Learning to Optimize Molecules with a Chemical Language Model

Jerret Ross IBM Research
Samuel C Hoffman IBM Research
Brian Belgodere IBM Research
Vijil Chenthamarakshan IBM Research
Youssef Mroueh IBM Research
Payel Das IBM Research

Abstract

Transformer-based chemical language models (CLM), trained on large and general purpose datasets consisting of molecular strings, have recently emerged as a powerful tool for successfully modeling various structure-property relations, as well as for proposing novel candidates. In this work, we propose a novel approach that harnesses a recent generative CLM, namely GP-MOLFORMER, to propose small molecules with more desirable properties. Specifically, we present a parameter-efficient fine-tuning method for the unconstrained property optimization, which uses property-ordered molecular pairs as input. We call this new approach *pair-tuning*. Our results show GP-MOLFORMER outperforms existing baselines in terms of generating *diverse* molecules with desired properties across three popular property optimization tasks, namely druglikeliness, penalized logP, and dopamine type 2 receptor activity. Results demonstrate the general utility of *pair-tuning* together with a generative CLM for a variety of molecular optimization tasks.

1 Introduction

Identifying molecules with desirable properties from the vast landscape of possibilities is daunting. It has been postulated that only 10^8 out of 10^{23}–10^{60} possible small drug-like molecules are synthesizable [1]. As the search space is enormous and high-throughput screening of molecules is costly and time-consuming, this requires a thorough understanding of the chemical data manifold. Recently, deep generative models have made great strides in modeling molecular distributions and sampling new molecules from them in de novo or targeted manners.

In this work, we present a family of generative pre-trained molecular foundation models for the targeted generation of novel molecules, which are based on the recently published Molecular Language transFormer (MOLFORMER) architecture [2]. We refer to these Generative Pre-trained models as GP-MOLFORMER. The base transformer architecture of our decoder-only GP-MOLFORMER, consisting of ≈ 47M parameters, uses an efficient linear attention mechanism, together with rotary positional encodings, analogously to MOLFORMER [2]. The model is then trained with a causal language modeling objective on a large corpus of 1.1 billion canonicalized SMILES strings of small molecules from publicly available chemical databases.

We further extend GP-MOLFORMER to two targeted molecular design tasks: scaffold-constrained molecular decoration and unconstrained property-guided optimization. For the first, we exploit GP-MOLFORMER's causal language modeling ability and establish GP-MOLFORMER's ability to handle the task without undergoing any task-specific tuning. For the second, we provide a prompt-tuning or soft prompt-learning algorithm that learns from partial orderings of molecules. We name this method *pair-tuning*. Results show that pair-tuning on GP-MOLFORMER provides on par or better

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

Table 1: Scaffold-constrained generation. Predicted active hits is the percentage of generated molecules that pass the DRD2 binding classifier. Baseline performance is taken from Arús-Pous et al. (2020) [5]. **Bold** value indicates the best performing model.

	predicted active hits (%)
Scaffold decorator[5]	3.64
de novo GP-MOLFORMER	0.83
scaffold-conditioned GP-MOLFORMER	**4.58**

performance in three different property optimization tasks, namely (i) drug-likeness optimization, (ii) penalized logP optimization, and (iii) optimization of dopamine type 2 receptor binding activity.

2 Methods

2.1 GP-MOLFORMER Framework

We construct the GP-MOLFORMER architecture using a 12-layer linear attention transformer with 12 attention heads and a hidden state size of 768 and generalized random Fourier feature maps [3] for the linear attention. The model is trained on the SMILES representation of 1.1 billion molecules from PubChem and ZINC databses. We choose a relative positional embedding, namely rotary positional embedding [4], to model the token dependency within SMILES better using rotary attention. During training, GP-MOLFORMER uses a causal modeling objective of predicting the next token given the context history of prior tokens in the input SMILES strings. Given the size of the model and the efficient linear attention, GP-MOLFORMER takes only around 3 milliseconds for a single forward pass during generation, using a single A100 GPU. More details on the architecture, datasets and training are in Appendex B.

2.2 Scaffold-constrained decoration

For this experiment, we first take the five unique scaffolds from the dopamine receptor D_2 (DRD2) active binder dataset validation split [5]. These scaffolds contain between two to four attachment points. We perform a pre-processing step for each unique scaffold, generating every possible randomized SMILES representation of that scaffold. Then we sort the resulting candidates according to the distance of the "*" characters, also known as the attachment point, to the end of the string. For multiple *s, we sum the distances. As an example, `C1(=O)N(CCN1*)*` would score $2(2+0)$ while `C1N(*)C(=O)N(*)C1` would score $15(12+3)$. If multiple representations are equivalently optimal, we save all of them. During the generation step, we provide the candidates produced in this pre-processing step as input to GP-MOLFORMER.

Next, the task is to generate multiple possible candidates for the first attachment point given an input scaffold. First, we collect all valid candidates from that generation. Then, we again generate multiple possible candidates for the recently extended scaffolds. This process is repeated until all the attachment points are decorated then we collect all valid molecules generated.

We compare the performance of GP-MOLFORMER in terms of generating DRD2 active molecules that will pass the DRD2 binding classifier ($p > 0.5$). For baselines of comparison, we consider our own random generations from GP-MOLFORMER-RAW, as well as an earlier scaffold-conditioned generation model [5] that was specifically trained for scaffold decoration tasks and was then used to decorate the same scaffolds under investigation here with fragments from ChEMBL. In contrast to this baseline model, GP-MOLFORMER-RAW has not seen scaffold-constrained generation during pre-training, nor is it specifically fine-tuned for this purpose. Table 1 shows that GP-MOLFORMER generates more DRD2 active hits compared to a random baseline of de novo generation, as well as a generative model trained on this specific task.

2.3 Unconstrained Single Property Optimization

Given GP-MOLFORMER demonstrates desirable performance in scaffold-constrained molecular decoration, it makes sense to extend GP-MOLFORMER to downstream task settings, where the goal is

Algorithm 1 Pair-tuning training

Require: Pretrained GP-MOLFORMER; Number of fine-tuning epochs, m; number of enhancement tokens, n; Pair-Tuning dataset $D = \{(a, b)|b > a\}$ for $a, b \in \Omega$ where $>$ is an order relation defined for a certain property T

1: Append $n + 1$ new tokens $\phi_T = \langle enh_1\rangle\langle enh_2\rangle \dots \langle enh_n\rangle, \langle sep\rangle$ to GP-MOLFORMER-RAW vocabulary
2: **for** m epochs **do**
3: Prepare training prompts as $\langle enh_1\rangle\langle enh_2\rangle \dots \langle enh_n\rangle\langle bos\rangle\langle a_1\rangle\langle a_2\rangle \dots \langle a_i\rangle\langle sep\rangle$ where $a = \langle a_1\rangle\langle a_2\rangle \dots \langle a_i\rangle$, i.e., tokenized training molecules
4: Compute the cross-entropy (CE) loss conditioned on enhancement tokens ϕ_T and molecule a with the auto-regressive CE loss with target the molecule $b = \langle b_1\rangle\langle b_2\rangle \dots \langle b_n\rangle\langle eos\rangle$
5: Compute the gradient of the auto-regressive CE loss with respect to enhancement tokens ϕ_T
6: Update enhancement tokens via gradient descent optimizer
7: **end for**

to generate molecules with a desired property. In light of current LLM adaptation efforts, one obvious path is model tuning (or "fine-tuning"), where all model parameters are tuned during adaptation. This approach often is highly data-hungry. As an alternative, prompt-tuning or "soft prompt" learning has been proposed, which includes an additional n tuneable tokens for each downstream task, which is prepended to the input text [6]. This soft prompt is then trained end-to-end on a labeled dataset, whereas the pre-trained LLM remains frozen. This method has been demonstrated to close the gap in model tuning, even when combined with a smaller LLM [6] and is lower cost compared to full fine-tuning. We exploit prompt-tuning to introduce a novel means for enabling GP-MOLFORMER to tackle property-specific molecular optimization tasks. We call this method pair-tuning and show that it performs well on a set of three tasks.

Pair-tuning framework Following a "text-to-text" approach, we formalize the task of generating a (property-)optimal molecule as follows: Given a molecule a, translate it to another molecule b with a more optimal property value where a, b come from domain Ω. This conditional generation task is $P_\theta(b|a)$, where θ is the parametrization of the generative language model. This task is handled via learning soft prompts, i.e., prompt-tuning, which is a parameter-efficient task adaptation method for a frozen language model [6]. Specifically, we add a small number of task-specific parameters ϕ_T, such that the conditional task becomes $P_\theta(b|\phi_T, a)$ and is trained through maximizing the probability likelihood of b. Only ϕ_T is updated during gradient backpropagation. This procedure is explained in Algorithm 1.

In this formulation, we do not need absolute property values of the molecules, rather only ordered pairs of molecules are needed. This is to mimic the scenario of many drug and material development tasks, in which two molecules are compared with each other to guide molecular optimization and prioritization, especially for tasks with limited available data. For example, Matched Molecular Pair (MMP) analysis allows the rapid estimation of property differences [7, 8]. However, MMP analysis is limited to comparing close molecular derivatives, and it can fail to model important chemical contexts. The present formulation of optimizing molecules is free from such constraints and only aims to learn task-specific soft prompts to generate more optimal molecules given a seed molecule.

Pairtuning tasks We evaluate pairtuning using GP-MOLFORMER on three property optimization benchmarks, namely drug-likeness (QED) maximization, penalized logP maximization, and activity maximization for DRD2. The first two properties, QED and penalized logP, are important considerations in drug discovery, and these task shows the ability of a model to optimize salient aspects of a molecule, even if maximization of these properties by themselves is of low utility [9]. More details about these tasks are in Appendix C.

Penalized logP optimization Table 2 shows results of pair-tuning on GP-MOLFORMER, as well as of the baselines, in terms of the generated molecules with high penalized logP. We report pair-tuning performances as a function of two different k, where k is the number of targeted generation attempts per molecule. For $k = 125$, using a test set containing 800 molecules gives a total number of generated molecules of 100k, which is the same used for the baselines. The baselines under consideration are JT-VAE [11], GCPN [12], MolDQN [9], MARS [13], GraphDF [14], and LIMO [10]. Penalized logP can be artificially inflated simply by generating molecules with increased length,

Table 2: Performance on unconstrained penalized logP and QED optimization. Baseline performances are taken from Zhou et al. (2019)[9] and Eckmann et al. (2022)[10]and are reported on 100k generations as per LIMO[10]. For GP-MOLFORMER, we set k, the number of targeted generation attempts per molecule, to 125 — given a test set of size 800 this results in 100k total generations. Values in parentheses are after post hoc length filtering. **Bold** values indicate the highest property values found (both length-constrained and unlimited).

	Penalized $\log \mathrm{P}$			QED		
	1 st	2nd	3rd	1st	2nd	3rd
JT-VAE	5.30	4.93	4.49	0.925	0.911	0.910
MARS	**45.0**	**44.3**	**43.8**	**0.948**	**0.948**	**0.948**
GRAPHDF	13.7	13.2	13.2	**0.948**	**0.948**	**0.948**
LIMO on z	6.52	6.38	5.59	0.910	0.909	0.892
LIMO	10.5	9.69	9.60	0.947	0.946	0.945
GCPN	7.98	7.85	7.80	**0.948**	0.947	0.946
MolDQN-bootstrap	**11.84**	**11.84**	**11.82**	**0.948**	0.944	0.943
Pair-tuning ($k = 125$)	13.18 (7.12)	12.24 (6.61)	11.51 (6.40)	**0.948**	0.947	0.947
Pair-tuning ($k = 1000$)	19.59 (9.35)	15.51(8.93)	15.27 (8.64)	**0.948**	**0.948**	**0.948**

specifically by adding alkyl carbons [13, 10]. Many works, e.g., GCPN, MolDQN, and LIMO, avoid this by reporting top property scores given length constraints, e.g., limiting the length up to the maximum molecule size of the ZINC250k dataset [15]. MARS, on the other hand, does not consider such a length constraint. We also report the top 3 scores for pair-tuning with a length constraint (length < 38), added post generation, in Table 2 as the value within parentheses. Compared to the strongest length-constrained baselines, pair-tuning generates molecules with comparably high values. When the length constraint is not considered, pair-tuning still generates molecules with higher but reasonable penalized logP values. Note that pair-tuning does not require feedback or evaluation on generations from an additional reward model or a property predictor, nor is the generative model updated during the tuning. We also report top 3 scores for 1M generations ($k = 1000$), which requires less than an hour to generate. Altogether, Table 2 shows that the proposed method can generate molecules with even higher penalized logP values, both with and without a length constraint.

QED optimization As with penalized logP, we show results for QED optimization in Table 2 compared with the same baselines. Again, pair-tuning performances are reported for two different values of k, showing comparable performances with respect to baselines. Supplementary Tables 4 and 5 further demonstrate that pairtuning with GP-MOLFORMER produces higher scoring molecules that also share high diversity as well as high closeness to training distribution, compared to baselines.

DRD2 activity optimization DRD2 activity optimization results are reported in supplementary Table 3. Activity scores are calculated using the trained predictor from Olivecrona et al. (2017) [16]. Average activity scores of the input seed molecules are also shown. Different baseline performances are reported using different test seed molecules, and we are interested in comparing the activity improvement of an experiment with the set of test seed molecules. For pair-tuning, performance reported considers $k = 20$ generations per seed molecule and we use the top 1 for each, similar to the baselines, Mol-CycleGAN [17] — a graph-based generation method that uses a CycleGAN optimization scheme — and Gargoyles [18], which uses Monte Carlo Tree Search to optimize molecular graphs based on an evaluation function. Results show that pair-tuning generates molecules with the highest activity improvement with respect to the seed molecules, when compared to baselines.

Conclusion

In this work, we propose a parameter efficient fine tuning method on top a chemical language model for scaffold constrained molecule generation and unconstrained molecular property optimization. Unlike other methods, our approach does not require a trained property predictor or absolute property values and can learn from pairs of sub-optimal and optimal molecules. We show that our method can generate diverse molecules with desired properties on three property optimization benchmarks. In

future work, we would like to apply our method to a variety of tasks including the design of easily synthesizable molecules that satisfy multiple properties.

References

[1] Sunghwan Kim, Paul A Thiessen, Evan E Bolton, Jie Chen, Gang Fu, Asta Gindulyte, Lianyi Han, Jane He, Siqian He, Benjamin A Shoemaker, et al. Pubchem substance and compound databases. *Nucleic acids research*, 44(D1):D1202–D1213, 2016.

[2] Jerret Ross, Brian Belgodere, Vijil Chenthamarakshan, Inkit Padhi, Youssef Mroueh, and Payel Das. Large-scale chemical language representations capture molecular structure and properties. *Nature Machine Intelligence*, 4(12):1256–1264, 2022.

[3] Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *International Conference on Machine Learning*, pages 5156–5165. PMLR, 2020.

[4] Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021.

[5] Josep Arús-Pous, Atanas Patronov, Esben Jannik Bjerrum, Christian Tyrchan, Jean-Louis Reymond, Hongming Chen, and Ola Engkvist. Smiles-based deep generative scaffold decorator for de-novo drug design. *Journal of cheminformatics*, 12:1–18, 2020.

[6] Brian Lester, Rami Al-Rfou, and Noah Constant. The power of scale for parameter-efficient prompt tuning, 2021.

[7] Alexander G Dossetter, Edward J Griffen, and Andrew G Leach. Matched molecular pair analysis in drug discovery. *Drug Discovery Today*, 18(15-16):724–731, 2013.

[8] Ziyi Yang, Shaohua Shi, Li Fu, Aiping Lu, Tingjun Hou, and Dongsheng Cao. Matched molecular pair analysis in drug discovery: methods and recent applications. *Journal of Medicinal Chemistry*, 66(7):4361–4377, 2023.

[9] Zhenpeng Zhou, Steven Kearnes, Li Li, Richard N Zare, and Patrick Riley. Optimization of molecules via deep reinforcement learning. *Scientific Reports*, 9(1):10752, 2019.

[10] Peter Eckmann, Kunyang Sun, Bo Zhao, Mudong Feng, Michael K Gilson, and Rose Yu. Limo: Latent inceptionism for targeted molecule generation. *Proceedings of machine learning research*, 162:5777, 2022.

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

[12] Jiaxuan You, Bowen Liu, Zhitao Ying, Vijay Pande, and Jure Leskovec. Graph convolutional policy network for goal-directed molecular graph generation. In *NeurIPS*, pages 6410–6421, 2018.

[13] Yutong Xie, Chence Shi, Hao Zhou, Yuwei Yang, Weinan Zhang, Yong Yu, and Lei Li. Mars: Markov molecular sampling for multi-objective drug discovery. In *International Conference on Learning Representations*, 2021.

[14] Youzhi Luo, Keqiang Yan, and Shuiwang Ji. Graphdf: A discrete flow model for molecular graph generation. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning, ICML 2021, 18-24 July 2021, Virtual Event*, volume 139 of *Proceedings of Machine Learning Research*, pages 7192–7203. PMLR, 2021.

[15] 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.

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

[17] Łukasz Maziarka, Agnieszka Pocha, Jan Kaczmarczyk, Krzysztof Rataj, Tomasz Danel, and Michał Warchoł. Mol-cyclegan: a generative model for molecular optimization. *Journal of Cheminformatics*, 12(1):2, 2020.

[18] Daiki Erikawa, Nobuaki Yasuo, Takamasa Suzuki, Shogo Nakamura, and Masakazu Sekijima. Gargoyles: An open source graph-based molecular optimization method based on deep reinforcement learning. *ACS omega*, 8(40):37431–37441, 2023.

[19] Daniil Polykovskiy, Alexander Zhebrak, Benjamin Sanchez-Lengeling, Sergey Golovanov, Oktai Tatanov, Stanislav Belyaev, Rauf Kurbanov, Aleksey Artamonov, Vladimir Aladinskiy, Mark Veselov, Artur Kadurin, Sergey Nikolenko, Alan Aspuru-Guzik, and Alex Zhavoronkov. Molecular sets (MOSES): A benchmarking platform for molecular generation models. *arXiv:1811.12823*, 2018.

[20] Matt J Kusner, Brooks Paige, and José Miguel Hernández-Lobato. Grammar variational autoencoder. In *Proceedings of the 34th ICLR*, pages 1945–1954. JMLR.org, 2017.

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

[22] Linde Schoenmaker, Olivier JM Béquignon, Willem Jespers, and Gerard JP van Westen. Uncorrupt smiles: a novel approach to de novo design. *Journal of Cheminformatics*, 15(1):22, 2023.

[23] Noel O'Boyle and Andrew Dalke. Deepsmiles: an adaptation of smiles for use in machine-learning of chemical structures. *ChemArxiv preprint: chemrxiv.7097960.v1*, 2018.

[24] Mario Krenn, Florian Häse, AkshatKumar Nigam, Pascal Friederich, and Alan Aspuru-Guzik. Self-referencing embedded strings (SELFIES): A 100% robust molecular string representation. *Machine Learning: Science and Technology*, 1(4):045024, nov 2020.

[25] Austin H Cheng, Andy Cai, Santiago Miret, Gustavo Malkomes, Mariano Phielipp, and Alán Aspuru-Guzik. Group selfies: a robust fragment-based molecular string representation. *Digital Discovery*, 2(3):748–758, 2023.

[26] Michael A Skinnider. Invalid smiles are beneficial rather than detrimental to chemical language models. *Nature Machine Intelligence*, pages 1–12, 2024.

[27] Vijil Chenthamarakshan, Payel Das, Samuel Hoffman, Hendrik Strobelt, Inkit Padhi, Kar Wai Lim, Benjamin Hoover, Matteo Manica, Jannis Born, Teodoro Laino, et al. Cogmol: Target-specific and selective drug design for covid-19 using deep generative models. *Advances in Neural Information Processing Systems*, 33:4320–4332, 2020.

[28] Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Hierarchical generation of molecular graphs using structural motifs. In *International conference on machine learning*, pages 4839–4848. PMLR, 2020.

[29] Maksim Kuznetsov and Daniil Polykovskiy. Molgrow: A graph normalizing flow for hierarchical molecular generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 8226–8234, 2021.

[30] Tengfei Ma, Jie Chen, and Cao Xiao. Constrained generation of semantically valid graphs via regularizing variational autoencoders. *Advances in Neural Information Processing Systems*, 31, 2018.

[31] Nicola De Cao and Thomas Kipf. MolGAN: An implicit generative model for small molecular graphs. *arXiv:1805.11973*, 2018.

[32] Yair Schiff, Vijil Chenthamarakshan, Samuel C Hoffman, Karthikeyan Natesan Ramamurthy, and Payel Das. Augmenting molecular deep generative models with topological data analysis representations. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 3783–3787. IEEE, 2022.

[33] Alex Zhavoronkov, Yan A Ivanenkov, Alex Aliper, Mark S Veselov, Vladimir A. Aladinskiy, Anastasiya V. Aladinskaya, Victor A. . Terentiev, Daniil A. Polykovskiy, Maksim D. Kuznetsov, Arip Asadulaev, Yury Volkov, Artem Zholus, Rim R. Shayakhmetov, Alexander Zhebrak, Lidiya I. Minaeva, Bogdan A. Zagribelnyy, Lennart H. Lee, Richard Soll, David Madge, Li Xing, Guo Tao, and Alán Aspuru-Guzik. Deep learning enables rapid identification of potent DDR1 kinase inhibitors. *Nature Biotechnology*, 37:1038—1040, 2019.

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[35] Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A Shoemaker, Paul A Thiessen, Bo Yu, Leonid Zaslavsky, Jian Zhang, and Evan E Bolton. PubChem 2019 update: improved access to chemical data. *Nucleic Acids Research*, 2018.

[36] John J Irwin and Brian K Shoichet. ZINC–a free database of commercially available compounds for virtual screening. *Journal of Chemical Information and Modeling*, 45(1):177–182, 2005.

[37] Philippe Schwaller, Teodoro Laino, Théophile Gaudin, Peter Bolgar, Christopher A. Hunter, Costas Bekas, and Alpha A. Lee. Molecular transformer: A model for uncertainty-calibrated chemical reaction prediction. *ACS Central Science*, 5(9):1572–1583, 2019.

[38] RDKit: Open-source cheminformatics. `http://www.rdkit.org`, 2021. [Online; accessed 28-May-2021].

[39] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the NAACL: HLT, Vol 1*, June 2019.

[40] Wengong Jin, Kevin Yang, Regina Barzilay, and Tommi Jaakkola. Learning multimodal graph-to-graph translation for molecular optimization, 2019.

Appendix

A Related Work

Earlier work on de novo molecule generation mapped a SMILES representation of a molecule into a continuous latent space using character based recurrent neural networks [19] and variational autoencoders (VAE)[15]. The VAE models have an encoder that converts the discrete molecular representation of a molecule into a continuous representation and a decoder that converts the continuous representation back to a discrete representation. The continuous representation allows for efficient de novo generation of molecules by sampling from the latent space. A property predictor is often trained on the latent embeddings from the encoder and is used to search for compounds that have specific desirable properties (conditional generation) using gradient-based or Bayesian optimization techniques.

However, there are several drawbacks for this approach. One drawback is that the decoded SMILES string may not always represent a valid molecule, often violating chemical constraints like valency. To alleviate this problem, several approaches have been proposed to augment the model with rules or mechanisms to ensure that the decoded SMILES strings are always valid [11, 20, 21, 22]. Another solution is to use alternate more robust string representations like DeepSMILES [23] SELFIES[24] and Group SELFIES [25]. While SELFIES-based representation ensures that all the decoded molecules are structurally valid, there is recent work that indicates that invalid SMILES are low-likelihood samples and are beneficial for chemical language models to better model the input molecule distribution [26].

Another drawback of this approach is that the property predictors trained on the latent embeddings are not as accurate as predictors trained end-to-end on the molecule itself. This necessitates a post-processing step, where the generated molecules have to be filtered using another, more accurate, predictor [27]. LIMO [10] avoids this problem by freezing the encoder and using a property predictor trained directly on the discrete representation of a molecule to optimize the latent space using an inceptionism-like approach.

Along with string generation methods, several graph and topological generative models have also been proposed [14, 28, 29, 30, 31, 13, 12, 32]. Another approach to molecule optimization is to use reinforcement learning to generate or modify a molecular graph [12, 9, 33, 16, 14]. While these methods have been shown to be successful against a variety of molecular optimization tasks, they often require multiple calls to the property predictor during the optimization process and hence are difficult to use along with computationally expensive property predictors.

B Details of Datasets and Model Training

B.1 Model Details

The GP-MoLFormer decoder uses the transformer block used in MoLFormer. To avoid the quadratic complexity associated with regular attention computations in vanilla transformers [34], MoLFormer utilized a base 12-layer transformer architecture with linear attention [3], wherein each layer has 12 attention heads and a hidden state size of 768. A Generalized Feature map [3] for the linear attention was chosen. To better model positional dependence of tokens within a SMILES string, MoLFormer deviates from using the default absolute position embeddings and instead uses rotary embeddings [4]:

$$\text{Attention}_m(Q,K,V) = \frac{\sum_{n=1}^{N}\langle\varphi(R_m q_m),\varphi(R_n k_n)\rangle v_n}{\sum_{n=1}^{N}\langle\varphi(R_m q_m),\varphi(R_n k_n)\rangle},$$

where Q, K, V are the query, key, and value respectively, and φ a random feature map. The GP-MoLFormer is trained on the next token prediction task using a cross-entropy objective: $L_{LM}(w1,...,wn) = \sum_i \log P(w_i|w_{j<i})$.

B.2 Datasets and Tokenization

We used two datasets for pre-training by combining SMILES from the PubChem [35] and ZINC [36] databases with varying proportions from each. The dataset contains a total of 1.1B SMILES

strings; 111M of them are from the PubChem dataset, whereas the larger 1B portion comes from the ZINC database. We utilized the tokenizer from Schwaller et al. (2019) [37] to construct a vocabulary. All SMILES sequences from both PubChem and ZINC are converted to a canonical format with no isomeric information using RDKit [38] followed by tokenization. All unique tokens extracted from the resulting output give us a vocabulary of 2357 tokens plus 5 special tokens, resulting in a total of 2362 vocabulary tokens. The post tokenization sequence length of the molecules range from 1 to just over 2000 tokens. We decide to restrict the sequence length range from 1 token to 202 tokens, special tokens inclusive, to reduce computation time. Since over 99.4 percent of all molecules from our dataset contain less than 202 tokens, we hypothesize that the removal of molecules with more than 202 tokens would be of minimal negative impact on pre-training.

B.3 Large Scale Training and Parallelization

For pre-training, we use the causal language model objective defined in Devlin et al. (2019) [39]. The training was performed for 4 epochs through the entire combined PubChem+ZINC dataset with a fixed learning rate of 1.6×10^{-4} and a batch size of 1600 molecules per GPU on a total of 16 A100 80 GB GPUs over 2 servers connected via EDR Infiniband fabric. It should be noted that as the number of GPUs utilized increased, we found an increase in learning rate was necessary up to a factor of 8.

In order to scale our training to large datasets (>1B data points), we relied on adaptive bucketing of mini-batches by sequence length, as well as parallelization via distributed data-parallel training.The combination of linear attention, bucketing, and data parallelism allowed us to reduce the number of GPUs needed from roughly 1000 for quadratic attention with no bucketing to 16.

C Pair-tuning

The paired molecule datasets for pair-tuning experiments were taken from Jin et al. (2019) [40]; The QED paired data used for training consists of 70644 molecule pairs where the first/seed molecule has a QED value in the range of 0.7–0.8 while the second/target molecule has a QED of 0.9–1.0. The penalized logP paired data consists of 60227 molecule pairs. It should be noted that while the paired datasets were collected such that molecular similarity within the pair is 0.4 and 0.6 for QED and logP, respectively, we demonstrate pair-tuning only on unconstrained property optimization tasks — we do not account for similarity preservation. The test set size for both QED and penalized logP optimization was 800. For the DRD2 binding optimization task, we used 34,404 molecule pairs from ZINC and Olivecrona et al. (2017) [16] for training and a test set of 1000 molecules [40]. For scoring the generated molecules, the bioactivity prediction model from Olivecrona et al. (2017) [16] is used; inactive compounds were defined with $p < 0.5$ and actives were with $p \geq 0.5$.

The vocabulary includes 20 randomly initialized prompt embeddings as well as the <unk> embedding from GP-MoLFormer pre-training. For pair-tuning training, we prepended all 20 prompt embeddings to the <bos> embedding, followed by the embeddings of the first/seed molecule in a specific pair. We then add the <unk> embedding at the end of the first/seed molecule. After the <unk> embedding, we add the embeddings of the target molecule, followed by the <eos> embedding.

For evaluation, we do a forward pass using the following sequence: the first 20 prompt embeddings + the <bos> embedding + the input molecule embeddings + the <unk> embedding. After that, we sample from the token distribution generated by GP-MoLFormer until <eos> is encountered. For all pair-tuning experiments, batch size was set to 35, the learning rate was fixed at 3×10^{-2}, and the number of epochs run was 1000. Each epoch took 6 minutes to complete on a single GPU. During pairtuning generation, GP-MoLFormer takes 0.00375 seconds for a single forward pass.

D DRD2 Activity Optimization

The performance of pair-tuning on DRD2 activity optimization task is given below. We show that Pair-tuning achieves the best predicted activity score when compared to the baselines.

Table 3: Performance on the unconstrained DRD2 activity optimization task with respect to the initial value. Baseline performances are reported from Erikawa et al. (2023)[18]. **Bold** value indicates the best performing model.

	pred. activity score	avg. seed score
Mol-CycleGAN	0.381	0.179
Gargoyles	0.782	0.122
Pair-tuning	**0.844**	0.007

E Diversity Analysis in property-optimized molecules

In this section, we demonstrate the diversity of generated molecules for two property optimization molecules. We show that the optimized molecules created by using pair tuning are highly novel with high internal diversity compared to existing methods. For the penalized logP task, pair tuning achieves the best score in both unlimited and limited length scenarios.

Table 4: Penalized logP molecule diversity (ll-38 = length-limited to 38 atoms). Results for LIMO are reproduced using their published model while results for JT-VAE use their published set of generated molecules. Size = # of examples, valid = fraction of valid SMILES, unique = fraction of unique valid molecules, novel = fraction of novel molecules, IntDiv = internal diversity (p=1) of valid molecules, SNN = average Taniomoto similarity to nearest neighbor in train, best score = maximum PLogP. **Bold** values indicate the best models for a given metric (both unlimited and length-limited).

		size	valid	unique	novel	IntDiv	SNN	best score
unlimited	train	75284	1.000	0.928	0.000	0.860	1.000	5.481
	LIMO	100000	1.000	0.994	1.000	0.857	0.226	6.810
	JT-VAE	2481	1.000	0.765	1.000	0.874	**0.372**	5.300
	Pair-tuning ($k = 125$)	100000	0.944	0.939	1.000	**0.896**	0.370	13.187
	Pair-tuning ($k = 1000$)	800000	0.946	0.916	1.000	**0.896**	0.371	**19.595**
ll-38	train	73154	1.000	0.926	0.000	0.860	1.000	5.288
	LIMO	99618	1.000	0.994	1.000	0.857	0.226	6.768
	JT-VAE	2475	1.000	0.765	1.000	0.874	0.372	4.935
	Pair-tuning ($k = 125$)	86193	1.000	0.933	1.000	**0.898**	0.372	7.123
	Pair-tuning ($k = 1000$)	692082	1.000	0.908	1.000	**0.898**	**0.373**	**9.355**

Table 5: QED molecule diversity. Results for LIMO are reproduced using their published model. Size = # of examples, valid = fraction of valid SMILES, unique = fraction of unique valid molecules, novel = fraction of novel molecules, IntDiv = internal diversity (p=1) of valid molecules, SNN = average Taniomoto similarity to nearest neighbor in train, best score = maximum QED. **Bold** values indicate the best models for a given metric.

	size	valid	unique	novel	IntDiv	SNN	best score
train	88306	1.000	0.150	0.000	0.840	1.000	**0.948**
LIMO	100000	1.000	0.972	1.000	**0.906**	0.203	0.943
Pair-tuning (k=125)	100000	0.990	0.991	1.000	0.860	**0.429**	**0.948**
Pair-tuning (k=1000)	800000	0.927	0.861	1.000	0.892	0.356	**0.948**