Multi-Objective Molecule Generation using Interpretable Substructures

Wengong Jin 1  Regina Barzilay 1  Tommi Jaakkola 1

Abstract
Drug discovery aims to find novel compounds with specified chemical property profiles. In terms of generative modeling, the goal is to learn to sample molecules in the intersection of multiple property constraints. This task becomes increasingly challenging when there are many property constraints. We propose to offset this complexity by composing molecules from a vocabulary of substructures that we call molecular rationales. These rationales are identified from molecules as substructures that are likely responsible for each property of interest. We then learn to expand rationales into a full molecule using graph generative models. Our final generative model composes molecules as mixtures of multiple rationale completions, and this mixture is fine-tuned to preserve the properties of interest. We evaluate our model on various drug design tasks and demonstrate significant improvements over state-of-the-art baselines in terms of accuracy, diversity, and novelty of generated compounds.

1. Introduction
The key challenge in drug discovery is to find molecules that satisfy multiple constraints, from potency, safety, to desired metabolic profiles. Optimizing these constraints simultaneously is challenging for existing computational models. The primary difficulty lies in the lack of training instances of molecules that conform to all the constraints. For example, for this reason, Jin et al. (2019a) reports over 60% performance loss when moving beyond the single-constraint setting.

In this paper, we propose a novel approach to multi-property molecular optimization. Our strategy is inspired by fragment-based drug discovery (Murray & Rees, 2009) often followed by medicinal chemists. The idea is to start with substructures (e.g., functional groups or later pieces) that drive specific properties of interest, and then combine these building blocks into a target molecule. To automate this process, our model has to learn two complementary tasks: (1) identification of the building blocks that we call rationales, and (2) assembling multiple rationales together into a fully formed target molecule. In contrast to competing methods, our generative model does not build molecules from scratch, but instead assembles them from automatically extracted rationales already implicated for specific properties (see Figure 1).

We implement this idea using a generative model of molecules where the rationale choices play the role of latent variables. Specifically, a molecular graph $G$ is generated from underlying rationale sets $S$ according to:

$$P(G) = \sum_{S} P(G|S)P(S) \quad (1)$$

As ground truth rationales (e.g., functional groups or subgraphs) are not provided, the model has to extract candidate rationales from molecules with the help of a property predictor. We formulate this task as a discrete optimization problem efficiently solved by Monte Carlo tree search. Our rationale conditioned graph generator, $P(G|S)$, is initially trained on a large collection of real molecules so that it is capable of expanding any subgraph into a full molecule. The mixture model is then fine-tuned using reinforcement learning to ensure that the generated molecules preserve all the properties of interest. This training paradigm enables us to realize molecules that satisfy multiple constraints without observing any such instances in the training set.

The proposed model is evaluated on molecule design tasks under different combinations of property constraints.1 Our baselines include state-of-the-art molecule generation methods (Olivecrona et al., 2017; You et al., 2018a). Across all tasks, our model achieve state-of-the-art results in terms of accuracy, novelty and diversity of generated compounds. In particular, we outperform the best baseline with 38% absolute improvement in the task with three property constraints. We further provide ablation studies to validate the benefit of our architecture in the low-resource scenario. Finally, we show that identified rationales are chemically meaningful in a toxicity prediction task (Sushko et al., 2012).

1 https://github.com/wengong-jin/multiobj-rationale

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

Reinforcement Learning One of the prevailing paradigms for drug design is reinforcement learning (RL) (You et al., 2018a; Olivecrona et al., 2017; Popova et al., 2018), which seeks to maximize the expected reward defined as the sum of predicted property scores using the property predictors. Their approach learns a distribution \( P(\mathcal{G}) \) (a neural network) for generating molecules. Ideally, the model should achieve high success rate in generating molecules that meet all the constraints, while maintaining the diversity of \( P(\mathcal{G}) \).

The main challenge of RL is the reward sparsity, especially when there are multiple competing constraints. For illustration, we test a state-of-the-art reinforcement learning method (Olivecrona et al., 2017) under four property constraints: biological activity to GSK3, JNK3, drug-likeness, and synthetic accessibility (Li et al., 2018). As shown in Figure 1, initially the success rate and diversity is high when given only one of the constraints, but they decrease dramatically when all the property constraints are added. The reason of this failure is that the property predictor (i.e., reward function) remains black-box and the model has limited understanding of why certain molecules are desirable.

Our framework offsets this complexity by understanding property landscape through rationales. At a high level, the rationales are analogous to options (Sutton et al., 1999; Stolle & Precup, 2002), which are macro-actions leading the agent faster to its goal. The rationales are automatically discovered from molecules with labeled properties.

Molecule Generation Previous work have adopted various approaches for generating molecules under specific property constraints. Roughly speaking, existing methods can be divided along two axes — representation and optimization. On the representation side, they either operate on SMILES strings (Gómez-Bombarelli et al., 2018; Segler et al., 2017; Kang & Cho, 2018) or directly on molecular graphs (Simonovsky & Komodakis, 2018; Jin et al., 2018; Samanta et al., 2018; Liu et al., 2018; De Cao & Kipf, 2018; Ma et al., 2018; Seff et al., 2019). On the optimization side, the task has been formulated as reinforcement learning (Guimaraes et al., 2017; Olivecrona et al., 2017; Popova et al., 2018; You et al., 2018a; Zhou et al., 2018), continuing optimization in the latent space learned by variational autoencoders (Gómez-Bombarelli et al., 2018; Kusner et al., 2017; Dai et al., 2018; Jin et al., 2018; Kajino, 2018; Liu et al., 2018), or graph-to-graph translation (Jin et al., 2019b).

In contrast to existing approaches, our model focuses on the multi-objective setting of the problem and offers a different formulation for molecule generation based on rationales.

Interpretability Our rationale based generative model seeks to provide transparency (Doshi-Velez & Kim, 2017) for molecular design. The choice of rationales \( P(\mathcal{S}) \) is visible to users and can be easily controlled by human experts. Prior work on interpretability primarily focuses on finding rationales (i.e., explanations) of model predictions in image and text classification (Lei et al., 2016; Ribeiro et al., 2016; Sundararajan et al., 2017) and molecule property prediction (McCloskey et al., 2019; Ying et al., 2019; Lee et al., 2019). In contrast, our model uses rationales as building blocks for molecule generation.
molecule $G_i$ is called positive to property $i$ if $r_i(G_i) \geq \delta_i$ and negative otherwise.

Following previous work (Olivecrona et al., 2017; Popova et al., 2018), $r_i(G_i)$ is output of property prediction models (e.g., random forests) which effectively approximate empirical measurements. The prediction model is trained over a set of molecules with labeled properties gathered from real experimental data. The property predictor is then fixed throughout the rest of the training process.

**Overview** Our model generates molecules by first sampling a rationale $S$ from the vocabulary $V^M_S$, and then completing it into a molecule $G$. The generative model is defined as

$$P(G) = \sum_{S \in V^M_S} P(S)P(G|S)$$

(3)

As shown in Figure 2, our model consists of three modules:

- **Rationale Extraction**: Construct rationale vocabulary $V^M_S$, each individual property $i$ and combines these rationales for multiple properties $V^M_S$ (see §3.1).

- **Graph Completion**: $P(G|S)$: Generate molecules $G$ using multi-property rationales $S^M_i \in V^M_S$. The model is first pre-trained on natural compounds and then fine-tuned to generate molecules satisfying multiple constraints (see §3.2 for its architecture and §3.3 for fine-tuning).

- **Rationale Distribution**: $P(S)$: The rationale distribution $P(S)$ is learned based on the properties of complete molecules $G$ generated from $P(G|S)$. A rationale $S$ is sampled more frequently if it is more likely to be expanded into a positive molecule $G$ (see §3.3).

3.1. Rationale Extraction from Predictive Models

**Single-property Rationale** We define a rationale $S^i$ for a single property $i$ as a subgraph of some molecule $G$ which causes $G$ to be active (see Figure 1). To be specific, let $V^i_S$ be the vocabulary of such rationales for property $i$. Each rationale $S^i \in V^i_S$ should satisfy the following two criteria to be considered as a rationale:

1. The size of $S^i$ should be small (less than 20 atoms).
2. Its predicted property score $r_i(S^i) \geq \delta_i$.

For a single property $i$, we propose to extract its rationales from a set of positive molecules $D^p_i$ used to train the property predictor. For each molecule $G_i^p \in D^p_i$, we find a rationale subgraph with high predicted property and small size ($N_a = 20$):

Find subgraph $S^i \subset G_i^p$

Subject to $r_i(S^i) \geq \delta_i$, $|S^i| \leq N_a$ and $S^i$ is connected

Solving the above problem is challenging because rationale $S^i$ is discrete and the potential number of subgraphs grows exponentially to the size of $G_i^p$. To limit the search space, we have added an additional constraint that $S^i$ has to be a connected subgraph. In this case, we can find a rationale $S$ by iteratively removing some peripheral bonds while maintaining its property. Therefore, the key is learning to prune the molecule.

This search problem can be efficiently solved by Monte Carlo Tree Search (MCTS) (Silver et al., 2017). The root of the search tree is $G^p$ and each state $s$ in the search tree is a subgraph derived from a sequence of bond deletions. To ensure that each subgraph is chemically valid and stays connected, we only allow deletion of one peripheral non-aromatic bond or one peripheral ring from each state. As shown in Figure 3, a bond or a ring $a$ is called peripheral if $G^p$ stays connected after deleting $a$.

During search process, each state $s$ in the search tree contains edges $(s, a)$ for all legal deletions $a$. Following Silver et al. (2017), each edge $(s, a)$ stores the following statistics:

- $N(s, a)$ is the visit count of deletion $a$, which is used for exploration-exploitation tradeoff in the search process.

This assumption is valid in many cases. For instance, rationales for toxicity (i.e., toxicophores) are connected subgraphs in most cases (Sushko et al., 2012).

---

Figure 2. Overview of our approach. We first construct rationales for each individual property and then combine them as multi-property rationales. The method learns a graph completion model $P(G|S)$ and rationale distribution $P(S)$ in order to generate positive molecules.
In the end, we collect all the leaf states with $r_i(s) \geq \delta_i$ and add them to the rationale vocabulary $V_S^M$.

**Multi-property Rationale** For a set of $M$ properties, we can similarly define its rationale $S^{[M]}$ by imposing $M$ property constraints at the same time, namely

$$\forall i : r_i(S^{[M]}) \geq \delta_i, i = 1, \cdots, M$$

In principle, we can apply MCTS to extract rationales from molecules that satisfy all the property constraints. However, in many cases there are no such molecules available. To this end, we propose to construct multi-property rationales from single-property rationales extracted by MCTS. Specifically, each multi-property rationale $S^{[M]}$ is merged from single-property rationales $S^1, \cdots, S^M$. We merge two rationales $S^j$ and $S^l$ by first finding their maximum common substructure (MCS) and then superposing $S^j$ on $S^l$ so that their MCS coincides (see Figure 4). This gives us a set of candidate rationales:

$$C^M_S = \bigcup_{(S^1, \cdots, S^M)} \text{MERGE}(S^1, \cdots, S^M)$$

where $(S^1, \cdots, S^M) \in V^1_S \times \cdots \times V^M_S$. Note that the output of MERGE is a set as there are multiple ways of superposing two rationales. Finally, the vocabulary of multi-property rationales is the subset of $C^M_S$ which satisfies all the property constraints:

$$V^M_S = \{ S \in C^M_S | r_i(S^{[M]}) \geq \delta_i, \forall i \}$$

For notational convenience, we will denote both single and multi-property rationales as $S$ from now on.

### 3.2. Graph Completion

This module is a variational autoencoder which completes a full molecule $G$ given a rationale $S$. Since each rationale $S$...
can be realized into many different molecules, we introduce a latent variable \( z \) to generate diverse outputs:

\[
P(G|S) = \int_z P(G|S, z) P(z) dz
\]

where \( P(z) \) is the prior distribution. Different from standard graph generation, our graph decoder must generate graphs that contain subgraph \( S \). Our VAE architecture is adapted from existing atom-by-atom generative models (You et al., 2018b; Liu et al., 2018) to incorporate the subgraph constraint. For completeness, we present our architecture here:

**Encoder** Our encoder is a message passing network (MPN) which learns the approximate posterior \( Q(z|G, S) \) for variational inference. Let \( e(a_u) \) be the embedding of atom \( u \) with atom type \( a_u \), and \( e(b_{uv}) \) be the embedding of bond \((u, v)\) with bond type \( b_{uv} \). The MPN computes atom representations \( \{h_u, v \in G\} \):

\[
\{h_v\} = \text{MPN}_e(G, \{e(a_u)\}, \{e(b_{uv})\})
\]  

For simplicity, we denote the MPN encoding process as \( \text{MPN}(\cdot) \), which is detailed in the appendix. The atom vectors are aggregated to represent \( G \) as a single vector \( h_G = \sum_v h_v \). Finally, we sample latent vector \( z_G \) from \( Q(z|G, S) \) with mean \( \mu(h_G) \) and log variance \( \log \sigma(h_G) \):

\[
z_G = \mu(h_G) + \exp(\Sigma(h_G)) \cdot \epsilon; \quad \epsilon \sim N(0, I)
\]

**Decoder** The decoder generates molecule \( G \) according to its breadth-first order. In each step, the model generates a new atom and its connecting edges. During generation, we maintain a queue \( Q \) that contains frontier nodes in the graph who still have neighbors to be generated. Let \( G_t \) be the partial graph generated till step \( t \). To ensure \( G \) contains \( S \) as subgraph, we set the initial state of \( G_0 = S \) and put all the peripheral atoms of \( S \) to the queue \( Q \) (only peripheral atoms are needed due to the rationale extraction algorithm).

In the \( t \)-th generation step, the decoder first runs a MPN over current graph \( G_t \) to compute atom representations \( h_v^{(t)} \):

\[
\{h_v^{(t)}\} = \text{MPN}_d(G_t, \{e(a_u)\}, \{e(b_{uv})\})
\]

The current graph \( G_t \) is represented as the sum of its atom vectors \( h_G = \sum_{v \in G_t} h_v^{(t)} \). Suppose the first atom in \( Q \) is \( v_t \). The decoder decides to expand \( G_t \) in three steps:

1. Predict whether there will be a new atom attached to \( v_t \):

\[
p_t = \text{sigmoid}(\text{MLP}(h_{v_t}, h_{G_t}, z_G))
\]

where \( \text{MLP}(\cdot, \cdot, \cdot) \) is a ReLU network whose input is a concatenation of multiple vectors.

2. If \( p_t < 0.5 \), discard \( v_t \) and move on to the next node in \( Q \). Stop generation if \( Q \) is empty. Otherwise, create a new atom \( u_t \) and predict its atom type:

\[
p_{u_t} = \text{softmax}(\text{MLP}(h_{u_t}, h_{G_t}, z_G))
\]

3. Predict the bond type between \( u_t \) and other frontier nodes in \( Q = \{q_1, \ldots, q_n\} \) (\( q_1 = v_t \)). Since atoms are generated in breadth-first order, there are no bonds between \( u_t \) and atoms not in \( Q \).

To fully capture edge dependencies, we predict the bonds between \( u_t \) and atoms in \( Q \) sequentially and update the representation of \( u_t \) when new bonds are added to \( G_t \). In the \( k \)-th step, we predict the bond type of \((u_t, q_k)\) as follows:

\[
b_{u_t, q_k} = \text{softmax}\left(\text{MLP}(g_{u_t, q_k}, h_G, z_G)\right)
\]

where \( g_{u_t, q_k} \) is the new representation of \( u_t \) after bonds \( \{(u_t, q_1), \ldots, (u_t, q_{k-1})\} \) have been added to \( G_t \):

\[
g_{u_t}^{(k-1)} = \text{MLP}\left(e(a_u), \sum_{j=1}^{k-1} \text{MLP}(h_{q_j}, e(b_{q_j, u_t}))\right)
\]

### 3.3. Training Procedure

Our training objective is to maximize the expected reward of generated molecules \( G \), where the reward is an indicator of \( r_i(G) \geq \delta_i \) for all properties \( 1 \leq i \leq M \)

\[
\sum_G \mathbb{I}\left[\bigwedge_i r_i(G) \geq \delta_i\right] P(G) + \lambda H[P(S)]
\]

where \( \mathbb{I} \) is the indicator function, \( H[P(S)] \) is the entropy of the distribution over the rationale vocabulary. Let \( H[S] = \mathbb{I}\left[\bigwedge_i r_i(S) \geq \delta_i\right] \).

It is easy to show that the optimal \( P(S) \) has a closed form solution:

\[
P(S_k) \propto \exp\left(\frac{1}{\lambda} \sum_G \mathbb{I}[G] P(G|S_k)\right)
\]

The remaining question is how to train graph generator \( P(G|S) \). The generator seeks to produce molecules that are realistic and positive. However, Eq.(16) itself does not take into account whether generated molecules are realistic or not. To encourage the model to generate realistic compounds, we train the graph generator in two phases:

- **Pre-training** \( P(G|S) \) using real molecules.
- **Fine-tuning** \( P(G|S) \) using policy gradient with reward from property predictors.

The overall training algorithm is shown in Algorithm 1.

### 3.3.1. Pre-training

In addition to satisfying all the property constraints, the output of the model should constitute a realistic molecule. For this purpose, we pre-train the graph generator on a large set of molecules from ChEMBL (Gaulton et al., 2017). Each training example is a pair \((S, G)\), where \( S \) is a random connected subgraph of a molecule \( G \) with up to \( N \)
We evaluate our method (RationaleRL) on molecule design. We set the positive threshold $\delta$ (see §3.1)
4. Experiments

For each property, we split its property dataset into 80%, 10% and 10% for training, validation and testing. The test AUROC score is 0.86 for both GSK3β and JNK3.

Mult-property Constraints We also consider combinations of property constraints that are biologically relevant:
- GNK3β + JNK3: Jointly inhibiting JNK3 and GSK-3β may provide potential benefit for the treatment of Alzheimer’s disease (Li et al., 2018). There exist 316 dual inhibitors already available in the dataset.
- GNK3β + JNK3 + QED + SA: We further require the generated dual inhibitors to be drug like and synthetically accessible (i.e., easy to be synthesized). These two properties are quantified by QED (Bickerton et al., 2012) and synthetic accessibility (SA) score (Ertl & Schuffenhauer, 2009). In particular, we require QED $\geq 0.6$ and SA $\leq 4.0$.

Evaluation Metrics Our evaluation effort measures various aspects of molecule design. For each method, we generate $n = 5000$ molecules and compute the following metrics:
- Success: The fraction of sampled molecules predicted to be positive (i.e., satisfying all property constraints). A good model should have a high success rate.
- Diversity: It is also important for a model generate diverse range of positive molecules. To this end, we measure the diversity of generated positive compounds by computing their pairwise molecular distance $\text{sim}(X,Y)$, which is defined as the Tanimoto distance over Morgan fingerprints of two molecules.

\[
\text{Diversity} = 1 - \frac{2}{n(n-1)} \sum_{X,Y} \text{sim}(X,Y)
\]
- Novelty: Crucially, a good model should discover novel positive compounds. In this regard, for each generated positive compound $G$, we find its nearest neighbor $G_{\text{SNN}}$ from positive molecules in the training set. We define the novelty as the fraction of molecules with nearest neighbor similarity lower than 0.4 (Olivecrona et al., 2017):

\[
\text{Novelty} = \frac{1}{n} \sum_{G} 1[\text{sim}(G, G_{\text{SNN}}) < 0.4]
\]

Baselines We compare our method against the following state-of-the-art generation methods for molecule design:
- JT-VAE (Jin et al., 2018) is a generative model which generate molecules based on substructures such as rings and bonds. The model contains auxiliary property predictors over the VAE latent space. During testing, we perform multi-objective optimization in the latent space using the gradient from the property predictors.
Table 1. Results on molecule design with one or two property constraints.

<table>
<thead>
<tr>
<th>Method</th>
<th>GSK3/β</th>
<th>JNK3</th>
<th>GSK3/β + JNK3</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Success</td>
<td>Novelty</td>
<td>Diversity</td>
</tr>
<tr>
<td>JT-VAE</td>
<td>32.2%</td>
<td>11.8%</td>
<td>0.901</td>
</tr>
<tr>
<td>GCPN</td>
<td>42.4%</td>
<td>11.6%</td>
<td><strong>0.904</strong></td>
</tr>
<tr>
<td>GVAE-RL</td>
<td>33.2%</td>
<td>76.4%</td>
<td>0.874</td>
</tr>
<tr>
<td>REINVENT</td>
<td>99.3%</td>
<td><strong>61.0%</strong></td>
<td>3.0%</td>
</tr>
<tr>
<td>RationaleRL</td>
<td><strong>100%</strong></td>
<td>53.4%</td>
<td>0.888</td>
</tr>
</tbody>
</table>

Table 2. Molecule design with four property constraints. The novelty and diversity of JT-VAE, GVAE-RL, and GCPN are not reported due to their low success rate.

<table>
<thead>
<tr>
<th>Method</th>
<th>GSK3/β + JNK3 + QED + SA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Success</td>
</tr>
<tr>
<td>JT-VAE</td>
<td>1.3%</td>
</tr>
<tr>
<td>GVAE-RL</td>
<td>2.1%</td>
</tr>
<tr>
<td>GCPN</td>
<td>4.0%</td>
</tr>
<tr>
<td>REINVENT</td>
<td>47.9%</td>
</tr>
<tr>
<td>RationaleRL</td>
<td><strong>74.8%</strong></td>
</tr>
</tbody>
</table>

- REINVENT (Olivecrona et al., 2017) is a RL model generating molecules based on their SMILES strings (Weininger, 1988). To generate realistic molecules, their model is pre-trained over one million molecules from ChEMBL and then fine-tuned under policy reward.
- GCPN (You et al., 2018a) is a RL model which generates molecular graphs atom by atom. It uses GAN (Goodfellow et al., 2014) to help generate realistic molecules.
- GVAE-RL is a graph variational autoencoder which generates molecules atom by atom. The graph VAE architecture is the same as our model, but it generates molecules from scratch without using rationales. The model is pre-trained on the same ChEMBL dataset and fine-tuned for each property using policy gradient. This is an ablation study to show the importance of using rationales.

Rationales Details of the rationales used in our model:
- GSK3/β, JNK3: For single properties, the rationale size is required to be less than 20 atoms. For each positive molecule, we run 20 iteration of MCTS with $cpuct = 10$. There are 6299 and 417 rationales for GSK3/β and JNK3.
- GSK3/β + JNK3: We construct two-property rationales by merging the single-property rationales (see Figure 4). For computational efficiency, we only consider rationales $S_k$ receiving the highest $P(S_k)$ within each property. This gives us two rationale subsets with $|V^{GSK3/β}_S| = 238$ and $|V^{JNK3}_S| = 39$. In total, there are 9719 two-property rationales merged from $V^{GSK3/β}_S$ and $V^{JNK3}_S$.
- GSK3/β + JNK3 + QED + SA: We construct four-property rationales by selecting the two-property rationales whose QED $\geq 0.6$ and SA $\leq 4.0$. There are in total 61 different four-property rationales.

Model Setup We pre-train all the models on the same ChEMBL dataset, which contains 1.02 million training examples. On the four-property generation task, our model is fine-tuned for $L = 50$ iterations, with each rationale being expanded for $K = 200$ times.
4.1. Results

The results are reported in Table 1 and 2. On the single-property generation task, both REINVENT and RationaleRL demonstrate nearly perfect success rate since there is only one constraint. On the two-property generation task, our model achieves 100% success rate while maintaining 97.3% novelty and 0.824 diversity score, which is much higher than REINVENT. Meanwhile, GCPN has a low success rate (3.5%) due to reward sparsity.

Table 2 shows the results on the four-property generation task, which is the most challenging. The difference between our model the baselines become significantly larger. In fact, GVAE-RL and GCPN completely fail in this task due to reward sparsity. Our model outperforms REINVENT with a wide margin (success rate: 74.8% versus 47.9%; diversity: 0.701 versus 0.621).

Ablation study In all tasks, our method significantly outperforms GVAE-RL, which has the same generative architecture but does not utilize rationales and generates molecules from scratch. Thus we conclude the importance of rationales for multi-property molecular design.

Visualization We further provide visualizations to help understand our model. In Figure 5, we plotted a t-SNE (Maaten & Hinton, 2008) plot of the extracted rationales for GSK3/β and JNK3. For both properties, rationales mostly cover the chemical space populated by existing positive molecules. The generated GSK3/β+JNK3 dual inhibitors are distributionally close to the true dual inhibitors in the training set. In Figure 6, we show samples of rationales extracted by MCTS. In Figure 7, we show examples of generated molecules that satisfy all the four constraints (GSK3/β+JNK3+QED+SA).

4.2. Property Predictor Applicability

Since the reported success rate is based on property predictors, it is possible for the generated molecules to exploit flaws of the property predictors. In particular, the predicted properties may be unreliable when the distribution of generated compounds is very different from the distribution of molecules used to train the property predictors.

This issue is alleviated by our framework because our generated compounds are built from rationales extracted from true positive compounds used to train the property predictors. Therefore, our generated compounds are closer to the true compounds than compounds generated from scratch. To show this, we adopt Frechet ChemNet Distance (FCD)4 to measure distributional discrepancy between generated molecules and positive compounds in the training set (Preuer

---

4The FCD score is analogous to the Frechet Inception Distance (Heusel et al., 2017) used in image synthesis.
Table 3. Frechet ChemNet Distance (FCD) between generated compounds and true positive molecules in the training set.

<table>
<thead>
<tr>
<th></th>
<th>GSK3β</th>
<th>JNK3</th>
<th>GSK3β+JNK3</th>
</tr>
</thead>
<tbody>
<tr>
<td>REINVENT</td>
<td>28.63</td>
<td>25.16</td>
<td>36.75</td>
</tr>
<tr>
<td>RationaleRL</td>
<td>6.88</td>
<td>9.68</td>
<td>25.52</td>
</tr>
</tbody>
</table>

Table 4. Rationale accuracy on the toxicity dataset. Our rationales are more faithful to the property of interest.

<table>
<thead>
<tr>
<th>Method</th>
<th>Partial Match</th>
<th>Exact Match</th>
</tr>
</thead>
<tbody>
<tr>
<td>Integrated Gradient</td>
<td>0.857</td>
<td>39.4%</td>
</tr>
<tr>
<td>MCTS Rationale</td>
<td>0.861</td>
<td>46.0%</td>
</tr>
</tbody>
</table>

et al., 2018). As shown in Table 3, the FCD of RationaleRL is much lower than REINVENT, which means our generated compounds are distributionally closer to the true compounds. As a result, the predicted properties are more reliable for molecules generated by our model.

4.3. Faithfulness of Rationales

While our rationales are mainly extracted for generation, it is also important for them to be chemically relevant. In other words, the extracted rationales should accurately explain the property of interest. As there is no ground truth rationales available for JNK3 and GSK3β, we turn to an auxiliary toxicity dataset for evaluating rationale quality.

Data The toxicity dataset contains 125K molecules randomly selected from ChEMBL. Each molecule is labeled as toxic if it contains structural alerts (Sushko et al., 2012) — chemical substructures that is correlated with human or environmental hazards (see Figure 7).5 Under this setup, the structural alerts are ground truth rationales and we evaluate how often the extracted rationales match them. The dataset is split into 105K for training and 20K for testing. In total, there are 26.5K toxic molecules and 164 types of structural alerts. We train a graph convolutional network (Yang et al., 2019) to predict toxicity, which achieves 0.99 AUROC score on the test set.

Results We compare our MCTS based rationale extraction with integrated gradient (Sundararajan et al., 2017), which has been applied to explain property prediction models (McCloskey et al., 2019). We report two metrics: partial match AUC (attribution AUC metric used in McCloskey et al. (2019)) and exact match accuracy which measures how often a rationale graph exactly matches the true rationale in the molecule. As shown in Table 4, our method significantly outperforms the baseline in terms of exact matching. The extracted rationales has decent overlap with true rationales, with 0.86 partial match on average. Therefore, our model is capable of finding rationales faithful to the properties.

5. Conclusion

In this paper, we developed a rationale based generative model for molecular design. Our model generates molecules in two phases: 1) identifying rationales whose presence indicate strong positive signals for each property; 2) expanding rationale graphs into molecules using graph generative models and fine-tuning it towards desired combination of properties. Our model demonstrates strong improvement over prior reinforcement learning methods in various tasks.

Acknowledgements

We would like to thank Patrick Walters, Jiaming Luo, Adam Yala, Benson Chen, Adam Fisch, 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


