Deep Generative Models for Graph Generation

Jian Tang

HEC Montreal

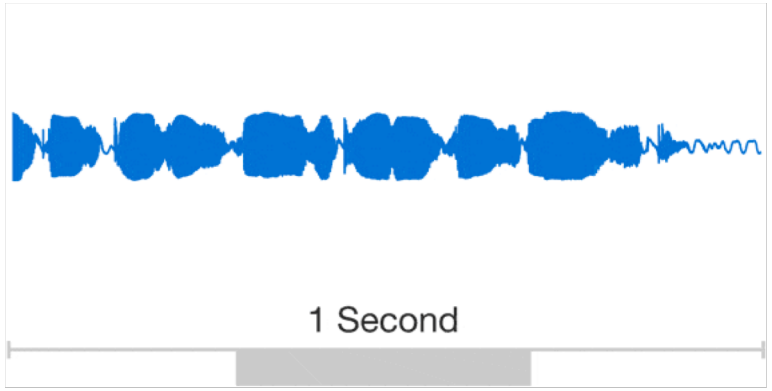
CIFAR AI Chair, Mila

Email: jian.tang@hec.ca

Deep Generative Models

- Goal: model data distribution p(**x**) explicitly or implicitly, where **x** is a high-dimensional random variable.
  - Images, speech, and natural language

Image Generation
(BigGAN, Brock et al. 2018)



Speech Generation
(WaveNet, Oord, et al. 2016)

i love disney movies but this one was not at all what i expected . the story line was so predictable , it was dumb .
i liked the movie but it didn't hold my attention as much as i expected . they just don't make movies like this anymore .
my son loves this movie and it is a good family movie . i would re-comend it for anyone who likes to watch movies with kids .
i bought this case for my ipad 3 . it was not as pictured and it was too small and the ipad mini was not secured inside it , so i returned it .

Natural Language Generation
(Tang et al. 2016)

Applications for Graph Generation

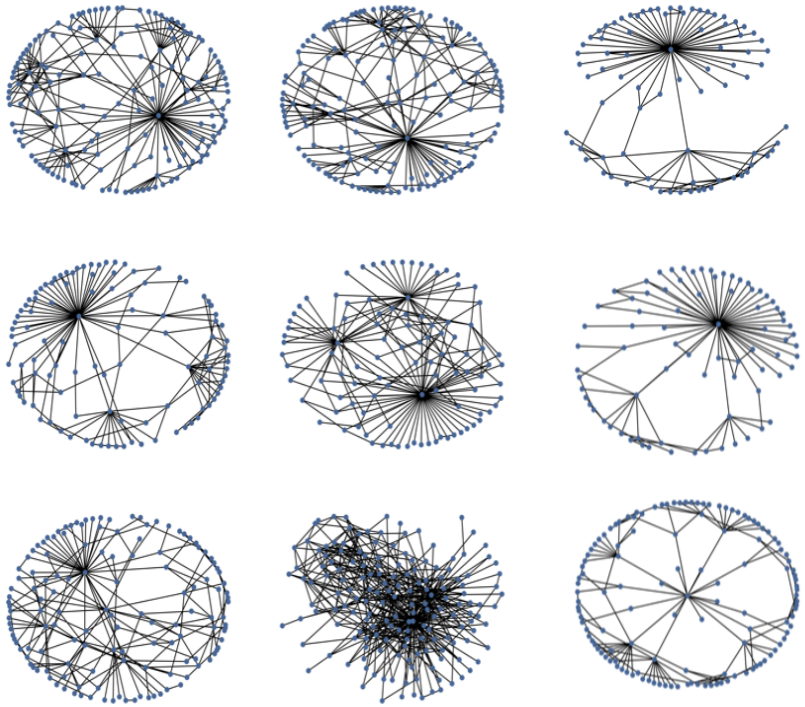
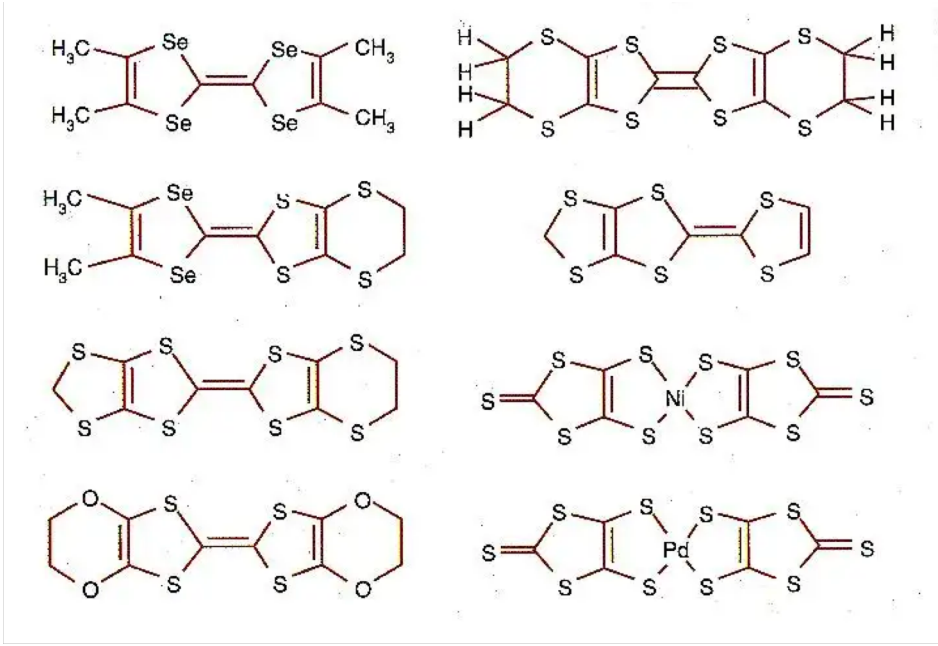


Image from You et al. 2018

Social networks

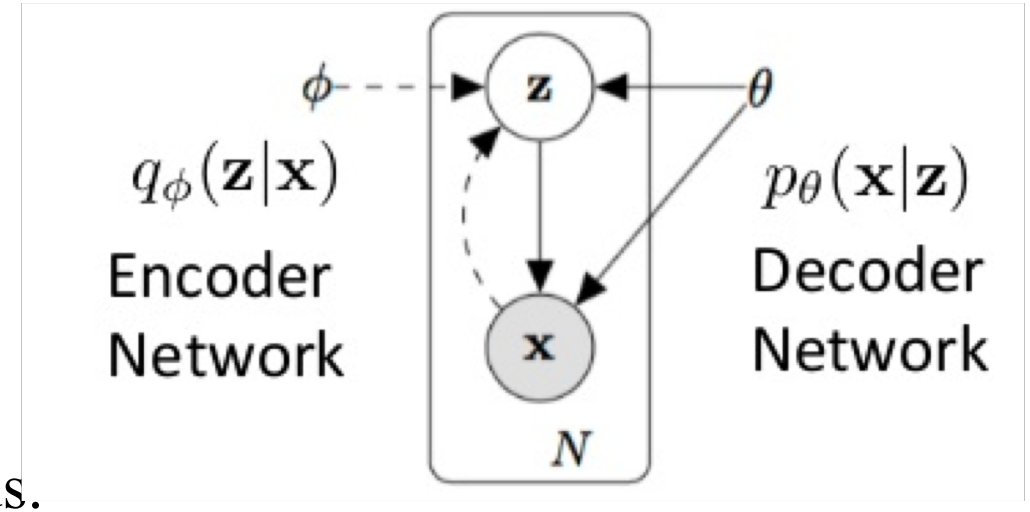


Molecules

Three Types of Deep Generative Models

- Variational Autoencoders (VAEs)
- Generative Adversarial Networks (GANs)
- Deep Auto-regressive Models

Variational AutoEncoders (VAEs, Kingma et al. 2014)



- Latent variable model
  - An encoder $q_\phi\,(\boldsymbol{z}|\boldsymbol{x})$
  - A decoder $p_\theta\,(\boldsymbol{x}|\boldsymbol{z})$
- Maximizing the likelihood $\log \mathrm{p}(\boldsymbol{x})$
  - Inference intractable since $\boldsymbol{z}$ is continuous.
- Maximizing the variational lower-bound $\mathcal{L}(\phi, \theta; x)$
  - Reparametrization trick for jointly optimizing encoder and decoder

$$\mathcal{L}(\phi, \theta; x)$$
$$= \mathbb{E}_{q_\phi\,(\boldsymbol{z}|\boldsymbol{x})}\ \log p_\theta(\boldsymbol{x}|\boldsymbol{z}) - KL[q_\phi(\boldsymbol{z}|\boldsymbol{x})||p(\boldsymbol{z})]$$

Reconstruction Regularization

Generative Adversarial Networks (GANs, Goodfellow et al. 2014)

- A two-player minimax game
  - Generator G: **z** -> **x**
  - Discriminator D: **x**-> {0,1}
- Discriminator aims to distinguish between real data and generated data
- Generator aims to fool the discriminator

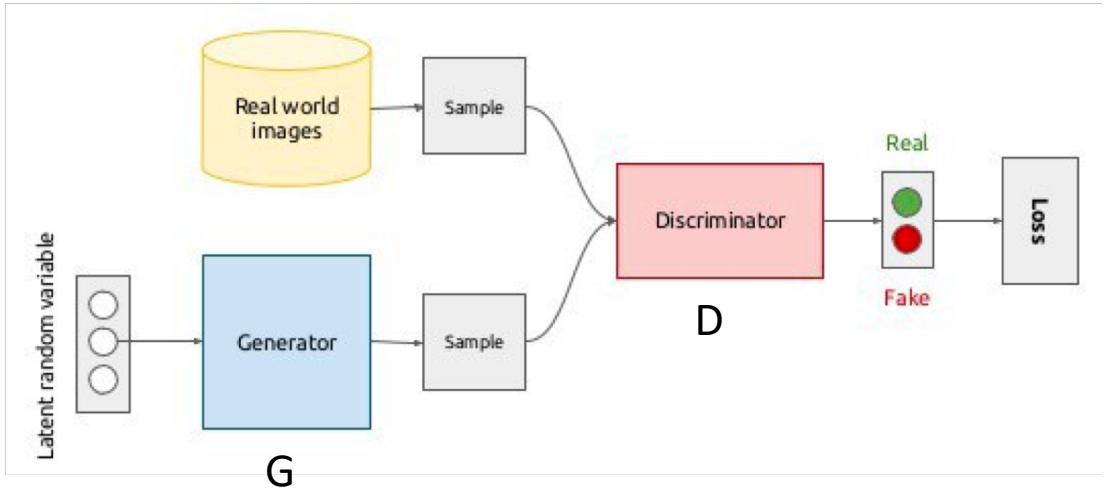


Image from: https://medium.com/coinmonks/celebrity-face-generation-using-gans-tensorflow-implementation-eaa2001eef86

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

Deep Auto-regressive Models (Oort et al. 2016)

- Example of deep auto-regressive model
  - Recurrent Neural Networks
- PixelRNN, Pixel CNN (Oord et al. 2016)
  - Generate an image pixel by pixel
  - A neural network is used to model the conditional distribution
- WaveNet (Oort et al. 2016)

$$p(\mathbf{x}) = \prod_{i=1}^{n^2} p(x_i | x_1, \ldots, x_{i-1})$$

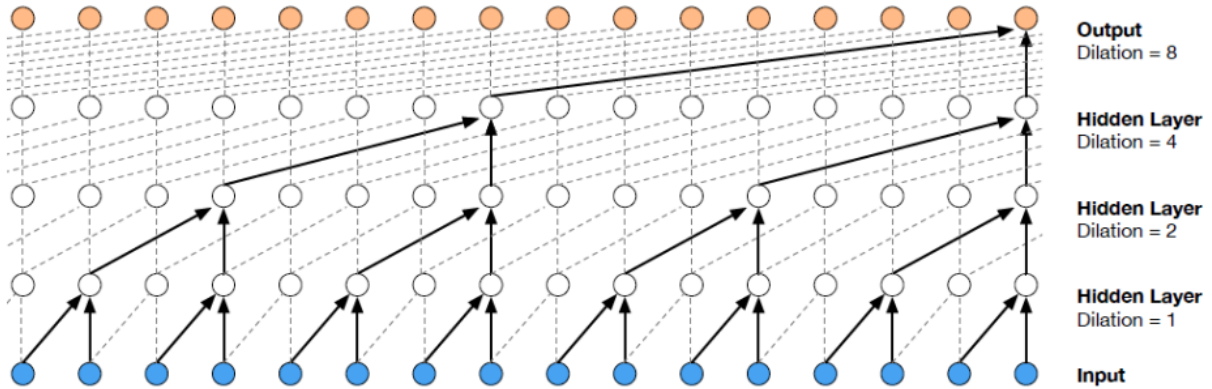


Figure from WaveNet

Challenges for Graph Generation

- The structures and sizes of graphs are different
- No orders between the nodes
- Discrete

GraphVAE (Simonovsky and Komodakis 2018)

- VAE framework for graph generation
  - Graph as input data
  - Encoder: graph neural networks + gated pooling => graph representation (Li et al. 2015)
  - Decoder: output a probabilistic fully-connected graph of a predefined maximum size
    - Model the existence of nodes, edges, and their attributes independently
    - Graph matching is required

GraphVAE (Simonovsky and Komodakis 2018)

- Input graph G=(A,E,F)
  - A: the adjacency matrix, E: edge attribute tensor, F: node attribute matrix

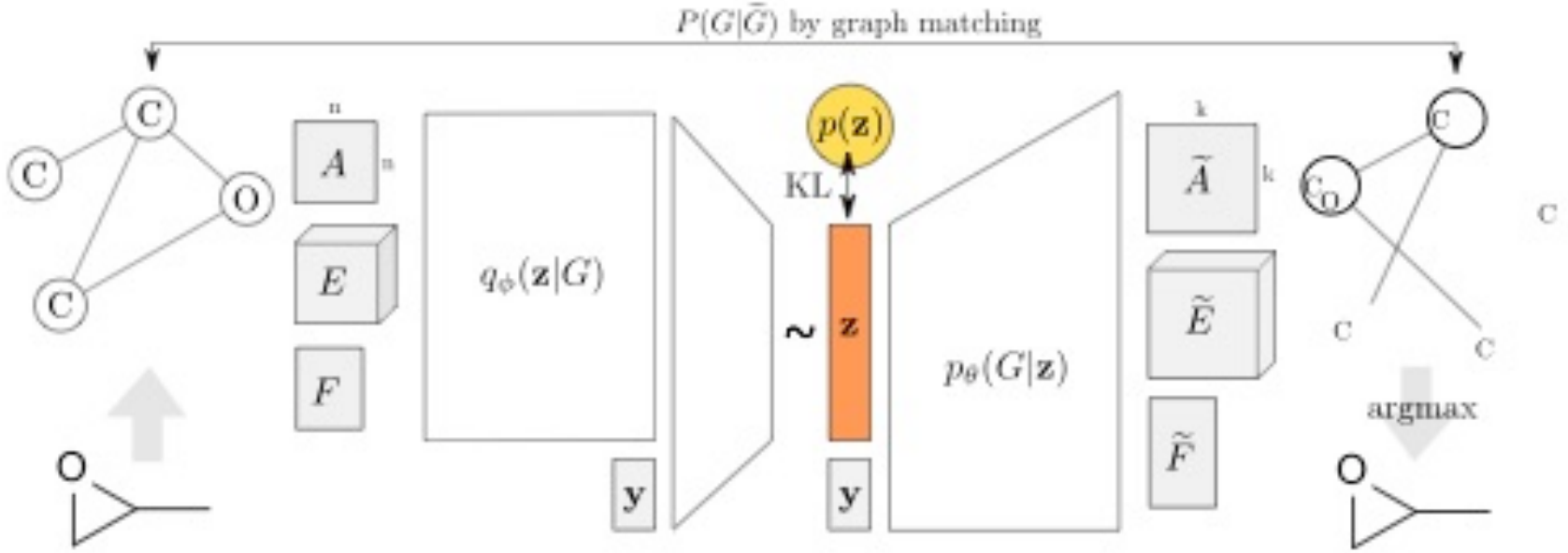


Image from Simonovsky and Komodakis 2018

Probabilistic Graph Decoder

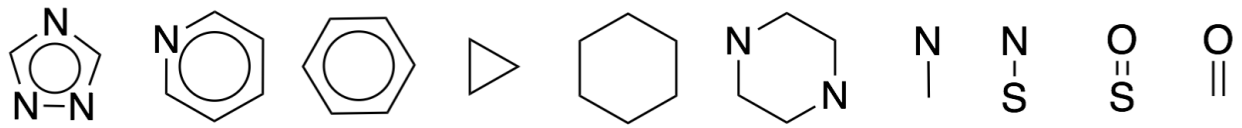
- Restrict the domain to the set of all graphs on maximum k nodes (k is around tens)
- Output a probabilistic fully-connected graph $\tilde{G} = (\tilde{A}, \tilde{E}, \tilde{F})$ on k nodes at once
  - Model the existence of nodes and edges as Bernoulli variables
  - Model the node and edge attributes as Multinomial variables
  - $\tilde{A} \in [0,1]^{k \times k}$ contains both node probabilities $\tilde{A}_{aa}$ and edge probabilities $\tilde{A}_{ab}$ for nodes $a \neq$b
  - $\tilde{E} \in [0,1]^{k \times k \times d_e}$ indicates the probabilities for edge attributes
  - $\tilde{F} \in [0,1]^{k \times d_n}$ indicates the probabilities for node attributes
- Inference: taking edge- and node-wise argmax in $\tilde{A}$, $\tilde{E}$, and $\tilde{F}$.
- Graph Matching must be used for calculating the reconstruction loss

Limitations

- The max size of the graphs/molecules must be predefined.
- Graph matching is required

JTVAE (Jin et al. 2018)

- Leverage chemical domain knowledge
  - Each molecule can be represented a tree-structured scaffold over chemical substructures (e.g., rings, bonds)



- Generate a tree-structured object
  - Represent the scaffold of subgraph components
- Assemble the substructure into a coherent molecular graph

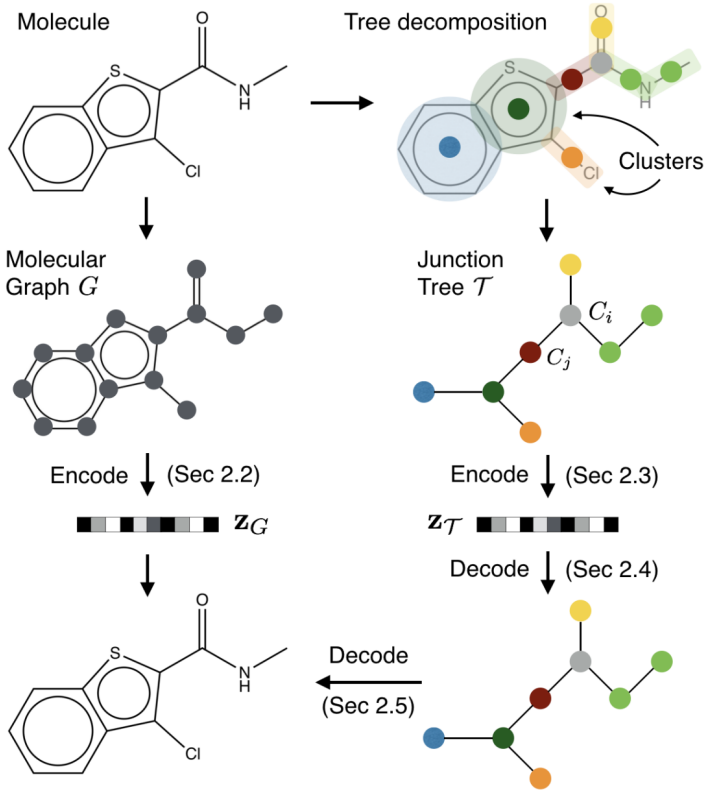
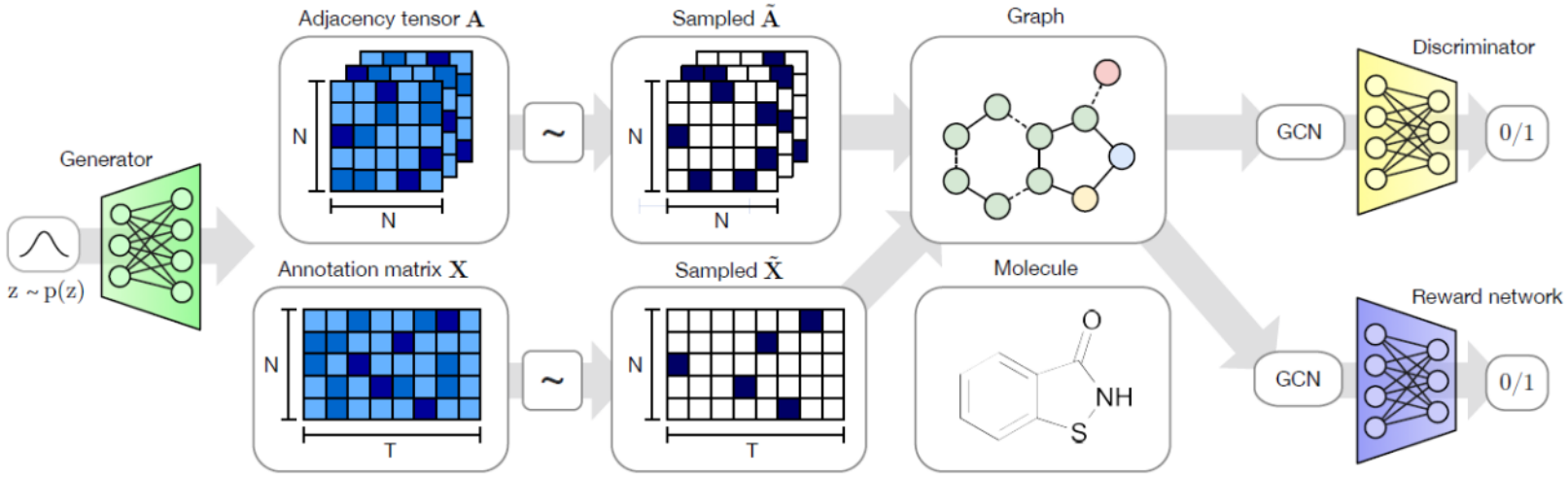


Figure from Jin et al. 2018

MolGAN (Cao and Kipf 2018)

- An implicit, likelihood-free generative model for molecule generation
- Combined with reinforcement learning to encourage the generated molecules with desired chemical properties
- **Generator**: generating molecules from a prior distribution
- **Discriminator**: distinguishing the generated samples and real samples
- **Reward network**:
  - Learns to assign a reward to each molecule to match a score provided by an external software
  - Invalid molecules always receive zero rewards.

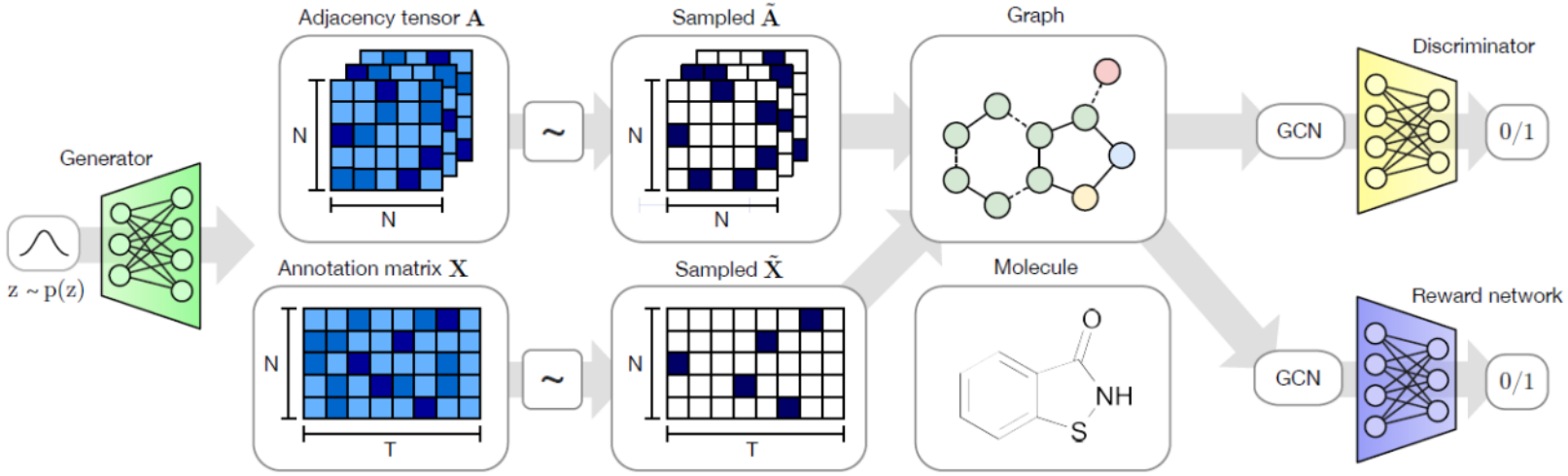
Generator



- A probabilistic fully-connected graph
  - $X \in R^{N \times T}$: atom types
  - $A \in R^{N \times N \times Y}$: bond types
- Objective function:

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

Discriminator and Reward Network



- Learning molecule/graph representations with a variant of neural message passing algorithms (Schlichtkrulle et al. 2017)
- Same architectures for discriminator and reward network
  - No parameter sharing
- Reward network for approximating the score by an external software
  - Trained with real samples and generated samples

Advantages and Limitations

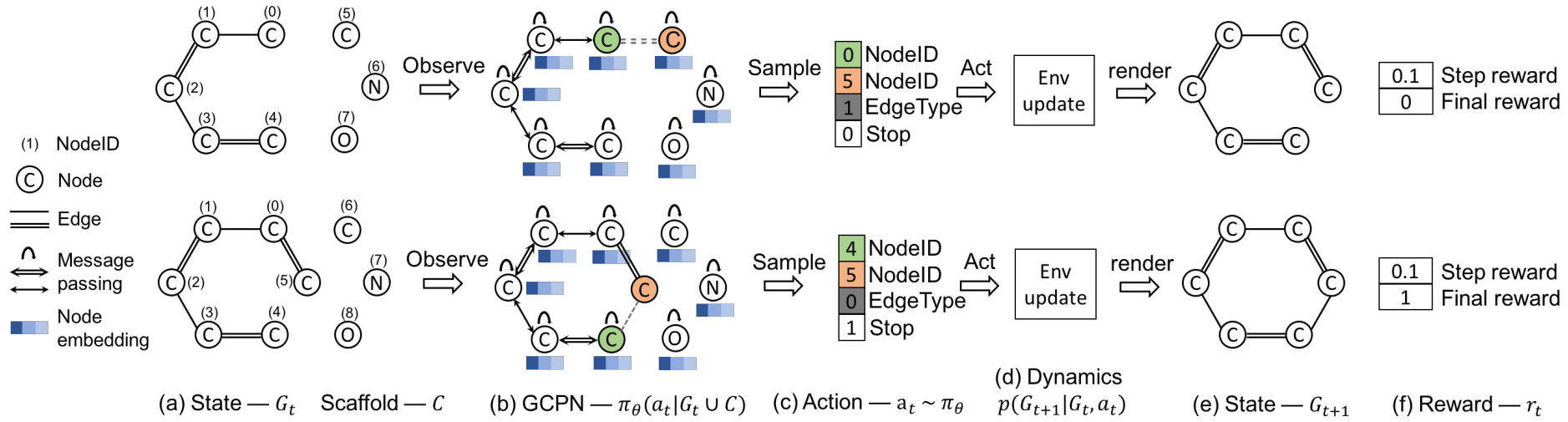
- No graph matching is required
- The max size of the graphs/molecules must be predefined.

GCPN: Graph Convolutional Policy Network (You et al. 2018)

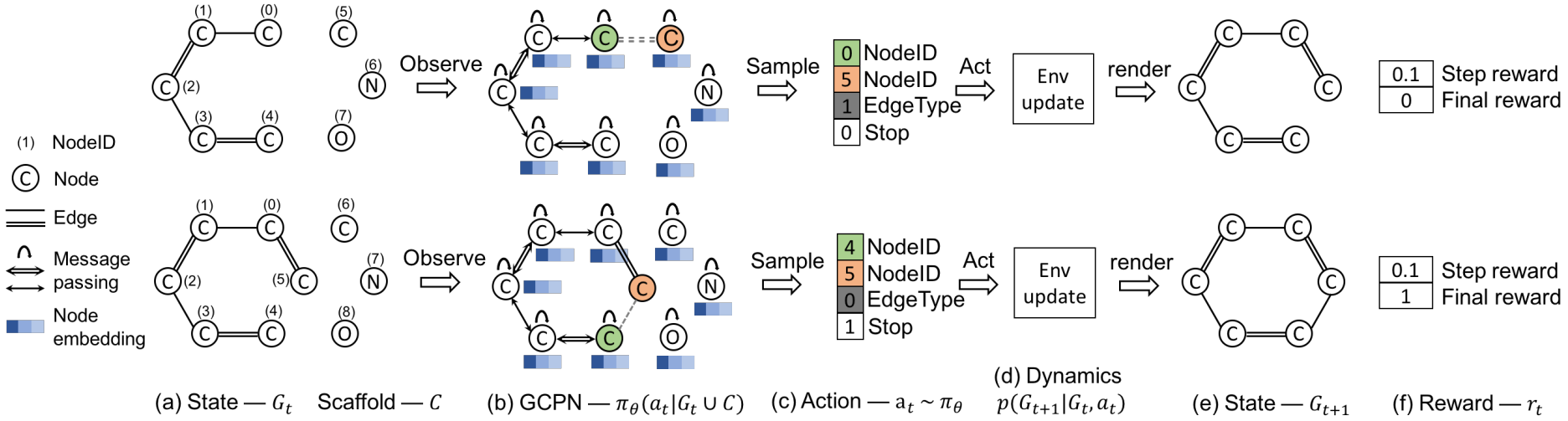
- Molecule generation as sequential decisions
  - Add nodes and edges
  - A Markov decision process
- Goal: discover molecules that optimize desired properties while incorporating chemical rules.
- GCPN: A general model for goal-directed graph generation with RL
  - Optimize adversarial loss and domain-specific rewards with policy gradients
  - Acts in an environment that incorporates domain-specific rules.

Graph Generation as MDP

- $M = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma)$
  - States $\mathcal{S} = \{s_i\}$: consists of all possible intermediate and final graphs
  - Actions $\mathcal{A} = \{a_i\}$: modification made to the current graph at each step
  - State Transitional dynamics $\mathcal{P}$:
  - Reward function $\mathcal{R}$
  - Discount factor γ

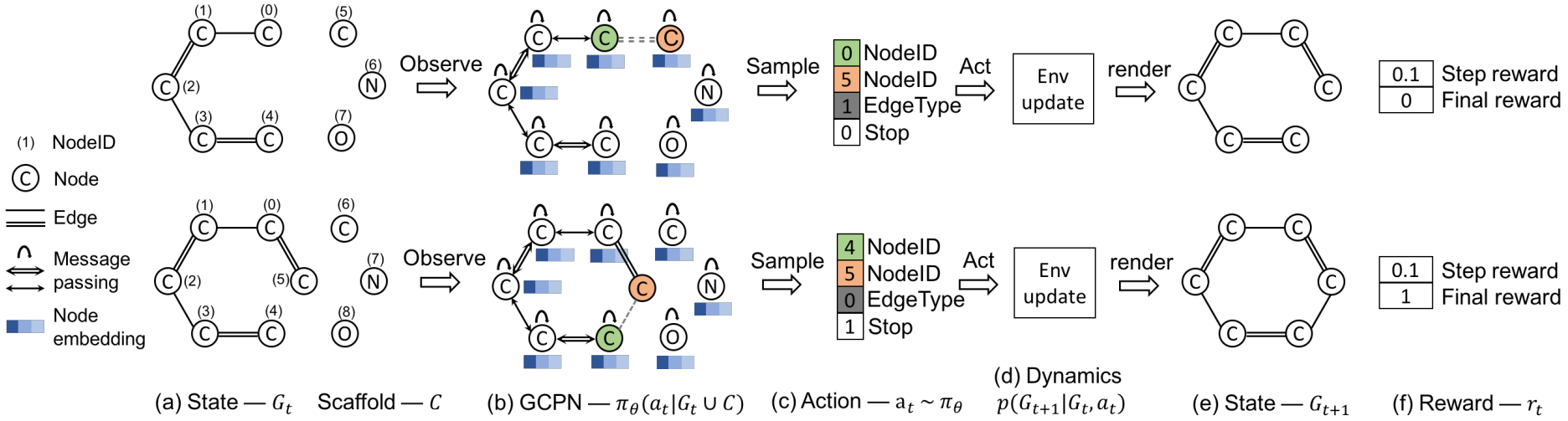


State Space



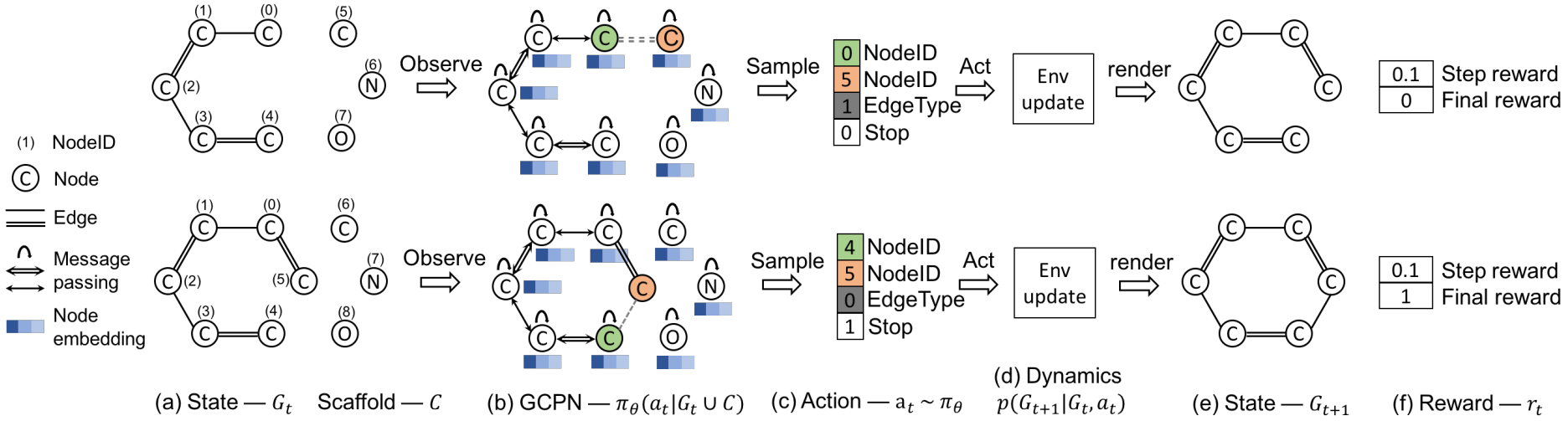
- s_t as the intermediate generated graph G_t
- G_0 contains a single node that represents a carbon atom

Action Space



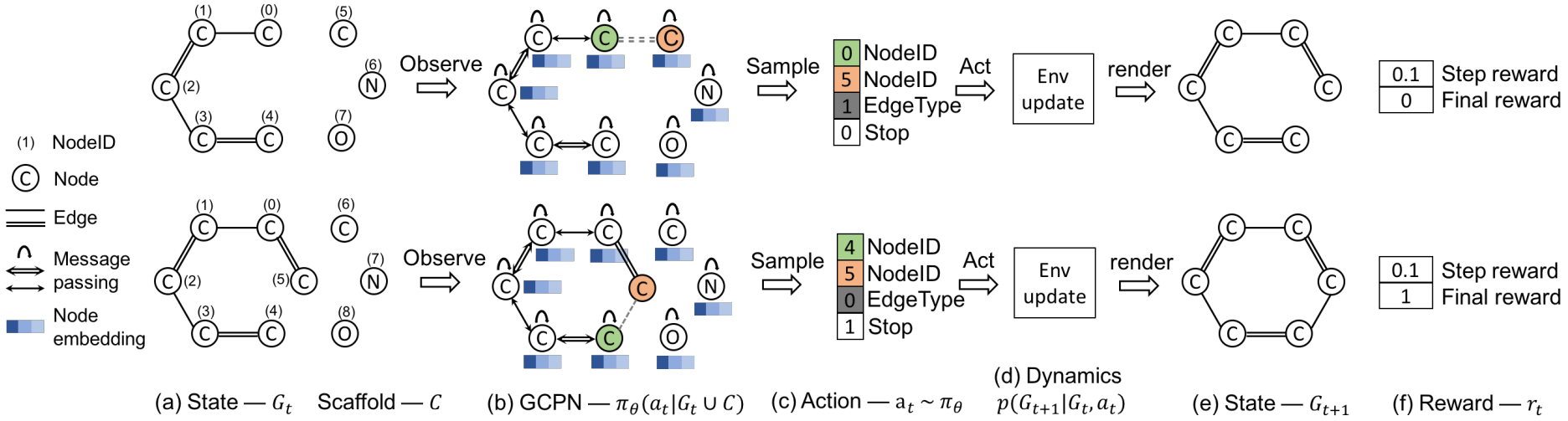
- A set of atoms $C = \cup_{i=1}^{S} C_i$ to be added during each step
- Actions:
  - Connecting a new atom C_i to a node in G_t
  - Connecting exiting nodes within G_t

State Transition Dynamics



- Incorporate domain-specific rules in the state transition dynamics. Only carry out actions that obey the given rules
- Infeasible actions by the policy network are rejected and state remains unchanged

Reward Design



- Final rewards: a sum over domain-specific rewards (e.g., final property scores, penalization of unrealistic molecules and adversarial rewards
- Intermediate rewards: step-wise validity rewards and adversarial rewards

Graph Convolutional Policy Network

- Compute node embeddings with neural message passing algorithms
- Action prediction
  - Selection of two nodes
  - Prediction of edge types
  - Prediction of termination

$$a_t = \text{CONCAT}(a_{\text{first}}, a_{\text{second}}, a_{\text{edge}}, a_{\text{stop}})$$

$$\begin{aligned}
f_{\text{first}}(s_t) &= \text{SOFTMAX}(m_f(X)), \\
f_{\text{second}}(s_t) &= \text{SOFTMAX}(m_s(X_{a_{\text{first}}}, X)), \\
f_{\text{edge}}(s_t) &= \text{SOFTMAX}(m_e(X_{a_{\text{first}}}, X_{a_{\text{second}}})), \\
f_{\text{stop}}(s_t) &= \text{SOFTMAX}(m_t(\text{AGG}(X))),
\end{aligned}$$

$$\begin{aligned}
a_{\text{first}} &\sim f_{\text{first}}(s_t) \in \{0,1\}^n \\
a_{\text{second}} &\sim f_{\text{second}}(s_t) \in \{0,1\}^{n+c} \\
a_{\text{edge}} &\sim f_{\text{edge}}(s_t) \in \{0,1\}^b \\
a_{\text{stop}} &\sim f_{\text{stop}}(s_t) \in \{0,1\}
\end{aligned}$$

Data Sets

- **ZINC,** 250K drug-like organic molecules of up to 38 heavy atoms with 9 distinct atomic numbers and 4 bond types.

Results on Reconstruction and Validity

- Reconstruct the input molecules and decoding valid molecules from the prior distribution

Method	Reconstruction	Validity
CVAE	44.6%	0.7%
GVAE	53.7%	7.2%
SD-VAE[2]	76.2%	43.5%
GraphVAE	-	13.5%
JT-VAE	**76.7%**	**100.0%**

Table from Jin et al. 2018

Results on Property Optimization

- **Property Optimization**: generate novel molecules whose specified molecular properties are optimized.

Table 1: Comparison of the top 3 property scores of generated molecules found by each model.

Method	Penalized logP				QED			
	1st	2nd	3rd	Validity	1st	2nd	3rd	Validity
ZINC	4.52	4.30	4.23	100.0%	0.948	0.948	0.948	100.0%
Hill Climbing	–	–	–	–	0.838	0.814	0.814	100.0%
ORGAN	3.63	3.49	3.44	0.4%	0.896	0.824	0.820	2.2%
JT-VAE	5.30	4.93	4.49	100.0%	0.925	0.911	0.910	100.0%
GCPN	**7.98**	**7.85**	**7.80**	**100.0%**	**0.948**	**0.947**	**0.946**	**100.0%**

Table from You et al. 2018

Results on Property Targeting

- **Property Targeting:** generate novel molecules whose specified molecular properties are as close to the target scores as possible.

Method	$-2.5 \leq \text{logP} \leq -2$		$5 \leq \text{logP} \leq 5.5$		$150 \leq \text{MW} \leq 200$		$500 \leq \text{MW} \leq 550$	
	Success	Diversity	Success	Diversity	Success	Diversity	Success	Diversity
ZINC	0.3%	0.919	1.3%	0.909	1.7%	0.938	0	–
JT-VAE	11.3%	**0.846**	7.6%	0.907	0.7%	0.824	16.0%	0.898
ORGAN	0	–	0.2%	**0.909**	15.1%	0.759	0.1%	0.907
GCPN	**85.5%**	0.392	**54.7%**	0.855	**76.1%**	**0.921**	**74.1%**	**0.920**

Table from You et al. 2018

Results on Constrained Property Optimization

- **Constrained Property Optimization**: generate novel molecules whose specified molecular properties are optimized, while also containing a specified molecular substructure.

Table 3: Comparison of the performance in the constrained optimization task.

δ	JT-VAE			GCPN		
	Improvement	Similarity	Success	Improvement	Similarity	Success
0.0	1.91 ± 2.04	0.28 ± 0.15	97.5%	$\mathbf{4.20 \pm 1.28}$	$\mathbf{0.32 \pm 0.12}$	**100.0%**
0.2	1.68 ± 1.85	0.33 ± 0.13	97.1%	$\mathbf{4.12 \pm 1.19}$	$\mathbf{0.34 \pm 0.11}$	**100.0%**
0.4	0.84 ± 1.45	$\mathbf{0.51 \pm 0.10}$	83.6%	$\mathbf{2.49 \pm 1.30}$	0.47 ± 0.08	**100.0%**
0.6	0.21 ± 0.71	0.69 ± 0.06	46.4%	$\mathbf{0.79 \pm 0.63}$	$\mathbf{0.68 \pm 0.08}$	**100.0%**

Table from You et al. 2018

Generated Molecules

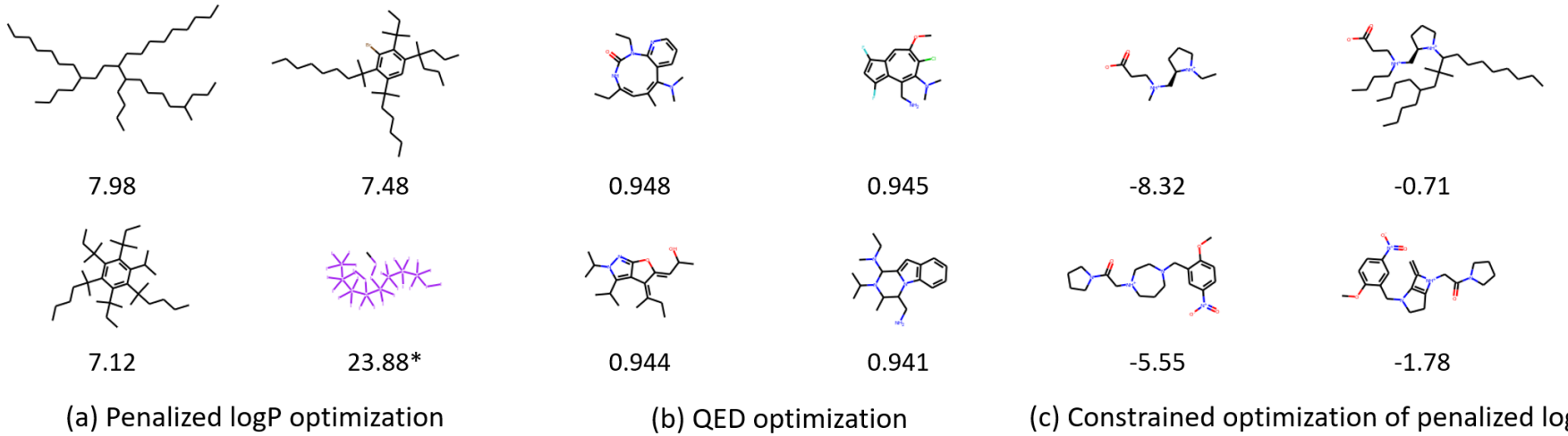


Figure 2: Samples of generated molecules in property optimization and constrained property optimization task. In **(c)**, the two columns correspond to molecules before and after modification.

Table from You et al. 2018

Summary

- Graph generation is an important problem in many areas
  - Generating social networks
  - Drug discovery
  - Designing electric circuits
- Deep generative models for graph
  - GraphVAE, JTVAE
  - MolGAN
  - GCPN

References

- GraphVAE: Towards Generation of Small Graphs Using Variational Autoencoders. https://arxiv.org/pdf/1802.03480.pdf
- MolGAN: An implicit generative model for small molecular graphs. https://arxiv.org/pdf/1805.11973.pdf
- GraphRNN: Generating Realistic Graphs with Deep Auto-regressive Models. https://arxiv.org/pdf/1802.08773.pdf
- Graph Convolutional Policy Network for Goal-Directed Molecular Graph Generation. https://arxiv.org/pdf/1806.02473.pdf
- Junction Tree Variational Autoencoder for Molecular Graph Generation. https://arxiv.org/pdf/1802.04364.pdf

Future Directions

- Application to more domains and applications
  - Graphs are ubiquitous
- Graph neural networks for reasoning
  - Computer vision
  - Natural language understanding
- Graph neural networks + reinforcement learning
  - Relational deep reinforcement learning (Zambaldi et al. 2018)
- Better theoretical understanding of graph neural networks
- …

Thanks!

Contact: jian.tang@hec.ca