Charles University in Prague
Faculty of Science

BACHELOR THESIS

Jakub Telčer

Detection of Similar Binding Sites in Protein Structure Databases

Department of Cell Biology

Supervisor of the bachelor thesis: doc. RNDr. David Hoksza, Ph.D.

Study programme: Bioinformatika

Study branch: B-BINF

Prague 2024

Prohlašuji, že jsem tuto bakalářskou práci vypracoval(a) samostatně a výhradně s použitím citovaných pramenů, literatury a dalších odborných zdrojů. Tato práce nebyla využita k získání jiného nebo stejného titulu.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona v platném znění, zejména skutečnost, že Univerzita Karlova má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle §60 odst. 1 autorského zákona.

V ..... dne ....

Podpis autora

I would like to sincerely thank doc. RNDr. David Hoksza, Ph.D., for his ability to get me unstuck from a local minima and for constantly directing me towards the desired output. I also very much appreciate the friendliness of everyone else in the Charles University Structural Bioinformatics Group. I'd especially like to thank Troll for the much-needed proofreading.

A největší díky patří Verunce za neustálé zásobování a neutuchající podporu.

Title: Detection of Similar Binding Sites in Protein Structure Databases

Author: Jakub Telčer

Department: Department of Cell Biology

Supervisor: doc. RNDr. David Hoksza, Ph.D., Department of Software Engineering

Abstract: The evaluation of protein-ligand binding site similarity is crucial in many fields, from drug repurposing trials to evolutionary studies. Current state-of-the-art methods achieve good results on the benchmarking datasets. However, the current approaches operate over pairs of binding sites and are not applicable for searching databases of unprocessed protein structures. In cases when the binding sites are unknown, they have to be firstly located by using binding site prediction algorithms. That significantly increases the upfront costs of creating large databases of similar binding sites. This work covers the current methods for assessing binding site similarity and explores the possibility of fast searching of large databases of related structures by presenting a simple method that allows faster than linear search without the need for identification of the precise locations of the putative binding sites. The proposed approach shows promising preliminary results that merit further investigation, although more insight is still needed.

Keywords: protein-ligand interaction binding site similarity database searching

Název práce: Detekce podobných vazebných míst v databázích proteinových struktur

Autor: Jakub Telčer

Katedra: Katedra buněčné biologie

Vedoucí bakalářské práce: doc. RNDr. David Hoksza, Ph.D., Katedra softwarového inženýrství

Abstrakt: Vyhodnocování podobnosti protein-ligand vazebných míst je důležité v mnoha oblastech, jako je výzkum nových využití stávajících léčiv nebo evoluční studie. Nejmodernější současné přístupy dosahují dobrých výsledků, ale pracují pouze s definovanými vazebnými místy, což není možné v nepředzpracovaných proteinových databázích. Pokud tato místa nejsou známá, je nejprve nutné prohledat proteinové struktury nástroji pro jejich predikci. To významně navyšuje cenu velkých databází podobných vazebných míst. Tato práce se zaměřuje na popis stávajících metod pro hodnocení podobnosti protein-ligand vazebných míst a zkoumá možnosti rychlého vyhledávání podobných míst ve velkých databázích příbuzných struktur. Je zde navržena jednoduchá metoda umožňující rychlejší než lineární vyhledávání bez nutnosti predikce potenciálních vazebných míst. Předběžné výsledky naznačují, že tento přístup je hoden další pozornosti, ačkoliv je stále zapotřebí více vhledu do dané problematiky.

Klíčová slova: protein-ligandové interakce vazebná místa podobnost databázové vyhledávání

Contents

Introduction

Protein-ligand interactions play a key role in all living organisms. From enzymatic reactions to signaling pathways, the binding of non-protein molecules is essential to the correct function of our cells. Understanding these interactions allows us to inspect various metabolic and signaling pathways more thoroughly, shed light on unknown protein functions, detect potential side effects of medications due to off-target binding of drugs, potentially discover new treatments by repurposing existing drugs, and understand evolutionary relationships between proteins.

Unraveling the intricate details of some protein-ligand interactions might not be possible without the use of more sophisticated methods, such as molecular simulations or docking. However, many workflows require fast and accessible large-scale searching and screening of proteins for similar binding sites. Various tools that simplify the problem to a matter of correct spatial distribution of chemically suitable atoms, residues or interaction-capable points exist. However, their usage is limited to proteins with pre-identified binding sites, as these tools only compare defined binding sites and assess their similarity. That can be achieved by using various existing putative pocket detection tools.

For certain applications, searching large databases, such as the Protein Data Bank with more than 200,000 entries or the AlphaFold database with more than 200,000,000, is required. Although many databases of similar binding sites already exist, they are usually limited in size to, at most, tens of thousands of protein structures. Just to locate the potential cavities in the AlphaFold database, a prerequisite for searching similar binding sites, current state-of-the-art methods for detecting putative binding sites would require months of CPU time on dozens of cores. Subsequent searches might be faster, depending on the used method, nonetheless the upfront costs would be significant.

This work dives into the current state-of-the-art methods for comparing binding sites and assessing their similarity, compares their performance on exemplary datasets and explores the possibilities of faster than linear search in large databases without the need for costly screening of all structures for putative binding sites. One such approach, based on the idea of exploiting the partial structural similarity of related structures, is proposed. Its implementation is shown along

with preliminary results on multiple datasets.

Implications of this novel approach are discussed, and the directions for future work are highlighted at the very end of this work.

Chapter 1

Literature Overview

Binding site similarity is a very helpful concept. It is particularly important in fields such as *drug repurposing* [1], *drug promiscuity* [2], *protein and protein-complex function prediction* [3, 4], *drug design* [5], *RNA-binding drug discovery* [5], *precision medicine* [5] and others.

However, the exact definition of *similarity* is highly dependent on a given context and is often defined based on the sought objective. For example, in drug repurposing assays, the goal might be to either (i) Find known drugs that bind to a specific protein or (ii) Find different proteins binding to a known drug.

Based on this objective, we could infer a reasonable definition of similarity for this specific purpose, i.e., *similar binding sites bind similar ligands.* The similarity between the ligands, usually small molecules, could be then assessed via multiple methods [6].

However, for the prediction of protein function, the objective is to find, e.g., active sites similar to those known and well-annotated. From this, it may be possible to deduce the protein function. That needs a slightly different definition, though, as the ligand needs to be bound in the same manner. Consider the difference between the enzymatic binding of a ligand as the substrate, a molecule on which the enzyme acts, or as an effector, which modulates the enzyme's activity.

In an evolutionary examination of binding site similarity, the need is to determine whether the binding sites are related, that is, whether they originated from the same ancestor. However, that might not mean the ligand is bound still in the same way, nor even that the sites still bind the same or at least similar ligand, as required by the 'definition' used in drug repurposing.

The variation and contradiction of the 'definitions' imply no ultimate definition of binding site similarity exists. As Christiane Ehrt et al. stated in their ProSPECCTs benchmark compilation, "binding site similarity 'lies in the eye of the beholder'." [7].

For purposes of this work, we suppose binding sites are similar, provided they share common physicochemical (similar amino acid residues) and structural (relative spatial positioning of given residues) features. This 'definition' neglects the inherent flexibility of binding sites, phenomenons such as induced fit, conformational selection, flexibility of the ligand, and other factors. However, these simplifications need to be done to reduce the sheer difficulty of the problem. The effects of these simplifications can later be evaluated on the existing datasets, and findings concerning related and unrelated binding site similarities are discussed.

1.1 Pocket Detection

Correct pocket identification is a prerequisite of reliable comparative methods. Many different approaches can be utilized; some rely on the presence of the ligand, so-called ligand-centric, and some rely on statistical descriptions of binding sites, called ligand-free approaches. In these methods, some frequent characteristics of binding sites are searched. For example, the complementarity of protein-protein interfaces is frequently driven by apolar contacts [8] whilst polar interactions and existing shape complementarity between protein and the binding molecule tends to be more important in protein-ligand complexes [9].

Ligand-centric approaches usually delimit the binding site by a specific distance cutoff from the ligand or its heavy atoms. Commonly used cutoffs are 4 to 6.5 Å. However, more intricate approaches were also used [10] to increase the specificity of the method and more carefully select the residues or atoms that interact with the ligand. An example of such an approach is finding protein-ligand interaction fingerprints from docking poses in a discretized grid around the ligand.

Ligand-free approaches are more useful in terms of detecting putative binding pockets in large databases, where the ligands are very likely to be absent. They can be classified as geometric, energetic, or data-driven.

Geometric methods use various methods of identifying sufficiently buried zones free of protein atoms. Many utilize grid-searching strategies to do so [12, 13, 14, 15]. Another frequent strategy is to use spherical probes to fill the cavity or to coat the surface [16, 17].

Energetic methods use the theoretical potential energy of probes distributed on the protein surface [18] or in a discrete grid surrounding the protein [19].

Different ligand-free methods, such as ConSeq [20], locate binding sites using sequential information. In the case of ConSeq, the predictions are also made based on solvent-accessibility and evolutionary rates [20].

Lastly, data-driven approaches utilize supervised machine learning trained on known binding sites. They differ in the selected features and their representations,

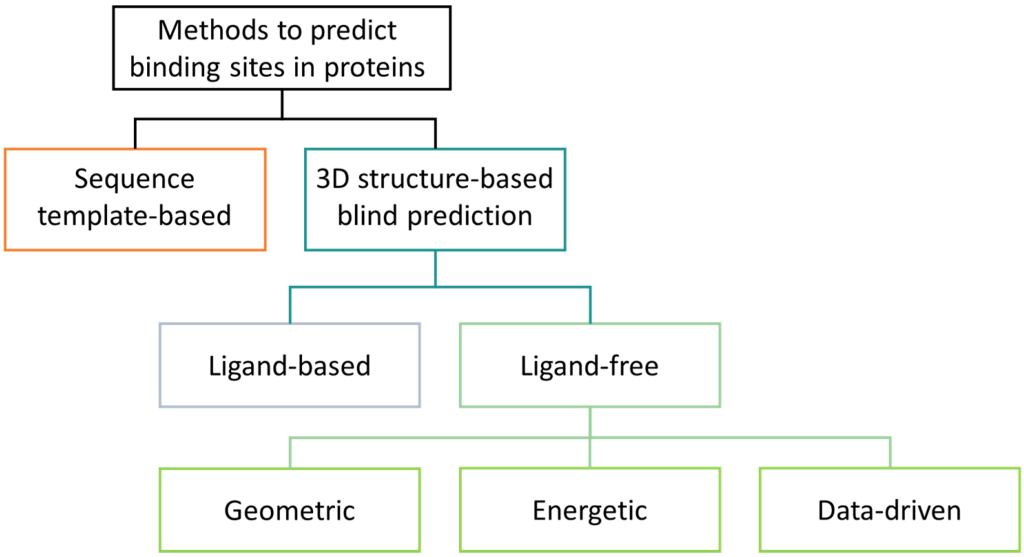

Figure 1.1 Classification of binding site detection methods. Adapted from Eguida and Rognan [11].

machine learning models, hyper-parameters, pre-processing and training datasets. For example, P2Rank [21] is based on a random forest classifier having local solvent-exposed atoms encoded into topological and physicochemical feature vectors as inputs. Recently, methods using convolutional neural networks, such as DeepPocket [22], PUResNet [23], or DeepSite [24], 3D point cloud deep learning and 3D sparse convolution [25] have emerged.

Overall, the methods seem to be quite reliable in detecting known binding sites on *holo* datasets, although they usually tend to predict more potential binding sites than reported in literature [21].

1.1.1 Runtime of Current State-of-the-art Methods

The runtime of the methods is heavily dependent on the size of the protein structure. Fpocket [26] and P2Rank [21] have reported single-core runtimes in seconds on most of the structures [21]. LIGSITE [27] takes between 5 to 20 seconds on a single core per the prediction of medium-sized protein [28].

1.2 Existing Methods for Protein Binding Site Similarity Evaluation

Once the pockets are delimited, they can be compared. Current state-of-the-art methods for evaluating protein binding site similarity are all based on pairwise comparisons. The similarity itself is not a metric[1] though, which prevents the usage of many helpful techniques that eliminate the need for pairwise comparisons of all-vs-all[2]. However, some, e.g., SiteMine [29], are extended with a more conventional database search, where the found matches are then aligned and scored. The current methods usually expect either the delimited binding site as an input or perform the binding site detection, and the user can later specify the site of interest. In the latter case, the binding site detection algorithm, as the ones mentioned above, is used prior to the comparison. Many require the ligand as an input to perform the binding site isolation or require its presence to correctly locate the protein-ligand interactions.

For comparing two given pockets, the methods usually follow these three steps: (i) Convert the cavities into appropriate representations; (ii) Locate patterns shared between the two compared cavities; (iii) Score the match between them based on the found patterns.

1.2.1 Site Representation

Initially, the binding site has to be converted to a suitable representation. That means selecting important features and characteristics relevant to the problem and ignoring unnecessary information. Almost all of the methods consensually presume that the 3D location of residues and their selected attributes (as size, physicochemical properties, flexibility) explain the specific binding, or recognition, of the ligands in the cavity [30, 2].

Subsequently, they usually differ in the following aspects:

1. The level of simplification of the cavity

   Whether to work with an all-atom model of the cavity, residues, or even coarser representations

2. Considered viewpoint

   The protein viewpoint considers points at the protein surface, however, a few methods adapt the ligand viewpoint, e.g., by projecting the cavity

[1] Binding site similarity cannot be considered a metric as it does not adhere to the axiomatic definition of metric, as triangle inequality and symmetry are required.

[2] Such as metric trees, locality-sensitive hashing, metric embedding or nearest neighbor search.

features onto a polyhedron, voxels, or points, which should represent the ligand and thus can be used for comparisons of the environments in which the ligands would reside (if they would bind).

3. The physicochemical features

   The shape information of the cavity alone is insufficient for the correct similarity estimation [31] when the bound ligand similarity is also of interest. Thus, almost all methods annotate the selected points with pharmacophoric[3] features. In coarse-grained representations, the C_α or C_β atoms are usually annotated based on the properties of their residues. The residues are frequently binned into a few classes based on properties like hydrophobicity or interaction capabilities. Some tools allow residue to be in multiple classes (e.g., SiteAlign). Single or groups of atoms are often discriminated based on their interaction capabilities (CavBase). The amount of defined pharmacophoric classes is usually in the span of five to eight but can range up to 40 (e.g., PocketFEATURE [33]). Some methods consider more specific properties like atomic density, types of individual atoms, or geometrical patterns. For example, CavBase and RAPMAD [34] use the directionality of polar features represented by vectors along with other properties.

The approaches to cavity representation are quite diverse, although all tools seek to find the optimum between detailed representation with high resolution, tolerating greater shape variations, and coarser representation, usually lowering the noise floor and providing a significant speedup.

1.2.2 Methods for Similarity Matching

Once the cavities are converted to their respective representations, common patterns or features usually shared between similar cavities are located. Based on the specified approach and subsequently used algorithm, we can classify the methods into multiple categories, see Figure 1.2.

Geometric Matching

The first category of methods focuses on matching geometrical patterns—mostly pairwise distances, angles, or shapes, such as triangles or polyhedrons—complemented by matching of chemical constraints (similar or compatible residues, atoms, or point types). Imperfect matches are expected, given the 3D

[3] Pharmacophore is an abstract concept in medicinal chemistry. It describes an ensemble of different features essential for specific supramolecular interactions with distinct biological target [32].

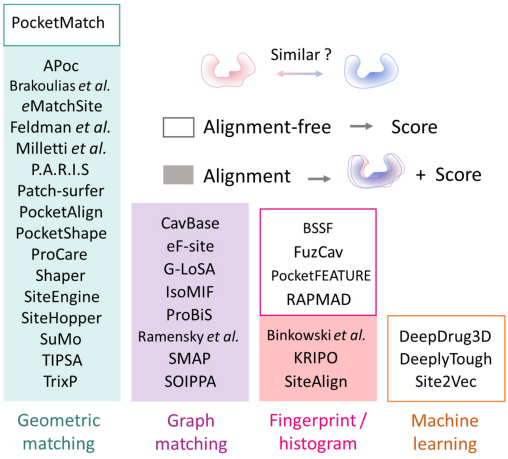

Figure 1.2 Classification of current state-of-the-art methods for protein-ligand binding site comparison. Adapted from Eguida and Rognan [11].

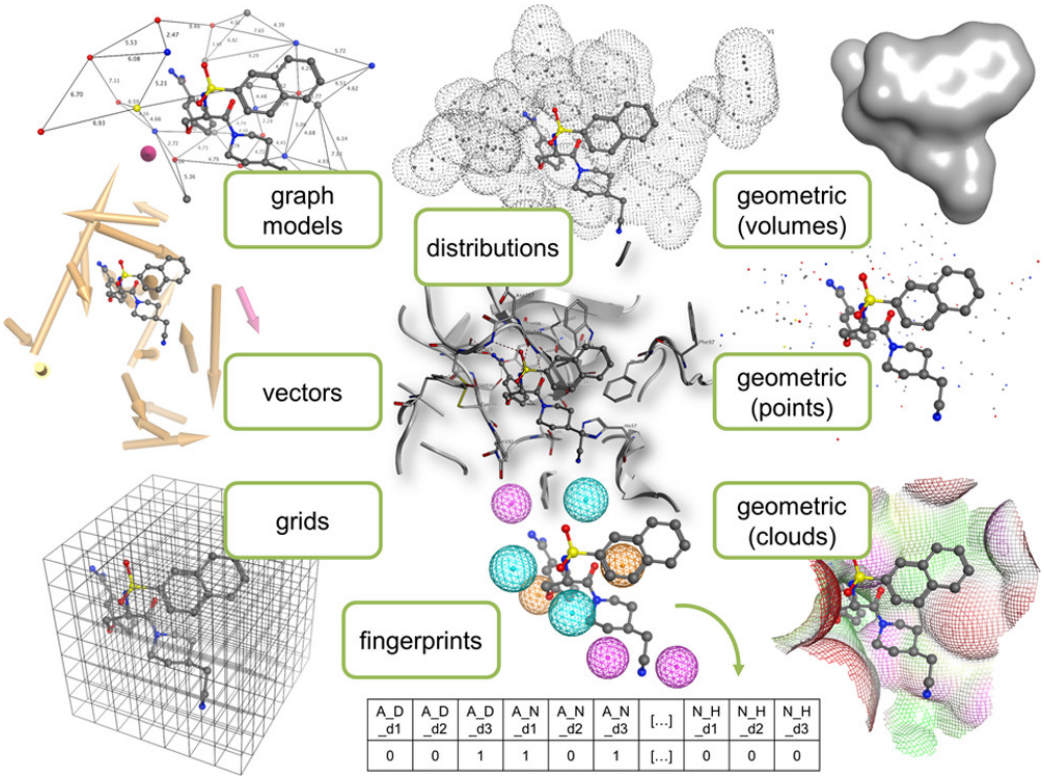

Figure 1.3 Different approaches to represent and match binding site similarities. Adapted from Ehrt, Brinkjost, and Koch [35].

structure resolution (more than 84 % of structures in PDB come from X-ray crystallography[4], while the most common resolution of structures is around 2 Å[5]). The flexibility of proteins and the objective of searching unobvious similarities also contribute to the difficulty of finding correct matches. Thus, a certain margin of error (both geometric and chemical) is allowed.

PocketMatch [31] represents each site by 90 lists of sorted distances, capturing all combinations of residue groups (five groups based on physicochemical properties) and type of points (C_α, C_β, and $C_{centroid}$ – the mean position of all atoms of the side chain of the residue). The distance lists are compiled from all possible pairs of points in the cavity belonging to the given groups and being of the given point types. Similarity score (PMScore) is computed by comparing distance sets with their corresponding equivalents by aligning them (in the alignment, the corresponding distances in the sets, i.e., distances differing only by the method parameter τ, are matched) using a greedy strategy. The net average of the number of matching distances in all of the 90 lists as a fraction of the total number of distances in the bigger list is marked as *PMScore*.

P.A.R.I.S. [36] compares the cavities, represented as a 3D point cloud of atoms, using a convolutional Gaussian kernel of (i) the pairwise distances of atoms in the cavity, and (ii) Gaussian kernel of partial charges of atoms. The principal idea is that the role of a single atom in one cavity could be overtaken by a group of atoms in the other. Hence, by considering the neighborhood (the importance of nearby atoms is set with the parameter σ of the Gaussian kernel), the method can eliminate the effect of single or multiple atom changes, where the surrounding group of atoms remains similar. The comparison between atom-centered kernels requires the knowledge of the optimal superposition between the pockets. The optimum with respect to a distribution of masses defined as a sum of Gaussian functions centered on a given atom x from the pocket is found by the gradient ascent algorithm. The distributions of masses, f_{P_i}, is defined as in 1.1,

$$f_{P_i(x)} = \sum_{x_j \in P_i} e^{-\frac{\|x - x_j\|^2}{\sigma^2}} \tag{1.1}$$

where P_i is the given pocket, and x_j are atoms belonging to P_i. The partial charges are computed in a similar manner. The optimal rotation and translation maximizes

[4]RCSB PDB Statistics Summary, see at www.rcsb.org/stats/summary or on the Internet Archive: web.archive.org/web/20240213022155/https://www.rcsb.org/stats/summary

[5]RSCB PDB Statistics PDB Data Distribution by Resolution, see at rcsb.org/stats/distribution-resolution or on the Internet Archive web.archive.org/web/20240416131257/https://www.rcsb.org/stats/distribution-resolution

the sum of the Gaussian functions over all atoms from the superimposed cavities, as in 1.2,

$$\sum_{x_j \in P_1, y_j \in P_2} e^{-\frac{\|x_j - (Ry_j + T)\|^2}{\sigma^2}} \tag{1.2}$$

where R is an orthonormal rotation matrix, and T is a translation vector. Since the gradient of 1.2 can be calculated, the gradient ascent algorithm can be used. Since many local minima exist, the alignment rotation is seeded by superposing the principal axes of the pockets and the translation vector by superposing the centroids (mean position of all atoms) of the pockets. The best-achieved maxima of 1.2, labeled $sup - CK$, is then used for the similarity scoring.

APoc [37] in the first phase guesses several alignments based on gapless sequential alignments, secondary structure comparison, alignments of fragments and local contact pattern alignments. Then, it employs dynamic programming to iteratively improve the sequential alignment between the two pockets. Lastly, an iterative procedure searching for non-sequential alignment is used, and such solutions are accepted, provided they score better (have a higher PS-score) than the optimal sequential alignment. To find the optimal non-sequential alignment, it solves the Linear Sum Assignment Problem, an equivalent of maximum weight matching in a weighted bipartite graph. For that, they implemented the shortest augmenting path algorithm [38] with polynomial time complexity of $O(N^3)$, where N is the largest of C_α atom counts in the cavities. For scoring, the authors developed PS-score. A scoring function based on (i) alignment length, (ii) distance of aligned residues, (iii) differences of angles between C_α and C_β atoms of aligned residues, (iv) the chemical similarity of the aligned residues.

SiteMine [29] populates a classical relational database with information about each atom in the indexed cavities (source molecule type and element, type of the atom and the residue, solvent-exposed surface area, functional group, and the secondary structure of the fragment the atom belongs to) along with distances to all other search points (indexed atoms) closer than 15 Å. This allows SiteMine to perform classical database searching, apart from comparing pairs, by searching for similar tetrahedrons with similar vertices (entries in the database). The similarity is then evaluated based on scoring matches of close atoms in the found alignment.

Other global alignment methods (SiteHopper, Shaper) try to maximize the overlap between the cavities (similarly to P.A.R.I.S. but not necessarily by comparing the Gaussian kernels). ProCare [39] is built around the cloud registration concept, where each point is characterized by a 41-bin histogram describing both pharmacophoric features and shape. Many methods perform the alignment in two stages. Firstly, by identifying equivalent points by the random sample consensus

algorithm [40], and secondly, by refining the alignment by the iterative closest point algorithm [41].

Graph Matching

The cavity can be identified by a graph[6] with vertices labeled by point labels (e.g., atoms, residues, or pseudo-centers) and edges by the distances between the given points. The problem can be then translated to the problem of finding the largest common subgraph in the graphs of the two cavities while allowing the graphs to differ partially in both the distances and the point labeling.

To achieve this, a product graph[7] is built by associating similar points (e.g., atoms of the same type or amino acid residues belonging to the same group under a particular grouping scheme) and edges of comparable length.

Then, the largest common subgraph can be identified by finding the maximum clique[8] using conventional clique detection algorithms. Commonly used are the Bron-Kerbosch algorithm [42] with an asymptotic runtime complexity of $O(3^{\frac{n}{3}})$, being an optimum in the worst case, as any n-vertex graph can have at most $3^{\frac{n}{3}}$ maximal cliques [43]. In practice, the Bron-Kerbosch algorithm runs very fast and is used to this day, as it remains an algorithm of choice in cases, where all maximal (not just the maximum) cliques need to be reported. Other variations of the Bron-Kerbosch algorithm are also used [44], as well as different algorithms, such as the algorithm of Carraghan and Pardalos [45], which was shown to be several times faster on molecular 3D structures than the Bron-Kerbosch algorithm [46].

The methods for binding site similarity prediction subsequently differ in the construction of the product graph—which points and distances are used and associated—and scoring of the matches.

CavBase [47] represents the cavity with a 0.5 Å grid of points that are in contact with the protein surface. The points are assigned a property of possible interactions of the nearest close-by residue (only interactions that have meaningful directionality are taken into account; those that are directed towards the protein interior are ignored).

[6] For a given finite set of vertices V and list of edges E, where $E \subseteq V \times V$, we denote $G = (V, E)$ as a simple undirected graph.

[7] Given graphs $G_1 = (V_1, E_1), G_2 = (V_2, E_2)$, graph product is a graph $G = (V, E)$ where $V \subseteq V_1 \times V_2$ and edge $e = ((v_1^a, v_2^a), (v_1^b, v_2^b))$ is added given a certain condition is satisfied (e.g., distance between v_1^a and v_1^b matches the distance between v_2^a and v_2^b).

[8] A clique is a subset of the vertices where all the vertices are adjacent. In other words, a subgraph induced by the clique is complete. Maximal clique is clique that cannot be extended by adding any vertex. Maximum clique is the largest of all maximal cliques in a graph.

IsoMIF [48] uses Molecular Interaction Fields[9] (MIFs) to encode the physicochemical environment of the pocket. The cavity is discretized into a grid where each point is assigned a MIF based the on interaction potential energies of six different chemical probes. Fields of the matched cavities, composed of the MIFs from the grids, are subsequently matched using the Bron-Kerbosch algorithm. IsoMIF performs best when the bound ligand is present. By relying on predicted binding sites, its performance significantly decreases.

Fingerprint and Histogram

These methods do not directly compare the pockets but rather generate specific fingerprints or histograms of the pockets, and these are later compared (e.g., by cosine distance). BSSF [50], KRIPO [51], and FuzCav [52] compute n-tuples (couples or triplets) of certain pharmacophoric features, which are separated by binned distances. Some methods count the number of such triangles found in the cavity, e.g., KRIPO only marks whether the given combination is present in the cavity. SiteAlign projects the properties of the residues on an 80-face polyhedron. Binkowski et al. [53] and RAPMAD are based on comparison of distance distributions of certain features in the cavities. For example, RAPMAD generates multiple distance histograms for given pharmacophoric features and center points. These histograms are then compared. Although these methods generally suffer from the simplification and loss of information (e.g., the numbers of specified triplets are valid, but their actual geometrical conformation is different and unrelated), some of them are particularly suited for database applications, as the search is quite straightforward and computationally undemanding.

Machine Learning

The latest state-of-the-art methods focus on leveraging data-driven approaches, typically binary classifiers based on neural networks. DeepDrug3D [54] and DeeplyTough [55] use convolutional neural networks and deep learning to compare the discretized cavities. Site2Vec [56] uses a different approach – the transformation of the cavities' features into vectors of fixed lengths that can be subsequently fed into a random forest classifier or clustered using conventional clustering algorithms.

Generally, these methods suffer from a lack of balanced, representative and diverse datasets, as well as from the low interpretability of predictions. Deep neural networks are also more heavyweight and usually require GPUs to run at practical speeds.

[9] A molecular interaction field describes the interactions formed around a given point using a three-dimensional map [49].

1.3 Evaluation of Existing Methods

With the plethora of methods for estimating the similarity of binding sites, a good benchmarking method is needed to select the optimal one for the specified purpose. That will always be dependent on the specified task, however, more generalized dataset also exist.

1.3.1 Datasets

With the focus of various binding site similarity estimating methods on different aspects of the problem (e.g., methods focusing on interaction patterns or methods focusing on the overall similarity), many datasets are used for the method optimization and testing. Frequent choices are binding sites in one protein family or enzymes with identical EC numbers. Datasets prepared for evaluating certain tools exist as well, e.g., the APoc dataset [37]. However, the universal dataset used to evaluate most of the methods was for a long not available. Thanks to the work of Christiane Ehrt et al., the ProSPECCTs dataset [7], covering multiple areas of interest, was compiled. Her group also evaluated the performance of many standalone tools. Also, new methods are frequently validated using this dataset. Another benchmarking dataset is the Vertex dataset [57], published earlier than ProSPECCTs, although not as comprehensive. A common source of active protein-ligand pairs is the sc-PDB database [9] containing 16,034 protein-ligand pairs[10], where proteins with suitable drug-like binding pockets were extracted from the PDB.

A different approach was taken by the authors of the THOUGH-M1 dataset, where only sequentially and structurally dissimilar binding sites were gathered.

Most of the datasets consider binding sites binding similar ligands as similar and usually disregard the effect of different ligand binding modes on the site similarity. However, the source of the data is often shared — the Protein Data Bank. Thus, many datasets might adopt the existing bias in the PDB, as disulfide bonds, metal binding sites, and sites involved in enzyme activity are over-represented, while others, as low-complexity regions or transmembrane structures, were found to be underrepresented[11] [58]. Bias in the training and evaluation dataset has already been found to negatively influence the performance of data-driven approaches in similar tasks [59].

[10] As of April 2024. Last update was done in 2017 based on frozen PDB data from November of 2016.

[11] In comparison to SwissProt using PDB version from 2003.

ProSPECCTs [7]

ProSPECCTs (Protein Site Pairs for the Evaluation of Cavity Comparison Tools) contains seven subsets focused on different aspects of binding site similarity and different challenges therein. It contains sequentially identical pairs, NMR structures dataset (focused on sensitivity with respect to the flexibility of the binding sites), identical pairs with multiple artificial residue substitutions in the site (subset containing only physicochemical changes and subset containing physicochemical and shape changes), the dataset of Kahraman and his co-workers [60] (compiled to assess binding site and ligand shape complementarity), Barelier data set [61] (containing pairs of identical ligands binding to unrelated structures), and the dataset of successful applications (protein-ligand pairs described in the literature).

TOUGH-M1 [62]

The TOUGH-M1 dataset compiled by Rajiv Gandhi Govindaraj and Michal Brylinski contains globally dissimilar pairs of structures from the Protein Data Bank (PDB) binding similar drug-like molecules (positive and negative pairs bind ligands with Tanimoto Coefficient[12] ≥ 0.7 and ≤ 0.4 respectively). The TOUGH-M1 dataset contains 505,116 positive cases—pairs of dissimilar proteins binding similar ligands—and 556,810 negative cases—protein pairs with different structures binding dissimilar ligands.

1.3.2 Results of Existing Methods on ProSPECCTs

The following results show the performance of various methods on the ProSPECCTs dataset. From the seven subsets, three subsets were selected: (i) the NMR structure dataset, which was selected as it highlights the differences between predicted and ligand-based binding site definitions and various approaches for similarity estimation; (ii) the Kahraman dataset, as it still poses a major challenge for many methods; (iii) the dataset of successful applications, which would ideally represent the discussed pairs, relevant for the problem of protein-ligand binding, and is expected to be more representative.

The benchmark distinguishes the tools based on what is compared—the residue-based methods compare the geometrical and physicochemical similarity of residues in space. The surface-based methods discretize the surface and compare these points (atoms, pseudo-centers or other representations). The

[12]Commonly used measure of similarity of molecules, represents the ratio of intersection of molecular fingerprints over their union [63].

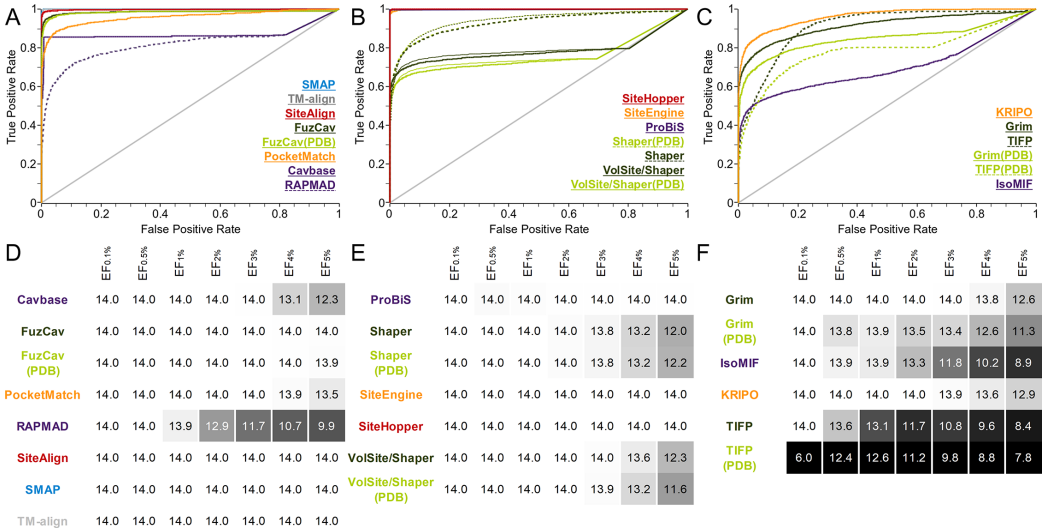

Figure 1.4 Evaluation of different binding site comparison tools with respect to the NMR structures. A-C) ROC curves for residue (A), surface (B) and interaction-based (C) methods. The names are sorted in descending order with respect to the method's AUC. D-F) Enrichment factors for the residue, surface and interaction-based comparison methods. A linear color gradient from white (highest value) to black (lowest) was applied. Thin and dashed lines correspond to the tools of the same colour with differently set parameters. Adapted from Ehrt, Brinkjost, and Koch [7]

interaction-based methods compare the spatial coordination of different types of chemical interactions.

TM-align [64], not a binding site alignment method but a global structural alignment tool, aligns the cavities based on residues at a distance of less than 10 Å from the ligand.

NMR dataset. Figure 1.4 shows the results for various tools tested by Ehrt et al. on the NMR dataset. The NMR dataset includes structures where the similar pockets contain larger and smaller conformational variations, resulting from induced fit and conformational selection [65]. Cavbase and RAPMAD show significant AUC decrease compared to the other tools in the residue-based category which can be explained by the different cavity definition—both methods use LIGSITE [27] pocket identification instead of the ligand-based delimitation. This approached omits some pockets and was unable to compare all the pairs in the dataset.

The sub-optimal performance of some residue- and surface-based methods most probably stems from the structural changes of the structures, which could be probably mitigated by first compiling an ensemble of many similar or identical

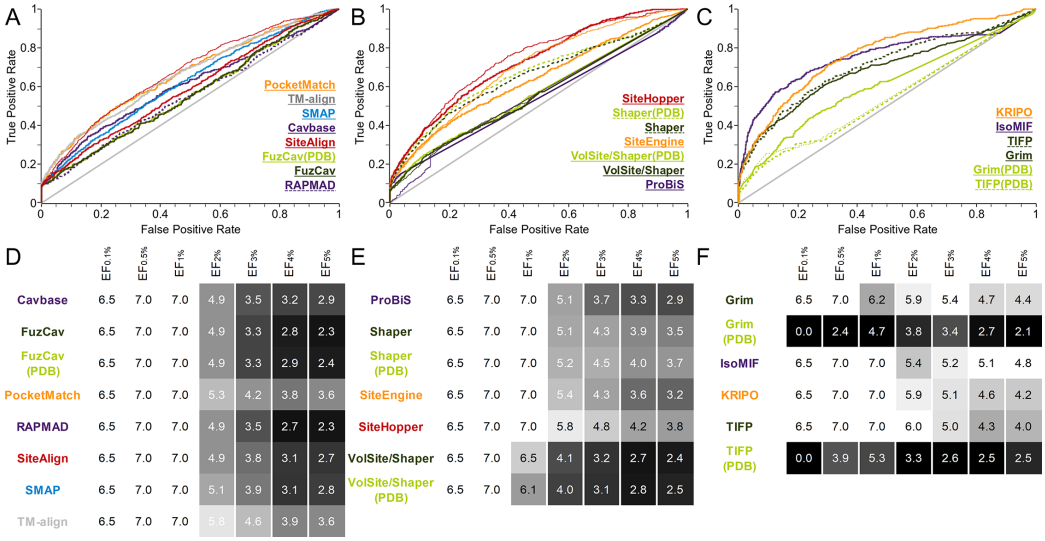

Figure 1.5 Evaluation of different binding site comparison tools with respect to the dataset of Kahraman structures [60] after the exclusion of phosphate binding sites. A-C) ROC curves for residue (A), surface (B) and interaction-based (C) methods. The names are sorted in descending order with respect to the method's AUC. D-F) Enrichment factors for the residue, surface and interaction-based comparison methods. A linear color gradient from white (highest value) to black (lowest) was applied. Thin and dashed lines correspond to the tools of the same colour with differently set parameters. Adapted from Ehrt, Brinkjost, and Koch [7]

binding sites with different conformations and subsequently performing the matching.

Interaction-based methods struggle similarly. The conformational changes induce a shift in the interacting atoms, and thus, the interaction patterns might change slightly. The results of IsoMIF are particularly affected by the pocket's flexibility, whereas KRIPO, an interaction fingerprinting method that introduces fuzziness into the fingerprints to account for the flexibility, performs convincingly better.

Kahraman structures. The dataset of Kahraman and his co-workers [60] is commonly used for the evaluation of different binding site comparison methods. It consists of different cofactor sites and cavities binding small molecules. The dataset was compiled to assess the cavity-ligand shape complementarity. Kahraman et al. concluded, that the shape complementarity alone is not sufficient to determine, whether the ligand will bind, as multiple differently shaped cavities are able to bind the same ligand, however, the shape complementarity is an important driver for the recognition of the ligand.

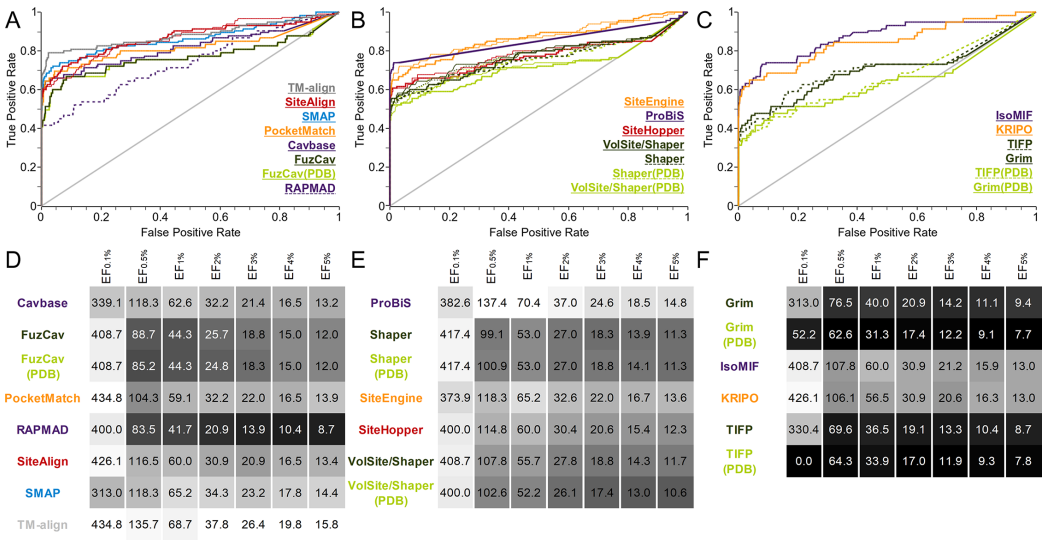

Figure 1.6 Evaluation of different binding site comparison tools with respect to the dataset of successful application. A-C) ROC curves for residue (A), surface (B) and interaction-based (C) methods. The names are sorted in descending order with respect to the method's AUC. D-F) Enrichment factors for the residue, surface and interaction-based comparison methods. A linear color gradient from white (highest value) to black (lowest) was applied. Thin and dashed lines correspond to the tools of the same colour with differently set parameters. Adapted from Ehrt, Brinkjost, and Koch [7]

Figure 1.5 portrays the results with the exclusion of the phosphate binding sites, as due to their small size and a low number of interactions or important residues, some methods failed to process them. Interaction-based methods clearly outperform almost all others. KRIPO, an interaction-based method, outperformed all except SiteHopper [66]. Both KRIPO and IsoMIF show high early enrichment and their early EFs are the highest from the interaction-based methods when all the structures are considered. SiteHopper and IsoMIF also significantly outperform many other methods, which is to be expected in the case of IsoMIF, as it was validated and perhaps optimized for this dataset [48].

Dataset of successful applications. It was compiled from known binding site similarities that appeared in literature [35]. This dataset is an sc-PDB [9] subset and portrays a realistic scenario in the binding site similarity estimation. Ideally, all the tools should be able to reliably detect similar pairs. These are not just pairs with obvious similarities within one protein family but also pairs that are otherwise unrelated. Figure 1.6 shows that most residue-based methods outperform the methods based on surface or interaction similarities. From the interaction-based methods, KRIPO and IsoMIF show the highest early enrichment.

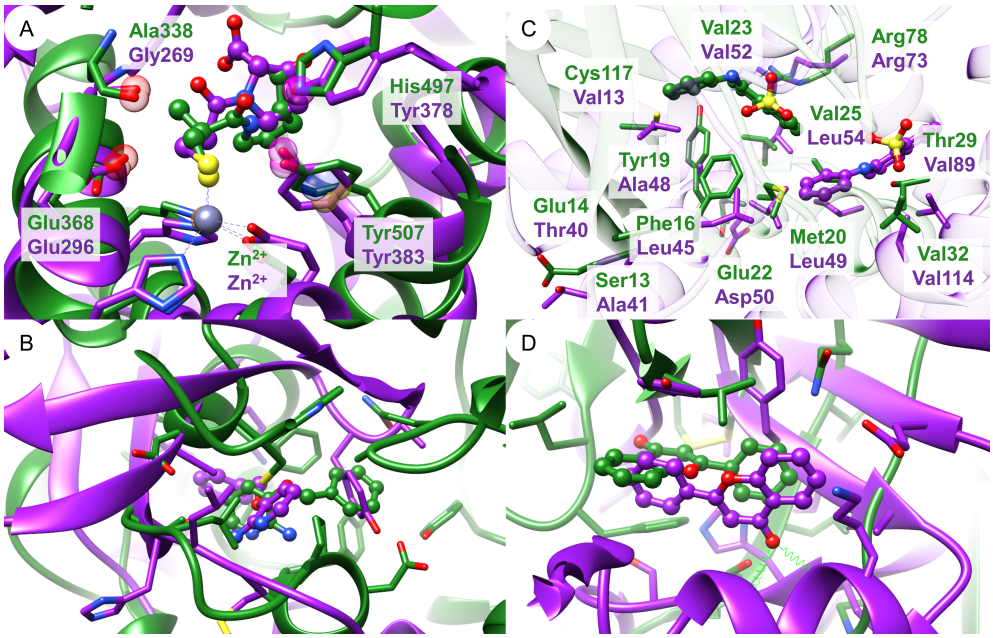

Figure 1.7 Binding site alignments of high-scoring pairs of the Barelier data set generated by (A) Cavbase, (B) TM-align, (C) SMAP, and (D) Shaper. While some superpositions, as in (A) seem reasonable, other such as (C) seem incorrect, as the ligands are not superposed. Adapted from Ehrt, Brinkjost, and Koch [7].

Method	Site2Vec [56]	BindSiteS-CNN [68]	DeeplyTough [55]
Identical Sequences	1.00	0.94	0.95
NMR Structures	1.00	0.83	0.90
Decoy Structures Rational	0.99	0.91	0.76
Decoy Structures Shape	0.99	0.79	0.75
Kahraman	0.86	0.66	0.63
Barelier	0.53	0.62	0.54
Successful Applications	0.66	0.78	0.83
Mean	0.87	0.79	0.77

Table 1.1 Performance of three machine-learning-based methods on all the subsets of the ProSPECCTs benchmark dataset. For each subset, AUCs[a] of the methods are reported. BindSiteS-CNN is uses a spherical convolutional network for the classification of binding sites as either similar or dissimilar.

[a]Area under the ROC (receiver operating characteristics) curve.

Tools offering visual inspection of the resulting alignment are valuable, as it still proves to be a crucial step in the assessment of the significance of the identified match. In Figure 1.7, the SMAP alignment does not superpose the ligands and seems to be incorrect. Hence, tools that provide this visual feedback should be used, preferably in low-throughput scenarios.

Predominantly hydrophobic pockets might be a harder case for the similarity estimation, as many promiscuous and super-promiscuous ligands bind in extreme binding modes often driven by apolar contacts [67] and thus, lower accuracy is generally expected when confronting the methods with such cavities.

On the subject of binding modes, the overall similarity of the binding sites does not necessarily imply identical or even similar binding modes. Rather, hydrophobic and aromatic ligands can adopt various binding modes [7, 67]. Additionally, multiple acceptable alignments are possible for highly hydrophobic pockets, and it might not be possible to distinguish the correct ones.

Performance of Machine-learning Methods

Results for the newer data-driven approaches are reported in Table 1.1. Site2Vec performs the best with the mean area under the curve (AUC) of 0.87 and outperforms all other methods, algorithmic or data-driven, on most of the datasets.

1.3.3 Runtime Analysis

Ehrt et al. also analyzed the runtime cost of the methods, which varies greatly. For reference, consider matching one query site versus all binding sites in the

method	data basis for comparison	run time preparation [s] (number of structures)	run time comparison [s]	total run time [s]	average run time per comparison [s]
PocketMatch[24]	distance lists	28.97*	0.28	29.25	0.000028
KRIPO[56]	fingerprint	446.50	0.92	447.42	0.000092
RAPMAD[31]	histogram	71.42 (100)	2.36 (8,281)	73.78	0.000285
FuzCav[36]	fingerprint	399.88 (96)	5.59 (9,216)	405.47	0.000607
FuzCav(PDB)[36]	fingerprint	236.73 (96)	5.64 (9,216)	242.37	0.000612
TM-align[27]	matrix	25.72*	65.96	91.68	0.006596
Shaper(PDB)[23]	3D points (grid)	181.16 (96)	364.42 (9,216)	545.58	0.039542
Shaper[23]	3D points (grid)	384.21 (96)	367.21 (9,216)	751.42	0.039845
VolSite/Shaper[23]	3D points (grid)	537.00 (76)	248.77 (5,776)	785.77	0.043070
ProBiS[48]	graph	6.95	479.32	486.27	0.047932
VolSite/Shaper(PDB)[23]	3D points (grid)	259.54 (57)	162.26 (3,249)	421.80	0.049942
TIFP[19]	fingerprint	228.30 (77)	550.88 (5,929)	779.18	0.092913
TIFP(PDB)[19]	fingerprint	194.36 (47)	205.56 (2,209)	399.92	0.093056
Grim(PDB)[19]	graph	169.33 (96)	1,714.49 (9,216)	1,883.82	0.186034
Grim[19]	graph	220.17 (95)	2,104.99 (9,025)	2,325.16	0.233240
IsoMIF[22]	graph	752.83	2,561.44	3,314.27	0.256144
SiteHopper[25]	3D points	154.01	3,828.61	3,982.62	0.382861
Cavbase[20,21]	graph	67.89 (100)	21,823.71 (8,281)	21,891.60	2.635396
SMAP[43]	graph	1.69	42,346.74	42,348.43	4.234674
SiteEngine[51]	3D points	328.81	81,193.54	81,522.35	8.119354
SiteAlign[18]	fingerprint	28.97*	286,326.41	286,355.38	28.632641

* exemplary run times for methods that demand a pre-processing by the user

https://doi.org/10.1371/journal.pcbi.1006483.t006

Figure 1.8 Run time evaluation of different binding site comparison methods with respect to the data set of Kahraman et al. The numbers in brackets signify how many pockets were sucesfully prepared for the comparisons or the number of comparisons respectively. The test was performed on an Intel Xeon workstation (E5-2690 with 2.90GHz and 32 GB RAM) using a single core. Adapted from Ehrt, Brinkjost, and Koch [7].

sc-PDB [9]. SiteAlign would perform such a task in a matter of days, whereas PocketMatch would do so in less than a second. The results can be seen in Figure/Table 1.8.

The histogram and fingerprint-based methods, such as PocketMach, KRIPO, RAPMAD and FuzCav, were unsurprisingly the fastest, usually taking milliseconds (or a fraction of a millisecond in the case of PocketMatch) per comparison on average on a single CPU core. The exception was TIFP, which was significantly slower than all others and required tens of milliseconds per comparison. The geometrical methods, based on 3D point comparisons, usually required tens of milliseconds, although SiteEngine required more than 8 seconds per comparison on average. Graph matching methods were one of the slowest, as they usually required hundreds of milliseconds per comparison with the notable exception of ProBis, requiring only tens of milliseconds. The slowest method analyzed was SiteAlign, which needed more than 28 seconds per comparison.

The analysis was done in bulk for more matches whenever possible.

1.4 Existing Databases

Many online comparison services already exist. Some are only online versions of the same comparison tools intended to match two structures against each other, such as SiteEngine [69][13]. However, complete databases containing thousands and even millions of protein pockets also exist. Some, such as CavSimBase [70], pre-calculated all the similarities and are basically constituted by a single similarity matrix, where all the pair combinations were evaluated prior to the search. Some store the pockets in the form of indexed points, triangles or tetrahedrons with certain properties, that can be searched more effectively. The matched points are later extended and joined to either calculate a possible alignment or to evaluate the score of the given match. Here we can mention IsoMIF Finder [71] or SiteMine [29]. Databases directly derived from the PDB are also available, such as the aforementioned sc-PDB [9], which contains over 16,000 pocket-ligand pairs corresponding to roughly 4,700 unique structures.

PLIC [72] clusters pockets of structures of protein-ligand complexes from the PDB based on binding site similarity calculated by PocketMatch. It currently contains over 80,000 pocket-ligand pairs from roughly 30,000 proteins.[14]

ProBis [73] uses the ProBis algorithm [74] to compare the query structures against more than 42,000 structures from the PDB. The algorithm runs parallel, and a single comparison should take several minutes (excluding queue time). Since 2015, it also allows geometry optimization and interaction energy calculation[15] using the CHARMMing [75] web servers.[16]

SiteMine [29] populates a classical relational database with all atoms from each binding site, along with their properties and distances to all other search points closer than 15Å. Tetrahedral search patterns are later processed and searched using GeoMine [76] to retrieve the matches. Although the SiteMine algorithm performs very well on multiple benchmarking datasets, it is not freely accessible at the time of writing.

PoSSuM [77] encodes each binding site as a vector of 1,540 elements, where each element denotes the frequency at which certain triangle types appear in the pocket. Triangles are sets of three residues categorized based on the vertex

[13] Accessible at http://bioinfo3d.cs.tau.ac.il/SiteEngine/php.php
[14] Accessible at http://proline.biochem.iisc.ernet.in/PLIC/.
[15] Accessible at https://probis.nih.gov/.
[16] Accessible at http://probis.cmm.ki.si/

label (four residue labels) and edge label (binned length of the edge between the residues). Multiple feature vectors are created for seven residue classes (e.g., triangles from only positively charged or only aromatic residues are considered). Searching is done using the neighbor-search algorithm SketchSort [78], which searches the feature vectors. Neighbors are later compared using cosine similarity, and pairs of high cosine similarity are aligned using TM-align [64]. PoSSuM can search in multiple databases. PoSSuMAF was created from over 20,000 AlphaFold protein models from the *Homo sapiens* reference genome. PoSSuM database contains about 10,000,000 putative and known binding sites from structures from the PDB. Putative binding pockets in the AlphaFold models and PDB structures were located using the GHECOM [79] program that searches for concave regions on the protein surface.

All aforementioned databases either use only known ligand-binding sites (sites that appear in the PDB as protein-ligand complexes) or screen all entries for putative pockets which are later indexed in the database. Some even compare all the pockets to speed up the subsequent searches. However, this comes at a cost—e.g., CavSimBase authors estimated preparing the database on a regular consumer machine would require approximately 50 years, and by leveraging parallel execution on enterprise-grade hardware (Intel Xeon X5570, 22 GB RAM, two NVIDIA Tesla M2050 GPUs with 2 GB RAM) this would be reduced to about 206 days. After running on eight instances parallel, the computation took 22 days, and the financial cost was over US $7,000 [70]. For context, CavSimBase contains 248,686 putative binding sites extracted from 61,516 protein structures using the LigSite [27] algorithm.

Chapter 2

Proposed Method

Given the size of current large protein structure model databases, such as the AlphaFold database, and given the costs required to perform the location of the putative pockets and pairwise matching for even small subsets, a question arises whether there is a possibility of reducing the upfront costs of building such database. Most promising methods use histogram- and fingerprint-based searches in classical database systems, which comes at the cost of accuracy, and the results should ideally be filtered again using more precise algorithms. That still requires the knowledge of the location and the definition of all putative binding sites that shall be compared. Skipping this step could enable a drastic decrease in the upfront costs, but it has to be done in a way that does not lead to overly slow (and expensive) searches.

From the results of various methods on the ProSPECCTs dataset, TM-align [64] stands out. TM-align is not a binding site similarity assessment tool but an algorithm that identifies the best structural alignment between two protein structures. It firstly aligns the secondary structures of the two proteins using dynamic programming, secondly it uses threading to get a gapless alignment of the structures, and finally iteratively optimized the current alignment. TM-align scores surprisingly well on the testing datasets, which implies that purely structural alignment is a good indicator of binding site similarity. That is in line with other research showing that structural similarity is an utmost important factor in binding site similarity and ligand binding specificity, although it does not explain it completely [80]. Another interesting thing is the partial similarity of the protein sequences, which is exploited by APoc [37] in the first phase.

Although the structural and sequential similarities hold mainly for related proteins, as structurally similar cavities of unrelated proteins binding identical ligands are rarely known and usually differ substantially [61]. Even in cases where the partial similarity of unrelated sites is present, one site is rarely more than approximate of the other [61].

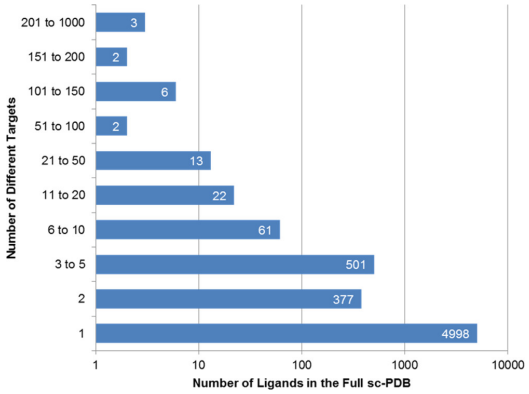

Figure 2.1 The number of ligands in the sc-PDB binned by the number of targets in the sc-PDB. Adapted from Ehrt, Brinkjost, and Koch [35].

On the other hand, super-promiscuous ligands binding to many unrelated structures are also not as common, Figure 2.1 shows that most ligands in the sc-PDB are indeed not super-promiscuous and bind to smaller amount of targets.

At the same time, when two proteins are related, the active sites tend to be more evolutionary conserved [81]. This conservation is very apparent for catalytic sites and residues near bound ligands [82]. Combined with the fact that structure is more conserved than sequence [83, 84], related binding sites have a high probability of being matched by comparing conserved structural motifs near the pockets.

This alone is not sufficient to build a robust tool that would surpass the current state-of-the-art methods in terms of its accuracy. But most current methods also struggle with unrelated binding sites, which most probably stems from the significant structural difference of unrelated sites mentioned above. However, to search in related proteins, this information could allow a search without the need for pre-indexing of the putative pockets, as we expect their resemblance to their homologs. Provided we have clusters of related and similar structures, only search across the cluster representatives would have to be carried out using the more sophisticated method, which is able to identify similarities even for unrelated proteins. Later, to identify similar cavities within the cluster, a method relying on the structural similarity could be used.

Effective matching of similar structures can be done in multiple ways, common is matching of secondary structure elements (as used in the mentioned TM-align [64]), however interesting and highly scalable approach is conversion to 'structural alphabet'. This is not a novel idea [85, 86, 87, 88], but it was recently very successfully applied in FoldSeek [89]. FoldSeek's alphabet differs from the traditional structural alphabets describing the backbone, as its 3D alphabet rather describes the interaction and geometric conformation of each residue with its

spatially closest residue. The authors of FoldSeek claim their alphabet has the following advantages: (i) Weaker dependency of consecutive letters; (ii) More even distribution of state frequencies, which should enhance the information density and reduce false positives; and, finally (iii) The highest information density in conserved protein cores and the lowest the non-conserved regions, whereas backbone structural alphabets seem to have this reversed [89].

This approach is expected to be particularly well suited for searching conserved substrings in related proteins. FoldSeek also provides a clustered version of the AlphaFold database [90], clustered by sequential and structural identity. This reduces the size of the AlphaFold database from over 200 million entries to over 50 million after sequential clustering and later down to 2.3 million after structural clustering and removal of fragments and singletons (clusters with only one structures).

Searching across the clusters would be done most efficiently by encoding each putative pocket by a numerical, easily searchable and indexable representation. That can be done using histogram- or fingerprinting-based methods or by using vectorizers based on machine learning, such as Site2Vec [56], currently also one of the best methods of assessing binding site similarity (based on the ProSPECCTs dataset [7]). These putative pockets would have to be located using cavity-detection algorithms.

Hence, the proposed method aims to provide faster than linear search in the related structures by locating spatially constrained structurally similar K-mers, which would not require screening most of the structures for the putative binding sites. However, complete development, thorough evaluation, and deployment of a final version of such a tool are beyond the scope of this work. The next sections focus on the proposal, implementation and testing of an approach meant as a proof-of-concept.

2.1 Method Implementation

2.1.1 Algorithm Overview

Given query pocket Q in a protein P, represented as a list of residue indices in the protein P, the algorithm shall find structurally conserved cavities in the pre-defined list of related proteins (given their structures). This pre-defined list is referred to as *a database*, where each protein's structure (and thus also the sequence) is stored.

To prevent confusion, multiple terms have to be defined first. *Query site, pocket, or cavity* mean the residues, that interact the ligand, along with their structure. *Conserved region, or substring*, is a continuous part of the protein

sequence, that had undergone minimal changes in time. *K-mer* is a substring of the protein sequence of defined length K (usually odd). *K-mer center* is the residue on the middle position of the K-mer sequence. *Alignment* of two structures, or point clouds, is their superposition, usually defined by specified rotation and translation of one of the structures.

The proposed method consists of four major steps for each search query: (i) Identifying proteins in the database, whose sequences contain similar regions (finding conserved regions); (ii) Locating sites enriched by multiple occurrences of the conserved regions (clustering); (iii) Fast optimal superposition of the found clusters with the query pocket and possible refinement of the mapping; (iv) Scoring the found match.

2.1.2 Identifying Sequences with Conserved Regions — K-mer Searching

Firstly, the searched sequences have to be prepared. However, we expect that the residues in the query pocket do not form a continuous stretch of the protein sequence. Thus, each position of the protein, that is in the query pocket, is treated as potentially isolated. A short substring of length K around every such position is extracted from the protein sequence and is called a K-mer. These short substrings are centered on the given position (for simplicity, positions at the ends of the protein sequence are ignored).

By searching only substrings of fixed length (the generated K-mers) in the databse, the initial filtering can be done very effectively by using a hash table with asymptotic complexity $\Theta(1)$ on average or direct indexing with asymptotic complexity of $O(1)$ in the worst case per one substring lookup in the hash table. To prepare this table, for every unique K-mer list of proteins containing this K-mer are stored. That can be done asymptotically linearly with respect to the total length of all proteins in the database.

However, the K-mers cannot be expected to be conserved perfectly, i.e., no change in the sequences occurred. In reality, it is expected that many changes might have happened, but the different sequences remained closely similar[1]. To find also these similar matches, we can query the hash table with many K-mers from a universum of similar K-mers:

$$U_K = \{K' \mid sim(K', K) \geq \zeta\} \tag{2.1}$$

[1]The subject of sequential similarity of proteins is a known and well studied topic and various methods of scoring the sequence similarity exist. Here, we expect no insertions or deletions to the short sequences and compute their score as a direct sum of similarity scores of residues on the corresponding positions given a scoring table.

where K is one K-mer, $sim(a, b)$ is a similarity scoring function (for natural sequences the BLOSUM 62 scoring matrix [91] was selected), and ζ is a parameter of the method.

Subsequently, only proteins from the database, whose sequences contain a certain number of the searched K-mers are reported and evaluated in the next step. For a hash table H, queried for substring s as $H[s]$, the element $H[s]$ is a list of identifiers of sequences containing the given K-mer, as can be seen in Algorithm 1.

Algorithm 1 Function to locate proteins with enough similar K-mers. Only distinct K-mers are counted, and proteins with less than τ distinct similar K-mers are filtered out. Hash table H on position $H[k]$ contains a list of protein identifiers marking proteins containing the exact match of K-mer k.

```
function FilterSequencesUsingKmers(Hash table H, List of searched K-mers
L_K, Similarity threshold ζ, Minimum K-mers τ)
    f ← an empty list of found protein identifiers
    o ← empty mapping maps identifiers to the number found unique K-mers
    for K-mer K ∈ L_K do
        U_K ← set of K-mers ζ-similar to K
        for K-mer K' ∈ U_K do
            Append all H[K'] to f
        end for
        for Identifier p ∈ f do
            if p ∈ o then
                Increment o[p] by one
            else
                set o[p] ← 1
            end if
        end for
        f ← emptylist
    end for
    r ← empty list of identifiers
    for Identifier p ∈ o do
        if o[P] ≥ τ then
            Add P to r
        end if
    end for
     return r
end function
```

As an optimization for the subsequent steps, the hash table can also map the

substrings to their respective positions in the original protein sequence, thus bypassing the K-mer search in the next step.

2.1.3 Locating Sites with Multiple Conserved Regions — Clustering

The similar K-mers for every positively evaluated protein from Step 1 are located. This steps focuses on clustering of the spatially close K-mer hits in each of the filtered structures from the database. The spatial regions in each of the positively evaluated protein structure from the database with a high density of conserved K-mers are located using a conventional clustering algorithm. Each conserved K-mer is represented as one 3D point (the center of mass of the residue at the sequential center of the K-mer). Ideally, the algorithm should locate sites where are many *different* conserved K-mers, so clusters that contain many K-mers that resemble only one of the query K-mers are ignored. However, that would be more complicated and it is not used in this implementation, as the generated false-positives should be filtered out in the next step.

Many different clustering algorithms exist, such as the commonly used K-means [92]. However, the requirement of knowledge of the number of clusters prior to the clustering is a limiting factor here. The only known information about the clusters is their approximate maximum size (approximately the size of the query cavity; the minimum cannot be set due to the potential loss of part of the cavity resulting from lack of conservation). The second requirement is the hard minimal limit of required similar K-mers. (The next phase requires at least 3 points. However, a higher hard limit could be set to lower the amount of potential false positives.)

A handful of various clustering algorithms were compared on a simple 2D datasets. Even on comparably easy datasets, some methods struggled with noisy data. Particularly interesting were DBSCAN [93], HDBSCAN [94] and OPTICS [95], which performed well on all the supplied datasets. Another evaluated option were hierarchical linkage clustering methods, as single or complete linkage clustering. These generally seemed to struggle more with noisy data. However, in this case the noisy data are to be expected, as similar K-mers might be randomly distributed in the protein sequences. Thus, these did not seem as an ideal option.

Based on the performance on the artificial datasets, OPTICS [95], an algorithm based on DBSCAN [93], was chosen for the clustering of the K-mers. It shows satisfactory performance even on real data, and its implementation is available in Scikit-learn (a library used for the clustering). However, the optimal clustering strategy should be more explored in future work.

OPTICS is a density-based clustering algorithm improving DBSCAN, mainly in the detection of meaningful clusters in noisy data of varying density. It has two parameters: ϵ, which describes the maximum distance for two points to be considered as neighbors, and *MinPts*, a minimum number of points capable of forming a cluster. These are set accordingly to the maximal distance of two points within the query cavity with a given margin μ. The *MinPts* is set to 3, as at least 3 points are required in the alignment step.

2.1.4 Query and Putative Binding Site Superposition

From the previous step, we end up with many clusters in various protein structures from the database. These clusters might be putative binding sites, some of which might be similar to the query pocket, some might not. At the same time, a coincidental cluster of similar K-mers (without any resemblance to the query pocket) is still possible. To discriminate between these cases and to rate the structural similarity between the query and the possible hit, the query pocket and the found cluster are superposed. This at the same time provides a useful visual feedback to the user, which itself is a very good indicator of the grossest errors (e.g., by visualizing the ligand binding to the query pocket in the alignment, it can be easily seen, whether the binding seems possible or not and how well do the structures correspond).

The superposition of the query and the potential pocket (a chance of finding a coincidental cluster has to be kept in mind) is done geometrically whilst respecting the found K-mer similarities. For the purposes of this section, the pockets can be represented by a set of points in the three-dimensional space. Also, we'll suppose, that if the pockets can be overlapped, there exists a mapping of corresponding points from the query to the points in the putative pocket. Superposition of the query pocket and the putative pocket (now two sets of points) is defined by a rotation and translation of the putative pocket, which results in an *optimal* overlap of the two pockets, i.e., rotation and translation that minimizes the RMSD [2] between the mapped points. Finding such rotation and translation is not as difficult, if we known the mapping between the cavities, and is briefly covered in section 2.1.4.

Optimal Superposition of Points

The actual problem of aligning two ordered sets of points (finding alignment that minimizes the RMSD between the corresponding points) is well-known and

[2]Root Mean Square Deviation, calculated as $\sqrt{|\text{Points}| \sum_{(p_a,p_b)\in\text{Points}} \|p_a - p_b\|^2}$ where p_a and p_b are the overlapped points represented as vectors in the 3D space. *Points* is a set of pairs of corresponding points.

explored. Many different solutions exist, the most notable are the Kabsch algorithm [96], which uses the *singular value decomposition* (SVD) of the correlation matrix to find the optimal rotation. Another interesting approach uses *quaternions* to find the optimal rotation as a leading eigenvector of a specific matrix [97]. Both methods have asymptotically linear complexity of $O(n)$, although the Kabsch algorithm seems to perform slightly better in terms of required computing time when executed on random structures [98].

Hereafter, the algorithm to find the optimal rotation and translation of two ordered sets of points to minimize the RMSD is referred to as $Superpose(A, B)$, where point A_i corresponds to point B_i.

Finding the Optimal Mapping

However, finding the best possible mapping is not a trivial task. The optimal mapping could be found by using, e.g., graph matching, as described in 1.2.2. However, probabilistic methods are used instead to improve the runtime significantly. The problem now is to find such mapping, that minimizes the RMSD of the mapped points after the optimal superposition. However, we also have to respect the properties of the residues, as once we map the points, we've also mapped the residues between the pockets. Instead of restricting the mapping of points only to other points, whose underlying residues are similar, we enforce that the K-mers surrounding these residues in the respective protein sequences have to also be similar. Thus, this mapping has to respect the given minimal similarity score $sim(kmer_a, kmer_b) \geq \zeta$ between the K-mers, which are extracted from the protein sequences around the mapped residues. We'll denote a function Φ, where $\Phi(a, b) = 1$ for points a, b ($a \in$ the query pocket, $b \in$ the putative pocket), whose underlying K-mers are ζ-similar, else 0.

We can leverage the required minimal similarity score to build a table of all possible mappings of points from the potential pocket to points in the query pocket, based on similarity comparisons of the K-mers around the residues identified with these points. Generation of this table can be done naively in $O(n^2)$, however, a more effective version of the algorithm that pre-generates all the K-mers similar to the query cavity K-mers and stores them in a hash table can be used, which lowers the asymptotic complexity down to $O(n)$ on average, where n is the number of residues in the potential cavity. The pre-generation of all the K-mers in the query pocket is done only once for all the searched structures. It can be done in $O(m)$, where m is the size of the query pocket (number of residues in the query pocket), and the asymptotic constant is dependent on ζ (depends on how many similar K-mers will be generated for each original K-mer). The pre-calculation of the hash table can be neglected for a high number of searches.

Identification of Shared Primitives

The algorithms that seek to identify the optimal mapping between the points (residues) in the query and putative pockets, are based on checking whether a certain geometrical constrains are fulfilled. A small subset of points of fixed length from each pocket, called a *primitive*, is selected first; similarity between the two selected primitives is evaluated and provided they *match* (are similar) the alignment is later extended or this information is used otherwise. As a primitive, one can consider e.g., a triangle, or a tetrahedron, from points in the pocket, which now represent vertices of the primitive. The primitive is thus represented by an ordered set of points from each pocket. The similarity is assessed while supposing that the i-th point from the first primitive corresponds to the i-th point from the second primitive. This section covers the methods used to examine the similarity of such primitives.

We'll work with two primitives, S, and H, of size n (they consist of n vertices) as an input. S_i denotes the i-th vertex of the primitive S, the same applies for the primitive H. The goal is to output 1, if these primitives are similar, else 0. We cannot expect a perfect geometrical match of the primitives, same as the geometrical methods mentioned in 1.2.2. Additionally, the vertices of the primitives correspond to the residues in the query and the putative pocket, and the similarity of these residues has to be respected. For this, the function Φ is utilized. It distinguishes the residues in the pockets, that can be mapped on each other based on the similarity of their underlying K-mers. If for any two vertices, the function $\Phi \neq 1$, the primitives are rejected as dissimilar. The second requirement is, that the overall shape of the primitives is similar (e.g., for primitives of size 4, we'd like to know, if the shape of the tetrahedrons is similar).

A commonly used solution for evaluating similarity of such three-dimensional polyhedrons (and triangles) is based on transforming the primitives to distance matrices[3] and later applying *spectral analysis* and measuring so-called *spectral distance* using the found eigenvalues of the distance matrices [99]. While this method is very interesting, it requires time $O(nk^3)$ to compare the first k eigenvalues for primitive with n vertices. Due to the required complexity, other more simple and faster methods were selected in this work.

These accept the given primitives as similar, provided the differences in the pairwise distances between the vertices are absolutely or relatively bound in a certain range. Suppose two distance matrices of the two compared primitives, $D^Q, D^H \in \mathbb{R}^{n \times n}$. The primitives will be accepted as similar, if the selected requirement is satisfied. Here, multiple different possible requirements are listed.

The most straightforward is the *maximal allowed absolute deviation* of differ-

[3] For a set of points P, distance matrix D has on position $D_{i,j}$ the distance between point P_i and P_j. Equivalently $D_{i,j} = \|P_i - P_j\|$ for all $0 \leq i, j < |P|$.

ences of the lengths of the edges of the primitives, see 2.2. δ is a parameter of the method.

$$\forall 0 \leq i, j < n \; : \; |D^Q_{i,j} - D^H_{i,j}| \leq \delta \tag{2.2}$$

Maximal allowed relative deviation normalizes the deviation in the lengths of the edges of the primitives by the length of the edge of the first (query) primitive, as in 2.3.

$$\forall 0 \leq i, j < n, i \neq j \; : \; \frac{|D^Q_{i,j} - D^H_{i,j}|}{D^Q_{i,j}} \leq \delta \tag{2.3}$$

The evaluation time for both of these is identical, as 2.3 can be rewritten as 2.4, where the matrix of minimal and maximal allowed distances can be pre-calculated and thus requires only two comparisons.

$$\nexists 0 \leq i, j < n, i \neq j \; : \; (1 - \delta) D^Q_{i,j} < D^H_{i,j} < (1 + \delta) D^Q_{i,j} \tag{2.4}$$

The maximal allowed absolute deviation has the drawback of preferring short distances, e.g., those found on the protein backbone. The relative, on the other hand, neglects the limitations imposed by the structures' resolutions. Also, neither takes into account all the deviations in that sense, that they do not penalize the primitives, that have all the deviations in the lengths of the edges large. Potentially, the limitation of the mean square error (MSE) or of the mean deviation is also possible, as in 2.5, which limits the mean relative error.

$$\frac{2}{n^2 - n} \sum_{i=0}^{n} \sum_{j=i+1}^{n} \frac{|D^Q_{i,j} - D^H_{i,j}|}{D^Q_{i,j}} \leq \delta \tag{2.5}$$

The specific used primitive matching is another parameter of the method. Hereafter, it will be referred to as a function $SimilarPrimitives(A, B, \Phi, \delta)$.

Random Consensus Algorithm

As the methods for finding the optimal superposition require knowledge of the mapping between the points, it is required to identify this mapping very quickly. One such potential mapping comes from mapping residues from the query pocket to those in the putative pocket, which share the similar K-mer around them. That is unfortunately not possible, as while lowering of the K-mer similarity threshold, ζ, many, not just one, K-mers from the putative pocket become similar to each K-mer from the query pocket.

Naively, one might try to consider all possible reasonable pairwise mappings between the points, however, such approach has a fatal flaw—in the case of multiple feasible mappings, which generate distinct rotations and translations, this approach tries to consider them all at once, subsequently generating nonsensical superposition of the points.

The random sample consensus algorithm [40] based on work by Martin A. Fischer and Robert C. Bolles is approaching the problem differently. Rather than trying to consider all the mappings at once, it uses the smallest possible randomly selected subset of points (a triangle) along with their mappings, and tries to expand it as much as possible. The selected triangle with the respective mapping is enough to distinguish, whether all the other points can be reasonably mapped to anything and if so, what mapping would fit the best. This is done iteratively for many different random starting subsets and the largest found mapping, called the consensus, is reported. The algorithm might terminate after specified number of rounds, or after the mapping is of a satisfactory size.

The implementation of the algorithm for the problem of finding the best mapping, along with the optimal superposition, is illustrated in Algorithm 2. The negative of this algorithm is its slower runtime. For each random triangle, it checks all other points along with their potential mappings. Although in theory, certain optimizations can be done to speed up this repeated checking. The resulting alignment and superposition can be later refined using the iterative closest point algorithm 2.1.4.

Iterative Closest Point

To refine the found superposition, the iterative closest point algorithm [41] by Paul J. Besl and Neail D. McKay is used. Given set of points A and B in some (not yet optimal) superposition specified by given rotation and translation, it iteratively maps each point in A to the closest point in the transformed set B[4], and then it recalculates the superposition of the two sets (hence, a new rotation and translation is generated).

Here, a modified version of the iterative closest point is used. It aims not just to refine the alignment of the already mapped points but also tries to extend the mapping by trying to map all the points from the first set to the closest points in the other set (after the transformation of the points using the found rotation and translation). However, only points closer than a certain threshold, ι, are considered.

For two sets of points A, B, rotation matrix Rot, translation vector $Trans$, the mapping function Φ (defined as in **??**), distance threshold ι, and termination thresh-

[4]By transformation is meant the application of rotation and translation to the given set of points.

Algorithm 2 A stochastic algorithm based on the random sample consensus paradigm. The algorithm seeks to find the largest possible subset of points from set A that correspond to points in B, along with their mappings, with respect to the parameter δ and mapping function Φ (constraints on which points can be mapped on each other). It returns the found mapping, rotation matrix *Rot*, translation vector *Trans*, and the *RMSD* of the alignment.

function RANDOMCONSENSUSALIGNMENT(Point set A, Point set B, Mapping function Φ, Primitive similarity cutoff δ, Satisfactory threshold t, Max Rounds R)
    best ← None
    $i \leftarrow 0$
    while $i < R$ and $|best| < t$ **do**
        model ← (triangle $a \subset A$, triangle $b \subset B$), that satisfies *SimilarPrimitive*(a, b, δ)
        consensus ← empty mapping
        for point a in $A \setminus a$ **do**
            for point b in $B \setminus b$ where $\Phi(a, b) == 1$ **do**
                expanded ← *model*
                expanded[a] ← b
                if *SimilarPrimitive*(expanded $\cap A$, expanded $\cap B, \delta$) **then** *. . . We check whether the mapping of point a to point b corresponds to the constraints imposed by the triangle*
                    consensus[a] ← b
                end if
            end for
        end for
        if $|consensus| > |best|$ **then**
            best ← consensus
        end if
        $i \leftarrow i + 1$
    end while
    Rot, *Trans*, *RMSD* ← Superpose(ordered keys from *best*, corresponding values from *best*)
    return best, Rot, Trans, RMSD
end function

old τ and the maximal number of rounds R, the algorithm can be implemented as in 3.

Algorithm 3 Variation of the iterative closest point algorithm, which tries to refine the superposition of the sets and expand the mapping as much as possible, while respecting the mapping constraints imposed by the function Φ. *Rot* is a rotation matrix, *Trans* is a translation vector. transform(B, Rot, Trans) is a function that applies the specified rotation and translation on all points in the set B. The function returns the found mapping, refined rotation matrix *Rot*, refined translation vector *Trans*, and the *RMSD* of the alignment.

function IterativeClosestPointRefinement(Point set A, Point set B, Rotation *Rot*, Translation *Trans*, Mapping function Φ, Distance threshold ι, Termination threshold θ, Max Rounds R)
    rmsd $\leftarrow$ Inf
    ClosestMappings $\leftarrow$ Empty mapping
    $i \leftarrow 0$
    while $i < R$ and $rmsd > \tau$ **do**
        transformed $\leftarrow$ transform(B, *Rot*, *Trans*)
        ClosestMappings $\leftarrow$ Empty mapping
        for Point $a \in A$ **do**
            closest $\leftarrow c \in$ transformed such as $\Phi(a, c) == 1$ and no other $d \in$ transformed for which $\Phi(a, d) == 1$ and $||a - d|| < ||a - c||$
            if $||\text{closest} - a|| \leq \iota$ **then**
                ClosestMappings[a] $\leftarrow c$
            end if
        end for
        Rot, *Trans*, *rmsd* $\leftarrow$ Superpose(ordered keys from *ClosestMappings*, corresponding values from *ClosestMappings*)
    end while
    return ClosestMappings, Rot, Trans, rmsd
end function

2.1.5 Scoring of Found Alignments

For the scoring of the found alignments, various different scoring functions and metrics were evaluated; see Figure 2.2. The most promising were different measures of overlap of the query binding site with the entire structure, which had potentially similar putative binding sites. This overlap is based on the rotation and translation found in the previous phase. On various datasets (the metrics were mostly inspected on the TOUGH-M1 dataset), one of the most discriminatory

metrics was the RMSD in an optimal alignment of residues (their centers of masses) from the query cavity and their closest counterparts from the putative cavity, while not enforcing any required similarity between the residues. More intricate approaches, combining various other metrics were inspected, however, to prevent the introduction of new parameters and potentially fitting them to one specific dataset, they were not used at the end.

The use of the full-site RMSD has the advantage of being easily interpretable while admittedly losing some information, particularly on harder datasets with more structurally similar positive and negative pairs (or even positive pairs being very dissimilar). The final score is counted as *pseudocounted inverse of the RMSD*, as in 2.6, where $0 < Score \leq 100$.

$$Score = \frac{1}{RMSD + 0.01} \tag{2.6}$$

2.1.6 Used Parameters

Many parameters were introduced during the method presentation; however, the best values were not identified. The values currently used for all the testing come from empirical observation and intuition rather than from a well-established analysis. Further inspection of these is required in the future.

K...3

K-mer similarity ζ...13

Scoring Table...BLOSUM 62

Min. K-mers τ...$\frac{1}{2}$ of query K-mers

Primitive Size...3 (triangle)

OPTICS - ϵ...1.2 × max q. dist. [5]

OPTICS - MinPts...3

Prim. Sim. Metric[6]...Mean Relative Error

Prim. Similarity - δ...0.15

Random Cons. Rounds...1000[7]

Sufficient Expansions...15[8]

ICP Close Point Cutoff...10 Å

ICP Rounds...3

2.1.7 Implementation Notes

The current implementation is written in Python to decrease the development time. The implementation itself is rather a proof-of-concept, as the work is still

[6]Metric used for the evaluation of primitive similarity.

[7]1000 tries to find a similar primitive, if found, it is then expanded.

[8]After finding 15 similar primitives in the random consensus algorithm, the execution is interrupted.

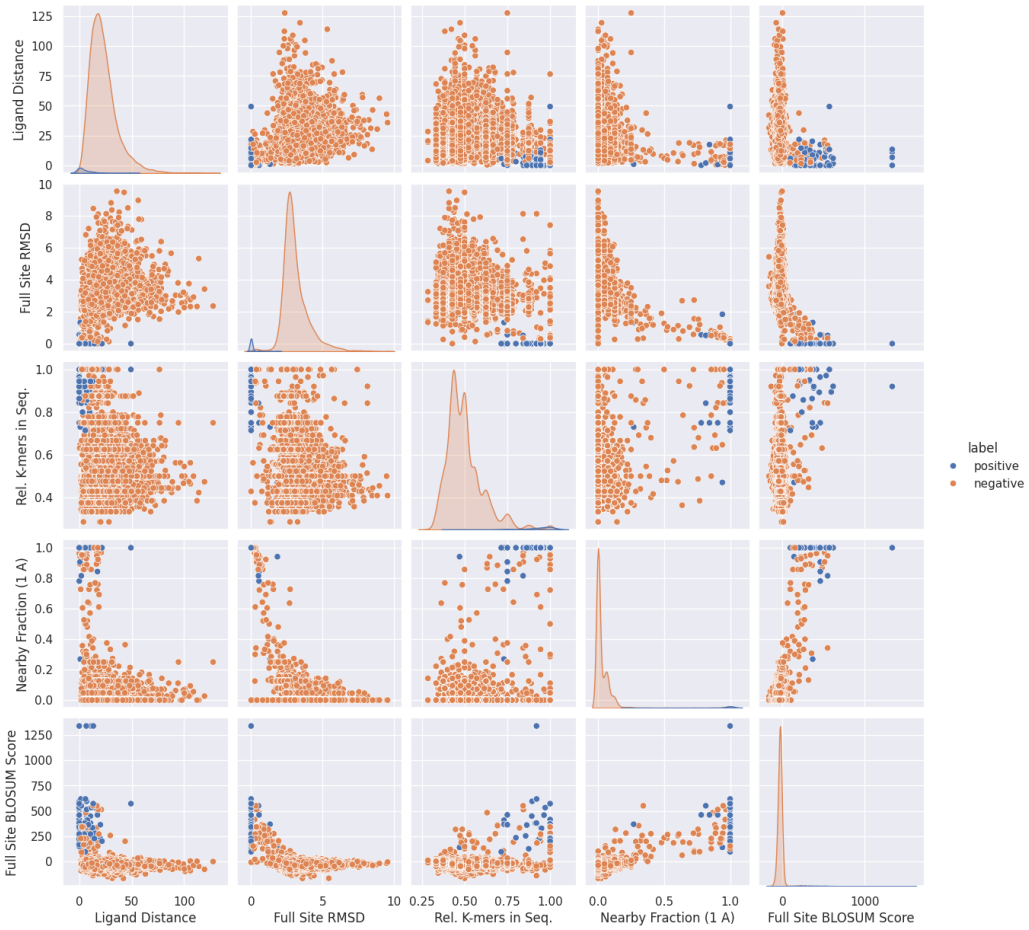

Figure 2.2 Considered Scoring Metrics on the Review Structures Dataset. Ligand distance is the measured distance of the actually binding ligands after the found rotation and translation and is reported purely for comparison, as it cannot be known without the knowledge of the actual binding ligands. The third column shows the number of similar K-mers found in the whole sequence of the potential match divided by the number of K-mers generated from the query. Nearby fraction shows how many residues from the query had a counterpart in the other structure within 1 Å normalized by the size of the query pocket. The BLOSUM Score was calculated by summing scores of the closest residues between the pockets in the BLOSUM 62 scoring matrix.

in the exploratory phase. Major optimizations and the most effective algorithms or approaches were not included in the current implementation, although they are partially described in the following sections, as the method's runtime is a key factor in its applicability. Minor optimizations are not mentioned.

The method currently uses natural protein sequences instead of artificial sequences over the structural alphabet. The implementation itself won't change; only the data inserted into the database (and the search query) need to be translated appropriately.

Used External Libraries

The following external Python libraries are used in the implementation:

NumPy. A comprehensive mathematical library covering random number generators, linear algebra, and more.

BioPython. Various tools for computational biology and bioinformatics. Used for loading of the PDB or mmCif structures and subsequent processing.

SciPy. A library providing many algorithms for scientific computing. Used for indexing 3D protein structures using KD-trees.

Scikit-learn. Library for machine learning in Python. Used for clustering of K-mers, evaluation and generation of machine-learning based scoring functions.

Matplotlib. Library for creating visualizations in Python. Used for figure generation.

Seaborn. Statistical data visualization library, used along matplotlib to generate figures and for data analysis.

Pandas. Library for data analysis and manipulation. Used for data analysis.

Code Availability

The latest version of the code is publicly accessible at `https://gitlab.mff.cuni.cz/telcerj/detectionofsimilarbindingsites` and is distributed under the MIT licence. Note that the tool is currently not meant for standalone use.

Chapter 3

Results

For evaluation of the method the ProSPECCTs benchmark was chosen. The results are rather preliminary as the parameter optimization was not carried out. Results on the same three datasets from the ProSPECCTs benchmark, as in section 1.3, are shown along with the analysis of the TOUGH-M1 dataset to explore the hypothetical performance on unrelated structures.

Usually, the performance of the binding site similarity estimation methods is measured by calculating so-called ROC cruve, or Receiver operating characteristic curve, which is often used to visualize the performance of a binary classifier (the reported similarity score of the two pockets can be seen as a measure of probability, that the pockets are similar). The ROC curve plots the true positive and false positive rates at various thresholds of the probability that the pockets are similar. For direct comparisons, the AUC, or area under the ROC curve, is calculated. The higher it is the more confident are the predictions of the classifier (AUC of 1 means that all the positive examples were correctly identified and no false positives were reported; AUC of 0.5 is equivalent to random choice). However, the performance evaluation of this method is not as straight forward as for methods that are based on the scoring of pairs and it is not directly comparable, as this method is not a binary classifier but a database searching tool which searches databases of unprocessed structures (where the putative binding sites are not known). As it applies a series of filters (only sequences with similar K-mers are selected, this is later filtered to only structures with sufficiently large spatial clusters of the K-mers, and these are later aligned and scored), it can happen that some pairs are not scored at all, if e.g., the protein sequence did not contain enough similar K-mers. Such entries are assigned default score of -1.

Also, as the putative binding sites are not known, the method tries to discover them. However, it can happen that multiple putative binding sites are found in one protein structure and all can be aligned and scored. In such cases, only the highest scoring match is kept for the given protein structure. Nevertheless,

even the high-scoring putative binding sites might still be incorrect as a different putative site, or not a binding site at all, might had been aligned instead. To eliminate this, for each reported hit the distance between center of mass of the ligand binding to the query site and the center of mass of the ligand binding to the structure containing the aligned cluster is measured, hereafter reffered to as *ligand distance*. Subsequently, ROC curves where hits whose superpositions generated higher then the specified ligand distance, are marked as false negatives. Each structure in the dataset contains only one ligand, which is also used for the delimitation of the query binding site in the distance of 4 Å.

Finally, the ROC curves are calculated as follows: for each pair of protein structures in the dataset, the pocket of the first structure is used as a query to search the database. Score of this pair is either the score of the second structure in the returned results or -1 if the second structure was not discovered. In the ROCs where the ligand distance is limited by a certain threshold (such as 1 Å), all positive pairs that were superposed with ligands farther apart than was the specified distance are marked as false negatives (receive a score of -1). Thus, these aren't counted in the true positives.

For the comparisons, the `random` module and `numpy.random`, were both seeded with 0.

3.1 Results on the ProSPECCTs Dataset

3.1.1 NMR Structures

Figure 3.1 shows the performance of the method on the NMR structures dataset. Similarly as for other tools tested on the ProSPECCTs dataset, the distances of the ligands after the optimal alignments were not checked. The confusion matrix can be seen in Figure 3.5. Out of the 7,729 total positive cases, 7,517 were reported as positive with 5,724 having the distance of centers of the aligned ligands below one Å. Out of the 100,512 negative pairs, 99.76 % received score -1 and were not aligned (either no sufficient mapping was found, or the sequences did not pass through the K-mer filter). Figure 3.2 shows the histogram of the ligand distance after the alignment.

For scoring, the pseudocounted inverse of the full site RMSD was used. The ability of the scoring metric to discriminate well-aligned pairs can be seen in Figure 3.4. The performance on the NMR dataset is directly comparable with the current state-of-the-art methods, such as FuzCav. It even outperforms RAPMAD, PocketMatch, Shaper, VolSite, ProBis, KRIPO, IsoMif and all other interaction based methods.

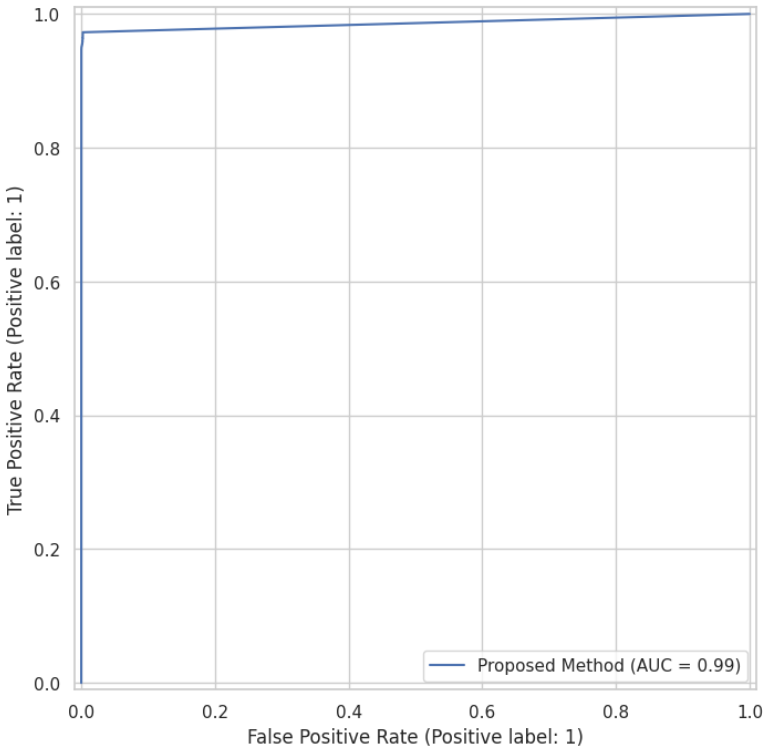

Figure 3.1 ROC curve of the proposed method with respect to the NMR structures dataset. The random consensus algorithm was used for the mapping. No restrictions on the maximal allowed ligand distance were imposed. For scoring, the pseudocounted inverse of full site RMSD was used.

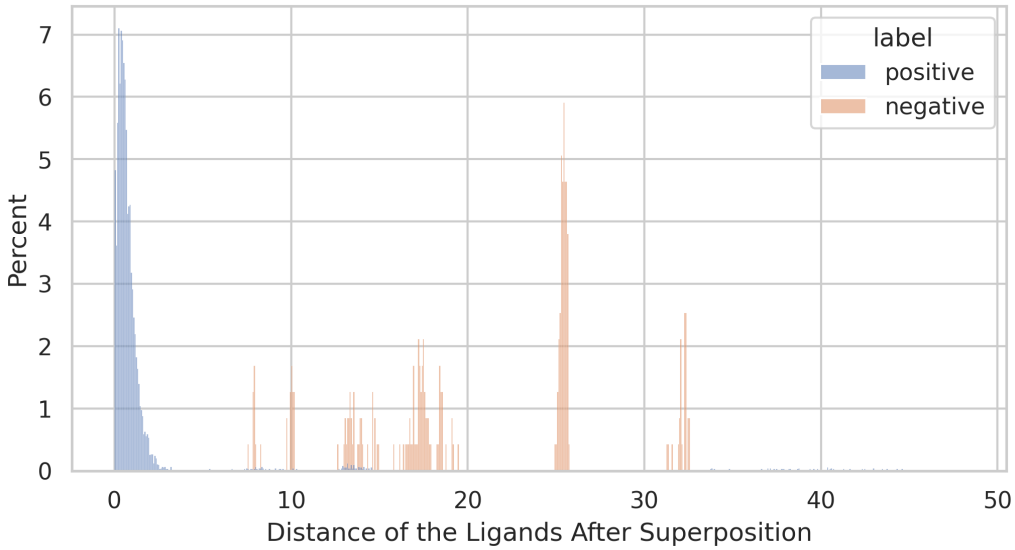

Figure 3.2 Histogram of binned distances of ligands of aligned pairs after applying the found rotation and translation with respect to the NMR dataset.

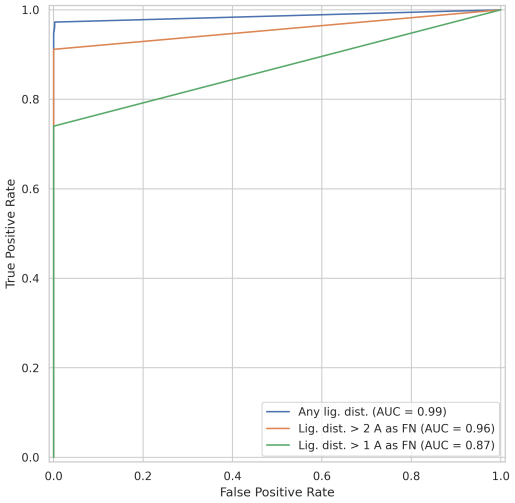

Figure 3.3 ROC curves for the proposed method on the NMR dataset, where found matches with high distance of the superposed ligands are labeled as false negatives.

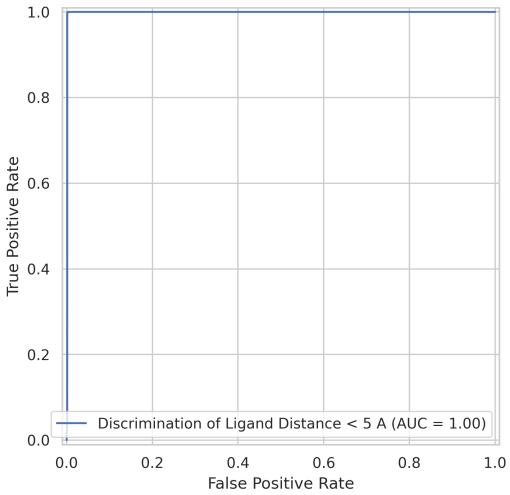

Figure 3.4 ROC curve when the classifying task is to discriminate superpositions of structures resulting in closely superposed ligands (ligand distance less than 5 Å) without any requirement on the binding site similarity.

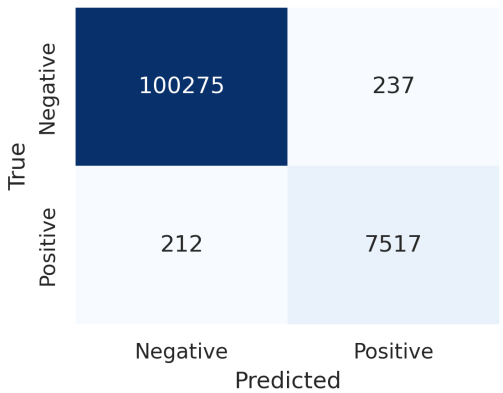

Figure 3.5 Confusion matrix on the NMR dataset. All matches with score at least 0 were taken as positive.

3.1.2 Kahraman Structures

Performance on the Kahraman structures is very poor. When labeling matches with distance of the superposed ligands more than 1 Å as false negatives, it decreases to the chance level, as shown in 3.6. Out of 1,320 positive pairs, only 158 were labeled as positive, with only 92 having ligand distance below 1 Å. Although the median ligand distance of the aligned positive pairs was 1.15×10^{-6} Å, that was likely due to the correct identification of identical pairs (in the Karhaman dataset, identical structures are labeled as positive pairs). The confusion matrix is in Figure 3.7. Despite the poor performance, the method is still comparable with methods such as RAPMAD, FuzCav or SiteAlign.

3.1.3 Review Structures

The AUC on the review structures dataset is significantly better than on the Kahraman structures, although not as good once the matches with high distance of the superposed ligands are labeled as false negatives, as can be seen in Figure 3.8. Out of 115 positive pairs, 67 were retrieved, while 48 had ligand distance under 1 Å. The confusion matrix, in Figure 3.9, shows that significantly more incorrect matches had higher than negative score (they were superposed). On this dataset, the method seemed to perform similarly as RAPMAD, Grim or Shaper.

However, while inspecting the negative hits, multiple pairs labeled as negative in the dataset generated exceptional alignments. Some resembled almost identical matches, as in Figure 3.10, some weren't as close, nonetheless the found alignment still seamed reasonable, as in Figure 3.11.

3.2 Runtime Analysis

The execution time of the searches was meassured using Python's `timeit` module, using the `timeit.default_timer`. All tests were performed on a single core of Intel(R) Core(TM) i7-10750H processor with base clock of 2.60GHz. Times for the three evaluated datasets are reported in Table 3.1.

Such a low runtimes make this method comparable to the fingerprinting and histogram based methods as all other, except TM-align, are more than 10 times slower per comparison. The most similar was the mentioned TM-align, which needs around 6 milliseconds per comparison. However, PocketMatch, the fastest of all methods in the ProSPECCTs benchmark, is more than 100 times faster.

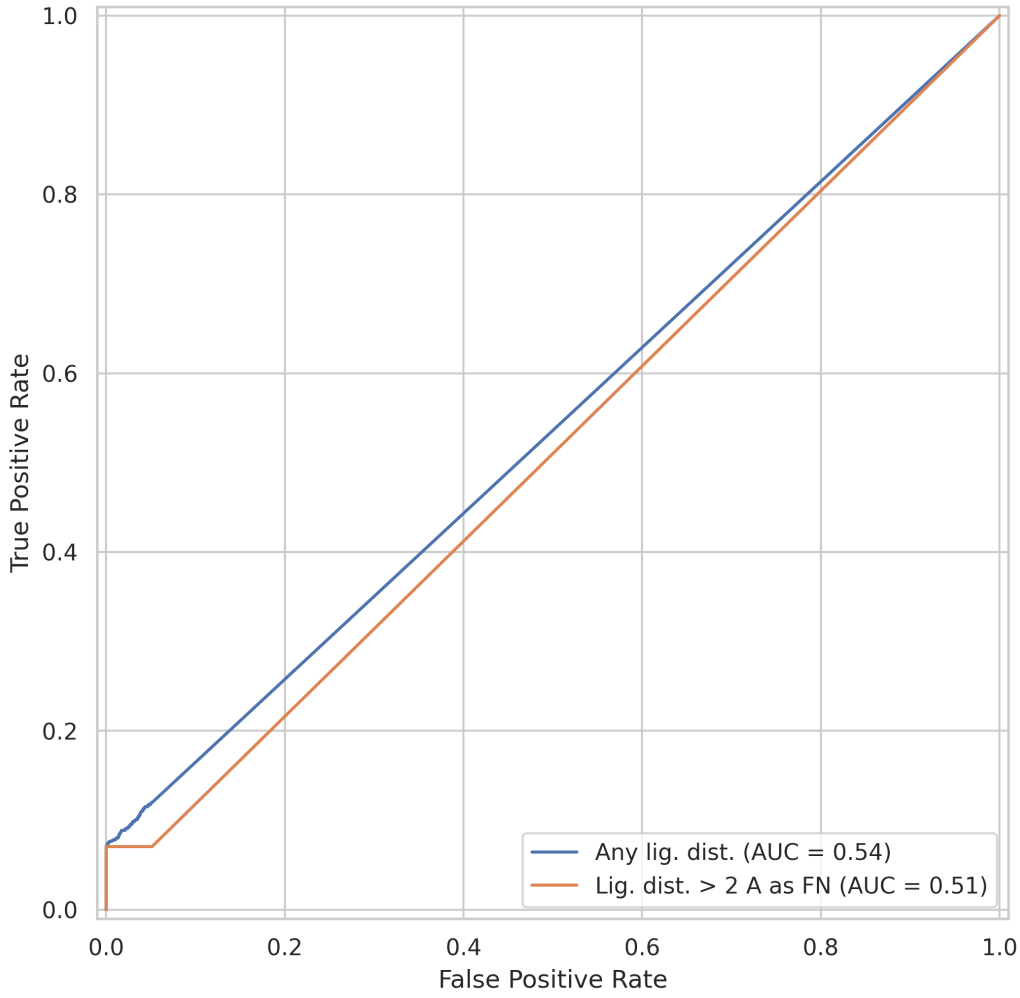

Figure 3.6 ROC curve of the classifier on the Kahraman structures dataset, where found matches with high distance of the superposed ligands are labeled as false negatives.

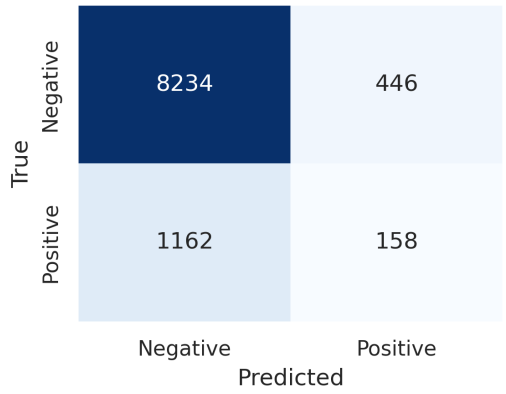

Figure 3.7 Confusion matrix on the Kahraman structures. All matches with score at least 0 were taken as positive.

Dataset	Elapsed Time (s)	Time Per Comparison (ms)
NMR Structures	220.391	2.036
Kahraman Structures	63.348	6.335
Review Structures	149.310	2.647
Average	-	3.673

Table 3.1 Runtimes of searches in the evaluation datasets. Elapsed times are sum of running times for all IDs that were the first in any active pair in the dataset. The tests were executed serially on a single CPU core. Times don't include the required time to load the query structure and isolate the cavity, as well as the time needed to initialize and create the database.

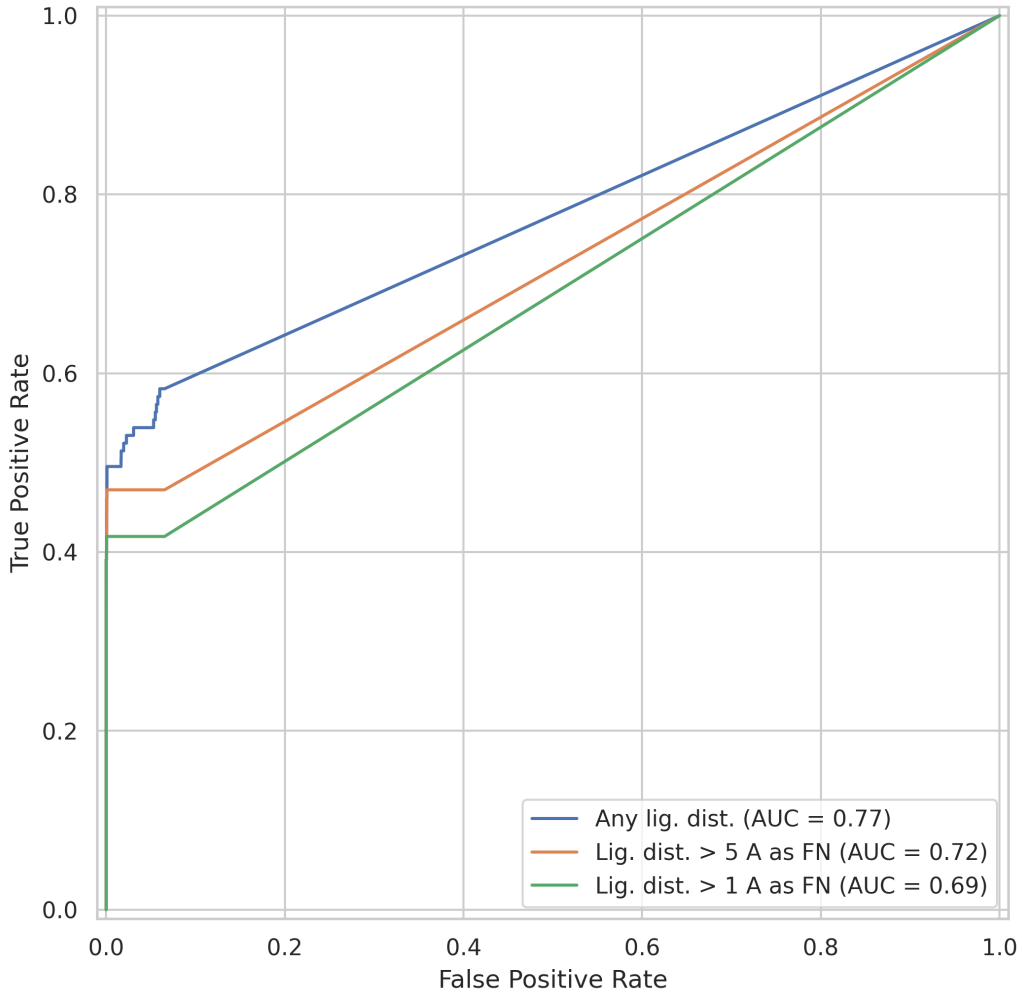

Figure 3.8 ROC curves on the review structures dataset. Orange and green curves show the ROC when matches with high distance of the superposed ligands are labeled as false negatives.

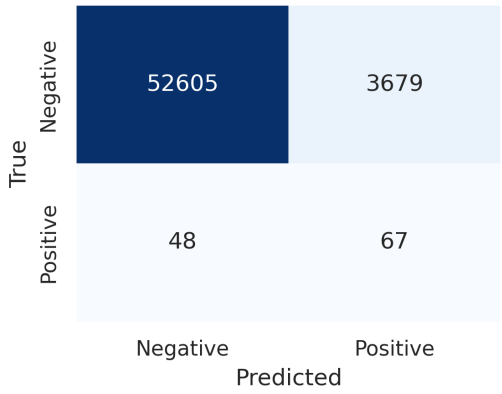

Figure 3.9 Confusion matrix on the review structures. All matches with score at least 0 were taken as positive.

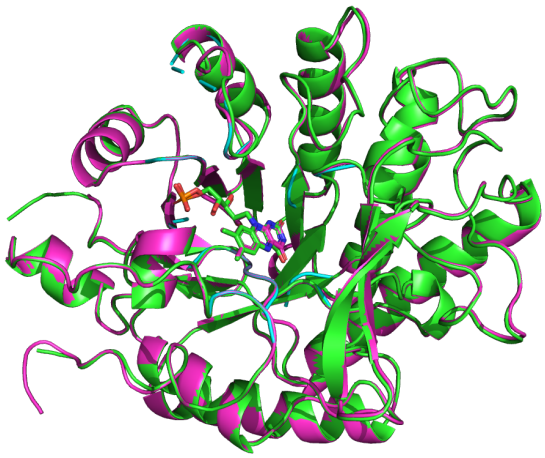

Figure 3.10 Chain A of 1gvr (green) aligned with chain A of 3p74 (magenta). Both bind flavin mononucleotide (FMN). More than half of the structures is mapped to the other (in cyan, cartoon drawing). Labeled as inactive pair with dissimilar binding sites in the ProSPECCTs dataset.

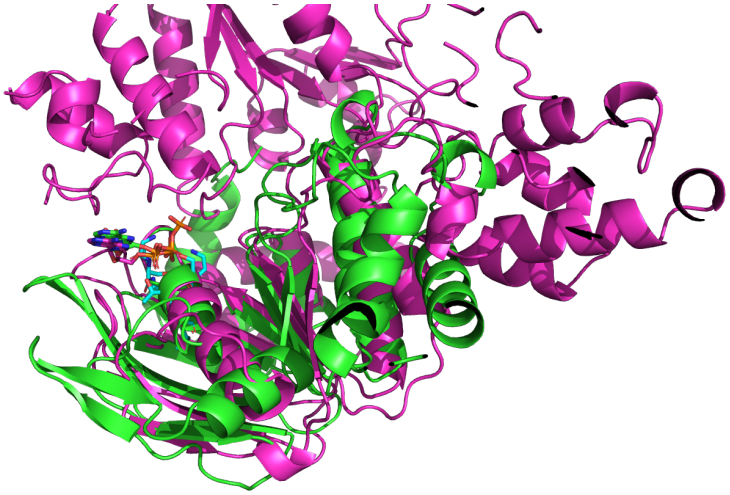

Figure 3.11 Reported alignment of chain A of 1b0u (green) and chain A of 3ux8 (magenta). Found mappings between the two are in cyan. While 1b0u binds ATP, 3ux8 binds ADP. Labeled as inactive pair with dissimilar binding sites in the ProSPECCTs dataset.

3.3 Analysis of the TOUGH-M1 Dataset

To inspect the similarities of pockets binding similar ligands, while differing in their sequence and structure, the TOUGH-M1 dataset was analyzed. The proposed searching method was able to reliably locate at most 10 % of the positive pairs, while generating significant amount of noise. To inspect the similarities and determine whether to blame the implementation of the method, or the nature of the problem, positive and negative pairs in the dataset were aligned based on an optimal alignment of their ligands.

For each of 100 randomly selected proteins, which had a binding site similar to at least one other binding site of different protein in the dataset, up to 100 of the other positive (similar) binding sites were selected randomly. The ligands of the similar pairs were then aligned using an evolutionary algorithm that tried to minimize the RMSD between the ligands and the number of atom type mismatches, while trying to map the entire smaller structure to the larger one. This was repeated for 50 structures participating in negative pairs, along with their dissimilar binding sites. This way, 4,677 similar binding sites pairs and 879 dissimilar, were aligned. In this alignment, multiple features were calculated. Such as, the total BLOSUM 62 score of the nearest residues from the second pocket to residues of the first pocket, the fraction of residues from the first pocket having any residue from the second structure nearby (with cutoff values of 1, 2, 3, 4, and 5 Å), the total number of K-mers of length K and similarity at least s around

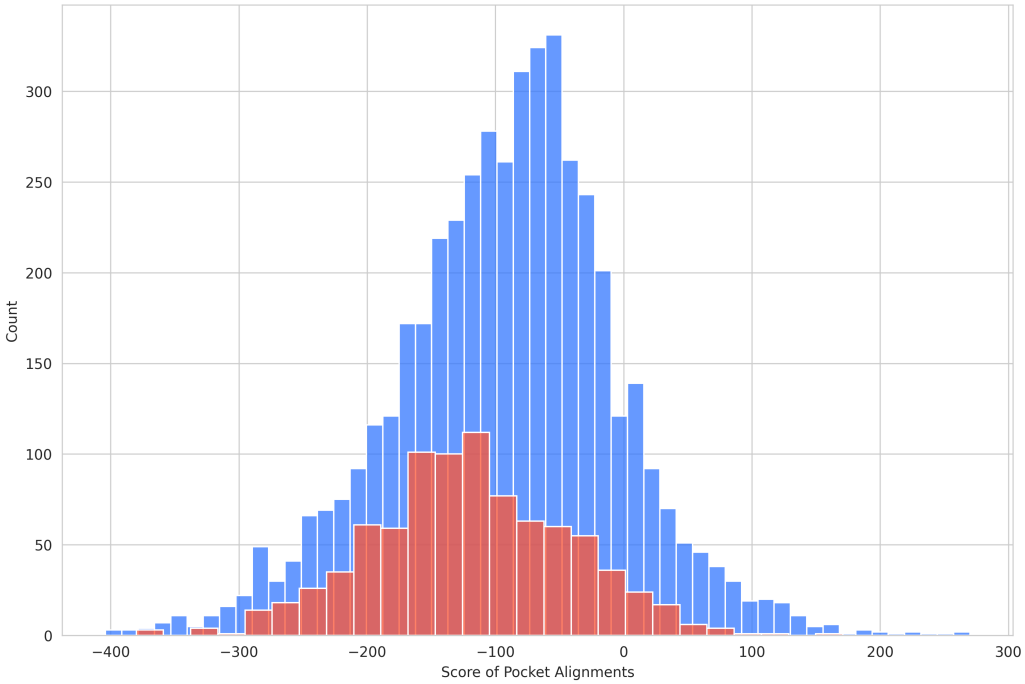

Figure 3.12 Total score of the closest residues between the two pockets after the alignment. Scores of negative pairs (dissimilar pockets) are in red, scores of positive pairs (similar pockets) are in blue.

residue from the first pocket and it's closest residue from the second pocket, the total number of similar K-mers of specified length and minimal similarity, and the total number of such K-mers of minimal similarity that were no farther then 4 Å apart in the found alignment.

Figure 3.12 shows the distribution of the scores in the cavities. The medium number of shared K-mers on close positions in the alignment is shown in Figure 3.13.

To compare the distributions of the total number of shared K-mers between similar and dissimilar binding sites, the tables of the total number of shared close K-mers were summed to calculate the percentile distributions. Figure 3.14 shows the corresponding percentiles.

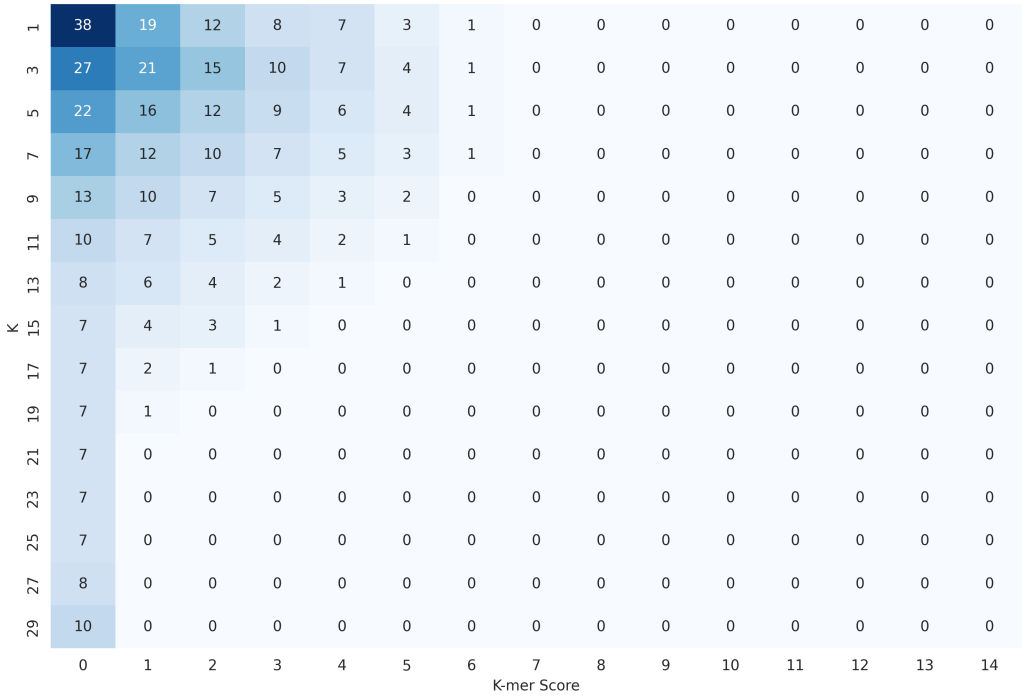

Figure 3.13 Median count of K-mers with their center residues being the closest ones in the alignment of positive pairs. On each position is the total number of K-mers of length specified by the vertical axis and with minimal BLOSUM 62 score specified by the horizontal axis. Such table was calculated for each pair and the median table for positive pairs is reported.

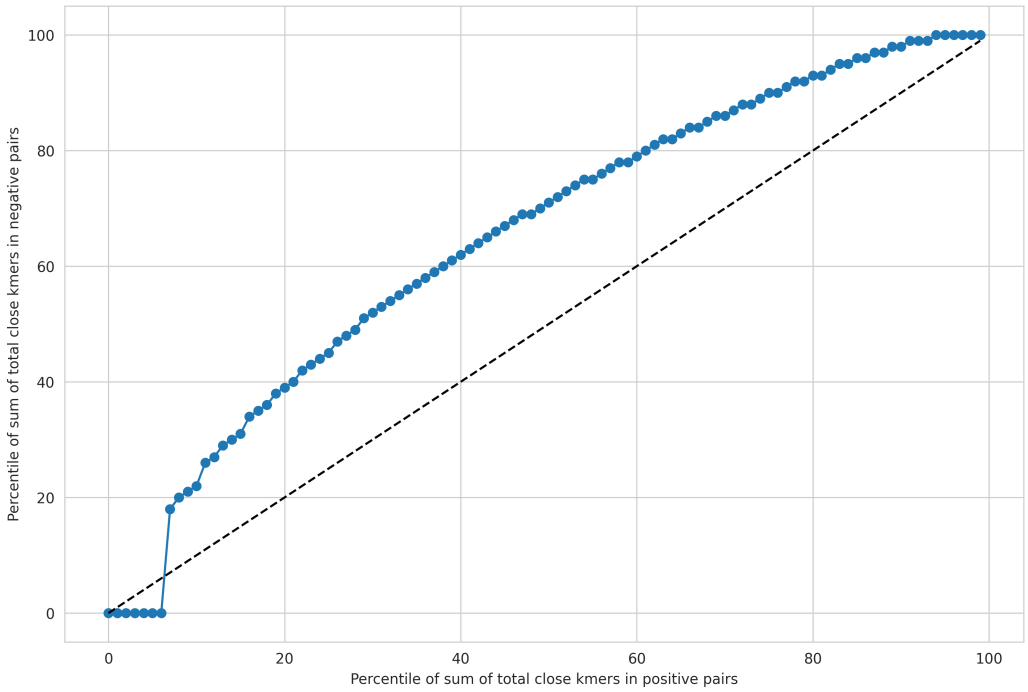

Figure 3.14 Corresponding percentiles of the total number of shared K-mers of all lengths and non-negative similarities between positive and negative pairs. For example, 50 on the horizontal axis corresponds to roughly 70 on the vertical, thus, the 70th percentile of the sum of shared K-mers in negative pairs corresponds to the 50th percentile of shared K-mers in positive pairs.

Chapter 4

Discussion

The optimistic results on the NMR dataset, where the method outperformed methods as RAPMAD, PocketMatch, or IsoMIF, and the dataset of successful applications (review structures) indicate a certain potential of the proposed method. While the performance on the Kahraman structures and the TOUGH-M1 dataset is rather underwhelming, it should be noted that many other methods do struggle with these datasets as well. One of the underlying reasons might be the frequent significant difference in the structure of unrelated binding sites, even though the same or similar ligand is bound, as described by Barelier et al. [61]. Once we inspect the similarities in the shared K-mers, the TOUGH-M1 datasets provides a similar point of view, as to reject at least 60 % of the negatives and find 60 % of the positives, a certain point where the dissimilar binding sites begin to have more shared K-mers with the query than the actual similar binding sites on close position is reached. However, this method is not intended for searching unrelated or very distant binding sites. The high early enrichment, even on the dataset of review structures, indicates that similar structures are located very easily, while the more distant ones are rather not found at all using this method. The reported runtime is also very promising, as, without any optimizations running in pure Python, apart from the clustering and linear algebra functions, the time is in single milliseconds per comparison. Such speeds are only achieved by TM-align, and the fingerprinting or histogram based methods. That is particularly promising, mainly in terms of large-scale database searching.

The most notable discovery is that such a simple method, without any knowledge of the actual binding sites, can confidently locate and match them and, on the easier datasets, be almost equivalent to other currently used methods.

However, the question of the required similarity remained unanswered. If the required clusters of similar sequences would have to be of a minimal size, then all the advantages of this method would disappear. Also, highly similar sequences can already be searched more effectively using conventional sequential

search. A more thorough examination of this is required to decide on the method's applicability. Another area of future interest should be the optimization of various parameters, translation of the sequences to FoldSeek's structural alphabet, and possibilities of parallelization.

Conclusion

In this work, the current state-of-the-art approaches for the evaluation of binding site similarity were discussed, along with their weaknesses—such as the costly requirement of screening the whole database for putative binding sites and sometimes even the need for all versus all pairwise comparisons. A simple method for the identification of related binding sites based on locating the persisting sequential and structural similarity was presented, along with preliminary results on three testing datasets.

Despite the inferior performance on the hard datasets, the method is comparable, or on certain datasets even better, than current state-of-the-art methods for assessing binding site similarity, even without the exact knowledge of the location of the putative binding sites. The results show a certain potential worth exploring in the future. However, one should remain cautious, as many questions were left unanswered. More thorough testing and evaluation of the method, using a wider range of parameters, is still required.

In conclusion, the performance of the proposed method encourages a slight optimism and proves this area is worth more attention in the future.

Bibliography

[1] Maria Cristina De Rosa, Rituraj Purohit, and Alfonso T García-Sosa. "Drug repurposing: a nexus of innovation, science, and potential". In: *Scientific Reports* 13.1 (2023), p. 17887.

[2] V Joachim Haupt, Simone Daminelli, and Michael Schroeder. "Drug promiscuity in PDB: protein binding site similarity is key". In: *PLoS one* 8.6 (2013), e65894.

[3] György Abrusán and Joseph A Marsh. "Ligand binding site structure influences the evolution of protein complex function and topology". In: *Cell reports* 22.12 (2018), pp. 3265–3276.

[4] Britta Nisius, Fan Sha, and Holger Gohlke. "Structure-based computational analysis of protein binding sites for function and druggability prediction". In: *Journal of biotechnology* 159.3 (2012), pp. 123–134.

[5] Janez Konc. "Binding site comparisons for target-centered drug discovery". In: *Expert Opinion on Drug Discovery* 14.5 (2019), pp. 445–454.

[6] Miquel Duran-Frigola et al. "Extending the small-molecule similarity principle to all levels of biology with the Chemical Checker". In: *Nature Biotechnology* 38.9 (2020), pp. 1087–1096.

[7] Christiane Ehrt, Tobias Brinkjost, and Oliver Koch. "A benchmark driven guide to binding site comparison: An exhaustive evaluation using tailor-made data sets (ProSPECCTs)". In: *PLoS computational biology* 14.11 (2018), e1006483.

[8] Franck Da Silva et al. "Exhaustive repertoire of druggable cavities at protein–protein interfaces of known three-dimensional structure". In: *Journal of Medicinal Chemistry* 62.21 (2019), pp. 9732–9742.

[9] Jérémy Desaphy et al. "sc-PDB: a 3D-database of ligandable binding sites—10 years on". In: *Nucleic acids research* 43.D1 (2015), pp. D399–D404.

[10] Franck Da Silva, Jeremy Desaphy, and Didier Rognan. "IChem: a versatile toolkit for detecting, comparing, and predicting protein–ligand interactions". In: *ChemMedChem* 13.6 (2018), pp. 507–510.

[11] Merveille Eguida and Didier Rognan. “Estimating the similarity between protein pockets”. In: *International Journal of Molecular Sciences* 23.20 (2022), p. 12462.

[12] David G Levitt and Leonard J Banaszak. “POCKET: a computer graphies method for identifying and displaying protein cavities and their surrounding amino acids”. In: *Journal of molecular graphics* 10.4 (1992), pp. 229–234.

[13] Andrea Volkamer et al. “DoGSiteScorer: a web server for automatic binding site prediction, analysis and druggability assessment”. In: *Bioinformatics* 28.15 (2012), pp. 2074–2075.

[14] Jérémy Desaphy et al. *Comparison and druggability prediction of protein–ligand binding sites from pharmacophore-annotated cavity shapes*. 2012.

[15] Jean-Rémy Marchand et al. “CAVIAR: a method for automatic cavity detection, description and decomposition into subcavities”. In: *Journal of Computer-Aided Molecular Design* 35.6 (2021), pp. 737–750.

[16] Kuan Pern Tan, Raghavan Varadarajan, and Mallur S Madhusudhan. “DEPTH: a web server to compute depth and predict small-molecule binding cavities in proteins”. In: *Nucleic acids research* 39.suppl_2 (2011), W242–W248.

[17] Jian Yu et al. “Roll: a new algorithm for the detection of protein pockets and cavities with a rolling probe sphere”. In: *Bioinformatics* 26.1 (2010), pp. 46–52.

[18] Sebastian Schneider and Martin Zacharias. “Combining geometric pocket detection and desolvation properties to detect putative ligand binding sites on proteins”. In: *Journal of Structural Biology* 180.3 (2012), pp. 546–550.

[19] Chi-Ho Ngan et al. “FTSite: high accuracy detection of ligand binding sites on unbound protein structures”. In: *Bioinformatics* 28.2 (2012), pp. 286–287.

[20] Carine Berezin et al. “ConSeq: the identification of functionally and structurally important residues in protein sequences”. In: *Bioinformatics* 20.8 (2004), pp. 1322–1324.

[21] Radoslav Krivák and David Hoksza. “P2Rank: machine learning based tool for rapid and accurate prediction of ligand binding sites from protein structure”. In: *Journal of cheminformatics* 10 (2018), pp. 1–12.

[22] Rishal Aggarwal et al. “DeepPocket: ligand binding site detection and segmentation using 3D convolutional neural networks”. In: *Journal of Chemical Information and Modeling* 62.21 (2021), pp. 5069–5079.

[23] Jeevan Kandel, Hilal Tayara, and Kil To Chong. “PUResNet: prediction of protein-ligand binding sites using deep residual neural network”. In: *Journal of cheminformatics* 13 (2021), pp. 1–14.

[24] José Jiménez et al. “DeepSite: protein-binding site predictor using 3D-convolutional neural networks”. In: *Bioinformatics* 33.19 (2017), pp. 3036–3042.

[25] Xu Yan et al. “PointSite: a point cloud segmentation tool for identification of protein ligand binding atoms”. In: *Journal of Chemical Information and Modeling* 62.11 (2022), pp. 2835–2845.

[26] Vincent Le Guilloux, Peter Schmidtke, and Pierre Tuffery. “Fpocket: an open source platform for ligand pocket detection”. In: *BMC bioinformatics* 10 (2009), pp. 1–11.

[27] Manfred Hendlich, Friedrich Rippmann, and Gerhard Barnickel. “LIGSITE: automatic and efficient detection of potential small molecule-binding sites in proteins”. In: *Journal of Molecular Graphics and Modelling* 15.6 (1997), pp. 359–363.

[28] Jingtian Zhao, Yang Cao, and Le Zhang. “Exploring the computational methods for protein-ligand binding site prediction”. In: *Computational and structural biotechnology journal* 18 (2020), pp. 417–426.

[29] Thorben Reim et al. “SiteMine: Large-scale binding site similarity searching in protein structure databases”. In: *Archiv der Pharmazie* (2024), e2300661.

[30] Esther Kellenberger, Claire Schalon, and Didier Rognan. “How to measure the similarity between protein ligand-binding sites?” In: *Current Computer-Aided Drug Design* 4.3 (2008), p. 209.

[31] Kalidas Yeturu and Nagasuma Chandra. “PocketMatch: a new algorithm to compare binding sites in protein structures”. In: *BMC bioinformatics* 9 (2008), pp. 1–17.

[32] Camille-Georges Wermuth et al. “Glossary of terms used in medicinal chemistry (IUPAC Recommendations 1998)”. In: *Pure and applied Chemistry* 70.5 (1998), pp. 1129–1143.

[33] Tianyun Liu and Russ B Altman. “Using multiple microenvironments to find similar ligand-binding sites: application to kinase inhibitor binding”. In: *PLoS computational biology* 7.12 (2011), e1002326.

[34] Timo Krotzky et al. “Large-scale mining for similar protein binding pockets: with RAPMAD retrieval on the fly becomes real”. In: *Journal of chemical information and modeling* 55.1 (2015), pp. 165–179.

[35] Christiane Ehrt, Tobias Brinkjost, and Oliver Koch. “Impact of binding site comparisons on medicinal chemistry and rational molecular design”. In: *Journal of medicinal chemistry* 59.9 (2016), pp. 4121–4151.

[36] Brice Hoffmann et al. “A new protein binding pocket similarity measure based on comparison of clouds of atoms in 3D: application to ligand prediction”. In: *BMC bioinformatics* 11 (2010), pp. 1–16.

[37] Mu Gao and Jeffrey Skolnick. “APoc: large-scale identification of similar protein pockets”. In: *Bioinformatics* 29.5 (2013), pp. 597–604.

[38] Clyde L Monma. “Algorithms and software for optimization”. In: *(No Title)* (1986).

[39] Merveille Eguida and Didier Rognan. “A computer vision approach to align and compare protein cavities: application to fragment-based drug design”. In: *Journal of Medicinal Chemistry* 63.13 (2020), pp. 7127–7142.

[40] Martin A Fischler and Robert C Bolles. “Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography”. In: *Communications of the ACM* 24.6 (1981), pp. 381–395.

[41] P Besl. “A Method forRegisttation of 3-D Shapes”. In: *Trans. PAMI* 14.2 (1992).

[42] Coen Bron and Joep Kerbosch. “Algorithm 457: finding all cliques of an undirected graph”. In: *Communications of the ACM* 16.9 (1973), pp. 575–577.

[43] Etsuji Tomita, Akira Tanaka, and Haruhisa Takahashi. “The worst-case time complexity for generating all maximal cliques and computational experiments”. In: *Theoretical computer science* 363.1 (2006), pp. 28–42.

[44] HC Johnston. “Cliques of a graph-variations on the Bron-Kerbosch algorithm”. In: *International Journal of Computer & Information Sciences* 5.3 (1976), pp. 209–238.

[45] Randy Carraghan and Panos M Pardalos. “An exact algorithm for the maximum clique problem”. In: *Operations Research Letters* 9.6 (1990), pp. 375–382.

[46] Eleanor J Gardiner, Peter J Artymiuk, and Peter Willett. “Clique-detection algorithms for matching three-dimensional molecular structures”. In: *Journal of Molecular Graphics and Modelling* 15.4 (1997), pp. 245–253.

[47] Daniel Kuhn et al. “Functional classification of protein kinase binding sites using Cavbase”. In: *ChemMedChem: Chemistry Enabling Drug Discovery* 2.10 (2007), pp. 1432–1447.

[48] Matthieu Chartier and Rafael Najmanovich. "Detection of binding site molecular interaction field similarities". In: *Journal of chemical information and modeling* 55.8 (2015), pp. 1600–1615.

[49] Hans-Dieter Holtje et al. *Molecular modeling*. Vol. 5. Wiley-VCH Weinheim, Germany, 2003.

[50] Bing Xiong et al. "BSSF: a fingerprint based ultrafast binding site similarity search and function analysis server". In: *BMC bioinformatics* 11 (2010), pp. 1–11.

[51] David J Wood et al. "Pharmacophore fingerprint-based approach to binding site subpocket similarity and its application to bioisostere replacement". In: *Journal of chemical information and modeling* 52.8 (2012), pp. 2031–2043.

[52] Nathanaël Weill and Didier Rognan. "Alignment-free ultra-high-throughput comparison of druggable protein- ligand binding sites". In: *Journal of chemical information and modeling* 50.1 (2010), pp. 123–135.

[53] T Andrew Binkowski and Andrzej Joachimiak. "Protein functional surfaces: global shape matching and local spatial alignments of ligand binding sites". In: *BMC structural biology* 8 (2008), pp. 1–23.

[54] Limeng Pu et al. "DeepDrug3D: classification of ligand-binding pockets in proteins with a convolutional neural network". In: *PLoS computational biology* 15.2 (2019), e1006718.

[55] Martin Simonovsky and Joshua Meyers. "DeeplyTough: learning structural comparison of protein binding sites". In: *Journal of chemical information and modeling* 60.4 (2020), pp. 2356–2366.

[56] Arnab Bhadra and Kalidas Yeturu. "Site2Vec: a reference frame invariant algorithm for vector embedding of protein–ligand binding sites". In: *Machine Learning: Science and Technology* 2.1 (2020), p. 015005.

[57] Yu-Chen Chen et al. "Prediction of protein pairs sharing common active ligands using protein sequence, structure, and ligand similarity". In: *Journal of chemical information and modeling* 56.9 (2016), pp. 1734–1745.

[58] Kang Peng, Zoran Obradovic, and Slobodan Vucetic. "Exploring bias in the Protein Data Bank using contrast classifiers". In: *Biocomputing 2004*. World Scientific, 2003, pp. 435–446.

[59] Lieyang Chen et al. "Hidden bias in the DUD-E dataset leads to misleading performance of deep learning in structure-based virtual screening". In: *PloS one* 14.8 (2019), e0220113.

[60] Abdullah Kahraman et al. "Shape variation in protein binding pockets and their ligands". In: *Journal of molecular biology* 368.1 (2007), pp. 283–301.

[61] Sarah Barelier et al. “The recognition of identical ligands by unrelated proteins”. In: *ACS chemical biology* 10.12 (2015), pp. 2772–2784.

[62] Rajiv Gandhi Govindaraj and Michal Brylinski. “Comparative assessment of strategies to identify similar ligand-binding pockets in proteins”. In: *BMC bioinformatics* 19 (2018), pp. 1–17.

[63] Dávid Bajusz, Anita Rácz, and Károly Héberger. “Why is Tanimoto index an appropriate choice for fingerprint-based similarity calculations?” In: *Journal of cheminformatics* 7 (2015), pp. 1–13.

[64] Yang Zhang and Jeffrey Skolnick. “TM-align: a protein structure alignment algorithm based on the TM-score”. In: *Nucleic acids research* 33.7 (2005), pp. 2302–2309.

[65] Fabian Paul and Thomas R Weikl. “How to distinguish conformational selection and induced fit based on chemical relaxation rates”. In: *PLoS computational biology* 12.9 (2016), e1005067.

[66] Jose Batista et al. “SiteHopper-a unique tool for binding site comparison”. In: *Journal of Cheminformatics* 6.Suppl 1 (2014), P57.

[67] Noé Sturm et al. “Structural insights into the molecular basis of the ligand promiscuity”. In: *Journal of chemical information and modeling* 52.9 (2012), pp. 2410–2421.

[68] Oliver B Scott, Jing Gu, and AW Edith Chan. “Classification of protein-binding sites using a spherical convolutional neural network”. In: *Journal of Chemical Information and Modeling* 62.22 (2022), pp. 5383–5396.

[69] Alexandra Shulman-Peleg, Ruth Nussinov, and Haim J Wolfson. “SiteEngines: recognition and comparison of binding sites and protein–protein interfaces”. In: *Nucleic acids research* 33.suppl_2 (2005), W337–W341.

[70] Matthias Leinweber et al. “CavSimBase: a database for large scale comparison of protein binding sites”. In: *IEEE Transactions on Knowledge and Data Engineering* 28.6 (2016), pp. 1423–1434.

[71] Matthieu Chartier, Etienne Adriansen, and Rafael Najmanovich. “IsoMIF Finder: online detection of binding site molecular interaction field similarities”. In: *Bioinformatics* 32.4 (2016), pp. 621–623.

[72] Praveen Anand et al. “PLIC: protein–ligand interaction clusters”. In: *Database* 2014 (2014), bau029.

[73] Janez Konc and Dušanka Janežič. “ProBiS-2012: web server and web services for detection of structurally similar binding sites in proteins”. In: *Nucleic acids research* 40.W1 (2012), W214–W221.

[74] Janez Konc and Dušanka Janežič. “ProBiS algorithm for detection of structurally similar protein binding sites by local structural alignment”. In: *Bioinformatics* 26.9 (2010), pp. 1160–1168.

[75] Bernard R Brooks et al. “CHARMM: the biomolecular simulation program”. In: *Journal of computational chemistry* 30.10 (2009), pp. 1545–1614.

[76] Joel Graef et al. “Searching geometric patterns in protein binding sites and their application to data mining in protein kinase structures”. In: *Journal of Medicinal Chemistry* 65.2 (2021), pp. 1384–1395.

[77] Yuko Tsuchiya et al. “PoSSuM v. 3: A Major Expansion of the PoSSuM Database for Finding Similar Binding Sites of Proteins”. In: *Journal of Chemical Information and Modeling* 63.23 (2023), pp. 7578–7587.

[78] Yasuo Tabei and Koji Tsuda. “Sketchsort: Fast all pairs similarity search for large databases of molecular fingerprints”. In: *Molecular informatics* 30.9 (2011), pp. 801–807.

[79] Takeshi Kawabata. “Detection of cave pockets in large molecules: Spaces into which internal probes can enter, but external probes from outside cannot”. In: *Biophysics and physicobiology* 16 (2019), pp. 391–406.

[80] Christiane Ehrt, Tobias Brinkjost, and Oliver Koch. “Binding site characterization–similarity, promiscuity, and druggability”. In: *Medchemcomm* 10.7 (2019), pp. 1145–1159.

[81] Anna R Panchenko, Fyodor Kondrashov, and Stephen Bryant. “Prediction of functional sites by analysis of sequence and structure conservation”. In: *Protein science* 13.4 (2004), pp. 884–892.

[82] John A Capra and Mona Singh. “Predicting functionally important residues from sequence conservation”. In: *Bioinformatics* 23.15 (2007), pp. 1875–1882.

[83] Kristoffer Illergård, David H Ardell, and Arne Elofsson. “Structure is three to ten times more conserved than sequence—a study of structural response in protein cores”. In: *Proteins: Structure, Function, and Bioinformatics* 77.3 (2009), pp. 499–508.

[84] Cyrus Chothia and Arthur M Lesk. “The relation between the divergence of sequence and structure in proteins.” In: *The EMBO journal* 5.4 (1986), pp. 823–826.

[85] Laurent Fourrier, Cristina Benros, and Alexandre G De Brevern. “Use of a structural alphabet for analysis of short loops connecting repetitive structures”. In: *BMC bioinformatics* 5 (2004), pp. 1–13.

[86] Anne-Cloude Camproux, Romain Gautier, and Pierre Tuffery. "A hidden markov model derived structural alphabet for proteins". In: *Journal of molecular biology* 339.3 (2004), pp. 591–605.

[87] Manoj Tyagi et al. "Protein structure mining using a structural alphabet". In: *Proteins: Structure, Function, and Bioinformatics* 71.2 (2008), pp. 920–937.

[88] Agnel Praveen Joseph, Narayanaswamy Srinivasan, and Alexandre G de Brevern. "Improvement of protein structure comparison using a structural alphabet". In: *Biochimie* 93.9 (2011), pp. 1434–1445.

[89] Michel Van Kempen et al. "Fast and accurate protein structure search with Foldseek". In: *Nature Biotechnology* 42.2 (2024), pp. 243–246.

[90] Inigo Barrio-Hernandez et al. "Clustering predicted structures at the scale of the known protein universe". In: *Nature* 622.7983 (2023), pp. 637–645.

[91] Steven Henikoff and Jorja G Henikoff. "Amino acid substitution matrices from protein blocks." In: *Proceedings of the National Academy of Sciences* 89.22 (1992), pp. 10915–10919.

[92] Mohiuddin Ahmed, Raihan Seraj, and Syed Mohammed Shamsul Islam. "The k-means algorithm: A comprehensive survey and performance evaluation". In: *Electronics* 9.8 (2020), p. 1295.

[93] Martin Ester et al. "A density-based algorithm for discovering clusters in large spatial databases with noise". In: *kdd*. Vol. 96. 34. 1996, pp. 226–231.

[94] Leland McInnes, John Healy, Steve Astels, et al. "hdbscan: Hierarchical density based clustering." In: *J. Open Source Softw.* 2.11 (2017), p. 205.

[95] Mihael Ankerst et al. "OPTICS: Ordering points to identify the clustering structure". In: *ACM Sigmod record* 28.2 (1999), pp. 49–60.

[96] Wolfgang Kabsch. "A solution for the best rotation to relate two sets of vectors". In: *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography* 32.5 (1976), pp. 922–923.

[97] Evangelos A Coutsias, Chaok Seok, and Ken A Dill. "Using quaternions to calculate RMSD". In: *Journal of computational chemistry* 25.15 (2004), pp. 1849–1857.

[98] Rafael Dolezal et al. "Computational Complexity of Kabsch and Quaternion Based Algorithms for Molecular Superimposition in Computational Chemistry". In: *Proceedings of the 21st EANN (Engineering Applications of Neural Networks) 2020 Conference: Proceedings of the EANN 2020 21*. Springer. 2020, pp. 473–486.

[99] Peter Wills and François G Meyer. "Metrics for graph comparison: a practitioner's guide". In: *Plos one* 15.2 (2020), e0228728.