Research meeting - January 19, 2022

Graph Representation Learning & Deep Generative Models On Graphs

Truong Son Hy *

*Department of Computer Science
The University of Chicago

Ryerson Physical Laboratory

Content

1. Graph representation learning
   - Message passing neural networks
   - Permutation equivariance
   - Covariant compositional networks
2. Deep generative models on graphs
   - Variational Autoencoder (VAE)
   - Equivariant graph/molecule generation
   - Multiresolution graph VAE

Graph neural networks

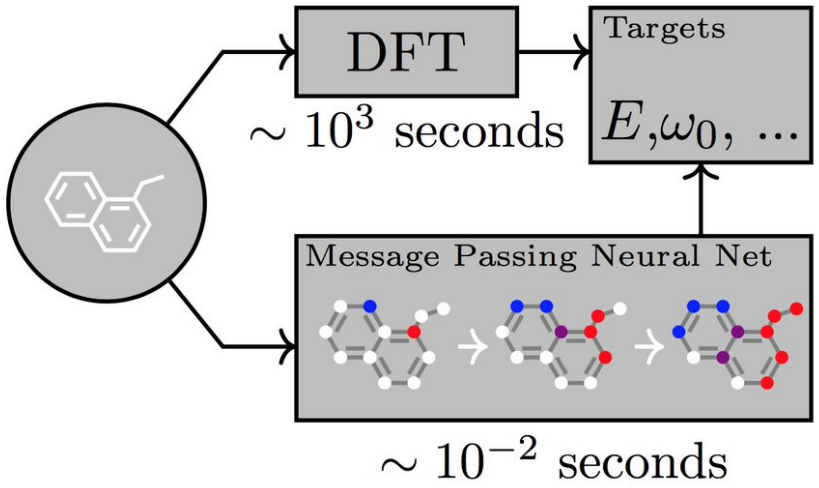
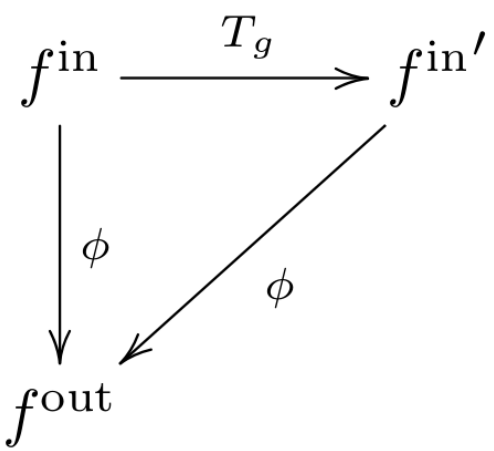


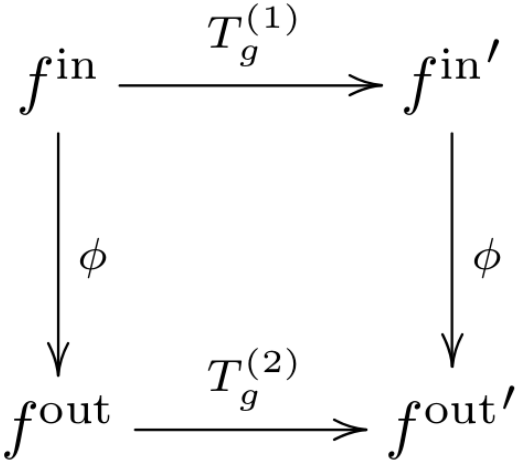
Figure 1. A Message Passing Neural Network predicts quantum properties of an organic molecule by modeling a computationally expensive DFT calculation.

Gilmer et al., *Neural Message Passing for Quantum Chemistry*, ICML 2017

Invariance vs. Equivariance

T_g is an action of a group G on the space of inputs and outputs. In case of graphs, G is the symmetry group $\mathbb{S}_n$.





Invariance: $\phi(T_g(f)) = \phi(f)$

Equivariance: $\phi(T_g^{(1)}(f)) = T_g^{(2)}(\phi(f))$

Message Passing Neural Networks and its limitation (1)

To preserve the **permutation invariance**, the aggregation function of MPNNs basically sums the messages from each node's neighborhood. The algorithm is simply expressed in matrix form as:

$$F^t = \sigma(AF^{t-1}W^t)$$

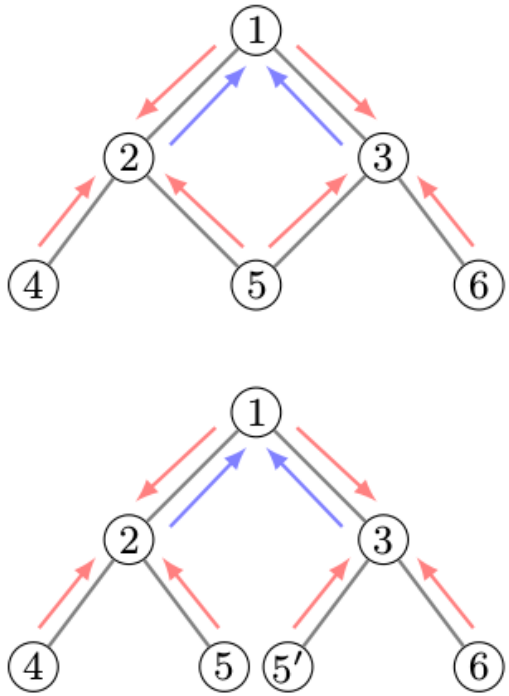
where:

- $A \in \{0,1\}^{n \times n}$ is the adjacency matrix.
  (or graph Laplacian $I_n - D^{-1/2}AD^{-1/2}$)
- $F^t \in \mathbb{R}^{n \times d}$ is the node feature matrix.
- $W^t \in \mathbb{R}^{d \times d'}$ is the weight (channels mixing) matrix, that is learnable.
- σ is the nonlinearity.

Message Passing Neural Networks and its limitation (2)

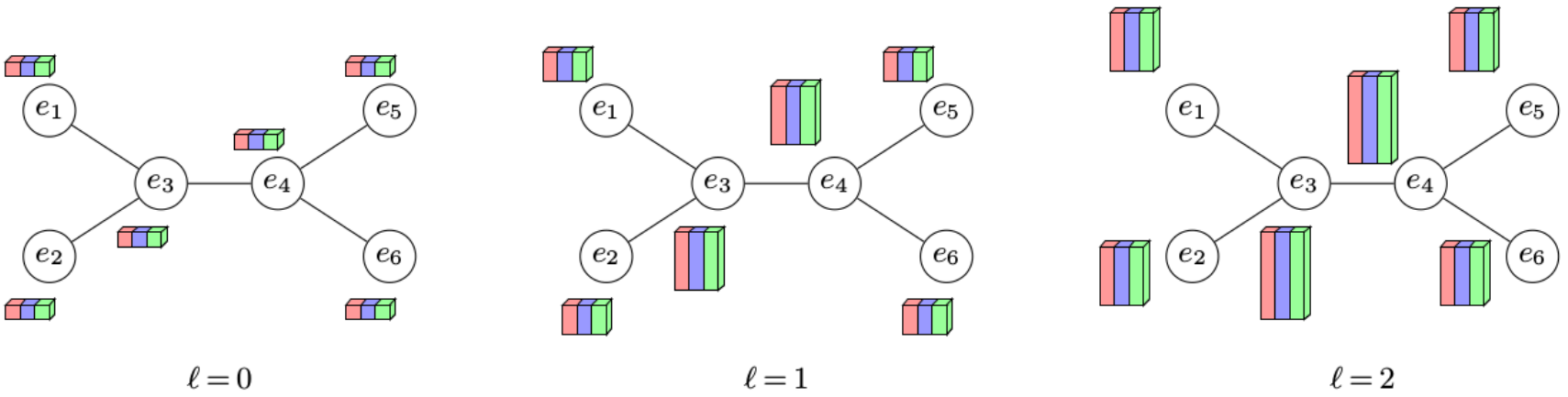
The summing operator **limits** the representative power of MPNNs such that each node loses their identity after being aggregated. For example:

These two graphs are **not** isomorphic, but after a single round of message passing (red arrows), the messages at vertices 2 and 3 will be identical in both graphs. In the second round, vertex 1 will get the same messages in both graphs (blue arrows), and will have **no** way to distinguish whether 5 and 5' are the same vertex or not.



Covariant Compositional Networks (1)

We propose a new general architecture called **Covariant Compositional Networks** (CCNs) in which the messages are represented by higher order tensors and transform covariantly/equivariantly according to a specific representation of the symmetry group of its receptive field.

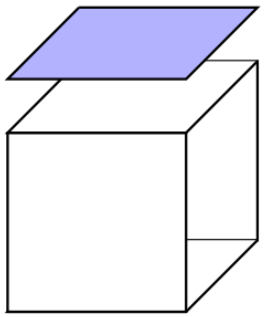


Feature tensors in a **first order** CCN for ethylene (C_2H_4) assuming three channels (red, green, blue).

Covariant Compositional Networks (2)

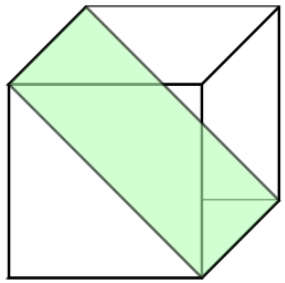
Permutation covariant operators:

1. Projections



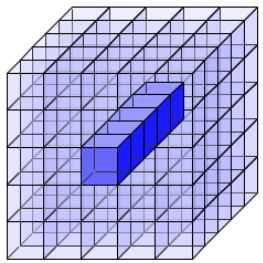
$$C_{i,j} = \sum_a A_{a,i,j}$$

2. Diagonals



$$C_{i,j} = \sum_i A_{i,i,j}$$

3. Contractions

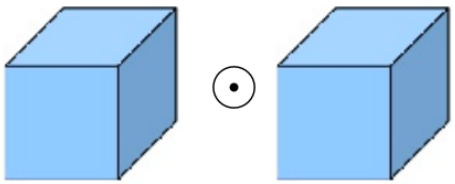


$$C_k = \sum_{i,j} A_{i,j,k}$$

Covariant Compositional Networks (3)

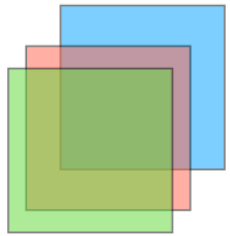
Permutation covariant operators (continued):

4. Hadamard products



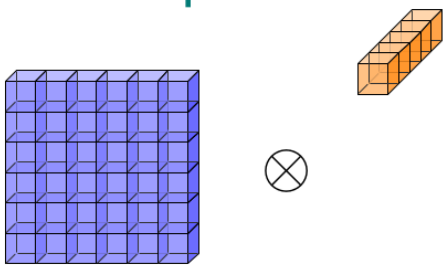
$$C_{i,j,k} = A_{i,j,k} B_{i,j,k}$$

5. Stacking



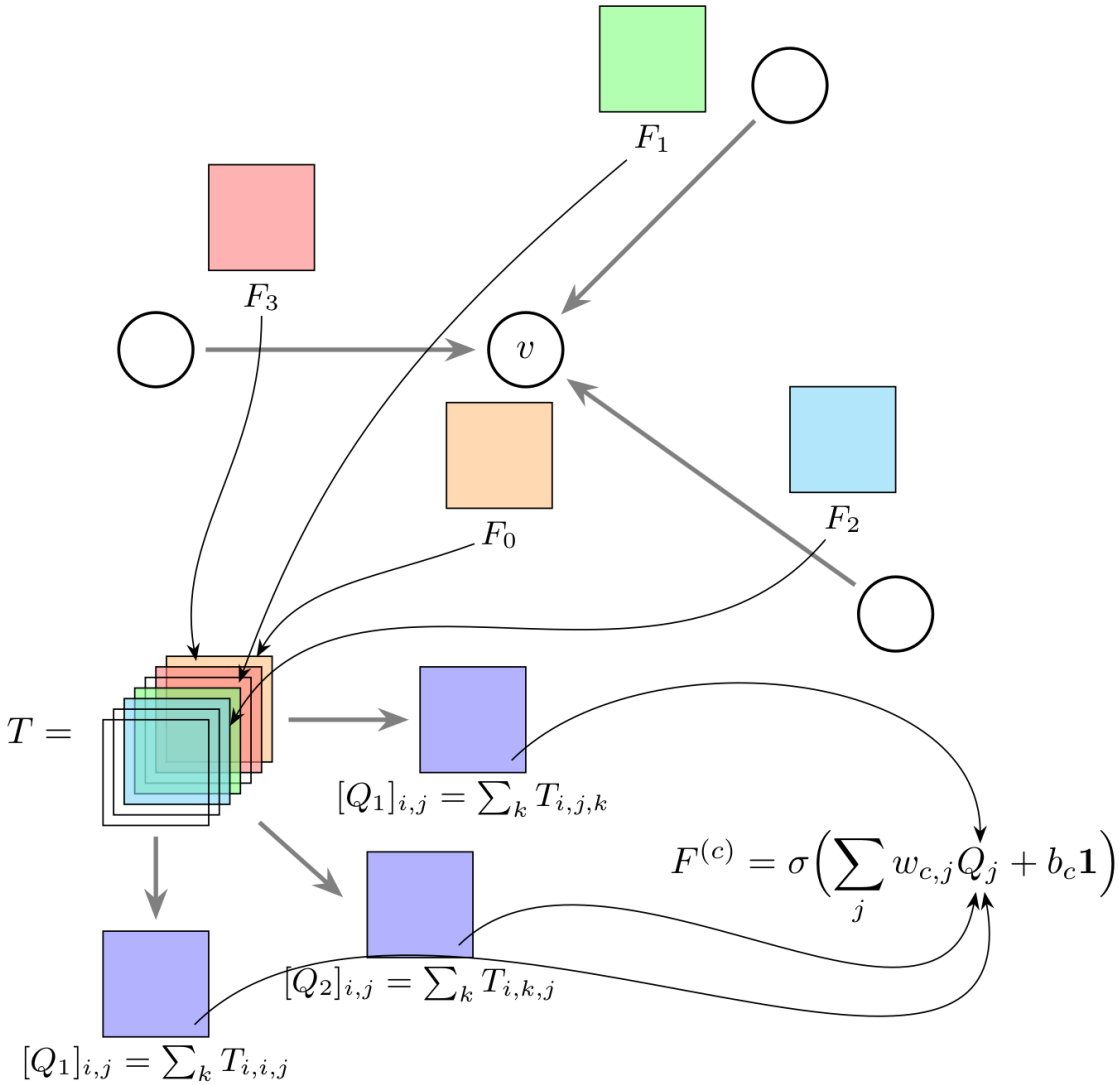
$$C_{i,j,k} = A^{(k)}_{i,j}$$

6. Tensor products



$$C_{i,j,k} = A_{i,j} B_k$$

Covariant Compositional Networks (4)

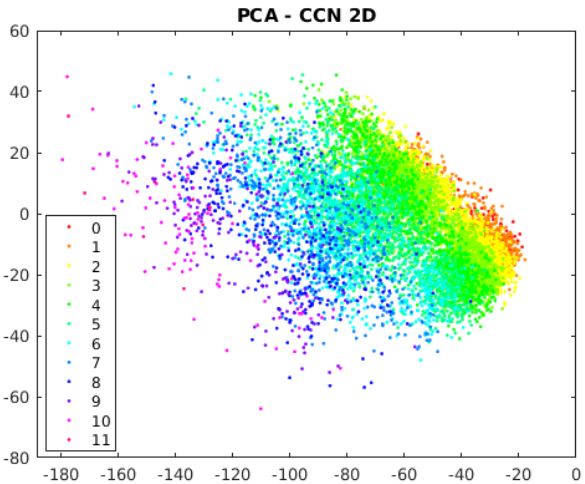


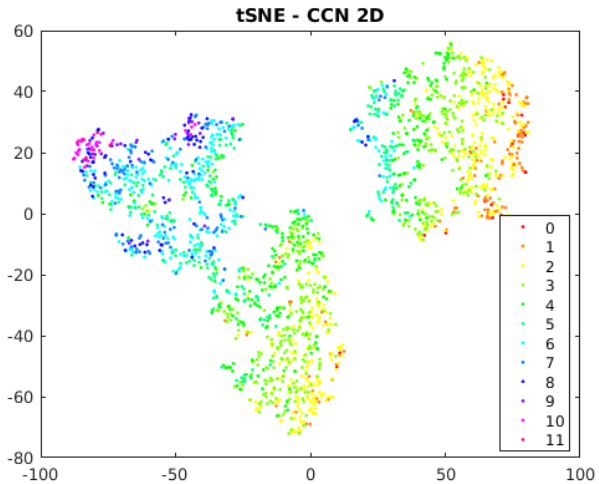
Second-order CCN

Covariant Compositional Networks (5)

	Test MAE	Test RMSE
Lasso	0.867	1.437
Ridge regression	0.854	1.376
Random forest	1.004	1.799
Gradient boosted trees	0.704	1.005
Weisfeiler–Lehman kernel[33]	0.805	1.096
Neural graph fingerprints[21]	0.851	1.177
PSCN ($k = 10$)[37]	0.718	0.973
Second order CCN (our method)	**0.340**	**0.449**

TABLE II. HCEP regression results. Error of predicting power conversion efficiency in units of percent.





(Kondor et al., 2018), (Hy et al., 2018)

Covariant Compositional Networks (6)

	CCN	DFT error
α (Bohr^3)	**0.22**	0.4
C_v (cal/(mol K))	**0.07**	0.34
G (eV)	**0.06**	0.1
GAP (eV)	**0.12**	1.2
H (eV)	**0.06**	0.1
HOMO (eV)	**0.09**	2.0
LUMO (eV)	**0.09**	2.6
μ (Debye)	0.48	**0.1**
ω_1 (cm^{-1})	**2.81**	28
R_2 (Bohr^2)	4.00	-
U (eV)	**0.06**	0.1
U_0 (eV)	**0.05**	0.1
ZPVE (eV)	**0.0039**	0.0097

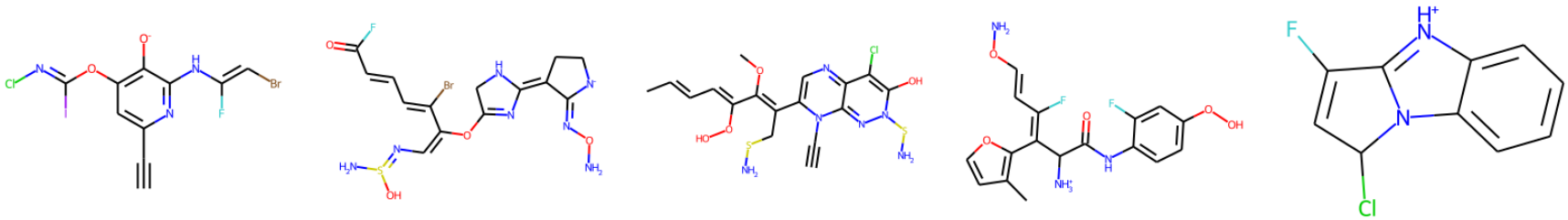
TABLE IV. The mean absolute error of CCN compared to DFT error when using the complete set of physical features used in Ref. 25 in addition to the graph of each molecule.

(Kondor et al., 2018), (Hy et al., 2018)

Graph generation (1)

Goal

Design models that can observe a set of graphs $\{\mathcal{G}_1, .., \mathcal{G}_n\}$ and learn to generate graphs with similar characteristics as this training set.



Look-alike molecules generated from Multiresolution VAE trained on ZINC dataset

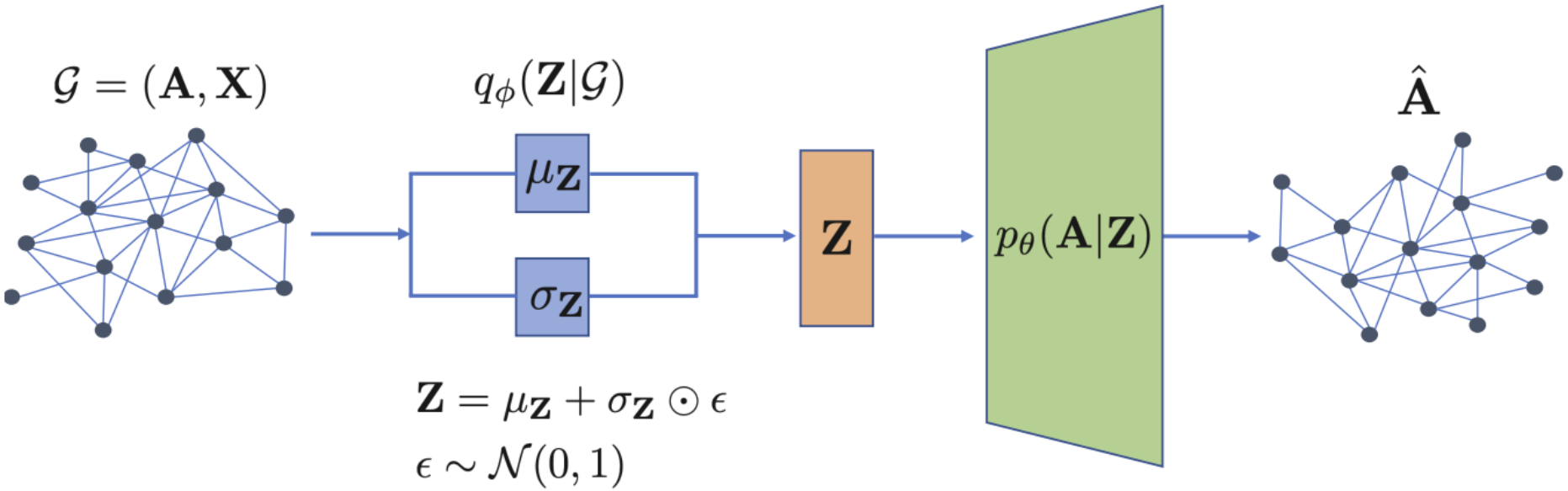
Graph generation (2)

Methods:

1. Traditional graph generation approaches:
   - Erdös-Rényi (ER) Model
   - Stochastic Block Models (SBM)

   Limitation: Rely on a fixed, hand-crafted generation process. Lack the ability to learn a generative model from data.
2. Deep generative models:
   - **All-at-once**:
     - Generate the whole adjacency matrix with all node (atomic) features
     - VAEs, GANs
   - **Autoregressive**:
     - Generate a graph *incrementally* by adding one node (atom) or one edge (bond) at a time
     - Reinforcement Learning, LSTM-based language models, GRNN, GRANs, etc.

Graph Variational Autoencoder



Graph Representation Learning, William L. Hamilton (McGill University, 2020) `https://www.cs.mcgill.ca/~wlh/grl_book/`

Why we need equivariant generation? (1)

Suppose that we want to map the latent vector $\boldsymbol{z}_{\mathcal{G}}$ to a matrix $\hat{\boldsymbol{A}} \in [0,1]^{|\mathcal{V}|\times|\mathcal{V}|}$ of edge probabilities. The posterior distribution:

$$p_\theta(\mathcal{G}|\boldsymbol{z}_{\mathcal{G}}) = \prod_{(u,v)\in\mathcal{V}\times\mathcal{V}} \hat{\boldsymbol{A}}_{u,v}\boldsymbol{A}_{u,v} + (1-\hat{\boldsymbol{A}}_{u,v})(1-\boldsymbol{A}_{u,v})$$

where $\boldsymbol{A}$ denotes the true adjacency, and $\hat{\boldsymbol{A}}$ denotes the predicted edge probabilities.

Problem

But we do **not** know the correct ordering of nodes.

Why we need equivariant generation? (2)

Graph matching problem (**NP-hard**, quadratic assignment problem):

$$p_\theta(\mathcal{G}|\boldsymbol{z}_\mathcal{G}) = \max_{\pi \in \Pi} \prod_{(u,v)\in\mathcal{V}\times\mathcal{V}} \hat{\boldsymbol{A}}^{\pi}_{u,v}\boldsymbol{A}_{u,v} + (1-\hat{\boldsymbol{A}}^{\pi}_{u,v})(1-\boldsymbol{A}_{u,v})$$

Approximate solution: Specify a set of particular orderings $\{\pi_1, .., \pi_n\}$ (this is also how autoregressive methods work in practice)

$$p_\theta(\mathcal{G}|\boldsymbol{z}_\mathcal{G}) \approx \sum_{\pi_i \in \{\pi_1,..,\pi_n\}} \prod_{(u,v)\in\mathcal{V}\times\mathcal{V}} \hat{\boldsymbol{A}}^{\pi_i}_{u,v}\boldsymbol{A}_{u,v} + (1-\hat{\boldsymbol{A}}^{\pi_i}_{u,v})(1-\boldsymbol{A}_{u,v})$$

Almost perfect solution

- Equivariant (higher-order) latent, encoder, and decoder
- Thiede, Hy & Kondor, 2020

Markov Random Fields

Problem with the prior

- $\mathcal{N}(0, 1)$ is not a good prior for graph generation, because each node (atom)'s latent is sampled **independently**.
- We want a new prior $\mathcal{N}(\mu, \Sigma)$ that is both **learnable** and **equivariant**.
- That also requires a new **reparameterization trick**.
- Hy & Kondor, 2021

Markov network

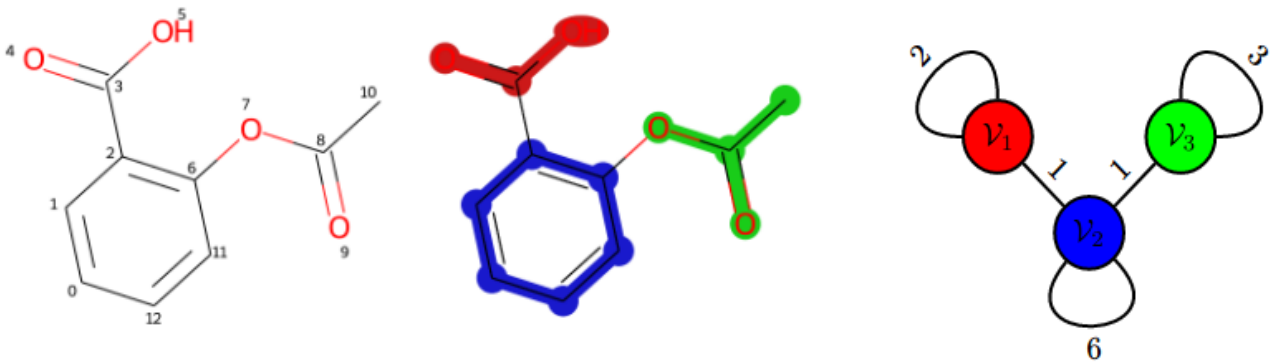
In general, k-th order graph encoders encode an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ into a k-th order latent $\boldsymbol{z} \in \mathbb{R}^{n^k \times d_z}$, with learnable parameters $\boldsymbol{\theta}$, can be represented as a parameterized Markov Random Field (MRF) or Markov network.

Multiresolution Graph Network (1)

What we want more?

Learning and then generating graphs in multiple levels of granulity.

The backbone of this coarse-graining architecture, Multiresolution Graph Network (MGN), is the **Learning to cluster** algorithm. The hard clustering can be differentible (for back-propagation) by the Gumbel-softmax trick.



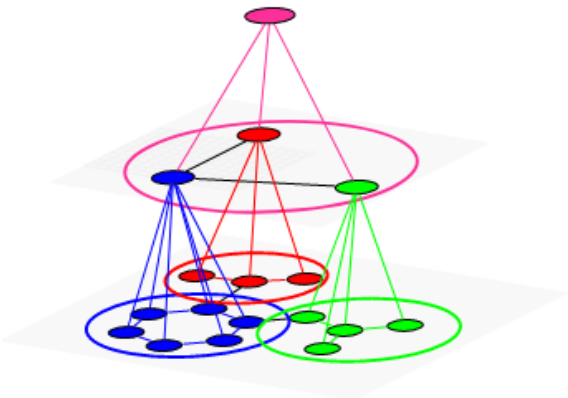
Aspirin $C_9H_8O_4$, its 3-cluster partition and the corresponding coarsen graph.

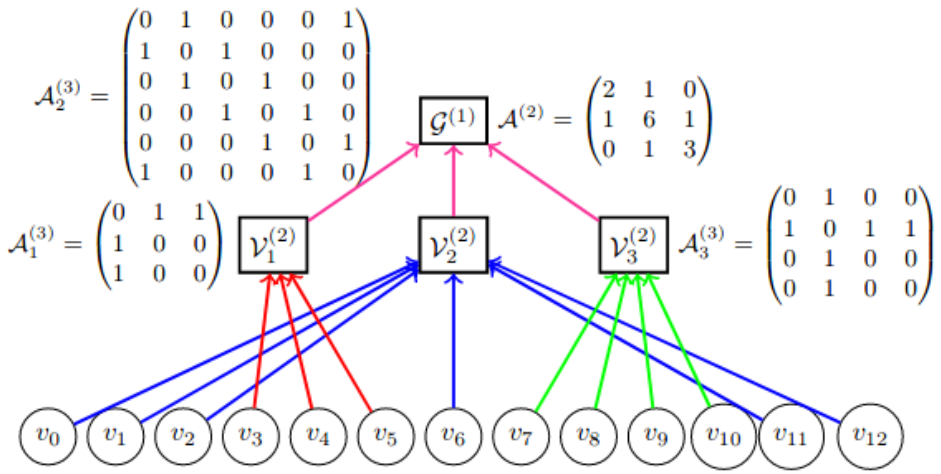
Multiresolution Graph Network (2)

It is desirable to have a *balanced* K-cluster partition in which clusters $\mathcal{V}_1^{(\ell)}, .., \mathcal{V}_K^{(\ell)}$ (at the ℓ-th resolution level) have similar sizes that are close to $|\mathcal{V}^{(\ell)}|/K$.

We enforce the clustering procedure to produce a balanced cut by minimizing the following Kullback–Leibler divergence:

$$\mathcal{D}_{KL}(P||Q) = \sum_{k=1}^{K} P(k) \log \frac{P(k)}{Q(k)}, \quad P = \left(\frac{|\mathcal{V}_1^{(\ell)}|}{|\mathcal{V}^{(\ell)}|}, .., \frac{|\mathcal{V}_K^{(\ell)}|}{|\mathcal{V}^{(\ell)}|} \right), \quad Q = \left(\frac{1}{K}, .., \frac{1}{K} \right)$$





Multiresolution Equivariant Graph VAE (1)

Based on the construction of multiresolution graph network, the latent hierarchy is partitioned into disjoint groups, $\mathcal{Z}_i = \{\mathcal{Z}_i^{(1)}, \mathcal{Z}_i^{(2)}, .., \mathcal{Z}_i^{(L)}\}$ where $\mathcal{Z}_i^{(\ell)}$ is the set of latents at the ℓ-th resolution level. We employ the use of **hierarchical VAEs**.

We write our multiresolution variational lower bound $\mathcal{L}_{\text{MGVAE}}(\phi, \theta)$ on $\log p(\mathcal{G})$ compactly as

$$\mathcal{L}_{\text{MGVAE}}(\phi, \theta) = \sum_i \sum_\ell \Big[\mathbb{E}_{q_\phi(\mathcal{Z}_i^{(\ell)} | \mathcal{G}_i^{(\ell)})} [\log p_\theta(\mathcal{G}_i^{(\ell)} | \mathcal{Z}_i^{(\ell)})] - \mathcal{D}_{\text{KL}}(q_\phi(\mathcal{Z}_i^{(\ell)} | \mathcal{G}_i^{(\ell)}) || p_0(\mathcal{Z}_i^{(\ell)})) \Big]$$

Multiresolution Equivariant Graph VAE (2)

In general, the overall optimization is given as follows:

$$\min_{\phi,\theta,\{\hat{\boldsymbol{\mu}}^{(\ell)},\hat{\boldsymbol{\Sigma}}^{(\ell)}\}_\ell} \mathcal{L}_{\text{MGVAE}}(\phi,\theta;\{\hat{\boldsymbol{\mu}}^{(\ell)},\hat{\boldsymbol{\Sigma}}^{(\ell)}\}_\ell) + \sum_{i,\ell} \lambda^{(\ell)} \mathcal{D}_{\text{KL}}(P_i^{(\ell)} || Q_i^{(\ell)}),$$

where

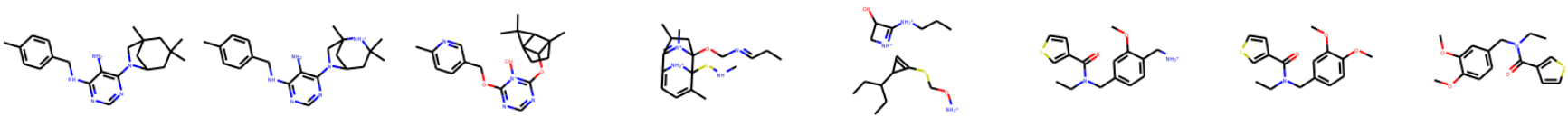
- ϕ denotes all learnable parameters of the encoders,
- θ denotes all learnable parameters of the decoders,
- $\mathcal{D}_{\text{KL}}(P_i^{(\ell)} || Q_i^{(\ell)})$ is the balanced-cut loss for graph $\mathcal{G}_i$ at level ℓ,
- $\hat{\boldsymbol{\mu}}^{(\ell)}$and $\hat{\boldsymbol{\Sigma}}^{(\ell)}$ are learnable parameters of the prior in an equivariant manner.

Multiresolution Equivariant Graph VAE (3)

Dataset	Method	Training size	Input features	Validity	Novelty	Uniqueness
QM9	GraphVAE	~ 100K	Graph	61.00%	85.00%	40.90%
	CGVAE			100%	94.35%	98.57%
	MolGAN			98.1%	94.2%	10.4%
	Autoregressive MGN	10K		100%	95.01%	97.44%
	All-at-once MGVAE			100%	100%	95.16%
ZINC	GraphVAE	~ 200K	Graph	14.00%	100%	31.60%
	CGVAE			100%	100%	99.82%
	JT-VAE			100%	-	-
	Autoregressive MGN	1K		100%	99.89%	99.69%
	All-at-once MGVAE	10K	Chemical	99.92%	100%	99.34%

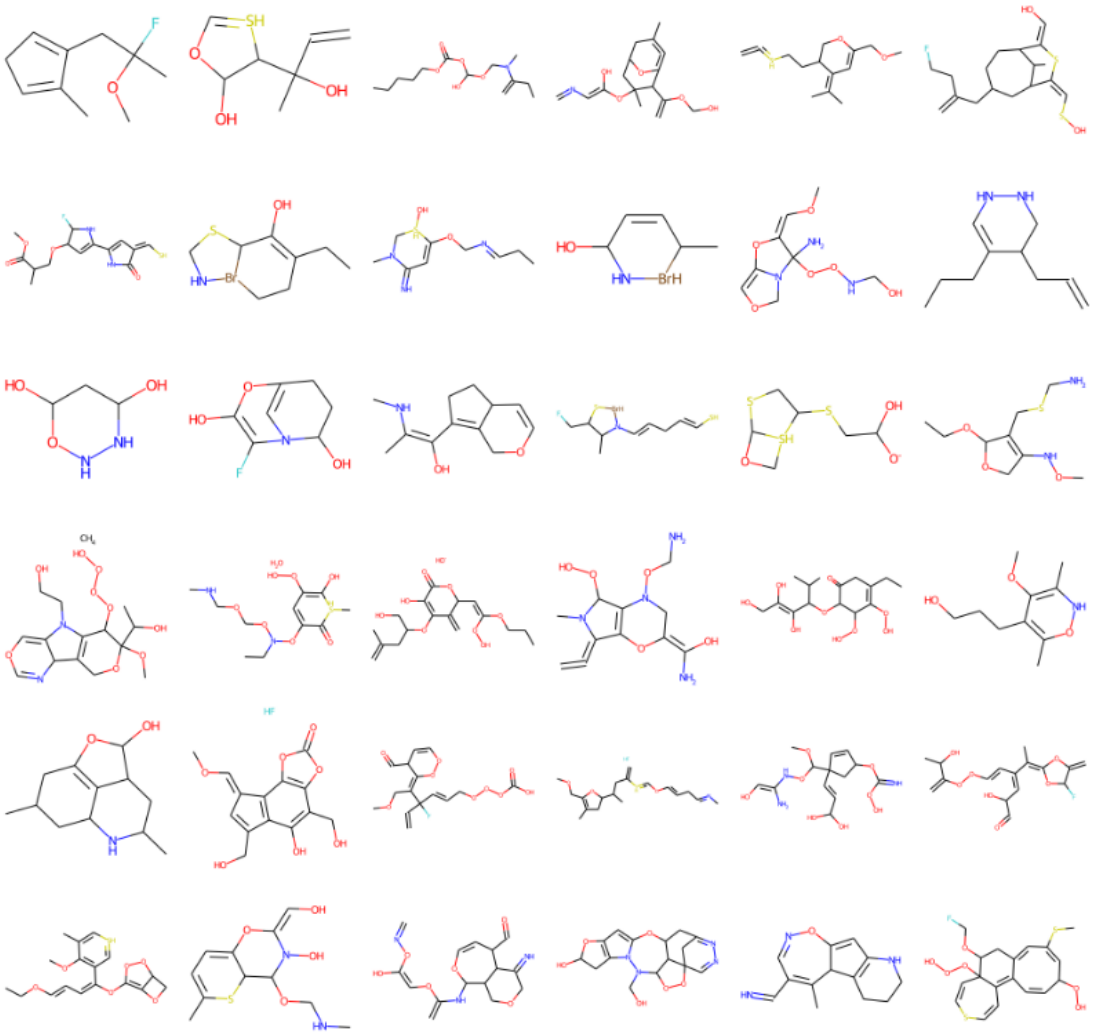
Table 1: Molecular graph generation results. GraphVAE results are taken from (Liu et al., 2018).

Interpolation on the latent:



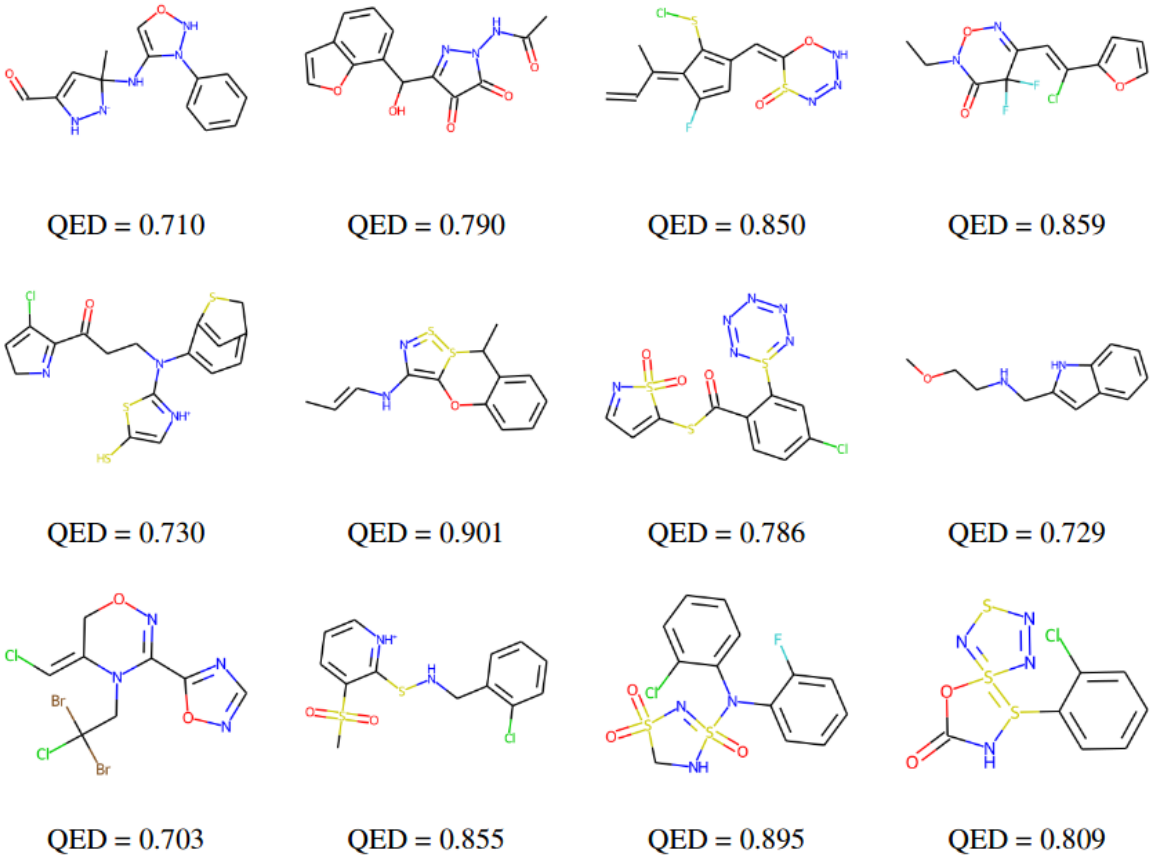
(Hy & Kondor, 2021)

Multiresolution Equivariant Graph VAE (4)



Some generated examples on ZINC by the all-at-once MGVAE with second order $\mathbb{S}_n$-equivariant decoders.

Multiresolution Equivariant Graph VAE (5)



Some generated molecules on ZINC by the autoregressive MGN with high QED (drug-likeness score).

Future works

1. **Drug discovery:** Application of deep generative models on graphs into the lead optimization process that enhances the most promising compounds to improve effectiveness, safety and tolerability.
2. **Material science:** Constrained generative models to generate stable crystal structures by optimizing the formation energy in the latent space.
3. **Proteins:** Multiscale modeling of proteins for the purpose of function prediction and protein design.