Curiosity in exploring chemical space: Intrinsic rewards for molecular reinforcement learning

Luca A. Thiede
Department of Physics, University of Göttingen, Germany
Department of Computer Science, University of Toronto, Canada
luca.thiede@yahoo.com

Mario Krenn
Chemical Physics Theory Group, Department of Chemistry, University of Toronto, Canada
Department of Computer Science, University of Toronto, Canada
Vector Institute for Artificial Intelligence, Toronto, Canada
mario.krenn@utoronto.ca

AkshatKumar Nigam
Department of Computer Science, University of Toronto, Canada
Chemical Physics Theory Group, Department of Chemistry, University of Toronto, Canada
akshat.nigam@mail.utoronto.ca

Alán Aspuru-Guzik
Chemical Physics Theory Group, Department of Chemistry, University of Toronto, Canada
Department of Computer Science, University of Toronto, Canada
Vector Institute for Artificial Intelligence, Toronto, Canada
Canadian Institute for Advanced Research (CIFAR) Lebovic Fellow, Toronto, Canada
alan@aspuru.com

1 Introduction

The development of new drugs and functional materials is an important but expensive process. It can be framed as an optimization problem of desired properties over chemically stable and synthetically feasible molecules, denoted as inverse molecular design problem [1, 2]. The search space is enormous though [3] and therefore exhaustive search is not feasible. Therefore, various A.I. approaches exist to tackle this problem, including variational autoencoders (VAEs) [4, 5], generative adversarial networks (GANs) [6] or genetic algorithms [7, 8, 9].

One other approach is Reinforcement Learning which allows for de novo molecular design [10], potentially far away from any known data distribution [11, 12, 13, 14]. However, due to the vast chemical space, efficient exploration is necessary.

Here we take inspiration from the field of RL for video games, where the idea of curiosity [15] was able to demonstrate exceptional results without access to actual rewards from the environment [16]. Curiosity falls under the wider category of intrinsic motivation techniques [15, 17, 18], which are loosely modeled after human curiosity. Inspired by this work we propose intrinsic motivation for molecular design and show that the most *curious agents* perform best in three different benchmarks.

34th Conference on Neural Information Processing Systems (NeurIPS 2020), Vancouver, Canada.

2 Reinforcement Learning for molecular design

In Reinforcement Learning, we try to find a policy $\pi(a_t|s_t)$, that outputs an action a_t given a state s_t, so that the reward r_t it receives is maximized over an episode. We use PPO to train the policy with hyperparameter provided in the original paper [19].
For molecular design, we define the state s_t as the SELFIE [20] string (a string representation for molecules with 100% validity for any string) that is so far constructed. The action a_t is the next character to be appended to the string. The molecule is finished either when the max number of steps is reached, which we set to 35 throughout our experiments, or the agents use the *[STOP]* symbol.
For some property p that we wish to optimize, and by denoting the molecule at time step t as $\text{mol}(t)$, the reward can be formulated as

$$r_t = p(\text{mol}(t)) - p(\text{mol}(t-1)) \tag{1}$$

since the cumulative reward $\sum_{t=0}^{T} \gamma^t r_t = p(\text{mol}(t))$ for $\gamma = 1$.

We identify two problems with RL for molecular design. One is, that the agent potentially has to navigate a vast chemical space. The second problem is that RL does not optimize for what we really care about in molecular design: We are interested in the molecule with the highest reward. However, RL optimizes for a policy that yields the highest expected reward, which is not the same as the highest reward for any policy with entropy $\mathbb{H}(\pi) > 0$ (so for any non-deterministic policy). $\mathbb{H}(\pi) > 0$ is needed for exploration though. So the need for exploration is in direct conflict with the fidelity of the objective function. This can be seen in figure 1. Note, that the highest peak of the reward function is around 10, but the highest peak of the expected reward is around -10. For molecular string representations, the high but skinny peak translates to an optimal string with high reward, where just a few errors from the optimal string cause a significant drop in reward. The lower but wider peak corresponds to a locally optimal string that is more robust to errors.

3 Related Work

The literature on reinforcement learning often distinguishes between intrinsic and extrinsic rewards. An extrinsic reward is anything that comes directly from the environment. Intrinsic rewards are any rewards that are generated by the agent itself.
[15] introduced an intrinsic reward called curiosity. The basic idea of curiosity is to guide the exploration of an agent into regions of the state space, where it has not understood the effect of its actions on the environment. They introduced a separate network, that tries to predict the next state after taking an action. Then, the error of that prediction is the intrinsic reward, and the total reward is

$$r_{\text{total}}(t) = r_{\text{extrinsic}}(t) + \alpha r_{\text{intrinsic}}(t) \tag{2}$$

The curiosity module can also be seen as a memory in the state-action space, since the more often the agent is in a certain region of that space, the smaller the prediction error is going to be. By implicitly remembering the regions of the state-action space it has already explored, the agent will continue exploring new regions and not get stuck in local optima. This situation is depicted in 1. This way, when applied to molecular design, it can also help with the problem described in 2 as the agent will not get stuck in the global optima of the expected reward that is actually only a local optimum for the reward function we care about.

4 Curiosity for molecular design

Building on this intuition, we propose to use a prediction module that predicts the property of the next molecule, and add the prediction error

$$r_{\text{intrinsic}}(t) = \text{dist}(\widehat{\text{p}}(\text{mol}(t,\theta),\eta), \text{p}(\text{mol}(t,\theta)) \cdot \text{mask}(\text{mol}(t,\theta)_{1\ldots\text{batch size}}) \tag{3}$$

as an additional reward signal (see figure 2). Here $\hat{\text{p}}(\cdot,\eta)$ is the model parameterized by η that tries to predict the real value of the considered property of the molecule. $\text{mol}(t,\theta)$ is the molecule the agent

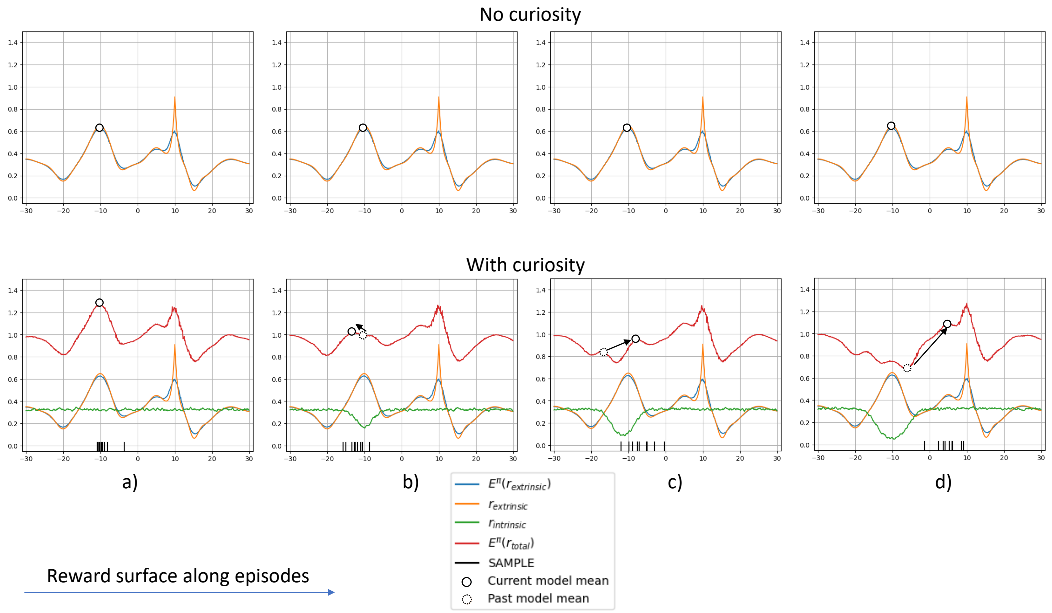

Figure 1: A schematic illustration of how the reward surface changes during training.
Upper row: Without curiosity, the reward stays constant during training. Once the agent is on a local or global maximum, it will not leave it anymore. Also note, that the global maximum of the extrinsic reward (what we care about), is not the same as the global maximum of the expected extrinsic reward (what we optimize for without curiosity)
Bottom row: Step a): The predictor network is not trained yet. The model generates a batch of samples and trains the predictor; Step b) The intrinsic reward (prediction error) goes down around the optimum, and the agent moves to the left into a new optimum; Step c) The intrinsic reward around the local optimum goes down after a while and the agent moves to the right; Step d): Again, the intrinsic reward goes down around the local optimum and the agent moves further to the right, approaching the desired optimum

parameterized by θ generates at time step t and $\text{dist}(\cdot,\cdot)$ is a distance metric, for example $L1$ or $L2$. We also optionally multiply the prediction error by a function $\text{mask}(\text{mol}(t,\theta)_{1\ldots\text{batch size}})$ that gets the whole batch of molecules as input, and masks off the curiosity reward for all molecules which target property is worse than the average in the batch. We call this formulation the greedy curiosity.

We also consider two options for training the predictor network: The first is to update the predictor network after every episode with the new batch of generated samples. A potential downside of this is, that the predictor might forget about older samples. The second option is to use a buffer and collect and train the predictor on all samples. One can either reinitialize the predictor every time before training, which makes it very resource-intensive or once can do warm starts. However, old samples will be seen more often than new samples, leading to overfitting. Thus we opted for reinitializing and training the predictor only two times, after 200 and again after 500 episodes.

Instead of memory in the state-action space, our curiosity module can be viewed as a memory only in the state space. We do not predict the state as in [15]. The reason is, that given the current state (the string so far), and the next action (the character to append), predicting the next state (the string so far with the new character appended) is trivial.

We also consider a very simple alternative where we explicitly store the last N molecules into a buffer and calculate the average Tanimoto Similarity (TS) of the Morgan Fingerprints (MF):

$$r_{\text{intrinsic, alternative}}(t) = -\frac{1}{N}\sum_{i=0}^{N} \text{TS}(\text{MF}(\text{mol}(t)), \text{MF}(\text{mol}_i)) \tag{4}$$

This approach has the downside, that the Tanimoto similarity of Morgan fingerprints is not problem specific. In comparison, molecules with a low prediction error are close to previously encountered molecules in some problem specific feature space.

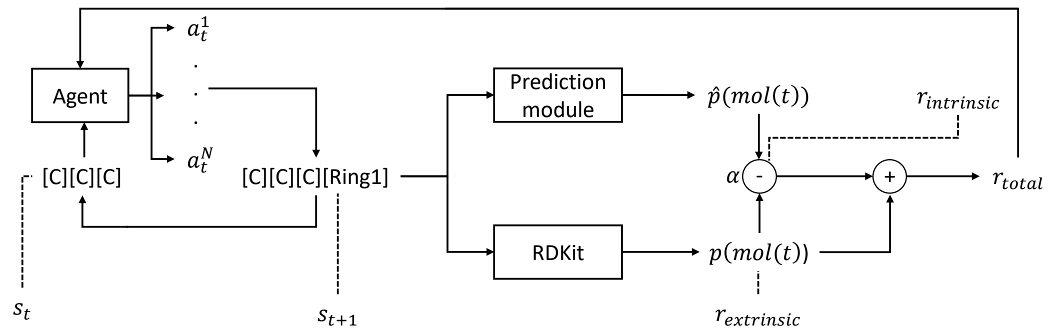

Figure 2: Architecture of the curiosity module. The so far generated string s_t is encoded and used to predict the next action a_t to append to the string. The new string is used to calculate the property that is ought to be optimized. The same string is used to estimate the target property.

Curiosity weight α	$r_{\text{intrinsic, alternative}}$	dist	Greedy curiosity	Use buffer	Best QED
1	**False**	$\boldsymbol{L_2}$	**False**	**False**	**0.918**
1	False	L_1	False	False	0.916
0.1	False	L_2	False	False	0.898
0	-	-	-	-	0.889
0.1	True	False	False	-	0.883

(a) Results for the QED task

Curiosity weight α	$r_{\text{intrinsic, alternative}}$	dist	Greedy curiosity	Use buffer	Best pLogP
1	**False**	$\boldsymbol{L_2}$	**False**	**False**	**10.364**
1	False	L_1	False	False	10.364
0	-	-	-	-	9.580
0.1	True	-	-	-	9.580

(b) Results for the plogP task

Curiosity weight α	$r_{\text{intrinsic, alternative}}$	dist	Greedy curiosity	Use buffer	Best similarity
1	**False**	$\boldsymbol{L_1}$	**False**	**False**	**0.239**
1	False	L_1	True	False	0.237
1	False	L_2	False	False	0.236
0.1	True	L_2	True	-	0.224
0	-	-	-	-	0.186

(c) Results for the similarity task

Table 1: The best values of the generated molecules, averaged over the 3 runs for the 3 best performing hyperparameter settings over all tasks. Additionally the average best value of an agent without curiosity ($\alpha = 0$), and one that uses $r_{\text{intrinsic, alternative}}$ are shown. The best agents all used the intrinsic reward ($\alpha = 1$).

5 Experiments

We test our method by training 3 agents for each set of hyperparameters on 3 different tasks. The three tasks are optimizing for QED, pLogP, and similarity (in terms of Tanimoto similarity of Morgan fingerprints) to the target molecule Celecoxib. For pLogP the global optimum is the sulfur chain. It turned out, that all agents got stuck in the local minima of the carbon chain though. Thus, we made the task slightly easier by providing the [S] symbol as the initial state.
The considered hyperparameter sets are all possible combinations of $\alpha = \{0, 0.01, 0.1, 1\}$, $r_{\text{intrinsic, alternative}}$, $L1/L2$, Greedy curiosity and Usage of the buffer.

5.1 Results

The averaged results of the 3 runs for the best performing hyperparameter sets are shown in table 1a - 1c for the 3 different tasks. Additionally the best value for an agent without curiosity ($\alpha = 0$) and the best value for an agent using $r_{\text{intrinsic, alternative}}$ are shown. The agents with curiosity perform the best, moreover, the best-performing agents always have the highest α from all tested hyperparameter sets. For the pLogP task, only two agents, both of which use curiosity, have found the sulfur chain. This indicates, that curiosity indeed can help to escape local optima.
The alternative formulation of the intrinsic reward seems to help for the similarity task but not on the QED task and does not help to find the sulfur chain.
The version where we optimize the predictor network after every step of the agent consistently performed better than the version where we used a buffer, which is probably due to the fact, that we trained the predictor only two times during the agents life time.

6 Conclusion

In this work, we develop *curious agents* in the domain of molecular design, and show that they outperform their lesser curious competitors in three distinct molecular design tasks. Our results point towards a new, efficient RL-based exploration strategy for identifying new high-performance molecules and compounds.

In order to apply this technique to practical applications that require molecules that are one order of magnitude larger, we need to better understand how this technique scales with the size of the chemical space. While it seems clear that stronger exploration techniques are advantageous, this intuition actually needs to be confirmed in computational experiments.

References

[1] Benjamin Sanchez-Lengeling and Alán Aspuru-Guzik. Inverse molecular design using machine learning: Generative models for matter engineering. *Science*, 361(6400):360–365, 2018.

[2] Piotr S Gromski, Alon B Henson, Jarosław M Granda, and Leroy Cronin. How to explore chemical space using algorithms and automation. *Nature Reviews Chemistry*, 3(2):119–128, 2019.

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

[4] 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.

[5] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction tree variational autoencoder for molecular graph generation. *arXiv preprint arXiv:1802.04364*, 2018.

[6] 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.

[7] AkshatKumar Nigam, Pascal Friederich, Mario Krenn, and Alán Aspuru-Guzik. Augmenting genetic algorithms with deep neural networks for exploring the chemical space. *arXiv preprint arXiv:1909.11655*, 2019.

[8] Jan H Jensen. A graph-based genetic algorithm and generative model/monte carlo tree search for the exploration of chemical space. *Chemical science*, 10(12):3567–3572, 2019.

[9] Emilie S Henault, Maria H Rasmussen, and Jan H Jensen. Chemical space exploration: how genetic algorithms find the needle in the haystack. *PeerJ Physical Chemistry*, 2:e11, 2020.

[10] Théophile Gaudin, AkshatKumar Nigam, and Alan Aspuru-Guzik. Exploring the chemical space without bias: data-free molecule generation with dqn and selfies.

[11] Esben Jannik Bjerrum and Richard Threlfall. Molecular generation with recurrent neural networks (rnns). *arXiv preprint arXiv:1705.04612*, 2017.

[12] MHS Segler, T Kogej, C Tyrchan, and MP Waller. Generating focused molecule libraries for drug discovery with recurrent neural networks. acs cent sci 4 (1): 120–131. *arXiv preprint arXiv:1701.0132*, 9, 2018.

[13] P Ertl, R Lewis, E Martin, and V Polyakov. In silico generation of novel, drug-like chemical matter using the lstm neural network. arxiv e-prints. *arXiv preprint arXiv:1712.07449*, 2017.

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

[15] Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 16–17, 2017.

[16] Yuri Burda, Harri Edwards, Deepak Pathak, Amos Storkey, Trevor Darrell, and Alexei A Efros. Large-scale study of curiosity-driven learning. *arXiv preprint arXiv:1808.04355*, 2018.

[17] Arthur Aubret, Laetitia Matignon, and Salima Hassas. A survey on intrinsic motivation in reinforcement learning. *arXiv preprint arXiv:1908.06976*, 2019.

[18] Jürgen Schmidhuber. Formal theory of creativity, fun, and intrinsic motivation (1990–2010). *IEEE Transactions on Autonomous Mental Development*, 2(3):230–247, 2010.

[19] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[20] Mario Krenn, Florian Hase, AkshatKumar Nigam, Pascal Friederich, and Alan Aspuru-Guzik. Self-referencing embedded strings (selfies): A 100% robust molecular string representation. *Machine Learning: Science and Technology*, 2020.