Linking QSAR-Based Drug-Target prediction with AlphaFold

George Iniatis (2329642i)

April 14, 2023

ABSTRACT

Drug-target interactions (DTIs) play a crucial role in drug discovery and pharmacology. However, their experimental determination is time-consuming and limited. Unwanted or unexpected DTIs could cause severe side effects. Therefore, the creation of machine learning models that can quickly and confidently predict whether thousands of drugs and proteins bind together and how much could be crucial.

The project aimed to create a new curated dataset and then use this dataset to train machine learning models. Our models were split into two categories, baseline and enhanced, with baseline using just the drug and protein sequence descriptors and the enhanced using our protein structural embeddings derived from AlphaFold's predictions in addition to those descriptors.

The created embeddings proved to be ineffective. However, our trained models could still prove useful in uncovering interesting relationships that could be investigated further.

1. INTRODUCTION

This chapter will introduce the project on a high level and examine its motivation and objectives.

1.1 Motivation

Drug-target interactions (DTIs) refer to the interactions of chemical compounds and biological targets, proteins in our case, inside the human body (Sachdev and Gupta 2019). Given that both proteins and drugs are chemically active molecules in the bloodstream, it would make sense that they interact in some way (Yartsev 2022). These interactions usually form an ever-changing, benign and reversible binding where both molecules move through the bloodstream interlocked together (Yartsev 2022). The vast majority of drugs administered take this into account and use this process (Yartsev 2022). A protein-bound drug is usually too big to pass through a biological membrane like that of a cell. Therefore only the unbound drug, usually in equilibrium with the bound drug, can pass through and produce the desired pharmacological effect, like the treatment of a disease, or the targeting of a tumour (Davis 2018).

Consequently, the degree of how much a drug binds to a protein can enhance or diminish the drug's effectiveness and performance. For example, minimally protein-bound drugs tend to penetrate tissue better and are excreted much faster from the body than those highly bound (Scheife 1989). In contrast, highly protein-bound drugs, usually meaning that the protein binding is so impactful that we have to pay attention to it, tend to last much longer. This is because the protein acts as a drug ”depot” that slowly releases the drug into the bloodstream, again keeping the bound and free drug in equilibrium (Yartsev 2022; Davis 2018).

DTIs play a crucial role in drug discovery and pharmacology. However, the experimental determination of these interactions with methods, such as fluorescence assays, is time-consuming and limited due to funding and the difficulty of purifying proteins (Shar et al. 2016; Wang et al. 2020). Past quantitative structure-relationship activity (QSAR) studies discussing protein-drug binding focused on testing thousands of drugs with just a single protein that they deemed important enough (Colmenarejo 2003; Vallianatou et al. 2013). These studies often did not consider the protein's sequence or structural information, concentrating their efforts on the drug molecules and their descriptors. However, this is not what we aimed to do in this study. Unwanted or unexpected DTIs could cause severe side effects, therefore, the creation of in silico machine learning models with high throughput that can quickly and confidently predict whether thousands of drugs and proteins bind together and how much could be crucial for medicinal chemistry and drug development, acting as a supplement to biological experiments (Shar et al. 2016; Wang et al. 2020).

1.2 Objectives

The project aims to gather publicly available data on known drug-target interactions and place them into a new curated dataset. Then, using this new dataset, train multiple machine learning models using simple QSAR descriptors derived from a drug's chemical properties and a protein's sequence and 3D structural information to predict whether they bind together. Each protein's 3D structure will be extracted from the AlphaFold protein structure database (Jumper et al. 2021; Varadi et al. 2022) and one of the main challenges of the project will be in creating a simple embedding, which efficiently encodes the structural information of the protein, that can then be used in our training process. The models' performance should then be evaluated and a rudimentary system using these models should be constructed.

2. BACKGROUND

This chapter will cover some essential background information without going too in-depth, especially in biomedical concepts. We will also explore how past studies have tackled variations of the problem, or parts of it, and any valuable techniques, strategies, and knowledge that could be extracted from their experiments.

The following background research was critical in better understanding this previously unknown problem which included complex chemical and medical concepts, while also introducing us to key machine learning concepts and best practices.

Given the important nature of the problem, as discussed in Section 1.1, there have been numerous attempts to construct machine learning models using various methods, techniques, and biochemical properties (Shar et al. 2016; Wang et al. 2020; Jiang et al. 2020).

Classification models try to predict whether a particular drug will bind with a selected protein (Active) or not (Inactive). However, their accuracy will be influenced by the threshold used to separate the two classes as suggested by Shar et al. (2016), and the definition of a successful binding may vary substantially for different proteins. Regression models can address these problems by trying to predict the drug-binding affinity, which can take multiple forms, with the dissociation constant (K_d), the inhibition constant (K_i), and the 50% inhibitory concentration (IC_{50}) being the most common amongst them (Jiang et al. 2020). These values are usually represented by their logarithmic versions.

2.1 Proteins Outline

Proteins are large complex molecules essential to all biological processes in every living thing (*AlphaFold Blog* 2022; O'Connor et al. 2010). They help with digestion, blood circulation, and muscle movement, provide structures, and defend our bodies from diseases. They are made from a combination of amino acids, and the interactions between these chains of amino acids make the protein fold. Each amino acids sequence usually maps 1-to-1 to a 3D structure, and that 3D structure defines what the protein does and how it works (Fridman 2022). If a protein is misfolded, it can lead to diseases such as Alzheimer's and Parkinson's (Chaudhuri and Paul 2006). There are about 10^{300} ways to fold a protein given its amino acid sequence (Levinthal 1969). An almost impossible-to-solve problem, known as 'Levinthal's paradox' or more simply as the 'Protein Folding Problem', that nature solves in milliseconds, and a problem that scientists worldwide have been trying to solve for the past 50 years (*AlphaFold Blog* 2022; Torrisi et al. 2020).

2.2 AlphaFold Breakthrough

We are aware of billions of proteins, and the number keeps increasing, but we only know the exact 3D shape of a small minority of these, roughly 170,000 (Jumper et al. 2021). Mapping these proteins using state-of-the-art methods such as X-ray crystallography and nuclear magnetic resonance is costly, time-consuming, and relies on extensive trial and error, making them highly inefficient and unsuitable for high-throughput screening (HTS). So naturally, scientists worldwide wanted to create a system that could predict a protein's 3D structure just by its amino acid makeup. This is precisely what DeepMind tried to achieve by creating an AI system called AlphaFold, trained on the known sequences and structures of the manually mapped-out proteins.

DeepMind's latest AlphaFold AI system, AlphaFold2, provided the first highly accurate and novel computational solution to this problem, not solving it in its entirety but arguably taking a significant step forward (Jumper et al. 2021). This was demonstrated at the 14th Critical Assessment of protein Structure Prediction (CASP), where AlphaFold outperformed all the other entries in the competition and achieved accuracy similar to that of experimental methods. CASP is organised every two years and uses recently discovered protein 3D structures as a blind test for the prediction systems submitted. It serves as the gold-standard assessment for the prediction accuracy of protein structures.

This breakthrough allowed DeepMind to release protein structure predictions covering almost the entire human proteome (98.5%). This mapping out of previously unknown human protein structures can provide highly beneficial information, allowing science to understand biological processes better and create more targeted, and therefore more effective, interventions (Tunyasuvunakool et al. 2021).

Discussing AlphaFold in detail is out of the scope of this project. However, it is an incredibly complex state-of-the-art system that directly predicts the 3D coordinates of all heavy atoms for a given protein just by using its amino acid sequence.

2.3 Molecular Descriptors

Molecular descriptors are numerical features extracted from chemical structures that can be one-dimensional (0D or 1D), 2D, 3D or 4D (Lo et al. 2018).

Although very simple to compute or extract, one-dimensional descriptors contain little contextual information on their own. Instead, they describe aggregate information such as counts and chemical properties. In addition, multiple chemical structures can have the same value for a common descriptor, making the usage of just a single 1D descriptor nearly meaningless. Therefore, they are usually expressed as feature vectors of multiple 1D descriptors or used together with descriptors of higher dimensionality.

Two-dimensional descriptors are the most common type found in literature and include molecular profiles, topological indices and 2D auto-correlation descriptors.

Three-dimensional descriptors extract chemical features and information from 3D coordinate representations and are considered to be the most sensitive to chemical structural differences. However, one of their fundamental limitations is their computational complexity. They include auto-correlation descriptors, quantum-chemical descriptors, substituent constants and surface:volume descriptors.

Four-dimensional descriptors are an extension of 3D descriptors with the addition that they simultaneously consider multiple structural confrontations. They include descriptors like GRID, Volsurf and Raptor.

2.3.1 *Molecular Fingerprints*

The molecular fingerprints of sub-structures can effectively capture the molecular information of drugs by converting them into a bit vector containing 0s and 1s (Wang et al. 2020). Each molecular sub-structure is mapped to a position in the bit vector. If a molecule contains a molecular sub-structure, a value of 1 is assigned to the corresponding bit in the vector or a 0 otherwise (*PubChem Fingerprints* 2022).

One of the most common molecular fingerprint sources is PubChem (Kim et al. 2021) which contains 881 molecular sub-structures (*PubChem Fingerprints* 2022), with some padding that needs to be removed before making use of them.

2.3.2 *Important Molecular Descriptors*

Shar et al. (2016), after analysing their trained random forest model, found that 2D autocorrelation, topological charge indices and 3D-MoRSE descriptors of compounds were the most essential chemical descriptors in predicting K_i.

2.4 Protein Sequence Descriptors

Structural and physiochemical descriptors calculated from amino acid sequences are widely used in protein-related machine learning research approaches such as the prediction of structural and functional classes and protein-protein interactions (Xiao et al. 2015). The type of descriptors chosen, the numerical representation that encodes the amino acids sequence, is a critical step and can significantly affect the predictive performance of the models a study is trying to train.

Past web servers and stand-alone programs like PROFEAT (Zhang et al. 2017), which currently seems inactive, and PseAAC (Shen and Chou 2008) that tried to calculate these descriptors were often limited in the number of descriptors they were providing, not flexible enough and difficult to integrate into the machine learning pipeline (Xiao et al. 2015).

Protr (Xiao et al. 2015), on the other hand, is a comprehensive package, written in R, that generates various numerical representations of proteins and peptides from amino acid sequences, calculating 8 descriptor groups composed of 22 types of commonly used descriptors that include roughly 22,700 descriptor values. In addition, this package also allows users to create custom descriptors, calculate similarity scores between pairs of proteins and provides useful helper functions.

2.4.1 *Position-Specific Scoring Matrix*

Position-Specific Scoring Matrix (PSSM). also known as Position Weight Matrix (PSW), is a typical representation of motifs, patterns in biological sequences, and therefore proteins (Jiang et al. 2020; Ranganathan et al. 2019). Motifs are represented as a vector of values, often probabilities, although different representations can also be found for every possible amino acid or residue at each sequence position.

2.4.2 *Important Protein Sequence Descriptors*

Shar et al. (2016), after analysing their trained random forest model, found that autocorrelation descriptors, amphiphilic pseudo-amino acid composition, and quasi-sequence-order descriptors of protein sequences were found to be the most effective in predicting K_i.

Ong et al. (2007) also evaluated the performance of different standard protein sequence descriptor sets individually and in various combinations and concluded that every set on its own is beneficial. However, the predictive performance of models can be enhanced by utilising different combinations of them. This selection could be made through standard feature selection processes.

2.5 Ligand-Based Approaches

Ligand-based methods are the most widely used and include QSAR and similarity search-based approaches (Shar et al. 2016). They make use of a drug's chemical and a protein's sequence descriptors without considering the protein's 3D structure (Aparoy et al. 2012).

Such approaches include the classification study conducted by Wang et al. (2020) and the regression study by Shar et al. (2016). In both studies, machine-learning models were trained, optimised and evaluated to predict drug-target interactions. However, their methodologies varied from one another, using different databases, datasets and tools, clearly expressing the myriad of distinct approaches one can use to solve this problem.

Shar et al. (2016) utilised the *Ki Database* (2022) from the Psychoactive Drug Screening Program (PDSP) (Roth et al. 2016) to retrieve DTIs. Then for each drug and protein combination used PubChem (Kim et al. 2021), ChemSpider (Pence and Williams 2010) and DrugBank (Wishart et al. 2018) to retrieve each drug's structure and UniProt (Consortium 2022) to retrieve each protein's sequence. Molecular descriptors were then calculated using Dragon (Mauri et al. 2006) and protein sequence descriptors using PROFEAT (Zhang et al. 2017). These descriptors were then fed into two models, one based on a support vector machine and another based on a random forest.

Wang et al. (2020) utilised the DTIs datasets of Yamanishi et al. (2008), split into Enzymes, Ion Channels, GPCRs and Nuclear Receptors. Then for each drug and protein combination used a PSSM, as mentioned in Subsection 2.4.1, to convert the protein sequence into numerical descriptors containing biological evolutionary information and then a discrete cosine transform (DCT) algorithm to extract the hidden features and integrate them with the molecular fingerprints extracted from PubChem (Kim et al. 2021). These features were then passed to a rotation forest model.

Both datasets used were relatively small and had less than 10,000 DTI entries, but that did not stop the models trained from achieving excellent predictive performances and even outperforming state-of-the-art models, possibly highlighting the dataset quality and the processes used.

2.6 Receptor-Based Approaches

Receptor-based approaches such as reverse docking try to predict the preferred conformation and binding strength of a compound to a protein pocket (Shar et al. 2016). They are used when the 3D structure of a protein is mapped and large quantities of data are present. However, such methods are only accurate if the 3D structure of a protein is known, but this could be overcome with predicted 3D protein structures.

One such approach was the study conducted by Jiang et al. (2020), where the structural information of molecules and the predicted structural information of proteins were used, creating two different graphs that were then fed into two graph neural networks (GNN) to obtain their representations. These representations were then concatenated and used to make DTI predictions.

Jiang et al. (2020) utilised the Davis (Davis et al. 2011) and KIBA (He et al. 2017; Tang et al. 2014) datasets, with Davis containing selected entries from the kinase protein family, quantified with K_d values, and KIBA containing entries quantified by a combination of kinase inhibitor bioactivities, K_i, K_d, and IC_{50}, called KIBA score.

Graph neural networks have been widely used in various research fields to solve different problems. A graph made of nodes and edges, irrespective of its size, is passed as the input to the GNN, providing a flexible format to extract in-depth information (Jiang et al. 2020).

The drug graph was constructed using its SMILE notation, which describes its unique chemical structure, taking the atom as nodes and the bonds between them as edges. Then the related adjacency matrix was created. Finally, a selection of node features based on atoms was also used, shown in Table 1.

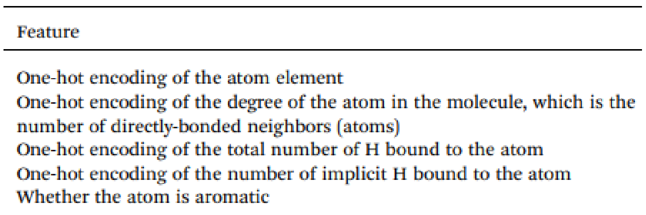
The protein graph was constructed by predicting the protein's contact map, with a threshold of 8Å, from its sequence, using a tool called Pconsc4 (Michel et al. n.d.). A contact map is a 2D representation, usually a matrix, of a protein's 3D structure and can be passed directly to a GNN as an adjacency matrix.

More formally, the contact map of a protein sequence with length L is a 'matrix M with L rows and L columns where each element M_{ij} indicates whether the corresponding residue pair, residue i and residue j, are in contact or not', i.e. have a euclidean distance less than a set threshold, usually 6, 8 or 10 Å.

After getting the protein adjacency matrix, the node features were extracted for further processing. Since the graph was constructed with the residues as the nodes, the features should be selected around them. These properties are shown in Table 2, with PSSM being especially important.

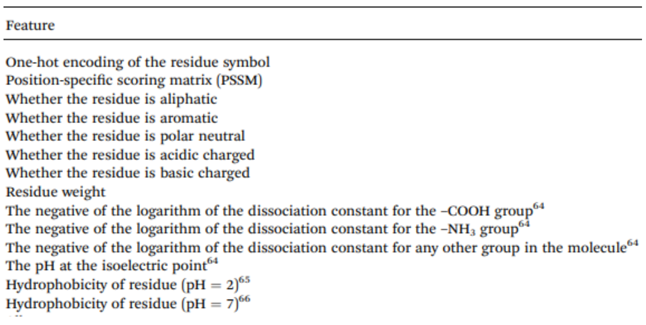
Another interesting study, not for DTI prediction, but for protein function prediction, was that of Gligorijević et al. (2021) where protein sequences and structures were fed into a two-stage architecture model involving a task-agnostic language model and a graph convolutional network (GCN).

The language model was used to extract residue-level features from PDB sequences, and then these together with contact maps with a threshold of 10Å, constructed from the protein structures, were fed into the GCN. Their approach, even if it is trying to solve a different problem, uses a very similar procedure to process a protein's 3D structure.



Feature
One-hot encoding of the atom element
One-hot encoding of the degree of the atom in the molecule, which is the number of directly-bonded neighbors (atoms)
One-hot encoding of the total number of H bound to the atom
One-hot encoding of the number of implicit H bound to the atom
Whether the atom is aromatic

Table 1: Part of a table taken from Jiang et al. (2020) showcasing the atom node features used.



Feature
One-hot encoding of the residue symbol
Position-specific scoring matrix (PSSM)
Whether the residue is aliphatic
Whether the residue is aromatic
Whether the residue is polar neutral
Whether the residue is acidic charged
Whether the residue is basic charged
Residue weight
The negative of the logarithm of the dissociation constant for the –COOH group[64]
The negative of the logarithm of the dissociation constant for the $-NH_3$ group[64]
The negative of the logarithm of the dissociation constant for any other group in the molecule[64]
The pH at the isoelectric point[64]
Hydrophobicity of residue (pH = 2)[65]
Hydrophobicity of residue (pH = 7)[66]

Table 2: Part of a table taken from Jiang et al. (2020) showcasing the residue node features used.

2.7 Valuable Strategies & Concepts

All studies highlighted that the complicated structure of proteins and molecules make the creation of accurate representations, the features that will be passed into the models, one of the hardest parts of the whole process. This is an active area of research in it of itself in computer-aided medicine. (Jiang et al. 2020)

Both Jiang et al. (2020) and Gligorijević et al. (2021) agree that the most efficient way to process a protein 3D structure is with a GCN as it generalises convolutional operations on efficient graph-like molecular representations. GCNs have also shown vast success in problems such as the prediction of biochemical activity of drugs and prediction of interfaces between pairs of proteins (Gligorijević et al. 2021).

3. DESIGN

This chapter will discuss the high-level decisions made to narrow the project's scope.

Although Section 1.2 clearly specified our objectives, it did so on a very high level, giving us the freedom to choose the techniques and methods we employed to achieve them. These decisions were primarily influenced by the background research found in Chapter 2 and could be split into two distinct but interconnected parts that would determine the project's direction.

3.1 Model Decisions

The decisions that needed to be made for the creation of our DTI models were the following:

- Which machine-learning libraries would we use.
- What type of models would we train.
- What models would we train.
- How would the models be categorised.
- What metrics and methods would we use to evaluate the models' predictive performance.
- How would these models be optimised.
- How would we make the models' predictions more interpretable.
- What would we use as our dataset labels.
- How would the dataset be constructed.
- How would we present our findings.

3.1.1 Machine-Learning Libraries

We decided to make use of Scikit-Learn (Pedregosa et al. 2011) and Scikit-Optimize (Head et al. 2021) as they are well-documented learning libraries with thorough tutorials and examples available.

3.1.2 Model Types

To simplify the problem we had first decided to treat it as binary, a drug can bind itself to a protein or not, even though, as mentioned in Section 1.1, the reality is much more complicated and nuanced than that, as anything dealing with the human body. A drug can be highly bound to a particular protein and less so to another, but in the context of this project, we would still consider the drug to bind to both proteins. This was chosen to simplify the problem we would be trying to solve as we felt that trying to predict how much a drug binds to a particular protein would add unnecessary complexity that would be unproductive for the project, especially in the early stages.

However, after starting to build our dataset we realised that some DTIs had their binding affinity attached in addition to the binary binding relationship and we decided to also build regression models just as a proof of concept which could be the subject of some future work.

3.1.3 Chosen Models

The classification models we decided to train were the Dummy Classifier, Logistic Regression, Linear Support Vector Classifier, K-Nearest Neighbour Classifier, Decision Tree Classifier, Random Forest Classifier and the Stochastic Gradient Descent Classifier.

The regression model we decided to train were the Dummy Regressor, Linear Regression, Linear Support Vector Regression, K-Nearest Neighbour Regressor, Decision Tree Regressor, Random Forest Regressor and Stochastic Gradient Descent Regressor.

3.1.4 Model Categories

To tackle the problem we decided to split our models into two distinct categories, baseline and enhanced. The baseline models would serve as one would expect as our baseline, trained on only the selected drug and protein sequence descriptors, and the enhanced models, which would be compared against the baseline ones, trained with the created protein structure embeddings in addition to the selected drug and protein sequence descriptors.

3.1.5 Metrics

To evaluate our models we decided to use numerous metrics for both types of models. This was done not only because it is considered good practice but also to give a more complete view of the models' predictive performance and shortcomings.

The classification models would be evaluated using Accuracy, Precision, Recall, F1 Score and Matthews correlation coefficient (MCC), and the regression models would be evaluated using R2 Score and Negated Mean Absolute Error (MAE), which simply adds a negative sign in front of MAE to make sure that all of our metrics follow the same 'greater is better' principle, making their interpretation easier.

3.1.6 Model Evaluation

To evaluate our models we decided to use holdout test sets and dummy models.

Holdout test sets are subsets of our data that have not been used for either training or validation purposes when training and optimising our model. They are used to estimate a model's real-world performance on previously unseen data.

Dummy models usually predict the most frequent class in the case of classification models and the mean label in the case of regression models, although there are many variations that can be used. These would serve as the random threshold for our models.

3.1.7 Model Optimisation

The model's hyper-parameters would be tuned using the BayesSearchCV function from the Scikit-Optimize (Head et al. 2021) library. BayesSearchCV is very similar to the GridSearchCV and RandomizedSearchCV functions offered by Scikit-Learn (Pedregosa et al. 2011). However, instead of using an exhaustive grid or a random search approach it uses a Bayesian optimisation algorithm which we would argue is much faster than an exhaustive search and much more effective than a random search, particularly when dealing with continuous values.

Just like the Scikit-Learn functions, BayesSearchCV uses cross-validation to optimise a model for a chosen metric we deem the most important for our specific task. The metrics we chose to optimise our models were the F1 and R2 scores for the classification and regression models, respectively.

3.1.8 Model Interpretability

To shine some light into our models' inner workings and to instil some confidence into their predictions, or to at the very least help the user understand what led to a specific prediction, we decided to use *ELI5* (2023) to examine the weights of each model's features and *Local Interpretable Model-Agnostic Explanations (LIME)* (2023) to explain how these features and their respective values led to a specific prediction.

3.1.9 Dataset Labels

For our classification models, the label we would use would be whether a drug or chemical compound can bind with a protein, designated by a 1 and a 0, respectively, and for our regression models, the label would be the logKd value.

3.1.10 Dataset Sources

To create our dataset we would make use of the AlphaFold proteins database (Jumper et al. 2021; Varadi et al. 2022) to extract each protein's structure and sequence and then use PubChem (Kim et al. 2021) to retrieve the DTIs associated with each protein and the corresponding binding relationship and affinity if available.

Once we had the DTIs we would then retrieve each drug's descriptors, again using PubChem, and each protein's sequence descriptors and embeddings, using the ProtR R library (Xiao et al. 2015), already mentioned in Section 2.4, and UniProt database (Consortium 2022) respectively. Finally, we would retrieve each protein's structural embedding if available. Figure 2 nicely summarises our methodology.

3.1.11 Presenting Findings

To present our findings we decided that in addition to the various notebooks we would produce we would also create a very simple *Streamlit* (2023) web application to not only showcase the project's work but also to allow people who might not be computer scientists to use our models and make predictions.

3.2 Embeddings Decisions

In order to construct our protein structure embeddings we decided to create a neural network that would be trained on a specific classification task, making use of protein 3D structures, involving as many proteins as possible in order to maximise the number of created embeddings. Once the model was trained we would then extract the protein structural embeddings from one of the model layers.

The decisions that needed to be made to create our embeddings were the following:

- What deep-learning libraries would we use.
- What would be the classification task.
- How would we use the 3D structures of proteins.
- How would the dataset be constructed.
- What would be the model architecture,
- How would we extract the embeddings,

3.2.1 Deep-Learning Libraries

We decided to use PyTorch (Paszke et al. 2019) and PyTorch-Geometric (Fey and Lenssen 2019) libraries again due to their expanding resources and detailed documentations.

3.2.2 Classification Task

Given that each protein has one or more molecular functions we decided that it would be the perfect classification task to build our embedding model around.

The molecular functions of each protein would be extracted from UniProt (Consortium 2022) and to simplify the problem we decided to only focus on the most prevalent molecular function among our proteins, which was "DNA Binding".

3.2.3 Protein Structure Representation

Inspired by the studies conducted by (Jiang et al. 2020; Gligorijević et al. 2021) we decided to use contact maps with a threshold of 10Å to represent the 3D structures of proteins.

To extract the dense version of the contact maps we decided to use a nanoHUB library (Rafferty et al. 2010) created precisely for this type of task. We would then convert this dense representation into a sparse one and create our protein graphs which would then be passed into our neural network.

3.2.4 Dataset Sources

To create our dataset, as we already mentioned in Subsections 3.2.2 and 3.2.3, we would make use of UniProt (Consortium 2022) and a nanoHUB library (Rafferty et al. 2010) to retrieve each protein's "DNA Binding" molecular function and contact map, respectively.

We would also retrieve each protein's amino acid descriptors using the *Peptides* (2023) R library, amino acid and protein sequence embeddings again through UniProt and protein sequence descriptors and PSSM using the ProtR R library (Xiao et al. 2015). Figure 3 summarises our methodology.

3.2.5 Model Architecture

Inspired by the study conducted by Jiang et al. (2020) we decided to use a very similar architecture, showcased in Figure 1.

Our architecture utilised a protein graph, using a sparse contact map as its adjacency matrix and amino acid descriptors, embeddings and PSSM as its node features, fed into three PyTorch Geometric graph convolutional layers, with ReLU activation and a global mean pooling layer at the end. The protein graph representation would then be flattened using two linear layers and concatenated with the protein sequence descriptors representation which would have passed through three linear layers, again using ReLU activation but also dropout layers. Once concatenated they would be passed through two more linear layers, again with ReLU activation and dropout layers, and a prediction would be made regarding the "DNA Binding" molecular function.

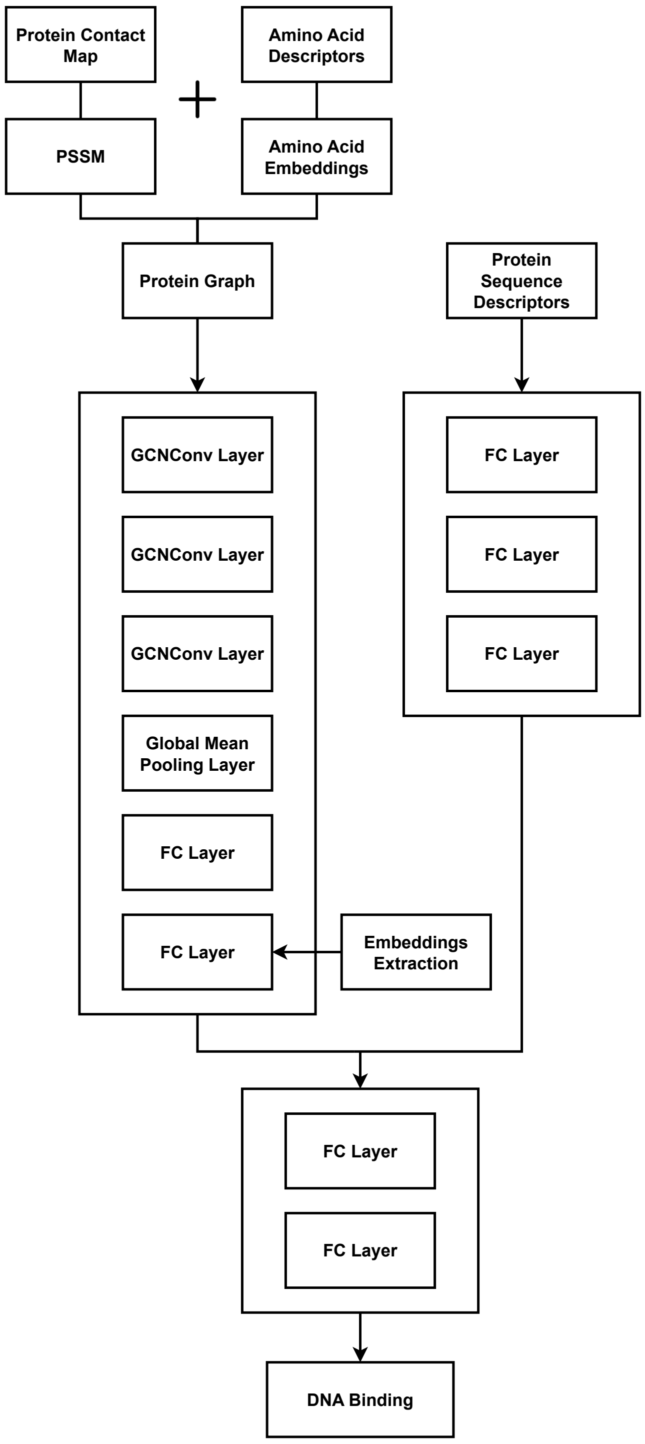
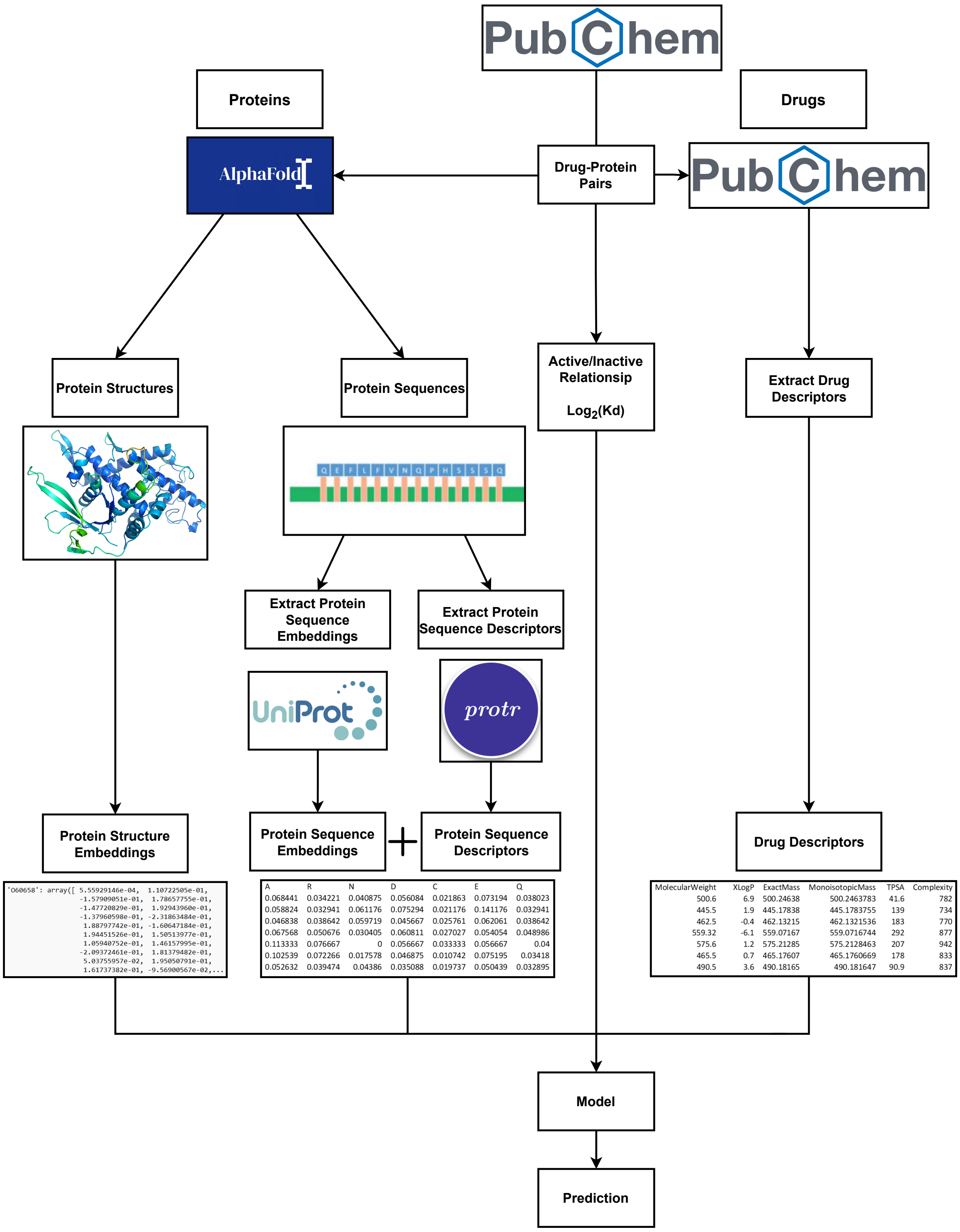


Figure 1: Figure showcasing our embedding model's architecture.

3.2.6 Embedding Extraction

To extract our embeddings from our trained model we would use a forward hook on the second dense linear layer before the concatenation, showcased in Figure 1, and perform a sequential forward pass over all of our proteins. Each forward pass would extract the embedding and place it into a dictionary which we would then use with our models.

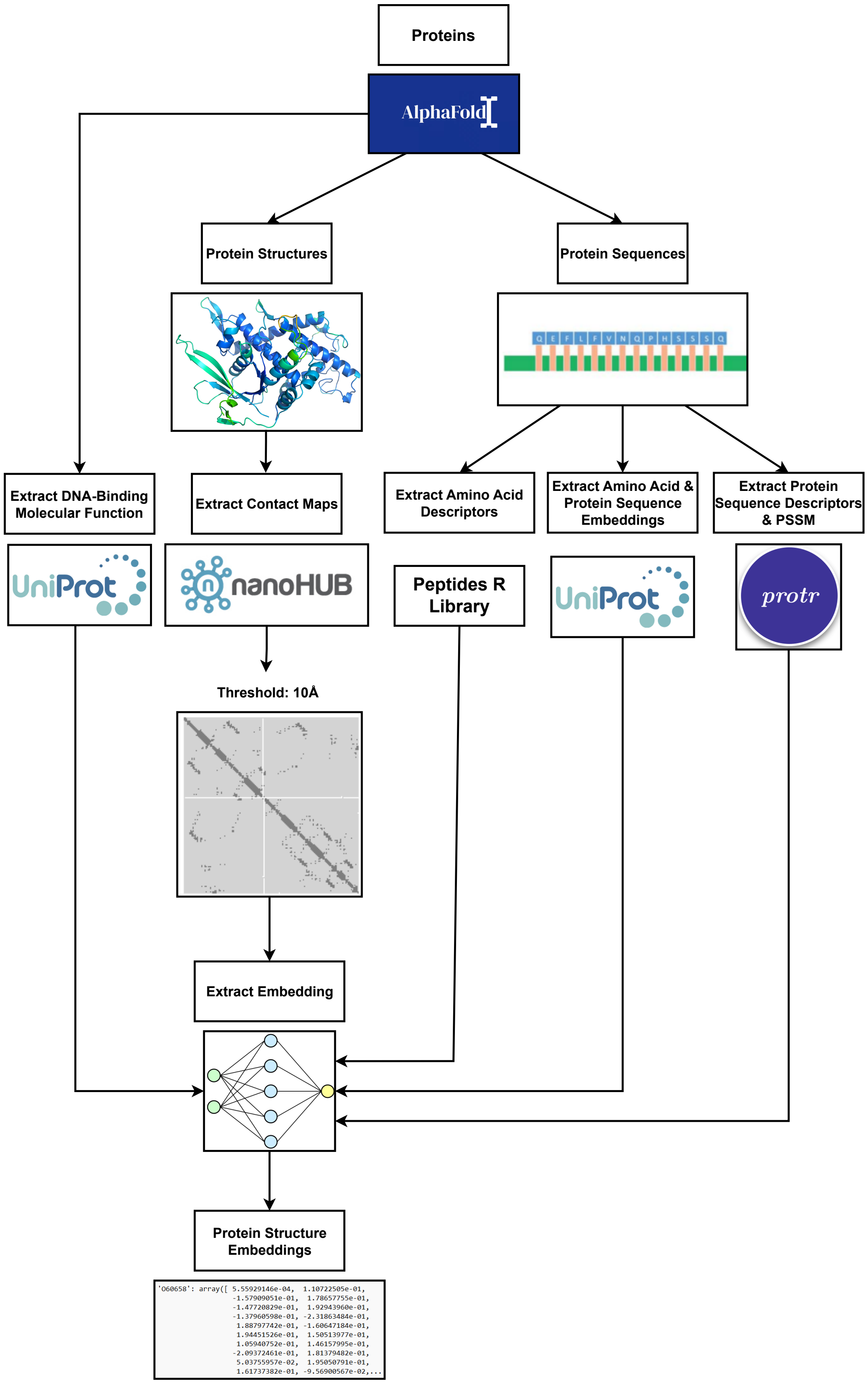

PubChem
Proteins
Drugs
AlphaFold
Drug-Protein Pairs
PubChem
Protein Structures
Protein Sequences
Active/Inactive Relatioship
$\text{Log}_2(\text{Kd})$
Extract Drug Descriptors
Q E F L F V N Q P H S S S Q
Extract Protein Sequence Embeddings
Extract Protein Sequence Descriptors
UniProt
protr
Protein Structure Embeddings
Protein Sequence Embeddings
Protein Sequence Descriptors
Drug Descriptors
Model
Prediction

```
'O60658': array([ 5.55929146e-04,  1.10722505e-01,
                 -1.57909051e-01,  1.78657755e-01,
                 -1.47720829e-01,  1.92943960e-01,
                 -1.37960598e-01, -2.31863484e-01,
                  1.88797742e-01, -1.60647184e-01,
                  1.94451526e-01,  1.50513977e-01,
                  1.05940752e-01,  1.46157995e-01,
                 -2.09372461e-01,  1.81379482e-01,
                  5.03755957e-02,  1.95050791e-01,
                  1.61737382e-01, -9.56900567e-02,...
```

A	R	N	D	C	E	Q
0.068441	0.034221	0.040875	0.056084	0.021863	0.073194	0.038023
0.058824	0.032941	0.061176	0.075294	0.021176	0.141176	0.032941
0.046838	0.038642	0.059719	0.045667	0.025761	0.062061	0.038642
0.067568	0.050676	0.030405	0.060811	0.027027	0.054054	0.048986
0.113333	0.076667	0	0.056667	0.033333	0.056667	0.04
0.102539	0.072266	0.017578	0.046875	0.010742	0.075195	0.03418
0.052632	0.039474	0.04386	0.035088	0.019737	0.050439	0.032895

MolecularWeight	XLogP	ExactMass	MonoisotopicMass	TPSA	Complexity
500.6	6.9	500.24638	500.2463783	41.6	782
445.5	1.9	445.17838	445.1783755	139	734
462.5	-0.4	462.13215	462.1321536	183	770
559.32	-6.1	559.07167	559.0716744	292	877
575.6	1.2	575.21285	575.2128463	207	942
465.5	0.7	465.17607	465.1760669	178	833
490.5	3.6	490.18165	490.181647	90.9	837

Figure 2: Figure showcasing our methodology behind the DTI models.


Proteins
AlphaFold
Protein Structures
Protein Sequences
Extract DNA-Binding Molecular Function
Extract Contact Maps
Extract Amino Acid Descriptors
Extract Amino Acid & Protein Sequence Embeddings
Extract Protein Sequence Descriptors & PSSM
UniProt
nanoHUB
Peptides R Library
UniProt
protr
Threshold: 10Å
Extract Embedding
Protein Structure Embeddings

```
'O60658': array([ 5.55929146e-04,  1.10722505e-01,
                 -1.57909051e-01,  1.78657755e-01,
                 -1.47720829e-01,  1.92943960e-01,
                 -1.37960598e-01, -2.31863484e-01,
                  1.88797742e-01, -1.60647184e-01,
                  1.94451526e-01,  1.50513977e-01,
                  1.05940752e-01,  1.46157995e-01,
                 -2.09372461e-01,  1.81379482e-01,
                  5.03755957e-02,  1.95050791e-01,
                  1.61737382e-01, -9.56900567e-02,...
```

Figure 3: Figure showcasing our methodology behind the embedding model.

4. IMPLEMENTATION

This chapter will discuss how our datasets, embeddings and various models were created using the decisions made in Chapter 3.

4.1 Drug-Target Interactions Dataset

To create our DTIs dataset we started by downloading all the predicted human proteins from the AlphaFold protein structure database (Jumper et al. 2021; Varadi et al. 2022) and retrieving all the protein accession numbers and sequences. Then using these protein accession numbers *PubChem API* (2022) calls were made to retrieve the DTIs associated with each protein.

Some proteins had thousands of DTIs available and we felt including every single one would be not only difficult but also counterproductive. Therefore, we decided to place a limit on the number of DTIs retrieved for each protein. This number was arbitrarily set to 100 and if a particular protein's DTIs exceeded that then we would randomly select 100 of them.

Once we had our DTIs it was time to retrieve the descriptors associated with each drug and protein sequence. For each drug we again made use of *PubChem API* (2022) calls to retrieve every single chemical descriptor stored by PubChem (Kim et al. 2021).

For each protein we used the Protr (Xiao et al. 2015) library as mentioned in Subsection 2.4 to calculate a large selection of protein sequence descriptors from a variety of descriptors families and UniProt (Consortium 2022) to extract each protein's sequence embedding. We decided to use a variety of protein sequence descriptor families instead of focusing on a single one because, as we have already discussed in Subsection 2.4.2, a combination of them can enhance our models' predictive performance.

Once we had everything we needed we performed data cleaning, removing any protein without any DTIs discovered and any dataset entry with missing descriptors, drug or protein related. This process decreased the size of our dataset from 190,028 DTIs to 163,080, with 112,597 classified as having an active relationship and the remaining 50,483 classified as having an inactive one.

As for the entries having binding affinity available, 72,908 had IC_{50} available, but with just 14 of these being classified as inactive. This was clearly not diverse enough and therefore we moved on to the second most common binding affinity which was K_d, with 20,372 entries, 15,007 classified as having an active relationship and 5,365 classified as having an inactive one.

4.1.1 Feature Selection

To improve our models' predictive performance, training times but to also discover the drug and protein sequence descriptors holding the most predictive power we decided to use the recursive feature elimination with cross-validation function (RFECV) offered by Scikit-Learn with a random forest classifier or regressor depending on the dataset we would be using and 5-fold cross-validation.

Before running RFECV we decided to reduce the number of tripeptide protein sequence descriptors using principal component analysis (PCA), a dimensionality reduction method, to reduce their size to a number holding 95% of their variance. As a result, PCA managed to reduce the tripeptide descriptors from 8000 to 2616.

RFECV managed to reduce the features for the classification dataset from 6,474 to 388 and for the regression dataset from 6,474 to 693.

4.2 Holdout Test Sets

As mentioned in Subsection 3.1.6 we decided to use holdout test sets to evaluate our models with previously unseen data. To achieve this we used the train test split function from Scikit-Learn.

Given the nature of our dataset, many proteins can be associated with many drugs, so we could not do the traditional 80/20 split. What we chose to do instead was to take a small subset of our dataset as our test set and remove any proteins and drugs associated with it from the training set. This naturally led to some entries from the dataset not being utilised at all, but we were still left with a substantial amount of training and test data.

For our classification models the training set was 99,705 DTIs and the test set was 816 DTIs and for our regression models the training set was 10,956 DTIs and the test set was 102 DTIs. The same holdout test set will be used for both baseline and enhanced models in order to compare them properly. However, we expect to lose some entries when testing the enhanced models due to not being able to create embeddings for every single protein present in the dataset.

4.3 DNA-Binding Dataset

We again started by using all the predicted human proteins from the AlphaFold protein structure database (Jumper et al. 2021; Varadi et al. 2022) and retrieving all the protein accession numbers and sequences. Then using these protein accession numbers we extracted each protein's molecular functions from UniProt (Consortium 2022). As mentioned in Subsection 3.2.2 we decided to simplify the problem from multi-label to just a single one and that single one was the ”DNA Binding” molecular function as it was the most prevalent.

For each protein we used Protr (Xiao et al. 2015) to retrieve its protein sequence descriptors and PSSM, UniProt (Consortium 2022) for its protein sequence embedding and amino acids embedding, *Peptides* (2023) and more specifically its aaDescriptors function to calculate 66 amino acid descriptors and finally its contact map using a threshold of 10Å using the nanoHUB library (Rafferty et al. 2010).

The amino acid descriptors and embeddings, PSSM, and contact maps were all saved as NumPy files in order to easily pass them on to the neural network during its training.

Once we had everything we needed we performed data cleaning, removing any proteins with missing descriptors and any proteins whose acid descriptors and embeddings, PSSM and contact maps were mismatched, meaning having a different amount of amino acids. This process decreased the size of available proteins from 11,202 to 11,034, with 1,989 classified as having a positive ”DNA Binding” and 9,045 as having a negative ”DNA Binding”.

4.3.1 Feature Selection

Before running RFECV we again decided to try to reduce the large number of tripeptide protein sequence descriptors using PCA with PCA managing to reduce their number from 8000 to 4813. RFECV was then able to reduce the amount of features from 7757 to 144.

4.3.2 *Training & Test Sets*

Given that this dataset and the subsequent embedding model were just a means to an end to create as many embeddings as possible we decided against using a holdout test set as the performance of the embedding model was not of interest to us.

4.4 Embeddings Creation Process

Following the model architecture discussed in Subsection 3.2.5 we trained our neural network using balanced batches of 16 over 50 epochs, which we found were enough for our model to converge. Our model was optimised using Adam with a learning rate of 0.00001 and weight decay of 0.001 and for our loss function we used binary cross entropy loss.

Once the model was trained we used the process discussed in Subsection 3.2.6 to extract our embeddings, each with 256 dimensions, and place them in a dictionary which we then pickled for easy storage and retrieval.

4.5 Model Training & Testing Process

Our training and testing process was quite simple and used consistently for all models.

We would first create a pipeline, containing a standard scaler, used to normalise our features by "removing their mean and scaling to unit variance", and our model. The pipeline would then be passed to BayesSearchCV as the estimator along with the model variables we would like to tune, the metric we would like to optimise the model for and the number of folds to use for cross-validation.

All models were optimised using a 5-fold cross-validation except in the case of the dummy and linear regression models as in their case there was nothing to tune and therefore we did not make use of the BayesSearchCV.

Once the models were optimised we evaluated their performance on the respective test set using 95% confidence intervals of 1000 bootstrapped samples and through the use of confusion matrices in the case of the classification models.

All models, except the dummy ones, had access to one or both of the model interpretability tools mentioned in Subsection 3.1.8 which in combination with the mentioned evaluation processes and a visualisation of the test set errors could be used to investigate further any of the errors. Our training and test process is summarised in Figure 4.

We should also mention that we trained two neural networks, one for classification and one for regression, that used the same architecture as the embedding model, showcased in Figure 3.2.5. Both models were trained using balanced batches of 8 utilising an Adam optimiser with a weight decay of 0.001 and early stopping.

Their only differences were the learning rates and the loss functions used. The classification neural network used a learning rate of 0.00001 and binary cross entropy loss whereas the regression neural network used a learning rate of 0.000001 and L1 loss.

4.6 Streamlit Web App Development

As mentioned in Subsection 3.1.11 the web app was created to summarise our work and to allow users to perform predictions using our trained models, excluding the neural networks which could not be provided due to size constraints.

We should mention that the model interpretability tools we have already discussed in Subsection 3.1.8 were also made

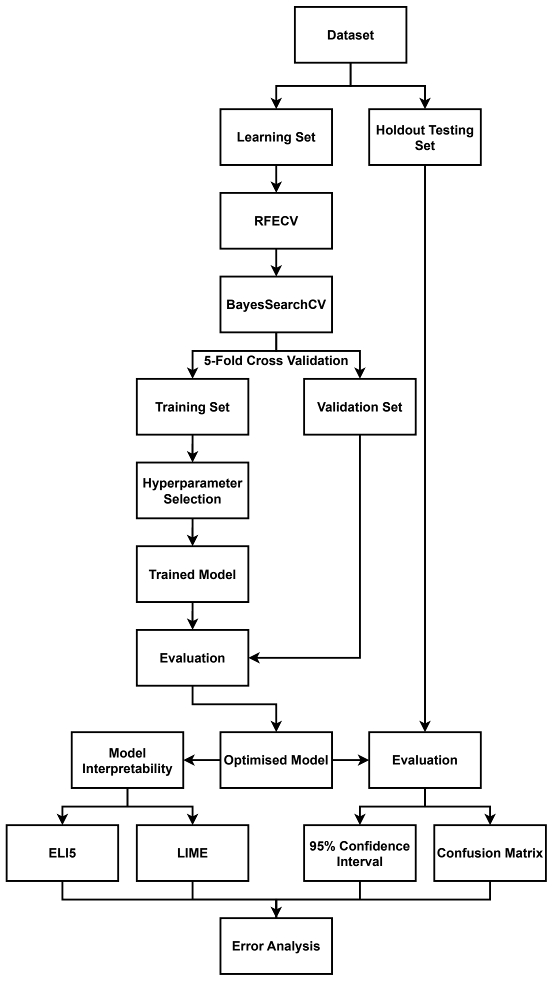


Figure 4: Figure showcasing our model training process.

available but not every model can make use of them. **You can access our web app using this link**

5. RESULTS

This chapter will discuss the performance of our models and any important descriptors discovered.

5.1 Model Performance

5.1.1 *Classification Models*

Tables 3 and 4 showcase the predictive performance of our baseline and enhanced classification models and as we can clearly see our embeddings did not have much of an impact, except possibly in the cases of the logistic regression where they seem to have negatively impacted it and the random forest classifier where they seem to have slightly improved performance as we can see from their metrics, and particularly the MCC score.

Our best models in both cases seem to be the K-nearest neighbour and random forest classifier, although all models, except the dummy ones obviously and the decision tree classifier, achieve close enough performances.

We should mention that all models seem to have particular difficulty in recognising the negative class, meaning an inactive DTI, with actually the number of False Positives surpassing True Negatives which could be in part explained by the class imbalance present in our dataset. We expect that actively tackling this problem in future work will lead to a noticeable improvement in the predictive performance of the models.

We also speculate that our choice of 10 Å instead of the 8 Å used to construct the contact maps by Jiang et al. (2020) may have played a significant role in our embeddings ineffectiveness. However, without any further experimentation this remains an unproven theory.

5.1.2 Regression Models

Tables 5 and 6 showcase the predictive performance of our baseline and enhanced regression models and it could be argued that our embeddings in this case had a much more evident effect, positive in most cases, except obviously in the case of the linear regression model. However, given that the intervals are so spread out a decisive conclusion cannot be confidently reached.

In the case of the enhanced models, our best models seem to be the K-nearest neighbour regressor and the neural network. As for the case of the baseline models given that all our models achieve a negative R2 score, meaning worse than random, evidenced by achieving worse scores than the dummy regressor, we feel that it would be pointless to point out any of them.

There seem to be some promising results for the regression models and it would be interesting to see what performances they could achieve with a more extensive and diverse dataset than the one we used.

5.2 Important Descriptors

As we have previously discussed in Subsection 4.1.1, RFECV was used to find the most important drug and protein sequence descriptors. The intersection between the descriptors found to be important for both classification and regression includes the drug descriptors showcased in Table 7 and the protein sequence descriptors showcased in Table 8.

6. CONCLUSION

This chapter will summarise our work and discuss valuable lessons learned and any potential future work.

6.1 Summary

Drug-target interactions (DTIs) refer to the interactions of chemical compounds and biological targets, proteins in our case, inside the human body (Sachdev and Gupta 2019). They play a crucial role in drug discovery and pharmacology. However, their experimental determination is time-consuming and limited due to funding and the difficulty of purifying proteins (Shar et al. 2016; Wang et al. 2020). Moreover, unwanted or unexpected DTIs could cause severe side effects. Therefore, the creation of in-silico machine learning models with high throughput that can quickly and confidently predict whether thousands of drugs and proteins bind together and how much could be crucial for medicinal chemistry and drug development, acting as a supplement to biological experiments (Shar et al. 2016; Wang et al. 2020).

The project aimed to gather publicly available data on known DTIs and place them into a new curated dataset. Then, using this new dataset, train multiple machine learning models using simple QSAR descriptors derived from a drug's chemical properties and a protein's sequence and 3D structural information to predict whether they bind together. Our models were split into two categories, baseline and enhanced, with baseline using just the drug and protein sequence descriptors and the enhanced using our protein structural embeddings in addition to those descriptors.

A dataset of 163,080 DTIs was gathered using a variety of databases, libraries and biochemical APIs, subsets of which were used to train both our classification and regression models, evaluated using dummy models, holdout test sets and model interpretability tools. Unfortunately, our embeddings seemed to have little effect on our baseline models, which reasonably falls down to our embeddings creation process.

A Streamlit web app was also created to showcase our work and to allow non-technical users to use our models to make predictions. Model interpretability tools were also made available to allow users to better understand what led to a particular prediction by a model.

Even though our embeddings did not have a significant impact, our high-throughput models could still be used to uncover some interesting relationships between drugs and proteins that could be later confirmed or rejected by molecular docking simulations and actual experimental trials.

6.2 Reflection

This project allowed us to work on a fascinating and challenging problem. Even though it could not be called a complete success, we certainly learned a lot, not only about machine learning techniques and best practices but also about bioinformatics in general.

Looking back at the project we should have tried to mitigate the dataset class imbalance, examined further the errors of our models and experimented more with our embedding model's structure and the protein graph we used but also with other extraction methods, possibly utilising each protein's individual pockets instead of its structure as a whole to create the embeddings. However, overall we believe that we have conducted ourselves professionally and responsibly and presented our research in an accurate and unbiased manner.

6.3 Future Work

Multiple project areas could be explored and improved in future work.

The dataset could be improved by expanding it to include more entries with their binding affinity available and trying to reduce the class imbalance through various mitigation techniques.

A larger improved dataset could lead to improved models, but also different training, optimisation, deep learning architectures, and a thorough investigation with the help of professionals with chemical and biological knowledge into the already created models' errors and blind spots could also be used to create better, more accurate, and robust models.

Finally, the embeddings creation process could also be improved by using a more targeted protein binding-pocket analysis approach instead of the one we used that utilised the protein structure as a whole, which proved ineffective.

Acknowledgments. I want to thank my supervisor Dr. Jake Lever for always guiding me in the right direction and for the wonderful conversations we had over the past two years.

Model	Accuracy	Recall	Precision	F1	MCC
Dummy Classifier	0.69 (0.65-0.73)	1.00 (1.00-1.00)	0.69 (0.65-0.73)	0.82 (0.79-0.84)	0 (0-0)
Logistic Regression	0.73 (0.69-0.77)	0.86 (0.83-0.90)	0.77 (0.73-0.81)	0.82 (0.78-0.85)	0.33 (0.24-0.42)
Linear Support Vector Classification	0.74 (0.70-0.77)	0.88 (0.84-0.91)	0.77 (0.73-0.81)	0.82 (0.79-0.85)	0.34 (0.24-0.42)
K-Nearest Neighbour Classifier	0.76 (0.72-0.79)	0.83 (0.79-0.87)	0.82 (0.78-0.86)	0.82 (0.79-0.85)	0.42 (0.33-0.50)
Decision Tree Classifier	0.64 (0.60-0.68)	0.72 (0.67-0.76)	0.75 (0.70-0.80)	0.73 (0.70-0.77)	0.18 (0.09-0.27)
Random Forest Classifier	0.75 (0.71-0.79)	0.93 (0.90-0.96)	0.76 (0.72-0.80)	0.84 (0.81-0.86)	0.36 (0.26-0.44)
Stochastic Gradient Descent Classifier	0.73 (0.69-0.77)	0.86 (0.82-0.90)	0.77 (0.73-0.81)	0.81 (0.78-0.84)	0.32 (0.23-0.41)

Table 3: Testing set (816 DTIs) performance of baseline classification models

Model	Accuracy	Recall	Precision	F1	MCC
Dummy Classifier	0.68 (0.64-0.73)	1.00 (1.00-1.00)	0.68 (0.64-0.73)	0.81 (0.78-0.84)	0 (0-0)
Logistic Regression	0.72 (0.68-0.76)	0.92 (0.89-0.95)	0.73 (0.69-0.78)	0.82 (0.79-0.84)	0.28 (0.18-0.37)
Linear Support Vector Classification	0.72 (0.68-0.76)	0.85 (0.81-0.89)	0.77 (0.73-0.81)	0.80 (0.77-0.84)	0.32 (0.22-0.41)
K-Nearest Neighbour Classifier	0.75 (0.71-0.79)	0.86 (0.82-0.90)	0.79 (0.75-0.83)	0.82 (0.79-0.85)	0.40 (0.31-0.48)
Decision Tree Classifier	0.62 (0.58-0.66)	0.69 (0.64-0.74)	0.74 (0.69-0.79)	0.71 (0.67-0.75)	0.17 (0.07-0.26)
Random Forest Classifier	0.76 (0.72-0.80)	0.92 (0.89-0.95)	0.77 (0.73-0.81)	0.84 (0.81-0.87)	0.41 (0.32-0.50)
Stochastic Gradient Descent Classifier	0.73 (0.69-0.77)	0.92 (0.88-0.94)	0.75 (0.70-0.79)	0.82 (0.79-0.85)	0.31 (0.22-0.40)
Neural Network	0.7	0.7	0.83	0.79	0.36

Table 4: Testing set (740 DTIs) performance of enhanced classification models

Model	Negated MAE	R2
Dummy Regressor	-1.99 (-2.38 to -1.67)	-0.03 (-0.35 to 0)
Linear Regression	-3.73 (-4.52 to -2.95)	-3.26 (-7.18 to -1.40)
Linear Support Vector Regression	-2.35 (-2.85 to -1.80)	-0.73 (-2.41 to -0.02)
K-Nearest Neighbour Regressor	-1.53 (-2.11 to -1.05)	-0.18 (-0.49 to 0.20)
Decision Tree Regressor	-3.12 (-3.91 to -2.38)	-2.39 (-6.16 to -0.87)
Random Forest Regressor	-2.48 (-2.83 to -2.13)	-0.46 (-2.02 to -0.01)
Stochastic Gradient Descent Regressor	-2.50 (-3.12 to -1.93)	-1.13 (-2.93 to -0.21)

Table 5: Testing set (102 DTIs) performance of baseline regression models.

Model	Negated MAE	R2
Dummy Regressor	-1.95 (-2.30 to -1.66)	-0.03 (-0.29 to 0)
Linear Regression	-130.78 (-157.40 to -105.42)	-5231.97 (-11775.09 to -2645.58)
Linear Support Vector Regression	-2.20 (-2.60 to -1.83)	-0.34 (-1.44 to 0.05)
K-Nearest Neighbour Regressor	-1.36 (-1.89 to -0.92)	0.01 (-0.20 to 0.33)
Decision Tree Regressor	-2.77 (-3.65 to -1.93)	-2.35 (-6.89 to -0.61)
Random Forest Regressor	-2.43 (-2.82 to -2.10)	-0.50 (-1.79 to 0)
Stochastic Gradient Descent Regressor	-1.75 (-2.18 to -1.34)	-0.09 (-0.65 to 0.15)
Neural Network	-1.56	0.09

Table 6: Testing set (95 DTIs) performance of enhanced regression models.

MolecularWeight	XLogP	ExactMass	MonoisotopicMass
TPSA	Complexity	HBondDonorCount	HBondAcceptorCount
RotatableBondCount	HeavyAtomCount	AtomStereoCount	DefinedAtomStereoCount
Volume3D	XStericQuadrupole3D	YStericQuadrupole3D	ZStericQuadrupole3D
FeatureCount3D	FeatureAcceptorCount3D	FeatureDonorCount3D	FeatureCationCount3D
FeatureRingCount3D	FeatureHydrophobeCount3D	ConformerModelRMSD3D	EffectiveRotorCount3D
ConformerCount3D	Fingerprint2D		

Table 7: Important drug descriptors for both classification and regression.

CHOC760101.lag4.1	hydrophobicity.Group3	secondarystruct.Group1	prop3.Tr1221
prop5.Tr1221	VS562	Schneider.Xr.N	

Table 8: Important protein sequence descriptors for both classification and regression.

References

AlphaFold Blog (2022). Last accessed: 28-11-2022.
URL: *https://rb.gy/mnqomi*

Aparoy, P., Reddy, K. K. and Reddanna, P. (2012), 'Structure and ligand based drug design strategies in the development of novel 5-lox inhibitors', *Current Medicinal Chemistry* **19**, 3763.
URL: *ncbi.nlm.nih.gov/pmc/articles/PMC3480706/*

Chaudhuri, T. K. and Paul, S. (2006), 'Protein-misfolding diseases and chaperone-based therapeutic approaches', *The FEBS journal* **273**, 1331–1349.
URL: *https://pubmed.ncbi.nlm.nih.gov/16689923/*

Colmenarejo, G. (2003), 'In silico prediction of drug-binding strengths to human serum albumin', *Medicinal research reviews* **23**, 275–301.
URL: *https://pubmed.ncbi.nlm.nih.gov/12647311/*

Consortium, T. U. (2022), 'UniProt: the Universal Protein Knowledgebase in 2023', *Nucleic Acids Research* **51**(D1), D523–D531.
URL: *https://doi.org/10.1093/nar/gkac1052*

Davis, J. L. (2018), 'Pharmacologic principles', *Equine Internal Medicine: Fourth Edition* pp. 79–137.

Davis, M. I., Hunt, J. P., Herrgard, S., Ciceri, P., Wodicka, L. M., Pallares, G., Hocker, M., Treiber, D. K. and Zarrinkar, P. P. (2011), 'Comprehensive analysis of kinase inhibitor selectivity', *Nature biotechnology* **29**, 1046–1051.
URL: *https://pubmed.ncbi.nlm.nih.gov/22037378/*

ELI5 (2023). Last accessed: 16-02-2023.
URL: *https://eli5.readthedocs.io/en/latest/overview.html*

Fey, M. and Lenssen, J. E. (2019), 'Fast graph representation learning with PyTorch Geometric'.

Fridman, L. (2022), 'Deepmind solves protein folding'. Last accessed: 28-11-2022.
URL: *https://youtu.be/W7wJDJ56c88*

Gligorijević, V., Renfrew, P. D., Kosciolek, T., Leman, J. K., Berenberg, D., Vatanen, T., Chandler, C., Taylor, B. C., Fisk, I. M., Vlamakis, H., Xavier, R. J., Knight, R., Cho, K. and Bonneau, R. (2021), 'Structure-based protein function prediction using graph convolutional networks', *Nature Communications 2021 12:1* **12**, 1–14.
URL: *https://doi.org/10.1038/s41467-021-23303-9*

He, T., Heidemeyer, M., Ban, F., Cherkasov, A. and Ester, M. (2017), 'Simboost: a read-across approach for predicting drug-target binding affinities using gradient boosting machines', *Journal of cheminformatics* **9**.
URL: *https://pubmed.ncbi.nlm.nih.gov/29086119/*

Head, T., Kumar, M., Nahrstaedt, H., Louppe, G. and Shcherbatyi, I. (2021), 'scikit-optimize/scikit-optimize'.
URL: *https://zenodo.org/record/5565057*

Jiang, M., Li, Z., Zhang, S., Wang, S., Wang, X., Yuan, Q. and Wei, Z. (2020), 'Drug–target affinity prediction using graph neural network and contact maps', *RSC Advances* **10**, 20701–20712.
URL: *https://doi.org/10.1039/D0RA02297G*

Jumper, J., Evans, R., Pritzel, A., Green, T., Figurnov, M., Ronneberger, O., Tunyasuvunakool, K., Bates, R., Žídek, A., Potapenko, A., Bridgland, A., Meyer, C., Kohl, S. A., Ballard, A. J., Cowie, A., Romera-Paredes, B., Nikolov, S., Jain, R., Adler, J., Back, T., Petersen, S., Reiman, D., Clancy, E., Zielinski, M., Steinegger, M., Pacholska, M., Berghammer, T., Bodenstein, S., Silver, D., Vinyals, O., Senior, A. W., Kavukcuoglu, K., Kohli, P. and Hassabis, D. (2021), 'Highly accurate protein structure prediction with alphafold', *Nature 2021 596:7873* **596**, 583–589.
URL: *https://doi.org/10.1038/s41586-021-03819-2*

Ki Database (2022). Last accessed: 05-12-2022.
URL: *https://pdsp.unc.edu/databases/kidb.php*

Kim, S., Chen, J., Cheng, T., Gindulyte, A., He, J., He, S., Li, Q., Shoemaker, B. A., Thiessen, P. A., Yu, B., Zaslavsky, L., Zhang, J. and Bolton, E. E. (2021), 'Pubchem in 2021: new data content and improved web interfaces', *Nucleic Acids Research* **49**, D1388–D1395.
URL: *https://doi.org/10.1093/nar/gkaa971*

Levinthal, C. (1969), 'Levinthal's paradox'. Last accessed: 28-11-2022.
URL: *https://rb.gy/ydb8o3*

Lo, Y. C., Rensi, S. E., Torng, W. and Altman, R. B. (2018), 'Machine learning in chemoinformatics and drug discovery', *Drug Discovery Today* **23**, 1538–1546.

Local Interpretable Model-Agnostic Explanations (LIME) (2023). Last accessed: 16-02-2023.
URL: *https://lime-ml.readthedocs.io/en/latest/*

Mauri, A., Consonni, V., Pavan, M. and Todeschini, R. (2006), 'Dragon software: An easy approach to molecular descriptor calculations'.
URL: *shorturl.at/oDEFH*

Michel, M., Hurtado, D. M. N. and Elofsson, A. (n.d.), 'Pconsc4: fast, accurate and hassle-free contact predictions'.
URL: *https://doi.org/10.1093/bioinformatics/bty1036*

Ong, S. A., Lin, H. H., Chen, Y. Z., Li, Z. R. and Cao, Z. (2007), 'Efficacy of different protein descriptors in predicting protein functional families', *BMC Bioinformatics* **8**, 1–14.
URL: *https://doi.org/10.1186/1471-2105-8-300*

O'Connor, C., Adams, J. U. and Fairman, J. (2010), 'Proteins are responsible for a diverse range of structural and catalytic functions in cells'. Last accessed: 09-11-2022.
URL: *https://rb.gy/8a6x7e*

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J. and Chintala, S. (2019), 'Pytorch: An imperative style, high-performance deep learning library', pp. 8024–8035.
URL: *https://rb.gy/yni8b1*

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. and Duchesnay, E. (2011), 'Scikit-learn: Machine learning in Python', *Journal of Machine Learning Research* **12**, 2825–2830.

Pence, H. E. and Williams, A. (2010), 'Chemspider: An online chemical information resource', *Journal of Chemical Education* **87**, 1123–1124.
URL: *https://pubs.acs.org/doi/full/10.1021/ed100697w*

Peptides (2023). Last accessed: 10-03-2023.
URL: *https://rdrr.io/cran/Peptides/*

PubChem API (2022). Last accessed: 05-12-2022.
URL: *https://pubchemdocs.ncbi.nlm.nih.gov/pug-rest*

PubChem Fingerprints (2022). Last accessed: 05-12-2022.
URL: *https://rb.gy/p8mpnm*

Rafferty, B., Flohr, Z. C. and Martini, A. (2010), 'Protein contact maps'.
URL: *https://nanohub.org/resources/contactmaps*

Ranganathan, S., Gribskov, M., Nakai, K. and Schönbach, C. (2019), 'Applications, volume 3', *Encyclopedia of Bioinformatics and Computational Biology* **3**, 938–952.
URL: *https://rb.gy/r1nlwt*

Roth, B. L., Lopez, E., Patel, S. and Kroeze, W. K. (2016), 'The multiplicity of serotonin receptors: Uselessly diverse molecules or an embarrassment of riches?', *http://dx.doi.org/10.1177/107385840000600408* **6**, 252–262.
URL: *https://doi.org/10.1177/107385840000600408*

Sachdev, K. and Gupta, M. K. (2019), 'A comprehensive review of feature based methods for drug target interaction prediction', *Journal of Biomedical Informatics* **93**, 103159.

Scheife, R. T. (1989), 'Protein binding: what does it mean?', *DICP : the annals of pharmacotherapy* **23**.
URL: *https://pubmed.ncbi.nlm.nih.gov/2669380/*

Shar, P. A., Tao, W., Gao, S., Huang, C., Li, B., Zhang, W., Shahen, M., Zheng, C., Bai, Y. and Wang, Y. (2016), 'Pred-binding: large-scale protein–ligand binding affinity prediction', *Journal of Enzyme Inhibition and Medicinal Chemistry* **31**, 1443–1450.
URL: *https://pubmed.ncbi.nlm.nih.gov/26888050/*

Shen, H. B. and Chou, K. C. (2008), 'Pseaac: a flexible web server for generating various kinds of protein pseudo amino acid composition', *Analytical biochemistry* **373**, 386–388.
URL: *https://pubmed.ncbi.nlm.nih.gov/17976365/*

Streamlit (2023). Last accessed: 16-02-2023.
URL: *https://streamlit.io/*

Tang, J., Szwajda, A., Shakyawar, S., Xu, T., Hintsanen, P., Wennerberg, K. and Aittokallio, T. (2014), 'Making sense of large-scale kinase inhibitor bioactivity data sets: a comparative and integrative analysis', *Journal of chemical information and modeling* **54**, 735–743.
URL: *https://pubmed.ncbi.nlm.nih.gov/24521231/*

Torrisi, M., Pollastri, G. and Le, Q. (2020), 'Deep learning methods in protein structure prediction', *Computational and Structural Biotechnology Journal* **18**, 1301–1310.

Tunyasuvunakool, K., Adler, J., Wu, Z., Green, T., Zielinski, M., Žídek, A., Bridgland, A., Cowie, A., Meyer, C., Laydon, A., Velankar, S., Kleywegt, G. J., Bateman, A., Evans, R., Pritzel, A., Figurnov, M., Ronneberger, O., Bates, R., Kohl, S. A., Potapenko, A., Ballard, A. J., Romera-Paredes, B., Nikolov, S., Jain, R., Clancy, E., Reiman, D., Petersen, S., Senior, A. W., Kavukcuoglu, K., Birney, E., Kohli, P., Jumper, J. and Hassabis, D. (2021), 'Highly accurate protein structure prediction for the human proteome', *Nature 2021 596:7873* **596**, 590–596.
URL: *https://doi.org/10.1038/s41586-021-03828-1*

Vallianatou, T., Lambrinidis, G. and Tsantili-Kakoulidou, A. (2013), 'In silico prediction of human serum albumin binding for drug leads.', *Expert Opinion on Drug Discovery* **8**, 583–595.
URL: *https://europepmc.org/article/med/23461733*

Varadi, M., Anyango, S., Deshpande, M., Nair, S., Natassia, C., Yordanova, G., Yuan, D., Stroe, O., Wood, G., Laydon, A., Zídek, A., Green, T., Tunyasuvunakool, K., Petersen, S., Jumper, J., Clancy, E., Green, R., Vora, A., Lutfi, M., Figurnov, M., Cowie, A., Hobbs, N., Kohli, P., Kleywegt, G., Birney, E., Hassabis, D. and Velankar, S. (2022), 'Alphafold protein structure database: massively expanding the structural coverage of protein-sequence space with high-accuracy models', *Nucleic Acids Research* **50**, D439–D444.
URL: *https://doi.org/10.1093/nar/gkab1061*

Wang, L., You, Z. H., Li, L. P., Yan, X. and Zhang, W. (2020), 'Incorporating chemical sub-structures and protein evolutionary information for inferring drug-target interactions', *Scientific Reports 2020 10:1* **10**, 1–11.
URL: *https://doi.org/10.1038/s41598-020-62891-2*

Wishart, D. S., Feunang, Y. D., Guo, A. C., Lo, E. J., Marcu, A., Grant, J. R., Sajed, T., Johnson, D., Li, C., Sayeeda, Z., Assempour, N., Iynkkaran, I., Liu, Y., MacIejewski, A., Gale, N., Wilson, A., Chin, L., Cummings, R., Le, D., Pon, A., Knox, C. and Wilson, M. (2018), 'Drugbank 5.0: A major update to the drugbank database for 2018', *Nucleic Acids Research* **46**, D1074–D1082.

Xiao, N., Cao, D.-S., Zhu, M.-F. and Xu, Q.-S. (2015), 'protr/ProtrWeb: R package and web server for generating various numerical representation schemes of protein sequences', *Bioinformatics* **31**(11), 1857–1859.
URL: *https://doi.org/10.1093/bioinformatics/btv042*

Yamanishi, Y., Araki, M., Gutteridge, A., Honda, W. and Kanehisa, M. (2008), 'Prediction of drug–target interaction networks from the integration of chemical and genomic spaces', *Bioinformatics* **24**, i232–i240.
URL: *https://doi.org/10.1093/bioinformatics/btn162*

Yartsev, A. (2022), 'Protein binding of drugs'. Last accessed: 09-11-2022.
URL: *https://rb.gy/r8tmc8*

Zhang, P., Tao, L., Zeng, X., Qin, C., Chen, S. Y., Zhu, F., Yang, S. Y., Li, Z. R., Chen, W. P. and Chen, Y. Z. (2017), 'Profeat update: A protein features web server with added facility to compute network descriptors for studying omics-derived networks', *Journal of Molecular Biology* **429**, 416–425.