Multi-Property Molecular Optimization using an Integrated Poly-Cycle Architecture

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

Galia Nordon
Technion - Israel Institute of Technology
Haifa, Israel
galiasn@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 molecules similar to a drug candidate but with enhanced properties. Most prior works focused on optimizing a single property. However, in real settings, we wish to find molecules that satisfy multiple constraints, e.g. , potency and safety. Simultaneously optimizing these constraints was shown to be difficult, mostly due to the lack of training examples satisfying all constraints. In this work, we present a novel approach for multi-property optimization. Unlike prior approaches, that require a large training set of pairs of a lead molecule and an enhanced molecule, our approach is unpaired. Our architecture learns a transformation for each property optimization separately, while constraining the latent embedding space between all transformations. This allows generating a molecule which optimizes multiple properties simultaneously. We present a novel adaptive loss which balances the separate transformations and stabilizes the optimization process. We evaluate our method on optimizing for two properties: dopamine receptor (DRD2) and drug likeness (QED), and show our method outperforms previous state-of-the-art, especially when training examples satisfying all constraints are sparse.

CCS CONCEPTS

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

KEYWORDS

AI Multi-property Lead Optimization, Drug Discovery, ML for Healthcare

ACM Reference Format:
Guy Barshatski, Galia Nordon, and Kira Radinsky. 2021. Multi-Property Molecular Optimization using an Integrated Poly-Cycle Architecture. In *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM '21), November 1–5, 2021, Virtual Event, QLD, Australia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3459637.3481938

CIKM '21, November 1–5, 2021, Virtual Event, QLD, Australia

ACM ISBN 978-1-4503-8446-9/21/11…$15.00
https://doi.org/10.1145/3459637.3481938

1 INTRODUCTION

Developing a new drug is a very costly process, taking up to 15 years and more than 2 billion dollars in cost. Lead optimization [1, 5, 8, 11, 18] is a task in which a substance is identified as having desired properties and its chemical properties are improved to create an optimal substance which will become a potential drug candidate.

While early unsupervised machine-learning approaches to this problem showed sub-par results [8], more recent work show more promise. The current state-of-the-art (SOTA) [1] leverages a type of dual learning for single-property molecular optimization. Other leading approaches [5, 11] are supervised, involve the acquisition of a set of paired molecules – the original molecule and an enhanced molecule with a more desirable property and training a supervised generative model. Although prior work mostly focused on optimizing a single property while keeping similarity to the original lead molecule, in real applications, there is often a need to generate molecules that satisfy multiple constraints, e.g., potency and safety. This task was shown to be challenging [9], as those approaches not only require large training sets, but also significant number of training instances of molecules conforming to all the constraints simultaneously. The latter is highly complicated to obtain in nature for most molecular properties. For example, the authors of [9] report that only 1.6% of the training pairs are both drug-like and DRD2-active – the main properties molecular optimizations focus on due to their significance to the drug discovery industry [1, 9, 11].

In this work, we present a novel Integrated Poly-Cycle Architecture (IPCA) for multi-property optimization. The architecture learns parallel transformations for each optimization property while constraining the transformations to maintain a single multi-property optimized molecule rather than an optimized molecule for each property. Our approach does not require any paired data and is less sensitive to lack of training instances that satisfy all target properties. Our model belongs to the family of models that optimize molecules given their SMILES representation [25] – a common molecular string representation. For each target property an encoder is trained to translate a discrete molecule to a continuous representation and a translator to translate this embedding to an embedded source domain. Then, we concatenate these components to a central translator, shared by all properties, forcing a common optimized embedding space, and a central decoder to translate it back to the target domain. Thus a cycle is formed for each property. We use these multiple cycles to train our model. At test time, given a discrete source molecule, we first embed it using the encoder, then use the shared translator to map it to a continuous representation of a molecule which has the enhanced properties. Finally, we use the shared decoder to generate an optimized molecule. Molecule

fingerprint is supplied to the translator in order to encourage a molecular similarity preservation to the source molecule. As different properties induce different optimization paths, with different training complexity, we present a novel adaptive loss, which adjusts components' relative importance by adapting their coefficients during learning.

The contributions of this work are threefold:
(1) We design a novel unpaired end-to-end generative model and a unique poly-cycle training scheme for molecule translation preserving molecule similarity while optimizing multiple molecular properties. To the best of our knowledge, our work is the first in the family of models considering the SMILES molecular representation to tackle the multi-property optimization problem.
(2) We present an adaptive loss function that allows to balance the multi-property optimization during the training, thus yielding superior results.
(3) We present empirical results on numerous datasets and optimize multiple levels of DRD2 (dopamine receptor) and QED (drug likeness) properties. We demonstrate superior performance over SOTA baselines in generating molecules with higher property scores and success rates, especially notable when few examples confirming to multiple properties exist in the data. We perform ablation tests studying the behavior of IPCA in different settings. All code and data in this work are published on our GitHub[1] for further research.

The system is currently being deployed for use in a personalized medicine and nanotechnology research laboratory, which is focusing on RNA based therapeutics [23]. The goal is to use our method to optimize candidate molecules in order to generate novel RNA carrier molecules. We believe our method lays the foundations to an automatic-algorithmic drug discovery process to enable multi-property lead optimizations.

2 RELATED WORK

Molecule Generation: Numerous previous works focused on generating molecules. The models can be characterized by the molecular representation they generate. Some of these formulated molecule generation as a sequence generation problem, representing a molecule as a sequence of characters. Most commonly, SMILES notation was used and showed merit for this task [4, 6, 7, 15, 19, 21], others used molecular graphs [10, 24].

The generation process was formulated either by leveraging variational autoencoders [4, 6, 7, 15, 24] or reinforcement learning [10, 19, 21]. Most of the aforementioned models mainly attempt to generate valid molecules without the constraint of similarity to an initial molecular lead. These models overcome the initial task of generating valid SMILES sequences or molecule graphs, but without specifically addressing the optimization task and similarity to the lead, their performance is poor [5].

Molecule Single-Property Optimization: Is a separate task from molecule generation as it requires to not only generate an enhanced molecule but also to keep similarity to the original molecule lead. JTVAE [8] presents an encoder-decoder architecture for graph-to-graph translation and molecule optimization. An input molecular graph is encoded into a tree structure, that is embedded, decoded and transformed back into a molecular graph. Several later works extended this approach with attention mechanism [11], adversarial learning [11, 18] or copy&refine decision mechanism [5]. UGMMT [1] uses SMILES notation to represent molecules and presents an encoder-decoder combined with a translator architecture, leveraging dual learning based technique to perform the optimization. The aforementioned methods do not address the task of optimizing multiple properties simultaneously.

Molecule Multi-Property Optimization: Jin el al. [9] was the first to address the problem of multi-property optimization. Their work extends the graph-to-graph translation model incorporating conditional translation. The translation condition is a vector describing the required property values after translation (e.g., 11 for high QED and high DRD2, 01 for low QED and high DRD2, etc.) However, these models require large training sets and fail to perform well when the datasets contain small number of molecules that comply to all constrains. We present a novel architecture in which the property optimization translators are learned in parallel while being constrained by two factors: (1) a cyclic constraint on each translator maintaining similarity between the source molecule and its optimization, (2) the latent space from encoder to decoder is shared between translators thus creating one molecule that is optimized in multiple aspects. We show our model reaches SOTA results and is not as sensitive to the existence of molecules that comply to all constrains in the dataset. To the best of our knowledge, our work is the first to tackle the multi-property molecular optimization in the family of models considering the SMILES representation.

3 POLY-CYCLE ARCHITECTURE

We denote a domain of molecules, e.g. high drug likeness, 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 $< x >$. It belongs to the domain of all the embedded molecules of X, which we denote by $< X >$ with a distribution $p(< X >)$. We also denote a molecule property by $prop_i$. For example, if $prop_1$ is QED, $prop_1(x)$ is the QED value of molecule x. We wish to transform a molecule m with degraded properties to a molecule m' with optimized properties. Suppose A is a domain of molecules with degraded N properties, $prop_i$, $i \in [1 \dots N]$. We wish to transform $m \in A$ into $m' \in B_i$ in which $prop_i(m') > prop_i(m)\ \forall i \in [1 \dots N]$. In addition, we wish that m' remains similar to m. Figure 1 presents the poly-cyclic architecture for $N = 2$. Our model contains an optimization path for each property $prop_i$. The paths are then joined by a shared embedding space from which a joined optimization can be induced (solid red line). Thus, in the case of double property optimization, one optimization path will transform molecules from A to B_1, the second optimization path will transform molecules from A to B_2 and the transformation to domain $B_{1,2}$, in which both properties are optimal, will be achieved through a shared embedding space.

3.1 Molecular Optimization Path

In this section, we describe an end-to-end optimization path for a single property. For simplicity, we use the $prop_1$ notation for the

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

property (the upper half of Figure 1). The dotted lines in the figure represent *training paths* and the solid lines represent *inference paths*.

During inference, following the translation along the path from A to B_1 (solid red line), an input molecule $a \in A$, given in a discrete textual SMILES representation, is encoded by an encoder En_A to its continuous representation $< a >$ (Section 3.4), then using a translator $T_{AB_{1,2}}$ it is mapped to a continuous space of molecules with the enhanced property $prop_1$. Lastly, it is decoded to its SMILES representation using the decoder $De_{B_{1,2}}$. En and De are respectively an encoder and decoder GRU networks followed by fully connected layers, and T are convolutional bottleneck networks with resnet layers (see Section 4.5 for details).

The training for this path is described in the dotted lines. The upper red dotted line forms a counter-clockwise circle from A through En_A, $T_{AB_{1,2}}$, T_{B_1A} and De_A back to A. En_A encodes $a \in A$ to a latent embedded space $< A >$, $T_{AB_{1,2}}$ and T_{B_1A} transform the encoding sequentially to $< B'_{1,2} >$ and $< A' >$, and De_A decodes it back to A'. The distribution of A' should be indistinguishable from the distribution of A. We encourage this by penalizing a and a' difference in the loss function. The upper purple dotted line describes an identical mirrored cycle starting (and ending) in domain B_1.

Similarly to Barshatski and Radinsky [1], to encourage similarity between the source and the optimized molecules, we concatenate the molecule's extended connectivity fingerprints (fp_a and fp_{b_1}) [22] to its latent representation. This forces the embedding spaces to become fingerprint dependant, representing molecules with similar molecular structures with similar embeddings.

3.2 Multi-Property Optimization

For multi-property optimization, we combine several optimization paths. The paths are linked by a shared embedding space (in our example $< B'_{1,2} >$) which is now constrained by both B_1 and B_2. Applying the decoder $De_{B_{1,2}}$ on this latent space will produce molecules that are optimized both for $prop_1$ and $prop_2$. Adding another property optimization ($prop_3$) will entail adding an encoder (En_{B_3}), translator (T_{B_3A}) and linking them to the main translator (would be $T_{AB_{1,2,3}}$) that generates the shared embedding space. See Figure 2. Some details are not shown for readability (e.g. fp_{b_2}), however it contains all the components that were presented in Figure 1. That will contribute two additional cycles to the training phase: $En_A \rightarrow T_{AB_{1,2,3}} \rightarrow T_{B_3A} \rightarrow De_A$ (the additional red dotted path) and $En_{B_3} \rightarrow T_{B_3A} \rightarrow T_{AB_{1,2,3}} \rightarrow De_{B_{1,2,3}}$ (the additional purple dotted path). In the same manner, additional properties for optimization may be simply added.

Training: Algorithm 1 describes the end-to-end training of the model. We first pre-train all encoders and decoders to generate valid continuous molecule representation for A, B_1 and B_2 molecule domains. This allows quality inputs to the translators, which, in turn, produce results with higher validity. We then simultaneously train four cycles: $En_A \rightarrow T_{AB_{1,2}} \rightarrow T_{B_1A} \rightarrow De_A$ (lines 8-11), $En_A \rightarrow T_{AB_{1,2}} \rightarrow T_{B_2A} \rightarrow De_A$ (lines 12-14), $En_{B_1} \rightarrow T_{B_1A} \rightarrow T_{AB_{1,2}} \rightarrow De_{B_{1,2}}$ (lines 15-18) and $En_{B_2} \rightarrow T_{B_2A} \rightarrow T_{AB_{1,2}} \rightarrow De_{B_{1,2}}$ (lines 19-22). The translators together with the unique training technique encourage similar distribution in domains $< A >$ and $< A' >$, $< B_1 >$ and $< B'_{1,2} >$, $< B_2 >$ and $< B'_{1,2} >$. T_i is a translation neural network from one latent embedding domain to another. For example: $T_{AB_{1,2}}$ maps $< a > \in < A >$ to $< b'_{1,2} > \in < B'_{1,2} >$. During training, $T_{AB_{1,2}}$ is trained to transform $< a > \in < A >$ to $< b'_{1,2} > \in < B'_{1,2} >$ through two training cycles (upper left and lower left in Figure 1). During inference, $< b'_{1,2} > \in < B'_{1,2} >$ is then decoded into $b'_{1,2} \in B'_{1,2}$ which is a SMILES representation of the optimized molecule.

Translation between embedding domains is further conditioned on the molecules fingerprints fp_a, fp_{b_1} and fp_{b_2} (lines 8, 9, 12, 15, 16, 19, 20) to maintain similarity between the source and the optimized molecules. Since SMILES notation is a discrete representation, we use multi-layer GRU cells in the decoder to predict the next character in the SMILES representation given the current state and the current input character. Hence, the correct loss for this classification task is the Cross-Entropy (CE), where $CE(a', a)$ means the mean CE loss between the original a molecule (SMILES characters) and the reconstructed a' molecule.

The overall loss is composed of the four cycles' cross-entropy losses (CE). Note that each cycle may be easily formalized as an objective function to minimize, e.g., for the upper left cycle that starts in domain A:

$$CE(De_A(T_{B_1A}(T_{AB_{1,2}}(En_A(a), fp_a), fp_a)), a) \tag{1}$$

To ease the readability and reproduciblity of the paper, we present an algorithmic formulation of the above (Algorithm 1). We suggest a novel loss function that automatically weighs the cross entropy losses of the cycles: $En_{B_1} \rightarrow T_{B_1A} \rightarrow T_{AB_{1,2}} \rightarrow De_{B_{1,2}}$ and $En_{B_2} \rightarrow T_{B_2A} \rightarrow T_{AB_{1,2}} \rightarrow De_{B_{1,2}}$. The loss component coefficients are adapted during the training as described in Section 3.3.

Inference: During inference a lead molecule $m \in A$ is encoded by En_A to the embedding domain $< A >$, mapped by $T_{AB_{1,2}}$ to the embedding space $< B'_{1,2} >$ and decoded by $De_{B_{1,2}}$. Since $T_{AB_{1,2}}$ is shared by all cycles during the training, the embedding space $< B'_{1,2} >$ represents molecules with optimized values for both $prop_1$ and $prop_2$ and so is the consequent SMILES representation created by the decoder $De_{B_{1,2}}$.

3.3 Adaptive Loss

Since our model's goal is to generate molecules which have multiple enhanced properties, there might be properties that are more challenging to optimize than others. In this case, the optimization may be leaning towards the "easy to optimize" ones. Although setting constant coefficients in the loss might help, we notice that adjusting these coefficients during training yields smoother, more stable training process and consequently improved performance (see Section 5.3.2). We achieve this by dynamically adjusting the loss component coefficients during training (λ_{B_1} and λ_{B_2} in Algorithm 1).

We initialize $\lambda_{B_1} = 1$ and $\lambda_{B_2} = 1$ and following every validation step we update

$$\lambda_{B_1} = \lambda_1 * \frac{wanted_1}{validation_1} \tag{2}$$

and

$$\lambda_{B_2} = \lambda_2 * \frac{wanted_2}{validation_2} \tag{3}$$

where λ_1 and λ_2 are initial constant values, $wanted_i$ is the desired $prop_i$ property value and $validation_i$ is the average $prop_i$ property value of the molecules which are generated in the validation process.

3.4 METN for Molecule-Embedding Translation

One of the main challenges of models leveraging SMILES representations in deep generative settings is their discrete representations. We describe an architectural component that allows their transformation to a continuous representation, on which optimization can be performed. We follow the design of UGMMT [1] and use the METN component which enables a sequence-to-sequence translation with a bidirectional GRU-based encoder and a multilayered GRU-based decoder [3, 14]. Following Barshatski and Radinsky [1], in order to boost the validity of the molecules that are generated by the decoder, we leverage the teacher-forcing method [26] during training: we provide each GRU cell the correct input character even if the previous GRU cell predicted it incorrectly. We draw the reader attention that the encoder's architecture can be replaced with stronger architectures (e.g. transformers). In this work, we present simple GRU architecture to emphasize that the key for IPCA's success lies in the cycle constraints with molecular fingerprints and the adaptive loss components.

4 EXPERIMENTAL SETUP

4.1 Datasets

A common multi-property optimization task is to optimize drug likeness (QED) [2] and Dopamine Receptor D2 (DRD2). We used the datasets by [11] as reported by [9]. QED and DRD2 values for optimized molecules are calculated using the RDKit package [16] and the trained model for DRD2 activity from [19]. The original training set contains 122,710 molecule pairs and the test set contains 780 molecules. Different application may require different thresholds of DRD2 and QED. In this work, we experiment with several such thresholds. We chose to test the above two attributes following the latest related work (Section 2, e.g. [9]). Note that QED, i.e. drug likeness, is a combination of molecular properties such as solubility, ligand efficiency, molecular weight, etc.. Hence, by optimizing QED we effectively optimize multiple molecular properties.

4.2 Metrics

We evaluate our task using several metrics:

(1) Validity – the percentage of source molecules that have at least one valid optimized molecule.
(2) Average Property Value – the average QED, DRD2 and similarity values of the optimized molecules.
(3) Novelty – the percentage of the valid optimized molecules which were not seen in the training set.
(4) Diversity – the percentage of unique generated molecules, i.e., the number of unique molecules divided by the number of valid molecules.
(5) Success rate (SR) – the percentage of optimized molecule which have similarity larger than 0.3, and QED and DRD2 values above their respective thresholds.

Algorithm 1 IPCA Training algorithm. Corresponds to the four dotted paths in Figure 1, the red paths along $En_A \rightarrow T_{AB_{1,2}} \rightarrow T_{B_1A} \rightarrow De_A$ and $En_A \rightarrow T_{AB_{1,2}} \rightarrow T_{B_2A} \rightarrow De_A$ and the purple paths along $En_{B_1} \rightarrow T_{B_1A} \rightarrow T_{AB_{1,2}} \rightarrow De_{B_{1,2}}$ and $En_{B_2} \rightarrow T_{B_2A} \rightarrow T_{AB_{1,2}} \rightarrow De_{B_{1,2}}$.

Input: A_t, $B_{1,t}$, $B_{2,t}$ molecule training sets. A_v molecule validation set.
Train domain's A METN on A_t for Ep_A epochs (Section 3.4)
Train domain's B_1 METN on $B_{1,t}$ for Ep_{B_1} epochs (Section 3.4)
Train domain's B_2 METN on $B_{2,t}$ for Ep_{B_2} epochs (Section 3.4)
for $epoch = 1, 2, \ldots, E_{max}$ **do**
  Sample mini-batches $a \in A_t$, $b_1 \in B_{1,t}$, $b_2 \in B_{2,t}$
  $< a >= En_A(a), < b_1 >= En_{B_1}(b), < b_2 >= En_{B_2}(b)$
  Calculate fp_a, fp_{b_1}, fp_{b_2}
  $< b'_{1,2} >= T_{AB_{1,2}}(< a >, fp_a)$
  $< a' >= T_{B_1A}(< b'_{1,2} >, fp_a)$
  $a' = De_A(< a' >)$
  $L_{AB_1} = CE(a', a)$
  $< a' >= T_{B_2A}(< b'_{1,2} >, fp_a)$
  $a' = De_A(< a' >)$
  $L_{AB_2} = CE(a', a)$
  $< a' >= T_{B_1A}(< b_1 >, fp_{b_1})$
  $< b'_{1,2} >= T_{AB_{1,2}}(< a' >, fp_{b_1})$
  $b'_{1,2} = De_{B_{1,2}}(< b'_{1,2} >)$
  $L_{B_1} = CE(b'_{1,2}, b_1)$
  $< a' >= T_{B_2A}(< b_2 >, fp_{b_2})$
  $< b'_{1,2} >= T_{AB_{1,2}}(< a' >, fp_{b_2})$
  $b'_{1,2} = De_{B_{1,2}}(< b'_{1,2} >)$
  $L_{B_2} = CE(b'_{1,2}, b_2)$
  $L = L_{AB_1} + L_{AB_2} + \lambda_{B_1} \cdot L_{B_1} + \lambda_{B_2} \cdot L_{B_2}$
  Minimize L using Adam optimizer
  Evaluate the model every V_f epochs on A_v
  Update λ_{B_1} and λ_{B_2} according to the validation performance
  if evaluation criterion improved **then**
    save model
  end if
  if no improvement for P evaluations **then**
    stop training
  end if
end for

4.3 Empirical Methodology

We create 100 optimization candidates for each test molecule, resulting in 78,000 optimized molecules for each model. We randomly select one candidate for each source molecule and compute the metrics on the 780 optimised molecules selected (if there exists a source molecule with no valid optimized molecule for one of the models, this source molecule is discarded from the comparison for all models). This process is repeated 10 times, and the average metrics are reported. For the 780 optimized molecules we calculate the average QED and DRD2 scores, the average similarity to the source molecule and the overall success rate.

4.4 Baselines

We compare IPCA both to the SOTA supervised and unpaired methods for multi-property lead optimization:

(1) **SOTA Supervised Baseline**: **HG2G** [9] is a supervised method for multi-property lead optimization. It uses a paired training set of a lead molecule and an enhanced molecule.

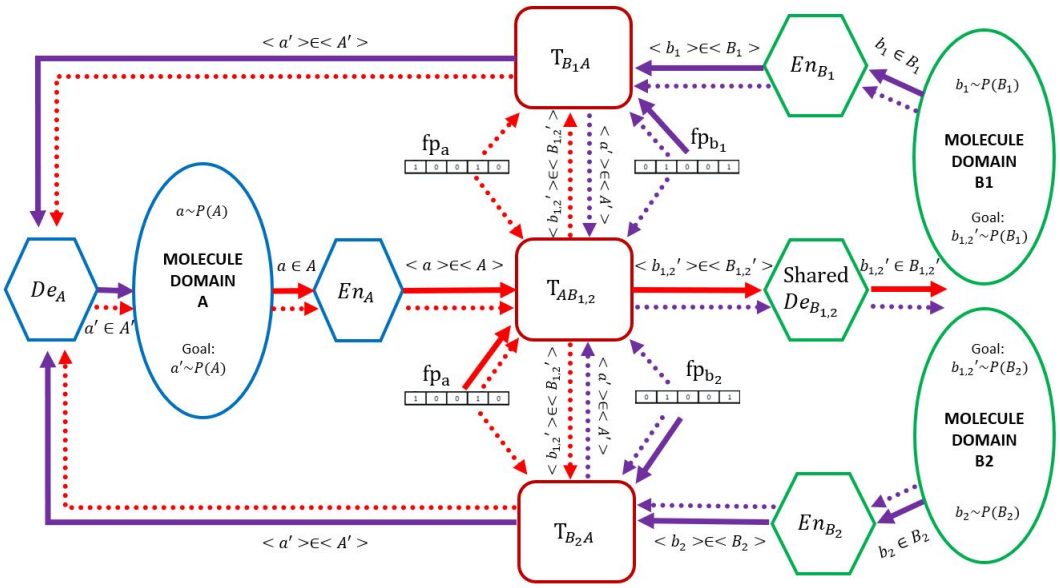

Figure 1: Molecule-to-Molecule end-to-end architecture. The training paths are marked with dotted arrows while the inference paths are marked with thick solid arrows.

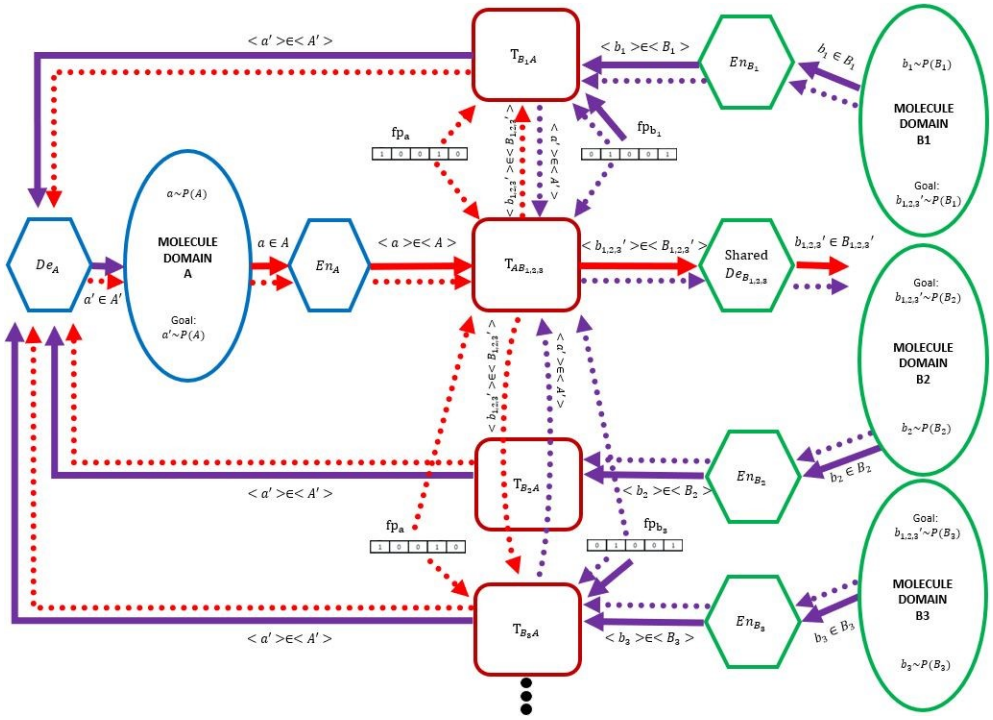

Figure 2: Molecule-to-Molecule end-to-end architecture with additional property $prop_3$. Encoder En_{B_3} and translator T_{B_3A} are added to support the additional property. The same approach may be used to support more properties.

(2) **Leading Unpaired Baseline**: As IPCA method is unpaired, i.e., our training set does not contain pairs of source and destination molecules as opposed to HG2G for example, we also include the leading unpaired method **JTVAE** [8]. We adapt it to the multi-property lead optimization task. For single-property molecule optimization JTVAE is jointly trained with a property predictor C (commonly a feed-forward network) to predict the property from the latent embedding [8]. For

multi-property optimization we trained a predictor for each property. Gradient ascent is then applied in the latent space to improve the sum of the predicted scores of the classifiers. The molecule with the highest score that satisfies the similarity constraint is outputted. Note that this model produces one output molecule per source molecule. As a result, calculating the metrics on its outputs leads to zero standard deviation.

4.5 Implementation Details for IPCA

For the training of IPCA, molecule domain A is created by selecting molecules with low QED and DRD2 values from the training set. Molecule domain B_1 is created by selecting molecules with high QED values ($QED \geq 0.92$) and domain B_2 is created by selecting molecules with high DRD2 values ($DRD2 \geq 0.65$).

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. **Reproducibility**: We set seed 50, other seeds yield similar results. Hyper-parameters are set following evaluation on a validation set. **METN:** Embedding dimension 625 (experimented with 500, 625, 750), λ_0=10 (experimented with 1, 10, 20), max molecule length during inference 90 (experimented with 80-120). Encoder – bidirectional GRU, hidden dimension 625, followed by 2 FC with 625 neurons, for μ and σ. Decoder – FC layer with 1250 neurons, followed by 3-layered GRU (experimented with 1-5), dropout 0.5 between layers, hidden dimension 1250 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 $Ep_A = 20$, $Ep_{B_1} = 5$ (QED), $Ep_{B_2} = 25$ (DRD2) epochs (experimented with 5, 10, 15, 20, 25, 30). **Translators:** 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 800 and 625 neurons separated by BN, LeakyReLU and Dropout (0.2). These types of structures presented outstanding results in image translation and style transfer tasks [12, 27].

Whole model training: Adam optimizer, initial learning rate $1.5 \cdot 10^{-4}$ (experimented with $1 \cdot 10^{-4}$-$5 \cdot 10^{-4}$) and a linear decay towards 0 from epoch 120 (experimented with 80-140). Mini-batch size 32. Max epochs E_{max}=150 (experimented with 120-170). Data is shuffled during training. Evaluation frequency V_f=3, early stopping with P=15 evaluations. λ_1=30, λ_2=90 (experimented with 1,30,60,90,120). $(wanted_1, wanted_2) \in \{(0.7, 0.3), (0.75, 0.35), ..., (0.9, 0.5)\}$ according to the specific experiment property threshold settings. Training time: 5 hours. Inference time: $\sim$ 10 seconds on the whole test set. Our code and data are publicly available.

5 EXPERIMENTAL RESULTS

In this section, we present the comparison of our model with the baselines, show qualitative examples of molecule optimization and perform numerous ablation tests.

5.1 Main Result

We study the behavior of the algorithm on several settings with different QED and DRD2 success thresholds. Given a lead molecule, an optimization is successful if the generated molecule's QED and DRD2 are above those success thresholds and it is similar enough to the lead molecule. Each pair of thresholds constitutes a separate dataset. For each pair of thresholds we perform an experiment studying the impact of the number of examples that satisfy all properties on the performance of the algorithm. Figure 3 presents the results for QED threshold of 0.7 and DRD2 threshold of 0.3. Similar results were obtained with other thresholds (we experimented with (0.7,0.3) – (0.9,0.5) with lags of 0.05). We observe that the performance of IPCA is stable, slightly decreases as the amount of training examples satisfying all conditions decreases, whereas HG2G performance is highly impacted. We hypothesize that while IPCA focuses on optimization of each property and its contribution to the shared optimized embedding space, HG2G tries to optimize all properties at once. As a result, IPCA's performance is stable as long as molecules with even one high property exist, whereas HG2G quickly deteriorates as the number of molecules satisfying all properties decreases. Similar results of HG2G were reported by [9]. Similarly to IPCA, JTVAE presents stable results, but with significantly degraded SR. It suggests that "simple" gradient ascent optimizing both properties is insufficient for the task of multi-property optimization.

We now dwell deeper into the common scenario in nature, where molecules satisfying all properties to be optimized are scarce. We remove training examples with both QED and DRD2 above the success thresholds. As this naturally affects the number of training examples, we repeat this process for several property thresholds. We start from the thresholds set in [9], according to which $QED > 0.9$ and $DRD2 > 0.5$. We then lower QED and DRD2 threshold values by 0.05 for each experiment. Table 1 summarizes the amount of examples removed from each training set. The results of IPCA and the other baselines over these datasets are reported in Table 2.

Note that JTVAE generates a single molecule, therefore the zero standard deviation. Observing the results summarized in Table 2, we notice that as the success thresholds increase, creating a molecule with such high DRD2 and QED values becomes an increasingly more challenging optimization task for all algorithms, even though the number of training examples increases. However, the success rate of the IPCA model is consistently superior to that of JTVAE and HG2G (regardless of the success thresholds and number of training examples). The difference between success rates grows as the thresholds are lowered, reaching a factor of 2 and over at the lowest thresholds. We draw the reader's attention that although the SR might seem generally low, it is quite dramatic for the drug development industry. Even one successful molecule can progress us towards a drug. Note that we calculated the standard deviation and Cohen's d effect size and ensured it is over 1, i.e. our model's SR is substantially different with a large effect from the SOTA.

The IPCA model achieves higher average DRD2 while the HG2G model achieves higher QED. JTVAE struggles to optimize DRD2. This might stem from the fact that naturally there are more molecules with high QED, but high DRD2 is relatively rare making its optimization more challenging compared to QED.

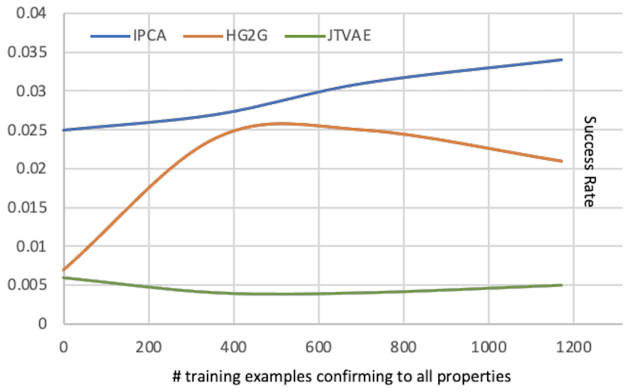

Figure 3: Success rate as a function of the number of training examples confirming to all properties.

Note that the average QED, DRD2 and similarity values alone do not provide the complete performance analysis since the multi-optimization task requires all properties to exist at once in the optimized molecule. The success rate is therefore a more appropriate metric by which to evaluate the overall performance.

As expected, the validity of the HG2G model is superior to that of the IPCA model. This is less of a concern as we can produce a large number of candidates for each molecule, screening out non-valid molecules automatically using RDkit.

Overall, the results show that our method outperforms both supervised paired methods, e.g. HG2G, the leading method for multi-property optimization, and unsupervised unpaired methods, e.g. JTVAE.

5.2 Qualitative Examples

Figures 4 – 6 present several examples of molecules generated during the testing of IPCA $_{70,30}$. Note that IPCA $_{70,30}$ model has never seen a molecule with $QED > 0.7$ and $DRD2 > 0.3$, nevertheless, it is able to generate novel molecules above these thresholds and similarity over 0.3.

The generated molecules depicted in Figure 4 preserve high fingerprint similarity (over 0.6) to the lead molecules. IPCA $_{70,30}$ manages to optimize the lead molecule not only when its QED property is already above the threshold (Figure 4B), but also when it is below the threshold (Figure 4A). In Figure 4A, the QED score improves from 0.6493 to 0.7209 and the DRD2 score improves from 0.0062 to 0.9901, while keeping a high similarity of 0.6875 during this optimization. In Figure 4B, the model keeps the already successful QED property (slight improvement from 0.7856 to 0.7862) and optimizes the DRD2 score from 0.0284 to 0.4733, while reaching 0.6363 similarity between the generated and the lead molecules.

Figure 5 shows several examples where the lead molecules have relatively low properties ($QED < 0.4$ and $DRD2 < 0.05$). Nevertheless, IPCA $_{70,30}$ performs well and optimizes both properties while keeping similarity to the lead molecules. In Figure 5A, the model improves the QED score from a low value of 0.3940 to 0.9079 and the DRD2 score from 0.0425 to 0.7138 with a similarity of 0.3602 between the lead and the generated molecules. In Figure 5B, the model improves the QED score from even a lower value of 0.3097 to 0.8209 and the DRD2 score from 0.0327 to 0.5324 preserving a similarity of 0.3559 between the lead and the generated molecules.

Thresholds (QED, DRD2)	Training Pairs Removed (Above Thresholds)	Training Pairs In Set (Below Thresholds)
(0.90, 0.50)	1.6%	120, 745
(0.85, 0.45)	4.5%	117, 252
(0.80, 0.40)	9.4%	111, 131
(0.75, 0.35)	12.96%	106, 811
(0.70, 0.30)	17.07%	101, 758

Table 1: Training set sizes.

Figure 6 presents several examples where the generated molecules have relatively high properties. In Figure 6A, our model got a lead molecule with a QED score of 0.7493 and a DRD2 score of 0.0015 and was able to generate a molecule with a QED score of 0.9139 and a DRD2 score of 0.8973. The similarity is 0.4677. In Figure 6B, the QED score improved from 0.5041 to 0.8376, which is a meaningful. However the truly impressive improvement is the DRD2 score improvement from 0.0005 in the lead molecule to 0.9999 in the generated molecule.

In all the examples, investigating the SMILES strings and their corresponding chemical structure sketch reveals high resemblance between the lead and the generated molecules. It shows once more that IPCA not only improves molecule's properties, but it does it while preserving its chemical features.

5.3 Ablation Tests

We perform several ablation experiments evaluating different aspects of our model. Table 3 describes the results of these experiments on IPCA $_{70,30}$.

5.3.1 Unified Target Domains. A plausible baseline for the multi-property optimization problem might be a reduction of the problem to a single property optimization problem. That is, instead of creating a target domain per property, a simplified solution might be to follow the SOTA single property optimization architecture [1] and construct a single target domain for all the high property molecules. We then train the model to translate molecules between the source domain, which contains poor property molecules, and the target domain, which contains high property molecules. Furthermore, in that approach the number of training cycles is reduced from four to two and the adaptive loss is not used.

Hence, we examine a simplified version of our architecture in which there is only one optimized domain B which consists of the union of B_1 and B_2, i.e., B contains molecules with high $prop_1$ or $prop_2$. Figure 7 illustrates this architecture. Marked in yellow is the unified component. In this architecture, there is only one optimization path with two parallel training cycles: $A \rightarrow < A > \rightarrow < B' > \rightarrow < A' > \rightarrow A'$ and $B \rightarrow < B > \rightarrow < A' > \rightarrow < B' > \rightarrow B'$.

We observe that the performance is degraded, likely due to the model's inability to separate between the two optimization goals. It focuses on the "easy to optimize" property and neglects the other. We notice that QED remains stable while DRD2 is much lower. We deduce that having a target domain per property, together with its encoder, translator and training cycle is useful. It enables

Type	Method	DRD2	QED	Similarity	SR	Validity	Novelty	Diversity
Paired	HG2G $_{90,50}$	0.115 ± 0.006	0.867 ± 0.003	0.232 ± 0.003	**0.003** ± 0.002	1.000	0.875 ± 0.008	0.960 ± 0.006
	HG2G $_{85,45}$	0.178 ± 0.009	0.848 ± 0.004	0.221 ± 0.002	0.005 ± 0.002	1.000	0.888 ± 0.011	0.965 ± 0.006
	HG2G $_{80,40}$	0.143 ± 0.009	0.853 ± 0.005	0.233 ± 0.004	0.006 ± 0.002	1.000	0.886 ± 0.015	0.960 ± 0.006
	HG2G $_{75,35}$	0.206 ± 0.013	0.826 ± 0.004	0.230 ± 0.004	0.008 ± 0.003	1.000	0.871 ± 0.012	0.948 ± 0.008
	HG2G $_{70,30}$	0.102 ± 0.009	0.850 ± 0.005	0.228 ± 0.003	0.007 ± 0.002	1.000	0.906 ± 0.012	0.962 ± 0.004
Unpaired	JTVAE$_{90,50}$	0.016 ± 0.000	0.782 ± 0.000	0.412 ± 0.000	0.000 ± 0.000	1.000	0.994 ± 0.000	1.000 ± 0.000
	JTVAE$_{85,45}$	0.012 ± 0.000	0.791 ± 0.000	0.436 ± 0.000	0.001 ± 0.000	1.000	0.995 ± 0.000	1.000 ± 0.000
	JTVAE$_{80,40}$	0.014 ± 0.000	0.783 ± 0.000	0.434 ± 0.000	0.001 ± 0.000	1.000	0.995 ± 0.000	1.000 ± 0.000
	JTVAE$_{75,35}$	0.018 ± 0.000	0.773 ± 0.000	0.404 ± 0.000	0.004 ± 0.000	1.000	0.995 ± 0.000	1.000 ± 0.000
	JTVAE$_{70,30}$	0.020 ± 0.000	0.777 ± 0.000	0.458 ± 0.000	0.006 ± 0.000	1.000	0.992 ± 0.000	1.000 ± 0.000
	IPCA $_{90,50}$	0.365 ± 0.008	0.770 ± 0.003	0.206 ± 0.001	0.001 ± 0.001	0.999	0.989 ± 0.001	0.929 ± 0.006
	IPCA $_{85,45}$	0.398 ± 0.013	0.754 ± 0.005	0.190 ± 0.002	**0.008** ± 0.003	0.999	0.987 ± 0.003	0.867 ± 0.007
	IPCA $_{80,40}$	0.248 ± 0.009	0.784 ± 0.004	0.233 ± 0.002	**0.014** ± 0.004	1.000	0.995 ± 0.003	0.991 ± 0.004
	IPCA $_{75,35}$	0.243 ± 0.009	0.732 ± 0.003	0.195 ± 0.002	**0.016** ± 0.003	0.850	0.998 ± 0.001	0.999 ± 0.001
	IPCA $_{70,30}$	0.174 ± 0.009	0.762 ± 0.003	0.239 ± 0.002	**0.025** ± 0.003	0.946	0.995 ± 0.002	0.997 ± 0.002

Table 2: Evaluation over multiple datasets of DRD2 and QED properties. In bold: best result for SR (cohen's d effect size over 1).

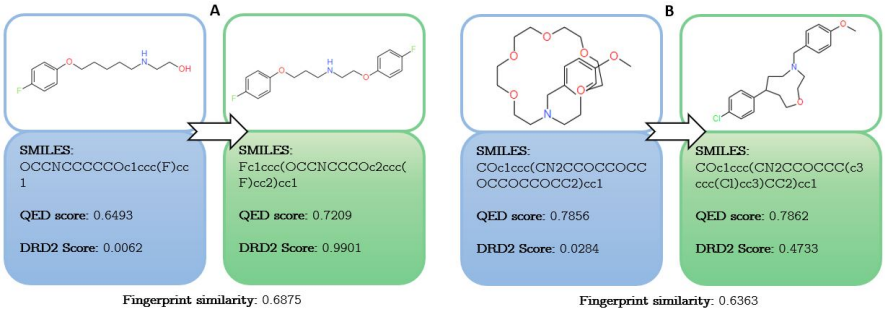

Figure 4: Sample of molecules generated using IPCA $_{70,30}$. The generated molecules preserve high similarity to the lead molecules.

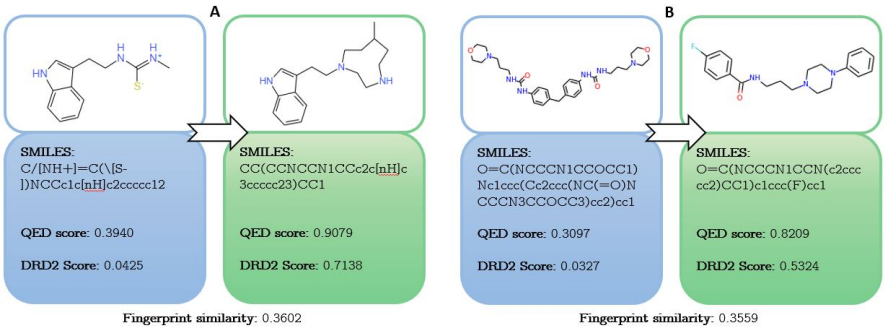

Figure 5: Sample of molecules generated using IPCA $_{70,30}$. The lead molecules have low QED and DRD2 properties.

Method	DRD2	QED	Similarity	SR	Validity	Novelty	Diversity
IPCA $_{70,30}$	0.174 ± 0.009	0.762 ± 0.003	0.239 ± 0.002	**0.025** ± 0.003	0.946	0.995 ± 0.002	0.997 ± 0.002
Unified target domains$_{70,30}$	0.071 ± 0.005	0.760 ± 0.003	0.224 ± 0.002	0.013 ± 0.002	0.849	0.999 ± 0.000	0.998 ± 0.002
Non-adaptive loss$_{70,30}$	0.129 ± 0.001	0.809 ± 0.003	0.257 ± 0.002	0.019 ± 0.002	0.980	0.999 ± 0.000	0.997 ± 0.002
No embedding$_{70,30}$	0.951 ± 0.006	0.600 ± 0.004	0.136 ± 0.000	0.004 ± 0.001	1.000	1.000 ± 0.000	0.175 ± 0.011

Table 3: Ablation experiment evaluation results for IPCA. In bold: best result for SR (cohen's d effect size over 1).

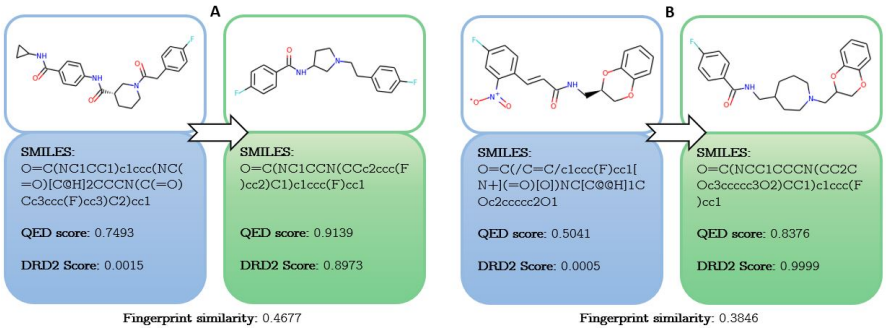

Figure 6: Sample of molecules generated using IPCA $_{70,30}$. The generated molecules have high QED and DRD2 properties.

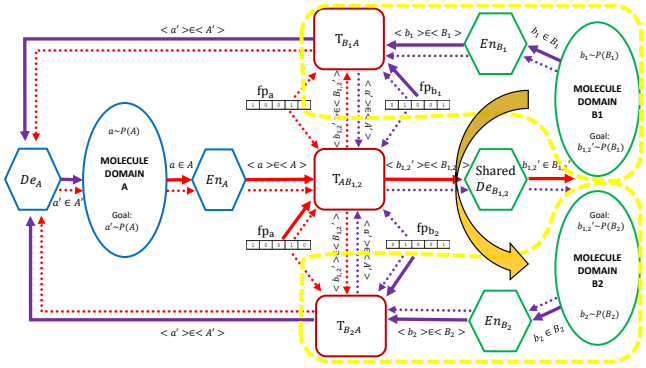

Figure 7: Unified target domains ablation architecture. Marked in yellow is the unified component.

IPCA to perform high quality multi-property optimization by first focusing on each property individually and then constraining all these using a shared embedding space with the help of the adaptive loss. Additionally, we conclude that leveraging the strategies of single-property optimization, such as UGMMT [1], don't translate directly to the multi-property optimization task.

5.3.2 Non-adaptive Loss. To examine the contribution of the adaptive loss function, we compare our model to a model with a fixed weighted loss: $L = L_{AB_1} + L_{AB_2} + \lambda_{B_1} \cdot L_{B_1} + \lambda_{B_2} \cdot L_{B_2}$ where $\lambda_{B_1} = \lambda_1 = 30$ and $\lambda_{B_2} = \lambda_2 = 90$. We experimented on several *lambda* and report the best results. Although the weights should support the learning of "hard to optimize" properties, we observe that adapting the loss coefficients during training provides superior results.

5.3.3 No Embedding Constraint. To examine the contribution of the METN embedding (Section 3.4), we perform an experiment and discard the embedding $< a >$ created from the SMILES encoding and rely only on the source molecule fingerprints (fp_a) during inference. It is evident by the results that this embedding is indeed an important factor to the diversity, similarity and the success rate. Intuitively, eliminating the embedding and relying on a subset of information – the source molecule's fingerprints – limits the ability to generate different outputs.

6 CONCLUSIONS

For drug development, the task of optimizing multiple properties of a molecule, while remaining similar to the source molecule, is of extreme importance. As the task is extremely challenging, most prior works in molecular optimization focused on single-property optimization. In this work, we tackled the problem of multi-property molecular lead optimization.

Unlike prior approaches, that required a large training set of pairs of a lead molecule and an enhanced molecule, our approach is unpaired and does not require large paired datasets which are hard to obtain. We introduced a novel integrated poly-cyclic architecture that consists of an optimization path for each property. Those are joined by a shared embedding space from which a multi-property optimization can be inferred. This architecture can be extended to a varying number of optimized properties by adding an optimization path for each additional property and joining it with the shared embedding space. Each optimization path is composed of two training cycles. The parallel independent cycles in our architecture allow it to scale up well for additional properties. For additional property only one encoder and translator should be added (Figure 2).

A significant advantage of IPCA is its ability to infer optimized molecules in cases where such examples are not available in the training set. We empirically show that our architecture outperforms current SOTA architectures in terms of the percentage of optimized molecules which meet all optimization goals. We show that the performance gap increases as the training set contains less examples which meet all property constraints. Additionally, we perform multiple ablation tests and identify that our adaptive loss function, that automatically learns to weigh different properties during training, shows notable performance boost. We also compare against other architectural designs and learn that optimizing properties separately with common constraints, rather than together, is not only scalable as the number of optimized properties grows, but it also allows our model to learn with a small number or even without any molecules satisfying all property constraints. Due to the challenging nature of this task, only two properties datasets have been investigated in prior works. For future work, we plan to test our architecture for a larger number of properties and study the impact of using powerful known architectures such as transformers to replace IPCA's encoders-decoders, which might boost the performance even more. We believe our method lays strong foundations to an automatic-algorithmic drug discovery process.

REFERENCES

[1] Guy Barshatski and Kira Radinsky. 2021. Unpaired Generative Molecule-to-Molecule Translation for Lead Optimization. In *KDD '21: The 27th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, Singapore, August 14-18, 2021* (Virtual Event, Singapore). ACM, 2554–2564.

[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] 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*.

[5] 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)*.

[6] 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.

[7] 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 and Data Mining* (London, United Kingdom) *(KDD '18)*. 331–339.

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

[9] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. 2019. Hierarchical graph-to-graph translation for molecules. (2019).

[10] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. 2020. Multi-objective molecule generation using interpretable substructures. In *International Conference on Machine Learning*. PMLR, 4849–4859.

[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] Marcus Olivecrona, Thomas Blaschke, Ola Engkvist, and Hongming Chen. 2017. Molecular de-novo design through deep reinforcement learning. *Journal of cheminformatics* 9, 1 (2017), 48.

[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] Mariya Popova, Olexandr Isayev, and Alexander Tropsha. 2017. Deep Reinforcement Learning for De-Novo Drug Design. *CoRR* abs/1711.10907 (2017). arXiv:1711.10907 http://arxiv.org/abs/1711.10907

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

[23] 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.

[24] Martin Simonovsky and N. Komodakis. 2018. GraphVAE: Towards Generation of Small Graphs Using Variational Autoencoders. In *ICANN*.

[25] 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.

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

[27] 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.