UNIVERSITY OF TORONTO | Engineering

Development of digital twin via generative neural networks

Kuilin Chen

June 12, 2019

Outline

- Review of previous work
- Review of generative neural network
- Dynamic generative neural network
- Seq2Seq generation
- Future work

Previous work

- Completed the literature review of digital twin and pointed out research opportunities in current digital twin research
- Current research focus is to model the relationship between set points and actual temperature inside combustion system
- ARX, LSTM and GRU models have been developed to predict one-step-ahead temperature based on past set points and temperature
- Try to develop new generative models for time-series

Generative neural network

- We want to learn a probability distribution over high-dimensional x (e.g. picture and long time-series)
- $p_{\mathcal{D}}(x)$ is the true distribution, and $p_\theta(x)$ is the modelled distribution
- Direct optimization over p_θ to approximate $p_{\mathcal{D}}$ is very challenging (e.g. high-dimensionality, existence of $p_{\mathcal{D}}$...)
- We define a low-dimensional z with a fixed prior distribution $p(z)$, and pass z through g_θ (deep neural network): $\mathcal{Z} \to \mathcal{X}$
- High-dimensional x can be generated without explicitly knowing high-dimensional density

Generative adversarial networks (GAN)

Adversarial training

$$\min_G \max_D V(D, G) = \mathbb{E}_{x \sim p_{\mathcal{D}}(x)}[\log D(x)] + \mathbb{E}_{z \sim p_z}(z)[\log(1 - D(G(z)))]$$

- G is a generator, D is a discriminator
- Train D to discriminate the real and generated samples
- Simultaneously train G to generate samples close to real samples
- $p(x)$ is not explicitly modeled in GAN
- Evaluation of generated samples from GAN can be done by human subjectively

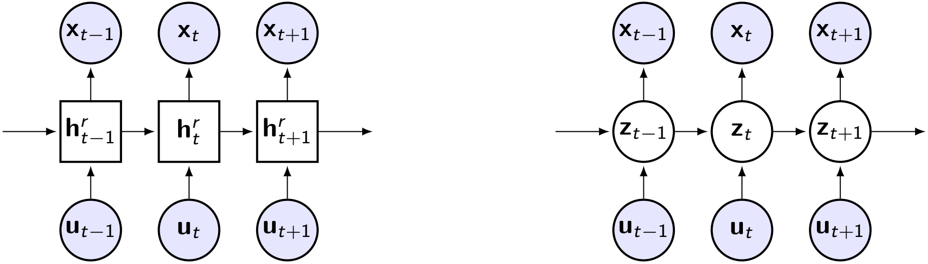
Variational autoencoder (VAE)

Evidence lower bound (ELBO)

$$\mathcal{L}(x) = \underbrace{-D_{\mathrm{KL}}\left(q_\phi(z|x) \| p(z)\right)}_{\text{regularization}} + \underbrace{\mathbb{E}_{q_\phi(z|x)}\left[\log p_\theta(x|z)\right]}_{\text{log-likelihood}}$$

- $q_\phi(z|x)$ is a probabilistic encoder, $p_\theta(x|z)$ is a probabilistic decoder
- Maximize $\mathcal{L}$ by varying ϕ and θ to train the generative model
- ELBO or log-likelihood could be maximized by overfitting x (memorize the training sample)
- Good ELBO or log-likelihood values does not imply good inference
- ELBO or log-likelihood should not be used to evaluate generated samples

RNN and SSM



(a) RNN

(b) SSM

Figure: Graphical models to generate $\mathbf{x}_{1:T}$ with a recurrent neural network (RNN) and a state space model (SSM). Rectangle-shaped units are used for deterministic states, while circles are used for stochastic ones.

$$\mathbf{h}_t = f\left(\mathbf{h}_{t-1}, \mathbf{u}_t\right)$$
$$\mathbf{x}_t = g\left(\mathbf{h}_t\right)$$

$$\mathbf{z}_t \sim p_{\theta_z}\left(\mathbf{z}_t \middle| \mathbf{u}_t, \mathbf{z}_{t-1}\right)$$
$$\mathbf{x}_t \sim p_{\theta_x}\left(\mathbf{x}_t \middle| \mathbf{z}_t\right)$$

Combination of RNN and SSM

- RNN and SSM have been combined to develop generative models in some papers
- However, their models are limited to categorical input and output (e.g. rotated image generate, new drug development)
- A new generative model is proposed based on combination of bi-directional RNN and SSM
- The objective function and output decoding distribution are re-designed to make it suitable for time-series generation

Variational inference for dynamic generative model

ELBO

$$
\begin{aligned}
&\log p_\theta(\mathbf{x}|\mathbf{u}) - \mathcal{D}_{KL}\left(q_\phi(\mathbf{z}|\mathbf{x},\mathbf{u}) \| p_\theta(\mathbf{z}|\mathbf{x},\mathbf{u})\right) \\
=& \underbrace{\mathbb{E}_{\mathbf{z}\sim q_\phi}\left[\log p_\theta(\mathbf{x}|\mathbf{z},\mathbf{u})\right]}_{\text{log-likelihood}} - \underbrace{\mathcal{D}_{KL}\left[q_\phi(\mathbf{z}|\mathbf{x},\mathbf{u}) \| p_\theta(\mathbf{z}|\mathbf{u})\right]}_{\text{regularization}} \\
=& \mathcal{L}(\theta,\phi)
\end{aligned}
$$

Algorithm

Algorithm 1 Dynamic generative model

Initialize parameters θ, ϕ
repeat
  Get random minibatch datapoints $\mathbf{x}, \mathbf{u}$
  Get Monte Carlo samples $\mathbf{z}^*$ from distribution $q_\phi(\mathbf{z}|\mathbf{x}, \mathbf{u})$
  Evaluate $\mathbb{E}_{\mathbf{z}\sim q_\phi}[\log p_\theta(\mathbf{x}|\mathbf{z}, \mathbf{u})]$ using $\mathbf{z}^*$
  Update parameters using gradients $\nabla_{\theta,\phi}\mathcal{L}$ (e.g. SGD)
until convergence of parameters θ, ϕ
return θ, ϕ

Thank You!
Questions?