Graph Neural Networks for Modeling Small Molecules

Thomas Kipf*, Petar Veličković, Yujia Li

Google Research Brain Team — *DeepMind* — *DeepMind*

24 March 2020

Small Organic Molecules Workshop, University of Oxford

*talk content based on work done at University of Amsterdam, prior to joining Google

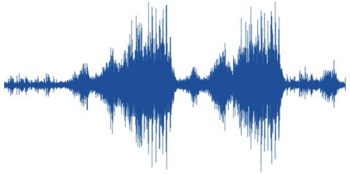
(Some) Deep Learning success stories

Vision

IMAGENET

http://www.image-net.org/

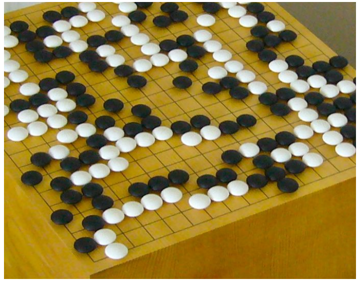
Audio / Speech

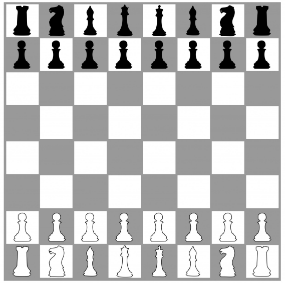


Language

The cat sat on the mat.
Die Katze saß auf der Matte.

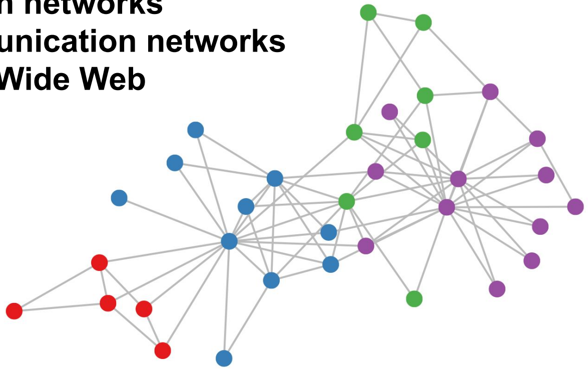
Grid-based games





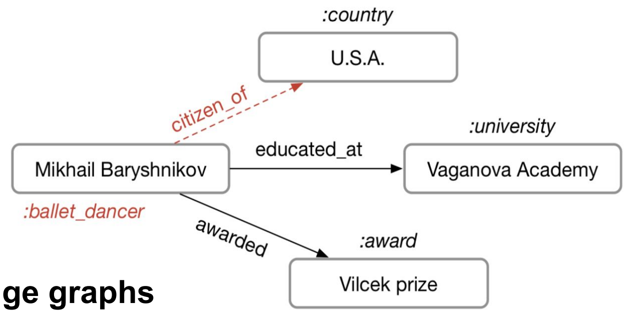
Graph-structured data

Social networks
Citation networks
Communication networks
World Wide Web



Protein interaction networks

Challenging for standard deep neural network architectures (CNNs / RNNs)



:country

U.S.A.

citizen_of

:university

Mikhail Baryshnikov

educated_at

Vaganova Academy

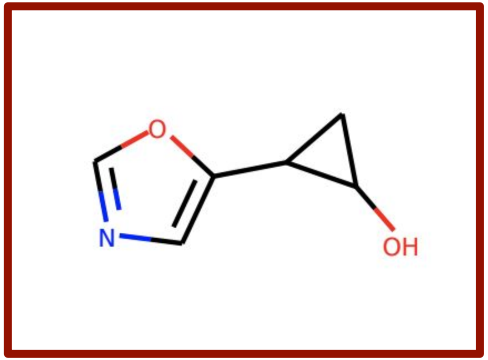
:ballet_dancer

awarded

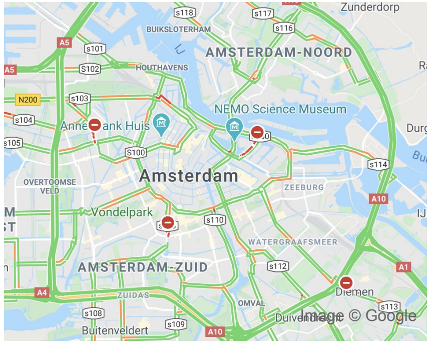
:award

Vilcek prize

Knowledge graphs



Road maps



Talk overview

1) **Introduction to Graph Neural Networks (GNNs)**
    a) **Convolution-based GNNs (“Graph Convolutional Networks”)**
    b) **GNNs with attention mechanisms**
    c) **Neural Message Passing formulation**
2) **Molecular property prediction with GNNs**
3) **Generative models for molecules**
4) **Modeling interactions between molecules**

Graph Neural Networks (GNNs)

Scarselli et al., The Graph Neural Network Model (2009)

Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$

Node features $\mathbf{h}_i$

Edge features $\mathbf{h}_{(i,j)}$

The bigger picture:

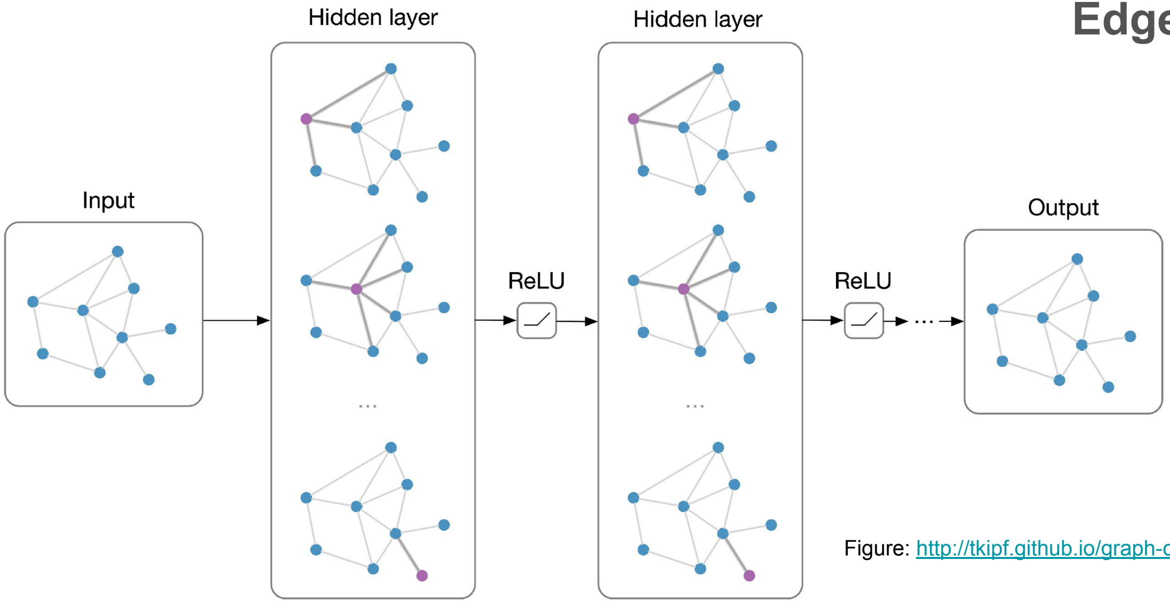
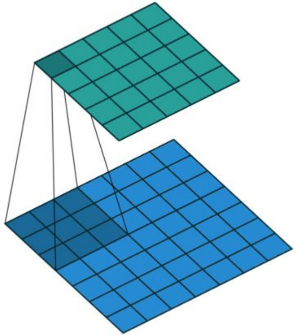


Figure: http://tkipf.github.io/graph-convolutional-networks/

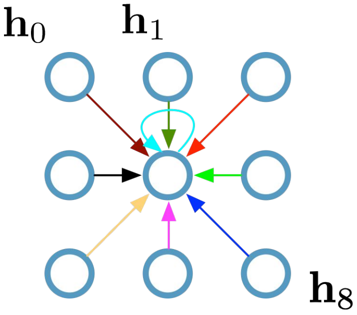
Main idea: Pass messages along edges of graph, agglomerate & transform

CNNs (on grids) as message passing

Single CNN layer with 3x3 filter:



(Animation by Vincent Dumoulin)



Update for a single pixel:

- Transform messages individually $\mathbf{W}_i \mathbf{h}_i$
- Add everything up $\sum_i \mathbf{W}_i \mathbf{h}_i$

$\mathbf{h}_i \in \mathbb{R}^F$ are (hidden layer) activations of a pixel/node

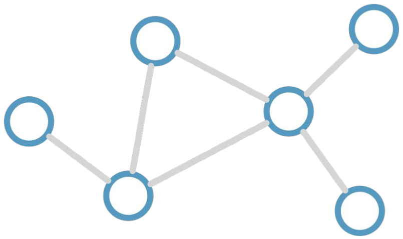
Full update:

$$\mathbf{h}'_4 = \sigma(\mathbf{W}_0 \mathbf{h}_0 + \mathbf{W}_1 \mathbf{h}_1 + \cdots + \mathbf{W}_8 \mathbf{h}_8)$$

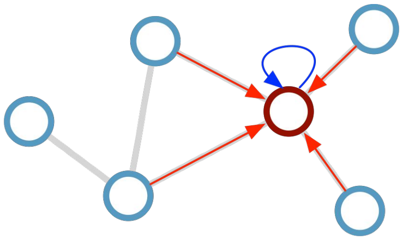
Convolution-based Graph Neural Networks

Neural FP: Duvenaud et al. (NIPS 2015), **Graph Convolutional Networks (GCN)**: Kipf & Welling (ICLR 2017)

Consider this
undirected graph:



Calculate update
for node in red:



$$\mathbf{h}_i' = \sigma\left(\mathbf{W}_0\mathbf{h}_i + \sum_{j\in\mathcal{N}_i} \alpha_{ij}\mathbf{W}_1\mathbf{h}_j\right)$$

Neighbor indices

(Optional) normalization constant

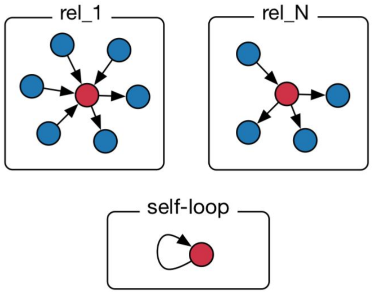
Figure from Vaswani et al. (NIPS 2017)

More expressive GNN variants

Adding support for relation types

GG-NN: Li et al. (ICLR 2016), **R-GCN**: Schlichtkrull et al. (2017)

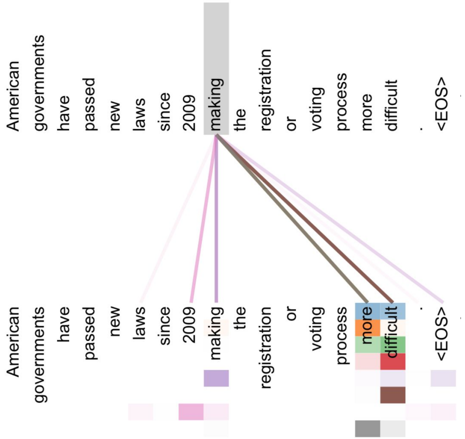
$$\mathbf{h}_i' = \sigma\left(\sum_{r=1}^{R} \sum_{j \in \mathcal{N}_i} \alpha_{ij}^{r} \mathbf{W}_r \mathbf{h}_j\right)$$



Attention-based aggregation

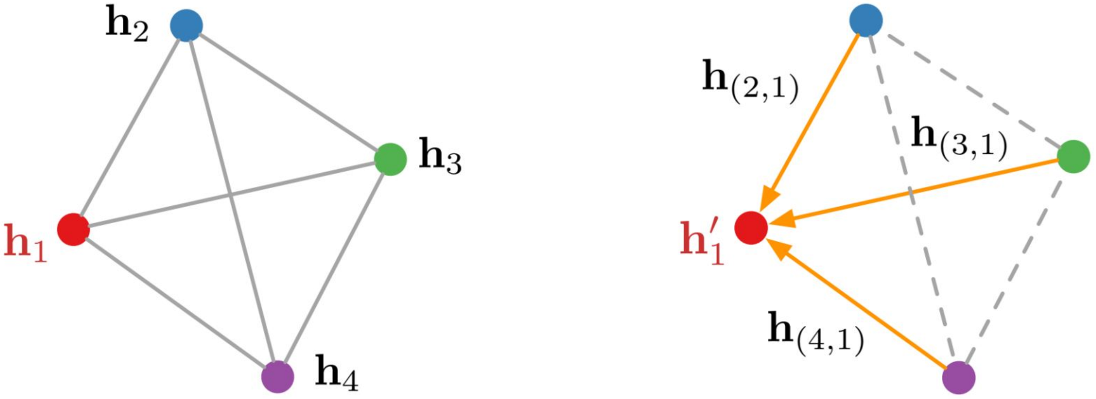
GAT: Veličković et al. (ICLR 2018), **Transformers**: Vaswani et al. (NIPS 2017)

$$\alpha_{ij}^{r} = \frac{\exp(\mathbf{h}_i^T \mathbf{W}_r' \mathbf{h}_j)}{\sum_{k \in \mathcal{N}_i} \exp(\mathbf{h}_i^T \mathbf{W}_r' \mathbf{h}_k)}$$



Neural Message Passing formulation of GNNs

Message Passing Neural Networks (MPNN): Gilmer et al. (ICML 2017), **GraphNets**: Battaglia et al. (2018)



Edge update (message) $\mathbf{h}_{(i,j)} = f_{\text{edge}}(\mathbf{h}_i, \mathbf{h}_j, \mathbf{x}_{(i,j)})$

Node features

Node update $\mathbf{h}'_i = f_{\text{node}}(\mathbf{h}_i, \sum_{j \in \mathcal{N}_i} \mathbf{h}_{(j,i)}, \mathbf{x}_i)$

Figure: T. N. Kipf, Deep Learning with Graph-Structured Representations (unpublished)

Code repositories

- Graph Convolutional Networks (Kipf and Welling, ICLR 2017)
  - https://github.com/tkipf/gcn (TensorFlow)

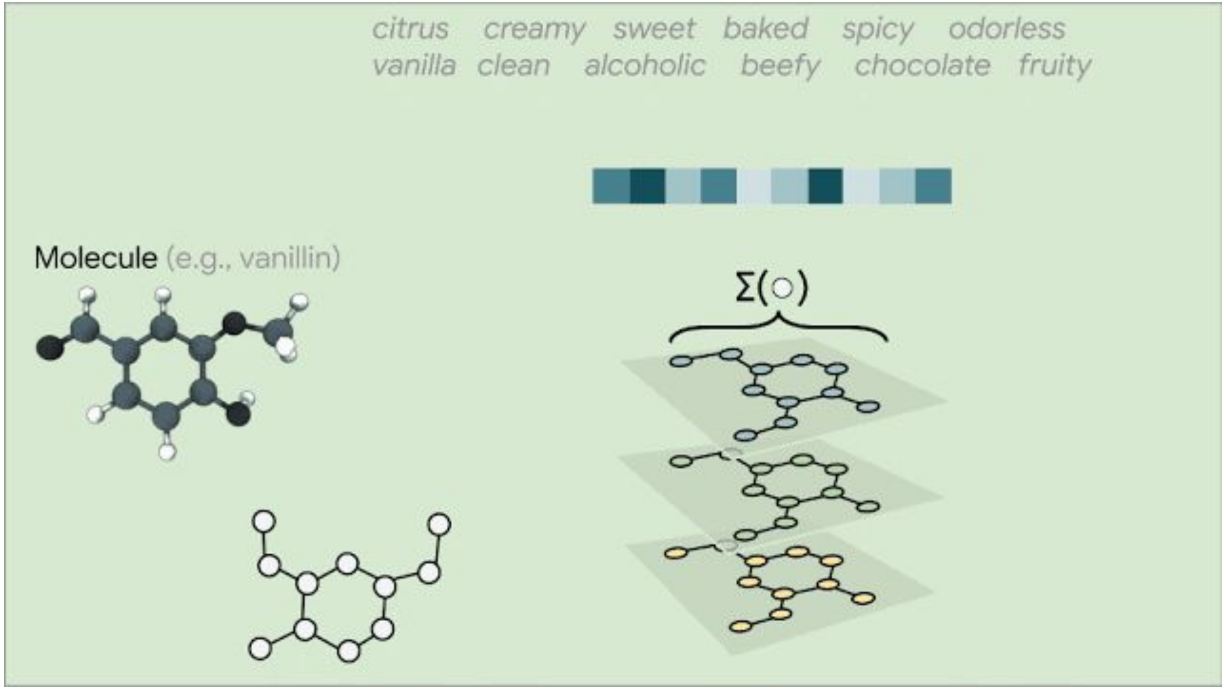
- Graph Attention Networks (Veličković *et al.*, ICLR 2018)
  - https://github.com/PetarV-/GAT (TensorFlow)

- Message Passing Neural Networks (Gilmer *et al.*, ICML 2017)
  - https://github.com/brain-research/mpnn (TensorFlow)

Molecular property prediction with GNNs

GNNs applied to molecular data

Sanchez-Lengeling et al., **Machine Learning for Scent: Learning Generalizable Perceptual Representations of Small Molecules** (2019)



Animation: https://ai.googleblog.com/2019/10/learning-to-smell-using-deep-learning.html

GNNs on molecules for discovery of antibiotics

Stokes et al., A Deep Learning Approach to Antibiotic Discovery (Cell 2020)

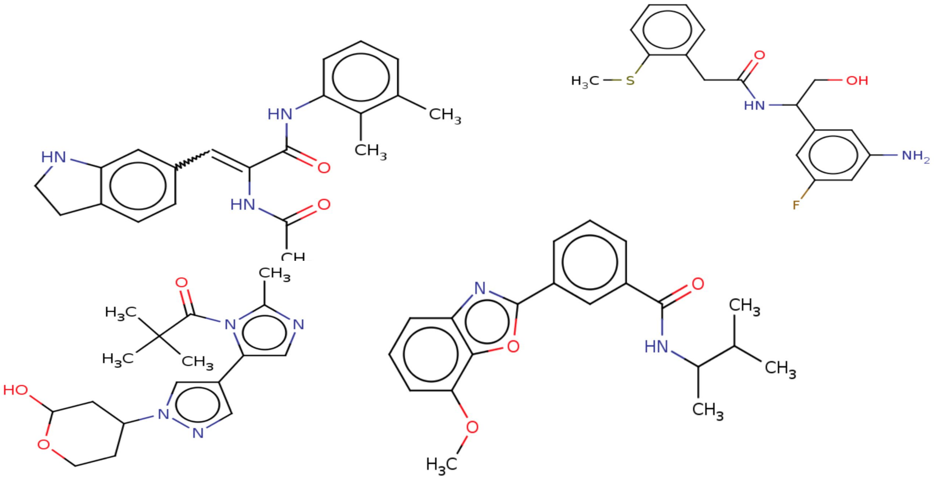
Simple recipe:

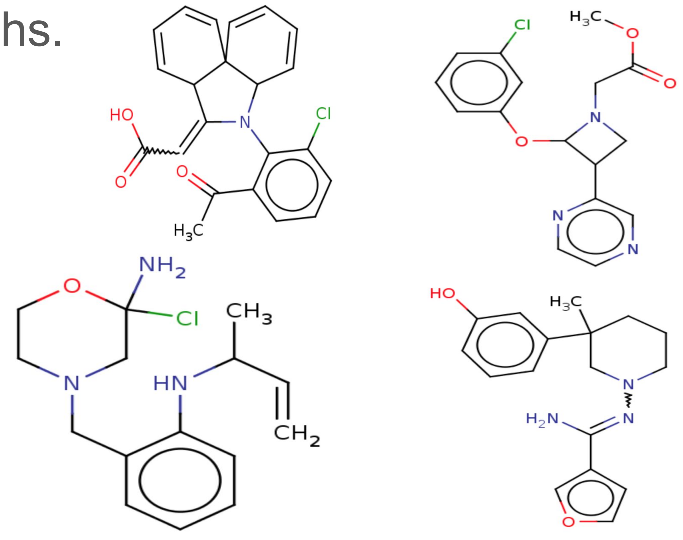
1) Train GNN (simple GCN, termed “ChemNet”) on database of molecules with known antibiotic properties
2) Evaluate GNN on a much larger database of molecules
3) Rank molecules by predicted property
4) Top-ranked molecules are good candidates for antibiotics!

Generative models of molecule graphs

Graph *generation*

- We seek a model capable of producing graphs that “capture” the empirical properties of a given **distribution** of graphs.





- **Very** challenging! (*discrete* decisions)
    - Until recently, most results in domain of *small chemicals* (< 50 nodes!)

See **“Efficient Graph Generation with Graph Recurrent Attention Networks”** (Liao *et al.*, NeurIPS’19)**:** recent large-scale results.

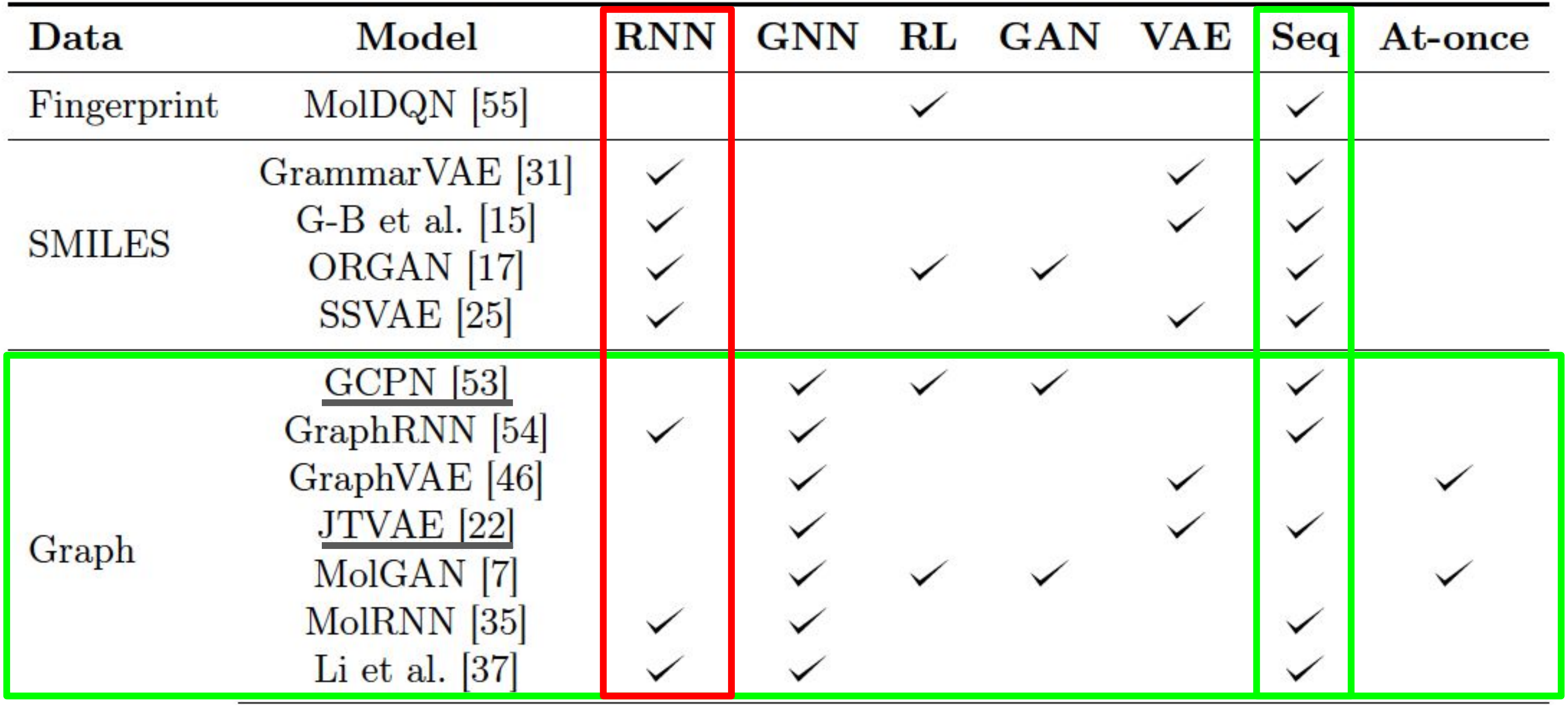
Drug discovery

Application: drug design is very *important*:

- Drug discovery is costly, time-expensive;
- Use drug databases and machine learning models to support searching the complex space of candidates (10^23 -- 10^60)

Want **targeted** molecules that optimise for specific **properties**.

Taxonomy of related work



Data	Model	RNN	GNN	RL	GAN	VAE	Seq	At-once
Fingerprint	MolDQN [55]			✓			✓	
SMILES	GrammarVAE [31]	✓				✓	✓	
	G-B et al. [15]	✓				✓	✓	
	ORGAN [17]	✓		✓	✓		✓	
	SSVAE [25]	✓				✓	✓	
Graph	GCPN [53]		✓	✓	✓		✓	
	GraphRNN [54]	✓	✓				✓	
	GraphVAE [46]		✓			✓		✓
	JTVAE [22]		✓			✓	✓	
	MolGAN [7]		✓	✓	✓			✓
	MolRNN [35]	✓	✓				✓	
	Li et al. [37]	✓	✓				✓	

JTVAE

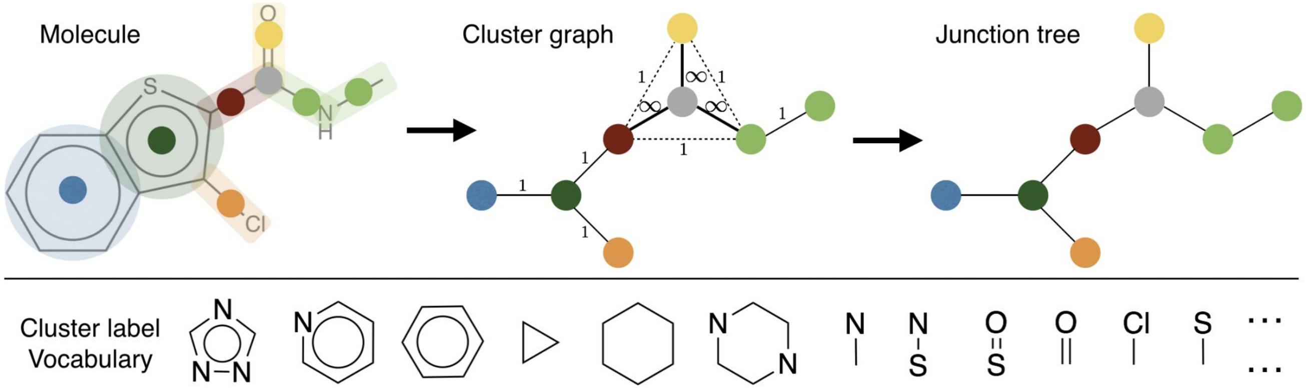
- Junction Tree Variational Autoencoder (Jin *et al.*, ICML 2018)

- Generate molecular graphs **iteratively**, guided by the molecule’s **substructures**, from a **variational representation**.

- Perform targeted optimisation within the *representation space*.

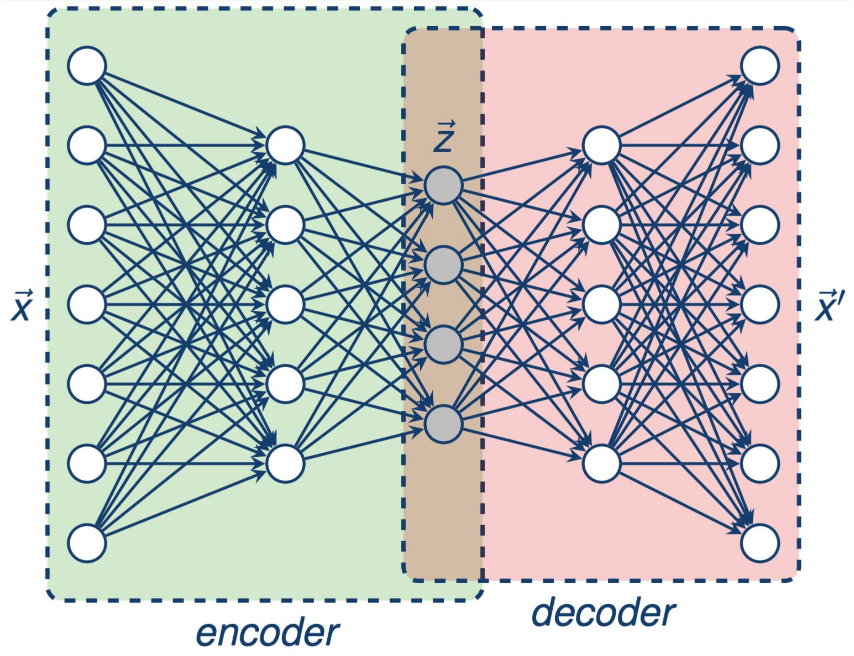
JTVAE components

- **Junction Tree** Variational Autoencoder (Jin *et al.*, ICML 2018)



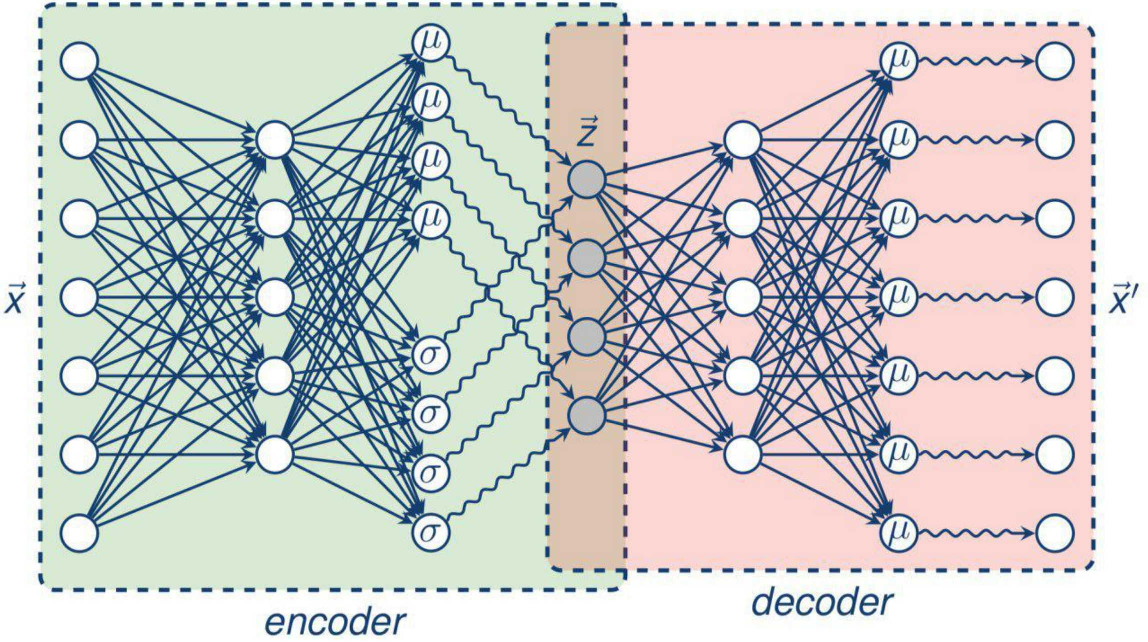
JTVAE components

- Junction Tree Variational **Autoencoder** (Jin *et al.*, ICML 2018)

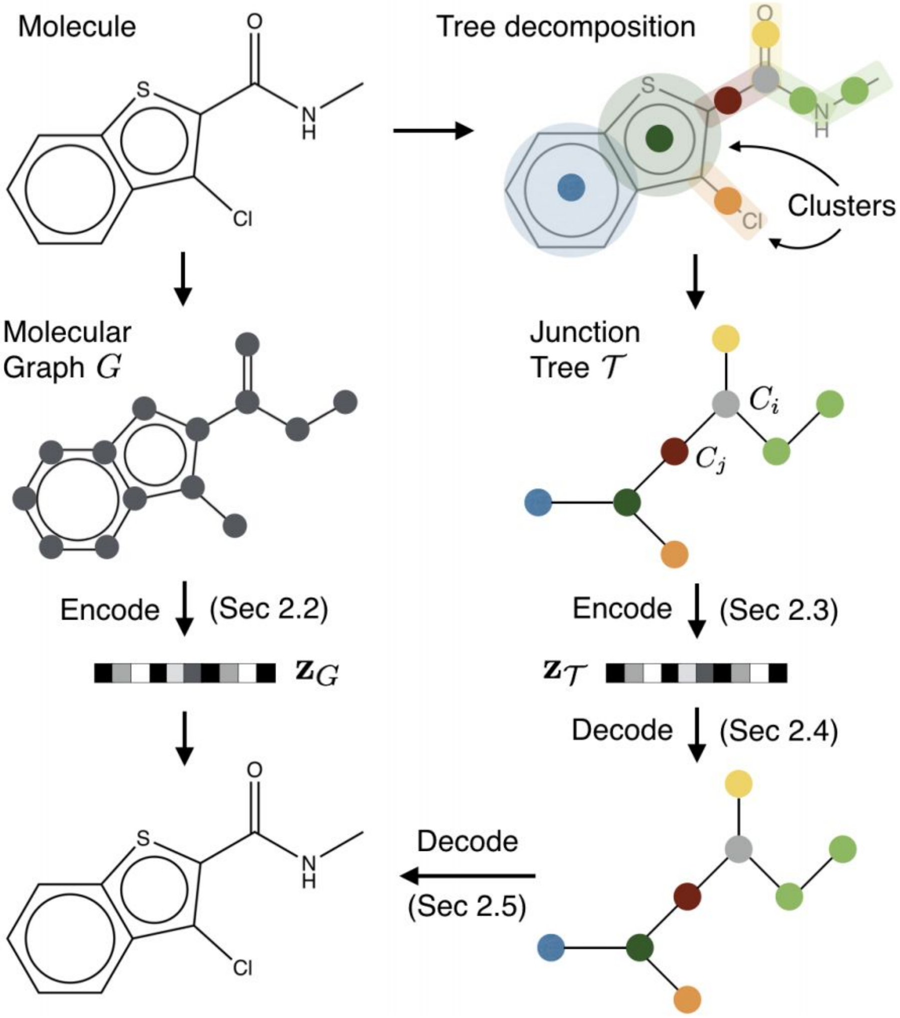


JTVAE components

- Junction Tree **Variational Autoencoder** (Jin *et al.*, ICML 2018)

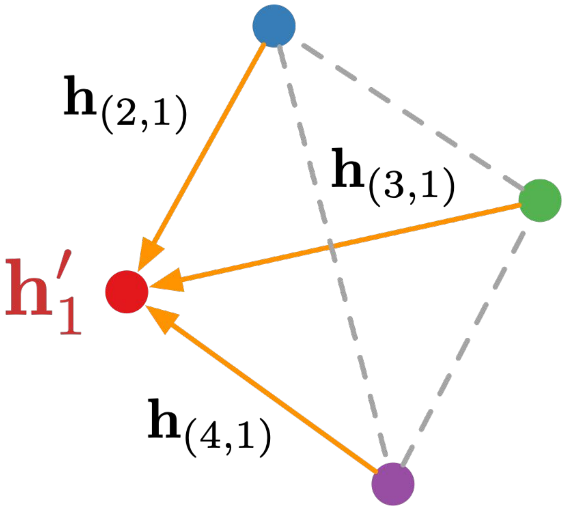


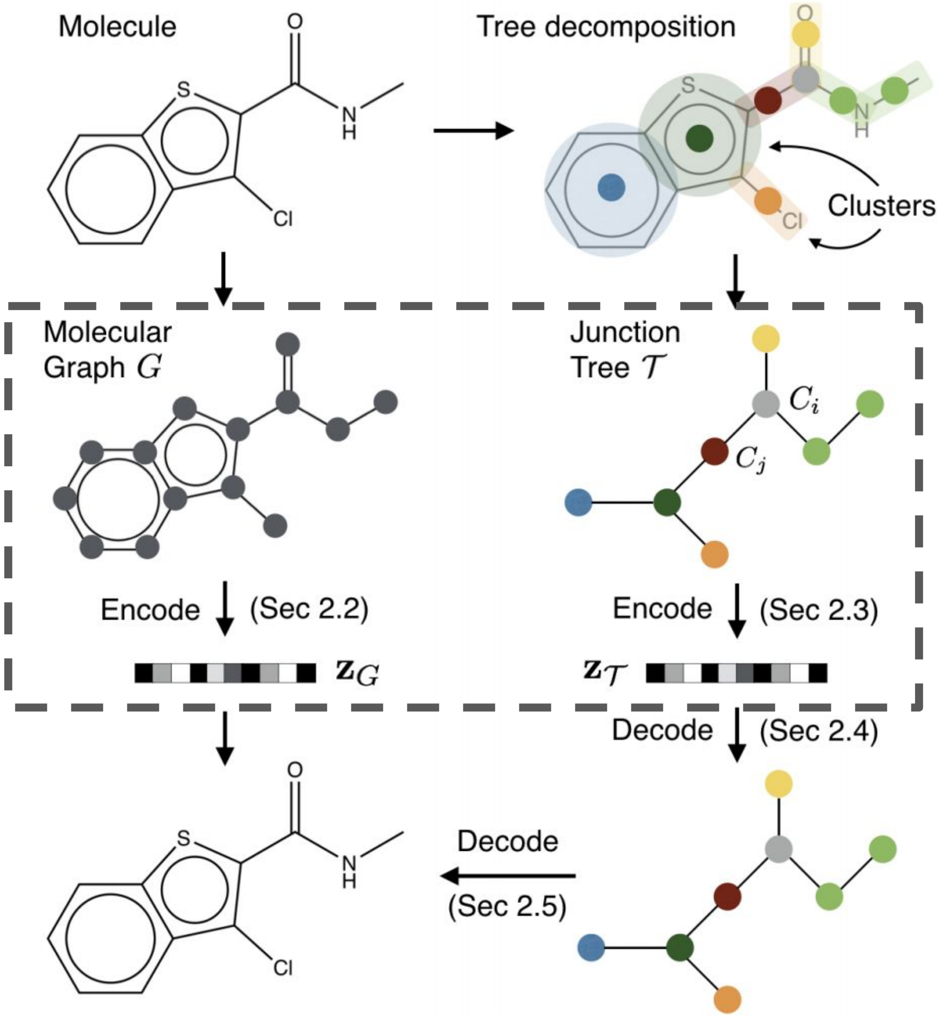
Putting it all together



Putting it all together

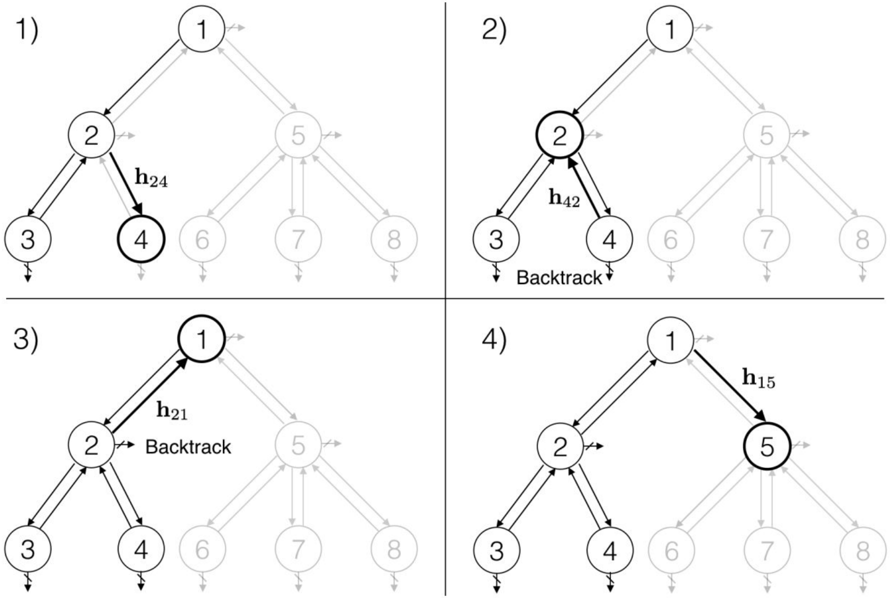
MPNN for encoding graphs and trees

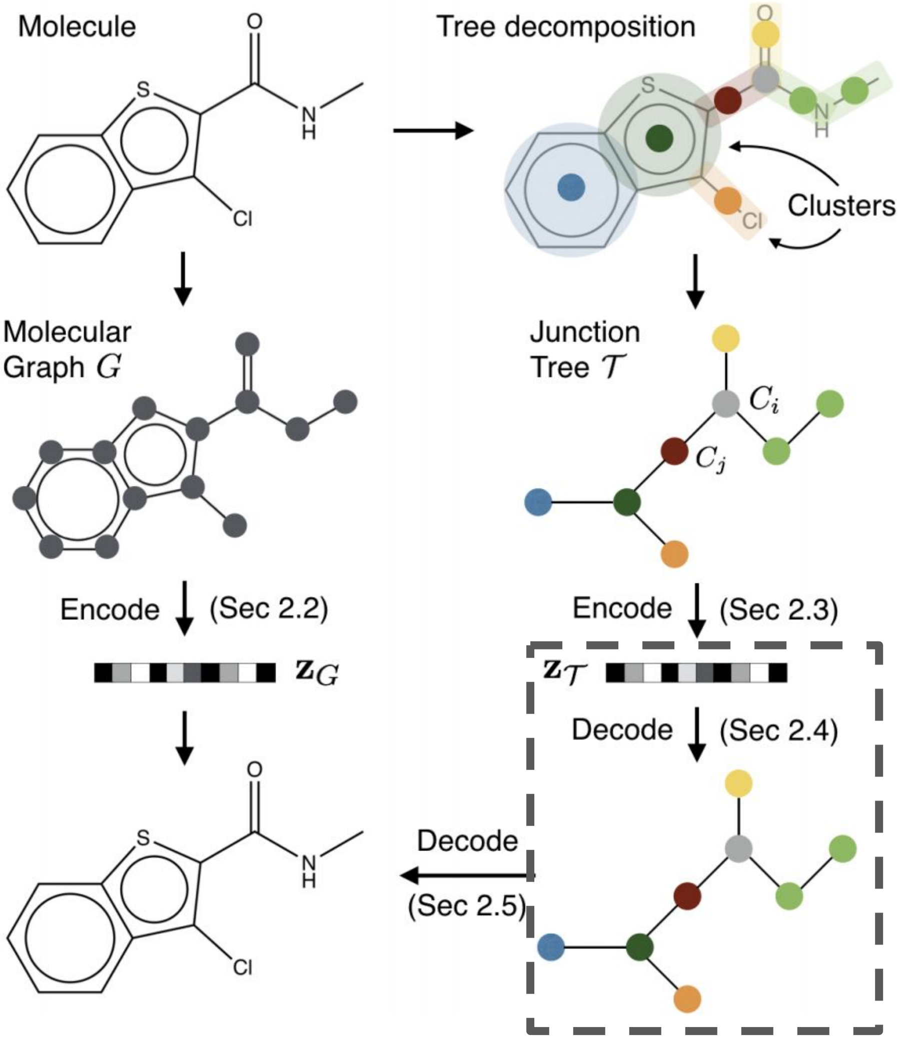




Putting it all together

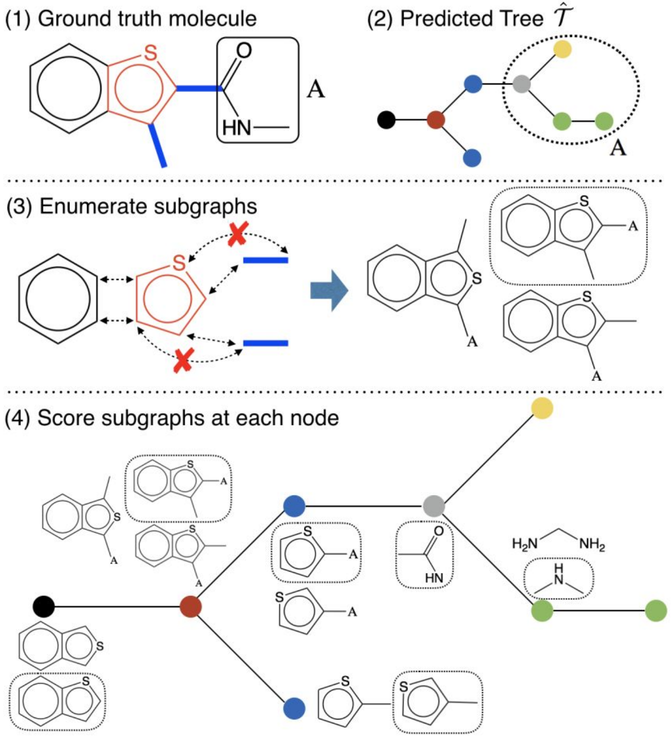
Decode tree in *depth-first* fashion

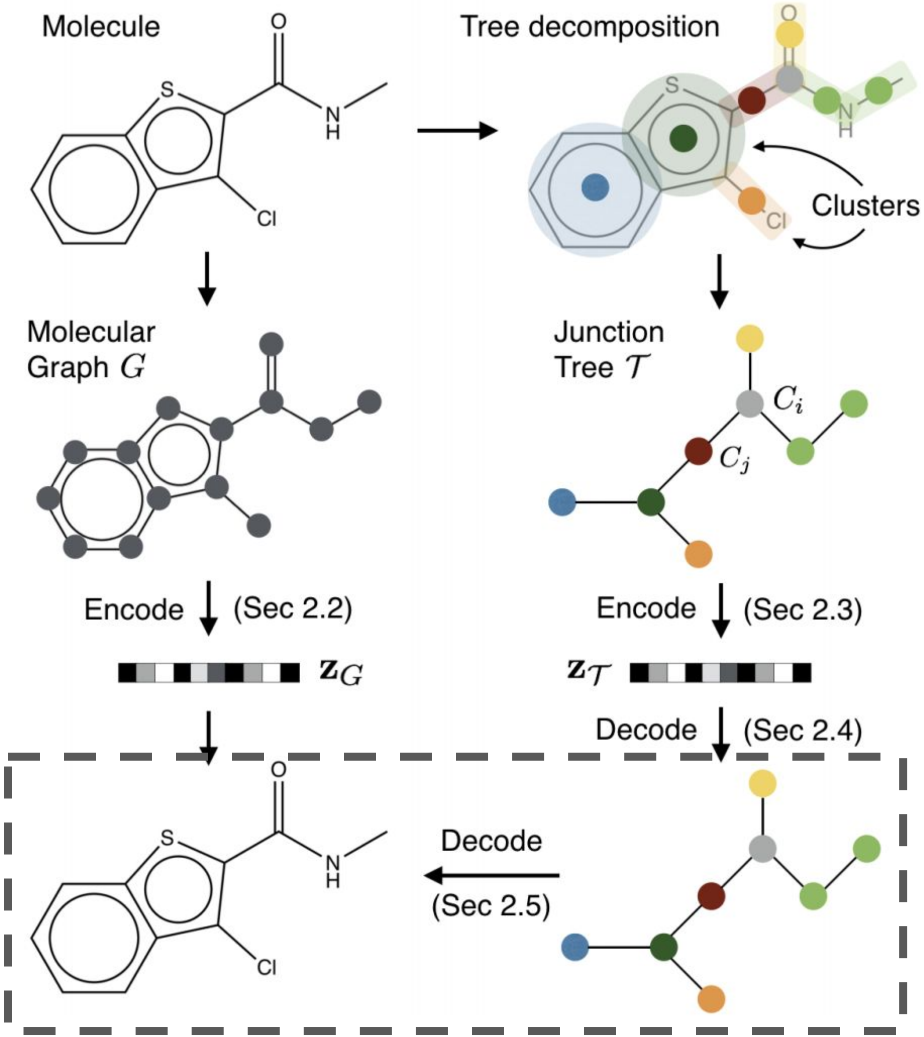




Putting it all together

Decode graph by reassembling from tree


(1) Ground truth molecule
S
O
HN
A
(2) Predicted Tree $\hat{\mathcal{T}}$
A
(3) Enumerate subgraphs
S
A
(4) Score subgraphs at each node
S
A
O
HN
H_2N
NH_2
H
N



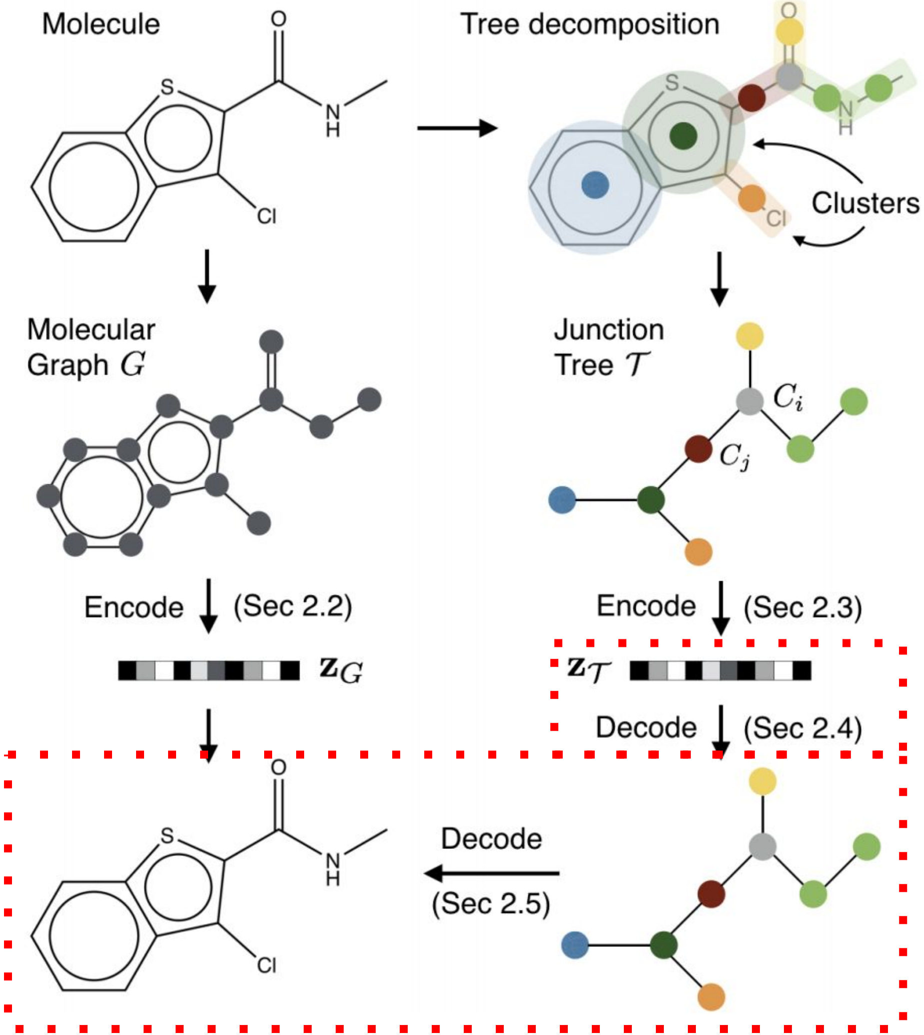
Once trained...

1. Sample tree codes z_T

2. Decode junction tree

3. Decode graph from junction tree

4. Repeat to keep generating :)



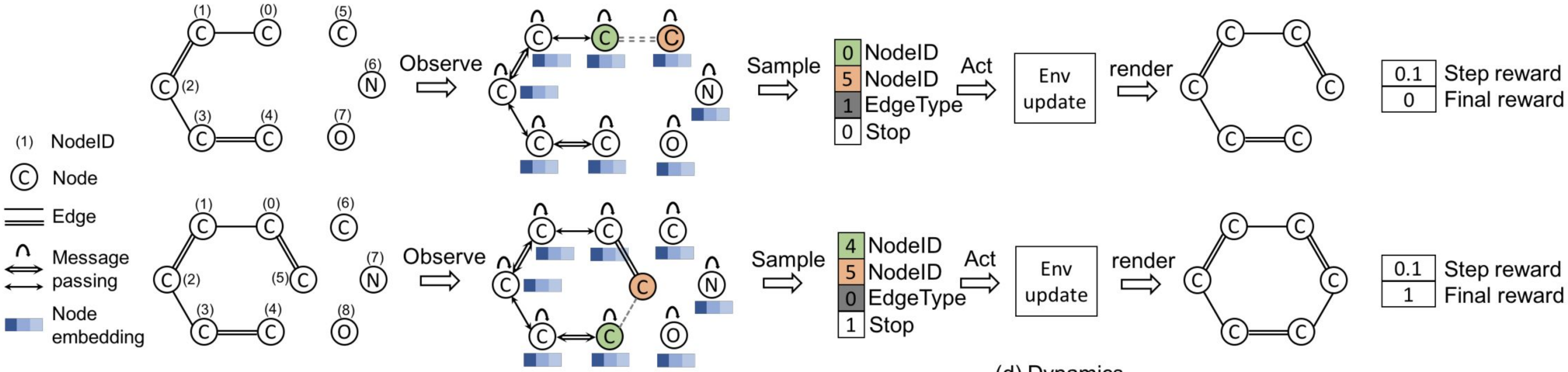
GCPN

- Graph Convolutional Policy Network (You *et al.*, NeurIPS 2018)

- At each step, take a *partially constructed molecule*, and a set of *motifs*, and discretely decide on:
  - **Two atoms** to connect (one must be in molecule);
  - **Bond type** between them;
  - Whether to **stop**.

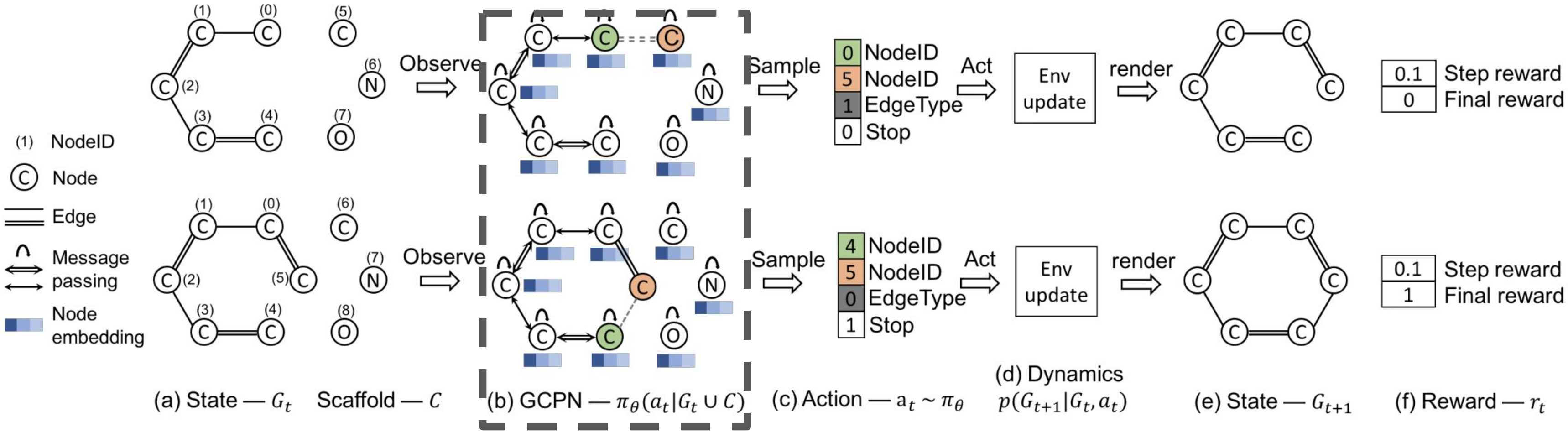
- Evaluate decisions via ***reinforcement learning*** framework.

GCPN Pipeline



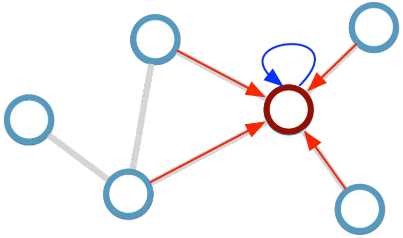
(a) State — G_t Scaffold — C (b) GCPN — $\pi_\theta(a_t|G_t \cup C)$ (c) Action — $a_t \sim \pi_\theta$ (d) Dynamics $p(G_{t+1}|G_t, a_t)$ (e) State — G_{t+1} (f) Reward — r_t

GCPN Pipeline



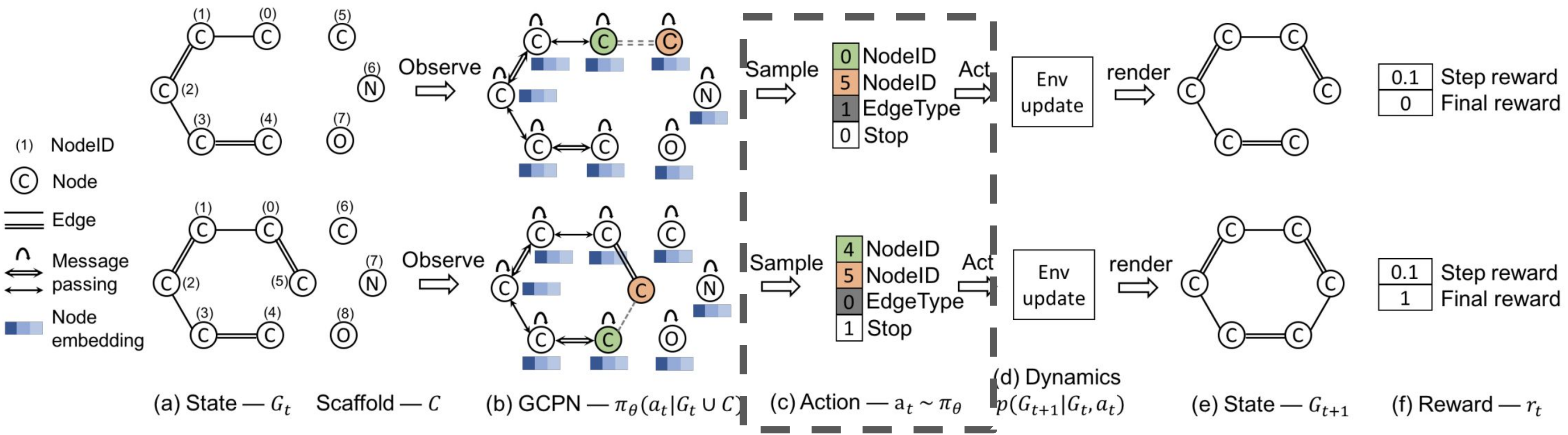
(a) State — G_t Scaffold — C (b) GCPN — $\pi_\theta(a_t|G_t \cup C)$ (c) Action — $a_t \sim \pi_\theta$ (d) Dynamics $p(G_{t+1}|G_t, a_t)$ (e) State — G_{t+1} (f) Reward — r_t

GCN (parametrised by bond type) for encoding graphs



$$\mathbf{h}'_i = \sigma\left(\sum_{r=1}^{R}\sum_{j\in\mathcal{N}_i}\alpha_{ij}^{r}\mathbf{W}_r\mathbf{h}_j\right)$$

GCPN Pipeline



Sample actions by **scoring** appropriate tuples of inputs

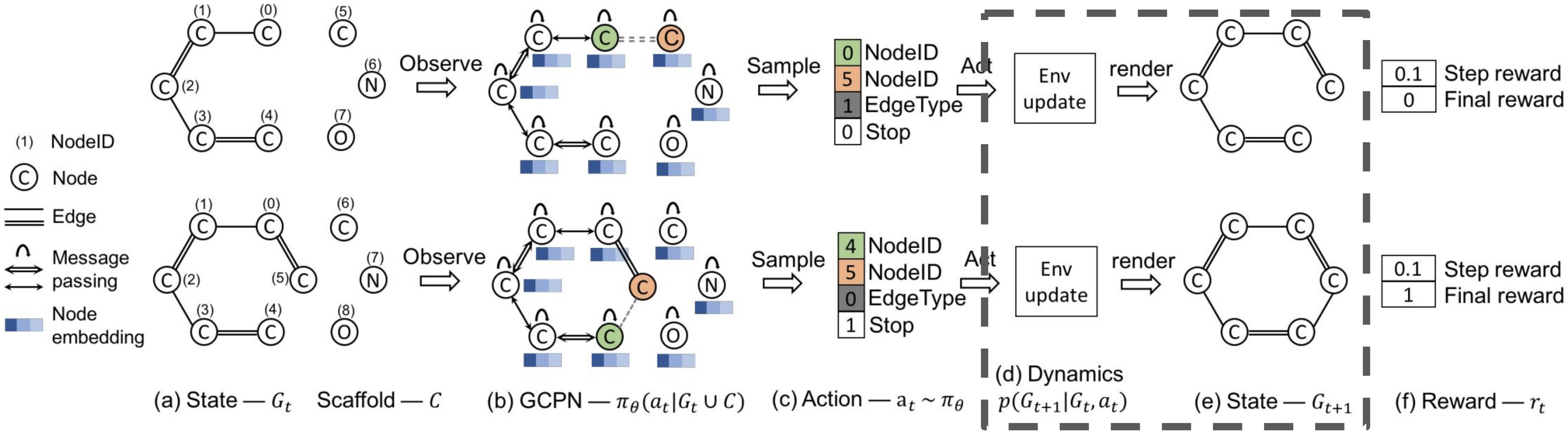
$$f_{\text{first}}(s_t) = \text{SOFTMAX}(m_f(X)), \qquad a_{\text{first}} \sim f_{\text{first}}(s_t) \in \{0,1\}^n$$

$$f_{\text{second}}(s_t) = \text{SOFTMAX}(m_s(X_{a_{\text{first}}}, X)), \qquad a_{\text{second}} \sim f_{\text{second}}(s_t) \in \{0,1\}^{n+c}$$

$$f_{\text{edge}}(s_t) = \text{SOFTMAX}(m_e(X_{a_{\text{first}}}, X_{a_{\text{second}}})), \qquad a_{\text{edge}} \sim f_{\text{edge}}(s_t) \in \{0,1\}^b$$

$$f_{\text{stop}}(s_t) = \text{SOFTMAX}(m_t(\text{AGG}(X))), \qquad a_{\text{stop}} \sim f_{\text{stop}}(s_t) \in \{0,1\}$$

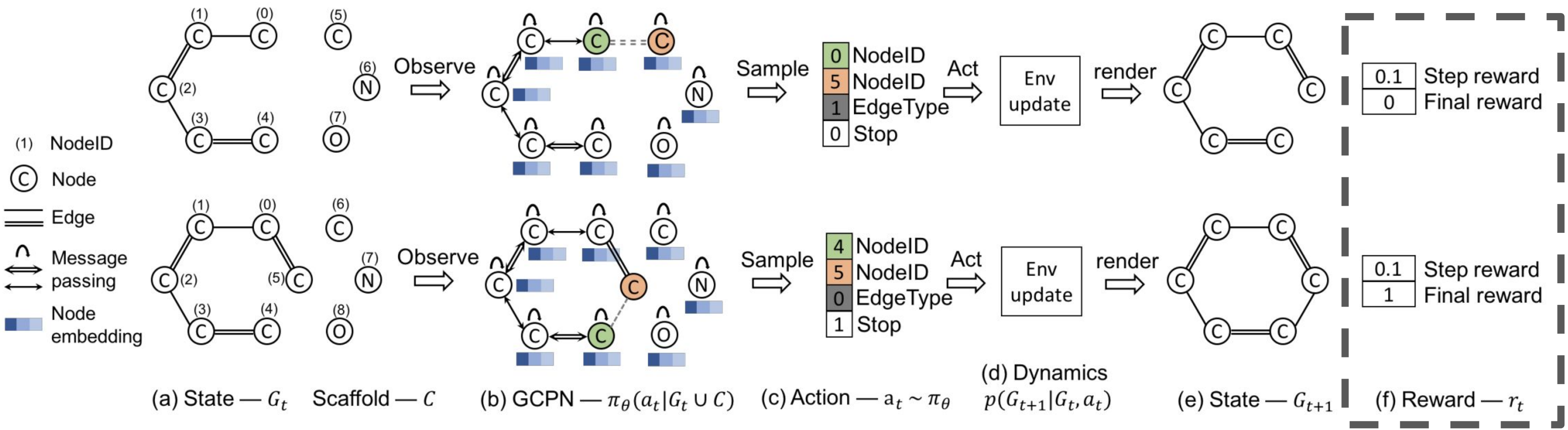
GCPN Pipeline



Check whether new molecule breaks **valency constraints**

If so---**restart** and give reward of -1.

GCPN Pipeline



Evaluate **reward** using *property scores* and *GAN score*

$$V(G_\theta, D_\phi) = \mathbb{E}_{\vec{x}\sim p_{data}}[\log D_\phi(\vec{x})] + \mathbb{E}_{\vec{x}\sim G_\theta}[\log(1 - D_\phi(\vec{x}))]$$

Optimise rewards using **reinforcement learning** (e.g. PPO).

Code repositories

- Junction Tree Variational Autoencoders (Jin *et al.*, ICML 2018)
  - https://github.com/wengong-jin/icml18-jtnn (PyTorch)

- Graph Convolutional Policy Network (You *et al.*, NeurIPS 2019)
  - https://github.com/bowenliu16/rl_graph_generation (TensorFlow)

GNN models of molecular interactions

“Drug-Drug Adverse Effect Prediction with Graph Co-Attention”
Deac *et al.*, ICML WCB 2019

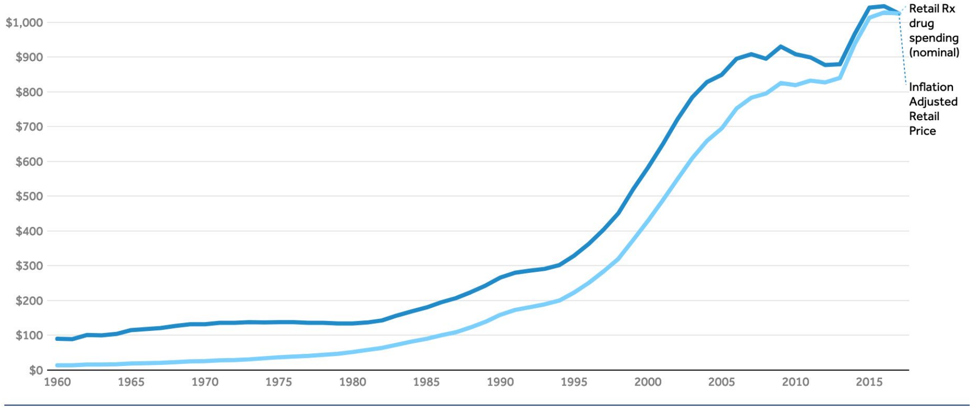
“Modeling polypharmacy side effects with graph convolutional networks”
Žitnik *et al.*, Bioinformatics

Also see:
“Graph Matching Networks for Learning the Similarity of Graph Structured Objects”
Li *et al.*, ICML 2019

Drug use is increasing

	2000	2011
Prescription Drug Use	51%	59%
>5 drugs	8.2%	15%

Nominal and inflation-adjusted per capita spending on retail prescription drugs, 1960-2017



Polypharmacy

Polypharmacy is the concurrent use of multiple medications by a patient.

It is necessary for chronic, complex or multiple conditions and most of the increase in cost comes from treating these.

“Hulk & Iron Man” analogy: drugs correspond to ‘heroes’, but putting them together can **destroy** the surrounding city!

Adverse side-effects

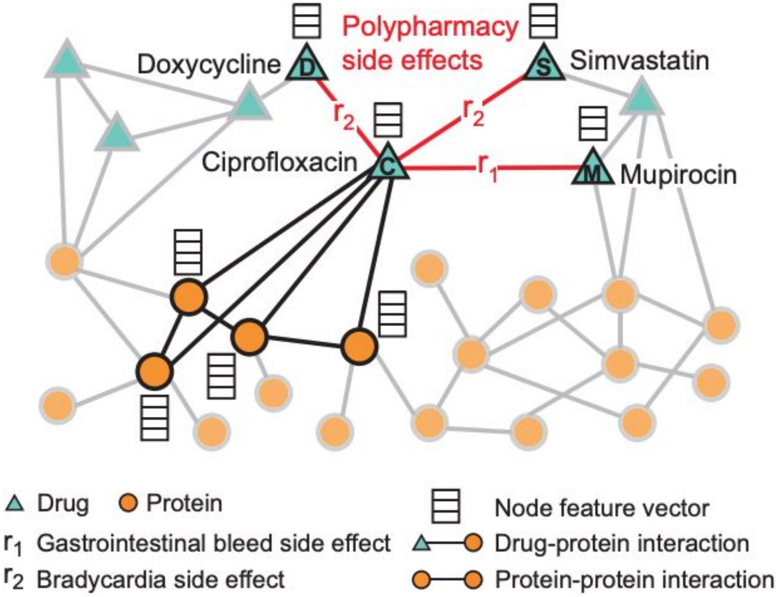
Side effects affecting 15% of the population, treatment costs exceeding $177 billion/year

Some found in Phase IV of clinical trials

But plenty are undiscovered when the drugs are put on the market

Related work

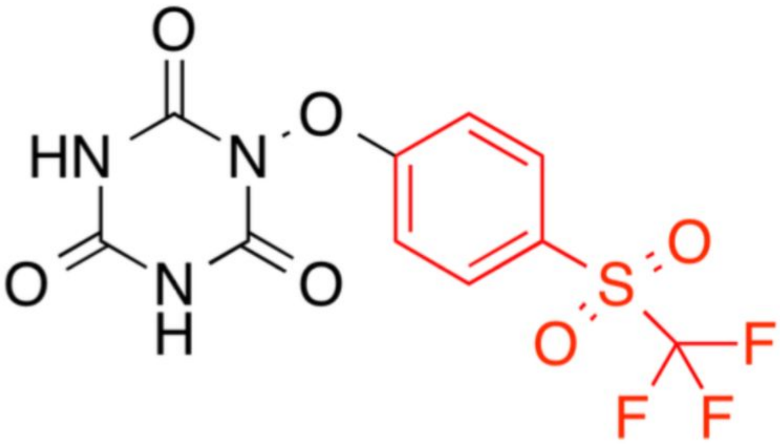
- Most models predict if a side-effect exists or not (using drug-drug similarity: chemical substructures, individual drug side effects, interaction profile fingerprints)
- Others model the interactions between pairs of drugs, pairs of proteins and drug-protein pairs to predict “missing” links between them.


Polypharmacy side effects
Doxycycline
D
Simvastatin
S
r_2
r_2
Ciprofloxacin
C
r_1
Mupirocin
M
△ Drug
● Protein
r_1 Gastrointestinal bleed side effect
r_2 Bradycardia side effect
Node feature vector
Drug-protein interaction
Protein-protein interaction

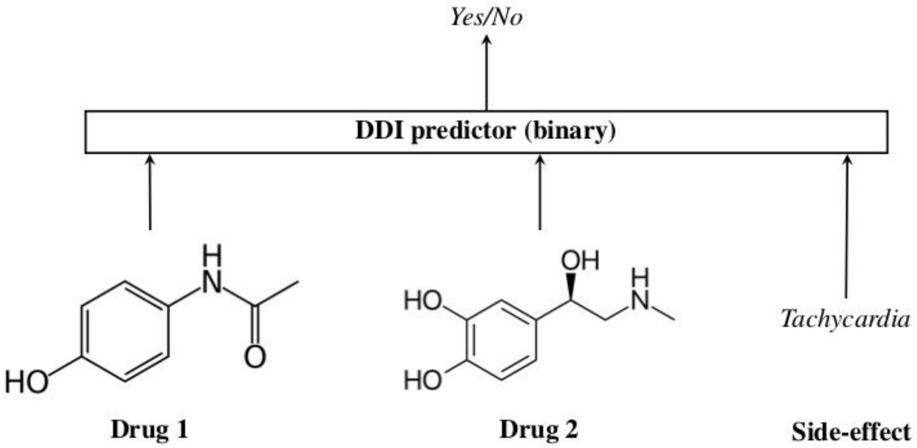
* Modeling polypharmacy side effects with graph convolutional networks, Žitnik et al, 2018

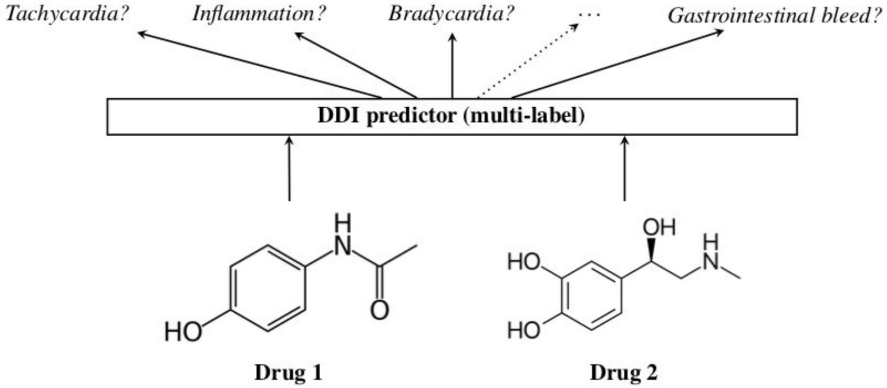
Molecules as graphs

Represent the input drugs as graphs consisting of *atoms* $a_i^{(d_x)}$ as nodes and bonds between these atoms $\left(a_i^{(d_x)}, a_j^{(d_x)}\right)$ as edges.

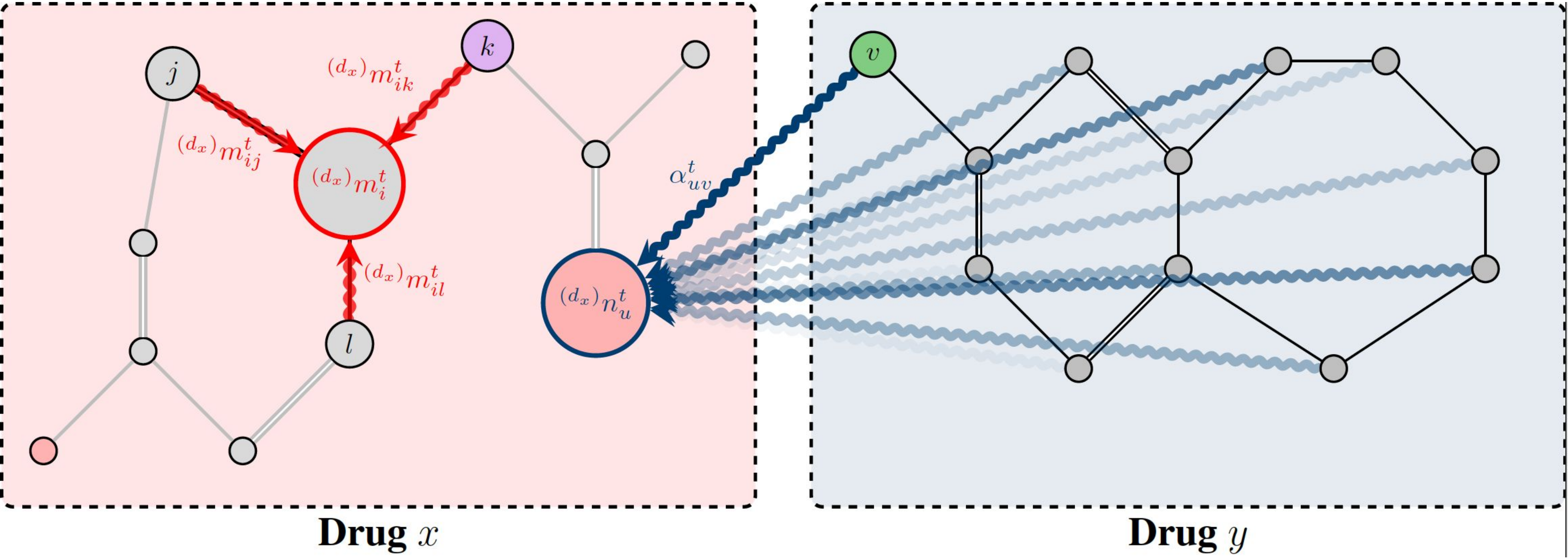


DDI - Tasks


Yes/No
DDI predictor (binary)
HO
H
N
O
OH
HO
HO
H
N
Tachycardia
Drug 1
Drug 2
Side-effect


Tachycardia?
Inflammation?
Bradycardia?
· · ·
Gastrointestinal bleed?
DDI predictor (multi-label)
HO
H
N
O
OH
HO
HO
H
N
Drug 1
Drug 2

Graph co-attention

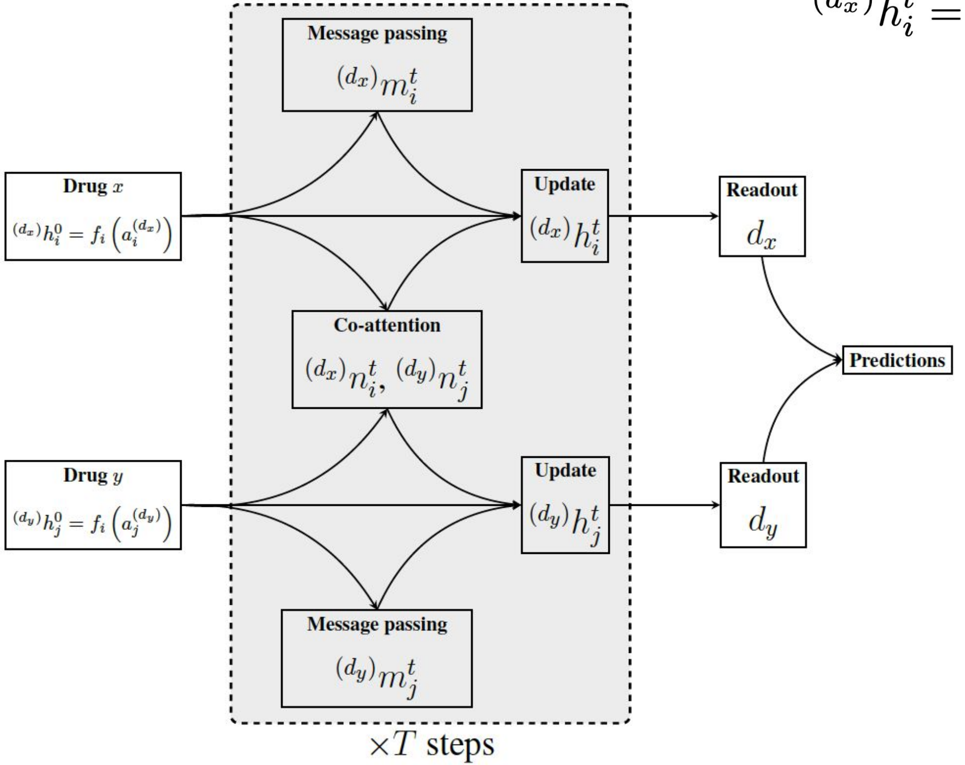


$$^{(d_x)}m^t_{ij} = f^t_e\left(e^{(d_x)}_{ij}\right) \odot f^t_v\left(^{(d_x)}h^{t-1}_j\right)$$

$$^{(d_x)}n^t_i = f^t_o\left(\Big\|_{k=1}^{K} \sum_{\forall j \in d_y} {}^{(k)}\alpha^t_{ij} \cdot {}^{(k)}\mathbf{W}^t_v \, {}^{(d_y)}h^{t-1}_j\right)$$

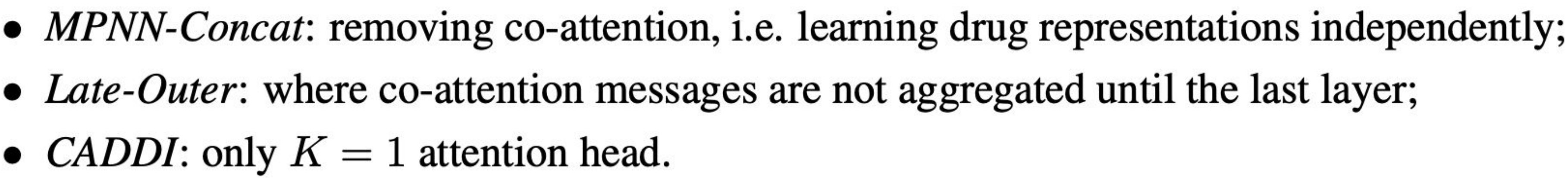
The architecture

$$^{(d_x)}h_i^t = \mathrm{LayerNorm}\left({}^{(d_x)}h_i^{t-1} + {}^{(d_x)}m_i^t + {}^{(d_x)}n_i^t \right)$$



$$d_x = \sum_{\forall j \in d_x} f_r\left({}^{(d_x)}h_j^T \right)$$

Variants considered



- *MPNN-Concat*: removing co-attention, i.e. learning drug representations independently;
- *Late-Outer*: where co-attention messages are not aggregated until the last layer;
- *CADDI*: only $K = 1$ attention head.

Quantitative results

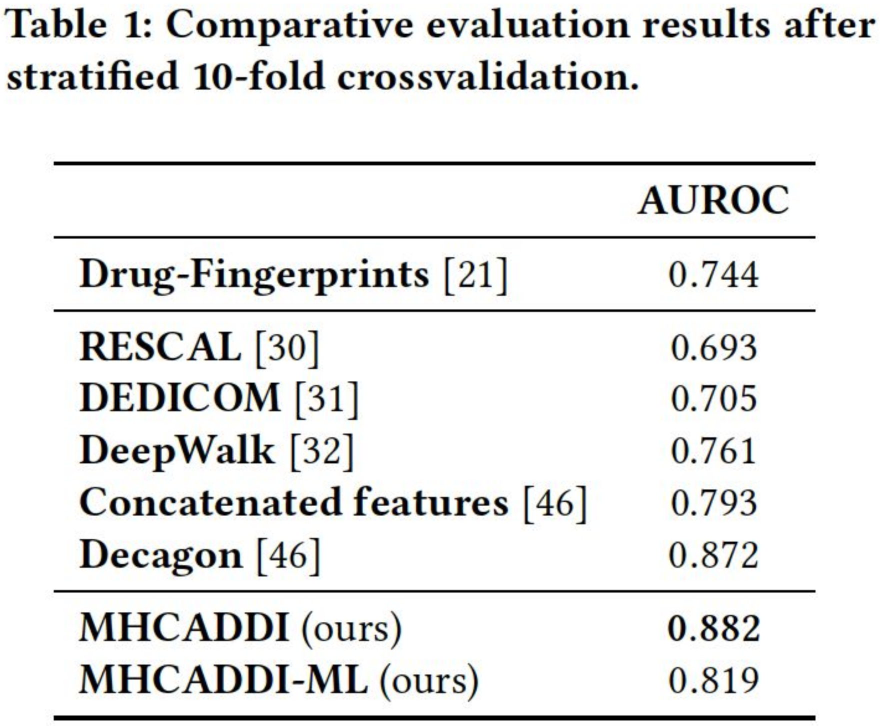


Table 1: Comparative evaluation results after stratified 10-fold crossvalidation.

	AUROC
Drug-Fingerprints [21]	0.744
RESCAL [30]	0.693
DEDICOM [31]	0.705
DeepWalk [32]	0.761
Concatenated features [46]	0.793
Decagon [46]	0.872
MHCADDI (ours)	**0.882**
MHCADDI-ML (ours)	0.819

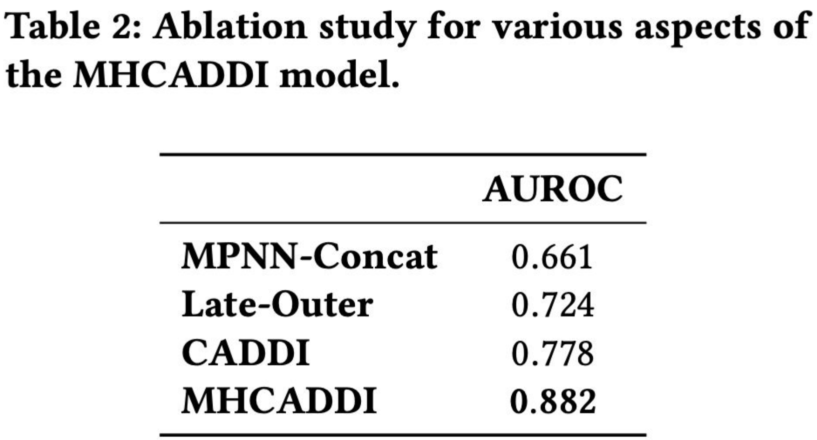


Table 2: Ablation study for various aspects of the MHCADDI model.

	AUROC
MPNN-Concat	0.661
Late-Outer	0.724
CADDI	0.778
MHCADDI	**0.882**

Code repositories

- Multi-head co-attentive drug-drug interactions (MHCADDI) (Deac *et al.*, ICML WCB 2019):
    - https://github.com/andreeadeac22/graph_coattention (PyTorch)