generalize to model heterogeneous temporal data.

Temporal Point Processes (TPPs). Different from evenly-sampled time-series, the timestamps of events carry rich information of dynamics. TPPs are powerful tools for modeling such continuous-time event sequence (Wu et al. 2018). Recently, neural point processes such as Recurrent Point Process (Xiao et al. 2017), Recurrent Marked Temporal Point process (Du et al. 2016) and Neural Hawkes process (Mei and Eisner 2017) are broadly discussed. (Xiao et al. 2019) jointly learns event sequences with time-series. However, like other models mentioned above, it can only predict the timestamp and type of the next event, and thus is not suitable for our multi-horizon time series forecasting problem.

One alternative is to convert event sequences to dummy-coded time-series aligned with others (0/1 sequence for each event type). However, this approach has obvious drawbacks: 1) The converted event sequence is sparse and contains many zeros between events; 2) The dimension equals to the number of event types, making the sequence even sparser; 3) The information of the intervals between events is lost, especially when more than one event occurs in a single interval.

Based on the observations above, if we model such data using neural networks such as GRUs or LSTMs, the required non-linearity of the model and the amount of information are imbalanced: The model need to have many layers to memory the previous event after many superfluous transitions, while an asynchronous approach just need to memory last event and type as they are. In conclusion, a hybrid model is welcomed to fully utilize information and learn correlations from heterogeneous temporal sequences.

Variational Synergetic Multi-Horizon Network

Problem Formulation. Denote an M-dimensional multivariate evenly-sampled time-series having T time steps:

$\mathcal{X}_{1:T} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$ where each $\mathbf{x}_t \in \mathbb{R}^M$. A corresponding irregularly-sampled event sequence can be represented as $\{(c_1, t_1), (c_2, t_2), \ldots (c_n, t_n)\}$, where $c_i \in \{1, \ldots, C\}$ denotes the class of the i-th event and t_i is the time of occurrence, and $t_n \leq T$. We assume the events arrive over time, i.e. $\{t_i \in \mathbb{R} | t_i > t_{i-1}\}$. Let $\mathcal{H}_t = \{(c_i, t_i) | t_i < t\}$ denote the event history until t. Our goal is to estimate the distribution of time-series in the future window $\tau \in \mathbb{N}^+$ given history of time-series and event (see Fig 1.):

$$p(\mathcal{X}_{T+1:T+\tau} | \mathcal{X}_{1:T}, \mathcal{H}_T, \Phi) \qquad (1)$$

where Φ denotes the model parameters. Note that time-series starts from $t = 1$, and event sequence start from $t = 0$. Note that different from exogenous variables of other models, event sequence is unknown in the forecast horizon.

Objective. The forecasting quality is generally measured by a metric between the predicted and actual values in the forecast horizon. Probabilistic forecasts estimate a probability distribution rather than predicting a single value as point forecasts do. Simple accuracy metrics such as MAE, RMSE are considered incomplete because they are unable to evaluate uncertainties. The Continuous Ranked Probability Score (CRPS) generalizes the MAE to evaluate probabilistic forecasts (Gneiting and Katzfuss 2014). Given the true observation x and the cumulative distribution function (CDF) of its forecasts distribution F_X, the CRPS is given by:

$$\text{CRPS}(F_X, x) = \int_{-\infty}^{\infty} \left(F_X(y) - \mathbb{1}(y - x) \right)^2 \, \mathrm{d}y \qquad (2)$$

where $\mathbb{1}$ is the Heaviside step function.

Model Architecture. Our task is to model the probability distribution of future time-series $\mathcal{F} = \mathcal{X}_{T+1:T+\tau}$ given heterogeneous past data $\mathcal{P} = \{\mathcal{X}_{1:T}, \mathcal{H}_T\}$. We denote the modeled conditional distribution by $p(\mathcal{F}|\mathcal{P})$. We introduce a latent variable $\mathbf{Z}$ conditioned on the observed data, and a variational autoencoder conditioned on $\mathcal{P}$ (CVAE (Sohn, Lee, and Yan 2015)) is trained to maximize the conditional log likelihood of $\mathcal{F}$ given $\mathcal{P}$, which involves an intractable marginalization over the latent variable $\mathbf{Z}$, i.e.:

$$p(\mathcal{F}|\mathcal{P}) = \int_Z p(\mathcal{F}, \mathbf{Z}|\mathcal{P}) \, \mathrm{d}\mathbf{Z} = \int_Z p(\mathcal{F}|\mathcal{P}, \mathbf{Z}) p(\mathbf{Z}|\mathcal{P}) \, \mathrm{d}\mathbf{Z} \qquad (3)$$

The CVAE can be trained by maximizing the variational evidence lower bound (ELBO) of the conditional log likelihood $\mathcal{L}(\theta, \phi)$, which is derived as:

$$\begin{aligned} \log p_\theta(\mathcal{F}|\mathcal{P}) = {} & \text{KL}\left(q_\phi(\mathbf{Z}|\mathcal{P}, \mathcal{F}) \| p_\theta(\mathbf{Z}|\mathcal{P}, \mathcal{F})\right) \\ & + \mathbb{E}_{q_\phi}\left[-\log q_\phi(\mathbf{Z}|\mathcal{P}, \mathcal{F}) + \log p_\theta(\mathcal{F}, \mathbf{Z}|\mathcal{P})\right] \\ \geq {} & -\text{KL}(q_\phi(\mathbf{Z}|\mathcal{P}, \mathcal{F}) \| p_\theta(\mathbf{Z}|\mathcal{P})) \\ & + \mathbb{E}_{q_\phi}\left[\log p_\theta(\mathcal{F}|\mathcal{P}, \mathbf{Z})\right] \coloneqq \mathcal{L}(\theta, \phi) \end{aligned}$$
$$(4)$$

where $q_\phi(\mathbf{Z}|\mathcal{P}, \mathcal{F})$ is simplified by notation q_ϕ, and KL denotes Kullback-Leibler (KL) divergence. A proposal distribution $q_\phi(\mathbf{Z}|\mathcal{P}, \mathcal{F})$ parameterized by ϕ, which is known as a 'recognition' model or probabilistic 'encoder', is introduced

to approximate the true posterior distribution $p_\theta(\mathbf{Z}|\mathcal{P}, \mathcal{F})$. The future time-series distribution is generated from the distribution $p_\theta(\mathcal{F}|\mathcal{P}, \mathbf{Z})$, which is also known as 'generative' model or probabilistic 'decoder'. The prior distribution $p_\theta(\mathbf{Z}|\mathcal{P})$ of latent variable $\mathbf{Z}$ is modulated by the past $\mathcal{P}$.

Then we need to parameterize the three distributions and optimize $\mathcal{L}$ with respect to θ and ϕ. The KL term of Eq. 4 can be analytically computed assuming a simple distribution of $\mathbf{Z}$, typically Gaussian, while the second expectation term cannot, making $\mathcal{L}$ not differentiable. To obtain a differentiable estimation of ELBO, we resort to Stochastic Gradient Variational Bayes (SGVB) (Kingma and Welling 2013). The basic idea is to sample from an auxiliary stochastic variable instead directly from the recognition model q_ϕ. As such, the gradient of the objective can be back-propagated through the sampled $\mathbf{Z}$. Formally we have the empirical lower bound:

$$\begin{aligned} \tilde{\mathcal{L}}_{\text{SGVB}}(\mathcal{P}, \mathcal{F}; \theta, \phi) = {} & -\text{KL}\left(q_\phi(\mathbf{Z}|\mathcal{P}, \mathcal{F}) \| p_\theta(\mathbf{Z}|\mathcal{P})\right) \\ & + \frac{1}{K} \sum_{k=1}^{K} \log p_\theta\left(\mathcal{F}|\mathcal{P}, \mathbf{Z}^{(k)}\right) \end{aligned}$$
$$(5)$$

where $\mathbf{Z}^{(k)} = q_\phi(\mathcal{P}, \mathcal{F}, \boldsymbol{\epsilon}^{(k)})$, $\boldsymbol{\epsilon}^{(k)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and K is the number of samples.

Remarks. Compared with a direct deterministic mapping such as $\mathcal{F} = f_\theta(\mathcal{P})$, the stochastic latent variable $\mathbf{Z}$ allow for modeling multiple modes in conditional distribution of future time-series $\mathcal{F}$ given past data $\mathcal{P}$, making the proposed model able to model one-to-many mapping, which is common in time-series forecasting tasks e.g. the same past data lead to different future data due to concept drift (Gama et al. 2014).

Moreover, in other probabilistic mapping approaches, such as quantile regression loss MQRNN (Wen et al. 2017) or likelihood models (Salinas et al. 2019b), the randomness only exists in the expectation term $\mathbb{E}$ in Eq. 4, and thus can be viewed as maximum likelihood approaches. In contrast, our model marginalizes the latent variable, and the KL-term can be viewed as a regularization term. As a result, compared with maximum likelihood approaches, our approach has the potential to learn better uncertainty of the data distribution and be more robust on small datasets.

Implementations. As previously discussed, the time-series is evenly-sampled discrete-time synchronous data, while the alongside event sequence is continuous-time asynchronous data, making it difficult to model them jointly, especially to preserve the actual time order, which is essential for causality, i.e. past data cannot be influenced by future data. To jointly learn representations of heterogeneous sequences, we introduce the Time-aware Hybrid RNN as a building block of recognition model and prior model as shown in Figure 2.

Time-Aware Hybrid RNN. Suppose an input of past time-series $\mathbf{X}_{1:T}$ with strict T observation vectors and event sequences $\mathcal{H}_T$, where there are $M \in \mathbb{Z}^{\geq 0}$ events within the past range $(0, T]$. Firstly, we use an RNN f_ϕ to model the time-series, which is transited at fixed time $\{1, \ldots, T\}$. For asynchronous events, motivated by (Du et al. 2016), we build an RNN g_ϕ that transits at each event, and takes timing and embedded event type as input. Moreover, to learn them in
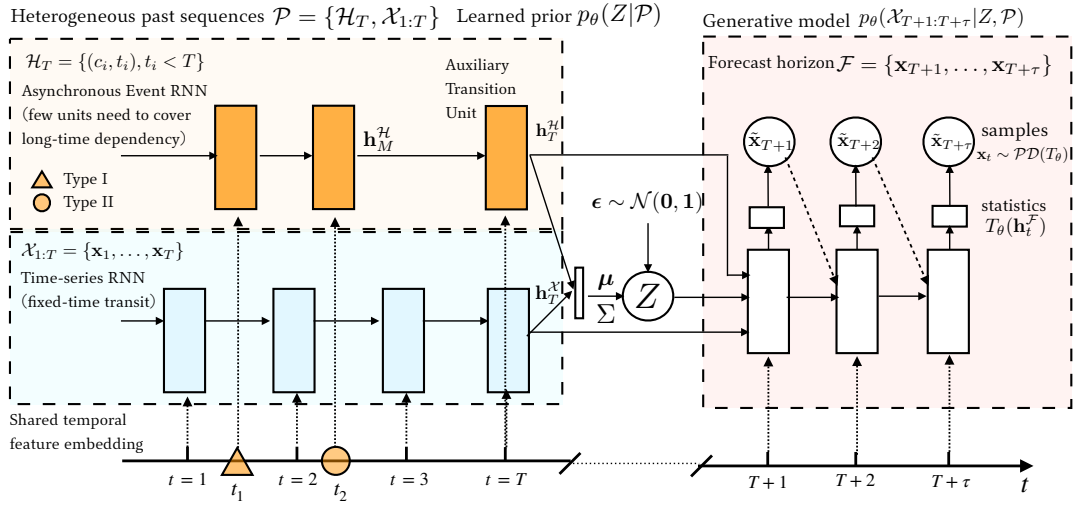
Figure 2: Overview of our synergetic model for probabilistic multi-horizon time series forecasting. In the plot, we set history length $T = 4$, and $M = 2$ event occurrences within the time range, the forecast horizon $\tau = 3$. The left side is the prior model $p_\theta(\mathbf{Z}|\mathcal{P})$. In training, the recognition model (not shown in the plot) shares the same parameters of the time-series RNN with the prior model. The recognition model transits τ more times with ground truth future value input and then outputs $\mathbf{h}_{T+\tau}^{\mathcal{X}}$. It aims to maximize the conditional probability $p(\mathcal{F}|\mathcal{P})$ of future value $\mathcal{F}$ given past $\mathcal{P}$, by minimizing the sum of KL-divergence between $q_\phi(\mathbf{Z}|\mathcal{P}, \mathcal{F})$ and $p_\theta(\mathbf{Z}|\mathcal{P})$ and negative log-likelihood of future data $p_\theta(\mathcal{X}_{T+1:T+\tau}|\mathbf{Z}, \mathcal{P})$.

the same timeline, we add a shared feature extractor of time-relevant features to both RNNs. Typical features include absolute time, hour of days, etc. Specifically:

$$
\begin{aligned}
\mathbf{h}_t^{\mathcal{X}} &= f_\phi\left([\varphi(\mathbf{x}_t), \psi(\mathbf{t}_t)], \mathbf{h}_{t-1}^{\mathcal{X}}\right) && \text{for } t = 1 \dots T \\
\mathbf{h}_m^{\mathcal{H}} &= g_\phi\left([\zeta(\mathbf{e}_m), \triangle t_m, \psi(\mathbf{t}_m)], \mathbf{h}_{m-1}^{\mathcal{H}}\right) && \text{for } m = 1 \dots M
\end{aligned}
\tag{6}
$$

where $[a, b]$ denotes the concatenation of a and b, $\triangle t_m = t_m - t_{m-1}$ denotes the inter-event duration, and $\mathbf{e}$ denotes one-hot coded event type vector. The bold $\mathbf{t}_t$ denotes the vector of temporal features, and ψ is the shared feature extractor, parameterized by MLP. φ and ζ are feature extractors of time-series and events, also parameterized by MLP. Both RNNs are initialized with zero vector $\mathbf{h}$.

Auxiliary Transition Unit. When modeling very long sequences, a typical practice is to split time-series into chunks that are overlapped at the time axis (Salinas et al. 2019b). However, with the introduction of event sequences, such an approach causes a problem. Because events occur irregularly, many adjacent chunks may share the same set of events, where the event type, timing, and all other features are the same. However, the forecast targets can be different for the adjacent chunks, which means that the same event sequence input may reflect different targets, making it difficult for the model to capture features within the event sequence. To solve the problem, we introduce Auxiliary Transition Unit, whose basic idea is to let the asynchronous RNN know the exact end time as the time-series RNN, which is done by an auxiliary transit at T with zero event type input:

$$
\mathbf{h}_T^{\mathcal{H}} = g_\phi\left([\zeta(\mathbf{0}), \triangle t_T, \psi(\mathbf{t}_T)], \mathbf{h}_M^{\mathcal{H}}\right)
\tag{7}
$$

where $\mathbf{0}$ denotes zero vector which has the same size as $\mathbf{e}$,

and $\triangle t_T = t_T - t_M$. By doing so, our model can be trained in batch effortlessly, without worrying about data conflicts.

Synergetic Layer. At time T, we have extracted features $\mathbf{h}_T^{\mathcal{X}}$ and $\mathbf{h}_T^{\mathcal{H}}$ by two RNNs from heterogeneous sequences. Then the problem is to parameterize the recognition model $q_\phi(\mathbf{Z}|\mathcal{P}, \mathcal{F})$ and prior model $p_\theta(\mathbf{Z}|\mathcal{P})$. We assume the latent variable follows diagonal Gaussian distribution $\mathbf{Z} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma)$. For the prior model $p_\theta(\mathbf{Z}|\mathcal{P})$, we take the following parameterization:

$$
\begin{aligned}
\boldsymbol{\mu} &= \text{MLP}_\theta\left(\mathbf{h}_T^{\mathcal{X}}, \mathbf{h}_T^{\mathcal{H}}\right), \\
\log(\Sigma) &= \text{diag}\left(\text{MLP}_\theta(\mathbf{h}_T^{\mathcal{X}}, \mathbf{h}_T^{\mathcal{H}})\right)
\end{aligned}
\tag{8}
$$

where the inputs of MLPs are concatenated vector. For the recognition model $q_\phi(\mathbf{Z}|\mathcal{P}, \mathcal{F})$ that depends on both past data $\mathcal{P}$ and future target $\mathcal{F}$, we just need to let the same time-series RNN transit τ more times, and the extracted feature is denoted by $\mathbf{h}_{T+\tau}^{\mathcal{X}}$. The parameterization of recognition model $q_\phi(\mathbf{Z}|\mathcal{P}, \mathcal{F})$ is:

$$
\begin{aligned}
\boldsymbol{\mu} &= \text{MLP}_\phi\left(\mathbf{h}_{T+\tau}^{\mathcal{X}}, \mathbf{h}_T^{\mathcal{H}}\right), \\
\log(\Sigma) &= \text{diag}\left(\text{MLP}_\phi(\mathbf{h}_{T+\tau}^{\mathcal{X}}, \mathbf{h}_T^{\mathcal{H}})\right)
\end{aligned}
\tag{9}
$$

Stochastic RNN Generation Model. Given sampled $\mathbf{Z} \sim q_\phi(\mathbf{Z}|\mathcal{P}, \mathcal{F})$ and inferred $\mathbf{h}_T^{\mathcal{X}}$ and $\mathbf{h}_T^{\mathcal{H}}$, the task becomes how to parameterize $p_\theta(\mathcal{F}|\mathcal{P}, \mathbf{Z})$. We do not assume any particular distribution of the generated data $\mathbf{x}$, but instead we assume it follows a parametric distribution $\mathcal{PD}$ such as ones in the exponential family, and it has sufficient statistic $\mathbf{T}$. Specifically, we initialize an RNN d_θ with:

$$
\mathbf{h}_T^{\mathcal{F}} = \text{MLP}_\theta^1\left([\mathbf{h}_T^{\mathcal{X}}, \mathbf{h}_T^{\mathcal{H}}, \text{MLP}_\theta^2(Z)]\right)
\tag{10}
$$

where the inner MLP^2 is a simple feed-forward network to map sampled $\mathbf{Z}$ as a concatenation with hidden state $\mathbf{h}$.

Algorithm 1: Forecasting by Monte-Carlo sampling

Input: Heterogeneous past data $\mathcal{P} = \{\mathcal{X}_{1:T}, \mathcal{H}_T\}$;
Input: Trained model $p_\theta(\mathcal{F}|\mathcal{P}, \mathbf{Z})$ and $p_\theta(\mathbf{Z}|\mathcal{P})$;
Input: Forecast horizon τ and number of samples N;

1   Evaluate Eq. 6 to get $\mathbf{h}_T^{\mathcal{X}}$ and $\mathbf{h}_T^{\mathcal{H}}$;
2   Compute $p_\theta(\mathbf{Z}|\mathcal{P}) = \mathcal{N}(\boldsymbol{\mu}, \Sigma)$ by Eq. 9;
3   **for** $n = 1$ *to* $n = N$ **do**
4   Sample $\mathbf{Z} \sim \mathcal{N}(\mu, \Sigma)$;
5     $\mathbf{h}_T^{\mathcal{F}} = \left[\mathbf{h}_T^{\mathcal{X}}, \mathbf{h}_T^{\mathcal{H}}, \mathrm{MLP}(Z)\right]$;
6   **for** $t = T + 1$ *to* $T + \tau$ **do**
7       $\mathbf{h}_t^F = d_\theta\left([\varphi(\mathbf{x}_{t-1}), \psi(\mathbf{t}_t)], \mathbf{h}_{t-1}^F\right)$;
8       $\mathbf{x}_t^{(n)} \sim \mathcal{PD}\left(\mathbf{T}_\theta(\mathbf{h}_t^F)\right)$;

9   Compute mean and quantiles from sampled $\mathbf{x}_t^{(n)}$,
    $n = 1 \ldots N$ for each time t;

The outside MLP[1] is to map the concatenated feature to the hidden dimension of the decoder RNN. Then, we iteratively compute from $T + 1$ to $T + \tau$:

$$\mathbf{h}_t^F = d_\theta\left([\varphi(\mathbf{x}_{t-1}), \psi(\mathbf{t}_t)], \mathbf{h}_{t-1}^F\right), \quad \mathbf{x}_t \sim \mathcal{PD}\left(\mathbf{T}_\theta(\mathbf{h}_t^F)\right) \tag{11}$$

where φ and ψ are the shared feature extractor of observation and temporal features respectively. Note that $\mathbf{x}_T$ is true value from past data $\mathcal{X}$.

For $\mathbf{T}_\theta(\mathbf{h}_t^{\mathcal{F}})$, we also use networks to parameterize the mapping. In order to constrain real-valued output $\mathbf{h}_t^{\mathcal{F}}$ to the parameter domain, we use the following transformations:

- Real-valued parameters: no transformation.

- Positive parameters: the SoftPlus function.

- Bounded parameters $[a, b]$: scale and shifted Sigmoid function $y = (b - a)\frac{1}{1 + \exp(-\tilde{y})} + a$

So far, the recognition model $q_\phi(\mathbf{Z}|\mathcal{P}, \mathcal{F})$, prior model $p_\theta(\mathbf{Z}|\mathcal{P})$, and generation model $p_\theta(\mathcal{F}|\mathcal{P}, \mathbf{Z})$ are fully specified. They can be trained by Eq. 5 w.r.t θ and ϕ.

Forecasting with Confidence Intervals

Once the model is learned, we can use the model to address the forecasting problem addressed in Eq. 1 for new coming time-series and event sequences. The straightforward way is to perform a deterministic inference without sampling $\mathbf{Z}$, i.e., $\mathcal{F}^* = \arg\max_{\mathcal{F}} p_\theta(\mathcal{F}|\mathcal{P}, \mathbf{Z}^*)$, $Z^* = \mathbb{E}[\mathbf{Z}|\mathcal{P}]$. However, since our transition model is an RNN, which is non-linear, computing of multi-horizon future values is difficult. We would like to make probabilistic forecast, so we resort to a Monte-Carlo sampling approach as Algorithm 1.

Experiments

To provide evidence for the practical effectiveness of our proposed VSMHN model, we conduct experiments with real-world datasets. We also compare to a wide range of machine learning models for multi-horizon forecasting.

We implement our model by Pytorch on a single RTX-2080Ti GPU. We set the dimension of hidden layers of all MLPs and RNNs in the generative model and inference model to 100, and the dimension of stochastic latent variable $\mathbf{Z}$ to 50. We set sample size K of the objective Eq. 5 to 5. We use Adam optimizer with learning rate 0.001. Forecast distribution is estimated by 1000 trails of Monte Carlo sampling. We use Gaussian observation model (Eq. 11) where mean and standard deviation are parameterized by two layer NNs. Additional details about hyper-parameter optimization are given in the supplementary materials.

Datasets and Protocols

We use two univariate datasets and a multivariate dataset. For all datasets, we extract 4 temporal features: absolute time, hour-of-day, day-of-week, and month-of-year for hourly-sampled time-series data. We held-out the last four weeks for evaluation, and the rest for training. We train the model to predict the future day (24 data points) given the past week (168 points), along with events within that week.

Electricity. The UCI household electricity dataset contains time-series of the electricity usage in kW recorded hourly for 370 clients. We work on an univariate series of length 21044, and time points with fluctuations larger than 30 are extracted as events, where the event types are up and down according to the direction.

Traffic. The dataset corresponds to hourly-sampled road occupancy rates in percentiles (0-100%) from the California Department of Transportation, we work on the first univariate series of length 17544. We extract time points with fluctuations larger than 10% as events, whose types are up and down according to the direction.

Environment. A public air quality multivarite dataset (Li et al. 2019a). We are interested in how atmospheric variables interact with weather events. We use four hourly-sampled variables: PM2.5, dew point, temperature and pressure. And we extract three types of events from minute-level data sources: wind start, wind stop and rain start.

Data Preprocessing. Given full training time-series $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_T\}$ and alongside event sequence $\mathcal{H}_T$, where T is large, we need to generate multiple training samples $\{\mathcal{P}^{(i)}, \mathcal{F}^{(i)}\}_i i = 1^N$. For the time-series part, we follow the protocol used in (Salinas et al. 2019b): The *shingling* technique is to convert long time-series into multiple chunks. Specifically, at each training iteration, we sample a batch of windows with width W at random start point $[1, T - W + 1]$ from the dataset and feed into the model. For the event sequence part, we first difference the timestamps to create inter-event duration feature $\triangle t$, and then join the event sequence to time-series chunks by the shared time index. When training, event sequences are padded ahead with zeros to the same length as a common practice in neural point process models (Xiao et al. 2016). Moreover, we generate four time features (similar to positional encoding mechanism in Transformer (Vaswani et al. 2017)): *absolute-time*, *hour-of-day*, *day-of-week* and *month-of-year* using the observation time stamps. To facilitate training, each of the variables in the time series is scaled to $\mathcal{N}(0, 1)$ respectively to make their log likelihood comparable.

Method	Event used?	PM2.5	Dewpoint	Temperature	Pressure	Electricity	Traffic
DeepAR		85.93/45.57	6.27/3.48	2.65/1.47	50.16/39.92	23.9/11.34	1.52/0.71
DeepAR-E	✓	88.82/48.44	5.86/3.21	3.10/1.79	35.95/28.00	25.23/12.50	1.55/0.70
MQRNN		116.94/70.48	8.72/6.86	3.46/2.35	6.88/4.81	27.00/14.01	1.50/0.74
DeepFactor		145.67/112.58	10.02/7.80	4.98/3.97	6.33/5.05	29.38/15.56	2.32/1.59
VAR		88.39/46.18	**3.57**/2.01	3.09/1.70	2.91/1.93	21.62/11.92	1.73/1.43
ETS		107.06/55.34	4.26/2.19	3.15/1.81	3.19/1.87	21.14/11.51	2.62/2.95
SARIMA		92.44/47.57	3.98/2.06	2.85/1.68	4.58/2.50	22.69/12.05	1.70/0.86
VSMHN-TS (ours)		78.47/36.98	4.12/2.05	2.16/1.20	2.91/1.53	20.51/9.21	1.68/0.53
VSMHN (ours)	✓	**72.84/33.67**	3.59/**1.94**	**2.10/1.19**	**2.47/1.33**	**19.08/8.86**	**1.36/0.48**

Table 1: RMSE/CRPS comparison of rolling-day forecast of last 28 days. Tick denotes event information is used in the model.

Compared Baselines. We compare three network models.

DeepAR (Salinas et al. 2019b), an RNN-based autoregressive likelihood model, which is essentially an univariate model. Multivariate time-series is trained jointly with static category features. It can learn correlations of target time-series with synchronous exogenous sequences, which are assumed to be known in the forecast horizon.

DeepFactor (Wang et al. 2019), a deep global-local model, in which the global model is an RNN, and the local model is a probabilistic time-series model such as Gaussian Processes or linear dynamical system.

MQRNN (Wen et al. 2017), a sequence-to-sequence model with LSTM encoder and decoder. The model is trained with quantile loss to learn uncertainties.

Neural network-based baseline are implemented by *Gluonts* python package, which is based on MXNet, and hyperparameters are tuned using grid search. Moreover, to evaluate the influence of event input, we set DeepAR-E, which takes dummy-coded event sequences as exogenous variables that are unknown in the forecast horizon. We also use VSMHN-TS i.e. a pure time-series model that takes no events input.

Recent study (Makridakis, Spiliotis, and Assimakopoulos 2018) shows that statistical models are still competitive in time series forecasting, of which three are compared.

Vector Autoregression (VAR), a model that learns a linear mapping between the value-vectors and their lagged value-vectors in multi-variate time series data.

Seasonal Autoregressive Integrated Moving Average (SARIMA), an additive model with autoregressive, differencing ,moving average and seasonal terms.

Exponential Smoothing (ETS), a model using exponential functions to weight past observations in evolution functions with exponentially decreasing weights over time. Particularly, we use Holt's Winters Seasonal Exponential Smoothing with an additive trend component, an additive seasonal component and an additive error component.

The VAR and ETS models are implemented using *statmodels* pyhton package. The SARIMA model is implemented using *pmdarima* python package. The seasonal period hyperparameter of ETS and SARIMA is set to 24 since the used data are hourly sampled. Other hyper-parameters are tuned by grid search according to Akaike Information Criterion (AIC) on training data.

Evaluation Metric CRPS is calculated analytically by the *properscoring* python package. We calculate Root Mean

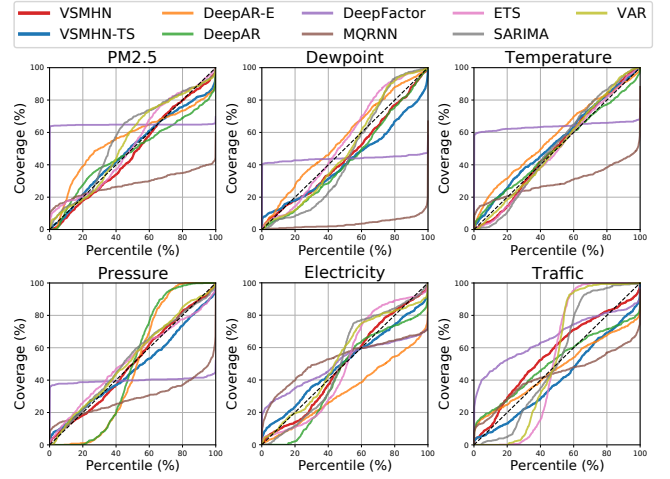

Figure 3: Uncertainty calibration. Perfect calibration corresponds to Coverage$(p) = p$ in diagonal black dotted line.

Square Error (RMSE) using the mean of forecast distribution. All scores are calculated on the original scale of data.

We use Gaussian observation model, and CRPS can be computed exactly (Gneiting et al. 2005). Consider the predicted distribution is $\mathcal{N}(\mu, \sigma^2)$, the CRPS score is given by:

$$\text{crps}[\mathcal{N}(\mu, \sigma^2), x] =$$
$$\sigma\left\{\frac{x-\mu}{\sigma}\left[2\Phi(\frac{x-\mu}{\sigma}) - 1\right] + 2\varphi(\frac{x-\mu}{\sigma}) - \frac{1}{\sqrt{\pi}}\right\} \quad (12)$$

where $\Phi(\frac{x-\mu}{\sigma})$ and $\varphi(\frac{x-\mu}{\sigma})$ denote PDF and CDF, respectively, of the normal distribution with mean 0 and variance 1. For time series with M variables, the average is:

$$\text{CRPS} = \frac{1}{M}\sum_i \text{crps}(F_i, x_i) \quad (13)$$

Note that the CRPS score reduces to MAE if each F_i is a deterministic forecast.

The RMSE is defined as the rooted mean squared error over all time series, i.e., $i = 1, \ldots, M$, and over the forecast horizon, i.e. $t = T + 1, T + 2, \ldots, T + \tau$:

$$\text{RMSE} = \sqrt{\frac{1}{M \times \tau}\sum_{i,t}(x_{i,t} - \hat{x}_{u,t})^2} \quad (14)$$

where x is the true value in the forecast horizon, and $\hat{x}$ is the predicted distribution mean.

Results and Discussion

Overall accuracy. The RMSE and CRPS results are shown in Table 1. Our proposed model (VSMHN with event input) beats state-of-the-art baselines on five of six variables from three datasets with respect to RMSE score. Our model outperforms all baselines on CRPS score notably, suggesting our model can provide sharp and accurate probabilistic forecast.

Uncertainty calibration. For probabilistic forecast, we obtain not only values but also their probability distributions to assist decision-making. It is important to measure how well the forecast distribution is calibrated against the forecast error. We use coverage metric $\text{Coverage}(p)$, where p is the percentile of the distribution. $\text{Coverage}(p)$ is the frequency of actual values lying within percentile p. Being closer to the perfect calibration line $\text{Coverage}(p) = p$ means better uncertainty estimation. Figure 3 shows the calibration curves of the forecast distribution. To quantify uncertainty calibration performance, we also compute the R-squared (R^2) values between the models' uncertainty calibration curves and the perfect calibration line, as shown in Table 2. In conclusion, compared with various baselines on six variables, our model provides better uncertainty calibration.

Impact of event sequence. To illustrate the influence of event sequence input to time-series, we convert dummy coded event sequence to time-series, and let DeepAR take it as exogenous input. We find event inputs can slightly improve the performance of DeepAR on some datasets. For our proposed VSMHN model, event sequence input constantly improves the performance, indicating that our model extracts valuable features from event sequence. Those features can help the model explain fluctuations within time-series in the training phase, thus improve overall forecast accuracy.

Impact of model architecture. Surprisingly, we find that our model is also ahead of baselines in the absence of event inputs, which means it is still advanced even for conventional multi-horizon time-series forecasting problem as in (Salinas et al. 2019b). We find that it is difficult for DeepAR to forecast pressure over other variables as shown in Table 1, which we think is because that as a univariate model, DeepAR cannot handle multivariate time-series with a wide range of scales properly. In contrast, our model is multivariate and performs stable on variables with different scales.

Performance of traditional statistical models. In Table 1, traditional statistical models are competitive. However, simple statistical models struggle to extract features from complex external information or do cross-learning from multivariate data. As shown in (Bojer and Meldgaard 2020), successful integration of external information can be key to accurate forecast on complex time series datasets.

Forecast visualization. Figure 4 gives examples of probabilistic forecast results for each of the six variables in our dataset, we can see that VSMHN makes accurate and sharp probabilistic forecast overall. For each of the forecast horizon, the forecast uncertainty grows over time naturally. More importantly, the model spontaneously places higher uncertainty when forecast may not be accurate, e.g. $t = 350$ for PM2.5 (top left), $t = 220$ for Dew point (top right), and $t = 170$ for Electricity (bottom left), which is a desired feature for downstream decision making processes.
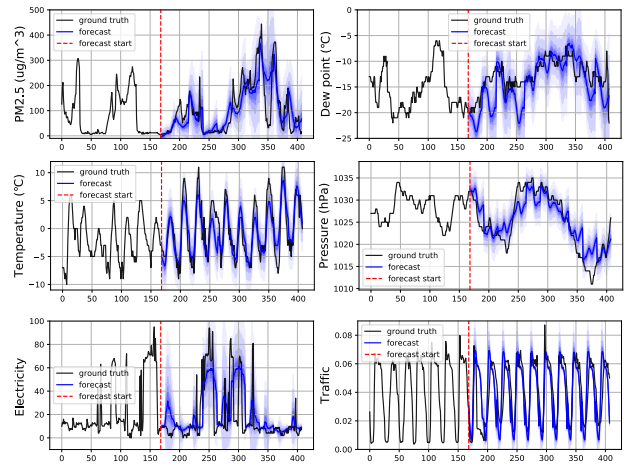

Figure 4: Rolling-day probabilistic forecast by VSMHN. Red vertical dashed lines show the start time of the forecast. Probability densities are plot by confidence intervals [20%, 30%,50%, 80%, 95%] in transparency. The past series are not shown in full length. The four variables from the first two rows i.e. PM2.5, Dew Point, Temperature, Pressure come from a multivariate dataset: Air Quality, and they are jointly modeled to obtain the shown results.

Method	PM2.5	Dewpoint	Temperature	Pressure	Electricity	Traffic
DeepAR	0.924	0.928	**0.988**	0.679	0.879	0.907
DeepAR-E	0.822	0.946	0.910	0.689	0.581	0.889
MQRNN	-0.12	0.062	-0.081	-0.022	0.738	0.436
DeepFactor	-0.235	0.062	-0.081	-0.022	0.738	0.436
VAR	0.944	0.944	**0.988**	0.965	0.923	0.562
ETS	0.969	0.933	**0.988**	0.971	0.885	0.451
SARIMA	0.877	0.863	0.962	0.974	0.904	0.797
VSMHN-TS	0.981	0.877	0.975	0.970	0.970	0.884
VSMHN	**0.987**	**0.971**	**0.988**	**0.995**	**0.984**	**0.918**

Table 2: Quantification comparison by R2 of uncertainty calibration quality in the forecasting horizon.

Conclusion

We have presented a novel approach based on the deep conditional generative model to jointly learn from heterogeneous temporal sequences (i.e. time series and event sequences). To our best knowledge, this is one of the first works to provide a joint modeling framework for multi-horizon probabilistic forecasting. Empirically we have shown that our proposed model outperforms the state-of-the-art forecast models on six variables from three public datasets.

Acknowledgements

Ethics Statement

As far as we know, we are the first to define and address the problem of jointly modeling time-series and event sequences for multi-horizon time-series forecasting. For existing time-series forecasting applications, our research shows that reasonable modeling of related event data can effectively reduce the deviation of time-series forecasting, making downstream predictive analysis and decision-making more consistent with the goal of risk control.

This research has many societal implications. For example, economists can study the impact of political events on economic indicators, formulating more predictive economic policies. Environmentalists can study the impact of events such as volcanic eruptions or industrial activities on the environment to understand climate change. Retailers can analyze the impact of promotions and other activities on sales to arrange stock and inventory.

References

Bacry, E.; Mastromatteo, I.; and Muzy, J.-F. 2015. Hawkes processes in finance. *Market Microstructure and Liquidity* 1(01): 1550005.

Baytas, I. M.; Xiao, C.; Zhang, X.; Wang, F.; Jain, A. K.; and Zhou, J. 2017. Patient subtyping via time-aware LSTM networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, 65–74. ACM.

Benidis, K.; Rangapuram, S. S.; Flunkert, V.; Wang, B.; Maddix, D.; Turkmen, C.; Gasthaus, J.; Bohlke-Schneider, M.; Salinas, D.; Stella, L.; et al. 2020. Neural forecasting: Introduction and literature overview. *arXiv preprint arXiv:2004.10240* .

Biński, M.; Marti, G.; and Donnat, P. 2017. Autoregressive convolutional neural networks for asynchronous time series. *arXiv preprint arXiv:1703.04122* .

Bojer, C. S.; and Meldgaard, J. P. 2020. Kaggle forecasting competitions: An overlooked learning opportunity. *International Journal of Forecasting* .

Box, G. E.; Jenkins, G. M.; Reinsel, G. C.; and Ljung, G. M. 2015. *Time series analysis: forecasting and control*. John Wiley & Sons.

Chandola, V.; Banerjee, A.; and Kumar, V. 2012. Anomaly detection for discrete sequences: A survey. *IEEE Transactions on Knowledge and Data Engineering* 24(5): 823–839.

Che, Z.; Purushotham, S.; Li, G.; Jiang, B.; and Liu, Y. 2018. Hierarchical deep generative models for multi-rate multivariate time series. In *International Conference on Machine Learning*, 783–792.

Du, N.; Dai, H.; Trivedi, R.; Upadhyay, U.; Gomez-Rodriguez, M.; and Song, L. 2016. Recurrent marked temporal point processes: Embedding event history to vector. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1555–1564. ACM.

Fan, C.; Zhang, Y.; Pan, Y.; Li, X.; Zhang, C.; Yuan, R.; Wu, D.; Wang, W.; Pei, J.; and Huang, H. 2019. Multi-horizon time series forecasting with temporal attention learning. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2527–2535. ACM.

Gama, J.; Žliobaitė, I.; Bifet, A.; Pechenizkiy, M.; and Bouchachia, A. 2014. A survey on concept drift adaptation. *ACM computing surveys (CSUR)* 46(4): 1–37.

Gneiting, T.; and Katzfuss, M. 2014. Probabilistic forecasting. *Annual Review of Statistics and Its Application* 1: 125–151.

Gneiting, T.; Raftery, A. E.; Westveld III, A. H.; and Goldman, T. 2005. Calibrated probabilistic forecasting using ensemble model output statistics and minimum CRPS estimation. *Monthly Weather Review* 133(5): 1098–1118.

Hyndman, R.; Koehler, A. B.; Ord, J. K.; and Snyder, R. D. 2008. *Forecasting with exponential smoothing: the state space approach*. Springer Science & Business Media.

Kingma, D.; and Welling, M. 2013. Auto-encoding variational Bayes. In *arXiv:1312.6114*.

Li, L.; Yan, J.; Yang, X.; and Jin, Y. 2019a. Learning interpretable deep state space model for probabilistic time series forecasting. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence*, 2901–2908. AAAI Press.

Li, S.; Jin, X.; Xuan, Y.; Zhou, X.; Chen, W.; Wang, Y.-X.; and Yan, X. 2019b. Enhancing the locality and breaking the memory bottleneck of transformer on time series forecasting. In *Advances in Neural Information Processing Systems*, 5244–5254.

Lütkepohl, H. 2005. *New introduction to multiple time series analysis*. Springer Science & Business Media.

Makridakis, S.; Spiliotis, E.; and Assimakopoulos, V. 2018. The M4 Competition: Results, findings, conclusion and way forward. *International Journal of Forecasting* 34(4): 802–808.

Mariet, Z.; and Kuznetsov, V. 2019. Foundations of Sequence-to-Sequence Modeling for Time Series. In *The 22nd International Conference on Artificial Intelligence and Statistics*, 408–417.

Mei, H.; and Eisner, J. M. 2017. The neural hawkes process: A neurally self-modulating multivariate point process. In *Advances in Neural Information Processing Systems*, 6754–6764.

Rangapuram, S. S.; Seeger, M. W.; Gasthaus, J.; Stella, L.; Wang, Y.; and Januschowski, T. 2018. Deep state space models for time series forecasting. In *Advances in Neural Information Processing Systems*, 7785–7794.

Salinas, D.; Bohlke-Schneider, M.; Callot, L.; Medico, R.; and Gasthaus, J. 2019a. High-dimensional multivariate forecasting with low-rank Gaussian Copula Processes. In *Advances in Neural Information Processing Systems*, 6824–6834.

Salinas, D.; Flunkert, V.; Gasthaus, J.; and Januschowski, T. 2019b. DeepAR: Probabilistic forecasting with autoregressive recurrent networks. *International Journal of Forecasting* .

Sohn, K.; Lee, H.; and Yan, X. 2015. Learning structured output representation using deep conditional generative models. In *Advances in neural information processing systems*, 3483–3491.

Sutskever, I.; Vinyals, O.; and Le, Q. 2014. Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems* .

Taieb, S. B.; and Atiya, A. F. 2015. A bias and variance analysis for multistep-ahead time series forecasting. *IEEE transactions on neural networks and learning systems* 27(1): 62–76.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

Vincent, L.; and Thome, N. 2019. Shape and time distortion loss for training deep time series forecasting models. In *Advances in Neural Information Processing Systems*, 4191–4203.

Wang, Y.; Smola, A.; Maddix, D. C.; Gasthaus, J.; Foster, D.; and Januschowski, T. 2019. Deep Factors for Forecasting. *arXiv preprint arXiv:1905.12417* .

Wen, R.; Torkkola, K.; Narayanaswamy, B.; and Madeka, D. 2017. A multi-horizon quantile recurrent forecaster. *arXiv preprint arXiv:1711.11053* .

Wu, W.; Yan, J.; Yang, X.; and Zha, H. 2018. Decoupled Learning for Factorial Marked Temporal Point Processes. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2516–2525. ACM.

Xiao, S.; Yan, J.; Farajtabar, M.; Song, L.; Yang, X.; and Zha, H. 2019. Learning time series associated event sequences with recurrent point process networks. *IEEE transactions on neural networks and learning systems* 30(10): 3124–3136.

Xiao, S.; Yan, J.; Li, C.; Jin, B.; Wang, X.; Yang, X.; Chu, S. M.; and Zha, H. 2016. On Modeling and Predicting Individual Paper Citation Count over Time. In *IJCAI*, 2676–2682.

Xiao, S.; Yan, J.; Yang, X.; Zha, H.; and Chu, S. M. 2017. Modeling the intensity function of point process via recurrent neural networks. In *Thirty-First AAAI Conference on Artificial Intelligence*.