Structured Inference for Recurrent Hidden Semi-Markov Model

Hao Liu[1,2], **Lirong He**[1], **Haoli Bai**[3], **Bo Dai**[4], **Kun Bai**[5] and **Zenglin Xu**[1]
[1] SMILE Lab, University of Electronic Science and Technology of China, Chengdu, China
[2] Yingcai Honors College, University of Electronic Science and Technology of China, Chengdu, China
[3] Dept. of Computer Science and Engineering, The Chinese University of Hong Kong, Hong Kong
[4] College of Computing, Georgia Institute of Technology, Atlanta, GA USA
[5] Mobile Internet Group, Tencent Inc., Shenzhen, China
liuhaosater@gmail.com, lirong_he@std.uestc.edu.cn, hlbai@cse.cuhk.edu.hk,
bohr.dai@gmail.com, kunbai@tencent.com, zenglin@gmail.com

Abstract

Segmentation and labeling for high dimensional time series is an important yet challenging task in a number of applications, such as behavior understanding and medical diagnosis. Recent advances to model the nonlinear dynamics in such time series data has suggested involving recurrent neural networks into Hidden Markov Models. Despite the success, however, this involvement has caused the inference procedure much more complicated, often leading to intractable inference, especially for the discrete variables of segmentation and labeling. To achieve both flexibility and tractability in modeling nonlinear dynamics of discrete variables and to model both the long-term dependencies and the uncertainty of the segmentation labels, we inherits the Recurrent Hidden Semi-Markov Model and presents an effective bi-directional inference method. In detail, the proposed bi-directional inference network reparameterizes the categorical segmentation with the Gumbel-Softmax approximation and resorts to the Stochastic Gradient Variational Bayes. We evaluate the proposed model in a number of tasks, including speech modeling, automatic segmentation and labeling in behavior understanding, and sequential multi-objects recognition. Experimental results have demonstrated that our proposed model can achieve significant improvement over the state-of-the-art methods.

1 Introduction

Unsupervised structure learning for high-dimensional sequential data has abstracted a lot of attention in a number of applications, such as machine translation, speech recognition, computational biology, and computational physiology [Sutskever *et al.*, 2014; Dai *et al.*, 2017b]. In this paper, we focus on a type of unsupervised structure learning with discrete variables. Discrete variables can be more interpretable and helpful in many applications like medical analysis and behavior prediction. For example, in medical diagnosis, learning the segment boundaries and labeling of complicated physical signals is very useful for doctors to understand the underlying behavior or activity types [Jang *et al.*, 2017]. An illustration of segmentation learning and labeling for sequential data can be found in Figure 1.

Standard models for sequential data analysis include recurrent neural networks which do well in capturing the long-range temporal dependencies (RNNs) [Rumelhart *et al.*, 1988; Sutskever *et al.*, 2014], and hidden Markov models (HMMs) [Chiappa and others, 2014] that are good at modeling uncertainty. Recent literature has suggested combining recurrent neural networks with probabilistic generative models for the sake of their respect and complementary strengths in the nonlinear representation learning and effective estimation of parameters [Johnson *et al.*, 2016; Fraccaro *et al.*, 2016]. However, despite the success of these methods, most of them are designed primarily for continuous situations [Johnson *et al.*, 2016; Krishnan *et al.*, 2015; 2016], probably due to the difficulty of inference for discrete variables in neural networks.

To address such issues, we present a neural sequential model with end-to-end training which is composed with a generative network and an inference network. In detail, to provide the flexibility in modeling discrete variables, the generative network adopted (as shown in Figure 2(a)) is the same as the Recurrent HSMM [Dai *et al.*, 2017b], which includes a continuous sequence (i.e., hidden states in RNN) as well as two discrete sequences (i.e., segmentation variables and labels in SSM). For keeping model flexibility and inference efficiency, we propose an efficient structured inference algorithm (as shown in Figure 2(b)), taking advantage of the bi-directional temporal information by introducing augmented variables. In particular, to keep the interpretable ability of discrete variables, and also make the inference tractable, we approximate the categorical variables in segmentation and segment labels via the recently proposed Gumbel-Softmax approximation [Jang *et al.*, 2017; Maddison *et al.*, 2016]. In contrast, in [Dai *et al.*, 2017b], although maximizing the negative Helmholtz free energy can make inference easier, it oversimplified the original data evi-

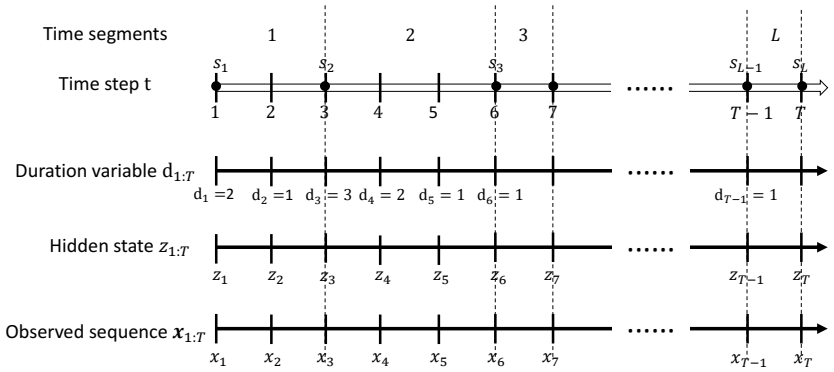


Figure 1: A visualization of the observed sequence $\mathbf{x}_{1:T}$ with the corresponding time segments, hidden states $z_{1:T}$ and duration variables $d_{1:T}$. In HMM, there is no duration variable d_t and segments are pre-determined.

dence and could impair the model flexibility, as also verified in the experiments (See Table 3 for more details). For the convenience of presentation, we denote the combined network as Stochastic Sequential Neural Network (SSNN).

In order to evaluate the performance of the proposed model, we compare our proposed model with the state-of-the-art neural models in a number of tasks, including automatic segmentation and labeling on datasets of speech modeling, behavior modeling, and medical diagnosis. Experimental results in terms of both model fitting and labeling of learned segments have demonstrated the promising performance of the proposed model.

2 Preliminaries

Before presenting our model, we introduce related backgrounds on recurrent neural networks and state space models.

Recurrent neural networks, as a wide range of sequential models for time series, have been applied in a number of applications [Sutskever *et al.*, 2014]. Consider a sequence of temporal sequences of vectors $\mathbf{x}_{1:T} = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_T]$ that depends on the inputs $\mathbf{u}_{1:T} = [\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_T]$, where $\mathbf{x}_t \in \mathbb{R}^m$ is the observation and $\mathbf{u}_t \in \mathbb{R}^n$ is the input at the time step t, and T is the maximum time step. RNN introduces hidden states $\mathbf{h}_{1:T} = [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_T]$, where $\mathbf{h}_t \in \mathbb{R}^h$ encodes the information before the time step t, and is determined by $\mathbf{h}_t = f_\theta(\mathbf{h}_{t-1}, \mathbf{u}_t)$, where $f_\theta(\cdot)$ is a nonlinear function parameterized by a neural network.

State space models such as hidden Markov models (HMMs) and hidden Semi-Markov models (HSMMs) are also widely-used methods in sequential learning [Murphy, 2002; Chiappa and others, 2014]. In HMM, given an observed sequence $\mathbf{x}_{1:T}$, each $\mathbf{x}_t$ is generated based on the hidden state $z_t \in \{1, 2, ..., K\}$, and $p_\theta(\mathbf{x}_t|z_t)$ is the emission probability. We set $p_\theta(z_1)$ as the distribution of the initial state, and $p_\theta(z_t|z_{t-1})$ is the transition probability. We use $z_{1:T} = [z_1, z_2, ..., z_T]$. Here θ includes all parameters necessary for these distributions. HSMM is a famous extension of HMM. Aside from the hidden state z_t, HSMM further introduces time duration variables $d_t \in \{1, 2, .., M\}$, where M is the maximum duration value for each $\mathbf{x}_t$. We set $d_{1:T} = [d_1, d_2, ..., d_T]$. HSMM splits the sequence into L segments, allowing the flexibility to find the best segment representation. 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 HSMM. An illustration is given in Figure 1.

There are many variants of HSMMs such as the Hierarchical Dirichlet-Process HSMM (HDP-HSMM) [Johnson and Willsky, 2013] and subHSMM [Johnson and Willsky, 2014]. The subHSMM and HDP-HSMM extend their HMM counterparts by allowing explicit modeling of state duration lengths with arbitrary distributions. While there are various types of HMM, the inference methods are mostly inefficient.

Although HMMs and HSMMs can explicitly model uncertainty in the latent space and learn an interpretable representation through d_t and z_t, they are not good at capturing the long-range temporal dependencies when compared with RNNs.

3 Stochastic Sequential Neural Network

We present the stochastic sequential neural network model, following the notations and settings of the preliminaries in the last section. For the simplicity of explanation, we present our model on a single sequence.

3.1 Generative Model

The generative model of Recurrent HSMM [Dai *et al.*, 2017b] combines the advantages of RNN and HSMM. Recurrent HSMM can model both the long-range temporal dependencies and and the uncertainty in segmentation and labeling of time series. As illustrated in Figure 2(a), we design a generative network 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:

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

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

$$p_\theta(z_t|z_{t-1}, d_{t-1}) = \begin{cases} \mathbb{I}(z_t = z_{t-1}) & \text{if } d_{t-1} > 1 \\ p_\theta(z_t|z_{t-1}) & \text{otherwise} \end{cases}, \quad (2)$$

$$p_\theta(d_t|z_t, d_{t-1}) = \begin{cases} \mathbb{I}(d_t = d_{t-1} - 1) & \text{if } d_{t-1} > 1 \\ p_\theta(d_t|z_t) & \text{otherwise} \end{cases}, \quad (3)$$

where $\mathbb{I}(x)$ 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

$$\begin{aligned} p_\theta(\mathbf{x}_{s_i:s_i+d_{s_i}-1}|z_{s_i}, d_{s_i}) &= \prod_{t=s_i}^{s_i+d_{s_i}-1} p_\theta(\mathbf{x}_t|\mathbf{x}_{s_i:t-1}, z_{s_i}) \\ &= \prod_{t=s_i}^{s_i+d_{s_i}-1} p_\theta(\mathbf{x}_t|\mathbf{h}_t, z_{s_i}), \end{aligned} \quad (4)$$

where $\mathbf{h}_t$ is the latent deterministic variable in RNN. As mentioned earlier, $\mathbf{h}_t$ 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 $\sigma(\cdot)$ is the tanh activation function, $\mathbf{W}_x \in \mathbb{R}^{K\times h\times m}$ and $\mathbf{W}_h \in \mathbb{R}^{K\times h\times h}$ are weight parameters, and $\mathbf{b}_h \in \mathbb{R}^{K\times h}$ is the bias term. $\mathbf{W}_x^{(z_{s_i})} \in \mathbb{R}^{h\times m}$ 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,

$$p_\theta(\mathbf{x}_t|\mathbf{h}_t, z_{s_i}) = \mathcal{N}(x; \boldsymbol{\mu}, \boldsymbol{\sigma}^2), \quad (5)$$

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.

3.2 Structured Inference

We are interested in maximizing the marginal log-likelihood $\log p(\mathbf{x})$, however, this is usually intractable since the complicated posterior distributions cannot be integrated out generally. Recent methods in Bayesian learning, such as the score function (or REINFORCE) [Archer *et al.*, 2015] and the Stochastic Gradient Variational Bayes (SGVB) [Kingma and Welling, 2013], are common black-box methods that lead to tractable solutions. We resort to the SGVB since it could efficiently learn the approximation with relatively low variances [Kingma and Welling, 2013], while the score function and REINFORCE suffer from high variances and heavy computational costs. We now focus on maximizing the evidence lower bound also known as $ELBO$,

$$\begin{aligned} \log p_\theta(\mathbf{x}) &\geq \mathcal{L}(\mathbf{x}_{1:T}; \theta, \phi) \\ &= E_{q_\phi(z_{1:T}, d_{1:T}|\mathbf{x}_{1:T})}[\log p_\theta(\mathbf{x}_{1:T}, z_{1:T}, d_{1:T}) \\ &\quad - \log q_\phi(z_{1:T}, d_{1:T}|\mathbf{x}_{1:T})], \end{aligned} \quad (6)$$

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

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 many previous work [Krishnan *et al.*, 2015; 2016], however they mainly focus on continuous variables, and little attention is paid to the discrete variable. Specifically, 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 and 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

$$\begin{aligned} q_\phi(z_{1:T}, d_{1:T}|\mathbf{x}_{1:T}) =& q_\phi(z_1|I_1)q_\phi(d_1|z_1, I_1) \\ &\cdot \prod_{t=2}^{T} q_\phi(z_t|d_{t-1}, I_t)q_\phi(d_t|d_{t-1}, z_t, I_t), \end{aligned} \quad (7)$$

and the graphical model for the inference network is shown in Figure.2(b). We use ϕ to denote all parameters in inference network.

We design the posterior distributions of d_t and z_t to be categorical distributions, respectively, as follows:

$$q(z_t|d_{t-1}, I_t; \phi) = \text{Cat}(\text{softmax}(\mathbf{W}_z^T I_t)), \quad (8)$$

$$q(d_t|d_{t-1}, z_t, I_t; \phi) = \text{Cat}(\text{softmax}(\mathbf{W}_d^T I_t)), \quad (9)$$

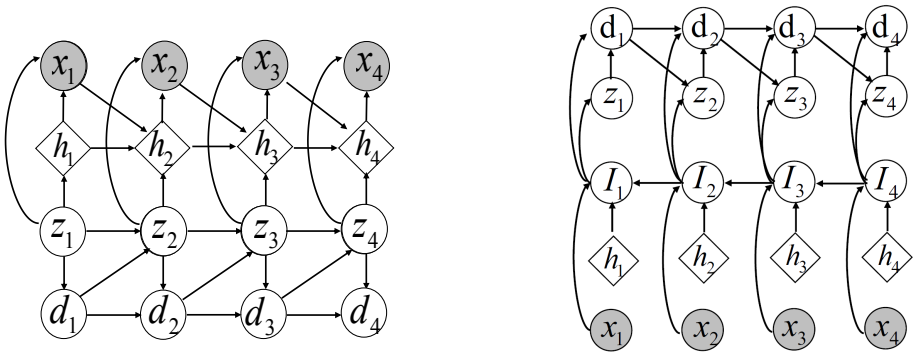
where Cat denotes the categorical distribution. Since the posterior distributions of z_t and d_t are conditioned on I_t, they depend on both the forward sequences (i.e., $h_{t:T}$ and $x_{t:T}$) and the backward sequences (i.e., $h_{1:t-1}$ and $x_{1:t-1}$), leading to a more informative approximation. However, the reparameterization tricks and their extensions [Chung *et al.*, 2016] are not directly applicable due to the discrete random variables, i.e., d_t and z_t in our model. Thus we turn to the recently proposed Gumbel-Softmax reparameterization trick [Jang *et al.*, 2017; Maddison *et al.*, 2016], as shown in the following.

Gumbel-Softmax Reparameterization

The Gumbel-Softmax reparameterization proposes an alternative to the back propagation through discrete random variables via the Gumbel-Softmax distribution, and circumvents the non-differentiable categorical distribution.

To use the Gumbel-Softmax reparameterization, 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 $N = K \times M$. Then we use $\mathbf{y}(t) \in \mathbb{R}^N$ to represent the Gumbel-Softmax distributed variable:

$$y_i(t) = \frac{\exp((\log(\pi_i(t)) + g_i)/\tau)}{\sum_{j=1}^{k}\exp((\log(\pi_j(t)) + g_j)/\tau)} \quad \text{for } i = 1, ..., N, \quad (10)$$



(a) Generative network (b) Inference network

Figure 2: The generative network and inference network of SSNN. The generative network can be viewed as a HSMM with recurrent structure on hidden states. The inference network is designed by backwards recurrent function through $\mathcal{I}_t$ for each segment. Diamond units are deterministic variables, while circles are random variables.

where $g_i \sim \text{Gumbel}(0, 1)$, and τ is the temperature that will be elaborated in the experiment. Via the Gumbel Softmax transformation, 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 $\boldsymbol{\gamma}(t)$, and use the back-propagation gradient with the ADAM [Kingma and Ba, 2014] optimizer to learn parameters θ and ϕ.

For simplicity, we denote $F(z, d) = \log p_\theta(\mathbf{x}_{1:T}, z_{1:T}, d_{1:T}) - \log q(z_{1:T}, d_{1:T}|\mathbf{x}_{1:T})$, and furthermore, $\tilde{F}(y, g)$ is the corresponding approximation term of $F(z, d)$ after the Gumbel-Softmax trick. The Gumbel-Softmax approximation of $\mathcal{L}(\mathbf{x}_{1:T}; \theta, \phi)$ is:

$$\begin{aligned}\mathcal{L}(\mathbf{x}_{1:T}; \theta, \phi) &\approx E_{y\sim\text{Concrete}(\boldsymbol{\pi}(t),\tau)}[\tilde{F}(y, g)] \\ &= E_{g\sim\prod_N \text{Gumbel}(0,1)}[\tilde{F}(y, g)]. \end{aligned} \tag{11}$$

Hence the derivatives of the approximated $ELBO$ w.r.t. the inference parameters ϕ can be approximated by the SGVB estimator:

$$\begin{aligned}&\frac{\partial}{\partial\phi} E_{g\sim\prod_N \text{Gumbel}(0,1)}[\tilde{F}(y, g)] \\ &= E_{g\sim\prod_N \text{Gumbel}(0,1)}\left[\frac{\partial}{\partial\phi}\left(\tilde{F}(y, g)\right)\right] \\ &\approx \frac{1}{B}\sum_{b=1}^{B}\frac{\partial}{\partial\phi}\left(\tilde{F}(y^b, g^b)\right), \end{aligned} \tag{12}$$

where $y^b := (g_1^b, \ldots, g_N^b), g^b$ is the batch samples and B is the number of batches. The derivative w.r.t the generative parameters θ does not require the Gumbel-Softmax approximation, and can be directly estimated by the Monte Carlo estimator

$$\frac{\partial}{\partial\theta} E_{q_\phi(z_{1:T}, d_{1:T})}[F(z, d)] \approx \frac{1}{B}\sum_{b=1}^{B}\frac{\partial}{\partial\theta}\left(F(z^b, d^b)\right). \tag{13}$$

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. (6).
4. Estimate the Monte Carlo approximation to $\nabla_\theta L$ using Eq. (13)
5. Estimate the SGVB approximation to $\nabla_\phi L$ using Eq.(12) with the Gumbel-Softmax approximation in Eq.(10);
6. Update $\theta^{(i)}$, $\phi^{(i)}$ using the ADAM.
end for

4 Related Work

It is an important approach to combine both SSMs and RNNs in time series modeling. The papers mostly close to our paper include [Johnson *et al.*, 2016; Krishnan *et al.*, 2015; 2016; Fraccaro *et al.*, 2016; Liu *et al.*, 2017]. For instance, [Krishnan *et al.*, 2015] combines variational auto-encoders with continuous state-space models, emphasizing the relationship to linear dynamical systems. [Krishnan *et al.*, 2016] lets inference network condition on both future and past hidden variables, which is an extension of Deep Kalman Filtering. And [Johnson *et al.*, 2016] employs general emission density for structured inference. [Fraccaro *et al.*, 2016] extends state space models by combining recurrent neural networks with stochastic latent variables. Different from the above methods that require the hidden states of SSMs be continuous, our paper utilizes discrete latent variables in the SSM part for better interpretability, especially in the applications of segmentation and labeling of high-dimensional time series.

In parallel, some research also works on variational inference with discrete latent variables recently [Bayer and Osendorfer, 2014; Mnih and Rezende, 2016]. [Bayer and Osendorfer, 2014] enhances RNNs with stochastic random variables, however, the score function or the REINFORCE approach used in that paper suffers from the high variance. [Mnih and Rezende, 2016] proposes a gradient estimator for multi-sample objectives that use the mean of other samples to construct a baseline. In Discrete VAE [Rolfe, 2016], the sampling is autoregressive through each binary unit, which allows every discrete choice to be marginalized out in a tractable manner.

In the aspect of optimization, [Khan *et al.*, 2015] splits the variational inference objective into the summation of one term to be linearized and another term being tractably concave. [Goyal *et al.*, 2017] takes into account the process of beam search itself through a continuous, sub-differentiable relaxation of beam search. In [Dai *et al.*, 2017a], the discrete optimization is replaced by the maximization over the negative Helmholtz free energy. In contrast to linearizing intractable terms around the current iteration as used in the above approaches, we handle intractable terms via recognition networks and amortized inference(with the aid of Gumbel-Softmax reparameterization [Jang *et al.*, 2017; Maddison *et al.*, 2016]) in this paper. That is, we use parametric function approximation to learn conditional evidence in a conjugate form.

5 Experiment

We evaluate the SSNN on several datasets across multiple scenarios. Specifically, we first evaluate its performance of finding complex structures and estimating data likelihood on a synthetic dataset and two speech datasets (TIMIT & Blizard). Then we test the SSNN for learning segmentation and latent labels on Human activity [Reyes-Ortiz *et al.*, 2016] dataset, Drosophila [Kain *et al.*, 2013] and PhysioNet Challenge[Springer *et al.*, 2016].

Temperatures of Gumbel-Softmax are fixed throughout training. We implement the proposed model based on Theano[1] and Block & Fuel [Van Merriënboer *et al.*, 2015]. The Adam [Kingma and Ba, 2014] optimizer was used in all experiments for our model.

5.1 Synthetic Experiment

To validate the ability of modeling 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 a 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 the 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 squares regression results are shown in Table 1. 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. The results indicate that generative model and inference network in SSNN are capable of capturing complex sequence dependency.

[1] http://deeplearning.net/software/theano/

5.2 Speech Modeling

We also test SSNN on the modeling of speech data, i.e., Blizzard and TIMIT datasets [Prahallad *et al.*, 2013]. Blizzard records the English speech with 300 hours by a female speaker. TIMIT is a dataset with 6300 English sentences read by 630 speakers. For the TIMIT and Blizzard datasets, the sampling frequency is 16KHz and the raw audio signal is normalized using the global mean and standard deviation of the training set. 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; 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], i.e. we report the average log-likelihood for half-second sequences on Blizzard, and report the average log-likelihood per sequence for the test set sequences on TIMIT. For the raw audio datasets, we use a fully factorized Gaussian output distribution.

In the experiment, We split the raw audio signals into 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 back-propagation 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. the temperature τ starts from a large value 0.1 and gradually anneals to 0.01.

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 are constrained to be independent across time steps. For SRNN [Fraccaro *et al.*, 2016], we compare with the smoothing and filtering performance denoted as SRNN (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]. From Table 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.

5.3 Segmentation and Labeling of Time Series

To show the advantages of SSNN over HSMM and its variants when learning the segmentation and latent labels from

		DVBF-LL		DKF		Our method (SSNN)	
		log-likelihood	R^2	log-likelihood	R^2	log-likelihood	R^2
Dependent	$\sin\phi$	3990.8	0.961	1737.6	0.929	4424.6	0.975
ground truth	$\cos\phi$	7231.1	0.982	6614.2	0.979	8125.3	0.997
variable	$\frac{d\phi}{dt}$	-11139	0.916	-20289	0.035	-9620	0.941

Table 1: Results for the ordinary least squares regressions of all latent states on respective dependent variable in pendulum dynamics. For both measures, the higher the better. Results of these models except SSNN were copied from [Karl *et al.*, 2016].

MODELS	Blizzard	TIMIT
VRNN-GMM	≥ 9107	≥ 28982
VRNN-GAUSS	≥ 9223	≥ 28805
VRNN-I-GAUSS	≥ 8933	≥ 28340
SRNN(smooth+Res_q)	≥ 11991	≥ 60550
SRNN(smooth)	≥ 10991	≥ 59269
SRNN(filt)	≥ 10846	50524
RNN-GMM	7413	26643
RNN-GAUSS	3539	-1900
Our Method(SSNN)	$\geq$ **13123**	$\geq$ **64017**

Table 2: Average log-likelihood per sequence on the test sets. The higher the better. Results of these models except SSNN were copied from original literature.

MODELS	DROSOPHILA	HUMANAC	PHYSIONET
HSMM	47.37 ± 0.27	41.59 ± 8.58	45.04 ± 1.87
SUBHSMM	39.70 ± 2.21	22.18 ± 4.45	43.01 ± 2.35
HDP-HSMM	43.59 ± 1.58	35.46 ± 6.19	42.58 ± 1.54
CRF-AE	57.62 ± 0.22	49.26 ± 10.63	45.73 ± 0.66
RHSMM-DP	36.21 ± 1.37	16.38 ± 5.06	31.95 ± 4.12
SSNN	**34.77** ± 3.70	**14.70** ± 5.45	**29.29** ± 5.34

Table 3: Mean and standard deviation of the error rate (%). Results of these models except SSNN were taken from original literature.

sequences, we take experiments on Human activity [Reyes-Ortiz *et al.*, 2016], Drosophila dataset [Kain *et al.*, 2013] and PhysioNet [Springer *et al.*, 2016] Challenge dataset. Both Human Activity and Drosophila dataset are used for segmentation prediction.

Human activity (HumanAc) 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.

Drosophila dataset records the time series movement of fruit flies' legs. At each time step t, $\mathbf{x}_t$ is a 45-dimension vector, which consists of the raw and some higher order features. In the experiment, we fix the τ at small value 0.0001, and the maximum time steps $T \approx 10,000$.

PhysioNet Challenge dataset 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.

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. We use the leave-one-sequence-out protocol to evaluate these methods, i.e., each time one sequence is held out for testing and the left sequences are for training. We set the truncation of max possible duration M to be 400 for all tasks. We also set the number of hidden states K to be the same as the ground truth.

We report the comparison with subHSMM [Johnson and Willsky, 2014], hdp-HSMM [Johnson and Willsky, 2013], CRF-AE [Ammar *et al.*, 2014] and rHSMM-dp [Dai *et al.*, 2017b]. For the hdp-HSMM and subHSMM, the observed sequences $\mathbf{x}_{1:T}$ are generated by standard multivariate Gaussian distributions. The duration variable d_t is modeled by the Poisson distribution. We need to tune the concentration parameters α and γ. As for the hyperparameters, 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 a mixture of Gaussian for the emission probability. For R-HSMM-dp, it is a version of R-HSMM with the exact MAP estimation via dynamic programming.

Experimental results are shown in Table 3. 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.

6 Conclusion

In order to learn the structures (e.g., the segmentation and labeling) of high-dimensional time series in an unsupervised way, we have proposed a Stochastic Sequential Neural Network (SSNN) with structured inference. For better model interpretation, we further restrict the labels and segmentation to be two sequences of discrete variables respectively. In order to exploit forward and backward temporal information, we carefully design a structured inference method. 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 in SSNN have been fully demonstrated in both synthetic and real-world sequential benchmarks.

Acknowledgments

This paper was in part supported by Grants from the Natural Science Foundation of China(No. 61572111), 1000-Talent Startup Funds (Nos.G05QNQR004, A1098531023601041), and three Fundamental Research Funds for the Central Universities of China (Nos.ZYGX2016Z003, ZYGX2016J078 and ZYGX2017KYQD17), a Project funded by China Postdoctoral Science Foundation (No.2016M602674).

References

[Ammar *et al.*, 2014] Waleed Ammar, Chris Dyer, and Noah A Smith. Conditional random field autoencoders for unsupervised structured prediction. In *Advances in Neural Information Processing Systems*, pages 3311–3319, 2014.

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

[Bayer and Osendorfer, 2014] Justin Bayer and Christian Osendorfer. Learning stochastic recurrent networks. *arXiv preprint arXiv:1411.7610*, 2014.

[Chiappa and others, 2014] Silvia Chiappa et al. Explicit-duration markov switching models. *Foundations and Trends® in Machine Learning*, 7(6):803–886, 2014.

[Chung *et al.*, 2015] Junyoung Chung, Kyle Kastner, Laurent Dinh, Kratarth Goel, Aaron C Courville, and Yoshua Bengio. A recurrent latent variable model for sequential data. In *Advances in neural information processing systems*, pages 2980–2988, 2015.

[Chung *et al.*, 2016] Junyoung Chung, Sungjin Ahn, and Yoshua Bengio. Hierarchical multiscale recurrent neural networks. *arXiv preprint arXiv:1609.01704*, 2016.

[Dai *et al.*, 2017a] Bo Dai, Ruiqi Guo, Sanjiv Kumar, Niao He, and Le Song. Stochastic generative hashing. *arXiv preprint arXiv:1701.02815*, 2017.

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

[Fraccaro *et al.*, 2016] Marco Fraccaro, Søren Kaae Sønderby, Ulrich Paquet, and Ole Winther. Sequential neural models with stochastic layers. In *Advances in Neural Information Processing Systems*, pages 2199–2207, 2016.

[Goyal *et al.*, 2017] Kartik Goyal, Graham Neubig, Chris Dyer, and Taylor Bergkirkpatrick. A continuous relaxation of beam search for end-to-end training of neural sequence models. *arXiv preprint arXiv:1708.00111*, 2017.

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

[Jang *et al.*, 2017] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017.

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

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

[Johnson *et al.*, 2016] Matthew Johnson, David K Duvenaud, Alex Wiltschko, Ryan P Adams, and Sandeep R Datta. Composing graphical models with neural networks for structured representations and fast inference. In *Advances in Neural Information Processing Systems*, pages 2946–2954, 2016.

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

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

[Khan *et al.*, 2015] Mohammad E Khan, Pierre Baqué, François Fleuret, and Pascal Fua. Kullback-leibler proximal variational inference. In *Advances in Neural Information Processing Systems*, pages 3402–3410, 2015.

[Kingma and Ba, 2014] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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

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

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

[Liu *et al.*, 2017] Hao Liu, Haoli Bai, Lirong He, and Zenglin Xu. Stochastic sequential neural networks with structured inference. *arXiv preprint arXiv:1705.08695*, 2017.

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

[Mnih and Rezende, 2016] Andriy Mnih and Danilo J Rezende. Variational inference for monte carlo objectives. *arXiv preprint arXiv:1602.06725*, 2016.

[Murphy, 2002] Kevin P Murphy. Hidden semi-markov models (hsmms). *unpublished notes*, 2, 2002.

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

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

[Rolfe, 2016] Jason Tyler Rolfe. Discrete variational autoencoders. *arXiv preprint arXiv:1609.02200*, 2016.

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

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

[Sutskever *et al.*, 2014] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*, pages 3104–3112, 2014.

[Van Merriënboer *et al.*, 2015] Bart Van Merriënboer, Dzmitry Bahdanau, Vincent Dumoulin, Dmitriy Serdyuk, David Warde-Farley, Jan Chorowski, and Yoshua Bengio. Blocks and fuel: Frameworks for deep learning. *arXiv preprint arXiv:1506.00619*, 2015.