Dissertation

Large-scale Automated Protein Function Prediction

Submitted by

Indika Kahanda

Department of Computer Science

In partial fulfillment of the requirements

For the Degree of Doctor of Philosophy

Colorado State University

Fort Collins, Colorado

Summer 2016

Doctoral Committee:

Advisor: Asa Ben-Hur

Chuck Anderson
Bruce Draper
Wen Zhou

Abstract

Large-scale Automated Protein Function Prediction

Proteins are the workhorses of life, and identifying their functions is a very important biological problem. The function of a protein can be loosely defined as everything it performs or happens to it. The Gene Ontology (GO) is a structured vocabulary which captures protein function in a hierarchical manner and contains thousands of terms. Through various wet-lab experiments over the years scientists have been able to annotate a large number of proteins with GO categories which reflect their functionality. However, experimentally determining protein functions is a highly resource-intensive task, and a large fraction of proteins remain un-annotated. Recently a plethora automated methods have emerged and their reasonable success in computationally determining the functions of proteins using a variety of data sources by sequence/structure similarity or using various biological network data, has led to establishing automated function prediction (AFP) as an important problem in bioinformatics.

In a typical machine learning problem, cross-validation is the protocol of choice for evaluating the accuracy of a classifier. But, due to the process of accumulation of annotations over time, we identify the AFP as a combination of two sub-tasks: making predictions on annotated proteins and making predictions on previously unannotated proteins. In our first project, we analyze the performance of several protein function prediction methods in these two scenarios. Our results show that GOstruct, an AFP method that our lab has previously developed, and two other popular methods: binary SVMs and guilt by association, find it hard to achieve the same level of accuracy on these two tasks compared to the performance

evaluated through cross-validation, and that predicting novel annotations for previously annotated proteins is a harder problem than predicting annotations for uncharacterized proteins. We develop GOstruct 2.0 by proposing improvements which allows the model to make use of information of a protein's current annotations to better handle the task of predicting novel annotations for previously annotated proteins. Experimental results on yeast and human data show that GOstruct 2.0 outperforms the original GOstruct, demonstrating the effectiveness of the proposed improvements.

Although the biomedical literature is a very informative resource for identifying protein function, most AFP methods do not take advantage of the large amount of information contained in it. In our second project, we conduct the first ever comprehensive evaluation on the effectiveness of literature data for AFP. Specifically, we extract co-mentions of protein-GO term pairs and bag-of-words features from the literature and explore their effectiveness in predicting protein function. Our results show that literature features are very informative of protein function but with further room for improvement. In order to improve the quality of automatically extracted co-mentions, we formulate the classification of co-mentions as a supervised learning problem and propose a novel method based on graph kernels. Experimental results indicate the feasibility of using this co-mention classifier as a complementary method that aids the bio-curators who are responsible for maintaining databases such as Gene Ontology. This is the first study of the problem of protein-function relation extraction from biomedical text.

The recently developed human phenotype ontology (HPO), which is very similar to GO, is a standardized vocabulary for describing the phenotype abnormalities associated with human diseases. At present, only a small fraction of human protein coding genes have

HPO annotations. But, researchers believe that a large portion of currently unannotated genes are related to disease phenotypes. Therefore, it is important to predict gene-HPO term associations using accurate computational methods. In our third project, we introduce PHENOstruct, a computational method that directly predicts the set of HPO terms for a given gene. We compare PHENOstruct with several baseline methods and show that it outperforms them in every respect. Furthermore, we highlight a collection of informative data sources suitable for the problem of predicting gene-HPO associations, including large scale literature mining data.

Acknowledgements

First, I would like to convey my gratitude to my adviser, Dr. Asa Ben-Hur, for providing me invaluable guidance in my research as well as publishing. I would like to thank my committee members Dr. Chuck Anderson, Dr. Bruce Draper and Dr. Wen Zhou for providing valuable advice throughout different stages of my research.

I would like to thank my research collaborators Dr. Karin Verspoor and Chris Funk at University of Colorado, Denver. I would also like to thank all the members of the Bioinformatics group at Colorado State University for providing various support throughout the years.

The work described in this dissertation was supported by the National Science Foundation (NSF) Advances in Biological Informatics program (award number 0965768) and the Critical Assessment of protein Function Annotation (CAFA) Travel Awards program. I am thankful to the National Science Foundation (NSF) and the Automated Function Prediction Special Interest Group (AFP/SIG) for providing me with financial support.

I would like to thank my parents, G.L.S.C.K.S. Kahanda and G.R.M. Gamlath, my younger sister D.S. Kahanda for always being there for me throughout my academic life. Last but not least, I thank my wife, Upulee Kanewala for her continuous support and encouragement in my research work.

Dedication

Dedicated to my loving parents and my loving wife.

Table of Contents

List of Tables

List of Figures

CHAPTER 1

Introduction

1.1. Functions of Proteins

Proteins are key players in the game of life, and recognizing the functions that they are responsible for is a very important problem in biology. These macromolecules make up half of the dry weight of cells and are major contributors to performing many activities in cells [2]. For example, *keratins* provide structural support which protects cells from damage or stress [3] while *immunoglobulins* play a key role in antibacterial and antiviral defense [4]. *Ferritin* supports iron storage and release [5] and *hemoglobins* facilitate the transport of oxygen in red blood cells [6].

However, the definition of *function* is ambiguous because it usually consists of multiple aspects. For example, a protein may be involved in a specific action such as an enzymatic activity (i.e., biochemical aspect) or be part of a signalling pathway that carries out a certain high level cellular process such as cell death (i.e., physiological aspect) or cause a certain disease due to a mutation in its sequence (i.e., phenotypic or medical aspect). Another fact that adds to this confusion is the fact that two proteins may have the same enzymatic activity but they may be involved in completely different biological processes. On the other hand, it is quite possible for two proteins that are part of the same biological process to have completely different functions at the molecular level. Because of all these factors, standardized vocabularies are of utmost importance for facilitating an unambiguous discussion of protein function.

1.2. Bio-Ontologies

The Gene Ontology (GO) [7] and the Human Phenotype Ontology (HPO) [8] are two structured vocabularies which describe different aspects of protein function in a hierarchical manner. Both of these ontologies are structured as directed acyclic graphs (DAGs). The Gene Ontology, which is the more mature of the two, is composed of three independent subontologies: molecular functions of proteins, biological processes they are part of and cellular components in which these occur. For example, GO consists of categories such as protein binding, RNA binding (molecular function), double-strand break repair, chromosome segregation (biological process) and nucleus, cytoplasm (cellular component). Figure 1.1 depicts part of the GO DAG for the biological process subontology category "cellular nitrogen compound metabolic process."

The Human Phenotype Ontology, which was developed more recently than the Gene Ontology, captures the phenotype aspect of proteins. In the medical context a phenotype is defined as a deviation from the normal morphology, physiology, or behavior [9]. For example, runny nose is a phenotype (i.e., symptom) associated with the disease common cold. HPO is composed of the three subontologies: organ abnormality, mode of inheritance, and onset and clinical course (see Fig. 1.2).

Organ abnormality is the main subontology which describes clinical abnormalities with categories such as "hearing abnormality" and "abnormality of eye". The onset and clinical course subontology describes the typical time of onset of clinical symptoms and their speed of progression. It consists of categories such as "neonatal onset", "childhood onset", and "adult onset" for describing the age of onset, while the categories "slow progression" and

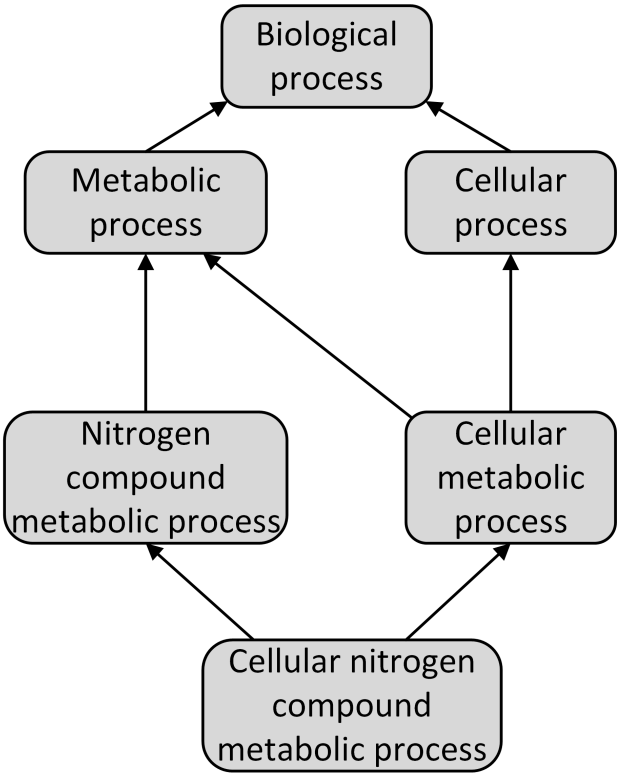

FIGURE 1.1. Part of the GO DAG showing all ancestors of the biological process category "cellular nitrogen compound metabolic process". Arrows indicate 'is-a' relationships.

"rapidly progressive" describe the pace of progression of the phenotypes. The mode of inheritance subontology describes the inheritance patterns of the phenotypes such as "autosomal dominant" (i.e., conditions which are expressed in individuals who have inherited just one copy of the mutated sequence from their parents).

Both the Gene Ontology and Human Phenotype Ontology are composed of thousands of categories. Experimentally determining the set of GO/HPO categories associated with a protein is a highly resource-intensive task [10]. However over the years, scientists have been able to annotate a large number of proteins with GO and HPO categories which reflect their functionality through various wet-lab experiments and clinical diagnosis. Currently there are more than 100,000 GO and 50,000 HPO annotations for a large number of proteins. But

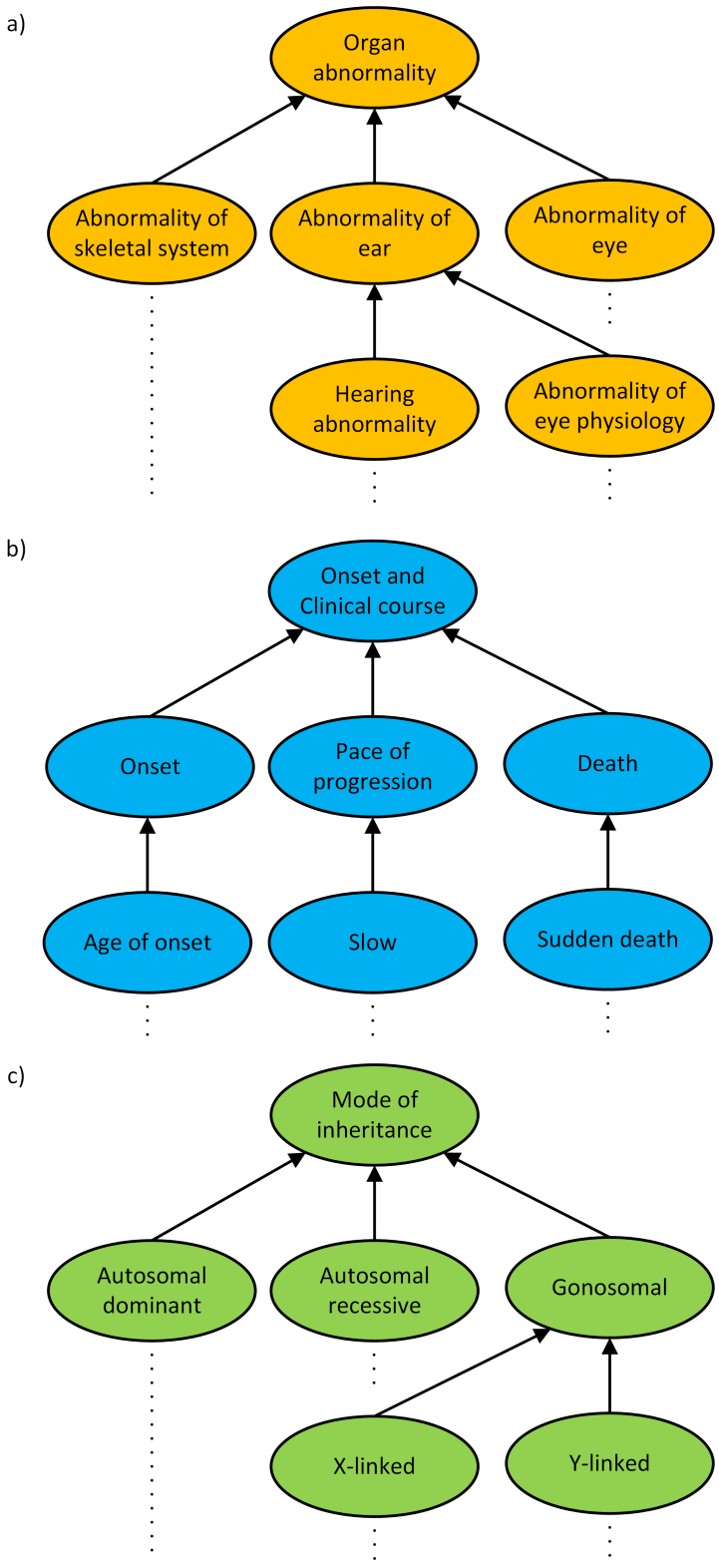

FIGURE 1.2. Portions of the HPO subontologies (a) organ abnormality (b) onset and clinical course (c) mode of inheritance.

as the cost of sequencing continues to decline rapidly [11] leading to an increasingly larger gap between sequences with and without annotations, it is not realistic to use experimental determination as the sole means of annotation. This has led to a large fraction of proteins without a single GO/HPO annotation – in fact only less than 1% of proteins have any experimentally validated annotations to-date [12]. So, it goes without saying that a significant interest in automated annotation efforts became commonplace during the last decade especially because of the fact that having so many annotations to work with makes it possible to use machine learning to predict new annotations as described below.

1.3. AFP: Automated Function Prediction

In recent years, the bioinformatics community has witnessed the rapid emergence of many computational methods that can automatically predict GO terms for a given protein [13]. Many of the earliest methods performed "transfer of annotation" using amino acid sequence similarity to proteins with known functions [14]. Over the years, discriminative algorithms such as SVMs [15, 16] and decision trees [17, 18], as well as probabilistic models which perform label propagation on graphs [19–21] have been developed. More recently, structured output methods, such as GOstruct [22] which was developed by our lab, have come to the forefront in GO term prediction. The reasonable success in computationally determining the functions of proteins using a variety of data sources – by protein sequence or structure similarity or using various biological network data [23, 24, 22, 13] – has led to the establishment of automated function prediction (AFP) as one of the most important bioinformatics challenges in the last decade. However, according to a community based competition for evaluating automated function prediction methods (CAFA, see below), levels of accuracy of current methods need significant further improvement [13].

1.4. CAFA: Critical Assessment of Function Annotation

Due to this emergence of a multitude of computational methods for GO term prediction, the community has realized the need for a systematic and organized means of comparing the performance of these methods in order to assess how far the area has progressed. Taking note from critical assessment efforts such as CASP (Critical Assessment of protein Structure Prediction) [25] and CAPRI (Critical Assessment of Prediction of Interactions) [26], the AFP community decided to hold its own competition: CAFA (Critical Assessment of Function Annotation) [13]. The main objective of CAFA is to gather all AFP researchers in one place to fairly assess and compare the latest computational methods using a centralized and independent assessment. In the first CAFA (CAFA1) the participants were provided with a list of protein targets that didn't have any previous GO annotations and were asked to submit computational predictions using their own AFP methods [13]. Once the predictions were submitted the organizers collected the experimentally validated GO annotations acquired for those target proteins over a period of 6 months. Finally the computational predictions were compared against those annotations to compute the accuracy of each AFP method.

The recent CAFA2 challenge[1] [27] had exactly the same setup, except that the list of 100,000 target proteins consisted of both annotated and unannotated proteins. The added requirement of making predictions on currently annotated proteins makes CAFA2 a more realistic representation of the function prediction problem, as it better models the accumulation of annotations over time.

[1] http://biofunctionprediction.org/

1.5. HMC: Hierarchical Multilabel Classification

As mentioned above, both the Gene Ontology and and the Human Phenotype Ontology are well defined hierarchies with directed acyclic graph structure. Therefore protein function prediction of either GO or HPO terms falls into the category of hierarchical multilabel classification (HMC) problems [28], as a given protein can be annotated with multiple labels (i.e., categories), and the set of those labels have a hierarchy associated with them.

The traditional approach for solving HMC problems is to decompose the problem into multiple single label problems and apply independent binary classifiers for each label separately [29]; however, this approach has several disadvantages. First, independent classifiers are not able to learn from the inter-relationships between the labels. Second, the leaf terms typically have a low number of annotated examples making it difficult to learn an effective classifier. Furthermore, the predicted labels are typically hierarchically inconsistent, i.e., a child term (e.g. "protein binding" in GO) is predicted while its parent term (e.g. "binding") is not—making it difficult to interpret the predictions. To remedy this problem, an additional reconciliation step of combining independent predictions to obtain a set of predictions that are consistent with the topology of the ontology is required (see e.g. [30] for a discussion of several of reconciliation methods that are effective for GO term prediction).

An alternative approach is to use a single classifier that learns a direct mapping from inputs (i.e., proteins) to the space of hierarchically consistent labelings; this can be achieved using structured prediction, which is a framework for learning a mapping from inputs to label spaces that have a structure associated with them [31]. This framework can capture information from the inter-relationships between labels and allows the prediction a set of labels that are hierarchically consistent, eliminating the need for multiple classifiers, and the

need for establishing hierarchical consistency between the predictions. Previously we have shown the effectiveness of modeling the GO term prediction problem using a structured prediction framework [22, 32].

1.6. Protein Function Prediction Evaluation Protocols

As mentioned above, the AFP problem posed in CAFA is more challenging than the typical machine learning problem, as the usual assumption in machine learning is that the distribution of examples in the training set is reflective of that in the test set. In the CAFA AFP problem this assumption likely does not hold because the training is performed on an older set of annotations while testing is performed on newer annotations; and it is known that distribution of GO categories changes over time due to strong biases in the annotation process [33]. Furthermore, the annotations acquired for annotated proteins and the annotations acquired for unannotated proteins can be expected to be different in frequency and specificity: an annotated protein can be expected to acquire more specific GO categories than an unannotated protein, and perhaps more of them, as the biology community tends to study proteins that are already characterized.

We identify the CAFA2 requirements as a combination of two subtasks: making predictions on annotated proteins and making predictions on unannotated proteins. In the task of making prediction on annotated proteins, methods are trained using the set of annotations acquired on or before a specific time-stamp t_1, and tested on the set of annotations gathered on the same set of proteins after t_1. In other words, the same set of proteins are used for training and testing, but the training labels are annotations that were available at t_1, while testing labels are annotations made available after t_1. In the task of making predictions on unannotated proteins, methods are trained using the set of annotations acquired on or before

t_1 while they are tested on the annotations acquired for proteins that were not annotated on or before t_1. In this setup, the proteins used for training and the proteins used for testing are disjoint sets. An overview of the different sets of annotations used for these two AFP tasks is given in Figure 1.3.

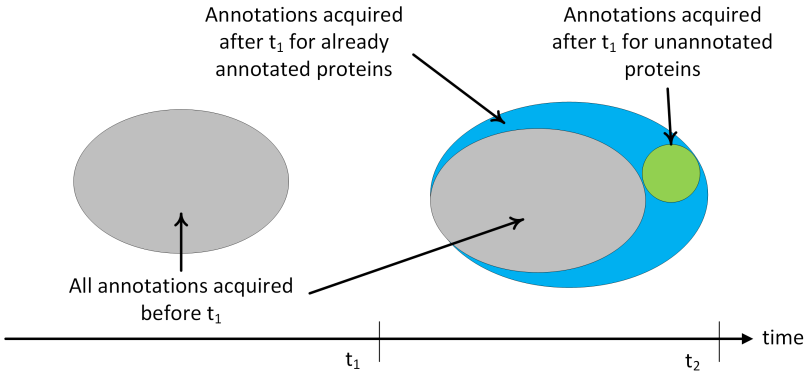

FIGURE 1.3. Overview of the two AFP subtasks. We distinguish between three sets of annotations that are used to define the train/test set with respect to the two tasks: making prediction for annotated proteins and making prediction for unannotated proteins. Annotations accumulate between an initial time t_0 until t_1 and form the set in grey, which is the training set for the task of making prediction for annotated proteins. The set of annotations acquired for those proteins after t_1 form the set in blue, which is the test set in the task of making prediction for annotated proteins. The set of annotations acquired after t_1 for proteins that were un-annotated before t_1 is denoted by the set in green, and is used as the test set in that task.

In our first project, we investigate the differences between these two AFP tasks with respect to GO term prediction and how different AFP methods perform in each case. We compare the performance of AFP methods on these tasks with their performance in cross-validation, which is typically used by many to assess and compare AFP methods. Furthermore, we determine whether the performance metric has an impact on the ranking of different methods. Our results have ramifications for practitioners in the area of AFP which should help them in the design of their computational experiments. Based on the findings of this study, we proposed several improvements in GOstruct which led to better handling of

these two subtasks and no degradation in performance in cross-validation. The new version of GOstruct based on the proposed improvements, denoted has GOstruct 2.0, has improved performance compared to original GOstruct, especially on the task of making predictions on annotated proteins.

1.7. Biomedical Literature Mining for Protein Function Prediction

The biomedical literature is the most up-to-date resource of biological knowledge obtained through experimentation. The role of bio-curators [34] is to perform the task of manually analysing, interpreting and integrating the various biological knowledge into standardized databases such as the Gene Ontology [35] and BioGRID [36]. The workflow of a typical bio-curation pipeline is composed of identification of relevant articles, detection of bio-entities (such as protein names), identification of relationships between entities along with supporting evidence (e.g. experimental evidence for the interaction of two proteins) and finally, entering that information into the database [37]. Due to the tedious nature of the various tasks associated with the workflow described above, it is no wonder that the speed of curation has not been able to keep up with the rate of publication [38].

Since the rate-limiting factor for bio-curator maintained databases such as the Gene Ontology is the process of manual curation of data, biomedical literature mining has gained significant attention as a potential solution in recent years. In fact, biomedical literature mining has shown great promise in tasks such as protein-protein interaction extraction [39–42]. But it is still unclear how useful it is for AFP, as little work has been done in the area [43]. Although most AFP methods developed in recent years make use of heterogeneous data source such as sequence, structure, and protein-protein interactions [13], only a few of them utilize information gathered from the biomedical literature. This is surprising given that

the biomedical literature contains a wealth of information related to functions of proteins. However, there is no clear consensus on how to best process the unstructured literature information and what types of features work well for predicting protein function.

In CAFA1, only two teams out of 54 that submitted predictions exclusively used literature-derived features [13]. Wong et al. used a nearest-neighbour/support vector machine approache with literature features extracted according the statistical significance of the difference in word co-occurrence probability between GO categories. However, they used only 34 GO categories in the second level of the hierarchy for their experiments [44, 45]. They observe that the biomedical literature data is highly competitive with other data sources and that combining literature features with others leads to improved performance [44, 45]. Another team utilized events (such as molecular interaction) extracted from the literature for complementing other data sources for predicting 385 of the most common GO categories [46]. They found that although literature features by itself showed poor performance, those features combined with other data sources outperform Blast2GO [14], a popular annotation transfer method based on sequence similarity. They concluded that even though event-type text mining shows promise as a valuable source of data for protein function prediction, further research is needed to make it more effective [46].

In our second project, we conduct the first ever comprehensive evaluation of the informativeness of literature data and their effectiveness in GO term prediction. Specifically, our collaborators developed a natural language processing (NLP) pipeline that processes millions of biomedical papers to extract two different types of literature features: co-occurrences of protein-GO terms ("co-mentions") as well as bag-of-words features. We compare the two types and assess how to best combine them for achieving the highest accuracy in GO term

prediction. We also show that this pipeline is able to recall many correct co-mentions; but a considerable amount of false positives are also extracted. Motivated by this observation, we developed a method for filtering/classifying co-mentions for improving the quality of predictions as described below.

Co-mention classification is the problem of classifying whether a given co-mention of a protein name and a functional category implies a functional relationship between the two entities according to the semantics of the context surrounding the two entities. In this problem, a co-mention is a positive example if the context containing the two entities suggest that the mentioned protein possesses the functionality described by the mentioned functional category (see Fig. 6.1). Therefore this can also be identified as the task of automatically extracting protein-function relations from the biomedical literature. We formulate this as a supervised learning problem and develop a novel classification method that uses random walk graph kernels as its underlying model. We also implement several other approaches based on (1) bag-of-words (2) word embeddings and (3) graphlet kernels for the purpose of comparison. We perform a comprehensive evaluation of these methods and the experimental results indicate that word embeddings based methods work best overall while the random walk kernel based method outperforms the graphlet kernel. Overall results indicate that automatic extraction of protein-GO relations shows great promise as a complementary approach that can help human curators make the process more efficient by prioritizing of co-mentions with high prediction scores. This is the first study of the problem of co-mention classification or automated protein-function relation extraction from the biomedical literature.

1.8. Human Phenotype Ontology Term Prediction

Our third and final project focuses on the problem of HPO term prediction. As mentioned above, currently, only a small fraction (∼3,000) of human protein coding genes are known to be associated with hereditary diseases, and only those genes have HPO annotations at the moment. But researchers believe that there are many other disease-causing genes in the human genome and estimate that another 5,000 genes can be associated with phenotypes. Therefore, it is important to explore the feasibility of developing computational methods for predicting gene-HPO associations. While there is a plethora of computational approaches for the related task of prediction of gene-disease associations [47], our method is one of the only two computational methods [48] that can directly predict gene-HPO term associations at the time of writing.

We define the HPO prediction problem as directly predicting the complete set of HPO terms for a given gene. As described above, this problem is a hierarchical multilabel classification problem [28], since a given protein can be annotated with multiple HPO categories that are hierarchically related to each other. In our previous work, we have shown that modeling the GO term prediction problem using structured prediction approach is highly effective [22, 32]. In this work we demonstrate the effectiveness of this strategy for HPO term prediction using the same methodology, and explore a variety of data sources that are useful for this task, including large scale data extracted from the biomedical literature. We compare our method with several baseline and non-structured methods and show that formulating the HPO term prediction as a HMC problem provides overall better performance.

1.9. Publications associated with the Presented Work

The details of the study on protein function evaluation protocols was published in the GigaScience Journal [49] where we demonstrated that the task of predicting for annotated proteins is more challenging. Consequently we developed GOstruct 2.0 by proposing modifications that help it handle this task better; we plan to submit our work on GOstruct 2.0 to The 7th ACM Conference on Bioinformatics, Computational Biology, and Health Informatics (ACM BCB). We published the findings of our exploratory study on the use of text-mining features for automated protein function prediction in the Journal of Biomedical Semantics [50] in which we detail an NLP pipeline that can extract co-mention and bag-of-word features from Biomedical text. Based on the finding that this pipeline produces a considerable amount of false positives, we developed co-mention classifier based on graph kernels which can be used to filter co-mentions. Finally, our work on HPO term prediction and PHENOstruct was published in the F1000 Journal [51].

1.10. Overview of Chapters

We start off by describing GOstruct and previous work on biomedical relation extraction in Chapter 2. In Chapter 3, we detail our comprehensive study on different protein function evaluation protocols and present the major findings of the study and their ramifications for the AFP community. We present GOstruct 2.0 and the proposed improvements that help it better handle the task of making predictions for annotated proteins in Chapter 4. We demonstrate its improved accuracy on human data by comparing to the original GOstruct. In Chapter 5, we describe our comprehensive study for evaluating the informativeness of literature features for protein function prediction. Based on the finding that our NLP pipeline produces a considerable number of false positives, we developed a novel method based on

random walk kernels for automatically extracting protein-function relations from biomedical text as described in Chapter 6. We evaluate the performance of this and several other approaches on manually curated co-mention datasets and demonstrate that an automated pipeline for extracting protein-function relations from text shows great promise as a tool that can be used by bio-curators for making the curation process significantly efficient. In Chapter 7, we focus our attention to the problem of HPO term prediction. We model this problem as a structured prediction problem and implement PHENOstruct which is capable of predicting a set of HPO terms for a given protein. We compare PHENOstruct with several baseline methods and show that modeling this as a structured prediction problem provides better performance overall. In Chapter 8, we summarize the major contributions of our work and discuss several open problems.

CHAPTER 2

Previous Work

2.1. Data sources for AFP

With the advent of various computational methods that can automatically predict GO categories for a given protein, many different informative data sources such as sequence and structure have come to light over the years [13]. Proteins are chains of amino acids that fold into a 3-dimensional structure [2]. The sequence and its structure typically define which functions are performed by different functional domains of a protein; therefore it is understood that proteins with similar sequence (even across different species), that arose from a common ancestor and known as *homologs*, tend to perform similar functions [2, 52]. Therefore, many of the earliest methods solely utilized the sequence and structural similarity between proteins for transferring the known annotations of a protein to its (unannotated) homologs (i.e., other proteins that share ancestry) based on the knowledge that sequence/structural similarity is correlated with functional similarity. Popular sequence alignment tools such as BLAST [53, 54] and BLAT [14] are used for computing sequence similarity. These tools align the sequence of a query protein to a large database of annotated protein sequences and retrieves a list of similar sequences along with confidence scores. Similarly, I-TASSER [55] and HHpred [56] are two of the leading structural alignment tools used for computing structural similarity.

Proteins must be co-localized to the same cell compartment in order to cooperate for performing a certain function. Therefore, information about protein localization can be informative of their functions due to the fact that many biological processes are known to be

localized to certain cellular compartments [57]. The 'sorting signals' that direct the transportation of a particular protein and ultimately determine where it ends up are contained in the amino acid sequence. These are called sequence motifs (i.e., short sub-sequences that tend to frequently re-occur across sequences and are believed have some biological significance). Tools such as WOLF PSORT program [58] can be used for extracting localization features. WOLF PSORT converts the amino acid sequence to a set of numerical features based on sequence properties such as motifs and amino acid composition [58]; these features are predictive of sub-cellular localization and in turn their functions.

Proteins perform their function by interacting with other proteins and macromolecules; therefore the set of interaction partners a protein has often can reveal valuable information about the types of processes they are involved in [2]. Based on this intuition, various protein-protein interaction and other functional association data networks are being used as input features in many AFP methods. Examples of other functional associations are co-expression (calculated based on the set of genes that tend to express together) and co-occurrence (calculated based on proteins that tend to be located in the same cell compartments often) [59]. BioGRID [36] and STRING [59, 60] are two of the most popular databases containing these biological network data. The BioGRID database provides binary interactions acquired from physical and genetic interaction experiments [36]. STRING provides interaction/association data between two given proteins based on several different evidence channels (such as co-expression and co-occurrence) [60].

The biomedical literature is the largest collection of functional knowledge available to the AFP community. A functional category, protein name or a name of other macromolecules mentioned alongside a particular protein name within a relatively short span (such as a

sentence, paragraph or an abstract) can be very informative of that protein's function [50]. Hence, there have been recent attempts to automatically extract co-occurrences from text such as protein-GO category pairs [50] and protein-molecule names [61] and use them as input features for predicting protein function. However, not all co-occurrences of protein names and other biological entities in text represent valid functional relationships; methods that can classify or filter these extracted co-occurrences are required to improve the quality of this data [50]. In Chapter 6 we propose a novel pipeline for classifying protein-GO category co-occurrences extracted from biomedical text.

2.2. AFP methods

Many of the earliest AFP methods were homology-based techniques which simply performed "transfer of annotation" using amino acid sequence similarity to proteins with known functions [14, 62, 63]. These methods use a threshold on the sequence similarity scores for finding the top-k other annotated sequences and simply assign their annotations to the query protein. For example, Henning et al. uses the confidence scores generated by BLAST for transferring GO annotations [63]. However, studies have shown that simple transfer of annotation is likely to produce lower accuracies due to the fact that the assumption of homologs having similar function has limited validity [64, 52]. Consequently there has been recent attempts that use homology features as input to much more advanced algorithms (rather than use them in a simple nearest-neighbour type approaches mentioned above). For example, FANN-GO represents each protein using a fixed-length vector in which each dimension corresponds to a function category. These scores are computed using sequence alignment similarity scores and represent the confidence that a particular protein is associated with the categories. These vectors are used for training an ensemble of neural networks which can

ultimately predict the functional categories of a previously unseen protein [24]. In our work, we utilize a similar set of scores as input to our methods; see Appendix A for a detailed description of how they are calculated.

Over the years, discriminative algorithms such as SVMs [15, 16] and decision trees [17, 18] have been extensively used for protein function predictions. Both Guan et al. [65] and Obozinski et al. [15] developed AFP methods that use ensembles of SVMs that are trained on a combination of several data sources such as protein sequence information and protein-protein interactions. Hayete et al. [18] developed an AFP method that trains an ensemble of decision tree classifiers for each GO category using protein sequence data.

Another popular approach for AFP is 'guilt-by-association' [66] which typically uses label propagation algorithms on a network generated from functional associations [19–21]. At its core, most of these methods utilize the concept of nearest-neighbour/neighbour voting to compute confidence scores for an unannotated node and a given functional category by their association to neighbouring nodes that are already annotated with that particular category. For example, GeneMANIA [21, 23], which is one of the most popular guilt-by-association AFP methods, uses combined functional association network generated from various data sources such as co-expression, co-localization and physical interactions. Networks are combined using different weights per each GO category. GeneMANIA first computes discriminant scores for each training node which measure the level of association between the corresponding protein and the particular GO category. Then a Gaussian field label propagation algorithm [67] is applied for propagating these scores to all the test proteins in the network.

All of the methods described above extensively model the AFP problem as multiple binary label prediction problem where each binary label represents a single functional category. So,

essentially these methods try to answer the question "Does protein A perform a certain function B?" The downsides of this approach are: (1) inability to learn from the inter-relationships between the labels, (2) some of the categories at the bottom of the hierarchy may have a low number of annotated examples making it difficult to learn an effective classifier and (3) predicted labels may be hierarchically inconsistent. To remedy the third issue, some binary classification methods use an additional reconciliation step of combining independent predictions to obtain a set of predictions that are consistent with the topology of the ontology using Bayesian networks or logistic regression [15, 16, 68]. For example, Obozinski et al. [30] describe 11 different reconciliation methods that are effective for GO term prediction such as simple heuristics and more sophisticated Bayesian methods.

In order to address all the above issues associated with binary prediction, an elegant solution is to model the AFP problem as a hierarchical multilabel classification problem (i.e., ask the question "which categories are associated with a particular protein?"). For example, Clus-HMC-Ens is a state-of-the-art HMC method based on decision tree ensembles which has been shown to be very effective for GO term prediction [69]. Modeling the AFP problem as an HMC problem can also be achieved using structured prediction [31], which is a learning framework for inferring a mapping between inputs and structured label spaces such as GO categories. In view of this, previously our lab has developed GOstruct [22, 32], an AFP method based on structured prediction framework, which is described in detail below.

2.3. GOstruct

GOstruct [22] is an AFP method developed by our lab that uses a structured SVM [31], which allows it to explicitly model GO term prediction as a hierarchical multi-label prediction problem [28]. The structured SVM is able to address prediction problems where the labels,

or outputs, have complex inter-relationships; in the AFP setup, this allows us to use a single classifier for each of the GO subontologies or even model them together. Like other structured output methods, the GOstruct structured SVM learns a compatibility function that models the association between a given input and a structured output [31] as described next.

Let $\mathcal{X}$ be the input space where proteins are represented and let $\mathcal{Y}$ be the space of labels (GO categories). The set of GO categories annotated to a given protein is collectively referred to as its (structured) label. $\mathcal{Y}$ represents each GO subontology in a vector space where component i represents category i. Given a training set $\{(x_i, y_i)\}_{i=1}^{n}$ where $x_i \epsilon \mathcal{X}$ and $y_i \epsilon \mathcal{Y}$, the *compatibility* function $f : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ maps input-output pairs to a score that indicates the strength of the association of an input to a set of GO categories. The predicted label $\hat{y}$ for an input x can then be obtained using the argmax operator as $\hat{y} = \arg\max_{y \in \mathcal{Y}_c} f(x, y)$ where $\mathcal{Y}_c \subset \mathcal{Y}$ is the set of all candidate labels. In this work we use the combinations of all GO categories present in the training set as the set of candidate labels $\mathcal{Y}_c$. The intuition is that the combinations of GO categories present in nature is more representative of combinations that are biologically meaningful.

In order to obtain correct classification, the compatibility value of the true label (correct set of GO annotations) of an input protein needs to be higher than that of any other candidate label. GOstruct uses structured SVM training where this is used as a soft constraint; it tries to maximize the margin, or the difference between the compatibility value for the actual label and the compatibility for the next best candidate [31].

GOstruct uses the following structured SVM formulation for learning,

$$\min_{w,\epsilon_i} \frac{1}{2}||w||_2^2 + \frac{C}{n}\sum_{i=1}^{n}\epsilon_i$$

(1) $$\text{subject to}: f(x_i, y_i) - \max_{y \in \mathcal{Y}_c} f(x_i, y) \geq 1 - \epsilon_i \qquad \text{for } i = 1, ..., n$$

(2) $$\epsilon_i \geq 0 \qquad \text{for } i = 1, ..., n$$

where w is the weight vector, C is a user-specified parameter, $\mathcal{Y}_c$ is the set of candidate labels, ϵ_i are the slack variables and $||\cdot||_2$ is the L^2 norm. The first constraint, Equation (1), ensures that the compatibility score for the actual label of a protein is higher than all other candidate labels.

In the structured-output setting, kernels correspond to dot products in the joint input-output feature space, and they are functions of both inputs and outputs. GOstruct uses a joint kernel that is the product of the input-space and the output-space kernels:

$$K((x_1, y_1), (x_2, y_2)) = K_{\mathcal{X}}(x_1, x_2)K_{\mathcal{Y}}(y_1, y_2).$$

Different sources of data are combined by adding linear kernels at the input-space level, and for the output space we use a linear kernel between label vectors. Each kernel is normalized according to

$$K_{norm}(z_1, z_2) = K(z_1, z_2)/\sqrt{K(z_1, z_1)K(z_2, z_2)}$$

before being used to construct the joint input-output kernel.

GOstruct uses a set of heterogeneous data sources as input features: features based on protein sequence properties (such as localization features), homology-based features, features

computed from functional association networks (such as protein-protein interaction data) and text mining features extracted from biomedical text [32]. Combining heterogeneous data sources and modeling the problem as an HMC problem makes GOstruct very effective in protein function prediction [22, 32]. In fact GOstruct produces state-of-the-art performance; it was one of the top performers in CAFA1 [13].

In our recent work we have shown that GOstruct struggles to achieve the same level of accuracy in the task of making predictions for already annotated proteins compared to its accuracy measured by cross-validation [49]. The main reason for this decline in performance is because GOstruct does not utilize a protein's current annotations when making predictions for an annotated protein. This makes the model less flexible because it does not take into account the incomplete nature of current annotations. Therefore we develop GOstruct 2.0 by proposing novel constraints that account for the above mentioned limitation of the model. Chapter 4 details these proposed improvements and the corresponding experimental results that demonstrate that GOstruct 2.0 clearly outperforms original GOstruct on the task of making predictions for annotated proteins.

2.4. AFP as a Biomedical Relation Extraction Problem

Relation extraction is the problem of automatically extracting relations between entities from text [70]. Biomedical relation extraction is the more specific problem of extracting a relationship between two (biological) entities in biomedical text [71]. Biomedical relation extraction methods have been used for extracting protein-protein interactions, drug-drug interactions, mutation-disease associations, drug-effect associations and several others [72]. The majority of biomedical relation extraction methods to date have been intended for protein-protein interaction extraction [39–42].

Many of the earliest relation extraction methods have been simple co-occurrence based methods which used concepts such as mutual information, chi-squared test and co-citation [73–80] or rule-based/pattern-based methods that used manually created relation templates such as "effector-relation-effectee" (for e.g. A activates B) [81–83]. Often these rule-based methods provided high precision but low recall due to the fact that manually created rules are too specific such that they captured all relations adhering to that template but lacked the ability to generalize [84].

Later, relation extraction methods based on supervised learning techniques have been developed which provided better performance. Many of the early supervised learning methods were based on explicitly computed features such as all words between the two entities, number of words between them, flags indicating whether the two mentions are part of the same phrase, words on which the entities are dependent on according to dependency tree, path connecting entities in the parse tree and bag-of-words features generated from the three regions (i.e., before, between and after) separated by the two entities [85, 86]. However, some of the most successful methods in this area are kernel-based methods [72]. Giuliano et al. [39] introduced a shallow linguistic kernel that combines a global context kernel (three sets of bag-of-words generated from the regions separated by the two entities) and a local context kernel in which ordered sequence of words within the neighbourhood of entities are compared; this was intended for extracting protein-protein interaction relations from text. Some of the other similar methods define kernels on various structural representations of the sentences such as trees and graphs; Kim et al. [40] introduced a kernel on dependency graphs generated from the sentences which counts the number of common sub-graphs for the task of protein-protein relation extraction. They also introduced a hybrid kernel which counts

the common substructures (i.e., very short walks) in dependency graphs [41]. For the same task, Chowdhury et al. use a tree kernel which counts the common sub-trees enclosing the two proteins in the parse tree generated by the corresponding sentence [42].

There does not exist any method for extracting protein-function relations from text at the time of writing. One reason for this absence is the fact that even recognizing entities such as a mention of a GO category (and to a lesser extent a protein) in itself is a very challenging task [50]. But the main reason is the complex nature of the typical context (for e.g. sentences) containing a functional relationships as opposed other relationship such as protein-protein interactions. For example, templates such as "protein A interacts with protein B" or "interaction of protein A and protein B" represent a large fraction of the sentences containing protein-protein interactions [87] while it is almost impossible to develop templates to capture protein-function relationship due to having vast differences between corresponding sentences. Therefore, using template-based or rule-based relation extraction methods for extracting protein-function relations is not feasible. In Chapter 6 we describe a novel method based on graph kernels for extracting protein-function relations from text.

CHAPTER 3

A close look at protein function prediction evaluation protocols

3.1. Introduction

The Gene Ontology (GO) [7, 35] is a community-based resource on functional genomics. It provides a structured vocabulary (i.e., ontology) which describes functions of gene products (e.g. genes, proteins, ncRNAs, complexes) in a hierarchical manner. GO is composed of thousands of categories and they are organized in a directed acyclic graph (DAG); more general terms are found at the top (i.e., near the root node), and term specificity increases from the root to the leaves. This implies the "true-path rule": whenever a gene is annotated with a given term, that implies the annotation of all its ancestor terms. GO is composed of three independent subontologies: molecular function, biological process and cellular component. The molecular function describes elementary activities at the molecular level. The biological process subontology describes the operations or sets of molecular events. The cellular component subontology describes locations, at the levels of subcellular structures and macromolecular complexes. The molecular function, biological process and cellular component subontologies are composed of more than 3,000, 8,000 and 1,000 categories, respectively.

A "GO annotation" provides the association between a specific GO category and a gene product. It is typically accompanied by an evidence code which indicates the type of experiment or methodology used for obtaining the annotation. Currently there are more than 100,000 experimentally validated annotations for about 25,000 distinct proteins belonging to

more than 1,000 species. However, due the highly resource-intensive nature of these studies the majority of the proteins are still unannotated [10].

For the last decade, developing methods for automatically predicting GO terms has become a very active research area. Initially, "transfer of annotation" methods were the most popular methods which solely utilized the amino acid sequence similarity to transfer exiting annotations to their unannotated homologs [14]. But in later years, many other advanced methods that use various discriminative algorithms such as Support Vector Machines [15, 16], decision trees [17, 18] and probabilistic models [19–21] emerged as better performers. In previous work our lab developed GOstruct [22] by modelling the problem of GO term prediction as a structured output problem which provides state-of-the-art performance [32].

CAFA (Critical Assessment of Function Annotation) [13] competition is organized every two years for gathering all AFP researchers in one place to evaluate the latest AFP methods using an independent and standardized assessment. The target proteins in the first CAFA (CAFA1) was a list of proteins that didn't have any previous annotations at the time but the target proteins for CAFA2 was composed of both annotated and unannotated proteins. Therefore CAFA2 requirements are a combination of two subtasks: making predictions on annotated proteins and making predictions on unannotated proteins.

In this work we clearly delineate the differences between these two AFP tasks and how performance of different AFP methods vary in each case. We compare the performance of AFP methods on these two tasks versus their performance in cross-validation, which is the popular technique for evaluating AFP methods. Additionally, we investigate how the performance metric impacts the ranking of different AFP methods.

3.2. Prediction Tasks

We identify two tasks in the area of automated function prediction: prediction of annotations for proteins without previous annotations (denoted as "novel-proteins", or NP) and prediction of novel annotations for proteins that already have some annotations associated with them (denoted as "novel annotations", or NA). In order to understand the relative difficulty of these tasks we compare the performance of several AFP methods under evaluation protocols that directly capture these tasks and also compare them to the typically used alternative, which is cross-validation. In what follows we describe in detail the protocols, the data used, and the algorithms that were compared.

3.3. Evaluation Protocols

We compare three experimental setups: cross-validation (CV), novel-annotations (NA) and novel-proteins (NP). CV is the standard cross-validation setup which is typically used for evaluating AFP methods; more specifically, we use 5-fold cross validation where each fold corresponds to a distinct randomly chosen set of proteins. In the NA protocol, methods are trained using the set of annotations acquired on or before the year 2009, and tested on the set of annotations gathered on the same set of proteins after 2009 (the "GO annotations" section explains the criteria used for selecting the final set of annotations). In other words, the same set of proteins are used for training and testing, but the training labels are annotations that were available in 2009, while testing labels are annotations made available after 2009. An annotation that was added with a new evidence code, but present in the training set was not included in testing; similarly, we do not include an annotation in the test set when a more specific annotation already exists in the training set. In the NP protocol methods are trained using the set of annotations acquired on or before the year 2009 while they are tested

on the annotations acquired for proteins that were not annotated on or before 2009. In this setup, the proteins used for training and the proteins used for testing are disjoint sets. An overview of the NA and NP setups is given in Figure 3.1.

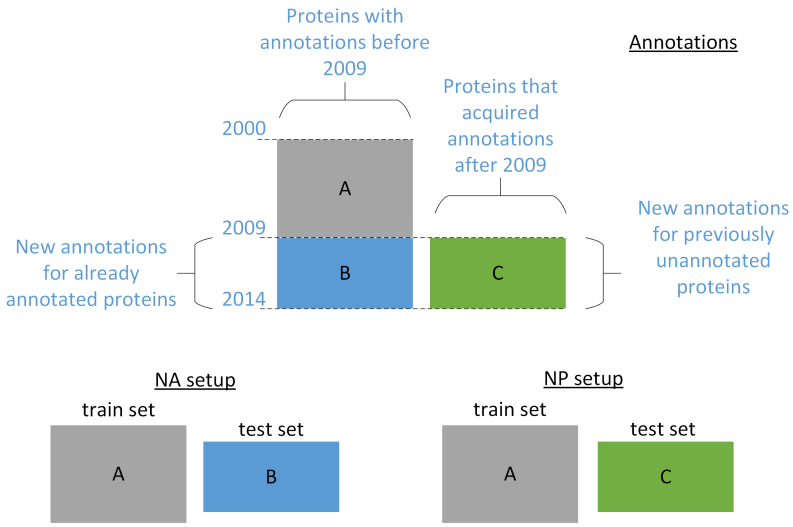

FIGURE 3.1. Overview of the NA and NP setups. We distinguish between three sets of annotations that are used to define the train/test set in the two setups. Annotations accumulate between an initial time t_0 until t_1 (beginning of 2009 in our experiments) and form a set A, which is the training set in both NA and NP. The set of annotations acquired for those proteins after t_1 form a set B, which is the test set in the NA setup. The set of annotations acquired after t_1 for proteins that were un-annotated before t_1 is denoted by the set C, and is used as the test set in the NP setup.

In our experiments we focused on yeast and human. Yeast (S. cerevisiae) was the first eukaryotic genome that was completely sequenced [88] which paved the way for it to be one of the most intensively studied model species while understanding the functional roles of human (H. sapiens) is a highest priority for tasks such as drug design [89]. The number of proteins/annotations in the train/test sets with respect to the three setups are given in Table 3.1. We note that our yeast/human test sets contain 5-10 times more proteins/annotations for each species and subontology combination than in the CAFA1 and CAFA2 challenges: In CAFA1 the test set consisted of 866 targets across 11 species consisting of

5 yeast proteins (1 with molecular function and 5 with biological process annotations) and 285 human proteins (182 with molecular function and 195 with biological process annotations) [13]. In the information made available during the CAFA2 workshop the organizers revealed that CAFA2 test sets were composed of 656, 773 and 991 proteins with molecular function, biological process and cellular component annotations, respectively. These come from 27 species, but the vast majority of them were human proteins. The large number of annotations allowed us to compute term-centric metrics in addition to the protein-centric metrics used in CAFA. We ignored GO categories that were annotated with less than 10 proteins.

TABLE 3.1. The number of proteins and the number of annotations in the train and test sets for the three setups. Each cell in the table contains #protein/#annotations. For CV setup, numbers represent average values computed across train/test folds (5-fold cross-validation).

		yeast		human	
	set.	train	test	train	test
F	CV	1532/2185	383/546	4532/8467	1133/2116
	NA	1367/1706	208/285	4305/6898	799/1343
	NP	1367/1706	521/677	4305/6898	1344/2174
P	CV	2752/5789	688/1447	7533/31794	1883/7948
	NA	2834/5161	633/990	5824/12196	3301/13192
	NP	2834/5161	604/1046	5824/12196	3574/12973
C	CV	3731/7053	932/1763	8440/19196	2110/4799
	NA	4189/6968	813/1162	5082/8185	2966/5511
	NP	4189/6968	476/681	5082/8185	5468/10200

In order to perform a fair comparison across setups we first identified the GO subgraph that consists of only the GO categories common to all three setups (CV, NA and NP). Then we computed the evaluation measures described next only on this subgraph.

3.4. Evaluation measures

Each method provides a confidence score for each of its predictions. Following the same procedure that was used in CAFA [13] these scores are recursively propagated upwards towards the root by assigning the highest score among children to their parent term. The true path rule is also applied to the ground truth in all three setups. However, in the NA setup, the older annotations (i.e., annotations acquired before 2009) are removed from the ground truth used for testing. By threshold on these propagated confidence scores we compute the following *protein-centric* measures.

In what follows N denotes the number of proteins and M is the number of GO categories. Protein-centric precision and recall at a threshold t are defined as

$$P^{pc}(t) = \frac{1}{N} \sum_{i=0}^{N} \frac{TP_i(t)}{TP_i(t) + FP_i(t)},$$

$$R^{pc}(t) = \frac{1}{N} \sum_{i=0}^{N} \frac{TP_i(t)}{TP_i(t) + FN_i(t)},$$

where $TP_i(t)$, $FP_i(t)$ and $FN_i(t)$ are the number of true positives, number of false positives and number of false negatives w.r.t. protein i at threshold t. Now we can define protein-centric F1 as

$$F1^{pc} = \frac{2P^{pc}(t)R^{pc}(t)}{P^{pc}(t) + R^{pc}(t)}.$$

which is the harmonic mean of protein-centric precision and protein-centric recall defined above.

3.5. Data

Each method was trained/tested using the same set of features and labels. We extracted GO annotations from the Gene Ontology web site, UniProt, and species specific annotation databases. We removed all annotations not originating from an experimental assay. We generated three types of sequence features (localization signals, low complexity regions and transmembrane data) as described elsewhere [32]. BLAST scores were represented using a simpler version of a score used in [24]. We extracted PPI and other functional associations data (co-expression, co-occurrence, etc.) from BioGRID 3.2.106, STRING 9.1 and GeneMANIA 3.1.2. To take advantage of the information found in the biomedical literature, an NLP pipeline was utilized to extract the co-occurrence of protein names and GO terms both at the sentence and paragraph level from all full-text publications available in PubMed; this pipeline is an improved version of the one used in our earlier work [32]. Appendix A describes the data in more detail.

3.6. Models

We used following three AFP methods in our experiments: GOstruct [22], a collection of binary SVMs, and a network-based guilt by association method (GBA). We describe all the above methods in detail below.

GOstruct. As mentioned above, GOstruct [22] is an AFP method developed by our lab. It is based on the structured SVM [31] and provides state-of-the-art performance in cross-validation [32].

The Strut library[1] with default parameter settings was used for running GOstruct.

[1] http://sourceforge.net/projects/strut/

Binary SVMs. We trained a collection of binary SVMs, each trained on a single GO category. Binary SVMs were trained using the PyML[2] machine learning library with default parameter settings. The SVMs used the same input-space kernels as GOstruct.

GBA. GBA is the in-house python implementation of the simplest form of guilt-by-association [66] AFP algorithm (i.e., neighbour-voting) also called the Basic GBA (BGBA) [90]. It is a binary classifier which computes a confidence score with respect to a given test protein and a GO category by using the connectivity of the test protein to other proteins annotated with that GO category in a given input network. More specifically, this score is equal to the fraction of direct neighbours which are annotated with that category. The kernel used for the SVM methods was used as a network for the GBA procedure.

3.7. Results and Discussion

In order to evaluate how different AFP methods perform on the NA and NP tasks, and how these evaluation protocols compare to cross-validation, we compared the performance of GOstruct, binary SVMs and GBA using the three evaluation setups, CV, NA and NP, using data from yeast and human. The results for the protein-centric F1 performance measure appear in Figures 3.2, 3.3 and 3.4. It can be observed that accuracy computed using cross-validation is much higher than in the NA and NP protocols in both human and yeast and across all three GO hierarchies. The only exception to this trend is the similar performance of GOstruct in the NP protocol, as discussed in more detail below. These results suggest that in most cases cross-validation is not a good proxy for the performance in the more realistic NA and NP protocols.

[2]http://pyml.sourceforge.net

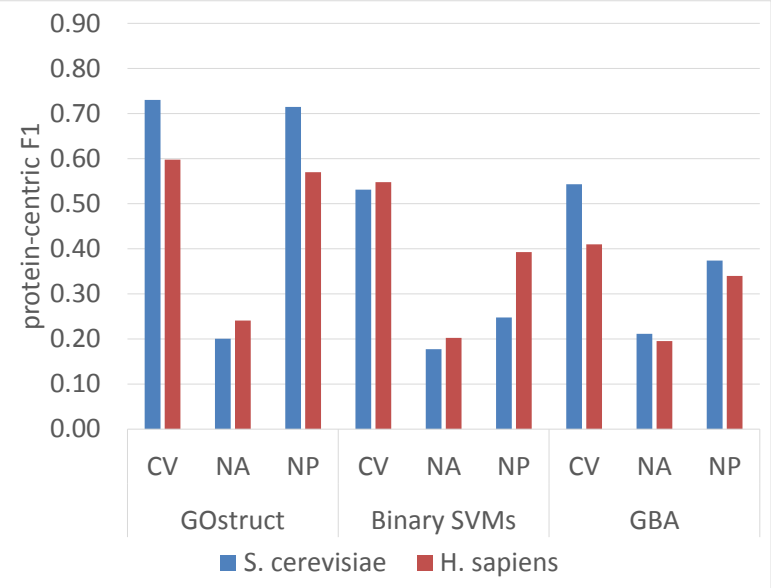

FIGURE 3.2. Performance comparison between CV, NA and NP in the molecular function subontology. GOstruct, Binary SVMs and GBA are evaluated in CV (cross-validation), NA (novel-annotation) and NP (novel-proteins) in yeast and human. Performance is evaluated using the protein-centric F1.

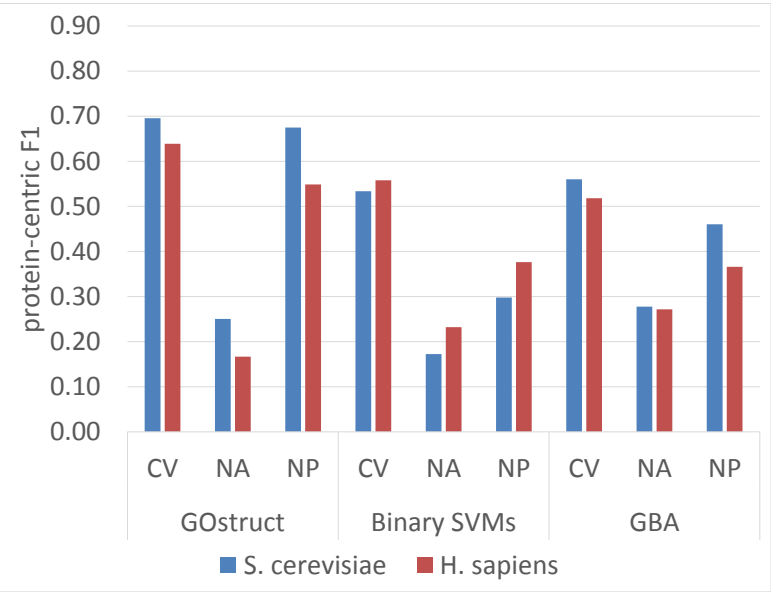

FIGURE 3.3. Performance comparison between CV, NA and NP in the biological process subontology. GOstruct, Binary SVMs and GBA are evaluated in CV (cross-validation), NA (novel-annotation) and NP (novel-proteins) in yeast and human. Performance is evaluated using the protein-centric F1.

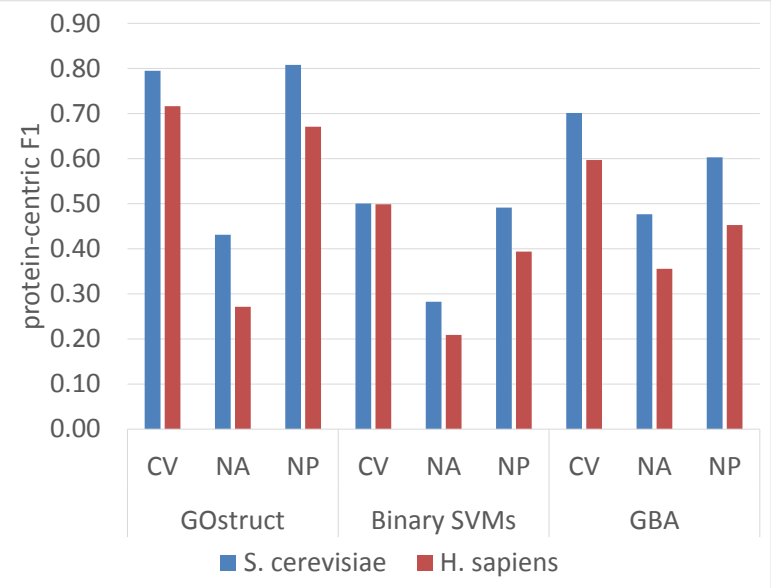

FIGURE 3.4. Performance comparison between CV, NA and NP in the cellular component subontology. GOstruct, Binary SVMs and GBA are evaluated in CV (cross-validation), NA (novel-annotation) and NP (novel-proteins) in yeast and human. Performance is evaluated using the protein-centric F1.

The observed difference in performance between cross-validation and the NA and NP protocols can be traced to the evolution of GO curation. It is known in the AFP community that the process of acquiring GO annotations is highly biased, leading to a distribution of categories that changes over time [33]. As machine learning methods rely on the assumption that the distribution of test examples is the same as the training examples, this bias makes the NA and NP tasks more difficult than performing well in cross-validation. Since CV mixes annotations across time, the training and test sets are more similar in terms of the categories annotated, and performance can therefore be expected to be higher.

In order to demonstrate the differences in the label distributions in the training set versus the test set, we performed the following analysis. First we computed the probability (no. of annotated proteins/ number of all proteins) of GO category i in the train and test sets for all three setups, denoted by p_i^{tr} and p_i^{tst}, respectively; in CV setup the calculation was

performed 5 times for each fold and averaged across the five folds. The discrepancy for category i is then defined as: $|p_i^{\text{tr}} - p_i^{\text{tst}}|/(p_i^{\text{tr}} + p_i^{\text{tst}})$. The average discrepancy, and individual signed discrepancy values (without the absolute value) are shown in Figures 3.5. We observe that the average discrepancy for NA and NP is significantly higher than in the CV setup in the three subontologies, suggesting that the label distributions between train and test sets in NA and NP is consistently different compared to CV.

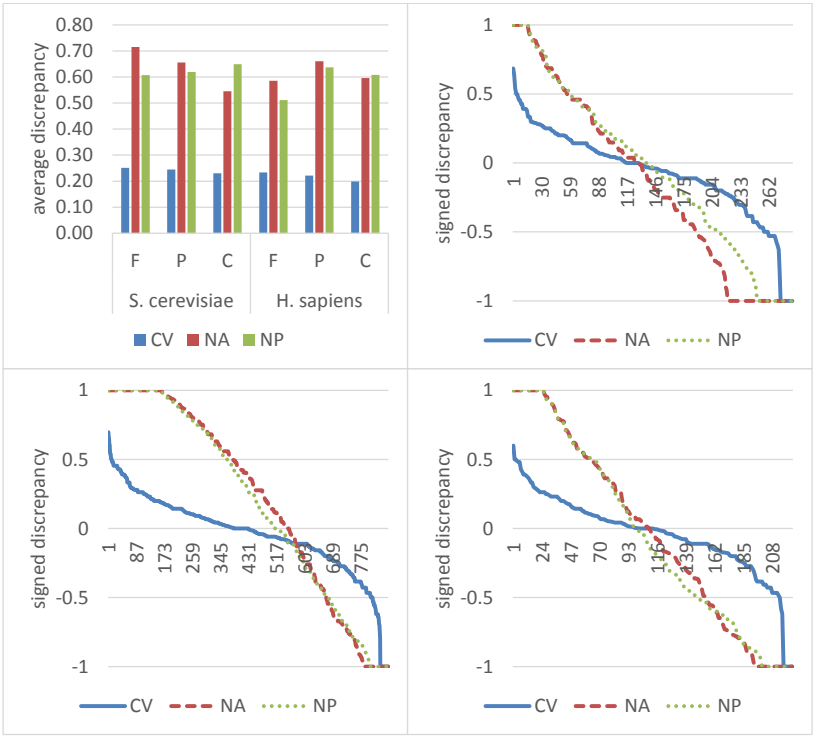

FIGURE 3.5. Label distribution comparison between CV, NA and NP. First we computed the probability (no. of annotated proteins/ no. of all proteins) of GO category i in the train and test sets for all three setups, denoted by p_i^{tr} and p_i^{tst}, respectively; in CV setup the calculation was performed 5 times for each fold and averaged across the five folds. The discrepancy for category i is then defined as: $|p_i^{\text{tr}} - p_i^{\text{tst}}|/(p_i^{\text{tr}} + p_i^{\text{tst}})$. The average discrepancy is shown in top-left panel. The individual signed discrepancy values (without the absolute value) are shown in the other three panels in sorted order by their magnitude for each setup.

We hypothesise that this characteristic is at least partly responsible for the lower performance in the NA and NP setups. To provide evidence for this hypothesis, we computed the

correlation between AUC scores and the discrepancy values for each GO category. As illustrated in Figure 3.6, the AUC scores are negatively correlated with the discrepancy values, suggesting that for GO categories which exhibit a larger difference between the train/test probabilities the performance tends to be lower. This is more pronounced for NA than NP, which is in agreement with our observation of lower accuracy for the NA protocol, as discussed below.

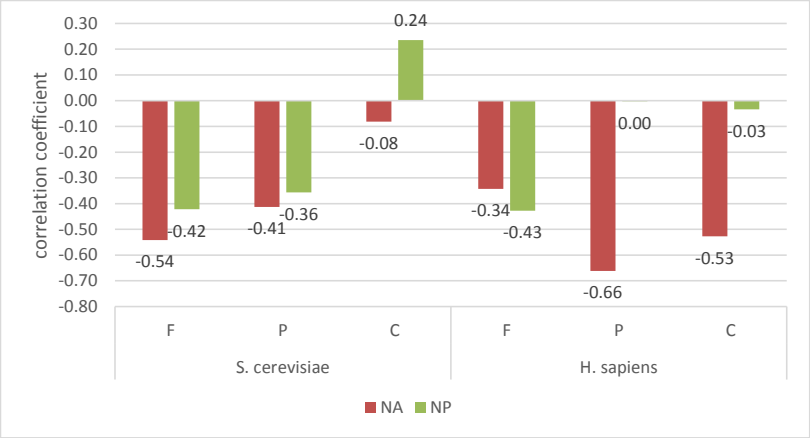

FIGURE 3.6. Pearson correlation coefficient between discrepancy of each GO category and its individual AUC.

Another observation is that the ranking of methods on the basis of cross-validation performance is not the same as that which is obtained using the other protocols. The protein-centric F1 of binary SVMs (0.55) is higher than that of GBA (0.41) in CV in the molecular function subontology in human, but its performance on the NA task is equal to that of GBA (0.20) (Figure 3.2); the protein-centric F1 of binary SVMs (0.53) is very similar to that of GBA (0.54) in CV in the molecular function subontology in yeast, but its performance in the NP task (0.25) is much lower than that of GBA (0.37) (Figure 3.2). These observations suggest that a ranking of methods established using cross-validation may not reflect how they

would rank on other AFP tasks, which has implications for the design of method evaluation in AFP.

Among the three protocols, NA yielded the lowest accuracy for all methods, i.e., the task of predicting novel annotations for previously annotated proteins is harder than prediction of novel annotations for unannotated proteins. There are several reasons why NA is harder than NP. First, unlike the NP protocol, the NA evaluation protocol uses only the annotations acquired after 2009 as the ground truth. This means that a small mistake in NA has a relatively larger impact on accuracy than what it would in NP. Second, our intuition suggests that in the beginning a protein is annotated with GO categories that are less specific or easier to experimentally verify (i.e., low-hanging fruit), and as time goes on, with the improvement of experimental assays, more specific annotations are added. We believe this is responsible for making the NA problem harder than NP. For the GOstruct method the difference between cross validation and NA accuracy is the most pronounced, although it is still the best performing method.

When it comes to the NP task, we observe different trends across methods. In this task, GOstruct performs almost equally well as in CV as can be seen in Figures 3.2-3.4, whereas all the other methods show a much bigger decline in performance. This can be attributed to the fact that GOstruct uses the set of all combinations of GO categories in the training set as its set of candidate labels. As a result, it is likely that when predicting on currently unannotated proteins those candidate labels more closely represent the GO category combinations that are expected to be annotated in these new proteins. The two other binary classifiers, namely binary SVMs and GBA do not leverage this information, and struggle to achieve the same

level of performance as in CV. But still, their performance in the NP task is always better than in the NA task.

CHAPTER 4

GOstruct 2.0: Automated Protein Function Prediction for Annotated Proteins

4.1. Introduction

Our results from the previous chapter demonstrate that the two AFP substasks of making predictions on annotated proteins and previously unannotated proteins are much more difficult than performing well in cross-validation. This suggested that the task of predicting annotations for already annotated proteins could benefit from algorithms that explicitly leverage existing annotations to better rank novel predicted annotations.

When it comes to the task of predicting for annotated proteins, there have been several recent attempts of using only the existing annotations for making predictions. In other words, given the labels associated with a protein, predict additional labels based only on its existing annotations. These methods are based on singular value decomposition [91], autoencoder neural networks [92], probabilistic latent semantic analysis [93] and Latent Dirichlet Allocation [94]. But it goes without saying that these methods miss out on the wealth of information available in all other types of genomic data sources such as protein-protein interactions. Therefore, it is worth exploring how both existing annotations and other data can be used together for this task; with methods like GOstruct and label reconciliation methods such as described in Guan et al. [65], information on existing annotations can be encoded in the inference procedure itself.

One of the factors that is likely affecting GOstruct performance for the NA task is the fact that, like most other AFP methods, it is not designed to utilize a protein's known annotations

when making predictions for it. To address this issue we implemented improvements in GOstruct to better handle the NA task by attempting to answer the question of how to best incorporate existing annotations.

As mentioned before, the key component of GOstruct's structured SVM formulation is the compatibility function, which computes the compatibility between a given protein and a label (i.e., a set of GO annotations). GOstruct learns by maximizing the difference (i.e., margin) between the compatibility score for its actual label and the second best candidate label [22]. However, while it does allow for the possibility of mis-annotations, it does not explicitly take into account that the existing annotations of a protein are partial, i.e., are only a subset of its true annotations.

GOstruct 2.0 is a modification of the original structured SVM formulation that reduces the penalty for margin violations involving examples for which the second best candidate label is an extension of its actual label. This allows the model to be more flexible with the candidate labels that are extensions of the actual labels. This models the process of annotation whereby a protein might accumulate annotations with increasing level of specificity.

4.2. Methods

In this section we present the proposed structured SVM formulation of GOstruct 2.0. As mentioned above, the original GOstruct formulation does not utilize the existing annotations of a protein to improve the quality of a label that is predicted as shown by the first constraint in structured SVM formulation of original GOstruct presented in Section 2.3.

In this work we propose a modification to the above model in order to help GOstruct better handle the NA substask. For that purpose, we first introduce the concept of a *protein-specific* label: the labels which are *consistent* with the existing annotations of a protein.

Various criteria can be introduced to define consistency, the simplest of which is to select only the combinations that contain all existing GO annotations of that protein (i.e., that combination is an "extension" of the existing label).

The original formulation does not take into account that the existing labels may or may not be complete. But in reality almost any existing annotation can be considered incomplete [38]. So in order to utilize the fact that an existing label can be incomplete, any second best candidate label that is an extension of the existing annotations should be penalized less. In view of this, we propose to modify the original structured SVM formulation as shown below.

$$\min_{w,\epsilon_i^1,\epsilon_j^2} \frac{1}{2}||w||_2^2 + \frac{C_1}{n}\sum_{i=1}^{|A|}\epsilon_i^1 + \frac{C_2}{n}\sum_{j=1}^{n-|A|}\epsilon_j^2$$

$$f(x_i, y_i) - \max_{y \in \mathcal{Y}_c} f(x_i, y) \geq 1 - \epsilon_i^1 \qquad \text{for } i = 1, ..., n_1 \tag{3}$$

$$f(x_j, y_j) - \max_{y \in \mathcal{Y}_c} f(x_j, y) \geq 1 - \epsilon_j^2 \qquad \text{for } j = 1, ..., n_2 \tag{4}$$

$$\epsilon_i^1 \geq 0 \qquad \text{for } i = 1, ..., n_1 \tag{5}$$

$$\text{and} \qquad \epsilon_j^2 \geq 0 \qquad \text{for } j = 1, ..., n_2 \tag{6}$$

where C_1 and C_2 ($\geq C_1$) are user-specified parameters, ϵ_i^1 and ϵ_j^2 are the slack variables, n_1 is number of examples for which the second best candidate label is a protein-specific label and $n_2 = n - n_1$ and Y_c is the candidate label set. These two new constraints (inequalities 3 and 4) allows the model to be more flexible towards picking second best candidate labels that are extensions of the current label. This allows it to use the information of the existing annotations to model the accumulation of annotations over time.

4.3. Experimental setup

We compare the performance of GOstruct 2.0 against the original GOstruct (denoted as 1.0) with respect to the CV, NA, and NP tasks on human proteins using sequence, network, and NLP features described in Section 3.5. Regular 5-fold cross-validation was used for evaluating performance in the CV task. In the NA task, methods are trained using the set of annotations acquired on or before the year 2009, and tested on the set of annotations gathered on the same set of proteins after 2009. An annotation that was added with a new evidence code, but present in the training set was not included in testing; similarly, we do not include an annotation in the test set when a more specific annotation already exists in the training set. In the NP task, methods are trained using the set of annotations acquired on or before the year 2009, while they are tested on the annotations acquired for proteins that were not annotated on or before 2009. We use $C_1 = 1$ and $C_2 = 10$ for all our experiments.

4.4. Results and Discussion

Figures 4.1, 4.2 and Tables 4.1, 4.2 show CV, NA and NP results in yeast and human for all three GO namespaces: molecular function, biological process and cellular component. First, as noted in section 3.7, performance for the NA task is much lower for both the species. We observe that GOstruct 2.0 outperforms GOstruct 1.0 in the NA task in all subontologies in human while it performs as well as or better than GOstruct 1.0 in molecular function and biological process subontologies in yeast. This demonstrates the ability of the new constraints to model the accumulation of annotations over time for improving performance in the NA task especially on human data.

The largest improvement for human happens in the biological process subontology (a jump from 0.58 to 0.71, see Figure 4.2). This observation can be attributed to the fact

that it is by far the deepest subontology [95] as well as it being the subontology with the highest rate of increase in the number of categories over the years [96] making biological process annotations more incomplete than the other two, thereby providing for more room for improvement.

In addition, we observe there is minor improvement in cross-validation in few cases for human as well (e.g. MF and CC). However, when it comes to the NP task, a decline in performance is observed in several cases (e.g. CC in yeast and MF in human) which warrants further investigation for the identification of the underlying cause. This suggests using GOstruct 2.0 only for already annotated proteins.

In general, improvement across all subontologies is shown for human on the NA task. That may be due to yeast annotations having a lower "annotation bias" compared to human [97] so that the added flexibility of the model does not help GOstruct 2.0 perform better on yeast as much as it does for human. It is also important to note that GOstruct 2.0 is able to produce AUCs above 0.7 for all three subontologies in human, which is clearly a major improvement over original GOstruct (which produced undesirable values of 0.58 and 0.66 in NA for BP and CC subontologies, respectively).

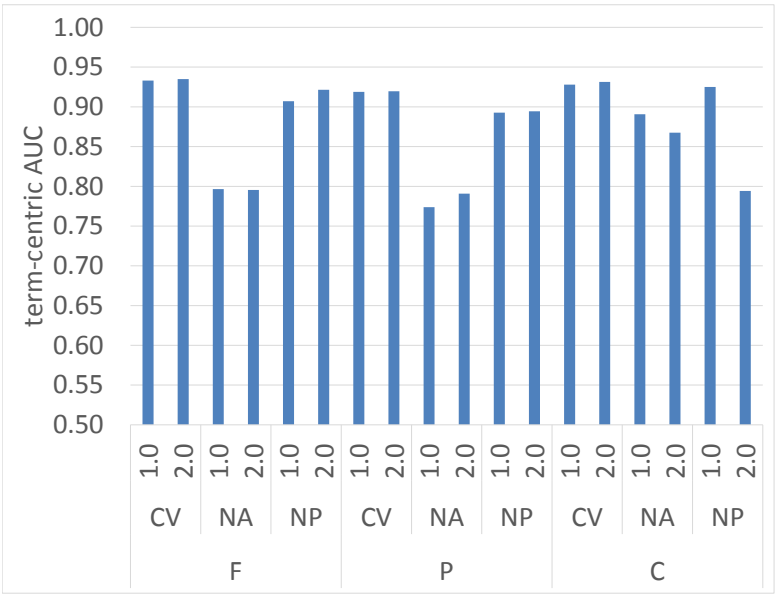

FIGURE 4.1. Performance comparison between GOstruct 2.0 and original GOstruct in CV, NA and NP for yeast. GOstruct 2.0 and original GOstruct (denoted as 1.0) are evaluated in CV (cross-validation), NA (novel-annotation) and NP (novel-proteins) on yeast. Performance is reported using the term-centric AUC in molecular function (F), biological process (P) and cellular component (C) subontologies.

TABLE 4.1. Performance comparison between GOstruct 2.0 and original GOstruct in CV, NA and NP for yeast. GOstruct 2.0 and original GOstruct (denoted as 1.0) are evaluated in CV (cross-validation), NA (novel-annotation) and NP (novel-proteins) on yeast. Performance is reported using the term-centric AUC in molecular function (F), biological process (P) and cellular component (C) subontologies.

		1.0		2.0	
Ontology	setup	AUC	Variance	AUC	Variance
F	CV	0.93	1.7E-04	0.93	3.0E-05
	NA	0.80	8.8E-04	0.80	9.1E-04
	NP	0.91	1.6E-02	0.92	1.4E-02
P	CV	0.92	1.2E-04	0.92	4.0E-05
	NA	0.77	5.4E-04	0.79	6.1E-04
	NP	0.89	1.6E-02	0.89	1.4E-02
C	CV	0.93	1.1E-05	0.93	3.1E-05
	NA	0.89	1.2E-04	0.87	1.9E-04
	NP	0.92	5.4E-03	0.79	2.6E-02

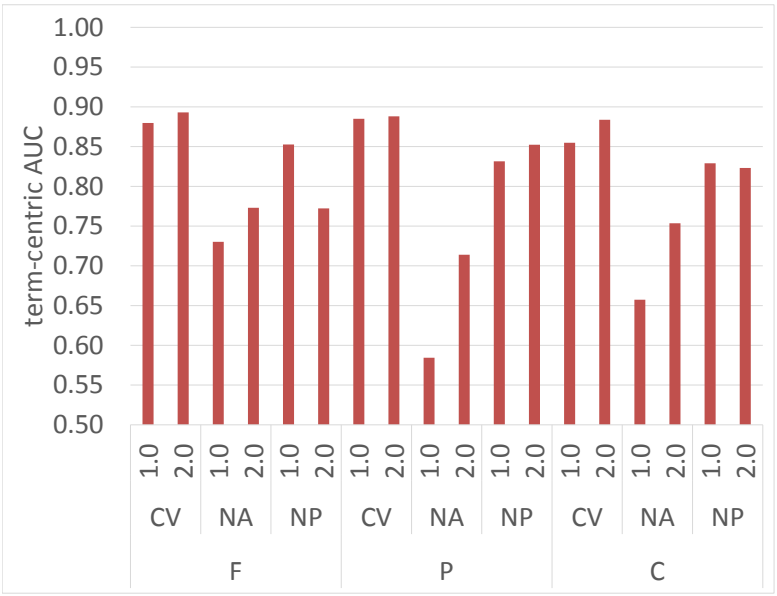

FIGURE 4.2. Performance comparison between GOstruct 2.0 and original GOstruct in CV, NA and NP for human. GOstruct 2.0 and original GOstruct (denoted as 1.0) are evaluated in CV (cross-validation), NA (novel-annotation) and NP (novel-proteins) on human. Performance is reported using the term-centric AUC in molecular function (F), biological process (P) and cellular component (C) subontologies.

TABLE 4.2. Performance comparison between GOstruct 2.0 and original GOstruct in CV, NA and NP for human. GOstruct 2.0 and original GOstruct (denoted as 1.0) are evaluated in CV (cross-validation), NA (novel-annotation) and NP (novel-proteins) on human. Performance is reported using the term-centric AUC in molecular function (F), biological process (P) and cellular component (C) subontologies.

		1.0		2.0	
Ontology	setup	AUC	Variance	AUC	Variance
F	CV	0.88	4.2E-04	0.89	2.0E-05
	NA	0.73	1.1E-03	0.77	3.5E-04
	NP	0.85	2.4E-02	0.77	2.9E-02
P	CV	0.89	5.8E-06	0.89	7.3E-06
	NA	0.58	1.5E-05	0.71	1.9E-04
	NP	0.83	2.0E-02	0.85	1.3E-02
C	CV	0.85	8.1E-04	0.88	4.1E-05
	NA	0.66	4.0E-04	0.75	1.6E-04
	NP	0.83	1.9E-02	0.82	1.7E-02

CHAPTER 5

Evaluating a Variety of Text-mined Features for Automatic Protein Function Prediction with GOstruct

5.1. Introduction

Biomedical literature is the most up-to-date resource of biological knowledge documenting experimental results. However, the speed of curation in to the relevant databases has not kept up with the rate of publication [38]. On top of that there is no clear consensus in how to use literature for best performance because there has never been an comprehensive study evaluating the effectiveness of literature data for AFP. In CAFA1, only two teams out of 54 that submitted predictions exclusively used literature-derived features [13]. Similarly, in CAFA2, only two teams (out of 38 that submitted predictions, excluding our team) used literature data [27].

In this study, we conduct the first ever comprehensive evaluation on the informativeness of literature sources and their effectiveness on the accuracy of protein function prediction. Specifically, our collaborators developed a natural language processing (NLP) pipeline that processes millions of biomedical literature to extract two different types of literature features, co-occurrences of protein-GO terms as well as bag-of-words features. We compare the two types and assess how to best combine them for achieving the highest accuracy.

5.2. Data

Document sources. If a protein is mentioned in the vicinity of a functional category in the literature (a *co-mention*) it is likely that this protein may be associated with that function. For example, the sentence "P-glycoprotein (Pgp) and multidrug resistance protein 1 (MRP1) are members of the ATP-binding cassette (ABC) family of transporter proteins." shows the relationship between the protein *MRP1* and the function *ATP-binding*. The sentence "The microtubule-dependent motor protein Eg5 plays a critical role in spindle assembly and maintenance in mitosis." shows the relationship between *Eg5* and *mitosis*. To leverage this source of information we extracted protein-functional category co-mentions using the natural language processing pipeline described in [50] from 13,530,032 Medline abstracts available on 23-Oct-13 and 595,010 full-text articles available from the PubMed Open Access Collection (PMCOA) on 06-Nov-13.

GO annotations. We obtained and prepared the GO annotations as described in the previous chapter. The final set of annotations consisted of 13,400 human proteins (with 11,000 categories).

Text-mining pipeline. A text-mining pipeline was developed by our collaborators (Chris Funk and Karin Verspoor at University of Denver) for extracting protein names and functional categories from the text [50]. A detailed description of the pipeline is given in Appendix B.

Literature features. We extracted two types of features from the text: *co-mentions* which represent a knowledge-based approach and *bag of words* which represents a knowledge-free approach. The co-mentions we considered are the pairs of protein names and GO terms

that co-occur in the document within a specified span. We used two spans: *sentence* and *non-sentence*. Sentence co-mentions only consider proteins and GO categories mentioned in the same sentence while non-sentence co-mentions consider such pairs mentioned together in the same paragraph or abstract, but not within the same sentence. The number of co-mentions extracted for human proteins are given in Table 5.1.

TABLE 5.1. Statistics of co-mentions extracted from both Medline and PMCOA .

Span	Unique proteins	Unique terms	Unique co-mentions	Total co-mentions
sentence	12,998	15,415	1,839,360	33,199,284
non-sentence	13,513	18,713	3,725,450	196,761,554
combined	13,536	18,920	3,897,951	229,960,838

Bag-of-words features are generated by associating proteins to all words in sentences in which this particular protein was mentioned [50]. All words were lower-cased and stop words were removed but no type of stemming (reducing words to their root form) or lemmatization (grouping together the different forms of a word so they can be analysed as a single item) was applied.

With co-mention features, a protein is represented by two vectors in which the ith element in each vector gives the number of times that protein is co-mentioned with the GO category i within either a sentence or non-sentence span. With bag-of-words features, a protein is represented as a vector in which the ith element gives the number of times that word was mentioned in any sentence the protein name was found in the text.

5.3. MODELS

The primary AFP method we use to assess the effectiveness of the literature features is the structured output method GOstruct [22]. As a baseline method we also utilized the

co-mentions themselves as a classifier (i.e., the co-mentions are used as the final predictions without any learning).

5.4. Evaluation

We use five-fold cross validation for assessing the performance. We report results using protein-centric F1, precision, recall and term-centric AUC as described in Section 3.4.

5.5. Results and Discussion

Exploring the use of co-mention features. We used four different strategies of using co-mention features, (1) SENT: only sentence co-mentions (2) NONS: only non-sentence co-mentions (3) COMB: combining counts from both types of features into one feature set and (4) BOTH: using both types of features at once but as separate feature vectors.

As illustrated in Table 5.2, the best strategy for using co-mention features is the last method (BOTH). Intuitively, since the two types of co-mentions (sentence and non-sentence) contain different types of information, having both at its disposal as two separate sets (as opposed to using only one or combining them in to one) helps the classifier make better predations.

However, against our intuition non-sentence co-mentions perform better than sentence co-mentions because in general it can be expected that a function of a protein is usually mentioned within the same sentence. However, previously it has been observed that function annotations may not necessarily occur within a single sentence [98]. Although *co-reference resolution* is required to relate entities across sentences, simply extracting protein-GO term pairs mentioned across sentences (i.e., within abstracts or paragraphs) still seems to be a useful approximation according to our results. Based on our observations, we exclusively use

TABLE 5.2. Performance from co-mention features. Performance is reported for (1) SENT: only sentence co-mentions (2) NONS: only non-sentence co-mentions (3) COMB: combining counts from both types of features into one feature set and (4) BOTH: using both types of features at once but as separate feature vectors. Variance is computed on term-centric AUC.

Subont.	Features	$F1^{pc}$	P^{pc}	R^{pc}	AUC^{tc}	Variance
F	COMB	0.38	0.38	0.38	0.76	2.2E-04
	BOTH	0.39	0.30	0.53	0.77	1.9E-04
	SENT	0.36	0.32	0.46	0.74	2.6E-04
	NONS	0.37	0.37	0.37	0.75	1.2E-04
P	COMB	0.43	0.43	0.43	0.74	5.1E-05
	BOTH	0.42	0.43	0.42	0.75	6.2E-05
	SENT	0.41	0.41	0.41	0.72	4.0E-05
	NONS	0.43	0.43	0.42	0.74	8.0E-05
C	COMB	0.58	0.61	0.56	0.73	1.5E-04
	BOTH	0.59	0.59	0.59	0.74	1.3E-04
	SENT	0.57	0.58	0.56	0.70	1.1E-05
	NONS	0.58	0.60	0.56	0.73	3.5E-05

the last strategy (i.e., BOTH) for all our following experiments – in other words all mentions of "co-mention features" refer to using both types of features as separate sets throughout the rest of this chapter.

COMPARISON OF CO-MENTIONS AND BAG OF WORDS FEATURES. First we look at the performance of the baseline method for understanding how informative co-mentions are by themselves. Table 5.3 shows performance of baseline method from combining counts between sentence and non-sentence co-mentions. It is interesting to see that the baseline leads to very low precision for all branches but impressive levels of recall can be observed. This shows that we are able to recall reasonable amounts of annotations directly from text which attests to the effectiveness of our pipeline in extracting protein names and functions. However we are also extracting many false positive which leads to the poor precision. This suggests that some form of filtering or classification of co-mentions would improve their quality; see section 6 on how we addressed this problem by modeling it as a relation extraction task.

However, surprisingly, BoW features outperform co-mention features in all cases given that it is a knowledge-free approach which uses all words associated with a protein. One main reason could be that biomedical literature contain many other words/features informative of a protein function other than the direct mentions. BoW features utilizes a lot more literature because it only relies on a protein to be recognized while co-mention features capture information only when both a protein and function is mentioned together – on average a protein has 2,375 BoW features while there is only 385 co-mention features per protein. Moreover, the quality of co-mention features is highly dependent on the accuracy of the automatic GO concept recognition, which has been shown to be a challenging task in itself [99]. As one might expect, the combination of both co-mentions and BoW features provides the best performance for all cases suggesting that the two types of features are complementary to each other.

TABLE 5.3. Performance comparison between co-mention and BoW features. Performance is reported for (1) Baseline: using co-mentions themselves as final predictions without any machine learning (2) Co-mentions: only co-mentions features (3) BoW: only bag-of-words features (4) co-mentions+BoW: using both types of features at once. Variance is computed on term-centric AUC.

Subont.	Features	$F1^{pc}$	P^{pc}	R^{pc}	AUC^{tc}	Variance
F	Baseline	0.06	0.04	0.32	0.68	1.4E-04
	Co-mentions	0.38	0.34	0.45	0.76	1.0E-04
	BoW	0.39	0.38	0.41	0.77	2.8E-05
	co-mentions+BoW	0.41	0.35	0.49	0.79	8.2E-05
P	Baseline	0.15	0.10	0.31	0.61	2.1E-04
	Co-mentions	0.43	0.43	0.43	0.75	8.8E-05
	BoW	0.46	0.47	0.45	0.77	1.1E-04
	co-mentions+BoW	0.46	0.43	0.51	0.78	1.2E-04
C	Baseline	0.07	0.04	0.32	0.64	6.3E-05
	Co-mentions	0.59	0.58	0.60	0.75	1.8E-04
	BoW	0.61	0.59	0.62	0.76	2.9E-05
	co-mentions+BoW	0.61	0.59	0.62	0.77	6.7E-05

Literature versus other data. In order to identify the informativeness of literature features among other popular data sources used for protein function prediction, we conducted several ablation studies where we provided a single source of data at a time and the combination of all sets of features to GOstruct using human proteins. Table 5.4 shows five-fold cross-validation results in human for all three subontologies. Our results show that even though literature data provides much better performance than sequence data there is definitely room for improvement.

Table 5.4. Performance comparison of literature features to others. Performance is measured using the term-centric AUC measure. Results are shown for each source of data: Trans/Loc (transmembrane and localization signals); homology; functional association data; literature mining data; and the model that combines all features together.

Subontology	Features	AUC^{tc}	Variance
F	Trans/Loc	0.68	1.6E-04
	homolgy	0.85	2.9E-05
	network	0.88	1.7E-05
	literature	0.77	1.9E-04
	combined	0.89	4.2E-04
P	Trans/Loc	0.64	1.6E-04
	homolgy	0.84	3.7E-05
	network	0.86	1.7E-05
	literature	0.75	7.5E-05
	combined	0.88	5.8E-06
C	Trans/Loc	0.69	6.7E-05
	homolgy	0.81	5.3E-05
	network	0.84	1.5E-05
	literature	0.75	1.3E-04
	combined	0.86	8.1E-04

Validating false positives. Due to various factors such as slowness of the curation process, the Gene Ontology database is incomplete [38]. This leads to performance estimates that underestimate the true performance of a classifier [100] because the set of GO annotations we considered as the gold standard does not fully represent all the functions that should be associated with the currently annotated genes. To explore this issue, we randomly

selected several predictions made by GOstruct which were considered false positives according to the current gold standard and looked for evidence in the current biomedical literature that can be used as evidence for those predictions.

For example, protein GCNT1 (Q02742) was predicted to be involved with carbohydrate metabolic process (GO:0006959). In PMID:23646466 we find "Genes related to carbohydrate metabolism include PPP1R3C, B3GNT1, and GCNT1...". Similarly, protein CERS2 (Q96G23) was predicted to play a role in ceramide biosynthetic process (GO:0046513). In PMID:22144673 we see "...CerS2, which uses C22-CoA for ceramide synthesis...". These examples suggest that it is very likely the performance of literature features are underestimated. But even more importantly, these examples clearly shows how literature features can be used not only as input features but also as means of validating predictions – this further boosts the importance of literature features since other features cannot be used for validation in this way.

CHAPTER 6

Extracting protein-GO relations from biomedical text using graph kernels

6.1. Introduction

As mentioned in the previous chapter, the performance of the baseline method which uses co-mentions themselves as the final predictions (i.e. without any learning) suggested that the text mining pipeline is able to extract useful co-mentions but includes high number of false positives. A filter or a classifier that could identify "good" co-mentions can be used in place of an AFP method, and can be used by annotators to significantly speed up the curation process. In addition, this can be used to provide much higher quality co-mentions as input to AFP methods, which would likely lead to better predictions. In view of this, we propose to develop a co-mention classifier that can be used as a filter to identify "good" co-mentions.

We define the problem of co-mention classification as the task of classifying whether a given co-mention of a protein and a function suggests a functional relationship between the two entities according to the context surrounding them (Figure 6.1 depicts a positive co-mention according to this definition). Therefore this can also be identified as the task of extracting protein-function relations from text. In this work we formulate this as a supervised learning problem as described below.

6.2. Methods

Problem definition. We formulate this problem as a classification task as follows. Given a context $C = w_1 w_2 .. e_1 .. w_3 .. e_2 .. w_{n-1} w_n$ composed of words w_i and the two entities e_1

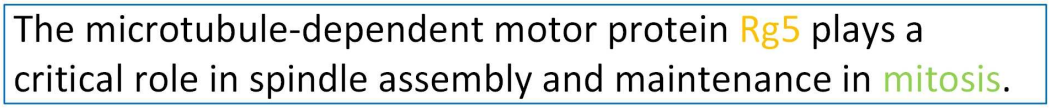

FIGURE 6.1. An example co-mention. The protein and the function are highlighted in orange and green, respectively. This is a positive example of a protein-function relation since the meaning of the sentence conveys that "Rg5" is associated with "mitosis".

and e_2, we define a mapping $f_R(\cdot)$ as:

$$f_R(T(C)) = \begin{cases} 1 & \text{if } e_1 \text{and } e_2 \text{ are related according to } R \\ 0 & \text{otherwise,} \end{cases}$$

where $T(C)$ is a structured representation of the context, e_1 and e_2 are the entities representing the protein and R is the relation that represents the protein-function relationship between the two. An example is considered a positive example if the meaning of the context suggests that the protein mentioned has this function.

In this work, the context C is a single sentence (i.e., the sentence containing the mentions of the two entities). Figure 6.1 depicts a sentence which is labeled as a positive example (i.e., $f_R = 1$) because it provides evidence for the functional relationship between the two entities Eg5 (protein) and mitosis (function). We model this problem as a supervised learning problem and use a binary SVM as the classifier for learning f_R. The PyML[1] package is used for implementing the SVM functionality.

THE RANDOM WALK KERNEL BASED METHOD. Here we propose a novel co-mention classification method based on random walk kernels [101]. In this method, we represent contexts (i.e., sentences) as *labeled* graphs in which nodes represent words and edges represent

[1]http://pyml.sourceforge.net

dependencies between words (i.e., the structured representation of context is a graph). Then we use the random walk kernel for measuring the similarity between labeled graphs which computes similarity by calculating the number of matching walks between the given pair of graphs [101].

Pre-processing and generation of labeled graphs. For generating the labeled graphs, we first preprocess the context C in two ways. We use the Stanford parser [1] for generating the typed dependency graph in which nodes are words and directed edges are dependencies between words. There are 52 types of dependencies used by the Stanford parser (Figure 6.2 shows the typed dependency graph generated for the sentence in Figure 6.1).

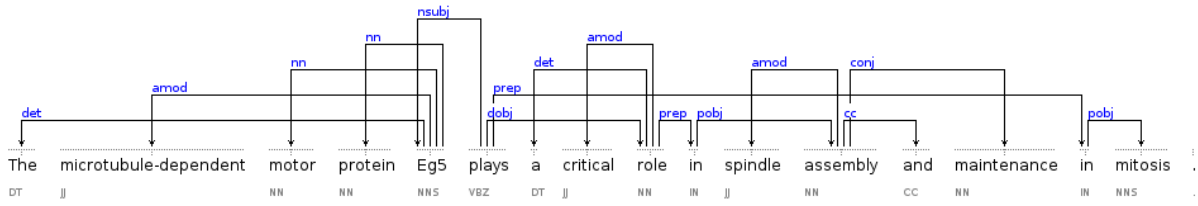

FIGURE 6.2. An example typed dependency graph. This is generated for the sentence in Figure 6.1 using the Stanford parser [1].

We then represent the context as an undirected graph in which nodes represent the words. A node in the labeled graph has two attributes, a type and a label. The type attribute can take the values of P: protein, F: function and O: other. The label attribute holds the actual lexical (string) value of the word represented by the node. Then, for each directed edge in the Stanford dependency graph, we add an undirected edge between the two corresponding nodes to this undirected graph. We also add an undirected edge between consecutive words within the context if there isn't an edge already present previous step. An undirected edge of this graph has only one attribute, a label; it holds the Stanford dependency type associated

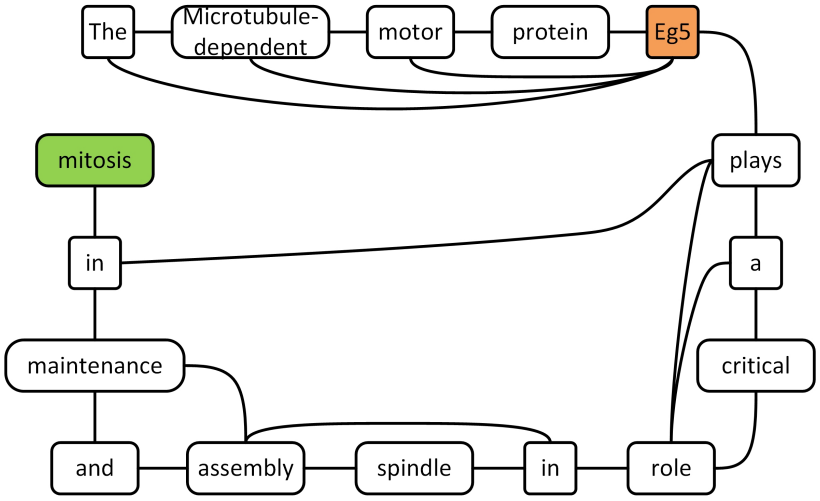

FIGURE 6.3. An example labeled graph. This is generated for the sentence in Figure 6.1. The nodes representing the protein and the function entities are in orange and green colors, respectively.

with the dependency edge between the two words represented by the two nodes. Figure 6.3 depicts the labeled graph generated for the sentence in Figure 6.1.

Formulation of the random walk kernel. For measuring the similarity between these labeled graphs we use a random walk kernel (RWK) [102]. The random walk kernel counts the number of matching labeled walks (of up to length l) between the two given labeled graphs. The intuition behind this is the fact that a higher number of matched walks between the graphs implies more similarity between the corresponding contexts. Kernel values are normalized using cosine normalization.

Suppose the labeled graph G is composed of nodes $v \in V$ and edges $e \in E$. Given two labeled graphs G_1 and G_2, the random walk kernel is defined as,

$$k_{graph}(G_1, G_2) = \begin{cases} \sum_{w_1 \in G_1} \sum_{w_2 \in G_2} k_{walk}(w_1, w_2) & \text{if } w_1, w_2 \text{ begin/end with a node of type P or F} \\ 0 & \text{otherwise} \end{cases}$$

where the kernel between two walks is denoted by k_{walk}. Note that walks not containing at least one of the entities (i.e., protein/function) are ignored. The reason for considering only the paths that go through the nodes containing the two entities is because it has been shown that the paths between the two entities contain the most informative features for other relation extraction tasks [103].

k_{walk} is defined using a kernel between steps (k_{step}) as,

$$k_{walk}(walk_1, walk_2) = \prod_{i=1}^{n-1} k_{step}((v_i, v_{i+1}), (w_i, w_{i+1}))$$

where n is the length of the walk and (v_i, v_{i+1}) and (w_i, w_{i+1}) are the edges belonging to $walk_1$ and $walk_2$, respectively. k_{step} is defined as,

$$k_{step}((v_i, v_{i+1}), (w_i, w_{i+1})) = k_{node}(v_i, w_i) \cdot k_{node}(v_{i+1}, w_{i+1}) \cdot k_{edge}((v_i, v_{i+1}), (w_i, w_{i+1}))$$

where k_{node} and k_{edge} are node and edge kernels, respectively. Those two kernels are defined as,

$$K_{node}(v_i, w_i) = \begin{cases} E_1(v_i) \cdot E_1(w_i) & \text{if type}(v_i) == \text{type}(w_i) \\ 0 & \text{otherwise} \end{cases}$$

$$k_{edge}((v_i, v_{i+1}), (w_i, w_{i+1})) = E_2(v_i, v_{i+1}) \cdot E_2(w_i, w_{i+1})$$

where E_1 and E_2 are word and edge embeddings (see below). When the type attributes of the two nodes do not match they are considered completely dissimilar (i.e., a value of zero); this is for ensuring that the entities (protein or function) are not considered similar to other nodes (even if the literal values of the correspond words may be similar).

Word embeddings. Word embeddings are vector representations of words [104]. They map words to lower (relative to vocabulary size) dimensional vectors. The intuition is that a pair of words that appear more frequently together are likely to have a similar meaning. We also generate edge embeddings using an analogous process. Again, the intuition is that two typed dependencies that connect to a common word many times are likely to be semantically more similar than others. If the actual word was used instead of word embeddings there will be too few matches (and sometimes none at all) when computing the node kernel. The same reasoning is applicable to using edge embeddings. By using the above definitions for node/edge kernels we capture the meaning of the contexts; the walk kernel for two walks with semantically more similar words and edges (even though the words or edge types do not match exactly) will have a higher value. In other words, through these definitions of node/edge kernels we have adapted the random walk kernel to the domain of (protein-function) relation extraction.

In this work we use Word2vec [105] as the word embedding generation method. The intuition behind Word2vec is that the words with similar distributional properties (i.e., that co-occur regularly) tend to share some aspect of semantic meaning. Or in other words "a word can be characterized by the company it keeps" [106]. To this end, Word2vec learns the probability of a context word occurring given a target word using a large corpus of text. As the model converges, lower dimensional continuous-space vector representations of target words are produced; the vectors for two words that are semantically similar are located closer in this lower dimensional space compared to that of two random words. It uses a simplified version of a feed forward neural network with back-propagation using stochastic gradient descent for learning the model [105].

Node and edge kernels. As mentioned above, the random walk kernel computes similarity between two graphs using node and edge kernels that in turn compute the similarities between the pairs of nodes/edges that make up the corresponding walks. We use the following three different node kernels: (a) elm - exact match between node labels (b) pos - similarity between part-of-speech (POS) tags such as Noun, Verb etc. and (c) w2v - word embedding similarity. We also use the following four edge kernels: (a) elm - exact match between edge labels (b) plm - partial label match between edge labels (e.g. nsubj and nsubjpass are considered partially similar) (c) w2v - edge embedding similarity and (d) con - uses an edge kernel that is always equal to 1.

Computation of the random walk kernel. As mentioned above the random walk kernel computes the similarity between two given graphs by calculating the similarity between all common walks that goes through the two entity nodes. The approach used for computing the kernels is through the use of a direct product graph [102]. The direct product graph $G_1 \times G_2$ of two labeled graphs $G_1 = (V, E)$ and $G_2 = (W, F)$ is defined as the graph $(V_\times, E_\times)$ with

$$\begin{aligned} V_\times(G_1 \times G_2) &= \{(v_1, w_1) \in V \times W : (type(v_1) = type(w_1))\} \\ E_\times(G_1 \times G_2) &= \{((v_1, w_1), (v_2, w_2)) \in V^2(G_1 \times G_2) : \\ &\qquad (v_1, w_2) \in E \wedge (w_1, w_2) \in F \wedge (type(v_1, v_2) = type(w_1, w_2))\}. \end{aligned}$$

Based on this direct product graph, the random walk kernel is defined as,

$$K_{graph}(G_1, G_2) = \sum_{i=p,f} \sum_{j=1}^{V_\times} \left[\sum_{n=0}^{l} \lambda^n A_\times^n \right]_{ij}$$

where $A_\times$ denotes the adjacency matrix of the direct product graph $G_\times$, p and f are the indexes of the two nodes representing the entities, l is maximum length of walks considered and λ is the decay factor (parameter which gives relatively lower significance to longer walks).

6.2.1. Other approaches. For the purpose of comparison we also implement several other approaches: (a) bag-of-words methods (b) word embedding kernel based methods and (c) graphlet kernel based methods.

Bag-of-words methods. In particular, we use three bag-of-words methods, (1) BoW: regular bag-of-words features (2) BoWr: bag-of-words features generated from three non-overlapping regions separated by the two entities (i.e., before, between, after) and (3) BoWn: bag-of-words features generated within a neighborhood of the two entities (window size = 3). Features are normalized using L1 normalization. We use a Gaussian kernel with the bag-of-words features.

Word embedding kernel based methods. Given N-dimensional (N = size of the vocabulary) vectors a and b representing the contexts of two co-mentions, the word embedding kernel is defined as,

$$K_{we}(a, b) = \sum_{i}^{N} \sum_{j}^{N} s_{i,j} a_i b_j$$

where similarity $s_{i,j}$ between words is defined using their word embedding vectors as,

$$s_{i,j} = E_1(word_i) \cdot E_1(word_j).$$

The intuition behind using word embedding kernel is to accommodate the relative similarities between features (i.e., words) in contrast to bag-of-words approach in which features are considered orthogonal dimensions (i.e., only exact match between words is considered).

This form of similarity measure is also referred to as the *soft similarity* and has been used effectively for the task of automated question answering by utilizing the Levenshtein distance between words [107] as the underlying similarity measure [108].

We implement three variants that use the word embedding kernel, (1) kernel defined on all words of the context (2) kernel defined on words from three non-overlapping regions separated by the two entities (i.e., before, between and after) and (3) kernel defined on words within a neighborhood of the two entities (window size = 3). We use Word2vec [105] for generating the word embeddings as discussed earlier. The kernel is normalized using cosine normalization.

Graphlet kernel based methods. Graphlet kernels compute similarity between two graphs by counting the number of common graphlets (subgraphs of k nodes) between them [109]. In terms of graphlet kernel methods, each sentence is represented by an undirected labeled graph in which nodes represent words and edges represent Stanford dependencies between words. Each node has a label property. Node labels are P (Protein), F (Function) or O (other). In particular, we use the two following variants, (1) Standard Graphlet Kernel (SGK) computes the number of common graphlets of size 3,4,5 and (2) Edit Distance Graphlet Kernel (EDK) [110] computes the number of common graphlets of size 3,4,5 with 1-vertex label mismatch and 1-edge indels allowed. Kernels are normalized using cosine normalization. These kernels are implemented using the Graphlet Kernel Framework package [110].

6.2.2. Data. In this study our data are composed of manually annotated co-mentions. These datasets were annotated by a human annotator utilizing the pipeline described elsewhere [50]. Each example (i.e., co-mention) is composed of the two entities (i.e., protein and the function), the context surrounding the two entities and the class label. The class label

TABLE 6.1. Co-mention datasets. The column titled 'categories' shows either the number of categories (for the MIXED dataset) or the name of the GO category (rest of the datasets) and 'ont' is the subontology of the GO category. The column 'depth' is the depth of the GO category. The columns 'examples', 'pos','%pos' and 'proteins' show the number of examples, number of positive examples, percentage of positive examples, and number of unique proteins associated with co-mentions, respectively.

dataset	categories	ont.	depth	examples	pos	%pos	proteins
MIXED	539 categories	N/A	N/A	1460	222	15%	1052
DRUGB	drug-binding	MF	3	202	88	44%	118
REGST	reg. of signal transduction	BP	5	214	101	47%	160
TRMTR	transmembrane transporters	MF	7	357	157	44%	231

is positive if the context suggests the protein is associated with the function and negative otherwise. Specifically, there are four datasets (1) MIXED: randomly selected co-mentions covering a variety of GO categories (2) DRUGB: co-mentions specific to the "drug binding" category (GO:0008144, molecular function subontology) (3) REGST: co-mentions specific to the "regulation of signal transduction" category (GO:0009966, biological process subontology) and (4) TRMTR: co-mentions specific to the "transmembrane transporter activity" category (GO:0022857, molecular function subontology). See Table 6.1 for the statistics on these datasets.

6.3. Experimental setup

Performance of above methods were evaluated through a customized nested cross-validation procedure (folds = 5) which also ensures that examples pertaining to the same protein are not split across train/test folds. This is to negate the bias that may be introduced by having co-mentions associated with the same protein across train/test folds which may make it unduly easier for the classifier to make better predictions. The parameter values used with the bag-of-words methods are: (1) C: 0.1, 1, 10, 100, 1000 and (2) γ: 0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10. The parameter values used with the word embedding kernel based methods

are 0.1, 1, 10, 100, 1000 (C). The parameter values used with the random walk kernel based methods are (1) l: 5, 10 (2) λ: 0.1, 0.5, 0.9 and (3) k : 1, 5. The parameter values used with the graphlet kernel methods are 0.1, 1, 10, 100, 1000 (C). In terms of Word2Vec we used values of 300, 10, 100 and 1e-5 for the embedding size, window size, min count and max cutoff, respectively.

6.4. Results and Discussion

Performance across datasets. The best area under the ROC curves (AUCs) observed for the MIXED and DRUGB dataset are above 70% while for REGST and TRMTR show AUCs above 80% (see Tables 6.2 - 6.5). This suggests that our protein-function relation extraction approach shows high promise as a method for complementing bio-curators. It is interesting to note that the two datasets associated with deeper GO categories (i.e., REGST and TRMTR) exhibit higher performance than DRUGB (labeled with a relatively shallow GO category) and the MIXED (labeled with multiple GO categories). It is understandable that most methods would have relatively lower performance on the MIXED dataset due to having co-mentions labeled with many GO categories. On the other hand, it is also not that surprising to observe that majority of methods perform worse on DRUGB dataset in general (Table 6.3). Performance of GOstruct on these individual GO categories in human (Table 6.7) suggest that drug binding category tends to be associated with relatively lower accuracy levels than the other two GO categories considered in this study.

Comparison of methods. In terms of overall comparison of performance among methods, the word embedding methods are the best performers; these methods produce the best performance for all datasets except DRUGB for which bag-of-words methods are the better performers (see Table 6.3). However, the performance of random walk kernel based

TABLE 6.2. Performance comparison on the MIXED dataset. Methods compared are (a) bag-of-words methods: BoW (regular bag-of-words features), BoWr (bag-of-words features in three separate regions defined by the two entities) and BoWn (bag-of-words features in the neighbourhood of the two entities, window size = 3) (b) word embedding methods: WoE (embeddings of all words), WoEr (embeddings of words in three separate regions defined by the two entities) and WoEn (embeddings of words in the neighbourhood of the two entities, window size = 3) (c) random walk kernel methods: denoted as RWK(node kernel, edge kernel) where elm, plm, pos, w2v, none, represent exact label match, partial label match, POS tag similarity, word embedding similarity and constant kernel in which value is always equal to 1 (d) graphlet kernel methods: GK(sgk) and GK(edk) represent the standard graphlet kernel and edit distance based graphlet kernel, respectively. Results are reported using area under the ROC curve (AUC), F1, Precision (P), and Recall (R). Variance is computed on AUC.

Method	AUC	F1	P	R	Variance
BoW	0.65	0.26	0.19	0.41	4.0E-04
BoWr	0.62	0.18	0.13	0.29	6.5E-04
BoWn	0.60	0.18	0.21	0.16	4.8E-04
WoE	0.73	0.40	0.28	0.70	2.9E-04
WoEr	0.73	0.39	0.28	0.62	5.3E-04
WoEn	0.60	0.22	0.16	0.37	3.0E-04
RWK(elm, elm)	0.65	0.30	0.32	0.29	5.1E-04
RWK(elm, plm)	0.65	0.30	0.32	0.29	5.1E-04
RWK(elm, con)	0.65	0.28	0.29	0.27	4.2E-04
RWK(pos, elm)	0.60	0.32	0.26	0.43	6.3E-04
RWK(pos, plm)	0.60	0.32	0.26	0.42	6.3E-04
RWK(pos, con)	0.58	0.26	0.19	0.38	5.4E-04
RWK(w2v, elm)	0.69	0.35	0.36	0.34	5.0E-04
RWK(w2v, plm)	0.69	0.35	0.36	0.33	5.0E-04
RWK(w2v, con)	0.71	0.39	0.37	0.42	3.2E-04
RWK(w2v, w2v)	0.70	0.35	0.35	0.36	4.9E-04
GK(sgk)	0.60	0.28	0.20	0.50	5.2E-04
GK(edk)	0.61	0.30	0.21	0.49	5.3E-04

methods are comparable to that of word embedding kernel based methods on the MIXED dataset (in terms of AUC); on top of that, they provide the highest precision for the same dataset (see Table 6.2). Not only that, they consistently produce the best precision values for DRUGB dataset as well (see Table 6.3). Graphlet kernel based methods seem to be the lowest performing methods across all the datasets except DRUGB.

TABLE 6.3. Performance comparison on the DRUGB dataset. Please refer to Table 6.2 for an explanation of column headers.

Method	AUC	F1	P	R	Variance
BoW	0.53	0.48	0.37	0.71	2.1E-03
BoWr	0.70	0.45	0.35	0.64	1.1E-03
BoWn	0.59	0.46	0.40	0.53	2.5E-03
WoE	0.58	0.42	0.33	0.56	2.5E-03
WoEr	0.57	0.46	0.43	0.49	1.7E-03
WoEn	0.63	0.50	0.45	0.55	1.8E-03
RWK(elm, elm)	0.56	0.47	0.45	0.50	1.9E-03
RWK(elm, plm)	0.56	0.47	0.45	0.50	1.9E-03
RWK(elm, con)	0.62	0.49	0.42	0.57	1.6E-03
RWK(pos, elm)	0.60	0.47	0.47	0.47	2.1E-03
RWK(pos, plm)	0.61	0.47	0.47	0.47	2.1E-03
RWK(pos, con)	0.60	0.43	0.41	0.44	2.1E-03
RWK(w2v, elm)	0.59	0.50	0.51	0.49	1.7E-03
RWK(w2v, plm)	0.58	0.50	0.51	0.49	1.8E-03
RWK(w2v, con)	0.61	0.46	0.47	0.46	1.6E-03
RWK(w2v, w2v)	0.59	0.51	0.48	0.54	1.7E-03
GK(sgk)	0.67	0.54	0.49	0.59	1.6E-03
GK(edk)	0.66	0.55	0.51	0.61	1.5E-03

TABLE 6.4. Performance comparison on the REGST dataset. Please refer to Table 6.2 for an explanation of column headers.

Method	AUC	F1	P	R	Variance
BoW	0.74	0.44	0.63	0.34	7.3E-04
BoWr	0.80	0.54	0.68	0.44	8.1E-04
BoWn	0.78	0.47	0.66	0.36	8.9E-04
WoE	0.71	0.60	0.64	0.56	1.3E-03
WoEr	0.74	0.62	0.67	0.57	8.6E-04
WoEn	0.84	0.71	0.72	0.70	1.7E-03
RWK(elm, elm)	0.76	0.69	0.59	0.82	1.7E-03
RWK(elm, plm)	0.76	0.69	0.59	0.82	1.7E-03
RWK(elm, con)	0.76	0.69	0.59	0.81	1.5E-03
RWK(pos, elm)	0.74	0.68	0.57	0.83	1.1E-03
RWK(pos, plm)	0.74	0.68	0.58	0.84	1.1E-03
RWK(pos, con)	0.72	0.68	0.61	0.76	1.2E-03
RWK(w2v, elm)	0.76	0.69	0.57	0.86	1.4E-03
RWK(w2v, plm)	0.76	0.68	0.57	0.86	1.4E-03
RWK(w2v, con)	0.77	0.69	0.59	0.84	1.4E-03
RWK(w2v, w2v)	0.76	0.69	0.57	0.86	1.5E-03
GK(sgk)	0.71	0.63	0.65	0.60	8.0E-04
GK(edk)	0.71	0.63	0.63	0.62	1.1E-03

TABLE 6.5. Performance comparison on TRMTR dataset. Please refer to Table 6.2 for an explanation of column headers.

Method	AUC	F1	P	R	Variance
BoW	0.64	0.49	0.43	0.58	1.3E-03
BoWr	0.79	0.65	0.61	0.69	4.9E-04
BoWn	0.71	0.56	0.57	0.55	8.2E-04
WoE	0.65	0.49	0.49	0.50	9.4E-04
WoEr	0.80	0.65	0.63	0.68	6.6E-04
WoEn	0.81	0.68	0.62	0.74	7.8E-04
RWK(elm, elm)	0.79	0.64	0.61	0.67	8.0E-04
RWK(elm, plm)	0.79	0.64	0.61	0.67	8.1E-04
RWK(elm, con)	0.78	0.64	0.62	0.66	8.1E-04
RWK(pos, elm)	0.71	0.56	0.58	0.55	6.9E-04
RWK(pos, plm)	0.71	0.56	0.58	0.55	6.6E-04
RWK(pos, con)	0.65	0.51	0.47	0.56	6.1E-04
RWK(w2v, elm)	0.80	0.67	0.64	0.69	8.2E-04
RWK(w2v, plm)	0.80	0.66	0.64	0.69	8.1E-04
RWK(w2v, con)	0.79	0.65	0.62	0.68	7.0E-04
RWK(w2v, w2v)	0.80	0.67	0.64	0.71	7.4E-04
GK(sgk)	0.62	0.48	0.45	0.51	5.3E-04
GK(edk)	0.65	0.50	0.47	0.54	5.6E-04

BAG-OF-WORDS AND WORD EMBEDDING KERNEL BASED METHODS. In general, word embedding kernel based methods outperform the bag-of-words methods across all datasets except DRUGB. This demonstrates that the soft similarity between words is better for protein-function relation extraction than exact similarity and that Word2vec is able to generate high quality word embeddings that accurately capture the underlying semantics. Among all word embedding methods, the variant that uses the words in the vicinity of the two entities (i.e., WoEn) is the best performer for all datasets except MIXED.

NODE KERNELS AND EDGE KERNELS. The random walk kernel based methods that use word embedding similarity as the node/edge kernels (i.e., w2v kernel) provide the best performance among all RWK methods across all data sets. This suggests that the usage of semantic similarity is more effective than exact word matching or syntactical similarity (i.e., POS tags) for the task of protein-function relation extraction.

Random walk kernels vs. graphlet kernels. For all datasets except DRUGB, the random walk kernel methods clearly outperform the graphlet kernel methods, demonstrating that the similarity between walks may be more informative than graph isomorphism for the task of protein-function relation extraction. There is no apparent difference between the performance of the standard graphlet kernels (sdk) and the edit distance graph kernels (edk) for any of the datasets.

Overall results. In order to further assess the effectiveness of the performance of proposed relation extraction methods we compared their best performances in the three co-mention datasets labeled with a single GO categories (DRUGB: drug binding, REGST: regulation of signal transduction and TRMTR: transmembrane transporter activity) against the performance of GOstruct on those same GO categories for human data (Tables 6.6 and 6.7). First, it is important to note that these results are not directly comparable because (1) co-mention datasets are associated with multiple species (see Table 6.6 for the proportion of human proteins in these datasets) and (2) GOstruct is provided with all annotations belonging to many GO categories as input. However, it is still encouraging to observe that relation extraction methods are able to produce comparable levels of accuracy not just to GOstruct that uses sentence-level literature data but to GOstruct that uses all data sources (for human). This is especially impressive because the relation extraction methods use only the knowledge of the individual GO categories as opposed to GOstruct which uses all annotations as mentioned above. On top of that, the predictions made by the relation extraction methods can be combined with their corresponding co-mentions so that the end-user (preferably a bio-curator) is able to first prioritize a set of co-mentions with best scores and then can validate them using the associated co-mentions. Our relation

TABLE 6.6. Overall best AUCs for single GO category datasets. %human shows the percentage of human proteins in each dataset.

dataset	% human	overall best AUC
DRUGB	58%	0.70
REGST	43%	0.84
TRMTR	30%	0.81

TABLE 6.7. Performance of GOstruct on individual GO categories in human. Performance is reported using AUC for the GO categories associated with DRUGB, REGST and TRMTR datasets. The 'all-data' and 'lit-sent' columns show results when using all data sources (i.e., homology, sequence, functional association and text mining features) and only sentence-level co-mentions as input to GOstruct, respectively.

GO category	all data	lit-sent
drug binding	0.72	0.67
regulation of signal transduction	0.85	0.71
transmembrane transporter activity	0.96	0.83

extraction methods produce precision and recall values in the range of 0.5 - 0.6 for the REGST and TRMTR datasets while producing precision and recall levels of 0.4 - 0.5 and 0.3 - 0.4 for the DRUGB and MIXED datasets, respectively. This is clearly below the reported accuracy levels for state-of-the-art protein-protein interaction relation extraction systems where values in the range of 0.7 - 0.8 have been reported [111]. However as described above, extraction of protein-function relations is a much harder problem than extracting protein-protein interactions due to the complex nature of sentences that provide evidence for functional relationships between a protein and a functional category.

CHAPTER 7

PHENOstruct: Prediction of Human Phenotype Ontology Terms Using Heterogeneous Data Sources

7.1. Introduction

The human phenotype ontology (HPO) is a standardized vocabulary that describes the phenotype abnormalities encountered in human diseases [8]. It was populated using databases of human genes and genetic disorders such as OMIM: Online Mendelian Inheritance in Man [112], Orphanet [113] and DECIPHER: DatabasE of Chromosomal Imbalance and Phenotype in Humans using Ensembl Resources [114], and was later expanded using literature curation. The hierarchical structure of the HPO is very similar to that of the Gene Ontology (GO) [7], and it too has the structure of a directed acyclic graph (DAG); like GO, more general terms are found at the top, and term specificity increases from the root to the leaves. This implies the "true-path rule": whenever a gene is annotated with a given term, that implies all its ancestor terms.

The organ abnormality, mode of inheritance and onset and clinical course subontologies are composed of ∼10000, 25, and 30 terms respectively. Throughout this paper, the organ abnormality, the mode of inheritance and the onset and clinical course subontologies will be referred to as the Organ subontology, Inheritance subontology and Onset subontology, respectively.

The HPO web site[1] provides gene-disease-HPO annotations that can be used for research involving human diseases. Over 50,000 annotations of hereditary diseases are available at

[1] human-phenotype-ontology.org

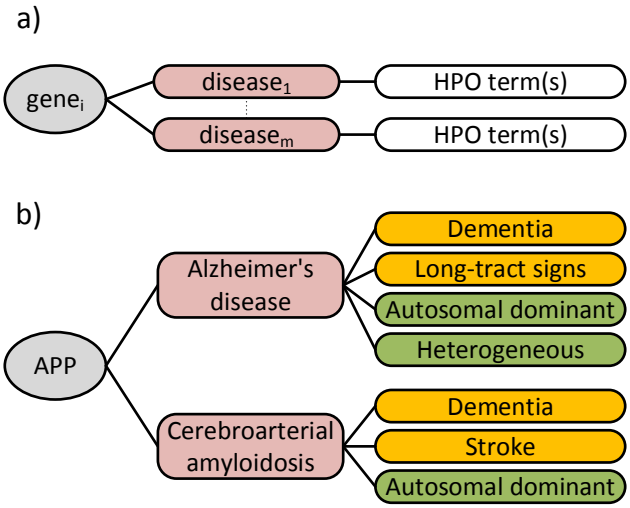

FIGURE 7.1. HPO annotations. a) general format of annotations: genes are annotated with a set of phenotype terms based on their known relationships with diseases b) an example annotation: the amyloid precursor protein (APP) gene is associated with Alzheimer's disease and cerebroarterial amyloidosis. Therefore, the APP gene is annotated with the set of HPO terms (Organ in orange, Inheritance in green) associated with these disease.

the moment. Specifically, the genes are annotated with a set of phenotype terms based on their known relationships with diseases (Fig. 7.1).

Currently, only a small fraction (∼3000) of human protein coding genes are known to be associated with hereditary diseases, and only those genes have HPO annotations at the moment. But researchers believe that there are many other disease-causing genes in the human genome and estimate that another 5000 genes can be associated with phenotypes. However, experimentally finding disease-causing genes is a highly resource consuming and difficult task [10]. Therefore, it is important to explore the feasibility of developing computational methods for predicting gene-HPO associations. While there is a plethora of computational approaches for the related task of prediction of gene-disease associations [47], only one other computational method that directly predict gene-HPO term associations exists at this time [48].

We define the HPO prediction problem as directly predicting the complete set of HPO terms for a given gene. This problem is a hierarchical multilabel classification (HMC) problem [28], as a given gene can be annotated with multiple labels, and the set of labels have a hierarchy associated with them.

Instead of decomposing the problem into multiple single label problems and applying independent binary classifiers for each label separately, we use a single classifier that learns a direct mapping from inputs to the space of hierarchically consistent labelings. Specifically, motivated by the success in modeling the GO term prediction problem using a structured prediction framework [22, 32], we demonstrate the effectiveness of this strategy for HPO term prediction using the same methodology framework [31]. Furthermore, we explore a variety of data sources that are informative for HPO term prediction, including text features.

7.2. Methods

7.2.1. Data. Our models are provided with feature vectors and HPO annotations. Labeled data was created by using annotations available from the HPO web site. Each gene was characterized by a collection of features: PPI and other functional association data (co-expression, co-occurrence, etc.) from BioGRID 3.2.106, STRING 9.1 and GeneMANIA 3.1.2; experimentally derived Gene Ontology annotations; sequence-based features that capture transmembrane and localization features described in [22]. Finally, an NLP pipeline was utilized to characterize genes by same-sentence word occurrences extracted from the biomedical literature, yielding a bag-of-words (BoW) representation for each gene, as described in [50]. Appendix C describes data in more detail.

7.2.2. Models. PHENOstruct is a computational method that directly predicts relevant HPO terms for a given gene. We compare PHENOstruct against a) binary support vector

machines (SVMs) and b) a state-of-the-art HMC method based on decision tree ensembles (Clus-HMC-Ens). In this section we describe PHENOstruct and the two baseline methods. In addition, we assessed the performance of: c) an indirect method that first predicts disease terms for a gene using a structured model and then maps them to HPO terms and d) a method that predicts OMIM disease terms using a nearest-neighbour approach, PhenoPPI-Orth [115]; we then map the OMIM terms to HPO terms. Methods a, b and c were provided the same set of data that was provided to PHENOstruct. For the method PhenoPPIOrth, we downloaded the pre-computed predictions from its website[2].

PHENOstruct. PHENOstruct takes features that describe a set of genes and the HPO annotations associated with those genes as input for training the model (Fig. 7.2). Once the model is learned, it can predict a set of hierarchically consistent HPO terms for a given test gene. More specifically, PHENOstruct learns a compatibility function that models the association between a given input and a structured output [31]. Let $\mathcal{X}$ be the input space where genes are represented and let $\mathcal{Y}$ be the space of labels. The set of HPO terms associated with a given gene is collectively refereed to as its (structured) label. $\mathcal{Y}$ represents each HPO subontology in a vector space where component i represents term i. Given a training set $\{(x_i, y_i)\}_{i=1}^{n}$ where $x_i \epsilon \mathcal{X} \quad \text{and} \quad y_i \epsilon \mathcal{Y}$, the *compatibility* function $f : \mathcal{X} \times \mathcal{Y} \rightarrow \mathcal{R}$ maps the input-output pairs to a score. The output label $\hat{y}$ for an unseen input x can then be obtained by using the argmax operator as $\hat{y} = \arg\max_{y \in \mathcal{Y}_c} f(x, y)$ where $\mathcal{Y}_c \subset \mathcal{Y}$ is the set of all candidate labels. In this work we use the combinations of all terms in the training set as the set of candidate labels $\mathcal{Y}_c$.

In order to obtain correct classification the compatibility value of the true label (correct set of HPO annotations) of an input needs to be higher than that of any other candidate label

[2]http://jjwanglab.org/PhenoPPIOrth/download

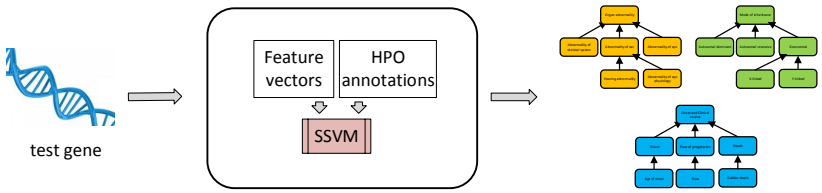

FIGURE 7.2. Overview of PHENOstruct. PHENOstruct takes the set of feature vectors and HPO annotations associated with each gene as input for training. Once trained, it can predict a set of hierarchically consistent HPO terms for a given test gene.

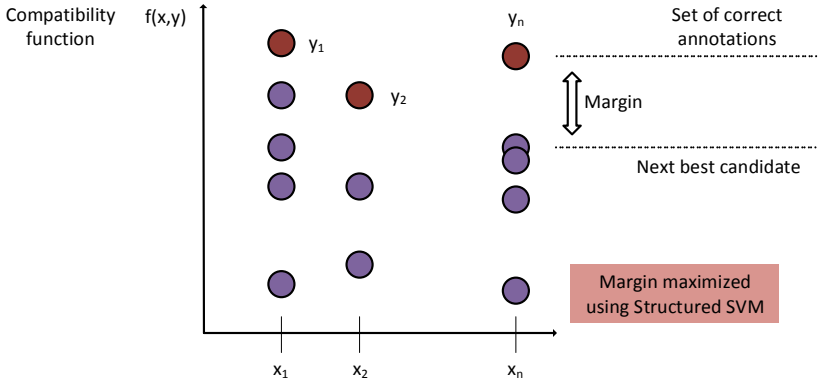

FIGURE 7.3. Visual interpretation of the structured prediction framework. The compatibility function, which is the key component of the structured prediction framework, measures compatibility between a given input and a structured output. The compatibility function of the true label (correct set of HPO annotations) is required to be higher than that of any other label. and the difference between these two scores (margin) is maximized.

(Fig. 7.3). PHENOstruct uses structured SVM (SSVM) training where this is used as a (soft) constraint; it tries to maximize the margin, or the difference between the compatibility value for the actual label and the compatibility for the next best candidate [31]. In the structured-output setting, kernels correspond to dot products in the joint input-output feature space, and they are functions of both inputs and outputs. PHENOstruct uses a joint kernel that is the product of the input-space and the output-space kernels:

$$K((x_1, y_1), (x_2, y_2)) = K_{\mathcal{X}}(x_1, x_2) K_{\mathcal{Y}}(y_1, y_2).$$

The motivation for this form is that two input/output pairs are considered similar if they are similar in both their input space features and their labels. The output space kernel, for which we use a linear kernel between label vectors, captures similarity of the annotations associated with two genes. The input space kernel combines several sources of data by the addition of multiple input-space kernels, one for each data source. Each kernel is normalized according to

$$K_{norm}(z_1, z_2) = K(z_1, z_2)/\sqrt{K(z_1, z_1)K(z_2, z_2)}$$

before being used with the joint input-output kernel. The Strut library[3] with default parameter settings was used for the implementation of PHENOstruct.

Binary SVMs. As a baseline method we trained a collection of binary SVMs, each trained on a single HPO term. Binary SVMs were trained using the PyML[4] machine learning library with default parameter settings. We used linear kernels for each set of input space features.

Clus-HMC-Ens. Clus-HMC-Ens is a state-of-the-art HMC method based on decision tree ensembles which has been shown to be very effective for GO term prediction [69]. In our study, we provide exactly the same set of features used with PHENOStruct as input to Clus-HMC-Ens and use parameter settings that provided the best performance for GO term prediction[5]. The number of bags used was 50 for the Inheritance and Onset subontologies; 10 bags were used for the Organ subontology because of the large running times for this subontology.

[3]http://sourceforge.net/projects/strut/
[4]http://pyml.sourceforge.net
[5]http://dtai.cs.kuleuven.be/clus/hmc-ens/

7.2.3. Evaluation. Classifier performance was estimated using five-fold cross-validation. Since typically scientists/biologists are interested in knowing the set of genes/proteins associated with a certain HPO term, we primarily use a *term-centric* measure for presenting results. Term-centric measures average performance across terms as opposed to *protein-centric* measures which average performance across proteins as described elsewhere [13]. More specifically, we use the term-centric AUC (area under the receiver operating curve), which is computed by averaging the AUCs across HPO terms. PHENOstruct assigns a confidence score to each predicted HPO term, which is computed using the compatibility function as described elsewhere [32]. The onset and clinical course subontology includes terms such as *pace of progression*, *age of onset* and *onset* which are only used for grouping terms. We ignore these grouping terms when computing performance.

7.3. Results and Discussion

7.3.1. PHENOstruct performance. As illustrated in Table 7.1, PHENOstruct significantly outperforms Clus-HMC-Ens and the binary SVMs in the Organ and Onset subontologies. This suggests that modeling the HPO prediction problem as a structured prediction problem is highly effective. It is interesting to note that the biggest improvement of PHENOstruct over binary SVMs is seen in the Organ subontology. Given its very large number of terms, as well as the deep hierarchy, this further confirms the value of the structured approach. PHENOstruct outperforms binary SVMs in the Inheritance and Onset subontologies but to a lesser extent than in the Organ subontology because they are far less complex than the Organ subontology. We note that the two methods that first predict OMIM terms, which are then mapped to HPO terms performed poorly (see details in the supplementary material). It is also interesting to see that Clus-HMC-Ens performs worse

TABLE 7.1. PHENOstruct vs. other methods. Performance across the three HPO subontologies for PHENOstruct, binary SVMs and Clus-HMC-Ens measured using the macro AUC.

Subont.	# categories	Method	AUC
Organ	1,796	Binary SVMs	0.66
		Clus-HMC-Ens	0.65
		PHENOstruct	**0.73**
Inheritance	12	Binary SVMs	0.72
		Clus-HMC-Ens	0.73
		PHENOstruct	**0.74**
Onset	23	Binary SVMs	0.62
		Clus-HMC-Ens	0.58
		PHENOstruct	**0.64**

than binary SVMs with respect to macro AUC (Table 7.1) but performs slightly better than binary SVMs according to protein-centric F-max (Table S3).

PHENOstruct's average AUC for the Organ and Inheritance subontologies are 0.73 and 0.74, respectively. Even though the Organ subontology is a far more complex subontology than the Inheritance subontology (with thousands of terms and 13 levels as opposed to tens of terms and only 3 levels) they show similar performance. The Onset subontology is the hardest to predict accurately, with an average AUC of 0.64. Only six Onset subontology terms have individual AUCs above 0.7 (Table 7.3).

Even though PHENOstruct outperforms the baseline methods, there is much room for improvement, especially in the Onset subontology. The small number of annotated genes in this subontology (Table C.1) makes it difficult to train an effective model while the incomplete nature of the current gold standard used for evaluation tends to underestimate performance of classifiers [100]. See Section 7.3.3 for a detailed analysis of false positives.

In general, Organ subontology terms with few annotations show a mix of both high and low performance as illustrated in Fig. 7.4. This suggests that PHENOstruct is not necessarily affected by the frequency of the terms. But, terms with more annotations tend

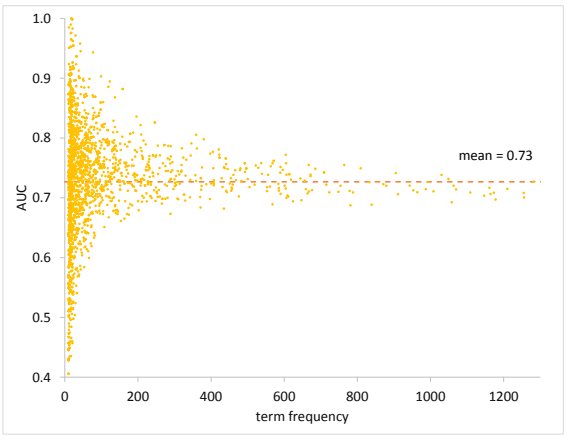

FIGURE 7.4. Performance of PHENOstruct in the Organ subontology. Performance for each term is displayed using AUC against its frequency. The average AUC for the Organ subontology is 0.73.

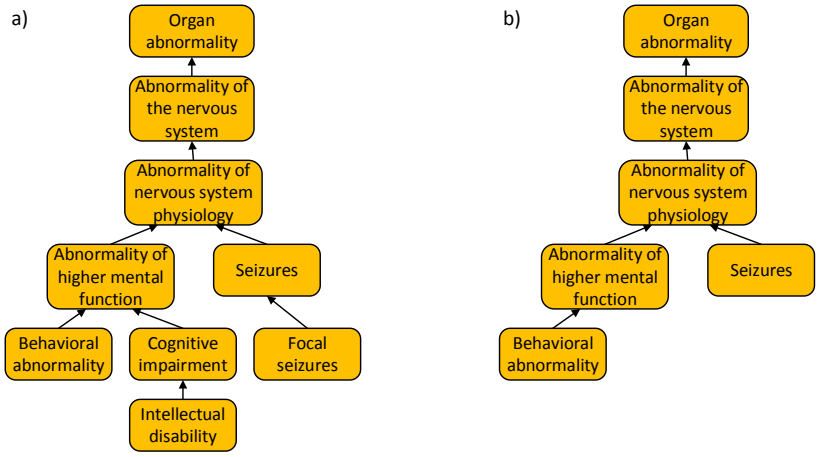

FIGURE 7.5. Example of experimental and predicted annotations. a) experimental annotation of protein P43681 b) PHENOstruct's prediction for P43681 (protein-centric precision and recall for this individual protein is 1.0 and 0.62, respectively).

to show moderate performance. See Fig. 7.5 for an example of experimental and predicted annotations (Organ subontology) for a protein.

It is interesting to note that "polygenic inheritance" and its parent term "mulifactorial inheritance" have the lowest number of annotations as well as the lowest individual AUCs in the Inheritance subontology (see Table 7.2). These two are the terms with the lowest AUC with binary SVMs as well (data not shown). It is not surprising that these two terms

have lower accuracy because each describes inheritance patterns that depend on a mixture of determinants. Moreover, the diseases inherited in this manner – termed complex diseases – are not as well characterized and annotated compared to Mendelian/single gene diseases. On the other hand, the mitochondrial inheritance term has an exceptional AUC of 0.98. It is also the term with the highest AUC with the binary SVMs as well (data not shown). The human mitochondrial DNA was the first significant part of the human genome to be fully sequenced, two decades before the completion of the human genome project [116]. Due to this, and the relative ease of sequencing the mitochondrial genome [117], diseases caused by mutations in the human mitochondrial DNA have been reported very early [118, 119]. It is likely that this well-studied nature of the mitochondrial DNA leads to the high performance of the mitochondrial inheritance term.

All experiments were performed on Linux running machines with 8 cores (64-bit, 3.3GHz) and 8GB memory. Combined running times for performing five-fold cross-validation for all three subontologies are: binary SVMs: 55 hours, Clus-HMC-Ens: 825 hours and PHENOstruct: 90 hours.

7.3.2. Effectiveness of individual data sources. We performed the following set of experiments in order to identify the most effective data sources for HPO prediction using PHENOstruct. First, to identify the individual effectiveness of each source, we performed a series of experiments in which we provided features generated from a single source of data at a time as input to PHENOstruct. Then to understand how much each data source is contributing to the overall performance we conducted leave-one-source-out experiments.

In all three subontologies, network data is the most informative individual data source as illustrated in Fig. 7.6. Moreover, it is by far the main contributor to the overall performance

TABLE 7.2. Performance of PHENOstruct in the Inheritance subontology. The macro AUC for the Inheritance subontology is 0.74. Terms are displayed in ascending order of frequency.

Idntifier	Name	Freq.	Depth	AUC
HP:0001426	Multifactorial inheritance	15	1	0.54
HP:0010982	Polygenic inheritance	15	2	0.54
HP:0001427	Mitochondrial inheritance	41	1	0.98
HP:0003745	Sporadic	52	1	0.61
HP:0001428	Somatic mutation	61	1	0.76
HP:0001423	X-linked dominant inheritance	62	3	0.83
HP:0001419	X-linked recessive inheritance	111	3	0.77
HP:0001425	Heterogeneous	148	1	0.69
HP:0010985	Gonosomal inheritance	198	1	0.80
HP:0001417	X-linked inheritance	198	2	0.80
HP:0000006	Autosomal dominant inheritance	1096	1	0.78
HP:0000007	Autosomal recessive inheritance	1665	1	0.73

TABLE 7.3. Performance of PHENOstruct in the Onset subontology. The macro AUC for the Onset subontology is 0.64. Terms are displayed in ascending order of frequency.

Identifier	Name	Freq.	Depth	AUC
HP:0003584	Late onset	11	4	0.70
HP:0003811	Neonatal death	14	2	0.54
HP:0001699	Sudden death	14	2	0.50
HP:0003680	Nonprogressive disorder	15	2	0.82
HP:0003826	Stillbirth	21	2	0.67
HP:0003819	Death in childhood	23	2	0.65
HP:0003623	Neonatal onset	23	3	0.64
HP:0003678	Rapidly progressive	33	2	0.50
HP:0011463	Childhood onset	41	3	0.62
HP:0001522	Death in infancy	44	2	0.70
HP:0003829	Incomplete penetrance	58	2	0.61
HP:0003621	Juvenile onset	90	3	0.70
HP:0003677	Slow progression	95	2	0.62
HP:0003581	Adult onset	98	3	0.71
HP:0011420	Death	111	1	0.61
HP:0003828	Variable expressivity	132	2	0.66
HP:0003577	Congenital onset	135	3	0.60
HP:0003676	Progressive disorder	141	2	0.70
HP:0003593	Infantile onset	245	3	0.66
HP:0003812	Phenotypic variability	310	1	0.65

both in the Organ and Inheritance subontologies (Fig. 7.7). This is intuitive because if two genes/proteins are known to be interacting and/or active in the same pathways it leads to association with the same/similar diseases/phenotypes.

Although the genetic variant features provide the lowest performance in the Organ and Onset subontologies, leaving out variant data hurts the overall performance noticeably in all three subontologies as can be seen in Fig. 7.7. This suggests that variant data are very useful especially as a complementary dataset to the others. Moreover, we found that variant data are very effective for predicting cancer-related terms in the Organ subontology (data not shown).

It is very encouraging to see that the literature data with a simple BoW representation by itself is very informative (Fig. 7.6) and leaving out literature features shows considerable performance drop in the other two subontologies (Fig. 7.7). In an analysis of the structured SVM weight vector, we found that the majority of the most important tokens extracted from literature consists of names of proteins, genes and diseases (data not shwon).

We also considered an alternative representation where a gene is represented by a vector in which the element i gives the number of times the word i occurred in the same sentence with that particular gene/protein divided by the total number of unique genes/proteins that word co-occurred with. This representation is analogous to the TFIDF (term frequency $*$ inverse document frequency) representation typically used in information retrieval and text mining [120]. However, these features led to slight deterioration of performance in all three subontologies (data not shown).

Although Gene Ontology features provide the second best individual performance both in the Organ and Onset subontologies (Fig. 7.6), their contribution to the overall performance

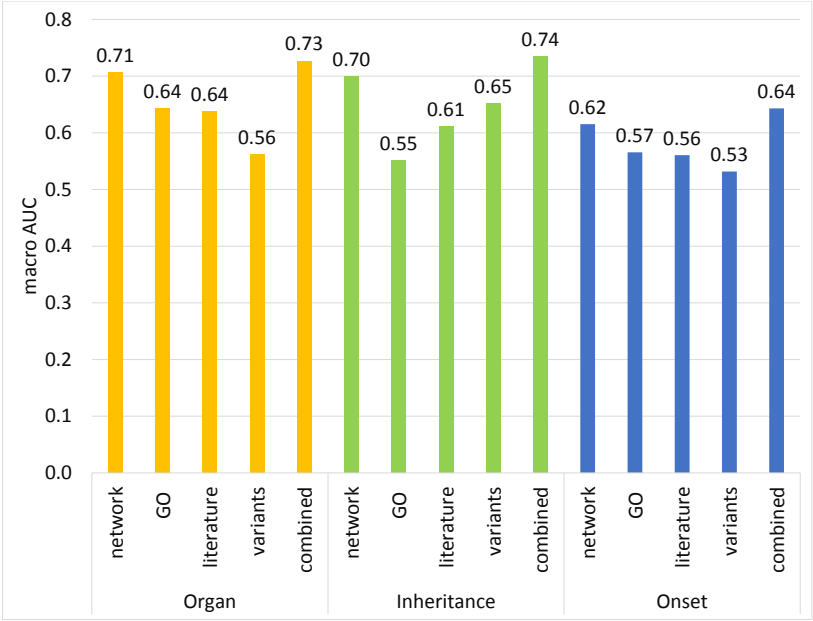

FIGURE 7.6. Performance of PHENOstruct with individual data sources. Results are shown for each source of data: network (functional association data); Gene Ontology annotations; literature mining data; genetic variants and the model that combines all features together.

is very minimal (Fig. 7.7). In fact leaving out GO features increases the overall performance. The incompleteness of Gene Ontology annotations may have contributed towards this.

Finally, the combination of all the features provides higher performance than individual feature sets in all three subontologies as can be seen in Fig. 7.6. However, leaving out GO features in the Inheritance and Onset subontologies, led to improved performance, suggesting that not all sources contribute to the overall performance. This shows that the selection of data sources must be performed carefully in order to find the optimal combination of sources for each subontology.

7.3.3. VALIDATING FALSE POSITIVES. Like other biological ontologies, the HPO is incomplete due to various factors such as slowness of the curation process [38]. In other words, the set of HPO annotations we considered as the gold standard does not fully represent all the phenotypes that should be associated with the currently annotated genes; this leads

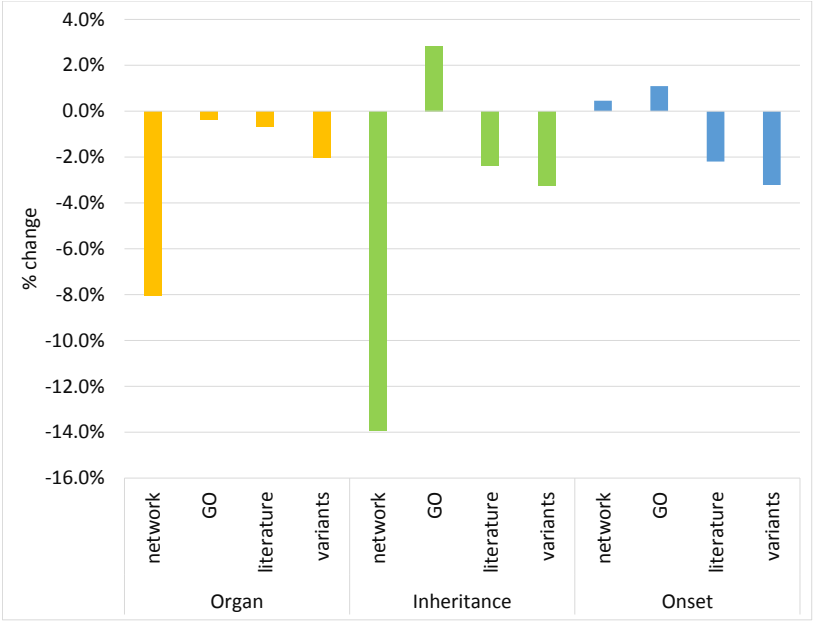

FIGURE 7.7. Performance of PHENOstruct in leave-one-source-out experiments (measured by the % change in macro AUC by leaving out a single selected source relative to its macro AUC obtained using all data sources; negative % change means the performance dropped after leaving out the particular source of data).

to performance estimates that underestimate the true performance of a classifier [100]. To explore this issue, we selected 25 predictions made by PHENOstruct which were considered false positives according to the current gold standard and looked for evidence in the current biomedical literature that can be used as evidence for those predictions. For 14 of those predictions we were able to find supporting evidence. The details of the complete validation process are given in the supplementary material.

CHAPTER 8

Conclusion

In this work we successfully addressed the following three of the most important challenges in large-scale automated protein function prediction (1) GO term prediction for annotated proteins (2) Biomedical literature mining for GO term prediction and (3) HPO term prediction. Here we summarize the major contributions of our work. First, we conducted the first comprehensive study on different evaluation protocols for protein function prediction. Findings of this study has implications for the design of computational experiments in the area of AFP and provide useful insight for the understanding and design of future CAFA competitions. Based on these findings, we developed GOstruct 2.0 by introducing several improvements that help the model make use of existing annotations; GOstruct 2.0 performs significantly better on the task of protein function prediction for already annotated proteins. Secondly, we conducted the first comprehensive evaluation of how to utilize biomedical literature for improving protein function prediction. Furthermore, we developed a novel method based on graph kernels for automatically extracting protein-function relations from Biomedical text; This method will help GO curators make the curation process efficient significantly. Finally, we investigated the problem of HPO term prediction and developed PHENOstruct which can predict a set of HPO terms for a given protein. This study introduced several data sources specifically informative of HPO prediction such as variation data which is different from data sources traditionally informative of GO prediction. This information will be very helpful for the practitioners developing new methods for this relatively unexplored area.

8.1. Open Problems

Several issues related to all three projects on which we focused in this work remain to be solved. The proposed improvements in GOstruct 2.0 did not work as well on yeast data; an extensive model selection procedure for selecting the optimal pair of C parameters with respect to the newly introduced constraints might be required. Although the proposed improvements for GOstruct 2.0 were not designed towards contributing to the increased classifier accuracy in the task of making predictions for unannotated proteins, a decline of the performance in some cases was observed. Further investigation is required to identify the underlying causes of this effect, which can help GOstruct perform better on unannotated proteins.

The novel protein-function relation extraction methods proposed in this work use single sentences as the context. However, often the functional relationships between a protein and a functional category is described across sentences; in our earlier studies we assessed inter-sentential co-mentions to be as valuable as their intra-sentential counterparts [50]. Therefore, techniques for expanding the currently proposed methods to incorporate multi-sentence contexts or sophisticated approaches dedicated for inter-sentence extraction of protein-function relations are required. Another aspect worth exploring is developing semi-supervised learning techniques for incorporating the large corpus of unannotated co-mentions for boosting performance. Furthermore, collaborative work with the bio-curators is required to incorporates these proposed methods in to their actual curation pipelines so that we can assess their practical usability. A related issue is the scalability of our random walk kernel methods; further work is required for proposing sophisticated kernels suitable for the large corpora encountered in biomedical text mining.

Although PHENOstruct outperformed the baseline methods, there is considerable room for improvement in all three subontologies. HPO is a relatively new ontology that will likely see substantial growth in the coming years, which will help in improving the accuracy of computational methods that contribute to its expansion. This makes it an important requirement to continuously monitor the evolution of HPO at regular intervals and understand the patterns of expansion. It is important to note that improvements proposed for the structured output framework in terms of GO term prediction for annotated proteins in chapter 4 are directly applicable to PHENOstruct as well. While some boost in performance can likely be obtained using those improvements, its performance will also improve as the number of HPO annotations increases.

There are several other ways in which our work on HPO term prediction can be extended. For the literature data we used a simple bag-of-words representation. An alternative, is to try and extract gene-HPO term co-mentions directly; in the context of GO term prediction we have found that both approaches lead to similar overall performance [50]. However, co-mentions have the added value that they are easy to verify by a human curator. And the natural next step following the extraction of protein-HPO co-mentions would be to explore the feasibility of utilizing a co-mention classifier similar to the methods we applied for GO co-mention prediction. m Another source of information that can be utilized is semantic similarity of HPO terms to other phenotypic ontologies such the mammalian phenotype ontology, which is currently used for annotating the rat genome [121]. Finally, exploring the effectiveness of combining all three subontologies, as opposed to treating them as three independent subontologies as we have done here, is also worth exploring.

References

[1] M.-C. De Marneffe, B. MacCartney, C. D. Manning, *et al.*, "Generating typed dependency parses from phrase structure parses," in *Proceedings of the fifth international conference on Language Resources and Evaluation (LREC)*, vol. 6, pp. 449–454, 2006.

[2] J. M. Berg, J. L. Tymoczko, and L. Stryer, *Biochemistry: international edition*. WH Freeman & Company Limited, 2006.

[3] R. Moll, M. Divo, and L. Langbein, "The human keratins: biology and pathology," *Histochemistry and cell biology*, vol. 129, no. 6, pp. 705–733, 2008.

[4] R. Rhoades and R. G. Pflanzer, *Human physiology*. Saunders College Pub., 1989.

[5] P. Ponka, C. Beaumont, and D. R. Richardson, "Function and regulation of transferrin and ferritin.," in *Seminars in hematology*, vol. 35, pp. 35–54, 1998.

[6] H. F. Bunn, B. G. Forget, and H. M. Ranney, *Human hemoglobins*. WB Saunders Company, 1977.

[7] 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," *Nature Genetics*, vol. 25, pp. 25–29, May 2000.

[8] S. Khler, S. C. Doelken, C. J. Mungall, S. Bauer, H. V. Firth, *et al.*, "The human phenotype ontology project: linking molecular biology and disease through phenotype data," *Nucleic Acids Research*, vol. 42, no. D1, pp. D966–D974, 2014.

[9] P. N. Robinson, "Deep phenotyping for precision medicine," *Human Mutation*, vol. 33, no. 5, pp. 777–780, 2012.

[10] P. N. Robinson, S. Khler, A. Oellrich, S. M. G. Project, K. Wang, C. J. Mungall, S. E. Lewis, N. Washington, S. Bauer, D. Seelow, P. Krawitz, C. Gilissen, M. Haendel, and D. Smedley, "Improved exome prioritization of disease genes through cross-species phenotype comparison," *Genome Research*, vol. 24, no. 2, pp. 340–348, 2014.

[11] P. Muir, S. Li, S. Lou, D. Wang, D. J. Spakowicz, L. Salichos, J. Zhang, G. M. Weinstock, F. Isaacs, J. Rozowsky, and M. Gerstein, "The real cost of sequencing: scaling computation to keep pace with data generation," *Genome Biology*, vol. 17, no. 1, pp. 1–9, 2016.

[12] R. P. Huntley, T. Sawford, M. J. Martin, and C. O'Donovan, "Understanding how and why the gene ontology and its annotations evolve: the GO within UniProt," *GigaScience*, vol. 3, no. 1, pp. 1–9, 2014.

[13] P. Radivojac, W. T. Clark, I. Friedberg, *et al.*, "A large-scale evaluation of computational protein function prediction," *Nat Meth*, vol. 10, pp. 221–227, Mar 2013.

[14] A. Conesa, S. Götz, J. M. García-Gómez, J. Terol, M. Talón, and M. Robles, "Blast2go: a universal tool for annotation, visualization and analysis in functional genomics research," *Bioinformatics*, vol. 21, no. 18, pp. 3674–3676, 2005.

[15] G. Obozinski, G. Lanckriet, C. Grant, M. I. Jordan, and W. S. Noble, "Consistent probabilistic outputs for protein function prediction," *Genome Biology*, vol. 9, no. Suppl 1, p. S6, 2008.

[16] Y. Guan, C. L. Myers, D. C. Hess, Z. Barutcuoglu, A. Caudy, and O. Troyanskaya, "Predicting gene function in a hierarchical context with an ensemble of classifiers," *Genome biology*, vol. 9, no. Suppl 1, p. S3, 2008.

[17] W. Tian, L. V. Zhang, M. Tasan, F. D. Gibbons, O. D. King, J. Park, Z. Wunderlich,

J. M. Cherry, and F. P. Roth, "Combining guilt-by-association and guilt-by-profiling to predict saccharomyces cerevisiae gene function," *Genome Biology*, vol. 9, no. Suppl 1, p. S7, 2008.

[18] B. Hayete and J. R. Bienkowska, "GOtrees: predicting go associations from protein domain composition using decision trees.," in *Pacific Symposium on Biocomputing*, vol. 10, pp. 127–138, World Scientific, 2005.

[19] Y. Chen and D. Xu, "Global protein function annotation through mining genome-scale data in yeast saccharomyces cerevisiae," *Nucleic acids research*, vol. 32, no. 21, pp. 6414–6424, 2004.

[20] J. McDermott, R. Bumgarner, and R. Samudrala, "Functional annotation from predicted protein interaction networks," *Bioinformatics*, vol. 21, no. 15, pp. 3217–3226, 2005.

[21] S. Mostafavi, D. Ray, D. Warde-Farley, C. Grouios, Q. Morris, *et al.*, "GeneMANIA: a real-time multiple association network integration algorithm for predicting gene function," *Genome Biology*, vol. 9, no. Suppl 1, p. S4, 2008.

[22] A. Sokolov and A. Ben-Hur, "Hierarchical Classification of Gene Ontology terms using the Gostruct method.," *J. Bioinformatics and Computational Biology*, vol. 8, no. 2, pp. 357–376, 2010.

[23] D. Warde-Farley, S. L. Donaldson, O. Comes, K. Zuberi, R. Badrawi, P. Chao, M. Franz, C. Grouios, F. Kazi, C. T. Lopes, A. Maitland, S. Mostafavi, J. Montojo, Q. Shao, G. Wright, G. D. Bader, and Q. Morris, "The GeneMANIA prediction server: biological network integration for gene prioritization and predicting gene function," *Nucleic Acids Research*, vol. 38, no. suppl 2, pp. W214–W220, 2010.

[24] W. T. Clark and P. Radivojac, "Analysis of protein function and its prediction from

amino acid sequence," *Proteins: Structure, Function, and Bioinformatics*, vol. 79, no. 7, pp. 2086–2096, 2011.

[25] J. Moult, J. T. Pedersen, R. Judson, and K. Fidelis, "A large-scale experiment to assess protein structure prediction methods," *Proteins: Structure, Function, and Bioinformatics*, vol. 23, no. 3, pp. ii–iv, 1995.

[26] J. Janin, K. Henrick, J. Moult, L. T. Eyck, M. J. E. Sternberg, S. Vajda, I. Vakser, and S. J. Wodak, "CAPRI: A Critical Assessment of PRedicted Interactions," *Proteins: Structure, Function, and Bioinformatics*, vol. 52, no. 1, pp. 2–9, 2003.

[27] Y. Jiang, T. R. Oron, W. T. Clark, A. R. Bankapur, D. D'Andrea, R. Lepore, C. S. Funk, I. Kahanda, K. M. Verspoor, A. Ben-Hur, *et al.*, "An expanded evaluation of protein function prediction methods shows an improvement in accuracy," *arXiv preprint arXiv:1601.00891*, 2016.

[28] W. Bi and J. T. Kwok, "Multi-label classification on tree- and dag-structured hierarchies," in *Proceedings of the 28th International Conference on Machine Learning (ICML-11)* (L. Getoor and T. Scheffer, eds.), (New York, NY, USA), pp. 17–24, ACM, 2011.

[29] C. N. Silla and A. A. Freitas, "A survey of hierarchical classification across different application domains," *Data Mining and Knowledge Discovery*, vol. 22, no. 1, pp. 31–72, 2011.

[30] G. Obozinski, C. E. Grant, G. R. G. Lanckriet, M. I. Jordan, and W. W. Noble, "Consistent probabilistic outputs for protein function prediction," *Genome Biology*, vol. 9, no. Suppl 1, p. S6, 2008.

[31] I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun, "Large margin methods for structured and interdependent output variables," *J. Mach. Learn. Res.*, vol. 6, pp. 1453–1484, Dec. 2005.

[32] A. Sokolov, C. Funk, K. Graim, K. Verspoor, and A. Ben-Hur, "Combining heterogeneous data sources for accurate functional annotation of proteins," *BMC Bioinformatics*, vol. 14, no. 3, pp. 1–13, 2013.

[33] A. Schnoes, D. Ream, A. Thorman, P. Babbitt, and I. Friedberg, "Biases in the experimental annotations of protein function and their effect on our understanding of protein function space.," *PLoS computational biology*, vol. 9, no. 5, pp. e1003063–e1003063, 2012.

[34] S. Burge, T. K. Attwood, A. Bateman, T. Z. Berardini, M. Cherry, C. O'Donovan, P. Gaudet, *et al.*, "Biocurators and biocuration: surveying the 21st century challenges," *Database (Oxford)*, p. 59, 2012.

[35] T. G. O. Consortium, "Gene ontology consortium: going forward," *Nucleic Acids Research*, vol. 43, no. D1, pp. D1049–D1056, 2015.

[36] A. Chatr-aryamontri, B.-J. Breitkreutz, S. Heinicke, L. Boucher, A. Winter, C. Stark, J. Nixon, L. Ramage, N. Kolas, L. ODonnell, T. Reguly, A. Breitkreutz, A. Sellam, D. Chen, C. Chang, J. Rust, M. Livstone, R. Oughtred, K. Dolinski, and M. Tyers, "The BioGRID interaction database: 2013 update," *Nucleic Acids Research*, vol. 41, no. D1, pp. D816–D823, 2013.

[37] L. Hirschman, G. A. C. Burns, M. Krallinger, C. Arighi, K. B. Cohen, A. Valencia, C. H. Wu, A. Chatr-Aryamontri, K. G. Dowell, E. Huala, *et al.*, "Text mining for the biocuration workflow," *Database (Oxford)*, vol. 1, pp. 20–30, 2012.

[38] W. A. Baumgartner, K. B. Cohen, L. M. Fox, G. Acquaah-Mensah, and L. Hunter, "Manual curation is not sufficient for annotation of genomic databases," *Bioinformatics*, vol. 23, no. 13, pp. i41–i48, 2007.

[39] C. Giuliano, A. Lavelli, and L. Romano, "Exploiting Shallow Linguistic Information for

Relation Extraction from Biomedical Literature," in *11th Conference of the European Chapter of the Association for Computational Linguistics (EACL '06)*, (Trento, Italy), pp. 401–408, European Chapter of the Association for Computational Linguistics, Apr. 2006.

[40] S. Kim, J. Yoon, and J. Yang, "Kernel approaches for genic interaction extraction," *Bioinformatics*, vol. 24, no. 1, pp. 118–126, 2008.

[41] S. Kim, J. Yoon, J. Yang, and S. Park, "Walk-weighted subsequence kernels for protein-protein interaction extraction," *BMC bioinformatics*, vol. 11, no. 1, p. 1, 2010.

[42] F. M. Chowdhury, A. Lavelli, and A. Moschitti, "A study on dependency tree kernels for automatic extraction of protein-protein interaction," in *Proceedings of BioNLP 2011 Workshop*, pp. 124–133, Association for Computational Linguistics, 2011.

[43] K. M. Verspoor, "Roles for text mining in protein function prediction," in *Biomedical Literature Mining* (V. D. Kumar and H. J. Tipney, eds.), vol. 1159 of *Methods in Molecular Biology*, pp. 95–108, Springer New York, 2014.

[44] A. Wong and H. Shatkay, "Protein function prediction using text-based features extracted from the biomedical literature: The CAFA challenge," *BMC Bioinformatics*, vol. 14, pp. S14–S14, Feb 2013. 1471-2105-14-S3-S14[PII].

[45] H. Shatkay, S. Brady, and A. Wong, "Text as data: Using text-based features for proteins representation and for computational prediction of their characteristics," *Methods*, vol. 74, no. 0, pp. 54 – 64, 2015. Text mining of biomedical literature.

[46] J. Björne and T. Salakoski, "A machine learning model and evaluation of text mining for protein function prediction," *Automated Function Prediction Featuring a Critical Assessment of Function Annotations (AFP/CAFA)*, pp. 7–8, 2011.

[47] Y. Moreau and L.-C. Tranchevent, "Computational tools for prioritizing candidate

genes: boosting disease gene discovery," *Nature Reviews Genetics*, vol. 13, pp. 523–536, Aug 2012.

[48] G. Valentini, S. Köhler, M. Re, M. Notaro, and P. N. Robinson, "Prediction of human gene-phenotype associations by exploiting the hierarchical structure of the human phenotype ontology," in *Bioinformatics and Biomedical Engineering*, pp. 66–77, Springer, 2015.

[49] I. Kahanda, C. S. Funk, F. Ullah, K. M. Verspoor, and A. Ben-Hur, "A close look at protein function prediction evaluation protocols," *GigaScience*, vol. 4, no. 1, pp. 1–10, 2015.

[50] C. S. Funk, I. Kahanda, A. Ben-Hur, and K. M. Verspoor, "Evaluating a variety of text-mined features for automatic protein function prediction with gostruct," *Journal of Biomedical Semantics*, vol. 6, no. 1, p. 9, 2015.

[51] I. Kahanda, C. Funk, K. Verspoor, and A. Ben-Hur, "PHENOstruct: Prediction of human phenotype ontology terms using heterogeneous data sources," *F1000Research*, vol. 4, no. 259, 2015.

[52] Z.-H. Duan, B. Hughes, L. Reichel, D. M. Perez, and T. Shi, "The relationship between protein sequences and their gene ontology functions," *BMC bioinformatics*, vol. 7, no. 4, p. 1, 2006.

[53] C. Camacho, G. Coulouris, V. Avagyan, N. Ma, J. Papadopoulos, K. Bealer, and T. Madden, "BLAST+: architecture and applications," *BMC Bioinformatics*, vol. 10, no. 1, pp. 1–9, 2009.

[54] G. Boratyn, A. Schffer, R. Agarwala, S. Altschul, D. Lipman, and T. Madden, "Domain enhanced lookup time accelerated blast," *Biology Direct*, vol. 7, no. 1, pp. 1–14, 2012.

[55] Y. Zhang, "I-TASSER server for protein 3D structure prediction," *BMC bioinformatics*, vol. 9, no. 1, p. 40, 2008.

[56] A. Hildebrand, M. Remmert, A. Biegert, and J. Söding, "Fast and accurate automatic structure prediction with HHpred," *Proteins: Structure, Function, and Bioinformatics*, vol. 77, no. S9, pp. 128–132, 2009.

[57] B. Rost, J. Liu, R. Nair, K. Wrzeszczynski, and Y. Ofran, "Automatic prediction of protein function," *Cellular and Molecular Life Sciences CMLS*, vol. 60, no. 12, pp. 2637–2650, 2003.

[58] P. Horton, K.-J. Park, T. Obayashi, N. Fujita, H. Harada, C. Adams-Collier, and K. Nakai, "WoLF PSORT: protein localization predictor," *Nucleic Acids Research*, vol. 35, no. suppl 2, pp. W585–W587, 2007.

[59] C. v. Mering, M. Huynen, D. Jaeggi, S. Schmidt, P. Bork, and B. Snel, "String: a database of predicted functional associations between proteins," *Nucleic Acids Research*, vol. 31, no. 1, pp. 258–261, 2003.

[60] D. Szklarczyk, A. Franceschini, M. Kuhn, M. Simonovic, A. Roth, P. Minguez, T. Doerks, M. Stark, J. Muller, P. Bork, *et al.*, "The string database in 2011: functional interaction networks of proteins, globally integrated and scored.," *Nucleic acids research*, vol. 39, no. Database issue, pp. D561–8, 2011.

[61] K. Taha and P. D. Yoo, "Predicting the functions of a protein from its ability to associate with other molecules," *BMC Bioinformatics*, vol. 17, no. 1, pp. 1–28, 2016.

[62] G. Zehetner, "OntoBlast function: From sequence similarities directly to potential functional annotations by ontology terms," *Nucleic acids research*, vol. 31, no. 13, pp. 3799–3803, 2003.

[63] S. Hennig, D. Groth, and H. Lehrach, "Automated gene ontology annotation for anonymous sequence data," *Nucleic Acids Research*, vol. 31, no. 13, pp. 3712–3715, 2003.

[64] D. Devos and A. Valencia, "Practical limits of function prediction," *Proteins: Structure, Function, and Bioinformatics*, vol. 41, no. 1, pp. 98–107, 2000.

[65] Y. Guan, C. Myers, D. Hess, Z. Barutcuoglu, A. Caudy, and O. Troyanskaya, "Predicting gene function in a hierarchical context with an ensemble of classifiers," *Genome Biology*, vol. 9, no. Suppl 1, 2008.

[66] R. Sharan, I. Ulitsky, and R. Shamir, "Network-based prediction of protein function," *Molecular systems biology*, vol. 3, no. 1, p. 88, 2007.

[67] X. Zhu, Z. Ghahramani, J. Lafferty, *et al.*, "Semi-supervised learning using gaussian fields and harmonic functions," in *Proceedings of the International Conference on Machine Learning (ICML)*, vol. 3, pp. 912–919, 2003.

[68] Z. Barutcuoglu, R. E. Schapire, and O. G. Troyanskaya, "Hierarchical multi-label prediction of gene function," *Bioinformatics*, vol. 22, pp. 830–836, Apr. 2006.

[69] L. Schietgat, C. Vens, J. Struyf, H. Blockeel, D. Kocev, and S. Džeroski, "Predicting gene function using hierarchical multi-label decision tree ensembles," *BMC Bioinformatics*, vol. 11, pp. 2–2, Jan 2010. 1471-2105-11-2[PII].

[70] A. M. Cohen and W. R. Hersh, "A survey of current work in biomedical text mining," *Briefings in Bioinformatics*, vol. 6, no. 1, pp. 57–71, 2005.

[71] M. Abulaish and L. Dey, "Biological relation extraction and query answering from {MEDLINE} abstracts using ontology-based text mining," *Data & Knowledge Engineering*, vol. 61, no. 2, pp. 228 – 262, 2007.

[72] P. Zweigenbaum, D. Demner-Fushman, H. Yu, and K. B. Cohen, "Frontiers of biomedical text mining: current progress," *Briefings in bioinformatics*, vol. 8, no. 5, pp. 358–375, 2007.

[73] R. Bunescu, R. Mooney, A. Ramani, and E. Marcotte, "Integrating co-occurrence statistics with information extraction for robust retrieval of protein interactions from medline," in *Proceedings of the Workshop on Linking Natural Language Processing and Biology: Towards Deeper Biological Literature Analysis*, pp. 49–56, Association for Computational Linguistics, 2006.

[74] C. D. Manning and H. Schütze, *Foundations of statistical natural language processing*. MIT press, 1999.

[75] H. Cao, M. Markatou, G. B. Melton, M. F. Chiang, and G. Hripcsak, "Mining a clinical data warehouse to discover disease-finding associations using co-occurrence statistics," in *AMIA Annual Symposium Proceedings*, vol. 2005, p. 106, American Medical Informatics Association, 2005.

[76] L. Adamic, D. Wilkinson, B. Huberman, and E. Adar, "A literature based method for identifying gene-disease connections.," in *Proceedings of the IEEE Computer Society Bioinformatics Conference*, vol. 1, p. 109, 2002.

[77] E. S. Chen, G. Hripcsak, H. Xu, M. Markatou, and C. Friedman, "Automated acquisition of disease–drug knowledge from biomedical and clinical documents: an initial study," *Journal of the American Medical Informatics Association*, vol. 15, no. 1, pp. 87–98, 2008.

[78] T.-K. Jenssen, A. Lægreid, J. Komorowski, and E. Hovig, "A literature network of human genes for high-throughput analysis of gene expression," *Nature genetics*, vol. 28, no. 1, pp. 21–28, 2001.

[79] R. Hoffmann and A. Valencia, "Implementing the iHOP concept for navigation of biomedical literature," *Bioinformatics*, vol. 21, no. suppl 2, pp. ii252–ii258, 2005.

[80] X. Wang, G. Hripcsak, M. Markatou, and C. Friedman, "Active computerized pharmacovigilance using natural language processing, statistics, and electronic health records: a feasibility study," *Journal of the American Medical Informatics Association*, vol. 16, no. 3, pp. 328–337, 2009.

[81] K. Fundel, R. Küffner, and R. Zimmer, "Relexrelation extraction using dependency parse trees," *Bioinformatics*, vol. 23, no. 3, pp. 365–371, 2007.

[82] J. Hakenberg, C. Plake, U. Leser, H. Kirsch, and D. Rebholz-Schuhmann, "Lll05 challenge: Genic interaction extraction-identification of language patterns based on alignment and finite state automata," in *Proceedings of the 4th Learning Language in Logic workshop (LLL05)*, pp. 38–45, Citeseer, 2005.

[83] J. G. Caporaso, W. A. Baumgartner Jr, D. A. Randolph, K. B. Cohen, and L. Hunter, "Rapid pattern development for concept recognition systems: application to point mutations," *Journal of Bioinformatics and Computational Biology*, vol. 5, no. 06, pp. 1233–1259, 2007.

[84] Y. Peng, M. Torii, C. H. Wu, and K. Vijay-Shanker, "A generalizable NLP framework for fast development of pattern-based biomedical relation extraction systems," *BMC bioinformatics*, vol. 15, no. 1, p. 1, 2014.

[85] N. Kambhatla, "Combining lexical, syntactic, and semantic features with maximum entropy models for information extraction," in *The Companion Volume to the Proceedings of 42nd Annual Meeting of the Association for Computational Linguistics*, (Barcelona, Spain), pp. 178–181, Association for Computational Linguistics, July 2004.

[86] M. Miwa, R. Sætre, Y. Miyao, and J. Tsujii, "A rich feature vector for protein-protein interaction extraction from multiple corpora," in *Proceedings of the Conference on Empirical*

Methods in Natural Language Processing, vol. 1, pp. 121–130, Association for Computational Linguistics, 2009.

[87] K. Raja, S. Subramani, and J. Natarajan, "PPInterFindera mining tool for extracting causal relations on human proteins from literature," *Database (Oxford)*, vol. 1, pp. 52–62, 2013.

[88] A. Goffeau, B. Barrell, H. Bussey, R. Davis, B. Dujon, H. Feldmann, F. Galibert, J. Hoheisel, C. Jacq, M. Johnston, *et al.*, "Life with 6000 genes," *Science*, vol. 274, no. 5287, pp. 546–567, 1996.

[89] T. Liljefors, P. Krogsgaard-Larsen, and U. Madsen, *Textbook of drug design and discovery*. CRC Press, 2002.

[90] J. Gillis and P. Pavlidis, ""Guilt by Association" is the Exception Rather Than the Rule in Gene Networks," *PLoS Computational Biology*, vol. 8, no. 3, 2012.

[91] D. Chicco, M. Tagliasacchi, and M. Masseroli, "Genomic annotation prediction based on integrated information," in *Computational Intelligence Methods for Bioinformatics and Biostatistics* (E. Biganzoli, A. Vellido, F. Ambrogi, and R. Tagliaferri, eds.), vol. 7548 of *Lecture Notes in Computer Science*, pp. 238–252, Springer Berlin Heidelberg, 2012.

[92] D. Chicco, P. Sadowski, and P. Baldi, "Deep autoencoder neural networks for gene ontology annotation predictions," in *Proceedings of the 5th ACM Conference on Bioinformatics, Computational Biology, and Health Informatics*, BCB '14, (New York, NY, USA), pp. 533–540, ACM, 2014.

[93] M. Masseroli, D. Chicco, and P. Pinoli, "Probabilistic latent semantic analysis for prediction of gene ontology annotations," in *Proceedings of the International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, June 2012.

[94] P. Pinoli, D. Chicco, and M. Masseroli, "Latent Dirichlet allocation based on Gibbs sampling for gene function prediction," in *Proceedings of the IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology*, pp. 1–8, May 2014.

[95] O. Dameron, C. Bettembourg, and N. Le Meur, "Measuring the evolution of ontology complexity: the gene ontology case study," *PLoS One*, vol. 8, no. 10, p. e75993, 2013.

[96] G. O. Consortium *et al.*, "Gene ontology consortium: going forward," *Nucleic acids research*, vol. 43, no. D1, pp. D1049–D1056, 2015.

[97] J. Gillis and P. Pavlidis, "Assessing identity, redundancy and confounds in gene ontology annotations over time," *Bioinformatics*, 2013.

[98] M. Bada, D. Sitnikov, J. A. Blake, and L. E. Hunter, "Occurrence of gene ontology, protein ontology, and ncbi taxonomy concepts in text toward automatic gene ontology annotation of genes and gene products," *BioLink–an ISMB Special Interest Group. Berlin, Germany: Proceedings of BioLINK SIG*, vol. 2013, pp. 13–19, 2013.

[99] Y. Mao, K. Van Auken, D. Li, C. N. Arighi, P. McQuilton, G. T. Hayman, S. Tweedie, M. L. Schaeffer, S. J. F. Laulederkind, S.-J. Wang, J. Gobeill, P. Ruch, A. T. Luu, J.-j. Kim, J.-H. Chiang, Y.-D. Chen, C.-J. Yang, H. Liu, D. Zhu, Y. Li, H. Yu, E. Emadzadeh, G. Gonzalez, J.-M. Chen, H.-J. Dai, and Z. Lu, "Overview of the gene ontology task at BioCreative IV," *Database*, vol. 1, pp. 14–28, 2014.

[100] C. Huttenhower, M. A. Hibbs, C. L. Myers, A. A. Caudy, D. C. Hess, and O. G. Troyanskaya, "The impact of incomplete knowledge on evaluation: an experimental benchmark for protein function prediction," *Bioinformatics*, vol. 25, no. 18, pp. 2404–2410, 2009.

[101] K. M. Borgwardt, C. S. Ong, S. Schönauer, S. Vishwanathan, A. J. Smola, and H.-P. Kriegel, "Protein function prediction via graph kernels," *Bioinformatics*, vol. 21, no. suppl

1, pp. i47–i56, 2005.

[102] T. Gärtner, P. Flach, and S. Wrobel, "On graph kernels: Hardness results and efficient alternatives," in *Learning Theory and Kernel Machines*, pp. 129–143, Springer, 2003.

[103] R. C. Bunescu and R. J. Mooney, "A shortest path dependency kernel for relation extraction," in *Proceedings of the conference on Human Language Technology and Empirical Methods in Natural Language Processing*, pp. 724–731, Association for Computational Linguistics, 2005.

[104] R. Lebret and R. Collobert, "Word emdeddings through Hellinger PCA," *arXiv preprint arXiv:1312.5542*, 2013.

[105] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[106] J. R. Firth, "A synopsis of linguistic theory, 1930-1955," *Studies in Linguistic Analysis*, pp. 1–32, 1957.

[107] V. I. Levenshtein, "Binary codes capable of correcting deletions, insertions, and reversals," in *Soviet physics doklady*, vol. 10, pp. 707–710, 1966.

[108] G. Sidorov, A. Gelbukh, H. Gómez-Adorno, and D. Pinto, "Soft similarity and soft cosine measure: Similarity of features in vector space model," *Computación y Sistemas*, vol. 18, no. 3, pp. 491–504, 2014.

[109] N. Shervashidze, T. Petri, K. Mehlhorn, K. M. Borgwardt, and S. Vishwanathan, "Efficient graphlet kernels for large graph comparison," in *Proceedings of the International conference on artificial intelligence and statistics*, pp. 488–495, 2009.

[110] J. Lugo-Martinez and P. Radivojac, "Generalized graphlet kernels for probabilistic inference in sparse graphs," *Network Science*, vol. 2, no. 02, pp. 254–276, 2014.

[111] D. Zhou, D. Zhong, and Y. He, "Biomedical relation extraction: From binary to complex," *Computational and mathematical methods in medicine*, vol. 2014, 2014.

[112] 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*, vol. 33, no. suppl 1, pp. D514–D517, 2005.

[113] S. Aymé and J. Schmidtke, "Networking for rare diseases: a necessity for europe," *Bundesgesundheitsblatt - Gesundheitsforschung - Gesundheitsschutz*, vol. 50, no. 12, pp. 1477–1483, 2007.

[114] E. Bragin, E. A. Chatzimichali, C. F. Wright, M. E. Hurles, H. V. Firth, A. P. Bevan, and G. J. Swaminathan, "Decipher: database for the interpretation of phenotype-linked plausibly pathogenic sequence and copy-number variation," *Nucleic acids research*, vol. 42, no. D1, pp. D993–D1000, 2014.

[115] P. Wang, W.-F. Lai, M. J. Li, F. Xu, H. K. Yalamanchili, R. Lovell-Badge, and J. Wang, "Inference of gene-phenotype associations via protein-protein interaction and orthology," *PLoS ONE*, vol. 8, p. e77478, 10 2013.

[116] S. Anderson, A. T. Bankier, B. G. Barrell, M. H. L. de Bruijn, A. R. Coulson, J. Drouin, I. C. Eperon, D. P. Nierlich, B. A. Roe, F. Sanger, P. H. Schreier, A. J. H. Smith, R. Staden, and I. G. Young, "Sequence and organization of the human mitochondrial genome," *Nature*, vol. 290, pp. 457–465, Apr 1981.

[117] R. W. Taylor and D. M. Turnbull, "Mitochondrial DNA mutations in human disease," *Nat Rev Genet*, vol. 6, pp. 389–402, May 2005.

[118] D. Wallace, G. Singh, M. Lott, J. Hodge, T. Schurr, A. Lezza, L. Elsas, and E. Nikoskelainen, "Mitochondrial DNA mutation associated with Leber's hereditary optic neuropathy,"

Science, vol. 242, no. 4884, pp. 1427–1430, 1988.

[119] I. J. Holt, A. E. Harding, and J. A. Morgan-Hughes, "Deletions of muscle mitochondrial DNA in patients with mitochondrial myopathies," *Nature*, vol. 331, pp. 717–719, Feb 1988.

[120] K. S. Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of Documentation*, vol. 28, pp. 11–21, 1972.

[121] C. L. Smith and J. T. Eppig, "The mammalian phenotype ontology: enabling robust annotation and comparative analysis," *Wiley Interdisciplinary Reviews: Systems Biology and Medicine*, vol. 1, no. 3, pp. 390–399, 2009.

[122] J. Gillis and P. Pavlidis, "The impact of multifunctional genes on guilt by association analysis," *PloS one*, vol. 6, no. 2, p. e17258, 2011.

[123] J. Gillis and P. Pavlidis, "Characterizing the state of the art in the computational assignment of gene function: lessons from the first critical assessment of functional annotation (CAFA)," *BMC bioinformatics*, vol. 14, no. Suppl 3, p. S15, 2013.

[124] A. Krogh, B. Larsson, G. von Heijne, and E. L. Sonnhammer, "Predicting transmembrane protein topology with a hidden markov model: application to complete genomes," *Journal of Molecular Biology*, vol. 305, no. 3, pp. 567 – 580, 2001.

[125] A. Coletta, J. W. Pinney, D. Y. W. Solis, J. Marsh, S. Pettifer, and T. Attwood, "Low-complexity regions within protein sequences have position-dependent roles," *BMC Systems Biology*, vol. 4, no. 1, pp. 1–13, 2010.

[126] M. Bada, M. Eckert, D. Evans, K. Garcia, K. Shipley, D. Sitnikov, W. A. Baumgartner, K. B. Cohen, K. Verspoor, J. A. Blake, and L. E. Hunter, "Concept annotation in the craft corpus," *BMC Bioinformatics*, vol. 13, pp. 161–161, Jul 2012. 1471-2105-13-161[PII].

[127] M. A. Tanenblatt, A. Coden, and I. L. Sominsky, "The ConceptMapper approach to

named entity recognition.," in *The seventh international conference on Language Resources and Evaluation (LREC)*, 2010.

APPENDIX A

Data for GO term prediction

Each method was trained/tested using the same set of features and labels, prepared as described below.

A.1. GO annotations

We extracted GO annotations from the Gene Ontology web site[1] and Uniprot-goa[2]. We ignored the root categories of the three subontologies. We also ignored the category "protein-binding" according to what was done in the CAFA1 challenge [13]. This category is a very general category (i.e in level 2) and there is a reasonable fraction of proteins which is annotated with this term as its most specific term which leads to unusually high performance with methods that only cater to prevalence bias [122, 123]. We excluded all annotations that were obtained by computational methods and also did not include GO annotations with evidence codes that suggest the annotation was derived from an interaction assay (i.e., only EXP, IDA, IMP, IGI, IEP and TAS evidence codes were included). We also removed GO categories that were not annotated to 10 or more proteins.

A.2. Trans/Loc

We generated three sets of features using amino acid sequence properties: localization features, transmembrane features and low complexity features described elsewhere [32]. Information about protein localization can be informative of their associated functions due to the fact that many biological processes are known to be localized to certain cellular compartments [57]. We used the localization signals computed from the WOLF PSORT program [58]

[1] http://www.geneontology.org/
[2] http://www.ebi.ac.uk/GOA

as localization features. The number of transmembrane domains (the segments which are fully of partly located within the cell membrane) a protein has can be informative of function. For example, transmembrane proteins are known to be associated with functions that involve transport of various molecules. We used TMHMM [124] to predict the number of transmembrane domains of each protein and this number was associated with an indicator variable. Low complexity regions in the amino acid sequence of a protein are known to have an effect on protein function [125]. We used a sliding window approach (window length = 20) to identify the region with the lowest number of distinct amino acids and used the composition of that region as the representation of that region.

A.3. Homology

Homology can be loosely defined as shared ancestry and it is measured through sequence or structure similarity. Homology provides valuable information on function because it indicates the closeness to a common ancestor. Homology to other annotated proteins in other species was captured using an approach similar to GOtcha scores, as suggested in [24], and also used successfully by Radivojac's team in the first CAFA competition [13]. Each protein is characterized by a feature vector where the jth feature is a confidence score that the protein is similar to proteins that are annotated with the jth GO category. Let S_j be the set of all sequences annotated with GO category j and let $e(s, s')$ be the e-value reported by BLAST+ [53] for the alignment of sequences s and s' when querying s against the database containing s'. We define the e-max score for protein s and GO category j as:

$$e\text{-}max_j(s) = \max_{s' \in S_j}(-log(e(s, s')/10)).$$

This is a simpler version of *GOtcha* scores, observed to perform slightly better in preliminary experiments. The e-max scores efficiently integrate evidence for a given GO category across

multiple species, and a protein is represented as a vector of variables where component j provides the e-max-score for GO category j. When running BLAST+ we used *psiblast* with 1 iteration on a database composed of all annotated sequences from every species except the target species. For example, when running BLAST+ for computing e-max-scores for yeast, the query consisted of all annotated yeast sequences while the database was composed of the annotated sequences from rest of the species. This approach ensures that annotations of the test sequences are not leveraged by the classifier during the training phase. Additionally, we filtered out low complexity regions using the built-in *SEG* filter.

A.4. Network

In order to perform its function, a protein usually interacts with one or more other proteins. This means that looking at the functions of the interacting partners of a protein can provide valuable hints about the functions of that particular protein. We extracted protein-protein interactions and other functional association network data (protein-protein interactions, co-expression, protein co-occurrence, etc.) from BioGRID 3.2.106 [36], STRING 9.1 [60] and GeneMANIA 3.1.2 [3] in two species: human and yeast. The BioGRID database provides protein-protein interaction networks acquired from physical and genetic interaction experiments. STRING provides networks based on several different evidence channels (co-expression, co-occurrence, fusion, neighborhood, genetic interactions, physical interactions, etc.). For a given type of functional association data we combined edges from the two databases by taking the union of interactions from BioGRID and STRING and represented each gene by a vector of variables, where component i indicates if the corresponding protein interacts with protein i in the combined network. The GeneMANIA website provides a

[3] http://pages.genemania.org/data/

large number of protein-protein interaction and association networks generated using several types of evidence: co-expression, co-localization, genetic interactions, physical interactions and predicted interactions. A gene is represented by a vector of variables for each network, where component i indicates if the corresponding protein interacts with protein i.

A.5. Literature

If a protein name is mentioned in the close vicinity of a functional category in the literature (a *co-mention*) it is likely that this protein may be associated with that function. To leverage this source of information we extracted protein-functional category co-mentions using the natural language processing pipeline described in [50] from all Medline abstracts available on 23-Oct-13 and full-text articles available from the PubMed Open Access Collection (PMCOA) on 06-Nov-13.

The co-mentions we considered are the pairs of protein name and GO category mentioned in the document within a specified span. We used two spans: sentence and non-sentence. Sentence co-mentions only consider proteins and GO categories mentioned in the same sentence while non-sentence co-mentions consider such pairs mentioned together in the same paragraph or abstract, but not within the same sentence.

We provided the abstracts and full-text documents (one paragraph at a time) as input to our text mining pipeline. It detected protein entities in the given text and mapped them to UniProt identifiers through a specially constructed dictionary. This dictionary consists of all yeast and human target proteins from CAFA2. Similarly, another dictionary based on Gene Ontology categories available on 13-Nov-13 is used to recognize GO categories in the text. Finally the pipeline outputs the protein names and GO categories along with the counts of how often they appear together within the sentence or non-sentence spans.

A protein is represented by two vectors in which the ith element in each vector gives the number of times that protein is co-mentioned with the GO category i within either a sentence or non-sentence span. The vectors are concatenated to form the overall representation.

APPENDIX B

Text-mining pipeline

The text-mining pipeline was developed using Apache UIMA version 2.4[1]. It is used to extract two types of literature features, co-mentions and bag of words. We provided the abstracts and full-text documents (one paragraph at a time) as input to our text mining pipeline. It detects protein entities in the given text using a LingPipe trained on CRAFT [126] and maps them to UniProt identifiers through a specially constructed dictionary. This dictionary consists of all human target proteins from CAFA2.

Next, mentions of GO categories in the text are identified by using a state-of-the-art concept recognition system for GO term recognition which uses ConceptMapper [127]. We used a GO term dictionary based on Gene Ontology categories available on 13-Nov-13 as input to ConceptMapper system. The dictionary simply uses GO terms and their synonyms available from GO web site, as well as many additional synonyms generated through a manually generated set of rules by looking at the variation between terms and their annotated examples in the corpus.

[1]http://uima-framework.sourceforge.net/

APPENDIX C

Data for HPO term prediction

Each gene/protein was characterized by several sets of features generated using four data sources: Network, GO, literature and variants, which are described below. We used the UniProt ID mapping service[1] for mapping genes to proteins.

C.1. HPO annotations

Gene-HPO annotations were downloaded from the HPO website. We ignored the global root term ("ALL") and root terms of the three subontologies. We also removed terms that were not annotated to 10 or more genes. Then we mapped the genes to proteins and generated corresponding protein-HPO annotations (see Table C.1).

Table C.1. Number of genes, unique terms and annotations. The "unique terms" column provides both the number of terms and the number of leaf terms; the "annotations" column provides the number of annotations, as well as their number when expanded using the true-path rule.

Subontology	Annotated genes	Unique terms	Annotations
Organ	2,768	1,796/1,337	213,000/60,000
Inheritance	2,668	12/10	3,600/3,300
Onset	926	23/20	1,700/1,400k

C.2. Network

We extracted protein-protein interactions and other functional association network data (i.e., co-expression, co-occurrence, etc.) from BioGRID 3.2.106 [36], STRING 9.1 [60] and GeneMANIA 3.1.2 [2] databases.

[1] http://www.uniprot.org/mapping/
[2] http://pages.genemania.org/data/

The BioGRID database provides protein-protein interaction networks acquired from physical and genetic interaction experiments. STRING provides networks based on several different evidence channels (co-expression, co-occurrence, fusion, neighborhood, genetic interactions, physical interactions, etc.). We combined edges from the two databases by taking the union of interactions from BioGRID and STRING and represented each gene by a vector of variables, where component i indicates if the corresponding protein interacts with protein i in the combined network.

The GeneMANIA website provides a large number of protein-protein interaction/association networks generated using several types of evidence: co-expression, co-localization, genetic interactions, physical interactions and predicted interactions. A gene is represented by a vector of variables for each network, where component i indicates if the corresponding protein interacts with protein i with respect to that particular network.

C.3. GO

We extracted Gene Ontology [7] annotations from the Gene Ontology web site[3] and Uniprot-goa[4]. We excluded all annotations that were obtained by computational methods. A gene is represented as a vector of indicator variables in which variable i is 1 if it is annotated with GO term i.

C.4. Literature

We used two different sources for generating literature features: abstracts extracted from Medline on 10-23-13 and full-text articles extracted from PubMed Open Access Collection (PMCOA) on 11-06-13. A natural language processing pipeline was utilized to characterize

[3]http://www.geneontology.org/
[4]http://www.ebi.ac.uk/GOA

genes/proteins by same-sentence word occurrences extracted from these sources, forming a bag-of-words representation for each gene [50]. First, all words were lower-cased and stop words were removed. Then they were further filtered to keep only the low frequency words (i.e., words that are present only in less than 1% of the proteins in the data). A gene is represented by a vector in which the element i gives the number of times the word i occurred in the same sentence with that gene/protein.

C.5. Variants

We extracted all the disease variants in the human genome and their associated diseases from UniProt[5]. This data provides variants that have been found in patients and the disease-association is reported in literature. We also extracted gene-disease associations from the HPO website. This data associates a protein with diseases that are known to occur when the associated gene is mutated. To generate features from this data, we first extracted for each protein p_i its set of associated diseases (D_i) from the protein-disease associations. Then we retrieved the set of disease variants (V_i) associated with all diseases in D_i from the UniProt disease variants data. Finally, each gene was represented by a vector in which element j indicates if the variant j is in V_i .

[5]http://www.uniprot.org/docs/humsavar