TIME SERIES PREDICTION VIA RECURRENT NEURAL NETWORKS WITH THE INFORMATION BOTTLENECK PRINCIPLE

Duo Xu *Faramarz Fekri*

Department of Electrical and Computer Engineering
Georgia Institute of Technology
Atlanta, GA, 30332

ABSTRACT

In this paper, we propose a novel method for probabilistic time series prediction based on Recurrent Information Bottleneck (RIB). We propose to incorporate the stochastic latent states for modeling complex and non-linear time series optimized by RIB objective. Compared with previous work, the proposed method can yield better prediction and uncertainty estimation. It's built on the extension of information bottleneck principle to recurrent setting, to find the stochastic latent state maximally informative about the target with low complexity. The experiments over real-world datasets show the proposed method can outperform the state-of-the-art prediction performance on single and multi-dimensional data.

***Index Terms*—** prediction, recurrent neural network, latent state, information bottleneck, high dimension

1. INTRODUCTION

Time series prediction and modeling is an important interdisciplinary topic in computer sciences, statistics, and econometrics. In time series prediction, the deep neural networks [1] have recently received enormous interests among researchers. Deep belief networks (DBN) are frequently employed in current short-term traffic forecasting [2]. Moreover pre-training strategies with unsupervised learning algorithms such as Restricted Boltzmann machine (RBM) [3] and Stacked AutoEncoder (SAE) [4] are also used for time series prediction. However, these deep architectures cannot capture long-term dependencies at different time scales.

RNNs are particularly suitable for modeling dynamical systems as they operate on input information as well as a trace of previously acquired information. Among all RNN architectures, the most successful models to characterize long-term memory are the Long Short-Term Memory (LSTM) [5] and its variant Gated Recurrent Unit (GRU) [6]. However, these models still have some disadvantages, as their prediction is largely based on recent observations [7]. Further few of these methods can give uncertainty estimation on future values.

In this work, we perform time series prediction by using stochastic RNN trained using the Recurrent Information Bottleneck (RIB) as objective function. This objective function is to maximize the mutual information between latent state and the target with low latent complexity. In the proposed model, the input is first encoded to a stochastic latent state, then fed into the state transition in the RNN cell to generate next hidden state. Combining the hidden state and the latent state together, the decoder generates the distribution of the predicted value of time series. Our model has two advantages: i) the incorporation of stochastic latent state can augment the RNN model to better utilize recent observations and better model complicated and high-dimensional time series, ii) the incorporation of RIB helps to train the latent state to be maximally informative about the target with small complexity. To our knowledge, this is the first work to apply information bottleneck principle in time series prediction.

Based on experiments on real-world data, the proposed method not only improves the state-of-the-art accuracy, but also gives reasonable estimations on future uncertainty, such as confidence interval.

2. PRELIMINARY

2.1. Information Bottleneck

The information bottleneck (IB) [10] is an information-theoretic view of deep networks. To deploy IB, suppose the internal representation of some intermediate layers is regarded as a stochastic latent state Z of the input X, defined by an encoder $p(\boldsymbol{z}|\boldsymbol{x};\boldsymbol{\theta})$ parametrized by $\boldsymbol{\theta}$. The Markov chain $Y \leftrightarrow X \leftrightarrow Z$ holds [10]. The information bottleneck is formulated to learn an encoding that is maximally informative about the target Y, measured by the mutual information between the latent and the target $I(Z,Y;\boldsymbol{\theta})$ as:

$$I(Z,Y;\boldsymbol{\theta}) = \int p(z,y|\boldsymbol{\theta}) \log \frac{p(z,y|\boldsymbol{\theta})}{p(z|\boldsymbol{\theta})p(y|\boldsymbol{\theta})} dzdy$$

In order to find a representation with low complexity, a natural and useful constraint is applied to the mutual information between the latent random variable and the original data, $I(X,Z) < \epsilon$, where ϵ is a pre-defined constraint. Hence, we

978-1-5386-3512-4/18/$31.00 ©2018 IEEE

$$
\begin{aligned}
&I(Z_1 \dots Z_T, Y_1 \dots Y_T) \\
\overset{(a)}{\geq} \quad & \int p(y_1 \dots y_T, z_1 \dots z_T) \log \frac{q(y_1 \dots y_T | z_1 \dots z_T)}{p(y_1 \dots y_T)} dy_1 \cdots dy_T dz_1 \cdots dz_T \\
= \quad & \int p(y_1 \dots y_T, z_1 \dots z_T) \sum_{t=1}^{T} \log q(y_t|z_t) dy_1 \cdots dy_T dz_1 \cdots dz_T - \int p(y_1 \cdots y_T) \log p(y_1 \cdots y_T) dy_1 \cdots dy_T \\
= \quad & \sum_{t=1}^{T} \int p(y_t, z_t) \log q(y_t|z_t) dy_t dz_t + H(Y_1 \dots Y_T) \\
\overset{(b)}{=} \quad & \sum_{t=1}^{T} \int p(x_t) p(y_t|x_t) p(z_t|x_t) \log q(y_t|z_t) dx_t dy_t dz_t + H(Y_1 \dots Y_T) \qquad (1)
\end{aligned}
$$

have the following optimization problem:

$$\max_{\boldsymbol{\theta}} I(Z, Y; \boldsymbol{\theta}) \quad \text{s.t.} \quad I(X, Z; \boldsymbol{\theta}) < \epsilon$$

By introducing a Lagrange multiplier β, we have the objective

$$L_{IB}(\boldsymbol{\theta}) = I(Z, Y; \boldsymbol{\theta}) - \beta I(Z, X; \boldsymbol{\theta}) \qquad (2)$$

where $\beta \geq 0$ controls the trade-off between the expressiveness of Z about Y and the compression of Z about X. This is the information bottleneck (IB) principle first proposed by Tishby [9]. In our work, the training objective of the RNN is formulated by IB in recurrent setting.

2.2. Gated Recurrent Unit

The LSTM neural network is adopted in this study to model time series. In order to resolve the vanishing gradient problem of RNN, LSTM was initially introduced in [5], which can model long-term dependencies and capture the temporal correlation at different time scales. Recently a variant of LSTM denoted as Gated Recurrent Unit (GRU) was proposed [6]. Compared with LSTM, it has simpler structure and competitive performance. Similar to the LSTM unit, the GRU has gating units that modulate the information flow inside the unit, but, without having a separate memory cells. The operations in the GRU cell are described as below,

$$
\begin{aligned}
\boldsymbol{r}_t &= \sigma(\boldsymbol{W}_r \boldsymbol{x}_t + \boldsymbol{U}_r \boldsymbol{h}_{t-1} + \boldsymbol{b}_r) \\
\boldsymbol{u}_t &= \sigma(\boldsymbol{W}_u \boldsymbol{x}_t + \boldsymbol{U}_u \boldsymbol{h}_{t-1} + \boldsymbol{b}_u) \\
\boldsymbol{c}_t &= \sigma(\boldsymbol{W}_c \boldsymbol{x}_t + \boldsymbol{U}_c (\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}) + \boldsymbol{b}_c) \\
\boldsymbol{h}_t &= \boldsymbol{u}_t \odot \boldsymbol{h}_{t-1} + (1 - \boldsymbol{u}_t) \odot \boldsymbol{c}_t
\end{aligned}
\qquad (3)
$$

where $\boldsymbol{x}_t, \boldsymbol{h}_t, \boldsymbol{r}_t, \boldsymbol{u}_t$ and $\boldsymbol{c}_t$ are the input, the hidden (activation) state of GRU cell, the reset gate, the forgetting gate, and the candidate activation, respectively. $\boldsymbol{W}_., \boldsymbol{U}_.$ are weight matrices and the coefficients $\boldsymbol{b}_.$ are bias in the state transition, all of which need to be learned in training. Here $\odot$ is the elementwise multiplication.

The hidden state $\boldsymbol{h}_t$ of GRU is the linear interpolation between the previous hidden state and candidate activation $\boldsymbol{c}_t$, where a forgetting gate $\boldsymbol{u}_t$ controls the degree by which the unit forgets its previous hidden state. This procedure of taking a linear sum between the existing state and the newly computed state is similar to the LSTM unit. The GRU, however, does not have any mechanism to control the degree to which its state is exposed, but exposes the whole state each time. The reset gate $\boldsymbol{r}_t$ controls the influence of the previous hidden state on the candidate activation.

3. METHOD

3.1. Recurrent Information Bottleneck

We first establish the recurrent information bottleneck principle for the time series prediction, and then incorporate it into the training. We model the original and target time series as random processes $X_t, Y_t \in \mathbb{R}^N, t = 1, \dots, T$ respectively. At each time t, x_t, y_t are realizations of random variables $X_t, Y_t \in \mathbb{R}^N$. Assume at each time t, X_t has a stochastic latent state Z_t, satisfying the Markov chain $Y_t \leftrightarrow X_t \leftrightarrow Z_t$. For time series prediction, the information bottleneck principle can be formulated as $I(Z_1 Z_2 \dots Z_T, Y_1 Y_2 \dots Y_T) - \beta I(Z_1 Z_2 \dots Z_T, X_1 X_2 \dots X_T)$, termed as recurrent information bottleneck (RIB) objective. We will analyze these two mutual information and make them analytically tractable.

Since Z_t is only dependent on X_t, we have $p(Z_1 \dots Z_T | X_1 \dots X_T) = \prod_{t=1}^{T} p(Z_t|X_t)$, where $p(Z|X)$ can be a stochastic encoder realized by a multi-layer neural network. Let's examine the first term in RIB, i.e., $I(Z_1 \dots Z_T, Y_1 \dots Y_T)$. Note that the optimal decoding (prediction) distribution $p(Y|Z)$ is determined by the encoder and the Markov chain above, so it's intractable in practice. We use a multivariate Gaussian distribution $q(Y|Z)$ as a variational approximation for $p(Y|Z)$, whose mean and variance are generated by a multi-layer neural network . Then we can have the lower bound in (1) above, where (a) holds since the difference between the left and right side of (a) is equal to,

$$\begin{aligned}
&\int p(y_1 \dots y_T, z_1 \dots z_T) \log \frac{p(y_1 \dots y_T | z_1 \dots z_T)}{q(y_1 \dots y_T | z_1 \dots z_T)} \\
&\quad \cdot dz_1 \cdots dz_T dy_1 \cdots dy_T \\
&= \mathrm{KL}\big[p(Y_1 \dots Y_T | Z_1 \dots Z_T), q(Y_1 \dots Y_T | Z_1 \dots Z_T)\big] \\
&\geq 0
\end{aligned}$$

Further (b) in (1) holds due to Markov chain $Y_t \leftrightarrow X_t \leftrightarrow Z_t$. Since the optimal prior distribution of latent state $p(z_1 \dots z_T)$ in RIB objective is difficult to compute, we adopt the product distribution $\prod_{t=1}^{T} \tilde{p}_t(z_t)$ as the variational approximation to $p(z_1 \dots z_T)$. In practice at each time $\tilde{p}_t(z_t)$ is formulated by $\tilde{p}(z_t | \boldsymbol{h}_{t-1})$, a multi-layer neural network whose the input is the previous RNN hidden state $\boldsymbol{h}_{t-1}$. The temporal dependency of z_t is modeled by RNN states.

Now let's examine the second term in RIB, $I(Z_1 \dots Z_T, X_1 \dots X_T)$ as in (4). Here (i) is due to the positivity of the KL divergence, and (ii) is due to that z_t is only dependent on x_t at the encoder.

$$\begin{aligned}
&I(Z_1 \dots Z_T, X_1 \dots X_T) \\
&= \int p(x_1 \dots x_T, z_1 \dots z_T) \log \frac{p(z_1 \dots z_T | x_1 \dots x_T)}{p(z_1 \dots z_T)} \\
&\quad \cdot dz_1 \cdots dz_T dx_1 \cdots dx_T \\
&\overset{(i)}{\leq} \int p(x_1 \dots x_T, z_1 \dots z_T) \log \frac{p(z_1 \dots z_T | x_1 \dots x_T)}{\prod_{t=1}^{T} \tilde{p}_t(z_t)} \\
&\quad \cdot dz_1 \cdots dz_T dx_1 \cdots dx_T \\
&\overset{(ii)}{=} \sum_{t=1}^{T} \int p(x_t) p(z_t | x_t) \log \frac{p(z_t | x_t)}{\tilde{p}_t(z_t)} dz_t dx_t \qquad (4)
\end{aligned}$$

Denote N as the number of time series, x_{nt}, y_{nt} as the realization of n-th time series of X, Y at time t. Note that z_{nt} is the realization of latent state Z_{nt} of x_{nt} sampled from the stochastic encoder $p(z_{nt} | x_{nt})$. By approximating joint distribution $p(X_t, Y_t)$ with the empirical distribution $p(X_t, Y_t) = \frac{1}{N} \sum_{n=1}^{N} \delta_{x_{nt}}(X_t) \delta_{y_{nt}}(Y_t)$, combining (4) and (1), we approximate the RIB objective function as,

$$\begin{aligned}
\frac{1}{N} \sum_{n=1}^{N} \Bigg[\sum_{t=1}^{T} &\int p(z_{nt} | x_{nt}) \log q(y_{nt} | z_{nt}) dz_{nt} \\
&- \beta \int p(z_{nt} | x_{nt}) \log \frac{p(z_{nt} | x_{nt})}{\tilde{p}_t(z_{nt})} dz_{nt} \Bigg] \qquad (5)
\end{aligned}$$

Note that the joint entropy of $Y_1 \dots Y_T$ in (1) is omitted here because the target distribution $p(Y_1 \dots Y_T)$ is fixed and its entrop always keeps constant during the optimization.

Modeling the encoding distribution $p(Z_t | X_t)$, decoding distribution $q(Y_t | Z_t)$ and latent prior $\tilde{p}_t(Z_t)$ with multivariate Gaussian, the RIB objective (5) becomes analytically tractable. In the training, we apply ADAM [12] to maximize the RIB objective.

3.2. Prediction Model

Our model is a combination of RNN and autoencoder, where the input is first encoded to stochastic features, which then are fed into GRU cell for the state transition. $\boldsymbol{z}_t$ is then decoded using the hidden state. The diagram of the proposed model is shown in Figure 1. The dashed lines show the conditional dependency in both latent prior and encoding. This model encodes $\boldsymbol{x}_t$ to $\boldsymbol{z}_t$ following the distribution $p(\boldsymbol{z}_t | \boldsymbol{x}_t)$ as,

$$Z_t \sim \mathcal{N}\big(\mu_{\text{enc}}(\boldsymbol{x}_t, \boldsymbol{h}_{t-1}), \text{diag}(\sigma^2_{\text{enc}}(\boldsymbol{x}_t, \boldsymbol{h}_{t-1}))\big) \qquad (6)$$

Here the mean μ_{enc} and variance σ^2_{enc} are constructed by multi-layer neural networks, and $\boldsymbol{h}_t$ is the hidden state of the RNN cell, which is operated by GRU transition [6].

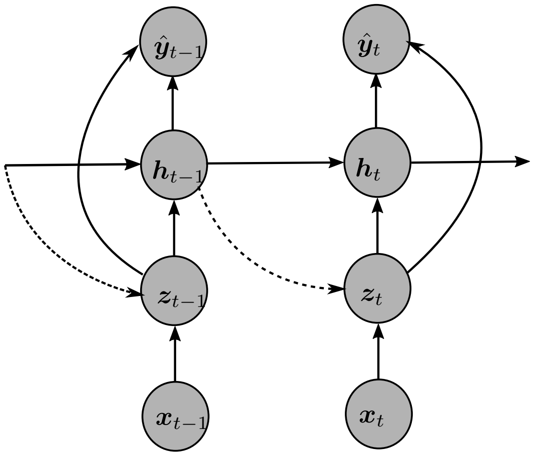

Fig. 1. Model Transition Diagram.

The stochastic decoder $q(\cdot | \boldsymbol{z}_t)$ is also realized by a Gaussian form, which can give distribution of the prediction as below,

$$\hat{Y}_t \sim \mathcal{N}\big(\mu_{\text{dec}}(\boldsymbol{z}_t, \boldsymbol{h}_{t-1}), \text{diag}(\sigma^2_{\text{dec}}(\boldsymbol{z}_t, \boldsymbol{h}_{t-1}))\big) \qquad (7)$$

where the functions μ_{dec} and σ^2_{dec} give the mean and variance, respectively.

The approximate prior distribution of the latent state $\boldsymbol{z}_t$ is multivariate Gaussian conditioned on previous RNN state $\boldsymbol{h}_{t-1}$, i.e.,

$$\tilde{p}(\boldsymbol{z}_t | \boldsymbol{h}_{t-1}) = \mathcal{N}(\mu_{\text{prior}}(\boldsymbol{h}_{t-1}), \text{diag}(\sigma^2_{\text{prior}}(\boldsymbol{h}_{t-1}))) \qquad (8)$$

where μ_{prior} and σ^2_{prior} give the mean and variance, respectively. Inside the GRU cell, the hidden state is translated as $\boldsymbol{h}_t = \text{GRU}_{\boldsymbol{\theta}}(\boldsymbol{z}_t, \boldsymbol{h}_{t-1})$ parametrized by $\boldsymbol{\theta}$. Note that function GRU is given in (3).

In implementation, to better extract meaningful information, we have feature extractors $\varphi_{\boldsymbol{x}}$ and $\varphi_{\boldsymbol{z}}$ for $\boldsymbol{x}_t$ and $\boldsymbol{z}_t$, before the operation of encoder and GRU state translation. We realize $\varphi_{\boldsymbol{x}}$ and $\varphi_{\boldsymbol{y}}$ by multi-layer neural networks.

Note that we can also use Normalizing Flow [13] or Householder Flow [14] to relax the Gaussian assumption on encoding, decoding and latent prior distribution, which can improve the prediction performance in some sense. But we don't show these results due to space limitation.

4. EXPERIMENTS

This section presents experimental study that evaluates the performance of the proposed prediction method. The performances on both single-dimension and mutli-dimension datasets are evaluated. In experiments, the prediction is the mean of the decoder output $\hat{Y}_t$, and confidence interval for prediction uncertainty is built on the predicted variance σ^2_{dec}.

4.1. Single-dimensional Prediction

This experiment uses sunspot dataset [11]. This dataset contains sunspot numbers collected in Zurich from Jan. 1749 to Dec. 1983. This is a one-dimensional nonlinear time series with 2820 time steps. The first 1000 data points are used for training, and next 1000 points are for testing. The benchmark methods are the deep belief network trained with restricted Boltzmann Machine (RBM) [3], stacked autoencoder (SAE) [4], and LSTM [5].

All the respective time series are scaled into the range $[0, 1]$. The prediction accuracy is measured by the mean absolute error (MAE) and root-mean-squared error (RMSE):

$$\text{MAE} = \frac{1}{T}\sum_{t=1}^{T}|y_t - \hat{y}_t| \tag{9}$$

$$\text{RMSE} = \sqrt{\frac{1}{T}\sum_{t=1}^{T}(y_t - \hat{y}_t)^2} \tag{10}$$

where $y_i, \hat{y}_i$ are observed and predicted data. The accuracy comparison is shown in Table 1. We also evaluated the confidence interval of the prediction. The confidence interval with confidence level $1 - \alpha$ is described as,

$$[\mu_{\text{dec}} - z_{1-\alpha/2}\sqrt{\sigma^2_{\text{dec}}}, \mu_{\text{dec}} + z_{1-\alpha/2}\sqrt{\sigma^2_{\text{dec}}}]$$

where μ_{dec} and σ^2_{dec} are mean and variance in (7) at time t, and $z_{1-\alpha/2}$ is the inverse cumulative density function of standard Gaussian. The prediction result and confidence interval are shown in Figure 2. The coefficient β in target function (5) can influence the prediction performance significantly. We set it to be 0.005.

	Proposed	RBM	SAE	LSTM
MAE	0.0453	0.0911	0.116	0.0524
RMSE	0.0612	0.132	0.150	0.0739

Table 1. Prediction Accuracy Comparison.

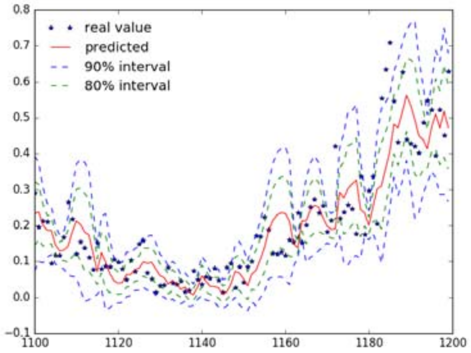

Fig. 2. Point Prediction and Confidence Intervals.

4.2. Multi-dimensional Prediction

In this experiment, we adopted the traffic dataset [15]. It is a collection of 15 months of daily data from California's Department of Transportation. The data describes the occupancy rate, between 0 and 1, of different car lanes of San Francisco bay area freeways. The data was sampled every 10 minutes. Same as [15] we aggregate the columns to obtain hourly traffic data. Hence for this dataset, the length T is 10560 and the dimension N is 963. The prediction is obtained by a rolling window of predictions same as in [15], and the model is trained on the data before first prediction window.

Define $\bar{y} = \frac{1}{NT}\sum_{n,t} y_{n,t}$. The performance metrics are normalized mean absolute error (NMAE) and normalized rooted mean squared error (NRMSE), derived from MAE (9) and RMSE (10) by dividing to $\bar{y}$, respectively. The benchmarks are the matrix factorization (MatFact) method in [15] which has the state-of-art prediction performance on this dataset, the variational recurrent autoencoder (VRNN) [8] and stacked auto-encoder (SAE) [4]. The performance comparison is shown in Table 2.

	Proposed	MatFact	VRNN	SAE
NMAE	0.1127	0.1935	0.2103	0.2234
NMRSE	0.3608	0.4263	0.4312	0.4566

Table 2. Prediction Accuracy Comparison.

5. CONCLUSION

In this work, we proposed a new method for probabilistic time series prediction. The input is encoded into a stochastic latent state and decoded by using hidden state and latent state, while training target is formulated by the recurrent information bottleneck which can find efficient latent states significantly informative about the target while keeping complexity low. The real-world experiments showed the advantage of our method over some state-of-art prediction models.

6. REFERENCES

[1] Y. LeCun, Y. Bengio, and G. Hinton, ”Deep learning,” Nature, vol. 521, no. 7553, pp. 436-444, 2015.

[2] W. Huang, G. Song, H. Hong, and K. Xie, ”Deep architecture for traffic flow prediction: deep belief networks with multi-task learning,” IEEE Transactions on Intelligent Transportation Systems, vol.15, no.5, pp.2191-2201, 2014.

[3] T. Kuremoto, S. Kimura, K. Kobayashi, and M. Obayashi, ”Time series forecasting using a deep belief network with restricted Boltzmann machines,” Neurocomputing, vol.137, pp.47-56, 2014.

[4] P. Romeu, F. Zamora-Martnez, P. Botella-Rocamora, and J. Pardo, ”Time-series forecasting of indoor temperature using pre-trained deep neural networks,” Artificial Neural Networks and Machine Learning-ICANN, pp.451-458, Springer, 2013.

[5] S. Hochreiter, and J. Schmidhuber, ”Long short-term memory,” Neural Computation, vol.9, no.8, pp.1735-1780, 1997.

[6] J. Chung, C. Gulcehre, K.-H. Cho, and Y. Bengio, ”Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling,” arXiv preprint arXiv: 1412.3555, 2014.

[7] F.A. Gers, D. Eck, and J. Schmidhuber, ”Applying LSTM to time series predictable through time-window approaches,” Neural Nets WIRN Vietri-01. Springer, London, pp. 193-200, 2002.

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

[9] N. Tishby, F.C. Pereira, and W. Biale, ”The information bottleneck method,” In The 37th annual Allerton Conf. on Communication, Control, and Computing, pp.368-377, 1999.

[10] N. Tishby and N. Zaslavsky, ”Deep learning and the information bottleneck principle,” In 2015 IEEE Information Theory Workshop, Apr. 2015, pp. 1-5.

[11] W. K. Hipel, and A. I. McLeod, ”Time series modeling of water resources and environmental systems,” Elsevier, vol. 45, vol. 45, 1994. Publicly available at https://datamarket.com/data/set/22t4

[12] D. Kingma, and J. Ba, ”Adam: A method for stochastic optimization,” arXiv preprint arXiv:1412.6980, 2014.

[13] D. Rezende, and S. Mohamed. Variational Inference with Normalizing Flows. In ICML, pp. 1530-1538, 2015.

[14] J. M. Tomczak, and M. Welling, ”Improving variational auto-encoders using householder flow,” arXiv preprint arXiv:1611.09630, 2016.

[15] H.-F. Yu, N. Rao, and I. Dhillon, ”Temporal regularized matrix factorization for high-dimensional time series prediction,” In 2016 Advances in Neural Information Processing Systems (NIPS), Dec. 2016, pp. 847-855.