STREAMLINING GENERATIVE MODELS FOR STRUCTURE-BASED DRUG DESIGN

Anonymous authors
Paper under double-blind review

ABSTRACT

Generative models for structure-based drug design (SBDD) aim to generate novel 3D molecules for specified protein targets *in silico*. The prevailing paradigm focuses on model expressivity - typically with powerful Graph Neural Network (GNN) models - but is agnostic to binding affinity during training, potentially overlooking better molecules. We address this issue with a two-pronged approach: learn an economical surrogate for affinity to infer an unlabeled molecular graph, and optimize for labels conditioned on this graph and desired molecular properties (e.g., QED, SA). The resulting model FastSBDD achieves state-of-the-art results as well as streamlined computation and model size (up to 1000x faster and with 100x fewer trainable parameters compared to existing methods), paving way for improved docking software. We also establish rigorous theoretical results to expose the representation limits of GNNs in SBDD contexts and the generalizability of our affinity scoring model, advocating more emphasis on generalization going forward.

1 INTRODUCTION

Identifying *ligands* that bind well to a protein target and have desired properties is a key challenge of Structure-based drug design (SBDD), while experimental determination of binding is costly. Recently deep generative models have proposed to accelerate SBDD by suggesting good candidate molecules from learned conditional ligand distribution given a target protein from experimental or *in silico* binding observations (Vamathevan et al., 2019; Bilodeau et al., 2022). These approaches include autoregressive models (Peng et al., 2022; Luo et al., 2021a), variational autoencoders (Ragoza et al., 2022), reinforcement learning (Li et al., 2021), and diffusion models (Schneuing et al., 2022).

However, existing methods are oblivious to binding affinity as they seek to match the observed ligand distribution, which often includes molecules with varying affinities. Furthermore, they optimize jointly atom types and 3D coordinates with powerful GNN encoders with typically expensive training and sampling procedures. We address these issues with a novel two-phase generative method *FastSBDD* that draws inspiration from the success of performance-aware methods in other domains (Joachims, 2005), and directly optimizes for the metrics of interest, i.e. properties and binding affinity.

Specifically, we first learn a cheap affinity surrogate to infer an unlabeled molecular graph, and then optimize its atom labels and coordinates. We demonstrate the versatility of our approach with three versions of FastSBDD tailored to the problems of (i) drug repurposing, which generates samples from a known ligand set; (ii) drug generation, which generates novel unseen ligands; and (iii) property optimisation, which seeks to control both molecular properties and binding. FastSBDD achieves SBDD state of the art with up to 1000x speed up and 100x fewer parameters.

A broader objective, and contribution, of this work is to foster theoretical underpinnings for SBDD. We take two important steps in this pursuit. First, we expose the limits on the expressivity of GNNs to distinguish distinct molecules conditioned on protein targets. Next we prove generalization bounds for our binding surrogate to ground its strong empirical performance. While such results on representational limits and generalization ability are known for GNN models in unconditional settings (Garg et al., 2020; Joshi et al., 2023; Ju et al., 2023), to our knowledge, these are the first results in conditional drug discovery contexts.

1.1 OUR CONTRIBUTIONS

We summarize our main contributions below. Specifically, we

1. **(Conceptual)** introduce metric-aware optimization for generative SBDD settings;
2. **(Methodological)** propose FastSBDD as a streamlined generative framework that optimizes for docking scores and physicochemical properties;
3. **(Empirical)** show that FastSBDD significantly outperforms existing methods with orders of magnitude fewer parameters and faster runtime; and
4. **(Theoretical)** analyze representational challenges of GNNs in SBDD contexts, and establish that FastSBDD generalizes well with respect to predicting scores for new target-ligand pairs.

2 PRELIMINARIES AND RELATED WORK

Deep (Geometric) Learning for Drug Design. Deep learning has enabled progress on various facets of drug design across molecular generation (Jin et al., 2018; Zang & Wang, 2020; Verma et al., 2022; Shi et al., 2020; Luo et al., 2021b; Hoogeboom et al., 2022; Igashov et al., 2022), molecular optimization (Jin et al., 2019) and drug repurposing (Pushpakom et al., 2019). It has also accelerated advancements in protein folding and design (Jumper et al., 2021; Wu et al., 2022b; Ahdritz et al., 2022; Verkuil et al., 2022; Hie et al., 2022; Shi et al., 2023; Watson et al., 2022; Ingraham et al., 2022; Wu et al., 2022a; Verma et al., 2023) and docking (Stärk et al., 2022b; Ganea et al., 2022; Corso et al., 2023). We refer the reader to the surveys by Chen et al. (2018) and Pandey et al. (2022).

Graph Neural Networks (GNNs) (Kipf & Welling, 2017; Hamilton et al., 2017; Veličković et al., 2018) have recently emerged as a workhorse for modeling data, such as molecules, that can be represented as graphs. They can be extended to encode important symmetries occurring in geometric graphs (Satorras et al., 2021), so have found success across diverse tasks in the drug design pipeline including docking (Corso et al., 2023) and molecular property prediction (Stärk et al., 2022a).

Molecular properties The goal of SBDD is to generate drug candidates with high binding affinity to a target protein, while controlling for other properties. We list below common molecular properties, calculated in practice using the RDKit software (Landrum et al., 2020), that are relevant to drug design (Bilodeau et al., 2022; Peng et al., 2022):

- Binding affinity – the strength of the natural connection between the ligand and the protein. A common surrogate for affinity is the Vina score (Alhossary et al., 2015).
- LogP – logarithm of the partition coefficient, which is a solubility indicator.
- QED – quantitative estimate of drug-likeness, a mixture of properties correlated with drugs.
- SA – Synthetic accessibility, an estimate of how easy it would be to synthesize the molecule.
- Lipinski's Rule of Five – A heuristic about whether a molecule would be an active oral drug.

Setting. We view proteins P as labelled graphs: each pocket $G = (V, E)$ consists of nodes $V = \{v_1, \ldots, v_N\}$ with v_i representing atoms, and edges $E \subseteq V \times V$. Each node $v = (a, \mathbf{s})$ is associated with an atom type $a \in \mathcal{A} = \{\mathtt{C}, \mathtt{N}, \mathtt{O}, \ldots\}$ and 3D coordinates $\mathbf{s} \in \mathbb{R}^3$, and each edge $e \in E$ with a bond type from $\{1, 2, 3\}$. We represent the molecule M analogously as another graph. We often include superscripts G^P and G^M to distinguish between protein and molecule graphs. We also distinguish between a labelled molecular graph G^M and unlabelled one U^M. The unlabelled graph contains information about the number of nodes and connectivity between them, but no information about atom types or 3D coordinates.

The problem of SBDD can be posed as modeling the conditional distribution $p(G^M|G^P)$. The main challenge of SBDD is to generate candidate ligands with high binding affinity to a target protein.

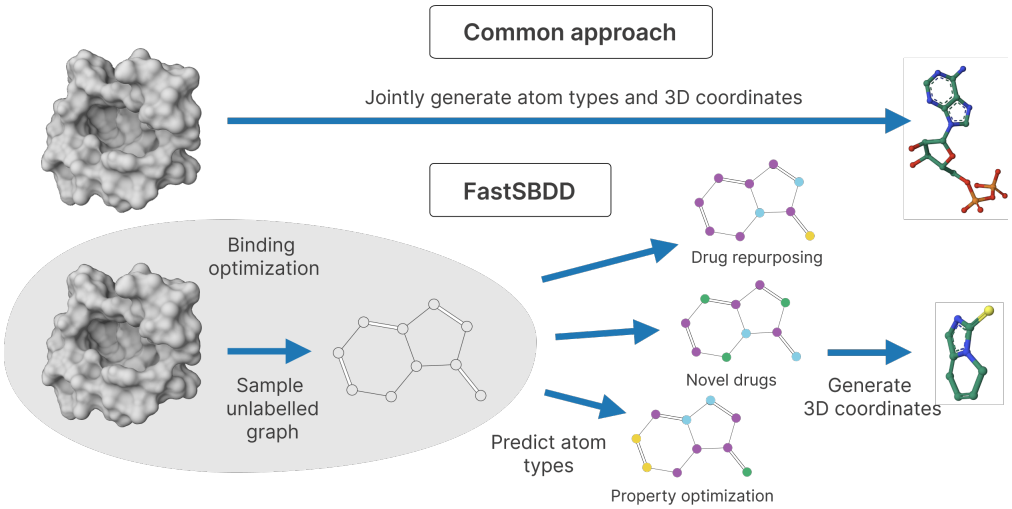


Figure 1: **Comparison of FastSBDD to common approaches**. Top: SBDD approaches commonly learn to approximate the data distribution of atom types and 3D coordinates conditioned on the protein pocket. Bottom: FastSBDD first generates the unlabelled graph explicitly optimized for binding affinity. Then it predicts atom types using different strategies designed for solving different tasks independently of the protein pocket. Finally, it generates a 3D configuration.

3 Fast Structure-Based Drug Design

3.1 Decoupling the Unlabelled Molecular Graph from Atom Types

One of the main innovations underlying our framework is the separation of molecular representation into the unlabeled molecular graph and atom types. As a motivation, we first investigate how Vina binding scores behave when we modify the atom types and coordinates, while keeping the underlying graph intact. After making these adjustments, we measured the Vina scores of the modified molecules and compared them to the scores of their unaltered counterparts.

Remarkably, our analysis showed a strong coefficient of determination, with an R^2 value of 0.76, between the scores of the original molecules and their modified counterparts. This significant result highlights that a considerable portion of the variability in binding affinity, as quantified by the Vina score, can be attributed to the information contained within the unlabeled molecular graph itself, even before incorporating atom-specific details. Please see Appendix A for details.

It is important to recognize that these findings pertain to the Vina software and not the intrinsic biological process of binding. While Vina is the most prevalent approximation, it remains just that – an approximation. Nonetheless, upon conducting similar experiments with Gnina (McNutt et al., 2021), a reportedly significantly more accurate binding approximation than Vina, we found analogous results. Please see Appendix A.4 for details. This suggests that the observed behavior is not solely an artifact of Vina but may be more widespread.

Inspired by these findings, we propose a new disentangled model FastSBDD (see Figure 1). We first generate the high-binding unlabelled graph structures U^M, and generate atom types $\mathbf{a}^M$ independently of the protein. The model is defined by a disentangled conditional distribution

$$p(G^M|G^P) \stackrel{\text{def}}{=} p(\mathbf{a}^M, \mathbf{s}^M|U^M, \cancel{G^P})p(U^M|G^P), \tag{1}$$

where we assume atoms $\mathbf{a}^M$ are conditionally independent of the protein pocket G^P given the unlabelled graph structure U^M. The generative model thus reduces to two independent components: unlabelled graph sampler and atom sampler. We discuss them below.

3.2 Unlabelled Graph Sampler

Typically, one would train the unlabelled graph model $p(U^M|G^P)$ to match the empirical distribution. However, we hypothesize that observed ligand-protein complexes contain examples with suboptimal

binding. We first learn a binding surrogate g_θ to predict binding affinity from unlabelled graph alone,

$$g_\theta(U^M, G^P) \approx \text{Vina}(G^M, G^P). \tag{2}$$

Next, we use g_θ to score unlabelled graphs from a repository of molecules, and choose those with the best predicted affinities. We define a generative distribution

$$p_\theta(U^M|G^P) = \text{UNIFORM}\Big(\big\{U^M \in \mathcal{U}^M \mid v_{\min} \leq g_\theta(G^P, U^M) \leq v_{\max}\big\}\Big), \tag{3}$$

where $\mathcal{U}^M$ is a database of unlabelled graph structures, and $v_{\min}, v_{\max}$ are threshold hyperparameters. We set $v_{\min}$ and $v_{\max}$ to be the 5th and 10th percentiles of all predictions for $\mathcal{U}^M$, and filter out the best 5% of the scores to avoid outliers. As $\mathcal{U}^M$, we used the ZINC250k dataset (Irwin et al., 2012) instead of the CrossDocked2020 (Francoeur et al., 2020), which only contains 8,000 unique ligands.

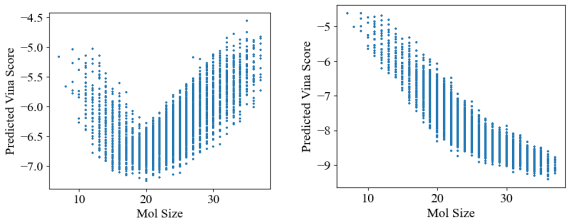
We note that instead of sampling from a database, one can define a deep generative model over unlabelled graphs. However, given that there is substantial variability in the molecular space even with predefined unlabelled graphs, we have not investigated this direction.

Architecture of the scoring model g_θ To parametrize the binding surrogate g_θ, we chose the E(3)-equivariant graph neural network (EGNN) (Satorras et al., 2021), which computes a learnable representation of the protein pocket. This protein pocket representation is concatenated with features describing the ligand's unlabelled graph. We chose as features (i) number of nodes, (ii) number of rings, (iii) number of rotatable bonds, and (iv) graph diameter. The features are passed through an MLP with ReLU activations. Remarkably, these four simple features already perform well, and more sophisticated methods like GNNs did not bring significant improvements. See Appendix B for details.

Training data for g_θ To train the binding affinity predictor g_θ, we need pairs of protein-ligand pairs with their affinity estimates. We chose not to use the CrossDocked2020 dataset, the gold standard for SBDD, for the following reasons: 1) even though it contains 100,000 protein-ligand pairs, there are only around 8,000 unique ligands, which can be insufficient for generalisation, 2) for a given protein pocket there may exist ligands with better affinity than what is present in the data and 3) for the scoring model to assign good values only to good structures, it needs to be trained also on examples with poor binding affinity that are not available.

To address the above, we construct a dataset ourselves. We sample 1000 diverse ligand-protein complexes from CrossDocked2020, and couple 50 additional random molecules from the ZINC250k. This results in 51,000 pairs, for which we compute the Vina ground-truth binding affinity. The scoring model g_θ is trained with stochastic gradient descent to minimize the mean squared error between prediction g and Vina scores. See Appendix B.1 for details.

Scoring model detects optimal molecule size To understand what the scoring learned, we visualized its predictions as a function of the size of the molecule. We notice that protein pockets fall under two cases: either there is an optimal size of the ligand (Figure 2a) or there is a monotonic relationship favouring larger ligands (Figure 2b). We hypothesize the model learnt the optimal ligand size per pocket, but for some pockets this size is larger than the largest available molecule (38 atoms) in the ZINC250k dataset.



(a) Protein pocket with optimal ligand size ≈ 20

(b) Protein pocket with optimal ligand size ≥ 38

Figure 2: **Optimal ligand size has two modes**. Representative examples showing that the scoring model learns the optimal ligand size which is different for different proteins.

3.3 ATOM SAMPLER

The atom sampler $p(\mathbf{a}^M, \mathbf{s}^M|U^M)$ generates atom types and coordinates of each atom. As mentioned in Section 3.1, the precise 3D conformation has marginal effect on the Vina score, and therefore we simply generate any valid 3D configuration with RDKit. We choose a factorisation

$$p(\mathbf{s}^M, \mathbf{a}^M|U^M) = p(\mathbf{s}^M|\mathbf{a}^M, U^M)p(\mathbf{a}^M|U^M), \tag{4}$$

where $p(\mathbf{s}^M|\mathbf{a}^M, U^M)$ is an off-the-shelf conformation generation model available in RDKit. In practice, the 3D configuration of the ligand must be in the vicinity of the protein pocket in order to correctly compute its binding affinity using Vina. We therefore train a separate model which predicts the center of mass of the ligand based on the protein pocket, and use it to center the conformers. See Appendix C for more details.

We propose three different approaches to modeling the atom type sampler $p(\mathbf{a}^M|U^M)$ that are tailored for solving different problems.

3.3.1 Drug repurposing

We begin with a model FastSBDD-$\mathcal{DR}$ for drug repurposing, i.e. sampling from known molecules that are synthesizable. We define the atom sampler to follow the empirical distribution of known ligands with a given unlabelled graph U^M:

$$p_{\text{DR}}(\mathbf{a}^M|U^M) = \text{UNIFORM}\left(\{\mathbf{a}^M|(\mathbf{a}^M, U^M) \in \mathcal{D}\}\right), \tag{5}$$

where $(\mathbf{a}^M, U^M)$ denotes a molecule with unlabelled graph U^M and atoms $\mathbf{a}^M$, and $\mathcal{D}$ is a molecular database. In our experiments, we used ZINC250 dataset (Irwin et al., 2012).

3.3.2 Novel drug generation

FastSBDD can also be applied for generation of novel drugs. We adapt an existing deep hierarchical generative model MoFlow (Zang & Wang, 2020), consisting of an unlabelled graph model f_U and a conditional atom model $f_{\mathbf{a}|U}$. The atom model induces a conditional distribution $p_{f_{\mathbf{a}|U}}(\mathbf{a}^M|U^M)$, which we use as our atom sampling distribution for novel drug generation:

$$p_{\mathcal{ND}}(\mathbf{a}^M|U^M) = p_{f_{\mathbf{a}|U}}(\mathbf{a}^M|U^M). \tag{6}$$

We use a pre-trained MoFlow with no further training or fine-tuning, so omit them from the trainable parameter count of the result (Table 1). We denote this model variant as FastSBDD-$\mathcal{ND}$.

3.3.3 Property optimization

Finally, we show that our approach allows for joint optimization of molecular properties and binding affinity. As an illustrative example, we follow Gómez-Bombarelli et al. (2018) and choose $5 \cdot \text{QED} + \text{SA}$ as the property mixture to optimize. We first sample 50 candidates for atom assignments using $\mathbf{a}^M \sim p_{\mathcal{ND}}(\mathbf{a}^M|U^M)$, and choose the one with the best value of the desired property. We emphasize that this optimization is performed with respect to atom types, while keeping the unlabelled graph structure U^M fixed. We denote this model FastSBDD-$\mathcal{PO}$.

We also consider optimization in the latent space of the atom type generative model. Since the atom type generative model of Section 3.3.2 is a normalizing flow, we can compute a latent representation of each atom type assignment. We use the pre-trained model for predicting molecular properties based on the latent representation of a molecule, and used its gradients to perform optimization in the latent space. This approach has the benefit of being faster as it does not require passing through the flow model at each iteration. In our experiments however this yielded worse results than the random search, suggesting suboptimal performance of the property predictor model.

4 Results

Data We follow related work of Peng et al. (2022); Schneuing et al. (2022) and use the CrossDocked2020 dataset (Francoeur et al., 2020) for evaluating the models in our experiments. We use the same train-test split, which seperates protein pockets based on their sequence similarity computed with MMseqs2 (Steinegger & Söding, 2017) and contains 100,000 train and 100 test protein-ligand pairs.

We follow the evaluation procedure described in Peng et al. (2022). For each of 100 test protein pockets, we

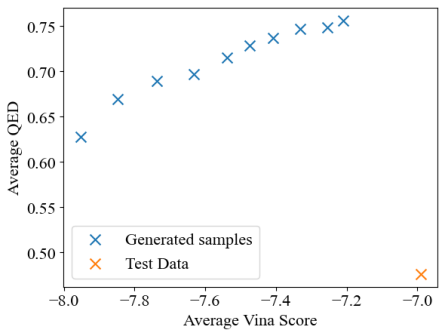


Figure 3: **Binding Affinity vs QED trade-off**. Our models allow for controlling the binding affinity of predictions.

Table 1: **FastSBDD finds high-quality drug candidates up to 1000x faster than competing methods**. Comparison of multiple properties of the CrossDocked2020 test data and other generation methods. Results for DiffSBDD were taken from Schneuing et al. (2022), where "High Affinity" was not reported. To report results for TargetDiff, we used samples provided by the authors. Reported runtime was taken from their paper, where standard deviation was not reported. † denotes method run exclusively on CPU. The #Params column refers to the number of trainable parameters.

	Vina Score (kcal/mol, ↓)	High Affinity (↑)	QED (↑)	SA (↑)	Diversity (↑)	#Params (↓)	Time (s, ↓)
Test set	-6.99 ± 2.16	-	0.48 ± 0.21	0.73 ± 0.14	-	-	-
DiffSBDD	-6.58 ± 2.06	-	0.50 ± 0.15	0.34 ± 0.09	**0.73** ± 0.11	3.5M	1634 ± 769
Pocket2Mol	-7.10 ± 2.56	0.55 ± 0.31	0.57 ± 0.16	0.74 ± 0.13	**0.72** ± 0.16	3.7M	2504 ± 220
TargetDiff	-6.91 ± 2.25	0.52 ± 0.32	0.48 ± 0.20	0.58 ± 0.13	**0.72** ± 0.09	2.5M	3428 ± NA
FastSBDD–$\mathcal{DR}$	**-7.82** ± 1.47	**0.72** ± 0.36	0.66 ± 0.15	**0.78** ± 0.08	0.68 ± 0.05	**23K**	**1.8**† ± 0.4
FastSBDD–$\mathcal{ND}$	**-7.78** ± 1.47	**0.71** ± 0.34	0.61 ± 0.18	0.69 ± 0.09	0.68 ± 0.06	**23K**	3.9† ± 0.9
FastSBDD–$\mathcal{PO}$	**-7.98**± 1.46	**0.75** ± 0.35	**0.80** ± 0.10	0.73 ± 0.08	0.66 ± 0.06	**23K**	115† ± 11

sample 100 molecules using each of our sampling strategies described in this section. As evaluation metrics we report 1) **Vina Score** estimating binding affinity (Alhossary et al., 2015), 2) **High Affinity**, which is the percentage of generated molecules with binding affinity at least as good as ground truth, 3) **QED**, a quantitative estimate of druglikeness, 4) **SA**, synthetic accesibility, 5) **Diversity** defined by average Tanimoto dissimilarity for the generated molecules in the pocket, 6) **#Params**, the number of trainable parameters and 7) **Time** to generate 100 molecules in seconds. We include more molecular metrics in Appendix D. As baselines, we use three recent methods for SBDD: Pocket2Mol (Peng et al., 2022), DiffSBDD (Schneuing et al., 2022) and TargetDiff (Guan et al., 2023).

We report the results in Table 1. Clearly, all our approaches significantly outperform the previous state of the art, Pocket2Mol while being 20-1000x faster despite being run solely on CPU (and not on GPU unlike the baselines). Additionally, our method has only 23k trainable parameters, which is two orders of magnitude less than the baselines. We also note a decrease in diversity compared to these baselines: this is expected given that our model uses known unlabelled graphs during sampling.

Trade-off between QED and Binding Affinity prediction As seen in Equation (3), our method allows for controlling the binding affinity of the generated ligands. We therefore investigated the performance of the model when we vary the $v_{\min}, v_{\max}$ thresholds. Specifically, we evaluated 10 different versions of the unlabelled graph sampler with $(v_{\min}, v_{\max})$ set to $(q_0, q_5), (q_5, q_{10}), \ldots, (q_{45}, q_{50})$, where q_k is the k^{th} percentile of the predictions for $\mathcal{U}^M$ (note that these percentiles depend on the protein pocket). We generated atom types using the repurposing model described in section 3.3.1. We note a trade-off between QED and Binding Affinity of these models, but all of them are still better on both of these criteria than the samples from the CrossDocked2020 dataset. We show this in Figure 3.

The model generalizes across molecular datasets In our experiments we have used the ZINC250k dataset (Irwin et al., 2012) to train the scoring model (2), to sample unlabelled graphs (3) and to sample atom types in the drug repurposing model (5). We now check whether the model generalizes across datasets. To do that, we evaluate the FastSBDD–$\mathcal{DR}$ model with, but with the ChEMBL dataset (Mendez et al., 2018; Davies et al., 2015) without retraining the scoring model. We found that the model performs very similarly in terms of binding affinity. See Table 2 for details. We discuss model's generalization ability from a theoretical perspective in Section 5.2.

We now move on to studying the representational challenges and generalization of models for SBDD.

5 Theoretical Analysis

We showed empirically that very simple representations of the unlabelled graph perform remarkably well in terms of predicting binding affinity, and allow for significantly streamlined computation compared to much larger and more sophisticated models. Here we bring attention to the representational issues concerning large models. We show that despite adding considerable complexity, such models might have weaknesses in terms of what they can represent. While these limitations can possibly be circumvented with even more complex models, the expense in terms of their weaker

Table 2: **The model generalizes to other molecular datasets.** Comparison of the drug repurposing model with different sets of molecular structures. Both use the scoring model trained on the ZINC250k dataset (Irwin et al., 2012), but FastSBDD–$\mathcal{DR}$ (ZINC) uses ZINC250k also during sampling, while FastSBDD–$\mathcal{DR}$ (ChEMBL) uses ChEMBL (Mendez et al., 2018; Davies et al., 2015).

	Vina Score (kcal/mol, ↓)	High Affinity (↑)	QED (↑)	SA (↑)	Lipinski (↑)	LogP
Test set	-6.99 ± 2.16	-	0.48 ± 0.21	0.73 ± 0.14	4.34 ± 1.14	0.89 ± 2.73
FastSBDD–$\mathcal{DR}$ (ZINC)	-7.82 ± 1.47	0.72 ± 0.36	0.66 ± 0.15	0.78 ± 0.08	4.99 ± 0.05	2.97 ± 1.37
FastSBDD–$\mathcal{DR}$ (ChEMBL)	-8.03 ± 1.56	0.76 ± 0.33	0.56 ± 0.15	0.79 ± 0.08	4.99 ± 0.12	3.48 ± 0.95

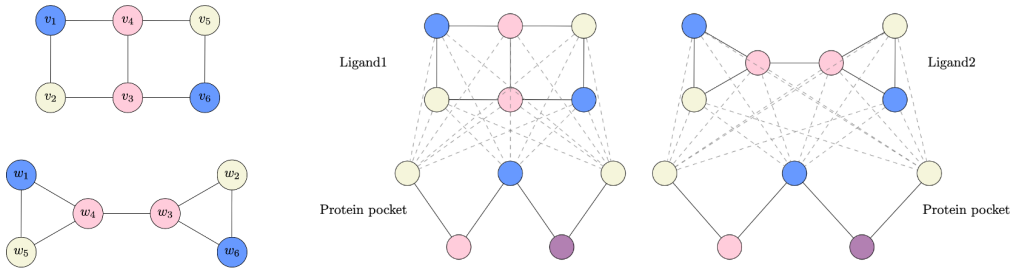


Figure 4: **Some ligand graphs cannot be distinguished by LU-GNNs even with additional protein context** Left: Construction for Lemma 1; Two non-isomorphic graphs differing in all properties stated in Lemma 1, but for which LU-GNNs produce identical embeddings. Right: Complex graphs constructed by joining the ligand graphs with the same protein graph remain identical whenever ligand graphs cannot be differentiated.

generalization ability (as known from learning theory results) could be excessive. Simpler, and likely less expressive, models like the ones considered in our work on the other hand might offer a better expressivity-generalization tradeoff and thus might be preferable from a theoretical perspective.

5.1 Representation limits of GNNs in SBDD

It is known that there exist graphs representing distinct molecular structures that cannot be distinguished using message passing graph neural networks (MPGNNs) (Sato, 2020; Garg et al., 2020). Following the notation in Garg et al. (2020), we consider *Locally Unordered* GNNs (LU-GNNs), which we summarize as follows. The updated embedding $h_v^{(l)}$ of node v at layer l is defined as

$$m_{u\to v}^{(l-1)} = \phi(h_u^{(l-1)}, e_{uv}); \tilde{h}_v^{(l-1)} = \text{AGG}\{m_{u\to v}^{(l-1)} \mid u \in N(v)\}; h_v^{(l)} = \text{COMBINE}\{h_v^{(l-1)}, \tilde{h}_v^{(l-1)}\},$$

where $N(v)$ denotes the set of neighbours of v, e_{uv} are edge features and ϕ is any function. It is known that there exist non-isomorphic graphs, for which LU-GNNs produce identical embeddings to every node at each layer (Garg et al., 2020). We now note that this result also holds for graphs embedded in 3D space. Let us introduce *Locally Unordered 3D* GNNs (LU3D-GNN) which aggregate embeddings for each node based on the embeddings of the neighbours, as well as their distance to these neighbours. Formally, in LU3D-GNNs, the messages are defined as:

$$m_{u\to v}^{(l-1)} = \phi\left(h_u^{(l-1)}, e_{uv}, \|x_u - x_v\|\right) \tag{7}$$

where x_v is the 3D position of v. We have the following result.

Lemma 1. *There exist connected non-isomorphic geometric graphs that differ in the number of conjoined cycles, girth, size of the largest cycle and cut-edges that LU3D-GNNs cannot distinguish.*

On the lefthand side of Figure 4, we show two graphs that are not isomorphic and differing in all mentioned properties, but which cannot be distinguished by LU3D-GNNs. For clarity of presentation, we presented the graphs in 2D, but any 3D configuration having all edges of equal length can be used. We prove that they cannot be distinguished by LU3D-GNNs in Appendix E.

In the context of SBDD, we are interested in modelling pairs of graphs corresponding to protein-ligand complexes. We now show that they also pose representational challenges for LU(3D)-GNNs.

Let $G_1 = (V_1, E_1), G_2 = (V_2, E_2)$ be any two graphs and $\mathcal{C}(G_1, G_2)$ denote a *complex* graph, i.e. $\mathcal{C}(G_1, G_2) = (V, E)$, where

$$V = V_1 \cup V_2 \text{ and } E = E_1 \cup E_2 \cup V_1 \times V_2 \tag{8}$$

and, importantly, features for all added edged $e_{uv} \in V_1 \times V_2$ only depend on the features of nodes at their endpoints. We have the following results.

Proposition 1.

(i) If G_1 and G_2 are indistinguishable for LU-GNNs, then for any graph P, the complex graphs $\mathcal{C}(P, G_1)$, $\mathcal{C}(P, G_2)$ are also indistinguishable for LU-GNNs.

(ii) There exist ligand graphs G_1, G_2 differing in graph properties listed in Lemma 1, such that for any protein P, the complexes $\mathcal{C}(P, G_1)$, $\mathcal{C}(P, G_2)$ are indistinguishable for models using LU3D-GNNs or LU-GNNs for intra-ligand and intra-protein message passing and LU-GNNs for inter ligand-protein message passing.

The righthand side of Figure 4 visualizes the proposed statement. The proof can be found in Appendix E. Proposition 1 shows that a scoring model defined in Equation (2) parametrized as a LU-GNN would not differentiate between certain ligand graphs, no matter what the protein context would be. This stems from the fact that we model unlabelled graphs without 3D coordinates and therefore cannot use LU3D-GNNs for inter ligand-protein message passing. We provide a real-world example of a pair of molecules that have different binding affinities to a specific protein pocket, but which are identical from LU-GNN perspective.

5.2 Generalization of the scoring model

We now delve into the theoretical analysis of the generalization capabilities of our proposed scoring model within FastSBDD. Understanding generalization is paramount, as it provides insight into how well our model can perform on unseen data, which is a fundamental aspect of its practical utility in structure-based drug design (SBDD). Our analysis is focused on deriving generalization bounds, which give a theoretical measure of the model's performance across different molecular structures and protein targets. We begin with defining the generalization error.

Definition 1. (Generalization error). *Let $f : \mathcal{X} \to \mathcal{Y}$ and $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ be a loss function. Let $S = \{(x_1, y_1), \ldots, (x_m, y_m)\} \subseteq \mathcal{X} \times \mathcal{Y}$ be a training sample. The generalization error of f is defined as the difference between expected loss and the sample loss:*

$$\mathcal{R}_S(f) = \mathbb{E}L(f(x), y) - \frac{1}{m}\sum_{i=1}^{m} L(f(x_i), y_i), \tag{9}$$

where the expectation is taken over (x, y) sampled from the underlying data distribution.

ε-soft normalized EGNN As described in Section 3.2, our scoring model first computes an embedding of the protein graph, then concatenates features of ligand's unlabelled graph and passes through an MLP. However, we use a slightly modified version of the EGNN model. As originally introduced, a layer of the EGNN network can be summarized as follows:

$$\mathbf{m}_{u\to v}^{(l-1)} = \phi_e\left(\mathbf{h}_u^{(l-1)}, \mathbf{h}_v^{(l-1)}, \|\mathbf{x}_u^{(l-1)} - \mathbf{x}_v^{(l-1)}\|, e_{uv}\right); \mathbf{m}_v^{(l-1)} = \sum_{u\in\mathcal{N}(v)} \mathbf{m}_{u\to v}^{(l-1)} \tag{10}$$

$$\mathbf{x}_v^{(l)} = \mathbf{x}_v^{(l-1)} + \frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)}\left(\mathbf{x}_v^{(l-1)} - \mathbf{x}_u^{(l-1)}\right)\phi_x(\mathbf{m}_{u\to v}^{(l-1)}) \tag{11}$$

$$\mathbf{h}_v^{(l)} = \phi_h(\mathbf{h}_v^{(l-1)}, \mathbf{m}_v^{(l-1)}), \tag{12}$$

where $\mathbf{h}_v^{(0)}$ and $\mathbf{x}_v^{(0)}$ are initialized with node's features and coordinates respectively and ϕ_x, ϕ_e, ϕ_h are learnable functions, typically MLPs. In our experiments, we replace (11) with

$$\mathbf{x}_v^{(l)} = \mathbf{x}_v^{(l-1)} + \frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)}\frac{\mathbf{x}_v^{(l-1)} - \mathbf{x}_u^{(l-1)}}{\|\mathbf{x}_v^{(l-1)} - \mathbf{x}_u^{(l-1)}\| + \varepsilon}\phi_x(\mathbf{m}_{u\to v}^{(l-1)}). \tag{13}$$

The authors include this version of the model in the official implementation[1], but do not use it in the experiments. With this formulation, we derive the following generalization bound:

Proposition 2 (Informal)**.** *Let f be the scoring model using ε-soft normalization in the EGNN network. Then for the sample of size m, the following holds*

$$\mathcal{R}_S(f) \leq \mathcal{O}\left(\frac{L^2+k}{\sqrt{m}}\sqrt{\log\left(\frac{m(L+k)}{\varepsilon}\right)}\right), \tag{14}$$

where L is the number of layers of the EGNN and k is the number of layers in the MLP.

We note that we make no assumptions about ε other than $\varepsilon > 0$. If generalization is the primary concern, one can set $\varepsilon = 1$.

Proof sketch. The key step is establishing that the model is Lipschitz continuous w.r.t. parameters and determining the Lipschitz constant. We show it by first proving that Lipschitz continuity is preserved under ε-soft normalization. We then use an inductive argument to derive recursive relationship for the constants and determine their growth rate as a function of the depth of the network. Once the Lipschitz constants are established, we leverage results from Learning theory relating the covering number of a function class to the empirical Rademacher complexity, which can subsequently be used to bound the geneneralization error. Please see Appendix F for a detailed formulation and proof. □

Impact of ε-soft normalization Interestingly, when analyzing the generalization bounds of normalized EGNN model vis-à-vis the original formulation, we observe an important difference. Our derivations indicate that, in the absence of normalization, the original EGNN model would exhibit significantly less favorable generalization bounds. Specifically, using the same proof technique, the bound for the EGNN without normalization would be exponential in L as opposed to polynomial. Please see Appendix F.5 for details. This discovery underscores the importance of the normalization component, emphasizing its benefits in both numerical stability and theoretical generalizability.

6 Conclusion, Broader Impact and Limitations

Our research brings to the forefront efficient methodologies for Structure-Based Drug Design (SBDD), advocating for a shift in focus from mere model expressivity to robust generalization. The successes of FastSBDD serve as a testament to the significance of integrating metrics of interest into the optimization process. The unprecedented efficiency and success of FastSBDD not only highlights the potential to reshape the approach to SBDD but also paves the way for streamlining docking software, especially when assessed with widely adopted tools like Vina and Gnina.

However, there are limitations to our findings. While we rely on Vina, the gold standard for evaluation in the SBDD realm, the ideal validation would entail wet lab experiments or more sophisticated molecular dynamics simulations. However, such methods are often prohibitively expensive. Additionally, the very nature of FastSBDD as a generative model, especially with its property optimization capabilities, brings with it risks. As SBDD models become increasingly adept at designing molecules with specific properties, we must remain vigilant of the potential unintended outcomes, since streamlining the design process could accelerate the design of harmful biochemicals.

Reproducibility statement All our experiments are reproducible. We will share our code as well as the trained models under the MIT License on GitHub upon the acceptance of the paper.

References

Gustaf Ahdritz, Nazim Bouatta, Sachin Kadyan, Qinghui Xia, William Gerecke, Timothy J O'Donnell, Daniel Berenberg, Ian Fisk, Niccolò Zanichelli, Bo Zhang, Arkadiusz Nowaczynski, Bei Wang, Marta M Stepniewska-Dziubinska, Shang Zhang, Adegoke Ojewole, Murat Efe Guney, Stella Biderman, Andrew M Watkins, Stephen Ra, Pablo Ribalta Lorenzo, Lucas Nivon, Brian Weitzner, Yih-En Andrew Ban, Peter K Sorger, Emad Mostaque, Zhao Zhang, Richard Bonneau, and

[1] https://github.com/vgsatorras/egnn

Mohammed AlQuraishi. Openfold: Retraining alphafold2 yields new insights into its learning mechanisms and capacity for generalization. *bioRxiv*, 2022.

Amr Alhossary, Stephanus Daniel Handoko, Yuguang Mu, and Chee-Keong Kwoh. Fast, accurate, and reliable molecular docking with QuickVina 2. *Bioinformatics*, 31(13):2214–2216, 2015.

Peter L Bartlett, Dylan J Foster, and Matus J Telgarsky. Spectrally-normalized margin bounds for neural networks. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (eds.), *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/b22b257ad0519d4500539da3c8bcf4dd-Paper.pdf.

Camille Bilodeau, Wengong Jin, Tommi Jaakkola, Regina Barzilay, and Klavs F Jensen. Generative models for molecular discovery: Recent advances and challenges. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 12(5):e1608, 2022.

Hongming Chen, Ola Engkvist, Yinhai Wang, Marcus Olivecrona, and Thomas Blaschke. The rise of deep learning in drug discovery. *Drug discovery today*, 23(6):1241–1250, 2018.

Minshuo Chen, Xingguo Li, and Tuo Zhao. On generalization bounds of a family of recurrent neural networks. In Silvia Chiappa and Roberto Calandra (eds.), *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, volume 108 of *Proceedings of Machine Learning Research*, pp. 1233–1243. PMLR, 26–28 Aug 2020. URL https://proceedings.mlr.press/v108/chen20d.html.

Gabriele Corso, Hannes Stärk, Bowen Jing, Regina Barzilay, and Tommi S. Jaakkola. Diffdock: Diffusion steps, twists, and turns for molecular docking. In *International Conference on Learning Representations*, 2023.

Mark Davies, Michał Nowotka, George Papadatos, Nathan Dedman, Anna Gaulton, Francis Atkinson, Louisa Bellis, and John P. Overington. ChEMBL web services: streamlining access to drug discovery data and utilities. *Nucleic Acids Research*, 43(W1):W612–W620, 2015.

R. Fletcher. *Newton-Like Methods*, chapter 3, pp. 44–79. John Wiley & Sons, 2000.

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, 2020.

Octavian-Eugen Ganea, Xinyuan Huang, Charlotte Bunne, Yatao Bian, Regina Barzilay, Tommi S. Jaakkola, and Andreas Krause. Independent SE(3)-equivariant models for end-to-end rigid protein docking. In *International Conference on Learning Representations*, 2022.

Vikas K. Garg, Stefanie Jegelka, and Tommi Jaakkola. Generalization and representational limits of graph neural networks. In *International Conference on Machine Learning*, 2020.

Rafael Gómez-Bombarelli, Jennifer N Wei, David Duvenaud, José Miguel Hernández-Lobato, Benjamín Sánchez-Lengeling, Dennis Sheberla, Jorge Aguilera-Iparraguirre, Timothy D Hirzel, Ryan P Adams, and Alán Aspuru-Guzik. Automatic chemical design using a data-driven continuous representation of molecules. *ACS central science*, 4(2):268–276, 2018.

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 *International Conference on Learning Representations*, 2023.

William L. Hamilton, Rex Ying, and Jure Leskovec. Inductive representation learning on large graphs. In *International Conference on Neural Information Processing Systems*, 2017.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Brian Hie, Salvatore Candido, Zeming Lin, Ori Kabeli, Roshan Rao, Nikita Smetanin, Tom Sercu, and Alexander Rives. A high-level programming language for generative protein design. *bioRxiv*, 2022.

Emiel Hoogeboom, Víctor Garcia Satorras, Clément Vignac, and Max Welling. Equivariant diffusion for molecule generation in 3D. In *International Conference on Machine Learning*, 2022.

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*, 2022.

John Ingraham, Max Baranov, Zak Costello, Vincent Frappier, Ahmed Ismail, Shan Tie, Wujie Wang, Vincent Xue, Fritz Obermeyer, Andrew Beam, and Gevorg Grigoryan. Illuminating protein space with a programmable generative model. *bioRxiv*, 2022.

John J. Irwin, Teague Sterling, Michael M. Mysinger, Erin S. Bolstad, and Ryan G. Coleman. Zinc: A free tool to discover chemistry for biology. *Journal of Chemical Information and Modeling*, 52 (7):1757–1768, 2012.

Stefanie Jegelka. Theory of graph neural networks: Representation and learning. *ArXiv*, abs/2204.07697, 2022.

Wengong Jin, Regina Barzilay, and Tommi Jaakkola. Junction tree variational autoencoder for molecular graph generation. In *International Conference on Machine Learning*, 2018.

Wengong Jin, Kevin Yang, Regina Barzilay, and T. Jaakkola. Learning multimodal graph-to-graph translation for molecular optimization. In *International Conference on Learning Representations*, 2019.

Thorsten Joachims. A support vector method for multivariate performance measures. In Luc De Raedt and Stefan Wrobel (eds.), *Proceedings of the 22nd International Conference on Machine Learning (ICML 2005), August 7-11, 2005, Bonn, Germany*, pp. 377–384. ACM Press, New York, NY, USA, 2005. ISBN 1-59593-180-5. URL http://doi.acm.org/10.1145/1102351.1102399.

Chaitanya K. Joshi, Cristian Bodnar, Simon V Mathis, Taco Cohen, and Pietro Lio. On the expressive power of geometric graph neural networks. 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. 15330–15355. PMLR, 23–29 Jul 2023. URL https://proceedings.mlr.press/v202/joshi23a.html.

Haotian Ju, Dongyue Li, Aneesh Sharma, and Hongyang R. Zhang. Generalization in Graph Neural Networks: Improved PAC-Bayesian Bounds on Graph Diffusion, June 2023.

John M. Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Zídek, Anna Potapenko, Alex Bridgland, Clemens Meyer, Simon A A Kohl, Andy Ballard, Andrew Cowie, Bernardino Romera-Paredes, Stanislav Nikolov, Rishub Jain, Jonas Adler, Trevor Back, Stig Petersen, David A. Reiman, Ellen Clancy, Michal Zielinski, Martin Steinegger, Michalina Pacholska, Tamas Berghammer, Sebastian Bodenstein, David Silver, Oriol Vinyals, Andrew W. Senior, Koray Kavukcuoglu, Pushmeet Kohli, and Demis Hassabis. Highly accurate protein structure prediction with alphafold. *Nature*, 596:583–589, 2021.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference on Learning Representations*, 12 2014.

Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *International Conference on Learning Representations*, 2017.

Greg Landrum, Paolo Tosco, Brian Kelley, sriniker, gedeck, NadineSchneider, Riccardo Vianello, Ric, Andrew Dalke, Brian Cole, AlexanderSavelyev, Matt Swain, Samo Turk, Dan N, Alain Vaucher, Eisuke Kawashima, Maciej Wójcikowski, Daniel Probst, guillaume godin, David Cosgrove, Axel Pahl, JP, Francois Berenger, strets123, JLVarjo, Noel O'Boyle, Patrick Fuller, Jan Holst Jensen, Gianluca Sforna, and Doliath Gavid. Rdkit, 2020.

Yibo Li, Jianfeng Pei, and Luhua Lai. Structure-based de novo drug design using 3d deep generative models. *Chemical Sciences*, 12:13664–13675, 2021.

Shitong Luo, Jiaqi Guan, Jianzhu Ma, and Jian Peng. A 3d generative model for structure-based drug design. In *Advances in Neural Information Processing Systems*, 2021a.

Youzhi Luo, Keqiang Yan, and Shuiwang Ji. GraphDF: A discrete flow model for molecular graph generation. In *International Conference on Machine Learning*, 2021b.

Andrew McNutt, Paul Francoeur, Rishal Aggarwal, Tomohide Masuda, Rocco Meli, Matthew Ragoza, Jocelyn Sunseri, and David Koes. Gnina 1.0: molecular docking with deep learning. *Journal of Cheminformatics*, 13, 06 2021. doi: 10.1186/s13321-021-00522-2.

David Mendez, Anna Gaulton, A Patrícia Bento, Jon Chambers, Marleen De Veij, Eloy Félix, María Paula Magariños, Juan F Mosquera, Prudence Mutowo, Michał Nowotka, María Gordillo-Marañón, Fiona Hunter, Laura Junco, Grace Mugumbate, Milagros Rodriguez-Lopez, Francis Atkinson, Nicolas Bosc, Chris J Radoux, Aldo Segura-Cabrera, Anne Hersey, and Andrew R Leach. ChEMBL: towards direct deposition of bioassay data. *Nucleic Acids Research*, 47(D1): D930–D940, 2018.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. *Foundations of Machine Learning*. The MIT Press, 2012. ISBN 9780262018258. URL http://www.jstor.org/stable/j.ctt5hhcw1.

Mohit Pandey, Michael Fernandez, Francesco Gentile, Olexandr Isayev, Alexander Tropsha, Abraham C Stern, and Artem Cherkasov. The transformational role of GPU computing and deep learning in drug discovery. *Nature Machine Intelligence*, 4(3):211–221, 2022.

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 *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.

Xingang Peng, Shitong Luo, Jiaqi Guan, Qi Xie, Jian Peng, and Jianzhu Ma. Pocket2Mol: Efficient molecular sampling based on 3D protein pockets. In *International Conference on Machine Learning*, 2022.

Sudeep Pushpakom, Francesco Iorio, Patrick A Eyers, K Jane Escott, Shirley Hopper, Andrew Wells, Andrew Doig, Tim Guilliams, Joanna Latimer, Christine McNamee, et al. Drug repurposing: progress, challenges and recommendations. *Nature reviews Drug discovery*, 18(1):41–58, 2019.

Matthew Ragoza, Tomohide Masuda, and David Koes. Generating 3d molecules conditional on receptor binding sites with deep generative models. *Chemical Science*, 13, 2022.

Ryoma Sato. A survey on the expressive power of graph neural networks. *arXiv*, 2020.

Víctor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E(n) equivariant graph neural networks. In *International Conference on Machine Learning*, 2021.

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. *arXiv*, 2022.

Chence Shi, Minkai Xu, Zhaocheng Zhu, Weinan Zhang, Ming Zhang, and Jian Tang. Graphaf: a flow-based autoregressive model for molecular graph generation. In *International Conference on Learning Representations*, 2020.

Chence Shi, Chuanrui Wang, Jiarui Lu, Bozitao Zhong, and Jian Tang. Protein sequence and structure co-design with equivariant translation. In *International Conference on Learning Representations*, 2023.

Hannes Stärk, Dominique Beaini, Gabriele Corso, Prudencio Tossou, Christian Dallago, Stephan Günnemann, and Pietro Lió. 3D infomax improves GNNs for molecular property prediction. In *International Conference on Machine Learning*, 2022a.

Hannes Stärk, Octavian Ganea, Lagnajit Pattanaik, Dr.Regina Barzilay, and Tommi Jaakkola. EquiBind: Geometric deep learning for drug binding structure prediction. In *International Conference on Machine Learning*, 2022b.

Martin Steinegger and Johannes Söding. Mmseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nature Biotechnology*, 35, 2017.

Oleg Trott and Arthur J. Olson. Autodock vina: Improving the speed and accuracy of docking with a new scoring function, efficient optimization, and multithreading. *Journal of Computational Chemistry*, 31(2):455–461, 2010.

Jessica Vamathevan, Dominic Clark, Paul Czodrowski, Ian Dunham, Edgardo Ferran, George Lee, Bin Li, Anant Madabhushi, Parantu Shah, Michaela Spitzer, et al. Applications of machine learning in drug discovery and development. *Nature reviews Drug discovery*, 18(6):463–477, 2019.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph attention networks. In *International Conference on Learning Representations*, 2018.

Robert Verkuil, Ori Kabeli, Yilun Du, Basile I. M. Wicky, Lukas F. Milles, Justas Dauparas, David Baker, Sergey Ovchinnikov, Tom Sercu, and Alexander Rives. Language models generalize beyond natural proteins. *bioRxiv*, 2022.

Yogesh Verma, Samuel Kaski, Markus Heinonen, and Vikas Garg. Modular flows: Differential molecular generation. In *Advances in Neural Information Processing Systems*, 2022.

Yogesh Verma, Markus Heinonen, and Vikas Garg. Abode: Ab initio antibody design using conjoined odes. In *International Conference on Machine Learning*, 2023.

Joseph L. Watson, David Juergens, Nathaniel R. Bennett, Brian L. Trippe, Jason Yim, Helen E. Eisenach, Woody Ahern, Andrew J. Borst, Robert J. Ragotte, Lukas F. Milles, Basile I. M. Wicky, Nikita Hanikel, Samuel J. Pellock, Alexis Courbet, William Sheffler, Jue Wang, Preetham Venkatesh, Isaac Sappington, Susana Vázquez Torres, Anna Lauko, Valentin De Bortoli, Emile Mathieu, Regina Barzilay, Tommi S. Jaakkola, Frank DiMaio, Minkyung Baek, and David Baker. Broadly applicable and accurate protein design by integrating structure prediction networks and diffusion generative models. *bioRxiv*, 2022.

Kevin E. Wu, Kevin K. Yang, Rianne van den Berg, James Y. Zou, Alex X. Lu, and Ava P. Amini. Protein structure generation via folding diffusion. *arXiv*, 2022a.

Ruidong Wu, Fan Ding, Rui Wang, Rui Shen, Xiwen Zhang, Shitong Luo, Chenpeng Su, Zuofan Wu, Qi Xie, Bonnie Berger, Jianzhu Ma, and Jian Peng. High-resolution de novo structure prediction from primary sequence. *bioRxiv*, 2022b.

Chengxi Zang and Fei Wang. Moflow: An invertible flow model for generating molecular graphs. In *ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020.

A DECOUPLING THE UNLABELLED MOLECULAR GRAPH FROM ATOM TYPES

Here we provide details of experiments we discuss in section 3.1.

A.1 VINA SCORE

Vina uses a scoring function, which approximates the experimentally measured binding affinity with several types of interactions between the protein and ligand atoms (Trott & Olson, 2010):

$$f(G^P, G^M) = \sum_{k=1}^{K} w_k \sum_{i=1}^{N_P} \sum_{j=1}^{N_M} f_k\Big(a_i^P, a_j^M, \mathbf{s}_i^P, \mathbf{s}_j^M\Big), \tag{15}$$

where $\{f_k\}_{k=1}^K$ represent K types of interatomic interactions that depend on atom types a and 3D coordinates $\mathbf{s}$. The weights $\{w_k\}_{k=1}^K$ are pretrained to reflect experimentally measured affinities. We emphasize that *lower values of f are interpreted as stronger binding.* To estimate the binding affinity for a given protein-ligand complex (G^P, G^M), Vina performs a local search for the lowest-score 3D configuration of the molecule. Specifically, gradient optimization is performed over global translations, global rotations, and torsional rotations $\mathcal{T}(G^M)$ of the ligand G^M to optimise its fit to the protein pocket:

$$\text{Vina}(G^P, G^M) = \min\Big\{f(G^P, \hat{G}^M) : \hat{G}^M \in \mathcal{T}(G^M)\Big\}. \tag{16}$$

Vina uses the BFGS optimiser (Fletcher, 2000).

A.2 INFLUENCE OF UNLABELLED MOLECULAR GRAPHS ON VINA SCORES

Equation (16) indicates that if Vina employed an ideal optimizer to compute the global minimum, the initial 3D configuration of the molecule would not influence the outcome. Yet, because Vina utilizes an approximate gradient-based optimization method, variations in the initial 3D configuration can indeed affect the Vina score. To measure that, we randomly selected 1,000 protein-ligand complexes from the CrossDocked2020 dataset (Francoeur et al., 2020) and computed the Vina score both before and after modifying the ligand. Here, by 'modifying the ligand', we refer to replacing its 3D configuration with a randomly generated conformer using RDKit. As expected, a high $R^2 = 0.94$ emerged indicating insignificant impact of the ligand's initial 3D configuration on the Vina score (See Figure 5a).

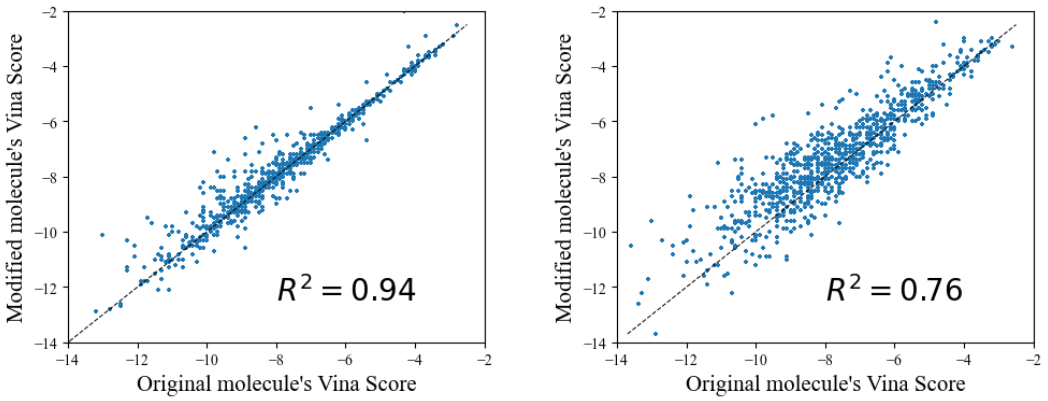
Furthermore, we experimented with changing ligand's atom types. For a fixed unlabelled graph U^M, we re-sample random atom assignments $\mathbf{a}^M$ from the empirical atom distributions, while avoiding valence violations (See Appendix A.3). This procedure changes the molecule and therefore its 3D configuration is most likely no longer energetically valid. We therefore use RDKit to generate a random conformer of the new molecule and use it to replace the original 3D coordinates. Surprisingly, the R^2 coefficient remains relatively high at $R^2 = 0.76$ (see Figure 5b). These findings suggest that using just the unlabelled graph of the ligand, one can estimate the Vina score with considerable accuracy. We use this observation in the definition of the model.

A.3 SAMPLING VALID MOLECULES FOR A GIVEN UNLABELLED GRAPH

Here we provide details on how to randomly modify atom types conditioned on a molecular graph's structure to obtain valid molecules. Simply assigning random atom types independently to each node usually yields invalid molecules due to violated valence constraints. Whether assigning a specific atom type to a given node v_i^M results in a violated valence constraint depends on how many and what kind of bonds are connected to it. Specifically, for a bond e_{ij}, let $o(e_{ij})$ denote its multiplicity, i.e. $o(e_{ij}) = 1$ for a single bond, $o(e_{ij}) = 2$ for a double bond etc. Let now

$$\deg(v_i^M) = \sum_{j:e_{ij}\in U^M} o(e_{ij}) \tag{17}$$

denote i-th node's degree. Node's degree determines what atom types satisfy valence constraints for that node. It is therefore sufficient to determine the degrees of all nodes in the unlabelled graph and



(a) Changing initial 3D configuration.

(b) Changing atom types and initial 3D configuration

Figure 5: Impact of different transformations of the ligand on the Vina score.

independently sample atom types for each node conditioned on its degree. Formally, we define the distribution over atom types given the molecule's unlabelled graph as

$$p(\mathbf{a}^M | U^M) := \prod_{i=1}^{N_M} p_{\text{emp}}(a_i^M | \deg(v_i^M)), \tag{18}$$

where $p_{\text{emp}}(a_i^M | \deg(v_i^M))$ is the empirical distribution of atom types for a given degrees of the nodes estimated using the CrossDocked2020 dataset (Francoeur et al., 2020). This allows for efficient sampling of valid molecules with a fixed unlabelled graph U^M so that the distribution of atom type counts matches the empirical one.

A.4 Justification of the model decomposition - Gnina

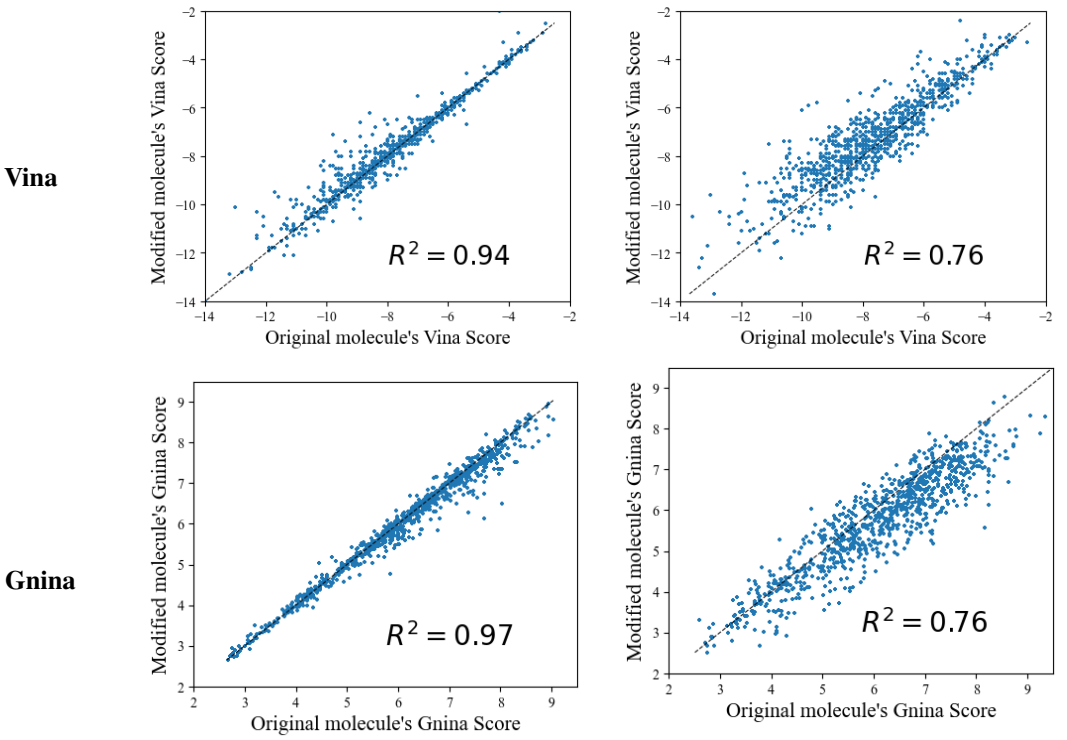
In section 3.1, we justified our model decomposition choice through empirical evidence indicating that the unlabeled molecular graph inherently contains substantial information, as reflected in the Vina score, even when the ligand's initial 3D configuration and atom type assignment are varied. We now show that a similar phenomenon can be observed when, instead of Vina, a reportedly significantly more accurate binding software Gnina (McNutt et al., 2021) is used.

We repeat the experiments from the section A.2 with Gnina instead of Vina and find that for changing the initial 3D configuration, there is even smaller impact indicated by $R^2 = 0.97$. For changing atom type assignments, $R^2 = 0.76$ as it was for Vina. Please see Figure 6 for details.

B Binding affinity approximation model details

In this section we provide more details of the scoring model from Eq. (2). Recall that we want to approximate the binding affinity estimation defined in Eq. (16) using the protein pocket information and ligand's unlabelled graph structure. To that end we represent the protein pocket as a graph with residues as nodes. Each node is equipped with a one-hot encoding of a residue type (1 of 20 possible amino acids) as a feature vector and 3D coordinates computed as a center of mass of the corresponding residue. Two nodes are connected with an edge if they are at most 15Å apart and we limit each node to have at most 24 neighbours. We do not use any edge features. To compute the protein pocket embedding, we use E(3)-Equivariant Graph Neural Networks (EGNNs) (Satorras et al., 2021).

An EGNN layer takes as input a set of node embeddings $\{\mathbf{h}_v^{(l-1)}\}_{v \in V}$, coordinate embeddings $\{\mathbf{x}_v^{(l-1)}\}_{v \in V}$ and edge information $\{e_{uv}\}$ and computes output embeddings $\{\mathbf{h}_v^{(l)}\}_{v \in V}$ $\{\mathbf{x}_v^{(l)}\}_{v \in V}$



(a) Changing initial 3D configuration

(b) Changing atom types and initial 3D configuration

Figure 6: The impact of molecule modifications on binding scores is consistent across Vina and Gnina software.

for nodes and coordinates respectively. One layer operation can be summarized as follows

$$\begin{aligned}
\mathbf{m}_{u\to v}^{(l-1)} &= \phi_e\left(\mathbf{h}_u^{(l-1)}, \mathbf{h}_v^{(l-1)}, \|\mathbf{x}_u^{(l-1)} - \mathbf{x}_v^{(l-1)}\|, e_{uv}\right) \\
\mathbf{x}_v^{(l)} &= \mathbf{x}_v^{(l-1)} + C\sum_{u\neq v}\left(\mathbf{x}_v^{(l-1)} - \mathbf{x}_u^{(l-1)}\right)\phi_x(\mathbf{m}_{u\to v}^{(l-1)}) \\
\mathbf{m}_v^{(l-1)} &= \sum_{u\neq v}\mathbf{m}_{u\to v}^{(l-1)} \\
\mathbf{h}_v^{(l)} &= \phi_h(\mathbf{h}_v^{(l-1)}, \mathbf{m}_v^{(l-1)}),
\end{aligned} \tag{19}$$

where ϕ_e, ϕ_x and ϕ_h are learnable functions and $\{\mathbf{h}_v^{(0)}\}_{v\in V}$ are initialized to input node features, i.e. one-hot encodings of amino acid types and $\{\mathbf{x}_v^{(0)}\}_{v\in V}$ are initialized with 3D coordinates of residues' centers of mass. Alternatively, one can use a version of the model with "ε-soft normalization", i.e. replace the coordinate embedding update with:

$$\mathbf{x}_v^{(l)} = \mathbf{x}_v^{(l-1)} + C\sum_{u\neq v}\frac{\mathbf{x}_v^{(l-1)} - \mathbf{x}_u^{(l-1)}}{\|\mathbf{x}_v^{(l-1)} - \mathbf{x}_u^{(l-1)}\| + \varepsilon}\phi_x(\mathbf{m}_{u\to v}^{(l-1)}). \tag{20}$$

We used the normalized version in our experiments. Please see Section 5.2 and Appendix F.5 for a discussion of normalization impact on generalization. We encode the protein as the average embedding of the nodes after L layers:

$$\mathbf{h}_P = \frac{1}{|V|}\sum_{v\in V}\mathbf{h}_v^{(L)}, \tag{21}$$

where we use $L = 3$ in our experiments. ϕ_e, ϕ_x and ϕ_h are all implemented as concatenation of the input followed by a fully connected 2-layer MLP with hidden and output dimensions of 16 and the SiLU activation function (Hendrycks & Gimpel, 2016).

We represent the ligand's unlabelled graph structure with the following four features:

- number of nodes,
- number of rings,
- number of rotatable bonds and
- graph diameter.

All these features are standardized by subtracting the mean and dividing by the standard deviation, where these statistics were computed on the train set. The standardized ligand features are subsequently concatenated with $\mathbf{h}_P$ and passed through 5 layers of a fully connected MLP with 50 hidden units, ReLU activation functions and 1 output unit. The model has a total of 15k trainable parameters.

B.1 Training details

The dataset for training the scoring model consists of triples (G^P, G^M, y) representing the protein, ligand and the binding affinity estimated with Vina respectively. We randomly select 10% of the protein pockets for validation and the remaining 90% for training. The model is trained to minimize the mean squared error with the Adam optimizer (Kingma & Ba, 2014) with default parameters of its PyTorch implementation (Paszke et al., 2019). The model was trained with batch size 128 for 10000 steps, after which we did not observe improvements in the validation loss. Model training took approximately 45 minutes on an Apple M1 CPU.

C Ligand's center of mass predictor

As mentioned in Section 3.3, in order to accurately estimate the binding affinity for a protein-ligand pair with Vina, the ligand's 3D configuration must be in the vicinity of the protein 3D configuration. The reason for that is that the scoring function defined in Eq. (15) vanishes for atom pairs that are far apart (Trott & Olson, 2010). Consequently, gradient based optimization used to find the best 3D configuration (Eq. (16)) will return the original configuration, because all gradients are zero whenever the protein and ligand are too far apart.

To that end, we train a model, which predicts the center of mass of ligand's 3D configuration based on the protein pocket. We subsequently use the model to place the ligands' 3D configurations computed with RDKit at the predicted center of mass. As model parametrization we use the EGNN as described in Appendix B. The model output is defined as:

$$\hat{\mathbf{x}} = \frac{1}{|V|} \sum_{v \in V} \mathbf{x}_v^{(L)}, \tag{22}$$

where $\mathbf{x}_v^{(L)}$ is defined in Eq. (53). As shown in (Satorras et al., 2021), $\hat{\mathbf{x}}$ is then equivariant to translations and rotations as required for the center of mass predictor model. In our experiments, we used $L = 4$ layers and a hidden dimension of 16. The model has 8k trainable parameters and *both its number of parameters and runtime were included when reporting the results for FastSBDD models in Table 1*.

To train the model, we used the training protein-ligand pairs of the CrossDocked2020 (Francoeur et al., 2020) dataset described in Section 4. We further split them into train and validation sets to monitor overfitting using a 85-15 train-validation split. Similarly to (Peng et al., 2022), the split was performed to ensure that all validation protein pockets will have sequence similarity lower than 30% to all training protein pockets.

The model was trained with Adam optimizer (Kingma & Ba, 2014) with default parameters of its PyTorch implementation (Paszke et al., 2019) to minimize the Euclidean distance between the predicted and ground truth centers of mass. We used a batch size of 64 and trained the model for 10000 steps until the validation loss converged to ~ 1.15Å. Training took approximately 25 minutes on an Apple M1 CPU.

D ADDITIONAL MOLECULAR METRICS

Here we provide additional metrics to complement the ones reported in Table 1, which were omitted for presentation clarity. Namely, we include the Lipinski's Rule of Five and LogP, which were defined in Section 2. For Lipinski's Rule of Five, 5 is the highest possible value. For LogP we do not indicate which method is the best as there is no straightforward criterion to compare LogP values. In (Peng et al., 2022) it is noted that values in the $(-0.4, 5.6)$ interval are considered good drug candidates. Note that for FastSBDD methods, the value of LogP is the closest to the midpoint of that interval (2.6).

Table 3: Additional molecular metrics.

	Lipinski (↑)	LogP
Test set	4.34 ± 1.14	0.89 ± 2.73
DiffSBDD	4.80 ± 0.49	−0.33 ± 1.18
Pocket2Mol	4.90 ± 0.35	1.71 ± 1.98
TargetDiff	4.59 ± 0.83	1.37 ± 2.37
FastSBDD–$\mathcal{DR}$	**5.00** ± 0.05	2.97 ± 1.37
FastSBDD–$\mathcal{ND}$	4.96 ± 0.20	3.33 ± 1.49
FastSBDD–$\mathcal{PO}$	**5.00** ± 0.03	3.24 ± 0.79

E REPRESENTATION LIMITS OF GNNS FOR SBDD

In this section we prove claims made in Section 5.1. We begin with the notion of indistinguishability for LU(3D)-GNNs. Recall that a layer of a LU-GNN network can be summarized as follows

$$\begin{aligned} m_{u\to v}^{(l-1)} &= \phi(h_u^{(l-1)}, e_{uv}) \\ \tilde{h}_v^{(l-1)} &= \text{AGG}\{m_{u\to v}^{(l-1)} \mid u \in N(v)\} \\ h_v^{(l)} &= \text{COMBINE}\{h_v^{(l-1)}, \tilde{h}_v^{(l-1)}\}, \end{aligned} \tag{23}$$

where $h_v^{(l)}$ is the feature vector of node v after applying l layers and $h_v^{(0)} = h_v$ are v's input features. A layer of a LU3D-GNN is defined analogously with messages depending on distances to neighbouring nodes:

$$m_{u\to v}^{(l-1)} = \phi(h_u^{(l-1)}, e_{uv}, \|x_u - x_v\|), \tag{24}$$

where x_v are 3D coordinates of v. Before we discuss indistinguishability, we formally define it.

Definition 2 (Indistinguishability). *We say that graphs $G_1 = (V, E_1)$, $G_2 = (W, E_2)$ are indistinguishable by LU(3D)-GNNs if there exist node orderings $(v_1, v_2, \ldots, v_N)$ and $(w_1, w_2, \ldots, w_N)$ of V and W respectively s.t. for any choice of ϕ, AGG, COMBINE and network depth L:*

$$h_{v_i}^{(l)} = h_{w_i}^{(l)} \quad \forall i = 1, \ldots, N, l = 1, \ldots, L$$

We denote indistinguishability by $G_1 \equiv G_2$ for LU-GNNs and by $G_1 \equiv_{3D} G_2$ for LU3D-GNNs.

In other words, G_1, G_2 are not distinguishable by LU(3D)-GNNs, if we can pair up their nodes in such a way that corresponding embeddings remain identical in both graphs after any number of layers of any LU(3D)-GNN. We first note that, since LU3D-GNNs are strictly more expressive than LU-GNNs, it holds that:

$$G_1 \equiv_{3D} G_2 \implies G_1 \equiv G_2. \tag{25}$$

A useful construct for determining indistinguishability is that of computation trees (Jegelka, 2022):

Definition 3 (Computation trees). *For a graph $G = (V, E)$, the computation tree of node v is defined recursively as follows:*

1. *$T_G^{(1)}(v) = h_v$, where h_v is v's feature vector,*
2. *for $l \geq 2$, the root of $T_G^{(l)}(v)$ is h_v, its children are $\{T_G^{(l-1)}(u) \mid u \in N(v)\}$ and the edge connecting the root to $T_G^{(l-1)}(u)$ inherits features e_{vu} from G.*

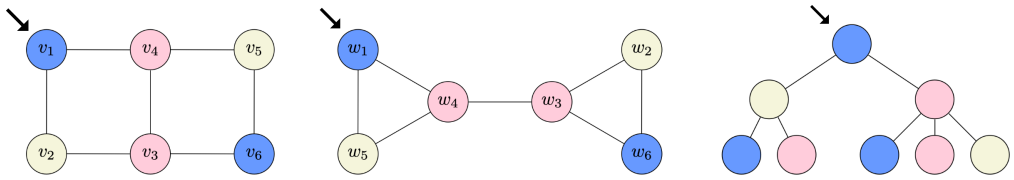


Figure 7: **Construction for Lemma 1.** Left and Middle: A pair of non-isomorphic connected graphs differing in all graph properties listed in Lemma 1. Right: An example computation tree of depth three, which is identical for nodes v_1 and w_1.

We denote the multiset of depth-L computation trees of G as $\{T_G^{(L)}(v)\}_{v\in V}$.

Since the multiset of depth-L computation trees represents all information LU-GNNs can extract (Jegelka, 2022), we see that

$$G_1 \equiv G_2 \Leftrightarrow \forall_L \ \{T_{G_1}^{(L)}(v)\}_{v\in V_1} = \{T_{G_2}^{(L)}(v)\}_{v\in V_2} \tag{26}$$

and analogously for $\equiv_{3D}$ whenever the edges of the computation trees carry the information about the distances between nodes. Using the computation tree terminology, we can now prove Lemma 1. We recall it for completeness:

Lemma 1. *There exist connected non-isomorphic geometric graphs that differ in the number of conjoined cycles, girth, size of the largest cycle and cut-edges that LU3D-GNNs cannot distinguish.*

Proof. In Figure 7 (left and middle), we provide an example of a pair of non-isomorphic connected graphs, which differ in all mentioned graph properties. Colors indicate feature vectors: nodes with the same color have identical features. Edge features can be defined as any function of their endpoints, i.e. $e_{uv} = f(h_u, h_v)$ for any f. All edges are of equal length. For clarity of presentation, the graphs are presented in 2D, but any 3D configuration preserving equal edge lengths could be used instead.

In the figure, we show node orderings such that $T_{G_1}^{(l)}(v_i) = T_{G_2}^{(l)}(w_i)$ for all $i = 1, \ldots, 6$ and $l \geq 1$. As an example, we show $T_{G_1}^{(3)}(v_1)$ in Figure 7 (Right), which is identical to $T_{G_2}^{(3)}(w_1)$. □

We now move on to proving Proposition 1. We recall it for completeness:

Proposition 1.

(i) If G_1 and G_2 are indistinguishable for LU-GNNs, then for any graph P, the complex graphs $\mathcal{C}(P, G_1)$, $\mathcal{C}(P, G_2)$ are also indistinguishable for LU-GNNs.

(ii) There exist ligand graphs G_1, G_2 differing in graph properties listed in Lemma 1, such that for any protein P, the complexes $\mathcal{C}(P, G_1)$, $\mathcal{C}(P, G_2)$ are indistinguishable for models using LU3D-GNNs or LU-GNNs for intra-ligand and intra-protein message passing and LU-GNNs for inter ligand-protein message passing.

Proof. We begin with proving (*i*). We will abuse notation slightly and for a graph $G = (V, E)$ write $v \in V$ and $v \in G$ interchangeably. We will also write $N_G(v)$ to denote the neighbourhood of v in G whenever we want to emphasize which graph we are considering. Let $C_i = \mathcal{C}(P, G_i)$ for notation brevity and let $(v_1, \ldots, v_N)$ and $(w_1, \ldots, w_N)$ be node orderings of G_1 and G_2 respectively such that $T_{G_1}^{(l)}(v_i) = T_{G_2}^{(l)}(w_i)$, for all $i = 1, \ldots, N$ and $l \geq 1$. We can find such an ordering, because $G_1 \equiv G_2$. Let now $(p_1, \ldots, p_K)$ be any ordering of nodes of P. We will show that for $(u_1, \ldots, u_{N+K}) = (p_1, \ldots, p_K, v_1, \ldots, v_N)$ and $(u'_1, \ldots, u'_{N+K}) = (p_1, \ldots, p_K, w_1, \ldots, w_N)$ the following holds:

$$T_{C_1}^{(l)}(u_i) = T_{C_2}^{(l)}(u'_i) \quad \forall l \geq 1. \tag{27}$$

We start with an auxiliary fact that for all $i, j = 1, \ldots, N$ and any $l \geq 1$ the following holds:

$$\left(T_{G_1}^{(l)}(v_i) = T_{G_1}^{(l)}(v_j)\right) \Rightarrow T_{C_1}^{(l)}(v_i) = T_{C_1}^{(l)}(v_j). \tag{28}$$

We show (28) by induction. The base case of $l = 1$ is trivial since $T_G^{(1)}(v) = h_v$ for any v and G. Assume now that (28) holds for $k = 1, \dots, l$ and $T_{G_1}^{(l+1)}(v_i) = T_{G_1}^{(l+1)}(v_j)$. We will show that $T_{C_1}^{(l+1)}(v_i) = T_{C_1}^{(l+1)}(v_j)$. We first note that the roots coincide:

$$\text{ROOT}(T_{C_1}^{(l+1)}(v_i)) = h_{v_i} = h_{v_j} = \text{ROOT}(T_{C_1}^{(l+1)}(v_j)) \tag{29}$$

from the base case. Furthermore

$$\begin{aligned} \text{CHILDREN}\left(T_{C_1}^{(l+1)}(v_i)\right) &= \{T_{C_1}^{(l)}(v) \mid v \in N_{C_1}(v_i)\} \\ &= \{T_{C_1}^{(l)}(v) \mid v \in N_{G_1}(v_i)\} \cup \{T_{C_1}^{(l)}(p) \mid p \in P)\} \end{aligned} \tag{30}$$

Now since $T_{G_1}^{(l+1)}(v_i) = T_{G_1}^{(l+1)}(v_j)$, it holds that:

$$\{T_{G_1}^{(l)}(v) \mid v \in N_{G_1}(v_i)\} = \{T_{G_1}^{(l)}(v) \mid v \in N_{G_1}(v_j)\}$$

and therefore by our induction assumption:

$$\{T_{C_1}^{(l)}(v) \mid v \in N_{G_1}(v_i)\} = \{T_{C_1}^{(l)}(v) \mid v \in N_{G_1}(v_j)\}$$

and subsequently, continuing (30):

$$\begin{aligned} \text{CHILDREN}\left(T_{C_1}^{(l+1)}(v_i)\right) &= \{T_{C_1}^{(l)}(v) \mid v \in N_{C_1}(v_i)\} \\ &= \{T_{C_1}^{(l)}(v) \mid v \in N_{G_1}(v_i)\} \cup \{T_{C_1}^{(l)}(p) \mid p \in P)\} \\ &= \{T_{C_1}^{(l)}(v) \mid v \in N_{G_1}(v_j)\} \cup \{T_{C_1}^{(l)}(p) \mid p \in P)\} \\ &= \{T_{C_1}^{(l)}(v) \mid v \in N_{C_1}(v_j)\} \\ &= \text{CHILDREN}\left(T_{C_1}^{(l+1)}(v_j)\right). \end{aligned} \tag{31}$$

Combining (29) and (31) we have shown that $T_{C_1}^{(l+1)}(v_i) = T_{C_1}^{(l+1)}(v_j)$ and therefore proven (28). We now proceed to show (27) by proving a stronger statement, i.e. that for all $l \geq 1$ and all $i, j = 1, \dots, N, k = 1, \dots, K$ the following hold:

$$(1) \quad \left(T_{G_1}^{(l)}(v_i) = T_{G_2}^{(l)}(w_j)\right) \Rightarrow T_{C_1}^{(l)}(v_i) = T_{C_2}^{(l)}(w_j) \tag{32}$$

$$(2) \quad T_{C_1}^{(l)}(p_k) = T_{C_2}^{(l)}(p_k) \tag{33}$$

Again, we proceed with a proof by induction and note that the base case of $l = 1$ is trivial, because all computation trees are just features of roots. Assume now that (32) and (33) hold for $k = 1, \dots, l$. We will show they also hold for $k = l + 1$. Let us first consider any $k = 1, \dots, K$ and note that

$$\text{ROOT}\left(T_{C_1}^{(l+1)}(p_k)\right) = h_{p_k} = \text{ROOT}\left(T_{C_2}^{(l+1)}(p_k)\right). \tag{34}$$

And for the children:

$$\begin{aligned} \text{CHILDREN}\left(T_{C_1}^{(l+1)}(p_k)\right) &= \{T_{C_1}^{(l)}(v) \mid v \in N_{C_1}(p_k)\} \\ &= \{T_{C_1}^{(l)}(p) \mid p \in N_P(p_k)\} \cup \{T_{C_1}^{(l)}(v) \mid v \in G_1\}. \end{aligned} \tag{35}$$

Since $G_1 \equiv G_2$, it holds that $\{T_{G_1}^{(l)}(v)\}_{v \in G_1} = \{T_{G_2}^{(l)}(w)\}_{w \in G_2}$ and therefore, by the induction assumption (32):

$$\{T_{C_1}^{(l)}(v) \mid v \in G_1\} = \{T_{C_2}^{(l)}(w) \mid w \in G_2\}. \tag{36}$$

Furthermore, by the induction assumption (33), it holds that

$$\{T_{C_1}^{(l)}(p) \mid p \in N_P(p_k)\} = \{T_{C_2}^{(l)}(p) \mid p \in N_P(p_k)\}. \tag{37}$$

Combining (36) and (37) yields:

$$\begin{aligned}\text{CHILDREN}\left(T_{C_1}^{(l+1)}(p_k)\right) &= \{T_{C_1}^{(l)}(v) \mid v \in N_{C_1}(p_k)\} \\ &= \{T_{C_1}^{(l)}(p) \mid p \in N_P(p_k)\} \cup \{T_{C_1}^{(l)}(v) \mid v \in G_1\} \\ &= \{T_{C_2}^{(l)}(p) \mid p \in N_P(p_k)\} \cup \{T_{C_2}^{(l)}(w) \mid w \in G_2\} \\ &= \{T_{C_2}^{(l)}(w) \mid w \in N_{C_2}(p_k)\} \\ &= \text{CHILDREN}\left(T_{C_2}^{(l+1)}(p_k)\right)\end{aligned} \tag{38}$$

and thus (34) and (38) imply

$$T_{C_1}^{(l+1)}(p_k) = T_{C_2}^{(l+1)}(p_k). \tag{39}$$

We will now show that $T_{C_1}^{(l+1)}(v_i) = T_{C_2}^{(l+1)}(w_i)$ for all i. Consider now any $i = 1, \ldots, N$ and note that:

$$\text{ROOT}\left(T_{C_1}^{(l+1)}(v_i)\right) = h_{v_i} = h_{w_i} = \text{ROOT}\left(T_{C_2}^{(l+1)}(w_i)\right). \tag{40}$$

Moreover:

$$\begin{aligned}\text{CHILDREN}\left(T_{C_1}^{(l+1)}(v_i)\right) &= \{T_{C_1}^{(l)}(v) \mid v \in N_{C_1}(v_i)\} \\ &= \{T_{C_1}^{(l)}(v) \mid v \in N_{G_1}(v_i)\} \cup \{T_{C_1}^{(l)}(p) \mid p \in P\}.\end{aligned} \tag{41}$$

Since $G_1 \equiv G_2$, it holds that $T_{G_1}^{(l+1)}(v_i) = T_{G_2}^{(l+1)}(w_i)$ and therefore $\{T_{G_1}^{(l)}(v) \mid v \in N_{G_1}(v_i)\} = \{T_{G_2}^{(l)}(w) \mid w \in N_{G_2}(w_i)\}$, so using the induction assumption (32):

$$\{T_{C_1}^{(l)}(v) \mid v \in N_{G_1}(v_i)\} = \{T_{C_2}^{(l)}(w) \mid w \in N_{G_2}(w_i)\}. \tag{42}$$

On the other hand, from the induction assumption (33):

$$\{T_{C_1}^{(l)}(p) \mid p \in P\} = \{T_{C_2}^{(l)}(p) \mid p \in P\}, \tag{43}$$

so from (42) and (43), we have:

$$\begin{aligned}\text{CHILDREN}\left(T_{C_1}^{(l+1)}(v_i)\right) &= \{T_{C_1}^{(l)}(v) \mid v \in N_{C_1}(v_i)\} \\ &= \{T_{C_1}^{(l)}(v) \mid v \in N_{G_1}(v_i)\} \cup \{T_{C_1}^{(l)}(p) \mid p \in P\} \\ &= \{T_{C_2}^{(l)}(w) \mid w \in N_{G_2}(w_i)\} \cup \{T_{C_2}^{(l)}(p) \mid p \in P\} \\ &= \{T_{C_2}^{(l)}(w) \mid v \in N_{C_2}(w_i)\} \\ &= \text{CHILDREN}\left(T_{C_2}^{(l+1)}(w_i)\right).\end{aligned} \tag{44}$$

Using (40) and (44), we have shown that for all $i = 1 \ldots, N$

$$T_{C_1}^{(l+1)}(v_i) = T_{C_2}^{(l+1)}(w_i). \tag{45}$$

Now take any $i, j = 1, \ldots, N$, such that $T_{G_1}^{(l+1)}(v_i) = T_{G_2}^{(l+1)}(w_j)$. Since $G_1 \equiv G_2$, it holds that $T_{G_1}^{(l+1)}(v_j) = T_{G_2}^{(l+1)}(w_j) = T_{G_1}^{(l+1)}(v_i)$ and thus:

$$T_{C_1}^{(l+1)}(v_i) \overset{(28)}{=} T_{C_1}^{(l+1)}(v_j) \overset{(45)}{=} T_{C_2}^{(l+1)}(w_j). \tag{46}$$

Putting together (46) and (39) concludes the induction step and therefore the proof of (32) and (33). An immediate consequence of (32) is that for all $i = 1, \ldots, N$ and $l \geq 1$:

$$T_{C_1}^{(l)}(v_i) = T_{C_2}^{(l)}(w_i), \tag{47}$$

which together with (33) show that for all $i = 1, \ldots, K + N$ and $l \geq 1$:

$$T_{C_1}^{(l)}(u_i) = T_{C_2}^{(l)}(u_i') \tag{48}$$

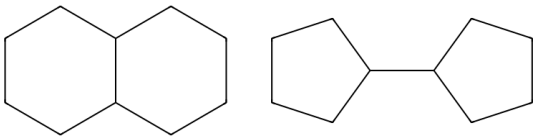


Figure 8: Molecular graphs indistinguishable by LU-GNNs.

and thus $C_1 \equiv C_2$, concluding the proof of (i).

To show (ii), we take G_1 and G_2 to be molecular graphs corresponding to SMILES representations: C1CCC2CCCCC2C1 and C1CCC(C1)C2CCCC2 respectively (see Figure 8). All heavy atoms are carbon and all bonds are single, so all node and edge features are identical. All bonds are also of equal length. We also take an arbitrary protein graph P to define complexes $C_1 = \mathcal{C}(P, G_1)$, $C_2 = \mathcal{C}(P, G_2)$.

The intra-ligand LU3D-GNN produces identical sets of embeddings for G_1 and G_2, which can be shown in the same way as in Lemma 1. Similarly for intra-ligand LU-GNNs since $G_1 \equiv_{3D} G_2 \Rightarrow G_1 \equiv G_2$.

The induced subgraphs of C_1 and C_2 corresponding to nodes of P are identical and therefore any intra-protein GNN will produce identical sets of embeddings for both.

Now since $G_1 \equiv G_2$, it follows from (i) that $C_1 \equiv C_2$. Therefore inter protein-ligand LU-GNN will produce identical embeddings for C_1 and C_2. □

F Generalization Bounds

In this section we derive generalization bounds of our scoring model defined in Section 3.2. We begin with recalling the definition of a generalization error.

Definition 1. (Generalization error). *Let $f : \mathcal{X} \to \mathcal{Y}$ and $L : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_+$ be a loss function. Let $S = \{(x_1, y_1), \ldots, (x_m, y_m)\} \subseteq \mathcal{X} \times \mathcal{Y}$ be a training sample. The generalization error of f is defined as the difference between expected loss and the sample loss:*

$$\mathcal{R}_S(f) = \mathbb{E}L(f(x), y) - \frac{1}{m}\sum_{i=1}^{m} L(f(x_i), y_i), \tag{9}$$

where the expectation is taken over (x, y) sampled from the underlying data distribution.

Now we introduce tools we will use to prove the main results.

F.1 Learning theory preliminaries

Similarly to (Garg et al., 2020; Chen et al., 2020), we will be using the concept of Empirical Rademacher Complexity (ERC) as the main tool to derive the bounds:

Definition 4 (Empirical Rademacher complexity)**.** *Let $L : \mathcal{Y} \times \mathcal{Y} \to [0, A]$ be a bounded loss function, $\mathcal{F} \subseteq \mathcal{Y}^{\mathcal{X}}$ a class of functions and $S = ((x_1, y_1), \ldots, (x_m, y_m)) \subseteq \mathcal{X} \times \mathcal{Y}$ a fixed sample of size m. Then the empirical Rademacher complexity of $\mathcal{F}$ with respect to the sample S and loss function L is*

$$\widehat{\mathfrak{R}}_{S,L}(\mathcal{F}) = \mathbb{E}_\sigma \left[\sup_{f \in \mathcal{F}} \frac{1}{m} \sum_{i=1}^{m} \sigma_i L(f(x_i), y_i) \right], \tag{49}$$

Intuitively, the ERC measures model class' ability to fit to random noise. A known result from learning theory relates ERC with the generalization error:

Theorem 1 ((Mohri et al., 2012))**.** *Let $L : \mathcal{Y} \times \mathcal{Y} \to [0, 1]$ be loss function and $\mathcal{F} \subseteq \mathcal{Y}^{\mathcal{X}}$ a class of functions. Then for any $\delta > 0$, with probability at least $1 - \delta$ over choosing a sample $S = ((x_1, y_1), \ldots, (x_m, y_m)) \subseteq \mathcal{X} \times \mathcal{Y}$ of size m, the following holds for any $f \in \mathcal{F}$:*

$$\mathcal{R}_S(f) \leq 2\widehat{\mathfrak{R}}_{S,L}(\mathcal{F}) + 3\sqrt{\frac{\log \frac{2}{\delta}}{2m}}. \tag{50}$$

From Theorem 1 it is evident that in order to find a generalization bound it suffices to find a bound for ERC. To do that, we need to introduce the concept of a covering of a function class.

Definition 5 (Covering). *Let $\mathcal{F} \subseteq \mathcal{Y}^{\mathcal{X}}$ be a class of functions. A covering of $\mathcal{F}$ at scale $r > 0$ w.r.t. to a norm $\|\cdot\|$ is a subset of $\mathcal{F}$ denoted by $\mathcal{C}(\mathcal{F}, r, \|\cdot\|) \subseteq \mathcal{F}$ such that for all $f \in \mathcal{F}$, there exists $\widehat{f} \in \mathcal{C}(\mathcal{F}, r, \|\cdot\|)$ such that*

$$\sup_{x \in \mathcal{X}} \|\widehat{f}(x) - f(x)\| \leq r.$$

We will refer to the minimal cardinality of coverings of $\mathcal{F}$ as the covering number and denote it by $\mathcal{N}(\mathcal{F}, r, \|\cdot\|)$.

We now recall a result allowing to bound ERC by using the covering number:

Lemma 2 ((Bartlett et al., 2017)). *Let $\mathcal{F} \subseteq \mathcal{Y}^{\mathcal{X}}$ be a class of functions. Now let $L : \mathcal{Y} \times \mathcal{Y} \to [0, A]$ be a bounded loss function, Lipschitz continuous in its first argument with a Lipschitz constant $K > 0$. Assume also that there exists $f_0 \in \mathcal{F}$ such that for all (x, y) in the data distribution $L(f_0(x), y) = 0$. Let $S = ((x_1, y_1), \ldots, (x_m, y_m)) \subseteq \mathcal{X} \times \mathcal{Y}$ be a given sample of size m. Then*

$$\widehat{\mathfrak{R}}_{S,L}(\mathcal{F}) \leq \inf_{\alpha > 0} \left(\frac{4\alpha}{\sqrt{m}} + \frac{12}{m} \int_{\alpha}^{2A\sqrt{m}} \sqrt{\log \mathcal{N}\left(\mathcal{F}, \frac{r}{K}, \|\cdot\|\right)} dr \right).$$

In our analysis we will use a bounded version of l_2 loss, i.e.

$$L(\widehat{y}, y) = \min\left\{ \frac{1}{2}(\widehat{y} - y)^2, 1 \right\}.$$

It can be easily verified that L is Lipschitz continuous with a constant $K = 1$ and is of course bounded by 1. For this loss function, the bound in Lemma 2 simplifies to:

$$\widehat{\mathfrak{R}}_{S,L}(\mathcal{F}) \leq \inf_{\alpha > 0} \left(\frac{4\alpha}{\sqrt{m}} + \frac{12}{m} \int_{\alpha}^{2\sqrt{m}} \sqrt{\log \mathcal{N}(\mathcal{F}, r, \|\cdot\|)} \mathrm{d}r \right). \tag{51}$$

We recall one last lemma that we will use in our reasoning:

Lemma 3 ((Chen et al., 2020)). *Let $\mathcal{G} = \{A \in \mathbb{R}^{d_1 \times d_2} : \|A\| \leq \lambda\}$ be the set of matrices with bounded spectral norm and $r > 0$ given. The covering number of $\mathcal{G}$ can be bounded by*

$$\mathcal{N}(\mathcal{G}, r, \|\|_F) \leq \left(1 + 2\frac{\min\{d_1, d_2\}\lambda}{r}\right)^{d_1 d_2}.$$

Our derivation of generalization bounds proceeds through the following steps:

1. Show that the scoring function is Lipschitz continuous w.r.t model parameters;
2. Use it to show that a cartesian product of coverings of weight matrices defines a covering of function class of scoring models;
3. Use Lemmas 3, 2 and Theorem 1 to compute the generalization bound.

We now move on to showing that the scoring model is Lipschitz w.r.t. model parameters and start with analyzing MLPs.

F.2 Boundedness and Lipschitz continuity of MLPs

Definition 6 (MLP). *A Multi-layer perceptron with k layers is given by $g_k = f_k \circ f_{k-1} \circ \cdots \circ f_1$, where f_i is the i-th layer given by:*

$$f_i(x; W_i) = \sigma(W_i x),$$

where σ is the activation function, W_i is the weight matrix.

Consider a function given by

$$f_i(x; W_i) = \sigma(W_i x + b_i).$$

One can augment the input space by appending a constant 1 to x to obtain $\tilde{x}$ and redefining the weight matrix as $\tilde{W} = \text{concat}(W, b)$. Thus the function can be expressed as

$$f_i(x; W_i) = \sigma(\tilde{W}_i \tilde{x}).$$

Given this representation, we can assume, without loss of generality, that $b_i = 0$.

Lemma 4 (Boundedness of MLP)**.** *Consider an MLP with k layers, where σ is the activation function, W_i is the weight matrix. Suppose that σ is Lipschitz with constant L_σ and the property $\sigma(0) = 0$. Suppose that the input is bounded $\|x\| \leq M$ and that spectral norms of weight matrices are bounded $\|W_i\| \leq M_W$. Then for all x:*

$$g_k(x) \leq M(L_\sigma M_W)^k.$$

Proof. Note that for all $\|x\| \leq M$:

$$\begin{aligned}\|g_k(x)\| = \|\sigma(W_k g_{k-1}(x)\| = \|\sigma(W_k g_{k-1}(x) - \sigma(0)\| \leq L_\sigma \left(\|W_k g_{k-1}(x)\|\right) \\ \leq L_\sigma M_W \|g_{k-1}(x)\| \leq \cdots \leq (L_\sigma M_W)^k \|x\| \leq M(L_\sigma M_W)^k.\end{aligned}$$

□

Lemma 5 (Lipschitz continuity of MLP wrt model parameters)**.** *Consider two k-layer MLPs given by two sets of parameters: W and $\tilde{W}$ sharing the same activation function σ with Lipschitz constant L_σ and $\sigma(0) = 0$. Assume further that the spectral norm of the weights matrices are bounded by M_W and that the input is bounded by M. Then it holds that for all $x, W, \tilde{W}$:*

$$\|g_k(x; W) - g_k(x; \tilde{W})\| \leq L_\sigma M (L_\sigma M_W)^{k-1} \|W - \tilde{W}\|, \tag{52}$$

where $\|W - \tilde{W}\| = \sum_{i=1}^k \|W_i - \tilde{W}_i\|$.

Proof.

$$\begin{aligned}\|g_k(x; W) - g_k(x; \tilde{W})\| &= \|\sigma(W_k g_{k-1}(x; W)) - \sigma(\tilde{W}_k g_{k-1}(x; \tilde{W}))\| \\ &\leq L_\sigma \|W_k g_{k-1}(x; W) - \tilde{W}_k g_{k-1}(x; \tilde{W}))\|\end{aligned}$$

Now using the telescoping technique:

$$\begin{aligned}&\|W_k g_{k-1}(x; W) - \tilde{W}_k g_{k-1}(x; \tilde{W})\| \\ &\leq \|W_k g_{k-1}(x; W) - W_k g_{k-1}(x; \tilde{W})\| + \|W_k g_{k-1}(x; \tilde{W}) - \tilde{W}_k g_{k-1}(x; \tilde{W})\| \\ &\leq \|W_k\| \cdot \|g_{k-1}(x; W) - g_{k-1}(x; \tilde{W})\| + \|g_{k-1}(x; W)\| \cdot \|W_k - \tilde{W}_k\| \\ &\leq M_W \|g_{k-1}(x; W) - g_{k-1}(x; \tilde{W})\| + M(L_\sigma M_W)^{k-1} \|W_k - \tilde{W}_k\|.\end{aligned}$$

Therefore

$$\begin{aligned}&\|g_k(x; W) - g_k(x; \tilde{W})\| \leq L_\sigma M (L_\sigma M_W)^{k-1} \|W_k - \tilde{W}_k\| + L_\sigma M_W \|g_{k-1}(x; W) - g_{k-1}(x; \tilde{W})\| \\ &\leq L_\sigma M (L_\sigma M_W)^{k-1} \|W_k - \tilde{W}_k\| \\ &+ L_\sigma M_W \left(L_\sigma M (L_\sigma M_W)^{k-2} \|W_{k-1} - \tilde{W}_{k-1}\| + L_\sigma M_W \|g_{k-2}(x; W) - g_{k-2}(x; \tilde{W})\| \right) \\ &= L_\sigma M (L_\sigma M_W)^{k-1} (\|W_k - \tilde{W}_k\| + \|W_{k-1} - \tilde{W}_{k-1}\|) + (L_\sigma M_W)^2 \|g_{k-2}(x; W) - g_{k-2}(x; \tilde{W})\| \\ &\leq L_\sigma M (L_\sigma M_W)^{k-1} \sum_{i=1}^k \|W_i - \tilde{W}_i\|.\end{aligned}$$

□

Lemma 6 (Lipschitz continuity of MLP wrt model input)**.** *Let g_k be a k-layer MLP with a Lipschitz constant L_σ and bounded spectral norm of weight matrices $\|W_i\| \leq M_W$. Then g_k is Lipschitz wrt model input with a constant:*

$$L = (L_\sigma M_W)^k.$$

Proof.

$$\|g_k(x) - g_k(y)\| \leq L_\sigma \|W_k\| \cdot \|g_{k-1}(x) - g_{k-1}(y)\| \leq L_\sigma M_W \|g_{k-1}(x) - g_{k-1}(y)\|.$$

□

Lemma 7 (Multiple inputs MLP)**.** *Suppose a k-layer MLP takes multiple inputs $x_1, \ldots, x_K$ by concatenating them into a single vector $x = CONCAT([x_1, \ldots, x_K])$. Suppose that all inputs are bounded by corresponding constants: $\|x_i\| \leq M_i$. Then the output is bounded:*

$$g_k(x_1, \ldots, x_K) \leq M(L_\sigma M_W)^k,$$

where $M = \sqrt{K} \sum_i M_i$.

Proof. Claim follows from Lemma 4 and the inequality:

$$\|x\| = \sqrt{\|x_1\|^2 + \cdots + \|x_K\|^2} \leq \sqrt{K} \max_i \|x_i\| \leq \sqrt{K} \sum_i \|x_i\| \leq \sqrt{K} \sum_i M_i.$$

□

F.3 Boundedness and Lipschitz continuity of EGNN embeddings

Recall the definition of the ε-soft normalized EGNN:

$$\begin{aligned}
\mathbf{m}_{u \to v}^{(l-1)} &= \phi_e\left(\mathbf{h}_u^{(l-1)}, \mathbf{h}_v^{(l-1)}, \|\mathbf{x}_u^{(l-1)} - \mathbf{x}_v^{(l-1)}\|, e_{uv}\right) \\
\mathbf{x}_v^{(l)} &= \mathbf{x}_v^{(l-1)} + \frac{1}{|\mathcal{N}(v)|} \sum_{u \in \mathcal{N}(v)} \frac{\mathbf{x}_v^{(l-1)} - \mathbf{x}_u^{(l-1)}}{\|\mathbf{x}_u^{(l-1)} - \mathbf{x}_v^{(l-1)}\| + \varepsilon} \phi_x(\mathbf{m}_{u \to v}^{(l-1)}) \\
\mathbf{m}_v^{(l-1)} &= \sum_{u \in \mathcal{N}(v)} \mathbf{m}_{u \to v}^{(l-1)} \\
\mathbf{h}_v^{(l)} &= \phi_h(\mathbf{h}_v^{(l-1)}, \mathbf{m}_v^{(l-1)}),
\end{aligned} \tag{53}$$

where ϕ_e, ϕ_x and ϕ_h are MLPs and $\{\mathbf{h}_v^{(0)}\}_{v \in V}$ are initialized to input node features and $\{\mathbf{x}_v^{(0)}\}_{v \in V}$ are initialized with 3D coordinates.

Lemma 8 (Boundedness of EGNN embeddings)**.** *Suppose that there exists $B^{(0)}$ such that for all $u, v \in V$.*

$$\max\{\|\mathbf{h}_v^{(0)}\|, \|\mathbf{x}_v^{(0)} - \mathbf{x}_u^{(0)}\|, \|e_{uv}\|\} \leq B^{(0)} \tag{54}$$

Suppose further that ϕ_x, ϕ_h and ϕ_e are all MLPs with k layers with Lipschitz activation functions with constants at most L_σ and spectral norms of all their weight matrices bounded by M_W. Then, for all u, v:

$$\begin{aligned}
\|\mathbf{h}_v^{(l)}\| &\leq MK^l \\
\|\mathbf{x}_v^{(l)} - \mathbf{x}_u^{(l)}\| &\leq MK^l \\
\|\mathbf{m}_v^{(l)}\| &\leq MK^l,
\end{aligned} \tag{55}$$

where

$$\begin{aligned}
M &= 8DB^{(0)}L_\phi \\
K &= \max\left\{2\sqrt{2}L_\phi, 1 + 16L_\phi^2, 4DL_\phi\left(1 + 2\sqrt{2}L_\phi + 8L_\phi^2\right)\right\} \\
L_\phi &= \max\{(L_\sigma M_W)^k, 1\}
\end{aligned} \tag{56}$$

and D is the maximum degree in the graph.

Proof. First note that from Lemma 7:

$$\|\mathbf{m}_{u\to v}^{(0)}\| = \|\phi_e\left(\mathbf{h}_u^{(0)}, \mathbf{h}_v^{(0)}, \|\mathbf{x}_u^{(0)} - \mathbf{x}_v^{(0)}\|, e_{uv}\right)\| \leq 2(2B^{(0)} + B^{(0)} + B^{(0)})L_\phi.$$

And therefore:

$$\|\mathbf{m}_v^{(0)}\| \leq 8DB^{(0)}L_\phi = M.$$

We will prove the result by induction. The base case of $l = 0$ obviously holds from the definition of M. Now assume that (55) holds for $l' < l$.

From Lemma 7 and the inductive hypothesis:

$$\|\mathbf{h}_v^{(l)}\| \leq \sqrt{2}L_\phi(\|\mathbf{h}_v^{(l-1)}\| + \|\mathbf{m}_v^{(l-1)}\|) \leq 2\sqrt{2}L_\phi MK^{l-1} \leq MK^l,$$

where the last inequality holds because $K \geq 2\sqrt{2}L_\phi$. Similarly for $\mathbf{x}$:

$$\begin{aligned}\mathbf{x}_u^{(l)} - \mathbf{x}_v^{(l)} &= \mathbf{x}_u^{(l-1)} - \mathbf{x}_v^{(l-1)} \\ &+ \frac{1}{|\mathcal{N}(u)|}\sum_{u'\in\mathcal{N}(u)} \frac{\mathbf{x}_u^{(l-1)} - \mathbf{x}_{u'}^{(l-1)}}{\|\mathbf{x}_u^{(l-1)} - \mathbf{x}_{u'}^{(l-1)}\| + \varepsilon}\phi_x(\mathbf{m}_{u'\to u}^{(l-1)}) \\ &- \frac{1}{|\mathcal{N}(v)|}\sum_{v'\in\mathcal{N}(v)} \frac{\mathbf{x}_v^{(l-1)} - \mathbf{x}_{v'}^{(l-1)}}{\|\mathbf{x}_v^{(l-1)} - \mathbf{x}_{v'}^{(l-1)}\| + \varepsilon}\phi_x(\mathbf{m}_{v'\to v}^{(l-1)})\end{aligned}$$

and therefore:

$$\begin{aligned}\|\mathbf{x}_u^{(l)} - \mathbf{x}_v^{(l)}\| &\leq MK^{l-1} + 2L_\phi^2 2(2MK^{l-1} + MK^{l-1} + MK^{l-1}) \\ &= (1 + 16L_\phi^2)MK^{l-1} \leq MK^l,\end{aligned}$$

because $K \geq 1 + 16L_\phi^2$. Finally for $\mathbf{m}$:

$$\begin{aligned}\|\mathbf{m}_v^{(l)}\| &\leq \sum_{u\in\mathcal{N}(v)} \|\mathbf{m}_{u\to v}^{(l)}\| \leq 2\sum_{u\in\mathcal{N}(v)} L_\phi\left(\|\mathbf{h}_v^{(l)}\| + \|\mathbf{h}_u^{(l)}\| + \|\mathbf{x}_v^{(l)} - \mathbf{x}_u^{(l)}\| + \|e_{uv}\|\right) \\ &\leq 2DL_\phi\left(4\sqrt{2}L_\phi MK^{l-1} + (1 + 16L_\phi^2)MK^{l-1} + MK^{l-1}\right) \\ &= 4DL_\phi\left(1 + 2\sqrt{2}L_\phi + 8L_\phi^2\right)MK^{l-1} \leq MK^l,\end{aligned}$$

where the last inequality follows from the fact that
$K \geq 4DL_\phi\left(1 + 2\sqrt{2}L_\phi + 8L_\phi^2\right)$

□

Before we show Lipschitz continuity of EGNN embeddings, we need one more auxiliary result:

Lemma 9 (Lipschitz continuity preserved under ε-soft normalization)**.** *Let f be a Lipschitz continuous function with Lipschitz constant L_f. Then the ε-softly normalized function*

$$g(x) \coloneqq \frac{f(x)}{\|f(x)\| + \varepsilon}$$

is Lipschitz continuous with Lipschitz constant

$$L_g = \frac{3L_f}{\varepsilon}$$

Proof.

$$\begin{aligned}\|g(x)-g(y)\| &= \left\|\frac{f(x)}{\|f(x)\|+\varepsilon}-\frac{f(y)}{\|f(y)\|+\varepsilon}\right\| = \left\|\frac{(\|f(y)\|+\varepsilon)f(x)-(\|f(x)\|+\varepsilon)f(y)}{(\|f(x)\|+\varepsilon)(\|f(y)\|+\varepsilon)}\right\| \\ &\leq \frac{\varepsilon}{(\|f(x)\|+\varepsilon)(\|f(y)\|+\varepsilon)}\|f(x)-f(y)\| + \left\|\frac{\|f(y)\|f(x)-\|f(x)\|f(y)}{(\|f(x)\|+\varepsilon)(\|f(y)\|+\varepsilon)}\right\| \\ &\leq \frac{L_f}{\varepsilon}\|x-y\| + \left\|\frac{\|f(y)\|f(x)-\|f(x)\|f(y)}{(\|f(x)\|+\varepsilon)(\|f(y)\|+\varepsilon)}\right\|.\end{aligned}$$

Focusing on the numerator of the last term:

$$\begin{aligned}\|\|f(y)\|f(x)-\|f(x)\|f(y)\| &= \|\|f(y)\|f(x)-\|f(x)\|f(x)+\|f(x)\|f(x)-\|f(x)\|f(y)\| \\ &\leq |\|f(y)\|-\|f(x)\|| \cdot \|f(x)\| + \|f(x)\| \cdot \|f(x)-f(y)\| \\ &\leq \|f(x)\|\left(L_f\|x-y\| + |\|f(x)\|-\|f(y)\|\right) \\ &\leq 2\|f(x)\|L_f\|x-y\|,\end{aligned}$$

where in the last step we used the fact that $|\|a\|-\|b\|| \leq \|a-b\|$. Therefore:

$$\left\|\frac{\|f(y)\|f(x)-\|f(x)\|f(y)}{(\|f(x)\|+\varepsilon)(\|f(y)\|+\varepsilon)}\right\| \leq \frac{2\|f(x)\|L_f\|x-y\|}{(\|f(x)\|+\varepsilon)(\|f(y)\|+\varepsilon)} \leq \frac{2L_f}{\varepsilon}\|x-y\|$$

and

$$\|g(x)-g(y)\| \leq \frac{3L_f}{\varepsilon}\|x-y\|.$$

□

Lemma 10 (Lipschitz continuity of EGNN wrt parameters)**.** *Let EGNN be defined as in (53). Assume further that the inputs $\{\mathbf{h}_v^{(0)}\}_{v\in V}, \{\mathbf{x}_v^{(0)}-\mathbf{x}_u^{(0)}\}_{v,u\in V}, e_{uv}$ are bounded by $B^{(0)}$. Then the embeddings produced by the EGNN model are Lipschitz wrt to the model parameters, i.e.*

$$\|\mathbf{h}_v^{(L)}(W)-\mathbf{h}_v^{(L)}(\tilde{W})\| \leq MB^L K^{\frac{L(L-1)}{2}}\|W-\tilde{W}\|, \tag{57}$$

where $W=(W_{\phi_x}, W_{\phi_h}, W_{\phi_e})$ and $\|W-\tilde{W}\| = \|W_{\phi_x}-\tilde{W}_{\phi_x}\| + \|W_{\phi_e}-\tilde{W}_{\phi_e}\| + \|W_{\phi_h}-\tilde{W}_{\phi_h}\|$ and

$$\begin{aligned}M &= 8DB^{(0)}L_\phi \\ L_\phi &= \max\{1,(L_\sigma M_w)^k\} \\ B &= \max\left\{\frac{96L_\phi^2 M}{\varepsilon}, 72L_\phi^2, 8DL_\phi\right\} \\ K &= \max\left\{2\sqrt{2}L_\phi, 1+16L_\phi^2, 4DL_\phi\left(1+2\sqrt{2}L_\phi+8L_\phi^2\right)\right\}\end{aligned} \tag{58}$$

Proof. First, recall from Lemma 5 that an MLP with input bounded by M is Lipschitz wrt parameters with a constant $M \cdot L_p$, where $L_p = L_\sigma(L_\sigma M_W)^{k-1}$. It is also Lipschitz wrt to model inputs with a constant: $L_{in} = (L_\sigma M_W)^k$. We will use an auxiliary constant:

$$L_\phi = \max\{1, L_p, L_{in}\} = \max\{1,(L_\sigma M_W)^k\}.$$

We will show the result by induction. We will show a stronger statement, namely that all $\mathbf{h}_v^{(L)}, \mathbf{x}_v^{(L)}$ and $\mathbf{m}_v^{(L)}$ are Lipschitz continuous. The base case obviously holds for $\mathbf{h}_v^{(0)}$ and $\mathbf{x}_v^{(0)}$ as they do not depend on model parameters.

We see that

$$\|\mathbf{m}_v^{(0)}(W)-\mathbf{m}_v^{(0)}(\tilde{W})\| \leq \sum_{u\in\mathcal{N}(v)}\|\mathbf{m}_{u\to v}^{(0)}(W)-\mathbf{m}_{u\to v}^{(0)}(\tilde{W})\|. \tag{59}$$

Now from Lemma 5 and our assumption that inputs are bounded:

$$\|\mathbf{m}_{u\to v}^{(0)}(W)-\mathbf{m}_{u\to v}^{(0)}(\tilde{W})\| \leq L_\phi 8B^{(0)}\|W_{\phi_e}-\tilde{W}_{\phi_e}\|. \tag{60}$$

Consequently:

$$\|\mathbf{m}_v^{(0)}(W) - \mathbf{m}_v^{(0)}(\tilde{W})\| \leq M\|W_{\phi_e} - \tilde{W}_{\phi_e}\|$$

and thus establishing the base case for induction. Assume now that for all $l' < l$ there exist $K_{\mathbf{h}}^{(l')}, K_{\mathbf{x}}^{(l')}, K_{\mathbf{m}}^{(l')}$ such that for all v and all $W, \tilde{W}$:

$$\begin{aligned}
\|\mathbf{h}_v^{(l')}(W) - \mathbf{h}_v^{(l')}(\tilde{W})\| &\leq K_{\mathbf{h}}^{(l')}\|W - \tilde{W}\| \\
\|\mathbf{x}_v^{(l')}(W) - \mathbf{x}_v^{(l')}(\tilde{W})\| &\leq K_{\mathbf{x}}^{(l')}\|W - \tilde{W}\| \\
\|\mathbf{m}_v^{(l')}(W) - \mathbf{m}_v^{(l')}(\tilde{W})\| &\leq K_{\mathbf{m}}^{(l')}\|W - \tilde{W}\|.
\end{aligned} \tag{61}$$

We start with $\mathbf{h}$ and proceed with a telescoping argument:

$$\begin{aligned}
&\|\mathbf{h}_v^{(l)}(W) - \mathbf{h}_v^{(l)}(\tilde{W})\| \\
&= \left\|\phi_h(\mathbf{h}_v^{(l-1)}(W), \mathbf{m}_v^{(l-1)}(W); W_{\phi_h}) - \phi_h(\mathbf{h}_v^{(l-1)}(\tilde{W}), \mathbf{m}_v^{(l-1)}(\tilde{W}); \tilde{W}_{\phi_h})\right\| \\
&\leq \left\|\phi_h(\mathbf{h}_v^{(l-1)}(W), \mathbf{m}_v^{(l-1)}(W); W_{\phi_h}) - \phi_h(\mathbf{h}_v^{(l-1)}(W), \mathbf{m}_v^{(l-1)}(W); \tilde{W}_{\phi_h})\right\| \\
&+ \left\|\phi_h(\mathbf{h}_v^{(l-1)}(W), \mathbf{m}_v^{(l-1)}(W); \tilde{W}_{\phi_h}) - \phi_h(\mathbf{h}_v^{(l-1)}(\tilde{W}), \mathbf{m}_v^{(l-1)}(\tilde{W}); \tilde{W}_{\phi_h})\right\|
\end{aligned}$$

Now we use Lemmas 5, 7 and 8 to bound the first term and Lemma 6 to bound the second

$$\begin{aligned}
&\|\mathbf{h}_v^{(l)}(W) - \mathbf{h}_v^{(l)}(\tilde{W})\| \\
&\leq \sqrt{2}L_\phi(\|\mathbf{h}_v^{(l-1)}(W)\| + \|\mathbf{m}_v^{(l-1)}(W)\|)\|W_{\phi_h} - \tilde{W}_{\phi_h}\| \\
&+ L_\phi\left(\|\mathbf{h}_v^{(l-1)}(W) - \mathbf{h}_v^{(l-1)}(\tilde{W})\| + \|\mathbf{m}_v^{(l-1)}(W) - \mathbf{m}_v^{(l-1)}(\tilde{W})\|\right) \\
&\leq 2\sqrt{2}L_\phi M K^{l-1}\|W_{\phi_h} - \tilde{W}_{\phi_h}\| + L_\phi(K_{\mathbf{h}}^{(l-1)} + K_{\mathbf{m}}^{(l-1)})\|W - \tilde{W}\| \\
&\leq L_\phi(2\sqrt{2}MK^{l-1} + K_{\mathbf{h}}^{(l-1)} + K_{\mathbf{m}}^{(l-1)})\|W - \tilde{W}\|.
\end{aligned}$$

Moving on to $\mathbf{m}$:

$$\left\|\mathbf{m}_v^{(L)}(W) - \mathbf{m}_v^{(L)}(\tilde{W})\right\| = \left\|\sum_{u\in\mathcal{N}(v)} \mathbf{m}_{u\to v}^{(L-1)}(W) - \sum_{u\in\mathcal{N}(v)} \mathbf{m}_{u\to v}^{(L-1)}(\tilde{W})\right\| \leq \sum_{u\in\mathcal{N}(v)} \left\|\mathbf{m}_{u\to v}^{(L-1)}(W) - \mathbf{m}_{u\to v}^{(L-1)}(\tilde{W})\right\|.$$

Let's now proceed to bound a single term and simplify notation with $\mathbf{w}_{uv}^{(l)}(W) \coloneqq (\mathbf{h}_u^{(l)}(W), \mathbf{h}_v^{(l)}(W), \|\mathbf{x}_u^{(l)}(W) - \mathbf{x}_v^{(l)}(W)\|, e_{uv})$:

$$\begin{aligned}
&\left\|\mathbf{m}_{u\to v}^{(l-1)}(W) - \mathbf{m}_{u\to v}^{(l-1)}(\tilde{W})\right\| = \left\|\phi_e\left(\mathbf{w}_{uv}^{(l-1)}(W); W_{\phi_e}\right) - \phi_e\left(\mathbf{w}_{uv}^{(l-1)}(\tilde{W}); \tilde{W}_{\phi_e}\right)\right\| \\
&= \left\|\phi_e\left(\mathbf{w}_{uv}^{(l-1)}(W); W_{\phi_e}\right) - \phi_e\left(\mathbf{w}_{uv}^{(l-1)}(W); \tilde{W}_{\phi_e}\right) + \phi_e\left(\mathbf{w}_{uv}^{(l-1)}(W); \tilde{W}_{\phi_e}\right) - \phi_e\left(\mathbf{w}_{uv}^{(l-1)}(\tilde{W}); \tilde{W}_{\phi_e}\right)\right\| \\
&\leq \left\|\phi_e\left(\mathbf{w}_{uv}^{(l-1)}(W); W_{\phi_e}\right) - \phi_e\left(\mathbf{w}_{uv}^{(l-1)}(W); \tilde{W}_{\phi_e}\right)\right\| + \left\|\phi_e\left(\mathbf{w}_{uv}^{(l-1)}(W); \tilde{W}_{\phi_e}\right) - \phi_e\left(\mathbf{w}_{uv}^{(l-1)}(\tilde{W}); \tilde{W}_{\phi_e}\right)\right\| \\
&\leq 4L_\phi M K^{l-1}\|W_{\phi_e} - \tilde{W}_{\phi_e}\| + L_\phi\|\mathbf{w}_{uv}^{(l-1)}(W) - \mathbf{w}_{uv}^{(l-1)}(\tilde{W})\|.
\end{aligned}$$

Now focusing on the last term:

$$\begin{aligned}
&\|\mathbf{w}_{uv}^{(l-1)}(W) - \mathbf{w}_{uv}^{(l-1)}(\tilde{W})\| \\
&= \|\mathbf{h}_u^{(l-1)}(W) - \mathbf{h}_u^{(l-1)}(\tilde{W}) + \mathbf{h}_v^{(l-1)}(W) - \mathbf{h}_v^{(l-1)}(\tilde{W}) + \|\mathbf{x}_u^{(l-1)}(W) - \mathbf{x}_v^{(l-1)}(W)\| - \|\mathbf{x}_u^{(l-1)}(\tilde{W}) - \mathbf{x}_v^{(l-1)}(\tilde{W})\|\| \\
&\leq 2K_{\mathbf{h}}^{(l-1)}\|W - \tilde{W}\| + \left|\|\mathbf{x}_u^{(l-1)}(W) - \mathbf{x}_v^{(l-1)}(W)\| - \|\mathbf{x}_u^{(l-1)}(\tilde{W}) - \mathbf{x}_v^{(l-1)}(\tilde{W})\|\right| \\
&\leq 2K_{\mathbf{h}}^{(l-1)}\|W - \tilde{W}\| + \|\mathbf{x}_u^{(l-1)}(W) - \mathbf{x}_u^{(l-1)}(\tilde{W})\| + \|\mathbf{x}_v^{(l-1)}(W) - \mathbf{x}_v^{(l-1)}(\tilde{W})\| \\
&\leq 2(K_{\mathbf{h}}^{(l-1)} + K_{\mathbf{x}}^{(l-1)})\|W - \tilde{W}\|
\end{aligned}$$

and therefore:

$$\left\|\mathbf{m}_{u\to v}^{(l-1)}(W)-\mathbf{m}_{u\to v}^{(l-1)}(\tilde{W})\right\|\leq L_\phi(4MK^{l-1}+2(K_{\mathbf{h}}^{(l-1)}+K_{\mathbf{x}}^{(l-1)}))\|W-\tilde{W}\| \tag{62}$$

and:

$$\left\|\mathbf{m}_{v}^{(L)}(W)-\mathbf{m}_{v}^{(L)}(\tilde{W})\right\|\leq DL_\phi(4MK^{l-1}+2(K_{\mathbf{h}}^{(l-1)}+K_{\mathbf{x}}^{(l-1)}))\|W-\tilde{W}\|$$

Finally, for $\mathbf{x}$:

$$\begin{aligned}
&\|\mathbf{x}_v^{(l)}(W)-\mathbf{x}_v^{(l)}(\tilde{W})\|\\
&=\|\mathbf{x}_v^{(l-1)}(W)-\mathbf{x}_v^{(l-1)}(\tilde{W})\|\\
&+\frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)}\left\|\frac{\mathbf{x}_v^{(l-1)}(W)-\mathbf{x}_u^{(l-1)}(W)}{\|\mathbf{x}_v^{(l-1)}(W)-\mathbf{x}_u^{(l-1)}(W)\|+\varepsilon}\phi_x(\mathbf{m}_{u\to v}^{(l-1)}(W);W_{\phi_x})\right.\\
&\left.-\frac{\mathbf{x}_v^{(l-1)}(\tilde{W})-\mathbf{x}_u^{(l-1)}(\tilde{W})}{\|\mathbf{x}_v^{(l-1)}(\tilde{W})-\mathbf{x}_u^{(l-1)}(\tilde{W})\|+\varepsilon}\phi_x(\mathbf{m}_{u\to v}^{(l-1)}(\tilde{W});\tilde{W}_{\phi_x})\right\|\\
&\leq K_{\mathbf{x}}^{(l-1)}\|W-\tilde{W}\|\\
&+\frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)}\left\|\frac{\mathbf{x}_v^{(l-1)}(W)-\mathbf{x}_u^{(l-1)}(W)}{\|\mathbf{x}_v^{(l-1)}(W)-\mathbf{x}_u^{(l-1)}(W)\|+\varepsilon}\right\|\|\phi_x(\mathbf{m}_{u\to v}^{(l-1)}(W);W_{\phi_x})-\phi_x(\mathbf{m}_{u\to v}^{(l-1)}(\tilde{W});\tilde{W}_{\phi_x})\|\\
&+\frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)}\|\phi_x(\mathbf{m}_{u\to v}^{(l-1)}(\tilde{W});\tilde{W}_{\phi_x})\|\left\|\frac{\mathbf{x}_v^{(l-1)}(W)-\mathbf{x}_u^{(l-1)}(W)}{\|\mathbf{x}_v^{(l-1)}(W)-\mathbf{x}_u^{(l-1)}(W)\|+\varepsilon}-\frac{\mathbf{x}_v^{(l-1)}(\tilde{W})-\mathbf{x}_u^{(l-1)}(\tilde{W})}{\|\mathbf{x}_v^{(l-1)}(\tilde{W})-\mathbf{x}_u^{(l-1)}(\tilde{W})\|+\varepsilon}\right\|.
\end{aligned}$$

Now, since (from the inductive assumption) $\mathbf{x}_v^{(l-1)},\mathbf{x}_u^{(l-1)}$ are both Lipschitz continuous with a constant $K_{\mathbf{x}}^{(l-1)}$, it follows that $\mathbf{x}_v^{(l-1)}-\mathbf{x}_u^{(l-1)}$ is Lipschitz continuous with a constant $2K_{\mathbf{x}}^{(l-1)}$. Therefore, using Lemma 9, we know that $\frac{\mathbf{x}_v^{(l-1)}-\mathbf{x}_u^{(l-1)}}{\|\mathbf{x}_v^{(l-1)}-\mathbf{x}_u^{(l-1)}\|+\varepsilon}$ is Lipschitz continuous with a constant $\frac{6K_{\mathbf{x}}^{(l-1)}}{\varepsilon}$ and we can bound:

$$\left\|\frac{\mathbf{x}_v^{(l-1)}(W)-\mathbf{x}_u^{(l-1)}(W)}{\|\mathbf{x}_v^{(l-1)}(W)-\mathbf{x}_u^{(l-1)}(W)\|+\varepsilon}-\frac{\mathbf{x}_v^{(l-1)}(\tilde{W})-\mathbf{x}_u^{(l-1)}(\tilde{W})}{\|\mathbf{x}_v^{(l-1)}(\tilde{W})-\mathbf{x}_u^{(l-1)}(\tilde{W})\|+\varepsilon}\right\|\leq\frac{6K_{\mathbf{x}}^{(l-1)}}{\varepsilon}\|W-\tilde{W}\|.$$

and consequently:

$$\begin{aligned}
&\|\mathbf{x}_v^{(l)}(W)-\mathbf{x}_v^{(l)}(\tilde{W})\|\\
&\qquad\leq\left(K_{\mathbf{x}}^{(l-1)}+\frac{6K_{\mathbf{x}}^{(l-1)}}{\varepsilon}L_\phi\|\mathbf{m}_{u\to v}^{(l-1)}(\tilde{W})\|\right)\|W-\tilde{W}\|\\
&\qquad+\frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)}\|\phi_x(\mathbf{m}_{u\to v}^{(l-1)}(W);W_{\phi_x})-\phi_x(\mathbf{m}_{u\to v}^{(l-1)}(\tilde{W});\tilde{W}_{\phi_x})\|
\end{aligned}$$

To bound the last term we perform additional telescoping step:

$$\begin{aligned}
&\|\phi_x(\mathbf{m}_{u\to v}^{(l-1)}(W);W_{\phi_x})-\phi_x(\mathbf{m}_{u\to v}^{(l-1)}(\tilde{W});\tilde{W}_{\phi_x})\|\\
&\qquad\leq\|\phi_x(\mathbf{m}_{u\to v}^{(l-1)}(W);W_{\phi_x})-\phi_x(\mathbf{m}_{u\to v}^{(l-1)}(W);\tilde{W}_{\phi_x})\|\\
&\qquad+\|\phi_x(\mathbf{m}_{u\to v}^{(l-1)}(W);\tilde{W}_{\phi_x})-\phi_x(\mathbf{m}_{u\to v}^{(l-1)}(\tilde{W});\tilde{W}_{\phi_x})\|\\
&\qquad\leq L_\phi\|\mathbf{m}_{u\to v}^{(l-1)}(W)\|\|W_{\phi_x}-\tilde{W}_{\phi_x}\|+L_\phi\|\mathbf{m}_{u\to v}^{(l-1)}(W)-\mathbf{m}_{u\to v}^{(l-1)}(\tilde{W})\|.
\end{aligned}$$

Now from Lemmas 4 and 8 we have $\|\mathbf{m}_{u\to v}^{(l-1)}(W)\|\leq 8L_\phi MK^{l-1}$ and from (62):

$$\|\mathbf{x}_v^{(l)}(W)-\mathbf{x}_v^{(l)}(\tilde{W})\|\leq\left(K_{\mathbf{x}}^{(l-1)}+\frac{48K_{\mathbf{x}}^{(l-1)}}{\varepsilon}L_\phi^2MK^{l-1}+12L_\phi^2MK^{l-1}+2L_\phi^2(K_{\mathbf{h}}^{(l-1)}+K_{\mathbf{x}}^{(l-1)})\right)\|W-\tilde{W}\|$$

In summary, we have the following recursive relationships:

$$\begin{aligned} K_{\mathbf{h}}^{(l)} &= 2\sqrt{2}L_\phi M K^{l-1} + L_\phi K_{\mathbf{h}}^{(l-1)} + L_\phi K_{\mathbf{m}}^{(l-1)} \\ K_{\mathbf{m}}^{(l)} &= 4DL_\phi M K^{l-1} + 2DL_\phi K_{\mathbf{h}}^{(l-1)} + 2DL_\phi K_{\mathbf{x}}^{(l-1)} \\ K_{\mathbf{x}}^{(l)} &= 12L_\phi^2 M K^{l-1} + (1 + 2L_\phi^2 + \frac{48L_\phi^2 M}{\varepsilon} K^{l-1}) K_{\mathbf{x}}^{(l-1)} + 2L_\phi^2 K_{\mathbf{h}}^{(l-1)}. \end{aligned} \tag{63}$$

We now determine the growth rates of the constants. Specifically, we will now show that:

$$\begin{aligned} K_{\mathbf{h}}^{(l)} &\leq M B^l K^{\frac{l(l-1)}{2}} \\ K_{\mathbf{m}}^{(l)} &\leq M B^l K^{\frac{l(l-1)}{2}} \\ K_{\mathbf{x}}^{(l)} &\leq M B^l K^{\frac{l(l-1)}{2}} \end{aligned} \tag{64}$$

for

$$B = \max\left\{\frac{96L_\phi^2 M}{\varepsilon}, 72L_\phi^2, 8DL_\phi\right\} \tag{65}$$

To show the base case, we recall that $K_{\mathbf{h}}^{(0)} = K_{\mathbf{x}}^{(0)} = 0$ and $K_{\mathbf{m}}^{(0)} = M$. Now, assuming that (64) holds for all $l' < l$, we will show that it also holds for l. Note that from (65) and the fact that $B \geq 1, K \geq 1$, the following hold for all l:

$$\begin{aligned} 2\sqrt{2}L_\phi K^{l-1} &\leq \frac{1}{2} B^l K^{\frac{l(l-1)}{2}} \\ 2L_\phi &\leq \frac{1}{2} B K^{l-1} \\ 4DL_\phi K^{l-1} &\leq \frac{1}{2} B^l K^{\frac{l(l-1)}{2}} \\ 4DL_\phi &\leq \frac{1}{2} B K^{l-1} \\ 12L_\phi^2 K^{l-1} &\leq \frac{1}{6} B^l K^{\frac{l(l-1)}{2}} \\ 1 + 2L_\phi^2 &\leq \frac{1}{6} B K^{l-1} \\ \frac{48L_\phi^2 M}{\varepsilon} &\leq \frac{1}{2} B \\ 2L_\phi^2 &\leq \frac{1}{6} B K^{l-1} \end{aligned}$$

and therefore:

$$\begin{aligned} K_{\mathbf{h}}^{(l)} &= 2\sqrt{2}L_\phi M K^{l-1} + L_\phi K_{\mathbf{h}}^{(l-1)} + L_\phi K_{\mathbf{m}}^{(l-1)} \leq 2\sqrt{2}L_\phi M K^{l-1} + 2L_\phi M B^{l-1} K^{\frac{(l-1)(l-2)}{2}} \\ &\leq \left(\frac{1}{2} + \frac{1}{2}\right) M B^l K^{\frac{l(l-1)}{2}} = M B^l K^{\frac{l(l-1)}{2}} \\ K_{\mathbf{m}}^{(l)} &= 4DL_\phi M K^{l-1} + 2DL_\phi K_{\mathbf{h}}^{(l-1)} + 2DL_\phi K_{\mathbf{x}}^{(l-1)} \leq 4DL_\phi M K^{l-1} + 4DL_\phi M B^{l-1} K^{\frac{(l-1)(l-2)}{2}} \\ &\leq \left(\frac{1}{2} + \frac{1}{2}\right) M B^l K^{\frac{l(l-1)}{2}} = M B^l K^{\frac{l(l-1)}{2}} \\ K_{\mathbf{x}}^{(l)} &= 12L_\phi^2 M K^{l-1} + (1 + 2L_\phi^2 + \frac{48L_\phi^2 M}{\varepsilon} K^{l-1}) K_{\mathbf{x}}^{(l-1)} + 2L_\phi^2 K_{\mathbf{h}}^{(l-1)} \\ &\leq 12L_\phi^2 M K^{l-1} + (1 + 2L_\phi^2 + \frac{48L_\phi^2 M}{\varepsilon} K^{l-1}) M B^{l-1} K^{\frac{(l-1)(l-2)}{2}} + 2L_\phi^2 M B^{l-1} K^{\frac{(l-1)(l-2)}{2}} \\ &\leq \left(\frac{1}{6} + \frac{1}{6} + \frac{1}{2} + \frac{1}{6}\right) M B^l K^{\frac{l(l-1)}{2}} = M B^l K^{\frac{l(l-1)}{2}}. \end{aligned}$$

□

F.4 GENERALIZATION BOUNDS OF THE SCORING MODEL

Now using results from Section F.3, we will derive the generalization bound for our scoring model defined in Section 3.2. Recall that the scoring model is defined as:

$$g(G^P, U^M) = \phi_{out}(\mathbf{h}^{(L)}(G^P), \pi(U^M)), \tag{66}$$

where $\mathbf{h}^{(L)}(G^P)$ is the average node embedding computed with EGNN on the graph representation of the protein pocket G^P, $\pi(U^M)$ is a vector of 4 features describing the unlabelled graph structure of the ligand U^M and ϕ_{out} is an MLP. In the following derivations, we will denote $\mathbf{h}^{(L)} \coloneqq \mathbf{h}^{(L)}(G^P)$ and $x \coloneqq \pi(U^M)$.

We first note that the average node embedding

$$\mathbf{h}^{(L)} = \frac{1}{|V|}\sum_{v \in V} \mathbf{h}_v^{(L)}$$

remains Lipschitz continuous w.r.t. EGNN's parameters with the same constant as each individual node embedding $\mathbf{h}_v^{(L)}$.

Lemma 11 (Lipschitz continuity w.r.t. parameters of the scoring model)**.** *Let g be the scoring model defined in (66). Assume that L-layer EGNN network satisfies assumptions in Lemma 10 and that ϕ_{out} is a $k_{out}-$layer MLP with a non-linearity having L_σ Lipschitz constant and weight matrices having spectral norm bounded by M_W. Assume further that the ligand feature vector is bounded $\|x\| \leq M_x$. Then, g is Lipschitz continuous w.r.t. model parameters:*

$$\|g(G^P, U^M; W) - g(G^P, U^M; \tilde{W})\| \leq L_{\phi_{out}} M_{out} B^L K^{\frac{L(L-1)}{2}} \|W - \tilde{W}\|,$$

where

$$\begin{aligned} M_{out} &= 2\sqrt{2}\max\{M_x, 2M\} \\ W &= (W_{\phi_x}, W_{\phi_e}, W_{\phi_h}, W_{\phi_{out}}) \\ L_{\phi_{out}} &= (L_\sigma M_W)^{k_{out}} \end{aligned}$$

and M, B, K are as defined in Lemma 10.

Proof.

$$\begin{aligned} \|g(G^P, U^M; W) - g(G^P, U^M; \tilde{W})\| &= \|\phi_{out}(\mathbf{h}^{(L)}(W), x; W_{\phi_{out}}) - \phi_{out}(\mathbf{h}^{(L)}(\tilde{W}_{\phi_{out}}), x; \tilde{W}_{\phi_{out}})\| \\ &\leq \|\phi_{out}(\mathbf{h}^{(L)}(W), x; W_{\phi_{out}}) - \phi_{out}(\mathbf{h}^{(L)}(W), x; \tilde{W}_{\phi_{out}})\| \\ &+ \|\phi_{out}(\mathbf{h}^{(L)}(W), x; \tilde{W}_{\phi_{out}}) - \phi_{out}(\mathbf{h}^{(L)}(\tilde{W}), x; \tilde{W}_{\phi_{out}})\| \\ &\leq L_{\phi_{out}}\sqrt{2}(M_x + MK^L)\|W_{\phi_{out}} - \tilde{W}_{\phi_{out}}\| + L_{\phi_{out}}\|\mathbf{h}^{(L)}(W) - \mathbf{h}^{(L)}(\tilde{W})\| \\ &\leq L_{\phi_{out}}(\sqrt{2}(M_x + MK^L) + MB^L K^{\frac{L(L-1)}{2}})\|W - \tilde{W}\| \\ &\leq L_{\phi_{out}} M_{out} B^L K^{\frac{L(L-1)}{2}} \|W - \tilde{W}\| \end{aligned}$$

□

Proposition 3 (Generalization bound of the scoring model)**.** *Let g be the scoring model satisfying assumptions from Lemma 11 and let $L(\widehat{y}, y) = \min\left\{\frac{1}{2}(\widehat{y} - y)^2, 1\right\}$ be the loss function. Then for any $\delta > 0$, with probability at least $1 - \delta$ over choosing a sample S of size m, the following holds:*

$$\mathcal{R}_S(g) \leq \frac{8}{m} + \frac{48}{\sqrt{m}}\sqrt{d^2(3kL + k_{out})\log\left(4\sqrt{dm}M_W(3kL + k_{out})\Lambda\right)} + 3\sqrt{\frac{\log\frac{2}{\delta}}{2m}}, \tag{67}$$

where

$$\begin{aligned}
\Lambda &= L_{\phi_{out}} M_{out} B^L K^{\frac{L(L-1)}{2}} \\
L_{\phi_{out}} &= (L_\sigma M_W)^{k_o ut} \\
M_{out} &= 2\sqrt{2}\max\{M_x, 2M\} \\
M &= 8DB^{(0)} L_\phi \\
L_\phi &= \max\{1, (L_\sigma M_w)^k\} \\
B &= \max\left\{\frac{96L_\phi^2 M}{\varepsilon}, 72L_\phi^2, 8DL_\phi\right\} \\
K &= \max\left\{2\sqrt{2}L_\phi, 1+16L_\phi^2, 4DL_\phi\left(1+2\sqrt{2}L_\phi+8L_\phi^2\right)\right\}
\end{aligned}$$

In particular

$$\mathcal{R}_S(g) \leq \mathcal{O}\left(\frac{L^2+k_{out}}{\sqrt{m}}\log\left(\frac{m(L+k_{out})}{\varepsilon}\right)\right) \tag{68}$$

Proof. Fix $r > 0$. We want to find a covering $\mathcal{C}(\mathcal{G}, r, \|\cdot\|)$, where $\mathcal{G} = \{g_W : \|W\| \leq M_W\}$ is the function class defined by a family of permissible scoring models. From Lemma 11:

$$\begin{aligned}
\sup_{(G^P, U^M)} \|g(G^P, U^M; W) - g(G^P, U^M; \tilde{W})\| &\leq L_{\phi_{out}} M_{out} B^L K^{\frac{L(L-1)}{2}} \|W - \tilde{W}\| \\
&\leq L_{\phi_{out}} M_{out} B^L K^{\frac{L(L-1)}{2}} \|W - \tilde{W}\|_F,
\end{aligned}$$

where the last inequality holds due to the relationship between spectral and Frobenius norm for matrices. Since there are $3kL$ weight matrices in the EGNN network and k_{out} layers in ϕ_{out}, it suffices to find a matrix covering for each weight matrix W_i:

$$\mathcal{C}\left(W_i, \frac{r}{(3kL+k_{out})\Lambda}, \|\cdot\|_F\right),$$

where $\Lambda = L_{\phi_{out}} M_{out} B^L K^{\frac{L(L-1)}{2}}$. Their Cartesian product will yield a covering of $\mathcal{G}$. Therefore, from Lemma 3:

$$\begin{aligned}
\mathcal{N}(\mathcal{G}, r, \|\cdot\|) &\leq \prod_i \mathcal{N}\left(W_i, \frac{r}{(3kL+k_{out})\Lambda}, \|\cdot\|_F\right) \\
&\leq \left(1 + 2\frac{\sqrt{d}M_W(3kL+k_{out})\Lambda}{r}\right)^{(3kL+k_{out})d^2},
\end{aligned} \tag{69}$$

where $d = \max_i dim(W_i)$, where for $A \in \mathbb{R}^{d_1 \times d_2}$, $dim(A) = \max(d_1, d_2)$. In the following we will be assuming r small enough so that $2\frac{\sqrt{d}M_W(3kL+k_{out})\Lambda}{r} > 1$, which implies

$$\log\mathcal{N}(\mathcal{G}, r, \|\cdot\|) \leq (3kL+k_{out})d^2 \log\left(4\frac{\sqrt{d}M_W(3kL+k_{out})\Lambda}{r}\right). \tag{70}$$

Since the loss function is bounded by 1 and 1-Lipschitz, we can use (51) to bound ERC of the function class of scoring models:

$$\widehat{\mathfrak{R}}_{S,L}(\mathcal{G}) \leq \inf_{\alpha>0}\left(\frac{4\alpha}{\sqrt{m}} + \frac{12}{m}\int_\alpha^{2\sqrt{m}} \sqrt{\log\mathcal{N}(\mathcal{G}, r, \|\cdot\|)}\mathrm{d}r\right). \tag{71}$$

Note that

$$\begin{aligned}\int_{\alpha}^{2\sqrt{m}} \sqrt{\log \mathcal{N}(\mathcal{G}, r, \|\cdot\|)}\mathrm{d}r &\leq d\sqrt{3kL+k_{out}} \int_{\alpha}^{2\sqrt{m}} \sqrt{\log\left(4\frac{\sqrt{d}M_W(3kL+k_{out})\Lambda}{r}\right)}\mathrm{d}r \\ &\leq d\sqrt{3kL+k_{out}} \int_{\alpha}^{2\sqrt{m}} \sqrt{\log\left(4\frac{\sqrt{d}M_W(3kL+k_{out})\Lambda}{\alpha}\right)}\mathrm{d}r \\ &\leq d\sqrt{3kL+k_{out}} \int_{0}^{2\sqrt{m}} \sqrt{\log\left(4\frac{\sqrt{d}M_W(3kL+k_{out})\Lambda}{\alpha}\right)}\mathrm{d}r \\ &= 2\sqrt{m}\sqrt{d^2(3kL+k_{out})\log\left(4\frac{\sqrt{d}M_W(3kL+k_{out})\Lambda}{\alpha}\right)}\end{aligned}$$

And choosing $\alpha = \frac{1}{\sqrt{m}}$ gives the bound on ERC:

$$\widehat{\mathfrak{R}}_{S,L}(\mathcal{G}) \leq \frac{4}{m} + \frac{24}{\sqrt{m}}\sqrt{d^2(3kL+k_{out})\log\left(4\sqrt{dm}M_W(3kL+k_{out})\Lambda\right)}.$$

Therefore, using Theorem 1, for any $\delta > 0$, with probability at least $1-\delta$ of choosing a sample of size m, the following holds:

$$\mathcal{R}_S(g) \leq \frac{8}{m} + \frac{48}{\sqrt{m}}\sqrt{d^2(3kL+k_{out})\log\left(4\sqrt{dm}M_W(3kL+k_{out})\Lambda\right)} + 3\sqrt{\frac{\log\frac{2}{\delta}}{2m}},$$

where $\Lambda = L_{\phi_{out}}M_{out}B^L K^{\frac{L(L-1)}{2}}$. After substituting for all constants, one can see that:

$$\mathcal{R}_S(g) \leq \mathcal{O}\left(\frac{L^2+k_{out}}{\sqrt{m}}\log\left(\frac{m(L+k_{out})}{\varepsilon}\right)\right).$$

□

F.5 Impact of ε-soft normalization

In our discussion, we used an E(3)-Equivariant Graph Neural Network (EGNN) as defined in (53), which differs from the original formulation in (Satorras et al., 2021). Namely, the update of the coordinate embedding did not use ε-soft normalization:

$$\mathbf{x}_v^{(l)} = \mathbf{x}_v^{(l-1)} + \frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)}\left(\mathbf{x}_v^{(l-1)} - \mathbf{x}_u^{(l-1)}\right)\phi_x(\mathbf{m}_{u\to v}^{(l-1)}). \tag{72}$$

The authors do include the normalized version of the model in the official implementation[2], but do not use it in any experiments. We first note that there is no disparity between the two formulations concerning their expressivity or equivariant characteristics. Both network versions possess identical expressivity, given that the messages $\mathbf{m}_{u\to v}^{(l)}$ depend on the distances between coordinate embeddings, $\|\mathbf{x}_v^{(l-1)} - \mathbf{x}_u^{(l-1)}\|$. In a similar vein, the model with normalization remains E(3)-equivariant due to $\|\mathbf{x}_v^{(l-1)} - \mathbf{x}_u^{(l-1)}\|$ being E(3)-invariant.

However, we informally argue that the version of the model with normalization may benefit from better generalization guarantees. As can be seen from the proof of Lemma 8, the reason we were able to obtain a bound $\|\mathbf{h}_v^{(l)}\| \leq MK^l$ was because the l-th layer bound $B^{(l)}$ satisfied the following recursive relationship:

$$B^{(l)} \leq KB^{(l-1)}. \tag{73}$$

In the absence of ε-soft normalization, the recursion would instead be:

$$B^{(l)} \leq \tilde{K}\left[B^{(l-1)}\right]^2, \tag{74}$$

[2]https://github.com/vgsatorras/egnn

leading to a bound:

$$B^{(l)} = \tilde{M}\tilde{K}^{2^l}. \tag{75}$$

Consequently, the reasoning in Lemma 10 would yield a Lipschitz constant of the order $\mathcal{O}(\tilde{K}^{2^L})$ instead of $\mathcal{O}(K^{L^2})$, which in turn would lead to a generalization bound exponential in L as opposed to polynomial as we showed in the case of ε-soft normalized version of the model.