Hierarchical Generation of Molecular Graphs using Structural Motifs

Wengong Jin ¹  Regina Barzilay ¹  Tommi Jaakkola ¹

Abstract

Graph generation techniques are increasingly being adopted for drug discovery. Previous graph generation approaches have utilized relatively small molecular building blocks such as atoms or simple cycles, limiting their effectiveness to smaller molecules. Indeed, as we demonstrate, their performance degrades significantly for larger molecules. In this paper, we propose a new hierarchical graph encoder-decoder that employs significantly larger and more flexible graph motifs as basic building blocks. Our encoder produces a multi-resolution representation for each molecule in a fine-to-coarse fashion, from atoms to connected motifs. Each level integrates the encoding of constituents below with the graph at that level. Our autoregressive coarse-to-fine decoder adds one motif at a time, interleaving the decision of selecting a new motif with the process of resolving its attachments to the emerging molecule. We evaluate our model on multiple molecule generation tasks, including polymers, and show that our model significantly outperforms previous state-of-the-art baselines.

1. Introduction

Deep learning models for molecule property prediction and molecule generation are improving at a fast pace. Work to date has adopted primarily two types of building blocks for representing and building molecules: atom-by-atom strategies (Li et al., 2018; You et al., 2018a; Liu et al., 2018), or substructure based (either rings or bonds) (Jin et al., 2018; 2019). While these methods have been successful for small molecules, their performance degrades significantly for larger molecules such as polymers (see Figure 1). The failure is likely due to many generation steps required to realize larger molecules and the associated challenges with gradients across the iterative steps.

Large molecules such as polymers exhibit clear hierarchical structure, being built from repeated structural motifs. We hypothesize that explicitly incorporating such motifs as building blocks in the generation process can significantly improve reconstruction and generation accuracy, as already illustrated in Figure 1. While different substructures as building blocks were considered in previous work (Jin et al., 2018), their approach could not scale to larger motifs. Indeed, their decoding process required each substructure neighborhood to be assembled in one go, making it combinatorially challenging to handle large components with many possible attachment points.

In this paper, we propose a motif-based hierarchical encoder-decoder for graph generation. The motifs themselves are extracted separately at the outset from frequently occurring substructures, regardless of size. During generation, molecules are built step by step by attaching motifs, large or small, to the emerging molecule. The decoder operates hierarchically, in a coarse-to-fine manner, and makes three key consecutive predictions in each pass: new motif selection, which part of it attaches, and the points of contact with the current molecule. These decisions are highly coupled and naturally modeled auto-regressively. Moreover, each decision is directly guided by the information explicated in the associated layer of the mirroring hierarchical encoder. The feed-forward fine-to-coarse encoding performs iterative graph convolutions at each level, conditioned on the results from layer below.

The proposed model is evaluated on various tasks ranging from polymer generative modeling to graph translation for molecule property optimization.¹ Our baselines include state-of-the-art graph generation methods (You et al., 2018a; Liu et al., 2018; Jin et al., 2019). On polymer generation, our model achieved state-of-the-art results under various metrics, outperforming the best baselines with 20% absolute improvement in reconstruction accuracy. On graph translation tasks, our model outperformed all the baselines, yielding 3.3% and 8.1% improvement on QED and DRD2 optimization tasks. We further conduct ablation studies to validate the advantage of using larger motifs and the proposed hierarchical architecture.

¹MIT CSAIL. Correspondence to: Wengong Jin <wengong@csail.mit.edu>.


¹https://github.com/wengong-jin/hgraph2graph
2. Background and Motivation

Molecules are represented as graphs $G = (V, E)$ with atoms $V$ as nodes and bonds $E$ as edges. Graphs are challenging objects to generate, especially for larger molecules such as polymers. For the polymer dataset used in our experiment, there are thousands of molecules with more than 80 atoms. To illustrate the challenge, we tested two state-of-the-art variational autoencoders (Liu et al., 2018; Jin et al., 2018) on this dataset and found these models often fail to reconstruct molecules from their latent embedding (see Figure 1).

The reason of this failure is that these methods generate molecules based on small building blocks. In terms of autoregressive models, previous work on molecular graph generation can be roughly divided in two categories:

- Atom-based methods (Li et al., 2018; You et al., 2018a; Liu et al., 2018) generate molecules atom by atom.
- Substructure-based methods (Jin et al., 2018; 2019) generates molecules based on small substructures restricted to rings and bonds (often no more than six atoms).

As the building blocks are typically small, it requires many decoding steps for current models to reconstruct polymers. Therefore they are prone to make errors when generating large molecules. On the other hand, many of these molecules consist of structural motifs beyond simple substructures. The number of decoding steps can be significantly reduced if graphs are generated motif by motif. As shown in Figure 1, our motif-based method achieves a much higher reconstruction accuracy.

**Motivation for New Architecture** Current substructure-based method (Jin et al., 2018) requires a combinatorial enumeration to assemble substructures whose time complexity is exponential to substructure size. Their enumeration algorithm assumes the substructures to be of certain types (single cycles or bonds). In practice, their method often fails when handling rings with more than 10 atoms (e.g., memory error). Unlike substructures, motifs are typically much larger and can have flexible structures (see Figure 1). As a result, this method cannot be directly extended to utilize motifs in practice.

To this end, we propose a hierarchical encoder-decoder for graph generation. Our decoder allows arbitrary types of motifs and can assemble them efficiently without combinatorial explosion. Our encoder learns a hierarchical representation that allows the decoding process to depend on both coarse-grained motif and fine-grained atom connectivity.

2.1. Motif Extraction

We define a motif $S_i = (V_i, E_i)$ as a subgraph of molecule $G$ induced by atoms in $V_i$ and bonds in $E_i$. Given a molecule, we extract its motifs $S_1, \cdots, S_n$ such that their union covers the entire molecular graph: $V = \bigcup_i V_i$ and $E = \bigcup_i E_i$. To extract motifs, we decompose a molecule $G$ into disconnected fragments by breaking all the bridge bonds that will not violate chemical validity (illustrations in the appendix).

1. Find all the bridge bonds $(u, v) \in E$, where both $u$ and $v$ have degree $\Delta_u, \Delta_v \geq 2$ and either $u$ or $v$ is part of a ring. Detach all the bridge bonds from its neighbors.
2. Now the graph $G$ becomes a set of disconnected subgraphs $G_1, \cdots, G_N$. Select $G_i$ as motif in $G$ if its occurrence in the training set is more than $\Delta = 100$.
3. If $G_i$ is not selected as motif, further decompose it into rings and bonds and select them as motif in $G$.

We apply the above procedure to all the molecules in the training set and construct a vocabulary of motifs $V_S$. In the following section, we will describe how we encode and decode molecules using the extracted motifs.
Hierarchical Generation of Molecular Graphs using Structural Motifs

3. Hierarchical Graph Generation

Our approach extends the variational autoencoder (Kingma & Welling, 2013) to molecular graphs by introducing a hierarchical decoder and a matching encoder. In our framework, the probability of a graph $G$ is modeled as a joint distribution over structural motifs $S_1, \ldots, S_n$ constituting $G$, together with their attachments $A_1, \ldots, A_n$. Each attachment $A_i = \{v_j \mid v_j \in \bigcup_k S_i \cap S_k\}$ indicates the intersecting atoms between $S_i$ and its neighbor motifs. To capture complex dependencies involved in the joint distribution of motifs and their attachments, we propose an auto-regressive factorization of $P(G)$:

$$P(G) = \int_z P(z) \prod_k P(S_k, A_k \mid S_{<k}, A_{<k}, z) dz \quad (1)$$

As illustrated in Figure 3, in each generation step, our decoder adds a new motif $S_k$ (motif prediction) and its attachment configuration $A_k$ (attachment prediction). Then it decides how the new motif should be attached to the current graph (graph prediction).

To support the above hierarchical generation, we need to design a matching encoder representing molecules at multiple resolutions in order to provide necessary information for each decoding step. Therefore, we propose to represent a molecule $G$ by a hierarchical graph $\mathcal{H}_G$ with three layers (see Figure 2):

1. **Motif layer**: This layer represents how the motifs are coarsely connected in the graph. This layer provides essential information for the motif prediction in the decoding process. Specifically, this layer contains $n$ nodes $S_1, \ldots, S_n$ and $m$ edges $\{(S_i, S_j) \mid S_i \cap S_j \neq \emptyset\}$ for all intersecting motifs $S_i, S_j$. This layer is tree-structured due to our way of constructing motifs.

2. **Attachment layer**: This layer encodes the connectivity between motifs at a fine-grained level. Each node $A_i = \{v_j \mid v_j \in \bigcup_k S_i \cap S_k\}$ in this layer represents a particular attachment configuration of motif $S_i$, where $\{v_j\}$ are atoms in the intersection between $S_i$ and one of its neighbor motifs (see Figure 2). This layer provides crucial information for the attachment prediction step during decoding, which helps reducing the space of candidate attachments between $S_i$ and one of its neighbor motifs (see Figure 2). This layer provides crucial information for the attachment prediction step during decoding, which helps reducing the space of candidate attachments between $S_i$ and one of its neighbor motifs. Just like the motif vocabulary $V_S$, all the attachment configurations of $S_i$ form a motif-specific vocabulary $V_A(S_i)$, which is computed from the training set.$^3$

3. **Atom layer**: The atom layer is the molecular graph $G$ representing how its atoms are connected. Each atom node $v$ is associated with a label $a_v$ indicating its atom type and charge. Each edge $(u, v)$ in the atom layer is

$^3$In our experiments, the average size of attachment vocabulary $|V_A(S_i)| \leq 10$ and the size of motif vocabulary $|V_S| < 500$. 
labeled with $b_{uv}$ indicating its bond type. This layer provides necessary information for the graph prediction step during decoding.

We further introduce edges that connect the atoms and motifs between different layers in order to propagate information in between. In particular, we draw a directed edge from atom $v$ in the atom layer to node $A_i$ in the attachment layer if $v \in S_i$. We also draw edges from $A_i$ to $S_i$ in the motif layer. This gives us the hierarchical graph $\mathcal{H}_G$ for molecule $G$, which will be encoded by a hierarchical message passing network (MPN). During encoding, each node $S_i$ is represented as a one-hot encoding in the motif vocabulary $V_S$. Likewise, each node $A_i$ is represented as a one-hot encoding in the attachment vocabulary $V_A(S_i)$.

### 3.1. Hierarchical Graph Encoder

Our encoder contains three MPNs that encode each of the three layers in the hierarchical graph. For simplicity, we denote the MPN encoding process as $\text{MPN}_{\psi_i}(\cdot)$ with parameter $\psi_i$, and denote $\text{MLP}(x, y)$ as a multi-layer neural network whose input is the concatenation of $x$ and $y$. The details of MPN architecture is listed in the appendix.

#### Atom Layer MPN

We first encode the atom layer of $\mathcal{H}_G$ (denoted as $\mathcal{H}_G^0$). The inputs to this MPN are the embedding vectors $\{e(a_u)\}, \{e(b_{uv})\}$ of all the atoms and bonds in $G$. During encoding, the network propagates the message vectors between different atoms for $T$ iterations and then outputs the atom representation $h_v$ for each atom $v$:

$$c_v^0 = \{h_v\} = \text{MPN}_{\psi_1}(\mathcal{H}_G^0, \{e(a_u)\}, \{e(b_{uv})\})$$

#### Attachment Layer MPN

The input feature of each node $A_i$ in the attachment layer $\mathcal{H}_G^0$ is an concatenation of the embedding $e(A_i)$ and the sum of its atom vectors $\{h_v | v \in S_i\}$:

$$f_{A_i} = \text{MLP}(e(A_i), \sum_{v \in S_i} h_v)$$

The input feature for each edge $(A_i, A_j)$ in this layer is an embedding vector $e(d_{ij})$, where $d_{ij}$ describes the relative ordering between node $A_i$ and $A_j$ during decoding. Specifically, we set $d_{ij} = k$ if node $A_i$ is the $k$-th child of node $A_j$ and $d_{ij} = 0$ if $A_i$ is the parent. Then we run $T$ iterations of message passing over $\mathcal{H}_G^0$ to compute the motif representations:

$$c_i^0 = \{h_{A_i}\} = \text{MPN}_{\psi_2}(\mathcal{H}_G^0, \{f_{A_i}\}, \{e(d_{ij})\})$$

#### Motif Layer MPN

Similarly, the input feature of node $S_i$ in this layer is computed as the concatenation of embedding $e(S_i)$ and the node vector $h_{A_i}$ from the previous layer. Finally, we run message passing over the motif layer $\mathcal{H}_G^0$ to obtain the motif representations:

$$f_{S_i} = \text{MLP}(e(S_i), h_{A_i})$$

Finally, we represent a molecule $G$ by a latent vector $z_G$ sampled through reparameterization trick with mean $\mu(h_{S_1})$ and log variance $\Sigma(h_{S_1})$:

$$z_G = \mu(h_{S_1}) + \exp(\Sigma(h_{S_1})) \cdot \epsilon; \quad \epsilon \sim \mathcal{N}(0, I)$$

where $S_1$ is the root motif (i.e., the first motif to be generated during reconstruction).

### 3.2. Hierarchical Graph Decoder

As illustrated in Figure 3, our graph decoder generates a molecule $G$ by incrementally expanding its hierarchical graph. In $t^{th}$ generation step, we first use the same hierarchical MPN architecture to encode all the motifs and atoms in $\mathcal{H}_G^{(t)}$, the (partial) hierarchical graph generated till step $t$. This gives us motif vectors $h_{S_k}$ and atom vectors $h_{v_j}$ for the existing motifs and atoms.

During decoding, the model maintains a set of frontier nodes $F$ where each node $S_k \in F$ is a motif that still has neighbors to be generated. $F$ is implemented as a stack because motifs are generated in their depth-first order. Suppose $S_k$ is at the top of stack $F$ in step $t$, the model makes the following predictions conditioned on latent representation $z_G$:

1. **Motif Prediction**: The model predicts the next motif $S_t$ to be attached to $S_k$. This is cast as a classification task over the motif vocabulary $V_S$:

$$p_{S_t} = \text{softmax}(\text{MLP}(h_{S_k}, z_G))$$

2. **Attachment Prediction**: Now the model needs to predict the attachment configuration $A_t$ of motif $S_t$ (i.e., what atoms $v_j \in S_t$ belong to the intersection of $S_t$ and its neighbor motifs). This is also cast as a classification task over the attachment vocabulary $V_A(S_t)$:

$$p_{A_t} = \text{softmax}(\text{MLP}(h_{S_k}, z_G))$$

This prediction step is crucial because it significantly reduces the space of possible attachments between $S_t$ and its neighbor motifs.

3. **Graph Prediction**: Finally, the model must decide how $S_t$ should be attached to $S_k$. The attachment between $S_t$ and $S_k$ is defined as atom pairs $M_{tk} = \{(u_j, v_j) | u_j \in A_k, v_j \in A_t\}$ where atom $u_j$ and $v_j$ are attached together. The probability of a candidate attachment $M$ is computed based on the atom vectors $h_{u_j}$ and $h_{v_j}$:

$$p_M = \text{softmax}(h_M \cdot z_G)$$

$$h_M = \sum_j \text{MLP}(h_{u_j}, h_{v_j})$$
Hierarchical Generation of Molecular Graphs using Structural Motifs

The number of possible attachments are limited because the number of attaching atoms between two motifs is small and the attaching points must be consecutive.\(^4\)

The above three predictions together give an autoregressive factorization of the distribution over the next motif and its attachment. Each of the three decoding steps depends on the outcome of previous step, and predicted attachments will in turn affect the prediction of subsequent motifs.

### Training

During training, we apply teacher forcing to the above generation process, where the generation order is determined by a depth-first traversal over the ground truth molecule. Given a training set of molecules, we seek to minimize the negative ELBO:

\[
-\mathbb{E}_{z \sim Q}[\log P(G|z)] + \lambda_{\text{KL}}D_{\text{KL}}[Q(z|G)||P(z)]
\]  (12)

#### 3.3. Extension to Graph-to-Graph Translation

The proposed architecture can be naturally extended to graph-to-graph translation (Jin et al., 2019) for molecular optimization, which seeks to modify compounds in order to improve their biochemical properties. Given a corpus of molecular pairs \{(X, Y)\}, where Y is a structural analog of X with better chemical properties, the model is trained to translate an input molecular graph into its better form. In this case, we seek to learn a translation model \(P(Y|X)\) parameterized by our encoder-decoder architecture. We also introduce attention layers into our model, which is crucial for translation performance (Bahdanau et al., 2014).

**Training** In graph translation, a compound X can be associated with multiple outputs Y since there are many ways to modify X to improve its properties. In order to generate diverse outputs, we follow previous work (Zhu et al., 2017; Jin et al., 2019) and incorporate latent variables \(z\) to the translation model:

\[
P(Y|X) = \int P(Y|X, z)P(z)dz
\]  (13)

where the latent vector \(z\) indicates the intended mode of translation, sampled from a prior \(P(z)\) during testing.

The model is trained as a conditional variational autoencoder. Given a training example \((X, Y)\), we sample \(z\) from the approximate posterior \(Q(z|X, Y) = \mathcal{N}(\mu_{X,Y}, \sigma_{X,Y})\). To compute \(Q(z|X, Y)\), we first encode \(X\) and \(Y\) into their representations \(c_X\) and \(c_Y\) and then compute difference vector \(\delta_{X,Y}\) that summarizes the structural changes from molecule \(X\) to \(Y\) at both atom and motif level:

\[
\delta_{X,Y}^a = \sum c_Y^a - \sum c_X^a \\
\delta_{X,Y}^m = \sum c_Y^m - \sum c_X^m
\]

Finally, we compute \((\mu_{X,Y}, \sigma_{X,Y}) = \text{MLP}(\delta_{X,Y}^a, \delta_{X,Y}^m)\) and sample \(z\) using reparameterization trick. The latent code \(z\) is passed to the decoder along with the input representation \(e_X\) to reconstruct output \(Y\). The training objective is to minimize negative ELBO similar to Eq.(12).

**Attention** For graph translation, the input molecule \(X\) is embedded by our hierarchical encoder into a set of vectors \(e_X = e_X^a \cup e_X^m \cup e_X^g\), representing the molecule at multiple resolutions. These vectors are fed into the decoder through attention mechanisms (Luong et al., 2015). Specifically, we modify the motif prediction (Eq. 8) into

\[
p_{S_i} = \text{softmax}(\text{MLP}(h_{S_i}, \alpha_{X,i}^s, z))
\]

\[
\alpha_{X,i}^s = \text{attention}(h_{S_i}, c_X^s)
\]  (15)

where attention\((h_s, c_X^s)\) is a bilinear attention over vectors \(c_X^s\) with query vector \(h_{S_i}\). The attachment prediction (Eq. 9) is modified similarly with its attention over \(c_X^a\). The graph prediction (Eq. 10) is modified into

\[
p_M = \text{softmax}(h_M \cdot \text{attention}(h_M, c_X^g))
\]

\[
h_M = \sum_j \text{MLP}(h_{a_j}, h_{e_j}, z)
\]  (17)

### 4. Experiments

We evaluate our method on two application tasks. The first task is polymer generative modeling. This experiment validates our argument in section 2 that our model is advantageous when the molecules have large sizes. The second task is graph-to-graph translation for small molecules. Here we show the proposed architecture also brings benefits to small molecules compared to previous state-of-the-art graph generation methods.

#### 4.1. Polymer Generative Modeling

**Dataset** Our method is evaluated on the polymer dataset from St. John et al. (2019), which contains 86K polymers in total (after removing duplicates). The dataset is divided into 76K, 5K and 5K for training, validation and testing. Using our motif extraction, we collected 436 different motifs (examples shown in Figure 4). On average, each motif has 5.24 different attachment configurations. The distribution of motif size and their frequencies are reported in Figure 5.

**Evaluation Metrics** Our evaluation effort measures various aspects of molecule generation proposed in Kusner et al. (2017); Polykovskiy et al. (2018). Besides basic metrics like chemical validity and diversity, we compare distributional statistics between generated and real compounds. A good generative model should generate molecules which present similar aggregate statistics to real compounds. Our metrics include (with details shown in the appendix):

- **Reconstruction accuracy**: We measure how often the model can completely reconstruct a given molecule from its latent embedding \(z\). The reconstruction accuracy is computed over 5K compounds in the test set.

---

\(^4\)In our experiments, the number of possible attachments are usually less than 20 for polymers and small molecules.
Hierarchical Generation of Molecular Graphs using Structural Motifs

Table 1. Results on polymer generative modeling. The first row reports the oracle performance using real data as generated samples. The last row (small motif) is a variant of our model where we restrict the motif vocabulary to contain only single rings and bonds (similar to JT-VAE). “Recon.” means reconstruction accuracy; “Div.” means diversity; SNN means nearest neighbor similarity; “Frag / Scaf” means fragment and scaffold similarity. Except property statistics, all metrics are the higher the better.

<table>
<thead>
<tr>
<th>Method</th>
<th>Reconstruction / Sample Quality (↑)</th>
<th>Property Statistics (↓)</th>
<th>Structural Statistics (↑)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Recon.</td>
<td>Valid</td>
<td>Unique</td>
</tr>
<tr>
<td>Real data</td>
<td></td>
<td>100%</td>
<td>100%</td>
</tr>
<tr>
<td>SMILES</td>
<td>21.5%</td>
<td>93.1%</td>
<td>97.3%</td>
</tr>
<tr>
<td>CG-VAE</td>
<td>42.4%</td>
<td>100%</td>
<td>96.2%</td>
</tr>
<tr>
<td>JT-VAE</td>
<td>58.5%</td>
<td>100%</td>
<td>94.1%</td>
</tr>
<tr>
<td>HierVAE</td>
<td>79.9%</td>
<td>100%</td>
<td>97.0%</td>
</tr>
<tr>
<td>Small motif</td>
<td>71.0%</td>
<td>100%</td>
<td>97.2%</td>
</tr>
</tbody>
</table>

Figure 4. Examples of motif structures utilized by our model. These motifs consist of multiple rings and bonds, which are substantially more complex than previous methods (Jin et al., 2018).

- **Validity**: Percentage of chemically valid compounds.
- **Uniqueness**: Percentage of unique compounds.
- **Diversity**: We compute the pairwise molecular distance among generated compounds. The molecular distance $\text{dist}(X, Y)$ is defined as the Tanimoto distance over Morgan fingerprints (Rogers & Hahn, 2010) of two molecules.
- **Property statistics**: We compare property statistics between generated molecules and real data. Our properties include partition coefficient (logP), synthetic accessibility (SA), drug-likeness (QED) and molecular weight (MW). To quantitatively evaluate the distance between two distributions, we compute Frechet distance between property distributions of molecules in the generated and test sets (Polykovskiy et al., 2018).
- **Structural statistics**: We also compute structural statistics between generated molecules and real data. Nearest neighbor similarity (SNN) is the average similarity of generated molecules to the nearest molecule from the test set. Fragment similarity (Frag) and scaffold similarity (Scaf) are cosine distances between vectors of fragment or scaffold frequencies of the generated and the test set.

**Baselines** We compare our method against three state-of-the-art variational autoencoders for molecular graphs. SMILES VAE (Gómez-Bombarelli et al., 2018) is a sequence to sequence VAE that generates molecules based on their SMILES strings (Weininger, 1988). CG-VAE (Liu et al., 2018) is a graph-based VAE generating molecules atom by atom. JT-VAE (Jin et al., 2018) is also a graph-based VAE generating molecules based on simple substructures restricted to rings and bonds. Finally, we report the oracle performance of distributional statistics by using real molecules in the training set as our generated samples.

4.1.1. RESULTS

The performance of different methods are summarized in Table 1. Our method (HierVAE) significantly outperforms all previous methods in terms of reconstruction accuracy (79.9% vs 58.5%). This validates the advantage of utilizing large structural motifs, which reduces the number of generation steps. In terms of distributional statistics, our method achieves state-of-the-art results on logP (0.525 vs 1.471), molecular weight Frechet distance (1928 vs 4863) and all the structural similarity metrics. Since our model requires fewer generation steps, our training speed is much faster than other graph-based methods (see Figure 5).
Ablation Study To validate the importance of utilizing large structural motifs, we further experiment a variant of our model (small motif), which keeps the same architecture but replaces the large structural motifs with basic substructures such as rings and bonds (with less than ten atoms). As shown in Table 1, its performance is significantly worse than our full model even though it builds on the same hierarchical architecture.

4.2. Graph-to-Graph Translation

We follow the experimental design by Jin et al. (2019) and evaluate our model on their graph-to-graph translation tasks. Following their setup, we require the molecular similarity between X and output Y to be above certain threshold $\text{sim}(X, Y) \geq \delta$ at test time. This is to prevent the model from ignoring input X and translating it into arbitrary compound. Here the molecular similarity is defined as $\text{sim}(X, Y) = 1 - \text{dist}(X, Y)$.

Dataset The dataset consists of four property optimization tasks. In each task, we train and evaluate our model on their provided training and test sets.

- **LogP**: The penalized logP score (Kusner et al., 2017) measures the solubility and synthetic accessibility of a compound. In this task, the model needs to translate input X into output Y such that $\logP(Y) > \logP(X)$. We experiment with two similarity thresholds $\delta = \{0.4, 0.6\}$.

- **QED**: The QED score (Bickerton et al., 2012) quantifies a compound’s drug-likeness. In this task, the model is required to translate molecules with QED scores from the lower range $[0.7, 0.8]$ into the higher range $[0.9, 1.0]$. The similarity constraint is $\text{sim}(X, Y) \geq 0.4$.

- **DRD2**: This task involves the optimization of a compound’s biological activity against dopamine type 2 receptor (DRD2). The model needs to translate inactive compounds ($p < 0.05$) into active compounds ($p \geq 0.5$), where the bioactivity is assessed by a property prediction model from Olivecrona et al. (2017). The similarity constraint is $\text{sim}(X, Y) \geq 0.4$.

Evaluation Metrics Our evaluation metrics include translation accuracy and diversity. Each test molecule $X_i$ is translated $K = 20$ times with different latent codes sampled from the prior distribution. On the logP optimization, we select compound $Y_i$ as the final translation of $X_i$ that gives the highest property improvement and satisfies $\text{sim}(X_i, Y_i) \geq \delta$. We then report the average property improvement $\frac{1}{K} \sum_{i} \logP(Y_i) - \logP(X_i)$ over test set $D$. For other tasks, we report the translation success rate. A compound is successfully translated if one of its $K$ translation candidates satisfies all the similarity and property constraints of the task. To measure the diversity, for each molecule we compute the average pairwise Tanimoto distance between all its successfully translated compounds.

Baselines We compare our method against the baselines including GCPN (You et al., 2018a), MMPA (Dalke et al., 2018) and translation based methods Seq2Seq and JTNN (Jin et al., 2019). Seq2Seq is a sequence-to-sequence model that generates molecules by their SMILES strings. JTNN is a graph-to-graph architecture that generates molecules structure by structure, but its decoder is not fully autoregressive.

To make a direct comparison possible between our method and atom-based generation, we further developed an atom-based translation model (AtomG2G) as baseline. It makes three predictions in each generation step. First, it predicts whether the decoding process has completed (no more new atoms). If not, it creates a new atom $a_t$ and predicts its atom type. Lastly, it predicts the bond type between $a_t$ and other atoms autoregressively to fully capture edge dependencies (You et al., 2018b). The encoder of AtomG2G encodes only the atom-layer graph and the decoder attention only sees the atom vectors $e^a_t$. All translation models are trained under the same variational objective. Details of baseline architectures are in the appendix.

4.2.1. Results

As shown in Table 2, our model (HierG2G) achieves the new state-of-the-art on the four translation tasks. In particular, our model significantly outperforms JTNN in both translation accuracy (e.g., 76.9% versus 59.9% on the QED task) and output diversity (e.g., 0.564 versus 0.480 on the logP task). While both methods generate molecules by structures, our decoder is autoregressive which can learn more expressive mappings. In addition, our model runs 6.3 times faster than JTNN during decoding. Our model also outperforms AtomG2G on three datasets, with over 10% improvement on the DRD2 task. This shows the advantage of our hierarchical model.

Ablation Study To understand the importance of different architecture choices, we report ablation studies over the QED and DRD2 tasks in Table 3. We first replace our hierarchical decoder with the atom-based decoder of AtomG2G to see how much the motif-based decoding benefits us. We keep the same hierarchical encoder but modified the input of the decoder attention to include both atom and motif vectors. Using this setup, the model performance decreases by 0.8% and 10.9% on the two tasks. We suspect the DRD2 task benefits more from motif-based decoding because biological target binding often depends on the presence of specific functional groups.

Our second experiment reduces the number of hierarchies in our encoder and decoder MPN, while keeping the same hierarchical decoding process. When the top motif layer is removed, the translation accuracy drops slightly by 0.8%
5. Related Work

Graph Generation Previous work have adopted various approaches for generating molecular graphs. Gómez-Bombarelli et al. (2018); Segler et al. (2017); Kusner et al. (2017); Dai et al. (2018); Guimaraes et al. (2017); Oliveira et al. (2018); Gao & Ji (2019) proposed to learn the graph hierarchy jointly with the encoding process. Despite some differences, all of these methods learns the hierarchy for regression or classification tasks. In contrast, our hierarchy is constructed for efficient graph generation.

Our work is most closely related to (Defferrard et al., 2016; Ying et al., 2018; Gao & Ji, 2019) that learn to represent graphs in a hierarchical manner. In particular, Defferrard et al. (2016) utilized graph coarsening algorithms to construct multiple layers of graph hierarchy and Ying et al. (2018); Gao & Ji (2019) proposed to learn the graph hierarchy jointly with the encoding process. Despite some differences, all of these methods learns the hierarchy for regression or classification tasks. In contrast, our hierarchy is constructed for efficient graph generation.

6. Conclusion

In this paper, we developed a hierarchical encoder-decoder architecture generating molecular graphs using structural motifs as building blocks. The experimental results show our model outperforms prior atom and substructure based methods in both small molecule and polymer domains.
Acknowledgements

We would like to thank Jiaming Luo, Adam Yala, Benson Chen, Adam Fisch, Octavian Ganea, Yujia Bao and Rachel Wu for their insightful comments. We also thank the anonymous reviewers for their helpful feedback. This work was supported by MLPDS, DARPA AMD project and J-Clinic.

References


Hierarchical Generation of Molecular Graphs using Structural Motifs


