Attentive state-space model revisit

Yan Song [1 2]

Abstract

This report provides experimental details and additional evaluation to the paper *Attentive State-space Model of disease progression*(Alaa & van der Schaar, 2019). In particular, we fill in the gap by providing a few small-scaled experiments on the posterior sampling methods that was not fully discussed in the original paper. We focus mainly on the machine learning part and propose a few sampling methods that are worth exploring, including Multinomial sampling, *Gumbel-max* sampling and the integration of these two methods. We perform primary tests of the models on synthetic data-set and concentrate on the quality of the attention weights that are recovered unsupervisingly. The results reveal practical issues faced by these two sampling methods and that the integration version is able to achieve balance in between.

1. Introduction

Disease progression has been an essential area of study in medical informatics (Mould, 2012). Since frequent medical tests can be either inconvenient and damaging to patients' health, a model that is capable of disease progression prediction can be beneficial to both the patients and the medical workers. Examples of these includes application on chronic diseases such as Alzheimer's Disease (Wang et al., 2014; Fisher et al., 2019; Nguyen et al., 2020). Traditional disease progression modelling favours HMM (hidden markov models) (Jackson et al., 2003; Liu et al., 2015) or its variants (Wang et al., 2014; Alaa et al., 2017). Recently with the rising of deep learning and the availability of electronic health record (EHR) data, many researchers have also investigated the performance of neural networks in this area. This report sets out to discuss and evaluate one deep probabilistic model among them, named *Attentive State-Space model* proposed by (Alaa & van der Schaar, 2019) in an unsupervised task. In particular, we look at the posterior sampling problems and propose a few potential methods purely from a machine learning perspective. The meaning of this in disease progression is to explore better algorithms that provide a more accurate estimate of the underlying unseen dynamics. Our contribution in this report includes:

- filling in the details on model formulation and the practical implementation.
- proposing a few posterior sampling methods and have done primary experiments on synthetic data-set.

2. related work

State-space modelling of disease progression

Many models in disease progression literature stem from Hidden Markov Model (HMM). (Jackson et al., 2003) estimates the transition rate and compute probability of stage error on HMM. (Liu et al., 2015) presented the continuous-time HMM along with an efficient EM algorithm for model learning. (Wang et al., 2014) investigate a multi-layer HMM which learns a continuous stage progression from discrete time in an unsupervised manner. However, the Markovian assumption weaken the model's interpretability on long-term sequential tasks (Alaa & van der Schaar, 2019).

Recurrent Neural Network

On the other hand, sequential deep learning models such as recurrent neural network (RNN) are capable of long-term prediction, an example is the application on Alzheimer's Disease (Wang et al., 2018). Meanwhile, there have also be multiple attempts on designing an interpretable deep learning model in disease progression modelling. One way is to apply attention mechanism (Vaswani et al., 2017), an idea coming from natural language processing aiming at explaining the interrelationship between words. *RETAIN* (Choi et al., 2016) applied RNN on data in an reverse order and make a prediction based on the generated attention. *Dipole* (Ma et al., 2017) employed a bi-directional RNN and evaluated the performance on three commonly-used attention formulation. Whereas Attentive state-space model (Alaa & van der Schaar, 2019) explored an non-Markovian probabilistic model in the light of attention.

Deep probabilistic models

Injecting uncertainty estimation in deep learning models has been shown to better capture the variation in the data compared to a deterministic model (Bayer & Osendorfer, 2014). A common way is by combining RNN with latent variable model, such as state-space models (Chung et al., 2015; Fraccaro et al., 2016; Hafner et al., 2019a;b). The main difference among these works is how the latent variable interacts with deterministic recurrent layer. In Attentive state-space models, the recurrent layer models the attention weights from the data and also provide summarisation of the future information to the posterior network. The latent discrete stages of disease progression are modelled by a state-space model with non-Markovian dynamics (more details in Section 3.2).

2.1. Posterior sampling

A major part of a deep probabilistic model is how the posterior distribution of latent given data is modelled. Due to the non-linearity in neural network, many researchers implemented extra networks or layers to capture the sufficient statistics that are essential for the target posterior distribution. Such sufficient statistics can be collected from past hidden state (Bayer & Osendorfer, 2014), past observations (Chung et al., 2015) and future observations (Fraccaro et al., 2016). As for the Attentive state-space model, the past observations are integrated into a sequence of attention weights and the future observation are encoded into a hidden representation through a backward RNN, which together with past latent states shape the posterior distribution. However, during inference time, the objective function needs to be estimated by sampling due to intractability. Common sampling method includes MCMC (Andrieu et al., 2003),*sequence Monte-carlo* (Smith, 2013), importance sampling (Liu et al., 2001), ,REINFORCE trick (Williams, 1992), reparametrisation trick (Kingma & Welling, 2014) and their integration (Parmas et al., 2018), etc.

However, to the best of our knowledge, not much works have been done on the sampling methods of a deep non-Markovian state-space models with transition matrix involved. In this report we are handling discrete posterior with a categorical distribution. According to the explicit structure of the model, we propose a standard sequential Monte-Carlo sampling methods with multinomial distribution and also a reparametrisation version where we apply Gumbel-Max trick (Gumbel, 1954; Maddison et al., 2014). This enables us to back-propagate the error through previous time step and obtain a relatively global estimate of gradients.

The next section discusses in details the structure of Attentive space-space models and also its objective function for training. Section 3.2 provides a description on two (three actually) different sampling methods and Section 5 shows the setting of experiments and the corresponding issues. Section 6 is where the results of the experiments are and the last section contains conclusion and potential future work.

3. Problem formulation

3.1. EHR data

The EHR data of a patient can be represents as sequences of time-labeled multivariate observations (Choi et al., 2016), where time represents each visit at a medical institution and each dimension of the observations contains information such as biomarkers and clinical events (Alaa & van der Schaar, 2019). Mathematically, we can denote a sequence of EHR data as:

$$\mathcal{X} = \{\boldsymbol{x}_i^{(n)}, t_i^{(n)}\}_{i=1}^{I} \quad , \quad n = 1, ..., N \tag{1}$$

where $\boldsymbol{x}_i^{(n)}$ denotes the observation of n^{th} patient at i^{th} visit and t_i^n is the time of visit. In this report we assume that the data is regularly-spaced and we omit the time label $t_i^{(n)}$ for simplicity. In the experiment, we generate synthetic data-set for evaluation.

3.2. Attentive state-space model revisit

Proposed by (Alaa & van der Schaar, 2019), the incorporation of attention mechanism enables the model to memorise historical information while providing interpretability for the progression of disease at the same time. A schematic plots of the model structure is given in Fig (1).

Emission and transition

The model is built on a probabilistic state-space model with latent states denoted as $\boldsymbol{z}_t, t = 1, ..., T$. In our case the latent has a discrete (categorical) distribution over all labels, which is the set of all (K) possible stages in disease progression, denoted as: $\boldsymbol{z} \in \mathcal{Z} = \{1, 2, ..., K\}$:

$$\boldsymbol{z} \sim \text{Cat}\big[p_1, p_2, ..., p_K\big] \tag{2}$$

In disease progression, different stages may corresponds to different level of severeness. For example, Chronic Obstructive Pulmonary Disease (COPD) can take more than 10 years to evolve from Stage I (mild) to Stage IV (very severe) (Wang et al., 2014). At each latent state $\boldsymbol{z}_t$, an probabilistic observation is generated following some *emission* distribution such as a Gaussian distribution:

$$P(\boldsymbol{x}_t|\boldsymbol{z}_t) = \mathcal{N}(\boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t) \tag{3}$$

Then the joint distribution of the latents $\boldsymbol{z}_{1:T}$ and observa-

tions $\boldsymbol{x}_{1:T}$ can be factorised as:

$$P(\boldsymbol{z}_{1:T}, \boldsymbol{x}_{1:T}) = P(\boldsymbol{z}_1) \prod_{t=1}^{T} P(\boldsymbol{x}_t|\boldsymbol{z}_t) P(\boldsymbol{z}_t|\boldsymbol{z}_{1:t-1}, \boldsymbol{x}_{1:t-1}) \tag{4}$$

As we can see that our model does not follow the *Markov property*. This emphases our assumptions that the disease progression is not Markov and so that our model need to have long-term memory, which is essential to chronic diseases modelling.

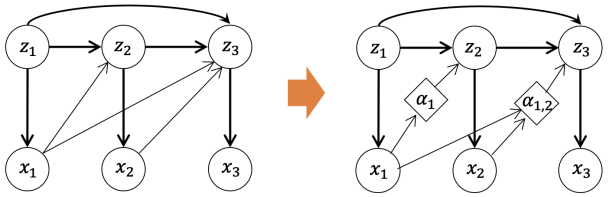

Figure 1. **Left:** The state-space model contains latent states $\boldsymbol{z}_t$ and observations $\boldsymbol{x}_t$. At each time step t, the transition in latent states take both previous states and observations into consideration. **Right:** an attentive state-space model summarising past observations with a sequence of attention weights $\boldsymbol{\alpha}$, and the past latent states get re-weighted correspondingly.

What differentiates the model from other state-space models is the attention mechanism (Vaswani et al., 2017) applied in latent transition (as shown in Fig 1 right). This is in order to capture the non-Markov dynamic by shifting the focus on past disease stages according to the interaction between what has been observed at current time step and the past observations. Even though such an interaction is uninterpretable (the attention weights are generated by a neural network), it can inform us how previous disease stages influence the current stage, providing interpretability in latent states.

Given a sequence of attention weights generated from the data, the latent transition follows some initial distribution $\boldsymbol{\pi} \in \mathbb{R}^K$ and baseline transition matrix $\boldsymbol{P} \in \mathbb{R}^{K \times K}$ and is formulated as [1]:

$$\begin{aligned} P(\boldsymbol{z}_t|\boldsymbol{z}_{1:t-1}, \boldsymbol{x}_{1:t-1}) &= P\big(\boldsymbol{z}_t|\boldsymbol{z}_{1:t-1}, \boldsymbol{\alpha}^t_{1:t-1}(\boldsymbol{x}_{1:t})\big) \\ &= \sum_{k=1}^{t-1} \alpha_k^t \boldsymbol{P}(\boldsymbol{z}_k, \boldsymbol{z}_t) \ , \quad \forall t \geq 1 \\ \Big(\sum_{k=1}^{t-1} \alpha_k^t = 1 \ &, \ \sum_j p_{ij} = 1 \Big) \end{aligned} \tag{5}$$

where $\alpha_k^t \in \mathbb{R}$ represent the attention weight at historical time step k when the current time step is t.

[1] The inclusion of current observation $\boldsymbol{x}_t$ in conditional set is valid in an unsupervised manner.

Attention network

The author proposed an end-to-end attention network which is different compared to common attention nets used in many natural language processing (NLP) tasks. According to Fig (2), an RNN encoder network first encode current and all previous observations $\mathbf{x}_{1:t}$ into a hidden representation $\boldsymbol{h}_t^e$, then it passes it to a RNN decoder where a target attention sequence $\{\alpha_1^t, ..., \alpha_{t-1}^t\}$ is generated, before it goes through the softmax layer for normalisation.

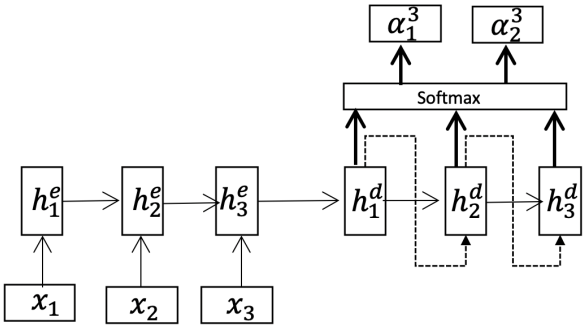

Figure 2. Sequence-to-sequence attention network. The last hidden state of the encoder $\boldsymbol{h}_3^e$ is the initial state $\boldsymbol{h}_1^d$ of the decoder RNN. After the initial time step, the decoder takes previous output as the input for the next time step.

Variational inference

The model training is performed by *Bayesian inference* (Barber, 2012) with maximum likelihood (ML) objective function. However, due to the use of neural network and the non-linearity it brings in, the computation of posterior distribution $p(\boldsymbol{z}|\boldsymbol{x})$ is intractable. Therefore the model is trained by variational inference with a posterior network $q(\boldsymbol{z}|\boldsymbol{x})$ aiming at approximating the true posterior. The objective is to maximise the following marginal likelihood

distribution:

$$
\begin{aligned}
logP(\boldsymbol{x}_{1:T}) &= log \int P(\boldsymbol{x}_{1:T}, \boldsymbol{z}_{1:T})\, d\boldsymbol{z}_{1:T} \\
&= log \int q(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T}) \frac{P(\boldsymbol{x}_{1:T}, \boldsymbol{z}_{1:T})}{q(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T})}\, d\boldsymbol{z}_{1:T} \\
&\geq \int q(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T})\, log \frac{P(\boldsymbol{x}_{1:T}, \boldsymbol{z}_{1:T})}{q(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T})}\, d\boldsymbol{z}_{1:T} \\
&= \mathbb{E}_q \Big[log\ P(\boldsymbol{x}_{1:T}|\boldsymbol{z}_{1:T}) + log\ P(\boldsymbol{z}_{1:T}|\boldsymbol{\alpha}_{1:T-1}) \\
&\qquad\qquad - log\ q(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T}) \Big] \\
&= \mathbb{E}_q \Big[\sum_{t=1}^{T} log\ P(\boldsymbol{x}_t|\boldsymbol{z}_t) + log\ P(\boldsymbol{z}_1) \\
&\qquad + \sum_{t=1}^{T} log\ P(\boldsymbol{z}_t|\boldsymbol{z}_{1:t-1}, \boldsymbol{\alpha}^t_{1:t-1}) - log\ q(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T}) \Big]
\end{aligned}
\tag{6}
$$

The inequality is due to the *Jensen's inequality* (Barber, 2012; Bishop, 2006) and $q(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T})$ is the estimated posterior distribution of the latents given the observations. Therefore, we have transformed the objective function into an expectation term, which is also known as *Free energy* or *Evidence lower bound* (ELBO). Note that $P(\boldsymbol{z}_{1:T}|\boldsymbol{\alpha}_{1:T-1})$ acts as a conditional prior for the latent states in this setting.

The formula for conditional likelihood distribution $P(\boldsymbol{x}|\boldsymbol{z})$ and the transition probability $P(\boldsymbol{z}|\boldsymbol{\alpha})$ have been given in Equation (3)&(4). The choice of posterior $q(\boldsymbol{x}_{1:T}|\boldsymbol{z}_{1:T})$ is subjected to change. The author factorised the q in a way that both historical and future information is injected to the network [2]:

$$
q(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T}) = q(\boldsymbol{z}_1|\boldsymbol{x}_{1:T}) \prod_{t=2}^{T} q(\boldsymbol{z}_t|\boldsymbol{z}_{1:t-1}, \boldsymbol{\alpha}^t_{1:t-1}, \boldsymbol{x}_{t:T})
\tag{7}
$$

In practice, the posterior sufficient statistics are modelled by neural network. Given that the latents are discrete variables so the posterior is a categorical distribution: $q(\boldsymbol{z}_t) \sim Cat(p_1, p_2, ..., p_K)$. Then the probabilities for each class $\{p_1, ..., p_K\}$ are the output of some posterior networks with softmax layer at the end. The network takes as input a sequence of attention weights along with future observations. The attentions are provided by the Seq2Seq attention net (Fig 2) and the future information $\boldsymbol{x}_{t:T}$ is summarised by the last hidden state of a backward RNN. A full illustration of the inference network is shown in Fig (3).

Posterior sampling

[2] This is a (Kalman) *smoothing*-style posterior. One can also try a *filtering* posterior without the future observation $\boldsymbol{x}_{t:T}$, which enables *online learning* and can be put as potential future works.

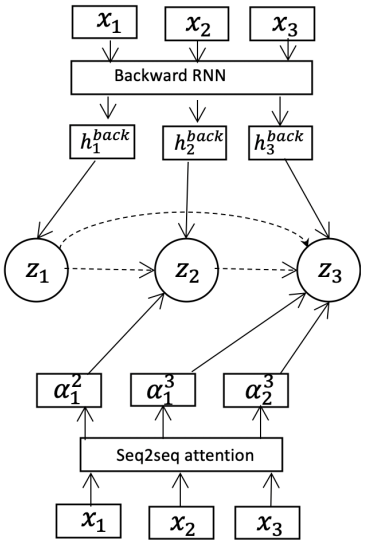

Figure 3. Inference network. At each latent states $\boldsymbol{z}_t$, the messages are coming from: (1) seq3seq attention (lower); (2) hidden states of backward RNN (upper) and (3) previous latent states $\boldsymbol{z}_{1:t-1}$. The dashed line represents sampling.

The expectation term in Equation (6) is intractable to compute. According to the original paper, the inference is performed by sampling from the posterior sequentially:

$$
(\boldsymbol{z}_1^{(i)}, ..., \boldsymbol{z}_T^{(i)}) \sim q(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T})\ ,\quad i = 1, ..., N
\tag{8}
$$

The detail procedure can be summarised as follow:

1. At time t, given all previous samples $\tilde{\mathcal{Z}} = \{\tilde{\boldsymbol{z}}_1, \tilde{\boldsymbol{z}}_2, ..., \tilde{\boldsymbol{z}}_{t-1}\}$, we compute the attention sequences $\{\alpha_1^t, \alpha_2^t, ..., \alpha_{t-1}^t\}$ and the backward RNN hidden state $\boldsymbol{h}_t^{back}$ from the observation sequence $\{\boldsymbol{x}_1, ..., \boldsymbol{x}_T\}$ (Fig 3).

2. Compute the latent transition probability $P(\boldsymbol{z}_t|\tilde{\boldsymbol{z}}_{1:t-1}, \boldsymbol{\alpha}^t_{1:t-1}(\boldsymbol{x}_{1:t}))$ (Equation 5) and denote it as $\tilde{\boldsymbol{p}}^t_{forward}$. In discrete cases this is a categorical distribution.

3. Concatenate the obtained forward distribution with the backward hidden state $\tilde{\boldsymbol{h}}_q^t = [\boldsymbol{h}_t^{back}, \tilde{\boldsymbol{p}}^t_{forward}]$ and feed it to a combiner function $C(\cdot)$ to get the posterior distribution (discrete):

$$
q(\boldsymbol{z}_t) = \boldsymbol{C}(\tilde{\boldsymbol{h}}_q^t) = \text{Softmax}(\boldsymbol{W}_q^T \tilde{\boldsymbol{h}}_q^t + \boldsymbol{b}_q)
$$

4. Sample again from the new posterior and append it to the sample lists:

$$
\tilde{\boldsymbol{z}}_t \sim q(\boldsymbol{z}_t)\ ,\quad \tilde{\mathcal{Z}} = \{\tilde{\boldsymbol{z}}_1, ..., \tilde{\boldsymbol{z}}_{t-1}, \tilde{\boldsymbol{z}}_t\}
$$

5. $t+1 \rightarrow t$. Go back to step 1 and repeat until the last time step T. The resulting sample list is $\tilde{\mathcal{Z}}$.

following this is the stochastic gradient descent on the *Monte-Carlo* estimate (Bolstad & Curran, 2016) of Equation 6. However, few detail has been given on how samples are explicitly generated. Therefore in the next section, we discuss a few potential sampling methods and how we implement them in practice.

4. Posterior sampling methods

The sampling procedure discussed above is called *Sequential Monte-Carlo sampling* (Smith, 2013) or *particle filtering* (Doucet & Johansen, 2009). At the initial state, we sample N amount of samples, or named 'particles', then these samples are passed through some non-linear function $\boldsymbol{f}$ and we obtain the transformed samples for the next time step. By repeating this procedure we gain multiple sampled trajectories from which we can estimate the gradient of the objective function. The model parameters includes:

- **Attention network**: encoder-decoder RNN model parameters
- **Backward network**: backward RNN model parameters
- **Inference network**: Combiner function $\boldsymbol{C}$

In the experiment, the initial distribution $\boldsymbol{\pi}$ and baseline transition matrix $\boldsymbol{P}$ are estimated from the data using *Gaussian Mixture models* (Reynolds, 2009), so are the emission distribution parameters $\boldsymbol{\mu}, \boldsymbol{\Sigma}$. This is in order to evaluate the effect of different sampling methods without the disturbance of emission network.

Next, we propose three sampling methods: Multinomial sampling, Gumbel-Mas sampling and periodic sampling.

4.1. Particle filtering with local gradient estimate. (Multinomial sampling)

As shown in Fig (4). This is a sampling method that compute the gradient of likelihood function only with the particles at current time step. At time t, the model receive a list of particles from the past $\tilde{\mathcal{Z}} = \{\boldsymbol{z}_1^{(i)}, ..., \boldsymbol{z}_{t-1}^{(i)} | ; i = 1, ..., N\}$, then for each set of particles (same set means the same index i) the model computes the corresponding categorical posterior distribution $q^i(\boldsymbol{z}_t)$, $i = 1, ..., N$ using the same attention weights $\{\boldsymbol{\alpha}_{1:t-1}^t\}$ and backward hidden state $\boldsymbol{h}_t^{back}$. Then a summation of Gaussian distribution can be calculated from each of these posterior $q^{(i)}$ and the resulting emission distribution is the average over all particles:

$$log\ P(\boldsymbol{x}_t|\boldsymbol{z}_t) = \frac{1}{N}\sum_{i=1}^{N} log\ \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}|q^{(i)}(\boldsymbol{z}_t)) \quad (9)$$

Moving on to the next time step $t+1$, a new set of particles are generated and the sample set is updated:

$$\boldsymbol{z}_t^{(i)} \sim q^{(i)}(\boldsymbol{z}_t), i = 1, ..., N \ \text{ (Categorical)}$$
$$\tilde{\mathcal{Z}} = \{\boldsymbol{z}_1^{(i)}, ..., \boldsymbol{z}_{t-1}^{(i)}, \boldsymbol{z}_t^{(i)} | i = 1, ..., N\}$$

Subsequently, the gradient $\nabla logP(\boldsymbol{x}_t|\boldsymbol{z}_t)$ is back-propagate to the backward hidden state $\boldsymbol{h}_t^{back}$ and the attention weights $\{\boldsymbol{\alpha}_{1:t-1}^t\}$ applied in time t only, demonstrating the myopia in model training. It is the sampling procedure that stop the gradient flowing backward to previous time step. Hence, an alternative sampling method is to use reparametrisation trick (Kingma & Welling, 2014), enabling the back-propagation through sampling, even in discrete cases.

4.2. Particle filtering with reparametrisation trick (Gumbel-Max sampling)

The reparametrisation trick has been widely-used on Gaussian distribution where a sample is represented by the sum of the mean $\boldsymbol{\mu}$ and the product of standard deviation $\boldsymbol{\sigma}$ and a i.i.d. Gaussian noise $\boldsymbol{\epsilon}$:

$$\boldsymbol{z} = \boldsymbol{\mu} + \boldsymbol{\sigma} \cdot \boldsymbol{\epsilon}\ , \quad \boldsymbol{\epsilon} \sim \mathcal{N}(0, 1)$$

Similarly, in discrete case, we can also achieve it by applying *Gumbel-Max trick* (Gumbel, 1954; Maddison et al., 2014; Kool et al., 2019). It enables sampling from the categorical distribution, by adding independent Gumbel distributed noises and returning the category with maximum perturbed log-probability. It has the same idea as Gaussian reparametrisation where the sampling procedure is separated from the sufficient statistics of the distribution. By applying this, the model is able to tune the parameters while considering the output from previous time step, just like the blue dashed line in Fig (4).

4.3. Periodic sampling

We have also tested an integrated sampling methods that interchange between multinomial sampling and Gumbel-Max sampling periodically. Therefore, the model is able to tune the parameter by looking back a few steps only while not overdoing it. For example, at time t, we passed the Gumbel-Max samples obtained from previous time step $\boldsymbol{z}_t^{(i)}$ to the combiner function and obtain the corresponding posterior distribution $q(\boldsymbol{z}_{t+1}^{(i)})$, then we switch to Multinomial sampling and pass the samples to the combiner function again.

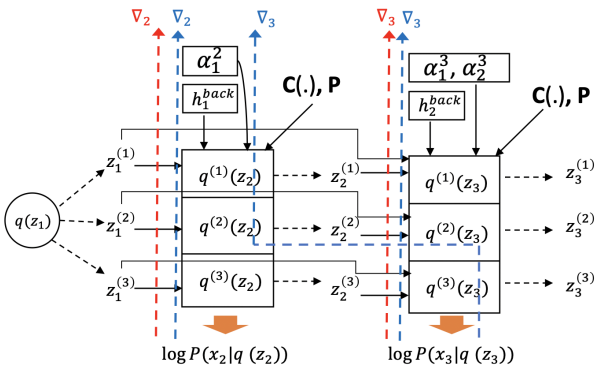

Figure 4. Start from the initial state, we sample 3 particles $\boldsymbol{z}_1^{(1)}, \boldsymbol{z}_1^{(2)}, \boldsymbol{z}_1^{(3)}$ from a categorical distribution $q(\boldsymbol{z}_1)$, then we feed these samples to the posterior network which also takes the attention and backward hidden states as input. For each particle a new posterior (discrete) distribution $q^{(i)}(\boldsymbol{z}_2)$ is generated from combiner function $\boldsymbol{C}$. Next we sample again from the new posterior distribution to obtain the particles for the next time step $\boldsymbol{z}_2^{(1)}, \boldsymbol{z}_2^{(2)}, \boldsymbol{z}_2^{(3)}$ and so on. At each time step a log likelihood function can be computed from the samples and the gradient is back-propagate to the combiner function C, the backward RNN and the Seq2Seq attention network. The dark dashed line represents sampling operation. the red and blue dashed line represents the direction of the gradient ∇_t from time step t, where color red refers to local gradient (Sec. 4.1) and blue refers to reparametrisation gradient (Sec. 4.2).

In next section, we discuss the experimental details on how to implement the idea of different sampling methods and the corresponding evaluation we applied.

5. Experiments

5.1. Synthetic data

To better evaluate the model performance from a machine learning perspective, we test the models on synthetic data-set first. The data-set is generated based on a fixed initial distribution $\boldsymbol{\pi}$, baseline transition matrix $\boldsymbol{P}$ and also the sufficient statistics for the emission distribution $\boldsymbol{\mu}, \boldsymbol{\Sigma}$. The unseen attention weights is decided by the current time step t and also a hyper-parameter ρ as:

$$\alpha_{t'}^{t} = \frac{exp\big(-\rho \cdot (t - t^{'} - 1)\big)}{\sum_{k=1}^{t-1} exp\big(-\rho \cdot (t - k - 1)\big)}, \quad t^{'} < t \tag{10}$$

Thus, higher the value of ρ is, more concentrated the attention weights are. A illustration is shown in Fig (5), higher ρ means closer to a Markovian dynamics. The default number of categories is set 3, the dimension of observations is set as 10, the length of data is set as 15, the baseline transition matrix and initial distribution are:

$$\boldsymbol{P} = \begin{pmatrix} 0.9 & 0.04 & 0.06 \\ 0.6 & 0.1 & 0.3 \\ 0.1 & 0.8 & 0.1 \end{pmatrix}$$

$$\boldsymbol{\pi} = (0.5, 0.3, 0.2)$$

The emission mean and diagonal covariance are:

$$\boldsymbol{\mu} = \begin{pmatrix} -10 \\ 5 \\ 10 \end{pmatrix}, \quad \boldsymbol{\Sigma} = \begin{pmatrix} 0.25 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 2.25 \end{pmatrix}$$

and the latent transition follows the probability $P(\boldsymbol{z}|\boldsymbol{z}_{1:t-1}, \boldsymbol{\alpha}_{1:t-1}^{t})$ in Equation 5. Fig (6) illustrate how the observed data $\boldsymbol{x}_t$ varies according to the latent states $\boldsymbol{z}_t$.

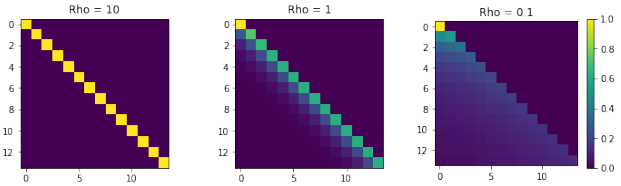

Figure 5. The matrix plot of normalised attention weights when **Left**: $\rho = 10$; **Mid**: $\rho = 1$ and **Right**: $\rho = 0.1$. The x-axis is the current time step minus one and the y-axis is the previous time step. Left graph represents (almost) a Markovian transition.

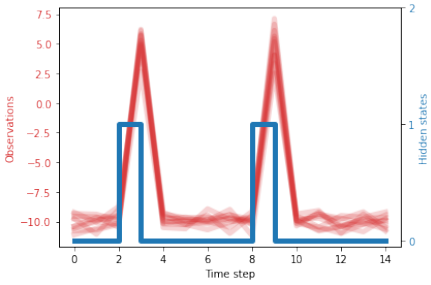

Figure 6. The observations (i.e. biomarkers) varies according to the hidden states (i.e. disease progression stage). The red curve represent the observations and the blue curve represents the hidden states.

5.2. Operates in batches

Fitting models in batches is essential for computational efficiency. In the experiment, we perform the transition of $\boldsymbol{z}$ (Equation 5) in batches as illustrated in Fig (7).

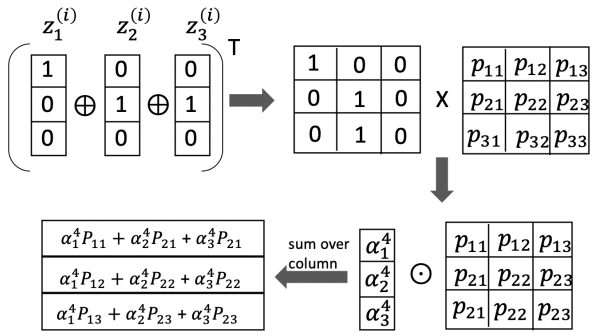

Figure 7. Operation in batches. **First**: concatenate all previous samples of $\boldsymbol{z}$ and take the transpose; **Second**: matrix multiplication with baseline matrix $\boldsymbol{P}$; **Third**: Multiply element-wisely with the attention weights; **Last**: summing over the column and obtain the (categorical) posterior distribution on next latent $\boldsymbol{z}_4$.

5.3. Hyper-parameters setting

We implement the model with *Pytorch* (Paszke et al., 2017). For model training, we use Adam (Kingma & Ba, 2014) with learning rate $1e-3$ and a mini-batch of 50 data samples. For sequential model (attention network and backward RNN) we use Long-Term-Short-Memory (LSTM) (Hochreiter & Schmidhuber, 1997) with hidden dimension 64. The combiner function $\boldsymbol{C}$ is modelled by a two-layer neural network with Tanh activation function. This is because in the experiment we found that a ReLU activation can cause instability to the posterior where the probability score for the categorical distribution becomes undesirably closed to one whereas a Tanh activation is better at regularising the output. The initialisation of the model parameter is set as the same for every experiments performed.

5.4. Issues with local-minima

We first experiment with the Multinomial and Gumbel-Max sampling models. We train both the models on 2,000 data sequences. The attention hyper-parameter ρ is set at 1. However, we discovered that the models occasionally suffer from local-minima and the model with local gradient estimate is worse. They both get stuck at where the attention network only look at the initial state, as illustrated in Fig (9)&(10) in Appendix A. This might be due to the structure of the attention network, where the first output of the attention weight is placed at the initial time step and the model may blindly focus on the initial state since the following states would experience much more uncertainty during early stage of model training. We can also observe from the graph that the model with Gumbel-max trick is able to escape from local-minima faster than the model with multinomial sampling. This is likely due to the long-sightedness of reparametrisation trick. However, this conclusion still needs more experiment backup and for the moment we avoid such local-minima behaviour by adjusting the attention network. We simply reverse the order of the generated attention weights sequence, such that the first generated weights is applied to the last states and the last weights is applied to the initial states. This is, however, not a valid solution since in practice we often have no idea on what the order of attention weight would be.

In next section, we discuss the results we get on models with both sampling methods and also the integrated version.

6. Results

6.1. Multinomial Versus Gumbel-Max

We train both models on 2,000 data for 1,000 epochs with sampling size 4,000. The training plots are shown in Fig (8). The training loss is the objective function (negative of Equation 6), the conditional log-likelihood is denoted as $P(\boldsymbol{x}_{1:T}|\boldsymbol{z}_{1:T})$ and the KL divergence is a measure of distance between the transition distribution (Equation 5) and the posterior distribution, denoted as:

$$\mathbf{KL}[q\|p] = \mathbb{E}_{q(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T})}\left[log\frac{q(\boldsymbol{z}_{1:T}|\boldsymbol{x}_{1:T})}{p(\boldsymbol{z}_{1:T}|\boldsymbol{\alpha}_{1:T-1})}\right] \tag{11}$$

Therefore, the objective function can be re-written as:

$$\mathcal{L} = -log\ P(\boldsymbol{x}_{1:T}) = -\mathbb{E}_q\left[\sum_{t=1}^{T} log\ P(\boldsymbol{x}_t|\boldsymbol{z}_t)\right] - \mathbf{KL}\left[q(\boldsymbol{z}_{1:T})\middle\|p(\boldsymbol{z}_{1:T}|\boldsymbol{\alpha}_{1:T-1}))\right] \tag{12}$$

and minimise $\mathcal{L}$ is equivalent to maximise the log-likelihood while minimising the KL divergence.

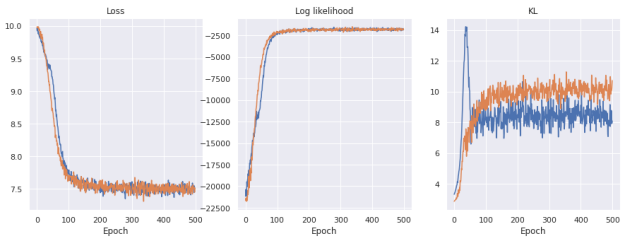

Figure 8. Training plots of loss (**Left**); conditional log likelihood (**Mid**) and KL divergence (**Right**). The blue and orange curve represents models with multinomial and Gumbel-max sampling methods respectively.

As we can see from the plots, both model are having a similar training pattern except a slight difference in KL term. This implies that both model have the similar ability to learn and reconstruct the observed data. The learned attention

weights are shown in Fig (11). The model with multinomial sampling method tends to capture the general pattern of attention weights much faster while the Gumbel-max model found it harder to shift its focus away from the initial states. Meanwhile, the multinomial sampling model converges to (or closed to) a Markovian dynamics where the attention weights are mainly localising on one step before, whereas the Gumbel-Max slowly converge to a reasonable pattern of attention weights. The slow convergence of Gumbel-Max sampling is likely because that the gradient need to flow backward to all previous time steps which can be more computational expensive and time-consuming. On the other hand, the reason for the behaviour of multinomial sampling model might be due to the myopia of local gradient estimate, however, since the scale of the experiment is relatively small, it is better to perform the experiment on a larger scale and we leave it for future work.

6.2. Periodic sampling

Fig (12) shows the results of predicted attention weight on the same set of unseen data after using a periodic sampling methods. The time interval is tuned to be 2, which means that the gradient can passed backward to only one previous time step (exactly what Fig 4 demonstrates). As we can observe that,even though the network learn a Markovian dynamics after epoch 150, it is able to escape from it and try to learn the true attention weights. Meanwhile, it has faster convergence compared to purely Gumbel-Max sampling. This is an interesting result since it absorbs the advantages from both methods and expectedly more testing can be performed in the future regarding the applicability of such method on more complex data.

7. conclusion and future work

In this report, we have revisited the Attentive state-space model on disease progression problem and proposed a few posterior sampling methods according to the model structure. The experimental results primarily show that the Multinomial sampling method which only estimate the gradient locally has the fastest convergence among all, but tends to recover a Markovian dynamics where the ground-truth attention is not. In contrast, Gumbel-Max sampling method requires more computational budget to get a reasonable estimate of the attention weights. The integration of them, however, is able to give fairly-reasonable estimate in an acceptable amount of computation time. Potentially this could results in a more stable and accurate unsupervised algorithm on disease progression task.

The future works for this are multiple. From the experiment perspective. It is clear that larger-scale tests are required to be done on more complex data or the application on real data, before any realistic conclusion has been made. *Re-sampling* (Bringmann & Panagiotou, 2012) is one approach that is closely-related to this report. Re-sampling refers to re-calculate the probability scores from the particles after some time steps and re-sample from the the new probability scores. This method is able to select the particles in higher probability region while discarding the particles with lower probability. During the experiment we have also test out re-sampling during inference, it does not provide an obvious improvement under such a small-scale setting, however. My thoughts are that we need to test it on longer sequence where the particle degeneracy problem get worsen (Daum & Huang, 2011). Other sampling methods such as *Bootstrap filtering* (Candy, 2007) can also be investigated along with a thorough variance evaluation experiment. One other potential topic is to apply local attention on long sequence task, so that the model does not need to re-generate the while attention weight sequence again, which can cut down the computational cost.

A. appendix

A.1. Local minima issues

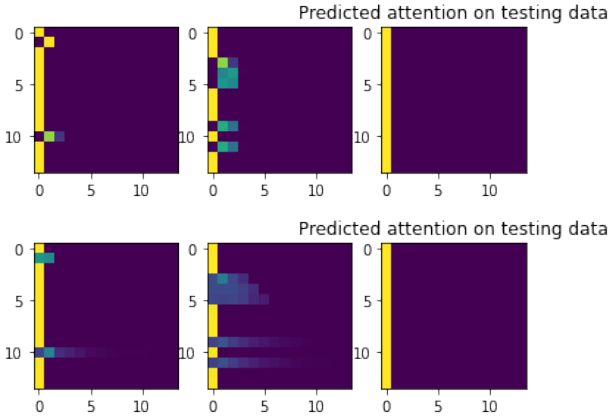

Figure 9. Predicted attention on unseen data from model with multinomial sampling methods. The top graphs show three prediction at **epoch 50**; the bottom graphs show three prediction on the same data at **epoch 100**. The true attention is the middle one in Fig (5) ($\rho = 1$)

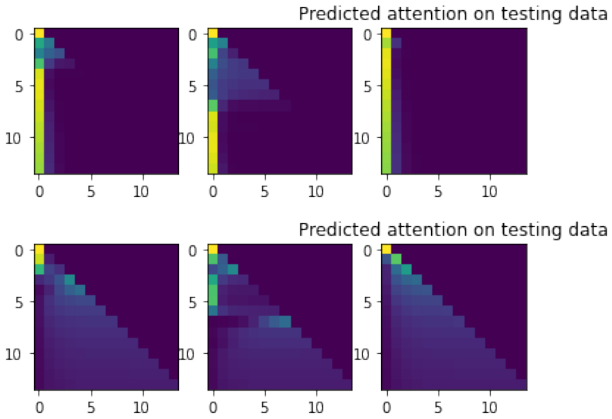

Figure 10. Predicted attention on the same unseen data as above from model with Gumbel-max trick. The top graphs show three prediction at **epoch 50**; the bottom graphs show three prediction on the same data at **epoch 100**.

A.2. Multinomial Versus Gumbel-Max

A.3. Periodic sampling method

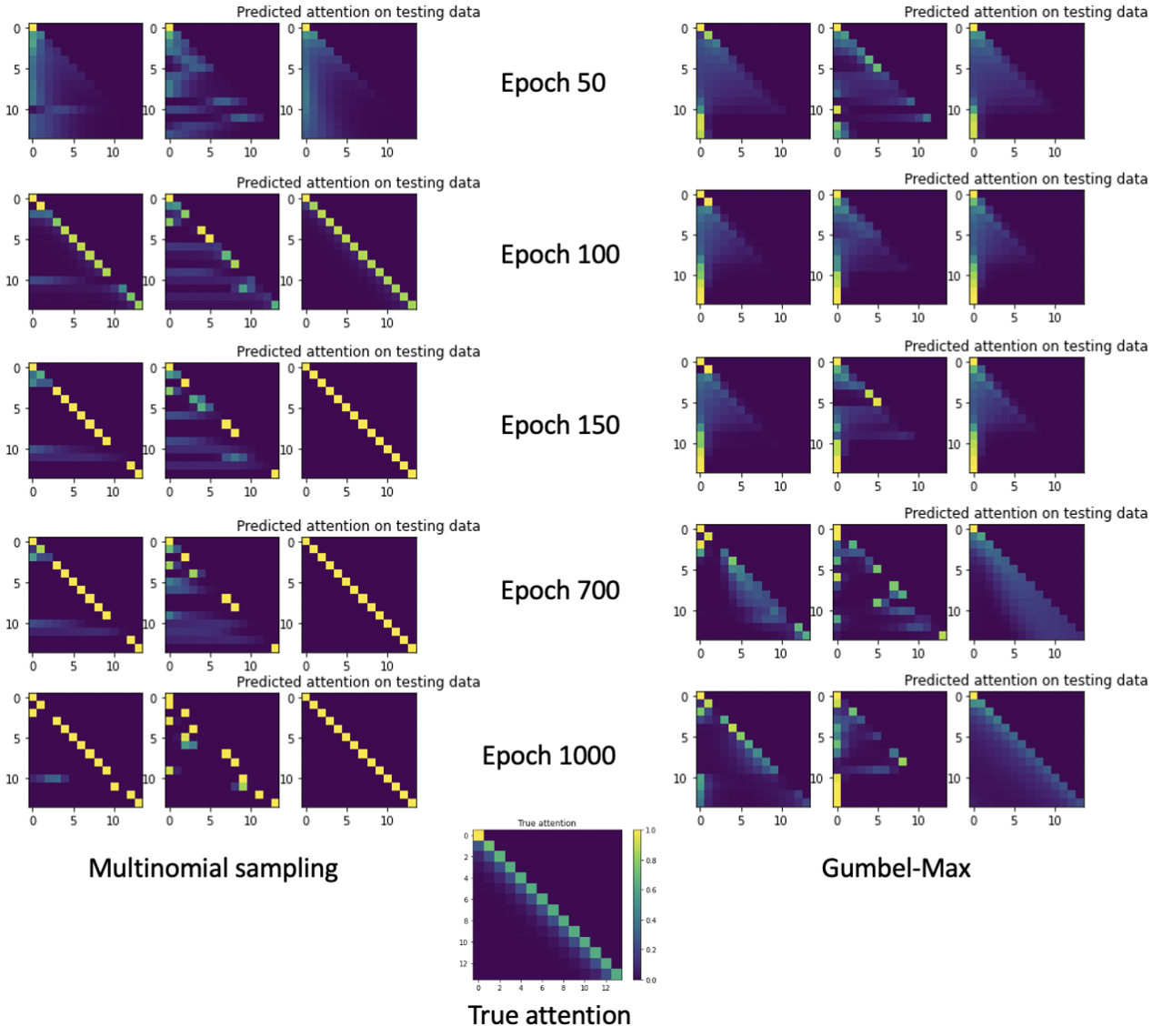

Figure 11. The change in attention weight prediction on three unseen data during training of model with **Left**: multinomial sampling methods and **Right**: Gumbel-Max sampling. The middle on at the bottom is the true attention weights. Note that the true attention weights pattern for different unseen data is the same as lone as the hyper-parameter ρ is fixed.

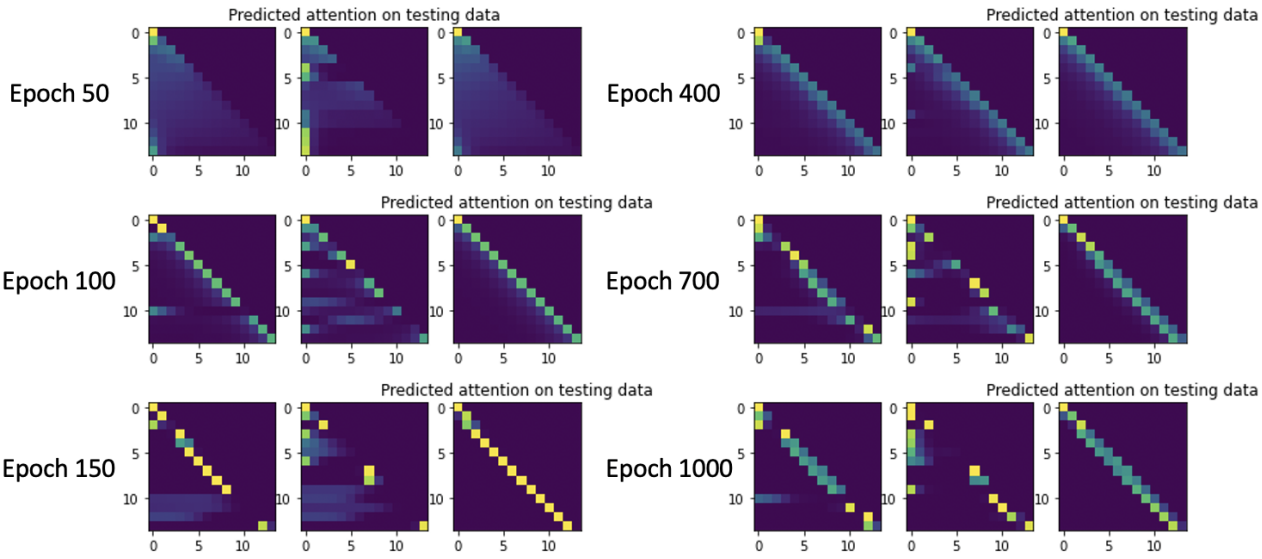

Figure 12. Change in prediction of attention weights on unseen data from model with periodic sampling method. The ground-truth attention is the same as in Fig (11).

References

Alaa, A. M. and van der Schaar, M. Attentive state-space modeling of disease progression. In *Advances in Neural Information Processing Systems*, pp. 11338–11348, 2019.

Alaa, A. M., Hu, S., and van der Schaar, M. Learning from clinical judgments: Semi-markov-modulated marked hawkes processes for risk prognosis. *arXiv preprint arXiv:1705.05267*, 2017.

Andrieu, C., De Freitas, N., Doucet, A., and Jordan, M. I. An introduction to mcmc for machine learning. *Machine learning*, 50(1-2):5–43, 2003.

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

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

Bishop, C. M. *Pattern recognition and machine learning*. springer, 2006.

Bolstad, W. M. and Curran, J. M. *Introduction to Bayesian statistics*. John Wiley & Sons, 2016.

Bringmann, K. and Panagiotou, K. Efficient sampling methods for discrete distributions. In *International Colloquium on Automata, Languages, and Programming*, pp. 133–144. Springer, 2012.

Candy, J. V. Bootstrap particle filtering. *IEEE Signal Processing Magazine*, 24(4):73–85, 2007.

Choi, E., Bahadori, M. T., Sun, J., Kulas, J., Schuetz, A., and Stewart, W. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. *Advances in neural information processing systems*, 29: 3504–3512, 2016.

Chung, J., Kastner, K., Dinh, L., Goel, K., Courville, A. C., and Bengio, Y. A recurrent latent variable model for sequential data. *Advances in neural information processing systems*, 28:2980–2988, 2015.

Daum, F. and Huang, J. Particle degeneracy: root cause and solution. In *Signal Processing, Sensor Fusion, and Target Recognition XX*, volume 8050, pp. 80500W. International Society for Optics and Photonics, 2011.

Doucet, A. and Johansen, A. M. A tutorial on particle filtering and smoothing: Fifteen years later. *Handbook of nonlinear filtering*, 12(656-704):3, 2009.

Fisher, C. K., Smith, A. M., and Walsh, J. R. Machine learning for comprehensive forecasting of alzheimer's disease progression. *Scientific reports*, 9(1):1–14, 2019.

Fraccaro, M., Sønderby, S. K., Paquet, U., and Winther, O. Sequential neural models with stochastic layers. *Advances in neural information processing systems*, 29: 2199–2207, 2016.

Gumbel, E. J. *Statistical theory of extreme values and some practical applications: a series of lectures*, volume 33. US Government Printing Office, 1954.

Hafner, D., Lillicrap, T., Ba, J., and Norouzi, M. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019a.

Hafner, D., Lillicrap, T., Fischer, I., Villegas, R., Ha, D., Lee, H., and Davidson, J. Learning latent dynamics for planning from pixels. In *International Conference on Machine Learning*, pp. 2555–2565. PMLR, 2019b.

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

Jackson, C. H., Sharples, L. D., Thompson, S. G., Duffy, S. W., and Couto, E. Multistate markov models for disease progression with classification error. *Journal of the Royal Statistical Society: Series D (The Statistician)*, 52 (2):193–209, 2003.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Kingma, D. P. and Welling, M. Stochastic gradient vb and the variational auto-encoder. In *Second International Conference on Learning Representations, ICLR*, volume 19, 2014.

Kool, W., Van Hoof, H., and Welling, M. Stochastic beams and where to find them: The gumbel-top-k trick for sampling sequences without replacement. *arXiv preprint arXiv:1903.06059*, 2019.

Liu, J. S., Chen, R., and Logvinenko, T. A theoretical framework for sequential importance sampling with resampling. In *Sequential Monte Carlo methods in practice*, pp. 225–246. Springer, 2001.

Liu, Y.-Y., Li, S., Li, F., Song, L., and Rehg, J. M. Efficient learning of continuous-time hidden markov models for disease progression. *Advances in neural information processing systems*, 28:3600–3608, 2015.

Ma, F., Chitta, R., Zhou, J., You, Q., Sun, T., and Gao, J. Dipole: Diagnosis prediction in healthcare via attention-based bidirectional recurrent neural networks. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 1903–1911, 2017.

Maddison, C. J., Tarlow, D., and Minka, T. A* sampling. *Advances in Neural Information Processing Systems*, 27: 3086–3094, 2014.

Mould, D. Models for disease progression: new approaches and uses. *Clinical Pharmacology & Therapeutics*, 92(1): 125–131, 2012.

Nguyen, M., He, T., An, L., Alexander, D. C., Feng, J., Yeo, B. T., Initiative, A. D. N., et al. Predicting alzheimer's disease progression using deep recurrent neural networks. *NeuroImage*, 222:117203, 2020.

Parmas, P., Rasmussen, C. E., Peters, J., and Doya, K. Pipps: Flexible model-based policy search robust to the curse of chaos. In *International Conference on Machine Learning*, pp. 4065–4074. PMLR, 2018.

Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. Automatic differentiation in pytorch. 2017.

Reynolds, D. A. Gaussian mixture models. *Encyclopedia of biometrics*, 741, 2009.

Smith, A. *Sequential Monte Carlo methods in practice*. Springer Science & Business Media, 2013.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. In *Advances in neural information processing systems*, pp. 5998–6008, 2017.

Wang, T., Qiu, R. G., and Yu, M. Predictive modeling of the progression of alzheimer's disease with recurrent neural networks. *Scientific reports*, 8(1):1–12, 2018.

Wang, X., Sontag, D., and Wang, F. Unsupervised learning of disease progression models. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 85–94, 2014.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.