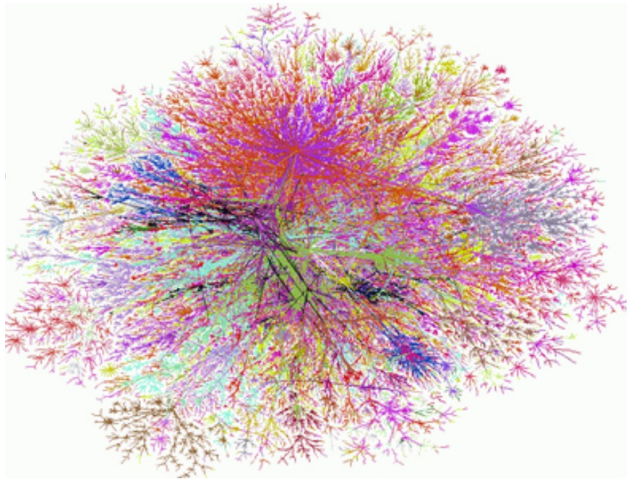
An Introduction to Deep Graph Generation

Xinyang Liu
2023.6.28

The Problem: Graph Generation

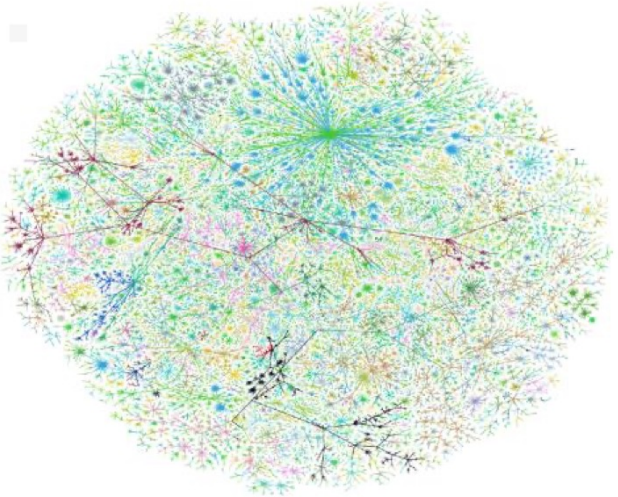
Graph Generative Model

Generate ⇨



Synthetic graph

which is similar to ≈



Real graph

Applications:

- Drug discovery, material design
- Social network modeling

Why do we study Graph Generation?

- **Insights** - Understand the formulation of graphs
- **Predictions** - Predict how will the graph further evolve
- **Simulations** - Use the same process to generate novel graph instances
- **Anomaly detection** - Decide if a graph is normal / abnormal

Graph Generation Tasks

Task 1: Realistic graph generation

- Generate graphs that are **similar to a given set of graphs**

Task 2: Goal-directed graph generation

- Generate graphs that **optimize given objectives/constraints**

  E.g., Drug molecule generation/optimization

Problem Definition

$G = (\mathcal{V}, \mathcal{E}, \boldsymbol{X}, \boldsymbol{A})$

$\mathcal{V}$ is the vertex set

$\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set

$\boldsymbol{X} \in \mathbb{R}^{N \times D}, \boldsymbol{A} \in \mathbb{R}^{N \times N \times F}$

Given a set of M observed graphs $\mathcal{G} = \{G_i\}_{i=1}^{M}$,

graph generation learns the distribution of these graph $p(\mathcal{G})$

Then, a new graph can be sampled $G_{new} \sim p(\mathcal{G})$

Problem Definition

$G = (\mathcal{V}, E, \boldsymbol{X}, \boldsymbol{A})$

$\mathcal{V}$ is the vertex set

$E \subseteq \mathcal{V} \times \mathcal{V}$ is the edge set

$\boldsymbol{X} \in \mathbb{R}^{N \times D}, \boldsymbol{A} \in \mathbb{R}^{N \times N \times F}$

Related Problems

- Link prediction
- Graph structure learning
- Generative sampling
- Set generation

Given a set of M observed graphs $\mathcal{G} = \{G_i\}_{i=1}^{M}$,

graph generation learns the distribution of these graph $p(\mathcal{G})$

Then, a new graph can be sampled $G_{new} \sim p(\mathcal{G})$

Generative Methods

Deep Generative Models (DGMs)

- Autoregressive Models (ARs)
- Variational Autoencoders (VAEs)
- Generative Adversarial Networks (GANs)
- Normalizing Flows (NFs)
- Energy-Based Models (EBMs)
- Diffusion models

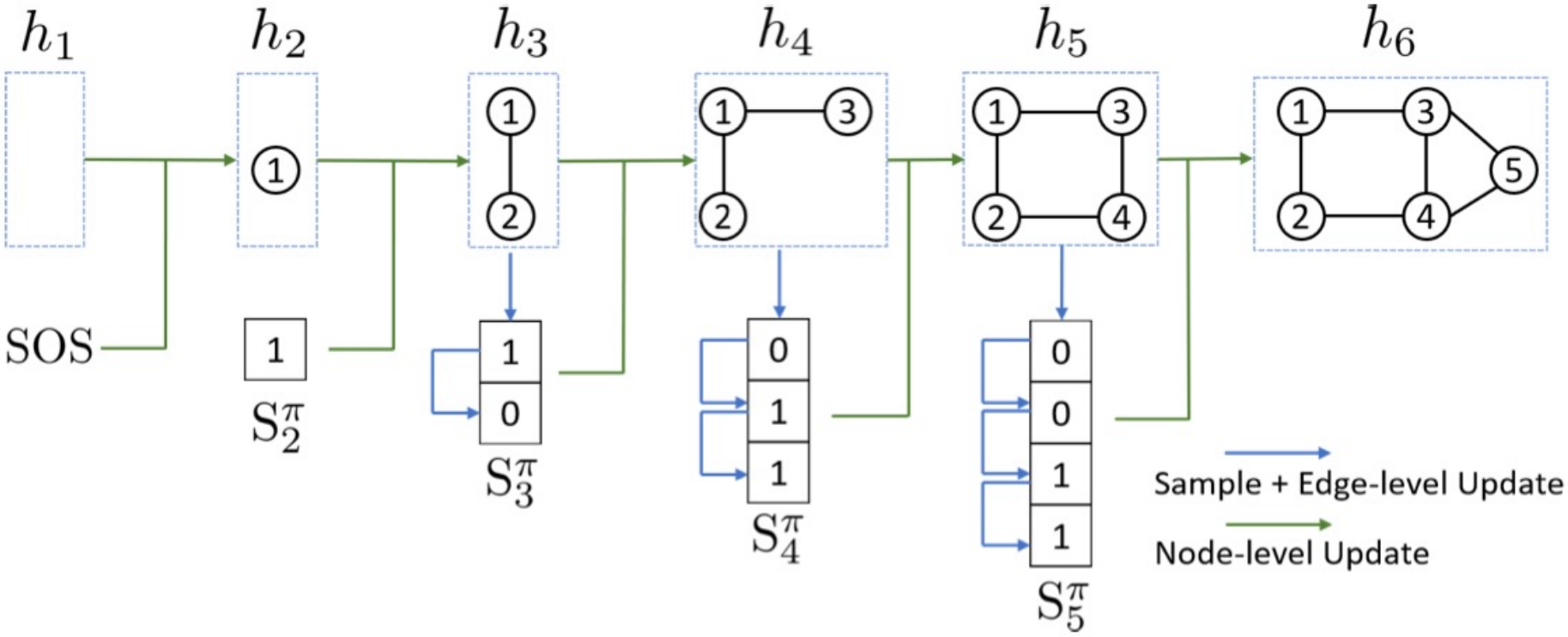
• Combinatorial Optimization Methods (COMs)

- Reinforcement Learning
- Bayesian Optimization (BO)
- Markov Chain Monte Carlo (MCMC)
- …

Autoregressive Models (ARs) for Graph Generation

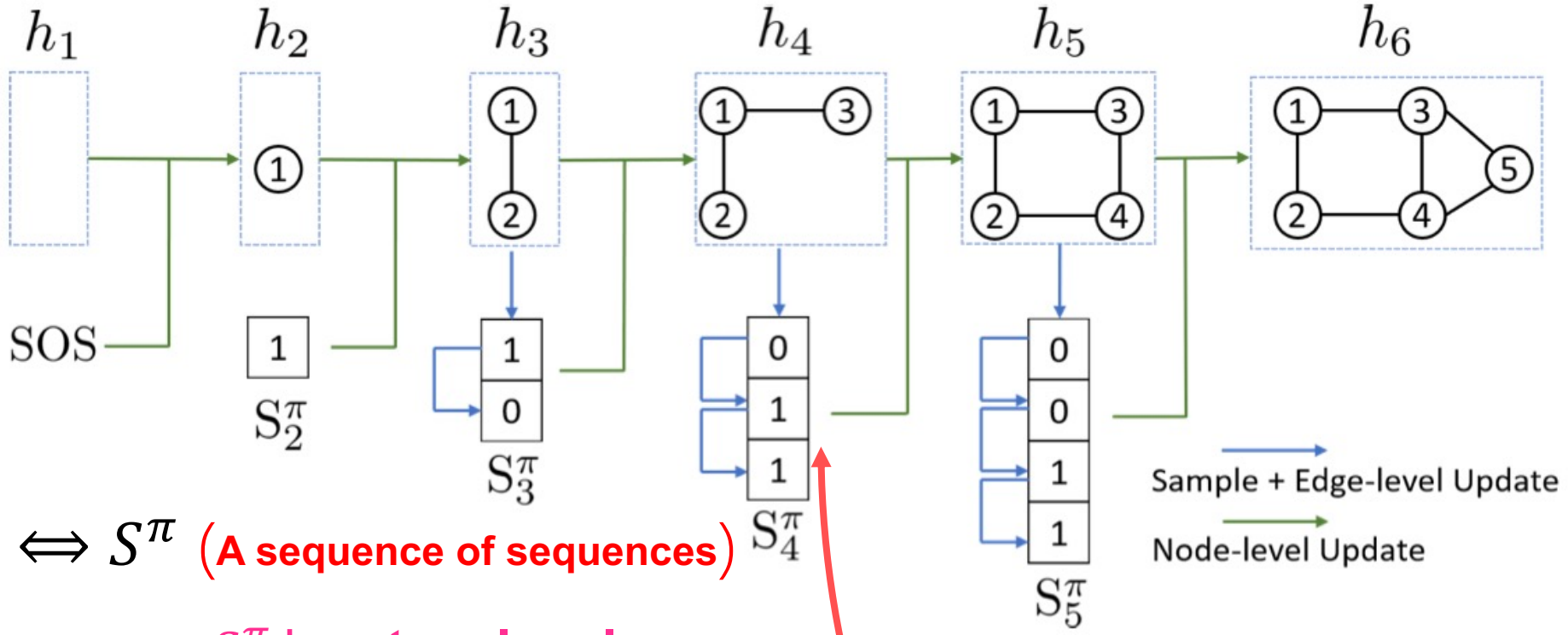
$$p_{model}(\boldsymbol{x};\theta) = \prod_{t=1}^{n} p_{model}(x_t | x_1, \ldots, x_{t-1}; \theta)$$

x_t will be the t-th action (add node, add edge)



GraphRNN: Generating Realistic Graphs with Deep Auto-regressive Models. Jiaxuan You, Rex Ying, Xiang Ren, William L. Hamilton and Jure Leskovec, *ICML 2018*

Autoregressive Models (ARs) for Graph Generation



$G + \pi \Leftrightarrow S^\pi$ (**A sequence of sequences**)

The sequence S^π has **two levels**

- **Node-level:** add nodes, one at a time $S^\pi = (S_1^\pi, S_2^\pi, S_3^\pi, S_4^\pi, S_5^\pi)$
- **Edge-level:** add edges between existing nodes $S_4^\pi = (S_{4,1}^\pi, S_{4,2}^\pi, S_{4,3}^\pi)$
  0 1 1

GraphRNN: Generating Realistic Graphs with Deep Auto-regressive Models. Jiaxuan You, Rex Ying, Xiang Ren, William L. Hamilton and Jure Leskovec, *ICML 2018*

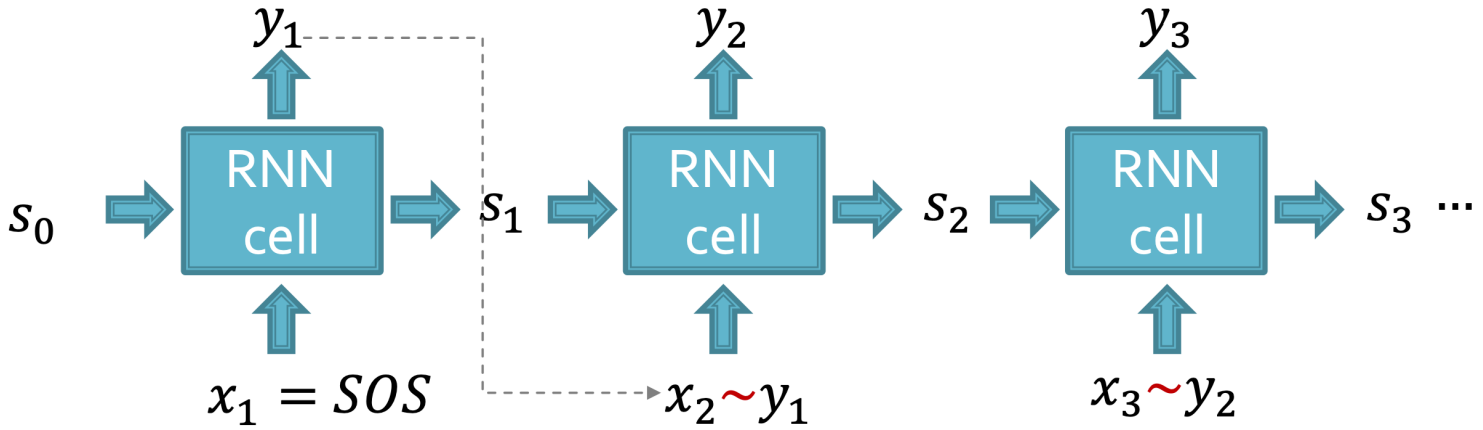
Autoregressive Models (ARs) for Graph Generation

Our goal: Model $p_{model}(\boldsymbol{x};\theta) = \prod_{t=1}^{n} p_{model}(x_t|x_1,\ldots,x_{t-1};\theta)$

Let $y_t = p_{model}(x_t|x_1,\ldots,x_{t-1};\theta)$

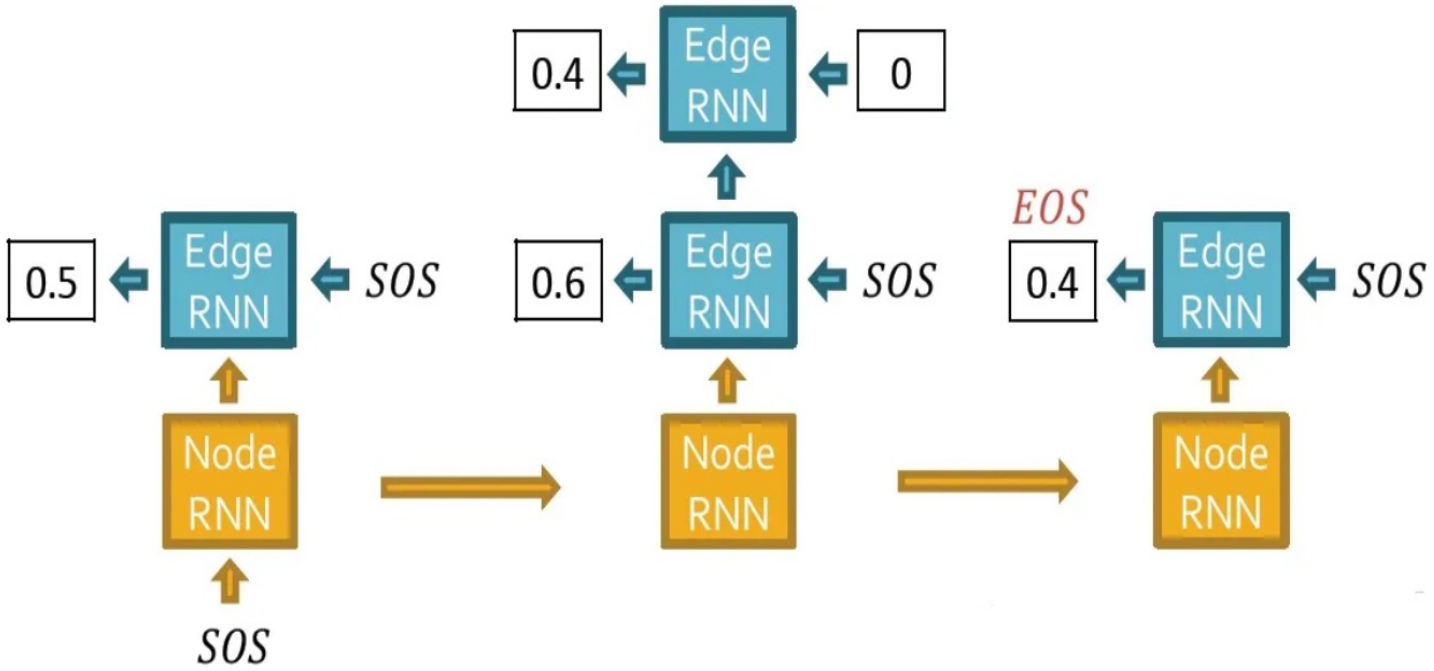
Then we need to sample x_{t+1} from y_t: $x_{t+1}\sim y_t$

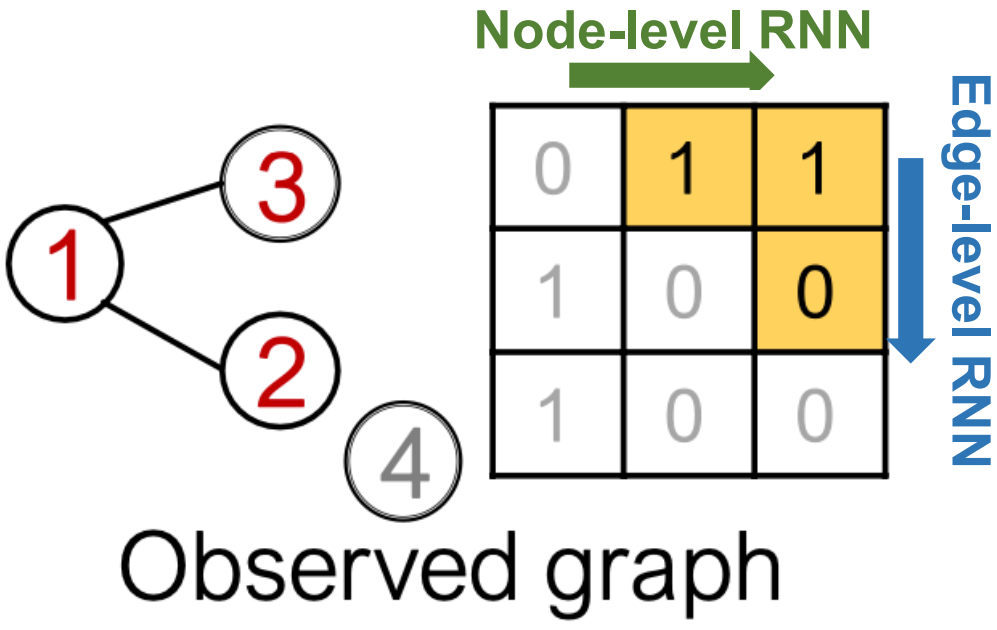
- Each step of RNN outputs a **probability of a single edge**
- We then sample from the distribution, and feed sample to next step:



GraphRNN: Generating Realistic Graphs with Deep Auto-regressive Models. Jiaxuan You, Rex Ying, Xiang Ren, William L. Hamilton and Jure Leskovec, *ICML 2018*

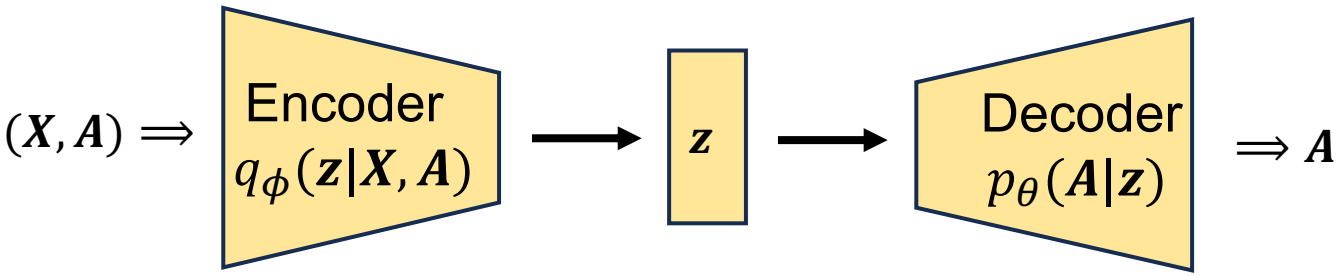
Autoregressive Models (ARs) for Graph Generation





GraphRNN: Generating Realistic Graphs with Deep Auto-regressive Models. Jiaxuan You, Rex Ying, Xiang Ren, William L. Hamilton and Jure Leskovec, *ICML 2018*

Variational Autoencoders (VAEs) for Graph Generation



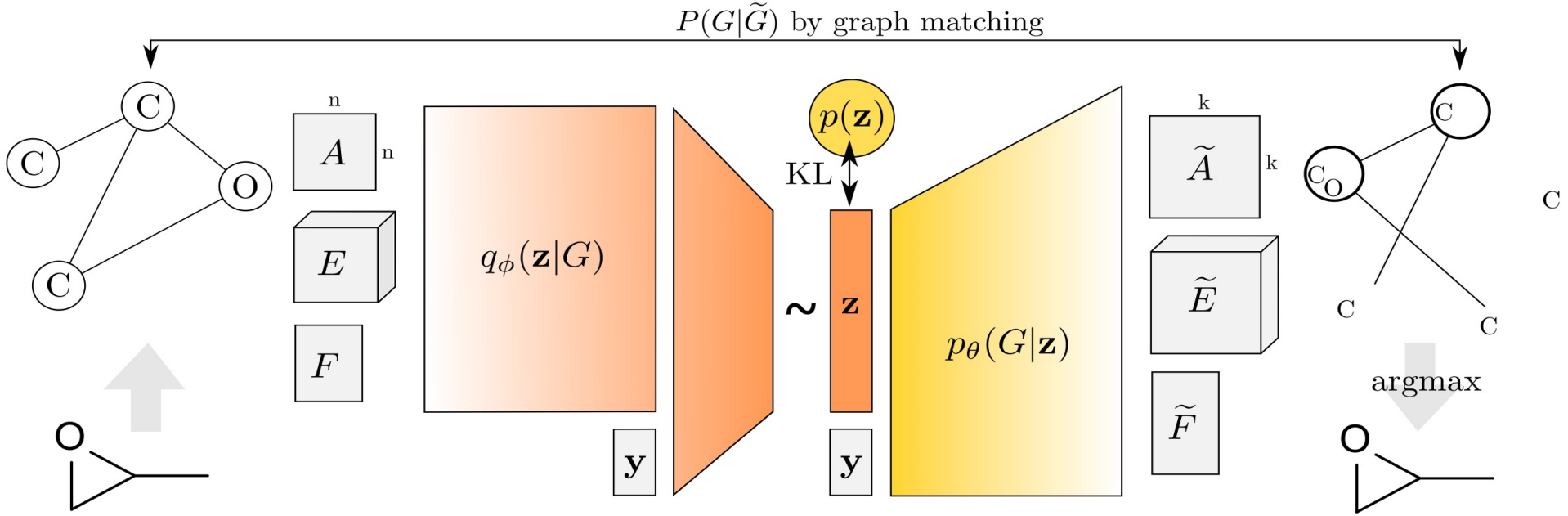
VGAE

$$\mathcal{L} = -\mathbb{E}_{q_\phi(Z|X,A)}\left[\log p_\theta\left(\hat{A}|Z\right)\right] - KL[q_\phi(Z|X,A)||p_\theta(Z)]$$

$$\hat{\mathbf{A}} = \sigma\left(\mathbf{Z}\mathbf{Z}^\top\right), \quad \text{with} \quad \mathbf{Z} = \mathrm{GCN}(\mathbf{X}, \mathbf{A})$$

Variational Graph Auto-Encoders. Thomas N. Kipf and Max Welling. *NIPS 2016 workshop*

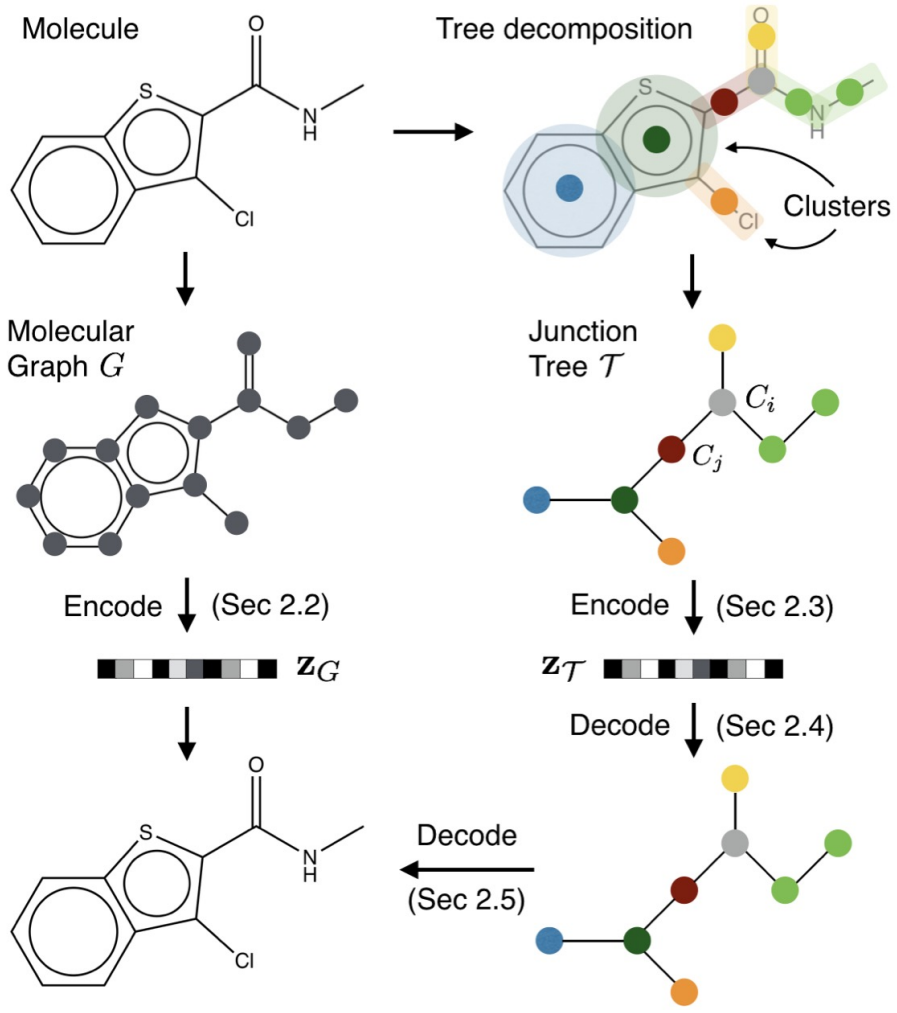
Variational Autoencoders (VAEs) for Graph Generation



GraphVAE $\mathcal{L} = \mathbb{E}_{q_{\boldsymbol{\phi}}(z|\boldsymbol{G})}[-\log p_\theta(\boldsymbol{G}|\boldsymbol{z})] + KL[q_{\boldsymbol{\phi}}(\boldsymbol{z}|\boldsymbol{G})||p_\theta(\boldsymbol{z})]$

GraphVAE: Towards Generation of Small Graphs Using Variational Autoencoders. Martin Simonovsky and Nikos Komodakis, *ICANN 2018*

Variational Autoencoders (VAEs) for Graph Generation

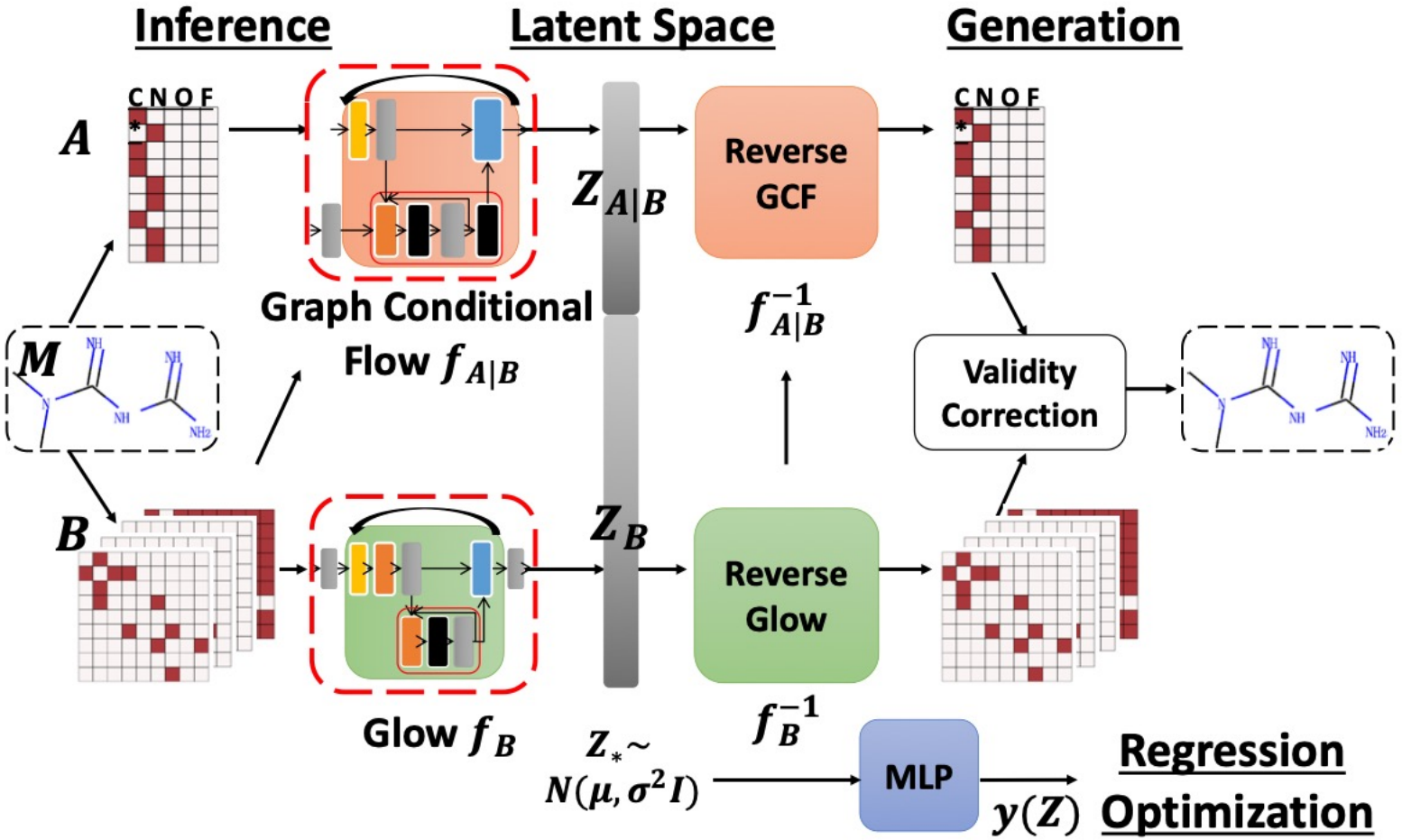


JTVAE

Junction Tree Variational Autoencoder for Molecular Graph Generation.
Wengong Jin, Regina Barzilay and Tommi Jaakkola, *ICML 2018*

Normalizing Flows (NFs) for Graph Generation

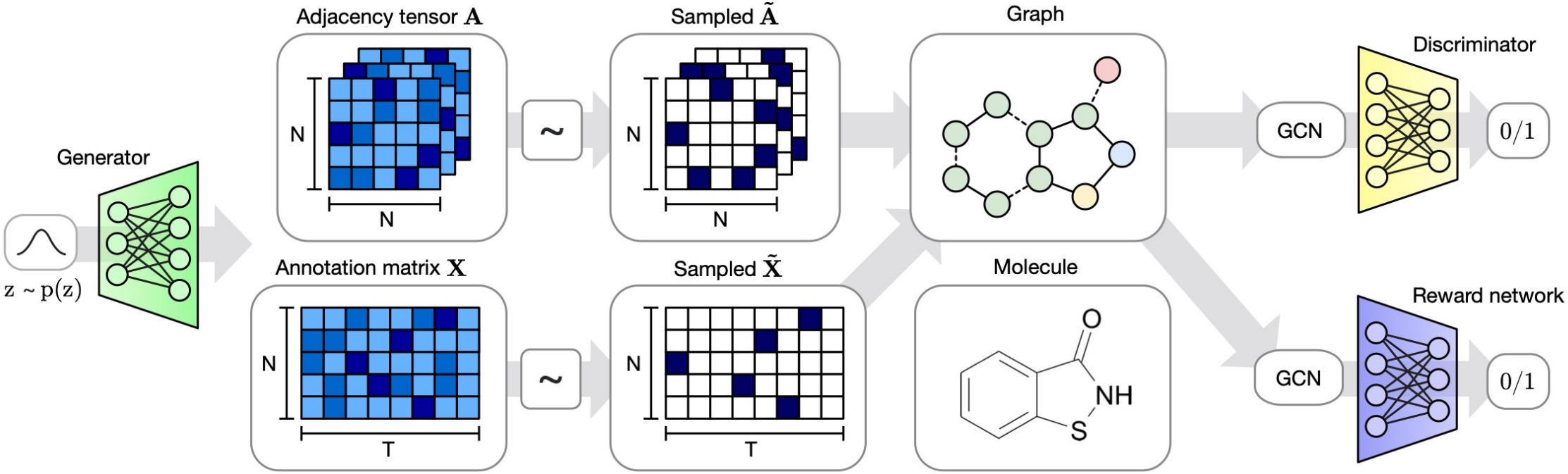
MoFLow



MoFlow: An Invertible Flow Model for Generating Molecular Graphs. Chengxi Zang and Fei Wang, *KDD 2020*

Generative Adversarial Networks (GANs)

for Graph Generation



MolGAN $$L(\theta) = \lambda \cdot L_{WGAN}(\theta) + (1-\lambda) \cdot L_{RL}(\theta)$$

MolGAN: An implicit generative model for small molecular graphs, Nicola De Cao and Thomas Kipf, *ICML 2018 workshop*

Energy-Based Models (EBMs) for Graph Generation

GraphEBM

$$H^{\ell+1} = \sigma\left(\sum_{k=1}^{c+1}\left(A_{(:,:,k)}H^{\ell}W_k^{\ell}\right)\right)$$

$$h_G = \sum_{i=1}^{n} H^{L}_{(i,:)} \qquad \in \mathbb{R}^d$$

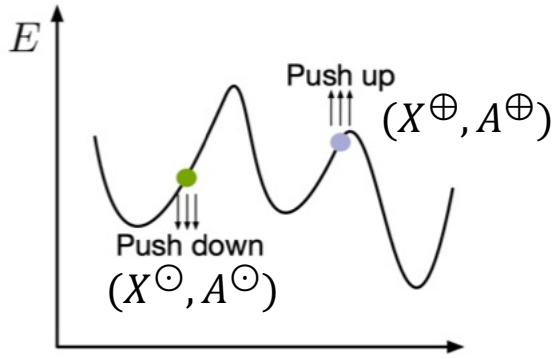
$$E_\theta(X, A) = h_G^T W \qquad \in \mathbb{R}$$

EBMs: $p_\theta(x) = \frac{e^{-E_\theta(x)}}{Z(\theta)}$

$$\mathcal{L}_{energy} = E_\theta(X^{\oplus}, A^{\oplus}) - E_\theta(X^{\odot}, A^{\odot})$$

$$X^k = X^{k-1} - \frac{\lambda}{2}\nabla_X E_\theta\left(X^{k-1}, A^{k-1}\right) + w^k$$

$$A^k = A^{k-1} - \frac{\lambda}{2}\nabla_A E_\theta\left(X^{k-1}, A^{k-1}\right) + \eta^k$$



GRAPHEBM: MOLECULAR GRAPH GENERATION WITH ENERGY-BASED MODELS, Meng Liu, Keqiang Yan, Bora Oztekin and Shuiwang Ji, *ICLR 2021 workshop*

Diffusion models for Graph Generation

EDP-GNN

$$\mathbf{s}_{\boldsymbol{\theta}}(\cdot;\sigma) : \mathcal{A} \to \mathcal{A}.$$

$$\begin{cases} \prod_{i<j} \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{ -\frac{(\tilde{\mathbf{A}}_{[i,j]} - \mathbf{A}_{[i,j]})^2}{2\sigma^2} \right\}, & \text{if } \tilde{\mathbf{A}} = \tilde{\mathbf{A}}^{\mathsf{T}} \\ 0, & \text{otherwise.} \end{cases}$$

$$\mathcal{L}(\boldsymbol{\theta}; \{\sigma_i\}_{i=1}^{L}) \triangleq \frac{1}{2L} \sum_{i=1}^{L} \sigma_i^2 \mathbb{E}\left[\left\| \mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{A}}, \sigma) + \frac{\tilde{\mathbf{A}} - \mathbf{A}}{\sigma^2} \right\|_2^2 \right]$$

sampling

$$(\tilde{\mathbf{A}}_0)_{[i,j]} = \begin{cases} |\varepsilon_{[i,j]}|, & i < j \\ (\tilde{\mathbf{A}}_0)_{[j,i]}, & \text{otherwise,} \end{cases}$$

$$\mathbf{A}_{[i,j]}^{(\text{sample})} = \mathbb{1}_{\tilde{\mathbf{A}}_{[i,j]} > 0.5}$$

Permutation Invariant Graph Generation via Score-Based Generative Modeling. Chenhao Niu, Yang Song, Jiaming Song, Shengjia Zhao, Aditya Grover, Stefano Ermon, *AISTATS 2020*

Diffusion models for Graph Generation

EDP-GNN

$$\mathbf{s}_{\boldsymbol{\theta}}(\cdot;\sigma) : \mathcal{A} \to \mathcal{A}.$$

$$\begin{cases} \prod_{i<j} \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{ - \frac{(\tilde{\mathbf{A}}_{[i,j]} - \mathbf{A}_{[i,j]})^2}{2\sigma^2} \right\}, & \text{if } \tilde{\mathbf{A}} = \tilde{\mathbf{A}}^{\mathsf{T}} \\ 0, & \text{otherwise.} \end{cases}$$

$$\mathcal{L}(\boldsymbol{\theta}; \{\sigma_i\}_{i=1}^{L}) \triangleq \frac{1}{2L} \sum_{i=1}^{L} \sigma_i^2 \mathbb{E}\left[\left\| \mathbf{s}_{\boldsymbol{\theta}}(\tilde{\mathbf{A}}, \sigma) + \frac{\tilde{\mathbf{A}} - \mathbf{A}}{\sigma^2} \right\|_2^2 \right]$$

sampling

$$(\tilde{\mathbf{A}}_0)_{[i,j]} = \begin{cases} |\varepsilon_{[i,j]}|, & i < j \\ (\tilde{\mathbf{A}}_0)_{[j,i]}, & \text{otherwise,} \end{cases}$$

$$\mathbf{A}_{[i,j]}^{(\text{sample})} = \mathbb{1}_{\tilde{\mathbf{A}}_{[i,j]} > 0.5}$$

Edgewise Dense Prediction Graph Neural Network (EDP-GNN)

Node feature inference

$$\mathbf{Z}^{(k+1)} = \text{MultiChannelGNN}^{(k)}(\mathbf{A}^{(k)}, \mathbf{Z}^{(k)})$$

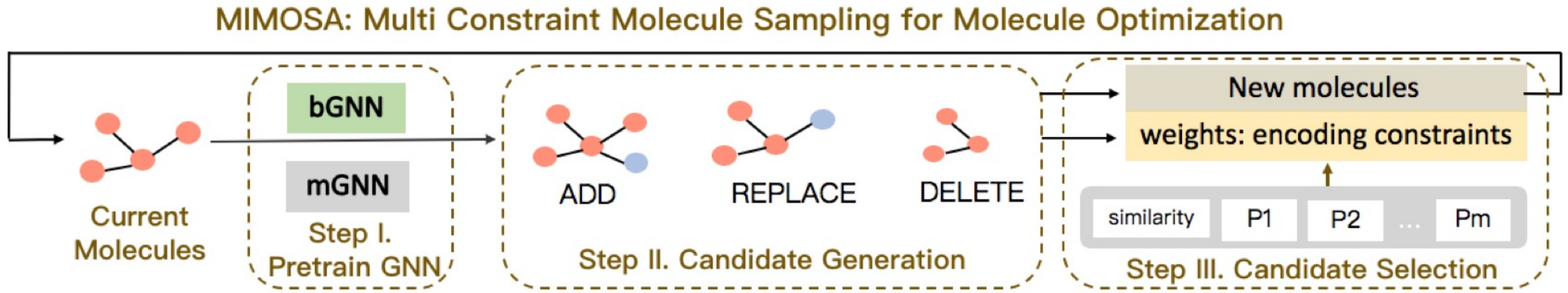
Edge feature inference

$$\tilde{\mathbf{A}}_{[\cdot,i,j]}^{(k+1)} = \text{MLP}_{\text{Edge}}^{(k)}\left(\text{CONCAT}(\mathbf{A}_{[\cdot,i,j]}^{(k)}, \mathbf{Z}_i^{(k+1)}, \mathbf{Z}_j^{(k+1)}) \right)$$

$$\mathbf{A}^{(k+1)} = \tilde{\mathbf{A}}^{(k+1)} + (\tilde{\mathbf{A}}^{(k+1)})^{\mathsf{T}}$$

Permutation Invariant Graph Generation via Score-Based Generative Modeling. Chenhao Niu, Yang Song, Jiaming Song, Shengjia Zhao, Aditya Grover, Stefano Ermon, *AISTATS 2020*

MCMC Method for Molecule Optimization



mGNN: Multi-class classification for the masked node
bGNN: Binary classification for the molecule topology

$$w_r = \frac{p_X(Y') \cdot [\mathrm{mGNN}(Y, v)]_{s'_v}}{p_X(Y) \cdot [\mathrm{mGNN}(Y, v)]_{s_v}},$$

$$Y' \sim \begin{cases} S_{\mathrm{replace}}(Y' \mid Y), & \text{prob } \gamma_1, \text{accept w. } \min\{1, w_r\}, \\ S_{\mathrm{add}}(Y' \mid Y), & \text{prob. } \gamma_2, \text{accept w. } \min\{1, w_a\}, \\ S_{\mathrm{delete}}(Y' \mid Y), & \text{prob. } \gamma_3, \text{accept w. } \min\{1, w_d\}, \end{cases}$$

$$w_a = \frac{p_X(Y') \cdot \mathrm{bGNN}(Y, u) \cdot [\mathrm{mGNN}(Y', v)]_{s_v}}{p_X(Y) \cdot (1 - \mathrm{bGNN}(Y, u))},$$

$$w_d = \frac{p_X(Y') \cdot \big(1 - \mathrm{bGNN}(Y', u)\big)}{p_X(Y) \cdot \mathrm{bGNN}(Y', u) \cdot [\mathrm{mGNN}(Y, v)]_{s_v}},$$

MIMOSA: Multi-constraint Molecule Sampling for Molecule Optimization. Tianfan Fu, Cao Xiao, Xinhao Li, Lucas M. Glass and Jimeng Sun, *AAAI 2021*

Permutation Equivariance and Invariance

Permutation Equivariance (PE): $f(PAP^T) = Pf(A)$

Permutation Invariance (PI): $f(PAP^T) = f(A)$

Example:

GCN, GAT: Inherently permutation invariant $H^{\ell+1} = \sigma\left(\sum_{k=1}^{c+1}\left(A_{(:,:,k)}H^{\ell}W_k^{\ell}\right)\right)$

$\mathbf{s}: \mathbb{R}^{N\times N} \to \mathbb{R}^{N\times N}$ Permutation equivariant

$\int_{\gamma[\mathbf{0},\mathbf{A}]} \langle \mathbf{s}(\mathbf{X}), \mathrm{d}\,\mathbf{X}\rangle_{\mathrm{F}} + C$ Permutation invariant

$\log p_{\boldsymbol{\theta}}(\mathbf{A}) = \int_{\gamma[\mathbf{0},\mathbf{A}]} \langle \mathbf{s}_{\boldsymbol{\theta}}(\mathbf{X}), \mathrm{d}\,\mathbf{X}\rangle_{\mathrm{F}} + \log p_{\boldsymbol{\theta}}(\mathbf{0})$ Permutation invariant

Reference

Survey

MolGenSurvey: A Systematic Survey in Machine Learning Models for Molecule Design. Yuanqi Du, Tianfan Fu, Jimeng Sun and Shengchao Liu, *2022*

A Survey on Graph Diffusion Models: Generative AI in Science for Molecule, Protein and Material. Mengchun Zhang, Maryam Qamar, Taegoo Kang, Yuna Jung, Chenshuang Zhang, Sung-Ho Bae and Chaoning Zhang*, 2022*

A Survey on Deep Graph Generation: Methods and Applications. Yanqiao Zhu, Yuanqi Du, Yinkai Wang, Yichen Xu, Jieyu Zhang, Qiang Liu and Shu Wu, *LoG 2022*

Methods

GraphRNN: Generating Realistic Graphs with Deep Auto-regressive Models. Jiaxuan You, Rex Ying, Xiang Ren, William L. Hamilton and Jure Leskovec, *ICML 2018*

Variational Graph Auto-Encoders. Thomas N. Kipf and Max Welling. *NIPS 2016*

GraphVAE: Towards Generation of Small Graphs Using Variational Autoencoders. Martin Simonovsky and Nikos Komodakis, ICANN 2018

Junction Tree Variational Autoencoder for Molecular Graph Generation. Wengong Jin, Regina Barzilay and Tommi Jaakkola, *ICML 2018*

MolGAN: An implicit generative model for small molecular graphs. Nicola De Cao and Thomas Kipf, ICML 2018 workshop

GRAPHEBM: MOLECULAR GRAPH GENERATION WITH ENERGY-BASED MODELS. Meng Liu, Keqiang Yan, Bora Oztekin and Shuiwang Ji*, ICLR 2021 workshop*

Permutation Invariant Graph Generation via Score-Based Generative Modeling. Chenhao Niu, Yang Song, Jiaming Song, Shengjia Zhao, Aditya Grover, Stefano Ermon, *AISTATS 2020*

MIMOSA: Multi-constraint Molecule Sampling for Molecule Optimization. Tianfan Fu, Cao Xiao, Xinhao Li, Lucas M. Glass and Jimeng Sun, *AAAI 2021*

Course

Deep Generative Models for Graph. CS224W, Stanford

Rethinking

What tasks can we benefit from Graph Generation?

- **Predictions** - Predict how will the graph further evolve
- **Anomaly detection** - Decide if a graph is normal / abnormal

Graph Generation Tasks

Task 1: Realistic graph generation

- Generate graphs that are **similar to a given set of graphs**

Task 2: Goal-directed graph generation

- Generate graphs that **optimize given objectives/constraints**