Graph Representation Learning of Molecules for *de novo* Drug Development

Dissertation

zur Erlangung des Grades eines
Doktors der Naturwissenschaften
(Dr. rer. nat.)
am Fachbereich Mathematik und Informatik
der Freien Universität Berlin

vorgelegt von

Tuan Le

Berlin, 2025

Erstgutachter: Prof. Dr. Frank Noé

Zweitgutachter: Ass. Prof. Dr. Johannes Brandstetter

Drittgutachter: Dr. Djork-Arné Clevert

Tag der Disputation: 14.02.2025

Abstract

Discovering novel pharmaceutical drugs is a lengthy, complex, and resource-intensive process. Traditional approaches in chemoinformatics have provided valuable tools for predicting molecular properties, but these methods can fall short in terms of efficiency and predictive accuracy. When used in molecular generation, these limitations become more pronounced, as inaccuracies in property prediction can lead to the design of suboptimal candidates. This dissertation explores innovative approaches to drug discovery through the application of deep graph representation learning techniques, focusing on two areas: molecular property prediction and molecular generation. We propose novel graph neural network architectures designed to enhance expressiveness in 2D and 3D molecular graphs. Our approach incorporates feature transformations inspired by hypercomplex algebras or integrates group equivariance into the models, facilitating data-efficient training. For 3D molecular data, we demonstrate that rotation-equivariant networks can be scaled to process larger biomolecules and outperform invariant networks while remaining computationally efficient. Additionally, we introduce generative models based on diffusion probabilistic models that sample new 3D molecular structures with targeted properties, either for ligands in isolation or within protein-ligand complexes, using rotation-equivariant graph networks as denoisers. This aspect of our research aims to enhance the drug discovery pipeline by improving the efficiency of identifying promising drug candidates that meet multiple criteria. The results of this thesis suggest that deep graph representation learning has the potential to advance drug discovery by providing more accurate predictive tools and enhancing the ability to generate novel molecular candidates. This work contributes to the development of computational methods in drug discovery and may pave the way for further research into applying graph representation learning to complex chemical problems.

Zusammenfassung

Die Entdeckung neuer pharmazeutischer Medikamente ist ein langwieriger, komplexer und ressourcenintensiver Prozess. Traditionelle Ansätze in der Chemieinformatik haben wertvolle Methoden zur Vorhersage molekularer Eigenschaften bereitgestellt, doch diese Methoden können in Bezug auf Effizienz und Vorhersagegenauigkeit unzureichend sein. Bei der Generierung von Molekülen werden diese Einschränkungen noch deutlicher, da Ungenauigkeiten bei der Vorhersage in der Eigenschaftsvorhersage zu suboptimalen Kandidaten führen können. Diese Dissertation untersucht innovative Ansätze zur Entdeckung von Arzneimittel durch die Anwendung von Lerntechnicken von tiefen Graph Repräsentationen, wobei der Schwerpunkt auf zwei Bereichen liegt: der Vorhersage molekularer Eigenschaften und der molekularen Generierung. Wir schlagen neuartige Architekturen von Graph Neuronalen Netzwerken vor, die entwickelt wurden, um die Ausdruckskraft in 2D- als auch in 3D-Molekülgraphen zu verbessern. Unser Ansatz integriert Transformationen, inspiriert von hyperkomplexen Algebren, oder integriert Gruppenequivarianz in die Modelle, um ein daten-effizientes Training zu ermöglichen. Für 3D-molekulare Daten zeigen wir, dass rotationsequivariante Netzwerke auf die Verarbeitung größerer Biomoleküle skaliert werden können und invariant Netzwerke übertreffen, während sie gleichzeitig recheneffizient bleiben. Zusätzlich stellen wir generative Modelle vor, die auf Diffusions-Wahrscheinlichkeitsmodellen basieren und neue 3D-molekulare Strukturen mit gezielten Eigenschaften erzeugen, entweder für Liganden in Isolation oder in Protein-Ligand-Komplexen, wobei rotatations-equivariante Graphnetzwerke in der Diffusion verwendet werden. Dieser Aspekt unserer Forschung zielt darauf ab, den Prozess zur Arzneimittelentdeckungs zu verbessern, indem die Effizienz bei der Identifizierung vielversprechender Molekül Kandidaten, die mehrere Kriterien erfüllen, erhöht wird. Die Ergebnisse dieser Dissertation deuten darauf hin, dass Lerntechnicken von tiefen Graph Repräsentationen das Potenzial haben, die Arzneimittelentdeckung voranzutreiben, indem es genauere Vorhersagewerkzeuge bereitstellt und die Fähigkeit zur Generierung neuer molekularer Kandidaten verbessert. Diese Arbeit trägt zur Entwicklung rechnerischer Methoden in der Arzneimittelentdeckung bei und könnte den Weg für weitere Forschungen zur Anwendung der Lerntechnicken von tiefen Graph Repräsentationen auf komplexe chemische Probleme ebnen.

Selbstständigkeitserklärung

Name: Le

Vorname: Tuan

Ich erkläre gegenüber der Freien Universität Berlin, dass ich die vorliegende Dissertation selbstständig und ohne Benutzung anderer als der angegebenen Quellen und Hilfsmittel angefertigt habe. Die vorliegende Arbeit ist frei von Plagiaten. Alle Ausführungen, die wörtlich oder inhaltlich aus anderen Schriften entnommen sind, habe ich als solche kenntlich gemacht. Diese Dissertation wurde in gleicher oder ähnlicher Form noch in keinem früheren Promotionsverfahren eingereicht. Mit einer Prüfung meiner Arbeit durch ein Plagiatsprüfungsprogramm erkläre ich mich einverstanden.

Datum: ..... Unterschrift: .....

Acknowledgements

This thesis is the culmination of four years of research conducted at Bayer AG and Pfizer GmbH in collaboration with the Freie Universität Berlin, Germany. I want to express my sincere gratitude to everyone who has contributed to this work in various ways.

First of all I would like to thank my industry supervisor Djork-Arné Clevert for his constant support throughout my time at Bayer and for facilitating my continuation of the PhD research at Pfizer. I am deeply grateful for his guidance throughout my scientific journey by advising me with goodwill, allowing me choose topics and collaborators, but still asking the right questions to help me choose fruitful research directions.

Furthermore, I am very appreciative to my academic supervisor Prof. Frank Noé for giving me the opportunity to write this thesis in his department, where I met many great people and received invaluable scientific insights and inspiration related to molecular modeling. I am grateful that Prof. Frank Noé accepted to supervise my work and supported me with feedback and research ideas.

I would also like to thank all the group members of the Machine Learning Research Team at Bayer and Pfizer who made it a memorable and joyful time, including Santiago Villalba, Floriane Montanari, Paula A. M. Zapata, Vincenzo Palmacci, Anastasia Pentina, Flavio Morelli, Andreas Pöhlmann, Andreas Steffen and Joren Retel. Furthermore, I want to thank Robin Winter, who was also a PhD student at Bayer, and with whom I spent a lot of time discussing deep learning methods for drug discovery with a focus on graph representation learning.

I want to extend my thanks to Marco Bertolini for his invaluable assistance throughout my PhD journey. His help in discussing research papers, brainstorming new ideas, and guiding me in scientific writing has been indispensable. With his background in theoretical and mathematical physics, I was very fortunate to have a mentor with whom I could talk about group and representation theory to the point of understanding.

Moreover, I would like to thank my other collaborators and co-authors: Kristof Schütt and Julian Cremer.

Lastly, I want to express my deepest gratitude to my family and friends for their unwavering support throughout my 4 years of research. They reminded me of the importance of balance and the need to take breaks from the PhD journey. I also want to thank Marta for her constant love, support, and belief in me as a researcher from the very beginning. Her faith in me has given me strength in accomplishing my goals.

List of Publications

First-author publications part of this thesis:

- **Le, Tuan**; Winter, Robin; Noé, Frank; & Clevert, Djork-Arné (2020). Neuraldecipher–reverse-engineering extended-connectivity fingerprints (ECFPs) to their molecular structures. *Chemical science*, 2020, 11(38), 10378-10389.
- **Le, Tuan**; Bertolini, Marco; Noé, Frank; & Clevert, Djork-Arné (2021). Parameterized Hypercomplex Graph Neural Networks for Graph Classification. *30th International Conference on Artificial Neural Networks.*
- **Le, Tuan**; Noé, Frank; & Clevert, Djork-Arné (2022). Representation Learning on Biomolecular Structures Using Equivariant Graph Attention. *Proceedings of the First Learning on Graphs Conference, PMLR 198:30:1-30:17, 2022.*
- **Le, Tuan**; **Cremer, Julian**; Noé, Frank; Clevert, Djork-Arné & Schütt, Kristof (2024). Navigating the Design Space of Equivariant Diffusion-Based Generative Models for De Novo 3D Molecule Generation. *12th International Conference on Learning Representations. 2024*
- **Cremer, Julian**; **Le, Tuan**; Noé, Frank; Clevert, Djork-Arné & Schütt, Kristof (2024). PILOT: Equivariant diffusion for pocket conditioned de novo ligand generation with multi-objective guidance via importance sampling. *Chemical science*, 15(36), 2024, 14954-14967.

Other publications not part of this thesis:

- **Le, Tuan**; **Bertolini, Marco**; **Boef, Marc Arne** & Clevert, Djork-Arné (2020). Going full hyper: hyperbolic and hypercomplex graph embeddings for ADMET modeling. *Machine Learning for Molecules Workshop at NeurIPS 2020.*
- **Clevert, Djork-Arné**; Le, Tuan, Winter, Robin; & Montanari, Floriane (2021). Img2Mol–accurate SMILES recognition from molecular graphical depictions. *Chemical science*, 12(42), 14174-14181.
- **Le, Tuan**; Noé, Frank & Clevert, Djork-Arné (2022). Equivariant Graph Attention Networks for Molecular Property Prediction. *arXiv preprint 2202.09891. 2022*
- **Winter, Robin**; **Bertolini, Marco**; Le, Tuan; Noé, Frank & Clevert, Djork-Arné (2022). Unsupervised Learning of Group Invariant and Equivariant Representations. *Advances in Neural Information Processing Systems*, 35, 31942-31956.
- **Marin Zapata, Paula Andrea**; **Méndez-Lucio, Oscar**; Le, Tuan; Beese, Carsten Jörn; Wichard, Jörg; Rouquié David & Clevert, Djork-Arné

(2022). Cell morphology-guided *de novo* hit design by conditioning GANs on phenotypic image features. *Digital Discovery*, 2023,2, 91-102.

- **Le, Tuan**; **Cremer, Julian**; Clevert, Djork-Arné & Schütt, Kristof (2024). Latent-Guided Equivariant Diffusion for Controlled Structure-Based De Novo Ligand Generation. *1st Machine Learning for Life and Material Sciences Workshop at ICML 2024*
- **Bertolini, Marco**; **Le, Tuan** & Clevert, Djork-Arné (2025). Generative Modeling on Lie Groups via Euclidean Generalized Score Matching. *Arxiv preprint*

Contents

Chapter 1

Introduction

Drug design is a very long and expensive process involving extensive experiments and multiple stages, taking over 10-15 years with an average cost of over 1-2 billion USD for each new drug to be approved for clinical use as a new medicine to patients (Paul et al., 2010; Wouters et al., 2020). To accelerate the drug discovery process and increase the success rate of compounds, the development of machine and deep learning models has been a major research focus in recent years to explore and navigate the vast drug-like chemical space of approximately 10^{60} molecules (Polishchuk et al., 2013) more efficiently. To explore the chemical space, a multi-objective search is usually performed, while several *in silico* models are used to score molecules concerning their pharmacological properties and to narrow down the search space.

Graph representation learning has emerged as a powerful and flexible tool in machine and deep learning gaining popularity in scientific domains such as chemistry and structural biology for drug discovery. Representing molecules as graphs has the advantage that local structures, such as the interaction between bonded or non-bonded atoms can be encoded, enabling one to learn more expressive and distinctive features to build a molecular descriptor that can be leveraged for several tasks. Such tasks commonly found in computational chemistry involve developing of methods for molecular property prediction, such as internal energy, its synthetic accessibility, or the binding affinity to a biological target. However, computers require the graph of molecules to be described with ordered lists, such as the adjacency matrix, the chemical elements, and, if spatial coordinates are available, the 3D conformation of the molecule. This poses challenges for some tasks, e.g., the prediction of internal energy, the ordering of the data, and the molecule's orientation in Cartesian coordinates is irrelevant, i.e., invariant, for the task at hand. These aspects are related to the *symmetry* of data, and building neural network architectures that are symmetry-aware, i.e., equivariant, enables practitioners to extract richer representation from the data while being model and data-efficient. Equivariant architectures are also required

and beneficial in a *de novo* molecule design setting since generating molecules in 3D naturally includes an arbitrary ordering and orientation that requires their preservation.

In the following, we describe how expressive molecular representations can be extracted using neural network architectures suitable for either molecular property prediction or *de novo* molecule design while preserving the symmetry of the data. We study the expressiveness of molecular descriptors by learning mappings between them, where the descriptors are constructed from the molecular graph. For this reason, we use graph neural networks (GNNs) as a learning framework for molecules to discuss the published works that are part of this thesis and put them in the context of related works.

1.1 Molecular Property Prediction

The task of molecular property prediction is to develop a mathematical model that describes a mapping $f_\theta : \mathcal{X} \to \mathcal{Y}$ from a molecule $x \in \mathcal{X}$ to a set of properties $y \in \mathcal{Y}$. The properties can combine various modalities, spanning discrete, integer, or continuous-valued observables/endpoints. In the field of cheminformatics, this task is often described as *quantitative structure-activity/property relationship* (QSAR/QSPR) modeling, which aims to find the relationship between molecular structure and property, such as the biological activity of a molecule towards a biological target of interest to regulate its mechanism in a known disease pathway. However, the hit identification of an active compound does not necessarily make it an attractive candidate for drug development. In fact, the estimation of a compound's absorption, distribution, metabolism, excretion, and toxicity (ADME/T) profiles is nowadays conducted in the early discovery stage to reduce attrition rates of potential drug candidates as they progress through development (Kassel, 2004; Shih et al., 2018; Göller et al., 2020). The labels $y \in \mathcal{Y}$, can be either obtained through (high-throughput) experiments in the lab or computational methods from first principle, where the latter is often time-consuming and expensive. By learning f_θ through supervision on acquired ground-truth labels y, the goal is to apply the property model f_θ on new molecules *in silico*, circumventing the usage of physical experiments, e.g., *in vivo* ADME assays. At this point, it is important to mention that the acquisition of labels is very often dependent on the context, e.g., *in vivo* experimental settings in the laboratory, which are not contained in the molecule structure itself. Therefore, building predictive models that define a mapping solely from a molecule to properties is very challenging in practice and requires more input in the form of metadata (Bender and Cortes-Ciriano, 2021).
To build f_θ a crucial component is the chosen representation of molecules $x \in \mathcal{X}$ that the mathematical model can process, which we will discuss in Section 1.2.

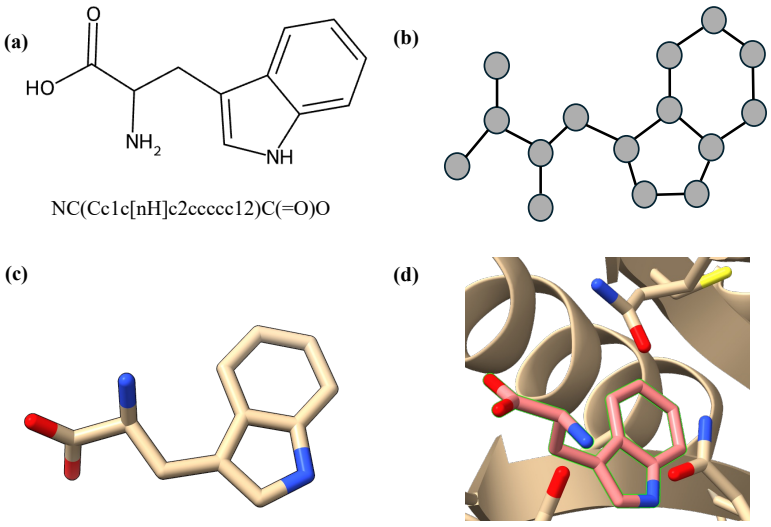


Figure 1.1: Molecular representations of Tryptophan. (a) Lewis/Kekulé 2D structure of molecule with its SMILES representation below. (b) Molecular graph with nodes and edges. (c) 3D conformer of molecule with atomic (xyz)-coordinates and bonded atoms. (d) 3D conformer (shown in light red) placed next to a protein, forming a complex.

1.2 Molecular Representations

Multiple ways exist to represent molecules in a computer-readable format for property prediction or analysis. The chemical formula of a compound, e.g., C_2H_6 for ethane, is probably the first way students encounter molecules in chemistry, followed by the Lewis/Kekule notation, where molecules are drawn as chemists envision them, indicating an abstract representation of a 2D graph, where atoms are displayed as nodes and bonds present the connectivity as illustrated in Figure 1.1a.

The choice of molecular representation often depends on the task in classical QSAR modeling and practitioners have usually relied on fixed molecular descriptors or fingerprints (Bender and Glen, 2004). Molecular descriptors are usually composed of selected physicochemical properties such as the molecular weight, the number of aromatic rings or topological indices, and geometric arrangements that are computable based on the 2D molecular graph or its conformation in 3D space (Katritzky and Gordeeva, 1993), for example as implemented in the PaDEL or Mordred -descriptors (Yap, 2011; Moriwaki et al., 2018). While such an approach leads to a fixed-sized representation suitable for standard machine learning (ML) algorithms, such as linear regression, random forest, or support vector machines, this approach also has limitations. The descriptor is restricted in the choice based on problem-specific user expertise, potentially biasing the

outcome of model accuracy in the case of molecular property prediction. As an alternative fixed representation, molecular fingerprints are often used, representing the presence or absence of particular substructures or other features in a bit or integer array. Among those commonly used are key-based fingerprints and circular fingerprints (Kassel, 2004; PubChem, 2009; Glen et al., 2006).

Extended-Connectivity Fingerprint (ECFP) The ECFP by Rogers and Hahn (2010) is a modified version of the Morgan algorithm (Morgan, 1965) which was initially designed as a graph isomorphism test for molecules to distinguish whether two molecular graphs are different. The ECFP is obtained by first assigning integer identifiers to atoms, initially collected into a fingerprint list. Initial integers are obtained from daylight atomic invariants. Next, each atom gathers its own integer identifier and those of its immediate local neighbors. A hash function is applied to compress this array into a new integer identifier. This process repeats for a predefined number of iterations. Once completed, duplicate identifiers in the list are removed, leaving behind a list of unique integers that define the ECFP fingerprint for the case of the binary ECFP. For count-based ECFP, duplicated integers are kept. The fingerprint is called circular because local neighborhoods for each atom are iteratively processed through several steps, increasing the receptive field, i.e., radius, for each atom. To obtain a computer-usable array of length k, the integers in the fingerprint list are folded into a fixed-sized array. The populating index is obtained by modulo dividing % each integer identifier by k, potentially causing bit collision because several different integers are mapped onto the same index. The mechanism of bit collision induces an information loss since for smaller sizes k, the initial fingerprint lists of two different molecules might fold into the same ECFP binary fingerprint as analyzed in Publication 1 listed in Section 2.1.

Simplified Molecular Input Line Entry System (SMILES) The SMILES representation introduced by Weininger (1988) describes a line notation of the 2D molecular graph into a 1D string of alphanumeric characters. Constructing a SMILES string involves sequentially traversing the connectivity of a 2D molecular graph from a starting point so that each atom is visited exactly once. For rings, a bond is temporarily broken and labeled with a number to denote closure, while branch points with their atoms are enclosed within parentheses (Weininger, 1988). This structured approach allows for representing complex molecular structures, including nested branches, in a linear format suitable for various chemical applications, as shown at the bottom of Figure 1.1a. Since the SMILES representation is obtained from an arbitrary starting atom followed by traversal of the molecular graph, the representation is not unique for a molecule. Depending on the computational chemistry toolkit, such as RDKit or OpenBabel, various canonicalization algorithms exist (Landrum et al., 2024; O'Boyle et al., 2011).

Instead of relying on a fixed representation of molecules, deep neural networks process molecules in some raw format, e.g., SMILES- or graph-based to *learn* a hidden representation of the molecule tailored to the particular endpoint(s)

that the network is meant to predict correctly. Deep neural networks are usually optimized using gradient-descent in the backpropagation algorithm (LeCun et al., 1998, 2015; Schmidhuber, 2015). Convolutional neural networks (CNNs) (Fukushima, 1980; LeCun et al., 1995) consecutively apply local filter banks of varying size to process neighboring pixels in a (high-dimensional) image to obtain a spatially reduced hidden representation suitable in e.g., predicting the correct image label. The *inductive bias* of CNNs lies in the locality assumption, which states that neighboring pixels are more related to each other than distant. While images can be regarded as 2D data with RGB values for each pixel location, 1D sequential data, including time-series, natural language, or amino acid sequences, which all encompass a natural order, can be processed by recurrent neural networks (RNNs) (Hopfield, 1982; Rumelhart et al., 1986; Hochreiter and Schmidhuber, 1997). These networks implement a hidden memory when processing the sequential data.

The pioneering work by Segler et al. (2018) proposed an RNN-based generative model for molecules that operate on the (canonical) SMILES representation with an LSTM network (Hochreiter and Schmidhuber, 1997). The model is trained through a maximum-likelihood objective predicting the next SMILES token given the previous sequence of tokens in an auto-regressive manner adapted from language models. The training is performed on a large corpus of molecular data extracted from the Chembl database. Focused library design is achieved by fine-tuning the generative SMILES RNN model on a subset of compounds for transfer learning. Meanwhile, Olivecrona et al. (2017) proposed to leverage reinforcement learning for property-specific molecule design.

Concurrently, Gómez-Bombarelli et al. (2018) used the variational autoencoder (VAE) model (Kingma and Welling, 2014) coupled with the (canonical) SMILES representation to obtain a generative model with latent space that can be used for multi-parameter optimization (MPO). In the SMILES VAE, the encoder maps the discrete SMILES string into a continuous representation, while the decoder takes this encoding as input to reconstruct the original SMILES. While common VAEs are trained by the reconstruction loss from the encoder-decoder architecture next to a Kullback-Leibler prior regularization loss to enforce a smooth latent space, Gómez-Bombarelli et al. (2018) trained property predictors on the latent space to correlate the encodings towards target properties of interest, e.g., the quantitative estimate of drug-likeness (QED).

Winter et al. (2019) builds open the work of Gómez-Bombarelli et al. (2018) and proposes a sequence-to-sequence autoencoder for molecules with the difference that the autoencoder is tasked to translate between string-based representation of molecules, e.g., translating between equivalent molecule representation such as the InCHI representation to canonical SMILES representation. The molecule embedding is called *continuous data-driven descriptor* (CDDD). Winter et al. (2019) show that their machine-learned molecular descriptor outperforms hand-crafted descriptors when taken as input to train support vector machines on classification or regression-based QSAR tasks. From a representation perspective,

the CDDD space indicates molecular expressivity that captures the essence of the input, as the translation task, e.g., from InCHI representation to canonical SMILES representation, also maintains high accuracy. This suggests that little information loss occurs during translation, where both encoder as well as decoder networks play a crucial role.

As mentioned at the beginning of this section, a molecular descriptor is a computer-readable abstraction and mathematical model of a molecule that, depending on the objective, should be as informative enough to solve user-defined tasks, such as QSAR modeling, virtual screening and similarity search within a database of compounds. Hence, a dataset of molecule descriptors can be regarded as *tabular* data, where ECFPs have been a popular choice for QSAR, ADME/T, or reaction prediction tasks (Göller et al., 2020; Mayr et al., 2016; Wei et al., 2016) using multilayer perceptrons (MLPS). The advantage of MLPs lies in being structure agnostic, that is, no particular assumptions are needed about the input. MLPs operate by stacking fully-connected layers where each neuron from the previous layer is connected to every neuron in the next layer. Hence, no spatial structure for neuron connectivity is made, and the model can complex non-linear relationships. However, the high number of trainable parameters can lead to fast over-fitting for high-dimensional data (Zhang et al., 2017).

Publication 1: Neuraldecipher – reverse-engineering extended - connectivity fingerprints (ECFPs) to their molecular structures As indicated earlier, a molecular descriptor is a mathematical model of a molecular structure with potential information loss, depending on how the descriptor was created in the first place. For example, a simple ring count as a descriptor cannot distinguish cyclohexane C_6H_{12} from benzene C_6H_6 because both molecules contain one ring. For this reason, ECFPs have been a popular choice for analysis and have also been exchanged between research groups in academia or private sectors, such as pharmaceutical companies. Different levels of collaboration are possible, balancing privacy concerns and computational demands. Protecting sensitive data, including intellectual property, is crucial, but this can lead to higher computational overhead for tasks like authentication and encryption. Centralizing data for joint model building could optimize computation but raises privacy risks as partners may access each other's data, such as fingerprints and activity data.

In publication 1 (Le et al., 2020), we show how the folded ECFP can be reverse-engineered to reconstruct the input molecular structure. It was often thought that the ECFP is non-invertible due to the large integer co-domain of the hashing function applied during fingerprint creation, spanning $\approx 2^{32}$ values. When folded into a fixed-sized vector of length k, the fingerprint values depend on the radius r and fingerprint length k, with possible bit collision and information loss. To showcase that the exchange of ECFP endangers the loss of intellectual property, we formulate the reverse-engineering task as a supervised regression problem by learning a mapping $f_\theta : \mathcal{X} \to \mathcal{Z}$ between folded ECFP and the unsupervised

learned CDDD representation of the autoencoder by Winter et al. (2019). Since the in- and output aim to describe the same molecule in different representation spaces, we introduce Neuraldecipher, an MLP trained to learn the relation between the two representations formulated as a supervised regression task. We used the ChEMBL25 dataset (Davies et al., 2015; Gaulton et al., 2016) including 1.8M compounds to train and validate the Neuraldecipher model using cluster splits to separate the chemical spaces from training and validation sets. Our experiments show that ECFP count fingerprints are better suited to reverse-engineer the molecular structure than the ECFP bit fingerprints. This is expected as the count ECFP describes how often a substructure is present in a molecule as opposed to the bit ECFP, which states that a substructure is available. Hence, ECFP count fingerprints retain more information about the molecule, enabling better behaved learning for Neuraldecipher, since there are more unique input ECFPs mapping to distinct CDDDs. Through multiple ablation studies, e.g., varying the fingerprint length k and/or the radius r for bit and count ECFPs, we observed that larger fingerprint sizes with small radii lead to less bit collision and therefore improved reconstruction of the original molecular structure, by accurately predicting the CDDD representation, followed by using the CDDD decoder network, to retrieve the SMILES representation.

We concluded that ECFPs are partially reversible and the exchange of ECFPs among institutions or private sectors should be avoided. Our work raised awareness about the potential risk of losing intellectual property (IP) when sharing data among partners, such that consortia rather pursue federated learning approaches (McMahan et al., 2017; Zhang et al., 2021b). The MELLODDY project (Oldenhof et al., 2023; Heyndrickx et al., 2024) was the first industry-scale platform developing a global federated deep learning model for drug discovery involving 10 large pharmaceutical companies and multiple academic research labs, ensuring the confidentiality of each partner's datasets without sharing them.

In context to other related works, Kotsias et al. (2020) leverages conditional RNNs (cRNNs) to predict the SMILES representation in an end-to-end fashion using teacher-forcing when the ECFP is input as context, while Kwon et al. (2021) combines cRNNs with a genetic algorithm. Recent work by Ucak et al. (2023) uses the Transformer neural network architecture to predict the molecular structure as SMILES in an end-to-end fashion based on the structural fingerprints. While a direct prediction of molecular structure is feasible, as shown by the related works, our proposed approach has the advantage of simplicity by only regressing the machine-learned latent CDDD representation and subsequently using the frozen decoder network for SMILES retrieval, achieving comparable reconstruction accuracies.

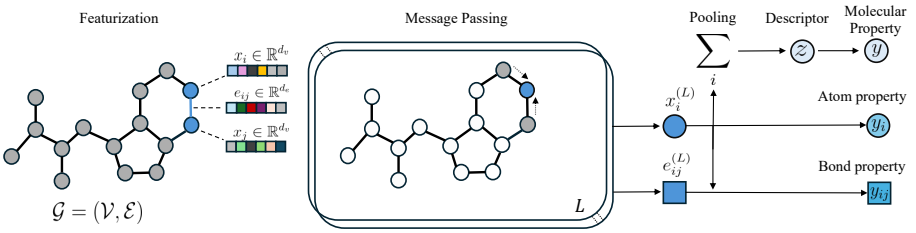


Figure 1.2: A general GNN pipeline starts with node and edge featurization, followed by L rounds of message passing. The latent node and edge embeddings $(x_i^{(L)}, e_{ij}^{(L)})$ can be used for atom or bond property prediction, or further aggregated to obtain a graph representation z suitable for molecular property prediction. Adapted from Atz et al. (2021).

1.3 Supervised 2D Topological Deep Learning

In the previous Section 1.2, we discussed the Neuraldecipher model, which reconstructs the molecular structure in terms of SMILES string given an input ECFP. Both representations have in common that they are derived from the 2D molecular graph of a chemical structure, see Figure 1.1a-b. A molecular graph is an abstract structure $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ whose N vertices ($v_i \in \mathcal{V}$) represent atoms, and whose edges ($e_{ij} \in \mathcal{E}$) describe the connectivity, e.g., through an adjacency matrix $A \in \mathbb{R}^{N \times N}$, or edge-type array $E \in \mathbb{R}^{N \times N \times k_e}$ which can describe single, double, triple or aromatic bonds. Usually, each node and edge also contain an initial feature, such as the chemical element as node- or the bond type as edge feature. The collection of initial node feature can also include multiple atom-level attributes $X \in \mathbb{R}^{N \times k_v}$, e.g., the number of attached hydrogens, the molecular weight of the atom, or an indicator of whether or not the atom is in a ring, in similar fashion to the daylight atomic invariant features used in the initialization for the ECFP.

Graph neural networks (GNNs) are formulated as functions that take a graph as input and output a representation of the graph in some feature space. The pivotal work by Duvenaud et al. (2015) introduced a neural network architecture $f_\theta : \mathcal{G} \to \mathcal{Z}$ that directly operates on the molecular graph and produces latent embeddings for each node in the molecular graph which can be aggregated to obtain a differentiable neural fingerprint optimized in an end-to-end fashion for solubility, drug efficacy or photovoltaic efficiency prediction. GNNs allow for feature extraction and operate in an iterative manner, similar to the ECFP algorithm, where each atom collects information from bonded atoms followed by an update function to obtain distinct new features when exploring the molecular environment, mimicking the ECFP algorithm.

The general framework of GNNs (Scarselli et al., 2009; Bruna et al., 2013; Defferrard et al., 2016; Kipf and Welling, 2017) falls under the umbrella of Message Passing Neural Networks (MPNNs) (Gilmer et al., 2017; Battaglia

et al., 2018). In MPNNs, each layer updates each node by aggregating messages sent from neighboring nodes, where the messages are usually computed through neural networks that input hidden node and edge features. Specifically, message and update functions are usually constructed using neural networks involving pairwise interactions to obtain intermediate messages from vertices v_j to v_i, which are aggregated/pooled together in the l-th layer as

$$\begin{aligned} m_i^{(l+1)} &= \sum_{j \in \mathcal{N}(i)} M_\theta^{(l)}(x_i^{(l)}, x_j^{(l)}, e_{ij}), \\ x_i^{(l+1)} &= U_\theta^{(l)}(x_i^{(l)}, m_i^{(l+1)}), \end{aligned} \tag{1.1}$$

where $\mathcal{N}(i)$ denotes the neighbor set of node v_i and M_θ, U_θ characterize the message and update functions. One fundamental property of most MPNNs is the equivariance or invariance with respect to permutations of the input. As the input is a (molecular) graph represented through X and A with arbitrary order along the row-index $i = 1, \ldots, N$, an action of permutation can be described with a permutation matrix $\Pi \in S_N \subset \mathbb{R}^{N \times N}$, where S_N is the symmetric group. If a permutation acts on a graph, the input representations for node- and adjacency matrices (X, A) change to

$$X_\pi = \Pi X \quad \text{and} \quad A_\pi = \Pi A \Pi^\top. \tag{1.2}$$

Permutation equivariance for a function $h : \mathcal{G} \to \mathcal{X}$ that inputs and outputs a set in the form of matrices with an arbitrary node order, e.g., the hidden embeddings for the nodes, can informally be defined as

$$h(X_\pi, A_\pi) = \Pi h(X, A). \tag{1.3}$$

This means that we can first apply function h and then permute the output, or apply h to the permuted input. The inductive bias of permutation equivariance is preserved because first, the message and update functions are shared among all nodes, and second, the aggregation function is chosen to be a permutation-invariant function such as a summation as shown in the first equation in 1.1. Furthermore, a composition of permutation equivariant functions $h_L \circ h_{L-1} \circ \cdots \circ h_0$ such as a stack of MPNN layers is again permutation equivariant.

Permutation invariance property for a function $g : \mathcal{G} \to \mathcal{Z}$ means that the output for a given set is the same regardless of the order of objects in the set, informally as

$$z = g(X_\pi, A_\pi) = g(X, A). \tag{1.4}$$

The output space is usually a lower-dimensional space independent of the graph size. A size-independent global descriptor can be accomplished through a permutation-invariant pooling function, such as summation of all hidden node embeddings from the last layer. The neural descriptor $z \in \mathcal{Z}$ can be processed with another MLP $k : \mathcal{Z} \to \mathcal{Y}$ for molecular property prediction in an end-to-end model $f : \mathcal{G} \xrightarrow{h} \mathcal{X} \xrightarrow{g} \mathcal{Z} \xrightarrow{k} \mathcal{Y}$ (Duvenaud et al., 2015; Kearnes et al., 2016; Gilmer et al., 2017; Yang et al., 2019) as illustrated in Figure 1.2.

Publication 2: Parameterized Hypercomplex Graph Neural Networks for Graph Classification While graphs fall into the domain of unstructured non-Euclidean data (Bronstein et al., 2017, 2021), generalizing images (rectangular 2D-grids) or text (1D line-graph), the hidden representations processed through a CNN, RNN or MPNN are often assumed to be embedded in the D-dimensional vector space of real numbers $\mathbb{R}^D$. Real numbers are described by one scalar value without any imaginary unit. In contrast, hypercomplex numbers, like complex $\mathbb{C}$ or quaternions $\mathbb{H}$ enrich the numerical representation with their additional imaginary dimensions (one for complex and three for quaternions) enabling them to capture more information more efficiently, e.g., in signal processing for audio or video data, where multi-dimensional signals are treated as one entity, or point cloud data, for which quaternions naturally describe rotations.

Data processing with neural networks heavily relies on multiplication when computing weighted sums, e.g., in fully-connected layers or 2D convolutions, for which complex and quaternion algebras lay down different multiplication rules, naturally enabling the interplay between the number components. For example, given two complex numbers $w = w_0 + w_1\mathbf{i}$ and $x = x_0 + x_1\mathbf{i}$, where w_0, w_1, x_0, x_1 are real coefficients and $\mathbf{i}$ describes the imaginary unit, the product reads $wx = (w_0 + x_0) + (w_1 + x_1)\mathbf{i}$, by distributivity and leveraging the fact that $\mathbf{i}^2 = -1$ with the additional property that the complex product commutes, i.e. $wx = xw$. The advantage of coefficient mixing has led to the introduction of hypercomplex-valued neural networks (Danihelka et al., 2016; Trabelsi et al., 2018; Parcollet et al., 2019; Comminiello et al., 2019; Tay et al., 2019) which enjoy parameter efficiency, e.g. up to four times in a linear layer, e.g. in the case of quaternions with the Hamilton product defined as

$$wx = \begin{bmatrix} 1 \\ \mathbf{i} \\ \mathbf{j} \\ \mathbf{k} \end{bmatrix}^\top \begin{bmatrix} w_0 & -w_1 & -w_2 & -w_3 \\ w_1 & w_0 & -w_3 & w_2 \\ w_2 & w_3 & w_0 & -w_1 \\ w_3 & -w_2 & w_1 & w_0 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} . \tag{1.5}$$

Since the Hamilton product does not commute, i.e., $wx \neq xw$ for two quaternions $w, x \in \mathbb{H}$, this property can be exploited in quaternion fully-connected (FC) layers in biasing the learning process, encouraging the model to capture interactions and relationships that are influenced by the non-commutativity. In contrast to this, a real-valued (FC) layer would include 16 degrees of freedom for the weights $\{w_i\}_{i=1}^{16}$ if x was treated as real-valued array with four components in Eq. (1.5).

Inspired by the quaternion algebra introduced by Hamilton (1844), we generalize the concept of hypercomplex layers suitable for GNNs in publication 2 (Le et al., 2021) allowing variable $(n-1)$ pseudo imaginary units. Specifically, we propose to learn the multiplication rule, which enables the interplay between the coefficients of the real and imaginary units using the sum of Kronecker/Tensor products ($\otimes$)

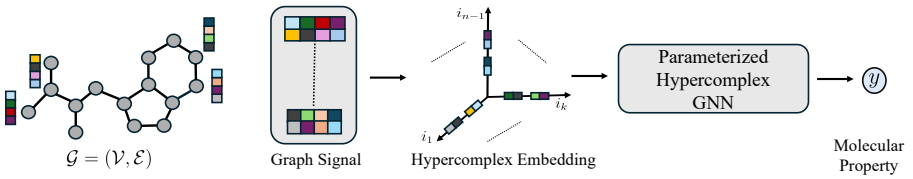


Figure 1.3: Hypercomplex GNN workflow for molecular property prediction. Multidimensional node features (here 4 channels) are treated as single hypercomplex number with one real and 3 imaginary units that are processed within the GNN that leveraged the PHM layer for linear transforming hidden embeddings.

between matrices when parameterizing the linear layer as PHM-layer defined as

$$U = \sum_{i=1}^{n} C_i \otimes W_i \ , \tag{1.6}$$

where $C_i \in \mathbb{R}^{n \times n}$ are denoted contribution matrices and $W_i \in \mathbb{R}^{\frac{k}{n} \times \frac{d}{n}}$ are the component weight matrices and k, d denote in- and output channels and the final weight matrix U lies in $\mathbb{R}^{k \times d}$. Most of the parameters are stored in the $\{W_i\}_{i=1}^{n}$ matrices, allowing up to $\frac{1}{n}$ less trainable parameters (Zhang et al., 2021a).

To show that the adaption of PHM-layer benefits in graph learning tasks, we adopt message passing functions from the graph isomorphism network (GIN) by Xu et al. (2019) due to its simplicity and expressivity and replace any linear transformation found in the MLPs of the message or aggregation function with the PHM layer whose weight matrix is constructed following Eq. (1.6) as visualized in Figure 1.3. We performed extensive studies and ablated variants of our proposed PHC-GNN on 2D molecular property prediction datasets from the OGB and BenchmarkingGNNs frameworks (Hu et al., 2020; Dwivedi et al., 2020) and observed that within the same model class with an equal number of trainable parameters, PHC-GNN with $n \geq 2$ adapting learnable hypercomplex multiplication, achieves superior performance on the Mol-HIV and Mol-PCBA compared to the real-valued PHC-GNN with $n = 1$, on the validation set created through scaffold splits. Both tasks are formulated as binary classification problems, while Mol-HIV is a single-task aiming to identify if a molecule inhibits HIV replication, and Mol-PCBA is a multi-task with 128 endpoints extracted from high-throughput screening measuring biological activity. Particularly on Mol-PCBA, which aims to predict a 'biofingerprint' indicated by the activity on 128 assays, the proposed PHC-GNN achieved SOTA results at the time of publication compared to other GNNs with sophisticated message-passsing functions, indicating stronger generalization capabilities. Since PHC-GNN enables parameter efficiency by exploiting interaction within the algebra components, i.e., real and pseudo-imaginary units, the embeddings can be richer and stronger regularized as in the real-valued case, desired in out-of-distribution settings. Since the Mol-PCBA dataset often does not contain a full 128-dimensional

bio-assay readout for a molecule, predictive models should learn a rich internal representation of a molecule to learn non-linear relations between compound and endpoints by exploiting the fact that learning on multiple tasks enhances the representation (Ramsundar et al., 2015, 2017; Montanari et al., 2020), which PHC-GNNs incorporate by design through their inductive bias of embedding composition realized in the Hamilton-like product.

Outside the realm of GNNs, parameterized hypercomplex neural networks (PHC-NNs) have been used in NLP tasks by Mahabadi et al. (2021) for fine-tuning large-scale language models with an improved trade-off between task performance and the number of trainable parameters, or for image classification or sound detection (Grassucci et al., 2022) and explainability (Lopez et al., 2024). Hypercomplex algebras including complex numbers or quaternions can be generalized into the framework of Clifford Algebras, which are defined by a basis set $\{e_1, e_2, \ldots, e_n\}$, commonly represented as geometric vectors (Clifford, 1871; Dorst and Mann, 2002). Geometric Clifford Algebra GNNs have been proposed by Ruhe et al. (2023) to model dynamical systems and partial differential equations (PDEs) as an extension to the work of Brandstetter et al. (2023) who introduced Clifford neural layers in CNNs on grid-like data for PDE modeling.

1.4 Supervised 3D Geometric Deep Learning

In the previous Section 1.3, we proposed a simple 2D GNN model for processing the topological graph containing a molecule's initial node and edge features without using 3D spatial coordinates represented in the conformer of a molecule. The representation of a molecular graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ can be extended to have a coordinate matrix $R \in \mathbb{R}^{N\times 3}$, next to the node and adjacency matrix $X \in \mathbb{R}^{N\times k_a}$, $A \in \mathbb{R}^{N\times N}$ with optional edge-feature array $E \in \mathbb{R}^{N\times N\times k_e}$. We call this graph a geometric or molecular conformer graph, as visualized in Figure 1.1c. Since atoms are embedded in 3D space, a natural question about invariance and equivariance concerning global rotations and translations for the point cloud arises. How can we design efficient neural network architectures f_θ that respect these symmetries? Many physical properties have well-defined transformation properties under translation, reflections, and rotation of a set of atoms, which can be summarized as actions from the Euclidean Group E(3). For example, if a molecule is rotated in space, the vectors of its atomic velocities or forces also rotate accordingly. On the other hand, the potential energy $y \in \mathbb{R}$ of a molecule $\mathcal{G}$ is invariant with respect to global roto-translations and permutation of atoms. In similar spirit to permutation invariance stated in Eq. (1.4), this invariance can be described as

$$y = f_\theta((Q(\Pi R)^\top)^\top + 1_N t^\top, \Pi X, \Pi A \Pi^\top) = f_\theta(R, X, A) \ , \tag{1.7}$$

where $Q \in \mathrm{SO}(3)$ is a rotation matrix, $t \in \mathbb{R}^3$ a translation vector, $\Pi \in S_N$ a permutation matrix and $1_N = (1, \ldots, 1)^\top \in \mathbb{R}^N$.

When modeling such quantity, invariance can be enforced by only operating on

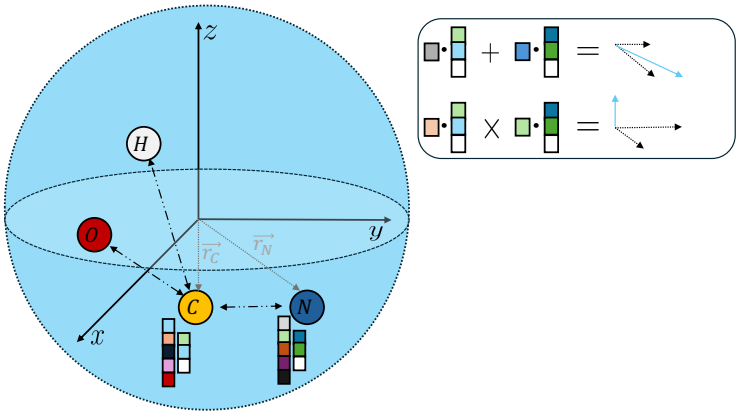


Figure 1.4: Left: Geometric graph embedded in 3D Cartesian space. The green vertex illustrated as carbon (C) atom aggregates information from the neighbouring atoms (O, H, N) that are within a radius cutoff. Scalar and vector embeddings are shown for the carbon (C) and nitrogen (N) atoms. The vectors $\vec{r_C}, \vec{r_N}$ describe the spatial coordinates of the Carbon and Nitrogen atoms. Right: Equivariant vector features can be constructed based on scalar products or using the cross product as indicated in the lower row. Output vectors are colored in light blue. The carbon and nitrogen atoms lie on the (x, y) plane and by using the vector cross product, the features on the z axis can be constructed.

invariant representation in neural architectures, e.g., in the form of distances or angles (Schütt et al., 2017; Schütt et al., 2017; Gasteiger et al., 2020) which implement MPNNs as machine learning force fields suitable for molecular dynamics simulations. Restricting the model class to invariant representations in GNN was shown to be incomplete for some geometric graphs (Garg et al., 2020; Schütt et al., 2021; Pozdnyakov and Ceriotti, 2022), potentially limiting the model expressiveness for the learning tasks. For this reason, another line of research has made use of irreducible representations (irreps) of the rotation group SO(3) through spherical harmonics combined with Clebsch-Gordan tensor products ($\otimes_{\text{cg}}$) to build E(3) equivariant GNNs (Thomas et al., 2018; Anderson et al., 2019; Fuchs et al., 2020; Batzner et al., 2022; Brandstetter et al., 2022) that are able to design higher-order equivariant features using spherical tensors. While the mathematical machinery allows these models for increased expressiveness, one disadvantage is the high computational cost of their training, resulting in slow convergence time. Furthermore, the highest order of (hidden) equivariant representation is unclear for a learning task, especially on larger biomolecular systems, where symmetric structures are less likely. As an alternative avenue of investigation, a number of works adopt a generic approach to modeling E(3)-equivariance directly on Cartesian tensors using scalarization as proposed in GVP-GNN, E(n)-GNN and PaiNN (Jing et al., 2021; Satorras et al., 2021b; Schütt et al., 2021). These works model invariant scalar (rank-0) and equivariant

vector (rank-1) through tensor products ($\otimes$) of scalars with vectors, to establish equivariant rank-1 features. While simple, this approach also applies for higher dimensions in $\mathbb{R}^n$, considering $\mathrm{SO}(n)$ as advocated by Villar et al. (2021).

Publication 3: Representation Learning on Biomolecular Structures Using Equivariant Graph Attention We propose an SE(3) equivariant GNN, termed EQGAT, in publication 3 (Le et al., 2022) which scales well on large biomolecular systems such as protein-protein or protein-ligand complexes (see Figure 1.1d), while maintaining high expressivity through rotation equivariance. We follow the scalarization approach that implements rank-0 scalar and rank-1 vector features. Distinct from other scalarized models at the time of publication, EQGAT allows the interaction between vector features by using the cross product between vectors, enabling higher expressivity, while keeping the computational cost low. Geometrically, the cross-product $c = a \times b$ between two vectors returns a vector $c \in \mathbb{R}^3$ that is perpendicular to the plane spanned by (a, b) and its computation follows

$$c = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \end{pmatrix} \times \begin{pmatrix} b_1 \\ b_2 \\ b_3 \end{pmatrix} = \begin{pmatrix} a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 \\ a_1 b_2 - a_2 b_1 \end{pmatrix} \ . \tag{1.8}$$

Methods that only operate on linear combinations of vector features are restricted when modeling this geometric aspect. For example, consider a molecule whose atomic coordinates reside in the canonical (x, y)-plane. Suppose no cross-product in the equivariant network is modeled. In that case, all hidden vectors remain in the plane, not occupying the $z-$axis, which restricts the representation capacity as illustrated in Figure 1.4 for the two carbon and nitrogen atoms. It can be shown that the cross-product contains elements from a rank-2 Cartesian tensor. A Cartesian rank-2 tensor $C = a \otimes b = ab^\top \in \mathbb{R}^{3\times 3}$ has 9 degrees of freedom and can be decomposed into

$$C = \frac{1}{2}\mathrm{Tr}(C)I + \frac{1}{2}(C - C^\top) + \frac{1}{2}(C + C^\top - \mathrm{Tr}(C)I) \ ,$$

where Tr() is the trace operation and $I \in \mathbb{R}^{3\times 3}$ the identity matrix. The three matrices each reveal 1, 3 and 5 degrees of freedom in terms of irreducible representations. It turns out that the second skew-symmetric matrix contains the elements that are computed in the cross-product $a \times b$ shown in Eq. (1.8), since the skew-symmetric matrix has the expression

$$C - C^\top = \begin{pmatrix} 0 & a_1 b_2 - a_2 b_1 & a_1 b_3 - a_3 b_1 \\ a_2 b_1 - a_1 b_2 & 0 & a_2 b_3 - a_3 b_2 \\ a_3 b_1 - a_1 b_3 & a_3 b_2 - a_2 b_3 & 0 \end{pmatrix} \ . \tag{1.9}$$

Although EQGAT only features scalar- and vector components, i.e., 1- and 3-dimensional representations, by using the cross-product, the model extracts (vector) components that are present in a rank-2 Cartesian tensor. Next to the geometric interpretation, this argument provides an additional algebraic

reason for enhanced expressivity when including the cross-product. However, it is important to note that the output of the cross product undergoes an additional sign flip in case the transformation is an improper rotation matrix with determinant -1, e.g., a reflection, which must be considered when the goal is to implement an E(3)-equivariant GNN, where the quantity to predict is sensitive to reflections.

To regulate the information flow for scalar and vector features in the message passing layers, we propose an attention-based mechanism that leverages invariant and equivariant features to obtain non-linear filters suitable for the final prediction task. The proposed EQGAT model was trained and evaluated on the Atom3D benchmark (Townshend et al., 2021) on molecular property prediction tasks that are permutation-, rotation-, and translation invariant. We compared our model against modern equivariant GNNs, including GVP-GNN (Jing et al., 2021), PaiNN (Schütt et al., 2021), which are both operating on rank-0 and rank-1 equivariant features by means of scalarization. Furthermore, we compared EQGAT against SEGNN (Brandstetter et al., 2022) up to rotation order $l = 2$, which is a recent equivariant GNN that operates on irreducible representations of SO(3) employing spherical harmonics and Clebsch-Gordan tensor products to enable the interaction between various equivariant features. We demonstrate the expressiveness of EQGAT by outperforming PaiNN, GVP-GNN and SEGNN in 4 out of 5 tasks, where GVP-GNN is an E(3)-equivariant GNN designed for processing biomolecular structures while PaiNN and SEGNN are specifically designed to model smaller dynamical systems. We ablate the design choices of EQGAT and remove the vector cross-product or attention-based filters in the message passing layers to investigate their effects. The ablation studies show the best performance for the full model, when evaluated on synthetic data as well as the LBA (Ligand-Pocket Binding Affinity) and PSR (Protein-Structure ranking) tasks from Atom3D.

Related work by Morehead and Cheng (2024) also implements the vector cross product but on raw input coordinates in the message passing layer of their equivariant GNN as opposed to EQGAT, which implements the vector cross-product in the hidden equivariant vector features. Similar to our ablation study, they identify improved model performance when the cross-product is included in their model architecture.

While EQGAT was proposed as an SE(3)-equivariant graph encoder network for molecular property prediction on biomolecules, its architecture is also applicable to smaller molecules, including ligands or peptides. In the next section, we discuss how geometric graph representation learning can be used for the targeted generation of small molecules that satisfy multiple user properties of interest.

1.5 Generative Models

Geometric GNNs that preserve E(3)-equivariance have shown to be powerful data encoders (Zhang et al., 2023) while new architectures have been developed with the motivation to better represent the molecular environment in a data-efficient manner for supervised learning tasks. Although developed for discriminative tasks, these expressive encoders also facilitate the generation of new complex geometric structures, including small molecules, commonly known as *de novo* molecule generation. Depending on the data representation, some works only generate atomic coordinates $R \in \mathbb{R}^{N\times 3}$ and chemical elements from a chosen subset of the periodic table, e.g., heavy atoms including carbon, nitrogen, oxygen, and chlorine represented as one-hot encodings in $X \in \mathbb{R}^{N\times k_a}$. Once the atomic coordinates have been generated, external software such as OpenBabel (O'Boyle et al., 2011) can be used to infer the 2D molecular graph, completing the data representation $M = (X, R, A)$, originating from the probabilistic model $p_\theta(M)$.

One class of 3D generative models for unconditional *de novo* molecule design utilized autoregressive models trained through maximum likelihood objective (Gebauer et al., 2019; Luo and Ji, 2022), which were further adapted for property conditioned molecule generation (Gebauer et al., 2022). These models define an order-dependent factorized probability distribution from which atoms with their coordinates and chemical elements are sampled sequentially. Models related to sequential sampling can also be framed within Reinforcement Learning (RL) as a sequential decision-making problem, in which an agent interacts with an environment to maximize a reward (Simm et al., 2020, 2021).

1.5.1 Diffusion Models for 3D Molecule Generation

An alternative category of generative models are transport-based methods, including continuous normalizing flows (CNFs) (Chen et al., 2018) or denoising probabilistic diffusion models (DDPMs) (Ho et al., 2020). These models are trained through maximum likelihood objective and iteratively generate the full atomic system by traversing a reverse (learned) dynamics in a latent space to map from a prior distribution to the data distribution. CNFs are known to be computationally expensive to train since they involve the integration of an ordinary differential equation (ODE) to compute the entire likelihood of a datapoint, making them difficult to scale on larger molecular structures (Satorras et al., 2021a). DDPMs, instead, are favored due to their efficient and scalable training objective by optimizing a component of the likelihood bound to approximate the generative diffusion process.

Diffusion models have been applied for 3D molecule design by (Hoogeboom et al., 2022) in the EDM model, which jointly generates atomic coordinates and their chemical elements using Gaussian diffusion (Sohl-Dickstein et al., 2015). In essence, diffusion models consist of two components. The first is the tractable forward inference (noising) model q that perturbs the data point to a corrupted version, while the second component is the denoising model p_θ, tasked

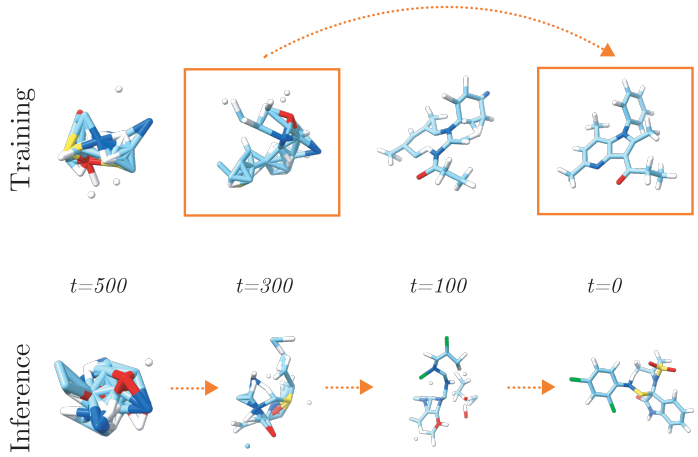


Figure 1.5: Training of 3D diffusion model is achieved by predicting the original data point based on a noisy datapoint. During inference, the molecule is sampled from a prior distribution and refined in multiple steps, here $T = 500$ total steps.

in reversing the process, e.g., by predicting the original data point or as an alternative, the noise added during the corruption, as visualized in the top row in Figure 1.5.

Diffusion models maximize a variational lower bound of the data likelihood, which can be decomposed through multiple loss terms as

$$\log p_\theta(x) \geq L_0 + L_{\text{prior}} + \sum_{t=1}^{T-1} L_t \ ,$$

where $L_t = -D_{\text{KL}}\left(q(x_{t-1}|x_t, x_0)|p_\theta(x_{t-1}|x_t)\right)$ is the diffusion loss, while the first two terms are reconstruction and prior loss (Luo, 2022). During training, instead of optimizing all timesteps, it is common practice to sample a timestep $t \sim U(1, T)$ and optimize the diffusion loss L_t, which exhibits a closed-form since $q(x_{t-1}|, x_t, x_0)$ is tractable. Therefore, diffusion models are compared to continuous normalizing flows more efficient to train, since diffusion models are *simulation-free* and intermediate points $x_t \sim q(x_t|x_0)$ are easy to sample due to construction of the forward noising process.

At inference, a random sample is obtained from a Gaussian prior, e.g., $x_T \sim N(0, I)$, and a less noisy version sampled through denoising model $p_\theta(x_{t-1}|x_t)$, as visualized at the bottom row in Figure 1.5 for atomic coordinates. As indicated, the diffusion loss is perfectly optimized if $p_\theta(x_{t-1}|x_t)$ matches $q(x_{t-1}|x_0, x_t)$, that is, iff. the model can predict the original data $\hat{x}_0$ accurately, and less corrupted (latent data) points can be sampled through

$$x_{t-1} \sim p_\theta(x_{t-1}|x_t) = q(x_{t-1}|\hat{x}_0 = f_\theta(x_t), x_t) \ . \tag{1.10}$$

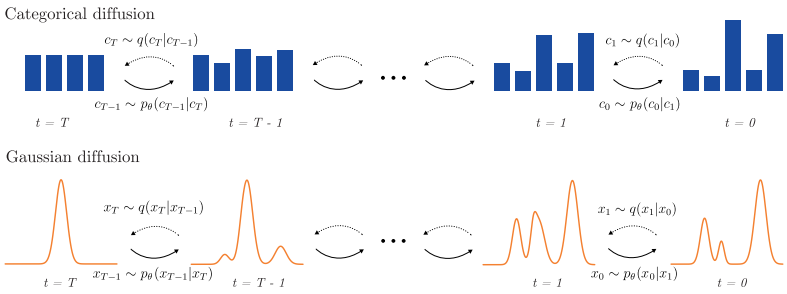


Figure 1.6: Categorical and Gaussian diffusion with forward transitions q and learnable reverse transitions p_θ.

The intuition with the reverse process is to move into the direction of state space that is more likely under the data distribution.

An essential requirement of diffusion models for 3D molecules is the equivariance with respect to permutation on the atoms as well as global roto-translations, which has to be baked in when designing $f_\theta : M \to M$.

Publication 4: Navigating the Design Space of Equivariant Diffusion-Based Generative Models for De Novo 3D Molecule Generation Diffusion models have become a very popular research field in the image generation community (Karras et al., 2022) and have been adapted to molecule design to correctly handle the symmetries of the data while applying common training strategies from the image domain. In the fourth publication (Le et al., 2024), we propose an equivariant diffusion model, termed EQGAT-diff, that generates the entire 3D molecule, including bond topology in an iterative manner through multiple denoising steps as shown in the lower panel in Figure 1.5. We use the EQGAT architecture proposed by Le et al. (2022) and modify it slightly so it is suitable as a denoising neural network. Since the model operates on multiple data modalities, including product spaces of continuous positions, discrete atom and bond types, we systematically explore the design choices for 3D molecular diffusion models through novel components and analyze their effects through extensive ablation studies, while keeping the EQGAT-diff architecture fixed.

We demonstrate that 3D molecule diffusion models achieve faster convergence and better generation quality when using our proposed loss weighting. As the diffusion model can be trained by predicting the original data point, i.e., the 3D coordinates, atom, and bond types, our weighting amplifies the learning signal for reconstructing the original molecule from a perturbed version when the timestep is closer to the data distribution. In contrast, the loss signal for timesteps closer to the prior states is decreased. Furthermore, we show that respecting the data modalities and applying categorical diffusion for discrete data modalities, see

Figure 1.6, such as atom and bond types led to better sample quality when EQGAT-diff was evaluated on the QM9 and GEOM-Drugs benchmark datasets with explicit hydrogens, overall reaching superior performance over the MiDi architecture (Vignac et al., 2023) which at the time of publication was the state of the art. Because EGAT-diff also operates on the bond graph of a molecule, no external software such as OpenBabel or empirical bond distances from a lookup table is required to infer the edge types for the molecule compared to previous works (Hoogeboom et al., 2022). Including bond diffusion also results in improved generation quality in terms of molecular stability, which involves the sanity check that correct chemical valencies for bonded atoms are preserved.

Because diffusion models can learn the data distribution they were trained on, we further show that this model class is suitable for learning distribution shifts. Since structural molecular data with a high level of theory are sparse and computationally expensive to produce, we created a 3D dataset with a lower level of theory from PubChem3D for pre-training the EQGAT-diff architecture. The rationale behind pre-training lies in building a strong base model that has learned general chemistry from a vast space involving 95.7M samples. We show that pre-training greatly expedites model convergence when fine-tuned on benchmark datasets like GEOM-Drugs, even when a subset of the original dataset is used for training.

Equipped with the findings from our study, we demonstrate that these also hold when EQGAT-diff is trained in a protein-ligand (PL) complex, where the protein pocket is fixed as a condition, and the diffusion model is tasked with generating a small molecule into the pocket, paving the way for target-aware ligand design. The pre-training strategy proposed on large-scale 3D molecular data notably accelerates EQGAT-diff's convergence. It enhances evaluation metrics, including high diversity and low docking scores, when applied to PL complexes, indicating a significant distribution shift.

Other related work on 3D molecule generation with diffusion models implements local and global encoder networks to differentiate close and distant interactions that can help optimize the denoising objective as in MDM (Huang et al., 2023). Apart from the *de novo* 3D molecule design, where an entire structure is generated from scratch, Igashov et al. (2024) proposed DiffLinker, a fragment-based 3D equivariant diffusion model that generates linkers between disconnected fragments to form the product molecule. Instead of training diffusion models from scratch, others leveraged pre-trained diffusion models and modified the reverse sampling process for conditioned molecule design, either through inpainting, which fixes atomic components of a molecule during generation (Runcie and Mey, 2023), or gradient guidance (Bao et al., 2023; Weiss et al., 2023) to steer the generation of molecules that satisfy defined properties. Guidance is a significant advantage of diffusion models. It enables iterative adjustments to steer sampled data towards desired properties. This is achieved by modifying the probability distribution of the sampled space without needing to retrain the diffusion model.

1.5.2 Structure-Based Drug Design

Structure-based drug design (SBDD) is a fundamental task in drug discovery and focuses on developing valid ligands that display robust binding affinity and specificity for a particular receptor protein pocket using the 3D structure (Anderson, 2003). While the ligand should be tailored to fit the binding site (e.g., as shown in Figure 1.1d), other essential chemical properties such as synthesizability and drug-likeness must be considered in the design stage. This can be regarded as a multi-objective sampling problem over a vast chemical space. A standard workflow in SBDD consists of two stages, including *screening* and *scoring*. During the first screening phase, a protein target and its binding site, i.e., the pocket, are selected, and an enumerated search over a large database of ligands is performed to find promising candidates. The second scoring phase involves high-throughput experimental techniques or computational methods such as molecular docking or free energy perturbation (FEP) (Lionta et al., 2014). This conventional virtual screening encounters several challenges because experimental and in-silico methods, especially FEP, are time-consuming and computationally expensive. Furthermore, the enumerated search over a chemical library does not allow the creation of new chemical matter and suggests *de novo* ligands.

Recent advances in machine learning, especially in generative modeling, offer a computationally efficient alternative to traditional Structure-Based Drug Design (SBDD) methods. These innovations address the limitations of extensive ligand screening databases used in conventional SBDD. Deep generative models start with the protein pocket and design ligands from scratch by modeling the underlying distribution of ligand-protein pairs. Early work by Ragoza et al. (2022) employ variational autoencoders with 3D convolutional neural networks (3D-CNNs) in voxel space to encode the atomic density grids of protein-ligand complexes into a latent space that can be used for sampling novel ligands. A similar approach was pursued by Green et al. (2021), which focused on fragment-based ligand optimization. Unlike VAEs, Wang et al. (2022) combined 3D CNNs in voxel space on density grids, changed the generative algorithm and trained generative adversarial networks (GANs) end-to-end. Since voxelized grid representations are large and have sparse values, 3D CNNs suffer from high memory consumption. Treating protein-ligand complexes as atomic point clouds can circumvent this problem and, combined with graph neural networks, enable the generative modeling of ligands bound to protein pockets. SBDD with autoregressive models were used in combination with SE(3)-invariant GNNs (Luo et al., 2021; Liu et al., 2022) or SE(3) equivariant GNNs such as in Pocket2Mol (Peng et al., 2022), which places individual atoms one after the other during generation. Autoregressive models can suffer from error accumulation and require an order to sample new atomic points in 3D space sequentially. On the contrary, diffusion models have the advantage of generating entire molecules in one shot but allow for iterative refinement of the entire structure through successive denoising steps as proposed in models like DiffSBDD (Schneuing et al., 2023) and TargetDiff

(Guan et al., 2023b).

Publication 5: PILOT: Equivariant diffusion for pocket conditioned de novo ligand generation with multi-objective guidance via importance sampling While diffusion models are able to generate ligands conditioned on a protein pocket, further constraints are necessary to find a suitable ligand. Often, the binding affinity or selectivity on a protein target is insufficient since criteria such as synthesizability, pharmacokinetics, and toxicity play a vital role in narrowing down the search space of possible drug candidates to reduce failure cases at later stages. Since diffusion models often only learn a distribution of molecules conditioned on protein pocket, i.e., $p_\theta(M|P)$, real-world applications require the generation of ligands that satisfy multiple criteria $c \in \mathcal{X}$ that can be ligand M or ligand-pocket (M, P) dependent such as synthesizability or the docking score. In publication 5 (Cremer et al., 2024), we leverage the EQGAT-diff architecture from publication 4 (Le et al., 2024) and propose a sampling algorithm to generate pocket-conditioned ligands that also reveal favorable properties, i.e., $M \sim p_\theta(M|P, c)$ through importance sampling guidance. This is achieved by leveraging Bayes' theorem and decomposing the probability into

$$p_\theta(M|P, c) \propto p_\theta(M|P)p_\delta(c|M, P), \tag{1.11}$$

and using surrogate models p_δ that can score molecules based on the property of interests c.

First, we show that our base EQGAT-diff model with additional modification achieves SOTA performance in unconditional ligand generation on the CrossDocked dataset, with higher molecule validity and improved geometries regarding the training set compared to TargetDiff, the SOTA model at the time of publication. While the amount of protein structures has increased over the years, surpassing 200K experimental protein structures, the amount of complexes with bound ligands in a protein combined, including binding affinity data, is smaller as in the PDBBind v2021 database with approximately 22.9K PL complexes (Wang et al., 2004). Therefore, a diffusion model trained from scratch only on these PL complexes is limited in the chemical space it has observed. To circumvent this, we pre-train our diffusion model on the Enamine Real Diversity subset, including 48.2M small molecules represented as SMILES, grounded by the fact that the Enamine subset contains synthesizable molecules and fall into the category of drug-like compounds. Similarly to publication 4 (Le et al., 2024), we observed faster convergence for the fine-tuned EQGAT-diff model on the CrossDocked dataset and enhanced evaluation metrics, including molecule validity that checks for correct valencies, reduced number of disconnected components, and better bond and angle distributions with respect to the training set. Based on this, the fine-tuned diffusion model can serve as a strong basis for conditioned ligand generation in the PILOT method.

Next, we condition on molecular synthesizability, the docking score of a ligand with respect to a protein target, and pIC50. We demonstrate that we can

efficiently bias the generation of ligands with both high synthesizability and low docking scores compared to a set that was sampled unconditionally without the importance sampling guidance. The main idea of filtering for samples that maximize/minimize the selected properties during the reverse diffusion trajectory is inspired by sequential Monte Carlo and evolutionary algorithms. Guidance is in our approach achieved by sampling intermediate latent data points with replacement from a finite set of points, i.e., a population, and evolving them further using the standard reverse diffusion model as stated in Eq. 1.10. This offers an efficient alternative to, e.g., gradient-based classifier guidance (Dhariwal and Nichol, 2021), which requires backpropagation through another surrogate model to compute spatial gradients that describe the direction in continuous state space to maximize a property. Compared to gradient-based guidance, our proposed importance sampling method achieves superior performance when given the same computational budget.

By using an ensemble of surrogate models, we further demonstrate that we can guide the generation of ligands that exhibit high predicted pIC50, a measure of the binding affinity of a ligand to a protein target, in this case, kinases utilized from the Kinodata-3D dataset. We observe that using only one surrogate model for guided sampling can introduce bias in the importance weights if that model is not well-calibrated. Therefore, ensemble models have the advantage of enhanced generalization through lower calibration error and reduced variance, as we also show in the paper. The test set generalization of the ensemble model is lower than the average test set error for each single model. With the PILOT architecture, we showcase how diffusion pre-training and the importance sampling algorithm can efficiently generate property-conditioned ligands by querying a surrogate model that acts as an oracle. While we use simple properties, the method is applicable to other properties as long as labeled data is available to train the surrogate models, which are not required to be differentiable.

Other related works proposed diffusion models trained from scratch for SBDD with structured priors for scaffold or linker design (Guan et al., 2023c,a), which both include a gradient-based guidance term avoid steric clashes with the protein pocket during the reverse sampling process. Another line of work leverages pre-trained diffusion models and proposes the use of classifier guidance to perform optimization during the reverse trajectory on the latent space motivated by constraining the ligand atoms to maintain pharmacophore features while avoiding steric clashes by Ziv et al. (2024) or use external differentiable surrogate models combined with the ADAM optimizer to perform gradient-based search in the latent space to improve docking and SA scores by Kadan et al. (2024) similar to our proposed PILOT architecture.

Chapter 2

Publications

2.1 Publication 1: Neuraldecipher – reverse-engineering extended-connectivity fingerprints (ECFPs) to their molecular structures

Full Reference: ***Le, Tuan****; Winter, Robin; Noé, Frank; & Clevert, Djork-Arné (2020). Neuraldecipher–reverse-engineering extended-connectivity fingerprints (ECFPs) to their molecular structures. Chemical science, 2020, 11(38), 10378-10389.*

DOI: 10.1039/D0SC03115A

Journal/Conference: Chemical Science

Source Code: `https://github.com/bayer-science-for-a-better-life/neuraldecipher`

Paper's main contributions:

- We analyze the information loss induced by folding ECFPs to fixed length vectors with varying radii.
- We propose a neural network architecture, a fast method, that predicts the molecular structure based on folded extended connectivity fingerprints.
- We evaluate to what extend molecular structures can be reverse-engineered based on the folding settings, raising awareness that exchange of molecule databases in terms of ECFPs should be considered thoroughly in agreement.

Author's contribution to the paper:

- Conceptualization of the original idea and its application to molecular representation learning.
- Development of the methodology and implementation.
- Design and evaluation of experiments, data curation and analysis.
- Preparation and creation of initial draft and visualizations.
- Revision of manuscript during the review phase at journal.

The manuscript was written by TL. RW provided access to the CDDD network and the inference code. The work was supervised by FN and DAC.

Chemical Science

ROYAL SOCIETY OF CHEMISTRY

EDGE ARTICLE

View Article Online
View Journal | View Issue

Check for updates

Cite this: *Chem. Sci.*, 2020, **11**, 10378

All publication charges for this article have been paid for by the Royal Society of Chemistry

Neuraldecipher – reverse-engineering extended-connectivity fingerprints (ECFPs) to their molecular structures†

Tuan Le, *[ab] Robin Winter, [ab] Frank Noé [b] and Djork-Arné Clevert *[a]

Received 3rd June 2020
Accepted 10th September 2020

DOI: 10.1039/d0sc03115a

rsc.li/chemical-science

Protecting molecular structures from disclosure against external parties is of great relevance for industrial and private associations, such as pharmaceutical companies. Within the framework of external collaborations, it is common to exchange datasets by encoding the molecular structures into descriptors. Molecular fingerprints such as the extended-connectivity fingerprints (ECFPs) are frequently used for such an exchange, because they typically perform well on quantitative structure–activity relationship tasks. ECFPs are often considered to be non-invertible due to the way they are computed. In this paper, we present a fast reverse-engineering method to deduce the molecular structure given revealed ECFPs. Our method includes the Neuraldecipher, a neural network model that predicts a compact vector representation of compounds, given ECFPs. We then utilize another pre-trained model to retrieve the molecular structure as SMILES representation. We demonstrate that our method is able to reconstruct molecular structures to some extent, and improves, when ECFPs with larger fingerprint sizes are revealed. For example, given ECFP count vectors of length 4096, we are able to correctly deduce up to 69% of molecular structures on a validation set (112 K unique samples) with our method.

1 Introduction

The data protection and privacy of molecular structures are of crucial importance for industrial and private sectors, especially for the pharmaceutical industry. As the process of drug discovery is known to last at least a decade (10–20 years),[2,3] pharmaceutical companies have utilized computational methods in the early stage to accelerate the generation of promising drug candidates that are active against a biological target, and the enrichment of chemical libraries for subsequent screening and analysis.

Molecular descriptors and fingerprints play a central role in computer-aided drug discovery, *i.e. in silico de novo* drug design, as they capture chemical information of the molecular structure as a vector of numbers that can be utilized for predictive modeling in several cheminformatic tasks.[4] In quantitative structure–activity (QSAR) modeling, the aim is to model the relationship between compound and biological or physico-chemical endpoints. One biological endpoint is usually the binding affinity of a drug candidate against a protein target. Because drug candidates with high binding affinity can still fail in later phases of clinical trials due to poor pharmacokinetic and toxicological (ADMET) profiles, modeling ADMET endpoints such as solubility or melting point, is nowadays also considered in *in silico de novo* drug design at early stages.[5]

Securely exchanging chemical data without revealing the molecular structure is especially nowadays of great importance, as sharing data such as fingerprints and/or measured endpoints between research groups within academia or private sectors through collaborations is often accomplished to improve drug discovery.

An example for a large-scale collaboration is the MELLODDY (Machine Learning Ledger Orchestration for Drug Discovery) project,[6] an Innovative Medicine Initiative (IMI) project by the European Union with a total funding of 18.4m EUR (2019–2022) including collaborations between pharmaceutical companies, research groups from universities but also small and medium-sized enterprises (SMEs).

Reconstructing the molecular structure that matches given chemical property values is a traditional (optimization) problem and often referred to as inverse-QSAR. One of the most commonly used molecular fingerprints in QSAR is the circular extended-connectivity fingerprint (ECFP).[7] The ECFP has found many scientific applications starting from virtual screening and similarity search[8,9] to biological target prediction,[10] proteochemometric modeling[11] and ADMET endpoint modeling.[12]

[a]Department of Digital Technologies, Bayer AG, Berlin, Germany. E-mail: tuan.le2@bayer.com; djork-arne.clevert@bayer.com

[b]Department of Mathematics and Computer Science, Freie Universität Berlin, Berlin, Germany

† Electronic supplementary information (ESI) available: Detailed information regarding the model architectures and computation time, degeneracy analysis for the ECFPs as well as the information loss due to hash collision. See DOI: 10.1039/d0sc03115a

The topological ECFP representation is a refinement of the Morgan algorithm[13] and usually hashed and folded into a fixed size 1024, 2048 or 4096 sparse bit or count vector to further utilize for predictive modeling tasks. During the fingerprint creation, the ECFP algorithm considers the atom environment, based on the maximum number of atomic neighbors, *i.e.* bond diameter d, and iteratively hashes the concatenated (unique) features to a new integer feature. Since the hash function is mapping randomly and uniformly to a 2^{32}-size space of integers, the ECFPs are often considered to be non-invertible.[14]

In a study published by Kogej *et al.*[15] in 2013 between the two large pharmaceutical companies AstraZeneca and Bayer AG, the extended-connectivity fingerprints (ECFP4) were exchanged among the two companies to subsequently do a nearest-neighbor search to enrich chemical libraries by the exploration of chemical space for prospective high-throughput-screening experiments. The choice for using the 2D binary molecular fingerprint was mainly referenced to the loss of intellectual property and competition issues in case a direct comparison of the two large collections of 1.41 M (AstraZeneca) and 2.75 M (Bayer AG) compounds was opted. It was thought that the ECFP4 'kept the molecular structures of both parties confidential, and in combination with a joint assessment workshop [they] could mitigate any concerns around intellectual property, reverse engineering or structure disclosure that would restrict individual scientists in project work'.[15]

The Joint European Compound Library (JECL)[16] between 2013 and 2018 is another IMI collaboration accomplishment of seven pharmaceutical companies as well as academic research groups and SMEs to accelerate drug discovery on a pre-competitive stage which resulted in a compound library of approximately 500 K small molecules for further screening. Among the 500 K compounds, 312 K non-commercial unique samples originate from pharmaceutical companies which were converted to ECFP6 and shared among the contributing pharmaceuticals as analyzed by Besnard *et al.*[17] Similar to Kogej *et al.*, the ECFP6 was chosen by means of structure-free comparison without disclosure of proprietary information.

The initial combined library of pharmaceutical companies was further utilized for focused library design by academic institutions and SMEs to add *in silico* generated compounds to increase and reach the final library size of 500 K compounds.[18]

In this paper, we describe a method to reverse-engineer the extended-connectivity fingerprint and deduce the molecular structure of the compound. A simple approach to counteract the reverse-engineering could be obtained by permuting all the indices of the ECFP representation of a dataset with an arbitrary indexing, on which any predictive model or analysis on that dataset can still be trained and achieved. However, when working in a collaboration, such as the MELLODDY project, or the previous study by Kogej *et al.* between AstraZeneca and Bayer AG or the JECL, combining several databases of (permuted) shared fingerprint descriptors to do successive analyses inevitably requires to share the permutation matrix as well. By sharing the reindexing scheme, we return to our initial position of our motivation on the reverse-engineering of compounds based on ECFPs.

Related work analyzes to what extent chemical descriptors can be shared until molecular structures can be reverse-engineered. Those studies focused on the disclosure of physico-chemical properties and topological indices. In Masek *et al.*,[19] the authors use an iterative genetic algorithm (GA) to suggest molecular structures that have the same chemical descriptor value(s) as a target compound. The genetic algorithm proceeds with suggested structures that match the descriptor value(s), *i.e.* minimize a total fitness function, which takes several descriptor values into account. The authors, however, test their method only on 100 selected target compounds and merely consider descriptors describing molecules that adhere to the Lipinski rule of five, or combination of BCUT descriptors and the MACCSkey fingerprint.[20,21] Using their genetic algorithm, they obtained a high number of false positives – molecular structures that match the descriptor values but are in fact not the real molecular structure. A similar approach to Masek *et al.*, but not in the context of deducing molecular structures, is done by Winter *et al.*[22] In their work for optimizing compounds in a drug discovery endeavor, the authors combine *in silico* prediction of molecular properties with an *in silico* optimization algorithm to suggest molecules that satisfy, or even positively improve, the desired characteristics defined by the user.

Faulon *et al.*[23] proposes a stochastic and deterministic reverse-engineering algorithm to deduce the molecular structure from simple topological indices such as shape[24] and connectivity[25] indices, the Wiener[26] and Balaban J and J_t distance indices[27] as well as their developed atomic signature descriptor.[28] In their analyses, the authors define the degeneracy as the number of structures having the same descriptor value in a given chemical database. From a computational point of view, descriptors with a high degeneracy are assumed to be safe to exchange, as those descriptors correspond to an 1-to-N mapping. This intuition becomes clear when the molecular weight (MW) is exchanged. Given the molecular weight, many possible molecular structures can be deduced. Similar to the work of Masek *et al.*, combining more chemical descriptors can improve the success rate of deciphering the true molecular structure. In their studies however, only 1000 compounds out of PubChem[29] were randomly selected for reverse-engineering and their best method achieves a reconstruction accuracy of 12.2% with drawbacks in computation time on (local) CPUs.

Recent work from Kotsias *et al.*[30] and Maragakis *et al.*[31] in conditional *de novo* drug design utilize the ECFP representation of compounds as input (seed) with additional bioactivity labels to narrow and navigate the generative process towards chemical regions of interest. They train a generative model to sample novel compounds that satisfy the bioactivity condition and are to some degree similar to the input ECFP seed. Their study reveals that the trained generative models are able to sample compounds that correspond to the input seed.

The motivation of our work differs from Kotsias *et al.* and Maragakis *et al.*, as we want to train a model that learns the relationship between ECFP and its corresponding molecular structure, in contrast to the aforementioned work, that aims to generate new compounds, and by chance can reconstruct the compound that corresponds to the input ECFP.

One common evaluation approach for *de novo* molecular design methods is the rediscovery task of selected compounds based on their extended-connectivity fingerprints. The rediscovery task is methodologically different from our approach, as the rediscovery task aims to evaluate whether a generative model, trained on a given dataset, can sample selected (target) compounds that have been intentionally excluded from the training set. By successfully achieving the rediscovery of target compounds by the generative model, its goodness among the ability to sample accessible real-world compounds is strengthened. The GuacaMol benchmark by Brown *et al.* implements the rediscovery task as one benchmark among many goal-directed benchmarks, to assess the quality of SMILES-based generative models to retrieve the three target compounds Celecoxib, Troglitazone and Thiothixene.

Our main contributions are two fold. First, we describe the Neuraldecipher (illustrated in Fig. 1), a fast method to decipher the circular extended-connectivity fingerprint (ECFP)[7] to their molecular structure as SMILES representation[33] by formulating the reverse-engineering task as a machine learning (ML) problem. Next, we show how our method is performing on several configurations for the ECFP, based on selected length k of the fingerprint and bond diameter d. These studies attempt to answer the question, to what extent ECFP can be securely shared, until our proposed method can fully reconstruct the molecular structure on unknown fingerprints. We want to emphasize the importance for the protection of intellectual property and raise awareness that exchanging possibly invertible fingerprints can cause damage on a competitive level for private institutions, such as pharmaceutical companies. Since it is nowadays common for private and public institutions to work in joint collaboration to accelerate drug discovery as seen in JECL or MELLODDY, the development of secure and appropriate molecular fingerprints for common downstream tasks in computational chemistry is desired. Our study shows how to reverse-engineer extended-connectivity fingerprints and should motivate research groups to start of a new field in cryptographic chemistry.

2 Methods

One computational way to achieve the reconstruction would be to compare the given ECFP sample against a large accessible chemical library, where the mapping from ECFP to SMILES representation is known. The molecular structure could then be deduced by either performing an identity check of a given ECFP and the corresponding chemical library, and then returning those samples which match the target ECFP. If the ECFP representation cannot be found in the chemical library, the ECFP should be screened against that chemical library by computing pairwise similarities between the target ECFP and each sample of the reference library. A similarity measure could be the Tanimoto similarity of the respective ECFP pair. Deducing the molecular structure is then achieved by returning those pairs with highest Tanimoto similarity τ satisfying a defined treshold, *e.g.* $\tau > 0.90$.

We formulate the reverse-engineering task as a machine learning problem with the goal to predict the molecular structure given an observed ECFP sample. Our reverse-engineering method is a two-step approach and utilizes the continuous and data-driven molecular descriptor (cddd), a neural network model for the generation of lower-dimensional vector representation of molecular structures.[1] This model utilizes a recurrent autoencoder trained on the task of translating SMILES representation of compounds into their canonical form. Translation works as follows: first, the encoder model translates the input SMILES representation into the cddd-representation, a 512-dimensional vector representation for compounds, that have been shown to be effective on QSAR prediction and virtual

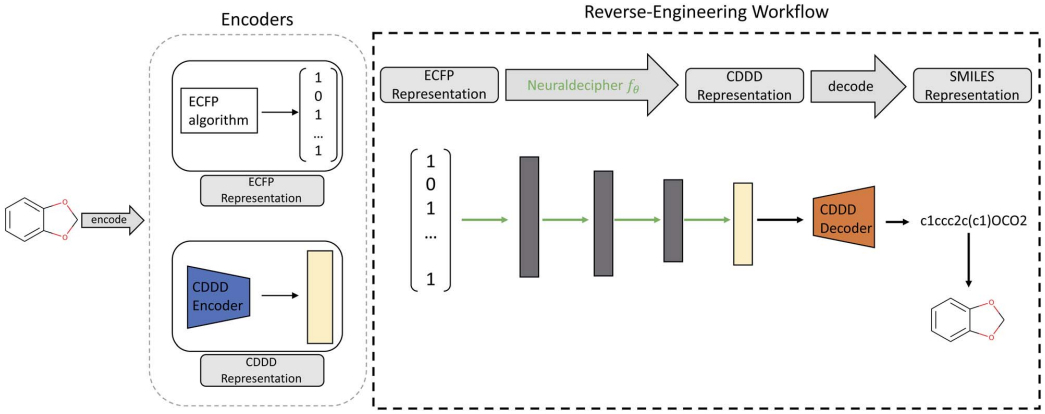


Fig. 1 Illustration of the reverse-engineering workflow. Given an ECFP representation (here exemplary as bit-vector), we predict the corresponding cddd-representation and utilize the fixed decoder network from Winter *et al.* to obtain the SMILES representation. Therefore the Neuraldecipher learns the mapping between the two encoded molecular representations. Trainable parameters for the Neuraldecipher are displayed as green arrows, while black arrows correspond to operations that are fixed and not optimized during training.

screening tasks.[1] Second, the decoder network translates the cddd into the canonical SMILES representation. The SMILES notation is a representation that encodes the topological molecular graph into a linear string of symbols.

Our goal for reverse-engineering is to predict the corresponding cddd vector, given an input ECFP sample. Once we have predicted the cddd vector, we can deduce the molecular structure by utilizing the fixed decoder network, which returns the SMILES representation. Our proposed method has the advantage that we obtain a regression model that is able to predict the molecular structures of ECFP samples more efficiently in a one-shot scenario, as opposed to an autoregressive model that predicts the SMILES representation given an input ECFP. By utilizing the pretrained CDDD model, the Neuraldecipher does not have to learn its own representation of chemical structures and to reconstruct SMILES strings with the correct syntax as illustrated in Fig. 1.

To obtain the SMILES representation, the decoder recurrent neural network (RNN) from Winter *et al.* takes the predicted cddd vector as input and feeds it into a fully-connected layer whose output is split into three parts to initialize three stacked recurrent layers. The output of the decoder network's RNN is a sequence of probability distributions of the different possible characters over the defined SMILES token by Winter *et al.* The deterministic decoder RNN applies a left-to-right beam search[34] with a beam-width of 10 to obtain the final SMILES representation.

2.1 Neuraldecipher model

The Neuraldecipher model is a standard feedforward neural network with fully connected layers. Let $\mathbb{F} \subset \mathbb{Z}^k$ be the ECFP-space with dimension k, where k is the length of the folded extended-connectivity fingerprint. Depending on bit or count extended-connectivity fingerprints, the entries of the ECFP are either populated with $\{0,1\}$ or positive integers $\mathbb{Z}$. The CDDD-space $\mathscr{C}$ is a bounded and compact 512-dimensional space, *i.e.* $\mathscr{C} \subset [-1,1]^{512}$. The Neuraldecipher f_θ is a regression model, mapping from ECFP-space to the corresponding CDDD-space, *i.e.* $f_\theta : \mathbb{F} \to \mathscr{C}$, where θ is the set of trainable model parameters. Fig. 1 illustrates the general reverse-engineering workflow.

The training of the Neuraldecipher is done *via* minimizing the distance $l(d) = l(\text{cddd}_{\text{true}} - \text{cddd}_{\text{predicted}})$, where l is the logarithmic cosine–hyperbolic function, which is a similar loss function as the L_2 squared-error loss. The logarithmic cosine–hyperbolic function is defined as

$$l(d) = \log\left(\frac{\exp(d) + \exp(-d)}{2}\right), \quad \text{where } d = \text{cddd}_{\text{true}} - \text{cddd}_{\text{predicted}}. \tag{1}$$

The number of hidden layers and corresponding hidden neuron units depend on the length of the input ECFP, *i.e.* k and will be discussed in the results Section 3.

We used ADAM optimizer with initial learning rate of 10^{-4} and 5×10^{-4} as weight decay coefficient. We trained the Neuraldecipher model for 300 epochs with a batch-size of 256. The learning rate was updated and multiplied by 0.7 according to a plateau scheduler with a patience of 10 epochs with respect to the validation metric. Additionally, we applied early stopping with a patience of 50 epochs with respect to the validation metric. Throughout all training experiments, the validation metric was the loss on a validation set.

2.2 Datasets

The data used in this study were extracted from the ChEMBL25 database[35] and consists of 1, 870, 461 molecular structures. We used RDKit[36] to retrieve the canonical SMILES representation and removed stereochemistry. We also removed duplicates and filtered with RDKit using the same criteria as done by Winter *et al.*: only organic molecules, molecular weight between 12 and 600 Da, more than 3 heavy atoms and a partition coefficient log *P* between – 7 and 5. Furthermore, we stripped the salts and only kept the largest fragments. After this procedure, our processed dataset contains 1, 526, 990 unique canonical SMILES representation. Yet, across many applications, machine learning models often fail to generalize when tested on data distributions different from training data.[37] In order to check whether our model is not overfitting and motivate a real-world scenario, we clustered the processed SMILES dataset into 10 groups. The clusters were obtained by first computing the MACCSkey fingerprint[21] for each SMILES representation using RDKit, and then utilizing sklearn's KMeans clustering implementation[38] on the MACCSkey fingerprints. To obtain training and validation set, we computed the average pairwise distances between each of the 10 cluster centroids. The validation cluster was then selected by retrieving the cluster (in our case, cluster 7) whose centroid was on average the most distant to the other cluster centroids. Finally, our training set consist of 1, 414, 658 samples and validation set of 112, 332 samples. We call this splitting procedure cluster split. To evaluate how our model performs on a random split, we randomly divided the processed dataset into training and validation set with the same validation set size as in the cluster split scenario. Training of the model is done with the training set and model selection is based on the evaluation on the validation set.

We also test our model on two unseen sets that have no overlap with the training set. The first set is the filtered ChEMBL26 temporal split (with 55, 701 unique compounds) and the second set consists of compounds from one of our internal databases (with 478, 536 unique compounds). The ChEMBL26 temporal split contains compounds that are novel in the ChEMBL26 database,[39] when compared to ChEMBL25. For the internal set, we randomly sampled 500, 000 compounds from one of our processed databases that have no overlap with the ChEMBL25 set. We applied the same preprocessing filter as done before for both datasets. Dataset statistics for the processed, internal and temporal sets are listed in Table 1 and distribution plots displayed in Fig. 2.

2.2.1 ECFP data. To analyze to which extend folded ECFPs can be securely exchanged, we created ECFP bit and count vectors for the lengths $k \in \{1024, 2048, 4096, 8192, 16\,384, 32\,768\}$. The bond diameter d was selected as $d = 6$, leading to

Table 1 Dataset statistics for the processed, internal and temporal datasets. The values listed are the mean (standard deviation) values for each descriptor. The descriptor values were computed with RDKit. The last column displays the number of unique samples in each dataset

Dataset	Mol. weight	Num. atoms	Num. bonds	Num. rings	Number of samples
Processed (train/valid.)	380.70 (90.76)	48.18 (12.98)	50.53 (12.64)	3.37 (1.24)	1, 414, 658/112, 332
Internal	418.85 (82.89)	51.73 (12.04)	54.48 (12.62)	3.76 (1.06)	478, 536
Temporal	401.75 (91.53)	50.38 (12.54)	53.03 (14.21)	3.66 (1.29)	55, 701

$ECFP_{6,k}$ bit- and count fingerprints. Since the collision of bits/counts with increasing fingerprint size decreases, more information about the molecular structure is preserved in the ECFP. Following this thought, our hypothesis is that deciphering molecular structures on larger ECFP size becomes more accurate, as the folded ECFP adheres a smaller information loss. To gain insight on how the model is performing on fingerprints created with different bond diameters and folded onto a fixed length, we calculated ECFPs of length 4096 and bond diameters {4, 8, 10}.

2.2.2 CDDD data. To train and validate our method, we obtained the cddd vector representation by utilizing the encoder network of Winter *et al.* for each unique SMILES representation in our processed datasets, *i.e.* training and validation set (Table 1).

3 Results

For each ECFP setting introduced earlier, we conducted a hyperparameter search by defining possible parameters and searched for the optimal parameters using grid- and random search with a maximal number of 200 trials. We refer to the ESI† for description of the hyperparameter optimization and report the general model architecture and training procedure in the following. Each hidden layer consists of three consecutive operations: affine linear transformation, batch-normalization, and ReLU activation. We tested other activation functions like leaky ReLU, ELU and SoftPlus in the initial experiments, but found ReLU to be superior to the aforementioned non-linearities.

We applied at least 3 hidden layers and decreased the hidden neuron units to 512, followed by the output layer with 512 neurons and applied tanh non-linearity as output activation, since the cddd-vectors are bounded within [−1, 1]. All models were implemented in PyTorch.[40]

3.1 Degeneracy analysis

One natural question that arises with any molecular descriptor or fingerprint is the degeneracy. Recall that molecular weight as descriptor has a high degeneracy, since many compounds can correspond to a certain molecular weight. As the ECFP algorithm iteratively maps atomic environments to features, we believe that the computed ECFP sets from our processed dataset (1.4 M compounds) contains many unique samples with increasing bond-diameter d. Generally speaking, the larger the bond-diameter d is selected, the more local features of a compound are used to create the final fingerprint. To analyze the uniqueness of ECFPs, we computed the degeneracy for each ECFP dataset obtained with increasing bond-diameter d and show the analysis for the bit ECFPs with length 4096 in Fig. 3.

The horizontal axis states the degeneracy, *i.e.* it flags the presence of duplicate, triplicate, and so on. Since we want to count the number how often duplicates, triplicates, and so on, occure, we excluded the degeneracy of 1, *i.e.* the number of unique ECFP samples that occure only one time in the processed dataset.

A degeneracy of 3 means that 3 different structures have the same ECFP. Since this can happen multiple times, the vertical axis counts the occurence of each degeneracy within the processed dataset. As the bond diameter d increases for the $ECFP_{d,4096}$ bit-vectors, the number of unique samples increases, *i.e.* the degeneracy counts for duplicates, triplicates, *etc.* decrease. A higher bond diameter in the ECFP algorithm leads to more uniqueness as more structural information is captured when iterating over larger atom environments (see Table 2).

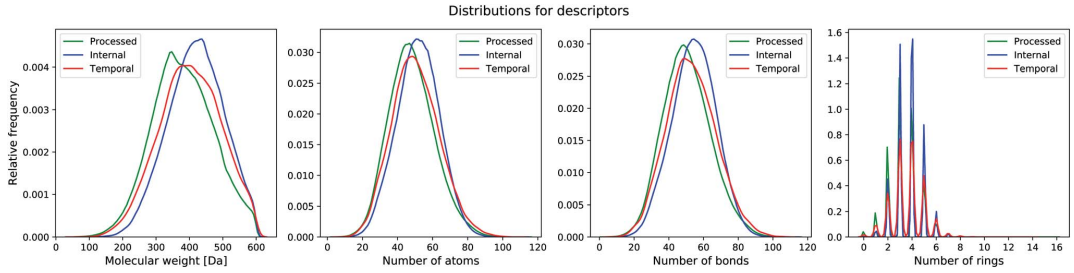


Fig. 2 Distribution of molecular properties (molecular weight, number of atoms, number of bonds, number of aromatic rings) in the different datasets.

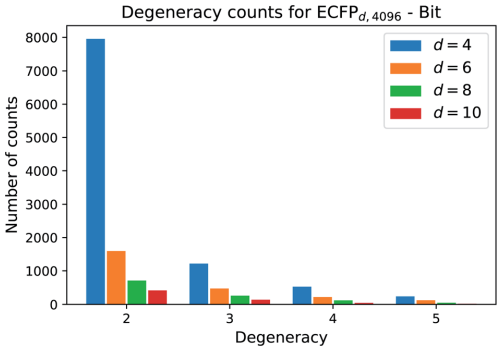


Fig. 3 Frequency count for each degeneracy. As the bond diameter d increases, the count for each degeneracy decreases, *i.e.* there are more unique ECFP samples. The barplot displays the counts for the degeneracies [2, 3, 4, 5]. Degeneracies larger than 6 are not displayed, since the frequency that 6 different structures map to the same ECFP is small.

Table 2 Illustration of non-unique samples for the ECFPs created with length 4096 and increasing bond diameter d. The first column describes the ECFP setting with bond diameter d and length k. The second column states the number of non-unique samples for ECFP-bit vectors, whereas the third column reveals the number of non-unique samples for ECFP-count vectors. To illustrate that the number of non-unique ECFPs is mainly influenced by the bond diameter d (for variable length k), the results for ECFP$_{6,1024}$ with length 1024 and bond diameter 6 is also listed

ECFP	# non-unique bit	# non-unique count
ECFP$_{4,4096}$	14, 382	2, 671
ECFP$_{6,1024}$	4, 569	232
ECFP$_{6,4096}$	4, 481	232
ECFP$_{8,4096}$	2, 509	14
ECFP$_{10,4096}$	1, 005	6

The number of non-unique samples for a fixed diameter $d = 6$ and increasing vector length k does not differ much, as the ECFPs (folded into fixed-length vectors of size k) represent the same structure in a larger fingerprint vector.

We refer to the ESI† for a detailed list of non-unique samples in each setting.

Since the encoded cddd-representation benefits from an injective mapping given SMILES in contrast to the ECFPs, an interesting bound to analyze is the distance between encoded cddd-representations, where the mapping from ECFP to SMILES is non-unique (we call that set of SMILES tuples S_d). Generally, the impact of the non-uniqueness from ECFPs can compromise the training of the Neuraldecipher in two scenarios. In the first preferable scenario, the (average) distance between cddd's encoded from S_d tuples is low. That means low distortion in the corresponding CDDD-space when learning the mapping from ECFP-space to CDDD-space. The second scenario includes a larger average distortion and could degrade the training of the Neuraldecipher, since learning the mapping from ECFP-space to CDDD-space is perturbed as the model encounters ECFP samples that map to diverse cddd-representations. To analyze the two possible scenarios, we retrieved the set of SMILES S_d that includes tuples (*i.e.* duplicates, triplicates, *etc.* see Fig. 3), of SMILES representation that map to the same ECFP. We retrieved the corresponding cddd's for each tuple set of S_d, and calculated the average cosine distance of each pair in the tuple sets.

Fig. 4 illustrates the results for the ECFP$_{6,1024}$ bit-vector setting. The ambiguity of binary ECFPs with different SMILES representation does not cause a large distortion in the corresponding CDDD-space, as the unsupervised learned representation maps structurally very similar SMILES into close space as indicated in the average cosine distance of 0.0417. The right plot in Fig. 4 illustrates two randomly selected pair of molecules for the first scenario (low distortion, *i.e.* cosine distance $\leq$0.05) and second scenario (high distortion, *i.e.* cosine distance $\geq$ 0.20). However in the second scenario, the binary ECFP can misleadingly map to a representation, where the molecules are more different (the molecule pair in the second row of Fig. 4 has a cosine distance of 0.3335). Since the binary-ECFP only captures presence of certain atomic environments (and not counts, as opposed to count-ECFP) the molecules in the second row of the right plot in Fig. 4 correspond to the same ECFPs but refer to different cddd-representations with larger distortion.

3.2 Results and discussion

We trained separate Neuraldecipher models on a cluster and random split, for each ECFP setting. The ECFP setting was determined by bond diameter d, fingerprint length k and exposure of bit- or count ECFPs.

After training, for the final evaluation on the validation, internal and temporal dataset, we predicted the corresponding cddd-vectors and retrieved the SMILES representation by utilizing the decoder network from Winter *et al.*

3.3 Varying the length k for fixed diameter $d = 6$

The results for the ECFP$_{6,k}$ bit-vectors with increasing length k, trained on cluster and random split are listed in Table 3.

The reconstruction columns in Table 3 correspond to the accuracy of binary string matching between true input SMILES representations and deduced SMILES representations. Hence, the reconstruction refers to the accuracy of correctly deducing the exact molecular structure given the ECFP$_6$-bit vectors. The Tanimoto columns state the average Tanimoto similarity between true input SMILES and deduced SMILES representations. To compute the Tanimoto similarity, we retrieved the ECFP$_{6,1024}$ bit fingerprints of true and deduced SMILES and utilized RDKit's Tanimoto similarity implementation. We included the Tanimoto similarity as proxy for the goodness of reverse-engineering, since our model might fail to fully deduce the exact molecular structure, but is still able to reconstruct (structurally) similar compounds that resemble the true compound, which could be optimized in a subsequent task.

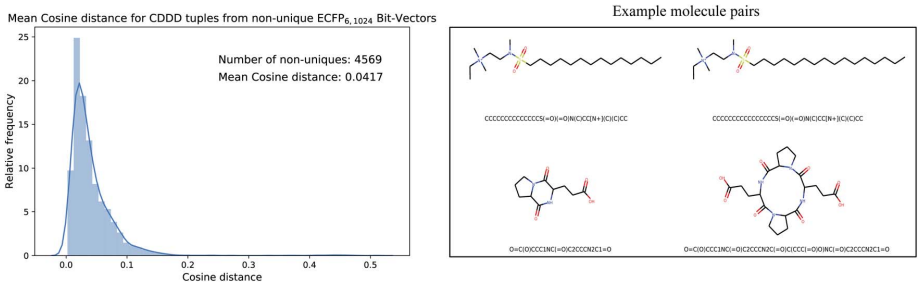


Fig. 4 Mean cosine distance between cddd-representations that belong to the same set of SMILES that map to the same ECFP representation. On average, the cosine distance is small with 0.0417.

Considering that we are using the decoder network to retrieve the reconstructed SMILES representations of predicted cddd-representations, the validity of reconstructed SMILES, *i.e.* if the string representation follows the SMILES grammar, is of great importance, especially in generative modeling.[32]

In all experiments, the SMILES validity on the test datasets (validation, internal, temporal), was most of the time around 98%. We refer to the ESI† for a detailed view of the validity for each configuration. All metrics were computed using the validation (Val.), temporal (Temp.) and internal (Inter.) datasets, which the models have not seen during training.

As expected, models trained on the random split perform better than models trained on the cluster split, when deducing molecular structures from the validation dataset. For example, the model for the $ECFP_{6,1024}$ is able to correctly deduce 12.14% from the validation dataset when trained on the cluster split. The reconstruction for the cluster split is smaller because the validation dataset contains compounds which likely lie in a chemical space, the model has not seen before during training. When the model is trained on a random split, 28.70% of the validation dataset can be correctly reverse-engineered. For the internal and temporal datasets, the performance for cluster split and random split are almost similar along all models. This insight is normal and expected, as the data distributions from the internal and temporal sets generally differ from the processed ChEMBL25 dataset.

One of our hypotheses was that the probability of reverse-engineering molecular structures from folded ECFPs increases with larger size, as the ECFPs are less prone to information loss due to hash collision. This is confirmed by our experiments (Table 3), as models trained with larger $ECFP_6$ input bit-vectors are more capable to correctly deduce the molecular structure in all evaluation datasets. Increasing the ECFP size from 16, 384 to 32, 768 does not improve the performance very much, as the information loss through the hash collision is small. For an analysis on the hash collision for the analyzed fingerprint lengths, we refer to ESI.†

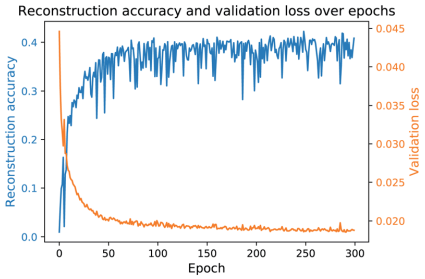
Our reverse-engineering workflow has the benefit of fast computation for the intermediate cddd-representation. The elapsed time for one forward pass of 1 M compounds, when predicting the cddd-representation given varying ECFP-representations, amounts to approximately 5 seconds given ECFPs of length 1024, and up to 100 seconds for ECFPs of size 32, 768. Using the cddd-decoder RNN model to obtain the SMILES representation requires more time due to the nature of sequential models and integration of beam-search. Decoding 1 M cddd-representations back to SMILES representations requires around 38 minutes. The complete reverse-engineering workflow of 1 M compounds takes about 39 minutes and 40 seconds in case the ECFP-representation of length 32, 768 is used as input for the Neuraldecipher. All computations were performed on a single modern Nvidia Tesla V100 GPU.‡

Table 3 Results for reverse-engineering molecular structures based on $ECFP_6$-bit vectors. To compute the average Tanimoto similarity for all lengths, we first calculated the $ECFP_{6,1024}$ bit-vectors for the true and reconstructed SMILES and then parsed the tuple into RDKit's Tanimoto similarity implementation. We selected a fixed ECFP configuration across all lengths *k*, to have a proper and comparable evaluation on the validation (Valid.), internal (Inter.) and temporal (Temp.) datasets. Larger values up to 100 are better

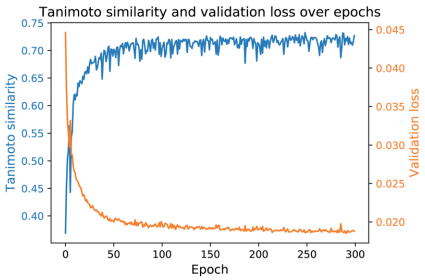
	Cluster split						Random split					
	Reconstruction [%]			Tanimoto [%]			Reconstruction [%]			Tanimoto [%]		
k	Valid.	Inter.	Temp.	Valid.	Inter.	Temp.	Valid.	Inter.	Temp.	Valid.	Inter.	Temp.
1, 024	12.14	11.32	13.34	47.08	45.31	46.84	28.70	12.11	14.14	60.64	40.30	47.60
2, 048	18.85	15.85	18.04	53.65	49.68	51.17	37.87	16.34	18.81	67.11	50.26	51.87
4, 096	32.90	25.08	28.12	63.02	57.06	59.11	57.35	25.30	28.43	79.36	57.39	59.55
8, 192	48.83	37.14	39.98	74.25	66.45	68.24	72.91	36.84	39.81	88.01	66.57	68.33
16, 384	57.85	44.64	47.38	79.80	71.86	73.46	79.79	46.22	48.86	91.30	72.96	74.34
32, 768	59.04	45.81	48.31	80.77	72.84	74.21	80.02	46.92	49.66	91.40	73.35	74.76

In the next study, we trained the models from Table 3 with the same network architectures for each fingerprint length on the $ECFP_6$-count vectors. As the $ECFP_6$-count vector preserves more information about a molecular structure than the corresponding $ECFP_6$-bit vector, the models trained on the $ECFP_6$-count vectors are expected to perform better than models trained on bit vectors only. Table 4 shows the results of this study. Training the Neuraldecipher models on $ECFP_6$-count vectors yields better performance metrics as seen in Table 4 compared to Table 3. For the model trained on 1024 length ECFP, the correct reconstruction of molecular structures in the validation dataset improves to 22.49% for the cluster split model when trained on count vectors as opposed to 12.14% when trained on bit vectors. The conclusions made earlier for better performance with increasing fingerprint size are also reflected in the results in Table 4. With our reverse-engineering method, we are able to correctly deduce around 150 K compounds from the Bayer internal dataset (478 K samples) with 31.73% accuracy, when ECFP-count vectors of length 4096 are shared (see Table 4, random split). Considering that we only used publicly available data from ChEMBL to train the Neuraldecipher model, extra caution has to be paid when exchanging ECFPs with legitimate partners, as the protection of molecular structures is of importance for pharmaceutical companies. The validity of SMILES for all models is as before on average 98%. Since the learning rate scheduler and early stopping mechanism for model selection during training is only affected by the validation loss per epoch, we only computed the evaluation metrics in Tables 3 and 4 based on the final selected model. To observe the progress of evaluation metrics (*i.e.* reconstruction accuracy and Tanimoto similarity), we trained the Neuraldecipher on $ECFP_{6,4096}$-count vectors on the cluster split for 300 epochs without early stopping and computed the corresponding metrics after each training epoch. Fig. 5 shows the progress of the reconstruction accuracy and Tanimoto similarity over epochs compared with the validation loss.

Fig. 5 shows that with decreasing validation loss, the reconstruction accuracy and mean Tanimoto similarity on the validation dataset increase. However, the reconstruction accuracy on the validation data (112, 332 samples) seems volatile and reaches on average 41%. Although the model is not capable to fully deduce the molecular structure, it is able to reconstruct on average compounds that have mean Tanimoto similarity of 72%.



(a) Reconstruction accuracy over epochs.



(b) Tanimoto similarity over epochs.

Fig. 5 Progress of the $ECFP_{6,4096}$-count model during training for the reconstruction accuracy and Tanimoto similarity over epoch. Each plot shows the corresponding metric and the validation loss (cluster split validation) after each training epoch.

A positive relationship between (1 – Tanimoto similarity) and validation loss in Fig. 5b is also shown in the analysis when plotting the Euclidean distance in the corresponding CDDD-space for true cddd and predicted cddd and plot it against (1 – Tanimoto similarity). We refer to ESI for more details.†

3.4 Varying the bond diameter d for fixed length $k = 4096$

The results in Tables 3 and 4 show that the performance on successfully reconstructing molecular structures improves,

Table 4 Results for reverse-engineering molecular structures based on $ECFP_6$-count vectors. To compute the average Tanimoto similarity for all lengths, we first calculated the $ECFP_{6,1024}$ count-vectors for the true and reconstructed SMILES and then parsed the tuple into RDKit's Tanimoto similarity implementation. We selected a fixed ECFP configuration across all lengths k, to have a proper and comparable evaluation on the validation (Valid.), internal (Inter.) and temporal (Temp.) datasets. Larger values up to 100 are better

	Cluster split						Random split					
	Reconstruction [%]			Tanimoto [%]			Reconstruction [%]			Tanimoto [%]		
k	Valid.	Inter.	Temp.	Valid.	Inter.	Temp.	Valid.	Inter.	Temp.	Valid.	Inter.	Temp.
1, 024	22.27	16.92	19.41	61.39	57.59	59.06	38.29	17.85	20.90	71.11	58.13	59.42
2, 048	30.45	22.35	25.94	66.25	61.32	62.90	47.73	22.22	25.77	76.36	61.34	62.99
4, 096	41.02	29.98	34.61	72.58	66.43	68.52	66.61	31.73	36.22	85.98	67.61	69.59
8, 192	55.01	39.63	44.56	80.49	72.77	74.85	77.07	40.89	44.97	90.98	73.60	75.29
16, 384	62.42	46.47	50.61	84.30	76.83	78.44	80.02	46.02	49.48	92.45	76.69	78.05
32, 768	64.03	48.52	52.32	85.07	78.01	79.30	83.52	50.35	54.25	93.85	79.09	80.44

when the fingerprint length k increases and count-vectors instead of bit-vectors are shared. Our next study analyzes how our model performs on a fixed ECFP input length $k = 4096$ and varying bond diameter d.

As the bond diameter d in the ECFP algorithm determines the number of iterations per atom to capture structural information of atom environments, an ECFP generated with bond diameter $d' > d$ is a superset of the ECFP, that was created with bond diameter d. At each diameter, the fingerprint is the combination of features from the previous diameter, plus any new features discovered by that step.[7] In other words, ECFP bit- or count vectors with a higher bond diameter d' can capture more information and the entries of the fingerprint can be more populated with 1's or integers for bit- or count vectors, as opposed to ECFPs created by smaller bond diameter d. We selected the same network architecture from the $ECFP_{6,4096}$ model and trained the model on $ECFP_{d,4096}$ bit- and count vectors, where $d \in \{4, 8, 10\}$. The results for the experiments trained on cluster split and random split are listed in Table 5.

The results in Table 5 go along with the finding that models trained on the random split (rs) perform better on the validation dataset, compared to models trained on the cluster split (cs). There seems to be no substantial difference between the performance on the internal and temporal datasets, when the model was trained on the cluster or random split. Models trained with count-vectors as input perform better than models trained with bit-vectors, as count-vectors preserve more information about the molecular structure.

However, we observe that the performance decreases with increasing bond diameter, regardless of which split the model was trained on. Recall that the unfolded ECFP with a larger bond diameter d' is a superset of the unfolded ECFP with smaller bond diameter d, because more substructures are captured with higher bond diameter ($d' > d$) during the fingerprint algorithm. So generally, the unfolded $ECFP_{d'}$ captures more information than the unfolded $ECFP_d$. Folding the $ECFP_{d'}$ to a fixed length of 4096, *i.e.* to $ECFP_{d',4096}$, comprises a higher information loss due to hash collision. Note that we concluded a similar observation when studying the behavior for increasing fingerprint length k: with increasing fingerprint length k, less information was lost, and therefore the model performance increased (see Tables 3 and 4) for the $ECFP_{d,4096}$. As a result of this, training the Neuraldecipher (with a fixed network architecture) on $ECFP_{4,4096}$ representation as input, leads to better performance compared to the setting, when the input is $ECFP_{6,4096}$. The performance decrease from diameter 8 to 10 is comparably small to the other differences (*i.e.* 4 to 6 and 6 to 8), as the unfolded $ECFP_8$ representations are in most cases the same as the unfolded $ECFP_{10}$ representations and folding these representations into fixed length of 4096 causes the same collision. For a detailed analysis on the hash collision, we refer to ESI.†

3.5 Comparison neuraldecipher against baseline

To further analyze the magnitude of Tanimoto similarity in the cluster validation dataset (112 K samples), we compare our method against a purely computation approach from virtual screening (referred as "Library-Analysis Baseline" and explained in the beginning of Section 2).

For each validation sample, we calculated all pairwise Tanimoto similarities§ to each sample from the reference (library) training set (1.4 M samples). We then computed the average Tanimoto similarity for each validation samples by computing the mean of the aforementioned pairwise similarities ("All-Average"). For the baseline, we selected the top-5 references (training) samples with highest Tanimoto similarity (from the pairwise similarities) and computed the mean of the top-5 references for each validation sample ("Top-5-Average"). The "Top-5-Average" approach demonstrates a weak¶ baseline from compound-library analysis. The "All-Average" procedure aims to show, how similar a validation sample is on average to all samples from reference set, while the "Top-5-Average" procedure aims to show, how similar a validation sample is on average to the top-5 most similar samples from a reference set. Fig. 6 displays the Tanimoto similarity distributions between

Table 5 Results for reverse-engineering molecular structures for ECFPs with fixed length of 4096 and increasing bond diameter d on the cluster- (cs) and random (rs) split. The results for $ECFP_{6,4096}$ from Tables 3 and 4 are listed for completeness. To compute the Tanimoto similarity, we always computed the $ECFP_{6,1024}$ count/bit-vectors for true SMILES and reconstructed SMILES representation in order to have a proper and comparable evaluation for all bond diameters. The first column states the ECFP with the bond diameter d and the flag for cluster (cs) or random (rs) split. Higher values up to 100 are better

	ECFP-count						ECFP-bit					
	Reconstruction [%]			Tanimoto [%]			Reconstruction [%]			Tanimoto [%]		
ECFP	Valid.	Inter.	Temp.	Valid.	Inter.	Temp.	Valid.	Inter.	Temp.	Valid.	Inter.	Temp.
$ECFP_{4,cs}$	43.60	33.19	37.44	74.27	68.93	70.77	34.62	27.21	29.70	65.72	59.72	61.53
$ECFP_{4,rs}$	68.92	34.23	38.78	87.40	69.76	71.60	60.98	28.22	30.98	82.01	60.32	62.20
$ECFP_{6,cs}$	41.02	29.98	34.61	72.58	66.43	68.52	32.90	25.08	28.12	63.02	57.06	59.11
$ECFP_{6,rs}$	66.61	31.73	36.22	85.98	67.61	69.59	57.35	25.30	28.43	79.36	57.39	59.55
$ECFP_{8,cs}$	36.56	26.72	30.56	70.10	64.20	66.34	27.27	21.91	25.14	59.90	54.53	56.75
$ECFP_{8,rs}$	60.21	27.09	31.17	83.09	64.59	66.50	53.03	22.22	25.52	76.70	54.83	57.15
$ECFP_{10,cs}$	34.37	25.52	29.51	68.88	63.37	65.27	23.95	19.82	22.92	57.73	52.89	55.15
$ECFP_{10,rs}$	59.52	26.56	30.98	82.58	64.18	66.33	51.52	21.42	24.55	75.41	53.96	55.94

the "All-Average", "Top-5-Average" and our Neuraldecipher model (trained on $ECFP_{6,4096}$ count-vectors).

As expected and intended through the cluster split, the Tanimoto similarity between the validation and training (reference) set is small on average with 0.1255. The "Top-5-Average" baseline (shaded in red in Fig. 6) obtains a mean Tanimoto similarity of 0.5053 with fat tails approaching the Tanimoto similarity of 0.8. However, the baseline (and even Top-1-Average||) cannot reconstruct the validation samples, *i.e.* reconstruction accuracy of 0. This means that the training (reference) set does not contain the "true" validation samples. This insight goes along with Table 2, displaying 232 non-unique samples for the $ECFP_{6,4096}$ count-dataset. In that case, all non-unique samples are represented in the training (reference) set. Our Neuraldecipher however, achieves a reconstruction of 0.4102 and mean Tanimoto similarity of 0.7218. The fat tail of the Neuraldecipher Tanimoto similarity distribution along the horizontal axis between 0.4 and 0.7 (green curve in Fig. 6) is likely caused by the contribution of Top-5-Average samples. This means that our Neuraldecipher reconstructs structurally similar molecular compounds of that Tanimoto similarity range, because on average the best structures the model can learn from, also share this Tanimoto similarity of 0.5053. Therefore, there is less probability mass in the Tanimoto range of [0.8–0.9]. To compare the performance between the baseline and our method, we plotted the Top-5-Average Tanimoto similarities against the Tanimoto similarities of our reconstructions in Fig. 7.

Fig. 7a and b show that our proposed method performs on average better than the baseline method. Out of 112 K validation samples, our method can reconstruct 85 K samples that have a higher Tanimoto similarity than the baseline model, *i.e.* in 75.89% of all cases. This is illustrated in the contour plot in Fig. 7a and more clearly in the distribution plot in Fig. 7b for $\tau_2 - \tau_1 > 0.0$. To analyze the role of approximate reconstruction we retrieved the subset of samples where our reverse-engineering workflow returned compounds with Tanimoto similarity less than 1.0. We applied the non-parametric paired Wilcoxon rank-sum test with the null hypothesis that the sample distributions of Tanimoto similarities for our reverse-engineering workflow is

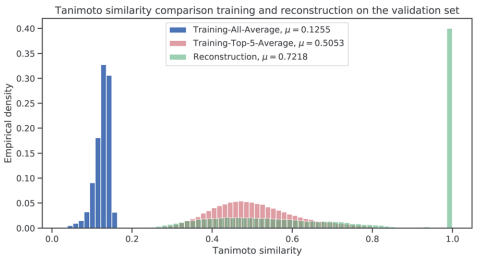
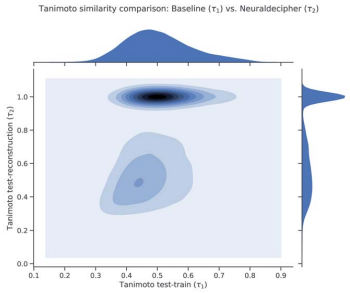
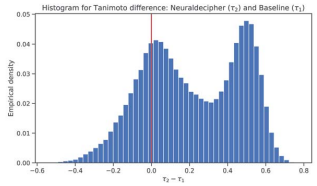


Fig. 6 Histogram to illustrate the distributions for the Tanimoto similarity between true SMILES representations and retrieved SMILES representations from the average training (blue), baseline model (red) and our reconstruction (green) on the validation set (112 K samples).



(a) Tanimoto similarity from Baseline (τ_1) vs. Neuraldecipher (τ_2).



(b) Distribution for Tanimoto similarity difference between Neuraldecipher (τ_2) and Baseline model (τ_1).

Fig. 7 Comparison between the Neuraldecipher and baseline model wrt. the Tanimoto similarity on the validation dataset (112 K samples).

equal to the baseline, and the alternative hypothesis that the sample distributions are not equal, *i.e.* H_0: $\tau_2 = \tau_1$ *vs.* H_1: $\tau_2 \neq \tau_1$. The Wilcoxon rank-sum test is highly statistically significant with a p-value of $p = 1.1921 \times 10^{-7} < \alpha = 0.05$, rejecting H_0 at the 5 – % significance level and indicating that the sample distributions are not equal. The mean Tanimoto similarity of (0.5363 ± 0.1512) from our method suggests that it performs on average better than the baseline (0.4925 ± 0.1105) on the selected subset with around 66.7 K samples.

Furthermore, our reverse-engineering workflow benefits from faster computation. Recall that the baseline model requires the computation of $N \times m$ pairwise similarities, where $N = 1,414,658$ and $m = 112,332$, which subsequently have to be sorted in decreasing order. The elapsed time for the baseline model approximately amounts to 3.75 hours using all cores of a 96-core CPU-machine. Our reverse-engineering workflow only requires approximately 5 minutes using one single Nvidia Tesla V100 GPU and achieves a better reconstruction accuracy.**

One could argue to preserve a stronger baseline by increasing the size of reference library where the overlap between target set and reference library is potentially larger. However, computing pairwise similarities between target and reference library is computationally expensive and does not scale well. Additionally, one should consider that in real-life scenarios, the target dataset consists of in-house compounds from a private institution, that are of interest for reverse-engineering. In general, the baseline method is not able to infer the true compound, based on ECFP. However, if there is an overlap between target and reference library, this overlap is often caused by publicly available molecules, which are also

present in open databases, as explored by Kogej *et al.* when screening the overlap between the AstraZeneca and Bayer AG libraries or other related work.[41,42]

4 Conclusion

In this work we proposed a reverse-engineering method to deduce the molecular structure given the extended-connectivity fingerprint (ECFP). To identify to what extend structures can be reconstructed, we tested our method on several fingerprint settings with varying length k and bond diameter d for the ECFP creation. In general, with increasing fingerprint size and count-vectors being revealed, our method is capable of better reconstructing molecular structures from large sets that our method has not seen before. We selected the ECFP to reverse-engineer from, as the ECFP is a commonly used fingerprint in QSAR and ADMET modeling and often considered as non-invertible. In case ECFP-count representations of length 4096 are exchanged (see Table 5), our method is able to correctly reconstruct up to 68.92% from a random subset of ChEMBL25 (112, 332 unique compounds), 38.78% from the ChEMBL26 temporal set (55, 701 unique compounds) and 34.23% from a random subset of one of our internal databases (478, 723 unique compounds). Although, and somehow fortunately, we did not reach a complete reconstruction on the test sets, due to information loss when folding the unfolded ECFP into fixed-length vectors, there might be small improvements by changing the training procedure. Since we have formulated the reverse-engineering task as a machine learning problem, and utilize neural networks as model class, finding the optimal network architecture and formulating different loss function for training entails the chance for better performance. We suggest that extended-connectivity fingerprints should be exchanged with precaution as this yields the potential to harm intellectual property and loss of competitive advantages since our method is capable to reconstruct molecular structures to some extent.

We hope we raised awareness about the danger when exchanging ECFP representations and motivated a new research field in cryptographic chemistry for the development of secure and appropriate fingerprints for cheminformatics.

Availability

Source code of the proposed method is openly available at https://github.com/bayer-science-for-a-better-life/neuraldecipher.

Conflicts of interest

There are no conflicts to declare.

Acknowledgements

This project was supported by several research grants. FN acknowledges funding from the European Commission (ERC CoG 772230 "ScaleCell") and MATH+ (AA1-6). DC acknowledges funding from the Bayer AG Life Science Collaboration ("DeepMinDS"). TL and RW acknowledge Bayer AG's PhD scholarships.

Notes and references

‡ This computation would cost around 3.06 $ per h$\times\frac{40}{60}$, h = 2.04 per $ on a p3.2× large AWS instance on demand.

§ Based on $ECFP_{6,4096}$ count-vectors.

¶ The baseline is weak because we are using the training set as reference library.

‖ This would be the retrieved sample from the reference training set that is the most similar to a target sample.

** The computation on 96-cores would cost around 4.128 $ per h × 3.75, h = 15.48 per $ on a m5a.24× large AWS instance on demand while our reverse-engineering workflow would only cost around 3.06 $ per h$\times\frac{5}{60}$, h = 0.255 per $ on a p3.2× large AWS instance on demand.

1 R. Winter, F. Montanari, F. Noé and D.-A. Clevert, *Chem. Sci.*, 2019, **10**, 1692–1701.
2 N. Brown, *ACM Comput. Surv.*, 2009, **41**, 8.
3 B. Sanchez-Lengeling, C. Outeiral, G. L. Guimaraes and A. A. Guzik, 2017, ChemRxiv preprint ChemRxiv.5309668.v3.
4 A. Cherkasov, E. N. Muratov, D. Fourches, A. Varnek, I. I. Baskin, M. Cronin, J. Dearden, P. Gramatica, Y. C. Martin, R. Todeschini, V. Consonni, V. E. Kuz'min, R. Cramer, R. Benigni, C. Yang, J. Rathman, L. Terfloth, J. Gasteiger, A. Richard and A. Tropsha, *J. Med. Chem.*, 2014, **57**, 4977–5010.
5 F. Montanari, L. Kuhnke, A. Ter Laak and D.-A. Clevert, *Molecules*, 2019, **25**, 44.
6 *MELLODDY Machine learning ledger orchestration for drug discovery*, https://www.imi.europa.eu/projects-results/project-factsheets/melloddy, accessed: September 8, 2020.
7 D. Rogers and M. Hahn, *J. Chem. Inf. Model.*, 2010, **50**, 742–754.
8 A. Cereto-Massagué, M. Montes, C. Valls, M. Mulero, S. Garcia-Vallve and G. Pujadas, *Methods*, 2014, **71**, 58–63.
9 G. Hu, G. Kuang, W. Xiao, W. Li, G. Liu and Y. Tang, *J. Chem. Inf. Model.*, 2012, **52**, 1103–1113.
10 N. Wale and G. Karypis, *J. Chem. Inf. Model.*, 2009, **49**, 2190–2201.
11 G. Van Westen, R. Swier, J. Wegner, A. Ijzerman, H. Vlijmen and A. Bender, *J. Cheminf.*, 2013, **5**, 41.
12 Q. Zang, K. Mansouri, A. J. Williams, R. S. Judson, D. G. Allen, W. M. Casey and N. C. Kleinstreuer, *J. Chem. Inf. Model.*, 2017, **57**, 36–49.
13 H. L. Morgan, *J. Chem. Doc.*, 1965, **5**, 107–113.
14 Z. Xu, S. Wang, F. Zhu and J. Huang, *Proceedings of the 8th ACM International Conference on Bioinformatics*, Computational Biology, and Health Informatics, New York, NY, USA, 2017, pp. 285–294.
15 T. Kogej, N. Blomberg, P. J. Greasley, S. Mundt, M. J. Vainio, J. Schamberger, G. Schmidt and J. Hüser, *Drug Discovery Today*, 2013, **18**, 1014–1024.
16 *JECL Joint European Compound Library*, https://www.europeanleadfactory.eu/elf-2013-2018/joint-european-compound-library, accessed: September 8, 2020.
17 J. Besnard, P. S. Jones, A. L. Hopkins and A. D. Pannifer, *Drug Discovery Today*, 2015, **20**, 181–186.

18 A. Karawajczyk, F. Giordanetto, J. Benningshof, D. Hamza, T. Kalliokoski, K. Pouwer, R. Morgentin, A. Nelson, G. Müller, A. Piechot and D. Tzalis, *Drug Discovery Today*, 2015, **20**, 1310–1316.
19 B. B. Masek, L. Shen, K. M. Smith and R. S. Pearlman, *J. Chem. Inf. Model.*, 2008, **48**, 256–261.
20 F. R. Burden, *J. Chem. Inf. Comput. Sci.*, 1989, **29**, 225–227.
21 J. L. Durant, B. A. Leland, D. R. Henry and J. G. Nourse, *J. Chem. Inf. Comput. Sci.*, 2002, **42**, 1273–1280.
22 R. Winter, F. Montanari, A. Steffen, H. Briem, F. Noé and D.-A. Clevert, *Chem. Sci.*, 2019, **10**, 8016–8024.
23 J.-L. Faulon, W. Brown and S. Martin, *J. Comput.-Aided Mol. Des.*, 2005, **19**, 637–650.
24 L. B. Kier, *Quant. Struct.-Act. Relat.*, 1985, **4**, 109–116.
25 M. Randic, *J. Am. Chem. Soc.*, 1975, **97**, 6609–6615.
26 P. Senn, *Comput. Chem.*, 1988, **12**, 219–227.
27 A. T. Balaban, *J. Chem. Inf. Comput. Sci.*, 1994, **34**, 398–402.
28 J.-L. Faulon, D. P. Visco and R. S. Pophale, *J. Chem. Inf. Comput. Sci.*, 2003, **43**, 707–720.
29 S. Kim, J. Chen, T. Cheng, A. Gindulyte, J. He, S. He, Q. Li, B. A. Shoemaker, P. A. Thiessen, B. Yu, L. Zaslavsky, J. Zhang and E. E. Bolton, *Nucleic Acids Res.*, 2018, **47**, D1102–D1109.
30 P.-C. Kotsias, J. Arús-Pous, H. Chen, O. Engkvist, C. Tyrchan and E. J. Bjerrum, *Nat. Mach. Intell.*, 2019, **2**, 254–265.
31 P. Maragakis, H. Nisonoff, B. Cole and D. E. Shaw, *A deep-learning view of chemical space designed to facilitate drug discovery*, 2020, aRxiv preprint aRxiv2002.02948.
32 N. Brown, M. Fiscato, M. H. Segler and A. C. Vaucher, *J. Chem. Inf. Model.*, 2019, **59**, 1096–1108.
33 D. Weininger, *J. Chem. Inf. Comput. Sci.*, 1988, **28**, 31–36.
34 I. Sutskever, O. Vinyals and Q. V. Le, *Advances in Neural Information Processing Systems 27*, Curran Associates, Inc., 2014, pp. 3104–3112.
35 A. Gaulton, A. Hersey, M. Nowotka, A. P. Bento, J. Chambers, D. Mendez, P. Mutowo, F. Atkinson, L. J. Bellis, E. Cibrián-Uhalte, M. Davies, N. Dedman, A. Karlsson, M. P. Magariños, J. P. Overington, G. Papadatos, I. Smit and A. R. Leach, *Nucleic Acids Res.*, 2016, **45**, D945–D954.
36 G. Landrum and Others, *Open-Source Cheminformatics*, 2006.
37 B. Zadrozny, *Proceedings, Twenty-First International Conference on Machine Learning*, ICML, 2004.
38 F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot and E. Duchesnay, *J. Mach. Learn. Res.*, 2011, **12**, 2825–2830.
39 *ChEMBL26 ChEMBL Database downloads, which includes Oracle, MySQL and PostgreSQL versions of the ChEMBL database, as well as SDF, FASTA and release note files*, Current Release: 26, Last Update: March 2020, https://chembl.gitbook.io/chembl-interface-documentation/downloads, accessed: September 8, 2020.
40 A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai and S. Chintala, *Advances in Neural Information Processing Systems 32*, Curran Associates, Inc., 2019, pp. 8024–8035.
41 J. Schamberger, M. Grimm, A. Steinmeyer and A. Hillisch, *Drug Discovery Today*, 2011, **16**, 636–641.
42 M. F. M. Engels, A. C. Gibbs, E. P. Jaeger, D. Verbinnen, V. S. Lobanov and D. K. Agrafiotis, *J. Chem. Inf. Model.*, 2006, **46**, 2651–2660.

Electronic Supplementary Material (ESI) for Chemical Science.

Supplementary Information for Neuraldecipher - Reverse-engineering extended-connectivity fingerprints (ECFPs) to their molecular structures

Tuan Le,[*,†,‡] Robin Winter,[†,‡] Frank Noé,[‡] and Djork-Arné Clevert[*,†]

†*Department of Digital Technologies, Bayer AG, Berlin, Germany.*

‡*Department of Mathematics and Computer Science, Freie Universität Berlin, Berlin, Germany*

E-mail: tuan.le2@bayer.com; djork-arne.clevert@bayer.com

Model architecture for varying fingerprint size

Table 1 displays the neural network architecture for each Neuraldecipher model. We applied at least 3 hidden layers and no dropout regularization throughout all of ours experiments. We tested dropout regularization with varying settings[1], but found that using dropout leads to inferior performance on the validation set, compared to models without dropout regularization. For hyperparamter tuning, we used the asynchronous Hyperband implementation of the open-source python library *tune*.[1] Figure 1 shows our selected logarithmic cosine-hyperbolid loss function (see Equation 1) and the standard L_2-loss.

[1]E.g., constant dropout probability of 0.1 for all hidden layers, applying dropout on further hidden layers as the input ECFP is sparse and dropping hidden units in the beginning might degrade the performance much, or exponentially decaying the dropout probability up 0.

Table 1: Architecture for each Neuraldecipher model. Each hidden layer consists of the composition of three operations, namely affine linear transformation, batch-normalization followed by ReLU activation. Each integer within the hidden layers bracket, indicates the number of hidden neurons in the hidden layer. The output layer consists of 512 neurons and is activated with Tanh. The last column (elapsed time) states the average duration of one forward pass of 1M compounds through the network for 10 forward passes.

ECFP input-size	Hidden layers	Output-size	Elapsed time [s]
1024	$[1024, 768, 512]$	512	5.46
2048	$[1024, 768, 768]$	512	7.39
4096	$[2048, 1024, 768, 512]$	512	10.61
8192	$[4096, 2048, 1024, 512]$	512	22.68
16384	$[8192, 4096, 2048, 1024]$	512	52.58
32768	$[8192, 4096, 2048, 1024]$	512	101.11

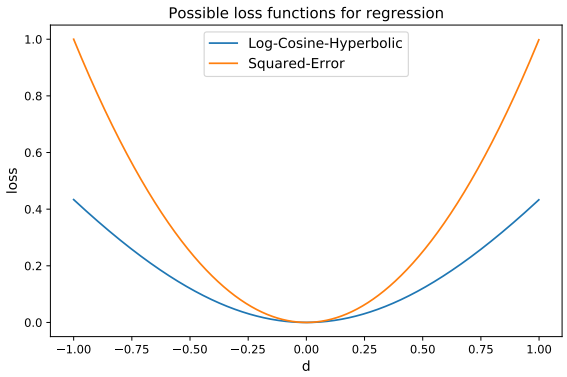


Figure 1: The logarithmic cosine-hyperbolic function and the classical L_2 loss function for regression. The first loss function penalizes stronger for $|d| \geq 0.25$.

$$l(d) = \log\left(\frac{\exp(d) + \exp(-d)}{2}\right), \text{ where } d = cddd_{\text{true}} - cddd_{\text{predicted}}. \tag{1}$$

Degeneracy analysis for $ECFP_6$ settings

The number of non-unique ECFPs for the processed dataset for training depends on the set bond diameter d. For the $ECFP_6$, i.e. generated with bond diameter $d = 6$ with increasing fingerprint length k, we computed the number of non-unique ECFP samples for the bit- and count vectors. The results are shown in Table 2. Given fixed bond diameter d, the number of non-unique samples does face large changes with increasing fingerprint length k. However, folding the ECFPs into smaller fingerprint sizes leads to more information loss, as explained in Section in detail.

Table 2: Number of non-unique samples within each ECFP6 dataset. As the bond diameter d is always the same with $d = 6$, the unfolded ECFPs are in all cases the same, and when folded into the fixed-vector length still remain "unique". The bond diameter is the decisive factor for a high number of degeneracy.

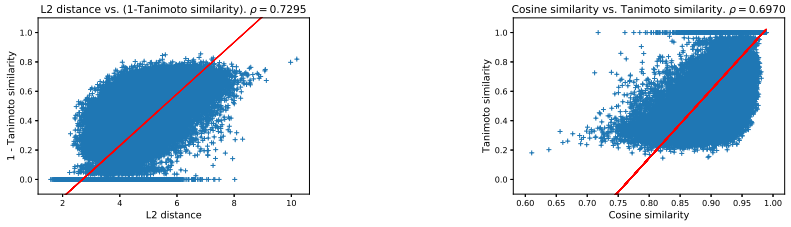
ECFP setting	# Non-unique Bit-ECFP	# Non-unique Count-ECFP
$ECFP_{6,1024}$	4569	232
$ECFP_{6,2048}$	4494	232
$ECFP_{6,4096}$	4481	232
$ECFP_{6,8192}$	4454	232
$ECFP_{6,16384}$	4454	232
$ECFP_{6,32768}$	4445	232

Validity on reconstructed SMILES in all experiments

Table 3: Validity [%] of reconstructed SMILES representation for the validation- (112, 322 unique samples), internal- (478, 536 unique samples) and temporal set (55, 701 unique samples) for the models trained on cluster and random split. In general, the validity of reconstructed SMILES is almost perfect with approximately $98 - 99\%$.

ECFP	ECFP-Count						ECFP-Bit					
	Cluster split			Random split			Cluster split			Random split		
	Valid	Inter	Temp	Valid	Inter	Temp	Valid	Inter	Temp	Valid	Inter	Temp
$ECFP_{6,1024}$	98.81	97.32	97.09	98.13	97.27	97.06	98.27	96.98	96.91	97.66	96.62	96.38
$ECFP_{6,2048}$	98.99	97.44	97.21	98.36	97.24	97.03	98.58	97.02	96.88	98.11	97.05	96.93
$ECFP_{4,4096}$	99.11	97.79	97.70	99.01	97.60	97.39	98.85	97.23	97.10	98.79	97.28	97.09
$ECFP_{6,4096}$	99.05	97.53	97.16	98.89	97.40	97.14	98.75	97.03	96.84	98.55	97.06	96.88
$ECFP_{8,4096}$	98.98	97.28	97.01	98.74	97.28	97.10	98.68	97.08	96.92	98.44	97.08	96.83
$ECFP_{10,4096}$	98.89	97.16	96.95	98.64	97.19	97.02	98.64	96.83	96.74	98.39	97.00	96.79
$ECFP_{6,8192}$	99.31	97.93	97.76	99.31	97.98	97.79	99.19	97.76	97.82	99.12	97.73	97.62
$ECFP_{6,16384}$	99.45	98.33	98.17	99.41	98.12	98.06	99.38	98.15	97.96	99.41	98.17	98.09
$ECFP_{6,32768}$	99.55	98.39	98.23	99.51	98.33	98.26	99.38	98.19	98.12	99.42	98.17	98.12

Analysis CDDD-space vs. ECFP-space



(a) (b)

Figure 2: Dependency between Euclidean (L2) distance and (1 – Tanimoto similarity) as well as Cosine similarity and Tanimoto similarity.

The $ECFP_{6,4096}$-count cluster-split model reports a reconstruction accuracy of 41.02% and mean Tanimoto similarity of 72.58% on the validation dataset (112, 332 samples). To illustrate the dependency between *CDDD*- and ECFP-space for the predicted deduced molecular structures, we computed the Euclidean distance and the Cosine similarity between predicted and true from the validation set. The dependency between Cosine similarity and Euclidean distance against Tanimoto similarity is shown in Figure 2. Since

we formulated the reverse-engineering task as machine learning problem of predicting a close sample, if not the correct sample, during training we aim to obtain a model f_θ, that minimizes the empirical loss function on the training set. Since the empirical loss function contains the deviance d, see Equation (1), the Euclidean distance is implicitly minimized as well. Figure 2a displays the positive correlation (pearson correlation coefficient of 0.7295) between L_2-distance and (1−Tanimoto similarity). As the Euclidean distance increases, the (1−Tanimoto similarity) increases. Interpreting the Euclidean distance and its magnitude in a high-dimensional space is difficult and not straightforward. The Cosine similarity benefits from its property being bounded within -1 and 1. Figure 2b shows the positive correlation (pearson correlation coefficient of 0.6970) between Cosine similarity and Tanimoto similarity. The red lines in Figure 2a and 2b display the linear functions, when regressing the y-axis on the x-axis, indicating the positive trend as well for both plots.

Analysis of hash collision

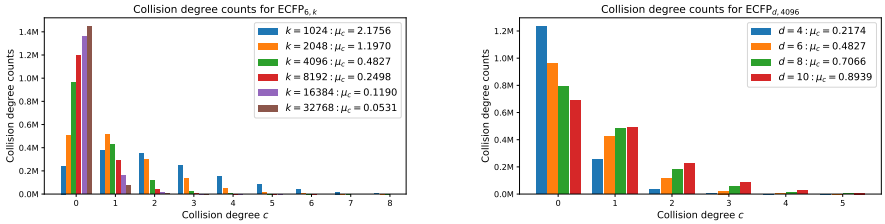
The classical ECFP is an *unfolded* fingerprint with no pre-defined size and its length depends on the input molecular structure. Since the ECFP algorithm iteratively uses a hash function that maps a list of atom environments (represented as integers) to a new atom environment $i \in 2^{32}$ and concatenates the result with the earlier list, the components of the final fingerprint can be large integers due to the target space of the hash function. To obtain *folded* binary or count-vectors from the unfolded fingerprint, the integer entries act as identifier for presence/counts and non-presence in the corresponding binary/count fingerprint. For example, consider a structure where the ECFP algorithm returns an unfolded fingerprint $[10, 10, 80, 999, 999999]$. This leads to an unfolded binary ECFP of length 99999, where the entries $\{10, 80, 999, 999999\}$ are populated with 1 and 0 elsewhere. The unfolded count fingerprint would have the component-value of 2 for the 10-th position, 1 in position $\{80, 999, 999999\}$ and 0 elsewhere. Now the "unfolded" binary/count-fingerprints are still variable in length, namely determined by

the maximum value of the unfolded ECFP, i.e. in the earlier example 999999. Since machine learning algorithms mostly require a fixed length feature input, the unfolded binary/count fingerprints are folded into fixed length k. The folding operation is usually applied with the modulo operation, by modulo-diving the "on"-positions/keys with k. Applying that, the folded bit/count fingerprint has length k. Assume we set $k = 10$ such that our bit/count fingerprints have fixed length of 10. Since the unfolded fingerprint is $[10, 10, 80, 999, 999999]$, indicating i-th's entries being "on", we now obtain the entries $[0, 0, 0, 9, 9, 9]$ being "on". For the binary ECFP this would mean that entries $\{0, 9\}$ are populated with 1 and 0 elsewhere. The count ECFP would be populated with the entry 3 in the components $\{0, 9\}$. Folding the fingerprint has led to a fixed fingerprint where only 2 unique keys $\{0, 9\}$ are on, whereas the original unfolded fingerprint had 4 unique keys $\{10, 80, 999, 9999\}$. Therefore, some information is lost. Here, we define the collision degree c as the difference of the number original unique keys and number of new unique keys, i.e. $c = 4 - 2 = 2$. Note that a collision degree of $c = 0$ means, that **no** information is lost after folding ECFP. Increasing k reduces the chance of collision and therefore the information loss.

For our ECFP_6 configurations, we computed the unfolded ECFP_6 vectors for all compounds in our processed dataset, obtained the number of unique keys and subtracted these values with the number of unique keys for the folded ECFP6 vectors. Figure 3a shows the results for increasing size k. As the fingerprint size k increases, the collision degree of larger than 1 decreases (or in other words, the collision degree of $c = 0$ increases).

Since our studies also include the analyses on the Neuraldecipher performance on a fixed fingerprint length $k = 4096$ but varying bond diameter $d \in \{4, 6, 8, 10\}$, we also computed the collision degrees for each of the five ECFP datasets with varying bond diameters. The results are shown in Figure 3b.

Figure 3a illustrates that the collision degree of $c = 0$, i.e. no information loss due to the folding operation, is highest for the ECFP_6 that was folded into length 32768, followed



(a) Hash collision analysis for fixed $d = 6$ and increasing k

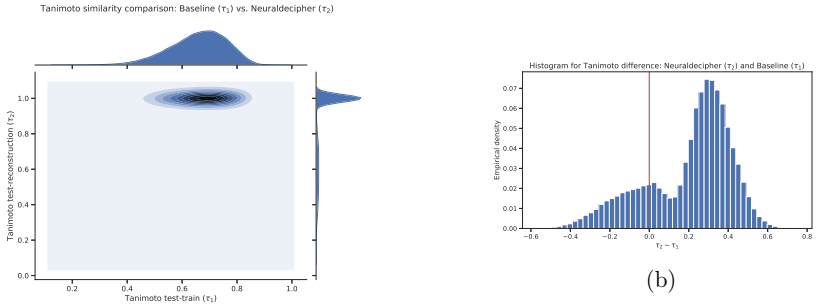
(b) Hash collision analysis for fixed $k = 4096$ and increasing d

Figure 3: Hash collision analysis for varying fingerprint length k and bond diameter d.

by 16384. The larger the fingerprint size, the smaller the average collision degree μ_c is for each setting. A higher average collision degree μ_c corresponds to more information loss.

When fixing the fingerprint length to $k = 4096$ and increasing the bond diameter d, we observe that the information loss also increases (see increasing average mean collision μ_c for increasing bond diamter d in Figure 3b). Since the unfolded $\text{ECFP}_{d'}$ with higher bond diameter $d' > d$ is a superset of the unfolded ECFP_d, the number of unique keys for the $\text{ECFP}_{d'}$ has *at least* the value of the number of unique keys for the ECFP_d. Since the two ECFPs are folded onto the same fixed length of $k = 4096$, it is natural that the ECFP with higher bond diameter suffers from more information loss. This information loss is shown in the higher number of counts for collisions degrees larger than 1, i.e. counts for $c \geq 1$.

Comparison Neuraldecipher trained on ECFP$_{6,4096}$-count vectors random split



(a) (b)

Figure 4: Comparison between the Neuraldecipher and Baseline model wrt. the Tanimoto similarity when trained on random split.

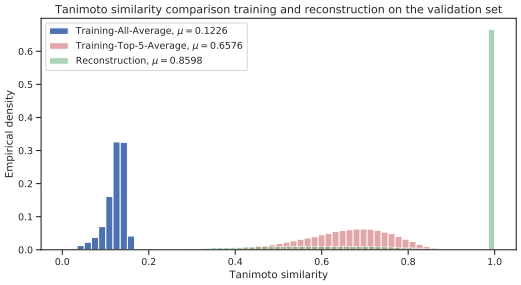


Figure 5: Histogram plot for the Tanimoto similarity between true SMILES representations and retrieved SMILES representation from the average training (blue), baseline model (red) and our reconstruction (green) on the validation set (112K samples) from the random split.

References

(1) Liaw, R.; Liang, E.; Nishihara, R.; Moritz, P.; Gonzalez, J. E.; Stoica, I. Tune: A Research Platform for Distributed Model Selection and Training. *arXiv preprint arXiv:1807.05118* **2018**,

2.2 Publication 2: Parameterized Hypercomplex Graph Neural Networks for Graph Classification

Full Reference: ***Le, Tuan****; Bertolini, Marco; Noé, Frank; & Clevert, Djork-Arné (2021). Parameterized Hypercomplex Graph Neural Networks for Graph Classification. 30th International Conference on Artificial Neural Networks.*

DOI to Journal: 10.1007/978-3-030-86365-4_17

Journal/Conference: ICANN: International Conference on Artificial Neural Networks

DOI to arXiv: 10.48550/arXiv.2103.16584

Source Code: `https://github.com/bayer-science-for-a-better-life/phc-gnn`

Paper's main contributions:

- We propose a novel class of graph representation learning models, named PHC-GNNs, that combine the expressiveness of GNNs and hypercomplex algebras to learn improved node and graph representations.
- We study the learning behaviour of hypercomplex products as a function of the algebra dimension n and introduce novel initialization and regularization techniques for the PHM-layer.
- We demonstrate the effectiveness of our proposed PHC-GNNs reaching state-of-the-art (SOTA) performance compared to other GNNs with a much lower memory footprint.

Author's contribution to the paper:

- Conceptualization of the original idea and methodology.
- Derivation and implementation of the method.
- Design and evaluation of experiments, data curation and analysis.
- Preparation and creation of initial draft and visualizations.
- Revision of the manuscript for the final conference submission.

The manuscript was written by TL. The work was supervised by MB, FN and DAC.

Parameterized Hypercomplex Graph Neural Networks for Graph Classification

Tuan Le[1,2], Marco Bertolini[1], Frank Noé[2], and Djork-Arné Clevert[1]

[1] Machine Learning Research, Digital Technologies, Bayer AG, 13353 Berlin, Germany
[2] Department of Mathematics and Computer Science, Freie Universität Berlin, 14195 Berlin, Germany
{tuan.le2, marco.bertolini, djork-arne.clevert}@bayer.com, frank.noe@fu-berlin.de

Abstract. Despite recent advances in representation learning in hypercomplex (HC) space, this subject is still vastly unexplored in the context of graphs. Motivated by the complex and quaternion algebras, which have been found in several contexts to enable effective representation learning that inherently incorporates a weight-sharing mechanism, we develop graph neural networks that leverage the properties of hypercomplex feature transformation. In particular, in our proposed class of models, the multiplication rule specifying the algebra itself is inferred from the data during training. Given a fixed model architecture, we present empirical evidence that our proposed model incorporates a regularization effect, alleviating the risk of overfitting. We also show that for fixed model capacity, our proposed method outperforms its corresponding real-formulated GNN, providing additional confirmation for the enhanced expressivity of HC embeddings. Finally, we test our proposed hypercomplex GNN on several open graph benchmark datasets and show that our models reach state-of-the-art performance while consuming a much lower memory footprint with 70% fewer parameters. Our implementations are available at
https://github.com/bayer-science-for-a-better-life/phc-gnn.

Keywords: Graph Neural Networks · Graph Representation Learning · Graph Classification

1 Introduction

Geometric deep learning, broadly considered, refers to deep learning methods in the non-Euclidean domain. Although being just in its infancy, the field has already achieved quite remarkable success [3]. The prime example is perhaps constituted by data naturally represented as graphs, which poses significant challenges to Euclidean-based learning [44,5]. This difficulty resides in the topological properties of a graph, which is defined by a pair of sets $G = (V, E)$, where V is the set of vertices and E is the set of edges between vertices, that is, $e_{ij} = (v_i, v_j) \in E$ is the edge between nodes $v_i, v_j \in V$. This structure is inherently discrete and does not possess a natural continuous metric, both fundamental properties for defining the Euclidean topology. These issues have cost the machine learning practitioner time and effort to develop feature engineering techniques to represent the data suitably for Euclidean-based learning methods. For instance, circular fingerprints encode a molecule's graph-like structure through a one-hot-encoding of certain pre-established chemical substructures. The steadily increasing number of applications in which graphs naturally represent the data of interest has driven the development of proper graph-based learning [44]. A prime source of applications stems from chemistry, where predictive models for bioactivity or physicochemical properties of a molecule are rapidly gaining relevance in the drug discovery process [30]. Other applications arise in the context of social and biological networks, knowledge graphs, e-commerce, among others [48,44].

A crucial step for defining graph-based learning is to extend the above definition of a graph by considering each element of the sets V, E as feature vectors, that is, $\mathbf{x}_v, \mathbf{e}_{ij} \in \mathcal{M}$, where $\mathcal{M}$ is a suitable manifold. In this context, the field of graph representation learning (GRL) is often divided into spectral [4,17,25] and non-spectral/spatial approaches [8,14,41]. The latter class, to which this works belong, is based on the idea of *local message passing*, where vector messages between connected nodes are exchanged and updated using neural networks [12]. Most of the literature on GNNs has focused on $\mathcal{M} = \mathbb{R}^n$, that is, vertex and edge embeddings are into Euclidean space. Therefore, it is natural to ask whether this choice is again a restriction imposed by history or simplicity and to which extent GRL could benefit from greater freedom in choosing the manifold $\mathcal{M}$. As first step we consider $\mathcal{M} = \mathbb{R}^n$ as a topological space, but generalize its algebra structure, that is, a vector space equipped with a bilinear product, beyond the real numbers. Example of these are the familiar complex and quaternion algebras, and these and more general

algebras are often referred to as hypercomplex number systems. In mathematics, a hypercomplex algebra is defined as a finite-dimensional unital algebra over the field of real numbers [23], where unital refers to the existence of an identity element e_I such that $e_I \cdot q = q \cdot e_I = q$ for all elements q in the algebra. Such property imposes strong constraints on the algebra structure and dimensionality, which turns out to be 2^n for $n \in \mathbb{Z}_{>0}$. Although hypercomplex number systems crucially inspired our proposed framework, the algebras learned by our models do not satisfy, in general, such constraints. While we are fully aware of this distinction, we will often loosely refer in the following to our models/embeddings as "hypercomplex" to better align with existing literature and avoid introducing additional unnecessary terminology.

2 Related Work

The present work lies at the intersection of three active research areas: (1) geometric approaches to GNNs, (2) hypercomplex/algebraic generalizations of deep learning methods, and (3) regularization/parameter efficiency techniques. This section illustrates how our work relates to these areas and which new aspects we introduce or generalize.

Geometric deep learning is the discipline that comprises the formalization of learning of data embeddings as functions defined on non-Euclidean domains [3]. Hyperbolic manifolds, for example, constitute an important class of non-Euclidean spaces which has been successfully deployed in deep learning. Here, basic manifold-preserving operations such as addition and multiplication in the context of neural networks have been extended to hyperbolic geometry [10]. Such advances led to the development of hyperbolic GNNs. The works [29,6] have empirically shown that the hyperbolic setting is better suited for representation learning on real-world graphs with scale-free or hierarchical structure.

As mentioned above, another defining property of the embedding function learned by a neural network is its underlying vector space structure. Complex- and hypercomplex-based neural networks have received increasing attention in several applications, from computer vision to natural language processing (NLP) tasks [40,11,33,32,43,39]. Hypercomplex representation learning offers promising theoretical properties and practical advantages. It has been argued that, as in the complex case [1], networks possess a richer representational capacity, resulting in more expressive embeddings. Hypercomplex models encompass greater freedom in the choice of the product between the algebra elements: in the case of the quaternion algebra, the Hamilton product naturally incorporates a weight-sharing within the component of the quaternion representation, yielding an additional form of regularization. This approach is, however, virtually unexplored in the graph setting. [31] recently introduced a quaternion based graph neural network, where they showed promising results for node and graph prediction tasks.

The characteristic of the hypercomplex product just mentioned, responsible for heavily reducing the number of parameters (for fixed model depth and width), can be generalized to yield more generic algebras. As a consequence, it is possible to train deeper models, avoiding to overfit the data while supplying more expressive embeddings. The crucial adaption in complex- and hypercomplex-valued neural networks, compared to their real-valued counterpart, lies in the reformulation of the linear transformation, i.e., of the fully-connected (FC) layer. Recent work in the realm of NLP by [47] introduces the PHM-layer, an elegant way to parameterize hypercomplex multiplications (PHM) that also generalizes the product to n-dimensional hypercomplex spaces. The model benefits from a greater architectural flexibility when replacing fully-connected layers with their alternative that includes the interaction of the constituents of a hypercomplex number.

Due to the pervasive application of FC layers in deep learning research, there exists rich literature of methods that aim to modify such transformation in neural networks with the goal to obtain improved parameter efficiency as well as generalization performance. Some examples include low-rank matrix factorization [35], knowledge distillation of large models into smaller models [18], or some other form of parameter sharing [36]. In this work, we embark on the first extensive exploration of hypercomplex graph neural networks. We benchmark our models in graph property prediction tasks in the `OGB` and `Benchmarking-GNNs` datasets [20,9].

The reader might wonder whether our models' performance gain is sufficient to justify the additional formalism of (parametrized) hypercomplex algebras. The answer is that, under many aspects, our models are less involuted than several of the current best-performing graph learning algorithms. For instance, we do not employ any unusually sophisticated aggregation method or message passing function. Moreover, we did not need any extensive amount of hyperparameter engineering/tuning for reaching state-of-the-art (SOTA) performance in the benchmark datasets. This motives our conclusion that the hypercomplex

representations are "easier to learn" and expressive enough to be adaptive to various types of graph data.

In summary, we make the following contributions:

- We propose *Parameterized Hypercomplex Graph Neural Networks* (PHC-GNNs), a class of graph representation learning models that combine the expressiveness of GNNs and hypercomplex algebras to learn improved node and graph representations.
- We study the learning behavior of the hypercomplex product as a function of the algebra dimensions n. We introduce novel initialization and regularization techniques for the PHM-layer, based on our theoretical analyses, and provide empirical evidence for optimal learning at large n.
- We demonstrate the effectiveness of our PHC-GNNs, reaching SOTA performance compared to other GNNs with a much lower memory footprint, making it appealing for further research in GRL to develop even more powerful GNNs with sophisticated aggregation schemes that use the idea of hypercomplex multiplication.

3 Hypercomplex Neural Networks

In this section, we introduce a few elemental concepts and useful terminology in hypercomplex representation learning for our upcoming generalization effort on graphs. We begin by reviewing basic facts about representation learning in complex and quaternion space from literature. We then turn to describe the building blocks and the key features of our class of GNNs in the next Section 4.

3.1 Review of Complex and Quaternion NNs

Complex numbers define an algebraic extension of the real numbers by an *imaginary unit* $\mathbf{i}$, which satisfies the algebraic relation $\mathbf{i}^2 = -1$. Since a complex number $z = a + b\mathbf{i} \in \mathbb{C}$ is specified by two real components $a, b \in \mathbb{R}$, complex numbers are a bi-dimensional algebra over the real numbers. The real components $a, b \in \mathbb{R}$ are called real and imaginary part, respectively. An extension of the same procedure of adding additional (but distinct!) imaginary units, give rise to higher dimensional algebras, known as hypercomplex number algebras. With three imaginary units we recover the most famous hypercomplex algebra, the quaternion algebra: quaternion numbers assume the form $q = a + b\mathbf{i} + c\mathbf{j} + d\mathbf{k} \in \mathbb{H}$ with $a, b, c, d \in \mathbb{R}$ and they define a four-dimensional associative algebra over the real numbers. The quaternion algebra is defined by the relations $\mathbf{i}^2 = \mathbf{j}^2 = \mathbf{k}^2 = \mathbf{ijk} = -1$, which determine the non-commutative *Hamilton product*, named after Sir Rowan Williams [15], who first discovered the quaternions in 1843. Crucial for neural network applications is the representation of the quaternions in terms of 4×4 real matrices, given by

$$Q_r = \begin{bmatrix} a & -b & -c & -d \\ b & a & -d & c \\ c & d & a & -b \\ d & -c & b & a \end{bmatrix} . \tag{1}$$

This representation[3], although not unique, has several advantages. First, the quaternion algebra operations correspond to the addition and multiplication of the corresponding matrices. Second, the first column of Q_r encodes the real and imaginary units' coefficients, simplifying the extraction of the underlying component-based quaternion representation. Finally, (1) is directly generalised as follows to represent linear combinations in higher-dimensional quaternion space $\mathbb{H}^d$. Given a quaternion vector $\mathbf{q} = \mathbf{q}_a + \mathbf{q}_b\mathbf{i} + \mathbf{q}_c\mathbf{j} + \mathbf{q}_d\mathbf{k} \in \mathbb{H}^d$, where $\mathbf{q}_a, \mathbf{q}_b, \mathbf{q}_c, \mathbf{q}_d \in \mathbb{R}^d$, the quaternion linear transformation associated to the quaternion-valued matrix $\mathbf{W} = \mathbf{W}_a + \mathbf{W}_b\mathbf{i} + \mathbf{W}_c\mathbf{j} + \mathbf{W}_d\mathbf{k} \in \mathbb{H}^{k \times d}$ is defined as

$$\mathbf{W} \otimes \mathbf{q} = \begin{bmatrix} 1 \\ \mathbf{i} \\ \mathbf{j} \\ \mathbf{k} \end{bmatrix}^\top \begin{bmatrix} \mathbf{W}_a & -\mathbf{W}_b & -\mathbf{W}_c & -\mathbf{W}_d \\ \mathbf{W}_b & \mathbf{W}_a & -\mathbf{W}_d & \mathbf{W}_c \\ \mathbf{W}_c & \mathbf{W}_d & \mathbf{W}_a & -\mathbf{W}_b \\ \mathbf{W}_d & -\mathbf{W}_c & \mathbf{W}_b & \mathbf{W}_a \end{bmatrix} \begin{bmatrix} \mathbf{q}_a \\ \mathbf{q}_b \\ \mathbf{q}_c \\ \mathbf{q}_d \end{bmatrix} . \tag{2}$$

[3] The same reasoning holds for the complex case, which is recovered by setting the $\mathbf{j}, \mathbf{k}$ components to zero. However, although $\mathbb{H}$ includes $\mathbb{C}$, the quaternions are not an associative algebra over the complex numbers.

This constitute one of the main building blocks for quaternion-valued neural networks [11,33]. The matrix version of the Hamilton product, denoted by the symbol $\otimes$ in Equation (2), encodes the interaction between the real part and three imaginary parts and introduces *weight-sharing* when performing the matrix-vector multiplication. A simple dimension counting shows this: the embedding vector $\mathbf{q}$ of (real) dimension $\dim_{\mathbb{R}} \mathbb{H}^d = \dim_{\mathbb{R}} \mathbb{R}^{4d} = 4d$ is assigned to a weight matrix of dimension $\dim_{\mathbb{R}} \mathbb{H}^{k\times d} = 4kd$, instead of $16kd$ as we would expect from the usual matrix product. Thus, the Hamilton product benefits from a parameter saving of $1/4$ learnable weights compared to the real-valued matrix-vector multiplication. Since the largest body of work in complex- and quaternion-valued neural networks merely utilizes the weight-sharing property just described, it is natural to ask whether the product (2) can be extended beyond the quaternion algebra, thereby allowing us to consider algebras of arbitrary dimensions.

3.2 Parameterized Hypercomplex Layer

The parameterized hypercomplex multiplication (PHM) layer introduced by [47] aims to *learn* the multiplication rules defining the underlying algebra, i.e., the interaction between the real and imaginary components of the algebra's product. One obvious advantage of the PHM layer is lifting the restriction on the (predefined) algebra dimensionality, otherwise being limited to $n = \{2, 4, 8, 16, ...\}$ as in the case of the algebra of complex, quaternion, octonion, and sedenion numbers, respectively.
The PHM layer takes the same form as a standard affine transformation, that is,

$$\mathbf{y} = \mathrm{PHM}(\mathbf{x}) = \mathbf{U}\mathbf{x} + \mathbf{b} \ . \tag{3}$$

The key idea is to construct $\mathbf{U}$ as a block-matrix, as in (2), through the sum of Kronecker products. The Kronecker product generalizes the vector outer product to matrices: for any matrix $\mathbf{X} \in \mathbb{R}^{m\times n}$ and $\mathbf{Y} \in \mathbb{R}^{p\times q}$, the Kronecker product $\mathbf{X} \otimes \mathbf{Y}$ is the block matrix

$$\mathbf{X} \otimes \mathbf{Y} = \begin{bmatrix} x_{11}\mathbf{Y} & \dots & x_{1n}\mathbf{Y} \\ \vdots & \ddots & \vdots \\ x_{m1}\mathbf{Y} & \dots & x_{mn}\mathbf{Y} \end{bmatrix} \in \mathbb{R}^{mp\times nq} \ ,$$

where $x_{ij} = (\mathbf{X})_{i,j}$. Now, let n be the dimension of the hypercomplex algebra, and let us suppose that k and d are both divisible by a user-defined hyperparameter $m \in \mathbb{Z}_{>0}$ such that $m \leq n$. Then, the block-matrix $\mathbf{U}$ in Equation (3) is given by a sum of n Kronecker products

$$\mathbf{U} = \sum_{i=1}^{n} \mathbf{C}_i \otimes \mathbf{W}_i \ , \tag{4}$$

where $\mathbf{C}_i \in \mathbb{R}^{m\times m}$ are denoted contribution matrices and $\mathbf{W}_i \in \mathbb{R}^{\frac{k}{m}\times\frac{d}{m}}$ are the component weight matrices. In the rest of our discussion, we make the simplifying assumption that $m = n$, for which (4) yields $n(\frac{kd}{n^2} + n^2) = \frac{kd}{n} + n^3$ degrees of freedom. Since k and d correspond to the output- and input-size for a linear transformation, and n determines the user-defined PHM-dimension, the overall complexity of the matrix $\mathbf{U}$ is $\mathcal{O}(\frac{kd}{n})$ under the mild assumption that $kd \gg n^4$. This shows that the PHM-layer enjoys a parameter saving factor of up to $\frac{1}{n}$ compared to a standard fully-connected layer [47].

4 Hypercomplex Graph Neural Network

In this section, we introduce our novel hypercomplex graph representation learning model with its fundamental building blocks.

4.1 Initialization of Linear Independent Contributions

While the authors [47] introduced the PHM layer, no further details on the initialization of the contribution matrices $\{\mathbf{C}_i\}_{i=1}^n$ have been elucitated. In the case of known hypercomplex algebras, these matrices can be chosen to be of full rank – that is, the rows/columns are linearly *independent* –, and whose elements belong to the set $\{-1, 0, 1\}$. Following the same logic, let $\tilde{\mathbf{I}}_n$ be a diagonal matrix with alternating signs on the diagonal elements

$$\tilde{\mathbf{I}}_n = \mathrm{diag}(1, -1, 1, -1, \dots) \ . \tag{5}$$

In our work, we initialize each contribution matrix $\mathbf{C}_i$ as a product between the matrix $\tilde{\mathbf{I}}_n$ and a power of the cyclic permutation matrix $\mathbf{P}_n$ that right-shifts the columns of $\tilde{\mathbf{I}}_n$, that is,

$$\mathbf{C}_i = \tilde{\mathbf{I}}_n \mathbf{P}_n^{i-1} \ , \tag{6}$$

where $(\mathbf{P}_n)_{i,j} = 0$ except for $j - i = 1$ and $i = n, j = 1$ where it has value 1. It is immediate to verify that the columns of the constructed $\mathbf{C}_i$'s are linearly independent, as desired. Note that the above construction is not the only one yielding contribution matrices with such properties. In fact, for $n \in \{2, 4\}$ we do not implement (6), but instead we initialize the contribution matrices as in the complex and quaternion algebra.

Learning Dynamics for Larger n With the initialization scheme defined above, each $\mathbf{C}_i$ in (6) contains n non-zero elements versus $n(n-1)$ zero entries. Hence, the sparsity for each contribution matrix scales quadratically as a function of n, while the number of non-zero entries only linearly. While it is still possible for our model to adjust the parameters of the contribution matrices during training, it is conceivable that initializing too sparsely the fundamental operation of algebra, will deteriorate training. To overcome this issue, we also implement a different initialization scheme

$$\mathbf{C}_i \sim U(-1, 1) \ , \tag{7}$$

by sampling the elements from the contributions matrices uniformly from $U(-1, 1)$. We will show in Section 5 that this initialization strategy greatly benefits the training and test performance for models with larger n.

4.2 Tensor Representation

In our work we make heavy use of the reshaping operation to flatten/vectorize the hypercomplex embeddings. This enables us to apply operations such as the PHM-layer, batch-normalization or the computation of "real" logits. Explicitly, let $\mathbf{H} \in \mathbb{R}^{b \times k}$ be a real embedding matrix, where the two axes correspond to the batch and feature dimension, respectively. In terms of hypercomplex embeddings, the second dimension has the size of $k = nm$, that is, where each component of the m-dimensional algebra embedding is concatenated as shown for the Hamilton product in (2). The reshape operation reverts the vectorization of the second axis, i.e., we reshape the embedding matrix as 3D tensor to $\mathbf{H} \in \mathbb{R}^{b \times n \times m}$.

4.3 Batch Normalization

Batch normalization [22] is employed almost ubiquitously in modern deep networks to stabilize training by keeping activations of the network at zero mean and unit variance. Prior work [40,11] introduced complex and quaternion batch normalization, which uses a general whitening approach to obtain equal variance among the $n = \{2, 4\}$ number constituents. The whitening approach, however, is computationally expensive compared to the common batch normalization, as it involves computing the inverse of a $(n \times n)$ covariance matrix through the Cholesky decomposition, which has a complexity of $\mathcal{O}(mn^3)$ for an embedding of size m. In our experiments, we found that applying the standard batch normalization for each algebra-component after 2D→3D reshaping is faster and achieves better performance.

4.4 Regularization

Regularization is one of the crucial elements of deep learning to prevent the model from overfitting the training data and increase its ability to generalize well on unseen, possibly out-of-distribution test data [27]. In what follows, we introduce a few concepts adapted from real-valued neural networks and we extend their applicability to our models.
Let $\mathbf{A} \in \mathbb{R}^{l \times n \times m}$ be a reshaped 3D tensor, where $\mathbf{A}$ could be a hidden embedding or a weight tensor of our model. We employ further regularization techniques on the second axis, which refers to the algebra dimension n. This is motivated by the idea to decouple the interaction between algebra components when performing the multiplication.

Weight Regularization Recall that an element of a n-dimensional algebra is specified by n real components, that is, Let us recall that the l_p-norm of an element $\mathbf{w} = w_1 + w_2 i_1 + \cdots + w_n i_{n-1}$ of a n-dimensional algebra is defined as

$$l_p(\mathbf{w}) = \left(\sum_{i=1}^{n} |w_i|^p \right)^{1/p} . \tag{8}$$

Now, given the set of weight matrices for a PHM-layer, i.e., $\{\mathbf{W}_1, \ldots, \mathbf{W}_n\}$, where each $\mathbf{W}_i \in \mathbb{R}^{k \times d}$, we compute the L_p norm on the stacked matrix $\mathbf{W} \in \mathbb{R}^{k \times n \times d}$ along the second dimension resulting to:

$$L_p(\mathbf{W}) = \frac{1}{kd} \sum_{a=1}^{k} \sum_{b=1}^{d} l_p(\mathbf{W}_{[\mathbf{a},\cdot,\mathbf{b}]}) . \tag{9}$$

This regularization differs from the commonly known regularization of weight tensors, where the l_p norm is applied to each element, such as the Frobenius-norm for $p = 2$:

$$||\mathbf{W}||_F = (\sum_{a,b,c} |\mathbf{W}_{[a,b,c]}|^2)^{\frac{1}{2}} .$$

Sparsity Regularization on Contribution Matrices. In our model implementation, we enable further regularization on the set of contribution matrices $\mathcal{C} = \{\mathbf{C}_i\}_{i=1}^{n}$ by applying the l_1-norm on each flattened matrix:

$$L(\mathcal{C}) = \frac{1}{n^3} \sum_{i=1}^{n} \left(\sum_{a,b} |\mathbf{C}_{i,[a,b]}| \right) . \tag{10}$$

4.5 Computation of Real Logits

Given an embedding matrix $\mathbf{H} \in \mathbb{R}^{b \times k} = \mathbb{R}^{b \times n \times m}$ there are several options to convert a hypercomplex number (along the second axis $i = 1, \ldots, n$) to a real number, such that the result lies in $\mathbb{R}^{b \times m}$. In our work, we utilize a fully-connected layer (FC) that maps from $\mathbb{R}^{b \times nm}$ to $\mathbb{R}^{b \times m}$, i.e.,

$$\begin{aligned} &\text{Real-Transformer}(\mathbf{H}) : \mathbb{R}^{b \times nm} \to \mathbb{R}^{b \times m}, \\ &\text{Real-Transformer}(\mathbf{H}) = \mathbf{H}\mathbf{A}_r + \mathbf{b}_r . \end{aligned} \tag{11}$$

Other possible choices of conversion are the sum or norm operations along the second axis of the 3D-tensor representation of $\mathbf{H}$.

4.6 Hypercomplex Graph Neural Network

Input Featurization. We implement hypercomplex input-feature initialization in GNNs by applying an encoder on the real-valued input features. For continuous features, we apply a standard linear layer to real-valued features $\mathbf{x}_v \in \mathbb{R}^F$ to obtain the hypercomplex zero-th hidden embedding $\mathbf{h}_v^{(0)} \in \mathbb{R}^k = \mathbb{R}^{nm}$ for each node v. According on the algebra dimension n, we then split the k-dimensional vector $\mathbf{h}_v^{(0)}$ into n sub-vectors, each of size m, yielding m dimensional hypercomplex features. Consequently, each hypercomplex (embedding) vector can be reshaped into size (n, m). The same procedure is applied to continuous raw edge-features $\mathbf{e}_{uv} \in \mathbb{R}^B$ for every connected pair of nodes $(u, v) \in E$. In case of molecular property prediction datasets, raw node- and edge-features are often categorical variables, e.g., indicating atomic number, chirality and formal charges of atoms. Categorical edge-features identify instead the bond type between two connected atoms. Categorical input node- and edge-features are transformed using an learnable embedding lookup table [21] that is commonly used in natural language processing. This lookup table maps word entities of a dictionary to continuous vector representations in $\mathbf{e}_{uv}^{(0)} \in \mathbb{R}^{nm}$.

Message Passing We build our PHC message passing layer based on the graph isomorphism network (GIN-0) introduced by [45] with the integration of edge features [21]. The GIN model is a simple, yet powerful architecture that employs injective functions within each message passing layer, obtaining representational power as expressive as the Weisfeiler-Lehman (WL) test [42].
Before any transformations on the embeddings are made, neighboring node representations are aggregated,

$$\mathbf{m}_v^{(l)} = \sum_{u \in \mathcal{N}(u)} \alpha_{uv} (\mathbf{h}_u^{(l-1)} + \mathbf{e}_{uv}^{(l)}) \ , \tag{12}$$

where the edge-embeddings $\mathbf{e}_{uv}^{(l)}$ are obtained through the same encoding procedure as for the $l = 0$ representations described above. The aggregation weights α_{uv} can be computed using different mechanisms [25,14,41,28]. The GIN model, for instance, utilizes the sum-aggregator, i.e., all weights $\alpha_{uv} = 1$. In our class of models we implement several common aggregation strategies, namely, $\alpha_{uv} \sim$\{sum, mean, min, max, softmax\}. Such flexibility is crucial for our models, as different aggregators learn different statistical property of a graph [45]. Often, datasets differ regarding the topological properties of graphs, such as density and size, and as a consequence, optimal embeddings for a given dataset are notably sensitive to the choice of message passing aggregation strategy [7]. The interpretation of Equation (12) is that the message received by node v is a variable aggregation of the sum of the neighboring node embeddings and its corresponding edge-embeddings. This message is then the key ingredient in the update strategy of the node v embedding through a Multi-Layer-Perceptron (MLP)

$$\widetilde{\mathbf{h}}_v^{(l)} = \text{MLP}^{(l)} \left(\mathbf{h}_v^{(l-1)} + \mathbf{m}_v^{(l)} \right) \ . \tag{13}$$

It is in this step that the PHM-layer from Equation (3) is implemented.
Our model differs from complex- and quaternion-based models by the fact that the multiplication rule to construct the final weight-matrix for the linear transformation is learned through the data, see. Eq. (4). Note that the multiplication rule for the quaternion-based model is fixed as shown in Equation (2).

The iterative application of the aggregation function (12) (when $\alpha \sim$ sum) on hidden node embeddings updated through (13) turns out to define an injective function. Our proposed message passing layer is therefore a simple generalization of the GIN module, but uses the parameterized hypercomplex multiplication layer from Equation (3). For the case we set the hyperparameter $n = 1$, our model reduces to a modified version of GIN-0, where the (block) weight-matrix for each affine transformation consists of the sum of Kronecker products from only one matrix, as shown in Equation (4).

Skip Connections We apply skip-connection (SC) after every message passing layer by including either the initial $\mathbf{h}_v^{(0)}$ or the previous layer $\mathbf{h}_v^{(l-1)}$ embedding information of node v,

$$\mathbf{h}_v^{(l)} = \text{SC}(\mathbf{h}_v^{(a)}, \widetilde{\mathbf{h}}_v^{(l)}) = \mathbf{h}_v^{(a)} + \widetilde{\mathbf{h}}_v^{(l)} \ , \quad a = 0, l-1 \tag{14}$$

Graph Pooling The graph-level representation $\mathbf{h}_G$ is obtained by soft-averaging the node embeddings from the final message passing layer, i.e.,

$$\mathbf{h}_G = \sum_{v \in G} \mathbf{w}_v \odot \mathbf{h}_v^{(L)} \ , \tag{15}$$

where $\mathbf{w}_v$ is a soft-attention weight-vector and $\odot$ denotes element-wise multiplication. We follow the proposal of Jumping-Knowledge GNNs [46] and assign attention scores to each hidden node embedding from the last embedding layer. Let $\mathbf{H}^{(L)} \in \mathbb{R}^{|V| \times k_L}$ denote the node embedding matrix, where $k_L = n \cdot m_L$ is the size of the final message passing layer L. We compute the soft-attention weights in (15) by calculating the real logits as defined in (11), followed by a sigmoidal activation function $\sigma(\cdot)$, that is,

$$\mathbf{W}_{\text{sa}} = \sigma(\text{Real-Transformer}(\mathbf{H}^{(L)})) \ . \tag{16}$$

The rows of the soft-attention matrix $\mathbf{W}_{\text{sa}} \in (0,1)^{|V| \times m_L}$ are the (broadcasted) vectors entering in the graph pooling (15).

Table 1. Results on the OGB graph classification datasets. The PHC-GNN can reduce the total number of model parameters and obtains improved averaged test performance over 10 (ogbg-molhiv) and 5 runs (ogbg-molpcba).

Model	ogbg-molhiv		ogbg-molpcba	
	# Params	ROC-AUC (%) ↑	# Params	PR (%) ↑
PHC-1	313K	78.18 ± 0.94	3.15M	29.17 ± 0.16
PHC-2	178K	79.25 ± 1.07	1.69M	$\mathbf{29.47 \pm 0.26}$
PHC-2-C	178K	79.13 ± 0.87	1.69M	29.41 ± 0.15
PHC-3	135K	79.07 ± 1.16	1.19M	29.35 ± 0.28
PHC-4	111K	$\mathbf{79.34 \pm 1.16}$	0.99M	29.30 ± 0.16
PHC-4-Q	111K	79.04 ± 1.89	0.99M	29.21 ± 0.23
PHC-5	101K	78.34 ± 1.64	0.87M	29.13 ± 0.24

Downstream Predictor The graph-level representations (15) are further passed to a task-based downstream predictor, which can (but does not have to) be a Neural Network. For example, a 3-layer MLP is applied in the Benchmarking-GNNs framework [9], while the baseline models from OGB [20] deploy a simple 1-layer MLP. In our work, we implement a 2-layer MLP that processes the graph embeddings through the PHM-layer (3), followed by an additional linear layer to compute the logits as described in (11).

Although we define our GNN as graph classification model, the model can in fact also be utilized for node classification tasks. Such a model can be obtained, by not applying the graph pooling as described in Equation (15), and instead use the last hidden layer nodes embedding $\mathbf{H}^L$ to compute the real logits with (11) before applying the Softmax activation.

5 Experiments

We evaluate the effectiveness of parameterized hypercomplex GNNs on six datasets from two recent graph benchmark frameworks [9,20]. We discuss our fundings by displaying results for three datasets in the main text, and we refer to the Supplementary Information (SI) for further evidence. The two recent graph benchmark frameworks address the inadequacy of past benchmark datasets, which are rather small in size, and thus not suitable for proper model evaluation. These issues become even more relevant for real-life graph-based learning applications, where often the datasets are fairly extensive and the issue of out-of-distribution samples is key in assessing the true predictive performance of the algorithm. To demonstrate the architectural advantage and effectiveness of the hypercomplex multiplication, we evaluate the performance of our GNNs for increasing algebra dimension n. We recall that in our framework, this hyperparameter controls the amount of parameter sharing in the PHM layer (3). In all our experiments, we report the test performance evaluated on the model saved from the epoch with the best validation performance.

Increasing n for a Fixed Network Architecture Table 1 shows results on two molecular property prediction datasets from OGB [20], where all the models share the same *fixed* network architecture. Note that, due to the inherent weight-sharing mechanism, the number of parameters decreases as n increases. We observe an improved performance of our GNN when we adopt the (parameterized) hypercomplex multiplication layers. In fact, all models that were trained with PHM-layer, with the exception of the PHC-5 model, performed better than the "real" baseline PHC-1. This result supports our hypothesis that the employment of hypercomplex multiplication acts as regularizer and aids to better generalization performance on the test set.

For the medium-scale ogbg-molpcba dataset, our models include 7 message passing layers as stated in (13), each of a fixed size of 512. We refer to the SI for further details regarding architecture and hyperparameters. In deep learning, it is often observed that parameter-heavy models tend to outperform parameter-scarce models, but incur the risk of overfitting the training data, as the high number of degrees of freedom tempts the model to simply "memorize" the training data, with the consequential detrimental effect of poor generalization on unseen test data. Consequently, significant effort needs to be invested in regularizing the model, often in an *ad-hoc* manner. This experiment showed that HC-based models offer an elegant and universally-applicable approach to regularization that does not require any extensive hyperparameter tuning. As our baseline PHC-1 model in the ogbg-molpcba benchmark seems

Table 2. Results of the PHC-GNNs on the `ZINC` graph property prediction dataset. Our model can increase its embedding size for a fixed-length network through the inherent weight-sharing component. All shown models are constraint to a capacity budget of approximately 100K (`L=4`) and 400K (`L=16`) parameters and the performances are averaged over 4 runs [9]. Models with †-suffix are initialized with (7).

Model	ZINC, L=4		ZINC, L=16	
	# Params	MAE ↓	# Params	MAE ↓
PHC-1	102K	0.198 ± 0.010	403K	0.178 ± 0.007
PHC-2	99K	0.197 ± 0.007	403K	0.170 ± 0.005
PHC-3	101K	0.191 ± 0.005	407K	0.169 ± 0.001
PHC-4	107K	0.188 ± 0.003	399K	0.167 ± 0.006
PHC-5	106K	$\mathbf{0.185 \pm 0.008}$	408K	$\mathbf{0.164 \pm 0.003}$
PHC-8	104K	0.201 ± 0.005	401K	0.177 ± 0.009
PHC-10	104K	0.218 ± 0.010	395K	0.184 ± 0.005
PHC-16	110K	0.225 ± 0.009	412K	0.199 ± 0.008
PHC-8-†	104K	0.193 ± 0.006	401K	0.166 ± 0.005
PHC-10-†	104K	0.195 ± 0.004	395K	0.165 ± 0.005
PHC-16-†	110K	0.210 ± 0.013	412K	0.172 ± 0.003

to overfit the training data (see SI for learning curves), having more parameter efficient models with the *same* architecture led to overall better performance. To further study the relation between n and model regularization, we trained the same model-architecture but with a much smaller embedding sizes of 64. Within this setting of under-parameterized models, the GNN with $n = 1$ performs best on the train/val/test dataset, followed by the model with increasing PHM-dim. This shows that, in a heavily underfitting setting, merely increasing the HC algebra dimension proves to be detrimental. Additionally, we empirically observe that models that can learn the multiplication rule from the training data (PHC-2 and PHC-4) outperform the complex- and quaternion-valued models (PHC-2-C and PHC-4-Q) in the `OGB` benchmarks.

Increasing n for a Fixed Parameter Budget For our next experiment we examine the test performance of our models with increasing hyperparameter n, while constraining the parameter budget to approximately 100K and 400K parameters [9]. We design a fixed parameter-budget experiment to explore the expressiveness of the hypercomplex embeddings independently of the regularization effect investigated above, as all models possess the same overfitting capacity. This also constitutes a realistic scenario on the production level, where a constrained and low model memory footprint is crucial [37].
A feature of our proposed PHC-GNN is the ability to increase the embedding size of hidden layers for larger hyperparameter n without increasing the parameter count. Table 2 shows the results of experiments conducted on the `ZINC` dataset for a fixed-length hypercomplex GNN architecture, with `L={4,16}` message passing- and 2 downstream-layers. The models differ merely in the embedding sizes, which are chosen so that the total parameter count respects the fixed budget. We observe that the models making use of the PHM-layer outperform the "real"-valued baseline, that uses standard FC layers. Particularly, being able to increase the embedding size seems to strengthen the performance of PHC-models on the test dataset. Nevertheless, we discover that above a certain value for the PHM-dimension n the performance deteriorates. One possible explanation for this behaviour lies in the learning dynamics between the set of contribution and weight matrices $\{\mathbf{C}_i, \mathbf{W}_i\}_{i=1}^{n}$ through the sum of Kronecker products in each PHM-layer (4). With increasing n, the initialization rule in (6) returns n (increasingly) sparse contribution matrices, which seem to negatively affect the learning behaviour for the PHC-$\{8, 10, 16\}$ models.

For example, in the $n = 16$ scenario, exactly 16 elements from each $\mathbf{C}_i$ matrix are non-zero, in comparison to the remaining $16 \cdot 15 = 240$ zero elements. This aggravates the learning process as the weight-sharing achieved by the i^{th} Kronecker product in (4) is not fully exploiting the interaction between all "algebra components". Using the initialization described in (7) enhanced the performance as shown in the undermost part of Table 2 and displayed in Figure 1 across the datasplits. Moreover, we are able to further improve performance of large-n models by adopting the sparse weight-decay regularization described in (10). We refer to the SI for a more thorough discussion.
Another reason for the performance decline of models with larger n, even when utilizing the different initialization scheme, is related to the complexity ratio of the PHM weight matrix $\mathbf{U}_i$ in (4). Recall that

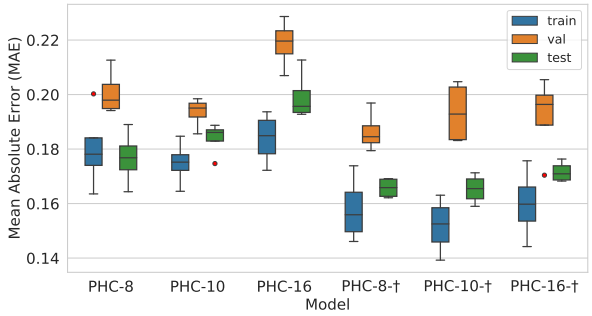


Fig. 1. Boxplot distribution from the datasplits over 4 runs of the 400K PHC-models with $n = 8, 10, 16$. Models-† that utilize the initialization strategy from (7) obtain better performance on the splits. Outliers are marked as a red points.

Table 3. Performance results of our model on molecular property prediction datasets against: DGN [2], PNA [7], GIN, GCN, and DeeperGCN [28] and DeeperGCN/GIN-FLAG [26]. The results for GIN and GCN are reported from [20,9]. Model performances marked with * include a virtual-node [12] in their underlying method.

	OGBG-MOLHIV		OGBG-MOLPCBA		ZINC	
MODEL	ROC-AUC ↑ (%)	# PARAMS	AP ↑ (%)	# PARAMS	MAE ↓	# PARAMS
DGN	**79.70 ± 0.97**	114K	28.85 ± 0.30	6,732K	**0.168 ± 0.003**	98K
PNA	79.05 ± 1.32	326K	28.38 ± 0.35	6,550K	0.188 ± 0.004	95K
GIN	77.07 ± 1.49*	3,336K	27.03 ± 0.23*	3,374K	0.387 ± 0.015	103K
GCN	76.06 ± 0.97	537K	24.24 ± 0.34*	2,017K	0.459 ± 0.006	103K
DEEPERGCN	78.58 ± 1.17	532K	27.81 ± 0.38*	5,550K	–	–
GIN+FLAG	77.48 ± 0.96*	3,336K	28.34 ± 0.38*	3,374K	–	–
DEEPERGCN+FLAG	79.42 ± 1.20*	531K	28.42 ± 0.43*	5,550K	–	–
PHC-GNN (OURS)	79.34 ± 1.16	111K	**29.47 ± 0.26**	1,169K	0.185 ± 0.008	106K

$\mathbf{U}_i$ consists of $\frac{kd}{n} + n^3 = \frac{k^2}{n} + n^3$ trainable parameters, where we assume $k = d$, the two contributions reflecting the trainable weight and contribution matrices, respectively. Now, given a fixed parameter budget, the allocation for the contribution matrices grows on a cubic scale with n, limiting the ability to increase the embedding size k, which plays a crucial role in the feature transformation through the weight matrices. As n grows, an increasingly higher share of parameters are allocated to the contribution matrices, and for large enough n, this negatively affects the learning behaviour of our models, as in the 100K case for $n = \{8, 6, 10\}$ in Table 2.

Finally, we compare our models with the current best performing algorithms on the datasets analyzed above. Table 3 shows that our GNNs are among the top-3 models in all datasets, and it defines a new state-of-the-art on `ogbg-molpcba`. Particularly significant is the comparison with the GIN+FLAG model. FLAG [26] is an adversarial data augmentation strategy which accomplishes a data-dependent regularization. Since, as we remarked in Section 4.6, our models can be considered as a generalization of GIN, we observe that our GNNs outperform the FLAG regularization strategy applied on the same underlying learning strategy.

6 Conclusion

We have introduced a model class we named *Parameterized Hypercomplex Graph Neural Networks*. Our class of models extends to the graph setting the expressive power and the flexibility of (generalized)

hypercomplex algebras. Our experiments showed that our models implement a powerful and flexible approach to regularization with a minimal amount of hyperparameter tuning needed. We have empirically shown that increasing the dimension of the underlying algebra leads to memory-efficient models (in terms of parameter-saving) and performance benefits, consistently outperforming the corresponding real-valued model, both for fixed architecture and fixed parameter budget. We have studied the learning behaviour for increasing algebra-dimension n, and addressed the sparsity phenomenon that manifests itself for large n, by introducing a different initialization strategy and an additional regularization scheme. Finally, we have shown that our models reach state-of-the-art performance on all graph-prediction benchmark datasets.

In this work, we have undertaken the first thorough study on the applicability of higher dimensional algebras in the realm of GNNs. Given the very promising results we have obtained with a relatively simple base architecture, it would be worthwhile to extend to the hypercomplex domain the recent progresses that have been achieved in "real" graph representation learning. For example, it would be interesting to improve the expressivity of our model by learning the aggregation function for the local message passing.

Code Availability

Source code of the proposed method is openly available on GitHub at
`https://github.com/bayer-science-for-a-better-life/phc-gnn`.

Conflicts of Interest

There are no conflicts to declare.

Acknowledgements

F.N acknowledges funding from the European Research Commission (ERC CoG 772230 "ScaleCell"). M.B and D.A.C acknowledge funding from the Bayer AG Life Science Collaboration ("Explainable AI","DeepMinDS"). T.L acknowledges funding from the Bayer AG PhD scholarship.
T.L and M.B would like to thank Robin Winter for helpful discussions.

Appendix

A Additional Model Implementation Details

In this section we report further details regarding the implementation of our class of graph neural network models.

A.1 Dropout

Dropout [19,38] is a simple, yet very effective regularization technique to prevent hidden neuron units from excessively co-adapting. Randomly excluding certain units in a neural network during training leads to more robust features, which are meaningful in conjunction with several random subsets of neurons. In our work we implement two dropout strategies. First, in the spirit of the hypercomplex approach, we randomly zero-out entire hidden units with their n algebra components. Explicitly, given an embedding matrix $\mathbf{H}^{b\times n\times m}$, we randomly sample (during training) a Bernoulli-mask $\mathbf{B}$ of shape $(b, 1, m)$ with probability $(1-p)$ and multiply the mask element-wise with $\mathbf{H}$ using broadcasting. The dropped tensor is further multiplied element-wise by a factor of $\frac{1}{1-p}$ to maintain the expected output values when dropout is turned off at inference time. As an alternative, we also implemented the commonly used dropout by randomly sampling the dropout mask from the flattened embedding 3D tensor, i.e., $\mathbf{H} \in \mathbb{R}^{b\times nm}$.
In our experiments, for fixed probability p, we did not observe a performance difference between the two approaches.

A.2 Parameter Initialization for Weight Matrices

We initialized the component weight matrices as initially described in complex- and quaternion Neural Networks [40,33]. Both works start with the decomposition of a variance term as

$$\mathrm{Var}(W) = \mathbb{E}(|W|^2) - [\mathbb{E}(|W|)]^2 \ , \tag{17}$$

where $[\mathbb{E}(|W|)]^2 = 0$ since the weight distribution is symmetric around 0. As derived in [33], W follows a Chi-distribution with $n = 4$ degrees of freedom in the quaternion case, and $n = 2$ degrees of freedom in the complex case. To adapt the initialization procedure from [13], the standard deviation in our cases is defined as

$$\sigma = \sqrt{\frac{2}{n \times (n_{in} + n_{out})}} \ . \tag{18}$$

We follow Algorithm 1 described in [33] with our defined standard deviation σ to initialize the weight matrices $\{\mathbf{W}_i\}_{i=1}^{n}$ of a PHM-layer. As an alternative, our implementations also include the initialization of each weight matrix $\mathbf{W}_i$ seperately using the Glorot or He initialization scheme [13,16].

A.3 Loss Function

Given a dataset of N graphs, specified by node and edge features, with their corresponding target labels $\mathcal{D} = \{(\mathbf{X}_i, \mathbf{E}_i, \mathbf{y}_i)\}_{i=1}^{N}$, we define the loss function as

$$\begin{aligned}\mathcal{L}_{\text{total}} = \frac{1}{N}\sum_{i=1}^{N}[&\mathcal{L}_{\text{task}}\left(\mathbf{y}_i, f_{\boldsymbol{\Theta}}(\mathbf{X}_i, \mathbf{E}_i)\right) \\ &+ \lambda_1 L_2(\boldsymbol{\Theta}_W) + \lambda_2 L(\boldsymbol{\Theta}_C)] \ .\end{aligned} \tag{19}$$

Here, $\mathcal{L}_{\text{task}}$ is the task-dependent loss function, $f_{\boldsymbol{\Theta}}$ is the output function learned by our GNN, $\mathbf{y}$ represent the target label and $L(\boldsymbol{\Theta}_C)$ is defined in Equation (10) in the main text, which controls the sparsity regularization on all the contribution matrices $[\{\mathbf{C}_i\}_{i=1}^{n}]$. Finally, $L(\boldsymbol{\Theta}_W)$ is the regularization term applied to all weight matrices $[\{\mathbf{W}_i\}_{i=1}^{n}]$ of our GNN and defined in Equation (9).

Table 4. Network architectures and hyperparameters for different models/datasets. The α column describes the aggregation method for gathering neighboring node embeddings. The `sc-type` column indicates which embedding are used for the skip-connection (see Eq. (14) in the main text). "Previous" means that the skip-connection is done with the embedding from the previous layer, i.e., $(l-1)$ and "initial" refers to the embedding from hidden layer 0. Abbreviations are as follows: MP = message passing, DN = downstream network. The column MP-MLP describes whether a 2-layer MLP is used in the message passing layer, as described in Eq. (13) in the main text. If `MP-MLP=False`, only a 1-layer MLP is used for feature transformation. We recall that the PHM-layer is used throughout the network (both MP and DN). The tuple (γ, p) deschribe the decay factor for adjusting the learning rate after p patience epochs if the validation performance has not improved. The tuple (λ_1, λ_2) refers to the regularization coefficients for the weight and contribution matrices, respectively, as described in (19). The (maximum) number of epochs for `ZINC, MNIST, CIFAR10` was set to 1000, but the training would be interrupted if the minimal learning rate of 10^{-6} was reached or if the execution time exceeded 72 hours.

Dataset	Model	α	sc-layer	MP-layers	MP-MLP	MP-dropout	DN-layers	DN-dropout	lr	(λ_1, λ_2)	(γ, p)	epochs
`ogbg-molhiv`	PHC-n	softmax	initial	$[200] * 2$	`True`	$[0.3] * 2$	$[128, 32]$	$[0.3, 0.1]$	$1 \cdot 10^{-3}$	$(10^{-1},\ 0)$	$(0.75, 5)$	50
`ogbg-molpcba`	PHC-n-default	sum	initial	$[512] * 7$	`False`	$[0.1] * 7$	$[768, 256]$	$[0.3, 0.2]$	$5 \cdot 10^{-4}$	$(10^{-4},\ 0)$	$(0.75, 5)$	150
	PHC-n-shallow	sum	initial	$[64] * 7$	`True`	$[0.1] * 7$	$[64, 32]$	$[0.3, 0.2]$	$5 \cdot 10^{-4}$	$(0,\ 0)$	$(0.75, 5)$	150
`ogbg-molppa`	PHC-6	softmax	initial	$[900] * 5$	`True`	$[0.2] * 5$	$[600, 300]$	$[0.2, 0.1]$	$1 \cdot 10^{-3}$	$(0,\ 0)$	$(0.75, 10)$	250
`ZINC`	PHC-1	sum	previous	$[104] * 14$	`True`	$[0.0] * 14$	$[100, 50]$	$[0.2, 0.1]$	$1 \cdot 10^{-3}$	$(10^{-2},\ 0)$	$(0.5, 10)$	1000
	PHC-2	sum	previous	$[144] * 14$	`True`	$[0.1] * 14$	$[180, 100]$	$[0.2, 0.1]$	$1 \cdot 10^{-3}$	$(10^{-2},\ 0)$	$(0.5, 10)$	1000
	PHC-3	sum	previous	$[177] * 14$	`True`	$[0.1] * 14$	$[180, 102]$	$[0.2, 0.1]$	$1 \cdot 10^{-3}$	$(10^{-2},\ 0)$	$(0.5, 10)$	1000
	PHC-4	sum	previous	$[202] * 14$	`True`	$[0.1] * 14$	$[224, 124]$	$[0.2, 0.1]$	$1 \cdot 10^{-3}$	$(10^{-2},\ 0)$	$(0.5, 10)$	1000
	PHC-5	sum	previous	$[225] * 14$	`True`	$[0.1] * 14$	$[225, 115]$	$[0.2, 0.1]$	$1 \cdot 10^{-3}$	$(10^{-2},\ 0)$	$(0.5, 10)$	1000
	PHC-8	sum	previous	$[272] * 14$	`True`	$[0.1] * 14$	$[280, 160]$	$[0.2, 0.1]$	$1 \cdot 10^{-3}$	$(10^{-2},\ 0)$	$(0.5, 10)$	1000
	PHC-10	sum	previous	$[290] * 14$	`True`	$[0.1] * 14$	$[330, 220]$	$[0.2, 0.1]$	$1 \cdot 10^{-3}$	$(10^{-2},\ 0)$	$(0.5, 10)$	1000
	PHC-16	sum	previous	$[304] * 14$	`True`	$[0.1] * 14$	$[304, 176]$	$[0.2, 0.1]$	$1 \cdot 10^{-3}$	$(10^{-2},\ 0)$	$(0.5, 10)$	1000
`MNIST / CIFAR10`	PHC-1	mean	previous	$[84] * 4$	`False`	$[0.1] * 4$	$[256, 128]$	$[0.2, 0.1]$	$1 \cdot 10^{-3}$	$(10^{-3},\ 0)$	$(0.5, 10)$	1000
	PHC-2	mean	previous	$[140] * 4$	`False`	$[0.1] * 4$	$[256, 128]$	$[0.2, 0.1]$	$1 \cdot 10^{-3}$	$(10^{-3},\ 0)$	$(0.5, 10)$	1000
	PHC-3	mean	previous	$[195] * 4$	`False`	$[0.1] * 4$	$[256, 128]$	$[0.2, 0.1]$	$1 \cdot 10^{-3}$	$(10^{-3},\ 0)$	$(0.5, 10)$	1000
	PHC-4	mean	previous	$[224] * 4$	`False`	$[0.1] * 4$	$[256, 128]$	$[0.2, 0.1]$	$1 \cdot 10^{-3}$	$(10^{-3},\ 0)$	$(0.5, 10)$	1000
	PHC-5	mean	previous	$[250] * 4$	`False`	$[0.1] * 4$	$[256, 128]$	$[0.2, 0.1]$	$1 \cdot 10^{-3}$	$(10^{-3},\ 0)$	$(0.5, 10)$	1000

B Architecture, Hyperparameter and Training Strategy

Table 4 lists the architectures and the hyperparameter setting for all the experiments presented in the main text and in the Supplementary Information (SI). In the remainder of this section we report some additional training strategy details specific to each dataset. In all our experiments we used ReLU as an activation function and ADAM optimizer [24] with decreasing learning-rate based on a plateau-scheduler or step-scheduler. In each dataset, we execute t runs, where the first run starts with the random seed 0. Subsequent runs are then executed with an increasing random seed.

B.1 Open Graph Benchmark (OGB)

Network Architectures `ogbg-molpcba` For the medium-scale `ogbg-molpcba` dataset, our default models include 7 message passing layers.In our experiments, we found that employing a MLP with 2 layers, as deployed in [45], results in inferior validation performance. Therefore, we only included 1 PHM-layer, but set a larger embedding size of 512. After the softattention graph-pooling , we employ a 2-layer MLP with 786 and 256 units, respectively. In each of the 5 runs, we trained for 150 epochs with an initial learning rate of 0.005 and weight-decay regularization $\lambda_1 = 10^{-5}$ but no sparsity regularization on the contribution matrices, i.e., $\lambda_2 = 0$. We used ADAM optimizer [24] and multiplied the learning rate with 0.75 after 5 epochs of patience if the validation performance did not improve. Additionally, we used gradient clipping with maximum l_2-norm of value 2.0. The α-function, i.e., the aggregation schema, such as {min, max, mean, sum, softmax} was set to "sum".

The models in Table 5 were trained with the same setting, but the embedding sizes for the 7 message passing layers were set to 64, and the 2 layers in the downstream network consists of 64 and 32 units, respectively. Additionally, the shallow models utilize a 2-layer MLP in each message passing layer, as opposed to the default models reported in Table 1.

Table 5. Results of the GNN on the `ogb-molpcba` graph property prediction dataset for a shallow model with embedding size 64. The number of message passing and downstream layers is set to 7 and 2, respectively.

Model	# Params	Precision-Recall (%) ↑		
		Training	Validation	Test
PHC-1	112K	20.47 ± 0.18	21.56 ± 0.18	$\mathbf{20.88 \pm 0.18}$
PHC-2	92K	18.08 ± 0.32	20.26 ± 0.31	19.80 ± 0.29
PHC-3	100K	17.35 ± 0.22	19.72 ± 0.32	19.30 ± 0.20
PHC-4	109K	16.08 ± 0.15	18.74 ± 0.12	18.31 ± 0.14
PHC-5	124K	16.17 ± 0.42	18.55 ± 0.40	18.28 ± 0.34

Table 6. Results of the PHC-1 on the `ogbg-molpcba` graph classification. The model with larger dropout (PHC-1-$*$) is prevented from overfitting the training data but also obtains lower test performance metric. Nonetheless, the best performing model is the PHC-2 model (as reported in the main article) *without* additional regularization.

Model	# Params	Precision-Recall (%)		
		Training	Validation	Test
PHC-1	3.15M	56.01 ± 0.76	30.38 ± 0.28	29.17 ± 0.16
PHC-1-$*$	3.15M	39.09 ± 0.34	29.99 ± 0.13	29.12 ± 0.26
PHC-2	1.69M	50.08 ± 0.29	30.68 ± 0.25	$\mathbf{29.47 \pm 0.26}$

B.2 Benchmarking GNNs

In this case, we strove to meet the parameter budgets of 100K for the two Computer Vision datasets, as well as of 100/400K for the molecular property prediction dataset. We did not perform any extensive hyperparameter search, but we limited ourselves to adapt the default configurations provided by [9].

C Additional Details about Experiment Section

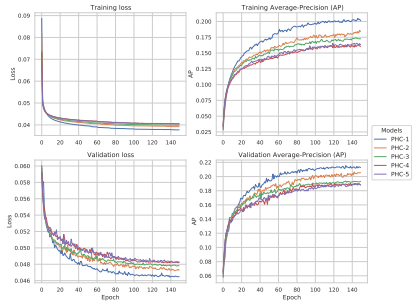
In this section we present further details concerning the experiments presented in Section 5 in the main text. Figure 2 reports the train/validation curves for the experiment described at the beginning of Section 5, in the setting of "increasing n for a fixed network architecture". Figure 2(a) shows that in a heavily underparameterized setting increasing the algebra dimension n *without* correspondingly increasing the embedding size leads to a worse perfomance, as the corresponding models are even more parameter-scarce. We list the train/validation/test performance of the models in Table 5. On the contrary, we observe in 2(b) that increasing the algebra dimension n *prevents* overfitting, which can be observed for the PHC-1 model, and yields a performance improvement.

D Further Experiments

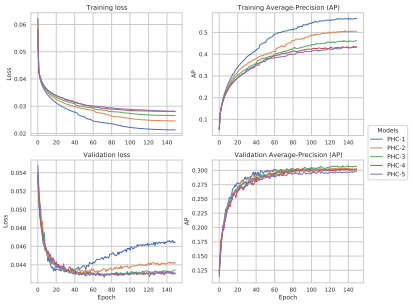
In this section we present results of further experiments that, for space constraint, were not included in the main text.

D.1 PHC is a Better Regularizer than Dropout

In this work, we provided evidence that our graph implementation of the PHM-layer acts as an effective and versatile regularizer. From this perspective, it is then natural to ask how the PHM-layer performs in comparison to other regularization techniques. Although we do not fully answer this question – which probably would require a dedicated work – we begin addressing it here. Namely, we examine whether a PHC-1 model regularized with a higher dropout value will outperform the PHC-($n > 2$) models. Table 6 and Figure 3(a) show that indeed increasing dropout does regularize model. In fact, the validation loss for the regularized PHC-1-$*$ model reaches a lower value as compared to PHC-1 (see bottom-left plot in Figure 3(a)). However, the overall test performance does not improve, and as a consequence PHC-1-$*$ cannot match the performance of the hypercomplex models. We increased the dropout in the message passing layers, i.e., `MP-dropout=`$[0.4] * 14$ and in the downstream layers, i.e., `DN-dropout=`$[0.5, 0.2]$, following the scheme presented in Table 4.

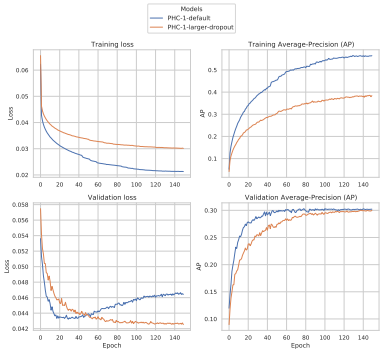


(a) Learning curves of the shallow PHC models (one run, seed= 0). The embedding size for every hidden layer is set to 64. The PHC-1 model performs best in the *underparamterized* setting (see also Table 5).

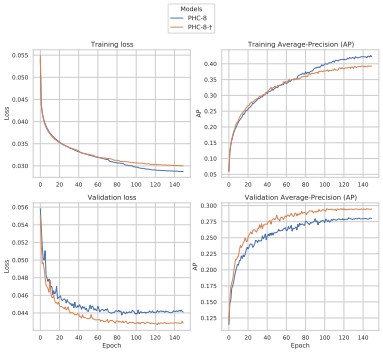


(b) Learning curves of the reported (in the main text) PHC models (one run, seed=0). The embedding sizes are set to 512 for every hidden layer. The PHC-1 model is overfitting the training data in this *overparamezerized* setting, while the PHC$-n$ models with $n > 1$ obtain better generalization performance.

Fig. 2. Learning curves for the PHC-models trained on the `ogbg-molcpba` dataset. In the *overparameterized* setting, using hypercomplex multiplication aids in better generalization and prevents from overfitting the training data.



(a) Learning curves for PHC-1 models trained with default settings, and one with larger dropout. The model with larger dropout is not overfitting the training data.



(b) Learning curves for PHC-8 models trained with default initialization of contribution matrices (PHC-8) and random initialization (PHC-8-†). The PHC-8-† model outperforms the default model on the validation set, as well as test dataset (see Table 7).

Fig. 3. Learning curves for the PHC-1 and PHC-8 trained on the `ogbg-molcpba` dataset.

D.2 Initialization Strategy for Large n

As mentioned in the main text, we observed that the PHC-n model performance deteriorates when n is above a certain value. This value highly depends on the dataset and the task at hand, but from our experience, the performance loss occurs already when n is at least 8. We address this phenomenon through a different initialization strategy (in Eq. (7)) for the algebra's multiplication, which aims at compensating the sparsity in the hypercomplex product introduced by the *standard* initialization strategy (Equation (6) in the main text). Table 7 shows the performance improvement of model PHC-8-† (non-sparse initialization) compared to model PHC-8 (sparse initialization) on the `ogbg-molpcba` dataset. Since this dataset is

Table 7. Results of the PHC-8 on the `ogbg-molpcba` graph classification. Displayed are the model performances with default (PHC-8) and random initialiaztion strategy (PHC-8-†) for the contribution matrices.

Model	# Params	Precision-Recall (%)		
		Training	Validation	Test
PHC-8	689K	41.89 ± 0.34	28.10 ± 0.18	27.00 ± 0.14
PHC-8-†	689K	39.42 ± 0.45	29.59 ± 0.13	$\mathbf{28.73 \pm 0.39}$

Table 8. Sample sizes for the splits in each dataset used in our experiments. For more details about split strategy and graph statistics, we refer to [20,9].

Dataset	Training	Validation	Test	Domain
`ogbg-molhiv`	32,901	4,113	4,113	Chemistry
`ogbg-molpcba`	350,343	43,793	43,797	Chemistry
`ogbg-molppa`	78,200	45,100	34,800	Biology
`ZINC`	10,000	1,000	1,000	Chemistry
`MNIST`	55,000	5,000	10,000	Computer Vision
`CIFAR10`	45,000	5,000	10,000	Computer Vision

considered as medium-scale benchmark with 350,343 training samples as listed in Table 8, and we only reported performances for the random initialization strategy on the small `ZINC` dataset with 400K model parameters, we provide further evidence that leveraging the random initialization scheme is also beneficial for models with more than 400K (learnable) parameters and trained on larger datasets.

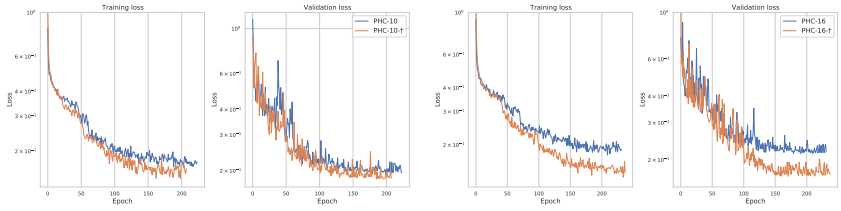
Furthermore, we show the learning curves for the PHC-10 and PHC-16 models trained on the (smaller) `ZINC` dataset in Figure 4. The learning curves display better learning ability for the PHC-GNN when the contribution matrices are uniform randomly initialized, leading to denser contribution matrices, which subsequently allow more interaction between the hypercomplex components in the sum of Kronecker products.

Table 9. Results of the PHC-10-† and PHC-16-† on the `ZINC` graph regression (400K parameters). With increasing sparse regularization factor λ_2, the performance across the datasplits improve. In total, 4 runs are executed and the mean performance measures are displayed. The entries in row 1^{st} to 6^{th}, correspond to the PHC-10-† model. The entries afterwards refer to the PHC-16-† model.

λ_2	Mean Absolute Error (MAE) ↓		
	Training	Validation	Test
0	0.152 ± 0.009	0.193 ± 0.010	0.165 ± 0.005
10^{-5}	0.155 ± 0.004	0.187 ± 0.015	0.165 ± 0.002
10^{-4}	0.152 ± 0.005	0.186 ± 0.012	0.164 ± 0.004
10^{-3}	0.154 ± 0.003	0.188 ± 0.015	0.163 ± 0.002
$\mathbf{10^{-2}}$	0.148 ± 0.010	0.184 ± 0.009	$\mathbf{0.163 \pm 0.001}$
10^{-1}	0.156 ± 0.008	0.193 ± 0.012	0.166 ± 0.007
0	0.160 ± 0.011	0.192 ± 0.013	0.172 ± 0.003
10^{-5}	0.163 ± 0.015	0.191 ± 0.013	0.165 ± 0.005
10^{-4}	0.162 ± 0.010	0.197 ± 0.012	0.172 ± 0.007
10^{-3}	0.158 ± 0.004	0.197 ± 0.005	0.170 ± 0.002
$\mathbf{10^{-2}}$	0.151 ± 0.012	0.181 ± 0.007	$\mathbf{0.164 \pm 0.006}$
10^{-1}	0.168 ± 0.006	0.200 ± 0.006	0.174 ± 0.007

D.3 Additional Regularization Strategy for Large n

The large-n initialization addresses the issue of sparsity in the algebra's multiplication that arises for large n. As briefly noted in the main body of the work, the resulting model might suffer from some



(a) `ZINC` performance of the PHC-10 and PHC-10-† models.

(b) `ZINC` performance of the PHC-16 and PHC-16-† models.

Fig. 4. Model performance increases if the random initialization is used (PHC-n-†) compared to the default initialization (PHC-n). The improved performance difference due to the random (dense) initialization of contribution matrices is especially noticeable in the case of $n = 16$, where denser contribution matrices alleviate training by allowing more weight-sharing in the final weight matrix that will be used for the affine transformations. (See also Figure 6).

degree of overfitting. We address this by including a new regularization term (Equation (10) in the main text). The effect of this term is to control the sparsity of the contribution matrices defining the algebra's product, allowing us to find the optimal amount of product sparsity for a given task. Table 9 lists the performance on the `ZINC` dataset of PHC-n models for $n = 10, 16$, for several values of the coefficient λ_2 of the additional regularization term. For $n = 10$, we observe an essentially constant performance of the differently regularized models. For $n = 16$, instead, we note a performance improvement of the regularized models. In Figure 5 we report the learning curves for these experiments.

For the PHC-16 model, we visualize the effect of the sparsity regularization coefficient λ_2 on the matrix defining the algebra's product (defined in Equation (4) in the main text). Figure 6 depicts a heat map for the coefficients of the 304×304 matrix defining the weight matrix $\mathbf{U}$ obtained after applying the sum of Kronecker products. Recall that this weight matrix $\mathbf{U}$ is utilized for the affine transformation in the PHM-layer.
Here, we define the sparsity s of the matrix $\mathbf{U} \in \mathbb{R}^{k \times d}$ as follows:

$$s(\mathbf{U}) = 1 - \frac{1}{kd} \sum_{i=1}^{k} \sum_{j=1}^{d} \left| \mathbf{U}_{[i,j]} \right| \ . \tag{20}$$

Values of s closer to 1 indicate that the weight-matrix $\mathbf{U}$ is more sparse.

Table 10. Execution time [$\frac{\text{seconds}}{\text{epoch}}$] from the models reported in the main article. The models for `hiv,pcba` were trained on a single NVIDIA Tesla V100-32GB GPU and the models for `ZINC` (400K) were trained on a single NVIDIA Tesla V100-16GB GPU.

Model	Dataset		
	`ogbg-molhiv`	`ogbg-molpcba`	`ZINC`
PHC-1	15.64	111.60	8.07
PHC-2	21.34	150.24	17.55
PHC-3	27.42	196.85	22.16
PHC-4	24.62	239.49	31.82
PHC-5	37.92	312.75	33.03
PHC-8	60.30	388.93	62.00
PHC-10	–	–	66.97
PHC-16	–	–	117.38

We see that when the contribution matrices of the model are sparsily initialized, only the diagonal terms are activated. That is, the model collapses to a quasi-real-valued network, in which the hypercomplex

Table 11. PHC-GNNs results on computer vision graph datasets. The performance measure is the accuracy on the test dataset.

Model	MNIST		CIFAR10	
	# Params	Acc. (%) ↑	# Params	Acc. (%) ↑
PHC-1	101.3K	97.08 ± 0.10	101.5K	66.32 ± 0.15
PHC-2	99.4K	97.32 ± 0.08	99.7K	66.79 ± 0.10
PHC-3	111.2K	97.32 ± 0.05	111.6K	$\mathbf{66.80 \pm 0.23}$
PHC-4	106.8K	$\mathbf{97.36 \pm 0.06}$	107.3K	66.47 ± 0.46
PHC-5	104.4K	97.24 ± 0.17	104.9K	66.42 ± 0.27

components do not mix with each other. When the contribution matrices of the model are instead uniformely initiated, the distribution of activated values is less concentrated, and more interaction between the hypercomplex components is present. Finally, turning on λ_2 causes the model to zero-out some matrix coefficients, while preserving the component-mixing.

D.4 Computing Performance

We report in Table 10 the execution time in s/epoch for several PHC-n models in three datasets. We observe an increasing computational time cost when n increases. This is due to the linearly increasing number of Kronecker products necessary to be computed (n for PHC-n). Our models were implemented in `PyTorch version 1.7.1` [34] which does not provide a CUDA implementation of the Kronecker product. We used our customized PyTorch implementation of the Kronecker product.

For the above reason, it is also important to mention that although our method is memory-efficient and enables us to reduce model parameters by using the PHM-layer, it involves more floating point operations (FLOPS) by computing n Kronecker products, which is reflected in an increase of the execution time.

D.5 Computer Vision Graph Datasets

For completeness, we report in Table 11 the performance of our PHC-n models on the computer vision graph datasets `MNIST` and `CIFAR10`. All the models satisfy a budget constraint of approximatively 100K parameters, and are trained following the guidelines of [9]. Once again, we observe that hypercomplex models achieve a higher accuracy in the classification tasks than the real-valued PHC-1 model.

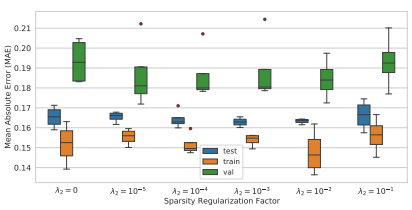
Table 12. Result of our PHC-GNN on the `ogb-molppa` graph property prediction dataset. We only conducted one run on this dataset and we reported the model on the validation set (PHC-6). We compare our model against other methods from the literature. The results for GIN, GIN+VN and GCN are taken from [20]. The FLAG method [26] is applied to GIN and DeeperGCN [28].

Model	# Params	Acc. (%) ↑	
		Validation	Test
DeeperGCN+Flag	2.34M	74.84 ± 0.52	$\mathbf{77.52 \pm 0.69}$
DeeperGCN	2.34M	73.13 ± 0.78	77.12 ± 0.71
GIN+VN+Flag	3.29M	67.89 ± 0.79	72.45 ± 1.14
GIN+VN	3.29M	66.78 ± 1.05	70.37 ± 1.07
GIN	1.84M	65.62 ± 1.07	68.92 ± 1.00
GCN	480K	64.97 ± 0.34	68.39 ± 0.84
PHC-GNN (ours)	1.84M	71.35	75.61

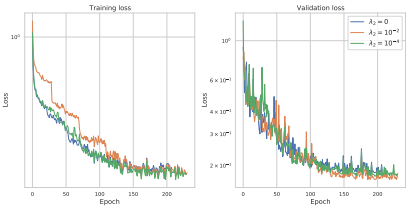
D.6 Experiments on Protein-Protein Association Networks

Finally, we also trained our PHC-GNN on the `ogbg-ppa` dataset, which consists of undirected association neighborhoods. The nodes in each association graph represent proteins, and the prediction task is to classify each association graph into 37 taxonomic groups [20]. In Table 12 we compare our method against models reported in the literature. Also for this dataset, our model shows a strong performance with a lower parameter budget. As the association graphs are densely connected (avg. # edges=$2,266$ and avg. node degree=18.3), choosing an appropriate aggregation function α in the message passing layer is crucial for training and generalization performance of the model. We selected the aggregation schema based on the best performing models, in this case, DeeperGCN [28], which adopted the softmax-aggregation function with learnable temperature factor τ. The softmax aggregation function can be regarded as a combination between the "max" and "mean" aggregation functions, depending on the value of τ. In the `ogbg-ppa` dataset, an aggregation function that tends to select the maximum value of connected neighboring nodes seems to boost the training and generalization performance. We tested the standard aggregation functions, such as "sum" and "mean", but have found that "max" and "softmax" perform better on the validation sets.
For the above reason, models like GCN or GIN, which utilize the "mean" and "sum" aggregators, cannot reach a validation accuracy of 70% even with the inclusion of a virtual node. Such aggregators seem to be non-optimal in very dense association graphs, where most likely only representative *mode* values of neighboring nodes are required to propagate messages to contribute to the final prediction task [45]. The "max" aggregator, on the other hand, is designed to achieve exactly this type of selected node propagation.

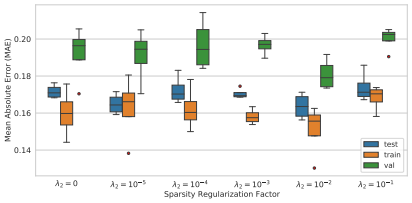
The choice of (appropriate) aggregation function α, as illustrated in Table 4, is a deciding component in every GNN and can vary for each dataset/task. In our work, we focused on the development of a GNN that utilizes feature transformations motivated by the idea of hypercomplex multiplications. For future research, it would be exciting to combine the benefits of hypercomplex feature transformations with learnable aggregation functions to develop even more powerful GNNs, suitable for a larger variety of graph datasets.



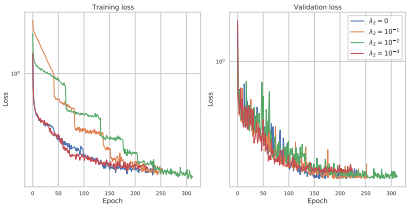
(a) `ZINC` performance of the PHC-10-† model over 4 runs.



(b) Learning curves of the models from the 1st run, i.e., seed=0.

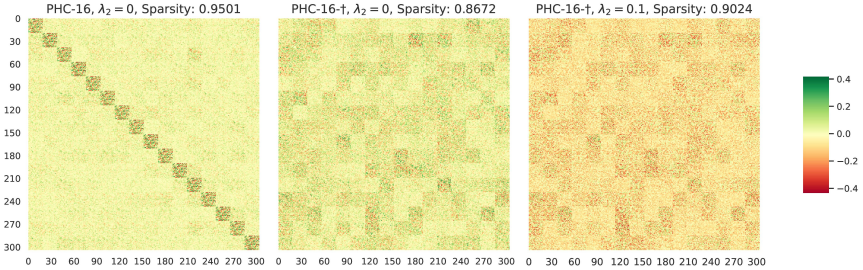


(c) `ZINC` performance of the PHC-16-† model over 4 runs.

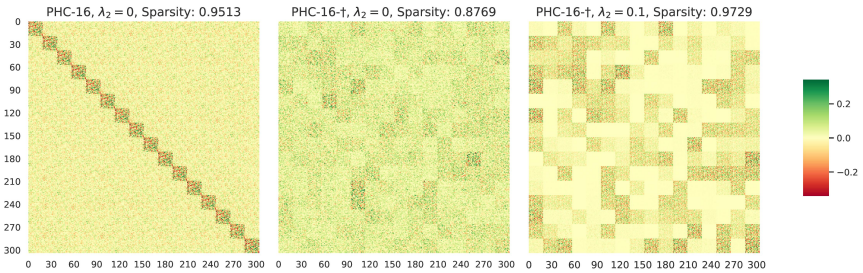


(d) Learning curves of the models from the 1st run, i.e., seed=0.

Fig. 5. Model performance for PHC-10-† (a-b) and PHC-16-† (c-d) models (400K parameter budget) with varying sparsity regularization factor λ_2.



(a) Weight matrix in an initial layer of of the PHC-16 and PHC-16-† networks.



(b) Weight matrix in a deeper layer of the PHC-16 and PHC-16-† networks.

Fig. 6. Weight matrix $\mathbf{U} = \sum_{i=1}^{n} \mathbf{C}_i \otimes \mathbf{W}_i$ used in the affine transformation. The default initialization for the contribution matrices $\mathbf{C}_i$ in the PHC-16 model (first column) leads to a sparser weight-matrix, due to a too restrictive interactions between the matrices $\mathbf{W}_i$.

References

1. Arjovsky, M., Shah, A., Bengio, Y.: Unitary evolution recurrent neural networks. In: International Conference on Machine Learning. pp. 1120–1128 (2016)
2. Beaini, D., Passaro, S., Létourneau, V., Hamilton, W.L., Corso, G., Liò, P.: Directional graph networks (2020)
3. Bronstein, M.M., Bruna, J., LeCun, Y., Szlam, A., Vandergheynst, P.: Geometric deep learning: Going beyond euclidean data. IEEE Signal Processing Magazine **34**(4), 18–42 (Jul 2017). https://doi.org/10.1109/msp.2017.2693418, http://dx.doi.org/10.1109/MSP.2017.2693418
4. Bruna, J., Zaremba, W., Szlam, A., LeCun, Y.: Spectral networks and locally connected networks on graphs (2014)
5. Chami, I., Abu-El-Haija, S., Perozzi, B., Ré, C., Murphy, K.: Machine learning on graphs: A model and comprehensive taxonomy (2021)
6. Chami, I., Ying, Z., Ré, C., Leskovec, J.: Hyperbolic graph convolutional neural networks. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 32, pp. 4868–4879. Curran Associates, Inc. (2019), https://proceedings.neurips.cc/paper/2019/file/0415740eaa4d9decbc8da001d3fd805f-Paper.pdf
7. Corso, G., Cavalleri, L., Beaini, D., Liò, P., Veličković, P.: Principal neighbourhood aggregation for graph nets (2020)
8. Duvenaud, D., Maclaurin, D., Aguilera-Iparraguirre, J., Gómez-Bombarelli, R., Hirzel, T., Aspuru-Guzik, A., Adams, R.P.: Convolutional networks on graphs for learning molecular fingerprints (2015)
9. Dwivedi, V.P., Joshi, C.K., Laurent, T., Bengio, Y., Bresson, X.: Benchmarking graph neural networks (2020)
10. Ganea, O., Becigneul, G., Hofmann, T.: Hyperbolic neural networks. In: Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 31, pp. 5345–5355. Curran Associates, Inc. (2018), https://proceedings.neurips.cc/paper/2018/file/dbab2adc8f9d078009ee3fa810bea142-Paper.pdf
11. Gaudet, C.J., Maida, A.S.: Deep quaternion networks. In: 2018 International Joint Conference on Neural Networks (IJCNN). pp. 1–8 (2018). https://doi.org/10.1109/IJCNN.2018.8489651
12. Gilmer, J., Schoenholz, S.S., Riley, P.F., Vinyals, O., Dahl, G.E.: Neural message passing for quantum chemistry. In: Precup, D., Teh, Y.W. (eds.) Proceedings of the 34th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 70, pp. 1263–1272. PMLR, International Convention Centre, Sydney, Australia (06–11 Aug 2017), http://proceedings.mlr.press/v70/gilmer17a.html
13. Glorot, X., Bengio, Y.: Understanding the difficulty of training deep feedforward neural networks. In: Teh, Y.W., Titterington, D.M. (eds.) AISTATS. JMLR Proceedings, vol. 9, pp. 249–256. JMLR.org (2010), http://dblp.uni-trier.de/db/journals/jmlr/jmlrp9.html#GlorotB10
14. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. Advances in Neural Information Processing Systems pp. 1024–1034 (2017)
15. Hamilton, W.R.: Theory of quaternions. Proceedings of the Royal Irish Academy (1836-1869) **3**, 1–16 (1844), http://www.jstor.org/stable/20489494
16. He, K., Zhang, X., Ren, S., Sun, J.: Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In: Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV). p. 1026–1034. ICCV '15, IEEE Computer Society, USA (2015). https://doi.org/10.1109/ICCV.2015.123, https://doi.org/10.1109/ICCV.2015.123
17. Henaff, M., Bruna, J., LeCun, Y.: Deep convolutional networks on graph-structured data (2015)
18. Hinton, G., Vinyals, O., Dean, J.: Distilling the knowledge in a neural network (2015)
19. Hinton, G.E., Srivastava, N., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Improving neural networks by preventing co-adaptation of feature detectors. CoRR **abs/1207.0580** (2012), http://arxiv.org/abs/1207.0580
20. Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M., Leskovec, J.: Open graph benchmark: Datasets for machine learning on graphs (2020)
21. Hu, W., Liu, B., Gomes, J., Zitnik, M., Liang, P., Pande, V., Leskovec, J.: Strategies for pre-training graph neural networks. In: International Conference on Learning Representations (2020), https://openreview.net/forum?id=HJlWWJSFDH
22. Ioffe, S., Szegedy, C.: Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37. p. 448–456. ICML'15, JMLR.org (2015)
23. Kantor, I.L. and Solodovnikov, A.: Hypercomplex Numbers. Springer-Verlag New York (1989), https://www.springer.com/gp/book/9781461281917
24. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2017)
25. Kipf, T.N., Welling, M.: Semi-Supervised Classification with Graph Convolutional Networks. In: Proceedings of the 5th International Conference on Learning Representations. ICLR '17 (2017), https://openreview.net/forum?id=SJU4ayYgl
26. Kong, K., Li, G., Ding, M., Wu, Z., Zhu, C., Ghanem, B., Taylor, G., Goldstein, T.: Flag: Adversarial data augmentation for graph neural networks (2020)

27. Kukačka, J., Golkov, V., Cremers, D.: Regularization for deep learning: A taxonomy (2017)
28. Li, G., Xiong, C., Thabet, A., Ghanem, B.: Deepergcn: All you need to train deeper gcns (2020)
29. Liu, Q., Nickel, M., Kiela, D.: Hyperbolic graph neural networks. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems. vol. 32, pp. 8230–8241. Curran Associates, Inc. (2019), https://proceedings.neurips.cc/paper/2019/file/103303dd56a731e377d01f6a37badae3-Paper.pdf
30. Montanari, F., Kuhnke, L., Ter Laak, A., Clevert, D.A.: Modeling physico-chemical admet endpoints with multitask graph convolutional networks. Molecules **25**(1) (2020)
31. Nguyen, D.Q., Nguyen, T.D., Phung, D.: Quaternion graph neural networks (2020)
32. Parcollet, T., Morchid, M., Linarès, G.: Quaternion convolutional neural networks for heterogeneous image processing. ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP) pp. 8514–8518 (2019)
33. Parcollet, T., Ravanelli, M., Morchid, M., Linarès, G., Trabelsi, C., Mori, R.D., Bengio, Y.: Quaternion recurrent neural networks. In: International Conference on Learning Representations (2019), https://openreview.net/forum?id=ByMHvs0cFQ
34. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., Garnett, R. (eds.) Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019), https://bit.ly/3q9mtLS
35. Sainath, T.N., Kingsbury, B., Sindhwani, V., Arisoy, E., Ramabhadran, B.: Low-rank matrix factorization for deep neural network training with high-dimensional output targets. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. pp. 6655–6659 (2013). https://doi.org/10.1109/ICASSP.2013.6638949
36. Savarese, P., Maire, M.: Learning implicitly recurrent CNNs through parameter sharing. In: International Conference on Learning Representations (2019), https://openreview.net/forum?id=rJgYxn09Fm
37. Sohoni, N.S., Aberger, C.R., Leszczynski, M., Zhang, J., Ré, C.: Low-memory neural network training: A technical report (2019)
38. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., Salakhutdinov, R.: Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research **15**(56), 1929–1958 (2014), http://jmlr.org/papers/v15/srivastava14a.html
39. Tay, Y., Zhang, A., Tuan, L.A., Rao, J., Zhang, S., Wang, S., Fu, J., Hui, S.C.: Lightweight and efficient neural natural language processing with quaternion networks (2019)
40. Trabelsi, C., Bilaniuk, O., Zhang, Y., Serdyuk, D., Subramanian, S., Santos, J.F., Mehri, S., Rostamzadeh, N., Bengio, Y., Pal, C.J.: Deep complex networks. In: International Conference on Learning Representations (2018), https://openreview.net/forum?id=H1T2hmZAb
41. Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: International Conference on Learning Representations (2018), https://openreview.net/forum?id=rJXMpikCZ
42. Weisfeiler, B.a.L.A.: A reduction of a graph to a canonical form and an algebra arising during this reduction. Nauchno-Technicheskaya Informatsia **2(9)**, 12–16 (1968)
43. Wu, J., Xu, L., Wu, F., Kong, Y., Senhadji, L., Shu, H.: Deep octonion networks. Neurocomputing **397**, 179 – 191 (2020). https://doi.org/https://doi.org/10.1016/j.neucom.2020.02.053, http://www.sciencedirect.com/science/article/pii/S0925231220302435
44. Wu, Z., Pan, S., Chen, F., Long, G., Zhang, C., Yu, P.S.: A comprehensive survey on graph neural networks. IEEE Transactions on Neural Networks and Learning Systems **32**(1), 4–24 (Jan 2021). https://doi.org/10.1109/tnnls.2020.2978386, http://dx.doi.org/10.1109/TNNLS.2020.2978386
45. Xu, K., Hu, W., Leskovec, J., Jegelka, S.: How powerful are graph neural networks? In: International Conference on Learning Representations (2019), https://openreview.net/forum?id=ryGs6iA5Km
46. Xu, K., Li, C., Tian, Y., Sonobe, T., Kawarabayashi, K.i., Jegelka, S.: Representation learning on graphs with jumping knowledge networks. In: Dy, J., Krause, A. (eds.) Proceedings of the 35th International Conference on Machine Learning. Proceedings of Machine Learning Research, vol. 80, pp. 5453–5462. PMLR, Stockholmsmässan, Stockholm Sweden (10–15 Jul 2018), http://proceedings.mlr.press/v80/xu18c.html
47. Zhang, A., Tay, Y., Zhang, S., Chan, A., Luu, A.T., Hui, S., Fu, J.: Parameterization of hypercomplex multiplications. In: International Conference on Learning Representations (2021), https://openreview.net/forum?id=rcQdycl0zyk
48. Zhou, J., Cui, G., Zhang, Z., Yang, C., Liu, Z., Wang, L., Li, C., Sun, M.: Graph neural networks: A review of methods and applications (2019)

2.3 Publication 3: Representation Learning on Biomolecular Structures Using Equivariant Graph Attention

Full Reference: ***Le, Tuan****; Noé, Frank; & Clevert, Djork-Arné (2022). Representation Learning on Biomolecular Structures Using Equivariant Graph Attention. Proceedings of the First Learning on Graphs Conference, PMLR 198:30:1-30:17, 2022.*

Licence: CC BY 4.0

Journal/Conference: Learning on Graphs Conference

Link to Article: `https://proceedings.mlr.press/v198/le22a.html`

Source Code: `https://github.com/Bayer-Group/eqgat`

Paper's main contributions:

- We introduce a computationally efficient equivariant GNN that leverages geometric information by operating on vector features in Cartesian space.
- We implemenent a novel feature attention mechanism to propagate neighbouring node features and we define equivariant operations to combine vector features in a geometrically meaningful way.
- We benchmark our porposed architecture on large molecular systems such as protein complexes and show its efficacy mostly relevant to industrial applications.

Author's contribution to the paper:

- Conceptualization of the original idea and its application to molecular representation learning.
- Derivation and implementation of the method.
- Design and evaluation of experiments, data curation and analysis.
- Preparation and creation of initial draft and visualizations.
- Revision of the manuscript for the final conference submission.

The manuscript was written by TL. The work was supervised by FN and DAC.

Representation Learning on Biomolecular Structures using Equivariant Graph Attention

Tuan Le
Bayer AG
Freie Universität Berlin
tuan.le2@bayer.com

Frank Noé
Microsoft Research AI4Science
Freie Universität Berlin
franknoe@microsoft.com

Djork-Arné Clevert*
Bayer AG
djork-arne.clevert@pfizer.com

Abstract

Learning and reasoning about 3D molecular structures with varying size is an emerging and important challenge in machine learning and especially in the development of biotherapeutics. Equivariant Graph Neural Networks (GNNs) can simultaneously leverage the geometric and relational detail of the problem domain and are known to learn expressive representations through the propagation of information between nodes leveraging higher-order representations to faithfully express the geometry of the data, such as directionality in their intermediate layers. In this work, we propose an equivariant GNN that operates with Cartesian coordinates to incorporate directionality and we implement a novel attention mechanism, acting as a content and spatial dependent filter when propagating information between nodes. Our proposed message function processes vector features in a geometrically meaningful way by mixing existing vectors and creating new ones based on cross products. We demonstrate the efficacy of our architecture on accurately predicting properties of large biomolecules and show its computational advantage over recent methods which rely on irreducible representations by means of the spherical harmonics expansion.

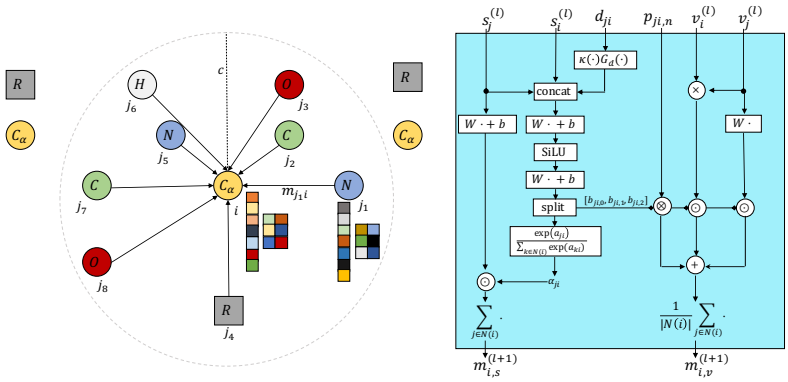
1 Introduction

Predicting molecular properties is of central importance to applications in pharmaceutical research and protein design with the incentive to establish accurate computational methods to accelerate the overall process of finding better molecular candidates in a faster and cost-efficient way. Learning on 3D environments of molecular structures is a rapidly growing area of machine learning with promising applications but also domain-specific challenges. While Deep Learning (DL) has replaced hand-crafted features to a large extent, many advances are crucially determined through inductive biases in deep neural networks. Developed neural models should maintain an efficient and accurate representation of structures with even up to thousand of atoms and correctly reason about their 3D geometry independent of orientation and position. A powerful method to restrict a neural network to the functions of interest, such as a molecular property, is to exploit the *symmetry* of the data by constraining *equivariance* with respect to transformations from a certain symmetry group [1, 2].

3D Graph Neural Networks (GNNs) have been applied on a broad field involving molecular structures, such as in the prediction of quantum chemistry properties of small molecules [3, 4] and also on macromolecular structures like proteins [5–8] due to the natural representation of structures as graphs, with atoms as nodes and edges drawn based on bonding or spatial proximity. These networks

*Work was done during time at Bayer AG.

T. Le et al., Representation Learning on Biomolecular Structures using Equivariant Graph Attention. *Proceedings of the First Learning on Graphs Conference (LoG 2022)*, PMLR 198, Virtual Event, December 9–12, 2022.



(a) Propagation flow for central node i. **(b)** Proposed equivariant message function $M_l(\cdot)$.

Figure 1: (a) Visualization of the local neighbourhood of central carbon atom i. Directed edges illustrate the message flow from neighbour j to central atom i, where scalar and vector features are propagated along the edges. Grey boxes R represent the side-chain atoms of each residue and serve here as visual compression that include many more atoms. Here, nodes comprise scalar and vector features with 7 and 2 channels, respectively. (b) Proposed equivariant message function that computes a geometric and content related feature attention filter for scalar features, while vector messages are created based on a weighted combination of newly constructed vectors.

generally encode the 3D geometry in terms of rotationally invariant representations, such as pairwise distances to model local interactions which leads to a loss of directional information, while including angular information into network architecture has shown to be beneficial in representing the local geometry [9–11].

Neural models that preserve equivariance on point clouds in 3D space have been proposed [12–15] which can be described as Tensorfield Networks. These group-theoretic inspired models leverage higher-order representations by means of the spherical harmonics expansion of normalized relative positions to initially create equivariant features. While these models enable the interaction between different-order representations, (often referred to as type-l representation), many data types are often restricted to scalar values (type-0 e.g., temperature or energy) and 3D vectors (type-1 e.g., velocity or forces). Another design choice is to define equivariant functions that directly operate on Cartesian coordinates [16–19], instead on the basis provided by the spherical harmonics. Following this approach, one could define (equivariant) transformations on Cartesian tensors, like rank 0 scalar(s) and rank 1 vector(s), which is the scope of this work and conceptually simpler and does not require Clebsch-Gordan tensor products of irreducible representations as commonly used in Tensorfield Network-like architectures.

In this work, we introduce Equivariant Graph Attention Networks (EQGAT) that operate on large point clouds such as proteins or protein-ligand complexes and show its superior performance compared to invariant models as well as our proposed model's faster training time compared to recent architectures that achieve equivariance through the usage of irreducible representations. Our model implements a novel feature attention mechanism which is invariant to global rotations and translations of inputs and includes spatial- but also content related information which serves as powerful edge embedding when propagating information in the Message Passing Neural Networks (MPNNs) [4] framework. Since we define equivariant functions on the original Cartesian space while restricting ourselves to tensor representations up to rank 1, i.e., scalars and vectors, we aim to capture as much geometrical information as possible through a geometrically motivated message function.

In summary, we make the following contributions:

- We introduce a computationally efficient equivariant Graph Neural Network that leverages geometric information by operating on vector features in Cartesian space.
- We implement a novel feature attention mechanism to propagate neighbouring node features and we define equivariant operations to combine vector features in a geometrically meaningful way.
- We benchmark our proposed architecture on large molecular systems such as protein complexes and show its efficacy mostly relevant to industrial applications.

2 Background

2.1 Message Passing Neural Networks (MPNNs)

MPNNs [4] generalize Graph Neural Networks (GNNs) [1, 2, 20] and aim to parameterize a mapping from a graph to a feature space. That feature space can either be defined on the node- or graph level. Formally, a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ contains nodes $i \in \mathcal{V}$ and edges $(j, i) \in \mathcal{E}$ which represent the relationship between nodes j and i. Since MPNNs utilize shared trainable layers among nodes, permutation equivariance is preserved.

In this work, we consider graphs representing molecular systems embedded in 3D Euclidean space, where atoms represent nodes and the edges are described through covalent bonds and/or by atom pairs within a certain cutoff distance c as illustrated in Figure 1(a). In the case of protein point clouds, a common design choice is the construction of residue graphs, where the nodes are represented through the C_α-atom of each amino acid residue [5, 6, 18].

We refer $x_i^{(l)} = (a_i, p_i, s_i^{(l)}, v_i^{(l)})$ to the state of the $i-$th atom, where $a_i \in \mathbb{Z}_+$ and $p_i \in \mathbb{R}^3$ denote atom i's chemical element and its spatial position, while $h_i^{(l)} = (s_i^{(l)}, v_i^{(l)}) \in \mathbb{R}^{1\times F_s} \times \mathbb{R}^{3\times F_v}$ are the hidden scalar and vector features that are iteratively refined through L message passing steps. We distinguish between scalar and vector features because scalars can be transformed without functional restrictions, e.g., with standard MLPs, and their domain spans the entire $\mathbb{R}$, while vector features that reside in $\mathbb{R}^3$ can only be transformed in certain ways to preserve rotation equivariance. In theory, one could also only rely on vector features (with a number of F_v channels), and perform a self-dot product reduction to make that representation invariant. This step however, restricts the domain space of scalars onto $\mathbb{R}_+$ only.

A general MPNN implements a learnable *message* and *update* function denoted as $M_l(\cdot)$ and $U_l(\cdot)$ to process atom $i-$th's hidden feature by considering its local environment $\mathcal{N}(i)$ through

$$m_i^{(l+1)} = \sum_{j\in\mathcal{N}(i)} M_l(x_i^{(l)}, x_j^{(l)}), \quad \text{and} \quad x_i^{(l+1)} = (a_i, p_i, U_l(x_i^{(l)}, m_i^{(l+1)})), \tag{1}$$

where $\mathcal{N}(i) = \{j : ||p_{ij}||_2 = ||p_j - p_i||_2 = d_{ij} < c\}$ denotes central atom's $i-$th neighbour set that is obtained through a distance cutoff $c > 0$.

For our 3D GNN, we wish to implement simple, yet powerful rotation equivariant transformations in the message and update functions, to accurately describe the local environment of nodes in the point cloud.

2.2 Invariance and Equivariance

In this work, we consider the special orthogonal group SO(3), i.e. the group of proper rotations in three dimensions. A group element of SO(3) is commonly represented as matrix $R \in \mathbb{R}^{3\times3}$ satisfying $R^\top R = RR^\top = I$ and $\det R = 1$.
For a node feature $h = (s, v) \in \mathbb{R}^{F_s} \times \mathbb{R}^{3\times F_v}$, an SO(3)-equivariant function $f(h) = h' = (s', v')$ must obey the following equation

$$f(g.h) = g.(s', v') = (Is', Rv') = (s', Rv') = g.f(h), \tag{2}$$

where $g.o$ in this work means, a group element g of SO(3) acting on the object o. As shown in (2), invariance can be regarded as special case of equivariance, where equivariance for a scalar representation means that the *trivial* representation, i.e. the identity, acts on the scalar embedding, while vectors are transformed with R, i.e., a change of basis is performed, where the new basis is determined by the columns in R.

3 Related Work

Neural networks that specifically achieve E(3) or SE(3) equivariance have been proposed in Tensorfield Networks (TFNs) [12] and its variants in the covariant Cormorant [13], NequIP [15] and SE(3)-Transformer [14] which includes the attention mechanism in their architecture. With TFNs, equivariance is achieved through the usage of equivariant function spaces such as spherical harmonics combined with Clebsch-Gordan tensor products in their intermediate layer to allow the multiplication of different ordered representations, while others resort to lifting the spatial space to higher-dimensional spaces such as Lie group spaces [21]. Since no restriction on the order of representations is imposed on these methods, sufficient expressive power of these models is guaranteed, but at a cost of enlarged computational calculations with increased time and memory. It was recently analyzed by Brandstetter et al. [22] that the implementation of non-linear equivariant Graph Neural Networks in their model, which they term Steerable E(3) Equivariant Graph Neural Networks (SEGNN) achieves strong empirical results on small point clouds like the N-Body experiment or QM9 dataset, but also larger systems as in the OC20 dataset. One of their insights is that the construction of their (non-linear) SEGNN-layer, allows the model to better capture the local environment and enables the reduction of radius cutoff when constructing the neighbour list for each central atom i, since the Clebsch-Gordan tensor products between neighbouring nodes is computationally expensive. To circumvent the expensive computational cost, another line of research proposed to implement equivariant operations in the original Cartesian space, providing and efficient approach to preserve equivariance as introduced in the E(n)-GNN [16], GVP [18, 23], PaiNN [17] and ET-Transformer [24] architectures without relying on irreducible representation of the orthogonal group by means of the spherical harmonics basis as originally introduced in TFN and implemented in the e3nn framework [25]. Aside of 3D atomistic GNNs, the attention mechanism has also been implemented in the GAT [26] and GATv2 [27] architectures, where GATv2 achieves superior performance over GAT due to the implementation of attention coefficients using a multilayer perceptron (MLP).

Our proposed model implements equivariant operations in the original Cartesian space and includes a continuous filter through the self-attention coefficients which serve as spatial- and content based edge embedding in the message propagation, as opposed to the PaiNN model where the filter solely depends on the distance. Additionally, our model constructs vector features from the given point cloud and leverages geometrical products that are efficient to compute. The E(n)-GNN architecture does not learn vector features with several channels, but only updates a single vector feature[2] through a weighted linear combination, where the (learnable) scalar weights are obtained from invariant embeddings. The GVP model which was initially designed to work on macromolecular structures includes a complex message functions of concatenated node- and edge features composed with a series of GVP-blocks that enables information exchange between scalar and vector features, through dot product reduction of vectors, with a potential disadvantage of discontinuities through non-smooth components for distances close to the cutoff.

4 Proposed Model Architecture

4.1 Input Embedding

We initially embed atoms of small molecules or proteins based on their element/amino acid type using a trainable look-up table through $s_i^{(0)} = \text{embed}(a_i)$, which provides a starting (invariant) scalar representation of the node prior to the message passing. As in most cases, no initial vector features for atoms are available, we initialize them as zero tensor $v_i^{(0)} = 0 \in \mathbb{R}^{3\times F_v}$.

4.2 Edge Filter through Feature Attention

For the two-body interaction between neighbouring node(s) j to central node i, we implement a non-linear edge filter that depends on content related information stored in the scalar features (s_j, s_i) and a radial basis expansion of the Euclidean distance $d_{ji} \leq c$. We choose the (orthonormal) Bessel basis $G_d : \mathbb{R} \to \mathbb{R}^K$ that projects the distance into K basis values as introduced by Gasteiger et al. [9] and their polynomial envelope function $\kappa : [0, c] \to (0, 1]$ that smoothly transitions from 1 to 0 as

[2] In the E(n)-GNN architecture, Cartesian coordinates of particles $p \in \mathbb{R}^3$ are updated.

the cutoff value c is approached. The computation of the attention edge-filter is obtained through

$$\begin{aligned} e_{ji}^{(l+1)} &= [s_i^{(l)} || s_j^{(l)} || \kappa(d_{ji}) G_d(d_{ji})] \in \mathbb{R}^{2F_s+K} \\ f_{ji}^{(l+1)} &= \text{MLP}(e_{ji}^{(l+1)}) \in \mathbb{R}^{F_s+3F_v}, \end{aligned} \tag{3}$$

where MLP refers to an 1-layer Multilayer-Perceptron with SiLU activation function [28]. The input to the MLP is a concatenation of scalar features as well as a by κ scaled radial basis expansion of the distance between nodes j and i. The SO(3)-invariant embedding $f_{ji}^{(l+1)}$ represents the $F_s + 3F_v$ attention logits which are further split into $f_{ji}^{(l+1)} = [a_{ji}, b_{ji}]^{(l+1)}$ to be used as a non-linear filter when propagating neighbouring features. A novelty of our approach is that the attention coefficient between two vertices j and i is in fact obtained per feature-channel instead for the entire embedding as commonly achieved through a single scalar value, as done in GATv2 [27], albeit we also include edge-features through distances. The feature attention for the scalar embeddings is computed using the standard softmax activation function

$$\alpha_{ji} = \frac{\exp(a_{ji})}{\sum_{k \in \mathcal{N}(i)} \exp(a_{ki})} \in (0,1)^{F_s}, \tag{4}$$

where the normalization in the denominator runs over all neighbours k and the exponential function is applied componentwise. We choose to compute a non-linear intermediate edge-filter f_{ji} due to increased expressivity through an 1-layer MLP. The embedding $b_{ji} \in \mathbb{R}^{3F_v}$ is processed to create coefficients that serve as weights for a linear combination of vector quantities to compute the vector message from j to i, which we will describe in the following subsection.

4.3 Equivariant Message Propagation

We follow the idea of standard convolution, which is a linear transformation of the input, and compute the scalar features message for central node i as

$$m_{i,s}^{(l+1)} = \sum_{j \in \mathcal{N}(i)} \alpha_{ji}^{(l+1)} \odot W_s^{(l+1)} s_j^{(l)}, \tag{5}$$

where $W_s^{(l+1)} \in \mathbb{R}^{F_s \times F_s}$ is a trainable weight matrix shared among all nodes and $\alpha_{ji}^{(l+1)}$ the non-linear attention filter obtained in (4).

In context of atomistic neural network potentials (NNPs), the filter $\alpha_{ji}^{(l+1)}$ is commonly implemented as an MLP that only inputs the distance d_{ji} (by means of a radial basis expansion) as in SchNet [3], PaiNN [17], NequIP [15], while recent NNPs such as Allegro [29] and BOTNet [30] implement edge-filters that depend on the distance as well as node content, e.g., the chemical elements, unifying the idea of MPNNs in the context of machine learning force fields.

The recent work by Brandstetter et al. [22] analyzes modern 3D equivariant GNNs with the insight that non-linear message and non-linear update functions combined with their proposed *steerable* features space leads to an improved model, which they term SEGNN. The SEGNN, in similar spirit to Tensorfield Networks, can leverage higher-order equivariant representations up to a maximal rotation order l_{max} through the spherical harmonics expansion of relative positions, which they take as steerable feature basis. Their proposed model implements *steerable* MLPs into the message- and update function to leverage non-linearity and geometric covariant information of the steerable features that go beyond $l = 0$, i.e., scalar features while our architecture is only restricted to scalar information, albeit vector information is still processed in the layers but then reduces to a scalar by a dot product operation. Our proposed message function for scalar features in Eq. (5) can also be formulated as a linear transformation where the weight matrix depends on distances but also hidden scalar information. To see this, we rewrite $\alpha_{ji}^{(l+1)} \in (0,1)^{F_s}$ as matrix using the diagonal operator $A_{ji}^{(l+1)} = \text{diag}(\alpha_{ji}^{(l+1)}) \in (0,1)^{F_s \times F_s}$ and observe that the filter scales the (independent) weight matrix $W_s^{(l+1)}$ leading to the message propagation

$$m_{i,s}^{(l+1)} = \sum_{j \in \mathcal{N}(i)} A_{ji}^{(l+1)} W_s^{(l+1)} s_j^{(l)} = \sum_{j \in \mathcal{N}(i)} W_{ji}^{(l+1)} s_j^{(l)},$$

where $W_{ji}^{(l+1)}$ defines the linear transformation matrix which depends on SO(3)-invariant information through $(s_i^{(l)}, s_j^{(l)}, d_{ji})$. The scalar message propagation can still be interpreted as non-linear convolution as the $A_{ji}^{(l+1)}$ weight matrix is obtained through an MLP and softmax activation function.

Building Equivariant Features. In many cases, no initial vector features are provided in raw point cloud data. However, when working with a protein backbone, i.e., the sequence of atoms $(C_\alpha, C, O, N)_i$, initial vectorial (node) features that describe the local environment of each backbone atom can be pre-computed as described by Ingraham et al. [6] and Jing et al. [18]. In a full-atom model, initial vector features for a node i can be obtained by averaging over relative position vectors $v_{i,0} = \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} p_{ji} \in \mathbb{R}^3$ which satisfies Eq. (2) due to linearity. In our work, we initialize the vectors as zero tensor as described in Subsection 4.1 and calculate equivariant features by utilizing normalized relative positions $p_{ji,n}$ in the first layer to describe the directional interaction between central node i and its neighbour j. In the subsequent layers, we extend the set of vectors by (1) constructing vectors based on normalized relative positions again, (2) mixing existing vector channels from the previous iteration, and (3) creating new vector quantities by making use of the cross product.

(1) We create equivariant vector features based on normalized relative position $p_{ji,n} = \frac{1}{d_{ji}}(p_i - p_j)$ as those provide directional information. Since we explicitly model scalar and vector features, each equipped with F_s and F_v channels, respectively, the tensor product offers a natural way to obtain a vector feature, by simply combining a vector and a scalar. Equivariant interactions between node j and i are computed through

$$v_{ji,0}^{(l+1)} = p_{ji,n} \otimes b_{ji,0}^{(l+1)} = p_{ji,n} b_{ji,0}^{(l+1)\top} \in \mathbb{R}^{3 \times F_v}, \tag{6}$$

which preserves SO(3) equivariance, due to the linearity of the tensor product. We note that the creation of 'initial' equivariant features in such manner is also performed in architectures, like [12, 13, 15, 22] just to name a few, that make use of irreducible representations of the SO(3) group by means of the spherical harmonics and implement the Clebsch-Gordan tensor product $(\otimes_{cg})$ that allows the mixing of possibly higher-order embedding representations of type $l > 1$, while we restrict ourselves to vector representations only, i.e. features of order $l = 1$ or equivalently Cartesian rank 1 tensors. The representation in Eq. (6) can be interpreted as F_v scaled relative position vectors.

(2) In similar fashion to the (independent) linear transformation of scalar channels, we mix the vector channels using a learnable weight matrix $W_v^{(l)} \in \mathbb{R}^{F_v \times F_v}$ which preserves SO(3) equivariance due to the linearity property

$$v_n^{(l+1)} = v^{(l)} W_v^{(l+1)},$$

and is shared among all nodes. For a particular neighbouring node j, we scale the linearly transformed vectors

$$v_{ji,1}^{(l+1)} = b_{ji,1}^{(l+1)} \odot v_{n,j}^{(l+1)}, \tag{7}$$

which can be interpreted as a gating of previously mixed vectors.

(3) To capture more geometric information, while restricting the representation to be of rank 1, we utilize the vector cross product $c = (a \times b) \in \mathbb{R}^3$ between two vectors a and b that satisfy the following rotation invariance property

$$Ra \times Rb = R(a \times b).$$

The output of the cross product $a \times b$ defines a vector c that is perpendicular to plane spanned by a and b. Here, we calculate the cross product on the same channels from the previous layer vector features of node i and j as

$$\tilde{v}_{ji,2}^{(l+1)} = (v_i^{(l)} \times v_j^{(l)}) \in \mathbb{R}^{3 \times F_v},$$

to reduce the computational complexity.

We highlight that recent equivariant GNNs which work with rank 1 Cartesian tensors, such as GVP, PaiNN or ET-Transformer do not include the cross product in their architecture and are restricted in the creation of vector features that may span the entire $\mathbb{R}^3$. These architecture make use of step (1) and (2) only. For example, when all atoms are placed on the xy-plane, using step (1) and (2) would

always create vectors on the xy plane, while the coordinate on z axis is always 0. By leveraging the cross product, vectors in the z direction can be computed, without increasing the rank order[3].

We note that our assumption on SO(3) equivariance is attributed to the fact of using the cross product in our architecture. For the case that practitioners care about O(3) equivariance, our proposed EQGAT might be suboptimal for usage since we do not distinguish polar or pseudo vectors in the internal network representation. If O(3) equivariance is desired, special care on the selection between input vectors in the cross product have to be made, in order to correctly assign the output parity type. E.g., a cross product of two polar vectors will return a pseudo vector, while a cross product of a polar and pseudo vector will return a polar vector.

In similar fashion to Eq. (6) and (7), each channel of the representation $\tilde{v}_{ji,2}^{(l)}$ is weighted by the SO(3) non-linear filter $b_{ji,2}^{(l)} \in \mathbb{R}^{F_v}$ to obtain

$$v_{ji,2}^{(l+1)} = b_{ji,2}^{(l+1)} \odot \tilde{v}_{ji,2}^{(l+1)}, \tag{8}$$

Finally, we define the vector message from node j to central node i as the sum of the three components in (6) to (8) and aggregate it across all neighbouring nodes $j \in \mathcal{N}(i)$ to obtain the vector message

$$m_{i,v}^{(l+1)} = \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} (v_{ji,0}^{(l+1)} + v_{ji,1}^{(l+1)} + v_{ji,2}^{(l+1)}), \tag{9}$$

which results into new weighted geometric vectors by utilizing the (static) relative positions as well as neighbouring vector features and lastly, normal vectors obtained through the cross product. Since we combine the three vector components through a gating mechanism, we do not use an attention mechanism on vector features to avoid additional computational steps and the fact that the calculation of attention logits had to be done using some SO(3) invariant input, which would make the model more complicated. We provide the full proof of SO(3) equivariance of Eq. (9) in Appendix C.

Equivariant Update Function. After obtaining the aggregated message for central node i in the representation $m^{(l+1)} \in \mathbb{R}^{F_s} \times \mathbb{R}^{3 \times F_v}$, we implement a residual connection as intermediate update step

$$\tilde{s}_i^{(l+1)} = s_i^{(l)} + m_{i,s}^{(l+1)}, \quad \text{and} \quad \tilde{v}_i^{(l+1)} = v_i^{(l)} + m_{i,v}^{(l+1)}$$

while in the update layer, we implement an equivariant non-linear transformation inspired by gated non-linearities proposed by [31] and used in [17] with minor modification as shown in Figure 2. Notably, the scalar features receive geometric information by concatenating the norm of linear transformed vector features, while the 1-layer scalar MLP is tasked to transform the combined embeddings to update the scalar states and retrieve non-linear weights that are used to reweight vector features. We apply these weights by element-wise multiplying with linearly transformed vector features as shown on the right which can also be interpreted as variants of the Gated Linear Unit [32, 33], followed by a linear layer to implement an equivariant MLP for vector features.

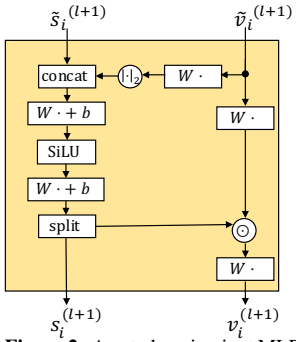


Figure 2: A gated equivariant MLP that transforms scalar and vector features into a new representation. Here we used this block as update function $U_l(\cdot)$.

5 Experiments and Results

We test the efficacy of our proposed EQGAT model on five publicly available molecular benchmark datasets which pose significant challenges for the development of efficient and accurate prediction models in protein design.

[3]Two rank 1 Cartesian tensors, i.e., two vectors can also be combined by computing the tensor product of the two, which results into a rank 2 Cartesian tensor with 9 elements in the matrix. This rank 2 Cartesian tensor contains 3 unique elements of the cross product in its antisymmetric part after a sum decomposition.

Table 1: Benchmark results on ATOM3D tasks. We report the results for the Baseline models from [34] and GVP-GNN [23]. We run our own experiments with the SchNet, PaiNN, SEGNN and our EQGAT model and report averaged metrics over 3 runs. For the SEGNN model we only report the results on a single run due longer training time. R_S stands for Spearman Rank Correlation, RMSE abbreviates Root Mean Square Deviation and ROCAUC the area under ROC curve.

Tasks	PSR (↑)		RSR (↑)		LBA (↓)	RES (↑)	PPI (↑)
Metric	Mean R_S	Global R_S	Mean R_S	Global R_S	RMSE	Accuracy	ROCAUC
CNN	0.431 ± 0.013	0.789 ± 0.017	0.264 ± 0.046	0.372 ± 0.027	$\mathbf{1.416 \pm 0.021}$	0.451 ± 0.002	0.844 ± 0.002
GNN	$\mathbf{0.515 \pm 0.010}$	0.755 ± 0.004	0.234 ± 0.006	$\mathbf{0.512 \pm 0.049}$	1.570 ± 0.025	0.082 ± 0.002	0.669 ± 0.001
GVP-GNN	0.511 ± 0.010	0.845 ± 0.008	0.211 ± 0.142	0.330 ± 0.054	1.594 ± 0.073	0.527 ± 0.003	0.866 ± 0.004
SchNet	0.448 ± 0.016	0.784 ± 0.013	0.247 ± 0.029	0.273 ± 0.017	1.522 ± 0.015	0.326 ± 0.003	0.839 ± 0.005
PaiNN	0.462 ± 0.015	0.809 ± 0.003	0.270 ± 0.062	0.462 ± 0.064	1.507 ± 0.033	0.370 ± 0.004	0.884 ± 0.002
SEGNN	0.474	0.833	-0.099	0.252	1.450 ± 0.011	0.454	0.854
EQGAT	0.491 ± 0.008	$\mathbf{0.847 \pm 0.006}$	$\mathbf{0.316 \pm 0.029}$	0.404 ± 0.096	1.440 ± 0.027	$\mathbf{0.540 \pm 0.017}$	$\mathbf{0.908 \pm 0.001}$

5.1 ATOM3D

The ATOM3D benchmark [34] provides datasets for representation learning on atomic-level 3D molecular structures of different kinds, i.e., proteins, RNAs, small molecules and complexes. Since proteins perform specific biological functions essential for all living organisms and hence, play a key role when investigating the most fundamental questions in the life sciences, we focus our experiments on the learning problems often encountered in structural biology with different difficulties due to data scarcity and varying structural sizes. We use provided training, validation and test splits from ATOM3D and refer the interested reader to the original work of Townshend et al. [34] for more details. For all benchmarks, we compare against the Baseline CNN and GNN models provided by Townshend et al. [34] from ATOM3D, GVP-GNN reported in [23] and we run experiments for SchNet [3], an $\mathrm{SO}(3)$ invariant GNN architecture that has shown strong performance on small molecule prediction tasks, PaiNN [17] as SchNet's improved SO(3) equivariant architecture and the recently proposed SEGNN [22] that leverages higher-order representations by means of the irreducible representations and Clebsch-Gordan tensor products using their official code base.

For SchNet, PaiNN and our proposed EQGAT architecture, we implement a 5-layer GNN with $F_s = 100$ scalar channels and $F_v = 16$ vector channels for the PSR, RSR, RES and PPI benchmark, as these benchmarks consists of more training samples and comprise larger biomolecules. For the Ligand Binding Affinity (LBA) task, we utilize a 3-layer GNN with the same number of scalar- and vector channels. For the SEGNN architecture, we implement a 3-layer GNN with $(100, 16, 8)$ channels for the embeddings of type $l = (0, 1, 2)$ that transform according to the irreducible representation of that order preserving SO(3) equivariance. The edges in the point clouds are constructed based on a radius cutoff of 4.5Å. All graphs are considered as full-atom graphs, i.e., the initial node feature is determined by the chemical element.

The Protein and RNA Structure Ranking tasks (PSR / RSR) in ATOM3D are both regression tasks with the objective to predict the quality score in terms of *Global Distance Test* (GDT_TS) or Root-Mean-Square Deviation (RMSD) for generated Protein and RNA models wrt. to its experimentally determined ground-truth structure. The ability to reliably rank a biopolymer structure requires a model to accurately learn the atomic environments such that discrepancies between a ground truth states an its corrupted version can be distinguished. We evaluated our model on the biopolymer ranking and obtained good results on the current benchmark, as reported in Table 1 in terms of Spearman rank correlation. Our proposed model performs particularly well on the PSR task outperforming the GVP-GNN [23] on the Global Rank Spearman correlation on the test set, while our model is more parameter efficient (383K vs. 640K). We believe our model could be further improved by additional hyperparameter tuning, e.g., by increasing the number of scalar or vector channels, which we did not do in our study to compare against the baseline models.

We noticed that the RSR benchmark was particularly difficult to validate as only a few dozen experimentally determined RNA structures are existent to date, and the structural models generated in the ATOM3D framework are labeled with the RMSD to its native structure, which is known to be sensitive to outlier regions, for exampling by inadequate modelling of loop regions [35], while the GDT_TS metric might be a better suited target to predict a ranking for generated RNA structures as in the PSR benchmark.

Another challenging and important task for drug discovery projects is estimating the binding strength (affinity) of a candidate drug atomistic's interaction with a target protein. We use the ligand binding affinity (LBA) dataset and found that among the GNN architectures, our proposed model obtains the best results, while also being computationally cheap and fast to train. The best performing model in the LBA-task is a 3D CNN model which works on the joint protein-ligand representation using voxel space and enforcing equivariance through data augmentation. The inferior performance of all equivariant GNNs might be caused by the need of larger filters to better capture the locality and many-body effects, where 3D CNNs have an advantage when using voxel representations, while GNNs commonly capture 2-body effects. Furthermore, as all GNN models jointly represent ligand- and protein as *one* graph by connecting vertices through a distance cutoff of 4.5Å, we believe that such union leads to an information loss of distinguishing the atom identity from the ligand and protein. A promising direction to investigate is to incorporate a ligand and protein GNN encoder seperately and merge the two embeddings prior the binding affinity prediction, similar to Graph Matching Networks [36] and recently realized by Stärk et al. [37] in a slightly different context.

EQGAT outperforms the current SOTA GVP-GNN model on the *Residue* and *Protein-Protein-Interaction* benchmarks which are both node classification tasks and require a model to accurately capture the local environment of a selected C_α atom to serve as expressive input for a downstream (decoder) network to obtain the final prediction.

Notably, our proposed EQGAT architecture performs on par with the SEGNN that implements internal representations of higher order, i.e., of rotation order up to $l = 2$. We believe that including the cross product in our vector message in (9) allows the model to capture more geometric detail in a possible protein ligand binding pose for accurately predicting the binding affinity, which is investigated in the following ablations.

5.2 Ablation Studies

To evaluate the benefits of our designed EQGAT architecture, we perform ablation studies and remove architectural components to isolate the effect of each design choice on performance.

Table 2: Results of the ablation studies.

	LBA [RMSE ↓]	PSR [Mean | Global R_S ↑]
No-Cross-Product	1.458 (0.011)	0.477 (0.012) | 0.827 (0.010)
No-Feature-Attention	1.466 (0.040)	**0.492 (0.007)** | 0.820 (0.002)
Full Model	**1.440 (0.027)**	0.491 (0.008) | **0.847 (0.006)**

Ablation study 1 (termed No-Cross-Product) removes the contribution of vector cross product (denoted as $v_{ji,2}$ in Eq. (9)). This leads to the effect that the vector message is solely constructed based on scaled versions of normalized relative positions ($v_{ji,0}$) and linear combinations of existing vector features ($v_{ji,1}$).

Ablation study 2 (termed No-Feature-Attention) replaces the feature attention coefficient $\alpha_{ji} \in (0,1)^{F_s}$ through a single coefficient $\alpha_{ji} \in (0,1)$.

We observe that the full EQGAT architecture obtains the best performance among the two datasets compared to the ablated models although we note that the improved performance of the full model in RMSE on the LBA benchmark and Global R_S in the PSR benchmark is difficult to attribute to the inclusion of architectural components due to the (larger) variance obtained through the 3 runs for each experiment.

6 Conclusion, Limitations and Future Work

In this work, we introduced a novel attention-based equivariant graph neural network for the prediction of properties of large biomolecules that achieves superior performance on the ATOM3D benchmark. Our proposed architecture makes use of rotationally equivariant features in their intermediate layers to faithfully represent the geometry of the data, while being computationally efficient, as all equivariant functions are directly implemented in the original Cartesian space without changing the representation through the spherical harmonics basis as commonly done in Tensorfield networks. As our proposed model operates on Cartesian tensors and we restrict the representation to be of rank 1 only, a general promising future direction of investigation is the implementation of Cartesian equivariant GNNs that

leverage higher-rank tensors in their layers, that are specifically implemented for learning purposes involving large biomolecules. As it is up to date not clear, how much improvement higher-order Cartesian tensors benefit for learning tasks that involve large biomolecular systems, we hope that our work and open-source code will be useful for the graph learning and computational biology community.

Code Availability

We provide the implementation of our model and experiments on `https://github.com/Bayer-Group/eqgat`. We use PyTorch [38] as Deep Learning framework and PyTorch Geometric [39] to implement our GNNs.

Author Contributions

T.L designed the model architecture, carried out the implementation and executed all experiments. T.L wrote the manuscript with input from F.N and D.A.C who both proofread the final manuscript. D.A.C supervised the work.

Acknowledgements

The authors are grateful to the anonymous reviewers of the LoG conference 2022 for proof-reading the paper and suggesting improvements. This work was supported by several research grants. F.N acknowledges funding from the European Commission (ERC CoG 772230 'ScaleCell'), MATH+ (AA1-6), Math AA1-10 and Deutsche Forschungsgemeinschaft (CRC1114/B08). D.A.C acknowledged funding from the Bayer AG Life Science Collaboration ('DeepMinDS'). D.A.C also received financial support from European Commission grant numbers 963845 and 956832 under the Horizon2020 Framework Program for Research and Innovation. T.L acknowledges funding from Bayer AG's PhD scholarship.

References

[1] Peter W. Battaglia, Jessica B. Hamrick, Victor Bapst, Alvaro Sanchez-Gonzalez, Vinícius Flores Zambaldi, Mateusz Malinowski, Andrea Tacchetti, David Raposo, Adam Santoro, Ryan Faulkner, Çaglar Gülçehre, H. Francis Song, Andrew J. Ballard, Justin Gilmer, George E. Dahl, Ashish Vaswani, Kelsey R. Allen, Charles Nash, Victoria Langston, Chris Dyer, Nicolas Heess, Daan Wierstra, Pushmeet Kohli, Matthew M. Botvinick, Oriol Vinyals, Yujia Li, and Razvan Pascanu. Relational inductive biases, deep learning, and graph networks. *CoRR*, abs/1806.01261, 2018. URL `http://arxiv.org/abs/1806.01261`. 1, 3

[2] Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges, 2021. 1, 3

[3] Kristof Schütt, Pieter-Jan Kindermans, Huziel Enoc Sauceda Felix, Stefan Chmiela, Alexandre Tkatchenko, and Klaus-Robert Müller. Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL `https://proceedings.neurips.cc/paper/2017/file/303ed4c69846ab36c2904d3ba8573050-Paper.pdf`. 1, 5, 8

[4] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural message passing for quantum chemistry. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1263–1272. PMLR, 06–11 Aug 2017. URL `https://proceedings.mlr.press/v70/gilmer17a.html`. 1, 2, 3

[5] Alex Fout, Jonathon Byrd, Basir Shariat, and Asa Ben-Hur. Protein interface prediction using graph convolutional networks. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL `https://proceedings.neurips.cc/paper/2017/file/f507783927f2ec2737ba40afbd17efb5-Paper.pdf`. 1, 3

[6] John Ingraham, Vikas Garg, Regina Barzilay, and Tommi Jaakkola. Generative models for graph-based protein design. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/f3a4ff4839c56a5f460c88cce3666a2b-Paper.pdf. 3, 6

[7] Federico Baldassarre, David Menéndez Hurtado, Arne Elofsson, and Hossein Azizpour. GraphQA: protein model quality assessment using graph convolutional networks. *Bioinformatics*, 37(3):360–366, 08 2020. ISSN 1367-4803. doi: 10.1093/bioinformatics/btaa714. URL https://doi.org/10.1093/bioinformatics/btaa714.

[8] Pedro Hermosilla, Marco Schäfer, Matej Lang, Gloria Fackelmann, Pere-Pau Vázquez, Barbora Kozlikova, Michael Krone, Tobias Ritschel, and Timo Ropinski. Intrinsic-extrinsic convolution and pooling for learning on 3d protein structures. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=1OmSUROpwY. 1

[9] Johannes Gasteiger, Janek Groß, and Stephan Günnemann. Directional message passing for molecular graphs. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=B1eWbxStPH. 2, 4

[10] Johannes Gasteiger, Florian Becker, and Stephan Günnemann. Gemnet: Universal directional graph neural networks for molecules. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 6790–6802. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper/2021/file/35cf8659cfcb13224cbd47863a34fc58-Paper.pdf.

[11] Yi Liu, Limei Wang, Meng Liu, Yuchao Lin, Xuan Zhang, Bora Oztekin, and Shuiwang Ji. Spherical message passing for 3d molecular graphs. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=givsRXsOt9r. 2

[12] Nathaniel Thomas, Tess Smidt, Steven Kearnes, Lusann Yang, Li Li, Kai Kohlhoff, and Patrick Riley. Tensor field networks: Rotation- and translation-equivariant neural networks for 3d point clouds, 2018. 2, 4, 6

[13] Brandon Anderson, Truong Son Hy, and Risi Kondor. Cormorant: Covariant molecular neural networks. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/03573b32b2746e6e8ca98b9123f2249b-Paper.pdf. 4, 6

[14] Fabian Fuchs, Daniel Worrall, Volker Fischer, and Max Welling. Se(3)-transformers: 3d roto-translation equivariant attention networks. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1970–1981. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/15231a7ce4ba789d13b722cc5c955834-Paper.pdf. 4

[15] Simon Batzner, Albert Musaelian, Lixin Sun, Mario Geiger, Jonathan P. Mailoa, Mordechai Kornbluth, Nicola Molinari, Tess E. Smidt, and Boris Kozinsky. E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *Nature Communications*, 13 (1):2453, 2022. doi: 10.1038/s41467-022-29939-5. URL https://doi.org/10.1038/s41467-022-29939-5. 2, 4, 5, 6

[16] Víctor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E(n) equivariant graph neural networks. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 9323–9332. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/satorras21a.html. 2, 4

[17] Kristof Schütt, Oliver Unke, and Michael Gastegger. Equivariant message passing for the prediction of tensorial properties and molecular spectra. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 9377–9388. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/schutt21a.html. 4, 5, 7, 8

[18] Bowen Jing, Stephan Eismann, Patricia Suriana, Raphael John Lamarre Townshend, and Ron Dror. Learning from protein structure with geometric vector perceptrons. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=1YLJDvSx6J4. 3, 4, 6, 16

[19] C. Deng, O. Litany, Y. Duan, A. Poulenard, A. Tagliasacchi, and L. Guibas. Vector neurons: A general framework for so(3)-equivariant networks. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 12180–12189, Los Alamitos, CA, USA, oct 2021. IEEE Computer Society. doi: 10.1109/ICCV48922.2021.01198. URL https://doi.ieeecomputersociety.org/10.1109/ICCV48922.2021.01198. 2

[20] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80, 2009. doi: 10.1109/TNN.2008.2005605. 3

[21] Marc Finzi, Samuel Stanton, Pavel Izmailov, and Andrew Gordon Wilson. Generalizing convolutional neural networks for equivariance to lie groups on arbitrary continuous data. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3165–3176. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/finzi20a.html. 4

[22] Johannes Brandstetter, Rob Hesselink, Elise van der Pol, Erik J Bekkers, and Max Welling. Geometric and physical quantities improve e(3) equivariant message passing. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=_xwr8gOBeV1. 4, 5, 6, 8

[23] Bowen Jing, Stephan Eismann, Pratham N. Soni, and Ron O. Dror. Equivariant graph neural networks for 3d macromolecular structure, 2021. 4, 8

[24] Philipp Thölke and Gianni De Fabritiis. Equivariant transformers for neural network based molecular potentials. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=zNHzqZ9wrRB. 4

[25] Mario Geiger, Tess Smidt, Alby M., Benjamin Kurt Miller, Wouter Boomsma, Bradley Dice, Kostiantyn Lapchevskyi, Maurice Weiler, Michał Tyszkiewicz, Simon Batzner, Dylan Madisetti, Martin Uhrin, Jes Frellsen, Nuri Jung, Sophia Sanborn, Mingjian Wen, Josh Rackers, Marcel Rød, and Michael Bailey. Euclidean neural networks: e3nn, April 2022. URL https://doi.org/10.5281/zenodo.6459381. 4

[26] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018. URL https://openreview.net/forum?id=rJXMpikCZ. 4

[27] Shaked Brody, Uri Alon, and Eran Yahav. How attentive are graph attention networks? In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=F72ximsx7C1. 4, 5

[28] Prajit Ramachandran, Barret Zoph, and Quoc V. Le. Searching for activation functions, 2017. URL https://arxiv.org/abs/1710.05941. 5

[29] Albert Musaelian, Simon Batzner, Anders Johansson, Lixin Sun, Cameron J. Owen, Mordechai Kornbluth, and Boris Kozinsky. Learning local equivariant representations for large-scale atomistic dynamics, 2022. URL https://arxiv.org/abs/2204.05249. 5

[30] Ilyes Batatia, Simon Batzner, Dávid Péter Kovács, Albert Musaelian, Gregor N. C. Simm, Ralf Drautz, Christoph Ortner, Boris Kozinsky, and Gábor Csányi. The design space of e(3)-equivariant atom-centered interatomic potentials, 2022. URL https://arxiv.org/abs/2205.06643. 5

[31] Maurice Weiler, Mario Geiger, Max Welling, Wouter Boomsma, and Taco S Cohen. 3d steerable cnns: Learning rotationally equivariant features in volumetric data. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/488e4104520c6aab692863cc1dba45af-Paper.pdf. 7

[32] Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 933–941. PMLR, 06–11 Aug 2017. URL https://proceedings.mlr.press/v70/dauphin17a.html. 7

[33] Noam Shazeer. Glu variants improve transformer, 2020. URL https://arxiv.org/abs/2002.05202. 7

[34] Raphael John Lamarre Townshend, Martin Vögele, Patricia Adriana Suriana, Alexander Derry, Alexander Powers, Yianni Laloudakis, Sidhika Balachandar, Bowen Jing, Brandon M. Anderson, Stephan Eismann, Risi Kondor, Russ Altman, and Ron O. Dror. ATOM3d: Tasks on molecules in three dimensions. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021. URL https://openreview.net/forum?id=FkDZLpK1Ml2. 8

[35] Adam Zemla. Lga – a method for finding 3d similarities in protein structures. *Nucleic acids research*, 31:3370–4, 08 2003. doi: 10.1093/nar/gkg571. 8

[36] Yujia Li, Chenjie Gu, Thomas Dullien, Oriol Vinyals, and Pushmeet Kohli. Graph matching networks for learning the similarity of graph structured objects. In Kamalika Chaudhuri and Ruslan Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 3835–3845. PMLR, 09–15 Jun 2019. URL https://proceedings.mlr.press/v97/li19d.html. 9

[37] Hannes Stärk, Octavian Ganea, Lagnajit Pattanaik, Dr.Regina Barzilay, and Tommi Jaakkola. EquiBind: Geometric deep learning for drug binding structure prediction. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 20503–20521. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/stark22b.html. 9

[38] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf. 10

[39] Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019. 10

[40] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015. URL http://arxiv.org/abs/1412.6980. 14

A Appendix

Full Model Details and Hyperparameters

All EQGAT models in this paper were trained on a single Nvidia Tesla V100 GPU.

Table 3: Description of architectural parameters on the ATOM3D benchmarks.

Parameter	LBA	PSR	RSR
Learning rate (lr.)	10^{-4}	10^{-4}	10^{-4}
Maximum epochs	20	30	30
Lr. patience	10	10	10
Lr. decay factor	0.75	0.75	0.75
Batch size	16	16	16
Num. layers	3	5	5
Num. RBFs	32	32	32
Cutoff [Å]	4.5	4.5	4.5
Scalar channels F_s	100	100	100
Vector channels F_v	16	16	16
Num. parameters	238k	383k	383k

We used the ADAM optimizer [40] apart from the defined learning rate all other standard hyperparameter setting from the PyTorch library.

B Model Efficiency

Model Efficiency. We assess the model efficiency of EQGAT in terms of computation time as well as trainable parameters and compare against SchNet, PaiNN and SEGNN on the LBA, PSR and RSR benchmarks. These datasets have on average 408, 1624, and 2390 nodes per graph with 9180, 26756 and 44233 directed edges, respectively for the training set of LBA, PSR and RSR.

Table 4: Comparison on model efficiency when passing a batch of 10 macromolecular structures.

Dataset	Model (# Param.)	Inference Time [ms]
LBA	EQGAT (238K)	11.94
	SchNet (240K)	8.25
	PaiNN (379K)	10.66
	SEGNN (238K)	89.53
PSR	EQGAT (383K)	49.96
	SchNet (240K)	18.36
	PaiNN (379K)	18.58
	SEGNN (238K)	255.44
RSR	EQGAT (383K)	75.45
	SchNet (240K)	27.27
	PaiNN (379K)	26.98
	SEGNN (238K)	390.69

As these datasets consist of graphs with up to thousands of atoms, computationally- and memory efficient models are preferred such that batches of graphs can be stored on GPU memory and processed fast during training. We measure the inference time of a random batch comprising 10 macromolecular structures on an NVIDIA V100 GPU. As shown in Table 4, SchNet and PaiNN are both parameter efficient and both achieve the fastest inference time on a forward pass, while our proposed EQGAT is slower mainly due to the softmax attention normalization in the denominator in Eq. (4) which could be improved when the softmax attention with its normalization is replaced by a sigmoid activation function, to obtain soft-attention weights. This step however, results into a edge-filter α_{ji} that does not sum up to 1 when iterating over all neighbours j. The SEGNN model has the longest runtime on the forward pass across the 3 datasets. This is mostly attributed to the Clebsch-Cordan tensor products which can be very expensive in learning tasks that involve proteins, as the CG product is always performed on edges.

C Proof Equivariance

We prove the rotation equivariance in Eq. (9) which consists of the sum of three vector components, and displayed here again

$$m_{i,v}^{(l+1)} = \frac{1}{|\mathcal{N}(i)|} \sum_{j \in \mathcal{N}(i)} (v_{ji,0}^{(l+1)} + v_{ji,1}^{(l+1)} + v_{ji,2}^{(l+1)}).$$

As the sum is a linear function, we require to show that each summand $(v_{ji,0}, v_{ji,1}, v_{ji,2})$ is equivariant. For brevity, we omit all top indices. The first term is computed as tensor product of an $l = 1$ representation and $l = 0$ representation through

$$v_{ji,0} = p_{ji,n} \otimes b_{ji,0} = p_{ji,n} b_{ji,0}^{\top} \in \mathbb{R}^{3 \times F_v},$$

where $b_{ji,0} \in \mathbb{R}^{F_v}$ is an SO(3)-invariant representation, i.e. a scalar representation with F_v channels, and $p_{ji,n} \in S_2 \subset \mathbb{R}^3$ a normalized relative vector, which lies on the 2-dimensional sphere.
If the point cloud is rotated, as defined in Eq. (2), (relative) position as well as vector features change to

$$p \xrightarrow{R} Rp \,,$$
$$v \xrightarrow{R} Rv \,,$$

while the cross product between two vector features v_0, v_1 is invariant to rotation, resulting to the property

$$(Rv_0 \times Rv_1) = R(v_0 \times v_1) \,.$$

In case a rotation is acting on the system, from Eq. (2) we know how vector and scalar quantities transform, resulting into:

$$R.v_{ji,0} \to Rp_{ji,n} \otimes b_{ji,0} = R(p_{ji,n} \otimes b_{ji,0}) = Rv_{ji,0}.$$

due to the linearity of the tensor product which proves SO(3) equivariance for the first term.
For the second term, we calculate

$$v_{ji,1} = b_{ji,1} \odot (v_i \times v_j),$$

where $b_{ji,1} \in \mathbb{R}^{F_v}$ is an SO(3)-invariant representation and the output of the cross product is a vector representation $\in \mathbb{R}^{3 \times F_v}$. To be precise, the elementwise multiplication from the left with the $b_{ji,1}$ has to be rewritten, to match the shape, i.e. unsqueeze a new dimension to scale each of the F_v vector by the scalar value, resulting into:

$$v_{ji,1} = (1 \otimes b_{ji,1}) \odot (v_i \times v_j),$$

where 1 is the one-vector in 3 dimensions. For a rotation acting on the system, we conclude that

$$\begin{aligned} R.v_{ji,1} &\to (1 \otimes b_{ji,1}) \odot (Rv_i \times Rv_j) \\ &= (1 \otimes b_{ji,1}) \odot R(v_i \times v_j) = R(1 \otimes b_{ji,1}) \odot (v_i \times v_j) \\ &= Rv_{ji,1}, \end{aligned}$$

which proves SO(3) equivariance for the second term.
The third term is obtained through

$$v_{ji,2} = (1 \otimes b_{ji,2}) \odot (v_j W_n),$$

where $b_{ji,2} \in \mathbb{R}^{F_v}$ is a scalar representation with F_v channels and W_n a linear transformation of shape $(F_v \times F_v)$. Due to linearity, we can see that

$$Rv_j W_n = (Rv_j) W_n = R(v_j W_n)$$

is SO(3) equivariant. As we elementwise multiply with a unsqueezed/expanded scalar representation, we conclude for the last term SO(3) equivariance

$$\begin{aligned} R.v_{ji,2} &\to (1 \otimes b_{ji,1}) \odot (Rv_j) W_n \\ &= (1 \otimes b_{ji,1}) \odot R(v_j W_n) = R(1 \otimes b_{ji,1}) \odot (v_j W_n) \\ &= Rv_{ji,2}. \end{aligned}$$

Since all three components in the sum are SO(3) equivariant, we conclude that the final sum is also SO(3) equivariant.

As the reader might have noticed, we build equivariant features based on linear functions and weighting $l = 1$ representations through $l = 0$ representations. This typical scaling is achieved through the tensor product $\otimes$. Our architecure however, also performs a multiplication between two $l = 1$ representations, through the cross product, which has the pleasant SO(3) invariance property that we can exploit to prove SO(3) equivariance, when scaling the output with an $l = 0$ representation.

A Note on Translation Equivariance. Our proposed model is translation invariant, as all vector features are initially created by means of a tensor product of (normalized) relative position $p_{ji,n}$. To see that, for any translation vector $t \in \mathbb{R}^3$ for relative positions, we can see that the calculation of such vectors[4] $p_{ji} = p_j - p_i$, are inherently translation invariant due to

$$t.p_{ji} \rightarrow (p_j + t) - (p_i + t) = p_j - p_i + t - t = p_j - p_i = p_{ji}.$$

Since we do not model absolute Cartesian coordinates, e.g., by updating the spatial coordinates through our layers, our model is not SE(3)-equivariant, i.e. next to rotation equivariance, also translation equivariant. We note that translation equivariance, however can be achieved through a simple operation such as the addition of an SE(3) representation with an SO(3) representation, e.g.

$$p_i = p_i + p_{ji,n} \otimes s,$$

where $s \in \mathbb{R}$ and reminiscent in the E(n)-GNN architecture, albeit the authors are not using the notation of the tensor product.

D Synthetic Dataset

We adopt the synthetic dataset from GVP [18] with slight modifications to make it a more challenging task. We create 50,000 'structures' where each 'structure' consists of $n = 100$ random points in $\mathbb{R}^3$, distributed uniformly in the ball of radius $r = 10$ with the constraint that no two points are less than distance $d = 2$ apart. Three points are randomly chosen and are labelled as 'special' which will define the vertices of a triangle. The learning task is a multitask regression of 3 targets, where the first target is to predict the distance between the center of mass (COM) of the entire structure and the COM of the triangles spanned by the three special points. The second and third task is the prediction of the perimeter and surface area of the triangle. The choice of the 3 targets refers to a structural learning task, where the model requires to learn about the global shape of the structure, while the second and third targets are relational. An example structure is depicted in Figure 3. The evaluation metric is the MSE of the three tasks. We split the dataset into 80% training, 10% validation and 10% test sets.

Table 5: Evaluation of our proposed EQGAT architecture on Triangle benchmark.

Model	Triangle [MSE ↓]	No. Params [10^3]
SchNet	37.545 (1.838)	16.8
PaiNN	10.259 (0.949)	27.1
SEGNN	**3.875 (0.879)**	60.9
GVP	10.115 (1.210)	61.6
EQGAT-Full	6.003 (0.432)	27.4
EQGAT-No-Cross-Product	6.835 (1.066)	27.4
EQGAT-No-Feature-Attention	6.808 (0.326)	27.4

For the synthetic task of multitask regression we notice that the SEGNN architecture equipped with higher-order equivariant features up to rotation order 2, obtains the best performance, followed by our proposed EQGAT model that only incorporates rank 1 (vector) features. For the synthetic dataset, we did not perform any hyperparameter tuning and set the number of layers to 3 with $F_s = 32$ scalar

[4]We omit the normalization to unit vectors for brevity.

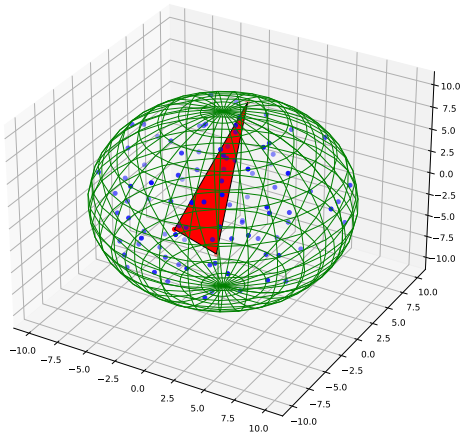


Figure 3: An example structure of the synthetic dataset. Three random points in the structure determine the vertices of a triangle, which is colored in red.

and $F_v = 8$ vector channels and train for 50 epochs. The number of trainable parameters for SchNet, PaiNN, SEGNN and EQGAT on the synthetic Triangle dataset are listed in the last column of Table 5.

2.4 Publication 4: Navigating the Design Space of Equivariant Diffusion-Based Generative Models for De Novo 3D Molecule Generation

Full Reference: ***Le, Tuan**; **Cremer, Julian**; Noé, Frank; Clevert, Djork-Arné & Schütt, Kristof (2024). Navigating the Design Space of Equivariant Diffusion-Based Generative Models for De Novo 3D Molecule Generation. 12th International Conference on Learning Representations. 2024*

Journal/Conference: International Conference on Learning Representations

Link to Article: `https://openreview.net/forum?id=kzGuiRXZrQ`

Source Code: `https://github.com/tuanle618/eqgat-diff`

Paper's main contributions:

- We propose EQGAT-diff – a fast and accurate 3D molecular diffusion model that employs E(3)-equivariant graph attention. Our proposed model achieves SOTA results in shorter training time and with less trainable parameters than previous architectures.
- We systematically explore various design choices for 3D molecular diffusion models and provide a thorough ablation study across the popular benchmark sets QM9 and GEOM-Drugs. We propose a time-dependent loss weighting as a crucial component for fast training convergence, better inference speed, and sample quality.
- We demonstrate the transferability of an EQGAT-diff model pre-trained on the PubChem3D dataset to smaller but complex molecular datasets. After a short fine-tuning on the target distribution, we show that the model outperforms models trained from scratch on the target data by only training on subsets.
- We extend the diffusion process by modeling chemically motivated additional features and show a further significant increase in performance.

Author's contribution to the paper:

- Conceptualization of the original idea and its application to molecular generation.
- Derivation and implementation of the method.
- Proposal and analysis of discrete diffusion.

- Design and evaluation of benchmarking experiments, data curation and analysis.
- Preparation and creation of initial draft and visualizations.
- Revision of the manuscript for the final conference submission.

TL and JC have shared first authorship. JC also contributed in the conceptualization, method development, proposed the pre-training strategy, and performed experiments. The manuscript was written by TL, JC and KS. The work was supervised by FN, DAC and KS.

NAVIGATING THE DESIGN SPACE OF EQUIVARIANT DIFFUSION-BASED GENERATIVE MODELS FOR DE NOVO 3D MOLECULE GENERATION

Tuan Le[*1,2], **Julian Cremer**[*1,4],
Frank Noé[2,3], **Djork-Arné Clevert**[1], **Kristof Schütt**[1]

[1]Pfizer Research & Development, [2]Freie Universität Berlin
[3]Microsoft Research, [4]University Pompeu Fabra

ABSTRACT

Deep generative diffusion models are a promising avenue for 3D de novo molecular design in materials science and drug discovery. However, their utility is still limited by suboptimal performance on large molecular structures and limited training data. To address this gap, we explore the design space of E(3)-equivariant diffusion models, focusing on previously unexplored areas. Our extensive comparative analysis evaluates the interplay between continuous and discrete state spaces. From this investigation, we present the EQGAT-diff model, which consistently outperforms established models for the QM9 and GEOM-Drugs datasets. Significantly, EQGAT-diff takes continuous atom positions, while chemical elements and bond types are categorical and uses time-dependent loss weighting, substantially increasing training convergence, the quality of generated samples, and inference time. We also showcase that including chemically motivated additional features like hybridization states in the diffusion process enhances the validity of generated molecules. To further strengthen the applicability of diffusion models to limited training data, we investigate the transferability of EQGAT-diff trained on the large PubChem3D dataset with implicit hydrogen atoms to target different data distributions. Fine-tuning EQGAT-diff for just a few iterations shows an efficient distribution shift, further improving performance throughout data sets. Finally, we test our model on the Crossdocked data set for structure-based de novo ligand generation, underlining the importance of our findings showing state-of-the-art performance on Vina docking scores.

1 INTRODUCTION

The enormous success of machine learning (ML) in computer vision and natural language processing in recent years has led to the adaptation of ML in many research areas in the natural sciences, such as physics, chemistry, and biology, with promising results. Specifically, modern drug discovery widely utilizes ML to efficiently screen the vast chemical space for *de novo* molecule design in the early-stage drug discovery pipeline. An important aspect is the structure-based or target-aware design of novel molecules in 3D space (Schneuing et al., 2023; Guan et al., 2023; Stärk et al., 2022; Corso et al., 2023). However, incorporating the 3D geometries of molecules for rational and structure-based drug design is challenging, and the development of ML models in this domain is anything but easy, as these models need to function with just a limited amount of data to learn physical rules in 3D space accurately. Fortunately, applying geometric deep learning to molecule generation has gained attention in the scientific community in recent years, paving the way for innovative approaches. These result in diffusion models quickly becoming state-of-the-art in this area due to their ability to effectively learn complex data distributions (Hoogeboom et al., 2022; Igashov et al., 2022; Schneuing et al., 2023; Vignac et al., 2023; Guan et al., 2023). While this has enabled researchers to develop generative models for molecular design that can sample novel molecules in 3D space, several drawbacks and open questions remain prevalent for practitioners. Molecule gener-

[*]Equal contribution. Correspondence to {tuan.le, julian.cremer}@pfizer.com

ative models are required to both generate realistic molecules in 3D space and preserve fundamental chemical rules, i.e., correct bonding and valencies. Various design decisions have to be taken into account that heavily impact the performance and complexity of those models. Hence, there is a high need to better understand the design space of diffusion models for molecular modeling. Moreover, the availability of molecular data is not as abundant, confronting ML models with relatively narrow and specific data distributions. That is, ML models are usually trained explicitly for each data set, which is unfavorable regarding the efficient use of training data and computing resources.

This work introduces the E(3)-equivariant graph attention denoising neural network EQGAT-diff. We systematically explore the design space of 3D equivariant diffusion models, including various parameterizations, loss weightings, data, and input feature modalities. Beyond that, we explore an efficient pre-training scheme on molecular data with implicit hydrogens. This enables a data- and time-efficient training and fine-tuning procedure leading to higher molecule stability. Our contributions are the following:

- We propose EQGAT-diff – a fast and accurate 3D molecular diffusion model that employs E(3)-equivariant graph attention. Our proposed model achieves SOTA results in shorter training time and with less trainable parameters than previous architectures.
- We systematically explore various design choices for 3D molecular diffusion models and provide a thorough ablation study across the popular benchmark sets QM9 and GEOM-Drugs. We propose a time-dependent loss weighting as a crucial component for fast training convergence, better inference speed, and sample quality.
- We demonstrate the transferability of an EQGAT-diff model pre-trained on the PubChem3D dataset to smaller but complex molecular datasets. After a short fine-tuning on the target distribution, we show that the model outperforms models trained from scratch on the target data by only training on subsets.
- We extend the diffusion process by modeling chemically motivated additional features and show a further significant increase in performance.

In summary, we found the following ingredients to be crucial: E(3)-equivariant graph attention, time-dependent loss weighting, unconditional pretraining on large databases comprising 3D conformers like PubChem3D, and adding chemical features like aromaticity and hybridization state as feature input to the denoising diffusion model.

2 Related Work

Denoising diffusion probabilistic models (DDPM) (Sohl-Dickstein et al., 2015; Ho et al., 2020; Kingma et al., 2021; Song et al., 2021b) have achieved great success in various generation tasks due to their remarkable ability to model complicated distributions in the image and text processing community (Popov et al., 2021; Kong et al., 2021; Salimans & Ho, 2022; Rombach et al., 2022; Karras et al., 2022; Li et al., 2022; Kingma & Gao, 2023). Deep generative modeling in the life sciences has become a promising research area, e.g., conditional conformer generation based on the 2D molecular graph, in which (Mansimov et al., 2019; Simm & Hernandez-Lobato, 2020) leverage the idea of variational autoencoders, while recent work by (Xu et al., 2022; Jing et al., 2022) use DDPMs, to predict the 3D coordinates with the help of 3D equivariant graph neural networks. In the *de novo* setting, another line of research focuses on directly generating the atomic coordinates and elements, using either autoregressive models (Gebauer et al., 2019; 2022; Luo & Ji, 2022), where atomic elements are generated one by one sequentially, or neural learning algorithms based on continuous normalizing flows (Satorras et al., 2021) that are computationally expensive due to the integration of an ordinary differential equation, leading to limited performance and scalability on large molecular systems. Diffusion models offer efficient training by progressively applying Gaussian noise to transform a complex data distribution to approximately tractable Gaussian prior, intending to learn the reverse process. Hoogeboom et al. (2022) introduced E(3) equivariant diffusion model (EDM) for *de novo* molecule design that simultaneously learns atomic elements next to the coordinates while treating chemical elements as continuous variables to utilize the formalism of DDPM. Follow-up works leverage EDM and develop diffusion models for linker design (Igashov et al., 2022) or ligand-protein complex modeling (Schneuing et al., 2023). Another line of work leverages the formalism of stochastic differential equations (SDEs) (Song et al., 2021b) and

Schroedinger Bridges with extension to manifolds (De Bortoli et al., 2021; 2022) to generating 3D conformer of a fixed molecule into a protein pocket (Corso et al., 2023), while (Wu et al., 2022) modifies the forward diffusion process to incorporate physical priors.

3 BACKGROUND

Problem Formulation and Notation We investigate the generation of molecular structures in a *de novo* setting, where atomic coordinates, chemical elements, and the bond topology are sampled. A molecular structure is given by , where the vertices $V = (v_1, \dots, v_N)$ refer to the N atoms. Each vertex is a tuple $v_i = (r_i, h_i)$ comprised of the atomic coordinate in 3D space r_i and chemical element h_i. The latter is one-hot encoded for K elements, i.e., $h_i = (0, 0, \dots, 1, 0)^\top$. The edges $E = (e_{ij})_{i,j=0}^N$ describe the connectivity of the molecule, where each edge feature can take five distinct values, namely the existence of no bond or a single-, double-, triple- or aromatic bond between atom i and j. Additionally, we exclude self-loops in our data representation. We write node features as matrices $\mathbf{X} \in \mathbb{R}^{N\times 3}$ and $\mathbf{H} \in \{0,1\}^{N\times K}$, while the bond topology is given by $\mathbf{E} \in \{0,1\}^{N\times N\times 5}$. We aim to develop a probabilistic model that is invariant to the permutation of atoms of the same chemical element and roto-translation of coordinates in 3D space. That means, regardless of how atom indices in the node-feature matrix $\mathbf{H}$ are shuffled and coordinates $\mathbf{X}$ roto-translated, the probability for a molecular structure $\mathcal{X}$ remains unchanged.

3.1 DENOISING DIFFUSION PROBABILISTIC MODELS

Discrete-time diffusion models (Sohl-Dickstein et al., 2015; Ho et al., 2020) are latent variable generative models characterized by a forward and reverse Markov process over T steps. Given a sample from the data distribution $x_0 \sim q(x_0)$, the forward process $q(x_{1:T}|x_0) = \prod_{t=1}^T q(x_t|x_{t-1})$ transforms it into a sequence of increasingly noisy latent variables $x_{1:T} = (x_1, x_2, \dots, x_T)$ and $x_i \in \mathcal{X}$. The learnable reverse Markov process $p_\theta(x_{0:T}) = p(x_T)\prod_{t=1}^T p_\theta(x_{t-1}|x_t)$ is trained to gradually denoise the latent variables approaching the data distribution. Sohl-Dickstein et al. (2015) initially proposed a diffusion process for binary and continuous data, while the latter consists of Gaussian transition kernels. The learning process for discrete data has been introduced by Hoogeboom et al. (2021) and Austin et al. (2021), leveraging categorical transition kernels in the form of doubly stochastic matrices. Crucially, both forward processes define tractable distributions determined by a noise schedule $\{\beta t\}_{t=1}^T$, such that the reverse generative model can be trained efficiently. As molecular data consists of atoms, bonds, and 3D coordinates, recent work leverages a combination of Gaussian and categorical diffusion for 3D molecular generation (Peng et al., 2023; Vignac et al., 2023; Guan et al., 2023). A subtle property of tractable transition kernels is that the distribution of a noisy state conditioned on a data sample is also tractable, and for continuous or discrete data follows a multivariate normal or categorical distribution

$$q(\mathbf{x}_t|\mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t|\sqrt{\bar{\alpha}}_t\mathbf{x}_0, (1-\bar{\alpha}_t)\mathbf{I}) \quad \text{and} \quad q(\mathbf{c}_t|\mathbf{c}_0) = \mathcal{C}(\mathbf{c}_t|\bar{\alpha}_t\mathbf{c}_0 + (1-\bar{\alpha}_t)\tilde{\mathbf{c}}), \tag{1}$$

where $\bar{\alpha}_t = \prod_{k=1}^t(1-\beta_k) \in (0,1)$, and $(1-\bar{\alpha}_t)$ determine a *variance-preserving* (VP) noise scheduler Song et al. (2021b). The vector $\tilde{\mathbf{c}}$ with $\tilde{\mathbf{c}}^\top\mathbf{1}_K = 1$ determines the prior distribution of the categorical diffusion, as $\bar{\alpha}_T \to 0$. Possible prior distributions are the uniform distribution over K-classes or the empirical distribution of categories in a dataset. In this work, we perturb atomic coordinates $\mathbf{X}$, chemical elements $\mathbf{H}$, and edge features $\mathbf{E}$ independently, using Gaussian and categorical diffusion. To conserve the edge-symmetry between atoms i and j, we only perturb the upper-triangular elements of $\mathbf{E}$. Diffusion models are trained by maximizing the variational lower-bound of the data log-likelihood (Sohl-Dickstein et al., 2015; Kingma et al., 2021; Austin et al., 2021) decomposed as $\log p(x) \geq L_0 + L_{\text{prior}} + \sum_{t=1}^{T-1} L_t$, where $L_0 = \log p(x_0|x_1)$ and $L_{\text{prior}} = -D_{KL}(q(x_T|p(x_T))$ denote the reconstruction, and prior loss. These two loss terms are commonly neglected during optimization, while the diffusion loss $L_t = -D_{KL}[q(x_{t-1}|x_t, x_0)|p_\theta(x_{t-1}|x_t)]$ has a closed-form expression since $q(x_{t-1}|x_t, x_0)$ is either a multivariate normal or categorical distribution, enabling efficient KL divergence minimization by predicting the corresponding distribution parameters. These are defined as a function of x_t and x_0, implying that the diffusion model is tasked to predict the clean data sample $\hat{x}_0$ to optimize L_t (Ho et al., 2020; Austin et al., 2021).

4 EQGAT-DIFF

An essential requirement to obtain a data-efficient model is to reflect the permutational symmetry of atoms of the same chemical element and the roto-translational symmetries of 3D molecular structures. In machine learning force fields, it has been shown that rotationally invariant features alone do not accurately represent the 3D molecular structure and hence require higher-order equivariant features (Schütt et al., 2021; Batzner et al., 2022; Thölke & Fabritiis, 2022; Batatia et al., 2022).

In short, a function $f : \mathcal{X} \to \mathcal{Y}$ mapping from input space $\mathcal{X}$ to output space $\mathcal{Y}$ is equivariant to the group G iff $f(g.x) = g.f(x)$, where $g.$ denotes the action of the group element $g \in G$ on an object $x, y \in \mathcal{X}, \mathcal{Y}$. As graph neural networks operate on graphs and map nodes into a feature space through shared transformations among all nodes, permutation equivariance is naturally preserved Bronstein et al. (2021). In contrast, point clouds are embedded in 3D space, so we additionally consider the rotation, reflection, and translation group in $\mathbb{R}^3$, often abbreviated as E(3). For the atomic coordinates, we require that $f(\mathbf{X}\mathbf{Q} + \mathbf{t}) = f(\mathbf{X})\mathbf{Q} + \mathbf{t}$, where $\mathbf{Q} \in O(3)$ is a rotation or reflection matrix and $\mathbf{t} \in \mathbb{R}^3$ a translation vector added row-wise. Group equivariance of a function f in the context of a diffusion model for molecular data is a requirement to preserve the group invariance for a probability density, as shown by Köhler et al. (2020) and Xu et al. (2022). To better address the challenge of molecular modeling, we propose a modified version of the EQGAT architecture Le et al. (2022), coined EQGAT-diff, which leverages attention-based feature aggregation of neighboring nodes. EQGAT-diff employs rotation equivariant vector features that can be interpreted as learnable vector bundles, which the denoising networks of EDM Hoogeboom et al. (2022) and MiDi Vignac et al. (2023) are lacking. Point clouds are modeled as fully connected graphs, so message passing computes all pairwise interactions. Equivariant vector features are obtained through a tensor product of scalar features with normalized relative positions $\mathbf{x}_{(ji,n)} = \frac{1}{||\mathbf{x_j} - \mathbf{x}_i||}(\mathbf{x}_j - \mathbf{x}_i)$ as similarly proposed in the works of Jing et al. (2021) and Schütt et al. (2021). We iteratively update hidden edge features within the EQGAT-diff architecture to handle the edge prediction between two atoms. To achieve this, we modify the message function of EQGAT as

$$\mathbf{m}_{ji}^{(l)} = \text{MLP}([\mathbf{h}_j^{(l)}; \mathbf{h}_i^{(l)}; \mathbf{W}_{e_0}^{(l)}\mathbf{e}_{ji}^{(l)}; d_{ji}^{(l)}; d_j^{(l)}; d_i^{(l)}; \mathbf{p}_j^{(l)} \cdot \mathbf{p}_i^{(l)}]),$$

where ; denotes concatenation of E(3) invariant embeddings and MLP is a 2-layer multi-layer perceptron. The message embedding $\mathbf{m}_{ji}^{(l)} = (\mathbf{a}_{ji}^{(l)}, \mathbf{b}_{ji}^{(l)}, \mathbf{c}_{ji}^{(l)}, \mathbf{d}_{ji}^{(l)}, s_{ji}^{(l)})^\top \in \mathbb{R}^K$ is further split into sub-embeddings that serve as filter to aggregate node information from all other source nodes j.

$$\mathbf{h}_i^{(l+1)} = \mathbf{h}_i^{(l)} + \sum_j \frac{\exp(\mathbf{a}_{ji}^{(l)})}{\sum_{j'} \exp(\mathbf{a}_{j'i}^{(l)})} \mathbf{W}_h^{(l)} \mathbf{h}_j^{(l)} \quad \text{and} \quad \mathbf{e}_{ji}^{(l+1)} = \mathbf{W}_{e_1}^{(l)} \sigma(\mathbf{e}_{ji}^{(l)} + \mathbf{d}_{ji}^{(l)}),$$

$$\mathbf{v}_i^{(l+1)} = \mathbf{v}_i^{(l)} + \frac{1}{N} \sum_j \mathbf{x}_{ji,n} \otimes \mathbf{b}_{ji}^{(l)} + (\mathbf{1} \otimes \mathbf{c}_{ji}^{(l)}) \odot \mathbf{v}_j^{(l+1)} \mathbf{W}_v^{(l)},$$

$$\mathbf{x}_i^{(l+1)} = \mathbf{x}_i^{(l)} + \frac{1}{N} \sum_j s_{ji}^{(l)} \mathbf{x}_{ji,n}^{(l)},$$

where $\mathbf{1} = (1, 1, 1)^\top$ and σ is the SiLU activation function. The embeddings are further updated and normalized with details explained in the Appendix A.1.

5 EXPLORING THE DESIGN SPACE OF 3D MOLECULAR DIFFUSION MODELS

The design space of diffusion models has many degrees of freedom concerning, among others, the data representation, training objective, forward inference process, and the denoising neural network. In *de novo* 3D molecular generation, Hoogeboom et al. (2022) (EDM) utilized the ϵ-parameterization and proposed to model chemical elements as well as atomic positions continuously. Vignac et al. (2023) proposed MiDi, which generates the molecular graph and 3D structure simultaneously. This model uses the x_0-parameterization and employs the framework developed by Austin et al. (2021) to model not only chemical elements but also formal charges and bond types in discrete state space. Both parameterizations optimize the same objective, i.e., aiming to minimize the KL divergence $D_{KL}[q(x_{t-1}|x_t, x_0)|p_\theta(x_{t-1}|x_t)]$. Ho et al. (2020) found that optimizing the diffusion model in noise-space on images results in improved generation performance than predicting

Table 1: Comparison of EQGAT-diff on QM9 and GEOM-Drugs trained with w_u or $w_s(t)$ loss-weighting. We report the mean values over five runs of selected evaluation metrics with the margin of error for the 95% confidence level given as subscripts. The best results are in bold.

	QM9			GEOM-Drugs		
Weighting	Mol. Stability ↑	Validity ↑	Connect. Comp. ↑	Mol. Stability ↑	Validity ↑	Connect. Comp. ↑
w_u	$97.39_{\pm 0.23}$	$97.99_{\pm 0.20}$	$99.70_{\pm 0.03}$	$87.59_{\pm 0.19}$	$71.44_{\pm 0.22}$	$86.57_{\pm 0.33}$
$w_s(t)$	$\mathbf{98.68}_{\pm 0.11}$	$\mathbf{98.96}_{\pm 0.07}$	$\mathbf{99.94}_{\pm 0.03}$	$\mathbf{91.60}_{\pm 0.14}$	$\mathbf{84.02}_{\pm 0.19}$	$\mathbf{95.08}_{\pm 0.12}$

the original image from a noised version. While noise prediction might benefit the image domain, this does not necessarily generalize to 3D molecular data. In fact, MiDi outperforms EDM across all standard benchmark metrics and datasets. However, whether the improved performance stems from the x_0-parameterization, the employment of categorical diffusion for discrete features, or using bond types and other chemical features has still been unclear, leaving researchers and practitioners guessing which kind of diffusion model to deploy in their respective tasks.

In this section, we explore the design space of *de novo* molecular diffusion models in these three aspects while consistently using EQGAT-diff as the denoising neural network to isolate the effect of each change for better comparison. The diffusion models are evaluated on the QM9 dataset (Ramakrishnan et al., 2014) containing molecules with up to 9 heavy atoms, and the GEOM-Drugs dataset (Axelrod & Gómez-Bombarelli, 2022) containing up to 15 heavy atoms. We utilize the data splits from Vignac et al. (2023) and benchmark all models on full molecular 3D graphs that include explicit hydrogens.

5.1 Training Details

We either employ noise prediction (ϵ-parameterization) or data prediction (x_0-parameterization) to train EQGAT-diff , such that the group equivariant network $f_\theta(x_t)$ receives a noisy molecule $x_t = (\mathbf{X}_t, \mathbf{H}_t, \mathbf{E}_t)$ and either outputs the applied noise $\hat{\epsilon}_t = (\hat{\epsilon}_{\mathbf{X}_t}, \hat{\epsilon}_{\mathbf{H}_t}, \hat{\epsilon}_{\mathbf{E}_t})$ or a prediction of the clean data $\hat{x}_0 = (\hat{\mathbf{X}}_0, \hat{\mathbf{H}}_0, \hat{\mathbf{E}}_0)$ of coordinates, chemical elements as well as bonds. We draw a random batch of molecules and uniformly sample steps $t \in \mathcal{U}(1, T)$ and optimize the diffusion loss L_t for each sample. While we use the mean squared error loss for the ϵ-model, the x_0-model is optimized using loss functions l_d depending on the data modality d. Here, l_d is a mean squared error for continuous and the cross-entropy loss for categorical data. This leads to a composite loss

$$L_{t,\epsilon} = w(t)||\epsilon_t - \hat{\epsilon}_\theta(x_t, t)||^2 \quad \text{and} \quad L_{t,x_0} = w(t) \cdot l_d(x_0, \hat{x}_\theta(x_t, t); \lambda_m), \tag{2}$$

where λ_m denotes a modality-dependent weighting, which we adopt from Vignac et al. (2023) and set to $\lambda_x = 3, \lambda_h = 0.4, \lambda_e = 2$. For noise learning, we adopt an atom-type feature scaling of 0.25 as in Hoogeboom et al. (2022). Notably, $w(t)$ is a loss weighting commonly set to 1 across all time steps, which has been previously found to work best (Ho et al., 2020). In contrast to this result, we find this term to be crucial for molecular design, as discussed in Sec. A.4. Following Vignac et al. (2023), we also employ an adaptive noise schedule (see Appendix A.1.1).

5.2 Metrics

Following (Hoogeboom et al., 2022), we measure validity using the success rate of RDKit sanitization over 10,000 molecules (pre-selecting connected components only) - with the caveat that the RDKit sanitization might add implicit hydrogens to the system to satisfy the chemical constraints. Therefore, checking atomic and molecular stability for the correct valencies using a predefined lookup table that complements the validation is essential. Further, we propose to include diversity/similarity measures. We evaluate the diversity of sampled molecules using the average Tanimoto distance and measure the similarity with the training dataset via Kullback-Leibler divergence and the Tanimoto distance. Lastly, following Vignac et al. (2023), we use the atom and bond total variations (AtomsTV and BondsTV) that measure the l_1 distance between the marginal distribution of atom types and bond types for the generated set and the test set, respectively. Moreover, we employ the Wasserstein distance between valencies, bond lengths, and bond angles, with the latter two being 3D metrics to evaluate conformer accuracy. For more details, we refer to Vignac et al. (2023) and Appendix A.2.

Kingma et al. (2021) have shown that the intermediate KL-divergence loss L_t in the variational lower bound (VLB) for a Gaussian diffusion can be simplified to

$$L_t = \frac{1}{2}(w(t))||x_0 - x_\theta(x_t, t)||_2^2 = \frac{1}{2}\mathbb{E}_{\epsilon\sim\mathcal{N}(0,I)}[(\text{SNR}(t-1) - \text{SNR}(t))||x_0 - x_\theta(x_t, t)||_2^2],$$

where $\text{SNR}(t) = \frac{\bar{\alpha}_t}{1-\bar{\alpha}_t}$ refers to the signal-to-noise ratio. However, the weighting coefficients in diffusion models for molecules are commonly set to 1, i.e., $w_u = 1$ in EDM or MiDi (Hoogeboom et al., 2022; Vignac et al., 2023).

We hypothesize that denoising requires high accuracy for timesteps close to the data distribution to generate valid molecules, while errors close to the noise distribution are neglectable. Such loss weighting has been proposed by Salimans & Ho (2022) as 'truncated SNR', which we modify for our use case. Specifically, we perform experiments with the loss weighting

$$w_s(t) = \max(0.05,\ \min(1.5,\ \text{SNR}(t))), \tag{3}$$

which matches our hypothesis about learning with higher weightings approaching the data distribution (see A.4.1 and Fig. 5). We clip the maximum value of 1.5 to enforce larger weightings to enhance learning compared to uniform weighting, followed by an abrupt exponential decay. We train EQGAT-diff using Gaussian diffusion on atomic coordinates and categorical diffusion for chemical elements, formal charges, and bond features following the parameterization proposed by Vignac et al. (2023), predicting a clean data sample $\hat{x}_0$ given a noisy version x_t. As shown in Table 1, training EQGAT-diff on GEOM-Drugs with $w_s(t)$ results in a better generative model that can sample molecules preserving chemistry rules, measured in increased molecule stability of 91.60%, compared to the EQGAT-diff which was trained with w_u, only achieving 87.59%. As the $w_s(t)$ loss weighting achieved better evaluation metrics and significantly faster training convergence on the QM9 and GEOM-Drugs datasets, we choose it as default for the following experiments conducted in this work. We provide further empirical evidence in Appendix A.3

Table 2: Overall performance of EQGAT-diff on QM9 and GEOM-Drugs for discrete and continuous diffusion as well as noise (ϵ) and data learning (x_0). Discrete or continuous diffusion is denoted as 'disc' and 'cont', respectively, given as subscripts, ϵ- and x_0-parameterization as superscripts. We report mean values over five sampling runs with 95% confidence intervals as subscripts. The best results are in bold.

Dataset	**QM9**			**GEOM-Drugs**		
Model	EQGAT$^{x0}_{disc}$	EQGAT$^{x0}_{cont}$	EQGAT$^{\epsilon}_{cont}$	EQGAT$^{x0}_{disc}$	EQGAT$^{x0}_{cont}$	EQGAT$^{\epsilon}_{cont}$
Mol. Stab. ↑	**98.68**$_{\pm 0.11}$	96.45$_{\pm 0.17}$	96.18$_{\pm 0.16}$	**91.60**$_{\pm 0.14}$	90.46$_{\pm 0.09}$	85.19$_{\pm 0.72}$
Atom. Stab ↑	**99.92**$_{\pm 0.00}$	99.79$_{\pm 0.01}$	99.68$_{\pm 0.02}$	**99.72**$_{\pm 0.01}$	**99.73**$_{\pm 0.01}$	99.32$_{\pm 0.04}$
Validity ↑	**98.96**$_{\pm 0.07}$	96.79$_{\pm 0.15}$	97.04$_{\pm 0.17}$	**84.02**$_{\pm 0.19}$	80.96$_{\pm 0.38}$	79.13$_{\pm 0.58}$
Connect. Comp. ↑	**99.94**$_{\pm 0.03}$	99.82$_{\pm 0.05}$	99.71$_{\pm 0.03}$	**95.08**$_{\pm 0.12}$	93.30$_{\pm 0.21}$	94.10$_{\pm 0.48}$
Novelty ↑	64.03$_{\pm 0.24}$	60.96$_{\pm 0.54}$	**73.40**$_{\pm 0.32}$	**99.87**$_{\pm 0.04}$	**99.83**$_{\pm 0.04}$	99.82$_{\pm 0.0}$
Uniqueness ↑	**100.00**$_{\pm 0.00}$	**100.0**$_{\pm 0.00}$	**100.00**$_{\pm 0.00}$	**100.00**$_{\pm 0.00}$	**100.00**$_{\pm 0.00}$	**100.00**$_{\pm 0.00}$
Diversity ↑	91.72$_{\pm 0.02}$	91.51$_{\pm 0.03}$	**91.89**$_{\pm 0.03}$	**89.00**$_{\pm 0.03}$	88.87$_{\pm 0.04}$	88.97$_{\pm 0.05}$
KL Divergence ↑	**91.36**$_{\pm 0.29}$	**91.41**$_{\pm 0.54}$	88.97$_{\pm 0.31}$	87.17$_{\pm 0.34}$	87.35$_{\pm 0.35}$	**87.70**$_{\pm 0.58}$
Train Similarity ↓	0.076$_{\pm 0.00}$	0.076$_{\pm 0.00}$	**0.075**$_{\pm 0.00}$	**0.113**$_{\pm 0.00}$	0.114$_{\pm 0.00}$	0.114$_{\pm 0.00}$
AtomsTV [10^{-2}] ↓	1.0$_{\pm 0.00}$	2.0$_{\pm 0.00}$	2.7$_{\pm 0.00}$	3.4$_{\pm 0.10}$	3.6$_{\pm 0.10}$	**2.9**$_{\pm 0.20}$
BondsTV [10^{-2}] ↓	1.2$_{\pm 0.00}$	1.8$_{\pm 0.00}$	1.2$_{\pm 0.00}$	**2.4**$_{\pm 0.00}$	**2.4**$_{\pm 0.00}$	**2.4**$_{\pm 0.00}$
ValencyW$_1$ [10^{-2}] ↓	0.6$_{\pm 0.10}$	1.9$_{\pm 0.00}$	0.9$_{\pm 0.00}$	1.2$_{\pm 0.10}$	1.9$_{\pm 0.10}$	1.6$_{\pm 0.00}$
BondLenghtsW$_1$ [10^{-2}] ↓	**0.2**$_{\pm 0.10}$	0.5$_{\pm 0.00}$	**0.2**$_{\pm 0.10}$	**0.2**$_{\pm 0.10}$	0.3$_{\pm 0.00}$	0.7$_{\pm 0.40}$
BondAnglesW$_1$ ↓	**0.42**$_{\pm 0.03}$	1.86$_{\pm 0.06}$	0.52$_{\pm 0.03}$	**0.92**$_{\pm 0.02}$	0.95$_{\pm 0.02}$	1.07$_{\pm 0.06}$

5.3 Diffusion Parameterization: ϵ vs x_0 and discrete vs continuous

Diffusion models for continuous data are commonly implemented using the ϵ-parameterization Ho et al. (2020), which is connected to denoising score matching models proposed by Song & Ermon (2019). Diffusion models have quickly adapted this setting for 3D molecular design (Hoogeboom et al., 2022; Igashov et al., 2022; Schneuing et al., 2023). However, no comparative study of x_0 and ϵ-parameterization in this domain has been performed yet. To close this gap, we benchmark the ϵ- vs. the x_0-parameterization on data modalities subject to a Gaussian diffusion. That is, we treat all node features (including atomic elements, charges, and coordinates) as well as the bond features as continuous variables and optimize our diffusion model using either the ϵ- or x_0-parameterization with the loss functions defined in Eq. (2).

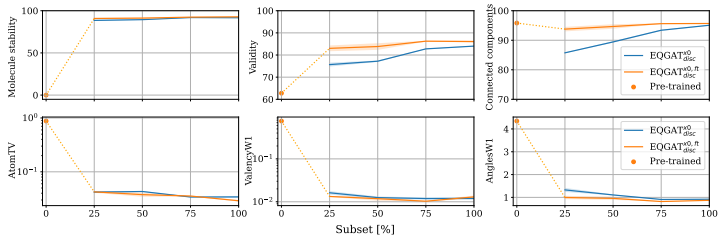


Figure 1: Selected evaluation metrics for EQGAT-diff trained on GEOM-Drugs subsets (25, 50, 75%) from scratch or fine-tuned. We also report the results of the pre-trained, not fine-tuned model (0%).

In the following, we abbreviate EQGAT-diff with EQGAT to keep the notation clear, depicting the diffusion type subscripted and the parameterization superscripted. Table 2 shows that the x_0-parameterization (EQGAT$^{x_0}_{cont}$) achieves higher molecule stability on QM9 and GEOM-Drugs than the ϵ-parameterization (EQGAT$^{\epsilon}_{cont}$). The performance gap is pronounced on the GEOM-Drugs dataset, which covers a broader range of larger and more complex molecules. On this more demanding benchmark, EQGAT$^{x_0}_{cont}$ outperforms the ϵ-model with 90.46% molecule stability against 85.19%. The lower molecule stability for the ϵ-model is due to the molecular graph not being accurately denoised during the sampling. Thus, the final edge features do not preserve the valency constraints of the chemical elements.

Next, we compare how the choice of categorical or Gaussian diffusion for modeling the chemical elements, charges, and edge features affects the generation performance. Recall that the noising process in the categorical diffusion perturbs the one-hot encoding of discrete features by jumping from one class to another, or staying on the same class. Alternatively, noise from a multivariate normal distribution is added to the (scaled) one-hot encodings, as described in Eq. (1). For both settings, the diffusion models (EQGAT$^{x_0}_{disc}$ and EQGAT$^{x_0}_{cont}$) are tasked with predicting the original data point x_0, as there is no ϵ-parameterization when employing categorical diffusion. The previous ablation has shown that data prediction is superior to noise prediction when dealing with molecular data in a continuous setting. We discover that EQGAT$^{x_0}_{disc}$ outperforms EQGAT$^{x_0}_{cont}$ in all evaluation metrics on the QM9 and GEOM-Drugs dataset as shown in Table 2. Hence, employing the categorical diffusion for discrete state-space in the x_0-parameterization is the preferred choice.

6 Transferability of Molecular Diffusion Models

In many molecular design scenarios, only a limited amount of training data is available for a desired target distribution, e.g., in structure-based drug design. However, 3D generative molecular diffusion models require a lot of training data to yield a high ratio of valid and novel molecules. This section investigates how well a diffusion model pre-trained on a general large set of molecules transfers to a target distribution specified by a small training set of complex molecular structures. We use the PubChem3D dataset Bolton et al. (2011) for pre-training, which consists of roughly 95.7 million compounds from the PubChem database. It includes all molecules with chemical elements H, C, N, O, F, Si, P, S, Cl, Br, and I with less than 50 non-hydrogen atoms and a maximum of 15 rotatable bonds. The 3D structures have been computed using OpenEye's OMEGA software (Hawkins & Nicholls, 2012). We train EQGAT-diff on PubChem3D on four Nvidia A100 GPUs for one epoch (~ 24 hours). Interestingly, we found that by reducing the size of molecular graphs using only implicit hydrogens, we could reduce the

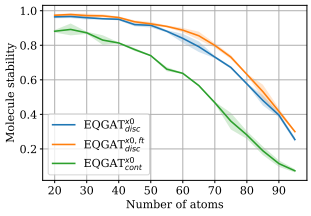


Figure 2: Comparing EQGAT$^{x_0,ft}_{disc}$ with EQGAT$^{x_0}_{disc}$, and EQGAT$^{x_0}_{cont}$ regarding molecule stability of 600 generated molecules with an increasing number of atoms. Standard deviations are plotted in shaded areas.

Table 3: Comparison of $\text{EQGAT}_{\text{disc}}$ models trained for 800 epochs on GEOM-Drugs. The superscripts 'ft' and 'af' abbreviate *fine-tuned* and *additional-features*. The margin of error for the 95% confidence level is given as subscripts. We also compare EDM and the current SOTA, MiDi. Training details for MiDi are given in Appendix A.6. The best results are in bold.

Dataset	**GEOM-Drugs**					
Model	$\text{EQGAT}^{x0}_{\text{disc}}$	$\text{EQGAT}^{x0,ft}_{\text{disc}}$	$\text{EQGAT}^{x0,af}_{\text{disc}}$	$\text{EQGAT}^{x0,af,ft}_{\text{disc}}$	EDM	MiDi
Mol. Stab. ↑	$93.11_{\pm 0.31}$	$93.92_{\pm 0.13}$	$\mathbf{94.51}_{\pm 0.18}$	$\mathbf{95.01}_{\pm 0.37}$	40.3	$89.7_{\pm 0.60}$
Atom. Stab ↑	$99.79_{\pm 0.01}$	$\mathbf{99.81}_{\pm 0.01}$	$\mathbf{99.83}_{\pm 0.01}$	$\mathbf{99.84}_{\pm 0.00}$	97.8	$99.7_{\pm 0.01}$
Validity ↑	$85.86_{\pm 0.33}$	$\mathbf{88.04}_{\pm 0.17}$	$87.89_{\pm 0.31}$	$\mathbf{88.42}_{\pm 0.26}$	87.8	$70.5_{\pm 0.41}$
Connect. Comp. ↑	$\mathbf{96.32}_{\pm 0.25}$	$\mathbf{96.57}_{\pm 0.18}$	$96.36_{\pm 0.25}$	$\mathbf{96.71}_{\pm 0.20}$	41.4	$88.76_{\pm 0.55}$
Novelty ↑	$99.82_{\pm 0.05}$	$99.84_{\pm 0.02}$	$99.82_{\pm 0.05}$	$99.82_{\pm 0.03}$	**100.00**	$\mathbf{100.00}_{\pm 0.00}$
Diversity ↑	$\mathbf{89.03}_{\pm 0.03}$	$\mathbf{89.05}_{\pm 0.05}$	$\mathbf{88.98}_{\pm 0.02}$	$\mathbf{88.96}_{\pm 0.01}$	-	-
KL Divergence ↑	$87.66_{\pm 0.31}$	$87.58_{\pm 0.56}$	$\mathbf{88.38}_{\pm 0.25}$	$87.62_{\pm 0.19}$	-	-
Train Similarity ↓	$0.114_{\pm 0.0}$	$\mathbf{0.113}_{\pm 0.0}$	$0.114_{\pm 0.0}$	$0.114_{\pm 0.0}$	-	-
AtomsTV $[10^{-2}]$ ↓	$3.02_{\pm 0.08}$	$3.02_{\pm 0.10}$	$\mathbf{2.88}_{\pm 0.10}$	$\mathbf{2.91}_{\pm 0.10}$	21.2	$5.11_{\pm 0.19}$
BondsTV $[10^{-2}]$ ↓	$2.44_{\pm 0.01}$	$\mathbf{2.40}_{\pm 0.00}$	$2.42_{\pm 0.00}$	$\mathbf{2.40}_{\pm 0.00}$	4.8	$2.44_{\pm 0.00}$
ValencyW_1 $[10^{-2}]$ ↓	$1.18_{\pm 0.09}$	$1.20_{\pm 0.00}$	$\mathbf{0.85}_{\pm 0.12}$	$\mathbf{0.90}_{\pm 0.10}$	28.5	$2.48_{\pm 0.52}$
BondLenghtsW_1 $[10^{-2}]$ ↓	$0.56_{\pm 0.38}$	$\mathbf{0.10}_{\pm 0.00}$	$0.50_{\pm 0.51}$	$0.20_{\pm 0.10}$	0.2	$0.2_{\pm 0.10}$
BondAnglesW_1 ↓	$0.83_{\pm 0.03}$	$0.79_{\pm 0.02}$	$0.65_{\pm 0.01}$	$\mathbf{0.62}_{\pm 0.01}$	6.23	$1.73_{\pm 0.32}$

pre-training time significantly without sacrificing performance in fine-tuning. For a comparison to keeping explicit hydrogens in the pre-training, see Appendix A.5. During fine-tuning, the diffusion model is tasked to adapt to the distribution of another dataset, now including explicit hydrogens.

To evaluate the effectiveness of pre-training, we fine-tune subsets of $(25, 50, 75\%)$ of the QM9 and GEOM-Drugs datasets. Our results suggest that using a pre-trained model and subsequent fine-tuning shows consistently superior performance across datasets, partly by a large margin (see Fig. 1). We demonstrate the importance of pre-training by evaluating molecule stability, validity, and the number of connected components of a fine-tuned model compared to training from scratch on the full data and its $25, 50, 75\%$ subsets. As a reference point (0%), we show the pre-trained model without fine-tuning evaluated on the aforementioned metrics. Interestingly, the fine-tuned model shares similar (best) scores with $\text{EQGAT}^{x_0}_{disc}$ trained from scratch on 100% of the data when looking at atom type variation and valency as well as angle distance metrics using a hold-out test set as a reference. These metrics capture how well the model learns the underlying data distribution.

We find that the fine-tuned model effectively learns a distribution shift on GEOM-Drugs by only being trained on small subsets of the data. We list more detailed evaluation metrics and the evaluation on QM9 in Appendix A.3. Comparing the fine-tuned model $\text{EQGAT}^{x_0,ft}_{disc}$ with $\text{EQGAT}^{x_0}_{disc}$, and $\text{EQGAT}^{x_0}_{cont}$, respectively, shown in Fig. 2, we can also observe that the fine-tuning leads to significantly more stable predictions for larger molecules. We suspect that these findings might also apply to learning building blocks on large databases like the Enamine REAL Space to bias the generative model towards, e.g., higher synthesizability while ensuring an efficient distribution shift on the target distribution.

7 Inserting chemical domain knowledge

In the previous sections, we examined and outlined the importance of design choices when employing diffusion models for 3D molecular generation. Taking these results, we select the best two models - with and without fine-tuning: $\text{EQGAT}^{x0,ft}_{\text{disc}}$ and $\text{EQGAT}^{x0}_{\text{disc}}$ - and train them to full convergence, comparing with EDM and MiDi. We demonstrate in Tab. 3 that $\text{EQGAT}^{x0,ft}_{\text{disc}}$, and even more so $\text{EQGAT}^{x0,af}_{\text{disc}}$ and $\text{EQGAT}^{x0,af,ft}_{\text{disc}}$, outperform MiDi on all evaluation metrics by a large margin, while, most notably, our models converge significantly faster and are twice as fast computationally (see Appendix A.6). Given the demonstrated ability of diffusion models to learn the data distribution of complex molecular structures, we insert more chemical domain knowledge into the diffusion model, going beyond bonding. We additionally utilize aromaticity, ring correspondence, and hybridization states to provide a more comprehensive description of the molecular structure. The new additional features are independently perturbed using the categorical transition kernels (see Eq. (1)) and subsequently denoised by our model. We observe that these additional chemical features again improve the performance of our models ($\text{EQGAT}^{x0,af}_{\text{disc}}$ and $\text{EQGAT}^{x0,af,ft}_{\text{disc}}$) compared to our previous models as well as EDM and MiDi.

8 STRUCTURE-BASED DE NOVO LIGAND DESIGN

We train EQGAT-diff on the Crossdocked dataset Francoeur et al. (2020) for de novo structure-based ligand design. Following (Guan et al., 2023) and (Schneuing et al., 2023), we consider the protein pocket as a condition to generate novel ligands. Here, the pocket is seen as a fixed 3D context, while the ligand's coordinates, atom and bond types get diffused and denoised. In Tab. 4, we report the validity, number of connected components as well as the Wasserstein distances of bond lengths and angles between generated set to the training set, respectively. We observe that the finetuned model with timestep loss weighting significantly outperforms the models that are trained from scratch on all metrics. For the models trained from scratch, using timestep weighting shows better performance than no loss weighting. These results further underline the relevance of our findings allowing for an effective transfer of our model to structure-based molecule generation.

Table 4: Comparison of EQGAT-diff models trained on the Crossdocked dataset for pocket-conditioned de novo ligand generation. EQGAT$^{x0}_{disc}$ and EQGAT$^{x0,ft}_{disc}$ are compared with and without loss weighting, each trained for 300 epochs. Mean values are reported over five runs of selected evaluation metrics with the margin of error for the 95% confidence level given as subscripts and best results in bold.

Model	Validity ↑	Connect. Comp. ↑	BondLengths W1 [10^{-2}] ↓	BondAngles W1 ↓
EQGAT$^{x0}_{disc}(w_u)$	85.51$_{\pm 0.09}$	95.15$_{\pm 0.14}$	0.20$_{\pm 0.0}$	4.37$_{\pm 0.20}$
EQGAT$^{x0}_{disc}(w_s(t))$	89.62$_{\pm 0.08}$	97.65$_{\pm 0.11}$	0.12$_{\pm 0.0}$	2.12$_{\pm 0.26}$
EQGAT$^{x0,ft}_{disc}(w_s(t))$	**95.65**$_{\pm 0.12}$	**99.66**$_{\pm 0.10}$	**0.11**$_{\pm 0.0}$	**1.55**$_{\pm 0.21}$

Based on these results, we sample ligands from EQGAT$^{x0,ft}_{disc}$ for docking. Following Luo et al. (2021), Peng et al. (2022), we draw 100 valid ligands per protein pocket and evaluate them using Vina (Hassan et al., 2017) as an empirical proxy of the ligand binding affinity. As shown in Tab. 5, EQGAT$^{x0,ft}_{disc}$ outperforms both TargetDiff Guan et al. (2023) and DiffSBDD Schneuing et al. (2023) on the docking score and across all other metrics while generating more diverse ligands.

Table 5: Docking performance comparison between EQGAT$^{x0,ft}_{disc}$, TargetDiff and DiffSBDD trained on the Crossdocked dataset for pocket-conditioned de novo ligand generation. Best results in bold.

Model	Vina (All) ↓	Vina (Top-10%) ↓	QED ↑	SA ↑	Lipinski ↑	Diversity ↑
EQGAT$^{x0,ft}_{disc}(w_s(t))$	**-7.423**$_{\pm 2.33}$	-9.571$_{\pm 2.14}$	**0.522**$_{\pm 0.18}$	**0.697**$_{\pm 0.20}$	4.66$_{\pm 0.72}$	**0.742**$_{\pm 0.07}$
TargetDiff	-7.318$_{\pm 2.47}$	**-9.669**$_{\pm 2.55}$	0.483$_{\pm 0.20}$	0.584$_{\pm 0.13}$	4.594$_{\pm 0.83}$	0.718$_{\pm 0.09}$
DiffSBDD-cond	-6.950$_{\pm 2.06}$	-9.120$_{\pm 2.16}$	0.469$_{\pm 0.21}$	0.578$_{\pm 0.13}$	4.562$_{\pm 0.89}$	0.728$_{\pm 0.07}$

9 CONCLUSIONS

In this work, we have introduced EQGAT-diff, a framework for fast and accurate end-to-end differentiable *de novo* molecule generation in 3D space, jointly predicting geometry, topology, chemical composition and optionally other chemical features like the hybridization. The findings presented here are underpinned by comprehensive ablation studies, which address a previously scientific blank spot by thoroughly exploring the design space of 3D equivariant diffusion models. We have specifically designed an equivariant diffusion model that combines Gaussian and discrete state space diffusion. Crucially, we have incorporated a timestep-dependent loss weighting that significantly enhances the performance and training time of EQGAT-diff and, furthermore, showcased the transferability of our model being pre-trained on PubChem3D on small datasets. Our proposed models have significantly surpassed the current state-of-the-art 3D diffusion models, particularly in generating larger and more complex molecules, as evidenced by their high molecule stability and validity, which evaluate that chemistry rules are preserved. Most notably, we also showcased that our framework seamlessly transfers to target-conditioned de novo ligand design superior docking scores while ensuring high diversity in samples. Given these achievements, we anticipate our findings will open avenues for ML-driven *de novo* structure-based drug discovery.

CODE AVAILABILITY

Our source code and implementation will be released under https://github.com/pfizer-opensource/eqgat-diff.

ACKNOWLEDGMENTS

This study was partially funded by the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie Innovative Training Network European Industrial Doctorate grant agreement No. 956832 "Advanced machine learning for Innovative Drug Discovery."

REFERENCES

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=h7-XixPCAL.

Simon Axelrod and Rafael Gómez-Bombarelli. Geom, energy-annotated molecular conformations for property prediction and molecular generation. *Sci. Data*, 9(1):185, Apr 2022. ISSN 2052-4463. doi: 10.1038/s41597-022-01288-4. URL https://doi.org/10.1038/s41597-022-01288-4.

Christoph Bannwarth, Sebastian Ehlert, and Stefan Grimme. Gfn2-xtb—an accurate and broadly parametrized self-consistent tight-binding quantum chemical method with multipole electrostatics and density-dependent dispersion contributions. *J. Chem. Theory Comput.*, 15(3):1652–1671, Mar 2019. ISSN 1549-9618. doi: 10.1021/acs.jctc.8b01176. URL https://doi.org/10.1021/acs.jctc.8b01176.

Ilyes Batatia, David Peter Kovacs, Gregor N. C. Simm, Christoph Ortner, and Gabor Csanyi. MACE: Higher order equivariant message passing neural networks for fast and accurate force fields. *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=YPpSngE-ZU.

Simon Batzner, Albert Musaelian, Lixin Sun, Mario Geiger, Jonathan P. Mailoa, Mordechai Kornbluth, Nicola Molinari, Tess E. Smidt, and Boris Kozinsky. E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *Nat. Commun.*, 13(1), may 2022. doi: 10.1038/s41467-022-29939-5. URL https://doi.org/10.1038%2Fs41467-022-29939-5.

Evan E. Bolton, Jie Chen, Sunghwan Kim, Lianyi Han, Siqian He, Wenyao Shi, Vahan Simonyan, Yan Sun, Paul A. Thiessen, Jiyao Wang, Bo Yu, Jian Zhang, and Stephen H. Bryant. PubChem3D: a new resource for scientists. *Journal of Cheminformatics*, 3(1):32, September 2011. ISSN 1758-2946. doi: 10.1186/1758-2946-3-32. URL https://doi.org/10.1186/1758-2946-3-32.

Michael M. Bronstein, Joan Bruna, Taco Cohen, and Petar Velivckovi'c. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *ArXiv*, abs/2104.13478, 2021. URL https://api.semanticscholar.org/CorpusID:233423603.

Gabriele Corso, Hannes Stärk, Bowen Jing, Regina Barzilay, and Tommi S. Jaakkola. Diffdock: Diffusion steps, twists, and turns for molecular docking. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=kKF8_K-mBbS.

Valentin De Bortoli, James Thornton, Jeremy Heng, and Arnaud Doucet. Diffusion Schrödinger Bridge with Applications to Score-Based Generative Modeling. In *Advances in Neural Information Processing Systems*, volume 34, pp. 17695–17709. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/hash/940392f5f32a7ade1cc201767cf83e31-Abstract.html.

Valentin De Bortoli, Emile Mathieu, Michael Hutchinson, James Thornton, Yee Whye Teh, and Arnaud Doucet. Riemannian Score-Based Generative Modelling. In S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh (eds.), *Advances in Neural Information Processing Systems*, volume 35, pp. 2406–2422. Curran Associates, Inc., 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/file/105112d52254f86d5854f3da734a52b4-Paper-Conference.pdf.

Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat GANs on image synthesis. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=AAWuCvzaVt.

Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019. URL https://api.semanticscholar.org/CorpusID:70349949.

Paul G. Francoeur, Tomohide Masuda, Jocelyn Sunseri, Andrew Jia, Richard B. Iovanisci, Ian Snyder, and David R. Koes. Three-dimensional convolutional neural networks and a cross-docked data set for structure-based drug design. *Journal of Chemical Information and Modeling*, 60(9):4200–4215, Sep 2020. ISSN 1549-9596. doi: 10.1021/acs.jcim.0c00411. URL https://doi.org/10.1021/acs.jcim.0c00411.

Niklas Gebauer, Michael Gastegger, and Kristof Schütt. Symmetry-adapted generation of 3d point sets for the targeted discovery of molecules. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/a4d8e2a7e0d0c102339f97716d2fdfb6-Paper.pdf.

Niklas W. A. Gebauer, Michael Gastegger, Stefaan S. P. Hessmann, Klaus-Robert Müller, and Kristof T. Schütt. Inverse design of 3d molecular structures with conditional generative neural networks. *Nature Communications*, 13(1), feb 2022. doi: 10.1038/s41467-022-28526-y. URL https://doi.org/10.1038%2Fs41467-022-28526-y.

Jiaqi Guan, Wesley Wei Qian, Xingang Peng, Yufeng Su, Jian Peng, and Jianzhu Ma. 3d equivariant diffusion for target-aware molecule generation and affinity prediction. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=kJqXEPXMsE0.

Nafisa M Hassan, Amr A Alhossary, Yuguang Mu, and Chee-Keong Kwoh. Protein-ligand blind docking using QuickVina-W with inter-process spatio-temporal integration. *Sci. Rep.*, 7(1):15451, November 2017.

Paul C. D. Hawkins and Anthony Nicholls. Conformer generation with omega: Learning from the data set and the analysis of failures. *Journal of Chemical Information and Modeling*, 52 (11):2919–2936, 2012. doi: 10.1021/ci300314k. URL https://doi.org/10.1021/ci300314k. PMID: 23082786.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 6840–6851. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/4c5bcfec8584af0d967f1ab10179ca4b-Paper.pdf.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=6nbpPqUCIi7.

Emiel Hoogeboom, Víctor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3D. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba

Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 8867–8887. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/hoogeboom22a.html.

Ilia Igashov, Hannes Stärk, Clément Vignac, Victor Garcia Satorras, Pascal Frossard, Max Welling, Michael M. Bronstein, and Bruno E. Correia. Equivariant 3d-conditional diffusion models for molecular linker design. *ArXiv*, abs/2210.05274, 2022. URL https://arxiv.org/abs/2210.05274.

Bowen Jing, Stephan Eismann, Patricia Suriana, Raphael John Lamarre Townshend, and Ron Dror. Learning from protein structure with geometric vector perceptrons. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=1YLJDvSx6J4.

Bowen Jing, Gabriele Corso, Jeffrey Chang, Regina Barzilay, and Tommi S. Jaakkola. Torsional diffusion for molecular conformer generation. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=w6fj2r62r_H.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=k7FuTOWMOc7.

Diederik P Kingma and Ruiqi Gao. Understanding diffusion objectives as the ELBO with simple data augmentation. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023. URL https://openreview.net/forum?id=NnMEadcdyD.

Diederik P Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. On density estimation with diffusion models. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=2LdBqxc1Yv.

Jonas Köhler, Leon Klein, and Frank Noe. Equivariant flows: Exact likelihood generative learning for symmetric densities. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 5361–5370. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/kohler20a.html.

Zhifeng Kong, Wei Ping, Jiaji Huang, Kexin Zhao, and Bryan Catanzaro. Diffwave: A versatile diffusion model for audio synthesis. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=a-xFK8Ymz5J.

Tuan Le, Frank Noe, and Djork-Arné Clevert. Representation learning on biomolecular structures using equivariant graph attention. In *The First Learning on Graphs Conference*, 2022. URL https://openreview.net/forum?id=kv4xUo5Pu6.

Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori Hashimoto. Diffusion-LM improves controllable text generation. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=3s9IrEsjLyk.

Shitong Luo, Jiaqi Guan, Jianzhu Ma, and Jian Peng. A 3d generative model for structure-based drug design. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, volume 34, pp. 6229–6239. Curran Associates, Inc., 2021. URL https://proceedings.neurips.cc/paper_files/paper/2021/file/314450613369e0ee72d0da7f6fee773c-Paper.pdf.

Youzhi Luo and Shuiwang Ji. An autoregressive flow model for 3d molecular geometry generation from scratch. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=C03Ajc-NS5W.

Elman Mansimov, Omar Mahmood, Seokho Kang, and Kyunghyun Cho. Molecular geometry prediction using a deep generative graph neural network. *Scientific Reports*, 9(1), dec 2019. doi: 10.1038/s41598-019-56773-5. URL https://doi.org/10.1038%2Fs41598-019-56773-5.

Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8162–8171. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/nichol21a.html.

Xingang Peng, Shitong Luo, Jiaqi Guan, Qi Xie, Jian Peng, and Jianzhu Ma. Pocket2Mol: Efficient molecular sampling based on 3D protein pockets. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 17644–17655. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/peng22b.html.

Xingang Peng, Jiaqi Guan, Qiang Liu, and Jianzhu Ma. MolDiff: Addressing the atom-bond inconsistency problem in 3D molecule diffusion generation. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett (eds.), *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 27611–27629. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/peng23b.html.

Vadim Popov, Ivan Vovk, Vladimir Gogoryan, Tasnima Sadekova, and Mikhail Kudinov. Grad-tts: A diffusion probabilistic model for text-to-speech. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8599–8608. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/popov21a.html.

Raghunathan Ramakrishnan, Pavlo O. Dral, Pavlo O. Dral, Matthias Rupp, and O. Anatole von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific Data*, 1, 2014. URL https://api.semanticscholar.org/CorpusID:15367821.

Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10684–10695, June 2022. URL https://openaccess.thecvf.com/content/CVPR2022/html/Rombach_High-Resolution_Image_Synthesis_With_Latent_Diffusion_Models_CVPR_2022_paper.html.

Tim Salimans and Jonathan Ho. Progressive distillation for fast sampling of diffusion models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=TIdIXIpzhoI.

Victor Garcia Satorras, Emiel Hoogeboom, Fabian Bernd Fuchs, Ingmar Posner, and Max Welling. E(n) equivariant normalizing flows. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan (eds.), *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=N5hQI_RowVA.

Arne Schneuing, Yuanqi Du, Charles Harris, Arian Jamasb, Ilia Igashov, Weitao Du, Tom Blundell, Pietro Lió, Carla Gomes, Max Welling, Michael Bronstein, and Bruno Correia. Structure-based drug design with equivariant diffusion models, 2023. URL https://arxiv.org/abs/2210.13695.

Kristof Schütt, Oliver Unke, and Michael Gastegger. Equivariant message passing for the prediction of tensorial properties and molecular spectra. In Marina Meila and Tong Zhang (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 9377–9388. PMLR, 18–24 Jul 2021. URL https://proceedings.mlr.press/v139/schutt21a.html.

Gregor Simm and Jose Miguel Hernandez-Lobato. A generative model for molecular distance geometry. In Hal Daumé III and Aarti Singh (eds.), *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 8949–8958. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/simm20a.html.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In Francis Bach and David Blei (eds.), *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pp. 2256–2265, Lille, France, 07–09 Jul 2015. PMLR. URL https://proceedings.mlr.press/v37/sohl-dickstein15.html.

Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021a. URL https://openreview.net/forum?id=St1giarCHLP.

Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/file/3001ef257407d5a371a96dcd947c7d93-Paper.pdf.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021b. URL https://openreview.net/forum?id=PxTIG12RRHS.

Hannes Stärk, Octavian Ganea, Lagnajit Pattanaik, Dr.Regina Barzilay, and Tommi Jaakkola. EquiBind: Geometric deep learning for drug binding structure prediction. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (eds.), *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pp. 20503–20521. PMLR, 17–23 Jul 2022. URL https://proceedings.mlr.press/v162/stark22b.html.

Philipp Thölke and Gianni De Fabritiis. Equivariant transformers for neural network based molecular potentials. *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=zNHzqZ9wrRB.

Clément Vignac, Nagham Osman, Laura Toni, and Pascal Frossard. Midi: Mixed graph and 3d denoising diffusion for molecule generation. In Danai Koutra, Claudia Plant, Manuel Gomez Rodriguez, Elena Baralis, and Francesco Bonchi (eds.), *Machine Learning and Knowledge Discovery in Databases: Research Track - European Conference, ECML PKDD 2023, Turin, Italy, September 18-22, 2023, Proceedings, Part II*, volume 14170 of *Lecture Notes in Computer Science*, pp. 560–576. Springer, 2023. doi: 10.1007/978-3-031-43415-0_33. URL https://doi.org/10.1007/978-3-031-43415-0_33.

Lemeng Wu, Chengyue Gong, Xingchao Liu, Mao Ye, and qiang liu. Diffusion-based molecule generation with informative prior bridges. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=TJUNtiZiTKE.

Minkai Xu, Lantao Yu, Yang Song, Chence Shi, Stefano Ermon, and Jian Tang. Geodiff: A geometric diffusion model for molecular conformation generation. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=PzcvxEMzvQC.

A APPENDIX

A.1 MODEL DETAILS

Before message passing, we create a time embedding $t_e = \frac{t}{T} = \frac{t}{500}$ and concatenate those to the geometric-invariant (scalar) features, including atomic elements and charges, to pass the timestep information into the network. After each round of message passing, we employ a normalization layer for the position updates as proposed by Vignac et al. (2023), while scalar and vector features $(\mathbf{h}, \mathbf{v})$ are normalized using a Layernorm followed by an update block using gated equivariant transformation as proposed in the original EQGAT architecture (Le et al., 2022). After L round of message passing and update blocks, we leverage the last layers' embeddings to perform the final prediction $\hat{x}_0 = (\hat{\mathbf{X}}, \hat{\mathbf{H}}, \hat{\mathbf{E}})$ as shown in Figure 3. For the case that additional (geometric) invariant features are modeled, including the atomic formal charges, aromaticity, or hybridization state, the hidden node matrix $\hat{\mathbf{H}}$ includes them as output prediction by simple concatenation, i.e., predicting more output channels.

We implement EQGAT-diff using PyTorch Geometric (Fey & Lenssen, 2019) and leverage the (sparse) coordinate (COO) format that stores the molecular data and respective edge indices of the fully connected graphs.

A.1.1 MODEL TRAINING

We optimize EQGAT-diff under x_0 parameterization utilizing Gaussian diffusion for coordinates and categorical diffusion for discrete-valued data modalities, including chemical elements and bond types.

$$L_{t-1} = w_s(t)\Big(\lambda_x ||\mathbf{X}_0 - \hat{\mathbf{X}}_0||^2 + \lambda_h \text{CE}(\mathbf{H}_0, \hat{\mathbf{H}}_0) + \lambda_e \text{CE}(\mathbf{E}_0, \hat{\mathbf{E}}_0)\Big), \tag{4}$$

where CE refers to the cross-entropy loss and $(\lambda_x, \lambda_h, \lambda_e) = (3, 0.4, 2)$ are weighting coefficients adapted from Vignac et al. (2023).

In all experiments, EQGAT-diff uses 256 scalar and vector features each and 128 edge features across 12 layers of fully connected message passing. This corresponds to 12.3M trainable parameters.

We train for 200 epochs on QM9 and 400 epochs on GEOM-Drugs to achieve comparability across models while ensuring computational feasibility regarding many ablation experiments. We use fewer epochs for QM9 since the diffusion models quickly overfit such that the novelty of sampled molecules decreases. This is not the case with GEOM-Drugs.

We use the AMSGrad with a learning rate of $2 \cdot 10^{-4}$, weight-decay of $1 \cdot 10^{-12}$, and gradient clipping for values higher than ten throughout all experiments. The weights of the final model are obtained by an exponential moving average with a decay factor of 0.999.

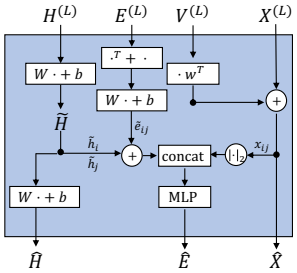


Figure 3: Prediction module that processes EQGAT-diff embeddings to obtain the predicted data modalities. The computational graph reads from top to bottom.

On the QM9 dataset, we use a batch size of 128; on the GEOM-Drugs dataset, we use an adaptive dataloader with a batch size of 800 following (Vignac et al., 2023). All models are trained on four Nvidia A100 GPUs.

For training, we use an adaptive noise schedule proposed by (Vignac et al., 2023):

$$\bar{\alpha}^t = \cos\left(\frac{\pi}{2}\frac{(t/T+s)^\nu}{1+s}\right)^2.$$

The respective scaling hyperparameter ν was set to $\nu_r = 2.5, \nu_y = 1.5, \nu_x = \nu_c = 1$ on the QM9 dataset. At the same time, for GEOM-Drugs we use $\nu_r = 2$ with ν_r, ν_x, ν_y and ν_c denoting atom coordinates, atom types, bond types, and charges, respectively. This noise scheduler accounts for the various variables of graph and 3D structure not being equally informative for the model and has been found by Vignac et al. (2023) to outperform the cosine schedule (Nichol & Dhariwal, 2021; Hoogeboom et al., 2022) significantly.

A.1.2 Model Sampling

As mentioned in Sec. 5.2, the diffusion loss term $L_t = -D_{KL}[q(x_{t-1}|x_t, x_0)|p_\theta(x_{t-1}|x_t)]$ is optimized by minimizing the KL-divergence. For the case of continuous data types, i.e., coordinates, the tractable reverse distribution (Sohl-Dickstein et al., 2015; Ho et al., 2020) is

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}|\mu_{t-1}(\mathbf{x}_t, \mathbf{x}_0), \Sigma_{t-1}), \tag{5}$$

with $\mu_{t-1}(\mathbf{x}_t, \mathbf{x}_0) = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\mathbf{x}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\alpha_t}\mathbf{x}_t$ and $\Sigma_{t-1} = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t\mathbf{I}$, where we assume that the coordinate matrix is vectorized to have shape $3N$.

Sampling from that reverse distribution is obtained through the denoising network that predicts the clean coordinate matrix to parameterize $p_\theta(\mathbf{x}_{t-1}|\mathbf{x}_t) = q(\mathbf{x}_{t-1}|\mathbf{x}_t, \hat{\mathbf{x}}_0)$ and sample via

$$\mathbf{x}_{t-1} = \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1-\bar{\alpha}_t}\hat{\mathbf{x}}_0 + \frac{\sqrt{\alpha_t}(1-\bar{\alpha}_{t-1})}{1-\alpha_t}\mathbf{x}_t + \sqrt{\frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t}\cdot\epsilon_{\mathbf{CM}}, \tag{6}$$

where $\epsilon_{\mathbf{CM}} = \epsilon - \frac{1}{3N}\sum_i^{3N}\epsilon_i$ is a Gaussian noise vector with zero mean.

For discrete variables, we obtain a tractable reverse distribution that is categorical Austin et al. (2021)

$$q(\mathbf{c}_{t-1}|\mathbf{c}_0, \mathbf{c}_t) = \mathcal{C}(\mathbf{c}_{t-1}|p_{t-1}(\mathbf{c}_0, \mathbf{c}_t)), \tag{7}$$

with probability vector defined as $p_{t-1}(\mathbf{c}_0, \mathbf{c}_t) = \frac{\mathbf{c}_t\mathbf{U}_t^\top \odot \mathbf{c}_0\bar{\mathbf{U}}_{t-1}}{\mathbf{c}_0\bar{\mathbf{U}}_t\mathbf{c}_t^\top}$ where the entry $[\mathbf{U_t}]_{ij}$ denotes the transition probability to jump from state i to j and is defined as

$$\mathbf{U}_t = (1-\beta_t)\mathbf{I}_K + \beta_t\mathbf{1}_K\tilde{\mathbf{c}}^\top = \alpha_t\mathbf{I}_K + (1-\alpha_t)\mathbf{1}_K\tilde{\mathbf{c}}^\top, \tag{8}$$

while the cumulative product after t timesteps starting from 1 can be simplified to

$$\bar{\mathbf{U}}_t = \mathbf{U}_1\mathbf{U}_2\ldots\mathbf{U}_t = \bar{\alpha}_t\mathbf{I}_K + (1-\bar{\alpha}_t)\mathbf{1}_K\tilde{\mathbf{c}}^\top. \tag{9}$$

We recall that the one-hot encoding of each node or edge is perturbed independently during the forward process, such that the encoding $\mathbf{c}_t \in \{0,1\}^K$ is obtained by sampling from the categorical distribution $q(\mathbf{c}_t|\mathbf{c}_0) = \mathcal{C}(\mathbf{c}_t|\bar{\alpha}_t\mathbf{c}_0 + (1-\bar{\alpha}_t)\tilde{\mathbf{c}})$ as described in Eq. (1).

Similar to (Austin et al., 2021; Vignac et al., 2023), we obtain the reverse process for discrete data types by marginalizing the network predictions (for each node in the graph)

$$p_\theta(\mathbf{c}_{t-1}|\mathbf{c}_t) \propto \sum_{k=1}^{K} q(\mathbf{c}_{t-1}|\mathbf{c}_t, \mathbf{e}_k)\hat{c}_{0,k}, \tag{10}$$

where $\mathbf{e}_k$ is an one-hot-encoding with 1 at index k and $\hat{c}_{0,k}$ is the k-th entry in the softmaxed probability vector $\hat{\mathbf{c}}_0$.

A.2 METRICS

The Wasserstein distance between valencies is given as a weighted sum over the valency distributions for each atom type

$$\text{ValencyW}_1 = \sum_{x \in \text{ atom types}} p(x) \mathcal{W}_1 \left(\hat{D}_{\text{val}}(x), D_{\text{val}}(x) \right), \tag{11}$$

with $p^X(x)$ being the marginal distribution of atom types in the training set and $\hat{D}_{\text{val}}(x)$ the marginal distribution of valencies for atoms of type x in the generated set and $D_{\text{val}}(x)$ the same distribution in the test set. For the bond lengths metric, a weighted sum of the distance between bond lengths for each bond type is used

$$\text{BondLenghtsW}_1 = \sum_{y \in \text{ bond types}} p(y) \mathcal{W}_1 \left(\hat{D}_{\text{dist}}(y), D_{\text{dist}}(y) \right), \tag{12}$$

where $p^Y(y)$ is the proportion of bond of types y in the training set, $\hat{D}_{\text{dist}}(y)$ is the generated distribution of bond lengths for the bond of type y, and $D_{\text{dist}}(y)$ is the same distribution computed over the test set. Lastly, the distribution of bond angles for each atom type is a weighted sum using the proportion of each atom type in the dataset, restricted to atoms with two or more neighbors, ensuring that angles can be defined

$$\text{BondAnglesW}_1(\text{generated, target}) = \sum_{x \in \text{ atom types}} \tilde{p}(x) \mathcal{W}_1 \left(\hat{D}_{\text{angles}}(x), D_{\text{angles}}(x) \right), \tag{13}$$

with $\tilde{p}^X(x)$ denoting the proportion of atoms of types x in the training set, and $D_{\text{angles}}(x)$ the distribution of geometric angles of the form $\angle(\boldsymbol{r}_k - \boldsymbol{r}_i, \boldsymbol{r}_j - \boldsymbol{r}_i)$, where i is an atom of type x, and k and j are neighbors of i (Vignac et al., 2023).

A.3 RESULTS AND DETAILS

We visualize the empirical distribution of the number of atoms and the chemical composition for the QM9, GEOM-Drugs, and PubChem3D datasets in Figure 4. For PubChem3D, we show the empirical distribution for the datasets with implicit and explicit hydrogens.

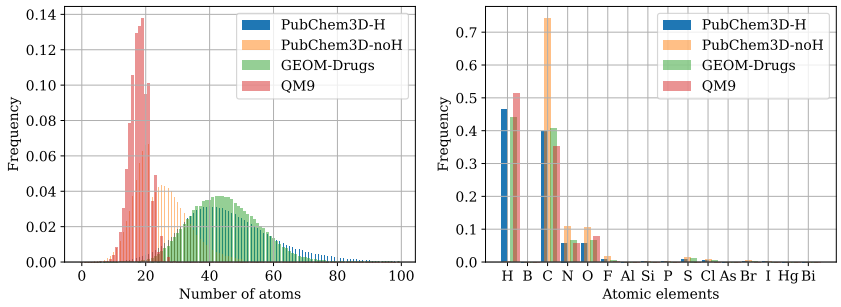


Figure 4: Empirical distributions over QM9, GEOM-Drugs, and PubChem3D with implicit and explicit hydrogens. a) Frequency for the number of atoms. b) Frequency for atomic elements.

A.4 TIME-DEPENDENT LOSS WEIGHTING

A.4.1 LOSS WEIGHTING AND FINE-TUNING

In the study in Section A.4, we conducted an ablation analysis to evaluate the efficacy of loss weighting, comparing two weighting strategies denoted as $w_s(t)$ and w_u, across different subsets (25, 50, 75, and 100%) of the QM9 and GEOM-Drugs datasets. In Fig. 5 the truncated loss weighting is

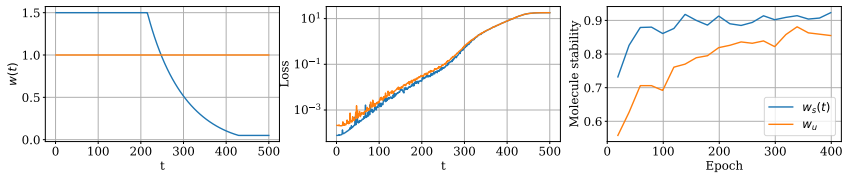


Figure 5: Comparison of EQGAT-diff trained with $w_s(t)$ and w_u, respectively, on GEOM-Drugs. a) Uniform (w_u) versus modified SNR(t) loss-weighting ($w_s(t)$). b) Unweighted prediction errors for models trained with w_u or $w_s(t)$ loss-weightings over increasing timesteps. c) Comparison between w_u and $w_s(t)$ regarding molecule stability convergence during training.

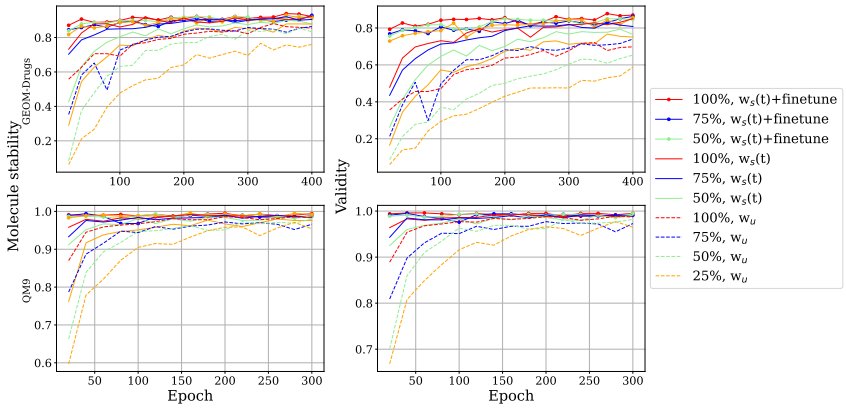


Figure 6: Comparison of different models and data subsets for training on GEOM-Drugs and QM9, respectively. The dotted, solid lines depict the fine-tuned model using $w_s(t)$-weighting. The solid lines show the model using $w_s(t)$-weighting and the dashed lines show the model trained without loss-weighting. While training, after every 20 epochs, 1000 sampled molecules are evaluated on molecule stability and validity.

depicted (left) besides the effect on the loss for lower timesteps illustrating the unweighted loss over time steps for a batch of 128 molecules, where the model trained with $w_s(t)$ achieves lower prediction error for steps closer to 1 (middle) and the effect on the molecule stability while training showing better performance and faster training convergence for molecule stability when using $w_s(t)$ (right).

As illustrated in Figure 6, applying loss weighting using $w_s(t)$ consistently results in performance enhancements for the model. These enhancements are characterized by accelerated training convergence, leading to improved molecule stability and validity, even when the model operates on smaller subsets of the data. Notably, in the case of GEOM-Drugs, when trained with only 25% of the data and optimized with $w_s(t)$ (indicated by the yellow solid line), the model exhibits convergence behavior similar to that of the model trained on 100% of the data with uniform weighting w_u (indicated by the yellow dashed line). Furthermore, fine-tuning leads to superior performance (dotted, solid lines). After just 20 epochs of fine-tuning and only using 25% of the data, the model already outperforms all its counterparts on molecule stability and validity even when they are trained on 100% of the data and holds for both the GEOM-Drugs (first row) and QM9 (second row) datasets. Our findings, as summarized in Table 6, underscore the critical role of loss weighting using $w_s(t)$ in the training of diffusion models for molecular data and also highlight the importance of pre-training, especially when the target distributions are small and do not contain many data points.

Table 6: Comparison of EQGAT-diff on QM9 and GEOM-Drugs trained on subsets of 25, 50 and 75% of the data. We report the mean values over five runs of Molecular Stability (Mol. Stability), Validity, and the number of Connected Components (Connect. Comp.) for training from scratch with and without modified SNR(t) weighting and compare it with the performance of the fine-tuned model (SNR(t)+fine-tune). The best results are written in bold, and results with overlapping margins of errors are underlined. The margin of error for the 95% confidence level is given as subscripts.

		QM9			GEOM-Drugs		
	Subset	Mol. Stability	Validity	Connect. Comp.	Mol. Stability	Validity	Connect. Comp.
w_u	25%	$96.01_{\pm 0.22}$	$96.68_{\pm 0.24}$	$99.59_{\pm 0.05}$	$74.12_{\pm 0.29}$	$51.32_{\pm 0.38}$	$68.88_{\pm 0.25}$
	50%	$96.84_{\pm 0.16}$	$97.45_{\pm 0.15}$	$99.75_{\pm 0.03}$	$85.20_{\pm 0.27}$	$64.19_{\pm 0.39}$	$82.76_{\pm 0.26}$
	75%	$96.19_{\pm 0.18}$	$96.83_{\pm 0.17}$	$99.84_{\pm 0.03}$	$87.08_{\pm 0.33}$	$74.27_{\pm 0.29}$	$88.69_{\pm 0.29}$
	100%	$97.39_{\pm 0.23}$	$97.99_{\pm 0.20}$	$99.70_{\pm 0.03}$	$87.59_{\pm 0.19}$	$71.44_{\pm 0.22}$	$86.57_{\pm 0.33}$
$w_s(t)$	25%	$97.34_{\pm 0.15}$	$97.77_{\pm 0.09}$	$99.81_{\pm 0.03}$	$88.39_{\pm 0.39}$	$75.44_{\pm 0.46}$	$85.35_{\pm 0.51}$
	50%	$98.32_{\pm 0.11}$	$98.65_{\pm 0.07}$	$99.93_{\pm 0.03}$	$89.41_{\pm 0.26}$	$77.21_{\pm 0.28}$	$89.43_{\pm 0.23}$
	75%	$98.45_{\pm 0.08}$	$98.77_{\pm 0.04}$	$99.93_{\pm 0.02}$	$91.88_{\pm 0.20}$	$82.77_{\pm 0.16}$	$93.39_{\pm 0.20}$
	100%	$98.68_{\pm 0.11}$	$98.96_{\pm 0.07}$	$99.94_{\pm 03}$	$91.66_{\pm 0.14}$	$84.02_{\pm 0.19}$	$95.08_{\pm 0.12}$
$w_s(t)$+ fine-tune	25%	$99.00_{\pm 0.13}$	$99.24_{\pm 0.10}$	$99.96_{\pm 0.01}$	$90.82_{\pm 0.67}$	$83.01_{\pm 1.30}$	$93.77_{\pm 0.76}$
	50%	$\mathbf{99.21}_{\pm 0.09}$	$\mathbf{99.41}_{\pm 0.07}$	$\mathbf{99.96}_{\pm 0.01}$	$91.24_{\pm 0.82}$	$83.83_{\pm 1.51}$	$94.66_{\pm 0.77}$
	75%	$98.79_{\pm 0.10}$	$99.12_{\pm 0.12}$	$99.95_{\pm 0.03}$	$\mathbf{92.97}_{\pm 0.15}$	$\mathbf{86.51}_{\pm 0.17}$	$\mathbf{95.92}_{\pm 0.14}$
	100%	$98.94_{\pm 0.07}$	$99.28_{\pm 0.09}$	$99.95_{\pm 0.02}$	$\mathbf{93.19}_{\pm 0.07}$	$\mathbf{86.83}_{\pm 0.20}$	$\mathbf{96.31}_{\pm 0.21}$

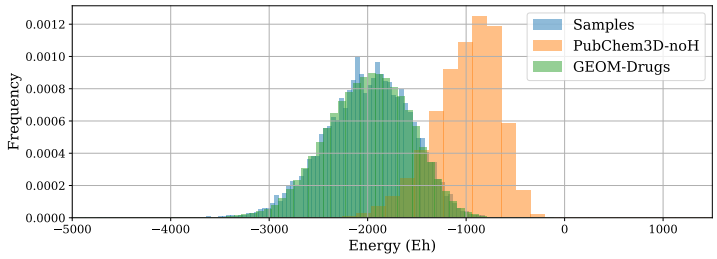


Figure 7: Comparison of the energy distributions calculated using xTB-GFN2 Bannwarth et al. (2019) for the GEOM-Drugs training dataset against the energies of sampled molecules. We also provide the energy distribution of PubChem3D (implicit hydrogens) to showcase the distribution shift; for quantum physics-based software, those molecules appear to be radicals, and hence, the energy distribution is shifted towards high energies. Nevertheless, the model effectively has to do this shift while fine-tuning.

A.5 Pre-Training on PubChem3D

To emphasize more the capability of the model to learn the underlying data distribution, we follow (Hoogeboom et al., 2022) and plot the distribution of energies for sampled molecules of a model trained on GEOM-Drugs against the energy distribution of the training dataset, as shown in Fig. 7. We observe that EQGAT-diff learns the training distribution well, showing a high overlap. Furthermore, to highlight the shift in physical space the diffusion model has to perform while fine-tuning, we also report the energy distribution of PubChem3D with implicit hydrogens. All energies were calculated using the semi-empirical xTB-GFN2 software (Bannwarth et al., 2019).

We also pre-trained a model on the PubChem3D dataset with explicit hydrogens. Interestingly, as shown in Tab. 7 we see a decrease in performance for the model that is fine-tuned on the pre-training with explicit hydrogens compared to the model using implicit hydrogens, even though pre-training with explicit hydrogens takes almost three times as long. We suspect that when using explicit hydrogens in pre-training, the model overfits too much on the PubChem3D data distribution, having a more challenging time transferring to the GEOM-Drugs distribution.

We subsampled 1M molecules from PubChem3D and GEOM-Drugs and enumerated over bonds of selected pair atoms including carbon, hydrogen, nitrogen and oxygen atoms. We computed distances

Table 7: Comparison of EQGAT-diff pre-trained with or without explicit hydrogens on PubChem3D and fine-tuned on GEOM-Drugs for 400 epochs. We report the mean values over five runs of selected evaluation metrics with the margin of error for the 95% confidence level given as subscripts. The best results are in bold.

Pretraining	Mol. Stab. ↑	Validity ↑	Connect. Comp. ↑
PubChem3D-noH	**93.19**$_{\pm 0.07}$	**86.83**$_{\pm 0.20}$	**96.31**$_{\pm 0.21}$
PubChem3D-H	92.70$_{\pm 0.09}$	85.46$_{\pm 0.19}$	94.78$_{\pm 0.19}$

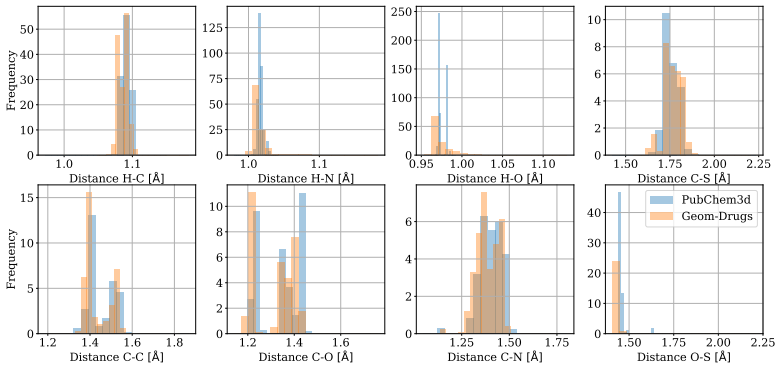


Figure 8: Selected atom-pair distance distribution on PubChem3D and GEOM-Drugs.

and noticed that the hydrogen-oxygen distance distribution in PubChem3D seems to have a smaller variance than GEOM-Drugs in the last panel of Figure 8.

A.6 EQGAT-DIFF VS MIDI

We found EQGAT-diff outperforming MiDi by a large margin across both datasets, QM9 and GEOM-Drugs, and all metrics. In Fig. 9, we underpin this observation by comparing training curves of EQGAT$^{x0}_{\text{disc}}$ and the MiDi model, observing that our model not only outperforms MiDi on molecule stability, validity and in the adaptation to the underlying data distribution, but also converges significantly faster. EQGAT$^{x0}_{\text{disc}}$ converges to SOTA performance already after 150-200 epochs, while MiDi needs roughly 700 epochs weakly indicating convergence but to lower values.

Furthermore, EQGAT-diff needs ∼5 minutes per epoch using four Nvidia A100 GPUs, adaptive dataloading (taken from the MiDi code based on `pyg.loader.Collater`) with a batch size of 200 per GPU. In contrast, MiDi takes ∼12 minutes, so EQGAT-diff is more than twice as fast.

For training MiDi, we used the official codebase on GitHub [1] and the given hyperparameter settings but trained on four Nvidia A100 GPUs (instead of two). As seen in Tab.3 and shown here in Fig. 9b, we could not reproduce the results reported in the paper. We also re-evaluated the checkpoint given on GitHub and again could not confirm the reported results.

A.7 EQGAT-DIFF WITH IMPLICIT HYDROGENS ON GEOM-DRUGS

We trained EQGAT-diff on GEOM-Drugs with implicit hydrogens. To this end, we pre-process the GEOM-Drugs dataset using the RDKit and remove hydrogens from a molecule object `mol` using the `Chem.RemoveHs` function, with subsequent kekulization `Chem.Kekulize`. We list the evaluation results of models EQGAT$^{x_0}_{disc}$ and EQGAT$^{x_0}_{cont}$ in Table 8 below. We discover that the Gaussian and categorical diffusion for the x_0 parameterization achieves similar performance

[1] https://github.com/cvignac/MiDi

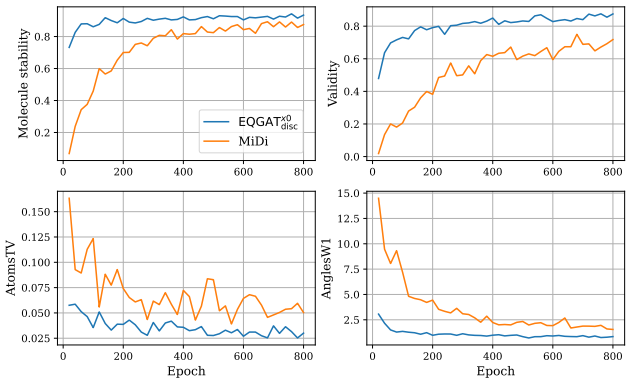


Figure 9: Comparison between EQGAT-diff and MiDi for training on GEOM-Drugs. We compare both models regarding a) molecular stability, b) validity, c) AtomsTV, and d) AnglesW1 while training by sampling 1000 molecules every 20 epochs over 800 epochs of training. For molecular stability and validity, higher is better; for AtomsTV and AnglesW1, lower is better.

related to validity and connected components. At the same time, the Wasserstein-1 distance on the histograms for empirical bond angles is lower for the generated set from EQGAT$^{x_0}_{disc}$ to the histogram of the reference set.

Table 8: Comparison of EQGAT-diff with implicit hydrogens on GEOM-Drugs for 400 epochs. We report the mean values over five runs of selected evaluation metrics with the margin of error for the 95% confidence level given as subscripts. The best results are in bold.

Model	Validity ↑	Connect. Comp. ↑	BondLengths W1 ↓	BondAngles W1 ↓
EQGAT$^{x_0}_{disc}$	**98.29**$_{\pm 0.08}$	**98.90**$_{\pm 0.10}$	**0.59**$_{\pm 0.62}$	**0.44**$_{\pm 0.01}$
EQGAT$^{x_0}_{cont}$	**98.48**$_{\pm 0.14}$	98.36$_{\pm 0.09}$	1.34$_{\pm 0.07}$	0.56$_{\pm 0.03}$

Table 9: Classifier-guidance on EQGAT-diff to shift the reverse sampling towards low or high polarizability values. We report the mean polarizability values of sampled molecules with standard deviations as subscripts.

Guidance	Polarizability
Minimization	195.19$_{\pm 4.9}$
Maximization	400.21$_{\pm 8.3}$

A.8 Classifier Guidance for Conditional Molecule Design

For conditional molecule design, we can use a trained unconditional EQGAT-diff model together with classifier-guidance, as proposed in (Dhariwal & Nichol, 2021), to steer the generation of samples using the gradient of an external classifier/regressor during the reverse sampling trajectory from noise to data. As a proof of concept, we explored classifier-guidance to generate molecules optimizing for low/high polarizability showing promising results. For this, we trained a polarizability regressor model and applied it to the reverse sampling of an unconditional EQGAT-diff model testing for guiding towards low and high polarizability values, respectively. Afterwards we re-calculate the polarizability of all sampled molecules for both cases and compared the mean values. In Tab.

9 we summarize the results. The mean value of the GEOM-Drugs training dataset is 245.9 ± 41.9. Hence, we see that we can successfully push the distribution of sampled molecules in the respective direction.

A.9 COMPARISON TO MOLDIFF

We compare against MolDiff Peng et al. (2023) by utilizing their evaluation pipeline that includes post-processing steps on the generated molecules to potentially fix valency and aromaticity issues when parsing into the RDKit. Selecting the $5 \times 10,000$ generated samples from our best performing model $\text{EQGAT}^{x0,af,ft}_{\text{disc}}$ we report mean validity and mean success rate in Table 10. As shown, our proposed best-performing EQGAT-diff model achieves superior performance over MolDiff in generating chemically valid molecules but has lower diversity, which we believe is caused by longer training time on our side. However, we believe the generative model should be able to faithfully sample molecules that satisfy valency constraints, as it was also trained in such data. Suppose we do not employ the post-processing scheme from MolDiff and determine the validity by parsing the generated molecule into RDKit's sanitization pipeline. In that case, EQGAT-diff obtains a mean validity of 0.916 and a mean success rate of 0.887. This shows that the post-processing applied in MolDiff substantially impacts model evaluation.

Table 10: Evaluation metrics from EQGAT-diff against MolDiff.

Model	EQGAT-diff	MolDiff
Validity	**0.998**	0.947
Connectivity	**0.968**	0.908
Succ. Rate	**0.966**	0.860
Novelty	1.000	1.000
Uniqueness	1.000	1.000
Diversity	0.320	**0.422**

A.10 IMPROVED SAMPLING TIME

We experimented with the DDIM Song et al. (2021a) sampling algorithm known for enhancing inference/sampling time in diffusion models trained via the standard DDPM procedure in image processing. The difference between DDIM and DDPM lies in the sampling algorithm, which we believe could also be applied in our molecular data setting. However, our best-performing scenario utilizes the x_0 parameterization to preserve the correct data modalities for coordinates, atom, and bond features. Hence, applying DDIM directly to discrete-valued data modalities is not straightforward. We restricted DDIM to continuous coordinate updates, while discrete-valued data modalities follow the approach outlined by (Austin et al., 2021) and explained in our Appendix in Eq. (10). Table 11 compares the evaluation performance of our base models when generating samples using DDIM or DDPM for varying numbers of reverse sampling steps $500, 250, 167$. Given that all models underwent training with $T = 500$ discretized timesteps, we conducted DDIM sampling every 2 or 3 steps of the reversed trajectory starting from index 500. Notably, we observed that employing DDIM did not enhance the quality of molecule generation with fewer sampling steps (250 or 166) compared to the 500 steps the models were trained on.

Another way to enhance sampling time, is to train the diffusion models with less discretized timesteps. We performed additional experiments and trained $\text{EQGAT}^{x_0}_{disc}$ with $T = 100$ timesteps using the uniform and truncated SNR(t) loss weighting. The rationale behind these experiments is to assess how the reduced number of timesteps affects performance while enabling faster inference time. We compare against the two corresponding models trained with $T = 500$ timesteps and observe that the model trained with truncated SNR(t) loss weighting over $T = 100$ timesteps performs better than the model trained with $T = 500$ timesteps but uniform loss weighting as illustrated in Figure 10. This result clearly speaks for the usage of the proposed loss weighting while also achieving a diffusion model that has a faster sampling time using 100 reverse sampling steps only, i.e. around 5x faster sampling time. Comparing the two models trained with truncated SNR(t) loss weighting, we notice that the model trained with $T = 500$ discretized steps still performs better than the identical model but trained with $T = 100$ timesteps.

Table 11: Sampling results of trained models for DDPM and DDIM for 500 Molecules.

Model	Steps	Sampling	Runtime	Mol. Stability	Validity	AnglesW1
Discrete-SNR(t)	500	DDPM	26min	0.9160	0.8100	0.83
Continuous-SNR(t)	500	DDPM	27min	0.8920	0.7600	0.90
Discrete-Uniform	500	DDPM	26min	0.8600	0.5960	1.36
Discrete-SNR(t)	250	DDIM	13min	0.6580	0.6260	2.26
Continuous-SNR(t)	250	DDIM	13min	0.5680	0.3920	3.95
Discrete-Uniform	250	DDIM	13min	0.5400	0.3880	3.21
Continuous-SNR(t)	250	DDPM	13min	0.5160	0.3400	4.74
Discrete-SNR(t)	250	DDPM	13min	0.4860	0.4600	4.60
Discrete-Uniform	250	DDPM	14min	0.2620	0.1940	7.60
Discrete-SNR(t)	166	DDIM	9min	0.1980	0.2280	5.58
Discrete-Uniform	166	DDIM	9min	0.1240	0.1080	8.24
Continuous-SNR(t)	166	DDPM	8min	0.1000	0.0380	13.68
Continuous-SNR(t)	166	DDIM	9min	0.0900	0.0520	14.11
Discrete-SNR(t)	166	DDPM	9min	0.0660	0.5200	12.21
Discrete-Uniform	166	DDPM	9min	0.0200	0.0120	14.83

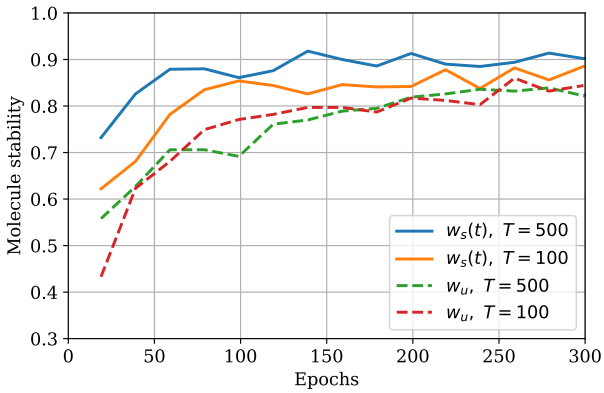


Figure 10: Molecule stability learning curves for diffusion models trained with $T = 500$ and $T = 100$ discrete timesteps. Again, we observe that the truncated SNR(t) loss weighting $w_s(t)$ greatly improves performance.

2.5 Publication 5: PILOT: Equivariant diffusion for pocket conditioned de novo ligand generation with multi-objective guidance via importance sampling

Full Reference: ***Cremer, Julian**; **Le, Tuan**; Noé, Frank; Clevert, Djork-Arné & Schütt, Kristof (2024). PILOT: Equivariant diffusion for pocket conditioned de novo ligand generation with multi-objective guidance via importance sampling. Chemical science, 15(36), 2024, 14954-14967.*

DOI: 10.1039/D4SC03523B

Journal/Conference: Chemical Science

Source Code: NA

Paper's main contributions:

- We propose a pre-training strategy for diffusion models to first learn on a vast space of synthesizable ligands and later fine-tuned on smaller datasets involving protein-ligand complexes. Through pre-training we demonstrate SOTA results in unconditional ligand generation.
- We suggest a sampling algorithm for diffusion models to perform property guided ligand generation without the use costly computed gradient through autograd
- We demonstrate that the proposed sampling algorithm is superior to gradient based guidance given the same computational budget.
- We show that that usage of an ensemble of surrogate models to perform guidance improves the sample quality by maintaining the property to be optimized, since surrogate models might not be well-calibrated.

Author's contribution to the paper:

- Conceptualization of the original idea and its application to molecular generation.
- Derivation and implementation of the method including network architecture.
- Proposal and analysis of importance sampling for diffusion guidance.
- Design and evaluation of experiments, data curation and analysis.

- Preparation and creation of initial draft and visualizations.
- Revision of manuscript during the review phase at journal.

TL and JC have shared first authorship. JC also contributed in the conceptualization, method development, proposed the usage of ensemble models, and performed experiments. The manuscript was written by TL, JC and KS. The work was supervised by FN, DAC and KS.

Acknowledgement: Reproduced from *Chemical science, 15(36), 2024, 14954-14967* with permission from the Royal Society of Chemistry.

Chemical Science

ROYAL SOCIETY OF CHEMISTRY

EDGE ARTICLE

View Article Online
View Journal | View Issue

Check for updates

Cite this: *Chem. Sci.*, 2024, **15**, 14954

All publication charges for this article have been paid for by the Royal Society of Chemistry

Received 29th May 2024
Accepted 19th August 2024

DOI: 10.1039/d4sc03523b

rsc.li/chemical-science

PILOT: equivariant diffusion for pocket-conditioned *de novo* ligand generation with multi-objective guidance *via* importance sampling†

Julian Cremer, *[ac] Tuan Le, *[ab] Frank Noé,[bd] Djork-Arné Clevert [a] and Kristof T. Schütt [a]

The generation of ligands that both are tailored to a given protein pocket and exhibit a range of desired chemical properties is a major challenge in structure-based drug design. Here, we propose an *in silico* approach for the *de novo* generation of 3D ligand structures using the equivariant diffusion model PILOT, combining pocket conditioning with a large-scale pre-training and property guidance. Its multi-objective trajectory-based importance sampling strategy is designed to direct the model towards molecules that not only exhibit desired characteristics such as increased binding affinity for a given protein pocket but also maintains high synthetic accessibility. This ensures the practicality of sampled molecules, thus maximizing their potential for the drug discovery pipeline. PILOT significantly outperforms existing methods across various metrics on the common benchmark dataset CrossDocked2020. Moreover, we employ PILOT to generate novel ligands for unseen protein pockets from the Kinodata-3D dataset, which encompasses a substantial portion of the human kinome. The generated structures exhibit predicted IC_{50} values indicative of potent biological activity, which highlights the potential of PILOT as a powerful tool for structure-based drug design.

1 Introduction

Structure-based drug discovery (SBDD) has fundamentally transformed the landscape of drug development by facilitating the design of molecules that precisely target biological macromolecules, such as proteins, which play a critical role in disease processes. These designed molecules interact with a specific pocket of a target protein, either activating or inhibiting its function, thus influencing the disease pathway. This strategy is underpinned by a detailed understanding of the 3D structure of the target, usually acquired through X-ray crystallography or nuclear magnetic resonance (NMR) spectroscopy.[1,2] By grasping the structural intricacies of the target protein, researchers are equipped to create ligands that specifically modulate its activity, offering potential therapeutic benefits.

A major challenge in SBDD is the vast chemical space that must be navigated to discover molecules with desired properties. Recently, machine learning (ML) has been applied to SBDD, promising to enable researchers to rapidly pinpoint drug candidates, significantly reducing the reliance on labor-intensive and costly experimental methods. ML algorithms are capable of analyzing vast datasets of molecular structures and properties to discern patterns, predict outcomes and generate *de novo* molecules. This might not only accelerate the drug discovery process but also enhance the efficiency and efficacy of identifying viable therapeutic agents. Early work by Ragoza *et al.*[3] used 3D convolutional neural networks (3D-CNNs) in voxel space and encoded the atomic density grids of protein–ligand complexes and the protein pockets in two separate latent spaces, both of which are used to decode 3D ligands with variational autoencoders (VAEs). A similar approach was applied in the DeepFrag architecture by Green *et al.*,[4] which focused on fragment-based ligand optimization. Wang *et al.*[5] also used 3D CNNs in voxel space on density grids, but instead of using VAEs that optimize a lower bound on the data probability, they train generative adversarial networks (GANs) end-to-end. Since voxelized grid representations are large and have sparse values (most voxels are empty), high memory consumption is a disadvantage. Treating protein–ligand complexes as atomic point clouds can circumvent this problem and, in combination with graph neural networks, enable the generative modeling of ligands bound to protein pockets. SBDD with autoregressive models that factorize the data probability were used in combination with SE(3)-invariant GNNs.[6–8] Autoregressive models for SBDD were further improved by using SE(3) equivariant networks such as in Pocket2Mol,[9] which places individual

[a]*Machine Learning & Computational Sciences, Pfizer Worldwide R&D, Berlin, Germany. E-mail: julian.cremer@pfizer.com; tuan.le@pfizer.com*
[b]*Department of Mathematics and Computer Science, Freie Universität Berlin, Germany*
[c]*Computational Science Laboratory, Universitat Pompeu Fabra, PRBB, Spain*
[d]*Microsoft Research AI4Science, Microsoft, Berlin, Germany*

† Electronic supplementary information (ESI) available. See DOI: https://doi.org/10.1039/d4sc03523b

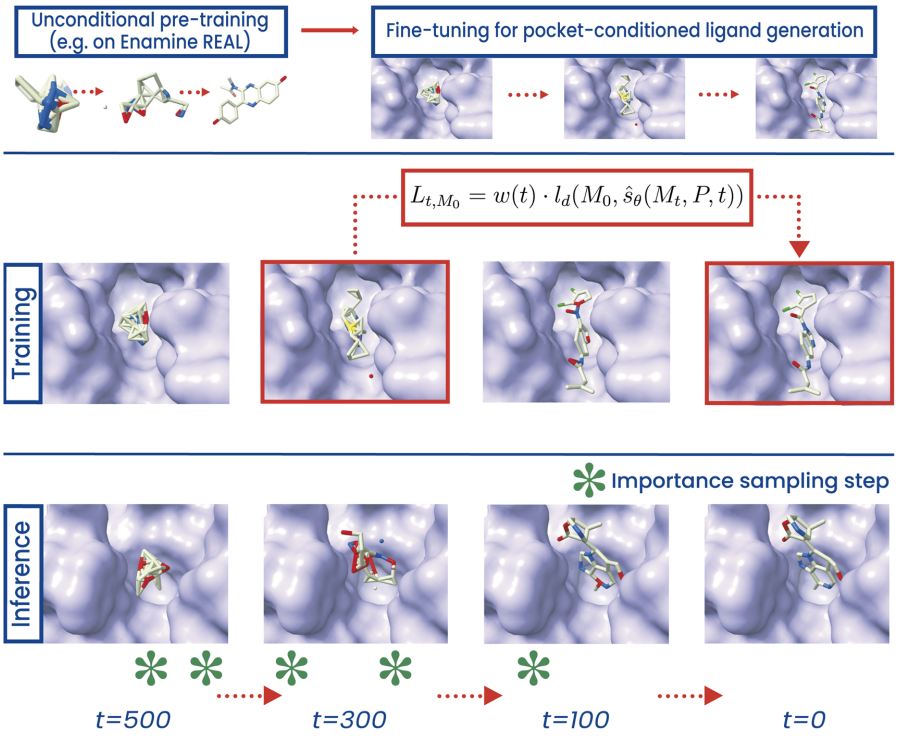


Fig. 1 Top: PILOT is first pre-trained unconditionally on an Enamine Real subset from the ZINC database.[20] We employ OpenEye's Omega to create at most five conformers per molecule.[21] Afterwards, we fine-tune the model on CrossDocked2020 conditioned on the atoms of the pocket.[22] Middle: Given the binding pocket of a protein, a noisy state of a ligand is sampled from the diffusion forward trajectory (here, $t = 300$) as input to the diffusion model during training. The model has to retrieve the ground truth ligand (M_0). For training, a composite loss (l_d) is used for continuous (mean squared error) and categorical features (cross-entropy loss), respectively, together with a timestep-dependent loss weighting ($w(t)$). Bottom: at inference, a point cloud is sampled from a Gaussian prior ($t = 500$). Given a binding pocket, the model retrieves a fitting ligand by following the reverse diffusion trajectory. At pre-specified steps, a property surrogate model (green crosses) guides the diffusion process towards desired regions in chemical space using importance sampling.

atoms one after the other during generation, or in FRAME,[10] which places fragments from a predefined library in successive steps.

Another innovative machine learning technique increasingly employed in structure-based drug discovery is the application of generative diffusion models which generate the entire structure in one-shot, but allow its refinement through successive steps. Originally utilized in fields like computer vision and natural language processing, these models also excel in capturing the complex patterns of 3D molecular structures, particularly when enhanced with features that reflect the symmetry and specific target-related characteristics of proteins.[6,9,11–13] Another line of research leverages diffusion models as methodology to build ML based docking models.[14,15]

The effectiveness of these models hinges on training with detailed protein structures, which allows for the generation of ligands that are not only structurally compatible but also specifically designed for the interaction with target proteins. However, while generated ligands fit well in a protein binding pocket, these methods lack a mechanism to guide the generative process towards ligands with desired chemical properties such as binding affinity, stability, or bioavailability. Additionally, 3D generative models often yield ligands with a high prevalence of fused rings and low synthetic accessibility.[12,13,16–18]

In this study, we introduce PILOT (Pocket-Informed Ligand OpTimization) – an equivariant diffusion model designed for *de novo* ligand generation. As shown in Fig. 1, PILOT operates in three distinct stages: unconditional diffusion pre-training, pocket-conditional fine-tuning, and property-guided inference. During the inference stage, we employ an importance sampling scheme to replace less desirable intermediate samples with more favorable ones, thereby re-weighting trajectories during generation. This strategy enables the use of any pre-trained, unconditioned diffusion score model for sampling, which is subsequently enhanced by integrating the capabilities of an external model, similar to classifier guidance.[19] However, while classifier guidance may drive the sampling trajectory to adversarial, out-of-distribution structures,[19] trajectory re-weighting

ensures that samples remain within distribution. As trajectory re-weighting can be conducted in parallel for multiple properties, we focus on three critical properties for drug discovery: synthetic accessibility (SA), docking score, and potency (IC_{50}). Our findings demonstrate that PILOT generates ligands that not only exhibit a significant improvement in synthesizability and drug-likeness but also achieve favorable docking scores and predicted inhibition.

2 Results and discussion

2.1 Pre-training of pocket-conditioned 3D diffusion models

Pre-training enables deep neural networks to build efficient representations by learning the underlying structure of the data. It proves to be a successful strategy across various fields of machine learning, particularly in the development of large language models (LLMs).[23,24] The success of these methods provides a compelling case for applying similar methodologies in the domain of scientific research, specifically in computational chemistry and drug discovery.[25–27] In the context of *de novo* molecular diffusion models, pre-training allows the models to learn fundamental chemical properties and interactions from large datasets of molecular structures. This foundational knowledge includes understanding bond types, molecular geometries, and basic physicochemical properties, which are critical for predicting how novel molecules might interact with biological targets.

Pre-training molecular diffusion models on extensive datasets of low-fidelity 3D molecule data is a beneficial strategy for enhancing *de novo* molecule generation capabilities. It significantly enhances the ability of the model to generate structurally diverse and chemically plausible molecules, when subsequently fine-tuned on smaller, high-fidelity datasets.[28] In this work, we train PILOT as illustrated in Fig. 2. For pre-training, we utilize the Enamine Real Diversity subset present in the ZINC database[20] which we downloaded from the Enamine website. To prepare the dataset, we employ OpenEye's Omega software,[21] which we use for the creation of up to five conformers per molecule with classic default parameters, resulting in a substantial corpus of approximately 109 million 3D structures. Additionally, we simplify the molecular representations by removing all hydrogens.

We study the impact of pre-training on molecules and fine-tuning on ligand–pocket complexes on model performance using the CrossDocked2020 dataset[6] following the methodologies described in the EQGAT-diff model by Le *et al.*[28] and detailed in Section 4. Table 1 shows the results with evaluation of metrics related to the generated molecular structures, such as molecular validity, the number of connected components, and the distribution of bond angles and lengths. This includes a comparison between models trained from scratch and those that have been pre-trained. The chosen distance cutoff of the pocket–ligand complex is a critical factor for model performance in terms of computational cost and accuracy (see Section 4.4). We find that pre-training improves our models across all measured metrics, and the pre-trained model with 7 Å cutoff achieves state-of-the-art performance for 8 out of the 9 evaluated metrics. In particular, over 98% of molecules sampled by the model are PoseBusters-valid (compared to 81% by TargetDiff). We measure PoseBusters-validity by summing over all non-overlapping evaluations of the "dock" and "mol" configuration in the PoseBusters tool and divide by the number of evaluations. The PoseBusters test suite validates chemical and geometric consistency of a ligand including its stereochemistry, and the physical plausibility of intra- and intermolecular measurements such as the planarity of aromatic rings, standard bond lengths, and protein–ligand clashes.[29]

The model achieves a Wasserstein distance error of 2.39 ± 0.98 for bond angles. This constitutes 4x improvement over TargetDiff, a recent SOTA baseline, which indicates a markedly

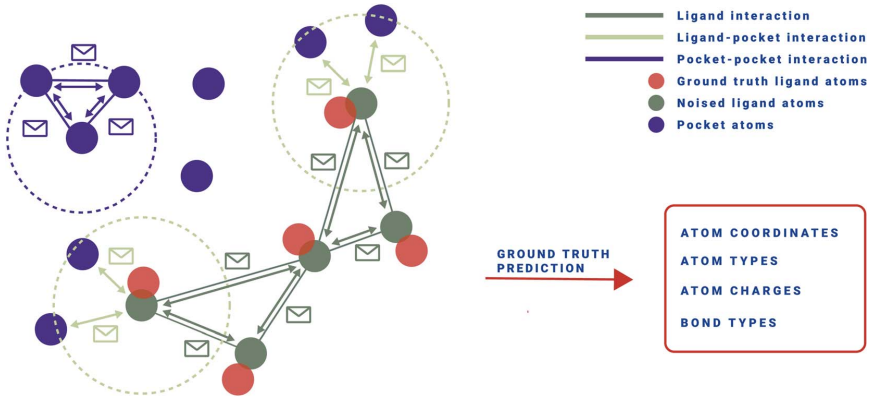


Fig. 2 Schematical depiction of the PILOT network. Given fixed pocket atoms (purple), ligand atom coordinates, types, and charges as well as the ligands' topology get noised (green) using forward diffusion. Afterwards, attention-weighted message-passing is done on the fully connected ligand atoms (here not shown for better visibility) and the ligand–pocket and pocket–pocket interactions, which each are obtained using a radius graph for computational feasibility. The task of the model is to retrieve the ground truth atom coordinates, types, charges, and the bond types (red).

Table 1 Diverse set of evaluation metrics on the CrossDocked2020 test set comprising 100 protein pockets to assess the distribution learning capability. For each protein pocket, 100 ligands are sampled. We compare metrics including novelty, bond lengths W_1, and bond angles W_1 with respect to the test set. The results are reported as mean values across all targets and ligands, with the standard deviation noted in the subscript

Model	$\text{PILOT}^{\text{scratch}}_{\text{pocket,5A}}$	$\text{PILOT}^{\text{pre-train}}_{\text{pocket,5A}}$	$\text{PILOT}^{\text{scratch}}_{\text{pocket,6A}}$	$\text{PILOT}^{\text{pre-train}}_{\text{pocket,6A}}$	$\text{PILOT}^{\text{scratch}}_{\text{pocket,7A}}$	$\text{PILOT}^{\text{pre-train}}_{\text{pocket,7A}}$	$\text{TargetDiff}_{\text{10A}}$
Validity ↑	93.40 ± 5.11	96.08 ± 3.53	93.48 ± 5.13	95.47 ± 3.91	92.06 ± 6.26	96.05 ± 3.83	78.91 ± 2.45
Pose busters-valid ↑	96.93 ± 1.91	97.39 ± 1.58	97.88 ± 1.41	97.49 ± 1.72	96.92 ± 1.91	98.21 ± 1.51	80.53 ± 1.21
Connect. comp. ↑	95.61 ± 4.15	97.44 ± 2.66	95.04 ± 5.02	97.19 ± 3.38	93.96 ± 5.99	97.81 ± 3.18	88.02 ± 2.54
Diversity ↑	72.12 ± 9.05	72.99 ± 9.01	72.03 ± 9.48	71.66 ± 9.79	70.36 ± 9.59	71.52 ± 9.84	75.12 ± 6.41
QED ↑	0.50 ± 0.12	0.51 ± 0.12	0.51 ± 0.14	0.53 ± 0.13	0.49 ± 0.14	0.53 ± 0.12	0.42 ± 0.09
SA ↑	0.67 ± 0.08	0.69 ± 0.07	0.66 ± 0.09	0.69 ± 0.07	0.66 ± 0.07	0.69 ± 0.06	0.61 ± 0.06
Lipinski ↑	4.53 ± 0.53	4.54 ± 0.49	4.54 ± 0.61	4.57 ± 0.56	4.46 ± 0.65	4.60 ± 0.51	4.64 ± 0.31
Bond angles W_1 ↓	4.03 ± 1.29	3.04 ± 1.19	3.47 ± 1.02	3.09 ± 1.06	4.00 ± 1.10	2.39 ± 0.98	9.71 ± 4.67
Bond lenghts W_1 $[10^{-2}]$ ↓	0.27 ± 0.01	0.24 ± 0.007	0.27 ± 0.09	0.23 ± 0.08	0.29 ± 0.09	0.21 ± 0.08	5.12 ± 2.05
Ligand size	23.70 ± 8.80	24.08 ± 8.83	24.56 ± 8.81	24.70 ± 8.74	24.39 ± 8.74	24.85 ± 8.94	22.21 ± 9.20

improved ability to learn the underlying data distribution. Beyond that, all PILOT models outperform TargetDiff in quantitative estimates of drug-likeness (QED) and synthetic accessibility (SA) scores, indicating that PILOT not only generates more structurally accurate molecules but also produces compounds that are more drug-like and better synthesizable.

We extend our evaluation using a range of metrics from PoseCheck[30] to assess their ability to generate ligands that form appropriate poses within the vicinity of the protein pocket. However, it is important to clarify that TargetDiff, and PILOT are not specifically designed or trained to produce exact poses, unlike tools like DiffDock,[14] which are explicitly developed and trained for docking applications. Still, *de novo* models should generate ligand poses without spatial conflicts, such as clashing with the pocket – a common issue highlighted in recent studies.[29,30] Furthermore, strain energy is a crucial metric used to evaluate ligands; it measures the energy required to alter a ligand's conformation to fit its binding pose. Those with lower strain energy are generally favorable as they are likely to exhibit stronger binding with the protein. The strain energy is calculated as the difference between the internal energy of a relaxed pose and the created pose. Both the relaxation and energy ratings are calculated using the Universal Force Field (UFF)[31] using RDKit as suggested by Harris *et al.*[30]. Fig. 3 shows that our

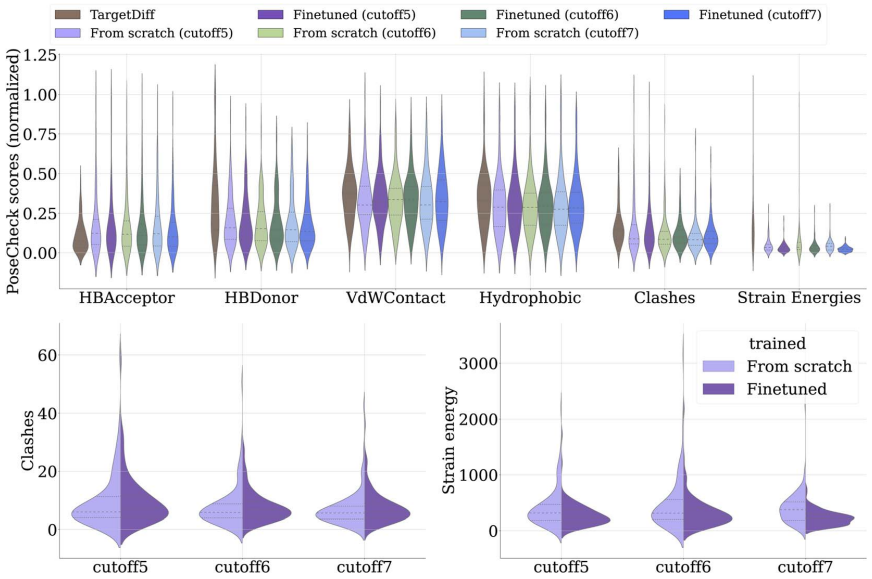


Fig. 3 The impact of varying dataset cutoffs and employing different training approaches (training from scratch *versus* pre-training) on the performance of our model and TargetDiff is analyzed. Top: we compare the sample quality using the PoseCheck metrics, where all values are min–max normalized to better evaluate the difference in performance. Bottom: we present the average clash counts (left) and average strain energies (right). Models with lower clashes and strain energies are considered to perform better and are thus preferred.

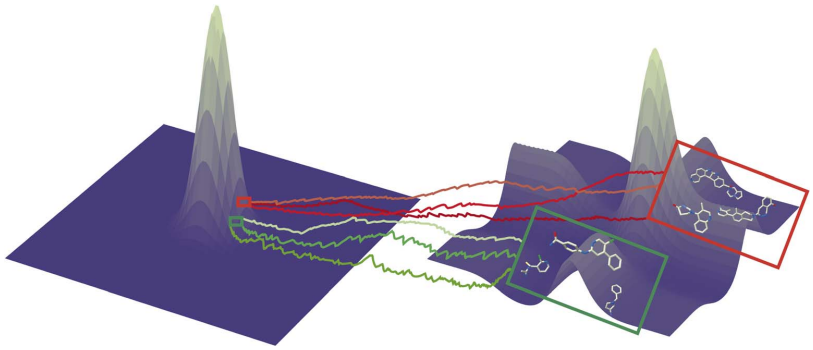


Fig. 4 Visualization of the importance sampling algorithm. The shape of the prior (left) and target (right) distribution, where ligands at the target distribution are highlighted in two different regions based on a property function, which is synthetical accessibility in this case. At $t = T$ (left), noisy samples are drawn from the prior, and during the reverse trajectory, stochastic paths that lead to promising candidates are selected and denoised in state-space to converge to samples from the data distribution at $t = 0$ (right). Ligands in the green box refer to molecules with high synthetic accessibility according to SA score, while molecules in the red box refer to rather inaccessible ones.

pre-trained model significantly excels in terms of reducing strain energy. Note that the pre-training on molecules without pocket does not lead to an increase of clashes between ligand and pocket atoms in the complex. The metrics concerning the number of hydrogen acceptors, donors, van der Waals contacts, and hydrophilicity remain consistent across models.

The reduction in strain energy observed in the pre-trained model might be attributed to two main factors. First, the diffusion model is exposed to a vast array of conformers during its pre-training phase, likely featuring low strain energy due to the conformer generation techniques employed. This results in the generation of 3D conformers with optimal torsional profiles and minimized torsional strains, contributing to overall lower energy values in the ligands produced. Second, the enamine real diversity subset used for pre-training typically includes a wide variety of stable ring systems. Thus, the model likely encounters fewer unfavorable ring systems (*e.g.* 3- or 9-membered rings), which could contribute to higher strain energies. These insights further underscore the importance of the initial pre-training phase to generate relevant and biologically active ligands, further validating the efficacy of our approach in advancing the field of structure-based drug discovery. Notice that the PILOT architecture closely resembles EQGAT-diff (see Section 4.5), and thus its superior performance over TargetDiff, *e.g.* in terms of molecular validity, arises from the application of timestep-dependent loss weighting as well as bond diffusion.[28] Higher validity comes from the correct construction of the bond graph whose atoms maintain the correct valencies, where EQGAT-diff and PILOT both have the advantage, unlike TargetDiff, of being able to directly predict the bond features. The TargetDiff architecture creates the bond graph in a post-processing step using OpenBabel with the predicted atom coordinates as input. While EQGAT-diff is pre-trained on the PubChem3D dataset, we pre-train PILOT on a subset of Enamine REAL to incorporate a larger and more diverse set of synthesizable molecules.

2.2 Multi-objective *de novo* generation using importance sampling

In previous studies utilizing 3D target-aware molecule generation, a significant challenge has been the poor synthetic accessibility (SA) of the generated molecules. These models often produce molecules with complex, fused, and uncommon ring systems, which are difficult to synthesize.[12,13,16] This issue underscores the need for approaches that not only produce molecules with strong binding affinities but also ensure that these molecules can be feasibly synthesized. To address this, we propose a trajectory-based importance sampling method that utilizes property-specific expert models explained in Section 4.

The evaluation of the importance sampling approach is performed for both single- and multi-objective optimization scenarios, focusing on SA and docking score guidance. We refer to guidance with an SA score model as SA-conditional and using a docking score model as docking-conditional. When both objectives are considered, we refer to the model as SA-docking-conditional. In each case, the unconditional base model is augmented with the respective property model during the sampling process.

Fig. 5 shows the correlation matrix of the CrossDocked2020 dataset. The SA scores exhibit a negative correlation with ligand size, *i.e.*, larger molecules tend to be less synthetically accessible on average. Conversely, the positive correlation between SA scores and QED suggests that molecules with higher QED are generally more synthetically accessible. Docking scores show a strong negative correlation with both the number of rings and the number of atoms. This implies that models driven by docking scores tend to generate larger molecules with more (fused) rings. However, such molecular characteristics typically result in decreased SA scores and QED, presenting a trade-off between optimizing for docking score and maintaining synthetic feasibility. By incorporating these insights into our modeling approach, we aim to balance the dual objectives of binding efficacy and synthetic accessibility, thereby enhancing

Table 2 Performance comparison among unconditional sampling, SA-conditional, docking-conditional, and SA-docking-conditional sampling using the CrossDocked test set, which includes 100 targets. For each target, 100 valid ligands were sampled. We assessed the performance based on several criteria: mean docking scores obtained from QVina2 re-docking, the top-10 mean docking scores per target, drug-likeness (QED), synthetic accessibility score (SA), compliance with Lipinski's Rule of Five (Lipinski), and mean diversity (Diversity) across targets and ligands

Model	QVina2 (all) ↓	QVina2 (Top-10%) ↓	QED ↑	SA ↑	Lipinski ↑	Diversity ↑
Training set	−7.57 ± 2.09	—	0.53 ± 0.20	0.75 ± 0.10	4.57 ± 0.91	—
Test set	−6.88 ± 2.33	—	0.47 ± 0.20	0.72 ± 0.13	4.34 ± 1.14	—
TargetDiff	−7.32 ± 2.47	−9.67 ± 2.55	0.48 ± 0.20	0.58 ± 0.13	4.59 ± 0.83	0.75 ± 0.09
Unconditional	−7.33 ± 2.19	−9.28 ± 2.26	0.49 ± 0.22	0.64 ± 0.13	4.40 ± 1.05	0.69 ± 0.07
SA-conditional	−7.32 ± 2.25	−8.91 ± 2.29	0.58 ± 0.19	0.77 ± 0.10	4.82 ± 0.54	0.73 ± 0.08
Docking-conditional	−9.17 ± 2.48	−10.94 ± 2.51	0.54 ± 0.13	0.62 ± 0.08	4.70 ± 0.41	0.57 ± 0.10
SA-docking-conditional	−8.35 ± 2.75	−10.36 ± 2.62	0.58 ± 0.17	0.72 ± 0.12	4.88 ± 0.44	0.68 ± 0.09
SA-docking-conditional (norm)	−7.92 ± 2.44	−9.85 ± 2.33	0.56 ± 0.19	0.78 ± 0.11	4.84 ± 0.47	0.75 ± 0.13

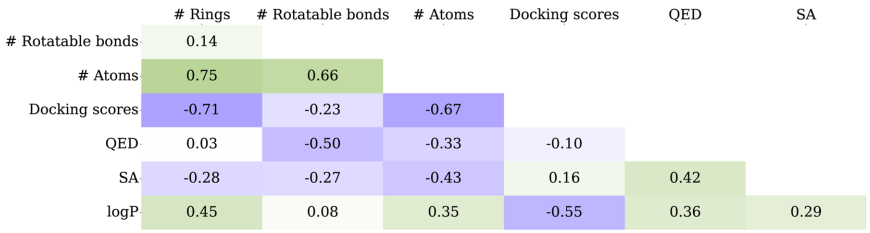


Fig. 5 Correlation matrix that includes the number of rings, number of atoms, docking scores, quantitative estimate of drug-likeness (QED), and synthetic accessibility (SA) scores using the CrossDocked2020 training set.

the practical utility of the generated molecules in drug discovery.

Table 2 shows that our model reproduces the observed correlations of the dataset. When guiding the unconditional model with the SA score, we notice a significant enhancement not only in the SA score, which increases to 0.77, but also improvements in QED and Lipinski's rule of five compliance. The mean docking scores remain consistent with those of the unconditional model. However, there is a notable reduction of docking performance in the top-10 ligands, consistent with the correlations observed in the dataset. Conversely, applying docking score guidance exclusively results in diminished SA scores and QED, while the docking scores themselves increase considerably. This reflects the trade-offs involved in optimizing for docking efficacy at the expense of synthetic accessibility and drug-likeness. When applying both SA and docking score guidance, the model achieves comparably high values for SA, QED, and Lipinski, while significantly improving docking scores and outperforming TargetDiff by a large margin.

To mitigate the adverse impact on SA scores and drug-likeness typically associated with high docking scores of larger molecules, we introduce a normalization strategy where docking scores are adjusted by the square root of the number of atoms per ligand. The results of this adjusted model, denoted as SA-docking-conditional (norm), are presented in the last row of Table 2. Here, we observe a significant increase in docking scores compared to the unconditional model, while the SA scores improve to 0.78, compared to 0.77 in the SA-conditional model. This illustrates how our multi-objective optimization strategy balances different property demands. Such balanced outcomes are critical for advancing the practical utility of generated molecules in drug discovery, ensuring that they not only bind effectively but are also feasible for synthesis.

We investigate how various molecular properties are affected by the application of guidance to further study the impact of importance sampling guidance on molecular design. Fig. 6 shows molecular characteristics such as ligand sizes, number of rings, number of rotatable bonds, and log*P* values across different models. Based on previous observations (Fig. 5), we expect SA guidance to result in smaller ligands with fewer rings, contrasting with the effect of docking guidance. First, we determine the most likely ligand size given a target from the training distribution and allow for the addition of up to ten atoms during inference. Fig. 6 (top) shows that ligands indeed tend to be smaller and possess fewer rings under SA guidance. The SA-docking-conditional model, which integrates both SA and docking objectives, represents a balanced compromise between these extremes.

Lipinski's rule of five is an important measure for assessing drug-likeness, including criteria such as the number of rotatable bonds and log*P* values. The number of rotatable bonds exhibits a strong positive correlation with the number of atoms mitigating the slight negative correlation with both SA and docking scores, while log*P* shows a positive correlation with SA- and docking scores. Fig. 6 (bottom) illustrates effective conditioning as both the SA- and docking-conditional models

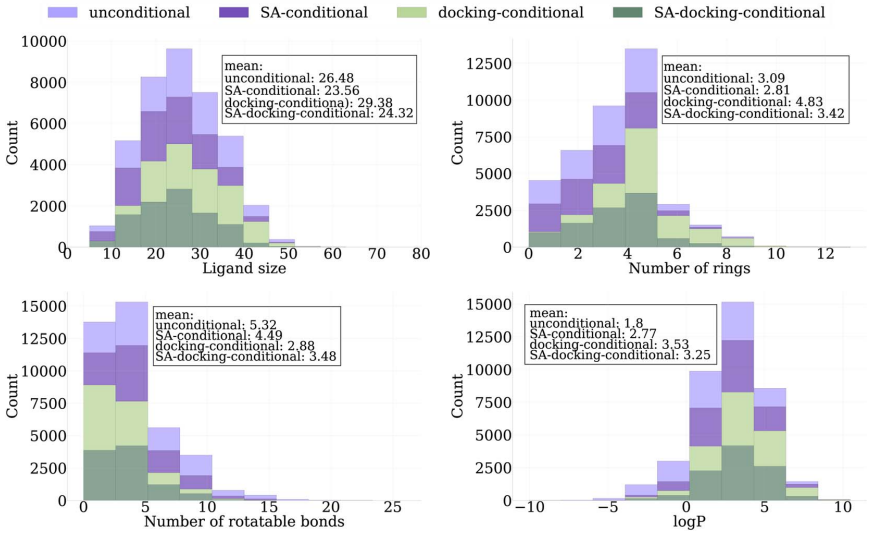


Fig. 6 Analysis of the distribution of certain ligand characteristics, including size, number of rings, number of rotatable bonds, and log*P* values, across three sampling methods to show the effect on physicochemical properties: unconditional sampling, SA-conditional, and SA-docking-conditioned sampling.

generally result in a lower average number of rotatable bonds compared to the unconditional model. In contrast, the partition coefficient logP tends to increase under both conditions.

Fig. 7 illustrates the evolution of the sample space across the unconditional, SA-conditional, docking-conditional, and SA-docking-conditional models. Each plot in this figure includes a red rectangle that identifies the regions where samples exceed the respective means of the test set, indicating improved property scores. The first row of Fig. 7 compares the drug-likeness (QED) of sampled ligands with their synthetic accessibility (SA) scores. The SA-conditional model shows a notable shift with most of the sample mass residing within the red rectangle. Thus, it successfully generates samples with notably higher SA scores compared to both the unconditional model and the test set ligands, while largely preserving docking scores. In contrast, the docking-conditional model exhibits lower

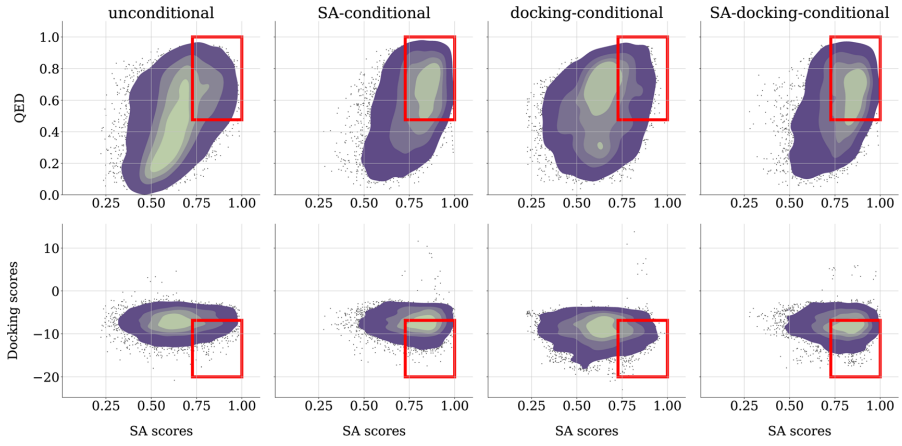


Fig. 7 Scatter plots with Gaussian kernel density estimation (KDE) were used to illustrate the evolution of QED, SA, and docking scores for all sampled ligands across test targets for different sampling methods: unconditional, SA-conditional, docking-conditional, and SA-docking-conditional sampling. Red rectangles within these plots highlight regions where sampled ligands demonstrate superior QED, SA, and docking scores compared to the test set. Upper row: relationship between QED and SA scores. Lower row relationship between docking scores and SA scores.

docking scores on average at the expense of the SA scores. The SA-docking-conditional model demonstrates a good balance, transitioning towards both high SA scores and low docking scores. Remarkably, most of the sampled ligands from this model not only fall within the red rectangle but also significantly surpass the test set ligands in terms of docking scores with equal SA scores as listed in Table 2, while the model with normalization improves in both metrics.

We compare our importance sampling approach against the classifier guidance method[19] using the fine-tuned model trained on the CrossDocked dataset with 5 Å cutoff. For classifier guidance, we calculate the gradients with respect to atomic coordinates by using the autograd engine for the outputs of the surrogate models during the reverse sampling trajectory. Guided by maximizing SA and minimizing docking scores, we find that the mean run time per protein pocket in the test set using classifier guidance is approximately 4.3 times slower than importance sampling, largely due to the gradient calculations (see Section C.1 in the ESI†). Notice that for classifier guidance the batch size has to be reduced in order to avoid out of memory issues which are caused by the autograd engine. The importance sampling approach does not require gradients and enables practitioners to maintain a significantly larger batch-size. The molecular validity for our proposed importance sampling method is also significantly higher at 93.40% compared to classifier guidance, which achieves a validity of 77.17% for 10 000 generated ligands. This shows that sampling a given set of valid molecules takes even longer, as classifier guidance results in a significant increase in adversarial structures. Nevertheless, we find that the mean SA and docking scores of 0.82 and −8.43, respectively, are better than those for importance sampling (0.75 and −7.7). However, if we perform classifier guidance with the same number of update steps as the importance sampling, the validity increases to 93.18% similarly to importance sampling, but the SA and docking scores are significantly less optimized, reaching 0.72 and −7.15, respectively. Additionally, we measure the effect of importance sampling and classifier guidance on the uniqueness rate (number of unique molecules per 100 sampled ligands). The unconditional model achieves a uniqueness rate of 0.83, which diminishes slightly to 0.75 when using importance sampling and more significantly to 0.65 using classifier guidance.

Overall, our findings demonstrate that using importance sampling as a guidance mechanism in the diffusion model is a potent strategy for steering the generation of molecules towards desired regions of chemical space while being computationally several times cheaper compared to classifier guidance. The method effectively modifies molecular properties in line with desired multi-objective property profiles, albeit within the constraints set by the data distribution used for training. Unlike classifier-guidance, our approach does not require (prohibitively) expensive backpropagation. Instead, we achieve the aforementioned results using only a few importance sampling steps (forward calls to the surrogate models).

2.2.1 Kinodata-3D. We leverage the Kinodata-3D dataset, annotated with experimental pIC_{50} values, to train PILOT on ligand–kinase complexes. Simultaneously, we train a property model predicting pIC_{50}, to guide the diffusion model with the proposed importance sampling towards ligands that are more likely to be potent inhibitors. All models are trained from scratch because the pre-trained Enamine model does not contain all atom types present in the Kinodata-3D dataset. We leave the evaluation with a pre-trained model to future work.

We evaluate the models on a hold-out test set comprising ten kinase targets that were not included in either the training or validation datasets. The performance of our pIC_{50}-conditional model is summarized in Table 3. The pIC_{50}-conditional model shows a significant improvement in predicted mean pIC_{50} values of 7.65 ± 0.78 compared to the test set ligands (6.41 ± 1.56). At the same time, it maintains robust performance metrics in terms of docking scores and other critical properties such as QED and compliance with Lipinski's rule of five.

Fig. 8 provides a visual comparison of the sample spaces generated by the unconditional and the pIC_{50}-conditional model. We observe a significant shift in the overall density of samples towards higher predicted pIC_{50} when using the importance sampling guidance (left panel). Fig. 8 (right) illustrates the relationship between docking scores and pIC_{50}. While the pIC_{50}-conditional model yields samples with higher pIC_{50} on average, the ligands maintain competitive docking scores. This suggests that the model does not compromise docking efficacy for higher expected pIC_{50}.

Note, that the current approach is limited as pIC_{50} values are inherently noisy, in particular when collected across various data sources.[32] Thus, the predicted binding affinities should be interpreted cautiously. To alleviate this problem, we propose to adopt ensemble modeling techniques to enhance the meaningfulness of predictions in the importance sampling pipeline. Similar approaches are, for example, commonly used for stabilizing machine learning force fields.[33]

Fig. 9 (top) demonstrates how ensemble techniques significantly improve the robustness of pIC_{50} predictions. We employ an

Table 3 Performance comparison among unconditional and pIC_{50}-conditional sampling using the Kinodata-3D test set, which includes 10 targets. For each target, 100 ligands were sampled. We assessed the performance based on several criteria: mean docking scores obtained from QVina2 re-docking, the top-10 mean docking scores per target, (predicted) pIC_{50}, drug-likeness (QED), synthetic accessibility score (SA), compliance with Lipinski's Rule of Five (Lipinski), and mean diversity (Diversity) across targets and ligands

Model	Vina (all) ↓	Vina (top-10%) ↓	pIC50 ↑	QED ↑	SA ↑	Lipinski ↑	Diversity ↑
Training set	-9.20 ± 1.13	—	7.05 ± 1.28	0.49 ± 0.16	0.75 ± 0.07	4.73 ± 0.52	—
Test set	-8.78 ± 1.13	—	6.41 ± 1.56	0.61 ± 0.14	0.79 ± 0.05	4.96 ± 0.22	—
Unconditional	-8.49 ± 1.05	-9.79 ± 0.87	6.28 ± 0.68	0.63 ± 0.14	0.75 ± 0.13	4.95 ± 0.25	0.65 ± 0.06
pIC_{50}-conditional	-8.60 ± 0.98	-9.75 ± 0.86	7.65 ± 0.78	0.62 ± 0.16	0.67 ± 0.09	4.94 ± 0.28	0.57 ± 0.06

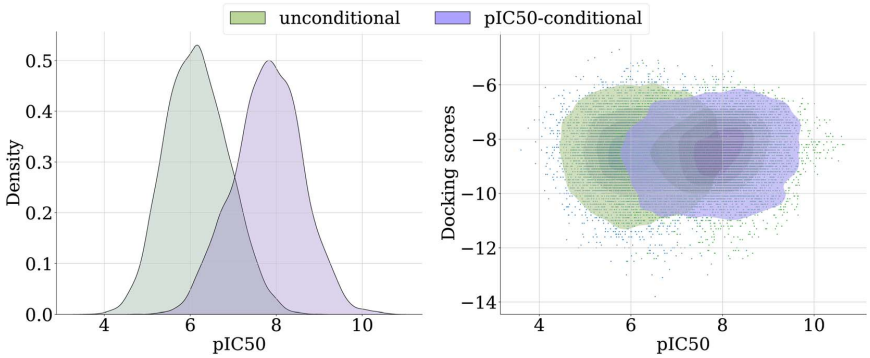


Fig. 8 Left: density plot comparing unconditional with pIC_{50}-conditional sampling. Right: scatter heatmap overlap of unconditional and pIC_{50}-conditional samples comparing docking scores and (predicted) pIC_{50} values.

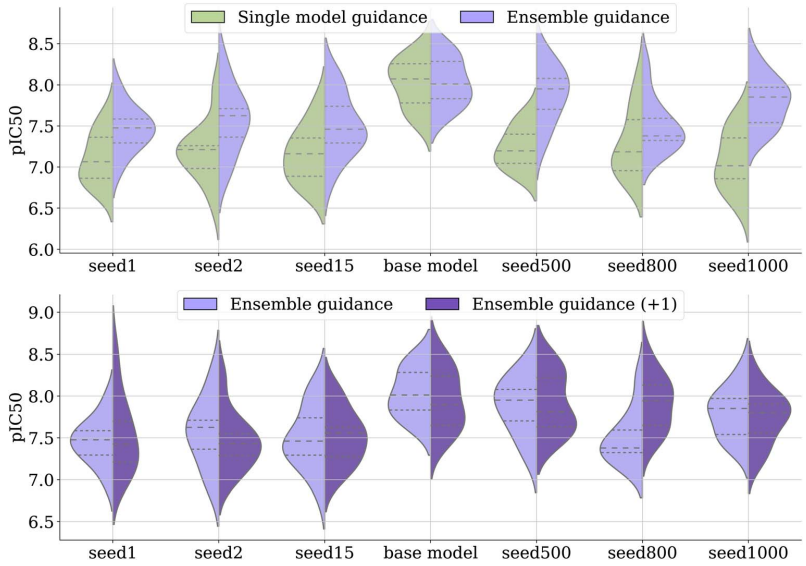


Fig. 9 A violin plot is used to display the distribution of predicted pIC_{50} values for 100 sampled ligands across ten test set targets, guided either by a single model or an ensemble approach. Upper panel: ligands generated under single model guidance, where the base model guides itself, or ensemble guidance that includes seed models 500 and 1000. All other models are utilized for evaluating the respective samples. Lower panel: here, the ensemble guidance for the base model is extended by incorporating an additional model, specifically seed800. This is referred to as "Ensemble guidance (+1)".

ensemble of property models for importance sampling guidance. Each property model, denoted as seed1, seed2, *etc.*, is trained with a different global seed. The base model is used to sample 100 ligands per test target, both with and without ensemble guidance. The term single model guidance refers to the base model guiding itself. We observe that single model guidance results in a notable offset between the predictions of the base model and those of all other property models, indicating poor generalization performance. That is, self-guidance exploits the predicted pIC_{50} value too much, as it was trained on. However, with ensemble guidance, even just two additional seed models (seed500 and seed1000) lead to greater improvement in generality. This enhancement is evident in the pIC_{50} predictions of all seed models not included in the ensemble guidance (*i.e.*, seed1, seed2, seed15, and seed800). As shown in Fig. 9 (bottom), further increasing the ensemble size, such as by adding another model, here seed800, leads to additional refinement in predictions and consequently, increased generality of pIC_{50} predictions.

We observe improved generalization performance for the ensemble compared to the single models. We evaluate the five

models on the Kinodata-3D test set, which achieve an average mean squared error of 1.34. In contrast, the ensemble built from these five models achieves a lower error of 1.23.

3 Conclusions

We have introduced PILOT, a novel equivariant diffusion-based model tailored for *de novo* ligand generation conditioned on protein pockets in three-dimensional space. Our research demonstrates the superior performance of PILOT compared to existing state-of-the-art models in this domain, as evidenced by a comprehensive evaluation across a spectrum of metrics critical in medicinal chemistry and drug design.

A significant finding of our study is the substantial enhancement in downstream performance achieved by pre-training our model on a vast dataset of molecular conformers. This underscores the pivotal role of pre-training in the structure-based drug discovery pipeline, demonstrating its efficacy in improving the quality of generated ligands. Beyond that, we have proposed a trajectory-based importance sampling strategy, which enables targeted steering of ligand generation towards desired chemical properties. This technique guides the generation process towards ligands with desired properties such as synthetic accessibility, drug-likeness, docking scores, and predicted binding affinities by using surrogate models trained on experimental data. This strategy represents an important advancement in structure-based drug discovery, offering researchers a powerful tool to design molecules with tailored properties using 3D equivariant diffusion models.

The dependency on the availability and quality of training data remains a critical challenge for deploying AI models like PILOT in drug discovery pipelines. In the domain of structure-based drug design, data can often be sparse, noisy, and of varying quality, which significantly impacts the learning and predictive capabilities of ML models. While our method heavily relies on surrogate models and proxies such as the RDKit synthetic accessibility (SA) scores to estimate the synthesizability of generated ligands, these scores may not fully capture the complexities and practical challenges of medicinal chemistry. Addressing these challenges will require a concerted effort to enhance data collection practices, improve data quality, and expand the variety of data sources.

Moving forward, we see potential applications of PILOT in the drug discovery pipeline by integrating this model with other AI-driven tools and technologies, such as automated synthesis platforms and high-throughput screening to accelerate drug design. Furthermore, the scope of our model may be extended from small molecule drugs to biologic therapeutics involving for example peptides or antibodies.

4 Methods

4.1 Pocket conditioned 3D diffusion models

We aim to generate novel molecules M *de novo*, conditioned on a protein pocket P while optimizing multiple objectives c, such as synthetic accessibility, docking score, and predicted half-maximal inhibitory concentration (IC_{50}). Recent developments have utilized 3D diffusion models to implement $p_\theta(M|P)$, where the task of the model is to denoise an initially random ligand structure, while maintaining the protein pocket as a fixed condition.[12,13,28] This is achieved by following a stochastic path that targets the distribution of training data, iteratively moving towards more defined structures $p_\theta(M_{t-1}|M_t, P)$ as illustrated in Fig. 1.

During training, the reverse distribution $p_\theta(M_{t-1}|M_t, P)$ is parameterized using the approach as proposed by Le *et al.*[28]. That is, a noisy ligand $M_t = (X_t, H_t, E_t)$ at time step t is represented by perturbed atomic coordinates X_t, element types H_t, and bond features E_t, while the diffusion model p_θ is tasked in predicting the noise-free structure $\hat{M}_0=(\hat{X}_0,\hat{H}_0,\hat{E}_0)$, acting as denoiser with the inherent goal to iteratively attain a cleaner structure. We optimize the variational lower bound of the log-likelihood $\log p(M_0|P)$ and minimize the timestep-dependent diffusion loss

$$L_t = \frac{1}{2}(w(t) \times l_d(M_0, p_\theta(M_t, t, P)) \quad (1)$$

where $l_d : \mathcal{M} \times \mathcal{M} \to \mathbb{R}^+$ reveals as mean-squared-error loss for 3D coordinates, and cross-entropy loss for discrete-valued data types like atom, bond, and charge-types.[28] To obtain the noisy ligand M_t, we apply the forward noising process with Gaussian diffusion for continuous valued coordinates, while discrete valued data like atom, bond- and charge-types are perturbed using categorical diffusion which both reads

$$q(X_t|X_0) = \mathcal{N}\left(X_t \middle| \sqrt{\overline{\alpha}_t}X_0, (1 - \overline{\alpha}_t)I\right) \quad (2)$$

$$q(C_t|C_0) = \mathcal{C}\left(C_t|\overline{\alpha}_t C_0 + (1 - \overline{\alpha}_t)\tilde{C}\right), \quad (3)$$

where $\overline{\alpha}_t = \prod_{k=1}^{t}(1 - \beta_k) \in (0, 1)$ determines a variance-preserving (VP) adaptive noise scheduler with empirical distribution $\tilde{C}$ estimated from the training set for categorical data (H, E).[34]

4.2 Multi-objective importance sampling

To sample ligands from the distribution $p_\theta(M|P, c)$, we utilize Bayes' theorem to decompose the probability density into $p_\theta(M|P, c) \propto p_\delta(c|M, P)p_\theta(M|P)$. We further assume that multiple properties $c = (c_1, c_2, \ldots, c_k)$ are conditionally independent, leading to the factorization $p_\delta(c|M, P) = \prod_{l=1}^{k} p_{\delta_l}(c_l|M, P)$, where each $p_{\delta_l}(c_l|M, P)$ can be interpreted as an expert surrogate model for a specific property. These surrogate models must be able to predict the properties of interest at any step of the diffusion trajectory, similar to classifier-guidance.[19] While classifier-guidance requires backpropagation at every step, making it quickly unfeasible for ligand–pocket complexes with several hundred atoms, our proposed importance sampling approach eliminates the need for backpropagation. Moreover, far fewer steps are needed to update the diffusion model compared to classifier-guidance, which also often tends to steer the model towards adversarial structures.[19]

Algorithm 1 Importance sampling for property-guided ligand generation, here maximization of c

Input: Pocket P, condition c, number of ligands K, τ temperature, every importance step N, diffusion model p_θ and property models p_δ.
Output: Set of generated ligands $\{M_i\}_{i=1}^K$ conditioned on (P, c).

Sample K ligands from prior distribution $M_T \sim N(0, I) \times C(\hat{p}_c)$
for $t = T-1, \ldots, 1$ **do** ▷ Run reverse diffusion trajectory
  Sample $M_{t-1} \sim p_\theta(M_{t-1}|M_t, P)$
  if $t \mod N = 0$ **then** ▷ Importance step
    for $k = 1, \ldots, K$ **do**
      if optimize for specific c **then**
        $\widetilde{w}_k = p_\delta(c|M_{k,t-1}, P)$ ▷ Compute probability value
      else
        $\widetilde{w}_k = f_\delta(M_{k,t-1}, P)$ ▷ Compute raw property value, here maximization
      end if
    end for
    Importance weight computation based on population using softmax with temperature τ:
    $\{(M_{k,t-1}, \widetilde{w}_k)\}_{k=1}^K$: $w_k = \frac{\exp(\widetilde{w}_k/\tau)}{\sum_{j=1}^K \exp(\widetilde{w}_j/\tau)}$
    Draw new population with replacement:
    $\{M_{k,t-1}\}_{k=1}^K \sim \text{Multinomial}(\{M_{k,t-1}\}_{k=1}^K, \{w_k\}_{k=1}^K)$
  end if
end for
return $\{M_{k,0}\}_{k=1}^K$

As properties such as synthetic accessibility are determined solely based on the ligand, whereas others, like docking scores, depend on the interaction between the ligand and the protein pocket, suitable property predictors p_{δ_i} may be defined as required. During the sampling process of a set of K noisy ligands $\{M_1, M_2, \ldots, M_K\}$, we use importance weights derived from $p_\delta(c|M, P)$ to rank each intermediate noisy sample at its current position in the state space, as described in Algorithm 1. Our goal is to generate samples from $p_\theta(M|c, P) \propto p_\delta(c|M, P)\, p_\theta(M|P)$ under the condition c, which specifies the property that the ligand M must achieve. For continuous properties, we choose a Gaussian distribution with a standard deviation of 1 to model $p(c|M, P)$. Specifically, this takes the form $p_\delta(c|M,P) = \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}(c - f_\delta(M,P))^2\right)$. This formulation also establishes a natural connection to maximum-likelihood training for the property predictor f_δ. Since the reverse diffusion trajectory is inherently stochastic, our goal is to preferentially select samples that are most likely to follow a path resulting in ligands meeting the specified conditions c. This process is schematically depicted in Fig. 4. To accurately predict these conditions, we train $p_\delta(c|M, P)$ as $p_\delta(c|M_t, P, t)$ along the forward noising diffusion trajectory, where M_t represents the state of the ligand at time step t. The property model p_δ is trained using the mean squared error and cross-entropy loss for continuous and discrete properties, respectively. The rationale behind this training approach is that denoising steps closer to the original data distribution retain a clearer signal of the input ligand, making them highly informative. In contrast, steps closer to the prior noise distribution, although less informative, can still provide valuable discriminative insights for p_δ. This strategy leverages the nuanced progression of information degradation during the diffusion process to efficiently guide the generation of desired ligands without mode collapse.

The algorithm is inspired by the Sequential Monte Carlo (SMC) method.[35,36] A similar replacement strategy has previously been applied by Trippe *et al.*[37] and Wu *et al.*[38] in the context of diffusion models for protein backbone modeling and motif scaffolding. In Algorithm 1, we focus on maximizing property values by scoring each predicted property value among the samples in the population. To achieve this, we employ softmax normalization on the predicted property values $f_\delta(M_k, P)$ for maximization. If the goal is to minimize a certain property, the predicted property values must be multiplied by -1 to compute the importance weights before applying the softmax operation. These importance weights represent the probability of selecting samples from the finite population set for the next iteration. When specific property values c are desired, instead of relying solely on the predicted property values $c_k = f_\delta(M_k, P)$, we compute the probability using a Gaussian kernel as described earlier. Notice that we additionally need to employ another normalization scheme to rank each unique probability value. For simplicity, we choose to use softmax normalization again. On CrossDocked, we employ the importance sampling every $N = 10$ steps and first filter for trajectories with highly synthetic accessible samples in timesteps 100–250, while ligands with better docking scores are weighted in steps 300–400 during the reverse trajectory which involves 500 steps. Both importance filtering steps are applied with temperature $\tau = 0.1$. We refer to the ESI section C† for more details.

4.2.1 Classifier guidance. We leverage the SA- and docking score predictions of f_{δ_1,δ_2} to compute gradients with respect to atomic coordinates that describe the direction to maximize/minimize the corresponding properties. Given a single molecule with n atoms, classifier guidance for SA and docking score optimization is described *via* the coordinate update equations

$$\begin{aligned} \tilde{X}_{t-1} &\sim p_\theta(X_{t-1}|M_t, P) \\ X_{t-1} &= \tilde{X}_{t-1} + \lambda_1 \nabla_{X_t} f_{\delta_1}(M_t, P) - \lambda_2 \nabla_{X_t} f_{\delta_2}(M_t, P), \end{aligned} \tag{4}$$

where $X_{t-1} \in \mathbb{R}^{n\times 3}$ and the first equation samples the atomic coordinates with respect to the current noisy molecule and protein pocket disregarding the SA and docking scores property. The second equation applies the gradient guidance with scales $\lambda_1, \lambda_2 > 0$ to maximize SA and minimize docking scores. In our experiments we set $\lambda_1 = \lambda_2 = 0.1$.

4.3 Datasets

4.3.1 Enamine. We use the Enamine REAL drug-like Diversity subset comprising 48.2 M compounds represented as SMILES string representations. To process the 3D dataset, we attempt to generate up to five conformers per SMILES string using OpenEye Omega (version 2022.1.2) classic with default parameters without hydrogens.

4.3.2 CrossDocked. We use the CrossDocked2020 dataset introduced in Francoeur *et al.*[39] and follow the same filtering and splitting strategies as in previous works, which utilized a protein sequence identity splitting.[6,9] This results in

approximately 100 000 protein–ligand complexes for the training set and 100 for the test set.

4.3.3 Kinodata-3D. We use the Kinodata-3D dataset,[40] a collection of kinase complexes curated and processed *in silico* using cross-docking data. To facilitate training of machine learning models for structural protein–ligand complexes, associated experimental binding affinity labels are included. The dataset builds on the cross-docking benchmark established by Schaller *et al.*,[41] adopting a template-based approach. For more details we refer to Backenköhler *et al.*[40]. We use approximately 105 000 pocket-ligand complexes for training, and save 310 and 136 complexes for validation and testing, respectively.

4.4 Choosing the cutoff for protein–ligand complex creation

The CrossDocked2020 dataset implements a pre-defined cutoff surrounding the bound ligands to cut out the protein pockets. Based on this, TargetDiff[12] uses a cutoff region of 10 Å with the centers of mass (CoM) of the residues acting as reference points for measuring distances to ligand atoms. Residues whose CoM are within or equal to the cutoff distance are included in the Protein–Ligand (PL) complex. Conversely, DiffSBDD includes the entire residue in the PL complex if any atom within that residue falls inside the cutoff region.[13] Our work adopts the latter approach as it offers a more physically plausible representation of the interaction space. We ablate different cutoff values {5,6,7}Å on the CrossDocked2020 dataset and observe that the model trained on the 7 Å cutoff performs best as illustrated in Table 1 for the pre-trained model. We hypothesize that the trade-off between smaller cutoff and model performance is caused by the complexity and tendency to overfit on smaller complexes. Note that a smaller cutoff leads to PL complexes with fewer atoms as shown in Figure and Table B1 in the ESI.†

4.5 Model architecture

The PILOT architecture is an extension of EQGAT-diff[28] with minor changes to handle protein–ligand (PL) complexes. To perform message-passing, we calculate the interactions in the protein–ligand and protein–protein graphs using a radius graph with a cutoff of 5 Å. We perform fully-connected message-passing for all ligand–ligand interactions. Unlike the EQGAT-diff architecture, we also incorporate a residual connection of the transformed initial ligand–ligand edge encodings into the PILOT architecture. Assuming that the small molecule consists of n atoms, the initial one-hot encoded edge features $E \in \mathbb{R}^{n\times n\times 5}$ categorize the presence of none, single, double, triple and aromatic bonds. We further calculate the distance matrix of $D \in \mathbb{R}^{n\times n}$ and compute an initial edge-feature between atom i and j as $e^*_{ij} = e_{ij} \otimes g_{\mathrm{rbf}}(d_{ij}) \in \mathbb{R}^{5\times 20}$, which computes an outer product between the one-hot encoding of the bond feature with an exponential radial basis function with 20 channels. The embedding e^*_{ij} is vectorized into shape $\mathbb{R}^{100}$ and linearly transformed to obtain the hidden edge embedding $e^{(0)}_{ij} \in \mathbb{R}^{128}$ prior to the message passing. After $L = 12$ rounds of message-passing, we use separate prediction heads for predicting coordinates, atoms, charges, and bond types, as suggested in the initial EQGAT-diff architecture. We use 256 scalar and vector channels and 128 edge channels across the network. We observe improved model performance when including the initial embedding of edge features through a residual connection after each message-passing layer. We hypothesize that this information enables better 3D coordinates as well as bond predictions by the diffusion model because the dependency between bonds and atomic coordinates is included in each message-passing layer.

4.6 Training details

We train PILOT with $T = 500$ diffusion timesteps (in contrast to TargetDiff, which uses $T = 1000$). For training, we draw a random batch of protein–ligand complexes and uniformly sample timesteps $t \in U(1, 500)$. The diffusion loss L_t is optimized for each sample, which under the data prediction parameterization means a mean-squared-error (MSE) loss for atomic coordinates and cross-entropy (CE) loss for discrete-valued modalities including atom- and bond types of the ligand molecule. For more details we refer to Le *et al.*[28]. The Enamine model was pre-trained for 10 epochs with the goal of learning a broad chemical space of molecules not limited to pocket–ligand complex data. We trained the models for 300 epochs from scratch on the CrossDocked2020 and Kinodata 3D dataset. When leveraging the pre-trained Enamine model as a starting point, we only fine-tuned for 100 epochs on the CrossDocked2020 dataset. In all (pre-)trainings we use the AdamW optimizer with AMSGrad and a learning rate of 2×10^{-4}, weight-decay of 1×10^{-12}, and gradient clipping for values higher than 10 throughout all experiments.

4.6.1 Property training. In this work, we utilize a joint training strategy for both the diffusion and property models within a single neural network architecture. Since both models take a noisy ligand $M_t = (X_t, H_t, E_t)$ as input, the joint model predicts both the clean molecule and the ground-truth property of the input sample, such as synthetic accessibility and/or docking score ($\hat{M}_0$ and $\hat{c}$, respectively). This is achieved by including additional prediction heads, *e.g.* MLPs, operating on the node/edge embeddings of the final message-passing layer. As the synthetic accessibility (SA) score only depends on the ligand we also pre-train the Enamine model to jointly predict the SA score in addition to the denoising task. When fine-tuning on CrossDocked, we load the model weights from Enamine and add an extra head for docking score prediction. However, importance sampling can be performed using any external model trained on a diffusion trajectory, as long as it uses the same transition kernels as the diffusion model. In preliminary studies, we experimented with separately trained models and found that they also worked. However, for simplicity, we used joint training in this work. We adapt the timestep dependent loss weighting as in Le *et al.*,[28] such that the gradient signal for larger timesteps is damped, following the property loss

$$L_{\mathrm{p,t}} = w(t)\|c_0 - p_{\theta,\delta}(M_t, t, P)\|^2. \quad (5)$$

Data availability

The processed CrossDocked2020 dataset can be downloaded from **https://github.com/pengxingang/Pocket2Mol/tree/main/data**. The Kinodata-3D dataset can be downloaded from **https://volkamerlab.org/projects/kinodata-3d**. The Enamine REAL data used in this study for pre-training the model is licensed and thus cannot be made available. However, we provide all information needed to reproduce the dataset.

Author contributions

Conceptualization: J. C., T. L.; data curation: J. C., T. L.; formal analysis: J. C., T. L.; investigation: J. C., T. L.; methodology: J. C., T. L.; resources: D. A. C.; supervision: D. A. C., K. T. S.; visualization: J. C., T. L.; writing — original draft: J. C., T. L., K. T. S.; proofreading: J. C., T. L., F. N., D. A. C., K. T. S.; writing — review and editing: J. C., T. L., K. T. S.

Conflicts of interest

There are no conflicts to declare.

Acknowledgements

JC and DAC acknowledge the support from the European Union's Horizon 2020 research and innovation program under the Marie Skłodowska-Curie Actions grant agreement "Advanced Machine Learning for Innovative Drug Discovery (AIDD)" No. 956832. DAC additionally acknowledges the funding from the European Commission's Horizon 2020 Framework Programme (AiChemist; grant no. 101120466).

References

1 A. C. Anderson, *Chem. Biol.*, 2003, **10**, 787–797.
2 M. Batool, B. Ahmad and S. Choi, *Int. J. Mol. Sci.*, 2019, **20**, 2783.
3 M. Ragoza, T. Masuda and D. R. Koes, *Chem. Sci.*, 2022, **13**, 2701–2713.
4 H. Green, D. R. Koes and J. D. Durrant, *Chem. Sci.*, 2021, **12**, 8036–8047.
5 L. Wang, R. Bai, X. Shi, W. Zhang, Y. Cui, X. Wang, C. Wang, H. Chang, Y. Zhang, J. Zhou, W. Peng, W. Zhou and B. Huang, *Sci. Rep.*, 2022, **12**, 15100.
6 S. Luo, J. Guan, J. Ma and J. Peng, *Adv. Neural Inf. Process. Syst.*, 2021, 6229–6239.
7 M. Liu, Y. Luo, K. Uchino, K. Maruhashi and S. Ji, *Proceedings of the 39th International Conference on Machine Learning*, 2022, pp. 13912–13924.
8 C. Tan, Z. Gao, S. Z. Li, Target-aware Molecular Graph Generation, *arXiv*, 2022, preprint, arXiv:2202.04829, DOI: **10.48550/arXiv.2202.04829**.
9 X. Peng, S. Luo, J. Guan, Q. Xie, J. Peng and J. Ma, *Proceedings of the 39th International Conference on Machine Learning*, 2022, pp. 17644–17655.
10 A. S. Powers, H. H. Yu, P. Suriana, R. V. Koodli, T. Lu, J. M. Paggi and R. O. Dror, *ACS Cent. Sci.*, 2023, **9**, 2257–2267.
11 E. Hoogeboom, V. G. Satorras, C. Vignac and M. Welling, *Proceedings of the 39th International Conference on Machine Learning*, 2022, pp. 8867–8887.
12 J. Guan, W. W. Qian, X. Peng, Y. Su, J. Peng and J. Ma, *The Eleventh International Conference on Learning Representations*, 2023.
13 A. Schneuing, Y. Du, C. Harris, A. Jamasb, I. Igashov, W. Du, T. Blundell, P. Lió, C. Gomes, M. Welling, M. Bronstein and B. Correia, Structure-based Drug Design with Equivariant Diffusion Models, *arXiv*, 2023, preprint, arXiv:2210.13695, DOI: **10.48550/arXiv.2210.13695**.
14 G. Corso, H. Stärk, B. Jing, R. Barzilay and T. S. Jaakkola, *The Eleventh International Conference on Learning Representations*, 2023.
15 J. Zhu, Z. Gu, J. Pei and L. Lai, *Chem. Sci.*, 2024, **15**, 7926–7942.
16 Y. Xia, K. Wu, P. Deng, R. Liu, Y. Zhang, H. Guo, Y. Cui, Q. Pei, L. Wu, S. Xie, S. Chen, X. Lu, S. Hu, J. Wu, C.-K. Chan, S. Chen, L. Zhou, N. Yu, H. Liu, J. Guo, T. Qin and T.-Y. Liu, *Target-aware Molecule Generation for Drug Design Using a Chemical Language Model*, 2024, **https://www.biorxiv.org/content/early/2024/01/08/2024.01.08.574635**.
17 R. Gómez-Bombarelli, J. N. Wei, D. Duvenaud, J. M. Hernández-Lobato, B. Sánchez-Lengeling, D. Sheberla, J. Aguilera-Iparraguirre, T. D. Hirzel, R. P. Adams and A. Aspuru-Guzik, *ACS Cent. Sci.*, 2018, **4**, 268–276.
18 R. Winter, F. Montanari, A. Steffen, H. Briem, F. Noé and D.-A. Clevert, *Chem. Sci.*, 2019, **10**, 8016–8024.
19 P. Dhariwal and A. Q. Nichol, Diffusion models beat GANs on image synthesis, *Adv. Neural Inf. Process. Syst.*, 2021, **34**, 8780–8794, **https://papers.nips.cc/paper_files/paper/2021/hash/49ad23d1ec9fa4bd8d77d02681df5cfa-Abstract.html**.
20 T. Sterling and J. J. Irwin, *J. Chem. Inf. Model.*, 2015, **55**, 2324–2337.
21 P. C. D. Hawkins and A. Nicholls, *J. Chem. Inf. Model.*, 2012, **52**, 2919–2936.
22 P. G. Francoeur, T. Masuda, J. Sunseri, A. Jia, R. B. Iovanisci, I. Snyder and D. R. Koes, *J. Chem. Inf. Model.*, 2020, **60**, 4200–4215.
23 T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever and D. Amodei, *Adv. Neural Inf. Process. Syst.*, 2020, 1877–1901.
24 J. Devlin, M. W. Chang, K. Lee and K. Toutanova, *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*, NAACL-HLT, 2019, 1(2019), pp. 4171–4186, DOI: **10.18653/V1/N19-1423**.
25 R. Winter, F. Montanari, F. Noé and D.-A. Clevert, *Chem. Sci.*, 2019, **10**, 1692–1701.

26 S. Liu, H. Guo and J. Tang, *The Eleventh International Conference on Learning Representations*, 2023.
27 S. Zaidi, M. Schaarschmidt, J. Martens, H. Kim, Y. W. Teh, A. Sanchez-Gonzalez, P. Battaglia, R. Pascanu and J. Godwin, *The Eleventh International Conference on Learning Representations*, 2023.
28 T. Le, J. Cremer, F. Noé, D.-A. Clevert and K. Schütt, *The Twelfth International Conference on Learning Representations*, 2024.
29 M. Buttenschoen, G. M. Morris and C. M. Deane, *PoseBusters: AI-based docking methods fail to generate physically valid poses or generalise to novel sequences*, 2024, DOI: **10.1039/D3SC04185A**.
30 C. Harris, K. Didi, A. R. Jamasb, C. K. Joshi, S. V. Mathis, P. Lio and T. Blundell, *Benchmarking Generated Poses: How Rational is Structure-based Drug Design with Generative Models?*, 2023.
31 A. K. Rappé, C. Casewit, K. S. Colwell, W. A. Goddard and W. M. Skiff, *J. Am. Chem. Soc.*, 1992, **114**, 10024–10035.
32 G. A. Landrum and S. Riniker, *J. Chem. Inf. Model.*, 2024, **64**, 1560–1567.
33 O. T. Unke, S. Chmiela, H. E. Sauceda, M. Gastegger, I. Poltavsky, K. T. Schütt, A. Tkatchenko and K.-R. Müller, *Chem. Rev.*, 2021, **121**, 10142–10186.
34 C. Vignac, N. Osman, L. Toni and P. Frossard, *Machine Learning and Knowledge Discovery in Databases: Research Track*, European Conference, ECML PKDD 2023, Turin, Italy, September 18-22, 2023, Proceedings, Part II, 2023, pp. 560–576.
35 A. Doucet, N. de Freitas and N. Gordon, in *An Introduction to Sequential Monte Carlo Methods*, Springer New York, New York, NY, 2001, pp. 3–14.
36 P. Del Moral, A. Doucet and A. Jasra, *J. R. Stat. Soc. Ser. B Stat. Method*, 2006, **68**, 411–436.
37 B. L. Trippe, J. Yim, D. Tischer, D. Baker, T. Broderick, R. Barzilay and T. S. Jaakkola, *The Eleventh International Conference on Learning Representations*, 2023.
38 L. Wu, B. L. Trippe, C. A. Naesseth, J. P. Cunningham and D. Blei, *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.
39 P. G. Francoeur, T. Masuda, J. Sunseri, A. Jia, R. B. Iovanisci, I. Snyder and D. R. Koes, *J. Chem. Inf. Model.*, 2020, **60**, 4200–4215.
40 M. Backenköhler, J. Groß, V. Wolf and A. Volkamer, *J. Chem. Inf. Model.*, 2024, **64**, 4009–4020.
41 D. Schaller, C. D. Christ, J. D. Chodera and A. Volkamer, *Benchmarking Cross-Docking Strategies for Structure-Informed Machine Learning in Kinase Drug Discovery*, 2023, **https://www.biorxiv.org/content/early/2023/09/14/2023.09.11.557138**.

Supplementary Information (SI) for Chemical Science.

Extended Supplementary Information
PILOT: Equivariant diffusion for pocket conditioned de novo ligand generation with multi-objective guidance via importance sampling

Julian Cremer[1,3,*], Tuan Le[1,2,*], Frank Noé[2, 4], Djork-Arné Clevert[1], and Kristof T. Schütt[1]

[1]Machine Learning & Computational Sciences, Pfizer Worldwide R&D, Berlin, Germany
[2]Department of Mathematics and Computer Science, Freie Universität Berlin, Germany
[3]Computational Science Laboratory, Universitat Pompeu Fabra, PRBB, Spain
[4]Microsoft Research AI4Science, Microsoft, Berlin, Germany
[*]Corresponding author: [julian.cremer, tuan.le]@pfizer.com

A. Learning curves: From scratch vs fine-tuned

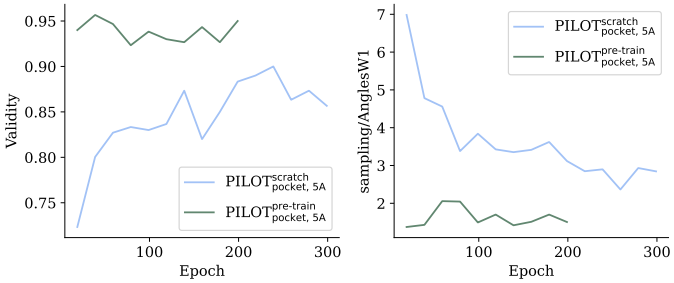


Figure A1: Learning curves comparison for the from scratch trained model against fine-tuned model. For better visibility, we fine-tuned the pre-trained model for 200 epochs to obtain more metrics for visualisation. We show the molecule validity as well as samples/AnglesW1 metric and observe that the fine-tuned model achieves much better metrics already after 20 epochs of training.

We observe that the pre-trained PILOT model achieves faster training convergence compared to the model that is trained from scratch on the CrossDocked dataset. In Figure A1 we show the evaluation curves for both models trained on the 5A PL-complex dataset, i.e., comparing $\text{PILOT}^{\text{scratch}}_{\text{pocket, 5A}}$ against $\text{PILOT}^{\text{pre-train}}_{\text{pocket, 5A}}$. As shown, the pre-trained model

achieves superior metrics compared to the model trained from scratch already in the first evaluation period after 20 epochs of training. Specifically, the molecule validity for the fine-tuned model accomplishes a highest value of 95.67% after 40 epochs while the model trained from scratch only achieves a maximum molecule validity of 90.00% after 240 epochs of training. As the pre-trained model has learned on a vast chemical space from the Enamine Real Diversity, this model achieved to learn general chemistry rules related to valency. Nonetheless, when trained on CrossDocked an extensive distribution shift is expected, since in this scenario, the model inputs a much larger sample in form of a protein-ligand complex. Learning correct geometries how a ligand might fit into a protein pocket is a non-trivial task. Although both models are not explicitly trained to generate a ligand that fit a pocket in a physical sense, like a docking tool, the ligands generated by the fine-tuned model achieves predominant angles distribution metrics compared to the from scratch model. This means that the ligands sampled from the fine-tuned model attain better geometries resembles the angles statistics present in protein-ligand-complexes in the validation set.

B. CrossDocked2020: Analysis

B.1 Pocket size distribution

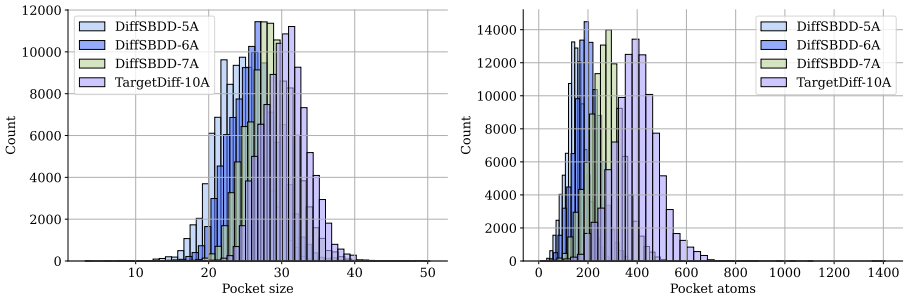


Figure B1: The size distribution for protein pockets in the CrossDocked2020 dataset based on the cutoff radius to create the ligand-pocket complex. The namings DiffSBDD or Targetdiff folowed by xA describe the method to determine the protein pocket with cutoff x. Left: size distribution. Right: distribution of pocket atoms.

To create the Protein-Ligand (PL) complex, the cutoff radius determines which protein atoms should be included next to all ligand atoms, to build the PL complex.

The work by[1] in DiffSBDD creates the PL-complex by computing for each atom in each residue in the protein all pairwise distances to the ligand atoms. As long as one distance from the residues' atom is below the defined cutoff to any ligand atom, the entire residue is included into the protein pocket. Hence choosing such selection through the minimum function potentially creates larger PL complexes, but ensures that all interactions between

Dataset	Mean	Standard Deviation	Median	Skewness
DiffSBDD-5A	24.16	3.35	24.20	0.02
DiffSBDD-6A	26.22	3.33	26.26	0.08
DiffSBDD-7A	28.47	3.25	28.40	0.13
TargetDiff-10A	30.49	3.10	30.47	0.17

Table B1: Statistics of pocket size for different datasets.

Dataset	Mean	Standard Deviation	Median	Skewness
DiffSBDD-5A	152.73	42.04	151.00	0.26
DiffSBDD-6A	198.62	51.90	197.00	0.17
DiffSBDD-7A	274.94	68.32	274.00	0.19
TargetDiff-10A	393.83	90.70	394.00	0.15

Table B2: Statistics of number of pocket atoms for different datasets.

protein-atoms to the ligands are considered.

The PL complex creation in TargetDiff[2] chooses a query point from each residue through the center of mass. Based on this query point the distance to all ligand atoms are computed and the residue with its atoms are included into the PL complex, if any distance between the query point to any ligand atom is below the cutoff. Note that by computing the CoM in the first place, an initial reduction has already been done which can lead so fewer interactions and hence smaller PL complexes. The authors of TargetDiff estimate the pocket size by computing the top 10 farthest pairwise distances of protein atoms. Based on that, they select the median of that as the pocket size for robustness. As the cutoff increases, we observe that the protein pocket also increases as shown in the both panels of Figure B1 for pocket size as well as the number of atoms in the protein pocket. The TargetDiff-10A PL-dataset is particularly large with protein pockets having mean size of 393 atoms. Having larger PL complexes might impede the optimization of the diffusion models since smaller batch sizes and backbones with less trainable parameters are only feasible to fit on a single GPU. Additionally, a smaller cutoff also enables faster training since less message passing steps are required. We believe that a cutoff of 7A is a good trade-off, in that it enables the diffusion model to propagate distant information but also does not fall into the risk of overfitting on a potentially smaller PL complex. The latter is particularly important in the setting of generalization when the diffusion model generates ligands on a new protein target.

B.2 Metrics dependency on ligand size

In this section we show how several metrics are dependent on the ligand size. To this end, we ran six additional sample experiments with $\text{EQGAT}_{\text{diff}}^{\text{pocket, 6A}}$ trained on CrossDocked2020, where ligands are generated without any guidance, i.e., unconditionally with only the protein pocket as context. Figure B2 shows the results. In particular, as de-

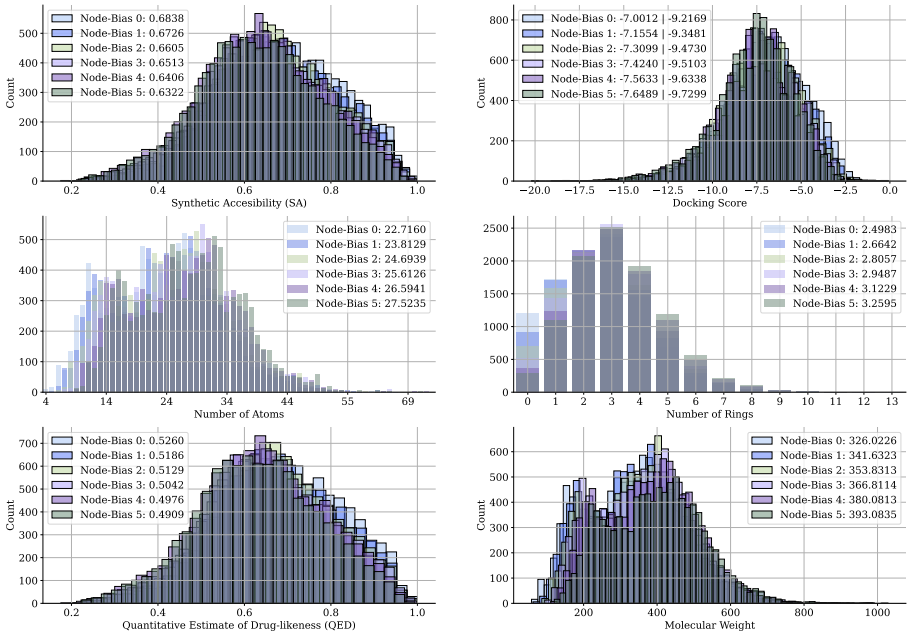


Figure B2: Evaluation metrics for 10,000 ligands where the number of atoms was first sampled from the training set prior and additionally a node bias of n added. Here $n = \{0, 1, 2, 3, 4, 5\}$. The caption shows the mean value across each of the 5 different sets per metric. For the docking scores panel, the first value describes the mean value among all targets, while the second value refers to the top-10% mean value among all targets.

scribed in the main text, larger ligands with more atoms, shown through increasing node bias n tend to have lower synthetic accessibility (SA) score as well as QED and a better (smaller) docking score. Therefore, it is important to take the ligand size distribution into account when evaluating generative models based on docking scores, since the later is negatively correlated with ligand size.

B.3 Ring distribution

To delve deeper into our analysis, we also examine the distribution of ring structures, a known challenge for 3D-based models[3]. The top panel in Fig. B3 illustrates the occurrence of fused and uncommon rings for all models. We observe that TargetDiff, as well as our models, tend to generate more uncommon rings compared to the train and test sets. However, both the SA- and SA-docking-conditional models effectively mitigate this issue by reducing the number of uncommon rings and aligning more closely with the distribution observed in the training and test data.

Consistent with our earlier discussion, the docking-conditional model exhibits a strong propensity for generating numerous rings, including fused and uncommon ones. As depicted in the lower panel of Fig. B3, all models also tend to produce rings that are less

common in drug-like molecules, such as three-, four-, seven-membered, or larger rings. These ring structures are often associated with poor synthetic accessibility, chemical stability, toxicity, or metabolic instability[4–6].

In contrast, five- and six-membered heterocycles containing one or more heteroatoms are considered the gold standard in drug-like molecules[4,6,7], and we observe that these are well represented in the sample space following the training distribution.

Notably, the SA-conditional model effectively regulates the formation of unfavorable ring systems, particularly three- and seven-membered rings. Conversely, the SA-docking-conditional model strikes a reasonable balance, with only a slight increase in seven-membered rings compared to the docking-conditional model, where such rings are more prevalent.

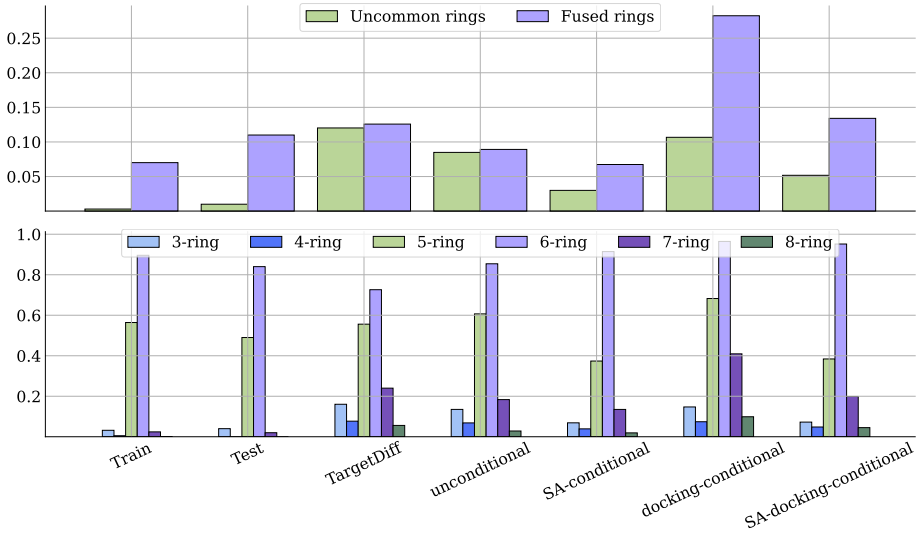


Figure B3: Evaluation of the ring systems in all sampled ligands across test targets. **Top**: Histogram detailing the percentage of uncommon and fused rings for all ligands. **Bottom**: Histogram displaying the distribution of ring sizes from three- to eight-membered rings. Five- and six-membered rings are considered the most drug-like.

C. Hyperparameters for Importance Sampling

In Algorithm 1, we present the property-guided sampling algorithm, which we will further discuss in this section. To perform SA- and docking-score optimization on CrossDocked, we first optimize the SA-score in the population for several iterations. After this initial optimization, we then optimize for docking-score, using a population that has been filtered or biased based on the SA-score guidance. In Table C1, we report the start and end points, as well as the temperature parameter for the unbounded SA-score maximization and unbounded docking-score minimization. Note that the reverse diffusion trajectory

sampling includes $T = 500$ timesteps. We set the population size to 40. This means that for a given protein target P, each batch consists of 40 ligands optimized for high SA-scores and low docking scores. We continue the sampling process until 100 valid ligands are generated.

Property	Start	End	N	Temperature τ
SA-score	0	200	10	0.1
Docking-score	10	250	10	0.1

Table C1: Settings for importance sampling to optimize SA- and docking scores. Start and end columns determine when the importance sampling is performed in the iteration ranging from 1 to 500.

In our experiments, we tried optimizing both SA- and docking score simultaneously by either pointwise adding or multiplying the importance weights $\{w_{k,\text{SA}}\}_{k=1}^{K}$ with $\{w_{k,\text{dock}}\}_{k=1}^{K}$ for each intermediate ligand k for varying time intervals including $(1, 200)$, $(100, 300)$ and $(300, 500)$. We did not see satisfying results where both criteria are optimized in the final ligands, which might be possible to achieve through different hyperparameters including temperature annealing. For this reason, we choose to optimize each property in consecutive order, where the SA-guidance shall act as an initial filter to discard synthetic infeasible (noisy) ligands between steps $(100, 250)$, while the filtered noisy ligands are then guided to minimize docking score in the interval $(300, 400)$. Notice that we do not employ the guidance in iteration in the intervals, but every 10, such that each SA- and docking-guidance include 15 and 10 guidance steps respectively. For sampling, we leveraged an Nvidia A100 with 40GB GPU memory.

C.1 Comparison to Gradient Guidance

We list the metrics for joint optimization fo SA- and docking score in Table C2. For a protein target, we do not sample the number of atoms for generated ligands, but fix the number to the reference ligand to have a fair comparison between importance sampling and classifier guidance.

Table C2: Comparison of importance sampling (IS) vs. classifier guidance (CG) on CrossDocked testset. We generated 100 ligands for each of the 100 pockets in the test set. We report mean time per pocket in minutes, validity, uniqueness and the corresponding property values.

Method	Time [min]	Validity [%]	Uniqueness [%]	SA				Docking			
				Steps	Every-N	λ, τ	Score	Steps	Every-N	λ, τ	Score
IS	3.51	92.29	75.55	$0-200$	10	0.1	0.7531	$150-250$	10	0.1	-7.679
CG	15.02	77.17	64.97	$0-500$	1	0.1	0.8248	$0-500$	1	0.1	-8.4397
CG	3.71	93.18	83.22	$0-200$	10	0.1	0.7205	$150-250$	10	0.1	-7.1523
IS & CG	3.72	92.54	58.79	$0-200$	10	0.1	0.7608	$150-250$	10	0.1	-8.0151
Unconditional	3.01	93.15	83.63	–	–	–	0.711	–	–	–	-7.0248

We experienced GPU memory issues when performing classifier (gradient) guidance because backpropagation is costly, especially for larger protein-ligand complexes, disabling

us to use the default batch size of 40 on an A100 40GB GPU . Reducing the batch size is necessary to avoid out of memory errors, with the disadvantage of increasing the overall running time. We set the batch size to 25 to perform both importance sampling as well as classifier guidance with the same settings on an Nvidia H100 with 80GB GPU memory. For importance sampling, we can easily set a batch size of 40 or even 50 ligands per pocket, achieving an average pocket time of about 3.28 minutes to generate 100 ligands, showing the advantage of importance sampling compared to classifier guidance, to allow for larger ligand batches during generation, effectively reducing runtime.

Since our proposed importance sampling filters for trajectories that either maximize or minimize a property, gradient guidance can also be performed afterwards. To enable both approaches (IS & GC), we choose the same filtering steps to maintain a low computational budget and perform backpropagation to obtain the gradients with respect to atomic coordinates to shift the positions in space after promising candidates were selected based on their importance weights. This results in a generated ligand set with a similar validity of 92.54% but a reduced uniqueness rate of 0.58. The mean SA score reaches 0.76, while the docking score reaches -8.01 as shown in the second last row in Table C2.

D. Kinodata-3D: Analysis

D.1 Correlation

	pIC50	# Rings	# Rotatable bonds	# Atoms	QED	SA
# Rings	0.19					
# Rotatable bonds	0.16	0.15				
# Atoms	0.28	0.66	0.68			
QED	-0.18	-0.53	-0.62	-0.79		
SA	-0.24	-0.37	-0.20	-0.39	0.18	
logP	0.04	0.22	0.13	0.30	-0.39	0.38

Figure D1: Correlation matrix of pIC50s, number of rings, number of atoms, QEDs, and SAs on the Kinodata-3D training set.

We show the correlation matrix on the Kinodata-3D dataset in Figure D1. Similar to the CrossDocked dataset, we observe that metrics like the QED and SA score are negatively correlated with the number of atoms and rings. As opposed to CrossDocked, in the Kinodata-3D a negative correlation of -0.39 between logP and QED is observed while in CrossDocked the correlation amounts to 0.36. One possible explanation for this is that Kinodata-3D consists of experimental kinase-ligand assay data and therefore only considers ligands that covers a smaller chemical space where the logP covers a potentially smaller range.

In Figure D3 we show the ring distributions on the Kinodata-3D dataset next to the distribution that our EQGAT-Diff-Pocket models produced in 4 settings, namely

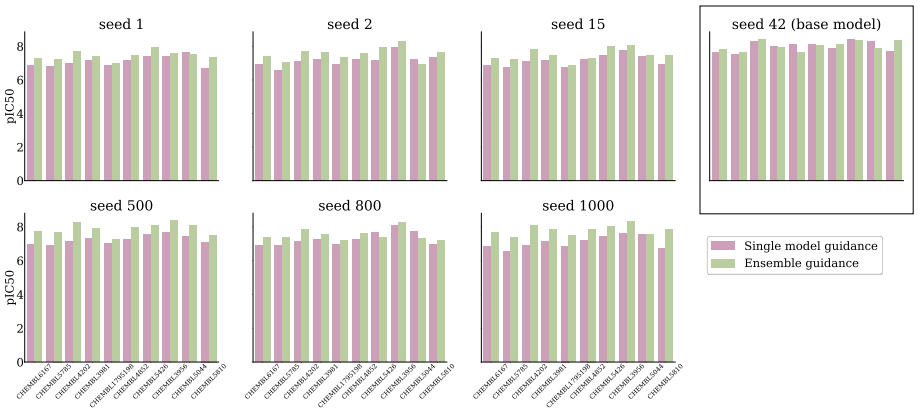


Figure D2: Single model guidance is compared to ensemble guidance. A base model, here the model with seed 42, is used to sample 100 ligands per target. In the case of single model guidance (seed 42 model guides itself), a variety of models trained with different seeds evaluate the sampled ligands with respect to pIC_{50} values. In the case of ensemble guidance, an ensemble of models, here seed 42, 500, and seed 1000, are used for pIC_{50} guidance. Again, all seed models evaluate the sampled ligands with respect to pIC_{50}. We can see that throughout targets and seeds, the ensemble guidance not only works best in terms of pIC_{50} but also in terms of stability and generality. The base model assigns similar pIC_{50} values to its samples for both, single model and ensemble guidance. Nevertheless, across seed models (involved in ensemble guidance or not) the samples taken from the single model guidance exhibit a significantly worse pIC_{50} prediction in contrast to ensemble guidance. Here, all seed models predict similarly high pIC_{50} values suggesting that ensemble guidance leads to a more stable and most importantly more general set of samples.

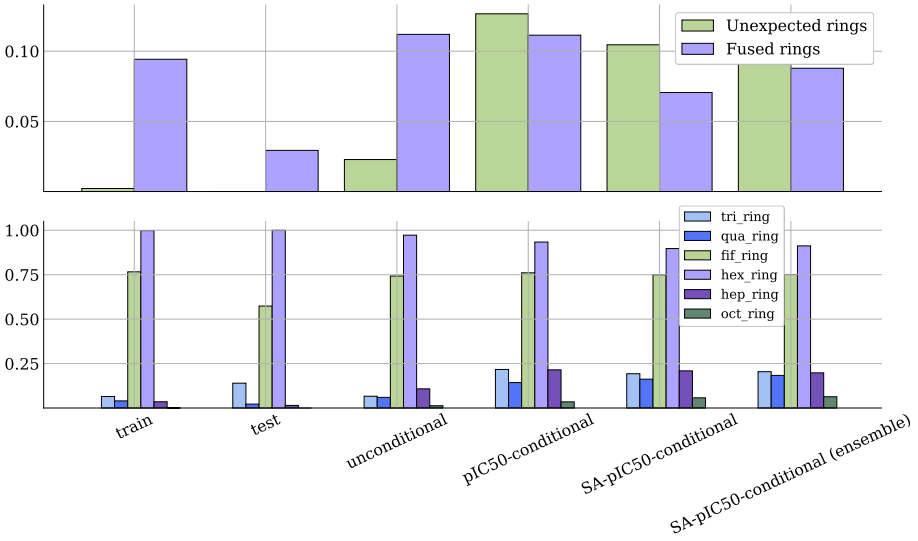


Figure D3: Ring distribution on the Kinodata-3D dataset

the unconditional, single pIC_{50}-conditional, joint SA-pIC_{50}-conditinoal as joint SA-pIC_{50}-conditional ensemble. Similar to CrossDocked, an unconditional model that only generates ligands based on a protein (kinase) pocket as context, the number of fused rings increases, leading to unfavourable molecules that might be difficult to synthesize. When leveraging the importance sampling in the pIC_{50}-conditional generation, we observe that the number of fused rings stays the same but unexpected rings increases. Also, an increase in 4, 7- and 8-membered rings is observed, when we aim in maximizing the pIC_{50} to search for ligands that exhibit high (predicted) binding affinity. One potential reason why that happens might lie in the dataset the pIC_{50} expert model was trained on. As shown in Figure D1 a negative correlation between pIC_{50} and QED as well as SA is observed. Thus, with higher pIC_{50}, ligands tend to become less synthesizable and drug-like according to those metrics. Fortunately, we can leverage our framework and jointly optimize for high pIC_{50} as well as SAscore, which is done in the SA-pIC_{50}-conditional samples. By leveraging expert models for both properties, we can reduce the number of unexpected rings as well as fused rings, although the 4,7- and 8-membered rings are still prevalent but potentially in ligands that according to the SAScore are still better evaluated compared to the pIC_{50}-conditional setting only.

References

[1] A. Schneuing, Y. Du, C. Harris, A. Jamasb, I. Igashov, W. Du, T. Blundell, P. Lió, C. Gomes, M. Welling, M. Bronstein and B. Correia, *Structure-based Drug Design with Equivariant Diffusion Models*, 2023, https://arxiv.org/abs/2210.13695.

[2] J. Guan, W. W. Qian, X. Peng, Y. Su, J. Peng and J. Ma, The Eleventh International Conference on Learning Representations, 2023.

[3] Y. Xia, K. Wu, P. Deng, R. Liu, Y. Zhang, H. Guo, Y. Cui, Q. Pei, L. Wu, S. Xie, S. Chen, X. Lu, S. Hu, J. Wu, C.-K. Chan, S. Chen, L. Zhou, N. Yu, H. Liu, J. Guo, T. Qin and T.-Y. Liu, *Target-aware Molecule Generation for Drug Design Using a Chemical Language Model*, 2024, https://www.biorxiv.org/content/early/2024/01/08/2024.01.08.574635.

[4] R. D. Taylor, M. MacCoss and A. D. G. Lawson, *Journal of Medicinal Chemistry*, 2014, **57**, 5845–5859.

[5] X.-C. Yu, C.-C. Zhang, L.-T. Wang, J.-Z. Li, T. Li and W.-T. Wei, *Organic Chemistry Frontiers*, 2022, **9**, 4757–4781.

[6] A. Rusu, I.-M. Moga, L. Uncu and G. Hancu, *Pharmaceutics*, 2023, **15**, 2.

[7] J. Jampilek, *Molecules*, 2019, **24**, 6.

Chapter 3

Conclusion

This thesis has contributed to the development of deep learning models to process molecular representation suitable for either molecular property prediction or molecular generation and optimization by exploring the symmetries of the data with the overall goal of improving the efficiency of computer-aided methods for accelerated drug discovery. As molecules can be described through different raw and unprocessed representations, e.g., line notation as in SMILES, 2D topological graphs with connected atoms through bonds, or point clouds as 3D conformers potentially embedded in a protein pocket forming a complex, all representations have in common that they form set, where the concept of group equivariance, such as permutations and rotations, plays a central role with practical implications on graph representation learning and generation.

In the first publication, we have shown with the Neuraldecipher model that reverse-engineer fixed-sized extended-circular fingerprints (ECFPs) of molecules back to their SMILES is to some extent possible depending on the input size of the ECFP. While this application has not been researched extensively, we demonstrated that exchanging ECFPs between institutions in academia or the private sector comprises the risk of loss of intellectual property. Unless a non-disclosure agreement has been placed, ECFPs should not be shared. With this in mind, collaborative drug discovery programs among private and public sectors are advised to opt for federated learning approaches, as pioneered in the MELLODY project (Oldenhof et al., 2023; Heyndrickx et al., 2024).

In the second publication, we demonstrated that the relatively underexplored hypercomplex neural networks also benefit for 2D graph representation learning. We show that learning the multiplication rule inspired by the complex or quaternion algebras reduced the number of trainable parameters in the network without performance decrease on molecular property prediction benchmarks compared to their real counterpart, suggesting that the assumed internal hypercomplex hidden representation is richer and better suited for generalization. While the proposed

PHC-GNN architecture is relatively simple and adapted from the GIN model (Xu et al., 2019), we believe future research can be dedicated to an improved version of PHC-GNN that use novel aggregations schemes, e.g., attention-based, and include additional positional encodings to increase expressivity, while also relying on the parameterized hypercomplex linear layer to mimic and learn the multiplication rule from the data.

In the third publication, we lift the space for graph representation learning by including the spatial coordinates in the geometric graph. This implies that the neural network architecture must consider rotation equivariance next to permutation symmetry. We proposed a novel and efficient 3D GNN architecture termed EQGAT that is equivariant to permutations as well as 3D rotations. Similar to other recent Cartesian-based SO(3) equivariant GNNs, we implement rotation equivariant rank-1 features through scalarization but enjoy increased model expressivity through the interplay of equivariant features by using the cross-product, a unique and novel feature, as well as attention mechanism. Especially on larger point clouds of bio-molecules, our proposed EQGAT architecture is competitive, if not superior, to other SO(3)-equivariant GNNs while being computationally inexpensive on protein property prediction benchmarks. One possible limitation of the EQGAT architecture on protein learning tasks, however, is the lack of initial featurization of node or edge features computed from the backbone atoms from each residue to extract relative spatial encodings as proposed by Ingraham et al. (2019) and also extended in the GVP-GNN model (Jing et al., 2021). Although EQGAT might be able to learn the features through several message passing layers, we believe that including such initial features has the potential to increase expressiveness next to the baked-in equivariance.

In the fourth publication, we used the EQGAT architecture from publication 3 for 3D molecule generation as a denoising model to sample *de novo* molecules. We proposed several critical design choices, such as timestep dependent loss weighting or the suitable noising scheme, to effectively train diffusion models on 3D molecular data, answering open questions for practitioners new to generative modeling with diffusion models for 3D molecules. Since diffusion models act as denoisers, gradually predicting more data-like molecules, the choice of EQGAT, which is permutation and rotation equivariant, was made due to the experience with this architecture. Nonetheless, we demonstrated that the EQGAT architecture with the same diffusion design choices is parameter-lighter and superior to the MiDi model (Vignac et al., 2023), which was state-of-the-art at the time of publication.

In the final and fifth publication, we delved into structure-based drug design following up with the EQGAT-diff architecture from publication 4. We show that pre-training 3D diffusion models on only small molecules has the advantage of improved sample quality when fine-tuned on data-scarce protein-ligand complexes. Since conditioning the ligand generation on the protein pocket alone is not sufficient to design an effective drug, we proposed an importance sampling algorithm that enables multi-objective generation of ligands tailored for fixed

protein pockets. We demonstrate that the novel importance sampling algorithm is a computationally efficient alternative to other conditioning methods, such as classifier (gradient) guidance. Since the exploration of chemical space heavily relies on the surrogate (property) model, we further show that an ensemble of surrogate models for properties like predicted pIC50 increases the reliability of generated ligands since the uncertainty is implicitly modeled through the ensemble.

The diffusion models introduced in publications 4 and 5 for unconditional and conditional 3D molecule design within the context of a protein pocket are currently limited in the quadratic complexity during message passing to exchange information between atoms from the ligand, while interactions between protein and ligand are restricted based on a distance cutoff. Furthermore, diffusion models tend to have slow sampling speed, particularly in larger PL-complexes where every forward pass amounts to a costly denoising step. For SBDD, developing customized protein-ligand encoder network architectures might be an exciting research direction, while computation time during sampling can be, e.g., accelerated by changing the generative algorithm from diffusion to, e.g., continuous normalizing flows that enjoy fewer sampling steps during the reverse trajectory by following a learned deterministic dynamics, but are trained through a diffusion like simulation-free training objective (Lipman et al., 2023; Liu et al., 2023; Albergo and Vanden-Eijnden, 2023) making them favorable and scalable.

Despite significant advances in deep graph representation learning for drug discovery, numerous challenges remain. Among the most critical issues are data-related concerns, including scarcity, heterogeneity, and labeling directly influenced by the origin where the data was produced in the first place (Bender and Cortes-Ciriano, 2021). Furthermore, another challenging task is properly evaluating generative machine learning algorithms, like diffusion models in SBDD. Including physical priors during the training of the diffusion model in a simulation-free manner is an interesting research direction to increase the credibility of the generated samples of the model as an alternative to existing methods that apply guidance at the inference stage during sampling. We are confident that the research community will continue to achieve remarkable advancements in these areas and beyond. We are proud to have contributed to this vibrant and evolving field. We hope that the work presented in this thesis has contributed to the development of improved graph representation and generation methods.

Bibliography

Albergo, M. S. and Vanden-Eijnden, E. (2023). Building normalizing flows with stochastic interpolants. In *The Eleventh International Conference on Learning Representations*.

Anderson, A. C. (2003). The process of structure-based drug design. *Chemistry & Biology*, 10(9):787–797.

Anderson, B., Hy, T. S., and Kondor, R. (2019). Cormorant: Covariant molecular neural networks. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Atz, K., Grisoni, F., and Schneider, G. (2021). Geometric deep learning on molecular representations. *Nature Machine Intelligence*, 3(12):1023–1032.

Bao, F., Zhao, M., Hao, Z., Li, P., Li, C., and Zhu, J. (2023). Equivariant energy-guided SDE for inverse molecular design. In *The Eleventh International Conference on Learning Representations*.

Battaglia, P. W., Hamrick, J. B., Bapst, V., Sanchez-Gonzalez, A., Zambaldi, V., Malinowski, M., Tacchetti, A., Raposo, D., Santoro, A., Faulkner, R., Gulcehre, C., Song, F., Ballard, A., Gilmer, J., Dahl, G., Vaswani, A., Allen, K., Nash, C., Langston, V., Dyer, C., Heess, N., Wierstra, D., Kohli, P., Botvinick, M., Vinyals, O., Li, Y., and Pascanu, R. (2018). Relational inductive biases, deep learning, and graph networks.

Batzner, S., Musaelian, A., Sun, L., Geiger, M., Mailoa, J. P., Kornbluth, M., Molinari, N., Smidt, T. E., and Kozinsky, B. (2022). E(3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *Nature Communications*, 13(1):2453.

Bender, A. and Cortes-Ciriano, I. (2021). Artificial intelligence in drug discovery: what is realistic, what are illusions? part 2: a discussion of chemical and biological data. *Drug Discovery Today*, 26(4):1040–1052.

Bender, A. and Glen, R. C. (2004). Molecular similarity: a key technique in molecular informatics. *Organic & biomolecular chemistry*, 2(22):3204–3218.

Brandstetter, J., Hesselink, R., van der Pol, E., Bekkers, E. J., and Welling, M. (2022). Geometric and physical quantities improve e(3) equivariant message passing. In *International Conference on Learning Representations*.

Brandstetter, J., van den Berg, R., Welling, M., and Gupta, J. K. (2023). Clifford neural layers for PDE modeling. In *The Eleventh International Conference on Learning Representations*.

Bronstein, M. M., Bruna, J., Cohen, T., and Veličković, P. (2021). Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*.

Bronstein, M. M., Bruna, J., LeCun, Y., Szlam, A., and Vandergheynst, P. (2017). Geometric deep learning: Going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42.

Bruna, J., Zaremba, W., Szlam, A., and LeCun, Y. (2013). Spectral networks and locally connected networks on graphs. *arXiv preprint arXiv:1312.6203*.

Chen, R. T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. (2018). Neural ordinary differential equations. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc.

Clifford (1871). Preliminary sketch of biquaternions. *Proceedings of the London Mathematical Society*, s1-4(1):381–395.

Comminiello, D., Lella, M., Scardapane, S., and Uncini, A. (2019). Quaternion convolutional neural networks for detection and localization of 3d sound events. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE.

Cremer, J., Le, T., Noé, F., Clevert, D.-A., and Schütt, K. T. (2024). Pilot: equivariant diffusion for pocket-conditioned de novo ligand generation with multi-objective guidance via importance sampling. *Chem. Sci.*, 15:14954–14967.

Danihelka, I., Wayne, G., Uria, B., Kalchbrenner, N., and Graves, A. (2016). Associative long short-term memory. In Balcan, M. F. and Weinberger, K. Q., editors, *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1986–1994, New York, New York, USA. PMLR.

Davies, M., Nowotka, M., Papadatos, G., Dedman, N., Gaulton, A., Atkinson, F., Bellis, L., and Overington, J. P. (2015). ChEMBL web services: streamlining access to drug discovery data and utilities. *Nucleic Acids Res*, 43(W1):W612–20.

Defferrard, M., Bresson, X., and Vandergheynst, P. (2016). Convolutional neural networks on graphs with fast localized spectral filtering. *Advances in neural information processing systems*, 29.

Dhariwal, P. and Nichol, A. Q. (2021). Diffusion models beat GANs on image synthesis. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems*.

Dorst, L. and Mann, S. (2002). Geometric algebra: a computational framework for geometrical applications. *IEEE Computer Graphics and Applications*, 22(3):24–31.

Duvenaud, D. K., Maclaurin, D., Iparraguirre, J., Bombarell, R., Hirzel, T., Aspuru-Guzik, A., and Adams, R. P. (2015). Convolutional networks on graphs for learning molecular fingerprints. *Advances in neural information processing systems*, 28.

Dwivedi, V. P., Joshi, C. K., Luu, A. T., Laurent, T., Bengio, Y., and Bresson, X. (2020). Benchmarking graph neural networks. *arXiv preprint arXiv:2003.00982*.

Fuchs, F., Worrall, D., Fischer, V., and Welling, M. (2020). Se(3)-transformers: 3d roto-translation equivariant attention networks. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1970–1981. Curran Associates, Inc.

Fukushima, K. (1980). Neocognitron: a self organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biol Cybern*, 36(4):193–202.

Garg, V., Jegelka, S., and Jaakkola, T. (2020). Generalization and representational limits of graph neural networks. In III, H. D. and Singh, A., editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 3419–3430. PMLR.

Gasteiger, J., Groß, J., and Günnemann, S. (2020). Directional message passing for molecular graphs. In *International Conference on Learning Representations*.

Gaulton, A., Hersey, A., Nowotka, M., Bento, A. P., Chambers, J., Mendez, D., Mutowo, P., Atkinson, F., Bellis, L. J., Cibrián-Uhalte, E., Davies, M., Dedman, N., Karlsson, A., Magariños, M. P., Overington, J. P., Papadatos, G., Smit, I., and Leach, A. R. (2016). The ChEMBL database in 2017. *Nucleic Acids Res*, 45(D1):D945–D954.

Gebauer, N., Gastegger, M., and Schütt, K. (2019). Symmetry-adapted generation of 3d point sets for the targeted discovery of molecules. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems 32*, pages 7566–7578. Curran Associates, Inc.

Gebauer, N. W. A., Gastegger, M., Hessmann, S. S. P., Müller, K.-R., and Schütt, K. T. (2022). Inverse design of 3d molecular structures with conditional generative neural networks. *Nature Communications*, 13(1).

Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O., and Dahl, G. E. (2017). Neural message passing for quantum chemistry. In *International conference on machine learning*, pages 1263–1272. PMLR.

Glen, R. C., Bender, A., Arnby, C. H., Carlsson, L., Boyer, S., and Smith, J. (2006). Circular fingerprints: flexible molecular descriptors with applications from physical chemistry to adme. *IDrugs*, 9(3):199.

Göller, A. H., Kuhnke, L., Montanari, F., Bonin, A., Schneckener, S., Ter Laak, A., Wichard, J., Lobell, M., and Hillisch, A. (2020). Bayer's in silico admet platform: A journey of machine learning over the past two decades. *Drug Discovery Today*, 25(9):1702–1709.

Gómez-Bombarelli, R., Wei, J. N., Duvenaud, D., Hernández-Lobato, J. M., Sánchez-Lengeling, B., Sheberla, D., Aguilera-Iparraguirre, J., Hirzel, T. D., Adams, R. P., and Aspuru-Guzik, A. (2018). Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276.

Grassucci, E., Zhang, A., and Comminiello, D. (2022). PHNNs: Lightweight neural networks via parameterized hypercomplex convolutions. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–13.

Green, H., Koes, D. R., and Durrant, J. D. (2021). Deepfrag: a deep convolutional neural network for fragment-based lead optimization. *Chem. Sci.*, 12:8036–8047.

Guan, J., Peng, X., Jiang, P., Luo, Y., Peng, J., and Ma, J. (2023a). Linkernet: Fragment poses and linker co-design with 3d equivariant diffusion. In Oh, A., Naumann, T., Globerson, A., Saenko, K., Hardt, M., and Levine, S., editors, *Advances in Neural Information Processing Systems*, volume 36, pages 77503–77519. Curran Associates, Inc.

Guan, J., Qian, W. W., Peng, X., Su, Y., Peng, J., and Ma, J. (2023b). 3d equivariant diffusion for target-aware molecule generation and affinity prediction. In *The Eleventh International Conference on Learning Representations*.

Guan, J., Zhou, X., Yang, Y., Bao, Y., Peng, J., Ma, J., Liu, Q., Wang, L., and Gu, Q. (2023c). DecompDiff: Diffusion models with decomposed priors for structure-based drug design. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J., editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 11827–11846. PMLR.

Hamilton, W. R. (1844). Theory of quaternions. *Proceedings of the Royal Irish Academy (1836-1869)*, 3:1–16.

Heyndrickx, W., Mervin, L., Morawietz, T., Sturm, N., Friedrich, L., Zalewski, A., Pentina, A., Humbeck, L., Oldenhof, M., Niwayama, R., Schmidtke, P., Fechner, N., Simm, J., Arany, A., Drizard, N., Jabal, R., Afanasyeva, A., Loeb, R., Verma, S., Harnqvist, S., Holmes, M., Pejo, B., Telenczuk, M.,

Holway, N., Dieckmann, A., Rieke, N., Zumsande, F., Clevert, D.-A., Krug, M., Luscombe, C., Green, D., Ertl, P., Antal, P., Marcus, D., Do Huu, N., Fuji, H., Pickett, S., Acs, G., Boniface, E., Beck, B., Sun, Y., Gohier, A., Rippmann, F., Engkvist, O., Göller, A. H., Moreau, Y., Galtier, M. N., Schuffenhauer, A., and Ceulemans, H. (2024). Melloddy: Cross-pharma federated learning at unprecedented scale unlocks benefits in qsar without compromising proprietary information. *Journal of Chemical Information and Modeling*, 64(7):2331–2344. PMID: 37642660.

Ho, J., Jain, A., and Abbeel, P. (2020). Denoising diffusion probabilistic models. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9(8):1735–1780.

Hoogeboom, E., Satorras, V. G., Vignac, C., and Welling, M. (2022). Equivariant diffusion for molecule generation in 3D. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S., editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 8867–8887. PMLR.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79(8):2554–2558.

Hu, W., Fey, M., Zitnik, M., Dong, Y., Ren, H., Liu, B., Catasta, M., and Leskovec, J. (2020). Open graph benchmark: Datasets for machine learning on graphs. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 22118–22133. Curran Associates, Inc.

Huang, L., Zhang, H., Xu, T., and Wong, K.-C. (2023). Mdm: Molecular diffusion model for 3d molecule generation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 37(4):5105–5112.

Igashov, I., Stärk, H., Vignac, C., Schneuing, A., Satorras, V. G., Frossard, P., Welling, M., Bronstein, M., and Correia, B. (2024). Equivariant 3d-conditional diffusion model for molecular linker design. *Nature Machine Intelligence*, 6(4):417–427.

Ingraham, J., Garg, V., Barzilay, R., and Jaakkola, T. (2019). Generative models for graph-based protein design. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Jing, B., Eismann, S., Suriana, P., Townshend, R. J. L., and Dror, R. (2021). Learning from protein structure with geometric vector perceptrons. In *International Conference on Learning Representations*.

Kadan, A., Ryczko, K., Roitberg, A., and Yamazaki, T. (2024). Guided multi-objective generative ai to enhance structure-based drug design.

Karras, T., Aittala, M., Aila, T., and Laine, S. (2022). Elucidating the design space of diffusion-based generative models. In Oh, A. H., Agarwal, A., Belgrave, D., and Cho, K., editors, *Advances in Neural Information Processing Systems*.

Kassel, D. B. (2004). Applications of high-throughput ADME in drug discovery. *Curr. Opin. Chem. Biol.*, 8(3):339–345.

Katritzky, A. R. and Gordeeva, E. V. (1993). Traditional topological indexes vs electronic, geometrical, and combined molecular descriptors in QSAR/QSPR research. *J. Chem. Inf. Comput. Sci.*, 33(6):835–857.

Kearnes, S., McCloskey, K., Berndl, M., Pande, V., and Riley, P. (2016). Molecular graph convolutions: moving beyond fingerprints. *Journal of Computer-Aided Molecular Design*, 30(8):595–608.

Kingma, D. P. and Welling, M. (2014). Auto-Encoding Variational Bayes. In *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*.

Kipf, T. N. and Welling, M. (2017). Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*.

Kotsias, P.-C., Arús-Pous, J., Chen, H., Engkvist, O., Tyrchan, C., and Bjerrum, E. J. (2020). Direct steering of de novo molecular generation with descriptor conditional recurrent neural networks. *Nature Machine Intelligence*, 2(5):254–265.

Kwon, Y., Kang, S., Choi, Y.-S., and Kim, I. (2021). Evolutionary design of molecules based on deep learning and a genetic algorithm. *Scientific Reports*, 11(1):17304.

Landrum, G., Tosco, P., Kelley, B., Rodriguez, R., Cosgrove, D., Vianello, R., sriniker, gedeck, Jones, G., NadineSchneider, Kawashima, E., Nealschneider, D., Dalke, A., Swain, M., Cole, B., Turk, S., Savelev, A., Vaucher, A., Wójcikowski, M., Take, I., Scalfani, V. F., Probst, D., Ujihara, K., Walker, R., Pahl, A., guillaume godin, tadhurst cdd, Lehtivarjo, J., Bérenger, F., and Bisson, J. (2024). rdkit/rdkit: 2024_03_4 (q1 2024) release.

Le, T., Bertolini, M., Noé, F., and Clevert, D.-A. (2021). Parameterized hypercomplex graph neural networks for graph classification. In *Artificial Neural Networks and Machine Learning – ICANN 2021: 30th International Conference on Artificial Neural Networks, Bratislava, Slovakia, September 14–17, 2021, Proceedings, Part III*, page 204–216, Berlin, Heidelberg. Springer-Verlag.

Le, T., Cremer, J., Noe, F., Clevert, D.-A., and Schütt, K. T. (2024). Navigating the design space of equivariant diffusion-based generative models for de novo

3d molecule generation. In *The Twelfth International Conference on Learning Representations*.

Le, T., Noe, F., and Clevert, D.-A. (2022). Representation learning on biomolecular structures using equivariant graph attention. In Rieck, B. and Pascanu, R., editors, *Proceedings of the First Learning on Graphs Conference*, volume 198 of *Proceedings of Machine Learning Research*, pages 30:1–30:17. PMLR.

Le, T., Winter, R., Noé, F., and Clevert, D.-A. (2020). Neuraldecipher–reverse-engineering extended-connectivity fingerprints (ecfps) to their molecular structures. *Chemical science*, 11(38):10378–10389.

LeCun, Y., Bengio, Y., et al. (1995). Convolutional networks for images, speech, and time series. *The handbook of brain theory and neural networks*, 3361(10):1995.

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *nature*, 521(7553):436–444.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lionta, E., Spyrou, G., Vassilatis, D. K., and Cournia, Z. (2014). Structure-based virtual screening for drug discovery: principles, applications and recent advances. *Curr Top Med Chem*, 14(16):1923–1938.

Lipman, Y., Chen, R. T. Q., Ben-Hamu, H., Nickel, M., and Le, M. (2023). Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*.

Liu, M., Luo, Y., Uchino, K., Maruhashi, K., and Ji, S. (2022). Generating 3D molecules for target protein binding. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S., editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 13912–13924. PMLR.

Liu, X., Gong, C., and qiang liu (2023). Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*.

Lopez, E., Grassucci, E., Capriotti, D., and Comminiello, D. (2024). Towards explaining hypercomplex neural networks.

Luo, C. (2022). Understanding diffusion models: A unified perspective.

Luo, S., Guan, J., Ma, J., and Peng, J. (2021). A 3d generative model for structure-based drug design. In Ranzato, M., Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems*, volume 34, pages 6229–6239. Curran Associates, Inc.

Luo, Y. and Ji, S. (2022). An autoregressive flow model for 3d molecular geometry generation from scratch. In *International Conference on Learning Representations*.

Mahabadi, R. K., Henderson, J., and Ruder, S. (2021). Compacter: Efficient low-rank hypercomplex adapter layers. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems*.

Mayr, A., Klambauer, G., Unterthiner, T., and Hochreiter, S. (2016). Deeptox: toxicity prediction using deep learning. *Frontiers in Environmental Science*, 3:80.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In Singh, A. and Zhu, X. J., editors, *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, AISTATS 2017, 20-22 April 2017, Fort Lauderdale, FL, USA*, volume 54 of *Proceedings of Machine Learning Research*, pages 1273–1282. PMLR.

Montanari, F., Kuhnke, L., Ter Laak, A., and Clevert, D.-A. (2020). Modeling physico-chemical admet endpoints with multitask graph convolutional networks. *Molecules*, 25(1).

Morehead, A. and Cheng, J. (2024). Geometry-complete perceptron networks for 3d molecular graphs. *Bioinformatics*.

Morgan, H. L. (1965). The generation of a unique machine description for chemical structures-a technique developed at chemical abstracts service. *Journal of chemical documentation*, 5(2):107–113.

Moriwaki, H., Tian, Y.-S., Kawashita, N., and Takagi, T. (2018). Mordred: a molecular descriptor calculator. *Journal of Cheminformatics*, 10(1):4.

O'Boyle, N. M., Banck, M., James, C. A., Morley, C., Vandermeersch, T., and Hutchison, G. R. (2011). Open babel: An open chemical toolbox. *Journal of Cheminformatics*, 3(1):33.

Oldenhof, M., Ács, G., Pejó, B., Schuffenhauer, A., Holway, N., Sturm, N., Dieckmann, A., Fortmeier, O., Boniface, E., Mayer, C., Gohier, A., Schmidtke, P., Niwayama, R., Kopecky, D., Mervin, L., Rathi, P. C., Friedrich, L., Formanek, A., Antal, P., Rahaman, J., Zalewski, A., Heyndrickx, W., Oluoch, E., Stößel, M., Vančo, M., Endico, D., Gelus, F., de Boisfossé, T., Darbier, A., Nicollet, A., Blottière, M., Telenczuk, M., Nguyen, V. T., Martinez, T., Boillet, C., Moutet, K., Picosson, A., Gasser, A., Djafar, I., Simon, A., Arany, A., Simm, J., Moreau, Y., Engkvist, O., Ceulemans, H., Marini, C., and Galtier, M. (2023). Industry-scale orchestrated federated learning for drug discovery. In *Proceedings of the Thirty-Seventh AAAI Conference on Artificial Intelligence and Thirty-Fifth Conference on Innovative Applications*

of Artificial Intelligence and Thirteenth Symposium on Educational Advances in Artificial Intelligence, AAAI'23/IAAI'23/EAAI'23. AAAI Press.

Olivecrona, M., Blaschke, T., Engkvist, O., and Chen, H. (2017). Molecular de-novo design through deep reinforcement learning. *Journal of cheminformatics*, 9(1):1–14.

Parcollet, T., Ravanelli, M., Morchid, M., Linarès, G., Trabelsi, C., Mori, R. D., and Bengio, Y. (2019). Quaternion recurrent neural networks. In *International Conference on Learning Representations*.

Paul, S. M., Mytelka, D. S., Dunwiddie, C. T., Persinger, C. C., Munos, B. H., Lindborg, S. R., and Schacht, A. L. (2010). How to improve r&d productivity: the pharmaceutical industry's grand challenge. *Nature Reviews Drug Discovery*, 9(3):203–214.

Peng, X., Luo, S., Guan, J., Xie, Q., Peng, J., and Ma, J. (2022). Pocket2Mol: Efficient molecular sampling based on 3D protein pockets. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvari, C., Niu, G., and Sabato, S., editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 17644–17655. PMLR.

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

Pozdnyakov, S. N. and Ceriotti, M. (2022). Incompleteness of graph neural networks for points clouds in three dimensions.

PubChem (2009). Pubchem fingerprint specification.

Ragoza, M., Masuda, T., and Koes, D. R. (2022). Generating 3d molecules conditional on receptor binding sites with deep generative models. *Chem. Sci.*, 13:2701–2713.

Ramsundar, B., Kearnes, S., Riley, P., Webster, D., Konerding, D., and Pande, V. (2015). Massively multitask networks for drug discovery. *ArXiv*.

Ramsundar, B., Liu, B., Wu, Z., Verras, A., Tudor, M., Sheridan, R. P., and Pande, V. (2017). Is multitask deep learning practical for pharma? *Journal of Chemical Information and Modeling*, 57(8):2068–2076.

Rogers, D. and Hahn, M. (2010). Extended-connectivity fingerprints. *Journal of chemical information and modeling*, 50(5):742–754.

Ruhe, D., Gupta, J. K., De Keninck, S., Welling, M., and Brandstetter, J. (2023). Geometric clifford algebra networks. In Krause, A., Brunskill, E., Cho, K., Engelhardt, B., Sabato, S., and Scarlett, J., editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 29306–29337. PMLR.

Rumelhart, D. E., Hinton, G. E., and Williams, R. J. (1986). Learning internal representations by error propagation. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundations*, page 318–362, Cambridge, MA, USA. MIT Press.

Runcie, N. T. and Mey, A. S. (2023). Silvr: Guided diffusion for molecule generation. *Journal of Chemical Information and Modeling*, 63(19):5996–6005.

Satorras, V. G., Hoogeboom, E., Fuchs, F. B., Posner, I., and Welling, M. (2021a). E(n) equivariant normalizing flows. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems*.

Satorras, V. G., Hoogeboom, E., and Welling, M. (2021b). E(n) equivariant graph neural networks. In Meila, M. and Zhang, T., editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 9323–9332. PMLR.

Scarselli, F., Gori, M., Tsoi, A. C., Hagenbuchner, M., and Monfardini, G. (2009). The graph neural network model. *IEEE Transactions on Neural Networks*, 20(1):61–80.

Schmidhuber, J. (2015). Deep learning in neural networks: An overview. *Neural networks*, 61:85–117.

Schneuing, A., Du, Y., Harris, C., Jamasb, A., Igashov, I., Du, W., Blundell, T., Lió, P., Gomes, C., Welling, M., Bronstein, M., and Correia, B. (2023). Structure-based drug design with equivariant diffusion models.

Schütt, K., Kindermans, P.-J., Sauceda Felix, H. E., Chmiela, S., Tkatchenko, A., and Müller, K.-R. (2017). Schnet: A continuous-filter convolutional neural network for modeling quantum interactions. In Guyon, I., Luxburg, U. V., Bengio, S., Wallach, H., Fergus, R., Vishwanathan, S., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Schütt, K., Unke, O., and Gastegger, M. (2021). Equivariant message passing for the prediction of tensorial properties and molecular spectra. In Meila, M. and Zhang, T., editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 9377–9388. PMLR.

Schütt, K. T., Arbabzadah, F., Chmiela, S., Müller, K. R., and Tkatchenko, A. (2017). Quantum-chemical insights from deep tensor neural networks. *Nature Communications*, 8(1):13890.

Segler, M. H., Kogej, T., Tyrchan, C., and Waller, M. P. (2018). Generating focused molecule libraries for drug discovery with recurrent neural networks. *ACS central science*, 4(1):120–131.

Shih, H.-P., Zhang, X., and Aronov, A. M. (2018). Drug discovery effectiveness from the standpoint of therapeutic mechanisms and indications. *Nature Reviews Drug Discovery*, 17(1):19–33.

Simm, G., Pinsler, R., and Hernandez-Lobato, J. M. (2020). Reinforcement learning for molecular design guided by quantum mechanics. In III, H. D. and Singh, A., editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 8959–8969. PMLR.

Simm, G. N. C., Pinsler, R., Csányi, G., and Hernández-Lobato, J. M. (2021). Symmetry-aware actor-critic for 3d molecular design. In *International Conference on Learning Representations*.

Sohl-Dickstein, J., Weiss, E., Maheswaranathan, N., and Ganguli, S. (2015). Deep unsupervised learning using nonequilibrium thermodynamics. In Bach, F. and Blei, D., editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2256–2265, Lille, France. PMLR.

Tay, Y., Zhang, A., Luu, A. T., Rao, J., Zhang, S., Wang, S., Fu, J., and Hui, S. C. (2019). Lightweight and efficient neural natural language processing with quaternion networks. In Korhonen, A., Traum, D., and Màrquez, L., editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1494–1503, Florence, Italy. Association for Computational Linguistics.

Thomas, N., Smidt, T., Kearnes, S., Yang, L., Li, L., Kohlhoff, K., and Riley, P. (2018). Tensor field networks: Rotation- and translation-equivariant neural networks for 3d point clouds.

Townshend, R. J. L., Vögele, M., Suriana, P. A., Derry, A., Powers, A., Laloudakis, Y., Balachandar, S., Jing, B., Anderson, B. M., Eismann, S., Kondor, R., Altman, R., and Dror, R. O. (2021). ATOM3d: Tasks on molecules in three dimensions. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*.

Trabelsi, C., Bilaniuk, O., Zhang, Y., Serdyuk, D., Subramanian, S., Santos, J. F., Mehri, S., Rostamzadeh, N., Bengio, Y., and Pal, C. J. (2018). Deep complex networks. In *International Conference on Learning Representations*.

Ucak, U. V., Ashyrmamatov, I., and Lee, J. (2023). Reconstruction of lossless molecular representations from fingerprints. *Journal of Cheminformatics*, 15(1):26.

Vignac, C., Osman, N., Toni, L., and Frossard, P. (2023). Midi: Mixed graph and 3d denoising diffusion for molecule generation. In *Machine Learning and Knowledge Discovery in Databases: Research Track: European Conference, ECML PKDD 2023, Turin, Italy, September 18–22, 2023, Proceedings, Part II*, page 560–576, Berlin, Heidelberg. Springer-Verlag.

Villar, S., Hogg, D. W., Storey-Fisher, K., Yao, W., and Blum-Smith, B. (2021). Scalars are universal: Equivariant machine learning, structured like classical physics. In Beygelzimer, A., Dauphin, Y., Liang, P., and Vaughan, J. W., editors, *Advances in Neural Information Processing Systems*.

Wang, L., Bai, R., Shi, X., Zhang, W., Cui, Y., Wang, X., Wang, C., Chang, H., Zhang, Y., Zhou, J., Peng, W., Zhou, W., and Huang, B. (2022). A pocket-based 3d molecule generative model fueled by experimental electron density. *Scientific Reports*, 12(1):15100.

Wang, R., Fang, X., Lu, Y., and Wang, S. (2004). The PDBbind database: collection of binding affinities for protein-ligand complexes with known three-dimensional structures. *J Med Chem*, 47(12):2977–2980.

Wei, J. N., Duvenaud, D., and Aspuru-Guzik, A. (2016). Neural networks for the prediction of organic chemistry reactions. *ACS Central Science*, 2(10):725–732.

Weininger, D. (1988). 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.

Weiss, T., Mayo Yanes, E., Chakraborty, S., Cosmo, L., Bronstein, A. M., and Gershoni-Poranne, R. (2023). Guided diffusion for inverse molecular design. *Nature Computational Science*, 3(10):873–882.

Winter, R., Montanari, F., Noé, F., and Clevert, D.-A. (2019). Learning continuous and data-driven molecular descriptors by translating equivalent chemical representations. *Chemical science*, 10(6):1692–1701.

Wouters, O. J., McKee, M., and Luyten, J. (2020). Estimated Research and Development Investment Needed to Bring a New Medicine to Market, 2009-2018. *JAMA*, 323(9):844–853.

Xu, K., Hu, W., Leskovec, J., and Jegelka, S. (2019). How powerful are graph neural networks? In *International Conference on Learning Representations*.

Yang, K., Swanson, K., Jin, W., Coley, C., Eiden, P., Gao, H., Guzman-Perez, A., Hopper, T., Kelley, B., Mathea, M., et al. (2019). Analyzing learned molecular representations for property prediction. *Journal of chemical information and modeling*, 59(8):3370–3388.

Yap, C. W. (2011). Padel-descriptor: An open source software to calculate molecular descriptors and fingerprints. *Journal of Computational Chemistry*, 32.

Zhang, A., Tay, Y., Zhang, S., Chan, A., Luu, A. T., Hui, S., and Fu, J. (2021a). Beyond fully-connected layers with quaternions: Parameterization of hypercomplex multiplications with $1/n$ parameters. In *International Conference on Learning Representations*.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. (2017). Understanding deep learning requires rethinking generalization. In *International Conference on Learning Representations*.

Zhang, C., Xie, Y., Bai, H., Yu, B., Li, W., and Gao, Y. (2021b). A survey on federated learning. *Knowledge-Based Systems*, 216:106775.

Zhang, X., Wang, L., Helwig, J., Luo, Y., Fu, C., Xie, Y., Liu, M., Lin, Y., Xu, Z., Yan, K., Adams, K., Weiler, M., Li, X., Fu, T., Wang, Y., Yu, H., Xie, Y., Fu, X., Strasser, A., Xu, S., Liu, Y., Du, Y., Saxton, A., Ling, H., Lawrence, H., Stärk, H., Gui, S., Edwards, C., Gao, N., Ladera, A., Wu, T., Hofgard, E. F., Tehrani, A. M., Wang, R., Daigavane, A., Bohde, M., Kurtin, J., Huang, Q., Phung, T., Xu, M., Joshi, C. K., Mathis, S. V., Azizzadenesheli, K., Fang, A., Aspuru-Guzik, A., Bekkers, E., Bronstein, M., Zitnik, M., Anandkumar, A., Ermon, S., Liò, P., Yu, R., Günnemann, S., Leskovec, J., Ji, H., Sun, J., Barzilay, R., Jaakkola, T., Coley, C. W., Qian, X., Qian, X., Smidt, T., and Ji, S. (2023). Artificial intelligence for science in quantum, atomistic, and continuum systems.

Ziv, Y., Marsden, B., and Deane, C. M. (2024). Molsnapper: Conditioning diffusion for structure based drug design. *bioRxiv*.