FlashAffinity: Bridging the Accuracy-Speed Gap in Protein-Ligand Binding Affinity Prediction

Songlin Jiang[1], **Yifan Chen**[1], **Ze Cao**[2], **Wengong Jin**[*1]
[1]Khoury College of Computer Sciences, Northeastern University
[2]Department of Molecular Biology, Princeton University
{jiang.song, chen.y60, w.jin}@northeastern.edu

Abstract

Accurate prediction of protein-ligand binding affinity is central to computational drug discovery. Recent foundation models, such as Boltz-2, have achieved remarkable accuracy, but their high computational cost poses a major barrier to large-scale virtual screening. We address this challenge by introducing a lightweight structure-based virtual screening model, **FlashAffinity**, that achieves similar performance as Boltz-2 in affinity prediction and binder classification tasks, while achieving a 50x speedup at inference time. FlashAffinity replaces the expensive protein structure prediction models with a simple protein-ligand docking model and the PairFormer-based affinity scoring module with a cheap EGNN architecture. In summary, this work bridges the gap between accuracy and efficiency, enabling ultra-fast virtual screening of massive chemical libraries.

1 Introduction

Accurate prediction of protein-ligand binding affinity is a cornerstone of computational drug discovery. This measure quantifies the interaction strength between a small molecule and its protein target, directly informing its potential to modulate biological function. Consequently, reliable affinity prediction is essential for prioritizing lead compounds, guiding molecular optimization, and accelerating the progression of therapeutic candidates.

A longstanding challenge in the field is the inherent trade-off between predictive accuracy and computational efficiency. Traditional computational methods have historically forced a compromise: they are either too slow for screening million-scale chemical libraries or lack the accuracy needed for reliable lead optimization, often because they do not fully leverage the rich 3D information of the protein-ligand complex.

Recent advances in structural foundation models are beginning to bridge this gap. Notably, **Boltz-2** [1] represents a significant milestone, delivering binding affinity predictions that approach the accuracy of physics-based methods while being orders of magnitude more computationally efficient. Its ability to model dynamic conformational ensembles has established a new state-of-the-art in accuracy for AI-driven methods.

Despite this remarkable progress, the inference time of Boltz-2 still presents a major barrier to its application in large-scale virtual screening. The screening of ultra-large chemical libraries, which now contain millions of molecules, demands inference speeds that are orders of magnitude faster than what even leading models currently offer. To address this challenge, we introduce a lightweight, structure-based model designed to retain the predictive power of methods like Boltz-2 while drastically reducing computational overhead. Specifically, FlashAffinity replaces the expensive protein structure prediction models with a simple protein-ligand docking model and the PairFormer-based affinity scoring module with a cheap equivariant graph neural network (EGNN)-based architecture [2].

Machine Learning for Structural Biology Workshop, 2025.

Our primary contributions are as follows:

- **A Lightweight and Efficient Architecture:** We propose a computationally efficient model based on a simple combination of docking and an EGNN-based scoring function, which captures essential protein-ligand structural interactions while minimizing computational overhead.
- **Near State-of-the-Art Performance:** Our model achieves performance approaching that of Boltz-2 on key tasks, demonstrating that competitive accuracy can be maintained within a lightweight framework.
- **50x Speedup for Large-Scale Virtual Screening:** The model provides a 50x inference speedup over Boltz-2. This efficiency unlocks the ability to perform high-fidelity virtual screening on massive chemical libraries, directly bridging the gap between state-of-the-art prediction and practical drug discovery workflows.

In the following sections, we describe our model architecture, data curation strategy, and present comprehensive experimental results that validate its accuracy and efficiency against leading benchmarks.

2 Methodology

2.1 Problem Description and Notations

Computational methods play a pivotal role in modern drug discovery by accelerating the identification and optimization of potential therapeutic molecules. Our work addresses two critical stages of this process: hit discovery and lead optimization. We formulate these challenges as distinct but related machine learning tasks that leverage the 3D structure of protein-ligand complexes.

Hit discovery constitutes the initial screening phase, which aims to identify novel active compounds, or binders, from vast chemical libraries. This stage prioritizes distinguishing these active compounds from inactive decoys over precisely quantifying their binding strength. Formally, this corresponds to a **binary classification task**.

Subsequently, the lead optimization stages involve refining a promising series of compounds to improve properties such as binding affinity. This requires the model to discern subtle activity differences among structurally similar molecules, a challenge that corresponds to a **regression task** for predicting precise, quantitative affinity measurements like K_i, K_d, or IC_{50}.

To address these distinct tasks, we designed an end-to-end prediction pipeline that transforms raw molecular data into accurate affinity predictions. The process commences with a protein-ligand pair, formally denoted as $(\mathcal{S}_p, \mathcal{S}_l)$, where $\mathcal{S}_p$ is the protein's amino acid sequence and $\mathcal{S}_l$ is the ligand's SMILES string. For inputs lacking experimental structures, a 3D complex is generated via docking (Appendix B.2). The resulting structure $(\mathbf{X}, \mathbf{H})$ is localized to the binding site using a cropping function $\mathcal{F}_{\text{crop}}$, and then converted into a molecular graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}, \mathbf{h}, \mathbf{x})$. This graph serves as the input to our core E(3)-equivariant graph neural network (EGNN) [2], denoted ϕ, whose output is passed to task-specific prediction heads for either affinity regression (y_{affinity}) or binary classification (p_{bind}).

2.2 Our Method

Our method is a lightweight and efficient framework built upon a carefully designed equivariant graph representation. We first isolate the protein's binding pocket, construct a rich multi-relational graph of the protein-ligand complex, and finally process this graph with an EGNN to predict binding affinity. All technical details, parameters, and mathematical formulations are provided in Appendix C.

Graph Construction and Representation. To focus on the relevant binding interface and ensure computational efficiency, we first employ an adaptive cropping function, $\mathcal{F}_{\text{crop}}$, which isolates a localized pocket region around the ligand based on spatial proximity (Appendix C.1). From this cropped structure, we build a multi-relational graph where atoms are nodes. Node features are a rich composite of pre-trained protein embeddings, ligand chemical properties, and categorical encodings (Appendix B.3). Inspired by MEAN [3], the graph's edge schema is designed to capture interactions at multiple scales: **internal edges** model covalent topology, **external edges** model non-covalent

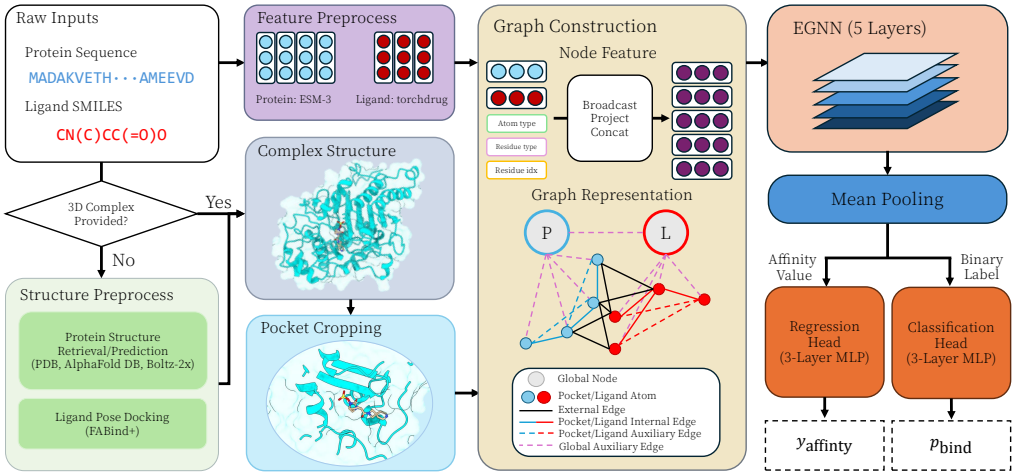

Figure 1: **The overview of our method**. Raw inputs undergo structure and feature preprocessing to generate a protein-ligand graph. This graph is processed by an E(3)-Equivariant Graph Neural Network (EGNN) [2] The network's final representation is then directed to a single, task-specific prediction head. The regression task uses a dedicated head to predict affinity values (y_{affinity}), while the classification task uses a separate head to predict binary labels (p_{bind}).

protein-ligand interactions, and **auxiliary edges** provide global structural context via global nodes (Appendix C.2).

Equivariant Network and Training. The core of our model is an E(3)-equivariant graph neural network (EGNN), which directly processes the 3D geometry of the graph while preserving rotational and translational symmetries. The network consists of a stack of $L = 5$ equivariant layers, and its final pooled representation is passed to a task-specific MLP head for prediction (Appendix E). To handle noisy biochemical data, we train our models using a group-based mini-batch sampling strategy, where samples in each group are drawn from the same experimental assay (Appendix C.3). We use task-specific loss functions—Focal Loss [4] for classification and a weighted Huber loss [5] for regression (Appendix C.4). At inference, predictive robustness is enhanced using a snapshot ensembling [6] technique (Appendix C.5).

Dataset	Method	Pearson R ↑	Kendall tau ↑	PMAE ↓	MAE ↓		PW1 ↑		PW2 ↑	
					non-cent.	cent.	non-cent.	cent.	non-cent.	cent.
OpenFE	Boltz-2	0.62	0.46	0.93	1.22	0.64	0.49	0.80	0.82	0.96
	BACPI	0.29	0.19	1.21	1.44	0.85	0.40	0.67	0.74	0.94
	GAT	0.28	0.20	1.30	1.42	0.91	0.40	0.64	0.75	0.92
	FlashAffinity	0.44	0.33	1.13	1.46	0.79	0.39	0.71	0.71	0.95
FEP+ 4 targets	Boltz-2	0.66	0.48	0.85	0.75	0.59	0.69	0.83	0.97	0.98
	BACPI	0.14	0.09	1.18	1.40	0.82	0.43	0.62	0.73	1.00
	GAT	0.40	0.28	1.07	1.19	0.71	0.43	0.72	0.86	0.95
	FlashAffinity	0.53	0.38	1.10	1.44	0.76	0.39	0.71	0.74	0.95
CASP16	Boltz-2	0.65	0.45	1.36	1.28	0.95	0.48	0.61	0.81	0.90
	BACPI	0.41	0.31	1.55	1.25	1.10	0.45	0.51	0.81	0.89
	GAT	0.50	0.35	1.58	1.28	1.13	0.44	0.49	0.79	0.84
	FlashAffinity	0.65	0.51	1.14	1.21	0.88	0.29	0.68	0.94	0.94

Table 1: Comprehensive performance comparison on affinity value prediction benchmarks. Metrics are averaged per-assay. "non-cent." and "cent." denote metrics computed on raw and centered predictions, respectively. PW1/PW2 refer to the percentage of predictions within 1 and 2 kcal/mol of the experimental value. Performance data for all baseline models are sourced from the original Boltz-2 publication.

Table 2: Performance comparison on the MF-PCBA benchmark for binder classification. Our model demonstrates competitive performance, particularly in enrichment factors, compared to established baselines. All baseline data is taken from original Boltz-2 publication.

Method	AP ↑	AUROC ↑		EF at 0.5% ↑	EF at 1% ↑	EF at 2% ↑	EF at 5% ↑
		target avg.	global				
Boltz-2	0.0248	0.8122	0.8056	18.3916	13.9540	10.5706	7.0448
BACPI	0.0131	0.7575	0.7205	9.4818	9.2397	7.3983	5.5533
GAT	0.0133	0.7928	0.7867	11.1279	8.9897	7.5630	5.9055
Chemgauss4	0.0051	0.5450	0.5706	1.9969	2.2257	2.1136	1.6462
Boltz-2 iptm	0.0046	0.5657	0.6134	2.4242	3.1728	2.6881	2.2263
FlashAffinity	0.0192	0.7762	0.7826	16.2832	14.1343	9.4587	6.7300

Table 3: Inference time comparison per protein-ligand pair on a single NVIDIA L40S GPU. Our full pipeline offers a 50x speedup over Boltz-2, while our core scoring model is orders of magnitude faster.

Method	Avg. Inference Time (ms)
Boltz-2	35,000
BACPI	0.65
GAT	0.28
FlashAffinity (Full Pipeline)	700
FlashAffinity (Scoring Only)	2.5

3 Experiments

3.1 Experimental Setup

To ensure a fair and direct comparison with the state-of-the-art, our evaluation strictly adheres to the protocol of Boltz-2 [1], using identical test sets and metrics. We evaluate our model's performance on two critical drug discovery tasks: binding affinity prediction and binder classification. For affinity prediction, we benchmark on the widely recognized **OpenFE** [7], **a 4-target FEP benchmark (CDK2, TYK2, JNK1, P38)** curated in the protein-ligand-benchmark [8], and **CASP16** [9] datasets. For classification, we use the **MF-PCBA** [10] benchmark. We primarily compare our model against Boltz-2. A comprehensive description of the datasets and evaluation metrics can be found in Appendices B and D, respectively.

3.2 Main Results

Hereafter, we refer to our proposed lightweight model as **FlashAffinity**, reflecting its high speed and function as a scoring model. We focus our comparison on machine learning (ML) based methods to ensure a direct and methodologically consistent evaluation.

We selected three key ML baselines from the Boltz-2 study: **Boltz-2**, the state-of-the-art foundation model representing the performance upper bound; **BACPI** [11], a sequence-based model to quantify the gains from using 3D structure; **GAT**, a ligand-only model to assess dataset biases;

Results on Affinity Value Prediction The full results are presented in Table 1. Our structure-based model consistently and significantly outperforms the non-structural baselines across all datasets. For instance, on the FEP+ 4 targets benchmark, our model achieves a Pearson's R of 0.53, a substantial improvement over BACPI (0.14). This demonstrates the effectiveness of our architecture in leveraging 3D information for affinity prediction.

When compared to the larger Boltz-2 model, our lightweight approach achieves competitive and often comparable results. This is particularly evident on the blind CASP16 benchmark, where our model's performance on rank correlation matches and even slightly exceeds that of Boltz-2 (Kendall tau of 0.51 vs. 0.45), highlighting its strong generalization capability in a challenging setting. While Boltz-2 generally shows lower error on the other benchmarks, our model's performance remains close, validating its position as an efficient and powerful alternative for rapid affinity estimation.

Results on Binary Classification For the binary classification task, which mimics a real-world virtual screening scenario, In addition to the aforementioned ML baselines, we include two more for broader context: **Chemgauss4** [12], a traditional, non-ML docking score from OpenEye FRED [13], and **Boltz-2 iptm**, the structure module's interface TM-score, to test if a simple structural confidence score is sufficient for ranking.

The results, summarized in Table 2, demonstrate the strong performance of our model in identifying binders from a large pool of decoys. Our model dramatically outperforms both the traditional docking

score (Chemgauss4) and the structural confidence metric (Boltz-2 iptm) across all reported metrics, underscoring the value of an end-to-end trained, structure-aware scoring function.

Compared to other machine learning methods, our model again shows the benefit of its 3D-aware architecture, achieving a higher Average Precision (AP) and superior enrichment factors than both the sequence-based BACPI and ligand-only GAT models. This highlights our model's enhanced ability to correctly rank true binders at the top of the list—the most critical outcome for an effective virtual screen.

Finally, our lightweight model is highly competitive with the state-of-the-art Boltz-2 model. To specifically isolate and compare the performance of the scoring architectures, we conducted an ablation study where both models used identical input structures generated by FABind+ (see Appendix F, Table S1). Crucially, for the task of early hit discovery, our model shows exceptional performance in enrichment, nearly matching Boltz-2 at the top 0.5% and even slightly surpassing it at the top 1% threshold (EF@1% of 14.13 vs. 13.95). These results validate our model as a powerful and efficient tool for large-scale virtual screening campaigns, where robust early enrichment is paramount.

Speed A primary advantage of our lightweight approach is the significant reduction in computational cost compared to large foundation models, enabling practical high-throughput virtual screening. To quantify this, we benchmarked the average inference time per protein-ligand pair on a single NVIDIA L40S GPU. The results, averaged over 100 randomly selected pairs, are summarized in Table 3.

As shown in the table, our model's full end-to-end pipeline requires approximately 700 milliseconds (0.7 seconds) per prediction. This achieves a 50-fold speedup over Boltz-2 (35,000 ms), making large-scale screening campaigns computationally feasible. The majority of this time is attributed to the initial structure generation, as the FABind+ docking step itself takes nearly 0.7 seconds.

Crucially, we distinguish a second, highly relevant use-case: scoring a pre-processed library. In scenarios where protein-ligand complex structures are already generated—a common workflow for screening large enumerated libraries—our core EGNN scoring model is exceptionally fast, requiring only 2.5 milliseconds per prediction. While sequence-based models like GAT and BACPI offer even faster inference, our method provides a vital trade-off, delivering substantially higher accuracy (as demonstrated in previous sections) for a computational cost that remains practical for screening millions of compounds. This efficiency firmly establishes our model as a powerful tool for rapidly prioritizing candidates from vast chemical spaces.

4 Conclusion

We introduced a lightweight, structure-based framework for high-throughput virtual screening that addresses the accuracy-efficiency trade-off prohibitive in large foundation models like Boltz-2 [1]. Our simple combination of a fast docking model and an EGNN-based [2] scoring function achieves near state-of-the-art performance on both affinity prediction and binder classification, while offering up to a 50x inference speedup. Crucially, our work demonstrates that a well-trained EGNN, when provided with a reasonably docked pose in a correctly identified pocket, can achieve performance that approaches the results of more complex and computationally expensive architectures. This highlights the foundational importance of rigorous data curation and offers a practical and efficient path toward the accurate screening of massive chemical libraries.

References

[1] Saro Passaro, Gabriele Corso, Jeremy Wohlwend, Mateo Reveiz, Stephan Thaler, Vignesh Ram Somnath, Noah Getz, Tally Portnoi, Julien Roy, Hannes Stark, David Kwabi-Addo, Dominique Beaini, Tommi Jaakkola, and Regina Barzilay. Boltz-2: Towards accurate and efficient binding affinity prediction. *bioRxiv*, 2025. doi: 10.1101/2025.06.14.659707. URL https://www.biorxiv.org/content/early/2025/06/18/2025.06.14.659707. (Cited on pages 1, 4, 5, 12, 13, 15, 19, 21 and 23)

[2] Victor Garcia Satorras, Emiel Hoogeboom, and Max Welling. E(n) equivariant graph neural networks, 2022. URL https://arxiv.org/abs/2102.09844. (Cited on pages 1, 2, 3, 5, 15, 21, 22 and 23)

[3] Xiangzhe Kong, Wenbing Huang, and Yang Liu. Conditional antibody design as 3d equivariant graph translation, 2023. URL https://arxiv.org/abs/2208.06073. (Cited on pages 2 and 17)

[4] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection, 2018. URL https://arxiv.org/abs/1708.02002. (Cited on pages 3 and 19)

[5] Peter J. Huber. *Robust Estimation of a Location Parameter*, pages 492–518. Springer New York, New York, NY, 1992. ISBN 978-1-4612-4380-9. doi: 10.1007/978-1-4612-4380-9_35. URL https://doi.org/10.1007/978-1-4612-4380-9_35. (Cited on pages 3 and 19)

[6] Gao Huang, Yixuan Li, Geoff Pleiss, Zhuang Liu, John E. Hopcroft, and Kilian Q. Weinberger. Snapshot ensembles: Train 1, get m for free, 2017. URL https://arxiv.org/abs/1704.00109. (Cited on pages 3 and 20)

[7] Richard J. Gowers, Irfan Alibay, David W.H. Swenson, Michael M. Henry, Benjamin Ries, Hannah M. Baumann, and James R. B. Eastwood. The open free energy library, September 2023. URL https://doi.org/10.5281/zenodo.8344248. (Cited on pages 4 and 13)

[8] David F. Hahn, Christopher I. Bayly, Melissa L Boby, Hannah E. Bruce Macdonald, John D. Chodera, Vytautas Gapsys, Antonia S. J. S. Mey, David L. Mobley, Laura Perez Benito, Christina E. M. Schindler, Gary Tresadern, and Gregory L. Warren. Best practices for constructing, preparing, and evaluating protein-ligand binding affinity benchmarks [article v1.0]. *Living Journal of Computational Molecular Science*, 4(1), 2022. ISSN 2575-6524. doi: 10.33011/livecoms.4.1.1497. URL http://dx.doi.org/10.33011/livecoms.4.1.1497. (Cited on pages 4 and 13)

[9] M. K. Gilson, J. Eberhardt, P. Škrinjar, J. Durairaj, X. Robin, and A. Kryshtafovych. Assessment of pharmaceutical protein-ligand pose and affinity predictions in casp16. 2025. doi: 10.22541/au.174562565.51283311/v1. (Cited on pages 4 and 13)

[10] David Buterez, Jon Paul Janet, Steven J. Kiddle, and Pietro Liò. Mf-pcba: Multifidelity high-throughput screening benchmarks for drug discovery and machine learning. *Journal of Chemical Information and Modeling*, 63(9):2667–2678, 2023. doi: 10.1021/acs.jcim.2c01569. URL https://doi.org/10.1021/acs.jcim.2c01569. PMID: 37058588. (Cited on pages 4 and 13)

[11] Min Li, Zhangli Lu, Yifan Wu, and Yaohang Li. Bacpi: a bi-directional attention neural network for compound-protein interaction and binding affinity prediction. *Bioinformatics*, 38, 01 2022. doi: 10.1093/bioinformatics/btac035. (Cited on page 4)

[12] Yoshio Nishimoto and Dmitri G. Fedorov. The fragment molecular orbital method combined with density-functional tight-binding and the polarizable continuum model. *Phys. Chem. Chem. Phys.*, 18:22047–22061, 2016. doi: 10.1039/C6CP02186G. URL http://dx.doi.org/10.1039/C6CP02186G. (Cited on page 4)

[13] Mark McGann. Fred pose prediction and virtual screening accuracy. *Journal of chemical information and modeling*, 51:578–96, 02 2011. doi: 10.1021/ci100436p. (Cited on page 4)

[14] 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, 2009. URL https://api.semanticscholar.org/CorpusID:30245244. (Cited on page 12)

[15] Marcel L. Verdonk, Jason C. Cole, Michael J. Hartshorn, Christopher W. Murray, and Richard D. Taylor. Improved protein–ligand docking using gold. *Proteins: Structure*, 52, 2003. URL https://api.semanticscholar.org/CorpusID:14634843.

[16] Thomas A. Halgren, Robert B. Murphy, Richard A. Friesner, Hege S. Beard, Leah L. Frye, W. Thomas Pollard, and Jay L. Banks. Glide: A new approach for rapid, accurate docking and scoring. 2. enrichment factors in database screening. *Journal of Medicinal Chemistry*, 47(7): 1750–1759, 2004. doi: 10.1021/jm030644s. URL https://doi.org/10.1021/jm030644s. PMID: 15027866. (Cited on pages 12 and 23)

[17] Talia B. Kimber, Yonghui Chen, and Andrea Volkamer. Deep learning in virtual screening: Recent applications and developments. *International Journal of Molecular Sciences*, 22(9), 2021. ISSN 1422-0067. doi: 10.3390/ijms22094435. URL https://www.mdpi.com/1422-0067/22/9/4435. (Cited on page 12)

[18] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. URL https://arxiv.org/abs/2103.00020. (Cited on page 12)

[19] Bowen Gao, Bo Qiang, Haichuan Tan, Minsi Ren, Yinjun Jia, Minsi Lu, Jingjing Liu, Weiying Ma, and Yanyan Lan. Drugclip: Contrastive protein-molecule representation learning for virtual screening, 2023. URL https://arxiv.org/abs/2310.06367. (Cited on page 12)

[20] Mostafa Karimi, Di Wu, Zhangyang Wang, and Yang Shen. Deepaffinity: interpretable deep learning of compound–protein affinity through unified recurrent and convolutional neural networks. *Bioinformatics*, 35(18):3329–3338, February 2019. ISSN 1367-4811. doi: 10.1093/bioinformatics/btz111. URL http://dx.doi.org/10.1093/bioinformatics/btz111. (Cited on page 12)

[21] Hakime Öztürk, Arzucan Özgür, and Elif Ozkirimli. Deepdta: deep drug–target binding affinity prediction. *Bioinformatics*, 34(17):i821–i829, 09 2018. ISSN 1367-4803. doi: 10.1093/bioinformatics/bty593. URL https://doi.org/10.1093/bioinformatics/bty593. (Cited on page 12)

[22] Shuke Zhang, Yanzhao Jin, Tianmeng Liu, Qi Wang, Zhaohui Zhang, Shuliang Zhao, and Bo Shan. Ss-gnn: A simple-structured graph neural network for affinity prediction, 2022. URL https://arxiv.org/abs/2206.07015. (Cited on page 12)

[23] Wei Lu, Qifeng Wu, Jixian Zhang, Jiahua Rao, Chengtao Li, and Shuangjia Zheng. Tankbind: Trigonometry-aware neural networks for drug-protein binding structure prediction. *bioRxiv*, 2022. doi: 10.1101/2022.06.06.495043. URL https://www.biorxiv.org/content/early/2022/06/06/2022.06.06.495043.

[24] Shuqi Lu, Zhifeng Gao, Di He, Linfeng Zhang, and Guolin Ke. Highly accurate quantum chemical property prediction with uni-mol+, 2023. URL https://arxiv.org/abs/2303.16982.

[25] Huan Yee Koh, Anh T.N. Nguyen, Shirui Pan, Lauren T. May, and Geoffrey I. Webb. Psichic: physicochemical graph neural network for learning protein-ligand interaction fingerprints from sequence data. *bioRxiv*, 2023. doi: 10.1101/2023.09.17.558145. URL https://www.biorxiv.org/content/early/2023/09/19/2023.09.17.558145. (Cited on page 12)

[26] Kang Zhang, Xin Yang, Yifei Wang, Yunfang Yu, Niu Huang, Gen Li, Xiaokun Li, Joseph Wu, and Shengyong Yang. Artificial intelligence in drug development. *Nature Medicine*, 31, 01 2025. doi: 10.1038/s41591-024-03434-4. (Cited on page 12)

[27] Limei Wang, Haoran Liu, Yi Liu, Jerry Kurtin, and Shuiwang Ji. Learning hierarchical protein representations via complete 3d graph networks, 2023. URL https://arxiv.org/abs/2207.12600.

[28] Bowen Gao, Yinjun Jia, Yuanle Mo, Yuyan Ni, Weiying Ma, Zhiming Ma, and Yanyan Lan. Profsa: Self-supervised pocket pretraining via protein fragment-surroundings alignment, 2024. URL https://arxiv.org/abs/2310.07229. (Cited on page 12)

[29] Krinos Li, Xianglu Xiao, Zijun Zhong, and Guang Yang. Accurate and generalizable protein-ligand binding affinity prediction with geometric deep learning, 2025. URL https://arxiv.org/abs/2504.16261. (Cited on page 12)

[30] Kairi Furui and Masahito Ohue. Boltzina: Efficient and accurate virtual screening via docking-guided binding prediction with boltz-2, 2025. URL https://arxiv.org/abs/2508.17555. (Cited on page 12)

[31] Sunghwan Kim, Jie Chen, Tiejun Cheng, Asta Gindulyte, Jia He, Siqian He, Qingliang Li, Benjamin A Shoemaker, Paul A Thiessen, Bo Yu, Leonid Zaslavsky, Jian Zhang, and Evan E Bolton. Pubchem 2023 update. *Nucleic Acids Research*, 51(D1):D1373–D1380, 10 2022. ISSN 0305-1048. doi: 10.1093/nar/gkac956. URL https://doi.org/10.1093/nar/gkac956. (Cited on page 12)

[32] Pablo Lemos, Zane Beckwith, Sasaank Bandi, Maarten van Damme, Jordan Crivelli-Decker, Benjamin J. Shields, Thomas Merth, Punit K. Jha, Nicola De Mitri, Tiffany J. Callahan, AJ Nish, Paul Abruzzo, Romelia Salomon-Ferrer, and Martin Ganahl. Sair: Enabling deep learning for protein-ligand interactions with a synthetic structural dataset. *bioRxiv*, 2025. doi: 10.1101/2025.06.17.660168. URL https://www.biorxiv.org/content/early/2025/06/21/2025.06.17.660168. (Cited on page 12)

[33] Jeremy Wohlwend, Gabriele Corso, Saro Passaro, Mateo Reveiz, Ken Leidal, Wojtek Swiderski, Tally Portnoi, Itamar Chinn, Jacob Silterra, Tommi Jaakkola, and Regina Barzilay. Boltz-1 democratizing biomolecular interaction modeling. *bioRxiv*, 2024. doi: 10.1101/2024.11.19.624167. URL https://www.biorxiv.org/content/early/2024/11/20/2024.11.19.624167. (Cited on page 12)

[34] Viet-Khoa Tran-Nguyen, Célien Jacquemard, and Didier Rognan. Lit-pcba: An unbiased dataset for machine learning and virtual screening. *Journal of Chemical Information and Modeling*, XXXX, 04 2020. doi: 10.1021/acs.jcim.0c00155. (Cited on page 13)

[35] Jonathan Bayldon Baell and Georgina A. Holloway. New substructure filters for removal of pan assay interference compounds (pains) from screening libraries and for their exclusion in bioassays. *Journal of medicinal chemistry*, 53 7:2719–40, 2010. URL https://api.semanticscholar.org/CorpusID:18795270. (Cited on page 13)

[36] The UniProt Consortium. Uniprot: the universal protein knowledgebase in 2025. *Nucleic Acids Research*, 53(D1):D609–D617, 11 2024. ISSN 1362-4962. doi: 10.1093/nar/gkae1010. URL https://doi.org/10.1093/nar/gkae1010. (Cited on page 14)

[37] David Rogers and Mathew Hahn. Extended-connectivity fingerprints. *Journal of chemical information and modeling*, 50(5):742–754, 2010. (Cited on page 14)

[38] Martin Buttenschoen, Garrett M. Morris, and Charlotte M. Deane. Posebusters: Ai-based docking methods fail to generate physically valid poses or generalise to novel sequences. *Chemical Science*, 15:3130 – 3139, 2023. URL https://api.semanticscholar.org/CorpusID:260865809. (Cited on page 14)

[39] H M Berman, J Westbrook, Z Feng, G Gilliland, T N Bhat, H Weissig, I N Shindyalov, and P E Bourne. The protein data bank. *Nucleic Acids Res.*, 28(1):235–242, January 2000. (Cited on page 15)

[40] Mihaly Varadi, Stephen Anyango, Mandar Deshpande, Sreenath Nair, Cindy Natassia, Galabina Yordanova, David Yuan, Oana Stroe, Gemma Wood, Agata Laydon, Augustin Žídek, Tim Green, Kathryn Tunyasuvunakool, Stig Petersen, John Jumper, Ellen Clancy, Richard Green,

Ankur Vora, Mira Lutfi, Michael Figurnov, Andrew Cowie, Nicole Hobbs, Pushmeet Kohli, Gerard Kleywegt, Ewan Birney, Demis Hassabis, and Sameer Velankar. Alphafold protein structure database: massively expanding the structural coverage of protein-sequence space with high-accuracy models. *Nucleic Acids Research*, 50(D1):D439–D444, 11 2021. ISSN 0305-1048. doi: 10.1093/nar/gkab1061. URL https://doi.org/10.1093/nar/gkab1061. (Cited on page 15)

[41] Kaiyuan Gao, Qizhi Pei, Gongbo Zhang, Jinhua Zhu, Kun He, and Lijun Wu. Fabind+: Enhancing molecular docking through improved pocket prediction and pose generation. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.1*, KDD '25, page 330–341. ACM, July 2025. doi: 10.1145/3690624.3709253. URL http://dx.doi.org/10.1145/3690624.3709253. (Cited on pages 15, 18, 21 and 23)

[42] Renxiao Wang, Xueliang Fang, Yipin Lu, Chao-Yie Yang, and Shaomeng Wang. The PDBbind database: methodologies and updates. *J. Med. Chem.*, 48(12):4111–4119, June 2005. (Cited on page 15)

[43] Gabriele Corso, Hannes Stärk, Bowen Jing, Regina Barzilay, and Tommi Jaakkola. Diffdock: Diffusion steps, twists, and turns for molecular docking, 2023. URL https://arxiv.org/abs/2210.01776. (Cited on page 15)

[44] Thomas Hayes, Roshan Rao, Halil Akin, Nicholas J. Sofroniew, Deniz Oktay, Zeming Lin, Robert Verkuil, Vincent Q. Tran, Jonathan Deaton, Marius Wiggert, Rohil Badkundri, Irhum Shafkat, Jun Gong, Alexander Derry, Raul S. Molina, Neil Thomas, Yousuf A. Khan, Chetan Mishra, Carolyn Kim, Liam J. Bartie, Matthew Nemeth, Patrick D. Hsu, Tom Sercu, Salvatore Candido, and Alexander Rives. Simulating 500 million years of evolution with a language model. *Science*, 387(6736):850–858, 2025. doi: 10.1126/science.ads0018. URL https://www.science.org/doi/abs/10.1126/science.ads0018. (Cited on page 15)

[45] Zhaocheng Zhu, Chence Shi, Zuobai Zhang, Shengchao Liu, Minghao Xu, Xinyu Yuan, Yangtian Zhang, Junkun Chen, Huiyu Cai, Jiarui Lu, Chang Ma, Runcheng Liu, Louis-Pascal Xhonneux, Meng Qu, and Jian Tang. Torchdrug: A powerful and flexible machine learning platform for drug discovery, 2022. URL https://arxiv.org/abs/2202.08320. (Cited on page 16)

[46] Greg Landrum, Paolo Tosco, Brian Kelley, Ricardo Rodriguez, David Cosgrove, Riccardo Vianello, sriniker, Peter Gedeck, Gareth Jones, Eisuke Kawashima, NadineSchneider, Dan Nealschneider, Andrew Dalke, tadhurst cdd, Matt Swain, Brian Cole, Samo Turk, Aleksandr Savelev, Alain Vaucher, Maciej Wójcikowski, Hussein Faara, Ichiru Take, Rachel Walker, Vincent F. Scalfani, Niels Maeder, Daniel Probst, Kazuya Ujihara, Axel Pahl, guillaume godin, and Juuso Lehtivarjo. rdkit/rdkit: 2025_09_1 (q3 2025) beta release, September 2025. URL https://doi.org/10.5281/zenodo.17193272. (Cited on page 16)

[47] Martin Steinegger and Johannes Söding. MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. *Nat. Biotechnol.*, 35(11):1026–1028, November 2017. (Cited on page 16)

[48] Shen Li, Yanli Zhao, Rohan Varma, Omkar Salpekar, Pieter Noordhuis, Teng Li, Adam Paszke, Jeff Smith, Brian Vaughan, Pritam Damania, and Soumith Chintala. Pytorch distributed: Experiences on accelerating data parallel training, 2020. URL https://arxiv.org/abs/2006.15704. (Cited on page 19)

[49] Ilya Loshchilov and Frank Hutter. Sgdr: Stochastic gradient descent with warm restarts, 2017. URL https://arxiv.org/abs/1608.03983. (Cited on pages 19 and 23)

[50] T. Konstantin Rusch, Michael M. Bronstein, and Siddhartha Mishra. A survey on oversmoothing in graph neural networks, 2023. URL https://arxiv.org/abs/2303.10993. (Cited on page 21)

[51] William Falcon and The PyTorch Lightning team. PyTorch Lightning, March 2019. URL https://github.com/Lightning-AI/lightning. (Cited on page 22)

[52] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning, 2017. URL https://arxiv.org/abs/1702.03118. (Cited on page 22)

[53] Keller Jordan, Yuchen Jin, Vlado Boza, You Jiacheng, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. Muon: An optimizer for hidden layers in neural networks, 2024. URL https://kellerjordan.github.io/posts/muon/. (Cited on page 23)

[54] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019. URL https://arxiv.org/abs/1711.05101. (Cited on page 23)

[55] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2017. URL https://arxiv.org/abs/1412.6980. (Cited on page 23)

Appendix

Contents

A Related Works

Binary Interaction Prediction. Virtual screening aims to efficiently identify potential drug candidates from vast chemical libraries that are likely to bind to a specific protein target. The dominant method for this task has traditionally been molecular docking[14–16], which predicts the binding energy and orientation of a molecule within a protein's binding site. However, docking is computationally intensive and time-consuming, making it impractical for screening the increasingly large chemical libraries that now contain billions of compounds. To address the speed limitations of docking, supervised learning methods[17] have been developed. These models are often trained as classifiers to discriminate between positive (binding) and negative (non-binding) protein-molecule pairs. Despite their speed, these approaches often suffer from poor generalization. More recently, a new paradigm has emerged that reframes virtual screening as a dense retrieval task[18]. Models like DrugCLIP[19] use contrastive learning to train separate encoders for proteins and molecules, learning a shared embedding space where binding pairs have high similarity. Its key advantage is the ability to pre-compute embeddings for entire molecule libraries, reducing the screening process to an ultra-fast similarity search and enabling efficient screening at the billion-compound scale.

Binding Affinity Prediction. Predicting binding strength—a regression task to determine a continuous affinity value (e.g., K_d, K_i)—is a long-standing challenge. Physics-based methods like free-energy perturbation (FEP) are highly accurate but too computationally expensive for large-scale screening. Conversely, faster methods like molecular docking often lack the required precision, creating a significant gap between accuracy and throughput. Deep learning models emerged to bridge this gap, with early methods learning features from structure-free inputs like SMILES and protein sequences using CNNs or GNNs[20, 21].

The field has since advanced toward geometric deep learning, which leverages 3D structural data for higher accuracy, often by representing the complex as an atomic graph[22–25]. However, these models face challenges with generalization to novel proteins and ensuring geometric invariance[26–28]. Recent methods like IPBind[29] address these issues by using machine-learning interatomic potentials to calculate the energy difference between bound and unbound states. Its SE(3)-invariant architecture correctly handles molecular chirality, leading to robust performance even with dissimilar proteins and imperfect, computationally predicted input structures.

Bridging the Accuracy-Speed Gap. Boltz-2[1] was a landmark effort to resolve this trade-off. It was presented as the first AI model to approach the accuracy of FEP simulations for small molecule-protein binding affinity while being over 1000 times more computationally efficient. By leveraging its powerful structural representations, Boltz-2 made high-fidelity affinity prediction feasible for discovery contexts. Despite this breakthrough, the inference time of Boltz-2 can still present a bottleneck for the routine screening of large chemical libraries. This limitation has spurred recent efforts to accelerate the Boltz architecture. However, these derivative methods, such as Boltzina[30], are not without their own drawbacks. For instance, Boltzina requires predefined binding pocket information as an input, restricting its applicability. Furthermore, by retaining the original trunk module of Boltz-2, it leaves significant room for further improvement in inference speed.

B Data Preprocess

We constructed a large-scale dataset to train our model, comprising both experimental binary labels and continuous affinity measurements (e.g., K_i, K_d, IC_{50}). Our training set integrates the following public datasets:

- **PubChem HTS**: PubChem[31] is a large, freely accessible chemical information database. After initial filtering, our binary label training set derived from its high-throughput screening (HTS) data contains 2,237,058 protein-ligand pairs, spanning 451 proteins and 368,812 unique ligands.
- **SAIR**: The SAIR[32] dataset is a large-scale synthetic structural dataset. Our continuous affinity value training set, selected from SAIR, comprises 796,557 pairs, covering 28,821 proteins and 403,124 unique ligands, with structures predicted by Boltz-1x[33].

For binary label validation, we utilized a curated subset of the **LIT-PCBA**[34] benchmark, a challenging virtual screening dataset derived from PubChem. Our validation set consists of 43,492 protein-ligand pairs, which were quantitatively sampled from each of LIT-PCBA's 15 targets. The sampling was designed to ensure that the hit rate for every target remained above 0.5%. For continuous affinity labels, we constructed our validation set by holding out a portion of the SAIR training data. This hold-out set was created by randomly selecting 20 different assays.

For our test set, we adopted publicly available datasets identical to those used in Boltz-2[1]. For binary labels, we used **MF-PCBA**[10] for testing. For continuous affinity values, we employed the **OpenFE subset**[7] of the FEP+ benchmark, a **4-target subset(CDK2, TYK2, JNK1, P38)**[8] of the FEP+ benchmark, and **CASP16**[9] for testing.

Following the data filtering, we performed structure preprocessing for entries lacking structural information and feature preprocessing for all data. Finally, to prevent data leakage, we filtered out data from the training set that is highly similar to the entries in the validation and test sets.

B.1 Data Filtering

For data filtering, our screening procedure is largely similar to the approach in Boltz-2, but with differences in some details. The specific filtering steps are as follows.

B.1.1 Binary Data Filtering

We applied the following filtering method to the PubChem HTS data to construct a reliable binary training set:

- We retained only assays that tested ≥ 100 compounds, had a hit rate (active/total) < 0.1, and targeted a single protein. This step ensures that the selected assays are valid HTS assays.
- We merged multiple assays for the same protein target and classified the compounds as `active`, `inactive`, or `conflict`, where `conflict` indicates contradictory labels across assays. This merging step maximizes chemical diversity by avoiding the high scaffold similarity within single assays. It also resolves common label conflicts between assays, which can otherwise be misinterpreted by the model as noise.
- For `active` and `conflict` protein-ligand pairs, we sought corresponding quantitative measurements from PubChem for secondary confirmation. We retained `active` compounds only if they were also reported as active in a quantitative experiment and resolved `conflict` compounds using the label from their quantitative measurement. The samples were discarded if their corresponding quantitative measurements were also conflicting. This secondary confirmation step helps to confirm the true labels for `conflict` compounds and filters out majority false positives from the HTS data.
- We remove PAINS[35] compounds and standardize the SMILES for all molecules. This step filters out this common class of false positive compounds.
- For each target, we subsampled the `inactive` molecules to an active-to-inactive ratio of 1:9. This step serves to mitigate data imbalance.

This filtering pipeline is deliberately designed to construct a large-scale, chemically diverse training set suitable for a hit-discovery task. By merging assays, we maximize scaffold diversity and resolve common label conflicts. The secondary confirmation against quantitative data and the removal of PAINS compounds further serve to minimize the impact of false positives inherent in HTS data.

However, we acknowledge the limitations of this approach. While effective for broadening chemical scope, merging assays inevitably obscures nuances arising from different experimental conditions. Consequently, this pipeline yields a dataset for an initial, relatively coarse hit-discovery model, rather than for precise affinity prediction. As noted in prior work like Boltz-2, a more refined strategy would involve leveraging assay metadata to assess data quality, a step that remains challenging due to inconsistencies in public database annotations.

B.1.2 Affinity Value Data Filtering

SAIR, the dataset we use for continuous affinity value labels, integrates data from ChEMBL and BindingDB. The dataset has been preprocessed to include only entries with a qualifier of "=" and removed some invalid entries. Furthermore, for each protein-ligand complex, SAIR provides five candidate structures predicted by Boltz-1x, with their structural fidelity evaluated by Posebusters.

Therefore, we apply the following filtering pipeline to the SAIR dataset, focusing on both label and structural quality, to construct a reliable dataset for continuous affinity modeling.

- For each entry in the SAIR dataset, we group entries by a unique combination of `protein` (UniProt ID[36]), `source` (ChEMBL or BindingDB), and `description` (the assay description). This ensures that all data points within a group originate from a single assay with a consistent experimental setup. Unlike binary labels which represent a general binding propensity, continuous affinity values (IC_{50}, K_i, K_d, etc.) are highly dependent on the specific experimental context. Therefore, we do not merge data from different assays, even for the same protein target.
- For each assay, we filter based on label quality. Our approach here is consistent with that of Boltz-2, as follows:
  - All affinity values are standardized to a common logarithmic scale relative to a 1 μM baseline.
  - Discard assays where the mean pairwise Tanimoto similarity (using ECFP4 Morgan fingerprints[37]) between compounds is below 0.25. Such assays lack a discernible chemical series, making them uninformative for learning structure-activity relationships.
  - Discard assays with fewer than 10 data points, as their small size makes them prone to spurious correlations.
  - Discard assays where the standard deviation of internal affinity values is below 0.25. A narrow activity range is uninformative for learning Structure-Activity Relationship.
  - Discard assays with fewer than 10 unique affinity values or a unique-to-total ratio below 0.2, as this indicates low measurement precision.
  - Discard assays containing extreme affinity values less than 10^{-6} μM, which likely result from data entry or unit inconsistencies.
- For data points within each qualified assay, we filter based on structural quality, as follows:
  - Discard data points where the mean ipTM of the corresponding predicted structures is ≤ 0.5. This indicates low confidence in the predicted binding interface.
  - Discard data points where the Posebusters[38] pass rate for the five predicted structures is ≤ 0.5 (i.e., fewer than three poses pass). This filters out entries where the majority of predicted poses are physically implausible and lack structural fidelity.

  The threshold of 0.5 for structural quality metrics was chosen as a deliberate trade-off between rigor and data retention. While a stricter cutoff would yield higher-quality structures, it would also discard a prohibitive amount of data. This threshold ensures that the retained structures are generally plausible in terms of both interface confidence and physical realism, while still maintaining a sufficiently large dataset for model training.

After applying this filtering pipeline to the SAIR dataset, we obtain a reliable final dataset. From this, we randomly sample all data points belonging to 20 assays to form the validation set, ensuring a consistent data distribution with the remaining training set.

B.2 Structure preprocessing

After filtering the data, we perform structure preprocessing on datasets that lack structural information (e.g., Pubchem, MF-PCBA). The process involves two main steps: first, determining the protein structure from its sequence, and second, determining the docking pose from the protein structure and the ligand's SMILES string. In virtual screening, the number of proteins is far smaller than the number of ligands, making ligand docking pose prediction the primary computational bottleneck.

B.2.1 Protein Structure preprocessing

We employ the following hierarchical strategy to obtain protein structures:

- First, we query the Protein Data Bank (PDB)[39] for an experimental structure. We search for `polymer_entity` entries with 100% sequence identity (`identity_cutoff=1.0`) and a resolution of 3.5Å or better, prioritizing higher resolutions. The results are categorized as either a full sequence match or a subsequence match (i.e., the query is a single chain within a larger complex). We prioritize full matches. Each candidate is then validated: we ensure all backbone atoms (N, C_α, C, O) are present for every residue. Furthermore, we verify that the Levenshtein distance between our query sequence and the actual sequence extracted from the PDB file is no greater than 5. This threshold is chosen to accommodate minor discrepancies inherent to experimental structures, such as terminal missing residues or point mutations, which are unlikely to significantly alter the overall protein fold or binding pocket. Structures that pass these checks are renumbered and saved.
- If no suitable structure is found in the PDB, we search the AlphaFold DB[40]. The corresponding UniProt ID is identified from the sequence, and we download the predicted structure with the highest mean pLDDT score. The mean pLDDT serves as a global indicator of the prediction's overall quality, allowing us to select the most reliable model. As these are computational models, they are guaranteed to be complete and match the input sequence perfectly, requiring no further validation.
- If a structure is available in neither the PDB nor the AlphaFold DB, we predict 10 structures using Boltz-2x[1] and select the one with the highest pLDDT score.

This procedure allows us to quickly and accurately source high-quality structures. In practice, we found that over 60% of protein structures were obtained from the PDB and AlphaFold DB, significantly reducing the computational cost of de novo prediction. Our strict criteria ensure high fidelity between the final structure and the query sequence.

B.2.2 Ligand Pose preprocessing

Once the protein structure is prepared, we employ the **Regression FABind+**[41] model to predict the ligand's docking pose. We selected FABind+ over traditional docking software primarily for its inference speed, a critical factor for processing our large-scale dataset. The model exhibits strong predictive accuracy, achieving a success rate where 43.5% of predicted poses have a ligand RMSD below 2Å on the PDBbind[42] benchmark, a result that surpasses even powerful generative models like DiffDock[43].

A key architectural feature of FABind+ is its decomposition of the blind docking task into two stages: pocket prediction followed by pose generation. We leverage this by using the pocket coordinates predicted by the model to define a localized structural region for our downstream affinity prediction model. In terms of efficiency, the official reported runtime for the regression model is 0.16 seconds per complex. In our production pipeline on a single NVIDIA L40S GPU, this translated to an end-to-end processing time of about 0.7 seconds per pair, including all data I/O and featurization steps.

Finally, we acknowledge the limitations of using the regression mode. As noted by the authors of FABind+, a regression-based approach cannot capture multiple potential binding sites or a diverse set of ligand conformations in the way a sampling-based method can. However, for our task of generating a single, high-quality structural complex for affinity prediction, the regression model provides an optimal balance of accuracy and high-throughput efficiency.

B.3 Feature preprocessing

To construct the node representations for our EGNN[2] model, we preprocess proteins and ligands to generate their respective initial features.

Protein Feature We generate protein base embeddings using the **ESM-3**[44] (`esm3_sm_open_v1`) language model. From this model, we extract the final hidden layer's representation for each residue. These residue-level embeddings are then broadcasted to every atom within their respective residue. These vectors constitute the initial sequence-based features for the protein atoms and are subsequently

concatenated with other structural features (e.g., atom type, residue type) within the model itself to form the final EGNN node representations. The amortized computational cost of this step is manageable, given the relatively small number of unique protein targets in a typical virtual screening task.

Ligand Feature For ligand features, we adopt a featurization pipeline inspired by FABind+, generating features using the `property_prediction` utilities of **torchdrug**[45], which internally leverages **RDKit**[46]. This process extracts a feature vector for each atom composed of the following standard chemical properties:

- One-hot encodings for the atom symbol (e.g., C, N, O).
- One-hot encodings for the atom's degree (number of heavy-atom neighbors).
- One-hot encodings for the total number of attached hydrogens.
- One-hot encodings for the atom's total valence.
- One-hot encodings for the atom's formal charge.
- A boolean flag indicating if the atom is part of an aromatic system.

This featurization pipeline is highly efficient, requires no GPU, and is easily parallelized across CPUs. In our implementation, we can generate features for over 800 ligands per second on a single CPU core, enabling on-the-fly feature generation within our data loader.

Node Feature To create a unified representation, the initial protein and ligand feature sets, which may have different dimensions, are first projected into a common hidden dimension. These projected vectors are then concatenated to form a base embedding. Finally, to construct the complete node feature vector $\mathbf{h}_i$ for each node i, this base embedding is further augmented by concatenating it with several categorical and positional features: one-hot encodings for the atom type and residue type, a binary indicator for the molecule type (protein or ligand), and a positional encoding for the residue's sequence index. This multi-faceted feature design ensures that each node is aware not only of its local chemical environment and pre-trained semantics but also of its identity and position within the larger biomolecular complex.

B.4 Avoiding Data Leakage

To ensure a rigorous evaluation of our model's generalization capabilities, we implemented stringent protocols to prevent data leakage between the training and test sets at both the protein and ligand levels.

Protein-level Filtering We aimed to evaluate our model on protein families not encountered during training, a protocol also employed by methods like Boltz-2 to ensure robust evaluation . To this end, we removed any protein from the training set sharing $\geq$90% sequence identity with any protein in our validation or test sets. This was carried out using **MMseqs2**[47] to cluster the complete protein set with the command: `mmseqs easy-cluster -min-seq-id 0.9 -cov-mode 0 -c 0.01`. Subsequently, all training proteins belonging to a cluster that contained a validation or test protein were discarded.

Ligand-level Filtering To prevent the model from exploiting learned chemical scaffolds, we filtered the training set based on Tanimoto similarity to the test set. Any ligand in the training set with a Tanimoto similarity of $\geq$0.4 to any **active** ligand in the **test set** was removed. This threshold is commonly used to distinguish between different chemical scaffolds. Our decision to filter only against active compounds was twofold: first, it directly addresses the most critical leakage scenario where a model could achieve artificially high performance by recognizing analogs of known actives. Second, given the vast number of inactive compounds, filtering against the entire test set would have excessively reduced the size and diversity of our training data. This targeted approach is therefore designed to prevent the most direct and meaningful forms of compound leakage.

C Supplementary Details about the Our Method

C.1 Pocket Cropping

To create a computationally tractable and interaction-focused representation of the protein-ligand complex, we apply a deterministic cropping algorithm that generates a localized subgraph around the ligand. The algorithm selects a subset of protein residues based on three simultaneous criteria: spatial proximity to the ligand, a total atom budget (B_a), and a total residue budget (B_r). For budgeting, each atom in the ligand is treated as a single-atom residue, so the effective budgets for protein residue selection are pre-calculated by subtracting the number of ligand atoms from both B_a and B_r. This process can be initialized in two modes: operating on the entire protein structure to identify a pocket *de novo*, or refining a smaller, predefined set of candidate pocket residues provided as a prior.

The selection process, detailed in Algorithm S1, begins by calculating the distance from each input protein residue to the ligand. This distance is defined as the minimum Euclidean distance between any atom in the residue and any atom in the ligand. All protein residues are then sorted based on this distance in ascending order. A greedy selection is performed on this ranked list, iteratively adding the closest residues until either the atom or residue budget is exhausted.

Finally, this budget-constrained set of residues is filtered to retain only those within a specified distance threshold, d_{max}. To ensure the resulting pocket is not overly sparse, a fallback mechanism is triggered if this final set contains fewer than a minimum number of residues, k_{min}. In such cases, the algorithm instead returns the top k_{min} closest residues from the budget-constrained set, overriding the distance threshold. For all experiments, we used the following parameters: $B_a = 2048$, $B_r = 512$, $d_{\text{max}} = 20.0$ Å, and $k_{\text{min}} = 100$.

C.2 Edge Construction

Inspired by MEAN[3], we provide a detailed specification for the construction rules of each edge category defined in the main text. All generated edges are directed and added bidirectionally to the graph representation unless otherwise noted. The parameters used in our experiments are summarized at the end of this section.

Internal Edges ($\mathcal{E}_{\text{internal}}$) These edges capture the covalent topology and local connectivity within the protein and the ligand separately.

- **Ligand Covalent Bonds:** Edges are derived directly from the ligand's molecular graph. We define four distinct relation types to represent single, double, triple, and aromatic bonds, respectively. This explicitly encodes the chemical structure of the ligand.
- **Protein Covalent (Intra-Residue) Edges:** Covalent bonds connecting atoms within each amino acid residue are added based on predefined, chemically-valid bonding templates for each of the 20 standard amino acid types.
- **Protein Sequential Edges:** To represent the polypeptide backbone, edges are created between the α-carbon (Cα) atoms of sequentially adjacent residues (i.e., from residue k to $k+1$).

External Edges ($\mathcal{E}_{\text{external}}$) This category contains a single but crucial type of edge that models the non-covalent interactions at the protein-ligand interface.

- **Protein-Ligand Proximity Edges:** An edge is created between a protein atom $v_i \in \mathcal{V}_{\text{P}}$ and a ligand atom $v_j \in \mathcal{V}_{\text{L}}$ if their Euclidean distance $\|\mathbf{x}_i - \mathbf{x}_j\|$ is less than a predefined cutoff, $d_{\text{cross}} = 10.0$ Å. These edges are fundamental for modeling binding interactions.

Auxiliary Edges ($\mathcal{E}_{\text{aux}}$) This class provides multi-scale contextual information, from local atomic neighborhoods to the global structure of the complex.

- **Global Edges:** We first define two global nodes, v_{P} and v_{L}, whose initial coordinates and features are the mean of their constituent atoms. Edges are then established: (1) from v_{P} to every protein atom $v_i \in \mathcal{V}_{\text{P}}$; (2) from v_{L} to every ligand atom $v_j \in \mathcal{V}_{\text{L}}$; and (3) a single edge connecting v_{P} and v_{L}.

Algorithm S1 Pocket Cropping Algorithm

Input: Protein residues $\mathcal{R}_P$; Ligand atoms $\mathcal{A}_L$; Budgets B_a, B_r; Thresholds $d_{\text{max}}, k_{\text{min}}$.

 # Calculate effective budgets by accounting for the ligand
$B'_a = B_a - |\mathcal{A}_L|$
$B'_r = B_r - |\mathcal{A}_L|$ *# Each ligand atom consumes one residue slot*
 # Pair each residue with its distance to the ligand and sort
residue_distances = []
for r **in** $\mathcal{R}_P$ **do**
  dist = $\min_{\mathbf{x}_i \in r, \mathbf{x}_j \in \mathcal{A}_L} \|\mathbf{x}_i - \mathbf{x}_j\|_2$
  Add (r, dist) to residue_distances
end for
sorted_residues = Sort(residue_distances, key=distance)
 # Greedily select residues satisfying budgets, and track those also within distance threshold
$\mathcal{P}_{\text{budget}}$ = [] *# Stores residues respecting budgets only (for fallback)*
$\mathcal{P}_{\text{dist}}$ = [] *# Stores residues respecting budgets AND distance threshold*
atom_count, residue_count = 0, 0
for (r, dist) **in** sorted_residues **do**
  if (atom_count + r.num_atoms $\leq B'_a$) **and** (residue_count + 1 $\leq B'_r$) **then**
    Add (r, dist) to $\mathcal{P}_{\text{budget}}$
    if dist $\leq d_{\text{max}}$ **then**
      Add (r, dist) to $\mathcal{P}_{\text{dist}}$
    end if
    atom_count = atom_count + r.num_atoms
    residue_count = residue_count + 1
  else
    break
  end if
end for
 # Apply robustness fallback if the distance-filtered set is too small
if len($\mathcal{P}_{\text{dist}}$) $< k_{\text{min}}$ **then**
  # Fallback: take the k_{min} closest residues that fit the budget
  result_pairs = $\mathcal{P}_{\text{budget}}$[:min($k_{\text{min}}$, len($\mathcal{P}_{\text{budget}}$))]
else
  result_pairs = $\mathcal{P}_{\text{dist}}$
end if
 # Extract only the residue objects for the final output
$\mathcal{P}$ = [r **for** (r, dist) **in** result_pairs]
Output: A selected subset of protein residues for the pocket, $\mathcal{P}$.

- **Protein Spatial Edges:** To model long-range interactions within the protein pocket, an edge connects two protein atoms if they belong to *different* residues and their spatial distance is below a cutoff, $d_{\text{protein}} = 4.0$ Å. This explicitly avoids duplicating the intra-residue covalent bonds.
- **Ligand LAS (Local Atomic Structure) Edges:** To enrich the ligand's local conformational information, we add "virtual" edges between pairs of ligand atoms that satisfy either of two conditions: (1) they are within a $k_{\text{LAS}} = 2$-hop distance in the ligand's covalent graph, or (2) they are members of the same minimal ring structure (identified via SSSR algorithm). This strategy is inspired by a similar application in FABind+[41].

Graph Finalization and Truncation After generating all edges according to the rules above, a finalization procedure is applied. First, all self-loops (i.e., edges from a node to itself) are removed. Second, duplicate edges are removed by retaining only the first occurrence of each unique directed pair (v_i, v_j). This step is crucial as it respects the priority ordering of the edge creation process. Finally, if the total number of edges exceeds the maximum budget $B_e = 16384$, the edge list is truncated to size B_e. Given the construction order, this ensures that higher-priority edges are always preserved. The priority scheme, from highest to lowest, is as follows:

The priority scheme, from highest to lowest, is: External (Protein-Ligand) > Auxiliary (Global) > Internal (Ligand Covalent Bonds) > Internal (Protein Sequential) > Internal (Protein Covalent) > Auxiliary (Ligand LAS) > Auxiliary (Protein Spatial) Edges.

This prioritization scheme is motivated by a hierarchy of informational importance for the affinity prediction task. The highest priority is granted to edges that explicitly model the protein-ligand interaction itself (External and Global). Subsequently, the ligand's internal chemical structure is prioritized over the protein's, reflecting its central role in the binding event. Finally, the auxiliary edges that provide additional, broader spatial context are assigned the lowest priority.

C.3 Sampling

To optimize the training signal for both affinity prediction and binary classification, we employ a group-based mini-batch sampling strategy inspired by Boltz-2[1]. This approach supports training on a mixture of multiple datasets and is designed for use with Distributed Data Parallel (DDP)[48], ensuring that each parallel process (rank) receives a unique, non-overlapping stream of data. The sampler constructs each training batch from multiple small, independent groups of samples, where all samples within a single group are drawn from the same biochemical assay (i.e., they share the same protein target). This ensures a consistent experimental context for stable gradient estimation. A training batch of size B is partitioned into $K = B/N$ groups of size N; our experiments used $B = 20$ and $N = 5$.

Prior to training, a one-time filtering step is applied to the assays of each dataset. For binary datasets, we retain only those assays that contain at least one known binder and one decoy. For affinity datasets, we retain only assays with at least two measured ligands. This ensures that all subsequent sampling operations occur on valid and informative assays. When multiple datasets are provided, the sampler first selects a dataset for the entire batch based on a predefined categorical distribution.

The procedure for populating the groups is then chosen based on the selected dataset's type. For a **binary dataset**, we first sample a valid assay uniformly at random. From this assay, we then form a group using a fixed 1:4 ratio of binders to decoys. For an **affinity value dataset**, we first assign a sampling weight to each valid assay proportional to the interquartile range (IQR) of its affinity values. An assay is then selected according to these IQR-based weights, and a group is formed by uniformly sampling N ligands from it.

For all sampling steps, we sample without replacement by default. However, if the number of available ligands in a pool is less than the number requested, the sampler automatically switches to sampling with replacement to ensure each group is fully populated. This entire process is driven by a seeded random number generator to guarantee reproducibility while maintaining stochasticity across epochs and ranks. The detailed logic is formalized in Algorithm S2.

C.4 Training Objectives

This section details the task-specific formulations of the group-level loss function, $\mathcal{L}_{\text{group}}$. Our overall training objective strategy is consistent with that of Boltz-2, employing task-specific losses for groups sampled from binary or affinity datasets.

Binary Classification Loss For binary classification groups, the group loss is the mean Focal Loss[4] computed between the true labels y and the predicted logits $\hat{y}$, i.e., $\mathcal{L}_{\text{Focal}}(y, \hat{y})$, for all samples in the group. This loss is effective for handling class imbalance. We use a focusing parameter $\gamma = 1$ and a positive class weighting factor $\alpha = 0.7$.

Affinity Regression Loss For affinity regression groups, the group loss is a weighted combination of a pairwise difference loss ($\mathcal{L}_{\text{dif}}$) and an absolute value loss ($\mathcal{L}_{\text{abs}}$):

$$\mathcal{L}_{\text{group}}^{\text{affinity}} = 0.9 \cdot \mathcal{L}_{\text{dif}} + 0.1 \cdot \mathcal{L}_{\text{abs}} \tag{1}$$

Here, $\mathcal{L}_{\text{dif}}$ is the Huber loss applied to the pairwise differences between true (y_i, y_j) and predicted $(\hat{y}_i, \hat{y}_j)$ affinities, i.e., $\mathcal{L}_{\text{Huber}}((y_i - y_j), (\hat{y}_i - \hat{y}_j))$, for all pairs within the group. $\mathcal{L}_{\text{abs}}$ is the Huber loss[5] on the absolute values for each sample, i.e., $\mathcal{L}_{\text{Huber}}(y_i, \hat{y}_i)$. For both components, the Huber loss delta parameter is set to $\delta = 0.5$.

C.5 Snapshot Ensembling

Our use of a cosine annealing learning rate scheduler with warm restarts[49] allows the model to converge to several distinct local minima over the course of a single training run. This enables the use

Algorithm S2 Sampling Algorithm

Input: A collection of datasets $\mathcal{D}$; Batch size B; Group size N; Binder-to-decoy ratio for binary tasks α; Dataset sampling probabilities $\boldsymbol{\pi}$.

 # Pre-computation: Filter assays and build pools for each dataset
ValidAssayPools = {}
AssayWeights = {}
for dataset D_i **in** $\mathcal{D}$ **do**
  ValidAssays = []
  for assay a **in** D_i **do**
    if D_i is binary **and** a has ≥ 1 binder **and** ≥ 1 decoy **then**
      Add a to ValidAssays
    else if D_i is affinity **and** a has ≥ 2 ligands **then**
      Add a to ValidAssays
      AssayWeights[a] = InterQuartileRange(affinity values of a)
    end if
  end for
  ValidAssayPools[D_i] = ValidAssays
end for
 # Batch Generations Loop
num_groups_per_batch = $\frac{B}{N}$
loop
  batch_indices = []
  # Select a dataset for the entire batch
  selected_dataset = WeightedSample($\mathcal{D}$, $\boldsymbol{\pi}$)
  for $k = 1$ **to** num_groups_per_batch **do**
    if selected_dataset is binary **then**
      selected_assay = UniformSample(ValidAssayPools[selected_dataset])
      binders = UniformSample(all binders of selected_assay, count=1) *# with replacement if needed*
      decoys = UniformSample(all decoys of selected_assay, count=4) *# with replacement if needed*
      group_indices = binders + decoys
    else*# selected_dataset is affinity value*
      weights = [AssayWeights[a] for a in ValidAssayPools[selected_dataset]]
      selected_assay = WeightedSample(ValidAssayPools[selected_dataset], weights=weights)
      group_indices = UniformSample(ligands of selected_assay, count=N) *# with replacement if needed*
    end if
    Add group_indices to batch_indices
  end for
  yield Flatten(batch_indices)
end loop

of snapshot ensembling[6], where we aggregate predictions from multiple high-performing model checkpoints to enhance predictive accuracy and robustness. The procedure involves two stages: selecting the ensemble of checkpoints and then combining their predictions.

Checkpoint Selection During the training process, we save model checkpoints at predefined epoch intervals. After the full training process is complete, we evaluate the performance of every saved checkpoint on a held-out validation set. From this pool of candidates, we greedily select the final ensemble of $k = 2$ models. The process starts by selecting the single best-performing checkpoint. Subsequently, we iteratively add the next-best checkpoint that is at least $T = 5000$ training steps separated from all previously selected models. This minimum separation constraint ensures a degree of diversity within the ensemble, preventing the selection of overly correlated models from the same local performance basin. The ensemble size k and the minimum step separation T are hyperparameters determined empirically on the validation set.

Prediction Aggregation Once the $k = 2$ snapshot models are selected, their predictions are combined in a task-specific manner, following the strategy proposed by Boltz-2.

- For the Binary Model, the final prediction is a simple average of the probabilities from each model in the ensemble. If p_m is the predicted probability from model m for a given sample,

the ensembled probability p_{ensemble} is calculated as:

$$p_{\text{ensemble}} = \frac{1}{k} \sum_{m=1}^{k} p_m \tag{2}$$

- For the Affinity Model, we employ a calibrated ensembling strategy to correct for systematic prediction biases. First, the raw mean prediction, $\bar{y}$, is calculated across the k models. This average is then refined using a polynomial correction function that includes a bias term related to the ligand's molecular weight (MW):

$$\hat{y}_{\text{ensemble}} = C_0 \cdot \bar{y} + C_1 \cdot \text{MW}^{0.3} + C_2 \tag{3}$$

  The coefficients $C_0 = 2.177929, C_1 = -0.218761$, and $C_2 = 1.472271$ are calibrated by fitting this function to the ensemble's predictions on the validation set. This allows the model to learn a final correction for systematic errors, such as those correlated with ligand size. These fitted coefficients are then used for inference on the test set.

C.6 Limitations and Future Work

Our work successfully demonstrates a lightweight and efficient alternative for virtual screening, but we acknowledge several limitations that offer clear directions for future research.

Architectural Constraints Our model intentionally employs a standard EGNN[2] architecture with a modest parameter count (about 4.5M). While this is the key to its efficiency, it also presents a performance bottleneck. We observed that the model's limited capacity made it challenging to perform joint training on structural refinement and affinity prediction, a strategy successfully employed by larger models like Boltz-2[1]. Furthermore, while GNNs are powerful, traditional architectures can face challenges such as over-smoothing[50] when scaled to the depth of very large models, potentially limiting their performance ceiling compared to the well-established scaling properties of transformer-based architectures like the PairFormer used in Boltz-2's affinity module.

Future work will involve exploring more scalable graph-based architectures, possibly by integrating attention mechanisms or other advanced layers to increase model capacity while maintaining a favorable efficiency profile.

Dependence on External Structure Prediction Our pipeline operates as a fast scoring function, meaning its performance is inherently dependent on the quality of the input 3D structures generated by external tools like FABind+[41]. This reliance creates an upper bound on our model's accuracy—an incorrect or poorly docked binding pose cannot be easily rescued by the scoring function. While structure prediction is becoming increasingly reliable, this dependency does constrain our model's performance and positions our work as an improvement in scoring rather than a fully end-to-end solution. In the future, we plan to investigate methods to mitigate this dependency, such as incorporating lightweight, local structure refinement capabilities directly into our model.

Data Dependency and Generalization Finally, like all data-driven methods, the performance of our model is highly contingent on the quality and diversity of the training data. Despite our rigorous curation pipeline, public biochemical datasets are known to contain significant experimental noise and biases. Our model's ability to generalize to novel protein families or regions of chemical space is therefore directly linked to the breadth and quality of the data it was trained on. A continuous and vital area for future work will be the exploration of more diverse data sources and the development of more sophisticated data filtering and cleaning methodologies to further enhance model robustness and generalization.

D Details about the Adopted Metrics

Our evaluation protocol follows that of Boltz-2. We assess model performance on two tasks: binary binder classification and continuous affinity value prediction.

D.1 Binary Label Prediction Metrics

For the classification task, all metrics are computed per-assay and then uniformly averaged across assays.

- **Average Precision (AP)**: The area under the Precision-Recall curve, which is particularly informative for imbalanced datasets.
- **Enrichment Factor (EF)**: The ratio of active compounds found in the top $k\%$ of the ranked list compared to a random distribution. We report EF at k = 0.5, 1, 2, and 5 to evaluate early retrieval performance.
- **Area Under the Receiver Operating Characteristic Curve (AUROC)**: Representing the probability that a randomly chosen binder is ranked higher than a random non-binder.
- **Global AUROC**: Calculated by treating all protein-ligand pairs across the entire dataset as a single, unified ranking task, rather than averaging per-assay results.

D.2 Affinity Value Prediction Metrics

For the regression task, metrics are computed per-assay, with final scores aggregated via a weighted average proportional to the number of compounds in each assay. All affinity predictions are converted to kcal/mol before evaluation.

- **Pearson's R**: Measures the linear correlation between predicted and true binding affinities.
- **Kendall's τ**: Assesses the rank correlation (i.e., monotonic agreement) between predicted and true affinities within each assay.
- **Pairwise Mean Absolute Error (PMAE)**: The MAE applied to the predicted versus true affinity differences between all pairs of compounds within an assay.
- **Mean Absolute Error (MAE)**: The average absolute difference between predicted and true affinity values.
- **Percentage within (PW) 1 and 2 kcal/mol**: The percentage of predictions with an MAE less than X kcal/mol, reported for $X = 1$ and $X = 2$.

Note on Centering: For MAE, PMAE and PW, we also report a centered version. In this calculation, predictions for each assay are first shifted to match the mean of the ground truth values. This adjustment isolates the model's performance on relative affinities, enabling fairer comparison to methods like relative FEP.

E Details about the Implementations

This section provides supplementary details regarding our model architecture and training pipeline.

E.1 Training Setup

We trained two independent models for the binary classification and affinity value prediction tasks, respectively. Each model was trained using PyTorch Lightning[51] on a cluster of 4 NVIDIA L40S GPUs with 32-bit precision. To ensure training stability, we applied gradient clipping with a maximum norm of 2.0. Training was conducted using a group-based mini-batch strategy, with each batch containing 20 protein-ligand complexes, partitioned into 4 groups of 5 samples.

E.2 Model Architecture

Our model is built upon an E(3)-Equivariant Graph Neural Network (EGNN)[2]. The key architectural parameters are as follows:

- **EGNN Core:** The network consists of 5 EGNN layers with a hidden dimension of 192. A dropout rate of 0.1 is applied within the network to prevent overfitting.
- **Prediction Head:** The final graph representation is passed to a 3-layer MLP prediction head, which utilizes SiLU[52] activation functions and has a hidden dimension of 192, to produce the final scalar output.

E.3 Optimization Strategy

The two models were trained with different optimization strategies to maximize performance on their respective tasks.

Optimizer Selection For the binary classification task, we employed a hybrid optimization strategy by mixing two different optimizers. Specifically, for the parameters within the EGNN hidden layers, we used the **Muon**[53] optimizer. This allowed us to apply a high learning rate of 2×10^{-3} and a weight decay[54] of 0.01 to accelerate the convergence of the model's core representation learning component. All other model parameters (e.g., embedding layers, prediction head) were optimized using a standard **AdamW**[55, 54] optimizer with more conservative hyperparameters ($\beta_1 = 0.9$, $\beta_2 = 0.95$, and a weight decay of 1×10^{-3}).

However, for the more sensitive affinity value regression task, this aggressive, hybrid strategy led to training instability. We therefore opted for a simpler and more stable approach, using only the standard **AdamW** optimizer for all model parameters.

Learning Rate Scheduler For both training runs, we employed a **cosine annealing learning rate scheduler with warm restarts** (`cosine_restart`)[49]. This scheduler is essential for our snapshot ensembling strategy described in Appendix C.5. The learning rate for the AdamW optimizer was warmed up over 100 steps to a maximum value of 1×10^{-4}, followed by cosine decay cycles with a restart period (`t_0`) of 80,000 steps.

F Ablation Study on Key Architectural Choices

To rigorously validate the two core design principles of FlashAffinity, we conducted an ablation study to decouple the effects of our simplified structure generation and scoring modules. Our central hypotheses are: (1) a fast protein-ligand docking model can effectively replace the computationally expensive structure prediction modules of large foundation models like Boltz-2[1] without a significant loss in performance, and (2) a lightweight EGNN-based[2] scoring architecture is sufficient to achieve effective performance for virtual screening tasks.

Due to the significant computational cost and time required to run the full Boltz-2 pipeline, conducting this study on the entire MF-PCBA[16] test set was infeasible. Therefore, we created a representative subset containing 50,000 protein-ligand pairs, sampled proportionally from each target based on its hit rate. This provides a balanced and computationally tractable benchmark for direct comparison. We evaluated three models on this subset:

- **Boltz-2**: The original state-of-the-art model, using its end-to-end structure prediction and affinity scoring modules.
- **Boltz-2(FABind+)**: An ablated version where the input to Boltz-2's affinity scoring module is replaced with protein-ligand complex structures generated by the fast docking model, FABind+[41].
- **FlashAffinity**: Our proposed model, which uses FABind+ for structure generation and our lightweight EGNN for scoring.

The results of our ablation study are summarized in Table S1. It provides strong evidence supporting our design choices. First, by comparing Boltz-2 and Boltz-2(FABind+), we observe that replacing

Method	AP ↑	AUROC ↑		EF at 0.5% ↑	EF at 1% ↑	EF at 2% ↑	EF at 5% ↑
		target avg.	global				
Boltz-2	0.0233	0.7994	0.7788	15.3087	10.4519	8.2599	7.1524
Boltz-2(FABind+)	0.0244	0.8054	0.7852	16.1420	11.1186	7.7377	6.4675
FlashAffinity	0.0247	0.7414	0.7778	14.9905	12.8531	7.9154	6.7143

Table S1: Ablation study results on a 50,000-pair subset of the MF-PCBA benchmark. The study compares the original Boltz-2, Boltz-2 using pre-computed docking poses from FABind+, and our FlashAffinity model.

the expensive structure prediction module with FABind+ docking poses does not harm performance. In fact, the Boltz-2(FABind+) variant shows slightly improved AP and AUROC, demonstrating that high-quality docked structures are sufficient for the downstream scoring task. This validates our first hypothesis and justifies the use of fast docking to bridge the speed gap.

Second, the comparison between FlashAffinity and Boltz-2 (using FABind+ structures) isolates the performance of the scoring architectures, as both use identical inputs. In this direct comparison, FlashAffinity achieves a higher Enrichment Factor at the 1% threshold (EF@1%) and a marginally improved Average Precision (AP). While its overall ranking ability as measured by AUROC is lower, its effectiveness in these early enrichment metrics—which are critical for practical hit discovery—demonstrates that our lightweight EGNN architecture is a valid and useful approach for prioritizing top-ranking compounds. This result lends support to our second hypothesis regarding the utility of a simpler scoring architecture.

In conclusion, this ablation study effectively demonstrates that the combination of a fast docking model for structure generation and a computationally efficient EGNN for scoring is a powerful and valid strategy. It allows FlashAffinity to achieve effective performance in binder classification while offering a substantial speedup, thus successfully addressing the accuracy-efficiency trade-off in large-scale virtual screening.