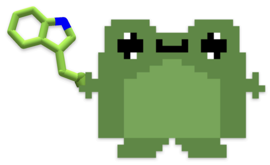
wisefrog



JONE GUTIÉRREZ DÍEZ
MARTÍN SUÁREZ ÁLVAREZ

MSc Bioinformatics for Health Science
Universitat Pompeu Fabra

INTRODUCTION

Studying the ligand binding sites (LBS) of proteins has a great impact on biological science due to its importance on fields like drug discovery. Despite the extensive [number of] years of experimental determination and classification of these sites, new LBS motifs are still being discovered nowadays – the work has not finished yet. It's usual that, due to the low number of experimentally determined LBS, researchers use computational methods to determine these sites [1].

The fast development of artificial intelligence has had significant repercussions on structural biology, and protein binding site prediction is not an exception. Further to this, many approaches have been built in order to carry on this task, taking into account different perspectives both in the biological side (e.g.: chemical/physical, geometrical and evolutionary characteristics of proteins) and engineering part of the process (different machine learning or deep learning models) [2]. Understanding the differentiating features of LBS is not only important for understanding the implication of ligands in protein function, but also for the computational design of new proteins that bind new ligands with new functionalities [3].

Ligand binding sites are usually more evolutionarily conserved than the rest of the structure [4], and as they have differential characteristics which residues contribute, according to its atoms, this is an interesting approach to direct our project. WiseFrog is a Deep Learning based model that classifies LBS according to the physical and chemical properties of atoms. Moreover, our model characterizes atoms by its environment (geometrically, the atoms that surround it), which gives a precise and specific perspective of how LBS are composed and classifies whether an atom or residue belongs to a LBS or not.

We decided on the "wisefrog" name for many reasons. The term "Wise" conveys the idea of intelligence, sagacity, and deep understanding. Meanwhile, "Frog" symbolizes adaptability, agility, and precision, akin to the predictor's ability to navigate diverse environments and accurately identify LBS. All in all, just as a frog captures its prey with precision and efficiency, "WiseFrog" encapsulates the intelligence, adaptability, and precision essential for a successful ligand binding site predictor.

DEEP LEARNING MODEL

Several approaches have been proposed and followed to predict the regions that are relevant for ligand binding. Some of these are based on geometrical measurements of the protein cavities, probing the protein surface to find energetically stable regions for binding, similarity based searches that use template structures, or on identification of sequence differences between LBS and non-LBS. Additionally, several machine learning and deep learning algorithms have proved to be useful for LBS prediction [3] [5].

In our case, the decision for the approach was not only guided by the efficiency and suitability of an algorithm to solve the problem, but also had a pedagogical motivation. We saw this project as a way of getting introduced to a field and a set of tools in which we did not have any hands-on experience. Hence, we decided to use a deep learning or a neural network to do the LBS prediction.

Several deep learning algorithms have been developed in the field of ligand binding prediction, with different architectures depending on the features that are considered. Methods using both deep learning, mutual information and physical distances use graph neural networks [6] while others use fully-connected neural networks for Ca^{2+} and Mg^{2+} binding prediction [7] or convolutional neural networks for methods that consider a structure as a 3D image [8].

Taking into account the huge diversity of algorithms and approximations, and also being aware of the limitations we had to face for this project (lack of computational power, time, and domain knowledge) we decided to use a simple fully-connected neural network. For this, we followed the steps proposed in [9]. See the structure of the network in **Figure 1**.

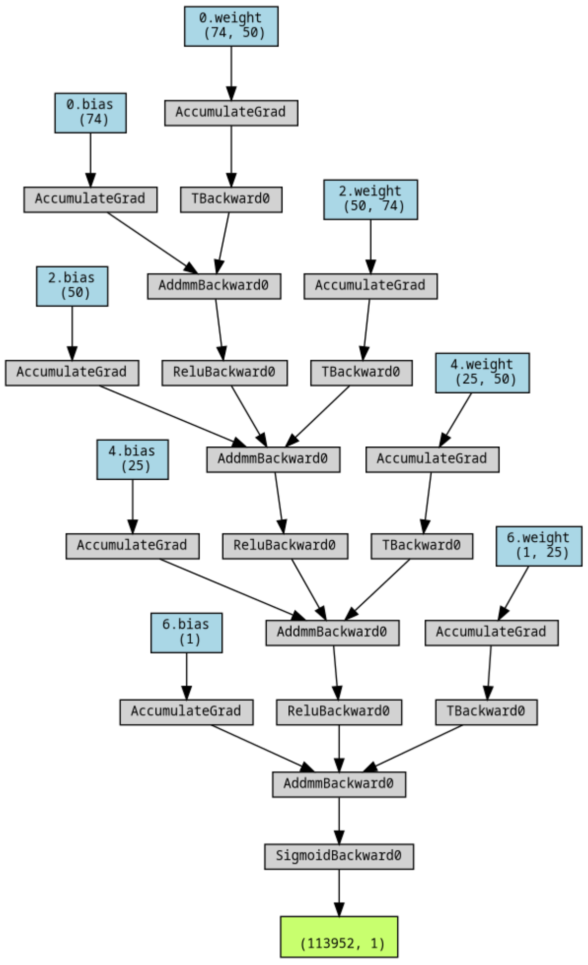


Figure 1. Structure of the neural network model. Obtained with the library torchviz.

This model used the binary cross entropy function as loss function, and the optimizer Adam as the optimization algorithm for the gradient descent. We used an exponential learning rate scheduler with the parameter gamma set to 0.95. We used early stop to avoid overfitting with the patience set to 5, 100 epochs and a batch size of 32.

The model was assessed by measuring the receiver operating characteristic (ROC) area under the curve (AUC), and the F1 score.

Once the model had been defined, we designed a strategy for feature extraction to create a training set. that could be used by the neural network.

ENVIRONMENT CHARACTERIZATION: TRAINING SET AND PREDICTION

Our first aim was to create a training dataset by measuring several features in the environments of the atoms that participate in the binding of ligands, and compare them with the environments of the atoms that do not. Since the ligand binding cavity of a protein has typically a volume of around 10^2 to 10^3 Å^3 [10], we considered an environment to be a sphere with radius 6.2 Å.

For the creation of the training set, we determined the atoms that participate in the ligand binding (LBS atoms) by detecting the ligands of a given PDB file (we considered them to be residues that are not water or the 20 standard amino acids), and annotating the atoms of the protein that are located at less distance than 6.2 Å from any of the ligand atoms. Since it was expected that the number of LBS atoms was significantly inferior to the number of non-LBS atoms, the procedure that was followed was to first determine the LBS atoms of the structure, and then randomly sample the same number of non-LBS atoms. This was done to avoid bias in the training of the deep learning algorithm for LBS prediction due to class imbalance.

After having annotated the LBS and non-LBS atoms that were going to be considered, the features of their environment were measured. The approach that was followed for feature extraction was to focus on the characteristics of the atoms that are comprised in the sphere of 6.2 Å, rather than the residues. This decision was made after comparing other groups' approaches, which made us wonder if a more "residue agnostic" perspective that focused on the properties of the atoms that belonged to an environment instead of the residues would be successful in predicting LBS.

Finally, we got a dataset of 50 predictors that can be fully consulted in the data preparation section.

A similar approach was followed to extract the features from the input PDB files: when the user introduces a structure for LBS prediction, the structure is parsed and the features for the environments of the atoms are computed. In this case, there is no distinction between LBS and non-LBS sites since the aim is to predict the atoms that are likely to be participating in the binding of ligands in non-annotated structures. The dataset that is extracted from the PDB file is fed into the neural network, and the potential LBS atoms are predicted based on the characteristics of their environment.

FEATURE EXTRACTION AND DATA PREPARATION

The environments' dataset was created from PDB files extracted from RCSB Protein Data Bank. To select protein structures identified as protein-ligand complexes we used the scPDB database [11] to obtain a list of PDB identifiers of structures of interest. From these identifiers we randomly selected 1995 structures to perform feature extraction. We did a random selection of the structures from scPDB to avoid choosing related structures that would bias the search towards a determined set of proteins.

Each PDB file was parsed to record the features of the environments of LBS and non-LBS atoms. Each row of the data frame consists of an atom's environment, that is identified with the residue number and the atom name in a chimera-recognizable format (i.e., *:X@Y* where "*X*" is the residue number and "*Y*" is the atom's name). The feature extraction is based on the characteristics of the atoms of the sphere with radius of 6.2 Å, taking into account the proportion of each type of atom and their chemical properties. The final number of predictors was 50, which can be consulted in **Table 1**. The final number of observations (environments) was 113952.

Most of the features were directly extracted from the PDB file, such as the proportion of each atom type or element, or the average B factor. However, some features required external software. This was the case for the solvent-accessible surface area (SASA) calculation [12] that was performed using the Bio.PDB.SASA module from Biopython [13]. For the extraction of the chemical properties of the atoms, we manually created a dictionary based on the information on [14] that associated some atom types in specific residues with a given property (aliphatic, aromatic, donor and acceptor of hydrogen bonds). We also created a dictionary for charge assignment to the atoms assuming a pH of 7.4, based on the pK$_a$ of the side chains which they belong to. Finally, the training set contains the target variable, a binary label with value 1 for atoms that are categorized as LBS, and 0 for non-LBS.

Description	Variable name
Proportion of each atom type in the environment	CG, ND2, OE2, CD, OG1, CG2, CB, CA, N, C, CD2, CE2, CD1, OE1, O, CE, SD, CZ, NZ, OH, CE1, CG1, OD1, NE2, NH1, OD2, OG, NH2, NE, ND1, SG, NE1, CH2, CZ2, CE3, CZ3, OXT
Proportion of each element in the environment	totalC,totalN, totalO, totalS,
Density of the environment (number of atoms/Å^3)	environment_density
Average B factor for the atoms in the environment	b_factor
Average SASA for the atoms of the environment	sasa
Proportion of aliphatic, aromatic, hydrogen bond donors, hydrogen bond acceptors and hydrogen bond donors and acceptors in the environment	aliphatic, aromatic, donor, acceptor, don_acc
Average charge of the environment	charge

Table 1: name of the predictor variables from the training dataset and description.

The initial dataset created from the selected 1995 PDB structures was splitted into a training and test set (80% and 20% of the dataset respectively). Both datasets were normalized by subtracting the mean and dividing by the standard deviation of the training set.

EVALUATION OF THE MODEL

To evaluate the model, the test set was predicted using the neural network trained on the training dataset. The receiver operating characteristic (ROC) curve was computed for the prediction of the normalized test set, with an area under the curve (AUC) of 0.79 (see **Figure 2**).

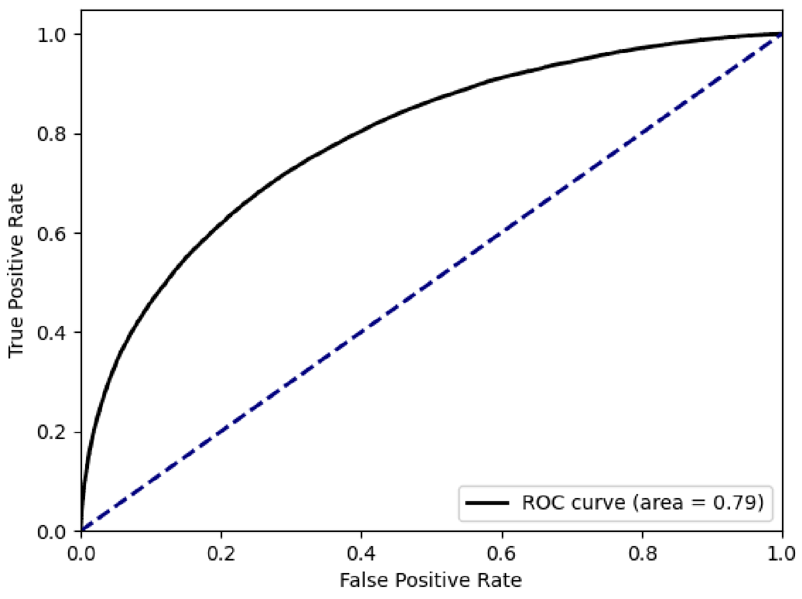


Figure 2: ROC curve for the binary prediction of the ligand binding sites based on the normalized test set.

Since the last activation function of the neural network is a sigmoid function, our model gives a value between 0 and 1. The closer to 1, the more likely it is for an atom to belong to a LBS based on its environment. To set a threshold to convert this value into a binary prediction, we manually inspected the performance of the model with different values. The final decision was to use a value of 0.4, with the aim of optimizing the precision and recall of the prediction bot for the LBS and non-LBS atoms, but also prioritizing detection of LBS atoms at the expense of a higher number of false positives. The performance metrics that were assessed can be consulted in **Table 2**, as they are presented by the PyTorch performance report.

	Precision	**Recall**	**F1-score**
Non-LBS	0.78	0.71	0.75
LBS	0.63	0.71	0.67
Macro average	0.71	0.71	0.71
Weighted average	0.72	0.71	0.72

Accuracy
0.71

Table 2. Performance metrics on the prediction of the normalized test set.

METHODS

All the algorithms were performed in Python 3, in this section, we outline the libraries and packages used in our project for the implementation of various functionalities. Each tool serves a specific purpose in facilitating data manipulation, neural network construction, command-line interface development, process management, and more.

Standard python libraries
For the simplification of writing user-friendly command-line interfaces we used **argparse**, a standard Python library. Moreover, for the data manipulation part of the work we used **pandas**, which was really useful for reading and writing data, handling missing data, merging datasets, reshaping datasets, etc.

Furthermore, to allow our scripts to interface seamlessly with the system environment in which they are run we used **sys,** a library that provides access to Python interpreter variables and functions, allowing us to manipulate the Python runtime environment. And, also **os** library, which provides a way of using operating system dependent functionality, enabling our scripts to interact with the file system, process ID's, and environment variables, among other things.

Neural Network design
Meanwhile, for the development of the neural network infrastructure, the key libraries were **torch** and **torch.nn**. torch provides a flexible and efficient interface for defining and computing tensors, with the added benefit of GPU acceleration, making our computations faster and more efficient. Complementing this, torch.nn provides a set of classes to construct and train neural networks. It simplifies the process of defining layers and forward passes, allowing us to focus on the higher-level design of our models.

BioPython
Biopython is a set of freely available tools for biological computation written in Python. It contains classes to represent biological sequences and sequence annotations [13].

According to the use we have given to the library, **Bio.PDB** is a module that provides classes for the representation and manipulation of biological macromolecules. It allows for the parsing of PDB (Protein Data Bank) formatted files into python data structures. In particular, **Bio.PDB.SASA** is a submodule we used for calculating the Solvent Accessible Surface Area (SASA) of a protein.

Graphical representations

For the purpose of plotting the model’s performance we used matplotlib library’s matplotlib.pyplot, which was practical for understanding the efficiency and behavior of wisefrog.

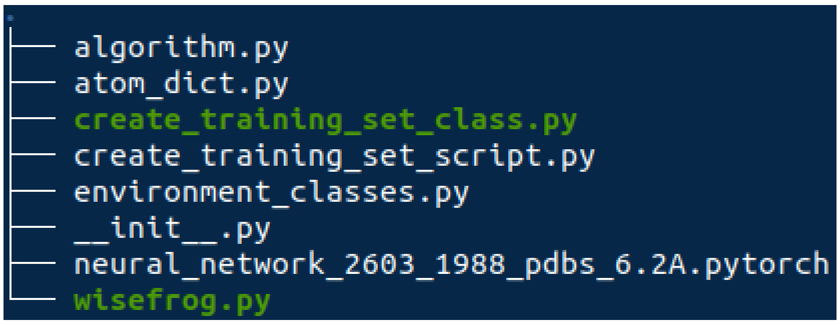
Customized modules

On the one hand, we developed some integral modules tailored for our project. The **environment_classes** module provides classes and methods for analyzing protein structures. It contains several methods in order to get the ligands in the structure, the atoms labeled as LBS, the features of the environment of each atom and the dictionary of environments of each residue in the structure of the input PDB file.

On the other hand, **create_training_set** contains the TrainingSet class, which is used to create a formatted training set from a directory of PDB files. Additionally, it also encloses methods in order to list all the PDB files in the input directory, extract features and format them into a training set in pandas DataFrame format.

Additionally, the **algorithm** module was implemented to process the training set and train the proper deep learning model. This module also includes utilities for tasks such as model saving/loading and performance visualization.

The structure of the wisefrog package can be consulted in **Figure 3.**



```
├── algorithm.py
├── atom_dict.py
├── create_training_set_class.py
├── create_training_set_script.py
├── environment_classes.py
├── __init__.py
├── neural_network_2603_1988_pdbs_6.2A.pytorch
└── wisefrog.py
```

Figure 3. Structure of the package wisefrog

DATA AVAILABILITY

All the code can be accessed from the GitHub repository https://github.com/joneg10/wisefrog/, and all the data (full set, splitted training and test sets in parquet format, compressed installable and the PDB files that were used to train the model) is accessible from https://drive.google.com/drive/folders/1Eyo0LAPbV2SAHIiHZi1y6lzYhelta3pw?usp=sharing.

REFERENCES

[1] Broomhead, N. K., & Soliman, M. E. (2017). Can We Rely on Computational Predictions To Correctly Identify Ligand Binding Sites on Novel Protein Drug Targets? Assessment of Binding Site Prediction Methods and a Protocol for Validation of Predicted Binding Sites. *Cell biochemistry and biophysics*, *75*(1), 15–23. https://doi.org/10.1007/s12013-016-0769-y

[2] Zhao J, Cao Y, Zhang L. Exploring the computational methods for protein-ligand binding site prediction. Comput Struct Biotechnol J. 2020 Feb 17;18:417-426. doi: 10.1016/j.csbj.2020.02.008. PMID: 32140203; PMCID: PMC7049599.

[3] Feldmeier, K., & Höcker, B. (2013). Computational protein design of ligand binding and catalysis. *Current opinion in chemical biology*, *17*(6), 929–933. https://doi.org/10.1016/j.cbpa.2013.10.002

[4] Capra JA, Singh M. (2007). Predicting functionally important residues from sequence conservation. *Bioinformatics.*1;23(15), 1875-82. https://doi.org/10.1093/bioinformatics/btm270

[5] Xie, Z. R., & Hwang, M. J. (2015). Methods for predicting protein-ligand binding sites. *Methods in molecular biology (Clifton, N.J.)*, *1215*, 383–398. https://doi.org/10.1007/978-1-4939-1465-4_17

[6] Wang, K., Zhou, R., Tang, J., & Li, M. (2023). GraphscoreDTA: optimized graph neural network for protein-ligand binding affinity prediction. *Bioinformatics (Oxford, England)*, *39*(6), btad340. https://doi.org/10.1093/bioinformatics/btad340

[7] Sun, K., Hu, X., Feng, Z., Wang, H., Lv, H., Wang, Z., Zhang, G., Xu, S., & You, X. (2022). Predicting Ca^{2+} and Mg^{2+} ligand binding sites by deep neural network algorithm. *BMC bioinformatics*, *22*(Suppl 12), 324. https://doi.org/10.1186/s12859-021-04250-0

[8] Kandel, J., Tayara, H., & Chong, K. T. (2021). PUResNet: prediction of protein-ligand binding sites using deep residual neural network. *Journal of cheminformatics*, *13*(1), 65. https://doi.org/10.1186/s13321-021-00547-7

[9] Tam, A. (2023, April 7). Develop your first neural network with pytorch, step by step. *MachineLearningMastery.com*. https://machinelearningmastery.com/develop-your-first-neural-network-with-pytorch-step-by-step/ Accessed: 2024, March 3.

[10] Liang J, Edelsbrunner H, Woodward C. Anatomy of protein pockets and cavities: measurement of binding site geometry and implications for ligand design. *Protein Sci*. 1998 Sep;7(9):1884-97. doi: 10.1002/pro.5560070905. PMID: 9761470; PMCID: PMC2144175.

[11] Desaphy J, Bret G, Rognan D, Kellenberger E, sc-PDB: a 3D-database of ligandable binding sites—10 years on, *Nucleic Acids Research*, Volume 43, Issue D1, 28 January 2015, Pages D399–D404, https://doi.org/10.1093/nar/gku928

[12] Shrake, A., & Rupley, J. A. (1973). Environment and exposure to solvent of protein atoms. Lysozyme and insulin. *Journal of molecular biology*, 79(2), 351–371. https://doi.org/10.1016/0022-2836(73)90011-9

[13] Cock, P. J., Antao, T., Chang, J. T., Chapman, B. A., Cox, C. J., Dalke, A., Friedberg, I., Hamelryck, T., Kauff, F., Wilczynski, B., & de Hoon, M. J. (2009). Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics (Oxford, England)*, *25*(11), 1422–1423. https://doi.org/10.1093/bioinformatics/btp163

[14] Schmitt, S., Kuhn, D., & Klebe, G. (2002). A new method to detect related function among proteins independent of sequence and fold homology. *Journal of molecular biology*, *323*(2), 387–406. https://doi.org/10.1016/s0022-2836(02)00811-2