Deep Factors for Forecasting

Yuyang Wang [1]   Alex Smola [1]   Danielle C. Maddix [1]   Jan Gasthaus [1]   Dean Foster [1]   Tim Januschowski [1]

Abstract

Producing probabilistic forecasts for large collections of similar and/or dependent time series is a practically relevant and challenging task. Classical time series models fail to capture complex patterns in the data, and multivariate techniques struggle to scale to large problem sizes. Their reliance on strong structural assumptions makes them data-efficient, and allows them to provide uncertainty estimates. The converse is true for models based on deep neural networks, which can learn complex patterns and dependencies given enough data. In this paper, we propose a hybrid model that incorporates the benefits of both approaches. Our new method is data-driven and scalable via a latent, global, deep component. It also handles uncertainty through a local classical model. We provide both theoretical and empirical evidence for the soundness of our approach through a necessary and sufficient decomposition of exchangeable time series into a global and a local part. Our experiments demonstrate the advantages of our model both in term of data efficiency, accuracy and computational complexity.

1. Introduction

Time series forecasting is a key ingredient in the automation and optimization of business processes. In retail, decisions about which products to stock, when to (re)order them, and where to store them depend on forecasts of future demand in different regions; in (cloud) computing, the estimated future usage of services and infrastructure components guides capacity planning; regional forecasts of energy consumption are used to plan and optimize the generation of power; and workforce scheduling in warehouses and factories depends on forecasts of the future workload.

The prevalent forecasting methods in statistics and econo-

metrics have been developed in the context of forecasting individual or small groups of time series. The core of these methods is formed by comparatively simple (often linear) models, which require manual feature engineering and model design by domain experts to achieve good performance (Harvey, 1990). Recently, there has been a paradigm shift from these model-based methods to fully-automated data-driven approaches. This shift can be attributed to the availability of large and diverse time series datasets in a wide variety of fields, e.g. energy consumption of households, server load in a data center, online user behavior, and demand for all products that a large retailer offers. These large datasets make it possible and necessary to learn models from data without significant manual work (Böse et al., 2017).

A collection of time series can exhibit various dependency relationships between the individual time series that can be leveraged in forecasting. These include: (1) local co-variate relationships (e.g. the price and demand for a product, which tend to be (negatively) correlated), (2) indirect relationships through shared latent causes (e.g. demand for multiple products increasing because an advertising campaign is driving traffic to the site), (3) subtle dependencies through smoothness, temporal dynamics, and noise characteristics of time series that are measurements of similar underlying phenomena (e.g. product sales time series tend to be similar to each other, but different from energy consumption time series). The data in practical forecasting problems typically has all of these forms of dependencies. Making use of this data from related time series allows more complex and potentially more accurate models to be fitted without overfitting.

Classical time series models have been extended to address the above dependencies of types (1) and (2) by allowing exogenous variables (e.g. the ARIMAX model and control inputs in linear dynamical systems), and employing multivariate time series models that impose a certain covariance structure (dynamic factor models), respectively. Neural network-based models have been recently shown to excel in extracting complex patterns from large datasets of related time series by exploiting similar smoothness and temporal dynamics, and common responses to exogenous input, i.e. dependencies of type (1) and (3) (Flunkert et al., 2017; Wen et al., 2017; Mukherjee et al., 2018; Gasthaus et al., 2019). These models struggle in producing calibrated uncertainty

---

[1]Amazon Research. Correspondence to: Yuyang Wang <yuyawang@amazon.com>.

estimates. They can also be sample-inefficient, and cannot handle type (2) dependencies. See (Faloutsos et al., 2018) for a recent survey on traditional and modern methods for forecasting.

The two main challenges that arise in the fully-automated data-driven approaches are: how to build statistical models that are able to borrow statistical strength and effectively learn to forecast from large and diverse data sources exhibiting all forms of dependencies, and how to combine the data efficiency and uncertainty characterization of classical time series models with the expressive power of deep neural networks. In this paper, we propose a family of models that efficiently (in terms of sample complexity) and effectively (in terms of computational complexity) addresses these aforementioned challenges.

1.1. Background

Classical time series models, such as general State-Space Models (SSMs), including ARIMA and exponential smoothing, excel at modeling the complex dynamics of individual time series of sufficiently long history. For Gaussian State-Space Models, these methods are computationally efficient, e.g. via a Kalman filter, and provide uncertainty estimates. Uncertainty estimates are critical for optimal downstream decision making. Gaussian Processes (Rasmussen & Williams, 2006; Seeger, 2004) (GPs) are another family of the models that have been applied to time series forecasting (Girard et al., 2003; Brahim-Belhouari & Bermak, 2004). These methods are local, that is, they learn one model per time series. As a consequence, they cannot effectively extract information across multiple time series. Finally, these classical methods struggle with cold-start problems, where more time series are added or removed over time.

Mixed effect models (Crawley, 2012) consist of two kinds of effects: fixed (global) effects that describe the whole population, and random (local) effects that capture the idiosyncratic of individuals or subgroups. A similar mixed approach is used in Hierarchical Bayesian (Gelman et al., 2013) methods, which combine global and local models to jointly model a population of related statistical problems. In (Ahmed et al., 2012; Low et al., 2011), other combined local and global models are detailed.

Dynamic factor models (DFMs) have been studied in econometrics for decades to model the co-evolvement of multiple time series (Geweke, 1977; Stock & Watson, 2011; Forni et al., 2000; Pan & Yao, 2008). DFMs can be thought as an extension of principal component analysis in the temporal setting. All the time series are assumed to be driven by a small number of dynamic (latent) factors. Similar to other models in classical statistics, theories and techniques are developed with assuming that the data is normally distributed and stationary. Desired theoretical properties are often lost

when generalizing to other likelihoods. Closely related are the matrix factorization (MF) techniques (Yu et al., 2016; Xie et al., 2017; Hooi et al., 2019) and tensor factorization (Araujo et al., 2019), which have been applied to the time series matrix with temporal regularization to ensure the regularity of the latent time series. These methods are not probabilistic in nature, and cannot provide uncertainty estimation for non-Gaussian observations.

1.2. Main Contributions

In this paper, we propose a novel global-local method, Deep Factor Models with Random Effects. It is based on a global DNN backbone and local probabilistic graphical models for computational efficiency. The global-local structure extracts complex non-linear patterns globally while capturing individual random effects for each time series locally.

The main idea of our approach is to represent each time series, or its latent function, as a combination of a global time series and a corresponding local model. The global part is given by a linear combination of a set of deep dynamic factors, where the loading is temporally determined by attentions. The local model is stochastic. Typical local choices include white noise processes, linear dynamical systems (LDS) or Gaussian processes (GPs). The stochastic local component allows for the uncertainty to propagate forward in time. Our contributions are as follows: i) Provide a unique characterization of exchangeable time series (Section 2); ii) Propose a novel global-local framework for time series forecasting, based on i), that systematically marries deep neural networks and probabilistic models (Section 3); iii) Develop an efficient and scalable inference algorithm for non-Gaussian likelihoods that is generally applicable to any normally distributed probabilistic models, such as SSMs and GPs (Section 3). As a byproduct, we obtain new approximate inference methods for SSMs/GPs with non-Gaussian likelihoods; iv) Show how state-of-the-art time series forecasting methods can be subsumed in the proposed framework (Section 4); v) Demonstrate the accuracy and data efficiency of our approach through scientific experiments (Section 5).

2. Exchangeable Series

In this section, we formulate a general model for exchangeable time series. A distribution over objects is exchangeable, if the probability of the objects is invariant under any permutation. Exchangeable time series are a common occurrence. For instance, user purchase behavior is exchangeable, since there is no specific reason to assign a particular coordinate to a particular user. Other practical examples include sales statistics over similar products, prices of securities on the stock market and the use of electricity.

2.1. Characterization

Let $z_i \in \mathcal{Z}^T$, where z_i denotes the ith exchangeable time series, $\mathcal{Z}$ denotes the domain of observations and $T \in \mathbb{N}$ denotes the length of the time series.[1] We denote individual observations at some time t as $z_{i,t}$. We assume that we observe $z_{i,t}$ at discrete time steps to have a proper time series rather than a marked point process.

Theorem 1. *Let p be a distribution over exchangeable time series z_i over $\mathcal{Z}$ with length T, where $1 \leq i \leq N$. Then p admits the form*

$$p(z) = \int \prod_{t=1}^{T} p(g_t|g_{1:t-1}) \prod_{i=1}^{N} p(z_{i,t}|z_{i,1:t-1}, g_t) dg.$$

In other words, $p(z)$ decomposes into a global time series g and N local times series z_i, which are conditionally independent given the latent series g.

Proof. It follows from de Finetti's theorem (Diaconis, 1977; Diaconis & Freedman, 1980) that

$$p(z) = \int p(g) \prod_{i=1}^{N} p(z_i|g) dg. \tag{1}$$

Since z_i are time series, we can decompose $p(z_i|g)$ in the causal direction using the chain rule as

$$p(z_i|g) = \prod_{t=1}^{T} p(z_{i,t}|z_{i,1:t-1}, g).$$

Substituting this into the de Finetti factorization in Eqn. (1) gives

$$p(z) = \int p(g) \prod_{i=1}^{N} \prod_{t=1}^{T} p(z_{i,t}|z_{i,1:t-1}, g) dg.$$

Lastly, we can decompose g, such that g_t contains a sufficient statistic of g with respect to $z_{\cdot,t}$. This holds trivially by setting $g_t = g$, but defeats the purpose of the subsequent models. Using the chain rule on $p(g)$ and substituting the result in proves the claim. □

Theorem 2. *For tree-wise exchangeable time series, that is time series that can be grouped hierarchically into exchangeable sets, there exists a corresponding set of hierarchical latent variable models.*

The proof is analogous to that of Theorem 1, and follows from a hierarchical extension of de Finetti's theorem (Austin & Panchenko, 2014). This decomposition is useful when dealing with product hierarchies. For instance, the sales events within the category of iPhone charger cables and the category of electronics cables may be exchangeable.

---

[1]Without loss of generality and to avoid notational clutter, we omit the extension to time series beginning at different points of time. Our approach is general and covers these cases as well.

2.2. Practical Considerations

We now review some common design decisions used in modeling time series. The first is to replace the decomposition $\prod_{t=1}^{T} p(z_{i,t}|z_{i,1:t-1})$ by a tractable, approximate statistic h_t of the past, such that $p(z_{i,t}|z_{i,1:t-1}) \approx p(z_{i,t}|h_{i,t})$. Here, h_t typically assumes the form of a latent variable model via $p(h_{i,t}|h_{i,t-1}, z_{i,t-1})$. Popular choices for real-valued random variables are SSMs and GPs.

The second is to assume that the global variable g_t is drawn from some $p(g_t|g_{t-1})$. The inference in this model is costly, since it requires constant interaction, via Sequential Monte Carlo, variational inference or a similar procedure between local and global variables at prediction time. One way to reduce these expensive calculations is to incorporate past local observations $z_{\cdot,t-1}$ explicitly. While this somewhat negates the simplicity of Theorem 1, it yields significantly higher accuracy for a limited computational budget, $g_t \sim p(g_t|g_{t-1}, z_{\cdot,t-1})$.

Lastly, the time series often comes with observed covariates, such as a user's location or a detailed description of an item being sold. We add these covariates $x_{i,t}$ to the time series signal to obtain the following model:

$$p(z|x) = \int \prod_{t=1}^{T} p(g_t|g_{t-1}, x_{\cdot,t}, z_{\cdot,t-1}) \times \tag{2}$$
$$\prod_{i=1}^{N} \Big[p(h_{i,t}|g_t, h_{i,t-1}, z_{i,t-1}, x_{i,t}) \times$$
$$p(z_{i,t}|g_t, h_{i,t}, z_{i,t-1}, x_{i,t}) \Big] dg dh.$$

Even though this model is rarely used in its full generality, Eqn. (2) is relevant because it is by some measure the most general model to consider, based on the de Finetti factorization in Theorem 1.

2.3. Special Cases

The global-local structure has been used previously in a number of special contexts (Xu et al., 2009; Ahmadi et al., 2011; Hwang et al., 2016; Choi et al., 2011). For instance, in Temporal LDA (Ahmed et al., 2012) we assume that we have a common fixed Dirichlet-Multinomial distribution capturing the distribution of tokens per topic, and a time-variant set of latent random variables capturing the changes in user preferences. This is a special case of the above model, where the global time series does not depend on time, but is stationary instead.

A more closely related case is the Neural Survival Recommender model of (Jing & Smola, 2017). This models the temporal dynamics of return times of a user to an app via survival analysis. In particular, it uses a LSTM for the global dynamics and LSTMs for the local survival probabilities. In

this form, it falls into the category of models described by Theorem 1. Unlike the models we propose in this paper, it does not capture local uncertainty accurately. It also primarily deals with point processes rather than proper time series, and the inference algorithm differs quite significantly.

3. Deep Factor Models with Random Effects

Motivated by the characterization of exchangeable time series, in this section, we propose a general framework for global-local forecasting models, called Deep Factor Models with Random Effects, that follows the structure given by the decomposition in Theorem 1. We describe the family of the methods, show three concrete instantiations (DF-RNN, DF-LDS, and DF-GP), and derive the general inference and learning algorithm. Further models that can be obtained within the same framework, and additional details about the design choices, are described in Appendix A.

We are given a set of N time series, with the ith time series consisting of tuples $(\boldsymbol{x}_{i,t}, z_{i,t}) \in \mathbb{R}^d \times \mathbb{R}, t = 1, \cdots, T$, where $\boldsymbol{x}_{i,t}$ are the input co-variates, and $z_{i,t}$ is the corresponding observation at time t. Given a forecast horizon $\tau \in \mathbb{N}^+$, our goal is to calculate the joint predictive distribution of future observations,

$$p(\{z_{i,T+1:T+\tau}\}_{i=1}^N | \{\boldsymbol{x}_{i,1:T+\tau}, z_{i,1:T}\}_{i=1}^N),$$

i.e. the joint distribution over future observations given all co-variates (including future ones) and past observations.

3.1. Generative Model

Our key modeling assumption is that each time series $z_{i,t}, t = 1, 2, \ldots$ is governed by a fixed global (non-random) and a random component, whose prior is specified by a generative model $\mathcal{R}_i$. In particular, we assume the following generative process:

$$\text{global factors}: \quad g_k(\cdot) = \text{RNN}_k(\cdot), \quad k = 1, \cdots, K,$$

$$\text{fixed effect}: \quad f_i(\cdot) = \sum_{k=1}^K w_{i,k} \cdot g_k(\cdot), \qquad (3)$$

$$\text{random effect}: \quad r_i(\cdot) \sim \mathcal{R}_i, \quad i = 1, \cdots, N, \qquad (4)$$

$$\text{latent function}: \quad u_i(\cdot) = f_i(\cdot) + r_i(\cdot), \qquad (5)$$

$$\text{emission}: \quad z_{i,t} \sim p(z_{i,t} | u_i(\boldsymbol{x}_{i,t})),$$

The observation model p can be any parametric distribution, such as Gaussian, Poisson or Negative Binomial. All the functions $g_k(\cdot), r_i(\cdot), u_i(\cdot)$ take features $\boldsymbol{x}_{i,t}$ as input, and we define $u_{i,t} := u_i(\boldsymbol{x}_{i,t})$, the embedding $\boldsymbol{w}_i := [w_{i,k}]_k$.

3.1.1. GLOBAL EFFECTS (COMMON PATTERNS)

The global effects are given by linear combinations of K latent global deep factors modeled by RNNs. These deep
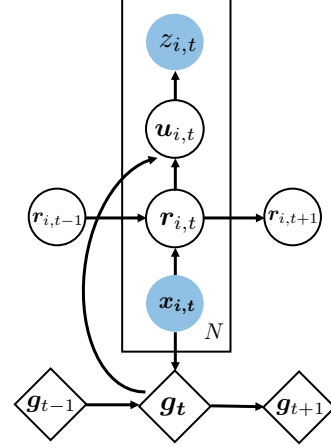

Figure 1. Plate graph of the proposed Deep Factor Model with Random Effects. The diamond nodes represent deterministic states.

factors can be thought of as dynamic principal components or eigen time series that drive the underlying dynamics of all the time series. As mentioned in Section 2.2, we restrict the global effects to be deterministic to avoid costly inference at the global level that depends on all time series.

The novel formulation of the fixed effects from the RNN in Eqn. (3) has advantages in comparison to a standard RNN forecaster. Figure 2 compares the generalization errors and average running times of using Eqn. (3) with the L_2 loss and a standard RNN forecaster with the same L_2 loss and a comparable number of parameters on a real-word dataset electricity (Dheeru & Karra Taniskidou, 2017). Our fixed effect formulation shows significant data and computational efficiency improvement. The proposed model has less variance in comparison to the standard structure. Detailed empirical explorations of the proposed structures can be found in Section 5.1.
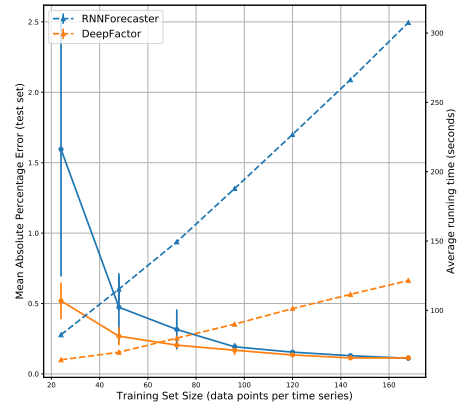

Figure 2. Generalization Error (solid line), Mean Absolute Percentage Error (MAPE) on the test set and running time in seconds (dashed line) vs. the size of the training set, in terms of data points per time series. The experiments are repeated over 10 runs.

NAME	DESCRIPTION	LOCAL	LIKELIHOOD (GAUSSIAN CASE)
DF-RNN	Zero-mean Gaussian noise process given by RNN	$r_{i,t} \sim \mathcal{N}(0, \sigma_{i,t}^2)$	$p(\boldsymbol{z}_i) = \prod_t \mathcal{N}(z_{i,t} - f_{i,t}|0, \sigma_{i,t}^2)$
DF-LDS	State-space models	$r_{i,t} \sim \text{LDS}_{i,t}$ (cf. Eqn. (6))	$p(\boldsymbol{z}_i)$ given by Kalman Filter
DF-GP	Zero-mean Gaussian Process	$r_{i,t} \sim \text{GP}_{i,t}$ (cf. Eqn. (7))	$p(\boldsymbol{z}_i) = \mathcal{N}(\boldsymbol{z}_i - \boldsymbol{f}_i|\boldsymbol{0}, \mathbf{K}_i + \boldsymbol{\sigma}_i^2 \mathbf{I})$

Table 1. Summary table of Deep Factor Models with Random Effects. The likelihood column is under the assumption of Gaussian noise.

3.1.2. RANDOM EFFECTS (LOCAL FLUCTUATIONS)

The random effects $r_i(\cdot)$ in Eqn. (4) can be chosen to be any classical probabilistic time series model $\mathcal{R}_i$. To efficiently compute their marginal likelihood $p(\boldsymbol{z}_i|\mathcal{R}_i)$, $r_{i,t}$ should be chosen to satisfy the normal distributed observation assumption. Table 1 summarizes the three models we consider for the local random effects models $\mathcal{R}_i$.

The first local model, DF-RNN, is defined as $r_{i,t} \sim \mathcal{N}(0, \sigma_{i,t}^2)$, where $\sigma_{i,t}$ is given by a noise RNN that takes input feature $\boldsymbol{x}_{i,t}$. The noise process becomes correlated with the covariance function implicitly defined by the RNN, resulting in a simple deep generative model.

The second local model, DF-LDS, is a part of a special and robust family of SSMs, Innovation State-Space Models (ISSMs) (Hyndman et al., 2008; Seeger et al., 2016). This gives the following generative model:

$$\boldsymbol{h}_{i,t} = \mathbf{F}_{i,t}\boldsymbol{h}_{i,t-1} + \boldsymbol{q}_{i,t}\epsilon_{i,t}, \quad \epsilon_{i,t} \sim \mathcal{N}(0,1),$$
$$r_{i,t} = \boldsymbol{a}_{i,t}^\top \boldsymbol{h}_{i,t}. \tag{6}$$

The latent state $\boldsymbol{h}_{i,t}$ contains information about the level, trend, and seasonality patterns. It evolves by a deterministic transition matrix $\mathbf{F}_{i,t}$ and a random innovation $\boldsymbol{q}_{i,t}$. The structure of the transition matrix $\mathbf{F}_{i,t}$ and innovation strength $\boldsymbol{q}_{i,t}$ determines which kinds of time series patterns the latent state $\boldsymbol{h}_{i,t}$ encodes (cf. Appendix A.2 for the concrete choice of ISSM).

The third local model, DF-GP, is defined as the Gaussian Process,

$$r_{i,t} \sim \text{GP}(0, \mathcal{K}_i(\cdot, \cdot)), \tag{7}$$

where $\mathcal{K}_i(\cdot, \cdot)$ denotes the kernel function. In this model, with each time series has its own set of GP hyperparameters to be learned.

3.2. Inference and Learning

Given a set of N time series, our goal is to jointly estimate $\boldsymbol{\Theta}$, the parameters in the global RNNs, the embeddings and the hyper-parameters in the local models. We use maximum likelihood estimation, where $\boldsymbol{\Theta} = \arg\max \sum_i \log p(\boldsymbol{z}_i|\boldsymbol{\Theta})$. Computing the marginal likelihood may require doing inference over the latent variables. The general learning algorithm is summarized in Algorithm 1.

Algorithm 1 Training Procedure for Deep Factor Models with Random Effects.

1: **for** each time series $\{(\boldsymbol{x}_i, \boldsymbol{z}_i)\}$ **do**
2:     Sample the estimated latent representation from the variational encoder $\widetilde{\boldsymbol{u}}_i \sim q_\phi(\cdot|\boldsymbol{z}_i)$ for non-Gaussian likelihood, otherwise $\widetilde{\boldsymbol{u}}_i := \boldsymbol{z}_i$.
3:     With the current estimate of the model parameters $\boldsymbol{\Theta}$, compute the fixed effect $f_{i,t} = \sum_{k=1}^K w_{i,k} \cdot g_k(\boldsymbol{x}_{i,t})$, and corresponding ISSM parameters for DF-LDS or the kernel matrix $\mathbf{K}_i$ for DF-GP.
4:     Calculate the marginal likelihood $p(\boldsymbol{z}_i)$ as in Table 1 or its variational lower bound as in Eqn. (10).
5: **end for**
6: Accumulate the loss in the current mini-batch, and perform stochastic gradient descent.

3.2.1. GAUSSIAN LIKELIHOOD

When the observation model $p(\cdot|u_{i,t})$ is Gaussian, $z_{i,t} \sim \mathcal{N}(u_{i,t}, \sigma_{i,t}^2)$, the marginal likelihood can be easily computed for all three models. Evaluating the marginal likelihood for DF-RNN is straightforward (see Table 1).

For DF-LDS and DF-GP, the Gaussian noise can be absorbed into the local model, yielding $z_{i,t} = u_{i,t} = f_{i,t} + r_{i,t}$, where $r_{i,t}$, instead of coming from the noiseless LDS and GP, is generated by the noisy versions. More precisely, for DF-LDS, $r_{i,t} = \boldsymbol{a}_{i,t}^\top \boldsymbol{h}_{i,t} + \nu_{i,t}$ and $\nu_{i,t} \sim \mathcal{N}(0, \sigma_i^2)$, and the marginal likelihood is obtained with a Kalman filter. In DF-GP, it amounts to adding $\sigma_{i,t}^2 \cdot \delta(\cdot, \cdot)$ to Eqn (7), where $\delta(\cdot, \cdot)$ is the Dirac delta function. The marginal likelihood becomes the standard GP marginal likelihood, which is the multivariate normal with mean $\boldsymbol{f}_i := [f_i(\boldsymbol{x}_{i,t})]_t$ and covariance matrix $\mathbf{K}_i + \boldsymbol{\sigma}_i^2 \mathbf{I}$, where $\mathbf{K}_i := [\mathcal{K}(\boldsymbol{x}_{i,t}, \boldsymbol{x}_{i,t})]_t$ and $\mathbf{I}$ is the identity matrix of suitable size.

3.2.2. NON-GAUSSIAN LIKELIHOOD

When the likelihood is not Gaussian, the exact marginal likelihood is intractable. We use variational inference, and optimize a variational lower bound of the marginal likelihood $\log p(\boldsymbol{z})$:

$$\log \int p(\boldsymbol{z}, \boldsymbol{u}, \boldsymbol{h}) \geqslant \mathbb{E}_{q_\phi(\boldsymbol{u}, \boldsymbol{h})} \log \left[\frac{p(\boldsymbol{z}, \boldsymbol{u}, \boldsymbol{h})}{q_\phi(\boldsymbol{u}, \boldsymbol{h})} \right], \tag{8}$$

where $\boldsymbol{u}$ is the latent function values, and $\boldsymbol{h}$ is the latent states in the local probabilistic models [2]. Optimizing this stochastic variational lower bound for the joint model over all time series is computationally expensive.

We propose a new method that leverages the structure of the local probabilistic model to enable fast inference at the per-time series level. This enables parallelism and efficient inference that scales up to a large collection (millions) of time series. Motivated by (Fraccaro et al., 2017), we choose the following structural approximation $q_\phi(\boldsymbol{h}, \boldsymbol{u}|\boldsymbol{z}) := q_\phi(\boldsymbol{u}|\boldsymbol{z})p(\boldsymbol{h}|\boldsymbol{u})$, where the second term matches the exact conditional posterior with the random effect probabilistic model $\mathcal{R}$. With this form, given $\boldsymbol{u}$, the inference becomes the canonical inference problem with $\mathcal{R}$, from Section 3.2.1. The first term $q_\phi(\boldsymbol{u}|\boldsymbol{z})$ is given by another neural network parameterized by ϕ, called a recognition network in the variational Auto-Encoder (VAE) framework (Kingma & Welling, 2014; Rezende et al., 2014). After massaging the equations, the stochastic variational lower bound in Eqn. (8) becomes

$$\mathbb{E}_{q_\phi(\boldsymbol{u})}\left[\log\left[\frac{p(\boldsymbol{z}|\boldsymbol{u})}{q_\phi(\boldsymbol{u}|\boldsymbol{z})}\right] + \log p(\boldsymbol{u})\right] \qquad (9)$$

$$\approx \frac{1}{L}\left(\log p(\boldsymbol{z}|\widetilde{\boldsymbol{u}}_j) + \log p(\widetilde{\boldsymbol{u}}_j) - \log q_\phi(\widetilde{\boldsymbol{u}}_j|\boldsymbol{z})\right), \quad (10)$$

with $\widetilde{\boldsymbol{u}}_j \sim q_\phi(\boldsymbol{u})$ for $j = 1, \cdots, L$ sampled from the recognition network. The first and third terms in Eqn. (10) are straightforward to compute. For the second term, we drop the sample index j to obtain the marginal likelihood $\log p(\widetilde{\boldsymbol{u}})$ under the normally distributed random effect models. This term is computed in the same manner as in Section 3.2.1, with $\boldsymbol{z}_i$ substituted by $\widetilde{\boldsymbol{u}}$. When the likelihood is Gaussian, the latent function values $\boldsymbol{u}$ are equal to $\boldsymbol{z}$, and we arrive at $\log p(\boldsymbol{z})$ from Eqn. (9).

4. Related Work and Discussions

Effectively combining probabilistic graphical models and deep learning approaches has been an active research area. Several approaches have been proposed for marrying RNN with SSMs through either one or both of the following: (i) extending the Gaussian emission to complex likelihood models; (ii) making the transition equation non-linear via a multi-layer perceptron (MLP) or interlacing SSM with transition matrices temporally specified by RNNs. Deep Markov Models (DMMs), proposed by (Krishnan et al., 2017; 2015), keep the Gaussian transition dynamics with mean and co-variance matrix parameterized by MLPs. Stochastic RNNs (SRNNs) (Fraccaro et al., 2016) explicitly incorporate the deterministic dynamics from RNNs that do not depend on latent variables by interlacing them with a SSM. Chung et al. (2015) first proposed Variational RNNs (VRNNs), which is

another way to make the transition equation non-linear, by cutting ties between the latent states and associating them with deterministic states. This makes the state transition non-linearly determined by the RNN. VRNNs are also used in Latent LSTM Allocation (LLA) (Zaheer et al., 2017) and State-Space LSTM (SSL) (Zheng et al., 2017). These models require expensive inference at the global level through a recognition network, with is in stark contrast with Deep Factors, where the structural assumption of the variational approximation decomposes the inference problem to local probabilistic inference that is easily parallelizable and global standard RNN learning (cf. Section 3.2).

In Fraccaro et al. (2017) and the recent Deep State Models (Rangapuram et al., 2018), the linear Gaussian transition structure is kept intact, so that the highly efficient Kalman filter/smoother is readily applicable. Our model differs from the former in that we avoid sampling the latent states in the ELBO, and eliminate the variance coming from the Monte-Carlo estimate of the second integral in Eqn. (10). Deep State is designed for a Gaussian likelihood with time varying SSM parameters per time series. In contrast, with time invariant local SSMs and flexible global effects, our model DF-LDS offers a parsimonious representation that can handle non-Gaussian observations.

Deep Gaussian Processes (Damianou & N.D., 2013) have attracted growing interests in recent years. Inspired by GP-LVM structure (Lawrence, 2004), Deep GPs stacks GPs on top of latent variables, resulting in more expressive mappings. Our framework provides an alternative approach to utilize GPs more efficiently.

Due to its flexibility of interpolating between purely local and purely global models, there are a variety of common methods that can be subsumed in our proposed model framework. Deep Factor with one factor and no random effects, accompanied with autoregressive inputs, reduces to DeepAR (Flunkert et al., 2017). One difference in our formulation is that the scale of each time series is automatically estimated rather than pre-specified as it is in DeepAR. Changing the emission probability to Gaussian Mixtures results in AR-MDN (Mukherjee et al., 2018). Sequence-to-Sequence models for forecasting (Wen et al., 2017) are another family of models that are a part of our framework. These methods make predictions discriminatively rather than generatively. By dropping the random effects, using GPs as the prior and removing the restriction of RNNs on the global factors, we recover the semi-parametric latent factor model (Teh et al., 2005). By dropping the global effects, we arrive at the standard SSM or GP. Our newly developed general variational inference algorithm applies to both of these methods and other normally distributed local stochastic models (cf. subsubsection 3.2.2). While we have built upon existing works, to the best of our knowledge, Deep

---

[2]For cleaner notation, we omit the time series index i

Factors provide the first model framework that incorporate SSMs and GPs with DNNs in a systematic manner.

5. Experiments

We conduct experiments with synthetic and real-world data to provide evidence for the practical effectiveness of our approach. We use a p3.8xlarge SageMaker instance in all our experiments. Our algorithms are implemented in MXNet Gluon (Chen et al., 2015), and make extensive use of its Linear Algebra library (Seeger et al., 2017; Zhenwen et al., 2018). Further experiments with GPs as the local model are detailed in (Maddix et al., 2018).

5.1. Model Understanding and Exploration

The first set of experiments compares our proposed structure in Eqn. (3) with no probabilistic component to the standard RNN structure on the electricity dataset. For each time series i, we have its embedding $w_i \in \mathbb{R}^{10}$, and two time features $x_t \in \mathbb{R}^2$, day of the week and hour of the day. Given an RNN Cell (LSTM, GRU, etc.), the RNN Forecaster predicts the time series values $\hat{z}_{i,t} = \text{RNN}_t(x_{i,t})$, where $x_{i,t} = [w_i; x_t]$. Deep Factor generates the point forecast as $\hat{z}_{i,t} = w_i^\top \text{RNN}_t(x_t)$. The RNN cell for RNN Forecaster has an output dimension of 1 while that of DeepFactor is of dimension 10. The resulting number of parameters of both structures are roughly the same.

Figure 2 demonstrates that the proposed structure significantly improves the data efficiency while having less variance. The runtime of Deep Factor scales better than that of the standard RNN structure. By using the concatenation of $[w_i; x_t]$, the standard RNN structure operates on the outer product space of x_t and w_i (of dimension $12 \times T$), while Deep Factor computes the intrinsic structure (of dimension 12 and T separately).

Next, we focus on the DF-LDS model, and investigate (i) the effect of the recognition network for non-Gaussian observations with the purely local part ($f_{i,t} = 0$) (variational LDS cf. Appendix A.2), and (ii) the recovery of the global factors in the presence of Gaussian and non-Gaussian noise. We generate data according to the following model, which is adapted from Example 24.3 in (Barber, 2012). The two-dimensional latent vector h_t is rotated at each iteration, and then projected to produce a scalar observation,

$$h_{t+1} = \mathbf{A}h_t + \epsilon_h, \quad \mathbf{A} = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix},$$

where $\epsilon_h \sim \mathcal{N}(0, \alpha^2\mathbf{I}_2)$, $u_{t+1} = e_1^T h_{t+1} + \epsilon_v$, $\epsilon_v \sim \mathcal{N}(0, \sigma^2)$, $\mathbf{I}_2$ is the 2×2 identity matrix, and $e_1 \in \mathbb{R}^2$ is the standard basis vector. The true observations are generated by a Poisson distribution, $z_t = \text{Poisson}[\lambda(u_t)]$, where $\lambda(u_t) = \log[1 + \exp(u_t)]$. This could be used to model seasonal products, where most of the sales happen when an item is in season, e.g. snowboards normally sell shortly before or during winters, and less so afterwards. Figure 3 shows the reconstructed intensity function $\lambda(u_t)$, as well as corresponding forecasts for each choice of recognition network. Visual inspections reveal that RNNs are superior over MLPs as recognition networks. This is expected because the time series are sequential. We also test the ability of our algorithm to recover the underlying global factors. Our experiments show that even with the Poisson noise model, we are able to identify the true latent factors in the sense of distances of the subspaces spanned by the global factors (cf. Appendix B.1).

5.2. Empirical Studies

In this subsection, we test how our model performs on several real-world and publicly available datasets: electricity (**E**) and traffic (**T**) from the UCI data set (Dheeru & Karra Taniskidou, 2017; Yu et al., 2016), nyc taxi (**N**) (Taxi & Commission, 2015) and uber (**U**) (Flowers, 2015) (cf. Appendix B.2).

In the experiments, we choose the DF-RNN (**DF**) model with a Gaussian likelihood. To assess the performance of our algorithm, we compare with DeepAR (**DA**), a state-of-art RNN-based probabilistic forecasting algorithm on the publicly available AWS SageMaker (Januschowski et al., 2018), MQ-RNN (**MR**), a sequence model that generates conditional predictive quantiles (Wen et al., 2017), and Prophet (**P**), a Bayesian structural time series model (Taylor & Letham, 2017). The Deep Factor model has 10 global factors with a LSTM cell of 1-layer and 50 hidden units. The noise LSTM has 1-layer and 5 hidden units. For a fair comparison with DeepAR, we use a comparable number of model parameters, that is, an embedding size of 10 with 1-layer and 50 hidden LSTM units. The student-t likelihood in DeepAR is chosen for its robust performance. The same model structure is chosen for MQ-RNN, and the decoder MLP has a single hidden layer of 20 units. We use the adam optimization method with the default parameters in Gluon to train the DF-RNN and MQ-RNN. We use the default training parameters for DeepAR.

We use the quantile loss to evaluate the probabilistic forecasts. For a given quantile $\rho \in (0, 1)$, a target value z_t and ρ-quantile prediction $\hat{z}_t(\rho)$, the ρ-quantile loss is defined as

$$\text{QL}_\rho[z_t, \hat{z}_t(\rho)] = 2\big[\rho(z_t - \hat{z}_t(\rho))\mathbb{I}_{z_t - \hat{z}_t(\rho) > 0} + (1 - \rho)(\hat{z}_t(\rho) - z_t)\mathbb{I}_{z_t - \hat{z}_t(\rho) \leqslant 0}\big].$$

We use a normalized sum of quantile losses, $\sum_{i,t} \text{QL}_\rho[z_{i,t}, \hat{z}_{i,t}(\rho)] / \sum_{i,t} |z_{i,t}|$, to compute the quantile losses for a given span across all time series. We include results for $\rho = 0.5, 0.9$, which we abbreviate as the P50QL (mean absolute percentage error (MAPE)) and P90QL, respectively.
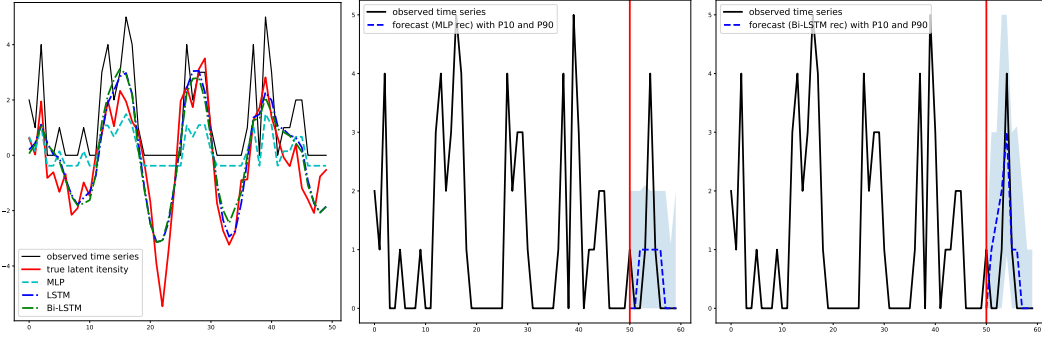
Figure 3. DeepFactor (DF-LDS) with no global effects (Variational LDS). Left: reconstructed Intensity (u_t) with different recognition networks. Center and Right: predictive distributions with MLP (center) and Bidirectional LSTM (right).

DS	H	P50QL				P90QL			
		DA	P	MR	DF	DA	P	MR	DF
E	72	0.216±0.054	0.149	0.204±0.042	**0.101± 0.006**	0.182 ±0.077	0.103	0.088±0.008	**0.049±0.004**
	24	0.272±0.078	0.124	0.185±0.042	**0.112±0.012**	0.100±0.013	0.091	0.083±0.008	**0.059 ±0.013**
N	72	0.468±0.043	0.397	0.418±0.031	**0.212±0.044**	0.175±0.005	0.178	0.267±0.038	**0.104±0.028**
	24	0.390±0.042	0.328	0.358±0.029	**0.239±0.037**	0.167± 0.005	0.168	0.248±0.088	**0.139±0.035**
T	72	0.337±0.026	0.457	0.383±0.040	**0.190±0.048**	0.184±0.051	0.207	0.226±0.017	**0.127±0.026**
	24	0.296 ±0.021	0.380	0.331±0.011	**0.225±0.050**	**0.149±0.011**	0.191	0.154±0.020	0.159±0.059
U	72	0.417±0.011	0.441	0.730±0.031	**0.344±0.033**	0.296±0.017	0.239	0.577±0.059	**0.190±0.013**
	24	**0.353±0.009**	0.468	0.879±0.156	0.425±0.063	**0.238±0.009**	0.239	0.489±0.069	0.238±0.026

Table 2. Results for the short-term (72-hour) and near-term (24-hour) forecast scenarios with one week of training data.
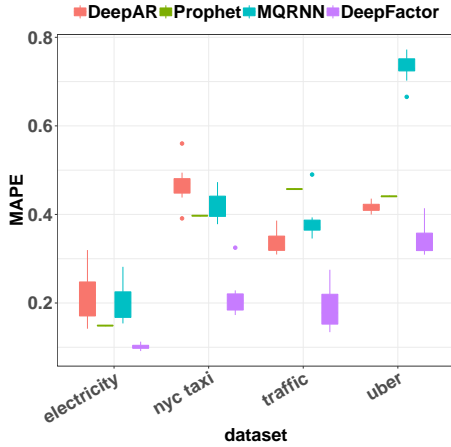

Figure 4. P50QL (MAPE) results for the short-term (72-hour) forecast in Table 2. Purple denotes the proposed method.

For all the datasets, we limit the training length to only one week of time series (168 observations per time series). This represents a relevant scenario that occurs frequently in demand forecasting, where products often have only limited historical sales data. We average the Deep Factor, MQ-RNN and DeepAR results over ten trials. We use one trial for Prophet, since classical methods are typically less variable than neural-network based models. Figure 4 illustrates the performances of the different algorithms in terms of the MAPE (P50 quantile loss) for the 72 hour forecast horizon.

Table 2 shows the full results, and that our model outperforms the others in terms of accuracy and variability in most of the cases. For DeepAR, using SageMaker's HPO, our preliminary results (cf. Appendix B.2) show that with a larger model, it achieves a performance that is on-par with our method, which has much less parameters. In addition, the sequence-to-sequence structure of DeepAR and MQ-RNN limits their ability to react flexibly to changing forecasting scenarios, e.g. during on-demand forecasting, or interactive scenarios. For a forecasting scenario with a longer prediction horizon than during training horizon, DeepAR needs to be retrained to reflect the changes in the decoder length. Similarly, they cannot generate forecasts that are longer than the length of the training time series, for example, the case in Figure 2. Our method has no difficulty performing this task and has greater data efficiency.

6. Conclusion

We propose a novel global-local framework for forecasting a collection of related time series, accompanied with a result that uniquely characterizes exchangeable time series. Our main contribution is a general, powerful and practically relevant modeling framework that scales, and obtains state-of-the-art performance. Future work includes comparing variational dropout (Gal & Ghahramani, 2016) or Deep Ensemble (Lakshminarayanan et al., 2017) of non-probabilistic DNN models (e.g., RNNForecaster (cf. 5.1)) for uncertainty.

References

Ahmadi, B., Kersting, K., and Sanner, S. Multi-evidence lifted message passing, with application to pagerank and the kalman filter. In *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

Ahmed, A., Aly, M., Gonzalez, J., Narayanamurthy, S., and Smola, A. J. Scalable inference in latent variable models. In *Proceedings of the fifth ACM international conference on Web search and data mining*, pp. 123–132. ACM, 2012.

Araujo, M., Ribeiro, P., Song, H. A., and Faloutsos, C. Tensorcast: forecasting and mining with coupled tensors. *Knowledge and Information Systems*, 59(3):497–522, 2019.

Austin, T. and Panchenko, D. A hierarchical version of the de finetti and aldous-hoover representations. *Probability Theory and Related Fields*, 159(3-4):809–823, 2014.

Barber, D. *Bayesian reasoning and machine learning*. Cambridge University Press, 2012.

Böse, J.-H., Flunkert, V., Gasthaus, J., Januschowski, T., Lange, D., Salinas, D., Schelter, S., Seeger, M., and Wang, Y. Probabilistic demand forecasting at scale. *Proceedings of the VLDB Endowment*, 10(12):1694–1705, 2017.

Brahim-Belhouari, S. and Bermak, A. Gaussian process for nonstationary time series prediction. *Computational Statistics & Data Analysis*, 47(4):705–712, 2004.

Chen, T., Li, M., Li, Y., Lin, M., Wang, N., Wang, M., Xiao, T., Xu, B., Zhang, C., and Zhang, Z. Mxnet: A flexible and efficient machine learning library for heterogeneous distributed systems. *arXiv preprint arXiv:1512.01274*, 2015.

Choi, J., Guzman-Rivera, A., and Amir, E. Lifted relational kalman filtering. In *Twenty-Second International Joint Conference on Artificial Intelligence*, 2011.

Chung, J., Kastner, K., Dinh, L., Goel, K., Courville, A. C., and Bengio, Y. A recurrent latent variable model for sequential data. In *Advances in neural information processing systems*, pp. 2980–2988, 2015.

Crawley, M. J. Mixed-effects models. *The R Book, Second Edition*, pp. 681–714, 2012.

Damianou, A. and N.D., L. Deep gaussian processes. *In Proceedings of the International Conference on Artificial Intelligence and Statistics (AISTATS)*, pp. 207–215, 2013.

Dheeru, D. and Karra Taniskidou, E. UCI machine learning repository. http://archive.ics.uci.edu/ml, 2017. University of California, Irvine, School of Information and Computer Sciences.

Diaconis, P. Finite forms of de finetti's theorem on exchangeability. *Synthese*, 36:271–281, 1977.

Diaconis, P. and Freedman, D. Finite exchangeable sequences. *The Annals of Probability*, 8:745–764, 1980.

Faloutsos, C., Gasthaus, J., Januschowski, T., and Wang, Y. Forecasting big time series: old and new. *Proceedings of the VLDB Endowment*, 11(12):2102–2105, 2018.

Flowers, A. Uber TLC foil response. https://github.com/fivethirtyeight/uber-tlc-foil-response/blob/master/uber-trip-data/uber-raw-data-apr14.csv, 2015.

Flunkert, V., Salinas, D., and Gasthaus, J. Deepar: Probabilistic forecasting with autoregressive recurrent networks. *arXiv preprint arXiv:1704.04110*, 2017.

Forni, M., Hallin, M., Lippi, M., and Reichlin, L. The generalized dynamic-factor model: Identification and estimation. *Review of Economics and statistics*, 82(4):540–554, 2000.

Fraccaro, M., Sønderby, S. K., Paquet, U., and Winther, O. Sequential neural models with stochastic layers. In *Advances in neural information processing systems*, pp. 2199–2207, 2016.

Fraccaro, M., Kamronn, S., Paquet, U., and Winther, O. A disentangled recognition and nonlinear dynamics model for unsupervised learning. In *Advances in Neural Information Processing Systems*, pp. 3604–3613, 2017.

Gal, Y. and Ghahramani, Z. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pp. 1050–1059, 2016.

Gasthaus, J., Benidis, K., Wang, Y., Rangapuram, S. S., Salinas, D., Flunkert, V., and Januschowski, T. Probabilistic forecasting with spline quantile function rnns. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 1901–1910, 2019.

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., and Rubin, D. B. *Bayesian data analysis*. CRC press, 2013.

Geweke, J. The dynamic factor analysis of economic time series. *Latent variables in socio-economic models*, 1977.

Girard, A., Rasmussen, C. E., Candela, J. Q., and Murray-Smith, R. Gaussian process priors with uncertain inputs application to multiple-step ahead time series forecasting. In *Advances in neural information processing systems*, pp. 545–552, 2003.

Harvey, A. C. *Forecasting, structural time series models and the Kalman filter*. Cambridge university press, 1990.

Hooi, B., Shin, K., Liu, S., and Faloutsos, C. Smf: Drift-aware matrix factorization with seasonal patterns. In *Proceedings of the 2019 SIAM International Conference on Data Mining*, pp. 621–629. SIAM, 2019.

Hwang, Y., Tong, A., and Choi, J. Automatic construction of nonparametric relational regression models for multiple time series. In *International Conference on Machine Learning*, pp. 3030–3039, 2016.

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

Januschowski, T., Arpin, D., Salinas, D., Flunkert, V., Gasthaus, J., Stella, L., and Vazquez, P. Now available in amazon sagemaker: Deepar algorithm for more accurate time series forecasting. *https://aws.amazon.com/blogs/machine-learning/now-available-in-amazon-sagemaker-deepar-algorithm-for-more-accurate-time-series-forecasting/*, 2018.

Jing, H. and Smola, A. J. Neural survival recommender. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*, pp. 515–524. ACM, 2017.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *ICLR*, 2014.

Krishnan, R. G., Shalit, U., and Sontag, D. Deep kalman filters. *arXiv preprint arXiv:1511.05121*, 2015.

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

Lakshminarayanan, B., Pritzel, A., and Blundell, C. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems*, pp. 6402–6413, 2017.

Lawrence, N. D. Gaussian process latent variable models for visualisation of high dimensional data. In *Advances in neural information processing systems*, pp. 329–336, 2004.

Low, Y., Agarwal, D., and Smola, A. J. Multiple domain user personalization. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 123–131. ACM, 2011.

Maddix, D. C., Wang, Y., and Smola, A. Deep factors with gaussian processes for forecasting. *NeurIPS workshop on Bayesian Deep Learning*, 2018.

Mukherjee, S., Shankar, D., Ghosh, A., Tathawadekar, N., Kompalli, P., Sarawagi, S., and Chaudhury, K. Armdn: Associative and recurrent mixture density networks for eretail demand forecasting. *arXiv preprint arXiv:1803.03800*, 2018.

Pan, J. and Yao, Q. Modelling multiple time series via common factors. *Biometrika*, 95(2):365–379, 2008.

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

Rasmussen, C. E. and Williams, C. K. *Gaussian process for machine learning*. MIT press, 2006.

Rezende, D. J., Mohamed, S., and Wierstra, D. Stochastic backpropagation and approximate inference in deep generative models. In *International Conference on Machine Learning*, pp. 1278–1286, 2014.

Seeger, M. Gaussian processes for machine learning. *International journal of neural systems*, 14(02):69–106, 2004.

Seeger, M., Hetzel, A., Dai, Z., Meissner, E., and Lawrence, N. D. Auto-differentiating linear algebra. *arXiv preprint arXiv:1710.08717*, 2017.

Seeger, M. W., Salinas, D., and Flunkert, V. Bayesian intermittent demand forecasting for large inventories. In *Advances in Neural Information Processing Systems*, pp. 4646–4654, 2016.

Stock, J. H. and Watson, M. Dynamic factor models. *Oxford handbook on economic forecasting*, 2011.

Taxi, N. and Commission, L. TLC trip record data. http://www.nyc.gov/html/tlc/html/about/trip_record_data.shtml, 2015.

Taylor, S. J. and Letham, B. Forecasting at scale. *The American Statistician*, 2017.

Teh, Y., Seeger, M., and Jordan, M. Semiparametric latent factor models. In *AISTATS 2005-Proceedings of the 10th International Workshop on Artificial Intelligence and Statistics*, pp. 333–340, 2005.

Wen, R., Torkkola, K., and Narayanaswamy, B. A multi-horizon quantile recurrent forecaster. *NIPS Workshop on Time Series*, 2017.

Xie, C., Tank, A., Greaves-Tunnell, A., and Fox, E. A unified framework for long range and cold start forecasting of seasonal profiles in time series. *arXiv preprint arXiv:1710.08473*, 2017.

Xu, Z., Kersting, K., and Tresp, V. Multi-relational learning with gaussian processes. In *Twenty-First International Joint Conference on Artificial Intelligence*, 2009.

Yu, H.-F., Rao, N., and Dhillon, I. S. Temporal regularized matrix factorization for high-dimensional time series prediction. In *Advances in neural information processing systems*, pp. 847–855, 2016.

Zaheer, M., Ahmed, A., and Smola, A. J. Latent lstm allocation: Joint clustering and non-linear dynamic modeling of sequence data. In *International Conference on Machine Learning*, pp. 3967–3976, 2017.

Zheng, X., Zaheer, M., Ahmed, A., Wang, Y., Xing, E. P., and Smola, A. J. State space lstm models with particle mcmc inference. *arXiv preprint arXiv:1711.11179*, 2017.

Zhenwen, D., Meissner, E., and Lawrence, N. D. Mxfusion: A modular deep probabilistic programming library. *NIPS 2018 Workshop MLOSS*, 2018.

Acknowledgments

A. Deep Factor Models with Random Effects

In this section, we provide additional details on the models discussed in Section 3.1. In what follows, we overload the notation and use $\boldsymbol{g}_t(\cdot) : \mathbb{R}^d \to \mathbb{R}^K$ to denote the vector function consisting of the global factors.

NAME	DESCRIPTION	LOCAL	LIKELIHOOD (GAUSSIAN CASE)
DF-RNN	Zero-mean Gaussian noise process given by RNN	$r_{i,t} \sim \mathcal{N}(0, \sigma_{i,t}^2)$	$p(\boldsymbol{z}_i) = \prod_t \mathcal{N}(z_{i,t} - f_{i,t}\|0, \sigma_{i,t}^2)$
DF-LDS	State-space models	$r_{i,t} \sim \text{LDS}_{i,t}$ (cf. Eqn. (6))	$p(\boldsymbol{z}_i)$ given by Kalman Filter
DF-GP	Zero-mean Gaussian Process	$r_{i,t} \sim \text{GP}_{i,t}$ (cf. Eqn. (7))	$p(\boldsymbol{z}_i) = \mathcal{N}(\boldsymbol{z}_i - \boldsymbol{f}_i\|\boldsymbol{0}, \mathbf{K}_i + \boldsymbol{\sigma}_i^2\mathbf{I})$

Table A.1. Summary table of Deep Factor Models with Random Effects. The likelihood column is under the assumption of Gaussian noise.

A.1. DF-RNN: Gaussian noise process as the local model

The local model in DF-RNN is zero-mean i.i.d. Gaussian, where the variance $\sigma_{i,t}^2$ is generated by a noise RNN process. The generative model is given as:

$$\begin{aligned}
\text{random effect}: \quad & \varepsilon_{i,t} \sim \mathcal{N}(0, \sigma_{i,t}^2), \quad \sigma_{i,t} := \text{RNN}(x_{i,t}), \\
\text{fixed effect}: \quad & f_{i,t} = \boldsymbol{w}_i^\top \boldsymbol{g}_t(x_{i,t}), \\
\text{emission}: \quad & z_{i,t} \sim p(\cdot|u_{i,t}), \; u_{i,t} = f_{i,t} + r_{i,t}.
\end{aligned}$$

Although the noise is i.i.d., there is correlated noise from the latent RNN process. For the non-Gaussian likelihood, the latent innovation terms $\boldsymbol{\varepsilon}$ in the latent function $\boldsymbol{u}$ requires inference. To do so, we use Algorithm 1. The variational lower bound in Eqn. (10) is straightforward to calculate, since the marginal likelihood $\log p(\widetilde{\boldsymbol{u}})$ is the log-likelihood of a multivariate Gaussian with mean $w^\top g_t$ and a covariance matrix with ε_t^2 on the diagonal entries. DF-RNN can be seen as a type of deep Gaussian latent variable model.

A.2. DF-LDS: LDS as the local model

The generative model for DF-LDS is given as:

$$\begin{aligned}
\text{random effect}: \quad & \boldsymbol{h}_{i,t} = \mathbf{F}_{i,t}\boldsymbol{h}_{i,t-1} + \boldsymbol{q}_{i,t}\epsilon_{i,t}, \quad \epsilon_{i,t} \sim \mathcal{N}(0, 1), \\
& r_{i,t} = \boldsymbol{a}_{i,t}^\top \boldsymbol{h}_{i,t} \\
\text{fixed effect}: \quad & f_{i,t} = \boldsymbol{w}_i^\top \boldsymbol{g}_t(x_{i,t}), \\
\text{emission}: \quad & z_{i,t} \sim p(\cdot|u_{i,t}), \; u_{i,t} = f_{i,t} + r_{i,t}.
\end{aligned}$$

We consider the level-trend ISSM model with damping,

$$\boldsymbol{a}_{i,t} = \begin{bmatrix} \delta_i \\ \gamma_i \end{bmatrix}, \quad \mathbf{F}_{i,t} = \begin{bmatrix} \delta_i & \gamma_i \\ 0 & \gamma_i \end{bmatrix}, \quad \text{and} \quad \boldsymbol{q}_{i,t} = \begin{bmatrix} \alpha_i \\ \beta_i \end{bmatrix},$$

where δ_i, γ_i control the damping effect, and the parameters $\alpha_i > 0$ and $\beta_i > 0$ contain the innovation strength for the level and trend (slope), respectively. The initial latent state $h_{i,0}$ is assumed to follow an isotropic Gaussian distribution. We choose a simple level-trend ISSM because we expect the global factors with attention to explain the majority of the time series' dynamics, such as trend and seasonality. This avoids having to hand design the latent states of the ISSM.

As a special case, we recover SSM with non-Gaussian likelihood, and our inference algorithm gives a new approach to perform inference on such models, which we call *Variational LDS*.

Variational LDS: DF-LDS with no global factors Variational LDS is a state-space model (SSM) with an arbitrary likelihood as its emission function. In this subsection, we supply additional details of the inference of the variational LDS

using the synthetic example in Section 5.1. The data is generated as,

$$h_{t+1} = Ah_t + \epsilon_h, \quad A = \begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix},$$

$$\epsilon_h \sim \mathcal{N}(0, \alpha^2 \mathbf{I}_2), \quad v_{t+1} = e_1^T h_{t+1} + \epsilon_v, \quad \epsilon_v \sim \mathcal{N}(0, \sigma^2).$$

Our observations are generated as:

$$z_t = \text{Poisson}[\lambda(v_t)], \qquad \lambda(v_t) = \log[1 + \exp(v_t)],$$

resulting in the popular Poisson LDS.

Our goal is perform inference over the latent variables, $\{h, \boldsymbol{u}\}$, where $h = h_{1:T}$ and $\boldsymbol{u} = v_{1:T}$, given the observations $z := z_{1:T}$. As in Section 3.2.2, we use a variational auto-encoder (VAE) with structural approximation, and a variational posterior to approximate the true posterior $p(h, \boldsymbol{u}|z)$ as $q_\phi(h, \boldsymbol{u}|z) = q_\phi(\boldsymbol{u}|z)p(h|\boldsymbol{u})$. As in DF-LDS, the first term is generated by the recognition network, and the second term chosen to match the conditional posterior. For the training procedure, we resort to Algorithm 1 with no global factors, i.e., $\widetilde{r}$ is the sample generated by the recognition network.

There are different neural networks structures to choose from for the variational encoder $q_\phi(h|\boldsymbol{z})$. We choose the bi-directional LSTM since it considers information from both the past and the future. This is similar to the backwards message in the Kalman smoother. Similar preference can also be found in (Krishnan et al., 2015), although their model structure differs. The recognition network is only used in the training phase to better approximate the posterior distribution of the latent function values, and so there is no information leak for the desired forecasting case.

A.3. DF-GP: Gaussian Process as the local model

With DF-GP, the random effects is assumed to be drawn from a zero-mean Gaussian process.

$$\begin{aligned} \text{random effect}: \quad & r_i(\cdot) \sim \text{GP}(0, \mathcal{K}_i(\cdot, \cdot)), \\ \text{fixed effect}: \quad & f_{i,t} = \boldsymbol{w}_i^\top \boldsymbol{g}_t(x_{i,t}), \\ \text{latent function values}: \quad & u_{i,t} = f_{i,t} + r_{i,t}, \quad r_{i,t} = r_i(\boldsymbol{x}_{i,t}), \\ \text{emission}: \quad & z_{i,t} \sim p(\cdot|u_{i,t}). \end{aligned}$$

Simiarily with Variational LDS, with the proposed inference algorithm, we get a new approach for doing approximate inference for GP with non-Gaussian likelihood. We leave the comparison with classical approaches (Rasmussen & Williams, 2006) such as Laplace approximation, Expectation Propagation, Variational Inference to future work.

B. Detailed Experimental Analysis

In this appendix, we provide further results in both the synthetic and empirical cases.

B.1. Synthetic Experiments

The local model and observations are generated using the variational-LDS example in Appendix A.2. We want to test the ability of the algorithm to recover the underlying latent factors. To do so, we add global factors that are generated from Fourier series of different orders, to the synthetic dataset described in Section 5.1. For each time series, the coefficients of the linear combination are sampled from a uniform distribution.

In the presence of Gaussian LDS noise ($\boldsymbol{u}_t$ observed), Figure B.1 shows that the algorithm does not precisely recover the global factors. The distance between the subspaces spanned by the true and estimated factors reveals that they are reasonably close. Even with Poisson observations in Figure B.2, the method is still able to roughly recover the factors.
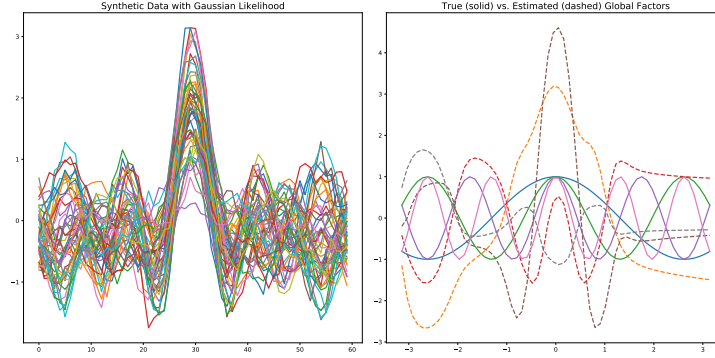
Figure B.1. Synthetic data with Gaussian noise and true (solid) vs. estimated (dashed) global factors.
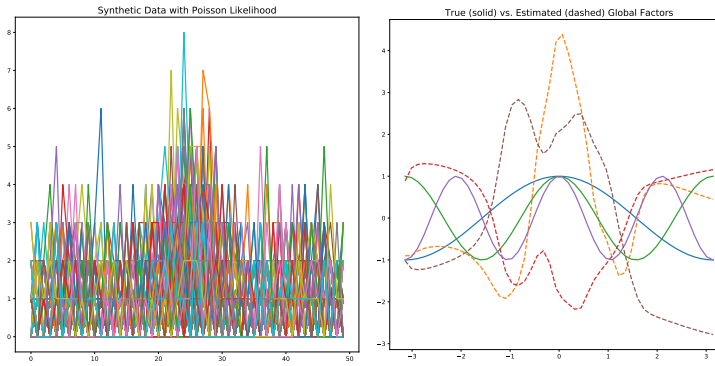

Figure B.2. Synthetic data with Poisson noise and true (solid) vs. estimated (dashed) global factors.

B.2. Empirical Studies

Table B.1 summarizes the datasets that are used to test our Deep Factor models.

NAME	SUPPORT	GRANULARITY	NO. TS	COMMENT
electricity	$\mathbb{R}^+$	hourly	370	electricity consumption of different customers
traffic	$[0, 1]$	hourly	963	occupancy rate of SF bay area freeways
taxi	$\mathbb{N}^+$	hourly	140	No. of taxi pick ups in different blocks of NYC
uber	$\mathbb{N}^+$	hourly	114	No. of Uber pick ups in different blocks of NYC

Table B.1. Summary of the datasets used to test the models.

We first compare our two methods, DF-RNN and DF-LDS with level-trend ISSM, on the electricity dataset with forecast horizon 24. Table B.2 shows the averaged results over 50 trials. We see that both methods have similar accuracy with DF-RNN being slightly preferrable. Due to its simplicity and computational advantages, we choose DF-RNN as the Deep Factor model to compare against the other baseline methods. Regarding DF-GP, although it shows a good performance in our perliminary results, still extra exploration is needed for which we left to future work.

METHOD	P50QL	P90QL	RMSE
DF-RNN	0.144 ± 0.019	0.101 ± 0.044	888.285 ± 278.798
DF-LDS	0.153 ± 0.016	0.135 ± 0.052	1012.551 ± 240.118

Table B.2. Comparison of DF-RNN and DF-LDS with level-trend ISSM on electricity over 50 runs.

We provide the result of one trial for Prophet in Table 2 since its classical methods have less variability between experiments and are more computationally expensive. The acronyms DA, P, MR, DF in these tables stand for DeepAR, Prophet, MQ-RNN and DeepFactor, respectively. Figure B.5 illustrates example forecasts from our model.

In our experiments, we do not tune the hyper-parameters of the methods. With tuning and a much bigger model, DeepAR would achieve higher accuracy in our preliminary experiments. For example, with the SageMaker HPO, we achieve the DeepAR's best performance of 0.118 MAPE and 0.042 P90 Loss on the `electricity` dataset for horizon 72. The hyper-parameters are selected to be a two-layer LSTM each with 40 units, a learning rate of 5e-3, 2000 epochs and an early stopping criteria of 200 epochs. As we can see, this metric is on par with our results in Table 2 and DF achieves such accuracy with much less number of parameters.
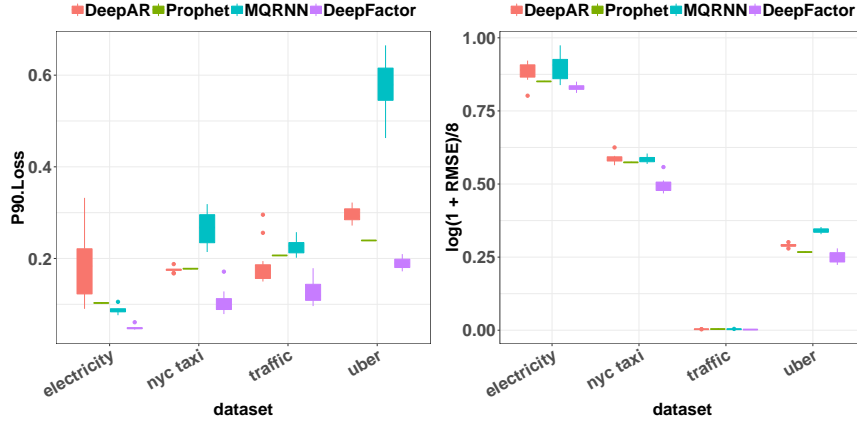

Figure B.3. P90QL and $\log(1 + \text{RMSE})$ normalized by 8 results for the forecast horizon 72 in Tables 2 and B.3, respectively. Purple denotes the proposed method.
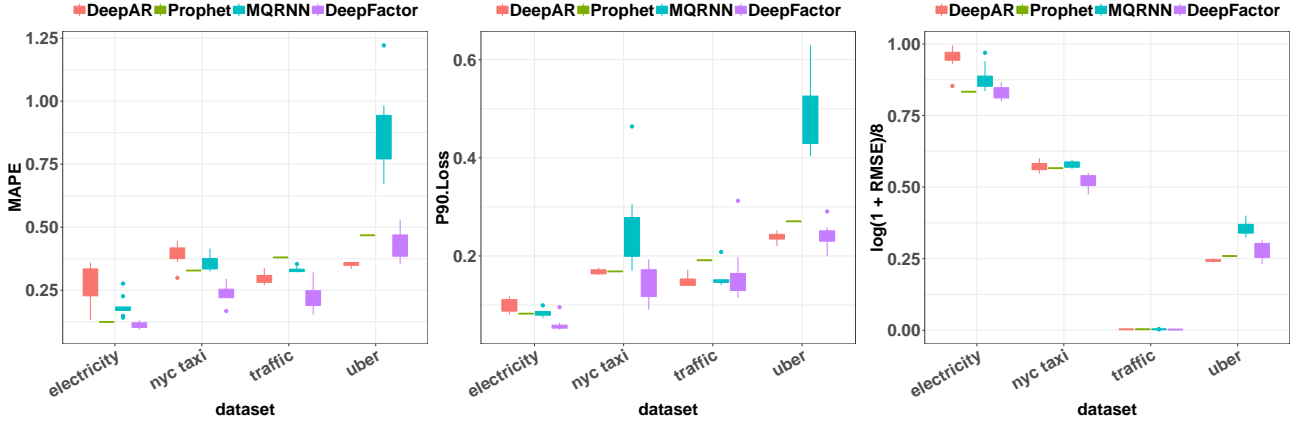

Figure B.4. P50QL (MAPE) P90QL and $\log(1 + \text{RMSE})/8.0$ (to make them in the same scale) results for forecast horizon 24 in Tables 2 and B.3, respectively. Purple denotes the proposed method.

DS	H	RMSE			
		DA	P	MR	DF
E	72	1194.421± 295.603	902.724	1416.992 ± 510.705	**770.437 ± 68.151**
	24	2100.927 ± 535.733	**783.598**	1250.233 ± 509.159	786.667 ± 144.459
N	72	108.963 ± 15.905	97.706	106.588 ± 9.988	**53.523 ± 13.096**
	24	98.545 ± 13.561	91.573	102.721 ± 9.640	**63.059 ± 11.876**
T	72	0.027 ± 0.001	0.032	0.029 ± 0.003	**0.019 ± 0.002**
	24	0.025 ± 0.001	0.028	0.028 ± 0.001	**0.021 ± 0.003**
U	72	9.218 ± 0.486	7.488	14.373 ± 0.997	**6.393 ± 1.185**
	24	**6.043 ± 0.295**	6.948	16.585 ± 3.452	8.044 ± 2.307

Table B.3. RMSE: Results for short-term (3-day forecast, 72-hour) and near-term (24-hour forecast) scenario with one week of training data.
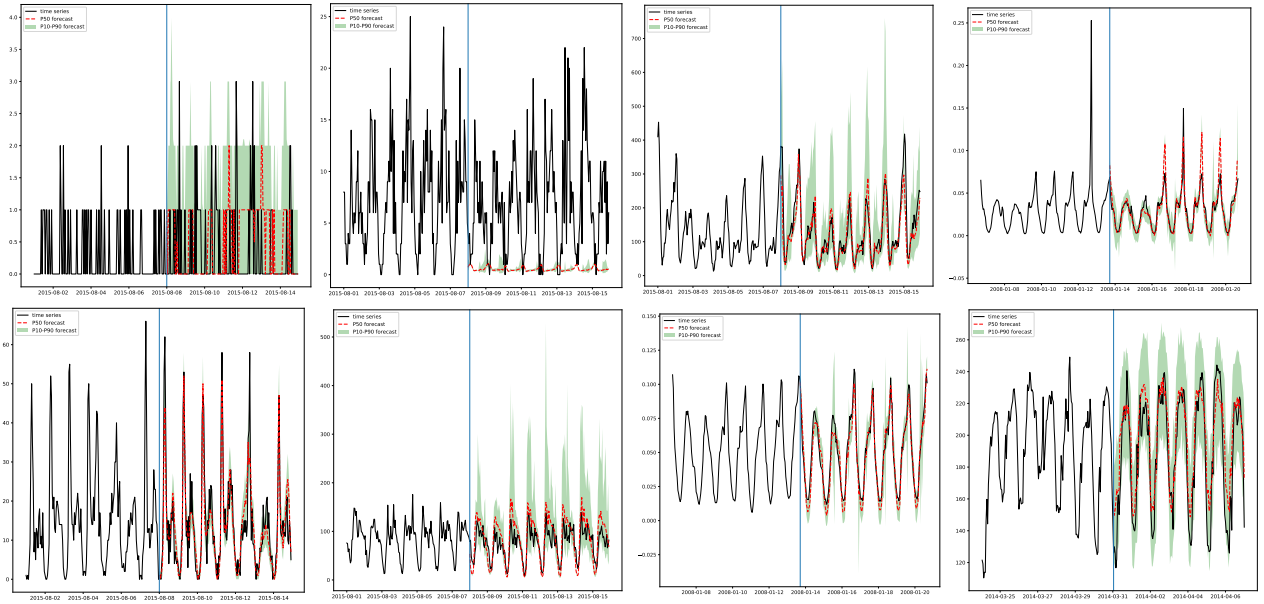

Figure B.5. Example forecast for (top row) `uber` (left) and `taxi` (center left) in the cold-start situation, and regular setting (center right) and `traffic`. For the `taxi` cold-start example, our model fails to produce a reasonable forecast, possibly due to insufficient information of corresponding latitude and longitude. Bottom role: another set of example forecasts for `uber,` `taxi,` `traffic` and `electricity.`