Stochastic Sequential Neural Networks with Structured Inference

Hao Liu [1] **Haoli Bai** [1] **Lirong He** [2] **Zenglin Xu** [2]

Abstract

Unsupervised structure learning in high-dimensional time series data has attracted a lot of research interests. Recent advances in generative sequential modeling have suggested to combine recurrent neural networks with state space models (e.g., Hidden Markov Models). In order to explore the advantages of such combination, we propose a structured and stochastic sequential neural network, which models both the long-term dependencies via recurrent neural networks and the uncertainty in the segmentation and labels via discrete random variables. For accurate and efficient inference, we present a bi-directional inference network by reparamterizing the categorical segmentation and labels with the recent proposed Gumbel-Softmax approximation, and resort to the Stochastic Gradient Variational Bayes. We evaluate the proposed model in a number of tasks such as speech modeling and automatic segmetation and labeling, and the experimental results have demonstrated that our proposed model can achieve significant improvement over the state-of-the-art methods.

1. Introduction

Unsupervised structure learning in high-dimensional sequential data is an important research problem in a number of applications, such as machine translation, speech recognition, computational biology, and computational physiology (Sutskever et al., 2014; Dai et al., 2017).

Models for sequential data analysis such as recurrent neural networks (RNNs)(Rumelhart et al., 1988) and hidden Markov models (HMMs)(Rabiner, 1989) are widely used. Recent literature have investigated approaches of combining probabilistic generative models and recurrent neural networks for the sake of their complementary strengths in nonlinear representation learning and effective estimation of parameters (Johnson et al., 2016; Dai et al., 2017; Fraccaro et al., 2016; Jang et al., 2017). However, most of existing models are designed primarily for continuous situations and do not extend to discrete latent variables (Johnson et al., 2016; Krishnan et al., 2015; Archer et al., 2015; Krishnan et al., 2016), probably due to the difficulty of inference for discrete variables in neural networks.

To address such issues, we propose the Stochastic Sequential Neural Network (SSNN) consisting of a generative network and an inference network. The generative network is composed with a continuous sequence (i.e., hidden states in RNN) as well as two discrete sequences (i.e., segmentation variables and labels in SSM). The inference network can take the advantages of bi-directional temporal information by augmented variables, and efficiently approximate the categorical variables in segmentation and segment labels via the recently proposed Gumbel-Softmax approximation (Jang et al., 2017; Maddison et al., 2016). Thus, SSNN can model the complex and long-range dependencies in sequential data, but also maintain the structure learning ability of SSMs with efficient inference.

we compare our proposed model with the state-of-the-art neural models in a number of tasks. Experimental results in terms of both model fitting and labeling of learned segments have demonstrated the promising performance of the proposed model.

2. Model

In this section, we present our stochastic sequential neural network model. We begin with the model notations, and then we discuss the generative part of the model.

2.1. Notations

Consider a sequence of temporal sequences of vectors $\mathbf{x}_{1:T} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T]$ that depend on the deterministic variables $\mathbf{h}_{1:T} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_T]$ in RNN, hidden state variables $z_{1:T} = [z_1, z_2, ..., z_T]$ and time duration variables $d_{1:T} = [d_1, d_2, ..., d_T]$ in HSMM. Here $\mathbf{x}_t \in R^m$,

[1]SMILE Lab & Yingcai Honors College, University of Electronic Science and Technology of China [2]SMILE Lab & Big Data Research Center School of Computer Science and Engineering, University of Electronic Science and Technology of China. Correspondence to: Zenglin Xu <zenglin@gmail.com>.

ICML 2017 Time Series Workshop, Sydney, Australia, 2017.

$\mathbf{h}_t \in R^h$, $z_t \in \{1, 2, ..., K\}$ and $d_t \in \{1, 2, .., M\}$. We set $\mathbf{s}_{1:L} = [s_1, s_2, .., s_L]$ as the beginning of the segments. A difference from HMM is that for segment i, the latent state $z_{s_i:s_i+d_{s_i}-1}$ is fixed in HMM. An illustration is given in Figure 1.

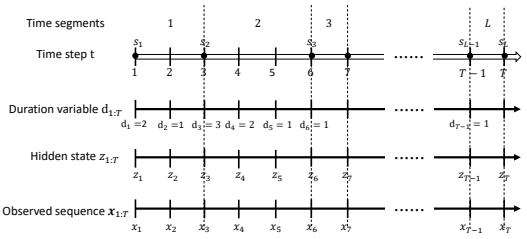


Figure 1. A visualization of the observed sequence $\mathbf{x}_{1:T}$ with the corresponding time segments, hidden states $z_{1:T}$ and duration variables $d_{1:T}$.

For the simplicity of explanation, we present our model on a single sequence. It is straightforward to apply the model multiple sequences.

2.2. Generative Model

In order to model the long-range temporal dependencies and the uncertainty in segmentation and labeling of time series, we aim to take advantages from RNN and HSMM, and learn categorical information and representation information from the observed data recurrently. As illustrated in Figure 2(a), we design an Stochastic Sequential Neural Network (SSNN) with one sequence of continuous latent variables modeling the recurrent hidden states, and two sequences of discrete variables denoting the segment duration and labels, respectively. The joint probability can be factorized as:

$$p_\theta(\mathbf{x}_{1:T}, z_{1:T}, d_{1:T}) = p_\theta(\mathbf{x}_{1:T}|z_{1:T}, d_{1:T})p_\theta(z_1)p_\theta(d_1|z_1) \prod_{t=2}^{T} p_\theta(z_t|z_{t-1}, d_{t-1})p_\theta(d_t|z_t, d_{t-1}). \quad (1)$$

To learn more interpretative latent labels, we follow the design in HSMM to set z_t and d_t as categorical random variables, The distribution of z_t and d_t is

$$p_\theta(z_t|z_{t-1}, d_{t-1}) = \begin{cases} I(z_t = z_{t-1}) & \text{if } d_{t-1} > 1 \\ p_\theta(z_t|z_{t-1}) & \text{otherwise} \end{cases},$$

$$p_\theta(d_t|z_t, d_{t-1}) = \begin{cases} I(d_t = d_{t-1} - 1) & \text{if } d_{t-1} > 1 \\ p_\theta(d_t|z_t) & \text{otherwise} \end{cases},$$

where $I(x)$ is the indicator function (whose value equals 1 if x is True, and otherwise 0). The transition probability $p_\theta(z_t|z_{t-1})$ and $p_\theta(d_t|z_t)$, in implementation, can be achieved by learning a transition matrix.

The joint emission probability $p_\theta(\mathbf{x}_{1:T}|z_{1:T}, d_{1:T})$ can be further factorized into multiple segments. Specifically, for the i-th segment $\mathbf{x}_{s_i:s_i+d_{s_i}-1}$ starting from s_i, the corresponding generative distribution is

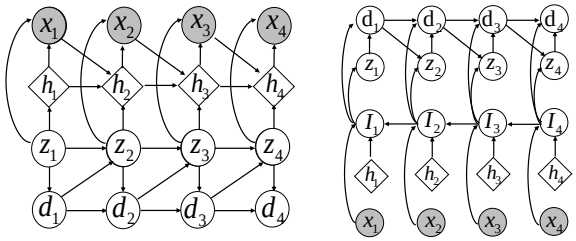
$$p_\theta(\mathbf{x}_{s_i:s_i+d_{s_i}-1}|z_{s_i}, d_{s_i}) = \prod_{t=s_i}^{s_i+d_{s_i}-1} p_\theta(\mathbf{x}_t|\mathbf{x}_{s_i:t-1}, z_{s_i}) = \prod_{t=s_i}^{s_i+d_{s_i}-1} p_\theta(\mathbf{x}_t|\mathbf{h}_t, z_{s_i}), \quad (2)$$

where $\mathbf{h}_t$ is the latent deterministic variable in RNN. It can better model the complex dependency among segments, and capture past information of the observed sequence $\mathbf{x}_{t-1}$ as well as the previous state $\mathbf{h}_{t-1}$. We design $\mathbf{h}_t = \sigma(\mathbf{W}_x^{(z_{s_i})}\mathbf{x}_{t-1} + \mathbf{W}_h^{(z_{s_i})}\mathbf{h}_{t-1} + \mathbf{b}_h^{(z_{s_i})})$, where $\sigma()$ is a tanh activation function, $\mathbf{W}_x \in R^{K\times h\times m}$ and $\mathbf{W}_h \in R^{K\times h\times h}$ are weight parameter, and $\mathbf{b}_h \in R^{K\times h}$ is the bias term. $\mathbf{W}_x^{(z_{s_i})} \in R^{h\times m}$ is the z_{s_i}-th slice of $\mathbf{W}_x$, and it is similar for $\mathbf{W}_h^{(z_{s_i})}$ and $\mathbf{b}_h^{(z_{s_i})}$.

Finally, the distribution of $\mathbf{x}_t$ given $\mathbf{h}_t$ and z_{s_i} is designed by a Normal distribution,

$$p_\theta(\mathbf{x}_t|\mathbf{h}_t, z_{s_i}) = \mathcal{N}(x; \boldsymbol{\mu}, \boldsymbol{\sigma}^2), \quad (3)$$

where the mean satisfies $\boldsymbol{\mu} = \mathbf{W}_\mu^{(z_{s_i})}\mathbf{h}_t + b_\mu^{(z_{s_i})}$, and the covariance is a diagonal matrix with its log diagonal elements $\log \boldsymbol{\sigma}^2 = \mathbf{W}_\sigma^{(z_{s_i})}\mathbf{h}_t + \mathbf{b}_\sigma^{(z_{s_i})}$. We use θ to include all the parameters in the generative model.



(a) Generative network (b) Inference network

Figure 2. The generative network and inference network of SSNN.

3. Structured Inference

The marginal log-likelihood $\log p_\theta(\mathbf{x})$ is generally intractable and we are interested in maximizing its evidence lower bound (ELBO) as follows,

$$\mathcal{L}(\mathbf{x}_{1:T}; \theta, \phi) = E_{q_\phi(z_{1:T}, d_{1:T}|\mathbf{x}_{1:T})}[\log \frac{p_\theta(_{1:T}, z_{1:T}, d_{1:T})}{q_\phi(z_{1:T}, d_{1:T}|_{1:T})}], \quad (4)$$

where $q_\phi(\cdot)$ denotes the approximate posterior distribution, and θ and ϕ denote parameters for their corresponding distributions, repsectively.

We devise a bi-directional inference scheme and resort to the Stochastic Gradient Variational Bayes (SGVB) method

since it could efficiently learn the approximation with relatively low variances (Kingma & Welling, 2013).

3.1. Bi-directional Inference

In order to find a more informative approximation to the posterior, we augment both random variables d_t, z_t with bi-directional information in the inference network. Such attempts have been explored in previous work (Krishnan et al., 2016; Khan & Lin, 2017; Krishnan et al., 2015), however they mainly focus on continuous variables. We first learn a bi-directional deterministic variable $\hat{\mathbf{h}}_t = \text{BiRNN}(x_{1:t}, x_{t:T})$, where BiRNN is a bi-directional RNN with each unit implemented as an LSTM (Hochreiter & Schmidhuber, 1997). Similar to (Fraccaro et al., 2016), we further use a backward recurrent function $I_t = g_{\phi_I}(I_{t+1}, [\mathbf{x}_t, \hat{\mathbf{h}}_t])$ to explicitly capture forward and backward information in the sequence via $\hat{\mathbf{h}}_t$, where $[\mathbf{x}_t, \hat{\mathbf{h}}_t]$ is the concatenation of $\mathbf{x}_t$ and $\hat{\mathbf{h}}_t$.

The posterior approximation can be factorized as

$$q_\phi(z_{1:T}, d_{1:T}|\mathbf{x}_{1:T}) = q_\phi(z_1|I_1)q_\phi(d_1|z_1, I_1) \prod_{t=2}^{T} q_\phi(z_t|d_{t-1}, I_t)q_\phi(d_t|d_{t-1}, z_t, I_t), \quad (5)$$

and the graphical model for the inference network is shown in Figure.2(b). We use ϕ to denotes all parameters in inference network. Furthermore, we design the posterior distributions of d_t and z_t to be categorical distributions, i.e.:

$$q(z_t|d_{t-1}, I_t; \phi) = \text{Cat}(\text{softmax}(\mathbf{W}_z^T I_t)), \quad (6)$$

$$q(d_t|d_{t-1}, z_t, I_t; \phi) = \text{Cat}(\text{softmax}(\mathbf{W}_d^T I_t)). \quad (7)$$

The parameters of the distributions depend on both the forward sequences (i.e., $h_{t:T}$ and $x_{t:T}$) and the backward sequences (i.e., $h_{1:t-1}$ and $x_{1:t-1}$), leading to a more informative approximation. Since the reparamterization of discrete variables is a challenging task, we turn to the recently proposed Gumbel-Softmax reparameterization trick (Jang et al., 2017; Maddison et al., 2016), as shown in the following section.

3.2. Gumbel-Softmax Reparameterization

The Gumbel-Softmax reparameterization proposes an alternative way to approximately reparameterize the discrete random variable and allow the back propagation of the parameter gradients. To use the Gumbel-Softmax trick, we first map the discrete pair (z_t, d_t) to a N-dimensional vector $\boldsymbol{\gamma}(t)$, and $\boldsymbol{\gamma}(t) \sim \text{Cat}(\boldsymbol{\pi}(t))$, where $\boldsymbol{\pi}(t)$ is a N-dimensional vector on the simplex and $N = K \times D$. Then we use $\mathbf{y}(t) \in R^N$ to represent the Gumbel-Softmax distributed variable:

$$y_i(t) = \frac{\exp((\log(\pi_i(t)) + g_i)/\tau)}{\sum_{j=1}^{k} \exp((\log(\pi_j(t)) + g_j)/\tau)} \quad (8)$$

where $g_i \sim \text{Gumbel}(0, 1)$, $i = 1, 2, ...N$, and τ is the temperature that will be elaborated in the experiment. We set $\mathbf{y}(t) \sim \text{Concrete}(\boldsymbol{\pi}(t), \tau)$ according to (Maddison et al., 2016).

Now we can sample $\mathbf{y}(t)$ from the Gumbel-Softmax posterior in replacement of the categorically distributed $\boldsymbol{\gamma}(t)$. We denote $F(z, d) = \log p_\theta(z_{1:T}, z_{1:T}, d_{1:T}) - \log q(z_{1:T}, d_{1:T}|_{1:T})$, and furthermore, $\tilde{F}(y, g)$ is the corresponding approximation term of $F(z, d)$ after the Gumbel-Softmax trick. Finally, we summarize the inference algorithm in Algorithm 1.

Algorithm 1 Strucutured Inference Algorithm for SSNN

inputs: Observed sequences $\{\mathbf{x}^{(n)}\}_{n=1}^N$
Randomly initialized $\phi^{(0)}$ and $\theta^{(0)}$;
Inference Model: $q_\phi(z_{1:T}, d_{1:T}|\mathbf{x}_{1:T})$;
Generative Model: $p_\theta(\mathbf{x}_{1:T}, z_{1:T}, d_{1:T})$;
outputs:Model parameters θ and ϕ;
for $i = 1$ to $Iter$ **do**
1. Sample sequences $\{x^{(n)}\}_{n=1}^M$ uniformly from dataset with a mini-batch size B.
2. Estimate and sample forward parameters using Eq.(1).
3. Evaluate the *ELBO* using Eq. (4).
4. Estimate the Monte Carlo approximation to $\nabla_\theta L$.
5. Estimate the SGVB approximation to $\nabla_\phi L$ with the Gumbel-Softmax approximation.
6. Update $\theta^{(i)}$, $\phi^{(i)}$ using the ADAM.
end for

4. Experiment

In this section, we evaluate SSNN on several datasets across multiple scenarios. We first evaluate the performance of finding complex structures on two speech datasets (TIMIT & Blizard), and then test SSNN with learning segmentations and latent labels on Human activity (Reyes-Ortiz et al., 2016) dataset, Drosophila dataset (Kain et al., 2013) and PhysioNet (Springer et al., 2016) Challenge dataset. Finally, we provide another test on the multi-object recognition problem using multi-MNISTdataset.

Due to the limited space, more experiments on synthetic datasets and more details about parameter setting and datasets description can be found in the Appendix 6.

4.1. Speech Modeling

We also test SSNN on the modeling of speech data, i.e., Blizzard and TIMIT datasets. Blizzard records the English speech with 300 hours by a female speaker. TIMIT is a dataset with 6300 English sentences read by 630 speakers.

we report the average log-likelihood for half-second sequences on Blizzard, and report the average log-likelihood

MODELS	Blizzard	TIMIT
VRNN-GMM	$\geq$ 9107	$\geq$ 28982
VRNN-GAUSS	$\geq$ 9223	$\geq$ 28805
VRNN-I-GAUSS	$\geq$ 9223	$\geq$28805
SRNN(smooth+Res_q)	$\geq$ 11991	$\geq$ 60550
SRNN(smooth)	$\geq$10991	$\geq$59269
SRNN(filt)	$\geq$10846	50524
RNN-GMM	7413	26643
RNN-GAUSS	3539	-1900
Our Method(SSNN)	$\geq$ **13123**	$\geq$ **64017**

Table 1. Average log-likelihood per sequence on the test sets. The higher the better.

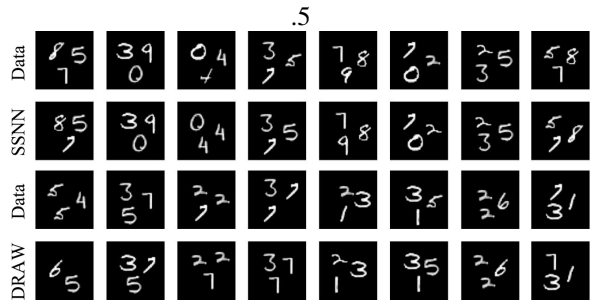


Figure 3. Visualization and comparison of SSNN and DRAW on multi-object recognition problems.

per sequence for the test set sequences on TIMIT. For the raw audio datasets, we use a fully factorized Gaussian output distribution. We compare our method with a number of methods, which is introduced in Appendix 6.5

From Table 4.2 it can be observed that on both datasets SSNN outperforms the state of the art methods by a large margin, indicating its superior ability in speech modeling.

4.2. Segmentation and Labeling of Time Series

To show the advantages of SSNN when learning the segmentation and latent labels from sequences, we take experiments on Human activity dataset (Human) (Reyes-Ortiz et al., 2016), Drosophila dataset (Dros) (Kain et al., 2013) and PhysioNet (Springer et al., 2016) Challenge dataset (Physio). Human activity dataset consists of time series signals from sensors mounted on the volunteers. Drosophila dataset records the time series movement of fruit flies' legs. Both Human Activity and Drosophila dataset are used for segmentation prediction. PhysioNet Challenge dataset (Springer et al., 2016) records observation labeled with one of the four hidden states, namely Diastole, S1, Systole and S2, and it is used for latent label prediction.

Specifically, we compare the predicted segments or latent labels with the ground truth, and report the mean and the standard deviation of the error rate for all methods. Details of hyper-parameters and setting are shown in Appendix 6.2.

MODELS	DROS	HUMAN	PHYSIO
HSMM	47.3 $\pm$ 0.3%	41.6 $\pm$ 8.6 %	45.0 $\pm$ 1.9 %
SUBHSMM	39.7 $\pm$ 2.2%	22.2 $\pm$ 4.5%	43.0 $\pm$ 2.4 %
HDP-HSMM	43.6 $\pm$ 1.6%	35.5 $\pm$ 6.2%	42.6 $\pm$ 1.5 %
CRF-AE	57.6 $\pm$ 0.2%	49.3 $\pm$ 10.6%	45.7 $\pm$ 0.7 %
RHSMM-DP	36.2 $\pm$ 1.4%	16.4 $\pm$ 5.1%	31.2 $\pm$ 4.1 %
SSNN	**34.8** $\pm$ 3.7%	**14.7** $\pm$ 5.5%	**29.3** $\pm$ 5.3 %

Table 2. Mean and standard deviation of the error rate.

We report the comparison with subHSMM (Johnson & Willsky, 2014), HDP-HSMM (Johnson & Willsky, 2013), CRF-AE (Ammar et al., 2014) and rHSMM-dp (Dai et al., 2017). Experimental results are shown in Table 2. It can be observed that SSNN achieves the lowest mean error rate, indicating the effectiveness of combining RNN with HSMM to collectively learn the segmentation and the latent states.

4.3. Sequential Multi-objects Recognition

To further verify the ability of modeling complex spatial dependency, we test SSNN on the multiple objects recognition problem. we construct a small image dataset including 3000 images, named as multi-MNIST. Each image consists of three non-overlapping random MNIST digits with an equal probability.

Our goal is to sequentially recognize each digit in the image. In our experiment, we train our model with 2500 images and test on the rest 500 images. We compare the proposed model to DRAW (Gregor et al., 2015) and visualize our learned latent representations in Figure 3. It can be observed that our model identifies the number and locations of digits correctly, while DRAW sometimes misses modes of data. The result shows that our method can accurately capture not only the number of objects but also locations.

5. Conclusion

In order to learn the structures (e.g., the segmentation and labeling) of high-dimensional time series in a unsupervised way, we have proposed a Stochastic sequential neural network(SSNN) with structured inference. For better model interpretation, we further restrict the label and segmentation duration to be two sequences of discrete variables, respectively. In order to exploit forward and backward temporal information, we carefully design structured inference, and to overcome the difficulties of inferring discrete latent variables in deep neural networks, we resort to the recently proposed Gumbel-Softmax functions. The advantages of the proposed inference method have been demonstrated in both synthetic and real-world sequential benchmarks.

References

Ammar, Waleed, Dyer, Chris, and Smith, Noah A. Conditional random field autoencoders for unsupervised structured prediction. In *NIPS*, pp. 3311–3319, 2014.

Archer, Evan, Park, Il Memming, Buesing, Lars, Cunningham, John, and Paninski, Liam. Black box variational inference for state space models. *arXiv preprint arXiv:1511.07367*, 2015.

Chung, Junyoung, Kastner, Kyle, Dinh, Laurent, Goel, Kratarth, Courville, Aaron C, and Bengio, Yoshua. A recurrent latent variable model for sequential data. In *NIPS*, pp. 2980–2988, 2015.

Dai, Hanjun, Dai, Bo, Zhang, Yan-Ming, Li, Shuang, and Song, Le. Recurrent hidden semi-markov model. *ICLR*, 2017.

Fabius, Otto and van Amersfoort, Joost R. Variational recurrent auto-encoders. *arXiv preprint arXiv:1412.6581*, 2014.

Fraccaro, Marco, Sønderby, Søren Kaae, Paquet, Ulrich, and Winther, Ole. Sequential neural models with stochastic layers. In *NIPS*, pp. 2199–2207, 2016.

Gan, Zhe, Li, Chunyuan, Henao, Ricardo, Carlson, David E, and Carin, Lawrence. Deep temporal sigmoid belief networks for sequence modeling. In *NIPS*, pp. 2467–2475, 2015.

Gregor, Karol, Danihelka, Ivo, Graves, Alex, Rezende, Danilo Jimenez, and Wierstra, Daan. Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*, 2015.

Gu, Shixiang, Ghahramani, Zoubin, and Turner, Richard E. Neural adaptive sequential monte carlo. In *NIPS*, pp. 2629–2637, 2015.

Hochreiter, Sepp and Schmidhuber, Jürgen. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Jang, Eric, Gu, Shixiang, and Poole, Ben. Categorical reparameterization with gumbel-softmax. *stat*, 1050:1, 2017.

Johnson, Matthew and Willsky, Alan. Stochastic variational inference for bayesian time series models. In *International Conference on Machine Learning*, pp. 1854–1862, 2014.

Johnson, Matthew, Duvenaud, David K, Wiltschko, Alex, Adams, Ryan P, and Datta, Sandeep R. Composing graphical models with neural networks for structured representations and fast inference. In *NIPS*, pp. 2946–2954, 2016.

Johnson, Matthew J and Willsky, Alan S. Bayesian nonparametric hidden semi-markov models. *Journal of Machine Learning Research*, 14(Feb):673–701, 2013.

Kain, Jamey, Stokes, Chris, Gaudry, Quentin, Song, Xiangzhi, Foley, James, Wilson, Rachel, and De Bivort, Benjamin. Leg-tracking and automated behavioural classification in drosophila. *Nature communications*, 4:1910, 2013.

Karl, Maximilian, Soelch, Maximilian, Bayer, Justin, and van der Smagt, Patrick. Deep variational bayes filters: Unsupervised learning of state space models from raw data. *arXiv preprint arXiv:1605.06432*, 2016.

Khan, Mohammad Emtiyaz and Lin, Wu. Conjugate-computation variational inference: Converting variational inference in non-conjugate models to inferences in conjugate models. *arXiv preprint arXiv:1703.04265*, 2017.

Kingma, Diederik P and Welling, Max. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

Krishnan, Rahul G, Shalit, Uri, and Sontag, David. Deep kalman filters. *arXiv preprint arXiv:1511.05121*, 2015.

Krishnan, Rahul G, Shalit, Uri, and Sontag, David. Structured inference networks for nonlinear state space models. *arXiv preprint arXiv:1609.09869*, 2016.

Maddison, Chris J, Mnih, Andriy, and Teh, Yee Whye. The concrete distribution: A continuous relaxation of discrete random variables. *arXiv preprint arXiv:1611.00712*, 2016.

Prahallad, Kishore, Vadapalli, Anandaswarup, Elluru, Naresh, Mantena, G, Pulugundla, B, Bhaskararao, P, Murthy, HA, King, S, Karaiskos, V, and Black, AW. The blizzard challenge 2013–indian language task. In *Blizzard Challenge Workshop*, volume 2013, 2013.

Rabiner, Lawrence R. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

Reyes-Ortiz, Jorge-L, Oneto, Luca, Sama, Albert, Parra, Xavier, and Anguita, Davide. Transition-aware human activity recognition using smartphones. *Neurocomputing*, 171:754–767, 2016.

Rumelhart, David E, Hinton, Geoffrey E, and Williams, Ronald J. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.

Springer, David B, Tarassenko, Lionel, and Clifford, Gari D. Logistic regression-hsmm-based heart sound segmentation. *IEEE Transactions on Biomedical Engineering*, 63(4):822–832, 2016.

Sutskever, Ilya, Vinyals, Oriol, and Le, Quoc V. Sequence to sequence learning with neural networks. In *NIPS*, pp. 3104–3112, 2014.

6. Appendix

6.1. Speech Modeling

Speech modeling on these two datasets has shown to be challenging since there's no good representation of the latent states (Chung et al., 2015; Fabius & van Amersfoort, 2014; Gu et al., 2015; Gan et al., 2015; Sutskever et al., 2014). The data preprocessing and the performance measures are identical to those reported in (Chung et al., 2015; Fraccaro et al., 2016).

We compare our method with the following methods, For RNN+VRNNs (Chung et al., 2015), VRNN is tested with two different output distributions: a Gaussian distribution (VRNN-GAUSS), and a Gaussian Mixture Model (VRNN-GMM). We also compare to VRNN-I in which the latent variables in VRNN are constrained to be independent across time steps. For SRNN (Fraccaro et al., 2016), we compare with the smoothing and filtering performance denoted as SRRR (smooth), SRNN (filt) and SRNN (smooth+ Res_q) respectively. The results of VRNN-GMM, VRNN-Gauss and VRNN-I-Gauss are taken from (Chung et al., 2015), and those of SRNN (smooth+Res_q), SRNN (smooth) and SRNN (filt) are taken from (Fraccaro et al., 2016).

6.2. Drosophila and Human activity

We first introduce the dataset in more detail. For Human activity, it is collected by (Reyes-Ortiz et al., 2016) that consists of signals collected from waist-mounted smartphones with accelerometers and gyroscopes. Each volunteer is asked to perform 12 activities. There are 61 recorded sequences, and the maximum time steps $T \approx 3,000$. Each $\mathbf{x}_t$ is a 6 dimensional vector. For Drosophila (Kain et al., 2013), at each time step t, $\mathbf{x}_t$ is a 45-dimension vector, which consists of the raw and some higher order features. the maximum time steps $T \approx 10,000$. In the experiment, we fix the τ at small value 0.0001.

For the HDP-HSMM(Johnson & Willsky, 2014) and subHSMM(Johnson & Willsky, 2013), the observed sequences $\mathbf{x}_{1:T}$ are generated by standard multivariate Gaussian distributions. The duration variable d_t is from the Poisson distribution. We need to tune the concentration parameters α and γ. As for the hyper parameters, they can be learned automatically. For subHSMM, we tune the truncation threshold of the infinite HMM in the second level. For CRF-AE, we extend the original model to learn continuous data. We use mixture of Gaussian for the emission probability. For R-HSMM-dp, it is a version of R-HSMM(Dai et al., 2017) with the exact MAP estimation via dynamic programming.

6.3. Physionet

PhysioNet Challenge dataset (Springer et al., 2016) records observation labeled with one of the four hidden states, i.e., Diastole, S1, Systole and S2. The experiment aims to exam SSNN on learning and predicting the labels. In the experiment, we find that annealing of temperature τ is important, we start from $\tau = 0.15$ and anneal it gradually to 0.0001.

6.4. TIMIT and Blizzard

For the TIMIT and Blizzard dataset (Prahallad et al., 2013), the sampling frequency is 16KHz and the raw audio signal is normalized using the global mean and standard deviation of the training set. We split the raw audio signals in the chunks of 2 seconds. The waveforms are divided into non-overlapping vectors with size 200. For Blizzard we split the data using 90% for training, 5% for validation and 5% for testing. For testing we report the average log-likelihood for each sequence with segment length 0.5s. For TIMIT we use the predefined test set for testing and split the rest of the data into 95% for training and 5% for validation.

During training we use backpropagation through time (BPTT) for 1 second. For the first second we initialize hidden units with zeros and for the subsequent 3 chunks we use the previous hidden states as initialization. In the experiment, the temperature τ starts from a large value 0.1 and gradually anneals to 0.01.

6.5. Multi-MNIST

We first describe the generation of this dataset. We begin with a 50×50 dataset of multi-MNIST digits. Each image contains three non-overlapping random MNIST digits with equal probability. The desired goal is to train a network that produces sensible explanations for each of the images. First we fix the maximum time steps $T = 3$ and feed the same image as input sequentially to SSNN. We interpret the latent variable d_t as intensity and z_t as the location variable in the training images. Then We train SSNN with random initialized parameters on 60,000 multi-MNIST images from scratch, i.e., without a curriculum or any form of supervision. All experiments were performed with a batch size of 64. The learning rate of model is 1×10^{-5} and baselines were trained using a higher learning rate 1×10^{-3}. The LSTMs in the inference network had 256 cell units.

6.6. Synthetic Experiment

To validate that our method is able to model high dimensional data with complex dependency, we simulated a complex dynamic torque-controlled pendulum governed by a differential equation to generate non-Markovian observations from a dynamical system: $ml^2 \frac{d^2\phi(t)}{dt^2} = -\mu \frac{d\phi(t)}{dt} + mgl \sin \phi(t) + u(t)$. For fair comparison with (Karl et al.,

2016), For fair comparison with (Karl et al., 2016), we set $m = l = 1$, $\mu = 0.5$, and $g = 9.81$. We convert the generated ground-truth angles to image observations. The system can be fully described by angle and angular velocity.

We compare our method with Deep Variational Bayes Filter(DVBF-LL) (Karl et al., 2016) and Deep Kalman Filters(DKF) (Krishnan et al., 2015). The ordinary least square regression results are shown in Table 3. Our method is clearly better than DVBF-LL and DKF in predicting $\sin\phi$, $\cos\phi$ and $\frac{d\phi}{dt}$. SSNN achieves a higher goodness-of-fit than other methods.

	DVBF-LL		DKF		SSNN	
	log ll	R^2	log ll	R^2	log ll	R^2
$\sin\phi$	3990.8	0.961	1737.6	0.929	4424.6	0.975
$\cos\phi$	7231.1	0.982	6614.2	0.979	8125.3	0.997
$\frac{d\phi}{dt}$	-11139	0.916	-20289	0.035	-9620	0.941

Table 3. The results measured on the log-likelihood(denoted as log ll) and the goodness-of-fit (denoted by R^2) given by three methods on the prediction of all latent states on respective dependent variables in pendulum dynamics. For both measures, the higher the better.