RESEARCH

# Molecular Optimization by Capturing Chemist's Intuition Using Deep Neural Networks

Jiazhen He[1*], Huifang You[1,3], Emil Sandström[1,4], Eva Nittinger[2], Esben Jannik Bjerrum[1], Christian Tyrchan[2], Werngard Czechtizky[2] and Ola Engkvist[1]

**Abstract**

A main challenge in drug discovery is finding molecules with a desirable balance of multiple properties. Here, we focus on the task of molecular optimization, where the goal is to optimize a given starting molecule towards desirable properties. This task can be framed as a machine translation problem in natural language processing, where in our case, a molecule is translated into a molecule with optimized properties based on the SMILES representation. Typically, chemists would use their intuition to suggest chemical transformations for the starting molecule being optimized. A widely used strategy is the concept of matched molecular pairs where two molecules differ by a single transformation. We seek to capture the chemist's intuition from matched molecular pairs using machine translation models. Specifically, the sequence-to-sequence model with attention mechanism, and the Transformer model are employed to generate molecules with desirable properties. As a proof of concept, three ADMET properties are optimized simultaneously: *logD*, *solubility*, and *clearance*, which are important properties of a drug. Since desirable properties often vary from project to project, the user-specified desirable property changes are incorporated into the input as an additional condition together with the starting molecules being optimized. Thus, the models can be guided to generate molecules satisfying the desirable properties. Additionally, we compare the two machine translation models based on the SMILES representation, with a graph-to-graph translation model HierG2G, which has shown the state-of-the-art performance in molecular optimization. Our results show that the Transformer can generate more molecules with desirable properties by making small modifications to the given starting molecules, which can be intuitive to chemists. A further enrichment of diverse molecules can be achieved by using an ensemble of models.

**Keywords:** molecular optimization; matched molecular pairs; seq2seq; transformer; recurrent neural networks; ADMET

## Introduction

A main challenge in drug discovery is finding molecules with desirable properties. A drug requires a balance of multiple properties, *e.g.* physicochemical properties, ADMET (absorption, distribution, metabolism, elimination and toxicity) properties, safety and potency against its target. To find such a drug in the extremely large chemical space (*i.e.* $10^{23}$-$10^{60}$) [1] is challenging. It is often that a promising molecule needs to be improved to achieve a balance of multiple properties. This problem is known as molecular optimization. Traditionally, chemists would use their knowledge, experience and intuition [2] to apply some chemical transformations to the promising molecule. In particular, the matched molecular pair (MMP) analysis [3, 4]—which compares the properties of two molecules that differ only by a single chemical transformation—has been widely used as a strategy by medicinal chemists to support molecular optimization [5, 6, 7]. Typically, similarity, transferability, and linear analoguing [8, 9, 10] are assumed and applied by the chemists to suggest transformations to improve the promising molecule. However, they are not generally true, and become more problematic and difficult to apply when optimizing multiple properties simultaneously.

To address these shortcomings, this work uses deep learning models to learn the transformations involved in molecular optimization directly from MMPs. Recently, deep generative models, *e.g.* recurrent neural networks (RNNs) [11, 12], variational autoencoders (VAEs) [13, 14, 15, 16, 17, 18], and generative adversarial networks (GANs) [19, 20, 21, 22], coupled with reinforcement learning [23, 19, 22, 20], adver-

*Correspondence: jiazhen.he@astrazeneca.com
[1]Hit Discovery, Discovery Sciences, R&D, AstraZeneca, Gothenburg, Sweden
Full list of author information is available at the end of the article

sarial training [24, 25, 26], transfer learning [11], and different optimization techniques [13, 27], have been investigated to generate molecules towards desirable properties. Additionally, conditional generative models [15, 18, 28, 29] have been developed where the desirable properties are incorporated as condition to directly control the generating process. Another approach is to use reinforcement learning to modify a molecule directly based on molecular graph representation [30, 31]. However, all the above methods are not direct and intuitive methods for molecular optimization. When given a promising molecule and the desirable properties, the direct way would be applying intuitive chemical transformations to achieve the desirable properties, while the above methods ignore the domain knowledge of chemical transformations.

In this paper, we focus on utilizing chemical transformations (*i.e.* MMPs), which reflect the chemist's intuition to optimize a promising molecule. In particular, given a starting molecule and the desirable property changes, the goal is to generate molecules, which (i) have the desirable properties and (ii) are structurally similar to the starting molecule. As a proof of concept, we focus on optimizing three ADMET properties simultaneously: *logD*, *solubility* and *clearance*, which are important properties of a drug. *LogD* measures the hydrophobicity of a molecule, which influences the molecule's potency, metabolism and pharmacokinetic properties. *Solubility* influences absorption and bioavailability. *Clearance* is a measure of the capacity of drug removal by various organs, which is a key parameter to understand metabolic stability and dosing.

The problem of molecular optimization can be framed as a machine translation problem [32] in natural language processing (NLP), where a text is translated from one language to another. For molecular optimization, an input starting molecule is translated into a target molecule with optimized properties based on the simplified molecular-input line-entry system (SMILES) representation. The sequence-to-sequence (Seq2Seq) model [33] with attention mechanism [34] has been developed and applied in machine translation successfully. Recently, the Transformer, which only uses attention, has been shown to achieve the state-of-the-art (SOTA) performance in machine translation [35], and has become the basic building block of most SOTA architectures in NLP. Lately, it has been applied to predict chemical reactions and achieved SOTA performance (above 90% accuracy) on a common benchmark data set [36]. This motivated us to investigate the Seq2Seq with attention and the Transformer for molecular optimization tasks in this work.

The models are trained on MMPs extracted from ChEMBL. Since it is difficult to obtain the experimental property values for molecules in ChEMBL, we built a property prediction model for each ADMET property based on in-house experimental data. Then the models are applied on the extracted ChEMBL MMPs. In order to generate molecules towards customized desirable properties, the desirable property changes are concatenated with the source molecules' SMILES, as input to the models.

The most relevant work to this paper are Jin et al. [24, 37, 38], who view molecular optimization as a graph-to-graph translation problem. A variational junction tree encoder-decoder (VJTNN) [24] was trained on a set of MMPs, and achieved the SOTA performance in molecular optimization. Based on VJTNN, Jin et al. [37] proposed a multi-resolution, hierarchically coupled encoder-decoder for graph-to-graph translation, and extended it to be conditioned on desirable property criteria, to allow for different user-specified property criteria and multi-property optimization. Recently, Jin et al. [38] proposed a new hierarchical graph encoder-decoder (HierG2G) by utilizing graph motifs as building blocks, which are frequently occurring substructures, to facilitate generating large molecules. It was also extended to graph-to-graph translation for molecular optimization, and outperformed VJTNN.

All the above models are based on molecular graph representations, while our models are based on SMILES representations and utilize the SOTA machine translation models, the Seq2Seq with attention and the Transformer. Although Jin et al. [24, 37, 38] compared their models with Seq2Seq and showed that they performed better, the Seq2Seq used only one one-layer long short-term memory (LSTM) in the encoder and decoder, while we use multiple layers. Additionally, the Transformer, has not been explored in molecular optimization. Therefore, we conduct a comparison over these three models, Seq2Seq with attention, Transformer and HierG2G. For HierG2G [38], the conditional extension in [37] is applied to support customized property optimization and multi-property optimization.

## Methods

### Molecule Representation and Property Representation

The models are trained on a set of MMPs together with the property changes between source and target molecules. Figure 1 shows an example of a MMP, and the properties of source and target molecules. The SMILES representation of molecules [39], as a string-based representation, is used in our study to facilitate the use of machine translation models from NLP.

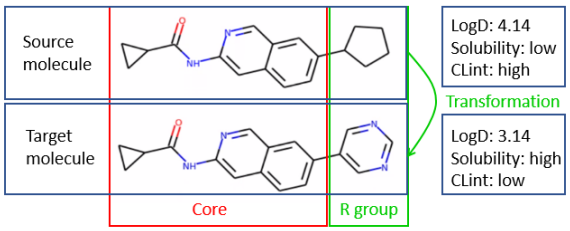



**Figure 1** An example of a matched molecular pair and the properties.

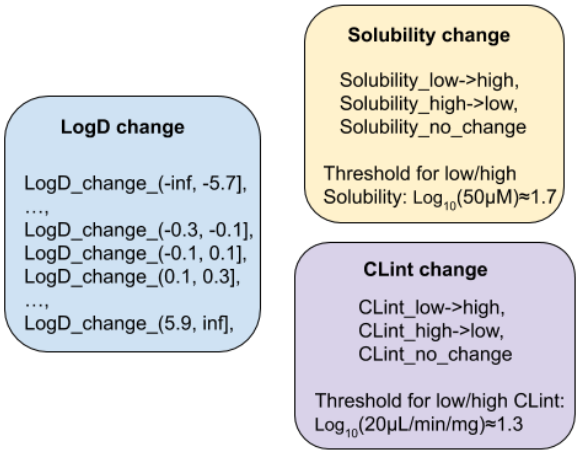



**Figure 2** Property change encoding

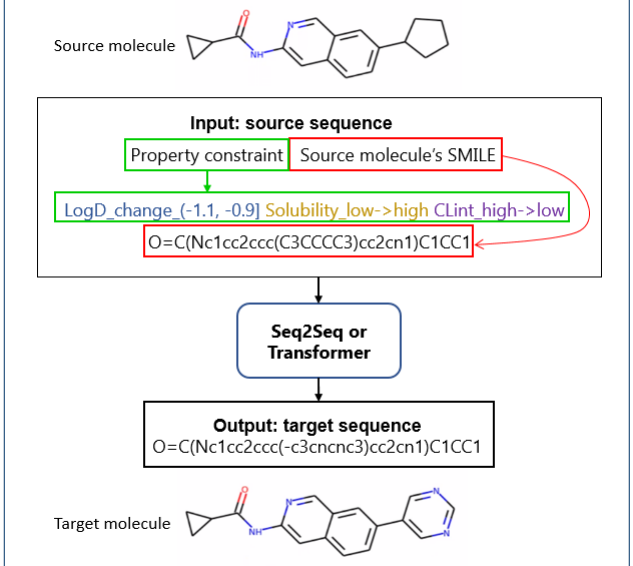



**Figure 3** An example of source and target sequences fed into Seq2Seq or Transformer during training

Considering practical desirable criteria and experimental errors, *solubility* and *clearance* changes are encoded using three categories, while the change in *logD* is encoded into range intervals, with each interval length=0.2 except for the two open intervals on the sides (Figure 2). For *clearance*, human liver microsome intrinsic clearance (HLM CLint) is used in this work, and the thresholds for low/high *solubility* and low/high *CLint* are 50 $\mu$M and 20 $\mu$L/min/mg respectively (1.7 and 1.3 respectively in $log_{10}$ scale).

In order to translate source molecules into target molecules with customized properties, the encoded property changes are concatenated with the SMILES representation of starting molecules as input sequences for machine translation models, while the target sequences are the SMILES representation of target molecules. Figure 3 shows an example of source and target sequences which are fed into machine translation models during training.

Given a set of MMPs $\{(X, Y, Z)\}$ where $X$ represents source molecule, $Y$ represents target molecule, and $Z$ represents the property change between source molecule $X$ and target molecule $Y$, the Seq2Seq with attention and the Transformer will learn a mapping $(X, Z) \in \mathcal{X} \times \mathcal{Z} \rightarrow Y \in \mathcal{Y}$ during training where $\mathcal{X} \times \mathcal{Z}$ represents the input space and $\mathcal{Y}$ represents the target space. During testing, given a new $(X, Z) \in \mathcal{X} \times \mathcal{Z}$, the models will be expected to generate a diverse set of target molecules with desirable properties.

## Seq2Seq with Attention

The Seq2Seq [33] is a framework that maps an input sequence to an output sequence, which has wide applications, such as machine translation, text summarization, chatbot, question answering system, and image captioning. In particular, it has brought a major breakthrough in neural machine translation. The Seq2Seq is based on an encoder-decoder architecture using RNN. The encoder takes an input sequence, and compresses it into a context vector, defined by the hidden state in the last time step of encoder, which captures the information of the whole input sequence. Specifically, at each time step in the encoder, the RNN takes a word from the input sequence and a hidden state from the previous time step, and output a hidden state. The hidden state memorizes the words seen earlier and is updated at each time step, and the one from the last time step is called context vector, which captures the information of the whole input sequence. The context vector is then passed to the decoder to predict an output sequence. The drawback of the Seq2Seq is that it becomes difficult to deal with long sequence because the encoder has to compress the whole sequence into a single context vector in the last time step. To over-

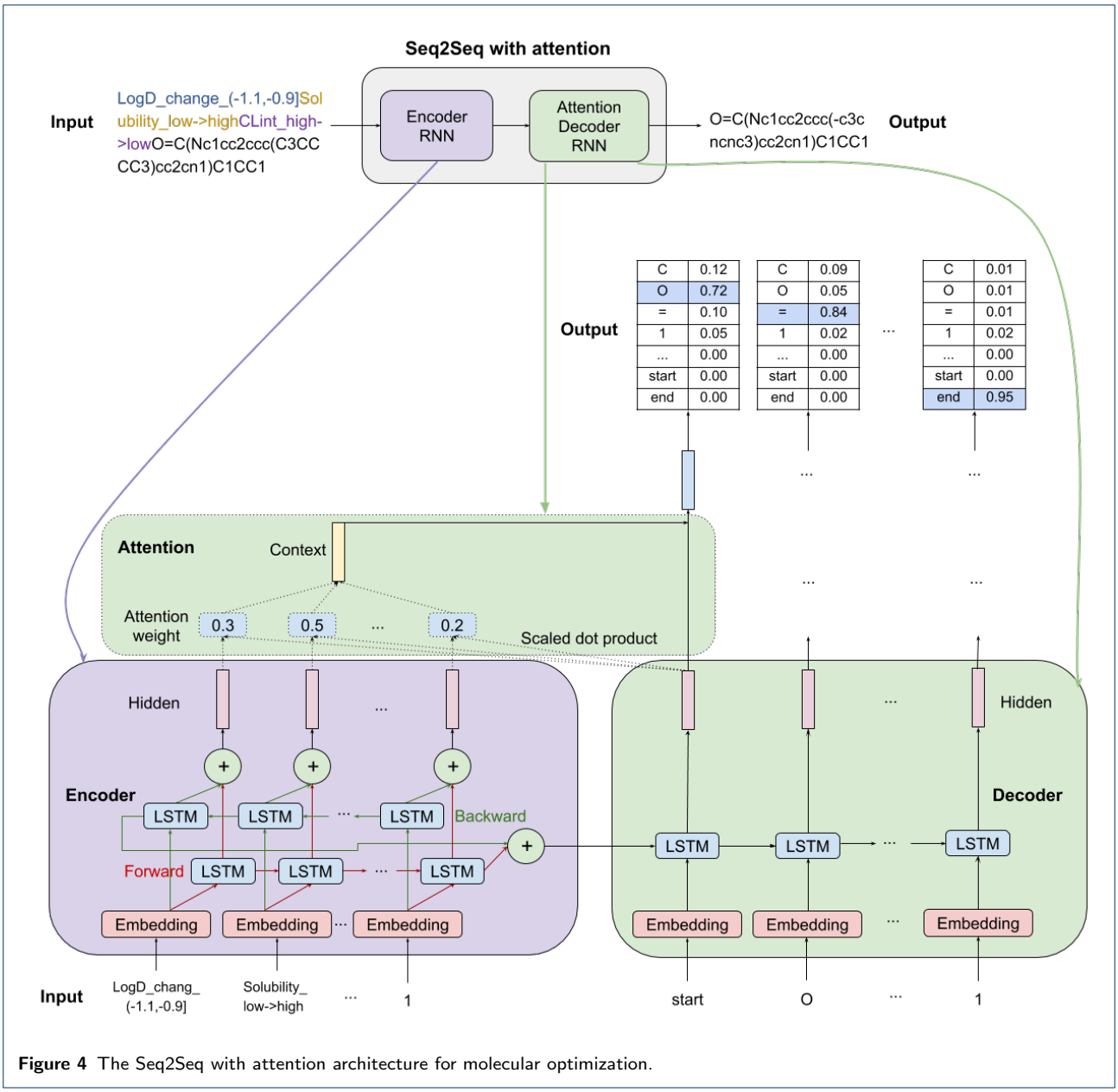



**Figure 4** The Seq2Seq with attention architecture for molecular optimization.

come this problem, attention mechanism [34] was introduced, which utilizes the hidden states at each time step from the encoder. It enables the decoder to focus on specific tokens in the input sequence when predicting each token in the output sequence.

In this paper, the Seq2Seq with attention is explored for molecular optimization (Seq2Seq refers to Seq2Seq with attention in the rest of this paper). First, all the source and target SMILES in our dataset were tokenized to construct a vocabulary, which contains all the possible tokens. Two special symbols were added, *start* and *end*, representing the start and end of a sequence respectively. In order to guide the model to generate molecules with different specified property constraints, the property changes between source and target molecule were concatenated with the source SMILES as the input sequence to Seq2Seq as illustrated in Figure 3. Therefore, each possible single property change was treated as a token (*e.g.* LogD_change_(-1.1,-0.9]) and added to the vocabulary.

The model architecture is shown in Figure 4. It consists of an encoder RNN and an attention decoder RNN, with LSTM cells. The encoder consists of an embedding layer of 256 dimensions and 5 stacked bidirectional LSTM layers with hidden size of 512 and

dropout of 0.3. The embedding layer converts the input token at each time step into a continuous representation, which is then passed through the stacked bidirectional LSTM. At the last time step, the LSTM ouputs for both directions are summed and passed to the decoder. Similarly, the LSTM hidden states at each time step for both directions are summed and used to compute the attention with the hidden state from the current time step of decoder.

The decoder consists of an embedding layer of 256 dimensions and 5 stacked unidirectional LSTM layers with hidden size of 512 and dropout of 0.3. Additionally, it includes an attention layer. The initial input token is the *start* token. At each time step in decoding, the attention layer computes the attention weights which captures the importance of each source token for predicting the next target token. The attention weights are computed by the scaled dot product between the the hidden states at all time steps in the encoder and the hidden state at the current time step in the decoder, followed by a softmax function. Then a context vector is obtained by a weighted sum of the encoder hidden states at all time step. The context vector captures relevant information from every source token to help predict the next target token. It is concatenated with the hidden state at the current time step in decoder and then passed through a linear layer with a hyperbolic tangent activation function. The output is lastly passed through a linear layer to reshape to the vocabulary size, and a softmax activation function is applied to obtain the probabilities of each token in the vocabulary.

The model is trained to predict the next token of the target sequence, given previous tokens of the target sequence conditioned on the input sequence. In particular, teacher forcing, as a commonly used training technique for aiding efficient training of RNN, is used in the decoder, where the ground-truth target token in the training set at the current time step rather than the output generated by the model, is used as input for the next time step. Specifically, given a training set, $D = \{(\mathbf{x}_i, \mathbf{y}_i)\}$ where $\mathbf{x}_i$ and $\mathbf{y}_i$ represents the $i$th source sequence and target sequence respectively in the dataset $D$, we find $\theta$ to minimize negative log likelihood (NLL):

$$\mathrm{NLL}(\theta) = -\sum_{i \in D} \sum_{t=1}^{|\mathbf{y}_i|} \log \mathrm{P}\left(y_{i,t} \mid \mathbf{y}_{i,1:t-1}, \mathbf{x}_i; \theta\right) \quad (1)$$

where $\theta$ represents all the parameters in the model, $y_{i,t}$ represents the $t$th token of $\mathbf{y}_i$.

Model training and inferencing was performed on a NVIDIA GeForce RTX 2080 Ti. The Adam optimizer with learning rate 0.0001 and a batch size of 128 were used. The hyperparameters were tuned based on previous experience [35, 33]. After training, when using the model for generation, the ground-truth target sequence is not available and the output sequence is generated by sampling one token at a time from the distribution over the vocabulary until the *end* token is sampled. Specifically, multinomial sampling is used to generate multiple molecules for a given input sequence.

### Transformer Architecture

Although the Seq2Seq with attention has achieved great success in machine translation, it is still challenging to deal with long-range dependencies, and the sequential nature of the RNN prevents parallelization. The Transformer [35] was proposed to discard the RNN and rely only on attention mechanism instead. In this paper, the Transformer architecture illustrated graphically in [35] is explored for molecular optimization, as shown in Figure 5. The Transformer consists of an encoder and a decoder. First, a vocabulary is constructed the same way as the Seq2Seq with attention. Before feeding the input sequence to the encoder, each token in the input sequence is converted into an embedding vector, followed by a positional encoding, which gives the embeddings order information.

**Encoder** The encoder consists of a stack of N identical encoder layers. Each encoder layer takes a list of input encodings from the previous encoder layer (list size is determined by the input sequence length) as input, and generates a list of output encodings that pass through the next layer. The input for the first encoder layer is the embedding of each token in the input sequence. Each encoder layer has two sub-layers: a multi-head self-attention sub-layer and a position-wise fully connected feed-forward network sub-layer. A residual connection is used around each of the two sub-layers, followed by layer normalization.

In the first encoder layer, for each positional token embedding at position $t$ in the input sequence, it first flows through the multi-head self-attention sub-layer, which helps the encoder focus on relevant tokens in the input sequence to better encode it. Specifically, three vectors, Q (query), K (key), V (value) are first created from the input token embedding by multiplying with three weight matrices that are learned during training. These three vectors are used to compute the self-attention score for the input token at position $t$, which determines the importance of all the tokens in the input sentence for encoding the input token being processed. The score is computed by the scaled dot-product between Q of the input token at $t$ being processed and K of each token in the input sequence, followed by a softmax function. Then a weighted sum

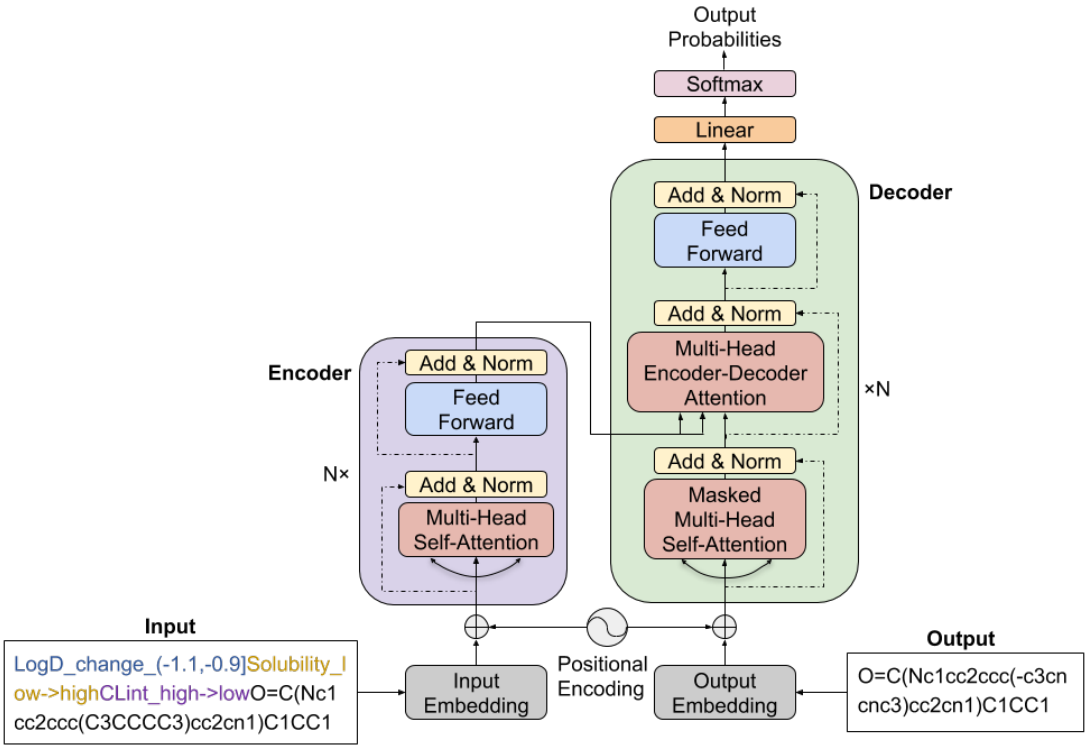



**Figure 5** The Transformer architecture (following Vaswani et al. [35]) for molecular optimization

of the value vectors of all tokens in the input sequence is obtained as the attention vector for the token at position $t$ being processed. This process is for obtaining a single head attention. Multi-head attention [35] was introduced to help to focus on the input from different perspectives. Specifically, multiple weight matrices are learned to project the input embedding into multiple sets of Q, K, V, which are then used to derive multiple attention vectors. These attention vectors are then concatenated and projected to produce the final output of the multi-head self-attention sub-layer, which is then passed through the feed-forward neural network sub-layer. The output from the feed-forward neural network sub-layer is fed to the next encoder layer.

**Decoder** Similar to the encoder, the decoder consists of a stack of N identical decoder layers. Each decoder layer has three sub-layers, masked multi-head self-attention sub-layer, fully connected feed-forward network sub-layer, and encoder-decoder multi-head attention sub-layer. The decoder operates in a similar fashion to the encoder, except that the attention differ from those in encoder in the following: (i) while self-attention in encoder allows each position to attend all positions from previous encoder layer, self-attention in decoder only allows each position to attend earlier positions by masking the future positions. (ii) An additional encoder-decoder multi-head attention was introduced to help the decoder focus on specific parts of the input sequence, which is similar to the role of encoder-decoder attention mechanism in Seq2Seq with attention. Specifically, the output of the top encoder layer is transformed into a set of vectors K and V, which is used by each encoder-decoder multi-head attention sub-layer.

Model training and inferencing was performed on a NVIDIA Tesla K80. The hyperparameters were tuned, and most remained the same as [35], except that the input and output encoding dimension was changed from 512 to 256, and label smoothing was changed from 0.1 to 0. Similar to Seq2Seq with attention, the model was trained using teacher forcing, and multinomial sampling was used to generate multiple molecules for a given input sequence.

## Data Preparation

The models are trained on a set of MMPs extracted from ChEMBL together with the property changes between the source and target molecules.

### *Constructing Matched Molecular Pairs*

The matched molecular pairs (including reverse transformations) are extracted from ChEMBL using an

open-source matched molecular pair tool [40]. All the molecules were standardized using MolVS [41]. There are 9,927,876 pairs considering the following constraints,

- the number of heavy atoms of the core $< 50$
- the number of heavy atoms in R group $< 13$
- the ratio of heavy atoms in the R group to the molecule $R_{group} < 0.33$
- the number of H-bond donors in R group $< 3$
- the number of H-bond acceptors in R group $< 3$
- AstraZeneca's AZFilter="CORE" [42] to filter out bad-quality compounds
- each molecule's property values are within 3 standard deviations of all molecules' property values

2% from the full 9,927,876 pairs are randomly sampled for comparing three models because HierG2G does not scale well on large data. We split it into 90% as training and validation, and 10% as test, and further split the 90% into 90% as training and 10% as validation, which results in 160,831 training, 17,871 validation and 19,856 test.

*ADMET Property Prediction Model*

The property prediction models are built based on message passing neural network [43]. They are used for constructing data during training and also for evaluating the generated molecules during testing. In particular, in-house experimental data are used for building property prediction models. Table 1 shows the train and test size, root-mean-square error (RMSE), normalized RMSE (NRMSE) and $R^2$ for each property prediction model. More results on the experimental properties and predicted properties can be found in Figure S1 in Supplementary.

**Table 1** Property prediction model performance

| | LogD | Solubility | HLM CLint |
|---|---|---|---|
| Train size | 170,337 | 184,883 | 144,300 |
| Train RMSE | 0.304 | 0.485 | 0.264 |
| Train NRMSE | 0.041 | 0.079 | 0.083 |
| Train $R^2$ | 0.935 | 0.774 | 0.749 |
| Test size | 18,927 | 20,543 | 16,034 |
| Test RMSE | 0.395 | 0.602 | 0.350 |
| Test NRMSE | 0.054 | 0.104 | 0.113 |
| Test $R^2$ | 0.892 | 0.658 | 0.557 |

## Experimental Settings

*Test Sets*

Each test sample $(X, Z)$ consists of two parts, the starting molecule being optimized $X$ and the desirable property change $Z$, which therefore determines the input data space $\mathcal{X}$. In order to evaluate our models comprehensively, three test sets are constructed:

- **Test-Original** is the original test set $\{(X, Z)_{test}\}$ (10% of dataset) with 19,856 samples, where the desirable property changes are determined by the MMPs in the test set. It has the same input space as the training set, which is typical in machine learning models. But note that each test sample $(X, Z)_{test}$ has not been unseen in the training set. This test set is used to test if our models can generalize well on unseen different combinations of $X$ and $Z$ in the input space $\mathcal{X} \times \mathcal{Z}$.
- **Test-Molecule** is a subset of Test-Original with 12,721 samples where the starting molecules are not seen in the training set, $\{(X, Z)_{test} | X \notin \mathcal{X}_{train}\}$ where $\mathcal{X}_{train}$ represents the set of source molecules in the training set. This set is used to test if our models can generalize well on unseen $(X, Z)$ with further constraint of unseen starting molecules.
- **Test-Property** consists of 7,813 starting molecules with low *solubility*, high *CLint*, and *logD* between 2 and 4.4 in Test-Original. We are interested in optimizing the starting molecules to achieve lower *logD*, high *solubility* and low *CLint*. The desirable target *logD* is further constrained to be between 1.0 and 3.4 because the in-house experimental *logD* lie mostly in this range. The desirable property change for all starting molecules is set to LogD_change_=(-1.1,-0.9], Solubility_low→high, CLint_high→low. Therefore, the test set can be represented by $\{(X, Z\} | (X \in \mathcal{X}_{test}) \wedge (solubility(X) = low) \wedge (CLint(X) = high) \wedge (logD(X) \in [2.0, 4.4]) \wedge (Z = LogD\_change\_(-1.1, -0.9] Solubility\_low \rightarrow high\ CLint\_high \rightarrow low)\}$. This test set is used to test if our model can generalize well on a particular property change we are interested in.

*Evaluation Metrics*

Aligning with our goal, the models are evaluated in two main aspects,

- **Satisfying all three desirable properties.** For each starting molecule in the test set, 10 unique valid molecules, which are not the same as the starting molecule, were generated, and the number of molecules satisfying all three desirable properties out of the 10 generated molecules was counted. The ADMET property prediction model described earlier is used to compute the properties of generated molecules. Additionally, the model error (Test RMSE) in Table 1 is considered to determine if a generated molecule satisfies its desirable properties. For *logD*, the generated molecules with $|logD_{generated} - logD_{target}| \leq 0.4$ will be considered as satisfying desirable *logD* constraint. For

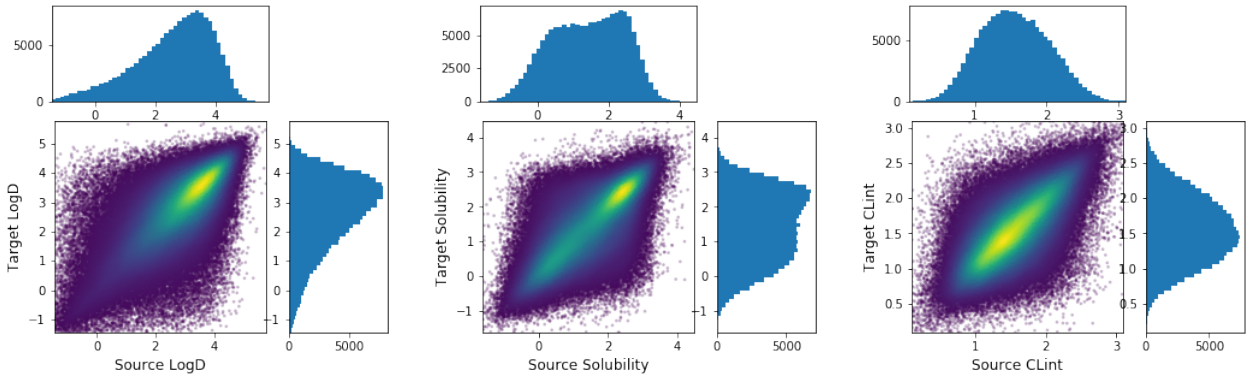



**Figure 6** Source vs. target molecule's properties and their distributions on the training set (*solubility* and *CLint* are in $log_{10}$ scale)

*solubility*, the threshold for low and high will be a range considering the model error, *i.e.* 1.7±0.6. The generated molecules with *solubility* $\leq 2.3$ will be considered as low, and those with *solubility* $\geq 1.1$ will be considered as high. Similarly, for *CLint*, the threshold is 1.3±0.35.

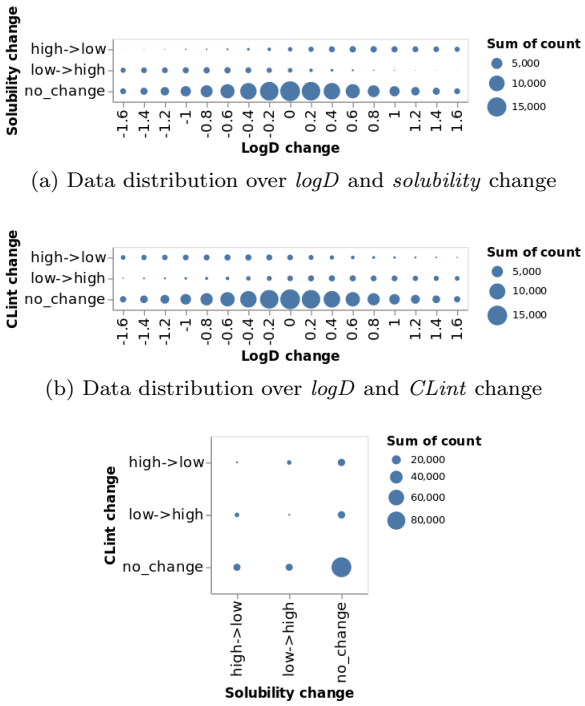
- **Generation of MMPs.** The MMP analysis was performed on the starting molecules and generated molecules to check if the generated molecules have single transformation to the starting molecules. Furthermore, the ratio of heavy atoms in the R group to the generated molecule $R_{group} < 0.33$ and $R_{group} < 0.50$ are examined.

*Baseline*

We compare our models, Seq2Seq and Transformer with the baseline HierG2G. HierG2G training and inferencing was performed on a NVIDIA Tesla V100. The hyperparameters were tuned, and most remained the same as [38], except that *beta* was changed from 0.3 to 0.6.

*K-Sample Anderson–Darling Test*

The $K$-Sample Anderson–Darling Test [44] is a nonparametric test for testing if $k$-samples are drawn from the same population without having to specify the distribution function of that population. It is applicable to continuous and discrete data. This test is used to compare different models' performance in terms of satisfying all three desirable properties.




(a) Data distribution over *logD* and *solubility* change

(b) Data distribution over *logD* and *CLint* change

(c) Data distribution over *solubility* and *CLint* change

**Figure 7** Data distribution over pairwise property changes on the training set where the circle size corresponds to counts. In (a) and (b), each tick $x$ in horizontal axis represents $(x-0.1, x+0.1]$, *e.g.* 0 represents $(-0.1, 0.1]$. For ease of presentation, $x < -1.6$ and $x > 1.6$ are not shown here.

## Results and Discussion

### Data Statistics

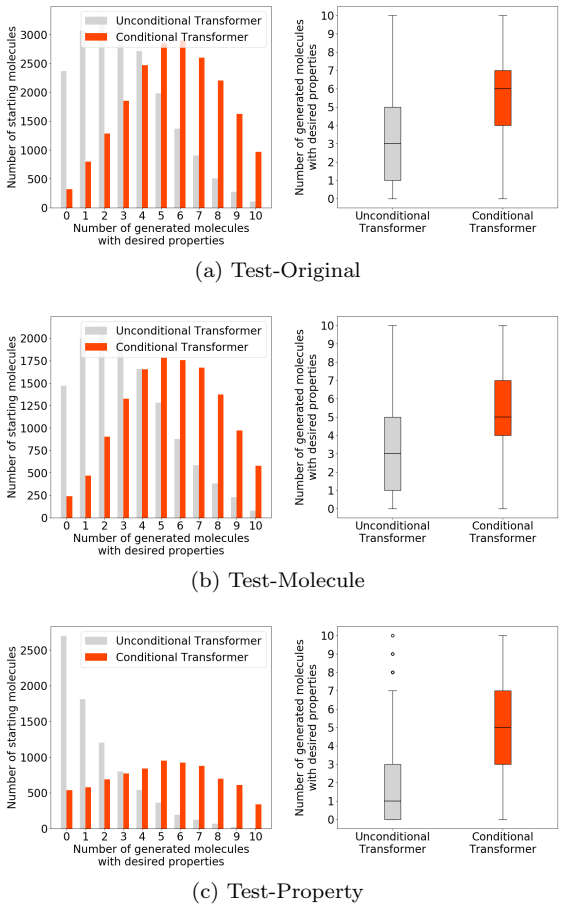
Figure 6 shows the source and target molecule's properties distribution on the training set. The property distribution for the source molecules is the same as the one for the target molecules because reverse transformations are included in the data set. Figure 7 shows the training data distribution over pairwise property changes where most MMPs result in no change in properties. However, such MMPs are still useful because we could generate molecules with same properties, but different structures. It would also be useful when it is desired to keep some properties unchanged

and change some other properties. Additionally, it can be seen from Figure 7a and Figure 7b that *solubility* tends to be negatively correlated with *logD*, and *CLint* tends to be positively correlated with *logD*.

Figure 10a shows the top 20 most frequently occurring transformations on the training set, which are encoded as SMIRKS [40]. The most frequently occurring transformation is [*:1][H]>>[*:1]C where a hydrogen is replaced by a methyl group and its reverse transformation. Note that they do not occur with exactly same frequency because we sampled 2% from the full MMPs. Table 2 shows the statistics of transformations on the training set where around 51.9% of transformations only occur once. The top 20 most frequently occurring transformation only accounts for around 8.2% of the training set, which means there are no dominant transformations.

**Table 2** Transformation statistics on the training set

| | |
|---|---|
| Percentage of unique transformations | 59.4% |
| Percentage of transformations that occur only once | 51.9% |
| Percentage of the most frequently occurring transformation | 1.2% |
| Percentage of the top 20 most frequently occurring transformations | 8.2% |




(a) Test-Original

(b) Test-Molecule

(c) Test-Property

**Figure 8** Number of generated molecules with desirable properties per molecule using unconditional Transformer and conditional Transformer on three test sets. Conditional Transformer outperforms unconditional Transformer using K-Sample Anderson–Darling Test at significance level of 0.1%.

## Unconditional Models vs. Conditional Models

This set of experiment compares conditional models—which use source molecule and property criteria as input, with unconditional models—which use only source molecule as input. For each starting molecule in the test set, 10 unique valid molecules, which are not the same as the starting molecules, were generated. Figure 8 shows the performance of unconditional Transformer and conditional Transformer in terms of satisfying all three desirable properties on three test sets. K-Sample Anderson–Darling Test was performed, and conditional Transformer was found to statistically outperform unconditional Transformer. In particular, 50% of the starting molecules in Test-Original and Test-Molecule have at least 6 molecules with desirable properties out of the 10 generated molecules using conditional Transformer, while the number dropped to 3 using unconditional Transformer. Similar results have been found on the comparison of unconditional Seq2Seq and conditional Seq2Seq (Figure S2).

*Discussion*

*Why does conditional models perform better than unconditional models?* It is shown that using property criteria as additional input can help generating molecules with desirable properties. One reason is that unconditional models are trained only on MMPs $(X, Y)$ without property changes $Z$, with only source molecule $X$ as input and target molecule $Y$ as output. In this case, given a source molecule, it can be mapped to different target molecules with different properties because there could be multiple target molecules that are MMP with the source molecule in the training set. However, when using conditional models, the property change $Z$ between source molecule and target molecule is used as part of input, which guides the model to generate molecules with desirable properties.

*What do the results on three test sets convey?* The performance on Test-Original shows that conditional models can generalize well on the unseen combination of starting molecules and desirable property changes, while the performance on Test-Molecule shows conditional models can generalize well on the combination of unseen starting molecules and seen/unseen property changes. On Test-Property, both unconditional and

conditional models perform worse, especially unconditional models. Note that Test-Property is challenging because only 344 (0.2%) samples out of 160,831 training samples have the particular change. Even in this case, conditional models still perform well with 50% of starting molecules having at least 5 molecules with desirable properties out of the 10 generated molecules.

### Model Comparison for Conditional Models

This set of experiment compares the conditional version of the three models: Seq2Seq, Transformer and HierG2G.

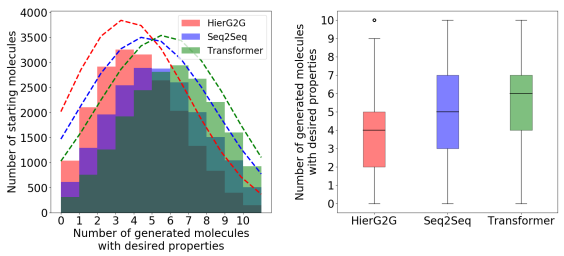
*Satisfying Multiple Desirable Properties*

For each starting molecule in the three test sets, 10 unique valid molecules, which are not the same as starting molecules, were generated. Figure 9 shows the performance of the Seq2Seq, the Transformer and HierG2G in terms of satisfying multiple desirable properties on three test sets. The Transformer outperforms the Seq2Seq and HierG2G on all the three test sets, with more generated molecules satisfying desirable properties.

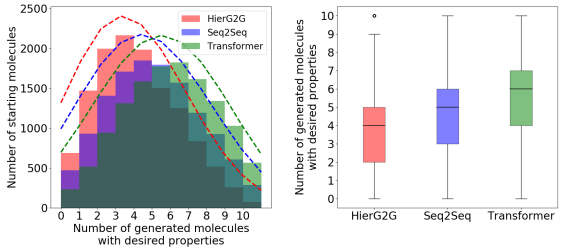
*Generation of MMPs*

For each starting molecule in the test set and the 10 generated molecules, MMP analysis [40] was performed between the starting molecule and each generated molecule. Table 3 shows the percentage of generated molecules that are MMPs with their corresponding starting molecules and the ratio of heavy atoms in R group to the generated molecule $R_{group} < 0.33$ and $R_{group} < 0.50$. Additionally, among all the transformations results from $R_{group} < 0.33$, the percentage of transformations that are seen in the training set is reported. It can be seen that the Transformer generates much more MMPs than the Seq2Seq and HierG2G for all the three test sets.

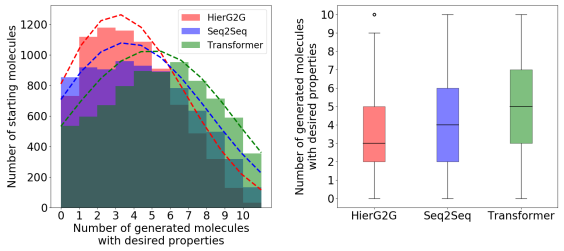
*Discussion* The Transformer generated much more molecules with single transformation to the starting molecules. This mimics the chemist's strategy that applying single transformations when optimizing a starting molecule. Additionally, looking at "in Train", all three models have learned to use not only the existing transformations in the training set but also novel transformations that have not been seen in the training set, to optimize novel combinations of starting molecules and specified desirable properties. Note that the MMP concept is used as a general concept for capturing the chemist's intuition. This does not imply that the MMP concept is the only viable and solely strategy applied, but nevertheless due to its simplicity of linear analoguing it is commonly used. Furthermore, there are several well established algorithms [40] to access and analyze MMPs readily supporting structure–property relationship analysis.




(a) Test-Original




(b) Test-Molecule




(c) Test-Property

**Figure 9** Number of molecules with desirable properties among 10 generated molecules per molecule using conditional version of HierG2G, Seq2Seq and Transformer on three test sets. The difference between each two is significant using K-Sample Anderson–Darling test at significance level of 0.1%.

*Top 20 most frequently occurring transformations generated by Transformer* We further check the top 20 most frequently occurring transformations among the generated molecules from the Transformer, and compare them with the ones on the training set, as shown in Figure 10. Most of the generated transformations on Test-Original and Test-Molecule are very similar to the ones on the training set.

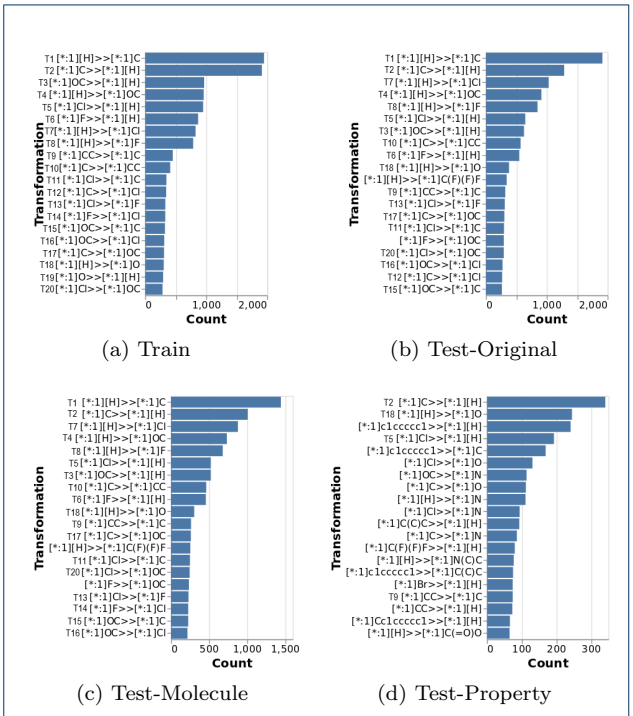
*Discussion* The result indicates that the Transformer model has captured the transformations on the training set. The generated transformations on Test-Property are more different from the ones on the training set compared with the other two test sets. It is reasonable because the input space on Test-Property is very different from the one on the training set due to the constraint of particular property change. The generated transformations on Test-Property are biased to-

**Table 3** Comparison of the percentage of generated molecules from HierG2G, Seq2Seq and Transformer that are MMPs with starting molecules. Among **all the generated molecules** for each test set, **MMP_0.33** and **MMP_0.50** represent the percentage of generated molecules that are MMPs with their corresponding starting molecules and the ratio of heavy atoms in R group to the generated molecule $R_{group} < 0.33$ and $R_{group} < 0.50$ respectively. Among all the transformations results from MMP_0.33, **in Train** represents the percentage of transformations that are seen in the training set.

| | HierG2G | | | Seq2Seq | | | Transformer | | |
|---|---|---|---|---|---|---|---|---|---|
| | MMP_0.33 | MMP_0.50 | in Train | MMP_0.33 | MMP_0.50 | in Train | MMP_0.33 | MMP_0.50 | in Train |
| Test-Original | 52.50% | 61.19% | 45.46% | 71.98% | 81.01% | 44.01% | **89.98%** | **96.12%** | 47.60% |
| Test-Molecule | 25.87% | 30.98% | 54.99% | 65.84% | 76.00% | 52.40% | **88.12%** | **94.94%** | 55.27% |
| Test-Property | 13.04% | 15.20% | 41.49% | 70.57% | 79.45% | 38.00% | **89.13%** | **95.53%** | 36.72% |

**Table 4** Comparison of the percentage of generated **desirable** molecules from HierG2G, Seq2Seq and Transformer that are MMPs with starting molecules. **Desirable** represents the percentage of molecules with desirable properties among all the generated molecules. Among **all these generated molecules with desirable properties** for each test set, **MMP_0.33_D** and **MMP_0.50_D** represent the percentage of generated molecules that are MMPs with their corresponding starting molecules and the ratio of heavy atoms in R group to the generated molecule $R_{group} < 0.33$ and $R_{group} < 0.50$ respectively.

| | HierG2G | | | Seq2Seq | | | Transformer | | |
|---|---|---|---|---|---|---|---|---|---|
| | Desirable | MMP_ 0.33_D | MMP_ 0.50_D | Desirable | MMP_ 0.33_D | MMP_ 0.50_D | Desirable | MMP_ 0.33_D | MMP_ 0.50_D |
| Test-Original | 38.66% | 62.94% | 70.77% | 47.74% | 78.88% | 86.62% | **55.67%** | **92.41%** | **97.41%** |
| Test-Molecule | 37.62% | 56.81% | 65.44% | 45.91% | 73.32% | 82.32% | **54.48%** | **90.96%** | **96.54%** |
| Test-Property | 34.90% | 55.74% | 63.30% | 39.73% | 72.95% | 81.68% | **49.14%** | **90.01%** | **96.36%** |




(a) Train (b) Test-Original

(c) Test-Molecule (d) Test-Property

**Figure 10** Top 20 most frequently occurring transformations on (a) Training set, and the top 20 most frequently occurring transformations among generated molecules from Transformer on (b) Test-Original (c) Test-Molecule and (d) Test-Property.

wards generating molecules with that particular property change.

*MMPs within Generated Desirable Molecules*

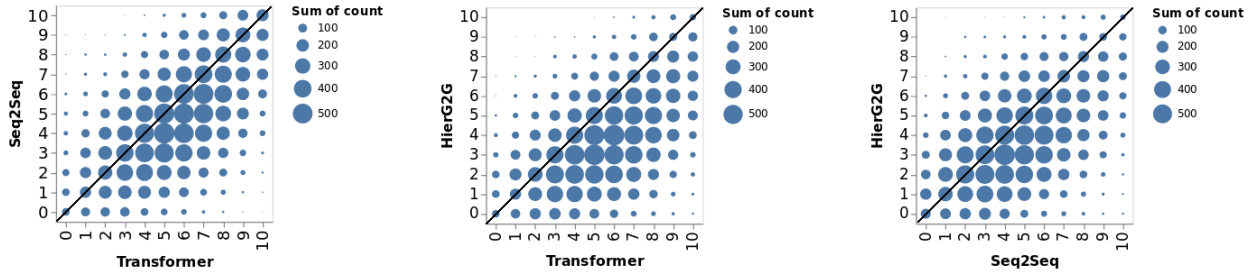
We take a closer look at the generated molecules with desirable properties from three models, and examine the percentage of MMPs (Table 4). The Transformer can generate much more desirable molecules than HierG2G and Seq2Seq, with 15%-17%, and 8%-10% absolute improvement respectively. Within the generated molecules with desirable properties, above 90% of the desirable molecules generated from the Transformer make single transformation to starting molecules and the ratio of the change (R group) compared to the generated molecule is no more than 1/3, while the number dropped significantly to around 73% and above 35% for Seq2Seq and HierG2G respectively. Clearly, the Transformer can generate molecules with desirable properties by making small modifications to starting molecules as a chemist would do.

## Varying Performance of Three Models

Although the Transformer outperforms the Seq2Seq and HierG2G overall, it is not clear if it performs best for each test starting molecule. Therefore, we are interested in the following questions, which will help us understand if it will be beneficial to use all three models together.

1 Does the Transformer always perform best for each starting molecules in the test set?
2 Are the generated molecules from the three models the same or different?

To answer the first question, we examine if one model always generates more desirable molecules for each starting molecule than the other two models.




(a) Transformer vs. Seq2Seq (b) Transformer vs. HierG2G (c) Seq2Seq vs. HierG2G

**Figure 11** Pairwise comparison of HierG2G, Seq2Seq and Transformer on the number of molecules with desirable properties out of 10 generated molecules per molecule on Test-Original. The x axis and y axis represent the number of molecules with desirable properties out of 10 generated molecules. If two models always generate the same number of desirable molecules for each same starting molecule, all points will lie on the diagonal line.

**Table 5** Among all the starting molecules in Test-Original, the percentage of starting molecules for which each model generates either less or more desirable molecules when compared with the other two models.

| | less than the others | more than the others |
|---|---|---|
| HierG2G | 47.43% | 10.23% |
| Seq2Seq | 20.99% | 21.94% |
| Transformer | 9.52% | 45.37% |

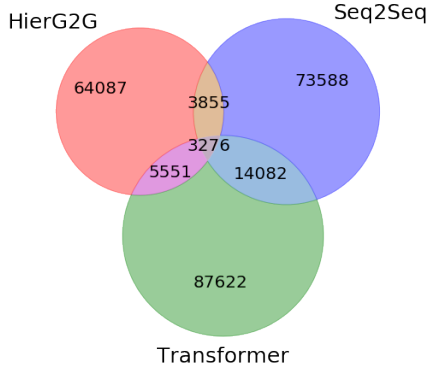
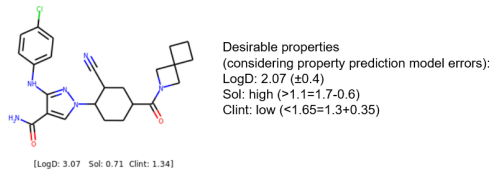



**Figure 12** The overlap of the generated molecules with desirable properties from HierG2G, Seq2Seq and Transformer given each starting molecule in Test-Original. Each circle represents the set of molecules with desirable properties for each model. Transformer generated more desirable molecules than HierG2G and Seq2Seq, with the biggest circle. Most generated molecules with desirable properties from three models given the same starting molecule are different from each other, with only 3,276 identical molecules. Transformer and Seq2Seq have more identical desirable molecules (14,082) compared with (HierG2G, Seq2Seq) and (Transformer, HierG2G).

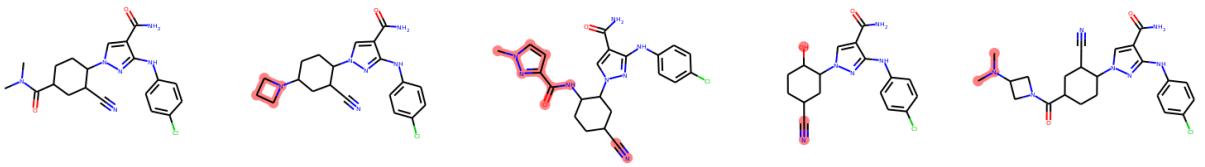
Figure 11 shows the pairwise comparison of three models on the number of molecules with desirable properties out of 10 generated molecules on Test-Original. The Transformer generated more desirable molecules than the Seq2Seq and HierG2G for most starting molecules, with those dots lie below the diagonal line in Figure 11a and Figure 11b. However, there are still some starting molecules where the Transformer generated less desirable molecules. Table 5 shows the percentage of starting molecules for which each model generates either less or more desirable molecules compared with the other two models on Test-Original. For 45.37% of the starting molecules on Test-Original, the Transformer generated more desirable molecules than HierG2G and Seq2Seq. But there are 10.23%+21.94%=32.17% of the starting molecules, where either HierG2G or Seq2Seq generated more desirable molecules. Therefore, it could be beneficial to use all three models together.

The second question was examined to see if the three models can generate diverse desirable molecules. For each starting molecule in Test-Original, we compute the overlapping and non-overlapping set of generated desirable molecules from HierG2G, Seq2Seq and Transformer, and sum the numbers over all starting molecules, which results in the Venn diagram in Figure 12. First, the Transformer generated more desirable molecules than the Seq2Seq and HierG2G. Second, there is not much overlap of the generated desirable molecules among the three models. Third, the Transformer and Seq2Seq generate identical desirable molecules more frequently than the other two pairs. The reason might be that the Transformer and Seq2Seq are both based on SMILES representation and have a more similar working mechanism compared to HierG2G which is based on graph representation. Overall, the three models can generate diversified sets of molecules with desirable properties, which encourages us to use them in an ensemble way to enrich the generated desirable molecules.

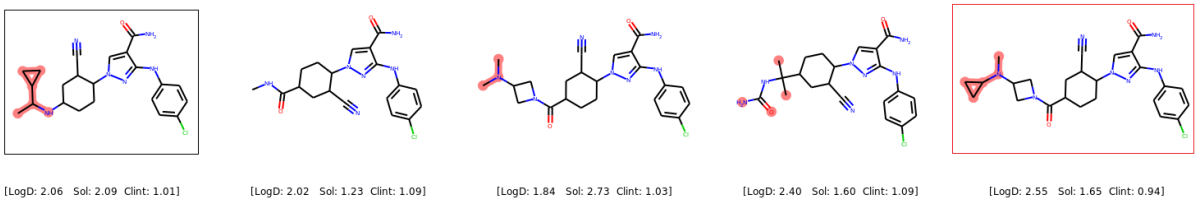
Figure 13 shows an example of the diverse molecules with desirable properties generated by HierG2G,



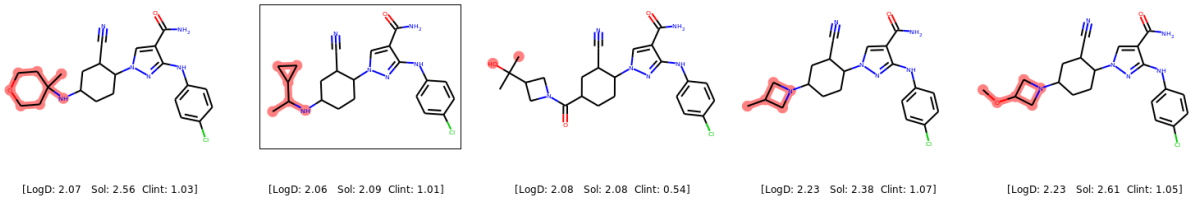

(a) Starting molecule and desirable properties




(b) HierG2G




(c) Seq2Seq




(d) Transformer

**Figure 13** Example of diverse molecules with desirable properties generated by (b) HierG2G (c) Seq2Seq and (d) Transformer. The changes in the generated molecules compared with starting molecule are highlighted in red. The last molecule outlined by red rectangle in (c) is not a desirable molecule because of *logD*. Seq2Seq and Transformer have one molecule overlapping outlined by black rectangle.

Seq2Seq and Transformer. Given the starting molecule [45] and desirable properties in Figure 13a, the three models generate diverse molecules with desirable properties (except the molecule outlined by the red rectangle in Figure 13c) with different small modifications to the starting molecule. The Seq2Seq and the Transformer have one molecule overlapping as outlined by the black rectangles. Overall, we see the diverse set of desirable molecules that the three models generated.

## Conclusions and Future Work

The molecular optimization problem was framed as a machine translation problem where a given molecule is translated into a molecule with optimized properties based on the SMILES representation. Two machine translation models, the Seq2Seq with attention and the Transformer have been investigated to generate molecules with desirable properties by capturing the chemist's intuition, *i.e.* MMPs. The property changes have been incorporated into the input (starting molecule being optimized) as condition to guide

the models to generate molecules with different combinations of property constraints. Given a starting molecule and user-specified properties, our models can generate molecules satisfying multiple property constraints which are structurally similar to the starting molecule. Specifically, for the Transformer, around half of all the generated molecules satisfied all target properties, and within the generated molecules with desirable properties, around 90% had a single transformation with respect to the starting molecules and no more than 1/3 change.

This can be beneficial to lead optimization, where a promising molecule needs to be improved to achieve a balance of multiple properties. The small modifications to the starting molecule, which mimic the chemist's strategy, would be intuitive for chemists and provide insights for designing new molecules. Our deep learning models start from capturing the chemist's intuition from MMPs, but they go beyond the working assumptions of chemists, e.g. transferability where the effect of a chemical transformation is assumed to be generalized to different molecular context. As data-driven approaches, our models can learn the intuitive chemical transformations without explicit assumptions.

The SOTA method for molecular optimization, HierG2G, which is a graph-to-graph translation model was compared with Seq2Seq and Transformer. The Transformer performed best overall in generating more molecules with desirable properties and structurally similar to starting molecules. However, Seq2Seq and HierG2G can still generate different molecules with desirable properties. We believe the ensemble use of three models will contribute to a further enrichment of diverse molecules.

We have extracted a dataset of MMPs from ChEMBL with the source and target molecules' ADMET properties (*e.g. logD*, *solubility* and *clearance*) predicted by the property prediction models trained on a large number of in-house experimental data. We believe it will be beneficial for the studies of MMP analysis and optimizing ADMET properties.

As a proof a concept, we have focused on optimizing three ADMET properties. For future work, additional properties, *e.g.* permeability and bioactivity will be included.

#### Availability of data and materials

All source code and datasets used to produce the reported results can be found online later.

#### Competing interests

The authors declare that they have no competing interests.

#### Author's contributions

Jiazhen He performed the research. She also developed the code for Seq2Seq with attention. Huifang You developed the code and ran experiments for Transformer, while Emil Sandström developed the code and ran experiments for HierG2G. Eva Nittinger provided the scripts for extracting matched molecular pairs. Christian Tyrchan, Werngard Czechtizky and Ola Engkvist proposed and supervised the project. All authors provided helpful feedback on methods used, experiment and results on the project. Jiazhen He wrote the manuscript, and all authors read and approved the final manuscript.

#### Acknowledgements

Jiazhen He thanks the Molecular AI group at AstraZeneca, especially Atanas Patronov and Thierry Kogej for helpful discussions and technical support, Rocío Mercado for helpful feedback on the manuscript. Jiazhen He also thanks the postdoc program at AstraZeneca.

#### Author details

[1]Hit Discovery, Discovery Sciences, R&D, AstraZeneca, Gothenburg, Sweden. [2]Medicinal Chemistry, Respiratory Infammation, and Autoimmune (RIA), BioPharmaceutical R&D, AstraZeneca, Gothenburg, Sweden. [3]Department of Pharmaceutical Biosciences, Uppsala University, Uppsala, Sweden. [4]Institution of Mathematics and Mathematical Statistic, Umeå University, Umeå, Sweden.

#### References

1. Polishchuk, P.G., Madzhidov, T.I., Varnek, A.: Estimation of the size of drug-like chemical space based on gdb-17 data. Journal of computer-aided molecular design **27**(8), 675–679 (2013)
2. Topliss, J.G.: Utilization of operational schemes for analog synthesis in drug design. Journal of medicinal chemistry **15**(10), 1006–1011 (1972)
3. Kenny, P.W., Sadowski, J.: Structure modification in chemical databases. Chemoinformatics in drug discovery **23**, 271–285 (2005)
4. Tyrchan, C., Evertsson, E.: Matched molecular pair analysis in short: algorithms, applications and limitations. Computational and structural biotechnology journal **15**, 86–90 (2017)
5. Weber, J., Achenbach, J., Moser, D., Proschak, E.: Vammpire: a matched molecular pairs database for structure-based drug design and optimization. Journal of medicinal chemistry **56**(12), 5203–5207 (2013)
6. Griffen, E., Leach, A.G., Robb, G.R., Warner, D.J.: Matched molecular pairs as a medicinal chemistry tool: miniperspective. Journal of medicinal chemistry **54**(22), 7739–7750 (2011)
7. Leach, A.G., Jones, H.D., Cosgrove, D.A., Kenny, P.W., Ruston, L., MacFaul, P., Wood, J.M., Colclough, N., Law, B.: Matched molecular pairs as a guide in the optimization of pharmaceutical properties; a study of aqueous solubility, plasma protein binding and oral exposure. Journal of medicinal chemistry **49**(23), 6672–6682 (2006)
8. Hansch, C., Maloney, P.P., Fujita, T., Muir, R.M.: Correlation of biological activity of phenoxyacetic acids with hammett substituent constants and partition coefficients. Nature **194**(4824), 178–180 (1962)
9. Hansch, C., Fujita, T.: p-$\sigma$-$\pi$ analysis. a method for the correlation of biological activity and chemical structure. Journal of the American Chemical Society **86**(8), 1616–1626 (1964)
10. Free, S.M., Wilson, J.W.: A mathematical contribution to structure-activity studies. Journal of Medicinal Chemistry **7**(4), 395–399 (1964)
11. Segler, M.H., Kogej, T., Tyrchan, C., Waller, M.P.: Generating focused molecule libraries for drug discovery with recurrent neural networks. ACS central science **4**(1), 120–131 (2018)
12. Gupta, A., Müller, A.T., Huisman, B.J., Fuchs, J.A., Schneider, P., Schneider, G.: Generative recurrent networks for de novo drug design. Molecular informatics **37**(1-2), 1700111 (2018)
13. 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 **4**(2), 268–276 (2018)
14. Dai, H., Tian, Y., Dai, B., Skiena, S., Song, L.: Syntax-directed variational autoencoder for molecule generation. In: Proceedings of the International Conference on Learning Representations (2018)
15. Lim, J., Ryu, S., Kim, J.W., Kim, W.Y.: Molecular generative model based on conditional variational autoencoder for de novo molecular design. Journal of cheminformatics **10**(1), 1–9 (2018)

16. Jin, W., Barzilay, R., Jaakkola, T.: Junction tree variational autoencoder for molecular graph generation. In: International Conference on Machine Learning, pp. 2323–2332 (2018)
17. Liu, Q., Allamanis, M., Brockschmidt, M., Gaunt, A.: Constrained graph variational autoencoders for molecule design. In: Advances in Neural Information Processing Systems, pp. 7795–7804 (2018)
18. Simonovsky, M., Komodakis, N.: Graphvae: Towards generation of small graphs using variational autoencoders. In: International Conference on Artificial Neural Networks, pp. 412–422 (2018). Springer
19. Guimaraes, G.L., Sanchez-Lengeling, B., Outeiral, C., Farias, P.L.C., Aspuru-Guzik, A.: Objective-reinforced generative adversarial networks (organ) for sequence generation models. arXiv preprint arXiv:1705.10843 (2017)
20. Putin, E., Asadulaev, A., Ivanenkov, Y., Aladinskiy, V., Sanchez-Lengeling, B., Aspuru-Guzik, A., Zhavoronkov, A.: Reinforced adversarial neural computer for de novo molecular design. Journal of chemical information and modeling **58**(6), 1194–1204 (2018)
21. Putin, E., Asadulaev, A., Vanhaelen, Q., Ivanenkov, Y., Aladinskaya, A.V., Aliper, A., Zhavoronkov, A.: Adversarial threshold neural computer for molecular de novo design. Molecular pharmaceutics **15**(10), 4386–4397 (2018)
22. De Cao, N., Kipf, T.: Molgan: An implicit generative model for small molecular graphs. arXiv preprint arXiv:1805.11973 (2018)
23. Olivecrona, M., Blaschke, T., Engkvist, O., Chen, H.: Molecular de-novo design through deep reinforcement learning. Journal of cheminformatics **9**(1), 48 (2017)
24. Jin, W., Yang, K., Barzilay, R., Jaakkola, T.: Learning multimodal graph-to-graph translation for molecular optimization. arXiv preprint arXiv:1812.01070 (2018)
25. Kadurin, A., Nikolenko, S., Khrabrov, K., Aliper, A., Zhavoronkov, A.: drugan: an advanced generative adversarial autoencoder model for de novo generation of new molecules with desired molecular properties in silico. Molecular pharmaceutics **14**(9), 3098–3104 (2017)
26. Blaschke, T., Olivecrona, M., Engkvist, O., Bajorath, J., Chen, H.: Application of generative autoencoder in de novo molecular design. Molecular informatics **37**(1-2), 1700123 (2018)
27. Winter, R., Montanari, F., Steffen, A., Briem, H., Noé, F., Clevert, D.-A.: Efficient multi-objective molecular optimization in a continuous latent space. Chemical science **10**(34), 8016–8024 (2019)
28. Li, Y., Zhang, L., Liu, Z.: Multi-objective de novo drug design with conditional graph generative model. Journal of cheminformatics **10**(1), 33 (2018)
29. Kotsias, P.-C., Arús-Pous, J., Chen, H., Engkvist, O., Tyrchan, C., Bjerrum, E.J.: Direct steering of de novo molecular generation with descriptor conditional recurrent neural networks. Nature Machine Intelligence **2**(5), 254–265 (2020)
30. You, J., Liu, B., Ying, Z., Pande, V., Leskovec, J.: Graph convolutional policy network for goal-directed molecular graph generation. In: Advances in Neural Information Processing Systems, pp. 6410–6421 (2018)
31. Zhou, Z., Kearnes, S., Li, L., Zare, R.N., Riley, P.: Optimization of molecules via deep reinforcement learning. Scientific reports **9**(1), 1–10 (2019)
32. Bahdanau, D., Cho, K., Bengio, Y.: Neural machine translation by jointly learning to align and translate. In: 3rd International Conference on Learning Representations, ICLR 2015 (2015)
33. Sutskever, I., Vinyals, O., Le, Q.V.: Sequence to sequence learning with neural networks. In: Advances in Neural Information Processing Systems, pp. 3104–3112 (2014)
34. Luong, M.-T., Pham, H., Manning, C.D.: Effective approaches to attention-based neural machine translation. In: Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pp. 1412–1421 (2015)
35. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. In: Advances in Neural Information Processing Systems, pp. 5998–6008 (2017)
36. 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 **5**(9), 1572–1583 (2019)
37. Jin, W., Barzilay, R., Jaakkola, T.: Hierarchical graph-to-graph translation for molecules. arXiv, 1907 (2019)
38. Jin, W., Barzilay, R., Jaakkola, T.: Hierarchical generation of molecular graphs using structural motifs. arXiv preprint arXiv:2002.03230 (2020)
39. 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)
40. Dalke, A., Hert, J., Kramer, C.: mmpdb: An open-source matched molecular pair platform for large multiproperty data sets. Journal of chemical information and modeling **58**(5), 902–910 (2018)
41. Swain, M.: MolVS: molecule validation and standardization (2018)
42. Cumming, J.G., Davis, A.M., Muresan, S., Haeberlein, M., Chen, H.: Chemical predictive modelling to improve compound quality. Nature reviews Drug discovery **12**(12), 948–962 (2013)
43. Yang, K., Swanson, K., Jin, W., Coley, C., Eiden, P., Gao, H., Guzman-Perez, A., Hopper, T., Kelley, B., Mathea, M., *et al.*: Analyzing learned molecular representations for property prediction. Journal of chemical information and modeling **59**(8), 3370–3388 (2019)
44. Scholz, F.W., Stephens, M.A.: K-sample anderson–darling tests. Journal of the American Statistical Association **82**(399), 918–924 (1987)
45. Brubaker, J., Dinsmore, C., Hoffman, D.M., Jung, J., Liu, D., Peterson, S., Siu, T., Torres, L.E., Zhang, H., Wei, Z., et al.: Cycloalkylnitrile pyrazole carboxamides as janus kinase inhibitors. Google Patents. US Patent 8,962,608 (2015)

**Abbreviations**

SMILES: Simplified Molecular-Input Line-Entry System Seq2Seq: sequence to sequence SOTA: state-of-the-art ADMET: absorption, distribution, metabolism, elimination and toxicity RNNs: recurrent neural networks VAEs: variational autoencoders GANs: generative adversarial networks MMP: matched molecular pair NLP: natural language processing VJTNN: variational junction tree encoder-decoder HierG2G: hierachical graph encoder-decoder LSTM: long short-term memory HLM CLint: Human Microsome Intrinsic Clearance NLL: negative log likelihood RMSE: root-mean-square error NRMSE: normalized RMSE

**Additional Files**

Additional file 1 — Supplementary figures