Predicting protein-protein interaction interfaces using protein coevolution

Esha Atolia*
Department of Chemical and Systems Biology
Stanford University
eatolia@stanford.edu

Abstract

Protein-protein interaction interface detection is imperative for understanding molecular mechanism, drug design, and drug discovery. Previous work has used structure information to predict interfaces. However, here we use a structure agnostic method to identify protein interaction interface using just protein sequence information. We re-purposed a UNet based method called CellUNet used for image segmentation algorithm for protein interface prediction. Our model has a 71% categorical accuracy, which is up to par with methods that use additional information such as structure.

1 Introduction

Proteins are the building blocks of life since they make up the network of machinery that control most of the function of living cells. Therefore, understanding protein-protein interactions is vital for describing, characterizing, and manipulating biological systems. Specifically, it is important to be able to identify inter-proteasmal interaction interfaces, i.e. the residues that are in physical contact when two proteins directly bind. Knowing this interface not only provides general mechanistic understanding, but also provides a better avenue for drug discovery. Some proteins that are implicated in disease, such as the Tau protein in Alzheimer's, can be inhibited by disrupting the protein-protein interface.

In addressing this problem, large research efforts are dedicated to parsing out the sequence and function of proteins. Previous work includes using support vector machines on only sequence information to predict protein-protein interactions, Bayesian networks for genome-wide protein interactions, and a stacked autoencoder to predict protein-protein interactions based on sequence [4, 9, 10]. Despite ever-increasing collections of protein sequences available for analysis, the regions of each protein that are responsible for its specific function remain mysterious in the vast majority of cases. A powerful approach to overcoming both of these problems is protein co-evolution analysis. Coevolution generally refers to the coordinated changes that occur between organisms, molecules, or components of those molecules (such as the amino acids of a protein) that are under natural selection to maintain functional interactions.

As alluded to earlier, with the advent of sequencing and tools such as X-ray crystallography and other protein structural determination methods, there are >50 million sequences in National Center for Biotechnology Information's (NCBI) Reference Sequence (RefSeq) database and >100k 3D protein structures in the Protein Data Bank (PDB) at our disposal. I leverage this large data set here, by utilizing a image segmentation framework, UNet, to identify protein-protein interaction interfaces using both sequence data and co-evolution signatures of proteins as features. The input for the algorithm is a coevolution matrix of dimension n-by-n-by-16 and the output is a matrix of the n-by-n matrix with 1/0 labels for interface vs not interface residues.

*PhD student in KC Huang's lab in Bioengineering.

CS230: Deep Learning, Winter 2018, Stanford University, CA.

2 Related work

A significant amount of work has been done to identify pairs of interacting proteins using statistical and machine learning methods such as CNNs, SVMs, decision trees, random forests, neural networks, and conditional random fields [3, 4, 9, 10]; however, less work has been done to identify specific residues responsible for the protein-protein interaction, i.e. the residues at the interface of the two interacting proteins. Papers that have focused on identifying protein-protein interaction sites have used protein 3D structural information to do so [2, 5], instead of just sequence data. Previous work on predicting protein-protein interaction interfaces has primarily focused on using sequence information as features [7, 12], whereas the work presented here proposes to use co-evolution signatures in addition to sequence data. Ofran et. al even explicitly say "Incorporating evolutionary [...] information may improve our method" as a future direction. Previous papers have incorporated evolutionary information by including the conservation [1], i.e. entropy, of a protein; few have included both sequence information and protein co-evolution information to predict interaction interfaces.

The current state-of-the-art is a new method called Siamese Atomic Surfacelet Network (SASNet), the first end-to-end learning method for protein interface prediction [11]. It performs better than the models that have been recently predicted. However, like much work that has been previously pubslished, all of these methods depend on having structure information of the protein to make interface preditions. What I a doing is novel because I am using just protein sequence information and protein coevolution data that is derived from sequence information to make these interface predictions. This would be incredibly useful since structural data on protein is cumbersome to attain.

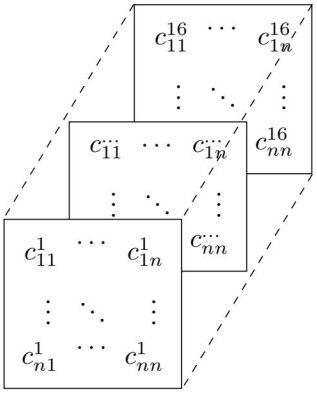

Figure 1: The structure of the Nested Coevolution matrix. This is the input to UNet.

Coevolution generally refers to the coordinated changes that occur between organisms, molecules, or components of those molecules (such as the amino acids of a protein) that are under natural selection to maintain functional interactions. However, coevolution measurements are often obscured by genetic drift, which defines a protein's phylogenetic history and gives rise to artefactual, covariation of positions across the protein sequence. To overcome this problem, KC Huang's lab at Stanford developed a robust methodology, called Nested Coevolution (NC), for explicitly separating the phylogenetic dimension of coevolution signal. This approach generally improves the signal from any existing coevolution algorithm, and NC has been used to demonstrate that coevolution often occurs on multiple phylogenetic timescales within a single protein. Interestingly, NC has been used to identify groups of residues that are evolving together but independently of the rest of the protein, as evidenced by the lack of similarity in the phylogenetic histories in these groups as compared with the protein as a whole. These findings suggest that these coevolving groups embody different, functionally coherent parts of proteins such as active sites, allosteric interactions, protein-protein interaction interfaces, etc. NC can be used to generate a cevolution matrix for our interacting proteins of interest and use it as an additional feature set to just the protein sequence information.

3 Dataset and Features

I construct three separate datasets for training, testing, and validating my method. I obtained co-crystal structures of proteins from the Protein Data Bank to generate a training set, test set and a validation set with a 60/20/20 split totalling about 10,000 proteins.

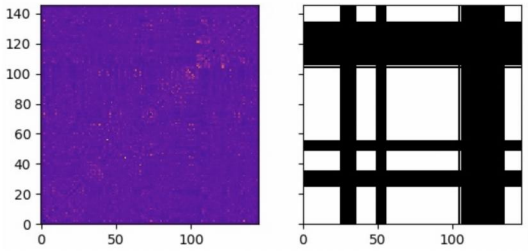

Figure 2: On the left is one layer of the NC matrix, and on the right is the label matrix where black represents the regions of the interface.

The pipeline for dataset generation included first obtaining the pdb ids for all co-crystal structures from the Protein Data Bank (PDB). These ids were used to download the structure file from the pdb. The files were parsed to obtain name of the two proteins, sequence of the two proteins, and the indices residues in each protein that interact with each other ($\|CA_A - CA_B\| \leq 12$Å (Figure 1)),

where CA represents the (x, y, z) position of an α-carbon of one amino acid residue. NCBI BLAST+ was then used to get the top $1,000$ sequences from the refseq database. The outputted FASTA file was then aligned using ClustalΩ. These aligned files are the multiple sequence alignments (MSA) of the protein.

Coevolution matrices were then generated from these MSAs using normalized joint entropy with an average product correction and nested coevolution. The coevolution matrices are of dimension $n \times n \times 16$ (Figure 2), where n is the number is amino acids in the protein and 16 is the number of windows for Nested Coevolution. After the coevolution matrices are generated, the labels for the residues at the interface vs not at the interface are generated using the interaction information parsed previously (Figure 3). These labels are of dimension $n \times n$. Lastly, the Nested Coevolution matrix text files and label text files are converted into tiffs.

4 Methods

Implementing a model for this application is a complex problem for three of reasons: (1) sequence of the input features matters, (2) sequence lengths of the two proteins are variable, and (3) Both local and global structure are important for protein binding

Thus, instead of treating each amino acid's covariation with all the amino acids (i.e. a row of the coevolution matrix) as a observation or doing pairwise comparisons between each interaction pair of amino acids like people have done previously for similar problems, I decided to treat the problem like a segmentation problem. However, instead of having 3 channels like a normal colored image, my "image" has 16 channels. This gets around the issue of matrices of different sizes, of local and global structure being important.

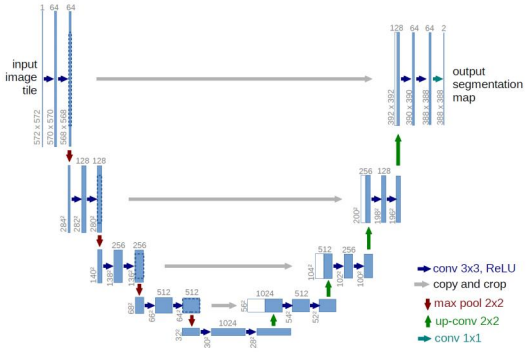

Figure 3: U-net architecture (example for 32x32 pixels in the lowest resolution) [8]. Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations.

The tiffs of the Nested Coevolution matrix and the label matrix were used to train CellUNet [6], a segmentation method based on UNet [8]. The authors describe the architecture as "consist[ing] of a contracting path to capture context and a symmetric expanding path that enables precise localization." Note a few key changes were made for the algorithm to work on this dataset of different dimensions, including modifying the first layer to late the larger input, changing the dataset augmentation methods used, and change the number of labels that are required to be inputted.

5 Results and Discussion

The next step was to train the data on the UNet model. The dataset contains a total of 7909 proteins, where 4745 were used for training and 1582 each were used for testing and validating. There were also a few different hyperparameters that were optimized, such as the number of gradient descent steps taken during training and the number of epochs on the test set. The performance of the model was determined by calculating recall and precision for the validation set.

As stated above, one of the hyperparameters that was tuned was the number number of steps of gradient descent for each epoch of training (Table 1). In optimizing the number of steps of gradient descent for each epoch of training, it was clear that there is a trade-off between the recall and precision as the number of steps is increased. I ended up using 1000 steps for training. Additionally, the number of epochs of training was also optimized (Table 2). After 11 epochs there didnt seem to be a large improvement in recall and precision and there was an increase in validation loss indicating that additional training was leading to overfitting. This I ended up training the final model for 11 epochs.

Other hyperparameters that were tinkered with in a less systematic way included the patch size used by the algorithm and loss function. The final patch size used was 56 and the final loss function was

Steps	Categorical Accuracy	Recall: Interface	Precision: Interface	Recall: Not Interface	Precision: Not Interface	Validation Loss
100	0.51	0.62	0.46	0.42	0.55	0.78
1000	0.76	0.64	0.76	0.81	0.70	0.46
4000	0.59	0.34	0.73	0.84	0.55	0.58

Table 1: Optimizing the number of steps of gradient descent for each epoch of training.

Epochs	Categorical Accuracy	Recall: Interface	Precision: Interface	Recall: Not Interface	Precision: Not Interface	Validation Loss
1	0.58	0.32	0.57	0.74	0.51	0.65
2	0.61	0.54	0.57	0.58	0.56	0.61
3	0.64	0.53	0.61	0.66	0.58	0.59
4	0.65	0.44	0.68	0.78	0.57	0.57
5	0.60	0.35	0.74	0.85	0.56	0.59
6	0.69	0.54	0.69	0.75	0.61	0.54
7	0.70	0.66	0.64	0.64	0.65	0.54
8	0.71	0.66	0.66	0.65	0.66	0.53
9	0.70	0.70	0.63	0.59	0.67	0.55
10	0.70	0.64	0.66	0.67	0.65	0.53
11	**0.71**	**0.68**	**0.67**	**0.66**	**0.67**	**0.52**
12	0.71	0.64	0.68	0.69	0.66	0.55
13	0.71	0.64	0.69	0.71	0.66	0.53
14	0.73	0.67	0.69	0.69	0.68	0.53
15	0.72	0.68	0.68	0.68	0.68	0.56
16	0.73	0.68	0.69	0.69	0.68	0.58
17	0.73	0.71	0.68	0.67	0.70	0.56
18	0.73	0.71	0.68	0.67	0.70	0.56
19	0.73	0.62	0.73	0.76	0.66	0.56
20	0.74	0.67	0.70	0.72	0.68	0.56

Table 2: Optimizing the number epochs for training.

the cross entropy loss. Additionally, after testing data augmentation methods, flipping, rotation, and adding noise were used to augment the training data set.

The model overall performs pretty well for a majority of the proteins (Figure 4-6), however there are some that it fails for (Figure 7). There wasnt a consistent pattern of failure that was evident. Below I show some examples of proteins interaction interface classifications.

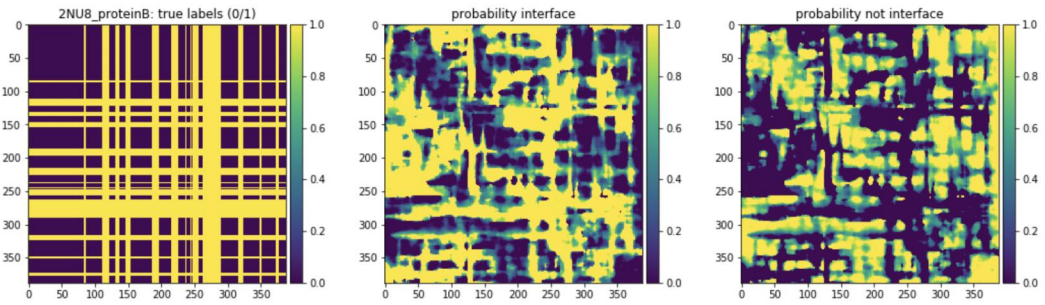

Figure 4: Example of the interface of Succinyl-CoA synthetase beta chain binding with Succinyl-CoA ligase [ADP-forming] subunit alpha (PDB: 2NU8).

6 Conclusion and Future Work

The UNet based CellUNet image segmentation algorithm re-purposed for protein interface prediction works fairly well. There are a few different avenues that can be pursued to improve the model. First it will be important to try to understand why the proteins that do not function well are not functioning

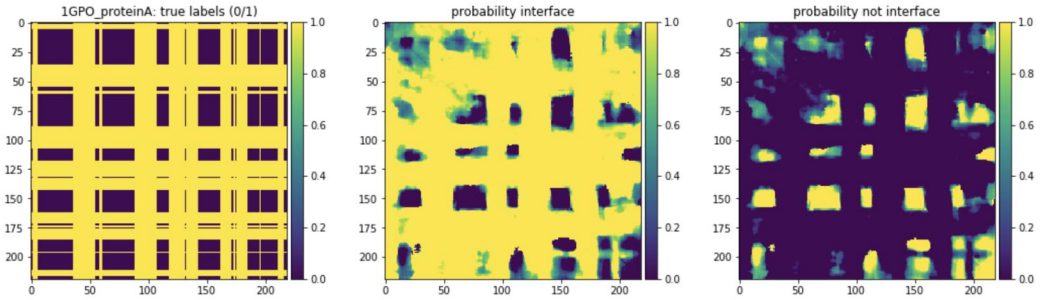

Figure 5: Example of the interface of antibody M41 dimer (PDB: 1GPO).

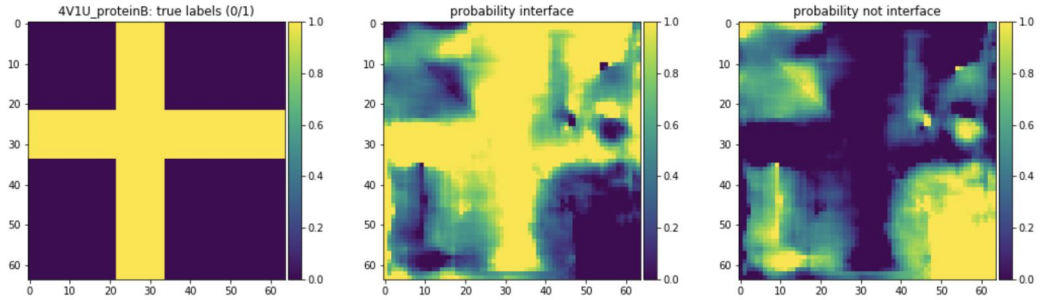

Figure 6: Example of the interface of heterocyclase in complex with substrate and cofactor (PDB: 4v1U).

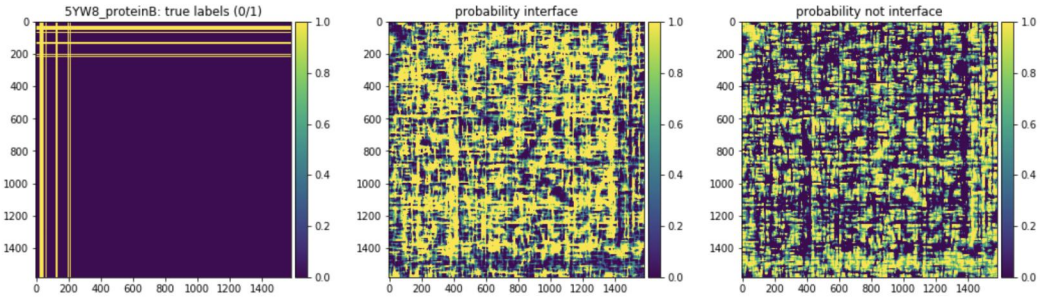

Figure 7: Example of the interface of ATP-binding cassette sub-family C member 8 isoform X2 binding with ATP-sensitive inward rectifier potassium channel 11 (PDB: 5YW8).

well. This can be done by looking at the class activation maps for the a protein that works versus those that dont.

Additionally, there is an added layer of complexity that had been added to the model that is not necessary for the problem at hand. The final label matrix is of size n-by-n but we really just need a final vector of size n to classify all the n amino acids in a protein sequence at being at the interface of not. To be able to do this, we can take the encoding part of UNet and instead of building the image back up add some fully connected layers at the end to get a final output of size n. Furthermore, the current literature has been lacking it trying to identify what additional features lead to a residue being at the interface. There is novel biology to be discovered by looking at the class activation maps of what regions of the original matrix lead to the classification of each residue as at or not at the interface.

7 Github

Code for the milestone: github.com/eatolia/CS230-Final-Proposal. This is a private repository since I do not want public access to this code. I have given my TA (Cristian Bartolomé Arámburu) collaborator access to the git repo. Here is the link invite: github.com/eatolia/CS230-Final-Proposal/invitations.

References

[1] J. R. Bradford and D. R. Westhead. Improved prediction of protein–protein binding sites using a support vector machines approach. *Bioinformatics*, 21(8):1487–1494, 2005.

[2] P. Fariselli, F. Pazos, A. Valencia, and R. Casadio. Prediction of protein–protein interaction sites in heterocomplexes with neural networks. *European Journal of Biochemistry*, 269(5):1356–1361, 2002.

[3] S. Hashemifar, B. Neyshabur, A. A. Khan, and J. Xu. Predicting protein–protein interactions through sequence-based deep learning. *Bioinformatics*, 34(17):i802–i810, 2018.

[4] R. Jansen, H. Yu, D. Greenbaum, Y. Kluger, N. J. Krogan, S. Chung, A. Emili, M. Snyder, J. F. Greenblatt, and M. Gerstein. A bayesian networks approach for predicting protein-protein interactions from genomic data. *Science*, 302(5644):449–453, 2003.

[5] A. Koike and T. Takagi. Prediction of protein–protein interaction sites using support vector machines. *Protein Engineering, Design and Selection*, 17(2):165–173, 2004.

[6] T. Kudo. Cellunet. 2018.

[7] Y. Ofran and B. Rost. Predicted protein–protein interaction sites from local sequence information. *FEBS Letters*, 544(1):236 – 239, 2003.

[8] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015.

[9] J. Shen, J. Zhang, X. Luo, W. Zhu, K. Yu, K. Chen, Y. Li, and H. Jiang. Predicting protein–protein interactions based only on sequences information. *Proceedings of the National Academy of Sciences*, 104(11):4337–4341, 2007.

[10] T. Sun, B. Zhou, L. Lai, and J. Pei. Sequence-based prediction of protein protein interaction using a deep-learning algorithm. *BMC Bioinformatics*, 18(1):277, May 2017.

[11] R. J. L. Townshend, R. Bedi, and R. O. Dror. Transferrable end-to-end learning for protein interface prediction. *arXiv*, 2018.

[12] M. Šikić, S. Tomić, and K. Vlahoviček. Prediction of protein–protein interaction sites in sequences and 3d structures by random forests. *PLOS Computational Biology*, 5(1):1–9, 01 2009.