npj computational materials

www.nature.com/npjcompumats

BRIEF COMMUNICATION OPEN

Check for updates

Accelerating material design with the generative toolkit for scientific discovery

Matteo Manica[1]✉, Jannis Born[1], Joris Cadow[1], Dimitrios Christofidellis[1], Ashish Dave[2], Dean Clarke[2], Yves Gaetan Nana Teukam[1], Giorgio Giannone[1], Samuel C. Hoffman[3], Matthew Buchan[2], Vijil Chenthamarakshan[3], Timothy Donovan[2], Hsiang Han Hsu[4], Federico Zipoli[1], Oliver Schilter[1], Akihiro Kishimoto[4], Lisa Hamada[4], Inkit Padhi[3], Karl Wehden[3], Lauren McHugh[3], Alexy Khrabrov[5], Payel Das[3], Seiji Takeda[4] and John R. Smith[3]

With the growing availability of data within various scientific domains, generative models hold enormous potential to accelerate scientific discovery. They harness powerful representations learned from datasets to speed up the formulation of novel hypotheses with the potential to impact material discovery broadly. We present the Generative Toolkit for Scientific Discovery (GT4SD). This extensible open-source library enables scientists, developers, and researchers to train and use state-of-the-art generative models to accelerate scientific discovery focused on organic material design.

npj Computational Materials (2023)9:69 ; https://doi.org/10.1038/s41524-023-01028-1

INTRODUCTION

The rapid technological progress in the last centuries has been largely fuelled by the success of the scientific method. However, in some of the most important fields, such as material or drug discovery, productivity has been decreasing dramatically[1], and by today it can take almost a decade to discover new material and cost upwards of $10–$100 million. One of the most daunting challenges in materials discovery is hypothesis generation. The reservoir of natural products and their derivatives has been largely emptied[2] and bottom-up human-driven hypotheses have shown that it is extremely challenging to identify and select novel and useful candidates in search spaces that are overwhelming in size, e.g., the chemical space for drug-like molecules is estimated to contain $>10^{33}$ structures[3].

To overcome this problem, in recent years, machine learning-based generative models, e.g., variational autoencoders (VAEs[4]), generative adversarial networks (GANs;[5]) have emerged as a practical approach to designing and discovering molecules with desired properties leveraging different representations for molecular structure, e.g., text-based like SMILES[6] and SELFIES[7] or graph-based[8]. Compared to exhaustive or grid searches, generative models more efficiently and effectively navigate and explore vast search spaces learned from data based on user-defined criteria. Leveraging these approaches in With a series of seminal works[9–13], research has covered a wide variety of applications of generative models, including design, optimisation and discovery of sugar and dye molecules[14], ligands for specific targets[15–18], anti-cancer hit-like molecules[19,20], antimicrobial peptides[21] and semiconductors[22].

At the same time, we have witnessed growing community efforts for developing software packages to evaluate and benchmark machine learning models and their application in material science. On the property prediction side, models, data-mining toolkits and benchmarking suites for material property prediction, such as CGCNN[23], `pymatgen`[24], Matminer[25] or Matbench/Automatminer[26] were released. On the generative side, initial efforts for generic frameworks implementing popular baselines and metrics such as GuacaMol[27] and Moses[28] paved the way for domain-specific generative model software that is gaining popularity in the space of drug discovery such as TDC (Therapeutics Data Commons[29,30]).

More recently, novel families of methods have been proposed. Generative flow networks (GFN[31–33]), a generative model that leverages ideas from reinforcement learning to improve sample diversity, provides a non-iterative sampling mechanism for structured data over graphs. GFNs are particularly suited for molecule generation, where sample diversity is challenging. Diffusion models (DM[34–36]) are generative models that learn complex high-dimensional distributions denoising the data at multiple scales. DMs achieve impressive results in terms of sample quality and diversity for unconditional and conditional vision tasks. Recently, text-conditional diffusion models[37–39] have paved the way for a new age of human–machine interaction. Leveraging such advances in conditioning generative models, DMs have been used in the biological domain for molecule conformation using equivariant graph networks[40], conditioning on a 2D representation of the molecule to generate the 3D pose in space[41], for protein generation[42,43] and docking[44].

In this landscape, there is a growing need for libraries and toolkits that can lower the barrier to using generative models. This need is becoming significantly more pressing given the growing models' size and their significant requirements for considerable computational resources for training them. This trend creates an imbalance between a small, privileged group of researchers in well-funded institutions and the rest of the scientific community, thus impeding open, collaborative, and fair science principles[45].

We introduce the generative toolkit for scientific discovery (GT4SD) as a remedy. This Python library aims to bridge this gap by developing a framework that eases the training, execution and development of generative models to accelerate scientific discovery. As visualised in Fig. 1, GT4SD provides a harmonised interface with a singular application registry for all generative models and a separate registry for properties. This expenses the need to familiarise with the original developer's code, thus significantly lowering the access barrier. Moreover, the high

[1]IBM Research Europe - Zurich, Rüschlikon, Switzerland. [2]IBM Research - UK, Hursley, UK. [3]IBM Research - Yorktown Heights, New York, USA. [4]IBM Research - Tokyo, Tokyo, Japan. [5]IBM Research - Almaden, San Jose, USA. ✉email: tte@zurich.ibm.com

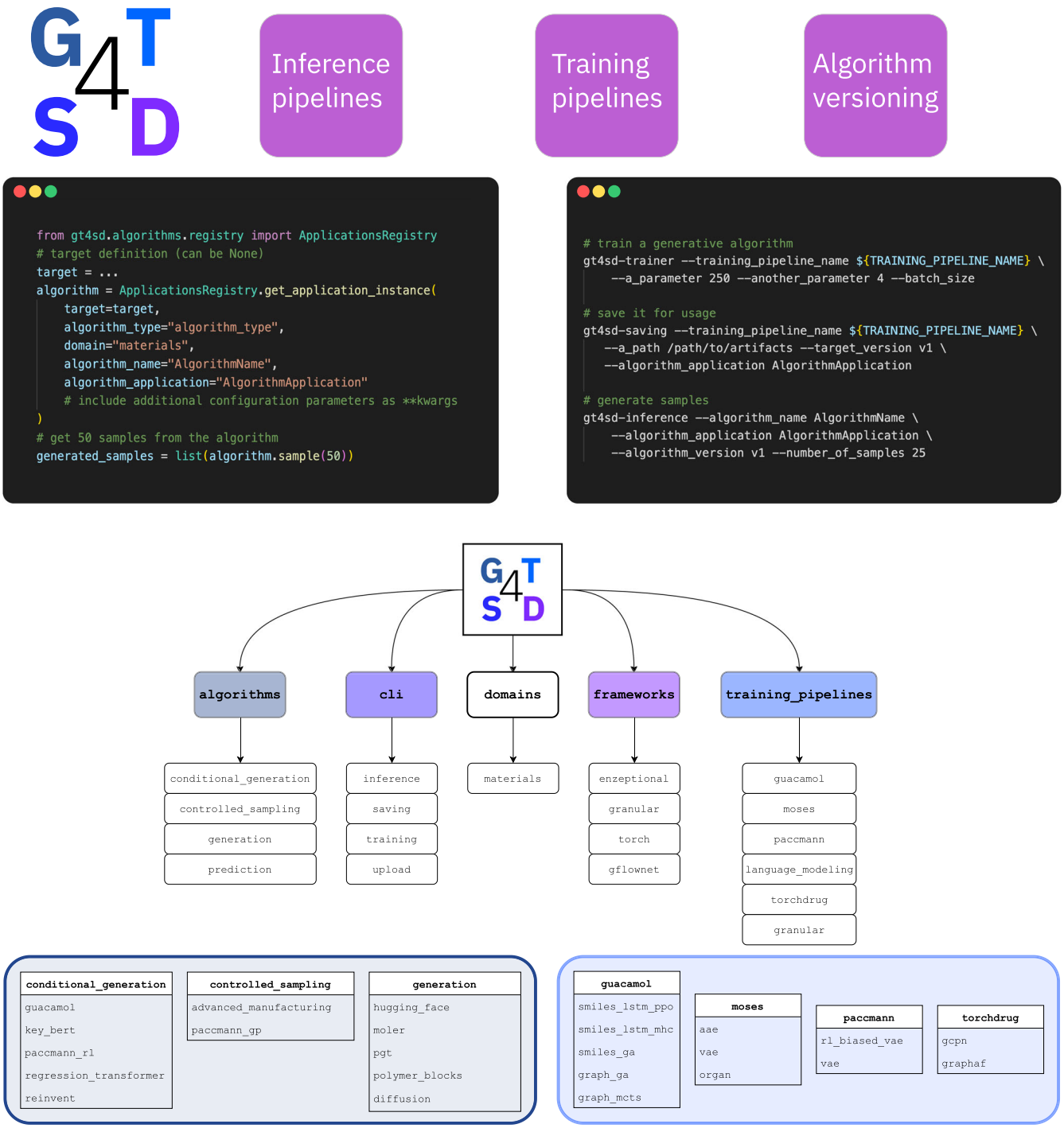

Fig. 1 GT4SD overview and structure. The library implements pipelines for the inference and training of generative models. In addition, GT4SD offers utilities for algorithm versioning and sharing for broader usage in the community. The standardised interface enables algorithm instantiation and runs for generating samples with less than five lines of code (top, left panel). Furthermore, the CLI tools ease the run of a full discover pipeline in the terminal (top, right panel). The library provides (bottom, from left to right) algorithms for inference, a CLI utility, target domains, a property prediction interface, interfaces and implementations of generative modelling frameworks and training pipelines. In the blue box, we provide a sample of available frameworks and methodologies for inference algorithms.

standardisation across models eases the integration of new models and facilitates consumption by containerisation or distributed computing system. To the best of our knowledge, GT4SD provides the largest framework for accessing state-of-the-art generative models. It can be used to execute, train, fine-tune and deploy generative models, all either directly through Python or via a highly flexible command line interface (CLI). All pre-trained models can be executed directly from the browser through web apps hosted on Hugging Face Spaces. Last, for advanced users, the GT4SD model hub simplifies the release of existing algorithms trained on new datasets for instant and continuous integration in their discovery workflows.

GT4SD offers a set of capabilities for generating novel hypotheses (inference pipelines) and for fine-tuning domain-

specific generative models (training pipelines). It is designed to be compatible and inter-operable with existing popular libraries, including PyTorch[46], PyTorch Lightning[47], Hugging Face Transformers[48], Diffusers[49], GuacaMol[27], Moses[28], TorchDrug[50], GFlowNets[33] and MoLeR[51]. It includes a wide range of pre-trained models and applications for material design.

GT4SD provides simple interfaces to make generative models easily accessible to users who want to deploy them with just a few lines of code. The library provides an environment for researchers and students interested in applying state-of-the-art models in their scientific research, allowing them to experiment with a wide variety of pre-trained models spanning a broad spectrum of material science and drug discovery applications. Furthermore, GT4SD provides a standardised CLI, APIs for inference and training without compromising on the ability to specify an algorithm's finer-grained parameters and >15 web apps of various pre-trained models.

RESULTS

A case study in molecular discovery

Arguably, the most considerable potential for accelerating scientific discovery lies in the field of de novo molecular design, particularly in material and drug discovery. With several (pre) clinical trials underway[52], it is a matter of time until the first AI-generated drug will receive FDA approval and reach the market. In a seminal study by[15], a deep reinforcement learning model (GENTRL) was utilised for the discovery of potent DDR1 inhibitors, a prominent protein kinase target involved in fibrosis, cancer, and other diseases[53]. Six molecules were synthesised, four were found active in a biochemical assay, and one compound (in the following called *gentrl-ddr1*) demonstrated favourable pharmacokinetics in mice. As an exemplary case study in molecular discovery, we consider a contrived task of adapting the hit-compound *gentrl-ddr1* to a similar molecule with an improved estimated water solubility (ESOL; Delaney[54]). Low aqueous solubility affects >40% of new chemical entities[55], thus posing major barriers to drug delivery. Improving solubility requires exploring the local chemical space around the hit (i.e., *gentrl-ddr1*) to find an optimised lead compound.

A summary of how this task can be addressed using the GT4SD is shown in Fig. 2. In the first step, a rich set of pre-trained molecular generative models is accessed with the harmonised interface of the GT4SD. Two main model classes are available. The first category is represented by graph generative models, such as MoLeR[51] or models from the TorchDrug library, specifically a graph-convolutional policy network[12] and a flow-based autoregressive model (GraphAF[56]). The second model class is chemical language models (CLM), which treat molecules as text (SMILES[6] or SELFIES[7] sequences). Most of the chemical language models in the GT4SD are accessed via the libraries MOSES[28] or GuacaMol[27]; in particular, a VAE[9], an adversarial autoencoder (AAE[57]) or an objective-reinforced GAN model (ORGAN;[58]). In the first step, we randomly sample molecules from the learned chemical space of each model. Assessing the Tanimoto similarity of the generated molecules to *gentrl-ddr1* reveals that this approach, while producing many molecules with satisfying ESOL, did not sufficiently reflect the similarity constraint to the seed molecule (cf. Fig. 2, *bottom left*). This is expected because the investigated generative models are *unconditional*.

As a more refined approach, the GT4SD includes conditional molecular generative models that can be primed with natural text queries (Text+Chem T5[59]), continuous property constraints or molecular substructures (e.g., scaffolds) such as MoLeR[51], REINVENT[60] or even with combinations of property constraints and molecular substructures (Regression Transformer;[61]). The molecules obtained from those models, in particular MoLeR and RT, largely respected the similarity constraint and produced many

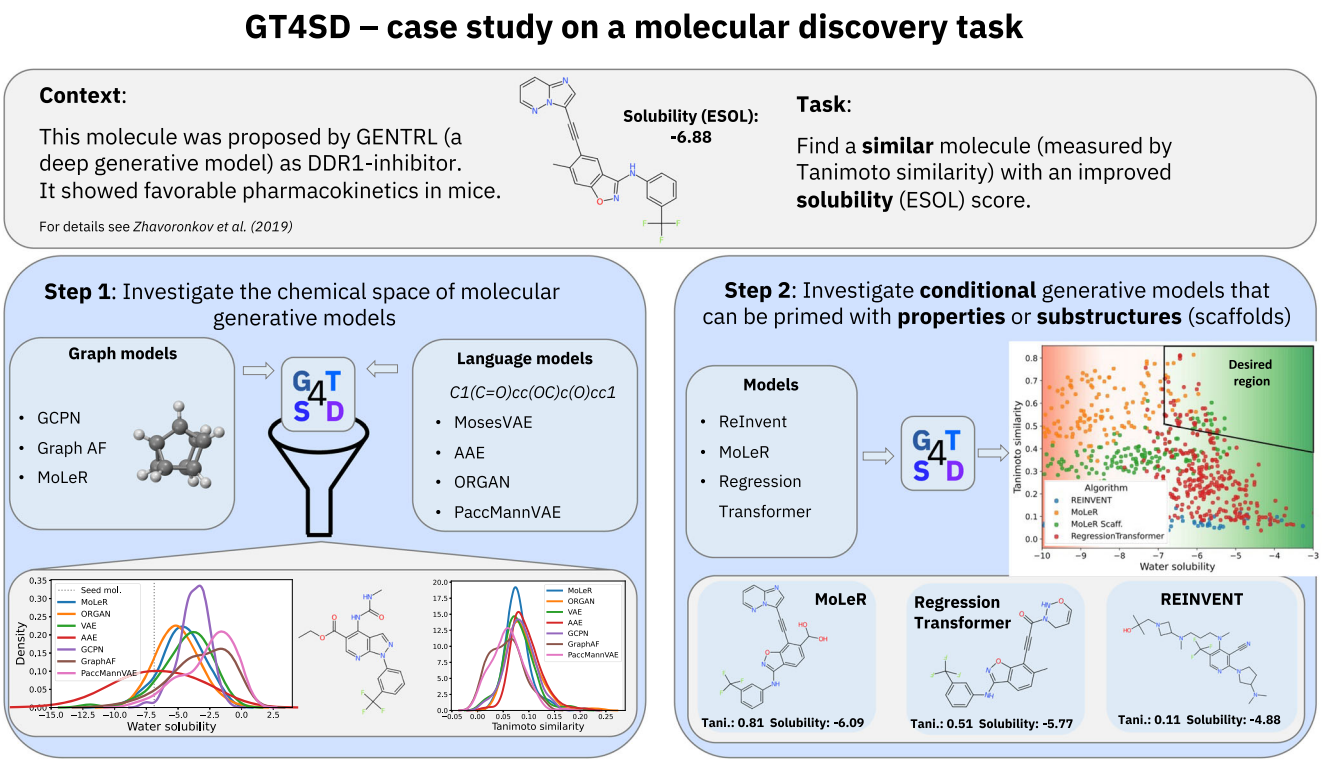

Fig. 2 Case study using the GT4SD for molecular discovery. Starting from a compound designed using generative models by[15] (*gentrl-ddr1*), we show how GT4SD can be used to swiftly design molecules with desired properties using a battery of algorithms available in the library in two settings: unconditional (bottom left) and conditional (bottom right). The conditional models can be constrained with chemical scaffolds or conditioned on desired property values.

molecules with a Tanimoto similarity > 0.5 to *gentrl-ddr1*. MoLeR and the RT improved the ESOL by more than 1*M*/*L* (cf. Fig. 2, *right*). In a realistic discovery scenario, the molecules generated with the described recipes could be manually reviewed by medicinal chemists and selectively considered for synthesis and screening.

DISCUSSION

The GT4SD is the first step toward a harmonised generative modelling environment for accelerated material discovery. For the future, we plan to expand application domains (e.g., inorganic materials, climate, weather[62], sustainability, geo-informatics and human mobility[63]), and integrate novel algorithms, ideally with the support of a steadily growing open-science community.

Future developments will focus on two main components: expanding model evaluation and sample properties predictions; developing an ecosystem for sharing models built on top of the functionalities exposed via the existing CLI commands for model lifecycle management. For the first aspect, we will expand the currently integrated metrics from GuacaMol and Moses and explore bias measures to better analyse performance in light of the generated examples and their properties. Regarding the sharing ecosystem, we believe GT4SD will further benefit from an intuitive application hub that facilitates the distribution of pre-trained generative models (largely inspired by the Hugging Face model hub[48]) and enables users to easily fine-tune models on custom data for specific applications.

We anticipate GT4SD to democratise generative modelling in the material sciences and to empower the scientific community to access, evaluate, compare and refine large-scale pre-trained models across a wide range of applications.

METHODS

Library structure

The GT4SD library follows a modular structure (Fig. 1) where the main components are: (i) algorithms for serving models in inference mode following a standardised API; (ii) training pipelines sharing a common interface with algorithm families-specific implementations; (iii) domain-specific utilities shared across various algorithms; (iv) a property prediction interface to evaluate generated samples (currently covering small molecules, proteins and crystals); (v) frameworks implementing support for complex workflows, e.g., granular for training mixture of generative and predictive models or exceptional for enzyme design. Besides the core components, there are sub-modules for configuration, handling the cloud object storage-based cache and error handling at the top level.

Inference pipelines

The API implementation underlying the inference pipelines has been designed to support various generative model types: generation, conditional generation, controlled sampling and simple prediction algorithms. All the algorithms implemented in GT4SD follow a standard contract that guarantees a standardised way to call an algorithm in inference mode. The specific algorithm interface and applications are responsible for defining implementation details and loading the model files from a cache synced with a cloud object storage hosting their versions.

Training pipelines

Training pipelines follow the same philosophy adopted in implementing the inference pipelines. A common interface allows implementing algorithm family-specific classes with an arbitrary customisable training method that can be configured using a set of data classes. Each training pipeline is associated with a class implementing the actual training process and a triplet of configuration data classes that control arguments for model hyper-parameters, training parameters and data parameters.

CLI commands

To ease consumption of the pipelines and models implemented in GT4SD, a series of CLI endpoints are available alongside the package: (i) `gt4sd-inference`, to inspect and run pipelines for inference; (ii) `gt4sd-trainer`, to list and configure training pipelines; (iii) `gt4sd-saving`, to persist in a local cache a model version trained via GT4SD for usage in inference mode; (iv) `gt4sd-upload`, to upload model versions trained via GT4SD on a model hub to share algorithms with other users. The CLI commands allow to implement a complete discovery workflow where, starting from a source algorithm version, users can retrain it on custom datasets and make a new algorithm version available in GT4SD.

DATA AVAILABILITY

The complete documentation for the GT4SD code base is available at https://gt4sd.github.io/gt4sd-core/. Pre-trained models and property predictors are available for automated download via the library itself.

CODE AVAILABILITY

GT4SD source code is available on GitHub (Zenodo). The repository also contains exemplary notebooks and examples for users, including code and data to reproduce the presented case study. Pre-trained generative models and property predictors are also available as Gradio[64] apps with the corresponding model cards in the GT4SD organisation on Hugging Face Spaces: https://huggingface.co/GT4SD.

Received: 19 November 2022; Accepted: 15 April 2023;
Published online: 01 May 2023

REFERENCES

1. Smietana, K., Siatkowski, M. & Møller, M. Trends in clinical success rates. *Nat. Rev. Drug Discov.* **15**, 379–80 (2016).
2. Atanasov, A. G., Zotchev, S. B., Dirsch, V. M. & Supuran, C. T. Natural products in drug discovery: advances and opportunities. *Nat. Rev. Drug Discov.* **20**, 200–216 (2021).
3. Polishchuk, P. G., Madzhidov, T. I. & Varnek, A. Estimation of the size of drug-like chemical space based on gdb-17 data. *J. Comput. Aided Mol. Des.* **27**, 675–679 (2013).
4. Kingma, D. P. & Welling, M. Auto-encoding variational bayes. Preprint at *arXiv* https://arxiv.org/abs/1312.6114 (2013).
5. Goodfellow, I. et al. Generative adversarial nets.*Adv. Neural Inf. Process. Syst.* **27**, 2672–2680 (2014).
6. Weininger, D. Smiles, a chemical language and information system. 1. Introduction to methodology and encoding rules. *J. Chem. Inf. Comp. Sci.* **28**, 31–36 (1988).
7. Krenn, M., Häse, F., Nigam, A., Friederich, P. & Aspuru-Guzik, A. Self-referencing embedded strings (SELFIES): A 100% robust molecular string representation. *Mach. Learn.* **1**, 045024 (2020).
8. King, R. Chemical Applications of Topology and Graph Theory: A Collection of Papers from a Symposium Held at the University of Georgia, Athens, Georgia, USA, 18–22 April 1983. *Developments in Geotectonics* (Elsevier, 1983).
9. Gómez-Bombarelli, R. et al. Automatic chemical design using a data-driven continuous representation of molecules. *ACS Cent. Sci.* **4**, 268–276 (2018).
10. Segler, M. H., Kogej, T., Tyrchan, C. & Waller, M. P. Generating focused molecule libraries for drug discovery with recurrent neural networks. *ACS Cent. Sci.* **4**, 120–131 (2018).
11. Jin, W., Barzilay, R. & Jaakkola, T. Junction tree variational autoencoder for molecular graph generation. in *International Conference on Machine Learning*, 2323–2332 (PMLR, 2018).
12. You, J., Liu, B., Ying, Z., Pande, V. & Leskovec, J. Graph convolutional policy network for goal-directed molecular graph generation. *Adv. Neural Inf. Process. Syst.* **31**, 6410–6421 (2018).
13. Prykhodko, O. et al. A de novo molecular generation method using latent vector based generative adversarial network. *J. Cheminform.* **11**, 1–13 (2019).

14. Takeda, S. et al. Molecular inverse-design platform for material industries. in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2961–2969 (2020).
15. Zhavoronkov, A. et al. Deep learning enables rapid identification of potent ddr1 kinase inhibitors. *Nat. Biotechnol.* **37**, 1038–1040 (2019).
16. Chenthamarakshan, V. et al. Cogmol: target-specific and selective drug design for covid-19 using deep generative models. *Adv. Neural Inf. Process. Syst.* **33**, 4320–4332 (2020).
17. Born, J. et al. Data-driven molecular design for discovery and synthesis of novel ligands: a case study on sars-cov-2. *Mach. Learn.: Sci. Technol.* **2**, 025024 (2021).
18. Hoffman, S. C., Chenthamarakshan, V., Wadhawan, K., Chen, P.-Y. & Das, P. Optimizing molecules using efficient queries from property evaluations. *Nat. Mach. Intell.* **4**, 21–31 (2022).
19. Méndez-Lucio, O., Baillif, B., Clevert, D.-A., Rouquié, D. & Wichard, J. De novo generation of hit-like molecules from gene expression signatures using artificial intelligence. *Nat. Commun.* **11**, 1–10 (2020).
20. Born, J. et al. PaccMann^{RL}: de novo generation of hit-like anticancer molecules from transcriptomic data via reinforcement learning. *iScience* **24**, 102269 (2021).
21. Das, P. et al. Accelerated antimicrobial discovery via deep generative models and molecular dynamics simulations. *Nat. Biomed. Eng.* **5**, 613–623 (2021).
22. Siriwardane, E. M. D., Zhao, Y., Perera, I. & Hu, J. Generative design of stable semiconductor materials using deep learning and density functional theory. *npj Comput. Mater.* **8**, 164 (2022).
23. Xie, T. & Grossman, J. C. Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties. *Phys. Rev. Lett.* **120**, 145301–145306 (2018).
24. Ong, S. P. et al. Python materials genomics (pymatgen): a robust, open-source python library for materials analysis. *Comput. Mater. Sci.* **68**, 314–319 (2013).
25. Ward, L. et al. Matminer: an open source toolkit for materials data mining. *Comput. Mater. Sci.* **152**, 60–69 (2018).
26. Dunn, A., Wang, Q., Ganose, A., Dopp, D. & Jain, A. Benchmarking materials property prediction methods: the matbench test set and automatminer reference algorithm. *npj Comput. Mater.* **6**, 138 (2020).
27. Brown, N., Fiscato, M., Segler, M. H. & Vaucher, A. C. Guacamol: benchmarking models for de novo molecular design. *J. Chem. Inf. Model.* **59**, 1096–1108 (2019).
28. Polykovskiy, D. et al. Molecular sets (Moses): a benchmarking platform for molecular generation models. *Front. Pharmacol.* **11**, 1931 (2020).
29. Huang, K. et al. Therapeutics data commons: Machine learning datasets and tasks for drug discovery and development. *Adv. Neural Inf. Process. Syst.* **35** https://datasets-benchmarks-proceedings.neurips.cc/paper_files/paper/2021/hash/4c56ff4ce4aaf9573aa5dff913df997a-Abstract-round1.html (2021).
30. Huang, K. et al. Artificial intelligence foundation for therapeutic science. *Nat. Chem. Biol.* **11**, 191–200 (2022).
31. Bengio, E., Jain, M., Korablyov, M., Precup, D. & Bengio, Y. Flow network based generative models for non-iterative diverse candidate generation. *Adv. Neural Inf. Process. Syst.* **34**, 27381–27394 (2021).
32. Bengio, Y. et al. Gflownet foundations. Preprint at *arXiv* https://arxiv.org/abs/2111.09266 (2021).
33. Jain, M. et al. Biological sequence design with gflownets. In *International Conference on Machine Learning*, 9786–9801 (PMLR, 2022).
34. Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N. & Ganguli, S. Deep unsupervised learning using nonequilibrium thermodynamics. in *International Conference on Machine Learning*, 2256–2265 (PMLR, 2015).
35. Song, Y. & Ermon, S. Generative modeling by estimating gradients of the data distribution. *Adv. Neural Inf. Process. Syst.* **32**, 11918–11930 (2019).
36. Ho, J., Jain, A. & Abbeel, P. Denoising diffusion probabilistic models. *Adv. Neural Inf. Process. Syst.* **33**, 6840–6851 (2020).
37. Ramesh, A., Dhariwal, P., Nichol, A., Chu, C. & Chen, M. Hierarchical text-conditional image generation with clip latents. Preprint at *arXiv* https://arxiv.org/abs/2204.06125 (2022).
38. Rombach, R., Blattmann, A., Lorenz, D., Esser, P. & Ommer, B. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10684–10695 (2022).
39. Saharia, C. et al. Photorealistic text-to-image diffusion models with deep language understanding. *Adv. Neural Inf. Process. Syst.* **35**, 36479–36494 (2022).
40. Hoogeboom, E., Satorras, V. G., Vignac, C. & Welling, M. Equivariant diffusion for molecule generation in 3d. in *International Conference on Machine Learning*, 8867–8887 (PMLR, 2022).
41. Xu, M. et al. Geodiff: A geometric diffusion model for molecular conformation generation. In *The Tenth International Conference on Learning Representations, ICLR* (2022).
42. Anand, N. & Achim, T. Protein structure and sequence generation with equivariant denoising diffusion probabilistic models. Preprint at *arXiv* https://arxiv.org/abs/2205.15019 (2022).
43. Wu, K. E. et al. Protein structure generation via folding diffusion. Preprint at *arXiv* https://arxiv.org/abs/2209.15611 (2022).
44. Corso, G., Stärk, H., Jing, B., Barzilay, R. & Jaakkola, T. Diffdock: Diffusion steps, twists, and turns for molecular docking. in *The Eleventh International Conference on Learning Representations, ICLR* (2023).
45. Probst, D. Aiming beyond slight increases in accuracy. *Nat. Rev. Chem.* **7**, 1–2 (2023).
46. Paszke, A. et al. Pytorch: an imperative style, high-performance deep learning library. *Adv. Neural Inf. Process. Syst.* **32**, 8024–8035 (2019).
47. Falcon, W. & The PyTorch Lightning team. *PyTorch Lightning*. https://github.com/PyTorchLightning/pytorch-lightning (2022).
48. Wolf, T. et al. Transformers: State-of-the-art natural language processing. in *Proceedings of the 2020 conference on empirical methods in natural language processing: system demonstrations*, 38–45 (2020).
49. von Platen, P. et al. Diffusers: state-of-the-art diffusion models. https://github.com/huggingface/diffusers (2022).
50. Zhu, Z. et al. Torchdrug: A powerful and flexible machine learning platform for drug discovery. Preprint at *arXiv* https://arxiv.org/abs/2202.08320 (2022).
51. Maziarz, K. et al. Learning to extend molecular scaffolds with structural motif. in *The Tenth International Conference on Learning Representations, ICLR* (2022).
52. Jayatunga, M. K., Xie, W., Ruder, L., Schulze, U. & Meier, C. Ai in small-molecule drug discovery: a coming wave? *Nat. Rev. Drug Discov.* **21**, 175–176 (2022).
53. Hidalgo-Carcedo, C. et al. Collective cell migration requires suppression of actomyosin at cell–cell contacts mediated by ddr1 and the cell polarity regulators par3 and par6. *Nat. Cell Biol.* **13**, 49–59 (2011).
54. Delaney, J. S. Esol: estimating aqueous solubility directly from molecular structure. *J. Chem. Inf. Comp. Sci.* **44**, 1000–1005 (2004).
55. Savjani, K. T., Gajjar, A. K. & Savjani, J. K. Drug solubility: importance and enhancement techniques. *Int. Sch. Res. Notices* **2012** https://www.hindawi.com/journals/isrn/2012/195727/ (2012).
56. Shi, C. et al. Graphaf: a flow-based autoregressive model for molecular graph generation. in *The Eighth International Conference on Learning Representations, ICLR* (2020).
57. Kadurin, A. et al. The cornucopia of meaningful leads: applying deep adversarial autoencoders for new molecule development in oncology. *Oncotarget* **8**, 10883 (2017).
58. Guimaraes, G. L., Sanchez-Lengeling, B., Outeiral, C., Farias, P. L. C. & Aspuru-Guzik, A. Objective-reinforced generative adversarial networks (organ) for sequence generation models. Preprint at *arXiv* https://arxiv.org/abs/1705.10843 (2017).
59. Christofidellis, D. et al. Unifying molecular and textual representations via multi-task language modelling. Preprint at *arXiv* https://arxiv.org/abs/2301.12586 (2023).
60. Blaschke, T. et al. Reinvent 2.0: an ai tool for de novo drug design. *J. Chem. Inf. Model.* **60**, 5918–5922 (2020).
61. Born, J. & Manica, M. Regression transformer enables concurrent sequence regression and generation for molecular language modeling. *Nat. Mach. Intell.* **5**, 432–444 (2023).
62. Ravuri, S. et al. Skilful precipitation nowcasting using deep generative models of radar. *Nature* **597**, 672–677 (2021).
63. Yan, X.-Y., Wang, W.-X., Gao, Z.-Y. & Lai, Y.-C. Universal model of individual and population mobility on diverse spatial scales. *Nat. Commun.* **8**, 1–9 (2017).
64. Abid, A. et al. Gradio: hassle-free sharing and testing of ml models in the wild. Preprint at *arXiv* https://arxiv.org/abs/1906.02569 (2019).

ACKNOWLEDGEMENTS

The authors acknowledge Helena Montenegro, Yoel Shoshan, Nicolai Ree, Miruna Cretu and Helder Lopes for their open-source contributions to the GT4SD.

AUTHOR CONTRIBUTIONS

All authors contributed to the design and implementation of different library components before and after its release. M.M., J.B., D.C., G.G., V.C., A.K., L.M. and J.R.S. contributed to writing and revising the brief communication. J.B. designed and implemented the case study as well as the Gradio apps.

COMPETING INTERESTS

The authors declare no competing interests.

ADDITIONAL INFORMATION

Correspondence and requests for materials should be addressed to Matteo Manica.