Junction Tree Variational Autoencoder for Molecular Graph Generation

Wengong Jin, Regina Barzilay and Tommi Jaakkola

Presentation adapted from slides by: Wengong Jin
Presenter: Yevgeny Tkach

2019 Spring @
https://qdata.github.io/deep2Read/
Executive Summary

• Molecule generation using VAE. Encoding and decoding is based on spacial graph message passing algorithm.
• Instead of generating the molecule node by node which can be looked at as “character level” generation, this work builds higher level vocabulary based on tree decomposition of the molecule graph.
• Using proper “words/parts of speech” helps to make sure that the final molecule is valid.
Drug Discovery

Generate molecules with high potency
Drug Discovery

Modify molecules to increase potency
Molecular Variational Autoencoder

Encoder

Decoder

Bayesian optimization over latent space

Find “best” drugs

Potency Prediction

Gradient ascent over latent space

Make “better” drugs

How to generate graphs?

- Not every graph is chemically valid
- Invalid intermediate states $\rightarrow$ hard to validate
- Very long intermediate steps $\rightarrow$ difficult to train (Li et al., 2018)

---

[2] Li et al., Learning Deep Generative Models of Graphs, 2018
Functional Group

Aromatic rings

Functional Groups

N
O
Cl
S
N
O
S
Cl
S
How to generate graphs?

Node by Node
- Valid
- Invalid
- Invalid
- Invalid
- Valid

Group by Group
- Valid
- Valid
- Valid

• Shorter action sequence
• Easy to check validity
Tree Decomposition

- Generate junction tree

- Generate graph group by group

- Vocabulary size: less than 800 given 250K molecules
Our Approach

Molecule

Encode

Decode

Molecular Graph $G$

Junction Tree $T$

Tree Decomposition

Clusters
Graph & Tree Encoder

Neural Message Passing Network (MPN)
Graph Encoding

Graph Encoding

Graph Encoding

Graph Encoding

\[ \nu_{uv}^{(t)} = \tau(W_1^g x_u + W_2^g x_{uv} + W_3^g \sum_{w \in N(u) \setminus v} \nu_{wu}^{(t-1)}) \]

Graph Encoding

\[ h_u = \tau \left( U_1^g x_u + \sum_{v \in N(u)} U_2^g \nu_{vu}^{(T)} \right) \]

To capture long range interactions

\[ m_{ij} = \text{GRU}(x_i, \{m_{ki}\}_{k \in N(i) \setminus j}) \]
Graph & Tree Encoder

\[ z_G \]  
\[ z_T \]  

average-pooling

root node
Tree Decoder

Label Prediction

**Tree Decoder**

**1. Topological Prediction**: Whether to expand a child or backtrack?

**2. Label Prediction**: What is the label of a node?

**Message vector**

**Topological Prediction**: Whether to expand a child or backtrack?

**Label Prediction**: What is the label of a node?
Tree Decoder

Topological Prediction: Whether to expand a node or backtrack?

Label Prediction: What is the label of a node?
Tree Decoder

\[ h_{ij} = \text{GRU}(x_i, \{h_{ki}\}_{k \in N_t(i) \setminus j}) \]

Encodes the entire subtree of current state
Tree Decoder

**Algorithm 1** Tree decoding at sampling time

**Require:** Latent representation $z_T$

1. **Initialize:** Tree $\hat{T} \leftarrow \emptyset$
2. **function** SampleTree($i, t$)
   3. Set $\mathcal{X}_i \leftarrow$ all cluster labels that are chemically compatible with node $i$ and its current neighbors.
   4. Set $d_t \leftarrow$ expand with probability $p_t$. \(\triangleright\) Eq.(11)
   5. **if** $d_t = \text{expand and } \mathcal{X}_i \neq \emptyset$ **then**
   6. Create a node $j$ and add it to tree $\hat{T}$.
   7. Sample the label of node $j$ from $\mathcal{X}_i$. \(\triangleright\) Eq.(12)
   8. SampleTree($j, t + 1$)
   9. **end if**
10. **end function**

\[
p_t = \sigma(u^d \cdot \tau(W_1^d x_i + W_2^d z_T + W_3^d \sum_{(k,i_t) \in \tilde{E}_t} h_{k,i_t}) \quad (11)
\]

\[
q_j = \text{softmax}(U^l \tau(W_1^l z_T + W_2^l h_{i,j})) \quad (12)
\]

\[
L_c(T) = \sum_t L^d(p_t, \hat{p}_t) + \sum_j L^l(q_j, \hat{q}_j) \quad (13)
\]
Graph Decoder

Predicted Junction Tree

Molecular Graph
Graph Decoder

Enumerate how clusters are merged together

Encode each candidate graph by graph encoder

Score each candidate:
\[ f_i^a(G_i) = h_{G_i} \cdot z_G \]

Enumerated subgraphs \( G_i \)

Graph encoder

\[ \mathcal{L}_g(G) = \sum_i \left[ f^a(G_i) - \log \sum_{G' \in G_i} \exp(f^a(G'_i)) \right] \]  \hspace{1cm} (16)
Training? VAE?

- The KL divergence part on the latent space is not discussed in the paper.

- $z_G$ is only used for generated subgraphs ranking so not clear how it falls in the VAE paradigm.

- From the code, training is with KL annealing following “Generating Sentences from a continuous space” paper by Bowman et al.
Experiments

- **Data**: 250K compounds from ZINC dataset

- **Molecule Generation**: How many molecules are valid when sampled from Gaussian prior?

- **Molecule Optimization**
  - **Global**: Find the best molecule in the entire latent space.
  - **Local**: Modify a molecule to increase its potency
Baselines

SMILES string based:

1. Grammar VAE (GVAE) (Kusner et al., 2017);
2. Syntax-directed VAE (SD-VAE) (Dai et al., 2018)

Graph based:

1. Graph VAE (Simonovsky & Komodakis, 2018)
2. DeepGMG (Li et al., 2018)

[2] Li et al., Learning Deep Generative Models of Graphs, 2018
[7] Simonovsky & Komodakis, GraphVAE: Towards generation of small graphs using variational autoencoders
Molecule Generation (Validity)

- GVAE: 7.2
- GraphVAE: 13.5
- SD-VAE: 43.5
- DeepGMG: 89.2
- Ours (w/o checking): 93.5
- Ours: 100
Sampled Molecules
Molecule Optimization (Global)

Property Prediction

Bayesian Optimization

Gaussian Process

Encoder

Decoder

Property: Solubility + Ease of Synthesis

Property Score of the Best Molecule

CVAE: 1.98
GVAE: 2.94
SD-VAE: 4.04
Ours: 5.3
Molecule Optimization (Global)

Property Score of the Best Molecule

Property: Solubility + Ease of Synthesis
Molecule Optimization (Local)

Property Prediction

Encoder

Decoder

Gradient Ascent

Neural Network

Average Improvement

<table>
<thead>
<tr>
<th>Preservation</th>
<th>Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.91</td>
</tr>
<tr>
<td>0.2</td>
<td>1.68</td>
</tr>
<tr>
<td>0.4</td>
<td>0.84</td>
</tr>
<tr>
<td>0.6</td>
<td>0.21</td>
</tr>
</tbody>
</table>
Molecule Optimization (Local)

Preservation ≈ 0.6

Average Improvement

Preservation

- 1.91
- 1.68
- 0.84
- 0.21

0 0.2 0.4 0.6
Molecule Optimization (Local)

Preservation \approx 0.4

Average Improvement

<table>
<thead>
<tr>
<th>Preservation</th>
<th>Improvement</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1.91</td>
</tr>
<tr>
<td>0.2</td>
<td>1.68</td>
</tr>
<tr>
<td>0.4</td>
<td>0.84</td>
</tr>
<tr>
<td>0.6</td>
<td>0.21</td>
</tr>
</tbody>
</table>
Discussion

• “word level” prediction can offer significant improvement by shortening the decision process.
• Latent space optimization is an interesting and powerful technique.
• “Teacher forcing” introduces data bias which can be reduced via RL techniques and the GAN complete graph valuation approach.
• Similar to SMILES this paper samples a random order in the graph tree structure when: using an arbitrary minimal spanning tree, choosing an arbitrary node to be the root of the tree, choosing a random ordering of the children of each tree node.
Thanks

Original code is available at: https://github.com/wengong-jin/icml18-jtnn