Time Series Forecasting Based on Variational Recurrent Model

Daniel Hsu

October 1, 2019

Abstract

In this paper, we use variational recurrent model to investigate the time series forecasting problem. Combining recurrent neural network (RNN) and variational inference (VI), this model has both deterministic hidden states and stochastic latent variables while previous RNN methods only consider deterministic states. Based on comprehensive experiments, we show that the proposed methods significantly improves the state-of-art performance of chaotic time series benchmark and has better performance on real-worl data. Both single-output and multiple-output predictions are investigated.

1 Introduction

Time series forecasting and modeling is an important interdisciplinary field of research, involving among others Computer Sciences, Statistics, and Econometrics. Made popular by Box and Jenkins [1] in the 1970s, traditional modeling procedures combine linear autoregression (AR) and moving average. But, since data are nowadays abundantly available, often complex patterns that are not linear can be extracted. So, the need for nonlinear forecasting procedures arises. Recurrent Neural Networks (RNNs) are powerful nonlinear models capable of exhibiting a rich set of dynamical behaviors [2].

Latent variable models also often used in time-series analysis. The core of these approaches is to assume that latent variables $\boldsymbol{z}_1, \ldots, \boldsymbol{z}_T \in \mathbb{R}^n$, which are correlated across t, underlie correlated observations $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_T \in \mathbb{R}^m$ [3]. Historically, the approaches based on dynamic Bayesian networks (DBN) are dominant, such as hidden Markov Models (HMMs) and Kalman filters. DBNs have typically been limited either to relatively simple state transition structures (e.g., linear models in the case of the Kalman filter) or to relatively simple internal state structure (e.g., the HMM state space consists of a single set of mutually exclusive states). So, the DBN based approaches cannot approximate intractable posterior, which results in the recent resurgence of the recurrent neural networks (RNN) based approaches.

RNNs are particularly suitable for modeling dynamical systems as they operate on input information as well as a trace of previously acquired information (due to recurrent connections) allowing for direct processing of temporal dependencies. RNNs can be employed for a wide range of tasks as they inherit their flexibility from plain neural networks. This includes universal approximation capabilities, since RNNs are capable of approximating any measurable sequence to sequence mapping and have been shown to be Turing complete [4].

However, the internal transition structure of the standard RNN is entirely deterministic. There is recent evidence that when complex sequences are modeled, the performances of RNNs can be dramatically improved when uncertainty is included in their hidden states [5] [6] [7].

In [6], authors propose to extend the variational autoencoder (VAE) into a recurrent framework for modeling complex sequences which is called variational RNN (VRNN). However they only show the performance of generating high-dimensional sequential data such as speech.

In this work, we investigate time series forecasting by VRNN model. In this model, the state transitions have both deterministic and stochastic parts. The deterministic transition is in LSTM cell, capturing long and short term dependencies, and the stochastic transition is from variational inference part, which can make the model robust to the uncertainty of input data points. The experiments show that this model can not only improve the performance of benchmark chaotic time series prediction, but also performs well on real-world data. Moreover, the performance of multi-output prediction doesn't degrade as the number of output increases, showing the proposed algorithm has strong scalability.

2 Preliminary

2.1 Recurrent Neural Network

RNNs are discrete-time statespace models trainable by specialized weight adaptation algorithms. The input to RNN is a variable-length sequence $\boldsymbol{x} = (\boldsymbol{x}_1, \dots, \boldsymbol{x}_T)$ which can be recursively processed. And when processing each symbol, RNN maintains its internal hidden state $\boldsymbol{h}$. The operation of RNN at each timestep t can be formulated as

$$\boldsymbol{h}_t = f_\theta(\boldsymbol{x}_t, \boldsymbol{h}_{t-1})$$

where f is the deterministic state transition function and θ is the parameter of f. In implementation, the function f can be realized by long short-term memory [8]. RNNs model sequences by parameterizing a factorization of the joint sequence probability distribution as a product of conditional probabilities such that:

$$\begin{aligned} p(\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_T) &= \prod_{t=1}^{T} p(\boldsymbol{x}_t | \boldsymbol{x}_{<t}) \\ p(\boldsymbol{x}_t | \boldsymbol{x}_{<t}) &= g_\tau(\boldsymbol{h}_{t-1}) \end{aligned} \tag{1}$$

where g is the function mapping the hidden state to the output distribution conditioned on previous observations, which is parameterized by τ.

The representational power of an RNN is limited by the output function g in (1). With the deterministic transition function f, the joint probabilities $p(\boldsymbol{x}_1, \dots, \boldsymbol{x}_T)$ which can be represented by the RNN are determined by the function g.

Inspired from [9], when forecasting complex and real-valued time series, the output function g can be modeled by Gaussian mixture model (GMM). With mixture coefficients α_t, means $\boldsymbol{\mu}_{\cdot,t}$ and covariance matrix $\Sigma_{\cdot,t}$, the output distribution conditioned on previous observations is

$$p_{\alpha_t, \boldsymbol{\mu}_{\cdot,t}, \Sigma_{\cdot,t}}(\boldsymbol{x}_t | \boldsymbol{x}_{<t}) = \sum_j \alpha_t \mathcal{N}(\boldsymbol{x}_t; \boldsymbol{\mu}_{j,t}, \Sigma_{j,t})$$

However, given that the state transition function is deterministic, when forecasting highly variable and highly structured time series, the variability embedded in the hidden states cannot be expressed, since the conditional output probability density is the only source of variability. So, it is necessary to introduce stochasticity into the hidden state transition. And in analyzing some highly structured time series, the hidden state $\boldsymbol{h}_t$ of RNNs is not expressive enough to capture the state transition dynamics. Some previous work have introduced extra latent variables to model the output functions [7] [6] [10]. Different from other work, following [6], this work makes the prior distribution of the latent random variable $\boldsymbol{z}_t$ at timestep t dependent on all the preceding inputs via the RNN hidden state $\boldsymbol{h}_{t-1}$ which can improve the representational power of the model.

2.2 Variational Autoencoder

Recently variational autoencoder (VAE) [11] has been shown to be a powerful tool to approximate the intractable complex posterior in the data space. The VAE introduces a set of latent random variables $\boldsymbol{z}$, designed to capture the variations underlie the observed variables $\boldsymbol{x}$. Typically VAE models the conditional $p(\boldsymbol{x}|\boldsymbol{z})$ as a highly flexible function approximator such as a neural network, which makes the inference of the posterior $p(\boldsymbol{z}|\boldsymbol{x})$ intractable. Thus VAE uses a variational approximation $q(\boldsymbol{z}|\boldsymbol{x})$ of the posterior, which introduces the evidence lower bound (ELBO):

$$\log p(\boldsymbol{x}) \geq -\mathrm{KL}(q(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})) + \mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}[\log p(\boldsymbol{x}|\boldsymbol{z})] \tag{2}$$

where $\mathrm{KL}(P|Q)$ is the Kullback-Leibler divergence between two distributions P and Q.

In [11], the approximate posterior $q(\boldsymbol{z}|\boldsymbol{x})$ is a Gaussian $\mathcal{N}(\boldsymbol{\mu}, \mathrm{diag}(\boldsymbol{\sigma}^2))$ and the mean $\boldsymbol{\mu}$ and variance $\boldsymbol{\sigma}^2$ are modeled as the output of neural networks of $\boldsymbol{x}$. The prior $p(\boldsymbol{z})$ is assumed to simple standard Gaussian distribution. The training process is to maximize the ELBO, which can yield optimal selection of parameters for generative model $p(\boldsymbol{x}|\boldsymbol{z})$ and inference model $q(\boldsymbol{z}|\boldsymbol{x})$. Based on reparametrizing trick, we formulate $\boldsymbol{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon}$ and rewrite:

$$\mathbb{E}_{q(\boldsymbol{z}|\boldsymbol{x})}[\log p(\boldsymbol{x}|\boldsymbol{z})] = \mathbb{E}_{p(\boldsymbol{\epsilon})}[\log p(\boldsymbol{x}|\boldsymbol{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \odot \boldsymbol{\epsilon})]$$

where $\boldsymbol{\epsilon}$ is drawn from standard Gaussian distribution. Then, both generative and inference model can be trained by standard backpropagation technique based on gradient descent.

3 Time Series Forecasting Model

In this work, we use similar model as VRNN [6] which is a recurrent extension of VAE. By LSTM cells, this model explicitly captures both the long and short-term dependencies between latent variables across timesteps. By introducing stochastic latent variables, this model is expressive enough to capture the highly variate and highly structured dynamics embedded in the complex time series.

3.1 Generative Model

This model applies VAE in a recurrent setting. At each timestep, different conventional VAE, the prior of latent variables is dependent on the hidden state $\boldsymbol{h}_{t-1}$ of RNN, which makes the output function incorporate the temporal structure of the time series. Then we have:

$$\boldsymbol{z}_t \sim \mathcal{N}(\boldsymbol{\mu}_{0,t}, \mathrm{diag}(\boldsymbol{\sigma}_{0,t}^2)), \text{ where } [\boldsymbol{\mu}_{0,t}, \boldsymbol{\sigma}_{0,t}] = \varphi_\tau^{\mathrm{prior}}(\boldsymbol{h}_{t-1}) \tag{3}$$

where $\boldsymbol{\mu}_{0,t}$ and $\boldsymbol{\sigma}_{0,t}$ denote the mean and variance of the prior distribution. Then, the generating distribution will be conditioned on both latent variable $\boldsymbol{z}_t$ and hidden state $\boldsymbol{h}_{t-1}$ such that:

$$\boldsymbol{x}_t|\boldsymbol{z}_t \sim \mathcal{N}(\boldsymbol{\mu}_{x,t}, \mathrm{diag}(\boldsymbol{\sigma}_{x,t}^2)), \text{ where } [\boldsymbol{\mu}_{x,t}, \boldsymbol{\sigma}_{x,t}] = \varphi_\tau^{\mathrm{dec}}(\varphi_\tau^{\boldsymbol{z}}(\boldsymbol{z}), \boldsymbol{h}_{t-1}) \tag{4}$$

where $\boldsymbol{\mu}_{x,t}$ and $\boldsymbol{\sigma}_{x,t}$ denote the mean and variance of the generating distribution, $\varphi_{\tau}^{\text{prior}}$ and $\varphi_{\tau}^{\text{dec}}$ can be modeled by output of neural networks. The latent variables at the same timestep are assumed to be independent with each other, which are in mean-field family. In order to improve expressive capability, this model uses feature extractor $\varphi_{\tau}^{\boldsymbol{z}}$ to further utilize the dependencies among latent variables at the same timestep. We also apply feature extractor $\varphi_{\tau}^{\boldsymbol{x}}$ to input $\boldsymbol{x}$ before using it, which can discard some redundant information to make the learning more efficient. Both $\varphi_{\tau}^{\boldsymbol{z}}$ and $\varphi_{\tau}^{\boldsymbol{x}}$ are implemented by neural networks in experiments. Inside the RNN, the hidden state is updated as follows:

$$\boldsymbol{h}_t = f_\theta(\varphi_{\tau}^{\boldsymbol{x}}(\boldsymbol{x}_t), \varphi_{\tau}^{\boldsymbol{z}}(\boldsymbol{z}_t), \boldsymbol{h}_{t-1}) \tag{5}$$

where transition function f parameterized by θ can be implemented by LSTM cell to capture both long and short-term temporal dependencies. From (5), we know that hidden state $\boldsymbol{h}_t$ is dependent on $\boldsymbol{x}_{\leq t}$ and $\boldsymbol{z}_{\leq t}$. The distributions (3) and (4) define $p(\boldsymbol{z}_t|\boldsymbol{x}_{<t}, \boldsymbol{z}_{<t})$ and $p(\boldsymbol{x}_t|\boldsymbol{z}_{\leq t}, \boldsymbol{x}_{<t})$ respectively. We can see that the parameterization of this generative model is motivated by the factorization of joint probability as below:

$$p(\boldsymbol{x}_{\leq T}, \boldsymbol{z}_{\leq T}) = \prod_{t=1}^{T} p(\boldsymbol{x}_t|\boldsymbol{z}_{\leq t}, \boldsymbol{x}_{<t}) p(\boldsymbol{z}_t|\boldsymbol{x}_{<t}, \boldsymbol{z}_{<t}) \tag{6}$$

3.2 Inference Model

Based on Gaussian assumption, the approximate posterior is a function of both input $\boldsymbol{x}_t$ and hidden state $\boldsymbol{h}_{t-1}$ as below:

$$\boldsymbol{z}_t|\boldsymbol{x}_t \sim \mathcal{N}(\boldsymbol{\mu}_{z,t}, \text{diag}(\boldsymbol{\sigma}_{z,t}^2)), \text{ where } [\boldsymbol{\mu}_{z,t}, \boldsymbol{\sigma}_{z,t}] = \varphi_{\tau}^{\text{enc}}(\varphi_{\tau}^{\boldsymbol{x}}(\boldsymbol{x}), \boldsymbol{h}_{t-1}) \tag{7}$$

where $\boldsymbol{\mu}_{z,t}$ and $\boldsymbol{\sigma}_{z,t}$ denote the mean and variance of the approximate posterior. Conditioning on $\boldsymbol{h}_{t-1}$, the posterior follows the factorization:

$$q(\boldsymbol{z}_{\leq T}|\boldsymbol{x}_{\leq T}) = \prod_{t=1}^{T} q(\boldsymbol{z}_t|\boldsymbol{x}_{\leq t}, \boldsymbol{z}_{<t}) \tag{8}$$

3.3 Objective Function

Applying ELBO at each timestep, based on factorizations (6) and (8), we have the accumulative ELBO as below:

$$\mathbb{E}_{q(\boldsymbol{z}_{\leq T}|\boldsymbol{x}_{\leq T})}\left[\sum_{t=1}^{T}(-\text{KL}(q(\boldsymbol{z}_t|\boldsymbol{x}_{\leq T}, \boldsymbol{z}_{<T})\|p(\boldsymbol{z}_t|\boldsymbol{x}_{<T}, \boldsymbol{z}_{<T})) + \log p(\boldsymbol{z}_t|\boldsymbol{x}_{<T}, \boldsymbol{z}_{\leq T}))\right] \tag{9}$$

which is learning objective function of our model. With all functions implemented by neural networks, the accumulative ELBO (9) is maximized by ADAM [12] algorithm. The optimal generative and inference model can be learned simultaneously.

4 Simulation Results

This section presents an experimental study that evaluates the performance of the proposed method in terms of predictive ability. The performance is compared with other neural time-series models, such as the an RNN trained with the extended Kalman filter (EKF) [17], robust echo state network (RESN) [18] and co-evolutionary multi-task learning (CMTL) [19]. These are cutting-edge methods for time series modeling for both single-output and multiple-output predictions. RNN-EKF is a classical method for time-series prediction which estimates the state transition by Kalman filter. RESN is a robust extension of echo state network (ESN) method. The basic idea of RESN is to train ESN in a Bayesian framework, while replacing the Gaussian likelihood function with a Laplace likelihood function, which is less sensitive to outliers and can enhance the robustness of the model. CMTL employs a divide and conquer approach for training neural networks and predictive recurrence to feature knowledge from previous states.

The experiments are conducted using three simulated and one real-world benchmark time series. The simulated time series are chaotic nonlinear systems such as Mackey and Glass [13], Lorenz [14], and Rossler [15]. One real world dataset is individual household electric power consumption dataset [16]. All the respective time series data sets are scaled in the range $[0, 1]$ in order to be used for sigmoid units in the feedforward neural network. We evaluate the performance of one-step, 5-step, and 10-step ahead prediction. The performance is measured by the root-mean-squared error (RMSE):

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}$$

where $y_i, \hat{y}_i$ are observed and predicted data.

4.1 Chaotic Processes Prediction

Across four simulations in this section, the length of training sequence is 1000, and the length of testing sequence is 500. The optimization is performed by ADAM with learning rate 0.0005.

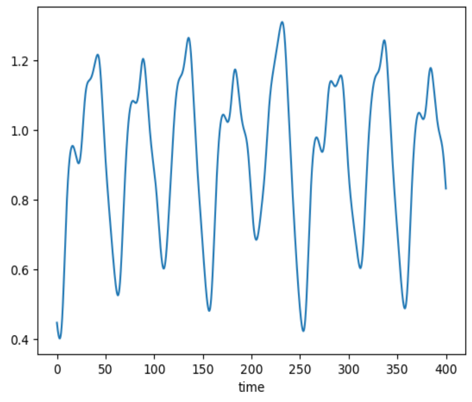
4.1.1 Mackey and Glass Time Series

We first experiment on Mackey-Glass time series [13], which is derived from differential equation as below:

$$\frac{dx}{dt} = \beta x(t) + \frac{\alpha x(t-\delta)}{1 + x(t-\delta)^{10}} \tag{10}$$

The parameters are chosen to be $\alpha = 0.2, \beta = -0.1$ and $\delta = 17$. The dataset is constructed by second-order Runge-Kutta method with the step size of 0.1, with plot as below.

Figure 1: Plot of Mackey and Glass Time Series.



For one-step ahead prediction, the input is a sequence of 5 samples with stride of 6. For 5-step ahead prediction, the input is a sequence of 15 samples with stride of 6. For 10-step ahead prediction, the input is a sequence of 10 samples with stride of 6. The RMSE and its variance of the proposed method (VRNN), EKF, RESN and CMTL are shown in tables in the following.

	VRNN	RNN-EKF	RESN	CMTL
RMSE	0.006778	0.012218	0.010233	0.008231
Variance	0.011232	0.015124	0.022553	0.009421

Table 1: Performance Comparison for One-step Ahead Prediction

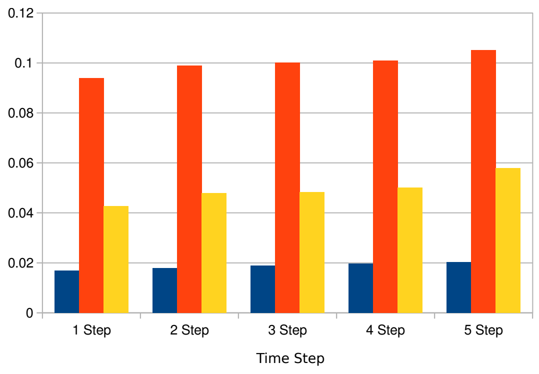


Figure 2: RMSE for 5-step Ahead Prediction

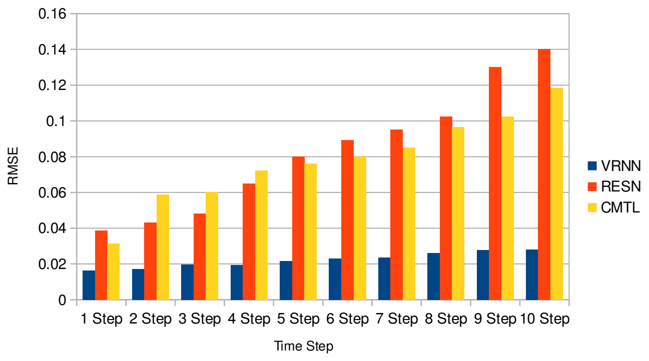


Figure 3: RMSE for 10-step Ahead Prediction

	VRNN	RESN	CMTL
1 Step	0.012839	0.022342	0.021562
2 Step	0.013292	0.026897	0.019023
3 Step	0.013740	0.035123	0.015021
4 Step	0.014451	0.045213	0.010228
5 Step	0.015305	0.054234	0.009342

Table 2: Variance for 5-step Ahead Prediction

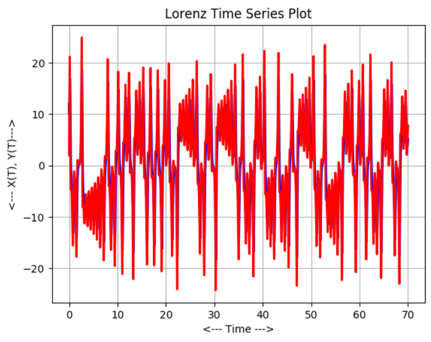
The variance of VRNN for 10-step ahead prediction is 0.011232, which is comparable with CMTL and RESN. The complete variance table is omitted.

4.1.2 Lorenz Time Series

The Lorenz attractor [14] is a nonlinear 3-D system that provides a simplified model of convection in the atmosphere. The Lorenz equations are given by:

$$\begin{aligned}\frac{dx}{dt} &= \sigma(y-x)\\ \frac{dy}{dt} &= \rho x - y - xz\\ \frac{dz}{dt} &= xy - \beta z\end{aligned}$$

In this experiment, the parameters are set to be $\sigma = 10, \rho = 28$ and $\beta = 8/3$. The 3-D attractor and x, y components are plotted as below.



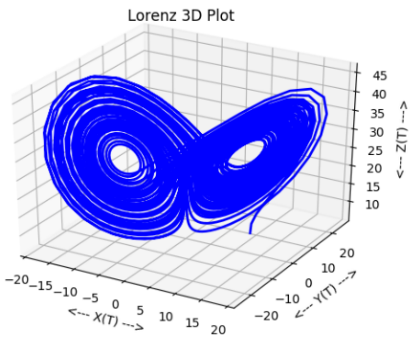


Figure 4: Plots of Lorenz Attractor

The performance of one-step ahead prediction is shown in the following table.

	VRNN	RNN-EKF	RESN	CMTL
RMSE	0.008772	0.043278	0.020632	0.018992
Variance	0.012102	0.011142	0.015234	0.009567

Table 3: Performance Comparison for One-step Ahead Prediction

From (4), at time t, the Bayesian confidence interval can be defined as $\boldsymbol{\mu}_{x,t} \pm z_{0.0025}\boldsymbol{\sigma}_{x,t}$, where $(\boldsymbol{\mu}_{x,t}, \boldsymbol{\sigma}_{x,t})$ are mean and variance of the generating distribution and $z_{0.0025}$ is the critical point of the normal distribution. These are bands within which the true underlying process is expected to fall with 95% confidence.

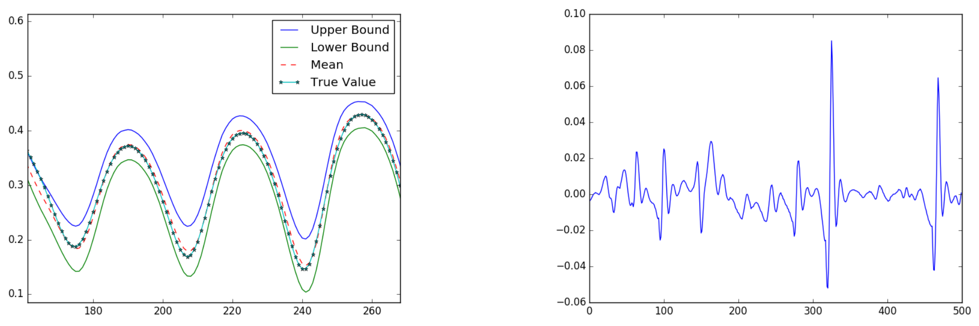


Figure 5: One-step prediction performance on test data. Left: plots of confidence interval and mean of generating distribution. Right: plot of error, which is the difference between the mean of generating distribution and ground truth.

For 5-step ahead, we use 6 samples with stride of 6. For 10-step ahead, we use 10 samples with stride of 4. The results for 5-step and 10-step ahead prediction are shown below.

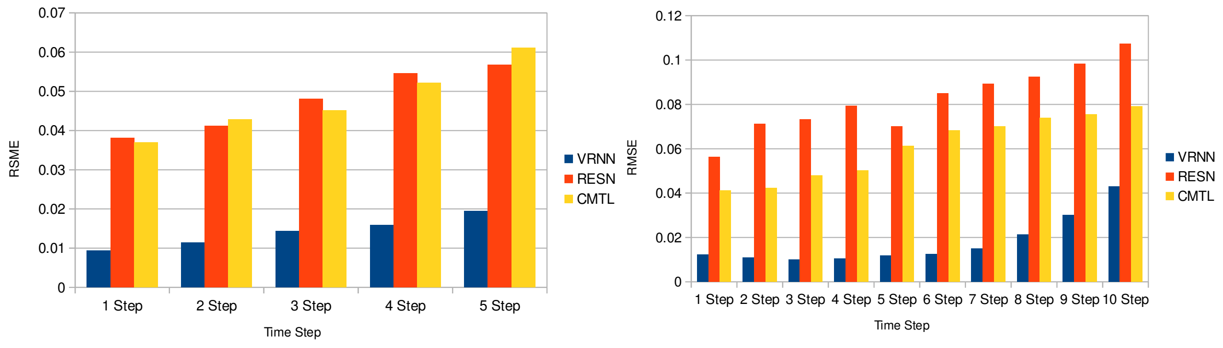


Figure 6: RMSE for 5-step and 10-step Ahead Prediction on Lorenz

	VRNN	RESN	CMTL
1 Step	0.011467	0.015583	0.014023
2 Step	0.010589	0.018234	0.012882
3 Step	0.012034	0.018199	0.010568
4 Step	0.012589	0.019202	0.010228
5 Step	0.014892	0.022021	0.011772

Table 4: Variance for 5-step Ahead Prediction

4.1.3 Rossler Time Series

The Rossler attractor [15] is generated by a system of three differential equations as below:

$$
\begin{aligned}
\frac{dx}{dt} &= -z - y \\
\frac{dy}{dt} &= x + ay \\
\frac{dz}{dt} &= b + z(x - c)
\end{aligned}
$$

where coefficients are set to be $a = 0.2, b = 0.2$ and $c = 4.6$. The figure below plots the 3-D Rossler Attractor and its x, y components.

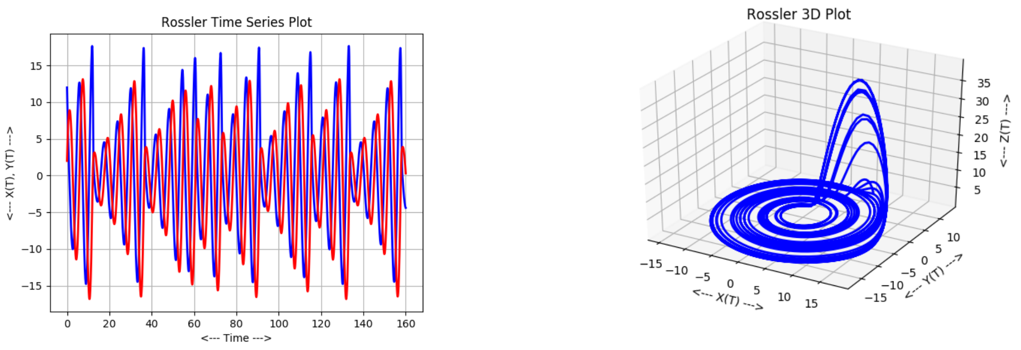


Figure 7: Plots of Rossler Attractor. Right: blue line represents x and red line shows y.

For one-step ahead prediction, we use 5 samples with stride of 6. The plots of prediction results are shown in the following figure.

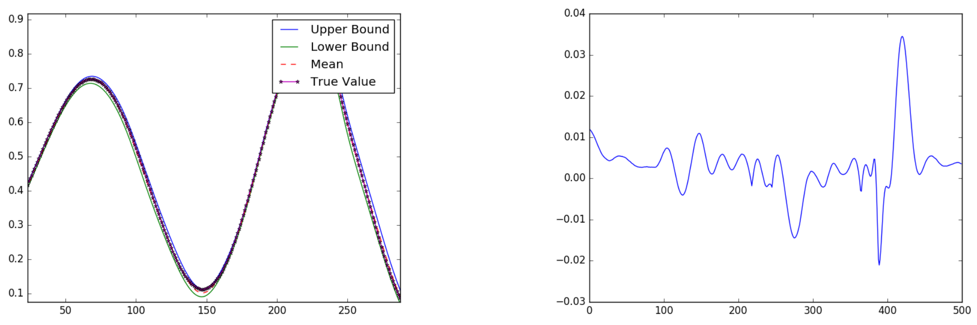


Figure 8: One-step prediction performance on test data. Left: plots of confidence interval, mean of generating distribution and ground truth. Right: prediction error.

The performance comparison for one-step ahead prediction is shown as below:

	VRNN	RNN-EKF	RESN	CMTL
RMSE	0.007824	0.043278	0.020632	0.038992
Variance	0.010558	0.009832	0.011342	0.012421

The performance comparison for 5-step and 10-step ahead predictions are shown below.

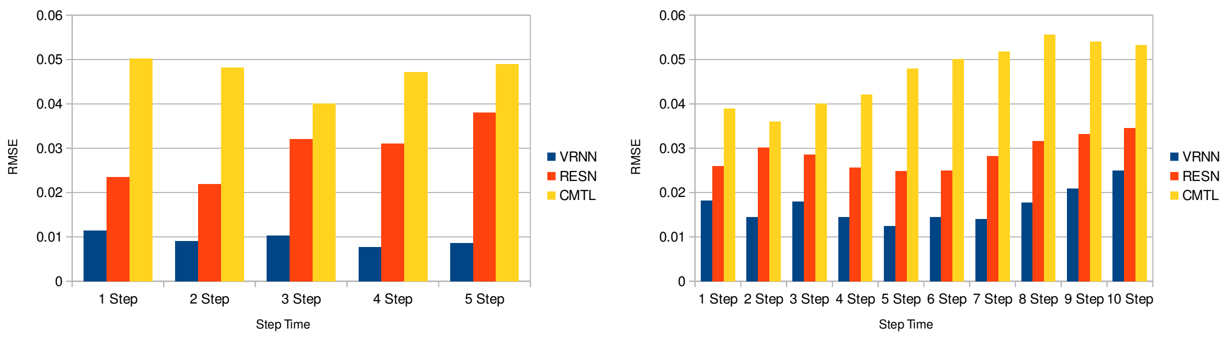


Figure 9: RMSE for 5-step and 10-step Ahead Prediction on Rossler

The variance of VRNN is 0.103214, and the complete table for variance is omitted.

4.2 Power Consumption Data

This dataset contains measurements of electric power consumption in one household with a one-minute sampling rate over a period of almost 4 years [16], which is equivalent to 7000 data points. The first 1000 data points are used for training and the remaining for testing.

The experiment shows that VRNN model can also work well on real-world data. The following plots show the estimated value and its error, compared with ground truth.

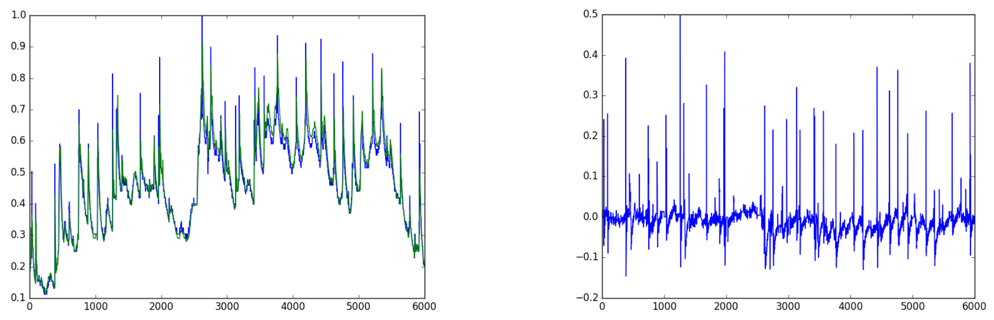
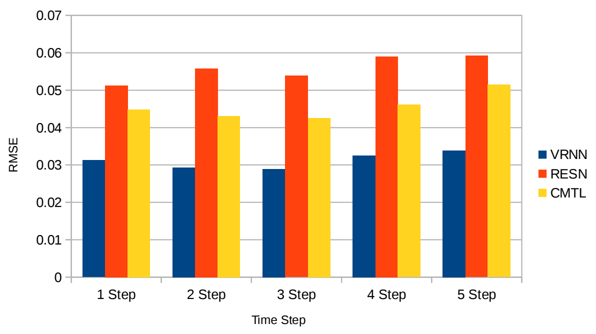


Figure 10: One-step prediction performance on test data. Left: plots of predicted data (green) and ground truth (blue). Right: plot of prediction error.

	VRNN	RNN-EKF	RESN	CMTL
RMSE	0.021562	0.045586	0.040211	0.051520

Table 5: One-step Ahead Prediction Preformance Comparison



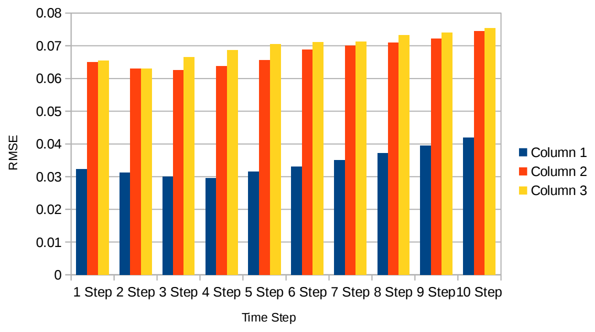


Figure 11: RMSE for 5-step and 10-step Ahead Prediction on Power Data

Comparing the performance of experiments with different number of output, we can see that performance of VRNN model doesn't degrade in multi-output cases, which proves the scalability in practical applications.

5 Conclusion

This paper shows the advantage of VRNN model on time series prediction. With comprehensive experiments, we prove that the VRNN model can outperform cutting-edge algorithm on time series prediction and improve the state-of-art performance on chaotic benchmarks. Moreover, the predictions from VRNN model have comparable variance with previous methods, resulting in small confidence intervals. Comparing performance of different number of prediction outputs, we can see the performance of 10-step prediction doesn't degrade too much, showing that the proposed method has strong ability of scaling to multi-output prediction. In the future, we are going to apply VRNN to predict the financial data which has more variance and uncertainty.

References

[1] G. Box and G. Jenkins, Time Series Analysis: Forecasting and Control. San Francisco, CA: Holden-Day, 1970.

[2] G. Deco and B. Schurmann, Neural learning of chaotic dynamics. Neural Process Lett., vol. 2, no. 2, pp. 23-26, 1995.

[3] Archer, Evan, et al. Black box variational inference for state space models. arXiv preprint arXiv:1511.07367 (2015).

[4] B. Hammer. On the approximation capability of recurrent neural networks. Neurocomputing, 31(1):107-123, 2000.

[5] J. Bayer and C. Osendorfer. Learning stochastic recurrent networks. arXiv:1411.7610, 2014.

[6] J. Chung, K. Kastner, L. Dinh, K. Goel, A. C. Courville, and Y. Bengio. A recurrent latent variable model for sequential data. In NIPS, pages 2962-2970, 2015.

[7] N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent. Modeling temporal dependencies in high-dimensional sequences: Application to polyphonic music generation and transcription. arXiv:1206.6392, 2012.

[8] S. Hochreiter and J. Schmidhuber. Long short-term memory. Neural Computation, 9(8):1735-1780, 1997.

[9] A. Graves. Generating sequences with recurrent neural networks. arXiv preprint arXiv:1308.0850, 2013.

[10] J. Bayer and C. Osendorfer. Learning stochastic recurrent networks. arXiv preprint arXiv:1411.7610, 2014.

[11] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In Proceedings of the International Conference on Learning Representations (ICLR), 2014.

[12] D. Kingma and J. Ba. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980.

[13] M. Mackey and L. Glass. Oscillation and chaos in physiological control systems, Science 197 (4300), pages 287-289, 1977.

[14] E. Lorenz. Deterministic non-periodic flows, J. Atmos. Sci., vol. 20, pages 267-285, 1963.

[15] O. E. Rossler. An equation for continuous chaos. Physics Letters A, 57(5), 397-398, 1976.

[16] UCI Machine Learning Repository. Available at http://archive.ics.uci.edu/ml/datasets

[17] G. V. Puskorius and L. A. Feldkamp, Neurocontrol of nonlinear dynamical systems with Kalman filter trained recurrent networks, IEEE Trans. Neural. Netw., vol. 5, no. 2, pages 279-297, Mar. 1994.

[18] D. Li, M. Han, and J. Wang. Chaotic time series prediction based on a novel robust echo state network." IEEE Transactions on Neural Networks and Learning Systems 23.5, pages 787-799, 2012.

[19] R. Chandra, Y.-S. Ong, and C.-K. Goh. Co-evolutionary multi-task learning for dynamic time series prediction." arXiv preprint arXiv:1703.01887, 2017.