Institut für Informatik
Lehrstuhl für Bioinformatik/I12

SEQUENCE BASED PREDICTION OF PROTEIN-PROTEIN INTERACTIONS

Tobias Hamp

Vollständiger Abdruck der von der Fakultät für Informatik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Naturwissenschaften (Dr. rer. nat.)

genehmigten Dissertation.

Vorsitzender		Univ.-Prof. Dr. Daniel Cremers
Prüfer der Dissertation:		
	1.	Univ.-Prof. Dr. Burkhard Rost
	2.	Univ.-Prof. Dr. Dimitrij Frischmann

Die Dissertation wurde am 11.03.2014 bei der Technischen Universität München

eingereicht und durch die Fakultät für Informatik am 07.05.2014 angenommen.

Abstract

Proteins are the driving force behind cellular processes. They perform their tasks by interacting with other molecules, in particular other proteins. Understanding life on the molecular level therefore requires collecting and analyzing protein-protein interactions (PPIs) on a large scale. Experimental methods are limited, however. Capturing the precise molecular mechanisms of a PPI is highly exceptional. Determining which proteins interact and which do not is easier and faster, but accompanied with high error rates caused by missed and falsely detected PPIs. Many other aspects of PPIs fall between these two extremes, but the picture of largely incomplete data remains. Using existing PPI data in an optimal way is therefore of crucial importance. With machine learning, for example, we can rank protein pairs with respect to their probability to interact. Statistical analyses connect diverse small-scale experiments and unravel hidden relationships. This thesis contributes to these fields of research and particularly focuses on protein sequences, which are now abundantly available thanks to great technical progress. We first collected and analyzed all known structures of PPIs with respect to their interfaces, i.e. the surface regions of the two proteins that are in touch. Surprisingly, many proteins interact through different interfaces, adding yet another twist to the problem of predicting which pairs of amino acids are in contact. Nevertheless, we developed a new machine learning tool using artificial neural networks that helps solving exactly this problem, given only the sequences of the interacting proteins. For determining whether two proteins interact, this predictor was not as accurate as specialized tools. We managed to come up with another method, however, that improves over the state-of-the-art also in this respect. It relies on evolutionary patterns found in today's sequence databases and support vector machines as the discriminatory method. We used it to annotate all 200 million protein pairs in human. Finally, PPIs can be represented as networks and a common task is to find overrepresented protein functions in certain network regions. Largely missing experimentally determined function annotations again asks for computational tools. Addressing this problem, we developed a new state-of-the-art function predictor that only relies on sequential similarity of the target protein to proteins with known annotations.

Zusammenfassung

Zelluläre Prozesse werden durch Proteine gesteuert, die mit anderen Molekülen interagieren, insbesondere mit anderen Proteinen. Um Leben auf molekularer Ebene zu verstehen, müssen solche Protein-Protein Interaktionen (PPIs) daher in großem Umfang gesammelt und analysiert werden. Heutige experimentelle Möglichkeiten sind jedoch stark eingeschränkt. Nur selten gelingt es atomare Details einer Interaktion zu erfassen. Die Trennung von interagierenden und nicht-interagierenden Proteinen ist einfacher und schneller, aber verbunden mit hohen Fehlerraten, verursacht durch nicht oder falsch erkannte PPIs. Auch Aspekte zwischen diesen beiden Extremen verändern nicht das Bild, dass Daten großflächig fehlen. Umso wichtiger ist es, bestehende PPI Daten optimal zu nutzen. In Kombination mit maschinellem Lernen etwa erlauben sie, Proteinpaarungen nach ihrer Interaktionswahrscheinlichkeit zu ordnen. Statistische Analysen können Einzelexperimente miteinander verbinden und dadurch unbekannte Beziehungen aufdecken. Diese Arbeit leistet einen Beitrag auf genau diesem Gebiet und fokussiert sich insbesondere auf Proteinsequenzen, die dank großer technischer Fortschritte nun in enormer Zahl zur Verfügung stehen. Zunächst sammelten wir alle bekannten Strukturen von PPIs und analysierten sie hinsichtlich der Berührungsfläche der beiden Proteine. Überraschenderweise ändern sich diese oft in unterschiedlicher Weise, was das ohnehin erhebliche Problem verschärft, festzustellen, welche Aminosäuren miteinander interagieren. Trotzdem haben wir ein auf künstliche neuronale Netze basierendes Programm entwickelt, das genau diese Frage adressiert. Nur anhand der Sequenzen zweier interagierender Proteine bestimmt es die Tendenz jedes Aminosäurepaares eine Bindung einzugehen. Die Genauigkeit liegt dabei deutlich über der Hintergrundwahrscheinlichkeit. Leider war das Programm bei der Unterscheidung interagierender und nicht-interagierender Proteine schlechter als spezialisierte Methoden. Mit einem neuen Ansatz konnten wir jedoch auch dort Fortschritte erzielen. Dabei benutzten wir evolutionäre Muster, die sich in großem Umfang aus Sequenzdatenbanken ablesen lassen, und sogenannte „Support Vector Machines“, Werkzeuge des maschinellen Lernens, spezialisiert auf die Diskriminierung zwischen zwei Klassen. In einem großangelegten Einsatz bestimmten wir die Interaktionswahrscheinlichkeit aller 200 Millionen Proteinpaare im Menschen. PPIs können als Netzwerke dargestellt und bestimmte Netzwerkregionen auf signifikant überrepräsentierte Proteinfunktionen analysiert werden. Erneut sind computergestützte Methoden notwendig, um großflächig fehlende experimentelle Funktionsannotationen zu ergänzen. Als finalen Beitrag haben wir ein neues Funktionsvorhersageprogramm entwickelt, das in seiner Genauigkeit dem neuesten Stand der Technik entspricht und nur auf Sequenzähnlichkeit zwischen Zielprotein und bereits annotierten Proteinen zurückgreift.

Acknowledgments

First of all, I want to thank my supervisor Professor Burkhard Rost. He gave me the opportunity to become a PhD student, always supported me throughout the years and even allowed me to start my thesis in New York City. Many thanks also go to Laszlo Kajan and Tim Karl who did a great job of building and maintaining a stable high performance computing environment. Without them, large parts of this work would not have been possible. At the beginning of my thesis, Mikhail Veshtort gave me great input and discussion from a physicist's point of view, and I also want to thank him very much.

Finally, I am very grateful to Marlena Drabik who brought order into chaos countless times, to Dedan Githae for a lot of fun moments and to the whole Rostlab for all the great celebrations and excursions. Finally, I want to thank my parents, the rest of my family and all of my friends for their never-ending support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Proteins are the main actors in the living cell and the manifestation of the information encoded in genes. They perform their tasks by interacting with other molecules, in particular other proteins. For well over a decade, the study of protein-protein interactions (PPIs) has been a major activity in computational biology. For instance, representing proteins as nodes and interactions as edges creates so-called PPI networks. They provide a bird's-eye view of cellular processes and allow the application of PPIs in a range of other areas. One recent major innovation, for example, is the use of certain network properties and the OMIM (Online Mendelian Inheritance in Men) database [Hamosh et al., 2005] to rank genes with respect to their probability of being causative for a particular disease.

1.1 Protein-protein Interactions (PPIs) on the Molecular Level

Despite these successes, data on the molecular level remains largely incomplete. Precise interaction mechanisms are known for only a few PPIs (see e.g. Schmeing and Ramakrishnan [2009]). Binding position, orientation, strength, oligomeric context as well as binding-induced structural changes are mostly unresolved [Goh et al., 2004, Kastritis et al., 2011, Kastritis and Bonvin, 2010]. While evidence of functionally important alternative splicing events is mounting [Kalsotra and Cooper, 2011], knowledge about their impact on the level of PPIs is still an exception [Talavera et al., 2013, Wojtowicz et al., 2004]. Over the last years, structural biologists addressed some of these issue with an increased output of structurally resolved PPI complexes. In this context, computational biology has helped understanding PPIs with a series of quantitative analyses. By looking at their amino acid compositions, Ofran and

Rost found statistically significant differences in interfaces between proteins coming from the same gene and from different genes [Ofran and Rost, 2003b]; two manually curated data sets created by Mintseris and Weng [Mintseris and Weng, 2003, 2005] to distinguish between two-state and three-state complexes have facilitated a series of analyses focusing e.g. on the difference between permanent and transient or specific and unspecific interactions [Block et al., 2006, Lukatsky et al., 2007, Madaoui and Guerois, 2008, Bera and Ray, 2009]; related work investigated the co-evolution of protein-protein interfaces [Mintseris and Weng, 2005, Choi et al., 2009] and even the contribution of water to their binding modes [Teyra and Pisabarro, 2007].

We have complemented these studies by analyzing changes in the binding mode of two interaction partners (Chapter 2). This challenges the long standing assumption that, when measuring the same protein-protein interaction twice, the interface remains same. We collected all hetero-complexes from the Protein Data Bank (PDB) [Berman et al., 2000] and compared their interfaces over different levels of sequence divergence. Then we calculated the probability of an interface change and its severity. This revealed that even sequence-identical protein pairs can interact differently, contributing over one third to the interface variability observed in pairs from the same families (*Interologs*). One reason is oligomerization: the more copies of an interaction in a complex, the higher the variability. Often, multiple copies are introduced by homomers. Nevertheless, even dimeric complexes can have very different binding modes. These results withstand numerous interface similarity measures and the many challenges of the PDB. We performed our analysis after redundancy reduction, ensured that variability is not limited to certain subgroups and marginalized the influence of crystal contacts. Many databases have appeared in recent years that classify and cluster hetero-interfaces by families (e.g. *SCOPPI* [Winter et al., 2006]). They leave it open whether interface variability is caused by sequence changes or actually naturally encoded into a protein pair. We now demonstrate that different binding modes are often observed without any amino acid change and provide explanations by various follow-up analyses. Striking sample cases fall into diverse categories, ranging from protein conformation-specific binding modes to flexible interfaces that overcome steric constraints in the assembly of higher order complexes.

1.2 Sequence-based Residue-residue Interaction Prediction

One of the oldest applications of computational biology is ‘protein-protein docking’. This is the problem of predicting the structure of two proteins bound to each other, given the structures of the monomeric proteins. Knowing this binding mode enables various downstream analyses, e.g. targeting the suppression or enhancement of the PPI, for example via site-directed mutagenesis. This, in turn, is often a crucial step in drug development and can generally contribute to our understanding of the greater biological process the PPI is part of.

Protein docking can be divided into two different sub-categories, 'rigid body docking' and 'flexible body docking'. In rigid body docking, the structures of the monomers are assumed to stay the same upon binding to each other. Methods usually first perform a coarse-grained search in a large space of possible binding modes to identify the most likely candidates and then refine them. A popular approach for the former step is ZDOCK [Chen et al., 2003]. The latter can be addressed, e.g. by RosettaDock [Gray et al., 2003]. In particular in combination with additional experimental information, e.g. in the form of structures of interologous complexes or other experiments indicative of the binding mode, it appears from the latest independent assessments that the problem of rigid body docking can be solved quite accurately [Lensink et al., 2007].

The picture looks very different for flexible body docking, however. Often, the monomers undergo large conformational changes upon binding to each other. One example of a method trying to also account for such changes is FiberDock [Mashiach et al., 2010]. If no suitable template complex is available, however, today's methods usually fail to accurately predict the target complex [Lensink et al., 2007].

It has often been demonstrated that certain structural or even functional aspects of a protein can be predicted from sequence without prior knowledge about its structure and without a close and structurally resolved homolog [Rost and Liu, 2003]. This is in particular true for the residues capable of binding other proteins [Ofran and Rost, 2003a]. Often, such predictions even correlate with hot spot residues [Ofran and Rost, 2007b]. We therefore hypothesized that, using state-of-the-art sequence based prediction tools and enough computational power, it might be possible to also predict, from sequence alone, some of the residue pairs that are in contact in a PPI. The data needed for this task was readily available in form of the collection of PPI structures from the entire PDB that we created in our analysis on differential binding (Chapter 2). In Chapter 3, we describe the training and evaluation of such a predictor. For every residue pair, it encodes both amino acids with various sequence-based features, including predicted aspects of structure and function from the PredictProtein suite of programs [Rost and Liu, 2003]. We evaluate the resulting classifier via cross-validation on the training data and on non-redudant PPI structures recently added to the PDB. As a summary, even though not reaching high accuracies in absolute terms, it improves greatly over random and can thus be of valuable help for experimental determination of key residue-residue interactions.

1.3 Sequence-based Binary PPI Prediction

Not only data about the molecular details of PPIs is very limited, but also the information whether two protein interact or not ('binary interaction'). It is typically presented in the form of PPI networks, where each protein is a node and each edge indicates that the two proteins interact. In research dealing with PPI networks, the term 'interaction' has been used to describe various different phenomena, reaching from protein co-localization to functional association. Even though experimental data is often inconclusive about the true nature of an interaction, we want to emphasize that we still focus on direct physical interactions. Put simply, for us, two proteins actually have to physically touch each other to be interacting (e.g. both proteins must have at least 5 residue closer than 6Å to the other protein).

Such PPIs are supported by an increasing amount of inconclusive data that becomes meaningful only in concert. This implies that the raw data no longer holds the evidence. Instead, it needs to be connected through simple reasoning or more advanced combinations in Bayesian networks [Troyanskaya et al., 2003]. This blurs the line between experiment and prediction. For instance, if we use our definition of PPIs (physical contacts between different proteins), using expression data in a Bayesian network may decreases performance, unless we train another machine learning device (SVM) that 'predicts' PPIs only using expression data [Soong et al., 2008]. Many in-silico methods enrich PPI networks [Liu et al., 2008b, Lees et al., 2011, Mosca et al., 2013]. Data sources include protein sequences, structures, co-evolution, co-expression, domain co-occurrence, text-mining, subcellular localization and already known interactions (network topology). Model types range from Naive Bayes, to Support Vector Machines (SVM) and Conditional Random Fields and are typically combined with other tools such as homology-based inference or other sequence-derived features.

Homology-based inference describes the following rationale: Assume a detailed experiment establishes that protein P1 has function F and that protein P2 has significant sequence similarity to P1. Predicting P2 to also have function F constitutes homology-based inference. Interestingly, this approach is still one of the best ways to predict Gene Ontology terms [Radivojac et al., 2013, Hamp et al., 2013b], and in some cases it is even enough to outperform advanced methods for the prediction of subcellular localization (Goldberg et al., personal communication). For PPIs, however, it is substantially more challenging [Mika and Rost, 2006]. Advanced tools reaching into the twilight zone of sequence similarity [Rost, 1999] are therefore a requirement for successful sequence based interaction prediction.

Our first attempt at predicting whether two proteins interact was the residue-residue contact predictor developed in Chapter 3. The idea was to predict all interprotein residue-residue interactions for any given pair of proteins and to consider the score of the highest-scoring pair as the sole indicator of whether two proteins interact. This was superior to various other approaches such as considering the top-N hits or maximum likelihood based inference relying on the distribution of scores. As the PPIs in our data set were derived from the entire PDB, they involved various different organisms. Large differences between proteomes and interactomes demanded that negative interactions, i.e. non-interacting pairs of proteins, follow the same distribution of organisms as the positive PPIs. Our data set was too small for accurate taxonomy-based negative sampling, however, and we did not find a proper evaluation setup. For example, it is unclear whether redundancy between negative interactions is acceptable in general or only within or across organisms or among positive and negative interactions. As our results largely depended on such questions and we did not find answers in the literature, we decided to focus on the prediction of PPIs in human. Comparing our method to those recently assessed by [Park and Marcotte, 2012], however, we found its accuracy to be clearly inferior.

Our next and final approach focused on the evolutionary profile based kernel developed by the Leslie group [Kuang et al., 2004]. We discovered that - when applied properly - it achieves state-of-the accuracies for predicting the subcellular localization of a protein [Goldberg et al., 2012]. Initially, it was too slow for training with large-scale data sets and predicting new proteins, but we managed to accelerate it by orders of magnitudes. This procedure is detailed in Chapter 4. Accurate prediction of the subcellular localization of a protein alone helps to predict interaction partners by eliminating protein pairs that cannot interact from a compartmental perspective. However, the profile kernel can also directly be employed to predict whether two proteins are interacting or not. In particular in combination with the prediction of subcellular localization, this approach significantly outperforms all other sequence-based methods for difficult prediction cases. We demonstrate this in Chapter 5.

1.4 PPI Networks and Protein Function Prediction

With data missing on a large scale, the interplay between experimental verification and computational prediction of PPIs is iterative. In-silico predictors are trained with existing experimental data and output candidate residues, residue pairs, proteins or PPIs with a high probability of leading to new insights into molecular mechanisms or cellular processes in the wet lab. The final factor for the success of a computational method is therefore its availability and applicability by biologists in the context of focused, small-scale experiments. For PPIs, this often involves visualizing interactions as networks, followed by manual enrichment and analysis. Not surprisingly, one of the most successful tools in computational biology helps doing exactly this

(Cytoscape; [Shannon et al., 2003]).

For the network of interest, many downstream analyses are imaginable, involving for example node degree or path length distributions. Gaining insights into the functions of a network, however, also includes single-protein annotations. For example, grouping the nodes by their subcellular localization dissects the network into modules of interactions that happen at the same place, and thus presumably also in the context of the same process [Ofran et al., 2006]. A generalization of this idea is Gene Ontology (GO) [Ashburner et al., 2000] term enrichment, in which proteins are mapped to a controlled vocabulary of a wide variety of functions (not only, but including subcellular localization), followed by statistical analysis of GO terms that are overrepresented in certain regions of the network. A highly successful tool for this task is a plug-in for Cytoscape, named BiNGO [Maere et al., 2005].

The problem with GO terms, however, is the same as for PPIs, namely that only a tiny fraction of all proteins have experimental GO term annotations. Fast and reliable computational methods are therefore once again indispensable. As mentioned earlier, sequence similarity has been found to be the most informative source of information for function prediction in a recent independent assessment [Radivojac et al., 2013]. We participated in this assessment successfully with three simple homology-inference based methods developed by students of a lecture at the Technical University Munich. In Chapter 6, we show to combine them into one classifier that is fast, achieves state-of-the-art accuracy and outputs annotations that can easily be traced back to experimental information.

1.5 Contributions of This Thesis

In the following a summary of the scientific contributions of this thesis.

Chapter 1: On the Alternativeness of PPI Interfaces

In this chapter, we extract all high-resolution structures of PPIs from the Protein Data Bank and determine in how far the same two proteins interact differently with each other. Our assumption is that the intricate molecular details PPIs are crucial for function and that we would see no difference when measuring the same interacting protein pair twice. Our analysis takes various factors into account that could bias or otherwise influence the results, such as crystal contacts, the oligomeric context of a PPI or the interface similarity measure. We establish that some change in the interfaces occurs in almost one quarter of all interactions even for sequence-identical pairs of proteins and that this fraction increases to more than third for close homologs. Strong differences can be observed for 12-29% of all comparisons. One important factor was the number of copies of an interaction in the same complex. The more copies, the more different the interfaces. Even after correcting for this factor, however, a significant variability remained. We illustrate our results with various sample complexes.

Chapter 2: Predicting Residue-residue Contacts in PPIs

Using the data set created in the previous chapter, we train an artificial neural network based predictor that determines whether two amino acids from two different proteins will bond to each other upon complex formation. To this end, we first extract all pairs of contacting amino acid residues from structures of known PPIs. Non-interacting pairs are sampled randomly from the remaining non-contacting residues. Then, for each pair, we encode both amino acids with various sequence-based features, including predicted aspects of structure and function from the PredictProtein suite of programs [Rost and Liu, 2003]. We evaluate the resulting classifier via cross-validation on the training data and on non-redudant PPI structures recently added to the PDB. Despite low absolute precision, we find it to improve greatly over random and thus to be of great help for guiding site-directed mutagenesis experiments.

Chapter 3: Acceleration of the Original Profile Kernel

Our residue-residue based predictor trained in the previous chapter performed well on the level of residue pairs, but was not competitive for discriminating between interacting and non-interacting protein pairs. Initial experiments with the original profile kernel [Kuang et al., 2004] showed great potential for this task, but was too slow for large-scale applications. In this chaper, we show how to accelerate the original implementation by orders of magnitudes with various low-level and algorithmic optimizations.

Chapter 4: Improving Sequence-based Binary PPI Prediction

In this chapter, we apply our accelerated profile kernel to the binary prediction of PPIs in human. In order to improve our understanding of biological processes on the molecular level, we need to determine which proteins are interacting. The slow speed, inaccuracy and limited applicability of experimental methods leaves sequence based prediction of protein-protein interactions (PPIs) a crucial subject in computational biology. Using highly reliable human PPIs, we show how evolutionary profiles and subcellular localization increase precision over the state-of-the-art even for low recall levels. A new rigorous way to reduce protein-protein interaction redundancy reveals that only a fraction of available PPIs is needed to build more accurate classifiers. Two cross-validations differing in the similarity amongst non-interacting protein pairs investigate their impact on PPI prediction. We conclude by predicting all 200 million protein pairs in human and estimating their accuracy in terms of recall and precision.

Chapter 5: Homology-based Inference of Protein Function

Predicting PPIs is immediately linked to network visualization and functional analysis. One of the most-often used types of analysis is Gene Ontology (GO; [Ashburner et al., 2000]) term enrichment. Most proteins, however are not experimentally annotated and need to be predicted. In this chapter, we introduce a new sequence-based in-silico GO term predictor that is fast, achieves state-of-the-art accuracy and outputs annotations that can easily be traced back to experimental information.

Chapter 2

On the Alternativeness of PPI Interfaces*

2.1 Outline

The intricate molecular details of protein-protein interactions (PPIs) are crucial for function. Therefore, measuring the same interacting protein pair again, we expect the same result. This work measured the similarity in the molecular details of interaction for the same and for homologous protein pairs between different experiments. All scores analyzed suggested that different experiments often find exceptions in the interfaces of similar PPIs: up to 22% of all comparisons revealed some differences even for sequence-identical pairs of proteins. The corresponding number for pairs of close homologs reached 68%. Conversely, the interfaces differed entirely for 12-29% of all comparisons. All these estimates were calculated after redundancy reduction. The magnitude of interface differences ranged from subtle to the extreme, as illustrated by a few examples. An extreme case was a change of the interacting domains between two observations of the same biological interaction. One reason for different interfaces was the number of copies of an interaction in the same complex: the probability of observing alternative binding modes increases with the number of copies. Even after removing the special cases with alternative hetero-interfaces to the same homomer, a substantial variability remained. Our results strongly support the surprising notion that there are many alternative solutions to make the intricate molecular details of PPIs crucial for function.

*This chapter is based on the publication [Hamp and Rost, 2012]

2.2 Introduction

PPIs in high-resolution reveal molecular details of network edges. The study of high-resolution three-dimensional (3D) structures of proteins as deposited in the PDB, the Protein Data Bank [Berman et al., 2000], began with peptides [Kendrew et al., 1960, Perutz et al., 1960] and has increasingly included larger complexes of interacting proteins [Dutta and Berman, 2005]. These complexes, also referred to as protein-protein interactions (PPIs), capture the molecular details of interaction networks. The network view, in turn, has become increasingly important for, e.g., the ranking of genes according to their probability of being causative for a particular disease [Xu and Li, 2006, Chen et al., 2009, Taylor et al., 2009] as needed for Genome-wide Association Studies.

Despite this wealth of high-resolution interaction data, the set of interactions for which the exact molecular mechanisms are known remains immensely incomplete [Schmeing and Ramakrishnan, 2009] and with it experimental and computational descriptions of binding positions and binding-induced conformational changes [Goh et al., 2004, Kastritis and Bonvin, 2010, Kastritis et al., 2011]. Nevertheless, studies of available structures have shown that related proteins have similar binding sites [Zhang et al., 2010], that permanent and transient interactions differ so substantially from each other [Ofran and Rost, 2003b] that PPI hotspots can be predicted from sequence [Ofran and Rost, 2007a,b], and that we can accurately distinguish between specific and unspecific contacts [Mintseris and Weng, 2003]. Many others have addressed related tasks [Zhao et al., 2012, Mintseris and Weng, 2005, Res et al., 2005, Block et al., 2006, Lukatsky et al., 2007, Madaoui and Guerois, 2008, Bera and Ray, 2009, Choi et al., 2009], including even the contribution of water to the binding modes of PPIs [Teyra and Pisabarro, 2007].

Study of external PPIs from many new perspectives. An excellent recent work reviews various types of protein interactions [Keskin et al., 2008]. We want to complement it with a quantitative analysis of the interface variability of external interactions, i.e. interactions between two protein chains coming from different genes. These typically correspond to the edges in a PPI network. The atomic structures of their interfaces often seem to cluster into particular architectures [Mintseris and Weng, 2003, Tuncbag et al., 2008] and it has been suggested that they are conserved within and between organisms [Mika and Rost, 2006, Scott and Barton, 2007, Shin et al., 2009, Gophna and Ofran, 2010, Ranea et al., 2010, Nehrt et al., 2011]. Many authors have also analyzed the molecular details of binding within and between their domain families [Aloy et al., 2003, Ispolatov et al., 2005, Korkin et al., 2005, Kim et al., 2006, Shoemaker et al., 2006, Stein et al., 2009]. For example, they found that two different SCOP [Murzin et al., 1995] domain families exhibit more than one orientation of binding about 24% of the time [Kim et al., 2006]. Beside this number, however, only few more details were given about the underlying biological variety and in particular the causes of differential interfaces. The problem we see with this approach is that members of a SCOP family only share similar 3D structures and that the observed variability in binding might simply be explained by sequence variation. In fact, the inference of similarity in structure (homology modeling) is much more accurate than the inference of protein-protein interactions [Mika and Rost, 2006]. So far no studies based on significantly sized data sets have addressed the question to which extent the interface between two different proteins is biologically conserved, i.e. excluding diversity due to sequential differences.

Another challenge for the analysis of large-scale data sets have been crystal contacts and the difficulties of automated methods to correct such problems (e.g. the PQS [Henrick and Thornton, 1998] or the PISA [Krissinel and Henrick, 2007] service). Authors 'addressed' these problems by either entirely excluding different interfaces suspecting that those originated from non-biological contacts, or by leaving it open to which extent their results might have been created by such contacts.

Finding biologically relevant differences in binding modes. Here, we address both issues. First, we realized that the number and quality of author-assigned biological assemblies in the PDB now suffices to enable a quantitative study like this one. For the large majority of entries, the PDB now provides biologically relevant structures from the crystallographers themselves. Similar to PQS or PISA, they describe a complex as it occurs in the living cell. At the same time, however, they are more accurate and easier to verify than de novo predictions. Therefore, we did not discard any high-resolution complex or interface therein.

Secondly, put most extremely, we ask the question: if X-ray crystallographers measure the same interaction twice, do they get the same result? The main focus is first on the variability of the interaction between identical variants of the same two proteins (SameSeq). In other words, we look at external interactions corresponding to the same pair of protein sequences and estimate how often the interfaces are different (Fig. 1A; Fig. 1B: the red arrow compares two sequence-identical interactions). We then extend our analysis by allowing minor sequence variations in corresponding interactors (e.g. in the form of point mutations; SameProt). However, we still maintain the comparison between essentially the same proteins, because we make sure that a sequence change does not go hand-in-hand with a change of the original protein (Fig. 1B: for the blue interface comparisons, the sequences have changed [S1/S3 vs. S2/S3], but the original proteins [Px/Py] remained the same). Finally, we compared two external interactions corresponding to the same family pair, i.e. 'interologs' (Interolog). In a dimer-dimer comparison on this Interolog-level, corresponding interactors still had a similar 3D structure, but their sequences could be very different. (Several authors have been using the term ‘interolog’ [Walhout et al., 2000, Yu et al., 2004]; it has the advantage over the term ‘homolog’ that no evolutionary relation is implied in the definition; Fig. 1B, green: interfaces between proteins Px and Py are compared to those between Pz and Py).

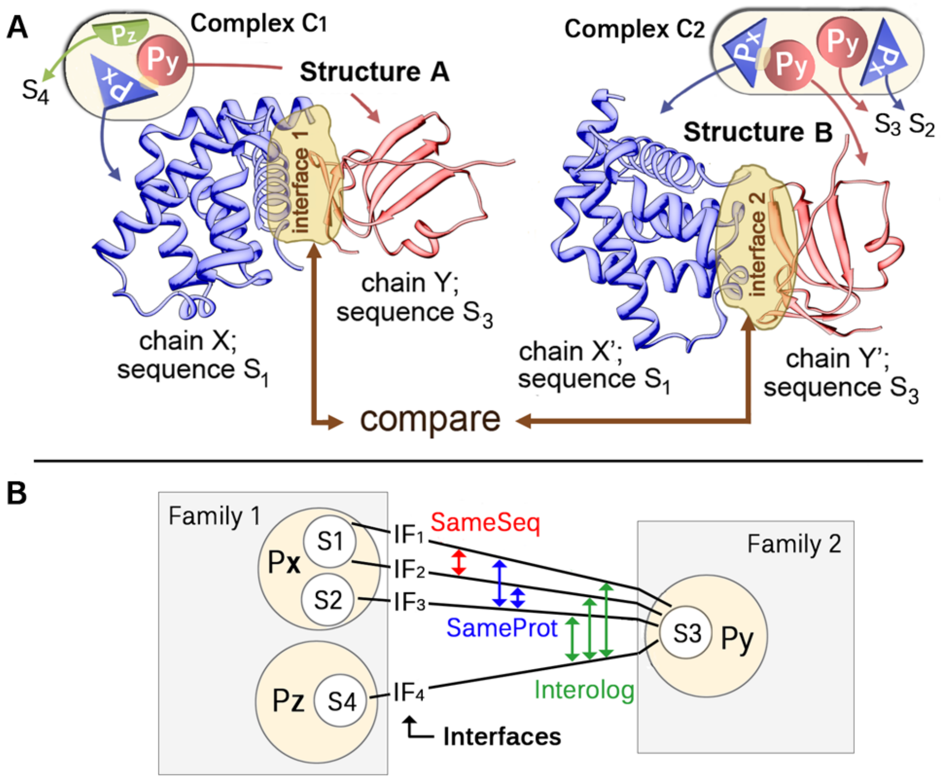


Figure 2.1: (A) Sketch for interface comparison. Two proteins Px and Py always interact in the same way, do they? We compared pairs of proteins for which we found several experimental solutions for their interaction. Assume that we have two high-resolution protein complexes C1 and C2. From these, we pick two hetero-dimers (Structure A and Structure B) for the interaction between proteins Px and Py (identified by the chains X and Y in Structure A, and by X' and Y' in Structure B). We then compared the interface of the same interaction between those two experimental solutions. **(B) PPI network induced by complexes C1 and C2.** Complexes C1 and C2 contain two protein-protein interactions: Px-Py and Px-Pz. We differentiated between three types of interface comparisons. First, we only compared interactions corresponding to same pair of sequences (SameSeq; red; shown in A). Then, sequences could change as long as the original proteins remained the same (SameProt; blue; interfaces S1/S3 are compared to S2/S3; both sequences S1 and S3 are variants of protein Px). Finally, we compared interologous interactions (green; interfaces Px/Py are compared to Pz/Py; Px and Pz come from the same family).

2.3 Methods

2.3.1 PPI Data Set from the PDB

Each node in a PPI network typically refers to a UniProt [Consortium, 2011] entry. While UniProt stores information about proteins, its first layer of organization is genetic: every entry corresponds to a unique location on a genome. Hence, in order to find reliable structural evidence of PPI network edges, we mined the PDB [Berman et al., 2003, Rose et al., 2010] for interacting proteins which map to different UniProt/Swiss-Prot [Schneider et al., 2009] identifiers. We extracted such external protein-protein interactions (i.e. interactions originating from two different genes) in the following way: first, we downloaded all author assigned biological assemblies from the PDB. We then retained only X-ray structures that had a resolution < 2.5 Å and mapped to at least two different UniProt/Swiss-Prot entries (author assignment available for 99% of all such structures). We primarily used the PDB<=>Swiss-Prot mapping provided by the PDB and only performed the following step if this mapping was not available: we BLASTed [Altschul et al., 1997] the PDB SEQRES sequence (at least 30 residues long) against the Swiss-Prot database, thresholding at E-Values $< 10^{-3}$ and requiring at least 90% of the PDB chain to be aligned. (When we found more than one hit, we took the one with the lowest E-Value; when we had none, we discarded this complex.) Having found those 'interesting' complexes, we extracted all interacting pairs of chains pointing to two different Swiss-Prot entries. At this early stage of our procedure, we only required one pair of atoms of the two chains to be closer than 0.6 nm (6 Å) in order to consider them interacting.

2.3.2 Interlude: Using PISA for Construction of Biologically Relevant Assemblies

An alternative that we considered was using the PISA [Krissinel and Henrick, 2007] service to obtain biologically relevant assemblies instead of author assigned complexes. In the following, we give reasons why we switched to author assigned complexes, an accuracy estimation of PISA in the context of hetero-complexes and other results compiled with the PISA based data set.

Crystallographic methods often do not allow the accurate determination of the biologically relevant protein assembly, especially in the case of larger complexes. Experimental ways to look at assemblies in the living cells (e.g. cryo electron microscopy) are limited. For our studies, we exclusively used author assigned biological assemblies as annotated in the PDB. In most cases, this means that some interfaces from the asymmetric unit (ASU) have been deemed crystallographic artifacts and that the original complex has been broken down into smaller fragments.

Actual State	Sensitivity	Incorrect States
1-mer	0/2	2 x 2-mer
2-mer	4/7	2 x 4-mer, 8-mer
4-mer	34/36	2 x 2-mer
6-mer	3/3	-
8-mer	1/1	-
9-mer	29/29	-
12-mer	31/32	6-mer
16-mer	0/2	2 x 8-mer
20-mer	0/1	18-mer
21-mer	4/4	-
24-mer	2/2	-
Overall Accuracy: 90.7±2.7% (108/119)		

Table 2.1: The accuracy of PISA evaluated with 119 heterocomplexes. "Actual State" refers to the actual oligomeric state of a given complex, "Sensitivity" to the fraction of correctly predicted complexes in the respective state and "Incorrect States" to the oligomeric states the complex was assigned to in case it was predicted incorrectly.

Crystallographic interfaces are often determined on the basis of, e.g., interface size (the smaller, the less likely to be relevant) or homology (e.g. a protein is similar to another tetramer, thus it is also a tetramer). The PDB provides biologically relevant complexes in the form of downloadable structures, besides the ASUs. They cover about 99% of all high-resolution PDB entries with external interfaces (data not shown) and appear to be quite reliable: in about 100 manual checks whether the author-assigned biological unit as deposited in the PDB was also described as such in the publication introducing the structure, we found only 1 clear mistake.

Despite this high accuracy and coverage, PISA, the successor of PQS [Henrick and Thornton, 1998], provides yet another view onto the PDB interactome: it cannot only differentiate between specific and non-specific interfaces, but also re-assemble the fragments of the ASU in order to build the most probable quaternary structure in the living cell. Hence, it might find assemblies which crystallographic methods miss due limitations of experimental methods. We took the opportunity of this work to study its accuracy in context of external interfaces, i.e. interfaces coming from different proteins. While in the majority of cases, there is no experimental data to verify its predictions (which is also why we did not use PISA for our final results), a number of highly accurate biological complexes were available in the PIQSI [Levy, 2007] database.

PIQSI is a manually curated database annotating the oligomeric state of around 15,000 PDB entries (May 2011). Each complex is given an error attribute indicating whether it has the correct oligomeric state. We only considered PIQSI complexes 'without errors', thus reducing the number of annotated complexes to 10,000. Unfortunately, only a small fraction of those were hetero-complexes, so that after intersecting PIQSI with our data, only 119 complexes remained for evaluation. Corresponding coordinate files were downloaded from the PIQSI website. We then compared the oligomeric state of each of those complexes to the most probable complex predicted by PISA. Results are given in Table 2.1 in a similar way as found in [Krissinel and Henrick, 2005].

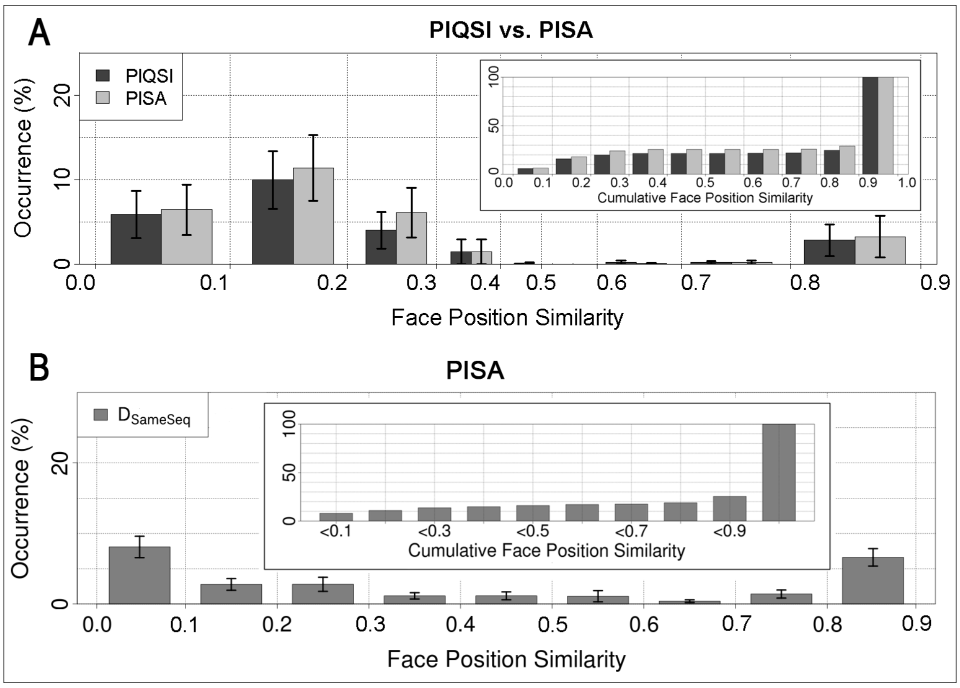


Figure 2.2: (A) Comparison of Face Position Similarity distributions derived from complexes common to our data and PIQSI. This panel shows the Face Position Similarity Distribution $D_{SameSeq}$ (c.f. Sections 2.3.6,2.3.7) when only comparing complexes common to both PISA and PIQSI. The range 0.9-1.0 is omitted in order to emphasize the other ranges. It is shown again in the inlet. **(B) Comparison of Face Position Similarity distribution $D_{SameSeq}$ on the PISA data set.** Here, we replaced all author assigned complexes in our data set with those predicted by PISA and re-calculated the Face Position Similarity distribution.

The evaluation reveals quite a high precision of PISA with an accuracy of 90.7%. This is in line with previous accuracy reports [Krissinel and Henrick, 2005]. Note that an erroneous prediction does not automatically lead to distorted similarity distributions for all external interactions of this complex as only a fraction of interfaces should undergo changes in the transition from the incorrect to the correct quaternary structure. Also notice that errors are not systematic over or underestimates of complex sizes so that missing or superfluous interfaces introduced by the wrong oligomeric states should roughly equal each other out with respect to complementarity. To validate these claims, we applied our clustering procedure (c.f. Section 2.3.6) to the PIQSI data set and compared the Face Position Similarity distributions (c.f. Section 2.3.7) to those derived from the PISA data set (distribution $D_{SameSeq}$; Fig. 2.2A). We did not require two or more PDB entries per external interactions, however, in order to increase the number of interactions for PIQSI. Thus, in absolute terms, curves should be biased towards low similarity.

Due to the small sample size, only a distorted curve could be derived in Fig. 2.2, but it becomes clear that interfaces are largely the same in both complex sets. Given these estimates, it is unlikely that the more accurate determination of the quaternary state of a protein will have a significant impact on the results. In fact, after replacing all author assigned PDB assemblies in our data set with the most probable PISA assemblies and re-filtering and re-clustering all interactions, the according face similarity distribution was mostly within the standard errors (Fig. 2.2B).

2.3.3 Definition of PPI Interfaces

Having found all structures of external interactions, we annotated their interfaces. Given a hetero-dimer with chains X and Y (X and Y come from different genes), we considered a pair of residues Rx and Ry as part of the interface if it contained at least one pair of atoms closer than 0.6 nm (6 Å) or if it met all three conditions: (i) both residues changed their accessible surface area upon binding (ΔASA: replacing the binding partner by water), (ii) Rx had no other interaction partner within 0.6 nm (6 Å), (iii) of all residues in protein chain Y that changed their accessible surface area (ASA), Ry was the closest to Rx. The latter included interactions that fell slightly above the 0.6 nm (6 Å) threshold but should still be considered interacting by their ASA change (we present a brief analysis of the effect of including ΔASA in the interface annotations in Section 2.4.1.1). We annotated each interface residue by two structural descriptors: ΔASA and d reflecting the distance (in Ångstrom) of the closest binding residue. Having defined all interface residues, we removed each hetero-dimer with fewer than five interacting residues on either chain from our data set.

2.3.4 Assignment of Copy Number to PPI Interfaces

Interface variability may be linked to the number the two proteins interact in the same complex. We prepared our data to verified this hypothesis in the following way: first, for a given interface, we determined its corresponding interaction and complex. In Fig. 2.1, for example, interface X-Y corresponds to interaction S1-S3 and complex C1. Then, we counted how often this interaction occurred in the complex (Fig. 2.1: S1-S3 occurs twice in complex C1) and assigned this number to the interface (Fig. 2.1: interface X-Y has copy number 2). This was repeated for all interfaces in our data set.

2.3.5 Measures for Face and Interface Similarity

Overall, we applied nine different interface similarity measures to our data, covering various types of changes. They are defined in detail in the following. The variety of these measures guaranteed that we captured as many aspects of ‘interaction similarity’ as possible. We found significant differences between these measures, but with respect to our overall conclusions, the reader may as well only consider the two most representative and intuitive measures, namely the Face Position Similarity and the L_rms. In the following, we refer to ‘interface’ as all the residues that ‘touch each other’ between two interacting proteins (Fig. 2.1), and as ‘face’ as all the residues on one side of the interface.

We compare the interface between chains C_X^A and C_Y^A in hetero-dimeric structure A with the interface between chains C_X^B and C_Y^B in hetero-dimeric structure B. X and Y indicate two different proteins (SameSeq, SameProt) or families (Interolog).

Accounting for Missing Residues. The atomic sequences of C_X^A and C_X^B are related, but not necessarily the same. In particular, parts of C_X^A might be missing in C_X^B and vice versa because of, e.g., experimental inaccuracies or evolutionary insertions and deletions (analogous statements for C_Y^A and C_Y^B). Those cases should not lead to low interface similarities as they should only be reported for actual binding mode changes. Consequently, we reduced A and B to common residues before comparing their interfaces. We found common residues in the following ways. (In case you want to skip the details of this procedure, simply assume that corresponding chains had the same number of residues with a 1:1 mapping between them and continue with Section 2.3.5.1)

Let $atomseq_X^A$ be the atomic sequence, $seqres_X^A$ the SEQRES sequence and sp_X^A the Swiss-Prot sequence of chain C_X^A. Let the same sequences be defined analogously for the other chains C_X^B, C_Y^A and C_Y^B.

If corresponding chains had the same SEQRES sequence (i.e. $seqres_X^A = seqres_X^B$), both of their atomic sequences $atomseq_X^A$ and $atomseq_X^B$ were semi-globally aligned to $seqres_X^A$ (BLOSUM62 as alignment matrix). Common residues could then be identified by identical positions in this 3-sequence alignment: all columns without gaps corresponded to common residues, all other residues were discarded. Analogous statements hold again for chains Y, but we will restrict ourselves to the case of X here and in the following for reasons of simplicity.

In case corresponding chains had different SEQRES sequences, but were variants of the same protein (i.e. mapped to the same Swiss-Prot entry), the atomic sequences $atomseq_X^A$ and $atomseq_X^B$ were first semi-globally aligned to the SEQRES sequences $seqres_X^A$ and $seqres_X^B$, respectively. Then we aligned both $seqres_X^A$ and $seqres_X^B$ to the representative Swiss-Prot sequence sp_X^A (Note that $sp_X^A = sp_X^B$ because C_X^A and C_X^B come from the same Swiss-Prot entry). In case we had to introduce gaps in $seqres_X^A$ during the alignment to sp_X^A, they were also added at the same positions in the $atomseq_X^A$ alignment, thus preserving each alignment and its length (analogous steps for B). In the end, common residues of $atomseq_X^A$ and $atomseq_X^B$ mapped to the same position in the Swiss-Prot sequence and were identified by gapless columns in the final 5-sequence alignment.

If the protein chains C_X^A and C_X^B came from different Swiss-Prot entries but the same family, the atomic residues were aligned to Swiss-Prot positions as before, i.e. $atomseq_X^A$ was aligned to $seqres_X^A$ and $seqres_X^A$ was aligned to sp_X^A, with analogous steps for B. Finally, we aligned sp_X^A and sp_X^B. In case of gaps in the alignment between sp_X^A and sp_X^B, they were added at according positions in both the atomic and SEQRES sequence alignments. In the final 6-sequence alignment, atomic residues mapping to the same position in this last alignment were considered common residues and could again be identified by columns without a gap in each of the 6 rows.

Note that structure alignments were not used for two reasons. Firstly, we wanted to safely identify missing residues. Using SEQRES and Swiss-Prot sequences somewhat improves the sensitivity in this context. Consider for example the case of C_X^A and C_X^B coming from different parts of the same gene (as it might happen for example after post-translational cleavage). A structure alignment might align C_X^A and C_X^B, find common residues and hence allow the comparison of their interfaces. The above procedure, on the other hand, correctly suggests that the two proteins have no common background and thus have to be excluded from the comparison. Secondly, we would have had to align structures before any pairwise interface comparison because the atomic sequences often differ slightly. This was unfeasible. Following the above procedure, we only had to align the atomic sequence to the SEQRES sequence once and could then infer common residues via this alignment.

2.3.5.1 Face Position Similarity

Outline. The Face Similarity tries to measure the conservation of face residues in both interfaces. For instance, assume that residues 1,2,3 interact on X, and residues 1,3,7,8 on X' (Fig. 2.1). The size of the intersection is then 2 (1, 3), i.e. the two faces on X and X' have two residue positions in common (residues 1 and 3). The average face size is 3.5=sqrt(3*4) (geometric mean) and the Face Position Similarity for X-X' becomes 2/3.5. The calculations of the same number for the pair Y-Y' yielded two values of Face Position Similarity. Among all the measures that we tried, the Face Position Similarity represented a good average interface similarity. Other measures were either more robust against smaller changes (e.g. Sphere Radius Ratio [Section 2.3.5.3]) or more sensitive, e.g., in terms of rotations (Interface Position Similarity [Section 2.3.5.5]) or side chain movements (Convex Hull Overlap; Section 2.3.5.4).

Formal definition. Face Position Similarity first creates four different residue sets $F_X^A, F_Y^A, F_X^B, F_Y^B \subset \mathbb{N}$ by determining the position of each interacting amino acid on X and Y for both A and B and then peforms comparisons between F_X^A and F_X^B and between F_Y^A and F_Y^B via s_{fps}:

$$\begin{aligned} s_{fps} &: 2^{\mathbb{X}} \times 2^{\mathbb{X}} \to [0,1] \\ (F_1, F_2) &\mapsto \frac{|F_1 \cap F_2|}{\sqrt{|F_1||F_2|}} \end{aligned} \tag{2.1}$$

where $\mathbb{X} = \mathbb{N}$ and F_1 and F_2 are two sets of face residue positions so that each of their elements points to exactly one amino acid on a common protein sequence. For simple single distributions (Section 2.3.7), the two similarities were subsequently arithmetically averaged. In the 2D plot (Section 2.3.7.2), they were treated separately so that only corresponding faces were compared.

2.3.5.2 Interface Position Similarity

Interface Position Similarity captures similarity analogously to Face Position Similarity, but also includes information about which particular pairs of amino acids interact. A and B are projected onto two sets $IF_A, IF_B \subset \mathbb{N} \times \mathbb{N}$ which both contain tuples of positions of interacting residues. An element $(i, j) \in IF_A$ for example indicates that in structure A, the i-th residue of X has contact with the j-th residue of Y. The similarity between IF_A and IF_B is calculated via s_{fps} with $\mathbb{X} = \mathbb{N} \times \mathbb{N}$ (see Eq. 2.1).

2.3.5.3 Sphere Radius Ratio

Sphere Radius Ratio first uses the residue position sets $F_X^A, F_X^B \subset \mathbb{N}$ as defined in Section 2.3.5.1 to specify two different face locations on chain C_X^A and stores the coordinates of the corresponding atoms in sets $K_A^{C_X^A}, K_B^{C_X^A} \subset \mathbb{R}^3$ (note that only one of the two sets contains atoms which are actually interacting in structure A). Their similarity is then calculated via s_{srr}

$$
\begin{aligned}
s_{srr} : 2^{\mathbb{R}^3} \times 2^{\mathbb{R}^3} &\rightarrow [0,1] \\
(K_1, K_2) &\mapsto \frac{r_{mbs}(K_1)}{r_{mbs}(K_1 \cup K_2)}
\end{aligned}
\tag{2.2}
$$

where $r_{mbs} : 2^{\mathbb{R}^3} \rightarrow \mathbb{R}$ returns the radius of the smallest sphere which encompasses all of the given coordinates.

Applying this procedure not only to C_X^A, but also to C_Y^A,C_X^B and C_Y^B, one obtains all in all four distinct similarities $s^{A_X}, s^{A_Y}, s^{B_X}$ and s^{B_Y}. As differences between similarities corresponding to the same chain but different structures (e.g. s^{A_X} and s^{B_X}) should mainly be attributed to backbone flexibilities, we always averaged those arithmetically, reducing the number of similarities for one pair of external interfaces to two. Analogously to Face Position Similarity, these were then again averaged when deriving 1D distributions (Section 2.3.7).

2.3.5.4 Convex Hull Overlap

Using $K_A^{C_X^A}$ and $K_B^{C_X^A} \subset \mathbb{R}^3$ as defined above, Convex Hull Overlap first calculates their convex hulls $H_A^{C_X^A}$ and $H_B^{C_X^A}$ and then defines s_{cho} as:

$$
\begin{aligned}
s_{cho} : \mathbb{P}_3 \times \mathbb{P}_3 &\rightarrow [0,1] \\
(H_1, H_2) &\mapsto \frac{vol(H_1 \cap H_2)}{max(vol(H_1), vol(H_2))}
\end{aligned}
\tag{2.3}
$$

where $\mathbb{P}_3$ is the space of polyhedra in $\mathbb{R}^3$, $\cap : \mathbb{P}_3 \times \mathbb{P}_3 \rightarrow \mathbb{P}_3$ a function which determines the intersection of two polyhedra and $vol : \mathbb{P}_3 \rightarrow \mathbb{R}$ a function which returns the volume of a polyhedron.

Analogously to Sphere Radius Ratio, creating the convex hulls and calculating s_{cho} not only for C_X^A, but also for C_Y^A,C_X^B and C_Y^B, one obtains four distinct similariy values for each pair of external interfaces. These were then averaged as in the previous Section. Convex Hulls were calculated with the QHull package [Barber et al., 1996].

2.3.5.5 Interface Composition Similarity

For both sets, $F_A = F_X^A \cup F_Y^A$ and $F_B = F_X^B \cup F_Y^B$ (Section 2.3.5.1), Interface Composition Similarity replaces each residue position $i \in F_A, F_B$ with the corresponding amino acid $aa_i \in \Sigma$, where Σ is the set of 20 amino acids. An interface can then be represented as a function $c_X : \Sigma \to \mathbb{N}$ giving the number of occurrences of each amino acid in an interface X. To compare interface compositions we define s_{aac} similarly to s_{fps}:

$$\begin{aligned} s_{aac} &: C \times C \to [0,1] \\ (c_1, c_2) &\mapsto \frac{\sum_{\sigma\in\Sigma} |c_1(\sigma) - c_2(\sigma)|}{\sqrt{\sum_{\sigma\in\Sigma} c_1(\sigma) * \sum_{\sigma\in\Sigma} c(\sigma)}} \end{aligned} \tag{2.4}$$

where C is the space of interface functions c_X.

2.3.5.6 Domain Number Ratio

We mapped each SCOP [Murzin et al., 1995] and CATH [Orengo et al., 1997] domain onto our sequences and counted the number of domains involved in the interaction for each external interface. A domain was involved if it contributed at least one binding residue to the interface. For each fully annotated pair of interfaces, we defined Domain Number Ratio as the ratio between the smaller and the larger of the two domain numbers.

2.3.5.7 Family Interaction Similarity

The interface of an external interaction not only corresponds to pairs of interacting residues as defined by Interface Position Similarity, but also to pairs of interacting protein families. Thus, we employed SCOP and the domain mapping of Domain Number Difference to study to what degree pairs of interacting families change when comparing two external interfaces. For each structure, we compiled the set of all family pairings, analogously to IF_A and IF_B of Interface Position Similarity (Section 2.3.5.2). The actual similarity could then be derived by s_{fps} with $\mathbb{X} = \mathbb{F} \times \mathbb{F}$, where $\mathbb{F}$ was the space of SCOP families .

2.3.5.8 RMSD

Actually not a similarity measure for interfaces, but rather for pairs of protein chains, we calculated the Root Mean Square Deviation (RMSD) of two binary heterocomplexes. To this end, we first separately superimposed C_X^A and C_X^B and then C_Y^A and C_Y^B. The average RMSD of both superpositions then represented the final similarity value which could be processed like any other similarity described before.

2.3.5.9 L_rms

Most notably used in the CAPRI [Lensink et al., 2007] experiments, this measure first optimally superimposes the two larger proteins under consideration ('receptors', i.e. either C_X^A and C_X^B or C_Y^A and C_Y^B) and then applies this transformation to the two smaller proteins ('ligands'). The classical Root Mean Square Deviation (RMSD) of the backbone atoms of the ligands is the L_rms. Note that this approach differs substantially from the other measures tested. Firstly, it returns a distance in Å instead of a similarity between 0 and 1. Secondly, it measures the displacement of the entire protein, not only the interface. Conformational changes of the ligands can lead to high distances as well as different binding positions. We still used this measure in order to link our results to related work.

2.3.5.10 I_rms

The I_rms is again a measure of the CAPRI experiments [Lensink et al., 2007]. First, we redefine the interface between C_X^A and C_Y^A: Now, every pair of residues with at least one atom pair closer than 10Å is part of the interface (before, it was 6Å). Then, we determine equivalent *interface* residues of A and B: we first create two sets of residue positions: $R^X_{commonIF} = F^A_X \cap F^B_X$ and $R^Y_{commonIF} = F^A_Y \cap F^B_Y$ (see 2.3.5.1 for definitions of the F sets). Then we reduce C_X^A and C_X^B to the residues in $R^X_{commonIF}$, with analogous steps for Y. In these new interface structures, we remove all non-backbone atoms.

This leaves us with two new structures which have the same amount of atoms. All these atoms are both part of the interface and the protein backbone. Next, we optimally superimpose the structures and calculate the RMSD. This RMSD is the I_rms. In case we could not find common interface residues before, we returned maximum RMSD.

2.3.5.11 Comparison of Measures

We have introduced a number of measures which capture different aspects of interface similarity. If they find dissimilarity, they assign lower values with different amplitudes, what makes them essentially incomparable in absolute terms. We can, however, compile a hierarchy of which measure sees differences more often than others based on the empirical results presented throughout the manuscript. We show it in Fig. 2.3. Here, we see that Convex Hull Overlap is the overall most sensitive face similarity measure. Only closely behind in terms of sensitivity follows L_rms, which is, however, strongly influenced by backbone movements of the entire proteins. The Face and Interface Position Similarity measures, coming next in the list, are both

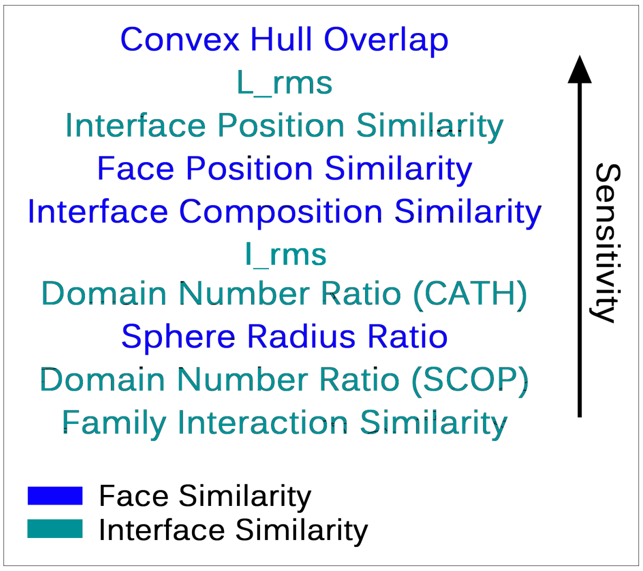

Convex Hull Overlap
L_rms
Interface Position Similarity
Face Position Similarity
Interface Composition Similarity
I_rms
Domain Number Ratio (CATH)
Sphere Radius Ratio
Domain Number Ratio (SCOP)
Family Interaction Similarity
Sensitivity
Face Similarity
Interface Similarity

Figure 2.3: A hierarchy of interface similarity measures. We have looked at the 0.9 to 1.0 bins of the $D_{SameSeq}$ distributions and used them to compile a hierarchy of measures based on sensitivity. We differentiate between measures that look at both sides of the interface at the same time or incorporate interfacial contacts (Interface Similarity) and those which first treat each side of the interaction separately and then average their conservations (Face Similarity).

exclusively residue based. Interface Position Similarlity takes into account conservartion of residue-residue contacts and is therefore more susceptible to change. The I_rms, on one hand, is in principle quite sensitive because we chose very fine grained RMSD thresholds (steps of 0.5Å). One the other hand, it misses dissimilarities due to the reduction to common interface residues and also similarities if no such residues could be found. The Interface Composition Similarity (residue based) and Sphere Radius Ratio (atom based) already fall into the class of rather robust measures. As a specialty, both almost never assign low similarities. In case of the Interface Composition Similarity, this is mainly due to random effects. Two interfaces usually have similar dimensions which is why Sphere Radius Ratio always sees some similarity. Finally, we have the domain based similarity measures. CATH domain assignments change slightly more often between interfaces than SCOP assignments. Obviously, entire SCOP domain families are even more conserved than domains.

2.3.6 Grouping Interfaces

Before we could apply the interface similarity measures to our entire collection of external interactions, we needed to group the structures so that we could differentiate between (and not mix) different types of sequence divergence when calculating interface similarity distribution. This also addressed the redundancy immanent in the PDB in the form of, e.g., overrepresented protein families. We hierarchically clustered the hetero-dimers over three levels, corresponding to increasing levels of sequence divergence. In the following, we describe this procedure in two ways. First, we provide an intuitive and easier to understand protocol and then a more formal definition.

2.3.6.1 Outline of the Procedure

First, we assigned two hetero-dimers to the same Level SameSeq group if they corresponded to same pair of SEQRES sequences (Fig. 2.1B: we add interfaces 1 and 2 to the same Level SameSeq group; other interfaces become single member Level SameSeq groups). Next, we reduced the influence of over-represented proteins. This was achieved by adding Level SameSeq groups to the same Level SameProt group if they corresponded to the same pair of associated Swiss-Prot identifiers (Fig. 2.1B: Level SameSeq groups S1-S3 and S2-S3 go into one Level SameProt group, S3-S4 to another). Clusters obtained in this way should represent the classical notion of edges and nodes in a PPI network. Our final Level Interolog addressed overrepresented families: we merged Level SameProt groups that pointed to the same pair of Pfam [Punta et al., 2012] families into one Level Interolog group (Fig. 1B: both Level SameProt groups are merged into one Level Interolog cluster; Fig. 2.4 for a graphical illustration of the clustering).

2.3.6.2 Formal Description

Let A be a hetero-dimer from our data set. It has two chains from two different proteins X and Y. We denote the two chains as C_X^A and C_Y^A. Let further $seqres(C_X^A)$ be the SEQRES sequence, $sp(seqres(C_X^A))$ the Swiss-Prot sequence and $pfam(sp(seqres(C_X^A)))$ the Pfam Punta et al. [2012] families of chain C_X^A. Let those sequences and families be defined analogously for the other chain C_X^B.

On the first clustering Level ('SameSeq'), we assign two hetero-dimers A and B (chains $C_{X'}^B$ and $C_{Y'}^B$) to the same cluster if they have the same pair of SEQRES sequences, i.e. $seqres(C_X^A) = seqres(C_{X'}^B)$ and $seqres(C_Y^A) = seqres(C_{Y'}^B)$. Consequently, we can represent a Level SameSeq cluster by a pair of SEQRES sequences $(seqres_i, seqres_j)$, because all the hetero-dimers in a cluster have exactly the same SEQRES sequence pair.

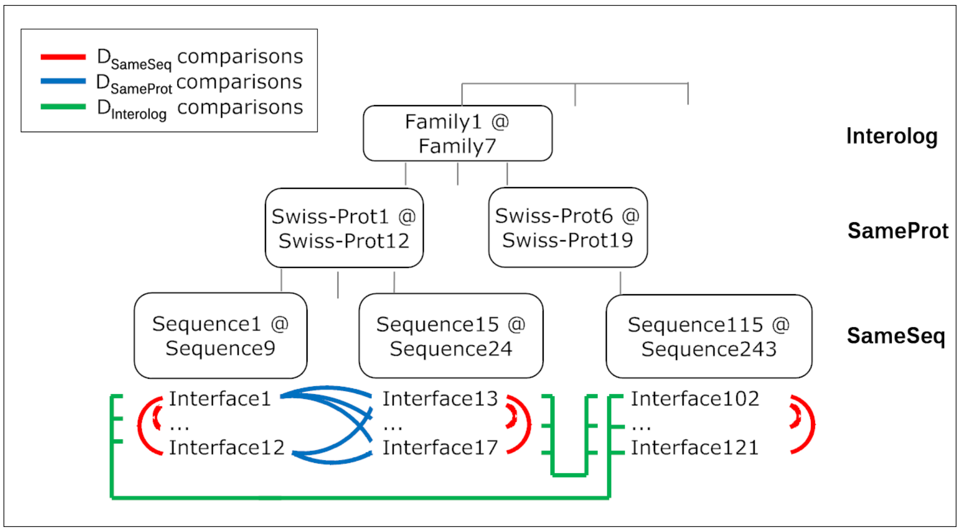

$D_{SameSeq}$ comparisons
$D_{SameProt}$ comparisons
$D_{Interolog}$ comparisons
Family1 @ Family7
Interolog
Swiss-Prot1 @ Swiss-Prot12
Swiss-Prot6 @ Swiss-Prot19
SameProt
Sequence1 @ Sequence9
Sequence15 @ Sequence24
Sequence115 @ Sequence243
SameSeq
Interface1 ... Interface12
Interface13 ... Interface17
Interface102 ... Interface121

Figure 2.4: Clustering procedure and calculation of similarity distributions. We clustered interfaces over 3 different levels. In the first level, 'SameSeq', interfaces coming from identical sequence pairs were grouped together. Level SameSeq clusters from the same Swiss-Prot pair ('protein pair') were then merged in the same Level SameProt cluster. Level SameProt clusters were grouped together into the same Level Interolog clusters if they came from the same pair of Pfam [Punta et al., 2012] families. We used this clustering to derive different interface similarity distributions, based on three different types of interface comparisons (c.f. Section 2.3.7): First, we compared only interfaces from the same pair of sequences (red; distribution $D_{SameSeq}$), then only those from the same pair of proteins (blue; distribution $D_{SameProt}$) and finally we only required them to come from the same family pair (green; distribution $D_{Interolog}$)

On the second clustering level ('SameProt'), each cluster consists of several SameSeq clusters. We merge two Level SameSeq clusters $(seqres_i, seqres_j)$ and $(seqres_{i'}, seqres_{j'})$ if they point to the same pair of Swiss-Prot entries, i.e. $sp(seqres_i) = sp(seqres_{i'})$ and $sp(seqres_j) = sp(seqres_{j'})$. We denote a SameProt cluster by its pair of Swiss-Prot entries (sp_k, sp_l).

In the third clustering Level ('Interolog'), finally, two Level SameProt clusters (sp_k, sp_l) and $(sp_{k'}, sp_{l'})$ are grouped together if they have the same Pfam [Punta et al., 2012] family composition, i.e. $pfam(sp_k) = pfam(sp_{k'})$ and $pfam(sp_l) = pfam(sp_{l'})$.

2.3.7 Interface Similarity Distributions

Without the grouping above, any distribution of pairwise interface similarities would have been highly dominated by large and well-studied complexes for which many structures are available. Avoiding this bias demanded to group PPIs differently (Levels SameSeq to Interolog) and also to embrace this alternative grouping when trying to infer biologically meaningful similarity distributions. While the following procedure successfully reduced the bias stemming from overrepresented sequences and sequence families, we deliberately left Level SameSeq clusters unchanged in the assumption that all binding modes are biologically meaningful and that eliminating this redundancy would remove more biology than noise. In the following, we again first give an informal and easy to follow description of the procedure. Then we define it in a more formal way. The calculation of standard errors and distributions for combinations of similarity measures are exclusively given in formal definitions.

2.3.7.1 Outline of the Procedure

Distribution $D_{SameSeq}$. The interface similarity distribution in a SameSeq group describes the variety of the binding modes of sequence-identical pairs of proteins (Fig. 2.1A; Fig 1B: red). We calculated this similarity distribution by using all pairwise interface similarities of the members of a SameSeq group: we estimated the probability that a similarity falls into a particular similarity range (e.g. 0.0 to 0.1) and repeated this for all similarity ranges. SameProt groups contain several SameSeq groups. Thus, for a SameProt group distribution, we averaged over the distributions of all its SameSeq groups. Correspondingly, Interolog distributions originate from averaging over the distributions of the member SameProt groups. Essentially, the above leaves us with many Interolog distributions. For a view of all those distributions, we simply averaged over all Interolog distributions to obtain distribution $D_{SameSeq}$. It can be interpreted in the following way: First, we randomly choose a family pair (Interolog group; e.g. globin - globin). From this family pair, we randomly pick a Swiss-Prot pair (SameProt group; e.g. bovine hemoglobin-A – bovine hemoglobin-B) and then a sequence pair (SameSeq group; e.g. wild-type bovine hemoglobin-A – wild-type bovine hemoglobin-B). Distribution $D_{SameSeq}$ then gives the chance that the similarity between two interfaces of this sequence pair ('two observations') lies in a specific similarity range. For example, $D_{SameSeq}$ may define the probability that two interfaces have a similarity between 0.1 and 0.2 to be 7%. Note that the role of SameProt and Interolog groups here is simply to reduce sequence redundancy. We are still only comparing interfaces between sequentially identical protein pairs. By leaving out Level Interolog for example, we would highly bias $D_{SameSeq}$ towards the globin family. The PDB not only contains many structures of one particular hemoglobin variant, but of many variants (for example from different organisms). Only by combining all these variants in one Interolog group and giving the distribution of this group the same weight as any other group, we limit the influence of hemoglobin. Similarly, combining SameSeq clusters in one SameProt cluster makes sure that one hemoglobin variant does not suppress the influence of others.

Distribution $D_{SameProt}$. Distribution $D_{SameSeq}$ compared observations of sequentially completely identical PPIs (Fig. 2.1A; 1B: red). This provided information about native interface variability, i.e. variability coming from environmental and conformational changes or simply different energetic minima between two proteins. The edge in a PPI network, however, allowed for some sequential variation as a node only referred to a gene, not to a specific gene product (Fig. 2.1B). The distribution $D_{SameProt}$ revealed how such modifications affected interface variability. First choose a family pair and then a Swiss-Prot pair. This yields several sequence pairs (SameSeq clusters) that all map to the same edge in a PPI network (Fig. 2.1B: S1/S3, S2/S3). Pick two of these pairs and compare all of their interface structures to derive a similarity distribution (Fig. 2.1B: blue). For example, we compared all wild-type bovine hemoglobin interfaces to those where residue 90 of subunit A was changed from H to Y (natural variant of bovine hemoglobin-A; P01966). Repeating this for all sequence pairs (e.g. for all bovine hemoglobin variants) we calculated many distributions. Averaging them yielded the overall Level SameProt distribution (e.g., the average interface variability of bovine hemoglobin). From here on, the same procedure as for $D_{SameSeq}$ applies: calculate all Level SameProt distributions of the parent Interolog cluster (e.g. distributions for all globin-globin clusters, not only bovine hemoglobine). Average over these to obtain the Interolog distribution; calculate and average all Interolog distributions. This yields the overall distribution $D_{SameProt}$. Put simply, it reflects the chance that two interfaces from the same PPI edge lie in a particular similarity range, given small sequence changes have occurred.

Distribution $D_{Interolog}$. Finally, we want to investigate the diversity of binding modes between proteins from the same family pair, but different gene pairs (Fig. 2.1B: Px/Py, Pz/Py; green). The procedure to derive this overall similarity distribution $D_{Interolog}$ is analogous to that for $D_{SameProt}$: first, choose a family pair (e.g. cyclin - protein kinase) and two of its Swiss-Prot pairs (e.g. cyclin E1 – protein kinase 2 and cyclin B1 – protein kinase 2). Then, pick one sequence pair from each Swiss-Prot pair (e.g. the wild-type variants), compare all of their interfaces and calculate the Level SameSeq distribution. Repeat this for all sequence pairs from the two Swiss-Prot pairs and obtain the Level SameProt distribution by averaging over all

Level SameSeq distributions. This again is repeated for all possible Swiss-Prot pair combinations in order to derive the Level Interolog distribution. Finally, average over all single Level Interolog distributions in order to derive the overall distribution $D_{Interolog}$. Put simply: randomly choosing a pair of interacting families and then two of its protein pairs, $D_{Interolog}$ gives the chance that a typical comparison of their interfaces will result in a particular similarity. (Note that this procedure quickly leads to unfeasible amounts of interface comparisons. We have therefore limited the number of protein pairs per family and the number of sequence pairs per protein pairs to a maximum of 50.)

2.3.7.2 Formal Description

In the following, we formally describe the calculation of interface similarity distributions. Mathematically, they fall into the category of so-called 'finite mixture distributions', i.e. weighted averages over many uncorrelated individual distributions. We reduce the influence of the redundancy found in the PDB by giving the same weight not only to sequence pairs, but also to protein pairs and eventually family pairs.

Distribution $D_{SameSeq}$. Here, we describe how we can use the clustering to derive non-redundant interface similarity distributions. Let $Cl_x \in \{Cl_1, \ldots, Cl_s\}$ be the set of Level SameProt clusters in Level SameFam cluster x, $Cl_{x,y} \in \{Cl_{x,1}, \ldots, Cl_{x,t}\}$ the set of Level SameSeq clusters in Level SameProt cluster (x,y) and $EI^1_{x,y,z}, \ldots, EI^v_{x,y,z}$ the external interfaces of Level SameSeq cluster (x,y,z). We first calculate the set of pairwise similarities $S_{x,y,z} = \{sim(EI^i_{x,y,z}, EI^j_{x,y,z}) \mid i \neq j \wedge i,j \in \{1,\ldots,v\}\}$ where $sim(EI^i_{x,y,z}, EI^j_{x,y,z}) \in [0,1]$ was the result of one of the similarity measures as described in Section 2.3.5 and $EI^i_{x,y,z}$ always came from a different PDB entry than $EI^j_{x,y,z}$. We then defined a discrete probability distribution $P_{x,y,z}(X \in [a_k, a_{k+1}]) = \frac{|\{s|s \in [a_k, a_{k+1}] \wedge s \in S_{x,y,z}\}|}{|\{S_{x,y,z}\}|}$ with $a_k = \frac{k}{n}, k \in (0,1,\ldots,n-1)$ and n typically set to 10, which gave the chance of the interface similarity lying between a_k and a_{k+1} after randomly picking two structures corresponding to the pair of protein sequences given by $Cl_{x,y,z}$ (Note that this is essentially a maximum likelihood estimation for the unknown parameters p of typically $n = 10$ different Bernoulli-distributed random variables). Repeating this procedure for all Level SameSeq clusters in $Cl_{x,y}$ leads to the i.i.d set $\{P_{x,y,1}, \ldots, P_{x,y,|Cl_{x,y}|}\}$ and subsequently to the Level SameProt similarity distribution $P_{x,y}(X \in [a_k, a_{k+1}]) = \frac{1}{|Cl_{x,y}|}[P_{x,y,1}(X \in [a_k, a_{k+1}]) + \cdots + P_{x,y,|Cl_{x,y}|}(X \in [a_k, a_{k+1}])]$. We obtain P_x in the same way as $P_{x,y}$, i.e.: $P_x(X \in [a_k, a_{k+1}]) = \frac{1}{|Cl_x|}[P_{x,1}(X \in [a_k, a_{k+1}]) + \cdots + P_{x,|Cl_x|}(X \in [a_k, a_{k+1}])]$. Finally we define the overall distribution $D_{SameSeq}$ as: $D_{SameSeq}(X \in [a_k, a_{k+1}]) = \frac{1}{s}[P_1(X \in [a_k, a_{k+1}]) + \cdots + P_s(X \in [a_k, a_{k+1}])]$.

Distribution $D_{SameProt}$. Now, we compare interfaces coming from different Level SameSeq clusters. Given two Level SameSeq clusters $Cl_{x,y,z}$ and $Cl_{x,y,z'}$ with interfaces $EI^1_{x,y,z}, \ldots, EI^v_{x,y,z}$ and $EI^1_{x,y,z'}, \ldots, EI^{v'}_{x,y,z'}$, we define the pairwise similarities as $S_{x,y,(z,z')} = \{sim(EI^i_{x,y,z}, EI^j_{x,y,z'}) \mid i \in \{1, \ldots, v\} \wedge j \in \{1, \ldots, v'\}\}$. The corresponding distribution is then defined as $P'_{x,y,(z,z')}(X \in [a_k, a_{k+1}]) = \frac{|\{s | s \in [a_k, a_{k+1}] \wedge s \in S_{x,y,(z,z')}\}|}{|\{S_{x,y,(z,z')}\}|}$ with $a_k = \frac{k}{n}, k \in (0, 1, \ldots, n-1)$. Consequently, we can calculate the Level Same-Prot distribution $P'_{x,y}(X \in [a_k, a_{k+1}]) = \frac{1}{|Cl_{x,y}|(|Cl_{x,y}|-1)}[P_{x,y,(1,2)}(X \in [a_k, a_{k+1}]) + \cdots + P_{x,y,(1,|Cl_{x,y}|)}(X \in [a_k, a_{k+1}]) + \cdots + P_{x,y,(|Cl_{x,y}|,|Cl_{x,y}|-1)}(X \in [a_k, a_{k+1}])]$. Substituting P with P', we obtain P'_x in the same way as P_x in $D_{SameSeq}$, and ultimately also the overall distribution $D_{SameProt}$ in the same way as $D_{SameSeq}$.

Distribution $D_{Interolog}$. Finally, we compare interfaces coming from the same Level Interolog cluster x, but different Level SameProt clusters (x, y) and (x, y'). To this end, we define a Level SameProt similarity distribution $P''_{x,(y,y')}(X \in [a_k, a_{k+1}]) = \frac{1}{|Cl_{x,y}||Cl_{x,y'}|}[P''_{x,(y,1),(y',1)}(X \in [a_k, a_{k+1}]) + \cdots + P''_{x,(y,1),(y',|Cl_{x,y'}|)}(X \in [a_k, a_{k+1}]) + \cdots + P''_{x,(y,|Cl_{x,y}|),(y',|Cl_{x,y'}|)}(X \in [a_k, a_{k+1}])]$ where
$P''_{x,(y,z),(y',z')}(X \in [a_k, a_{k+1}]) = \frac{|\{s | s \in [a_k, a_{k+1}] \wedge s \in S_{x,(y,z),(y',z')}\}|}{|\{S_{x,(y,z),(y',z')}\}|}$ with
$a_k = \frac{k}{n}, k \in (0, 1, \ldots, n-1)$ and
$S_{x,(y,z),(y',z')} = \{sim(EI^i_{x,y,z}, EI^j_{x,y',z'}) \mid i \in \{1, \ldots, |Cl_{x,y,z}|\} \wedge j \in \{1, \ldots, |Cl_{x,y',z'}|\}\}$.
Before, we used the set of distributions defined as $P_{x,y}$ in order to calculate P_x. We now replace this set with the new newly derived $P''_{x,(y,y')}$ distributions and obtain P''_x. Performing analogous steps for the overall distribution, we calculate $D_{Interolog}$.

Standard Errors. We calculated standard errors of distributions $D_{SameSeq}$ to $D_{Interolog}$ with a multi-level bootstrapping approach. From the n Interolog clusters which contributed to a D distribution, we first re-sampled with replacement until we had a new list of n Interolog clusters (in which some entries might have been duplicates). In other words, we bootstrapped the Interolog clusters. For each Interolog cluster in this bootstrap, we then bootstrapped its SameProt clusters. For each of the SameProt clusters in this sub-bootstrap, we bootstrapped the SameSeq clusters and finally the interface similarities in each bootstrapped SameSeq cluster. Thus, in the end, our overall bootstrap was a re-sampling over all Levels and clusters. Next, we re-calculated the D distributions with this bootstrap and saved it. Then, we repeated all of the above 200 times.

Ultimately, we had 200 different estimates for each of the 10 bins in a D distribution. The standard error being defined as the standard deviation of a target statistic and the occurrence of a bin being this target statistic, the standard deviation of a bin over those 200 estimates defined its standard error. Note that this procedure is independent of the number of bins: for example, if we are interested in the occurrence of the range 0.0 to 0.5, we only have to change the n parameter for the D distribution to 2 (creating two bins: 0.0 to 0.5 and 0.5 to 1.0) and repeat all of the above.

Cross-correlating Distributions. In order to capture interface differences with two measures simultaneously, we compiled so-called '2D distributions'. In this context, the function $sim(EI^i, EI^j)$ of $D_{SameSeq}$ no longer returned a single number, but a triple, with both elements coming from different similarity measures. Subsequently, we redefined:

$$P_{x,y,z}(X \in ([a_k,a_{k+1}],[a_l,a_{l+1}])) = \frac{|\{(s_1,s_2)|s_1 \in [a_k,a_{k+1}] \wedge s_2 \in [a_l,a_{l+1}] \wedge (s_1,s_2) \in S_{x,y,z}\}|}{|\{S_{x,y,z}\}|}$$

with

$$a_k = \frac{k}{n}, a_l = \frac{l}{n}, k,l \in (0,1,...,n-1) \text{ and } n \text{ typically set to 10.}$$

2.3.8 Analyzing the Influence of Homo-oligomeric Assemblies

The same proteins may aggregate to form a homo-oligomer and bind as such a complex to another protein. In this case, the other protein often 'sees' different parts of the homomeric chain, resulting in very different external interfaces. For example, a homo-dimer with chains X_1 and X_2 might bind to another chain Y with two different interfaces (Fig. 2.5). Hence, we will have two hetero-dimers X_1/Y and X_2/Y with low interface similarity. As it can be argued whether both of these interfaces should be considered as one big interface or treated separately (c.f. Discussion), we analyzed their influence on the distributions $D_{SameSeq}$ to $D_{Interolog}$. To this end, we defined homo-oligomers in two different ways. Firstly, we used the classical notion, namely that all chains of a homomer have the same SEQRES sequence. Secondly, we introduced 'structural homomers' as interacting chains from the same family. This corresponded to all complexes that look homo-oligomeric on a structural level (low RMSD; c.f. Section 2.8), but can differ in sequence.

Consequently, when comparing two interfaces X/Y and X'/Y' from two different PDB entries, it was checked whether or not one of the chains X' and Y' were part of homo-oligomers (i.e. whether there were homomers $X'/X'_1/\ldots/X'_n$ or $Y'/Y'_1/\ldots/Y'_m$) and whether or not these homo-oligomers had other external interfaces with the same interaction partner as in X'/Y' (i.e. whether X' had interfaces with $Y'_1/\ldots/Y'_m$ or Y'

interfaces with $X'_1/\ldots/X'_n$). Having identified the set of all those sequence- or family-identical interfaces (including X'/Y'), they were compared to X/Y. Only if X/Y <=> X'/Y' was the best match among all alternatives, the corresponding similarity was used. Otherwise, the entire comparison was discarded (Fig. 2.5.)

Eventually, the roles of X/Y and X'/Y' are switched, and the procedure is repeated because all interfaces are compared with all others in the distributions $D_{SameSeq}$ to $D_{Interolog}$. In this way asymmetries arising from only considering the oligomeric context of one side of the comparison were filtered out.

We applied 'structural homomerization' only in the context of $D_{Interolog}$. For the two other distributions, it would have led to comparisons of interfaces between different protein pairs, thereby invalidating the constraints of these distributions. Also note that the above definition only allowed for comparisons of interactions between two different families.

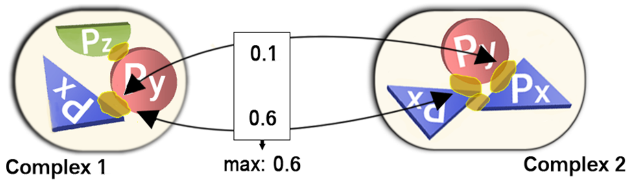


Figure 2.5: Filtering out interface diversity introduced by homomers. Assume you want to compare an interface Px-Py in complex 1 to the interfaces in complex 2. Usually, you will calculate two similarities (0.1 and 0.6), because there are two Px-Py interfaces in complex 2. Looking for homomers, you will find the two sequence identical Px chains in complex 2 interacting and connecting the two Px-Py interfaces. Now, you can correct the comparison by using only the one best match (0.6). The comparisons of the 'worse' alternatives are discarded.

2.4 Results

Our complete data set of external protein-protein interactions (PPIs) comprised 37,338 hetero-dimers. We grouped and filtered them on three different levels with decreasing sequence redundancy (Section 2.3.6). For instance, the first clustering level had 634 groups that contained sequence-identical hetero-dimers from at least two different high-resolution PDB entries. We compiled various statistics on this data set, including the distribution of cluster sizes on each clustering level, of oligomeric states, interface sizes and even of the conservation of protein function (Section 2.4.1).

In order to capture different facets of interface similarity, we introduced and evaluated nine different similarity measures (Section 2.3.5). Our main focus lies on two measures that we considered most important in hindsight (Face Position Similarity [Section 2.3.5.1] and L_rms [Section 2.3.5.9]). The first measure (Face Position Similarity) was most representative for all other seven measures while the second (L_rms) enabled direct comparisons of our results to related work, e.g. to the CAPRI [Lensink et al., 2007] experiments. For each measure, we used our clustering to calculate three different interface similarity distributions, corresponding to increasing levels of sequence divergence between interactions ($D_{SameSeq}$ to $D_{Interolog}$; Section 2.3.7). These distributions constitute the main result of this chapter. They were calculated such that all proteins and families of our data set contributed equally, regardless of their respective over-representation in the PDB. Finally, we measured how the distributions change when excluding the interface variability introduced by a protein binding differently to the same homomer. We give a short summary of this after the presentation of the unmodified distributions in Section 2.4.3.1.

2.4.1 Data Set Analysis

2.4.1.1 Influence of Initial Parameters

In related publications, there are usually a number of ad-hoc decision when selecting structures and defining interfaces. To exclude experimental or sampling bias in this context, we investigated the influence of some of the arguably most crucial alternatives: including structures with a resolution above 2.5Å; setting the minimal distance of two residues to be considered interacting to 4Å instead of 6Å; using ΔASA (change of accessible surface area) as a means to correct for interacting residues slightly above a given distance cutoff; using level SameSeq clusters with less than 5 members instead of 5 or more members.

While the first two points can be commonly encountered, the use of ΔASA when defining interfaces (as opposed to faces) was mainly an effort to retain consistency with the PISA service, which exclusively uses ΔASA to define face residues. The use of clusters with less than 5 members finally was a test in how far cluster size influences the expected similarity of two interfaces.

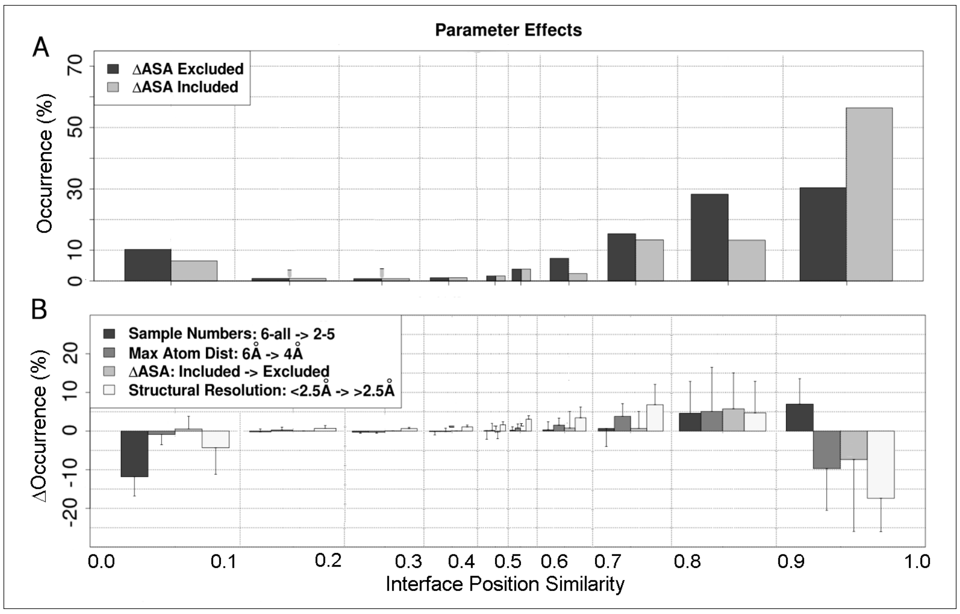


Figure 2.6: The effect of different parameters on the similarity distribution of interfaces. Both curves were derived with the Interface Position Similarity measure (Section 2.3.5.2). **(A)** An example of a distribution change when altering a single parameter value. It corresponds to switching the inclusion of ΔASA on and off when considering large complexes with resolutions above 2.5Å and an interaction distance cutoff of 4.0Å. **(B)** The average (thick bars) and maximum (error bars) differences for each parameter and bin with respect to all parameter combinations.

We interpreted the options above as parameters which can adopt two distinct values, on and off, and used one of our most sensitive similarity measures, Interface Position Similarity (Section 2.3.5.2), to assess their effect. For a particular parameter value combination, we first calculated the discrete probability distribution $D_{SameSeq}$ (Section 2.3.7). Put simply, this corresponds to the average distribution of pairwise interface similarities for identical protein pairs in a redundancy reduced version of the PDB. This procedure was repeated for all possible value combinations, resulting in $2^4 = 16$ different distributions. One could then create 8 pairs of distributions for each parameter p_i by only changing the value of p_i and keeping the other parameters fixed. For example, there were 8 combination with low-resolution structures included and 8 where they were excluded, thus creating 8 pairs of corresponding distributions. The mean and maximal change in distribution for each parameter value and bin then allowed to combine everything in one image (Fig. 2.6). To stay with the example of including low-resolution structures, a mean change of 5.0 for this option in a particular value range meant that the probability of observing such an interface similarity in this range on average changed by 5.0 percent when this option was turned on. Note that this was an exclusively graphical approach to estimate effects, as already changing the number of bins could lead to different values. Since the presentation of results will be kept in this form throughout the rest of this paper, however, it was sufficient in our case.

According to Fig. 2.6, each parameter can have a considerable effect. Including ΔASA, using 6Å instead of 4Å and high instead of low resolution structures generally stabilized interfaces, i.e. similarities shifted from intermediately high ranges (0.7 to 0.9) to very high conservation (0.9 to 1.0). A different phenomenon arises when using small instead of large clusters: interfaces of small clusters with only 5 or less members tend to be less complementary and more conserved, thus supporting the hypothesis that big complexes encourage alternative binding modes for the same pair of proteins. As they produce more binary structures of the same external interaction, they tend to be found in larger clusters. This is studied in more detail in Section 2.4.3.2.

2.4.1.2 PPI Data Set Properties

Cluster and Complex Sizes. In order to give a better picture of the data at hand, we compiled histograms of the number of protein chains per complex, external interfaces per Level SameSeq cluster, Level SameSeq clusters per Level SameProt cluster and Level SameProt clusters per Level Interolog cluster (Fig. 2.7; Section 2.3.6 for a description of the clustering procedure). All plots were calculated with the final data set after filtering (exclusion of bad resolution complexes, small interfaces, ...; see Section 2.4.1.1)

The curves in Fig. 2.7 display typical exponentially decreasing distributions. The distribution of oligomeric states and heterodimers per Level SameSeq cluster are influenced by a natural preference for symmetrical assemblies so that counts for even numbers are overrepresented. Additionally, some larger Level SameSeq cluster sizes are quite frequent (not shown). An over-representation of size 17 clusters, e.g., stems from a significant abundance of Cytochrome C Oxidase in the PDB. The effect is already remedied in the second clustering level, however, where external interactions are grouped according to the proteins involved.

Interface Sizes. Next, we analyzed the size of interfaces. To this end, we first counted the number of residue-residue interactions in each interface. Then, we calculated the distribution of interface sizes individually for each Level SameSeq cluster. These distributions were subsequently normalized in the same way as a

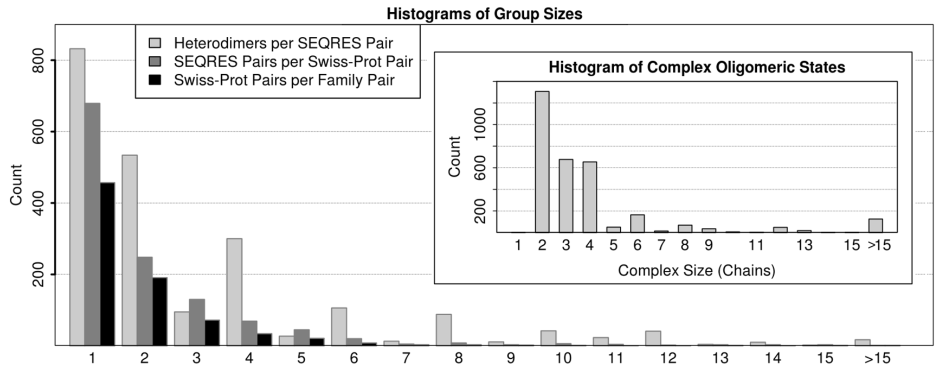


Figure 2.7: Histograms of various properties of our final data set. We show the distribution of the number of binary interactions per Level SameSeq cluster (heterodimers per SEQRES pair), the number of Level SameSeq clusters per Level SameProt cluster (SEQRES pairs per Swiss-Prot pair) and the number of Level SameProt clusters per Level Interolog cluster (Swiss-Prot pair per Pfam [Punta et al., 2012] family pair). The inlet additionally displays the histogram of the complex oligomeric states, i.e. the number of protein chains per complex.

$D_{SameSeq}$ distribution (Section 2.3.7), i.e. we first normalized for overrepresented sequences, then for proteins and then for families. The final distribution is presented in Fig. 2.8A.

We observe quite a far stretched distribution of interface sizes, as some population exists even beyond 400 residue-residue contacts. Compiling the data with bin size 100, it corresponds to an exponentially decreasing curve, i.e. the smaller the interface, the more frequent. Looking closer at interfaces with 0 to 100 contacts, however, (Fig. 2.7A, inlet), we see a peak in the range from 40 to 60 residues. The bin from 0 to 20 contacts only plays a minor role.

Structural Similarities. Then, we looked at the Root Mean Square Deviations (RMSDs). As the RMSD typically involves two structures, we implemented it as a standard similarity measure: first, we split the two hetero-dimers under consideration into their four chains. Then, we superimposed corresponding chains, calculated the two RMSDs and returned the average of both (Section 2.3.5.8). This allowed us to embed the RMSD into our evaluation framework and observe how values change across different types of comparisons ($D_{SameSeq}$ through $D_{Interolog}$; Section 2.3.7). It featured comparisons within clusters (e.g., we calculate one distribution for each Level SameSeq cluster) and across clusters (final distributions $D_{SameSeq}$ to $D_{Interolog}$ are averages over within-cluster distributions). See Fig. 2.8B for results.

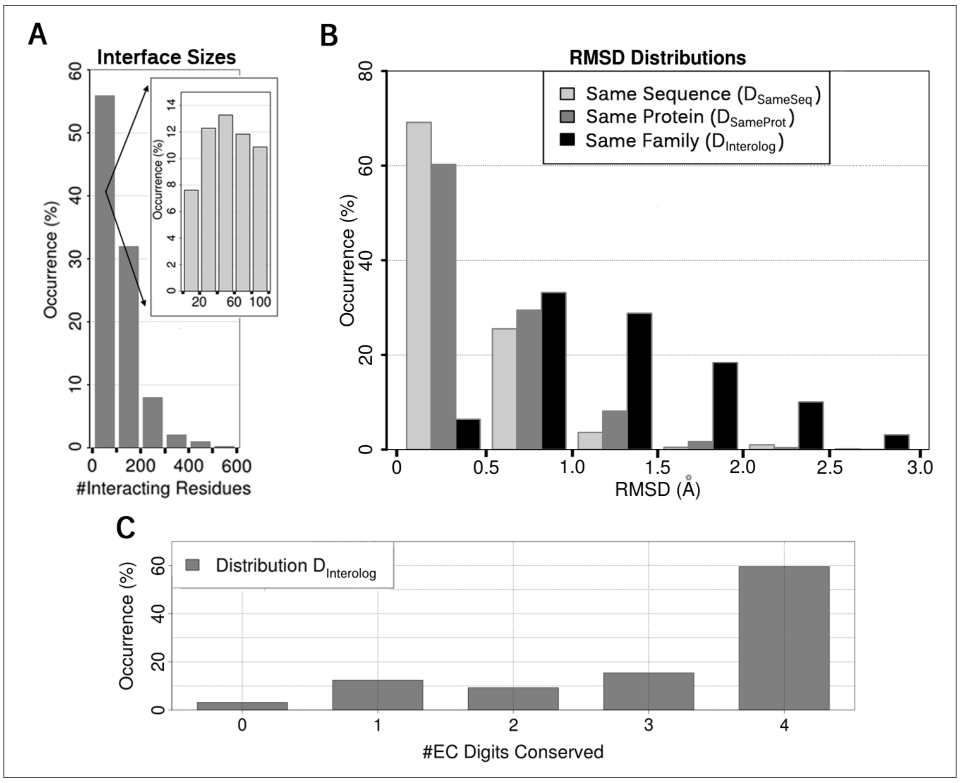


Figure 2.8: (A) Distribution of interface sizes. We counted the number of residue-residue contacts per interface and calculated the distribution of interface sizes for each Level SameSeq cluster. In this way we could derive distribution $D_{SameSeq}$ like for any other similarity measure (Section 2.3.7). Subfigure (a) shows this distribution first on a rather coarse scale (bin size 100). As it peaks between 40 and 60 contacts, we additionally zoom into the first bin (inlet). **(B) Distributions of RMSDs.** We interpreted the classical Root Mean Square Deviation as a similarity measure and superimposed corresponding chains when comparing two binary heterocomplexes in a Level SameSeq cluster. The two resulting RMSD values were always averaged. This allowed us to calculate $D_{SameSeq}$ through $D_{Interolog}$ (Section 2.3.7). **(C) Function Conservation within Level Interolog clusters.** We analyzed how often and to what degree the EC number between two proteins from the same Pfam family is conserved. The distribution shows the average over all families after averaging the pairwise conservations within each family. This corresponds to the procedure used to calculate the $D_{Interolog}$ distribution in the context of interface similarity measures (Section 2.3.7).

The RMSD distributions impressively show the effects of sequence variations.

Comparisons between proteins with the same sequence ($D_{SameSeq}$) most often result in very low RMSD values (0.0Å to 0.5Å). Occurrence of higher values then decreases exponentially and disappears beyond 1.5Å. The distribution of RMSDs between chains from the same protein but different sequences ($D_{SameProt}$) exhibits a clear decrease of high similarity (0.0Å to 0.5Å) and an increase of other ranges up to 2.0Å. When comparing chains from different proteins but the same family, finally, occurrence peaks in the range from 0.5Å to 1.0Å and then steadily declines. As also this distribution stops early for values above 3.0Å, we can say that any $D_{Interolog}$ distribution compares proteins with the same structure, but different sequences.

Function Conservation. We tried to analyze the relationship between protein interactions and protein functions. Unfortunately, large scale function annotations as found, e.g., in Swiss-Prot, in the form of, e.g. Gene Ontology (GO; Ashburner et al. [2000]) terms or Enzyme Commission (EC) numbers, only reach the protein and not the sequence level. Therefore we limited our analysis to the functional diversity found in Level Interolog of our clustering (Section 2.3.7), where proteins are grouped by Pfam [Punta et al., 2012] families. (Note that Pfam itself aspires to only group functionally related proteins. Hence, we are to some degree comparing different function classification systems.). In this context, we first have to report a negative result: after mapping level Interolog cluster with more than 1 member to experimental GO annotations (evidence codes IDA, IMP, IPI, IGI, IEP, TAS, IC, and EXP), only 26 clusters had more than one functionally annotated protein pair. This was clearly not enough to generally link functional and interfacial diversity. Even a case-by-case reasoning failed: 15 of 26 clusters contained proteins which differed in their annotation already on the first level of the Molecular Function ontology. Curiously, the term leading to by far the most diversity was ‘binding’. This means despite clear evidence of protein binding in the PDB, the experimental evidence had not made it into Swiss-Prot, yet. Manually curating the annotations in the 26 clusters after this finding, we could not find clear evidence of functional diversity.

Consequently, we switched from GO annotations to EC numbers, in the hope of more and more complete functional annotations. Indeed, the majority of proteins were annotated with EC numbers, and we could derive a distribution of function conservation in the following way: First, we defined a pairwise function similarity measure: given two proteins, it returns the number of conserved EC number digits. For example, if protein A has EC number 3.4.11.4 and protein B number 3.4.16.1, the measure returns the number 2, because the first two digits are conserved. Calculating the distribution of pairwise functional similarities in this way for each cluster and then averaging over all clusters, we obtained a $D_{Interolog}$ distribution (Section 2.3.7). We present it in Fig. 2.8C. Here, we see that 60% of protein pairs in a cluster have exactly the same EC number. Conservation then sharply drops for 3 and 2 conserved digits, but rises again for 1. This should mainly be due to random effects, however, since complete enzymatic heterogeneity (no conserved digits) has the lowest occurrence (3%).

2.4.2 Interface Similarity Distributions

2.4.2.1 Sequentially Identical Pairs of Interactions ($D_{SameSeq}$)

Most interfaces were mostly robust for identical pairs of interactions ($D_{SameSeq}$). When two different experiments measured exactly the same external interaction (distribution $D_{SameSeq}$; Section 2.3.7), usually the interface between the two proteins was identical. Depending on the similarity measure, the number of largely conserved interfaces varied between 60 and 89% (Fig. 2.6 and Fig. 2.10). The most representative measure (Face Position Similarity) found the same interface in 75-79% of all cases (Fig. 2.9A, $D_{SameSeq}$, rightmost bar). Conversely, interfaces largely differed between two observations in about 12% (Fig. 2.9A, Face Position Similarity <0.5).

Other measures introduced in this work were, for instance, very sensitive to side-chain movements (Convex Hull Overlap), or only roughly assessed the conservation of the interface location (Sphere Radius Ratio). Taking into account two similarity measures simultaneously, small differences were observed in as many as 49% of all comparisons (Section 2.4.2.6).

In contrast to our measures, the L_rms (used by CAPRI) returned distances in Å for the entire protein rather than for the interface alone. This perspective could capture conformational changes outside the binding regions that would be missed by other measures. L_rms found 64-69% of all 'ligands' (per definition the smaller of the two proteins in the interaction) not to exhibit conformational changes and to bind to the larger proteins at the same positions (RMSD <1.0 Å). Conversely, 10-14% of the interfaces differed very substantially between alternative experiments (>9.0 Å). Considering Face Position Similarity and L_rms at the same time suggested that about 1% of all comparisons did not differ by the first but differed substantially (>9 Å) by L_rms (Section 2.4.2.6). In other words, at least 1% of all the changes between different experiments can be attributed to conformational changes outside the binding region.

Another CAPRI measure, the I_rms, compared the shapes of the interface regions common to both protein pairs. We found these common regions to be very different in about 4% (e.g. due to a rotation of one of the proteins) and largely conserved in 80% (Section 2.4.2.5).

We confirmed the surprising result of interface variability without sequential changes through a variety of additional analyses. The degree to which interfaces were mostly robust (ratio between rightmost and leftmost bars in Fig. 3) was a function of the number of copies of a particular interaction in a complex (i.e., a function of the 'interface copy number'; Section 2.3.4; e.g. Fig. 1: S1/S3 observed once in C1): the more copies, the relatively lower the bars on the right and the higher on the left (Section 2.4.3.2). But all of this also varied between families and particular complex subgroups (Section 2.4.3.3). For instance, MHC (Major Histocompatibility Complex) interactions were much less diverse than others. In fact, they contributed importantly to our overall results, although they constituted only a small fraction of all interactions. Like many before us, we also had to choose key parameters to define an interface (Section 2.3.3). As previous studies differed in these parameters, we also provided results for several alternative choices (Section 2.4.1). For instance, we included structures with a resolution >2.5 Å, used 4 Å instead of 6 Å as the minimal distance between two interacting residues or did not consider the change in solvent accessibility upon binding (ΔASA) when defining interface residues. These additional analyses demonstrated that some of our quantitative results depended crucially on the choice of critical parameters while the qualitative findings did not.

2.4.2.2 Genetically Identical Pairs of Interactions ($D_{SameProt}$)

Minor sequence variations slightly increased binding diversity ($D_{SameProt}$). Two hetero-dimers can differ by minor sequence variations and still correspond to the same external interaction. Comparing these slightly different pairs (Fig. 2.9, $D_{SameProt}$) suggested considerably lower interface conservation than for the same pairs (Fig. 2.9, $D_{SameSeq}$): the most conserved bin (0.9-1.0) was reduced by about five percentage points for Face Position Similarity (Fig. 2.9A black vs. dark gray) and by nine percentage points for the L_rms measure (Fig. 2.9B black vs. dark gray). These reductions were evenly distributed over the other similarity ranges. This result can be cast into two opposing views. On the one hand, it suggested that a PPI network accurately reflected the interactions because different protein variants only had a small effect on interfaces. On the other hand, there was a significant influence of small sequence changes. For instance, the range of very different interfaces (0.0-0.5) by the Face Position Similarity measure rose from 12% to 17%. In other words, about one interface pair in six differed substantially.

2.4.2.3 Interologous Interactions ($D_{Interolog}$)

Conservation broke down when comparing interologous interfaces ($D_{Interolog}$). When two experiments measured external interactions that did not correspond to the same protein pair, but to the same family pair ($D_{Interolog}$), interface conservation dropped significantly by both measures (Fig. 2.9, $D_{Interolog}$, rightmost bars; Face Position Similarity: 28-36%; L_rms: 7-11%). For Face Position Similarity, these differences largely originated from a shift toward intermediate levels of conservation, suggesting that most changes partially preserved the approximate interface location. The Sphere Radius Ratio supported this interpretation (Section 2.4.2.4). Nevertheless, the interfaces with clear dissimilarity also increased from 13% ($D_{SameSeq}$) to almost 30% ($D_{Interolog}$, Fig. 2.9, cumulative black to light gray bin with <0.5). This effect was stronger for L_rms: 33-40% of all comparisons were by this measure clearly dissimilar (>9 Å; Fig. 2.9B, light gray vs. black). For these strong differences, the effects from conformational changes (Fig. 2.8) and from local interfaces appeared to act synergistically.

We hypothesized that families of interologs without alternative binding might have similar functions and that the same could be true for families with extreme binding diversity. Unfortunately, only for 18 Level Interolog clusters, interfaces were always maintained (Face Position Similarity >0.9 at 100%), while only 17 others always used very different interfaces (Face Position Similarity <0.5 at 100%). These numbers were too small to permit statistically significant analyses on the functional differences between those interactions. We still provided a list of those cases in Appendix A. The two most extreme findings of this analysis were that the Gene Ontology (GO) [Ashburner et al., 2000] term 'tetrapyrrole binding' appeared over-represented in the interactions that differed, while 'cytoskeletal protein binding' appeared over-represented in the interactions that did not change.

2.4.2.4 Additional Face Similarity Measures

In the following, we present the results for additional face similarity measures (Fig. 2.10; measures introduced in Section 2.3.5). The calculation of the distributions is described in Section 2.3.7.

Overall, the curves for Convex Hull Overlap are similar to those of L_rms. Note, however, that this measure exclusively compares the interface area and not the entire proteins. Its high sensitivity comes from a comparison of interface shapes and their overlap in volume. Side-chain movement on the edge of an interface, for example, can lead to different interface shapes and thus to lower Convex Hull Overlap.

Sphere Radius Ratio, on the other hand, is the most robust of all measures. This is mainly because it is very difficult to achieve low similarities the way it is calculated (radius of one interface devided by radius of both interfaces combined) and considering the typical proportions of a protein. It is interesting to see that the rough positions of the interfaces on the proteins appear to be conserved even for $D_{Interolog}$.

Interface Composition Similarity compares the amino acid compositions. Also here, dissimilarity is present in all distributions and proofs that differences in terms of residue positions are not due to interface duplications, as expected for example for a protein with two sequentially identical domains. As the measure is also prone to return higher similarities simply by chance (any two interfaces larger than 10 amino acids will have at least one in common), we additionally show the random distribution which was derived by randomly picking interfaces and comparing their compositions. While $D_{SameSeq}$ shows typical values in the area of high interface conservation, lower ranges are not populated. This is in line with the random distributions. In $D_{SameProt}$ and $D_{Interolog}$, the divergence of sequences sets in, because we now allow different amino acids at the same position in the interfaces. The range 0.9-1.0 in $D_{Interolog}$ has the lowest absolute value. Curiously, the influence of sequence divergence and the random distribution merge seamlessly: $D_{Interolog}$ resembles an equal distribution until about a value of 0.5.

The I_rms distribution is quite complex: in the range >4.5Å-4.0Å, we see a difference between distributions similar to L_rms , but with lower overall occurrence. However, the fraction of comparisons with highest I_rms, i.e. where no common interface residues could be found, accounts for only 1-2% *in all three distributions* (not shown). Consequently, their differences must come from comparisons for which common interface residues could be found, but where these residues had very different atom coordinates. This could be due to, e.g., interface rotations, different overlapping binding modes, but also to overall conformational changes (for the I_rms, we included residues in the interface which were as far as 10Å apart). We can hypothesize from this result that 'binding clouds', as shown in the sample structures, should be more frequent in D-Interolog than the other two distributions. Moving on to intermediately low similarities, they seem to be surprisingly rare. One issue is that common interface residues can often be found for only one side of the interaction, i.e. for only one face. In this case, I_rms turns into a comparison of identical fragments of the same protein (family), instead of a comparison of interfaces. Such comparisons usually result in low similarity/high distance ranges (1.0Å- 0.0Å). Similarly, even if all chains have common residues, the exclusion of non-common interfaces residue can lead to low I_rms despite actually different interfaces. Compared to other measures, I_rms should therefore be considered as rather insensitive. Nevertheless, in absolute terms, occurrences of identical interfaces fell well in between the ranges of the other measures.

2.4.2.5 Additional Interface and Domain Similarity Measures

Now, we present the curves for interface and domain similarity measures (Sections 2.3.5.2, 2.3.5.6 and 2.3.5.7). As results are overall very similar to those already shown in the context of other measures, we focus on simplicity and only present $D_{SameSeq}$ distributions.

In general, Interface Position Similarity (Fig. 2.11) draws a similar picture as the face counterpart. As could be expected, though, the chance that a pair of interacting residues is conserved becomes very small when one or both of the interacting faces change according to the Face Position Similarity distributions, thus leading to an increase of high complementarity for Interface Position Similarity compared to Face Position Similarity. Only around 10% of all similarities now lie in the intermediate range between 0.1 and 0.9. Changes in domain numbers or families as measured by Domain Number Ratio and Family Interaction Similarity occur in about 5-15% of all $D_{SameSeq}$ comparisons. We can interpret this observation to imply that interfaces are slightly more similar in terms of this measure than in terms of the others. However, we could also argue that this low difference is partially explained by ambiguities in terms of the domain definitions (and directly connected to this: in terms of the number of domain families), as for instance SCOP and CATH domain assignments differ by about this magnitude [Liu and Rost, 2003, 2004]. In any case, it indicates a steady rate of multi-domain proteins which have learned to interact with their domains in more than one way. Only considering domain-domain interactions, there is a chance we miss parts of the interface between two entire proteins and do not see, for example, that not all domains are always needed for the interaction.

2.4.2.6 Cross-correlations

The full extent to which interfaces vary is best appreciated when cross-correlating multiple measures. The same interface pair might appear identical by one measure and different by another. To identify such cases, we selected a few pairs of measures and derived the correlation between their distributions. To this end, we no longer calculated the occurrence of a single similarity range (e.g. 0.9 to 1.0) for one particular measure. Instead, we looked at two similarity ranges simultaneously, each corresponding to a different measure. For example, we measured how often the similarity of a pair of interfaces lies in the range between 0.9 and 1.0 according measure 1 while being between 0.1 and 0.2 for measure 2 (Section 2.3.7.2 for details). The visualization of the entire distribution, i.e. all possible range combinations, then obviously required three axes: one determining the value of the first measure; one the value of the second measure; and one the probability of observing both similarity values at the same time. Simple matrix plots displayed this: the first measure was assigned to the x-axis and the second measure to the y-axis. The square at a particular x-y coordinate then corresponded to the occurrence of this particular combination of interface similarity: The darker the square, the more often the combination was observed. Hypothetically, these distributions could again be derived for distributions $D_{SameSeq}$ to $D_{Interolog}$. The relations between the measures is already evident for $D_{SameSeq}$, however, so that we limited the analysis to comparisons between interfaces from the same sequence pair ($D_{SameSeq}$). Results are presented in Fig. 2.12.

If one measure suggests that an interface is identical between repeated observations and another that it differs, which one is right? Clearly, the cross-correlation of measures adds to the view that interfaces have some flexibility to cope with change. In 62% of all comparisons, both Face Position Similarity and L_rms agree that the interfaces between two measurements were largely identical (Fig. 2.12A). The remaining 38% mainly appeared in ranges above 0.5 in both measure, with a slight tendency of L_rms to be more sensitive than Face Position Similarity. This trend strikingly manifests itself for L_rms values above 9 (7% of all comparisons): here, we see at least some population in every range of Face Position Similarity. Apparently, conformational changes of the entire protein can lead to high RMSDs, but preserve the interfaces to a point where no interface residue changes are detectable. The matrix comparison between Convex Hull Overlap and L_rms shows a similar trend, but is more pronounced (Fig. 2.12C). Now, we observe almost any pair of similarities. Again, highest L_rms is often accompagnied by intermediate Convex Hull Overlap. Note that both measures agree on conservation of an interface pair in only 51% of all cases. This means that there is a 49% chance that the same interfaces from two different PDBs will differ from each other by at least one measure. Comparing Face Position Similarity and Convex Hull Overlap, finally, we see that Convex Hull Overlap is significantly more sensitive (Fig. 2.12B). It assigns lower values to the same interface pair in about 27% of all cases, while Face Position Similarity does the same in only 2%. In case the two measures differ, the difference is usually not large: most values are close to the diagonal and there are no cases where one measure assigns highest difference and the other highest similarity.

2.4.3 Additional Analyses

2.4.3.1 Effect of Alternative Binding to the Same Homomer

Trivially, removing alternative binding to the same homomer reduces diversity. With a special filter, we might remove all alternative binding of a protein to the same homomer from $D_{SameSeq}$ to $D_{SameFam}$ Section 2.3.8. Obviously, filtering out diversity will reduce the signal of diversity observed. Nevertheless, we performed this analysis.

As expected, the observed effects dropped significantly (Table 2.2), most extremely for $D_{SameSeq}$, i.e. for the same pairs. The differential behavior between $D_{SameSeq}$ and $D_{Interolog}$ might be explained by sequence divergence increasingly leading to very different interfaces for the same protein pair and ultimately to different quaternary states. Despite all the filtering for homomers, varying interfaces remain frequent and still almost one third of the change seen in interologous pairs ($D_{Interolog}$) is also observed between the same pairs ($D_{SameSeq}$).

Distribution	Original	Homomer filtered (sequence)	Homomer filtered (structure)
D-SameSeq: 0.0-0.5	11-16%	3-4%	-
D-SameSeq: 0.9-1.0	75-79%	84-88%	-
D-SameProt: 0.0-0.5	14-19%	4-7%	-
D-SameProt: 0.9-1.0	69-75%	80-84%	-
D-Interolog: 0.0-0.5	26-32%	11-16%	8-12%
D-Interolog: 0.9-1.0	29-35%	34-41%	38-47%

Table 2.2: Influence of homo-oligomerization. This table shows a summary of the Face Position Similarity distributions of Fig. 2.9 after excluding diversity introduced by a protein chain binding to the same homo-oligomer at different positions. We used two different definitions for homomers: at the sequence level, all chains in the assembly come from the same protein. In a 'structural homomer', they only come from the same family.

2.4.3.2 Effect of Interface Copy Number

By restricting measurements to interfaces with the same associated copy number (Section 2.3.4), we could derive one overall similarity distribution $D_{SameSeq}$ for each observed copy number. These distributions were calculated with the procedure described in Section 2.3.7. In case of copy number 3, e.g., this meant that we calculated the distribution of a single Level SameSeq cluster only with interfaces which come from a complex in which this interaction occurred 3 times. Comparisons between interfaces with different copy numbers were discarded. Due to lack of samples for specific copy numbers above 4, we had to limit this analysis to copy numbers 1 to 4 plus an average over interfaces with a copy number higher than 4. Results are given in Fig. 2.13.

As can immediately be seen from the high similarity range (0.9 to 1.0), interface similarity is clearly linked to the copy number of the interaction in the respective complex. Interfaces in dimers are highly conserved, with differences almost exclusively staying in the range from 0.8 to 1.0. High similarity then gradually declines so that the curve for complexes with more than 4 copies of the same interaction shows similarity to an equal distribution.

2.4.3.3 Effect of Significant Complex Subgroups

There is a possibility that the observed interface variability is confined to particular complex families or subgroups which are known to be overrepresented in the PDB. To address this, we first looked at the occurrence of the 0.9 to 1.0 similarity range in the distribution of each Level Interolog cluster (no sequence divergence [$D_{SameSeq}$]; data not shown). We found a continuum of values, ranging from 0% to 100%. This means there is no particular group of families responsible for the overall observed interface variability. Instead, there are examples for each degree of variability. Furthermore, we found that the size of a family (number of Level SameSeq and Level SameProt clusters in a Level Interolog cluster) does not correlate with its 0.9 to 1.0 bin. Secondly, we queried the PDB for known Virus, Antigen/Antibody and Major Histocompatibility (MHC) complexes and determined the intersection of these subgroups with our data set. This revealed 39 viral, 235 Antigen/Antibody and 198 MHC structures in our data, accounting for 3.0%, 18.2% and 15.4% of all complexes, respectively, and, due to overlap between the sets, together to 21.8%. Distributions for the Face Position Similarity measure ($D_{SameSeq}$; Section 2.3.7) and each subgroup are given in Fig. 2.14.

Even though the distributions of the three subgroups substantially differ from the overall distributions, it becomes clear that they are not exclusively responsible for interface variabilities observed in the entire data set. MHC complexes actually appear to be the cause for quite a high fraction of conserved interfaces. In contrast, the Antibody/Antigen subgroup populates the area from 0.0 to 0.9 with 8% of all similarities. In the same range, viral interface similarities are more frequent at a cumulative rate of 21%. Due to small sample sizes, error estimates in all distributions are generally large.

2.4.4 Sample Structures

Examples illustrate that interfaces can really differ substantially. Our finding that most interactions form identically when repeating the experiment might not be surprising. The observation that many interactions differed substantially, in contrast, appears much more counter-intuitive. Readers might attribute the difference to some mistake in the way we measure similarity or build our data sets. We addressed these concerns by expanding our analysis in many directions. On top, we analyzed ten case studies in more detail. Three are discussed in detail in the following, the other seven shortly in Sections 2.4.4.4. Section 2.4.4.5 lists even more samples that we did not investigate in detail.

2.4.4.1 Ras - Son Of Sevenless

The same proteins binding at entirely different interfaces. Our first example is the enzyme ras in complex with the nucleotide exchange factor SOS (Son Of Sevenless; Fig. 2.15A). Ras catalyzes the conversion of guanosin tri- to diphosphate (GTP $\rightarrow$ GDP). It needs the interaction with SOS in order to release GDP again after conversion. To this end, SOS provides a binding site that is highly specific for the ras-GDP complex. However, despite this specificity to ras-GDP, also SOS and ras-GTP can form a complex [Margarit et al., 2003]. SOS has a second interface far away from the first that is specific for ras-GTP. It accelerates the reaction 'ras-GDP $\rightarrow$ ras + GDP'. In other words, the ras-GTP-SOS complex separates GDP from ras faster than the uncomplexed SOS. Consequently, we have a positive allosteric modulation, in which both the active and the allosteric site are specific for exactly the same protein. This represents the rare case of sequence-identical protein pairs ($D_{SameSeq}$) binding very differently without prior homomerization of subunits (Table 1) and with a low interface copy number (2; Section 2.4.3.2). In Section 2.4.4.4, we discuss another similar case (Yersinia Pestis Antigen).

2.4.4.2 Cyclin - Protein Kinase

Related work mostly differentiates alternative binding modes by clustering approaches. This implicitly suggests the assumption that the system could fall into alternative minima. Our results seem to support this assumption: interface similarities follow an extreme distribution, i.e. either they are very similar or very dissimilar (Fig. 2.9). When digging deeper, however, we easily found exceptions. Binding modes can form a contiuum. We discuss one case in the following and two others in Section 2.4.4.4.

The dimeric interactions between cyclins and protein kinases (interface copy number 1) reveal a largely conserved binding cloud (Fig. 2.15B, green and cyan). Pairwise Face Position Similarities span a range from 0.6 to 1.0. An automated clustering of the structures would not accurately reflect the reality of this case because it would discriminate between the interfaces. Such a clustering might be improved by using the functions of the respective proteins in order to find correlations between interface and functional similarities. However, we found our current functional classification (GO) not to be comprehensive enough for this, yet (Section 2.4.1.2).

Another complicating factor is that continua can not only be observed on the $D_{Interolog}$ Level (as is the case here), but also on the Levels of $D_{SameProt}$ and even of $D_{SameSeq}$ (Section 2.4.4.4; namely hemoglobin and choleraholotoxin). For those two examples, GO-like functional annotation systems are trivially insufficient as they operate on the level of proteins, not protein variants or even interfaces.

Further extending the interface variability of cyclins and protein kinases is the recently discovered interface between cyclin D1 and protein kinase 4 (Fig. 2.15B, blue). Its discovery seems mainly due to improvements in experimental methods [Day et al., 2009]. Hence, this might only be the beginning of an entirely new class of binding modes.

2.4.4.3 F1-ATPase

So-called ‘structural homomers’ often act as a gate-way to higher-order complexation. Homomers can bind to another protein through alternative interfaces (Fig. 2.16D,F,E). This concept can be generalized by introducing structural homomers, i.e. assemblies between proteins from the same family (Section 2.3.8). The F1-ATPase structure (Fig. 2.15C) is such a case. Here, the alpha and beta subunit (Fig. 2.15C: orange and cyan) bind alternatingly to each other, thereby forming a hexameric ring with a central pore. The gamma subunit (Fig. 2.15C: green) winds through this pore with two long helices. This hexameric ring alone already represents two entirely different binding modes between the same protein pairs ($D_{SameSeq}$) and without homomerization of subunits. Two alpha subunits are always separated by a beta subunit and vice versa. Furthermore, all of these six chains bind to different positions on the gamma subunit, leading to two other interactions with great variability. Interface clouds of different structures, e.g. under varying conditions, reveal the dynamic nature of the interaction. Especially a rotation of the hexameric ring around the central helices is frequent.

The missing homo-oligomeric context is hidden: the hexameric ring appears to be a homomer. Subunits alpha and beta have a RMSD of 1.3 Å. Only the sequences and eventually their original proteins, reveal that we are actually dealing with heteromers. However, both of the proteins come from the same family, so that we have a case of a ‘structural homomer’, i.e. an assembly that appears to be homomeric, but has actually undergone significant sequence divergence. Similar to traditional homomers, the alternative binding to another protein (subunit gamma) is evident from the structure within seconds and homomerization might be essential (i.e. interactions between alpha and gamma subunits might not be stable without the hexameric ring). Filtering alternative binding to the same structural homomer (Section 2.3.8), the only remaining variability is the rotation around the two central helices of subunit gamma. This is a good example why homomeric filtering might come short from a biological point of view: What if subunits alpha and beta would not have the same family?

Would it make the variability between alpha and gamma subunits any more or less biologically relevant? Unfortunately, current data does not lead to specific answers, yet. Similar examples for structural homomers are some hemoglobin structures (Section 2.4.4.4) and the exosome complex by Lorentzen et al. [2007] (2JE6).

2.4.4.4 Additional Sample Structures

In the following, we discuss other cases of differing interfaces for the same pair of sequences. They add to the picture of binding diversity.

Yersinia pestis (Fig. 2.16A) represents one of the very few cases of clear alternative binding between two interfaces with copy number 1 (Section 2.3.4). All measures agree in this finding. The antigen usually forms fibers by repeatedly binding to itself. In the two structures displayed here, this has been disabled by mutating the N terminus in various ways. What remains are two original interactions with a chaperone protein, representing snapshots of different stages in the fiber assembly process. Note that 1P5U has actually three chains with two corresponding to the antigen, but only one of them also corresponds to the N-modified variant present in the binary structure 1P5V.

Choleraholotoxin is an extreme case of an interaction with Face Position Similarities almost exclusively lying in the intermediate range 0.1-0.9 (Fig. 2.16E, Fig. 2.17C,D). The shorter B chain forms a cyclic homo-pentamer with a pore in the middle. The A subunit occupies this space with a terminal loop. This results in several different binding positions of a single B chain on the A subunit in one PDB structure. Furthermore, another structure of the same complex reveals that the pentamer exhibits some translational freedom with respect to the A subunit, leading to even more interface diversity. These types of rotational interfaces can generally be found by cross-correlating the measures Sphere Radius Ratio and Face Position Similarity. A high Sphere Radius Ratio suggests that interface sizes and locations are conserved. If the Face Position Similarity is low, change must therefore come from a rotation around a central axis. Only in this way, we preserve the radius and position and yet change the residues of the face.

For hemoglobin, (Fig. 2.16C), a clustering should easily interpret the two big interface 'cloud' as such and identify two distinct binding modes. This is further supported by the distribution of pairwise interface similarities (Fig. 2.17C) that is mainly populated either in regions of low (0.0-0.2) or high similarity (0.9-1.0). Closer inspection, however, reveals that the clouds are by no means biologically irrelevant: A main contributor to their variety for example is the change of hemoglobin from the T to the R conformation when releasing or binding oxygen. A typical clustering would not only miss this conformational change, but also its seemingly continuous nature with many intermediate states. Note, however, that a detailed functional annotation of each interface according to the conformational state could improve that.

We discuss Cytochrome BC1 and Type IV collagen in the caption of Fig. 2.16 and RuBisCO in the next Section.

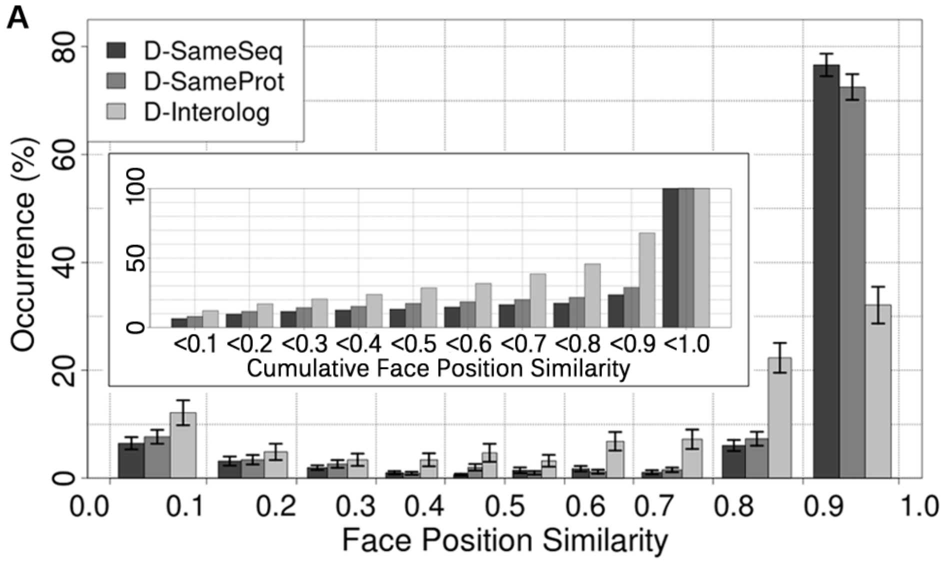
Difference in Number of Interacting Families. Over 7% of all interface comparisons suggest a difference in the domain families that interact (Fig. 2.11). Two examples illustrate the above conclusion that many of those are limits in the reliability of our domain/family definition. The first is RuBisCO (Fig. 2.17A) that has a short chain (single domain, single domain family). It has evolved three clearly distinct binding positions on the large chain (2 domains, 2 domain families). Two of these faces fall on the same domain; the third falls on a different domain. Consequently, we see two different family pairings in the same pair of proteins. The second example is the complex of aldolase-dehydrogenase (Fig. 2.17B). The two aldolase domains both contribute almost equally to one interface while on the other interface, only the catalytic domain interacts with the dehydrogenase. The same pattern is true for the comparisons between the faces on the two-domain dehydrogenase (not shown).

2.4.4.5 Other Interesting Cases

We had to exclude other interesting examples to limit the length of this work. These included the $D_{SameProt}$ comparisons of two nitrogenase complexes (2AFH, 1QH1) exhibiting alternating quaternary states [Tezcan et al., 2005]; two amine dehydrogenase complexes (2J57, 2IUP) with different enzymatic activities (Section 2.4.1.2); dual binding modes of cohesion and dockerin (2CCL, 1OHZ); and two interologous interactions for which the according Swiss-Prot sequences are identical and only differ in their organisms (2PE6, 2VRR; dimeric). Notable previous publications reporting different binding modes include, e.g., [Park et al., 2004] (histidine kinases; 1U0S) and [Kang and Crane, 2005] (cytochrome C; 2B11).

2.4.4.6 Summary

While our examples confirmed the overall trends, they also suggested that the averages above reveal only the tip of the iceberg: if there is one reasonable measure for interface similarity by which two experimental solutions differ, then this observation suggests variability. To complicate matters further, we observe ‘rotational interfaces’ (F1-ATPase) and see that ligand binding can have great impact on interface specificity (ras-SOS). On another note, our tests with alternative data set parameters, e.g. changing structural resolution, revealed that the variability that we see is not explained by experimental or procedural inaccuracies. Thus, the PDB structures clearly tell a tale of unexpected variability and dynamics of biological interactions.

A


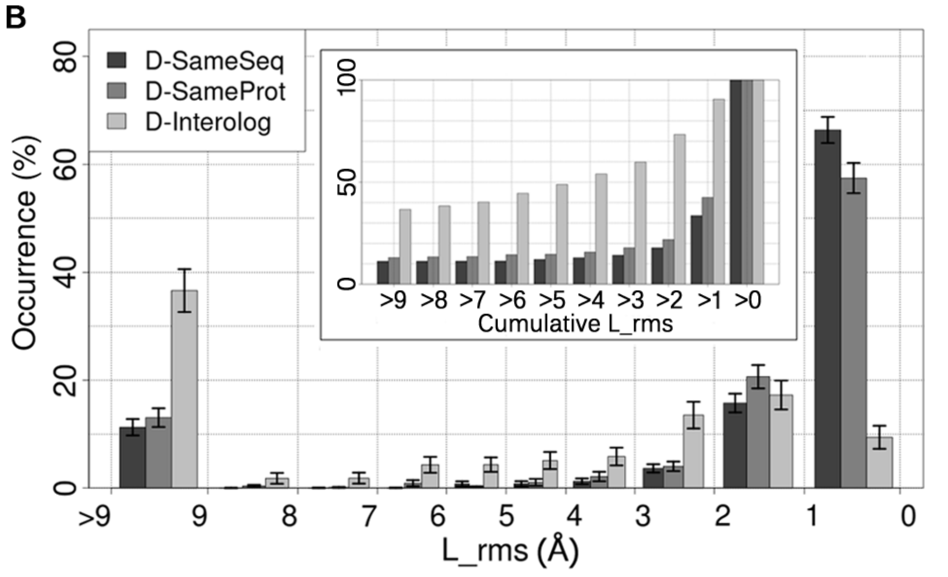
B


Figure 2.9: Faces are similar yet different. For two different ways to measure interface similarity - Face Position Similarity **[A]** and L_rms **[B]** - we present the similarity distribution for all interfaces. The rightmost interval shows largely identical faces, the leftmost completely different faces. For each similarity range and measure, there are three bars: one for each type of sequence divergence ($D_{SameSeq}$ to $D_{Interolog}$). For example, Face Position Similarity finds about 7% of all the interface similarities at $D_{SameProt}$ to fall in the range 0.0-0.1, i.e. suggests in 7% of the cases completely different outcomes when experimentally measuring the same interaction again. The error bars show standard errors and are explained in Section 2.3.7. The inlet displays the cumulative distribution giving the fraction of all similarities that differ by a certain value. For instance, 21% of all interface comparisons result in a value above 2 Å according to the L_rms in $D_{SameProt}$. In these cases, the two smaller proteins are clearly not in the same position after superimposing the two larger proteins.

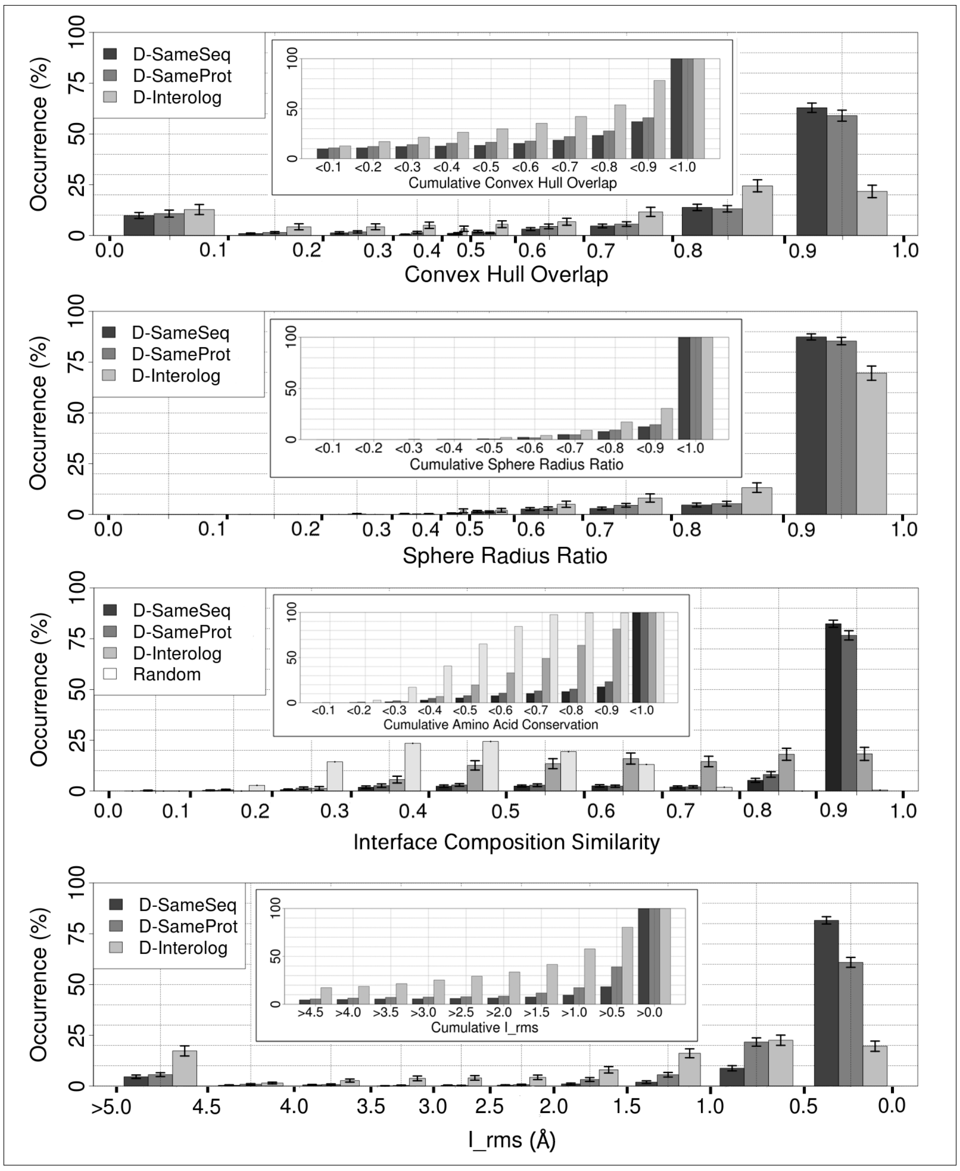


Figure 2.10: The face similarity distributions for Convex Hull Overlap, Sphere Radius Ratio and Interface Composition Similarity. The random variant of Interface Composition Similarity was derived by repeatedly randomly picking two interfaces from two different clusters in the respective level and applying the measure. For each similarity range and measure, there are three bars. Their order corresponds to the three Distributions $D_{SameSeq}$ to $D_{Interolog}$. The inlet in each plot displays the corresponding cumulative distributions.

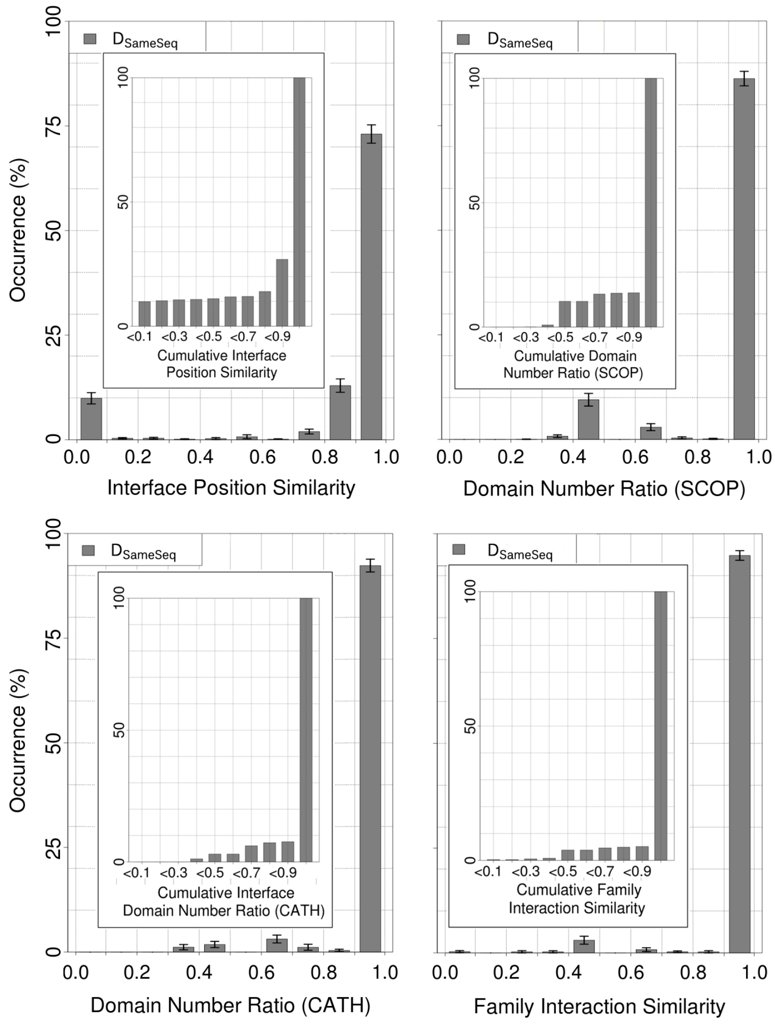


Figure 2.11: The interface similarity distributions for Interface Position Similarity, Domain Number Ratio (SCOP and CATH) and Family Interaction Similarity. We limited the analysis to distribution $D_{SameSeq}$ for reasons of simplicity. In the context of the Domain Number Ratio, we present the results for both SCOP and CATH. The Family Interaction Similarity was only derived for SCOP. In each plot, the inlet shows the corresponding cumulative distribution.

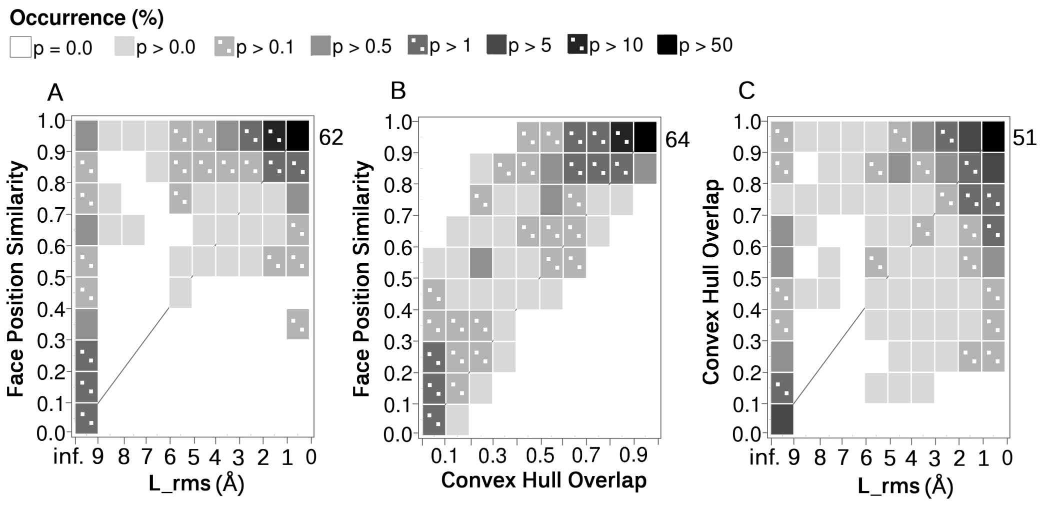


Figure 2.12: Cross-comparing measures reveals more diversity. Overall, the different measures agree that most repeated measurements yield similar results. However, they often disagree in that one considers a pair of two measurements to be similar while the other measure considers the same to differ. Here, we show three cross-correlations of measures (A-C). Simply put, if the measures agreed, all diagonals would be black, and everything else would be white (blank cells mean that the particular pair of interface similarities has not been observed). The uppermost rightmost cells correspond to very high interface similarity according to both measures and also show the exact percentage of the respective combination. All plots show $D_{SameSeq}$ distributions.

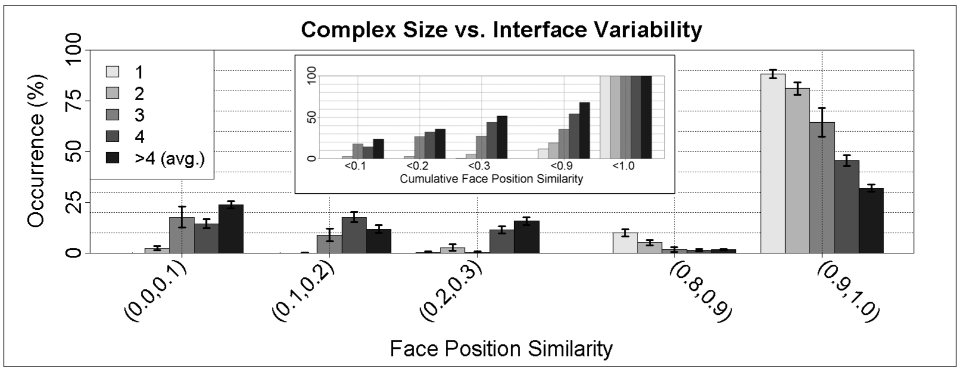


Figure 2.13: The change of the similarity distribution when restricting comparisons to interfaces with a particular copy number. A given interface corresponds to a certain interaction of two proteins and originates from a particular complex (Fig. 2.1). We counted how often the same interaction occurred in a complex and assigned this number, the interface copy number, to the interface. We repeated this for all interfaces. In the Figure above, we show $D_{SameSeq}$ distributions derived exclusively with interfaces sharing a particular copy number. This number is depicted by shades of gray: the darker, the higher.

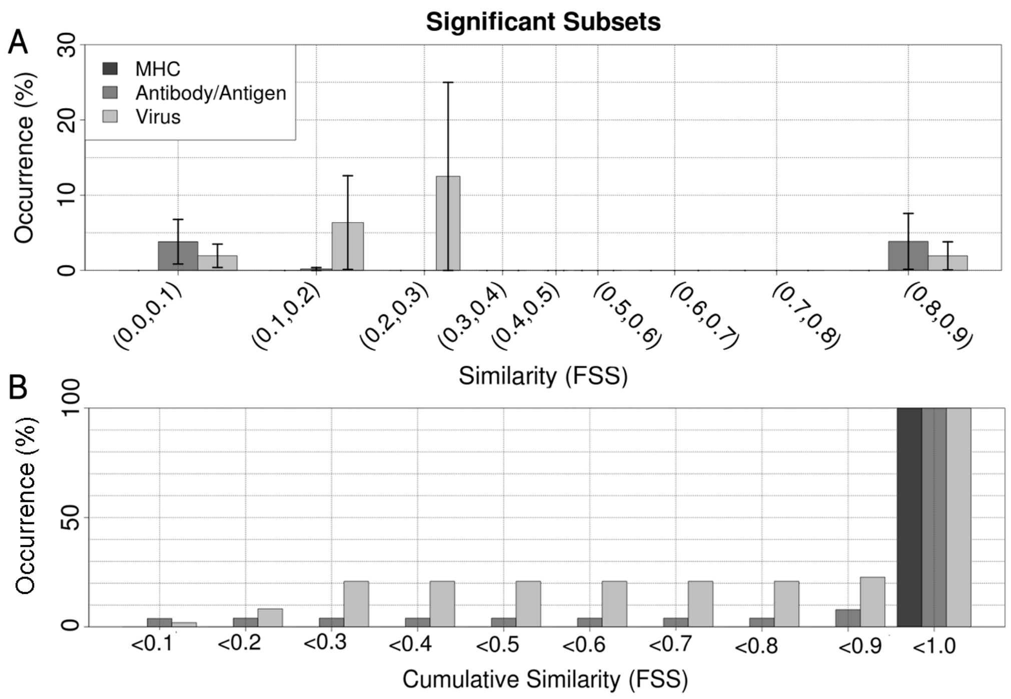


Figure 2.14: Distributions of interface similarity in particular complex subgroups. In **(A)**, we show the $D_{SameSeq}$ distribution without the range 0.9-1.0. It was omitted due to its overall dominance. Instead, we present it in the corresponding cumulative distribution **(B)**.

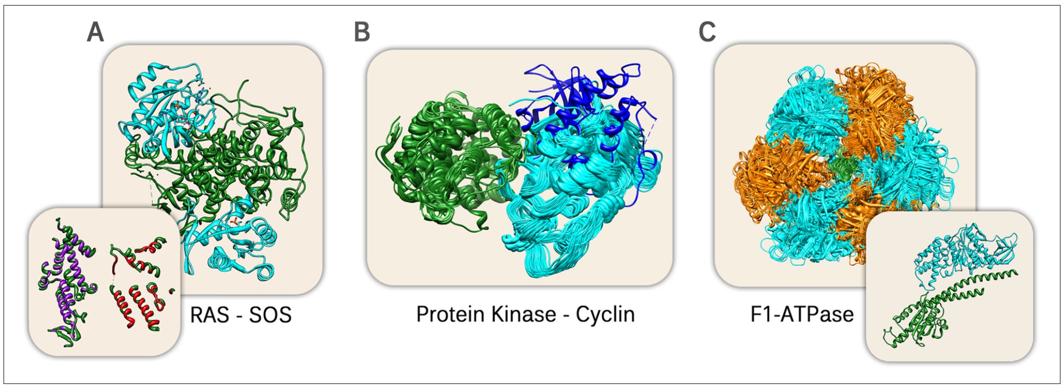


Figure 2.15: Three typical interactions exhibiting surprising variety. (A) Protein 'ras' binds to 'son of sevenless' (PDB ID 1NVV): alternative binding for sequence-identical pairs of proteins and without a multimeric context; the lower left panel shows the residues of the two interfaces in purple and red. **(B)** Natural dimeric interactions between proteins from the protein kinase and cyclin families (interface copy number 1; e.g. 1OI9). Cyclin chains (green) have been structurally aligned and superimposed. Protein kinases (cyan and blue) were subject to the same geometric translations. The blue chain has a recently discovered outlier interface (see text). **(C)** Superimposition of entire sequence-identical F1-ATPase complexes. Complexes were aligned and superimposed with the gamma chains (green). Alpha (orange) and beta (cyan) subunits were subject to the same geometric translations. In the main panel, we look at the complexes from the top. The inlet displays an interaction between a beta and a gamma subunit from the side.

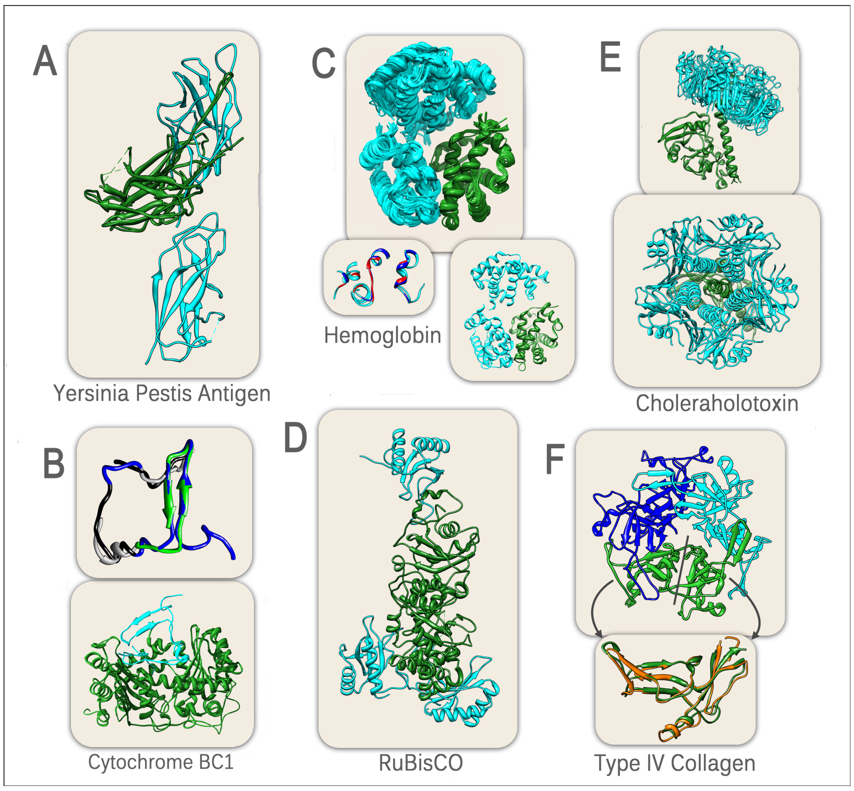


Figure 2.16: Six additional interactions with surprising interface variety. Each panel (A-F) shows superimpositions of multiple heterodimers with sequences X (dark green, note: all green have RMSD<1Å to each other) and Y. The interacting chains Y were subject to the same geometric translations as their X counterpart and are displayed in cyan. **(A)** The rare case (Section 2.3.4) of large interface differences when looking at two structures with interface copy number 1 (1P5U, 1P5V). **(B)** Example for 'low Interface Position Similarity with high Face Position Similarity' (1PP9, 2FYU): lower frame: comparison of two structures of Iron-sulfur subunit precursor (cyan) interacting with core protein 1 (green); upper frame: iron-sulfur subunit precursor superimposition in detail; black and gray: identical subsequences at different spatial locations; green and blue: different subsequences at same spatial location. **(C)** Superimposition of 251 Hemoglobin complexes (e.g. 1A3N); lower left side frame: superposition of two sample faces of upper interface cloud; face residues (blue and red) colored by chain; lower right side frame: one sample complex of each interface cloud; also see Fig. 2.17C **(D)** Ribulose-1,5-bisphosphate carboxylase oxygenase (1AA1): three highly distinct binding positions; also see Fig. 2.17A **(E)** Even distribution of Face Position Similarity vs. constantly high Sphere Radius Ratio (1S5C, 1S5D); upper frame: side view; lower frame: view from top; also see Fig. 2.17 **(F)** High Amino Acid Coupling vs. low Interface Position Similarity (1M3D, 1T61); upper frame: X consists of two homologous domains (domain boundary indicated by separating line) and interacts with Y at two different positions; dark blue used instead of cyan to show chain boundaries; lower frame: domains of X superimposed (domains highly homologous; sequences not shown).

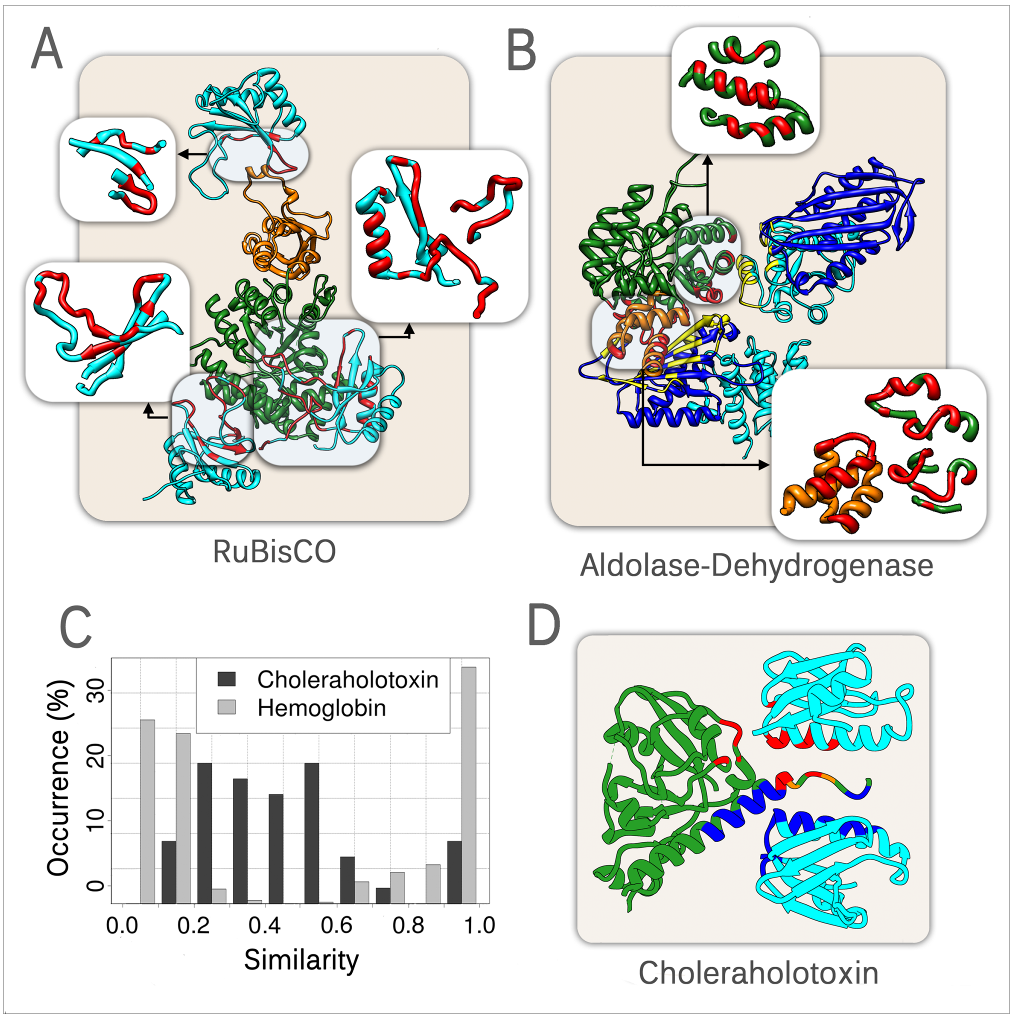


Figure 2.17: Multi-domain proteins interacting through different SCOP families and difficult cases of interface clustering. Colors as in Fig. 2.16; exceptions: second domain of X is orange, second domain of Y is blue; interface residues in red. **(A)** RuBisCO from Fig. 2.16B; interfaces magnified; the short chain (cyan; one domain) has two binding positions on the green domain of the large chain and one on the orange domain. **(B)** Aldolase-Dehydrogenase complex (1NVM); Aldolase domains: green and orange; Dehydrogenase domains: cyan and blue; Aldolase faces magnified; in the first interaction, only the green domain contributes to the face (upper frame); in the second, both the green and the orange domain are involved (lower frame). **(C)** Comparison of the Face Position Similarity distribution of two Level SameSeq groups corresponding to Hemoglobin (Fig. 2.16C) and Choleraholotoxin (Fig. 2.16E, 2.17D). **(D)** Superimposition of two diverse Choleraholotoxin interactions; created from Fig. 2.16E by removing all but two Y chains.

2.5 Discussion

How can proteins interact differently? Empirically, we found several reasons for the same two proteins to have different interfaces ($D_{SameSeq}$). The simplest was merely technical: some experimental findings may not have been completely correct. We reduced this effect by excluding complexes with resolutions >2.5 Å, but even structures at 1.2 Å can contain errors [Ginzinger et al., 2010, 2011]. Another reason was local flexibility or disorder: many proteins have local regions that are natively unstructured and these often form protein-protein interfaces [Dunker and Uversky, 2008]; such regions are difficult to track experimentally. Often, the N- and C-termini contributed to the observed interface variability. Another reason was environmental differences: despite all efforts, we could not entirely exclude artificial interfaces due to crystal packing. Different pH values could trigger conformational changes, as was the case for small ligands or other interaction partners. The presence of another protein changing the overall structure of a complex played a similar role. In all that, however, we still miss one important aspect: proteins often have evolved to interact in different ways. For such cases of biologically important alternatives, we might interpret the variety observed in a single PDB structure as an example of one protein binding to multiple copies of the same interaction partner.

There were various reasons why variability in binding was higher between sequentially modified proteins than for identical proteins. The modifications that preserved the original protein ($D_{SameProt}$) were usually point mutations (i.e. changes of single amino acids, e.g. by site directed mutagenesis or in the form of Nucleotide Polymorphisms [SNPs]). Others included protein tags at the N- or C-terminus (e.g. to facilitate protein purification), post-translational modifications (protein cleavage) and alternative splicing. For interologs ($D_{Interolog}$), finally, there was also evolutionary driven sequence divergence. As described before, however, the mere presence of insertions or deletions was not enough for low interface similarity: we reduced structures to common residues before comparing them. Thus, the increase in variability was actually the result of changes in the common parts of two structures.

Continuum of binding modes rather than major clusters? Using similar measures as we did, other groups [Kim et al., 2006, Shoemaker et al., 2006] have found that many families interact in more than one way. Our analyses support this result. However, they also reveal that the differences in interfaces span the entire spectrum of the distribution, especially for $D_{Interolog}$. Only 18 of the 151 pairs of families completely conserved the binding modes. This finding suggests the model of a continuum of binding modes rather than clearly defined groupings, e.g. as obtained by clustering at predefined thresholds. Furthermore, in our results, about one third of the variability observed in a family-family interaction appeared to be protein-intrinsic in the sense that it was also observed between sequence-identical pairs ($D_{SameSeq}$), i.e. did not originate from sequence variations (as, e.g. for $D_{Interolog}$).

Is variability caused by homomers a special case? As mentioned before, alternative interfaces might be due to the intrinsic capability of proteins to bind at different positions. This is often encountered among homo-oligomers [Levy et al., 2008]. In our case, however, it leads to a debatable scenario: a protein A can bind to multiple copies of protein B, all of which alone form a homo-oligomeric complex (Fig. 2.5). Do we then have to treat the various external interfaces between the same proteins as one interface, or are they indeed individual interfaces that ought to be differentiated? We argue for the second case: first, considering the homomer a requirement for the hetero-interactions implies that by disabling the homomerization (e.g. through site-directed mutagenesis), we also loose the interaction. This is not always the case [Zavialov et al., 2003]. Secondly, it is unclear why such a filtering should be limited to homo-complexes. Also the formation of a hetero-multimer could be a requirement for the interaction with another protein. Studying which interactions remain after disabling the potentially highly complex hetero-multimer is much beyond the currently available data. Finally, also the original publication of a complex usually describes different interfaces to the same homomer as separate interfaces.

2.6 Conclusion

Our results raise the question whether the molecular details of protein-protein interactions (PPIs) are crucial for function. Protein crystallography captures static views on those molecular details along with some information about the dynamic nature of PPIs. If the details always had to be the same to guarantee function, different experiments would identify the same interfaces. We applied many reasonable ways of measuring interface similarity in order to analyze the consistency of the molecular details of protein-protein interactions between different experiments. For sequence-identical pairs of proteins, i.e. the same biological interaction, most interfaces were almost completely conserved by all measures. However, all measures also revealed an unexpected variety. Depending on how much detail we required to be similar in order to consider two experiments to yield the same results, we found 11-37% of all observations to have significant differences, and up to 10% to be completely different. One important result was that this was a significant fraction of the difference observed between homologous PPIs. Put differently, over a third of the differences in the interactions between pairs of homologous proteins are also observed between identical proteins. These numbers may challenge the notion that the maintenance of the molecular details is crucial for function. At least, our results suggest that there appear to be many alternative solutions to maintain or actually enable the intricate molecular details: change seems an extremely frequent exception for protein-protein interfaces.

Chapter 3

Predicting Residue-residue Contacts in PPIs

3.1 Introduction

PPIs are crucial for almost all facets of cellular processes. However, the exact mechanisms of binding still escape our knowledge [D'Alessandro et al., 2010]. Even worse, there is still not a single organism for which we know all the PPIs i.e. the full interactome. Experiments to determine the structure of a protein complex could possibly shed light on the molecular details of all interactions, but are too slow to be applied on an interactomic scale. The question whether two proteins interact or not, on the other hand, can be answered much more rapidly, e.g. by high-throughput yeast-two-hybrid [Shoemaker and Panchenko, 2007], but give no molecular details. Such details, however, are usually a requirement for therapeutic progress. Unfortunately, it is likely that this imbalance in available data will persist for the next years or even decades.

From another perspective, the monomeric structure of many proteins is available either in the PDB or can be predicted to a satisfying degree. Together with the information which proteins interact, it might seem as an easy task to also model the molecular details of the two (or more) proteins bound to each other. This problem, however, can still only be solved for cases in which the structures of the proteins do not change upon binding [Lensink et al., 2007]. If this is not the case, this so-called 'docking' largely fails. In this respect, it would already be of great help to know which pairs of residues are interacting and leave other tasks, such as finding suitable conformations, to other specialized algorithms. Surprisingly, sequences might provide valuable information in this context. For example, it has been shown that it is possible to predict, from sequence alone, which residues can bind to other proteins [Ofran and Rost, 2003a].

In this chapter, we will go yet a step further and try to predict, again from sequence alone, the exact pairwise residue-residue interactions that happen upon formation of a PPI complex. We extract all residue-residue interactions from the data set introduced in the previous chapter and encode each residue and residue pair with various sequence and evolutionary features and incorporate annotations from other tools predicting other aspects of protein structure or function from sequence. This data set is then used to train a neural networks that predict for a new pair of protein sequences which residues interact and which do not.

3.2 Methods

3.2.1 Data Sets

The basis for our data set of residue-residue interactions (each of two residues from a different protein) is the SameSeq level clustering created in the previous chapter (Chapter 2). It contains 2870 clusters and each cluster corresponded to a pair of interacting sequences. Each sequence pair was observed to be interacting in at least one high-resolution complex (<2.5Å) in the PDB (Aug 2011). Consequently, each SameSeq cluster maps to a number of binary complexes and the next task is to define a fixed set of residue-residue interactions for each such cluster. Embracing our findings on alternative interfaces, we use the union of all contacts in a cluster. The only requirement is that each single interface is large enough, i.e. has at least 5 residues with ΔASA>0 (accessible surface area) for both of the interacting proteins.

In order not to skew our predictor towards certain over-represented protein families and construct difficult test cases, we redundancy the data set on the level of protein sequences. To this end, we randomly picked a cluster (pair of sequences) from the full set and added it to the new redundancy reduced set if none of its two sequences had a HVAL$>$ 20 to any other sequence that was already in the new set (HVAL 20 corresponds roughly to 40% sequence identity for 250 aligned residues [Rost, 1999]). Otherwise, the pair was discarded. This left us with 451 non-redundant interacting sequence pairs.

We split this set into four parts, three with 100 and one with 151 interactions. We used the first three for training, while the fourth was treated as a hold-out set and untouched until the predictor was fully trained. For training, we alternately merged two of the three splits with 100 interactions (resulting in sets of 200 interactions) and used the third split for cross-training.

For a second test set consisting of PPI structures recently added to the PDB, we first followed the same protocol of PPI data extraction as in the previous chapter, only with a newer version of the PDB (Sept 2013). Once the SameSeq clustering was completed, we redundancy it internally exactly as the older training data set before and then also externally against the older redundancy reduced training data set with the same criteria used before (no protein more similar than HVAL 20). This resulted in 187 new non-redundant interacting sequence pairs.

3.2.2 Data Subsampling

Taking all the interacting residue pairs as the ‘positive’ class and all others as the ‘negative’ class would not only have lead to a data set too large for fine grained parameter and feature selection, but also to an extreme over-representation of pairs from interactions with longer sequences. For example, an average interaction in which both sequences are 200 residues long leads to $200*200 = 4*10^4$ residue pairs, whereas a $1000*1000$ interaction results in 10^6 pairs, accounting for $10^6/(4*10^4) = 25$ 'average' interactions. We therefore limited the number of residue pairs per interaction to the geometric mean of both sequences, i.e. $n_i = \sqrt{len(s_1)len(s_2)}$, where n_i is the number of pairs to sample for the i-th interaction, len a function returning the length of a sequence and $s_{1,2}$ the first and second sequence of the interaction. In order to maximize resource utilization, we additionally multiplied the n_i's with a constant factor (5 for feature selection and parameter optimization and 40 for the final network learning). Obviously, this did not alter the relative contribution of each interaction to the full data set. We also always made sure that the fraction of positive to negative residue-residue interactions remained the same for each sequence-sequence interaction and was not altered due to subsampling the residue-residue pairs in a given interaction.

3.2.3 Evaluation Measures

Determining whether a residue pair interacts or not is a standard two-class prediction problem. This means we can apply default evaluation measures. In particular, we used recall-precision curves as they focus on the 'positive' class, i.e. interacting residue pairs. They are calculated as follows: Predicting a test set associates each residue pair with a score reflecting the reliability that the residues interact. Choosing a threshold separates the residue pairs with a score higher than this threshold from those that fall below it. Precision is then defined as the number of interacting pairs above the threshold divided by all the pairs above the threshold. Recall is the number of interacting pairs above the threshold divided by all interacting pairs. In order to summarize the performance on a data set in one number, we calculate the average precision up to a recall of 50% (recall step size of 5%).

3.2.4 Features

We encoded every residue in the data set as a vector of floating point numbers. These numbers, called 'features', were derived from various data sources and existing sequence-based prediction methods. In the following, we give a short overview. For details, please see the respective publications.

Amino Acid Sequence. Every residue corresponds to 1 of 20 amino acids. This information was encoded in 20-dimensional vector in which each amino acid corresponded to one dimension. The amino acid of the residue to encode had a value of 1, all others a value of 0.

Evolutionary Profile Frequencies. We PSI-BLASTed [Altschul et al., 1997] each sequence against Uniprot [Consortium, 2011], redundancy reduced to 80% max. pairwise sequence identity with 3 iterations and an E-Value threshold of 10^{-3}. This resulted in a sequence 'profile' that contained for each residue a frequency for each of the 20 amino acids reflecting how often the respective amino acid was conserved at this position in similar sequences. Following the same principle as before, every residue was consequently encoded as a 20-dimensional vector where each dimension represented one amino acid and each value the frequency of the amino acid.

Evolutionary Profile Scores. As for 'Evolutionary Profile Frequencies', but using normalized PSI-BLAST substitution scores instead of substitution frequencies.

Evolutionary Profiles (Strict). As in the two previous Sections, but with a different E-Value cutoff (10^{-25}). Hence, this leads to two new features, each corresponding to 20 dimensions.

Evolutionary Profile (HHBlits). As for the previous three features, but using HHBlits [Remmert et al., 2012] to create the evolutionary profiles instead of PSI-BLAST. This leads to four new features, each corresponding to 20 dimensions.

PSIC. As described in [Sunyaev et al., 1999, Bromberg and Rost, 2007], an evolutionary profile is generated from the hits found in a PSI-BLAST run like for the previous features, but amino acids at a certain position are scored according to position specific independent counts (PSIC). This means, for example, that large clusters of highly similar sequences in the alignment may have the same contribution as a single sequence that is very different from others. We calculated this feature as in [Bromberg and Rost, 2007] and used PSI-BLAST with parameters as described for 'Evolutionary Profile Frequencies'.

reprof. Reprof (unpublished) is a neural network based tool to predict the solvent accessibility and secondary structure for each residue in a protein sequence. It was developed and trained in the same way as PROF PHD in [Rost and Sander, 1993], but with newer data and an advanced neural network implementation [Nissen, 2003]. For each residue, it outputs the probability of belonging to one of three secondary structure element (helix, sheet, loop), together with an overall reliability index reflecting the confidence in the prediction. The degree of solvent accessibility is also predicted as a probability and accompanied by a reliability index.

ISIS. ISIS is another neural network based method that predicts for a protein sequence which residues are able to bind to other proteins [Ofran and Rost, 2007a]. Predictions have also been shown to correlate with hot spots [Ofran and Rost, 2007b]. ISIS outputs a single value reflecting the probability that the respective residue can bind other proteins, together with a score reflecting the reliability of the prediction.

metadisorder. Some parts of a protein may not fold into a stable structure. Such segments are 'disordered' and predicted by metadisorder [Schlessinger et al., 2009] by merging the output of various other tools predicting various disorder subtypes. Its output is a single value reflecting the degree of disorder, together with a reliability index.

SomeNA. SomeNA (diploma thesis in computational biology by Peter Hoenigschmidt at the TUM) is an overhauled version of the neural network based classifier DISIS [Ofran et al., 2007] and predicts whether a residue can bind to DNA and/or RNA molecules. Amongst various other values, it outputs the probabilities that a residue binds to DNA and RNA. We only use those 2 values.

SNAP2. SNAP2 (master thesis in computational biology by Maximilian Hecht at the TUM), a faster and more accurate version of SNAP [Bromberg and Rost, 2007], predicts the impact of function on the protein upon changing the amino acid type of one residue to another amino acid. In order to obtain a single value for a residue, we changed each residue to Alanine and recorded the impact. We used this value as a feature.

Chemical Features. A residue was encoded with a number of chemical properties such as charge, volume, hydrophobicity or ability to break a helix.

Global Features. 'Global' features do not change their value across the residues of a protein. A simple example is the length of a protein. Global features can help estimating the pairing susceptibility of a protein, the fraction of interface residues and thus, on the level of protein pairs, the ratio of interacting to non-interacting residue pairs.

We encoded the protein length with 20 dimensions. The first value is set to 1 if the protein is longer than 60 residues, otherwise it is 0. The second is 1 if the protein is longer than 120 residues, otherwise 0. The third is 1 if it is longer than 180, etc. . Next, we counted the propensity of each amino acid in the entire sequence, again resulting in 20 values between 0.0 and 1.0 for this feature called 'amino acid composition'. Similarly, we calculated the secondary structure composition based on the predictions of reprof. SomeNA, ISIS and metadisorder values were averaged over the entire sequence, resulting in three values that reflect the overall tendency to bind RNA/DNA, other proteins or to avoid a fixed structure, respectively. A final tool, LocTree2 [Goldberg et al., 2012], predicted the localization of a protein. It outputs one class for each of the three kingdoms (archaea, bacteria, eukaryota) out of a total of 27 classes. Hence, LocTree2 predictions were encoded in 27 values, with 3 of them having a value between 0.0 and 1.0 reflecting the probability of the predicted class and all others being 0.

3.2.5 Residue-residue Feature Windows

For all proteins in our data sets, we split the protein sequences into overlapping fragments (windows) of a fixed size, in which the central amino acid determines which residue the window maps to and all flanking amino acids provide additional context information that putatively help with the prediction of the central residue. Replacing the actual residues in a window by features as listed above creates exactly *n* vectors of equal length, where *n* is the length of the sequence (features reaching out of the sequence initialized to reasonable default values).

For encoding residue-residue pairs, we concatenate the two feature vectors of the two windows under consideration. In order to avoid protein order dependent bias, we concatenated in both ways, i.e. residue-residue pair A-B was encoded as features(A)+features(B) and features(B)+features(A), resulting in two feature vectors per pair. The class is determined by whether the two residues interact or not. We empirically estimated the ideal single-protein window size by selecting the best performing size out of candidate sizes 3,5 and 7 for every network that we trained.

3.2.6 Artificial Neural Networks

Having defined a vector representation for every residue-residue pair in our data set, we could apply standard machine learning tools to train a predictive classifier capable of scoring new residue-residue pairs with respect to their probability to interact. Like many before us, we chose artificial neural networks. We defined a single hidden layer with a to-be-optimized number of hidden units and two output units, one for each class (interacting/not interacting). This fully connected feed-forward network was trained with the standard backpropagation algorithm implemented in the FANN library [Nissen, 2003].

3.2.7 Network Parameter Optimization

For every network that we trained, we first presented the training split of our data set to the network and applied backpropagation after each sample. Then we predicted the cross-validation set with the network and recorded the average precision until a recall of 50% (recall step size of 5%; see 'Evaluation Measures'). This is what we refer to as one 'epoch'. Next, we presented the training data again, and again calculated the average precision on the cross-training data. We stopped repeating this when the network error was more than 2% average precision below the maximum observed earlier for longer than 20 epochs. This was observed to be near-optimal on a smaller data set.

For each learning task, we also optimized the number of hidden units, the learning rate of the network (i.e. the magnitude of the change of each connection weight after each sample during backpropagation) and the learning momentum (includes not only the last, but also the second last weight when calculating the new weight of a connection; this sometimes avoids getting stuck in local minima of the target function). Possible values were: for hidden units 5,10,25 and 50; for the learning rate $10^{-2,\dots,-5}$; and for the momentum 0.0 ad 0.7.

For each learning task, we consequently trained $4*4*2 = 32$ different networks and selected the one with the best average precision on the cross-validation set.

3.2.8 Feature Selection

In order to further optimize the neural network, we tried to eliminate irrelevant and redundant features. To this end, we performed a wrapped greedy forward feature selection. This means we first fully optimized a neural network for each feature individually, i.e. we trained $|\text{window sizes}| * |\text{network parameters}| = 3*32 = 96$ neural networks for each of the features described before, using only the respective feature to encode residue-residue pairs. Then we selected the feature leading to the network with the best average precision on the cross-training set and added it to our final feature set. In any iteration thereafter, we again evaluated all features (except for those in the final set), but always in combination with those already in the final set. The procedure stopped when all features had been selected into the final set. The feature combination leading to the highest average precision over all the iterations was chosen to train our final model. We performed this procedure three times, i.e. separately for for each train/cross-train data set.

To train our final model, we increased the size of our data set as described in 'Data Sampling' and again optimized the window size and other neural network parameter. A new residue-residue window was predicted by predicting it with all three networks (one for each data set fold) and averaging their output.

3.2.9 EV-fold

We compared our approach to EV-fold [Marks et al., 2011], a protein contact prediction method with original applications in the prediction of contacting PPI residues. It is based on correlated mutations, i.e. changes in the sequence of one protein that are accompanied by changes in the other proteins. The idea is to concatenate the sequences of two interacting proteins if they come from the same species S1. A homologous interaction is then defined as two proteins from another species S2 that interact and are sequentially similar to both proteins of the first interactions. Concatenating the two S2 proteins allows us to create a single pairwise sequence alignment of the interaction. Repeating this for all species with the S1 interaction as the target, we can create a multiple sequence alignment for the S1 interaction and finally apply a standard single-protein contact prediction method. Interacting PPI residues are those that cross the sequence border in the multiple sequence alignment of concatenated proteins.

In practice, we first searched for similar sequences in the complete UniProt database for each of our proteins using JackHMMer [Eddy, 2011]. We experimented with three different E-Value cutoffs ($10^0, 10^{-3}, 10^{-25}$) and found 10^{-3} to perform best in downstream assessments. All filters that would limit the number of sequence alignments in the output were turned off. This generated two sequence alignments for each interacting protein pair. The two alignments were merged by matching the sequences from either alignment by their taxonomic ID. A sequence in the first alignment that had a sequence from the same organism in the second alignment was concatenated with this second sequence (one per taxonomic ID). Sequences without matches were discarded. We then applied EV-fold to the merged alignment (default parameters) and removed all residue pairs in which both residues came from the same protein. This created a scored list of residue-residue pairs for every protein pair that could be evaluated exactly as described before.

3.3 Results and Discussion

We have used our data set from the previous chapter to create a collection of interacting and non-interacting PPI residue pairs. Various features encode the pairs as high-dimensional numerical vectors which we used in turn to train an artificial neural network (NN) based classifier (Methods). For new PPIs, it can predict from sequence which residue-residue pairs are interacting and which are not. We compare this approach to EV-fold [Marks et al., 2011], a recent state-of-the-art method to predict contacts from sequence alignments (Methods).

3.3.1 Evaluation on Hold-out Set

148 non-redundant interactions were completely held-out during training and could be used for independent evaluation. We predicted all of their residue-residue pairs which come from two different proteins with both, our neural networks and EV-fold. For each method, this associated each pair with a score and allowed us to calculate recall-precision curves (Methods).

Fig. 3.1 (solid lines) shows that our method greatly outperforms EV-fold for all but the very lowest levels of recall in which precision values still have large error margins. The coverage of proteins that contain high-confidence predictions climbs faster for EV-fold than for our method (dotted lines). This means the residue-residue pairs that receive the highest scores are limited to a few proteins. However, even predictions with lower reliability (e.g. those that cover 100% of all proteins) are more accurate than the high-reliability predictions by EV-fold.

Both methods improve greatly over random (0.26%; roughly corresponds to precision at 100% recall). In absolute terms, however, precision values are low and do not climb above 6-7% in precision regions with small error margins. This is due to the extreme ratio of interacting to non-interacting pairs (roughly 1 in 400). Consequently, the primary use case of our method is to assist in choosing candidate residues for site-directed mutagenesis experiments (e.g. Alanin scans). If structures of the proteins are available, it may also help in the ranking of docking solutions.

3.3.2 Evaluation on New PPIs

168 non-redundant PPIs were added to the PDB since we started working on our predictor. We evaluated them in the same way as the data set in the previous Section (Fig. 3.2). Results are largely consistent with those of the hold-out set so that our conclusions remain the same.

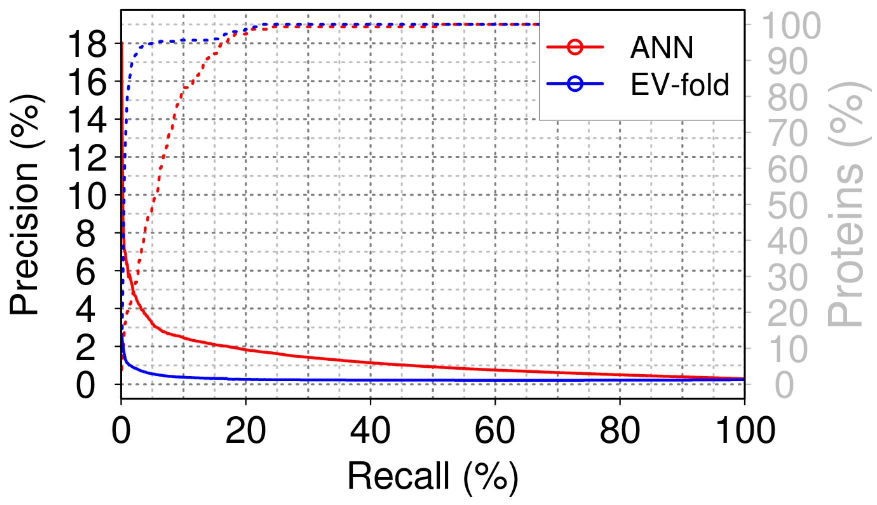


Figure 3.1: Recall-precision curves for the hold-out data set. The plot compares EV-fold to our new artificial neural network (ANN) based method on the data set that was held out during training. Solid lines are recall-precision curves. Dotted lines indicate the percentage of proteins that have residue-residue pairs with a score above the threshold corresponding to the current recall level.

3.3.3 Limiting Factors for EV-fold

There are a number of reasons why EV-fold performs dismally on our data compared to the proteins it has been applied to before [Weigt et al., 2008]. First, the performance is critically dependent on the number of sequences in the alignment. This number, however, is always limited by the number of hits of the protein with the fewer hits. The size of the alignment is measured by EV-fold as the ‘effective sequence count‘. It should ideally be orders of magnitude larger than the length of the protein. For us, this was never the case, however, and exceeded the length of the protein in only about 25% of the cases. Secondly, the signal for correlated mutation likely stands and falls with the interaction of the two proteins. If we concatenate two proteins from the same species that are distantly related to the target proteins and include the merged sequence in our alignment, we implicitly assume that the two proteins are interacting. This is probably often not the case. Another unresolved issue is how to make scores comparable across proteins. So far no universal rule is available.

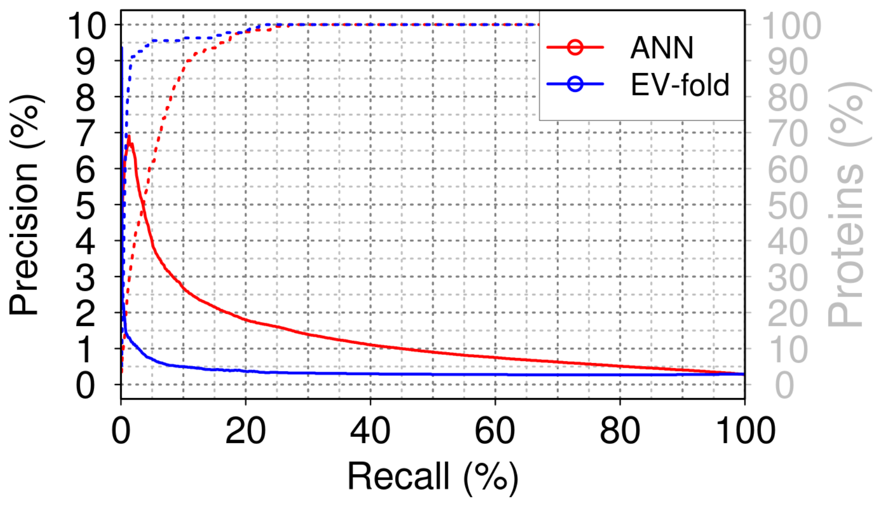


Figure 3.2: Recall-precision curves for new PPIs. The plot compares EV-fold to our new artificial neural network (ANN) based method on PPIs that have recently been added to the PDB and that do show sequential similarity to PPIs used for training (Methods). Solid lines are recall-precision curves. Dotted lines indicate the percentage of proteins that have residue-residue pairs with a score above the threshold corresponding to the current recall level.

3.3.4 EV-fold as a Feature

Although EV-fold was not competitive as a stand-alone predictor, it may still have been helpful as a feature. We tested this by encoding the output of EV-fold as a feature for our predictor. For example, we mapped each score of a residue-residue pair to its percentile among the scores of the entire training data set. This score could then be treated as a feature of the residue-residue pair. When including this new feature in our feature selection (Methods), we could observe it to slightly improve over random in the beginning, but it was always outperformed by other features and not selected.

3.4 Conclusion

We used a battery of sequence-derived features to train artificial neural networks that can predict from sequence whether two residues from two different proteins interact or not. It outperforms a recent state-of-the-art contact predictor that was originally used for exactly this problem - PPI contact prediction. Due to the extreme imbalance of interacting to non-interacting pairs, absolute values for precision are low over all recall levels. Nevertheless, they improve greatly over random. This makes the application of our method an ideal primary step in site-directed mutagenesis experiments targeting PPI interfaces. Another potential, yet still unexplored, use could be the ranking of candidate docking solutions.

Chapter 4

Acceleration of the Original Profile Kernel*

4.1 Outline

Our residue-residue based predictor trained in the previous chapter performed well on the level of residue pairs, but was not competitive for discriminating between interacting and non-interacting protein pairs (not shown). Initial experiments with the original profile kernel [Kuang et al., 2004] showed great potential for this task, but its CPU requirements render it too slow for practical applications of large-scale classifications. Here, we introduce several software improvements that enable significant acceleration. Using various non-redundant data sets, we demonstrate that our new implementation reaches a maximum speed-up as high as 14-fold for calculating the same kernel matrix. Some predictions are over 200 times faster and render the kernel as possibly the top contender in a low ratio of speed/performance. Additionally, we explain how to parallelize various computations and provide an integrative program that reduces creating a production-quality classifier to a single program call.

4.2 Introduction

Profile kernels provide state-of-the-art accuracy. The characterization of proteins often begins with their assignment to different classes. Examples for such classes are protein families, distant structural relations, or sub-cellular localization.

*This chapter is based on the publication [Hamp et al., 2013a]

GO, the Gene Ontology [Ashburner et al., 2000], is the most comprehensive functional vocabulary that defines over 38,000 different 'GO terms', i.e. classes into which a protein could be grouped. The simplest classification is through homology-based inference [Rost, 1999, Rost and Liu, 2003]. A PSI-BLAST [Altschul et al., 1997] or HHBlits [Remmert et al., 2012] query against a database with annotations such as Swiss-Prot (Consortium 2011) creates a list of proteins that have reliable experimental annotations and are sequentially similar to the target. Choosing the annotation of the best hit for the query then constitutes one simple means of annotating function [Radivojac et al., 2013].

Such a naive prediction method has disadvantages: query results are usually ordered by the e-value or the HVAL [Rost, 1999] of the best local alignment. This is not the best choice for all classification problems. A membrane-integral domain, for example, might be located at the N-terminus of the target, whereas the alignment with the best hit begins near the C-terminus. Therefore, advanced machine learning methods such as Neural Networks or Support Vector Machines (SVMs) often outperform simple homology-based inference, even for very small classes [Hamp et al., 2011].

These methods represent proteins in a high-dimensional space, as given, for example, by the frequencies of the 20 amino acids in a protein. Some of the most popular and accurate classifiers are sequence-profile based kernels in conjunction with SVMs [Kuang et al., 2004, Rangwala and Karypis, 2005a, Weston et al., 2005, Liu et al., 2008a, Man-Wai, 2008, Thanh and Rui, 2009, Toussaint et al., 2010] . They do not require a protein to be represented explicitly, but only implicitly via dot-products to other proteins. Without this limitation, even the score of a local alignment can be turned into a kernel function and harness the advantages of the maximum-margin hyperplanes computed by SVMs [Rangwala and Karypis, 2005a, Man-Wai, 2008, Thanh and Rui, 2009].

Methodological limitations difficult to address. This advantage, however, comes at a computational cost. The dot products required for training are stored as kernel matrices, which are quadratic in the number of training samples. Furthermore, in order to classify a new query, dot products have to be calculated with respect to all support vectors. Their number, however, is typically proportional to the amount of classes and template proteins. This puts strong limitations on data sets sizes and some kernels that are sufficiently fast for today's searches might become infeasible soon because the growth of the bio-sequence data far outpaces the growth of computing hardware. Current solutions to the problem of data set sizes that are preventative for training include the use of linear SVMs, keeping only parts of the kernel matrix in memory or massive parallelization. All three options are mostly inapplicable to profile kernels. The first two (linear SVMs; caching the kernel matrix) are complicated because explicit sample vectors are either unknown or too large and calculating the same kernel values multiple times slows down training unacceptably. The second (parallelization) has, to the best of our knowledge, not been implemented by any state-of-the-art method, yet. In some cases, predictions can be accelerated much more elegantly: if a kernel operates directly in the feature space, the normal vector separating one class from the other may be calculated explicitly, instead of implicitly via support vectors and associated weights. This reduces predicting a new query to calculating a single dot product.

Accelerating the original profile kernel. Here, we show how to apply these concepts to the kernel introduced by the Leslie group [Kuang et al., 2004]. It is arguably the most popular profile-based kernel today and its outstanding performance for many tasks has been repeatedly confirmed ([Rangwala and Karypis, 2005a, Weston et al., 2005, Thanh and Rui, 2009, Toussaint et al., 2010]. We have recently applied it in the development of a state-of-the-art method for the prediction of sub-cellular localization, LocTree2 [Goldberg et al., 2012]. On top of its high performance, the original profile kernel has other advantages, such as the ability to extract sequence motifs from trained SVMs. In particular, its hyper-planes can be made explicit as long as also the underlying k-mer trie based algorithm is modified accordingly. Consequently, our first and most important improvement is calculating the matrix product of input profiles and pre-computed SVM normal vectors at full use of the k-mer trie based data structure. This corresponds to an efficient and highly parallel classification of many protein profiles with many SVMs at the same time, without the need for multiple CPU cores. Secondly, addressing the training phase, we can now distribute the computation of a single kernel matrix to an arbitrary amount of parallel processes. Due to optimizations of procedures required both for training and testing, also existing un-parallelized routines now run about five times faster than in the original implementation. Finally, we have combined all the necessary steps for training a classifier in a single program. It automatically calculates the kernel matrix, learns a user-defined SVM-based multi-class model, extracts and compresses the SVM normal vectors and stores everything as a ready-to-use predictor.

4.3 Methods

4.3.1 Original Profile Kernel

The algorithm to calculate the kernel matrix with the original profile kernel has been thoroughly introduced in [Kuang et al., 2004]. It maps every profile to a 20^k-dimensional vector of integers. Each dimension represents one k-mer of k consecutive residues and a particular value gives the number of times this k-mer is conserved in a profile of related proteins. Conservation is calculated as the sum of the substitution scores for each residue in the k-mer profile and has to fall below a certain user defined threshold σ. Conserved k-mers are found very efficiently by traversing a trie based data structure (Fig. 4.1). Each leaf corresponds to one of the 20^k dimensions and defines a set of conserved k-mers. With this set, the kernel matrix is updated so that each kernel matrix value is increased by the number of k-mers shared by the two corresponding profiles at that leaf.

In the following, we describe our own modifications and extensions to this approach. Where appropriate, a non-detailed outline is followed by a detailed technical description. Our speed-up focuses on two different steps in the profile kernel algorithm: the trie traversal and the matrix update. Combined, these two always account for about 90% of the overall runtime, but their individual fraction depends on the respective kernel parameters and input. On average, we estimate that the two contribute equally to the runtime.

4.3.2 Modification 1: Reducing Kernel Matrix Updates to Matrix Multiplication.

Outline. At each leaf node during the traversal of the k-mer trie, a set of conserved k-mers of the input profiles has remained (Fig. 4.1). At this point, the original profile kernel updates the kernel matrix: if, e.g., k-mer 1 belongs to input profile 3 and k-mer 2 to input profile 8, then the value of the kernel matrix at row 3, column 8 has to be increased by 1. Repeating this for all k-mer pairs updates the entire kernel matrix for this particular leaf node and the traversal continues. This operation can be greatly simplified: first, we count how many conserved k-mers each profile has at a particular leaf node. Only the profiles with non-zero counts are added to a sparse matrix in which each row stands for a profile and each column for a particular leaf. (To save space, the matrix is stored as a 'coordinate list', i.e. as a list of triplets of the form [x-coordinate, y-coordinate, value].) For most leaves, we only add elements to this sparse matrix; only when the buffer is almost full, we update the actual kernel matrix. This can be done in arithmetically the same way as described above, but

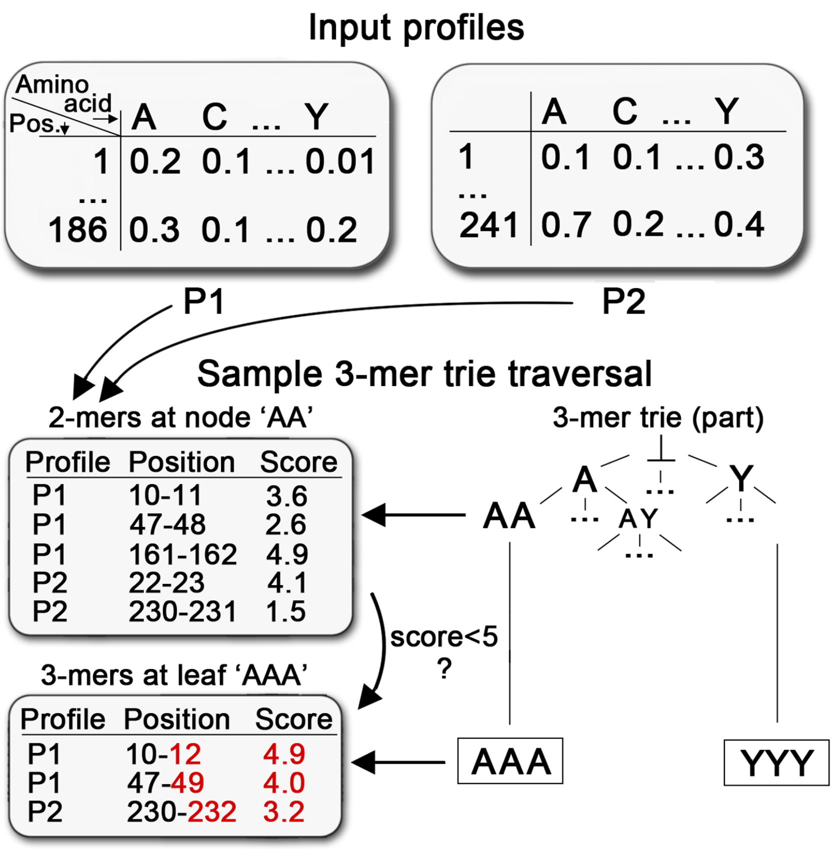


Figure 4.1: Sample *k*-mer tree traversal. Sketched is one part of a 3-mer trie traversal with two input profiles (P1 and P2). These profiles were generated with proteins that were 186 (P1) and 241 residues long (P2; tables on the top). During traversal, some conserved multi-mers remain at each node that fall below the substitution score threshold σ. The 'Sample 3-mer trie traversal' illustrates the transition from two-letter node 'AA' to node 'AAA' ('AAA' is also a leaf, because $k = 3$). At node 'AA', five 2-mers have remained from previous transitions (root -> 'A' ->'AA') that still fall below the substitution score threshold $\sigma = 5$. In the transition to node 'AAA', each such 2-mer is extended to a 3-mer and each score re-calculated (*k*-mer extension and new scores in red). 3-mers with a score>5 are discarded (2/5) and those that remain (3/5) are used in the kernel matrix update. Afterwards, the traversal continues until reaching the lexicographically last leaf ('YYY').

operationally by a very efficient self-multiplication of the buffered sparse matrix and an on-the-fly addition of the result to the kernel matrix.

Detailed description. Let $\mathbb{P}$ be the space of all possible protein profiles and $p_1, ..., p_m \in \mathbb{P}$ be the input protein profiles. Let $length : \mathbb{P} \to \mathbb{N}$ be a function that returns the length of each profile (i.e. the length of the sequence that a profile p_i was generated with). Let further $Kmers = \{(l,r) | 1 \leq l \leq m \wedge 1 \leq r \leq length(p_l) - k + 1\}$ be the set of all k-mer starting positions with k being the user-defined k-mer length and let $A = [a_{i,j}]_{m \times m}$ be the kernel matrix.

At a leaf node L, there remains a set $Kmers_L \subset Kmers$ of k-mers whose cumulative substitution score is lower than a pre-defined threshold [Kuang et al., 2004]. In the original profile kernel implementation, the subsequent kernel matrix update is performed in the following way:

```
for all $(l,r) \in Kmers_L$ do
    for all $(l',r') \in Kmers_L$ do
        $A_{l,l'} \leftarrow A_{l,l'} + 1$
    end for
end for
```

By definition, a kernel matrix value $A_{l,l'}$ is the result of a dot product $\phi(p_l)\phi(p_{l'})$ and one of the 20^k dimensions of any $\phi(p_i)$ is processed at every leaf node. However, the implementation above never calculates a dimension of a $\phi(p_i)$ explicitly, but directly updates the kernel matrix. Hence, it could even be argued to be a 'kernel trick', which is commonly regarded as an acceleration of a kernel. In the following, however, we show how its replacement with the explicit feature mapping and the subsequent reduction to matrix multiplication leads to a more efficient procedure.

First, note that r and r' are never used and can be ignored. Secondly, after one entire inner **for all** loop, the value $A_{l,l'}$ has been increased by the number of k-mers that have remained from profile $p_{l'}$. This number can be calculated as $s^L(l') = \sum_{(l,r) \in Kmers_L} I(l = l')$, where $I(x)$ is 1 if x is true and 0 otherwise, and allows re-writing the matrix update in the following way:

```
$\mathbf{v^L} \leftarrow \langle s^L(1), ..., s^L(m) \rangle$
for all $(l,r) \in Kmers_L$ do
    for $j \leftarrow 1; j \leq m; j \leftarrow j+1$ do
        $A_{l,j} \leftarrow A_{l,j} + v^L_j$
    end for
end for
```

In this new version, instead of iterating over $Kmers_L$ in the inner loop, we only iterate over the number of profiles. Since s^L only depends on $Kmers_L$ and not on A, it can be pre-computed before the matrix update and the time complexity of the latter is reduced from $O(|Kmers_L|^2)$ to $O(m \cdot |Kmers_L|)$. Now, note again that v^L_j is added to $A_{l,j}$ exactly as often as there are kmers from profile l in $Kmers_L$. This means we can also re-write the outer for loop:

```
$\mathbf{v^L} \leftarrow \langle s^L(1), ..., s^L(m) \rangle$
**for** $i \leftarrow 1; i < m; a \leftarrow i+1$ **do**
    **for** $j \leftarrow 1; j < m; b \leftarrow j+1$ **do**
        $A_{i,j} \leftarrow A_{i,j} + v_i^L \cdot v_j^L$
    **end for**
**end for**
```

Obviously, the time complexity for this operation is now $O(m^2)$. Next, assume that the matrix update is not executed at every leaf node, but instead only every x leaf nodes ($1 \leq x \leq 20^k$). At leaves without an update, the vector $\mathbf{v^L}$ is added to a matrix $\tilde{V}$ as a new row. When reaching an update leaf, $\tilde{V}$ has the form $\tilde{V} = [\tilde{v}_{i,j}]_{x \times m}$ and the bulk matrix update can be performed in the following way:

```
**for** $i \leftarrow 1; i < m; i \leftarrow i+1$ **do**
    **for** $j \leftarrow 1; j < m; j \leftarrow j+1$ **do**
        $A_{i,j} \leftarrow A_{i,j} + \tilde{V}_{1,i} \cdot \tilde{V}_{1,j} + ... + \tilde{V}_{x,i} \cdot \tilde{V}_{x,j}$
    **end for**
**end for**
```

Simplifying even further, we obtain: $\tilde{V}_{1,i} \cdot \tilde{V}_{1,j} + ... + \tilde{V}_{x,i} \cdot \tilde{V}_{x,j} = \tilde{V}^T_{i,1} \cdot \tilde{V}_{1,j} + ... + \tilde{V}^T_{i,x} \cdot \tilde{V}_{x,j} = (\tilde{V}^T \tilde{V})_{i,j}$ and the entire matrix update reduces to matrix multiplication:

$A = A + \tilde{V}^T \tilde{V}$

This operation can be implemented very efficiently with a slight modification of the IKJ algorithm introduced in [Sulatycke and Ghose, 1998] so that matrices are stored in a sparse format and cache efficiency is preserved (Section 4.3.8 for more details). As each value $(\tilde{V}^T \tilde{V})_{i,j}$ is calculated separately by the algorithm, there is also no need for a temporary matrix to store $\tilde{V}^T \tilde{V}$. $(\tilde{V}^T \tilde{V})_{i,j}$ can be added directly to the kernel matrix. In our actual implementation, $\tilde{V}$ is a triplet-based sparse matrix with a maximum size of 300MB. In order to make sure that every row addition still fits into memory, the current matrix size is checked at every leaf node and an update is performed as soon as more than 67% of the buffer are used or the last leaf is reached. After the matrix update, the buffer is emptied and the traversal continues until all leaves are reached.

Interpreting the above in terms of the explicit feature mapping ϕ, we calculate one of the 20^k dimensions of every feature vector $\phi(p_i)$ at each leaf node L and store it in $\tilde{V}$. Before a kernel matrix update, $\tilde{V}$ contains x dimensions of every $\phi(p_i)$ and the actual matrix update calculates all of their pairwise dot products.

4.3.3 Modification 2: SSE2 Instructions and New Data Structure During Tree Traversal

Profiling the profile kernel executable with perf (part of the Linux kernel) revealed that during traversal of the k-mer trie, most of the time is spent on checking whether the substitution score of the k-mers is below the user-defined threshold. Implementing this double comparison with Streaming SIMD Extensions 2 (SSE2) instructions, two values can be compared in one CPU cycle, thus significantly improving overall runtime.

4.3.4 Modification 3: Multi-process Kernel Matrix Calculation

Outline. Too large kernel matrices can no longer be kept in main memory and may require several days for computation on a single CPU. Therefore, we have added the feature to split this task among several individual processes. Given m training profiles, we first assign each to one of n groups of size p=m/n (n is user defined). Then we compute the dot products of the profiles for one group to those of another group. This creates a p x p sub-matrix of the original kernel matrix. Repeating this for all O(p^2) possible group pairs calculates all sub-matrices which then have to be joined together to build the original kernel matrix. The creation of a single sub-matrix can be accelerated by only computing dot products between profiles from different groups and again by applying Modifications 1 and 2.

Calculating a Kernel Submatrix. In certain situations, only a consecutive submatrix $A'[a_{i,j}]_{n \times n'}$ of A has to be calculated, i.e. $A'_{i,j} = A_{n_{start}+i-1, n'_{start}+j-1}$ where n_{start} denotes first row in A and n'_{start} the first column.

A certain value $A_{i,j}$ only depends on the columns i and j of matrix $\tilde{V}$. Hence, in order to reduce $\tilde{V}$ to the required columns for A', we project it to two submatrices $\tilde{V}' = [\tilde{v}_{i,j}]_{x \times n}$ and $\tilde{V}'' = [\tilde{v}_{i,j}]_{x \times n'}$ with $\tilde{V}'_{i,j} = \tilde{V}_{i,n_{start}+j-1}$ and $\tilde{V}''_{i,j} = \tilde{V}_{i,n'_{start}+j-1}$. Submatrix A' can then be calculated as:

```
for $i \leftarrow 1; i < n; a \leftarrow i+1$ do
    for $j \leftarrow 1; j < n'; j \leftarrow j+1$ do
        $A'_{i,j} \leftarrow A'_{i,j} + \tilde{V}'_{1,i} \cdot \tilde{V}''_{1,j} + ... + \tilde{V}'_{x,i} \cdot \tilde{V}''_{x,j}$
    end for
end for
```

This can be simplified to:

$$A' = A' + \tilde{V}'^T \tilde{V}''$$

In practice, $\tilde{V}'$ and $\tilde{V}''$ can be calculated directly by accepting two sets of input profiles $p \subset \mathbb{P}$ and $p' \subset \mathbb{P}$ that correspond to the rows and columns of A', respectively.

When traversing the trie, conserved k-mers of p are added to $\tilde{V}'$ and conserved k-mers of p' to $\tilde{V}''$, as described in Section 4.3.2 for $\tilde{V}$.

4.3.5 Modification 4: Predicting New Queries Through Normal Vectors (Application of Model)

Outline. In contrast to the kernels described elsewhere [Rangwala and Karypis, 2005a], the original profile kernel introduced by the Leslie group allows the explicit calculation of the discriminative normal vector w of a SVM. The 'SVM score' of a new query profile p, i.e. its scaled distance to the hyper-plane, can then be calculated as a single dot product $s = w * \phi(p)$, where $\phi(p)$ is the feature vector of p and $\phi(p)_j$ the number of conserved k-mers at leaf node j. Before, dot products to all support vectors were required (Section 4.3.5).

In order to extract normal vectors from trained SVMs, we can again use the k-mer trie. A single traversal can determine the normal vectors of many SVMs and create a 'normal matrix' in which each row represents one of 20^k k-mers and each column one normal vector (Section 4.3.5). This greatly accelerates the additional training time, as classification problems are hardly ever limited to two classes in computational biology.

In order to calculate the SVM score $s = w * \phi(p)$ of a single query p and a single normal vector w, we multiply w_j with $\phi(p)_j$ at each leaf node j and add the result to s (s is initialized to 0). By using the normal matrix (above), this can be modified so that the scores of all SVM normals are updated at each leaf node, resulting in a vector of SVM score for query p. Traversing the trie with multiple queries at once consequently generates a matrix of SVM scores in which each row represents a target profile and each column a SVM.

With another extension similar to Modification 1, we can again store k-mer counts in a sparse matrix and use matrix multiplication to update the SVM scores matrix (Section 4.3.5). SSE2 instructions again accelerate the transition from one node to the next (Modification 2).

Calculating the matrix of normal vectors. With one profile kernel matrix, several binary Support Vector Machines (SVMs) can be trained. The exact number of SVMs depends on the number of classes and the multi-class scheme. For a traditional 1-vs-All classifier, e.g., we train as many SVMs as there are classes.

Classifying a new target profile p_{m+1} requires calculating its scaled distance to the SVM hyperplanes. One distance is computed with the well-known formula $s_{m+1} = w \cdot \phi(p_{m+1}) = \sum_{i=1...m} \alpha_i \phi(p_i)\phi(p_{m+1})$ where m is the number of support vectors, α_i the weight, p_i the original profile and $\phi(p_i)$ the feature vector of support vector i. This means s_{m+1} can either be calculated via support vectors (Modification 6) or via a pre-computed normal vector w. In the following, we describe how to use the k-mer trie traversal to efficiently calculate several of these normal vectors in one run with support vector profiles and associated α's as input. In the next section, we apply this matrix of normal vectors to predict new queries.

Let $\mathbb{P}$ be the set of all possible protein profiles and $length : \mathbb{P} \rightarrow \mathbb{N}$ be a function that returns the length of each profile (i.e. the length of the sequence that a profile p was generated with). Let $(p_1, ..., p_m) \in \mathbb{P}^m$ be the original training profiles and $T = [t_{i,j}]_{m \times n}$ be the matrix of support vector weights extracted from n pre-computed SVMs, where each SVM corresponds to a column and each row to a training profile. One value $T_{i,j}$ indicates the weight of profile p_i in SVM j (the weights of non-support vectors are set to 0). Let $Q = \{i | 1 \leq i \leq m \wedge \exists j : T_{i,j} \neq 0\}$ be the indices of those training profiles that appear as a support vector at least once. Let $q = (q_i)_{q_i \in Q}$ be Q as an ordered tuple and let $m' = |Q|$ be the size of Q. Let further $Kmers = \{(l,r) | l \in Q \wedge 1 \leq r \leq length(p_l) - k + 1\}$ be the set of all k-mer starting positions of all support vector profiles. Our goal is to compute a matrix $W = [w_{i,j}]_{20^k,n}$ where each $W_{i,j}$ indicates the value of the i-th dimension in the normal vector of SVM j and is defined as $W_{i,j} = \sum_{z=1...m'} T_{q_z,j} \phi(p_{q_z})$ (follows directly from $w = \sum_{i=1...m} \alpha_i \phi(p_i)$). Once W is calculated, each column j corresponds to the normal vector of SVM j.

We start the trie traversal with all support vector profiles p_{q_i} and matrix T as input. At a leaf node L, there remains a set $Kmers_L \subset Kmers$ of k-mers whose cumulative substitution score is lower than a pre-defined threshold. This set $Kmers_L$ is converted to a vector v^L of size m so that $v^L_i = \phi(p_{q_i})_L = \sum_{(l,r) \in Kmers_L} I(l = q_i)$ (analogously to Section 4.3.2). One row W_L is then defined as $W_{L,j} = \sum_{z=1...m'} T_{q_z,j} v^L_z$.

As W can become very large and sparse, we only store non-zero entries of each row and write it directly to a file. As soon as it grows over a certain size, it is compressed with zlib [Gailly and Adler, 2012] (DEFLATE compression [Deutsch, 1996]) and the output is directed to a new file.

Applying the Matrix of Normal Vectors. In order to classify a new profile p_{m+1} (m is the number of training samples) with a single SVM, we need to calculate its scaled distance to the hyperplane. This distance, which will be referred to as the 'SVM score', is given by the formula $s_{m+1} = w \cdot \phi(p_{m+1})$, where w is the normal vector of the SVM and $\phi(p_{m+1})$ the feature vector of profile p_{m+1}. In practice, often many samples have to be classified by many SVMs at the same time, e.g. to predict all proteins of a genome in a multi-class setting. In the following, we show how to achieve this very efficiently in a single k-mer trie traversal and using the matrix of pre-computed normal vectors (previous Section).

Let $\mathbb{P}$ be the set of all possible protein profiles. Let $length : \mathbb{P} \rightarrow \mathbb{N}$ be a function that returns the length of each profile (i.e. the length of the sequence that profile p_i was generated with) and $(p_1, ..., p_m) \in \mathbb{P}^m$ be the target profiles. During the trie traversal, we use a matrix $\tilde{V}$ as a buffer which stores dimensions x_{start} to x_{end} of all $\phi(p_i)$'s (x_{start} and x_{end} are initialized to 0 and x, respectively; x is user-defined). Let $W = [w_{i,j}]_{20^k \times n}$ be the matrix of normal vectors, where k is the user-defined k-mer length and n the number of SVMs. Each column of W corresponds to a normal vector and each row to a k-mer. As W is usually too large to be kept entirely in memory, we use a temporary matrix $\tilde{W} = [\tilde{w}_{i,j}]_{x \times n}$ to store only the rows of W currently needed. Our goal is to calculate matrix $\tilde{D} = [d_{i,j}]_{m,n}$ where each $D_{i,j}$ indicates the SVM score of profile p_i after classification by SVM j, i.e. $D_{i,j} = w^j \cdot \phi(p_i)$, where w^j is the normal vector of the j-th SVM.

The trie traversal begins with all support vector profiles p_i. At a leaf node L, there remains a set $Kmers_L \subset Kmers$ of k-mers whose cumulative substitution score is lower than a predefined threshold. This set $Kmers_L$ is converted to a vector v^L of size m so that $v_i^L = \phi(p_i)_L = \sum_{(l,r) \in Kmers_L} I(l = i)$ (identical to previous sections). Each vector v^L is added to matrix $\tilde{V}$ as a new row. When rows x_{start} to x_{end} have been added, we load rows of matrix W with the same indices into $\tilde{W}$ and perform the following procedure:

```
**for** $i \leftarrow 1; i < m; i \leftarrow i+1$ **do**
    **for** $j \leftarrow 1; i < n; j \leftarrow j+1$ **do**
        $D_{i,j} \leftarrow D_{i,j} + \tilde{V}_{1,i} \cdot \tilde{W}_{1,j} + ... + \tilde{V}_{x,i} \cdot \tilde{W}_{x,j}$
    **end for**
**end for**
```

With a proof analogous to that of Section 4.3.2, this procedure reduces to

$D \leftarrow D + \tilde{V}^T \tilde{W}$

The matrix multiplication is carried out with the Eigen package ([Guennebaud et al., 2010]; Section 4.3.8 for more details). After the matrix update, the value of x_{end} is assigned to x_{start} and x_{end} is adjusted to fit the next update of D. $\tilde{V}$ and $\tilde{W}$ are emptied.

The above calculates the dot product between profile i and normal vector j, reduced to the summands x_{start} to x_{end} and adds the result to $D_{i,j}$. As the range $[x_{start}, x_{end}]$ has contained every k-mer index exactly once after the trie traversal, we obtain $D_{i,j} = w^j \cdot \phi(p_i)$ in the end.

4.3.6 Modification 5: Pipelining the Training and Prediction Process

Using both, the normal and the SVM score matrices described above, renders training and applying a multi-class profile kernel based classifier a tedious task that requires many data management steps. We have therefore pipelined the entire model creation and application workflow in a Perl script. In ‘model creation’ mode, it calculates the kernel matrix, uses it to learn an SVM multi-class classifier, extracts all weights for the Support Vectors from the resulting binary SVMs, converts these vectors into a matrix of normal vectors and stores all files and parameters that are required for predictions in a ‘model’ folder. The user only has to provide the input profiles with class labels and to specify the kernel parameters, a Weka [Frank et al., 2004] multi-class model and the number of processes to use. The ‘model application’ mode then uses this model to first calculate SVM scores with the normal matrix and the profile kernel and then forwards them to Weka which finally calculates the class probabilities of the queries.

4.3.7 Modification 6: Predicting New Targets with Support Vectors (Baseline Predictor)

In the original implementation of the profile kernel, there is no prediction mode. In order to classify a query, its profile has to be added those of all support vectors and the kernel matrix has to be re-calculated. Comparing the impact of our modifications to this approach would be unfair, because a simple prediction mode can easily be added: first, the kernel matrix updates can be restricted to dot products between targets and support vectors only; secondly, at each node in the k-mer trie, we can stop going down further in the trie as soon as there are no more k-mers left that belong to the queries. Another difference to normal matrix based predictions (Modification 5) is the output of dot products to support vectors instead of SVM scores. This can be neglected, however, because the time needed by external multi-class classifiers to calculate SVM scores given dot products is minimal. In the following, we will refer to this slightly altered original implementation as the ‘baseline’ implementation.

4.3.8 Matrix Multiplications

In the previous Sections, we either had to multiply a sparse matrix with a dense matrix or a sparse matrix with a sparse matrix. In both cases, the result had to be added to another third matrix.

For sparse-dense multiplications, we used the operations provided by the Eigen package [Guennebaud et al., 2010]. For sparse-sparse multiplications, however, all

Eigen operations resulted in at least one temporary matrix and a significant additional memory overhead. As the result of sparse-sparse multiplications is as big as the kernel matrix, we found this to be unacceptable and implemented the operation ourselves. As a template, we used the IKJ algorithm detailed in [Sulatycke and Ghose, 1998]. This algorithm described the cache efficient multiplication of a sparse matrix with a dense matrix, but we extended it to the sparse-sparse case. It turned out to not only save memory but also CPU time in comparison to Eigen.

So far, we stored sparse matrices as coordinate lists. Before a multiplication, we convert them to the following format (runtime and space complexity of the conversion is trivially linear in the number of coordinates).

- $M_{sparse} \rightarrow data$: an array of all non-zero data values of M_{sparse} in row-major format, i.e. the series of non-zero values obtained by reading the first row of M_{sparse} from left to right and then repeating the same for the second, third, ... row (exactly as in [Sulatycke and Ghose, 1998]).

- $M_{sparse} \rightarrow index\_col$: an array of column indices, with one index for each value in $M_{sparse} \rightarrow data$. For example, if the first two non-zero elements of M_{sparse} are in row one, columns three and five, then the first two values of $M_{sparse} \rightarrow index\_col$ are 2 and 4 (exactly as in [Sulatycke and Ghose, 1998]).

- $M_{sparse} \rightarrow index\_row$: an array of offsets where each offset points to the first element of a row in $M_{sparse} \rightarrow data$, in row-major format. For example, if the first two rows of M contain three and seven elements, then the first three values of $M_{sparse} \rightarrow index\_row$ are 0, 3 and 10 ($3+7$; new).

- $M_{sparse} \rightarrow length$: an array containing the counts of non-zero data elements of each row, in row-major format. For example, if the first two rows of M_{sparse} contain three and seven elements, the first two values of $M_{sparse} \rightarrow length$ are 3 and 7 (exactly as in [Sulatycke and Ghose, 1998]).

Note that $M_{sparse} \rightarrow data$ and $M_{sparse} \rightarrow index\_col$ have the same length (the number of non-zero elements in M_{sparse}), as do $M_{sparse} \rightarrow length$ and $M_{sparse} \rightarrow index\_row$ (the number of rows in M_{sparse}).

$M_{sparse} \rightarrow index\_col$, $M_{sparse} \rightarrow index\_row$ and $M_{sparse} \rightarrow length$ only need (short) integer data types, however.

We can now perform the sparse-sparse matrix multiplication and the on-the-fly addition of the result to a third dense matrix in the following way.

```
procedure MULTIPLYANDADD($A, B, R$)

    $A \leftarrow$ the first sparse matrix of size $m \times n$
```

```
        $B \leftarrow$ the second sparse matrix of size $n \times m$
        $R \leftarrow$ the dense matrix

        $index_A = 0$
        **for** $i = 0; i < m; i = i + 1$ **do**
            $len_A^{row} = A \rightarrow length[i]$
            **for** $k = 0; k < len_A^{row}; k = k + 1$ **do**
                $col_A = A \rightarrow index\_col[index_A + k]$
                $data_A = A \rightarrow data[index_A + k]$
                $len_B^{row} = B \rightarrow length[col_A]$
                $index_B = B \rightarrow index\_row[col_A]$
                **for** $j = 0; j < len_B^{row}; j = j + 1$ **do**
                    $value_{old} = C[i][B \rightarrow index\_col[index_B + j]]$
                    $value_{toAdd} = data_A \cdot B \rightarrow data[index_B + j]$
                    $C[i][B \rightarrow index\_col[index_B + j]] = value_{old} + value_{toAdd}$
                **end for**
            **end for**
            $index_A = index_A + len_A^{row}$
        **end for**
    **end procedure**
```

4.3.9 Equivalence of Old and New Implementation

Mathematically, both the old and the new implementation produce the same output for the same input. In practice, however, differences could arise due to the imprecision of floating point numbers.

It has to be stressed that even the old implementation will not always produce the exact same results. The result of a double multiplication, for example, depends on the order of CPU instructions produced by the compiler. Even given the same compiler, different architectures and optimization levels will change this order. Another source for differences are downstream programs. The kernel matrix will be used in other algorithms, for example a support vector machine. Even different implementations of the same algorithm (e.g. Sequential Minimal Optimization [Platt, 1998]) will most likely generate different models.

Our changes to the generation of the kernel matrix have not added any additional source for differences. The comparison of cumulative substitution scores to the user-defined threshold is still carried out with 64 bit double precision types. The actual kernel matrix values are stored as integers and hence not subject to precision loss.

The question that remains is whether the additional (preprocessing) steps for predicting queries with our pipeline (Modification 5) has resulted in a loss of precision compared to the old baseline method (Modification 6). To answer this, we have predicted 2,000 random targets from our 20,000 targets test set with models created from the 'Euka (5920)' dataset (multi-class model: 'Nested Dichtomoy for Eukaryota' [Goldberg et al., 2012]; 17 SVMs; $k = 6$; $\sigma = 11$) in two different ways. One time, we used the original baseline implementation of the profile kernel to generate dot products which were then used as input for a Weka model. The latter processed these dot products in 17 precomputed SVMs to output 17 SVM scores for each query, i.e. scaled distances to the hyperplanes. The other time, we calculated these SVM scores with a normal matrix generated by our pipeline (Modification 5; pipeline used the same Weka model) and our new profile kernel implementation.

This resulted in $17 * 2,000 = 34,000$ value pairs that were mathematically identical, but could differ due to floating point imprecision. Next, we determined for each value pair their difference, which we call the 'distance fluctuation'. It corresponds to the difference in the distances to the SVM hyperplane. Then we divided the distance fluctuation by the distance of the supposedly correct distance, i.e. the distance calculated with the original implementation. The result, called 'relative distance fluctuation', can be interpreted as the degree with which the fluctuation influenced the actual decision made by the SVM. A value greater than 1.0 ($> 100\%$ change) means that the fluctuation changed the class of a sample point for this particular SVM (i.e. the side of the hyperplane the sample was on), a value below 0.01 ($< 1\%$ change) indicates that the fluctuation had virtually no impact. Fig. 4.2 shows the distribution of relative distance fluctuations.

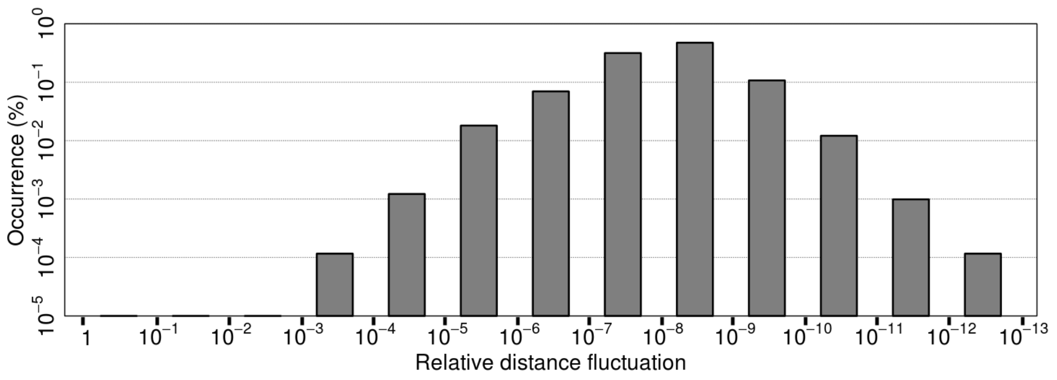


Figure 4.2: Distribution of relative distance fluctuations. This figure shows how much the new kernel implementation changed SVM scores compared to the old implementation. A 'relative distance fluctuation' is the difference between the old and the new score, divided by the old score. The plot is a histogram of 34,000 such fluctuations. Please see the text for how the SVM scores were calculated.

None of the 34,000 relative distance fluctuations was greater than 0.001 and 4 were between 0.001 and 0.0001, which correspond to a frequency of 4/34,000$\approx$0.0001. This means about 1 in 10,000 distances to the hyperplane will be at most 0.1% different from the actual value. Such an effect will not have a measurable impact on a classifier. For example, the 2,000 class probabilities calculated by the Weka multi-class model from the old and the new SVM scores (one value for the 'winning' class in percentage, rounded to three decimal digits) were absolutely identical. (In case of multiple classes, even a dramatic change of a SVM score will most likely not result in a different classification due to the influence of other SVMs on the final class.)

To extrapolate probabilities of more severe changes, note that the occurrences in the outer bins of Fig. 4.2 decrease with about a factor of 10. A relative distance fluctuation between 0.1% and 1% will probably occur in about 1 of every 100,000 SVM decisions, and an actual hyperplane side change ($\geq$100%) in much less than 1 of every 1,000,000 SVM decisions.

4.3.10 Data Sets

Overview. In order to measure the runtime improvement of our new implementation, we use four different data sets for kernel matrix computations and three for classifying new queries. All profiles are taken from a redundancy reduced Swiss-Prot database and readily available as part of the PredictProtein [Rost and Liu, 2003] cache.

The four training data sets correspond to 5920 profiles assigned to 18 classes (set 'Euka (5920)'), 12,500 profiles assigned to 125 classes (set 'SP60_25k'), 25,000 profiles assigned to 250 classes (set 'SP60_25k') and 100,000 profiles assigned to 1000 classes (set 'SP60_100k').

The runtimes for classifying new profiles were measured with models created from these four training data sets. As queries, we used three other data sets containing 1, 200 and 20,000 non-redundant protein profiles. They simulate typical classification tasks, ranging from the frequent single-user single-target case to the prediction of an entire genome.

Detailed Description. Modifications 1 to 3 should significantly accelerate the computation of kernel matrices compared to the original profile kernel implementation. Similarly, Modification 5 is expected to speed up the classification of new query proteins, even if the original implementation is extended by a simple prediction mode (Modification 6). In order to obtain exact measurements, we created four different data sets for training and three for testing.

The first training set ('Euka (5920)') comes from the recent LocTree2 method [Goldberg et al., 2012] which predicts protein localization. It consists of 5920 eukaryotic proteins from the UniProt/Swiss-Prot [Schneider et al., 2009] database, assigned to one of 18 sub-cellular localization classes based on experimental evidence. For the other three, we first redundancy reduced all Swiss-Prot sequences to 60% pairwise sequence identity using cd-hit [Fu et al., 2012] and then sub-sampled 12,500 ('SP60_13k'), 25,000 ('SP60_25k') and 100,000 ('SP60_100k') proteins. While irrelevant for kernel matrix computations, classes of training samples strongly affect prediction speed as they determine the number of SVMs and Support Vectors. We estimated these parameters for the three larger data sets using the data of the final online version of LocTree2 and our own experience. For the LocTree2 data, 18 classes in a 1-vs-all schema (nuclear-vs-all, cytoplasm-vs-all, etc.) sufficed to render 84% of all training profiles Support Vectors. This implied that in order to classify one protein through the baseline method (Modification 6), dot products to 4954 other profiles have to be calculated. Given the 84% value, we assumed all profiles of the larger data sets to be support vectors in order to create realistic, large-scale classification problems. The speed of the normal vector based classification (Modification 4) depends on the number of SVMs. We assumed 125, 250 and 1000 SVMs for the three large-scale profile sets (SP60_13k, SP60_25k, and SP60_100k respectively), corresponding to about 100 proteins per class with a 1-vs-all or 'Nested Dichotomy' [Fox, 1997] based multi-class schema. Note that this underestimates the relative performance of the normal vector based classification: considerably fewer classes should suffice to render almost all training samples Support Vectors and the class sizes in this example were smaller than those for LocTree2. Actual normal vector values were randomly sampled from the LocTree2 vectors, but have no impact on the speed of the application/prediction. For the testing sets, we again used the redundancy reduced Swiss-Prot database and sub-sampled 1, 200 and 20,000 proteins (the one protein was a transpeptidase with 246 amino acids). We did not need class assignments for the testing sets, as we only measured prediction speed, not accuracy. The three sets simulate typical classification tasks, ranging from the frequent single-user single-target case to the prediction of an entire genome.

All profiles of all seven data sets were readily available as part of the Predict-Protein [Rost and Liu, 2003] cache and created by three iterations of PSI-BLAST [Altschul et al., 1997] against UniProt [Consortium, 2011] (E-Value 10^{-3}).

4.4 Results and Discussion

4.4.1 Speed Measurements Under Stringent Conditions

We measured the impact of our modifications on the speed of both, the kernel matrix creation and the final application of the model, i.e. the prediction of new queries. The baseline for kernel matrix computations was the original and publicly available profile kernel implementation from the Leslie lab; for predictions, we implemented the baseline ourselves (Methods: Modification 6). None of our modifications changed the original kernel arithmetically and the chance that floating point imprecisions will lead to different classifications is minimal (Section 4.3.9). Therefore, all previously published values for accuracy are still valid.

Experiments were conducted on a 2 x 6-Core AMD Opteron Processor 2431 (2.4 Ghz) with 32GB DDR2 main memory using various data sets (Section 4.3.10). Each kernel run was executed as the only active process on the entire computer, so that the conditions with respect to memory, disk and hyper-threading were similar for all experiments. Repeating the same measurements 20-30 times revealed a universal runtime standard error below 5%. The profile kernel has two free parameters: the length of the k-mer (k) and the substitution score threshold σ. Parameter combinations were taken from the original publication [Kuang et al., 2004] and LocTree2 [Goldberg et al., 2012]. To our knowledge, only the latter optimized these parameters and found it preferable to use substantially higher substitution score thresholds than reported originally (‘k=5, σ=9’ and ‘k=6, σ=11’). Other papers using the profile kernel appeared to have copied the combinations reported in the original publication.

4.4.2 Kernel Matrix Creation Five Times Faster and Parallelizable

Modifications 1 and 2 (Methods) yielded a constant acceleration, ranging from twice to up to 14 times faster with respect to the original implementation (Fig. 4.3A). On average, the new implementation was about five times faster, with the speed-up increasing proportionally to the data set size. The kernel matrix computation for the SP60_100k data set (Methods) no longer fit into the main memory of our machine (approx. 56GB). Hence, we used our new splitting technique (Methods; Modification 3) to distribute its calculation amongst 100 individual processes that were run simultaneously on a computer cluster (the CPU conditions described in the paragraph above no longer applied for this proof-of-concept run). This took about 40 minutes. The speed of the kernel critically depends on its two parameters (Fig. 4.3). The large difference between, e.g. ‘k=6, σ=9’ and ‘k=6, σ=11’, is due to a loss of sparseness and an accumulation of conserved k-mers during the trie traversal.

However, in our hands, this actually improved performance for the development of LocTree2 [Goldberg et al., 2012], suggesting a relative enhancement of the conserved k-mer signal despite a probable increase of background noise. Indeed, we found the feature vectors resulting from 'k=6, σ=11' to be sparse but less so than those resulting from training with 'k=6, σ=9'.

4.4.3 Predictions Accelerated by Orders of Magnitudes

Besides a general code optimization, our modifications include the feature to calculate the SVM scores for many queries and SVMs in one profile kernel run (model application mode; Methods: Modifications 4 and 5). We compare this variant to the original implementation extended by a support vector based application mode (Methods: Modification 6). The normal vector based variant that we introduced here, is at least five times faster than the support vector based alternative (Fig. 4.3B, Euka data set, 20,000 targets, 'k=5, σ=7.5'), with a maximum acceleration of 205-fold (Fig. 4.3B, SP60_100k, 200 target, 'k=5, σ=7.5'). On average (arithmetic mean over all experiments), our new implementation turned out to be about 66 times faster than the original implementation. Again: for larger data sets, the speed-up would increase. As long as the models are queried only with a few targets (up to about 200), the most limiting factor is the size of the normal vector matrix. For k=5, even the matrix with 1000 SVMs still remains below 10GB (8.2GB), but it grows to 39GB for k=6 and 250 classes and consequently takes about 20 minutes to be read from disk.

4.4.4 Comparison to SVM-Fold and SW-PSSM

Generating the same output as the original version, our new profile kernel implementation can directly be used in existing profile kernel based classifiers like SVM-Fold [Melvin et al., 2007]. The latter is a web-server for the prediction of SCOP classes from protein sequence. Multiple binary SVMs are trained and embedded in a multi-class scheme, called 'adaptive codes', which exploits the hierarchical structure of SCOP. Extending or replacing the Weka-based multi-class models with the adaptive codes approach, our new workflow script (Methods; Modification 5) could generate SVM-Fold automatically. For predictions, SVM-Fold uses the baseline implementation (Methods; Modification 6) with an additional caching of k-mers in the higher levels of the k-mer trie. Prediction speed could be greatly increased by using pre-computed normal matrices (Methods; Modification 4).

A popular competitor of the original profile kernel in terms of classification accuracy is SW-PSSM (Smith-Waterman Position Specific Scoring Matrix; [Rangwala and Karypis, 2005b]). We have compared our implementation of the original profile

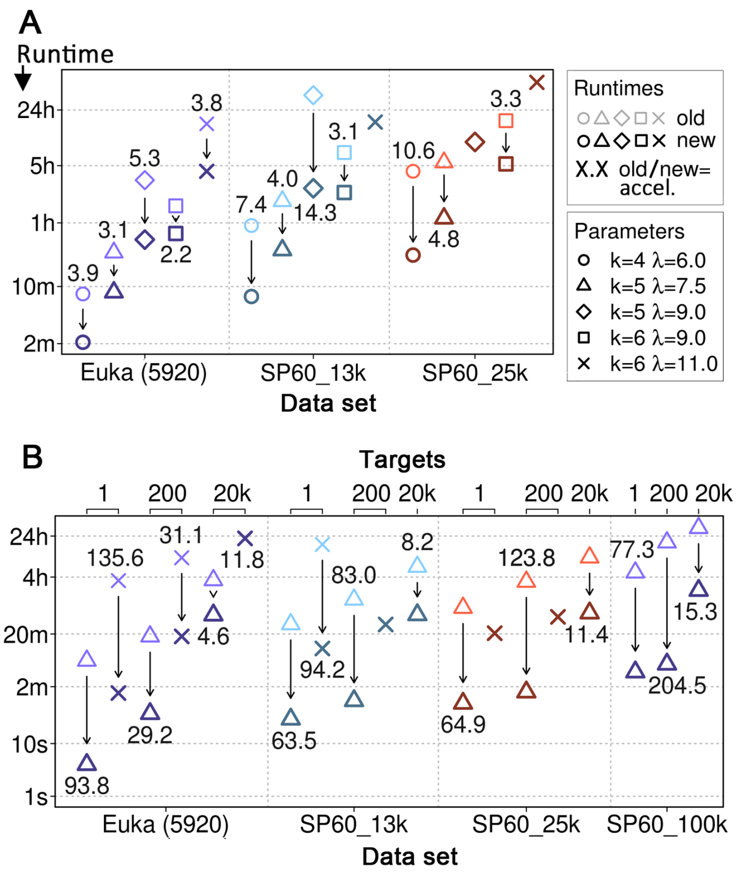


Figure 4.3: Speed measurements. Each arrow compares the runtime of the original implementation (upper symbol) to the new implementation (lower symbol). The symbol type indicates the parameter combination. The number above or below an arrow is the acceleration (original runtime divided by new runtime). All runtimes are wall-clock times of single processes. We did not perform an experiment if it was clear that it would take longer than 24 hours. **(A) Kernel matrix calculations.** In this subfigure we compare kernel matrix creation runtimes. Data sets correspond to subsets of a redundancy reduced Swiss-Prot database with 5920 ('Euka (5920)'), 12,500 ('SP60_13k'), 25,000 ('SP60_25k') and 100,000 ('SP60_100k') samples, respectively. The SP60_100k experiment ('k=5, σ=7.5') for which we used 100 CPUs in parallel took 40 minutes and is not shown. **(B) Prediction of new targets.** This subfigure displays the runtimes for predicting three sets of targets (1, 200 and 20,000 profiles; axis on top) using models created with the training data sets ('Euka (5920)' to 'SP60_100k'; axis on bottom)

kernel to this method and found our program to be multiple orders of magnitudes faster.

SW-PSSM (Smith-Waterman Position Specific Scoring Matrix) [Rangwala and Karypis, 2005b] calculates a profile-profile alignment and converts its score into a valid kernel value. ‘Valid’ means that it makes the score comply with Mercer’s conditions [Burges, 1998], i.e. it turns finding the maximum margin hyperplane into convex optimization problem, implying that it does not have a local minimum. As the feature space is unknown, each value in the kernel matrix has to be calculated separately and only predictions based on the Support Vectors are possible. Using the publicly available implementation of SW-PSSM with the optimal parameter choices taken from elsewhere [Rangwala and Karypis, 2005b] and our own SP60 data set (Section 4.3.10), we found that the average time to compute the score of two profiles is 21 ms. Hence, creating a kernel matrix takes about 4.4 CPU days with 6000 samples, 19 CPU days with 12,500 samples and 3.3 CPU years with 100,000 samples. Classifying a single target with the same number of support vectors takes about 2, 4, and 35 minutes, respectively. Comparing this to the runtime for our new implementation with the parameter combination ‘k=5, σ=7.5’, our new implementation is roughly 744, 980 and 433 times faster than SW-PSSM for the three training data sets and 30, 8 and 9 times faster when classifying a single target. Additionally, in contrast to the profile kernel, simultaneously applying the model to multiple queries gains no speed-up. The runtime, therefore, is directly proportional to the number of queries and the superiority for predictions easily exceeds that of matrix creations for many queries. In summary, our implementation of the original profile kernel hugely outperforms SW-PSSM.

4.5 Conclusion

The original profile kernel proposed by the Leslie group is highly accurate and can be applied to many classification problems. Our new implementation produces the identical results with considerably fewer computer resources (in terms of runtime and memory). Additionally, we have implemented a prediction workflow. It can both automatically create new models and apply them to new queries.

Chapter 5

Improving Sequence-based Binary PPI Prediction

5.1 Outline

As mentioned in the previous chapter, it eventually became apparent that our tool to predict the interaction of residue-residue pairs in a PPI (Chapter 3) was not accurate enough to compete with state-of-the art classifiers for the binary prediction of PPIs (i.e. whether two proteins interact or not). In this chapter, we follow a different approach. We apply our accelerated profile kernel (Chapter 4) to the binary prediction of PPIs in human. This is a new sequence-only based approach that improves over state-of-the-art methods. Using highly reliable human PPIs, we show how evolutionary profiles and subcellular localization increase precision even for low recall levels. A new rigorous way to reduce protein-protein interaction redundancy reveals that only a fraction of available PPIs is needed to build more accurate classifiers. Two cross-validations differing in the similarity amongst non-interacting protein pairs investigate their impact on PPI prediction. We conclude by predicting all 200 million protein pairs in human and estimating their accuracy in terms of recall and precision.

5.2 Introduction

PPIs: physical protein-protein Interactions between different proteins. We define as PPI (Protein-Protein Interaction) only the interaction between two proteins A-B if the following three conditions are fulfilled (1) external interaction: A and B are different proteins. (2) physical interaction: the interaction is manifested in that residues from A are in physical contact with residues from B (e.g. closest atoms within 6Å, i.e. 0.65 nm). Note that this ‘molecular’ or ‘structural biology’ view of PPIs differs substantially from what most publications in PubMed referring to PPIs would label as PPIs: assume that P1 activates P2 activates P3, then most publications would also consider P1-P3 to interact, while we would only consider P1-P2, P2-P3. Similarly, most authors referring to ‘physical’ interactions tend to include all components of large complexes to physically interact, e.g. all constituents of a ribosome. Tandem Purification Pull-downs (TAP) measures exactly such physical complexes. Again: for us the crucial aspect of an interaction pair is the direct physical contact of its constituents.

Experimental PPIs have come a long way. Better understanding physical, external protein-protein interactions is crucial for grasping cellular processes on the molecular level [Ofran and Rost, 2003b, Hamp and Rost, 2012]. Substantial investments in large-scale experimental studies, such as yeast-2-hybrid (y2h) [Fields and Song, 1989, Uetz et al., 2000, Ito et al., 2001, Uetz et al., 2006], mass-spectrometry [Meissner and Mann, 2014], or - just beginning - next-generation sequencing techniques [Yu et al., 2011], has created a wealth of information. Despite the Herculean efforts, our view remains importantly incomplete even for the most-worked on model organisms. Ultimately, the problem is as simple as this: we cannot apply all experimental methods to all model proteins in any organism. To begin with the highest-resolution experiments, namely the determination of PPIs through X-ray crystallography: we have high-resolution three-dimensional (3D) structures for fewer than half of all human proteins [Rose et al., 2010, Haas et al., 2013], and we have at least one single 3D structures giving the details of interaction for fewer than 20% of all human proteins (Chapter 2). Even this high-resolution information comes with many challenges: many of the 3D complexes do not correspond to cellular interactions [Krissinel and Henrick, 2005, Hamp and Rost, 2012], and even those that are often are observed to interact very differently [Hamp and Rost, 2012]. Clearly, the issue of in vitro vs. in vivo (what interacts in the tube may not interact in the cell and vice versa), cannot be addressed by high-resolution structures alone. Leaving the realm of high-resolution data, we immediately widen our coverage, e.g. of human, but we also bring in information that is much less reliable. All large-scale experiments setting out to capture ‘all PPIs’ in some set of proteins representing an organism continue to suffer from a number of shortcomings. They often detect protein-protein interactions (PPIs) by mistake (false positives) and miss interactions that actually happen (false negatives). In addition, errors often depend on factors related to subcellular localization or particular features of the proteins that cannot be penciled into the error assessment. Furthermore, errors may originate from how exactly an experimental protocol is executed: the y2h scan from one lab may be all faulty while the ‘same’ y2h technique from another may be mostly right [Huang et al., 2007, Stynen et al., 2012]. These issues become most evident in estimates of the total number of interactions in an organism [Stumpf et al., 2008, Venkatesan et al., 2009]. Even for yeast, THE most studied model organisms for y2h scans, the number of expected interactions has just been almost doubled after a decade of intense work [Sambourg and Thierry-Mieg, 2010]. For human, known interactions accumulate fast, but still only constitute a small fraction of all expected PPIs [Venkatesan et al., 2009].

Predictions and experimental evidence are intertwined. To increase interactome coverage and help finding interactions that elude high-throughput approaches, a number of computational methods have been developed (excellent reviews in [Liu et al., 2008b, Lees et al., 2011, Mosca et al., 2013]). They use diverse experimental data sources to build statistical models, typically one for each target organism, and produce lists of scored protein pairs, with higher scores indicating higher likelihood of interaction. Data sources used include protein sequences, structures, co-evolution, co-expression, domain co-occurrence, text-mining, subcellular localization and already known interactions (network topology). Model types range from Naive Bayes, to Support Vector Machines and Conditional Random Fields and are typically combined with other tools such as homology models or a-priori knowledge, e.g. biophysical features of amino acids.

Similar to wet-lab methods, each computational predictor has its own speed and error characteristics. Using different types of experimental data can increase their accuracy, but missing annotations reduce their applicability (as is the case for SCOP [Murzin et al., 1995], for example). A recent assessment of function prediction tools also revealed that homology is still more informative than other types of high-throughput data [Hamp et al., 2013b, Radivojac et al., 2013]. Hence, great important lies on developing fast and accurate sequence-only based methods.

A recent review of PPI prediction methods [Park and Marcotte, 2012] found that the accuracy of correctly identifying an interaction between proteins A and B significantly depends on whether or not interactions of A and B with other partners have been used for development. Best are cases in which A and B were used, second best if this was case for either A or B, and all methods tested performed dismally if both A and B had not been seen before. The latter two classes of targets are by far the most frequent as reliable interactions usually cover only a fraction of the complete proteome (e.g. still less than half for human [Schaefer et al., 2012]). This finding invalidates most of the performance measurements published previously as they are typically based on cross-validations with data sets only reduced on the level of pairwise sequence similarity. Randomly splitting such data sets will largely create cases of the first/easiest of the three classes above.

Here, we introduce a new method to predict that two different proteins A and B interact physically, i.e. that they have residues touching each other without predicting the binding site. The method exclusively uses sequence-derived features that are available for all proteins of known sequence. Our method uses a combination of empirical rules along with several machine learning-based protocols. When we apply the formalism to capturing all PPIs in human, we present data that suggests the addition of important novelty. Assume the example above, i.e. we want to predict the interaction between A-B. If A and B are in our data set, our method slightly outperforms other approaches. If either A or B is new, however, it improves greatly.

5.3 Methods

5.3.1 Data Set ParkMarcotte

Park and Marcotte [2012] used PPIs from the Protein Interaction Network Analysis Platform [Wu et al., 2009] (version 3/2010), a collection of PPIs from various databases, and redundancy reduced it with cd-hit [Fu et al., 2012] so that no protein had more than 40% pairwise sequence identity to any other protein in the data set. For their analysis, they divided this 'ParkMarcotte' set into 10 partitions and, in a cross-validation manner, used 9 partitions for training and one for testing. Each interaction between proteins A and B in a test partition was further assigned to one of three subsets, corresponding to different classes of difficulty: set 1 (C1) if both A and B had other interactions in the corresponding training set, set 2 (C2) if this was the case for either A or B and set 3 (C3) if neither of the two was in the training set (illustration in Fig. 5.1A). Non-interacting ('negative') pairs were generated by randomly re-wiring the proteins in each of the four sets (1 training + 3 testing) separately.

5.3.2 Data Set Hippie

The Hippie database [Schaefer et al., 2012] collects PPIs from human with experimental annotations. Each interaction is graded by a reliability score that considers information such as the number of publications per interaction or the type of experimental support. We followed the Hippie procedure [Schaefer et al., 2012] and reduced version 1.2 (Aug 2011) to the top 10% highest scoring interactions to obtain a high-quality subset, dubbed 'HippieHQ1.2' with 7,237 human PPIs from 3,915 different proteins representing 43% of the human Pfam families. We redundancy reduced HippieHQ1.2 by excluding sequence similar interactions as follows. When we included an interaction A-B in the non-redundant set, we excluded all interactions A'-C and B'-D. A is similar to A' if HVAL(A,A')>20 ([Rost, 1999]; corresponding to 40% pairwise sequence identity for 250 aligned residues). The same holds for B and B'. In other words: A was sequentially dissimilar to any other protein in the data set - only its interaction partner B was a possible exception (5% of all cases). This level of non-redundancy is similar to the C3 class defined by Park and Marcotte [2012]. We will therefore refer to the set as 'HippieHQ1.2_C3' (842 PPIs; 30% of human Pfam families). Topological differences between HippieHQ1.2_C3 and ParkMarcotte are illustrated in Fig. 5.1. Most importantly, a protein can have many interaction partners in ParkMarcotte, but only one in HippieHQ1.2_C3.

We sampled the 'negatives' (A-B do not interact) in two different ways (below), but we always ascertained that no negative was listed as positive in the full Hippie

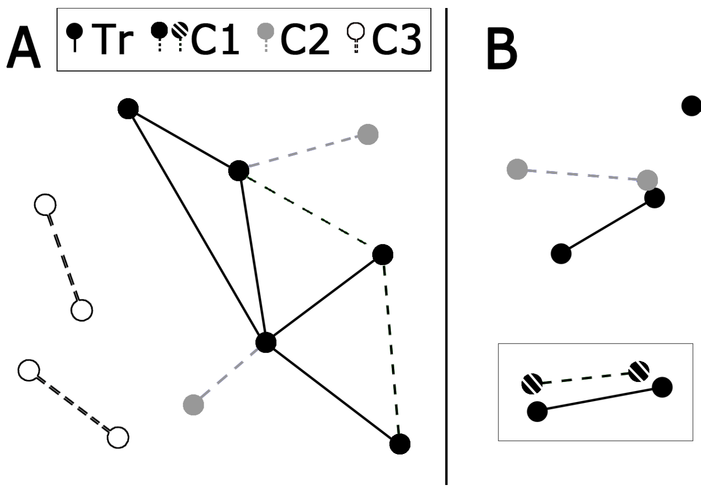


Figure 5.1: Data sets in terms of interactions and sequence similarity. Every node is a protein and every edge is a PPI. Close distance between two proteins indicates high sequence similarity. Proteins that are part of the training set are black filled nodes. All other nodes are proteins only occurring in the test set. Training interactions ('Tr') are solid lines, test interactions are dashed lines ('C1-3'). **(A)** Park et al. redundancy reduced their data set to max. 40% pairwise sequence identity. Consequently, all nodes are far apart. They split the set into 10 folds to perform a cross-validation and the figure displays one such fold. Training proteins may have many interaction partners and form networks. Each test interaction between proteins A and B is grouped into one of three classes (C1-3) based on whether A and/or B are already in the training set (main text for details). **(B)** In our cross-validations, two interaction partners are allowed to be sequentially similar, but there is no sequence similarity between interactions. Hence training and test interactions all fall into the C3 category amongst each other. The other two classes C1 and C2 only occurred in the context of the full interactome prediction. They are defined so that they also account for sequence similarity between test and training interactions. This is necessary because Park et al. assumes non-redundancy between all proteins, what is clearly not the case for full interactome prediction. For example, assume there is a test interaction A-B and a training interaction A'-B' and pairs A-A' and B-B' both have 99% sequence identity (in (B) there is a box around such a case). This would still be a C3 case according to [Park and Marcotte, 2012] but a C1 case in our definition.

1.2 database (i.e. as a PPI with experimental evidence that we deemed insufficient for training). Hippie had already mapped protein identifiers to a reference nomenclature which perfectly matched the EBI human reference proteome [Dessimoz et al., 2012, Gabaldon et al., 2009] with 20,249 proteins. The latter was our source for protein identifiers and sequences for every Hippie related experiment.

5.3.3 Cross-validation Hippie

We performed two different cross-validations with HippieHQ1.2_C3 that differed in the way of sampling the negatives (non-interactions).

First cross-validation. We split HippieHQ1.2_C3 into 20 equally sized partitions $p_1, \ldots, p_{20}$. Next, we merged the partitions in the following way to create 10 unique classification tasks. The positives (observed PPIs) for the first of 10 test partitions ('test1') were those in p_1 and p_2. Randomly re-pairing the proteins involved in any of the PPIs from p_1 and p_2 created the negatives (non-interactions) of test1. While re-pairing, we ascertained that no negative A-B had another protein C in test1 with HVAL(A,C)>20 or HVAL(B,C)>20 (this might happen because in our redundancy reduction we had not filtered out sequence similarity between interacting pairs of proteins, i.e. HippieHQ1.2_C3 contained samples of PPIs A-B with HVAL(A,B)>20.

For the positive interactions in the first of 10 training sets ('training1'), we merged $p_3, \ldots, p_{11}$. For the negative interactions of training1, we merged the interactions in $p_{12}, \ldots, p_{20}$ and then randomly re-wired them exactly as for test1. The sets resulting from this procedure implied that (i) no test protein had HVAL>20 to any training protein, (ii) all training interactions, irrespectively of negative or positive, were pairwise dissimilar and (iii) all negative test interactions were pairwise dissimilar and all positive test interaction. To improve mixture, we also randomized the assignment of partitions to either the negatives or positives in the training set. Then we repeated the entire procedure ten times, creating 100 train-test setups. This ensured statistically significant precision estimates even for low recall levels (c.f. 'Evaluation measures').

Second cross-validation. The second cross-validation split followed the more or less trivial standard many developers are familiar with: we randomly split the entire set of positives into ten partitions and used nine for training and one for testing. We repeated this ten times until each PPI had become part of a test set exactly once. Negatives were randomly sampled from the full human proteome until there were 10 times more negatives than positives in each partition. This means, unlike for the first cross-validation, negative test interactions can now fall into the C1 and C2 category. Again, we repeated all this 10 times, ending up with 100 unique train-test setups.

5.3.4 Prediction of All PPIs in Human

We downloaded the human reference proteome from EBI [Dessimoz et al., 2012] and predicted all possible interactions between proteins longer than 50 and shorter than 5,000 residues. Our positive training set for this task was the full HippieHQ1.2_C3 (reasons for not using all of HippieHQ1.2 given in Results). Negatives for the training were sampled randomly from the human proteome until there were 100 times more negatives than positives. Additionally, no negative was sampled twice or already reported in the full Hippie 1.2 data set. Positive test interactions were taken from the 10% most reliable interactions in Hippie version 1.6 (Oct 2013: set 'HippieHQ1.6'), excluding those of HippieHQ1.2. Negative test interactions were all protein pairs not in HippieHQ1.6. We distinguished between three types of difficulty for each test interaction A-B: Class 1 ('C1') if both A or B had HVAL>20 to a protein in HippieHQ1.2_C3 (3,425 positives; 12,577,484 pairs total); 'C2' if this was the case for only one interactor (2,924 positive pairs; 76,273,451 pairs total) and 'C3' if neither protein was sequentially similar to any protein in HippieHQ1.2_C3 (836 positive pairs; 115,558,003 pairs total). This definition of the C1-3 classes accounts for sequence similarity between proteins, a feature that is missing in the definitions by Park et al., but crucial for full interactome predictions (illustration in Fig 5.1B). We refer to these subsets containing both negative and positive test pairs as HippieHQ1.6_(C1,C2,C3).

5.3.5 Evaluation Measures

Overview. All methods under consideration associate a pair of proteins with a score reflecting the likelihood of a physical interaction. We can therefore apply standard two-class evaluation measures. In particular, we used the area under the ROC curve (AUROC) and recall-precision curves. For the full interactome predictions, we additionally re-estimated precisions to correct for HippieHQ1.6 constituting only a fraction of all positive interactions (much in contrast to the negatives). For this task, we used estimates of the ratio of positive to negative interactions in human [Hart et al., 2006, Stumpf et al., 2008, Venkatesan et al., 2009]. Despite large error margins, we feel that such a correction is still better than leaving it to the user to calculate actual precisions him/herself, e.g. from ROC curves (which are not as susceptible), or to not report it all.

Estimating precision of full interactome predictions. Every point in the $recall-precision$ curve corresponds to a threshold that separates predictions with higher scores from those with lower scores. Pairs above the threshold can either be actual PPIs (true positives [TPs]) or non-interacting pairs (false positives [FPs]). Analogously, pairs below the threshold can be non-interacting (true negatives [TNs]) or interacting (false negatives [FNs]).

Precision is the number of actual PPIs above the threshold (TPs [true positives]) divided by the number of all predictions above the threshold (TPs + FPs [false positives]). Recall is the fraction of actual PPIs above the threshold (TPs) among all actual interactions (TPs + FNs).

In full interactome predictions, we have a small sample of actual PPIs and treat all other protein pairs as negatives. Any precision calculated in such a setting will be an underestimate because many FPs are actually TPs. We can, however, estimate the actual precision.

We interpret the observed precision as a conditional probability: $precision_{obs} = \frac{TPs}{TPs+FPs} = P(I|A)$, where the event I indicates that a pair of proteins is an actual PPI and A that the pair's score is above the threshold. Following Bayes' theorem, we can rewrite this as $P(I|A) = \frac{P(A|I)*P(I)}{P(A)}$. Here, $P(A|I)$ is the fraction of positive PPIs above the threshold among all positive PPIs, in other words $recall_{obs}$. Hence $P(A|I) = \frac{TPs}{TPs+FNs}$. $P(A) = \frac{TPs+FPs}{TPs+FPs+TNs+FNs}$ is the fraction of pairs above the threshold among all pairs and $P(I) = \frac{TPs+FNs}{TPs+FPs+TNs+FNs}$ is the fraction of positive PPIs among all pairs.

Recall is calculated solely from actual PPIs and therefore independent of the ratio of interacting to non-interacting pairs. Hence, $P(A|I)$ remains the same even for the corrected precision. The same holds for $P(A)$ as we will not add or remove any interactions (we have already predicted every possible pair) or change the threshold. The only value we need to adjust is $P(I)$. Our estimate for $P(I)$ has been the number of test PPIs (833 in HippieHQ1.6_C3) divided by all possible pairs. It should, however, be the number of all PPIs in human (e.g 225,000 as estimated by Hart et al. [2006]) divided by all possible pairs ($P(I)'$). Performing this correction and using $P(I)'$ instead of $P(I)$, we obtain the estimated actual precision $precision'$.

5.3.6 Profile Interaction Kernel

5.3.6.1 Overview

Our method relies on 'Support Vector Machines' (SVMs) (introduction e.g. in [Schölkopf and Smola, 2002]), which calculate a hyperplane that optimally separates data points of one class from those of another class. A user-given 'kernel function' defines the actual space of the data points and calculates all of their pairwise dot products, the 'kernel matrix'. As a specialty, only this matrix is needed to calculate the hyperplane, not the explicit feature vectors of the data points. Hence, our task was to define a good feature space for protein-protein interactions and to find a fast way to calculate dot products in this space. Our solution is built upon the evolutionary profile based kernel developed by the Leslie group [Kuang et al., 2004]. It is defined on single proteins, but can be extended to pairs of proteins (PPIs; Section 5.3.6.2).

The single-protein profile kernel represents a protein as a very high dimensional feature vector in which each dimension stands for a k amino acids long sequence (k-mer). For example, if k=4, the feature vector has $20^4 = 160,000$ dimensions. The value of a dimension is the number of times this k-mer is conserved in the protein profile, i.e. how often the sum of amino acid substitution scores is below a user-defined threshold σ. Substitution scores are read off an evolutionary profile calculated by PSI-BLAST [Altschul et al., 1997]. For example, if we wanted to check whether 4-mer 'WTGG' is conserved at position 37 in the profile, we first read off the frequency of 'W' at that position and convert it to a score by taking the negative logarithm. Then we do the same for 'T' at position 38 and 'G' at positions 39 and 40, sum up the four scores and check whether the result is smaller than σ. (Note that there can be more than one conserved k-mer per position: e.g. with $\sigma = \infty$ there are 20^k conserved k-mers for every position in a sequence of length n and hence $n(20^k)$ in total). The profile kernel is then defined as the dot product of two feature vectors. In the actual implementation of the profile kernel, an efficient k-mer trie based algorithm takes all PSI-BLAST profiles as input and calculates the entire kernel matrix in one traversal of the trie. In the previous Chapter, we further accelerate this algorithm by various technical and methodological modifications and make it easier to use.

In the feature space that we define for protein-protein interactions, each dimension represents a pair of k-mers. For example, besides 'WTGG' in protein A, we now also look for conserved k-mers 'LGAH' in protein B and count how often 'WTGG' and 'LGAH' occur together in the interaction. This new feature space has $20^k * 20^k$ dimensions, but it in the next Section, we show that the dot product between two feature vectors only requires dot products in the 20^k dimensional single-protein feature space.

5.3.6.2 Profile Interaction Kernel for PPI Classification

Let P_A, $P_{A'}$, P_B,$P_{B'}$ be four proteins and let $A, A', B, B' \in \mathbb{N}^{20^k}$ be the protein feature vectors as defined by the original profile kernel (k is the k-mer length). Every dimension A_i is the number of times k-mer i has been conserved in the sequence profile of P_A (analogously A'_i, B_i, B'_i). For example, 4-mer 'AAAA' ($i = 1, k = 4$) may have been conserved 12 times in P_A, i.e. $A_1 = 12$.

(P_A, P_B) and $(P_{A'}, P_{B'})$ are the two interactions for which a kernel value has to be calculated. In the feature space of the profile interaction kernel, every dimension stands for one pair of k-mers and its value for the number of times this pair has been conserved in the interaction. For example, in interaction (P_A, P_B) we may find

'AAAA' conserved at position 8 in the profile of P_A and 'WWTG' conserved at position 150 in the profile of P_B. Hence, we increase the dimension 'AAAA-WWTG' in the feature vector for (P_A, P_B) by one. Knowing how to calculate the feature vector for a PPI, the profile interaction kernel is simply the dot product of two such vectors. This corresponds to the number times the two PPIs have the same k-mer pairs.

The protein order matters. In other words, the dot product between (P_A, P_B) and $(P_{A'}, P_{B'})$ is different to the dot product between (P_A, P_B) and $(P_{B'}, P_{A'})$. We can resolve this by calculating both and taking the maximum (corresponding to finding the better match between the single proteins). Alternatively, we can add the occurrences of one dimension to the dimension of the reversed k-mer pair (e.g. adding the counts of 'AAAA-WWTG' to those of 'WWTG-AAAA'). In preliminary studies, the latter solution was consistenly better than the former so that we chose the latter in all experiments presented here.

More formally, the feature vector of (P_A, P_B) is defined as

$$\begin{aligned} X &\mapsto \mathbb{N}^{20^k 20^k} \\ \phi((P_A, P_B)) &= (\ldots, A_i B_j + A_j B_i, \ldots) \end{aligned} \tag{5.1}$$

where $1 \leq i, j \leq 20^k$ and X is the space of all protein pairs. The profile interaction kernel consequently is

$$\begin{aligned} X \times X &\mapsto \mathbb{N} \\ K_{ppi}(\phi((P_A, P_B)), \phi((P_{A'}, P_{B'}))) &= \sum_{i,j} (A_i B_j + A_j B_i)(A'_i B'_j + A'_j B'_i) = \\ = \sum_{i,j} A_i A'_i B_j B'_j + \sum_{i,j} A_i A'_j B'_i B_j &+ \sum_{i,j} A_j A'_i B_i B'_j + \sum_{i,j} A_j A'_j B_i B'_i = \\ = 2\sum_{i,j} A_i A'_i B_j B'_j + 2\sum_{i,j} A_i A'_j B'_i B_j &= 2\sum_i (A_i A'_i \sum_j B_j B'_j) + \\ +2\sum_i (A_i B'_i \sum_j A'_j B_j) = 2\sum_j B_j B'_j \sum_i A_i A'_i &+ 2\sum_j A'_j B_j \sum_i A_i B'_i = \\ 2\hat{K}(B, B')\hat{K}(A, A') &+ 2\hat{K}(A', B)\hat{K}(A, B') \\ &= \text{(due to scale invariance)} \\ \hat{K}(B, B')\hat{K}(A, A') &+ \hat{K}(A', B)\hat{K}(A, B') \end{aligned} \tag{5.2}$$

where $\hat{K}$ is the original profile kernel. The same mathematical formulation has also been found in [Martin et al., 2005].

5.3.6.3 Incorporating Subcellular Localization Prediction

Finally, knowing the subcellular localization of a protein, we can check whether a PPI is actually possible from a compartmental point of view. Again using the profile kernel and multi-class machine learning techniques explored in [Hamp et al., 2011] and an additional PSI-BLAST pre-filter, we recently developed LocTree3 (unpublished) which assigns a eukaryotic protein to one of 18 localization classes. (Loctree3 uses Loctree2 [Goldberg et al., 2012] for difficult prediction cases and Blast for easy cases. Difficulty is measured by the sequence identity of the best annotated Blast hit.) The state-of-the-art accuracy of Loctree3 might improve PPI prediction by filtering out all interactions that are predicted to happen between proteins from different compartments. For both LocTree3 and the profile interaction kernel, we use the PSI-BLAST profiles generated and cached by our PredictProtein [Rost and Liu, 2003] server (3 iterations against Uniprot [Consortium, 2011] reduced to 80% pairwise sequence identity; E-value 0.001).

5.3.6.4 Optimization of Free Parameters

The k-mer length, the substitution score threshold and the SVM complexity parameter C have to be optimized empirically. Preliminary studies with various different interaction sets, e.g. from the PDB [Berman et al., 2000] and various organisms, constantly showed that a k-mer length of 5 is the best choice (in particular better than 3, as chosen by a number of other methods). For lambda, we propose three different values from previous applications of the profile kernel: 7, 9 and 11. As SVM, we used the Sequential Minimal Optimization [Platt, 1998] implementation in Weka [Frank et al., 2004], which also provided the functionality to optimize C by an internal 10-fold cross-validation for every SVM that we trained (values for C were $10^{-2,-1,\dots,2}$). We always chose the parameter combination that led to the highest average precision up to a recall of 50% (precision step size of 10%). We favored this approach over alternatives such as AUROC because we knew a-priori that performance for higher recall levels would be unacceptable. It is a compromise between a reduced sample size (due to a dominance of low recall levels) and a focus on high precision.

5.3.7 Other Methods

Park et al. recently compared sequence-based PPI prediction methods [Park and Marcotte, 2012]. We test our method on the same data set and additionally compare it against the following top 2 methods for C3 targets using our own data sets. Implementations that allow complete re-training with custom interaction data were thankfully provided by Park, Y..

5.3.7.1 PIPE2

For a target interaction between proteins A and B, PIPE2 (Protein-protein Interaction Prediction Engine 2; Pitre et al. [2008]) counts how often two particular 20-mers from A and B co-occur in other interactions in the training set. The result is stored in a matrix with all 20-mers of A as the X and all 20-mers of B as the Y axis. The value of a cell is the number of positive interactions this pair of 20-mers has been observed in. The matching of 20-mers is inexact, i.e. not exactly the same 20-mer has to be found in a training protein A' (B') to be considered a hit for A (B). A PAM120 alignment score above a certain threshold suffices. The matrix is smoothed by a sliding 3x3 window that replaces the central value by the median of the window. Two proteins are predicted to interact when the average score in a 3x3 window of the new matrix is above a certain threshold.

5.3.7.2 Sigprod

Sigprod [Martin et al., 2005] represent a protein sequence as a vector in which each dimension stands for a 'signature', i.e. a pair of 3-mers. Each 3-mer in a pair has the same central amino acid and the same order-independent flanking amino acids. For example, dimension 'GTW' represents both 'GTW' and 'WTG'. Each value in a vector is the number of times this signature has been observed in the sequence. A PPI is then represented as a vector of signature co-occurrences, as explained for our method before. Our approach can hence be seen as an enhancement of the signature PPI kernel by using protein profiles instead of sequences, a much larger single-protein feature space ($\sim$4,000 for signatures vs. 3.2M for the profile kernel with k=5), a thorough selection of the SVM complexity parameter (not performed by Martin et al.) and a state-of-the-art localization predictor as a filter.

5.4 Results

5.4.1 Overview

We have developed a new approach to predicting PPIs from sequence alone. It is based on evolutionary profiles, a fast and high dimensional profile-based SVM kernel, rigorous optimization of free parameters and a filter for protein pairs predicted to localize in non-neighboring compartments (Methods). We compare it against the best two other sequence-based remote interolog detectors according to Park and Marcotte [2012] by means of three different cross-validations on human PPIs (Methods). Each sheds light on the prediction of remotely interologous interactions from a different angle. Finally, we predict all possible interactions between human proteins while differentiation between different classes of prediction difficulty. Accuracy is measured by highly reliable experimental interactions that have been recently added to PPI databases.

5.4.2 Cross-validation on ParkMarcotte Data Set

The PPI data set created by [Park and Marcotte, 2012] is non-redundant on the protein level and includes for each protein all its interactions reported before March 2010 (Methods). Negatives are sampled randomly from the non-redundant proteins. The authors perform a cross-validation with 7 different PPI prediction methods that are based on sequence alone. Each test interaction between proteins A and B is assigned to one of three categories: 'C1' if both A and B have other interactions in the training set (say A-C and B-D), 'C2' if this is the case for only one interactor (say A-C) and 'C3' if neither of the two has a similar protein in the training set ('C3'). As a result, they found that the accuracy of all 7 methods significantly depends on the category and that there are clear differences among the methods. We trained and tested our method on the same data set. Results are listed in Table 5.1, together with those of the two best other performers in the C3 category, namely Sigprod [Martin et al., 2005] and PIPE2 [Pitre et al., 2008] (Methods). We did not apply the LocTree3 filter (Methods) in order not to lose any predictions given the full area under ROC curve is the evaluation measure.

5.4.3 Cross-validation on Hippie Data Sets

We complement the study by Park and Marcotte [2012] with two additional 10-fold cross-validations to address some weaknesses in the ParkMarcotte data and show that our method really constitutes advancement in remote interolog detection (Methods). For both, we take the top 10% most reliable human interactions according

Method	CV	C1	C2	C3
Sigprod	81 ± 1	81 ± 1	61 ± 1	58 ± 3
PIPE2	76 ± 1	77 ± 1	64 ± 1	59 ± 2
PK	87 ± 1	87 ± 1	69 ± 1	66 ± 2

Table 5.1: Cross-validation on the ParkMarcotte data set. This table shows the area under the ROC curves of the three best sequence-based remote interolog detectors, calculated with the ParkMarcotte data sets in a 4 times 10-fold cross validation [Park and Marcotte, 2012]. 'CV' is the performance on all test cases, 'C1' on those for which both interactors already had other interactions in the training set, 'C2' on those for which this was true for only one protein and 'C3' on the interactions that were sequentially dissimilar to the training proteins. Errors are standard deviations. Values for Sigprod [Martin et al., 2005] and PIPE2 [Pitre et al., 2008] were taken from [Park and Marcotte, 2012], PK (profile kernel) is our new method without the LocTree3 filter (Methods).

to the Hippie database [Schaefer et al., 2012]. The first cross-validation tests the detection of pure interaction patterns (as opposed to 'hooking' to a protein and then only detecting a compatible partner as for C1,2 cases) by constructing training and test data in a way that any interaction, negative or positive, is at best a remote interolog of any other interaction. This compares methods in their ability to detect interactions solely by interaction signals. Negative interactions are sampled from the same proteins as the positive interactions and every protein occurs only once in a cross-validation fold (Methods). With this setup, we again compare our method to the best two other remote interolog detectors according to Park et al.. Results are presented as recall-precision curves in Fig. 5.2A.

In the second cross-validation, positive samples are the same as in the first, i.e. every positive interaction is a C3 remote interolog to any other positive interaction. The negatives, however, are sampled randomly from the human proteome until there are ten times more negative interactions than positive interactions for every training-/test set (Methods). Fig. 5.2B shows the results.

It may be argued that the rigorous redundancy reduction of the Hippie data sets has excluded so many interactions that accuracies are unnecessarily low. We tested this by re-adding all previously excluded high quality interactions among the positive training proteins of the second cross-validation, thus creating a level of non-redundancy very similar to the ParkMarcotte data set. Then we repeated the entire second cross-validation. In the same way we also tested the effect of adding all interactions (not only the ones among the positive training proteins) as long as they did not violate the C3 rules between train and test set. Surprisingly, the precision of all methods dropped significantly, especially for lower levels of recall.

In the 882 HippieHQ1.2_C3 data set, every interactor of an interaction between proteins A and B was sequentially dissimilar ($HVAL < 20$) to all proteins of all other interactions in the data set. We performed a 10-fold cross-validation with this data set. Negatives were sampled randomly from the human proteome up to a positive to negative ratio of 1:10 in every data set split. The entire procedure was repeated 10 times, creating 100 train/test setups.

Keeping the negative sets the same, we compared the performances resulting from this cross-validation to two other types of positive sampling. First, we kept the proteins of the positive interactions in a training set the same, but added all interactions between them that were previously excluded. All new interactions came from HippieHQ1.2. This increased the number of positive interactions in a training set from $\sim$760 to $\sim$2,050 and created a level on non-redundancy very similar to the one used by Park and Marcotte [2012]. The negatives were unchanged. Next, we also added interactions to proteins that had so far not been in the positive training set (source again HippieHQ1.2), while making sure no new protein was sequentially similar to a protein in the test set. This increased the number of positives per training set to $\sim$4,750. Again, negatives were not changed.

Results are shown in Fig. 5.3 for the same methods as in Fig. 5.2. The larger training sets consistenly and significantly decreased prediction performance on the test sets for lower recall levels in all methods.

5.4.4 Full Interactome Prediction

The primary incentive to develop our method was to identify interactions between proteins for which no interaction has been reported so far. Following the cross-validated result that a highly non-redundant training set leads to highest precision, we re-trained our method on the full HippieHQ1.2_C3 data set (negative-positive ratio reduced to 1:100; Methods) and predicted all 20,000*20,000=20 million possible interactions in human. The accuracy of this experiment was measured by high-quality positive interactions that had been added to Hippie in the two years after the release of version 1.2 (HippieHQ1.6; Methods). Negatives were all other pairs of proteins except for HippieHQ1.2_C3. We again differentiated between three classes of difficulty, namely ‘C1’ if both interactors were sequentially similar to proteins in Hippie1.2_C3, ‘C2’ if this was the case for only one protein and ‘C3’ otherwise (Methods). Results are presented in Fig. 5.4A.

The values for precision in Fig. 5.4A are strong underestimates because they are susceptible to the ratio of positive to negative interactions. (The test set comprises all non-interacting, but only a fraction of the interacting pairs.) Assuming HippieHQ1.6

is a random sample from the full positive interactome, we can correct the values for precision (Methods), but we need to know the actual positive to negative ratio. The number of direct interactions in human has been estimated numerous times, but with very different outcomes [Hart et al., 2006, Stumpf et al., 2008, Venkatesan et al., 2009]. It ranges from 130,000 [Venkatesan et al., 2009] to 650,000 [Stumpf et al., 2008] (225,000 in [Hart et al., 2006]). We therefore decided to perform two corrections of precision, one for what we believe to be the lower boundary (100,000) and one for an upper boundary (300,000). Results are shown in Fig. 5.4B. We did not perform this correction for the C1 class because it would have required new ways to correct for the interactions in the training set. For example, the lower bound positive to negative ratio was violated when considering HippieHQ1.2 and 1.6 together. We now would have had to estimate the ratio of false positives within our data sets, then how many actual interactions between proteins (or close homologs) in HippieHQ1.2 are left for 1.6 etc. Such problems are much less prevalent if at least one target interactor is different from all training proteins. By definition, all its interactions are unknown at prediction time.

5.4.5 Analysis of Most Reliable PPIs

Predictions of PPIs are most useful when it is feasible to validate them via accurate low-throughput wet-lab experiments. This is the roughly case when at least 1 in 20 predicted PPIs is an actual interaction, corresponding to a precision of 5%. Rather pessimistic estimates of precision in Fig. 5.4B reach this point at a recall of about 4%, corresponding to 31,390 pairs for C2 and 60,754 pairs for C3. There is risk that the pairs are somehow biased for certain proteins with which all other proteins interact or that interactions only happen between a few highly connected proteins. We performed a series of analysis on these pairs refuting such concerns.

We analyzed the highest scoring protein pairs of our full interactome prediction. Our first concern was that all the pairs would only map to a small number proteins. This was not the case as shown in Fig. 5.5. Going through the list of pairs from most to least reliable (Fig. 5.5 (A) and (B): left to right on x-axis), the number of unique proteins observed grows almost linearly for both pair sets C2 and C3 (black lines) and in the end maps to 5,687 proteins for C2 and 5,473 for C3. The average node degree is 5.5 for the C2 set and 11.1 for the C3 set.

Another possible issue was that the proteins, despite their diversity, only interact with a few highly connected hub proteins. This would manifest in a few proteins having very high degrees. Also this was not the case: again going from the most to the least reliable pair, the maximum number of interaction partners of any protein (the maximum degree) only jumps in the very beginning at around interaction 1,000 for C2 and 5,000 for C3 (blue lines in Fig. 5.5 (A) and (B)). The worst point is arguably at interaction 2,500 in the C2 distribution, where 276 of 873 proteins interact with the same protein. Even at this point, however, still almost 90% of the interactions (2,500-276=2,224) do not involve this hub protein.

Finally, we calculated the degree distributions for all the highest scoring pairs, again separately for C2 and C3 (Fig. 5.5C). They decrease exponentially, indicating scale freeness according to Barabasi and Oltvai [2004], but the exponent is very small ($\sim$1.3 for C2 and $\sim$0.9 for C3), again showing that interactions are fairly evenly distributed among the proteins.

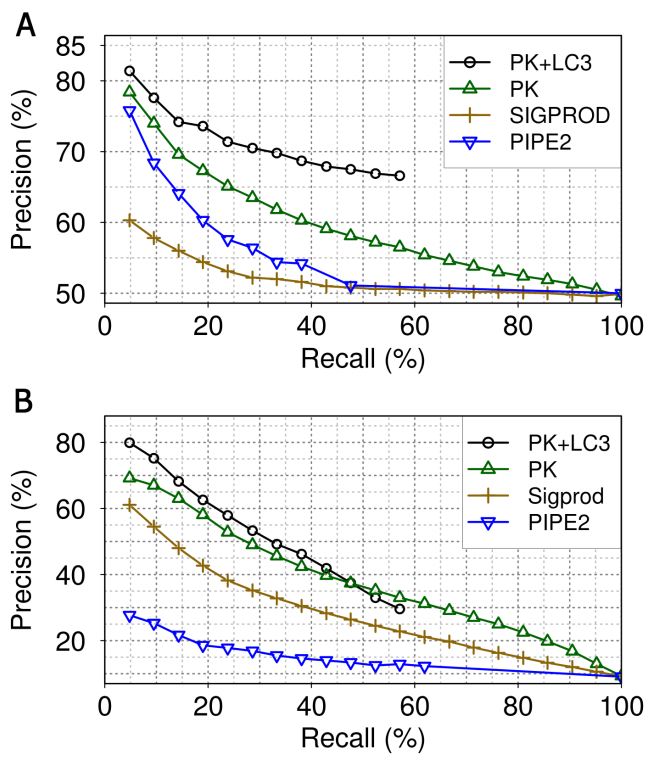


Figure 5.2: Comparison of the best two sequence-based remote interolog detectors and our profile interaction kernel on the Hippie [Park and Marcotte, 2012] data sets. Cross-validations were performed with 10 folds and repeated 10 times (including resampling). In a single fold, precision was calculated every 4 proteins (∼5% recall). Consequently, every point is the average over 10*10=100 values. Standard deviations of precisions are high in low recall levels due to the small sample sizes (overlaps between curves until ∼15% recall; not shown), but standard errors are always below 2% (due to the high repetition number). The differences between the curves are statistically significant at any point except for 'PK-LC3'/'PK' at 47% recall. 'Sigprod' [Martin et al., 2005] and 'PIPE2' [Pitre et al., 2008] are the two best remote interolog predictors according to [Park and Marcotte, 2012]. PK (profile kernel) is our profile interaction kernel and PK+LC3 also includes the LocTree3 filter (Methods). 'PK+LC3' does not span the entire recall range because pairs of proteins predicted not to co-localize are removed from the test sets. (A) Every protein pair is a remote interolog of any other pair. This applies across train/test and positive/negative sets (Methods). Positive-negative ratio is 1:1. (B) Only positive interactions are pairwise dissimilar. Negatives are sampled randomly from the human proteome separately for each fold (ratio 1:10; Methods).

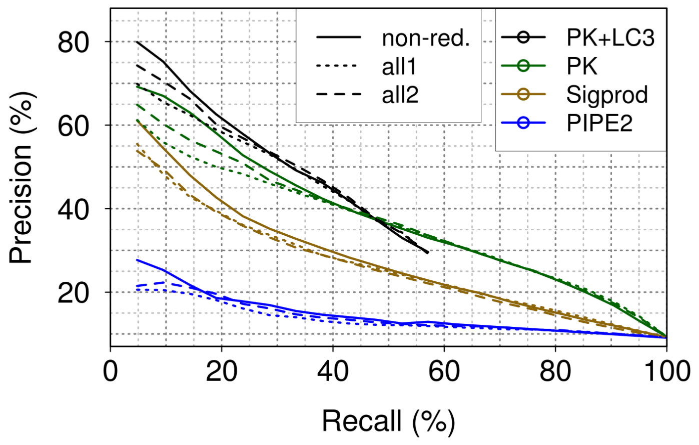


Figure 5.3: Comparison of performance with more positive interactions in the training sets. We increased the number of training interaction compared to the original HippieHQ1.2_C3 data set ('**non-red**') in two different ways. First ('**all1**'), we kept the original proteins but added all high-quality interactions between them. Then ('**all2**'), we also added interactions to other proteins while making sure no new protein had an HVAL greater than 20 to any protein in a test interaction.

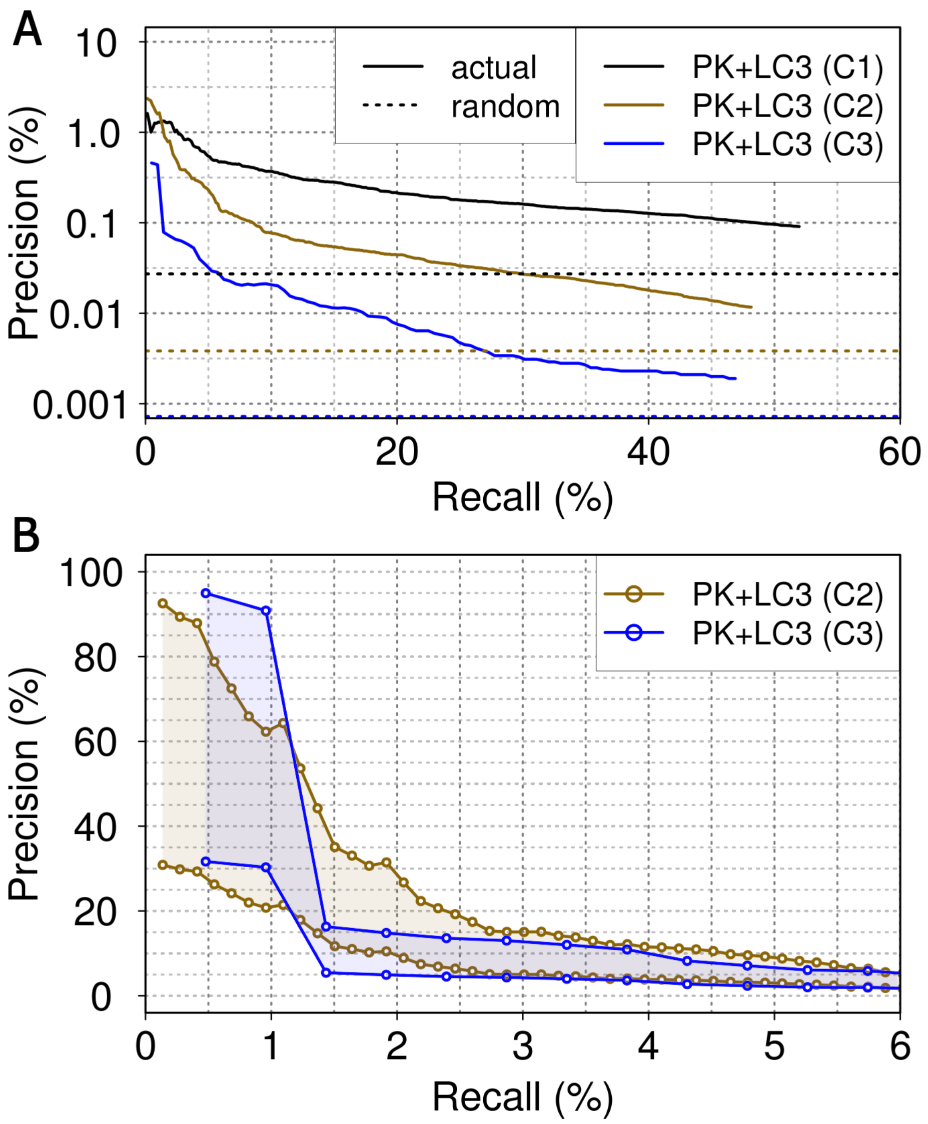


Figure 5.4: Results of full interactome prediction. **(A)** Recall-precision curves of raw results. We predicted all possible pairs between human proteins with our new method, only excluding those used for training (HippieHQ1.2_C3). Each pair was assigned to one of three categories (C1-3) based on the sequence similarity of its proteins to the proteins in the training set (Methods). We tested the accuracy of the predictions with high-quality interactions that had recently been added to the Hippie database (HippieHQ1.6; solid lines). The dotted lines show the precisions of random predictions and correspond to the ratio of positive to all interaction in the respective category. **(B)** Corrected recall-precision curves. The results of (A) are strong underestimates in absolute terms because the actual fraction of positive interactions in human is much higher than in our data set (random lines in (A)). We adjusted the precision values of the curves in (A) to what we believe is a lower (lower curves) and an upper bound (upper curves) for this actual fraction based on recent estimates in the literature. We again differentiated between categories C2 and C3, but not C1 due to strong interference of training interactions.

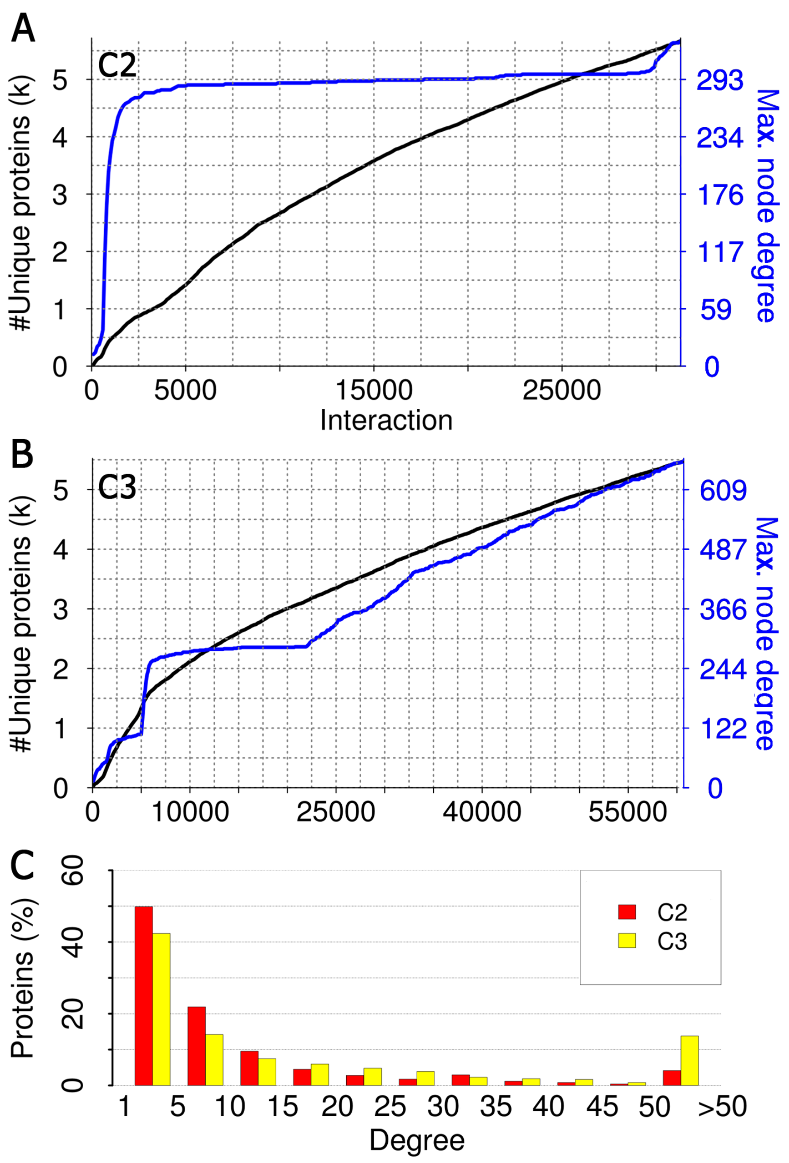


Figure 5.5: Analysis of highest scoring interaction pairs. We analyzed the interactions scoring highest in our full interactome prediction (Results) separately for the C2 and the C3 types of pairs (Methods). **(A)** The C2 pairs are sorted according to score (x-axis), from highest (left) to lowest (right). We go through this list from high to low scoring and record how many unique proteins are among the interactions that we have already observed (black line) and how many interaction partners (i.e. node degree) a protein has at the most (blue line). **(B)** Same as (A), only for C3 pairs. **(C)** The degree distributions, i.e. the probabilities that a protein has a certain number of interaction partners, calculated on all highest scoring pairs. For example, about 50% of all proteins in the C2 set have between 1 and 5 interaction parters.

5.5 Discussion

5.5.1 Hippie as the Source of High-quality PPIs

The most reliable sources for PPIs today are arguably the asymmetric units of high-resolution crystal complexes. We had recently collected all such PPIs from the PDB, but it was too small for statistically significant training/testing after redundancy reduction (160 PPIs). Based on previous experience with the PDB, we also discarded the idea of including lower resolution complexes due to a considerably higher error rate. The Hippie database was a good alternative because of its simple yet effective PPI scoring scheme, combining various indicators of PPI reliability. We chose the top-10% cutoff after sampling the experimental evidence of a number of PPIs in the full Hippie database sorted by score. As a verification, taking the top 25% like the original authors increased the non-redundant data set size to only 1,075 interactions (842 before), but significantly decreased prediction performance on the high-quality interactions in the cross-validations (results not shown). The classification of target interactions is slightly different than in [Park and Marcotte, 2012], but more suited for full interactome predictions because of the way sequence similarity is accounted for. For example, the two highlighted interactions in Fig. 5.1B are likely interologs, but would be category C3 according to [Park and Marcotte, 2012]. They are C1 in our classification.

5.5.2 Profile Interaction Kernel and LocTree3 Improve Sequence-based PPI Prediction

We demonstrate that our series of small methodological changes drastically improve sequence-based PPI prediction. Properly using the evolutionary profile kernel developed by the Leslie group alone is significantly better than other state-of-the-art methods. LocTree3 as a filter is accurate enough to increase precision by filtering out interactions that are improbable compartmentally while retaining those that are. A new rigorous way to reduce interaction redundancy improves classifiers in low recall regions. We show this with three different cross-validations. The first was independently set up by others, the second measures prediction ability without exploiting sequence similarity between proteins and the third simulates human PPI prediction on an interactome scale.

5.5.3 Sampling Strategy of Positive and Negative Interactions of Crucial Importance

Park and Marcotte [2012] discovered a ‘flaw’ in the evaluation of PPI prediction methods as their accuracy on a new pair of proteins depends on whether the proteins have already appeared in the training set. There is, however, no reason to assume that this bias affects only test interactions. Machine learning devices are known to react sensitively to bias in the training set and it is usually a good choice to eliminate it as long as enough data is available. We show that even 6 times more positive redundant interactions in the training set consistently leads to worse classifiers than the original unbiased data set.

Another issue that remained unaddressed by Park and Marcotte [2012] was the effect of the non-interacting pairs. A machine learning device does not differentiate between interacting/non-interaction, only between one class or the other. Therefore, the flaw in the positives should be observable for the negatives, too. This shows in the absolute values of the first and second Hippie-based cross-validation. In the first, the positive-to-negative ratio is ten times higher than in the second and yet our method is hardly better, as is the case for Sigprod. This happens because of sequence similarity between negatives of the training and the test sets (PIPE2 is trained only on positive interactions!). Ultimately, this is not a mistake: we can safely sample random pairs from the human proteome until every protein is selected at least once and claim that negative C2 and C3 cases do not exist. A crucial point, however, arrives when sampling so many negatives that the sample would contain a considerable number of actually interacting pairs. Not including pairs found interacting in any experiments would ultimately transform the problem of PPI prediction into finding reliable among unreliable interactions and neglect the problem of discovering yet unknown interactions. At this point, we believe a compromise is the best solution for optimal performance: there must be enough negatives to cover the human proteome entirely, ideally multiple times, but the set must remain small enough to not contain too much bias from excluded known interactions or wrongly included actual PPIs. The $\sim$80,000 negative pairs that we sampled for our full interactome prediction fell well within this range.

5.5.4 Full Interactome Prediction

We predicted all human protein pairs to provide candidate interactions that can be verified in low-throughput experiments. At the same time, this full interactome scan eliminated the need to subsample negatives in the test set, making estimates particularly in the lower recall regions more precise. The new positive interactions of HippieHQ1.6 constituted an independent positive test set and minimized potential over-optimizations on the HippieHQ1.2 interactions. Finally, the need for computational predictions of PPIs remains big: reliable interactions in Hippie1.6 as defined by the original authors (top 25%) still cover only 5,859 of the 20,249 human proteins. With the quality we demanded for training our classifiers, this is number is further reduced to 3,878.

5.6 Conclusion

In order to improve our understanding of biological processes on the molecular level, we need to determine which proteins are interacting. The slow speed, inaccuracy and limited applicability of experimental methods leaves sequence based prediction of protein-protein interactions (PPIs) a crucial subject in computational biology. We introduced a new SVM-based based approach that relies only on sequence and improves over the best other state-of-the-art methods for human interactions. Using evolutionary profiles and subcellular localization, we increase precision across all levels of recall. We also demonstrated that crucial importance lies on the data sampling strategy for the training data set. Only a carefully selected fraction of all available PPIs is necessary for optimal training and much importance lies on the similarity between among negative interactions. Finally, we presented the first interactome-scale recall-precision curve by predicting all 200 million protein pairs in human and interpolating precision values using recent interactome size estimates for human.

Chapter 6

Accurate Homology-based Inference of Protein Function*

6.1 Outline

Predicting PPIs is immediately linked to network visualization (e.g. in Cytoscape [Shannon et al., 2003]; Introduction) and functional analysis (e.g. with BiNGO [Maere et al., 2005]). One of the most-often used types of analysis is Gene Ontology (GO; [Ashburner et al., 2000]) term enrichment. Most proteins, however are not experimentally annotated and need to be predicted. Any method that de novo predicts protein function should do better than random. More challenging, it also ought to outperform simple homology-based inference. Here, we describe a few methods that predict protein function exclusively through homology. Together, they set the bar or lower limit for future improvements. During the development of these methods, we faced two surprises. Firstly, our most successful implementation for the baseline ranked very high in a recent independent assessment (CAFA; [Radivojac et al., 2013]). However, this work also revealed that the precise details of the implementation are crucial: methods span from top to bottom performers at CAFA. Secondly, we propose a new rigorous measure to compare predicted and experimental annotations. It puts more emphasis on the details of protein function than the other measures employed by CAFA and may best reflect the expectations of users. Clearly, the definition of proper goals remains one major objective for CAFA. Finally, we introduce a new combined sequence-based in-silico GO term predictor that is fast, achieves state-of-the-art accuracy and outputs annotations that can easily be traced back to experimental information.

*This chapter is based on the publication [Hamp et al., 2013b]

6.2 Introduction

6.2.1 Our Contribution to CAFA

UniProt [Consortium, 2011] holds over 22 million sequences (May 2012), but reliable and detailed experimental annotations exist for fewer than 1% of these. GO, the Gene Ontology [Ashburner et al., 2000] has become the gold standard for function annotation and many methods have emerged that predict GO annotations [Rentzsch and Orengo, 2009]. Due to various problems, it has been almost impossible to assess how well these methods perform. CAFA (Critical Assessment of Function Annotations) has arisen to address the challenges by a comprehensive independent comparison [Radivojac et al., 2013]. CAFA also drove our work presented here. Three teams of students implemented three different methods predicting function through homology, i.e. through inference from experimental annotations of related proteins. All three groups began with the same description what to do, and that description was more comprehensive and detailed than many descriptions of methods in papers. Two of the three methods were surprisingly competitive in CAFA and outperformed other similar methods. This triggered the decision to enhance and combine these classifiers in one meta predictor. This post-CAFA method did not participate in CAFA. Would it have, it might have reached the top-10 ranks among all participants. Either way, it suggests several ways for the improvement of function prediction by homology, as demonstrated in this post-CAFA evaluation.

Additionally, we developed a new metric to compare predicted and actual GO annotations. It provides insight into how methods perform with respect to the prediction of the exact functions. This turns out to be largely neglected by existing measures.

6.2.2 Related Work

Several advanced methods have appeared that also predict protein function through homology-based inference. ConFunc [Wass and Sternberg, 2008] first assigns the proteins found via PSI-BLAST to groups so that all members of a group share a particular GO term. Looking at the alignments in each group, the method then identifies conserved functional residues, scores them and only outputs the groups above a certain combined score. GOSLING [Jones et al., 2008] first derives various features of the terms found in the BLAST result (e.g. GO evidence code, E-Value and bit score). Using many decision trees, the prediction is then flagged as either correct or incorrect. PFP [Hawkins et al., 2006] follows an approach very similar to GOtcha [Martin et al., 2004], but also considers highly insignificant BLAST hits and co-occurrence between GO terms. An extension of GOtcha, ESG [Chitale et al., 2009], additionally differentiates between the hits found in different PSI-BLAST iterations. GOstruct [Sokolov and Ben-Hur, 2010] takes the idea of co-occurrence to the next level and builds a sophisticated SVM machinery around 'structured output spaces'. This refers to the extension of the input space (E-Values, asf.) with all experimentally observed GO-subgraphs. FANN-GO [Clark and Radivojac, 2011] uses E-Values as inputs to neural networks. Methods based on data sources other than similarity to already annotated proteins are described in a recent review [Rentzsch and Orengo, 2009].

6.3 Methods

6.3.1 GO (Gene Ontology) for CAFA

GO has three parts: Molecular Function Ontology (MFO), Biological Process Ontology (BPO) and Cellular Component Ontology (CCO). CAFA considered only the MFO and BPO. Both correspond to two directed acyclic graphs and capture different aspects of protein function. Functional keywords (‘GO terms’) are nodes and their relationships are labeled edges. The ontology is hierarchical: following the edges from a node, each new term corresponds to a more general concept of the original function. All paths converge at the root node, which can simply be interpreted as, e.g., has a molecular function. The complete functional annotation of each protein has two subgraphs (MFO and BPO). If a subgraph does not contain all the terms that can be inferred by going from its nodes to the root, we perform a propagation. Given a set of GO terms, this operation refers to its extension with all ancestral terms. Ancestors can be found by following all outgoing paths from the terms to the root: each visited node is an ancestor. If the GO terms have scores (e.g. to reflect their reliability), the latter are also propagated: each parent term is simply assigned the maximum score of its children. Sometimes, a propagated subgraph needs to be reduced again to the leaf terms. A leaf term is not the parent of any other term in the propagation and corresponds to the most exact description of a function for the given protein. In order to integrate the operations above into our methods, we used the graph_path table provided by the GO consortium. It contains all possible paths in the entire GO graph, pre-calculated by specific path inference rules.

6.3.2 Assessment of Predicted GO Annotations

Analogously to CAFA, we use fixed sets of target proteins to compare prediction methods. Each target corresponds to one or two propagated GO subgraphs of experimentally validated terms (depending on whether both BPO and MFO annotations are available or only one of the two). A method is supposed to predict these subgraphs (e.g. the left tree in Fig. 6.1) and assign a reliability between 0.0 and 1.0 to each predicted term (e.g. green nodes in Fig. 6.1). Then we assess their accuracy in the following ways, separately for the MFO and BPO. For the first two measures, we exclusively used the original CAFA implementations, GO version, targets and target annotations. Only to implement our new leaf threshold measure, we slightly adapted the programs.

Top-20. Given the prediction of a single protein, the top-20 measure first reduces the prediction to the terms with the highest reliability (Fig 6.1: green nodes with score 0.8). It then defines recall as the number of correctly predicted GO terms divided by the number of all true GO terms. Precision corresponds to the number of correctly predicted GO terms divided by the number of all predicted GO terms. In Fig. 6.1, for example, recall is 1/11 = 0.09 and precision is 1/2 = 0.5. If a target is not predicted at all, it is assigned a recall of 0.0. Precision is not calculated in such a case and has no influence. Repeating this for all targets, we obtain the average recall and precision. This is the first point in the recall-precision curve. In order to fill the curve, we gradually add the terms with 2nd, 3rd, . . . , 20th highest reliability to the predictions and recalculate all of the above.

Threshold. The threshold measure [Clark and Radivojac, 2011] follows a similar concept as top-20. Instead of considering a certain number of terms for each target at a time the measure demands a threshold between 0.0 and 1.0. In case of a threshold of 0.82, for example, each prediction is reduced to terms with a reliability greater than or equal to 0.82. Recall and precision can then be calculated analogously to the top-20 measure. A curve is obtained by lowering the threshold in steps of 0.01 from 1.0 to 0.0. Leaf threshold: The leaf threshold measure, finally, operates exclusively on the leaves of a propagation (red nodes in Fig. 6.1). First, predicted and experimental subgraphs are reduced to their leaf terms (Fig. 6.1: experimental leaves on the left, predicted leaves on the right). Then, we define a threshold T as before, e.g. T=0.82, and reduce each prediction to the leaves with a reliability $\geq$T. The recall of a single prediction is given by the number of correctly predicted leaves divided by the number of all experimental leaves. Precision is defined analogously. Consequently, we can derive a recall-precision curve in the same way as for the threshold measure. In Fig. 6.1, we obtain the first non-empty prediction as soon as the threshold reaches 0.80 (the highest score of all predicted leaves is 0.8). In this case, recall and precision correspond to 0/3 = 0.0 and 0/1 = 0.0.

Leaf Threshold. The leaf threshold measure is orthogonal to the top-20 and threshold measure: in the case of low recall, for example, the former two measures remove specific GO terms from the prediction and retain only the more general terms. Naturally, more general terms have a higher chance to overlap with the experimental propagation than specific terms, resulting in higher precision. However, the leaves of this reduced prediction are not more likely to overlap with the leaves of the experimental annotation. If the full prediction was the best estimate of the experimental leaves, the reduced version could even result in recall=precision=0.0 by the leaf threshold measure, because the reduction might remove all correctly predicted leaves. Our new measure assesses how well the exact functions of a protein are predicted. Too general or specific predictions are penalized.

Maximum F1 score. The top-20 and threshold measure were the two main metrics in the CAFA meeting. The leaf measure is introduced here for the first time. In order to rank methods, the CAFA organizers additionally used the maximum F1 score over all recall-precision pairs obtained with the threshold measure (Fmax). The F1 score is defined as:

$$F_1 = \frac{2 * precision * recall}{precision + recall} \tag{6.1}$$

We also employed Fmax in order to choose among alternative parameters during method development after CAFA.

6.3.3 Homology-based Methods

All the methods that we presented at CAFA were developed as part of the exercises of a regular lecture at the Technical University in Munich (year 1-3 in bioinformatics master). Due to limitations in time and resources, our methods had to focus on a k-nearest-neighbor approach and to only use the hits returned from a PSI-BLAST [Altschul et al., 1997] query against Swiss-Prot [Schneider et al., 2009]. They were supposed to optimize the F1 score (threshold measure) of the leaf term with the highest reliability. We split the 16 participating students into three groups, each of which had to develop an own implementation. After 8 weeks of time and one week before the CAFA submission deadline, we received the following three methods. Their key features are summarized in Table 6.1.

6.3.3.1 StudentA

Begin with 2-iteration PSI-BLAST against all Swiss-Prot proteins with GO annotations (E-Value <0.1). Extract GO terms of the six top PSI-BLAST hits (or all if fewer than 6 hits found). Each identified GO term is scored 1.0 if the term is found in all 6 hits and 0.5 otherwise. Once the term-score pairs have been extracted, only the leaf terms of their propagation are retained. Then apply the following filter to reduce functional redundancy: (i) create branches by propagating each predicted leaf term separately; (ii) calculate all pairwise branch overlaps, with the overlap being defined as the number of common GO terms in both branches divided by the average branch size.

Next, cluster all branches so that each pair from two different clusters overlaps less than 10%. For each cluster, the branch with the longest path to the root is chosen, reduced to its original leaf term with the original score and output to the user. As the redundancy reduction may filter out highly supported terms, we apply a

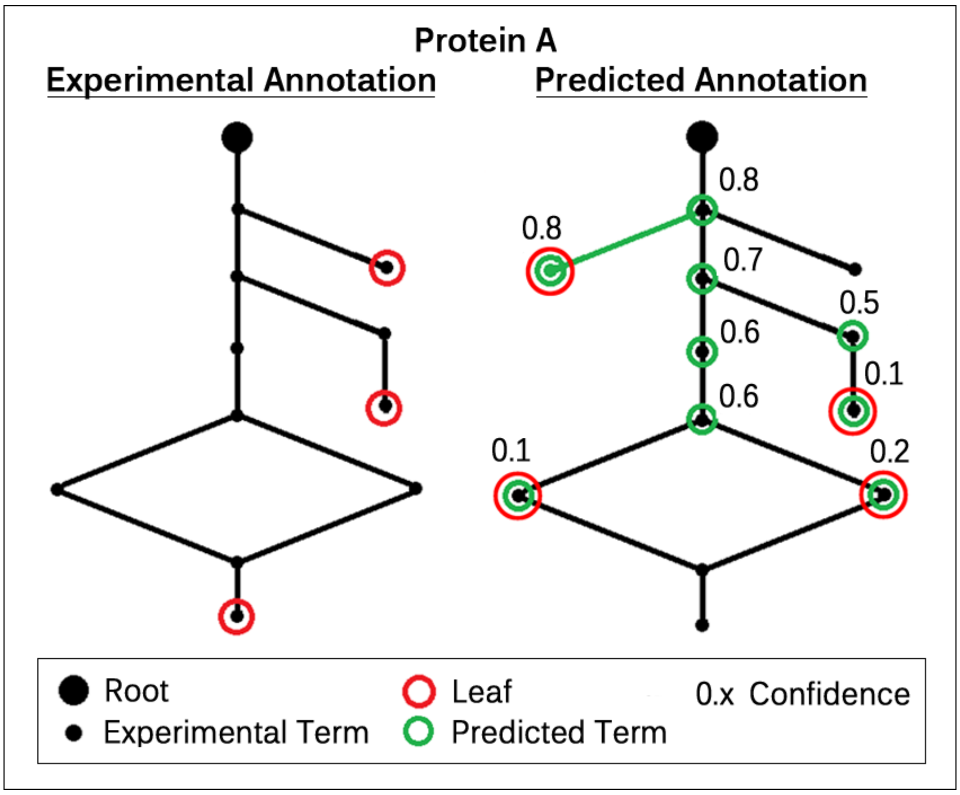


Figure 6.1: A functional annotation and its prediction. This figure shows one annotation of a sample protein A and its prediction. Each node in a graph corresponds to one GO term and the edges to relationships such as 'is a' or 'part of'. The edges always point to the root node (either 'MFO' or 'BPO'), which by itself is not informative and discarded in every evaluation. For clarity, the left subgraph only shows the experimental annotation of A. This means, all GO terms have either been experimentally verified or inferred from the same. The red circles indicate the leaf terms, i.e. the nodes which are not a parent of any other term. In the right subgraph, we see the experimental annotation again, but overlaid with predicted terms (green) and their reliabilities. This time, the leaf terms correspond to the predicted GO annotation, instead of the actual annotation.

final correction: if any pair of branches from previous steps overlaps over 90%, the term common to both and with the longest path to the root, i.e., the lowest common ancestor, is added to the result. See Fig. 6.2 for an illustration.

	StudentA	StudentB	StudentC
Input Features	GO term counts	E-Values	GO term counts; percentage positives
Scoring Scheme	1	2	2
Propagation Types	maximum child	maximum child	maximum child; cumulative
Score Normalization Across Proteins	No	Yes	No

Table 6.1: Comparison of student methods. In this table, we have summarized the key differences between student methods. Input features include: the number of times a GO term appeared in the annotations of homologous proteins; the E-Values of the homologous proteins; and the percentage of 'positive' columns in their alignment matrices. Some groups used more than one way to score a GO term or differed during the propagation of a prediction by assigning a node the maximum value of its children or their sum. StudentB normalized the final score of a GO term to improve comparability among proteins.

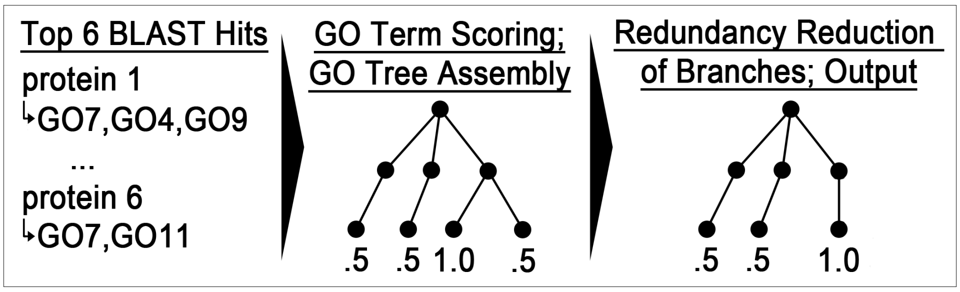


Figure 6.2: Flow chart of StudentA. StudentA first reduces the BLAST output to the best 6 hits. GO terms that are part of the annotation in all 6 hits are assigned a score of 1.0, all others 0.5. Then the predicted GO graph is assembled by propagating the scores and pruned again during a functional redundancy reduction (see text). This reduced graph is output to the user..

6.3.3.2 StudentB

Begin with 2-iteration PSI-BLAST against all Swiss-Prot proteins with GO annotations (E-Value <0.002 for 1st and E-Value <0.01 for 2nd round). Each PSI-BLAST hit is associated with the propagation of its GO terms and each term in the propagation is associated with the PSI-BLAST E-Value of the hit. We then define two scores.

The template quality score gauges the reliability of the entire PSI-BLAST query with respect to the goal of assigning function. First, we calculate the raw template score as the average over the logarithms of all returned PSI-BLAST E-Values plus twice the standard deviation (also derived from the log(E-Value)). The standard deviation is added to correct for cases with relatively low top scores and relatively high averages. This raw template score is normalized into a value from 0 to 1 by mapping it into a percentile bin obtained from running PSI-BLAST in the same fashion on a sample of all Swiss-Prot proteins (e.g. a score obtained by 90% of the samples for all Swiss-Prot is scored 0.1=1-0.9). We call this percentile the template quality score.

The combined leaf score measures the reliability of each predicted leaf. First, we compile the propagated set of all GO terms for all PSI-BLAST hits. Each term can occur in multiple hits and thus be associated with multiple E-Values. The support of a term is defined as the sum over the logarithm of its E-Values divided by the sum of the logarithm over the E-Values of all hits. The combined leaf score of a leaf in the set of GO terms above is then given by the average support of itself and all of its ancestors. Finally, we multiply template quality and combined leaf score for each leaf, combine all the leaf-score pairs in one set and output its propagation to the user. See Fig. 6.3 for an illustration.

6.3.3.3 StudentC

Begin with 2-iteration PSI-BLAST against all Swiss-Prot proteins with GO annotations (E-Value <0.1). Count how often a particular GO term appeared in the PSI-BLAST hits (without propagation). All nodes with counts are propagated through the GO tree. Instead of taking the maximum count of all children at each parent node, however, their values are summed up and added to that of the parent node (normalization to [0,1] by division by maximal value). We call this type of scoring the max support. The PSI-BLAST scores, on the other hand, are considered as follows. For each PSI-BLAST hit, we first read off the positive identity. This value is included in the default BLAST output and corresponds to the number of positives divided by the alignment length. (Each mutation column in the default BLAST output with a positive score by BLOSUM62 is a positive.) Then, we multiply the max support of each term with the highest associated positive identity (we may have many positive identities, because a GO term can be associated with multiple PSI-BLAST hits). The method outputs only the one branch corresponding to the highest scoring leaf term. See Fig. 6.4 for an illustration.

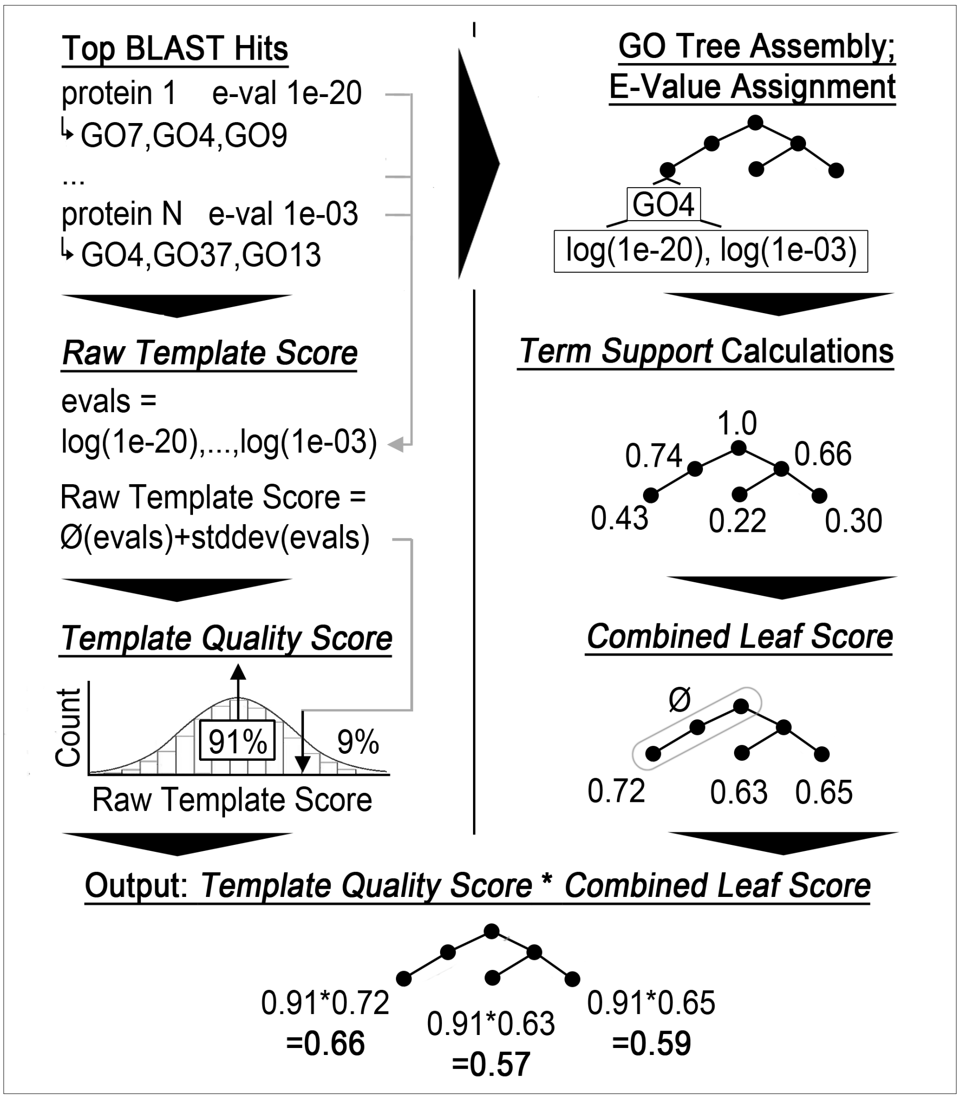


Figure 6.3: Flow chart of StudentB. StudentB first logarithmizes the E-Values of all BLAST hits and averages them. The result is mapped into a range from 0 to 1 by looking up its percentile in a precompiled distribution. This percentile is the Template Quality Score and reflects how well we can predict the entire target. To score single terms, we multiply it with the score of each predicted leaf, i.e. the Combined Leaf Score. This is the average of the logarithmized E-Values of the nodes on the path from the leaf to the root. The propagation of the leaf terms and their scores is output to the user.

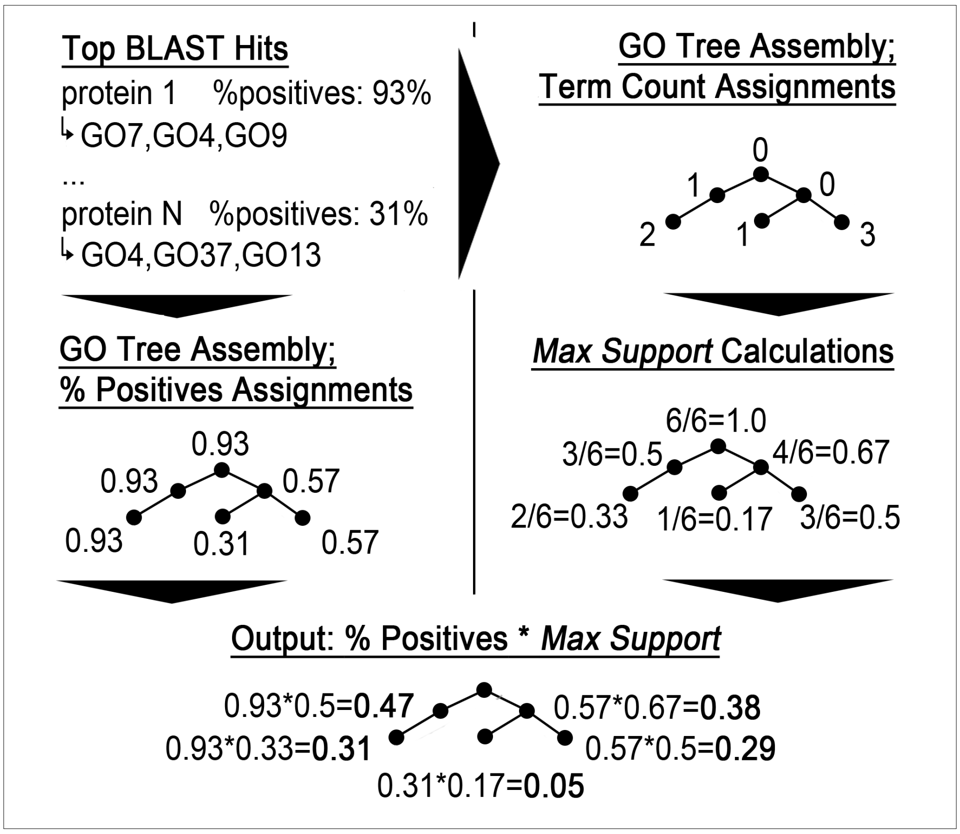


Figure 6.4: Flow chart of StudentC. StudentC first counts how often each GO term appeared in the BLAST hits and performs a cumulative propagation: for each inner node, all counts of its child terms are summed up and added to its own count (depth-first traversal). Dividing each count by that of the root term, we obtain a score between 0 and 1 for each term. In parallel, we calculate a second score for each term by assigning it the maximum percentage of positives of all associated hits (see text). Finally, we multiply the two scores, determine the highest scoring leaf term and output only its propagation to the user.

6.3.4 Post-CAFA Re-parameterization

After CAFA, we parameterized the above three basic homology-inference methods. For StudentA, we introduced the options to exclude predictions with a score of 0.5 and to choose the number of PSI-BLAST hits to consider (before: 6; now: 1, 5 or 9). For StudentC, we added alternative PSI-BLAST E-Value thresholds (before: 10^{-1}; now: $10^{0}, 10^{-3}$ or 10^{-6} and percentage pairwise sequence identity as an alternative to the positive identity. We also enabled the optional output of all branches, instead of restricting it to the most probable one. The original implementation of StudentB had a bug: an alternative graph_path table inverted the order of the columns by mistake. The results of this bug were submitted to CAFA. We fixed the bug and allowed for alternatives in the thresholds for E-Values and maximum numbers of PSI-BLAST hits (E-Value before: 10^{-2}; now: $10^{0}, 10^{-3}$ or 10^{-6}; max. number of hits before: 250 [PSI-BLAST default]; now: 5, 50 or 500).

For all methods, we also add the choice of the number of PSI-BLAST iterations (before: 2 for all methods; now: 1, 2 or 3). Finally, we enabled the filtering out of Swiss-Prot annotations with unclear experimental support (optional restriction to the following experimental GO evidence codes: IDA, IMP, IPI, IGI, IEP, TAS, IC, EXP).

The re-parameterization created 36, 54, and 72 different parameter combinations for StudentA- C, respectively. We optimized the parameters by picking the combination leading to the highest Fmax (threshold measure; Eq. 6.1) on a hold-out data set. This data set comprised all Swiss-Prot proteins annotated with experimentally verified GO terms in 2010 ('Set 2010'). All proteins annotated before 2010 served as templates ('Set <2010'). This ascertained that there was no overlap to the CAFA targets. In the following, we refer to the optimized student methods as StudentA'-C'.

6.3.5 Post-CAFA Method Combination

Due to the end of the lecture during which the methods were developed, we could not combine them. We did this also post-CAFA. We randomly split Set 2010 into two equal parts (Set 2010a and 2010b). Parameters were optimized on the first split (2010a; as before, only with 2010a instead of 2010). These optimized variants of StudentA-C (say StudentA"-C") were applied to the second split (2010b). Then, we switched the roles of the two sets and repeated the procedure to obtain predictions for each protein in Set 2010. With these predictions, we trained a commonly used meta classifier [Ming and Witten, 1999], namely a weighted least-squares linear regression model. This corresponded to the formula x*A' + y*B' + z*C' + i = p, where A', B' and C' are the results of the student methods for each predicted GO term and [x-z] and i are the coefficients to optimize in the regression so that p reflects the reliability of the GO term. In order to meta-predict a new target protein, we first annotate it with methods StudentA'-C'. Each predicted GO term is then converted into a vector of three elements (one dimension for each method) and put into the formula above. The resulting value of p is the reliability of the GO term for the given target. We refer to this predictor as MetaStudent'.

6.3.6 Baseline Classifiers

The CAFA organizers implemented the following three baseline classifiers to gauge the improvement of current function predictiors over old or naive methods [Clark and Radivojac, 2011]. (1) Priors. Every target has the same annotations and each term's score is the probability of that term occurring in Swiss-Prot. (2) BLAST. Target annotations are simply the maximum sequence identity returned by BLAST under default parameters when aligning a target with all proteins annotated with a given term. (3) GOtcha. Using the same BLAST results as BLAST, GOtcha [Martin et al., 2004] I-Scores are calculated as the sum of the negative logarithm of the E-Value of the alignment between the target protein and all proteins associated with a given term. Additionally, we introduce Priors', which simply returns the entire GO annotation of a random Swiss-Prot protein. Scores are assigned as in Priors.

6.3.7 Data Sets

We used five different data sets for method development and evaluation. All are exclusively derived from GO and the GO annotated proteins from Swiss-Prot and only differ in their release dates. The first three methods used the GO/Swiss-Prot releases from Oct. 2010 ('Set <2010_10') for both development and group-internal evaluations. We updated to the versions from Dec. 2010 ('Set <2010_12') and submitted all 48,298 CAFA targets with each method. For post-CAFA developments, we used the release of Jan. 2010 as the source for template annotations ('Set <2010'). The independent data set needed for post-CAFA parameter optimization then contained all proteins annotated between January and December 2010 ('Set 2010'). Analogously to CAFA, we ignored proteins that had any GO annotation before January 2010 and only retained experimental annotations in the remaining proteins. Experimental GO evidence codes were: IDA, IMP, IPI, IGI, IEP, TAS, IC, and EXP (same as in CAFA). 'Set_2010' contained 1752 targets with BPO and 1351 with MFO annotations.

The CAFA organizers provided the original CAFA targets (436 with BPO and 366 with MFO annotations). They correspond to the proteins annotated between January and May 2011 ('Set 2011'). This set was derived following a similar algorithm as those in 'Set 2010'. The difference was that the CAFA organizers also excluded annotations from the GOA project in proteins annotated before January 2011 (a resource we left untouched). We used the annotations in 'Set <2010_12' to predict proteins in 'Set 2011'. Note that this implied that all our post-CAFA optimization could have been accomplished completely before the submission to CAFA (we just had not been fast enough). Nevertheless, we clearly label all the new methods as 'post-CAFA' in this work.

6.4 Results

6.4.1 Wide Performance Spread of Homology-based Inference

Our three homology-based predictors of protein function (StudentA-C) performed very differently (Fig. 6.5, dark blue; note: all data compiled exclusively on the CAFA targets and with data available before the CAFA submission). This was true for both categories, namely for biological process (BPO, Fig. 6.5, top panels) and for molecular function (MFO, Fig. 6.5, lower panels) and for all performance measures (Fig. 6.5: each column signifies one particular measure). For instance, StudentA performed slightly better than StudentC by the top-20 measure (Methods) and slightly worse by the threshold criterion (Methods). While StudentA and StudentC mostly surpassed the baseline tests (PRIORS and BLAST), they even topped the GOtcha baseline (dark green) for many thresholds. In the BPO category (threshold measure), StudentC actually outperformed all but two of the other 36 CAFA predictors until a recall of about 0.2 (not shown). Note that the curves for StudentA-C in Fig. 6.5 are identical to those calculated by the CAFA organizers.

6.4.2 Post-CAFA Optimization Renders Homology-based Inference Competitive

When we changed our methods post-CAFA, we carefully avoided using any information that was not available at the CAFA submission deadline. Nevertheless, we are treating our optimized predictors (StudentA'-C', and MetaStudent') differently to clearly mark the point that these methods did not compete at CAFA. All changes were straightforward (e.g. optimization of simple thresholds) in the sense that they did not require any of the knowledge that we gained at CAFA. They would have been done by anyone with enough time before submission. This reality is important because they improved performance markedly. Our best single method that exclusively used homology information (StudentC') even outperformed the advanced method GOtcha in almost all respects. MetaStudent', the combination of all three methods, was consistently on par or better than all others, including GOtcha.

6.4.3 Leaf Threshold Measure Suggested Very Different View

There is evidence that the top-20 and the threshold measure penalize methods that provide a decision as to which function is predicted and favor methods that output huge lists of scored GO terms (Discussion). If so, their use as the scoring to be optimized may go against the interest of users (Discussion). In contrast, our new leaf threshold measure (Methods) favors predictions with reasonable amounts of terms over those with overly many. It first reduces all predicted GO terms to the leaf terms and then compares those to the leaves of the true annotation. Achieving, e.g., a recall of 1.0 simply by outputting the entire GO is therefore impossible. This reveals just how bloated predictions can be (Fig. 6.5: rightmost panels): For instance, the baseline background 'method' Priors (i.e., predicting all GO terms for each target and scoring them by frequency) is now numerically reduced to where it belongs, namely to a very bad performance. Priors' (i.e., randomly picking a protein annotation from Swiss-Prot and scoring terms by frequency), on the other hand, shows up competitive for levels of recall <0.1 in the MFO category. Since also Priors stopped at recall 0.1, there appears to exist a very common low level leaf in GO. In the BPO, a larger and more complex hierarchy, Priors' fails, too. It remains unclear whether the bad performance of other baseline classifiers (BLAST, GOtcha) under the leaf threshold is due to unnecessarily large predictions in order to achieve high recall or to deeper methodological problems. In any case, our results show that even under this most rigorous measure, we can see fine grained separations between methods.

6.4.4 Homology-based Method Ranks Very High

CAFA decided to rank methods according to the Fmax score on the threshold measure (Eq. 6.1). For compatibility, we followed this approach (Table 6.2: all methods provided in this contribution, plus the top mark presented at CAFA, namely FunctionSpace). For comparison, we also provide the scores for the top ranking method (FunctionSpace). The complete list presented at CAFA contained 36 methods; for 15 of these, the ranks have been released. Of all methods in Table 6.2, only StudentA, Gotcha, and 'best' were in the list. StudentB-C were excluded because only one method per group was considered (we had correctly chosen Student A for this). StudentA'-C' and MetaStudent' were not ranked as they were developed post-CAFA with pre-CAFA data. The best homology-only method (StudentA) was in the top 8 only for BPO, while it dropped to rank 13 for MFO. In contrast, two of the baseline methods, namely Priors and Gotcha both ranked higher for MFO than for BPO. In fact, StudentA ranked worse than both baseline methods according to MFO and better than both for BPO. All our homology-only post-CAFA methods reached F1 scores consistently higher than that of StudentA. Our best method (MetaStudent') performed rather well by this criterion and would have ranked in the list of the top three at CAFA, had the method been completed in time. Its F1 scores would have been very similar to those of the top contender (FunctionSpace).

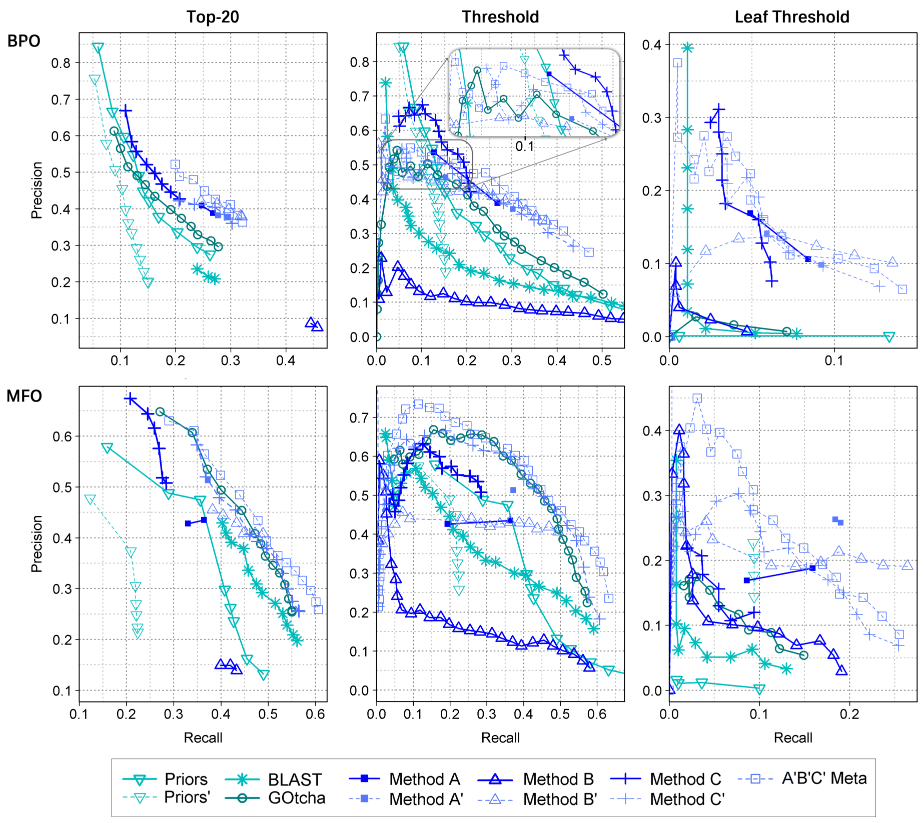


Figure 6.5: Results of evaluations before and after CAFA. Here, we show the results of all methods for each ontology and measure. Baseline classifiers share the same color (cyan), just like methods corresponding to the same design, but different parameter values (blue). Curves derived from the CAFA organizers are solid and bold, otherwise thin and dotted. As the area between recall 0.0 - 0.2 and precision 0.45 – 0.55 is extremely crowded in the BPO threshold measure plot, we provide an enlarged view with the inlet. In the BPO leaf threshold measure plot, Priors' is at the origin (0.0, 0.0).

6.4.5 Ranks Varied Significantly with Measure

We expanded the ranking to include all measures shown in Fig. 6.5 (Table 6.3). As before, we reduced each recall-precision curve to one maximum F1 score and used this value to define the rank of the respective method for this measure. There were 11 methods, so that ranks always ranged from 1 to 11. Depending on the method, the top and lowest ranks differed by 2 to 7. The average difference for alternative

	F1 BPO	Rank BPO	F1 MFO	Rank MFO
Presented@CAFA				
GOtcha	0.29	13	0.47	4
BLAST	0.21	-	0.34	-
Priors	0.27	15	0.41	12
StudentA	0.32	8	0.40	13
StudentB	0.15	-	0.20	-
StudentC	0.28	14	0.36	-
Best@CAFA	0.37	1	0.49	1
Post-CAFA				
Priors’	0.20	-	0.29	-
Student A’	0.33	8	0.43	10
Student B’	0.36	3	0.45	7
Student C’	0.34	6	0.48	3
MetaStudent’	0.36	3	0.48	3

Table 6.2: Ranking of methods with respect to the maximum F1 score of the threshold measure curves. This table shows the maximum F1 score (Fmax) of each threshold measure curve in Fig. 6.5 and its rank in the list of competing methods which was shown at CAFA. This list actually consists of 36 predictors, but only the scores and ranks of the top 15 performers have been released. Classifiers which are actually part of this list are kept in bold. Ranks of other methods are hypothetical, either because calculated after CAFA or because discarded by the CAFA organizers. They considered only one method per participating group and we chose method A. Results for StudentB were compiled with the bug (Methods).

measures was 3.8. Put differently, no single method always had the same rank and, on average, the differences spanned over one third of the entire spectrum.

	BPO			**MFO**		
	Top-20	Threshold	Leaf	Top-20	Threshold	Leaf
Priors	8	8	11	7	6	11
Priors’	10	10	10	10	10	6
BLAST	9	9	9	6	9	10
GOtcha	6	6	8	2	3	9
StudentA	5	5	5	8	7	5
StudentA’	3	4	4	5	5	2
StudentB	11	11	7	11	11	7
StudentB’	2	2	1	3	4	1
StudentC	7	7	6	9	8	8
StudentC’	4	3	3	4	2	4
MetaStudent’	1	1	2	1	1	3

Table 6.3: Ranking of methods by maximal F1 score for various measures. We calculated the maximum F1 score (Eqn. 6.1) for each method and curve presented in Fig. 6.5 and ranked the methods accordingly. The number in each cell is the rank of the method in the respective category. As we evaluated 11 different methods, ranks range from 1 (best) to 11 (worst). Results for StudentB were compiled with the bug (Methods).

6.5 Discussion

6.5.1 A New Baseline for Simple Homology Based Inference

CAFA provided a common ground to test our student methods with experimental annotations unknown at the time of prediction. These initial methods defined some new lower bounds for the performance of simple homology-based inference (e.g. StudentA and StudentC for BPO). Our post-CAFA optimizations were carried out exclusively with data available before the CAFA submission deadline. Hence, we postulate that our new results could have been presented at CAFA, had we been ready in time. They show that simple homology based inference can compete with state-of-the-art prediction methods. Considering the wealth of data that we did not use, this suggests large room for improvement.

6.5.2 Similar Methods Can Differ Substantially

Nearest neighbor based homology inference can be realized in surprisingly many ways. The details of an implementation may lead to almost random predictions (StudentB, BLAST) or to state-of-the-art tools (optimized student methods, GOtcha). This pertains to both design choices and other free parameters. For example, score normalization across targets appeared deleterious for low-recall precision (StudentB'). In contrast, restricting predictions to the most probable leaf can boost this aspect of performance (StudentC). Analyzing the impact of various free parameters during the optimization process (data not shown), we also found the choice between using all or only experimental GO annotations to be critical: StudentA' and StudentC' only approached the performance of GOtcha in the MFO category because they focused on experimental annotations.

6.5.3 CAFA Measures Seem to Favor Unspecific Predictions

The difference in the performance of Priors and Priors' for the top-20 and threshold measures must be the result of Priors' reducing predictions to observed protein annotations. Both predictors use the same scoring (term frequencies). To understand this effect, consider a minimal scoring threshold of 0.1. This defines one point in a recall precision curve. Any term with a frequency of, e.g., 0.15 will be a correct prediction for about 15% of the targets with Priors, because this predictor always predicts the entire GO ontology. In contrast, Priors' will pick this term in only about 15% of all cases, reducing the chance to predict it correctly to 0.15*0.15 = 0.02 = 2%. Put simply, Priors finds many more true positives than Priors' at any reasonable threshold. Superfluous predictions, on the other hand, should be more frequent for Priors than for Priors', because Priors always predicts all terms above 0.1 and Priors' only a fraction. Depending on which of those two error forces is stronger, either Priors or Priors' is preferable for top-20 and threshold. At least for the GO and typical Swiss-Prot annotations, the incorporation of many false positives in the prediction seems highly favorable (Priors). We would be surprised if this effect was limited to Priors and Priors' and not observable for most other scoring schemes.

6.5.4 The Leaf Threshold Measure Challenges Existing Metrics

The leaf threshold measure sheds a brighter light onto many results presented at CAFA. Now, baseline classifiers are hugely outperformed by, for instance, homology-based methods and also a different type of random predictor is favorable (Priors'). This can be explained by a simple example: assume the propagated annotation of a single-function aldolase enzyme contains four terms. Predicting it, we obtain, e.g., a subgraph of 20 terms in which the four highest scoring terms are correct and the others are wrong According to the top-20 and threshold measure, this is very good: We reach a recall of 1.0 with a precision of 1.0 (first four terms). Only when considering all predictions, precision drops to 0.2 (4/[4+16]). We argue that this type of measure does not reflect performance from a user's point of view. Using all predicted terms, he or she will end up with the precision of the highest recall. Even considering the predicted reliability of each term, the user still has to decide which terms are correct and which are not: the reliability threshold separating true from false terms is unknown. The odds of choosing a threshold exactly between the score of the 4th term (correct) and the 5th term (false) are low. But exactly this choice is assumed by the top-20 and threshold measures and can highly bias exact function prediction, as evident in the results of the leaf threshold measure. The latter yields a recall and precision of 1.0 only if the prediction solely consists of the first four highest scoring terms. Note that also restricting the prediction to scores above a certain threshold per default does not solve the problem: first of all, we would have to find this global threshold that leads to the best leaf accuracy (a minimal change of the threshold can lead to entirely new leaves and a new number of leaves). This, however, should clearly be the task of the method developer, not of the assessor. Secondly, even with the best threshold, a small internal change of the method might still lead to better performance. For instance, consider a variable threshold that depends on the target (instead of one global threshold) or a restriction of the output to terms that have already been observed as leaves in Swiss-Prot. Again, this is the task of the method developer, not of the assessors.

We are convinced that the leaf threshold measure will be an important extension for CAFA2. Getting to points such as 80% recall at 10% precision with the current measures is really not a valid goal for function prediction. Rather the opposite, best performance should imply high accuracy/precision. This direction is supported by our new measure.

6.5.5 Method Ranking Might Be Misleading

Our ranking of methods (Tables 6.2 and 6.3) followed guidelines proposed earlier [Marti-Renom et al., 2002, Eyrich et al., 2003, Koh et al., 2003]. For example, we always used the same data set and scoring schemes for all methods. However, no clear 'winner' emerged, as it depended on the measure which method ranked top. MetaStudent', for instance, performed best on average, but ranked only 3rd behind StudentB' and StudentA by the leaf threshold measure in the MFO category (Table 6.3). In addition, there are many alternative relevant performance measures and many new methods are yet to be published. For future CAFA experiments, it will therefore become even more important to avoid 'crowning winners' (unless methods stand out by all means) and to focus on method groups suited best for certain disciplines.

6.6 Conclusion

In this chapter, we have explored the design and parameter space of homology-based function prediction based on nearest-neighbor principles. We find that small methodological and parametric changes can cause dramatic differences in performance. Consequently, we propose several new algorithms that outperformed similar methods either at the CAFA meeting or in the assessment presented here. Consistently showing superior accuracies, our best predictor even imposes itself as a substitution of the popular GOtcha method suggesting that a loose coupling of diverse nearest-neighbor methods can yield state-of-the-art performance. Finally, we challenge existing evaluation protocols. Apparently, the performance measures on which CAFA focused inadequately encourage methods to abstain from making specific function predictions and to instead provide huge lists of scored GO terms. This appears a push into the wrong direction. Therefore, we introduced a new rigorous measure that corrects for this shortcoming as a candidate for the assessment at CAFA2.

Appendix A

Functions of Families With and Without Interface Variability

We collected interacting families which show absolutely no sign of interface variability (0.9-1.0 bin in distribution $D_{Interolog}$ at 100% with Face Position Similarity). Then, we determinded the GO terms of the proteins in each of these family pairs and counted how many family pairs were associated with a particular GO term. We show the results in the first half of the following table. Then, we performed the same analysis for family pairs which show very high interface variability (0.0-0.5 bins in distribution $D_{Interolog}$ sum up to 100% with Face Position Similarity). In both tables, the column "Unique" indicates that the respective GO term was only found in that particular group of family pairs. Terms which only appeared once in either group are not shown.

GO Number	GO Term	Family Pairs (max. 17)	Unique
GO:0003674	molecular function	16	No
GO:0005488	binding	16	No
GO:0043169	cation binding	11	No
GO:0043167	ion binding	11	No
GO:0046906	tetrapyrrole binding	7	Yes
GO:0051540	metal cluster binding	6	No
GO:0016787	hydrolase activity	6	No
GO:0003824	catalytic activity	6	No
GO:0051536	iron-sulfur cluster binding	6	No
GO:0004175	endopeptidase activity	5	No
GO:0008233	peptidase activity	5	No
GO:0046872	metal ion binding	5	No
GO:0070011	peptidase activity, acting on L-amino acid peptides	5	No
GO:0046914	transition metal ion binding	4	No
GO:0030234	enzyme regulator activity	3	No
GO:0030246	carbohydrate binding	2	Yes
GO:0008236	serine-type peptidase activity	2	Yes
GO:0004857	enzyme inhibitor activity	2	No
GO:0017171	serine hydrolase activity	2	Yes
GO:0015077	monovalent inorg. cation transmembr. transp. act.	2	No
GO:0070003	threonine-type peptidase activity	2	Yes
GO:0005506	iron ion binding	2	Yes
GO:0005215	transporter activity	2	No
GO:0008324	cation transmembrane transporter activity	2	No
GO:0022891	substrate-specific transmembrane transporter activity	2	No
GO:0022890	inorganic cation transmembrane transporter activity	2	No
GO:0022892	substrate-specific transporter activity	2	No
GO:0015078	hydrogen ion transmembrane transporter activity	2	No
GO:0015075	ion transmembrane transporter activity	2	No
GO:0022857	transmembrane transporter activity	2	No
GO:0005515	protein binding	2	No

Table A.1: GO Terms of families with exclusively alternative interfaces

GO Number	GO Term	Family Pairs (max. 18)	Unique
GO:0003674	molecular function	17	No
GO:0005488	binding	16	No
GO:0043169	cation binding	11	No
GO:0043167	ion binding	11	No
GO:0003824	catalytic activity	10	No
GO:0005515	protein binding	10	No
GO:0046872	metal ion binding	9	No
GO:0016787	hydrolase activity	6	No
GO:0000166	nucleotide binding	5	No
GO:0017076	purine nucleotide binding	5	No
GO:0008092	cytoskeletal protein binding	5	Yes
GO:0030554	adenyl nucleotide binding	5	No
GO:0001883	purine nucleoside binding	5	No
GO:0001882	nucleoside binding	5	No
GO:0046914	transition metal ion binding	4	No
GO:0032559	adenyl ribonucleotide binding	4	No
GO:0032555	purine ribonucleotide binding	4	No
GO:0032553	ribonucleotide binding	4	No
GO:0016491	oxidoreductase activity	4	No
GO:0008233	peptidase activity	3	No
GO:0019899	enzyme binding	3	No
GO:0005102	receptor binding	3	Yes
GO:0030528	transcription regulator activity	3	Yes
GO:0030234	enzyme regulator activity	3	No
GO:0070011	peptidase activity, acting on L-amino acid peptides	3	No
GO:0004866	endopeptidase inhibitor activity	2	No
GO:0060089	molecular transducer activity	2	No
GO:0003712	transcription cofactor activity	2	Yes
GO:0019900	kinase binding	2	Yes
GO:0019902	phosphatase binding	2	Yes
GO:0044212	transcription regulatory region DNA binding	2	Yes
GO:0016563	transcription activator activity	2	Yes
GO:0003676	nucleic acid binding	2	No
GO:0003677	DNA binding	2	Yes

Table A.2: GO Terms of families without interface variability

GO Number	GO Term	Family Pairs (max. 18)	Unique
GO:0043565	sequence-specific DNA binding	2	Yes
GO:0022891	substrate-specific transmembr. transp. act.	2	No
GO:0022892	substrate-specific transporter activity	2	No
GO:0051536	iron-sulfur cluster binding	2	No
GO:0015075	ion transmembrane transporter activity	2	No
GO:0046983	protein dimerization activity	2	No
GO:0004857	enzyme inhibitor activity	2	No
GO:0008237	metallopeptidase activity	2	No
GO:0042802	identical protein binding	2	No
GO:0005215	transporter activity	2	No
GO:0008134	transcription factor binding	2	Yes
GO:0008324	cation transmembrane transporter activity	2	No
GO:0048037	cofactor binding	2	Yes
GO:0030414	peptidase inhibitor activity	2	No
GO:0004175	endopeptidase activity	2	No
GO:0051540	metal cluster binding	2	No
GO:0003702	RNA polymerase II transcription factor activity	2	Yes
GO:0022857	transmembrane transporter activity	2	No

Table A.3: GO Terms of families without interface variability, continued

Author's Publication List

T. Goldberg[†], T. Hamp[†], and B. Rost. LocTree2 Predicts Localization for All Domains of Life. *Bioinformatics*, 28(18):i458–i465, 2012.

T. Hamp and B. Rost. Alternative Protein-Protein Interfaces are Frequent Exceptions. *PLoS Comput Biol*, 8(8):e1002623, 2012.

T. Hamp, F. Birzele, F. Buchwald, and S. Kramer. Improving Structure Alignment-based Prediction of SCOP Families Using Vorolign Kernels. *Bioinformatics*, 27 (2):204–10, 2011.

T. Hamp, T. Goldberg, and B. Rost. Accelerating the Original Profile Kernel. *PLoS One*, 8(6):e68459, 2013a.

T. Hamp, R. Kassner, S. Seemayer, E. Vicedo, C. Schaefer, D. Achten, F. Auer, A. Boehm, T. Braun, M. Hecht, M. Heron, P. Honigschmid, T. A. Hopf, S. Kaufmann, M. Kiening, D. Krompass, C. Landerer, Y. Mahlich, M. Roos, and B. Rost. Homology-based Inference Sets the Bar High for Protein Function Prediction. *BMC Bioinformatics*, 14 Suppl 3:S7, 2013b.

P. Radivojac, W. T. Clark, T. Hamp, B. Rost, et al. A Large-scale Evaluation of Computational Protein Function Prediction. *Nature Methods*, 10(3):221–7, 2013.

L. Zhao, S. C. H. Hoi, L. Wong, T. Hamp, and J. Li. Structural and Functional Analysis of Multi-Interface Domains. *PLoS ONE*, 7(12):e50821, 12 2012.

[†] Authors contributed equally

References

P. Aloy, H. Ceulemans, A. Stark, and R. B. Russell. The Relationship Between Sequence and Interaction Divergence in Proteins. *Journal of Molecular Biology*, 332(5):989 – 998, 2003.

S. F. Altschul, T. L. Madden, A. A. Schaeffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped Blast and PSI-Blast: A New Generation of Protein Database Search Programs. *Nucleic Acids Research*, 25:3389–3402, 1997.

M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, and G. Sherlock. Gene Ontology: Tool for the Unification of Biology. The Gene Ontology Consortium. *Nature Genetics*, 25(1):25–9, 2000.

A.-L. Barabasi and Z. N. Oltvai. Network Biology: Understanding the Cell's Functional Organization. *Nature Reviews Genetics*, 5(2):101–113, 2004.

C.B. Barber, K. Henrick, D.P. Dobkin, and H.T. Huhdanpaa. The Quickhull Algorithm for Convex Hulls. *ACM Transactions on Mathematical Software*, 22(4): 469–483, 1996.

I. Bera and S. Ray. A Study of Interface Roughness of Heteromeric Obligate and Non-obligate Protein-Protein Complexes. *Bioinformation*, 4(5):210–215, 2009.

H. Berman, K. Henrick, and H. Nakamura. Announcing the worldwide Protein Data Bank. *Nat Struct Biol*, 10(12):980, 2003.

H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The Protein Data Bank. *Nucleic Acids Research*, 28(1):235–242, 2000.

P. Block, J. Paern, E. Hüllermeier, P. Sanschagrin, C. A. Sotriffer, and G. Klebe. Physicochemical Descriptors to Discriminate Protein–Protein Interactions in Permanent and Transient Complexes Selected by Means of Machine Learning Algorithms. *Proteins: Structure, Function, Bioinformatics*, 65(3):607–622, 2006.

Y. Bromberg and B. Rost. SNAP: Predict Effect of Non-synonymous Polymorphisms on Function. *Nucleic Acids Research*, 35(11):3823–35, 2007.

C. J. C. Burges. A Tutorial on Support Vector Machines for Pattern Recognition. *Data Mining and Knowledge Discovery*, 2(2):121–167, 1998.

J. Chen, B. Aronow, and A. Jegga. Disease Candidate Gene Identification and Prioritization Using Protein Interaction Networks. *BMC Bioinformatics*, 10(1):73, 2009.

R Chen, L Li, and Z Weng. ZDOCK: An Initial-stage Protein-Docking Algorithm. *Proteins: Structure, Function, and Bioinformatics*, 52(1), 2003.

M. Chitale, T. Hawkins, C. Park, and D. Kihara. ESG: Extended Similarity Group Method for Automated Protein Function Prediction. *Bioinformatics*, 25(14): 1739–45, 2009.

Y. S. Choi, J.-S. Yang, Y. Choi, S. H. Ryu, and S. Kim. Evolutionary Conservation in Multiple Faces of Protein Interaction. *Proteins: Structure, Function, Bioinformatics*, 77(1):14–25, 2009.

W. T. Clark and P. Radivojac. Analysis of Protein Function and its Prediction from Amino Acid Sequence. *Proteins*, 79(7):2086–96, 2011.

Uniprot Consortium. Ongoing and Future Developments at the Universal Protein Resource. *Nucleic Acids Research*, 39(Database issue):D214–9, 2011.

A. D'Alessandro, P. G. Righetti, and L. Zolla. The Red Blood Cell Proteome and Interactome: An Update. *Journal of Proteome Research*, 9(1):144–163, 2010.

P. J. Day, A. Cleasby, I. J. Tickle, M. O'Reilly, J. E. Coyle, F. P. Holding, R. L. McMenamin, J. Yon, R. Chopra, C. Lengauer, and H. Jhoti. Crystal Structure of Human CDK4 in Complex with a D-type Cyclin. *Proceedings of the National Academy of Sciences U S A*, 106(11):4166–70, 2009.

C. Dessimoz, T. Gabaldon, D. S. Roos, E. L. Sonnhammer, J. Herrero, and Quest for Orthologs Consortium. Toward Community Standards in the Quest for Orthologs. *Bioinformatics*, 28(6):900–4, 2012.

P. Deutsch. Deflate Compressed Data Format Specification version 1.3, 1996.

A. K. Dunker and V. N. Uversky. Signal Transduction via Unstructured Protein Conduits. *Nature Chemical Biology*, 4(4):229–30, 2008.

S. Dutta and H. M. Berman. Large Macromolecular Complexes in the Protein Data Bank: A Status Report. *Structure*, 13(3):381–8, 2005.

S. R. Eddy. Accelerated Profile HMM Searches. *PLoS Computational Biology*, 7 (10):e1002195, 10 2011.

V. A. Eyrich, D. Przybylski, I. Y. Koh, O. Grana, F. Pazos, A. Valencia, and B. Rost. CAFASP3 in the Spotlight of EVA. *Proteins*, 53 Suppl 6:548–60, 2003.

S. Fields and O. Song. A Novel Genetic System to Detect Protein-protein Interactions. *Nature*, 340(6230):245–6, 1989.

J. Fox. *Applied Regression Analysis, Linear Models, and Related Methods*. SAGE Publications, February 1997. ISBN 080394540X.

E. Frank, M. Hall, L. Trigg, G. Holmes, and I. H. Witten. Data Mining in Bioinformatics Using Weka. *Bioinformatics*, 20(15):2479–81, 2004.

L. Fu, B. Niu, Z. Zhu, S. Wu, and W. Li. CD-HIT: Accelerated for Clustering the Next-generation Sequencing Data. *Bioinformatics*, 28(23):3150–2, 2012.

T. Gabaldon, C. Dessimoz, J. Huxley-Jones, A. J. Vilella, E. L. Sonnhammer, and S. Lewis. Joining Forces in the Quest for Orthologs. *Genome Biology*, 10(9):403, 2009.

J. Gailly and Mark Adler. zlib, 2012. URL `http://zlib.net`. http://zlib.net.

S. W. Ginzinger, C. X. Weichenberger, and M. J. Sippl. Detection of Unrealistic Molecular Environments in Protein Structures Based on Expected Electron Densities. *Journal of Biomolecular NMR*, 47(1):33–40, 2010.

S. W. Ginzinger, M. Gruber, H. Brandstetter, and M. J. Sippl. Real Space Refinement of Crystal Structures with Canonical Distributions of Electrons. *Structure*, 19(12): 1739–43, 2011.

C.-S. Goh, D. Milburn, and M. Gerstein. Conformational Changes Associated with Protein–Protein Interactions. *Current Opinion Structure Biology*, 14(1):104–109, 2004.

T. Goldberg, T. Hamp, and B. Rost. LocTree2 Predicts Localization for all Domains of Life. *Bioinformatics*, 28(18):i458–i465, 2012.

U. Gophna and Y. Ofran. Lateral Acquisition of Genes is Affected by the Friendliness of their Products. *Proceedings of the National Academy of Sciences U S A*, 108 (1):343–8, 2010.

J. J. Gray, S. Moughon, C. Wang, O. Schueler-Furman, B. Kuhlman, C. A. Rohl, and D. Baker. Protein–Protein Docking with Simultaneous Optimization of Rigid-body Displacement and Side-chain Conformations. *Journal of Molecular Biology*, 331 (1):281 – 299, 2003.

Gaël Guennebaud, Benoît Jacob, et al. Eigen v3. http://eigen.tuxfamily.org, 2010.

J. Haas, S. Roth, K. Arnold, F. Kiefer, T. Schmidt, L. Bordoli, and T. Schwede. The Protein Model Portal–A Comprehensive Resource for Protein Structure and Model Information. *Database (Oxford)*, 2013:bat031, 2013.

A. Hamosh, A. F. Scott, J. S. Amberger, C. A. Bocchini, and V. A. McKusick. Online Mendelian Inheritance in Man (OMIM), a Knowledgebase of Human Genes and Genetic Disorders. *Nucleic Acids Research*, 33(suppl 1):D514–D517, 2005.

T. Hamp and B. Rost. Alternative Protein-Protein Interfaces are Frequent Exceptions. *PLoS Comput Biol*, 8(8):e1002623, 2012.

T. Hamp, F. Birzele, F. Buchwald, and S. Kramer. Improving Structure Alignment-based Prediction of SCOP Families Using Vorolign Kernels. *Bioinformatics*, 27 (2):204–10, 2011.

T. Hamp, T. Goldberg, and B. Rost. Accelerating the Original Profile Kernel. *PLoS One*, 8(6):e68459, 2013a.

T. Hamp, R. Kassner, S. Seemayer, E. Vicedo, C. Schaefer, D. Achten, F. Auer, A. Boehm, T. Braun, M. Hecht, M. Heron, P. Honigschmid, T. A. Hopf, S. Kaufmann, M. Kiening, D. Krompass, C. Landerer, Y. Mahlich, M. Roos, and B. Rost. Homology-based Inference Sets the Bar High for Protein Function Prediction. *BMC Bioinformatics*, 14 Suppl 3:S7, 2013b.

G. T. Hart, A. K. Ramani, and E. M. Marcotte. How Complete are Current Yeast and Human Protein-interaction Networks? *Genome Biology*, 7(11):120, 2006.

T. Hawkins, S. Luban, and D. Kihara. Enhanced Automated Function Prediction Using Distantly Related Sequences and Contextual Association by PFP. *Protein Science*, 15(6):1550–6, 2006.

K. Henrick and J. M. Thornton. PQS: A Protein Quaternary Structure File Server. *Trends in Biochemical Science*, 23(9):358–61, 1998.

H. Huang, B. M. Jedynak, and J. S. Bader. Where Have All the Interactions Gone? Estimating the Coverage of Two-hybrid Protein Interaction Maps. *PLoS Computational Biology*, 3(11):e214, 2007.

I. Ispolatov, A. Yuryev, I. Mazo, and S. Maslov. Binding Properties and Evolution of Homodimers in Protein-Protein Interaction Networks. *Nucleic Acids Research*, 33 (11):3629–3635, 2005.

T. Ito, T. Chiba, R. Ozawa, M. Yoshida, M. Hattori, and Y. Sakaki. A Comprehensive Two-hybrid Analysis to Explore the Yeast Protein Interactome. *Proceedings of the National Academy of Sciences U S A*, 98(8):4569–4574, 2001.

C. E. Jones, J. Schwerdt, T. A. Bretag, U. Baumann, and A. L. Brown. GOSLING: A Rule-based Protein Annotator Using BLAST and GO. *Bioinformatics*, 24(22): 2628–9, 2008.

A. Kalsotra and T. A. Cooper. Functional Consequences of Developmentally Regulated Alternative Splicing. *Nature Reviews Genetics*, 12:715–729, 2011.

S. A. Kang and B. R. Crane. Effects of Interface Mutations on Association Modes and Electron-Transfer Rates Between Proteins. *Proceedings of the National Academy of Sciences U S A*, 102(43):15465–70, 2005.

P. L. Kastritis and A. M. J. J. Bonvin. Are Scoring Functions in Protein-Protein Docking Ready to Predict Interactomes? Clues from a Novel Binding Affinity Benchmark. *Journal of Proteome Research*, 9(5):2216–2225, 2010.

P. L. Kastritis, I. H. Moal, H. Hwang, Z. Weng, P. A. Bates, A. M. J. J. Bonvin, and J. Janin. A Structure-based Benchmark for Protein–Protein Binding Affinity. *Protein Science*, 20(3):482–491, 2011.

J. C. Kendrew, R. E. Dickerson, B. E. Strandberg, R. J. Hart, D. R. Davies, and D. C. Phillips. Structure of Myoglobin: A Three-Dimensional Fourier Synthesis at 2 Å Resolution. *Nature*, 185:422–427, 1960.

O. Keskin, A. Gursoy, B. Ma, and R. Nussinov. Principles of Protein-Protein Interactions: What are the Preferred Ways for Proteins to Interact? *Chem Rev*, 108(4):1225–1244, 2008.

W. K. Kim, A. Henschel, C. Winter, and M. Schroeder. The Many Faces of Protein-Protein Interactions: A Compendium of Interface Geometry. *PLoS Computational Biology*, 2(9):e124, 2006.

I. Y. Koh, V. A. Eyrich, M. A. Marti-Renom, D. Przybylski, M. S. Madhusudhan, N. Eswar, O. Grana, F. Pazos, A. Valencia, A. Sali, and B. Rost. EVA: Evaluation of Protein Structure Prediction Servers. *Nucleic Acids Research*, 31(13):3311–5, 2003.

D. Korkin, F. P. Davis, and A. Sali. Localization of Protein-Binding Sites Within Families of Proteins. *Protein Science*, 14(9):2350–60, 2005.

E. Krissinel and K. Henrick. Detection of Protein Assemblies in Crystals. *Lecture Notes in Computer Science*, 3695:163–174, 2005.

E. Krissinel and K. Henrick. Inference of Macromolecular Assemblies from Crystalline State. *Journal of Molecular Biology*, 372(3):774–97, 2007.

R. Kuang, E. Ie, K. Wang, M. Siddiqi, Y. Freund, and C. Leslie. Profile-based String Kernels for Remote Homology Detection and Motif Extraction. *Proceedings of the IEEE Computational System Bioinformatics Conference*, pages 152–60, 2004.

J. G. Lees, J. K. Heriche, I. Morilla, J. A. Ranea, and C. A. Orengo. Systematic Computational Prediction of Protein Interaction Networks. *Physical Biology*, 8 (3):035008, 2011.

M. F. Lensink, R. Mendez, and S. J. Wodak. Docking and Scoring Protein Complexes: CAPRI 3rd Edition. *Proteins: Structure, Function, and Bioinformatics*, 69(4): 704–718, 2007.

E. D. Levy. PiQSi: Protein Quaternary Structure Investigation. *Structure*, 15(11): 1364–1367, 2007.

E. D. Levy, E. B. Erba, C. V. Robinson, and S. A. Teichmann. Assembly Reflects Evolution of Protein Complexes. *Nature*, 453(7199):1262–1265, 2008.

B. Liu, X. Wang, L. Lin, and Q. Dong. A Discriminative Method for Protein Remote Homology Detection and Fold Recognition Combining Top-n-grams and Latent Semantic Analysis. *BMC Bioinformatics*, 9:510, 2008a.

J. Liu and B. Rost. Domains, motifs and clusters in the protein universe. *Current Opinion in Chemical Biology*, 7(1):5–11, 2003.

J. Liu and B. Rost. Sequence-based prediction of protein domains. *Nucleic Acids Research*, 32(12):3522–30, 2004.

Y. Liu, I. Kim, and H. Zhao. Protein Interaction Predictions from Diverse Sources. *Drug Discovery Today*, 13(9–10):409 – 416, 2008b.

E. Lorentzen, A. Dziembowski, D. Lindner, B. Seraphin, and E. Conti. Rna Channelling by the Archaeal Exosome. *EMBO Reports*, 8(5):470–6, 2007.

D. B. Lukatsky, B. E. Shakhnovich, J. Mintseris, and E. I. Shakhnovich. Structural Similarity Enhances Interaction Propensity of Proteins. *Journal of Moleular Biology*, 365(5):1596–1606, 2007.

H. Madaoui and R. Guerois. Coevolution at Protein Complex Interfaces can be Detected by the Complementarity Trace with Important Impact for Predictive Docking. *Proceedings of the National Academy of Sciences U S A*, 105(22): 7708–7713, 2008.

S. Maere, K. Heymans, and M. Kuiper. BiNGO: A Cytoscape Plugin to Assess Overrepresentation of Gene Ontology Categories in Biological Networks. *Bioinformatics*, 21(16):3448–3449, 2005.

M. Man-Wai. PairProSVM: Protein Subcellular Localization Based on Local Pairwise Profile Alignment and SVM. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 5(3):416–422, 2008.

S. M. Margarit, H. Sondermann, B. E. Hall, B. Nagar, A. Hoelz, M. Pirruccello, D. Bar-Sagi, and J. Kuriyan. Structural Evidence for Feedback Activation by Ras.GTP of the Ras-specific Nucleotide Exchange Factor SOS. *Cell*, 112(5): 685–95, 2003.

D. S. Marks, L. J. Colwell, R. Sheridan, T. A. Hopf, A. Pagnani, R. Zecchina, and C. Sander. Protein 3D Structure Computed from Evolutionary Sequence Variation. *PLoS ONE*, 6(12):e28766, 12 2011.

M. A. Marti-Renom, M. S. Madhusudhan, A. Fiser, B. Rost, and A. Sali. Reliability of Assessment of Protein Structure Prediction Methods. *Structure*, 10(3):435–40, 2002.

D. M. Martin, M. Berriman, and G. J. Barton. GOtcha: A New Method for Prediction of Protein Function Assessed by the Annotation of Seven Genomes. *BMC Bioinformatics*, 5:178, 2004.

S. Martin, D. Roe, and J. L. Faulon. Predicting Protein-protein Interactions Using Signature Products. *Bioinformatics*, 21(2):218–26, 2005.

E. Mashiach, R. Nussinov, and H. J. Wolfson. FiberDock: Flexible Induced-fit Backbone Refinement in Molecular Docking. *Proteins: Structure, Function, and Bioinformatics*, 78(6), 2010.

F. Meissner and M. Mann. Quantitative Shotgun Proteomics: Considerations for a High-quality Workflow in Immunology. *Nature Immunology*, 15(2):112–7, 2014.

I. Melvin, E. Ie, R. Kuang, J. Weston, W. N. Stafford, and C. Leslie. SVM-Fold: A Tool for Discriminative Multi-class Protein Fold and Superfamily Recognition. *BMC Bioinformatics*, 8 Suppl 4:S2, 2007.

S. Mika and B. Rost. Protein–protein Interactions More Conserved Within Species than Across Species. *PLoS Computational Biology*, 2(7):e79, 2006.

K. Ming and I. Witten. Issues in Stacked Generalization. *Journal of Artificial Intelligence Research*, 10:271–280, 1999.

J. Mintseris and Z. Weng. Atomic Contact Vector in Protein-Protein Recognition. *Proteins: Structure, Function, Bioinformatics*, 53(3):629–639, 2003.

J. Mintseris and Z. Weng. Structure, Function, and Evolution of Transient and Obligate Protein-Protein Interactions. *Proceedings of the National Academy of Sciences U S A*, 102(31):10930–10935, 2005.

R. Mosca, T. Pons, A. Ceol, A. Valencia, and P. Aloy. Towards a Detailed Atlas of Protein-protein Interactions. *Current Opinion Structural Biology*, 2013.

A. G. Murzin, S. E. Brenner, T. Hubbard, and C. Chothia. SCOP: A Structural Classification of Proteins Database for the Investigation of Sequences and Structures. *Journal of Molecular Biology*, 247(4):536–40, 1995.

N. L. Nehrt, W. T. Clark, P. Radivojac, and M. W. Hahn. Testing the Ortholog Conjecture with Comparative Functional Genomic Data from Mammals. *PLoS Computational Biology*, 7(6):e1002073, 2011.

S. Nissen. Implementation of a fast artificial neural network library (fann). Technical report, Department of Computer Science University of Copenhagen (DIKU), 2003. http://fann.sf.net.

Y. Ofran and B. Rost. Predicted Protein-protein Interaction Sites from Local Sequence Information. *FEBS Lett*, 544(1-3):236–9, 2003a.

Y. Ofran and B. Rost. Analysing Six Types of Protein-Protein Interfaces. *Journal of Molecular Biology*, 325(2):377–387, 2003b.

Y. Ofran and B. Rost. ISIS: Interaction Sites Identified from Sequence. *Bioinformatics*, 23(2):e13–16, 2007a.

Y. Ofran and B. Rost. Protein-Protein Interaction Hot Spots Carved Into Sequences. *PLoS Comput Biol*, 3(7):e119, 2007b.

Y. Ofran, G. Yachdav, E. Mozes, T. Soong, R. Nair, and B. Rost. Create and Assess Protein Networks Through Molecular Characteristics of Individual Proteins. *Bioinformatics*, 22(14):e402–e407, 2006.

Y. Ofran, V. Mysore, and B. Rost. Prediction of DNA-binding Residues from Sequence. *Bioinformatics*, 23(13):i347–53, 2007.

C. A. Orengo, A. D. Michie, S. Jones, D. T. Jones, M. B. Swindells, and J. M. Thornton. CATH–A Hierarchic Classification of Protein Domain Structures. *Structure*, 5(8):1093–108, 1997.

S. Y. Park, B. D. Beel, M. I. Simon, A. M. Bilwes, and B. R. Crane. In Different Organisms, the Mode of Interaction Between Two Signaling Proteins is not Necessarily Conserved. *Proceedings of the National Academy of Sciences U S A*, 101(32):11646–51, 2004.

Y. Park and E. M. Marcotte. Flaws in Evaluation Schemes for Pair-input Computational Predictions. *Nature Methods*, 9:1134–1136, 2012.

M. F. Perutz, M. G. Rossmann, A. F. Cullis, G. Muirhead, G. Will, and A. T. North. Structure of Haemoglobin: A Three-Dimensional Fourier Synthesis at 5.5 Å Resolution, Obtained by X-ray Analysis. *Nature*, 185:416–422, 1960.

S. Pitre, C. North, M. Alamgir, M. Jessulat, A. Chan, X. Luo, J. R. Green, M. Dumontier, F. Dehne, and A. Golshani. Global Investigation of Protein-protein Interactions in Yeast Saccharomyces Cerevisiae Using Re-occurring Short Polypeptide Sequences. *Nucleic Acids Research*, 36(13):4286–94, 2008.

J. C. Platt. Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines. Technical report, ADVANCES IN KERNEL METHODS - SUPPORT VECTOR LEARNING, 1998.

M. Punta, P. C. Coggill, R. Y. Eberhardt, J. Mistry, J. Tate, C. Boursnell, N. Pang, K. Forslund, G. Ceric, J. Clements, A. Heger, L. Holm, E. L. Sonnhammer, S. R.

Eddy, A. Bateman, and R. D. Finn. The Pfam Protein Families Database. *Nucleic Acids Research*, 40(Database issue):D290–301, 2012.

P. Radivojac, W. T. Clark, T. Hamp, B. Rost, et al. A Large-scale Evaluation of Computational Protein Function Prediction. *Nature Methods*, 10(3):221–7, 2013.

J. A. Ranea, I. Morilla, J. G. Lees, A. J. Reid, C. Yeats, A. B. Clegg, F. Sanchez-Jimenez, and C. Orengo. Finding the "Dark Matter" in Human and Yeast Protein Network Prediction and Modelling. *PLoS Computational Biology*, 6(9), 2010.

H. Rangwala and G. Karypis. Profile-based Direct Kernels for Remote Homology Detection and Fold Recognition. *Bioinformatics*, 21(23):4239–47, 2005a.

H. Rangwala and G. Karypis. Profile-based Direct Kernels for Remote Homology Detection and Fold Recognition. *Bioinformatics*, 21(23):4239–47, 2005b.

M. Remmert, A. Biegert, A. Hauser, and J. Soding. HHblits: Lightning-fast Iterative Protein Sequence Searching by HMM-HMM Alignment. *Nature Methods*, 9(2): 173–5, 2012.

R. Rentzsch and C. A. Orengo. Protein Function Prediction–the Power of Multiplicity. *Trends in Biotechnology*, 27(4):210–9, 2009.

I. Res, I. Mihalek, and O. Lichtarge. An Evolution Based Classifier for Prediction of Protein Interfaces without using Protein Structures. *Bioinformatics*, 21(10): 2496–501, 2005.

P. W. Rose, B. Beran, C. Bi, W. F. Bluhm, D. Dimitropoulos, D. S. Goodsell, A. Prlic, M. Quesada, G. B. Quinn, J. D. Westbrook, J. Young, B. Yukich, C. Zardecki, H. M. Berman, and P. E. Bourne. The RCSB Protein Data Bank: Redesigned Web Site and Web Services. *Nucleic Acids Res*, 39(Database issue):D392–401, 2010.

B. Rost. Twilight Zone of Protein Sequence Alignments. *Protein Engineering*, 12 (2):85–94, 1999.

B. Rost and J. Liu. The PredictProtein Server. *Nucleic Acids Research*, 31(13): 3300–4, 2003.

B. Rost and C. Sander. Improved Prediction of Protein Secondary Structure by Use of Sequence Profiles and Neural Networks. *Proceedings of the National Academy of Sciences U S A*, 90(16):7558–7562, 1993.

L. Sambourg and N. Thierry-Mieg. New Insights into Protein-protein Interaction Data Lead to Increased Estimates of the S. Cerevisiae Interactome Size. *BMC Bioinformatics*, 11:605, 2010.

M. H. Schaefer, J. F. Fontaine, A. Vinayagam, P. Porras, E. E. Wanker, and M. A. Andrade-Navarro. HIPPIE: Integrating Protein Interaction Networks with Experiment Based Quality Scores. *PLoS One*, 7(2):e31826, 2012.

A. Schlessinger, M. Punta, G. Yachdav, L. Kajan, and B. Rost. Improved Disorder Prediction by Combination of Orthogonal Approaches. *PLoS ONE*, 4(2), 02 2009.

B. Schölkopf and A. J. Smola. *Learning With Kernels Support Vector Machines, Regularization, Optimization, and Beyond*. Adaptive computation and machine learning. MIT, Cambridge, Massachusetts, 2002.

T. M. Schmeing and V. Ramakrishnan. What Recent Ribosome Structures Have Revealed About the Mechanism of Translation. *Nature*, 461:1234–1242, 2009.

M. Schneider, L. Lane, E. Boutet, D. Lieberherr, M. Tognolli, L. Bougueleret, and A. Bairoch. The UniprotKB/Swiss-Prot Knowledgebase and its Plant Proteome Annotation Program. *J Proteomics*, 72(3):567–73, 2009.

M. S. Scott and G. J. Barton. Probabilistic Prediction and Ranking of Human Protein-Protein Interactions. *BMC Bioinformatics*, 8:239, 2007.

P. Shannon, A. Markiel, O. Ozier, N.S. Baliga, J. T. Wang, D. Ramage, N. Amin, B. Schwikowski, and T. Ideker. Cytoscape: A Software Environment for Integrated Models of Biomolecular Interaction Networks. *Genome Research*, 13(11):2498–2504, 2003.

C. J. Shin, M. J. Davis, and M. A. Ragan. Towards the Mammalian Interactome: Inference of a Core Mammalian Interaction Set in Mouse. *Proteomics*, 9(23): 5256–66, 2009.

B. A. Shoemaker and A. R. Panchenko. Deciphering Protein–protein Interactions. Part i. Experimental Techniques and Databases. *PLoS Computational Biology*, 3 (3):e42, 2007.

B. A. Shoemaker, A. R. Panchenko, and S. H. Bryant. Finding Biologically Relevant Protein Domain Interactions: Conserved Binding Mode Analysis. *Protein Science*, 15(2):352–61, 2006.

A. Sokolov and A. Ben-Hur. Hierarchical Classification of Gene Ontology Terms Using the GOstruct Method. *Journal of Bioinformatics and Computational Biology*, 8(2):357–76, 2010.

T. Soong, K. O. Wrzeszczynski, and B. Rost. Physical Protein–protein Interactions Predicted from Microarrays. *Bioinformatics*, 24(22):2608–2614, 2008.

A. Stein, A. Panjkovich, and P. Aloy. 3did Update: Domain-Domain and Peptide-mediated Interactions of Known 3d Structure. *Nucleic Acids Research*, 37(suppl 1):D300–D304, 2009.

M. P. Stumpf, T. Thorne, E. de Silva, R. Stewart, H. J. An, M. Lappe, and C. Wiuf. Estimating the Size of the Human Interactome. *Proceedings of the National Academy of Sciences U S A*, 105(19):6959–64, 2008.

B. Stynen, H. Tournu, J. Tavernier, and P. Van Dijck. Diversity in Genetic in Vivo Methods for Protein-protein Interaction Studies: From the Yeast Two-hybrid System to the Mammalian Split-luciferase System. *Microbiology and Molecular Biology Reviews*, 76(2):331–82, 2012.

P.D. Sulatycke and K. Ghose. Caching-Efficient Multithreaded Fast Multiplication of Sparse Matrices. In *Parallel Processing Symposium, 1998. IPPS/SPDP 1998. Proceedings of the First Merged International ... and Symposium on Parallel and Distributed Processing 1998*, pages 117–123, 1998.

S. R. Sunyaev, F. Eisenhaber, I. V. Rodchenkov, B. Eisenhaber, V. G. Tumanyan, and E. N. Kuznetsov. PSIC: Profile Extraction from Sequence Alignments with Position-specific Counts of Independent Observations. *Protein Engineering*, 12 (5):387–394, 1999.

D Talavera, D. L. Robertson, and S. C. Lovell. Alternative Splicing and Protein Interaction Data Sets. *Nature Biotechnology*, 31:292–293, 2013.

I. W. Taylor, R. Linding, D. Warde-Farley, Y. M. Liu, C. Pesquita, D. Faria, S. Bull, T. Pawson, Q. Morris, and J. L. Wrana. Dynamic Modularity in Protein Interaction Networks Predicts Breast Cancer Outcome. *Nature Biotechnology*, 27(2):199–204, 2009.

J. Teyra and M. T. Pisabarro. Characterization of Interfacial Solvent in Protein Complexes and Contribution of Wet Spots to the Interface Description. *Proteins: Structure, Function, Bioinformatics*, 67(4):1087–1095, 2007.

F. A. Tezcan, J. T. Kaiser, D. Mustafi, M. Y. Walton, J. B. Howard, and D. C. Rees. Nitrogenase Complexes: Multiple Docking Sites for a Nucleotide Switch Protein. *Science*, 309(5739):1377–80, 2005.

N. Thanh and K. Rui. Partial Profile Alignment Kernels for Protein Classification. In *Genomic Signal Processing and Statistics, 2009. GENSIPS 2009. IEEE International Workshop on*, pages 1–4, 2009.

N. C. Toussaint, C. Widmer, O. Kohlbacher, and G. Ratsch. Exploiting Physicochemical Properties in String Kernels. *BMC Bioinformatics*, 11 Suppl 8:S7, 2010.

O. G. Troyanskaya, K. Dolinski, A. B. Owen, R. B. Altman, and D. Botstein. A Bayesian Framework for Combining Heterogeneous Data Sources for Gene Function Prediction (in Saccharomyces Cerevisiae). *Proceedings of the National Academy of Sciences U S A*, 100(14):8348–8353, 2003.

N. Tuncbag, A. Gursoy, E. Guney, R. Nussinov, and O. Keskin. Architectures and Functional Coverage of Protein-Protein Interfaces. *Journal of Molecular Biology*, 381(3):785–802, 2008.

P. Uetz, L. Giot, G. Cagney, T. A. Mansfield, R. S. Judson, J. R. Knight, D. Lockshon, V. Narayan, M. Srinivasan, P. Pochart, A. Qureshi-Emili, Y. Li, B. Godwin, D. Conover, T. Kalbfleisch, G. Vijayadamodar, M. Yang, M. Johnston, S. Fields, and J. M. Rothberg. A Comprehensive Analysis of Protein-protein Interactions in Saccharomyces Cerevisiae. *Nature*, 403(6770):623–627, 2000.

P. Uetz, Y. A. Dong, C. Zeretzke, C. Atzler, A. Baiker, B. Berger, S. V. Rajagopala, M. Roupelieva, D. Rose, E. Fossum, and J. Haas. Herpesviral Protein Networks and Their Interaction with the Human Proteome. *Science*, 311(5758):239–42, 2006.

K. Venkatesan, J. F. Rual, A. Vazquez, U. Stelzl, I. Lemmens, T. Hirozane-Kishikawa, T. Hao, M. Zenkner, X. Xin, K. I. Goh, M. A. Yildirim, N. Simonis, K. Heinzmann, F. Gebreab, J. M. Sahalie, S. Cevik, C. Simon, A. S. de Smet, E. Dann, A. Smolyar, A. Vinayagam, H. Yu, D. Szeto, H. Borick, A. Dricot, N. Klitgord, R. R. Murray, C. Lin, M. Lalowski, J. Timm, K. Rau, C. Boone, P. Braun, M. E. Cusick, F. P. Roth, D. E. Hill, J. Tavernier, E. E. Wanker, A. L. Barabasi, and M. Vidal. An Empirical Framework for Binary Interactome Mapping. *Nature Methods*, 6(1): 83–90, 2009.

A. J. Walhout, R. Sordella, X. Lu, J. L. Hartley, G. F. Temple, M. A. Brasch, N. Thierry-Mieg, and M. Vidal. Protein Interaction Mapping in C. elegans Using Proteins Involved in Vulval Development. *Science*, 287(5450):116–22, 2000.

M. N. Wass and M. J. Sternberg. ConFunc–Functional Annotation in the Twilight Zone. *Bioinformatics*, 24(6):798–806, 2008.

M. W. Weigt, R. A. White, H. Szurman, J. A. Hoch, and T. Hwa. Identification of Direct Residue Contacts in Protein-protein Interaction by Message Passing. *Proceedings of the National Academy of Sciences U S A*, 106(1):67–72, 2008.

J. Weston, C. Leslie, E. Ie, D. Zhou, A. Elisseeff, and W. S. Noble. Semi-supervised Protein Classification Using Cluster Kernels. *Bioinformatics*, 21(15):3241–7, 2005.

C. Winter, A. Henschel, W. K. Kim, and M. Schroeder. SCOPPI: A Structural Classification of Protein–Protein Interfaces. *Nucleic Acids Research*, 34(suppl 1): D310–D314, 2006.

M. W. Wojtowicz, J. J. Flanagan, S. S. Millard, S. L. Zipursky, and J. C. Clemens. Alternative Splicing of Drosophila Dscam Generates Axon Guidance Receptors That Exhibit Isoform-Specific Homophilic Binding. *Cell*, 118(5):619 – 633, 2004.

J. Wu, T. Vallenius, K. Ovaska, J. Westermarck, T. P. Makela, and S. Hautaniemi. Integrated Network Analysis Platform for Protein-protein Interactions. *Nature Methods*, 6(1):75–7, 2009.

J. Xu and Y. Li. Discovering Disease-Genes by Topological Features in Human Protein-Protein Interaction Network. *Bioinformatics*, 22(22):2800–2805, 2006.

H. Yu, N. M. Luscombe, H. X. Lu, X. Zhu, Y. Xia, J. D. Han, N. Bertin, S. Chung, M. Vidal, and M. Gerstein. Annotation Transfer Between Genomes: Protein-protein Interologs and Protein-DNA Regulogs. *Genome Research*, 14(6):1107–18, 2004.

H. Yu, L. Tardivo, S. Tam, E. Weiner, F. Gebreab, C. Fan, N. Svrzikapa, T. Hirozane-Kishikawa, E. Rietman, X. Yang, J. Sahalie, K. Salehi-Ashtiani, T. Hao, M. E. Cusick, D. E. Hill, F. P. Roth, P. Braun, and M. Vidal. Next-generation Sequencing to Generate Interactome Datasets. *Nature Methods*, 8(6):478–80, 2011.

A. V. Zavialov, J. Berglund, A. F. Pudney, L. J. Fooks, T. M. Ibrahim, S. MacIntyre, and S. D. Knight. Structure and Biogenesis of the Capsular F1 Antigen from Yersinia pestis: Preserved Folding Energy Drives Fiber Formation. *Cell*, 113(5): 587–96, 2003.

Q. C. Zhang, D. Petrey, R. Norel, and B. H. Honig. Protein Interface Conservation Across Structure Space. *Proceedings of the National Academy of Sciences U S A*, 107(24):10896–901, 2010.

L. Zhao, S. C. H. Hoi, L. Wong, T. Hamp, and J. Li. Structural and Functional Analysis of Multi-Interface Domains. *PLoS ONE*, 7(12):e50821, 12 2012.