Augmenting Genetic Algorithms with Neural Networks

AkshatKumar Nigam[1,†], **Pascal Friederich**[1,2,3,†]
Mario Krenn[1,2,4], **Alán Aspuru-Guzik**[1,2,4,5,*]

[1]Department of Computer Science, University of Toronto, Canada.
[2]Department of Chemistry, University of Toronto, Canada.
[3]Institute of Nanotechnology, Karlsruhe Institute of Technology, Germany.
[4]Vector Institute for Artificial Intelligence, Toronto, Canada.
[5]Lebovic Fellow, Canadian Institute for Advanced Research (CIFAR), Toronto, Canada
[†]These authors contributed equally
[*]Correspondence to: alan@aspuru.com

1 Introduction

Design of optimal structures under constraints is an important problem spanning multiple domains in the physical sciences. Specifically, in chemistry the design of tailor-made organic materials and molecules requires efficient methods to explore the chemical space. Purely experimental approaches are often time consuming and expensive. Reliable computational tools can accelerate and guide experimental efforts to find new materials faster.

We present a genetic algorithm (GA) [1, 2] for molecular design that is enhanced with two features:

1. A neural network based adaptive penalty [3]. This promotes exploratory behaviour of the GA and thus improve the diversity of generated molecules.
2. Exploiting the robustness of SELF-referencIng Embedded Strings (SELFIES [4]), we do not need to incorporate any expert based mutation or cross-over rules.

Starting from only simple methane molecules, our algorithm outperforms other generative models in optimization tasks for molecular design. By introducing machine learning (ML) techniques, the long term behaviour is not subject to stagnation, thus solving a major problem in genetic algorithms [5].

2 Related Works

Inverse design of molecules has been tackled as an optimization problem, mainly in the form of GA and ML based Generative Models. Variational autoencoders (VAEs) [6] are a widely used method for direct generation of molecular string or graph representations [7]. They encode discrete representations into a continuous (latent) space. Molecules resembling a known structure can be found by searching around the region of the encoded point. Making using of the continuous latent representation, it is possible to search via gradients or Bayesian Optimization. However, the generation of semantically and syntactically valid molecules is a challenging task.

Thus, several follow up works to the VAE have been proposed for inverse design in chemistry. Among them, CVAE [7], GVAE [8] and SD-VAE [9] work directly on string molecular representations. Alternatively, JT-VAE [10] works on molecular graphs. Unlike latent space property optimization, GCPN [11] proposes a reinforcement learning (RL)-based method for direct optimization on molecular graphs. ORGAN [12] demonstrate training string-based generative adversarial networks (GANs) [13] via RL.

Second Workshop on Machine Learning and the Physical Sciences (NeurIPS 2019), Vancouver, Canada.

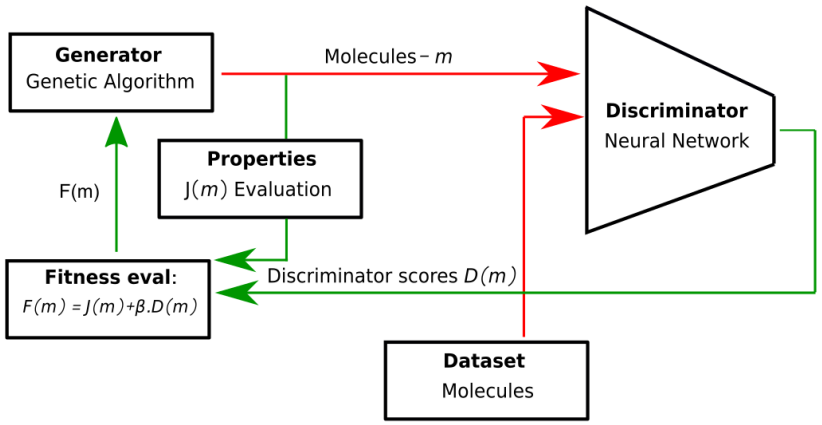

Figure 1: Overview of our hybrid structure, which augments genetic algorithms with ML based neural networks.

In all the approaches mentioned above, generative models are trained to mimic the reference dataset distributions, thus limiting the explorational ability of VAEs and GANs.

There exist several examples of GA based molecular optimization algorithms in literature [14, 15, 16, 17]) as well as examples of combining evolutionary strategies with ML[18]. While some of these examples pre-define mutations on a Simplified Molecular Input Line Entry Specification (SMILES [19]) level to ensure validity of the molecules, other approaches use fragment based assembly of molecules. GAs are likely to get trapped in regions of local optima [5]. Thus, for selection of the best molecules, [14, 17] report multiple restarts upon stagnation.

We include aforementioned models as baselines for numerical comparison.

3 GA-D Architecture

3.1 Overview

Our approach is illustrated in Figure 1. Our **generator** is a genetic algorithm with a population of molecules m. In each generation, the fitness of all molecules is evaluated as a linear combination of molecular **properties** $J(m)$ and the discriminator score $D(m)$:

$$F(m) = J(m) + \beta \cdot D(m) \ . \tag{1}$$

Random mutations of high **fitness** (best performing) molecules replace inferior members, while best performing molecules continue to a subsequent generation. The probability of replacing a molecule is evaluated using a smooth logistic function based on a ranking of fitness among the molecules of a generation. At the end of each generation, a neural network based **discriminator** is trained jointly on molecules generated by the GA and a reference **dataset**. The fitness evaluation accounts for the discriminator predictions for each molecule. Therefore, the discriminator plays a role in the selection of the subsequent population.

3.2 Mutation rules

Mutation of molecules to populate subsequent generations is an important element of the GA. A low degree of mutation can lead to a slow exploration of the chemical space, causing stagnation of the fitness function. Robustness of SELFIES allows us to do random mutations to molecular strings, while preserving their validity. Thus, our mutations only include 50% insertions or 50% replacements of single SELFIES characters. To accelerate the exploration of the GA, we add one domain specific mutation rule – the direct addition of phenyl groups in approximately 4% of cases. Character deletion is implicitly taken into account in SELFIES mutations and we do not use cross-over rules.

3.3 Role of the Discriminator

The fundamental role of the discriminator is to increase molecular diversity by removing similar, long-surviving molecules. Consider the realistic scenario in which a GA has found a molecule close to the optimal properties. As a result, this molecule survives for multiple generations. During such periods, the GA has limited view of the chemical space as it repeatedly explores mutations of the same high-fitness molecule.

A straight-forward solution would be the addition of a linear penalty in the fitness function that accounts for the number of successive generations a molecule survives. However, this method assigns independent scores to similar-looking molecules, which again results in less variety.

Our solution is the addition of an adaptive function like a neural network discriminator, thus reducing this problem. Similar molecules receive similar classification score. Furthermore, long surviving molecules are trained longer and receive weaker scores, resulting in decreasing fitness - reducing the chance of long periods of stagnation (illustrated in Figure 2).

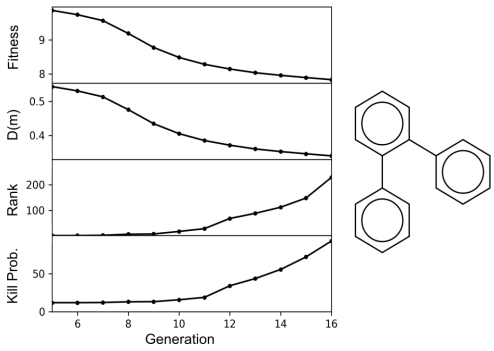

Figure 2: Reducing fitness of a molecule, initially possessing the largest fitness in generation 5. Due to decreasing discriminator predictions $D(m)$, the fitness decreases leading to an increased probability of getting replaced. The molecule is completely replaced beyond generation 16. The rank denotes the position of the molecule ordered by their fitness.

4 Experiments

Comparing to the literature standard, we consider maximizing the penalized logP objective $J(m)$ proposed by [7]. For molecule m, penalized logP is defined as:

$$J(m) = \text{logP}(m) - \text{SA}(m) - \text{RingPenalty}(m) \,, \tag{2}$$

logP indicates the the water-octanol partition coefficient. SA [20] represents the synthetic accessibility and prevents the formation of chemically unfeasible molecules. RingPenalty linearly penalizes the presence of rings of size larger than 6. Our reference dataset consists of 250,000 commercially available molecules extracted from the ZINC database [21]. All three quantities in the above equation are normalized based on this dataset. We note that it is of limited practical relevance to increase the penalized logP values to very high values, however, it is an accepted standard for comparison. Section 4.3 shows experiments to find molecules with given target properties, which is a more realistic setting for many practical applications.

4.1 Unconstrained optimization and comparison to other generative models

We define our fitness function according to Eq. 1 and 2 with $\beta = 0$ and 10. The algorithm is run for 100 generations with a population size of 500. All generated molecules are constrained to a canonical smile length of 81 characters (as in [22, 14]). We train the discriminator (fully connected neural network with relu activation and sigmoid output layer, the input is a vector of chemical and geometrical properties characterizing the molecules) at the end of each generation for 10 epochs on 500 molecules proposed by the GA and 500 randomly drawn molecules from the Zinc dataset. The molecular encoding provided to the discriminator consists of 50 properties calculated using RDKit [23]. Specifically, we provide the discriminator atom, bond, and ring information (ratio of non-carbon to carbon atoms, number of single, double and triple bonds, number of rings of sizes 1 to 20, etc.). We report maximum $J(m)$ scores, averaged over 10 independent runs. The highest $J(m)$ achieved by our approach are 13.31 ± 0.63 ($\beta = 10$) and 12.61 ± 0.81 ($\beta = 0$), respectively, which is almost twice as high es the highest literature value of 7.87 ± 0.07 (See Table 1). We observe that a higher β value leads to similar results. Furthermore, we compare to 50,000 random valid SELFIES strings, which surprisingly outperforms some existing generative models. The GA-D(t) results are explained in the next section.

Table 1: Comparison of our model with maximum penalized logP scores reported in literature.

Model	Max. Penalized logP
GVAE + BO [8] [1]	2.87 ± 0.06
SD-VAE [9] [1]	3.50 ± 0.44
CVAE + BO [7] [2]	4.85 ± 0.17
ORGAN [12] [1]	3.52 ± 0.08
JT-VAE [10] [1]	4.90 ± 0.33
ChemTS [22]	5.6 ± 0.5
Random SELFIES	6.19 ± 0.63
GCPN [11] [1]	7.87 ± 0.07
GB-GA [14] [3]	7.4 ± 0.9
GB-GA [14] [4]	$\mathbf{15.76 \pm 5.71}$
GA (here)	12.61 ± 0.81
GA + D (here)	13.31 ± 0.63
(**GA + D(t)** (here) [5]	$\mathbf{20.72 \pm 3.14}$)

[1] average of three best molecules after performing Bayesian optimization on the latent representation of a trained VAE
[2] two best molecules of a single run
[3] averaged over 10 runs with 20 molecules per generation with a molecular weight (excl. hydrogen) smaller than 39.15 ± 3.50 g/mol run for 50 generations
[4] averaged over 10 runs with 500 molecules up to 81 characters per generation and 100 generations
[5] averaged over 5 runs with 1000 generations each

4.2 Long term experiment with a time-dependent adaptive penalty

In Figure 3 we show the results of runs where we use a time dependent adaptive penalty. During periods of saturation (no higher $J(m)$ score for 5 generations, earliest in generation 100), the weight of the discriminator predictions is switched from 0 to 1000 until stagnation is overcome. The genetic algorithm is hence forced to propose new families of molecules to increase $J(m)$. As it can be observed in Figure 3, even after steep decreases in $\max J(m)$, the scores recover and potentially reach values higher than in previous plateaus. We observe that this approach significantly outperforms all previous methods in maximizing the objective $J(m)$.

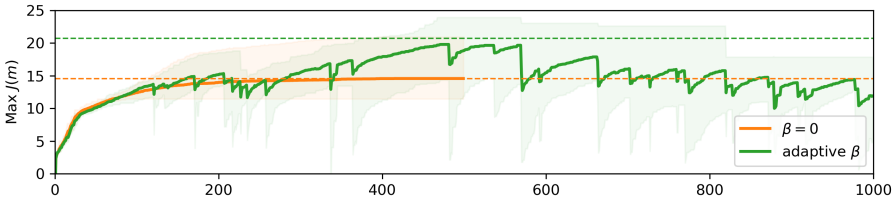

Figure 3: Maximum $J(m)$ values found for 10 independent runs with no discriminator ($\beta = 0$) and 5 independent runs with the introduction of a time-dependent adaptive penalty. The full line is the average of all runs, the shaded areas are the boundaries of all runs and the dashed lines denote the average of all maximal $J(m)$ values found at any generation.

4.3 Constrained Optimization

In the previous sections, our aim was to maximize the penalized logP objective. Here, we consider a different task: generating molecules of specific chemical interest. We run 250 instances of randomly selected sets of target properties (logP in the interval from -5 to 10, SA scores in the interval from 1 to 5 and ring penalty for 0 to 3). In each run, we have a generation size of 500 molecules initialized with methane. The fitness function is modified to minimize the summed squared difference between the actual and desired properties. We compute the number of experiments that successfully proposed molecules with a squared difference less than 1.0 (within 100 generations). Each run is constrained

to run for 100 generations with a maximum canonical SMILES length of 81 characters. In 98.0% of the cases, our approach proposes the desired molecules.

5 Conclusions

We presented a hybrid GA and ML-based generative model and demonstrated its application in chemical design. The model outperforms literature approaches in generating molecules with desired properties. For computationally more expensive property evaluations, we will extend our appraoch by the introduction of an on-the-fly trained ML property evaluation method, which will open new ways of solving the inverse design challenge in chemistry and materials sciences.

Our approach is independent of domain knowledge (except the phenyl mutation), thus applicable to design questions in other scientific disciplines beyond chemistry. An important future expansion will show that the GA-D is a general concept of a new generative model. On top of the high performance in design problems, the influence of the discriminator can be tuned using the continuous parameter β. This allows to adjust similarity with a given dataset and diversity of the generated samples.

Acknowledgement

P.F. acknowledges funding the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement no. 795206 (MolDesign). M.K. acknowledges support from the Austrian Science Fund (FWF) through the Erwin Schrödinger fellowship No. J4309. A.A.-G. thanks Anders G. Frøseth for his generous support. A.A.-G. acknowledges the generous support of Natural Resources Canada and the Canada 150 Research Chairs program.

References

[1] John Henry Holland et al. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992.

[2] David E Goldberg. *Genetic algorithms*. Pearson Education India, 2006.

[3] Pruettha Nanakorn and Konlakarn Meesomklin. An adaptive penalty function in genetic algorithms for structural design optimization. *Computers & Structures*, 79(29-30):2527–2539, 2001.

[4] Mario Krenn, Florian Häse, AkshatKumar Nigam, Pascal Friederich, and Alán Aspuru-Guzik. Selfies: a robust representation of semantically constrained graphs with an example application in chemistry. *arXiv preprint arXiv:1905.13741*, 2019.

[5] W Paszkowicz. Properties of a genetic algorithm equipped with a dynamic penalty function. *Computational Materials Science*, 45(1):77–83, 2009.

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

[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, Jan 2018.

[8] Matt J Kusner, Brooks Paige, and José Miguel Hernández-Lobato. Grammar variational autoencoder. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1945–1954. JMLR. org, 2017.

[9] Hanjun Dai, Yingtao Tian, Bo Dai, Steven Skiena, and Le Song. Syntax-directed variational autoencoder for molecule generation. In *Proceedings of the International Conference on Learning Representations*, 2018.

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

[11] Jiaxuan You, Bowen Liu, Zhitao Ying, Vijay Pande, and Jure Leskovec. Graph convolutional policy network for goal-directed molecular graph generation. In *Advances in Neural Information Processing Systems*, pages 6410–6421, 2018.

[12] 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.

[13] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[14] 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.

[15] Noel M O'Boyle, Casey M Campbell, and Geoffrey R Hutchison. Computational design and selection of optimal organic photovoltaic materials. *The Journal of Physical Chemistry C*, 115(32):16200–16210, 2011.

[16] Aaron M Virshup, Julia Contreras-García, Peter Wipf, Weitao Yang, and David N Beratan. Stochastic voyages into uncharted chemical space produce a representative library of all possible drug-like compounds. *Journal of the American Chemical Society*, 135(19):7296–7303, 2013.

[17] Chetan Rupakheti, Aaron Virshup, Weitao Yang, and David N Beratan. Strategy to discover diverse optimal molecules in the small molecule universe. *Journal of chemical information and modeling*, 55(3):529–537, 2015.

[18] Tim Salimans, Jonathan Ho, Xi Chen, Szymon Sidor, and Ilya Sutskever. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017.

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

[20] Peter Ertl and Ansgar Schuffenhauer. Estimation of synthetic accessibility score of drug-like molecules based on molecular complexity and fragment contributions. *Journal of cheminformatics*, 1(1):8, 2009.

[21] John J Irwin, Teague Sterling, Michael M Mysinger, Erin S Bolstad, and Ryan G Coleman. Zinc: a free tool to discover chemistry for biology. *Journal of chemical information and modeling*, 52(7):1757–1768, 2012.

[22] Xiufeng Yang, Jinzhe Zhang, Kazuki Yoshizoe, Kei Terayama, and Koji Tsuda. Chemts: an efficient python library for de novo molecular generation. *Science and technology of advanced materials*, 18(1):972–976, 2017.

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