GPMO: Gradient Perturbation-Based Contrastive Learning for Molecule Optimization

Xixi Yang[1] , **Li Fu**[2] , **Yafeng Deng**[2] , **Yuansheng Liu**[1*] , **Dongsheng Cao**[3*] and **Xiangxiang Zeng**[1*]

[1]College of Computer Science and Electronic Engineering, Hunan University, Changsha, China
[2]CarbonSilicon AI Technology Co., Ltd, Hangzhou, Zhejiang, China
[3]Xiangya School of Pharmaceutical Sciences, Central South University, Changsha, Hunan, China
{yangxixi, yuanshengliu, xzeng}@hnu.edu.cn, {fuli, dengyafeng}@carbonsilicon.ai, oriental-cds@163.com

Abstract

Optimizing molecules with desired properties is a crucial step in de novo drug design. While translation-based methods have achieved initial success, they continue to face the challenge of the "exposure bias" problem. The challenge of preventing the "exposure bias" problem of molecule optimization lies in the need for both positive and negative molecules of contrastive learning. That is because generating positive molecules through data augmentation requires domain-specific knowledge, and randomly sampled negative molecules are easily distinguished from the real molecules. Hence, in this work, we propose a molecule optimization method called GPMO, which leverages a gradient perturbation-based contrastive learning method to prevent the "exposure bias" problem in translation-based molecule optimization. With the assistance of positive and negative molecules, GPMO is able to effectively handle both real and artificial molecules. GPMO is a molecule optimization method that is conditioned on matched molecule pairs for drug discovery. Our empirical studies show that GPMO outperforms the state-of-the-art molecule optimization methods. Furthermore, the negative and positive perturbations improve the robustness of GPMO.

1 Introduction

Molecule Optimization (MO) aims to identify a target molecule, that exhibits improved pharmacological properties while retaining its similarity to the original molecule. Traditionally, chemists rely on their knowledge, experience, and intuition to apply chemical transformations to promising molecules in order to achieve a balance of desirable properties. Additionally, this manual approach can be time-consuming, labor-intensive, and may not explore the full range of possible modifications that could enhance a molecule's pharmacological properties. High throughput screening tests a large library of compounds to identify those with desirable pharmacological properties, but it can be an expensive and inefficient approach to drug discovery.

In contrast to the traditional method, in silico MO through automated processes to improve drug properties, which can accelerate the drug design process [Zeng *et al.*, 2022; Ma *et al.*, 2022; Lin *et al.*, 2020]. Deep learning technologies, such as reinforcement learning (RL) [Jin *et al.*, 2018a; You *et al.*, 2018; Zhou *et al.*, 2019] and deep generative model (DGM) [Zang and Wang, 2020; Chen *et al.*, 2021; Jin *et al.*, 2020; Liu *et al.*, 2020], have been applied in MO. RL methods view MO as the Markov decision process, in which the current state of the molecule is modified through actions guided by a reward function that reflects desired property. While RL methods have shown promise in improving property, they can significantly alter structure of molecule [Gao *et al.*, 2021]. DGMs optimize molecules by generating new structures through translation between molecule pairs while maintaining the scaffold of the original molecule through similarity constraints. The translation-based DGMs are trained using the teacher-forcing training strategy to predict gold-standard labels of the previous tokens, without being exposed to incorrectly generated tokens. During the testing phase, translation-based DGMs rely on the model's prediction of the previous ones, which can differ from the ground-truth contexts. This discrepancy between training and testing phases is known as the "exposure bias" problem [Lee *et al.*, 2020].

To address the aforementioned issues, we propose GPMO that utilizes contrastive learning [Shen *et al.*, 2022] to overcome the "exposure bias" problem in the teacher-forcing guided model. To generate negative and positive molecules for contrastive learning, we incorporate gradient perturbation. After pre-training on a large-scale unlabeled dataset, GPMO decodes the output of the encoder to generate a target molecule with multiple property constraints. The input of GPMO is the condition tokens and the SMILES of start molecules. During the encoder-decoder process, GPMO generates the negative and positive molecules without domain-specific knowledge, and these molecules are used to train the model with adversarial loss. GPMO generates negative molecules by adding a small gradient perturbation to the hidden state of the target molecule, and the model is trained to minimize the conditional likelihood of the negative molecule

*Corresponding authors.

to ensure it is classified as a negative example during contrastive learning. In GPMO, the positive molecule is generated using a two-step gradient perturbation on the hidden state of the target molecule. The Kullback-Leibler (KL) divergence ensures that the distributions of the positive and real molecules are similar, guaranteeing that the positive molecule is indeed a positive example. The two-step gradient perturbation keeps the positive molecule far away from the real molecule in the embedding space. This will produce challenging molecules that the model is unable to accurately differentiate, providing it with more useful pairs to learn from. It is worth noting that GPMO is different from the one proposed in [Lee *et al.*, 2020]. On one hand, GPMO focuses on optimizing molecules based on conditional constraints, rather than simply generating text without specific criteria. In this way, GPMO can generate molecules with specific property values. On the other hand, GPMO provides a comprehensive framework for molecule optimization applications, making it highly suitable for large-scale optimization tasks. Experimental results show that GPMO outperforms all baseline methods in identifying molecules that satisfy specific property constraints under various conditions. Our contributions can be summarized as follows:

- GPMO is the first model to address the “exposure bias” problem in MO, and it does so by generating positive and negative molecules for effective contrastive learning.
- GPMO applies positive and negative molecules generation strategy which is free from domain-specific knowledge and easily accessible in the back-propagation process.
- Experimental results show that GPMO achieves SOTA in properties optimization problems, and can optimize one molecule into another molecule with better properties while maintaining the scaffold and exhibiting more desired properties.

2 Related Work

2.1 Reinforcement Learning Method

The objective of RL is to teach intelligent agents to take action in an environment to maximize the cumulative reward. The environment of the RL method is defined by chemical rules, and the improvement of a molecule’s property is defined as the reward in RL. The RL method demonstrates effective property improvement and has the capability to operate in a significantly larger chemical space compared to deep learning methods [Wang *et al.*, 2021]. However, the RL method often overcomplicates the molecular structure when optimizing multiple properties [Nigam *et al.*, 2019]. A newly developed RL method [Gao *et al.*, 2021] introduces a synthesis tree to alleviate the synthesis problem in MO that arises from complex structures. The synthesis of an optimized molecule is ensured by following a path in the synthesis tree.

2.2 Deep Generative Model

DGM are widely employed for molecule optimization, utilizing various approaches such as VAE-based [Gómez-Bombarelli *et al.*, 2018], GAN-based [Guimaraes *et al.*, 2017], encoder-decoder based [He *et al.*, 2021], and flow-based models [Zang and Wang, 2020; Shi *et al.*, 2020]. DGMs for molecule optimization treat the task as a generative problem, where the starting molecules serve as input, and the target molecules are the output. Typically, the target molecules are aimed to be more drug-like compared to the starting molecules. Modof [Chen *et al.*, 2021] is a DGM designed for molecule optimization. It predicts the disconnection site and modification between the starting and target molecules.

2.3 Hybrid Method

Hybrid methods include genetic algorithms (GA) method [Nigam *et al.*, 2019], Markov Chain Monte Carlo (MCMC) method [Fu *et al.*, 2021], and Bayesian optimization (BO) method [Moss *et al.*, 2020]. The GA method [Nigam *et al.*, 2019] incorporates an adaptive penalty mechanism based on a neural network to promote the exploratory behavior of GA, thus improving the diversity of generated molecules. The MCMC method [Fu *et al.*, 2021] first pre-trains two graph neural networks for substructure-type and molecule topology predictions. Subsequently, the optimized molecules are sampled from the pre-trained molecule distribution using MCMC sampling with multiple constraints. The BO method [Moss *et al.*, 2020] utilizes BO loops on string kernels to encode molecules into feature-rich spaces and employs GA to obtain the targeted molecules.

In this work, we present a novel molecule optimization method that treats molecule optimization as a machine translation problem. This approach offers contrastive learning without the requirement of domain-specific knowledge for generating negative and positive molecules, which helps address the “exposure bias” problem encountered in translation-based methods for molecule optimization.

3 Method

This section outlines the specifics of GPMO, which comprises two main steps: pre-training of encoder-decoder and conditional molecule optimization with contrastive learning. GPMO learns the grammar of SMILES and the relationships between molecules in the pre-training stage by masked language model (MLM) and SMILES translation task. Condition tokens are used for conditional molecule optimization. During conditional molecule optimization, positive and negative molecules are generated by gradient perturbation for contrastive learning.

Figure 1a illustrates that conditional molecule optimization involves utilizing both the condition tokens and the SMILES of the start molecule as inputs for the encoder-decoder model. The outputs of the encoder-decoder model are the SMILES of the target molecule. By adopting this approach, the model is capable of generating a novel molecule

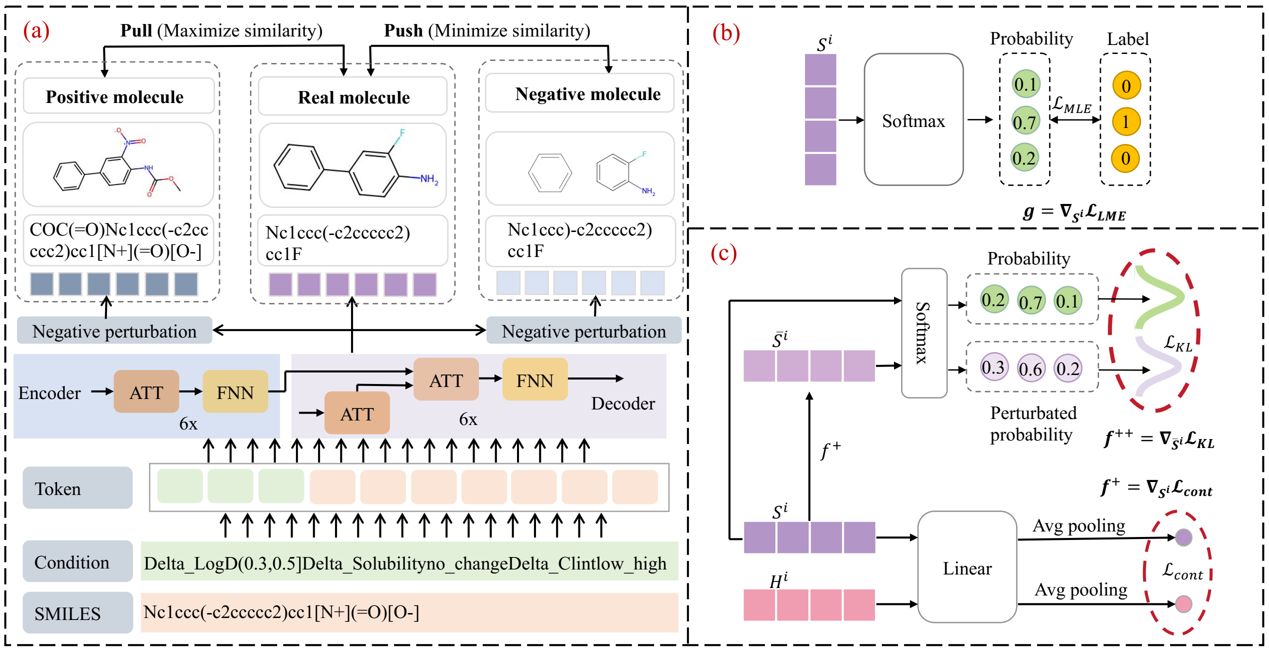

Figure 1: (a) The overall framework of multi-property MO with gradient-based contrastive learning. The transformer encoder-decoder is trained for MO with ADMET properties. Condition tokens are used to mark the property changes between start and target molecules. To facilitate contrastive learning, a gradient perturbation-based method is employed for generating both negative and positive molecules. (b) Negative perturbation: enforcing the model to distinguish "hard" negative molecules that are near real molecules. (c) Positive perturbation: enforcing the model to distinguish "hard" positive molecules that are far away from real molecules.

that adheres to the specified conditions while preserving the essential structural characteristics of the start molecule. The encoder-decoder model is trained using additional negative and positive molecules, which are produced through negative (Figure 1b) and positive perturbations (Figure 1c). Negative molecules share a similar SMILES string with real molecules but differ in their context, as modifying a single token can result in an invalid molecule (the negative molecule in Figure 1a). Compared to the real molecule, the positive molecules in Figure 1a vary with SMILES but share a similar chemical structure.

3.1 Problem Definition

MO can be formulated as a seq2seq problem, where the optimization from the start molecule to the target molecule is the translation between SMILES of molecules. This task is typically accomplished using an encoder-decoder architecture. For a given input molecule $\mathbf{x}^i$, the encoder converts it into a high-level representation. Given the latent representation of $\mathbf{x}^i$, the decoder autoregressively generates one token of prediction molecule at a time. The encoder-decoder process can be defined as $\mathbf{H}^i = f(\mathbf{x}^i)$ and $\mathbf{S}^i = g(\mathbf{H}^i)$, where $\mathbf{H}^i = [\mathbf{h}_1^i, \mathbf{h}_2^i, ..., \mathbf{h}_L^i]$ and $\mathbf{S}^i = [\mathbf{s}_1^i, \mathbf{s}_2^i, ..., \mathbf{s}_T^i]$ represent the hidden states of the input molecule with a length of L and the output molecule with a length of T, respectively. f and g denote the encoder and decoder respectively. The auto-regressive decoding process during training with teacher forcing, denoted as $p_\theta\left(y_t|\mathbf{y}_{<\mathbf{t}}^i, \mathbf{x}^i\right)$, can be gained by the decoder hidden state $\mathbf{s}_t^i$ with a softmax layer as $p_\theta(y_t^i|\mathbf{y}_{<\mathbf{t}}^i, \mathbf{x}^i) = \mathrm{softmax}(\mathbf{W}\mathbf{s}_\mathrm{t}^\mathrm{i}+\mathbf{b})$, where $\mathbf{W}$ and $\mathbf{b}$ are the weight and bias of the linear layer, respectively. Given a dataset $(\mathbf{x}^i, \mathbf{y}^i)_{i=1}^N$ with N molecules, the seq2seq model is trained using a maximized likelihood estimate (MLE), which can be formulated as follows:

$$\mathcal{L}_{MLE} = \sum_{i=1}^{N} \log p_\theta(\mathbf{y}^i|\mathbf{x}^i), \tag{1}$$

where $\log p_\theta\left(\mathbf{y}^i|\mathbf{x}^i\right)$ represents the log likelihood of $\mathbf{y}^i$ conditional on $\mathbf{x}^i$. $\mathcal{L}_{MLE}$ is maximized to estimate the parameter of the encoder-decoder.

3.2 Condition Token

GPMO is an approach used to optimize a given start molecule to a target molecule that exhibits structural similarity while satisfying conditional constraints. In order to ensure the similarity of molecular structures, the start and target molecules used in GPMO are constrained to comply with MMP relation [Dalke *et al.*, 2018]. Molecules with MMP relations share the same scaffold but differ in a single chemical transformation. Conditional molecule optimization is achieved by using condition tokens, which encode the changes of desired properties between the start and target molecules. By incorporating condition tokens, the optimization process can be guided to search for molecules that satisfy the desired properties encoded in the condition tokens. Three ADMET properties—LogD, Solubility, and Clint—are used as the properties to be optimized. LogD refers to the partition coefficient of all forms of a compound at a specific pH. Solubility is the degree to which a substance dissolves in water to make a solution. Clint is the overall hepatic intrinsic clearance.

Specifically, as the condition shown in Figure 1a, the condition tokens are included as part of the GPMO input. The property changes of LogD can be defined as: $\Delta(\mathrm{LogD}) = \mathrm{LogD}(\mathbf{y}^{\mathrm{i}}) - \mathrm{LogD}(\mathbf{x}^{\mathrm{i}})$. Then, $\Delta(\mathrm{LogD})$ is map into a code table: $(-inf, -6.9], (-6.9, -6.7], ..., (6.9, inf)$. -6.9 and 6.9 are the minimum and maximum values of $\Delta(\mathrm{LogD})$. $\Delta(\mathrm{LogD})$ is mapped into the corresponding code. For example, the condition token of $\Delta(\mathrm{LogD}) = 0.82$ can be denoted as Delta_LogD(0.7,0.9]. In terms of Solubility, the code table of $\Delta(\mathrm{Solubility})$ can be defined as:

$$\mathrm{code} = \begin{cases} \text{high\_low}; & \text{if } \Delta(\mathrm{Solubility}) \le -0.6, \\ \text{no\_change}; & \text{if } -0.6 < \Delta(\mathrm{Solubility}) < 0.6, \\ \text{low\_high}; & \text{if } 0.6 \le \Delta(\mathrm{Solubility}). \end{cases} \tag{2}$$

The condition token of solubility can be defined as Delta_Solubilityhigh_low, Delta_Solubilityno_change, and Delta_Solubilitylow_high. The condition token of Clint is similar to Solubility, but the threshold is 0.35. Besides property improvement where higher values are desirable, GPMO can optimize molecules by selectively improving one property while keeping other properties unchanged or lower. This is accomplished by incorporating the condition tokens that specify the desired property changes.

3.3 Positive and Negative Molecules Generation

To generate negative and positive molecules for contrastive learning without domain-specific knowledge, two kinds of gradient perturbation are used. The first kind is a small negative perturbation, which generates a negative molecule that is close to the real molecule. The second kind is a large positive perturbation, which involves two steps to generate a positive molecule that is far away from the real molecule.

Small Negative Perturbation for Negative Molecule

For negative perturbation d, the log likelihood with respect to d is $p_\theta(y_t^i|\mathbf{y}_{<\mathbf{t}}^i, \mathbf{x}^i; \mathbf{s}_t^i + d) = \mathrm{softmax}(\mathbf{W}(\mathbf{s}_t^i + d) + \mathbf{b})$. As d is a negative perturbation, the log-likelihood with respect to d should be minimized. Generally, a linear approximation can be defined as:

$$\check{\mathbf{S}}^i = \mathbf{S}^i - \lambda \frac{\mathbf{g}}{\|\mathbf{g}\|_2}, \tag{3}$$

where $\mathbf{g} = \nabla_{\mathbf{S}^i} \mathcal{L}_{MLE}$ is the negative perturbation, which is the gradient of $\mathbf{S}^i$ in the back-propagation of $\mathcal{L}_{MLE}$. $\|.\|_2$ is the l2 norm function and λ controls the weight of negative perturbation. As g is the gradient of $\mathbf{S}^i$, thus $\check{\mathbf{S}}^i$ is near $\mathbf{S}^i$ in the embedding space and different from $\mathbf{S}^i$. The contrastive learning with negative molecule gained by negative perturbation can be defined as:

$$\mathcal{L}_{cont-}(\theta) = \sum_{i=1}^{N} \log \frac{\exp\left(\cos\left(\mathbf{z}_x^i, \mathbf{z}_y^i\right)/\tau\right)}{\sum_{\mathbf{z}_y^- \in U^i} \exp\left(\cos\left(\mathbf{z}_x^i, \mathbf{z}_y^-\right)/\tau\right)}, \tag{4}$$

where $\mathbf{z}_x^i = \varphi(\mathbf{H}^i)$ and $\mathbf{z}_y^i = \varphi(\mathbf{S}^i)$. $\mathbf{U}^i = \{\mathbf{S}^j \cup \check{\mathbf{S}}^i\}$, $\mathbf{S}^j = \{\mathbf{S}^j : j \neq i\}$, and $\mathbf{z}_y^- = \varphi(\mathbf{U}^i)$. $\varphi(.)$ is a projection function with a linear layer and an average pooling operation. $\mathbf{S}^j$ is the random molecule in the batch which acts as the negative molecule. Minimizing of $\mathcal{L}_{cont-}(\theta)$ will force the negative molecule $\check{\mathbf{z}}_y^i$ away from the real molecule.

Large Positive Perturbation for Positive Molecule

Two-step perturbation is used to generate a positive molecule that is far away from the real molecule. The first perturbation is the gradient with respect to contrastive loss, which maximizes the similarity between paired molecules and minimizes the similarity between unpaired molecules in the same batch. Then, the intermediate positive molecule is obtained based on the first step of positive perturbation. After that, KL loss is used to constrain distribution similarity between the intermediate positive molecule and the real molecule. Next, the second positive perturbation is the gradient based on KL loss. Finally, another positive molecule is obtained based on the second step of positive perturbation.

To gain the first gradient perturbation, the contrastive loss is used to train the model to learn the representations of the real molecule by contrasting the molecule pairs with the unpaired molecules. The unpaired molecules are the randomly sampled non-target output molecule in the same batch. As shown in Figure 1c, the start and target molecules are projected onto the latent embedding space. Then, contrastive learning is used to contrast the paired molecules with the unpaired molecules. We define this as:

$$\mathcal{L}_{cont}(\theta) = \sum_{i=1}^{N} \log \frac{\exp\left(\cos\left(\mathbf{z}_x^i, \mathbf{z}_y^i\right)/\tau\right)}{\sum_{\mathbf{z}_y^j \in S^i} \exp\left(\cos\left(\mathbf{z}_x^i, \mathbf{z}_y^j\right)/\tau\right)}, \tag{5}$$

where $\mathcal{L}_{cont}(\theta)$ maximizes the representation similarity between molecule pairs and minimizes the similarity between unpaired molecules within the same batch. $\mathcal{L}_{cont}(\theta)$ trains the model to learn the representations of the real molecule. The gradient perturbation of $\mathbf{S}^i$ in the back-propagation of $\mathcal{L}_{cont}(\theta)$ is a positive perturbation. Thus the first positive perturbation can be defined as $\mathbf{f}^+ = \nabla_{\mathbf{S}^i} \mathcal{L}_{cont}$.

The first positive perturbation is used to generate the intermediate positive molecule. The hidden state of the intermediate positive molecule can be defined as:

$$\bar{\mathbf{S}}^i = \mathbf{S}^i - \mu \frac{\mathbf{f}^+}{\|\mathbf{f}^+\|_2}, \tag{6}$$

where μ controls the weight of the first positive perturbation. As $\bar{\mathbf{S}}^i$ is the intermediate positive molecule, the distribution of $\bar{\mathbf{S}}^i$ after the softmax layer should be similar to the real molecule. The KL loss is used to minimize the distribution between intermediate positive molecule and the real molecule which can be defined as:

$$\mathcal{L}_{KL}(\theta) = \sum_{i=1}^{N} D_{KL}(p_\theta(\mathbf{y}^i, \mathbf{x}^i) || p_\theta(\bar{\mathbf{y}}^i, \mathbf{x}^i)), \tag{7}$$

where $p_\theta(\bar{\mathbf{y}}^i, \mathbf{x}^i) = \mathrm{softmax}(\mathbf{W}\bar{\mathbf{S}}_t^i + \mathbf{b})$. The distribution similarity of the positive molecule and the real molecule is achieved by minimizing KL loss.

As the KL loss maximizes the similarity between the intermediate positive and real molecules, the gradient perturbation obtained from KL loss can also be used as the second positive perturbation to generate the final positive molecule. The hidden state of the positive molecule gained by the second positive perturbation can be defined as:

$$\hat{\mathbf{S}}^i = \bar{\mathbf{S}}^i - \mu \frac{\mathbf{f}^{++}}{\|\mathbf{f}^{++}\|_2}, \tag{8}$$

where $\mathbf{f}^{++} = \nabla_{\bar{\mathbf{S}}^i} \mathcal{L}_{KL}$. The molecule representation $\hat{\mathbf{S}}^i$ is obtained through a two-step positive perturbation, resulting in a positive molecule that is significant far away from the real molecule in the embedding space. Then, the contrastive loss with the positive molecule can be written as:

$$\mathcal{L}_{cont+}(\theta) = \sum_{i=1}^{N} \log \frac{\exp\left(\cos\left(\mathbf{z}_{\mathbf{x}}^i, \hat{\mathbf{z}}_{\mathbf{y}}^i\right)/\tau\right)}{\sum_{\mathbf{z}_y^- \in U^i} \exp\left(\cos\left(\mathbf{z}_x^i, \mathbf{z}_y^-\right)/\tau\right)}, \tag{9}$$

where $\hat{\mathbf{z}}_y^i = \varphi(\hat{\mathbf{S}}^i)$. $\mathcal{L}_{cont+}$ is the contrastive loss, which trains the model to learn the representation of the real molecule by contrasting the hard distinguished positive molecule with a negative molecule.

$\mathcal{L}_{cont-}(\theta)$ and $\mathcal{L}_{cont+}(\theta)$ contrast real molecule with generated negative and positive molecules, aiming to differentiate between them and align the generated positive molecules with the real ones. The real molecule acts as an anchor and its similarity with positive molecule is maximized, while its similarity with negative molecule is minimized (Figure 1a). Finally, the parameters of the encoder-decoder can be estimated by minimizing the following objective:

$$\mathcal{L} = \max_\theta \mathcal{L}_{MLE}(\theta) - \mathcal{L}_{KL}(\theta) + \mathcal{L}_{cont-}(\theta) + \mathcal{L}_{cont+}(\theta).$$

The minimize of $\mathcal{L}$ trains the seq2seq model using diverse negative and positive examples through contrastive learning, thereby mitigating the "exposure bias" problem.

3.4 Self-Supervised Pre-Training

To autoregressively decode molecules from input molecules and conditions, GPMO is pre-trained using self-supervised tasks such as MLM and SMILES translation. This pre-training enables GPMO to learn the grammar of SMILES and the relationship between molecules.

- MLM. Canonical SMILES are used in MLM. For the input of the MLM, 15% of the input tokens are randomly masked. However, due to the large number of carbon atoms in molecules, reconstructing only the masked tokens may lead to overfitting on carbon atoms. Therefore, the reconstruction of the entire Canonical SMILES is predicted instead. The MLM is employed to learn the grammar of SMILES.
- SMILES translation. A major challenge in MO is understanding the relationship between two molecules, which is not directly captured by MLM task. To address this, the SMILES translation pre-training task is employed to train the GPMO model to learn the relationship between two molecules. In the SMILES translation, a many-to-one translation approach is employed. It involves using SMILES with a randomly chosen root atom as the input and generating the corresponding Canonical SMILES as the output, where multiple random SMILES representations of a molecule are encoded into similar embeddings. This encourages the decoder to focus on improving its ability to accurately decode Canonical SMILES.

In the pre-training of GPMO, the MLM and SMILES translation tasks are selected randomly with a 50% chance.

	Pairs	LogD	Solubility	Clint
Train	$160,832$	$[-1.44, 5.65]$	$[-1.51, 5.65]$	$[0.18, 3.03]$
Val	$17,872$	$[-1.37, 5.64]$	$[-1.44, 4.03]$	$[0.20, 2.99]$
Test	$19,857$	$[-1.43, 5.50]$	$[-1.45, 4.01]$	$[0.24, 3.01]$

Table 1: Statistics of the dataset. Val is the abbreviation of validation. The train, validation, and test set are divided with a random split.

4 Experiment

To showcase the effectiveness of GPMO, extensive experiments were conducted on benchmark tasks. GPMO outperforms deep learning techniques, showcasing its potential as a powerful tool for molecule optimization.

4.1 Experiment Setup

GPMO is trained in the pre-training stage using the dataset from MOSES [Polykovskiy *et al.*, 2020]. In the molecule optimization stage, GPMO utilizes a dataset from previous work [He *et al.*, 2021], and the statistical information is presented in Table 1. The table displays the number of molecule pairs and the property ranges in the train, validation, and test sets. For property prediction, an ensemble random forest model trained on [He *et al.*, 2021] is employed.

Baseline Methods

The following MO baselines are compared with GPMO:

- JTNN [Jin *et al.*, 2018b] approaches MO by learning to translate molecule pairs.
- HierG2G [Jin *et al.*, 2020] employs a hierarchical approach to encode molecule graphs and creates compounds by connecting structural motifs.
- Modof [Chen *et al.*, 2021] conducts MO by predicting the removal, addition, or replacement at the disconnection site.
- Transformer [Vaswani *et al.*, 2017] utilizes a seq2seq approach for MO, where the starting and target molecules serve as the input and output, respectively.
- GPMO-w/o-per is a variant of GPMO that does not use gradient perturbation-based contrastive learning.

	LogD	Solubility	Clint	All	Validity	MMP	Similarity
JTNN	81.72±1.32	99.04±0.61	98.64±0.85	78.76±0.71	89.77±0.68	50.00±0.91	54.14±0.87
HierG2G	88.84±1.08	99.78±0.57	99.78±0.73	85.42±0.62	90.26±0.74	60.42±0.84	44.66±0.62
Modof	91.65±0.99	99.83±0.46	99.86±0.53	87.39±0.44	92.34±0.69	61.66±0.71	59.72±0.56
Transformer	91.12±0.54	99.93±0.21	98.78±0.37	87.38±0.30	93.88±0.48	90.41±0.62	67.37±0.51
GPMO-w/o-per	95.62±0.31	99.73±0.36	99.89±0.14	93.34±0.32	99.91±0.17	88.87±0.46	**67.78±0.34**
GPMO-w/o-con	93.97±0.11	99.62±0.10	99.80±0.09	91.59±0.08	99.87±0.05	89.46±0.11	66.03±0.07
GPMO	**97.28±0.04**	**99.99±0.00**	**99.98±0.03**	**96.72±0.02**	**99.93±0.04**	**91.31±0.07**	67.33±0.02

Table 2: Overall comparison of baseline methods and variants of GPMO.

Ratio	JTNN		HierG2G		Modof		Transformer		GPMO	
	All	Validity	All	Validity	All	Validity	All	Validity	All	Validity
50%	76.01	9.19	83.96	80.24	85.44	89.12	87.26	91.45	**94.86**	**97.31**
20%	73.33	78.64	81.97	77.39	82.69	86.99	84.93	88.21	**92.82**	**95.90**
10%	70.65	76.21	78.61	75.54	79.74	80.17	76.55	79.34	**88.77**	**86.94**

Table 3: Results on smaller datasets. The ratio refers to the proportion of the train, validation, and test datasets we used as the smaller dataset.

- GPMO-w/o-con is a variant of GPMO that does not include the use of condition tokens in the input of the encoder.

Metric

We consider the following metrics for the evaluation of generated molecules.

- LogD: the proportion of generated molecules that satisfy the LogD constraint, where the absolute difference in LogD between the generated molecule and target molecule is no more than 0.4.
- Solubility: the proportion of code(Solubility($y\prime$)-Solubility(x)) equal to code(Solubility(y)-Solubility(x)), where x is the start molecule, $y\prime$ is the generated molecule, and y is the target molecule. Code(.) refers to the code table defined in Eq. (2).
- Clint: the definition of Clint is similar to Solubility.
- All: the proportion of generated molecules that satisfy the constraints for LogD, Solubility, and Clint constraints simultaneously.
- Validity: the proportion of chemically valid molecules among all generated molecules.
- MMP: the proportion of generated molecules that exhibit an MMP relation with the starting molecules. The MMP relation is defined using the MMP query tool mmpdb [Dalke *et al.*, 2018].
- Similarity: the Tanimoto similarity between the starting and generated molecules with Morgan fingerprint.

4.2 Multi-Property Optimization

As shown in Table 2, GPMO outperforms the state-of-the-art models on all seven metrics. From the results shown in Table 2, GPMO outperforms baselines on single-property performance metrics such as LogD, Solubility, and Clint. In terms of multi-property optimization, GPMO outperforms the best baseline method by a significant margin of 9.14% on the All metric. GPMO achieves the best performance on Validity due to it is pre-trained on unlabeled data, which enables it to learn the grammar on SMILES. Our result for the MMP and Similarity metrics indicate that the molecules generated from GPMO maintain the MMP relation and chemical similarity with the input molecules. Based on our analysis, we can conclude that GPMO has the ability to generate valid molecule with improved properties while retaining the starting molecule.

One key factor contributing to GPMO's impressive performance is the use of the condition tokens, which allow for the differentiation of distinct optimization objectives among pairs that share the same starting molecule. In contrast, JTNN, HierG2G, and Modof are unable to distinguish between different molecule pairs that have the same starting molecule. Consequently, the optimization of these molecule pairs will be an average of all molecule pairs that share the same starting molecule in the training set. Compared with Transformer, which also uses condition tokens, GPMO achieves better performance on Validity metric due to its pre-trained on unlabeled data, which enables it to learn the grammar of SMILES.

When compared with GPMO-w/o-per, GPMO lags behind on Similarity metric since it learns a broader range of valid or incorrect samples, resulting in less chemical similarity with a starting molecule. Our findings reveal that GPMO outperforms GPMO-w/o-per by an additional 3.38% on the All score, demonstrating that our model generates a greater number of desired molecules when compared to its counterpart. Given that the variance of GPMO-w/o-per is higher than GPMO, GPMO is more robust than GPMO-w/o-per. This is likely due to the contrastive learning approach employed by GPMO. When condition tokens are removed from GPMO, as in the case of GPMO-w/o-con shown in Table 2, the Validity rate remains high. However, the All rate drops down to 0.9356 due to the loss of ability to distinguish molecule pairs with the same starting molecule.

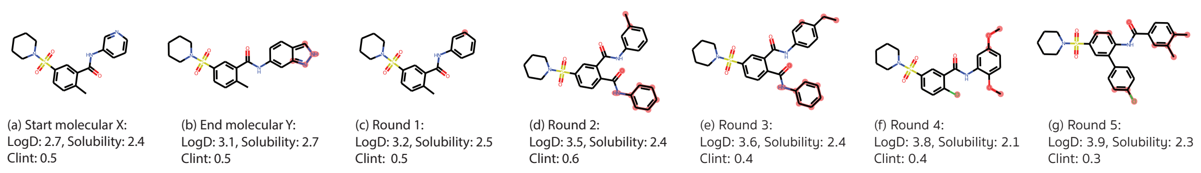
(a) Start molecular X:
LogD: 2.7, Solubility: 2.4
Clint: 0.5
(b) End molecular Y:
LogD: 3.1, Solubility: 2.7
Clint: 0.5
(c) Round 1:
LogD: 3.2, Solubility: 2.5
Clint: 0.5
(d) Round 2:
LogD: 3.5, Solubility: 2.4
Clint: 0.6
(e) Round 3:
LogD: 3.6, Solubility: 2.4
Clint: 0.4
(f) Round 4:
LogD: 3.8, Solubility: 2.1
Clint: 0.4
(g) Round 5:
LogD: 3.9, Solubility: 2.3
Clint: 0.3

Figure 2: The multi-step molecule optimization with five iterations. The multi-property molecule optimization setting is {(0.3,0.5], no_change, no_change}. The modified fragments in each step are highlighted in red.

λ, μ	All	Validity	MMP	Similarity
$1, 1$	96.47	99.94	91.31	67.32
$1, 3$	96.51	99.87	91.29	67.30
$1, 5$	96.54	99.85	91.33	67.33
$3, 1$	96.71	99.89	91.24	67.31
$3, 3$	**96.73**	**99.97**	**91.38**	**67.35**
$3, 5$	96.44	99.84	91.37	67.34
$5, 1$	96.36	99.83	91.32	67.35
$5, 3$	96.37	99.83	91.37	67.35
$5, 5$	96.35	99.82	91.35	67.34

Table 4: Performance of GPMO under different hyper-parameters under different gradient perturbation strengths. Note: λ is the weight of negative perturbation and μ is the weight of positive perturbation.

4.3 Model Performance on Smaller Datasets

Since experimentally obtained datasets are both expensive and rare, it is crucial to employ deep learning models that can handle small-scale datasets. In our study, we conducted experiments to evaluate the performance of GPMO in comparison to baseline methods on reduced versions of the train, validation, and test datasets, with only 50%, 20%, and 10% of the original dataset sizes, respectively.

The performance of each method is measured in terms of the All and Validity metrics, which respectively measure the method's ability to perform property optimization and generate valid molecules. From the results shown in Table 3, we observe that the All and Validity metrics decayed with the reduction of samples. Despite the performance decay observed with the reduction of training samples, our model still shows competitive results compared to the baseline methods on the small-scale dataset. More specifically, in the case of the 10% dataset, GPMO achieves an 88.77% All metric on propriety optimization, which is 9.03% higher than the best baseline method. Moreover, GPMO exhibits a 6.77% improvement on the Validity metric. These results highlight the benefiting from pre-training tasks, which enable GPMO to exhibit reduced reliance on labeled data.

4.4 Sensitivity Analysis

Table 4 provides an overview of the overall performance of GPMO under across various combinations hyper-parameters λ and μ. We set the weight of negative perturbation $\lambda = [1, 3, 5]$ and the weight of positive perturbation $\mu = [1, 3, 5]$. We observe that the variance in performance across different gradient perturbations combinations was no more than 0.14, indicating the robustness of GPMO. Our results indicate that the hyper-parameter combination of $\lambda = 3$ and $\mu = 3$ achieved the best performance, suggesting that overly large or small perturbation are inefficient. This is likely due to the fact that small perturbations produce negative and positive samples that show less contrast with real samples, while large perturbation can generate artificial samples that make it difficult for the model to convergence.

4.5 Multi-Step Molecule Optimization

In multi-step MO, molecules are continuously optimized based on prior experience, much like the way chemists optimize molecules. To achieve this, we use the currently optimized molecule as the starting molecule in the next round of optimization with the same property constraint. We repeat this process five times to generate a set of molecules with the same property constraint.

Figure 2 presents the results of multi-step MO with five steps, under the property constraints {(0.3,0.5], no_change, no_change}, which require improving the LogD of the target molecule while maintaining unchanged Solubility and Clint properties. The results of multi-step molecule optimization show an increase in LogD from 2.7 to 3.9, while the Solubility and Clint remain unchanged. Moreover, the scaffold is preserved throughout the optimization process. In the multi-step optimization setting, our model is capable of optimizing the molecule with desired properties while constraining the same scaffold with a starting molecule.

5 Conclusion

In this work, we propose GPMO, a gradient perturbation-based contrastive learning network for molecule optimization. Extensive experiments demonstrate that with proper design, GPMO is able to find molecules that satisfy multiple properties and its performance is comparable to advanced methods using deep learning. The reasons for GPMO's strong performance include the use of gradient perturbation contrastive learning, the pre-trained LM which reduces the need for labeled molecule pairs, the condition tokens on properties distinguish different molecule pairs with the same starting molecules, and the use of the MMP dataset to ensures scaffold similarity between the starting and generated molecules. GPMO can be readily adapted to real drug discovery projects, such as large-scale MMP analysis processes without chemist guidance. Furthermore, properties other than LogD, Solubility, and Clint, such as QED/SA and other ADMET properties, can also be included to search for more promising candidates.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (62122025, U22A2037, 62102140), Hunan Provincial Natural Science Foundation of China (2021JJ10020), and the Science and Technology Innovation Program of Hunan Province (2022RC1100).

References

[Chen *et al.*, 2021] Ziqi Chen, Martin Renqiang Min, Srinivasan Parthasarathy, and Xia Ning. A deep generative model for molecule optimization via one fragment modification. *Nature Machine Intelligence*, 3(12):1040–1049, 2021.

[Dalke *et al.*, 2018] Andrew Dalke, Jerome Hert, and Christian Kramer. mmpdb: An open-source matched molecular pair platform for large multiproperty data sets. *Journal of Chemical Information and Modeling*, 58(5):902–910, 2018.

[Fu *et al.*, 2021] Tianfan Fu, Cao Xiao, Xinhao Li, Lucas M Glass, and Jimeng Sun. Mimosa: Multi-constraint molecule sampling for molecule optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 125–133, 2021.

[Gao *et al.*, 2021] Wenhao Gao, Rocío Mercado, and Connor W Coley. Amortized tree generation for bottom-up synthesis planning and synthesizable molecular design. *arXiv preprint arXiv:2110.06389*, 2021.

[Gómez-Bombarelli *et al.*, 2018] Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS Central Science*, 4(2):268–276, 2018.

[Guimaraes *et al.*, 2017] Gabriel Lima Guimaraes, Benjamin Sanchez-Lengeling, Carlos Outeiral, Pedro Luis Cunha Farias, and Alán Aspuru-Guzik. Objective-reinforced generative adversarial networks (organ) for sequence generation models. *arXiv preprint arXiv:1705.10843*, 2017.

[He *et al.*, 2021] Jiazhen He, Huifang You, Emil Sandström, Eva Nittinger, Esben Jannik Bjerrum, Christian Tyrchan, Werngard Czechtizky, and Ola Engkvist. Molecular optimization by capturing chemist's intuition using deep neural networks. *Journal of Cheminformatics*, 13(1):1–17, 2021.

[Jin *et al.*, 2018a] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction tree variational autoencoder for molecular graph generation. In *International Conference on Machine Learning*, pages 2323–2332. PMLR, 2018.

[Jin *et al.*, 2018b] Wengong Jin, Kevin Yang, Regina Barzilay, and Tommi Jaakkola. Learning multimodal graph-to-graph translation for molecule optimization. In *International Conference on Learning Representations*, 2018.

[Jin *et al.*, 2020] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Hierarchical generation of molecular graphs using structural motifs. In *International Conference on Machine Learning*, pages 4839–4848. PMLR, 2020.

[Lee *et al.*, 2020] Seanie Lee, Dong Bok Lee, and Sung Ju Hwang. Contrastive learning with adversarial perturbations for conditional text generation. In *International Conference on Learning Representations*, 2020.

[Lin *et al.*, 2020] Xuan Lin, Zhe Quan, Zhi-Jie Wang, Tengfei Ma, and Xiangxiang Zeng. KGNN: Knowledge Graph Neural Network for Drug-Drug Interaction Prediction. In *IJCAI*, volume 380, pages 2739–2745, 2020.

[Liu *et al.*, 2020] Xianggen Liu, Qiang Liu, Sen Song, and Jian Peng. A chance-constrained generative framework for sequence optimization. In *International Conference on Machine Learning*, pages 6271–6281. PMLR, 2020.

[Ma *et al.*, 2022] Tengfei Ma, Xuan Lin, Bosheng Song, S Yu Philip, and Xiangxiang Zeng. KG-MTL: Knowledge Graph Enhanced Multi-Task Learning for Molecular Interaction. *IEEE Transactions on Knowledge & Data Engineering*, pages 1–12, 2022.

[Moss *et al.*, 2020] Henry Moss, David Leslie, Daniel Beck, Javier Gonzalez, and Paul Rayson. Boss: Bayesian optimization over string spaces. *Advances in Neural Information Processing Systems*, 33:15476–15486, 2020.

[Nigam *et al.*, 2019] AkshatKumar Nigam, Pascal Friederich, Mario Krenn, and Alán Aspuru-Guzik. Augmenting genetic algorithms with deep neural networks for exploring the chemical space. *arXiv preprint arXiv:1909.11655*, 2019.

[Polykovskiy *et al.*, 2020] Daniil Polykovskiy, Alexander Zhebrak, Benjamin Sanchez-Lengeling, Sergey Golovanov, Oktai Tatanov, Stanislav Belyaev, Rauf Kurbanov, Aleksey Artamonov, Vladimir Aladinskiy, Mark Veselov, et al. Molecular sets (MOSES): a benchmarking platform for molecular generation models. *Frontiers in Pharmacology*, 11:565644, 2020.

[Shen *et al.*, 2022] Xinke Shen, Xianggen Liu, Xin Hu, Dan Zhang, and Sen Song. Contrastive learning of subject-invariant EEG representations for cross-subject emotion recognition. *IEEE Transactions on Affective Computing*, pages 1 – 1, 2022.

[Shi *et al.*, 2020] Chence Shi, Minkai Xu, Zhaocheng Zhu, Weinan Zhang, Ming Zhang, and Jian Tang. GraphAF: a Flow-based Autoregressive Model for Molecular Graph Generation. In *International Conference on Learning Representations*, 2020.

[Vaswani *et al.*, 2017] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.

[Wang *et al.*, 2021] Jike Wang, Chang-Yu Hsieh, Mingyang Wang, Xiaorui Wang, Zhenxing Wu, Dejun Jiang, Benben Liao, Xujun Zhang, Bo Yang, Qiaojun He, et al. Multi-constraint molecular generation based on conditional transformer, knowledge distillation and reinforcement learning. *Nature Machine Intelligence*, 3(10):914–922, 2021.

[You *et al.*, 2018] Jiaxuan You, Bowen Liu, Zhitao Ying, Vijay Pande, and Jure Leskovec. Graph convolutional policy network for goal-directed molecular graph generation. *Advances in Neural Information Processing Systems*, 31, 2018.

[Zang and Wang, 2020] Chengxi Zang and Fei Wang. MoFlow: an invertible flow model for generating molecular graphs. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 617–626, 2020.

[Zeng *et al.*, 2022] Xiangxiang Zeng, Hongxin Xiang, Linhui Yu, Jianmin Wang, Kenli Li, Ruth Nussinov, and Feixiong Cheng. Accurate prediction of molecular properties and drug targets using a self-supervised image representation learning framework. *Nature Machine Intelligence*, 4(11):1004–1016, 2022.

[Zhou *et al.*, 2019] Zhenpeng Zhou, Steven Kearnes, Li Li, Richard N Zare, and Patrick Riley. Optimization of molecules via deep reinforcement learning. *Scientific Reports*, 9(1):1–10, 2019.