# LIGAND BINDING SITE PREDICTOR - THEORETICAL BACKGROUND

## INTRODUCTION

Protein-ligand interactions play a critical role in biological functions such as enzyme catalysis and signal transduction, among others. Therefore, identifying the binding site of a protein and predicting its interactions with ligands is of great importance for studying the structure and function of proteins, identifying potential drug targets and predicting the effects of mutations on these regions.

While experimental methods are considered the gold standard for determining the structure and function of proteins, they are expensive, time-consuming, and challenging (Nitsche and Otting, 2018). On the other hand, bioinformatic methods offer a more efficient and cost-effective approach to ligand binding site prediction, particularly for high-throughput applications, as for instance virtual screening of compound libraries for drug discovery (Komiyama et al., 2016).

Several approaches have been developed for predicting ligand binding sites, divided into sequence-based and structure-based methods. Even though we have more information about protein sequences, the structural approaches have shown a better performance, as structures are more conserved and predictions can be made in cases without sequence similarity (Macari et al., 2019).

These structure-based strategies use atomic spatial coordinates and physicochemical properties, and can also be divided in several methods. Geometric-based methods analyze the geometry and shape of the protein structure to identify surface cavities as Fpocket (Le Guilloux et al., 2009) and MSPocket (Zhu and Pisabarro, 2011), while energetic-based ones search for favorable regions on the protein surface for a ligand binding interaction in terms of energy, with developed methods such as SiteHound (Ghersi and Sanchez, 2009) and FTSite (Ngan et al., 2012). Template-based methods rely on structural alignments between the input structure and homologues with a bound ligand from a database as LT-scanner (Hwang et al., 2017).

More recently, genomic and proteomic data proliferation have allowed machine learning algorithms as P2Rank (Krivák and Hoksza, 2018) to be trained on datasets of known binding sites to identify common features and predict new binding sites, or even using neural networks for detecting more complex relationships between features, as it is done with DeepSite (Jiménez et al., 2017) and GRaSP (Santana et al., 2020).

# PREDICTOR

We developed a standalone program based on machine learning in Python for ligand binding site prediction given a protein structure.
Binding annotation to train the model was extracted from the BioLip database, which contains biologically relevant ligand-protein interactions based on structural information from the Protein Data Bank (PDB) (Yang et al., 2013). We downloaded a non-redundant dataset in a tabulated format with 75,236 binding annotations, and preprocessed it by keeping the relevant information for training the model in every entry such as PDB ID, chain of the receptor protein, ligand ID and the binding site residues of the receptor.
We took 450 binding annotations from different protein interactions and represented each residue as a row of a matrix. We considered a set of relevant features and calculated them for every residue in the matrix as in the GRaSP method.

The pdb files according to the annotation in the BioLip database were extracted with the following code in Python:

```
#### CREATE BIOLIP DICTIONARY ####
# Load the file separated by tabs and with no header
df = pd.read_csv('BioLiP_nr.txt', sep='\t', header=None)

# Delete duplicate lines of the first column of df
df_unique = df.drop_duplicates(subset=0)
df_450 = df_unique.head(450) # Take only the first 450 lines of df

# Select the relevant columns of the df
df_model = df_450[[0, 1, 4, 5, 8, 9, 19, 20]]

# Name the columns
df_model.columns = ['pdb_id', 'receptor_chain', 'ligand_id', 'ligand_chain',
'binding_site_residues', 'catalytic_site_residues', 'ligand_residues_seq_num',
'receptor_sequence']

# Extract and save the relevant columns in another dataframe (biolip_relevant)
relevant_cols = ["pdb_id", "receptor_chain", "ligand_id",
"binding_site_residues"]
biolip_relevant = df_model.loc[:, relevant_cols]

biolip_dict = {}
for _, row in biolip_relevant.iterrows():
   pdb_id = row["pdb_id"]
   receptor_chain = row["receptor_chain"]
   ligand_id = row["ligand_id"]
   binding_site_residues = row["binding_site_residues"]

   if pdb_id not in biolip_dict:
       biolip_dict[pdb_id] = []

   biolip_dict[pdb_id].append((receptor_chain,ligand_id,binding_site_residues))
```

```
### DOWNLOAD PDB FILES FROM BIOLIP ####
# Download the entire PDB files of the BioLip database
pdbl = PDBList()
for pdb_id in biolip_dict.keys(): # Take each key of the biolip_dict (PDB code)
    pdbl.retrieve_pdb_file(pdb_id, file_format="pdb", pdir="./pdbs")
```

Then, we generated a list that contained the residues in the binding site according to the BioLip database (result_bindingsite_list) using the code that we can see below in Python.

```
# Extract binding site information from a dictionary of protein structures and
sequences, and store the information in a list for further processing

aa_letter_codes = {'A': 'ALA', 'R': 'ARG', 'N': 'ASN', 'D': 'ASP', 'C': 'CYS',
'E': 'GLU', 'Q': 'GLN', 'G': 'GLY', 'H': 'HIS', 'I': 'ILE', 'L': 'LEU', 'K':
'LYS', 'M': 'MET', 'F': 'PHE', 'P': 'PRO', 'S': 'SER', 'T': 'THR', 'W': 'TRP',
'Y': 'TYR', 'V': 'VAL'}

result_bindingsite_list = []
for key, values in biolip_dict.items():
    for value in values:
        chain, _, sequence = value # Store the chain and sequence in a tuple

        for aa_code in sequence.split():

            if aa_code[0] in aa_letter_codes.keys():
                aa_name = aa_letter_codes[aa_code[0]]
                residue_num = aa_code[1:]


result_bindingsite_list.append(f'pdb{key}_{chain}_{aa_name}_{residue_num}')
```

The pdb files were downloaded in .ent format, so we transformed them into .pdb format and stored them in a new folder called “pdbs_pdb/” with the following Python code:

```
ent_folder = './pdbs' # Folder of .ent files
pdb_folder = './pdbs_pdb' # Folder where .pdb files will be saved

# Create the folder if it does not exist
if not os.path.exists(pdb_folder):
    os.mkdir(pdb_folder)

# Iterate on the .ent files
for file in os.listdir(ent_folder):
    if file.endswith('.ent'):
        ent_path = os.path.join(ent_folder, file)
        pdb_path = os.path.join(pdb_folder, file.replace('.ent', '.pdb'))

        # Convert .ent to .pdb
        with open(ent_path, 'r') as f_ent, open(pdb_path, 'w') as f_pdb:
            shutil.copyfileobj(f_ent, f_pdb)
```

The final training matrix consisted of every residue extracted and an associated feature vector with 14 descriptors that can be divided in three blocks:

- Atomic properties:

We considered a set of atomic properties, relevant to a protein-ligand interaction, that defined every residue in terms of which atoms it has. These properties were assigned for each individual atom of the residue in a file used in GRaSP ("atom_types.csv"):

- Hydrophobic (HPB): In residues that are insoluble in water and tend to be found in the interior of proteins forming hydrophobic cores. These residues can interact with ligand hydrophobic regions.
- Donor (DON) and Acceptor (ACP): Residues with hydrogens as donors and electronegative atoms as acceptors of hydrogen bonds. They are important to form hydrogen bonds with ligands with these groups of atoms.
- Aromatic (ARM): Residues with aromatic carbons that can interact with other aromatic rings or pi-electron systems of the ligand.
- Positive (POS) and negative (NEG): Positively and negatively charged residues can interact with oppositely charged regions of a ligand in the binding site.
- Cysteine (CYS): This residue has the capability to form disulfide bonds, so it can be important in case the ligand has it too.

Each of these properties represents a column in the protein matrix, with a particular score for each residue that depends on itself.

- Interactions:

After defining individual features for each residue, we also gave importance to possible interactions between nearby residues (distance threshold of 5 Å, described by Karlin et al., 1999). For that, euclidean distances were calculated to determine neighbor residues and count possible interactions based on the same distance intervals used in GRaSP:

- Aromatic stacking: ARM residues can stack on top of each other, and a multiple presence of these residues in a region can help an aromatic ligand to bind there. The distance interval for aromatic stacking was 1.5-3.5 Å.
- Hydrogen bonds: DON and ACP neighbor residues can form a hydrogen bond stabilizing the region and possibly contributing to the presence of a binding site. The distance interval was 2-3 Å.
- Hydrophobic: HPB residues can form non-polar interactions with each other. These interactions can stabilize the binding site, especially in the presence of a hydrophobic ligand. The distance interval for hydrophobic interactions was set to 2-3.8 Å.
- Repulsive: Repulsive interactions between the same charge POS-POS and NEG-NEG in a region can destabilize it and create a barrier for ligand binding, so these regions can be discarded. The distance interval was 2-6 Å.
- Salt bridge: Interactions between opposite charges POS-NEG can stabilize regions for ligand-protein complexes, so they can indicate a potential interaction there. Same 2-6 Å interval is applied.
- Disulfide bridge: Disulfide bonds can covalently link two CYS residues, stabilizing the local protein structure. The presence of a disulfide bridge can affect the protein's

conformation and, therefore, the binding site. The distance interval was set to 3-7.5 Å, having as a reference a machine learning prediction (Gao et al., 2020).

Distances between residues were calculated and these columns were added to the matrix.

- Solvent accessibility:

Another important factor considered is solvent accessibility, calculated as solvent-accessible surface area (SASA) with DSSP. Residues with high SASA are more likely to be involved in binding interactions.
A new accessibility (ACC) descriptor was added to the matrix to complete it.

# MACHINE LEARNING MODEL

Prediction results are sometimes represented as a ranked list of pockets with grid points in some approaches such as Fpocket and P2Rank. As we are doing a binary classification for the presence (or not) of each residue in the binding site, our algorithm has a more residue-centric perspective. We consider our strategy a better approach as it considers each individual residue in the binding site instead of treating the binding site as a whole. This can lead to a more accurate prediction, especially when the binding site is composed of multiple pockets or the pockets are irregularly shaped.

Therefore, we added a new column to the matrix to represent which residues are in the binding site. To fill it, we extracted the information about binding site residues for each PDB file given in our BioLip dataset and attributed "1" for presence and "0" for not presence in the binding site.

Once all the features were extracted from each residue, the matrix was complete for training and testing the model.

Considering that just a few residues are implicated in the protein-ligand interaction, the feature of being or not in the binding site was highly unbalanced towards negative cases, affecting the development of a model. To solve that, we upsampled the minority class and combined it with the majority class in a balanced way. The upsampled dataset was then splitted into training (70%) and test (30%) sets.
The model was developed using this training set and a Random Forest classifier with 100 decision trees and a class weight of 1:1000 in favor of the binding site residues to improve sensitivity and reduce false negatives. Prediction on the test set was done with an accuracy of 0.98. This was computed using the following code in Python:

```
# Select all features of the matrix of the training dataset for the prediction
X = full_matrix.iloc[:, 1:]

# Select the first column (binding_site), which will be compared with the result
of the prediction
Y = full_matrix.iloc[:, 0]
# Separate majority and minority classes
```

```
X_majority = X[Y==0]
X_minority = X[Y==1]

# Upsample minority class
X_minority_upsampled = resample(X_minority,
                                replace=True, # Sample with replacement
                                n_samples=len(X_majority), # Match majority class
                                random_state=50) # Reproducible results

# Combine majority class with upsampled minority class
X_upsampled = pd.concat([X_majority, X_minority_upsampled])

# Create new target variable
Y_upsampled = pd.Series([0]*len(X_majority) + [1]*len(X_minority_upsampled))

# Split the upsampled dataset into training set and test set
X_train, X_test, Y_train, Y_test = train_test_split(X_upsampled, Y_upsampled,
test_size=0.3)

# Create a Random Forest Classifier, balancing for the classes
rfc = RandomForestClassifier(n_estimators=100, class_weight={0: 1, 1: 1000})

# Train the classifier on the training data
rfc.fit(X_train, Y_train)

# Predict the species for the testing data
Y_pred = rfc.predict(X_test)

# Calculate the accuracy score
accuracy = accuracy_score(Y_test, Y_pred)
print(f"Accuracy: {accuracy}")
```

The model works with an input protein structure file (PDB file), where it scans for possible candidate residues to be in the ligand binding site. When residues are identified, outputs are generated. On the one hand, a list of these residues is printed and, on the other hand, a new PDB file which contains only these residues is generated. This can be visualized using a molecular graphics software (Chimera, PyMOL).

All the steps carried out in this project are represented in Figure 1.

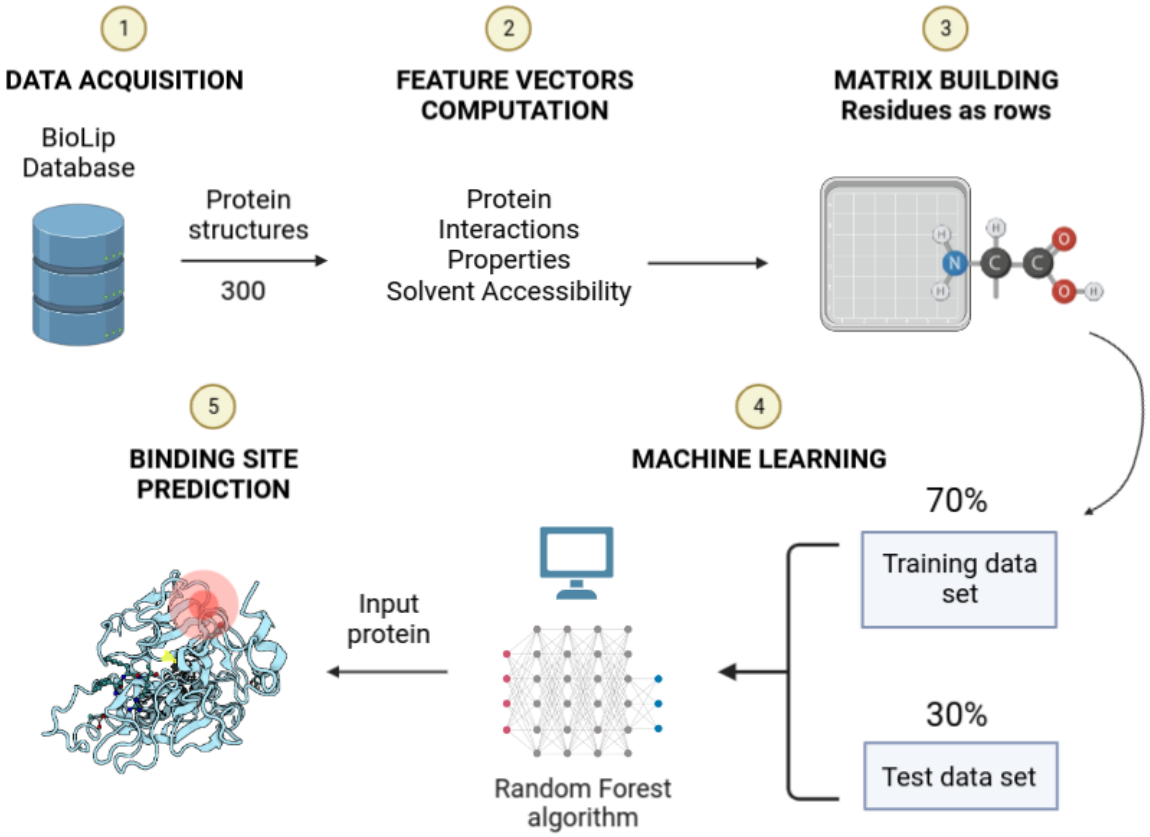


*Figure 1. Workflow of the Binding Site predictor, based on supervised learning.*

# Bibliography

Santana, C. A., Silveira, S. A., Moraes, J. P. A., Izidoro, S. C., de Melo-Minardi, R. C., Ribeiro, A. J. M., Tyzack, J. D., Borkakoti, N., & Thornton, J. M. (2020). GRaSP: a graph-based residue neighborhood strategy to predict binding sites. Bioinformatics (Oxford, England), 36(Suppl_2), i726–i734.

Ghersi, D., & Sanchez, R. (2009). EasyMIFS and SiteHound: a toolkit for the identification of ligand-binding sites in protein structures. Bioinformatics (Oxford, England), 25(23), 3185–3186.

Hwang, H., Dey, F., Petrey, D., & Honig, B. (2017). Structure-based prediction of ligand-protein interactions on a genome-wide scale. Proceedings of the National Academy of Sciences of the United States of America, 114(52), 13685–13690.

Jiménez, J., Doerr, S., Martínez-Rosell, G., Rose, A. S., & De Fabritiis, G. (2017). DeepSite: protein-binding site predictor using 3D-convolutional neural networks. Bioinformatics (Oxford, England), 33(19), 3036–3042.

Karlin, S., Zhu, Z., & Baud, F. (1999). Atom density in protein structures. PNAS, 96(22), 12500-12505.

Komiyama, Y., Banno, M., Ueki, K., Saad, G., & Shimizu, K. (2016). Automatic generation of bioinformatics tools for predicting protein-ligand binding sites. Bioinformatics (Oxford, England), 32(6), 901–907.

Krivák, R., & Hoksza, D. (2018). P2Rank: Machine learning based tool for rapid and accurate prediction of ligand binding sites from protein structure. Journal of Cheminformatics, 10, 39.

Le Guilloux, V., Schmidtke, P., & Tuffery, P. (2009). Fpocket: an open source platform for ligand pocket detection. BMC bioinformatics, 10, 168.

Macari, G., Toti, D., & Polticelli, F. (2019). Computational methods and tools for binding site recognition between proteins and small molecules: from classical geometrical approaches to modern machine learning strategies. Journal of computer-aided molecular design, 33(10), 887–903.

Ngan, C. H., Hall, D. R., Zerbe, B., Grove, L. E., Kozakov, D., & Vajda, S. (2012). FTSite: high accuracy detection of ligand binding sites on unbound protein structures. Bioinformatics (Oxford, England), 28(2), 286–287.

Nitsche, C., & Otting, G. (2018). NMR studies of ligand binding. Current opinion in structural biology, 48, 16–22.

Zhu, H., & Pisabarro, M. T. (2011). MSPocket: an orientation-independent algorithm for the detection of ligand binding pockets. Bioinformatics (Oxford, England), 27(3), 351–358.