Sequential Neural Models with Stochastic Layers

Ashwin D'Cruz, Qingyun Dou, Jonathan Gordon, Mara Graziani
Department of Engineering University of Cambridge

OBJECTIVES

The paper is concerned with the propagation of uncertainty through RNNs. The main objectives are the following:

- Combine the advantages of SSMs and RNNS.
- Model the sequential dependence of stochastic layers in temporal VAEs.

Introduction

RNNs capture non-linear dependencies in temporal data, but do not model uncertainty. They have been extended to include latent variables, but the sequential nature of these variables is not modeled [1]. SSMs explicitly model the dependence in the hidden state. However, they are difficult to train and are restricted to simple distributions. This paper unifies the two approaches, producing a RNN with sequential stochastic layers.

Stochastic Recurrent Neural Networks

The SRNN is a generative model over sequences that stacks an SSM on a RNN, and is defined as :

$$p_\theta(\boldsymbol{x}_{1:T}, \boldsymbol{z}_{1:T}, \boldsymbol{d}_{1:T} | \boldsymbol{u}_{1:T}, \boldsymbol{z}_0) = \prod_{t=1}^{T} p_{\theta_x}(\boldsymbol{x}_t|\boldsymbol{z}_t, \boldsymbol{d}_t) p_{\theta_z}(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}, \boldsymbol{d}_t) p_{\theta_d}(\boldsymbol{d}_{t-1}, \boldsymbol{u}_t) \quad (1)$$

Where the generative distributions $p_{\theta_x}, p_{\theta_z}$ are parameterized by neural networks, and

$$p_{\theta_d}(\boldsymbol{d}_t|\boldsymbol{d}_{t-1}, \boldsymbol{u}_t) = \delta(\boldsymbol{d}_t - \tilde{\boldsymbol{d}}_t) \quad (2)$$

is a deterministic function parameterised by a GRU.

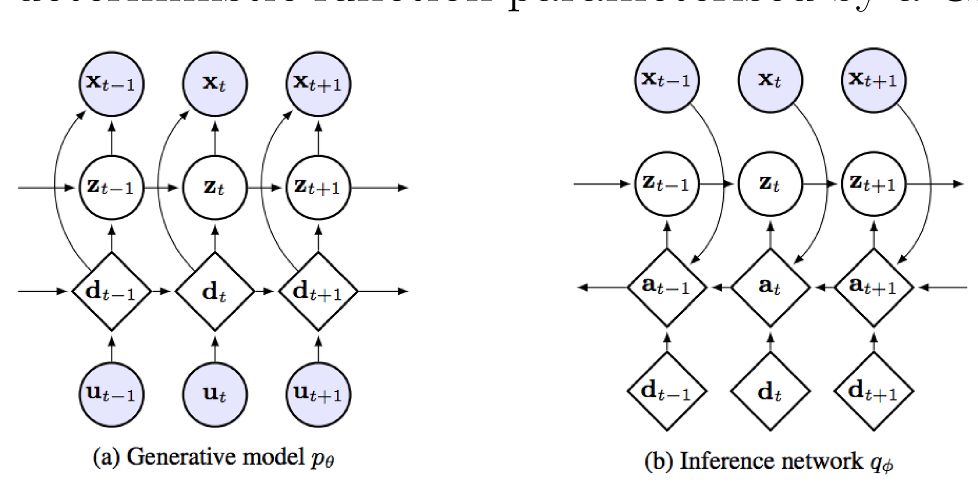


Figure: Graphical model depiction of (left) generative model and (right) inference network of an SRNN.

Variational Inference for SRNNs

Variational Inference (VI) is employed for training the SRNN. We introduce the approximate distribution q_ϕ over the latent variables $\boldsymbol{z}$, and note that $\boldsymbol{d}$ is deterministic:

$$q_\phi(\boldsymbol{z}_{1:T}, \boldsymbol{d}_{1:T} | \boldsymbol{x}_{1:T}, \boldsymbol{u}_{1:T}) = q_\phi(\boldsymbol{z}_{1:T} | \tilde{\boldsymbol{d}}_{1:T}, \boldsymbol{x}_{1:T}, \boldsymbol{z}_0) \quad (3)$$

Where the inference network is also parameterized by a neural network. The evidence lower-bound (ELBO) for a complete sequence is then:

$$\mathcal{F}_i(\theta, \phi) = \mathbb{E}_{q_\phi}\left[\log p_\theta(\boldsymbol{x}_{1:T}|\boldsymbol{z}_{1:T}, \tilde{\boldsymbol{d}}_{1:T})\right] - \mathrm{KL}\left(q_\phi(\boldsymbol{z}_{1:T}|\tilde{\boldsymbol{d}}_{1:T}, \boldsymbol{x}_{1:T}) \| p_\theta(\boldsymbol{z}_{1:T}|\tilde{\boldsymbol{d}}_{1:T})\right) \quad (4)$$

We can encode the temporal dependence into the inference network as well:

$$q_\phi(\boldsymbol{z}_{1:T}|\tilde{\boldsymbol{d}}_{1:T}, \boldsymbol{x}_{1:T}) = \prod^{t} q_\phi(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}, \boldsymbol{a}_t) \quad (5)$$

where $\boldsymbol{a}_t = g_{\phi_a}(a_{t+1}, \boldsymbol{d}_t, \boldsymbol{x}_t)$ is parameterized by a backwards-in-time GRU. The generative and inference networks now both factorize over time steps. Expressing the ELBO as a sum over time steps:

$$\mathcal{F}_i(\theta, \phi) = \sum_t \mathbb{E}_{q_{\phi(z_{t-1})}} \mathbb{E}_{q_{\phi}(z_t|z_{t-1})}\left[\log p_\theta(\boldsymbol{x}_t|\boldsymbol{z}_t, \tilde{\boldsymbol{d}}_t)\right] - \mathrm{KL}\left(q_\phi(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}, \tilde{\boldsymbol{d}}_{1:T}, \boldsymbol{x}_{1:T}) \| p_\theta(\boldsymbol{z}_t|\boldsymbol{z}_{t-1}, \tilde{\boldsymbol{d}}_t)\right) \quad (6)$$

where we can express the marginal approximate distribution recursively as:

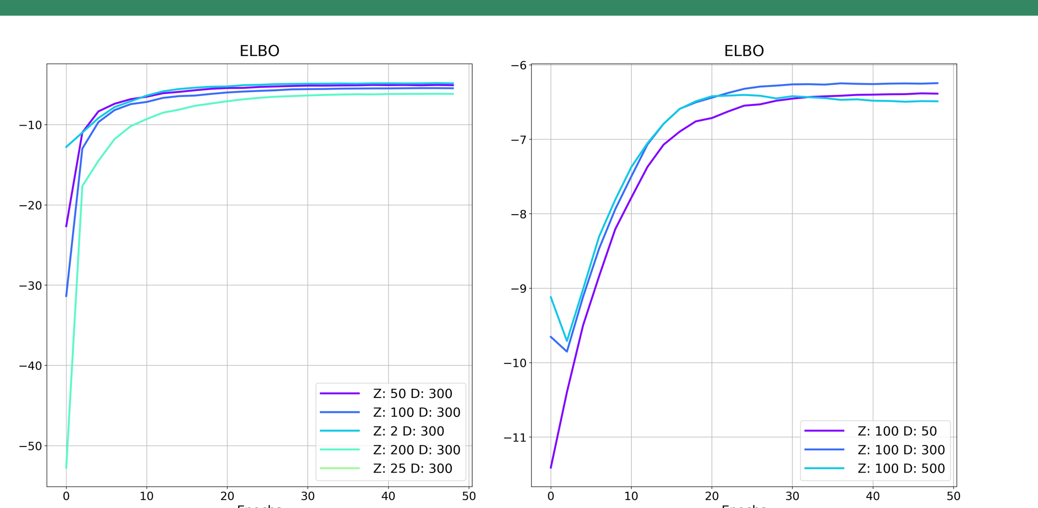
$$q_\phi(\boldsymbol{z}_{t-1}) = \mathbb{E}_{q_{\phi(z_{t-2})}}\left[q_\phi(\boldsymbol{z}_{t-1}|\boldsymbol{z}_{t-2}, \tilde{\boldsymbol{d}}_{t-1:T}, \boldsymbol{x}_{t-1:T})\right] \quad (7)$$

The AEVB algorithm and reparameterization trick [2] can then be applied for joint learning of the model parameters θ and inference network parameters ϕ.

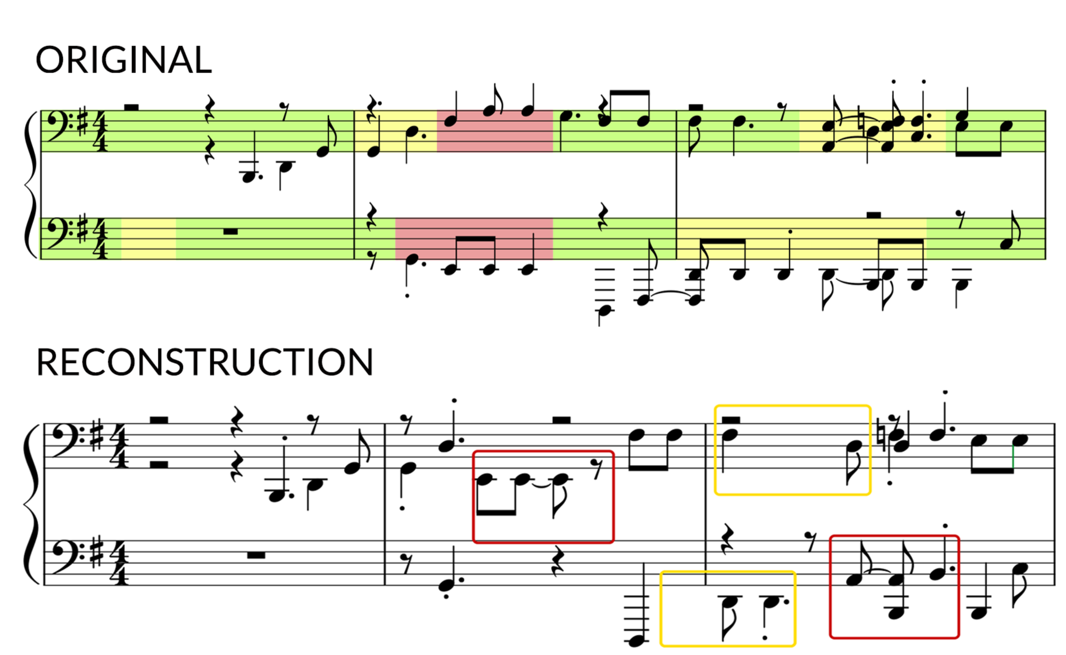
Experimental Results

Z	MUSE	JSB	PIANO	NOTTS
2	-6.4784	-4.8659	-8.3599	-3.3252
10	-6.2908	-4.8187	-8.2811	-3.1740
25	-6.3001	-4.8999	-8.2323	-3.1382
50	-6.2638	-5.0851	-8.1985	-3.1101
100	-6.2445	-5.4700	-8.2100	-3.0879

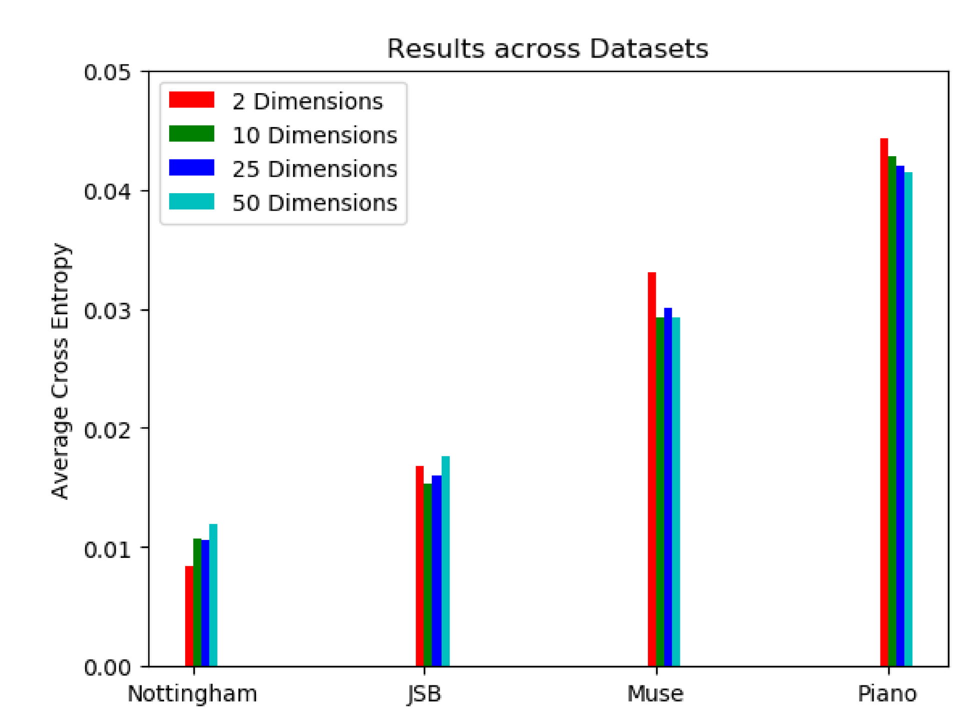
(a) ELBO values across architectures



(b) ELBO curves for varying dimensions of z (left) and d (right)



(c) Reconstruction of midi files



(d) Cross-entropy of different datasets and dimensions of z

Experimental Setup

We trained the SRNN on polyphonic music of varying complexity. We then used a separated testing set to measure the ELBO of different SRNN architectures, namely for $z \in \mathcal{R}^{(2,10,25,50,100,200)}$ and for $d \in \mathcal{R}^{(50,300,500)}$.

Figure 1d shows the average cross entropy for the held-out test data as a function of the different datasets and stochastic variable dimension. These values correlate strongly with the average log likelihoods obtained.

Discussion

- SRNN propagates uncertainty through time, producing state-of-the-art results in modeling polyphonic music. We hypothesize that for music generation this may be especially beneficial.
- Comparing performance across models is difficult due to the intractability of the log-likelihoods.

Future Work

- Implement a standard RNN and a VRNN [1], perform the same experiments and compare results with SRNN.
- Use SRNN as a predictor for music generation.
- Combine SRNN and reinforcement learning [3], to improve performance for music generation.

References

[1] 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.

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

[3] Natasha Jaques, Shixiang Gu, Richard E Turner, and Douglas Eck. Tuning recurrent neural networks with reinforcement learning. *arXiv preprint arXiv:1611.02796*, 2016.