Updates and Progress

Yogesh Verma
Doctoral Candidate
Aalto University

Updates

- Paper read: 36/1280
  - Energy-Inspired Molecular Conformation Optimization
  - Chemical-Reaction-Aware Molecule Representation Learning
  - Denoising Probabilistic Diffusion models
  - Estimation of Non-Normalized Statistical Models by Score Matching
  - Reliable Categorical Variational Inference with Mixture of Discrete Normalizing Flows [Reading]
  - Mixture of Discrete Normalizing Flows for Variational Inference [Reading]

Updates

- Paper read: 36/1280
  - Energy-Inspired Molecular Conformation Optimization
  - Chemical-Reaction-Aware Molecule Representation Learning
  - Denoising Probabilistic Diffusion models
  - Estimation of Non-Normalized Statistical Models by Score Matching
  - Reliable Categorical Variational Inference with Mixture of Discrete Normalizing Flows [Reading]
  - Mixture of Discrete Normalizing Flows for Variational Inference [Reading]
- CNF for generating valid molecules, coding, debugging.....
- Reversible SDEs for graphs
- Ideas in Stack: Molecular surface by 3D Zerneike Descriptors

Modular Flowing Graphs for MG

- Aim: Learn realistic molecular distributions and generating valid molecules
- We represent atom configurations by modelling local neighborhoods with coupled PDE CNFs in graph domain, Attack validity by accurate local densities

Modular Flowing Graphs for MG

- Aim: Learn realistic molecular distributions and generating valid molecules
- We represent atom configurations by modelling local neighborhoods with coupled PDE CNFs in graph domain, Attack validity by accurate local densities
- Given a molecule with a graph representation (connectivity C) $\mathcal{G} = (V, E)$, one can define a probability distribution at each node over the atomic vocabulary (Vocabulary of all the unique atoms spanning the whole molecule data-set) conditional over each node(atom) neighbours.

$$P(X) = \Pi_i^{N(V)} p(x_i | N_{J(i)}) \tag{1}$$

- Building on CNFs, we present flows on graphs specially applied in the molecular regime where we model the continuous time dynamics of random variables on graphs with respect to some conditionals over connectivity of the graph applied to graph structured data

Modular Flowing Graphs for MG

- Given a set of vertices $\mathbf{X}$ for a graph (molecule), the goal is to learn the joint distribution $P(X)$ given by Eq.1 of the set of nodes.
- For continuous time dynamics of the set of variables X, by following Eq. 3,4 we formulate an ODE system as follows

$$\begin{bmatrix} \dot{\mathbf{log}}(\mathbf{p}(\mathbf{x_1(t)}|\mathbf{N_{J(1)}})) \\ \dot{\mathbf{log}}(\mathbf{p}(\mathbf{x_2(t)}|\mathbf{N_{J(2)}})) \\ \vdots \\ \dot{\mathbf{log}}(\mathbf{p}(\mathbf{x_n(t)}|\mathbf{N_{J(n)}})) \end{bmatrix} = \begin{bmatrix} -Tr(\frac{\partial \mathbf{f(X(t),N_{J(1)}})}{\partial \mathbf{x_1(t)}}) \\ -Tr(\frac{\partial \mathbf{f(X(t),N_{J(2)}})}{\partial \mathbf{x_2(t)}}) \\ \vdots \\ -Tr(\frac{\partial \mathbf{f(X(t),N_{J(n)}})}{\partial \mathbf{x_n(t)}}) \end{bmatrix} \tag{2}$$

where each $\mathbf{x_i(t)}$ is i^{th} node, $\mathbf{N_{J(i)}}$ its neighbours and $\mathbf{x_i} \in \mathbf{X}$.

f and Optimization

- Options for f
  - NN

$$f = \text{NN}(X(t), N_{J(n)}) \tag{3}$$

  - RBF+NN: Transform the neighborhood into a distribution ($\phi_i(\mathbf{x}_j)$), and then input the distribution into a NN

f and Optimization

- Options for f
  - NN
  $$f = \text{NN}(X(t), N_{J(n)}) \tag{3}$$
  - RBF+NN: Transform the neighborhood into a distribution ($\phi_i(\mathbf{x}_j)$), and then input the distribution into a NN
- Minimize the $D_{KL}\ [p(x(t)|N_J, \theta)\ ||\ p(x^*|N_J)]$ or maximize likelihood (defined by Bernoulli which gives cross-entropy) to fit the flow based model, which can be formally written as

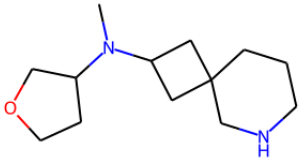
$$\mathcal{L}(\theta) = D_{KL}\ [p(x(t)|N_J, \theta)\ ||\ p(x^*|N_J)] \tag{4}$$
$$= \mathbb{E}_{p(x(t)|N_J,\theta)}\ [\log(p(x(t)|N_J, \theta)) - \log(p(x^*|N_J)] \tag{5}$$

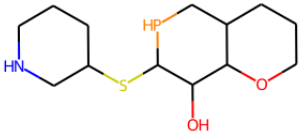
Some Preliminary checks and results

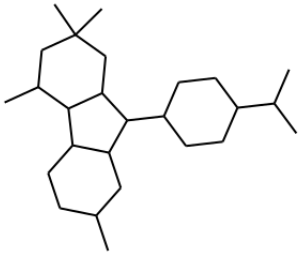
Some Preliminary checks and results

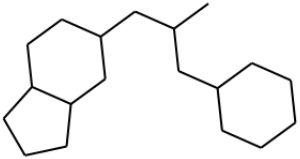
Model	Reconstruction	Validity	Novelty
NN	**100**	**94**$\sim$**98**%	to do
RBF+NN	100	to do	to do
MoFloW	100	50.3%	100%
MRNN	100	65.3%	99.8%
GraphDF	100	89.03%	99.8%

Sample Molecule









Reversible SDE on Graphs

SDE

An Ito SDE can be written as:

$$d\mathbf{X}_t = \mathbf{f}_t(\mathbf{X}_t)dt + \mathbf{g}_t(\mathbf{X}_t)d\mathbf{w} \tag{6}$$

where $\mathbf{f}_t$ is the drift coefficient, $\mathbf{g}_t$ is diffusion coefficient and $\mathbf{w}$ is standard weiner process.

SDE

An Ito SDE can be written as:

$$d\mathbf{X}_t = \mathbf{f}_t(\mathbf{X}_t)dt + \mathbf{g}_t(\mathbf{X}_t)d\mathbf{w} \tag{6}$$

where $\mathbf{f}_t$ is the drift coefficient, $\mathbf{g}_t$ is diffusion coefficient and $\mathbf{w}$ is standard weiner process. The reverse-time SDE for above can be written as (Ref)

$$d\mathbf{X}_t = [\mathbf{f}_t(\mathbf{X}_t) - \mathbf{g}_t^2 \nabla_{\mathbf{X}_t} \log p_t(\mathbf{X}_t)]d\tilde{t} + \mathbf{g}_t(\mathbf{X}_t)d\tilde{\mathbf{w}} \tag{7}$$

where $\tilde{w}$ is reverse-time standard wiener process and $d\tilde{t}$ is an infinitesimal negative time step.

SDE on Graphs

A graph $\mathbf{G}$ can represented by $\mathbf{G} = (\mathbf{X}, \mathbf{E})$ where $\mathbf{X}$ are the node features and $\mathbf{E}$ are the edges determining the connections.

SDE on Graphs

A graph $\mathbf{G}$ can represented by $\mathbf{G} = (\mathbf{X}, \mathbf{E})$ where $\mathbf{X}$ are the node features and $\mathbf{E}$ are the edges determining the connections.
The forward diffusion process can be represented as $\{\mathbf{G}_t = (\mathbf{X}_t)\}_{t=0}^{T}$ [assuming $\mathbf{E}$ remain same during time] with continuous time variable $t \in [0, T]$ where $G_0 \sim p_{data}$ and $G_T \sim p_T$

$$d\mathbf{G}_t = \mathbf{f}_t(\mathbf{G}_t)dt + \mathbf{g}_t(\mathbf{G}_t)d\mathbf{w} \tag{8}$$

SDE on Graphs

A graph $\mathbf{G}$ can represented by $\mathbf{G} = (\mathbf{X}, \mathbf{E})$ where $\mathbf{X}$ are the node features and $\mathbf{E}$ are the edges determining the connections.
The forward diffusion process can be represented as $\{\mathbf{G}_t = (\mathbf{X}_t)\}_{t=0}^{T}$ [assuming $\mathbf{E}$ remain same during time] with continuous time variable $t \in [0, T]$ where $G_0 \sim p_{data}$ and $G_T \sim p_T$

$$d\mathbf{G}_t = \mathbf{f}_t(\mathbf{G}_t)dt + \mathbf{g}_t(\mathbf{G}_t)d\mathbf{w} \tag{8}$$

Following the same analogy, the reverse process can be defined as

$$d\mathbf{G}_t = [\mathbf{f}_t(\mathbf{G}_t) - \mathbf{g}_t^2 \nabla_{\mathbf{G}_t} \log p_t(\mathbf{G}_t)]d\tilde{t} + \mathbf{g}_t(\mathbf{G}_t)d\tilde{\mathbf{w}} \tag{9}$$

SDE on Graphs

- The reverse process can be defined as

$$d\mathbf{G}_t = [\mathbf{f}_t(\mathbf{G}_t) - \mathbf{g}_t^2 \nabla_{\mathbf{G}_t} \log p_t(\mathbf{G}_t)]d\tilde{t} + \mathbf{g}_t(\mathbf{G}_t)d\tilde{\mathbf{w}} \tag{10}$$

SDE on Graphs

- The reverse process can be defined as

$$d\mathbf{G}_t = [\mathbf{f}_t(\mathbf{G}_t) - \mathbf{g}_t^2 \nabla_{\mathbf{G}_t} \log p_t(\mathbf{G}_t)]d\tilde{t} + \mathbf{g}_t(\mathbf{G}_t)d\tilde{\mathbf{w}} \tag{10}$$

- $p_t(\mathbf{G}_t)$ can also be factorized over distribution for i^{th} node and conditional over $\mathbf{N}_{\mathbf{J(i)}}$ being 1-hop neighbours as

$$p_t(\mathbf{G}_t) = \Pi_i^{N(V)} p(\mathbf{X}_t^i | \mathbf{N}_{\mathbf{J(i)}}) \tag{11}$$

$$\nabla_{\mathbf{G}_t} \log p_t(\mathbf{G}_t) = \sum_i^{N(V)} \nabla_{\mathbf{X}_t^i} \log p_t(\mathbf{X}_t^i | \mathbf{N}_{\mathbf{J(i)}}) \tag{12}$$

SDE on Graphs

- The reverse process can be defined as

$$d\mathbf{G}_t = [\mathbf{f}_t(\mathbf{G}_t) - \mathbf{g}_t^2 \nabla_{\mathbf{G}_t} \log p_t(\mathbf{G}_t)]d\tilde{t} + \mathbf{g}_t(\mathbf{G}_t)d\tilde{\mathbf{w}} \tag{10}$$

- $p_t(\mathbf{G}_t)$ can also be factorized over distribution for i^{th} node and conditional over $\mathbf{N}_{\mathbf{J(i)}}$ being 1-hop neighbours as

$$p_t(\mathbf{G}_t) = \Pi_i^{N(V)} p(\mathbf{X}_t^i | \mathbf{N}_{\mathbf{J(i)}}) \tag{11}$$

$$\nabla_{\mathbf{G}_t} \log p_t(\mathbf{G}_t) = \sum_i^{N(V)} \nabla_{\mathbf{X}_t^i} \log p_t(\mathbf{X}_t^i | \mathbf{N}_{\mathbf{J(i)}}) \tag{12}$$

- The reverse process becomes

$$d\mathbf{G}_t = [\mathbf{f}_t(\mathbf{G}_t) - \mathbf{g}_t^2 \sum_i^{N(V)} \nabla_{\mathbf{X}_t^i} \log p_t(\mathbf{X}_t^i | \mathbf{N}_{\mathbf{J(i)}})]d\tilde{t} + \mathbf{g}_t(\mathbf{G}_t)d\tilde{\mathbf{w}} \tag{13}$$

SDE on Graphs

- Based on the sparsity we can define $\mathbf{f}_t(\mathbf{G}_t)$ is more sparse, depending on 1-hop neighbourhood for each node i.e. $\mathbf{f}_t^i(\mathbf{X}_t^i, \mathbf{N}_{\mathbf{J(i)}})$ for i^{th} node and $\mathbf{N}_{\mathbf{J(i)}}$ being 1-hop neighbours
- The reverse process becomes

$$d\mathbf{X}_t^i = [\mathbf{f}_t^i(\mathbf{X}_t^i, \mathbf{N}_{\mathbf{J(i)}}) - (\mathbf{g}_t^i)^2 \nabla_{\mathbf{X}_t^i} \log p_t(\mathbf{X}_t^i, \mathbf{N}_{\mathbf{J(i)}})] d\tilde{t} + \mathbf{g}_t^i(\mathbf{X}_t^i, \mathbf{N}_{\mathbf{J(i)}}) d\tilde{\mathbf{w}} \tag{14}$$

SDE on Graphs

- Based on the sparsity we can define $\mathbf{f}_t(\mathbf{G}_t)$ is more sparse, depending on 1-hop neighbourhood for each node i.e. $\mathbf{f}_t^i(\mathbf{X}_t^i, \mathbf{N}_{\mathbf{J(i)}})$ for i^{th} node and $\mathbf{N}_{\mathbf{J(i)}}$ being 1-hop neighbours
- The reverse process becomes

$$d\mathbf{X}_t^i = [\mathbf{f}_t^i(\mathbf{X}_t^i, \mathbf{N}_{\mathbf{J(i)}}) - (\mathbf{g}_t^i)^2 \nabla_{\mathbf{X}_t^i} \log p_t(\mathbf{X}_t^i, \mathbf{N}_{\mathbf{J(i)}})] d\tilde{t} + \mathbf{g}_t^i(\mathbf{X}_t^i, \mathbf{N}_{\mathbf{J(i)}}) d\tilde{\mathbf{w}} \tag{14}$$

- For score matching [Hyvärinen 2005], lets define $\psi(\mathbf{G}_t, \theta)$ the model density and likewise $\psi_D(\mathbf{G}_t)$ the score function of distribution of observed data D.

$$\psi(\mathbf{X}_t, \theta) = \sum_i^{N(V)} \underbrace{\nabla_{\mathbf{X}_t^i} \log p_t(\mathbf{X}_t^i | \mathbf{N}_{\mathbf{J(i)}})}_{\psi(\mathbf{X}_t^i, \theta)} \tag{15}$$

$$\psi_D(\mathbf{X}_t) = \nabla_{\mathbf{X}_t} \log p_D(\mathbf{X}_t) \tag{16}$$

Objective

- We can use the expected square distance as

$$J(\theta) = \int_{\mathbf{X}_t} p_D(\mathbf{X}_t) \|\psi(\mathbf{X}_t, \theta) - \psi_D(\mathbf{X}_t)\| d\mathbf{X}_t \approx \mathbb{E}_{\mathbf{X}_t} \|\psi(\mathbf{X}_t, \theta) - \psi_D(\mathbf{X}_t)\| \tag{17}$$

$$\hat{\theta} = \text{argmin}_\theta J(\theta) \tag{18}$$

Objective

- We can use the expected square distance as

$$J(\theta) = \int_{\mathbf{X}_t} p_D(\mathbf{X}_t) \|\psi(\mathbf{X}_t, \theta) - \psi_D(\mathbf{X}_t)\| d\mathbf{X}_t \approx \mathbb{E}_{\mathbf{X}_t} \|\psi(\mathbf{X}_t, \theta) - \psi_D(\mathbf{X}_t)\| \tag{17}$$

$$\hat{\theta} = \text{argmin}_\theta J(\theta) \tag{18}$$

- One can modify it using integration by parts to write

$$J(\theta) = \int_{\mathbf{X}_t} p_D(\mathbf{X}_t) \sum_{i=1}^{n} \left[\partial_i \psi(\mathbf{X}_t, \theta) + \frac{1}{2} \psi(\mathbf{X}_t, \theta)^2 \right] \tag{19}$$

$$\partial_i \psi(\mathbf{X}_t, \theta) = \frac{\partial \sum_i^{N(V)} \nabla_{\mathbf{X}_t^i} \log p_t(\mathbf{X}_t^i | \mathbf{N}_{\mathbf{J(i)}})}{\partial \mathbf{X}_t^i} \tag{20}$$

- Use the $\nabla_\theta J(\theta)$ to compute gradients and optimize

THANK YOU
FEEDBACK?