# BLRM: A Basic Linear Ranking Model for Protein Interface Prediction

1st Basir Shariat
*Department of Computer Science*
*Colorado State University*
Fort Collins, USA
b.shariat@gmail.com

2nd Don Neumann
*Department of Computer Science*
*Colorado State University*
Fort Collins, USA
factored@rams.colostate.edu

3rd Asa Ben-Hur
*Department of Computer Science*
*Colorado State University*
Fort Collins, USA
asa@cs.colostate.edu

***Abstract*—We consider the problem of prediction of the interfaces of protein-protein interactions, a challenging problem with important applications in drug discovery and design. The standard machine learning approach is to attempt to predict the interface in its entirety. Because of the difficulty of the problem, we propose to treat the problem as a ranking problem and focus on getting at least a few correctly predicted interface residues in the top ranked predictions. Our results demonstrate that a simple linear model out-performs more complicated models that try to solve the corresponding classification problem. The source code is available at https://bitbucket.org/afrasiab/blrm.**

***Index Terms*—Protein-Protein Interface Prediction, Learning to Rank.**

## I. Introduction

Detailed information on protein-protein interactions, and the interfaces through which they occur, is crucial for understanding cellular functions at the molecular level. Many experimental imaging technologies are available that can characterize the structure of a protein or a protein complex at the atomic level, including X-ray crystallography and nuclear magnetic resonance (NMR) spectroscopy. These experimental methods are time consuming and expensive, and a reliable high-throughput technology is not available yet [24]. Therefore, computational methods that can reliably predict protein-protein interaction interfaces are of great value. In this work, we will focus on prediction of the interfaces of protein complexes from their unbound components.

A variety of different methods for this problem have been proposed [3], [10], [27]. Methods for predicting interfaces are typically divided into two major categories: docking, and data-driven methods [27]. Docking methods model the bio-physics and geometry of the interacting partners at the atomic level and produce a set of candidate conformations that minimize an energy function. The interface then can be extracted from docked conformations. While docking methods are potentially more powerful in terms of the resolution of their predictions and non-reliance on any additional data, they are computationally more demanding, and more importantly, finding a near-native conformation among the large list of candidates conformations is not an easy task. For a recent review of docking methods and tools see [25]. The three major categories of data-driven methods are homology- or template-based methods, co-evolution methods and machine learning methods. Homology-based methods operate under the assumption that the interface and other properties of proteins are conserved among homologous proteins and they use the experimentally available interface of a given protein's homologs to produce a prediction [13], [22], [28], [29]. These methods provide reliable results when close homologs are available [27]. Co-evolution methods assume that the interface residues co-evolve, and use large multiple sequence alignments to determine the interface residues [18]. Machine learning-based methods usually model the problem as a binary classification problem where a residue or a pair of residues from different partners is classified as belonging to the interface or not, based on a collection of features of the residues. These methods rely on the available experimentally determined data for training the models to learn the patterns that characterize interface residues [1], [2], [6]–[8], [11], [15], [16], [19], [20].

Machine learning methods for interface prediction can be categorized based on the types of data and features they use for prediction. Some methods use only sequence-based features or structural features that are predicted from the sequence [1], [2], [20] while other methods use both sequence- and structure-based features [1], [11], [20]. As can be expected, methods that use both sources of information exhibit better performance. Common sequence-based features include sequence conservation, predicted accessible surface area and residue properties such as hydrophobicity and charge. Common structural features are accessible surface area, torsion angles, residue depth, the protrusion index and various shape descriptors (see [3], [27] for a review of the most commonly used features for this task). Among machine learning methods, it is important to distinguish between partner-specific and partner-independent methods. Partner-independent methods classify residues of a protein as being part of some interface or not, independent of the other proteins it interacts with, while partner-specific methods consider the interacting partners. It has been observed that partner-specific methods are more accurate [1], [10], [20], [27]. Machine learning methods also vary in terms of the type of model used. A variety of methods have been applied to this problem including neural networks [2], support vector machines [1], random forests [20], [23], naive Bayes [19], and

This work is funded by NSF ABI grant # 1564840

graph convolutional neural networks [11]. In this paper, we present a simple linear model that is designed for ranking the pairs of residues of a given complex in such a way that it is very likely to have an interacting pair of residues in the few top-ranked elements of the list.

Successful prediction of interfaces from structure using machine learning critically depends on the availability of sufficient and reliable training data. Although the protein data bank includes tens of thousands of proteins structures, one needs the structure of partners in both bound and unbound conformations for a complex to be used for training of a model. The Docking Benchmark Dataset (DBD) [26] and ProPairs [14] are the two major non-redundant benchmark datasets that are commonly used for training and evaluation. These datasets are quite small—version 5.0 of DBD contains only 230 complexes. Because of this shortage of data, virtually all machine learning methods work at the residue-level where all the pairs of residues in a complex or individual residues of a single protein can be considered as positive and negative examples.

In this work, we demonstrate the value of formulating the interface prediction problem as a ranking problem, showing that a simple linear model outperforms more complicated non-linear models in the task of producing a ranking that contains good predictions near the top of the list.

## II. Methods

We consider interface prediction from the perspective of learning to rank objects with respect to their relevance to the given task, a problem which is well-studied in the area of information retrieval [17]. Our objective is to learn a ranking of our labeled training examples such that the top predictions are positive examples. The motivation for formulating interface prediction this way comes from the extreme imbalance between the number of interacting versus non-interacting pairs of residues, which can easily lead to classifiers that appear to work well, and yet contain very few top-ranking positive examples. We believe that most users will be interested in looking primarily at the top predictions of the classifier, so it is important to make sure that those are good predictions. Nearly all previous studies that model the problem as a classification problem, use the area under the receiver operating characteristic curve (AUC-ROC) as their measure of performance. However, given the class imbalance for this problem, a method with a high AUC-ROC score might still have a large number of false positives in its top-ranking predictions, making the predictions not useful from the user's perspective. For example, the recently published state-of-the-art BIPSPI method [20] has an AUC-ROC of $0.94$ while its AUC-PRC (area under the precision recall curve) is very small. This issue is especially important when making partner-specific predictions, where the degree of imbalance is much more pronounced than in partner-independent binding site prediction. This is an indicator that the problem of binding site prediction is far from being solved. Because of that, we propose to focus on the easier problem of finding a few pairs of interacting residues rather than predicting the entire interface. For that reason, we use the Rank of the First Positive Prediction (RFPP) [1] as the measure of accuracy. Those few top predictions can also be used as input to docking tools such as HADDOCK, which accepts a set of hints for the identity of the residues that participate in the interaction between the docked structures [9]. This set is used to prioritize the search space of docking solutions which increases the chance of finding near-native solutions.

### A. Problem Statement

We consider the problem of protein interface prediction at the residue level. In the partner-specific version of the problem, each example is a pair of residues with a label that indicates if the two residues are interacting. For the $i$th complex in our dataset, we define $P_i$ as the set of pairs of residues that are interacting. Each member of the set is a pair $(l, r)$, where $l$ is a residue from the ligand, and $r$ is a residue from the receptor. Every pair of residues in complex $i$ that is not known to interact belongs to $N_i$. The pair of sets $(P_i, N_i)$ are the *bags* of examples associated with complex $i$. Let $m$ be the number of bags (complexes) in our labeled dataset, and we denote the vector representation of a pair of residues indexed by $j$ as $x_j$. Our objective is to learn a linear ranking function

$$f(x) = w^{\intercal} x, \tag{1}$$

where $x$ is a vector representing a pair of residues, and $w$ is a vector of the same dimensionality as $x$. In this work, we will search for a function $f(x)$ such that for each $i = 1, \ldots, m$, $\exists p \in P_i$ such that $\forall n \in N_i$ we have that $f(x_p) > f(x_n)$. This is a weaker constraint than requiring that each example be classified correctly, and also weaker than the constraint of requiring all positive examples within a complex to be ranked higher than all negative examples.

### B. Pair Representation

Since each input object is a pair of vectors and the order of vectors in the pair is arbitrary, we need to devise a method to build a vector representation from a given pair that is order invariant. There are obvious candidates such as the outer product among others that have been used in previous studies. The outer product produces a rich representation of the pair and was used in the context of pairwise kernels with support vector machines [1], [4]; however, when not working with kernels, it is quadratic in the number of features used to represent a residue. In this study, we use the following pair representation that works well for this problem:

$$K_{CSP}((l, r)) = (l + r) || (l \odot r),$$

where $||$ denotes vector concatenation and $\odot$ denotes element-wise product. The size of the new representation is twice the size of the original vectors, and can be computed very efficiently. In order to shed some light on the intuition behind the design of this representation one can consider the element-wise summation and product as the logical AND and OR operators for each feature. We experimented with other pair representations, and none of them gave higher accuracy with our method.

### C. BLRM Formulation

The proposed model which we call the Basic Linear Ranking Model (BLRM for short) is a large margin linear ranking model which assigns a score to each pair of residues using Equation (1). The objective of the BLRM is to find a vector $w$ that maximizes the margin of misranking across all input bags while pushing a few of the positive objects to the top of the ranking. The cost function at epoch $t$, which consists of a regularization term and misranking cost, is defined as:

$$C_t = \frac{\lambda}{2}||w_t||^2 + \frac{1}{m}\sum_{i=1}^{m}\frac{1}{k}\sum_{n\in N_i^*}\frac{1}{|P_i|}\sum_{p\in P_i}\frac{l(w_t : x_n, x_p)}{j_p}, \quad (2)$$

where $N_i^*$ is the set of the $k$ highest scoring negative example from bag $i$ at epoch $t$, $j_p$ is the rank of positive example $p$ among other positive examples based on its current score, and $l$ is the hinge loss function:

$$l(w_t : x_n, x_p) = \max(0, 1 - (w_t^{\intercal} x_p - w_t^{\intercal} x_n)).$$

The cost function is designed in such a way that it discourages misranking of a positive example based on its score. The higher the score of a positive example, the higher the cost of its misranking, which is implemented in by the presence of $j_p$ in the denominator of the hinge loss in Equation 2. We propose an approximate stochastic sub-gradient descent algorithm for this problem as described next.

### D. Algorithms

*1) Batch BLRM:* For solving the above optimization problem we present two approximate sub-gradient solvers. The idea of solving large margin problems with sub-gradients has been around since the seminal work of Shalev-Shwartz [21] . Here, we present two different solvers for this problem. The sub-gradient of $C_t$ with respect to $w_t$ is:

$$g_t = \lambda w_t + \frac{1}{m}\sum_{i=1}^{m}\frac{1}{k}\sum_{n\in N_i^*}\frac{1}{|P_i|}\sum_{p\in P_i}\mathbb{1}[w_t^{\intercal}(x_p - x_n) < 1]\frac{(x_p - x_n)}{j_p}, \quad (3)$$

where $\mathbb{1}[\cdot]$ is the indicator function and its output is one when its argument is true and zero otherwise. The weight vector is then updated by taking a step in the direction opposite to the sub-gradient:

$$w_{t+1} = w_t - \frac{1}{\lambda t^2} g_t, \quad (4)$$

where we use a step size $\frac{1}{\lambda t^2}$.

Algorithm 1 is the pseudocode of the batch BLRM. In order to develop a geometric intuition about the algorithm, we draw the reader's attention to argument of the indicator function. This simply states that if a positive example is scored lower than one of the $k$ highest scoring negative examples, we perform an update on $w_t$ by adding the difference of the positive and negative example to $w_t$, making it more likely that this pair of examples is correctly ranked.Next, we mention a few additional points regarding this algorithm. Note that $N_i^*$ may change at each iteration. After the update to $w$ we might get a new set of $k$ negative examples that are scored above all the other negative examples and the order of positive examples may change as well. We also want to draw the reader's attention to the multiplier in the update to the gradient $g_b$ at line 18. This multiplier makes the updates to the gradient smaller for the lower scoring positive examples that have been misranked. This will guarantee a consistent direction of update that makes it unlikely for the algorithm to misrank a previously well-ranked positive example, and puts stronger emphasis on the top of the list. The quadratic learning rate was found to be crucial for fast convergence of the algorithm. BLRM is a pocket-algorithm where the model that is performing the best on the validation set is chosen as the final model. The complexity of the algorithm is $O(Tm\sum_{b\in B}|P_b|k)$, where $T$ is the number of epochs; if $Tk < \sum_{b\in B}|N_b|$ (which is the case for our dataset), the running time is sublinear with respect to the total number of residue pairs in all the complexes.

**Algorithm 1** Batch BLRM

```
 1: $B : \{(P_1, N_1), ...(P_m, N_m)\}$
 2: $\lambda$: Regularization parameter.
 3: $T$: Number of epochs.
 4: $k$: Number of top scoring negative examples to consider.
 5: procedure TrainBLRM(B, $\lambda$, $T$)
 6:     $w$ is initialized randomly or zeros.
 7:     $j \leftarrow 0$
 8:     for each epoch t do
 9:         $\eta = \frac{1}{\lambda t^2}$
10:         $g_b \leftarrow \lambda w$
11:         for each bag $b = (P_b, N_b) \in B$ do
12:             $S_{N_b} \leftarrow$ sorted list of the of negative examples of bag b by their score.
13:             $S_{P_b} \leftarrow$ sorted list of the of positive examples of bag b by their score.
14:             for each $p \in S_{P_b}$ do
15:                 $j \leftarrow j + 1$
16:                 for each $n \in$ first $k$ elements of $S_{N_b}$ do
17:                     if $w^T p$ < 1+$w^T n$ then
18:                         $g_b \leftarrow g_b - \frac{1}{j}(x_p - x_n)$
19:                     end if
20:                 end for
21:             end for
22:         end for
23:         $w \leftarrow w - \frac{\eta}{m} g_b$
24:     end for
25:     return $w$
26: end procedure
```

*2) Online BLRM:* In the online version of the algorithm, instead of accumulating the gradient from all the bags we update $w$ one bag at a time. In each iteration, we randomly select a bag $b$ from the set of bags without replacement and perform the update using only examples from that bag. For the online BLRM we need to move the initialization of $g_b$ at line 10 of algorithm 1 and the update of $w$ at line 23 to the inner loop.

*3) Ensemble BLRM:* We also explore an ensemble of BLRMs (either online or batch), in which a different sample of negative examples is used for training each member of the ensemble. The overall score assigned to each object is the average of the scores assigned by the members of the ensemble. In order to make the scores comparable, we normalize the weight vectors of each BLRM after training.

## III. Evaluation

### A. Datasets.

In our experiments we used the data from Version 5 of the Docking Benchmark Dataset, which is the standard dataset for assessing docking and interface prediction methods [26]. These complexes are a carefully selected subset of structures from the Protein Data Bank (PDB), and contain both bound and unbound forms of the proteins in each complex. Our features are computed from the unbound form, since proteins can alter their shape upon binding, and the labels are derived from the structure of the proteins in complex. As in previous work [1], [11], [20], two residues from different proteins are considered part of the interface if any non-Hydrogen atom in one is within 6Å of any non-Hydrogen atom in the other when in complex. For our test set we used the 54 complexes that were added since version 4.0 of DBD. Because in any given complex there are vastly more residue pairs that don't interact than those that do, we downsampled the negative examples in the training set to obtain a 20:1 ratio of negative and positive examples. Downsampling was done based on the distance of non-interacting residues rather than uniform random selection among all the negative examples as in our previous work [1], [11]. In order to have our negative examples sampled across all the non-interacting pairs from all the volume of partner proteins we computed the distance of all non-interacting pairs and created a quantization of these pairs into 50 buckets based on their distance. Samples were drawn from each bucket based on the 20:1 ratio uniformly. This turns out be very effective in the performance of our model as shown later. Dataset sizes are shown in Table I.

| Dataset | Number of Complexes | Positive Examples | Negative Examples |
|---|---|---|---|
| DBD 4.0 Complexes | 174 | 15,851 (0.15%) | 10,326,914 (99.85%) |
| DBD 5.0 New Complexes | 54 | 4,782 (0.1%) | 4,737,222 (99.9%) |

TABLE I: Number of complexes and examples in the Docking Benchmark Dataset. For training we downsample the negative examples for an overall ratio of 20:1 of negative to positive examples.

### B. Features.

For each residue in the dataset we computed a set of features from the protein's sequence and structure. We used the same features used in our earlier work [1], as summarized next. Protein sequence alone can be a good indicator of the propensity of a residue to form an interface, because each amino acid exhibits unique electrochemical and geometric properties. Furthermore, the level of conservation of a residue in alignments against similar proteins also provides valuable information, since surface residues that participate in an interface tend to be more conserved than surface residues that do not. The identity and conservation of a residue are quantified by 20 features that capture the relative frequency of each of the 20 amino acids in alignments to similar proteins. As in our previous work we used a sequence window of size 21 centered around the residue of interest [1].

In addition to these sequence-based features, we also incorporated several features computed from the structure. These include a residue's surface accessibility, torsion angles, secondary structure, a measure of its protrusion, its distance from the surface, and the composition of amino acids within 8Å in two directions—towards the residue's side chain, and in the opposite direction.

### C. Performance Measures

In addition to the AUC-ROC, we evaluate performance using the rank of the first positive prediction (RFPP), first introduced in our earlier work [1]. As we mentioned earlier, RFPP is a relevant measure in this setup since we are interested in quantifying how well the method does putting a few positive examples at the top of the ranking produced by the method.

### D. Training, validation, and testing

Five-fold cross-validation on the complexes of DBD 4.0 was used to perform an extensive search over the space of possible hyperparameters and different flavors of our model. Aside from $\lambda$, we also did a search over $k$, learning rate, different choices of down sampling and pair representations. The optimal value of $\lambda$ turned out to be $10^{-7}$. Values of $k$ between 100 to 200 resulted similar performance and we chose $k = 100$ for computational efficiency. The weight vector was initialized to zero; random initialization had no significant effect on the performance of the model.

Training times roughly vary from 18-20 minutes. All the experiments were performed on a XeonE5-2620v3 @2.2GHz with 20 cores and 128 GB RAM. An implementation based on TensorFlow yielded training times of 5-8 minutes.

### E. Results

Results comparing different methods are summarized in Table II. We report accuracy using three different measures: AUC-ROC, RFPP, and PoI-RFPP, which stands for part-of-interface RFPP. PoI-RFPP is the lowest rank of a pair of residues such that both are part of the interface, although might not interact directly. Median RFPP and PoI-RFPP are the median of RFPPs and PoI-RFPPs of all the new complexes in DBD 5.0. We compare different flavors of the BLRM (online/ensemble/batch) with PAIRpred [1], BIPSPI [20], and Graph Convolutional Networks (GCNs) [11]. We observe that the online ensemble-BLRM is doing better than all the other

| Model | Performance Metrics | | |
|---|---|---|---|
| | Median RFPP | Median PoI-RFPP | Median AUC-ROC |
| Online-BRLM | 18 | 10 | 0.835 |
| Batch-BRLM | 21 | 14 | 0.841 |
| Online Ensemble BLRM | **16** | **9** | 0.837 |
| PAIRpred [1] | 21 | 10 | 0.866 |
| GCN [11] | 42 | 30 | 0.898 |
| GCN Features + BLRM | 25 | 18 | 0.846 |
| GCN + BLRM Cost-Function | 24 | 17 | 0.841 |
| BIPSPI [20] | 23 | 14 | **0.942** |

TABLE II: Median RFPP, PoI-RFPP and area under the receiver operating characteristic curve (AUC-ROC) of all the new complexes in DBD 5.0. PoI-RFPP stands for part-of-interface RFPP and is the lowest rank of a pair of residues that are both part of the interface but they may not necessarily interact directly. Results shown are the average over ten runs with different random seeds for methods that have random choices in their training process. Bold faced values indicate best performance for each measure.

models in terms of RFPP and PoI-RFPP, and produces the state-of-the-art results for this dataset, while BIPSPI [20] is doing better in terms of AUC-ROC. It is worth mentioning that all the methods have AUC-PRC of at most $10^{-2}$, demonstrating the fact that AUC-ROC can be misleading when it comes to imbalanced datasets. We also note that the BLRM's success with respect to the RFPP comes at the expense of a lower AUC-ROC. All results except for BIPSPI and PAIRpred are the mean of ten runs with different random number generator seeds. The standard deviation of the RFPP and PoI-RFPP for different runs was less than two, and less than $0.005$ for the AUC-ROC, except for the model which was trained with GCN Features and the BLRM cost function, which had a standard-deviation around five for RFPP and PoI-RFPP. The standard deviation of ensemble online BLRM was 1 for RFPP and PoI-RFPP, and $0.003$ for AUC-ROC. The online version of the BLRM is performing better than the batch version, which is expected since stochastic gradient decent is proven to converge faster and avoid over fitting better [5]. The ensemble of online BLRMs, on the other hand produces more robust results (smaller standard deviation in RFPP for different runs), and also gives better overall performance, which is expected since the negative examples for each BLRM are drawn independently and the model as a whole has been exposed to more data (the down-sampling is performed prior to the training). Figure 1 provides histograms of the RFPP and AUC-ROC for the DBD 5.0 test set.

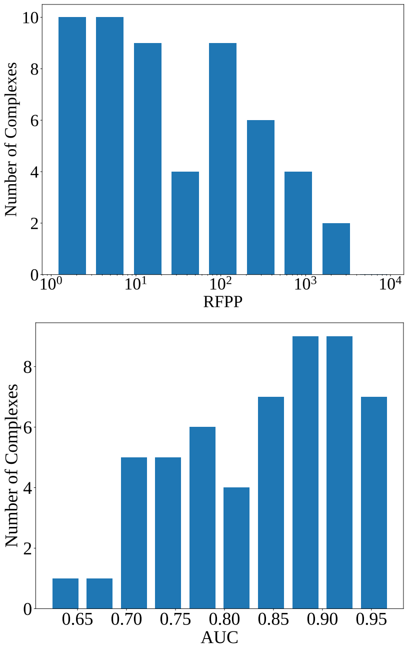

Fig. 1: **Top Row:** Histogram of RFPPs for the new complexes of DBD 5.0 (test set). **Bottom Row:** Histogram of AUC-ROCs for the new complexes of DBD 5.0 (test set).

Graph convolutional neural networks are a powerful tool for learning representations of complicated structured objects [11]. This motivated us to use the learned GCN representations with the BLRM. This was done by removing the last layer of the model and using the vector representation of the previous layer as the feature representation of examples (GCN features + BLRM in Table II). We observe that the BLRM was able to significantly improve the RFPP of the GCN. We also changed the cost function of the GCN to the BLRM cost function, which provided further improvement in the RFPP (GCN + BLRM cost function in Table II).

We visualized the high scoring residues on each protein by taking the average of scores with respect to all the other residues in its partner. The examples in Figure 2 illustrates the performance of the BLRM on a complex for which it does well (3A4S), and a complex for which it does not (BAAD). Smoothing the score of a pair of residues by averaging over the score of its neighbours as in previous work improved the AUC-ROC, but degraded the RFPP as also noted elsewhere [1].

We also tested the online ensemble-BLRM that was trained on DBD 5.0 on eight CAPRI targets [12] (see The Table III). We observe that the BLRM is doing well with respect to the RFPP except for one complex. For comparison, we also ran BIPSPI, which is showing better AUC-ROC scores, but much worse RFPPs.

*a) Feature Importance:* We looked into the contribution of each type of feature to the performance of the BLRM. To do so, we removed one feature group at a time and recorded the increase in the median RFPP in the test set. The results in Table

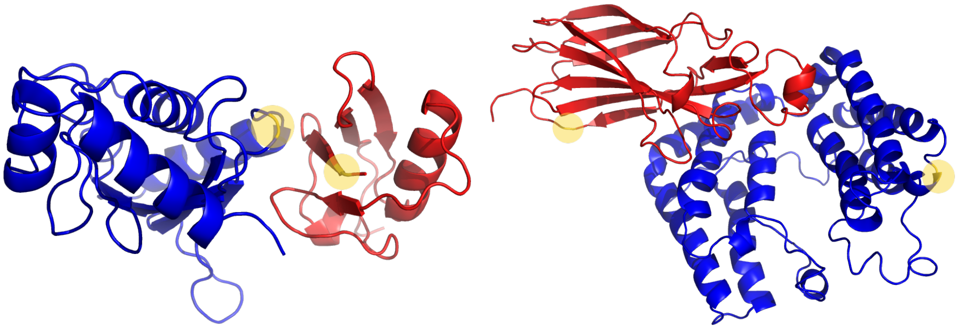

Fig. 2: **Left:** The complex **3A4S** from the Docking benchmark 5.0 whose RFPPs is 1. **Right:** The complex BAAD from the Docking benchmark 5.0 whose RFPP is 2047. The ligand and receptor are shown in red and blue, respectively. The highest scoring residue on each protein is highlighted in yellow.

| PDB ID. | CAPRI Target ID | Ligand backbone RMSD (Å) | Receptor backbone RMSD (Å) | RFPP | AUC-ROC | BIPSPI RFPP | BIPSPI AUC-ROC |
|---|---|---|---|---|---|---|---|
| 4G9S | T58 | 0.3 | 0.7 | 2 | 0.853 | 3 | 0.924 |
| 3U43 | T50 | 0.5 | 0.6 | 2 | 0.827 | 1 | 0.954 |
| 4EEF | T56 | 0.7 | 0.6 | 4 | 0.792 | 192 | 0.796 |
| 2WPT | T41 | 2.0 | 0.7 | 4 | 0.746 | 3 | 0.937 |
| 3E8L | T40 | 0.2 | 0.4 | 4 | 0.858 | 253 | 0.908 |
| 3FM8 | T39 | 0.0 | 1.6 | 16 | 0.805 | 816 | 0.839 |
| 3BX1 | T32 | 2.0 | 0.4 | 23 | 0.866 | 234 | 0.882 |
| 3R2X | T50 | 0.5 | 0.6 | 240 | 0.809 | 903 | 0.834 |

TABLE III: RFPP and AUC of 8 CAPRI targets for the BLRM and BIPSPI. The results obtained by training an ensemble of 10 online BLRMS on DBD 5.0. The RMSD values indicate the amount conformational change upon binding.

IV summarize this experiment, and demonstrate that removal of each feature group yielded a large increase in RFPP, confirming their relevance. Sequence profiles are the most important set of features, and we note that the window size used in computing the profiles is crucial for the BLRM's performance. A window size of 11 instead of 21 gave an RFPP of around 25 instead of 18 for the online BLRM. The next most important features are those computed using DSSP, which include the relative accessible surface area, whose importance is obvious for this problem.

| Feature Group | RFPP after removal |
|---|---|
| All features | 18 |
| Sequence Profiles | 35 |
| rASA and secondary structure | 34 |
| Half-Sphere Amino Acid Exposure | 33 |
| Protrusion Index | 31 |
| Residue Depth | 27 |

TABLE IV: Feature Importance Analysis: Every feature group was removed from the set of features and RFPP was computed without it using the online BLRM. Baseline performance with all features is reported in the first line.

*b) Performance by difficulty class:* We evaluated the performance of the BLRM by difficulty class as annotated in Docking benchmark 5.0, where difficulty is assessed by the amount of conformational change upon binding (Table V). Interestingly, performance does not degrade by much for the "hard" class, suggesting that the classifier is not strongly affected by conformational change.

| Category | Number of complexes with RFPP below the median | Percentage in the training dataset |
|---|---|---|
| Easy | 8 out of 15 (53.4%) | 16.6% |
| Medium | 4 out of 7 (67.2%) | 15.5% |
| Hard | 15 out of 32 (47.9%) | 67.8% |

TABLE V: Performance of the BLRM by level of difficulty, determined by the degree of conformational change upon binding.

## IV. Conclusions

In this paper we presented a method for interface prediction that models the problem as a ranking problem, and designed to provide high quality top predictions. On this more limited version of the problem the proposed large margin ranking method (the BLRM) performed better than existing state-of-the-art methods. However, even this simpler problem is far from solved.

## Acknowledgment

We thank the authors of the BIPSPI method for their help in running their code framework.

# References

[1] Fayyaz ul Amir Afsar Minhas, Brian J. Geiss, and Asa Ben-Hur. PAIRpred: Partner-specific prediction of interacting residues from sequence and structure. *Proteins: Structure, Function, and Bioinformatics*, 82(7):1142–1155, 2014.

[2] Shandar Ahmad and Kenji Mizuguchi. Partner-aware prediction of interacting residues in protein-protein complexes from sequence data. *PLoS One*, 6(12):e29104, 2011.

[3] Tristan T Aumentado-Armstrong, Bogdan Istrate, and Robert A Murgita. Algorithmic approaches to protein-protein interaction site prediction. *Algorithms for Molecular Biology*, 10(1):7, 2015.

[4] Asa Ben-Hur and William Stafford Noble. Kernel methods for predicting protein–protein interactions. *Bioinformatics*, 21(suppl_1):i38–i46, 2005.

[5] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[6] Huiling Chen and Huan-Xiang Zhou. Prediction of interface residues in protein–protein complexes by a consensus neural network method: test against nmr data. *Proteins: Structure, Function, and Bioinformatics*, 61(1):21–35, 2005.

[7] Sjoerd J de Vries and Alexandre MJJ Bonvin. CPORT: a consensus interface predictor and its performance in prediction-driven docking with HADDOCK. *PloS one*, 6(3):e17695, 2011.

[8] Sjoerd J de Vries, Aalt DJ van Dijk, and Alexandre MJJ Bonvin. WHISCY: what information does surface conservation yield? application to data-driven docking. *Proteins: Structure, Function, and Bioinformatics*, 63(3):479–489, 2006.

[9] Sjoerd J De Vries, Marc Van Dijk, and Alexandre MJJ Bonvin. The HADDOCK web server for data-driven biomolecular docking. *Nature protocols*, 5(5):883, 2010.

[10] Reyhaneh Esmaielbeiki, Konrad Krawczyk, Bernhard Knapp, Jean-Christophe Nebel, and Charlotte M Deane. Progress and challenges in predicting protein interfaces. *Briefings in bioinformatics*, 17(1):117–131, 2015.

[11] Alex Fout, Jonathon Byrd, Basir Shariat, and Asa Ben-Hur. Protein interface prediction using graph convolutional networks. In *Advances in Neural Information Processing Systems*, pages 6533–6542, 2017.

[12] Joël Janin. Docking predictions of protein-protein interactions and their assessment: the CAPRI experiment. In *Identification of Ligand Binding Site and Protein-Protein Interaction Area*, pages 87–104. Springer, 2013.

[13] Rafael A Jordan, EL-Manzalawy Yasser, Drena Dobbs, and Vasant Honavar. Predicting protein-protein interface residues using local surface structural similarity. *BMC bioinformatics*, 13(1):41, 2012.

[14] Florian Krull, Gerrit Korff, Nadia Elghobashi-Meinhardt, and Ernst-Walter Knapp. ProPairs: a data set for protein–protein docking. *Journal of chemical information and modeling*, 55(7):1495–1507, 2015.

[15] Irina Kufareva, Levon Budagyan, Eugene Raush, Maxim Totrov, and Ruben Abagyan. Pier: protein interface recognition for structural proteomics. *Proteins: Structure, Function, and Bioinformatics*, 67(2):400–417, 2007.

[16] Shide Liang, Chi Zhang, Song Liu, and Yaoqi Zhou. Protein binding site prediction using an empirical scoring function. *Nucleic acids research*, 34(13):3698–3707, 2006.

[17] Tie-Yan Liu et al. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval*, 3(3):225–331, 2009.

[18] Debora S Marks, Thomas A Hopf, and Chris Sander. Protein structure prediction from sequence variation. *Nature biotechnology*, 30(11):1072, 2012.

[19] Yoichi Murakami and Kenji Mizuguchi. Applying the Naïve Bayes classifier with kernel density estimation to the prediction of protein–protein interaction sites. *Bioinformatics*, 26(15):1841–1848, 2010.

[20] Ruben Sanchez-Garcia, COS Sorzano, JM Carazo, Joan Segura, and Alfonso Valencia. Bipspi: a method for the prediction of partner-specific protein-protein interfaces. *Bioinformatics (Oxford, England)*, 2018.

[21] Shai Shalev-Shwartz, Yoram Singer, Nathan Srebro, and Andrew Cotter. Pegasos: Primal estimated sub-gradient solver for svm. *Mathematical programming*, 127(1):3–30, 2011.

[22] Benjamin A Shoemaker, Dachuan Zhang, Ratna R Thangudu, Manoj Tyagi, Jessica H Fong, Aron Marchler-Bauer, Stephen H Bryant, Thomas Madej, and Anna R Panchenko. Inferred biomolecular interaction server—a web server to analyze and predict protein interacting partners and binding sites. *Nucleic acids research*, 38(suppl_1):D518–D524, 2009.

[23] Mile Šikić, Sanja Tomić, and Kristian Vlahoviček. Prediction of protein–protein interaction sites in sequences and 3D structures by random forests. *PLoS computational biology*, 5(1):e1000278, 2009.

[24] Raymond C Stevens. The cost and value of three-dimensional protein structure. *Drug Discovery World*, 4(3):35–48, 2003.

[25] Ilya A Vakser. Protein-protein docking: From interaction to interactome. *Biophysical journal*, 107(8):1785–1793, 2014.

[26] Thom Vreven, Iain H Moal, Anna Vangone, Brian G Pierce, Panagiotis L Kastritis, Mieczyslaw Torchala, Raphael Chaleil, Brian Jiménez-García, Paul A Bates, Juan Fernandez-Recio, et al. Updates to the integrated protein–protein interaction benchmarks: docking benchmark version 5 and affinity benchmark version 2. *Journal of molecular biology*, 427(19):3031–3041, 2015.

[27] Li C Xue, Drena Dobbs, Alexandre MJJ Bonvin, and Vasant Honavar. Computational prediction of protein interfaces: A review of data driven methods. *FEBS letters*, 589(23):3516–3526, 2015.

[28] Li C Xue, Drena Dobbs, and Vasant Honavar. HomPPI: a class of sequence homology based protein-protein interface prediction methods. *BMC bioinformatics*, 12(1):244, 2011.

[29] Qiangfeng Cliff Zhang, Lei Deng, Markus Fisher, Jihong Guan, Barry Honig, and Donald Petrey. PredUs: a web server for predicting protein interfaces using structural neighbors. *Nucleic acids research*, 39(suppl_2):W283–W287, 2011.