Generative Model for Small Molecules with Latent Space RL Fine-Tuning to Protein Targets

Ulrich A. Mbou Sob [*1] **Qiulin Li** [*1 2] **Miguel Arbesú** [1] **Oliver Bent** [1] **Andries P. Smit** [1] **Arnu Pretorius** [1]

Abstract

A specific challenge with deep learning approaches for molecule generation is generating both syntactically valid and chemically plausible molecular string representations. To address this, we propose a novel generative latent-variable transformer model for small molecules that leverages a recently proposed molecular string representation called SAFE. We introduce a modification to SAFE to reduce the number of invalid fragmented molecules generated during training and use this to train our model. Our experiments show that our model can generate novel molecules with a validity rate $> 90\%$ and a fragmentation rate $< 1\%$ by sampling from a latent space. By fine-tuning the model using reinforcement learning to improve molecular docking, we significantly increase the number of hit candidates for five specific protein targets compared to the pre-trained model, nearly doubling this number for certain targets. Additionally, our top 5% mean docking scores are comparable to the current state-of-the-art (SOTA), and we marginally outperform SOTA on three of the five targets.

1. Introduction

De novo drug design is the process of designing novel chemicals with desired pharmacological properties. This process is time-consuming and requires screening numerous compounds to find potential drug candidates (Bohacek et al., 1996). The chemical space for potential drugs is vast, up to $\sim 10^{60}$ (Polishchuk et al., 2013), and sparse, making exploration challenging. Deep learning methods for drug design aim to optimize the search process and reduce the time required for drug discovery and development (see Atance et al. (2022) and Lee et al. (2023)).

Deep learning approaches for molecule generation can be categorized into three groups: *sequence* based, *molecular graphs*, and *structure* based approaches (Du et al., 2022). Molecular graphs and structure-based approaches use graph and equivariant (Satorras et al., 2021) neural networks, generally following a lead-based generation strategy, where new compounds are created by expanding or altering an existing core molecular fragment of known desirable properties with smaller functional groups — i.e. sampling the chemical space around a scaffold. Examples of this approach include Madhawa et al. (2019), Zang & Wang (2020), Luo et al. (2021), Bongini et al. (2021), Jiang et al. (2021), and Powers et al. (2022). On the other hand, sequence-based approaches utilize molecular string representations such as SMILES (Weininger, 1988) and rely on sequence-based learning models like recurrent neural networks and transformers (see Olivecrona et al. (2017), Podda et al. (2020), and Irwin et al. (2022)). Lead-based generation strategies used in molecular and structure-based models are limited in their ability to generate molecules with completely new scaffolds and thus in diversity of the generated molecules. Sequence-based models, on the other hand, struggle to generate both syntactically valid representations and chemically plausible molecules with desired drug-like properties. For instance, deep learning models trained with SMILES often generate a high rate of syntactically invalid molecules, while models trained with SELFIES (Krenn et al., 2020) tend to produce implausible molecules that typically consist of very long chains and large rings (see Tarasov et al. (2023) and Noutahi et al. (2024)).

In this paper, we introduce a new generative latent-variable model for small molecules. This model consists of a variational auto-encoder (VAE) embedded within an encoder-decoder transformer architecture. We train our model using a novel molecular representation called **SAFER** which is based on the recently introduced **SAFE** representation (Noutahi et al., 2024). Additionally, we perform various experiments with this new architecture and apply reinforcement learning (RL) fine-tuning to generate potential hit candidates with high molecular docking scores to the same targets presented in Lee et al. (2023).

[*]Equal contribution [1]InstaDeep [2]Work done during internship at InstaDeep. Correspondence to: Ulrich A. Mbou Sob <u.mbousob@instadeep.com>.

Proceedings of the ICML 2024 Workshop on Accessible and Efficient Foundation Models for Biological Discovery, Vienna, Austria. 2024.

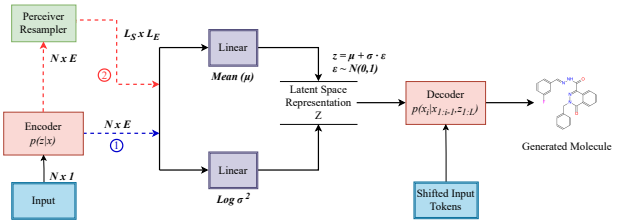


Figure 1. Schematic representation of our model's architecture. A sequence of N tokens is passed as input to our encoder which is a transformer model. The output encoded embeddings of shape $N \times E$ are either passed directly to the `mean` and `logvar` layers (path 1) or they are first passed to the perceiver resampler layer which maps the encoded embeddings to a reduced dimension of shape $L_S \times L_E$ (path 2). The `mean` and `logvar` layers are linear layers that are applied independently to each sequence dimension. The final reparametrised embeddings are then passed to the decoder transformer model to be used as encoder embeddings in the decoder's cross-attention layers.

2. Related Work

Molecular string representations. The most popular and versatile notation for string encoding of molecules is the SMILES notation (Weininger, 1988). Approaches such as Olivecrona et al. (2017), Gupta et al. (2018), and Gómez-Bombarelli et al. (2018) have trained different models such as RNNs for molecular generation using SMILES. Due to the difficulty of generating a high percentage of valid molecules with SMILES notation, various other notations have been proposed and different models have been trained using DEEPSMILES (O'Boyle & Dalke, 2018) and SELFIES (Krenn et al., 2020; Feng et al., 2024). Recently, the SAFE (Podda et al., 2020) representation was introduced, which in our opinion has the most potential to alleviate current challenges with single token string representations. A new study (Skinnider, 2024) demonstrated that models that generate invalid molecules are more powerful than generate only valid molecules. However this study only compares SMILES and SELFIES.

Score-based approaches. A major challenge in *de novo* drug design is the need to solve a multi-objective problem, where the generated molecules need to possess various and usually unrelated physicochemical and pharmacological properties. Additionally, some of these properties are difficult to measure or computationally expensive to approximate with simulation tools such as AutoDock (Trott & Olson, 2010) — especially those related to the binding energy and affinity to a given target. Furthermore, there are a limited number of available datasets to train deep learning models to accurately predict these properties. Some attempts to learn molecular docking scores were made by Corso et al. (2022) and Ganea et al. (2021). Different score-based approaches, such as using RL or diffusion models, have been proposed by Olivecrona et al. (2017), Jeon & Kim (2020), Jo et al. (2022), Lee et al. (2023), and Tarasov et al. (2023). Models such as (Lee et al., 2023) have achieved SOTA performance in molecular docking. Additionally, multi-objective Bayesian methods, as shown by (Fromer & Coley, 2023), have demonstrated promising results in small molecules design. These approaches have mainly been applied to fine-tune models for specific single protein targets. The architecture proposed in this work opens the possibility of having a single model that can generalize to multiple protein targets if the structural intimation about the protein is provided to the model.

3. Methods

Tokenization In SMILES notation, digits identify opening and closing ring atoms. This makes it difficult to train deep learning models since multiple SMILES can correspond to the same molecule. Furthermore, ordering does not matter in SMILES notation, making it more challenging to train sequence models that pay attention to the position of the tokens in the sequence.

The SAFE tokenization modifies the SMILES notation by representing the molecule as a sequence of connected fragments. The molecule is fragmented using the BRICS algorithm (Degen et al., 2008). Within the individual fragments, the SAFE tokenization preserves the ordering of the tokens but the ordering of interconnected fragments is not retained. Instead, the fragments are sorted by their molecular weights in decreasing order.

Challenges with SAFE We believe training a deep learning model using SAFE tokenization remains challenging due to the following two reasons:

1. The fact that the ordering of the fragments is not preserved produces a similar challenge as when using SMILES due to the positional ambiguity and multiple SMILES corresponding to the same molecule.

2. The model still has to learn to use digits to represent the opening and closing ring atoms. Hence the model suffers from the same challenges faced when using SMILES in its ability to generate valid molecules.

A SAFER tokenizer for small molecules To improve upon SAFE tokenization, we propose the following modifications to the SMILES to SAFE conversion algorithm:

1. We canonicalize the input SMILES to enforce that each molecule has a unique SMILES notation.

2. After breaking the molecules into different fragments, we do not sort these fragments by their molecular

Table 1. Example of SMILES, SAFE and our SAFER representation for the same molecular string.

Representation	String
SMILES	CC(C)(C)c1ccc2occ(CC(=O)Nc3ccccc3F)c2c1
SAFE	CC7(C)C.c17ccc2occ6c2c1.C6C4=O.N45.c15ccccc1F
SAFER	CC[@](C)C.c1[@@]ccc2occ[@]c2c1.C[@@]C[@]=O.N[@@][@].c1[@@]ccccc1F

weight. Instead, we preserve the ordering of the fragments as they appear in the original canonicalized sequence.

3. We introduce two new tokens [@] and [@@] to denote open and closing ring atoms. Hence, instead of using digits to represent the ring atoms for the bonds that link the various fragments we instead use these two unique tokens. The need to preserve the ordering of the fragments arises as a consequence of the order in which the various attachment points must appear. This also reduces the necessity for the language model to learn different digit tokens which intrinsically do not have unique meanings.

Algorithm 1 shows the modifications we make to the SAFE algorithm to produce our SAFER tokenizer. The lines in blue correspond to the new steps we introduce and the strike-out text are the steps from the original algorithm that we do not use. Our approach takes inspiration from (Podda et al., 2020) where each fragment is expected to have one or two bonding atoms. Fragments at either end of the sequence have one bonding atom and fragments in the middle have two. It is important to note that our algorithm can fail if we have a fragment with more than two bonding atoms. However such cases are extremely rare and will only occur if we have fragments consisting of a single atom with valence >2. Fortunately, this can be prevented by constraining the bonds-breaking algorithm (Degen et al., 2008) similarly to how it is implemented in (Podda et al., 2020) to generate avoid breaking bonds that will lead to fragments such single atom fragments with more than two bonds. Table 1 provides an example molecule represented using the SMILES, SAFE and SAFER notations.

Model Architecture Our objective is to train a powerful generative model for small molecules. Furthermore, with a latent-variable model design our generative model can perform *de novo* molecule generation with specific properties by exploring different regions of the latent space distribution. Thus our goal is to learn a multi-dimensional latent distribution over the space of small molecules. The flowchart in Figure 1 depicts our model's architecture. We leverage the success of transformers (Vaswani et al., 2017) in learning embeddings for various sequence tokens when trained in a self-supervised setting. We construct our architecture by adding a variational auto-encoder (VAE) latent space sampling layer in between the encoder and the decoder of the canonical transformer architecture. This is similar to the approach use in Fang et al. (2021) for controllable story generation.

The input molecule is tokenized and passed to the encoder. The encoder, which is an N-Layer transformer model, computes embeddings for the input molecule. These encoded embeddings are then passed to a `mean` and `logvar` layer which computes the mean and log variance of the molecule's latent space distribution. Using the mean and log variance, we apply the reparameterization trick to sample the latent vectors which then serve as our final encoder embeddings to be passed to the decoder. The decoder, which is also a transformer, is conditioned to predict the next token in the sequence given the previous tokens. In order to scale our architecture while still maintaining the same latent space dimension, we insert an optional perceiver layer (Jaegle et al., 2021) which maps the encoded embeddings to a fixed sequence length and embedding dimension before passing the output to the `mean` and `logvar` layers. The perceiver achieves this by perfoming cross-attentions with learned query vectors that have the desired latent space dimensions.

RL Fine-Tuning to Protein Targets One of the most challenging aspects of *de novo* drug design is to generate molecules with high binding affinity to specific target proteins. This binding affinity is usually predicted through an expensive computational process called molecular docking. Molecular docking typically consist of physics-based simulations used to find the most stable 'docked' conformation of the generated molecule (ligand) with a specific binding site or pocket on target protein. Hence, a crucial step in the *in silico* drug screening process is predicting the binding affinity of the candidate molecules to specific target sites. In this section, we demonstrate a proof-of-concept, for fine-tuning various parts of our architecture to generate molecules with high docking scores for a specific protein target.

We formulate our RL fine-tuning problem as follows: given an input molecule, encode and generate a new molecule from its embeddings, and provide a reward signal as the improvement in docking score over that of the original molecule. Thus the reward is defined as follows:

$$R = \mathrm{DS}_{\mathrm{new\_molecule}} - \mathrm{DS}_{\mathrm{original\_molecule}} \tag{1}$$

where DS denotes the docking score. The `mean` and `logvar` layers of our model are considered as policy parameters.

The model is fine-tuned using the REINFORCE algorithm (Williams, 1992) minimising the following loss function:

$$\mathcal{L} = -\log \pi(a|s) \times R, \tag{2}$$

where $\pi(a|s)$ denotes our policy. The state s is represented by the encoded embeddings of the original molecule whose

docking score we aim to improve, while the action a involves choosing the `mean` and `variance` of the region in our latent space from which to sample the modified molecule.

4. Experiments

In this section, we compare the performance of both our models and the SAFER representation. Recall that our main goal is to train a generative model for small drug-like molecules. To comprehensively evaluate the performance of our various models we measure the properties of the types of molecules generated. Hence, during evaluation we randomly sample a set of molecules using each model and compute the following quantitative metrics: Validity rate, Fragmentation rate, Uniqueness, Similarity, Quantitative Estimate of Drug-likeness (QED) and Synthetic Accessibility (SA) (see Appendix A.4). To compare models during training we use the following combined metric:

$$\begin{aligned}\text{Validation\_Metric} = \text{Validity\_Rate} \times \\ (1 - \text{Fragmentation\_Rate}) \times \text{Mean\_QED}\end{aligned} \tag{3}$$

For drug-like molecules, we want our models to have a high validity rate, low fragmentation rate and high QED values.

Model Scaling We perform various experiments to evaluate how our model's performance scales with model size. Due to the need to have an architecture with a fixed latent dimension, we include a perceiver-resampler layer in these experiments to map the encoder's output to a fixed embedding dimension before passing these to the `mean` and `logvar` layers. We trained two sets of models: one with an embedding dimension of 128 ($\sim$4.5M parameters) and another with an embedding dimension of 256 ($\sim$16M parameters). For each embedding dimension, we trained another model in which we included the perceiver layer to map the embeddings to a smaller latent dimension equal to half of the original embedding dimension.

Table 2 presents the average metrics from 10k samples generated, with the best parameters for each model using greedy decoding. All of our models are capable of generating molecules with a validity rate $> 90\%$ and a fragmentation rate $< 1\%$. Additionally, all our models have a mean QED > 0.75, an encouraging result. Interestingly, the model with an embedding dimension of 128 and the inclusion of the perceiver layer (Emb-128 + P-64) significantly outperforms all other models in terms of uniqueness (diversity). However, in terms of our selected validation metric, the model with an embedding dimension of 128 and without the perceiver (Emb-128) performs the best. The notable difference in uniqueness when the perceiver is applied to the model with a dimension of 64 (Emb-128 + P-64), suggests that an embedding dimension less than 128 is sufficient to represent the latent distribution of our dataset. In contrast, the results from the model with an embedding dimension of 256 are more challenging to interpret, as there are no noticeable differences between the models with and without the perceiver.

Table 2. Average scores of 10k molecules sampled using different model configurations trained with our SAFER representation and temperature parameter set to 0. $\downarrow$ and $\uparrow$ indicate that the metric must be minimized or maximized, respectively. All metrics have been scaled to be in the range [0,1]. The first number on the configuration label indicates the embedding dimension and the number after "+" indicates the perceiver's output dimension for configurations in which we include the perceiver layer.

Configuration	Validity ↑	Frag rate ↓	Uniqueness ↑	QED ↑	SA ↓	Val Metric ↑
Emb-128	**0.97**	0.002	0.59	0.86	**0.72**	**0.83**
Emb-256	0.91	0.013	0.73	**0.87**	0.76	0.782
Emb-128 + P-64	0.90	0.0498	**0.855**	0.75	0.87	0.65
Emb-256 + P-128	0.92	**0.0015**	0.71	0.86	0.76	0.788

Model Finetuining We *independently* fine-tune our model for the same five human target proteins in (Lee et al., 2023) i.e. PARP-1 (Poly [ADP-ribose] polymerase 1), F7 (Coagulation factor VII), 5-HT-1B (5-hydroxytryptamine receptor 1B), B-raf (Serine/threonine-protein kinase B-raf), and JAK2 (Tyrosine-protein kinase JAK2). Our experiment methodology follows (Lee et al., 2023). Specifically, we use the same ZINC 250k dataset (Irwin & Shoichet, 2005), which in our presented setup is used for fine-tuning with RL. For evaluation, we sample 3000 molecules and compute the same statistics, namely mean docking scores of top 5% hit molecules and percentage of hit molecules. As done by (Lee et al., 2023), we define hit molecules as molecules with QED > 0.5, SA < 0.5 and docking scores less than the hit threshold of the specific target (-9.1, -10.3, -8.5, -10.0 and -8.78 kcal/mol for JAK2, B-raf, F7, PARP-1 and 5-HT-1B respectively). Our results are presented in Tables 3 and 4 alongside the best results reported from the MOOD model in (Lee et al., 2023). We refer to our model as **MoGeL**, which stands for **Mo**lecule **G**eneration with **L**atents. We fine-tuned the models Emb-128 and Emb-128 + P-64 (see section 4). In this section, we will refer to these models as MoGeL-128 and MoGeL-64, with the numbers indicating the embedding dimension of the latent space, respectively.

Our results indicate that our pre-trained model performs on par with the MOOD w/o OOD model (i.e when MOOD is not trained to encourage out of training distribution sampling) on most of the targets except 5-HT-1B. Our RL fine-tuning improves the performance of our pre-trained model by almost doubling the percentage of hit molecules across all targets. The mean docking scores of our top 5% of molecules closely match those of the MOOD model, and our models surpass MOOD in terms of the percentage of hits for the targets JAK2, F7, and PARP1-1. This is an

Table 3. Mean Docking Scores (kcal/mol) and standard deviations (in brackets) of the top 5% hit molecules from 3000 samples using different pre-trained and RL-fine-tuned models, along with the best results from the MOOD model. Lower docking scores correspond to stronger binding energies, indicating more stable protein-ligand complexes. The asterisk is used to indicate that, based on the standard deviation there is not much statistical difference with the best performing model.

Model	JAK2	F7	PARP-1	B-raf	5-HT-1B
MOOD w/o OOD (Lee et al., 2023)	-9.575 (0.075)	-7.947 (0.034)	-10.409 (0.030)	-10.421 (0.05)	-10.487 (0.068)
MOOD (Lee et al., 2023)	**-10.147 (0.060)**	-8.160 (0.071)	**-10.865 (0.113)**	**-11.063 (0.034)**	**-11.145 (0.042)**
MoGeL-128 (ours)	-9.53 (0.34)	-8.10 (0.24)	-10.16 (0.36)	-9.94 (0.34)	-10.04 (0.32)
MoGeL-128 + RL (ours)	-9.74 (0.31)	-8.29 (0.29)	-10.62 (0.33)	-10.14 (0.49)	-10.40 (0.39)
MoGeL-64 (ours)	-9.71 (0.35)	-8.33 (0.24)	-10.49 (0.35)	-10.52 (0.39)	-10.40 (0.32)
MoGeL-64 + RL (ours)	**-9.89 (0.41)***	**-8.59 (1.21)**	**-10.72 (0.35)***	-10.57 (0.36)	-10.52 (0.24)

Table 4. Percentage of hits molecules (%) from 3000 samples using different pre-trained and RL-fine-tuned models, along with the best results from the MOOD model

Model	JAK2	F7	PARP-1	B-raf	5-HT-1B
MOOD w/o OOD (Lee et al., 2023)	3.953	0.433	3.4	2.207	11.873
MOOD (Lee et al., 2023)	9.200	0.733	7.017	**5.240**	**18.673**
MoGeL-128 (ours)	5.13	0.27	2.54	0.46	1.89
MoGeL-128 + RL (ours)	8.91	0.91	**7.11**	1.14	4.64
MoGeL-64 (ours)	7.35	0.88	5.84	2.17	5.43
MoGeL-64 + RL (ours)	**11.03**	**1.32**	6.77	3.48	7.86

important result because our architecture allows for easy conditioning of the model based on various properties and exploration of different regions in the latent distribution. Fig. 2 displays some molecules from which we can generate hit molecules using the MoGeL-64 RL fine-tuned model to the target JAK2. These images validate our model's ability to generate molecules with QED values similar to those in our training set. We observed that the docking scores of the molecules before fine-tuning are very similar to the original docking scores, while there is a significant increase in docking scores after fine-tuning. Another interesting observation is that the original molecules and the fine-tuned molecules often contain similar types of functional groups and atoms but in different arrangements and numbers. The structures of the molecules are thus altered, with moieties added or removed to improve the docking scores, sometimes with major changes to the core structure — the scaffold. This suggests that MoGeL is capable of both hit optimization and *scaffold hopping*, i.e. improving around a scaffold or finding alternative structural motifs which retain or enhance target binding affinity (Böhm et al., 2004).

5. Discussion

This work introduces a new generative model for small molecule generation, built on a transformer encoder-decoder architecture with a latent sampling space. We propose a molecular representation called SAFER, an adaptation of the SAFE representation, to train this model. Our experiments demonstrate the model's ability to generate small drug-like

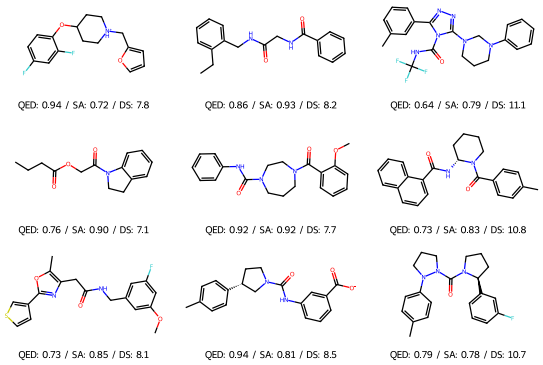


Figure 2. Visualisation of generated molecules. **Left:** Original molecules from the dataset. **Middle:** Generated molecules with the pre-trained model. **Right:** The generated molecules obtained after RL fine-tuning. These results were obtained after fine-tuning MoGeL-64 for molecular docking to the target JAK2. The labels indicate the molecules measured QED, SA and docking score.

molecules and show that fine-tuning various parts of the model significantly improves generation of new molecules with high docking scores to specific protein targets. This flexibility is a key feature of our architecture. Future work will involve scaling the model, studying the latent distribution properties, and conditioning the fine-tuned model on protein target structural information. If this conditioning is successful, it could allow the model to generalise to generating molecules with high docking scores for unseen protein targets, improving the exploration of the chemical space in the drug discovery process.

References

Alhossary, A., Handoko, S. D., Mu, Y., and Kwoh, C.-K. Fast, accurate, and reliable molecular docking with QuickVina 2. *Bioinformatics*, 31(13):2214–2216, 02 2015. ISSN 1367-4803. doi: 10.1093/bioinformatics/btv082. URL https://doi.org/10.1093/bioinformatics/btv082.

Atance, S. R., Diez, J. V., Engkvist, O., Olsson, S., and Mercado, R. De novo drug design using reinforcement learning with graph-based deep generative models. *Journal of Chemical Information and Modeling*, 62(20):4863–4872, 2022.

Bickerton, G. R., Paolini, G. V., Besnard, J., Muresan, S., and Hopkins, A. L. Quantifying the chemical beauty of drugs. *Nat Chem*, 4(2):90–98, Jan 2012.

Bohacek, R. S., McMartin, C., and Guida, W. C. The art and practice of structure-based drug design: a molecular modeling perspective. *Medicinal research reviews*, 16(1): 3–50, 1996.

Bongini, P., Bianchini, M., and Scarselli, F. Molecular generative graph neural networks for drug discovery. *Neurocomputing*, 450:242–252, 2021.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q. JAX: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

Böhm, H.-J., Flohr, A., and Stahl, M. Scaffold hopping. *Drug Discovery Today: Technologies*, 1(3):217–224, 2004. ISSN 1740-6749. doi: https://doi.org/10.1016/j.ddtec.2004.10.009. URL https://www.sciencedirect.com/science/article/pii/S1740674904000460.

Choi, S.-S., Cha, S.-H., Tappert, C. C., et al. A survey of binary similarity and distance measures. *Journal of systemics, cybernetics and informatics*, 8(1):43–48, 2010.

Corso, G., Stärk, H., Jing, B., Barzilay, R., and Jaakkola, T. Diffdock: Diffusion steps, twists, and turns for molecular docking. *arXiv preprint arXiv:2210.01776*, 2022.

Degen, J., Wegscheid-Gerlach, C., Zaliani, A., and Rarey, M. On the art of compiling and using 'drug-like' chemical fragment spaces. *ChemMedChem*, 3(10):1503–1507, 2008. doi: https://doi.org/10.1002/cmdc.200800178. URL https://chemistry-europe.onlinelibrary.wiley.com/doi/abs/10.1002/cmdc.200800178.

Du, Y., Fu, T., Sun, J., and Liu, S. Molgensurvey: A systematic survey in machine learning models for molecule design. *arXiv preprint arXiv:2203.14500*, 2022.

Fang, L., Zeng, T., Liu, C., Bo, L., Dong, W., and Chen, C. Transformer-based conditional variational autoencoder for controllable story generation. *arXiv preprint arXiv:2101.00828*, 2021.

Feng, Y., Zhang, Y., Deng, Z., and Xiong, M. Gcardti: Drug–target interaction prediction based on a hybrid mechanism in drug selfies. *Quantitative Biology*, 2024.

Fromer, J. C. and Coley, C. W. Computer-aided multi-objective optimization in small molecule discovery. *Patterns*, 4(2):100678, 2023. ISSN 2666-3899. doi: https://doi.org/10.1016/j.patter.2023.100678. URL https://www.sciencedirect.com/science/article/pii/S2666389923000016.

Ganea, O.-E., Huang, X., Bunne, C., Bian, Y., Barzilay, R., Jaakkola, T., and Krause, A. Independent SE(3)-equivariant models for end-to-end rigid protein docking. *arXiv preprint arXiv:2111.07786*, 2021.

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., and Aspuru-Guzik, A. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

Gupta, A., Müller, A. T., Huisman, B. J., Fuchs, J. A., Schneider, P., and Schneider, G. Generative recurrent networks for de novo drug design. *Molecular informatics*, 37(1-2):1700111, 2018.

Irwin, J. J. and Shoichet, B. K. ZINC- a free database of commercially available compounds for virtual screening. *Journal of chemical information and modeling*, 45(1): 177–182, 2005.

Irwin, R., Dimitriadis, S., He, J., and Bjerrum, E. J. Chemformer: a pre-trained transformer for computational chemistry. *Machine Learning: Science and Technology*, 3(1):015022, 2022.

Jaegle, A., Borgeaud, S., Alayrac, J.-B., Doersch, C., Ionescu, C., Ding, D., Koppula, S., Zoran, D., Brock, A., Shelhamer, E., et al. Perceiver io: A general architecture for structured inputs & outputs. *arXiv preprint arXiv:2107.14795*, 2021.

Jeon, W. and Kim, D. Autonomous molecule generation using reinforcement learning and docking to develop potential novel inhibitors. *Scientific reports*, 10(1):22104, 2020.

Jiang, D., Wu, Z., Hsieh, C.-Y., Chen, G., Liao, B., Wang, Z., Shen, C., Cao, D., Wu, J., and Hou, T. Could graph neural networks learn better molecular representation for drug discovery? a comparison study of descriptor-based and graph-based models. *Journal of cheminformatics*, 13: 1–23, 2021.

Jo, J., Lee, S., and Hwang, S. J. Score-based generative modeling of graphs via the system of stochastic differential equations. In *International Conference on Machine Learning*, pp. 10362–10383. PMLR, 2022.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Krenn, M., Häse, F., Nigam, A., Friederich, P., and Aspuru-Guzik, A. Self-referencing embedded strings (SELFIES): A 100% robust molecular string representation. *Machine Learning: Science and Technology*, 1(4):045024, 2020.

Landrum, G. et al. Rdkit: A software suite for cheminformatics, computational chemistry, and predictive modeling. *Greg Landrum*, 8(31.10):5281, 2013.

Langley, P. Crafting papers on machine learning. In Langley, P. (ed.), *Proceedings of the 17th International Conference on Machine Learning (ICML 2000)*, pp. 1207–1216, Stanford, CA, 2000. Morgan Kaufmann.

Lee, S., Jo, J., and Hwang, S. J. Exploring chemical space with score-based out-of-distribution generation. In *International Conference on Machine Learning*, pp. 18872–18892. PMLR, 2023.

Luo, Y., Yan, K., and Ji, S. Graphdf: A discrete flow model for molecular graph generation. In *International conference on machine learning*, pp. 7192–7203. PMLR, 2021.

Madhawa, K., Ishiguro, K., Nakago, K., and Abe, M. Graphnvp: An invertible flow model for generating molecular graphs. *arXiv preprint arXiv:1905.11600*, 2019.

Noutahi, E., Gabellini, C., Craig, M., Lim, J. S., and Tossou, P. Gotta be safe: A new framework for molecular design. *Digital Discovery*, 2024.

O'Boyle, N. and Dalke, A. Deepsmiles: An adaptation of SMILES for use in machine-learning of chemical structures. *ChemRxiv*, 2018. doi: 10.26434/chemrxiv.7097960.v1.

Olivecrona, M., Blaschke, T., Engkvist, O., and Chen, H. Molecular de-novo design through deep reinforcement learning. *Journal of cheminformatics*, 9:1–14, 2017.

Podda, M., Bacciu, D., and Micheli, A. A deep generative model for fragment-based molecule generation. In *International conference on artificial intelligence and statistics*, pp. 2240–2250. PMLR, 2020.

Polishchuk, P. G., Madzhidov, T. I., and Varnek, A. Estimation of the size of drug-like chemical space based on gdb-17 data. *Journal of computer-aided molecular design*, 27:675–679, 2013.

Powers, A. S., Yu, H. H., Suriana, P., and Dror, R. O. Fragment-based ligand generation guided by geometric deep learning on protein-ligand structure. *bioRxiv*, pp. 2022–03, 2022.

Satorras, V. G., Hoogeboom, E., and Welling, M. E(n) equivariant graph neural networks. In *International conference on machine learning*, pp. 9323–9332. PMLR, 2021.

Skinnider, M. A. Invalid smiles are beneficial rather than detrimental to chemical language models. *Nature Machine Intelligence*, 6(4):437–448, 2024.

Tarasov, D., Mbou Sob, U. A., Arbesu, M., Siboni, N., Boyer, S., Skwark, M., Smit, A., Bent, O., and Pretorius, A. Offline RL for generative design of protein binders. *bioRxiv*, pp. 2023–11, 2023.

Trott, O. and Olson, A. J. Autodock Vina: improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *Journal of computational chemistry*, 31(2):455–461, 2010.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., and Polosukhin, I. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

Weininger, D. SMILES, a chemical language and information system. 1. introduction to methodology and encoding rules. *Journal of Chemical Information and Computer Sciences*, 28(1):31–36, 1988. doi: 10.1021/ci00057a005. URL https://doi.org/10.1021/ci00057a005.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256, 1992.

Zang, C. and Wang, F. Moflow: an invertible flow model for generating molecular graphs. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pp. 617–626, 2020.

A. Appendix

A.1. Conversion Algorithm

Algorithm 1 Conversion from SMILES to SAFER representation

procedure CONVERTSAFE(*molecule*)
  molecule ← Standardized *molecule*
  ring_digits ← extract all unique ring digits from *molecule*
  fragments ← fragment *molecule* on specified bonds ▷ We use BRICS bonds here
  fragments_str ← {}
  ~~Sort *fragments* by size in descending order~~
  for *each frag* in *fragments* **do**
    Add SMILES of *frag* to *fragments_str*
  end for
  safe_str ← join all elements in *fragments_str* with "."
  attach_pos ← extract all attachment points from *safe_str*
  ~~*i* ← max(*ring_digits*) + 1 ▷ Find the next possible ring digits~~
  for *each attach* in *attach_pos* **do**
    ~~Replace *attach* in *safe_str* with *i*~~
    ~~*i* ← *i* + 1~~
    Replace *opening attachment* in *safe_str* with "[@]"
    Replace *closing attachment* in *safe_str* with "[@@]"
  end for
  return *safe_str*
end procedure

A.2. Datasets

For model pre-training, we use a large publicly available SMILES dataset. Specifically, we make use of the SAFE dataset on HuggingFace ((Noutahi et al., 2024)). This dataset is one of the largest publicly available SMILES datasets, with over 1 billion molecules, curated from various datasets such as the ZINC and UniCHEM libraries (Noutahi et al., 2024). It consists of various molecular types ranging from drug-like molecules, peptides, multi-fragment molecules, polymers, reagents and non-small molecules.

In our pre-training pipeline, we select a subset of the data consisting of the first 21 parquet files. Since the data is provided in the SAFE representation, we use the SAFE package[1] to convert the molecules to their corresponding SMILES representation, and then convert to our SAFER representation. We discard any molecule that fails the SAFE to SMILES conversion and we filter all molecules to have a maximum of 140 tokens. This results in the final dataset consisting of roughly 150 million small molecules.

In order to fine-tune the models, we used the zinc250k dataset (Irwin & Shoichet, 2005). This dataset comprises 250k small drug-like molecules from the ZINC database. While the SAFE dataset we used for pre-training already includes molecules from the ZINC database, we specifically utilized the zinc250k dataset for fine-tuning to maintain consistency with the MOOD model (Lee et al., 2023), which we are using for comparison. We created a SAFER version of this dataset for this purpose.

A.3. Model's implementation and pre-training details

To train the entire network we use the following loss function:

$$\mathcal{L} = \sum_{x_{1:L} \in \mathcal{D}}^{|\mathcal{D}|} \sum_{i=1}^{L} \log p(x_i | x_{1:i-1}, z_{1:L}) \; - \; \mathrm{KL}\left(\mathcal{N}(\mu_{1:L}, \mathrm{diag}(\sigma^2_{1:L})) \,\middle\|\, \mathcal{N}(0, I)\right), \tag{4}$$

[1] SAFE-github: https://github.com/datamol-io/safe/

where $x_{1:L}$ is a molecule sequence of length L sampled from the dataset $\mathcal{D}$ and $z_{1:L}$ is the corresponding latent vector sampled from a multivariate Gaussian distribution $\mathcal{N}(\mu_{1:L}, \text{diag}(\sigma^2_{1:L}))$ and used during cross attention. The loss function consists of two main terms. The first term is the canonical self-supervised language model loss, which in our setting can also be interpreted as a type of reconstruction loss which measures the model's ability to reconstruct the input sequence. The second term is a regularisation term that aims to maintain the learned latent space distribution as close as possible to a prior distribution which in this case is a standard Gaussian distribution. Hence to generate molecules we sample random embeddings from a Gaussian distribution and pass that to the decoder network to autoregressively generate a new molecule.

We implement and train our model using JAX (Bradbury et al., 2018) taking advantage of JAX accelerator functions. We use the pre-trained SAFE BPE tokenizer[2]. This tokenizer is adapted to include the two additional tokens required for the SAFER representation. We used 95 % of the data from training and 5 % for validation. The models were trained on a v3-8 Google TPU. All model training was performed in parallel across each of the 8 TPU devices using a batch size of 128 per device with a mini-batch size of 16 (resulting in gradient accumulation of 8). Both the encoder and decoder are N-layer transformer models with learned positional embeddings. The decoder output layer is supplemented with a linear head. The encoder's output layer is linked to either the perceiver layer or the `mean` and `logvar` layers. The `mean` and `logvar` layers are implemented as single linear layers. Table 5 presents the configuration details and hyperparameters for each of our pre-trained models.

Table 5. Model parameters

Shared parameters	Encoder layers	8
	Decoder layers	8
	Attention heads	8
Emb-128 / Emb-128 + P-64	Embedding dim	128
	Forward feedward dim	512
	Final dim	128 / 64
	Perceiver layer	False / True
Emb-256 / Emb-256 + P-128	Embedding dim	256
	Forward feedward dim	1024
	Final dim	256 / 128
	Perceiver layer	False / True
Hyperparameters	Batch size	128
	Gradient accumulations	8
	Optimizer	Adam (Kingma & Ba, 2014)
	Learning rate	5e-5
	Training epochs	1

A.4. Evalution metrics

1. **Validity rate**: The percentage of syntactically valid SMILES i.e. the number of molecule sequences we are able to successfully construct from their SMILES representation using software such as RDKit (Landrum et al., 2013).

2. **Fragmentation rate**: The number of molecules with SMILES representations that lead to fragmented (disconnected) molecules.

3. **Uniqueness**: The percentage of unique SMILES representations in a set of molecules.

4. **Similarity**: To measure the novelty of the generated molecule, we use RDKit to compute the Tanimoto similarity distance between the Morgan fingerprints of generated samples and the molecules in the training set (see Choi et al. (2010)). Due to the large size of the training set, we only compare against a subset of the training set molecules, hence we do not put too much emphasis on this metric in our analysis.

[2]Tokenizer: `https://huggingface.co/datamol-io/safe-gpt`

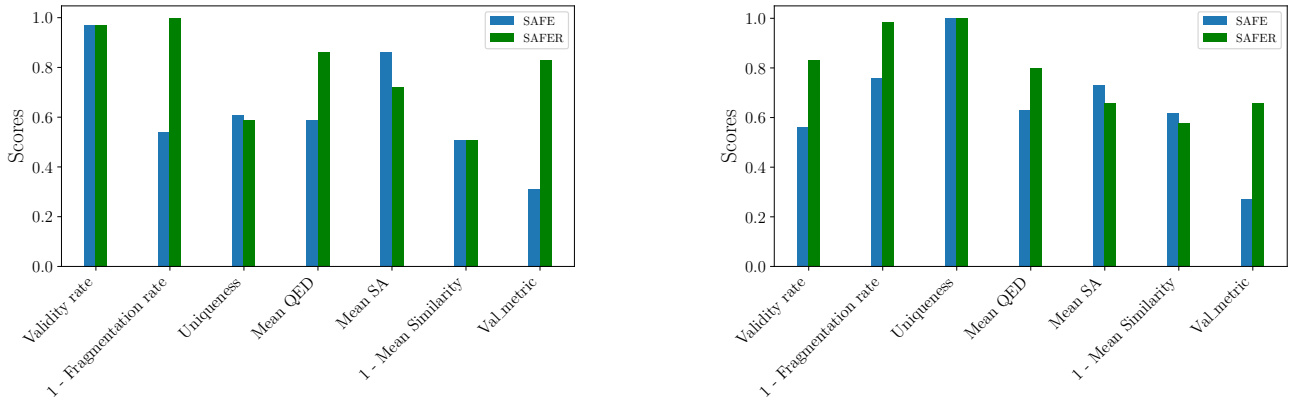
5. **Quantitative Estimate of Drug-likeness (QED)**: The likelihood of a generated molecule to have molecular properties similar to known drugs (Bickerton et al., 2012).

6. **Synthetic Accessibility (SA)**: A metric that scores the difficulty of experimentally synthesising the generated molecule from its fragments.

A.5. SAFE vs SAFER

This experiment compares the SAFE tokenizer with our SAFER tokenizer when used to train our model. For this experiment, we choose an embedding dimension of 128 and we do not include the perceiver layer. Each model is trained for one epoch using the reduced SAFE dataset.

Our results are shown in Figure 3 comparing SAFE and SAFER representations across various metrics. Both representations are capable of generating high-quality molecules. However, our SAFER representation tends to produce more chemically plausible molecules when trained with our architecture. In contrast, the SAFE representation exhibits a fragmentation rate of almost 40% with greedy decoding and a validity rate of less than 60% with stochastic decoding (temperature = 1) compared to a fragmentation rate of less than 1% with greedy decoding and a validity rate greater than 80% with stochastic decoding for our SAFER representation.



(a) Greedy decoding

(b) Stochastic decoding

Figure 3. Comparing SAFE vs SAFER. We compute the average scores of 10k molecules sampled using two models with embedding dimension 128 trained using the SAFE and SAFER representations. **Left:** Greedy decoding (temperature = 0). **Right:** Stochastic decoding (temperature = 1). Both representations are capable of generating molecules with high QED > 0.5 but the SAFER representation significantly outperforms the SAFE representation on our combined validation metric (see Eq. 3) due to the lower fraction of fragmented molecules that are generated using the SAFER representation.

A.6. Reinforcement Learning (RL) fine-tuning

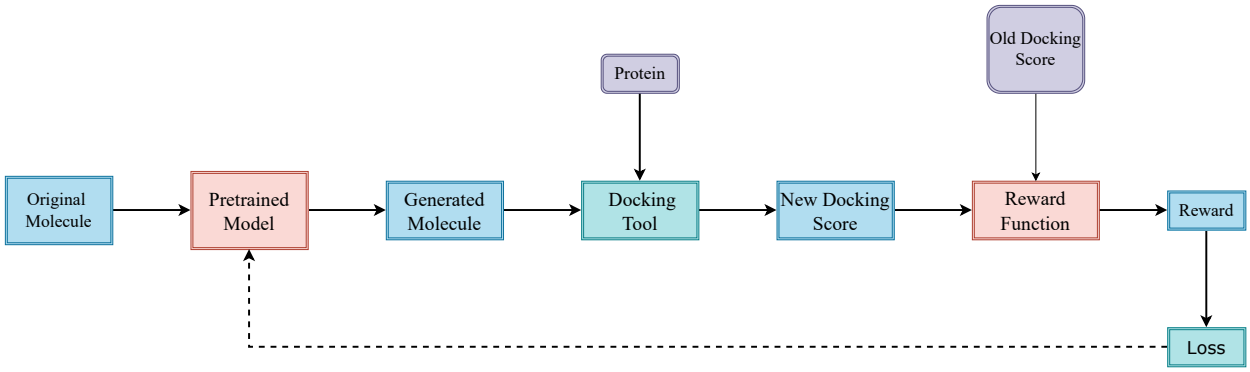
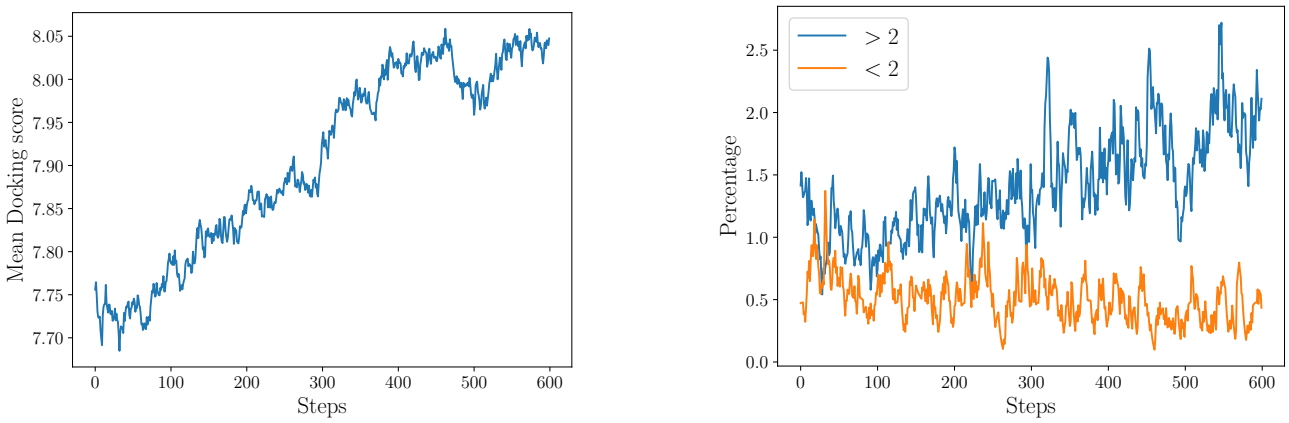


Figure 4. Schematic representation of the RL fine-tuning pipeline. The original molecule is passed to the pre-trained model that maps it to a region in our latent distribution using the `mean` and `logvar` layers. A latent vector is sampled for this region and passed to the decoder to generate a new molecule. The new molecule and the protein target are passed to the docking tool to perform molecular docking and produce a docking score for the new molecule. Following this, a reward is assigned to the molecule based on the comparison between the new docking score and the original docking score. We then use the reward to compute the loss and update the model's parameters.

For RL fine-tuning, we start by initializing both the sampling policy and the training policy with the best pre-trained checkpoint that we want to fine-tune. The policy model is used to generate samples from the input molecules in our dataset. The data loader is designed such that each input molecule is utilized multiple times to generate multiple experiences for each molecule. We use the Python wrapper tool Qvina2 (Alhossary et al., 2015) to dock the generated molecules to the target proteins and compute the docking scores. Qvina2's multi-processing capability allows us to dock multiple molecules at once. All molecules with failed docking attempts are assigned a dummy docking score of -99, which is used for masking during the loss computation. We fine-tuned the model with various learning rates and found that either 5e-4 or 5e-5 performed the best for each of the targets. Every 10 training iterations, the sampling policy is updated to the training policy. We run different experiments and fine-tuned each model with multiple seeds to vary that model doesn't perform well only for specific seeds.

Fig. 5 show some training metrics for the fine-tuning of the model MoGel-128 to the target JAK2. These metrics were obtained using samples from our validation dataset and show a steady increase in the mean docking score of the samples. Additionally, from Fig. 5(b), we can observe that the fraction of molecules whose docking scores decrease by 2 does not increase during training, unlike the fraction of molecules whose docking scores increase by 2. This suggests that our reward model is capable of guiding our model's latent space to sample molecules with higher docking scores.

Fig. 6 displays example hit molecules obtained after MoGel-128 is fine-tuned to the JAK2 target alongside the original molecules, the pre-trained generated molecule, their QED, SA, and docking scores.



(a) Mean docking scores

(b) Molecules with docking scores change > 2

Figure 5. (a) Mean docking scores on the validation set during the fine-tuning of the model MoGel-128 to the protein target JAK2. The blue and orange curves in (b) are plots of the percentage of molecules whose docking scores increase or reduce by values greater than 2, respectively.

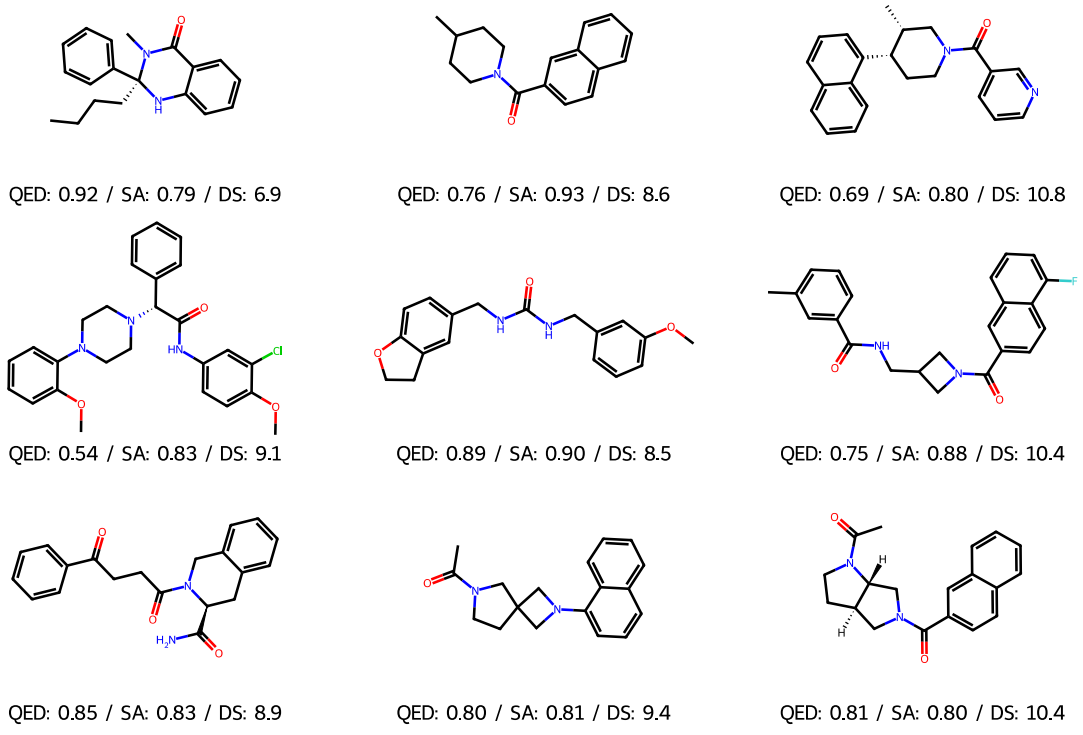


Figure 6. Visualisation of generated molecules. **Left:** Original molecules from the dataset. **Middle:** generated molecules with the pre-trained model. **Right:** The generated molecules obtained after RL fine-tuning. These results were obtained after fine-tuning MoGel-128 for molecular docking to the target JAK2. The labels indicate the molecules measured QED, SA and docking score.