InFrag: Using Attribution-based Explainability to Guide Deep Molecular Optimization

Pierre Wüthrich,[*,†,‡] Jun Jin Choong,[*,†,‡] and Shinya Yuki[*,†]

†*Elix, Inc., Tokyo, Japan*

‡*Contributed equally to this work*

E-mail: pierre.wuthrich@elix-inc.com; junjin.choong@elix-inc.com; shinya.yuki@elix-inc.com

Abstract

The recently proposed Genetic expert guided learning (GEGL) framework has demonstrated impressive performances on several *de novo* molecular design tasks. Despite the displayed state-of-the art results, the proposed system relies on an expert-designed Genetic expert. Although hand-crafted experts allow to navigate the chemical space efficiently, designing such experts requires a significant amount of effort and might contain inherent biases which can potentially slow down convergence or even lead to sub-optimal solutions. In this research, we propose a novel genetic expert named *InFrag* which is free of design rules and can generate new molecules by combining promising molecular fragments. Fragments are obtained by using an additional graph convolutional neural network which computes attributions for each atom for a given molecule. Molecular substructures which contribute positively to the task score are kept and combined to propose novel molecules. We experimentally demonstrate that, within the GEGL framework, our proposed attribution-based genetic expert is either competitive or outperforms the original expert-designed genetic expert on goal-directed optimization tasks. When limiting the number of optimization rounds to one and three rounds, a performance increase of approximately 43% and 20% respectively is

observed compared to the baseline genetic expert. Furthermore, we empirically show that combining several experts that share a fixed sampling budget at each optimization round generally improves or maintains the overall performance of the framework.

1 Introduction

The ability to discover and design *de novo* molecules with desired properties is of great interest in a wide range of applications ranging from drug discovery [1] to materials engineering. [2,3] This high-dimensional optimization task can be addressed via the inverse molecular design paradigm [2] which tries to find suitable candidate compounds given some target properties. This task is non-trivial considering the size of the molecular space; the drug-like chemical space alone is estimated to be about 10^{60}. [4] To tackle this challenge, one has to design a model that is computationally tractable and feasible. This optimization challenge can be framed broadly into several classes. Coley et al. [5] divides these classes into physical matters, which encompasses most drug discovery efforts on the output of the compound. These compounds are potentially used as part of a therapeutic. The second class is the process, which is seen as an important step to transition from *in silico* to a real-world experimental phase. The final class is defined as the model itself, which is the core engine driving the design choices behind each generation of *de novo* molecules.

In this paper, we shall focus our effort towards modeling of *de novo* molecules. The key challenges of seeking a *de novo* design is the process of (1) molecule generation, (2) the scoring function of the molecules, and (3) an optimization process for training the model with respect to the scoring function. [6] Each of these components have their own set of challenges. For instance, the process of generating *de novo* molecules requires models that are capable of ensuring structural fidelity (i.e., generated molecules should be sensible) and structurally valid (i.e., molecules are stable). The scoring function plays an important role to guide the model during the optimization process. A trivial scoring function might explore undesired or obvious parts of the chemical space. In contrast, an over-complicated scoring function, in

terms of imposed constraints, could cause instability during optimization as the model might not be able find suitable solutions. Finally, the optimization process should inherently reduce the training complexity from an enormous search space to circumvent the difficulty of an intractable computation problem.

1.1 Generative Models

Generation of *de novo* molecules can be achieved through several means. There are three distinct strategies that can be used by methods proposing novel molecules, each tackling the generative problem at a different level of granularity.[7] Atom-based strategies construct molecules atom by atom, while fragment-based strategies consider molecular structure combination with more than one atom. The third type of strategies are reaction-based ones which utilize a set of reactants and reaction rules to produce new molecules.

The simplest approach to molecular generation is to consider it in terms of manipulating molecules in the SMILES strings format.[8] Gómez-Bombarelli et al., Segler et al.[9,10] perform interpolation of SMILES string in the latent space with GrammarVAE[11] such that new SMILES strings can be sampled. Since SMILES strings are brittle representations of molecules, these models have the potential to generate invalid SMILES that are unparsable and therefore cannot be generated. As an alternative, Dai et al.[12] introduced a decoder with syntactic and semantic constraints of SMILES via context-free and attribute grammars. On the other hand, techniques such as active learning[13] and reinforcement learning[14] are popular approaches to constrain the model towards generating valid SMILES through label signals and valid representation states during training. These approaches may further be refined with transfer learning to ensure a reasonable search space is first established before an applied learning regime is performed.[15]

In addition, models that induce structural constraints are among popular approaches which are able to ensure validity of the generated molecules. For instance, Simonovsky and Komodakis[16] represented molecules with their corresponding graph structure in terms of

adjacency matrices. The model is then trained to reconstruct these matrices while preserving one-to-one matching from the input graph. The outcome is a trained variational distribution that allows new samples to be drawn, in order to produce novel molecules. However, this approach is extremely costly due to complexity of the graph-matching algorithm; requiring a time complexity of $\mathcal{O}(n^4)$, where n denotes the number of nodes in a single molecular graph. From a different perspective, You et al., Li et al. [17,18] considered a node-by-node generation approach in attempt to alleviate the aforementioned problem. This specific approach is largely seen as successful up to a certain degree, but perfect validity for all generated molecules was not demonstrated. You et al. [19] combined reinforcement learning and a graph convolutional neural network to construct molecular graphs one node at a time.

In terms of fragment-based methods, Jin et al. [20] showed that using node by node generation could lead to invalid intermediate states. Instead, generating structure by structure is a preferable approach with which the aforementioned pitfall can be circumvented. The proposed solution, called Junction Tree Variational Autoencoder (JTVAE), [20] enforces structure by structure generation through means of tree decomposition before projecting to a latent space. A similar choice of strategy is proposed in ChemTS [21] which presents a simpler algorithm utilizing Monte Carlo Tree Search (MCTS) instead of a Variational Autoencoder (VAE). This approach ensures that functional groups are preserved while mutations are induced via a Recurrent Neural Network (RNN). Since the algorithm employs a MCTS strategy alongside a RNN, it is one of the fastest generative model capable of generating approximately 40 molecules per minute. In contrast, CVAE [9] with Bayesian Optimization (BO) which employs a VAE architecture was the slowest, generating approximately 0.1 molecules per minute. Polishchuk [22] recently proposed a simple framework called CReM which memorizes structures as well as their atomic context within the molecule they originated from. New valid molecules are generated by swapping structures with other structures sharing the same context.

In terms of practicality, reaction-based methods are arguably the most common choice to generate a set of new molecules. These approaches utilize a model that predicts the

forward reaction *in silico*. SYNOPSIS[23] is one of the earliest methods that considers applying virtual reactions iteratively to maximize the objective function. AutoGrow4[24] used genetic algorithms and a set of reaction libraries derived from robust organic reactions to mutate molecules in a population. Jin et al.[25] proposed RexGen which utilizes a Weisfeiler-Lehman Network to predict reactions using a highly popular reaction-product dataset, USPTO.[26] Bradshaw et al.[27] proposed MoleculeChef that optimizes the latent space via VAE and utilizes a reaction-prediction model like Molecular Transformer[28] to approximate the forward reaction while ensuring synthesizability. ChemBO[29] approaches the same problem in a black box optimization manner where Bayesian Optimization is leveraged as the optimization algorithm. The oracle, in particular, is the RexGen forward reaction-prediction model.

1.2 Interpretability Methods for Graph Neural Networks

Interest in interpretability methods has surged in recent years. Explainability methods like gradient-based saliency maps,[30,31] Class Activation Mapping (CAM),[32] and Excitation Backpropagation (EB)[33] are widely applicable to the computer vision domain. These methods are the most common explainability methods originally designed for Convolutional Neural Networks (CNN). Recently, Pope et al.[34] extended these methods to Graph Convolutional Neural Networks (GCNN).[35] However, their ablation study was limited to visualization only. Adding interpretability to graph-based deep neural networks is currently an active research field.

To the best of our knowledge, there is a limited number of work utilizing interpretability to explicitly guide the generative optimization process. In an attempt to exploit interpretability, Jin et al.[36] proposed RelationaleRL. The proposed method tries to solve multi-objective problems by first identifying molecular substructures for which the desired property is highly probable and subsequently combining these structures using a graph generative model. Although our proposed method is similar in its fundamental idea, our work is distinct in several ways to this prior work: 1) Our method is significantly simpler compared to the above

work as it does not require a parameterized graph generative model as well as a fine-tuning phase. We ensure simplicity in our fragment-based genetic expert by randomly sampling and combining the extracted fragments without needing to fall back to reinforcement learning methods such as MCTS; 2) The molecular representation to manipulate the fragments is both string-based when generating molecules as well as during the generation of novel candidate molecules. The graph-based representation of molecules is only used during the creation of the atom-wise attributions.

1.3 Genetic Expert Guided Learning

Our work is heavily inspired and builds upon the recently introduced *Genetic Expert Guided Learning* (GEGL) framework proposed by Ahn et al.[37] which combines meta-heuristic optimization with reinforcement learning. GEGL has demonstrated strong capabilities and is at the time of writing the state-of-the-art in deep molecular optimization. The framework is composed of 4 distinct components: a neural apprentice, two reward priority queues and a genetic expert.

The **Neural Apprentice** is a long short-term memory network tasked with reasoning over and generating high-rewarding molecules in SMILES representation. At each optimization step, the neural apprentice first generates novel candidate molecules token-by-token. These molecules are then passed and processed by the first reward priority queue. Formally, the neural apprentice is represented by its *apprentice policy*, $\pi(\boldsymbol{x};\theta)$, with $\boldsymbol{x}$ denoting the molecule $\boldsymbol{x}$ and θ denoting the trainable parameters. The *apprentice policy* is then expected to generate a new set of molecules $\boldsymbol{x}_{new}$ that maximizes the reward $r(\boldsymbol{x})$. The reward is a function that scores the generated molecules $\boldsymbol{x}_{new}$. In this work, we follow the original work by Ahn et al., utilizing molecular properties as the reward.

Reward Priority queues function as a long-term memory storage as they first score the candidate molecules, add them and finally rank them against previously memorized high-rewarding molecules. Each priority queue has a limited memory capacity such that

molecules with lower scores are dropped out of the memory if better candidates are present. The best performing molecules generated by the neural apprentice are extracted from the priority queue related to the neural apprentice. These molecules are utilized as the mating pool for the genetic expert. Formally, the queue $\mathcal{Q}$ and $\mathcal{Q}_{\text{ex}}$ can store up to K generated molecules. Here, $\mathcal{Q}$ and $\mathcal{Q}_{\text{ex}}$ is used to define the reward priority queue for the neural apprentice and the genetic expert respectively.

Genetic Expert. The original GEGL work made use of the graph-based genetic expert proposed by Jensen.[38] Genetic experts allow methods to rapidly traverse the chemical space and drive exploration. The child molecules from the expert are passed and processed in a separate priority queue (denoted by $\mathcal{Q}_{\text{ex}}$) in the same manner as described above. The last step consists of combining all high performing molecules from both queues and to improve the neural apprentice via imitation learning. As noted in the original work, GEGL can be interpreted as a Reinforcement Learning method with a Markov decision process where the episode length is fixed to one and actions correspond to the sampled molecules. The framework setup ensures that the performance is either maintained or increased over time. The combination of a genetic expert, which is capable to rapidly propose a diverse set of novel molecules, with a neural apprentice, which is able to reason over and steer generation towards a desirable molecular distribution, explains the displayed performances by the base framework.

The framework can be summarized in the following steps:

Step I: The neural apprentice with apprentice policy $\pi(\boldsymbol{x};\theta)$ generates a new set of molecules $\boldsymbol{x}_{\text{new}}$ and is stored in a priority queue Q, ordered according to the reward.

Step II: The priority queue is passed to the genetic expert and the expert policy π_{ex} generates a new set of molecules using the initial priority queue as seed molecules. These molecules are added to a new priority queue Q_{ex}.

Step III: Using both queues, Q and Q_{ex}, the apprentice policy update its trainable parameter θ via imitation learning, such that molecules are sampled from the union of the priority queues $Q \cup Q_{\text{ex}}$.

The above framework has several considerable advantages. We note that each component is purposefully kept modular, allowing to exchange them with more sophisticated components if desired. In addition, the neural apprentice policy and expert policy are both computationally efficient generator functions.

In this research, we propose a novel genetic expert which makes use of the SELFIES[39] representation of fragments extracted from high-rewarding molecules. The fragments are randomly recombined and a mutation operation is applied with some predefined probability. A graph convolutional neural network is trained to imitate a given scoring function of interest, allowing us to compute atom-wise attributions with respect to the predicted score for any queried molecule. Substructures of molecules where the sum of the atom-wise attributions contribute positively to the predicted score are kept in a fragment library from which the genetic expert samples and recombines fragments. We apply selection pressure on the fragment library by only keeping fragments from high-scoring molecules.

The proposed framework's contribution can be summarized as follows:

1. We propose a fragment-based genetic expert that requires no handcrafted rules. Fragments are extracted from molecules using an explainability method and a graph convolutional network.

2. We present an in-depth ablation study to investigate the importance of different genetic experts and their combination.

More specifically, we show that (1) the proposed genetic expert is much more robust and could circumvent certain pitfalls proposed by Ahn et al. and (2) the proposed framework is able to leverage beyond a single genetic expert, leaving higher possibility for the model to explore and exploit certain genetic experts.

2 Methods

In this section, we discuss the overall framework as well as our proposed *InFrag* expert and other mechanisms in further details. We first discuss how the method extracts fragments from arbitrary molecules based on the computed attribution for each involved atom. Next, we show how the obtained fragments are recombined to propose novel candidate molecules. Finally, we demonstrate how this expert can be integrated into the GEGL framework to either replace or alternatively be combined with other genetic experts. We denote the complete framework as *eGEGL* for *enhanced GEGL*. The extended framework is depicted in Figure 1.

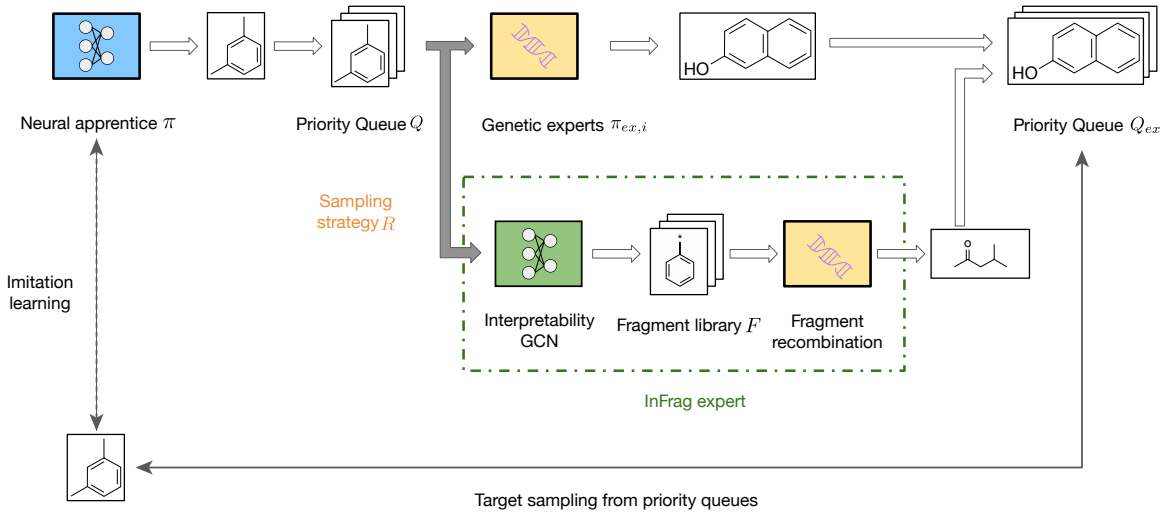


Figure 1: Complete depiction of the *eGEGL* framework. The original framework is enhanced with a fragment-based genetic expert denoted as *InFrag*. *InFrag* consists of a GCN model which creates attribution for queried molecules from which fragments are extracted. The extracted fragments are then randomly recombined to obtain novel candidate molecules. In addition, we include the possibility to leverage several experts via a chosen sampling strategy R which determines how the sampling budget is to be allocated to each expert.

2.1 Molecular representation

Molecules are represented as undirected graphs $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with a total number of N nodes $v_i \in \mathcal{V}$ representing the atoms and edges between nodes denoted as $e_{i,j} \in \mathcal{E}$ representing the

bonds. Furthermore, we extract the binary Adjacency matrix $\mathcal{A} \in \mathbb{R}^{N \times N}$ from the edges. Each node is represented as a 74-dimensional vector $\boldsymbol{n_i} \in \mathbb{R}^{74}$ containing multiple distinct features. More details about the meanings of the used features can be found in supplementary material Table S2.

2.2 Attribution-based fragment generation

We are motivated by understanding which parts of a molecule are contributing towards its property score. Attribution-based model interpretability methods allow us to reason between model input features and predicted scores. Such attributions cannot be directly computed from the optimization scoring function which is assumed to be an available but black-box function used to evaluate the fitness of candidate molecules. To overcome this issue, we train a pseudo-scoring function that is encouraged to imitate the original black-box scoring function of the optimization task, hence allowing us to obtain the desired attributions through this surrogate model. We implement a Graph Convolutional Network (GCN)[35] that takes the above-mentioned node features and adjacency matrix as inputs and is trained to predict task scores. Please refer to supplementary materials Table S1 for a detailed description of all used hyperparameters and their values.

To generate attributions, we use the Class Activation Maps (CAM)[32] method due to its implementation simplicity and performance. More precisely , we compute the attribution for each atom via

$$attribution(atom_i) = w_{out}^T \cdot h_i, \tag{1}$$

where w_{out} corresponds to the weights of the final dense layer of the GCN after the average pooling layer and h_i represents the latent node features of the node corresponding to atom i just before the pooling layer. Sanchez-Lengeling et al. demonstrated that CAM is able to perform relatively well, especially when combined with GCN, compared to other interpretability methods in the context of graph neural networks.[40] We note that although we did not

experiment with other attribution methods and graph neural networks, other methods and models can be used to generate the necessary attributions.

Fragments of interest are identified by comparing the sign of the attribution for each atom involved in a given bond. In the case where the signs are found to be opposite, we fragment the molecule along that bond. To preserve complex substructures, we only consider and fragment single bonds and bonds which are not aromatic (as determined by RDKit[41]). We then proceed to sum up the individual attributions of all the atoms in the fragment. If that sum is greater than 0.0, we consider the fragment to contribute positively to the predicted score and keep that fragment. Otherwise, the fragment is discarded.

Our objective is to keep and recombine fragments from high-rewarding molecules only. After processing all of the queried molecules as described above, we end up with fragments contributing positively to the predicted score regardless of its value. In order to sort and only consider high-rewarding fragments, we associate each fragment with the score from the black-box scoring function of the molecule it originated from. All fragments are added to a data-structure we call the fragment library. The fragment library allows to memorize the best scoring fragments and we impose uniqueness of the fragments in the library. When a new fragment is being added, we first check if the fragment is already present in memory. In case the fragment is already present, we set the score associated with the fragment to the maximum score between the original in-memory fragment and the new duplicate fragment. We apply selection pressure on the fragment library by limiting it's memory size to 1,000 fragments. This operation allows to remove low-rewarding fragments and leads the memory to only contain high-rewarding fragments which can be used in future optimization rounds. The complete process of fragment generation is graphically depicted in Figure 2.

Like the neural apprentice, the GCN model is trained every epoch such that it can reason over newly discovered and potentially better molecules. We also experimented with the setting where the weights of the GCN model are fixed and not updated during optimization.

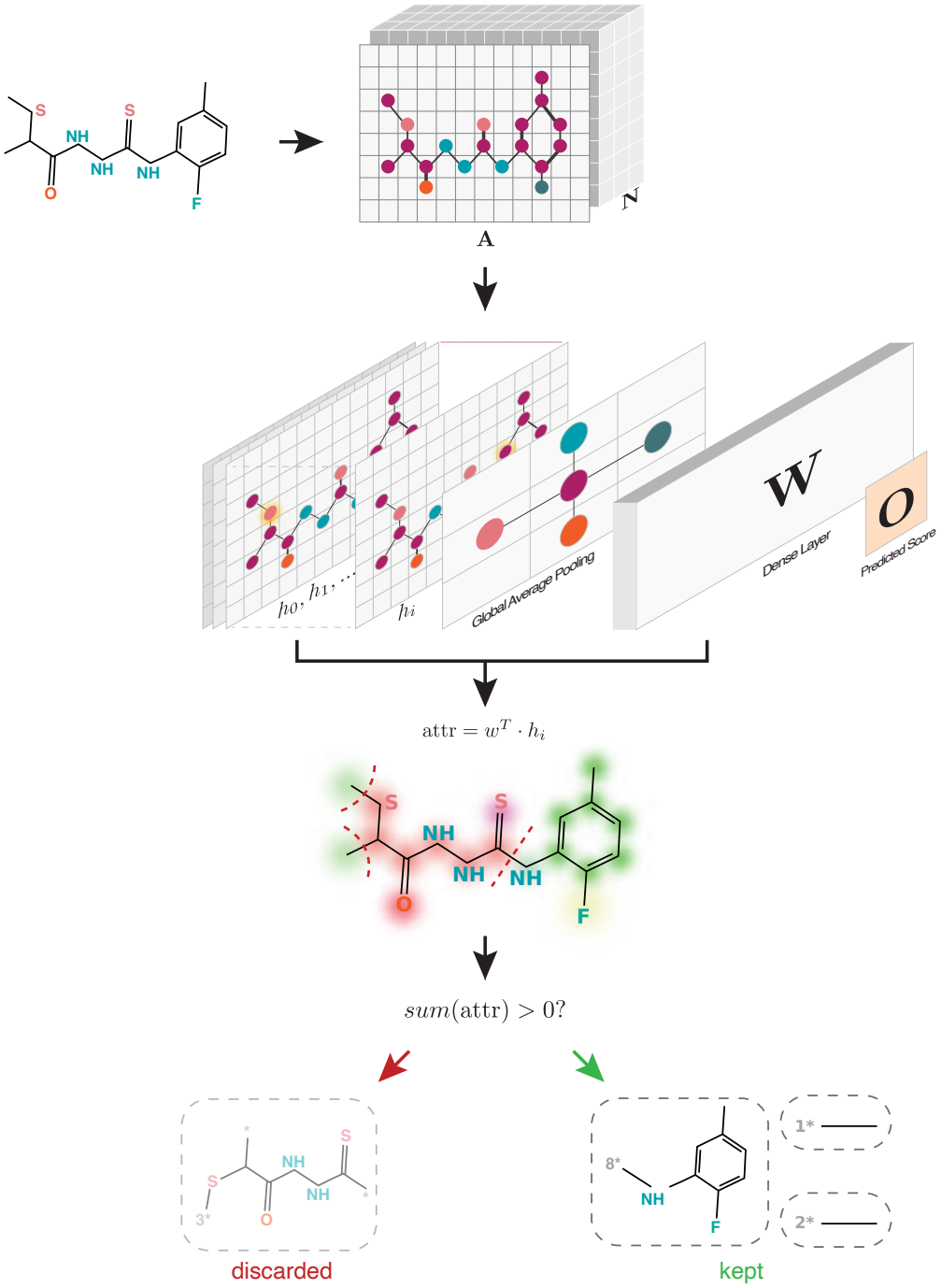


Figure 2: Generation of fragments using interpretability methods. A molecule is first processed to extract the adjacency matrix and node features. A graph convolutional network is used to calculate the last latent node embedding and final dense layer weights to compute the atom-wise class activation maps. The molecule is then fragmented along bonds where the sign of the attributions are opposite to each other. Only fragments contributing positively to the score, as determined by taking the sum over all involved atoms, are kept (depicted by the green arrow) and considered in further steps.

2.3 Fragment recombination

To generate novel candidate molecules with our genetic expert, we first collect all fragments of the fragment library which are in SMILES format. New molecules are generated by randomly sampling 2 to 5 fragments and translating the sampled fragments into a SELFIES representation. Selfies representations posses the convenient property to always be valid, making their use very convenient for the fragment-crossover operation. The sampled fragments are randomly shuffled and combined to obtain a novel candidate SELFIES string from this fragment-crossover procedure. With some predefined probability, we furthermore apply a mutation operation on the obtained string in order to increase exploration and diversity of the generated molecules. Mutations include deletion, insertion or exchange of one of the SELFIES token on the currently operated string. Each type of mutation has the same probability of occurrence. This generative process is repeated until the number of desired molecules is reached. We note that the SELFIES representation makes use of an internal state when translating from the conventional SMILES representation. This implies that one might require a more complex recombination strategy compared to the one described above to obtain better crossover molecules from the sampled fragments. We empirically found that the simple recombination strategy as described above performs competitively, regardless of the fact that the number of atoms is not necessarily preserved during the crossover operation as illustrated in Figure 3. This representation-blind recombination strategy could be used in other problem settings since no assumptions are made about the task or the underlying representation. In addition, it does not require us to define complicated recombination rules and include expert knowledge on how fragments should relate to each other. This minimizes the potential for biases induced by such expert rules.

2.4 GEGL with multiple genetic experts

An additional proposal we make to improve overall performance is to simultaneously leverage multiple genetic experts during molecular optimization. We motivate this approach by the

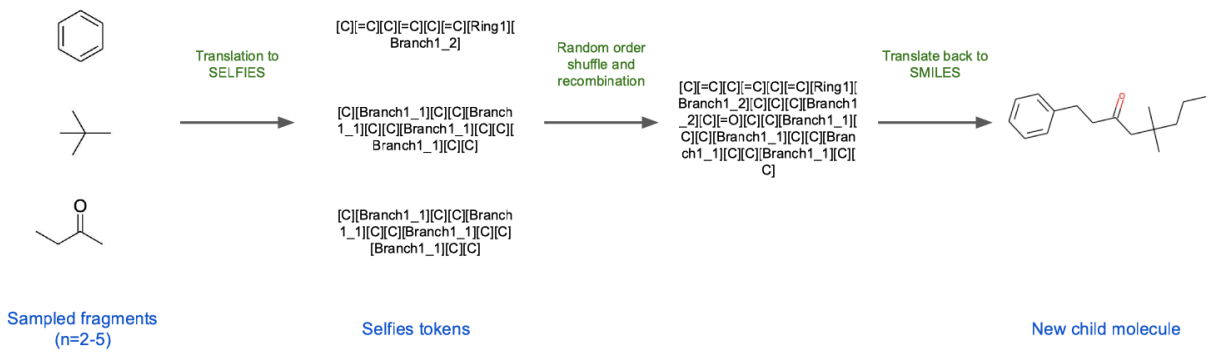


Figure 3: Recombination method for high-rewarding fragments. Fragments are sampled from the fragment library which contains high-rewarding fragments only. The fragments are first translated into their respective SELFIES representation, randomly shuffled and recombined. The final SELFIES token string is translated back to a SMILES representation and the resulting string is the new candidate child molecule resulting from the fragments. Please note that the proposed crossover rule is highly general and unbiased as it makes no assumptions about the used molecular representation.

observation that it is not readily obvious what type of genetic expert, and hence inductive biases related to it, is going to perform well on a given optimization task. Therefore, we allow for multiple genetic experts to be queried while maintaining the total sampling budget for a fair comparison. We evaluate several sampling strategies, denoted as R in Figure 1, to determine the sampling proportion for each expert in each optimization round:

- Fixed: We keep the proportion for each expert fixed and equal for each expert. The obtained distribution is essentially a discrete uniform distribution u.

- Softmax: This strategy first computes the ratio of each expert in the expert memory r_i and passes these values through a softmax function to squeeze preference towards high-performing experts. The probability of choosing expert i out of all K experts to sample a new candidate molecule is calculated via

$$prob(i) = \frac{\exp(r_i)}{\sum_{j=0}^{K} \exp(r_j)} \tag{2}$$

- Rebalancing: In this method, we first determine the improvement or deterioration of each expert compared to the previous optimization round which we denote δ_i. The new distribution for each expert i at time t is calculated as:

$$prob(i,t) = (prob(i,t-1) + \delta_i) + 0.2 * (u_i - (prob(i,t-1) + \delta_i)) \tag{3}$$

where u_i is the discrete uniform probability mass. This method ensures that no expert dominates the sampling process for too many optimization rounds and as the name implies re-calibrates the sampling proportions back to a uniform distribution. The shift back to a uniform distribution can easily be seen if the change in best performing experts δ_i is set to a value of 0.

A pseudo-code algorithm of this multi-expert framework and the differences to the original GEGL-framework are provided in algorithm 1. We furthermore illustrate the complete *eGEGL* framework (*InFrag* + multiple genetic experts) in Figure 1.

Algorithm 1 Multi-expert Genetic expert-guided learning pseudocode. Differences to the original algorithm are shown in red.

Require: Number of genetic experts N, Sampling budget M, sampling strategy R

Set $Q \leftarrow \emptyset$, $Q_{ex} \leftarrow \emptyset$ ▷ Initialize the max-reward priority queues

Set $S \leftarrow uniform(n) = \frac{1}{N}$ ▷ Initialize the sampling distribution for experts

for $t = 1, ..., T$ **do** ▷ Optimize for T rounds

  for m=1,..,M **do** ▷ Samples from neural apprentice

    Update $Q \leftarrow Q \cup \mathbf{x}$ where $\mathbf{x} \sim \pi(\mathbf{x}; \theta)$

    If $|Q| > K$, update $Q \leftarrow Q \setminus \{\mathbf{x}_{min}\}$, where $\mathbf{x}_{min} = argmin_{x \in Q} r(\mathbf{x})$

  end for

  for m=1,...,M **do** ▷ Sample from the genetic experts

    $id \sim S$ ▷ Determine from which expert to sample from

    Update $Q_{ex} \leftarrow Q_{ex} \cup \mathbf{x}$ where $\mathbf{x} \sim \pi_{ex,id}(\mathbf{x}; \theta)$

    If $|Q_{ex}| > K$, update $Q_{ex} \leftarrow Q_{ex} \setminus \{\mathbf{x}_{min}\}$, where $\mathbf{x}_{min} = argmin_{x \in Q_{ex}} r(\mathbf{x})$

  end for

  Update $S \leftarrow R(S)$ ▷ Update sampling distribution according to strategy

  Maximize $\sum_{\mathbf{x} \in Q \cup Q_{ex}} \log \pi(\mathbf{x}; \theta)$ over θ

end for

return $Q \cup Q_{ex}$ ▷ Output best generated molecules

3 Results and Discussion

3.1 Baselines and benchmarks

We benchmark and report performances of the enhanced framework against the original framework. In addition, we also add a baseline consisting of a simplified version of the STONED genetic operators proposed by Nigam et al.[42] which we will denote as *simplified-STONED* henceforth. More precisely, we limit the number of sampled chemical paths between any two parent molecules to a single one for this genetic expert. For other tasks, we include and compare against results obtained for selected baselines as reported in the original GEGL paper unless noted otherwise.

As for benchmarks, we follow the original GEGL work and compared the trained models on the penalized logp task and a subset goal-directed Guacamol benchmarks which we will describe in more details below. Furthermore, we used the same pretrained neural apprentice LSTM model as described in the original work.

Penalized logp is a standard benchmark to evaluate *de novo* generative methods. The objective is to maximize the penalized octanol-water partition coefficient score defined as:

$$PenalizedLogP(x) = LogP(x) - SyntheticAccessibility(x) - RingPenalty(x) \quad (4)$$

where *LogP* is the unpenalized octanol-water partition coefficient,[9] *SyntheticAccessibility* is a penalty term accounting for synthesizability[43] and *RingPenalty* is a penalty for rings with a size larger than 6. We impose a constraint onto the generative model by limiting the number of SMILES characters to 81 following previous work.

Goal-directed Guacamol benchmarks are a set of 20 benchmarks proposed by Brown et al.[44] which were specifically designed for comparing generative models. The benchmarks evaluate a set of molecules to account for molecular diversity. More specifically and following the notation in Ahn et al.,[37] the final benchmark score for a given molecule set X is computed

as

$$Guacamol(X) \coloneqq \sum_{S \in Q} \sum_{s=1}^{S} \frac{r(x_{\Pi(s)})}{S|Q|} \text{ for } s = 1, ..., |X| - 1 \tag{5}$$

where Q is a list of integers and Π is a permutation function which ensures that the molecules $x \in X$ are sorted in descending order with respect to their respective scores. The goal-directed benchmarks contain a variety of tasks such as *rediscovery*, *similarity* or *multi property optimization*. We refer readers to either the original Guacamol and GEGL papers for a more detailed description of each task type.[37,44] For the tested tasks, we limit the number of characters to 100 for all generated SMILES strings.

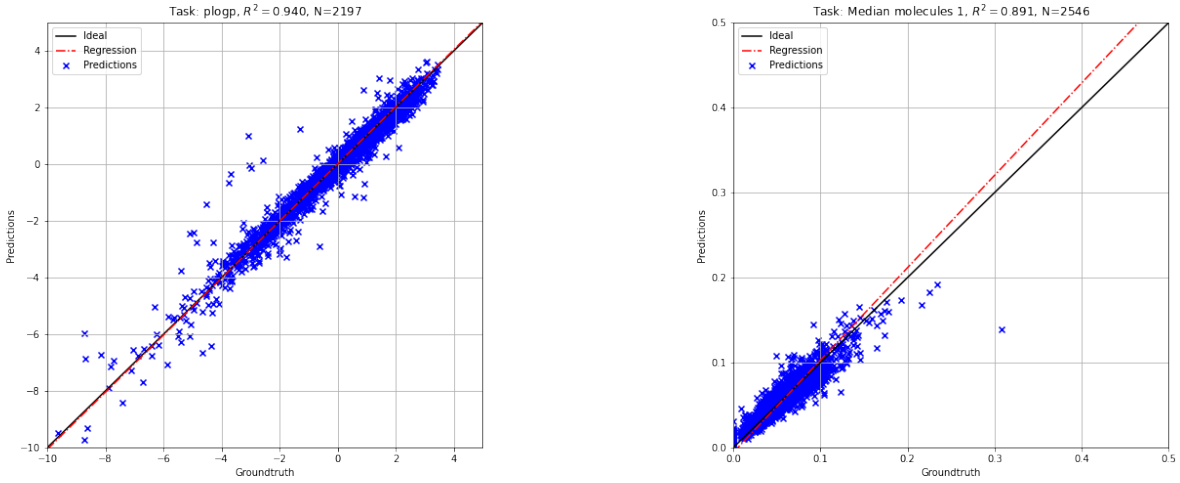
3.2 GCN pretraining

One core assumption and requirement of our fragment-based genetic expert is that it is possible to leverage a GCN model which is able to approximate an arbitrary scoring function of interest. Therefore, we start by first designing a simple experiment to ensure that the GCN model which functions as a pseudo-scorer is able to mimic the true scoring function and thus can be used as as proxy model for it when calculating the atom-wise attributions. For each of the tasks described below, we checked the capability of the scoring function model to imitate the oracle function.

Similar to the neural apprentice in the original GEGL framework, the GCN model is first pretrained either on the ZINC[45] or ChEMBL[46] dataset. We formulate the problem as a regression task where the model is tasked to predict the oracle score given by the original scoring function. Hyperparameters for the model and training can be found in the supplementary material Table S1. We did not tune the hyperparameters of the model or other training parameters as the model empirically is reasonably able to predict the true scores as shown below. Figure 4 displays the comparison between the predictions by the trained model versus the ground-truth for the *plogp* task and the Guacamol goal-directed benchmark *Median molecule 1* as described in sections 3.3 and 3.4. Figures for additional

tasks can be found in the supplementary material Figures S1-S9.

The obtained results indicate that the learning of useful representations is heavily task-dependent. For example, the graph neural network model wasn't able to learn meaningful representations for both *Isometry* tasks of the Guacamol benchmark. Results shown below demonstrate that the eGEGL framework making use of the fragment-based genetic expert is nonetheless able to achieve competitive performances compared to the best method.



(a) *Penalized logp task* (b) *Median Molecules 1*

Figure 4: Comparison between the groundtruth score and predicted score by a pretrained GCN model.

3.3 Penalized logP task

Table 1 summarizes the mean and standard deviation comparison between several reported baselines and the enhanced GEGL framework we propose. We ran 5 independent experimental runs for each of our models where each round consists of 200 optimization rounds following previous work. The results demonstrate that there is no gain or decrease in performance by swapping out the genetic expert. This is remarkable because it indicates that expert-designed genetic experts can potentially be replaced with simpler and bias-free genetic-experts without loss in performance.

Table 1: PenalizedLogP results. Results are displayed as mean and standard deviation.

Algorithm	Objective
JT-VAE[20]	4.90 ± 0.33
ChemTS[21]	5.60 ± 0.50
GCPN[19]	7.86 ± 0.07
GB-GA[38]	15.76 ± 5.76
DA-GA[47]	20.72 ± 3.14
GEGL (GB-GA expert)[37]	$\mathbf{31.40 \pm 0.00}$
GEGL (simplified-STONED expert)	$\mathbf{31.40 \pm 0.00}$
GEGL (InFrag expert)	$\mathbf{31.40 \pm 0.00}$

We furthermore note that the *InFrag* genetic expert is able to extract useful information from the pretraining phase of the GCN which can be leveraged from the start of the optimization process. To demonstrate this, we additionally compared the different genetic experts in the setting where obtaining and evaluating candidate molecules is assumed to be undesirable. Concretely, we limit the optimization process to a single and three optimization rounds respectively and evaluate the generated molecules. As can be seen in Table 2, our *InFrag* genetic expert is able to generate higher-rewarding molecules in this setting, even surpassing the *JT-VAE* and *ChemTS* baselines as can be seen from Table 1 and achieving similar results to the reinforcement learning based *GCPN* method. These results indicate that our genetic expert is appropriate in settings where obtaining *additional* labels for novel candidate molecules is expensive or difficult, as is usually the case in real-world applications. This is to be expected as the GCN is able to leverage the learned representations from the pretraining phase and therefore can reason and propose high-rewarding novel molecular candidates from the beginning of the optimization process.

3.4 Goal-directed guacamol benchmarks

Finally, we evaluated the proposed enhancements and genetic expert on a subset of the goal-directed benchmark tasks proposed in the Guacamol benchmark. More precisely, we excluded all *Rediscovery* and all *Similarity* tasks for which multiple baselines are able to

Table 2: One- and Three-round PenalizedLogP results. Entries are displayed as mean and standard deviation.

Algorithm	1-round optimization	3-round optimization
GEGL (GB-GA expert)[37]	5.44 ± 0.17	7.92 ± 0.42
GEGL (simplified-STONED expert)	4.29 ± 0.14	6.82 ± 0.25
GEGL (InFrag expert)	$\mathbf{7.81 \pm 0.61}$	$\mathbf{9.50 \pm 0.24}$
Improvement over baseline	**43.6**%	**19.9**%

obtain a perfect score of 1.0. Table 3 summarizes the obtained experimental results.

Table 3: Results for the goal-directed Guacamol benchmarks for different genetic experts leveraging the GEGL framework. We show selected baselines for comparison as reported by the original GEGL work.[37]

	Baselines			GEGL-based models		
Task	GB-GA[38]	MSO[48]	CReM[22]	GB-GA[37]	simplified-STONED	InFrag
C11H24	0.971	0.997	0.966	**1.000**	**1.000**	**1.000**
C9H10N2O2PF2Cl	0.982	**1.000**	0.940	**1.000**	**1.000**	**1.000**
Median molecules 1	0.406	0.437	0.371	**0.455**	**0.455**	**0.455**
Median molecules 2	0.432	0.395	0.434	**0.437**	0.419	0.427
Osimertinib MPO	0.953	0.966	0.995	**1.000**	**1.000**	**1.000**
Fexofenadine MPO	0.998	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
Ranolazine MPO	0.920	0.931	0.969	0.958	**0.981**	0.959
Perindopril MPO	0.792	0.834	0.815	0.882	**0.886**	0.882
Amlodipine MPO	0.894	0.900	0.902	**0.924**	0.905	0.905
Sitagliptin MPO	0.891	0.868	0.763	0.922	**0.958**	0.952
Zaleplon MPO	0.754	0.764	0.770	0.834	**0.840**	**0.840**
Valsartan SMARTS	0.990	0.994	0.994	**1.000**	**1.000**	**1.000**
Deco Hop	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
Scaffold Hop	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**

We observe that the genetic experts can be exchanged with each other without significant loss or increase in performance. Furthermore, the empirical results show that no expert dominates across all of the tested tasks. This indicates that combining and sampling several experts simultaneously to leverage the best of all experts can potentially lead to better overall results. Next, we present the results for the GEGL framework when using multiple experts as explained in section 2.4 in table 4. Experiments were conducted with the described sampling strategies. We furthermore report results for freezing the GCN model weights versus updating

them during the optimization process.

Table 4: Results for non-trivial goal-directed Guacamol benchmarks and different sampling strategies when using multiple experts.

GCN type	Frozen			Updating		
Sampling strategy Task	Fixed	Softmax	Rebalancing	Fixed	Softmax	Rebalancing
C11H24	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
C9H10N2O2PF2Cl	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
Median molecules 1	**0.455**	**0.455**	**0.455**	**0.455**	**0.455**	**0.455**
Median molecules 2	0.423	**0.429**	0.422	0.426	0.427	0.427
Osimertinib MPO	0.996	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
Fexofenadine MPO	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
Ranolazine MPO	0.976	**0.977**	0.962	0.967	0.967	0.962
Perindopril MPO	0.882	**0.886**	**0.886**	**0.886**	**0.886**	**0.886**
Amlodipine MPO	**0.924**	**0.924**	0.905	0.905	0.905	0.905
Sitagliptin MPO	**0.969**	0.967	0.956	0.965	0.957	0.956
Zaleplon MPO	**0.840**	**0.840**	0.834	**0.840**	0.834	0.834
Valsartan SMARTS	**1.000**	**1.000**	0.999	0.999	**1.000**	0.999
Deco Hop	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**
Scaffold Hop	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**	**1.000**

We observe that generally there is no significant difference between each experimental setting. Moreover, the obtained results demonstrate that it is possible for multiple experts to maintain and even retrieve performances from the single genetic expert setting. Notably, the *Softmax-sampling/frozen GCN* is able to perform well across all tasks and only shows lower performances compared to the original GEGL framework in the *Median molecules 2* benchmark.

4 Conclusions

We have shown in this work that it is possible to pretrain and utilize an attribution-based genetic expert to propose novel molecules. The expert we called *InFrag* is able to leverage a pretraining phase to reason over the potential score of candidate molecules and produces atom-wise attributions to generate high-rewarding fragments. Our genetic expert which is

capable of expert- and bias-free fragment-level crossover operations is able to produce high-rewarding molecules and performs comparable to other genetic experts when embedded into the GEGL framework. Furthermore, we demonstrated that *InFrag* significantly outperforms other genetic experts when the number of optimization rounds is limited. This implies that *InFrag* is a good choice for real-world optimization cases were evaluation of novel generated molecules might be difficult or expensive. We have not yet experimented with different genetic experts, attribution methods, graph neural network architectures or sampling strategies, leaving a potentially large room for improvement. We leave this area of research for future work.

5 Data and Software Availability

The source code to replicate the experiments is available at `https://github.com/elix-tech/infrag`. The README-file of the repository contains instructions on how to download the ZINC and Guacamol datasets as well as running optimization scripts.

Acknowledgement

The authors would like to thank Casey Galvin, David Jimenez, Haris Hasic, Joshua Owoyemi, Laurent Dillard, Nazim Medzhidov and Romeo Cozac for their valuable insights and feedback during research discussions.

The icon used for the genetic experts in Figure 1 was downloaded from *freeicons.io* and designed by User *Nine One*.

Supporting Information Available

Experimental hyperparameters, GCN input features, GCN pretraining results

References

(1) Schneider, G. Automating Drug Discovery. *Nature Reviews Drug Discovery* **2018**, *17*, 97–113.

(2) Sanchez-Lengeling, B.; Aspuru-Guzik, A. Inverse Molecular Design Using Machine Learning: Generative Models for Matter Engineering. *Science* **2018**, *361*, 360–365.

(3) Pollice, R.; Dos Passos Gomes, G.; Aldeghi, M.; Hickman, R. J.; Krenn, M.; Lavigne, C.; Lindner-D'Addario, M.; Nigam, A.; Ser, C. T.; Yao, Z.; Aspuru-Guzik, A. Data-Driven Strategies for Accelerated Materials Design. *54*, 849–860.

(4) Bohacek, R. S.; McMartin, C.; Guida, W. C. The Art and Practice of Structure-Based Drug Design: A Molecular Modeling Perspective. *Medicinal Research Reviews* **1996**, *16*, 3–50.

(5) Coley, C. W.; Eyke, N. S.; Jensen, K. F. Autonomous Discovery in the Chemical Sciences Part I: Progress. *Angewandte Chemie International Edition* **2020**, *59*, 22858–22893.

(6) Hartenfeller, M.; Schneider, G. Enabling Future Drug Discovery by de Novo Design. *WIREs Computational Molecular Science* **2011**, *1*, 742–759.

(7) Meyers, J.; Fabian, B.; Brown, N. De Novo Molecular Design and Generative Models. *Drug Discovery Today* **2021**,

(8) Weininger, D. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *28*, 31–36.

(9) 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 central science* **2018**, *4*, 268–276.

(10) Segler, M. H. S.; Kogej, T.; Tyrchan, C.; Waller, M. P. Generating Focused Molecule Libraries for Drug Discovery with Recurrent Neural Networks. *ACS Central Science* **2018**, *4*, 120–131.

(11) Kusner, M. J.; Paige, B.; Hernández-Lobato, J. M. Grammar Variational Autoencoder. International Conference on Machine Learning. 2017; pp 1945–1954.

(12) Dai, H.; Dai, B.; Song, L. Discriminative Embeddings of Latent Variable Models for Structured Data. PMLR. 2016; pp 2702–2711.

(13) Janz, D.; van der Westhuizen, J.; Hernández-Lobato, J. M. Actively Learning What Makes a Discrete Sequence Valid. *arXiv:1708.04465 [cs, stat]* **2017**,

(14) 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:1705.10843 [cs, stat]* **2018**,

(15) Blaschke, T.; Arús-Pous, J.; Chen, H.; Margreitter, C.; Tyrchan, C.; Engkvist, O.; Papadopoulos, K.; Patronov, A. REINVENT 2.0: An AI Tool for De Novo Drug Design. *Journal of Chemical Information and Modeling* **2020**, *60*, 5918–5922.

(16) Simonovsky, M.; Komodakis, N. GraphVAE: Towards Generation of Small Graphs Using Variational Autoencoders. *arXiv:1802.03480 [cs]* **2018**,

(17) You, J.; Ying, R.; Ren, X.; Hamilton, W. L.; Leskovec, J. GraphRNN: Generating Realistic Graphs with Deep Auto-Regressive Model. Proceedings of the 35th International Conference on Machine Learning. 2018.

(18) Li, Y.; Vinyals, O.; Dyer, C.; Pascanu, R.; Battaglia, P. Learning Deep Generative Models of Graphs. *arXiv preprint arXiv:1803.03324* **2018**,

(19) You, J.; Liu, B.; Ying, Z.; Pande, V.; Leskovec, J. Graph Convolutional Policy Network for Goal-Directed Molecular Graph Generation. Advances in Neural Information Processing Systems. 2018.

(20) Jin, W.; Barzilay, R.; Jaakkola, T. Junction Tree Variational Autoencoder for Molecular Graph Generation. International Conference on Machine Learning. 2018; pp 2323–2332.

(21) Yang, X.; Zhang, J.; Yoshizoe, K.; Terayama, K.; Tsuda, K. ChemTS: An Efficient Python Library for de Novo Molecular Generation. *Science and Technology of Advanced Materials* **2017**, *18*, 972–976.

(22) Polishchuk, P. CReM: chemically reasonable mutations framework for structure generation. *12*, 28.

(23) Vinkers, H. M.; de Jonge, M. R.; Daeyaert, F. F. D.; Heeres, J.; Koymans, L. M. H.; van Lenthe, J. H.; Lewi, P. J.; Timmerman, H.; Van Aken, K.; Janssen, P. A. J. SYNOPSIS: SYNthesize and OPtimize System in Silico. *Journal of Medicinal Chemistry* **2003**, *46*, 2765–2773.

(24) Spiegel, J. O.; Durrant, J. D. AutoGrow4: An Open-Source Genetic Algorithm for de Novo Drug Design and Lead Optimization. *Journal of Cheminformatics* **2020**, *12*, 25.

(25) Jin, W.; Coley, C.; Barzilay, R.; Jaakkola, T. Predicting Organic Reaction Outcomes with Weisfeiler-Lehman Network. *Advances in Neural Information Processing Systems* **2017**, *30*.

(26) Lowe, D. M. Extraction of chemical structures and reactions from the literature. Publisher: Apollo - University of Cambridge Repository.

(27) Bradshaw, J.; Paige, B.; Kusner, M. J.; Segler, M.; Hernández-Lobato, J. M. In *Advances in Neural Information Processing Systems 32*; Wallach, H., Larochelle, H.,

Beygelzimer, A., d\textquotesingle Alché-Buc, F., Fox, E., Garnett, R., Eds.; Curran Associates, Inc., 2019; pp 7937–7949.

(28) 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 Central Science* **2019**, *5*, 1572–1583.

(29) Korovina, K.; Xu, S.; Kandasamy, K.; Neiswanger, W.; Poczos, B.; Schneider, J.; Xing, E. ChemBO: Bayesian Optimization of Small Organic Molecules with Synthesizable Recommendations. International Conference on Artificial Intelligence and Statistics. 2020; pp 3393–3403.

(30) Selvaraju, R. R.; Cogswell, M.; Das, A.; Vedantam, R.; Parikh, D.; Batra, D. Grad-CAM: Visual Explanations From Deep Networks via Gradient-Based Localization. Proceedings of the IEEE International Conference on Computer Vision (ICCV). 2017.

(31) Srinivas, S.; Fleuret, F. Full-Gradient Representation for Neural Network Visualization.

(32) Zhou, B.; Khosla, A.; Lapedriza, A.; Oliva, A.; Torralba, A. Learning Deep Features for Discriminative Localization. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016; pp 2921–2929.

(33) Zhang, J.; Bargal, S. A.; Lin, Z.; Brandt, J.; Shen, X.; Sclaroff, S. Top-Down Neural Attention by Excitation Backprop. *126*, 1084–1102.

(34) Pope, P. E.; Kolouri, S.; Rostami, M.; Martin, C. E.; Hoffmann, H. Explainability Methods for Graph Convolutional Neural Networks. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2019; pp 10764–10773.

(35) Kipf, T. N.; Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. Proceedings on the 5th International Conference on Learning Representations. 2017.

(36) Jin, W.; Barzilay, D. R.; Jaakkola, T. Multi-Objective Molecule Generation Using Interpretable Substructures. International Conference on Machine Learning. 2020; pp 4849–4859.

(37) Ahn, S.-S.; Kim, J.; Lee, H.; Shin, J. Guiding Deep Molecular Optimization with Genetic Exploration. *Advances in Neural Information Processing Systems* **2020**, *33*, 12008–12021.

(38) Jensen, J. H. A Graph-Based Genetic Algorithm and Generative Model/Monte Carlo Tree Search for the Exploration of Chemical Space. *Chemical Science* **2019**, *10*, 3567–3572.

(39) Krenn, M.; Häse, F.; Nigam, A.; Friederich, P.; Aspuru-Guzik, A. Self-Referencing Embedded Strings (SELFIES): A 100% Robust Molecular String Representation. *Machine Learning: Science and Technology* **2020**, *1*, 045024.

(40) Sanchez-Lengeling, B.; Wei, J.; Lee, B.; Reif, E.; Wang, P.; Qian, W.; McCloskey, K.; Colwell, L.; Wiltschko, A. Evaluating Attribution for Graph Neural Networks. *Advances in Neural Information Processing Systems* **2020**, *33*, 5898–5910.

(41) Landrum, G. et al. rdkit/rdkit: 2020_09_5 (Q3 2020) Release. `https://zenodo.org/record/4570805`.

(42) Nigam, A.; Pollice, R.; Krenn, M.; Gomes, G. d. P.; Aspuru-Guzik, A. Beyond generative models: superfast traversal, optimization, novelty, exploration and discovery (STONED) algorithm for molecules using SELFIES. *12*, 7079–7090.

(43) Ertl, P.; Schuffenhauer, A. Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *1*, 8.

(44) Brown, N.; Fiscato, M.; Segler, M. H.; Vaucher, A. C. GuacaMol: Benchmarking Models for de Novo Molecular Design. *59*, 1096–1108.

(45) Irwin, J. J.; Sterling, T.; Mysinger, M. M.; Bolstad, E. S.; Coleman, R. G. ZINC: A Free Tool to Discover Chemistry for Biology. *52*, 1757–1768.

(46) Gaulton, A.; Bellis, L. J.; Bento, A. P.; Chambers, J.; Davies, M.; Hersey, A.; Light, Y.; McGlinchey, S.; Michalovich, D.; Al-Lazikani, B.; Overington, J. P. ChEMBL: a large-scale bioactivity database for drug discovery. *40*, D1100–D1107.

(47) Nigam, A.; Friederich, P.; Krenn, M.; Aspuru-Guzik, A. Augmenting Genetic Algorithms with Deep Neural Networks for Exploring the Chemical Space. 2020.

(48) Winter, R.; Montanari, F.; Steffen, A.; Briem, H.; Noé, F.; Clevert, D.-A. Efficient multi-objective molecular optimization in a continuous latent space. *Chem. Sci.* **2019**, *10*, 8016–8024.

Graphical TOC Entry