JCIM JOURNAL OF CHEMICAL INFORMATION AND MODELING

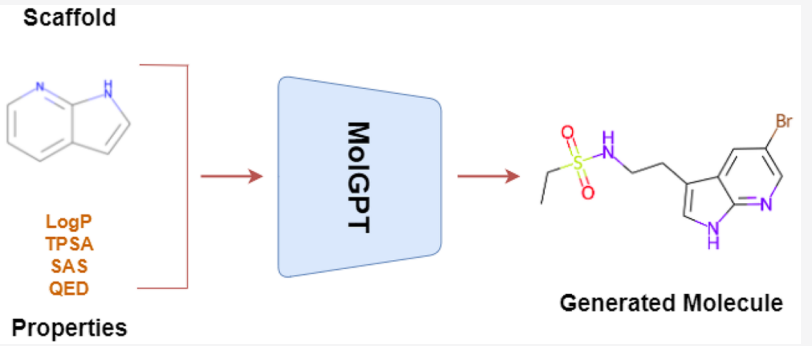
pubs.acs.org/jcim Article

MolGPT: Molecular Generation Using a Transformer-Decoder Model

Viraj Bagal, Rishal Aggarwal, P. K. Vinod, and U. Deva Priyakumar*

Cite This: https://doi.org/10.1021/acs.jcim.1c00600 Read Online

ACCESS | Metrics & More | Article Recommendations | Supporting Information



ABSTRACT: Application of deep learning techniques for *de novo* generation of molecules, termed as inverse molecular design, has been gaining enormous traction in drug design. The representation of molecules in SMILES notation as a string of characters enables the usage of state of the art models in natural language processing, such as Transformers, for molecular design in general. Inspired by generative pre-training (GPT) models that have been shown to be successful in generating meaningful text, we train a transformer-decoder on the next token prediction task using masked self-attention for the generation of druglike molecules in this study. We show that our model, MolGPT, performs on par with other previously proposed modern machine learning frameworks for molecular generation in terms of generating valid, unique, and novel molecules. Furthermore, we demonstrate that the model can be trained conditionally to control multiple properties of the generated molecules. We also show that the model can be used to generate molecules with desired scaffolds as well as desired molecular properties by conditioning the generation on scaffold SMILES strings of desired scaffolds and property values. Using saliency maps, we highlight the interpretability of the generative process of the model.

INTRODUCTION

It has been postulated that the total number of potential drug like candidates range from 10^{23} to 10^{60} molecules,[1] of which only about 10^{8} molecules have been synthesized.[2] Since it is difficult to screen a practically infinite chemical space, and there is a huge disparity between synthesized and potential molecules, generative models are used to model a distribution of molecules for the purpose of sampling molecules that have desirable properties. Deep generative models have made great strides in modeling data distributions in general data domains such as Computer Vision[3,4] and Natural Language Processing (NLP).[5,6] Such methods have also been adopted to model molecular distributions.[7,8] Such models learn probability distributions over a large set of molecules and therefore are able to generate novel molecules by sampling from these distributions.[7,9] The rapid adoption of the deep generative model has also led to the development of benchmark data sets such as the Molecular Sets (MOSES)[10] and GuacaMol[9] data sets.

The representation of molecules in the Simplified Molecular Input Line Entry System (SMILES)[11] notation as a string of characters enables the usage of modern NLP deep learning models for their computation.[12] Some of the earliest deep learning architectures for molecular generation involved the usage of Recurrent Neural Networks (RNNs) on molecular SMILES.[13,14] Such models have also previously been trained on a large corpus of molecules and then focused through the usage of reinforcement learning[15,16] or transfer learning[13] to generate molecules of desirable properties and activity.

Auto-Encoder variants such as the Variational Auto-Encoder (VAE)[17−21] and Adversarial Auto-Encoder (AAE)[22−25] have also been employed for molecular generation. These models contain an encoder that encodes molecules to a latent vector representation and a decoder that maps latent vectors back to molecules. Molecules can then be generated by sampling from these latent spaces. Randomization of SMILES strings[26−28] have also been employed in such models as a data augmentation strategy. Junction Tree VAE (JT-VAE),[20] on the other hand, is an alternate solution for molecular

Special Issue: From Reaction Informatics to Chemical Space

Received: May 26, 2021

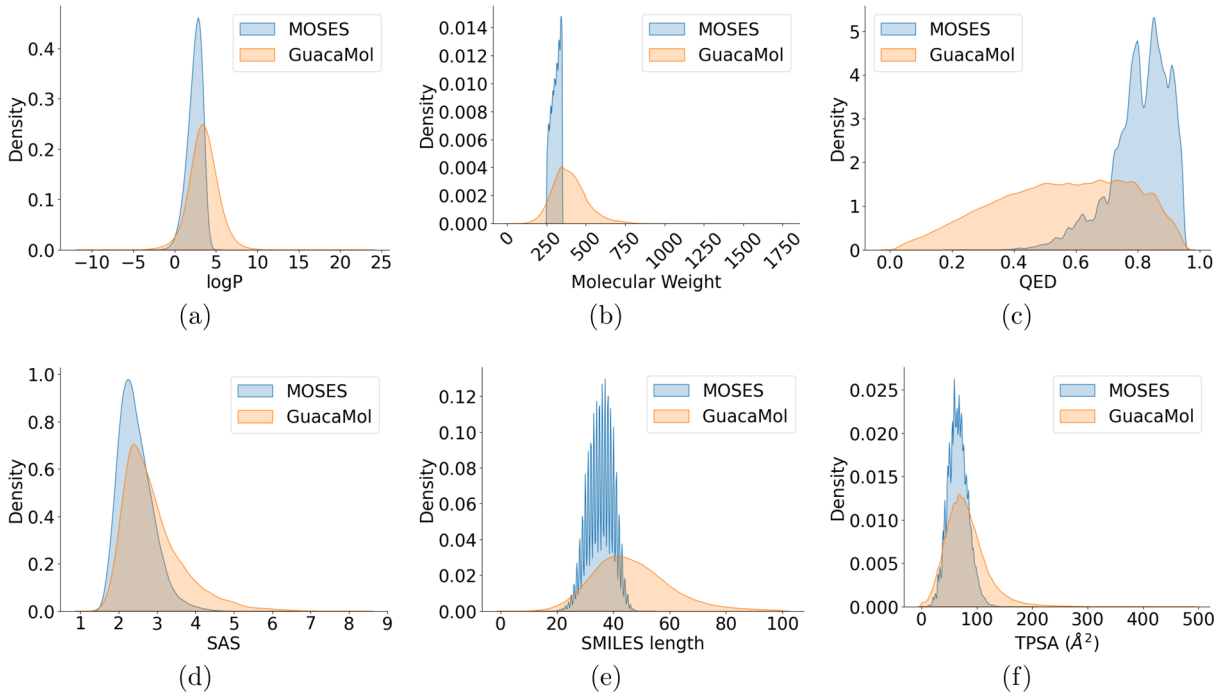


Figure 1. Probability distributions of properties (log P, molecular weight, QED, SAS, TPSA, and SMILES length) of molecules in the MOSES and GuacaMol data sets.

generation that represents molecules as graph tree structures. JT-VAE also ensures 100% validity of generated molecules by maintaining a vocabulary of molecular components that can be added at each junction of the molecule tree. Conditional Variational Auto-Encoders have also been used to generate molecules with desired properties.[29]

Generative Adversarial Networks (GANs) have also gained traction for molecular design.[30−34] This is mainly because of their ability to generate highly realistic content.[4] GANs are composed of generators and discriminators that work in opposition of each other. While the generator tries to generate realistic content, the discriminator tries to distinguish between generated and real content. ORGAN[31] was the first usage of GANs for molecular generation. RANC[34] introduced reinforcement learning alongside a GAN loss to generate molecules of desirable properties. LatentGAN[30] is a more recent method that uses latent vectors as input and outputs. These latent vectors are mapped to molecules by the decoder of a pretrained autoencoder. This ensures that the model can work with latent representations and does not have to handle SMILES syntax. Most of these methods have been benchmarked on either the MOSES[10] or the GuacaMol[9] data set for easy comparison.

Often, methods use Bayesian optimization,[35,36] reinforcement learning,[15,34] or other optimization methods[37,38] to generate molecules exhibiting desirable properties. MolCycleGAN[39] is a generative model that utilizes the JT-VAE architecture and applies the CycleGAN[40] loss to generate molecules of required properties using given molecule templates. Only a few methods employ conditional generation based on user defined property values. Conditional RNNs,[13,41] Deep Learning Enabled Inorganic Material Generator (DING),[29] and Conditional Adversarially Regularized Autoencoder (CARAE)[25] are three such methods that sample molecules based on exact values. RNNs have also been previously used to generate molecules based on given scaffolds.[42] A graph based method has been designed that ensures the presence of desired scaffolds while generating molecules with exact property values.[43]

A novel NLP architecture called the Transformer[5] has shown state-of-the-art performance in language translation tasks. Transformers consist of encoder and decoder modules. The encoder module gains context from all the input tokens through self-attention mechanisms. The decoder module gains context from both the encoder as well as previously generated tokens by attention. Using this context, the decoder is able to predict the next token. The decoder module has also been previously used independently for language modeling tasks and is known as the Generative Pre-Training Transformer model (GPT).[44] The GPT model has been shown to develop better language embeddings[44] that model longer-distance connections. Due to this, the embeddings have shown top performance when used for multiple language modeling tasks such as natural language inference, question answering, sentence similarity, and classification.[45]

To yield the added benefits of this architecture, we train a GPT model, named MolGPT, to predict a sequence of SMILES tokens for molecular generation. To the best of our knowledge, this is the first work that has used the GPT architecture for molecular generation. For this, we use a regular expression (later referred to as a SMILES tokenizer) that breaks the SMILES strings into a set of relevant tokens which are used to train the model. Since predicted tokens are a result of attention applied to all previously generated tokens, we believe that the model easily learns the SMILES grammar and, therefore, can focus on higher level understanding of molecular properties. To this end, we also train our models conditionally to explicitly learn certain molecular properties. The model displays performance that is on par with other methods benchmarked on the MOSES and Guacamol data sets. Furthermore, we show that MolGPT controls user specified molecular properties and scaffolds with good accuracy, leading

https://doi.org/10.1021/acs.jcim.1c00600
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

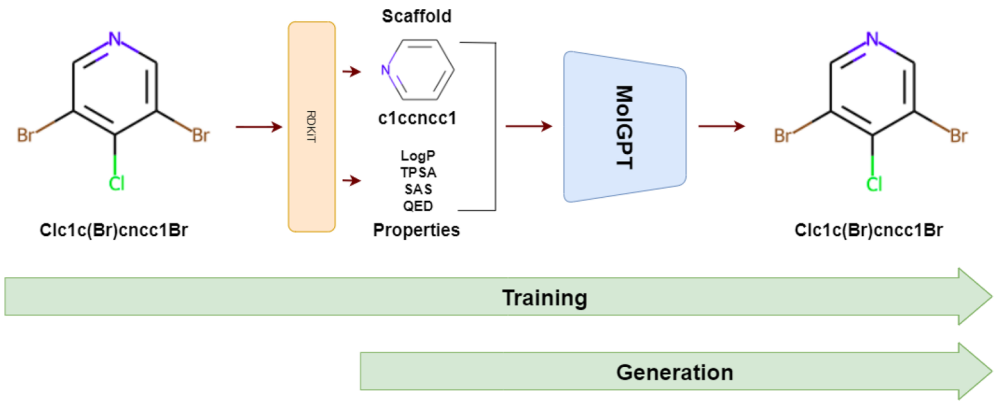


Figure 2. Pipeline for training and generation using the MolGPT model.

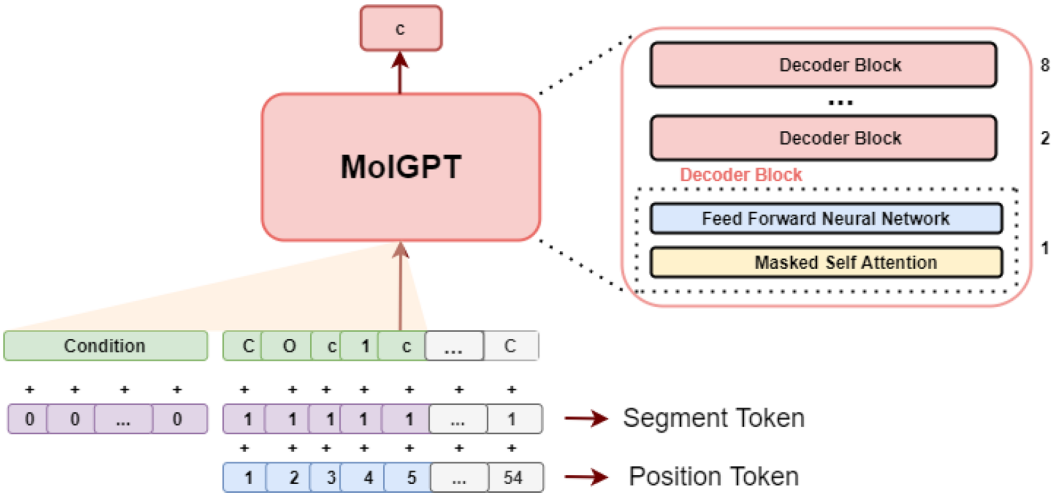


Figure 3. MolGPT model architecture.

to our conclusion that it learns a good representation of the chemical space.

METHODS

In this section, we first present the data sets used for all the experiments. We discuss the properties used for conditional generation. This is then followed by the overview of the proposed model. A schematic of the training and generation pipeline is shown in this section. Finally, the details of the experiments and the metrics used for the evaluation of different models are provided.

Data Sets. In this work, we used two benchmark data sets, MOSES and GuacaMol, for training and evaluation of our model. MOSES is a data set composed of 1.9 million clean lead-like molecules from the Zinc data set[46] with molecular weight ranging from 250 to 350 Da, number of rotatable bonds lower than 7, and XlogP below 3.5. GuacaMol on the other hand is a subset of the ChEMBL 24[47] database that contains 1.6 million molecules. We used the RDkit toolkit[48] to calculate molecular properties and to extract Bemis−Murcko scaffolds.[49] The MOSES data set was created mainly to represent lead like molecules and therefore has a distribution of molecules with ideal druglike properties. However, to test the models' control on conditional generation, we prefer the larger distribution of property values available in the Guacamol data set as can be seen in Figure 1. This is preferred so that we can test the model's ability to generate molecules having very different property values. Therefore, we use the GuacaMol data set to test property conditional generation. The MOSES data set also provides a test set of scaffolds which we use to evaluate scaffold and property conditional generation.

The models were trained to learn some properties of the molecules for controlled generation and optimization. The properties used are the following:

- **logP**: The logarithm of the partition coefficient. The partition coefficient compares the solubilities of the solute in two immiscible solvents at equilibrium. If one of the solvents is water and the other is a nonpolar solvent, then logP is a measure of hydrophobicity.
- **Synthetic Accessibility score (SAS[50])**: Measurement of the difficulty of synthesizing a compound. It is a score between 1 (easy to make) and 10 (very difficult to make).
- **Topological Polar Surface Area (TPSA)**: The sum of surface area over all polar atoms. It measures the drug's ability to permeate cell membranes. Molecules with a TPSA greater than 140 $Å^2$ tend to be poor in permeating cell membranes.

Quantitative Estimate of Drug-likeness (QED[51]): This quantifies drug-likeness by taking into account the main molecular properties. It ranges from 0 (all properties unfavorable) to 1 (all properties favorable).

Model Overview. The model schematic of MolGPT for training and generation is given in Figure 2. For non-conditioned training, molecular SMILES are first tokenized using a SMILES tokenizer, and the model is then trained on

https://doi.org/10.1021/acs.jcim.1c00600
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

the next token prediction task. For property conditioned and scaffold conditioned training, we extract molecular properties and scaffolds from molecules using RDkit[48] and pass them as conditions alongside the molecular SMILES. For generation, we feed the model a start token and the model then sequentially predicts the next token, thus generating a molecule. The start token is obtained via a weighted random sampling from a list of tokens that occur first in the SMILES strings of the training set. The weights of these tokens are determined by their frequency of occurrence in the first position of SMILES strings in the training set. Then, we provide the model a set of property and scaffold conditions along with the start token to sample a molecule.

Our model is illustrated in Figure 3. The model is essentially a mini version of the Generative Pre-Training Transformer (GPT) model.[44] Unlike GPT1 that has around 110 M parameters, MolGPT has only around 6 M parameters. MolGPT comprises stacked decoder blocks, each of which is composed of a masked self-attention layer and fully connected neural network. Each self-attention layer returns a vector of size 256 that is taken as input by the fully connected network. The hidden layer of the neural network outputs a vector of size 1024 and passes it through GELU activation layer. The final layer of the fully connected neural network returns a vector of size 256, that is then used as input for the next decoder block. MolGPT consists of eight such decoder blocks.

To keep track of the order of the input sequence, position value embeddings are assigned to each token. During conditional training, segment tokens are provided to distinguish between the condition and the SMILES tokens. Segment token embeddings represent whether a particular input is a condition or a molecule SMILES token for ease of differentiation between the two by the model. All the molecule SMILES tokens are mapped to a 256 dimensional vector using an embedding layer. Similarly, separate trainable embedding layers are used to map the position tokens and segment tokens to 256 dimensional vectors. These SMILES token embeddings, position embeddings, and segment token embeddings are then added, resulting in a vector of size 256 for each token of the SMILES string, which is then passed as input to the model.

GPT architectures work on a masked self-attention mechanism. Self-attention is calculated through "Scaled Dot Product Attention". This involves three sets of vectors, the query, key, and value vectors. Query vectors are used to query the weights of each individual value vector. They are first sent through a dot product with key vectors. These dot products are scaled by the dimensions of these vectors, and then a softmax function is applied to get the corresponding weights. The value vectors are multiplied by their respective weights and added. The query, key, and value vectors for each token are computed by weight matrices present in each decoder block. Attention can be represented by the following formula:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V$$

where Q, K, and V are query, key, and value vectors, respectively. d_k here is the dimension of query and key vectors, and T is transpose of the matrix.

Self-attention provides attention to all the tokens of a sequence for prediction. However, this is not ideal when we are training a model to predict the next token in a sequence. It is because, during generation, unlike during training, the network would have access only to the tokens predicted in the previous time-steps. Therefore, masked self-attention is applied to mask attention to all sequence tokens that occur in future time steps. Moreover, instead of performing a single masked self-attention operation, each block performs multiple masked self-attention operations (multihead attention) in parallel and concatenates the output. Multihead attention provides better representations by attending to different representation subspaces at different positions.

We train this model on molecules represented as SMILES strings. For this, we use a SMILES tokenizer to break up the string into a sequence of relevant tokens. Property conditions are sent through a separate fully connected linear layer that maps the conditions to a vector of 256 dimensions to provide a representation of the properties in a higher dimension. The resultant vector is then concatenated at the start of the sequence of the embeddings of the SMILES tokens. For scaffold conditions, we use the same embedding layer as molecule SMILES to map each token of the scaffold string to a 256-dimensional vector. Similar to property conditions, the scaffold representation is then concatenated at the start of the sequence of the embeddings of the SMILES tokens. The model is trained such that the predicted tokens are a result of attention to both the previous molecule tokens as well as the conditions.

Training Procedure and Evaluation Metrics. Each model is trained for 10 epochs using the Adam optimizer with a learning rate of 6×10^{-4}. During generation, a start token (that is randomly sampled from the list of first tokens of molecules in the training set) is provided to the network along with the conditions.

We trained and tested MolGPT on both the MOSES[10] and GuacaMol[9] data sets. We also conducted experiments to check MolGPT's capacity to control molecular properties and core structures. The models were trained on an NVIDIA 2080Ti GPU. Most of the models converged and showed best performance after 10 epochs. However, we noticed that training them for slightly fewer epochs led to similar results in terms of validity, novelty, and uniqueness of generated molecules, which are the metrics used here (details below).

- **Validity**: the fraction of a generated molecules that are valid. We use RDkit for validity check of molecules. Validity measures how well the model has learned the SMILES grammar and the valency of atoms.
- **Uniqueness**: the fraction of valid generated molecules that are unique. Low uniqueness highlights repetitive molecule generation and a low level of distribution learning by the model.
- **Novelty**: the fraction of valid unique generated molecules that are not in the training set. Low novelty is a sign of overfitting. We do not want the model to memorize the training data.
- **Internal Diversity (IntDiv_p)**: measures the diversity of the generated molecules, which is a metric specially designed to check for mode collapse or whether the model keeps generating similar structures. This uses the power (p) mean of the Tanimoto similarity (T) between the fingerprints of all pairs of molecules ($s1$, $s2$) in the generated set (S).

$$\text{IntDiv}_p(S) = 1 - \sqrt[p]{\frac{1}{|S|^2}\sum_{s1,s2\in S} T(s1, s2)^p}$$

https://doi.org/10.1021/acs.jcim.1c00600
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

Table 1. Comparison of the Different Metrics Corresponding to Nonconditioned Generation of Molecules Using Different Approaches Trained on MOSES Data Set

models	validity	unique@10K	novelty	$IntDiv_1$	$IntDiv_2$	FCD/Test	FCD/TestSF
CharRNN	0.975	**0.999**	0.842	0.856	0.85	0.0732	0.5204
VAE	0.977	0.998	0.695	0.856	0.85	0.099	0.567
AAE	0.937	0.997	0.793	0.856	0.85	0.555	1.057
LatentGAN	0.897	0.997	**0.949**	0.857	0.85	0.2968	0.8281
JT-VAE	**1.0**	**0.999**	0.914	0.855	0.849	0.395	0.938
MolGPT	0.994	**1.0**	0.797	0.857	0.851	**0.067**	**0.507**

We report $IntDiv_1$ ($p = 1$) and $IntDiv_2$ ($p = 2$) in this work.

- **Frechet ChemNet Distance**[52] **(FCD)**: calculated using the features of the generated molecules and the features of molecules in the data set. The features are obtained from the penultimate layer of the ChemNet model. Low FCD values indicate that the model has successfully captured the statistics of the data set. Mathematically, FCD between a generated distribution G and training data distribution D is defined as follows.

$$\mathrm{FCD}(G, D) = \|\mu_G - \mu_D\|^2 + Tr(\Sigma_G + \Sigma_D - 2(\Sigma_G\Sigma_D)^{1/2})$$

where μ_G is the mean and Σ_G is the covariance of the distribution G. For the Guacamol data set, the final FCD score is reported as

$$S = \exp(-0.2\mathrm{FCD})$$

So, for the Guacamol data set, a higher S value is considered to be better.

- **KL Divergence**: Following Brown et al.,[9] KL divergence is calculated using numerous physicochemical descriptors of the generated and the reference sets. Lower values indicate that the model has learned the distribution of these properties very well. KL divergence between two distributions P and Q for any given property is a measure of how well Q approximates P and is calculated as follows:

$$D_{KL}(P, Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)}$$

Reported here is the aggregated final score S over all the properties k calculated as

$$S = \frac{1}{k}\sum_i^k \exp(-D_{\mathrm{KL},i})$$

RESULTS AND DISCUSSION

In this section, we first present the results on nonconditioned generation of molecules. MolGPT's performance is then compared with other state-of-the-art approaches, followed by some insights on the interpretability of our model. We then demonstrate our model's ability of conditional generation based on property alone and scaffold alone. This is followed with the results on conditional generation based on property and scaffold together. Finally, we show examples of our model being used for optimization of QED value of a starting molecule and optimization of SAS value maintaining the scaffold, TPSA, and logP values.

Table 2. Comparison of the Different Metrics Corresponding to Nonconditioned Generation of Molecules Using Different Approaches Trained on GuacaMol Data Set

models	validity	unique	novelty	FCD	KL divergence
SMILES LSTM	0.959	**1.0**	0.912	**0.913**	0.991
AAE	0.822	**1.0**	0.998	0.529	0.886
Organ	0.379	0.841	0.687	0.000	0.267
VAE	0.870	0.999	0.974	0.863	0.982
MolGPT	**0.981**	0.998	**1.0**	0.907	**0.992**

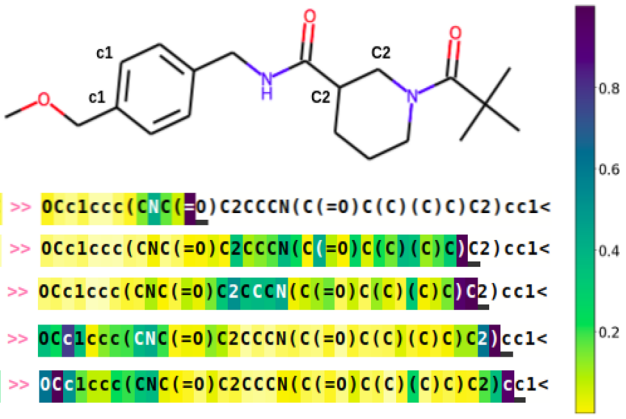


Figure 4. Input saliency maps for the shown generated molecule. The dark purple underlines are the tokens under consideration for saliency maps. The intensity of color of each token indicates the importance of that token for generating the underlined token.

Nonconditioned Molecular Generation. The chemical space is practically infinite and unexplored, and so a good generative model should try to generate a greater number of novel, valid molecules so as to help us explore that chemical space. High values of these metrics would ensure that the models have learned the molecule grammar well and are not overfitting to the training data simultaneously. Internal diversity scores give an idea about the extent of chemical space traversed by different models. FCD and KL divergence measure how well the model captures the statistics and distribution of the features of the data set, respectively. So, MolGPT is compared with previous approaches on these criteria. All metrics except validity are computed on the set of valid molecules generated by the model. We compare the performance of MolGPT on the MOSES data set to that of CharRNN, VAE, AAE, LatentGAN, and JT-VAE. JT-VAE uses graphs as input, while the others use SMILES. The model performance with a temperature of 1.0 on each data set is reported in Table 1 and Table 2.

On the MOSES benchmark, MolGPT has the best FCD score for molecules as well as their scaffolds. This indicates that the model has learned the data set statistics very well. It performs on par with other models in terms of the two internal diversity metrics. In the case of validity, as mentioned earlier,

https://doi.org/10.1021/acs.jcim.1c00600
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

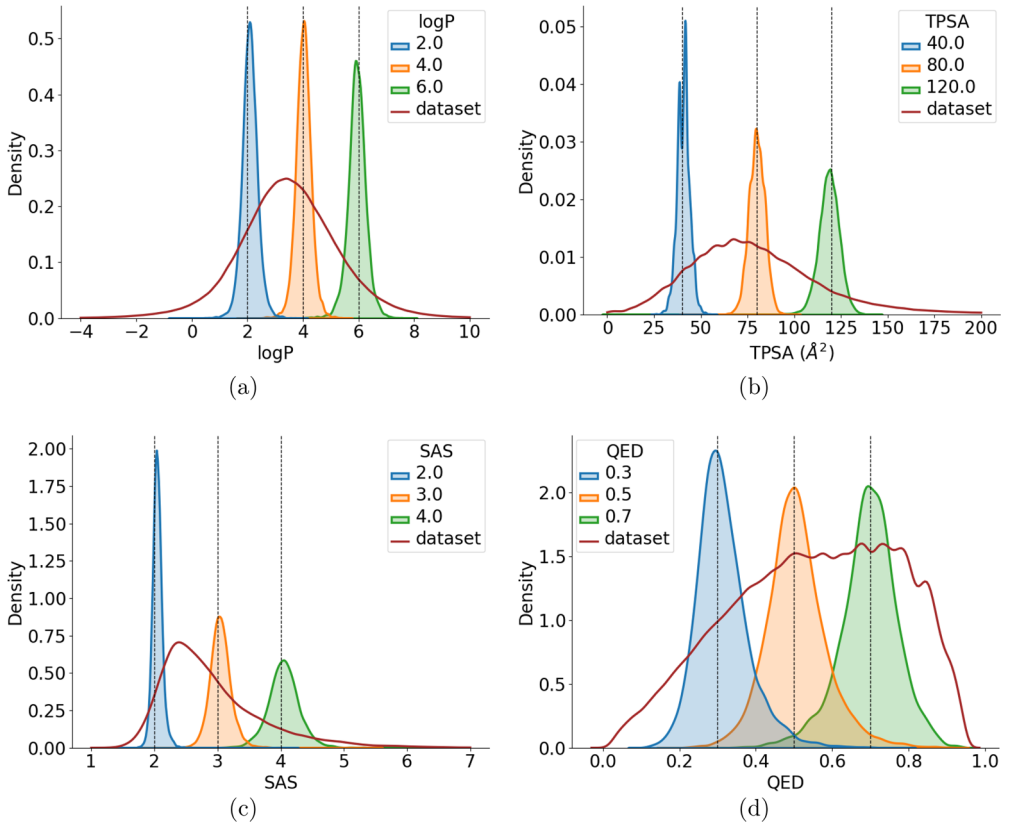


Figure 5. Distribution of property of generated molecules conditioned on (a) logP, (b) TPSA, (c) SAS, and (d) QED. Distribution depicted using a solid red line corresponds to the whole data set. Trained on GuacaMol data set with temperature = 1.0.

Table 3. Comparison of Different Metrics while Generating Molecules Conditioned on Single Property Based on Training on GuacaMol Data Set (Temperature Value of 1.0 Was Used)

condition	validity	unique	novelty	MAD	SD
logP	0.971	0.998	1.0	0.23	0.31
TPSA	0.972	0.996	1.0	3.52	4.66
SAS	0.977	0.995	1.0	0.13	0.2
QED	0.975	0.998	1.0	0.056	0.075

JT-VAE always generates a valid molecule because it checks validity at every step of generation. Barring JT-VAE, we observe that MolGPT performs best at generating valid and unique molecules. MolGPT has a near perfect validity score on the MOSES data set without the use of explicit constraints, indicating strong learning of the SMILES grammer and modeling of long-term dependencies that can be attributed to the attention mechanisms. MolGPT, however, has a low novelty score on the data set, being only slightly better than the AAE. On the GuacaMol benchmark, MolGPT demonstrates the best results on validity, novelty, and KL divergence, while its FCD is only 0.006 less than the RNN. So it is the preferred method when compared to other methods tested on it. It returns very high validity, uniqueness, and novelty scores on generation with a sampling temperature of 1.0. We believe this boost in novelty, as compared to MOSES, is due to a larger diversity in molecules in the GuacaMol data set (see Figure 1). Moreover, even though the GuacaMol data set has larger molecules as compared to the MOSES data set, MolGPT generates molecules with very high validity, also indicating that this method handles long-range dependencies very well.

While it is important to develop machine learning frameworks and pipelines for making tasks more efficient, it is desirable to also demonstrate that these models allow for interpretation. We use saliency maps to visualize the molecular generation process for our model. The usage of saliency maps has been well established in the context of transformer models and have been used previously for synthesis prediction.[53] Figure 4 shows input saliency maps for some of the generated tokens of the shown generated molecule. Input saliency methods assign a score to each input token that indicates the importance of that token in generating the next token. "(", "C", and "c" refer to the branching from chain, nonaromatic carbon, and aromatic carbon, respectively. From Figure 4, we see that when generating the "O" atom in the first saliency map, the model attends to the previous double bond and "N" atoms. The double bond satisfies the valency of the oxygen atom, and the "N" atom participates in the formation of the tautomer (Lactam and Lactim), which increases the stability of the structure. When generating the "C" atom in the second saliency map, the model attends to "(" and ")" to check if they are balanced and also attends to the atoms in the nonaromatic ring. In the nonaromatic ring, it attends mostly to the immediate neighbors—"2" and "N" atoms. When generating the "2" token, it attends to the immediate previous "C" token and the tokens in the nonaromatic ring. When generating "c" tokens in the last and second to last row of the saliency maps, the model rightly attends to the atoms in the aromatic ring since that ring is still incomplete. Thus, these saliency maps provide some insight into the chemical interpretability of the generative process.

https://doi.org/10.1021/acs.jcim.1c00600
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

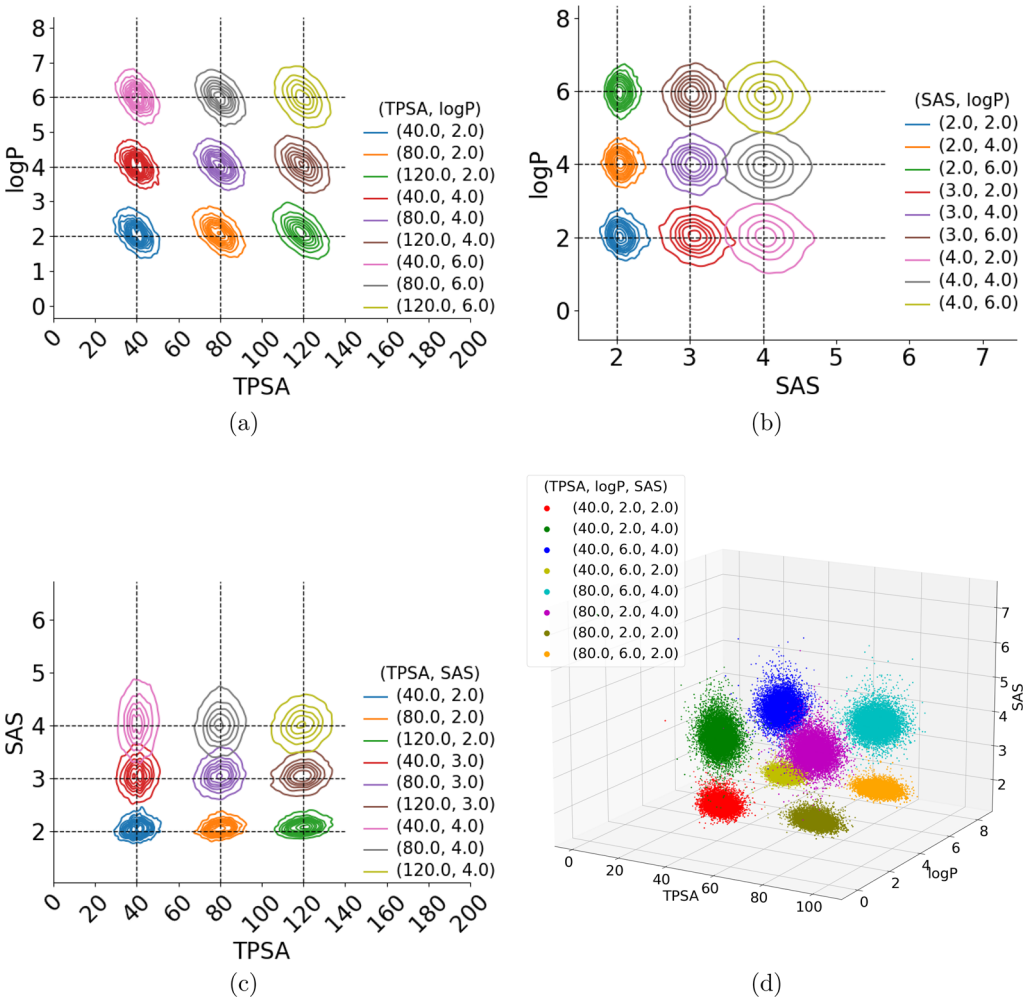


Figure 6. Distribution of property of generated molecules conditioned on (a) TPSA + logP, (b) SAS + logP, (c) SAS + TPSA, and (d) TPSA + logP + SAS. The values that the generation is conditioned to are given in the legends of the panels.

Table 4. Multiproperty Conditional Training on GuacaMol Data Set

				MAD/SD		
condition	validity	unique	novelty	TPSA	logP	SAS
SAS + logP	0.972	0.992	1.0		0.250/ 0.340	0.140/ 0.210
SAS + TPSA	0.971	0.988	1.0	3.760/ 4.940		0.150/ 0.220
TPSA + logP	0.965	0.994	1.0	3.710/ 4.770	0.240/ 0.320	
TPSA + logP + SAS	0.973	0.969	1.0	3.790/ 4.800	0.270/ 0.350	0.180/ 0.260

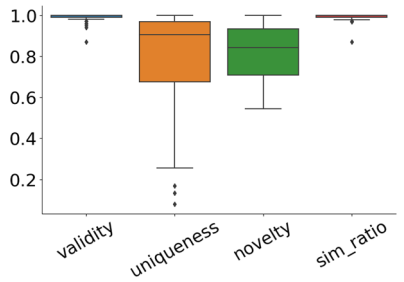


Figure 7. Boxplot of the evaluation metrics for the scaffold conditioned results.

Generation-based on Single and Multiple Properties. Many processes in biology and chemistry require molecules to have certain property values in order to perform some functions. For example, for molecules to penetrate the blood−brain barrier (and thus act on receptors in the central nervous system), a TPSA value less than 90 Å^2 is usually needed.[54] This motivates the need for models to have accurate conditional generation. So, the next objective is to evaluate the ability of MolGPT to generate molecules that exhibit specific properties (conditional generation). Since GuacaMol has a wider range in property values, we test the model's ability to control molecular properties trained on it. While only logP, SAS, TPSA, and QED are used for property control, we would like to note that the model can be trained to learn any property that is inferred from the molecule's 2D structure. For each condition, 10 000 molecules are generated to evaluate property control.

Distributions of molecular properties of MolGPT generated molecules while controlling a single property are depicted in Figure 5. The mean average deviation (MAD), standard deviation (SD), validity, uniqueness, and novelty values for each property are reported in Table 3. As seen in Figure 5, the distribution of properties is centered around the desired value. This is further exemplified by the low SD and MAD scores (relative to the range of the property values) in Table 3.

https://doi.org/10.1021/acs.jcim.1c00600
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

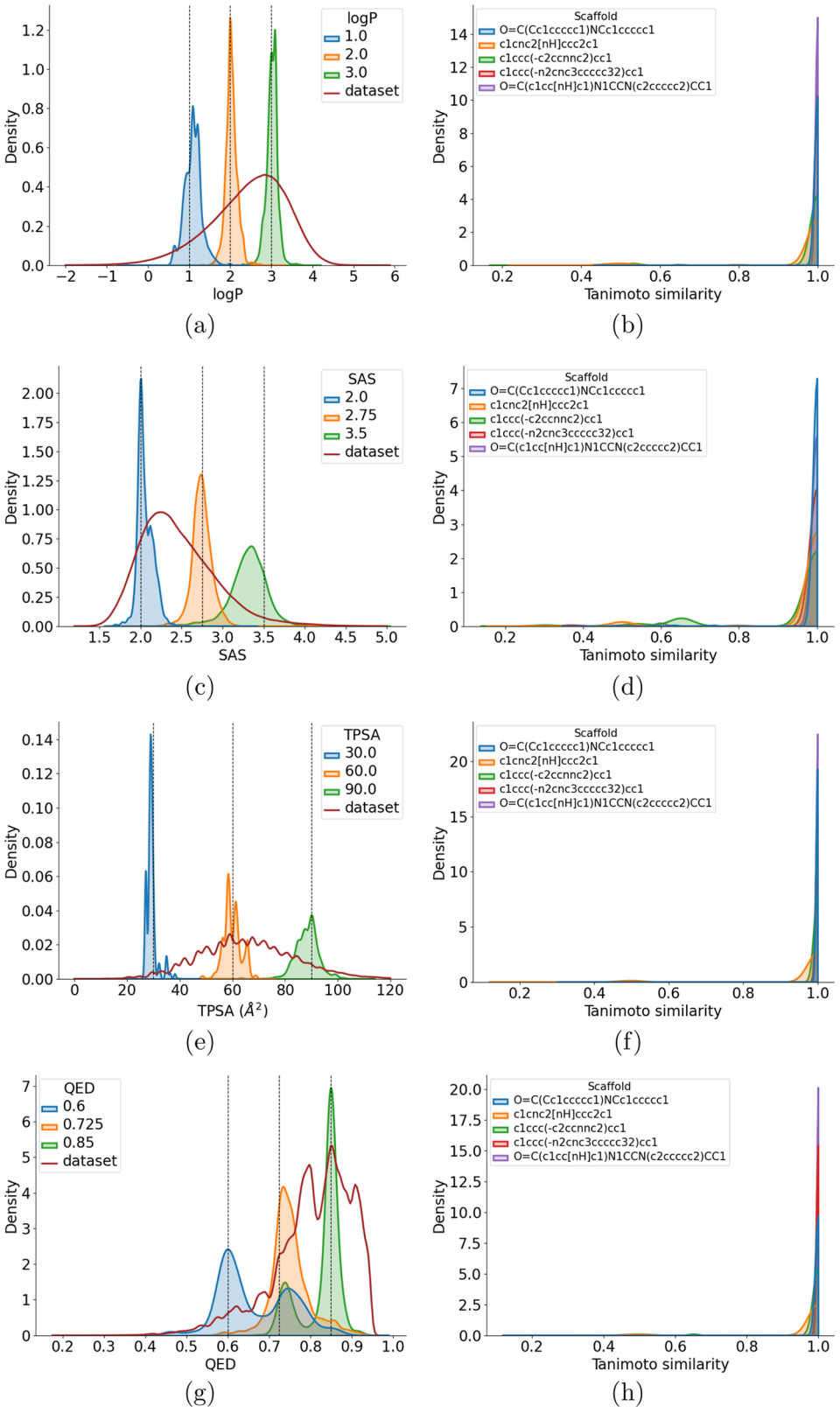


Figure 8. Distribution of property of generated molecules conditioned on Scaffold + (a) logP, (c) SAS, (e) TPSA, and (g) QED. Distribution of Tanimoto similarity of the scaffolds of the generated molecules and the scaffold used for conditioning for (b) logP, (d) SAS, (f) TPSA, and (h) QED. Trained on MOSES data set.

While generating molecules for specific purposes, in other words de novo design of molcules, it is necessary to optimize more than one property. For example, one may want molecules that have specific values for logP and TPSA. Hence, we check the model's capacity to control multiple properties simultaneously. For this, SAS, logP, and TPSA are used. We evaluate the model's ability to generate desired distributions using two and three property controls at a time. Generated distribution of

https://doi.org/10.1021/acs.jcim.1c00600
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

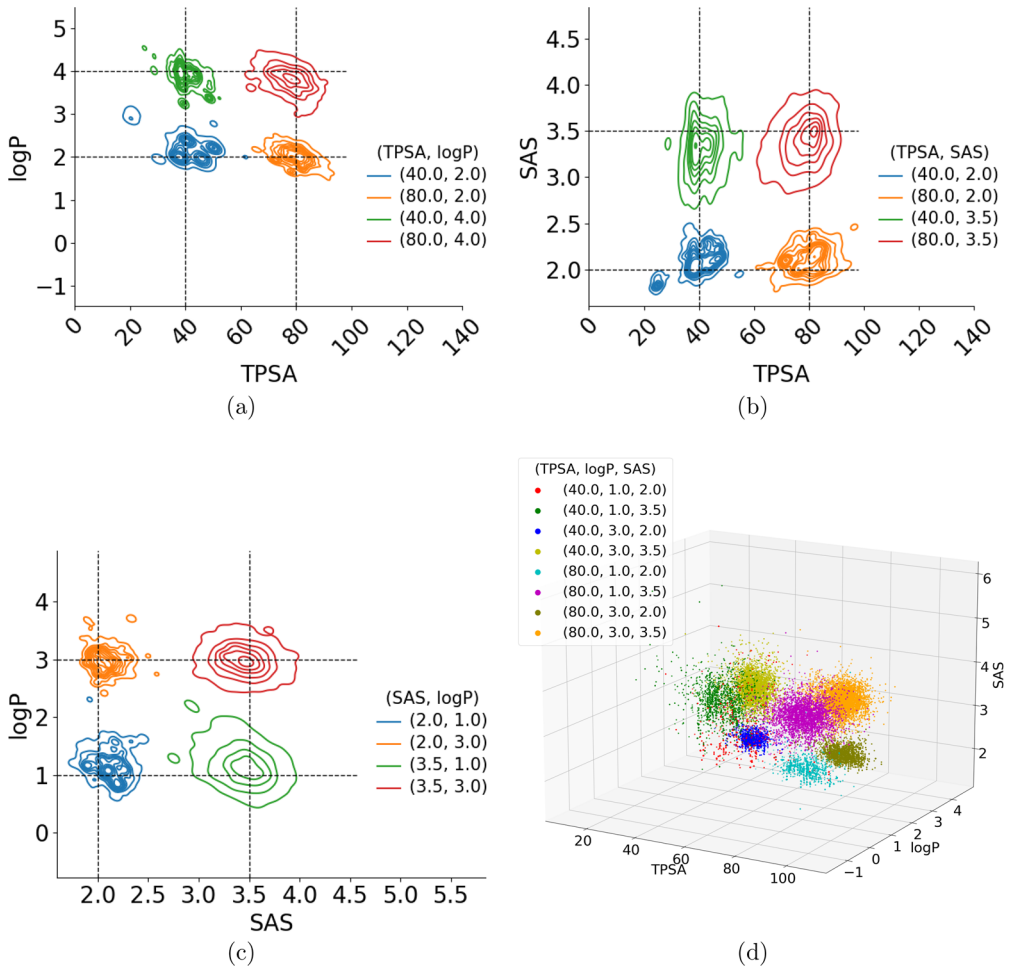


Figure 9. Distribution of property of generated molecules conditioned on Scaffold + (a) TPSA + logP, (b) SAS + TPSA, (c) SAS + logP, and (d) TPSA + logP + SAS. Trained on MOSES data set.

molecule properties is depicted in Figure 6. Well separated clusters centered at the desired property values are observed. As before, the low MAD and SD values for each property combination, reported in Table 4 (as compared to the range of property values), indicate the strong control that MolGPT has over multiple properties for accurate generation.

Generation Based on Scaffold. The above section demonstrated the ability of MolGPT to generate molecules with desired properties. In certain exercises, for example, lead optimization, chemists intend to generate molecules containing a specific scaffold/skeleton and at the same time achieve desired property values. We evaluate the ability of MolGPT to generate structures with certain property values while maintaining the structure of the scaffold, the results of which are presented in this and the next sections. We conduct these experiments on the MOSES benchmark data set as it contains a set of test scaffolds that are non-overlapping with the set of scaffolds that are present in the training set. We select a random set of 100 test scaffolds, then generate 100 molecules for each scaffold followed by calculation of validity, uniqueness, novelty, and "similarity ratio". "Similarity ratio" is defined as the fraction of valid generated molecules having Tanimoto similarity of the scaffold of the generated molecule and the conditioned scaffold greater than 0.8. Initially, Murcko scaffolds are obtained from the generated molecules. Tanimoto similarity is then calculated for the fingerprints of the Murcko scaffolds of the generated molecule and the conditional scaffold. We use the RDkit fingerprints with default settings for the same. The distribution of each of the metrics in terms of box plot is shown in Figure 7. From the boxplot, it can be seen that for all 100 scaffolds, the validity is greater than 0.8. Around 75% of scaffolds have uniqueness and novelty greater than 0.7. All of the scaffolds have a "similarity ratio" greater than 0.8, which suggests that most of the generated valid molecules have very similar scaffolds to the scaffold used for conditioning. The fraction of generated molecules that maintained the exact same scaffold structure as the conditioning is found to be 0.9897. Some examples of generated molecules for two scaffolds are given in Figure S2 of the Supporting Information. In all of the generated molecules, the conditioned scaffold structure is maintained.

Generation Based on Scaffold and Property. We evaluate the models' ability to generate structures containing desired scaffolds while also controlling multiple molecular properties. For our experiments, five scaffolds of different sizes were randomly chosen from the MOSES test set (Figure S1 of the Supporting Information). In these experiments, we define valid molecules as those molecular graphs that satisfy chemical valencies and contain scaffolds that have a Tanimoto similarity of at least 0.8 to the desired scaffold. The validity score of all scaffold based experiments is calculated based on this definition.

https://doi.org/10.1021/acs.jcim.1c00600
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

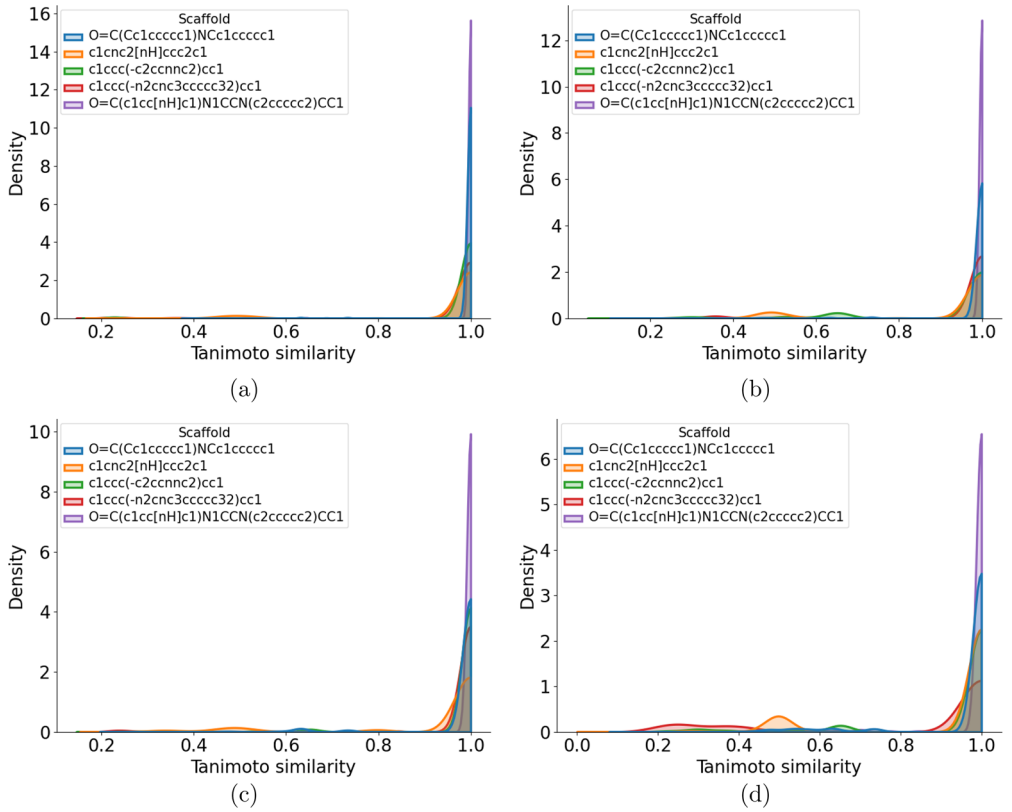


Figure 10. Distribution of Tanimoto similarity of the scaffolds of the generated molecules and the scaffold used for conditioning for (a) TPSA + logP, (b) SAS + TPSA, (c) SAS + logP, and (d) TPSA + logP + SAS. Trained on MOSES data set.

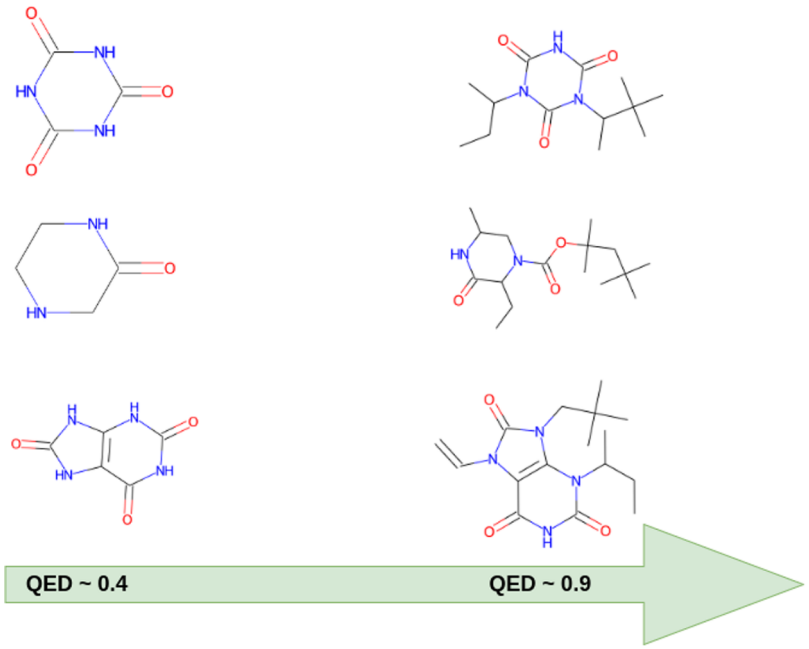


Figure 11. Optimization of QED value conditioned on the scaffold

Generated distributions for single property control can be seen in Figure 8. Tanimoto similarity is calculated between the scaffold of the generated molecule and the conditional scaffold. Distribution of these Tanimoto similarity scores is also plotted in Figure 8. The distribution plots peak at 1 for all of the scaffolds and properties. Since scaffold based generation is more constraining for property control, generated distributions are not as narrow and well separated as before. We also define a new metric called Same Scaffold Fraction (SSF) defined as the percentage of generated molecules that contain the same scaffold as the condition. The quantitative results along with SSF for single property control are reported in Table S1 in the Supporting Information. The low MAD and SD scores still show that MolGPT deviates only slightly from intended values

https://doi.org/10.1021/acs.jcim.1c00600
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

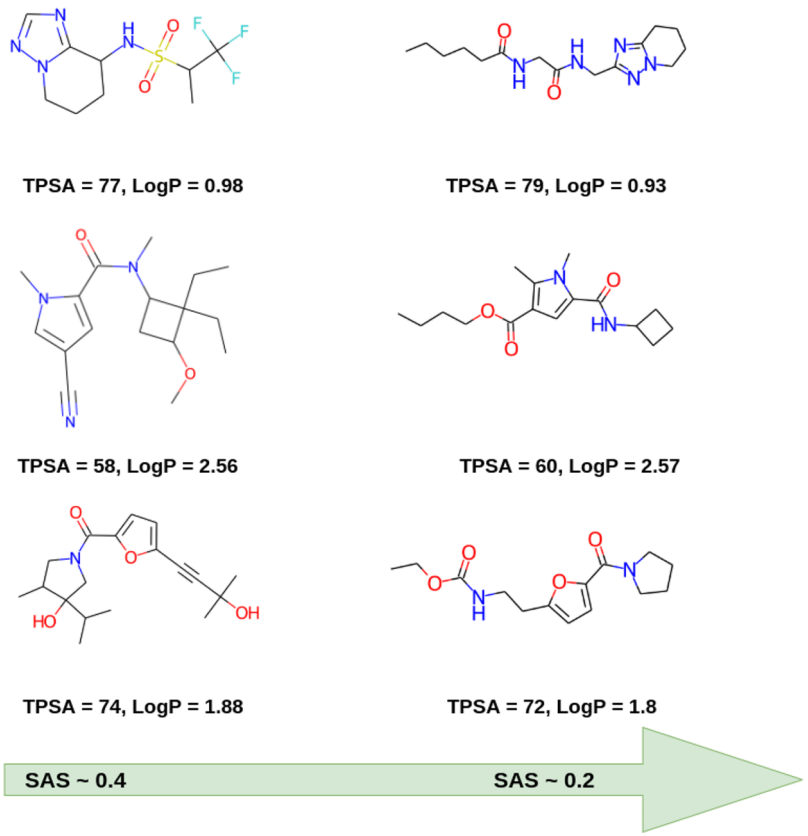


Figure 12. SAS reduced, maintaining TPSA, logP, and scaffold structure.

despite the constraints. QED is a function that is dependent on multiple molecular properties simultaneously. Therefore, QED is greatly influenced by the structure of the scaffold itself, making it very hard to control under such constraints. We believe such competing objectives are the reason for large overlap between distributions generated for QED control. Figure S3 of the Supporting Information shows the molecules conditioned on scaffold + logP and scaffold + SAS. MolGPT adds different functional groups to the scaffold in order to get the desired property value. Multiproperty control clusters are plotted in Figure 9. Even when using multiple properties, we see the Tanimoto similarity distributions peaking at 1 in Figure 10. Understandably, property-based clusters are not as well formed as before. However, there is a good separation between the clusters for two property control. The intended values of molecular properties are close to the centers of these clusters. This can further be verified by results reported for multi-property control in Table S2 in the Supporting Information. For three property control, one of the clusters (red) is not well formed due to highly constraining property values. We see that the rest of the clusters are largely well formed and separated.

Next, we show examples where conditional generation could be used to optimize simple properties of a molecule. To demonstrate this, three scaffolds from the test set having a QED around 0.4 are sampled. Using these scaffolds and a QED of 0.9 as the condition, we generate molecules using MolGPT. Sample generated molecules are shown in Figure 11. The scaffold structure is maintained in the generated molecules and their QED values are around 0.9. We also show other examples, where the TPSA, LogP, and scaffold strucuture are maintained and the SAS is improved to more desirable values in Figure 12. We would like to note that the model here is used only to optimize simple molecular properties and does not ensure its usability in complex tasks such as lead optimization. The model design, however, points to a possible research direction for other conditional generation tasks, such as generating molecules with better docking scores, for which sufficiently large data sets can be created.

CONCLUSION

In this work, we designed a Transformer−Decoder model called MolGPT for molecular generation. This model utilizes masked self-attention mechanisms that make it simpler to learn long-range dependencies between string tokens. This is especially useful to learn the semantics of valid SMILES strings that satisfy valencies and ring closures. We see through our benchmarking experiments that MolGPT shows very high validity and uniqueness scores for the MOSES and GuacaMol data sets. It also demonstrates good FCD and KL divergence scores on both the MOSES and Guacamol data sets. Furthermore, as shown, the generative process can be interpreted using saliency maps. Thus, MolGPT is able to show good performance on both data sets with it out-performing all other methods benchmarked on the GuacaMol data set in terms of validity and novelty.

We also show that the model learns higher level chemical representations through molecular property control. MolGPT is able to generate molecules with property values that deviate only slightly from the exact values that are passed by the user. It is also able to generate molecules containing user specified scaffolds while controlling these properties. It does this with good accuracy despite the constraining conditions of scaffold-based drug design. Consequently, we believe that the MolGPT model should be considered a strong architecture to be used by

https://doi.org/10.1021/acs.jcim.1c00600
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX

itself or incorporated into other molecular generation techniques.

■ ASSOCIATED CONTENT

Supporting Information

The Supporting Information is available free of charge at https://pubs.acs.org/doi/10.1021/acs.jcim.1c00600.

Results of scaffold and property conditioning, figures of scaffolds, generated molecules from scaffold conditioning, as well as scaffold and property conditioning experiments (PDF)

■ AUTHOR INFORMATION

Corresponding Author

U. Deva Priyakumar − *International Institute of Information Technology, Hyderabad 500 032, India;* orcid.org/0000-0001-7114-3955; Email: deva@iiit.ac.in

Authors

Viraj Bagal − *International Institute of Information Technology, Hyderabad 500 032, India; Indian Institute of Science Education and Research, Pune 411 008, India*

Rishal Aggarwal − *International Institute of Information Technology, Hyderabad 500 032, India*

P. K. Vinod − *International Institute of Information Technology, Hyderabad 500 032, India*

Complete contact information is available at: https://pubs.acs.org/10.1021/acs.jcim.1c00600

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

The authors thank Manasa Kondamadugu, Yashaswi Pathak, Sarvesh Mehta, and Manan Goel for their comments during the preparation of the manuscript. We thank IHub-Data, IIIT Hyderabad, and Kohli Center on Intelligent Systems, IIIT Hyderabad for financial support. Data and method implementation is available at https://github.com/devalab/molgpt.

■ REFERENCES

(1) Polishchuk, P. G.; Madzhidov, T. I.; Varnek, A. Estimation of the size of drug-like chemical space based on GDB-17 data. *J. Comput.-Aided Mol. Des.* **2013**, *27*, 675−679.

(2) Kim, S.; Thiessen, P. A.; Bolton, E. E.; Chen, J.; Fu, G.; Gindulyte, A.; Han, L.; He, J.; He, S.; Shoemaker, B. A.; et al. PubChem substance and compound databases. *Nucleic Acids Res.* **2016**, *44*, D1202−D1213.

(3) Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative adversarial nets. *Advances in neural information processing systems* **2014**, *27*, 2672−2680.

(4) Karras, T.; Laine, S.; Aila, T. A style-based generator architecture for generative adversarial networks. *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* **2019**, 4401−4410.

(5) Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. *Advances in Neural Information Processing Systems* **2017**, 5998−6008.

(6) Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint* **2018**, arXiv:1810.04805.

(7) Chen, H.; Engkvist, O.; Wang, Y.; Olivecrona, M.; Blaschke, T. The rise of deep learning in drug discovery. *Drug Discovery Today* **2018**, *23*, 1241−1250.

(8) Sanchez-Lengeling, B.; Aspuru-Guzik, A. Inverse molecular design using machine learning: Generative models for matter engineering. *Science* **2018**, *361*, 360−365.

(9) Brown, N.; Fiscato, M.; Segler, M. H.; Vaucher, A. C. GuacaMol: benchmarking models for de novo molecular design. *J. Chem. Inf. Model.* **2019**, *59*, 1096−1108.

(10) Polykovskiy, D.; Zhebrak, A.; Sanchez-Lengeling, B.; Golovanov, S.; Tatanov, O.; Belyaev, S.; Kurbanov, R.; Artamonov, A.; Aladinskiy, V.; Veselov, M.; Kadurin, A.; Johansson, S.; Chen, H.; Nikolenko, S.; Aspuru-Guzik, A.; Zhavoronkov, A. Molecular sets (moses): A benchmarking platform for molecular generation models. *Front. Pharmacol.* **2020**, *11*, 11.

(11) Weininger, D. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *J. Chem. Inf. Model.* **1988**, *28*, 31−36.

(12) Pathak, Y.; Laghuvarapu, S.; Mehta, S.; Priyakumar, U. D. Chemically interpretable graph interaction network for prediction of pharmacokinetic properties of drug-like molecules. *Proceedings of the AAAI Conference on Artificial Intelligence* **2020**, *34*, 873−880.

(13) Segler, M. H.; Kogej, T.; Tyrchan, C.; Waller, M. P. Generating focused molecule libraries for drug discovery with recurrent neural networks. *ACS Cent. Sci.* **2018**, *4*, 120−131.

(14) Gupta, A.; Müller, A. T.; Huisman, B. J.; Fuchs, J. A.; Schneider, P.; Schneider, G. Generative recurrent networks for de novo drug design. *Mol. Inf.* **2018**, *37*, 1700111.

(15) Popova, M.; Isayev, O.; Tropsha, A. Deep reinforcement learning for de novo drug design. *Sci. Adv.* **2018**, *4*, No. eaap7885.

(16) Olivecrona, M.; Blaschke, T.; Engkvist, O.; Chen, H. Molecular de-novo design through deep reinforcement learning. *J. Cheminf.* **2017**, *9*, 48.

(17) Liu, Q.; Allamanis, M.; Brockschmidt, M.; Gaunt, A. Constrained graph variational autoencoders for molecule design. *Advances in neural information processing systems* **2018**, *31*, 7795−7804.

(18) Kusner, M. J.; Paige, B.; Hernández-Lobato, J. M. Grammar variational autoencoder. *arXiv preprint* **2017**, arXiv:1703.01925.

(19) Simonovsky, M.; Komodakis, N. Graphvae: Towards generation of small graphs using variational autoencoders. *International Conference on Artificial Neural Networks* **2018**, *11139*, 412−422.

(20) Jin, W.; Barzilay, R.; Jaakkola, T. Junction tree variational autoencoder for molecular graph generation. *arXiv preprint* **2018**, arXiv:1802.04364.

(21) Lim, J.; Ryu, S.; Kim, J. W.; Kim, W. Y. Molecular generative model based on conditional variational autoencoder for de novo molecular design. *J. Cheminf.* **2018**, *10*, 1−9.

(22) Kadurin, A.; Nikolenko, S.; Khrabrov, K.; Aliper, A.; Zhavoronkov, A. druGAN: an advanced generative adversarial autoencoder model for de novo generation of new molecules with desired molecular properties in silico. *Mol. Pharmaceutics* **2017**, *14*, 3098−3104.

(23) Putin, E.; Asadulaev, A.; Vanhaelen, Q.; Ivanenkov, Y.; Aladinskaya, A. V.; Aliper, A.; Zhavoronkov, A. Adversarial threshold neural computer for molecular de novo design. *Mol. Pharmaceutics* **2018**, *15*, 4386−4397.

(24) Polykovskiy, D.; Zhebrak, A.; Vetrov, D.; Ivanenkov, Y.; Aladinskiy, V.; Mamoshina, P.; Bozdaganyan, M.; Aliper, A.; Zhavoronkov, A.; Kadurin, A. Entangled conditional adversarial autoencoder for de novo drug discovery. *Mol. Pharmaceutics* **2018**, *15*, 4398−4405.

(25) Hong, S. H.; Ryu, S.; Lim, J.; Kim, W. Y. Molecular Generative Model Based on an Adversarially Regularized Autoencoder. *J. Chem. Inf. Model.* **2020**, *60*, 29−36.

(26) Arús-Pous, J.; Johansson, S. V.; Prykhodko, O.; Bjerrum, E. J.; Tyrchan, C.; Reymond, J.-L.; Chen, H.; Engkvist, O. Randomized SMILES strings improve the quality of molecular generative models. *J. Cheminf.* **2019**, *11*, 1−13.

(27) Bjerrum, E. J. SMILES enumeration as data augmentation for neural network modeling of molecules. *arXiv preprint* **2017**, arXiv:1703.07076.

(28) Bjerrum, E. J.; Sattarov, B. Improving chemical autoencoder latent space and molecular de novo generation diversity with heteroencoders. *Biomolecules* **2018**, *8*, 131.
(29) Pathak, Y.; Juneja, K. S.; Varma, G.; Ehara, M.; Priyakumar, U. D. Deep learning enabled inorganic material generator. *Phys. Chem. Chem. Phys.* **2020**, *22*, 26935−26943.
(30) Prykhodko, O.; Johansson, S. V.; Kotsias, P.-C.; Arús-Pous, J.; Bjerrum, E. J.; Engkvist, O.; Chen, H. A de novo molecular generation method using latent vector based generative adversarial network. *J. Cheminf.* **2019**, *11*, 74.
(31) Guimaraes, G. L.; Sanchez-Lengeling, B.; Outeiral, C.; Farias, P. L. C.; Aspuru-Guzik, A. Objective-reinforced generative adversarial networks (ORGAN) for sequence generation models. *arXiv preprint* **2017**, arXiv:1705.10843.
(32) Sanchez-Lengeling, B.; Outeiral, C.; Guimaraes, G. L.; Aspuru-Guzik, A. Optimizing distributions over molecular space. An Objective-Reinforced Generative Adversarial Network for Inverse-design Chemistry (ORGANIC). *ChemRxiv*; Cambridge: Cambridge Open Engage, 2017.
(33) De Cao, N.; Kipf, T. MolGAN: An implicit generative model for small molecular graphs. *arXiv preprint* **2018**, arXiv:1805.11973.
(34) Putin, E.; Asadulaev, A.; Ivanenkov, Y.; Aladinskiy, V.; Sanchez-Lengeling, B.; Aspuru-Guzik, A.; Zhavoronkov, A. Reinforced adversarial neural computer for de novo molecular design. *J. Chem. Inf. Model.* **2018**, *58*, 1194−1204.
(35) Gómez-Bombarelli, R.; Wei, J. N.; Duvenaud, D.; Hernández-Lobato, J. M.; Sánchez-Lengeling, B.; Sheberla, D.; Aguilera-Iparraguirre, J.; Hirzel, T. D.; Adams, R. P.; Aspuru-Guzik, A. Automatic chemical design using a data-driven continuous representation of molecules. *ACS Cent. Sci.* **2018**, *4*, 268−276.
(36) Mehta, S.; Laghuvarapu, S.; Pathak, Y.; Sethi, A.; Alvala, M.; Priyakumar, U. D. D. MEMES: Machine learning framework for Enhanced MolEcular Screening. *Chemical Science* **2021**, *12*, 11710.
(37) Winter, R.; Montanari, F.; Steffen, A.; Briem, H.; Noé, F.; Clevert, D.-A. Efficient multi-objective molecular optimization in a continuous latent space. *Chemical science* **2019**, *10*, 8016−8024.
(38) Kim, K.; Kang, S.; Yoo, J.; Kwon, Y.; Nam, Y.; Lee, D.; Kim, I.; Choi, Y.-S.; Jung, Y.; Kim, S.; Son, W.-J.; Son, J.; Lee, H. S.; Kim, S.; Shin, J.; Hwang, S. Deep-learning-based inverse design model for intelligent discovery of organic molecules. *npj Comput. Mater.* **2018**, *4*, 1−7.
(39) Maziarka, Ł.; Pocha, A.; Kaczmarczyk, J.; Rataj, K.; Danel, T.; Warchoł, M. Mol-CycleGAN: a generative model for molecular optimization. *J. Cheminf.* **2020**, *12*, 1−18.
(40) Zhu, J.-Y.; Park, T.; Isola, P.; Efros, A. A. Unpaired image-to-image translation using cycle-consistent adversarial networks. *Proceedings of the IEEE international conference on computer vision* **2017**, 2242−2251.
(41) Kotsias, P.-C.; Arús-Pous, J.; Chen, H.; Engkvist, O.; Tyrchan, C.; Bjerrum, E. J. Direct steering of de novo molecular generation with descriptor conditional recurrent neural networks. *Nature Machine Intelligence* **2020**, *2*, 254−265.
(42) Arús-Pous, J.; Patronov, A.; Bjerrum, E. J.; Tyrchan, C.; Reymond, J.-L.; Chen, H.; Engkvist, O. SMILES-based deep generative scaffold decorator for de-novo drug design. *J. Cheminf.* **2020**, *12*, 1−18.
(43) Lim, J.; Hwang, S.-Y.; Kim, S.; Moon, S.; Kim, W. Y. Scaffold-based molecular design using graph generative model. *arXiv preprint* **2019**, arXiv:1905.13639.
(44) Radford, A.; Narasimhan, K.; Salimans, T.; Sutskever, I. Improving language understanding by generative pre-training.
(45) Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I. Language Models are Unsupervised Multitask Learners. 2019.
(46) Irwin, J. J.; Shoichet, B. K. ZINC- a free database of commercially available compounds for virtual screening. *J. Chem. Inf. Model.* **2005**, *45*, 177−182.
(47) Gaulton, A.; Hersey, A.; Nowotka, M.; Bento, A. P.; Chambers, J.; Mendez, D.; Mutowo, P.; Atkinson, F.; Bellis, L. J.; Cibrián-Uhalte, E.; et al. The ChEMBL database in 2017. *Nucleic Acids Res.* **2017**, *45*, D945−D954.
(48) Landrum, G. *RDKit: A Software Suite for Cheminformatics, Computational Chemistry, and Predictive Modeling*, 2013.
(49) Bemis, G. W.; Murcko, M. A. The properties of known drugs. 1. Molecular frameworks. *J. Med. Chem.* **1996**, *39*, 2887−2893.
(50) Ertl, P.; Schuffenhauer, A. Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *J. Cheminf.* **2009**, *1*, 8.
(51) Bickerton, G. R.; Paolini, G. V.; Besnard, J.; Muresan, S.; Hopkins, A. L. Quantifying the chemical beauty of drugs. *Nat. Chem.* **2012**, *4*, 90−98.
(52) Preuer, K.; Renz, P.; Unterthiner, T.; Hochreiter, S.; Klambauer, G. Fréchet ChemNet distance: a metric for generative models for molecules in drug discovery. *J. Chem. Inf. Model.* **2018**, *58*, 1736−1741.
(53) Schwaller, P.; Laino, T.; Gaudin, T.; Bolgar, P.; Hunter, C. A.; Bekas, C.; Lee, A. A. Molecular transformer: a model for uncertainty-calibrated chemical reaction prediction. *ACS Cent. Sci.* **2019**, *5*, 1572−1583.
(54) Clark, D. E. Rapid calculation of polar molecular surface area and its application to the prediction of transport phenomena. 1. Prediction of intestinal absorption. *J. Pharm. Sci.* **1999**, *88*, 807−814.

https://doi.org/10.1021/acs.jcim.1c00600
J. Chem. Inf. Model. XXXX, XXX, XXX−XXX