Cross-modal Generation of Hit-like Molecules via Foundation Model Encoding of Gene Expression Signatures

Jia-Bei Cheng[1], Xiaoyong Pan[1], Kaiyuan Yang[1], Shenghao Cao[1], Bin Liu[1] and Ye Yuan[1,*]

[1]Institute of Image Processing and Pattern Recognition, Shanghai Jiao Tong University, and Key Laboratory of System Control and Information Processing, Ministry of Education of China, Shanghai 200240, China

*To whom correspondence should be addressed.

Contact: yuanye_auto@sjtu.edu.cn

Abstract: Designing hit-like molecules from gene expression signatures takes into account multiple targets and complex biological effects, enabling the discovery of multi-target drugs for complex diseases. Traditional methods relying on similarity searching against a database are limited by the quality and size of the databases. Instead, multimodal deep learning offers the potential to overcome this bottleneck. Additionally, the recent development of foundation models provides a new perspective to understand gene expression levels. Thus, we propose GexMolGen (Gene Expression-based Molecule Generator) based on a foundation model scGPT to generate hit-like molecules from gene expression differences. By taking the desired and control gene expression profile as inputs, GexMolGen designs molecules that can induce the required transcriptome profile. The molecules generated by GexMolGen

exhibit a high similarity to known gene inhibitors. Overall, GexMolGen explores the chemical and biological relationships in the drug discovery process. The source code and demo are available at https://github.com/Bunnybeibei/GexMolGen.

Keywords: Hit-like molecule design, Cross-modal generation, Foundation models, Contrastive Learning, Autoencoder, Autoregressive Model

Introduction

Drug discovery has two main groups of approaches: phenotype-based and target-based . Target-based drug discovery relies on identifying and selecting specific targets based on mechanistic models, but it may not effectively consider the complex context of human physiology. On the other hand, phenotypic drug discovery involves observing the therapeutic effects of new drugs on disease models to verify and screen potential drug candidates, i.e. ibuprofen and minoxidil [1]. One classic case of this method is the L1000 project, which enables searching for small molecules with similar gene expression signatures in a database. However, the effectiveness of this approach largely depends on the size and quality of the database, and the number of recorded small molecules is much smaller than the actual drug space [2]. Considering the correlation between small molecule data and gene expression changes, multimodal techniques are being exploited to overcome these limitations.

With the continuous advancement of multimodal learning [3] [4], many progresses have emerged in the field of AI for Science, including text-molecule

matching [5] and text-protein sequence generation [6]. In the context of phenotype-based drug discovery, deep learning techniques have also been employed. For example, Xie et al., [7] utilize gene expression changes to represent drug efficacy and screen molecules, while Oscar Méndez-Lucio [8] and Dibyajyoti Das [9] directly generate hit-like small molecules based on gene expression signatures. However, it should be noted that some of these approaches [7] do not fully utilize gene expression changes, and some methods [8] [9] solely rely on multilayer perceptrons (MLPs) to process gene expression changes. Consequently, these methods may lack comprehensive interaction between gene information and small molecule information. The Transformer framework, renowned for its consideration of global information and provision of flexibility in model design and expansion, offers a promising avenue for gene encoding.

Since Theodoris et al. first applied the Transformer to single-cell transcriptome data to build the Geneformer[10] model, there has been a surge in the development of large models specifically designed for transcriptome data. Prominent examples include scBERT[11], TOSICA [12], scGPT [13], scFoundation [14], SCimilarity[15], and GeneCompass[16]. These general large models serve as foundation models, offering deeper insights into gene interactions and achieving impressive results in various downstream tasks, demonstrating their zero-shot capabilities and robustness.

With the AI-driven evolution of phenotype-based drug discovery, the rapid advancement of multimodal technologies, and the impressive performance of foundation models, we propose GexMolGen—a model that utilizes gene

transcriptomic profiles to guide the generation of hit-like molecules. In this study, we explore the application of the gene transcriptional profiling-based foundation model scGPT in the field of molecule generation for the first time. Different from previous methods that commonly rely on one-dimensional SMILES representation [8] [9], we synergistically incorporate the two-dimensional molecular graph representation with the gene modality, offering potential for scalability and interpretability. Compared to traditional methods and other works, our model achieves higher similarity to known inhibitors in the task of generating specific gene inhibitors.

Dataset preprocessing

The datasets used in GexMolGen are derived from the L1000 public dataset (GEO ID: GSE70138) [17]. The L1000 technology uses microarray measurements to record the changes in gene expression caused by over 25,200 perturbagens, including around 19,800 small molecules, in cell lines. This dataset is processed hierarchically and divided into five levels: Levels 1-3 show the absolute expression levels of each gene in each bead, while Levels 4-5 show the differentially expression levels of each gene by comparing samples within each bead.

During the data preprocessing stage, three issues are considered. 1. The pre-trained model scGPT[13] is trained on absolute gene expression levels, while most L1000-related work primarily relies on Level 5 data. 2. The sample sizes for each small molecule are extremely imbalanced (maximum to minimum ratio of 22:1). 3. Splitting the dataset based on SMILES representations may result in the same

molecule being included in both the training and test sets, due to the non-uniqueness of SMILES.

To address the above issues, we process the data as follows: 1. Use Level 3 data to encode both the experimental and control groups and subsequently calculate gene expression signatures. 2. Average gene expression to ensure each small molecule can be represented by a single data point. 3. Use the main layer of InChI-key to partition the training, validation, and test sets, thus enabling unambiguous identification of compounds.

Specifically, we first select the experimental data and control data from Level 3, ensuring that they are from the same bead. These data belong to six major cell lines (VCAP, PC3, A549, A375, HT29, MCF7). We then calculate the average gene expression for each small molecule, resulting in a final dataset size of 2 × 15156 × 978: 2 represents the experimental and control groups, 15156 represents the sample size, and 978 represents the number of gene categories. Additionally, we divide the structures by the main layer of InChI-key, resulting in a total of 6360 structures. From this set, we randomly select 300 structures (comprising 708 types of small molecules) as the test set, and another 300 structures (comprising 691 types of small molecules) as the validation set. The remaining data is used for model training.

Overview of GexMolGen

GexMolGen divides the task of generating hit-like molecules from gene expression profiles into four steps (as shown in **Figure 1**): 1. Encoding of gene and small

molecule data, 2. Matching of the gene modality and small molecule modality, 3. Transforming gene modality to small molecule modality, 4. Decoding of small molecule modality. To simplify the process, we use pre-trained models for both the encoder and decoder in steps 1 and 4, respectively, namely scGPT [13] and hierVAE [18]. We introduce step 2 to address the significant differences and complex mechanisms between gene modality and small molecule modality. Matching and then transforming the modalities can improve the quality and controllability of the cross-modal generation model [19] [20].

Data Encoding

In Step 1, we encode the transcriptomic data and small molecule data separately based on scGPT[13] and hierVAE[18].

Gene Encoder

For gene encoder, scGPT introduces gene tokens to capture additional information. It treats a transcript as a sentence, with genes as words. This allows us to create a specific dictionary for each gene. We obtain the gene tokens for the i-th data using a pre-trained specific gene dictionary:

$$t_g^{(i)} = [id(g_1^{(i)}), id(g_2^{(i)}) \ldots\ldots id(g_M^{(i)})]$$

where $t_g^{(i)}$ represents the gene token encoding for the i-th data, and $id\left(g_{\mathrm{j}}^{(i)}\right)$ represents the unique integer identifier for gene $\mathrm{g_j}$ in the i-th data. In our case, we set M as 968 since there are 10 genes that do not appear in the pre-trained gene dictionary.

Next, we preprocess the expression values following the same steps as scGPT:

TPM normalization and log1p transformation.

$$x^{(i)} = \left[x_1^{(i)}, x_2^{(i)} \dots \dots x_M^{(i)}\right]$$

The gene embedding is then defined as follows:

$$z_g^{(i)} = emb_g(t_g^{(i)}) + emb_x(x^{(i)})$$

where $emb_g(\cdot)$ refers to the conventional embedding layers, and $emb_x(\cdot)$ represents the fully connected layer. The experimental group and control group are independently encoded using the same approach.

Molecule Encoder

For the molecule encoder, hierVAE regards motifs as the smallest informational units for generating small molecules and constructs a motif dictionary. This approach achieves high validity in molecular generation. We use the preprocessing program of hierVAE to obtain the atom layer $\mathcal{H}_{\mathcal{G}}^{g}$, the attachment layer $\mathcal{H}_{\mathcal{G}}^{\mathrm{a}}$, and the motif layer $\mathcal{H}_{\mathcal{G}}^{\mathrm{s}}$ for small molecules, where the previous layer can serve as the encoding condition for the subsequent layer, enabling hierarchical encoding.

The input for the atom layer $\mathcal{H}_{\mathcal{G}}^{g}$ is the molecular graph $\mathcal{G}$, the edge features $\{\mathrm{e}(\mathrm{b_{uv}})\}$, and the node features $\{\mathrm{e}(\mathrm{a_u})\}$. Through the message passing neural network[21][22] $MPN_{\psi}(\cdot)$, we obtain the encoded features:

$$\mathrm{c}_{\mathcal{G}}^{\mathrm{g}} = \{\mathrm{h_v}\} = MPN_{\psi_1}(\mathcal{H}_{\mathcal{G}}^{g}, \{\mathrm{e}(\mathrm{a_u})\}, \{\mathrm{e}(\mathrm{b_{uv}})\})$$

where $\mathrm{h_v}$ represents the updated atomic features.

The attachment layer $\mathcal{H}_{\mathcal{G}}^{\mathrm{a}}$ updates its nodes using the following equation:

$$\mathrm{c}_{\mathcal{G}}^{a} = \{\mathrm{h}_{\mathcal{A}_\mathrm{i}}\} = MPN_{\psi_2}(\mathcal{H}_{\mathcal{G}}^{g}, \{f_{\mathcal{A}_\mathrm{i}}\}, \{\mathrm{e}(\mathrm{d_{ij}})\})$$

where $\{\mathrm{e}(\mathrm{d_{ij}})\}$ denotes the edge features for the motif layer, and $\{f_{\mathcal{A}_\mathrm{i}}\}$ can be

calculated based on $\mathrm{h_v}$ and $\mathcal{A}_\mathrm{i}$:

$$f_{\mathcal{A}_\mathrm{i}} = emb_{\mathcal{A}_\mathrm{i}}(\mathrm{e}(\mathcal{A}_\mathrm{i}), \sum\nolimits_{\mathrm{v}\in\mathrm{S_i}} \mathrm{h_v})$$

Here, $emb_{\mathcal{A}_\mathrm{i}}(\cdot)$ represents a fully connected layer, and $\mathcal{A}_\mathrm{i}$ is the set of overlapping atoms between adjacent motifs.

The motif layer $\mathcal{H}_\mathcal{G}^\mathrm{s}$ updates its nodes using the following equation:

$$\mathrm{c}_\mathcal{G}^\mathrm{s} = \{\mathrm{h}_{\mathcal{S}_\mathrm{i}}\} = MPN_{\psi_3}(\mathcal{H}_\mathcal{G}^\mathrm{s}, \{f_{\mathcal{S}_\mathrm{i}}\}, \{\mathrm{e}(\mathrm{d_{ij}})\})$$

where $\{\mathrm{e}(\mathrm{d_{ij}})\}$ denotes the edge features for the motif layer, and $\{f_{\mathcal{S}_\mathrm{i}}\}$ can be calculated based on $\mathrm{h}_{\mathcal{A}_\mathrm{i}}$ and $\mathcal{S}_\mathrm{i}$:

$$f_{\mathcal{S}_\mathrm{i}} = emb_{\mathcal{S}_\mathrm{i}}(\mathrm{e}(\mathcal{S}_\mathrm{i}), \mathrm{h}_{\mathcal{A}_\mathrm{i}})$$

Here, $emb_{\mathcal{S}_\mathrm{i}}(\cdot)$ represents a fully connected layer, and $\mathcal{S}_\mathrm{i}$ represents the type of the motif.

Finally, the embedding of the i-th molecule is defined as:

$$z_m^{(\mathrm{i})} = \mu(\mathrm{h}_{\mathcal{S}_1}) + \exp(\sum(\mathrm{h}_{\mathcal{S}_1})) \cdot \epsilon$$

$$\epsilon \sim \mathcal{N}(0, \mathrm{I})$$

where $\mu(\mathrm{h}_{\mathcal{S}_1})$, $\sum(\mathrm{h}_{\mathcal{S}_1})$ are both fully connected layers representing the mean and log variance, respectively, and $\mathcal{S}_1$ represents the root motif, which is also the first motif for decoding.

Contrastive Learning-based multimodal matching

In Step 2, to better capture the mutual information between gene and small molecule modalities and enhance the controllability of the generation model, we adopted contrastive learning [23] [24] for multi-modal matching. During the training process, we encountered two challenges: 1. How to obtain gene expression signatures

encoding from the gene embeddings of the experimental and control groups. 2. How to address model collapse.

For the first question, we tried two approaches: direct concatenation into a larger vector (1936 x 512) and taking the difference while keeping the size as (968 x 512). Both methods yielded similar matching results. Due to the quadratic increase in computational complexity with vector length [25] [26], we chose the second representation.

To address the second issue, we propose auxiliary-guided matching, which is explained in detail below.

Auxiliary-guided matching

During the training process of step 2, we encountered model collapse, where different input vectors would result in nearly identical output vectors. We hypothesized that this was due to the model struggling to effectively differentiate between positive and negative sample pairs in the early stages of training, leading to weak gradient signals and a local optimum. Additionally, we observed two phenomena: 1. If we used cross-entropy loss, the model would not collapse, but due to the large number of small molecule types (13757), the model would not converge. 2. The gene encoder was easily prone to collapse. Based on our analysis and observations, we implemented the following steps to assist the model [23] in learning.

Hierarchical clustering of small molecules: Hierarchical clustering merges data into clusters step by step by calculating the similarity between them, until a complete tree-like structure is formed. We use hierarchical clustering to cluster the small

molecule embeddings into $C = \{c_1, c_2, c_3\}$ clusters. The detailed process is shown in **Algorithm 1**.

Auxiliary-guided gene encoding: The output of the final layer in our gene encoder is compared against small molecule vectors for contrastive learning. The optimization objective is to minimize the InfoNCE loss function[27]:

$$\mathcal{L}_{InfoNCE} = -\frac{1}{2} E_{z_m, z_g}\left[log \frac{exp\left(E\left(z_m, z_g\right)\right)}{exp\left(E\left(z_m, z_g\right)\right) + \sum_{g'} exp\left(E\left(z_m, z_g{}'\right)\right)}\right]$$

$$-\frac{1}{2} E_{z_m, z_g}\left[log \frac{exp\left(E\left(z_m, z_g\right)\right)}{exp\left(E\left(z_m, z_g\right)\right) + \sum_{m'} exp\left(E\left(z_m{}', z_g\right)\right)}\right]$$

Where m and g is $z_m{}'$ and $z_g{}'$ are unpaired molecule-gene negative sample pairs in a batch, $E(.,.)$ is an energy (scalar) function defined on the two modalities, and we use vector dot product for representation.

However, contrastive learning often faces model collapse, a common phenomenon in its results. The InfoNCE loss aims to minimize the distance between positive samples and maximize the distance between negative samples. Gradient analysis [28] reveals that the gradients related to negative samples are proportional to their similarity to the anchor, demonstrating that it is a hardness-aware loss function. Therefore, exploring difficult negative samples is crucial for the effectiveness of contrastive learning. Unfortunately, in the field of life sciences, the importance of difficult negative samples is often overlooked due to the lack of ground truth for cross-modal relationships. GexMolGen defines drugs with high similarity as hard negative samples for each other's gene expression profiles. Hierarchical clustering is employed to group molecules into clusters, and the model's learning is guided by the

cross-entropy loss function at a coarse resolution. Assuming that the small molecules have been clustered into $C = \{c_1, c_2, c_3\}$ classes through hierarchical clustering, we require the gene encoder to classify the corresponding perturbed drugs based on only two states of gene expression profiles. The quality of classification is measured by the cross-entropy loss function:

$$\mathcal{L}_{CE_c} = -\sum_{i=1}^{N} y_{c,i} log\left(p_{c,i}\right)$$

where $y_{c,i}$ and $p_{c,i}$ denote the labels and predicted labels, respectively, for the c-th cluster of small molecules based on gene embeddings.

During the learning process, our gene encoder optimizes two losses: InfoNCE loss $\mathcal{L}_{InfoNCE}$ for contrastive learning and cross-entropy loss $\mathcal{L}_{CE_c}$ for the auxiliary guidance in the classification task.

$$\mathcal{L}_{contrast} = \mathcal{L}_{InfoNCE} + \sum_{i=1}^{C} \mathcal{L}_{CE_c}$$

The values of c_1, c_2, and c_3 (10, 100, and 1000, respectively) are chosen based on the following considerations: providing adequate tolerance for similar drugs by limiting task categories, ensuring consistency in guided labels with the sufficiency of identical fine-grained labels for identical coarse-grained labels, and prioritizing simplicity by exponentially increasing the category count in each subsequent layer.

Facilitator-based autoregressive model

Assuming our dataset consists of sample pairs (m, g) where m represents a small molecule and g represents its corresponding gene expression profile, we model GexMolGen, denoted as $P(m|g)$, using a two-step process involving modality

facilitation and model generation [29]. Let z_m and z_g represent the embeddings of the small molecule and gene, respectively. Thus, $P(m|g)$ can be decomposed as

$$P(m|g) = P(m, z_m|g) = P(m|z_m, g)P(z_m|g).$$

The first equation holds because z_m is a deterministic function of the small molecule dataset m, and the second equation holds due to the chain rule [30] [19].

When the small molecule decoder $P(m|z_m, g)$ has been determined [18], the primary focus is on optimizing the facilitator $P(z_m|g)$. For a straightforward approach multi-layer perceptrons (MLPs), it obtained unsatisfactory results with almost 30% higher loss. Additionally, it did not facilitate further exploration of the relationship between small molecule structure and specific gene expression signatures. Taking these limitations into account, we adopt an autoregressive model for modality transformation [31] [32] [33].

When choosing a loss function for this module, cross-entropy is often considered a better choice than mean squared error (MSE) because it handles probability distributions better and its derivative directly reflects differences between probability distributions, facilitating gradient computation and observation of significant changes. However, when z_m is a continuous value, the cross-entropy loss function cannot be used directly. To address this, we follow the approach of DALL·E2 [30], and reduce the dimensionality of z_m from 64 to $L = 43$ using PCA while retaining almost all the information (reconstruction MSE error below 1%). Subsequently, we discretize the L-dimensional values into 1024 bins based on their magnitudes.

Figure 1 provides the details of the facilitator. Its input comprises 968 genes, a

special symbol [start], and L small molecule tokens, and it sequentially outputs predicted values. The loss function is defined as:

$$\mathcal{L}_p = -\sum_{i=0}^{L} t_i log(p_i(x^{(i)}|x^{(0:i-1)})),$$

where t_i and $p_i(x^{(i)}|x^{(0:i-1)})$ represent the i-th true token and predicted token, respectively, of the small molecule embeddings.

Beam search-based generation optimization

Beam search is a widely utilized search algorithm in the field of sequence generation tasks [34] [35]. It leverages the concept of dynamic programming by maintaining a set of candidate sequences, referred to as the beam, with a size of K in order to retain promising outcomes. This algorithm effectively controls the search space while enhancing the quality and accuracy of the generated results. Specifically, the beam search algorithm follows the subsequent steps: 1. Initialization of the beam. 2. For each time step, the probability of each candidate token as the subsequent token is calculated based on the current sequence. 3. The tokens are sorted according to their probabilities, and the top K tokens are selected to form a new sequence. Steps 2 and 3 are repeated until a complete sequence of small molecule tokens is generated. In our model, K is set to 20. 4. These K discrete small molecule tokens are transformed back into the original z_m through the utilization of PCA and are subsequently fed into the small molecule decoder [18] denoted as $P(m|z_m, g)$. This decoding process generates 20 types of hit-like small molecules' SMILES and molecular graphs.

Molecular

Result and Discussion

To demonstrate the advantages of GexMolGen, we conducted two experiments to verify the generalization ability and zero-shot ability of GexMolGen.

Generating Complex Hit-like Molecules from Compound-Induced Gene Expression

In this section, we train our model to generate drug-like small molecules from compound-induced gene expression data. We input 708 test samples into the model and obtain the molecular graphs and SMILES representations of the generated molecules. **Figure 2** illustrates a comparison between the generated molecules by GexMolGen and the actual active molecules (more details can be found in **Appendix Table 1**). The results demonstrate that our model GexMolGen successfully generates the majority of the scaffolds of the reference molecules, handles long-chain molecules (e.g., **Figure 2(4)(10)**), and effectively deals with complex structures such as cyclic structures (**Figure 2(1)**) and cross structures (**Figure 2(7)**).

To better assess the generative capabilities of GexMolGen, we compare it with two baseline methods: calculating the cosine similarity and Euclidean distance between the gene expression signatures of the test set and the training set to find the most similar hit-like small molecules [8]. To evaluate the impact of gene expression signatures on the generation of small molecules, we add a random group where the model directly samples from the molecule latent space. We use three measurement metrics: Tanimoto similarity based on Morgan fingerprints (radius=3, 1024 bits) to

capture global molecular features, Tanimoto similarity based on MACCS keys, and Fraggle similarity to capture local structural similarities..

Figure 3 shows the distribution of the metrics for our model and three other methods. Detailed data can be found in **Appendix Table 1-3.** The results generated by our model significantly outperform the other three methods (p-value < 0.001 from a one-sided Mann-Whitney U-test for the four different molecular representations).

It is worth noting that in the distribution of Tanimoto similarity based on Morgan fingerprints (**Figure 3(b)**), the trend of the results generated by our model is similar to the results obtained from cosine similarity search. We speculate that this similarity is due to the use of cosine distance in the contrastive learning process to calculate the similarity between the two modal vectors. Therefore, these two methods exhibit a similar trend in the distribution of metrics that focus on the global similarity. However, in the other two metrics (**Figure 3(a)** and **(c)**), our model outperforms the cosine similarity search method, with a mean Fraggle similarity of 0.57 and a mean MACCS similarity of 0.70. This indicates that our model generates more molecular fragments and feature keys that are similar to the target molecules. Interestingly, for all three metrics, the Euclidean distance search method performs worse than the cosine similarity method. This may be due to the curse of dimensionality, where the Euclidean distance becomes less sensitive to differences in gene expression profiles as the dimension increases. The randomly generated method performs the worst, as it occasionally generates molecules with high similarity scores (e.g., **Figure 3(b)**: 1.19% with a similarity > 0.29), but also produces more molecules with very low similarity

scores (e.g., **Figure 3(b)**: 26.27% with similarity < 0.08).

Generating Complete Hit-like Molecule Scaffolds from Knock-out Gene Expression

To demonstrate the ability of GexMolGen for generating hit-like small molecule skeletons from knock-out gene expression data, we compare them with known target inhibitors. The underlying hypothesis is that when a protein is knocked out, it leads to a gene expression signature resembling that observation when the same target protein is inhibited by a potent and selective inhibitor. This assumption is based on the notion that both scenarios hypothetically trigger similar cellular adjustments, resulting in equivalent changes in gene expression [8]. This experiment primarily aims to validate the zero-shot capability of our model GexMolGen.

To ensure a fair comparison, we initially gather 10 known inhibitors of human genes, which are the same as those used in the WGAN [8] experiments, from the ExCape dataset [38]. These genes include AKT1, AKT2, AURKB, CTSK, EGFR, HDAC1, MTOR, PIK3CA, SMAD3, and TP53. Our selection criteria include a Pic50 value greater than or equal to 5, InChi-key not appearing in the training set, and compliance with valence bond rules. In the end, we obtain 2,153, 1,293, 2,122, 1,609, 4,244, 1,939, 2,858, 2,292, 23,758, and 12,860 known inhibitors for AKT1, AKT2, AURKB, CTSK, EGFR, HDAC1, MTOR, PIK3CA, SMAD3, and TP53, respectively. Then, we extract 100 experimental and control group transcriptome profiles for each of these 10 genes from the CRISPR knock-out perturbation data in the L1000 dataset.

We employ the three evaluation metrics: Tanimoto similarity based on Morgan

fingerprints, Tanimoto similarity based on MACCS keys, and Fraggle similarity. We also employ the same three comparison methods as before: cosine similarity and Euclidean distance search to find the most similar small molecules based on gene expression signatures, and random generation using the small molecule decoder. Each method independently identifies 100 candidate hit-like compounds based on the provided transcriptome. We then record the highest similarity score between each candidate and the known inhibitors as the model's score.

Figure 4 shows the visual comparison between the molecules generated by our model and the known inhibitors based on Morgan fingerprint similarity scores. It can be observed that our model not only generates molecules that are overall highly similar to the known inhibitors (the average similarity score of WGAN[8] is only 0.26, refer to **Appendix Figure 1** and **Table 4** for detailed results) but also generates the complete skeleton parts of the molecules (e.g., **Figure 4(4)(5)(6)**).

Next, we present the experimental results in **Figure 5** using box plots. Overall, our model performs better than the other three methods, except for the Fraggle score in the task of finding MTOR inhibitors, where it is slightly lower than the cosine similarity and Euclidean distance-based methods (mean Fraggle similarity = 0.60). However, interestingly, our model achieves the highest Fraggle score (max Fraggle similarity = 0.96) and performs better than the traditional methods in terms of MACCS fingerprint and Morgan fingerprint scores. We analyze this phenomenon as the training set contains many molecular fragments similar to the MTOR inhibitors, so directly searching the training set using traditional methods can achieve higher

Fraggle scores. However, the molecular fragments in the training set do not necessarily have high similarity to the MTOR inhibitors in terms of molecular fingerprints, resulting in lower MACCS fingerprint and Morgan fingerprint scores. In addition, we find that the randomly generated method performs worse due to the diversity of the unconditional generation model, occasionally generating molecules with high similarity scores but also generating a larger number of molecules with very low similarity (e.g., **Figure 5 AKT2**). Conversely, traditional methods are constrained by the similarity of gene expression profiles. While they do not generate molecules with very low scores, they only identify a restricted number of hit-like small molecules for certain genes, resulting in many box plots appearing as lines (e.g., **Figure 5 CTSK**). In contrast, our generative model, achieves a trade-off between diversity and accuracy by conditioning on gene embeddings.

Furthermore, we have organized the performance of other methods on similar tasks. Although the tested genes may vary, it can be generally observed that our model achieves above-average results. Moreover, our model possesses inherent advantages such as high effectiveness and strong scalability.

Discussion

In this study, we design a model called GexMolGen to generate candidate hit-like molecules based on gene expression signatures. We apply gene transcriptional profiling-based foundation models for downstream multimodal molecule generation. Unlike the SMILES-based multimodal fusion used in previous studies [8] [9], we use

more informative two-dimensional molecular representations [18]. In the training process, we use assisted guidance to alleviate the collapse problem of the model and improve the alignment of modalities. By using a self-attention-based autoregressive modality transformer and beam search optimization, our model achieves good results in generating gene inhibitors. Additionally, we explore a new gene model mask strategy to prevent changes in the biological significance of gene spectra caused by masking.

However, our model has some limitations. First, we do not exhaustively test all available gene transcriptional profiling-based models. Because some models are not open-source [14] [36], directly converting certain foundation models into gene expression signatures is challenging [12] [11]. Taking various factors into consideration, we select scGPT [8], but this does not imply that other large-scale models perform worse. Second, for the sake of simplicity, we choose a unified model for molecular encoding and decoding [18]. However, we find that advancements in the field of molecular representation [39] [22] [40] [41] may outpace the encoding modules in molecular generators [42] [35]. Therefore, the molecular encoding part can be updated with other excellent works in the future.

In the future, we expect to do two things regarding GexMolGen. First, we want to explore the interpretability between molecular fragments and specific gene expression signatures, which is the original intention behind using autoregressive models and discretizing molecular embedding. Second, addressing the aforementioned limitations, we intend to investigate the potential of separating the molecular encoder and decoder

to improve cross-modal accuracy.

Reference

[1] F. Vincent, A. Nueda, J. Lee, M. Schenone, M. Prunotto, and M. Mercola, "Phenotypic drug discovery: recent successes, lessons learned and new directions," *Nat Rev Drug Discov*, vol. 21, no. 12, Art. no. 12, Dec. 2022, doi: 10.1038/s41573-022-00472-w.

[2] A. Musa *et al.*, "A review of connectivity map and computational approaches in pharmacogenomics," *Briefings in Bioinformatics*, vol. 18, no. 5, p. 903, Sep. 2017, doi: 10.1093/bib/bbx023.

[3] J. Li, R. R. Selvaraju, A. D. Gotmare, S. Joty, C. Xiong, and S. Hoi, "Align before Fuse: Vision and Language Representation Learning with Momentum Distillation." arXiv, Oct. 07, 2021. doi: 10.48550/arXiv.2107.07651.

[4] J. Li, D. Li, C. Xiong, and S. Hoi, "BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation." arXiv, Feb. 15, 2022. doi: 10.48550/arXiv.2201.12086.

[5] C. Edwards, C. Zhai, and H. Ji, "Text2Mol: Cross-Modal Molecule Retrieval with Natural Language Queries," in *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, 2021, pp. 595–607. doi: 10.18653/v1/2021.emnlp-main.47.

[6] S. Liu *et al.*, "A Text-guided Protein Design Framework." arXiv, Feb. 09, 2023.

421 Accessed: Jun. 29, 2023. [Online]. Available: http://arxiv.org/abs/2302.04611

422 [7] J. Zhu *et al.*, “Prediction of drug efficacy from transcriptional profiles with deep
423 learning,” *Nat Biotechnol*, vol. 39, no. 11, Art. no. 11, Nov. 2021, doi:
424 10.1038/s41587-021-00946-z.

425 [8] O. Méndez-Lucio, B. Baillif, D.-A. Clevert, D. Rouquié, and J. Wichard, “De novo
426 generation of hit-like molecules from gene expression signatures using artificial
427 intelligence,” *Nat Commun*, vol. 11, no. 1, Art. no. 1, Jan. 2020, doi:
428 10.1038/s41467-019-13807-w.

429 [9] D. Das, B. Chakrabarty, R. Srinivasan, and A. Roy, “Gex2SGen: Designing Drug-like
430 Molecules from Desired Gene Expression Signatures,” *J. Chem. Inf. Model.*, vol. 63,
431 no. 7, pp. 1882–1893, Apr. 2023, doi: 10.1021/acs.jcim.2c01301.

432 [10] C. V. Theodoris *et al.*, “Transfer learning enables predictions in network biology,”
433 *Nature*, vol. 618, no. 7965, Art. no. 7965, Jun. 2023, doi:
434 10.1038/s41586-023-06139-9.

435 [11] F. Yang *et al.*, “scBERT as a large-scale pretrained deep language model for cell
436 type annotation of single-cell RNA-seq data,” *Nat Mach Intell*, vol. 4, no. 10, Art. no.
437 10, Oct. 2022, doi: 10.1038/s42256-022-00534-z.

438 [12] J. Chen, H. Xu, W. Tao, Z. Chen, Y. Zhao, and J.-D. J. Han, “Transformer for one
439 stop interpretable cell type annotation,” *Nat Commun*, vol. 14, no. 1, Art. no. 1, Jan.
440 2023, doi: 10.1038/s41467-023-35923-4.

441 [13] H. Cui, C. Wang, H. Maan, K. Pang, F. Luo, and B. Wang, “scGPT: Towards
442 Building a Foundation Model for Single-Cell Multi-omics Using Generative AI.”

443 bioRxiv, p. 2023.04.30.538439, Jul. 02, 2023. doi: 10.1101/2023.04.30.538439.

444 [14] M. Hao *et al.*, “Large Scale Foundation Model on Single-cell Transcriptomics.”
445 bioRxiv, p. 2023.05.29.542705, Jun. 21, 2023. doi: 10.1101/2023.05.29.542705.

446 [15] G. Heimberg *et al.*, “Scalable querying of human cell atlases via a foundational
447 model reveals commonalities across fibrosis-associated macrophages.” bioRxiv, p.
448 2023.07.18.549537, Jul. 19, 2023. doi: 10.1101/2023.07.18.549537.

449 [16] X. Yang *et al.*, “GeneCompass: Deciphering Universal Gene Regulatory
450 Mechanisms with Knowledge-Informed Cross-Species Foundation Model.” bioRxiv,
451 p. 2023.09.26.559542, Sep. 28, 2023. doi: 10.1101/2023.09.26.559542.

452 [17] A. Subramanian *et al.*, “A Next Generation Connectivity Map: L1000 Platform and
453 the First 1,000,000 Profiles,” *Cell*, vol. 171, no. 6, pp. 1437-1452.e17, Nov. 2017, doi:
454 10.1016/j.cell.2017.10.049.

455 [18] W. Jin, R. Barzilay, and T. Jaakkola, “Hierarchical Generation of Molecular Graphs
456 using Structural Motifs.” arXiv, Apr. 18, 2020. Accessed: Jan. 28, 2023. [Online].
457 Available: http://arxiv.org/abs/2002.03230

458 [19] A. Ramesh *et al.*, “Zero-Shot Text-to-Image Generation.” arXiv, Feb. 26, 2021.
459 Accessed: Mar. 17, 2023. [Online]. Available: http://arxiv.org/abs/2102.12092

460 [20] A. Sanchez-Fernandez, E. Rumetshofer, S. Hochreiter, and G. Klambauer,
461 “Contrastive learning of image- and structure-based representations in drug
462 discovery,” presented at the ICLR2022 Machine Learning for Drug Discovery, Apr.
463 2022. Accessed: Aug. 26, 2023. [Online]. Available:
464 https://openreview.net/forum?id=OdXKRtg1OG

[21] H. Dai, B. Dai, and L. Song, “Discriminative Embeddings of Latent Variable Models for Structured Data.” arXiv, Jan. 10, 2020. Accessed: Nov. 02, 2023. [Online]. Available: http://arxiv.org/abs/1603.05629

[22] M. Fey, J.-G. Yuen, and F. Weichert, “Hierarchical Inter-Message Passing for Learning on Molecular Graphs.” arXiv, Jun. 22, 2020. doi: 10.48550/arXiv.2006.12179.

[23] P. Khosla *et al.*, “Supervised Contrastive Learning.” arXiv, Mar. 10, 2021. Accessed: Jul. 05, 2023. [Online]. Available: http://arxiv.org/abs/2004.11362

[24] Y. Tian, D. Krishnan, and P. Isola, “Contrastive Multiview Coding,” in *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XI*, Berlin, Heidelberg: Springer-Verlag, Aug. 2020, pp. 776–794. doi: 10.1007/978-3-030-58621-8_45.

[25] A. Vaswani *et al.*, “Attention is all you need,” in *Proceedings of the 31st International Conference on Neural Information Processing Systems*, in NIPS’17. Red Hook, NY, USA: Curran Associates Inc., Dec. 2017, pp. 6000–6010.

[26] T. Dao, D. Fu, S. Ermon, A. Rudra, and C. Ré, “FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness,” *Advances in Neural Information Processing Systems*, vol. 35, pp. 16344–16359, Dec. 2022.

[27] A. van den Oord, Y. Li, and O. Vinyals, “Representation Learning with Contrastive Predictive Coding,” arXiv.org. Accessed: Sep. 01, 2023. [Online]. Available: https://arxiv.org/abs/1807.03748v2

[28] F. Wang and H. Liu, “Understanding the Behaviour of Contrastive Loss.” arXiv, Mar.

20, 2021. doi: 10.48550/arXiv.2012.09740.

[29] M. Ding *et al.*, “CogView: Mastering Text-to-Image Generation via Transformers.” arXiv, Nov. 05, 2021. doi: 10.48550/arXiv.2105.13290.

[30] A. Ramesh, P. Dhariwal, A. Nichol, C. Chu, and M. Chen, “Hierarchical Text-Conditional Image Generation with CLIP Latents.” arXiv, Apr. 12, 2022. doi: 10.48550/arXiv.2204.06125.

[31] “Radford 等 - Language Models are Unsupervised Multitask Learner.pdf.” Accessed: Aug. 26, 2023. [Online]. Available: https://cdn.openai.com/better-language-models/language_models_are_unsupervised_multitask_learners.pdf

[32] M. Lewis *et al.*, “BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension.” arXiv, Oct. 29, 2019. doi: 10.48550/arXiv.1910.13461.

[33] C. Raffel *et al.*, “Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer.” arXiv, Jul. 28, 2020. doi: 10.48550/arXiv.1910.10683.

[34] OpenAI, “GPT-4 Technical Report.” arXiv, Mar. 27, 2023. doi: 10.48550/arXiv.2303.08774.

[35] X. Bresson and T. Laurent, “A Two-Step Graph Convolutional Decoder for Molecule Generation.” arXiv, Jun. 15, 2019. doi: 10.48550/arXiv.1906.03412.

[36] J. Gong *et al.*, “xTrimoGene: An Efficient and Scalable Representation Learner for Single-Cell RNA-Seq Data.” bioRxiv, p. 2023.03.24.534055, Mar. 25, 2023. doi: 10.1101/2023.03.24.534055.

[37] E. Nguyen *et al.*, “HyenaDNA: Long-Range Genomic Sequence Modeling at Single Nucleotide Resolution.” arXiv, Jun. 27, 2023. doi: 10.48550/arXiv.2306.15794.

[38] J. Sun *et al.*, “ExCAPE-DB: an integrated large scale dataset facilitating Big Data analysis in chemogenomics,” *Journal of Cheminformatics*, vol. 9, no. 1, p. 17, Mar. 2017, doi: 10.1186/s13321-017-0203-5.

[39] G. Zhou *et al.*, “Uni-Mol: A Universal 3D Molecular Representation Learning Framework.” ChemRxiv, May 26, 2022. doi: 10.26434/chemrxiv-2022-jjm0j.

[40] K. Atz, F. Grisoni, and G. Schneider, “Geometric deep learning on molecular representations,” *Nat Mach Intell*, vol. 3, no. 12, Art. no. 12, Dec. 2021, doi: 10.1038/s42256-021-00418-8.

[41] J. Xia *et al.*, “MOLE-BERT: RETHINKING PRE-TRAINING GRAPH NEURAL NETWORKS FOR MOLECULES,” 2023.

[42] C. Zang and F. Wang, “MoFlow: An Invertible Flow Model for Generating Molecular Graphs,” in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, in KDD ’20. New York, NY, USA: Association for Computing Machinery, Aug. 2020, pp. 617–626. doi: 10.1145/3394486.3403104.

[43] R. Shayakhmetov *et al.*, “Molecular Generation for Desired Transcriptome Changes With Adversarial Autoencoders,” *Frontiers in Pharmacology*, vol. 11, 2020, Accessed: Aug. 27, 2023. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fphar.2020.00269

[44] “TRIOMPHE: Transcriptome-Based Inference and Generation of Molecules with Desired Phenotypes by Machine Learning.” Accessed: Aug. 27, 2023. [Online].

Available: https://pubs.acs.org/doi/epdf/10.1021/acs.jcim.1c00967

[45] J. Born, M. Manica, A. Oskooei, J. Cadow, and M. Rodríguez Martínez, "PaccMannRL: Designing Anticancer Drugs From Transcriptomic Data via Reinforcement Learning," in *Research in Computational Molecular Biology*, R. Schwartz, Ed., in Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, pp. 231–233. doi: 10.1007/978-3-030-45257-5_18.

[46] T. Pereira, M. Abbasi, R. I. Oliveira, R. A. Guedes, J. A. R. Salvador, and J. P. Arrais, "Deep generative model for therapeutic targets using transcriptomic disease-associated data—USP7 case study," *Brief Bioinform*, vol. 23, no. 4, Jul. 2022, doi: 10.1093/bib/bbac270.

Caption

Figure 1: Flowchart of GexMolGen

The generation of hit-like molecules by GexMolGen mainly consists of four steps (refer to the Model Schematic section): 1. Encoding of genes and small molecule data, 2. Matching between gene modalities and small molecule modalities, 3. Transformation from gene modality to small molecule modality, 4. Decoding of small molecule modality. For simplicity, both the encoder and decoder for steps 1 and 4 utilize pre-trained models, namely scGPT[13] (orange section) and hierVAE[18] (blue section) respectively. Due to frequent collapse of the gene encoder during the model matching stage in step 2, our auxiliary learning primarily focuses on the gene encoder (upper orange section), while the small molecule encoder mainly provides guiding

labels through hierarchical clustering (blue arrows). In step 3 (lower green section), we first discretize the small molecules and then reconstruct them in an autoregressive manner.

Figure 2: Example of generating hit-like small molecules from compound-induced gene expression

This figure showcases examples of hit-like molecules generated by GexMolGen from compound-induced gene expression profiles, compared to reference small molecules. The similarity calculation metric is based on the Tanimoto similarity of Morgan fingerprints.

Figure 3: Benchmarking GexMolGen in generating hit-like molecules with similarity-based search and random generation.

(a), (b), and (c) represent the distributions of similarity calculated using MACCS fingerprints, Morgan fingerprints, and Fraggle, respectively. These calculations are done between small molecules generated by GexMolGen, those found by cosine similarity and Euclidean distance search, randomly generated small molecules, and reference small molecules. Each method is represented by a different color.

Figure 4: Example of generating hit-like molecules from knock-out gene expression profiles

This figure presents examples of hit-like small molecules generated by GexMolGen from knock-out gene expression profiles, compared to known inhibitors. The similarity calculation metric is based on the Tanimoto similarity of Morgan fingerprints. For fair comparison, we selected the same ten experimental genes as

WGAN[8], and the results of WGAN can be found in **Supplementary Figure 1**.

Figure 5: Benchmarking GexMolGen in generating potential gene inhibitors with similarity-based search and random generation.

The x-axis represents three similarity calculation metrics: Tanimoto based on MACCS fingerprints, Tanimoto based on Morgan fingerprints, and Fraggle. The y-axis represents ten genes. Different colors represent different methods.

Supplementary Figure 1: Example of generating hit-like molecules from knock-out gene expression profiles by WGAN

We used OCR technology to extract the SMILES of molecules from Figure 2 in the WGAN[8] paper and calculated the similarity results based on the Tanimoto similarity of Morgan fingerprints. For detailed results, please refer to **Supplementary Table 4**.

Table 1: Comparison of GexMolGen and other modern methods in generating potential inhibitors from knock-out transcriptomes.

The first column presents the method names and references, the second column indicates the dataset types used, and the third column specifies the Tanimoto similarity range.

Method	Input Data Type	Tanimoto Similarity
GexMolGen	L1000 dataset	**0.74 to 1.00**
WGAN[8]	L1000 dataset	0.15 to 0.70
Bidirectional Adversarial AE[43]	L1000 dataset	0.3 to 0.48

TRIOMPHE[44]	L1000 dataset	0.29 to 0.68
PaccMann[45]	GDSC/CCLE	0.27 to 0.45
Deep Generative with RL[46]	TCGA project	0.47 to 0.83

Appendix Table 1: Experimental results of four methods for generating hit-like molecules from compound-induced gene expression using Tanomoto similarity metric based on Morgan fingerprints.

Appendix Table 2: Experimental results of four methods for generating hit-like molecules from compound-induced gene expression using Tanomoto similarity metric based on MACCS fingerprints.

Appendix Table 3: Experimental results of four methods for generating hit-like molecules from compound-induced gene expression using Fraggle similarity metric.

Appendix Table 4: SMILES of generated molecules from the WGAN[8] in the OCR-recognized article, along with the SMILES of known inhibitors, and the Tanimoto similarity calculated based on Morgan fingerprints between them.

Algorithm 1: Hierarchical clustering

Input: Small molecule embedding vectors $Z_M = \{z_{m1}, z_{m2}, \ldots, z_{mN}\}$
Distance function $dist(c_1, c_2)$

Output: Categories C of small molecule embedding vectors after hierarchical clustering at each level.

```
For i = 1:n
    c_i = {Z_mi}
end for
C = {c_1, c_2, ..., c_n}
I = n + 1
while C.size > 10 do
    (c_min1, c_min2) = min dist(c_i, c_j) for all c_i, c_j ∈ C
    C = C/{c_min1, c_min2}
    C = C ∪ {(c_min1, c_min2)}
    I = n + 1
end while
return C
```

Figure2

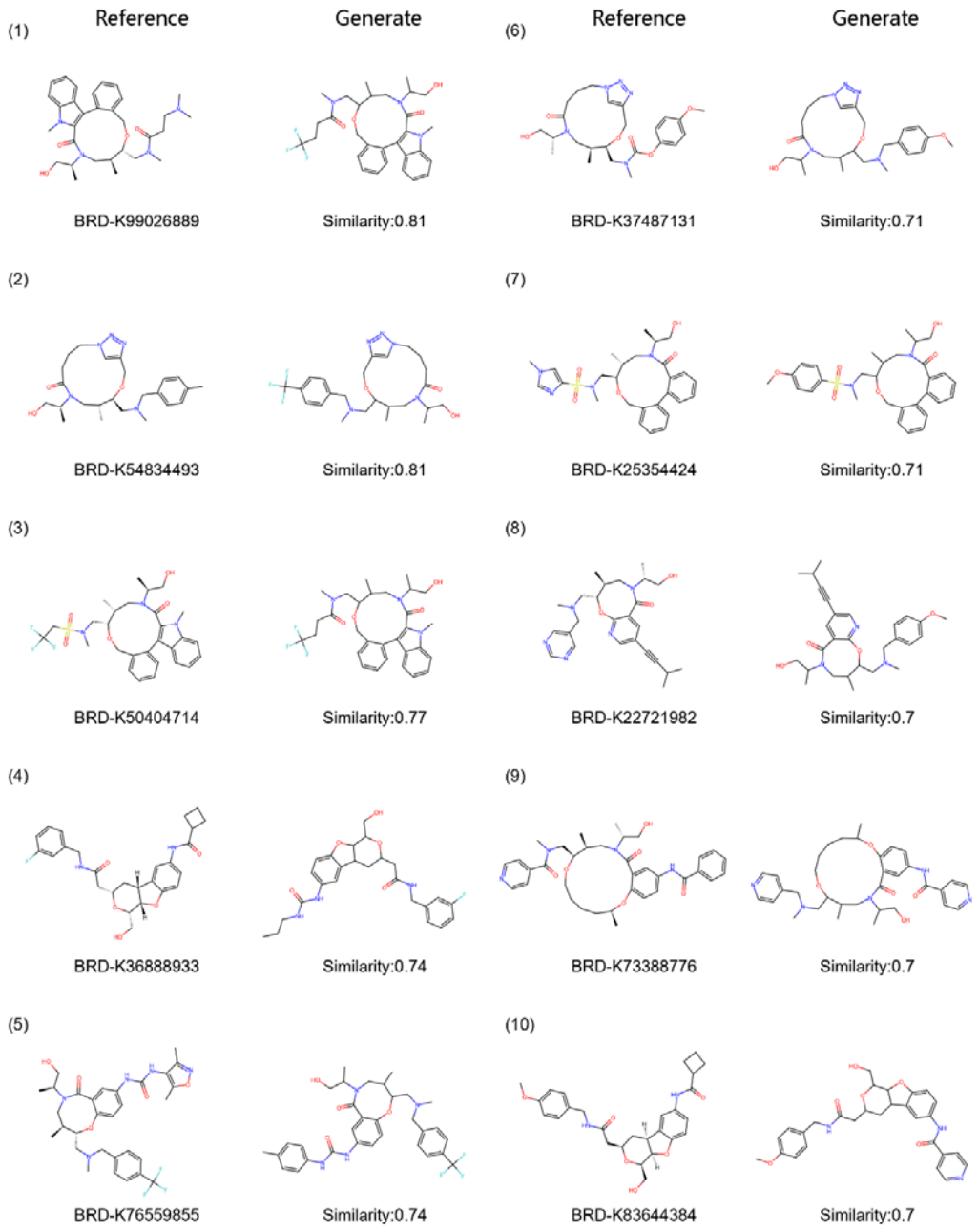

(1)
Reference
Generate
BRD-K99026889
Similarity:0.81
(2)
BRD-K54834493
Similarity:0.81
(3)
BRD-K50404714
Similarity:0.77
(4)
BRD-K36888933
Similarity:0.74
(5)
BRD-K76559855
Similarity:0.74
(6)
Reference
Generate
BRD-K37487131
Similarity:0.71
(7)
BRD-K25354424
Similarity:0.71
(8)
BRD-K22721982
Similarity:0.7
(9)
BRD-K73388776
Similarity:0.7
(10)
BRD-K83644384
Similarity:0.7

Figure3

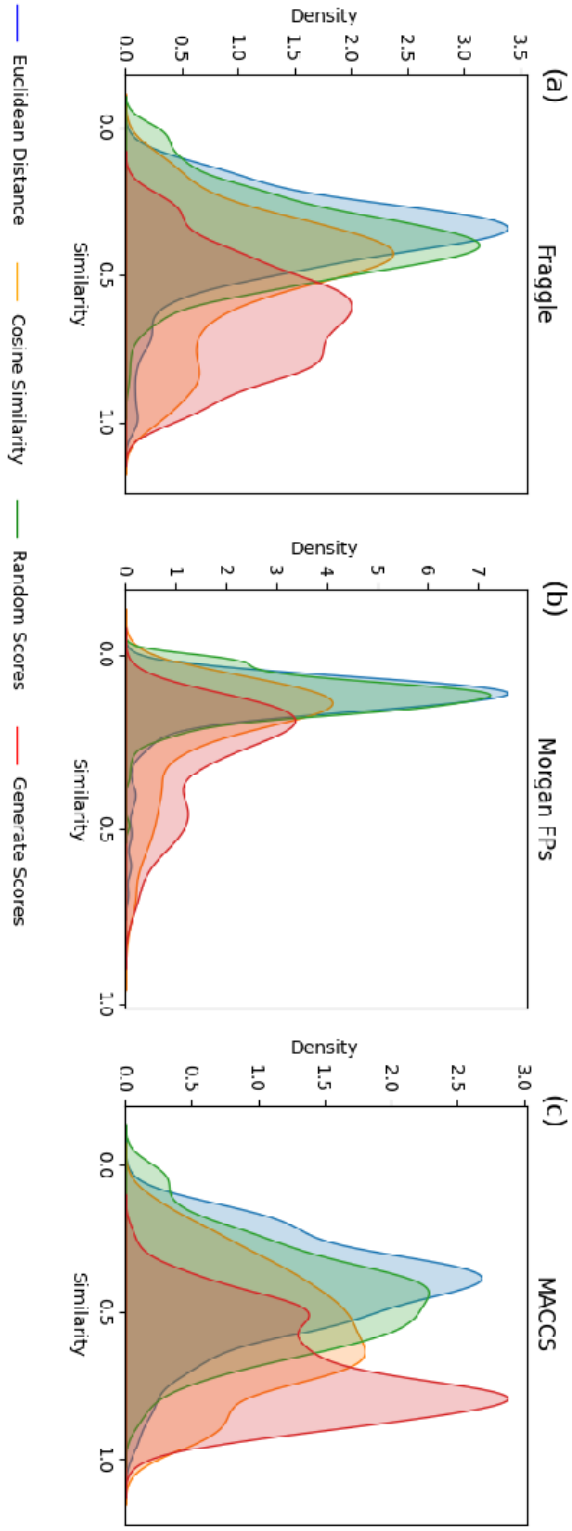

(a)
Fraggle
Density
Similarity
(b)
Morgan FPs
Density
Similarity
(c)
MACCS
Density
Similarity
Euclidean Distance
Cosine Similarity
Random Scores
Generate Scores

Figure4

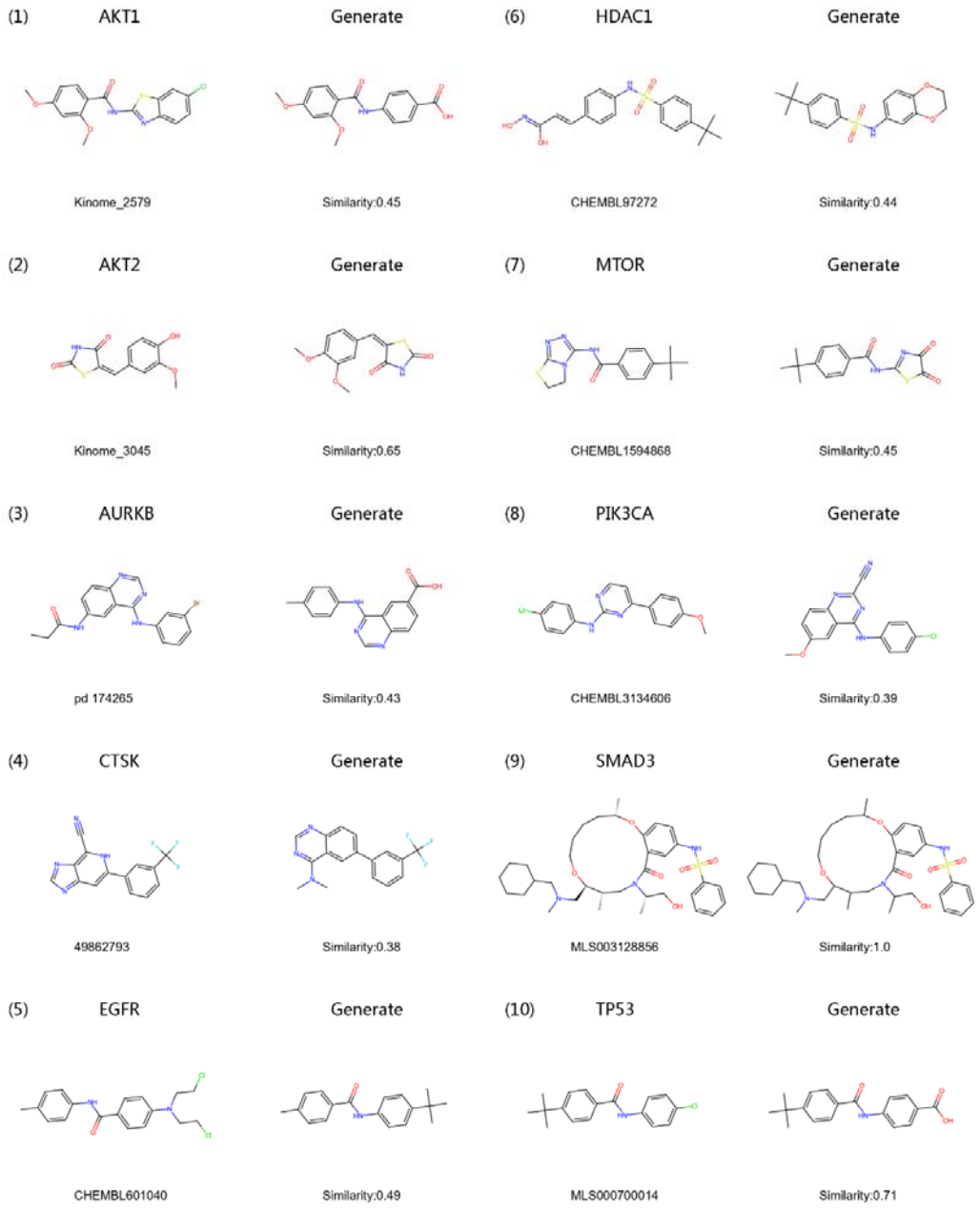
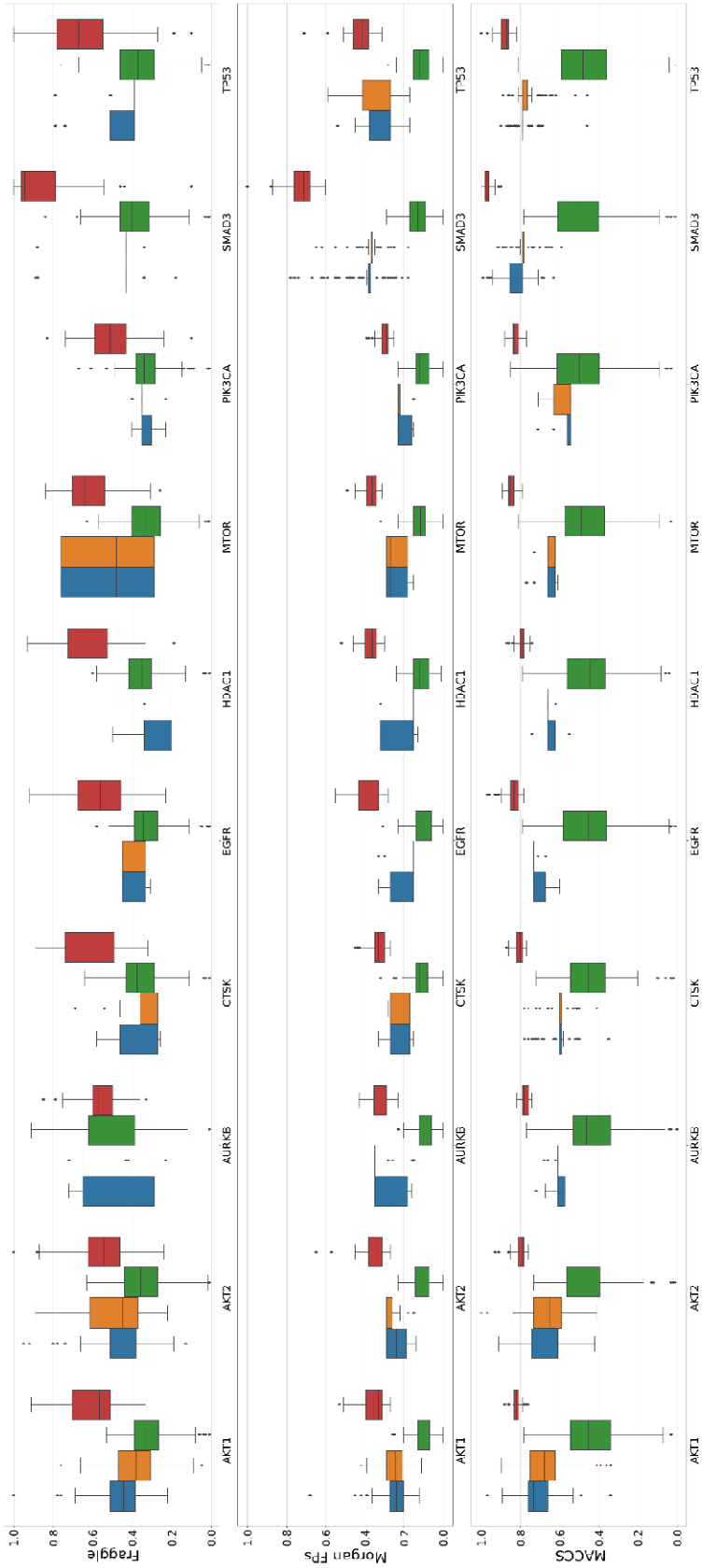

(1) AKT1
Generate
Kinome_2579
Similarity:0.45
(6) HDAC1
Generate
CHEMBL97272
Similarity:0.44
(2) AKT2
Generate
Kinome_3045
Similarity:0.65
(7) MTOR
Generate
CHEMBL1594868
Similarity:0.45
(3) AURKB
Generate
pd 174265
Similarity:0.43
(8) PIK3CA
Generate
CHEMBL3134606
Similarity:0.39
(4) CTSK
Generate
49862793
Similarity:0.38
(9) SMAD3
Generate
MLS003128856
Similarity:1.0
(5) EGFR
Generate
CHEMBL601040
Similarity:0.49
(10) TP53
Generate
MLS000700014
Similarity:0.71

Figure6


Fragg/e
Morgan FPs
MACCS
AKT1
AKT2
AURKB
CTSK
EGFR
HDAC1
MTOR
PIK3CA
SMAD3
TP53
1.0
0.8
0.6
0.4
0.2
0.0