Unpaired Generative Molecule-to-Molecule Translation for Lead Optimization

Guy Barshatski
Technion - Israel Institute of Technology
Haifa, Israel
guy.ba@cs.technion.ac.il

Kira Radinsky
Technion - Israel Institute of Technology
Haifa, Israel
kirar@cs.technion.ac.il

ABSTRACT

Molecular lead optimization is an important task of drug discovery focusing on generating novel molecules similar to a drug candidate but with enhanced properties. Prior works focused on supervised models requiring datasets of pairs of a molecule and an enhanced molecule. These approaches require large amounts of data and are limited by the bias of the specific examples of enhanced molecules. In this work, we present an unsupervised generative approach with a molecule-embedding component that maps a discrete representation of a molecule to a continuous space. The components are then coupled with a unique training architecture leveraging molecule fingerprints and applying double cycle constraints to enable both chemical resemblance to the original molecular lead while generating novel molecules with enhanced properties. We evaluate our method on multiple common molecular optimization tasks, including dopamine receptor (DRD2) and drug likeness (QED), and show our method outperforms previous state-of-the-art baselines. Moreover, we conduct thorough ablation experiments to show the effect and necessity of important components in our model. Furthermore, we demonstrate our method's ability to generate FDA-approved drugs it has never encountered before, such as Perazine and Clozapine, which are used to treat psychotic disorders, like Schizophrenia. The system is currently being deployed for use in the Targeted Drug Delivery and Personalized Medicine laboratories generating treatments using nanoparticle-based technology.

CCS CONCEPTS

• **Computer systems organization → Neural networks**; • **Applied computing → Life and medical sciences**; *Chemistry*.

KEYWORDS

AI Lead Optimization, Drug Discovery, ML for Healthcare

ACM Reference Format:
Guy Barshatski and Kira Radinsky. 2021. Unpaired Generative Molecule-to-Molecule Translation for Lead Optimization. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '21), August 14–18, 2021, Virtual Event, Singapore.* ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3447548.3467120

KDD '21, August 14–18, 2021, Virtual Event, Singapore.

ACM ISBN 978-1-4503-8332-5/21/08…$15.00
https://doi.org/10.1145/3447548.3467120

1 INTRODUCTION

Drug development is highly resource intensive, it might take over 10–15 years and more than 2 billion dollars. Potential drug leads are located in a process known as drug discovery. A common approach is to screen enormous number of chemical compounds in high throughput screening (HTS) assays to identify a modification in the target activity. This task is extremely challenging as the number of synthetically valid chemicals which are potentially drug-like molecules is estimated to be between $10^{23} - 10^{60}$ [21]. To overcome this challenge, a lot of focus was recently aimed at the task of lead optimization [6, 10, 11, 18]. In this task, a drug candidate is identified by HTS or using chemistry expertise to have some of the desired properties, and the goal is to explore similar molecules with better properties than the original candidate.

Initial machine-learning modeling of the problem were unsupervised approaches [10, 18], that showed low-performance results. The current state-of-the-art (SOTA) approaches involve the acquisition of a set of paired molecules – the original molecule and an enhanced molecule with more desirable properties. The goal is to learn a generative model to produce novel molecules with similar chemical properties to the input molecule but with enhanced properties. Specifically, SOTA approaches for this task formulate the problem as a graph-to-graph translation problem [6, 11]. One of the drawbacks of the paired approaches is the need for paired data for training, thus forcing a specific mapping between source and destination molecules although there may be various ways to translate an input molecule to other molecules with improved property. Furthermore, sometimes paired data in sufficient amounts is not available. In this work, we present an architecture that does not require any paired data reaching significant performance gains over graph-to-graph translation models.

We present the Unpaired Generative Molecule-to-Molecule (UGMMT) model – an end-to-end generative model trained to generate novel discrete molecules with desired properties. The architecture is composed of three components: (1) Given a discrete string representation of a molecule, it generates the molecule's continuous embedding. In this work, we leverage SMILES [28] for the molecule discrete representation – a popular ASCII string molecular representation that preserves the molecule's chemical information. (2) Using a translation component trained under double cycle constraints, it converts the continuous representation to an embedded molecule with desired properties. A novel molecular attention mechanism is added to drive the molecule generation to keep similarity to the original molecule's chemical fingerprints (fp). (3) Converting the embedded molecule back to a discrete SMILES string representing a molecule with the desired properties.

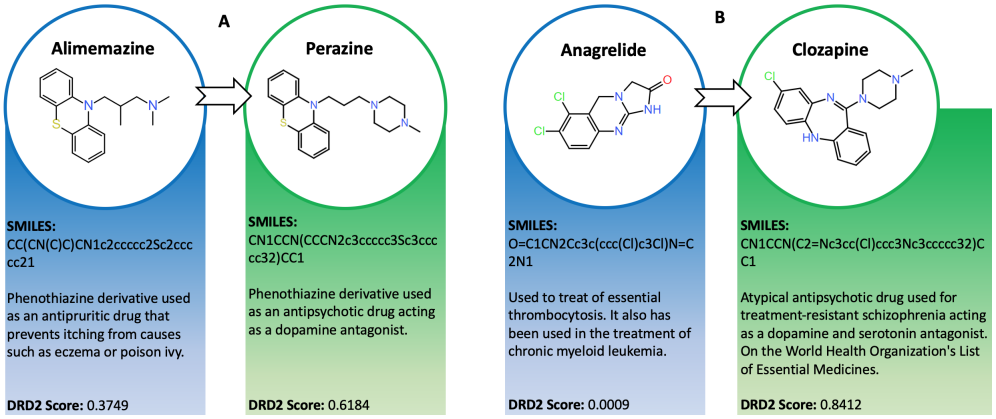

Figure 1: FDA approved drugs generated using UGMMT.

We perform empirical evaluation of our model on several common lead optimization tasks and present significant performance gains compared to various SOTA. In addition, we conduct comprehensive ablation experiments to further analyse our method. We also present the system ability to generate FDA-approved drugs, although it has never seen a drug before. Figure 1 presents several such examples. D2 is the main receptor for most antipsychotic drugs [27]. Any disorder in equilibration of the D2 receptor's states may lead to diverse serious disorders, such as Schizophrenia, autism and Parkinson's disease. DRD2 score is a measure of a molecule's biological activity against a biological target named the dopamine type 2 receptor. As input, our algorithm received a molecule with low DRD2 score and the goal was to generate a molecule with higher DRD2 score. All drugs have never been seen by our model during training, nor validation. In Fig. 1A, given Alimemazine (DRD2 score 0.3749), the algorithm generated Perazine (DRD2 score 0.6184) reaching 40% fingerprint similarity to Alimemazine. Both drugs are phenothiazine derivatives and structurally related to Chlorpromazine, older remedy for Schizophrenia. In Fig. 1B, Clozapine was generated using the prototype drug Anagrelide with 20.96% similarity and DRD2 score improvement from 0.0009 to 0.8412. Clozapine is used for treatment-resistant Schizophrenia and is on the World Health Organization's list of "essential, safest and most effective medicines" [19]. Interestingly, all drugs generated by our model to improve DRD2 score are indeed antipsychotic drugs.

The contributions of this work are threefold: (1) We design a novel unsupervised end-to-end generative model and a unique double-cycle training scheme for molecule translation preserving molecule similarity using molecular fp utilization, without the need of a large paired dataset; (2) We present empirical results on the DRD2 (dopamine receptor) and QED (drug likeness) properties and demonstrate superior performance over SOTA baselines in generating novel, chemically similar molecules with higher property scores and success rates; (3) We perform retrospective experiments to demonstrate our model's ability to produce FDA approved drugs, it has never seen before. Additionally, in collaboration with the Technion laboratory for targeted drug delivery and personalized medicine technologies, additional generated molecules are being tested today for impact for personalized treatments. We believe our method lays the foundations to an automatic-algorithmic HTS process to enable lead optimizations. All code and data in this work are published on our GitHub[1] for further research.

2 RELATED WORK

Early work formulated molecule generation as a sequence generation problem. Molecules were represented as a sequence of characters. Most commonly, SMILES notation was used and showed merit for this task [5, 7, 8, 15]. The aim of these works was to generate valid molecules rather than molecular optimization. Applying these models naively for the latter has yielded poor performance [6].

Molecular optimization methods mainly focused on graph-to-graph translation methods. Those represent molecules as graphs and try to translate input molecular graphs into improved graphs. Jin et al. [10] suggest an encoder-decoder architecture, JTVAE. Given input molecule's graph, the encoder generates a tree where each node represents molecule's substructure, then embeds both to get the latent representation. The decoder reproduces the tree and uses it to predict the output molecular graph. Optimization is done by first training a property score predictor on top of JTVAE's latent space, then gradient ascent is applied on input molecule's embedding to improve its score. Maziarka et al. [18] present MolCG as an extension of JTVAE. Instead of using gradient ascent, the model splits the embedded molecules to two distinct sets – with and without the enhanced property. Two sets of generators and discriminators are applied in an adversarial manner introduced by Zhu et al. [31] to perform the optimization. At test time, the generated embeddings are decoded back to molecules using the same JTVAE latent space decoder. Although showing initial success, the results were not sufficient for practical molecular optimization. To improve the performance, numerous supervised methods have been suggested requiring substantial amount of training data. Jin et al. [11] extended JTVAE by presenting attention mechanism during the tree decoding procedure and integrating adversarial training to further improve performance and thus generate molecules with improved desired property. Fu et al. [6] (CORE) extended the above by introducing the copy&refine technique where the generator at each step decides whether to copy the next substructure from the input molecule or to generate a new substructure. This relaxes the

[1] https://github.com/guy-ba/UGMMT

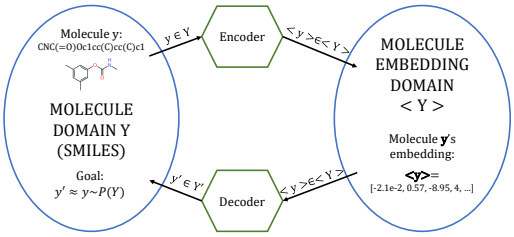

Figure 2: METN. Left: Discrete molecule domain Y represented by SMILES strings. Right: Molecule's continuous representation space $\ddot{Y}\ Y\ _{j}$, i.e. embedding.

problem of challenging substructure prediction from a large set in every iteration, which is especially problematic for infrequent substructures. These approaches require paired data for training, thus forcing a specific mapping between source and destination molecules although there may be various ways to translate an input molecule to other molecule with an improved property. Furthermore, sometimes paired data in sufficient amounts is not available. In this work, we will present an unsupervised approach, requiring no paired-data, which reaches SOTA results outperforming the supervised paired-approaches. We present a novel architecture which does not rely on molecular graph structures, but rather leverages raw SMILES representations, embeds them and learns to efficiently translate them using a novel molecular attention driven by molecular fp to an optimized SMILES molecule.

3 METHODS

We denote a domain of molecules by a capital Latin letter, e.g. X, their distribution by $p(X)$, and a molecule taken from this domain by a small Latin letter, x. That is, $x \in X$ denotes x is a sample of molecules taken from domain X and consequently $x \sim p(X)$. Similarly, we denote the embedding vector of a molecule x by $\ddot{Y}\ x\ _{j}$. It belongs to the domain of all the embedded molecules of X, which we denote by $\ddot{Y}\ X\ _{j}$ with a distribution $p(\ddot{Y}\ X\ _{j})$. Given a domain A of input molecules with some property and distribution $p(A)$, and a domain of the enhanced molecules, B, with distribution $p(B)$ our goal is to learn a mapping $M : A \rightarrow B'$, s.t. the distribution of molecules $B' = M(A)$, $p(B')$, is indistinguishable from the distribution $p(B)$. To achieve it and guarantee that the input molecule $a \sim p(A)$ and output $b' = M(a) \sim p(B')$ carry chemical molecular similarity, we enforce double cycle constraints, that share common translation components strongly relying on molecular fp. This procedure can be applied if molecules are represented in a continuous space. We therefore first present a Molecule-Embedding translation, where a discrete molecule's SMILES representation is transformed to an embedding and vice-versa (Section 3.1). We then present the Embedding-Embedding translation where embedded prototype molecule is translated to an embedded desired molecule and vice-versa (Section 3.2). Lastly, we provide the overall end-to-end architecture of our model (Section 3.3).

3.1 Molecule-Embedding Translation Network (METN)

We propose METN to translate between molecule's discrete SMILES representation and its continuous representation, i.e., embedding.

Algorithm 1 METN Training Algorithm.

Input: training set of molecules Y.

for $epoch = 1, 2, \ldots, E_{METN}$ **do**
  Sample mini-batche $y \in Y$
  $\ddot{Y}\ y\ _{j} = Encoder(y)$
  $y' = Decoder(\ddot{Y}\ y\ _{j})$
  $L = \lambda_0 \cdot CE(y', y) + KL(\ddot{Y}\ y\ _{j}\ ||N(0, I))$
  Minimize L using Adam optimizer
end for

The general structure is depicted in Figure 2. Given a molecule's SMILES representation (denoted by $\mathbf{y} \in Y$), an encoder generates its embedding (denoted by $\ddot{Y}\ \mathbf{y}\ _{j} \in \ddot{Y}\ Y\ _{j}$) and a decoder reverts the process – generating molecule's SMILES string given its embedding. This network is based on Variational Auto Encoder (VAE) [14] structure with a bidirectional GRU-based encoder and a multi-layered GRU-based decoder [3] creating a sequence-to-sequence translation trained using the teacher-forcing method [29] in order to increase validity. VAEs add stochasticity to the generation process adding variation to the latent representation learning and thus forcing the decoder to learn how to decode a wide range of latent points better suiting for embeddings. In addition, to make the latent space dense, a KL divergence term is added to the loss function, encouraging the encoder to distribute molecule encodings according to a known prior distribution. METN's training algorithm is shown in Algorithm 1. λ_0 is a regularization parameter, CE is the cross-entropy loss (reconstruction loss), KL is the KL-divergence (has close form in this case of normal distribution) and the loss is averaged over the number of samples in the mini-batch. In our system, we have two input molecule domains, A and B. Embedding all the molecules using only one METN would make the overall model training and convergence more difficult yielding worse results since the latent embedding space would not only have to represent molecular similarity but also molecule property. Hence, we enable "domain embedding specialization" by training two METNs, one for each domain as depicted in Figure 3. We denote domain A encoder as En_A. The encoder, given a molecule $a \in A$, generates its embedding $\ddot{Y}\ a\ _{j} \in \ddot{Y}\ A\ _{j}$ (a continuous vector), while the decoder, De_A, converts molecule continuous embedding $\ddot{Y}\ a'\ _{j} \in \ddot{Y}\ A'\ _{j}$ to its SMILES $a' \in A'$. We expect that $a' \sim p(A)$, i.e., a' has domain's A property or alternatively, the distribution of the decoded molecules domain A', $p(A')$, is indistinguishable from $p(A)$. Similar notations are used for domain B.

3.2 Embedding-Embedding Translation Network (EETN)

We propose EETN to translate between embedded molecules that belong to different domains, e.g. low and high DRD2 score. Following our notations, the EETN is composed of two translation networks. T_{AB} for $\ddot{Y}\ A\ _{j} \rightarrow \ddot{Y}\ B'\ _{j}$ translation and T_{BA} for $\ddot{Y}\ B\ _{j} \rightarrow \ddot{Y}\ A'\ _{j}$. the opposite direction. Figure 3 illustrates the architecture. We mark in dotted lines the training paths and in thick lines the inference paths. Our design for the EETN allows it to meet two main goals – property enhancement and preservation of the input-output molecular similarity. Given $\ddot{Y}\ a\ _{j}$, the EETN attempts to produce an embedded molecule $\ddot{Y}\ b'\ _{j}$, s.t. $b' = De_B(\ddot{Y}\ b'\ _{j}) \sim p(B)$,

i.e., b' has domain's B enhanced property (e.g. high DRD2 score). De_B must be able to successfully decode $\ddot{Y}\ b'\ _j$ and T_{AB} must successfully translate $\ddot{Y}\ a\ _j \in \ddot{Y}\ A\ _j$ into $\ddot{Y}\ b'\ _j$. During training, naively applying on a the sequence of components on the thick lines ($En_A \rightarrow T_{AB} \rightarrow De_B$) would not be successful. De_B is not necessarily producing valid molecules, therefore calculating their property score is not feasible. Furthermore, molecule's property calculation is not differentiable, hence penalizing molecule's property is not feasible for valid molecules as well. Therefore during training, we apply the sequence $En_B \rightarrow T_{BA} \rightarrow T_{AB} \rightarrow De_B$ (purple dotted path) on b. Requiring $b' \approx b$ might successfully train De_B, however T_{BA} would produce embedded molecules in domain $\ddot{Y}\ A'\ _j$, which may be distributed differently from $\ddot{Y}\ A\ _j$. Hence, we train two translation components simultaneously. In each iteration we train by applying two semi-shared sequences of components on the inputs a and b. We apply the sequence $En_B \rightarrow T_{BA} \rightarrow T_{AB} \rightarrow De_B$ (purple dotted path) on b and $En_A \rightarrow T_{AB} \rightarrow T_{BA} \rightarrow De_A$ (red dotted path) on a and require $b' \approx b$, and $a' \approx a$. This coupling between the two translation sequences sharing T_{AB} and T_{BA}, where one's output is the other's input, encourages proximity of the distribution of $\ddot{Y}\ A\ _j$ and $\ddot{Y}\ A'\ _j$. Intuitively, if T_{BA} translates $\ddot{Y}\ b\ _j \in \ddot{Y}\ B\ _j$ to $\ddot{Y}\ a'\ _j 8 \ddot{Y}\ A\ _j$ then T_{AB} (trained to translate $\ddot{Y}\ a\ _j \in \ddot{Y}\ A\ _j$) will perform poorly on the input $\ddot{Y}\ a'\ _j$ and produce b' notably different from b, which will be penalized by the loss function. Applying these two sequences on the inputs and demanding input-output proximity constitute our double cycle constraints, which we refer to as *double-cycle training scheme*. The technique can be considered a dual learning method [9], which leverages primal (e.g. $A \rightarrow B$) and dual (e.g. $B \rightarrow A$) tasks to create an informative feedback loop.

To encourage similarity to the original molecule, we wish the translation to keep its chemical characteristics. We leverage Morgan fp [23] to represent these characteristics. We introduce the input molecule fp to the translators (concatenated with the input embedding) during the training and the inference. As a result, their output latent embedding spaces become fp dependant, encouraging molecules with similar fp to be closely embedded. The translators T_{AB} and T_{BA} are designed to create a bottleneck by downsampling followed by upsampling architecture in order to extract vital information while dropping the redundant data. To make this process more efficient we pass the input molecule fp through an attention mechanism of a fully-connected (FC) layer followed by a softmax layer producing a weight vector, which multiplies the fp vector elementwise, highlighting the "important" information inside the fp vector. We refer to this component as *molecular attention*.

3.3 End-to-End Architecture

We introduce the UGMMT– an end-to-end unsupervised generative deep neural network architecture. The architecture is illustrated in Figure 3. The architecture consists of two METNs, one for domain A (blue) and one for domain B (green), and an EETN (red) for molecular translation. The input molecules are given as discrete SMILES strings, however, EETN's functionality described in Section 3.2 relies heavily on embedded molecule domains. Therefore, METNs enable converting discrete SMILES strings to continuous representations (embedding) and vise-versa.

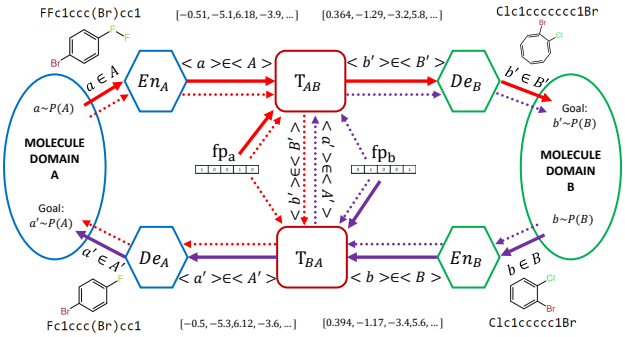

Figure 3: Molecule-to-Molecule end-to-end architecture. Consists of two METNs (two sets of $En + De$) and an EETN (two Ts). The two training paths are marked with dotted arrows while the inference paths are marked with thick solid arrows.

Training: We pre-train the METNs before training the end-to-end model to enable the preparation of the latent embeddings of domain A (e.g., low DRD2 scored molecules) and domain B (e.g., high DRD2 scored molecules) separately. We conjecture this allows the encoders (En_A and En_B) to produce better embeddings and the decoders (De_A and De_B) in turn would produce more valid molecules with the enhanced property. Our experiments (Section 5.2) support this hypothesis. We then perform end-to-end training, where the METNs and the EETN are trained together. Intuitively, this allows the embeddings generation and the translation between them to evolve simultaneously to decrease the overall loss. This enables generating a similar molecule with the desired property. The detailed algorithm is presented in Algorithm 2. CE is the cross-entropy loss, L_1 and L_2 are the two cycle constraints (Section 3.2) and λ_1 is a regularization parameter that controls their relative importance. The loss is averaged over the number of samples in the mini-batch.
Inference: Given a SMILES-represented molecule a to be enhanced from domain A, the En_A converts it to an embedding. The translator T_{AB} then translates the embedding along with the molecule fp (extracted from original molecule a using RDkit) to an embedded molecule of domain B'. Finally, De_B converts it to a molecule in domain B', represented by a SMILES string. We draw the reader attention that the double cycle constraints enforced during the training process enables bidirectional optimization, i.e., inference from B to A as well. Both optimization paths are shown in Figure 3, $A \rightarrow B$ in red thick arrows and $B \rightarrow A$ in purple thick arrows. Algorithm 3 presents the inference algorithm for $A \rightarrow B$ direction ($B \rightarrow A$ is symmetric).

4 EXPERIMENTAL SETTINGS

We provide implementation and hyperparameter details for reproducibility, the datasets we use and the baselines we compare to.

4.1 Implementation Details

Hardware: Nvidia GeForce RTX 2080 Ti 11GB GPU, 2 Intel Xeon Gold 6230 2.10GHZ CPUs, 64GB RAM. **Software**: Ubuntu 18.04.5, Pytorch 1.4.0 [20], Python 3.6.12. **Reproducibility**: We set seed 50, other seeds yield similar results. Hyper-parameters are set following

Algorithm 2 UGMMT Training algorithm. Corresponds to both dotted paths in Figure 3, the red path along $En_A \rightarrow T_{AB} \rightarrow T_{BA} \rightarrow De_A$ and the purple path along $En_B \rightarrow T_{BA} \rightarrow T_{AB} \rightarrow De_B$.

Input: A_t, B_t molecule training sets. A_v molecule validation set.
Train domain's A METN on A_t for E_{METN} epochs (Algorithm 1)
Train domain's B METN on B_t for E_{METN} epochs (Algorithm 1)
for $epoch = 1, 2, \ldots, E_{max}$ **do**
  Sample mini-batches $a \in A_t$, $b \in B_t$
  $\ddot{Y}\ a\ _j = En_A(a)$, $\ddot{Y}\ b\ _j = En_B(b)$
  Calculate fp_a, fp_b
  $\ddot{Y}\ b'\ _j = T_{AB}(\ddot{Y}\ a\ _j, fp_a)$
  $\ddot{Y}\ a'\ _j = T_{BA}(\ddot{Y}\ b'\ _j, fp_a)$
  $a' = De_A(\ddot{Y}\ a'\ _j)$
  $L_1 = CE(a', a)$
  $\ddot{Y}\ a'\ _j = T_{BA}(\ddot{Y}\ b\ _j, fp_b)$
  $\ddot{Y}\ b'\ _j = T_{AB}(\ddot{Y}\ a'\ _j, fp_b)$
  $b' = De_B(\ddot{Y}\ b'\ _j)$
  $L_2 = CE(b', b)$
  $L = \lambda_1 \cdot L_1 + L_2$
  Minimize L using Adam optimizer
  Evaluate the model every V_f epochs on A_v
  if evaluation criterion improves **then**
    save model
  end if
  if no improvement for P evaluations **then**
    stop training
  end if
end for

Algorithm 3 UGMMT $A \rightarrow B$ Inference algorithm. Corresponds to the red thick path along $En_A \rightarrow T_{AB} \rightarrow De_B$ in Figure 3.

Input: $a \in A$ mini-batch of molecules to optimize.
$\ddot{Y}\ a\ _j = En_A(a)$
Calculate fp_a
$\ddot{Y}\ b'\ _j = T_{AB}(\ddot{Y}\ a\ _j, fp_a)$
$b' = De_B(\ddot{Y}\ b'\ _j)$

evaluation on a validation set, where the optimization metric is the Success criterion defined in Section 5.1.

METN: Embedding dimension 256 (experimented with 128, 256, 512), λ_0=10 (experimented with 1, 10, 20), max molecule length during inference 90 (experimented with 80-120). Encoder – bidirectional GRU, hidden dimension 256, followed by 2 FC with 256 neurons, for μ and σ. Decoder – FC layer with 512 neurons, followed by 3-layered GRU (experimented with 1-5), dropout 0.5 between layers, hidden dimension 512 and a FC with vocabulary length neurons. Training– Adam optimizer [13]. Initial learning rate $3 \cdot 10^{-3}$ (experimented with $1 \cdot 10^{-3}$-$5 \cdot 10^{-3}$), final $3 \cdot 10^{-4}$ using cosine annealing scheduler with restart [17] after 10 epochs. Mini-batch size 32. Pre-trained for $E_{METN} = 1$ (QED) (experimented with 0-10) or $E_{METN} = 12$ (DRD2) (experimented with 0-20) epochs.

EETN: Translator – Bottleneck structure. Conv block (filter size 7), stride-2 conv downsampling block (filter size 3), 4 residual blocks (experimented with 4-8), stride-2 transposed conv upsampling block (filter size 3), conv (filter size 7) followed by Tanh and 2 FC with 1,152 and 256 neurons separated by BN, LeakyReLU and Dropout (0.2). It presented outstanding results in image translation and style transfer tasks [12, 31].

Whole model training: Adam optimizer, initial learning rate $3 \cdot 10^{-4}$ (experimented with $1 \cdot 10^{-4}$-$5 \cdot 10^{-4}$) and a linear decay towards 0 from epoch 90 (experimented with 70-100). Mini-batch size 32. Max epochs E_{max}=120 (experimented with 80-120). Data is shuffled during training. Evaluation frequency V_f=3, early stopping with P=15 evaluations. λ_1=2 (experimented with 1,2,4,8,16). Our code is available publicly (see **Appendix A** for more details).

4.2 Datasets

Following [6, 11], we focus on two common properties which are vital for generated drug's effectiveness evaluation:

(1) **Dopamine Receptor D2 (DRD2)**: DRD2 score measures molecule's biological activity against a biological target named the dopamine type 2 receptor.
(2) **Drug likeness (QED)**: QED score [2] measures, intuitively, how "druglike" a molecule is.

We use RDkit [16] library to calculate these scores as done by Fu et al. [6] in order to preserve comparability. We also use their datasets, which were adapted from Jin et al. [11]. For UGMMT and Mol-CG we split the train pairs and randomly sample molecules with low property score (DRD2, QED) (lower decimal) for domain A set and high (high decimal) for B set. For CDN and JTVAE we merge these two sets to get the final train set. UGMMT's validation set was also randomly sampled from the remaining low score molecules, for other baselines the original validation sets from Fu et al. [6] are used. All models are tested on the same test set taken from Fu et al. [6]. Publicly available DrugBank dataset [30], which contains a list of FDA approved drugs, is used for drug generation experiments. We share the datasets publicly and refer to **Appendix B** for details.

4.3 Baseline Methods

We compare to SOTA methods for lead optimization. **CORE** [6] is currently regarded as the SOTA, followed by **G2G** [11]. Both methods are supervised, using paired train sets. UGMMT method is unsupervised, hence we include the leading unsupervised **JTVAE** [10] and **Mol-CG** [18] methods. Following Fu et al. [6], we evaluate the checkpoints on the validation set for highest optimization success (Success), then evaluate on the test set and report the results. All models use datasets which are derived from CORE's datasets with the same test set.

5 EXPERIMENTS AND RESULTS

We conduct three main experiments. First, we evaluate our model's ability to generate valid, novel, diverse molecules which have a desired property and share similarity with a prototype and compare it to the current SOTA methods (Section 5.1). Then, we conduct thorough ablation experiments to show the effect and necessity of important components in our model (Section 5.2). Afterwards, we investigate and demonstrate our model's capability to generate FDA approved drugs with enhanced properties (Section 5.3).

5.1 Main Result: Molecule Optimization

Focusing on the $A \rightarrow B$ optimization direction, for each input test molecule we first generate K output molecules using K random

Table 1: Evaluation results of our method and other baselines for various metrics for QED property. In bold: best optimization Success result, statistically significant with huge effect size ($P - value$ Ÿ 0.05 and Cohen's d ¡ 2).

Type	Method	Property	Similarity	Novelty	Success	Validity	Diversity
Paired	G2G	0.895 ± 0.001	0.320 ± 0.003	0.953 ± 0.004	0.452 ± 0.010	1.000	0.993 ± 0.002
	CORE	0.882 ± 0.002	0.339 ± 0.005	0.963 ± 0.006	0.472 ± 0.012	1.000	0.997 ± 0.002
Unpaired	JTVAE	0.816 ± 0.000	0.304 ± 0.000	0.977 ± 0.000	0.237 ± 0.000	1.000	0.996 ± 0.000
	Mol-CG	0.783 ± 0.000	0.302 ± 0.000	0.980 ± 0.000	0.170 ± 0.000	0.998	1.000 ± 0.000
	UGMMT	0.855 ± 0.001	0.365 ± 0.003	0.997 ± 0.001	**0.513** ± 0.009	0.971	1.000 ± 0.000

Table 2: Evaluation results of our method and other baselines for various metrics for DRD2 property. In bold: best optimization Success result, statistically significant with huge effect size ($P - value$ Ÿ 0.05 and Cohen's d ¡ 2).

Type	Method	Property	Similarity	Novelty	Success	Validity	Diversity
Paired	G2G	0.797 ± 0.005	0.330 ± 0.001	0.379 ± 0.006	0.075 ± 0.005	0.999	0.657 ± 0.009
	CORE	0.758 ± 0.006	0.343 ± 0.002	0.400 ± 0.010	0.087 ± 0.005	1.000	0.648 ± 0.010
Unpaired	JTVAE	0.340 ± 0.000	0.239 ± 0.000	0.991 ± 0.000	0.033 ± 0.000	1.000	0.989 ± 0.000
	Mol-CG	0.382 ± 0.000	0.190 ± 0.000	0.992 ± 0.000	0.013 ± 0.000	1.000	0.775 ± 0.000
	UGMMT	0.826 ± 0.009	0.284 ± 0.001	0.799 ± 0.009	**0.192** ± 0.010	1.000	0.914 ± 0.007

seeds (following the practice of Fu et al. [6], in our experiments $K = 20$). We calculate and report:

- **Validity:** The proportion of the input test molecules which have at least one valid optimized molecule. The validity of a molecule is determined based on Landrum [16].

After that, we randomly select one valid output molecule b' for each input test molecule $a_{te} \in A_{te}$, if exists. Note, following Jin et al. [11], we exclude the input molecules which do not have valid output molecules from the other metrics calculations in order to isolate these from the validity measure. Otherwise, models with lower validity will always have lower metrics. We repeat the random molecule selection and the following metrics calculations 10 times and report their mean and standard deviation values:

- **Property:** The average desired property score (QED or DRD2) of all the optimized b' molecules. The property score of each b' is denoted as $Prop(b') \in [0, 1]$.
- **Similarity:** The average similarity of all (a, b') pairs. The similarity of each molecule pair is measured using Tanimoto similarity [1] over their Morgan fingerprints and denoted as $Sim(a, b') \in [0, 1]$.
- **Novelty:** The proportion of all the optimized b' molecules which are novel. b' is novel if it has not appeared in the train set and $b' < a_{te}$.
- **Optimization Success (Success):** The proportion of all the optimized b' molecules which are successful. b' is successful if it holds similarity, property and novelty simultaneously, i.e., $Sim(a, b')$ ¡ λ_s and $Prop(b')$ ¡ λ_p and b' is novel. Following the practice of Fu et al. [6], we use their "success rate 1" λ_s with their challenging "success rate 2" λ_p, yielding $(\lambda_s, \lambda_p) : (0.3, 0.8)$.
- **Diversity:** The proportion of all the optimized b' molecules which are unique, i.e., the number of unique b' molecules divided by the number of all b' molecules.

Tables 1 and 2 present these results for QED and DRD2 properties, respectively. We observe that for QED, UGMMT has slightly lower validity in QED compared to the other graph-based methods, however, since invalid molecules are easily and automatically disqualified without a human intervention (using RDkit), we mainly focus on the other metrics. For DRD2, UGMMT's validity is excellent. Our method outperforms the SOTA supervised methods in the main optimization success metric, i.e., it generates the highest number of successful molecules. For QED, the main contributors, according to table 1, are UGMMT's better similarity and novelty. Whereas, for DRD2, according to table 2, these are the property and the novelty. For DRD2, UGMMT's improved novelty stands out compared to the supervised methods. Although the supervised methods are able to generate high quality molecules, most of these have already been seen during the training so these models mainly memorize them. UGMMT also has better diversity, meaning that it generates more unique molecules for different input molecules and not generates the same molecule for many leads. The results are surprising as these baselines are trained in a supervised paired manner, where pairs of molecules are carefully selected s.t. their similarity and property improvements are extremely high. On the contrary, our method is unsupervised and only requires two sets of molecules with no similarity requirements. All methods are evaluated on the same test set. Our method also shows significant performance gains compared to the unsupervised methods. Not only the main optimization success metric, which is much higher, but also the property and the similarity show impressive improvements. In addition, these methods unable to generate different molecules given different seeds hence the zero standard deviation, whereas our method present diverse outputs. Note that in nature, high DRD2 molecules are more rare compared to high QED molecules. Hence, the main challenges are to generate high DRD2 novel molecules and highly similar novel QED molecules. According to the results, UGMMT

manages to successfully meet both challenges. In order to fully verify the results, it is not enough to calculate the P-value. While it indicates whether an effect (statistical significant difference) exists, it does not reveal the size of the effect [26]. Furthermore, with a sufficient large sample, a statistical test will almost always demonstrate a significant difference, unless there is no effect whatsoever. Hence, in addition to the P-value, we calculate Cohen's d effect size [4] and verify that our model's Success is statistically significant ($P - value \ \ddot{Y}\ 0.05$) and has huge effect size (Cohen's $d\ \ i\ 2$) [24] as compared to the other baselines.

5.2 Ablation Experiments

We perform various ablation experiments on UGMMT models for each dataset (DRD2 and QED). Results are presented in Table 3.

(1) No Pre-train experiment, we evaluate the contribution of pre-training the METNs before training the end-to-end model. Pre-training the MENTs enables the preparation of the latent embedding space of domain A and domain B separately. Thus, we hypothesize that the encoders would produce better embeddings and the decoders in turn would produce more valid molecules with the enhanced property. In our ablation we validate this hypothesis and observe that the pre-training has higher effect in the DRD2 model. We conjecture this is due to the fact we pre-trained the DRD2 model for more epochs compared to the QED model. Additionally, we observe that although pre-training harms the novelty, it greatly improves the validity, the property and even has a positive effect on the similarity which leads to higher optimization success

(2) No EETN experiment, we evaluate the contribution of the EETN component. The architecture without EETN is composed of separately pre-trained METNs using the reconstruction and the KL loss. We then train the models together optimizing the reconstruction loss only. The inference path is $En_A \rightarrow De_B$. During training, each domain's decoder is specialized in generating molecules that hold the domain's property, thus the property is relatively high. However, nothing preserves the input-output chemical connection so their similarity and hence optimization success are low. This experiment shows that having two METNs, one for each domain, is extremely important, enabling excellent initial opening point for property enhancement. This principal of METNs specialization partially explains UGMMT's success comparing to other baselines.

(3) No fingerprints (fp) experiment, we evaluate the contribution of the fp component during the translation. We remove the usage of molecules's fingerprints so the translators inside the EETN receive only the embedding without the fp. We observe the fp-dependant translation reaches high similarity since it encourages the translators to embed chemically similar molecules next to each other preserving input-output similarity. Without the fp, the similarity is extremely low and as a result so is the optimization success. We notice the high property compared to the baseline model which shows that the model without the fp is highly focused on property optimization disregarding similarity to the original molecule. The high property and validity imply that the double cycle constraints indeed train the translators to produce better embeddings for the following components, e.g. the decoders. We hypothesize that the double cycle constraints during training and the fp usage are the key for UGMMT's success comparing to other baselines.

(4) Only fingerprints (fp) experiment, we wish to evaluate the necessity of the translators (EETN) and the embeddings that are jointly trained under the double cycle constraints. Given the results of the previous ablation test ("No fingerprints experiment"), one might claim that the only signal which is needed for molecule optimization is the source molecule fp. Hence, we designed a baseline which uses merely the molecule fp as the molecule embedding. We discard the training cycle that originates in A together with its encoder, decoder and the translators (EETN). For domain B, we remove the encoder and leave the decoder. The training is performed by training B's decoder to generate a molecule from its fp. Then, during inference, given a source molecule, its fp is calculated and passed to B's decoder to generate the output molecule. This design makes sense since B's decoder is trained to produce molecules from domain B with the same fp. Meaning, a similar molecule but with the enhanced property. Observing the results, indeed the similarity is higher compared to the baseline, however the property is lower so the overall optimization success is significantly lower. Especially, we notice the DRD2 property which is extremely low and unacceptable for molecule optimization process. Note that the validity is much lower compared to the baseline as well. As a result, we conclude that although molecule's fp is a key signal in our model, it is not enough. The METNs (encoders and decoders) and EETN trained under the double cycle constraints are essential.

(5) Swap Cycle-fp experiment, we change the architecture s.t. T_{AB} always gets fp_a and T_{BA} always gets fp_b (changes lines 8 and 12 in Algorithm 2). Although it might seem intuitive to insert the fp according to the translation direction of the translator, preserving the fp of the original molecule helps optimizing similarity to the input molecule. Swapping the fp, as we see in the results, has a destructive effect on the similarity and the optimization success. The property is relatively high as we already deduced from the "No fingerprints experiment" (ablation experiment number 3) that it is mostly connected to the none fp parts of the model, which we kept intact in this experiment.

(6) Add adversarial experiment, we wish to study the impact of adversarial training on our model, similar to Mol-CG. We train our model in an adversarial manner by treating the translators as generators and adding a discriminator for each domain. Each discriminator, e.g. B's discriminator D_B, gets an embedded molecule , $\ddot{Y}\ b\ \ i \in \ddot{Y}\ B\ \ i$, or an embedding of a translated molecule, $\ddot{Y}\ b'\ \ i \in \ddot{Y}\ B'\ \ i$, classifies it as "real" or "fake", respectively. Furthermore, KL loss terms are added to encourage dense and continuous embedding spaces for training stability and easier convergence. Please see **Appendix C** for more details. This model is an improved version of Mol-CG model. Similar to Mol-CG, this baseline also uses cycle constraints and adversarial training, but additionally we train METN for each domain and use molecule fp, both of which we found to have high importance. Furthermore, we apply the cycle constrains directly on the molecules and not the embeddings which is more effective and we also train the METNs together with the EETN which is more target optimized. Nevertheless, taking into account all these improvement over Mol-CG, we see that this ablation model has still worse performance compared to the baseline UGMMT model we propose. Particularly, the similarity and the

Table 3: Ablation experiment evaluation results of our method for P∈ {QED, DRD2} properties. In bold: best optimization Success result, statistically significant with huge effect size ($P - value$ Ÿ 0.05 and Cohen's d ¡ 2).

P	Method	Property	Similarity	Novelty	Success	Validity	Diversity
QED	UGMMT	0.855 ± 0.001	0.365 ± 0.003	0.997 ± 0.001	**0.513** ± 0.009	0.971	1.000 ± 0.000
	No Pre-train	0.843 ± 0.002	0.332 ± 0.002	0.999 ± 0.001	0.431 ± 0.009	0.879	1.000 ± 0.000
	No EETN	0.824 ± 0.002	0.126 ± 0.001	1.000 ± 0.000	0.001 ± 0.001	0.925	0.936 ± 0.009
	No fp	0.903 ± 0.001	0.127 ± 0.001	0.977 ± 0.005	0.002 ± 0.002	1.000	0.995 ± 0.002
	Only fp	0.808 ± 0.002	0.391 ± 0.003	0.989 ± 0.002	0.411 ± 0.009	0.782	1.000 ± 0.000
	Swap Cycle fp	0.872 ± 0.002	0.124 ± 0.001	1.000 ± 0.000	0.001 ± 0.001	0.962	0.996 ± 0.003
	Add Adversarial	0.857 ± 0.002	0.299 ± 0.002	0.998 ± 0.001	0.344 ± 0.008	0.980	1.000 ± 0.000
DRD2	UGMMT	0.826 ± 0.009	0.284 ± 0.001	0.799 ± 0.009	**0.192** ± 0.010	1.000	0.914 ± 0.007
	No Pre-train	0.505 ± 0.006	0.283 ± 0.001	0.988 ± 0.003	0.163 ± 0.005	0.582	0.992 ± 0.002
	No EETN	0.836 ± 0.008	0.158 ± 0.002	0.812 ± 0.013	0.011 ± 0.003	1.000	0.508 ± 0.006
	No fp	0.889 ± 0.006	0.159 ± 0.001	0.664 ± 0.013	0.010 ± 0.003	1.000	0.718 ± 0.012
	Only fp	0.302 ± 0.006	0.279 ± 0.001	0.994 ± 0.004	0.085 ± 0.007	0.560	0.996 ± 0.002
	Swap Cycle fp	0.809 ± 0.009	0.157 ± 0.001	0.883 ± 0.011	0.010 ± 0.003	0.997	0.408 ± 0.006
	Add Adversarial	0.796 ± 0.009	0.253 ± 0.002	0.889 ± 0.006	0.156 ± 0.005	0.998	0.939 ± 0.008

overall optimization success are lower. This explains the significant performance gap over Mol-CG in Section 5.1.

5.3 Optimized Drug Generation

Drug repositioning involves the exploration and improvement of existing drugs for new therapeutic purposes. We explore the use of our system for this purpose, where the input to the system is an existing drug and the goal is to produce a similar yet enhanced drug. CDN [8], which is a sequence-based method for generating molecules, was the only algorithm attempting a similar task in the past. However, their task did not include the requirement of an enhanced property. We conduct a retrospective experiment using 1,897 FDA approved drugs. We ensured none of them was observed in the training or validation data. As we focus on existing drugs the enhancement of drug likeliness is less relevant and therefore we focus on DRD2 enhancement. We apply our DRD2 trained model (Sec. 5.1) on every drug in the approved drugs set as prototype, and generate 100 drugs for each drug using 100 different random seeds.

Table 4: Sample of automatically generated drugs along with their prototype drugs, their DRD2 score and fp similarity.

Input		Generated		
Drug Name	DRD2	Drug Name	DRD2	Similarity
Pitavastatin	0.0077	Benperidol	1.0000	0.2651
Anagrelide	0.0009	Clozapine	0.8412	0.2096
Dapiprazole	0.1885	Aripiprazole	1.0000	0.2278
Alimemazine	0.3749	Perazine	0.6184	0.4000
Terazosin	0.0028	Iloperidone	1.0000	0.2556
Doxepin	0.1607	Amitriptyline	0.4743	0.6842
Promazine	0.5579	Trifluprom azine	0.9908	0.7073

We observe that although the chance of generating a drug using exhaustive search without constrains, e.g. HTS, is negligible, our method generates approximately **one approved drug for every ~3260 valid molecules generated**. CDN for comparison reaches one drug per 14,596 generated molecules. Furthermore, the chance of discovering a drug-drug pair when going over all the drug-valid molecule pairs is 0.05971% which is relatively significant compared to CDN, 0.01185%. We observe that the **average generated drugs' DRD2 score is 0.879**. For CDN, the average score is 0.124. We evaluate the average DRD2 score improvement on all the unique drug-drug pairs reaching an **average property improvement of 0.627**. The generated drugs show high similarity to their prototypes with an average of **Tanimoto similarity of 0.3875**.

We investigate the generated drugs and observe that many of the generated drugs are indeed used to treat psychotic disorders, like Schizophrenia. We remind the reader that D2 is the main receptor for most antipsychotic drugs [27]. Table 4 presents a sample of the generated drugs.

We demonstrate our model's capability of generating FDA-approved drugs. Today, additional laboratory experiments are carried on novel molecules. The system is currently deployed for use in a personalized medicine and nanotechnology research laboratory, which is currently focusing on RNA based therapeutics [25]. We leverage RNA molecules to turn on and off pathogenic genes. Delivering these molecules to their destination is challenging and requires specific carrier molecules. Since a certain family of molecules already has some of the properties required for this task, our method is used to optimize them and generate novel carrier molecules. The generated molecules are currently being evaluated by the chemists. The most promising molecules would be synthesized and pass to advanced chemical trials.

6 CONCLUSIONS

We have recently witnessed the urgent need for accelerating the drug discovery process, e.g. COVID-19 drug. Lead optimization is one of the most important steps in the drug discovery chain. First, initial set of hits is identified through the HTS or expert-driven process. Next, in the hit-to-lead step, the most promising hits are selected to advance into the lead optimization stage, where the leads are chemically modified in order to improve biological activity and

other critical properties. However, this is a very lengthy and expensive process. The first computational models for this challenge addressed the molecule generation process as a sequence-to-sequence generation problem, representing molecules as sequences of characters, most commonly, SMILES notation was used [5, 7, 8, 15]. These models mainly focused on valid molecules generation, thus creating molecules unrelated to a target. To overcome this barrier, computational models were introduced for molecular lead optimization, which describes generating molecules similar to a drug candidate but with enhanced properties. The SMILES notation was abandoned, molecules were seen as graphs and the lead optimization was reformulated as a graph-to-graph translation problem [6, 10, 11, 18]. This approach is currently the SOTA, demonstrating impressive optimization results. In our work, we use the simple, yet powerful, SMILES notation and design an end-to-end lead optimization model, combines molecule embedding, fp and unique training technique, which leverages SMILES notation to generate high quality novel molecules and even FDA-approved drugs. Our model outperforms the SOTA, generating more successful molecules with higher desired property score. Our work may revolutionize the field of computational molecule generation, bringing the focus back to the sequence generation methods.

Unlike prior works that required paired supervised data for training, we propose an unsupervised deep generative method for molecule translation, i.e., bidirectional optimization with no paired data required. The model is unique in its ability to simultaneously translate between discrete molecule representation and a continuous representation coupled with a double cycle constrained training technique with shared translation components that leverage molecule fp to gain both resemblance to the original molecular lead and generation of novel molecules with enhanced properties. The algorithm shows significant performance gains in the success of generating novel optimized molecules that share similarity with a prototype. We conduct comprehensive ablation experiments supporting UGMMT's analysis and architecture design. Finally, we demonstrated UGMMT's ability of generating target-based approved drugs it has never encountered before. The system is currently being deployed for use in pharmaceutical laboratories to further analyze the additional generated molecules. We believe our method lays strong foundations to an automatic-algorithmic HTS process to enable lead optimizations without the need for large paired training datasets.

REFERENCES

[1] Dávid Bajusz, Anita Rácz, and Károly Héberger. 2015. Why is Tanimoto index an appropriate choice for fingerprint-based similarity calculations? *Journal of Cheminformatics* 7 (05 2015).

[2] Richard Bickerton, Gaia Paolini, Jérémy Besnard, Sorel Muresan, and Andrew Hopkins. 2012. Quantifying the chemical beauty of drugs. *Nature chemistry* 4 (02 2012), 90–8.

[3] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Doha, Qatar, 1724–1734.

[4] J. Cohen. 1988. *Statistical Power Analysis for the Behavioral Sciences*. Lawrence Erlbaum Associates.

[5] Hanjun Dai, Yingtao Tian, Bo Dai, Steven Skiena, and Le Song. 2018. Syntax-Directed Variational Autoencoder for Structured Data. In *International Conference on Learning Representations*.

[6] Tianfan Fu, Cao Xiao, and Jimeng Sun. 2020. CORE: Automatic Molecule Optimization Using Copy & Refine Strategy. In *34th AAAI Conference on Artificial Intelligence (AAAI-20)*.

[7] Rafael Gómez-Bombarelli, Jennifer N. Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D. Hirzel, Ryan P. Adams, and Alán Aspuru-Guzik. 2018. Automatic Chemical Design Using a Data-Driven Continuous Representation of Molecules. *ACS Central Science* 4, 2 (Jan 2018), 268–276.

[8] Shahar Harel and Kira Radinsky. 2018. Accelerating Prototype-Based Drug Discovery Using Conditional Diversity Networks. In *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining* (London, United Kingdom) *(KDD '18)*. 331–339.

[9] Di He, Yingce Xia, Tao Qin, Liwei Wang, Nenghai Yu, Tie-Yan Liu, and Wei-Ying Ma. 2016. Dual Learning for Machine Translation. In *Proceedings of the 30th International Conference on Neural Information Processing Systems* (Barcelona, Spain) *(NIPS'16)*.

[10] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. 2018. Junction Tree Variational Autoencoder for Molecular Graph Generation. In *International Conference on Machine Learning*. 2323–2332.

[11] Wengong Jin, Kevin Yang, Regina Barzilay, and Tommi Jaakkola. 2019. Learning Multimodal Graph-to-Graph Translation for Molecule Optimization. In *International Conference on Learning Representations*.

[12] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. 2016. Perceptual losses for real-time style transfer and super-resolution. In *European Conference on Computer Vision*.

[13] Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*.

[14] Diederik P. Kingma and Max Welling. 2014. Auto-Encoding Variational Bayes.. In *ICLR*, Yoshua Bengio and Yann LeCun (Eds.).

[15] Matt J. Kusner, Brooks Paige, and José Miguel Hernández-Lobato. 2017. Grammar Variational Autoencoder. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70* (Sydney, NSW, Australia) *(ICML'17)*. 1945–1954.

[16] G Landrum. 2016. Rdkit: Open-source cheminformatics software.

[17] Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*.

[18] Łukasz Maziarka, Agnieszka Pocha, Jan Kaczmarczyk, Krzysztof Rataj, Tomasz Danel, and Michał Warchoł. 2020. Mol-CycleGAN: a generative model for molecular optimization. *Journal of Cheminformatics* 12 (12 2020).

[19] World Health Organization. 2019. *World Health Organization model list of essential medicines: 21st list 2019.* Technical documents. 60 p. pages.

[20] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*. 8026–8037.

[21] Pavel G Polishchuk, Timur I Madzhidov, and Alexandre Varnek. 2013. Estimation of the size of drug-like chemical space based on GDB-17 data. *Journal of computer-aided molecular design* 27, 8 (2013), 675–679.

[22] Alec Radford, Luke Metz, and Soumith Chintala. 2016. Unsupervised Representation Learning with Deep Convolutional Generative Adversarial Networks. In *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

[23] D. Rogers and Mathew Hahn. 2010. Extended-Connectivity Fingerprints. *Journal of chemical information and modeling* 50 5 (2010), 742–54.

[24] Shlomo Sawilowsky. 2009. New Effect Size Rules of Thumb. *Journal of Modern Applied Statistical Methods* 8 (11 2009), 597–599.

[25] Ryan Setten, John Rossi, and Si-ping Han. 2019. The current state and future directions of RNAi-based therapeutics. *Nature Reviews Drug Discovery* 18 (03 2019), 1.

[26] Gail Sullivan and Richard Feinn. 2012. Using Effect Size—or Why the P Value Is Not Enough. *Journal of graduate medical education* 4 (09 2012), 279–82.

[27] Sheng Wang, Tao Che, Anat Levit, Brian Shoichet, Daniel Wacker, and Bryan Roth. 2018. Structure of the D2 dopamine receptor bound to the atypical antipsychotic drug risperidone. *Nature* 555 (03 2018), 269–273.

[28] David Weininger. 1988. SMILES, a chemical language and information system. 1. Introduction to methodology and encoding rules. *Journal of chemical information and computer sciences* 28, 1 (1988), 31–36.

[29] R. J. Williams and D. Zipser. 1989. A Learning Algorithm for Continually Running Fully Recurrent Neural Networks. *Neural Computation* 1, 2 (1989), 270–280.

[30] David S Wishart, Craig Knox, An Chi Guo, Savita Shrivastava, Murtaza Hassanali, Paul Stothard, Zhan Chang, and Jennifer Woolsey. 2006. DrugBank: a comprehensive resource for in silico drug discovery and exploration. *Nucleic acids research* 34, suppl_1 (2006), D668–D672.

[31] J. Zhu, T. Park, P. Isola, and A. A. Efros. 2017. Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*. 2242–2251.

A FULL IMPLEMENTATION DETAILS

We run all training and experiments on Ubuntu 18.04.5 using one Nvidia GeForce RTX 2080 Ti 11GB GPU, two Intel Xeon Gold 6230 2.10GHZ CPUs and 64GB RAM memory. We use Pytorch 1.4.0 API [20], Python 3.6.12 for UGMMT model training and experiments. Other baselines are trained using their publicly available implementation with the setting they set and reported in their papers. Our code and data are publicly available on our GitHub. See file `README.md` for installation, training, ablation experiments and inference instructions. Our model depends on randomness, we set seed 50 for reproducibility, other seeds lead to close results as well. However, small variations in the results might occur following different training sessions. The source of variation is in the EETN networks, probably due to PyTorch functions that use CUDA functions that can be a source of non-determinism (e.g. atomic operations), it is a known problem in the community and we are trying to handle it. Some hyper-parameters are set following evaluation on the validation set, the metric is the optimization Success. Values we experimented with are specified in the brackets beside each hyper-parameter.

METN: Latent space dimension (embedding dimension) 256 (experimented with 128, 256, 512), λ_0=10 (experimented with 1, 10, 20), max molecule length during test 90 (experimented with 80-120). Encoder – bidirectional GRU with 1 layer, hidden dimension 256 (En, h_d), followed by 2 fully-connected layers with 256 neurons, one for μ and one for σ. Decoder – fully-connected layer with 512 neurons, then unidirectional GRU with 3 layers (experimented with 1-5), dropout 0.5 between layers, hidden dimension 512 (De, h_d) and a fully-connected layer with vocabulary length number of neurons.

METN's training: Optimizer – Adam algorithm [13]. Learning rate – Cosine annealing scheduler with restart [17] after 10 epochs, initial learning rate $3 \cdot 10^{-3}$ (experimented with $1 \cdot 10^{-3}$-$5 \cdot 10^{-3}$), final learning rate $3 \cdot 10^{-4}$. Weights initialization – GRU's weights uniformly distributed ranging from $-h_d^{-1}$ to h_d^{-1}, linear layer's weights uniformly distributed ranging from $-f_{in}^{-1}$ to f_{in}^{-1}, where f_{in} is the number of input features. Mini-batch size 32 and trained for E_{METN} = 1 (QED) (experimented with 0-10) or E_{METN} = 12 (DRD2) (experimented with 0-20) epochs before training the whole model together. Trained using the teacher-forcing method [29] in order to increase validity.

EETN: Translator – Bottleneck structure, i.e. downsampling followed by upsampling. Contains initial convolution block (filter size 7), stride-2 convolution block for downsampling (filter size 3), 4 residual blocks (experimented with 4-8), stride-2 transposed convolution block for upsampling (filter size 3), final convolution layer (filter size 7) followed by Tanh layer and 2 fully connected layers with 1,152 and 256 neurons separated by batch normalization, LeakyReLU and Dropout (0.2) layers (all convolution blocks contain instance normalization and ReLU layers). This type of structures showed outstanding results in image translation and style transfer tasks [12, 31].

We pass molecule's fingerprints through a basic attention mechanism consists of a fully-connected layer followed by a softmax layer to generate a weights vector, which multiplies the fingerprints vector elementwise, highlighting the "important" information inside the fingerprints vector. Embedded molecule and molecule's weighted fingerprints vector are concatenated before applying the Translator.

Whole model training: Optimizer – Adam algorithm. Learning rate - LambdaLR scheduler with initial learning rate $3 \cdot 10^{-4}$ (experimented with $1 \cdot 10^{-4}$-$5 \cdot 10^{-4}$) and linear decay towards 0 from epoch 90 (experimented with 70-100). Weights initialization – from zero-centered normal distribution with a standard deviation of 0.02 [22]. Mini-batch size 32. Maximal epochs for training E_{max}=120 (experimented with 80-120). Training data is shuffled during training, in addition, to further reduce overfitting and improve generalization, we evaluate our model every V_f=3 epochs on the validation set and save a checkpoint only if improvement in the criterion is achieved. Furthermore, we add early stopping mechanism, i.e. if this criterion does not improve for P=15 evaluations, we stop the training. Regularization hyper-parameter λ_1=2 (experimented with 1,2,4,8,16).

B FULL DATASETS DETAILS

We provide experiments demonstrating our model's capability of molecule and drug optimization. Hence we use two different datasets.

(1) **Molecule Dataset**: The current SOTA method in molecule optimization is CORE [6]. Therefore we use their datasets, which were adapted from [11] and are publicly available on their GitHub[2].

Train set: These datasets are paired, designed for supervised models, however UGMMT, CDN, JTVAE and Mol-CG are unsupervised. Hence we use the paired train set to construct unpaired train set. UGMMT method requires 2 input sets, set for domain A which contains low property molecules and set for domain B which contains high property molecules. We construct these by unpairing the pairs, removing duplicates and then for each molecule we calculate its property score {DRD2, QED} and add it to the relevant set if the score exceeds a certain threshold. i.e., the molecule is added to set A if its property score is lower than domain's A threshold and it is added to set B if its property score is higher than domain's B threshold. In order to avoid unbalanced domains issues we randomly sample equal number of molecules from each set and thus obtain the final train set for A and B. Since JTVAE and CDN require only one training set, we merge A's and B's train sets to one set and use it to train them. We set empirically the following domain thresholds: DRD2 – A 0.02 (experimented with 0.01-0.2) ; B 0.85 (experimented with 0.75-0.9) / QED – A 0.78 (experimented with 0.75-0.8) ; B 0.91 (experimented with 0.88-0.93), other thresholds yield similar results. All dataset files are located inside `dataset/DRD2` and `dataset/QED` folders on our GitHub. Details:
UGMMT and Mol-CG– The train sets contain 2,097 molecules for DRD2 and 8,968 molecules for QED (in each set among $\{A, B\}$). Files names: `A_train.txt` and `B_train.txt`.
CDN and JTVAE– The train sets contain 4,194 molecules for DRD2 and 17,936 molecules for QED.
Files names: `DRD2_mergedAB_specific_train.txt` and `QED_mergedAB_specific_train.txt`.

[2] https://github.com/wengong-jin/iclr19-graph2graph

G2G and CORE– Taken from their GitHub, contains 34,404 molecule pairs for DRD2 and 88,306 molecule pairs for QED. Files names: `DRD2_DATASET.txt` and `QED_DATASET.txt`.
Validation set: For UGMMT, after sampling the final train set from set A, we randomly sample the validation set from set A remaining molecules. For the other models we use the original validation sets from Fu et al. [6].
UGMMT– The validation sets contain 800 molecules for DRD2 and 800 molecules for QED.
File name: `A_validation.txt`.
Other models– The validation sets contain 500 molecules for DRD2 and 360 molecules for QED.
File name: `g2g_validation.txt`.
Test set: The test set is taken as is from Fu et al. [6], thus all methods are evaluated on exactly the same data. The test sets contain 1,000 molecules for DRD2 and 800 molecules for QED. File name: `A_test.txt`.

(2) **Drug Dataset**: We use DrugBank dataset [30], which contains a list of FDA approved drugs, to conduct retrospective experiments and confirm our model's capability of potential drug optimization. We extract a set of 1,897 drugs and ensure none of them appeared during training or validation. The drug dataset is located in the `FDA_approved_canon_clean.csv` file inside the `dataset/FDA_approved_drugs_drugbank` folder on our GitHub.

C ADD ADVERSARIAL ABLATION EXPERIMENT DETAILS

We wish to study the impact of adversarial training on our model, similar to Mol-CG. In addition to our cycle constraints with fp attention we add adversarial training. In this experiment, similarly to our baseline UGMMT model, we pre-train the METNs before training the whole model.

We treat the translators as embedding generators and add two embedding discriminators, D_A and D_B, one for each domain. Each discriminator, e.g. D_B, gets an embedded molecule and decides whether it is an original embedded molecule , $\ddot{Y}\ b\ _i \in \ddot{Y}\ B\ _i$, or an embedding of a translated molecule, $\ddot{Y}\ b'\ _i \in \ddot{Y}\ B'\ _i$, classifies it as "real" or "fake", respectively. Hence, generator's goal is to create embeddings indistinguishable from the real data distribution while discriminator's goal is to distinguish candidates artificially created from the real data distribution. This process is expressed in the additional adversarial loss terms for translators training:

$$MSE(D_B(T_{AB}(En_A(a), fp_a)), 1_v) +$$
$$MSE(D_A(T_{BA}(En_B(b), fp_b)), 1_v)$$

Where MSE is the mean square error loss, 1_v is a vector of ones. We also add KL loss terms to encourage dense and continuous embedding spaces for training stability:

$$0.2 \cdot (KL(En_A(a)||N(0, I)) + KL(En_B(b)||N(0, I)))$$

Where 0.2 is a small scalar. Its purpose is, on the one hand, to make these terms significant enough for improving training stability, and on the other hand, to be negligible enough compared to the other terms to not interrupt the adversarial training.
After training the translators, in the same iteration, discriminator A is trained by minimizing:

$$0.5 \cdot MSE(D_A(En_A(a)), 1_v) +$$
$$0.5 \cdot MSE(D_A(T_{BA}(En_B(b), fp_b)), 0_v)$$

Similarly, discriminator B is trained by minimizing:

$$0.5 \cdot MSE(D_B(En_B(b)), 1_v) +$$
$$0.5 \cdot MSE(D_B(T_{AB}(En_A(a), fp_a)), 0_v)$$

Where MSE is the mean square error loss, 1_v is a vector of ones and 0_v is a vector of zeros. The first term in each objective function encourages the discriminator to identify "real" molecules and the second – "fake" molecules.
Each Discriminator contains 3 convolution blocks, each one consists of convolution (filter sizes 3, 3, 4), instance normalization and LeakyReLU (0.2) layers. Then convolution (filter size 4) and average pooling. First two convolutions are stride-2.

One might consider this model as an improved version of Mol-CG model as in addition to adversarial training introduced by the latter, we also train METN for each domain and use molecule fp. Furthermore, we apply the cycle constraints directly on the molecules and not the embeddings which is more effective and we also train the METNs together with the EETN which is more target optimized. However, taking into account all these improvements over Mol-CG, we see that this ablation model has still worse performance compared to the baseline UGMMT model we propose. This explains the significant performance gap of UGMMT over Mol-CG in Section 5.1 in the paper.