Neural ODE & ODE^2VAE

Yogesh Verma
Doctoral Candidate
Aalto University

Why ODE?

- Discrete sequence of transformations over hidden state (ResNets, RNN)

$$h(t+1) = h(t) + f(h(t), \theta) \tag{1}$$

Why ODE?

- Discrete sequence of transformations over hidden state (ResNets, RNN)

$$h(t+1) = h(t) + f(h(t), \theta) \tag{1}$$

- Parameterize the continuous dynamics of hidden units as an ODE with f describing a neural network

$$\frac{dh(t)}{dt} = f(h(t), t, \theta) \tag{2}$$

Why ODE?

- Discrete sequence of transformations over hidden state (ResNets, RNN)

$$h(t+1) = h(t) + f(h(t), \theta) \tag{1}$$

- Parameterize the continuous dynamics of hidden units as an ODE with f describing a neural network

$$\frac{dh(t)}{dt} = f(h(t), t, \theta) \tag{2}$$

- Solved by a black-box differential equation solver with h(0) as first and h(T) as output layer

Why ODE?

- Discrete sequence of transformations over hidden state (ResNets, RNN)

$$h(t+1) = h(t) + f(h(t), \theta) \tag{1}$$

- Parameterize the continuous dynamics of hidden units as an ODE with f describing a neural network

$$\frac{dh(t)}{dt} = f(h(t), t, \theta) \tag{2}$$

- Solved by a black-box differential equation solver with h(0) as first and h(T) as output layer

Benefits

- Memory Efficiency
- Continuous time-series model
- Adaptive computation

How to optimize?

- Compute gradient using adjoint sensitivity method (ASM) (Pontryagin et al., 1962)

How to optimize?

- Compute gradient using adjoint sensitivity method (ASM) (Pontryagin et al., 1962)
- **ASM**: Computes gradients by solving a second, augmented ODE backwards in time

ASM

Consider

$$L(z(t_1)) = L(ODESolve(z(t_0, f, t_0, t_1, \theta))) \tag{3}$$

Need: Gradients w.r.t θ

ASM

Consider

$$L(z(t_1)) = L(ODESolve(z(t_0, f, t_0, t_1, \theta))) \tag{3}$$

Need: Gradients w.r.t θ

$$a(t) = \frac{\partial L}{\partial z(t)} \text{ [} Adjoint \text{]} \tag{4a}$$

$$\frac{da(t)}{dt} = -a(t)\frac{\partial f}{\partial z(t)} \tag{4b}$$

$$\frac{dL}{d\theta} = -\int_{t_1}^{t_0} a(t)\frac{\partial f}{\partial \theta}\, dt \tag{4c}$$

ODE for supervised learning

- Comparison with:
  - ResNet which down samples the input twice then applies 6 standard residual blocks
  - RK-Net where gradients are back-propagated directly through a Runge-Kutta integrator

ODE for supervised learning

- Comparison with:
  - ResNet which down samples the input twice then applies 6 standard residual blocks
  - RK-Net where gradients are back-propagated directly through a Runge-Kutta integrator

	Test Error	Param	Memory	Time
1-Layer MLP	1.60%	0.24 M	-	-
ResNet	0.41%	0.60 M	O(L)	O(L)
RK-Net	0.47%	0.22 M	$O(L*)$	$O(L*)$
ODE-Net	0.42%	0.22 M	O(1)	$O(L*)$

Continuous Normalizing flow

Normalizing flow:

$$log(p(z_1)) = log(p(z_0)) - log|det\frac{\partial f}{\partial z_0}| \tag{5}$$

Bottleneck: Computing the determinant

Continuous Normalizing flow

Normalizing flow:

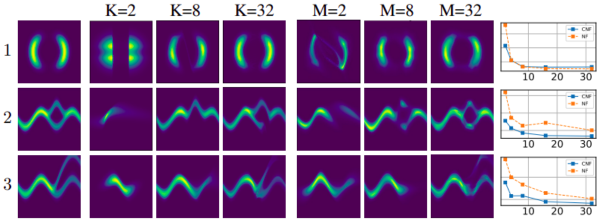
$$log(p(z_1)) = log(p(z_0)) - log|det\frac{\partial f}{\partial z_0}| \tag{5}$$

Bottleneck: Computing the determinant
Moving to a continuous transformation, where an ODE describe the transformation in z(t):

$$\frac{\partial log(p(z(t)))}{\partial t} = -tr(\frac{\partial f}{\partial z(t)}) \tag{6}$$

Experiments

- Compare the continuous and discrete planar flows at learning to sample from a known distribution and shows that a planar CNF with M hidden units can be at least as expressive as a NF with K = M layers.
- Train CNF for 10,000 iterations using Adam and NF for 500,000 iterations using RMSprop. We minimize $KL(q(x)||p(x))$ as the loss function where q is the flow model and the target density $p()$



(a) Target (b) NF (c) CNF (d) Loss vs. K/M

Time series modeling

- Represent each time series by a latent trajectory
- Given a local initial state z_{t_0}, ODE solver produces $z_{t_1}, ...z_{t_n}$ at each observation time

$$z_{t_0} \sim p(z_{t_0}) \tag{7a}$$

$$z_{t_1}, ...z_{t_n} = ODESolve(z_{t_0}, f, \theta_f, t_0, ..., t_n) \tag{7b}$$

$$each\ x_{t_i} \sim p(x|z_{t_i}, \theta_x) \tag{7c}$$

Time series modeling

- Represent each time series by a latent trajectory
- Given a local initial state z_{t_0}, ODE solver produces $z_{t_1}, ...z_{t_n}$ at each observation time

$$
\begin{aligned}
z_{t_0} &\sim p(z_{t_0}) && \text{(7a)} \\
z_{t_1}, ...z_{t_n} &= ODESolve(z_{t_0}, f, \theta_f, t_0, ..., t_n) && \text{(7b)} \\
each\ x_{t_i} &\sim p(x|z_{t_i}, \theta_x) && \text{(7c)}
\end{aligned}
$$

Training:

- Encode the data points sequentially using RNN to output $q_\phi(z_0|x_1, x_2, ..., x_n)$ from which local initial state can be sampled
- Using ODEs as a generative model allows us to make predictions for arbitrary time points with maximizing ELBO

ODE + VAE ?

- Can we model the latent dynamic state with an ODE as previously shown?

ODE + VAE ?

- Can we model the latent dynamic state with an ODE as previously shown?
- Learning low-rank latent representations of possibly high-dimensional sequential data trajectories

ODE + VAE ?

- Can we model the latent dynamic state with an ODE as previously shown?
- Learning low-rank latent representations of possibly high-dimensional sequential data trajectories
- Extend VAEs for sequential data with a latent space governed by a continuous-time probabilistic ODE

Bayesian 2^{nd} order ODE

- First-order ODEs are incapable of modelling high-order dynamics.

Bayesian 2^{nd} order ODE

- First-order ODEs are incapable of modelling high-order dynamics.

$$\ddot{z}_t = \frac{d^2 f(z_t)}{d^2 t} = f_W(z_t, \dot{z}_t) \tag{8}$$

The above equation can be decomposed as:

$$\dot{s}_t = v_t \tag{9a}$$
$$\dot{v}_t = f_W(s_t, v_t) \tag{9b}$$

$$\begin{bmatrix} s_t \\ v_t \end{bmatrix} = \begin{bmatrix} s_0 \\ v_0 \end{bmatrix} + \int_0^T \begin{bmatrix} v_t \\ f_W(s_t, v_t) \end{bmatrix} \tag{10}$$

where $z_t = (s_t, v_t)$, s_t is state position and v_t is state velocity. The $f_W(s_t, v_t)$ is governed by a BNN.

ODE^2VAE Model

- VAE formalism $+ \ 2^{nd}$ order Bayesian neural ODE model in the latent space to model the data dynamics
- Infer continuous-time latent position and velocity trajectories while matching data as well

ODE^2VAE Model

- VAE formalism + 2^{nd} order Bayesian neural ODE model in the latent space to model the data dynamics
- Infer continuous-time latent position and velocity trajectories while matching data as well

Consider a generative model defined as:

$$
\begin{aligned}
s_0 &\sim p(s_0) && \text{(11a)}\\
v_0 &\sim p(v_0) && \text{(11b)}\\
s_t &= s_0 + \int_0^t v_t \, dt && \text{(11c)}\\
v_t &= v_0 + \int_0^t f(s_t, v_t) \, dt && \text{(11d)}\\
x_i &\sim p(x_i | s_i) && \text{(11e)}
\end{aligned}
$$

ODE^2VAE Model

- Position encoder maps the first item (x_0) of a high-dimensional data sequence into a distribution of the initial position s_0 characterized by μ_s, σ_s
- Velocity encoder maps the first m items of a high-dimensional data sequence ($x_{0:m}$) into a distribution of the initial position v_0 characterized by μ_v, σ_v
- Probabilistic latent dynamics are implemented by a second order ODE model f parameterised by a Bayesian deep neural network

Variational Inference

Variational approximation for unobserved quantities:

$$q(\mathcal{W}, z_{0:N}|x_{0:N}) = q(\mathcal{W})q_{enc}(z_0|x_{0:N})q_{ode}(z_{1:N}|x_{0:N}, z_0, W) \tag{12}$$

where

$$q(\mathcal{W}) = \mathcal{N}(\mathcal{W}|m, s\mathcal{I}) \tag{13}$$

$$q_{enc}(z_0|x_{0:N}) = \mathcal{N}(\begin{bmatrix} \mu_s(x_0) \\ \mu_v(x_{0:m}) \end{bmatrix}, \begin{bmatrix} diag(\sigma_s(x_0)) & 0 \\ 0 & diag(\sigma_v(x_{0:m})) \end{bmatrix}) \tag{14}$$

$$\frac{\partial log(q_{ode}(z(t)|\mathcal{W}))}{\partial t} = -tr(\frac{\partial f_{\mathcal{W}}}{\partial v_t}) \tag{15}$$

Variational Inference

ELBO

$$log(p(X)) \geq ODE\ \ regularization\ +\ VAE\ \ loss\ +\ dynamic\ \ loss \tag{16}$$

$$ODE\ \ regularization = -KL[q(\mathcal{W})||p(\mathcal{W})] \tag{17}$$

$$VAE\ \ loss = \mathbb{E}_{q_{enc}(z_0|X)}[-log\frac{q_{enc}(z_0|X)}{p(z_0)} + log(p(x_0|z_0))] \tag{18}$$

$$dynamic\ \ loss = \sum_{i=1}^{N} \mathbb{E}_{q_{ode}(\mathcal{W},z_i|X,z_0)}[-log\frac{q_{ode}(z_i|\mathcal{W},X)}{p(z_i)} + log(p(x_i|z_i))] \tag{19}$$

Penalized Variational loss

- Optimizing the ELBO objective does not necessarily result in accurate inference
- To counteract the imbalance between the KL term and reconstruction likelihood : Weight the regularization term by β where,

$$\beta = \frac{|q|}{|\mathcal{W}|} \tag{20}$$

Penalized Variational loss

- Optimizing the ELBO objective does not necessarily result in accurate inference
- To counteract the imbalance between the KL term and reconstruction likelihood : Weight the regularization term by β where,

$$\beta = \frac{|q|}{|\mathcal{W}|} \tag{20}$$

- In long input sequences, dynamic loss term can easily dominate VAE loss, which may cause the encoders to underfit
- Propose to minimize the distance between the encoder distribution and the distribution induced by the ODE flow

$ODE^2VAE - KL$

Alternative penalized target function:

$$\mathcal{L}_{ODE^2VAE} = -\beta KL[q(\mathcal{W})||p(\mathcal{W})] - \gamma KL[q_{ode}(Z|X)||q_{enc}(Z|\mathcal{W}, X)]$$

$$+ \mathrm{E}_{q(\mathcal{W},Z|X)}[-log\frac{q(Z|\mathcal{W},X)}{p(Z)} + log(p(X|\mathcal{W}, Z)](21)$$

- Choose the constant γ by cross-validation

- CMU walking data: First two-third of each sequence is reserved for training and validation, and the rest is used for testing

Model	Test error: Mocap-1	Test error: Mocap-2	Reference
GPDM	126.46 ± 34	N/A	Wang et al. (2008)
VGPLVM	142.18 ± 1.92	N/A	Damianou et al. (2011)
DTSBN-S	80.21 ± 0.04	34.86 ± 0.02	Gan et al. (2015)
NPODE	45.74	22.96	Heinonen et al. (2018)
NEURALODE	87.23 ± 0.02	22.49 ± 0.88	Chen et al. (2018b)
ODE^{2}VAE	93.07 ± 0.72	10.06 ± 1.4	current work
ODE^{2}VAE-KL	$\mathbf{15.99 \pm 4.16}$	$\mathbf{8.09 \pm 1.95}$	current work

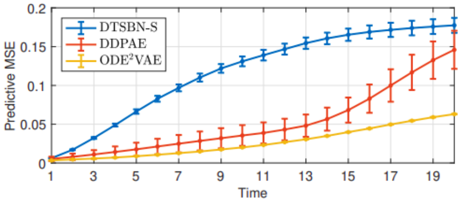
Experiments and Results

- CMU walking data: First two-third of each sequence is reserved for training and validation, and the rest is used for testing
- Rotating MNIST: Constructing a dataset by rotating the images of handwritten "3" digits with 16 rotation angles. Moreover, four rotation angles are randomly removed from each rotation sequence to introduce non-uniform sequences and missing data.

Model	Test Error
GPPVAE-DIS$^{\diamond}$	0.0309 ± 0.00002
GPPVAE-JOINT$^{\diamond}$	0.0288 ± 0.00005
ODE^{2}VAE	0.0194 ± 0.00006
ODE^{2}VAE-KL	$\mathbf{0.0188 \pm 0.0003}$

Experiments and Results

- CMU walking data: First two-third of each sequence is reserved for training and validation, and the rest is used for testing
- Rotating MNIST: Constructing a dataset by rotating the images of handwritten "3" digits with 16 rotation angles. Moreover, four rotation angles are randomly removed from each rotation sequence to introduce non-uniform sequences and missing data.
- Bouncing Balls: Generated a training set of 10000 sequences of length 20 frames and a test set of 500 sequences where each frame is 32x32x1



Connection to our work!

Connection to our work!

- Molecules indeed follow continuous transformations, can be modelled by ODEs for dynamics, conformer generation or property prediction

Connection to our work!

- Molecules indeed follow continuous transformations, can be modelled by ODEs for dynamics, conformer generation or property prediction
- A spectral density field can be formed over graph nodes, where nodes can act as a measurement point of spectral density creating kernel at each node with entanglement with other nearby nodes. The evolution of latent space of spectral density can be done via neural ODEs.

Connection to our work!

- Molecules indeed follow continuous transformations, can be modelled by ODEs for dynamics, conformer generation or property prediction
- A spectral density field can be formed over graph nodes, where nodes can act as a measurement point of spectral density creating kernel at each node with entanglement with other nearby nodes. The evolution of latent space of spectral density can be done via neural ODEs.
- One go place for all stuff: `https://yogeshverma1998.github.io/`

Repository

./ yogeshverma1998.github.io

View on GitHub

An evolving trajectory in drug design

The purpose of this note is to act as a one place repository for all the literature read, presentations and ideas which have been formulated reside here and act as a one go place. The literature read has been segregated in different domains ranging from dynamics to representing molecule as a 3d object. Paper which are being currently read and analyzed are marked with . The slides presented by me in subsequent meetings are dated and can be found at the end of this page.

Research Ideas (:lock:)

>> Dynamics

- >> ODE^{2}VAE
- >> Neural ODEs
- >> Neural Flows
- >> FFJORD
- >> Review of Normalizing flows

>> Conformer generation and molecule as 3d object

- >> ODE^{2}VAE

>> Slides Presented:

- >> 26/01/2022 [Regular meeting]

THANK YOU