PocketQuest: Predicting Ligand Binding Sites Using Connolly Points and Feature-Based Learning

Anna Korda, Montse Garcia Cerqueda

INTRODUCTION

In silico ligand binding site prediction has been a real challenge in the field of Structural Bioinformatics for a while now, and a lot of different methods have been implemented to achieve it. Traditional geometry-based approaches focused on identifying concave surface regions, while more recent developments incorporate physico-chemical properties and machine learning to boost accuracy[1].

In this project, we introduce PocketQuest, a feature-based and machine learning-enhanced framework for predicting ligand binding sites on protein structures. Our pipeline begins by extracting Connolly surface points from a given PDB structure — each point representing a location on the protein's solvent-excluded surface (SES). For each of these points, we compute a feature vector using structural and chemical descriptors. This transforms the protein surface into a rich set of observations. We then use a trained XGBoost classifier to predict the ligandability of each point. Finally, rather than relying on the model alone to define binding sites, we apply spatial clustering to the high-scoring points, allowing us to identify complete binding pockets based on cavity geometry and point density. This hybrid strategy combines the sensitivity of machine learning with the spatial awareness needed to detect biologically meaningful binding regions.

This idea was inspired by previous work that used machine learning to predict ligandability scores from solvent-accessible surface points[2]. We build upon that concept by integrating a Python-based feature extraction pipeline with a flexible clustering step to define complete binding sites after prediction. For training and evaluation, we used literature curated protein-ligand complexes from the LigandDB Database[3], which we extensively preprocessed to extract relevant binding residues and generate point-level features.

OVERVIEW OF THE MODEL

As mentioned before, our model is highly based on an already existing protein binding site prediction tool, P2Rank[4]. Both models take as an input a PDB file and generate as output a list of predicted protein binding sites, which are generated following these steps:

1. Calculate the SES of the protein and generate a series of evenly spaced points of said surface, called Connolly points.
2. Generate a feature vector for each point that includes information about:
    i. The physicochemical characteristics of all atoms that are found in a 6 Å radius, weighted by their distance between the atom and the Connolly point.
    ii. The physicochemical and binding characteristics of the residue of each atom found in a 6 Å radius, also weighted according to the distance.

iii. Other characteristics that describe the Connolly point as a whole and its atomic neighborhood (understood as the atoms found in a 6 Å radius from the point).

3. Classify the Connolly points as belonging or not to a binding site and generate a ligandability score using a previously trained XGBoost model.
4. Cluster the Connolly points according to their coordinates using single linkage clustering to generate pockets.
5. Rank resulting pockets using the cumulative ligandability score of the points.
6. Visualization: Predicted clusters are saved as separate PDB files and can be visualized using Chimera. Each file contains the cluster that makes up a predicted binding site, allowing inspection of their spatial arrangement on the protein surface.

HOW TO USE POCKETQUEST

1. Make sure you have installed all required packages.

To run PocketQuest, the packages required to install in Python before execution are: **numpy, pandas, xgboost, scikit-learn, matplotlib** and **scipy**. The script also uses standard libraries such as **argparse, os, sys, subprocess, glob, shutil, warnings, pickle, and colorsys**, which are **included by default with Python and do not require installation.** In addition to these, **UCSF Chimera** must be installed separately if the user wishes to run the visualization component.

```
pip install numpy pandas xgboost scikit-learn matplotlib scipy
```

UCSF Chimera**:** https://www.cgl.ucsf.edu/chimera/download.html

2. Clone the repository

```
git clone https://github.com/annakorda/PocketQuest
```

```
cd PocketQuest/
```

3. Install PocketQuest

```
pip install -e .
```

4. Run the Prediction Pipeline.

Run PocketQuest from any folder — the results will be saved in that same location.

```
pocketquest -i input.pdb -prob 0.9 -distance 3 -size 4 -max_residues 30 -vis 3 -rotate
```

where:

- **i:** input protein structure in PDB format.
- **prob:** minimum probability threshold to consider a Connolly point as part of a pocket.
- **distance:** clustering distance threshold (in Å).
- **size:** minimum number of Connolly points to form a valid cluster.
- **max_residues:** limits the number of residues assigned to a pocket (to reduce noise).
- **vis:** number of top-scoring clusters to visualize

- **rotate:** applies a specific rotation to the input protein during visualization. This is useful in the case of transmembrane proteins or others, that when the PDB is opened the protein structure is shown from above and not from the side. In these proteins if rotate is not applied the automated screenshot taken will not be informative.

5. Access the Output

- Predicted binding sites are saved as .pdb files (e.g., input_cluster_0.pdb) in the results/ directory.
- If visualization is enabled, these clusters will be visualized in Chimera.
- A high-resolution screenshot of the visualization is also saved in the results/ folder.

DATA PREPROCESSING & BINDING SITE EXTRACTION

To construct a reliable dataset for training our ligand binding site prediction model, we began by downloading article-curated data from **BindingDB** (https://www.bindingdb.org/rwd/bind/chemsearch/marvin/Download.jsp). Specifically, we used the file BindingDB_BindingDB_Articles.tsv, which contains experimentally validated protein–ligand complexes. Although the file provides extensive chemical and pharmacological annotations, we retained only the column listing PDB IDs of ligand–target complexes, as our objective was to derive high-confidence binding site annotations from structure alone.

Using a custom Python script, we extracted up to 4,000 unique PDB identifiers and downloaded the corresponding .pdb files from the RCSB Protein Data Bank via its public API. A total of 3,960 structures were successfully retrieved, accounting for redundancy and invalid entries in the original list.

To prepare these structures for downstream analysis, we performed several stages of structural filtering and binding site extraction. All structural processing, including ligand identification, homomer filtering, distance calculations, and binding site extraction, was carried out using the **MDAnalysis**[5] Python library, which provides flexible tools for working with atomic-level molecular data.
The steps were as follows:

- **Ligand Identification:** We parsed each PDB file to identify non-protein, non-water, non-buffer, non-metal molecules as potential ligands, and excluded those with fewer than five atoms to avoid small or irrelevant molecules.
- **Homomer Filtering:** Chains with identical amino acid sequences (homomers) were removed unless they participated in interactions with singleton ligands — ligands present only once in the structure — to preserve meaningful contacts.
- **Binding Site Annotation:** For each remaining ligand, protein residues with at least one atom within 4 Å of any ligand atom were annotated as binding site residues[6].
- **Minimum Size Filter:** Binding sites with fewer than five residues were discarded to remove weak or non-specific interactions.
- **Redundancy Elimination:** Within each PDB, binding sites associated with the same ligand were merged if they shared at least 70% of their residues or had centroids within 4 Å.
- **Ligand Prioritization:** To reduce noise from small or trivial ligands, only the top two ligands with the highest atom counts were retained per structure.

This processing pipeline resulted in a high-quality filtered dataset of 3,140 PDB structures, each associated with one or more cleaned binding sites. The final output was stored in a CSV file (binding_sites.csv) containing PDB ID, ligand name, ligand residue ID, a semicolon-separated list of binding site residues, and the number of residues. The average size of a binding site was 12.89 residues, with values ranging from 5 to 39.

To ensure biological relevance and enable balanced training, we further analyzed the dataset using protein function information. First, we mapped PDB IDs to UniProt IDs, which were then submitted to the Panther Classification System. Of the 3,140 structures, 1,518 had clear Panther annotations and were retained.

Next, we manually curated the remaining dataset to remove low-quality or biologically ambiguous structures. For this, we used Python to create Chimera scripts (.cmd) that displayed each protein in default color, its ligands in light blue, and the annotated binding sites in magenta. A total of 1,518 screenshots were generated and manually reviewed. Structures were excluded if they were poorly resolved, excessively large, missing key regions, or contained multiple unrelated ligands. After this final curation step, 1,082 PDB files remained.

To enable stratified training and evaluation, the curated dataset was split into 10 balanced batches based on Panther protein class distribution. This stratification ensured functional diversity across batches and was implemented using Scikit-learn's stratified model selection tools. Fig.1 shows the distribution of Panther protein classes across the full dataset of 1,082 curated PDBs. This view highlights the diversity of protein functions retained after filtering and Fig. 2 displays the balanced stratification of protein classes across the 10 batches, confirming that no class is over- or under-represented within a single batch.

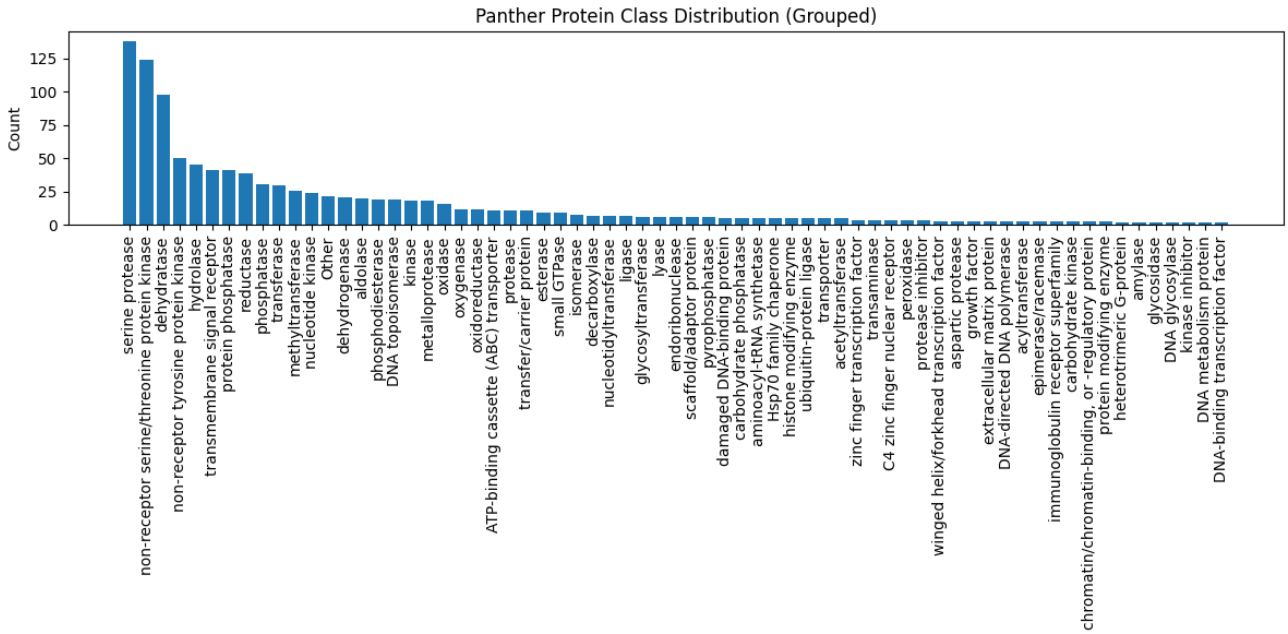

Figure 1. Panther protein class distribution across the 1,082 high-quality PDB structures after filtering and manual curation.

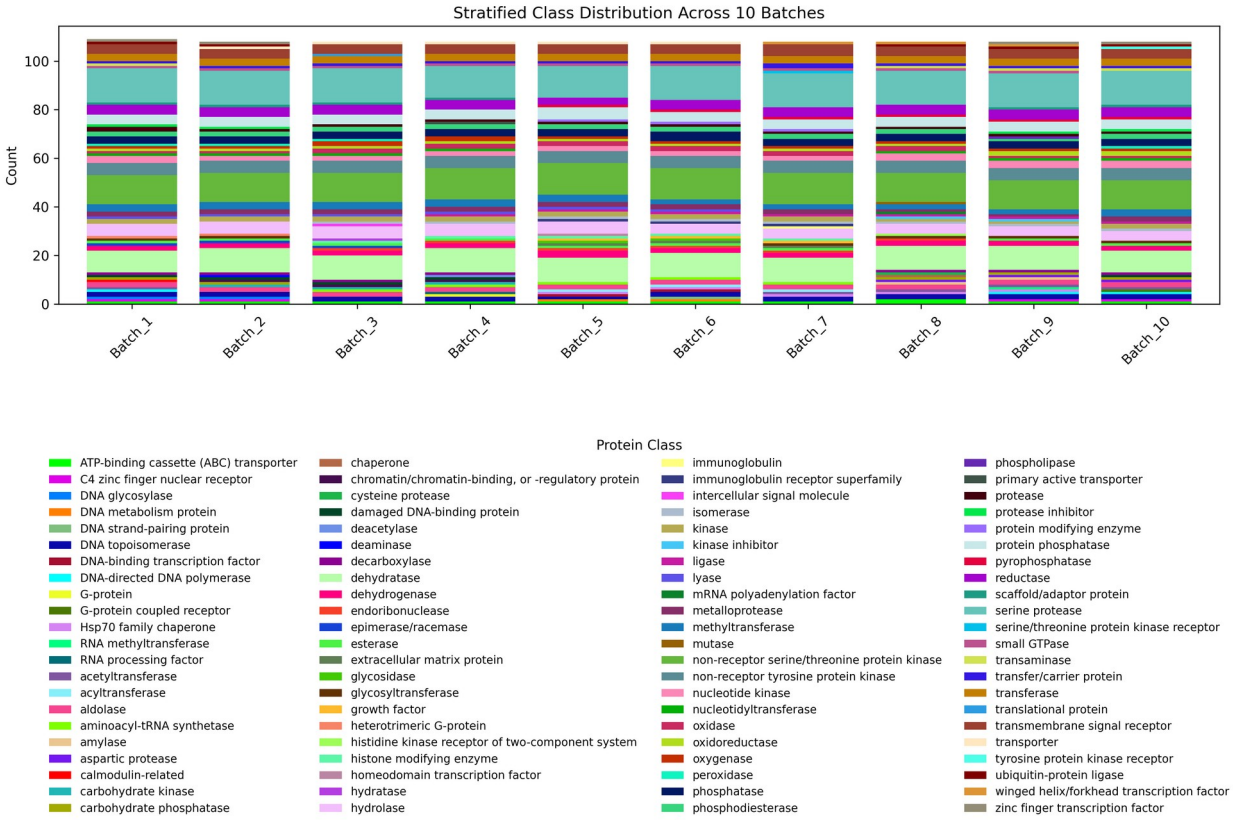

Figure 2. Class balance across the 10 stratified batches used for training and evaluation

Each batch folder contains two types of files per structure:

- The cleaned full PDB file.
- One or more .pdb files containing only the binding site residues.

The final step involved extracting the annotated binding site residues into individual .pdb files from the CSV file. This was done using an automated script that reloaded the filtered structures, removed homomeric chains again, and extracted each set of binding site residues based on their resid values. These files will serve as positive labels during Connolly surface point classification in model training.

SURFACE EXCLUDED SOLVENT & CONNOLLY POINTS

As explained in the overview, the first step of our model is to calculate the **solvent excluded surface** (SES) of the protein and generate a series of evenly spaced points found on said surface.

The concept of the solvent excluded surface derives from the **solvent accessible surface** (SAS) a method to calculate the molecular surface of proteins[7]. This method works by rolling a sphere of radius r_p, called the probe, over the Van der Waals surface of the protein. As a note, said surface is calculated by taking the Van der Waals radii of each atom and overlapping the resulting circles, which can be seen in Figure 3.

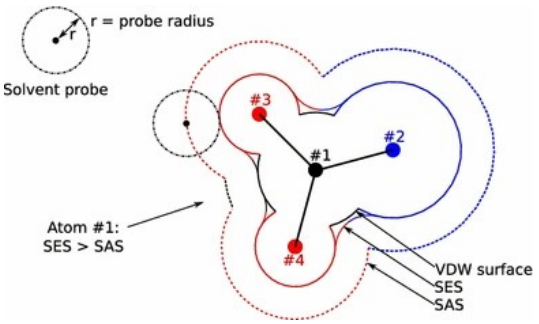

Figure 3. Representation of the Van der Waal's surface (VDW), solvent accessible surface (SAS) and solvent excluded surface (SES) in a toy example. Adapted from Santos-Martins D. *et al* (2016)

While "rolling" the probe, the **accessible surface area** of each atom, described as the "*area on the surface of a sphere of radius R (where R is the sum of the van der Waal's radius of the atom and the chosen radius of the solvent molecule), on each point of which the center of a solvent molecule can be placed in contact with this atom without penetrating any other atoms of the molecule*", was also calculated. Finally, the accessible surface area of each atom was united to generate the SAS of the protein, which is represented in Figure 3. As we can observe and in a very simple way, the SAS "represents" the path the center of the probe makes when it is being rolled across the surface. Consequently, it seems like the area is "separated" from the Van der Waal's surface roughly by the value of r_p.

On the other hand, the SES[8] is also computed by rolling a probe of radius r_p but represents the "*topological boundary of the union of all possible probes that do not overlap with the molecule*". In a very simple way, it is defined by the surface of the protein that the probe is in contact with and represents the boundary that excludes the solvent from interacting with the protein. As seen in Figure 3, the SES is much closer to the Van der Waal's surface with the exception of some cavities that it can't access.

While the original article used the SAS we ended up using the SES, mostly due to the inability of finding a program that could compute the SAS and at the same time be integrated with our python scripts. Additionally, we believed that the model wouldn't be significantly affected by this change.

The SES and Connolly points were computed using the program MSMS[9] with a probe radii of 1.6 and a density of 0.5.

FEATURE ENGINEERING

To be able to train an XGBoost model capable of differentiating between Connolly points belonging or not to a binding site, we had to first design a Connolly point feature vector (CFV) composed of attributes that we considered were relevant to make this classification.

The final CFV consisted of 32 features, shown in Table 1, spread across three "groups" or elements of interest: atoms (7 attributes), amino acids or residues (17 attributes) and the point neighborhood (8 attributes). It is important to notice that we only considered the 20 natural amino acids in our classification.

Table 1. Description of the attributes of the feature vector.

Name	Element of Interest	Description
Hydrophobicity[10]	Atom	Binary attribute, 1 for atoms that are hydrophilic
Valid Ligand Propensity[11]	Atom	Average number of contacts with valid ligands.
Invalid Ligand Propensity[11]	Atom	Average number of contacts with invalid ligands.
Aromatic	Atom	Binary attribute, 1 for atoms that are part of an aromatic ring.
Acceptor	Atom	Number of hydrogen bonds it can accept.
Donor	Atom	Number of hydrogen bonds it can "donate".
B-factor	Atom	B-factor from the PDB file.
Hydrophobic	Amino acid	Binary attribute, 1 for hydrophobic residues.
Hydrophilic	Amino acid	Binary attribute, 1 for hydrophilic residues.
Hydropathy Index[12]	Amino acid	Side-chain hydropathy index.
Aliphatic	Amino acid	Binary attribute, 1 for aliphatic residues.

Aromatic	Amino acid	Binary attribute, 1 for aromatic residues.
Sulfur	Amino acid	Binary attribute, 1 for residues containing sulfur atoms.
Hydroxyl	Amino acid	Binary attribute, 1 for residues containing hydroxyl groups.
Basic	Amino acid	Binary attribute, 1 for basic residues.
Acidic	Amino acid	Binary attribute, 1 for acidic residues.
Amide	Amino acid	Binary attribute, 1 for residues containing amide groups.
Positive Charge	Amino acid	Binary attribute, 1 for positively charged residues.
Negative Charge	Amino acid	Binary attribute, 1 for negatively charged residues.
Donor	Amino acid	Binary attribute, 1 for residues containing hydrogen bond donor atoms.
Acceptor	Amino acid	Binary attribute, 1 for residues containing hydrogen bond acceptor atoms.
Donor & Acceptor	Amino acid	Binary attribute, 1 for residues containing both hydrogen bond donor and acceptor atoms.
Polar	Amino acid	Binary attribute, 1 for polar residues.
Ionizable	Amino acid	Binary attribute, 1 for ionizable residues.
Number of Atoms	Point	Absolute number of atoms in a 6 Å radius from the point.
Atom Density	Point	Number of atoms in a 6 Å radius from the point weighted by the distance.
Number of Carbons	Point	Number of carbon atoms in the neighborhood.
Number of Oxygens	Point	Number of oxygen atoms in the neighborhood.
Number of Nitrogens	Point	Number of nitrogen atoms in the neighborhood.
Number of Donors	Point	Number of hydrogen bond donor atoms in the neighborhood.
Number of Acceptors	Point	Number of hydrogen bond acceptor atoms in the neighborhood.
Protrusion[13]	Point	Number of all atoms within a 10 Å radius from the point.

To generate the CFV, the program first calculates an atomic feature vector (AFV) for each atom in the atomic neighborhood, meaning in a 6 Å radius from the point. This AFV contains the atom and amino acid related features. Afterwards, each AFV is weighted according to a function of the Euclidean distance between the atom and the point described as:

$$w(d) = 1 - d/6$$

By doing this, atoms that are closer to the point, thus having a small distance, will have a higher weight and a bigger influence in the final feature vector. On the other hand, those atoms that are further away will have higher distances and lower weights, thus having a smaller effect on the classification.

Finally, all the weighted AFVs are added together and the point-related features (PFV) are concatenated, generating the final CFV. We could mathematically describe this process as:

$$CFV(P) = \sum_{Ai \in A(P)} AFV(Ai) \cdot w(d(P, Ai)) \| PFV(P)$$

ATOM FEATURES

As we have mentioned we defined 7 atom-related features. The values for each atom can be found on the Supplementary Material, Table 1. Among them we would like to delve more in depth into four of them that we find particularly interesting: hydrophobicity, valid ligand propensity, invalid ligand propensity and B-factor.

Firstly, it has long been described that **hydrophobicity** is a major driving force in the folding and stability of a protein's tridimensional structure[10,14]. That is because in an aqueous polar environment, like the cytoplasm of a cell, many different non-covalent interactions occur between the solvent and the protein, such as electrostatic interactions, Van der Waals forces and especially **hydrogen bonds**, which are involved in the folding and stabilization of a protein. However, not all molecules are capable of forming said hydrogen bonds, such as hydrophobic residues, which generates **unfavorable interactions** between water and them. These unfavorable interactions cause the protein to fold in a way where **hydrophobic residues pack together**, forming the core of the protein, consequently r**educing the contacts with water**, and the hydrophilic residues are left on the surface of the macromolecule[14]. This is called the **hydrophobic effect**. Taking this into account, it is particularly interesting to consider the overall hydrophobicity of our points, as we would expect points with more hydrophobic elements to be less likely to be on a protein binding site.

However, classifying residues according to hydrophobicity can be tricky, especially in those that have **amphipathic** character. One explanation could be that residues are heterogeneous at the atomic level, as they can contain both polar and nonpolar residues, but that this heterogeneity is usually not taken into account as residues are "forced" into one or the other category[10]. Consequently, we found it particularly important to consider the **hydrophobicity at the atomic level** which could more accurately describe the overall hydrophilic or hydrophobic character of each Connolly point.

Following the original article, we decided to use a classification where atoms were separated using "the physically based partial charges assigned in the OPLS (optimized potentials for liquid simulations) modeling parameter set"[10]. Afterwards, atoms that had a partial charge magnitude from 0 to 0.25 were considered nonpolar or hydrophobic and given the value -1, and atoms with a particle charge magnitude higher than 0.25 were considered hydrophilic or polar and given the value 1.

As shown in Figure 4, the backbone of all amino acids has similar properties, with the carbon α and β being hydrophobic, and with the carbon and oxygen of the carboxyl group as well as the nitrogen of the amine group being hydrophilic. Regarding the side chains, there is a lot of variability, with some amino acids, such as phenylalanine or leucine, being completely hydrophobic and some others, like glutamate or serine, having some hydrophobic atoms.

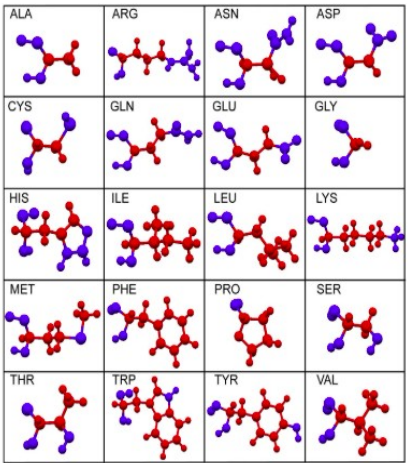

Figure 4. The atoms of the 20 natural amino acids are colored according to their atomic binary hydrophobicity value. Hydrophobic atoms are colored red, and hydrophilic atoms are colored blue. Adapted from Kapcha, L. H., and Rossky, P. J (2014).

We also decided to include two attributes, **valid** and **invalid ligand propensity**[11], a measure that aims to characterize the atom and amino acid composition of protein binding sites while differentiating between "valid", meaning biologically relevant, and "invalid", meaning biologically irrelevant, ligands.

Among the different parameters reported on the study, we focused on the so-called "raw" contacts between ligands and binding sites. The authors defined as a contact any amino acid with at least one heavy atom that was within 4.0 Å of a ligand's heavy atom. Additionally, those contacts that had distances below 3.5 Å were considered to be hydrogen bonds and the rest, with distances between 3.5 Å and 4 Å, were considered to be Van der Waals interactions. In the article the authors mostly reported contacts at the amino acid level, however, we were able to obtain the contacts per atom from their Supplementary Material, Dataset S2. For each atom, its average number of contacts was obtained by dividing the total number of contacts, both hydrogen bonds and Van der Waals interactions, by the times its amino acid was found in a protein binding site. While the average number of hydrogen bonds and Van der Waals interactions was also reported, we decided to directly use the total number of contacts, as we believe both types of interactions are important in the binding with the ligand. The results are shown in Figure 5.

We can see that in general the **higher number of contacts** happen in the **side chain** of the amino acids, and that the backbone residues tend to have values lower than 1, meaning that when the residue is found in a binding site these atoms usually don't make contact with the ligand. The only exception is glycine. Additionally we can observe that some amino acids have atoms with an overall low average number of contacts, like methionine or proline, while other amino acids, such as lysine or arginine, have atoms with a high number of contacts, between 2 and 2.5, meaning that every time this residue is found in a protein binding site, these atoms are in contact with two or more atoms from the ligand.

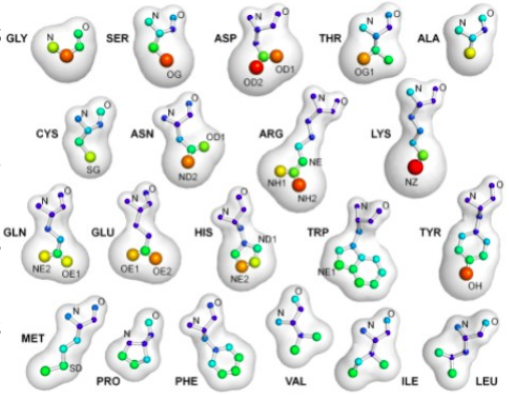

Figure 5. The atoms of the 20 natural amino acids are coloured by the average number of "raw" contacts. The hotter colors indicate more contacts per atom: deep blue ≤ 0.30, cyan = 0.70, green = 1.00, yellow = 1.55, orange = 2.00, and red ≥ 2.30. Adapted from Khazanov, N. A., & Carlson, H. A (2013).

Clearly, this information can help us better characterize the Connolly points, as we would expect that those points that have atoms that on average make more contacts with ligands would have a higher probability of being part of a protein binding site.

Finally we also included the **B-factor,** also known as Debye-Waller factor or temperature factor, which can be obtained from the PDB files. This term was described for the first time in the context of the study "*of X-ray scattering and thermal motion in solid materials*". At its core, the B-factor represents the **displacement of an atom from an average mean value**, and can be calculated following this formula:

$$Bfactor = 8\pi^2 \langle u^2 \rangle$$

where u is the mean displacement of a scattering center[15].

In the context of protein structure, the B-factor has been linked to the **flexibility** of the atoms and residues, with higher B-factors implying higher flexibility and vice versa. One study showed that regions with high B-factor values had "*higher average flexibility index, higher average hydrophilicity, higher average absolute net charge, and higher total charge*" than regions with lower values[16].

Additionally, many studies have been conducted to determine if differences in the B-factors are related in some way to the functional activity of proteins. One such study showed that the **active site** of the 69 studied apo-proteins was found on regions that had on average **low B-factor value**, while non-active sites were found in regions with higher B-factors[17].

These results might be surprising, as the current understanding of enzyme function is based on the induced fit model, which requires the protein to undergo conformational changes once the substrate is bound to it, for the enzyme to be able to catalyze the reaction. Under this model, one would expect the catalytic site to be found in more flexible or mobile regions that would allow said conformational change.

However, the authors propose that one should differentiate between the concept of flexibility, "*moving in a broad energy minimum with no barriers*", with plasticity, "*populating a number of closely spaced, discrete conformational energy wells*". In this sense, atoms that are flexible are able to move or vibrate within an energy well while atoms that are plastic can "jump" between clearly differentiated discrete energy states. It is this capacity that allows the catalytic activity and not so much by the "vibrational" motions represented by the B-factor.

In conclusion, we believe that the B-factor reflects a series of characteristics that are relevant for differentiating and characterizing binding and non-binding sites, and thus it is interesting to add it to our model.

It is worth mentioning that because the B-factor is obtained through experimental methods, its value is **affected by other factors related to the experimental procedures**, such as the resolution of the structure, the crystal contacts or particular refinement procedures that have been carried on, which makes it difficult to "compare" B-values between proteins. Consequently, B-values tend to be **normalized** using the following expression:

$$B_{norm} = (B - \langle B \rangle)/\sigma$$

where 〈B〉 is the average of the B-values of a given structure and σ is the standard deviation[18]. However due to computational and time constraints we decided to not normalize the B-factors and directly use the B-factors reported in the PDBs. We recognize that this is not ideal and might affect the prediction ability as well as the accuracy in finding binding sites and it is something that could be optimized in the future.

AMINO ACID FEATURES

Amino acids are the "group" or element with the highest number of attributes, 17 in total, which are mostly related to their **physicochemical characteristics**. The values for each amino acid can be found on the Supplementary Material, Table 2.

As we have mentioned before, it is sometimes hard to classify amino acids, for example in terms of hydrophobicity and hydrophilicity. In this case, we decided to follow the classification proposed in Lehninger principles of biochemistry, as it is a well known reference book.

The only exception is the hydropathy index[12]. In the cited study, the authors assigned **hydropathy values** to the 20 natural amino acids based on their water-vapor transfer free energy as well as the fraction of the residues that are more than 95% or 100% buried in a protein's native structure, results that had been published before. After some subjective adjustments, which are discussed in the original article, they reported the final hydropathy index, whose values span from 4.5, indicating high hydrophobicity, to -4.5, indicating high hydrophilicity.

POINT FEATURES

Finally, we defined 8 point features, which aim to describe the **characteristics** of each point's **atom neighborhood** in terms of the number of atoms that might be important in establishing contacts with the ligand, such as the number of donors and acceptors.

One interesting attribute is the **protrusion**, based in a measure, the **Cx index**[13], capable of identifying protruding regions of a protein, which can be part of binding sites, protein-protein interaction sites or proteolysis cleavage regions. The Cx index is calculate in the following way.

For each heavy atom in a protein the program takes the number of heavy atoms within a set distance R (10 Å by default, which we maintained) and calculates the volume they occupied by multiplying the number of atoms in the sphere by the mean atomic volume. This value is called **Vint** and represents the volume within the 10 Å sphere occupied by the protein. The remaining value, **Vext**, is then calculated as the difference between the total volume of the sphere and Vint. Finally, the Cx index is defined as **Vext/Vint**.

Consequently, those atoms that have few atoms around them will have high Cx values (because Vext will be high and Vint low). As having few neighbors is characteristic of protrusions, because they are convex parts of the protein, the authors argued that the Cx index could be considered an approximate measure of the protrusion.

Following this concept and the idea presented in the original article, we defined the protrusion as the number of atoms that are found in a 10 Å radius. While we are not considering the ratio between volumes, we believe that the fundamental logic of the Cx index is maintained, as we would expect points that are found in a protrusion to have less number of atoms in a 10 Å radius than points that are not protruding.

MACHINE LEARNING FEATURES

A final attribute was calculated only for the **training** and **testing** of the **machine learning model**, which represented whether or not the point **belonged** to a **protein binding site**. This attribute was calculated by dividing the number of atoms in the point that were also part of the protein's binding site between the total number of atoms in the point. If the proportion was higher than a certain threshold, which we set to 0.3, the point was considered to be in a protein binding site.

We acknowledge that this threshold is relatively low, as we are accepting as protein binding sites Connolly points that may have half of the atoms or less truly being in one. However, as the average number of atoms per binding site tends to be low, we worried that there wouldn't be enough points that reached a higher frequency and that would lead to losing information, which could complicate the training of the machine learning model.

Finally, we believe that in future studies, it would be interesting to perform the training using a range of thresholds, to assess how this attribute affects the final machine learning model.

MODEL TRAINING & EVALUATION

Following the preprocessing and feature extraction pipeline, the next stage of our project focused on training a machine learning model capable of accurately distinguishing ligand binding sites from non-functional surface areas.

Starting from a curated collection of protein–ligand complexes in PDB format, we had already filtered and retained biologically relevant examples (1081 PDBs). These final PDBs were converted into Connolly surface points, and each point was described using a vector of 32 atom, residue and point features. A binary label was added to indicate whether the point was part of a binding site.

To handle the process efficiently, we **initially divided the dataset of filtered PDB files into 10 balanced batches**. These batches were created to contain roughly equal numbers of structures and were **stratified based on PANTHER[19] protein classification**, ensuring that different protein families and functional types were evenly distributed across the dataset. Once the batches were defined, we generated **Connolly surface points** for each PDB file within them and computed **feature vectors** for each point, resulting in `.npy` files containing approximately 1 million points per batch, each represented by **32 features and 1 label**. This batching strategy enabled efficient memory usage, streamlined feature engineering, and allowed us to debug and monitor each batch independently.

However, for training the machine learning model, it was important to **merge all batches into a single matrix**. This ensured that the model had access to the **full diversity of proteins and structural patterns** during training. It also avoided introducing artificial data distribution shifts that could occur when training only on isolated subsets.

We selected **XGBoost[20]** as the core model for several reasons. First, it is a highly efficient and scalable implementation of gradient boosting, especially well-suited for high-dimensional datasets. In comparison to P2Rank[4], which used Random Forest, XGBoost offered additional benefits including native support for GPU acceleration, regularization mechanisms to prevent overfitting, better control over decision thresholds and better performance on imbalanced datasets[21]. These capabilities aligned perfectly with the demands of our dataset, which included over 11 million Connolly points across all batches, and only about **5% of them labeled as positives (binding site points)**.

This **class imbalance** was a central challenge throughout the modeling process. We decided **not to use synthetic oversampling techniques** to artificially generate positive points. While such methods can be helpful in some tasks, they are **risky in structural biology[22]** because the 3D spatial organization and physicochemical context of surface points is highly specific and difficult to reproduce artificially. Instead, we relied on XGBoost's **scale_pos_weight parameter**, which increases the loss penalization for misclassified positive examples. This parameter was included in **Optuna hyperparameter tuning[23]**, allowing the model to find the best weight during optimization. By letting the optimizer choose the best balance, we ensured that this imbalance was handled both statistically and adaptively.

The training process followed a classical split: merged data was divided into training (70%) and validation (30%) sets, stratified by the binary labels to maintain the overall positive/negative ratio. We used **Optuna** to search for the best set of hyperparameters, optimizing for AUC (area under the ROC curve) as the primary metric. AUC (Area Under the Curve) is particularly suitable in our case as it remains **robust to class imbalance**, providing a reliable measure of the model's ability to discriminate between classes, regardless of their proportions.

Hyperparameter optimization was conducted using 100 Optuna trials, where each trial evaluated a different configuration of XGBoost parameters. To ensure robust evaluation, each trial used 4-fold stratified cross-validation, maintaining the proportion of positive (binding site) and negative (non-binding site) samples across all folds. The average AUC across these four folds was used as the objective function to guide Optuna's optimization process.

This tuning process was performed on a training set of 7,844,183 Connolly surface points, of which 426,817 (5.44%) were labeled as positives. The full dataset comprised 11,205,983 points, with the remaining 3,361,800 samples set aside as a held-out test set, containing 182,921 positives (5.44%). These consistent ratios ensured that the model was evaluated under realistic conditions reflecting the strong class imbalance typical in ligand binding site prediction.

The following hyperparameters were tuned:

- **max_depth:** Maximum tree depth, which controls model complexity.
- **learning_rate:** Step size shrinkage used to prevent overfitting.
- **n_estimators:** Number of boosting rounds.
- **subsample:** Fraction of training data used per tree.
- **colsample_bytree:** Fraction of features sampled per tree.
- **gamma:** Minimum loss reduction required to make a split.
- **min_child_weight:** Minimum number of data points in a leaf node.
- **reg_lambda (L2 regularization) and reg_alpha (L1 regularization):** Used to penalize model complexity.
- **scale_pos_weight:** Crucial for addressing class imbalance; values were explored in the range 5.0 to 25.0 to upweight the minority class loss.

Training and prediction were accelerated using GPU support ('tree_method': 'hist', 'device': 'cuda') enabling the testing of computationally intensive configurations across large datasets in a practical timeframe.

Each fold trained a model using XGBClassifier from the xgboost library and evaluated it using roc_auc_score on the validation split. To conserve memory, model objects were deleted and garbage collected after each fold.

In total, the Optuna study conducted 400 individual model fits (100 trials × 4 folds), recording per-fold AUCs as user attributes for further analysis. The final model was selected based on the trial with the highest average cross-validated AUC, and its best hyperparameters were used to retrain on the full training set before making predictions on the unseen test set.

The final evaluation was performed on a large, diverse test set consisting of over 3.3 million Connolly points, and the results reflect the real-world difficulty of the task:

- **AUC**: 0.8133
- **Accuracy**: 0.8309
- **Precision**: 0.1819
- **Recall**: 0.6025
- **F1 Score**: 0.2795

These metrics illustrate the classic trade-off between **precision and recall**. In our case, we made a deliberate choice to **favor recall** over precision. The rationale was clear: it is far more damaging to miss true binding sites (false negatives) than to overpredict and include some non-binding surface points (false positives). Our downstream plan involves **clustering the high-confidence predicted positive Connolly points based on spatial proximity (Angstrom distance)** to reconstruct actual

binding pockets. This clustering step will act as a filter: many scattered or structurally implausible false positives will be naturally removed if they do not form tight, spatially coherent groups.

In other words, while our model is more on the side of overpredicting positives, it focuses on **capturing more potential binding sites** and letting a post-processing step eliminate noise. This strategy aligns well with the biological objective of the project: to generate a sensitive predictor that can later be refined spatially.

We also experimented with **active learning**, since we had access to a large number of additional PDB complexes that could not all be included in the initial training set due to computational constraints. The idea was to iteratively train the model on data it was most uncertain about, gradually improving its ability to handle ambiguous or **edge-case Connolly points**. We ran a total of **40 rounds of active learning**, saving each model to evaluate performance trends over time.

However, this strategy did not outperform the original model. On a new, highly diverse test set, all active learning models showed decreased recall, with many falling to around 0.30, and AUC dropping to approximately 0.70. Interestingly, accuracy increased to 0.899, indicating that the models were becoming more conservative in their predictions. However, this came at a significant cost: precision decreased even further, and the ability to recover true binding sites was notably impaired. The models would only predict a point as positive if they were extremely confident, resulting in many missed true positives.

This behavior highlighted a core issue: in tasks like binding site prediction, sensitivity (recall) is often more important than precision, and aggressive filtering during training can suppress meaningful biological signals. These results ultimately reaffirmed our earlier decision: **it is preferable to tolerate false positives and eliminate them later using spatial clustering, rather than risk missing true functional regions entirely.**

The initial best-scoring model, which offered the best balance between recall and precision, was saved in both `.json` and `.pkl` formats so it can be integrated later into the final standalone prediction program.

CLUSTERING & EXTRACTING BINDING SITES FROM RESULTS

Going back to our PocketQuest tool, after generating the feature vector, this trained XGBoost model allowed us to calculate the **probabilities of belonging to a protein binding site** for each point. Afterwards, those points that had a probability lower than a certain threshold were removed.

By default, we chose a probability of 0.8. Considering the performance of our model, which was effective at identifying true binding site points but also prone to false positives , we believed using a more “strict” cutoff would help reduce the number of incorrectly classified Connolly points. However, the user can choose the threshold they prefer using the argument `-prob` when calling the program.

The remaining points were then clustered according to their coordinates, as our objective was to group those points that were close together and thus could physically form a binding site. We used the **euclidean distance** to compute the distance matrix and decided to perform a **single linkage clustering**, which considers the distance between two cluster to be the minimum one between their members. The generated dendrogram was finally separated into clusters such that the observations in each cluster had a **cophenatic distance** lower than a threshold, which by default was set to 3 following the original article. The user can change this parameter using the argument `-distance`.

Of the resulting clusters, we decided to filter out those that had **characteristics incompatible with being a protein binding site**. On one hand, we discarded those clusters that had less than a certain amount of points, by default 3, as we believed the region they defined was too small to constitute a binding site and at most was defining a small pocket, thus being not relevant. The cutoff can be modified using the parameter `-size`.

On the other hand, through the binding site extraction process and reviewing the literature, we knew that most protein binding sites have between 10 and 35 residues, with and average of about 13. Consequently, we decided to discard any cluster that had a number of unique residues higher than a threshold, which we set to 30 residues as we considered that clusters with a higher amount of residues most likely weren't defining a binding site but instead were an artifact due to problems while assessing the Connolly points. Again, the threshold can be changed by the user with the parameter `-max_residues`.

Finally the remaining clusters were sorted according to their **score**. Said score was calculated by summing the squared probabilities of being in a binding site for all the points of the cluster, a method described in the original article.

This method has its merits, for example, by squaring the probabilities we are increasing the importance of those points with probabilities close to 1, which indicates a high degree of certainty, as they will vary less when squared. However, it also tends to rank the clusters according to the size, as more points will inevitably lead to higher score even if the individual scores are lower than in other clusters. Consequently we discussed that it would be interesting to explore other characteristics that can describe a binding site, such as the proportion of hydrophilic or hydrogen bond donor/acceptor residues, which we would use to supplement the scoring function.

Additionally, a functionality we would like to introduce in the future is to be able to submit some residues that might be in the binding site, for example because they appear in the binding site of a protein of the same family, and consider in the scoring function if said residues are found on the cluster.

As an output, by default the program generates a **`result.log` file**, which describes the found clusters with their scoring functions and **PDB files** for each cluster. Additionally, using the argument `-vis`, the top performing clusters are visualized in UCSF Chimera superposed to the original protein.

VISUALIZATION

To facilitate the interpretation of our predictions and provide a user-friendly output, **we developed an automatic visualization module integrated into our workflow**. This module is activated via the -vis argument, allowing users to visualize the top n predicted binding sites on the protein structure using **UCSF Chimera.**

Each binding site is represented on the protein surface with a unique RGB color from a custom palette. This color palette is generated using HSV-to-RGB transformations, ensuring that **each predicted site is visually distinguishable from the others.** The visualization employs Chimera's surface representation, which offers a clearer depiction of the spatial region involved in binding compared to other representations such as sticks or ribbons. To maintain clarity, we apply a uniform 50% transparency to all rendered surfaces. The background and general appearance are configured using Chimera's publication preset, resulting in a clean and professional output suitable for figures or presentations.

Users can optionally specify the --rotate flag for proteins known to be transmembrane or otherwise requiring reorientation. When enabled, the script applies a standard rotation (90 degrees around the x-axis to reorient the structure, ensuring better visibility of membrane-embedded proteins or others that by default are shown from above.

Behind the scenes, the visualization.py script, located in the utils/ directory, creates a **custom Chimera command script** based on the selected parameters. This script includes commands for opening the structure, coloring each cluster, applying surface rendering, and optionally rotating the structure. It is automatically saved in the results/ directory under the name visualize_{pdb_id}.cmd, providing full reproducibility and transparency of the visualization process.

Once the visualization is rendered, the program generates and saves a high-resolution screenshot ({pdb_id}_clusters.png) in the same directory. Chimera remains open at the end of the process, allowing the user to interact further with the visualized structure if desired.

This visualization functionality provides a clear and informative representation of the predicted binding sites, while also allowing users the flexibility to explore the structures further in Chimera if desired.

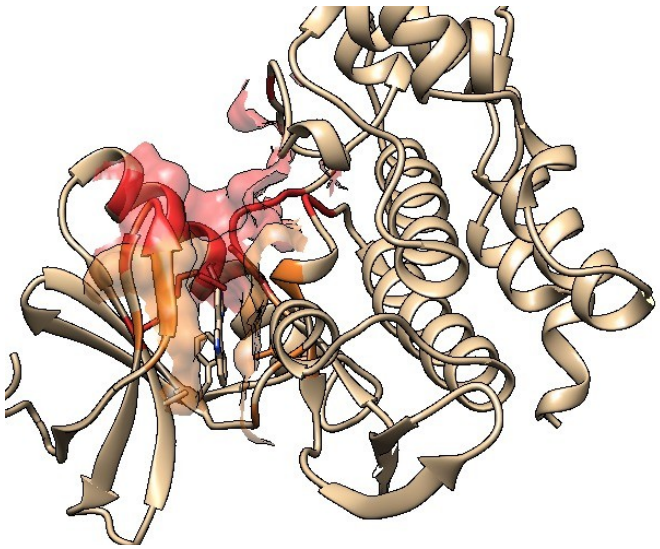

Figure 5. Predicted Binding Site Clusters in Chimera. This visualization shows a protein structure with predicted binding site clusters overlaid in color. Each colored patch represents a distinct high-probability binding cluster identified by the model. The surface rendering and coloring are automatically generated by the visualization.py script, using Chimera command files. This allows users to visually inspect and interact with the predicted binding regions in 3D.

POCKETQUEST ASSESSMENT

This evaluation was performed using a **new set of test PDBs that were originally curated for an active learning pipeline**, which ultimately was not implemented. As such, the analysis provides **a fresh and independent measure of how PocketQuest performs on completely unseen, diverse protein-ligand complexes.**

Using a classification threshold of **0.8** for Connolly point predictions, followed by clustering and filtering, we evaluated **1,498 labeled binding sites across 993 PDB structures.**

Highlights from the Evaluation include:

- **543 binding sites (~36%) achieved ≥ 50% residue-level recovery,** meaning a substantial part of the annotated binding site was successfully predicted.

- **460 PDBs contained at least one accurate cluster (among top 10), confirming that the model can reliably detect at least one correct binding site per structure in nearly half the test set.**
- **The mean overlap between predicted and true residues for the best cluster per site was 38.11%,** which is a solid result for such a challenging task.
- **Some predictions were spot-on: the model achieved 100% residue recovery for some binding sites of structures like 3TV8, 1XOR, 1NU4, 6NX8, and 1GCZ.**

Despite the absence of any fine-tuning or active retraining on these samples, **PocketQuest generalizes reasonably well.** It correctly identifies some binding regions in structurally and functionally diverse proteins, even when applied to new PDBs it had never seen during training.

However, **several limitations remain**. First, **false positives are a natural trade-off** of the chosen strategy. Even with a strict probability threshold of 0.8, the model tends to predict a high number of clusters, which helps recall but reduces precision, a pattern already discussed earlier in this report. Additionally, **the diversity of the training dataset was good but also a bit limited**, with some functional classes being represented by only a single protein. This makes it difficult for the model to generalize well across different protein families and binding chemistry.

The current features used for training are mostly basic, chemical, or local residue properties. **More abstract, higher-level, or learned features, especially ones that describe cluster-level geometry, could help reduce noise and improve the distinction between true binding pockets and irrelevant regions.** Another important issue is the **potential incompleteness of ligand annotations in the PDB structures. Some predicted clusters that appear to be false positives may actually correspond to real binding sites**, but those ligands are missing from the file and thus the region was labeled incorrectly during training.

Finally, **the clustering itself is performed using static parameters, a fixed distance threshold and a minimum size for forming a cluster. While this works in general, proteins vary significantly in size and surface topology.** Using dynamic or adaptive clustering parameters based on the protein's geometry could improve the grouping of predicted points and avoid both over-merging and under-splitting of meaningful sites.

More detailed evaluation metrics, including per-cluster overlap scores and individual protein statistics, are available in the assessment log file provided in the GitHub repository.

TOOLS & LIBRARIES

The PocketQuest pipeline was developed using **Python 3.9+**, leveraging a combination of scientific computing, structural bioinformatics, and machine learning tools. Some parts of the workflow were executed in **Jupyter Notebooks**, which facilitated **interactive development, iterative filtering, and modular experimentation** throughout the project. For small-scale data handling and file operations, we also made use of **Bash scripting**, particularly to streamline tasks such as file renaming and format conversions.

To process and analyze protein structures, we used **MDAnalysis**, which enabled efficient parsing of PDB files, ligand filtering, distance calculations, and spatial neighborhood detection. For sequence alignment and residue format conversion, we incorporated **Biopython**, specifically the **pairwise2** module and amino acid utilities from **SeqUtils**.

Structural visualization and manual curation were supported through **UCSF Chimera**. We generated .cmd scripts in Python to automatically visualize each protein-ligand complex, highlighting ligands and annotated binding sites in different colors. This step was crucial during the manual review phase to ensure biological relevance and structural quality.

Data manipulation was carried out using **NumPy** and **Pandas**, which handled large-scale feature arrays and batch processing. Filesystem operations were managed with built-in Python libraries such as **pathlib, os, and glob**, while **gc, re, datetime, and pickle** were used for memory management, regular expression handling, timestamping, and object serialization. Collections such as **Counter** and **defaultdict** also played a role in label tracking and batch structuring. The library **scipy** was used to perform the clustering.

For the machine learning component, we trained a **XGBoost classifier**, selected for its speed, scalability, and performance on structured, imbalanced datasets. Training was accelerated using **GPU support (tree_method='hist', device='cuda')** on an **NVIDIA GeForce RTX 3050**, which allowed us to process millions of Connolly points efficiently. Model parameters were optimized using **Optuna**, which performed 4-fold stratified cross-validation and applied early stopping through pruning callbacks. Trained models were saved using **joblib** for later integration into a standalone application.

Model evaluation relied on functions from **Scikit-learn**, including metrics such as AUC, accuracy, precision, recall, F1-score, and precision-recall curves. Stratified data splits and folds were managed with **train_test_split** and **StratifiedKFold**.

To generate plots and diagnostic figures, we used **Matplotlib**, while **Distinctipy** was employed for creating visually distinct color palettes in figures such as batch distributions. Long-running loops in active learning exploration were enhanced with **TQDM** for progress visualization.

Finally, for downloading external data, including PDB structures and annotations, we used the **Requests** library with retry logic implemented through **urllib3** to ensure robust and fault-tolerant retrieval.

DATA & CODE AVAILABILITY

All code developed for this project is publicly available at our GitHub repository: **https://github.com/annakorda/PocketQuest**
The starting dataset was obtained from the BindingDB database, using only data curated from scientific articles. The file used was: **BindingDB_BindingDB_Articles_202504_tsv.zip**
This file and other related datasets can be accessed via the BindingDB data download page: **https://www.bindingdb.org/rwd/bind/chemsearch/marvin/Download.jsp**
All input data, including Connolly point arrays, batch-wise training & assessment PDBs can be accessed via: **https://drive.google.com/drive/folders/1HJYW3KGppZ8ZeH5sYqZD81vYOaZJNxaR?usp=drive_link**

REFERENCES

1. Zhao, J., Cao, Y. & Zhang, L. Exploring the computational methods for protein-ligand binding site prediction. *Comput. Struct. Biotechnol. J.* **18**, 417–426 (2020).
2. Krivák, R. & Hoksza, D. P2Rank: machine learning based tool for rapid and accurate prediction of ligand binding sites from protein structure. *J. Cheminformatics* **10**, 39 (2018).
3. BindingDB: Measured Binding Data for Protein-Ligand and Other Molecular Systems. (2023).
4. Krivák, R. & Hoksza, D. P2RANK: Knowledge-Based Ligand Binding Site Prediction Using Aggregated Local Features. in *Algorithms for Computational Biology* (eds. Dediu, A.-H., Hernández-Quiroz, F., Martín-Vide, C. & Rosenblueth, D. A.) 41–52 (Springer International Publishing, Cham, 2015). doi:10.1007/978-3-319-21233-3_4.
5. Gowers, R. J. *et al.* MDAnalysis: A Python Package for the Rapid Analysis of Molecular Dynamics Simulations. *scipy* (2016) doi:10.25080/Majora-629e541a-00e.
6. Anand, P., Yeturu, K. & Chandra, N. PocketAnnotate: towards site-based function annotation. *Nucleic Acids Res.* **40**, W400–W408 (2012).
7. Lee, B. & Richards, F. M. The interpretation of protein structures: Estimation of static accessibility. *J. Mol. Biol.* **55**, 379-IN4 (1971).
8. Greer, J. & Bush, B. L. Macromolecular shape and surface maps by solvent exclusion. *Proc. Natl. Acad. Sci.* **75**, 303–307 (1978).
9. Sanner, M. F., Olson, A. J. & Spehner, J.-C. Reduced surface: An efficient way to compute molecular surfaces. *Biopolymers* **38**, 305–320 (1996).
10. Kapcha, L. H. & Rossky, P. J. A Simple Atomic-Level Hydrophobicity Scale Reveals Protein Interfacial Structure. *J. Mol. Biol.* **426**, 484–498 (2014).
11. Khazanov, N. A. & Carlson, H. A. Exploring the Composition of Protein-Ligand Binding Sites on a Large Scale. *PLOS Comput. Biol.* **9**, e1003321 (2013).
12. Kyte, J. & Doolittle, R. F. A simple method for displaying the hydropathic character of a protein. *J. Mol. Biol.* **157**, 105–132 (1982).
13. Pintar, A., Carugo, O. & Pongor, S. CX, an algorithm that identifies protruding atoms in proteins. *Bioinformatics* **18**, 980–984 (2002).
14. Lins, L. & Brasseur, R. The hydrophobic effect in protein folding. *FASEB J.* **9**, 535–540 (1995).
15. Sun, Z., Liu, Q., Qu, G., Feng, Y. & Reetz, M. T. Utility of B-Factors in Protein Science: Interpreting Rigidity, Flexibility, and Internal Motion and Engineering Thermostability. *Chem. Rev.* **119**, 1626–1665 (2019).
16. Radivojac, P. *et al.* Protein flexibility and intrinsic disorder. *Protein Sci.* **13**, 71–80 (2004).
17. Yuan, Z., Zhao, J. & Wang, Z.-X. Flexibility analysis of enzyme active sites by crystallographic temperature factors. *Protein Eng.* **16**, 109–114 (2003).
18. Schlessinger, A. & Rost, B. Protein flexibility and rigidity predicted from sequence. *Proteins Struct. Funct. Bioinforma.* **61**, 115–126 (2005).
19. Thomas, P. D. *et al.* PANTHER: a browsable database of gene products organized by biological function, using curated protein family and subfamily classification. *Nucleic Acids Res.* **31**, 334–341 (2003).
20. Chen, T. & Guestrin, C. XGBoost: A Scalable Tree Boosting System. in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* 785–794 (2016). doi:10.1145/2939672.2939785.
21. Imani, M., Beikmohammadi, A. & Arabnia, H. R. Comprehensive Analysis of Random Forest and XGBoost Performance with SMOTE, ADASYN, and GNUS Under Varying Imbalance Levels. *Technologies* **13**, 88 (2025).
22. Hosseini, A. & Serag, A. Is synthetic data generation effective in maintaining clinical biomarkers? Investigating diffusion models across diverse imaging modalities. *Front. Artif. Intell.* **7**, 1454441 (2025).
23. Akiba, T., Sano, S., Yanase, T., Ohta, T. & Koyama, M. Optuna: A Next-generation Hyperparameter Optimization Framework. in *Proceedings of the 25th ACM SIGKDD International Conference on*

Knowledge Discovery & Data Mining 2623–2631 (Association for Computing Machinery, New York, NY, USA, 2019). doi:10.1145/3292500.3330701.

SUPPLEMENTARY MATERIAL

Table 1. Constant values of the atom related attributes for all the atoms of each natural amino acid.

ALA	Hydrophobicity	Valid Ligand Propensity	Invalid Ligand Propensity	Aromatic	Acceptor	Donor
N	1	0.77	0.62	0	0	0
CA	-1	0.63	0.51	0	0	0
C	1	0.41	.035	0	0	0
O	1	0.71	0.55	0	0	0
CB	-1	1.55	1.05	0	0	0
ARG	**Hydrophobicity**	**Valid Ligand Propensity**	**Invalid Ligand Propensity**	**Aromatic**	**Acceptor**	**Donor**
N	1	0.32	0.21	0	0	0
CA	-1	0.23	0.17	0	0	0
C	1	0.16	0.12	0	0	0
O	1	0.2	0.18	0	0	0
CB	-1	0.38	0.26	0	0	0
CG	-1	0.43	0.28	0	0	0
CD	-1	0.71	0.43	0	0	0
NE	1	0.99	0.7	0	0	1
CZ	1	1.33	0.99	0	0	0
NH1	1	1.62	1.19	0	0	2
NH2	1	2.05	1.57	0	0	2
ASN	**Hydrophobicity**	**Valid Ligand Propensity**	**Invalid Ligand Propensity**	**Aromatic**	**Acceptor**	**Donor**
N	1	0.42	0.32	0	0	0
CA	-1	0.33	0.31	0	0	0
C	1	0.28	0.25	0	0	0
O	1	0.45	0.34	0	0	0
CB	.1	0.54	0.59	0	0	0
CG	1	0.93	0.93	0	0	0

OD1	1	1.35	0.92	0	2	0
ND2	1	1.89	1.82	0	0	2
ASP	**Hydrophobicity**	**Valid Ligand Propensity**	**Invalid Ligand Propensity**	**Aromatic**	**Acceptor**	**Donor**
N	1	0.33	0.36	0	0	0
CA	-1	0.39	0.35	0	0	0
C	1	0.27	0.26	0	0	0
O	1	0.29	0.39	0	0	0
CB	-1	0.38	0.47	0	0	0
CG	1	1.32	0.77	0	0	0
OD1	1	1.93	1.03	0	2	0
OD2	1	2.16	1.02	0	2	0
CYS	**Hydrophobicity**	**Valid Ligand Propensity**	**Invalid Ligand Propensity**	**Aromatic**	**Acceptor**	**Donor**
N	1	0.49	0.39	0	0	0
CA	-1	0.6	0.79	0	0	0
C	1	0.46	0.22	0	0	0
O	1	0.79	0.46	0	0	0
CB	-1	0.9	1.56	0	0	0
SG	1	1.42	2.64	0	0	0
GLN	**Hydrophobicity**	**Valid Ligand Propensity**	**Invalid Ligand Propensity**	**Aromatic**	**Acceptor**	**Donor**
N	1	0.31	0.36	0	0	0
CA	-1	0.31	0.29	0	0	0
C	1	0.23	0.18	0	0	0
O	1	0.37	0.32	0	0	0
CB	-1	0.44	0.45	0	0	0
CG	-1	0.56	0.5	0	0	0
CD	1	0.98	0.56	0	0	0
OE1	1	1.52	0.67	0	2	0

NE2	1	1.58	1.12	0	0	2
GLU	**Hydrophobicity**	**Valid Ligand Propensity**	**Invalid Ligand Propensity**	**Aromatic**	**Acceptor**	**Donor**
N	1	0.21	0.29	0	0	0
CA	-1	0.24	0.3	0	0	0
C	1	0.24	0.19	0	0	0
O	1	0.38	0.31	0	0	0
CB	-1	0.21	0.33	0	0	0
CG	-1	0.34	0.34	0	0	0
CD	1	1.12	0.71	0	0	0
OE1	1	1.72	0.96	0	2	0
OE2	1	1.79	1.07	0	2	0
GLY	**Hydrophobicity**	**Valid Ligand Propensity**	**Invalid Ligand Propensity**	**Aromatic**	**Acceptor**	**Donor**
N	1	1.42	0.84	0	0	0
CA	-1	1.97	1.46	0	0	0
C	1	0.97	0.78	0	0	0
O	1	0.93	0.92	0	0	0
HIS	**Hydrophobicity**	**Valid Ligand Propensity**	**Invalid Ligand Propensity**	**Aromatic**	**Acceptor**	**Donor**
N	1	0.2	0.18	0	0	0
CA	-1	0.23	0.32	0	0	0
C	1	0.18	0.19	0	0	0
O	1	0.31	0.22	0	0	0
CB	-1	0.43	0.44	0	0	0
CG	-1	0.44	0.36	1	0	0
ND1	1	0.86	0.81	1	1	1
CD2	-1	0.98	1.1	1	0	0
CE1	1	1.45	1.51	1	0	0
NE2	1	1.79	2.05	1	1	1

ILE	Hydrophobicity	Valid Ligand Propensity	Invalid Ligand Propensity	Aromatic	Acceptor	Donor
N	1	0.45	0.39	0	0	0
CA	-1	0.31	0.25	0	0	0
C	1	0.24	0.23	0	0	0
O	1	0.55	0.42	0	0	0
CB	-1	0.34	0.24	0	0	0
CG1	-1	0.67	0.3	0	0	0
CG2	-1	0.97	0.52	0	0	0
CD1	-1	1.12	0.68	0	0	0
LEU	**Hydrophobicity**	**Valid Ligand Propensity**	**Invalid Ligand Propensity**	**Aromatic**	**Acceptor**	**Donor**
N	1	0.48	0.47	0	0	0
CA	-1	0.29	0.21	0	0	0
C	1	0.31	0.22	0	0	0
O	1	0.59	0.37	0	0	0
CB	-1	0.42	0.31	0	0	0
CG	-1	0.31	0.13	0	0	0
CD1	-1	0.99	0.55	0	0	0
CD2	-1	0.89	0.62	0	0	0
LYS	**Hydrophobicity**	**Valid Ligand Propensity**	**Invalid Ligand Propensity**	**Aromatic**	**Acceptor**	**Donor**
N	1	0.64	0.29	0	0	0
CA	-1	0.4	0.25	0	0	0
C	1	0.21	0.18	0	0	0
O	1	0.22	0.27	0	0	0
CB	-1	0.45	0.33	0	0	0
CG	-1	0.49	0.25	0	0	0
CD	-1	0.56	0.51	0	0	0
CE	-1	1.34	0.88	0	0	0

NZ	1	2.42	1.6	0	0	3
MET	**Hydrophobicity**	**Valid Ligand Propensity**	**Invalid Ligand Propensity**	**Aromatic**	**Acceptor**	**Donor**
N	1	0.5	0.39	0	0	0
CA	-1	0.37	0.34	0	0	0
C	1	0.26	0.25	0	0	0
O	1	0.47	0.36	0	0	0
CB	-1	0.62	0.53	0	0	0
CG	-1	0.75	0.47	0	0	0
SD	1	0.95	0.58	0	0	0
CE	-1	1.15	0.72	0	0	0
PHE	**Hydrophobicity**	**Valid Ligand Propensity**	**Invalid Ligand Propensity**	**Aromatic**	**Acceptor**	**Donor**
N	1	0.29	0.41	0	0	0
CA	-1	0.22	0.3	0	0	0
C	1	0.22	0.23	0	0	0
O	1	0.43	0.34	0	0	0
CB	-1	0.5	0.43	0	0	0
CG	-1	0.45	0.21	1	0	0
CD1	-1	0.72	0.37	1	0	0
CD2	-1	0.71	0.34	1	0	0
CE1	-1	1.01	0.54	1	0	0
CE2	-1	1	0.5	1	0	0
CZ	-1	1.16	0.53	1	0	0
PRO	**Hydrophobicity**	**Valid Ligand Propensity**	**Invalid Ligand Propensity**	**Aromatic**	**Acceptor**	**Donor**
N	-1	0.24	0.41	0	0	0
CA	-1	0.36	0.43	0	0	0
C	1	0.38	0.36	0	0	0
O	1	0.76	0.23	0	0	0

CB	-1	0.63	0.52	0	0	0
CG	-1	1.01	0.56	0	0	0
CD	-1	0.99	0.56	0	0	0
SER	**Hydrophobicity**	**Valid Ligand Propensity**	**Invalid Ligand Propensity**	**Aromatic**	**Acceptor**	**Donor**
N	1	0.79	0.51	0	0	0
CA	-1	0.7	0.58	0	0	0
C	1	0.41	0.33	0	0	0
O	1	0.54	0.53	0	0	0
CB	-1	1.37	1.08	0	0	0
OG	1	2.05	1.43	0	2	2
THR	**Hydrophobicity**	**Valid Ligand Propensity**	**Invalid Ligand Propensity**	**Aromatic**	**Acceptor**	**Donor**
N	1	0.69	0.52	0	0	0
CA	-1	0.51	0.48	0	0	0
C	1	0.33	0.31	0	0	0
O	1	0.52	0.52	0	0	0
CB	-1	0.91	0.74	0	0	0
OG1	1	1.01	0.63	0	2	1
CG2	-1	1.8	1.46	0	0	0
TRP	**Hydrophobicity**	**Valid Ligand Propensity**	**Invalid Ligand Propensity**	**Aromatic**	**Acceptor**	**Donor**
N	1	0.2	0.36	0	0	0
CA	-1	0.21	0.19	0	0	0
C	1	0.2	0.15	0	0	0
O	1	0.33	0.2	0	0	0
CB	-1	0.49	0.39	0	0	0
CG	-1	0.52	0.33	1	0	0
CD1	-1	0.8	0.58	1	0	0
CD2	-1	0.67	0.43	1	0	0

NE1	1	1.18	1.05	1	0	1
CE2	-1	0.95	0.65	1	0	0
CE3	-1	0.71	0.49	1	0	0
CZ2	-1	1.14	0.71	1	0	0
CZ3	-1	0.78	0.66	1	0	0
CH2	-1	0.93	0.67	1	0	0
TYR	**Hydrophobicity**	**Valid Ligand Propensity**	**Invalid Ligand Propensity**	**Aromatic**	**Acceptor**	**Donor**
N	1	0.2	0.33	0	0	0
CA	-1	0.22	0.17	0	0	0
C	1	0.17	0.2	0	0	0
O	1	0.27	0.21	0	0	0
CB	-1	0.42	0.32	0	0	0
CG	-1	0.39	0.17	1	0	0
CD1	-1	0.59	0.3	1	0	0
CD2	-1	0.61	0.27	1	0	0
CE1	-1	0.9	0.61	1	0	0
CE2	-1	0.88	0.5	1	0	0
CZ	-1	0.96	0.62	1	0	0
OH	1	2.03	1.13	0	1	1
VAL	**Hydrophobicity**	**Valid Ligand Propensity**	**Invalid Ligand Propensity**	**Aromatic**	**Acceptor**	**Donor**
N	1	0.56	0.52	0	0	0
CA	-1	0.31	0.24	0	0	0
C	1	0.3	0.24	0	0	0
O	1	0.7	0.42	0	0	0
CB	-1	0.39	0.26	0	0	0
CG1	-1	0.95	0.64	0	0	0
CG2	-1	1.07	0.64	0	0	0

Table 2. Amino Acid Descriptors Used for Feature Encoding

	Hydrophobic	Hydrophilic	H-index	Aliphatic	Aromatic	Sulfur	Hydroxyl	Basic	Acid	Amide	Positive	Negative	HB D	HB A	D & A	Polar	Ionizable
ALA	1	0	1.8	1	0	0	0	0	0	0	0	0	0	0	0	0	0
ARG	0	1	-4.5	0	0	0	0	1	0	0	1	0	1	0	0	1	1
ASN	0	1	-3.5	0	0	0	0	0	0	0	0	0	1	1	1	1	0
ASP	0	1	-3.5	0	0	0	0	0	1	1	0	1	0	1	0	1	1
CYS	0	1	2.5	0	0	1	0	0	0	0	0	0	0	0	0	1	1
GLN	0	1	-3.5	0	0	0	0	0	0	1	0	0	1	1	1	1	0
GLU	0	1	-3.5	0	0	0	0	0	1	0	0	1	0	1	0	1	1
GLY	1	0	-0.4	1	0	0	0	0	0	0	0	0	0	0	0	0	0
HIS	0	1	-3.2	0	1	0	0	1	0	0	1	0	1	1	1	1	1
ILE	1	0	4.5	1	0	0	0	0	0	0	0	0	0	0	0	0	0
LEU	1	0	3.8	1	0	0	0	0	0	0	0	0	0	0	0	0	0
LYS	0	1	-3.9	0	0	0	0	1	0	0	1	0	1	0	0	1	1
MET	1	0	1.9	1	0	1	0	0	0	0	0	0	0	0	0	0	0
PHE	1	0	2.8	0	1	0	0	0	0	0	0	0	0	0	0	0	0
PRO	1	0	-1.6	1	0	0	0	0	0	0	0	0	0	0	0	0	0
SER	0	1	-0.8	0	0	0	1	0	0	0	0	0	1	1	1	1	0
THR	0	1	-0.7	0	0	0	1	0	0	0	0	0	1	1	1	1	0
TRP	1	0	-0.9	0	1	0	0	0	0	0	0	0	1	0	0	0	0
TYR	1	0	-1.3	0	1	0	0	0	0	0	0	0	1	1	1	0	1
VAL	1	0	4.2	1	0	0	0	0	0	0	0	0	0	0	0	0	0