LEARNING A PRECISE LANGUAGE FOR SMALL-MOLECULE BINDING

Anonymous authors
Paper under double-blind review

ABSTRACT

Querying a given protein target against a large set of candidate small-molecule binders is a crucial first step in the drug-discovery pipeline. With billion-compound libraries now available, accurate and fast virtual screening is critical for identifying potential drugs. Sequence-based drug-target interaction (DTI) prediction methods are fast and highly scalable, but leave important challenges in localizing binding sites, handling targets beyond single-chain proteins, and selecting candidates for expensive downstream assessment. We introduce PRECISE, which learns a structure-based protein representation and its compatibility with a quantized codebook of small-molecule space. Specifically, we build on the molecular codebook introduced by CoNCISE and formulate DTI prediction as querying quantized drug representations against a geometric neural network-based representation of the protein surface enriched with electrostatic and geometric features. PRECISE addresses key limitations of previous approaches while remaining highly accurate and scalable. It also generalizes automatically to protein complexes and metal-binding proteins without specialized training. High-throughput virtual screening often fails to align well with downstream workflows. To fix this, we introduce PRECISE-MCTS, a tree search algorithm that uses feedback from a downstream scoring oracle to refine PRECISE hits under a limited time or evaluation budget. Across Vina, Boltz, and Protenix, PRECISE-MCTS acts as a force multiplier for downstream methods. Under matched evaluation budgets, it improves top-10 mean oracle scores in 29 of 30 target–oracle comparisons.

1 INTRODUCTION

Ultra-fast drug-target interaction (DTI) prediction methods that co-embed protein language model representations of targets with compact representations of small molecules have opened a regime of ultra-high-throughput virtual screening (uHTVS) (Singh et al., 2023; Gao et al., 2023; Guichaoua et al., 2024; Erden et al., 2025), generating early-stage leads from billion-compound libraries such as Enamine (Grygorenko, 2021). Moving from these large hit sets to actionable candidates, however, requires more than predicting whether a compound binds: it requires identifying promising binding sites and deciding which candidates to prioritize for expensive downstream assessment. Many sequence-based uHTVS methods also remain limited to single-chain protein targets, leaving protein complexes and metalloproteins beyond their scope.

We introduce PRECISE, which learns to query a molecular codebook against local representations of a protein's 3D surface. A key advance in PRECISE is to transfer molecular codes learned through sequence-based DTI prediction to structurally-resolved binding prediction, without changing the codes themselves. Here we leverage the frozen codebook learned by a previous method, CoNCISE. Given an experimentally-determined or computationally-predicted target structure, PRECISE learns, for each surface location, which entries in the CoNCISE codebook correspond to candidate binders. Remarkably, PRECISE outperforms CoNCISE and many other methods on whole-protein DTI prediction, while also suggesting not only *whether* a drug might bind but *which pocket* it might bind to. PRECISE accomplishes this while still operating at speeds capable of screening billion-scale compound libraries such as Enamine. In addition, we demonstrate that PRECISE generalizes to multi-chain protein complex targets, since its representation is surface-based and agnostic to whether the surface comes from a single-chain protein or a multi-chain protein complex. Methods restricted to a single protein sequence cannot directly represent these multimeric surfaces.

Continuing down the funnel of refining and prioritizing hits, we must reduce the 10^3–10^6 hits a uHTVS method may report to a downstream-usable set that meets pocket-specificity, off-target

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

avoidance, and other prioritization criteria. A typical workflow applies slower structure-based assessment, using docking (Trott & Olson, 2010; Eberhardt et al., 2021; McNutt et al., 2021; Yu et al., 2023) or ML-based co-folding (Abramson et al., 2024; Corso et al., 2022; team et al., 2024; Passaro et al., 2025) methods as second-pass filters, to narrow 10^3–10^6 hits to a few hundred before experimentally validating a few dozen. Effective uHTVS therefore requires methods whose outputs are not only fast to compute, but also well aligned with a broad range of downstream assessments, from docking scores and co-folding confidence to predicted IC_{50}. Even with such alignment, a hit set of thousands, tens of thousands, or millions of candidates is too large to assess exhaustively. We therefore also need a way to hone in on the most promising candidates under the chosen assessment with only a small number of expensive evaluations.

We introduce PRECISE-MCTS to address this budget-constrained search problem. It uses PRECISE to retrieve a promising candidate pool, then uses feedback from a slower scoring oracle to guide further selection within that pool. The search is *method agnostic*: any such downstream scoring method can serve as its oracle. In the present paper, we benchmark PRECISE-MCTS using Vina (Trott & Olson, 2010), Boltz (Passaro et al., 2025) and Protenix (Zhang et al., 2026). Across these different assessments, PRECISE retrieves promising candidates, and PRECISE-MCTS refines them further under matched evaluation budgets. While there is spirited debate about which docking or co-folding method performs best, *the beauty of* PRECISE-MCTS *is that it acts as a force multiplier for each chosen method, helping it find candidates that score better by its own criteria within the same evaluation budget.*

2 METHODS

2.1 PROBLEM SETUP

PRECISE scores the compatibility between a small molecule and a local region of the protein surface. The model takes two inputs: (i) a protein structure represented as a surface mesh and (ii) a compound specified by a SMILES string, and returns binding scores at each surface vertex. We optionally also predict an overall binding score against the whole protein. We define three spatial scales: a **vertex** is a point on the triangulated surface mesh, characterized by four physicochemical features; a **patch** is the surface region within 7 Å of a vertex; and a **pocket** is a recessed region of the protein surface that can serve as a ligand-binding site. After message passing, each patch summarizes a receptive field of approximately 10 Å. This radius is standard in existing literature (Gainza et al., 2020). A pocket may span multiple patches; a formal geometric definition is provided in Definition 2.

We train two prediction heads on the surface and ligand representations. The **whole-protein head** predicts whether a compound binds to the protein, while the **patch-level head** predicts its binding compatibility with a specified surface patch. The latter supports pocket-directed screening by restricting the query to a site of interest. We describe both heads in Section 2.4.

For downstream refinement, we consider a budget-constrained virtual screening problem. Given a target structure, a specified binding site, and a compound library, a downstream oracle (such as a docking or co-folding method) evaluates one candidate compound at that site per invocation and returns a score. The goal is to select a shortlist with favorable oracle scores under a budget on the number of oracle invocations or the time available for evaluation. PRECISE-MCTS combines fast code-based retrieval with adaptive oracle-guided selection to address this problem (Section 2.6, Figure 1).

2.2 SURFACE REPRESENTATION

The surface mesh is constructed using *Michael Sanner's Molecular Surface (MSMS)* algorithm (Sanner et al., 1996), and augmented with four additional per-vertex features capturing key physicochemical and geometric properties. These properties, computed using MaSIF (Marchand et al., 2025), are: (a) hydropathy (Kyte & Doolittle, 1982), (b) Poisson–Boltzmann continuum electrostatics (Jurrus et al., 2018), (c) hydrogen bonding characteristics (Kortemme et al., 2003) as implemented in RDKit (RDKit Community, 2024), and (d) the local shape index (Koenderink & Van Doorn, 1992). If the target includes metal ions or post-translational modifications (PTMs), **MaSIF-neosurf** (Marchand et al., 2025) is used to extract these features. During surface processing we additionally remove hydrogens from the all-atom structure. Surface preprocessing and featurization are described in Appendix B.2.

Shape index. While the first three quantities encode chemical aspects of the surface region, the shape index captures local surface geometry. In ablation studies, we observe that the shape index is

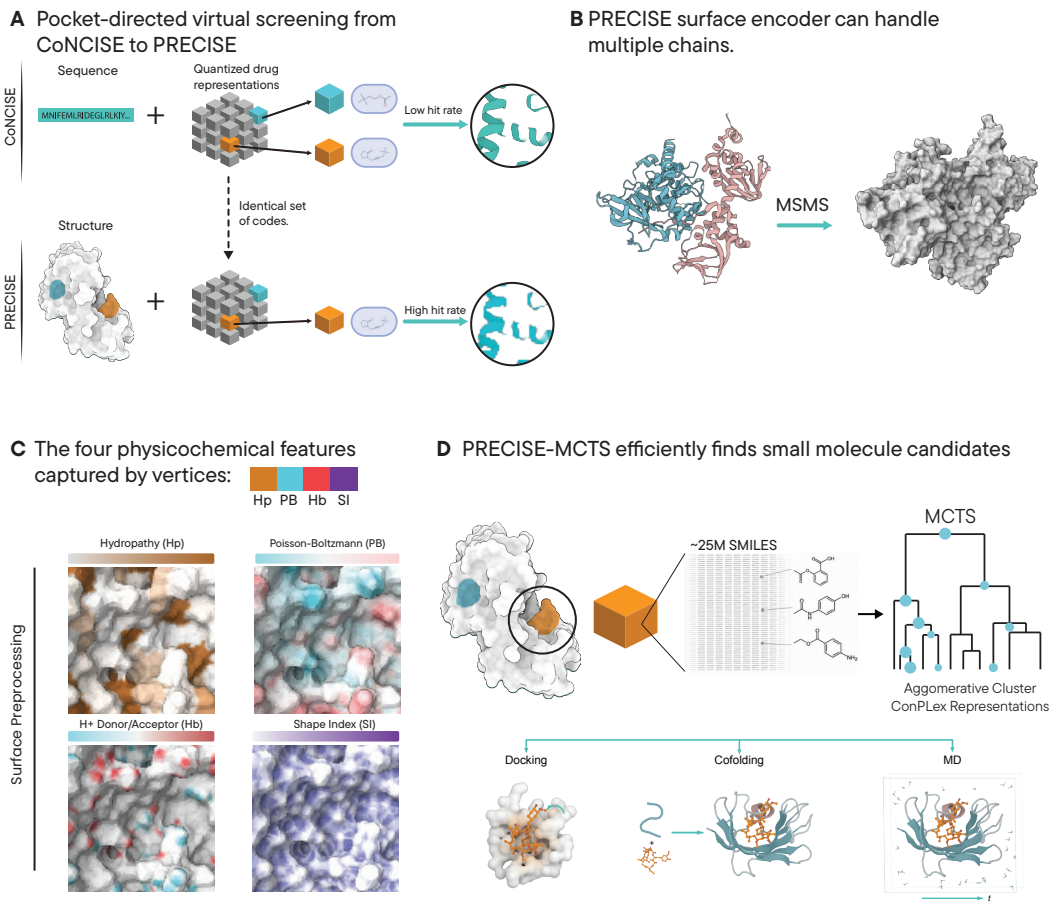


Figure 1: **Overview** A) PRECISE uses the same codebook index for small molecules as CoNCISE to enable ultra-fast screening of large databases but PRECISE takes as input a target's 3D structure, rather than the protein sequence B) PRECISE's surface-based approach accommodates quaternary structures (here, a complex consisting of two protein chains) without any model reconfiguration. C) PRECISE's surface representation consists of four physico-chemical and geometric properties. Once these are computed, we don't further utilize the protein sequence. D) PRECISE-MCTS uses PRECISE to identify site-specific codes, retrieves ligands mapped to those codes, and employs an efficient tree-based Monte Carlo sampling pipeline to select small-molecule candidates. The efficiency of the tree search makes usage of more expensive downstream methods tractable.

particularly important for enabling the model to learn and generalize surface–ligand binding characteristics (Table 3). Conceptually, the shape index helps distinguish pocket regions (which are more favorable for binding (Naderi et al., 2018)) from surface protrusions (Proposition 1, Appendix G).

Surface processing via geometric deep learning. The surface mesh naturally defines a graph structure, with vertices connected through local neighborhoods, making it well suited for graph-attention models. Accordingly, we process the protein surface using a recent model, GotenNet (Aykent & Xia, 2025), which efficiently combines graph-based message passing with the rotational equivariance required to represent 3D meshes. Collectively, the GotenNet-based surface module extracts three inputs from the surface mesh:

1. a set of 3D vertex coordinates, $\boldsymbol{X}_s \in \mathbb{R}^{n_s \times 3}$;
2. a graph structure, $G_s = (V_s, E_s)$, where $|V_s| = n_s$ defines the mesh connectivity; and
3. a set of per-vertex features, $Y_s \in \mathbb{R}^{n_s \times 4}$, capturing localized physicochemical and geometric properties.

GotenNet accepts these three inputs $(\boldsymbol{X}_s, G_s, Y_s)$ and applies message passing to propagate information across mesh vertices while strictly adhering to rotational equivariance of the steerable features (Appendix B.1). It produces two outputs: (a) an equivariant representation $\tilde{\mathbf{X}}_s \in \mathbb{R}^{n_s \times k_s \times d_s}$,

where k_s and d_s denote the steerable and invariant feature dimensions, respectively; and (b) a fully invariant embedding $\tilde{Y}_s \in \mathbb{R}^{n_s \times d_s}$.

The final surface representation is fully invariant and is obtained by normalizing the steerable component along the k_s dimension and concatenating it with the invariant output:

$$S = \|\tilde{\mathbf{X}}_s\|_2^{(\text{mid})} \oplus \tilde{Y}_s, \tag{1}$$

where $\|\cdot\|_2^{(\text{mid})}$ denotes the application of the L_2 norm along the middle (k_s) dimension, and $\oplus$ represents concatenation along the feature axis. Following a non-linear down-projection from $2d_s$ to d_s dimensions, the resulting tensor $S \in \mathbb{R}^{n_s \times d_s}$ serves as the final invariant surface representation produced by the PRECISE surface module. A more detailed description of the surface block is provided in Appendix B.1.

2.3 LIGAND REPRESENTATIONS

PRECISE processes the ligand in its drug module by first transforming its SMILES string into a Morgan fingerprint (specifically, ECFP-4 (Rogers & Hahn, 2010)). These fingerprints are then discretized using CoNCISE 's pretrained drug encoder. CoNCISE 's encoder was originally trained on binary DTI data derived solely from sequence–SMILES pairs, without any explicit structural information. Using finite scalar quantization (FSQ below; (Mentzer et al., 2023)), CoNCISE organized the entire space of small molecules into a 3-tier hierarchy, with 32 categories at each level, resulting in 32,768 (=32^3) drug codes. In principle, given ECFP-4 fingerprint $E \in \mathbb{R}^{2048}$ the ligand representation L is obtained using the following steps:

$$\begin{aligned}
F_0 &= Linear(E, d_l) \\
Q_i &= GELU(FSQ_i(F_{i-1})), && i = \{1, 2, 3\} \\
F_i &= F_{i-1} - Q_i && Q_i \text{ quantized} \\
C &= Q_1 \oplus Q_2 \oplus Q_3 && \textbf{code representation} \\
L &= NonLinear(C, d_s) && \oplus = \text{concat.}
\end{aligned} \tag{2}$$

PRECISE thus introduces a framework for systematically combining knowledge from binary DTI data with spatial supervision from protein–ligand structures. Molecular codes learned from sequence–SMILES interaction labels are transferred, without retraining the codebook, to a surface model trained on 3D complexes. This integration demonstrates that binary DTI supervision can support spatially resolved binding prediction, while improving whole-protein DTI accuracy over CoNCISE (Table 1) and capturing pocket-level binding specificity (Section 3.2).

2.4 WHOLE-PROTEIN AND POCKET-LEVEL PREDICTION

Whole-protein head. For whole-protein DTI prediction, we mean-pool the surface embeddings of a ligand-conditioned pocket and concatenate the resulting representation with the ligand's CoNCISE code representation. A logistic regression head maps this joint representation to a whole-protein DTI score (Appendix A).

Patch-level head. The patch-level head integrates the surface and ligand representations $S \in \mathbb{R}^{n_s \times d_s}$ and $L \in \mathbb{R}^{1 \times d_s}$ through an inner product operation to produce vertex-associated binding likelihoods:

$$p = \text{sigmoid}(S \cdot L^T) \in \mathbb{R}^{n_s} \tag{3}$$

The PRECISE forward pass therefore yields per-vertex binding scores for a given ligand. In practice, however, ligand binding is not localized to a single surface point but instead involves a contiguous surface region. To account for this, we average the entries of p over vertices within a standard 7Å surface patch centered at a given coordinate to estimate the binding likelihood at that location. Unless stated otherwise, we refer to these patch-aggregated values as PRECISE binding scores in the remainder of this work.

Pocket-directed screening. To screen a specific target and pocket against a database of compounds, the input to PRECISE is the target structure and, in that reference frame, the coordinates of the pocket center. We obtain all surface vertices within 4.5 Å of the center, run the GotenNet surface module on the whole surface, and extract the processed surface representations of the vertices of interest. We note that this 4.5 Å radius is used to define the pocket around the supplied center and is distinct from the 7 Å radius used to define surface patches for local binding-score aggregation. Each vertex is then scored against all 32,768 precomputed discretized drug codes, using our inner product formulation. Per-code scores are averaged across vertices, and PRECISE reports a ranked list of drug codes that

are predicted to bind strongly to the target at the pocket. Ligands associated with the top-scoring codes form the candidate pool for downstream evaluation. The geometric deep learning architecture for the surface track is the compute-heavy component, but needs to be computed only once.

2.5 TRAINING

We trained PRECISE on the PLINDER dataset (Durairaj et al., 2024), a curated collection of ≈74,000 protein–ligand interaction systems with holo (i.e., bound) 3D structures. We used default train, validation, and test splits for model training, hyperparameter optimization, and evaluation.

The model was trained on surface–ligand association prediction. Ground-truth binding patches were identified from input protein–ligand coordinates, with random non-binding regions sampled as negatives. Training ran for 10 epochs on a single Nvidia H200 GPU (30 hours) using AdamW with learning rate 1×10^{-4}. We addressed class imbalance with focal loss ($\alpha = 0.7$, $\gamma = 2$) (Lin et al., 2017). Training and validation statistics are provided in Appendix Figure 6.

The pretrained CoNCISE encoder was frozen during training, while the surface module and the ligand projection were learned from the structural data.

2.6 PRECISE-MCTS

PRECISE-MCTS combines Monte Carlo sampling of descendant ligands and backpropagation of oracle scores with depth-wise, beam-style pruning. This streamlined, MCTS-inspired design uses a depth-dependent slack rule in place of UCT selection to allocate a limited oracle budget. Its contribution lies in integrating molecular retrieval, hierarchical search, and downstream oracle feedback under matched evaluation budgets.

Given a protein site, PRECISE identifies high-affinity codes, which can then be used to retrieve the corresponding small molecules from code-annotated ligand databases. However, as ligand libraries increase in size, code occupancy rises accordingly, requiring additional, more sophisticated downstream approaches to reduce the candidate set to a scale suitable for experimental evaluation. This motivated us to develop PRECISE-MCTS, an end-to-end drug discovery pipeline that combines PRECISE with any downstream scoring oracle of choice.

Internally, PRECISE-MCTS uses ligands belonging to the top-scoring PRECISE codes and organizes them by their chemical and binding similarities into an agglomerative tree. Given a fixed computational budget, which imposes a hard limit on the number of docking, co-folding, or MD evaluations that can be performed, PRECISE-MCTS traverses the tree, pruning branches that show lower binding propensity while focusing exploration on regions with better oracle scores. To construct this ligand tree, we use ligand embeddings from ConPLex (Singh et al., 2023), a contrastive sequence-based DTI method that represents ligands by embedding their Morgan fingerprints into a vector space. Each leaf represents a compound, each internal node represents a cluster of its descendant compounds, and the root contains the full candidate pool. This clustering tree is separate from the CoNCISE codes used to select that pool. Once the pool is fixed, PRECISE's predicted scores do not enter tree construction, sampling, or pruning; exploration is guided solely by observed oracle scores.

The search starts at the root and proceeds one depth at a time. For each child of an active node, we sample a descendant leaf uniformly at random and evaluate that ligand using the downstream oracle; if the child is itself a leaf, we evaluate it directly. This randomized sampling is the Monte Carlo component: it probes a cluster's binding potential through individual compounds. Previously evaluated ligands reuse their cached scores. Each score is propagated along the path from the sampled leaf to the root, updating each node's best observed descendant score. Thus, an internal node is scored through actual oracle evaluations of compounds in its subtree.

The exploration strategy operates on a per-depth basis. We express the oracle objective so that lower scores are better, and denote a node's best observed descendant score by $b(q)$. Among nodes eligible for further expansion at depth d, let sc_d^* be the best of these scores. We retain a node q only if $b(q) < sc_d^* + \delta_d$, where the positive, depth-dependent *slack* δ_d is the allowed margin from the best observed score. Subsequent exploration is restricted to the retained nodes (Appendix Algorithm 7). Broader slack near the root encourages exploration across clusters, while progressively tighter slack at deeper levels focuses evaluations on promising branches. The algorithm returns the best evaluated ligands when the search finishes or the oracle budget is exhausted (Algorithm 6).

Theoretical Analysis. We analyze this search on trees in which the range of descendant oracle scores at depth d is less than α/β^d, for constants $\alpha > 0$ and $\beta > 1$ (Definition 3). Under this condition, choosing $\delta_d = \alpha/\beta^d$ preserves the branch containing an optimal ligand: any sampled

descendant on that branch scores within the slack of the optimum, and hence within the slack of the best observed score. With unrestricted exploration, the search therefore reaches an optimal leaf (Proposition 2). Once the within-branch score range is below ε, the associated early-termination result gives an ε-optimal evaluated candidate (Corollary 3). The assumptions and proofs are given in Appendix H; these results do not guarantee optimality under an arbitrary finite oracle budget. Finite-budget performance is evaluated empirically in Section 3.4.

3 RESULTS

3.1 PRECISE ACCURATELY PREDICTS WHOLE-PROTEIN DRUG–TARGET INTERACTIONS

The PLINDER (Durairaj et al., 2024) benchmark provides a machine-learning testbed. It splits both targets and ligands into structurally-dissimilar train/val/test sets. Using PLINDER-specified target-ligand complexes as positives, we constructed negative examples by randomly pairing targets and ligands from the same split. In this whole-protein binary classification setting, we compare PRECISE with other high-throughput DTI methods, some of which can also scale to billion-compound libraries: as a baseline we consider k-means, but we also test sequence-based methods Komet (Guichaoua et al., 2024), ConPLex (Singh et al., 2023) and CoNCISE (Erden et al., 2025). We also test a variant of ConPLex that replaces the original ConPLex PLM with the Raygun (Devkota et al., 2026) PLM. For PRECISE and DrugCLIP, the two methods that utilize the 3D structure of the target, we pass the unbound (apo) 3D structure (provided for all targets by PLINDER). Here we used PRECISE's prediction-head that predicts binding to the overall protein. However, DrugCLIP (Gao et al., 2023) needed adaptation for this setting: in addition to the structure of the target, it also requires that the drug-binding pocket be passed to the method. We therefore supply DrugCLIP with the true protein pocket. Notably, no other method received this pocket information. As Table 1 shows, PRECISE achieves state of the art performance on this whole-protein prediction task.

Method	AUPR	AUROC
k-Means	0.423	0.671
Komet	0.608	0.757
ConPLex	0.614	0.788
ConPLex+Raygun	0.620	0.821
DrugCLIP (+ true pocket)	0.562	0.834
CoNCISE	0.641	0.892
PRECISE	**0.663**	**0.908**

Table 1: Whole-protein DTI performance on PLINDER with a positive-to-negative ratio of $1:5$ (random-ranking AUPR: $1/6 \approx 0.167$). PRECISE uses the logistic head in Appendix A. Higher is better; best means are bolded.

3.2 PRECISE CAPTURES BINDING SPECIFICITY AT THE POCKET LEVEL

We assessed PRECISE's scoring of pocket-ligand compatibility in two complementary ways: given a pocket, can PRECISE retrieve codes containing favorable ligand candidates and given a ligand, can it localize the binding pocket?

We selected ten proteins (with known binding pockets) with weak baseline Vina scores from a published screening benchmark to form a stress-test panel (Appendix F.2). Because PRECISE scores codes rather than individual compounds, we randomly sample 500 molecules each from high-scoring and low-scoring codes, respectively. We then docked both groups at the specified site with Vina. Molecules from high-scoring codes consistently achieved more favorable docking scores across the ten pockets (Figure 2). Thus, PRECISE-predicted codes enrich for docking-compatible candidates. Later, we show how this can be further improved by oracle-guided search.

On the DUD-E decoy dataset (Mysinger et al., 2012), PRECISE also retrieves genuine binders from decoys with a mean AUROC of 0.766, comparable to DrugCLIP's 0.771, although only DrugCLIP receives the binding pocket as input (Appendix B.3).

We also investigated the reverse direction: given a ligand, can PRECISE predict its binding pocket? On protein–ligand samples from PLINDER's evaluation split and RCSB structures deposited after December 2024 ("Recent"), PRECISE achieves mean distances between predicted and true pocket centers of 4.27 Å and 3.40 Å, respectively (Appendix Table 2; Appendix C.1 for evaluation details). PRECISE performance was similar on a PRECISE variant trained with full ligand fingerprints, indicating that ligand discretization into codes does not materially affect pocket prediction accuracy.

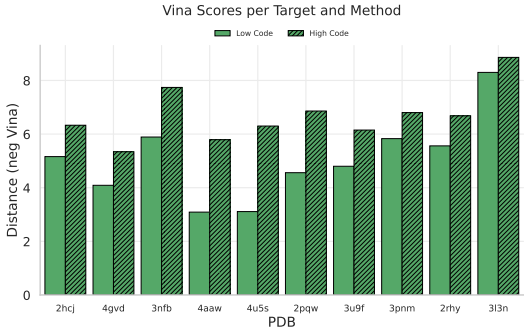


Figure 2: **High-ranked PRECISE codes identify stronger docking candidates with fewer evaluations.** Across ten proteins, ligands sampled from high-ranked codes have more favorable Vina scores than those from low-ranked codes. 500 candidates were randomly selected from each of high- and low-scoring codes. Scores are reported as the negative of Vina scores, so higher is better.

3.3 PRECISE GENERALIZES ZERO-SHOT TO COMPOSITE TARGET SURFACES

Our surface-based approach extends screening beyond single-chain targets to metal-ion-containing pockets and multi-chain interfaces, without retraining (evaluation details in Appendix E.2).

Metalloprotein surfaces. We randomly selected 40 proteins from the MetalProGNet training set (Jiang et al., 2023) and compared PRECISE scores with and without the metal ion. Scores were higher in the presence of metal ions ($p = 0.03$, one-sided t-test; Figure 3B), despite the model never being trained on such surfaces. As a case study, we examined Zn^{2+}-mediated ligand binding in adenosine deaminase (ADA) (Wilson et al., 1991). Using MaSIF-NeoSurf (Marchand et al., 2025), we constructed surfaces with and without Zn^{2+}; its presence increased the predicted pocket binding score, consistent with its known role in stabilizing ligand binding (Figure 3A).

Multi-chain interfaces. We evaluated PRECISE on 100 randomly sampled TernaryDB complexes (Xue et al., 2025), where ligands bind at shared interfaces between protein chains. Pocket scores were significantly higher for the full complexes than after removing Chain B (Figure 3D), despite training exclusively on monomeric inputs. For example, the antibody in PDB `1Y0L` requires both heavy and light chains to bind benzimidazolium (Debler et al., 2005). PRECISE assigns a score of ~ 0.4 with both chains present versus ~ 0.2 with only Chain A (Figure 3C).

3.4 PRECISE-MCTS IMPROVES SCREENING UNDER LIMITED ORACLE BUDGETS

Candidate selection across downstream oracles. We evaluated PRECISE-MCTS with Vina, Boltz, and Protenix on the ten-protein stress-test panel described in Section 3.2. These oracles assess protein–ligand interaction in varied ways: Vina scores docked poses, Boltz predicts affinity using an IC50-trained model, and Protenix reports protein–ligand ipTM confidence. For each oracle, we compared three strategies under matched evaluation budgets: random sampling from the compound library, random sampling from high-scoring PRECISE codes, and adaptive selection from these high-scoring codes using PRECISE-MCTS. Figure 4 reports the best score and the mean score of the top ten candidates.

Random sampling from high-scoring PRECISE codes improves candidate quality over random sampling from the library, showing that the code scores align with these downstream assessments. PRECISE-MCTS improves candidate quality further by using oracle feedback to guide subsequent evaluations. Because the comparison without MCTS already selects candidates from high-scoring codes, this second comparison tests the additional contribution of adaptive search. Across Vina, Boltz, and Protenix, PRECISE-MCTS improves the top-10 mean in 29 of 30 target–oracle comparisons.

Comparison with Rag2Mol-R. We benchmarked PRECISE-MCTS against Rag2Mol-R (Zhang et al., 2025), a retrieval-oriented method that generates ligand templates against a pocket and queries them against a compound library. We retrieved Rag2Mol-R's published results for 65 protein–site pairs. We chose a more demanding computational setting for PRECISE-MCTS: searching the same library with fewer docking calls. Rag2Mol-R searched ZINC-250M with an average of 20,426 docking evaluations per target; PRECISE-MCTS searches it with a budget of at most 500 docking calls. Despite this tighter budget, PRECISE-MCTS outperforms Rag2Mol-R in both top-1 and top-3 median Vina scores (Table 6). Here we report the negative Vina scores, so higher is better.

Larger libraries improve candidates at a fixed oracle budget. We replaced ZINC-250M with Enamine-7B and re-screened the ten-protein stress-test panel, holding the docking budget fixed.

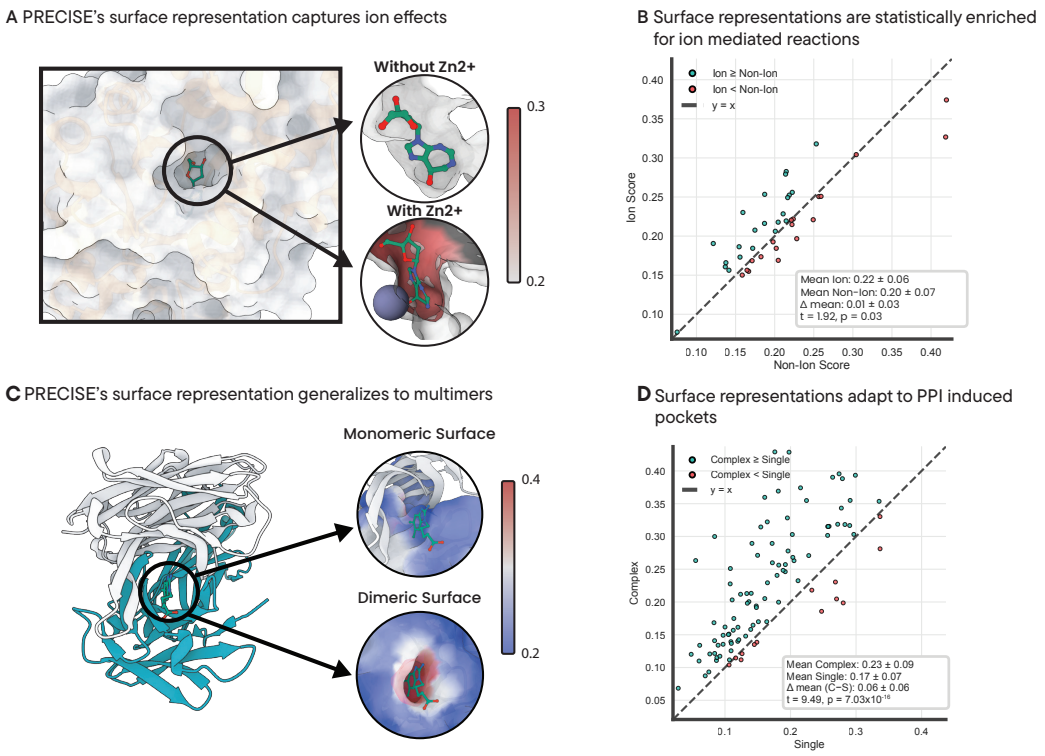


Figure 3: **Zero-shot prediction on composite surfaces.** A) PRECISE assigns a higher ligand-binding score to adenosine deaminase when Zn^{2+} is included in the surface representation. B) Paired scores for 40 metal-ion-containing binding sites and their corresponding ion-removed surfaces. C) Both chains of the catalytic antibody 1Y0L contribute to the predicted ligand-binding surface. D) Paired scores for 100 TernaryDB complexes and the corresponding surfaces after removal of Chain B.

PRECISE-MCTS returned a stronger best Vina score in eight of ten cases (Figure 7). The larger library gives the search access to better candidates without requiring additional docking.

Screening efficiency. PRECISE first scores 32,768 ligand codes rather than individual compounds, after which PRECISE-MCTS spends a fixed oracle budget within high-scoring code regions. This codebook formulation keeps the initial scoring cost independent of the indexed library's size and limits the growth in total screening time. Expanding the library 28-fold, from ZINC-250M to Enamine-7B, increased median processing time by only 80%, from 75 to 135 minutes, with no increase in docking calls. Across the libraries in Figure 8, PRECISE-MCTS completes screening in under five hours and requires 130 GB of storage at 70 billion compounds.

4 DISCUSSION

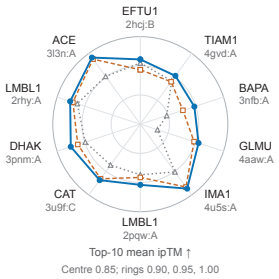
PRECISE addresses a fundamental limitation in ultra-high-throughput virtual screening (uHTVS): unlike sequence-based binary DTI predictors, it can identify binding sites and restrict searches to user-specified pockets. The surface-based abstraction provides an important additional benefit: automatic zero-shot generalization to complex targets. Once the surface mesh is characterized, PRECISE discards all sequence information and operates purely on local geometric and physicochemical properties. A metalloprotein is simply a surface with altered electrostatic patterns; a protein-protein interface presents as extended surface geometry spanning multiple chains.

Critically, CoNCISE's drug codes transfer effectively to this structure-based task despite being learned from sequence-based DTI data. A lightweight nonlinear projection suffices to map codes into the surface-compatible embedding space. This cross-modal transfer suggests that the hierarchical codes—learned from binary DTI data with only protein sequence inputs—capture fundamental principles of molecular recognition rather than simply sequence artifacts. They thus provide a powerful way to connect two data modalities of DTI studies: experimental binding affinity assays

A Protenix

	Protenix-PRECISE-MCTS		Protenix-PRECISE		Protenix Random	
Target	Best	Top-10 mean	Best	Top-10 mean	Best	Top-10 mean
EFTU1_ECOLI	0.974	**0.961**	0.974	0.943	**0.983**	0.955
TIAM1_HUMAN	**0.958**	**0.952**	0.954	0.940	0.956	0.942
BAPA_SPHXN	**0.966**	**0.947**	**0.966**	0.927	0.948	0.898
GLMU_STRPN	**0.984**	**0.955**	**0.984**	0.947	0.940	0.881
IMA1_HUMAN	**0.990**	**0.987**	**0.990**	0.983	0.985	0.951
LMBL1_HUMAN	0.960	**0.954**	0.953	0.941	**0.981**	0.937
CAT_ECOLX	**0.980**	**0.969**	0.977	0.966	0.951	0.937
DHAK_ECOLI	**0.983**	**0.975**	0.980	0.963	0.976	0.949
LMBL1_HUMAN	0.983	**0.976**	0.980	0.972	**0.988**	0.964
ACE_HUMAN	**0.993**	**0.991**	0.992	0.988	0.983	0.951

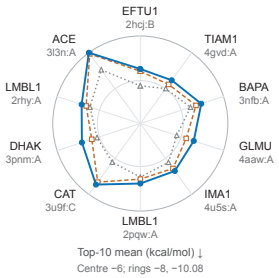
B Protenix confidence



C Boltz

	Boltz-PRECISE-MCTS		Boltz-PRECISE		Boltz Random	
Target	Best	Top-10 mean	Best	Top-10 mean	Best	Top-10 mean
EFTU1_ECOLI	-8.812	**-8.539**	-8.812	-8.434	**-8.894**	-7.754
TIAM1_HUMAN	**-8.975**	**-8.490**	-8.583	-8.249	-8.684	-8.012
BAPA_SPHXN	**-9.225**	**-8.983**	-9.153	-8.726	-8.697	-8.456
GLMU_STRPN	**-8.894**	**-8.616**	-8.824	-8.246	-8.384	-7.784
IMA1_HUMAN	**-9.810**	**-8.732**	**-9.810**	-8.585	-8.517	-8.234
LMBL1_HUMAN	**-9.415**	**-8.793**	-8.831	-8.587	-8.945	-8.495
CAT_ECOLX	**-9.776**	**-9.512**	-9.597	-9.372	-8.675	-8.222
DHAK_ECOLI	**-9.528**	**-8.858**	-9.054	-8.374	-8.806	-8.117
LMBL1_HUMAN	-9.119	**-8.871**	-9.119	-8.651	**-9.318**	-8.463
ACE_HUMAN	**-10.476**	**-10.075**	**-10.476**	-10.004	-9.413	-9.092

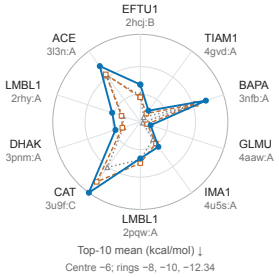
D Boltz affinity proxy



E Vina

	Vina-PRECISE-MCTS		Vina-PRECISE		Vina Random	
Target	Best	Top-10 mean	Best	Top-10 mean	Best	Top-10 mean
EFTU1_ECOLI	**-8.942**	**-8.701**	-8.473	-7.784	-8.141	-7.927
TIAM1_HUMAN	**-7.207**	**-7.004**	-7.011	-6.748	-6.552	-6.420
BAPA_SPHXN	**-11.215**	**-10.989**	-10.852	-10.216	-10.095	-9.518
GLMU_STRPN	**-7.300**	**-6.781**	-6.820	-6.523	-6.636	-6.198
IMA1_HUMAN	**-8.654**	**-8.225**	-8.419	-7.928	-7.785	-7.566
LMBL1_HUMAN	-8.882	-8.644	**-9.315**	**-8.983**	-8.936	-8.736
CAT_ECOLX	**-12.543**	**-12.335**	-11.777	-11.363	-10.620	-10.065
DHAK_ECOLI	**-8.427**	**-7.891**	-7.919	-7.370	-7.913	-7.278
LMBL1_HUMAN	**-8.434**	**-8.143**	-7.826	-7.430	-8.017	-7.339
ACE_HUMAN	**-11.302**	**-10.992**	-10.612	-10.252	-10.499	-10.124

F Vina docking scores



Method-PRECISE-MCTS Method PRECISE Method Random

Figure 4: **Search efficiency under each oracle's own objective.** Best and top-10 mean scores on ten target pockets with 500 oracle calls per strategy: PRECISE-MCTS, random sampling from high-scoring PRECISE codes, and random sampling from the library. A&B: Protenix ipTM confidence (Zhang et al., 2026); C&D: Boltz predicted affinity (Passaro et al., 2025); E&F: Vina docking scores (Trott & Olson, 2010). Best values are bolded. PRECISE-MCTS improves top-10 means in 29 of 30 target–oracle comparisons.

vs those based on structure elucidation. The resulting model contains just 17 million parameters yet provides robust, generalizable accuracy, suggesting these local properties capture the essential physics of molecular recognition when combined with rich drug representations.

PRECISE-MCTS addresses a critical pain-point in uHTVS: how to prioritize candidates after an initial ranking. The latent drug representations, sourced from ConPLex Singh et al. (2023), enable construction of an agglomerative tree linking structurally-similar molecules. Monte Carlo tree search then efficiently explores this space, using slower docking or co-folding scores as an oracle to further prioritize hits. These comparisons measure search efficiency under each oracle's own objective: finding better-scoring candidates within the same evaluation budget.

The design of PRECISE implies certain limitations. It is trained on rigid-body conformations, consistent also with most docking methods, and does not model apo-to-holo conformational changes. The limited availability of matched apo-holo training data makes it unclear whether such approaches substantially improve upon rigid-body assumptions. Future work could learn surface representations that predict conformational responses to ligand binding, potentially by training on molecular dynamics trajectories of binding events.

AI USE STATEMENT

PRECISE is a method based on protein language models, so in that sense AI is at the foundation of this work. However, we did not use generative AI tools for writing or other tasks with required disclosure except we did ask AI to polish some sentences and help us shorten the main narrative to fit within the required page limit. The authors take full responsibility for the final content of this work.

ETHICS STATEMENT

The authors disclose that they have filed a provisional patent related to this work. They are committed to making their methods free for academic and non-commercial use, but if the patent is awarded, they will charge commercial companies to use PRECISE

REPRODUCIBILITY STATEMENT

All details necessary to reproduce the method and results of this paper are given in the main paper plus the supplemental Appendix; datasets with training/testing splits are described in detail in Sections D and E of the Appendix; pseudocode for PRECISE-MCTS is Algorithm 6 in the Appendix, and PRECISE itself, is described in full in Appendix B including the forward algorithm for the Surface Block (Algorithm 1) and the description of how the GotenNet architecture is incorporated (Algorithm 2). In addition we provide a link to our code in an anonymous GitHub at https://anonymous.4open.science/r/PRECISE/README.md

AUTHOR CONTRIBUTIONS

Removed for anonymous review

ACKNOWLEDGMENTS AND FUNDING

Removed for anonymous review

REFERENCES

Josh Abramson, Jonas Adler, Jack Dunger, Richard Evans, Tim Green, Alexander Pritzel, Olaf Ronneberger, Lindsay Willmore, Andrew J Ballard, Joshua Bambrick, et al. Accurate structure prediction of biomolecular interactions with AlphaFold 3. *Nature*, 630(8016):493–500, 2024.

Sarp Aykent and Tian Xia. GotenNet: Rethinking Efficient 3D Equivariant Graph Neural Networks. In *The Thirteenth International Conference on LearningRepresentations*, 2025. URL `https://openreview.net/forum?id=5wxCQDtbMo`.

Nathan A Baker, David Sept, Simpson Joseph, Michael J Holst, and J Andrew McCammon. Electrostatics of nanosystems: application to microtubules and the ribosome. *Proceedings of the National Academy of Sciences*, 98(18):10037–10041, 2001.

Helen M Berman, John Westbrook, Zukang Feng, Gary Gilliland, Talapady N Bhat, Helge Weissig, Ilya N Shindyalov, and Philip E Bourne. The protein data bank. *Nucleic acids research*, 28(1): 235–242, 2000.

Gabriele Corso, Hannes Stärk, et al. Diffdock: Diffusion steps, twists, and turns for molecular docking. *arXiv preprint arXiv:2210.01776*, 2022.

Dawson-Haggerty et al. trimesh. URL `https://trimesh.org/`.

Erik W Debler, Shuichiro Ito, Florian P Seebeck, Andreas Heine, Donald Hilvert, and Ian A Wilson. Structural origins of efficient proton abstraction from carbon by a catalytic antibody. *Proceedings of the National Academy of Sciences*, 102(14):4984–4989, 2005.

Kapil Devkota, Daichi Shonai, Joey Mao, Young Su Ko, Wei Wang, Scott Soderling, and Rohit Singh. Miniaturizing and modifying natural proteins with raygun. *Nature*, pp. 1–9, 2026.

Todd J Dolinsky, Paul Czodrowski, Hui Li, Jens E Nielsen, Jan H Jensen, Gerhard Klebe, and Nathan A Baker. PDB2PQR: expanding and upgrading automated preparation of biomolecular structures for molecular simulations. *Nucleic acids research*, 35(suppl_2):W522–W525, 2007.

Janani Durairaj, Yusuf Adeshina, Zhonglin Cao, Xuejin Zhang, Vladas Oleinikovas, Thomas Duignan, Zachary McClure, Xavier Robin, Gabriel Studer, Daniel Kovtun, et al. PLINDER: The protein-ligand interactions dataset and evaluation resource. *BioRxiv*, pp. 2024–07, 2024.

Jerome Eberhardt, Diogo Santos-Martins, Andreas F Tillack, and Stefano Forli. Autodock vina 1.2. 0: new docking methods, expanded force field, and python bindings. *Journal of chemical information and modeling*, 61(8):3891–3898, 2021.

Mert Erden, Kapil Devkota, Lia Varghese, Lenore Cowen, and Rohit Singh. Learning a concise language for small-molecule binding. In *International Conference on Research in Computational Molecular Biology*, pp. 260–263. Springer, 2025.

Pablo Gainza, Freyr Sverrisson, Frederico Monti, Emanuele Rodola, Davide Boscaini, Michael M Bronstein, and Bruno E Correia. Deciphering interaction fingerprints from protein molecular surfaces using geometric deep learning. *Nature Methods*, 17(2):184–192, 2020.

Bowen Gao, Bo Qiang, Haichuan Tan, Yinjun Jia, Minsi Ren, Minsi Lu, Jingjing Liu, Wei-Ying Ma, and Yanyan Lan. Drugclip: Contrasive protein-molecule representation learning for virtual screening. In *NeurIPS 2023*, 2023. URL https://openreview.net/forum?id=lAbCgNcxm7.

Oleksandr O Grygorenko. Enamine ltd.: the science and business of organic chemistry and beyond, 2021.

Gwenn Guichaoua, Philippe Pinel, et al. Drug–target interactions prediction at scale: The Komet algorithm with the LCIdb dataset. *Journal of Chemical Information and Modeling*, 2024.

Dejun Jiang, Zhaofeng Ye, Chang-Yu Hsieh, Ziyi Yang, Xujun Zhang, Yu Kang, Hongyan Du, Zhenxing Wu, Jike Wang, Yundian Zeng, et al. MetalProGNet: a structure-based deep graph model for metalloprotein–ligand interaction predictions. *Chemical Science*, 14(8):2054–2069, 2023.

Elizabeth Jurrus, Dave Engel, Keith Star, Kyle Monson, Juan Brandi, Lisa E Felberg, David H Brookes, Leighton Wilson, Jiahui Chen, Karina Liles, et al. Improvements to the APBS biomolecular solvation software suite. *Protein science*, 27(1):112–128, 2018.

Jan J Koenderink and Andrea J Van Doorn. Surface shape and curvature scales. *Image and vision computing*, 10(8):557–564, 1992.

Tanja Kortemme, Alexandre V Morozov, and David Baker. An orientation-dependent hydrogen bonding potential improves prediction of specificity and structure for proteins and protein–protein complexes. *Journal of molecular biology*, 326(4):1239–1259, 2003.

Jack Kyte and Russell F Doolittle. A simple method for displaying the hydropathic character of a protein. *Journal of molecular biology*, 157(1):105–132, 1982.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988, 2017.

Anthony Marchand, Stephen Buckley, Arne Schneuing, Martin Pacesa, Maddalena Elia, Pablo Gainza, Evgenia Elizarova, Rebecca M Neeser, Pao-Wan Lee, Luc Reymond, et al. Targeting protein–ligand neosurfaces with a generalizable deep learning tool. *Nature*, 639(8054):522–531, 2025.

Andrew T McNutt, Paul Francoeur, Rishal Aggarwal, Tomohide Masuda, Rocco Meli, Matthew Ragoza, Jocelyn Sunseri, and David Ryan Koes. GNINA 1.0: molecular docking with deep learning. *Journal of cheminformatics*, 13(1):43, 2021.

Fabian Mentzer, David Minnen, Eirikur Agustsson, and Michael Tschannen. Finite scalar quantization: Vq-vae made simple. *arXiv preprint arXiv:2309.15505*, 2023.

Susan Miller, Joel Janin, Arthur M Lesk, and Cyrus Chothia. Interior and surface of monomeric proteins. *Journal of molecular biology*, 196(3):641–656, 1987.

Ron Milo. What is the total number of protein molecules per cell volume? a call to rethink some published values. *Bioessays*, 35(12):1050–1055, 2013.

Garrett M Morris, Ruth Huey, William Lindstrom, Michel F Sanner, Richard K Belew, David S Goodsell, and Arthur J Olson. Autodock4 and autodocktools4: Automated docking with selective receptor flexibility. *Journal of computational chemistry*, 30(16):2785–2791, 2009.

Alessandro Muntoni and Paolo Cignoni. PyMeshLab, January 2021.

Michael M Mysinger, Michael Carchia, John J Irwin, and Brian K Shoichet. Directory of useful decoys, enhanced (dud-e): better ligands and decoys for better benchmarking. *Journal of medicinal chemistry*, 55(14):6582–6594, 2012.

Misagh Naderi, Jeffrey Mitchell Lemoine, Rajiv Gandhi Govindaraj, Omar Zade Kana, Wei Pan Feinstein, and Michal Brylinski. Binding site matching in rational drug design: algorithms and applications. *Briefings in Bioinformatics*, 20(6):2167–2184, 08 2018. ISSN 1477-4054. doi: 10.1093/bib/bby078. URL https://doi.org/10.1093/bib/bby078.

Saro Passaro, Gabriele Corso, Jeremy Wohlwend, Mateo Reveiz, Stephan Thaler, Vignesh Ram Somnath, Noah Getz, Tally Portnoi, Julien Roy, Hannes Stark, et al. Boltz-2: Towards accurate and efficient binding affinity prediction. *BioRxiv*, 2025.

Rodrigo Quiroga and Marcos A Villarreal. Vinardo: A scoring function based on autodock vina improves scoring, docking, and virtual screening. *PloS one*, 11(5):e0155183, 2016.

RDKit Community. Rdkit: Open-source cheminformatics. https://www.rdkit.org, 2024. Accessed: 2026-01-28.

David Rogers and Mathew Hahn. Extended-connectivity fingerprints. *Journal of chemical information and modeling*, 50(5):742–754, 2010.

Michel F Sanner, Arthur J Olson, and Jean-Claude Spehner. Reduced surface: an efficient way to compute molecular surfaces. *Biopolymers*, 38(3):305–320, 1996.

Rohit Singh, Samuel Sledzieski, Bryan Bryson, Lenore Cowen, and Bonnie Berger. Contrastive learning in protein language space predicts interactions between drugs and protein targets. *Proceedings of the National Academy of Sciences*, 120(24):e2220778120, 2023.

C Sullivan and Alexander Kaszynski. Pyvista: 3d plotting and mesh analysis through a streamlined interface for the visualization toolkit (vtk). *Journal of Open Source Software*, 4(37):1450, 2019.

Chai Discovery team, Jacques Boitreaud, Jack Dent, Matthew McPartlon, Joshua Meier, Vinicius Reis, Alex Rogozhonikov, and Kevin Wu. Chai-1: Decoding the molecular interactions of life. *BioRxiv*, pp. 2024–10, 2024.

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.

David K Wilson, Frederick B Rudolph, and Florante A Quiocho. Atomic structure of adenosine deaminase complexed with a transition-state analog: understanding catalysis and immunodeficiency mutations. *Science*, 252(5010):1278–1284, 1991.

J Michael Word, Simon C Lovell, Jane S Richardson, and David C Richardson. Asparagine and glutamine: using hydrogen atom contacts in the choice of side-chain amide orientation. *Journal of molecular biology*, 285(4):1735–1747, 1999.

Fanglei Xue, Meihan Zhang, Shuqi Li, Xinyu Gao, James A. Wohlschlegel, Wenbing Huang, Yi Yang, and Weixian Deng. SE(3)-equivariant ternary complex prediction towards target protein degradation. *Nature Communications*, 16(1):5514, July 2025. ISSN 2041-1723. doi: 10.1038/s41467-025-61272-5.

Shuangye Yin, Elizabeth A Proctor, Alexey A Lugovskoy, and Nikolay V Dokholyan. Fast screening of protein surfaces using geometric invariant fingerprints. *Proceedings of the National Academy of Sciences*, 106(39):16622–16626, 2009.

Yuejiang Yu, Chun Cai, Jiayue Wang, Zonghua Bo, Zhengdan Zhu, and Hang Zheng. Uni-dock: Gpu-accelerated docking enables ultralarge virtual screening. *Journal of chemical theory and computation*, 19(11):3336–3345, 2023.

Peidong Zhang, Xingang Peng, Rong Han, Ting Chen, and Jianzhu Ma. Rag2mol: structure-based drug design based on retrieval augmented generation. *Briefings in Bioinformatics*, 26(3), 2025.

Yuxuan Zhang, Chengyue Gong, Jinyuan Sun, Jiaqi Guan, Milong Ren, Song Xue, Hanyu Zhang, Wenzhi Ma, Zhenyu Liu, Xinshi Chen, et al. Protenix-v2: Broadening the reach of structure prediction and biomolecular design. *bioRxiv*, pp. 2026–04, 2026.

A WHOLE-PROTEIN DTI PREDICTION

Once PRECISE has been trained on surface–ligand association prediction, we use it to score the protein surface for a given ligand. Mean-shift clustering then identifies candidate pockets from spatial concentrations of high-scoring patches. Let $\mathcal{V}_*(P, \ell)$ denote the vertices of the selected pocket for protein P and ligand ℓ, and let $\mathbf{s}_v(P)$ denote the surface embedding at vertex v. We obtain a pocket representation by mean pooling and concatenate it with the ligand's ConCISE code representation C_ℓ:

$$\mathbf{h}(P, \ell) = \frac{1}{|\mathcal{V}_*(P, \ell)|} \sum_{v \in \mathcal{V}_*(P,\ell)} \mathbf{s}_v(P), \tag{4}$$

$$\mathbf{z}(P, \ell) = \mathbf{h}(P, \ell) \oplus C_\ell. \tag{5}$$

A logistic regression head maps this joint representation to the whole-protein DTI score:

$$s_{\mathrm{DTI}}(P, \ell) = \sigma\big(\mathbf{w}^\top \mathbf{z}(P, \ell) + b\big), \tag{6}$$

where $\oplus$ denotes concatenation, σ is the sigmoid function, and $\mathbf{w}$ and b are the learned parameters of the logistic regression head.

The pocket-localization procedure is described in Appendix C.

B MODEL AND SURFACE DESCRIPTION

B.1 PRECISE ARCHITECTURE

Surface block. As summarized in the main paper, the steerable and non-steerable surface features are mainly processed through the GotenNet architecture. The forward algorithm for the surface block is described below:

Algorithm 1: Surface Block

Data: surface steerable feature: $\boldsymbol{X}_s \in \mathbb{R}^{n_s \times 3}$, invariant feature $Y_s \in \mathbb{R}^{n_s \times 4}$, edge index $E_s \in \mathbb{Z}_+^{n_e \times 2}$

Result: invariant surface feature $S \in \mathbb{R}^{n_s \times d_s}$

`/* use GotenNet model to process the surface inputs */`

$\tilde{\mathbf{X}}_s, \tilde{Y}_s$ = GotenNet($\boldsymbol{X}_s, Y_s, E_s$)

$S' = \|\tilde{\mathbf{X}}_s\|_2^{(\mathrm{mid})} \oplus \tilde{Y}_s$

$S = Linear(GeLU(Linear(S', d_s)), d_s)$

return S

GotenNet message passing. Internally, GotenNet processes the surface features through a graph-attention based, equivariant message passing mechanism. The overall framework begins with the initialization of three feature tensors derived from the surface inputs: non-steerable $\mathcal{Y} \in \mathbb{R}^{n_s \times d_s}$, steerable $\boldsymbol{\mathcal{X}} \in \mathbb{R}^{n_s \times k \times d_s}$ and edge-associated and invariant $t_{ij} \in \mathbb{R}^{n_e \times d_s}$ features (n_s, n_e denotes the number of surface nodes and edges respectively). Note that, as the default value of ℓ_{max} (maximum spherical harmonics degree) is set to 1, this leads to the steerable dimension (k) being equal to 3. GotenNet initialization is followed by the iterative application of GotenNet's specialized GATA module (described in more detail below), followed by an equivariant feed-forward network, until the final processed steerable and invariant embeddings are returned as output (Algorithm 2).

The GATA module in GotenNet performs message passing and mixing information between the steerable and non-steerable features. The high-level description of the GATA module is described in Algorithm 3. This algorithm is intended as an abstract description of GotenNet, omitting various minor reshaping and transformation steps for clarity. For more implementation details, see the PRECISE GitHub repository.

Ligand encoding block. The ligand encoder takes the CoNCISE discretized code embeddings and applies a nonlinear projection to match the dimension of the surface representation (Algorithm 4).

Surface–ligand scoring. Algorithm 5 combines the surface and ligand blocks to return per-vertex binding scores (Equation 3). Patch-level scores average these outputs over a surface patch; the whole-protein logistic head is described separately in Appendix A.

Algorithm 2: High-level description of GotenNet architecture used in PRECISE

Data: surface steerable feature: $\boldsymbol{X}_s \in \mathbb{R}^{n_s \times 3}$, invariant feature $Y_s \in \mathbb{R}^{n_s \times 4}$, edge index $E_s \in \mathbb{Z}_+^{n_e \times 2}$

/* Initializations */

$\boldsymbol{\mathcal{X}} = \mathbf{0} \in \mathbb{R}^{n_s \times 3 \times d_s}$

$\mathcal{Y} = Linear(Y_s, d_s)$

$\boldsymbol{r}_{ij}^0, \boldsymbol{r}_{ij}^1 = \texttt{SphericalHarmonics}_{0,1}(\boldsymbol{X}_{s,i} - \boldsymbol{X}_{s,j})$

$d_{ij} = \texttt{exp_norm_smearing}(\boldsymbol{X}_{s,i}, \boldsymbol{X}_{s,j})$ // Distance

$t_{ij} = d_{ij} \circ (\mathcal{Y}_i + \mathcal{Y}_j)$

for $i \in \{1, 2, \ldots, N_{layers}\}$ **do**

  $\boldsymbol{\mathcal{X}}, \mathcal{Y}, t_{ij} = GATA(\boldsymbol{\mathcal{X}}, \mathcal{Y}, t_{ij}, E_s, \boldsymbol{r}_{ij}^0, \boldsymbol{r}_{ij}^1)$

  $\boldsymbol{\mathcal{X}}, \mathcal{Y} = EQFF(\boldsymbol{\mathcal{X}}, \mathcal{Y})$ // Equivariant Neural Network

return $\boldsymbol{\mathcal{X}}, \mathcal{Y}$

Algorithm 3: High level overview of GATA module

Data: $\boldsymbol{\mathcal{X}}, \mathcal{Y}, t_{ij}$, edge index E, $\boldsymbol{r}_{ij}^0, \boldsymbol{r}_{ij}^1$. Number of heads assumed as 1

/* Initialize attentions per node */

$q = Linear(\mathcal{Y}, d_s)$

$k = Linear(\mathcal{Y}, d_s)$

$v_1 = Linear(\mathcal{Y}, 3d_s)$

$v_2 = Linear(\mathcal{Y}, 3d_s)$

$t_{ij}^{attn} = Linear(t_{ij}, d_s)$

$t_{ij}^{filter} = Linear(t_{ij}, d_s)$

/* Calculate attention across neighbors compute softmax (using the edge index E_{ij}) */

$att_{ij} = q_i * k_j * t_{ij}^{attn}$

$att_{ij} = softmax(att_{ij})$

$sea_{ij} = att_{ij} \circ v_{1,j}$ /* shape should be $[n_e, 1, 3d_s]$ */

$cutoff = \texttt{cutoff}(\boldsymbol{r}_{ij}^0)$

$spatial_{ij} = t_{ij}^{filter} \circ v_{2,j} \circ cutoff$ /* shape should be $[n_e, 1, 3d_s]$ */

$att_{ij}^* = spatial_{ij} + sea_{ij}$ /* combined attention */

$c_{ij}^0, c_{ij}^1, c_{ij}^2 = \texttt{split}(att^*, d_s, dim = -1)$ /* divide the final attention to three chunks */

/* updates per edge */

$\mathcal{Y}_{ij}^{diff} = c_{ij}^0$

$\boldsymbol{\mathcal{X}}_{ij}^{diff} = c_{ij}^1 \circ \boldsymbol{r}_{ij}^1$ /* This is an outer product; $c_{ij}^1 \in \mathbb{R}^{n_e \times 1 \times d_s}, \boldsymbol{\mathcal{X}}_{ij} \in \mathbb{R}^{n_e \times 3 \times 1} \rightarrow \boldsymbol{\mathcal{X}}_{ij}^{diff} \in \mathbb{R}^{n_e \times 3 \times d_s}$ */

$\boldsymbol{\mathcal{X}}_{ij}^{diff} = \boldsymbol{\mathcal{X}}_{ij}^{diff} + c_{ij}^2 \circ \boldsymbol{\mathcal{X}}_{ij}$

$t_{ij}^{diff} = \texttt{edge_update}(t_{ij}, E_{ij}, \boldsymbol{r}_{ij}^1, \ldots)$

/* aggregate edge updates through summation */

$\boldsymbol{\mathcal{X}}^{diff} = aggregate(\boldsymbol{\mathcal{X}}_{ij}^{diff}, E_{ij})$

$\mathcal{Y}^{diff} = aggregate(\mathcal{Y}_{ij}^{diff}, E_{ij})$

/* finally, sum and return */

$t_{ij} = t_{ij} + t_{ij}^{diff}$

$\boldsymbol{\mathcal{X}} = \boldsymbol{\mathcal{X}} + \boldsymbol{\mathcal{X}}^{diff}$

$\mathcal{Y} = \mathcal{Y} + \mathcal{Y}^{diff}$

return $\boldsymbol{\mathcal{X}}, \mathcal{Y}, t_{ij}$

Algorithm 4: PRECISE drug block

Data: Ligand morgan fingerprint $L_{mf} \in \mathbb{R}^{2048}$

$L_{concise} = \texttt{concise_drug}(L_{mf})$

$L = Linear(SiLU(Linear(L_{concise}, d_s)), d_s)$

return L

Algorithm 5: PRECISE per-vertex scoring

Data: Surface data $Surf$, Ligand fingerprint L_{mf}

$\boldsymbol{X}_s, Y_s, E_s = Surf$

$S = \text{Surface_Block}(\boldsymbol{X}_s, Y_s, E_s)$ `// Algorithm 1`

$L = \text{Ligand_Block}(L_{mf})$ `// Algorithm 4`

$p = \text{sigmoid}(SL^T)$

return p

B.2 SURFACE MESH CONSTRUCTION AND FEATURIZATION

The first step in the surface construction pipeline was the addition of hydrogen atoms using a protonation tool called `Reduce` (Word et al., 1999). Following this, we used `MSMS` (Sanner et al., 1996) to triangulate the molecular surface, applying standard parameters for vertex density (3.0) and water probe radius (1.5 Å). The resulting triangulated surface was then downsampled to a target edge length of approximately 1 Å using the `trimesh` (Dawson-Haggerty et al.) and `pymeshlab` (Muntoni & Cignoni, 2021) libraries. This processed mesh served as the substrate for calculating four key surface properties: Poisson–Boltzmann continuum electrostatics, proton donor/acceptor characteristics, hydropathy, and shape index.

Poisson–Boltzmann continuum electrostatics The first step in computing local surface electrostatics involved pre-processing the structural file using the `PDB2PQR` tool (Dolinsky et al., 2007). Following this, the Poisson-Boltzmann electrostatic potential for each protein molecule was calculated using the `APBS` (Baker et al., 2001) (v.3.4.1) suite.

The resulting `APBS` output assigned charge information to every vertex on the generated mesh surface, using the `Multivalue` utility within `APBS` for the computation. Finally, the charges were capped between –30 and +30 and then normalized to fall within the range –1 to +1.

Hydrogen-bond donors and acceptors. On the `MSMS`-constructed surface, a vertex was classified as a potential donor or acceptor if its closest associated atom was a polar hydrogen, a nitrogen, or an oxygen atom. Based on the orientation of the heavy atoms, a value was then assigned to each vertex, drawn from a Gaussian distribution (Kortemme et al., 2003). This value scale ranged from -1 (indicating the optimal position for a hydrogen bond acceptor) to +1 (indicating the optimal position for a hydrogen bond donor).

Hydropathy Each vertex on the molecular surface was also assigned a scalar hydropathy value based on the Kyte and Doolittle scale (Kyte & Doolittle, 1982). The specific value was determined by identifying the amino acid to which the vertex's closest atom belonged.

The original values from this scale ranged from -4.5 (most hydrophilic) to +4.5 (most hydrophobic). Consistent with the other surface metrics, these hydropathy values were also normalized to a final range of -1 to 1.

Shape index The shape index is a scalar value that describes the local topography of the protein surface at each vertex, with respect to its local curvature (Yin et al., 2009). This measure is defined by the principal curvatures, κ_1 and κ_2 (where $\kappa_1 \geq \kappa_2$).

To calculate this property, we first computed the Mean (H) and Gaussian (K) curvatures for each vertex on the mesh using the `PyVista`(Sullivan & Kaszynski, 2019) library. From these values, the principal curvatures (κ_1 and κ_2) were derived for each vertex. The final shape index (SI) was then computed using the standard formula:

$$SI = \frac{2}{\pi} \arctan\left(\frac{\kappa_1 + \kappa_2}{\kappa_1 - \kappa_2}\right)$$

During this computation, a small epsilon (1×10^{-8}) was added to the denominator to maintain numerical stability in flat or spherical regions where the principal curvatures may be nearly equal. This calculation yields a normalized value in $[-1, 1]$; its sign depends on the surface-normal convention.

B.3 DUD-E DECOY COMPARISONS USING PRECISE

The DUD-E dataset comprises 102 protein targets, each paired up with between 2700 and 51500 small molecules Mysinger et al. (2012), most of which are decoys. For each protein, PRECISE inferred the most likely binding pocket by identifying the surface patch with the highest ligand confidence score and used it to rank all the associated small molecules. DrugCLIP, however, requires a protein pocket to target as input. Therefore, providing additional pocket information, along with 20 RDKit-generated conformers for each ligand, we obtained the DrugCLIP-ranked list and calculated the average AUROC score as the comparison baseline. An additional Enrichment Factor (EF) score was also calculated for the two methods:

$$EF_{x\%} = \frac{(\text{actives in top } x\%)/(\text{compounds in top } x\%)}{(\text{total actives})/(\text{total compounds})} \tag{7}$$

PRECISE (AUROC=0.766; $EF_{5\%}$=8) and DrugCLIP (AUROC=0.771; $EF_{5\%}$=8.69) showed similar ranking performance, although only DrugCLIP was provided with pocket information. This result is particularly notable because PRECISE assigns the same code to some compounds and therefore cannot distinguish their ranks within a code.

B.4 SURFACE FEATURES FOR ION-CONTAINING COMPLEXES

The above pipeline was specific to protein-only calculations. However, for complexes containing monatomic ions (e.g., Zn^{2+}, Mg^{2+}), a modified protocol was implemented. For geometric features (shape index, curvature), the ion's surface vertices were computed analogously to the protein. For the chemical features (hydropathy, H-bond potential), which are defined by amino acid or organic functional group properties, vertices on the ion surface were assigned default neutral values, as these metrics were not directly applicable.

The most significant adaptation occurred in the Poisson–Boltzmann continuum electrostatics pipeline. A custom script enacted a branching logic where standard (non-metal) organic ligands were parameterized automatically by `PDB2PQR` using the `--ligand` flag. In contrast, for monatomic ions, `PDB2PQR` was executed without the `--ligand` flag. Our script then parsed the ion's MOL2 file, mapped its atom type to a predefined formal charge and PARSE-compatible radius, and appended this information as a new `HETATM` record to the `PDB2PQR` output. This augmented PQR file, integrating both protein and ion, was then used as the input for `APBS`'s `multivalue` tool, ensuring the ion's electrostatic contribution was accurately incorporated.

C POCKET LOCALIZATION

Given a ligand, we seek to identify the surface regions most likely to bind it. We apply the base PRECISE algorithm by uniformly sampling K_s points across the protein surface and computing patch-aggregated binding scores at each. Because nearby vertices produce similar scores, this sampling provides efficiency without sacrificing spatial resolution. Aggregation across these scores then reveals candidate binding sites: we apply a mean-shift procedure to locate regions where scores are both strong and spatially concentrated, returning cluster centers as prospective binding pockets.

Binding pocket prediction also provides an opportunity to assess the impact of ligand quantization on accuracy. Unlike high-throughput screening, where discretization is essential for scalability, pocket prediction involves a single ligand, removing the computational constraint. We therefore compared the standard PRECISE model against a variant that retains full ligand fingerprints without quantization, denoted 'PRECISE (no codes)' in Table 2.

C.1 POCKET-LOCALIZATION EVALUATION

We evaluated pocket prediction on PLINDER's evaluation split and on the Recent RCSB dataset described in Appendix E.1. Performance is measured as the distance between predicted and true pocket centers. We compare with LABind, DiffDock, Chai, Boltz-2, P2RANK, and VNEGNN (Table 2); the docking and co-folding methods address broader tasks, of which pocket identification is one component.

C.2 NUMBER OF SAMPLED PATCHES

We choose the number of sampled patches to cover the solvent-accessible surface and represent each vertex in multiple patches. For an average protein of approximately 350 amino acids (Milo, 2013),

Method	PLINDER	Recent	Time (s)*
Boltz-2	10.61 ± 8.99	<u>2.29</u> ± 7.05	48.9 ± 21.1
Chai	11.20 ± 9.95	**2.26** ± 7.37	41.3 ± 18.0
DiffDock	5.33 ± 12.72	13.54 ± 9.63	90.2 ± 33.6
LABind	4.70 ± 8.30	3.84 ± 7.40	30.2 ± 9.8
P2RANK	9.33 ± 9.85	6.80 ± 6.44	8.8 ± 1.3
VNEGNN	9.60 ± 9.28	7.77 ± 7.47	16.2 ± 21.7
PRECISE (no codes)	**4.18** ± 7.23	4.77 ± 6.79	<u>5.3</u> ± 2.9
+Refine	3.95 ± 6.84	3.41 ± 5.69	±
PRECISE (codes)	<u>4.27</u> ± 6.73	3.40 ± 6.27	$\mathbf{5.1 \times 10^{-5}}$ ± -
+Refine	4.02 ± 6.71	2.97 ± 5.38	±

Table 2: Distance (in Å) between the predicted and ground-truth pocket centers (lower is better). PRECISE is able to predict binding pockets for structures in low sequence- and pocket-homology test examples. On post-2024 PDB entries ("Recent"), PRECISE maintains similar performance. *Amortized time per ligand on ZINC-7M, including processing time.

the solvent-accessible surface area is about $A = 15000\,\text{Å}^2$ (Miller et al., 1987). Approximating each radius-r patch by a disk of area πr^2 and sampling centers independently and uniformly, a fixed vertex is covered with probability $p \approx \pi r^2/A$. Its number of covering patches therefore satisfies

$$K \sim \text{Binomial}(n, p), \qquad \Pr(K = 0) = (1-p)^n, \qquad \mathbb{E}[K] = np.$$

For $r = 7\,\text{Å}$ and $n = 2048$, this approximation gives a probability of about 7×10^{-10} that a fixed vertex is uncovered and approximately 21 covering patches per vertex.

C.3 Mean-shift bandwidth and score threshold

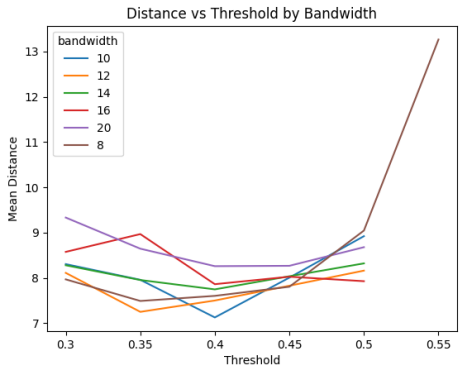


Figure 5: Pocket-localization distance on the PLINDER evaluation set across mean-shift bandwidths and score thresholds.

We varied the mean-shift kernel bandwidth and the threshold above which patch scores are treated as positive (Figure 5). Although a bandwidth of 10 Å and threshold 0.4 gave the lowest mean distance, we used 12 Å and 0.35, respectively, for a more robust pairing.

D Training diagnostics and feature ablations

D.1 Training and validation statistics

PRECISE was trained on the PLINDER training dataset, with the validation statistics (AUPR and AUROC scores) reported twice every epoch. These metrics are provided in Figure 6.

In addition to standard evaluation metrics, we also assessed how well our model performs relative to a randomized baseline on the validation datasets. Specifically, we computed the ratio between the validation AUPR achieved by PRECISE and that of a theoretical randomized baseline, which we refer to as the AUPR gain. Using this metric as our model-selection criterion, we selected the epoch-10 checkpoint as the default PRECISE model.

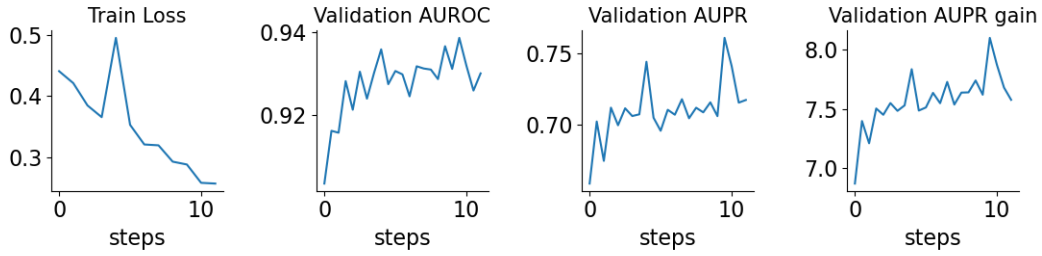


Figure 6: Train and validation statistics obtained from PRECISE training. For 'Validation AUPR gain', we computed the ratio of validation AUPR to the AUPR of a randomized algorithm

D.2 PER-VERTEX FEATURE ABLATIONS

We assessed the contribution of each surface feature by training PRECISE variants with one feature removed (Table 3).

Table 3: Ablation study on PLINDER. Removing electrostatics (PB) or shape (SI) features causes the largest performance drops, demonstrating their importance for binding site prediction.

Variant	AUPRC	Gain over random	AUROC
No PB	0.24	2.5	0.50
No SI	0.17	1.9	0.43
No Hp	0.51	5.5	0.81
No Hb	0.55	6.0	0.83
Full	**0.68**	**7.4**	**0.90**

Features: PB = Poisson–Boltzmann electrostatics;
SI = Shape Index; Hp = Hydropathy; Hb = Hydrogen-bond donor/acceptor.

E EVALUATION DATASETS

E.1 RECENT RCSB POCKET-LOCALIZATION DATASET

We supplemented PLINDER evaluation with a curated dataset of recent monomer–ligand structures from the RCSB PDB (Berman et al., 2000). We excluded revisions of structures released earlier. The query returned 253 CIF files; after identifying bound ligands and filtering out those with molecular weight below 100, we obtained 100 surface-mesh examples. Their PDB identifiers are listed in Table 4. This dataset was not filtered by sequence or pocket homology.

E.2 COMPOSITE-SURFACE EVALUATION

For the metal-ion evaluation, we randomly sampled 40 proteins from the MetalProGNet training set (Jiang et al., 2023) and compared pocket scores with and without the ion. Ion-containing surfaces were featurized as described in Appendix B.4.

For multi-chain evaluation, we randomly sampled 100 of the 22,303 complexes in TernaryDB (Xue et al., 2025). We compared scores at the ligand-binding site in the full complex with scores at the same site after removing Chain B. The evaluation structures are listed in the accompanying GitHub data.

F SCREENING PROTOCOLS AND SUPPLEMENTARY COMPARISONS

F.1 CONSTRUCTION OF NAIVE SCREENING BASELINES

For the 65-target comparison in Table 6, we compared PRECISE-MCTS with two naive PRECISE-based baselines. These baselines were generated by first constructing the agglomerative tree using the ligands' ConPLex representations (as in PRECISE-MCTS), then directly clustering the ligands and selecting a representative candidate from each cluster for docking. For both PRECISE-MCTS and the naive methods, the linkage method was Ward. Additionally, for the naive deep and shallow clustering methods, the distance thresholds were set to 0.4 and 0.7, respectively.

Algorithm 6: PRECISE-MCTS with an oracle evaluation budget

Data: Candidate ligands $\mathcal{C}$ retrieved from high-scoring PRECISE codes; target P; site center x; oracle $\mathcal{O}$; budget B; depth-dependent slack δ_d
Result: Evaluated ligands and their oracle scores, ranked by the chosen objective

Embed ligands in $\mathcal{C}$ using ConPLex and construct a binary agglomerative tree T;
Convert the oracle objective to a score g for which lower is better;
Initialize active nodes $Q = \{\mathrm{root}(T)\}$ and an empty ligand-score cache D;
Initialize each node's best observed score $b(q) = +\infty$;
while $Q \neq \varnothing$ **do**
    Initialize $Q_{\mathrm{next}} = \varnothing$;
    foreach *child c of an active node in Q* **do**
        Choose a descendant leaf ℓ of c uniformly at random (or $\ell = c$ if c is a leaf);
        if $\ell \notin D$ **then**
            if *the evaluation budget is exhausted* **then**
                return *evaluated ligands in D, ranked by score*;
            Evaluate $\mathcal{O}(P, x, \ell)$, store its transformed score $D[\ell] = g(\ell)$, and update budget usage;
        Update the best observed score of every node on the path from ℓ to the root using $D[\ell]$;
        if *c is not a leaf* **then**
            Add c to Q_{next};
    if $Q_{\mathrm{next}} = \varnothing$ **then**
        return *evaluated ligands in D, ranked by score*;
    Set $Q = \mathrm{TRIM}(Q_{\mathrm{next}}, b, \delta_d)$ at the current candidate depth d;
return *evaluated ligands in D, ranked by score*;

Table 4: PDB identifiers for the 100 Recent pocket-localization examples.

0	9NB4	9QK3	9NIJ	9QDJ	9VIS	9HWR	9I4Z	9MT3	9NIK	9MTB
10	9OVJ	9NJI	9M3O	9HWS	9R0Q	9NIM	9LR6	9I4H	9QT6	9I1N
20	9QEK	9QK5	9NSK	9R4L	9NXH	9LB6	9R4T	9OS6	9N6N	9NSX
30	9NT0	9NBF	9NBK	9P14	9LB7	9O63	9QDT	9N6L	9NSZ	9UDY
40	9QEI	9OAF	9U42	9OG8	9I0T	9NH9	9QQ1	9MT9	9N2A	9QVL
50	9QDH	9N6M	9O8X	9OBJ	9QEA	9M3U	9QK4	9RBG	9OAM	9ODX
60	9R1P	9QBR	9N1U	9M3P	9OOZ	9RUV	9LMX	9QQ0	9NSW	9R4R
70	9MRB	9RBT	9P15	9MS1	9NNM	9IAP	9QC3	9VD8	9U46	9QXN
80	9OHS	9OB8	9LMS	9NIL	9QGC	9R4M	9QPZ	9NNA	9QGD	9IAW
90	9NNI	9LMV	9NBJ	9IB4	9NNO	9MSV	9O0S	9UFR	9NJJ	9NSY

General approach for controlling exploration in the agglomerative tree The TRIM function retains nodes whose best observed oracle score is within the depth-dependent slack of the best score at that depth (Algorithm 7). Scores are expressed so that lower is better, as in Section 2.6.

Algorithm 7: TRIM

Data: Set $Q = \{(q, s)\}_{i=1}^{N}$ of nodes and their scores, Depth d, slack value δ_d
Result: A subset $Q' \subset Q$ selected for further exploration

$s_{min} = \min(s_1, \ldots, s_N)$ /* best binding score */
$Q' = \{\}$
for $(q, s) \in Q$ **do**
    if $s < s_{min} + \delta_d$ **then**
        Append (q, s) to Q'
return Q'

Table 5 lists the two per-depth slack schedules. The table indexes the root at depth 0; the theoretical analysis in Appendix H indexes it at depth 1.

Table 5: Depth-dependent slack for the two PRECISE-MCTS settings. The root has depth 0 in this table.

Depth	Shallow	Deep	Depth	Shallow	Deep
0	4.00	4.00	11	0.25	0.75
1	3.00	3.00	12	0.25	0.75
2	2.00	3.00	13	0.12	0.75
3	2.00	2.00	14	0.12	0.75
4	2.00	2.00	15	0.12	0.50
5	1.00	2.00	16	0.12	0.50
6	1.00	2.00	17	0.12	0.50
7	0.75	2.00	18	0.12	0.50
8	0.75	2.00	19	0.12	0.50
9	0.50	1.00	20	0.12	0.25
10	0.50	1.00	21	0.12	0.25

Note that the minimum slack values for the shallow and deep settings are 0.12 and 0.25, respectively. If the search proceeds beyond depth 21, PRECISE-MCTS continues to use the minimum slack until the search terminates.

Additional PRECISE-MCTS parameters For both MCTS settings, we selected the top 500 scoring codes. For the shallow setting, we randomly chose 20 ligands from each code to construct the agglomerative tree, whereas for the wide setting, we selected 100 ligands per code. Additionally, in one protein instance where the top 500 codes had minimal ligand coverage (TIAM1_HUMAN_840_931_0_4gvd_A), we increased the number of selected codes to 1000 for the deep setting. The evaluated protein structures are listed in the accompanying GitHub data.

Table 6: Top-1 and Top-3 docking scores, median across 65 diverse protein targets provided by Rag2Mol, screened on ZINC-250M.

Method	**Docking calls**	**Median score**	
		Top-1	**Top-3**
RAG2Mol	20426	8.58 ± 1.00	8.40 ± 1.01
PRECISE-MCTS	500	$\mathbf{8.81} \pm 0.72$	$\mathbf{8.52} \pm 0.75$

F.2 SELECTION OF 10 DIFFICULT-TO-DOCK PROTEINS AS A STRESS-TEST PANEL

Among the 65 Rag2Mol protein-pocket pairs where automated docking evaluation was possible, we selected 10 proteins where Rag2Mol-generated candidates had the weakest binding affinities. We used these proteins for more comprehensive benchmarking (Section 3.4).

F.3 SUPPLEMENTARY SCREENING COMPARISONS

Figure 7 collects the larger-library, docking-score, and runtime comparisons supporting Section 3.4.

F.4 COMPARISON ACROSS EMPIRICAL DOCKING SCORES

We evaluated selected ligands using Vina, Vinardo (Quiroga & Villarreal, 2016), and AD4 (Morris et al., 2009) (Table 7). High-code selections score better than low-code selections for all eight systems under each scoring function. MCTS further improves on high-code selection in five of eight systems for Vinardo and seven of eight for AD4.

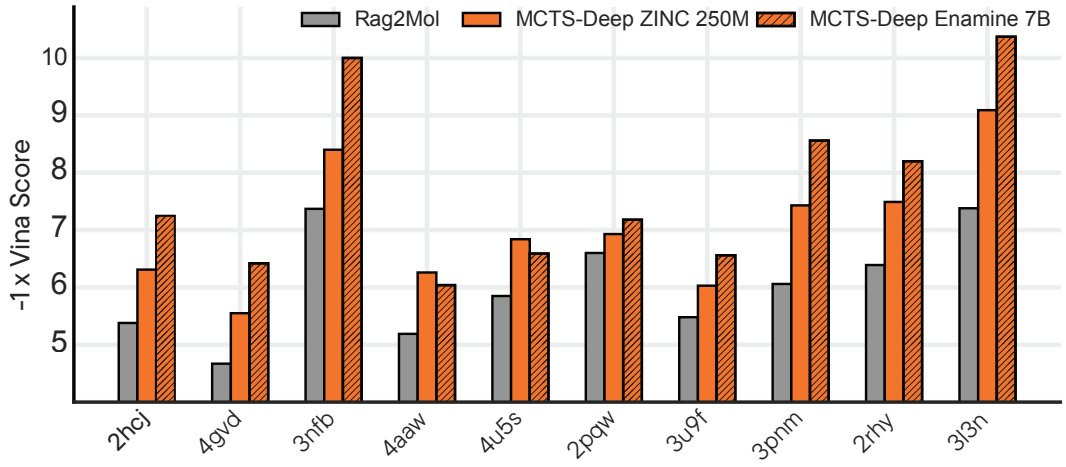


Figure 7: On 10 difficult-to-dock protein-pocket examples, PRECISE-MCTS finds higher scoring molecules on larger screening databases for 8 of the ten targets.

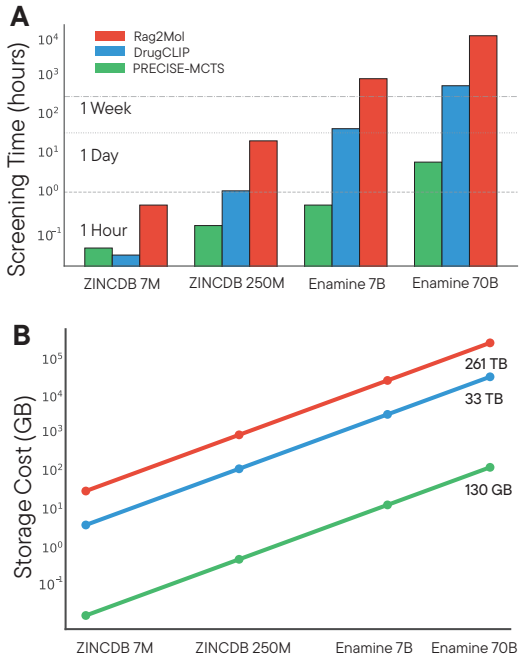


Figure 8: A) Runtime comparison across large compound libraries. PRECISE-MCTS screens the indexed libraries in under 5 hours, while DrugCLIP and Rag2Mol require hundreds to thousands of hours at 70 billion compounds. B) Storage requirements across the same compound libraries. At 70 billion compounds, PRECISE-MCTS uses 130 GB, compared with 33 TB for DrugCLIP and 261 TB for Rag2Mol.

G SHAPE-INDEX GEOMETRY

Definition 1. Let $M \subset \mathbb{R}^3$ be a C^2 embedded surface equipped with the induced Euclidean metric and a locally defined unit normal field N. The shape operator at p is

$$S_p(X) = -\nabla_X N, \qquad X \in T_pM. \tag{8}$$

The derivative is tangent because $\langle \nabla_X N, N\rangle = \frac{1}{2} X\langle N, N\rangle = 0$. Let $\kappa_1 \geq \kappa_2$ be the eigenvalues of S_p. For $\kappa_1 > \kappa_2$, define

$$\mathrm{si}_p = \frac{2}{\pi} \arctan\left(\frac{\kappa_1 + \kappa_2}{\kappa_1 - \kappa_2}\right). \tag{9}$$

For equal nonzero curvatures, use the limiting value $\mathrm{sign}(\kappa_1)$; set the index to zero when both curvatures vanish.

Table 7: Docking scores (kcal/mol) for ligands from low- and high-scoring codes, with MCTS selections additionally reported for Vinardo and AD4. Lower is better.

	Vina		**Vinardo**			**AD4**		
System	Low	High	Low	High	MCTS	Low	High	MCTS
2hcj	-5.16	-6.33	-4.98	-5.53	-5.66	-5.89	-6.33	-6.75
3nfb	-5.89	-7.74	-4.16	-6.72	-6.19	-5.69	-8.48	-8.64
2pqw	-4.56	-6.86	-3.62	-6.32	-6.54	-5.09	-7.29	-8.42
4u5s	-3.11	-6.30	-6.07	-6.52	-7.31	-6.55	-6.85	-7.39
3u9f	-4.80	-6.15	-3.92	-5.45	-5.42	-3.71	-5.55	-6.04
3pnm	-5.83	-6.80	-5.34	-5.67	-6.63	-6.20	-9.22	-8.82
2rhy	-5.56	-6.68	-4.59	-5.39	-5.67	-7.15	-8.95	-9.30
3l3n	-8.30	-8.86	-5.82	-7.03	-7.00	-7.32	-9.10	-9.27

Definition 2. Let $U \subset M$ be open and equipped with a smooth unit normal field N. We call U a pocket if $\langle q-p, N_p\rangle > 0$ for all distinct $p, q \in U$, and a protrusion if $\langle q-p, N_p\rangle < 0$ for all distinct $p, q \in U$.

Proposition 1. *Let M be a 2-dimensional embedded submanifold of $\mathbb{R}^3$. Let si_p denote the shape index at $p \in M$ (Definition 1). If $U \subset M$ is a pocket (Definition 2), $\forall p \in U$, the shape index $\mathrm{si}_p \geq 0$. If U is a protrusion, $\mathrm{si}_p \leq 0$.*

Proof. For tangent vector fields X, Y, differentiating $\langle N, Y\rangle = 0$ gives $\langle S(X), Y\rangle = \langle N, \nabla_X Y\rangle$. Since the Euclidean connection is torsion-free and $[X, Y]$ is tangent, this expression is symmetric in X, Y, so S has real eigenvalues.

Fix $p \in U$ and $X \in T_pM$. Choose a smooth surface curve γ with $\gamma(0) = p$ and $\dot\gamma(0) = X$. At a pocket,

$$0 \leq \langle N_p, \gamma(t) - p\rangle = \tfrac{1}{2}t^2\langle N_p, \ddot\gamma(0)\rangle + o(t^2), \tag{10}$$

$$\langle N_p, \ddot\gamma(0)\rangle = -\langle \nabla_X N, X\rangle = \langle S_p(X), X\rangle. \tag{11}$$

Dividing by t^2 and taking the limit shows that S_p is positive semidefinite. Thus both principal curvatures are nonnegative, and $\mathrm{si}_p \geq 0$. Reversing the inequalities gives $\mathrm{si}_p \leq 0$ for protrusions. □

H Search guarantees and early termination

Definition 3. Let $T = (V, E, d)$ denote a finite binary tree, where $d : V \to \mathbb{N}^+$ maps node to its depth beginning with $d(root) = 1$. Let $L \subset V$ denote all the leaf nodes and $Leaf(p), p \in V$ denote all the leaf nodes in the subtree rooted at p (including p itself if it is a leaf). Define $sc : L \to \mathbb{R}$ as a scoring function that maps leaf nodes to its corresponding 'score'. We call the tree 'well-formed' if there exists constants $\alpha > 0,\ \beta > 1$ such that, at each $p \in V$, maximum deviation dev_p, described below:

$$dev_p = \max_{q_1, q_2 \in Leaf(p)} |sc(q_1) - sc(q_2)| \tag{12}$$

is bounded by $\alpha/\beta^{d(p)}$. In other words:

$$dev_p < \alpha/\beta^{d(p)} \tag{13}$$

The condition bounds the range of oracle scores within each cluster, with a geometrically decreasing bound at greater depths. It is an assumption about the clustering tree and oracle, rather than a property guaranteed by ligand similarity alone.

Definition 4. Exploration is unrestricted if every node surviving the slack rule is expanded, without an additional beam-width cap or oracle-budget cutoff. Every visited node is scored using at least one evaluated descendant leaf.

Definition 5. At depth d, let $b(q)$ be the best observed descendant-leaf score at node q, and let sc^*_d be the minimum of these scores over the active nodes. For lower-is-better scores, the slack rule retains nodes satisfying $b(q) < sc^*_d + \delta_d$ (Algorithm 7).

Proposition 2. *Assume a tree $T = (V, E, d)$ is well-formed with constants α, β. Under unrestricted exploration, the slack assignment $\delta_d = \alpha/\beta^d$ preserves a branch containing a best-scoring leaf and finds a leaf with optimal score sc^*.*

Proof. Let q_d^* be the node at depth d on a path to an optimal leaf. Any evaluated descendant of q_d^* differs from the optimal score by less than α/β^d. Hence

$$b(q_d^*) < sc^* + \alpha/\beta^d \leq sc_d^* + \delta_d, \tag{14}$$

since every observed leaf score is at least sc^*. Therefore the optimal branch is never trimmed. Starting at the root and applying this argument at each depth, unrestricted exploration eventually reaches and evaluates an optimal leaf. □

Corollary 3. *Under the assumptions and slack assignment of Proposition 2, let $\varepsilon > 0$ and choose*

$$d_\varepsilon = \max\left\{1, \left\lceil \frac{\log(\alpha/\varepsilon)}{\log \beta} \right\rceil \right\}. \tag{15}$$

After evaluating a descendant of every surviving node through depth d_ε, or completing the search earlier, the best evaluated ligand $\hat{\ell}$ satisfies $sc(\hat{\ell}) \leq sc^* + \varepsilon$.

Proof. The optimal branch is never rejected. If its optimal leaf is reached before depth d_ε, it is evaluated directly. Otherwise, an evaluated descendant ℓ of its depth-d_ε node satisfies

$$sc(\hat{\ell}) \leq sc(\ell) < sc^* + \alpha/\beta^{d_\varepsilon} \leq sc^* + \varepsilon. \tag{16}$$

□

These statements assume the descendant-score bound and unrestricted exploration; they do not guarantee optimality under an arbitrary finite oracle budget.