Computational methods for protein function prediction in protein-protein interaction networks

Micah Reich, Sarah Fisher, Yoona Choi, Leon Xie [*]

Carnegie Mellon University School of Computer Science

May 6, 2022

Abstract

Proteins are an integral component of biological processes, serving as the tools with which cells replicate, transcribe genomes, regulate cell function, and catalyze the reactions necessary for a cell to survive. Given that the range of possible functions a protein can serve is highly diverse, researchers seek to narrow the range of functions that an non-annotated protein may serve. The existence of experimentally-found protein-protein interaction networks opens the door to graph-based approaches for function annotation of novel proteins. In this paper, we describe three such approaches: a majority neighbor approach, a network flow approach, and a neighbor sequence alignment approach. We find that Functional Flow scores best on measures of accuracy and specificity across different radii; we also note that Sequence Alignment performs similarly at small radii, while Majority Approach performs well at smaller radii though suffers in measures of specificity as r increases.

1 Introduction

Protein function prediction is an important problem in biology. With the advent of next-generation sequencing techniques, the amount of biological data available is at an all-time high. Moreover, computational techniques on genomic data have allowed researchers to identify new proteins. Traditional experimental techniques to research protein function are insufficient to keep up with this growth. In light of this, computational methods offer a scalable solution. [2]

Being able to predict protein functions allows for better understanding of key biological mechanisms; for example, understanding the functions of proteins involved in disease mechanisms can propel drug development, and identifying homologous proteins in different species can aid in evolutionary tracing. [2]

In the current literature, a plethora of methods for function annotation exist. These include analysis of sequence similarity, gene expression, structure, phylogeny, and more, employing graph analysis, deep learning, and traditional biological experimentation. We focus on graph analyses of protein-protein interaction networks for the purpose of function annotation.

1.1 Protein-protein interaction networks

In any given organism, proteins constantly interact with one another, both directly and indirectly, to maintain biological function. Proteins can serve as transcription factors, bind together to form complexes, or activate and deactivate each other. The set of protein interactions are known as the interactome and can be modeled by a protein-protein interaction network (PPIN). A PPIN is an undirected and optionally weighted graph, where a protein p is represented by a node P. For any pair of proteins $\{a, b\}$, if there exists some interaction between them, then the edge from A to B exists. In a weighted PPIN, edge weights correspond to confidence in the existence of the interaction.

[*]mreich@cmu.edu skfisher@andrew.cmu.edu yoonac@andrew.cmu.edu leonx@andrew.cmu.edu

The existence of an interaction can be determined experimentally, through direct methods like X-ray crystallography and pull-down experiments. These examine specific biophysical and chemical structure, and use this information to determine protein-protein interactions. The yeast two-hybrid protocol utilizes hybridization of two proteins of interest into transcription components to activate some signal gene when those two proteins interact. Coexpression data can also be used. [4]

The high cost of biological experiments leads naturally to the use of computational methods for forming PPINs. Protein-protein interactions can be inferred from existing interactions. Additionally, potential interactions based on other forms of data can be made.

1.2 Computational problem

Analysis of a PPIN can provide results about proteins and their functions. In particular, we can formalize the protein function prediction problem into a computational one, and utilize solutions from graph theory and computer science to develop a solution.

Protein Annotation Problem (Multilabel):

> **Input:** A weighted PPIN G, an assignment of function annotations to each protein in G, a set S of proteins of unknown function and their locations in G.
>
> **Output:** A multiple labelling of each protein in S such that the accuracy of the labelling is maximized.

Note that in this formulation of the computational problem, we assume that we can verify the correctness of our predictions. This formulation is analogous to the training phase of a neural network where the performance of a model may be verified before deploying a final algorithm to truly novel data.

2 Methods

The Python implementations of Majority Approach, Functional Flow, and Alignment Approach from which we drew our results can be found on our GitHub repository.

All of our implementations rely on the assumption that proteins which are close in proximity in a PPIN share similar function. This assumption is well-founded as shown by Figure 1.

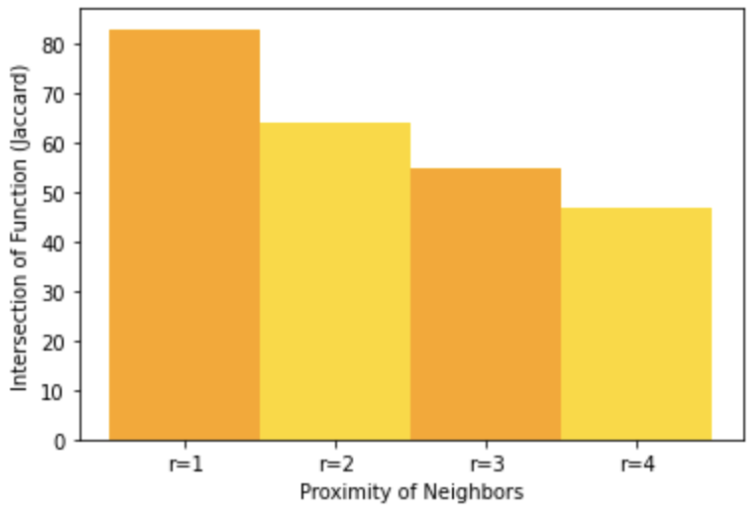


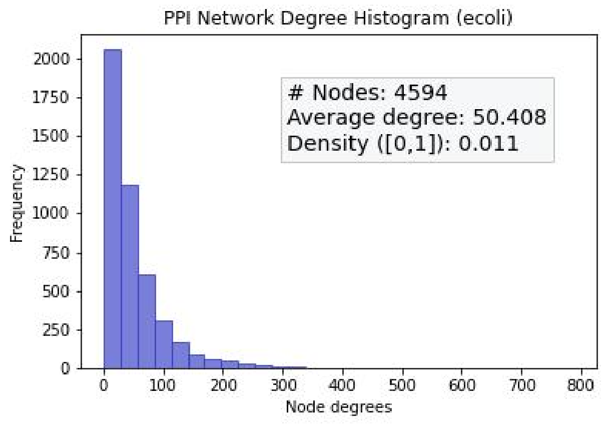
Figure 1: Intersection of protein function (Jaccard index) by radius in a STRING PPIN

2.1 Data sources

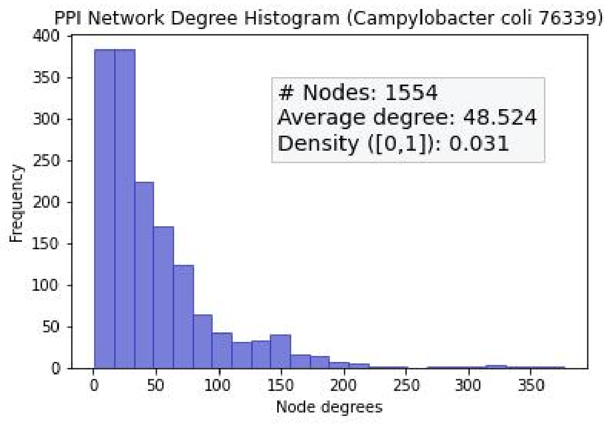
We utilized the STRING database [3], a collection of protein-protein interaction data available online for research use. STRING contains protein interaction data for over 14,000 genomes and over 67 million proteins. For a given organism, STRING assigns each pair of proteins that have an interaction a score between 0 and 1000, corresponding with the relative confidence in the existence of that interaction. STRING utilizes a variety of sources to establish this final score. These sources come from three different categories: genomic analysis, experimental data, and prior knowledge. Each source of knowledge is referred to as a channel.

Genomic context includes the neighborhood channel, the fusion channel, and the co-occurrence channels. For the neighborhood channel, genes that are closer in proximity are given higher scores, since their locations on genomes may be reflective of their shared function. For the fusion channel, STRING identifies likely fusion events. Corresponding genes in other organisms are then given association scores. Finally, genes that occur in similar evolution patterns are given high co-occurrence scores.

Next is the experimental data. First is the co-expression channel. Proteins and RNA that are expressed at similar levels under differing circumstances are likely to have similar function. Second, the experimental channel includes data from biophysical, biochemical, and genetic experiments. Some of these experiments directly investigated physical interactions between two proteins, providing strong evidence that they are involved with each other.



(a) *E. Coli* network statistics from STRING



(b) *Campylobacter coli* network statistics from STRING

Figure 2: Network degree histogram for organisms contained in the STRING PPIN database

Finally, there are the knowledge channels and the text-mining channels. The knowledge channel parses a number of curated pathway databases. The text-mining channel uses statistical methods to analyze large amounts of text pulled from existing literature. STRING looked for mentions of proteins in the same sentence, paragraph, and paper, and assigned association scores accordingly. Interactions from each of these channels are given a quality score according to the data from each channel, and then calibrated accordingly using the KEGG pathway map.

For our purposes, we used protein link data, function annotation data, and protein sequences. Protein link files contained three columns: protein A, protein B, and the score assigned to the interaction between them. We used these data to construct graphs representing PPINs. Protein cluster files assigned known proteins to clusters, represented by an ID number, where each cluster represents protein function. Note that proteins can have many functions, and thus be assigned to many clusters, and the clusters are hierarchical in nature: some are more specific than others. We also had access to sequence data, which mapped each protein to its amino-acid sequence. With these data we constructed PPINs with which we were able to implemented our algorithms.

2.2 Majority Neighbor

The first method we employ is the Majority Neighbor approach. Given a protein p and its radius r neighborhood (the radius r neighborhood are all $v \in V : \text{dist}(p, v) \leq r$) N, we assign to p the k most frequent functions that occur in the annotations of N. Note, however, that since some protein functions occur extremely frequently in the network G, merely taking the k most frequently occurring functions in N will yield results that are dominated by over-represented functions and our resulting assignment will lack specificity [12].

To combat this, we employ a metric [6] to decide which functions of N to take. We define new terms below as:

$$\begin{aligned} e(p, n, a) &= \text{expected number of proteins in a collection of } n \text{ with function } a \\ &= \frac{\text{number of proteins with function } a \text{ in } G}{\text{total number of proteins in } G} \cdot n \\ f(p, n, a) &= \text{actual number of proteins in a collection of } n \text{ with function } a \end{aligned}$$

Using these terms, we can calculate how "surprisingly frequent" a protein function a is by analyzing the metric:

$$M = \frac{(f(p, |N|, a) - e(p, |N|, a))^2}{e(p, |N|, a)} \qquad \text{(Probabilistic scoring metric)}$$

Note that we should expect M to be high for highly surprising functions since $f(p, r, a) > e(p, r, a)$. Thus we instead take the k highest-scoring protein functions according to M for a protein p. In the future, we hope to modify the metric M to consider edge weights in the PPIN so that the annotations of proteins with highly-rated interaction scores contribute more to the set of functions considered.

2.3 Functional Network Flow

The next method we employ is functional network flow. Given a protein p, we simulate the flow of information across the PPIN as modeled by the flow of liquids between reservoirs through pipes. In this case, each protein of known function is filled with an infinite amount of water for each of its functions at iteration $t = 0$. Throughout the iterations, we simulate flow of water out of nodes of known function and into nodes of unknown function, with stronger connections–as indicated by higher edge weights–allowing for more throughput into unknown reservoirs.

We now explain the mathematical models that regulate the pipe capacity and outline the update rules for a given reservoir for each iteration [5]. We denote the amount of water in a reservoir for function a at time t for a given protein p as $R_t^a(p)$. We similarly denote the pipe capacity of function a at time t between proteins p_1 and p_2 as $g_t^a(p_1, p_2)$. Initially, the reservoirs are initialized as:

$$R_0^a(p) = \begin{cases} 0 & \text{if } p \text{ has unknown function} \\ \infty & \text{if } p \text{ has known function } a \end{cases} \qquad \text{(Reservoir initialization)}$$

During each iteration, we must update the reservoir for each function of each unknown protein. At a high level, we expect the amount of water in the reservoir to be the amount present in the previous iteration minus the amount of water that leaves in the current iteration plus the amount of water that enters in the current iteration. Formally, we define this as:

$$R_t^a(p) = R_{t-1}^a(p) + \sum_{v:\{v,p\} \in E} g_t^a(v, p) - \sum_{v:\{v,p\} \in E} g_t^a(p, v) \qquad \text{(Reservoir update rule)}$$

To model the pipe capacity, or the amount of flow of water between two nodes, we take advantage of the given PPIN edge weights which represent the strength of confidence in the interaction between two proteins. Initially, the pipe capacity between all edges is 0. For an arbitrary iteration, we define the pipe capacity as:

$$g_t^a(p_1,p_2) = \begin{cases} 0 & \text{if } R_{t-1}^a(p_1) \le R_{t-1}^a(p_2) \\ w_{p_1,p_2} & \text{otherwise} \end{cases} \qquad \text{(Directional pipe capacity)}$$

Note that the pipe capacity between p_1 and p_2 is not necessarily equal to the capacity for p_2 to p_1; that is, the direction of flow matters. Also note that we define the capacity as 0 for proteins in which their previous iterations' reservoirs are equal or where the reservoir of p_1 was smaller than that of p_2. This is because we wish to only model the "downhill" flow of water as if our pipes were one-way. This is to ensure that unknown proteins receive enough signal from surrounding nodes by the end of the simulation and so that water is able to sufficiently spread throughout the network.

Finally, to determine the function annotations of our unknown proteins, we calculate the total flow of water that has entered an unknown proteins' reservoir for each function at the end of all iterations. The amount of water that has entered a protein's reservoir for function a is defined as:

$$f_a(p) = \sum_{i=1}^{t} \sum_{v:\{v,p\}\in E} g_i^a(v,p) \qquad \text{(Total flow)}$$

2.4 Peptide Sequence Alignment

The final method for function prediction we employ is that of neighborhood sequence alignment. Given that we have access to peptide sequences for proteins in the PPIN, we may use this information to determine sequence similarity and infer function. Given a protein p and its radius r neighborhood N, we assign p the function annotations of the protein p' such that $\text{AlignmentScore}(p, p')$ is maximized.

Performing sequence alignment is a relatively cheap way to assign functions to proteins based on structural similarities, as we may make a somewhat naive assumption that proteins with similar amino acid sequences fold into similar shapes. Biologically, a protein's function is in-large determined by its shape and which substrates can bind to the active site, so we expect proteins of similar function to have similar shape. Due to the fact that the protein-folding problem is difficult to solve and computationally expensive, using sequence alignment as a proxy for structural comparison serves as a cheaper, faster alternative. Additionally, since we have reason to suspect that proteins which are close in a PPIN share similar function, we can significantly restrict our search space by only aligning against those proteins which are close to p in the PPIN (within distance r).

We perform pairwise affine global alignment of p's peptide string against those of all proteins in N by solving the longest path in a directed acyclic graph (DAG) dynamic programming problem. To assign edge weights to the DAG, we use the BLOSUM62 scoring matrix for match and mismatch penalties, a commonly used matrix for amino acid sequence alignments based on experimental evidence, as well as a -2 gap creation penalty and -1 gap continuation penalty. The recurrence relations we use to guide the longest path in a DAG solution are defined as [11]:

$$\textbf{Case 1}: \mathrm{s}(i,j) = \max\{\mathrm{s}(i-1,j) - \sigma, \\ \mathrm{s}(i,j-1) - \sigma, \\ \mathrm{s}(i-1,j-1) + \mu_{p_i,p'_j}\} \qquad \text{(Gap opening recurrence)}$$

$$\textbf{Case 2}: \mathrm{s}(i,j) = \max\{\mathrm{s}(i-1,j) - \epsilon, \\ \mathrm{s}(i,j-1) - \epsilon, \\ \mathrm{s}(i-1,j-1) + \mu_{p_i,p'_j}\} \qquad \text{(Gap extension recurrence)}$$

Where σ is the gap opening penalty, ϵ is the gap extension penalty, and μ_{p_i,p'_j} is the score of the amino acids at positions i and j of p and p', respectively, based on the BLOSUM62 matrix. After performing pairwise sequence alignment on the sequence p and all proteins in N, we assign for all $p' \in N$ $\text{score}(p') = s(|p|, |p'|)$, the score of the alignment. Lastly, we take the maximum over all scores and assign to p the annotation list of the p' which scored the best. In the future, we may multiply the final alignment score by the edge weight between two proteins to weight the final assignment by interaction confidence.

3 Results

We preface our results section with a note on the cluster distribution imbalance present within many of the STRING PPINs. Below find visualization of the distribution of cluster sizes and the relative frequency of clusters for a protein in a PPIN.

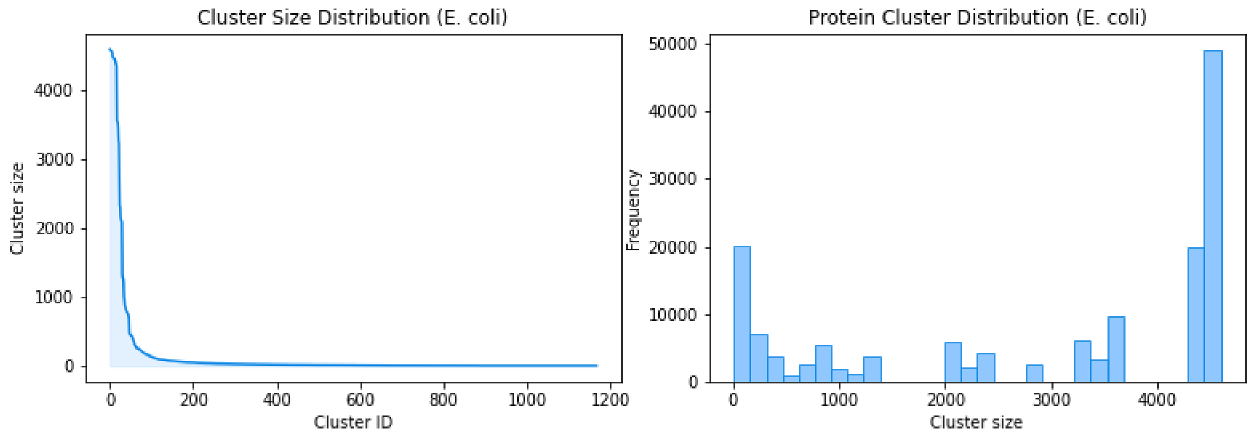


Figure 3: (left) Cluster IDs and the number of proteins in each cluster in the E. coli proteome; (right) histogram of cluster sizes of proteins in the E. coli proteome.

We note that there exist far more clusters of small size than clusters of large size, generally. In the E. Coli proteome, for instance, a large majority of clusters fall within the 5 to 50 protein bucket (out of nearly 5000 proteins in the network), while there exist a limited number of extremely general, large clusters that contain more than 1000 proteins. We also notice that the distribution is not uniform; that is, there exist clusters with very few proteins and clusters with many proteins with few clusters falling in between the two extremes. When analyzing global specificity results, it is important to note that the percentiles of cluster size reflect the cluster size distribution imbalance; in the E. coli PPIN, the 50th percentile cluster has size 12 and is (comparatively) extremely narrow in scope. Thus the difficulty of predicting functions in high percentile clusters is amplified by this cluster size imbalance, as individually, each small cluster is sparse, so there is little signal to work with. On the other hand, large clusters are smaller in numerosity yet highly dense.

3.1 Accuracy and specificity

We interpret the results gathered from our implementations of the majority neighbor approach, functional network flow, and peptide sequence alignment through two lenses of analysis: specificity and accuracy. Traditional measures of accuracy used in multi-label classification problems are examined across the three approaches at different measures of radii. The notion of specificity that we define is different from its typical definition in the field of data analysis, as it is redefined to serve our unique purpose, further detailed below.

3.1.1 Accuracy

For a multi-label classification problem such as protein function annotation, there exist three commonly used metrics of accuracy: precision, recall, and F1 Score. These metrics are often used in machine learning literature to quantify the accuracy of neural network models, and we adopt these metrics to assess the quality of our function annotations as well. We describe each metric in detail below.

- **Precision**: After making an annotation prediction, we are left with a number of true positive (TP), false positive (FP), true negative (TN), and false negative (FN) predictions. Given a set of predictions P, ground truth labels L, and a prediction x, we say x is a true positive if $x \in P \wedge x \in L$, x is a false positive if $x \in P \wedge x \notin L$, x is a false negative if $x \notin P \wedge x \in L$, and x is a true negative if $x \notin P \wedge x \notin L$. We define precision as:

$$\text{Precision} = \frac{\#\ \text{TP}}{\#\ \text{TP} + \#\ \text{FP}} \qquad \text{(Precision formulation)}$$

  Thus precision serves as a "traditional" metric for accuracy, measuring the ratio of correct predictions to total predictions.

- **Recall**: Continuing with the previously defined notions of true/false positive/negative predictions, we define recall as:

$$\text{Recall} = \frac{\#\ \text{TP}}{\#\ \text{TP} + \#\ \text{FN}} \qquad \text{(Recall formulation)}$$

  Since the number of true positives plus the number of false negatives equals the number of true annotations in the protein's annotation list, recall measures the ratio of true predictions to the total number of actual annotations.

- **F1 Score**: To combine the precision and recall scores into one metric, the F1 score is used, formulated as:

$$\text{F1 Score} = 2 \cdot \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \qquad \text{(F1 formulation)}$$

  The F1 score is commonly used when there is class distribution imbalance and there are a large number of true negatives for each sample in our dataset. In the case of function annotation, there exist on the order of thousands of functions for a given proteome while each protein is only assigned to on the order of tens of classes, so F1 score is a helpful metric in this use case since it combines both recall and precision while avoiding domination by true negatives. Note that the F1 score is functionally similar to the Jaccard index.

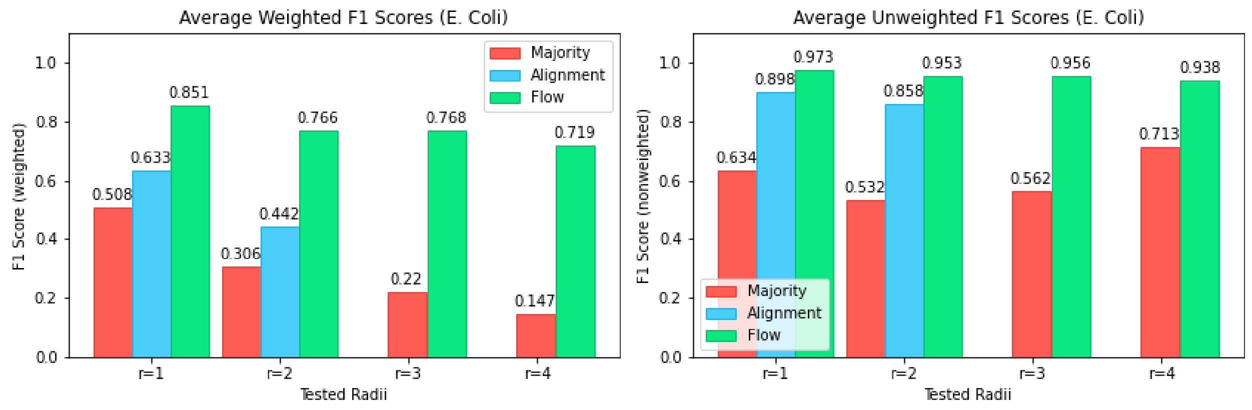


Figure 4: Weighted and nonweighted F1 scores for Majority Approach, Functional Flow, and sequence alignment for radii 1 to 4 on the E. Coli proteome

We note a few important details. First, due to limited computational resources, we were unable to run the peptide sequence alignment approach past a radius of 2. Second, we consider a weighted (by cluster size) F1 score. That is, instead of counting every predicted cluster and actual cluster as equal weight, we consider a predicted and actual annotation's cluster size (the reciprocal of cluster size), so that accurate predictions of small clusters contribute more to the score than do accurate predictions of large clusters. This rewards the algorithms for accurately predicting functions with small cluster size, as we take a small cluster size to imply a more specific function annotation.

3.1.2 Specificity

The STRING database provides a list of cluster assignments for each protein in an organism. Each of these clusters has an associated function that its proteins perform, as well as a cluster size, the number of proteins in the given cluster. The way in which STRING assigns cluster functions is hierarchical in the sense that nearly all proteins belong to a general cluster, fewer belong to each of the more specialized but still common clusters, and only a few belong to each of the most specific clusters. For this reason, our notion of specificity quantifies how specific correctly predicted functions are, categorizing those that belong to the smallest clusters as the most specific.

The specificity of protein function assignment can be viewed on two different scales: globally or locally. Globally, this metric refers to the specificity of a protein's cluster annotation in comparison to the cluster sizes of all clusters present in the organism. Locally, this metric examines the same property of a given cluster but, instead, compares it to only the cluster sizes present in the protein's list of true cluster annotations.

When evaluating the specificity of a function prediction algorithm, we examine only the algorithm's true positive predictions. Each cluster, based on its size, is placed into a percentile. The lowest percentiles correspond with the largest, and thus least specific, clusters, while the highest percentiles correspond with the most specific clusters smallest in size. For a given percentile, specificity is calculated by taking the ratio of true positive predictions in this percentile to the number of functions in the same percentile in the ground truth annotation.

The following histograms that visualize specificity for each of our function prediction approaches have common features which are necessary to note for correct interpretation. Along the x-axes, a traditional percentile, from 0 to 100, shows the size of the clusters that are present in the set of tested proteins, either on a global or a local scale. The y-axes show, proportional to the total number of clusters present in each percentile, the percent of true positive predictions that fall within its corresponding percentile bucket. That is, we visualize the percent of correct predictions for each cluster size percentile. For example, in Figure 4, we say that Majority Approach correctly predicted 40% of functions in the 4th percentile globally. For sake of visualization, we bucket percentiles into sets of two, resulting in 50 data points.

3.2 Majority Approach

For a full set of global and local specificity plots at radii $r = 1, 2, 3, 4$ for majority appraoch, please see the Appendix.

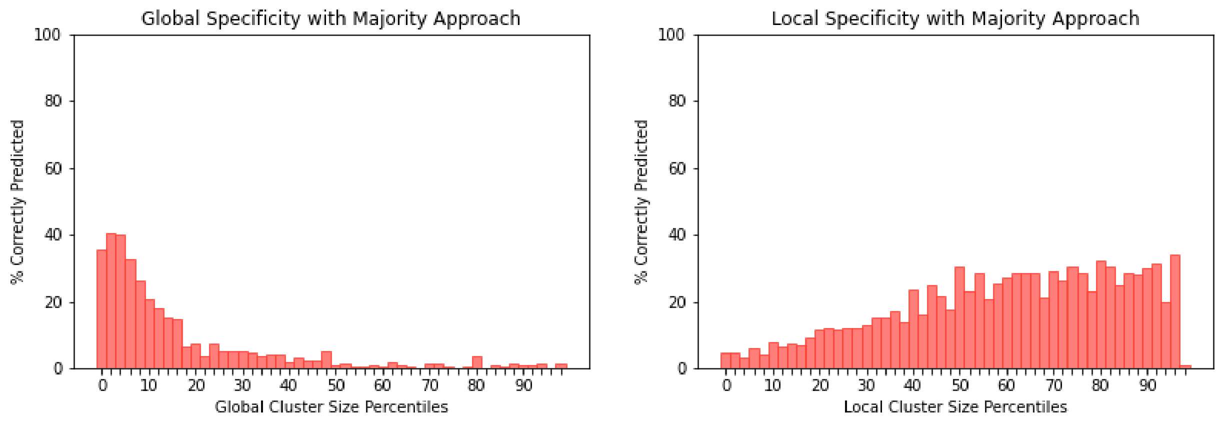


Figure 5: Global and local specificity of Majority Approach function annotation on the E. Coli proteome at $r = 2$.

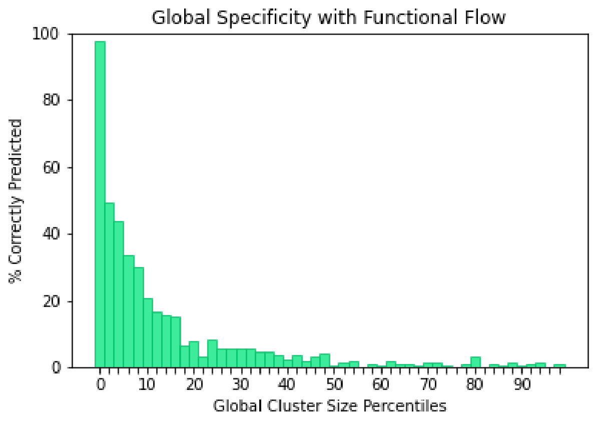
The simplest behavior of Majority Approach exists in the raw accuracy scores, displayed in Figure 4. These show that the average weighted F1 Score decreases as its tested radius increases. However, after an initial drop in unweighted accuracy at a test radius of 2, the unweighted F1 Score increases across radii 3 and 4. This behavior is predictable, as Majority Approach prioritizes the functions that appear most commonly in its neighborhood. As the size of a protein's neighborhood increases, its exposure to the more common clusters that nearly every protein has in its annotation list also increases, making it far more likely that there will be a true positive assignment of very general functions that are not weighted highly.

This conclusion is supported by the graphs of Majority Approach's predictions in both global and local specificity, a sample of which is shown in Figure 5. Global specificity at a testing radius of 1 has representation at all points, skewed toward less specific clusters. As the radius of the neighborhood increases, global specificity at lower percentiles, namely the 0th, greatly increase, as representation above greater percentiles (at radius 2: 50, at radius 3: 30, and at radius 4: 15) become scarce.

In terms of local specificity, when tested at a radius of 1, Majority Approach performs with 100 percent correct predictions on clusters in the 0th-3rd, and 52nd-94th percentiles. At a testing radius of 2, the local specificity is skewed toward the right, achieving more true positive predictions at clusters with lower sizes than at clusters with higher sizes. At a radius of 3, the correctness of Majority Approach is relatively level at all percentiles, performing the best between the 50th and 66th percentiles. At a testing radius of 4, the local specificity is skewed toward the left, performing best on clusters of lower percentiles. This change in behavior portrays that Majority Approach excels at correctly predicting highly specific functions at low radii but shifts to better predicting the most general functions as the size of the neighborhood increases. Again, this is a reasonable observation since as the radius increases, an unknown protein will be exposed to higher amounts of highly-specific clusters, so the likelihood that we predict the true specific cluster of which the protein is a member decreases.

3.3 Functional Flow

For a full set of global and local specificity plots at radii $r = 1, 2, 3, 4$ for Functional Flow, please see the Appendix.



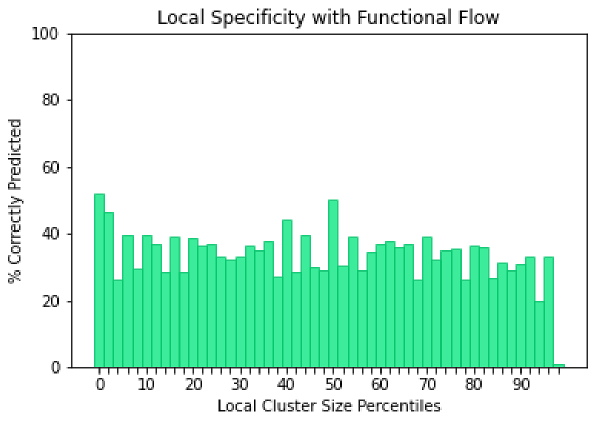


Figure 6: Global and local specificity of Functional Flow function annotation on the E. Coli proteome at $r = 2$.

The raw accuracy F1 Score of the Functional Flow algorithm, both when unweighted and weighted by cluster size, predictably decreases as the testing radius increases. However, as seen in Figure 4, this decrease is only marginally significant, spanning a mere 4 percent in unweighted F1 Score accuracy and remaining above 90 percent.

When examining Functional Flow's histograms of global specificity, there is representation at almost all percentiles for function predictions of proteins at each radius. Although in a neighborhood with a radius of 1, the best performance in global specificity is skewed toward lower percentiles, true positive predictions remain upwards of 15 percent through the 40^{th} percentile, which, as expressed in Figure 3, would be a highly specific cluster. At radii larger than 1, global specificity exponentially decreases, starting at nearly 100 percent at the 0th percentile, in a similar manner for each radius.

In analysis of local specificity, Functional Flow correctly predicts nearly 100 percent of the clusters from the 0^{th} to the 94^{th} percentile at a radius of 1, aligning accurately with its 97.3 percent unweighted F1 Score. After this initial nearly perfect performance, tests at radii 2, 3, and 4 are each visually similar, having the same shape across their corresponding local histograms. As seen in Figure 6, this common shape is represented by a near 30 percent average score after the higher 0^{th} percentile, whose larger value is due to the heavy amount of flow received from the most common functions in its neighborhood.

3.4 Peptide Sequence Alignment

For a full set of global and local specificity plots at radii $r = 1, 2$ for peptide sequence alignment, please see the Appendix.

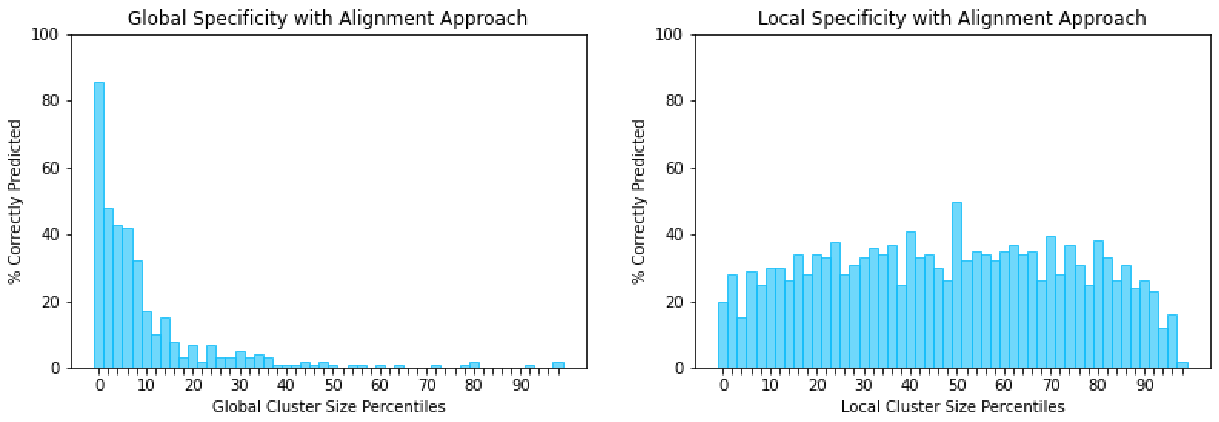


Figure 7: Global and local specificity of Majority Approach function annotation on the E. Coli proteome at $r = 2$.

Even with results of Alignment Approach being limited to testing radii of only 1 and 2, we have observed likely long-term behavior for the function prediction method. As the radius at which proteins are aligned increases, raw accuracy F1 Score, both weighted and unweighted, decreases, as seen in Figure 4. This directly follows from the assumption that proteins with more similar functions exist in closer proximity to a given protein, as Alignment Approach forces its comparisons to be influenced only by neighbors at the radius being tested, without additional influence from the lesser radii. Although Alignment Approach's unweighted accuracy is quite high, remaining above 85 percent at radius 2 and near the F1 Scores of Functional Flow, its weighted accuracy is significantly lower, dropping 25 percent at radius 1 and 40 percent at radius 2. This decreasing accuracy and stark difference between weighted and unweighted F1 Scores are mirrored by the largely left-skewed global specificity and low correctness in 92nd-100th percentiles of local specificity predictions.

Behaving similarly to Functional Flow at radii of 1 an 2, this approach has a global specificity distribution that is skewed to the left, predicting functions with cluster sizes before the 40th percentile with the highest percent of true positives. Between radius 1 and radius 2, this distribution becomes less specific, with a steeper visual decline, as seen in Figure 7. Although we have no results for neighborhoods beyond a radius of 2, we can predict that the method would continue to produce results with lower specificity as r increases.

Apart from a few incorrectly predicted functions in the 94th percentile of cluster size, Alignment Approach achieves correct labeling for 100 percent of the 0th through 96th percentiles in its graph of local specificity at radius 1. This again verifies that peptide sequence correlates strongly with function of proteins, as this method is capable of predicting functions that are very specific on the local scale of a protein's cluster annotations.

4 Discussion

4.1 Comparison of methods

4.1.1 Inter-method comparison

By comparing the results from our implementations of the above methods, Majority Approach, Functional Flow, and Alignment Approach, there are many conclusions that can be drawn. It is clear from Figure 4 that the raw accuracy F1 Score of Functional Flow is far superior to the other two methods in both weighted and unweighted calculations. It also follows that Functional Flow is the best method for predicting all types of functions, regardless of specificity, as its weighted accuracy remains above 70 percent through a neighborhood radius of 4. From this observation, it can be concluded that the mechanics unique to Functional Flow, like the implementation of a weighted influence, based upon the confidence score of the edges between a protein and its neighbors, are optimal for protein function prediction.

The unweighted raw accuracy scores also present evidence that Majority Approach has ideal attributes as the radii of testing neighborhoods increase. As the only method of the three tested implementations to increase in F1 Score as radius increases, it is likely that Majority Approach could continue to increase beyond a neighborhood radius of 4, as well. It is worth noting, however, that this increase in unweighted accuracy comes at a cost of a decreasing weighted accuracy. This is an example of the trade-offs that function predictions methods may have, as Majority Approach's implementation of the probabilistic scoring metric amounts to higher unweighted accuracy and lower specificity as the neighborhood radius increases.

In general, it can be concluded that the behavior across all approaches is largely similar in terms of global specificity. Each approach has some level of distribution across all percentiles of cluster size at a radius of 1 and, as the radius increases, higher specificity skews toward lower percentiles. Even locally, all approaches have a comparably high specificity for a radius of 1 that, apart from Majority Approach, decreases with an increasing neighborhood radius. These observations show deeper truths about the nature of protein function hierarchy and relation to neighboring proteins. Proteins directly neighboring a given protein are more similar than proteins at a farther distance, while clusters that have few proteins become hard to detect when influenced by a larger radius with more neighbors. In general, the combined behavior of the F1 Score accuracy, local specificity, and global specificity of each of the the presented data sets show that a higher raw accuracy score has strong correlation with a higher rate of correctness at more specific clusters of smaller size.

4.1.2 Comparison with prior literature

Our final results do not provide a 1:1 comparison to that of the work of Nabieva et. al, as shown in Figure 9. Although we implement the same approaches, there are two main differences in the format of our results: testing structure and type of assignment. Our testing structure differs in the sense that we analyze the behavior of our prediction accuracy and specificity only as the neighborhood radius is increasing. Our approaches, similar in functionality to those of Nabieva et. al, work such that a given number of unknown proteins n are tested at once. In our results for radii 1 and 2, n was set equal to 5 when gathering results. Due to limited time and computational resources, we tested only one protein at a time for radii 3 and 4. This range of n is far less than in the referenced results with upwards of 2,000 tested proteins at a time.

The computational problem that we solve is a multi-label classification problem, whereas Nabieva et. al assign each protein only one of multiple functions. This adaption leads to our accuracy being quite higher than that seen in the original paper, since there are opportunities for the algorithms to correctly predict a subset of true function annotations, instead of only scoring a protein as correct or incorrect in a binary manner.

Despite these differences, we can perform comparative and direct analysis of our raw accuracy scores to those of Nabieva et. al as radius increases, shown in Figure 8. As mentioned, our results are not entirely compatible numerically with those in Figure 9, but the pattern of behavior of the accuracy is the same for Majority Approach and Functional Flow, with Functional Flow scoring higher than Neighborhood Approaches.

	Weighted F1	Non-weighted F1	Nabieva et. al [5]
$r = 1$	0.508	0.634	0.242
$r = 2$	0.306	0.532	0.188

	Weighted F1	Non-weighted F1	Nabieva et. al [5]
$r = \frac{1}{2}d$	0.719	0.938	0.311

Figure 8: Comparison of accuracies for Majority Neighborhood Approach and Functional Flow Approach between our paper and Nabieva et. al. Note: d represents the diameter of the PPIN.

4.2 Current state of the art in function prediction

Within the literature that we reviewed, the best observed algorithm was the Functional Flow algorithm. The algorithm was compared against the Majority, Neighborhood, and GenMultiCut algorithms. Neighborhood is an extension of the Majority Approach, where r is the radius around the protein p that is searched and the goal is to find "over-represented functional annotations" (Nabieva et. al, 2). GenMultiCut is the application of the multi-cut approach as an application of the multiway k-cut problem and that can also be formulated as an Integer Linear Programming problem; we did not implement this approach as part of our project. The comparison of results are referenced in Figure 9 below.

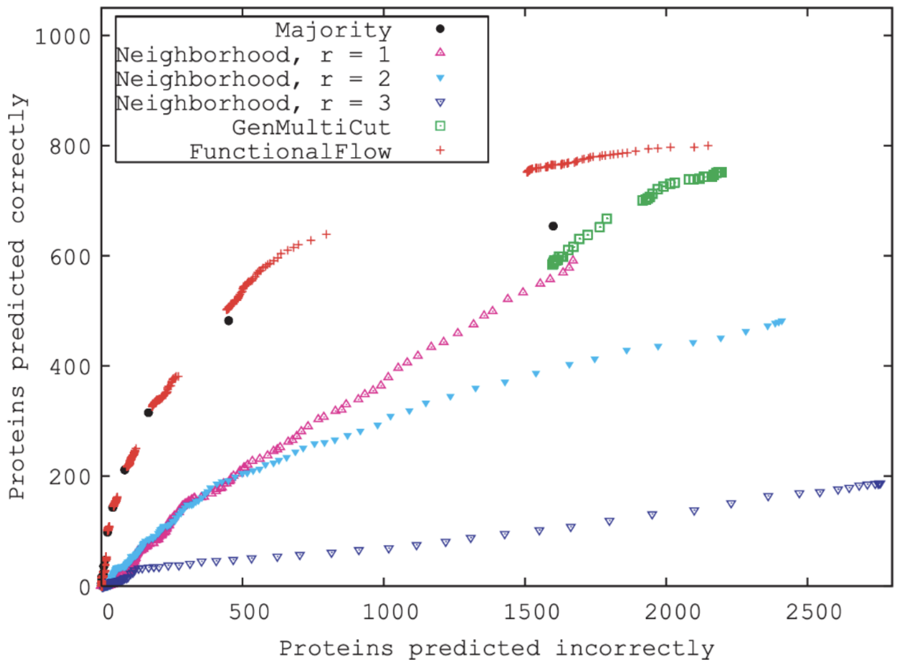


Figure 9: Results of Nabieva, et. al

The results concluded that Functional Flow was the best performing algorithm of the algorithms that were surveyed, with the Majority Approach performing better in scenarios where there were specifically more than 3 neighbors with known functions. The results also showed that increasing the neighborhood radius r in the Neighborhood approach leads to more inaccurate function prediction, which is consistent with the results that we saw throughout our implementation. The Neighborhood approach, which ignores the topology of the graph when performing its prediction and also instead uses χ^2 scoring, performed significantly more poorly than the Majority Approach, where scoring is done by summing up the number of times an annotation appears in the neighborhood of a protein.

More recently, there have been algorithms that utilize machine learning models in order to perform function prediction. Some of the experimental algorithms include support vector machines, decision trees, and Bayesian networks. There are some challenges with comparatively evaluating these models, including the necessity of potentially extensive biological experiments to determine “hotspots” of known protein activity, as well as overfitting as an issue due to the limitations on the amount of available data.

4.3 Future work

In our future work, we would like to further compare the performance of the approaches that we implemented with the GenMultiCut approach that we did not implement. Additionally comparing the approaches that were covered in the full scope of Nabieva, et. al would allow us to more comprehensively compare our results with the current literature. Additionally, we would want to compare the performance of machine learning algorithms with the approaches that we implemented. There are studies that compare different machine learning approaches, as well as studies that compare more naïve approaches; however, there has been no significant study that compares both groups of approaches.

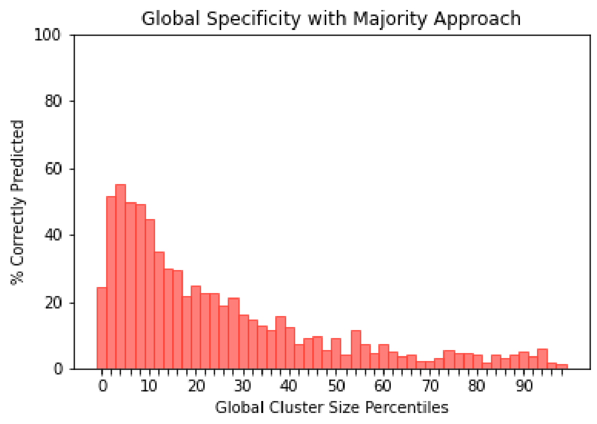
We also did not fully implement the Neighborhood approach, as our algorithm was an extension of the Majority Approach for varying values of r by using the summation of the neighbors’ functions and a Hishigaki label rather than their χ^2 score. An extension of our work could be to extend the comparison of different evaluation methods, such as using χ^2 evaluation in the context of Functional Flow. Additionally, we did not consider the local topology of the proteins within the graphs beyond the specificity scores when performing our evaluation. Another extension of our work would be to investigate methods that incorporate information about topology into their evaluation methods, such as approximation algorithms for graph similarity matching.

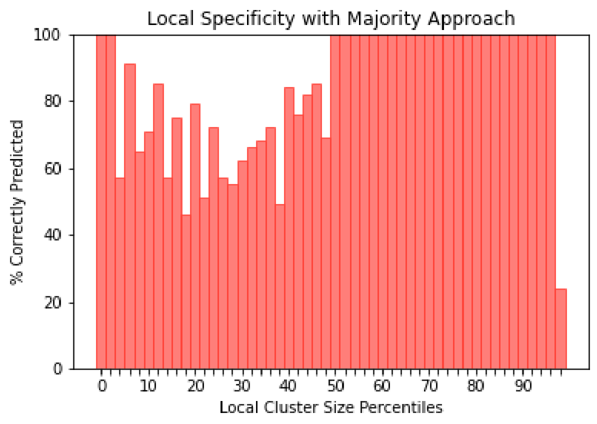
One additional idea that can be explored further is combining multiple methods to make a final prediction. This can be used to mitigate the disadvantages caused by each model specifically. For example, we might consider only marking an annotation as positive if two of the three methods agree. Taking a consensus can increase accuracy but may decrease specificity. We may also run the models with differing parameters and then take a consensus which can be weighted depending on the reliability of each model. Further, it is worth exploring how to combine other existing algorithms.

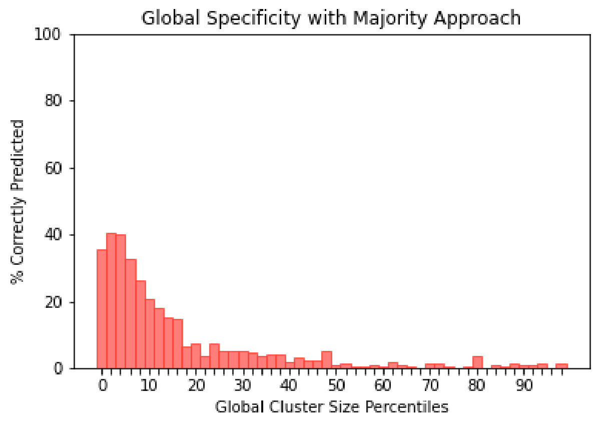
5 Appendix

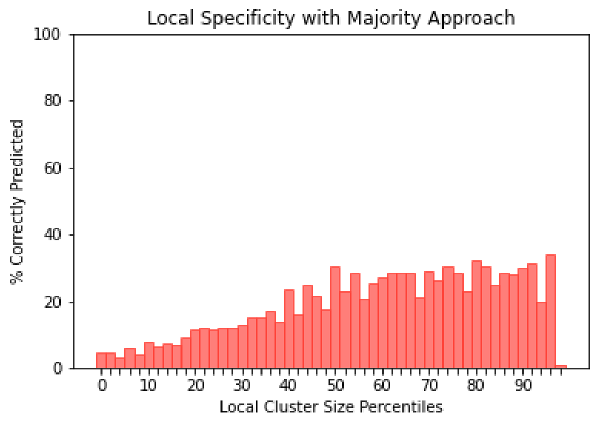
Below please find plots for global and local specificity of Majority, Functional Flow, and Sequence Alignment Approaches for all tested radii.

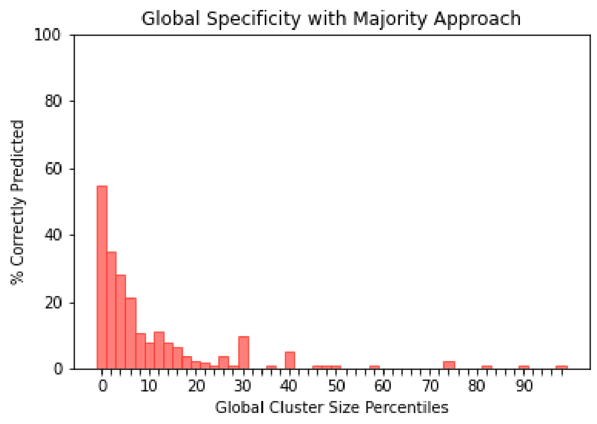
5.1 Majority Approach plots

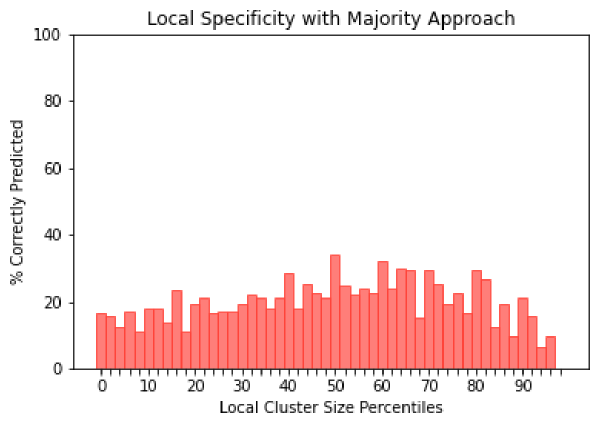


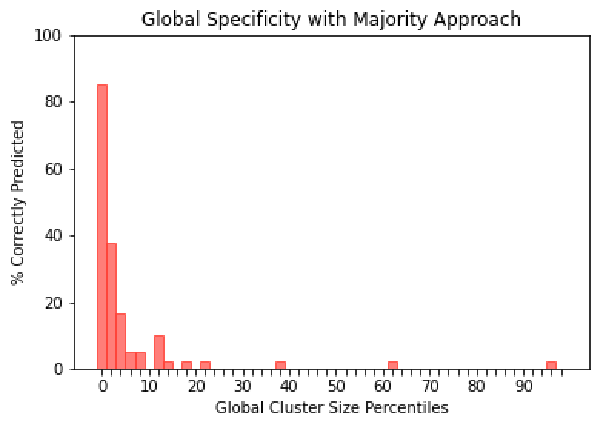


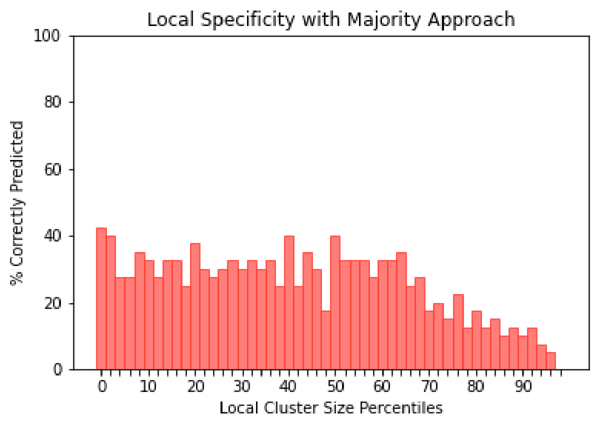




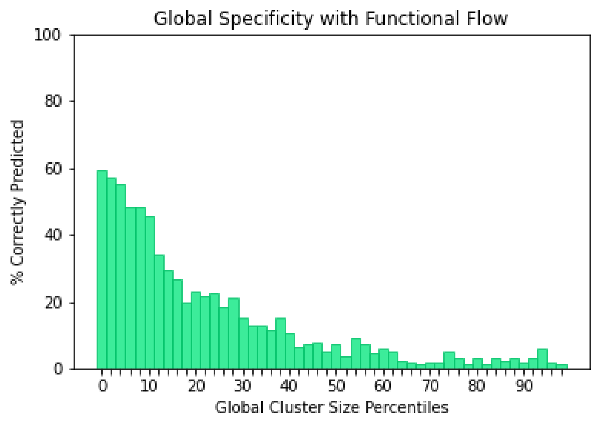


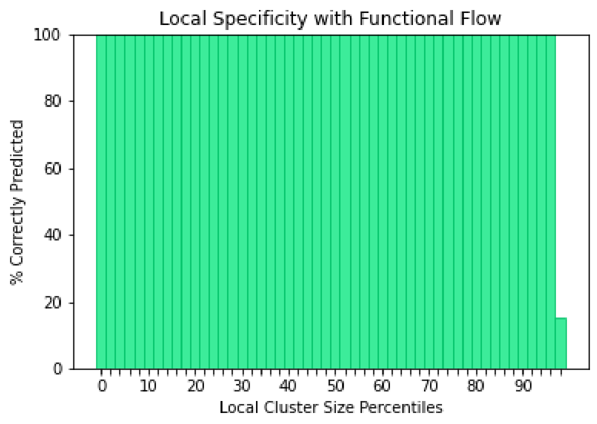


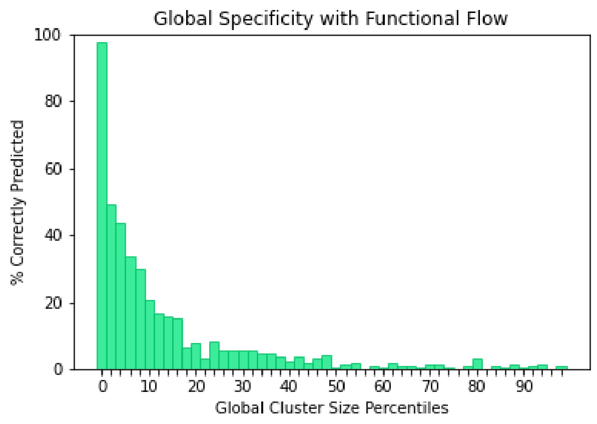


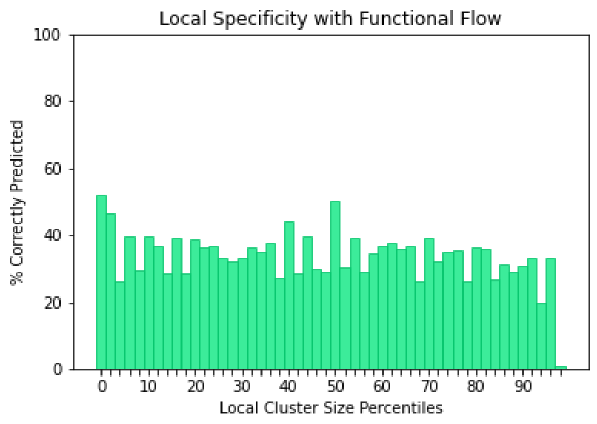


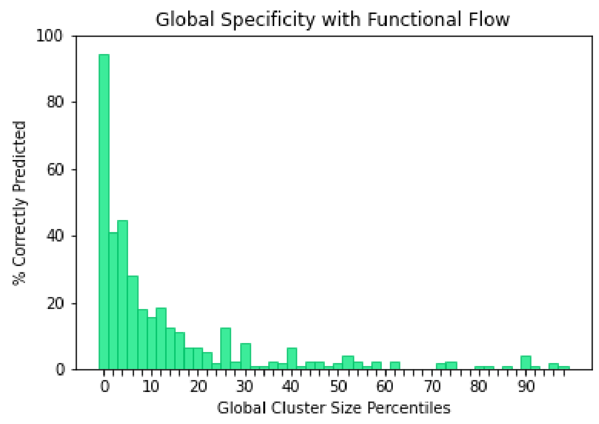
5.2 Functional Flow plots

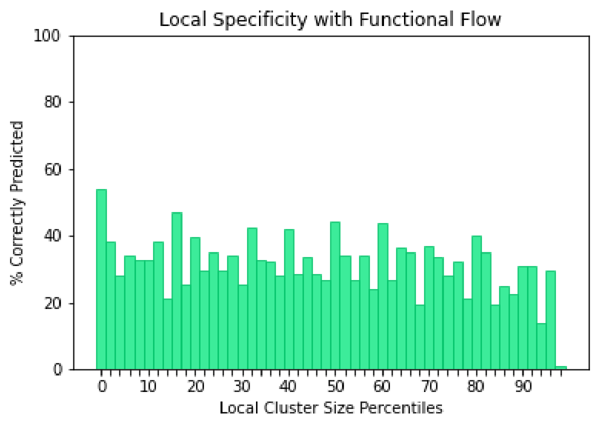


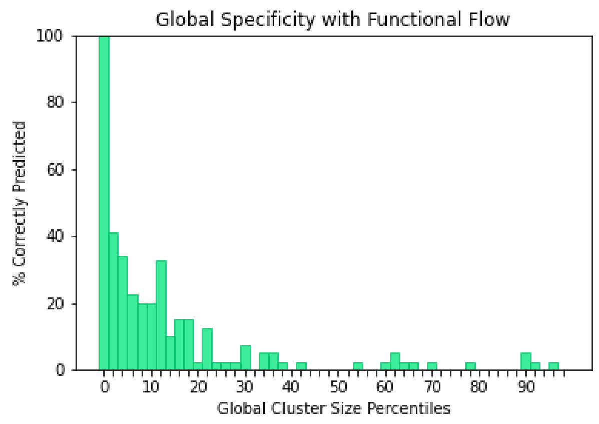


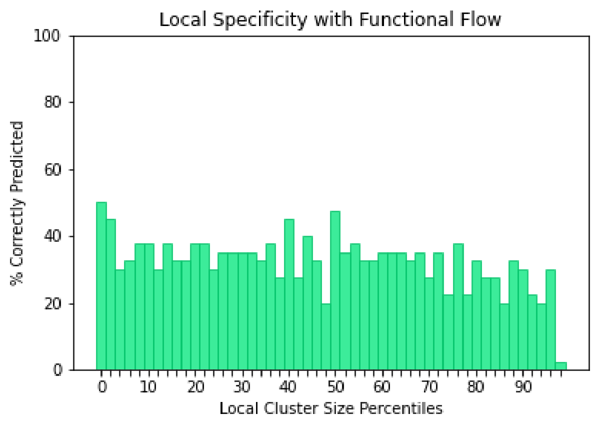




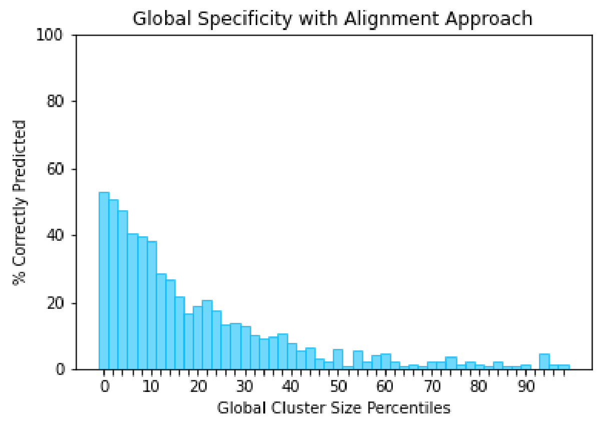


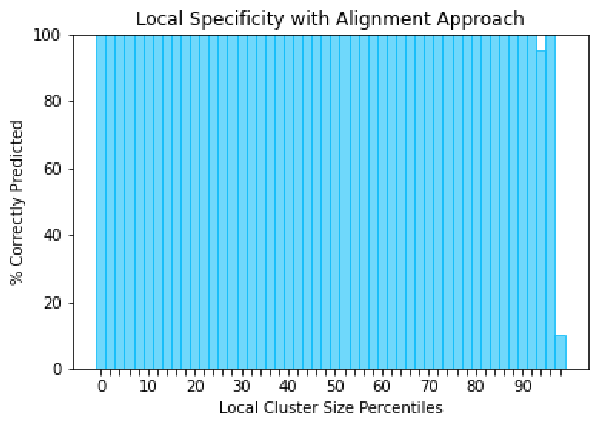


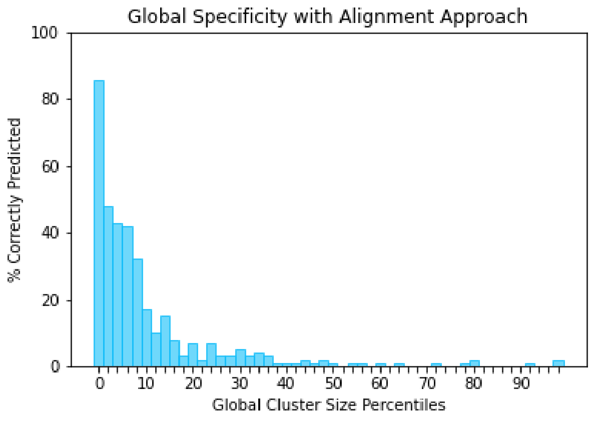


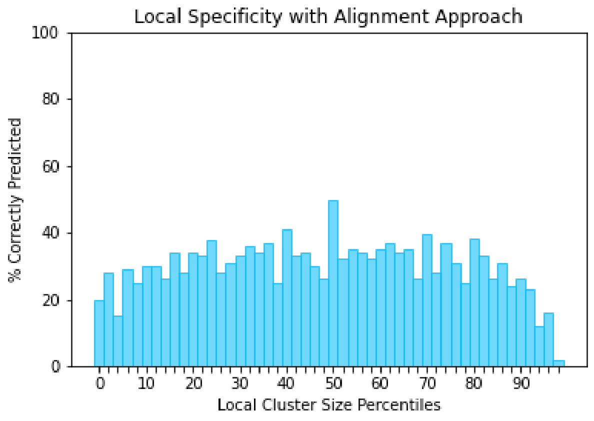


5.3 Peptide Sequence Alignment plots






Global Specificity with Alignment Approach
% Correctly Predicted
Global Cluster Size Percentiles
0
20
40
60
80
100
0
10
20
30
40
50
60
70
80
90


Local Specificity with Alignment Approach
% Correctly Predicted
Local Cluster Size Percentiles
0
20
40
60
80
100
0
10
20
30
40
50
60
70
80
90

References

[1] Strang, Gilbert. Introduction to Linear Algebra, Fifth Edition. Wellesley-Cambridge Press, 2016.

[2] Cruz, L. M., Trefflich, S., Weiss, V. A., Castro, M. A. (2017). Protein function prediction. Methods in Molecular Biology, 55–75. https://doi.org/10.1007/978-1-4939-7231-9_5

[3] Szklarczyk, D., Gable, A. L., Nastou, K. C., Lyon, D., Kirsch, R., Pyysalo, S., Doncheva, N. T., Legeay, M., Fang, T., Bork, P., Jensen, L. J., von Mering, C. (2021). The STRING database in 2021: customizable protein-protein networks, and functional characterization of user-uploaded gene/measurement sets. Nucleic acids research, 49(D1), D605–D612. https://doi.org/10.1093/nar/gkaa1074

[4] Protein-protein interaction networks. Protein-Protein Interaction Networks - an overview — ScienceDirect Topics. (2019). Retrieved May 6, 2022, from https://www.sciencedirect.com/topics/biochemistry-genetics-and-molecular-biology/protein-protein-interaction-networks

[5] Nabieva, E., Jim, K., Agarwal, A., Chazelle, B., Singh, M. (2005). Whole-proteome prediction of protein function via graph-theoretic analysis of interaction maps. Bioinformatics (Oxford, England), 21 Suppl 1, i302–i310. https://doi.org/10.1093/bioinformatics/bti1054

[6] Hishigaki, H., Nakai, K., Ono, T., Tanigami, A., Takagi, T. (2001). Assessment of prediction accuracy of protein function from protein-protein interaction data. Yeast, 18(6), 523–531. https://doi.org/10.1002/yea.706

[7] Loewenstein, Y., Raimondo, D., Redfern, O.C. et al. Protein function annotation by homology-based inference. Genome Biol 10, 207 (2009). https://doi.org/10.1186/gb-2009-10-2-207

[8] Maxat Kulmanov, Robert Hoehndorf, DeepGOPlus: improved protein function prediction from sequence, Bioinformatics, Volume 36, Issue 2, 15 January 2020, Pages 422–429, https://doi.org/10.1093/bioinformatics/btz595

[9] Lee, D., Redfern, O. & Orengo, C. Predicting protein function from sequence and structure. Nat Rev Mol Cell Biol 8, 995–1005 (2007). https://doi.org/10.1038/nrm2281

[10] Liu, S., Liu, C., Deng, L. (2018). Machine Learning Approaches for Protein–Protein Interaction Hot Spot Prediction: Progress and Comparative Assessment. Molecules, 23(10), 2535. https://doi.org/10.3390/molecules23102535

[11] Compeau, Phillip. "Sequence Alignment." Presentation at Carnegie Mellon University, Pittsburgh, PA, March 2022.

[12] Compeau, Phillip. "Function Prediction in Protein Interaction Networks." Presentation at Carnegie Mellon University, Pittsburgh, PA, March 2022.