Multiple-objective Reinforcement Learning for Inverse Design and Identification

Haoran Wei
University of Delaware
nancywhr@udel.edu

Mariefel Olarte
PNNL
mariefel.olarte@pnnl.gov

Garrett B. Goh*
PNNL
garrett.goh@pnnl.gov

Abstract

Inverse chemical design using generative deep learning (DL) models is an emerging tool, which has early success in generating molecular structures with desired properties. We enhance this tool by developing heuristics for curriculum-learning based multiple-objective (20+ objectives) reinforcement learning, and apply it to the context of chemical identification. We develop a generative DL framework that utilizes constraints pertaining to the unknown molecule's mass and functional group composition, and show that multiple-objective RL-based generative DL models can correctly identify unknown molecules with a 80% success rate, compared to the baseline approach of 0%. Lastly, the heuristics developed are not limited to just chemistry research challenges; we anticipate any problem that utilizes reinforcement learning with multiple-objectives will benefit from it.

1 Introduction

Traditional chemical design for a target use case, such as a new drug compound, is dependent on using high-throughput experiments or simulations to quickly screen through a list of candidate chemicals, in the hope that some of them will meet the desired design requirements. Given that the chemical space spans on the order of $10^{60} \sim 10^{100}$, it comes with no surprise that this brute-force approach is highly iterative with low success rates. To address this shortcoming, recent advances in deep learning (DL) have demonstrated preliminary success in the inverse design paradigm, where desired properties are used as input, for the DL models to generate chemical structures that would satisfy the design requirements. To date, various algorithms have been proposed, including models based on variational autoencoders[11], generative RNN models[15] and GANs [9].

Besides chemical design, generative models can also be used for chemical identification. Current approaches used in identifying chemicals are limited, often relying on database matching of MS and NMR spectra to known chemicals [2, 3]. However, given the extensiveness of the chemical space, this brute-force approach would not be effective to identify all but the most common chemicals. Instead of using generative DL models to optimize for a specific chemical property, one can optimize against a set of constraints (i.e. fingerprints), such as molecular weight (MW), elemental composition and presence of specific functional groups (FG) that will constrain the search space. With a sufficient number of constraints, one may arrive at a unique solution that would satisfy all listed constraints, and in doing so, identify the unknown chemical.

2 Related Work

One of the first generative models developed were based on conditional variational autoencoders (CVAE) [7, 13], which is used to convert molecules, represented as SMILES, into a continuous vector representation. A major issue in early CVAE models is the low accuracy of valid SMILES generated, although recent work has made progress towards this goal[4]. Algorithms based on generative

Preprint. Work in progress.

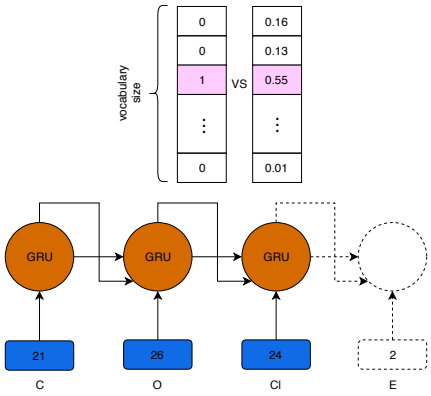


Figure 1: RNN learning the SMILES syntax

RNN models and reinforcement learning (RL) [15] and GANs [10] have also been reported. These generative models tend to result in a higher proportion of valid SMILES than CVAE-based models.

All prior work thus far have been using generative models for chemical design, typically optimizing for a single objective [16, 15]. More recent work has included a few objectives, with typically no more than 5 properties optimized simultaneously[13]. In addition, the use of generative models for chemical identification has yet to be reported. Our contributions are therefore as follows:

- We develop training heuristics for multiple-objective (20+) reinforcement learning using a modified curriculum training approach.
- We develop the first multiple-objectives RL-based generative DL model for chemical identification of simple organic molecules that are relevant to biofuels applications.

3 Network Design

In this section, we give an overview of the system design, including a prior model for generating valid SMILES strings, and an agent model fine-tuned through curriculum-based reinforcement learning.

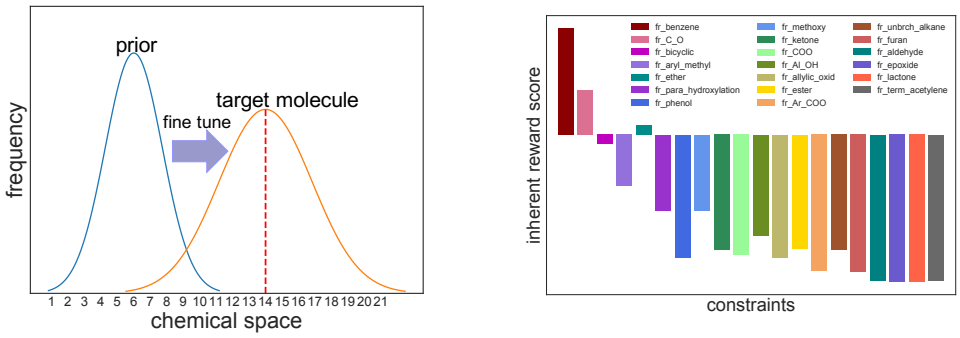
3.1 Pre-training the Prior Model

SMILES[18] is a molecule structure representation in the format of ASCII strings. Alphabets are used to represent atoms and symbols notate structures, for example, “cyclopropene" is written as “C1=CC1”. Valid formulation of molecules is therefore encoded in the syntax in the SMILES string. Previous research has demonstrated that the SMILES syntax can be learned efficiently with generative RNN models [15, 16]. Using a similar approach, we utilize a GRU-based model to learn the SMILES syntax which is termed as the “prior".

We process SMILES as text sequences , which is wrapped with a start token and a termination token. The vocabulary size of unique alphabets and symbols used in pre-training the prior model is of size 87. Each SMILES sequence is encoded to a one-hot vector as the input for pre-training the prior, and the sequence-shifted one-hot vector as its output. This means that when the input at time $t-1$ is fed into the prior, the character at time t will be predicted as a conditional probability distribution across the whole vocabulary, shown in Fig 1Assuming a single input/output pair (x, y) and the output from prior is $\hat{y}$, the loss function (cross-entropy) is as Eq 1:

$$L(y, \hat{y}; \theta) = -\sum_{i=1}^{T} y(log(\hat{y})) \tag{1}$$

where θ is the trainable weight of the prior model and $\hat{y}$ is the one-hot vector. To keep the notation simple, the loss function is written as $L(\theta)$. The prior model is optimized with mini batch $((X, Y))$ gradient descent (with batch size of 128), as Eq 2



(a) (b)

Figure 2: (a) Reinforcement learning shifts generated SMILES distribution to overlap better with desired molecules (higher reward score); (b) Average inherent reward score of the 20 FG constraints

$$\theta = \arg\min \sum_{(X,Y)} L^2(Y, \hat{Y}) \qquad \text{where} \quad \hat{Y} = \text{prior}(X; \theta) \tag{2}$$

3.2 Fine-Tuning the Agent Model through Reinforcement Learning

The pre-trained prior model was trained on ChEMBL [6] database that has 1.5 million syntactically valid SMILES and is able to generate SMILES with high validity. However, the generated SMILES will not necessarily satisfy the design or identification criteria. Generating desired SMILES, hereon defined as all valid SMILES that satisfies all constraints, is nearly intractable, as our model has a max output length of 140 and the vocabulary size is 87, which translates to a search space of up to 87^{140}. Therefore, it is necessary to tune an "agent" model initialized from the prior model to generate a higher fraction of desired SMILES.

3.2.1 Reinforcement Learning in the SMILES Context

Reinforcement learning (RL), is a commonly used method for the sequential decision-making problem and it learns by interacting with the environment. Therefore, it is a very natural way to tune an agent model. In this work, the use of RL is to find an optimal or approximate optimal policy to complete a given task. RL task is usually mathematically modeled as $\langle S, A, R \rangle$ where S is a space of states, A is a space of actions, and $R(s, a)$ is a reward measuring the performance of an action a executed in a state s. The policy is a probability distribution across all available actions in a given state. RL learns the policy by maximizing the accumulated rewards. Therefore, the action which gets higher rewards has a higher probability to be taken.

In this SMILES generation context, we define the state space (S) as the collection of previous inputs and each action is a selection of character for the next step out of the entire action space (A), which is the model's vocabulary. For example, given c_0 and c_1 as the first two-step inputs, the state for the 3rd input (indexed from 0) is $s_2 = c_0c_1$. There are 87 actions available at each state based on the 87 characters used in the vocabulary. The policy for action selection at state s_t, which is also the output of RNN network at step t, can be represented as $\pi_t(a_j|s_{t-1}), a_j \in A$, as illustrated in Fig 1. The reinforcement reward is defined at the individual SMILES level $R(y) \to r \in \mathbb{R}$. The value of each SMILES's reinforcement reward represents how many constraints are satisfied, and the reinforcement reward for a batch of SMILES is the reward summation across all SMILES.

In other words, by using RL, the SMILES generated by the model can be regarded as actions and the reinforcement reward as a measure of how far the generated SMILES are from the desired SMILES. The weights of the model can then be updated accordingly to improve its generative policy.

3.2.2 Reinforcement Fine-Tuning

The prior model is trained on a large dataset and is therefore able to capture the general SMILES syntax. Fine-tuning is necessary for improving the model's performance by addressing specific constraints identified (illustrated in Fig 2(a)). Hereon, we refer to the fine-tuned prior model as the "agent". In this work, we sample the outputs from the prior and gather the corresponding reinforcement rewards to feedback into the model for fine-tuning, represented as:

$$L_P^r(Y;\theta_P) = L_P(Y;\theta_P) + R(Y) \tag{3}$$

where $R(Y)$ is the total rewards for a generated SMILES batch Y. θ_P is the prior's trainable weights. The first term represents the syntax in SMILES while the second term measures the extent to which SMILES satisfy the desired constraint. From another perspective, the reward function trims the original prior loss function so that the desired state-action selections are enhanced by positive rewards and undesired selections are penalized by negative rewards. Therefore, the agent model can be fine-tuned from the prior by minimizing the square of an aggregated loss function:

$$L^r(Y;\theta_A) = -(L_A(Y;\theta_A) - L_P^r(Y;\theta_P)) = -(L_A(Y;\theta_A) - L_P(Y;\theta_P) - R(Y)) \tag{4}$$

where $L_A(Y;\theta_A)$ is the cross-entropy loss functions(Eg 1). In the existing literature, when there is more than 1 desired constraint, a naive (baseline) approach is to use an equally weighted loss function, as Eq 11.

$$R(Y) = \frac{1}{n}\sum_{i=0}^{n} R_i(Y) \tag{5}$$

However, the performance of this naive approach is unlikely to scale well to multiple constraints. We hypothesize that part of the issue is caused by the equal weighting, where different directional influences will cancel out one another. Another issue is the inability to find a good local minimum solution that satisfies all constraints in an extensive search space across multiple objectives. To address these limitations, we develop heuristics inspired from curriculum learning.

3.3 Curriculum Learning with Reinforcement Learning

3.3.1 Curriculum Learning

Curriculum learning (CL) is a strategy for multi-task learning, which draws parallels to human learning, where there is rank ordering of learning subtasks in a sequential manner of increasing difficulty[1]. It has been previously shown to result in improved learning speed as well as learning performance in language modeling, pattern recognition, etc [1, 8]. The difficulty level of subtasks are usually defined with respect to the training data. Specifically, easy-to-satisfy constraints due to its inherent distribution in the prior model are designated as easy task. Progressively more difficult tasks are designated by having a greater divergence from the inherent distribution relative to the prior model. Each training "phase" is a learning process at a certain level of difficulty. In this work, we propose a novel approach of combining curriculum learning heuristics in a reinforcement learning context, where we hypothesize that the agent model can be more effectively tuned to satisfy multiple constraints using this approach.

3.3.2 Curriculum-based Reinforcement Learning

In this study, we define the difficulty of each constraint as the likelihood that it is captured in the prior distribution. An inherent reward score f_k for each target constraint is calculated using Eq 6 where the reward is 1 for each SMILES string if it contains the target constraint, otherwise, -1. Using 1000 SMILES string generated by the prior model, we report the average inherent reward score as shown as Fig 2(b). The inherent reward score can be interpreted as the "default" probability that a certain constraint will be satisfied. For example, the FG constraint for benzene is about 0.8, which means that satisfying this constraint is relatively easy since it is generated 80% of the time.

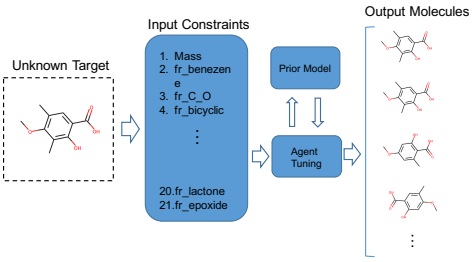


Figure 3: Workflow of inverse chemical identification: identified constraint limit the chemical search space

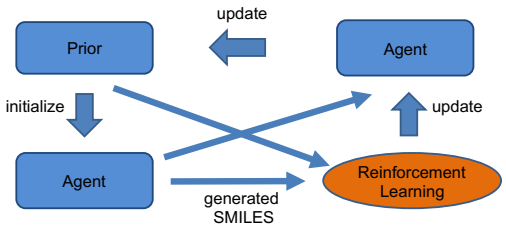


Figure 4: Illustration of Curriculum Fine-tuning

$$V_{f_k} = \frac{1}{|Y|} \sum_{y \in Y} R_{f_k}(y) \qquad R_{f_k}(y) \in [0, 1] \tag{6}$$

Each curriculum "phase" is a fine-tuning process with a reward function defined for a specified set of constraints. In each phase, the prior model is updated with the latest tuned agent model. As one progresses through phases, additional new constraints will be added into the reward function as illustrated by Fig 4. The reward function is updated accordingly as 7:

$$\hat{R}_d(Y) = (1 - \omega)\mathbb{E}(R_{j=1,\cdots,d-1}(Y)) + \omega R_d(Y) \tag{7}$$

where d represents the fine-tuning phase while ω is the weight to balance the previous fine-tuning constraints and the newly-added one(s).

4 METHODS

4.1 Data Preparation

The data preparation steps are identical to that reported previously. [15] Briefly, the prior model was trained on the ChEMBL database.[5] The characters 'Cl' and 'Br' are singularized to tokens 'R' and 'L'. A start token ('G') and a termination token ('E') is also added. The input of the prior is the encoded tokenized SMILES and output is the one-hot vectors of the same SMILES with offset by +1. Post zero padding is applied on the encoded input with a unified length (140).

4.2 Training Prior Model

The prior model network architecture starts with a single embedding layer, followed by a stacked 3-layer GRU with 512 cells per layer, and ending with a single fully-connected layer. The model was trained on mini-batch (size 128) for 20 epochs with early stopping. RMSprop is used as the optimizer with initial learning rate 0.001 and gradient values are clipped to $[-3, 3]$. The prior model generates the next token with a given input and advances time by one step and recurrently extends the input with the newly generated token until it generates the terminal token 'E'. After training, the prior model is able to generate around 95% valid SMILES.

4.3 Prospective Test Set

For the purpose of prospectively identifying unknown chemicals, we limit the search space to biomass-derived liquid relevant chemical space, which is defined as low molecular weight (MW < 200) organic compounds, containing only elements C, H and O. Using these criteria, we extracted a subset of 25,901 relevant chemicals from the original the ChEMBL database. We then used RDKit[12] to compute descriptors pertaining to molecular weight (MW) and functional groups (FG) to simulate experimental characterization data from MS and NMR sources. These descriptors are used as the constraints in our reward function.

Table 1: Fine-Tuning Methods

Method	No. of bins	No. of retrains (at each phase)	No. of phases
Retrain Fine-tuning (RF_2)	1	1	2
Curriculum Fine-tuning (CF_2)	2	0	2
Curriculum Retrain Fine-tuning (CRF_2)	2	1	2
Retrain Fine-tuning (RF_4)	1	1	4
Curriculum Fine-tuning (CF_4)	4	0	4
Curriculum Retrain Fine-tuning (CRF_4)	4	1	4
Retrain Fine-tuning (RF_6)	1	1	6
Curriculum Fine-tuning (CF_6)	6	0	6
Curriculum Retrain Fine-tuning (CRF_6)	6	1	6

4.4 Designing the Reward Function

MW constraints and 20 FG constraints were used. The reward for MW is adapted from [14]:

$$R_{\text{mw}}(y) = \max(-1, \frac{1}{10^4}(x - MW(y))^2 + 1) \tag{8}$$

The other 20 FG constraints can be represented as $\{f_1, f_2, \cdots, f_{20}\}$ and $f_i \in \{\text{False}, \text{True}\}$, where "True" means this FG is present and vice versa.

$$R_{f_i}(y) = \begin{cases} 1 & \text{if} \quad f_i \in y \quad \text{and} \quad f_i = \text{True} \\ 1 & \text{if} \quad f_i \notin y \quad \text{and} \quad f_i = \text{False} \\ -1 & \text{if} \quad f_i \in y \quad \text{and} \quad f_i = \text{False} \\ -1 & \text{if} \quad f_i \notin y \quad \text{and} \quad f_i = \text{True} \end{cases} \tag{9}$$

The MW and presence of specific functional groups can be easily evaluated with RDKit[12]. MW approximately controls the length of the SMILES string, which is a significant factor in determining the extensiveness of chemical space search. Therefore, we designated a constant non-changing weighting factor in the curriculum reinforcement reward function for MW. The ω is set as 0.5.

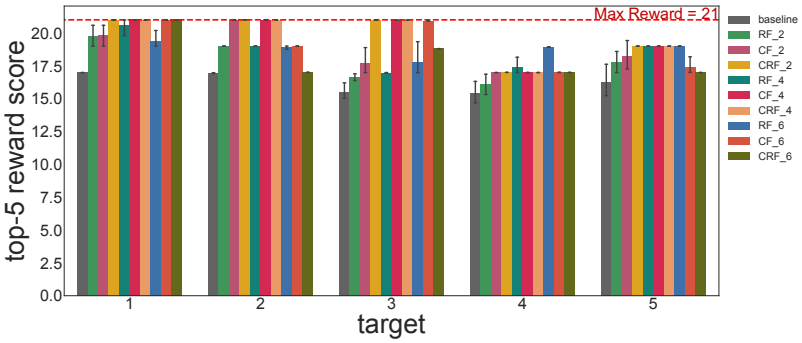
$$\hat{R}_k(Y) = \frac{1}{2}(R_{mw}(Y) + \mathbb{E}(R_{j=1,\cdots,k-1}(Y)) + \frac{1}{2}R_k(Y) \tag{10}$$

4.5 Training Agent Model

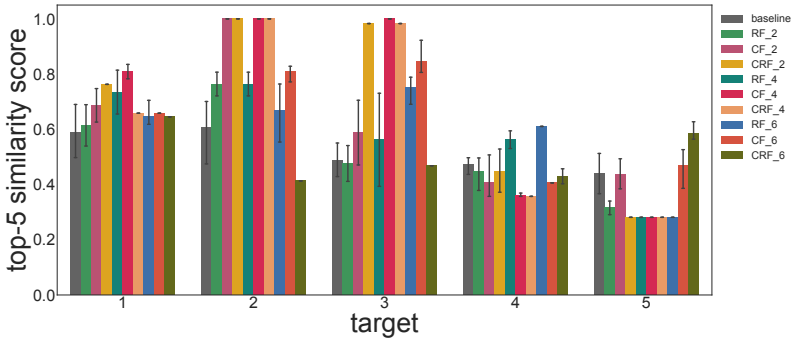
The baseline fine-tuning method uses the naive approach where all MW and 20 FG constraints are added to a single non-changing reward function, shown as Eq 11

$$R(Y) = \mathbb{E}(R_{mw} + \frac{1}{n}\sum_{i=1}^{n} R_i(Y)) \tag{11}$$

For curriculum learning methods, we investigated 3 different approaches to fine-tune the agent model: (1) Retrain-based Fine-tune (RF), (2) Curriculum Fine-tune (CF) and (3) Curriculum Retrain-based Fine-tune (CRF). RF is an extended case of the naive approach. It treats all constraints equally and uses the same reward function as the baseline method. However, at each phase, the prior is updated with the latest agent and the agent is continuously updated or retrained for a specific number of phases. In the CF method, the model is trained gradually with new constraints sequentially added at each phase. To reduce total training time, we group the constraints into multiple bins according to their difficulty. At each phase, one bin of constraints is added. For example, if we set 2 bins, there will be 10 constraints in each bin and the first bin is easier than the second bin. The reward for CF is shown in Eq 10. The CRF method combines and alternates between both approaches. This means that constraints are introduced into the reward function in a curriculum-based manner, but after each phase, we also re-train the model. A summary of the overall framework is illustrated in Fig 4 and details of the various methods settings is summarized in Table 1.



(a) top 5 average reward



(b) top 5 average similarity

Figure 5: Agent fine-tuning performance across 5 chemical targets

As with the baseline approach, the agent is updated with mini-batch gradient descent with size 128, and gradients are clipped to $[-3, 3]$. Each phase also has an early stopping algorithm, where training terminates if the reward does not improve after 50 iterations, and the last best model is saved.

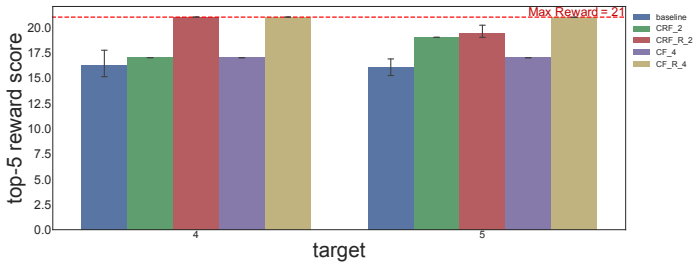
5 Experiment and Result

We randomly picked 5 biofuel-relevant molecules, as shown in Figure 5, to simulate a "blind" study of identifying an unknown chemical.

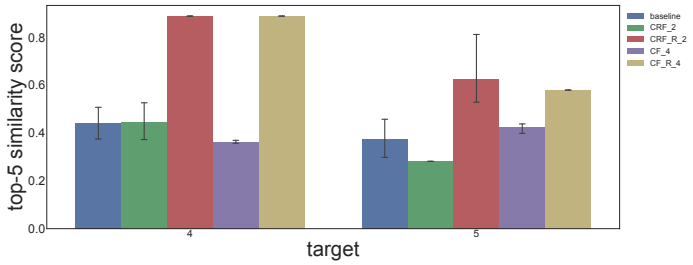
We evaluated the performance of CF heuristics with 2, 4 and 6 bins. Controlling for total training time and number of updates, we also evaluated the RF and CRF heuristics with 2, 4, and 6 retraining phases. In total, the baseline method and 9 variations CF/RF/CRF were tested for each of the 5 selected target molecules.

The final performance of each model was evaluated by the overall reinforcement reward score, and also using the Tanimoto similarity score with respect to the actual molecule. The similarity score is used only as a post ad-hoc evaluation once the model has completed generating its prediction. It had no bearing on the model's reward function. Only the MW and 20 FG constraints were used in the reward function to guide the model to a specific part of chemical space.

Model performance are summarized in Fig 5. We use "_n" to represent the number of bins. For example, "CF_2" means Curriculum Fine-tuning with 2 bins. The performance is measured with 256 generated SMILES at the last phase of each method. For each method, all generated SMILES are sorted based on the reward score in descending order. We selected the top 5 SMILES with the highest reward scores and compared their similarity with the target. An underlying assumption is that



(a) top 5 average reward improvement



(b) top 5 average similarity improvement

Figure 6: Adaptive weight refinement of reward function improves results

a model that generates SMILES with high reward scores (thus satisfying most if not all constraints), will exhibit high similarity to the target molecule.

As shown in Fig 5(a), our proposed heuristics all outperform the baseline approach in the top-5 reward score, with several models achieving close to the maximum reward score of 21. Overall, curriculum fine-tuning performs consistently better and the model that uses 4 constraint bins (CF_4) is the best method. When evaluated against similarity, our proposed methods all perform better than baseline on the first three targets (shown in Fig 5(b)), the exact molecules were identified for targets 2 and 3. However, there is an apparent disconnect for targets 4 and 5, as even though the top-5 reward score is higher than baseline, that is not translated to a higher similarity score. Based on these observations, there are two possibilities: (1) the model gets stuck at a local minimum during the early phases, and hence it was unable to arrive at a local minimum at the last phase, i.e. the agent learns early objectives “too-well” and is unable to learn subsequent objectives, or (2) certain constraints are mutually exclusive. For example, we observed that the existence of constraints "fr_allylic_oxid" limits the appearance of "fr_aldehyde" in the same molecule.

Based on the hypothesis of the model getting trapped at a "bad” local minimum in early phases, we explore additional heuristics that tune the agent model by using a reward function that increases the weighting of specific constraints that are not well learned, using Eq 12.

$$R^{'} = \mathbb{E}(R_{mw}, R_k, \mathbb{E}_{-k}(R_1, \cdots, R_{k-1}, R_{k+1} \cdots, R_{20})) \tag{12}$$

where R_k is the unlearned constraint and it is the average value if there are more than 1 unlearned constraint. Using these heuristics, we refined the CRF_2 and CF_4 models on targets 4 and 5, and used the ‘_R’ notation at the middle to represent this refinement approach, with ’CF_R_4’ representing the refined curriculum learning heuristics with 4 bins. Using this refined approach, the resulting top-5 reward score achieves near maximum (Fig 6(a)). Furthermore, we also observed that similarity to the targets were significantly improved (Fig 6(b)) .

Lastly, we visualized the outcome of the results using the best models for each target and summarized it in Fig 5. The generated SMILES correctly identified the unknown target 2, 3 and 5. Target 1 identification was sufficiently close, as structural isomers of the target was generated. We note that

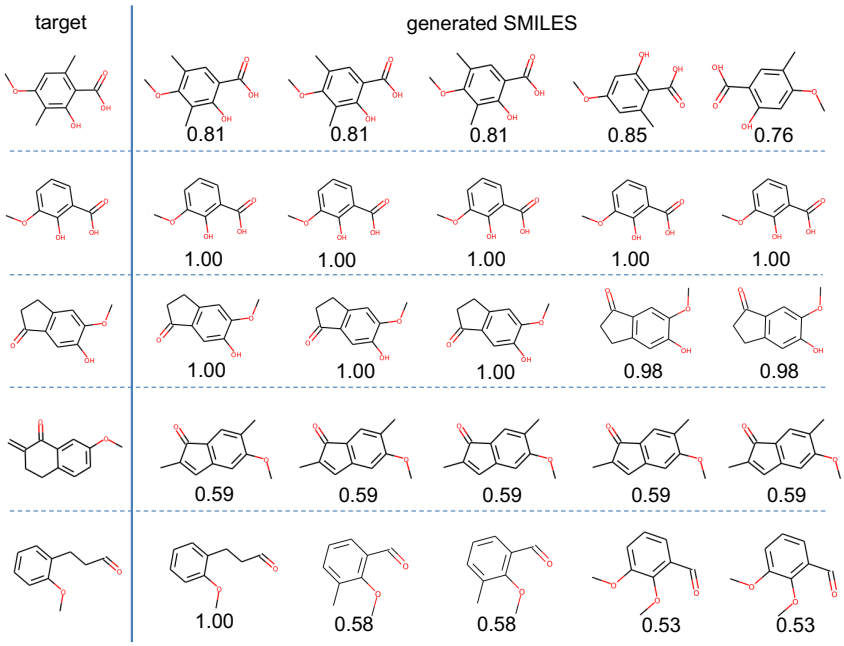


Figure 7: Agent generated molecules comparing with targets

the prediction of target 4 is not the same as its target, and we hypothesize that this is because there may not be sufficient or appropriate constraints for that structure, and so the search space remains large (i.e. it is an undetermined search problem).

6 Conclusion

In conclusion, building on earlier work [15, 17, 14] of RL-based generative RNN model, we enhance the functionality of such methods to include multiple-objectives in the reward function. In this study, we proposed and evaluated several curriculum-based reinforcement learning heuristics, and showed that at least 21 different objectives that incorporates both MW and FG constraints can be simultaneously optimized.

The naive approach of an equally weighted reinforcement learning heuristic with a single-pass training achieves poor results, both in its ability to achieve maximum reward score as well as similarity to the unknown target. In general, our results indicate that a curriculum-learning approach consistently outperforms baseline. However, there are complications associated with having too many learning phases, possibly because it gets trapped in a bad local optimum in the beginning, and so is unable to learn latter objectives effectively. To address these limitations, we developed further heuristics that adaptively overweights unlearned objectives. Our results show that for all 5 molecules tested, maximum top-5 reward score can be obtained in all 5 cases tested. In addition, a maximum top-5 reward score also translates well to chemical similarity, as 4 out of 5 molecules were identified correctly. Lastly, the curriculum-learning based heuristics developed in this work, is also a significant improvement to the baseline model, which does not even identify a single molecule correctly.

References

[1] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th annual international conference on machine learning*, pages 41–48. ACM, 2009.

[2] Kerem Bingol. Recent advances in targeted and untargeted metabolomics by nmr and ms/nmr methods. *High-throughput*, 7(2):9, 2018.

[3] Rene M Boiteau, David W Hoyt, Carrie D Nicora, Hannah A Kinmonth-Schultz, Joy K Ward, and Kerem Bingol. Structure elucidation of unknown metabolites in metabolomics by combined nmr and ms/ms prediction. *Metabolites*, 8(1):8, 2018.

[4] Hanjun Dai, Yingtao Tian, Bo Dai, Steven Skiena, and Le Song. Syntax-directed variational autoencoder for structured data. *arXiv preprint arXiv:1802.08786*, 2018.

[5] Anna Gaulton, Louisa J Bellis, A Patricia Bento, Jon Chambers, Mark Davies, Anne Hersey, Yvonne Light, Shaun McGlinchey, David Michalovich, Bissan Al-Lazikani, et al. Chembl: a large-scale bioactivity database for drug discovery. *Nucleic acids research*, 40(D1):D1100–D1107, 2011.

[6] Anna Gaulton, Anne Hersey, Michał Nowotka, A Patrícia Bento, Jon Chambers, David Mendez, Prudence Mutowo, Francis Atkinson, Louisa J Bellis, Elena Cibrián-Uhalte, et al. The chembl database in 2017. *Nucleic acids research*, 45(D1):D945–D954, 2016.

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

[8] Alex Graves, Marc G Bellemare, Jacob Menick, Remi Munos, and Koray Kavukcuoglu. Automated curriculum learning for neural networks. *arXiv preprint arXiv:1704.03003*, 2017.

[9] Gabriel Lima Guimaraes, Benjamin Sanchez-Lengeling, Carlos Outeiral, Pedro Luis Cunha Farias, and Alán Aspuru-Guzik. Objective-reinforced generative adversarial networks (organ) for sequence generation models. *arXiv preprint arXiv:1705.10843*, 2017.

[10] Artur Kadurin, Alexander Aliper, Andrey Kazennov, Polina Mamoshina, Quentin Vanhaelen, Kuzma Khrabrov, and Alex Zhavoronkov. The cornucopia of meaningful leads: Applying deep adversarial autoencoders for new molecule development in oncology. *Oncotarget*, 8(7):10883, 2017.

[11] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*, 2013.

[12] Greg Landrum et al. Rdkit: Open-source cheminformatics, 2006.

[13] Jaechang Lim, Seongok Ryu, Jin Woo Kim, and Woo Youn Kim. Molecular generative model based on conditional variational autoencoder for de novo molecular design. *arXiv preprint arXiv:1806.05805*, 2018.

[14] Daniel Neil, Marwin Segler, Laura Guasch, Mohamed Ahmed, Dean Plumbley, Matthew Sellwood, and Nathan Brown. Exploring deep recurrent models with reinforcement learning for molecule design. 2018.

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

[16] Mariya Popova, Olexandr Isayev, and Alexander Tropsha. Deep reinforcement learning for de novo drug design. *Science advances*, 4(7):eaap7885, 2018.

[17] Marwin HS Segler, Thierry Kogej, Christian Tyrchan, and Mark P Waller. Generating focused molecule libraries for drug discovery with recurrent neural networks. *ACS central science*, 4(1):120–131, 2017.

[18] David Weininger, Arthur Weininger, and Joseph L Weininger. Smiles. 2. algorithm for generation of unique smiles notation. *Journal of Chemical Information and Computer Sciences*, 29(2):97–101, 1989.