Recent Advances on Graph Analytics and Its Applications in Healthcare

KDD 2020 Tutorial

August 23, morning

Fei Wang, Peng Cui, Jian Pei, Yangqiu Song, **Chengxi Zang**

http://www.calvinzang.com/kdd2020_tutorial_medical_graph_analytics.html

Outline

- Introduction
- Network Embedding & GNNs
- Knowledge Graph Mining
- **Graph Generative Models & Drug Discovery**
- Discussions

Graph Generative Models & Drug Discovery

MoFlow: An Invertible Flow Model for Generating Molecular Graphs (KDD 2020 Research)

Chengxi Zang and Fei Wang

Weill Cornell Medicine

www.calvinzang.com

Drug Discovery and Development

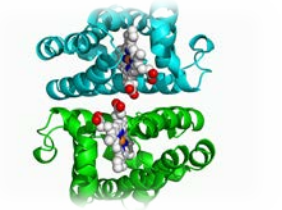
Target

Lead Discovery
1.5 years

Lead Optimization
3 years

Preclinical
1.5 years

Molecule for human Clinical trial 4-7 years



Screen millions of functional molecules to inform design

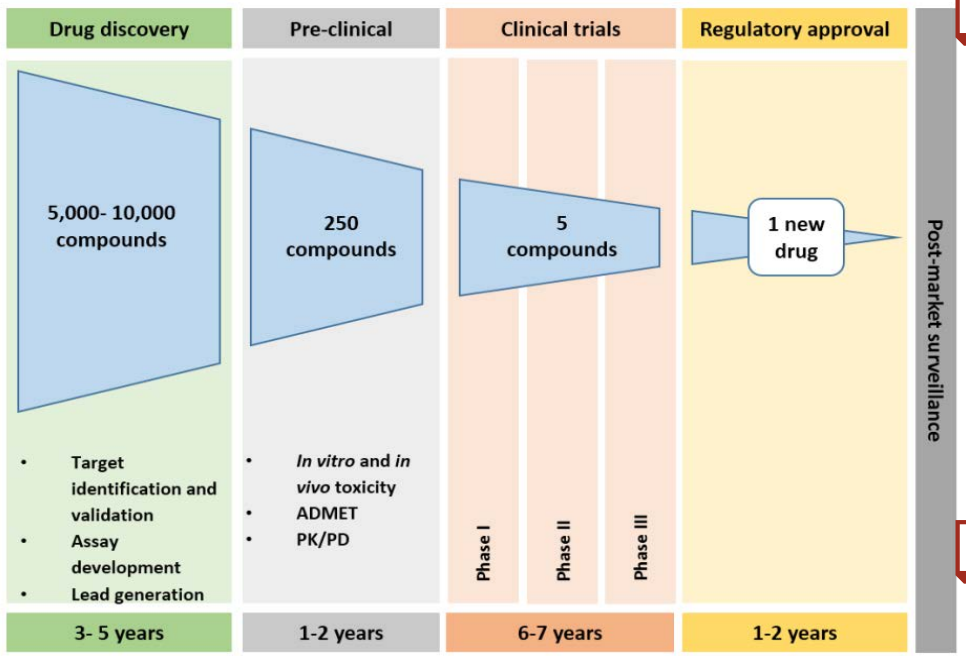
Design, make, test 1000s new and better molecules with **optimized** property

In-vitro and in-vivo **experiments**; synthesis

Phase I, II, III, Launch

Nature Biotechnology 2019

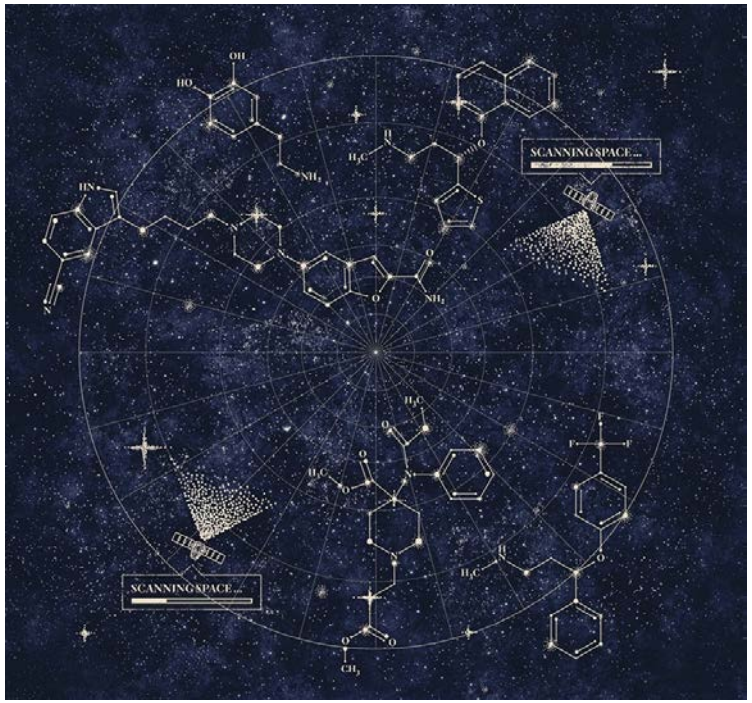
Background: Drug Discovery



- **1. Lengthy, costly, & with high failure rate**
  - $2.6 billion, ≥ 10 years in total, clinical success ~12%, poor translation in patients
  - **Our focus: Drug discovery (lead discovery and optimization)** ~ 5 years and 33% of total cost
- **How to accelerate the process, reduce its cost, and increase success rate?** *Nature 2010* *Proteomes 2016*

Background: Drug Discovery

Target → **Lead Discovery** 1.5 years → Lead Optimization 3 years → Preclinical 1.5 years → **Molecule for human Clinical trial**



- **2. Big chemical space but largely unexplored**
  - The scale of drug-like small molecules: $10^{33} \sim 10^{60}$
  - Existing chemical database to (linearly) screen: $\sim 10^{6}$
  - **A huge gap! Impossible to exhaustive enumeration!**
- **How to efficiently explore such a big chemical space?**

Nature 2017

Background: Drug Discovery

Target → Lead Discovery 1.5 years → **Lead Optimization** **3 years** → Preclinical 1.5 years → Molecule for human Clinical trial

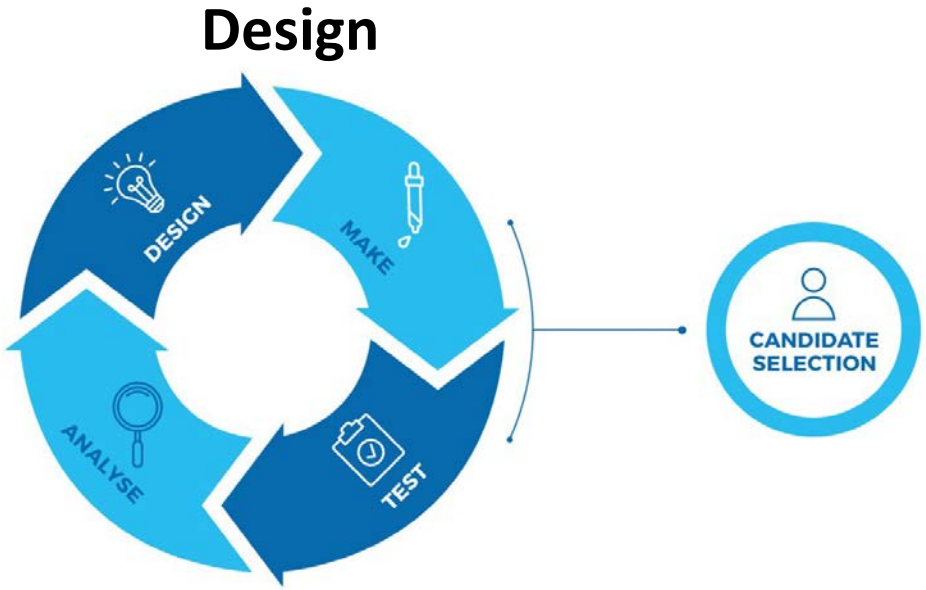


Image from Sygnaturediscovery

- **3. Difficult to optimize molecules**
  - Difficult to **design novel & better** molecules:
    - High-throughput virtual screen, or
    - Medicinal chemists' knowledge
  - Difficult to **evaluate**:
    - expensive experiments, In-vitro, in-vivo, in-silico
- **How to efficiently optimize molecules guided by the targeted properties?**

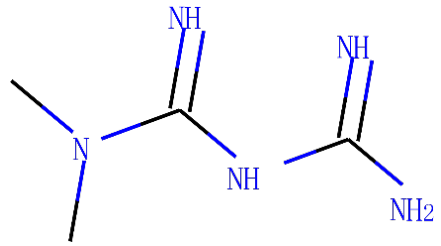
Our Vision: AI for Drug Discovery

- Driven by **AI** and **Big Chemical Data**
- to reduce time, cost and failure rate of drug discovery process
  - 3-5 **years** → 3-5 **months**
- to efficiently explore big chemical space
  - $\sim 10^{60}$ drug-like chemical space
- to efficiently and automatically design novel molecules with optimized properties
  - **automatic**, **in silico**, **learning** from data and human knowledge

Problem Definition

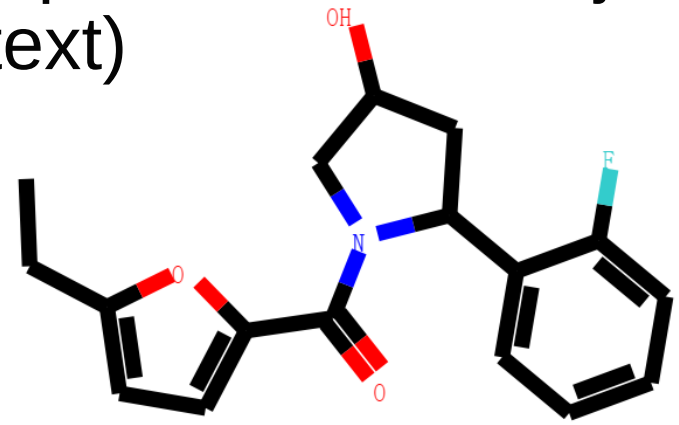
- **Goal: To generate novel molecular graphs with optimized properties**
- **Data Input:**
  - Discrete **2D molecular graphs**, etc.
  - $\{G_1, G_2, \dots, G_N\}$: Molecular graph data samples
  - $\{y_{1,k}, y_{2,k}, \dots, y_{N,i}\}_{k=1\dots K}$ Some properties of molecules
- **Output:**
  - **Novel** molecular graphs $\{G_{N+1}, G_{N+2}, \dots\}$ with **optimized** properties.

Metformin (二甲双胍)
CN(C)C(=N)N=C(N)N

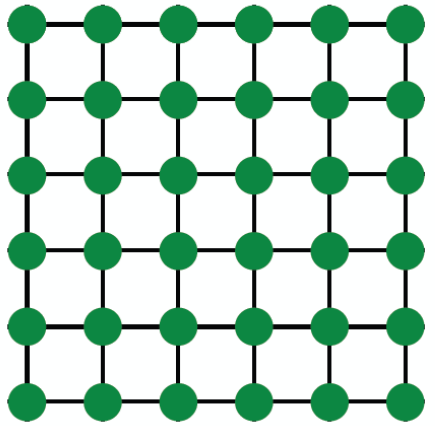


Why Is It Hard?

- **Discrete molecular graph data and its combinatorial complexity**
  - Nodes/atoms and edges/bonds can have multiple types
    - Node types: C, H, O, etc., Edge types: single, double, triple bond.
  - Combinatorial Complexity
    - the scale of small molecular graphs $\sim 10^{60}$
  - Deep models are majorly designed for regular grid structures (image or text)

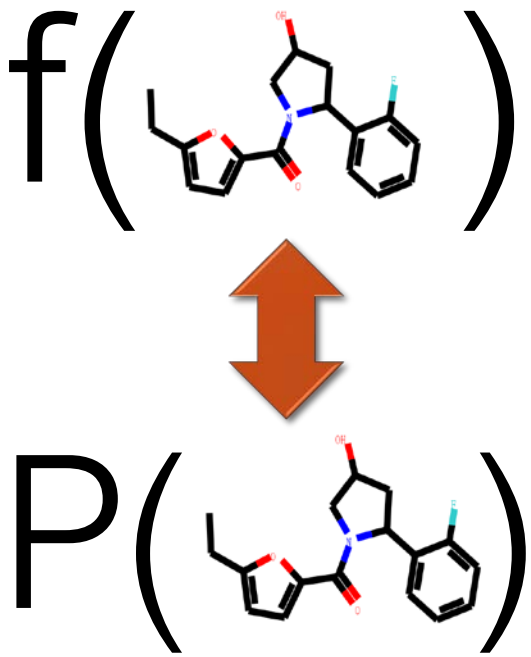


vs.



Why Is It Hard?

- **Complex molecular graph optimization task:**
  - **Graph generation**: $\mathrm{G} \sim \mathrm{P}(G)$
  - **Graph property prediction**: $f(G)$
  - **Graph optimization**: $\mathrm{G} \rightarrow \mathrm{G}'$ and maximizing $f(G') - f(G)$



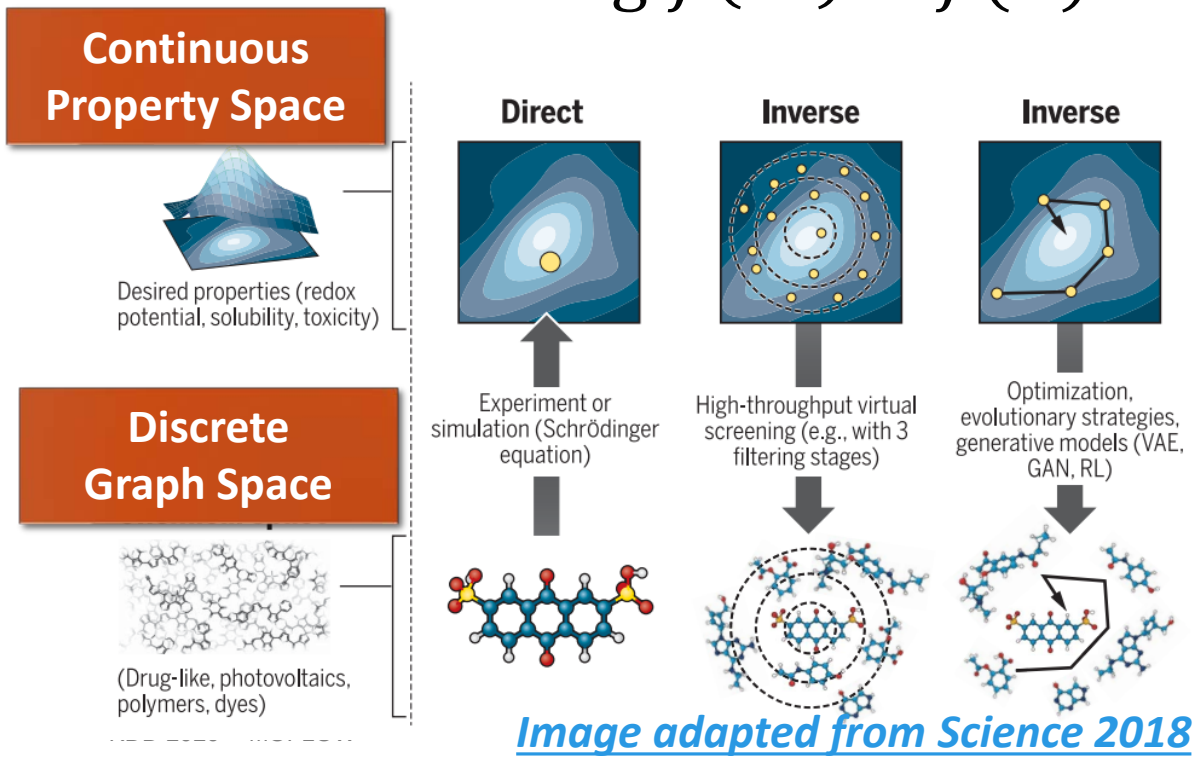
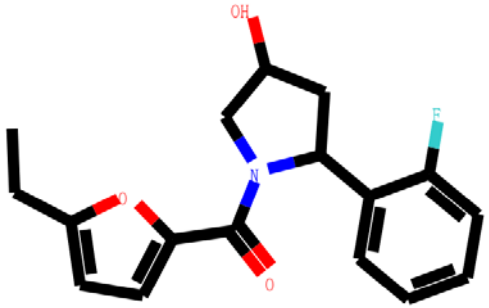


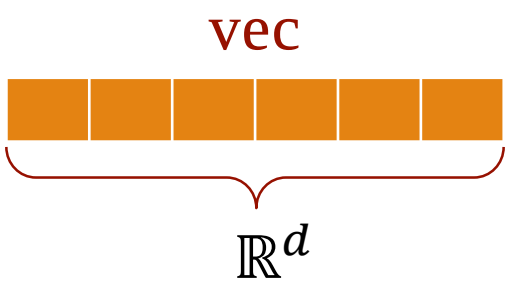
Image adapted from Science 2018

Why Is It Hard?

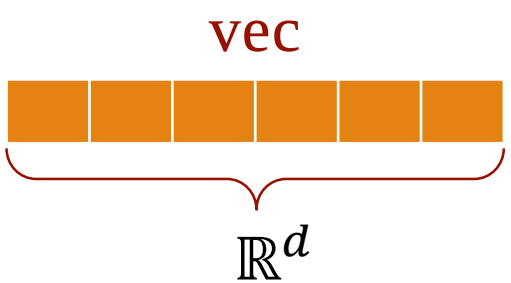
- **Encoding graph is hard, Decoding graph is much harder**
  - Encoding, embedding, inference with graph input



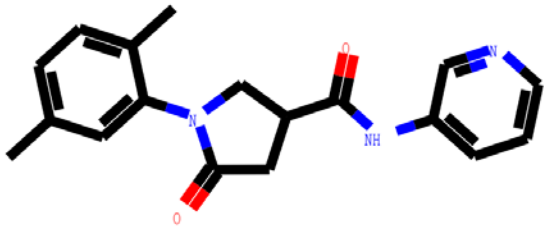
$$f: G \rightarrow \mathbb{R}^d$$



  - Decoding, generation with graph output
    - E.g. chemically valid molecular graphs with valency constraints, novel



$$f: \mathbb{R}^d \rightarrow G$$



Related Works

- **Sequence-based VAE model**
  - SMILES (Simplified molecular-input line-entry system) string
  - Grammar Variational Autoencoder (Grammar-VAE)
  - Limitation: Sequences lose structural information

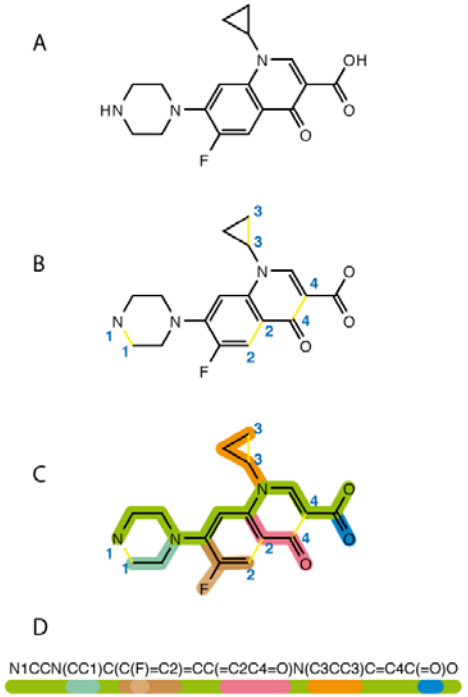


Image from https://en.wikipedia.org/wiki/Simplified_molecular-input_line-entry_system

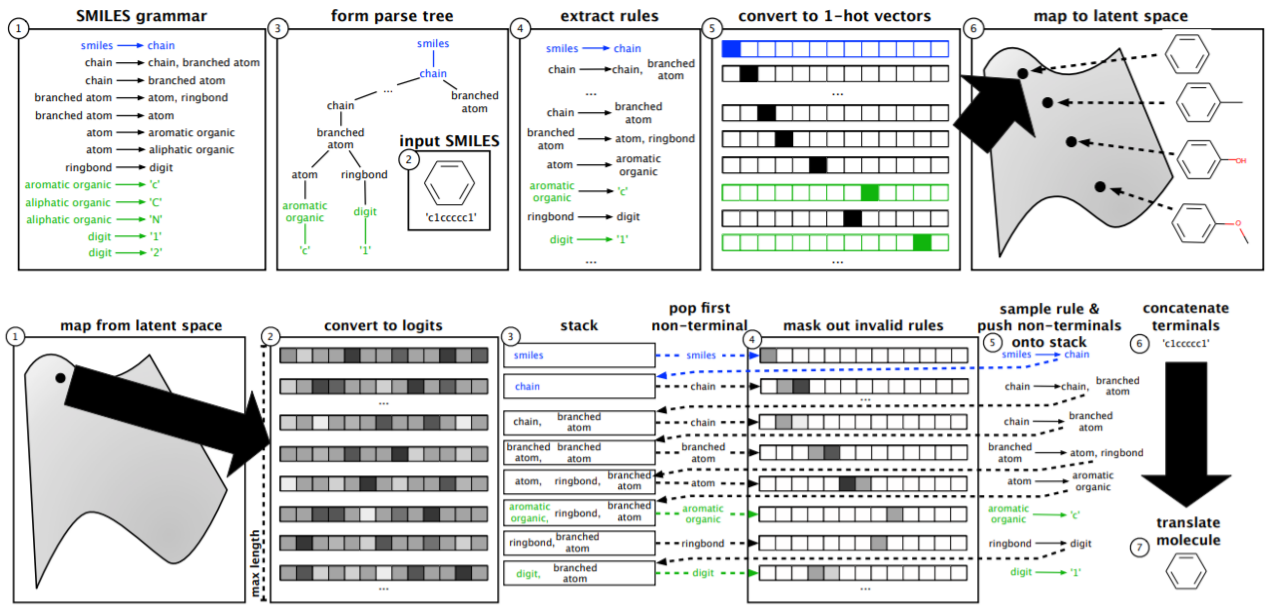
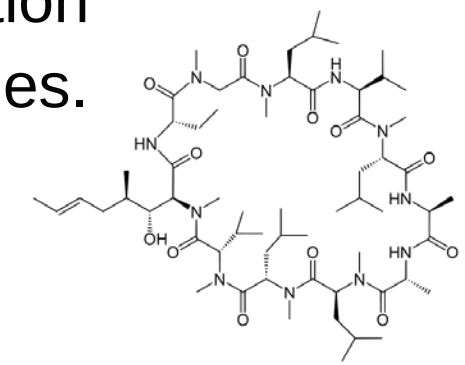


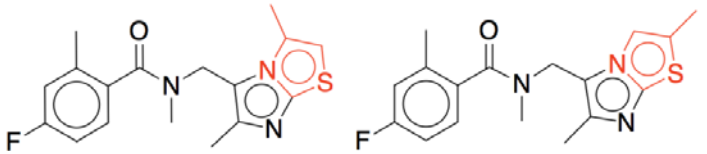
Image from: Kusner et al. 2017. Grammar Variational Autoencoder.

Related Works

- **Graph-based VAE model**
  - Structural information of molecules is better kept by graphs
    - E.g., similarity, chemical validity
  - Junction Tree Variational Autoencoder (JT-VAE)
  - Limitation
    - Expensive sampling for generation
    - Only for tree-structured molecules.
    - Ciclosporin: Large circle



https://en.wikipedia.org/wiki/Ciclosporin



Cc1cn2c(CN(C)C(=O)c3ccc(F)cc3C)c(C)nc2s1
Cc1cc(F)ccc1C(=O)N(C)Cc1c(C)nc2scc(C)n12

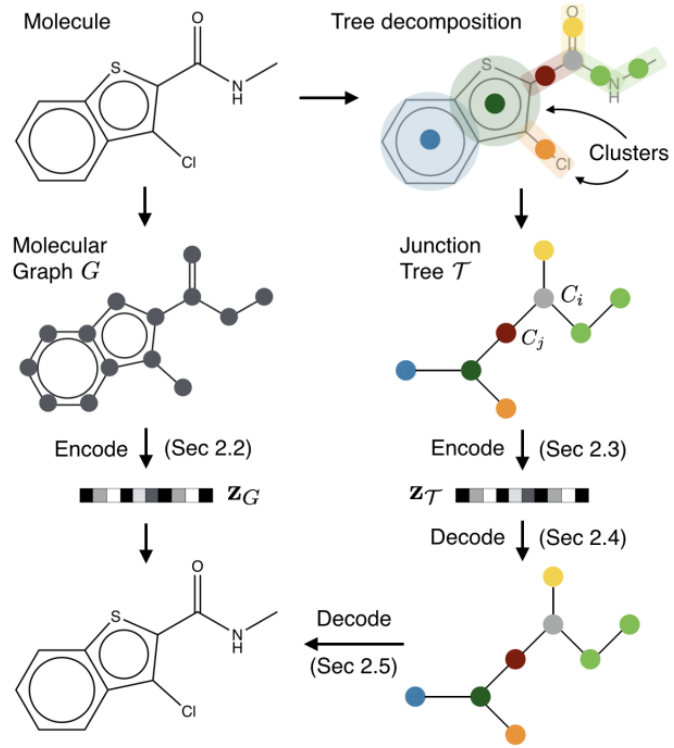


Image from: **Jin** et al. 2018. Junction Tree Variational Autoencoder for Molecular Graph Generation. *ICML*

Related Works

❑GAN-based models

- oMolecular Generative adversarial network (MolGAN)
- oLimitation
  - ❖No chemical validity guarantee; Mode collapse->tend to generate duplicated molecules → few novel molecules

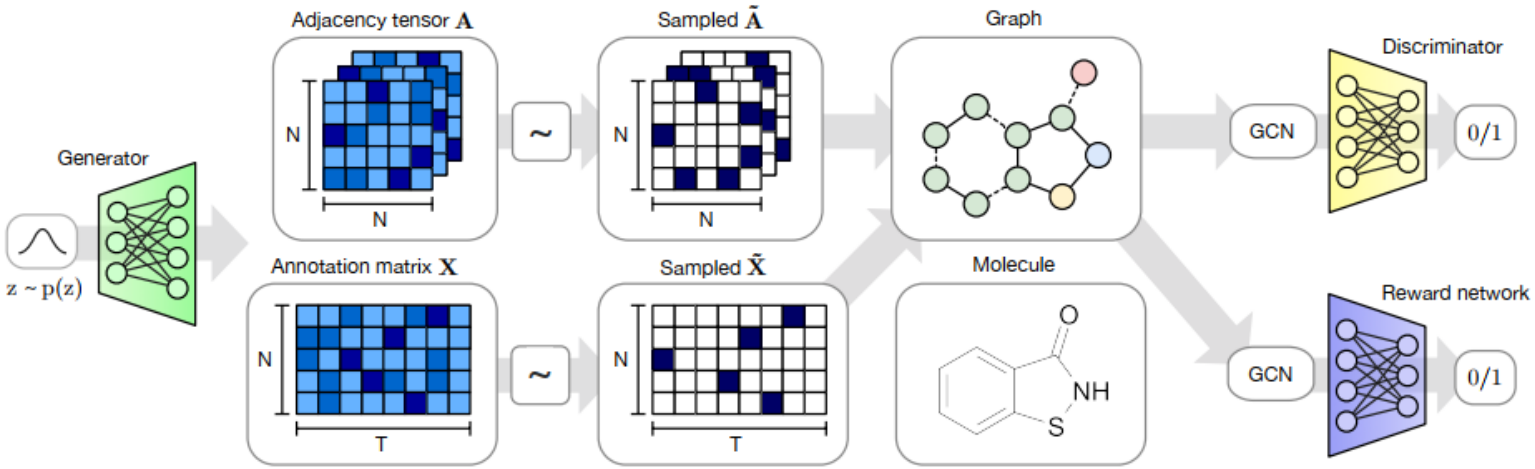
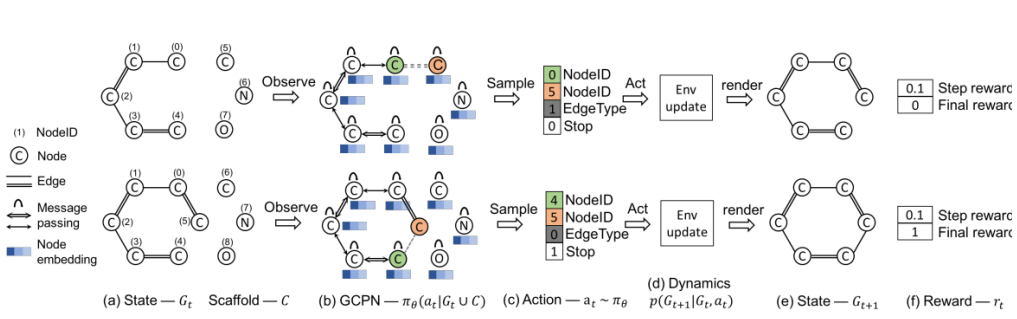
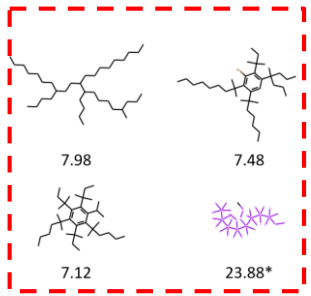


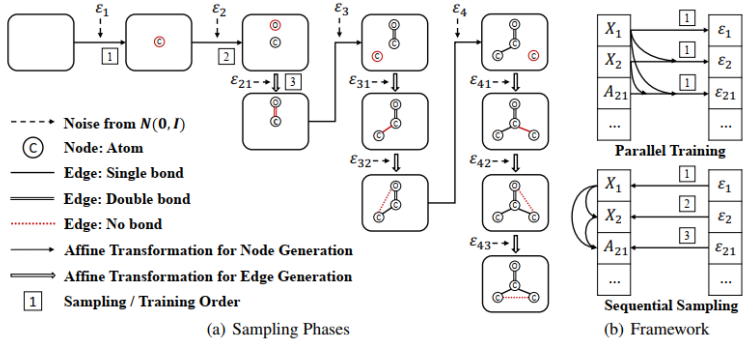
Image from: De Cao et al. 2018. MolGAN: An implicit generative model for small molecular graphs

Related Works

- **Autoregressive-based models**
  - Graph Convolutional Policy Network (GCPN)
  - Graph Autoregressive Flow model (GraphAF)
  - Reject sampling for validity + Reinforcement Learning for optimization
  - Limitations
    - Sequential generation, tend to generate long chains.







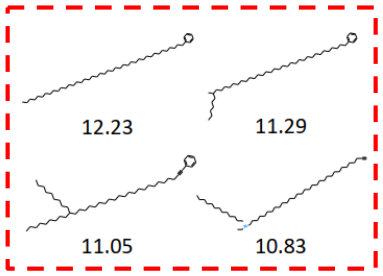


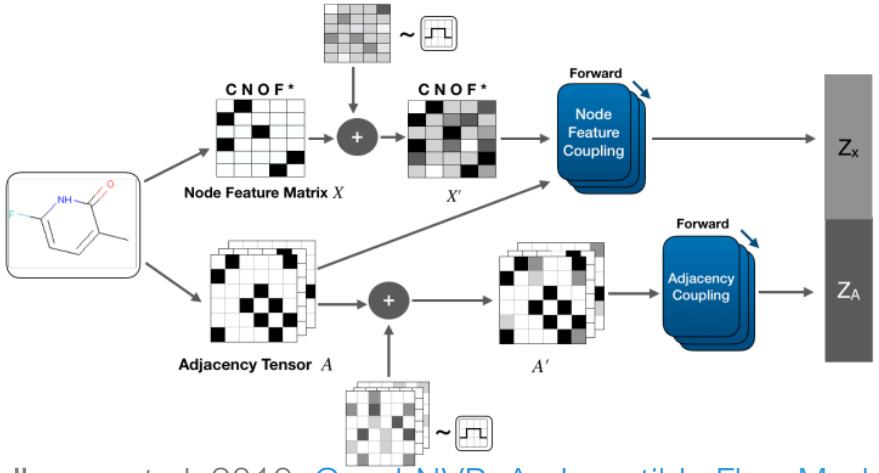
Image from: You et al. 2018. Graph Convolutional Policy Network for Goal-Directed Molecular Graph Generation. *NeurIPS*

Image from: Shi et al. 2020. GraphAF: a Flow-based Autoregressive Model for Molecular Graph Generation. *ICLR*

Related Works

❑Normalizing Flow-based models

- oGraphNVP: Graph Real-valued Non-Volume Preserving flow
  - ❖Only use add coupling
- oLimitations
  - ❖Unstable deep structures, No chemical validity guarantee, Few novel molecular graphs



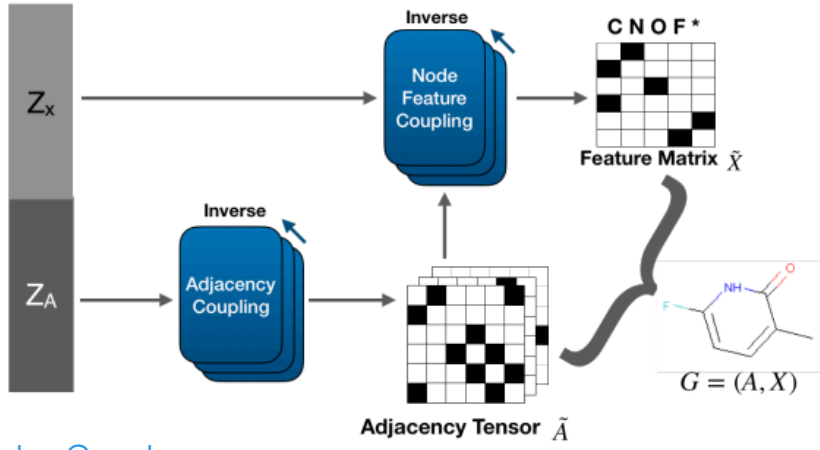


Image from: Madhawa et al. 2019. GraphNVP: An Invertible Flow Model for Generating Molecular Graphs

Related Works

- **Classified by Data:**
  - Sequence: SMILES
  - Graph: molecular graphs
- **Classified by Deep Generative Models:**
  - Autoregressive Models (AR)
  - Variational Autoencoders (VAE)
  - Generative Adversarial Networks (GAN)
  - Normalizing Flow Models (Flow)
- **Classified by Search & Optimization**
  - Gradient ascend
  - Reinforcement learning

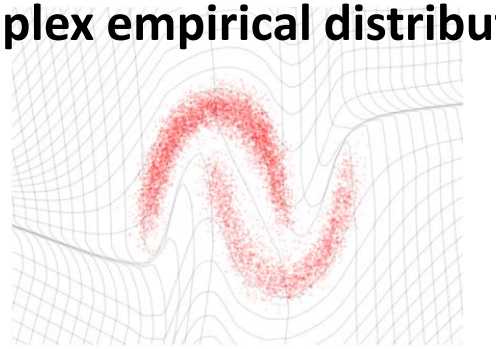
Our Choice

- **Classified by Data:**
  - Sequence: SMILES
  - **Graph: molecular graphs**
- **Classified by Deep Generative Models:**
  - Autoregressive Models (AR)
  - Variational Autoencoders (VAE)
  - Generative Adversarial Networks (GAN)
  - **Normalizing Flow Models (Flow)**
- **Classified by Search & Optimization**
  - **Gradient ascend**
  - Reinforcement learning

Basics of Normalizing Flow

- **An invertible generative model**
  - Goal: $X \sim \boldsymbol{P}(X)$
- **Inference**: $Z = f_\theta(X)$
  - From complex to simple, e.g. Z is Gaussian
- **Generation**: $X = f_\theta^{-1}(Z)$
  - Generate complex by invertible mapping
- **Exact Maximum Likelihood Training**
  - Change of variable $\log \boldsymbol{P}(X) = \log \boldsymbol{P}(Z) + \log \left| \det\left(\frac{\partial f_\theta}{\partial Z}\right) \right|$
  - $\underset{\theta}{\operatorname{argmax}} E_{M \sim P_{data}}[\log P_M(M;\theta)]$
- **Constraints of network structures:**
  - f_θ: invertible DNNs, each layer is invertible
  - Computing $\det\left(\frac{\partial f_\theta}{\partial Z}\right)$ should be efficient

$\boldsymbol{P}(\boldsymbol{X})$:
Complex empirical distribution



Inference ↓ ↑ **Generation**

$\boldsymbol{P}(\boldsymbol{Z})$:
Simple latent distribution

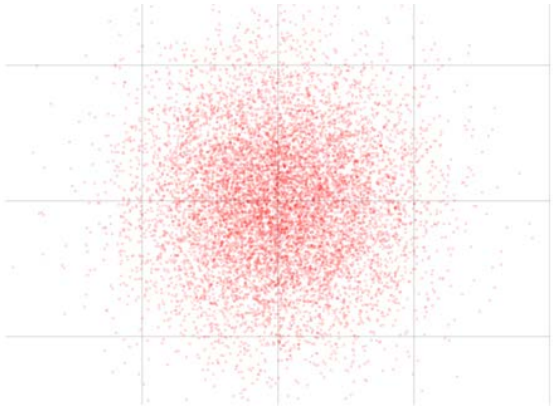
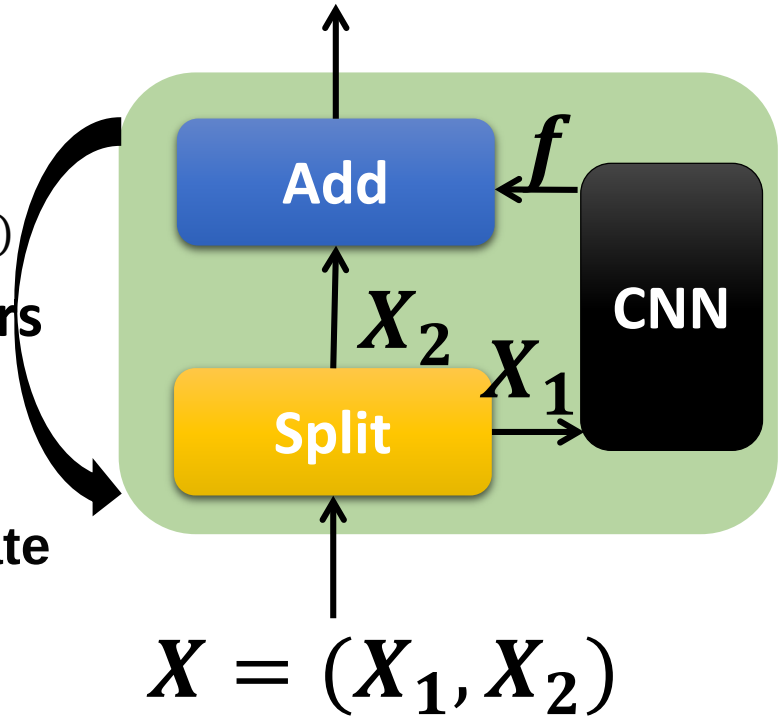


Image from: **Dinh** et al. 2017. Density Estimation using Real NVP. *ICLR*.

Related works: NICE Model

- **NICE: Non-linear Independent Components Estimation**
- **Invertible layers: splitting dimensions + residual flow updated alternately**
- **Split:**
  - $\mathbf{X} = (X_1, X_2)$
  - $\mathbf{Z} = (Z_1, Z_2)$
- **Add:**
  - $Z_1 = X_1$ (**Save information for reversing**)
  - $Z_2 = X_2 + f_\theta(X_1)$ (**Residual**)
  - **Reverse mapping:**
    - $X_1 = Z_1$
    - $X_2 = Z_2 - f_\theta(Z_1)$
- **Deep: Next layer by alternating update**
  - $Z_1 = X_1 + f_\theta(X_2)$ **(Residual)**
  - $Z_2 = X_2$ **(save information for reverse)**
- **…**

$$Z = (Z_1 = X_1, Z_2 = f(X_2, X_1))$$



Hamiltonian Systems

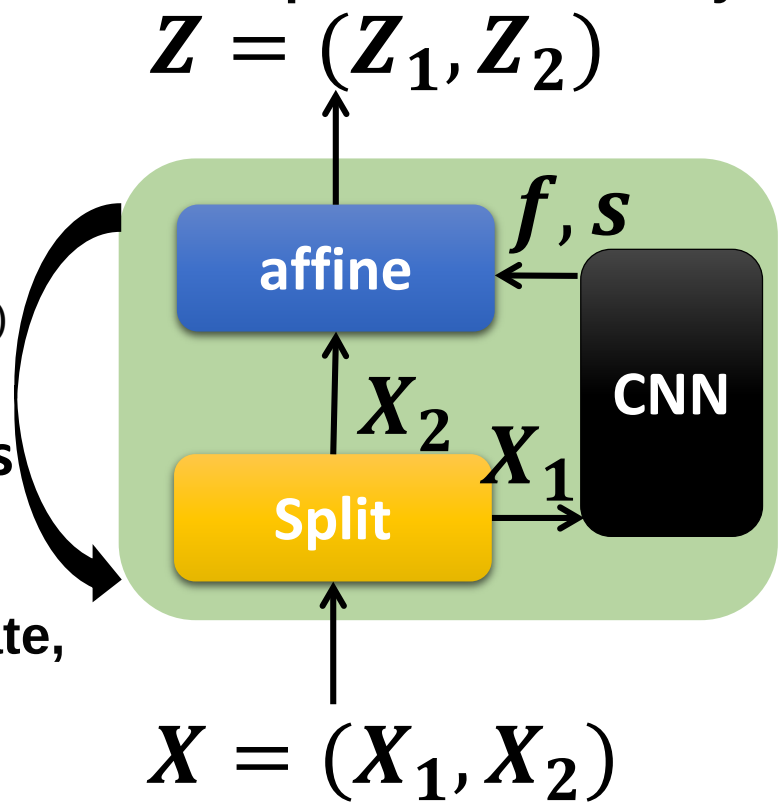
$$\begin{bmatrix} \frac{dZ_1}{dt} \\ \frac{dZ_2}{dt} \end{bmatrix} = \begin{bmatrix} f_\theta(Z_2) \\ f_\theta(Z_1) \end{bmatrix}$$

Dinh et al. 2014. Nice: Non-linear independent components estimation
Dinh et al. 2017. Density Estimation using Real NVP. *ICLR*.

Chen et al. 2019. Neural Ordinary Differential Equations. *NeurIPS*.

Related works: RealNVP Model

- **RealNVP: Real-valued Non-Volume Preserving flow**
- **Invertible layers: splitting dimensions + affine updated alternately**
- **Split:**
  - $\mathbf{X} = (X_1, X_2)$
  - $\mathbf{Z} = (Z_1, Z_2)$
- **Affine:**
  - $Z_1 = X_1$ (save information for reversing)
  - $Z_2 = X_2 e^{s_\theta(X_1)} + f_\theta(X_1)$ (affine)
  - Reverse mapping:
    - $X_1 = Z_1$
    - $X_2 = e^{-s_\theta(X_1)}[Z_2 - f_\theta(Z_1)]$
- **Deep: Next layer by alternating update,**
  - $Z_1 = X_1 e^{s_\theta(X_2)} + f_\theta(X_2)$ (Residual)
  - $Z_2 = X_2$ (save information for reverse)
- **…**



Hamiltonian Systems

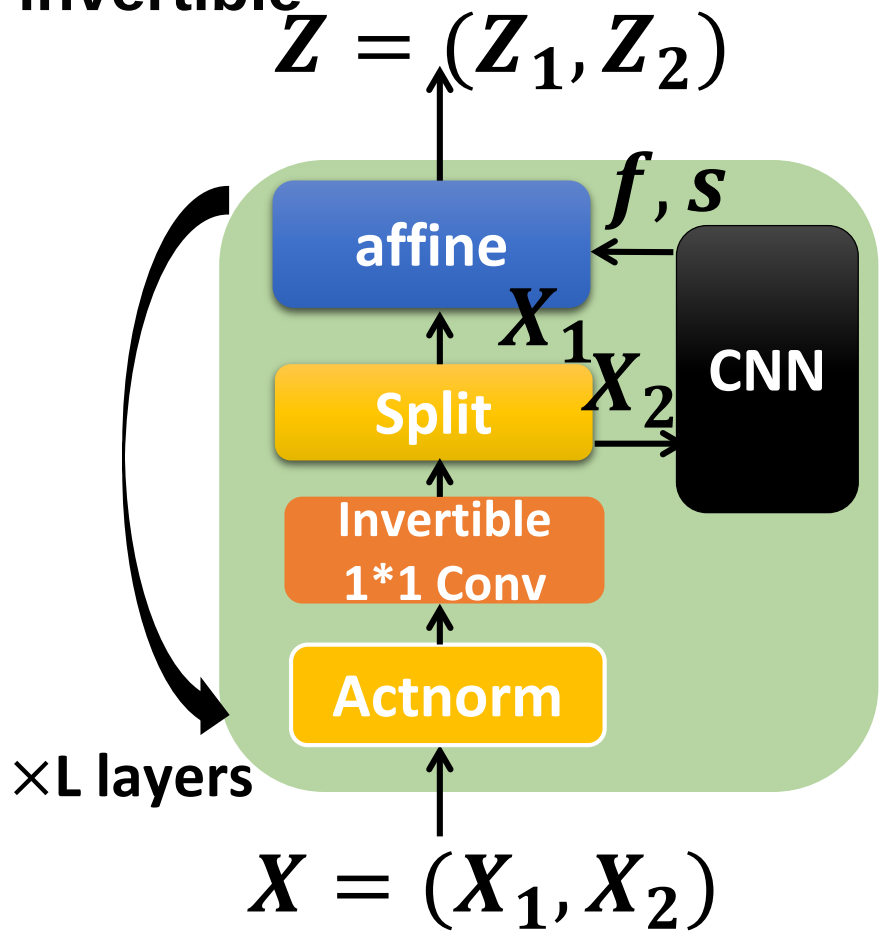
$$\begin{bmatrix} \frac{dZ_1}{dt} \\ \frac{dZ_2}{dt} \end{bmatrix} = \begin{bmatrix} f_\theta(Z_2) \\ f_\theta(Z_1) \end{bmatrix}$$

Dinh et al. 2014. Nice: Non-linear independent components estimation
Dinh et al. 2017. Density Estimation using Real NVP. *ICLR*.

Chen et al. 2019. Neural Ordinary Differential Equations. *NeurIPS*.

Related works: Glow Model

- **Glow: Generative flow with invertible 1*1 convolutions**
- **Actnorm:**
  - Stable dynamics
  - $B = \frac{B-\mu}{\sqrt{\sigma^2+\epsilon}}$ each channel over batch
  - invertible
- **Invertible 1*1 convolution:**
  - Expressive power
  - $\mathbb{R}^{c\times n\times n} \times \mathbb{R}^{c\times c} \rightarrow \mathbb{R}^{c\times n\times n}$
- **Affine:**
  - $Z_1 = X_1$
  - $Z_2 = X_2 e^{s_\theta(X_1)} + f_\theta(X_1)$



Hamiltonian Systems

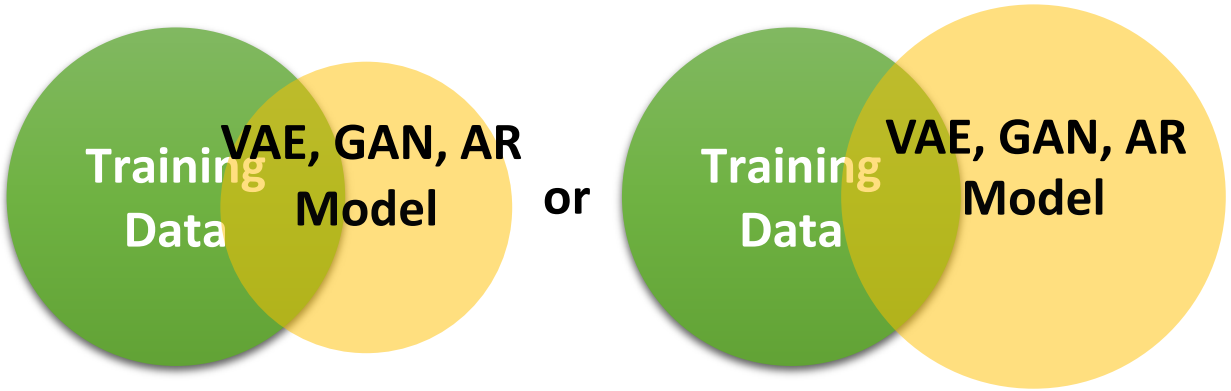
$$\begin{bmatrix} \frac{dZ_1}{dt} \\ \frac{dZ_2}{dt} \end{bmatrix} = \begin{bmatrix} f_\theta(Z_2) \\ f_\theta(Z_1) \end{bmatrix}$$

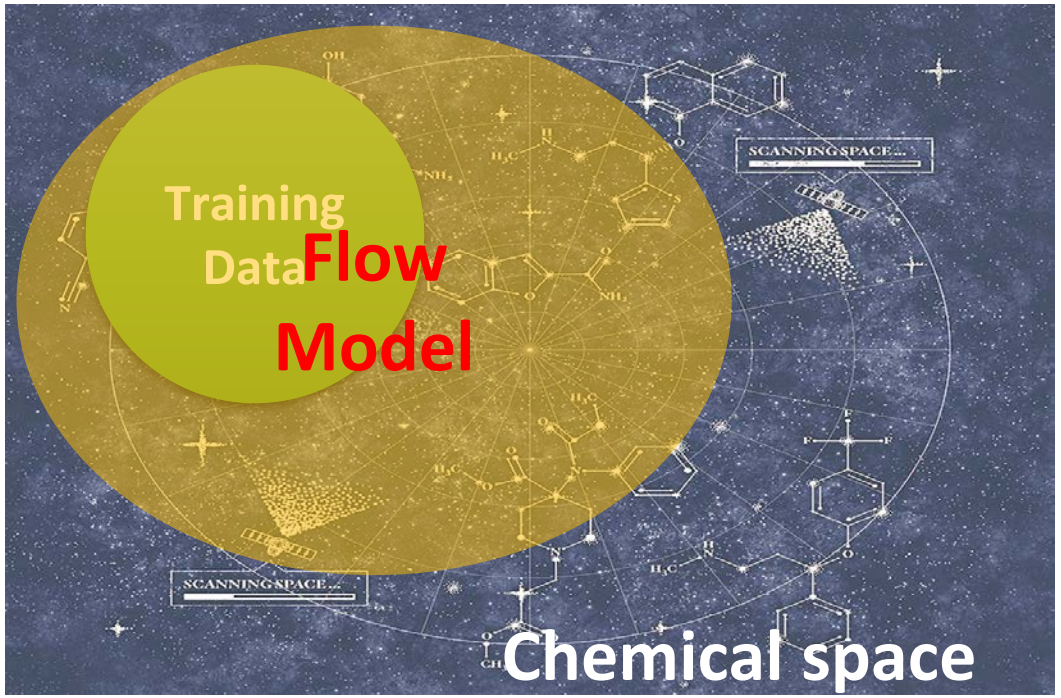
Kingma et al. 2018. Glow: Generative flow with invertible 1x1 convolutions. *NeurIPS*.

Chen et al. 2019. Neural Ordinary Differential Equations. *NeurIPS*.

Why Flow Frameworks

- **Invertible mappings**
  - Potentials to generate **more novel** molecules
  - VAE, GAN, AR are not invertible, see diagrams below
  - Flow learns a strict superset and explores chemical space better





Why Flow Frameworks

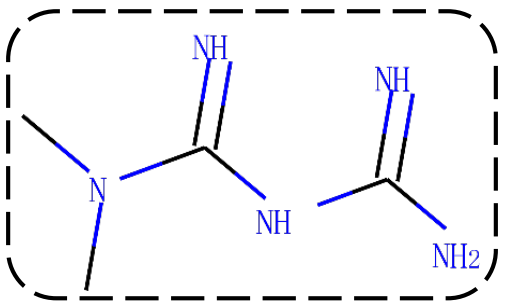
- **Exact maximum likelihood training**
  - VAE,GAN are not
- **Efficient one-shot inference and generation**
  - Capturing molecular structures in a holistic way v.s. AR's step-by-step way.
- **Better performance shown later**

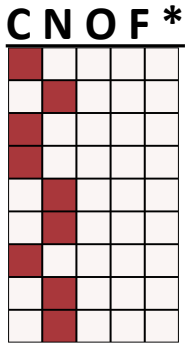
Idea of our MoFlow

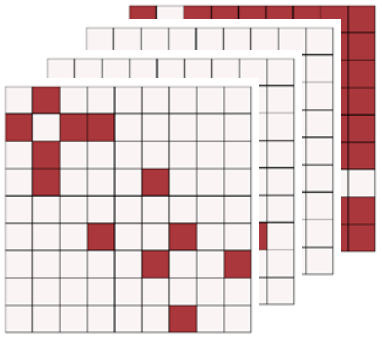
- **Molecular Graph:** **M**olecule = (**A**tom, **B**ond)
  - Atoms → Nodes, **A**tom ∈ $\{0,1\}^{n\times k}$, n Nodes in k (atom) types
  - Bonds → Edges, **B**ond ∈ $\{0,1\}^{c\times n\times n}$, Edges in c (bond) types

A one-hot atom matrix **A multi-channel tensor**

$\boldsymbol{M}$ = ($\boldsymbol{A}$, $\boldsymbol{B}$)



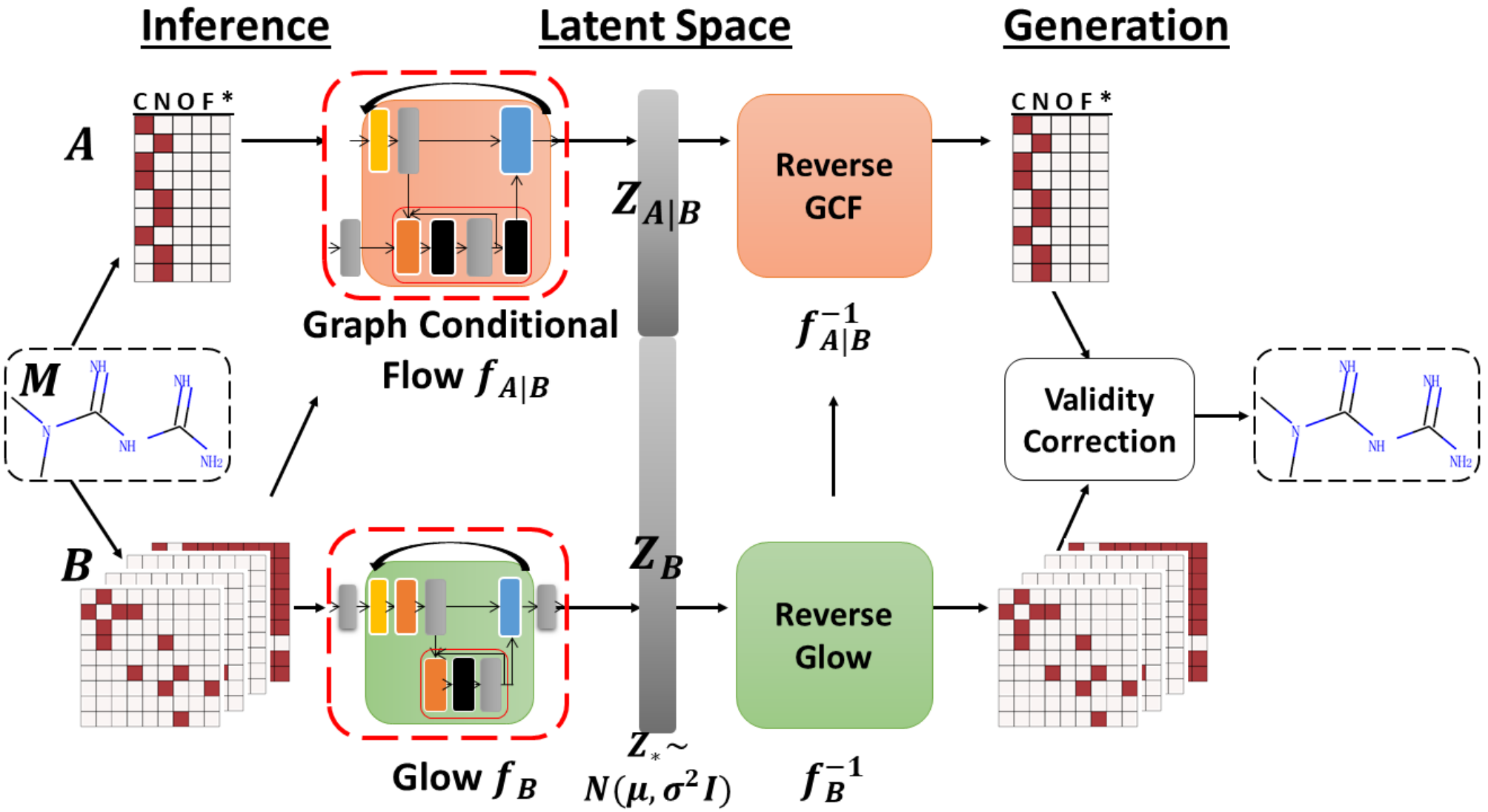




Idea of our MoFlow

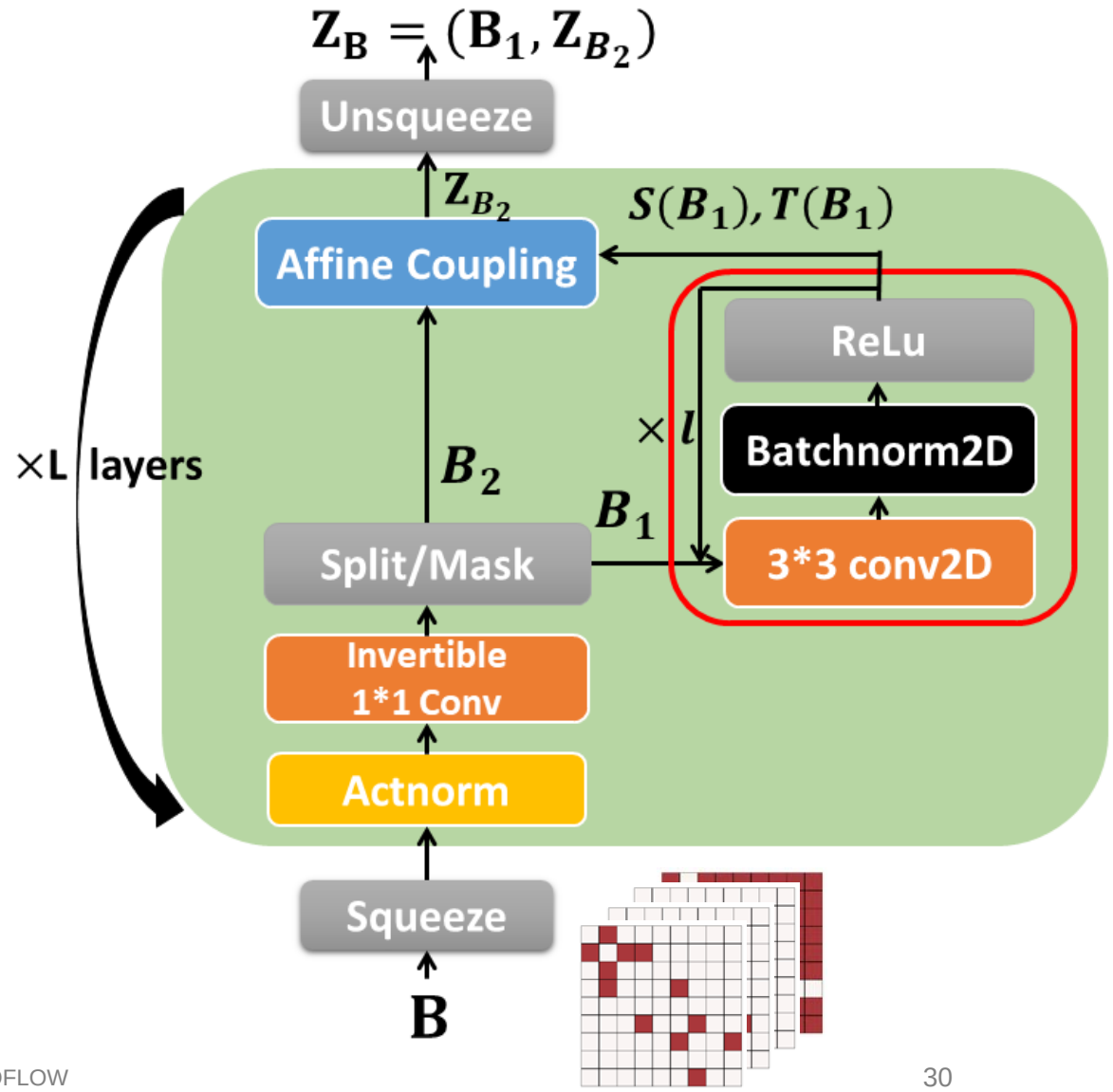
- **MoFlow**:
  - **M**olecule=(**A**tom, **B**ond) **How to model discrete atom-bond structures of molecule?**
  - $P_M(M) = P_M\big((A,B)\big) \approx P_{A|B}(A|B)P_B(B)$
  - Any flow model $f_B(\mathrm{B})$ for bonds $P_B(B)$
    - Generating graph skeleton by $P_B(B)$
  - **Graph conditional flow** $f_{A|B}(A|B)$ for atoms given bonds $P_{A|B}(A|B)$
    - Generating nodes given graph skeleton by $P_{A|B}(A|B)$
  - Assembling atom and bonds with validity correction

The Generative Framework



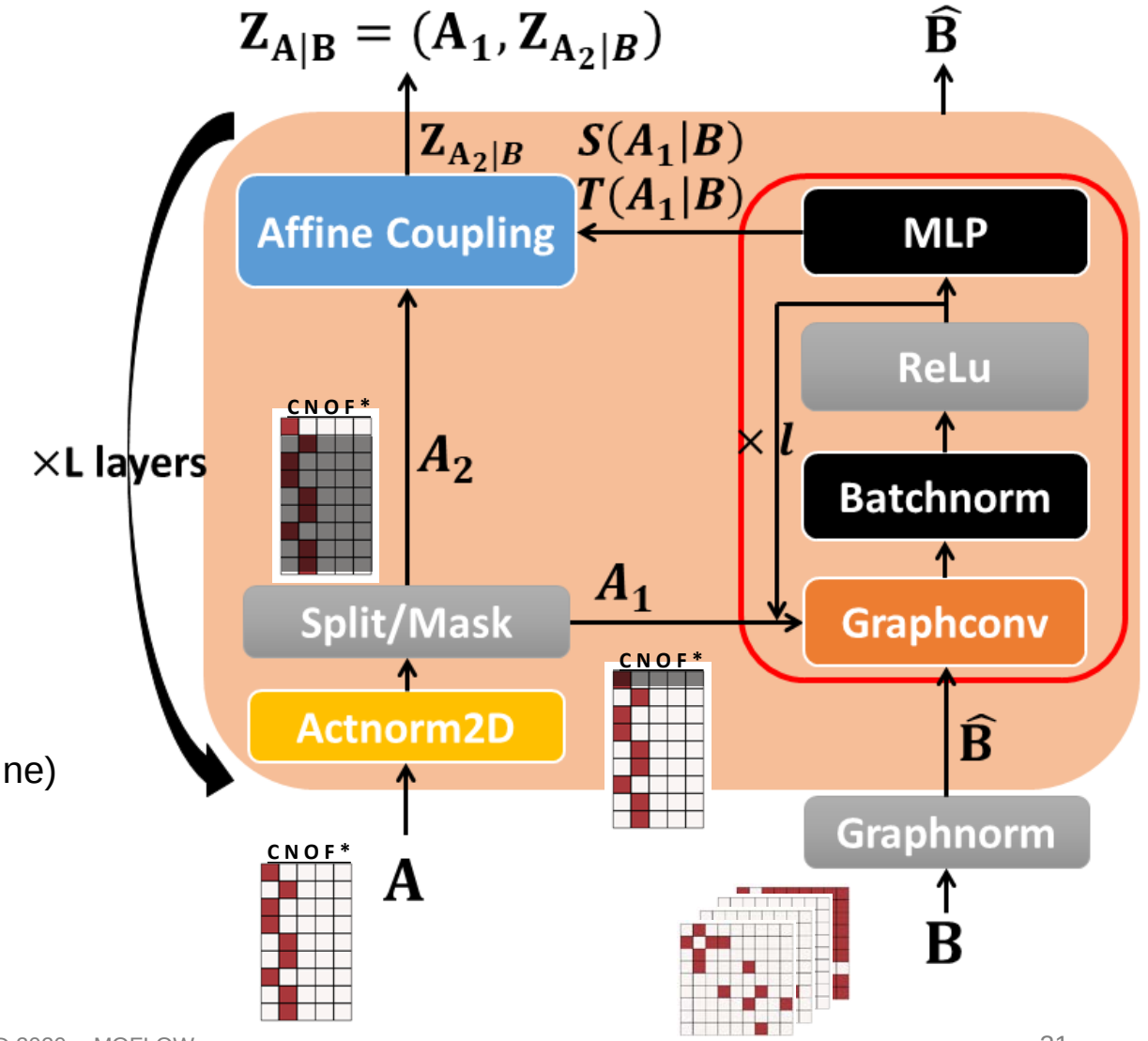
A variant of Glow for Bond/Edge

- **Squeeze**
  - $X \in \mathbb{R}^{c\times n\times n} \rightarrow \mathbb{R}^{ck^2\times\frac{n}{k}\times\frac{n}{k}}$
- **Actnorm:**
  - Stable dynamics
  - $B = \frac{B-\mu}{\sqrt{\sigma^2+\epsilon}}$ each channel over batch
- **Invertible 1*1 convolution:**
  - Expressive power
  - $\mathbb{R}^{c\times n\times n} \times \mathbb{R}^{c\times c} \rightarrow \mathbb{R}^{c\times n\times n}$
- **Split:**
  - Discretization of Hamiltonian system
  - $\mathbf{B} = (\boldsymbol{B_1}, \boldsymbol{B_2})$
  - $\mathbf{Z} = (\mathbf{Z_{B1}}, \mathbf{Z_{B2}})$
- **Affine coupling:**
  - Stable (batchnorm2D, Sigmoid) and expressive power (Affine)
  - $\boldsymbol{Z_{B1}} = \boldsymbol{B_1}$
  - $\boldsymbol{Z_{B2}} = \boldsymbol{B_2} \odot \boldsymbol{Sigmoid}(\boldsymbol{S_\theta}(\boldsymbol{B_1})) + \boldsymbol{T_\theta}(\boldsymbol{B_1})$
- **Deep: alternating update in next layer**

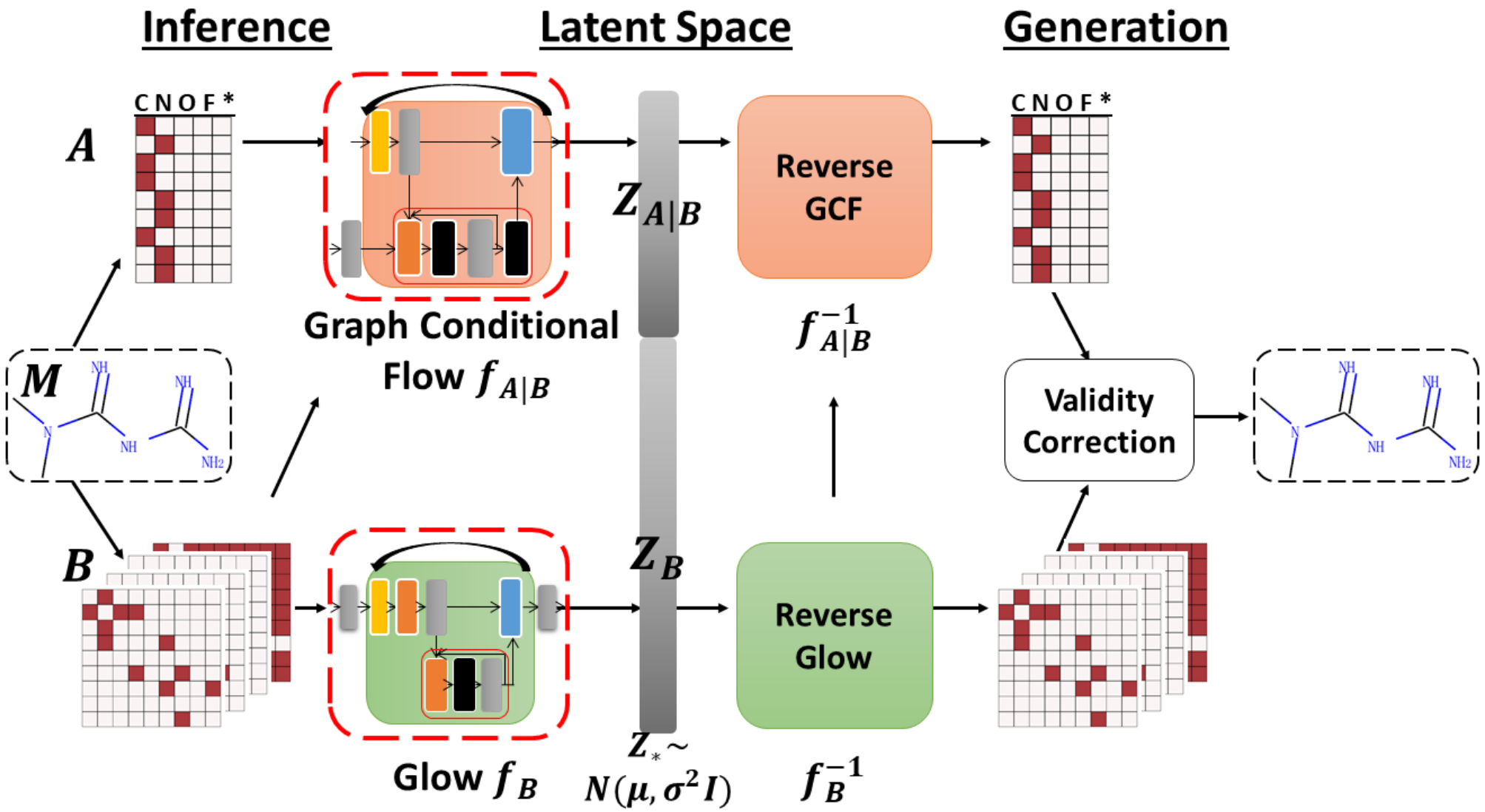


Graph Conditional Flow For Atoms Given Bonds

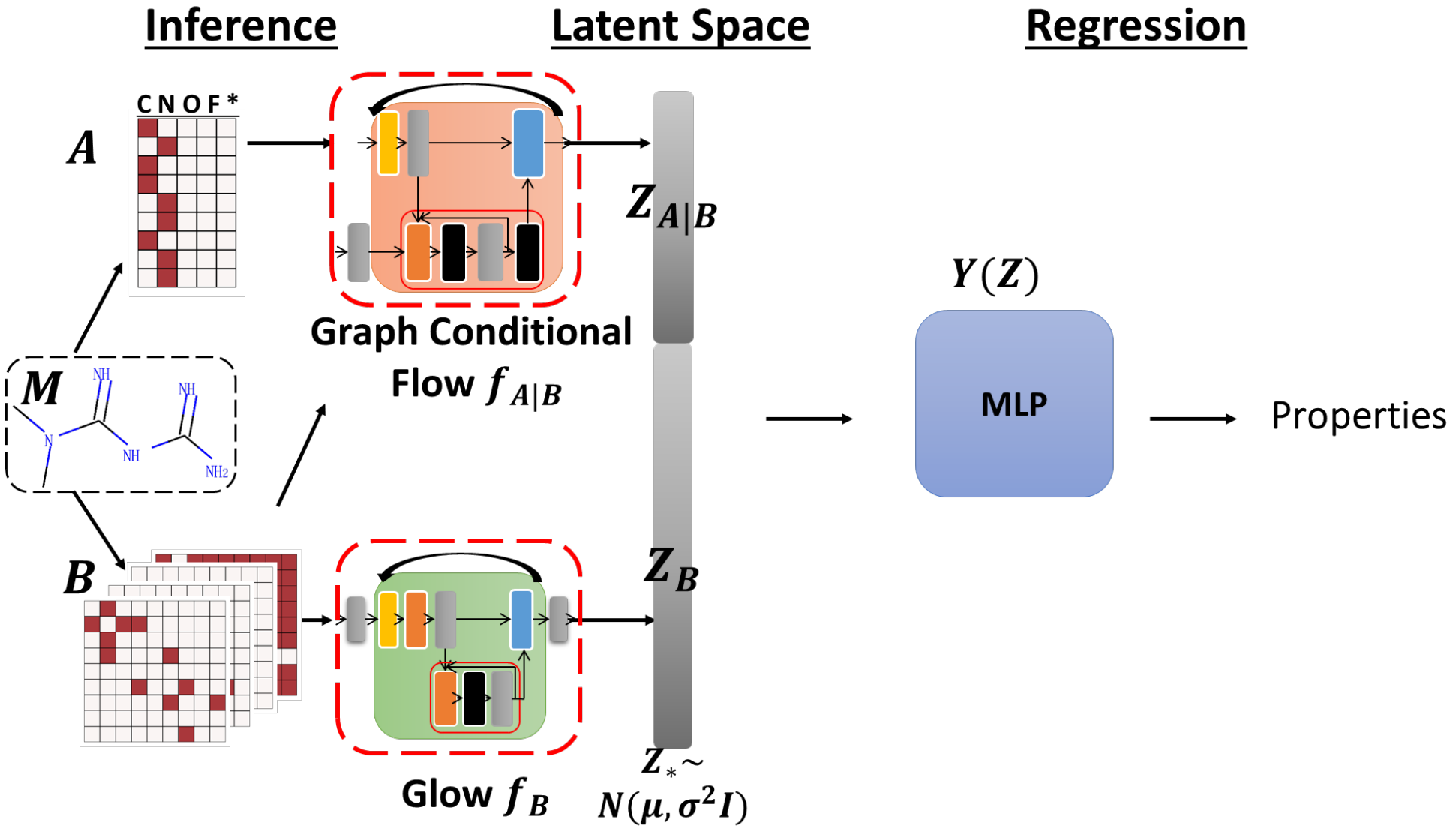
- **Actnorm2D:**
  - Stable dynamics
  - $B = \frac{B-\mu}{\sqrt{\sigma^2+\epsilon}}$ each row over batch
- **Split:**
  - Discretization of Hamiltonian system on Graphs
  - A= (A_1, A_2) by each row
  - $\mathrm{Z} = (\mathrm{Z_{A1|B}}, \mathrm{Z_{A2|B}})$
- **Graphnorm**
  - $\widehat{B}_i = D^{-1}B_i$, $D = \sum_{c,i} B_{c,i,j}$ in-degree over all channels
- **GraphConv(A|B), multi-channel**
  - $\sum_{i=1}^{c} \widehat{\boldsymbol{B}}_{\boldsymbol{i}}(\boldsymbol{M} \odot \boldsymbol{A})\,\mathbf{W_i} + (\mathbf{M} \odot \boldsymbol{A})\mathbf{W_0}$
  - update each row by the remaining rows
- **Affine coupling:**
  - Stable (batchnorm, Sigmoid) and expressive power (Affine)
  - $Z_{A1|B} = A_1$
  - $Z_{A2|B} = A_2 \odot Sigmoid(S_\theta(A_1|B)) + T_\theta(A_1|B)$
- **Deep: alternating update in next layer**



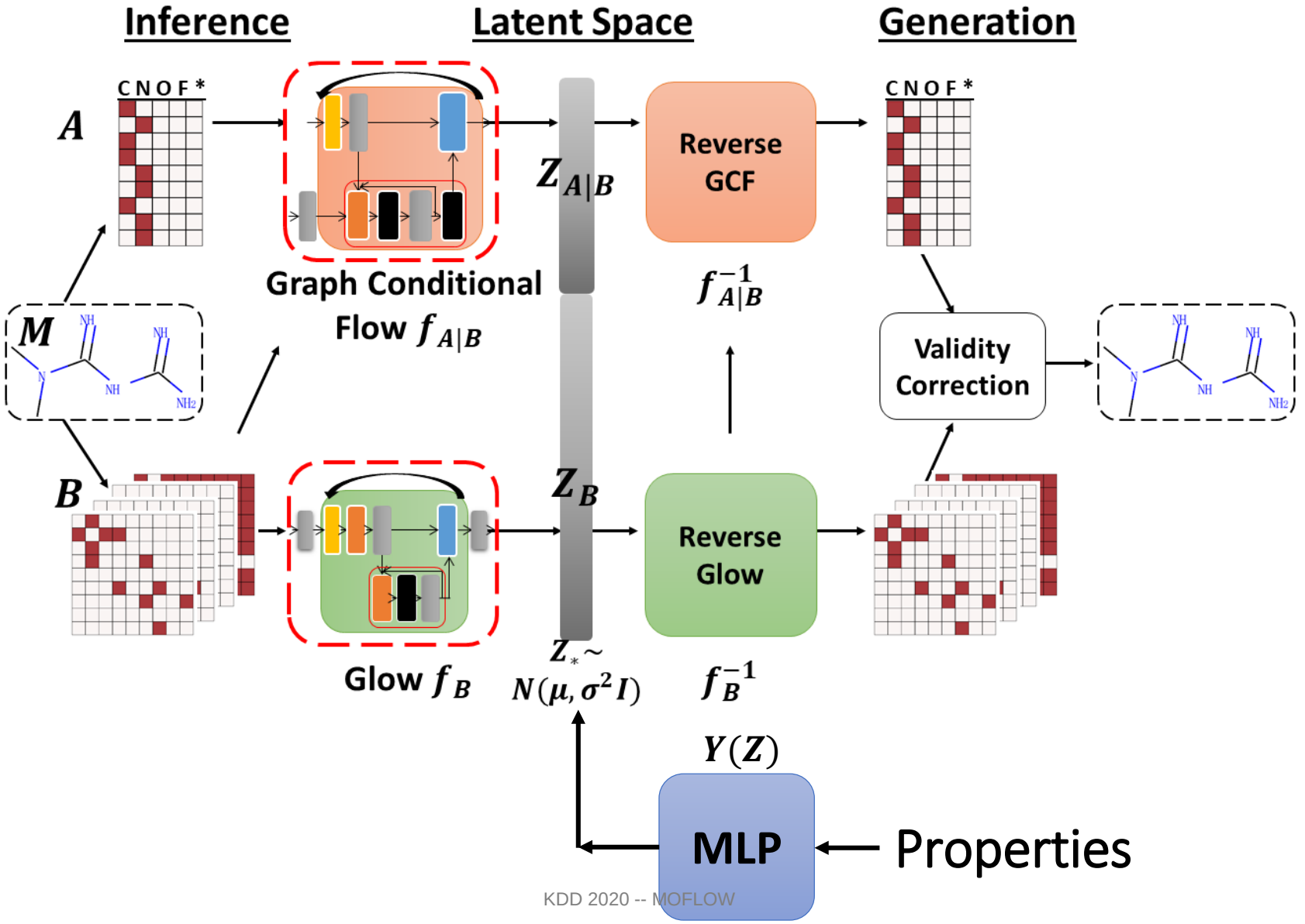
Molecular Graph Generation



Graph Property Prediction



Molecular Graph Optimization



Validity Correction

- **Valid molecules: valency constraints**
  - $\sum_{c,j} \mathrm{c} * B(c,i,j) \leq Valency(Atom_i) + Formal\_Charge$
  - C: 4, O:2, O+:3
- **Validity Correction**
  - While checking valency constraints:
    - if follows constraints:
      - Return the greatest connected component
    - else:
      - Delete unnecessary bond or add charge to invalid atoms according to chemical rules

Experiments

1. Molecular Generation & Reconstruction
2. Visualization of Continuous Latent Space
3. Property Optimization
4. Constrained Property Optimization

EXP1: Molecular Generation & Reconstruction

❑The Problem:

- Input: $\{G_1, G_2, \ldots\}$ molecular graphs
- Model
  - ❖Learned molecular generative model $\mathrm{P_M}$, and its invertible mapping f
  - ❖Generation: $G = f^{-1}(Z)$, where Z follows isotropic Gaussian
  - ❖Reconstruction: $G = f^{-1}(Z)$ where $Z = f(G)$
- **Goal**: To generate **valid & unique & novel** molecular graphs

❑Datasets:

-

	#Graphs	#Nodes	#Node/Atom Types	#Edge/Bond Types
QM9	134K	9	4	3
ZINC	250K	38	9	3

EXP1: Molecular Generation & Reconstruction

❑Evaluation metrics:

1. **Validity**: %chemically valid molecules in all the generated molecules
2. **Validity without check/correction**
3. **Uniqueness**: %chemically valid and unique molecules in all the generated molecules
4. **Novelty**: %generated valid molecules not in training dataset
5. **Reconstruction rate**: % training dataset which can be reconstructed from their latent representations
6. **N.U.V.:** %novel, unique and valid molecules in all the generated molecules

EXP1: Molecular Generation & Reconstruction

- **More novel & unique & valid molecules**
- **100% Reconstruction**
  - Strict superset of training dataset
- **Better validity without check**
  - Than AR models. One-shot models, a holistic way
- **Our MoFlow explores the big chemical space further and better!**

Table 1: Generative performance on QM9

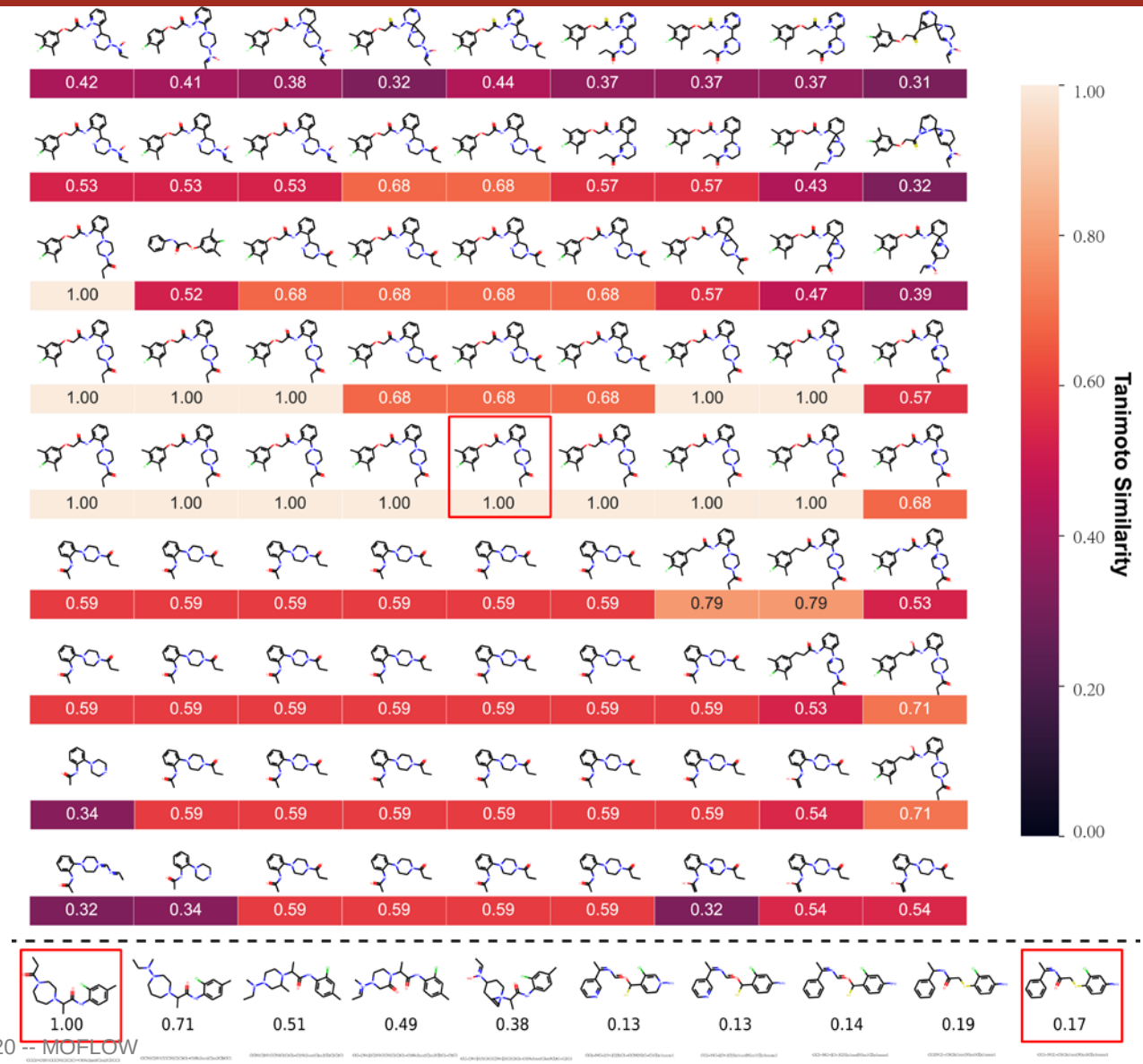
	% Validity	% Validity w/o check	% Uniqueness	% Novelty	% N.U.V.	% Reconstruct
GraphNVP	83.1 ± 0.5	-	99.2 ± 0.3	58.2 ± 1.9	47.97	100
GRF	84.5 ± 0.70	-	66.0 ± 1.15	58.6 ± 0.82	32.68	100
GraphAF	100	67	94.51	88.83	83.95	100
MoFlow	**100.00 ± 0.00**	**95.74 ± 0.65**	**99.48 ± 0.33**	**98.69 ± 0.39**	**98.18 ± 0.53**	**100.00 ± 0.00**

Table 2: Generative performance on Zinc250k

	% Validity	% Validity w/o check	% Uniqueness	% Novelty	% N.U.V.	% Reconstruct
JT-VAE	100	-	100	100	100	76.7
GCPN	100	20	99.97	100	99.97	-
MRNN	100	65	99.89	100	99.89	-
GraphNVP	42.6 ± 1.6	-	94.8 ± 0.6	100	40.38	100
GRF	73.4 ± 0.62	-	53.7 ± 2.13	100	39.42	100
GraphAF	100	68	99.10	100	99.10	100
MoFlow	**100.00 ± 0.00**	**81.94 ± 0.45**	**99.94 ± 0.05**	**100.00 ± 0.00**	**99.94 ± 0.05**	**100.00 ± 0.00**

EXP2: Visualization of latent space

- **Encode & decode between discrete graph space and continuous latent space!**
  - **Grid interpolation** around the latent representation of one molecular graph, and decode its neighbors
    - Smooth latent space ←→ Similar graph structures (Tanimoto similarity)
  - **Linear interpolation** between two molecules
    - Changing trajectory from one graph to another one.



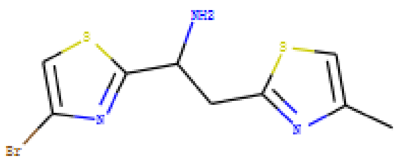
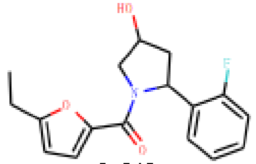
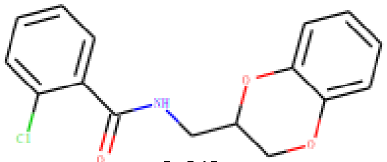
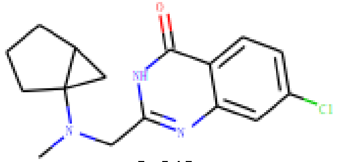
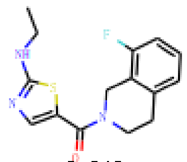
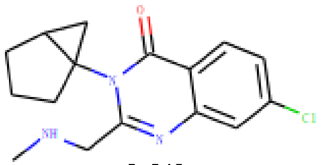
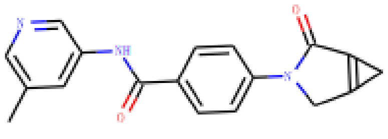
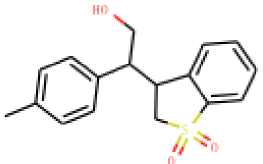
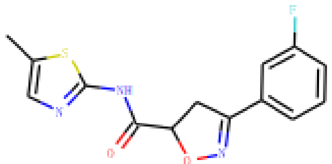
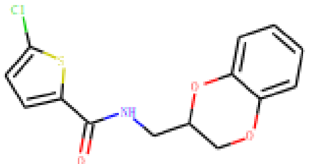
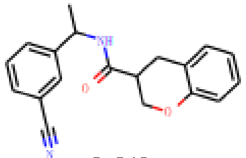
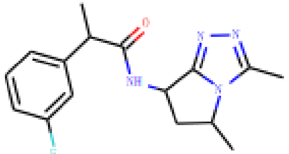
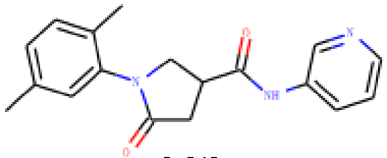
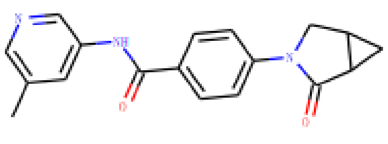
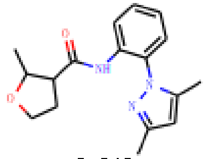
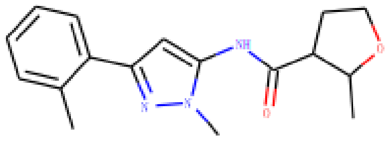
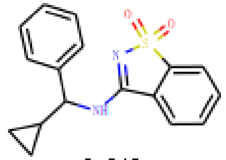
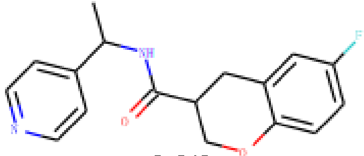
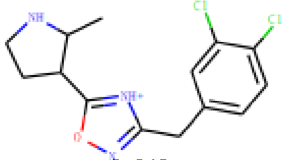
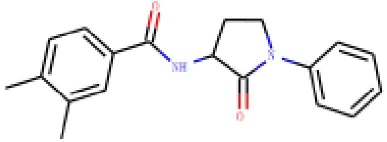
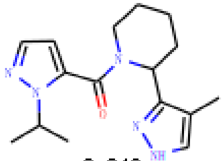
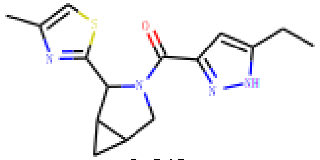
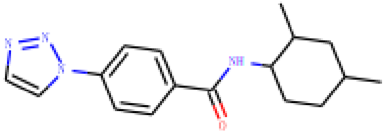
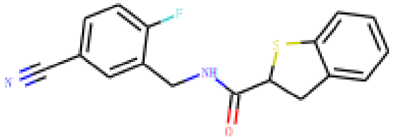
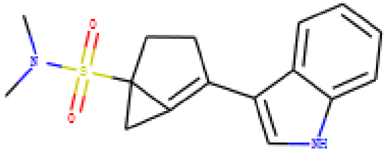
EXP3: Property Optimization

- **To Generate Novel Molecules with the best Quantitative Estimate of Druglikeness (QED) scores as many as possible**
  - Searching latent space by gradient ascend
- **Our MoFlow generates more novel molecules with top QED scores!**

Table 3: Discovered novel molecules with top QED score. Our MoFlow finds more molecules with the best QED score. More results in

Method	1st	2nd	3rd	4th
ZINC (Dataset)	0.948	0.948	0.948	0.948
JT-VAE	0.925	0.911	0.910	-
GCPN	0.948	0.947	0.946	-
MRNN	0.948	0.948	0.947	-
GraphAF	0.948	0.948	0.947	0.946
MoFlow	**0.948**	**0.948**	**0.98**	**0.948**

EXP3: Property Optimization

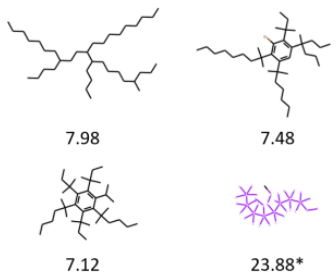

0.948

0.948

0.948

0.948

0.948

0.948

0.948

0.948

0.948

0.948

0.948

0.948

0.948

0.948

0.948

0.948

0.948

0.948

0.948

0.948

0.948

0.948

0.948

0.948

0.948

EXP4: Constrained Property Optimization

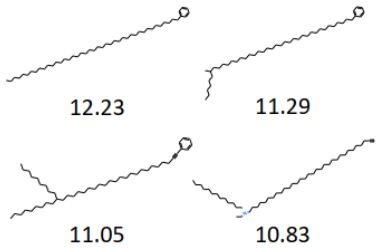
- **Find a new molecular graph $\mathbf{G}'$ from a seed molecular graph $\mathbf{G}$**
  - To maximize: similarity($\mathbf{G}$, $\mathbf{G}'$) and property $\mathrm{Y}(\mathbf{G}') - \mathrm{Y}(\mathbf{G})$
    - Tanimoto similarity of Morgan fingerprint
    - Target property Y: penalized logP (plogP), which is the octanol-water partition coefficients (logP) penalized by the synthetic accessibility (SA) score and number of long cycles.

EXP4: Constrained Property Optimization

- **Best similarity**
- **Second best improvement**
- **More realistic**
  - AR+RL model tends to generate long chains



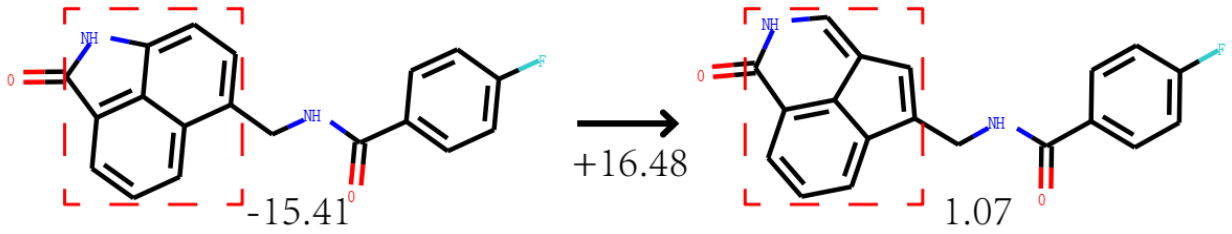
GCPN



GraphAF

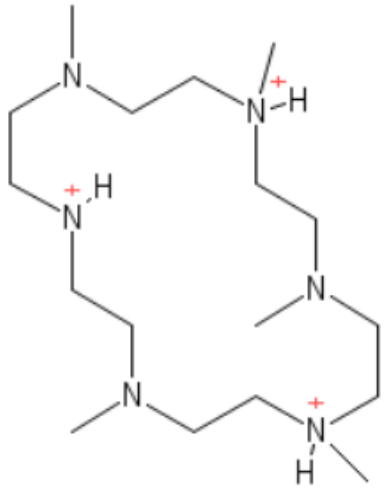
Table 4: Constrained optimization on Penalized-logP

	JT-VAE			GCPN		
δ	**Improvement**	**Similarity**	**Success**	**Improvement**	**Similarity**	**Success**
0.0	1.91 ± 2.04	0.28 ± 0.15	97.5%	4.20 ± 1.28	**0.32 ± 0.12**	100%
0.2	1.68 ± 1.85	0.33 ± 0.13	97.1%	4.12 ± 1.19	0.34 ± 0.11	100%
0.4	0.84 ± 1.45	0.51 ± 0.10	83.6%	2.49 ± 1.30	0.48 ± 0.08	100%
0.6	0.21 ± 0.71	0.69 ± 0.06	46.4%	0.79 ± 0.63	0.68 ± 0.08	100%
	GraphAF			**MoFlow**		
δ	**Improvement**	**Similarity**	**Success**	**Improvement**	**Similarity**	**Success**
0.0	13.13 ± 6.89	0.29 ± 0.15	100%	8.61 ± 5.44	0.30 ± 0.20	98.88%
0.2	11.90 ± 6.86	0.33 ± 0.12	100%	7.06 ± 5.04	**0.43 ± 0.20**	96.75%
0.4	8.21 ± 6.51	0.49 ± 0.09	99.88%	4.71 ± 4.55	**0.61 ± 0.18**	85.75%
0.6	4.98 ± 6.49	0.66 ± 0.05	96.88%	2.10 ± 2.86	**0.79 ± 0.14**	58.25%



EXP4: Constrained Property Optimization

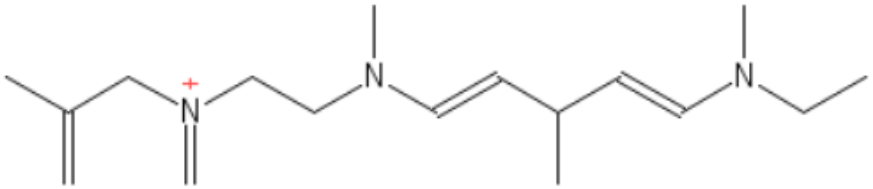
CN1CC[NH+](C)CCN(C)CC[NH+](C)CCN(C)CC[NH+](C)CC1



PlogP: -49.7182

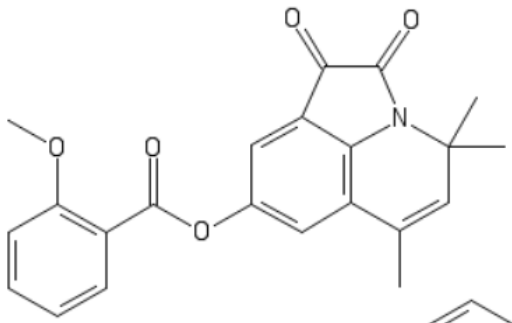
+47.87

PlogP: -1.849



C=C(C)C[N+](=C)CCN(C)C=CC(C)C=CN(C)CC

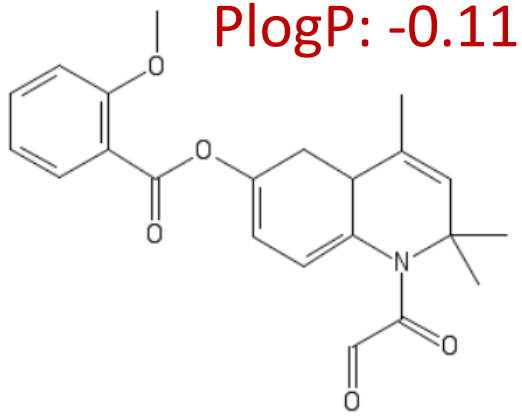
COc1ccccc1C(=O)Oc1cc2c3c(c1)C(C)=CC(C)(C)N3C(=O)C2=O



PlogP: -15.962

+15.85

PlogP: -0.11607



COc1ccccc1C(=O)OC1=CC=C2C(C1)C(C)=CC(C)(C)N2C(=O)C=O

Summary

- **Novel MoFlow model for molecular graph generation**
  - A variant of Glow for bonds
  - Novel Graph conditional flow for atoms given bonds
  - Novel validity correction
  - Invertible, fast inference and generation at one shot
- **The state-of-the-art results**
  - Best results for generation and reconstruction
    - w.r.t. novelty, uniqueness, validity, and reconstruction rate
  - Best results for QED property optimization
    - More drug-like molecules
  - Best similarity scores for constraint optimization and second best improvement scores for plogP

KDD2020

Weill Cornell Medicine

MoFlow: An Invertible Flow Model for Generating Molecular Graphs

Chengxi Zang and Fei Wang
Weill Cornell Medicine
www.calvinzang.com

Outline

- Introduction
- Network Embedding & GNNs
- Knowledge Graph Mining
- Graph Generative Models & Drug Discovery
- **Discussions**