Improving the Protein-Protein Interaction Prediction Engine (PIPE) with Protein Physicochemical Properties

by

Calvin Jary

A thesis submitted to the Faculty of Graduate and Postdoctoral Affairs
in partial fulfillment of the requirements for the degree of

Master of Applied Science
in Biomedical Engineering
(with a Specialization in Data Science)

Department of Systems and Computer Engineering
Carleton University
Ottawa, Ontario
December, 2019

Abstract

Protein-protein interactions (PPI) serve an important role in both protein and cell function. They are difficult and time consuming to determine experimentally and thus benefit from *in silico* prediction methods. This thesis improves a high throughput, sequence-based protein-protein interaction prediction method called the protein-protein interaction engine (PIPE). The initial contribution was in developing comprehensive documentation on how to run and use PIPE. Next, a Python implementation of the scoring of PIPE was developed. A software framework was created for the systematic exploration of how physicochemical properties can improve PPI prediction. Subsequently, a sequence-based solvent accessibility approach was integrated with PIPE, improving PPI prediction recall by 0.9% from 73.8% to 74.7% at 90% precision. Finally, 166 different sequence-based physicochemical properties were generated using the ProtDCal software tool and were integrated with PIPE using the framework developed in this thesis. The best of these properties improved the recall of PIPE by 2% at 90% precision. This improvement was shown to be statistically significant and was confirmed on a larger test set including 10,000 protein pairs known to interact and 10,000 randomly selected pairs, assumed not to interact.

Acknowledgements

I would like to thank my supervisor, Prof. James Green, for his exceptional levels of support, guidance and friendship throughout the course of this research. He has been an honour and a privilege to work with, and I'm grateful for the opportunity he has provided me with and the knowledge he has imparted on me.

I would also like to thank all the members of the Carleton University Bioinformatics Group for their advice and friendship throughout my work, particularly Bradley Barnes for helping me get started with PIPE.

Additionally, I would also like to thank the members of Carleton's Research Compute Services for their computational advice and computational resources. Specifically, Sylvain Pitre and Ryan Taylor.

A further thanks to all the other PIPE authors, whose initial work represents the seed for this research.

Finally, thank you to my mother, whose encouragement and support was absolutely critical to my success.

Table of Contents

List of Tables

List of Figures

List of Abbreviations

Abbreviation	Definition
AA	Amino Acids
CI	Class Imbalance
FPR	False positive rate (also known as 1 – specificity)
LOOCV	Leave-one-out cross-validation
PIPE	Protein-Protein Interaction Prediction Engine
PPI	Protein-Protein Interaction
P-R curve	Precision-recall curve (precision vs TPR curve)
ROC curve	Receiver Operating Characteristic
SLURM	Simple Linux Utility for Research Management
TPR	True positive rate (also known as sensitivity, recall)

1 Introduction

1.1 Background

Proteins carry out the vast majority of all structure, transportation, receptive, regulatory and catalytic activities within cells. They are essentially a linear polymer comprised of 20 different building blocks, referred to as amino acids. They have single letter representations that span every letter of the alphabet except for the letters: B, J, O, U, X and Z. These amino acids encompass a large gamut of physical and chemical properties. The instructions for which amino acids should be added and in what order is specified by a DNA sequence referred to as a gene. This linear polymer of amino acids can then fold into secondary structures such as α-helices and β-sheets, which are used to create a localized 3D structure and particular structural domains that can perform limited functions[1]. These secondary structures are formed and held together by polar bonds in which atoms that are slightly negative by having more electron distribution are attracted to atoms which are slightly positive by having less electron distribution. The entire protein can then fold into a specific global 3D structure, generally by means of thermodynamic favourability. This occurs when the interior of the protein is not polar and so does not benefit energetically from interacting with the water of the cellular environment (hydrophobic), and the exterior of the protein, which is more polar and so does indeed benefit energetically from interacting with water (hydrophilic)[1]. Additionally, there can be disulfide bonds in which distant regions of the protein are covalently bound together by means of two sulfur atoms so the protein may maintain its 3D dimensional shape. Finally, there are ionic interactions which also serve to create and maintain the 3D dimensional structure of a protein whereby the pH of the cell will turn polar amino acid residues into

ionic regions by adding or removing entire H^+ ions, and this will create what is essentially a far more powerful polar bond.

A protein's catalytic functions can often be performed in a lone wolf fashion, whereby the protein only needs to interact with its target small molecule and nothing else in order to perform a basic catalytic function. However, the structural and functional activities proteins must undertake often require the protein to positively interact with either sister proteins from its family, or proteins which are quite structurally different from itself which it must then inhibit, modify, or cooperatively engage with. Proteins forming complexes with or otherwise interacting with other proteins in order to fulfill their structural or functional duties is referred to as protein-protein interactions (PPIs), with a subset of PPIs being integral to a protein's final quaternary structure as shown in Figure 1. A famous example is hemoglobin, which is a complex of four globin proteins coming together and interacting positively, and this type of interaction is necessary for the hemoglobin complex to perform its function.

This critical quaternary structure is enabled by PPIs and these interactions are also critical to the mediation of a myriad of functions within the cell, and disrupting these interactions is often a drug target for small molecules, and interacting with PPIs is often the objective of peptide-based treatments.

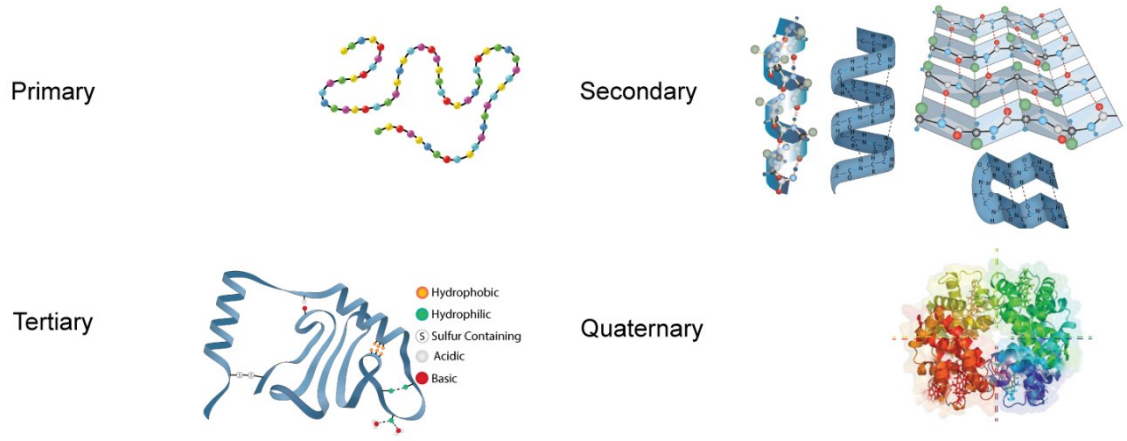


Figure 1: An illustration representing protein assembly. (a) features a linear polymer; (b) shows α-helix and β-sheets; (c) demonstrates a fully-formed 3D structure for an individual protein along with the five major forces driving this conformation; (d) displays multiple individual proteins interacting together via PPIs to form a larger functional complex. (Figure reproduced from [2])

Although the human genome has been thoroughly sequenced about two decades ago [3]; precisely identifying exactly how many genes we have, referring to sequences of functional and active DNA which code for unique proteins made at least in one of our tissues throughout our lifetime, has not been sharply defined, but is agreed to be between 20,000 and 25,000 [4]. The number of genes an organism contains in its genome has not been a reliable measure of an organism's complexity, as proven by the humble soybean plant having approximately 75,000 genes [5]. However these 20,000 to 25,000 genes, of which only a fraction is expressed in each of our approximately 200 different cell types, do not account for the necessary complexity required to maintain an individual cell, and all the diverse functions, signaling and transport that it must carry out. Additionally, proteins can be alternatively spliced, meaning one gene can code for multiple different proteins that

can be somewhat structurally different but fall within the same family. This effectively brings the human proteome closer to approximately 160,000 different proteins, however, this is still unlikely to account for all the complexity we see and that is required. Cells use additional mechanisms to introduce functional complexity, such as the post-translational modification of proteins, in which chemical functional groups of no more than a few atoms are added to diverse sites on proteins to change their shape or surface charge, as well as non-coding RNA which are significantly simpler than proteins but can fulfill some of the diverse functions of proteins. Thus the most compelling theory is that it is in fact PPIs that are critical not only to establishing quaternary structure, but to establish the systems and complexity required in molecular biology [6].

It therefore follows that the accurate and reliable determination of protein-protein interactions is a highly sought-after and crucial part of molecular biology and medical research, encompassing both the basic science of understanding how a cell functions, and also how a particular disease functions and can be treated. It therefore follows that the vast majority of diseases are the result of a genetic mutation producing a protein that malfunctions [7]. *In vitro* experiments have been devised decades ago that can accurately determine PPIs, especially if multiple techniques are deployed towards the same PPI objective. These techniques include protein affinity chromatography also referred to as pull-down experiments, yeast two-hybrid experiments, and immunoprecipitation experiments. These techniques however are all slow, expensive, and labour intensive [8].

High throughput methods have been developed alongside advances in robotics and automation, and they have proven to be methods capable of producing data for hundreds or even thousands of protein-protein interaction experiments, and have thus resulted in

datasets of observed interactions for entire organisms [9]. However, these high throughput experiments utilize techniques that are individually less reliable and the overlap of independent experiments has historically been low, which calls the dependability of high throughput results into question [10]. Fortunately, there is a brighter future for higher throughput methods in that new technological innovation has a greater potential to increase their accuracy than for low throughput methods, as there is simply more room for improvement available for high throughput methods [11].

1.2 Motivation

Due to PPIs being very valuable to elucidate and extremely difficult to determine accurately *in silico*; it is a ripe and worthwhile area of continual research and progress. To achieve maximum accuracy, a full quantum mechanical simulation including all electron dynamics and atomic movement of the proteins of interest and a shell of the surrounding cellular environment would be required. The necessary quantum electrodynamics understanding has long been available, however, the computational performance is likely centuries away [12]. This alludes to the field being fertile for clever computational approximations and compromises that allow for the best predictions with the fewest resources. Such approaches either aim for the greatest accuracy with a reasonable runtime suitable for a limited number of PPIs, or strive for reasonable accuracy with a very fast runtime suitable for high throughput applications [13]. Even in the current field of *in silico* PPI prediction, there exists software that focuses on higher accuracy predictions that are computationally expensive and suitable for a smaller number of PPI questions, and software which is less computationally demanding and less accurate individually but is feasible to run on an entire proteome. Expanding the capability of this latter category is the

domain of this work. Specifically, the protein-interaction prediction engine (PIPE), is an *in silico*, high throughput, sequence-based method developed here at Carleton University by the bioinformatics research group. Currently this algorithm only requires the amino acid sequences of the proteins of interest, and a database of known protein-protein interactions. The motivation therefore, is to maintain the versatility of PIPE by not requiring protein 3D structure as an input, but to leverage solvent accessibility or various physicochemical properties which can also be rapidly calculated from amino acid sequence, and use these features to inform the predictions of PIPE and make them more accurate.

1.3 Problem Statement

As a consequence of wet lab experiments being costly and time-consuming to determine PPIs, and as a consequence of computational resources being insufficient to simply simulate such interactions with complete accuracy *in silico*, there remains progress to be made on elegant approximations and strategies to *in silico* PPI predictions. Specifically, the objective of this work is to advance the accuracy and therefore predictive power of PIPE by leveraging the physicochemical properties of the pair of query proteins that can be computed entirely from sequence.

Furthermore, the goal is to keep the demand on computational resources reasonable, so that a single protein of interest can be screened against the entire human proteome in minutes, and every protein in the human proteome can be screened against every other protein in the proteome in under a month, given reasonable compute resources, typical of a research lab. Additionally, we want to continue to leverage the sequence-based advantages of determining PPIs, where no experimentally-determined 3D structure or sub-cellular localization is required.

1.4 Organization of Thesis

This work consists of four chapters. Chapter 1 presented an introduction into proteins, their immense structural and functional responsibility and why being able to accurately predict their interactions *in silico* encompasses significant applications. Additionally, Chapter 1 introduces PIPE as an *in silico*, sequence-based method to predict PPIs. Chapter 2 contains background on pattern classification, protein solvent accessibility, predicting PPIs *in vitro*, predicting PPIs *in silico* by leveraging: 3D structure, physicochemical properties, machine learning, and simply amino acid sequence. Lastly, the exact workings and scoring of PIPE are discussed in detail. Chapter 3 contains research contributions including implementing PIPE scoring in Python (PyPE), incorporating solvent accessibility by deploying NetsurfP 1 [14], incorporating a myriad of physicochemical properties by deploying ProtDCal [15], [16], and quantifying the PPI accuracy gained. Chapter 4 features discussion and conclusions of this work.

2 Background and Literature Review

2.1 Pattern Classification

Machine learning can be broadly described as using data in an automated way to implement an algorithm that trains a computer to make a decision autonomously. Pattern classification is considered a sub-topic of machine learning. The problem of predicting PPIs can be considered a binary classification problem, as a protein pair will either make enough intermolecular interactions to physically couple under normal cellular conditions, or the protein pair will not make enough of these attractive interactions to do so [1]. Therefore, a pattern classification algorithm can compute a score for how strong the prediction is that two proteins interact, and it can be up to the user to set a required score threshold to make the prediction of whether the protein pair will interacted or it will not. The reality will also be one of these two possibilities, and therefore the *in silico* prediction of protein interactions and the reality of protein interactions can only have one of four possible outcomes:

- True Positive (TP): in this case, the protein pair is predicted to interact and indeed does so in reality.
- False Positive (FP): The score assigned to the protein pair mistakenly lies above the set threshold and thus the pair was predicted to interact, but in reality, the pairs do not interact. This is also known as an incorrectly predicted positive, or type I error.

- False Negative (FN): The score assigned to the protein pair fell below the decision threshold and so was predicted not to interact, but in fact the pair does interact. This is also known as an incorrectly predicted negative, or type II error.
- True Negative (TN): in this case, the score for the protein pair correctly fell below the set threshold and thus the pair was predicted to not interact, and the pair indeed does not.

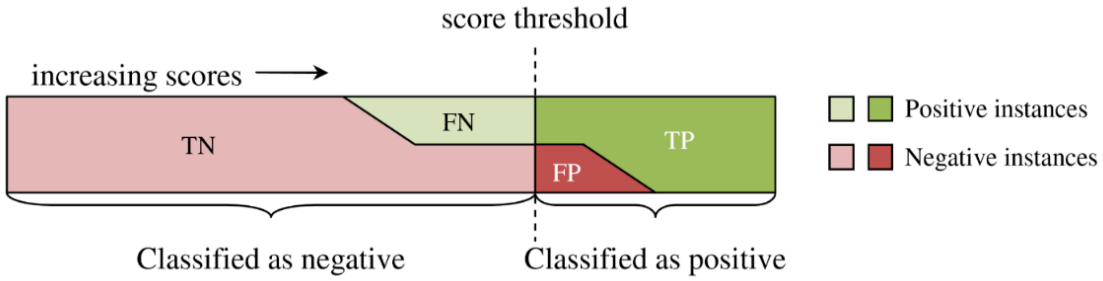


Figure 2: All four possible combinations by threshold of classifier score. Positives (P) are either correctly classified and are thus True Positives (TP) or incorrectly classified and become False Negatives (FN). Negatives (N) are either correctly classified and are thus True Negatives (TN) or are incorrectly classified and become False Positives (FP). (Figure reproduced from [17])

Thus, we can see from Figure 2 that if we increase the required threshold, we will have fewer true positives (TP) however, we will also have fewer false positives (FP). We can therefore see that the eternal struggle of binary classification in the field of pattern classification is between maintaining a stringent standard, causing fewer false positives but missing true positives, and having a looser standard allowing more true positives to be discovered but leading more false positives to slip through as well. Thus, this tradeoff must be made thoughtfully and should be considered entirely based on the application in question and the specific context. How many total positives (P) we may have, regardless of whether they are correctly classified and become true positives (TP) or incorrectly

classified and become false negatives (FN) should also be carefully considered, as this may be a perpetual unknown. This is also the case for the total negatives (N) and it is application-dependent as to which class we are more interested in, and which class can be more numerous. Figure 2 illustrates the concepts of TP, TN, FN, and FP.

2.1.1 Pattern Classification Performance Metrics

We can begin the discussion of how to gauge the performance of binary pattern classifiers by describing errors on the positive set as measured by using the true positive rate (TPR) also referred to as sensitivity, and it merely describes out of all positive protein interactions, what fraction were correctly identified.

$$Sensitivity\ (S_n) = \ True\ Positive\ Rate\ (TPR) = Recall\ (R) = \frac{TP}{P} = \frac{TP}{TP + FN} \quad (1)$$

$$Specificity\ \left(S_p\right) = 1 - False\ Positive\ Rate\ (FPR) = 1 - \frac{FP}{N} = \frac{TN}{TN + FP} \quad (2)$$

Whereas errors on the negative set are measured using the false positive rate (FPR), which is the fraction of negative interactions the classifier will incorrectly classify, and thus we use (1 – FPR) to convey the fraction of negative interactions the classifier will correctly classify, this I also referred to as specificity. Accuracy is more globally used term in the discipline and it is merely how many correctly identified positives and negatives were performed. The receiver operating characteristics (ROC) curve is ideally suited for the purposes of comparing sensitivity to specificity for a range of thresholds used by a classifier. However it is poorly suited for the use case of PPIs due to its equal attention

given to positives and negatives, as well as its susceptibility to class imbalance, which will be discussed subsequently [18]. Figure 4 illustrates a number of ROC curves.

$$Accuracy = \frac{TP + TN}{P + N} = \left(S_n \times \frac{P}{P + N}\right) + \left(S_p \times \frac{N}{P + N}\right) \quad (3)$$

However, for our purposes of binary classification for PPI, we are most concerned about two primary concepts: how many positive interactions we have captured over all possible positive interactions, this is captured by Sensitivity (S_n) also known as Recall (R). The second concept is of all the pairs we have predicted to be positive at our given score threshold, what is the ratio regarding how many are correct and thus are true positives, and how many are incorrect and thus false positives? This is known as Precision and it is described as:

$$Precision = \frac{TP}{TP + FP} = \frac{1}{1 + \frac{FP}{TP}} = \frac{1}{1 + \left(\frac{1 - S_p}{S_n}\right) \times \frac{N}{P}} \quad (4)$$

Class imbalance is a crucial concept in the training and evaluation of pattern classification algorithms, particularly in the field of protein-protein interactions. The premise is that if the negatives vastly outnumber the positives, then some performance metrics can fail to represent that reality and the desired objectives of the classifier. For example, if for every 1 positive PPI there are 999 negative PPIs, then a classifier that always predicts that PPIs do not occur will be correct 99.9% of the time and so have an accuracy of 99.9%! This is precisely why accuracy is a poor performance metric to use in the case of class imbalance, or in the case where positives are valued more than negatives, or vice versa.

To account for and attempt to quantify class imbalance, we can improve our measurement of precision to include class imbalance (CI), which is simply a ratio of the more numerous class to the less numerous class, and as such becomes a multiplication factor:

$$Precision = \frac{TP}{TP + FP} = \frac{1}{1 + \frac{FP}{TP}} = \frac{TPR}{TPR + FPR * CI} = \frac{S_n}{S_n + CI(1 - S_p)} \quad (5)$$

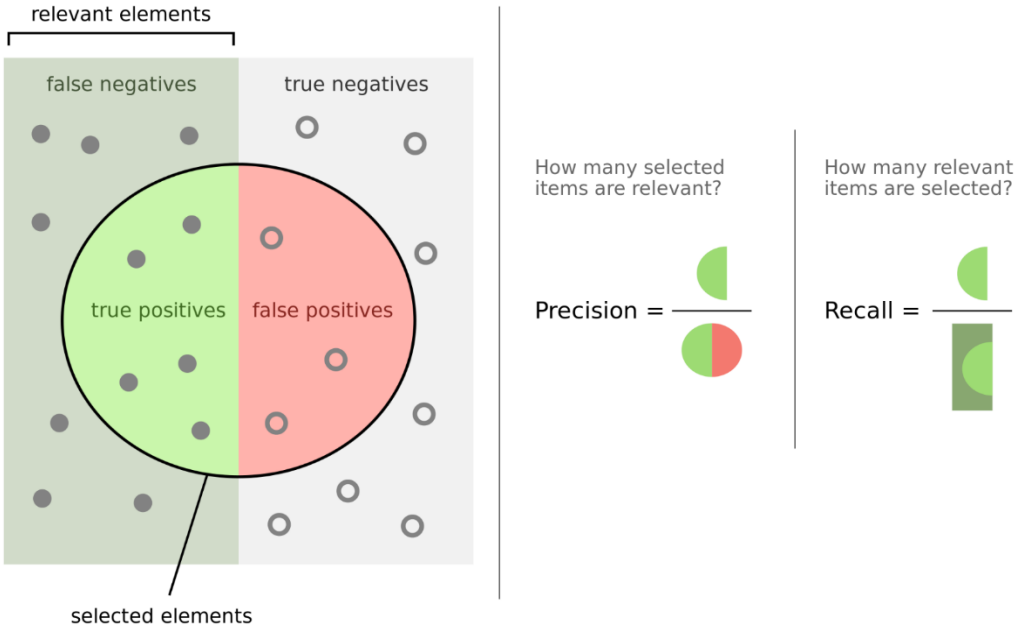


Figure 3: A cartoon representation of precision and recall. (Image source: https://en.wikipedia.org/wiki/Precision_and_recall creative commons license 3.0)

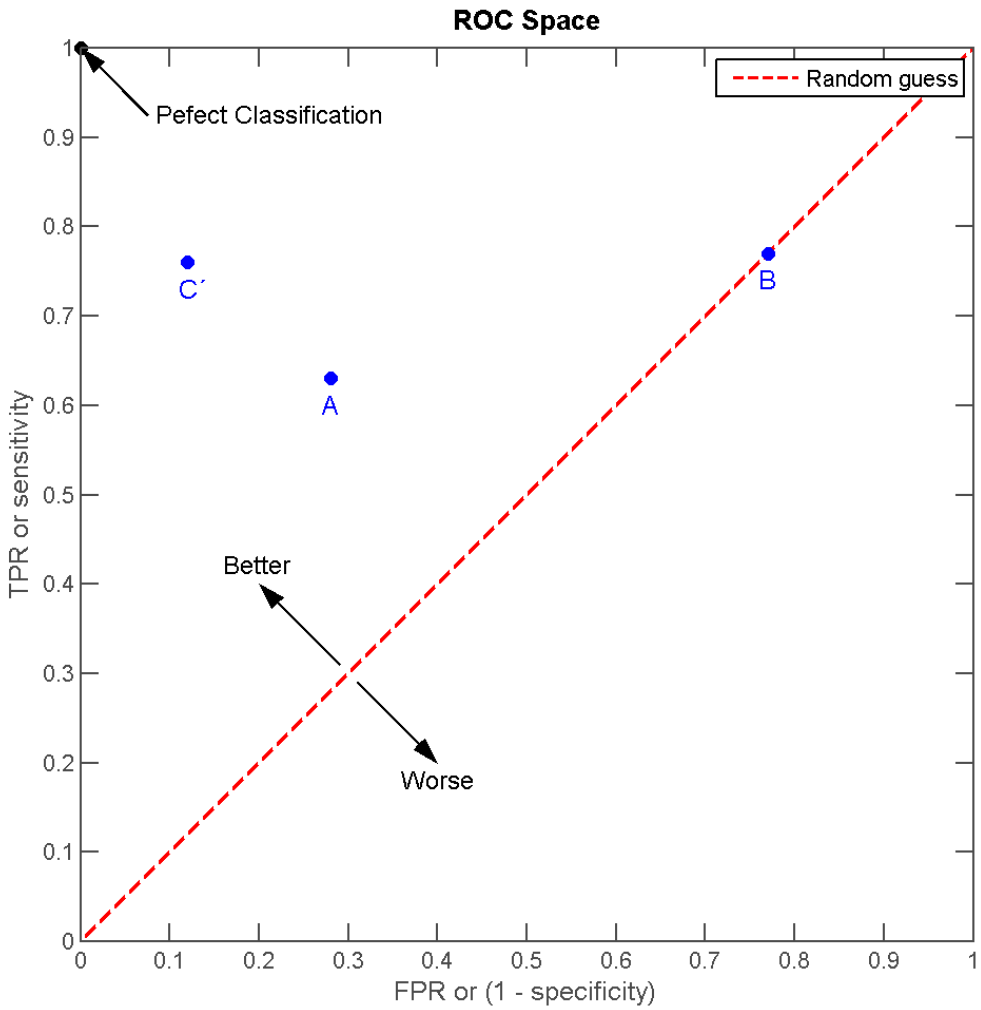


Figure 4: Receiver operating characteristic (ROC) curve Each point A, B and C represents the performance of a different classifier. The diagonal red line passing through B represents a 50/50 random guess for a balanced class. Point A represents a better classifier and point C represents an even better classifier. A perfect classifier would be a curve that reaches the top left of the ROC space. And any curve that reaches the bottom right area of the space falls below a random classifier in performance. (Image source: https://en.wikipedia.org/wiki/Receiver_operating_characteristic creative commons license 3.0)

The class imbalance found between interacting (PPI) and non-interacting (negative) protein pairs is not well known, even within a given species. An accurate assessment would require knowing the total number of proteins, which still eludes us even for *Homo sapiens*, and also the full knowledge of all PPIs, at which point predicting PPIs for that species would have limited value. That being said, there is precedent in the literature for this value being 1:100. Meaning on average; for every protein that our protein of interest does interact with natively, there are a hundred proteins that it does not interact with [19], [20].

Precision and recall are valuable performance measures for our application of determining PPIs as they not only utilize the ratios of correct predictions and mistaken predictions, rather than the raw numbers, but they also emphasize the positive predictions over the negative predictions by utilizing precision and recall. This is superior to another common performance measure known as the receiver operator character (ROC) curve, which is susceptible to class imbalance. The ROC curve also uses recall however instead of precision it uses false positive rate (1 – specificity). Measuring precision and recall values for a sliding score threshold can be turned into a curve known as a precision-recall curve, in which we see that as we set our required score threshold for a positive classification higher and higher, our recall necessarily decreases and our precision correspondingly increases.

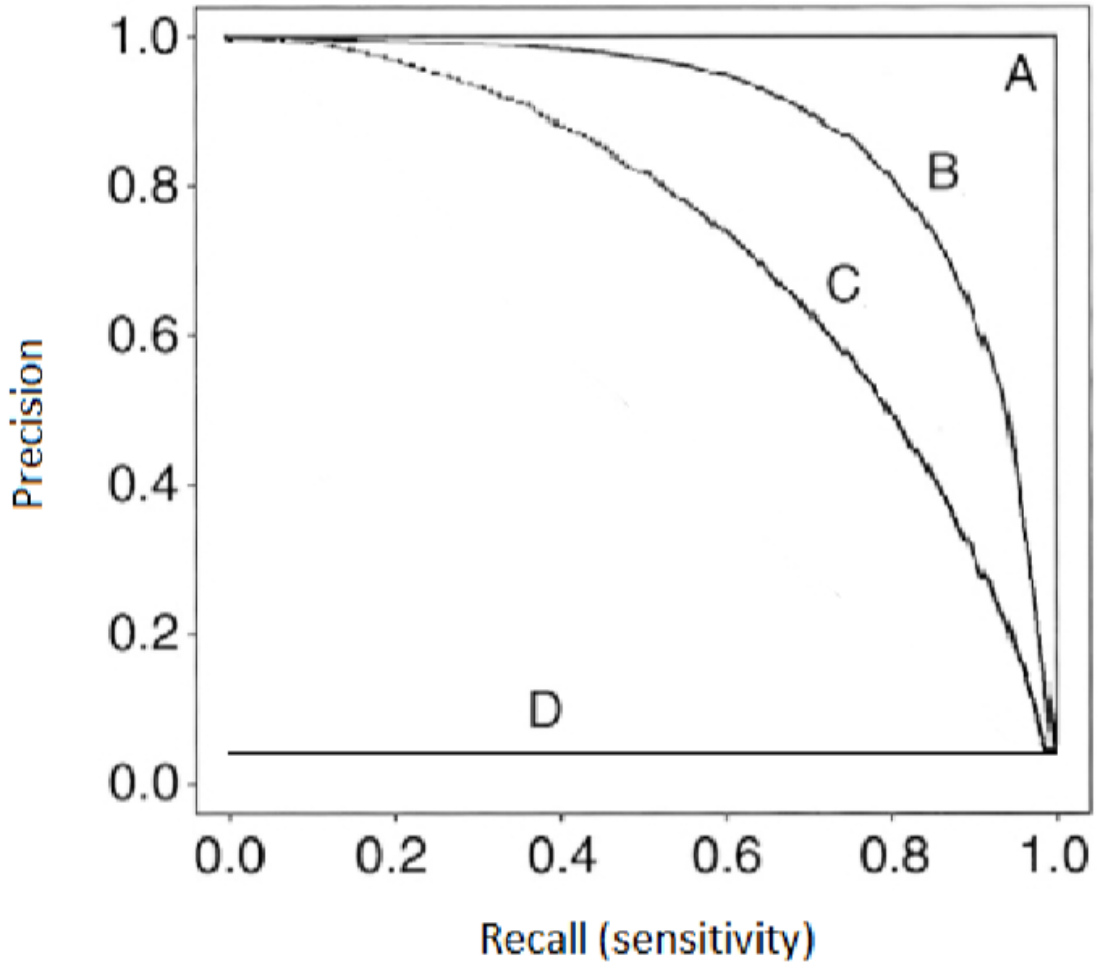


Figure 5: Precision recall curve Line A represents a perfect classifier, with no false positives and no false negatives. Lines B and C represent two non-perfect classifiers in which B has a higher performance characteristic than C, at all operating points. Line D represents a classifier made with random guesses, for a class imbalance of 20:1. (Figure adapted from [21])

2.2 Protein-protein Interactions and *in vitro* Experiments

Protein-Protein Interactions are interactions essentially based on favourable electrostatic interactions. These electrostatic interactions include hydrogen bonding, dipole-dipole interactions, ionic bonds, and hydrophobic interactions. As a result of their electrostatic nature, these interactions are far weaker than covalent bonds, also known as "chemical" bonds. PPIs can be broadly categorized into two groups: the first group are stable interactions, whereby sufficient portions of the proteins are interacting and enough interactions are taking place to result in a relatively strong interaction, which is stable over time in the cell and also stable when removed from a cell during an *in vitro* experiment, and the second group, known as weak or transient PPIs, in which relatively few positive interactions are taking place. Since these transient interactions can disassemble relatively easily *in vitro*, before they can be discovered, precautions must be taken to "fix" the proteins, turning these transient interactions into permanent covalent interactions before extraction, so the interactions are not lost [22].

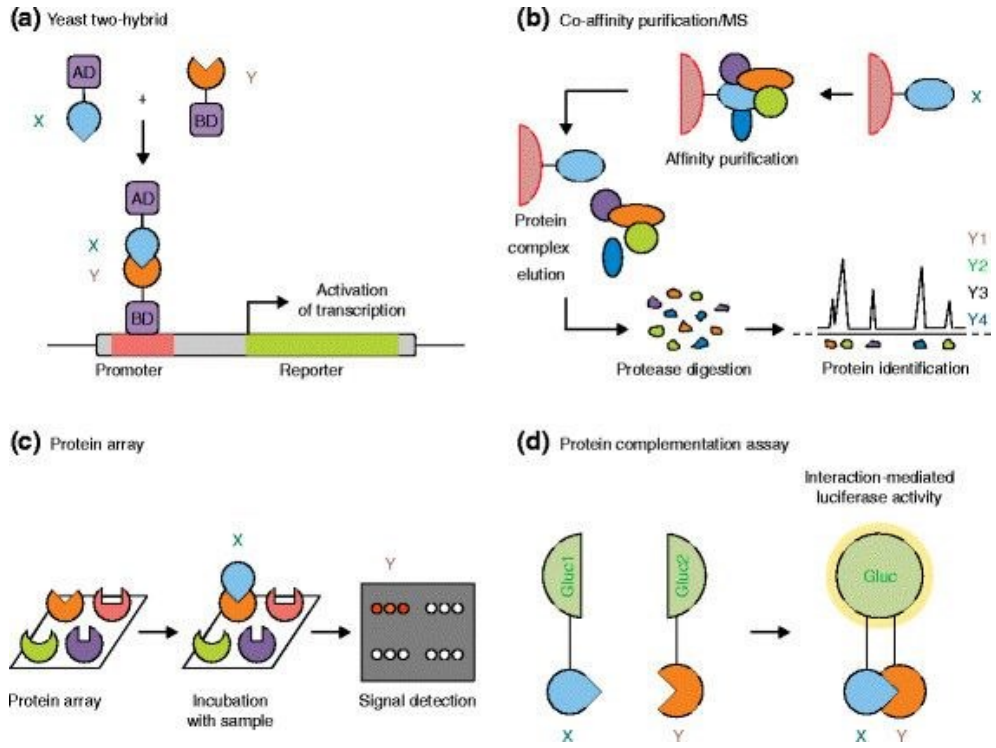


Figure 6: The gamut of techniques used for high throughput PPI screening (a) Yeast two-hybrid leverages the reconstitution of a working transcription factor. This is accomplished by attaching the prey protein to a transcription activation domain (AD) and the bait protein to the DNA-binding domain of the transcription factor (BD). The reporter gene will only be expressed if a PPI exists between predator and prey **(b)** Co-affinity purification/MS first captures prey with the bait protein and then identifies them with mass spectrometry (MS). **(c)** The protein array or "protein chip" can house thousands of different proteins attached at high density on a surface. Following binding of a prey protein, the interaction can emit fluorescent, photochemical or radioisotopic tags. **(d)** Protein-complementary assays require a fusion of two inactive fragments of a luciferase, one bound to the prey protein and one bound to the bait protein, to form the complete working complex which is then detectable. (Figure reproduced from [23])

Currently, protein-protein interaction experiments are performed *in vitro* by a few techniques which vary by reliability. Fundamentally, they all work on a similar principle whereby a protein of interest, referred to as the "bait" protein is "tagged" by some mechanism where it can easily be chemically removed later, and any protein that binds positively with this "bait" protein of interest is referred to as the "prey" protein, and it will also be removed along with the "bait" protein. Once the prey protein is removed, it can be identified using either antibodies, mass spectrometry, or column chromatography. These techniques include co-immunoprecipitation (co-IP) and Pull-down assay experiments as the most reliable experiments that require stable interactions, with crosslinking protein-interaction analysis, label transfer protein interaction analysis and far-western blot analysis able to elucidate weaker and more transient PPIs. These weaker interaction experiments work by first covalently crosslinking the proteins to turn sparse hydrogen bonding and polarity interactions into permanent covalent bonds which will persist during the subsequent purification steps [24]. High throughput versions of these techniques are summarized in Figure 6.

Unfortunately because a significant fraction of these *in* vitro experiments were performed several decades ago under the paradigm that over-expressing the bait protein and any prey protein interaction partners it may have, at levels far beyond those found naturally, in order to successfully perform the experiment, was a valid approach and would not alter any normal endogenous interactions, or lack thereof, these proteins would experience. As it turned out decades later, these high concentrations of bait and prey proteins did indeed result in artificially high numbers of interactions not found naturally, and this contributed to as much as a 40% rate of false positives, especially with yeast-two

hybrid experiments [8], [25], [26]. Additional issues arose if the proteins were altered by the tagging necessary in order to be easily purified out, and these alterations changed the shape, configuration and binding partners of the bait and prey proteins to an unknown extent. The final issue is that the act of lysing a cell to retrieve the bait protein could cause novel binding with prey proteins in an entirely different organelles and regions of the cell in which the bait protein would never be localized, and this can cause binding contamination of the bait protein with what could be argued are false positives.

2.3 Predicting Protein-Protein Interactions *in silico* from 3D Structure

The previous section described how to determine PPIs experimentally in a lab, with some discussions as to the difficulties, cost and time. This leaves room for faster and cheaper methods using *in silico* approaches. These computational approaches to PPI prediction can largely be grouped into two main categories: Whether there is available protein tertiary structure (3D structure) and secondary structure information (α-helicies and β-sheets), or whether such 3D structure is unavailable, which generally speaking is the case. Figure 7 illustrates the six general approaches to *in silico* protein structure prediction.

When protein tertiary and secondary structure is available, it enables approaches such as protein docking (Figure 7C) whereby the 3D structures of the two query proteins are tested for surface complementarity by way of calculating the electrostatic interactions using atomic and molecular dynamics techniques. Unfortunately, we currently have approximately 6,000 of these required 3D structures out of the estimated 20,000-25,000 human proteins, which are made freely available on the largest protein structure database of protein databank (PDB). This implies there is a strong likelihood that the protein of interest and a significant majority of its potential binding partners have unavailable 3D

structure, making structure-based, global PPI predictions unusable except for with particular proteins of interest. Furthermore, the full leveraging of 3D structure information using contemporary molecular dynamics simulations is very computationally intensive, and can require many months of processing even on powerful distributed clusters.

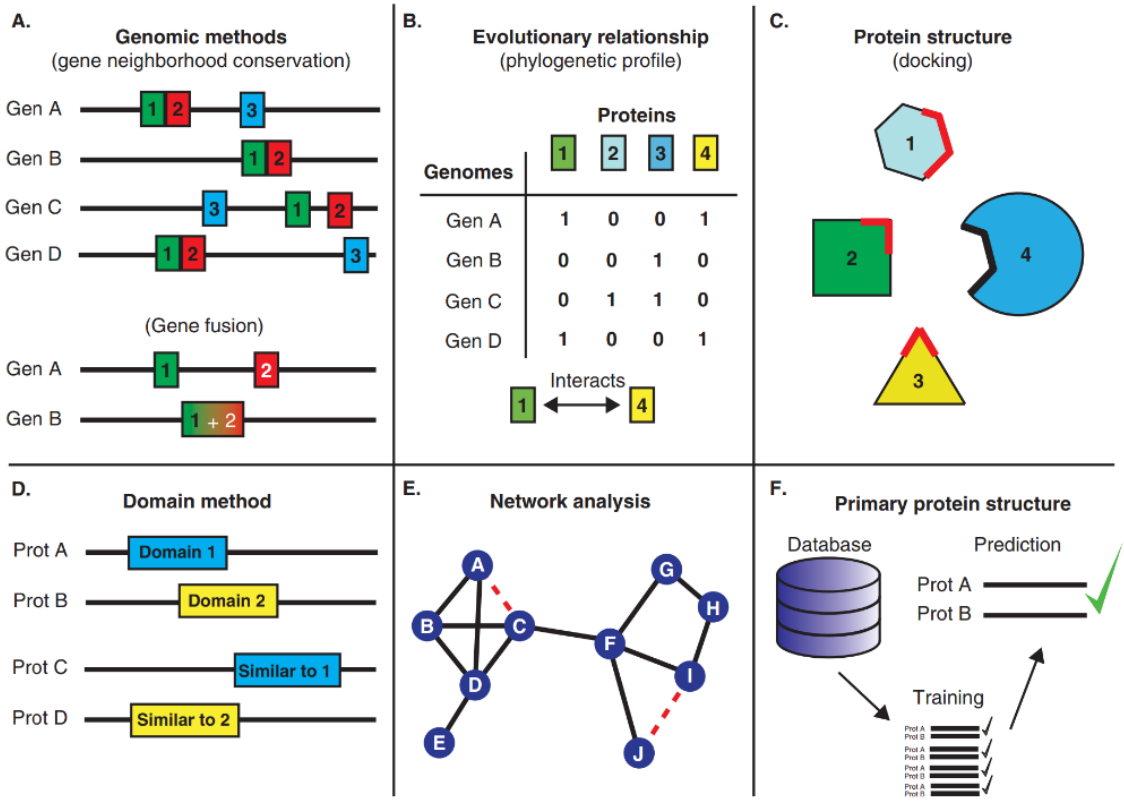


Figure 7: The six primary categories of *in silico* PPI methods (a) genomic methods, (b) evolutionary relationships, (c) protein structure, (d) domain methods, (e) network analysis and (f) primary protein structure. Only method (c) requires information about protein 3D structure. (Figure reproduced from [27])

The remaining five categories described in Figure 7 do not require 3D structure, with PIPE being similar to Figure 7F, whereby existing interaction knowledge is stored in a database and is leveraged to perform a prediction. The goal of this work is to introduce a method similar to Figure 7D, whereby knowledge about protein regions, secondary structures or domains can be computed from amino acid sequence and these physicochemical properties can essentially be used as an additional database leveraged in Figure 7F. Methods from Figure 7A, 7B, 7E attempt to leverage proximity to inform PPI predictions. This includes proximity in where proteins are localized on a genome, proximity across organisms on a similar evolutionary path, and proximity by functional association, meaning if protein A interacts with B, and B with C, then A is likely to have some functional relationship with C.

2.4 Introduction to Protein Physicochemical Properties

Proteins are linear polymers of amino acids, of which there are 20 unique types. Each amino acid has differing physical (e.g. size) and chemical (e.g. polarity) properties, which can be quantified using a number of scales and metrics. The physicochemical properties of a protein or peptide corresponds to the vector of quantified physicochemical properties of the linear chain of amino acids, comprising the protein. The "physicochemical properties" of a protein tell us a lot about how a protein folds or interacts, by studying how its atoms behave with each other, as well as with the atoms of the environment. Furthermore, because the surface of a protein primarily does the interacting, and the electrons of an atom are both the most active components of an atom and the components on the interacting surface of atoms, we can simplify the field of protein physicochemical properties further to simply: the study of the electron density on the surface amino acids of a protein. This is indeed a

simplification as there are many other features which can be considered, including amino acid residue mass, ionic charge, pKa, chemical functional groups, aromaticity, nonpolarity, thermodynamic stability, and more. However two critical pieces of information when it comes to predicting PPIs without prior 3D structure knowledge, *in silico,* are firstly: which of the amino acids of the protein are likely on the surface and thus can engage in PPIs, and secondly, can we determine what the electrostatic surface the protein will present, as this will strongly impact what electrostatic surfaces the protein will then be attracted to and interact with. The more advanced application of protein physicochemical properties is to utilize our full understanding of quantum chemistry as well as atomic and molecular physics to simulate the atoms and electrons within a protein and thus understand exactly the 3D structure it would undertake given its amino acid sequence, as well as which proteins it might bind with. This knowledge is well understood, as the quantum electrodynamics that underpins electrons and thus protein interactions is regarded as the best prediction in all of physics, as the theory matches the experiment to 11 significant digits [28]. A summary of how atomic physics and quantum chemistry can be built-up to model proteins can be seen in Figure 9.

2.4.1 Introduction to Absolute and Relative Solvent Accessibility

Solvent accessibility can be thought of as simply the property describing which amino acids find themselves on the surface of the protein where they are able to engage in PPIs and must also contend with the solvent environment, which in cells tends to be quite polar as it is highly populated by water molecules. Absolute solvent accessibility surface area (ASA) is defined by rolling a sphere the size of a water molecule over the protein surface. Relative surface area (RSA) is the ratio of the solvent accessible surface area

(ASA) of a given residue divided by the maximum obtainable solvent exposed area ASA_{MAX} for that particular amino acid within an intended peptide chain flanked by glycine or alanine, which are tiny amino acid residues.

$$RSA = \frac{ASA}{ASA_{MAX}} \times 100\% \quad (6)$$

2.4.2 ProtDCal: a Method for Computing Protein Descriptors from Physicochemical Properties

The ProtDCal software package is able to generate hundreds of distinct physicochemical features, and then aggregate these individual amino acid scores in a multitude of ways to generate tens of thousands of numerical features [15]. These physicochemical features are the result of amino acid inputs being converted to residue groups, then modified by the local neighbourhood of amino acids, and finally aggregated into scalar values. These values (descriptors) are then suitable for use in artificial neural networks, support vector machines and random forests trained to inference these diverse protein descriptors. A schematic of these features and how they are generated is shown in Figure 8. During the first step, residue indices are generated, which are extracted from the AAindex database. These do not leverage machine learning and are generally either a direct lookup, or can be calculated relatively easily as they generally only include the amino acid of interest and rarely a few of its neighbors. These indices can be described in three primary categories:

- Thermodynamic properties, which are physics-based and describe protein folding stability, they deal with inter-amino acid electronic interactions, Van der Waals

interactions, dihedral torsion potential, backbone hydrogen bond formation and hydrophobic effect. These indices include both 3D structure-based and sequence-based properties.

- Topographic properties, including versions of structural descriptors, most of them designed to assess protein folding rate. Some examples are relative contact order, long range order and total contact distance.
- Property-based indices, containing a set of empirical indices with fixed values for every particular amino acid, and so are a simple lookup. They include Kyte-Doolittle scale of hydrophobicity, Levitt's probabilities of adopting alpha helix, beta sheet or turn conformation, isoelectric point, and various transferable atom equivalent (TAE) values that refer to electrostatic potential distribution and electrostatic potential.

Following the first step of indices, the second step applies vicinity-based or weighting-based methods to modify the previous index values according to the particular electrostatic neighbourhood. These modulations include autocorrelation, Kier-Hall electrotopological state, Ivanshiuc-Balaban and Gravitational-like operators. The third step goes beyond merely the local electrostatic state of an amino acid of interest and looks at groups, domains and secondary structures. This is also a family of three general types:

- Type-based groups, these are groups that correspond to a particular native amino acid, all twenty amino acids are represented here, with the score given to how similar the current amino acid is to each reference amino acid
- Property-based groups, this includes the physicochemical properties of amino acids and includes polar, basic, acidic, aromatic, aliphatic, non-polar, common amino acids in alpha-helicies, common amino acids in beta-sheets, and similar

- Topographic or structure-based groups, these features describe which residues are part of a domain that is predicted to engage in alpha-helix, beta sheet, reverse turn and loop conformations, as well as regions deemed to be internal within the protein, or external to the protein's surface.

The fourth step includes aggregation operators, which are intended to dimensionally reduce the values for a group or domain of amino acids into a scalar value, and can be summarized into four main groups:

- Distance measures, which feature the Manhattan distance, Euclidian distance and the Minkowsky norm
- Measures of central tendency, this approach includes aggregation functions such as arithmetic mean, geometric mean, harmonic mean
- Measures of statistical dispersion, this includes standard deviation, variance, kurtosis, skewness, and examining quartiles.
- Measures on information theory, this approach measures total information content, mean information content and standardized information content. These approaches are based around binning and varying the number of indices per bin.

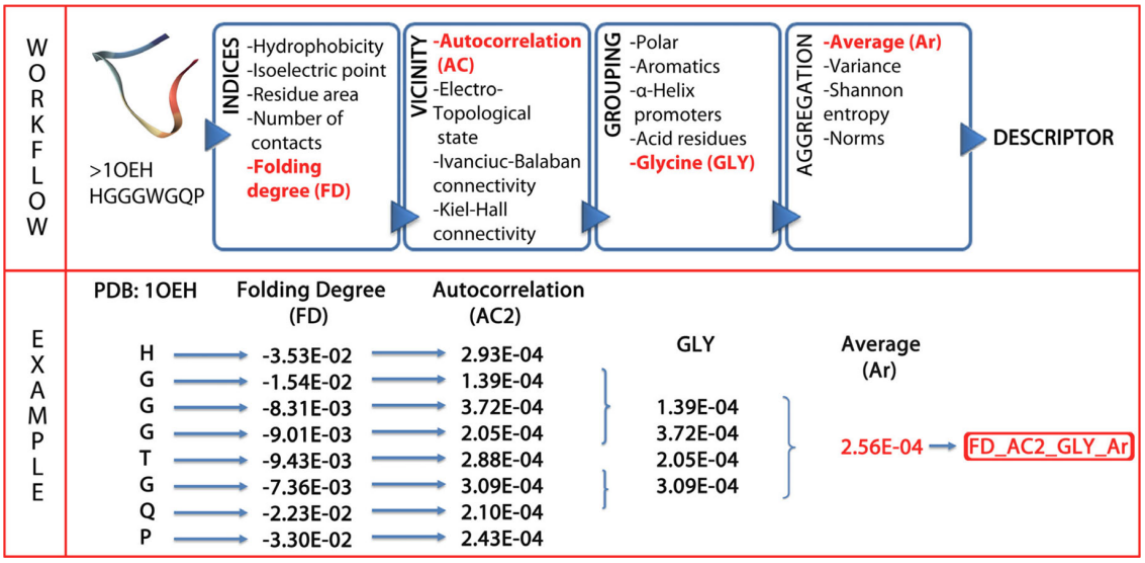


Figure 8: Simplified diagram of the physicochemical features generated by ProtDCal. Step 1 includes residue codification (Indices), Step 2 includes modification by vicinity, Step 3 includes grouping by subarrays, Step 4 features aggregation operators. (Reproduced from [16])

2.5 Predicting Protein Properties *in* silico from Physicochemical Properties

In the likely case that a 3D structure profile is not available for a given protein of interest, attempts have been made to leverage our knowledge of computational chemistry and computational atomic physics to help determine the 3D structure, function and possible interactions a protein may possess or engage in. This is achieved by using the particular amino acid sequence of the protein, and analyzing the particular global and local physicochemical properties the protein must possess given that particular amino acid sequence in that particular order [29]. Utilizing computational means to probe the physicochemical properties of a protein to understand more about that protein is a broad field with many approaches and a myriad objectives, however the application would be to leverage physicochemical understanding to determine which particular amino acid residues

would most likely be on the surface of a protein, and so could engage in PPIs, and which amino acid residues are likely to be in the interior of the protein, and so are unlikely to directly engage in PPIs. Physicochemical properties that could be used for this application would be the amino acid composition, along with what protein conformations and secondary structures this could induce; the most common secondary structures being the aforementioned α-helices and β-sheets, how polar or hydrophilic these amino acids and structures may be, whether they are likely to form any ionic bonds, hydrogen bonds, or disulfide bridges with neighbouring amino acid residue regions [30].

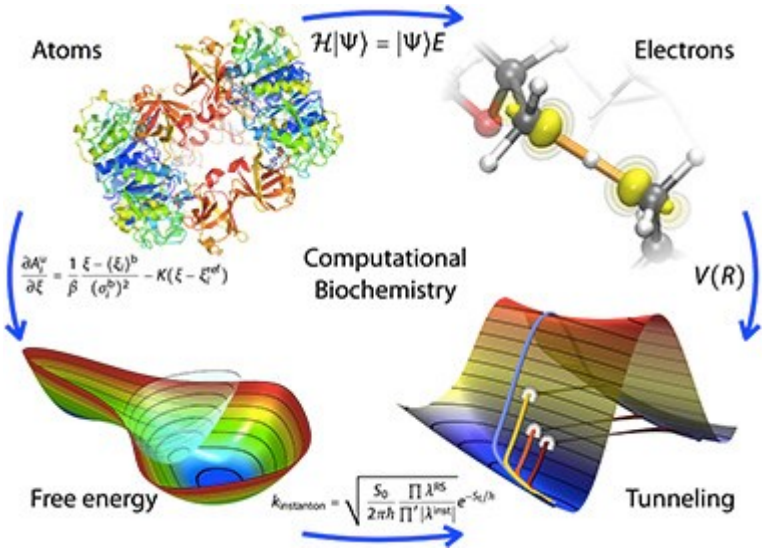


Figure 9: An illustration representing the quantum chemistry and electrodynamics understanding already possessed to determine protein 3D structure from sequence Unfortunately simulating our existing understanding *in silico* to generate accurate structures is beyond our computational means, necessitating clever simplifications and approximations. Image source: https://www.itheoc.uni-stuttgart.de/

One key approach to use these physicochemical properties to help inform PPIs would be essentially the approach of solvent accessibility, also known as surface accessibility. This is the approach of attempting to determine which portions of a protein or particular amino acid residues are the most likely to be on the surface of a protein and therefore most likely to engage in protein-protein interactions [31]. As an example, if we see a region of twenty amino acid residues which are overwhelmingly non-polar (hydrophobic), we can be informed that this region is unlikely to be on the surface of a protein, as when exposed to the water-rich interface of the protein surface, these amino acid residues would be energetically unstable, and through normal protein gyrations and pulsations, this region would likely find itself pushed to the predominantly hydrophobic interior of the protein where it is more stable, while a more polar (hydrophilic) amino acid region would take its place on the protein's surface where it would too be more stable. The level of sophistication for this approach can of course vary from a simple and intuitive method used in this example to complex modelling of electron density and electrostatic potentials exchanged with the interacting water molecules and neighbouring amino acid residues. This is perhaps an unnecessarily computationally complex approach as the solvent is constant, the properties of the various amino acids residues remain constant and the types of interactions between the residues also remain constant. All of this consistency lends itself to precomputation, approximation, caching and other computational approaches that leverage predictability for decreased computational effort [32].

2.6 Predicting Protein-Protein Interactions from Sequence *in silico*

Far greater versatility can thus be found using methods that can predict PPIs from the protein sequence alone. Two decades after the completion of the human genome

project, it has never been cheaper or easier to sequence an entire organism's genome, and with that a full understanding of the sequence of the proteome is within grasp. For example the human genome can be fully sequenced for under $1000 and can be done within one or two days, and even an hour using concerted high throughput sequencing [33]. With this being the case; all mainstream organisms used for research have had their entire genome sequenced many times, and as such this information has been validated and is very reliable [3], [34], [35]. Sequence-based methods can leverage and tend to leverage machine learning methods, whereby a binary classifier is trained on the available protein sequences, as well as on the available known protein-protein interactions. Fortunately, experimental PPI data is much easier to acquire and is more abundant in the literature than 3D structure data, and does not require computational horsepower or clever quantum chemical approximations as for physicochemical property data. Referring to the earlier example case of humans, there are approximately 55,000 known high confidence human PPI interactions, with 93,000 of lower confidence [36]. There is not a consensus in the literature as to what the true number of human PPIs should be, however it is safe to assume from what we know that it is many times more than what has currently been discovered. Luckily, the current amount of known PPIs is sufficient to train machine learning-based models with the objective of predicting PPIs. Such models leverage the aforementioned binary classification mathematical infrastructure and precision-recall performance metrics and are capable of strictly sequence-based classification, trained on known PPIs for a given species, and can represent very inexpensive ways of exploring entire PPI networks, or looking at "all versus all" interaction regimes for an entire organism, meaning examining

how likely every protein in a proteome is to interact with every other protein in the proteome.

These machine learning based methods can attempt to use protein domains; which are linear stretches of amino acids that appear numerous times throughout the proteome as they possess a consistent and often understood function, they may attempt to leverage particular amino acid regions and patterns which may appear arbitrary or uninterpretable and they may attempt to use previously stored or "learned" properties of amino acid residues which can certainly include solvent accessibility and physicochemical properties and previously known interactions, as well as evolutionary features and trends [37]–[40]. After using these features to attempt to understand the information or extract usable patterns from them, the machine learning classifier applies an algorithm, or mathematical approach, to build a model with the desired performance characteristics, so it may then classify future data. Popular algorithms include random forest (RF), support vector machines (SVM), as well as artificial neural networks (ANN), nearest neighbor (NN), and combination approaches referred to as ensemble learning.

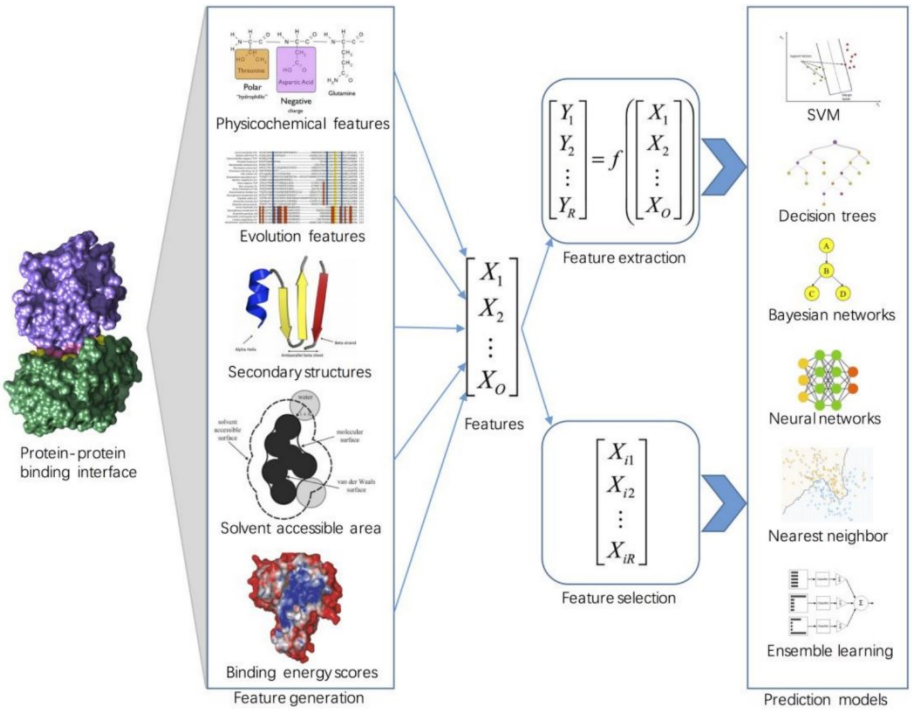


Figure 10: A summary of machine learning approaches used to predict PPIs For the binding of interface residues in protein–protein interactions, a large number and variety of features are extracted from diverse data sources. Then feature extraction and feature selection approaches are used for dimensionality reduction. Finally, the machine learning-based prediction models are trained and applied to make predictions of hot spots. (Figure reproduced from [41])

We do expect however that the PPIs that we know of, are not a perfectly proportional subset of all natural PPIs, due to technical bias in the lab techniques used to elucidate PPIs and also in terms of what proteins of interest the scientific community chooses to study and fund the discovery of PPIs of. Furthermore, precision and recall always remain tradeoffs and how much recall a research application is willing to sacrifice in order to achieve a desired precision always depend on the research application. An example set is presented at Table 1.

Method	Protein representation	Machine learning model	Performance
Park, 2009	Ensemble methods	Consensus of previous 4 methods	60% sensitivity at 90% specificity
PIPE. Pitre, 2008	Sequence windows compared using the PAM similarity matrix, leverages interaction database	Co-occurrence of window pairs in known interacting pair	14.6% sensitivity at 99.95% specificity, or 55% sensitivity at 90% specificity
Guo, 2008	Auto-covariance of physicochemical residue properties at lags of up to 30 positions	SVM	42% sensitivity, 75% specificity
Shen, 2007	Conjoint triad (frequency of occurrence of 3-tuples of residue groups)	SVM	85% sensitivity, 82% precision (81% specificity)
Martin, 2005	Signature molecular descriptor (frequency of occurrence of residue and its neighbours)	SVM	70-80% accuracy (90% sensitivity, 90% specificity)
PPI-Detect Romero-Molina, Ruiz-Blanco et al. 2019	physicochemical properties and domain-based features	SVM	~60% precision at ~80% recall
SPRINT Li & Illie, 2017	sequence-based spaced window using PAM similarity matrix	Spaced-seed hits	51.66% sensitivity at 95% specificity

Table 1: Summary of seven competitive protein-protein interaction prediction methods.

At least computational methods, if implemented correctly, do scale with available computational resources, though at the mercy of Amdahl's law, and these computational resources do not require existing experimentally determined PPI data and are decreasing

in cost as technology advances, both financial cost and in terms of energy requirement per unit of computation.

2.6.1 The Protein-Interaction Prediction Engine (PIPE)

Within the domain of predicting protein-protein interactions entirely from sequence, and via an algorithm that is entirely interpretable: enter the Protein-Protein Interaction Engine (PIPE). It is a method developed here at Carleton University by the bioinformatics research group. PIPE predicts PPIs from sequences [42], [43] and by leveraging a database of known protein interactions.

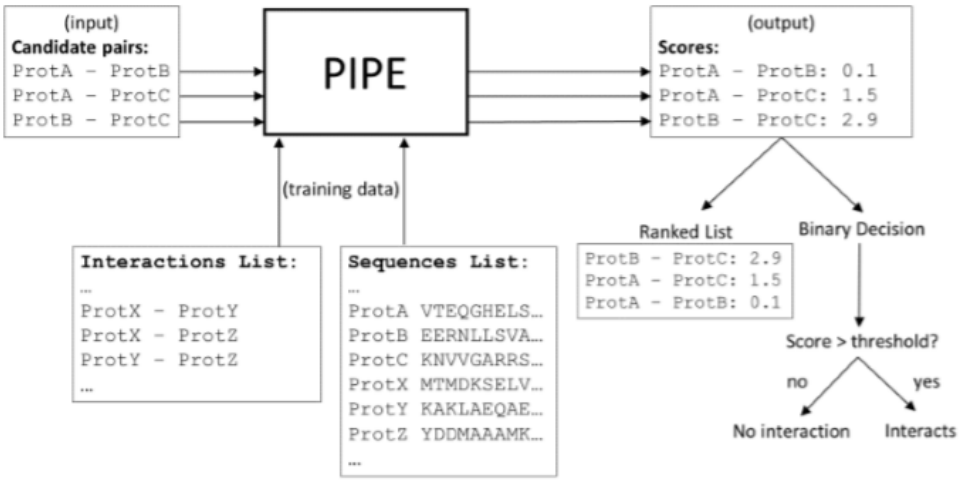


Figure 11: PIPE high-level overview PIPE computes a score for a particular protein interaction pair by leveraging a database of known interactions and the corresponding amino acid sequences for each protein. The generated score can be used to rank all computed pairs by likelihood of a positive PPI, or for a given threshold a binary decision can be made. (Figure reproduced from [44])

Essentially the algorithm works on the principle that input candidate protein pair A-B are sequence-similar to multiple protein pairs that are known to interact (e.g. X-Y, X-Z, Y-Z in Figure 11), then we are more confident that proteins A and B do indeed interact.

This approach is based on the underlying hypothesis that, across different proteins, similar local subsequences of amino acids will yield a similar 3D structure, and these similar 3D structures will interact similarly. From what has been observed experimentally this hypothesis does hold in the vast majority of cases [2].

A weakness of this approach however is the limitation to contiguous interactions. This implies a failure to account for any long-range effects, whereby dispersed amino acid regions or even individual amino acids become physically clustered following correct 3D protein folding, and these dispersed amino acids form the protein-protein interaction regime. It is difficult to estimate the percentage of PPIs which leverage these linear peptide dispersed but 3D clustered interaction approaches but it is certainly a legitimate concern that mediates some PPIs [2].

In detail, PIPE compares the query proteins A and B using a 20-amino acid sliding window, which iterates by 1 amino acid across the lengths of both proteins A and B. Each window is compared against the full known interactome database available for that species. The physicochemical similarity of one amino acid relative to another is translated by the Point Accepted Mutation (PAM) matrix. This PAM120 score matrix is a reasonable way to quantify how physiochemically similar one amino acid is from another amino acid, in the context of being a member of the aforementioned interactions database. As an example PIPE uses a PAM120 score threshold of 40 for *Homo Sapiens* and 35 for *Saccharomyces cerevisiae* [45].

Iterating through the full length of both protein A and protein B leads to the creation of a 2D matrix that is as wide as the amino acid length of protein A minus nineteen, and as tall as the amino acid length of protein B minus nineteen. Nineteen is subtracted due to the

first twenty amino acids in the window being summarized by the first element. Each element thereafter summarizes an increment in the sliding window of one amino acid. The numerical value of each element contains the total number of similar interactions found in the provided database, and this 2D matrix with scores can be represented as a 3D landscape, presented in Figure 13.

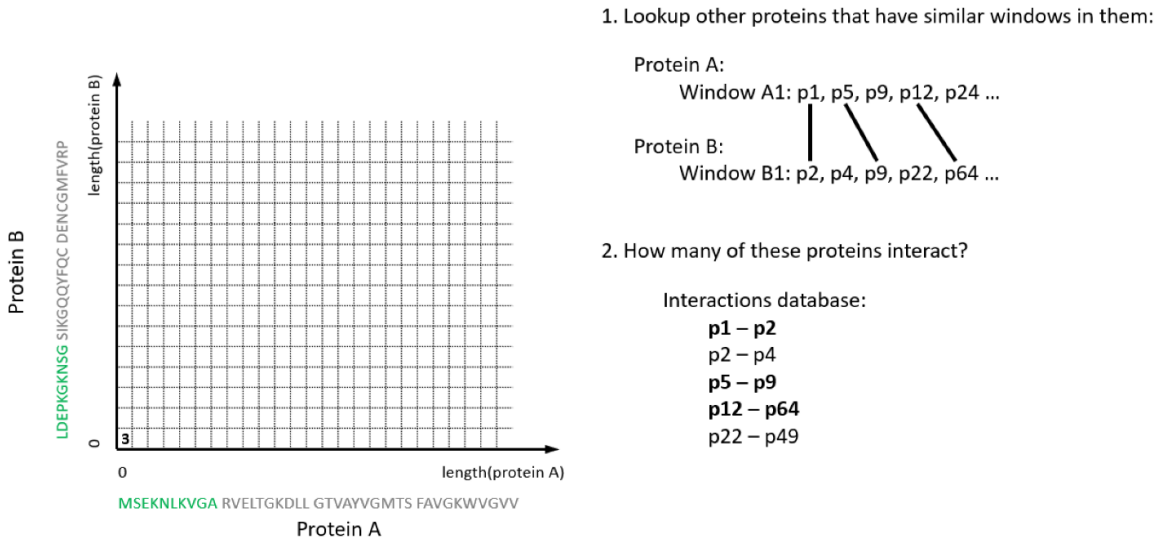


Figure 12: Overview of PIPE algorithm for one pair of sliding windows This figure examines the first sliding window of 10 amino acids in each protein, highlighted in green. In step 1, the similar proteins for the windows highlighted in green are identified. For example, Window A1 for Protein A is very similar to proteins p1, p5 and p12, which are known to interact with proteins p2, p9 and p64, respectively, which are very similar to Window B1 for Protein B. In step 2, the known interactions between the similar proteins are counted (the known interactions are highlighted in bold and drawn as black lines between proteins in the two sets). This count is entered in the interaction landscape for these windows, for this example it is three known interaction pairs. This process is then incremented by one amino acid residue and repeated for both window pairs for both protein pairs. (Figure reproduced from [44])

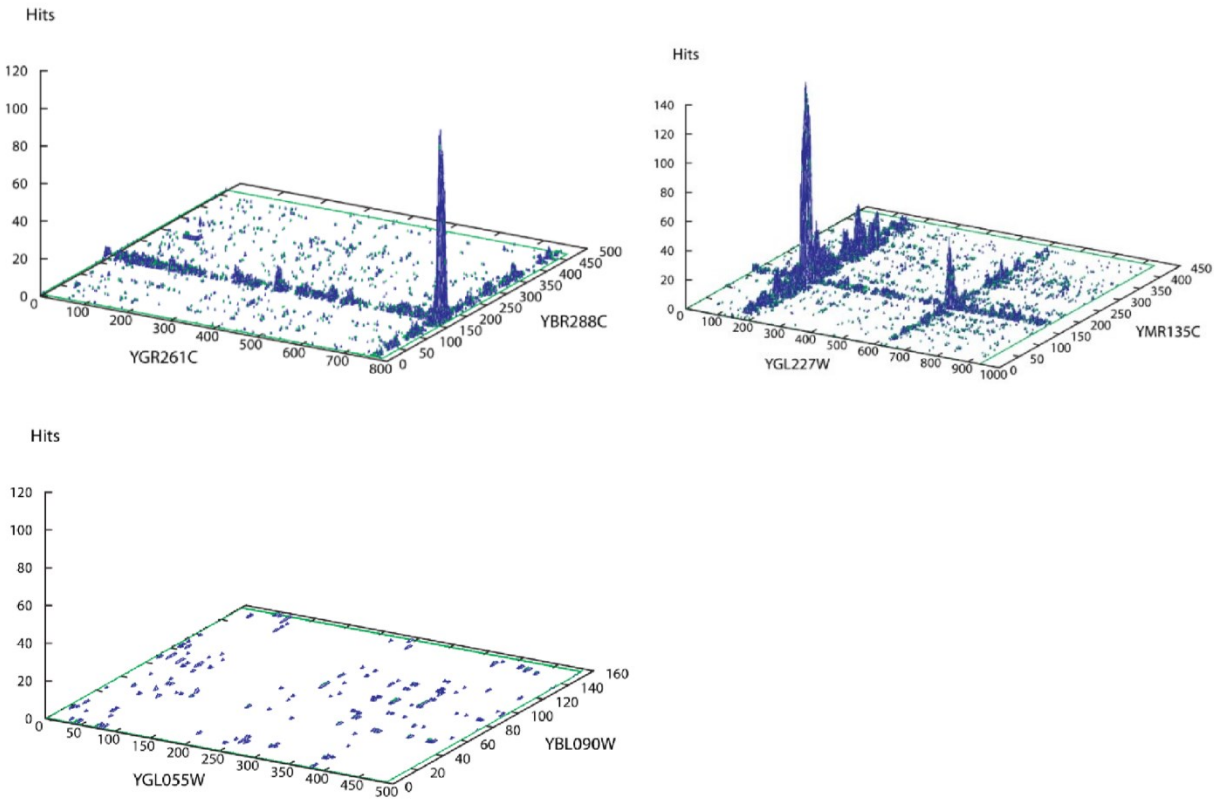


Figure 13: Sample PIPE landscapes from two positive protein pair interactions (top row) and a sample of a negative interaction pair (bottom row). (reproduced from [42])

This 3D landscape does need to be summarized in some way as all PPI scores need to be ranked and sorted as per a binary classification problem with a hierarchy of scores. This is performed by simply adding up all of the values in all of the elements and then normalizing by the size of the landscape. This normalization is to account for the fact that simply larger proteins will have more opportunities to accrue a higher score, and in reality, proteins of all sizes do form positive PPIs. A more recent innovation in the scoring was to normalize the scoring for each individual element, and this is done by dividing by the total number of interaction pairs by how common those pairs are in the database. This is referred

to as the sim-weighted score and it remains the current used to generate a single PIPE score for a given pair of protein-protein interaction partners.

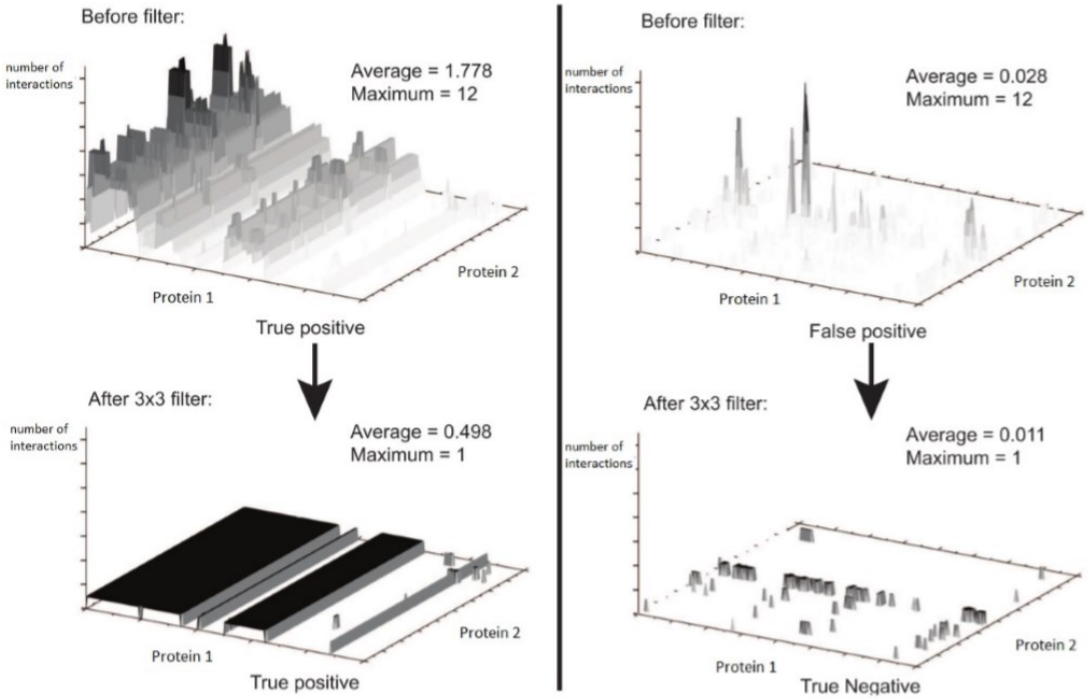
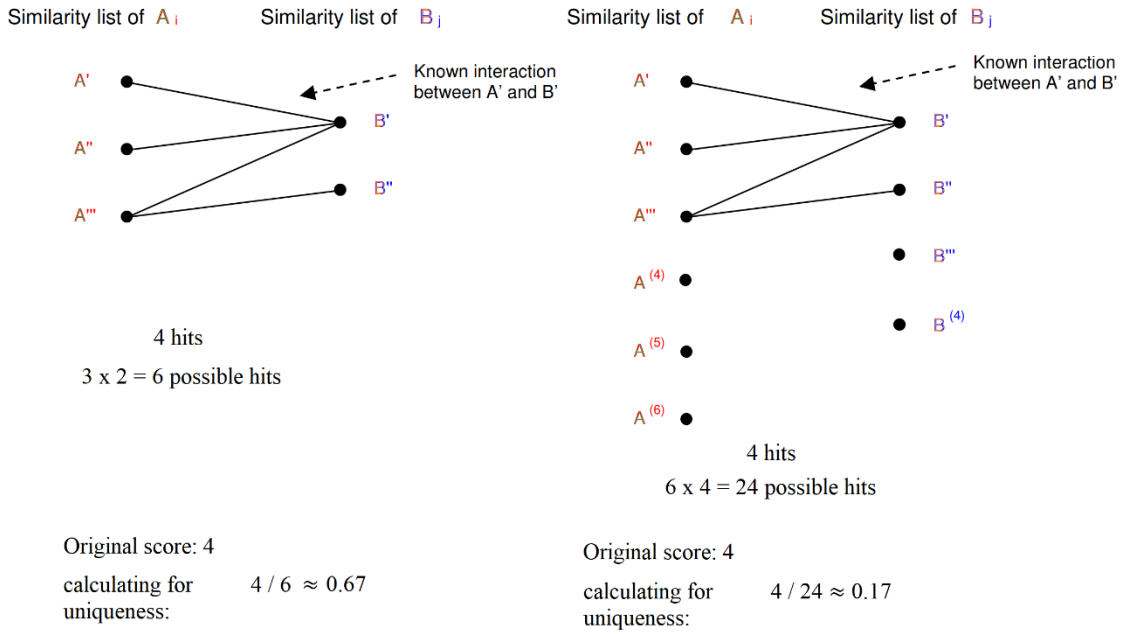


Figure 14: Traditional PIPE score using filtering of landscape. The original landscapes are shown in the top row, and both are classified as positives. The bottom row shows the result of applying filter to each of the original landscapes. Now, the right landscape is correctly labelled as a negative (adapted from [44])



(a) (b)

Figure 15: Example of the innovation of introducing a sim-weighted score (a) an original score of 4, due to 4 hits (represented by the black lines) is divided by 6 (the product of 3 similar A proteins potentially interacting with 2 similar B proteins) for a final score of 0.50. (b) another example where again 4 hits are divided by 24 (product of 6 similar A proteins and 4 similar B proteins. (Figure reproduced from [17])

Figure 16 describes the algorithm for computing the similarity-weighted (sim-weighted) PIPE score utilized in this research and also augmented by way of physicochemical properties from a given 2D matrix, whereby each element contains the value $NumberOfKnownPairs[w_A, w_B]$ representing the score of how many total hits w_A had with w_B in the interaction database by PAM120 matrix similarity. Additionally, the pre-computed window frequency for the relevant 20 amino acid window within Protein A designated by $Freq\_w_A$ and the relevant 20 amino acid window within protein B designated by $Freq\_w_B$ are required to compute the sim-weighted PIPE score.

1	For each window offset in Protein A, w_A
2	For each window offset in Protein B, w_B
3	$PIPE\ Score\ \mathrel{+}= 1000 \times \frac{\boldsymbol{NumberOfKnownPairs[w_A, w_B]}}{\boldsymbol{(Freq\_w_A \times Freq\_w_B)}} \times \boldsymbol{\gamma}$
1	$Final\ sim\ weighted\ PIPE\ Score = \frac{PIPE\ score}{\boldsymbol{Length_{Protein\ A} \times Length_{Protein\ B}}}$

Figure 16: Computing the sim-weighted PIPE score from a 2D matrix and window frequency in the interaction database. The blue section represents the conventional landscape score, the green section represents the sim-weighting, the orange section represents the final protein length normalization, and the red gamma term represents the physicochemical contribution of this work.

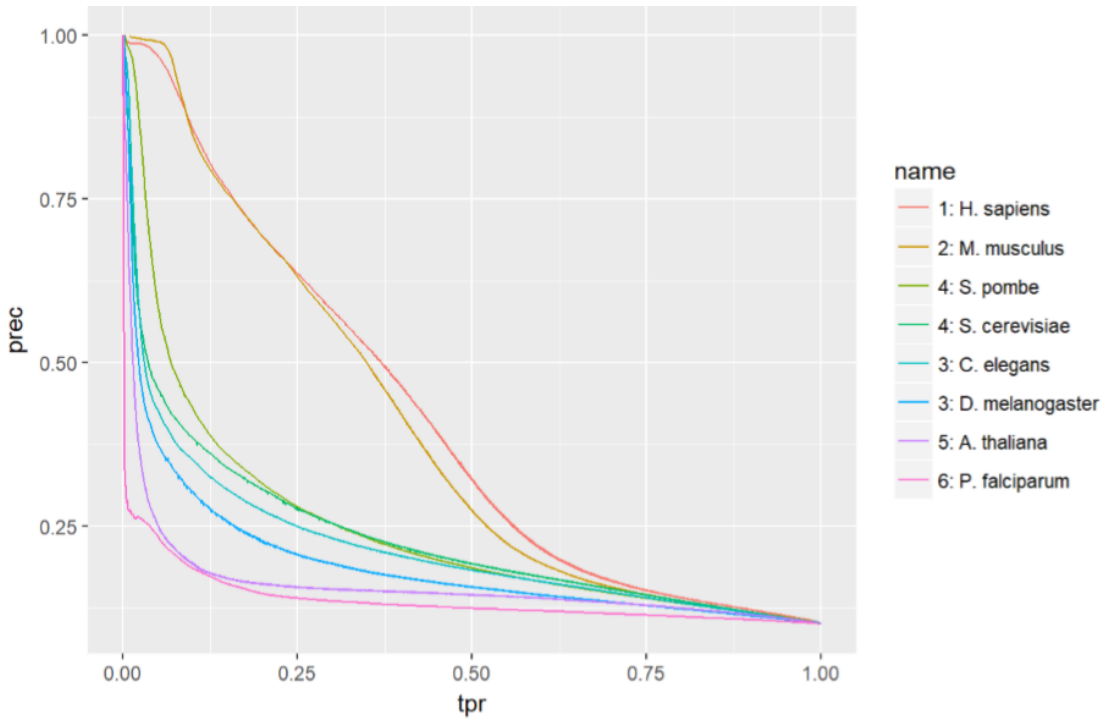


Figure 17: Precision-Recall curve for various species used to predict the known *Homo Sapiens* interactome. Performance decreases the further away the interaction database is evolutionarily. The data for each training species comprised 2000 randomly sampled interactions to control for interactome size. (Figure reproduced from [44])

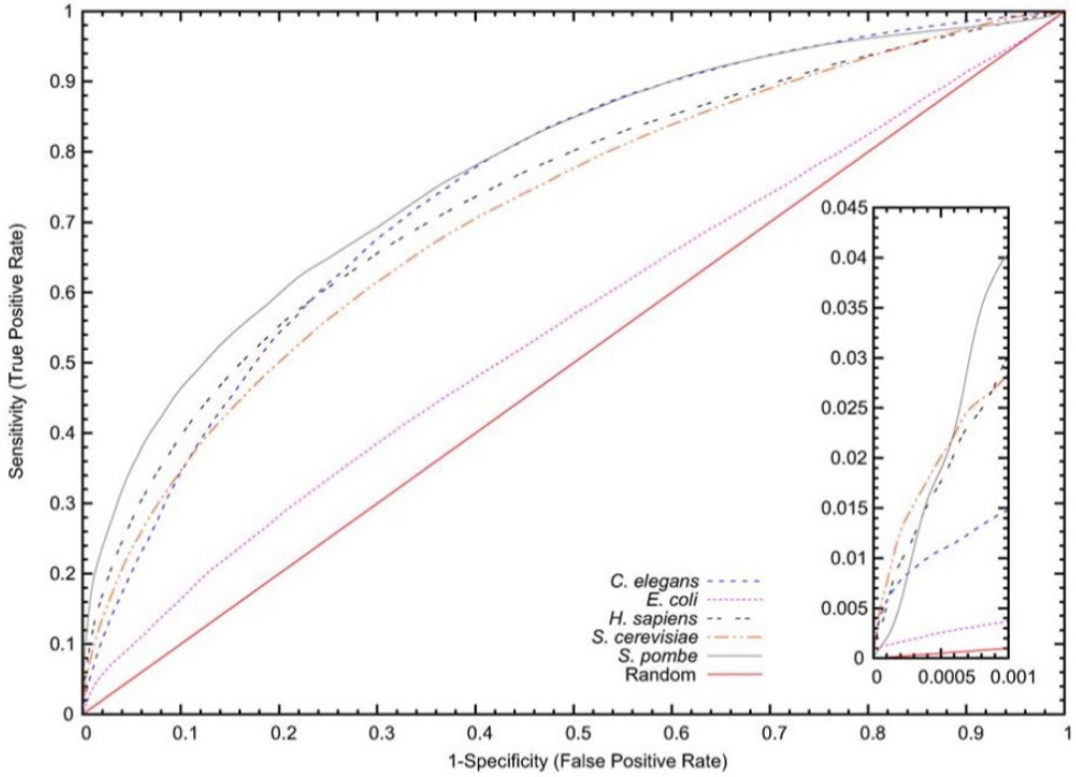


Figure 18: Sample ROC curve from PIPE with different species. The box in the bottom right shows the area the threshold is generally set in (Figure reproduced from [44])

2.6.2 Other Recent Sequence-based PPI Prediction Methods

This section reviews two other recent sequence-based PPI prediction methods: SPRINT [46] and PPI-Detect [47]. These were chosen here, since they are most relevant to the current thesis. SPRINT is conceptually similar to PIPE with slightly improved performance. Due to its conceptual similarity, the contributions made in this thesis that improve PIPE may well be applicable to SPRINT. PPI-Detect was chosen because it also leverages physicochemical profiles computed via ProtDCal [47].

SPRINT (Scoring Protein INTeractions) is another method of assessing PPIs by sequence, it detects similar subsequences in proteins however it uses "spaced seeds" to determine sequence similarity *in lieu* of 20 amino acid sliding windows. These "spaced x" are shown in Figure 19. Only the particular amino acids at designated locations contribute to scoring, indicated by the non-*positions within the spaced seed. As with PIPE the PAM120 is used to deem sufficient amino acid similarity, with the PAM120 scores based around physicochemical properties. If the amino acids at the spaced seed points are sufficiently similar, then the regions encompassing the space seeds are deemed similar. After similar regions between all protein pairs have been identified, SPRINT generates a score for a putative interaction by looping through all known interactions and increasing the score of proteins that possess similar areas to the interacting pair. As with PIPE and binary classification in general: after the score has been generated, all pairs that have a score higher than a tunable decision threshold are predicted to positively interact [46]. SPRINT did compare itself to an older version of PIPE which was implemented by the SPRINT authors, and within the context of their work and their dataset SPRINT was both

more accurate than PIPE and also showed itself to be the fastest method among those being compared.

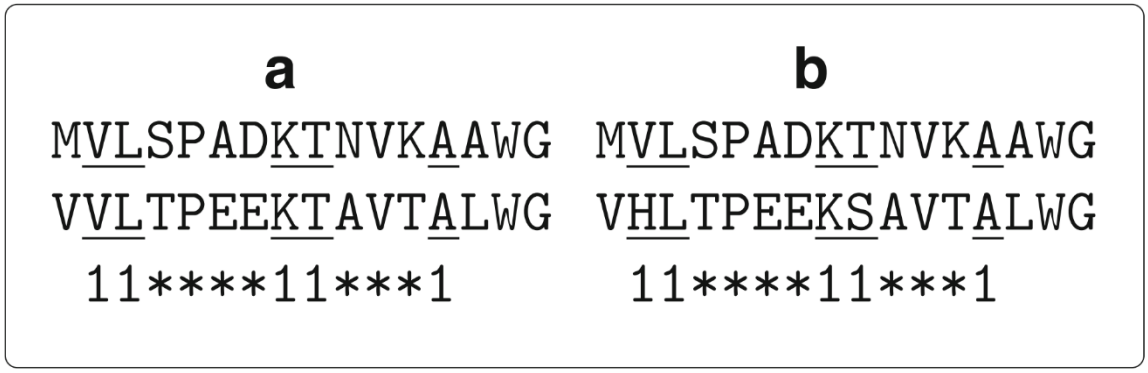


Figure 19: Spaced-seed hits. (a) portrays an exact hit, and (b) represents an approximate hit of the same spaced seed. (Figure reproduced from [46])

PPI-Detect is another competing PPI detection platform. It utilizes the ProtDCal platform to generate a large gamut of physicochemical features for the proteins of interest, and then uses a support vector machine (SVM) classifier to synthesize the data and make a scoring decision. PPI-Detect is trained and operates based on protein domains, which is not directly comparable to PIPE, which operates on a 20 amino acid sliding window. In its current form, PPI-Detect does take longer to run than PIPE as first physicochemical features must be generated, and then an SVM classifier must be run.

3 Integrating Protein Physicochemical Properties in PIPE

This chapter presents the four main contributions of the thesis. Section 3.1 describes the documentation created for PIPE. Section 3.2 describes PyPE, which was software developed to implement and test physicochemical properties added to improve PIPE's predictive performance.

3.1 Documenting PIPE

PIPE has been developed by several past graduate students and lacks a coherent workflow or any documentation. It requires the user to perform numerous steps and Linux command line functions in various directories, with little clear order or directive. While there is a provided script that should, in theory, automate the process and run PIPE in one command, the script is highly fragile in terms of file/directory names, user configurable parameters, software versions, dataset parameters, and network locations and therefore will invariably break in a multitude of steps and locations throughout the entire workflow, simultaneously. Coupled with poor error reporting, whereby the error reported does not reflect the cause of the error, makes debugging the script more daunting than simply attempting to run PIPE unassisted. With this being the case, PIPE requires several months of trial and error to begin to use, with some risk along the way for bugs and user mistakes that can lead to false results or bad PPI predictions.

This was set out to be rectified, as this learning curve could be reduced to merely several days should comprehensive documentation be available. This was partially demonstrated by the documentation being of great help to a subsequent graduate student. Thus writing such documentation was the first contribution of this work. This 8 page

document included in Appendix A details all the steps PIPE is required to progress through, including: how to format and input the known protein pairs and desired PPIs to be assessed, how to run the intermediate compute step known as GenTab and finally run PIPE, what parameters the user should configure at each step for their particular use case and also particular dataset, what Linux terminal commands must be inputted at each stage along with provided examples, and finally, common error outputs that can occur throughout the process and how to best address them.

3.2 PIPE Implementation in Python (PyPE)

To facilitate the prototyping of new solvent accessibility and physicochemical methods which can modify the 3D PIPE landscape, as well as the ability to import new data and new features, the latter stages of PIPE were re-implemented in Python with additional functionality and named PyPE.

Operationally, PIPE is still configured with a desired set of protein pairs, and is provided with a known interactions database. PIPE will then calculate how frequently each 20 amino acid sliding window will appear in the interaction database, as per the PAM120 matrix determining whether the sliding window sequence is similar enough to the interaction database. PIPE will then calculate the 2D scoring matrix according to how many interactions are found in the database for each 20 amino acid element. The number of database entries for each 20 amino acid window are stored in a computed database referred to as GenTab. At this point, both the matrix files and the GenTab files can be inputted into PyPE, along with any desired solvent accessibility or associated files, and PyPE will assemble these disparate data structures and output a final default sim-weighted score as well as any modulated sim-weighted scores. PyPE can then generate precision – recall

curves and assess whether the performance of these modulations has been an improvement over the default sim-weighted score. This workflow is summarized in Figure 21. PyPE also has the following features most of which are not found in PIPE and rarely, if ever, found in other PPI or physicochemical software packages:

- Detection and graceful error handing of bad, corrupt, or misformatted input files. This implies should a matrix file, GenTab file, solvent accessibility or companion file be missing, corrupted, or misformatted; PyPE will detect the misformat or error and simply alert the user of the name of the offending file, skip over that PPI pair prediction and proceed on to the next one, without crashing or halting.
- Progress reporting with timestamps. For every 1% of overall progress made in the PPI workload, PyPE will report the amount of progress made as well as the system time when this occurs, allowing the user to accurately predict the total runtime after only a few percent of the overall progress is completed, due to the homogenous nature of the PyPE workload.
- Built-in continual saving of the computed result file. As PyPE is running, results are dynamically saved to file as they are being computed, allowing for the user to continue their computational progress from where they left off in the event of an interruption or termination.
- The ability for hypotheses to be run in parallel. PyPE allows for hundreds of different hypotheses / hyperparameter experiments to be run simultaneously, this is outputted in a comma separated values (csv) file whereby each column represents one protein-protein interaction pair, and there can be hundreds of associated rows, each of which

represents a different score modulation approach. Multiple instances of PyPE can then be run in parallel (recommended one instance per CPU thread) to speed computational time.

- Included precision-recall curve generation powered by the Matplotlib library, as well as calculation of the resulting recall for any fixed level of precision, performed for every hypothesis attempted and sorted from best result to worst result.

Figure 20: PyPE logo as well as an example of PyPE being run Progress reporting, system time outputting and graceful handling of a corrupted matrix file are presented.

3.3 Integrating Solvent Accessibility into PyPE

Upon successful documentation of PIPE and creation of PyPE, the step was to introduce a method into PyPE to be able to include novel data to the PIPE scoring method so as to improve its predictive performance. As previously discussed, introducing the physicochemical property of solvent accessibility is a novel approach of improving the predictive power of PIPE. This is because PIPE currently uses no direct structural or physicochemical information in its predictions. Once we have a successful implementation

of solvent accessibility to modulate the sim-weighted PIPE score, it is also required we have a way to test these modulations and determine a sensible framework to use and see how it performs relative to default PIPE scores. The workflow for this is summarized in Figure 21.

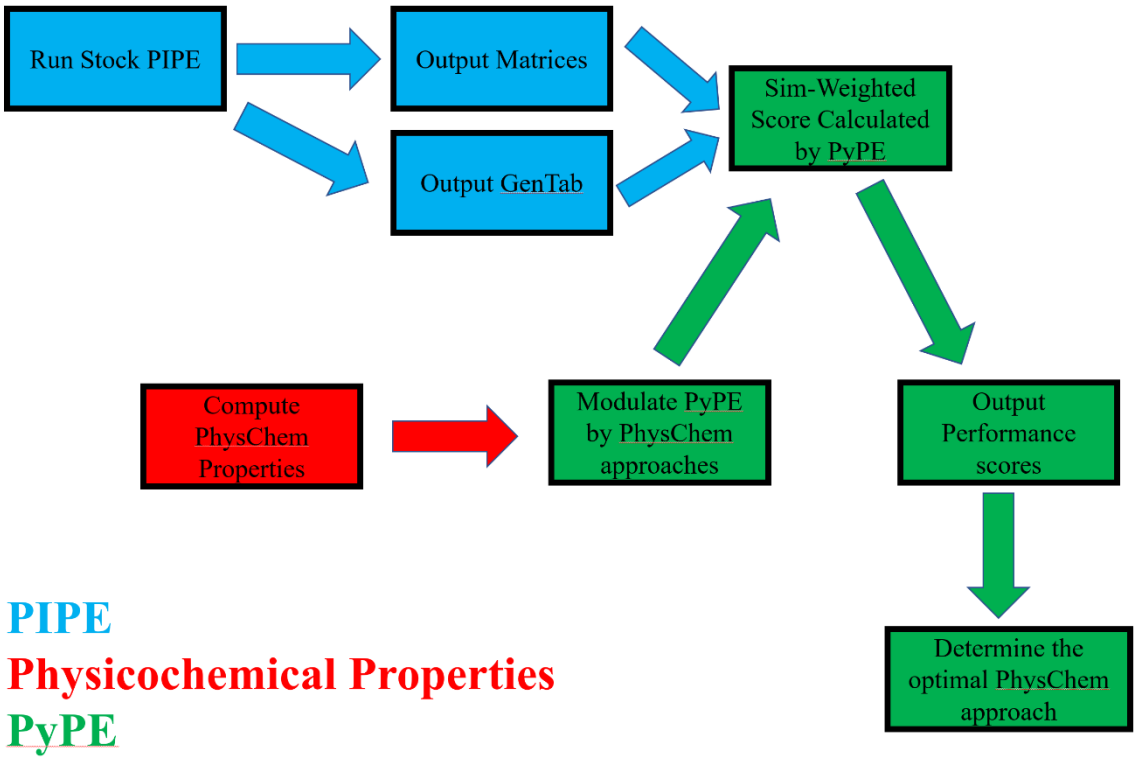


Figure 21: Flow diagram of the innovation introduced by this work Conventional PIPE is used to generate precursor files, while new approaches are used to calculate solvent accessibility and other physicochemical properties. These new approaches are implemented to modulate the 3D PIPE landscape and create new scores, which are then outputted and tested for the best approach.

A well-known state-of-the-art method to calculate solvent accessibility entirely from protein sequence is NetsurfP [14]. It is an ensemble method of multiple artificial

neural networks (ANNs) that has been trained on experimentally solved protein structures to generate scores for the relative and absolute solvent accessibility of amino acid sequences. As such, it can be seen as one of the approaches summarized in Figure 22, and it is arguably the most promising approach and method to improve PIPE.

As mentioned, NetsurfP is an ensemble of neural networks, the primary network is trained on sequence profiles and predicting secondary structure (α-helicies and β-sheets) and has two primary outputs simply designating buried (B) or exposed (E) residues. This is treated as a binary classification problem, whereby amino acids in the interior of the protein are deemed buried, and those at the surface are deemed to be exposed. This higher level output sets up a predicted category, and these outputs are used together with sequence profiles in the form of position-specific scoring matrices (PSSM) to train the secondary neural networks. This secondary neural network was trained using a window size of 11 amino acids and between 25 and 200 hidden neurons. The training set was the Cull-1764 dataset which featured 1764 distinct proteins comprised of 417,978 amino acids in total. The accuracy was found to be approximately 79%. This implies 79% of the time when the prediction for a single amino acid residue was to be exposed or buried, the prediction was correct. It should be noted the two classes of either buried or exposed amino acids were quite balanced.

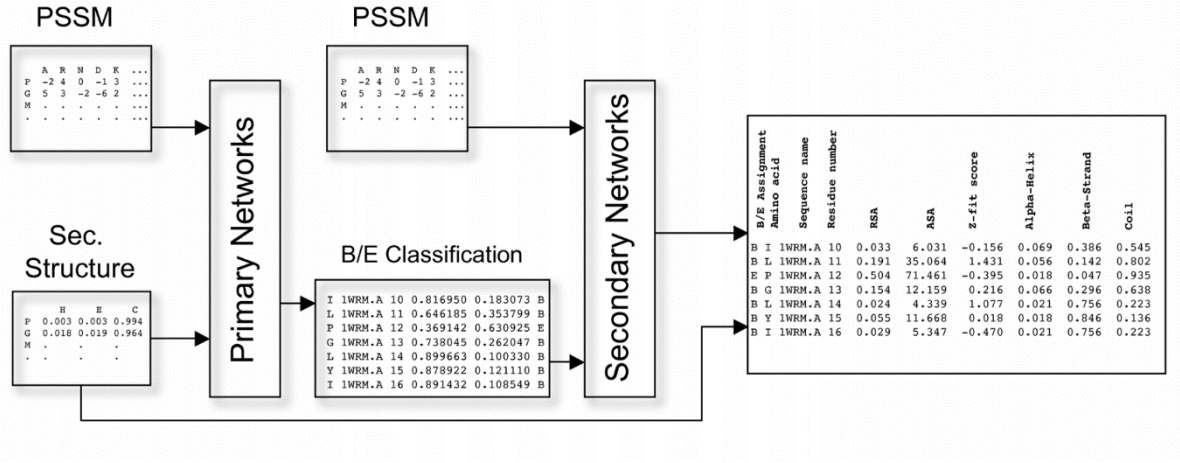


Figure 22: Schematic representation of the workflow of NetsurfP Sequences are initially passed through a neural network along with position-specific scoring matrices to be classified as either buried (B) or exposed (E). This output is then fed into a secondary neural network along with the same position-specific scoring matrices to generate both relative and absolute solvent accessibility scores for every amino acid. (Figure reproduced from [14])

The latest distribution of NetsurfP (v 1.1), which uses the PSSM of the human proteome from NCBI and is provided in a Linux x86 compatible distribution which unfortunately only parallelizes up to 5 CPU threads, regardless of how many more CPU threads are available. As such, applying NetSurfP to the entire human proteome of 21,225 proteins took 16 days to complete.

3.3.1 Two Hundred Integrations of Solvent Accessibility within PIPE

NetSurfP produces two scores for each amino acid: the absolute and relative solvent accessibility (ASA and RSA). We seek to use one or both of these scores to modulate the PIPE landscape. Therefore, we have two SA scores for each amino acid in two different protein subsequences (one from each query protein in the input pair). A number of different

strategies for combining the SA scores to effectively modulate the PIPE landscape were examined.

A brute force approach was used to explore different strategies for combining SA data with PIPE landscapes, leveraging the parallelizability of PyPE. This means both relative and absolute solvent accessibility would be utilized independently, then each approach would have five separate ways to modify these solvent accessibility values. These approaches are shown in Figure 23 step 2 and include: leaving the values as raw, squaring them, cubing them, taking the square root and taking the cube root. These modifications have the effect of either magnifying the impact of the most solvent accessible amino acids on the ultimate PIPE score by squaring or cubing all values, or decreasing the impact of the most solvent accessible amino acids by square rooting or cube rooting all values. Next, the scores for each individual amino acid must be aggregated into the format of the final PIPE 3D landscape, which exclusively deals with 20 amino acid sliding windows. Therefore the solvent accessibility score for 20 amino acids were reduced down into one value by four different methods, these were taking a mean, performing a root mean square (RMS), a mean square (MS) and a mean root (MR). These approaches are summarized as step 3 of Figure 23, and also shown below, where n = 20 as we are always reducing 20 amino acid scores to one score.

$$Mean = \frac{1}{n}\sum_{i=1}^{n} a_i = \frac{a_1 + a_2 + a_3 \ldots . a_n}{n} \quad (7)$$

$$Root\ Mean\ Square\ (RMS) = \sqrt{\frac{\sum_{i=1}^{n} x_i^2}{n}} = \sqrt{\frac{x_1^2 + x_2^2 + x_3^2 + \ldots . . + x_n^2}{n}} \quad (8)$$

$$Mean\ Square\ (MS) = \frac{1}{n}\sum_{i=1}^{n} a_i^2 = \frac{a_1^2 + a_2^2 + a_3^2 \ldots . a_n^2}{n} \quad (9)$$

$$Mean\ Root\ (MR) = \frac{1}{n}\sum_{i=1}^{n} \sqrt{a_i} = \frac{\sqrt{a_1} + \sqrt{a_2} + \sqrt{a_3} \ldots . \sqrt{a_n}}{n} \quad (10)$$

This will result in a 2D matrix of numbers compatible with the 2D matrix of PIPE, however the values will greatly differ between relative and absolute solvent accessibility, and thus need to be normalized to the range of 0 to 1. At this point we have two values that can be used to modify the PIPE landscape at any one particular element, the first value is from the first protein, and the second value is from the second protein. There are numerous ways to assemble these two values and five separate strategies were attempted and described in step 4 of Figure 23. These include: taking the product of the two values, only taking the larger of the two values, only taking the smaller of the two values, dividing the larger of the two values by the smaller of the two values, and taking the mean of the two values. Combinatorically, this results in two hundred separate ways to introduce these NetSurfP solvent accessibility values into PIPE, and all were attempted in PyPE serially.

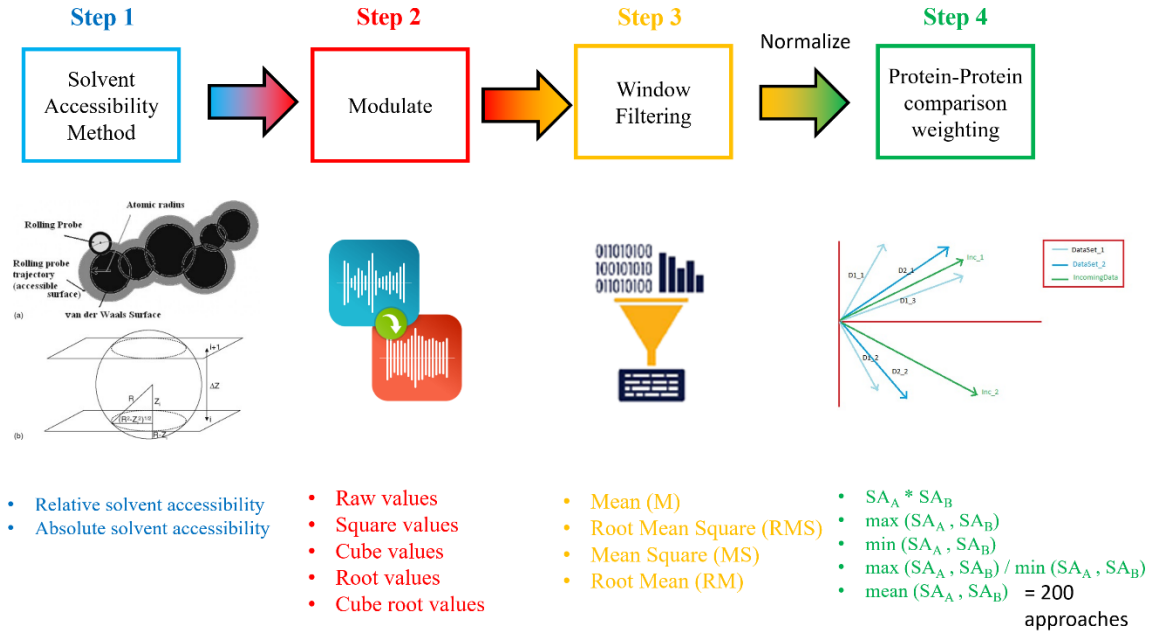


Figure 23: Flow chart representing the combinatorics involved in implementing solvent accessibility into PIPE The first step is encompasses a solvent accessibility strategy. The second step modifies all the values to up or downscale them. The third step aggregates 20 amino acid values down to 1 value by four different strategies. The values are then normalized and finally five different approaches are used to combine the two values from the two interacting proteins.

3.3.2 PIPE + NetSurfP Results

To evaluate the performance of the various solvent accessibility approaches, a dataset of 1000 "ground truth" and 1000 "negative pairs" PPI interaction landscapes were fed into PyPE, representing the positives and negatives, respectively. These 1000 ground truth PPIs were pairs known to interact with high confidence, meaning multiple

experimental techniques confirmed the interaction. These data were taken from the BioGRID server which houses such protein experimental and interaction data [48]. The remaining approximately 47,000 high confidence interactions provided by BioGRID were used as the known interactions database to use with PIPE, these interactions were curated by the methods of the positome server [49]. Conversely, the "negative pairs" were randomly generated PPI pairs that are highly unlikely to interact, given the aforementioned 1:100 class imbalance between positives and negatives with regard to PPIs. 1000 positive and 1000 negative pairs were chosen as a reasonable sample size with which to test approaches, while limiting computational time and data complexity to reasonable levels within PyPE, as each protein pair would be scored 200 times.

All PIPE landscapes were generated by running PIPE in leave-one-out (LOO) mode, whereby when a "ground truth" protein pair was being scored, the interaction database did not have access to the entry for that particular protein pair. This is important as otherwise the entire landscape would receive an undeserved +1 score to every element, since each 20AA sequence windows would match perfectly with the known PPI in the database!

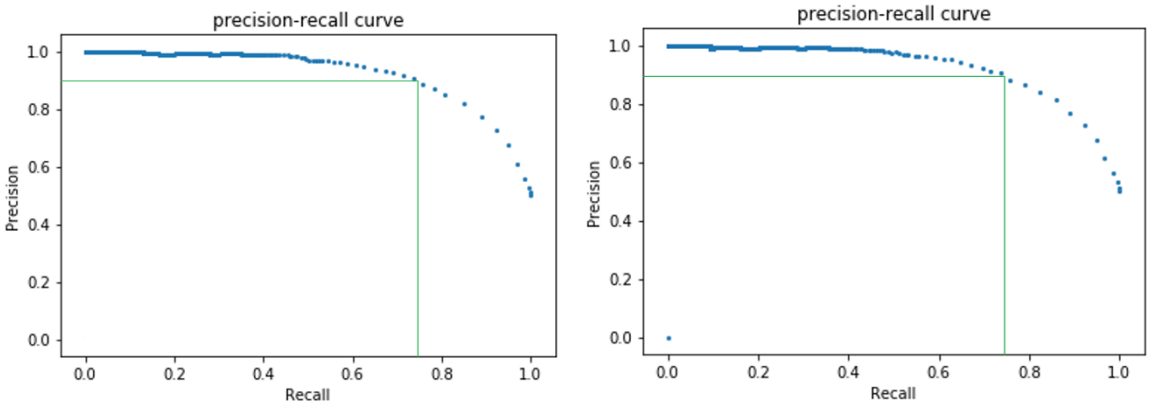
These 1000 positive and 1000 negative landscapes generated by PIPE were then modulated 200 separate ways within PyPE, along with the default sim-weighted scoring prediction. To assess the default PIPE sim-weighted scoring performance, a precision-recall curve was generated, shown in Figure 24a. In order to assess the performance of these 200 approaches in an objective way, as comparing 200 different precision-recall curves can be difficult and subjective, the decision was made instead to fix the precision at 0.9 and then strive to attain as much recall as possible. This was implemented by ordering

the 2000 scores (i.e. 1000 positive and negative protein pair scores) from lowest to highest, and a moving score threshold was set at the bottom of the list and it would move its way up the scores. At each step the precision was calculated, and when it finally achieved >=0.9, meaning there were at least 9 true positives above the threshold for every false positive; then the recall was calculated. The recall is simply a ratio of how many true positives were above this score threshold out of the total positives, which was 1000.

The results were rather modest, as the absolute best of the two hundred methods only increased the recall from 73.8% up to 74.7%, representing a 0.9% improvement. In other words, of the 1000 positive PPIs assessed; 9 were improved from false negatives to true positives, while maintaining the same 9:1 ratio of true positives to false positives. Referring to the steps illustrated in Figure 23: the most successful method utilized relative solvent accessibility for step 1, the raw values for step 2, the mean for step 3, and the "max ÷ min" approach for step 4. This points to a few trends among the best and worst performing approaches, the first being relative solvent accessibility appears more valuable than absolute solvent accessibility, the second is that mathematical operations of rooting and squaring values, as well as rooting or squaring during the window filtering all detract from the classification performance. Additionally, that looking for differences in solvent accessibility, which is represented in the "max ÷ min" approach, is a strong strategy.

The worst performing methods having a recall of below 50%, featured cubing and squaring the values, allowing the top scores for individual amino acids to dominate the scoring for the 20 amino acid window and thus likely removing significant nuance and context. Furthermore, multiple different scaling factors were tested, from 1x all the way to 32x. This scaling factor gives a geometric weighting to the contribution of the landscape

modulation from NetsurfP. Interestingly, 1x all the way to 8x scaling had the same improvement to recall, and beyond 8x, the performance fell drastically to far worse than default scoring performance.



(a) (b)

Figure 24: Precision-recall curves for default PIPE and best SA-augmented PIPE. (a) represents the default precision-recall curve for a balanced class of 1000 positive and 1000 negative PPIs, with a 73.8% recall at a precision of 90%. (b) represents the best result from solvent accessibility, which is a 74.7% recall at a precision of 90%.

3.4 Integrating ProtDCal Protein Descriptors into PyPE

Implementing the physicochemical property of solvent accessibility as inferenced by NetSurfP into PyPE yielded modest success. To attempt to improve on this, a more broad approach to implement physicochemical properties was attempted.For the purposes of this research, ProtDCal was run on the entire human proteome from NCBI, which featured approximately 21,000 proteins. Every ProtDCal descriptor compute option from the first step to the third step was enabled resulting in 166 features, all of which were tested in PyPE. The fourth ProtDCal step was not used, as these aggregation operators and strategies are incompatible with how PIPE is scored and the landscape that is generated. To explain,

PIPE requires an aggregation that is fixed to a 20 amino acid sliding window to be compatible with modulating a 3D landscape. The work from implementing NetsurfP showed that the most successful implementation from the two hundred methods was simply taking the mean of 20 scores as the aggregator, as initially modulating the individual amino acid scores with squaring or rooting placed too much or not enough emphasis on the contributions of all amino acids. Thus, step 2 of Figure 23 was to take the raw values, and step 3 of Figure 23 again gave the best result when taking the simple arithmetic mean. Step 4, which includes the ultimate implementation of the final scores for each of the two proteins, is much more subjective, and therefore many strategies were tested. This includes all of the strategies shown in Step 4 of Figure 23, as well as their negatives and inverses. Thus for example if calculating propensity to engage in turn conformations, which is perhaps a feature that would be detrimental to engaging in PPIs, a low score as determined by either (1 – turn propensity) or (1 ÷ turn propensity) may be a fruitful and positive landscape modulations.

3.4.1 ProtDCal Results

Once again, we have used precision-recall curves with a fixed precision at 90%, and determine our improvement in improving recall. All 166 aforementioned features were tested with a 2x, 4x and 8x multiplication factor, it was once again interesting that these scaling factors had no effect, with only a precipitous drop-off in performance after 8x scaling. Furthermore (1 – feature) and (1 ÷ feature) were tested. Once again, the same 1000 ground truth and 1000 random pairs were tested as with NetsurfP. This time there was a 2% improvement in recall, which is not trivial and represents an improvement of 20 false negatives being turned into true positives among the 1000 positive test pairs. As a recall of

73.8% implies that of the 1000 positives, 738 are true positives and 262 are false negatives, thus improving 20 out of 262 false negatives into true positives could be seen as a 7.6% reduction in the number of false negatives. All of the highest scoring physicochemical descriptors are summarized in Table 2. The algorithm used for all of the top scoring properties was:

$$sim\ weighted\ score\ \times (\ 4\ \times\ (1 - Score_{ProteinA})\ \times\ (1 - Score_{ProteinB})) \qquad (11)$$

Table 2: Summary of the highest scoring physicochemical properties.

Recall Score	Feature Name	Feature Description
73.8%	Default PIPE	No ProtDCal modifications
75.8%	SIEPIA_NO	Overall surface electrostatic potential distribution
75.3%	Fuk7_NO	Fukui Radical Reactivity Index. Spatial distribution of radical reactivity
75.2%	SIGA7_NO	Electronic kinetic energy density on the molecular surface
74.9%	EP10_NO	Bin of electrostatic potential descriptors
74.9%	SIEPA10_NO	Overall surface electrostatic potential distribution
74.9%	SIEPA2_NO	Overall surface electrostatic potential distribution
74.9%	Ap_NO	Molecular area of non-carbon atoms in the sidechain
74.8%	SIEP_NO	Overall surface electrostatic potential distribution
74.7%	EP1_NO	Electrostatic potential descriptor bin

With the most successful score coming from the property known within ProtDCal as SIEPIA_NO and representing overall surface electrostatic potential distribution. The success of normalizing such a score, and then taking the negative in order to augment the sim-weighted score at each element, implies that non-polarity is a valuable property of amino acids that engage in protein-protein interactions, and presumably are on the surface.

Numerous manual attempts were made to implement the top 3 and the top 5 most beneficial physicochemical properties in unison; however, none of these was more successful than using the single best ProtDCal-generated feature. One possible explanation is that incorporating numerous properties will cause them to interact in uncoordinated ways that reduce the value they have individually. Examples of attempts to incorporate multiple properties are as follows:

$$sim\ weighted\ score\ \times (1 - Property1)\ \times\ (1 - Property2)\ \times \ldots (1 - Property5) \quad (12)$$

$$sim\ weighted\ score\ \times \big((1 - Property1) + (1 - Property2) + \ldots (1 - Property5)\big) \quad (13)$$

$$sim\ weighted\ score\ \times max(1 - Property1), (1 - Property2), \ldots (1 - Property5) \quad (14)$$

$$sim\ weighted\ score\ \times min((1 - Property1), (1 - Property2), \ldots (1 - Property5) \quad (15)$$

Lastly, to validate this 2% improvement of 73.8% recall to 75.8% on 1000 ground truth and 1000 random pairs by implementing the feature SIEPIA_NO. This modest test set can be seen in Figure 25. A larger test was performed on 10,000 ground truth and 10,000 random pairs and this can be seen in Figure 26. In this larger test set the improvement was 1.9% from 72.1% to 74.0%. In terms of statistical analysis, this improvement has a p-value

of approximately 0, indicating that there is effectively no probability such a performance improvement was entirely due to chance with such a large sample size.

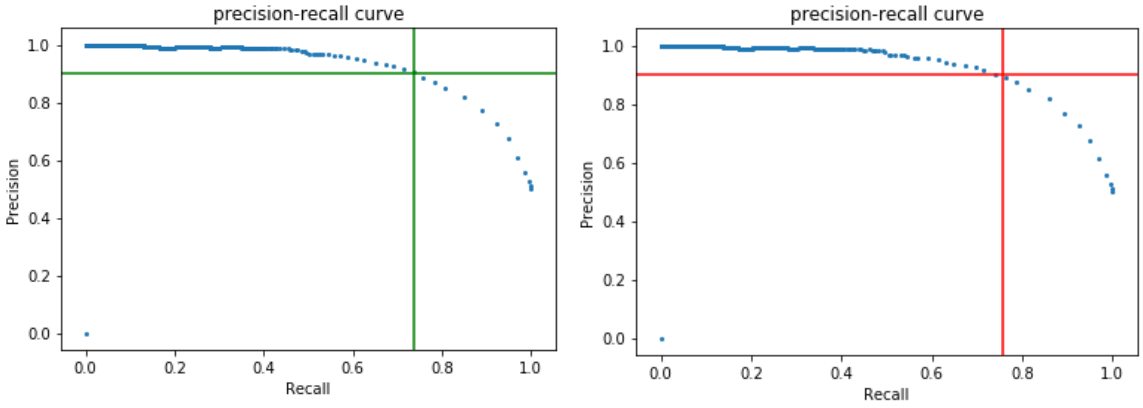


Figure 25: Precision-recall curves for default PIPE and the best Physchem Improvement. (a) the green left figure represents the default precision-recall curve for a balanced class of 1000 positive and 1000 negative PPIs, with a 73.8% recall at a precision of 90%. (b) the red right figure represents the best result from the charge distribution phys chem property, which is a 75.8% recall at a precision of 90%.

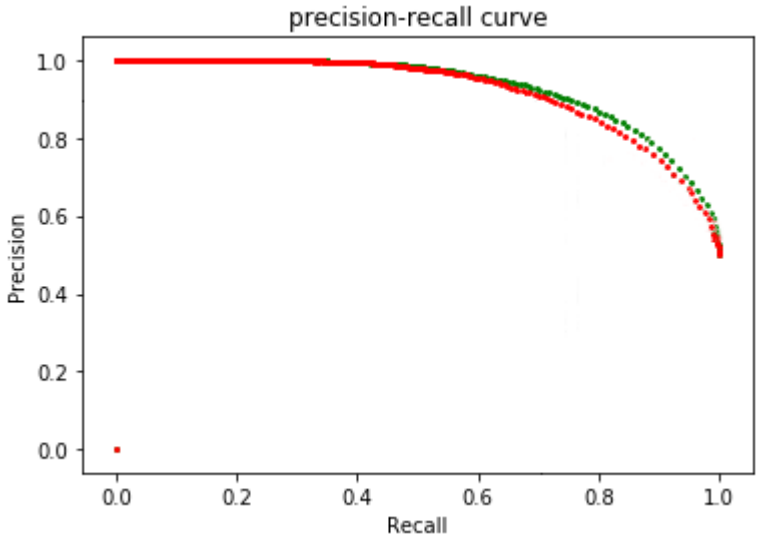


Figure 26: Precision-recall curve at 10,000 samples for default PIPE and the best Physchem Improvement. This figure shows the conventional PIPE curve in red, and the improved curve in green for the final 10,000 ground truth and 10,000 random pairs.

Figure 27 shows another way to visualize the improvement to the population of recall scores, with the new scores in red and reaching higher levels of recall at 90% precision with a greater population. Figure 28 show a histogram of PIPE scores for negative (random) pairs, whereby the newer scores in red have lower values and thus the distribution gravitates towards a score of zero. This is ideal as the closer to zero the score of random pairs are, the easier they are to discriminate from positive interactions which generally have a higher score.

In terms of runtime, PIPE takes approximately an hour to predict 10,000 PPIs on 10 CPU cores, including the outputting of the intermediate 2D landscape files. PyPE is rather similar in that it takes approximately an hour to score 1,000 PPIs using provided 2D landscape files and GenTab files, and additionally testing one physicochemical approach as well, on one CPU core. ProtDCal was run on the entire human proteome of 21,284 proteins overnight on 6 CPU cores, with the precise time not observed.

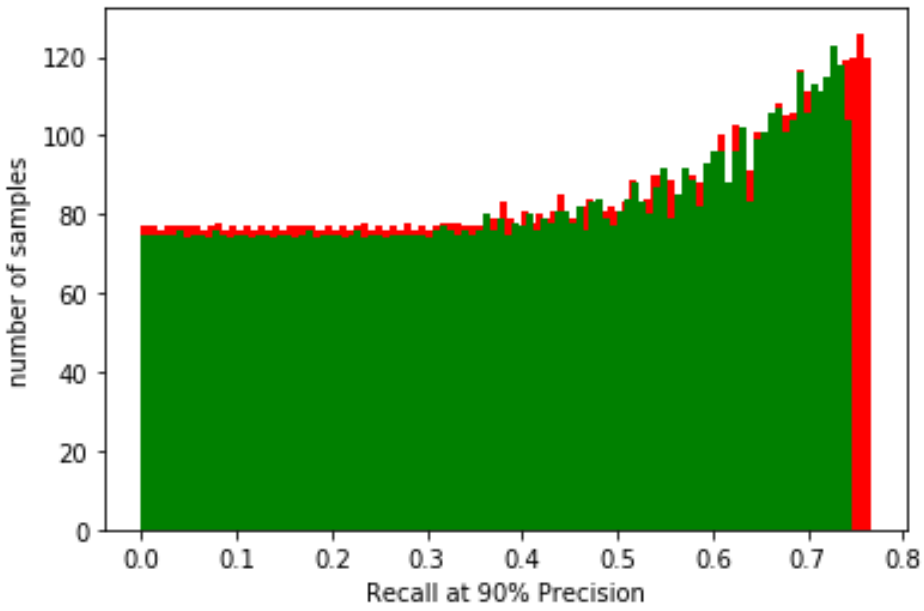


Figure 27: Histogram of recall scores at 90% precision. This histogram shows all of the conventional PIPE recall scores for a 90% precision in green, and the improved scores in red. As we can see the red scores represent higher recall values and a higher proportion of the recall values as compared to the conventional green scores. This was performed for the final 10,000 ground truth and 10,000 random pairs.

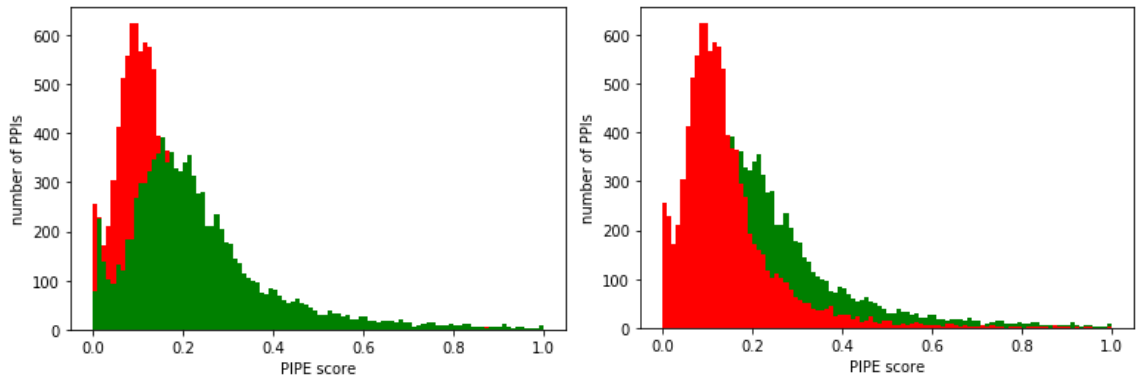


Figure 28: Histogram of PIPE scores for random pairs. This histogram shows all of the conventional PIPE recall scores in green and the new physicochemical scores in red specifically for the random pairs. We can see that the population for the new scores is shifted towards zero, which is ideal as this will help with correct binary classification. This was performed for the final 10,000 random pairs. Two histograms are shown alternating the original green PIPE scores and the new red PIPE scores taking the foreground to be more visible.

A more sophisticated precision-recall curve was generated for Figure 29. It includes the very right-hand side of equation 5 and features the incorporation of the previously discussed 100:1 class imbalance found in the literature for negative to positive PPIs. As per usual the original PIPE score curve is in green, with the new curve being in red. And we can see the shape of the curve is fairly different than a conventional precision-recall curve, however the new physicochemical influenced PIPE scores are an improvement to the curve. Figure 30 features a bootstrap test for statistical significance of the new PIPE score to the original PIPE score. The large 10,000 positives and 10,000 random interactions

dataset was used, the original and new PIPE scores were randomly scrambled with a 50% chance of being switched and 50% chance of being left in-place. At this point the recall was calculated for both scrambled sets at a 90% precision as per usual. Finally the difference in improvement to the scrambled new PIPE scores and the scrambled original PIPE scores was plotted. This bootstrap test was performed 1000 times, with a red vertical line representing the original physicochemical improvement of 1.9%. As we can see, only a 5 out of the 1000 bootstrap tests were able to meet or slightly exceed such an improvement, indicating a p-value of 0.005, thus providing a very strong statistical basis for the claim of rejecting the null hypothesis, which is that any apparently physicochemical improvement to PIPE was simply a result of chance.

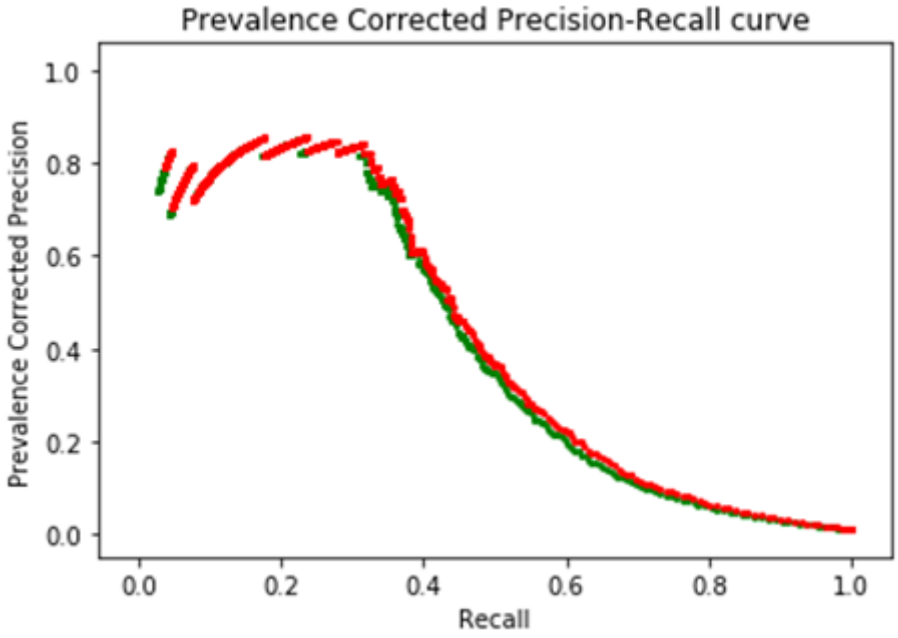


Figure 29: Prevalence-corrected Precision Recall Curve. Utilizing the very right hand side of equation 5, this figure takes into account the 100:1 class imbalance discussed throughout the work. The green curve represents the original performance of PIPE, and the red curve represents the improved physicochemical curve.

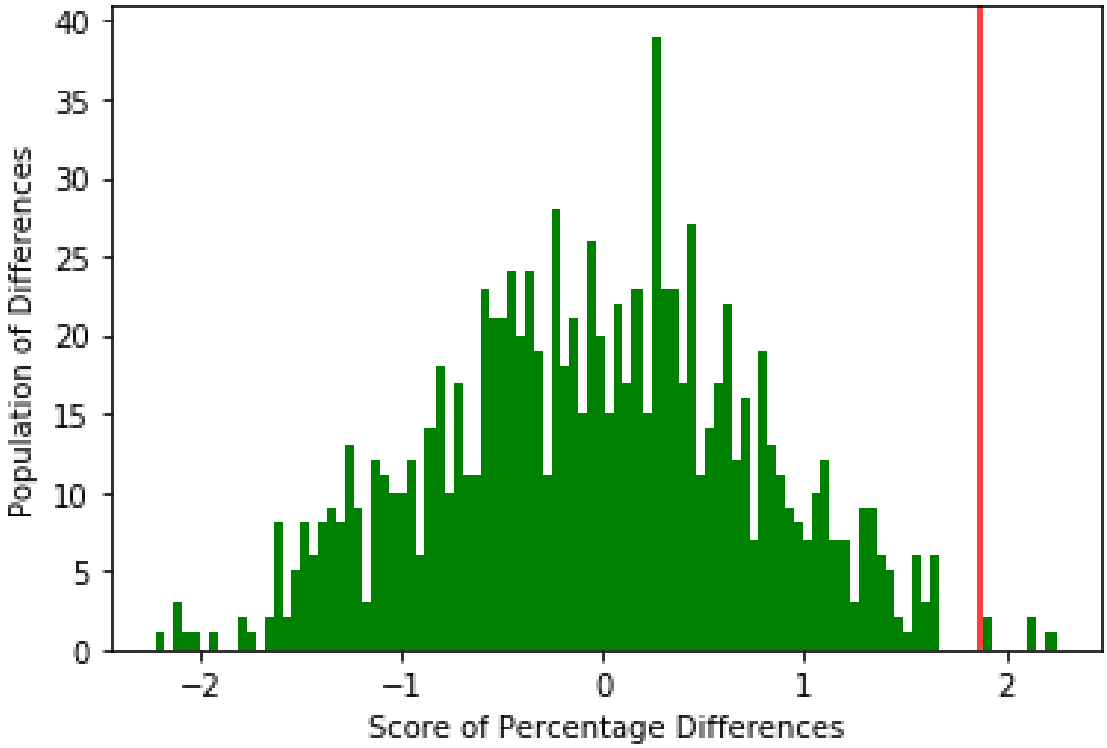


Figure 30: Bootstrap Analysis of the Improvement of PIPE. The 1.9% improvement to the recall of PIPE is represented by the vertical red line. 1000 random boostrap experiments were done whereby the original PIPE score and the improved PIPE score were randomly shuffled for the 10,000 ground truth and random pairs dataset. The percentage improvement to recall was then calculated for these 1000 bootstrap experiments. As can be seen, only a few out of the 1000 bootstrap experiments were able to out of pure chance exceed the improvement to PIPE by randomly finding a superior combination of original and new scores.

3.5 Discussion of Biological Plausibility of Findings

The physicochemical property of electrostatic potential distribution is an interesting one, as electrostatic potential has implications in a range of molecular properties including acid-base reactions, solvent behavior, pKa and others. It perhaps stands to reason that if organic chemistry and chemical reactions operate on election flow, then a property quantifying how unevenly distributed the electron density is initially, would be indicative of numerous chemical behaviours. It is therefore interesting that having evenly distributed electrons, and thus being non-polar, would be the single most valuable feature for PPI prediction at least out of the 166 features generated by ProtDCal that were attempted. It can also be seen from Table 2 that many of the other top-scoring improvements to PIPE from ProtDCal which were implemented in a (1 – feature) fashion had to do with radical reactivity (Fuk7) as well as features describing electronic kinetic energy density on the molecular surface (SIGA7, SIEPA10, SIEPA2). It is also interesting that (1 – electrostatic distribution) was a more valuable feature than simply calculating the non-polarity or aromaticity as physicochemical properties. Figure 31 features a visualization of the physicochemical property most successful in improving PIPE.

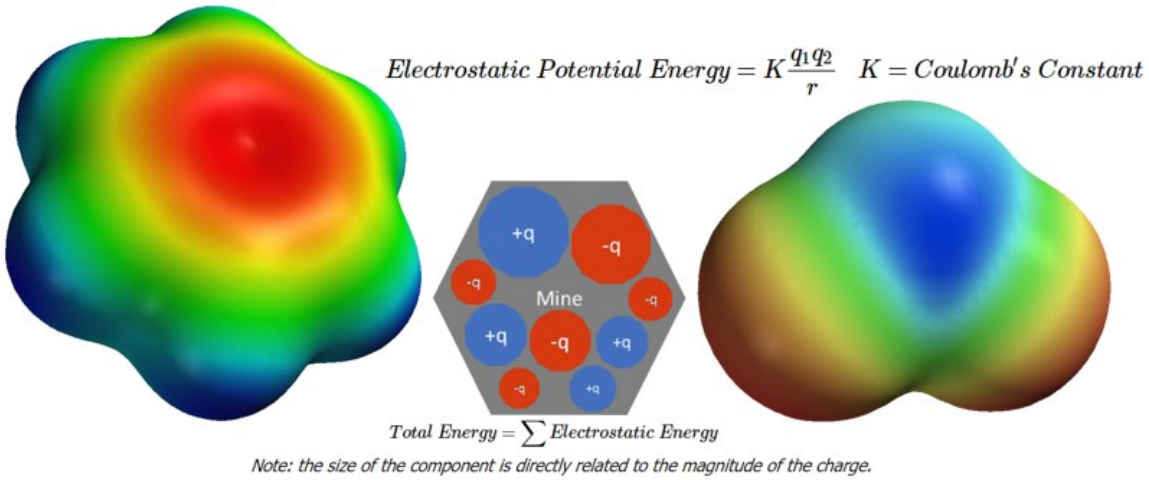


Figure 31: Overview of how electrostatic pote0ntial distribution is calculated by physics. Shown are how benzene and sulfur dioxide appear with such a mapping, respectively. (Figure reproduced from [50])

There is some contemporary discussion in the literature for understanding and discussing PPIs in terms of the non-polarity of the interacting amino acid residues, with recent publications describing "Class A complexes", that encompass ~60% of protein complexes and have a low interface polarity. These "Class A complexes" appear to leverage what we conventionally think of as a non-polar surface, which would be unstable in the presence of solvent water molecules. Water molecules are highly polar themselves, and these non-polar surfaces would seek each other out and engage in favourable hydrophobic interactions as a means of initiating PPIs, thereby losing their unfavourable interactions with water molecules. It is, in fact, the minority of ~40% of protein complexes, called "Class B complexes", that have a highly polar interface allowing them to be highly stable around solvent water molecules and able to engage with other polar PPIs via hydrophobic and polar interactions. Certainly one such difficulty these polar surfaces face is the lack of energetic advantage in changing from (relatively favourable) polar

interactions with water, to engaging in (equally favourable) polar interactions with a protein surface [51].

Some of the latest research in the area shows this Class A and Class B divide between non-polar surfaces for PPIs relative to polar surfaces can be related to the biological function of the protein complex. This is visualized in Figure 32 with Class A representing the PPIs most likely to be improved with this work. Furthermore a cellular specialization can be seen between Class A and B PPIs and this can be seen summarized in the first column of Table 3. For example, the highest electrostatic, and thus polar, PPI interfaces tend to be found in PPIs involved in the immune system. This is perhaps because these complexes must function in the most polar environments and thus be able to interact with the receptors of foreign agents which tend to be polar. Conversely, the least polar PPIs tend to be focused on biological assembly, as that may require the most specific interactions and take place in a less polar environment [52]. These data can be seen in Table 3. It perhaps stands to reason that if the majority of the surface of a protein is polar, allowing the protein to be generally stable in a water solvent, and given roughly half of all amino acids may be considered solvent accessible, then predicting solvent accessibility as with NetsurfP is insufficiently discriminatory to boost PPI prediction performance. In such a case, incorporating polarity is not very specific either, as once again much of the surface of a protein is polar. However modulating the PIPE landscape for non-polarity may be the most focused strategy, meaning if non-polar regions on a protein's surface are quite specific to PPIs, then increasing their PIPE score is disproportionately valuable and will introduce less noise and false positives than other approaches.

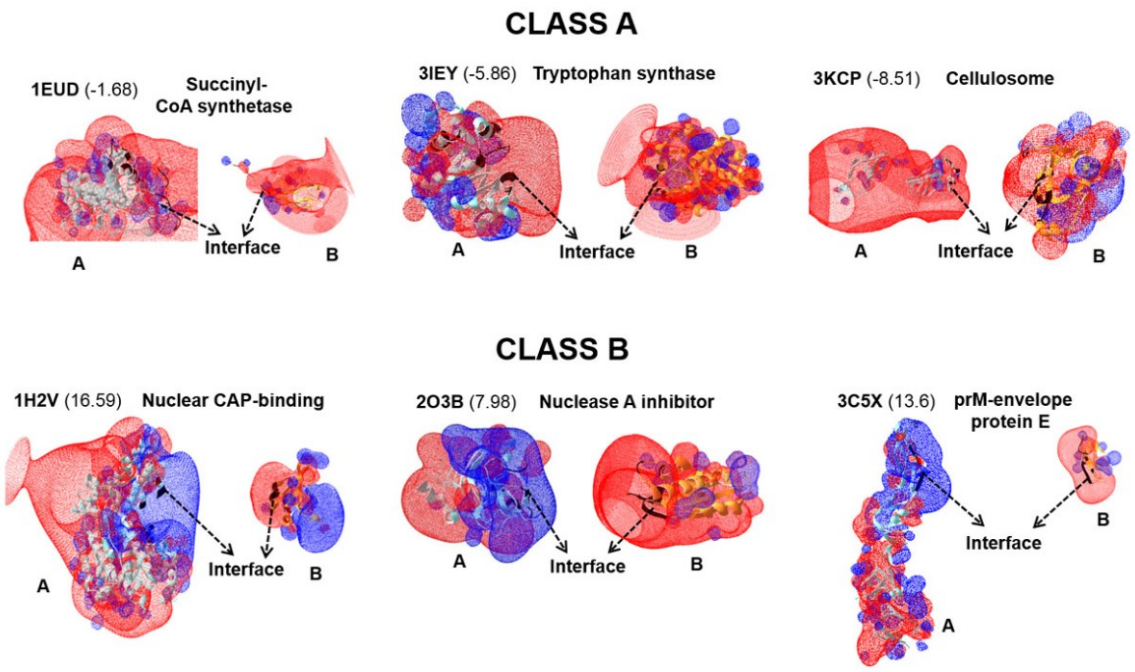


Figure 32: Surface electrostatic distribution of Class A and class B complexes. The electrostatic potential images of class A complexes show that the interface of chain A and of chain B have the same charges (same colours) implying electrostatic energy may not favour protein binding in class A complexes. Alternatively the electrostatic potential of class B complexes show the interface of proteins with opposite charges (different colours), suggesting electrostatic energy favours protein binding for class B complexes. (Figure reproduced from [51])

Table 3: Statistical analysis of different energies across different functional groups [52]

	Electrostatic energy (%)		H Bond energy (%)		vdW energy (%)	
	Mean	**SD**	**Mean**	**SD**	**Mean**	**SD**
All complexes (278)	11.26	8.7	15.03	6.54	74.93	11.37
Obligatory (208)	11.48	9.09	14.72	6.55	75.28	11.75
Enzyme (40)	9.73	9.41	15.49	5.03	74.78	9.64
Regulators (144)	11.87	9.28	14.72	7.22	75.07	12.47
Biological assembly (24)	9.20	9.04	13.41	3.98	77.39	10.6
Immune (18)	12.16	8.82	19.07	6.46	68.77	10.78
Non-obligatory (52)	10.08	7.01	14.89	6.16	75.68	9.42
Enzyme inhibitors (25)	9.81	7.94	15.35	6.05	74.84	10.23
Regulator inhibitors (27)	9.94	6.29	14.47	6.34	76.45	8.73

4 Conclusion

4.1 Thesis Summary

Chapter 1 featured an introduction to proteins, protein-protein interactions, and their difficulties and limitations in being determined in the lab and value in being determined computationally. Chapter 2 featured background introductions on the topics of pattern classification and how it can be scored and tested, how PPIs are determined *in vitro* using today's high throughput screening methods and how PPIs can be predicted *in silico* from 3D structure. Importantly for this work, it was also discussed how PPIs can be predicted *in silico* from amino acid sequence alone, and optionally with provided information on known PPIs which is information that can be leveraged. PIPE is just such an approach and how precisely it scores and functions was discussed. Chapter 3 featured the research contributions for this thesis, and this includes PIPE being left in a significantly improved state by means of comprehensive documentation. Next, PyPE was developed as a way to easily implement, score, and test large numbers of approaches to modulate, improve, or otherwise change the PIPE landscape and thus the PIPE score. Consequently, PyPE can be readily used by future researchers in future work to also improve the PIPE score and introduce new information to this sequence-based prediction method. Next, NetsurfP was implemented as a way to deploy the hypothesis that solvent accessibility can improve the accuracy of PPIs, it was unfortunately minimally effective at doing so. Finally, ProtDCal was deployed to generate 166 various physicochemical features for each amino acid in every protein in the human proteome. After much testing of algorithms and approaches, the lack of electrostatic potential, which perhaps could be called electrostatic uniformity, was seen as the most valuable physicochemical feature. Interestingly, there

exists some support in the literature for this being a sensible approach to improving PPI prediction [14], [40].

4.2 Discussion of Contributions

The contributions of this work include comprehensive documentation on how to utilize PIPE, this eight page documentation was requested and used by everyone currently working on PIPE. It features an overview of the file hierarchy and structure, the steps required to run the algorithm, what exact parameters in each file the user should set for a particular application, and what exact Linux commands must be entered to move through the PIPE workflow, with both single-server Linux SSH commands provided as well as distributed Simple Linux Utility for Research Management (SLURM) Linux SSH commands provided, which are more compatible with compute resources such as Compute Canada. Additionally common errors and pitfalls are also described along with corresponding solutions. The second contribution is the development of PyPE, which is an approximately 600 line Python program that can handle several hundred simultaneous modifications to the 3D PIPE landscape. PyPE can manually be run in many instances to leverage parallel computing. PyPE also features numerous *quality of life* (user interface) features and it exceeds other software packages in this regard. Furthermore, PyPE includes its own scoring and plotting features, powered by Matplotlib, to test the most successful feature. Thirdly, NetsurfpP was integrated with PIPE using 200 different configurations to test the hypothesis that introducing solvent accessibility information, derived from sequence via a neural network, would improve the predictive performance of PIPE. This was true; however, the improvement was only moderate at a 0.9% improvement in recall at a precision of 90%. This limited improvement could be due to NetsurfP only being 79%

accurate in its predictions of buried versus exposed amino acid residues, or because PIPE is already indirectly leveraging SA by inherently learning this from the database of known interactions used by PIPE. For example, the database of known PPI may reflect the fact that buried protein domains don't tend to interact, thereby resulting in low derived PIPE scores for these regions. It was discovered, through the 200 different configurations of NetSurfpP implementation, that raw scores and simple means with respect to window filtering perform best, this is perhaps because they do not adulterate or confound the provided scores. Finally, ProtDCal was leveraged to generate 166 individual amino acid physicochemical features and many different implementation approaches were used to combine, weight, and modulate these features. Ultimately, it was discovered that the "negative" of electrostatic potential distribution was clearly the best physicochemical feature to use and was able to improve recall by 2% or alternatively decrease false negatives by 7.6%. This was further validated on a test set of 10,000 ground truth and 10,000 random pairs and once again at a precision of 90%, the recall was increased from 72.1% to 74.0%, an improvement of 1.9%. The bootstrap test performed in Figure 30 shows a very strong statistical basis of $p = 0.005$, for why this improvement was significant. Attempts to leverage the top 3 and also top 5 physicochemical features in unison to modulate the PIPE landscape for even greater results were unsuccessful.

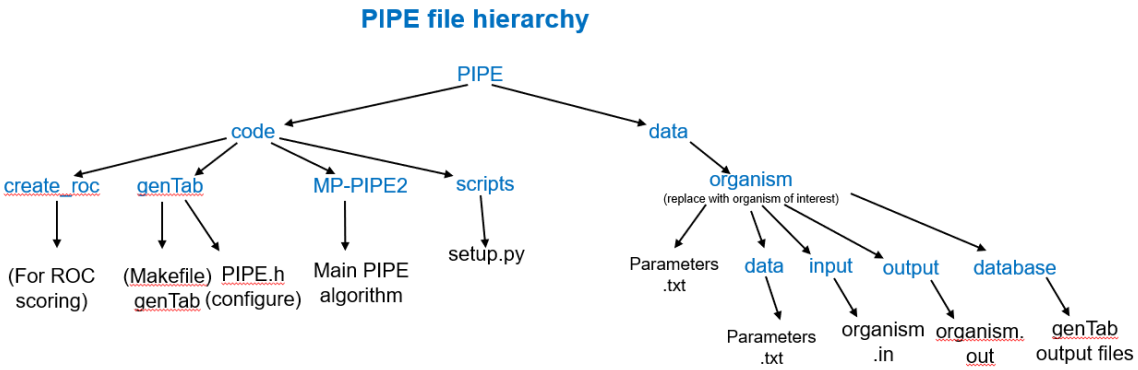
4.3 Recommendations for Future work

There is potential to leverage PyPE to attempt further and additional landscape modulations to continue to prove PIPE's scoring performance and accuracy, especially with multiple top features generated from ProtDCal, or even future improved versions of ProtDCal. Furthermore validation on species beyond *Homo sapiens* may be required.

One promising approach, that could not be attempted, is to leverage the top 5 or more ProtDCal physicochemical features summarized in Table 2, in order to train an artificial neural network (ANN) to determine the optimal weights and scaling factors for these numerous features and utilize them to improve the PIPE landscape. The attempted simple approach of merely taking the mean of these top 5 features or simply multiplying all these 5 features together was not sufficient to beat the performance of the single best feature alone. The approach of using machine learning to train the best feature weights is not possible within the framework of PyPE, as PyPE requires approximately one hour to test a hypothesis on 1000 ground truth pairs and 1000 random pairs on one CPU thread, and this is far too slow to train an ANN, where typical training requires rapid and repeated evaluation of the loss function (which would require modulation and assessment of 2000 PIPE landscapes). Such an approach would require implementing PyPE with a high performance ANN framework, such as PyTorch or Tensorflow. Fortunately, in theory such an approach is promising, as taking the best possible combination of five features, all of which exceed the default performance of PIPE, should be able to exceed the performance of simply the best single feature.

Appendix A

PIPE documentation



If you are on sharcnet / graham put the PIPE folder in your scratch directory to begin!
For example: /scratch/calvinj/PIPE/' is the working pipe version
You can always check what directory you are in by typing: "pwd" (meaning print working directory) the exact output you receive from this can be used within modifications to the code and within command line arguments

Step 1: Add proteins of interest or interactions of interest:

To this file for novel protein sequences:

PIPE\data\organism\data\protein_sequences.txt

To this file for novel protein interactions:

PIPE\data\organism\data\protein_pairs.txt

Pitfalls and concerns:
Large basepair proteins (over 30k basepairs) can lead to overflows
Integer overflows can also result from studying very large proteasomes (such as soy)

Step 2: Configure GenTab:

Open the
PIPE/code/genTab/convertPairs.pl

Set the absolute path of the protein sequences file: (this is an input) example:

$SEQ_FILE="/home/calvinjary/PIPE/data/organism/data/protein_sequences.txt";

Set the absolute path of the protein pairs file: (this is an input) example:
$STR_PAIR_FILE="/home/calvinjary/PIPE/data/organism/data/protein_pairs.txt";

Set the absolute path of the protein pairs index file: (this is an output) example:
$IDX_PAIR_FILE="/home/calvinjary/PIPE/data/organism/data/protein_pairs_index.txt";

Then in the PIPE/code/genTab directory run the script by typing:
./convertPairs.pl
This will create the protein_pairs_index.txt file that is required by GenTab
Note: you will receive the error: "Illegal division by zero at ./convertPairs.pl line 68" then your filepaths are not pointing to the correct directories (for protein_sequences.txt etc.)

Modify code for Sharcnet
Remove the optimization function "inline" from every line of code in the following files:
user/PIPE/code/genTab/routines.c
user/PIPE/code/genTab/PIPE.h
(Optional) run provided scripts to determine the max protein length, max neighbours (interacting partners per protein) and the max number of proteins

Set parameters for PIPE in PIPE.h
user/PIPE/code/genTab/PIPE.h specifically line 16:
#define MAX_LINE_LEN 24277
#define MAX_PROTEIN_LEN 8900 * there is a script that will tell you this from protein_sequences.txt file
#define MAX_NUM_PROTEINS 23000 * simply the number of lines in the protein_sequences.txt file
#define MAX_NEIGHBOURS 2000 *there is a script that will tell you this from protein_pairs_index.txt file
#define W 20 simply the length of the sliding window, really should be 20
#define SCORE 35 * PAM score threshold to be left at default as well
Note: protein_sequences.txt and protein_pairs_index.txt file are in the dir: PIPE\data\organism\data
Note: these suggested values will work perfectly with the default protein sequences and interactions

Set parameters for PIPE in genTab_org.txt
user/PIPE/data/organism/data/genTab_org.txt
default is:
1 // number of species
1000 // number of proteins
1 // number of species to generate a landscape for

0 // index of species 1

1: <Number of species>
2: <Number of proteins for species 1>
: ...
: <Number of proteins for species n>
: <Number of species we want tot generate files for>
: <Index of species 1>
: ...
: <Index of species n>

Set parameters for PIPE in parameters.txt

user/PIPE/data/organism/parameters.txt // mirrors the settings from other files via a script
num proteins 23000
num known pairs 67000 // simply the number of lines in **PIPE\data\organism\data\protein_pairs.txt**
num pairs 500500
max_db_file 530022400 // these numbers need to be bigger, check the
num_set_bits 151872 // these numbers need to be bigger
max_protein_len 8900 // 24177
max_neighbours 2000 // 3000

Set parameters for PIPE in PIPE_org.txt

PIPE/data/organism/data/PIPE_org.txt

1 <Number of species>
1 <Number of training datasets>
0,0 <index of training dataset of species>, <index of training dataset of species>

Note:
Ensure full read/write/execute permissions in the PIPE directory, example:
chmod -R ug+rwx PIPE (recursive, user, group, read, write & exec for the working dir)
if this doesn't work may need to do chmod -R ugo+rwx working (user, group, other)

Step 3: Run GenTab:

Clean up previous compiled versions and compile a new version

Go to **user/PIPE/code/genTab/**
type "make clean" (this will clean up and remove previously compiled executables
type: "make" (this will run the Makefile present in the genTab directory and generate the executables)

Ensure the output directory of GenTab exists, known as: "database"
Example dir is home/calvinjary/PIPE/data/organism/database by typing in: "mkdir database " (make sure to create this directory within data/organism)

RAM requirement:
Number of proteins * length of longest protein * 8 bytes * 2 (for redundant array)
To convert to gigabytes divide by 1024^3

Theoretical Example: 40,000 proteins * 20,000 amino acids * 8 bytes * 2 = 12 gb of ram
Experimental Example: 20,500 proteins * 8,900 amino acids was 1.65 GB of RAM (theoretical is 2.7 GB)

CPU Runtime:
Gentab should scale linearly across number of cores, and across execution time. Setting up gentab for 20,500 proteins across 8 cores took about 16 hours, so scale up and down according to cpu cores and number of proteins
11,000 proteins across 20 cores took 2 hours

Storage requirements:
20 GB of /database files were generated with those 20,500 proteins. This also scales linearly with number of proteins assuming the same average protein length (this was for humans)

The arguments required to run genTab are provided here
0 is the directory location of the compiled genTab
1 is the protein sequences file protein_sequences.txt (PROTEIN_SEQ_FILE)
2 is the protein database directory (to be populated by genTab) (DBDIR)
3 is the genTab_org.txt settings file (ORG_SETTINGS)
4 protein_pairs_index file created by the convertPairs.pl script (PAIRS_FILE)

Example Carleton research compute services) (8 cores, unspecified runtime and ram per CPU
mpirun -n 8 /home/calvinjary/PIPE/code/genTab/genTab
/home/calvinjary/PIPE/data/organism/data/protein_sequences.txt
/home/calvinjary/PIPE/data/organism/database
/home/calvinjary/PIPE/data/organism/data/genTab_org.txt
/home/calvinjary/PIPE/data/organism/data/protein_pairs_index.txt &> output.out &
disown

PIPE version 3 GenTab:
mpirun -n 6 /home/calvinjary/PIPE/code/genTab/genTab
/home/calvinjary/PIPE/data/organism/data/protein_sequences.txt
/home/calvinjary/PIPE/data/organism/database &> output.out & disown

Example (Sharcnet compute Canada) with relative path lengths
srun -t 16:00:00 --ntasks 12 --mem-per-cpu=4G ./code/genTab/genTab ./data/organism/data/protein_sequences.txt ./data/organism/database ./data/organism/data/genTab_org.txt ./data/organism/data/protein_pairs_index.txt & disown

Note: if you get the error: "ERROR: Lines in /home/calvinjary/PIPE/data/organism/data/protein_sequences.txt longer than buffer (24277 byte)"
you have underestimated the max length of your biggest protein in the genTab/PIPE.h file

If you get a segfault (segmentation fault) you need to add full linux permissions to the PIPE folder with:
sudo chmod -R ugo+rwx PIPE

Step 4: Configure PIPE:

Set parameters for PIPE in MP-PIPE2.c
user/PIPE/code/MP-PIPE2/mp-pipe2.c around code line #59
static const int WINDOW_SIZE = 20;
static const int FILTER_SIZE = 3;
static const double THRESHOLD = 0;
static const int PACKET_SIZE = 1000;
int THREADS = 2; // number of logical processors to be used, this is easily changed in the run command

//CAT'S constants
#define MAX_DB_FILE 505400 // this is very likely the total size (in kilobytes) of the genTab /database folder So 505500 for the entire genTab folder for the conservative pairs set
1712000 1717556 type "du database" in the "organism" directory
#define NUM_SET_BITS 600872 // equal to max protein length * 64
#define MAX_PROTEIN_LEN 8900 // maximum protein length, easily discovered
#define MAX_NEIGHBOURS 2000 // # of max interactions per protein, consider setting this to 3000

Update the organism.in file found in:
/home/calvinjary/PIPE/data/organism/input/organism.in
The first line refers to the number of other lines in the file (so the total number of interactions)
Every other line is the index (starting with 0) of your protein of interest. Protein X, a tab separation and then protein of interest Y. Example format:
4 < total number of other lines (total number of interactions to compute)
0 0 < (protein 0 interacting with itself)
0 1 < (protein 0 interacting with protein 1)
0 2 < (protein 0 interacting with protein 2)
0 3 < (protein 0 interacting with protein 3)

Note: be sure to execute any script to generate this comparison table in a linux environment (same environment to run PIPE) as a Windows environment will differently encode the file and cause errors

Update and compile the "mp-pipe2" executable
Within the MP-PIPE2 folder
Delete: "mp-pipe2" (so you can cleanly compile mp-pipe2.c) then type the command:
mpicc -O3 -fopenmp -Wall mp-pipe2.c -m64 -lm -o mp-pipe2

RAM requirement:
This is going to be half of the /home/calvinjary/PIPE/data/organism/database directory (the gentab directory) for any run that uses all proteins. So both all vs all and 1 vs all will use 10 GB of RAM if the /database directory is 20 GB

CPU Runtime:
Gentab should scale linearly across number of cores, and across execution time. There is one master thread and one slave thread. So for a 20 core CPU you would set n = 10 and this would use up all 20 cores (meaning 20 threads) Example runtime for 50 proteins vs 20,500 proteins was 3 hours using 3 threads (1 master thread and two threads using 10 GB of RAM each)

Storage requirements:
The organism.out file was 150 MB for a 50 vs 20,500 protein run, (about 1 million comparisons) and this file should scale linearly

Step 5: Run PIPE:

Note: if you fail to set define MAX_DB_FILE to a sufficient value (total size of the database directory) then you will notice your process being killed if you run PIPE, but no additional error will be shown)

Now run PIPE with the required arguments, examples provided

(the arguments required to run PIPE are provided here)
0 is the directory location of the compiled MP-PIPE2
1 is input file (pairs list), which uses number-based protein identifiers
2 is outpufit file, the results of PIPE, file created dynamically
3 is interaction graph file known as protein_pairs_index.txt
4 is pipe database directory (the results of GenTab)
5 is organism settings file PIPE_org.txt

(example run command for Carleton university research compute services) (3 cores, Linux, unspecified runtime and ram per CPU)

mpirun -n 3 /home/calvinjary/PIPE/code/MP-PIPE2/mp-pipe2
/home/calvinjary/PIPE/data/organism/input/organism.in
/home/calvinjary/PIPE/data/organism/output/organism.out
/home/calvinjary/PIPE/data/organism/data/protein_pairs_index.txt
/home/calvinjary/PIPE/data/organism/database
/home/calvinjary/PIPE/data/organism/data/PIPE_org.txt &> output.out & disown

This will create a file called "output.out" in whatever your current directory is, with all the status updates as PIPE is running. If you would like to receive those in real-time in the console simply delete:
"> output.out" from the command

PIPE version 3:
(example run command for Carleton university research compute services) (3 cores, linux, unspecified runtime and ram per CPU)
mpirun -n 12 /home/calvinjary/PIPE/code/MP-PIPE2/mp-pipe2
/home/calvinjary/PIPE/data/organism/input/organism.in
/home/calvinjary/PIPE/data/organism/output/organism.out
/home/calvinjary/PIPE/data/organism/data/protein_pairs_index.txt
/home/calvinjary/PIPE/data/organism/database &> output.out & disown

(For a SLURM scheduler such as Sharcnet's Graham, compute Canada)
(this is to run for 1 hour, with 2 threads and 4 gigs of ram per thread)
srun -t 1:00:00 --ntasks 2 --cpus-per-task=2 --mem-per-cpu=4G
/home/calvinj/working/PIPE/code/MP-PIPE2/mp-pipe2
/home/calvinj/working/PIPE/data/organism/input/organism.in
/home/calvinj/working/PIPE/data/organism/output/organism.out
/home/calvinj/working/PIPE/data/organism/data/protein_pairs_index.txt
/home/calvinj/working/PIPE/data/organism/database
/home/calvinj/working/PIPE/data/organism/data/PIPE_org.txt &> disown

Output of PIPE
home/calvinjary/PIPE/data/organism/output/organism.out

The column headings are explained in the header comment of the pipe c file.
But it should be pipe score, matrix max, run time, Sim score 1, Sim score 2

Step 6: Create an ROC Curve:

Go to the PIPE/code/create_roc/ directory

Run the python script that will create random (assumed negative) interactions. The program is called:
create_random_ID_pairs.py
and it accepts these inputs:

known_ID_pairs_file
/home/calvinjary/PIPE/data/organism/data/protein_pairs.txt
total_num_proteins 11138
num_pairs_to_produce 10000
output_file calrandompairs.txt

type in this example command:
python2 create_random_ID_pairs.py
/home/calvinjary/PIPE/data/organism/data/protein_pairs.txt 11138 1000000
calrandompairs.txt & disown

this completes the create random ID pairs section.

now type in “make clean” and then “make” following that

/home/calvinjary/PIPE/data/organism/data/protein_pairs.txt

"Usage: %s <pos file> <#pos> <neg file> <#neg>\n", argv[0]);

mpirun -n 10 /home/calvinjary/PIPE/code/create_roc/roc
/home/calvinjary/PIPE/code/create_roc/proteincal.csv 48106
/home/calvinjary/PIPE/code/create_roc/calrandompairs.txt 1000000 > output.out &
disown

Step 7: LOOCV:

Confirm you have gnuplot with the command:
gnuplot --version

dependencies required for gnuplot:

sudo apt-get install python2
sudo apt-get install python3
sudo apt-get install python-numpy
sudo apt-get install python-scipy
sudo apt-get install python-matplotlib
sudo apt-get install gnuplot
sudo apt-get install cclib
sudo apt-get install libnss-mdns
sudo apt-get install gnuplot

confirm gnuplot is installed with the commands:
gnuplot –version and also: which gnuplot

sudo apt-get install cclib --upgrade

in the /home/calvinjary/PIPE/code/scripts folder you will run the file loocv.py but first open it and:
A- At line 19, Change the local_dir to the directory where the PIPE folder is. for me it was “/home/calvinjary/PIPE” so I put “/home/calvinjary/"
B- At line 20, change the remote_dit to the same directory so “/home/calvinjary/" again
C- At line 28, change organisim_name to the name of the organism PIPE ran.

On line 124 and 137 replace the
str(num_mp_pipe_hosts)
With how many threads you want PIPE to run on. Keeping in mind pipe.c runs on 2 threads. So if you have 20 cores available set str(num_mp_pipe_hosts) to str(9) and this will run on 18 cores with 1 additional master core, so you will use 19.

In the same /home/calvinjary/PIPE/code/scripts folder open the update.py script and comment out the bottom two sections (for loops), which are for distributed clusters. they include both the “for machines in machines” loops

Open the
/home/calvinjary/PIPE/code/MP-PIPE2/PIPE_hosts file
Make sure all of the nodes including server and desk and node are commented out with a #. Make sure localhost is the only node not commented out. Example:
localhost
#server
#desk01
#node01

Now you can run the LOOCV script with the command:
python2 loocv.py > output.out & disown

and the results will be in the /organism/data/LOOCV folder

References

[1] M. Totrov, “Accurate and Efficient Generalized Born Model Based on Solvent Accessibility: Derivation and Application for LogP Octanol/Water Prediction and Flexible Peptide Docking,” *Journal of computational chemistry*, vol. 25, no. 4. John Wiley & Sons, New York :, pp. 609–619, 2004.

[2] A. Stein and P. Aloy, “Novel peptide-mediated interactions derived from high-resolution 3-dimensional structures,” *PLoS Comput. Biol.*, vol. 6, no. 5, pp. 1–16, 2010.

[3] J. Craig Venter *et al.*, “The sequence of the human genome,” *Science (80-.).*, vol. 291, no. 5507, pp. 1304–1351, 2001.

[4] L. D. Stein, “Human Genome: End of the beginning,” *Nature*, vol. 431, no. 7011, pp. 915–916, 2004.

[5] G. Brown, T. Tatusova, and K. Pruitt, “The NCBI handbook,” pp. 1–24, 2002.

[6] M. P. H. Stumpf *et al.*, “Estimating the size of the human interactome,” *Proc. Natl. Acad. Sci. U. S. A.*, vol. 105, no. 19, pp. 6959–6964, 2008.

[7] T. K. Chaudhuri and S. Paul, “Protein-misfolding diseases and chaperone-based therapeutic approaches,” *FEBS J.*, vol. 273, no. 7, pp. 1331–1349, 2006.

[8] A. Dziembowski and B. Séraphin, “Recent developments in the analysis of protein complexes,” *FEBS Lett.*, vol. 556, no. 1–3, pp. 1–6, 2004.

[9] N. Tuncbag, G. Kar, O. Keskin, A. Gursoy, and R. Nussinov, “A survey of available tools and web servers for analysis of protein-protein interactions and interfaces,” *Brief. Bioinform.*, vol. 10, no. 3, pp. 217–232, 2009.

[10] V. Mering *et al.*, “Comparative assessment of large-scale data sets of protein-protein interactions,” *Nature*, vol. 417, pp. 399–403, 2002.

[11] V. Mehta and L. Trinkle-Mulcahy, “Recent advances in large-scale protein interactome mapping [version 1; referees: 3 approved],” *F1000Research*, vol. 5, no. 0, pp. 1–9, 2016.

[12] W. Zhang *et al.*, “A Point-Charge Force Field for Molecular Mechanics Simulations of Proteins Based on Condensed-Phase,” *J. Comput. Chem.*, vol. 24, no. 16, p. 1999, 2003.

[13] S. Pronk *et al.*, “GROMACS 4.5: A high-throughput and highly parallel open source molecular simulation toolkit,” *Bioinformatics*, vol. 29, no. 7, pp. 845–854, 2013.

[14] B. Petersen, T. N. Petersen, P. Andersen, M. Nielsen, and C. Lundegaard, “A generic method for assignment of reliability scores applied to solvent accessibility predictions,” *BMC Struct. Biol.*, vol. 9, pp. 1–10, 2009.

[15] Y. B. Ruiz-Blanco, W. Paz, J. Green, and Y. Marrero-Ponce, “ProtDCal: A program to compute general-purpose-numerical descriptors for sequences and 3D-structures of proteins,” *BMC Bioinformatics*, vol. 16, no. 1, pp. 1–15, 2015.

[16] S. Romero-Molina, Y. B. Ruiz-Blanco, J. R. Green, and E. Sanchez-Garcia, “ProtDCal-Suite: A web server for the numerical codification and functional analysis of proteins,” *Protein Sci.*, vol. 28, no. 9, pp. 1734–1743, 2019.

[17] C. Patulea, “Targeted Optimization of Computational and Classification Performance of a Protein-Protein Interaction Predictor,” 2011.

[18] T. Fawcett, “An introduction to ROC analysis,” *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, 2006.

[19] S. Pitre *et al.*, “Short co-occurring polypeptide regions can predict global protein interaction maps,” *Sci. Rep.*, vol. 2, pp. 1–10, 2012.

[20] Y. Park, “Critical assessment of sequence-based protein-protein interaction prediction methods that do not require homologous protein sequences.,” *BMC Bioinformatics*, vol. 10, p. 419, 2009.

[21] S. H. Park, J. M. Goo, and C. H. Jo, “Receiver operating characteristic (ROC) curve: Practical review for radiologists,” *Korean J. Radiol.*, vol. 5, no. 1, pp. 11–18, 2004.

[22] B. A. Shoemaker and A. R. Panchenko, “Deciphering protein-protein interactions. Part I. Experimental techniques and databases,” *PLoS Comput. Biol.*, vol. 3, no. 3, pp. 0337–0344, 2007.

[23] B. de Chassey, L. Meyniel-Schicklin, J. Vonderscher, P. André, and V. Lotteau, “Virus-host interactomics: New insights and opportunities for antiviral drug discovery,” *Genome Med.*, vol. 6, no. 11, pp. 1–14, 2014.

[24] K. Miura, “An Overview of Current Methods to Confirm Protein- Protein Interactions,” *Protein Pept. Lett.*, vol. 25, no. 8, pp. 728–733, 2018.

[25] A. M. Edwards, B. Kus, R. Jansen, D. Greenbaum, J. Greenblatt, and M. Gerstein, “Bridging structural biology and genomics: Assessing protein interaction data with known complexes,” *Trends Genet.*, vol. 18, no. 10, pp. 529–536, 2002.

[26] N. Simonis *et al.*, “Empirically controlled mapping of the Caenorhabditis elegans protein-protein interactome network,” *Nat. Methods*, vol. 6, no. 1, pp. 47–54, 2009.

[27] M. Jessulat *et al.*, “Recent advances in proteinprotein interaction prediction: Experimental and computational methods,” *Expert Opin. Drug Discov.*, vol. 6, no. 9, pp. 921–935, 2011.

[28] Michael E. Peskin and D. V. Schroeder, *An Introduction to Quantum Field Theory*. NorthWestern State: Taylor & Francis Group, 2018.

[29] J. Hoskins, S. Lovell, and T. L. Blundell, “An algorithm for predicting protein-protein interaction sites: Abnormally exposed amino acid residues and secondary structure elements,” *Protein Sci.*, vol. 15, no. 5, pp. 1017–1029, 2006.

[30] J. Jia, Z. Liu, X. Xiao, B. Liu, and K. C. Chou, “iPPI-Esml: AN ensemble classifier for identifying the interactions of proteins by incorporating their physicochemical properties and wavelet transforms into PseAAC,” *J. Theor. Biol.*, vol. 377, pp. 47–56, 2015.

[31] J. A. Marsh and S. A. Teichmann, “Relative solvent accessible surface area predicts protein conformational changes upon binding,” *Structure*, vol. 19, no. 6, pp. 859–867, 2011.

[32] R. P. Saha, R. P. Bahadur, A. Pal, S. Mandal, and P. Chakrabarti, “ProFace: A server for the analysis of the physicochemical features of protein-protein interfaces,” *BMC Struct. Biol.*, vol. 6, pp. 1–5, 2006.

[33] National Human Genome Research Institute, “The Cost of Sequencing a Human Genome,” 2019. [Online]. Available: https://www.genome.gov/about-genomics/fact-sheets/Sequencing-Human-Genome-cost. [Accessed: 10-Dec-2019].

[34] K. Liolios *et al.*, “The Genomes On Line Database (GOLD) in 2009: Status of genomic and metagenomic projects and their associated metadata,” *Nucleic Acids Res.*, vol. 38, no. SUPPL.1, pp. 346–354, 2009.

[35] S. Lander *et al.*, “Initial sequencing and analysis of the human genome International Human Genome Sequencing Consortium* The Sanger Centre: Beijing Genomics Institute/Human Genome Center,” *Nature*, vol. 409, no. February, 2001.

[36] J. Wu, T. Vallenius, K. Ovaska, J. Westermarck, T. P. Mäkelä, and S. Hautaniemi, “Integrated network analysis platform for protein-protein interactions,” *Nat. Methods*, vol. 6, no. 1, pp. 75–77, 2009.

[37] B. Ma, T. Elkayam, H. Wolfson, and R. Nussinov, “Protein-protein interactions: Structurally conserved residues distinguish between binding sites and exposed protein surfaces,” *Proc. Natl. Acad. Sci. U. S. A.*, vol. 100, no. 10, pp. 5772–5777, 2003.

[38] Y. Ofran and B. Rost, “Predicted protein-protein interaction sites from local sequence information,” *FEBS Lett.*, vol. 544, no. 1–3, pp. 236–239, 2003.

[39] Y. Ofran and B. Rost, “Analysing six types of protein-protein interfaces,” *J. Mol. Biol.*, vol. 325, no. 2, pp. 377–387, 2003.

[40] X. Gallet, B. Charloteaux, A. Thomas, and R. Brasseur, “A fast method to predict protein interaction sites from sequences,” *J. Mol. Biol.*, vol. 302, no. 4, pp. 917–926, 2000.

[41] S. Liu, C. Liu, and L. Deng, “Machine learning approaches for protein-protein interaction hot spot prediction: Progress and comparative assessment,” *Molecules*, vol. 23, no. 10, 2018.

[42] S. Pitre *et al.*, “PIPE: A protein-protein interaction prediction engine based on the re-occurring short polypeptide sequences between known interacting protein pairs,” *BMC Bioinformatics*, vol. 7, pp. 1–15, 2006.

[43] S. Pitre *et al.*, “Global investigation of protein-protein interactions in yeast Saccharomyces cerevisiae using re-occurring short polypeptide sequences,” *Nucleic Acids Res.*, vol. 36, no. 13, pp. 4286–4294, 2008.

[44] B. D. Barnes, “Accelerated Transfer Learning for Protein-Protein Interaction Prediction,” 2018.

[45] W. R. Pearson, “Selecting the Right EMR Vendor Selecting the Right EMR Vendor,” *Curr Protoc Bioinforma.*, no. 43, pp. 3.5.1–3.5.9, 2013.

[46] Y. Li and L. Ilie, “SPRINT: Ultrafast protein-protein interaction prediction of the entire human interactome,” *BMC Bioinformatics*, vol. 18, no. 1, pp. 1–11, 2017.

[47] S. Romero-Molina, Y. B. Ruiz-Blanco, M. Harms, J. Münch, and E. Sanchez-Garcia, “PPI-Detect: A support vector machine model for sequence-based prediction of protein–protein interactions,” *Journal of Computational Chemistry*, vol. 40, no. 11. pp. 1233–1242, 2019.

[48] A. Chatr-Aryamontri *et al.*, “The BioGRID interaction database: 2015 update,” *Nucleic Acids Res.*, vol. 43, no. D1, pp. D470–D478, 2015.

[49] K. Dick, F. Dehne, A. Golshani, and J. R. Green, “Positome: A method for improving protein-protein interaction quality and prediction accuracy,” *2017 IEEE Conf. Comput. Intell. Bioinforma. Comput. Biol. CIBCB 2017*, 2017.

[50] T. Bottyan, “Electrostatic Potential maps.” [Online]. Available: https://chem.libretexts.org/Bookshelves/Physical_and_Theoretical_Chemistry_Textbook_Maps/Supplemental_Modules_(Physical_and_Theoretical_Chemistry)/Chemical_Bonding/Fundamentals_of_Chemical_Bonding/Electrostatic_Potential_maps. [Accessed: 02-Jan-2020].

[51] G. Sowmya and S. Ranganathan, “Discrete structural features among interface residue-level classes,” *BMC Bioinformatics*, vol. 16, no. 18, p. S8, 2015.

[52] C. Nilofer, A. Sukhwal, A. Mohanapriya, and P. Kangueane, “Protein-protein interfaces are vdW dominant with selective H-bonds and (or) electrostatics towards broad functional specificity,” *Bioinformation*, vol. 13, no. 06, pp. 164–173, 2017.