Combine and conquer: methods for multitask learning in biology and language

Meghana Kshirsagar

August 12, 2015

CMU-LTI-15-009

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213
www.lti.cs.cmu.edu

THESIS COMMITTEE
Jaime G. Carbonell (Chair)
Judith Klein-Seetharaman (University of Warwick)
Jeff Schneider (Robotics Institute, CMU)
Gunnar Rätsch (Memorial Sloan Kettering Cancer Center)

Submitted in partial fulfillment of the requirements for the degree of
DOCTOR OF PHILOSOPHY

Abstract

Generalizing beyond an individual task and borrowing knowledge from related tasks are the hallmarks of true intelligence. Knowing one language makes it easier to learn other languages, similar sports require learning similar skills to master them, etc. While building supervised machine learning models, such opportunities arise in machine translation for similar languages, modeling molecular processes of related organisms, predicting links across different types of social networks, extracting information from related sources of data etc. There are several benefits of borrowing from related tasks, beyond the ability to generalize. In many supervised learning applications the main bottleneck is insufficient labeled data (i.e annotations) to learn a good model. Obtaining additional labels is often expensive, requires resources and can be very time consuming. However there are often at hand, other related applications which have plentiful labeled information that can be utilized. Multitask learning [Caruana, 1997] is a family of machine learning methods that addresses this issue of building models using data from multiple problem domains (i.e 'tasks') by exploiting the similarity between them. The goal is to achieve performance benefits on the low-resource task called the target task or on all the tasks involved.

This thesis focuses on developing and extending multitask learning models for various types of data. Two diverse applications motivate the methods in this work. The first one is, modeling infectious diseases via host-pathogen interactions where we study molecular level interactions between pathogens such as bacteria and viruses and their hosts (such as humans). The question we address is: *Can we model host-pathogen interactions better by leveraging data across multiple diseases?*, towards which we develop new methods to jointly learn models across several hosts and pathogens. The other application that we consider, semantic parsing, is the process of mapping a natural-language sentence into a formal representation of its meaning. Since there are several ways to represent meaning, there are several linguistic resources (one per representation) and each annotates a different text corpus. Here we focus on: *how to leverage information from resources with different representations and distributions?* Overall, we explore various mechanisms of sharing information across tasks: by enforcing priors, structured similarity, feature augmentation and instance-level transfer. We show how our models can be interpreted to obtain additional insights into the problems.

In terms of impact, we build the first models for host-pathogen interactions for several bacteria and viruses and the first to involve a plant host. The methods we develop perform better than other computational methods. The predictions we obtain for the bacteria, *Salmonella* were validated by laboratory experiments, and we find that our model has a significantly higher recall compared to other computational models. Since there is very little known about how plant immune systems work, we exploit the data from other hosts. With the predictions from our model, we compare two hosts: human and the plant host *Arabidopsis thaliana*. The model we develop for viral pathogens leads us to some interesting insights on pathogen-specific protein sequence structures. Finally, leveraging several linguistic resources leads us to achieve impressive gains for the task of frame semantic role labeling.

Acknowledgements

They say, it's not the destination but the journey that matters the most. The same can be said of a PhD (which is a reasonably long and complex journey). For me, this journey would not have been possible without the support of my advisors, peers, family and friends. I am grateful to be able to acknowledge their contributions here.

A majority of my thesis work has been in the field of computational biology - at the confluence of machine learning and molecular biology. I was fortunate to have found the "right" research area - one that I am passionate about and deeply care for. For this, I am extremely grateful to my advisors, Jaime Carbonell and Judith Klein-Seetharaman for introducing some exciting problems and opportunities for contribution in this vast and upcoming area. I still remember my first research discussion with Jaime on transfer learning and with Judith on *Salmonella* and how much it had excited me. I came in with no prior experience in bioinformatics and little relevant background in biology. I am very thankful to them for having placed their trust in me, and for taking me on as an advisee. I transferred to CMU from UIUC (where I spent a year, unsuccessful in finding a research topic) and I am happy to have found the right advisor and research combination here.

Jaime's expertise in extremely diverse subject areas such as biology and language (to name a few) has made it possible for me to think of problems in these as manifestations of a more general underlying computational setting. Working with him has exposed me to a variety of rich problems and has led to my confidence in machine learning algorithms as a powerful ally for many unconventional data sources (industrial safety is one example). His emphasis on making methods simple and general has influenced my research greatly in later years. Jaime's advising style of nudging in just the right amount has led me to gain skills needed as an independent researcher, to view problems at an abstract level. I am thankful for his flexibility in always making time for meetings and for last minute discussions on papers. I would also like to thank Jaime's group for discussions and suggestions in group meetings, in particular Keerthiram Murugesan, Wan Li, Selen Uguroglu.

In my early years, I am thankful to Judith for making it easier for me to navigate the overwhelming sea of biological concepts and terminology by pointing me in the relevant direction. Judith is one of the rare biologists who sees computation as an exciting companion for biology (I have hence met many that either regard it suspiciously or that see it as a mere tool). Her excitement and belief in computational modeling of complex biological phenomena has influenced some of my ambitious efforts towards jointly modeling some daringly disparate hosts and pathogens. I am also thankful to Judith's research group, both at UPitt and Warwick for being my "quick reference" for concepts and tools. A very special thanks and gratitude to Sylvia Schlekar for being very patient with paper submissions, especially when the computational results were bad and when there were mistakes in my models. Most of the wet-lab experiments and the *Salmonella* data used in my thesis came from her diligent efforts. I am also thankful to her for hosting me in Jülich. I would also like to thank Joan Planas, Naveena Yanamalla, Dariush Mohammedyani for their help.

I benefitted greatly from discussions with other computational biologists here at CMU, in particular from Ziv Bar-Joseph and his group who have also been fun companions at conferences: Anthony Gitter, Saket Navlakha and Siddhartha Jain whose research being very related has influenced my own work.

In my last year I worked on frame semantic parsing, thanks to an early discussion of various multitask settings with Jaime and encouragement from Chris Dyer – whose approachable personality and excitement led me to pursue this direction further. Chris has a very broad understanding of machine learning and NLP and discussions with him have always led to new knowledge and ideas even in the computational biology aspects of my research. Very special thanks to Nathan Schneider and Sam Thomson, first for being friends and then for being such nice collaborators (indeed, it was all those lunches and dinners together that rubbed some NLP onto me). It was fun to write the ACL paper, and I am thankful for their help and patience in all the technical aspects that were very new to me. I would also like to thank Noah Smith for his help and the most for inspiring me to write better - concise and accurate.

Before I began my PhD, one of my main mentors was my Masters thesis advisor at IIT Bombay - Prof. S. Sudarshan. My basic understanding of research, of the best practices and priorities has come from his guidance. I will always be indebted to him for being such an inspirational advisor.

I made many friends here at CMU and also during my one year at UIUC. First and foremost, I would like to thank Kriti Puniyani would brought to me the idea of switching from UIUC to CMU and helping with it, and for also inspiring me to take up computational biology and for being a great TA and help in technical difficulties. I want to thank all my other friends as well, for their company made my PhD years a pleasant and diverse experience – Ravi Tumkur, Lavanya Anandan, Rajesh Karmani and many others at UIUC. At CMU – my batchmates and great friends, Dani Yogatama for constant dinner company and academic gossip, Derry Wijaya for being my fun 'offmie' and sounding board, Hobarters - Sunayana Sitaram and Anjali Menon for being 'counsellors' in the toughest of times and providing delicious new foods, Bhavana Dalvi for being together for so long as a great friend and flatmate, Ruta Desai for all the fun projects, random discussions and with Wolfgang Richter for making LaptopRehab possible, and with Lara Martins for the roadshows, Prasanna Kumar for providing constant potassium nourishment through bananas, organizing LTI events, my pingpong buddies and dinner companions, Nathan Schneider, Waleed Ammar, Sam Thomson. For daily scrum in my last semester, Subhodeep Moitra and Reyyan Yeniterzi. There are many others who have helped and been there and I want to thank you all.

Last but not the least, I want to thank my parents and my brother – for their unconditional love, for having faith in me always, for everything they have done for me throughout my life and being here for my defense. My love, Alekh for inspiring me to do my best, for being a part of everything I have withstood and achieved in the last few years.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Humans acquire knowledge and skills by categorizing the various problems/tasks encountered, recognizing how the tasks are related to each other and taking advantage of this organization when learning a new task. For instance, a person that knows how to drive a car will use that knowledge while learning to drive a truck obviating the need to learn every aspect of driving from scratch. Often, learning a new task can result in improvements in the ability to perform other tasks learned in the past. By transferring knowledge across related learning tasks, a learner can become "more experienced" and generalize better [Thrun, 1996].

Statistical machine learning methods can also benefit from exploiting such similarities in the learning problems. There are several benefits of borrowing from related tasks, beyond the ability to generalize. In many supervised learning applications the main bottleneck is insufficient labeled data (i.e annotations) to learn a good model. Labeled data typically comes from manual annotation, surveys, experiments measuring physical quantities such as temperatures, pressures, biological properties etc. These data collection efforts are often very expensive and time consuming (especially those requiring experiments). In such scenarios, labeled data available in other related problems can be tapped into. Such opportunities for sharing knowledge arise in several problems.

Consider the problem of web-page classification, where the goal is to automatically classify a given web-page into one (or more) of several categories [1]: 'clothing', 'movie', 'music', 'soccer', 'business', 'product' etc. Building good supervised classifiers requires a large number of training examples for each category – i.e labels assigned to web-pages manually by human annotators. Given the large number of possible categories, manual annotation will be a herculean effort, that is very expensive and time-consuming. However, we know that certain categories are related, for instance: webpages labeled 'soccer' and 'basketball' will both have sports related terms ("team", "player", "win" etc). The task of classifying 'soccer' webpages is thus similar to that of classifying 'basketball' webpages. If we can couple these tasks, very few labeled examples would be sufficient to obtain good classifiers for both.

The term Multitask Learning (MTL) was coined by Caruana [1997] to refer to learning methods and algorithms that can share information among related tasks and help to perform those tasks together more efficiently than in isolation. The conventional machine learning approach of learning each task independently, oblivious

[1] The DMOZ directory has a list of 1,017,500 different categories of web-pages

to other related tasks is often called Single task learning (STL) or independent task learning. Some more examples of 'multitask' settings are:

1. Modeling users' preferences for movies (i.e the 'Netflix' problem). Here, a task is predicting one user's ratings; often there are very few (or no) ratings available for most users. Yet, the ways different people make decisions about movies is related as there will be common patterns in their interests.

2. Task similarities also arise while studying the biology of organisms. The theory of evolution tells us that many organisms have evolved from the same ancestor species; this causes them to exhibit some common characteristics. An example of such a shared characteristic is: the splice sites (i.e gene boundaries) in DNA sequences have similar properties across related organisms. Here, the biological phenomenon being studied: 'detecting DNA splice sites in *an organism*' is a 'task'. Widmer et al. [2010] use genome sequence data from 15 eukaryotic organisms (i.e 15 tasks) to build a multitask model and show that one can indeed significantly improve the splice-site prediction performance compared to traditional approaches that look at individual organisms.

A very related problem called Transfer learning (TL) involves extracting knowledge from one or more *source* tasks and *transferring* it to a *target* task. In contrast to multi-task learning, where all tasks are learned simultaneously, transfer learning cares most about building a good model on the target task. The roles of the source and target tasks are not symmetric in transfer learning. The NIPS workshop on "Learning to Learn" [2] was one of the first to focus on the need for lifelong machine learning methods that retain and reuse previously learned knowledge. Published literature exploring this general idea has used different terms, such as 'life-long learning' [Thrun, 1996], 'inductive transfer' [Mitchell, 1980], 'transductive learning' [Vapnik, 1998], 'knowledge transfer' to refer to slightly different manifestations of the same problem. Transfer learning is sometimes also called 'domain adaptation' in some research communities, and can be viewed as a more targeted version of multitask learning.

What kind of information can be shared between tasks?

At a high level, the following are some possibilities:

- The *features* found to be relevant for learning one task can be used to learn another task

- The model parameters learned for a task can serve as a *prior* for other tasks (in the 'transfer' learning setting)

- Task *parameters* can lie close to each other in some geometric sense

- The *structure* of the underlying statistical model, such as: dependencies between the variables, the probability distribution of the model's parameters can be shared by several tasks

[2] http://socrates.acadiau.ca/courses/comp/dsilver/NIPS95_LTL/nips95.workshop.pdf

How is information shared between tasks?

Any MTL method has two main objectives: to minimize the training error (i.e empirical risk minimization) and to enforce task relatedness. While some methods achieve this in a pipeline fashion, where independent models are learned for tasks and then the task structure is enforced as a post-processing step (possibly followed by more model estimation steps), most recent methods perform this jointly. The task structure is enforced/learned along with parameter estimation. Objective functions which express this have the following general form, where θ_t is the set of parameters corresponding to the task t:

$$\min_{\theta_1,\ldots\theta_t} \sum_{t\in\{\text{all tasks}\}} l_t + \lambda \;\; \Omega(\theta_1,\ldots\theta_t)$$

The term l_t can represent a loss function such as least squared error. Ω is the mechanism by which the tasks are coupled together and is sometimes called a 'task regularizer' because it introduces a bias that favours models with a certain structure. The parameter λ controls the extent to which the multitask structure is enforced. In a bayesian setting, the objective function will involve the negative log likelihood derived for a probabilistic model that explains how the tasks are related to each other. A simple example of a function Ω that couples two tasks is:

$$\Omega(\theta_1, \theta_2) = \|\theta_1 - \theta_2\| \tag{1.1}$$

Here we are encoding the knowledge that the models corresponding to the two tasks have similar parameters. From the perspective of each task t, this term is bounding the variance of the task parameters θ_t.

Focus of this thesis

The overarching goal of this thesis is to *discover the underlying multitask models that can best explain the observed relationships between real-world tasks*. We find how task-relationships in real-world problems can be connected to mathematical models of the data that manifest similar relationships. We focus on the following two very distinct problems:

- *Modeling molecular mechanisms of infectious diseases*: Infectious diseases are caused by pathogens such as bacteria and viruses. At the molecular level, pathogenesis involves the pathogen introducing its' proteins into the host cells, where they interact with the host's proteins thereby enabling the pathogen to obtain nutrients, replicate and survive inside the host. These 'molecular mechanisms used by a pathogen' i.e host-pathogen protein interactions represent a task, and there is one task per pathogen. Though the microbial world is very diverse, we find that the infection mechanisms employed by different pathogens share some aspects: (1) they have similar proteins, by virtue of their common ancestors (2) they infect similar biological processes in their hosts. These biological similarities allow us to define task relationships which are then used to learn the tasks jointly. This challenging domain is the central focus of this thesis.

- *Semantic understanding of natural language text*: Semantic parsing is a problem in natural language processing where the objective is inferring the meaning of a

sentence. For instance, the sentence John stole a car indicates that 'John' is a `Thief`, who **stole** the `Property`: 'car'. The verb 'stole' is the *relation* or action connecting the two entities 'thief' and 'property'. The same meaning can be represented in different forms: we could say 'John' is a `Perpetrator` who stole the `Item`: 'car'. As a result of this, the various lexical resources that have been built for semantic parsing, use different representations though they capture similar semantics. Each resource differs in: the types of relations it focuses on, the text corpus that was annotated, the distribution of the entity types etc. Combining the information from all resources thus has obvious benefits in terms of improved coverage compared to what can be achieved with a single resource (due to its limited focus). We can consider a task to be: 'semantic parsing using one representation' and the different tasks (one task per representation) are related via the shared semantics that they encode.

We build several multitask and transfer learning models for the first problem. For semantic parsing, we develop transfer learning (domain adaptation) based models. The key contributions of this thesis are:

1. The methods in this thesis are one of the first to combine host-pathogen protein interactions data across several pathogens and hosts

2. Our multitask learning model for host-bacterial interactions involves a novel task regularizer that incorporates the biological hypothesis that: infection mechanisms are similar across pathogens

3. For multitask graph completion, we develop a new model that learns a lower dimensional representation of the data. Our model significantly outperforms methods from prior work

4. The human-bacterial interaction models are the first ever models involving these bacterial organisms

5. By combining data from multiple resources, we improve the state-of-the-art in semantic parsing

6. Unlike most prior work on modeling host pathogen interactions, our models are interpretable and lead us to interesting hypotheses and insights into the problem

7. Our code and data is made publicly available

MTL and TL literature: methods

Early work on MTL used a hidden layer neural network with few nodes and a set of network weights shared by all the tasks [Caruana, 1997, Thrun and Pratt, 1998, Baxter, 2000]. Later methods were based on variance regularizers [Evgeniou and Pontil, 2004, Maurer, 2006] which are based on assumptions similar to that of the simple regularizer Ω from Equation 1.1. Other assumptions on task parameters are: that they lie in a low dimensional subspace [Argyriou et al., 2008, Liu et al., 2009], or on a manifold [Agarwal et al., 2010]. Liu et al. [2009] use a $\ell_{1,2}$ regularization over the parameter matrix (consisting of parameter vectors from all tasks). Ando and Zhang [2005] present

a framework for learning predictive functional structures from multiple tasks that also exploits unlabeled data. Jalali et al. [2010] propose a model for joint learning of multiple linear regression functions. They decompose the parameter matrix into two components which are regularized independently via different norms.

Another widely studied approach for multi-task learning is the task clustering approach [Bakker and Heskes, 2003, Jacob et al., 2009, Kumar and Daume III, 2012]. Its main idea is to group the tasks into several clusters and then learn similar data features or model parameters for the tasks within each cluster. An advantage of this approach is its robustness against outlier tasks because they reside in separate clusters that do not affect other tasks.

A number of MTL approaches are Bayesian, where a probability model capturing the relations between the different tasks is estimated simultaneously with the models' parameters for each of the individual tasks. In [Allenby and Rossi, 1998, Bakker and Heskes, 2003] a hierarchical Bayes model is estimated which assumes that the parameters of all tasks are sampled from an unknown Gaussian distribution. Task relatedness is captured by the Gaussian distribution: the smaller the variance of the Gaussian the more related the tasks are. Bonilla et al. [2007] propose a Gaussian process based method to model and learn task relationships in the form of a task covariance matrix. Zhang and Schneider [2010] attempt to learn the full task covariance matrix and use it in learning of predictor functions by placing a matrix variate prior on the task parameter matrix.

Recent work [Maurer et al., 2013] on jointly learning features and models for multiple tasks uses ideas from sparse coding and dictionary learning. They assume that the tasks parameters are well approximated by sparse linear combinations of the atoms of a dictionary. [Yuan et al., 2012] address the problem of visual classification via a multitask joint sparse representation model that combines multiple features and/or instances.

While the focus on MTL methods is often on task relationships, Transfer Learning (TL) methods investigate what to transfer. The ability to transfer from the source task(s) to the target task depends on how much the tasks differ and in what way. There has been a lot of theoretical work characterizing the distance between tasks and its relationship to the classification error on the target task [Ben-David et al., 2007, Crammer et al., 2008, Mansour et al., 2009, Ben-David et al., 2010]. The setting where there is a covariate shift (i.e distribution of the features changes) in the target task has seen a lot of work. Instance-based transfer learning methods have been proposed that reweight the source instances to indicate their relevance for the target task [Fan et al., 2005, Huang et al., 2007, Sugiyama et al., 2008, Cortes et al., 2008]. Other ways in which the tasks can differ are: the distribution of the output (i.e labels) changes [Japkowicz and Stephen, 2002, Yu and Zhou, 2008], the conditional distribution (labels given features) changes [Jiang and Zhai, 2007]. More recent work has looked at combinations of these settings [Zhang et al., 2013, Wang and Schneider, 2014].

On the applications front, approaches that transfer feature representations try to learn a good feature representation for the target domain that encodes the knowledge to be transferred. Blitzer et al. [2006] proposed the structural correspondence learning (SCL) algorithm, which extends Ando and Zhang [2005], to make use of the unlabeled data from the target domain to extract some relevant features that reduce the distance between the tasks. Daumé [2007] proposed a feature augmentation approach, that learns task-specific weights, for NLP problems. Such feature augmen-

tation is now widely used in supervised domain adaptation scenarios. Uguroglu and Carbonell [2011] find the features that vary the most between the source and target tasks (i.e differently distributed features) using a maximum mean discrepancy based method. This allows them to use invariant features for the target task. Wu and Dietterich [2004] exploit the plentiful low quality source task data for image classification problems, where the target task data is inadequate. Parameter-transfer approaches [Schwaighofer et al., 2004, Raina et al., 2006, Bonilla et al., 2007] assume that the source tasks and the target tasks share some parameters or prior distributions of the hyperparameters of the models. The transferred knowledge is encoded into these parameters and priors. The two popular regimes of transfer: parameter transfer and representation transfer have been theoretically analyzed in recent work [Maurer et al., 2013], with Pentina and Lampert [2014] proposing PAC-style generalization bounds to analyze lifelong learning algorithms.

The recent developments in the field of deep learning are also relevant to multi-task learning: both deep learning and multi-task learning show that we can leverage auxiliary tasks to help solving a task of interest [Bottou, 2014]. There has been work on learning representations that benefit extraction from a variety of datasets [Bordes et al., 2012], representations that benefit a range of natural language processing tasks [Collobert et al., 2011]. Our work differs from these in that we do not learn representations but use and develop new features that work well in our problems, which suffer from data scarcity issues. Approaches from deep learning that have worked successfully on real-world datasets have typically involved problems where there is plentiful data to efficiently learn these parameter-rich models with several layers and non-linear functions.

The most recent and relevant prior work to the specific problems and approaches that we consider, has been cited in the appropriate chapters.

Thesis map

We begin with an introduction to the concepts, challenges and approaches concerning the problem of building models for infectious diseases. This problem forms the main focus of this thesis, and we cover the various aspects of computational prediction of host-pathogen interactions in Chapter §2. These key aspects are common to all the multitask and transfer learning approaches that we present in subsequent chapters. Chapter §3 presents the Multitask Pathway based learning approach, that combines host-pathogen interactions data from several bacterial species using domain knowledge related to their infection mechanisms. In Chapter §4, a different perspective of this problem is presented that uses matrix decomposition based methods to share information across various viruses. While the first two chapters use data coming from the host species: human, the next chapter §5 involves a new host species: the plant *Arabidopsis thaliana*. This is a transfer learning setting, as we do not have any supervised data available for plant-pathogen interactions. We exploit interactions data from various hosts and pathogens to build a model for the target task involving the plant host. Continuing in this transfer theme, Chapter §6 presents a different application: semantic analysis of natural language text. Here, we present domain adaptation based approaches to improve the semantic parsing on a target domain of interest.

We hope this brief introduction has provided a high level intuition of the contributions of this thesis and intrigued the reader's curiosity to delve into the rest of the

material.

Chapter 2

Modeling infectious diseases via host-pathogen protein interactions

The biological functions and processes in our body involve several types of molecules and various interactions between them. Protein molecules are the workhorses that facilitate most biological processes in a cell. And among the molecular interactions in our body, the majority and the most vital ones are the ones between proteins. Some of the early studies towards understanding protein-protein interactions focused on discovering interactions within single organisms such as yeast cells, human cells. The interaction maps from these studies gave us a glimpse into the biological processes within an organism pertaining to: cell growth, proliferation, tissue formation, digestion and nutrient uptake, reproduction, blood circulation etc.

However this knowledge only forms part of the picture concerning living organisms, because an important part of our sustenance depends on how we counter infectious diseases – those that are caused by external agents called pathogens. Infectious diseases are a major health concern worldwide, causing millions of illnesses and deaths each year. Newly emerging viral diseases, such as swine H1N1 influenza, severe acute respiratory syndrome (SARS) and bacterial infections, such as the recurrent *Salmonella* and *E. coli* outbreaks not only lead to wide-spread loss of life and health, but also result in heavy economic losses.

Key to the infection process are host-pathogen interactions at the molecular level, where pathogen proteins physically bind with human proteins to manipulate important biological processes in the host cell, to evade the host's immune response and to multiply within the host. Comprehending protein interactions between host species such as mammals and pathogen species such as viruses and bacteria, is thus crucial in order to advance our understanding of pathogenesis. Laboratory based experimental methods have emerged towards studying these interactions. The discovery of the molecular interactions between *Herpesvirus* and human cells [Uetz et al., 2006], pioneered this field of cross-species interaction studies. These experimental studies can be broadly categorized into small-scale or large-scale methods. Small-scale methods are very time consuming but give very reliable results. Large-scale methods are more efficient as they screen a large number of proteins, but they tend to have a high false positive rate.

- Small-scale methods: These refer to biochemical, biophysical and genetic experiments that involve a few proteins. Examples of small-scale methods are: co-

immunoprecipitation (co-IP), far-western blot analysis, pull-down assays, co-crystalization.

- Large-scale techniques: High-throughput screening methods which work with the entire proteome of organisms, for example yeast two-hybrid (Y2H) assays. Affinity purification is another large-scale methods and is followed by mass spectrometry, microarray analysis, western blot.

Databases like PHI-base, PIG, HPIDB, PHISTO aggregate host-pathogen protein interactions from several small-scale and high throughput experiments via manual literature curation. These databases are valuable sources of information for developing models of the modus-operandi of pathogens.

2.1 Predicting host-pathogen PPIs

The most reliable experimental methods for studying protein-protein interactions (PPI) are often very time-consuming and expensive, making it hard to investigate the prohibitively large set of possible host-pathogen interactions – for example, the bacterium *Bacillus anthracis* which causes anthrax has about 2321 proteins which when coupled with the 25000 or so human proteins gives ≈60 million protein pairs to test, experimentally. Computational techniques complement laboratory-based methods by predicting highly probable PPIs. These techniques use the known interactions data from previous experiments and predict the most plausible new interactions. Experimental biologists use the highest-scoring interactions thus obtained and design experiments to validate these and study them further. This helps in ruling out the validation of the vast majority of unlikely PPIs.

In particular, supervised machine learning based methods use the few experimentally-discovered interactions as training data and formulate the interaction prediction problem in a classification setting, with target classes: "interacting" or "non-interacting". Features are derived for each host-pathogen protein pair using various attributes of the two proteins such as: protein sequences from Uniprot [UniProt Consortium, 2011], protein family from Pfam [Finn et al., 2010], protein structure and domain from PDB, gene ontology from GO database [Ashburner et al., 2000], gene expression from GEO [Barrett et al., 2011], interactions between protein families from iPfam [Finn et al., 2005], protein domain interactions from 3DID [Stein et al., 2011], to name a few. The general outline of the supervised PPI prediction procedure is illustrated in Figure 2.1.

In this setting, some of the important challenges from the machine learning perspective, that are generally encountered are:

1. Highly unbalanced classes since the set of "interacting" proteins is very small (for example, yeast has around 6000 proteins allowing for about 18 million potential interactions, but the estimated number of actual interactions is below 100,000)

2. Absence of a clear "negative" class since there is no notion of provably "non-interacting" protein pairs

3. Missing values in the features where certain properties of the proteins are not available for various reasons.

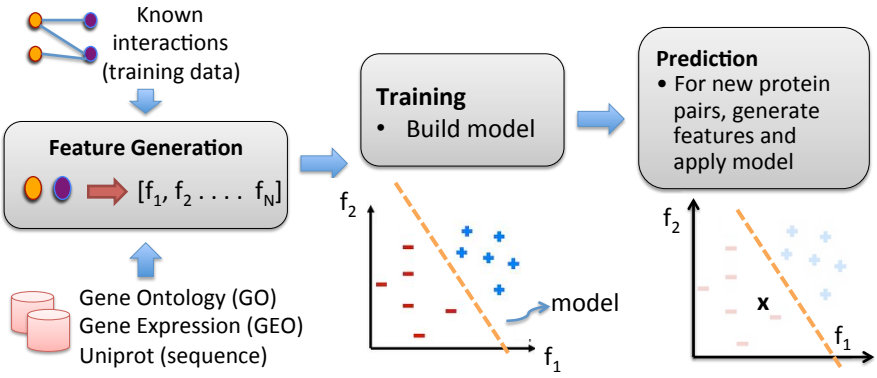


Figure 2.1: An example of a supervised classification method for predicting protein interactions

4. Sparse datasets: interactions data available in several databases is very small except for few well studied pathogens. For example: the PHI-base database covers 64 diseases but has only 1335 interactions, PIG covers only 12 pathogens.

2.2 Host-pathogen PPI datasets

Several data repositories like PHI-base [Winnenburg et al., 2008], PIG [Driscoll et al., 2009], HPIDB [Kumar and Nanduri, 2010], PHISTO [Tekir et al., 2012] aggregate host-pathogen protein interactions from several small-scale and high throughput experiments via manual literature curation. We use the PHISTO database for many of our datasets as it gives the UniprotKB protein ids for the interacting proteins. The other databases do not always list the host protein involved in a PPI. In this thesis, we work with nine different pathogens, five of which are bacterial species and the remaining are viruses. The genealogy of all bacterial species in PHISTO (version from 2012) is shown in Figure 2.2. We use four bacteria-human PPI datasets in our models (shown highlighted).

2.3 Features for host-pathogen PPI prediction

The host-pathogen PPI prediction problem is cast as a two-class classification problem: each protein pair $x = <p, h>$ is an instance belonging to either the positive, 'interaction' class or the negative, 'non-interaction' class. For each pair, we derived features which can belong to one of the three types: (a) feature derived on the pair x (b) features derived using either the host protein h or the pathogen protein p. Based on the source of information, we can also categorize our feature set into the following groups: (1) GO similarity, (2) graph based features using the human interactome, (3) gene expression, (4) sequence-$k$mer features, (5) features from protein family and protein domain interactions, (6) interolog based features.

1. **Protein sequence k-mer (or n-gram) features**: Since the sequence of a protein determines its structure and consequently its function, it may be possible to predict PPIs using the amino acid sequence of a protein pair. Shen et al. [2007] in-

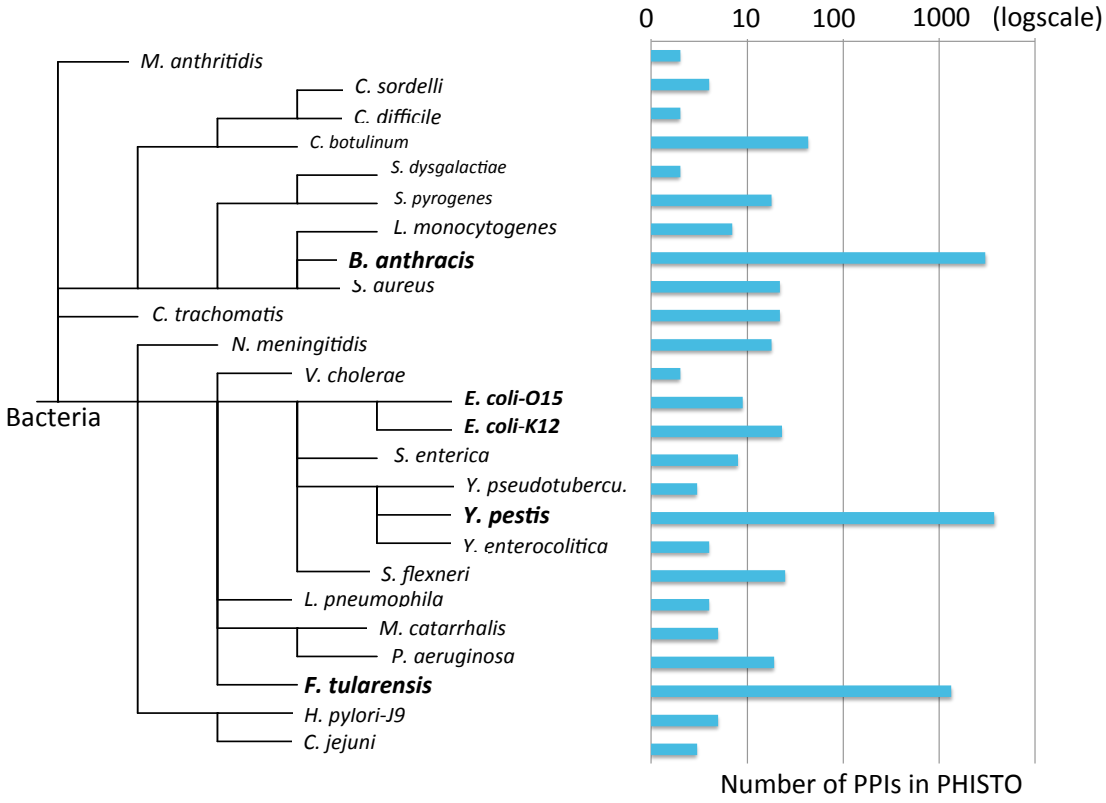


Figure 2.2: Phylogenetic tree of all the bacterial species and the number of bacteria-human PPIs (log-scale) in PHISTO database for each bacteria. Highlighted bacterial species represent the PPI datasets that we use for our models in Chapter §3.

troduced the "conjoint triad model" for predicting PPIs using only amino acid sequences. They partitioned the twenty amino acids into seven classes based on their electrostatic and hydrophobic properties. A protein's amino acid sequence is first transformed to a class-sequence (by replacing each amino acid by its class). For k=3, they count the number of times each distinct three-mer (set of three consecutive amino acids) occurred in the sequence. Since there are 343 (7^3) possible three-mers (with an alphabet of size 7), the feature vector containing the three-mer frequency counts will have 343 elements. To account for protein size, they normalized the counts by linearly transforming them to lie between 0 and 1. Thus the value of each feature in the feature vector is the normalized count for each of the possible amino acid three-mers. We use two-, three-, four-, and five-mers. For each hostpathogen protein pair, we compute the k-mer features for the two individual proteins and then concatenate the two feature vectors. Therefore, each hostpathogen protein pair had a feature vector of length at most 98 ($2 * 7^2$), 646, 4802, and 33614, in the cases of two-, three-, four-, and five-mers, respectively.

2. **GO similarity features**: These features model the similarity between the functional properties of two proteins. Gene Ontology [Ashburner et al., 2000] provides GO-term annotations for three important protein properties: molecular

function (F), cellular component (C) and biological process (P). We derive 3 types of features using these properties. For each of 'F', 'C' and 'P', GO similarity features were separately defined that compute the similarity of GO terms from host and pathogen proteins. The similarity between two individual GO terms was computed using the G-Sesame algorithm [Du et al., 2009]. This feature is a matrix of all the GO term combinations found in a given protein pair: $< p, h >$, the rows of the matrix represent GO terms from protein p and the columns represent GO terms from h.

3. **Graph based features using the human interactome**: These features are derived using only the host protein 'h' from the pair. Pathogens generally target host proteins that are important in several host processes; these host proteins interact with many other host proteins to carry out their tasks. This insight is captured in the form of three graph properties: *degree*, *between-ness centrality* and *clustering coefficient* of the host protein "node" in the host interactome graph. When the host is human, the interactome was downloaded from HPRD [Prasad et al., 2009]. The degree of a node is the number of its neighbouring nodes in the graph. The clustering coefficient of a node 'n' is defined as: the ratio of the number of edges present amongst n's neighbors to the number of all possible edges that could be present between the neighbours. Betweenness centrality for a node 'n', is defined as the sum over all pairs of nodes (u, v), the fraction of shortest paths from u to v, that pass through n. Mathematically, it is: $\sum_{u,v \in V \setminus n} \frac{\text{shortest\_paths}_n(u,v)}{\text{shortest\_paths}(u,v)}$. Intuitively, nodes that occur on many shortest paths between other vertices have higher betweenness than those that do not.

4. **Gene expression features**: The intuition behind this feature is that genes that are significantly differentially regulated upon being subject to *Salmonella*, are more likely to be involved in the infection process, and thereby in interactions with bacterial proteins. These features are derived using the gene of the host protein 'h' from the pair. We selected 3 transcriptomic datasets GDS77, GDS78, GDS80 from the GEO database [Barrett et al., 2011], which give the differential gene expression of human genes infected by *Salmonella*, under 7 different control conditions. The 3 datasets give us a total of 7 features: the dataset GDS77 has two samples representing two conditions and gave 2 features; datasets GDS78 and GDS80 had time series gene expression with 3 and 2 control conditions respectively – the time series in each condition was averaged resulting in 3 and 2 features, respectively. All datasets reported log-ratios and did not require further normalization.

5. **Features from PFam and protein domain interactions**: Two pair-level features were computed using protein family interactions from the iPFam database [Finn et al., 2005] and protein domain interactions from 3DID database [Stein et al., 2011]. For a pair, the first feature counts how many of all the possible interactions between the PFam families of the two proteins are present in iPFam. The second feature counts how many of the interactions between the domain sets of a protein pair are present as domain-domain interactions in 3DID.

6. **Interolog based features**: This feature uses known interactions between proteins from other organisms to infer new interactions. It was derived using the

FEATURE NAME (Count)	DESCRIPTION
Gene Ontology[a] (≈177 million)	Computed between GO terms of p and h. Let S = set of all GO terms, S_p = set of GO terms for protein 'p'. We set the entries of $S \times S$ corresponding to all pairs of GO terms from $S_p \times S_h$ to the similarity between the GO term pairs. Similarity between two individual GO-terms was computed using G-Sesame. Total number of features= $\|S\| \cdot \|S\|$
Network-based (3)	Uses three graph properties of h in the human protein interaction network: (1) '*degree*' = number of neighbours of h; (2) '*clustering coefficient*' = ratio of edges present amongst neighbours of h to all possible edges between them; (3) '*centrality*' = fraction of shortest paths in the network that pass through h
Gene Expression (7)	Derived using the gene of the human protein h. Uses three GEO datasets: GDS77, GDS78, GDS80 reporting differential gene expression of human genes infected by *Salmonella*, under 7 different control conditions.
Interologs (1)	Number of protein pairs from other species that are interologs of the given pair $<p, h>$.
Sequence n-grams (39200)	Used the "conjoint triad model" [Shen et al., 2007] to get n-gram features on the protein sequence for n=2,3,4,5. The amino-acid sequence is first converted to class-sequence and n-grams are computed separately for p and h and then concatenated to give a single feature vector (similar to [Dyer et al., 2011]) of size = $2(7^2 + 7^3 + 7^4 + 7^5)$.
Pfam interactions (1)	Counts the fraction of all possible interactions between the Pfam families of p and h, that are listed as known interactions in the iPfam database [Finn et al., 2005]
Domain interactions (1)	Similar to the above feature, computes the fraction of all possible domain-domain interactions between p and h that are present in the domain interactions database 3DID [Stein et al., 2011].

Table 2.1: Feature Set: summary of the various categories of features and the number of features in each category. h represents the host protein, and p represents the pathogen protein in a given protein pair $<p, h>$.

[a] sparse features, i.e only some of the millions of features are active in a single protein-pair

interologs information from the BIANA database [Garcia et al., 2010]. For a given pair 'x', if 'x_{hom}' a homologous protein pair involving any other organisms, BIANA uses the databases: BIND, DIP, IntAct to check if x_{hom} is an interacting pair. If yes, then x is an inferred interacting pair. For every pair 'x', this feature counts the number of homologous protein pairs x_{hom} that are interacting as per BIANA.

Table 2.1 shows a summary of the various features used in the predictive models we discuss in subsequent chapters.

2.4 The curse of missing negatives

Like many other problems in computational biology, PPI prediction suffers from the 'curse of missing negatives' – i.e we only have access to the positives. Most machine learning methods need a negative class (set of non-interactions) in order to identify the special characteristics of the positives (i.e interactions). However, there is no experimental evidence about proteins that do not interact, as it is difficult to design such an experiment that will rule out an interaction under all control conditions. There do exist protein domains that are known to not interact with each other, due to their conflicting tendencies to react with water (a hydrophobic domain is unlikely to come into contact with a hydrophyllic domain). Such negative interactions between protein domains have been catalogued and assembled in databases such as Negatome Blohm et al. [2013]. Although, each protein has several different domains which makes it hard to use such domain-level information to infer non-interaction at the protein-level.

To construct the negative class, we use a technique commonly used in PPI prediction literature. A set of random pairs of proteins is sampled from the set of all possible host-pathogen protein pairs, to serve as the negative class. The number of random pairs is chosen based on what we expect the interaction ratio to be. We chose a ratio of 1:100 meaning that we expect 1 out of every 100 random host-pathogen protein pairs to interact with each other. In general, there is no basis for choosing a more meaningful ratio, as there are few known interactions. We rely on previous work on better studied organisms, where a ratio of 1:100 was used, based on the number of known interactions. Further, prior studies [Tastan et al., 2009, Dyer et al., 2007, 2011] also use a similar ratio. This random selection strategy is likely to introduce about 1% false negatives into the training set, which is low enough to justify our choice of this heuristic. This ratio can be thought of as a parameter that can be changed as our knowledge of the size and nature of the host-pathogen interactome improves.

The ratio, called class skew is an important factor in any machine learning method. The choice of this parameter determines the properties of the resultant model. A very balanced class skew of 1:1 will result in a model that is over-predictive i.e has a very high false positive rate when applied on the target task. On the other hand, a very skewed setting of 1:1000 could give a lower false positive rate but is likely to have a poor recall as compared to models with lower class skews. This parameter thus offers a trade-off between the precision and recall of the resultant model. Our choice of a class ratio of 1:100 will result in a higher recall as compared to models trained on higher class skews. It will however, have some false positives. From a statistical perspective, a model trained with a high class skew such as 1:1000 will capture the distribution of the negatives since they hugely outnumber the positives. Since the negative class examples are not true negatives, the goodness of a model which depends mostly on noisy negatives is debatable. Computationally, the time required for training a model increases as we increase the number of examples. In the case of a high class skew such as 1:1000, there will be thousand times as many examples as the number of positives.

2.5 Evaluating PPI prediction methods

Our criteria to evaluate PPI prediction methods, do not use accuracy which measures the performance on both the classes. Since our datasets are highly imbalanced with a large number of negative samples[1], a naïve classifier that always says "no" would still have a very high accuracy. We instead use precision and recall computed on the interacting pairs (positive class). These quantities are defined as follows:

$$\text{Precision(P)} = \frac{\text{true positives}}{\text{predicted positives}}$$

$$\text{Recall(R)} = \frac{\text{true positives}}{\text{total true positives in data}}$$

$$\text{F-score} = \frac{2PR}{P+R}$$

We also report the area under the precision recall curve (AUC-PR). AUC-PR has been shown to give a more informative picture of an algorithm's performance than ROC curves in high class imbalance datasetsDavis and Goadrich [2006] such as ours. Note that the AUC-PR of a random classifier model on a dataset with class skew 1:100 is ≈ 0.01.

2.6 Motivation for multitask approaches

For a given disease, very little is known about PPIs between the pathogen and host proteins. However, such PPI data is available across many diseases, which leads us to ask the question: *Can we model host-pathogen PPIs better by leveraging data across multiple diseases?* This is of particular interest for lesser known disease where the data is really scarce. Figure 2.2 illustrates this: while there are many bacteria with very few PPIs, we notice that there are PPIs from other closely related bacteria which can be exploited. Combining information from many pathogens will also allow us to learn models that generalize better across disease by modeling global phenomena related to infection. Newly arising diseases where no data is available can also be modeled, thereby allowing us to derive important initial understanding about them.

[1]The positive:negative class ratio is 1:100

Chapter 3

Multi-Task Pathway-based Learning (MTPL)

To integrate interactions from several tasks (i.e diseases), we propose a method [Kshirsagar et al., 2013] that exploits the similarity in the infection process across the diseases. In particular, we use the biological hypothesis that similar pathogens target the same critical biological processes in the host, in defining a common structure across the tasks. In this work we consider host-pathogen PPI where the host is fixed and the pathogens are various bacterial species (see Figure 3.2(A)). The host organism that we consider is human and the bacterial species are: *Yersinia pestis*, *Francisella tularensis*, *Salmonella typhimurium*, *Escherichia coli* and *Bacillus anthracis*. Figure 3.1 lists their various characteristics.

Some recent work on infectious diseases has alluded to the hypothesis that *different pathogens target essentially the same critical biological processes in the human body*. The analysis by Chen et al. [2012b] suggests that HIV infection shares common molecular mechanisms with certain signaling pathways and cancers. Dyer et al. [2008] study bacterial and viral interactions with human genes and find infection mechanisms common to multiple pathogens. Jubelin et al. [2010] show how various bacterial cyclomodulins target the host cell cycle. The study by Mukhtar et al. [2011a] on plant pathogens, in particular *Arabidopsis* concludes that pathogens from different kingdoms deploy independently evolved virulence proteins that interact with a limited set of highly connected cellular hubs to facilitate their diverse life cycle strategies. Figure 3.2(B) illustrates an example depicting the commonality in various bacterial species, where they are targetting the same biological pathways in their human host.

This biological hypothesis which we hence-forth call the *commonality hypothesis* is exploited here to jointly learn PPI models for multiple bacterial species. We translate the hypothesis into a prior that will bias the learned models. We use a multi-task learning based approach, where each 'task' represents the protein interactions of one bacterial species with human. Our supervised learning based method jointly optimizes the prediction error over all tasks combined with a regularizer term that couples together all the tasks. The regularizer is a difference of histograms, with each histogram describing the distribution of host processes targeted by the various diseases. We use a convex concave procedure (CCCP) based algorithm to optimize this non-convex function. Our results indicate that introducing this 'bias' based on the biologically-derived hypothesis results in better predictive models.

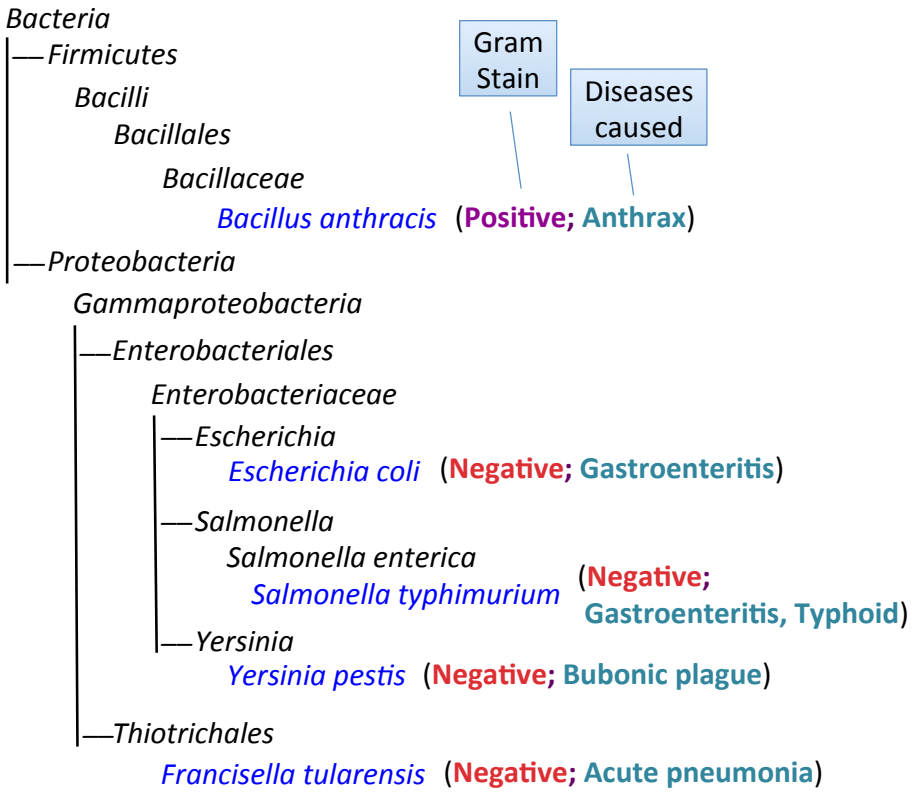


Figure 3.1: Genealogy of the bacterial species (highlighted in blue), for which we develop PPI prediction models in this chapter. The gram stain and the diseases caused by each bacterial species are also shown in paranthesis.

3.1 Related work

Most of the prior work in PPI prediction has focused on building models separately for individual organisms [Chen and Liu, 2005, Wu et al., 2006, Singh et al., 2006, Qi et al., 2006] or on building a model specific to a disease in the case of host-pathogen PPI prediction [Tastan et al., 2009, Qi et al., 2009, Dyer et al., 2007, Kshirsagar et al., 2012]. The use of PPI data from several organisms has predominantly been in the form of (1) features derived from various PPI datasets (2) use of common structural properties of proteins across organisms [Wang et al., 2007] or (3) methods that narrow down predicted interactions in the organism of interest [Garcia et al., 2010]. Some of these methods use the concepts of "homologs", "orthologs" and "interologs" to define a similarity measure between PPIs from various organisms [Garcia et al., 2010].

There has been little work on combining PPI datasets with the goal of improving prediction performance for multiple organisms. Qi et al. [2010] proposed a semi-supervised multi-task framework to predict PPIs from partially labeled reference sets. The basic idea is to perform multitask learning on a supervised classification task and a semi-supervised auxiliary task via a regularization term. Another line of work in PPI prediction [Xu et al., 2010] uses the Collective Matrix Factorization (CMF) approach proposed by Singh and Gordon [2008]. The CMF method learns models

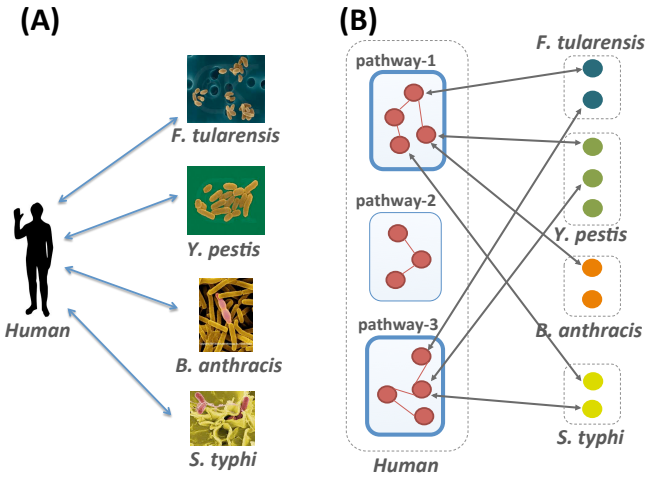


Figure 3.2: (A) Host-pathogen protein-protein interaction (PPI) prediction where the host is human and the pathogens are bacteria. (B) An example depicting the commonality in the bacterial attack of human proteins. Pathway-1 and pathway-3 (highlighted) represent critical processes targeted by all bacterial species.

for multiple networks by simultaneously factorizing several adjacency matrices and sharing parameters amongst the factors. Xu et al. [2010] use these ideas in their transfer learning setting, where the source network is a relatively dense interaction network of proteins and the objective is to infer PPI edges in a relatively sparse target network. To compute similarities between the nodes in the source and target networks, they use protein sequences and the topological structures of the interaction networks.

3.2 Approach

Multi-task learning is a family of machine learning methods that addresses the issue of building models using data from multiple problem domains (i.e 'tasks') by exploiting the similarity between them. The goal is to achieve performance benefits for all the tasks involved. This paradigm of building joint models has been applied successfully in many areas including text-mining, computer vision etc. Since bioinformatics datasets often represent an organism, a natural notion of a 'task' is an 'organism' – for example, Widmer et al. [2010] use a multi-task learning approach for splice-site prediction across many organisms. They use phylogenetic trees to incorporate similarity between organisms (i.e tasks). For a survey of multi-task learning in computational biology, see Xu and Yang [2011].

Our multi-task learning method is based on the task regularization framework, which formulates the multi-task learning problem as an objective function with two terms: an empirical loss term on the training data of all tasks, and a regularization term that encodes the relationships between tasks. Equation (3.1) shows the general form of such an objective, the term R being the regularizer raised to the power p and with a q-norm. Evgeniou and Pontil [2004] introduced a popular regularizer in this

framework that penalizes divergence of a task's parameters from the mean parameter computed over all tasks.

$$L = \sum_{i \in tasks} \text{Loss (task}_i) + \lambda \left\| R \right\|_q^p \tag{3.1}$$

We optimize this function by modifying the regularizer R to encode the biological hypothesis. Our approach differs greatly from prior work because we propose a technique to translate a problem-relevant biological hypothesis into a task-regularization based approach rather than applying existing general formalisms on a dataset. Our tasks try to capture a naturally occurring phenomenon. While our framework is developed in the context of a specific hypothesis, we also illustrate the incorporation of other hypotheses with an example. The key contributions of our work are:

- we present a novel way of combining experimental PPI data coming from several organisms
- we incorporate domain knowledge in designing a prior that causes the learned models to exhibit the requisite common structure across the tasks
- to optimize the resulting non-convex objective function, we implement a concave convex procedure based method

3.3 Datasets and features

For *S. typhi* we used the list of 62 interacting protein pairs reported in Schleker et al. [2012], which were obtained by the authors by manual literature curation. These interactions come from small-scale experiments. The other three PPI interactions datasets were obtained from the PHISTO database. Most of the reported interactions for these three bacterial species come from a single high-throughput experimental study [Dyer et al., 2010]. While *F. tularensis*, *S. typhi* and *Y. pestis* are gram-negative gamma-protobacteria, *B. anthracis* is a gram-positive bacteria. The number of unique proteins in each bacterial species, the sizes of all datasets and the number of all possible host-pathogen protein-pairs are listed in Table 3.1.

Feature set

For each protein-pair, we compute features, some of which use both proteins in the pair, while others are based on either the host protein or the pathogen protein. The following attributes of proteins were obtained from public databases: protein sequences from Uniprot [UniProt Consortium, 2011], gene ontology from GO database [Ashburner et al., 2000], gene expression from GEO [Barrett et al., 2011]. The features derived for each task (which we found to be useful) are listed in Table 3.2. In §2.3 we describe the features in detail.

Our features define a high dimensional and sparse space (the model size is listed in Table 3.1). Since our features are derived by integrating several databases, some of which are not complete, there are many examples and features with missing values. In our current work we eliminate all examples with more than 10% missing features. For the rest we use mean-value based feature imputation. Handling missing data effectively is an important aspect of the PPI prediction problem, however it is not the focus of this work. The remaining examples after elimination and imputation are also shown in Table 3.1.

	Bacillus anthracis	*Francisella tularensis*	*Yersinia pestis*	*Salmonella typhimurium*	*E. coli*
Total no. of bacterial proteins[a]	2321	1086	4600	3592	4003
No. of human-bacterial protein pairs[b]	59.4 M	27.8 M	117.7 M	87.7 M	101 M
No. of known PPI	3073	1383	4059	62	32
No. of PPI with no missing features	655	491	839	62	32
Size of training data with 1:100 class ratio	66155	49591	84739	6262	3232
Model size	694 k	469 k	886 k	349 k	128 k

Table 3.1: Characteristics of the datasets per task. Each task is human-X, where X is the bacterial species (for brevity we only list the bacterial species involved). The number of bacterial proteins (size of proteome), the number of PPI and other statistics are shown.

[a]We only consider the 'reviewed' proteins set from UniprotKB
[b]Note: total no. of human proteins: 25596. 'M': million

TASK	FEATURES USED
B. anthracis	Protein sequence k-mers, Gene Ontology (GO) co-occurance features, Gene expression features, human PPI network features
F. tularensis	Protein sequence k-mers, Gene Ontology (GO) co-occurance features, Gene expression features, human PPI network features
Y. pestis	Protein sequence k-mers, Gene Ontology (GO) co-occurance features, Gene expression features, human PPI network features
S. typhimurium	Protein sequence k-mers, Gene Ontology (GO) co-occurance features, Gene expression features, human PPI network features, Interolog features, Domain interaction features, PFam interaction features
E. coli	Protein sequence k-mers, Gene Ontology (GO) co-occurance features, Gene expression features, human PPI network features

Table 3.2: Features per task. Each task is human-X PPI, where X is the bacterial species (for brevity we only list the bacterial species involved). The details of each feature type are in §2.3.

Negative class examples The interactions listed in the table form the positive class. Since there is no experimental evidence about proteins that do not interact, we construct the "non-interacting" (i.e negative) class using a technique commonly used in PPI prediction literature. Please refer to Section §2.4 for details.

Analysing the known interactions

We analyse the known host-pathogen interactions from our datasets. This analysis also motivates our choice of a multi-task approach that uses a pathway-based similarity across tasks. The known PPIs are compared across datasets in two ways: (a) pathway enrichment and (b) presence of interologs.

(a) The human proteins involved in each interaction dataset are used to obtain the human pathways that are enriched. We use Fisher's test (based on the hypergeometric distribution) to compute the p-value of each pathway. We plot these p-values for each pathway, and for each dataset in the form of a heat-map shown in Figure 3.3. The heatmap shows how there are several commonly enriched pathways across the datasets (the black vertical lines spanning all 4 rows). It also shows the difference in the enrichment for the *S. typhi* dataset which comes from small-scale PPI experiments.

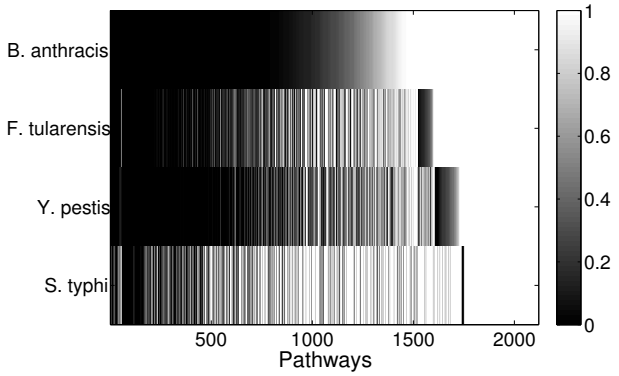


Figure 3.3: Heatmap showing pathways enriched in each bacterial-human PPI interactions dataset. The horizontal axis represents the pathways (about 2100 of them) and the vertical axis represents the 4 datasets. Each entry in the heat-map represents the p-value of a pathway w.r.t one dataset. Darker values represent more enrichment. The black columns that span across all 4 rows show the commonly enriched pathways.

(b) We analyse the similarity between the PPIs from various datasets. A natural way to determine similarity is to check if proteins known to interact in one dataset have homologous proteins that are also interacting in another dataset. Such pairs of proteins, also called "interologs" are defined as a quadruple of proteins A, B, A', B', where $A \leftrightarrow B$ (interaction) and $A' \leftrightarrow B'$. Further, A, A' are homologs and B, B' are also homologs. The number of such interologs existing between the four datasets is shown in Table 3.3. To compute homologs of a protein, we used BLASTP sequence alignment with an e-value cut off of 0.1. As evident from Table 3.3, there are very few interologs across the bacterial PPIs. None of the high-throughput datasets have an interolog in the small-scale *S. typhi* dataset. This seems to indicate that interolog-based approaches to compute task-similarity are not relevant here. The phenomenon

Human-bacteria PPI datasets compared	H-B vs. H-F	H-B vs. H-Y	H-B vs. H-S	H-F vs. H-Y	H-F vs. H-S	H-Y vs. H-S
Number of interologs	2	3	0	3	0	0

Table 3.3: Conserved interactions in the form of interologs across the various host-bacterial datasets. H-X: stands for human-pathogen where the pathogen 'X' can be B, F, Y and S referring to *B. anthracis*, *F. tularensis*, *Y. pestis* and *S. typhi.* respectively. The non-zero entry '2' for 'H-B vs H-F' means there are two PPIs in the H-B dataset that have interologs in the H-F dataset.

governing the similarity of these host-pathogen interactions is probably at a much higher level, rather than at the level of individual proteins. We explore one such possibility – the 'commonality hypothesis'.

3.4 The objective function and optimization

In this section we describe how we incorporate the commonality hypothesis into our multi-task classification framework formulating it as an optimization problem.

We consider each human-bacteria PPI prediction problem as one task. The prediction problem is posed as a binary classification task, with each instance $\mathbf{x}^i$ being a pair of proteins $<b, h>$, where one protein is the bacterial protein 'b' (e.g. *Y. pestis*) and the other 'h' is the host protein (i.e human). The class-label $y^i \in \{+1, -1\}$ represents interacting and non-interacting proteins respectively. Features are defined for every protein-pair using various properties of the individual proteins and combining them all into a single feature vector. The positive class in our training data comprises the known human-bacterial PPI which are obtained from databases like PHISTO [Tekir et al., 2012]. The construction of the negative-class data is explained in §2.4.

Our objective is to minimize the empirical error on the training data while favoring models that are biased toward the commonality hypothesis. To achieve this, we use a bias term in the form of a regularizer in our objective function. For brevity and without loss of generality, we will henceforth refer to each human-bacteria PPI prediction problem as a 'task' [1].

Our method first combines all tasks in a pairwise manner, and finally aggregates the output from the pairwise models. Let $\mathcal{T} = \{\mathcal{T}_t\}_{t=1}^m$ be the set of tasks to be combined, where m is the number of tasks. Consider two tasks $\mathcal{T}_s$ and $\mathcal{T}_t$. Let the training data for the task $\mathcal{T}_s$ be $X_s = \{\mathbf{x}_s^i \mid i = 1 \dots n_s\}$ where each example $\mathbf{x}_s^i \in \mathbb{R}^{d_s}$. Similarly, the training data for $\mathcal{T}_t$ is $X_t = \{\mathbf{x}_t^i \mid i = 1 \dots n_t\}$ where $\mathbf{x}_t^i \in \mathbb{R}^{d_t}$. n_s and n_t are the number of training examples and d_s and d_t denote the number of features in the two tasks. Let $\mathbf{w}_s \in \mathbb{R}^{d_s}$, $\mathbf{w}_t \in \mathbb{R}^{d_t}$ represent the parameter vectors i.e the models for the two tasks. We now describe how we combine these two tasks. §3.4 will show how such pairwise models are aggregated.

[1] We will also refer to a task by the name of the bacterial species only, since the host species i.e human is common across all tasks.

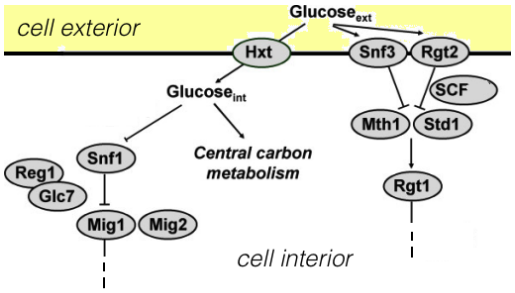


Figure 3.4: Part of the "glucose transport pathway" in human. Grey nodes represent the human proteins (genes) involved. Edges represent causality in the process. This pathway involves the transport of glucose from outside the cell to various components inside the cell.

The pathway-based objective : Biologists often represent the set of human proteins involved in a particular biological process by a graph called a "biological pathway". One such example, the "glucose transport pathway" in human is shown in Figure 3.4. In order to use this pathway construct, we revise our hypothesis to: proteins from different bacterial species are likely to interact with human proteins from the same biological pathway. Figure 3.2(B) illustrates an example where this hypothesis holds. The pathway information for each human protein can be obtained from pathway databases like Reactome [Matthews et al., 2008] and PID [Schaefer et al., 2009]. While pathways are generally represented as graphs, for our current work we do not use the edges. We treat a pathway as a *set* of proteins – a human protein h can be a member of several pathways depending on the biological processes it is involved in.

Let N be the total number of pathways in human. For a protein-pair $i = <b, h>$, let $\mathbf{p}^i \in \{0,1\}^N$ be the binary 'pathway vector' indicating the pathway membership of h.

The commonality hypothesis suggests that the pathway memberships of human proteins from interactions should be similar across tasks. We define a pathway-summary function S, which aggregates all pathway vectors for a given task $\mathcal{T}_s$. Since our hypothesis is about *interactions*, we only consider pathway vectors of *positive* examples. Let X_s^+, X_t^+ represent the set of positive examples from tasks $\mathcal{T}_s$ and $\mathcal{T}_t$; let n_s^+, n_t^+ be their sizes. In Figure 3.5 we depict the aggregation done by S. Mathematically, we have

$$S(\mathcal{T}_s) = \frac{1}{n_s^+} \sum_{i \in X_s^+} \mathbf{p}_s^i \; I_{pos}(\mathbf{w}_s^\intercal \mathbf{x}_s^i) \tag{3.2}$$

where $\mathbf{p}_s^i$ is the pathway vector for example i, and $I_{pos}(z) = I(z > 0)$. S sums up the pathway vectors of examples predicted to be positive. We normalize using n_s^+ to compensate for the different dataset sizes across tasks.

Let $P_s = \{\mathbf{p}_s^i \mid i = 1 \ldots n_s^+\}$ be a matrix containing all pathway vectors for positive examples from task $\mathcal{T}_s$. Analogously, $P_t \in \{0,1\}^{N \times n_t^+}$ is a matrix for the positive examples from task $\mathcal{T}_t$. Matrices P_s and P_t are constant matrices and are known apriori. Let $S(\mathcal{T}_s)$ and $S(\mathcal{T}_t)$ be the pathway summaries of the tasks. We want to penalize the dissimilarity between these summaries. Our objective function thus has the following

general form:

$$\begin{aligned} L(\mathbf{w}_s, \mathbf{w}_t) = \quad & l(\mathbf{w}_s) + l(\mathbf{w}_t) + \lambda \left\| R \right\|_2^2 + \sigma \left(\|\mathbf{w}_s\|_2^2 + \|\mathbf{w}_t\|_2^2 \right) \\ \text{where} \quad & R = S(\mathcal{T}_s) - S(\mathcal{T}_t). \end{aligned} \tag{3.3}$$

Here $l(\mathbf{w}_s)$ and $l(\mathbf{w}_t)$ can be any convex loss functions computed over the two tasks. We use logistic loss in our work based on prior experience with PPI datasets. The last two ℓ_2 norms over the parameter vectors $\mathbf{w}_s$ and $\mathbf{w}_t$ control over-fitting. The parameters λ and σ take positive values.

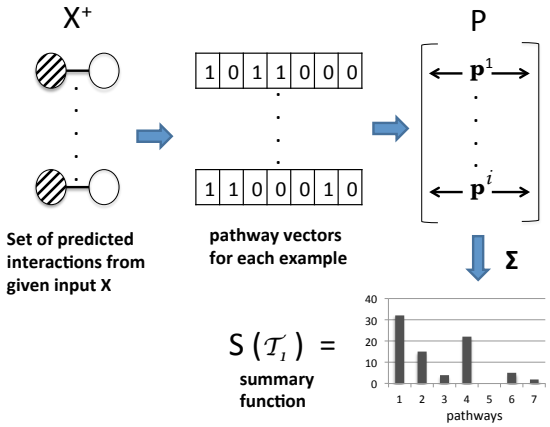


Figure 3.5: A schematic illustrating the pathway summarizing function S for a task $\mathcal{T}_1$. On the left are the examples from the input predicted to be positive, indicated by X^+. The matrix P has the pathway vectors for each example in X^+. The summary function aggregates the pathway vectors to get the distribution.

The indicator function I_{pos} is non-differentiable. So we approximate I_{pos} with the exponential function which is a convex upper bound of the indicator function and will make optimization easier. Let $\phi(z) = e^{z/C}$, where C is a positive constant. This function, for various values of C has been plot in Figure 3.6. Small positive values of $z = \mathbf{w}^\intercal \mathbf{x}^i$ indicate positive-class predictions that are closer to the decision boundary of the classifier. Examples predicted to be positive with a high confidence have a large z. With varying values of C, the function ϕ gives varying importance to predictions based on their classifier confidence 'z'. Negative values of z which correspond to examples predicted to be negative, are given close to zero importance by ϕ. The choice of an appropriate C is important so as to ensure the proper behaviour of the summary function S. A steeply increasing curve (C=1) is undesirable as it will assign too much weight to the summary from those examples. We chose a moderate value of C=30 for our experiments.

Replacing I_{pos} by ϕ in equation 3.2, our summary function S becomes:

$$S(\mathcal{T}_s) = \frac{1}{n_s^+} \sum_{i \in X_s^+} \mathbf{p}_s^i \; \phi(\mathbf{w}_s^\intercal \mathbf{x}_s^i) \tag{3.4}$$

Putting everything together in equation 3.3, our objective with the logistic loss

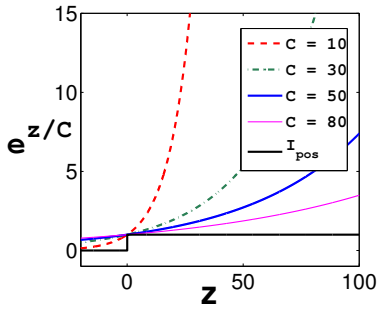


Figure 3.6: The exponential function $e^{z/C}$ for different values of C.

terms, the pathway summary function and the ℓ_2 regularizer terms has the form:

$$L(\mathbf{w}_s, \mathbf{w}_t) = \sum_{i=1}^{n_s} \log\left(1 + e^{-\mathbf{w}_s^\intercal \mathbf{x}_s^i y_s^i}\right) + \sum_{j=1}^{n_t} \log\left(1 + e^{-\mathbf{w}_t^\intercal \mathbf{x}_t^j y_t^j}\right) + \lambda \left\| \frac{1}{n_s^+} \sum_{i \in X_s^+} \mathbf{p}_s^i \, \phi(\mathbf{w}_s^\intercal \mathbf{x}_s^i) - \frac{1}{n_t^+} \sum_{j \in X_t^+} \mathbf{p}_t^j \, \phi(\mathbf{w}_t^\intercal \mathbf{x}_t^j) \right\|_2^2 + R_{\ell_2}(\mathbf{w}_s, \mathbf{w}_t) \tag{3.5}$$

where $R_{\ell_2}(\mathbf{w}_s, \mathbf{w}_t) = \sigma\left(\|\mathbf{w}_s\|_2^2 + \|\mathbf{w}_t\|_2^2\right)$

Solving the optimization problem

The objective in equation (3.5) is non-convex, and with some algebraic simplifications we can reduce it to a difference of convex functions (DC). To optimize this function, we implement the CCCP (Concave Convex procedure) algorithm which was originally introduced by Yuille and Rangarajan [2003]. We tried to optimize it directly using L-BFGS, but found that the objective does not decrease consistently.

Below, we show the reduction to difference of convex functions. The first two log-loss terms (we abbreviate them henceforth as $\ell(\mathbf{w}_s, \mathbf{w}_t)$) and the last R_{ℓ_2} term are all convex and do not pose any problem with optimization.

Proposition 1. *The objective (3.5) is a difference of convex functions.*

$$L(\mathbf{w}_s, \mathbf{w}_t) = F(\mathbf{w}_s, \mathbf{w}_t) - G(\mathbf{w}_s, \mathbf{w}_t) \tag{3.6}$$

Proof. Expanding the pathway vectors $\mathbf{p}_s^i$ and $\mathbf{p}_t^j$ and rewriting equation (3.5) we get:

$$\begin{aligned} L &= \ell(\mathbf{w}_s, \mathbf{w}_t) + R_{\ell_2}(\mathbf{w}_s, \mathbf{w}_t) + \lambda \sum_{k=1}^{N} \left(\frac{1}{n_s^+} \sum_{i \in X_s^+} p_s^{ki} \, \phi(\mathbf{w}_s^\intercal \mathbf{x}_s^i) - \frac{1}{n_t^+} \sum_{j \in X_t^+} p_t^{kj} \, \phi(\mathbf{w}_t^\intercal \mathbf{x}_t^j) \right)^2 \\ L &= \ell(\mathbf{w}_s, \mathbf{w}_t) + R_{\ell_2}(\mathbf{w}_s, \mathbf{w}_t) + \lambda \sum_{k=1}^{N} (f_k - g_k)^2, \text{ where} \\ f_k &= \frac{1}{n_s^+} \sum_{i \in X_s^+} p_s^{ki} \, \phi(\mathbf{w}_s^\intercal \mathbf{x}_s^i) \text{ and } g_k = \frac{1}{n_t^+} \sum_{j \in X_t^+} p_t^{kj} \, \phi(\mathbf{w}_t^\intercal \mathbf{x}_t^j). \end{aligned} \tag{3.7}$$

Note that f_k and g_k are non-negative convex functions. This follows because: $\phi(z) = e^{z/C}$ is a positive convex function and the matrices P_s and P_t are non-negative by

construction. f_k and g_k are both thus positive linear combinations of convex functions and hence convex. We now decompose the squared term in equation (3.7) as follows.

$$\sum_{k=1}^{N}(f_k - g_k)^2 = \sum_{k=1}^{N} 2(f_k^2 + g_k^2) - \sum_{k=1}^{N}(f_k + g_k)^2 \tag{3.8}$$

We further observe that f_k^2 is convex. To derive this, we use the following proposition: a composition of a monotonically increasing convex function and a convex function is still convex. The square function $h(z) = z^2$ is a monotonically increasing function for $z \geq 0$, thus the composition with f_k (i.e $h(f_k)$) is also convex by the positivity of f_k. Analogously, g_k^2 is also convex. Further, $(f_k + g_k)^2$ is also convex by the same argument. Substituting (3.8) back into equation (3.7) we get our result.

$$\begin{aligned} L &= \left[\ell(\mathbf{w}_s, \mathbf{w}_t) + R_{\ell_2}(\mathbf{w}_s, \mathbf{w}_t) + \ \lambda \sum_{k=1}^{N} 2(f_k^2 + g_k^2)\right] - \left[\ \lambda \sum_{k=1}^{N}(f_k + g_k)^2\right] \\ L &= F(\mathbf{w}_s, \mathbf{w}_t) - G(\mathbf{w}_s, \mathbf{w}_t) \end{aligned} \tag{3.9}$$

□

To optimize this function, we use a CCCP (Concave Convex procedure) algorithm [Yuille and Rangarajan, 2003], similar to the approach from Yu and Joachims [2009] used for learning structural SVMs. The idea is to compute a local upper bound on the concave function $(-G)$ and instead of optimizing L from equation (4) directly, use an approximation based on the upper bound of $-G$. Equation (7) shows this function L_{approx}. Let $\mathbf{w}$ represent the concatenation of the two parameter vectors $\mathbf{w}_s$ and $\mathbf{w}_t$. Let $\mathbf{w^k}$ be the k-th iterate. We have from Taylor's first order approximation that $-G(\mathbf{w}) \leq -G(\mathbf{w^k}) + (\mathbf{w} - \mathbf{w^k})^\intercal \nabla G$ for all $\mathbf{w}$. This allows us to obtain the following approximation which we get by substituting the above bound in place of $-G$ in equation (3.6):

$$\begin{aligned} \min_{\mathbf{w}} L_{approx}(\mathbf{w}) &= \min_{\mathbf{w}} \left[F(\mathbf{w}) - G(\mathbf{w^k}) + (\mathbf{w} - \mathbf{w^k})^\intercal \nabla G\right] \\ &= \min_{\mathbf{w}} \left[F(\mathbf{w}) + \mathbf{w}^\intercal \nabla G\right], \end{aligned} \tag{3.10}$$

since $\mathbf{w^k}$ is a constant. The optimization problem in equation (3.10) is now convex and can be solved using conventional techniques like L-BFGS, conjugate gradient etc. The outline of our CCCP based procedure is shown in Listing 1.

Yuille and Rangarajan [2003] show that such a CCCP based algorithm is guaranteed to decrease the objective function at every iteration and to converge to a local minimum or saddle point. We observe a similar behaviour in our experiments. Computationally, this algorithm is efficient since the regularizer works on a subset of the data - only the positive examples which are a small fraction of the complete training data.

Stopping criteria: The convergence criterion for algorithm 1 is: $\delta < \tau$, where τ is a threshold. We used $\tau = 1$ in our experiments. Smaller values required a very long time to convergence. The inner optimization (line # 5) which uses L-BFGS had a convergence threshold of 0.0001. This step took more iterations initially and fewer iterations getting closer to convergence.

Algorithm 1: CCCP procedure

1: Initialize $\mathbf{w} = \mathbf{w^0}$
2: **repeat**
3: Compute ∇G using $\mathbf{w^k}$
4: Compute current value L_{approx}
5: Solve $\mathbf{w^{k+1}} = \underset{\mathbf{w}}{\text{argmin}} \left[F(\mathbf{w}) + \mathbf{w}^\intercal \nabla G\right]$
6: Set $k = k + 1$
7: Compute new value $L^{'}_{approx}$
8: $\delta = L_{approx} - L^{'}_{approx}$
9: **until** $\delta < \tau$

Combining pairwise models

In the previous sections, we described how we combine two tasks. In particular, equation (3.5) involves pairwise learning which results in two models $\mathbf{w}_s$ and $\mathbf{w}_t$. Since our current framework can combine only two tasks at a time, for m tasks we perform $\binom{m}{2}$ pairwise learning experiments and then combine their outputs. Each task will thus have $m-1$ models as a result of pairing up with each of the other tasks.

Let the set of models for task $\mathcal{T}_s$ be $\mathcal{M}_s = \{\mathbf{w}_{s_1}, \mathbf{w}_{s_2} \ldots \mathbf{w}_{s_{m-1}}\}$. We treat $\mathcal{M}_s$ as an ensemble of models for this task and aggregate the output labels from all models to get the final labels on the test data. Let the output labels from each model for a given test instance $\mathbf{x}$ be $O_x = \{o_1, o_2 \ldots o_{m-1}\}$. Then the final output label y is computed by taking a vote and checking if it crosses a threshold:

$$y = \begin{cases} 1 & \text{if } \left(\sum_{o_j} I(o_j = 1)\right) \geq v \\ -1 & \text{otherwise} \end{cases} \tag{3.11}$$

where v is a vote-threshold that should be crossed in order for the label to be positive. In our experiments, we found that the predictions for $\mathcal{T}_s$ from all models in $\mathcal{M}_s$ overlapped greatly. Hence we used $v = 1$ which implies that $\mathbf{x}$ is an interaction if any one of our 4 tasks labels it as such.

3.5 Experiments

We use 10 fold cross-validation (CV) to evaluate the Precision, Recall and F-score of all algorithms (refer to §2.5). The baselines that we compare against are briefly described below.

Single Task Learning (STL): We train models independently on each task using two standard classifiers: Support Vector Machines and Logistic regression with ℓ_1 and ℓ_2 regularization. We used LibLinear [Fan et al., 2008] for these experiments and found that logistic regression with ℓ_1 regularization performs the best across all tasks. For conciseness, we report only the best model's performance.

STL with pathway features (STL Path.): This baseline incorporates the pathway information from the pathway vectors $\mathbf{p}^i$ as features. For each example i, the feature vector is appended by the pathway vector $\mathbf{p}^i$. While our method uses the pathway

vectors only for the positive class examples (via the matrices P_s and P_t), this baseline uses the pathway information for all examples via features. The performance of this baseline will indicate whether using raw pathway information without incorporating any biologically relevant coupling does well. We learn independent models for each task as before, and find that logistic regression with ℓ_1 regularization does the best (only these results are reported).

Coupled models: This baseline was implemented so as to couple the regularizer parameter across two tasks thereby keeping the basic framework similar to that in our technique. To achieve this we optimize the function in equation (3.12) and use the L-BFGS implementation from Mallet. Note that the previous baseline has separate regularization parameters for each task.

$$L = \sum_{i=1}^{n_s} \log\left(1 + e^{-\mathbf{w}_s^\intercal \mathbf{x}_s^i y_s^i}\right) + \sum_{j=1}^{n_t} \log\left(1 + e^{-\mathbf{w}_t^\intercal \mathbf{x}_t^j y_t^j}\right) + \sigma(\|\mathbf{w}_s\|_2^2 + \|\mathbf{w}_t\|_2^2) \quad (3.12)$$

Mean MTL: This is a logistic regression-based implementation of the multi-task SVM model proposed by Evgeniou and Pontil [2004]. The important feature of this work is the use of a regularizer that penalizes the difference between a model and the "mean" model formed by averaging over models from all m tasks. In the original paper, the loss functions $l(\mathbf{w}_i)$ were all hinge-loss. Since we find that logistic-regression does better on our datasets, we replaced the original hinge loss function by logistic loss. The objective we use is shown in equation (3.13).

$$L = \sum_{i=1}^{m} l(\mathbf{w}_i) + \lambda \sum_{i=1}^{m} \left\| \mathbf{w}_i - \frac{1}{m} \sum_j \mathbf{w}_j \right\|_2^2 + \sigma \sum_{i=1}^{m} \|\mathbf{w}_i\|_2^2 \quad (3.13)$$

L1/L2 regularization: Another way to combine tasks using their common features is by minimizing the ℓ_1/ℓ_2 norm over the feature matrix $\mathbf{W} = [\mathbf{w}_1 \ldots \mathbf{w}_m]$. The sparsity constraints due to the ℓ_1 norm decouples the rows of $\mathbf{W}$ allowing the selection of a subset of the features to regress upon. The ℓ_2 norm couples together the columns (i.e the $\mathbf{w}_i$s), as a consequence of which coefficients for a particular feature jointly remain non-zero across all m tasks.

$$L = \sum_{i=1}^{m} l(\mathbf{w}_i) + \lambda \|\mathbf{W}\|_{\ell_1/\ell_2} \quad (3.14)$$

where $\mathbf{W} = (w_{ij})$ with $1 \leq i \leq d,\ 1 \leq j \leq m$. The rows represent the d common features across all tasks. The columns representing the m task-specific parameter vectors $\mathbf{w}_i$. The term $\|\mathbf{W}\|_{\ell_1/\ell_2} = \sum_{i=1}^{d} \|\vec{w}_{i.}\|_2$ is the block ℓ_1/ℓ_2 norm. The regularization parameter λ controls the enforcement of the ℓ_1/ℓ_2 group-structure. The parameter range that gave the best performance on the held-out data was $[10^{-4}, 10^{-7}]$. This was used to compute the performance on the test data.

Multi-task pathway based learning (MTPL): This refers to our technique, which minimizes the sum of logistic loss over the two tasks with an ℓ_2 regularization penalizing the difference between the pathway summaries. We train two tasks at a time

METHOD	*B. anthracis*	*F. tularensis*	*Y. pestis*	*S. typhi*
STL	27.8 ± 4.0	25.7 ± 5.4	28.8 ± 4.0	72.5 ± 11.4
STL PATH.	26.5 ± 4.7	26.1 ± 6.9	26.7 ± 4.3	69.1 ± 12.7
COUPLED	27.0 ± 3.9	25.5 ± 5.0	27.9 ± 3.4	69.8 ± 12.4
MEAN MTL	25.2 ± 4.9	26.7 ± 4.0	27.5 ± 6.3	69.4 ± 12.1
L1/L2 REG.	31.2 ± 3.7	$\mathbf{30.6} \pm 6.7$	$\mathbf{32.0} \pm 3.9$	73.1 ± 16.9
MTPL	$\mathbf{32.0} \pm 3.9$	30.1 ± 5.8	$\mathbf{32.1} \pm 2.5$	$\mathbf{75.8} \pm 12.1$

Table 3.4: Averaged 10 fold cross-validation performance for all methods for a positive:negative class ratio of 1:100. Accuracy is reported as the F1 measure computed on the positive class. The standard deviation over the 10 folds is also reported.

and compute the performance for each task. Since we have four tasks, there are six such pairwise learning experiments in all. While evaluating performance during 10 fold CV, we obtain the F1 on one fold of a task $\mathcal{T}_t$ by averaging the F1 across all pairwise learning experiments that involve $\mathcal{T}_t$ (see Section 3.4 for details). The final CV performance reported in our results is an average over 10 folds.

Hyper-parameter tuning

We followed an identical procedure for all algorithms. For the 10 fold CV experiments we train on eight folds, use one fold as held-out and another as test. The optimal parameters (i.e the best model) was obtained by parameter tuning on the held-out fold. The test fold was used to evaluate this best model - these results are reported in Section 3.6. The range of values we tried during the tuning of the regularization parameter (λ) were: 150 to 10^{-4}. For σ - the parameter controlling overfitting in MTPL, we used a fixed value of $\sigma = 1$. For MeanMTL we tune both λ and σ. To handle the high class imbalance in our data, we used a weight-parameter W_{pos} to increase the weight of the positive examples in the logistic loss terms of our function. We tried three values and found $W_{pos} = 100$ performed the best on training data.

3.6 Results and Discussion

Overall performance

Table 3.4 reports for each bacterial species, the average F1 along with the standard deviation for the 10 fold cross validation (CV) experiments. The performance of all baselines is very similar, and our method outperforms the best of the baselines by a margin of 4 points for *B. anthracis*, 3.4 points for *F. tularensis* and 3.2 points for *Y. pestis* and 3.3 for *S. typhi*. The overall performance of all methods on this dataset is twice as good as that on the others. We believe that the difference in the nature of the datasets might explain the above observations. While the *S. typhi* dataset comprises small-scale interaction studies, the other datasets come from high-throughput experiments. Due to its smaller size it has less variance making it an easier task. This dataset is also likely to be a biased sample of interactions, as it comes from focused studies targeting select proteins.

The coupled learner (Coupled) performs slightly worse than STL. This is explained by the fact that STL has more flexibility in setting the regularization parameter for each task separately which is not the case in Coupled. It is interesting to note that the independent models that use the pathway matrices P_s and P_t as features (i.e STL Path) show a slightly worse performance than STL that does not use them. This seems to suggest that the cross-task pathway similarity structure that we enforce using our regularizer has more information than simply the pathway membership of proteins used as features.

Precision Recall curves from 10 fold CV results

We plot the recall vs the precision obtained by our method, MTPL on the 4 tasks in Figure 3.7. We used the results from the 10 fold CV experiments. The classifier score for each test instance was aggregated from the various pairwise models a manner similar to what is explained in §3.4. Let the classifier scores (i.e $\mathbf{w}^\intercal\mathbf{x}$) from each model for a given test instance $\mathbf{x}$ be $\{s_1, s_2 \ldots s_{m-1}\}$. The aggregated multi-task classifier score of $\mathbf{x}$ is given by:

$$s(\mathbf{x}) = \begin{cases} \max\limits_i \; s_i, & \text{if } \left(\sum_i I(s_i \geq 0)\right) \geq 1 \\ \min\limits_i \; s_i, & \text{otherwise} \end{cases} \tag{3.15}$$

The classifier threshold was then varied and the precision (P), recall (R) were computed for each threshold. The final curve was obtained by aggregating the P-R curves from each of the ten folds.

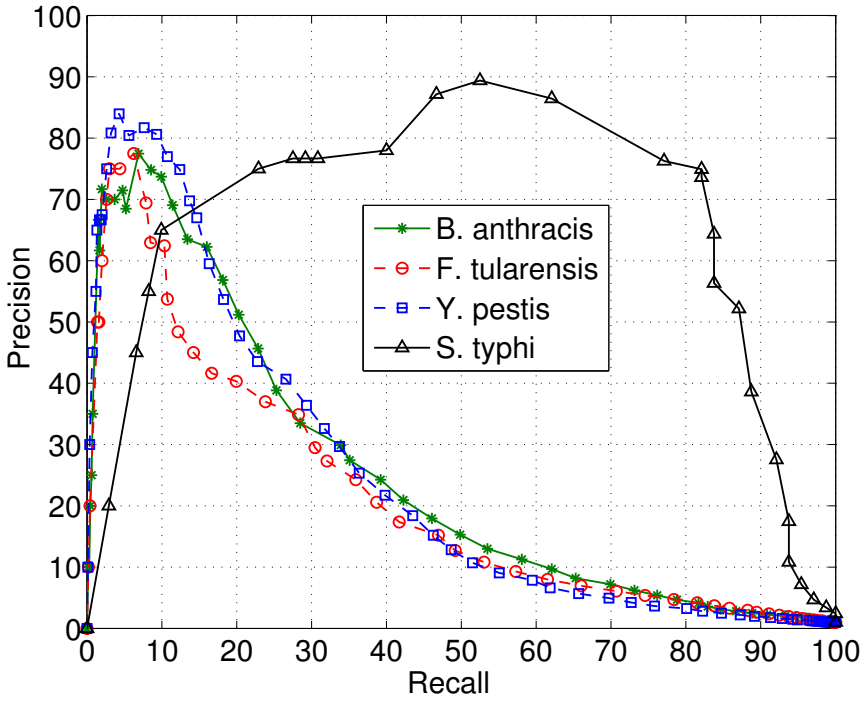


Figure 3.7: Precision-Recall curves for MTPL for all tasks

PAIRWISE MODEL PERFORMANCE OF MTPL

PAIRWISE TASKS	F1	
TASK-1, TASK-2	TASK-1	TASK-2
B. anthracis, *F. tularensis*	31.4	30.1
B. anthracis, *S. typhi*	**32.0**	**76.3**
B. anthracis, *Y. pestis*	31.6	32.0
F. tularensis, *S. typhi*	**30.3**	73.0
F. tularensis, *Y. pestis*	30.0	32.1
S. typhi, *Y. pestis*	74.2	**32.3**

Table 3.5: F1 computed during 10 fold cross-validation of various pairwise models from MTPL. Positive : negative class ratio was 1:100. The best F1 achieved for each task (i.e for each bacterial species) is shown in bold. For example, *B. anthracis* has the best performance of 32 when it is coupled with *S. typhi*.

Pairwise performance of tasks in MTPL

The previous section gave a summary of the aggregated performance of MTPL for every task. Here we present the performance of every pairwise learning experiment of MTPL in Table 5. This gives an idea of how various tasks benefit from being paired up with other tasks.

For each task, we check the task-pairing that gave the best performance (best F1 for each task is shown in bold). For instance, the best F1 of 32.3 for *Y. pestis* was obtained in the pairwise model learned with *S. typhi*. It is evident that coupling a model with one additional task seems to improve the performance over the baseline.

Feature importance across tasks

To get an understanding of inter-task model similarity, we compared the parameter vectors '$\mathbf{w}$' of all tasks with each other (each $\mathbf{w}$ was learned on the entire training data). Since the number of features is very large, we computed the cosine similarity between them. Note that we only use features which are common across tasks for this comparison. Gene expression features for instance were not used as they vary with regards to the number of expression time-points, the experiment protocol etc.

We found that the feature weights vary greatly across models – the cosine similarity ranges between 0.1 to 0.13. We also analyzed which features had the highest absolute weight. We found that the node-degree feature (computed using the human PPI graph) has a very high positive weight across all tasks. Gene expression features have large negative weights across all tasks. In general, the GO and protein sequence based n-gram features have very different weights across tasks.

This seems to imply that having similar parameter values across models is not particularly important for this multi-task problem. This explains why one of our baselines: the Mean-MTL method which penalizes differences between parameter vectors, does not perform very well. Instead, regularization using the pathway sum-

maries seems key in giving a better performance.

Sparsity of weights: We use ℓ_2 regularization in our optimization function, which does not produce very sparse weight vectors. We observe that about 50% of the features have 0 weight in all tasks. About 75 - 80% of the features have small weights in the range of (0.001, -0.001).

Analysis of predictions

The F1 measure gave us a quantitative idea of the performance of each method on training data. In this section, we present a qualitative analysis of the new interactions that our models predict. We first construct, for each task '$\mathcal{T}_t$', a random set R_t of protein pairs that is disjoint from the training dataset. We train the pairwise models on the training data and obtain predictions on R_t. The method described in Section 3.4 is used to aggregate predictions from all pairwise models. The subset of R_t labeled as 'positive' is used for the analysis described below.

Enriched human pathways : We perform enrichment analysis on the human pathways from the positive predictions of MTPL. We use Fisher's exact test with the hyper-geometric distribution. We intersect the top enriched pathways that satisfy p-value $\leq$ 1e-07 from each task to get the commonly enriched pathways. The sizes of the various intersections are shown in Figure 3.8. 17 pathways are commonly enriched across all four tasks. 104 pathways are enriched across the three high-throughput datasets - which is a significant fraction of the total number of pathways considered. This result indicates that the bias produced by our regularizer does produce predictions satisfying the commonality hypothesis.

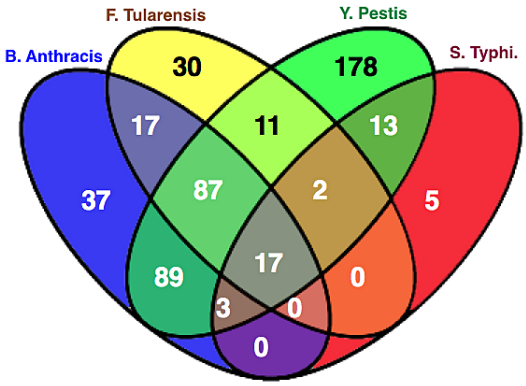


Figure 3.8: The intersection of enriched human pathways from predicted interactions. The total number of enriched pathways for each bacterial species are: *B. anthracis*: 250, *F. tularensis*: 164, *Y. pestis*: 400 and *S. typhi.*: 40. The size of the intersection between all tasks' enriched pathways is: 17. The size of this intersection for the high-throughput datasets (excluding *S. typhi*) is much larger: 104.

Comparing enriched pathways: gold-standard vs. predicted

We also analyze the overlap between the pathways enriched in the gold-standard positives and those enriched in the predictions. See Figure 3.9. For both enrichment computation, the human genes from the interactions are considered. We used Fisher's

exact test and a p-value cut-off of $10^{-}7$. The filled circles on the left of each intersection represent the enriched pathways in the predictions. The empty circles on the right show the enriched pathways in the training data. We can see that there are several new pathways enriched in the predictions as compared to those enriched in the gold-standard data.

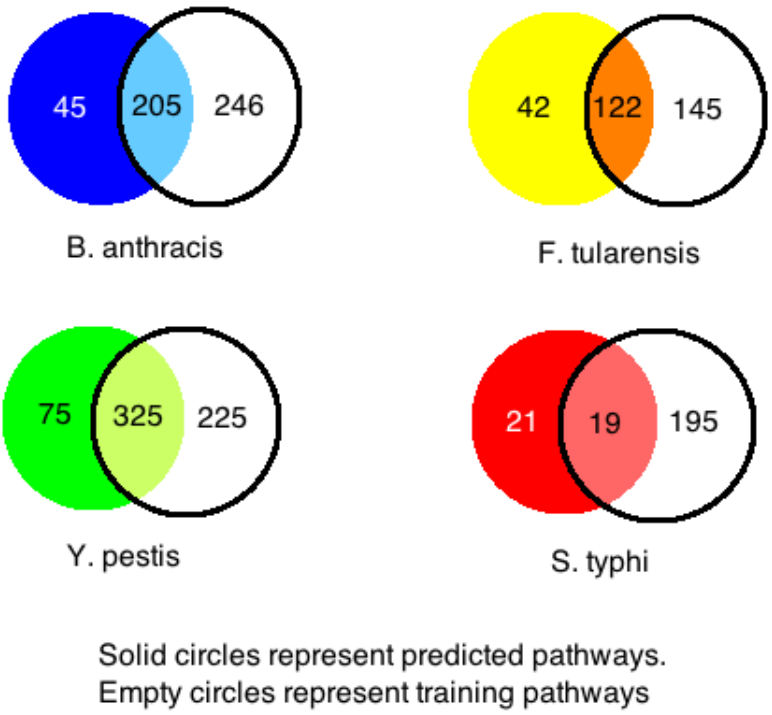


Figure 3.9: Enrichment intersection between training PPIs and predicted PPIs. Cut-off used for enrichment: $10^{-}7$.

Table 3.6 shows the 17 pathways commonly enriched from predictions across all bacterial datasets. The "Integrin alpha IIb beta3 (αIIbβ3) signaling" pathway is enriched only in *B. anthracis* and *Y. pestis* in the training data. However, in the predictions it is enriched in all four bacterial datasets. Integrin-αIIbβ3 is a trans-membrane receptor expressed in mast cells and plays an important role in innate immune responses against pathogens.

Incorporating other biological hypotheses

The regularizer in equation (3.5) uses the pathway information matrix to enforce pathway level similarity. The matrix can be used to represent any other common structure. For example, consider the hypothesis - *all pathogens target hub proteins in the host*, which implies that bacterial proteins are often found to interact with host proteins that have a high node degree in the PPI network of the host. We tried two variants to incorporate this hypothesis - (a) we identify "hubs" in the human PPI graph and use the binary vectors $\mathbf{p}^i$ as an indicator of the "hub" protein targeted by the bacterial protein (b) instead of a discrete 'hub' / 'not hub' indicator we use $\mathbf{p}^i$ to represent the node-degree (each component of $\mathbf{p}^i$ represents one node-degree bin say [10 - 20]). We

Adaptive Immune System
Developmental Biology
E-cadherin signaling events
E-cadherin signaling in the nascent adherens junction
Glypican pathway
Immune System
Integrin alphaIIb beta3 signaling
Integrin cell surface interactions
L1CAM interactions
N-cadherin signaling events
Platelet activation, signaling and aggregation
Platelet Aggregation (Plug Formation)
Posttranslational regulation of adherens junction stability and dissassembly
Signalling by NGF
Signal Transduction
Stabilization and expansion of the E-cadherin adherens junction
TNF alpha/NF-kB

Table 3.6: The 17 commonly enriched pathways in the predicted interactions from MTPL.

found that using (a) gives us an improvement of upto 2.5 F points over the baseline methods.

3.7 Unified multi-task pathway objective (U-MTPL)

The method presented in the previous sections integrates multiple tasks in a pairwise manner, and does not scale well while integrating several PPI datasets as we have to solve one optimization problem per task pair. For m tasks this means $O(m^2)$ pairwise regularization problems to solve. The most straightforward way of extending equation 3.3 to learning m tasks simultaneously involves loss terms for each of the tasks and $m(m-1)$ regularizer terms. Let $\mathbf{w}_t$ represent the parameters of task $\mathcal{T}_t$, and $\mathcal{T}$ is the set of all tasks. The unified objective can then be expressed as:

$$L(\mathbf{w}_1, \mathbf{w}_2 \dots \mathbf{w}_m) = \sum_{t=1}^{m} l(\mathbf{w}_t) + \sum_{\mathcal{T}_s, \mathcal{T}_t \in \mathcal{T}} \lambda_{st} \left\| S(\mathcal{T}_s) - S(\mathcal{T}_t) \right\|_2^2 + \sigma \sum_{t=1}^{m} \|\mathbf{w}_t\|_2^2 \quad (3.16)$$

The summary function $S(\mathcal{T}_t)$ is as defined in equation 3.4. Here, λ_{st} is the hyper-parameter controlling how tightly the two corresponding tasks should be coupled together. Task pairs that are very similar to each other should have a higher value for λ_{st}.

This formulation has the advantage of learning all parameters simultaneously and does not require the final model-aggregation step that the pairwise model needs (Section 3.4). The drawback is the growth of the hyper-parameter space. There are $O(m^2)$ hyper-parameters: λ_{st}, and searching for the best combination of these task-pair similarities requires a search over a grid in R^{m^2}.

METHOD	*B. anthracis*	*F. tularensis*	*Y. pestis*	*S. typhi*
INDEP.	27.8± 4	25.7± 5.4	28.8± 4	72.5 ± 11.4
MEAN MTL	25.2 ±4.9	26.7±4	27.5±6.3	69.4±12.1
L1/L2 REG.	31.2 ±3.9	**30.6** ±7.1	31.9 ±4	73 ± 17.8
MTPL	**32** ±3.9	30.1± 5.8	32.1 ± 2.5	75.8 ± 12.1
U-MTPL	31.7 ±4	30.1±6.6	**32.5** ±3	76.1 ± 11.6
U-MTPL PHY.	31.7 ±4.2	29.6±7.5	31.9 ±3	**76.4** ±12

Table 3.7: Performance on four tasks. Averaged F-score from a 10 fold cross-validation. The standard deviation over the 10 folds is also reported.

METHOD	*B. anthracis*	*F. tularensis*	*Y. pestis*	*S. typhi*	*E. coli*
STL	27.8 ± 4.0	25.7 ± 5.4	28.8 ± 4.0	72.5 ± 11.4	73.7 ± 18.1
L1/L2 REG.	30.8 ± 3.9	30.1 ± 5.9	**32.2** ± 3.5	71.0 ± 16.9	**79.2** ± 14.5
MTPL	31.4 ± 3.6	29.4 ± 6.6	31.7 ± 3.4	70.9 ± 22.5	73.4 ± 16.5
U-MTPL	30.6 ± 3.2	29.0 ± 5.4	30.0 ± 3.4	70.7 ± 14.0	72.4 ± 16.0
U-MTPL PHY.	**32.1** ± 3.5	**30.3** ± 7.0	**32.2** ± 3.6	**74.8** ± 13.9	75.7 ± 18.0

Table 3.8: Averaged 10 fold cross-validation performance for all methods for a positive:negative class ratio of 1:100. Accuracy is reported as the F1 measure computed on the positive class. The standard deviation over the 10 folds is also reported.

Tuning the hyper-parameters

The function in equation 3.16 is highly non-convex and has many local minima. Doing a complete grid-search over R^{m^2} is very expensive, so we use some local search strategies to tune the hyper-parameters. The range of values for a single λ_{st} was $r = [10^{-6}, 100]$. The task-similarity parameters can be considered to be a symmetric matrix $\Lambda \in R^{m \times m}$. We first randomly sample 50 matrices from $R_r^{m \times m}$ and pick the top five models (that have the best F-score on the held-out data). Using these five matrices as starting points, we do a local search, by gradually varying some of the lambdas to find other good matrices. This strategy gives us a better sampling of the hyper-parameter space, biased towards settings with a good held-out performance.

Since we know the task relationships via the phylogenetic tree connecting the various bacteria, we can use it to set the parameters as follows: $\lambda_{st} = \frac{1}{\text{dist(s,t)}}$, where 'dist(s,t)' is the hop distance between the tasks s and t.

Experiments

We show results in two settings: the first setting has the same four tasks that were used to evaluate MTPL and another setting where introduce a fifth task: *E. coli*-human PPI prediction. We evaluate the unified model U-MTPL as before and compare it with the baselines. The joint learning method, 'U-MTPL phy' uses the phylogenetic-similarity between two species to fix the value of λ_{st}. This does not require any hyper-parameter tuning and involves running a single optimization. This makes it the most computationally efficient of all the methods that we consider.

Table 3.7 shows the averaged F-score from a 10 fold CV experiment for the first

setting with four tasks. The F-scores upon addition of the *E. coli* task are shown in Table 3.8. Overall, we find that the performance of all methods reduces slightly going from four tasks to five tasks. The drop in performance is biggest for *Salmonella Typhimurium*. The L1/L2 regularization does much better on the *E. coli* task than U-MTPL.

3.8 Co-immunoprecipitation (co-ip) studies: validation of predicted *Salmonella* interactions

Several experimental techniques have been developed to identify protein-protein interactions. Generally, these use a "bait" protein of interest to search a pool of cellular proteins for an interacting partner, coupled with some form of mechanism to detect the partner proteins. The bait protein or antigen is the pathogen (i.e bacterial) protein. Figure 3.10 shows the details of this procedure and explains each step. At a high level, interactions that exist between two proteins inside a cell, remain intact when the cell is lysed (i.e dissolved) under nondenaturing conditions. By immunoprecipitating protein A, we can precipitate protein B as well, wherever it is stably interacting with A (hence the name "co-ip"). Co-immunoprecipitated proteins are finally detected by western blotting.

In the co-ip experiments performed by our collaborators, the HT29 human epithelial cell line was used. The goal was to validate interactions with Salmonella effector proteins only. Out of a total of 84 effectors, 32 were picked based on the highly ranking PPIs from the model. From these, they were able to successfully set-up the co-ip protocol for 10 effectors. Each experiment thus investigates possible interactions between 10 *Salmonella* proteins and $\approx$ 25000 human proteins. Three such screens (i.e replicates) were performed, with the same protocols being followed in all screens. A total of 7414 potential PPIs were obtained over the three screens (i.e these PPIs appeared in at least one screen). The overlap between all three screens was very small: 5 PPIs. Hence, we present results on the set of 7414 PPIs. Overall we would like to caution the interpretation of these results from the pulldown experiments. These are to be considered less reliable and more of a 'silver standard' set, the reason being that none of the gold standard PPIs were found to be interacting in our pulldown experiments.

We check how many of these were predicted by our model, and we compare this number to the results from five other prediction models:

- Single task learning (STL): the per task independent model described in §3.5
- Group lasso: the ℓ_1/ℓ_2 regularization model, also discussed in §3.5
- BIANA: the interologs based model from Garcia-Garcia et al. [2012]
- iLoops[2] [Planas-Iglesias et al., 2013] A method that uses local structural features with a Random Forest classifier to predict interactions
- U-MTPL: our unified multitask learning pathway based learning objective from §3.7

[2] `http://sbi.imim.es/iLoops.php`

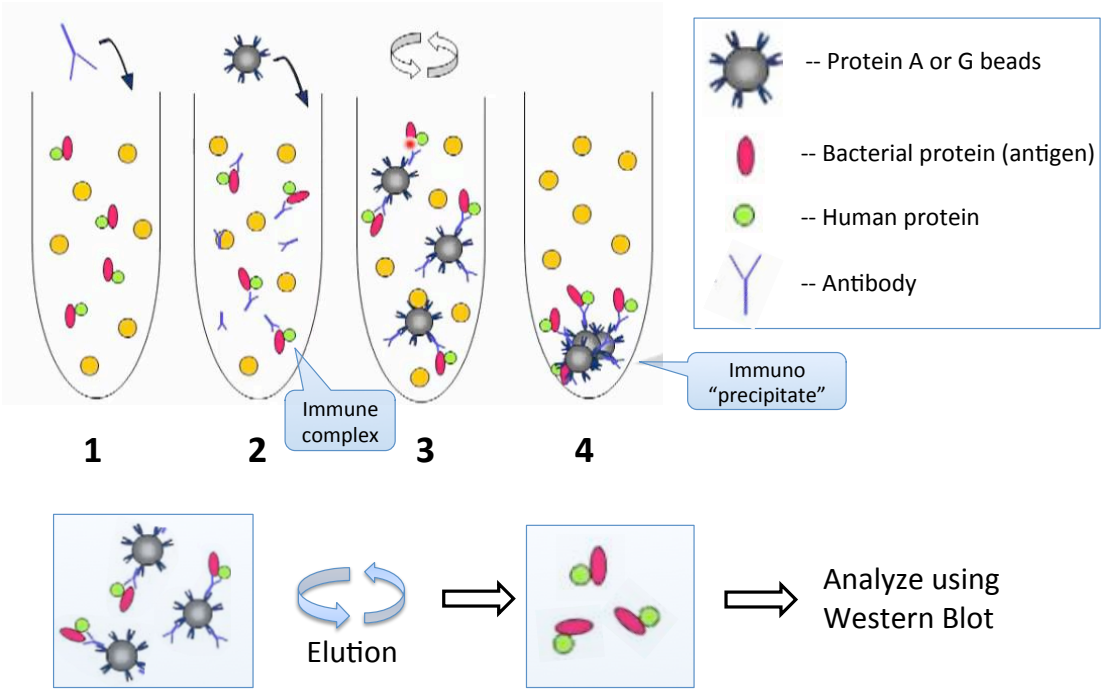


Figure 3.10: Schematic showing one standard procedure for Co-immunoprecipitation analysis of protein interactions. 1. The procedure starts with a cell lysate which contains the protein interactions. The bacterial protein is the antigen. An antigen-specific antibody (Y) is added, which binds to it. 2. Protein G-beads are introduced. The "immune complexes" bind to the beads (or are captured by a beaded support). 3. The beaded complexes are separated from the unbound proteins in step 4, which are then washed away. 5. Elution is used to remove the complexes from the G-beads. Finally the complexes left behind are analyzed using a method like Western Blot which helps identify the binding partner.

The interaction probabilities that we have from the other methods (BIANA, iLoops, Conformal predictor) are available only on the PPIs from the pulldown experiments (i.e the 7414 PPIs). Hence we can only compare the Recall across all these methods and ours.[3]

The recall for all methods was computed by assuming the default threshold for the method (sign($f(x)$ for the classifiers). U-MTPL's recall is substantially higher than that of the other methods. The interologs based method BIANA uses PPIs from several organisms (known PPIs both within species and cross-species) to infer potential human-*Salmonella* PPIs. As we noted in the analysis from §3.3, human-bacteria PPIs do not seem to exhibit interologs. Hence the recall of BIANA is very low. The I-loops server's reliance on protein structures makes it inapplicable on proteins that do not have a known 3D structure, explaining the poor recall. Surprisingly, Group lasso performs worse than the STL model.

In Figure 3.12 we show the precision as well for MTPL and two baseline models that we constructed. We consider the *Salmonella* and human proteins which were used in the pulldown experiments. This involves 10 *Salmonella* proteins (all effectors)

[3]To compare Precision (or F-score), we will need predictions on all possible *Salmonella*-human protein pairs, so that we can obtain the number of false positives.

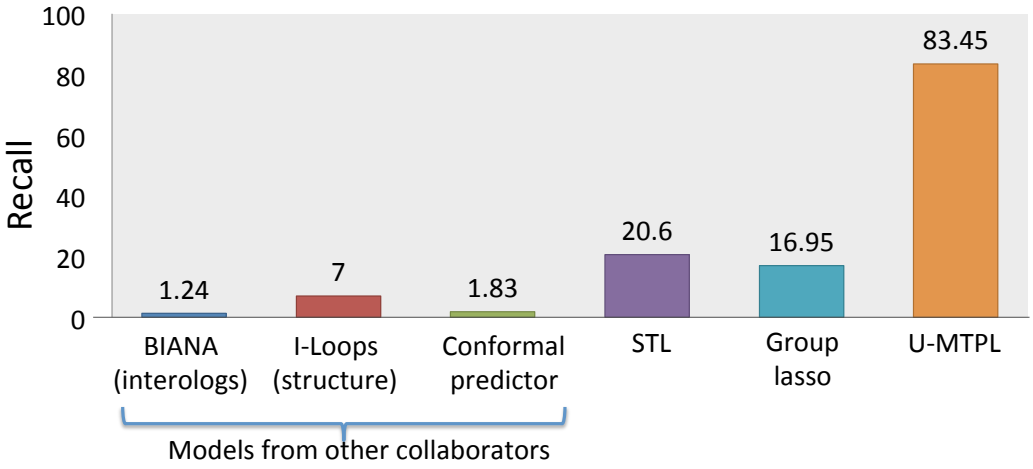


Figure 3.11: Recall on the 7414 PPIs from the co-immunoprecipitation experiments.

Number of true positives in top	Methods		
	STL	Group Lasso	MTPL
2000	147	143	132
5000	429	375	329
10000	930	793	706
20000	1907	1641	1589
50000	4460	4419	4747
70000	6622	6553	6720

Table 3.9: The number of positives retrieved by each method in their top predictions.

and the HT29 human cell line. For human proteins we consider the 24000 'reviewed' proteins from UniprotKB. The precision was computed by considering the false positives over this set. We notice that STL has a higher precision at lower recall values whereas MTPL has a higher precision at higher recall. We look at this performance in further detail in Table 3.8, where we list the number of positives reported in the top ranking predictions of each method. For instance the first row tells how many pull-down interactions were present in the top 2000 predictions (sorted by the classifier score / probability). Interestingly, STL has the best numbers in the initial rows and MTPL has better numbers in later rows, which is consistent with what we observe in the P-R curve.

3.9 Conclusion

We presented a method that uses biological knowledge in jointly learning multiple PPI prediction tasks. Using a task regularization based multi-task learning technique, we were able to encode a biological hypothesis into the optimization framework effectively thus enabling the commonality hypothesis to be tested. Our PPI prediction

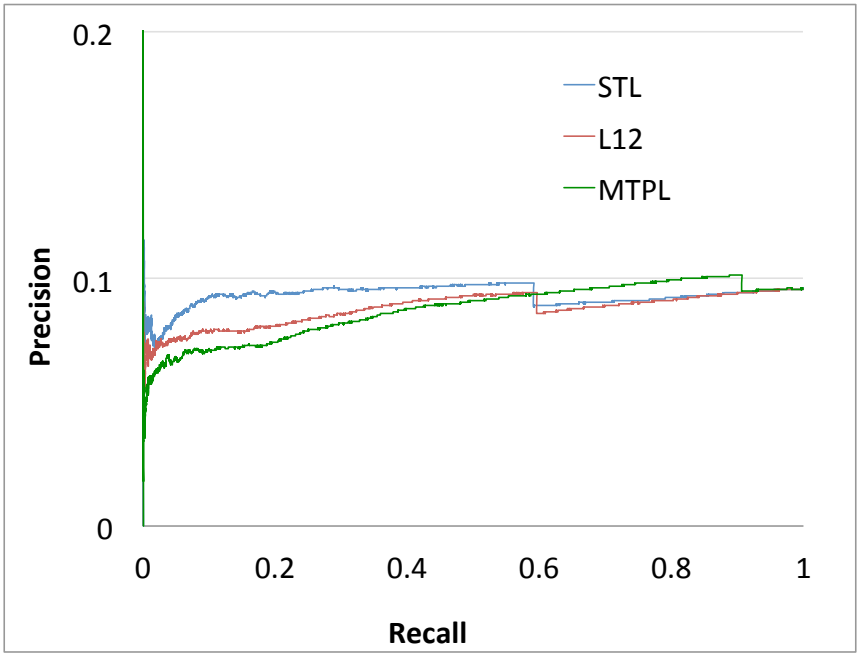


Figure 3.12: Precision-Recall curve for MTPL and two of our baselines on the 7414 PPIs from the co-immunoprecipitation experiments. The precision was computed w.r.t the set of all protein pairs investigated by the pulldown experiment.

results indicate that the tasks benefit from multitask learning as we see that the MTL methods outperform the STL baseline.

We validate the predictions from our model, via co-immunoprecipitation (pull-down) experiments. The higher recall we obtain clearly shows the improved coverage we get by incorporating PPI data from several pathogens. We also show how our model can be used to incorporate another hypothesis regarding host-pathogen PPI prediction. Our model has applications in other prediction problems such as: gene-gene interaction prediction across several organisms, gene-disease association prediction across diseases.

Chapter 4

Multitask matrix completion

In Chapter §3 we saw how the similarity in biological pathways was incorporated into a MTL framework. While we were able to obtain good performance on the tasks we considered, the MTPL method has the disadvantage of not being general enough to apply on problems arising in other areas. Here, we look at an alternate representation of the PPI prediction problem that allows us to explore a whole different mechanism to share information across tasks. The methods we develop here are very general and applicable to many applications that involve graphs and link prediction.

An elegant way to formulate the PPI prediction problem is via a graph completion based framework, where we have several bipartite graphs over multiple hosts and pathogens as illustrated in Figure 4.1. Nodes in the graphs represent host proteins (circles) and pathogen proteins (triangles), with edges between them representing interactions (host protein $\underline{interacts}$ pathogen protein). Given some observed edges (interactions obtained from laboratory based experiments), we wish to predict the other edges in the graphs. Such bipartite graphs arise in a plethora of problems including: recommendation systems (user $\underline{prefers}$ movie), citation networks (author $\underline{cites}$ paper), disease-gene networks (gene $\underline{influences}$ disease) etc. In our problem, each bipartite graph $\mathcal{G}$ can be represented using a matrix M, whose the rows correspond to pathogen proteins and columns correspond to host proteins. The matrix entry M_{ij} encodes the edge between host protein i and pathogen protein j from the graph, with $M_{ij} = 1$ for the observed interactions. Thus, the graph completion problem can be mathematically modeled as a matrix completion problem [Candes and Recht, 2008]. Traditional approaches towards matrix completion rely on the assumption that the underlying function that generated the matrix can be decomposed into a small number of 'latent' factors. The solution based on this assumption involves finding a low-rank matrix factorization for M, mathematically expressed as: finding U and V such that $M \approx UV^T$. Here the parameters U and V are called the factor matrices and represent the latent properties of the host and pathogen proteins respectively. In the recommendation systems example, the 'low rank' structure suggests that movies can be grouped into a small number of latent 'genres'. In the case of proteins these latent properties could correspond to various *biological functions* of proteins or encode *interaction propensities*.

Most of the prior work on host-pathogen PPI prediction has modeled each bipartite graph separately, and hence cannot exploit the similarities in the edges across the various graphs. Here we present a *multitask* matrix completion method that *jointly models* several bipartite graphs by sharing information across them. From the mul-

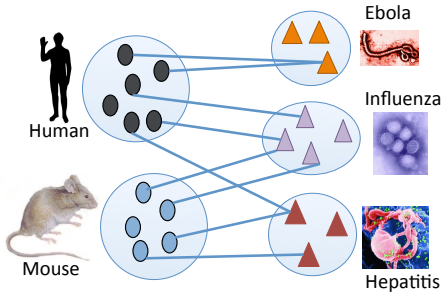


Figure 4.1: Multiple bipartite graphs with different types of nodes: on the left are proteins from host species and on the right virus species' proteins. Edges represent protein interactions. Each bipartite graph is one task.

Pathogen →	*Influ. A*	*Hep. C*	*Ebola*
No. of HP PPIs (positives)	848	981	90
no. unique virus proteins in PPIs	54	151	2
no. unique human proteins in PPIs	362	385	88
Density (%) of obs. graph‡	.006	.020	.038
total # of proteins in the virus	542	163	150
No. of negatives	84800	98100	9000

HP PPI: host-pathogen protein protein interactions
‡: considering all proteins in the two organisms involved.
Note: the total number of human proteins is ≈ 26000.

Table 4.1: Tasks and their sizes. Each column corresponds to one bipartite graph between human proteins and the pathogen indicated in the column header. All pathogens are single stranded RNA viruses. Row 4 shows that each of our graphs is extremely sparse.

titask perspective, a *task* is the graph between one host and one pathogen (can also be seen as interactions relevant to one disease). We focus on the setting where we have a single host species (human) and several related viruses, where we hope to gain from the fact that similar viruses will have similar strategies to infect and hijack biological processes in the human body. Such opportunities for sharing arise in other applications as well: for instance, predicting user preferences in movies may inform preferences in selection of books, or vice-versa, as movies and books are semantically related. Multitask learning based models that incorporate and exploit these correlations should benefit from the additional information.

Our multitask matrix completion based model is motivated by the following biological intuition governing protein interactions across diseases.

1. An interaction depends on the structural properties of the proteins, which are conserved across similar viruses as they have evolved from common ancestors. Our model thus needs a component to capture these latent similarities, which is *shared* across tasks.

2. In addition to the shared properties discussed above, each pathogen has also evolved specialized mechanisms to target host proteins. These are unique to the pathogen and can be expressed using a *task-specific* parameter in the model.

To incorporate the above ideas, we assume that the interactions matrix M is generated from two components. The first component low-rank latent factors over the human and virus proteins, with these latent factors jointly learned over all tasks. The second component involves task specific parameter, on which we additionally impose a sparsity constraint as we do not want this parameter to overfit the data. Section 4.4 discusses our model in detail. We trade-off the relative importance of the two

components using task-specific hyperparameters. Our model can thus learn what is conserved and what is different across pathogens, rather than having to specify it manually.

The key challenges in inducing such a model are:

- In addition to the interactions from each graph, it should exploit information available in the form of features.
- Exploiting features is particularly crucial since the graph $\mathcal{G}$ is often extremely sparse, i.e there are a large number of nodes and very few edges are observed. There will be proteins (i.e nodes) that are not involved in any known interactions – called the *cold start problem* in the recommendation systems community. The model should be able to predict the existence of links (or their absence) between such prior 'unseen' node pairs. This is of particular significance in graphs that capture biological phenomena. For instance, the host-pathogen PPI network of human-Ebola virus (column-3, Table 4.1) has ≈ 90 observed edges (equivalent to 0.038% of the network) which involve only 2 distinct virus proteins. Any biologist studying virus-human interactions will be interested in knowing more about the hundreds of other virus proteins (which have yet unknown interactions).
- A side-effect of having scarce data is the availability of a large number of unlabeled examples, i.e pairs of nodes with no edge between them. These unlabeled examples can contain information about the graph as a whole, and a good model should be able to use them.

The model we develop addresses these challenges, and has the following merits.

1. Our multitask extension of the matrix completion model from [Abernethy et al., 2009] is novel
2. Unlike most prior approaches (see Section 4.1 for details), our model exploits node-based features which allows us to deal with the 'cold start' problem (generating predictions on unseen nodes)
3. We apply the model to an important, real-world problem – prediction of interactions in disease-relevant host-pathogen protein networks, for multiple related diseases. We demonstrate the superior performance of our model over prior state-of-art multitask models
4. We use unlabeled data to initialize the parameters of our model, which serves as a prior. This gives us a modest boost in prediction performance

4.1 Prior work

Most of the prior work in PPI prediction has focussed on building models separately for individual organisms [Chen and Liu, 2005, Wu et al., 2006, Singh et al., 2006, Qi et al., 2006] or on building a model specific to a disease in the case of host-pathogen PPI prediction [Tastan et al., 2009, Qi et al., 2009, Dyer et al., 2007, 2011, Kshirsagar et al., 2012]. There has been little work on combining PPI datasets with the goal of improving prediction performance for multiple organisms. Qi et al. [2010] proposed

a semi-supervised multi-task framework to predict PPIs from partially labeled reference sets. Kshirsagar et al. [2013] proposed a task regularization based framework called MTPL that incorporates the similarity in biological pathways targeted by various diseases to couple multiple tasks together. Matrix factorization based protein-protein interaction (PPI) prediction has seen very little work, mainly due to the extremely sparse nature of these datasets which makes it very difficult to get reliable predictors. Xu et al. [2010] use a CMF-based approach in a multi-task learning setting for within species PPI prediction. The methods used in all prior work on PPI prediction do not explicitly model the features of the proteins and cannot be applied on proteins which have no known interactions available. Our work addresses both these issues in using a formulation of the matrix completion problem originally proposed in the work by Abernethy et al. [2009].

A majority of the prior work in the relevant areas of collaborative filtering and link prediction includes single relation models that use neighbourhood based prediction [Sarwar et al., 2001], matrix factorization based approaches [Koren et al., 2009, Menon and Elkan, 2011] and bayesian approaches using graphical models [Jin et al., 2002, Phung et al., 2009]. The matrix factorization based methods have been more popular than the other methods. There have also been multitask approaches on link prediction [Zhang et al., 2012, Cao et al., 2010, Li et al., 2009, Singh and Gordon, 2008]. Li [2011] presents a survey on multitask/transfer methods in collaborative filtering. The multi-relational learning literature [Xu et al., 2009] and the work on link prediction in heterogenous networks is not relevant as their setting involves *different types of relationships* between the *same* set of nodes. Menon and Elkan [2011] propose a single-graph model that combines linear and bilinear features, latent parameters on the nodes and several other parameters into a function that minimizes a ranking loss. In the matrix decomposition literature, Abernethy et al. [2009] proposed an approach where they cast the problem of matrix completion in terms of the abstract problem of learning linear operators. Their framework allows the incorporation of features and kernels. We extend their bilinear model for the multitask setting. There has been a lot of work on other low-rank models for multitask learning [Ando and Zhang, 2005, Ji and Ye, 2009, Chen et al., 2012a, 2013] that try to capture the task relationship via a shared low-rank structure over the model parameters.

4.2 Datasets and features

We use three human-virus PPI datasets from the PHISTO [Tekir et al., 2012] database, the characteristics of which are summarized in Table 4.1. The *Influenza A* task includes various strains of flu: H1N1, H3N2, H5N1 etc. Similarly, the *Hepatitis* task includes various sub-strains of the virus. All three are single-strand RNA viruses, with *Hepatitis* being a positive-strand ssRNA whereas *Influenza* and *Ebola* are negative-strand viruses. The phylogenetic tree that shows the connections between these viruses is shown in Figure 4.2. The density of the known interactions is quite small when considering the entire proteome (i.e all known proteins) of the host and pathogen species (row-4 in Table 4.1).

We use protein sequence based n-grams as features with n=2,3 and 4 for both human and viral proteins. The features which have been successfully applied in prior work [Dyer et al., 2011, Kshirsagar et al., 2013] also incorporate the properties of in-

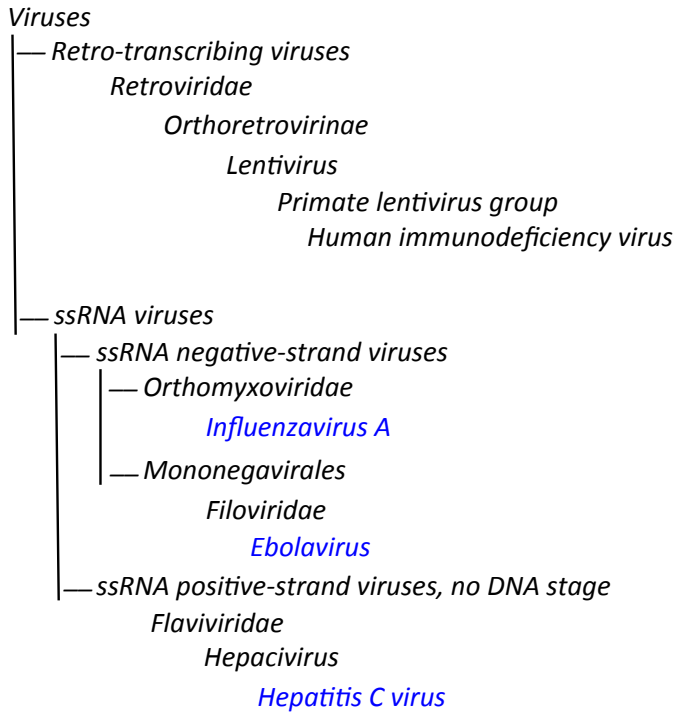


Figure 4.2: Genealogy of the viruses that we consider in this work.

dividual amino-acids such as charge, hydrophobicity etc. The number of features is ≈3000. Please refer to §2.3 for a detailed description of the protein sequence features.

4.3 Bilinear low-rank matrix decomposition

In this section, we present the matrix decomposition model [Abernethy et al., 2009] that we extend for the multitask scenario. In the context of our problem, at a high level, this model states that – protein interactions can be expressed as dot products of features in a lower dimensional subspace.

Let $\mathcal{G}_t$ be a bipartite graph connecting nodes of type υ with nodes of type ς. Let there be m_t nodes of type υ and n_t nodes of type ς. We denote by $M \in \mathbb{R}^{m_t \times n_t}$, the matrix representing the edges in $\mathcal{G}_t$. Let the set of observed edges be Ω. Let $\mathcal{X}$ and $\mathcal{Y}$ be the feature spaces for the node types υ and ς respectively. For the sake of notational convenience we assume that the two feature spaces have the same dimension d_t [1]. Let $\mathbf{x}_i \in \mathcal{X}$ denote the feature vector for a node i of type υ and $\mathbf{y}_j \in \mathcal{Y}$ be the feature vector for node j of type ς. The goal of the general matrix completion problem is to learn a function $f : \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ that also explains the observed entries in the matrix M. We assume that the function f is bilinear on $\mathcal{X} \times \mathcal{Y}$ and takes the following form:

$$f(\mathbf{x}_i, \mathbf{y}_j) = \mathbf{x}_i^\intercal H \mathbf{y}_j = \mathbf{x}_i^\intercal U V^\intercal \mathbf{y}_j \tag{4.1}$$

The factor $H \in \mathbb{R}^{d_t \times d_t}$ maps the two feature spaces $\mathcal{X}$ and $\mathcal{Y}$. This model assumes that H has a low-rank factorization given by $H = UV^\intercal$, where $U \in \mathbb{R}^{d_t \times k}$

[1] the dimensions being different does not influence the method or the optimization in any way

and $V \in \mathbb{R}^{d_t \times k}$. The factors U and V project the two feature spaces to a common lower-dimensional subspace of dimension k. While the dimensionality of the feature spaces $\mathcal{X}$ and $\mathcal{Y}$ may be very large, the latent lower dimensional subspace is sufficient to capture all the information pertinent to interactions. To predict whether two new nodes (i.e nodes with no observed edges) with features $\mathbf{p}_i$ and $\mathbf{q}_j$ interact, we simply need to compute the product: $\mathbf{p}_i U V^\intercal \mathbf{q}_j$. This enables the model to avoid the cold start problem that many prior models suffer from. The objective function to learn the parameters of this model has two main terms: (1) a data-fitting term, which imposes a penalty for deviating from the observed entries in Ω and (2) a low-rank enforcing term on the matrix H.

The first term can be any loss function such as squared error, logistic-loss, hinge loss. We tried both squared error and logistic-loss and found the performance to be similar. The squared error function has the advantage of being amenable to adaptive step-size based optimization which results in a much faster convergence. The low-rank constraint on H (mentioned in (2) above) is NP-hard to solve and it is standard practice to replace it with either the trace-norm or the nuclear norm. Minimizing the trace norm (i.e. sum of singular values) of $H = UV^\intercal$, is equivalent to minimizing $\|U\|_F^2 + \|V\|_F^2$. This choice makes the overall function easier to optimize:

$$\mathcal{L}(U,V) = \sum_{(i,j)\in\Omega} c_{ij}\ \ell(M_{ij}, \mathbf{x}_i^\intercal U V^\intercal \mathbf{y}_j) + \lambda(\|U\|_F^2 + \|V\|_F^2) \text{ where } \ell(a,b) = (a-b)^2 \quad (4.2)$$

The constant c_{ij} is the weight/cost associated with the edge (i,j) which allows us to penalize the error on individual instances independently. The parameter λ controls the trade-off between the loss term and the regularizer. The function in equation(4.2) is non-convex. To optimize this function a common procedure called alternating minimization (or alternating least squares) is used, which is similar to block coordinate descent. At every iteration in the optimization, it fixes one of the two parameters and optimizes w.r.t the other.

4.4 The bilinear sparse low-rank multitask model (BSL-MTL)

In the previous section, we described the bilinear low-rank model for matrix completion. Note that in order to capture linear functions over the features, we introduce a constant feature for every protein (i.e $[\mathbf{x}_i 1]$). We now discuss the multitask extensions that we propose. Let $\{\mathcal{G}_t\}$ where $t = 1 \ldots T$ be the set of T bipartite graphs and the corresponding matrices be $\{M_t\}$. Each matrix M_t has rows corresponding to node type υ_t and columns corresponding to the node type ς_t. The feature vectors for individual nodes of the two types be represented by $\mathbf{x}_{t_i}$ and $\mathbf{y}_{t_j}$ respectively. Let Ω_t be the set of observed links in the graph $\mathcal{G}_t$. Our goal is to learn individual link prediction functions f_t for each graph. In order to exploit the relatedness of the T bipartite graphs, we make some assumptions on how they share information. We assume that each matrix M_t has a low-rank decomposition that is shared across all graphs and a sparse component that is specific to the task t. That is,

$$f_t(\mathbf{x}_{ti}, \mathbf{y}_{tj}) = \mathbf{x}_{ti}^\intercal H \mathbf{y}_{tj}, \text{ where } H = UV^\intercal + S_t \quad (4.3)$$

As before, the shared factors U and V are both $\mathbb{R}^{d_t \times k}$ (where the common dimensionality d_t of the two node types is assumed for convenience). The matrix $S_t \in \mathbb{R}^{d_t \times d_t}$

is a sparse matrix. The objective function for the multitask model is given by:

$$\mathcal{L}(U, V, \{S_t\}) = \frac{1}{N} \sum_{t=1}^{T} \sum_{(i,j)\in\Omega_t} c_{ij}^t \left(M_{t_{ij}} - \mathbf{x}_{ti}^\intercal (UV^\intercal + S_t)\mathbf{y}_{tj}\right)^2 + \lambda(\|U\|_F^2 + \|V\|_F^2) + \sum_{t=1}^{T} \sigma_t \|S_t\|_1 \tag{4.4}$$

Here $N = \sum_t |\Omega_t|$, is the total number of training examples from all tasks. To enforce the sparsity of S_t we apply an ℓ_1 norm. In our experiments, we tried both ℓ_1 and ℓ_2 norms and found that the ℓ_1 norm works better.

Optimization

The function $\mathcal{L}(U, V, \{S_t\})$ is non-convex. However, it is convex in every one of the parameters (i.e when the other parameters are fixed) and a block coordinate descent method called alternating least squares (ALS) is commonly used to optimize such functions. To speed up convergence we use an adaptive step size. The detailed optimization procedure is listed in Algorithm 2.

Algorithm 2: AltMin algorithm

Input:
k : number of latent factors
Γ: pairs of entities for initialization
For every task t,
$\{\mathbf{x}_{ti}\}, \{\mathbf{y}_{tj}\}$: feature vectors from the two entity types
Ω_t: the observed entries of the matrix M_t
Initialization:
An iteration r, let $\mathbf{S}^r$ represent $\{S_t^r\}_{t=1}^T$
$\mathbf{S}^0 \leftarrow 0$
$U^0 \leftarrow$ top-k left singular vectors and $V^0 \leftarrow$ top-k right singular vectors from the SVD of $\sum_{(p,q)\in\Gamma} \mathbf{x}_p \mathbf{y}_q^\intercal$
$\mathcal{L}^0$: initial loss
repeat
  $U^{r+1} \leftarrow \underset{U}{\operatorname{argmin}}\ \mathcal{L}(U, V^r, \mathbf{S}^r)$
  $V^{r+1} \leftarrow \underset{V}{\operatorname{argmin}}\ \mathcal{L}(U^{r+1}, V, \mathbf{S}^r)$
  For each task t
    $S_t^{r+1} \leftarrow \underset{S_t}{\operatorname{argmin}}\ \mathcal{L}(U^{r+1}, V^{r+1}, \mathbf{S}_{-t}^r)$
  Compute $\mathcal{L}^{r+1}$ and let $\delta \leftarrow (\mathcal{L}^r - \mathcal{L}^{r+1})/\mathcal{L}^r$
until $\delta < \tau$

Convergence: The ALS algorithm is guaranteed to converge only to a local minimum. There is work showing convergence guarantees to global optima for related simpler problems, however the assumptions on the matrix and the parameter structure are not very practical and it is difficult to verify whether they hold for our setting.

Initialization of U and V: We tried random initialization (where we randomly set the values to lie in [0 1]), and also the following strategies that initialize: $U^0 \leftarrow$ top-k left singular vectors, and $V^0 \leftarrow$ top-k right singular vectors from the SVD of

$\sum_{(i,j)\in\Gamma} m_{ij}\mathbf{x}_i\mathbf{y}_j^\intercal$. We set Γ to (a) training examples, or (b) a random sample of 10000 unlabeled data from all tasks. We found that using the unlabeled data for initialization gives us a better performance

Handling the 'curse of missing negatives'

For the MC algorithm to work in practice the matrix entries M_{ij} should represent interaction scores (range [0 1]) or take binary values (1s for positives and 0s for negatives). Our experiments with PPI probabilities (obtained using the MINT-scoring algorithm) gave bad models. The binary matrix setting requires some observed 0s. However non-interactions are not available as they cannot be verified experimentally for various reasons. Hence we derived a set of 'probable negatives' using a heuristic often used in PPI prediction work [Qi et al., 2006, 2009, Dyer et al., 2007, 2011, Kshirsagar et al., 2013]. We pair up all virus proteins with all human proteins and sample a random set to be negatives. This heuristic works in practice as the interaction ratio (i.e number of positives in a large random set of protein pairs) is expected to be very low: $\approx 1/100$ to $1/500$. That is, the probability that our negatives contain true positives is negligible.

High class imbalance

We incorporate the prior on the interaction ratio by setting the size of our randomly sampled negatives set equal to 100 times the number of gold standard positives.

4.5 Experimental setup

Our baselines include recent low-rank and sparse models, conventional multitask methods and prior work on HP PPI prediction. For a uniform comparison we used least squared loss in all the methods. The MALSAR package was used to implement some of the models. For the baselines wherever appropriate, we concatenated the features of the two node types into a single feature vector. Let $W \in \mathbb{R}^{T\times d_t}$ be the matrix with the task-specific weight vectors $\mathbf{w}_t$.

- **Single task (STL)**: We used ridge regression with ℓ_2 regularization (which performed better than ℓ_1)
- **MMTL**: The mean regularized multitask learning model [Evgeniou and Pontil, 2004]
- **Dirty model**: This model [Jalali et al., 2010] assumes that $W = P + Q$, where P enforces group sparsity and Q is to control for element-wise sparsity. It uses the regularizer $\rho_1\|P\|_{1,\infty} + \rho_2\|Q\|_1$
- **Low rank model**: A low-rank structure is enforced on W by minimizing the nuclear norm $\|W\|_*$
- **Sparse + low-rank**: This model [Chen et al., 2012a] is the closest to our work, with two main distinctions: the linear dependence over the features and the manner in which the low-rank assumption is incorporated. W is assumed to

have the decomposition: $W = P + Q$, where P is sparse and Q has a low-rank structure

- **IMC**: This is the link-prediction model from Section 4.3, where data from all tasks is combined without incorporating any task relationships. This model has been used in prior work [Natarajan and Dhillon, 2014] for gene-disease association prediction. U and V are shared by all tasks. We use the same initialization for this method as we do for our model. A comparison to this model tells us how much we gain from the task-specific sparsity component S_t
- **MTPL**: The biologically inspired pathway regularizer from Kshirsagar et al. [2013] (Chapter 3) is used to capture task similarity
- **BSL-MTL**: Bilinear sparse low-rank multitask learning, the method developed in this chapter

Evaluation setup

We compare all the methods in two settings, where a small proportion of the available labeled data is randomly sampled and used to train a model which is then evaluated on the remaining data. For the first setting we randomly split the labeled data from each task into 10% training and 90% test, such that the class-skew of 1:100 is maintained in both splits (i.e stratified splits). The second setting uses a 30% training, 70% test split. In each setting we generate ten random splits and average the performance over the ten runs.
Parameter tuning: We tune the hyper-parameters using a 3 fold CV on the training split. To tune the regularization parameters from all baselines, we searched over the following range (or grid, depending on the method) of values: $[100, ...10^{-3}]$. To address the class-skew we also try assigning a higher weight to the positives. For our model, we tried k: 5, ... 100 and λ=$\{1...10^{-3}\}$. For each task t, σ_t was varied over the values $10^{-3}, ...10^{-6}$. The optimal setting was: $k = 10$, $\lambda = 0.01$, $\sigma_{ebola} = 10^{-5}$, $\sigma_{flu} = \sigma_{hepc} = 10^{-6}$.

4.6 Results

We report the area under the precision recall curve (AUC-PR) along with the standard deviation in Table 4.2. AUC-PR has been shown to give a more informative picture of an algorithm's performance than ROC curves in high class imbalance datasets [Davis and Goadrich, 2006] such as ours. Note that the AUC-PR of a random classifier model is ≈ 0.01.

The first row (STL) is the single-task baseline and all others are multitask models. In general, we notice that multitask learning benefits all tasks. The first three columns show the results in the 10% setting. The number of training positive examples from each task are 8 for Ebola, 85 for Influenza and 98 for Hepatitis-C. Our model (last row) has significant gains for Influenza (1.4 times better than the next best) and modest improvements for the other tasks. The variance in the performance is high for the Ebola task (column 1) owing to the small number of positives in the training splits (8 positives). The most benefits for our model are seen in the 30% setting for all tasks, with improvements of 39%, 3% and 12% on the Ebola, Hepatitis and Influenza

	10% training			30% training		
	Ebola	*Hep-C*	*Influenza*	*Ebola*	*Hep-C*	*Influenza*
STL	0.189±.09	0.702±.08	0.286±.02	0.130±.03	0.802±.03	0.428±.03
MMTL	0.113±.04	0.767±.03	0.321±.02	0.129±.02	0.802±.04	0.430±.03
Trace-norm	0.199±.11	0.767±.03	0.318±.02	0.207±.02	0.808±.02	0.409±.03
Sparse,low-rank	0.144±.07	0.767±.02	0.318±.02	0.153±.02	**0.814**±.01	0.414±.03
Dirty model	0.074±.03	0.767±.04	0.324±.02	0.165±.02	0.813±.03	0.412±.03
MTPL	**0.217**±.08	0.695±.02	0.345±.02	**0.260**±.05	0.713±.01	**0.496**±.03
IMC	0.087±.04	**0.779**±.02	**0.362**±.01	0.122±.02	0.801±.01	0.410±.03
BSL-MTL	**0.233**±.10	**0.807**±.02	**0.486**±.02	**0.361**±.03	**0.842**±.01	**0.560**±.02

Table 4.2: Area Under the Precision-Recall curve for each task in the two settings. X% training indicates the fraction of the labeled data used for training and tuning the model with the rest (100-X)% used as test data. We report the average AUC-PR over 10 random train-test splits (stratified splits that maintain the class-skew of 1:100). The standard deviation is also shown. The performance of the best baseline and the overall best method (BSL-MTL) is highlighted in bold.

tasks respectively. Here, we see great improvements in the data-poor task, Ebola. The two closely related tasks, Influenza and Ebola benefit a lot more than the slightly distant Influenza (see Figure 4.2). This is consistent with the expectations for multi-task learning, where weakly-performing tasks are lifted more than strongly-performing tasks by borrowing from other related tasks.

Biological significance of the model

The model parameters U, V and S contain a lot of rich information which can be used to further understand host-pathogen interactions. Note that our features are derived from protein amino acid sequences which allows the following possibilities to interpret the parameters.

Clustering proteins based on interaction propensities: We analyze the proteins by projecting them using the model parameters U and V into a lower dimensional subspace (i.e computing $XU^\intercal$ and $YV^\intercal$ to get projections of the virus and human proteins resp.). The principal component analysis (PCA) of this lower dimensional representation is compared with PCA in the original feature space (protein sequence features). The two plots are shown in Figure 4.3 for the virus proteins. Firstly, the projected data has a much better separation than the original data. Secondly, the bottom plot tells us that Hepatitis-C and Influenza have many proteins with similar binding tendencies, and that these behave differently than most proteins from Ebola virus. This observation is not obvious in the PCA of the original feature space (top plot), where proteins with similar sequences cluster together. We further analyze clusters of proteins from the projected data using Gene Ontology (GO) annotations.

Sequence motifs from virus proteins: In Figure 4.4, we show sequence motifs derived from the top 100 k-mers that contribute to interactions. The shared k-mers that were used to generate the motif in the top plot were derived from $UV^\intercal$. The task-specific

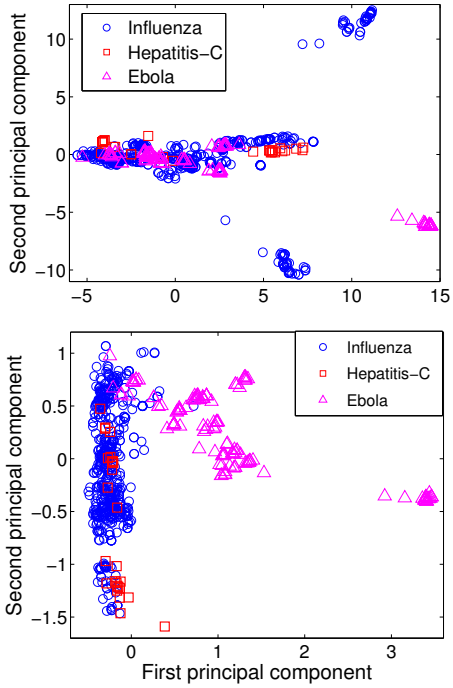


Figure 4.3: Principal component analysis of virus proteins in the original feature space (top) and projected subspace (bottom). Shape of the points indicates which virus that protein comes from. The first two principal components are shown.

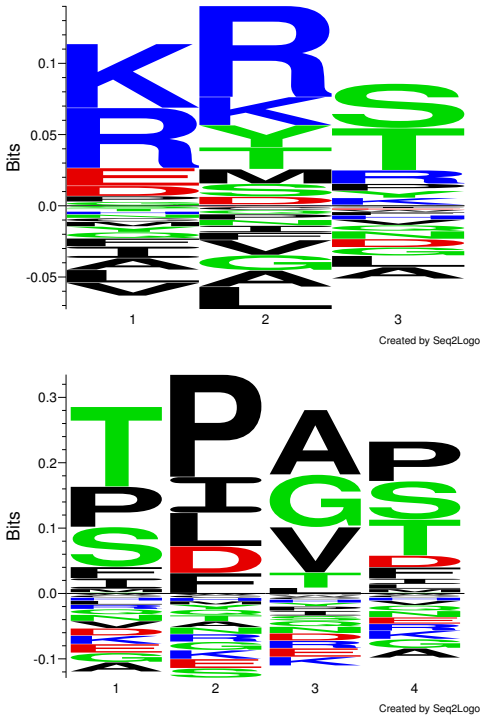


Figure 4.4: Sequence motifs that contribute significantly to interactions across all viruses (top) and that is specific to *Ebola* virus (bottom). See Section 4.6 for details.

k-mers for the *Ebola* virus (bottom plot) were obtained from the matrix S_t (t=ebola). Knowledge of these pathogen-specific k-mers can help in design of drugs to target specific pathogens. We observe that the shared motif (top) is dominated by positively charged and hydrophilic amino acids (blue in colour), whereas the *Ebola* specific motif has mostly hydrophobic residues (black). We found experimental evidence for the significance of the *Ebola* motif: the pattern `PPAP` is part of a well-studied epitope in the Immune Epitope database [Vita et al., 2015] (epitope id: 66946). Using higher dimensional k-mers (where k=7, 8, 9) as features in our model is likely to produce such epitopes which can then be verified by biochemical peptide interaction assays. Our model thus has applications in epitope prediction as well, where conventional methods consist of scanning all possible k-mers from protein sequences to identify likely epitopes.

Novel interactions and interaction interfaces: The top four Ebola-human PPI are all predictions for the Ebola envelope glycoprotein (GP) with four different human proteins (Note: GP is not in the gold standard PPIs). We found evidence in published literature [Nanbo et al., 2010] for the critical role played by GP in virus docking and

[1] an epitope is a very short sequence from the virus that bind to human antibodies

fusion with the host cell. Our model not only provides predictions on whether or not two proteins interact, but can also provide hypotheses as to the putative binding sites for the interaction. This is of significance for the viruses (esp. Ebola) as they have very few proteins with known 3D structures. Traditional linear models do not give us correlations between amino acid residues. We selected the top 10 Ebola-human PPI predictions and performed protein-protein docking i.e simulation of their binding.Some of the sites shown to be in contact by the docking model also corresponded to the host-pathogen feature pairs that were responsible for that particular prediction. The 3D structure of one predicted interaction and it's binding interface is shown in Figure 4.5.

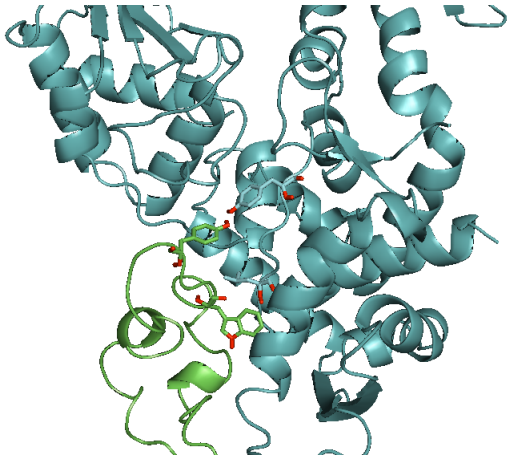


Figure 4.5: 3D structure obtained by docking ebola virion spike glycoprotein (green) with human ubiquitin-protein ligase (cyan). The putative binding sites are shown using sticks.

4.7 Conclusions and future extensions

Multitask link prediction is an important area with many applications ranging from recommendation systems to biomedical host-pathogen interactions. We developed and tested a new method based on low-rank matrix completion for sharing information across tasks and showed significant increases in prediction performance. The method was evaluated in the host-pathogen protein interaction domain for three pathogens (three tasks) and exhibited significant increases in link prediction accuracy. Analysis of the model parameters lead to interesting observations and insights about the data.

The model we present is general enough to be applicable on other problems such as: gene-disease relevance prediction across organisms or disease conditions, multitask collaborative filtering. As future work we intend to apply our method in some of these other settings.

We envision many potential extensions to this work, some of which are:
Applications: We would like to apply our method on a multi-host (e.g. human, mouse, bovine, etc.) multi-pathogen graph to determine if the same multitask learning advantages accrue. Beyond protein interaction graphs, the next step is to apply and evaluate the method for different families of tasks, such recommendation systems data.

Multiple levels: Task hierarchies can be exploited by our model by incorporating additional components to reflect the hierarchical relationships. *Multiple link types:* We investigated one type of link, namely interaction links, but there can be different types of related links, such as between people and scientific articles (e.g. authorship, citation, has-read, dislikes, etc.). It will be interesting to incorporate these into our models. The multiple tasks here are predictions in different but correlated link spaces: *authorship + citation = self-citation*, citation is correlated with has-read, citation is anti-correlated with dislikes, but not totally, e.g. if contrasting one's work with a less preferred alternative.

The model can also be further extended to incorporate complex task relationships available in the form of a hierarchy. Starting at the leaf-nodes of such a hierarchy, every subtree of sibling tasks will share parameters. Let $\mathcal{T}_{sib}$ be one such subtree of sibling tasks where each task's link matrix is expressed as a function of a shared low-rank component $U_{sib}V_{sib}^T$ and a task-specific component. Task-relationships of T_{sib} with other subtrees in the hierarchy and with higher level nodes will be expressed via additional shared parameters. These higher level relationships will again be expressed via low-rank components which are common to several subtrees. Let $U_{ancestor}$ and $V_{ancestor}$ be the matrices representing the shared low-rank component at a higher level of the hierarchy. Then, the link matrix of a task $T \in \mathcal{T}_{sib}$ can be written as: $X(U_{ancestor}V_{ancestor}^T + U_{sib}V_{sib}^T + S_T)Y^T$. In this manner, as we go higher and higher in the hierarchy we add low-rank components in the equation explaining the link matrices of the tasks. In order to avoid too many parameters we can restrict the extent to which information is shared by only introducing new parameters every alternate level as we go higher up in the hierarchy. Another possibility is not sharing any parameters at the top-most levels of the tree.

Overall, we feel that multitask learning in link prediction is still in the early stages of research and hope this contribution will stimulate further work.

Chapter 5

Transfer learning models for new hosts and pathogens

Understanding the workings of plant responses to pathogens is an important fundamental questions that also has enormous economic importance due to the role of pathogens in food production and processing. While "classical" plant pathogens cause crop losses during production by impacting on plant health, processing of plant-based food can lead to contamination by opportunistic pathogens. It is becoming increasingly supported by experimental evidence that some human bacterial pathogens can colonize plants and cause disease [Kirzinger et al., 2011]. *Salmonella* is one of these bacterial species with extremely broad host range that infects not only animals, but also plants [Hernandez-Reyes and Schikora, 2013]. Evidence increases that *Salmonella* can utilize plants as alternative host and can be considered as a bona fide plant pathogen. In this respect it has been reported that (a) *Salmonella* actively invades plant cells, proliferates there and can cause disease symptoms [Schikora et al., 2008, Berger et al., 2011] (b) the plant recognizes *Salmonella* and plant defense responses are activated [Iniguez et al., 2005, Schikora et al., 2008] and (c) that functional Type Three Secretion Systems (TTSS) 1 and 2 are important for *Salmonella* pathogenicity in plants with respect to bacterial proliferation and suppression of plant defense responses [Iniguez et al., 2005, Schikora et al., 2011, Shirron and Yaron, 2011]. *Salmonella* TTSS-1 and 2 encode proteins, so called effectors, which are known to be translocated into the animal host cell in order to manipulate host cell mechanisms mainly via PPIs [Schleker et al., 2012]. Hence, it may be assumed that *Salmonella* utilizes the same proteins during its communication with animals and plant. However, the details of this communication are not known. A critical component of the communication between any host and its pathogen are PPIs. However, the infection of plants by *Salmonella* is only a nascent field, so there are no known PPIs for *Salmonella* with any plant reported yet. Even for the well established pathogen-host pair, *Salmonella*-human, relatively few interactions are known [Schleker et al., 2012]. Only 62 interactions between *Salmonella* and mostly human proteins (some *Salmonella* interactions involve other mammalian species, such as mouse and rat) are known to date. Because there exists no plant-*Salmonella* interactions data, we need to rely on computational methods to predict them.

In our work [Kshirsagar et al., 2015a], we describe techniques to build computational models to predict interactions between the model plant, *A. thaliana*, and *S.* Typhimurium. Since there is no labeled data of this host-pathogen pair available,

we aim to transfer knowledge from known host-pathogen PPI data of other organisms. We use various statistical methods to build models for predicting host-pathogen PPIs. In each case, we cast the PPI prediction problem as a binary classification task, where given two proteins the goal is to learn a function that predicts whether the pair would interact or not. We derive features on every protein pair using protein sequence data. Each host-pathogen PPI prediction problem is considered as one task. Figure 5.1 shows our problem setting. The upper host-pathogen task with *Salmonella* as pathogen and human as the host is the *source* task. The lower task is the target task. The arrow shows the direction of knowledge transfer.

In order to transfer knowledge from one organism to another, we need to utilize some measure of similarity between them. This similarity can be defined between smaller units such as individual proteins or genes from the organisms or higher level units. The higher the similarity, the greater the information transfer between them. Hence the notion of similarity is very critical to the results we obtain from such a transfer based method and should be biologically motivated. Our methods enable the transfer of knowledge using the following mechanisms:

- We use the structural similarity between the individual proteins of the two hosts measured using protein sequence alignment. This follows from the biological intuition that structurally similar proteins in two different organisms are very likely to have similar functions. Hence a pathogen that wants to disrupt a specific function will target structurally similar proteins in different hosts.
- Interactome-level similarity, comparing the human PPI graph with the plant PPI graph. Any biological process in an organism involves the participation of several proteins and more importantly the interactions between these. By comparing the interactomes of different hosts, we are comparing them at the biological process-level. The components of the two graphs that are highly similar will most likely correspond to similar processes in the two organisms.
- Distributional similarity between the protein pairs: here, we identify which of the human-*Salmonella* protein pairs are the most similar (hence most relevant) to the plant-*Salmonella* protein pairs. This similarity is computed using the features of the protein-pairs. Since it is distributional similarity, it involves a comparison over all protein pairs from both organisms. Only the most relevant human-*Salmonella* protein pairs are used to build a model.

The main contributions of this work are:

1. We present methods that combine known PPIs from various sources to build a model for a new task
2. We evaluate our methods quantitatively and our results show the benefits in performance that are possible if we incorporate the similarity information discussed in the previous paragraphs
3. We present the first machine learning based predictions for plant-*Salmonella* PPIs

In the rest of this chapter, we start by describing the host-pathogen PPI datasets we use in §5.1, followed by a detailed description of our methods in §5.2 and a quantitative and qualitative analysis of the results in §5.4.

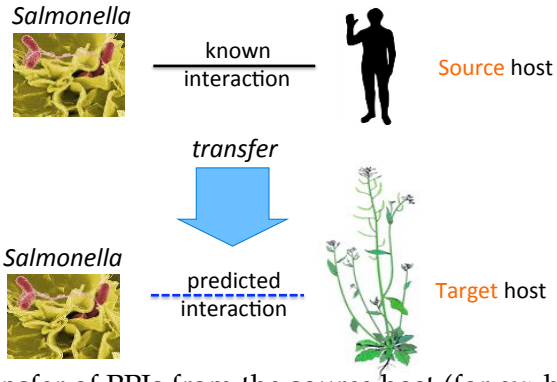


Figure 5.1: Transfer of PPIs from the source host (for ex: human) to another host, the target host (for example *Arabidopsis*), for the common pathogen, *Salmonella*.

5.1 Source tasks

As source tasks we used the known PPIs between various other hosts and pathogens. Many of these interactions were obtained from the PHISTO [Tekir et al., 2012] database which reports literature-curated known interactions. For PPIs between human and *Salmonella* we use the manually literature-curated interactions reported in Schleker et al. [2012]. Please note that all of these interactions come from biochemical and biophysical experiments. The details of the dataset used in each approach are shown in Table 5.1 and they are available for download from `http://www.cs.cmu.edu/~mkshirsa/data/frontiers2014/data.zip`. Our first approach is a rule-based approach and it uses human-*Salmonella* PPIs from two sources: the 62 experimentally generated PPIs reported in Schleker et al. [2012] and the predicted PPIs from Kshirsagar et al. [2012]. Please note that this is the only method that uses any predicted PPIs as "ground truth". All other methods discussed in subsequent sections do not use any predicted PPIs as source. They use only PPIs validated experimentally by biochemical and biophysical methods.

***Salmonella* species/strains considered:** The source data that we use for human-*Salmonella* from Schleker et al. [2012] comes from two different strains: *Salmonella* Typhimurium strain LT2 and *Salmonella* Typhimurium strain SL 1344. One of our three approaches (row-1 of Table 5.1) uses human-*Salmonella* predicted PPIs. These predicted PPIs from Kshirsagar et al. [2012] contain *Salmonella* proteins from two additional strains: *Salmonella* enteritidis PT4 and *Salmonella* Typhi. From henceforth, for the sake of brevity, we will refer to proteins from all strains as *Salmonella* proteins. For *Salmonella* proteins, we used the UniprotKB database [UniProt Consortium, 2011] to obtain all proteins from the various strains. For *Arabidopsis* thaliana proteins, we used the TAIR database [Lamesch et al., 2012].

5.2 Methods

In the previous section, we described the dataset used in our various approaches. We now describe the details of the methods we use.

Approach(es)	Source task(s)	Number of PPI	Dataset citation	Feature Set
1: Homology based	human-*Salmonella* (known PPI)	62	Schleker et al. [2012]	No feature set. Heuristics are used to infer interactions
	human-*Salmonella* (predictions)	190,868	Kshirsagar et al. [2012]⋆	
2: T-SVM#	human-*Salmonella* (known PPI)	62	Schleker et al. [2012]	(a) Protein sequence k-mers (b) Gene expression from GEO (c) GO term similarity
3: KMM†-SVM	human-*Salmonella* (known PPI)	62	Schleker et al. [2012]	Protein sequence k-mers
	human-*F. tularensis*	1380	PHISTO	
	human-*E. coli*	32		
	plant-*A. tumefaciens*	22		
	plant-*E. coli*	15		
	plant-*P. syringae*	13		
	plant-*Synechocyctis*	23		

⋆ This source reports predicted PPIs, while the others are all experimentally validated
† KMM: Kernel Mean Matching
# SVM: Support Vector Machine
GO: Gene Ontology

Table 5.1: Datasets used in the various approaches, their sizes and the appropriate citations

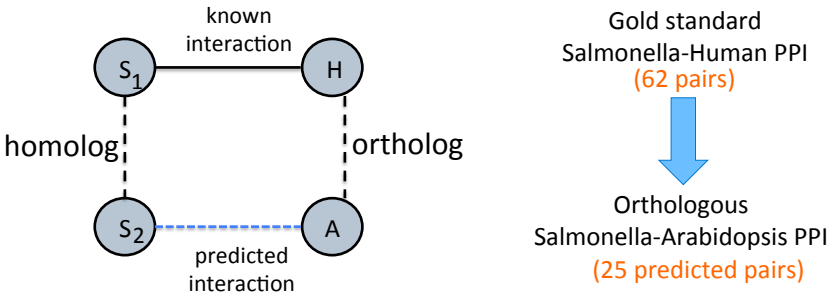


Figure 5.2: Approach-1 (a) Ortholog based protein interaction inference. 'S_1' represents a *Salmonella* protein and S_2 is the homolog of S_1 or S_1 itself. H represents a human protein and A represents an *Arabidopsis* protein that is an ortholog of the human protein.

Approach-1 : Homology based transfer

In this approach, we use the sequence similarity between the plant and human protein sequences to infer new interactions. We use two techniques to predict interactions between plant and *Salmonella* proteins. The first technique uses plant-human orthologs and the second is based on plant-human homology (sequence alignment scores). Both techniques use two sources of interactions: true PPIs from Schleker et al. [2012] and predicted PPIs from Kshirsagar et al. [2012]. Please note that this is the only method that uses any predicted PPIs as "ground truth".

Homologs and Orthologs (definition): Homologous pairs of genes are related by descent from a common ancestral DNA sequence. These can be either orthologs: genes that evolved from a common ancestral gene by speciation or paralogs: genes separated by the event of genetic duplication. We obtained orthologs from the InParanoid database [Ostlund et al., 2010]. To find homologous pairs of proteins, we used BLAST sequence alignment with an e-value threshold of 0.01.

(a) **Host ortholog based predictions**: We start with the known human-*Salmonella* PPIs. For each interaction, we search for an ortholog of the human protein in *Arabidopsis*. If one exists, we infer an interaction between the *Salmonella* and the *Arabidopsis* protein. This is similar to finding interologs, with the exception that we restrict ourselves to orthologs of the host protein rather than considering all possible homologs of both the host and pathogen proteins. Figure 5.2 illustrates this simple heuristic. There are 62 human-*Salmonella* PPIs in our dataset. Using this ortholog based inference for the host proteins, we obtained a total of 25 plant-*Salmonella* PPIs as some of the human proteins did not have any plant orthologs. The orthologous *Arabidopsis* proteins for the human proteins were obtained from the InParanoid database [Ostlund et al., 2010].

(b) **Host graph alignment based predictions**: This method uses homologs between the human and plant proteins. Since the set of known PPIs is very small (62 interactions), here we use them to generate 'bootstrap' interactions. The known 62 PPIs are used to build a classifier using the method published in Kshirsagar et al. [2012] to generate a total of 190,868 human-*Salmonella* PPI predictions. These predicted PPIs form the 'bootstrap' PPIs and will be used in a graph-based transfer approach. In this graph-based transfer method, we first align the PPI graphs of the two host organisms using NetworkBlast [Sharan et al., 2005]. The human PPI network was obtained from the HPRD database [Prasad et al., 2009] and the plant-plant PPIs from TAIR database [Lamesch et al., 2012]. The algorithm aligns the human PPI graph with the plant PPI graph using the pairs of homologous proteins between the two organisms. To find the homologous proteins, we used BLAST sequence alignment with an e-value threshold of 0.01. Next, we use NetworkBlast to find the graph components that are the most similar across the two graphs. We call them the 'enriched components'. By comparing the interactomes of the two hosts, we are comparing them at the biological process-level. The components of the two graphs that are highly similar will most likely correspond to similar processes in the two organisms. NetworkBlast finds a total of 2329 enriched protein complex pairs between the two host organisms. Figure 5.3 shows one such enriched protein complex pair: the complex on the left is from *Arabidopsis* and the one on the right is from human. Using these we determine the plant

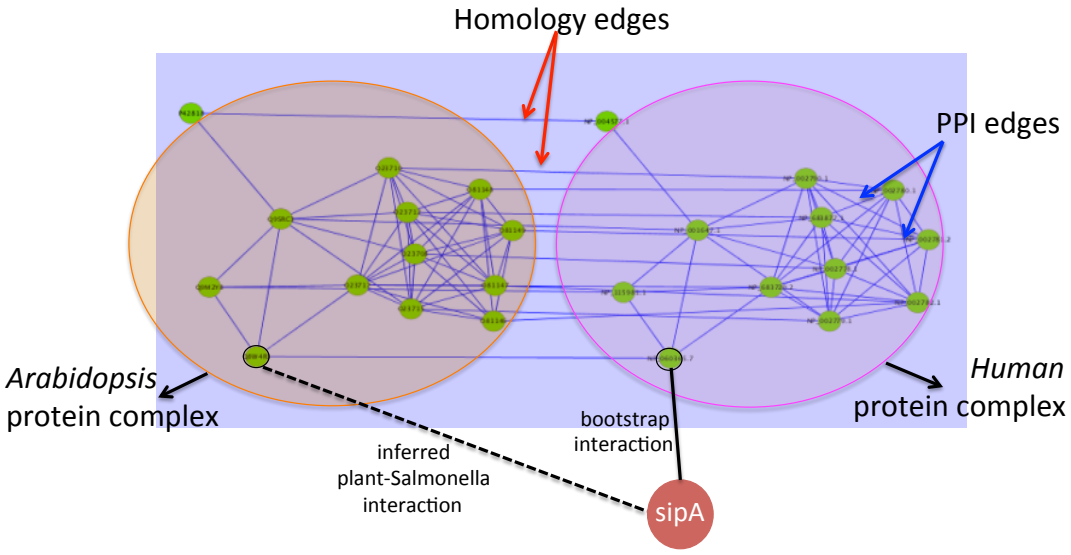


Figure 5.3: Approach-1(b) Graph based interaction transfer. The big circles show the two protein complexes found to be enriched by Network Blast : the *Arabidopsis* protein complex on the left, and the human protein complex on the right. The edges within a protein complex are the PPIs within the host organism. The edges connecting the two protein complexes (i.e the two circles) are the homology edges. The solid line connecting *sipA* with a human protein node is a bootstrap interaction. We use this to infer the new plant-*Salmonella* interaction indicated by the dotted line.

proteins that are the most likely targets for the different *Salmonella* proteins as shown in the Figure 5.3.

For each PPI between a human protein from an enriched protein complex, we infer an equivalent PPI between the corresponding plant protein and the *Salmonella* protein (example, *sipA* in the figure). This filtering procedure gives us a final of 23,664 plant-*Salmonella* PPIs. The biological relevance for using the enriched graph components lies in the premise that clusters of similarly interacting proteins across the two organisms will represent biological processes that have been conserved in the two organisms. Hence, the proteins in these components are also likely to be conserved as pathogen targets.

Approach-2: Transductive Learning

This method considers the target task i.e plant proteins while building a model. It provides a way of incorporating the target task information during model construction. Conventional inductive learning approaches such as the Support Vector Machine classifier use only the training examples to build a model. Transductive learning approaches also use the distribution of the unlabeled test examples. They jointly learn the labels on the test examples while minimizing the error on the labeled training examples. This often results in a good performance, as the classifier has additional information about the unseen test data. In our work here, we use transductive learning for transfer learning in particular the Transductive Support Vector Machine

algorithm (T-SVM) [Joachims, 1999]. The training examples are the source task examples, i.e human-*Salmonella* protein interactions. We use the target task examples as the test data.

Training negatives: Since there are 62 known PPIs in the source task, we sample a set of random 6200 human-*Salmonella* protein pairs to maintain the positive:negative class ratio at 1:100.

Figure 5.4 depicts this setting. This method thus builds a model by using data from both hosts. The optimization function of T-SVM jointly minimizes the training error on the known human-pathogen interactions and the label assignments on the unknown plant-pathogen interactions. The set of target examples can not be used entirely as it is very large and makes the T-SVM algorithm very computationally expensive. Hence we randomly sample 1 percent of the target dataset. For the T-SVM based algorithm to be effective, the kernel function that is used to compute the similarity between examples matters a lot. We use a homology-based kernel function that incorporates the BLAST similarity score between the proteins. Let x_s^i be the feature-vector representing a source task example: the protein pair $< s_s, h_s >$ where s_s is the *Salmonella* protein (i.e the pathogen protein) and h_s is the host protein. Let the target task example be the protein pair $< s_t, a_t >$ where a_t is the *Arabidopsis* protein; and the corresponding feature vector be x_t^k. The kernel function k that computes the similarity between the given two pairs of proteins (i.e their feature vectors) is defined as shown below.

$$k(x_s^i, x_t^k) = \text{sim}(p_s, p_t) + \text{sim}(h_s, a_t) \tag{5.1}$$

$$k(x_s^i, x_s^j) = dot(x_s^i, x_s^j) \tag{5.2}$$

$$k(x_t^i, x_t^j) = dot(x_t^i, x_t^j) \tag{5.3}$$

$$\text{where sim}(m, n) = \text{normalized-BLAST-score}(m, n) \tag{5.4}$$

Equation (5.1) is used in the case where the two protein pairs come from different tasks. We use homology-distance between the pathogen proteins and the host proteins to compute the kernel. The homology distance itself is simply the BLAST protein sequence alignment score. Equation (5.2) and Equation (5.3) show the computation when the examples both come from the same task. Here we simply take the dot product of the two feature vectors. This kernel is symmetric. The similarity between two sequences $sim(m, n)$ is computed using the bit-score from BLAST sequence alignment, normalized using the sequence length of the larger protein. We used the SVM^{light} package [Joachims, 2008] and incorporated our kernel function into it. The parameter tuning for T-SVM (the regularization parameter C) was done using cross validation on the PPIs where we have the true labels. We found $C = 0.1$ was the best setting. This best model is subsequently used to generate predictions on all *Arabidopsis-Salmonella* protein-pairs. The model outputs a score indicating the distance from the classifier hyperplane. A positive score indicates that the protein-pair is on the positive side of the hyperplane and hence closer to the known interacting protein-pairs. All such protein-pairs will be considered as potential interactions predicted by this model.

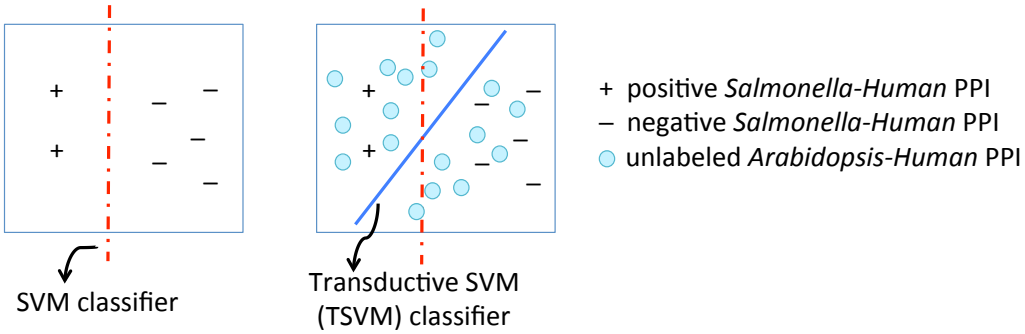


Figure 5.4: Transductive Support Vector Machine (SVM) for transfer learning. The first panel shows the conventional SVM classifier. The second panel shows T-SVM with circles representing unlabeled examples. We use examples from the target task i.e *Arabidopsis-Salmonella* protein pairs as the unlabeled examples to influence the classifier boundary.

Approach-3: Kernel Mean Matching

Our transfer learning scenario here consist of the following setting: multiple 'source' tasks with small amounts of labeled data, a single 'target' task with no labeled data. The first challenge is to pick the best instances from the source tasks, such that the resultant model when applied on the target task generates high confidence predictions. Towards this, we use the instance reweighting technique Kernel Mean Matching (KMM). The reweighted source task instances are used to build a kernelized support vector machine (SVM) model, which is applied on the target task data to get the predicted PPIs. This brings forth the second challenge - selecting appropriate hyperparameters while building a model for a task with no labeled data. For simplicity we also use the same set of features across all tasks (protein sequence features). However the data distribution will be different across tasks due to the different organisms involved.

This approach is based on instance-transfer where the goal is to pick from each of the source tasks, the most relevant instances w.r.t the target task. We use a two-step process: (1) the first step does the instance weighting on the source tasks. (2) the second step uses the reweighted instance to build several SVM classifier models – one model for each hyper-parameter setting. To deal with the second challenge, we present two heuristic methods to select the best set of hyperparameters.

Step-1: Instance reweighting

The similarity between the source and target data can be expressed using the similarity in their distributions $\mathbf{P}_S(x, y)$ and $\mathbf{P}_t(x, y)$. Here $\mathbf{P}_S$ represents the joint distribution of all source tasks. Since we do not have access to the labels y on the target, we make a simplifying assumption that there is only a covariate shift between the source and target tasks - i.e the conditional distribution $P(y|x)$ is the same for both tasks. Mathematically, $\frac{\mathbf{P}_S(x,y)}{\mathbf{P}_t(x,y)} = \frac{\mathbf{P}_S(x)}{\mathbf{P}_t(x)} = r(x)$. Many methods have been proposed for estimating the ratio r. Sugiyama et al. [2008] proposed an algorithm Kullback-Leibler Importance Estimation Procedure (KLIEP) to estimate r directly without estimating the densities of the two distributions.

We use the nonparametric Kernel Mean Matching (KMM) [Huang et al., 2007], which was originally developed to handle the problem of covariate shift between the training and test data distributions. KMM reweighs the training data instances such that the means of the training and test data distributions are close in a reproducing kernel Hilbert space (RKHS). This approach does not require distribution estimation. Let $x_i^S \sim P_S$ and n_S be the number of source instances from all source tasks. Let $x_i^t \sim P_t$ and n_t be the number of target instances. Let β_i represent the "importance" of the source instances. KMM uses a function based on the *maximum mean discrepancy* statistic (MMD). In the form written below, it minimizes the difference between the empirical means of the joint source and target distributions.

$$\min_{\beta} \left\| \frac{1}{n_S} \sum_{i=1}^{n_S} \beta_i \Phi(x_i^S) - \frac{1}{n_t} \sum_{j=1}^{n_t} \Phi(x_j^t) \right\|^2 \tag{5.5}$$

$$\Leftrightarrow \quad \min_{\beta} \ \frac{1}{n_S^2} \beta^T K \beta - \frac{2}{n_S^2} \kappa^T \beta \ + C \tag{5.6}$$

$$\text{subject to } \beta_i \in [0, B] \text{ and } \sum_i \beta_i \leq n_S \tag{5.7}$$

$$\text{where } K_{i,j} = k(x_i^S, x_j^S) \text{ and } \kappa_i = \frac{n_S}{n_t} \sum_{j=1}^{n_t} k(x_i^S, x_j^t) \tag{5.8}$$

K is the kernel matrix over all the source examples. The function (1) is a quadratic program and can be efficiently solved using sequential minimal optimization (SMO), projected gradient based methods. We use the KMM implementation from the Shogun [Sonnenburg et al., 2010] package.

Selecting an appropriate set of source and target instances:
Using all instances in the optimization problem in equation (1) is infeasible for two reasons. The optimization involves the computation of the gram matrix K of $O(n^2)$ where n is the number of instances. Typically the total number of protein-protein pairs between a host-pathogen are of the order of 100 million. Secondly, the total number of labeled source instances is quite small ($\approx$ 1500). This set is likely to get underweighted (i.e $\beta_i \approx 0$) if there are too many unlabeled source instances. To represent the source's empirical mean, in addition to the labeled instances we randomly sample four times as many unlabeled instances. For the target, we randomly sampled n_S instances.

Step-2: Model learning

Once we have the optimal set of source instances, we can train a Kernel-SVM model using these. Along with the first step, we thus call this two step process KMM-SVM. We pick a kernel-based learning algorithm since we plan to extend our work to deal with different feature spaces across the tasks. In such a scenario, the only mechanism to operate on the target data is via similarities, i.e the kernel. The dual formulation for the weighted version of SVM solves the following problem, where the weights β_i were obtained in Step-1.

$$\min_{\alpha} \sum_{i=1}^{n_S} \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j K(x_i^S, x_j^S) \tag{5.9}$$

$$\text{subject to} \sum_i \alpha_i y_i = 0 \quad \text{and} \quad \beta_i C \geq \alpha_i \geq 0 \tag{5.10}$$

Model selection

Parameter tuning and selecting the best model in the absence of labeled data is a very hard problem. The model built on the source data cannot be tuned using cross validation on the source data because doing so will optimize it for the source distribution. Hence we developed two heuristic approaches to select the best hyperparameters. The first one uses the expected class-skew on the target task while the second uses reweighted cross-validation.

Class-skew based parameter selection:

We first built several models by doing a grid-search on the classifier hyper-parameters. There are 3 parameters to tune for the Kernel-SVM: the kernel width γ, the cost parameter C, the weight parameter for the positive class w_+. The total number of parameter combinations in our grid-search were 50. We thus had 50 models trained on the reweighted source data obtained after KMM in Step-1 (Section 5.2). We applied each model on the target data and computed the predicted class-skew r_{pred} using the predicted class labels. The expected class skew based on our understanding of the PPI experimental literature is roughly 1:100 (= r_{true}). We ranked all 50 models on the statistic $|r_{pred} - r_{true}|$. The top k models were selected based on this criteria and a weighted voting ensemble was built using them. This ensemble was used to get the final class label on the target data. We used k=5.

Aggregating the models and assigning interaction scores:

In our experiments, we used k=5 to pick the best models w.r.t the ranking statistic described above. Note that each model gives us a classifier score for every protein-pair in the test data, which can be considered to be the probability of interaction. For k=5, we have five scores for each test protein-pair. These scores were aggregated using two criteria:

(a) the majority vote over the five models where each model votes 'yes' if the output probability score is greater than or equal to 0.5

(b) the averaged of all five probability scores

Spectrum RBF kernel

We used a variant of the spectrum kernel, based on the features used by Dyer et al. [2011] for HIV-human PPI prediction. The kernel uses the n-mers of a given input sequence and is defined as: $k_{sp}^n(x, x') = \exp\{-\frac{\|\phi_{sp}^n(x) - \phi_{sp}^n(x')\|^2}{\sigma^2}\}$, where x, x' are two sequences over an alphabet Σ. Instead of using the 20 amino acids as the alphabet Σ, we used a classification of the amino-acids. There are seven classes based on the electrostatic and hydrophobic properties of proteins, i.e $|\Sigma|$=7. Here ϕ_{sp}^n transforms a sequence s into a $|\Sigma|^n$-dimensional feature-space. One dimension of ϕ_{sp}^n corresponds to the normalized frequency of one of the 7^n possible strings in s. We use n=2,3,4,5.

Features

The features used in each approach are shown in Table 5.1 and discussed in detail in §2.3.

5.3 Negative examples

The PPI datasets from the publicly available databases give us only positive examples. We construct a set of negatives, using a heuristic often used in the PPI prediction literature. Please refer to §2.4 for background and a detailed discussion on negative examples and the approach we take to generate them. Note that the rest of this section assumes that you have perused §2.4.

The homology-based transfer method does not directly use any negative examples/ interaction ratios. In the case of T-SVM, while training the transductive model, we use negative examples from the source task. In the case of KMM-SVM, the data used to build the model comes from the source tasks, where negative examples from each source task are used. Next, during the model selection phase we pick the best models based on the interaction ratio of the model over the predictions on the target task (See Section 5.2 for details). No explicit negative examples are used in this part; the interaction ratio is simply used to pick the best model.

We initially chose a positive:negative class ratio of 1:100 meaning that we expect 1 in every 100 random bacteria-human protein pairs to interact with each other. This has been a common practice in host-pathogen PPI prediction in the past [Dyer et al., 2007, Tastan et al., 2009, Dyer et al., 2011]. Recently published work [Mukhtar et al., 2011b] involving a yeast-2-hybrid study on plant-bacterial PPIs suggests a higher interaction ratio of around 1:1000. Our choice of 1:100 as the class-skew is an overestimate when considering interactions with all *Salmonella* genes, but if we restrict the binding partners to only the so-called *Salmonella* effector proteins, the ratio we use is reasonable. (There are ≈85 known *Salmonella* effector genes). Further, a ratio of 1:1000 makes it very slow to train the Kernel-SVM and the Transductive SVM models. Nonetheless, we also calculated the predictions for a higher skew of 1:500. The results are described in §5.4.

Code: The executable files from the packages used to build our methods, and the scripts that we used to run these can be downloaded here: `http://www.cs.cmu.edu/~mkshirsa/data/frontiers2014/code.zip`

5.4 Results and Discussion

A quantitative evaluation on the target task i.e plant-*Salmonella* is currently not feasible as there is no known PPI data. Hence for the purpose of evaluation, we used some of the PPI datasets as 'sources' for building a model and one as the 'target'. We evaluate the machine-learning based methods in two settings of transfer: *pathogen-level transfer*, where the host is fixed to be human and the pathogen is one of various bacterial species. The second setting *host-level transfer*, is more relevant and refers to the case where the pathogen is fixed to be *Salmonella* and we modify the host species. Since there are few known PPIs involving *Salmonella*, we are only able to experiment with mouse as an alternate host. There are 14 known mouse-*Salmonella* PPIs. Interestingly they involve mouse proteins whose human homologs also interact with the

same *Salmonella* proteins - i.e these 14 PPIs have interologs in the human-*Salmonella* dataset.

The source tasks (i.e training data) and target task (i.e test datasets) are shown in the Table 5.2. Parameters for all methods are tuned using a class-skew based model selection similar to the one described in Section 5.2 for the KMM-SVM method. We compare the following machine-learning based methods:

- Inductive Kernel-SVM (Baseline): This model assumes that the source and target distributions are identical. All source data is pooled together and used to build a single model. For the kernel we used the RBF-spectrum kernel
- Transductive SVM (T-SVM): This is the method described in §5.2
- KMM-SVM: This method is discussed in §5.2

HOST-LEVEL TRANSFER

Source task(s) (training data)	**Target task** (test data)	**Method**	**P**†	**R**†	**F1**†
Salmonella-human	*Salmonella*-mouse	Baseline	42.8	**93.7**	58.8
		T-SVM	45.4	**93.7**	61.2
		KMM-SVM	**51.7**	**93.7**	**66.7**
Salmonella-mouse	*Salmonella*-human	Baseline	95.4	33.8	50.0
		T-SVM	67.5	43.5	**52.9**
		KMM-SVM	**100.0**	**35.5**	52.0

PATHOGEN-LEVEL TRANSFER

Source task(s) (training data)	**Target task** (test data)	**Method**	**P**†	**R**†	**F1**†
Francisella-human, *E.coli*-human	*Salmonella*-human	Baseline	17.8	12.9	14.9
		T-SVM	15.0	14.5	14.7
		KMM-SVM	**25.7**	**16.1**	**19.9**
Francisella-human, *Salmonella*-human	*E.coli*-human	Baseline	12.9	12.5	12.7
		T-SVM	10.4	15.6	12.5
		KMM-SVM	**15.9**	**21.9**	**18.4**

†- computed using the default classifier threshold: 0.5
The positive:negative class ratio in all datasets was 1:100
The performance of a random classifier would be F-score=1

Table 5.2: Performance of the machine learning based methods on various transfer settings. We compare them with a simple baseline: inductive kernel-SVM. We report precision (P), recall (R) and f-score (F1). The data that was used to build each of the models is shown in the first column. The second column shows the target task – the data on which we evaluate the model. The numbers in bold font indicate the highest performance in that column (i.e for that metric).

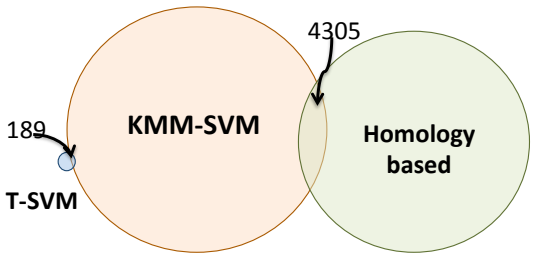
The host-level transfer performance is shown in the first two rows of Table 5.2. The KMM-SVM based method performs much better while transferring from *Salmonella*-human to *Salmonella*-mouse. The recall is very high at 93.7 since the mouse-pathogen PPIs are interologs of the human-pathogen PPIs. The precision is not as high as some additional positives are predicted and we found that they had a high classifier score. These 'false positives' are likely to be true interactions. For the reverse setting, T-SVM does slightly better than the KMM-SVM and 2 points higher than the baseline. Note that here, the source data is very small in size with only 14 PPIs. In the pathogen-level transfer, on the *Salmonella*-human target task, the F1 of the KMM-SVM method is the highest at 19.9 and is 5 points better than the other two methods. On the *E.coli*-human task, the performance is 18.4 which is 5.7 points better than the other methods.

A very interesting observation to make from the table is, the performance on the target task: *Salmonella*-human in the two settings. In the host-level transfer, the F1 is 52 whereas in the pathogen-level transfer it is much lower at 19.9. The hosts human and mouse are much more similar than the group of bacterial species namely: *Salmonella*, *E.* coli and *F.* tularensis. The source tasks are indeed very critical in determining the performance on the target.

Analysis

We apply the models trained using the procedures from previous sections on *Arabidopsis-Salmonella* protein-pairs to get predictions for potential interactions. The homology based approach does not assign any confidence scores to the predictions while both T-SVM and KMM-SVM allow us to obtain a score for every predicted interaction. All predictions from T-SVM with a positive score (>0) are considered to be interacting. For the KMM-SVM method, we filter the predictions using a threshold of 0.7 on the averaged probability-score. (See Section 5.2 for details on the probability score computation for the KMM-SVM method). We chose this threshold of 0.7 since all positives in our training data are assigned a score $\geq$0.7 by the classifier model. The full lists of predicted interactions from all three approaches are available at the following link: `http://www.cs.cmu.edu/~mkshirsa/data/frontiers2014/predictions.zip`.

The total number of PPI predictions based on the score thresholds described above are: 106,807 for homology-based, 1088 for T-SVM and 163,644 from KMM-SVM. Hundreds of thousands of interacting pairs may not be likely and we therefore expect that many of the predictions are likely to be false positives (FPs). We would like to emphasize that, by ranking the predictions on the classifier scores and picking only the top few we are likely to filter out most of the false positives, since the machine learning models are expected to score FPs lower than the true positives. The threshold of 0.7 for KMM-SVM was chosen just to ensure consistency with the threshold that we observed in the training data (i.e in the known interactions). If one considers say the top 10% of the predictions from the KMM-SVM method, we have 1636 PPIs over $\approx$1300 unique *Arabidopsis* proteins and 5 *Salmonella* proteins. Choosing by thresholding the prediction score is one way to select potential interactions for further scrutiny. Another approach is to analyze the predictions based on the biological functions one is interested in. To demonstrate the type of biological functions that are represented in the predictions, we performed GO term enrichment analysis of the *Arabidopsis* proteins involved in the predictions. We can then look at *Arabidopsis* genes with the most



Approach	Homology	T-SVM	KMM-SVM
# predicted interactions	106,807	1088	163,644
# PPIs involving effectors	72461	718	163,397
# unique arabidopsis genes	1107	92	25124
# unique *Salmonella* genes	221	34	31

Figure 5.5: Overlap amongst the novel PPI predictions from each approach. All predictions from the homology based approach and the T-SVM are shown. For the KMM-SVM method, we filter the predictions using a threshold of 0.7 on the interaction probability reported by the classifier. We picked this threshold based on the interaction probabilities reported on the known interactions.

enriched GO terms and what their predicted *Salmonella* partners are.

A Venn diagram depicting the overlap between the predicted pairs of proteins interacting according to the three approaches is shown in Figure 5.5. The PPIs reported by each approach are quite different from the others. Only 189 are shared between T-SVM and KMM-SVM and 4305 between the homology approach and KMM-SVM. No overlap was found between the homology approach and the T-SVM approaches. These relatively small overlaps are due to the different input sources (tasks) used by each approach. Further, the machine-learning based approaches KMM-SVM and T-SVM use a discriminative model which employs negative examples whereas the heuristics based approach does not use any such negative data and hence has a small overlap with the other two. The two machine-learning based approaches differ due to the use of different kernels. The KMM-SVM approach is the only approach that shows overlap in predictions to both, the heuristics and the T-SVM approaches.

Because the ratio of 1 positive to 100 negative pairs likely overestimates the number of interactions, we next changed this ratio to 1:500 and generated a new model. As expected, a much smaller number of pairs are predicted namely, 6035. This is a more manageable list and the predictions of the new model are provided at `http://www.cs.cmu.edu/~mkshirsa/data/frontiers2014/predictions_class_skew_500.txt`.

Qualitative analysis of predicted interactions

As with any predictions, experimental validation is ultimately needed to verify them. The choice depends on the interest of the experimentalist. Here we have chosen for discussion a few predictions that are interesting to us, but we encourage the reader to look at the list of predictions for others of potential biological interest.

We calculated Gene Ontology (GO) enrichment in the *Arabidopsis* proteins predicted to be targeted by the *Salmonella* proteins. We are interested in analyzing the characteristics of the plant proteins predicted to be the most popular targets for pathogenesis. We defined the 'popular targets' using the following criteria: (a) the *Arabidopsis* protein is predicted to be targeted by at least 3 *Salmonella* effectors with a proba-

bility greater than 0.9 and (b) the GO term annotations of the *Arabidopsis* protein are significantly enriched (with a p-value of <0.001 as obtained by GO enrichment analysis using FuncAssociate [Berriz et al., 2003]). There are a total of 5247 *Arabidopsis* proteins satisfying these criteria. In Table 5.5, we show 20 *Arabidopsis* genes selected randomly from this set of highly targeted *Arabidopsis* proteins. In Table 5.3, we show the list of all enriched GO terms.

For each gene we show the description and the enriched GO annotations. Among the presented *Arabidopsis* proteins, nearly one third are transcription factors. These function e.g. in hormone-mediated signalling pathways. It has been reported that jasmonic acid and ethylene signaling pathways are involved in plant defense response against *Salmonella* [Schikora et al., 2008]. Other examples that highlight the role of transcription factors in plant-pathogen interaction are e.g. that a *Xanthomonas* effector protein targets an ethylene responsive transcription factor (ERF) in tomato to inhibit ethylene induced transcription [Kim et al., 2013] and systemic immunity in barley induced by *Xanthomonas* and *Pseudomonas* bacteria may involve WRKY and ERF-like transcription factors [Dey et al., 2014]. Further, actin-11 and actin-related proteins involved in actin polymerization and depolymerization are obtained. It is well known that *Salmonella* translocates effectors into the mammalian host cell in order to interact with actin and e.g. modify the cell cytoskeleton to allow bacterial entry (for review see Schleker et al. [2012]). Our analysis revealed growth regulating factor 1 (GRF1)-interacting factor 2, a transcriptional co-activator which is part of a regulatory complex with GRF1 and microRNA (miRNA) 396. MiRNAs are involved in plant disease resistance to bacteria and miRNA396 has been shown to be upregulated in plants upon flg22 treatment [Li et al., 2010]. Liu et al. [2014] reported that putative GRF1 targets in *Arabidopsis* are heavily involved in biosynthetic and metabolic pathways, e.g. phenylpropanoid, amino acids and lignin biosynthesis as well as plant hormone signal transduction indicating the role of GRF1 in plant defense mechanisms. Other examples of predicted interactions and more details of their possible relevance in *Salmonella*-plant interplay are discussed in the our other work [Schleker et al., 2015].

Conclusions and Future work

In this section, we addressed the challenge of predicting the *Salmonella-Arabidopsis* interactome in the absence of any experimentally known interactions. Previous work in this area was based purely on homology between human and *Arabidopsis* proteins and was therefore limited to proteins that do display sequence similarity. Due to the large divergence between the two organisms, this approach neglects a large fraction of potential *Arabidopsis* targets. We therefore presented here three different sophisticated computational and machine learning methods to predict hereto unknown *Salmonella*-plant interactions from a relatively small list of known *Salmonella*-human interactions. This is a very challenging task because it is not possible to quantitatively validate the predictions. Nonetheless, the predictions provide a gold-mine for discovery because they provide experimentally testable hypotheses on the communication mechanisms between plant and *Salmonella* without restriction to known effectors in the pathogen or sequences of similarity to those observed in better studied eukaryotic organisms. With these advantages comes a set of limitations to be aware of.

Since machine learning methods need some known interactions to evaluate the

models on, and to pick the best set of predictions, their application in this current context has limitations. For example, we can obtain different predictions from our methods by varying the parameters, especially the class skew (we studied the ratios 1:100 and 1:500). Because there are currently no known *Salmonella*-plant interactions, we are not able to quantify which of these sets of predictions is more reliable. Augmenting the predictions with some other biological information from the target task can help in picking the most plausible PPIs. This is a direction for future research. Further,

1. The interactome predicted by each method is not the true interactome, but is a set of predictions. There will be false positive and false negative interactions. Thus, each individual prediction has to be considered a hypothesis not a fact.

2. In line with point 1 above, the size of the predicted interactomes does not necessarily relate to the true interactome. We dont know how many interactions to expect. Our different predictions vary greatly in size, with one method predicting only one thousand interactions, while others predict more than 100,000 interactions. While it is more likely that smaller numbers of interactions are more likely, it does not mean that this method is inherently better than the other methods.

3. The size of the predicted interactions list also depends on a critical parameter, the positive to negative class ratio. This parameter is important but it is tuneable, so the methods validity is not dependent on its choice. However, it is important to appreciate that the predictions will differ greatly when this parameter is changed. Thus, biological insight in choosing predictions to validate still needs to be applied, regardless of the prior choice of ratio in generating the model.

These general limitations in the context of the specific results of the models presented here translate to the following issues: The data presented for the KMM-SVM model indicate that 163,644 PPIs are predicted (Figure 5.5). This is of the same order of magnitude as the number of false positives that would be predicted, given the reported false positive rate of the method that indicate ≈180,000 false positive PPIs would be expected. This raises the possibility that the bulk of the predictions may be false positives. The data presented for the KMM-SVM model also indicates that 25,124 distinct *Arabidopsis* genes participate in PPIs with 31 distinct *Salmonella* genes (Figure 5.5). This implies that 91% of the *Arabidopsis* protein-coding gene complement (TAIR10: 27,416 genes - `http://www.arabidopsis.org/portals/genAnnotation/gene_structural_annotation/annotation_data.jsp`) enters into productive interaction with only 31 *Salmonella* proteins. It also implies that, on average, each interacting *Salmonella* protein is capable of productive interaction with over 5,000 *Arabidopsis* proteins. It is unlikely that this is the case, again suggesting that a large number of false positives have to be expected.

GO term	Description
GO:0003676	nucleic acid binding
GO:0003677	DNA binding
GO:0003700	sequence-specific DNA binding TF activity
GO:0003723	RNA binding
GO:0003735	structural constituent of ribosome
GO:0003755	peptidyl-prolyl cis-trans isomerase activity
GO:0003779	actin binding
GO:0003899	DNA-directed RNA polymerase activity
GO:0004298	threonine-type endopeptidase activity
GO:0004693	cyclin-dependent protein serine/threonine kinase activity
GO:0004842	ubiquitin-protein transferase activity
GO:0004871	signal transducer activity
GO:0005484	SNAP receptor activity
GO:0005507	copper ion binding
GO:0005509	calcium ion binding
GO:0005515	protein binding
GO:0005525	GTP binding
GO:0005576	extracellular region
GO:0005622	intracellular region
GO:0005634	nuclear envelope
GO:0005839	proteasome core complex
GO:0005840	ribosome
GO:0006351	transcription, DNA-templated
GO:0006355	regulation of transcription, DNA-templated
GO:0006412	translation
GO:0006413	translational initiation
GO:0006457	protein folding
GO:0006511	ubiquitin-dependent protein catabolic process
GO:0007264	small GTPase mediated signal transduction
GO:0007267	cell-cell signaling
GO:0008270	zinc ion binding
GO:0008794	arsenate reductase (glutaredoxin) activity
GO:0009408	response to heat
GO:0009409	response to cold
GO:0009414	response to water deprivation
GO:0009570	chloroplast stroma
GO:0009579	thylakoid
GO:0009651	response to salt stress
GO:0009733	response to auxin
GO:0008233	peptidase activity

Table 5.3: List of all enriched GO terms obtained by applying enrichment analysis tool FuncAssociate (Berriz et al. [2003]) on the set of highly targeted *Arabidopsis* proteins (i.e *Arabidopsis* proteins predicted to interact with at least 3 *Salmonella* effectors). The shown terms had a p-value less than 0.001.

GO term	Description
GO:0009737	response to abscisic acid
GO:0009739	response to gibberellin
GO:0009751	response to salicylic acid
GO:0009753	response to jasmonic acid
GO:0009828	plant-type cell wall loosening
GO:0009873	ethylene mediated signaling pathway
GO:0010200	response to chitin
GO:0015031	protein transport
GO:0015035	protein disulfide oxidoreductase activity
GO:0016491	oxidoreductase activity
GO:0016607	nuclear speck
GO:0016762	xyloglucan:xyloglucosyl transferase activity
GO:0022626	cytosolic ribosome
GO:0022627	cytosolic small ribosomal subunit
GO:0042254	ribosome biogenesis
GO:0042742	defense response to bacterium
GO:0043565	sequence-specific DNA binding
GO:0045454	cell redox homeostasis
GO:0045892	negative regulation of transcription, DNA-templated
GO:0045893	positive regulation of transcription, DNA-templated
GO:0046686	response to cadmium ion
GO:0046872	metal ion binding
GO:0051726	regulation of cell cycle

Table 5.4: Table 5.3 continued from above ...

Arabidopsis (TAIR id)	Protein name/gene	Enriched Gene Ontology annotations	Corresp. GO terms
AT1G01030	B3 domain containing transcription factor	sequence-specific DNA binding transcription factor activity ; regulation of transcription, DNA-templated	GO:0003700 GO:0006355
AT1G06160	Ethylene-responsive transcription factor ERF094	DNA binding ; sequence-specific DNA binding transcription factor activity ; regulation of transcription from RNA-polymerase II promoter ; response to jasmonic acid stimulus	GO:0003677 GO:0003700 GO:0006355 GO:0009753
AT1G01060	Myb-related putative transcription factor	response to cadmium ion ; response to salt stress ; response to auxin stimulus ; response to cold	GO:0046686 GO:0009651 GO:0009733 GO:0009409
AT1G13180	Actin-related protein 3	actin binding	GO:0003779
AT2G40220	Ethylene-responsive transcription factor ABI4. Protein glucose insensitive 6	DNA binding ; response to water deprivation ; positive regulation of transcription, DNA-dependent ; sequence-specific DNA binding	GO:0003677 GO:0009414 GO:0045893 GO:0043565
AT2G46400	Putative WRKY transcription factor 46	response to chitin	GO:0010200
AT1G01080	Ribonucleoprotein, putative	nucleic acid binding ; RNA binding	GO:0003676 GO:0003723
AT3G12110	Actin-11	chloroplast stroma	GO:0009570
AT3G56400	Probable WRKY transcription factor 70	response to salicylic acid stimulus ; sequence-specific DNA binding transcription factor activity ; protein amino acid binding	GO:0009751 GO:0003700 GO:0005515
AT1G01090	Pyruvate dehydrogenase E1 component subunit alpha-3, chloroplastic	chloroplast stroma	GO:0009570
AT4G09570	Ca-dependent protein kinase 4	protein amino acid binding	GO:0005515
AT1G01150	Homeodomain-like protein with RING-type zinc finger domain	zinc ion binding ; regulation of transcription, DNA-templated	GO:0008270 GO:0006355
AT4G18170	Probable WRKY transcription factor 28	regulation of transcription, DNA-templated ; sequence- specific DNA binding transcription factor activity	GO:0006355 GO:0003700
AT1G01160	GRF1-interacting factor 2	protein amino acid binding	GO:0005515
AT1G01200	Ras-related protein RABA3	GTP binding; small GTPase mediated signal transduction ; protein transport	GO:0005525 GO:0007264 GO:0015031
AT5G47220	Ethylene-responsive transcription factor 2	positive regulation of transcription, DNA-dependent ; ethylene mediated signaling pathway	GO:0045893 GO:0009873
AT1G01250	Ethylene-responsive TF ERF023	sequence-specific DNA binding transcription factor activity ; nuclear envelope	GO:0003700 GO:0005634
AT1G01350	Zinc finger CCCH domain-containing protein 1	nucleic acid binding ; zinc ion binding	GO:0003676 GO:0008270
AT1G01370	Histone H3-like centromeric protein HTR12	DNA binding ; protein amino acid binding	GO:0003677 GO:0005515

Table 5.5: GO terms that were enriched in the most targetted *Arabidopsis* proteins in our predictions. To get this list, we performed a GO enrichment analysis using the FuncAssociate [Berriz et al., 2003]. We then procure the set of *Arabidopsis* genes which correspond to the enriched GO terms; i.e GO terms with a p-value of < 0.001. We further filter this set to include only those *Arabidopsis* genes predicted to interact with at least 3 *Salmonella* effector proteins. In this table, we show around 20 such *Arabidopsis* genes for the lack of space. The remaining are available via the download link.

Chapter 6

Frame-Semantic Role Labeling with Heterogeneous Annotations

Semantic role labeling (SRL) is a task in natural language processing concerned with the computational detection of meaning in sentences. This sentence-level semantic analysis of text characterizes events and relations in the sentence. The *predicate* (typically a verb) establishes "what" took place, and other sentence components express the participants in the event, as well as further event properties. Consider the sentence:

Christopher Nolan directed the movie Batman Begins for Warner Bros in 2005.

The verb direct is the predicate connecting the 'who': 'Christopher Nolan' with the 'what': 'the movie Batman Begins' and 'when': '2005'. Each of these constituents has a *role* indicating their semantic meaning in the context of the sentence and the predicate. 'Christopher Nolan' is the **Artist**, 'the movie Batman Begins' is the **Production**, 'Warner Bros' is the **Studio** and '2005' is the **Time**. The predicate itself is also assigned a label to disambiguate its' sense in the sentence. For instance, direct could also appear in the sense of 'aim' or 'target': Let me direct your attention to this thesis's appendix.

Recently, several corpora have been manually annotated with semantic roles resulting in resources such as FrameNet [Baker et al., 1998, Fillmore and Baker, 2009] [1], PropBank [Palmer et al., 2005], NomBank. These have enabled the development of statistical approaches for SRL and it has become a well-defined task with a substantial body of work and comparative evaluation. The roles and predicates from SRL can be used in many downstream applications such as question answering, summarization of text, translation of sentences, information retrieval and extraction (search engines), dialogue systems etc.

The use of SRL systems in real-world applications has thus far been limited, mainly due to their limited coverage of semantics. The high cost of semantic structure annotation is one of the major obstacles to obtaining a broad coverage. The annotated datasets that exist are often small, hindering the accuracy and domain robustness of models trained on them. However, low-resource tasks may benefit from exploiting *out-of-domain* annotated data, as well as data with *different* (but related) forms of annotation, for additional training data or features.

[1] `http://framenet.icsi.berkeley.edu`

Figure 6.1: Part of a sentence from FrameNet full-text annotation. 3 frames and their arguments are shown: DESIRING is evoked by *want*, ACTIVITY_FINISH by *finish*, and HOLDING_OFF_ON by *hold off*. Thin horizontal lines representing argument spans are labeled with role names. (Not shown: *July* and *August* evoke CALENDRIC_UNIT and fill its **Unit** role.)

In this work [Kshirsagar et al., 2015b], we address the **argument identification** (a form of SRL), which is a subtask of frame-semantic parsing. Given a sentence, frame-semantic parsing methods [Gildea and Jurafsky, 2002, Das et al., 2014] find and map the predicates to the **frames** they evoke, and for each frame, find and label its **argument** phrases with frame-specific **roles**. An example appears in figure 6.1, which will be explained in detail in §6.1. This task is challenging because there are only a few thousand fully annotated sentences for supervised training. Our contribution addresses the paucity of annotated data for training using standard domain adaptation techniques. We exploit three annotation sources:

- the frame-to-frame relations in FrameNet, by using hierarchical features to share statistical strength among related roles (§6.2),
- FrameNet's corpus of partially-annotated **exemplar** sentences, by using "frustratingly easy" domain adaptation (§6.2), and
- a PropBank-style SRL system, by using guide features (§6.2).[2]

These expansions of the *training corpus* and the *feature set* for supervised argument identification are integrated into SEMAFOR [Das et al., 2014], the leading open-source frame-semantic parser for English. We observe a 4% F_1 improvement in argument identification on the FrameNet test set, leading to a 1% F_1 improvement on the full frame-semantic parsing task. Our code and models are available at `http://www.ark.cs.cmu.edu/SEMAFOR/`.

6.1 FrameNet

FrameNet represents events, scenarios, and relationships with an inventory of **frames** (such as SHOPPING and SCARCITY). Each frame is associated with a set of **roles** (or **frame elements**) called to mind in order to understand the scenario, and lexical **predicates** (verbs, nouns, adjectives, and adverbs) capable of evoking the scenario. For example, the BODY_MOVEMENT frame has **Agent** and **Body_part** as its core roles, and

[2]Preliminary experiments training on PropBank annotations mapped to FrameNet via SemLink 1.2.2c [Bonial et al., 2013] hurt performance, likely due to errors and coverage gaps in the mappings.

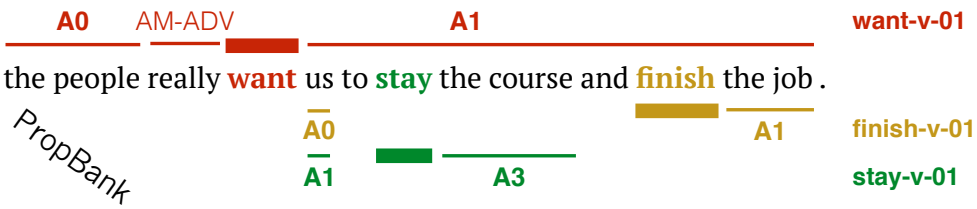


Figure 6.2: A PropBank-annotated sentence from OntoNotes [Hovy et al., 2006]. The PB lexicon defines rolesets (verb sense–specific frames) and their core roles: e.g., `finish-v-01` 'cause to stop', `A0` 'intentional agent', `A1` 'thing finishing', and `A2` 'explicit instrument, thing finished with'. (`finish-v-03`, by contrast, means 'apply a finish, as to wood'.) Clear similarities to the FrameNet annotations in figure 6.1 are evident, though PB uses lexical frames rather than deep frames and makes some different decisions about roles (e.g., `want-v-01` has no analogue to **Focal_participant**).

lexical entries including verbs such as bend, blink, crane, and curtsy, plus the noun use of curtsy. In FrameNet 1.5, there are over 1,000 frames and 12,000 lexical predicates.

Hierarchy

The FrameNet lexicon is organized as a network, with several kinds of **frame-to-frame relations** linking pairs of frames and (subsets of) their arguments [Ruppenhofer et al., 2010]. In this work, we consider two kinds of frame-to-frame relations:

Inheritance: E.g., ROBBERY inherits from COMMITTING_CRIME, which inherits from MISDEED. Crucially, roles in inheriting frames are mapped to corresponding roles in inherited frames: ROBBERY.**Perpetrator** links to COMMITTING_CRIME.**Perpetrator**, which links to MISDEED.**Wrongdoer**, and so forth. Another example is: PUNCTUAL_PERCEPTION (e.g., glimpse.v) inherits from PERCEPTION_EXPERIENCE (e.g., see.v), which inherits from PERCEPTION. Other frames inheriting from PERCEPTION include SENSATION (e.g., sight.n) and BECOMING_AWARE (e.g., notice.v). PUNCTUAL_PERCEPTION.**Perceiver** links to PERCEPTION_EXPERIENCE.**Perceiver_passive**, which links to PERCEPTION.**Perceiver**, which links to SENSATION.**Perceiver_passive** and BECOMING_AWARE.**Cognizer**.

Subframe: This indicates a subevent within a complex event. E.g., the CRIMINAL_PROCESS frame groups together subframes ARREST, ARRAIGNMENT and TRIAL. CRIMINAL_PROCESS.**Defendant**, for instance, is mapped to ARREST.**Suspect**, TRIAL.**Defendant**, and SENTENCING.**Convict**.

We say that a *parent* of a role is one that has either the **Inheritance** or **Subframe** relation to it. There are 4,138 **Inheritance** and 589 **Subframe** links among role types in FrameNet 1.5.

Related work

Prior work has considered various ways of grouping role labels together in order to share statistical strength. Matsubayashi et al. [2009] observed small gains from using the **Inheritance** relationships and also from grouping by the role name (SEMAFOR already incorporates such features). Johansson [2012] reports improvements in SRL for Swedish, by exploiting relationships between both frames and roles. Baldewein

	Full-Text		Exemplars	
	train	*test*	*train*	*test*
Sentences	2,780	2,420	137,515	4,132
Frames	15,019	4,458	137,515	4,132
Overt arguments	25,918	7,210	278,985	8,417
TYPES				
Frames	642	470	862	562
Roles	2,644	1,420	4,821	1,224
Unseen frames *vs. train:*		46		0
Roles in unseen frames *vs. train:*		178		0
Unseen roles *vs. train:*		289		38
Unseen roles *vs. combined train:*		103		32

Table 6.1: Characteristics of the training and test data. (These statistics exclude the development set, which contains 4,463 frames over 746 sentences.)

et al. [2004] learn latent clusters of roles and role-fillers, reporting mixed results. Our approach is described in §6.2.

Annotations

Statistics for the annotations appear in table 6.1.

Full-text (FT): This portion of the FrameNet corpus consists of documents and has about 5,000 sentences for which annotators assigned frames and arguments to as many words as possible. Beginning with the SemEval-2007 shared task on FrameNet analysis, frame-semantic parsers have been trained and evaluated on the full-text data [Baker et al., 2007, Das et al., 2014].[3] The full-text documents represent a mix of genres, prominently including travel guides and bureaucratic reports about weapons stockpiles.

Exemplars: To document a given predicate, lexicographers manually select corpus examples and annotate them *only with respect to the predicate in question*. These singly-annotated sentences from FrameNet are called lexicographic **exemplars**. There are over 140,000 sentences containing argument annotations and relative to the FT dataset, these contain an order of magnitude more frame annotations and over two orders of magnitude more sentences. As these were manually selected, the rate of overt arguments per frame is noticeably higher than in the FT data. The exemplars formed the basis of early studies of frame-semantic role labeling [e.g., Gildea and Jurafsky, 2002, Thompson et al., 2003, Fleischman et al., 2003, Litkowski, 2004, Kwon et al., 2004]. Exemplars have not yet been exploited successfully to improve role labeling performance on the more realistic FT task.[4]

[3] Though these were *annotated* at the document level, and train/development/test splits are by document, the frame-semantic parsing is currently restricted to the sentence level.

[4] Das and Smith [2011, 2012] investigated semi-supervised techniques using the exemplars and WordNet for frame identification. Hermann et al. [2014] also improve frame identification by mapping frames and predicates into the same continuous vector space, allowing statistical sharing.

PropBank

PropBank [PB; Palmer et al., 2005] is a lexicon and corpus of predicate–argument structures that takes a shallower approach than FrameNet. FrameNet frames cluster lexical predicates that evoke similar kinds of scenarios In comparison, PropBank frames are purely lexical and there are no formal relations between different predicates or their roles. PropBank's sense distinctions are generally coarser-grained than FrameNet's. Moreover, FrameNet lexical entries cover many different parts of speech, while PropBank focuses on verbs and (as of recently) eventive noun and adjective predicates. An example with PB annotations is shown in figure 6.2.

6.2 Model

We use the model from SEMAFOR [Das et al., 2014], detailed in §6.2, as a starting point. We experiment with techniques that augment the model's training data (§6.2) and feature set (§6.2, §6.2).

Baseline

In SEMAFOR, the argument identification task is treated as a structured prediction problem. Let the classification input be a dependency-parsed sentence $\mathbf{x}$, the token(s) p constituting the predicate in question, and the frame f evoked by p (as determined by frame identification). We use the heuristic procedure described by [Das et al., 2014] for extracting candidate argument spans for the predicate; call this $\mathit{spans}(\mathbf{x}, p, f)$. *spans* always includes a special span denoting an empty or non-overt role, denoted $\emptyset$. For each candidate argument $a \in \mathit{spans}(\mathbf{x}, p, f)$ and each role r, a binary feature vector $\boldsymbol{\phi}(a, \mathbf{x}, p, f, r)$ is extracted. We use the feature extractors from [Das et al., 2014] as a baseline, adding additional ones in our experiments (§6.2–§6.2). Each a is given a real-valued score by a linear model:

$$\mathit{score}_{\mathbf{w}}(a \mid \mathbf{x}, p, f, r) = \mathbf{w}^\top \boldsymbol{\phi}(a, \mathbf{x}, p, f, r) \tag{6.1}$$

The model parameters $\mathbf{w}$ are learned from data (§6.3).

Prediction requires choosing a joint assignment of all arguments of a frame, respecting the constraints that a role may be assigned to at most one span, and spans of overt arguments must not overlap. Beam search, with a beam size of 100, is used to find this argmax.[5]

Hierarchy Features

We experiment with features shared between related roles of related frames in order to capture statistical generalizations about the kinds of arguments seen in those roles. Our hypothesis is that this will be beneficial given the small number of training examples for individual roles.

All roles that have a common parent based on the **Inheritance** and **Subframe** relations will share a set of features in common. Specifically, for each base feature ϕ

[5] Recent work has improved upon global decoding techniques [Das et al., 2012, Täckström et al., 2015]. We expect such improvements to be complementary to the gains due to the added features and data reported here.

which is conjoined with the role r in the baseline model ($\phi \wedge$ "role=r"), and for each parent r' of r, we add a new copy of the feature that is the base feature conjoined with the parent role, ($\phi \wedge$ "parent_role=r'"). We experimented with using more than one level of the hierarchy (e.g., grandparents), but the additional levels did not improve performance.

Domain Adaptation and Exemplars

Daumé [2007] proposed a feature augmentation approach that is now widely used in supervised domain adaptation scenarios. We use a variant of this approach. Let $\mathcal{D}_{\text{ex}}$ denote the exemplars training data, and $\mathcal{D}_{\text{ft}}$ denote the full text training data. For every feature $\phi(a, \mathbf{x}, p, f, r)$ in the base model, we add a new feature $\phi_{\text{ft}}(\cdot)$ that fires only if $\phi(\cdot)$ fires and $\mathbf{x} \in \mathcal{D}_{\text{ft}}$. The intuition is that each base feature contributes both a "general" weight and a "domain-specific" weight to the model; thus, it can exhibit a general preference for specific roles, but this general preference can be fine-tuned for the domain. Regularization encourages the model to use the general version over the domain-specific, if possible.

Guide Features

Another approach to domain adaptation is to train a supervised model on a source domain, make predictions using that model on the target domain, then use those predictions as additional features while training a new model on the target domain. The source domain model is effectively a form of preprocessing, and the features from its output are known as **guide features** [Johansson, 2013, Kong et al., 2014].[6]

In our case, the full text data is our target domain, and PropBank and the exemplars data are our source domains, respectively. For PropBank, we run the SRL system of Illinois Curator 1.1.4 [Punyakanok et al., 2008][7] on verbs in the full-text data. For the exemplars, we train baseline SEMAFOR on the exemplars and run it on the full-text data.

We use two types of guide features: one encodes the role label predicted by the source model, and the other indicates that a span a was assigned *some* role. For the exemplars, we use an additional feature to indicate that the predicted role matches the role being filled.

6.3 Learning

Following SEMAFOR, we train using a **local** objective, treating each role and span pair as an independent training instance. We have made two modifications to training which had negligible impact on full-text accuracy, but decreased training time significantly:[8]

[6] This is related to the technique of model stacking, where successively richer models are trained by cross-validation on the same dataset [e.g., Cohen and Carvalho, 2005, Nivre and McDonald, 2008, Martins et al., 2008].

[7] `http://cogcomp.cs.illinois.edu/page/software_view/SRL`

[8] With SEMAFOR's original features and training data, the result of the above changes is that full-text F_1 decreases from 59.3% to 59.1%, while training time (running optimization to convergence) decreases from 729 minutes to 82 minutes.

- We use the online optimization method AdaDelta [Zeiler, 2012] with minibatches, instead of the batch method L-BFGS [Liu and Nocedal, 1989]. We use minibatches of size 4,000 on the full text data, and 40,000 on the exemplar data.
- We minimize squared structured hinge loss instead of a log-linear loss.

  Let $((\mathbf{x}, p, f, r), a)$ be the ith training example. Then the squared hinge loss is given by $L_{\mathbf{w}}(i) =$

$$\left(\max_{a'} \left\{ \begin{array}{c} \mathbf{w}^\top \boldsymbol{\phi}(a', \mathbf{x}, p, f, r) \\ +\mathbf{1}\{a' \neq a\} \end{array} \right\} - \mathbf{w}^\top \boldsymbol{\phi}(a, \mathbf{x}, p, f, r)\right)^2$$

We learn $\mathbf{w}$ by minimizing the ℓ_2-regularized average loss on the dataset:

$$\mathbf{w}^* = \operatorname*{argmin}_{\mathbf{w}} \frac{1}{N} \sum_{i=1}^{N} L_{\mathbf{w}}(i) + \frac{1}{2}\lambda \|\mathbf{w}\|_2^2 \tag{6.2}$$

6.4 Experimental Setup

We use the same FrameNet 1.5 data and train/test splits as Das et al. [2014]. Automatic syntactic dependency parses from MSTParserStacked [Martins et al., 2008] are used, as in Das et al. [2014].

Preprocessing. Out of 145,838 exemplar sentences, we removed 4,191 sentences which had no role annotations (under the assumption that these are likely to be incomplete annotations). We removed sentences that appeared in the full-text data. We also merged spans which were adjacent and had the same role label.

Hyperparameter tuning. We determined the stopping criterion and the ℓ_2 regularization parameter λ by tuning on the FT development set, searching over the following values for λ: 10^{-5}, 10^{-7}, 10^{-9}, 10^{-12}.

Evaluation. A complete frame-semantic parsing system involves frame identification and argument identification. We perform two evaluations: one assuming gold-standard frames are given, to evaluate argument identification alone; and one using the output of the system described by Hermann et al. [2014], the current state-of-the-art in frame identification, to demonstrate that our improvements are retained when incorporated into a full system.

6.5 Results

Argument Identification. We present precision, recall, and F_1-measure microaveraged across the test instances in table 6.2, for all approaches. The evaluation used in Das et al. [2014] assesses both frames and arguments; since our focus is on SRL, we only report performance for arguments, rendering our scores more interpretable. Under our argument-only evaluation, the system of Das et al. [2014] gets 59.3% F_1.

The first block shows baseline performance. The next block shows the benefit of FrameNet hierarchy features (+1.2% F_1). The third block shows that using exemplars

Training Configuration (Features)	Model Size	P (%)	R (%)	F_1 (%)
FT (Baseline)	1.1	65.6	53.8	59.1
FT (Hierarchy)	1.9	67.2	54.8	60.4
Exemplars $\xrightarrow{\text{guide}}$ FT	1.2	65.2	55.9	60.2
FT+Exemplars (Basic)	5.0	66.0	58.2	61.9
FT+Exemplars (DA)	5.8	65.7	59.0	62.2
PB-SRL $\xrightarrow{\text{guide}}$ FT	1.2	65.0	54.8	59.5
Combining the best methods				
PB-SRL $\xrightarrow{\text{guide}}$ FT+Exemplars	5.5	67.4	58.8	62.8
FT+Exemplars (Hierarchy)	9.3	66.0	60.4	**63.1**

Table 6.2: Argument identification results on the full-text test set. Model size is in millions of features.

as training data, especially with domain adaptation, is preferable to using them as guide features (2.8% F_1 vs. 0.9% F_1). PropBank SRL as guide features offers a small (0.4% F_1) gain.

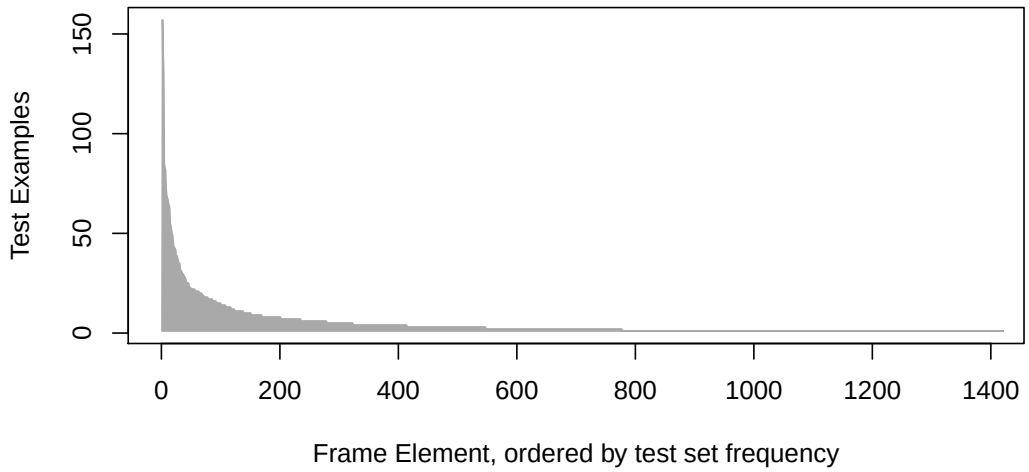


Figure 6.3: Frequency of each role appearing in the test set.

The last two rows of table 6.2 show the performance upon combining the best approaches. Both use full-text and exemplars for training; the first uses PropBank SRL as guide features, and the second adds hierarchy features. The best result is the latter, gaining 3.95% F_1 over the baseline.

Role-level evaluation. Figure 6.4 shows F_1 per frame element, for the baseline and the three best models. Each x-axis value is one role, sorted by decreasing frequency

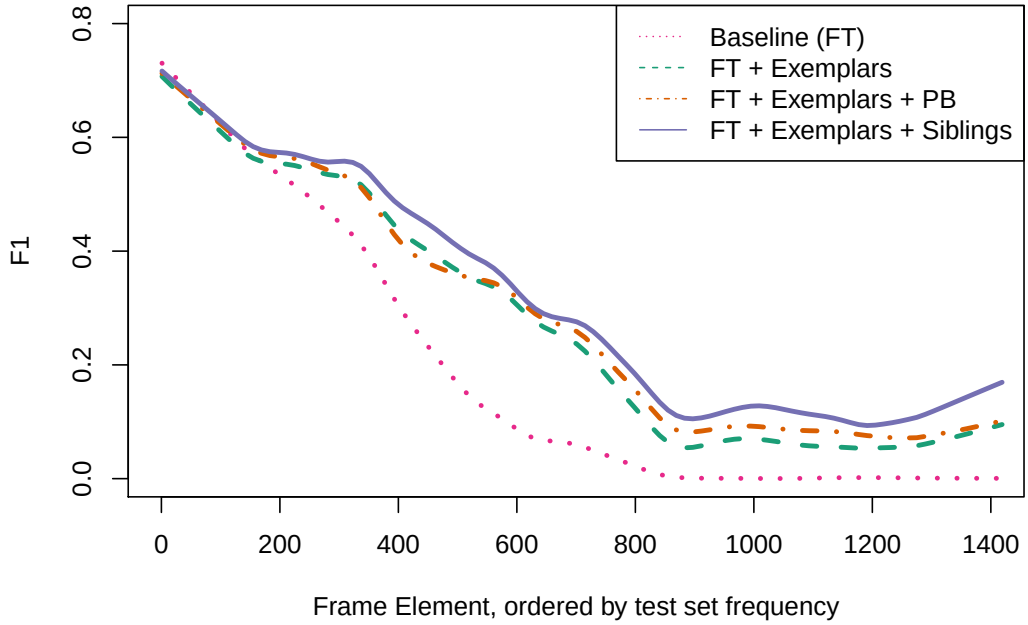


Figure 6.4: F_1 for each role appearing in the test set, ranked by frequency. F_1 values have been smoothed with `loess`, with a smoothing parameter of 0.2. "Siblings" refers to hierarchy features.

(the distribution of role frequencies is shown in figure 6.3). For frequent roles, performance is similar; our models achieve gains on rarer roles.

Full system. When using the frame output of Hermann et al. [2014], F_1 improves by 1.1%, from 66.8% for the baseline, to 67.9% for our combined model (from the last row in table 6.2).

6.6 Conclusion

We have empirically shown that auxiliary semantic resources can benefit the challenging task of frame-semantic role labeling. The significant gains come from the FrameNet exemplars and the FrameNet hierarchy, with some signs that the PropBank scheme can be leveraged as well.

We are optimistic that future improvements to lexical semantic resources, such as crowdsourced lexical expansion of FrameNet [Pavlick et al., 2015] as well as ongoing/planned changes for PropBank [Bonial et al., 2014] and SemLink [Bonial et al., 2013], will lead to further gains in this task. Moreover, the techniques discussed here could be further explored using semi-automatic mappings between lexical resources [such as UBY; Gurevych et al., 2012], and correspondingly, this task could be used to extrinsically validate those mappings.

Ours is not the only study to show benefit from heterogeneous annotations for semantic analysis tasks. Feizabadi and Padó [2015], for example, successfully applied

similar techniques for SRL of *implicit* arguments.[9] Ultimately, given the diversity of semantic resources, we expect that learning from heterogeneous annotations in different corpora will be necessary to build automatic semantic analyzers that are both accurate and robust.

[9] They applied frustratingly easy domain adaptation to learn from FrameNet along with a PropBank-like dataset of nominal frames.

Chapter 7

Conclusion

In this chapter we will review and summarize the developments of the previous sections, reiterating the main themes and contributions of this thesis. We will also outline some of the immediate and not so immediate directions for future work, that are borne out of this thesis. We start with a recap of the previous sections, before presenting a rough roadmap for future.

7.1 Summary and key contributions

With all the techniques and results in place now, we can step back and ask how they help address the key questions raised in the beginning of this thesis. We begin with the various techniques developed to address the host-pathogen PPI prediction problem. Prior to this thesis, bulk of the work in area had focused on picking one specific host-pathogen pair of interest, host being typically human, and either applying existing machine learning techniques to the problem, or developing better techniques tailored to the problem. Since the labeled examples for such a development are typically drawn from previous lab studies recorded in databases, this naturally restricted the development of computational models to pathogens that are already relatively well-studied. This thesis takes important steps in eliminating this crucial bottleneck in designing computational models for host-pathogen PPI prediction through the development of a series of multitask learning models.

In Chapter §3, we presented a novel multitask learning method that encodes a specific biologically motivated hypothesis–namely similar pathogens target similar processes in a host. A key challenge in this work was to take this high-level intuition, and encode it into a mathematical formulation that is also computationally tractable. We design the MTPL-regularization framework which is able to achieve this goal. The regularizer lends itself to fairly efficient optimization using the Convex-Concave Procedure algorithm. This enables us to develop predictive models not only for well-studied pathogens such as *Y. Pestis* and *B. Anthracis*, but also *S. Typhimurium* and *E. Coli* for which the labeled data is extremely scarce. By leveraging the MTPL hypothesis, our models nevertheless perform quite well on these tasks with scarce data, significantly improving upon several strong baselines which either ignore the task-relatedness or use generic machine learning methods not necessarily motivated by any biological insights. With our collaborators, we further validated the accuracy of our predictions via laboratory based experiments.

In Chapter §4, we take an alternative viewpoint towards the same problem, instead viewing it as a multitask link prediction problem across several related networks. We build on the recent successes of matrix completion approaches based on low-rank matrix factorization for this setup. We propose a bilinear predictive model that encodes task-relatedness through a shared low-rank weight matrix. To allow for deviations from this shared model, we also provide each task with an additional sparse weight matrix. Once again, we test the model across a set of tasks where each task is individually extremely data-poor. Despite this, our method substantially outperforms a number of baselines, including our MTPL technique. While being predictively strong, the model also has some appealing properties that can be used to develop hypotheses regarding how various biological aspects influence the interactions. Specifically, the shared low-rank component can be seen as discovering a latent feature space, where similar host and pathogen proteins tend to interact. It is then intuitive to try and understand the properties of two proteins that result in proximity in this latent feature space. Further, the sparse component leads us to sequence level properties that are specific to each pathogen. A detailed analysis of these has the potential to reveal new pathogen specific mechanisms.

While the previous chapters focus on human-pathogen interactions, a problem of natural interest, Chapter §5 aims to discover host-pathogen interactions where the host is the plant *Arabidopsis thaliana* and the pathogen of interest is *S. Typhimurium*. Since we have no previous known PPI predictions for *A. thaliana*, this rules out most existing supervised machine learning tehcniques. Nevertheless, we have labeled examples when the host is human, which naturally motivates a transfer learning approach. We apply the technique of Kernel Mean Matching, which intuitively finds human proteins most similar to those of *A. thaliana*, and then learns a predictive model based on the PPI interactions involving these human proteins. In our quantitative evaluation, we find that this approach works extremely well relative to several baselines.

In the last chapter (Chapter §6), we present a transfer learning setting from a different application area: natural language processing. In the semantic parsing problem that we consider, the tasks arise as a result of a variation in representations, a difference in distribution of labels and features. To leverage information from these disparate resources, we use known feature augmentation based approaches. These work by incorporating data from other resources in the form of additional features. Our results show that combining information improves the coverage of our semantic-role-labeling model, resulting in state-of-the-art performance on this task.

7.2 Future research directions

Many of the applications considered in this thesis were in the context of host-pathogen PPI prediction, however the built models and the lessons learned can be applied in other applications. In particular, the multitask matrix completion model that we develop in Chapter §4 can be applied to the semantic parsing problem presented in Chapter §6. The semantic role labeling problem can be cast as a link prediction problem, where we wish to predict links between arguments and their semantic roles. Given the span of an argument (where rows of the matrix represent spans) the entries of the matrix indicate which semantic role/roles (the columns represent the various roles), it is most likely to belong to. The features for the rows (i.e spans) would be

the part-of-speech tags, the dependency parse information, context information. For the columns (i.e the roles) an indicator feature vector can encode the specific role. We can further incorporate the role's relationship to other closely related roles. The main challenges in adapting our method to this problem are: the number of rows will be very large. This is because the possible contexts in which all the roles appear are potentially unlimited. Besides scaling the algorithms to work in this setting we also need to develop approaches to reduce the number of rows. One possibility is to use templates for the spans, with one template representing a set of very similar spans. This will collapse multiple spans into a single row, but it brings up another challenge: generating features at a template-level that are yet specific to the span seen.

Some other examples of applications arising in biological data that can benefit from some of the methods that we develop in this thesis are:

- *Biomarkers for disease prediction*: Bio-fluids such as urine which can be collected non-invasively have become attractive biomarkers for early diagnosis of diseases. Capillary-electrophoresis coupled to mass spectrometry (CE-MS) has been used to identify the proteins and peptides present in urine - this data has been used in supervised machine learning models [Kuznetsova et al., 2012] to find patterns which correlate with disease conditions. Each of these studies required the collection of many samples over an extended period of time. By combining information across similar disease conditions (for example: coronary artery disease, hypertensivity etc), it will be possible to obtain good models with few samples and for several diseases.

- Cancer is considered a heterogeneous disease specific to cell type and tissue origin. However, most cancers share a common pathogenesis and may share common mechanisms [Stratton et al., 2009]. Multi-tasking learning methods can thus be used to predict core cancer genes important for many cancers, cancer type and stage classification using tissue sample data.

In the general multitask learning direction, the following directions seem promising.

- *Multi-source, multitask learning*: Often the data in every task can come from a variety of sources each of which has its own characteristics such as distributional skew, noise etc. For instance, protein interactions broadly come from small-scale experiments (more reliable) or high-throughput studies (more noisy). The current models do not account for this explicitly. This is true of crowd-sourcing datasets as well. Information can be shared more easily across similar sources from different tasks.

- *Task clustering using natural language text*: Currently, the design of multitask learning models is limited to experts who have a good knowledge of the learning algorithms, the data distributions. The approaches towards learning models on new tasks (with little or no labeled data available), use feature distributions, problem structure or unlabeled data to cluster the new task with existing tasks. It will be interesting if a non-expert is able to achieve this clustering without having to generate features and structural information by merely specifying a natural language description of the new task.

Bibliography

J. Abernethy, F. Bach, T. Evgeniou, and J. P. Vert. A new approach to collaborative filtering: Operator estimation with spectral regularization. *Journal of Machine Learning Research (JMLR)*, 2009.

Arvind Agarwal, Samuel Gerber, and Hal Daume. Learning multiple tasks using manifold regularization. In *Advances in neural information processing systems*, pages 46–54, 2010.

Greg M Allenby and Peter E Rossi. Marketing models of consumer heterogeneity. *Journal of Econometrics*, 89(1):57–78, 1998.

Rie Kubota Ando and Tong Zhang. A framework for learning predictive structures from multiple tasks and unlabeled data. *The Journal of Machine Learning Research*, 6: 1817–1853, 2005.

A. Argyriou, T. Evgeniou, and M. Pontil. Convex multi-task feature learning. *Machine Learning*, 2008.

M. Ashburner et al. Gene ontology: tool for the unification of biology. *Nat. Genet.*, 25 (1):25–9, 2000. `http://www.geneontology.org/`.

Collin Baker, Michael Ellsworth, and Katrin Erk. SemEval-2007 Task 19: frame semantic structure extraction. In *Proc. of SemEval*, pages 99–104, Prague, Czech Republic, June 2007.

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. The Berkeley FrameNet project. In *Proc. of COLING-ACL*, pages 86–90, Montreal, Quebec, Canada, August 1998. URL `http://framenet.icsi.berkeley.edu`.

Bart Bakker and Tom Heskes. Task clustering and gating for bayesian multitask learning. *The Journal of Machine Learning Research*, 4:83–99, 2003.

Ulrike Baldewein, Katrin Erk, Sebastian Padó, and Detlef Prescher. Semantic role labelling with similarity-based generalization using EM-based clustering. In Rada Mihalcea and Phil Edmonds, editors, *Proc. of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 64–68, Barcelona, Spain, July 2004.

T. Barrett, D.B. Troup, S.E. Wilhite, et al. Ncbi geo: archive for functional genomics data sets 10 years on. *Nucleic Acids Res.*, 39(Database issue):D1005–10, 2011.

Jonathan Baxter. A model of inductive bias learning. *J. Artif. Intell. Res.(JAIR)*, 12: 149–198, 2000.

Shai Ben-David, John Blitzer, Koby Crammer, Fernando Pereira, et al. Analysis of representations for domain adaptation. *Advances in neural information processing systems (NIPS)*, 19:137, 2007.

Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79(1-2):151–175, 2010.

Cedric N Berger, Derek J Brown, Robert K Shaw, Florencia Minuzzi, Bart Feys, and Gad Frankel. Salmonella enterica strains belonging to o serogroup 1, 3, 19 induce chlorosis and wilting of arabidopsis thaliana leaves. *Environmental microbiology*, 13 (5):1299–1308, 2011.

G.F. Berriz, O.D. King, B. Bryant, et al. Characterizing gene sets with funcassociate. *Bioinf.*, 19(18):2502–04, 2003. `http://llama.mshri.on.ca/funcassociate/`.

John Blitzer, Ryan McDonald, and Fernando Pereira. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 conference on empirical methods in natural language processing*, pages 120–128. Association for Computational Linguistics, 2006.

Philipp Blohm, Goar Frishman, Pawel Smialowski, Florian Goebels, Benedikt Wachinger, Andreas Ruepp, and Dmitrij Frishman. Negatome 2.0: a database of non-interacting proteins derived by literature mining, manual annotation and protein structure analysis. *Nucleic acids research*, page gkt1079, 2013.

Claire Bonial, Kevin Stowe, and Martha Palmer. Renewing and revising SemLink. In *Proc. of the 2nd Workshop on Linked Data in Linguistics (LDL-2013): Representing and linking lexicons, terminologies and other language data*, pages 9–17, Pisa, Italy, September 2013.

Claire Bonial, Julia Bonn, Kathryn Conger, Jena D. Hwang, and Martha Palmer. PropBank: semantics of new predicate types. In Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis, editors, *Proc. of LREC*, pages 3013–3019, Reykjavík, Iceland, May 2014.

Edwin V Bonilla, Kian M Chai, and Christopher Williams. Multi-task gaussian process prediction. In *Advances in neural information processing systems*, pages 153–160, 2007.

Antoine Bordes, Xavier Glorot, Jason Weston, and Yoshua Bengio. Joint learning of words and meaning representations for open-text semantic parsing. In *International Conference on Artificial Intelligence and Statistics*, pages 127–135, 2012.

Léon Bottou. From machine learning to machine reasoning. *Machine learning*, 94(2): 133–149, 2014.

E. Candes and B. Recht. Exact matrix completion via convex optimization. *Foundations of Computational Mathematics*, 2008.

Bin Cao, Nathan N. Liu, and Qiang Yang. Transfer learning for collective link prediction in multiple heterogenous domains. *ICML*, 2010.

Rich Caruana. Multitask learning. *Machine learning*, 28(1):41–75, 1997.

Jianhui Chen, Ji Liu, and Jieping Ye. Learning incoherent sparse and low-rank patterns from multiple tasks. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 5(4):22, 2012a.

Jianhui Chen, Lei Tang, Jun Liu, and Jieping Ye. A convex formulation for learning a shared predictive structure from multiple tasks. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(5):1025–1038, 2013.

K. C. Chen, T. Y. Wang, and C. H. Chan. Associations between hiv and human pathways revealed by protein-protein interactions and correlated gene expression profiles. *PLOS One*, 2012b.

X.W. Chen and M. Liu. Prediction of protein-protein interactions using random decision forest framework. *Bioinformatics*, 21(24):4394–400, 2005.

William W. Cohen and Vitor R. Carvalho. Stacked sequential learning. In *Proc. of IJCAI*, pages 671–676, Edinburgh, Scotland, UK, 2005.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. Natural language processing (almost) from scratch. *The Journal of Machine Learning Research*, 12:2493–2537, 2011.

Corinna Cortes, Mehryar Mohri, Michael Riley, and Afshin Rostamizadeh. Sample selection bias correction theory. In *Algorithmic learning theory*, pages 38–53. Springer, 2008.

Koby Crammer, Michael Kearns, and Jennifer Wortman. Learning from multiple sources. *The Journal of Machine Learning Research (JMLR)*, 9:1757–1774, 2008.

Dipanjan Das and Noah A. Smith. Semi-supervised frame-semantic parsing for unknown predicates. In *Proc. of ACL-HLT*, pages 1435–1444, Portland, Oregon, USA, June 2011.

Dipanjan Das and Noah A. Smith. Graph-based lexicon expansion with sparsity-inducing penalties. In *Proc. of NAACL-HLT*, pages 677–687, Montréal, Canada, June 2012.

Dipanjan Das, André F. T. Martins, and Noah A. Smith. An exact dual decomposition algorithm for shallow semantic parsing with constraints. In *Proc. of *SEM*, pages 209–217, Montréal, Canada, June 2012.

Dipanjan Das, Desai Chen, André F. T. Martins, Nathan Schneider, and Noah A. Smith. Frame-semantic parsing. *Computational Linguistics*, 40(1):9–56, March 2014. URL `http://www.ark.cs.cmu.edu/SEMAFOR`.

Hal Daumé, III. Frustratingly easy domain adaptation. In *Proc. of ACL*, pages 256–263, Prague, Czech Republic, June 2007.

Jesse Davis and Mark Goadrich. The relationship between precision-recall and roc curves. *Proceedings of the 23rd international conference on Machine learning*, pages 233–240, 2006.

Sanjukta Dey, Marion Wenig, Gregor Langen, Sapna Sharma, Karl Kugler, Claudia Knappe, Bettina Hause, Marlies Bichlmeier, Va liollah Babaeizad, Jafargholi Imani, et al. Bacteria-triggered systemic immunity in barley appears to be associated with wrky and ethylene responsive factors but not with salicylic acid. *Plant physiology*, 166(4):pp–114, 2014.

T. Driscoll, Dyer M. D., Murali T. M., and Sobral B. W. Pig–the pathogen interaction gateway. *Nucleic Acids Res.*, 37:D647–50, 2009.

Z. Du, L. Li, Chin-Fu Chen, P. S. Yu, and J. Z. Wang. G-sesame: web tools for go term based gene similarity analysis and knowledge discovery. *Nucleic Acids Research*, 37 (Web Server issue):W345–9, 2009.

M.D. Dyer, T.M. Murali, and B.W. Sobral. Computational prediction of host-pathogen protein-protein interactions. *Bioinformatics*, 23(13):i159–66, 2007.

M.D. Dyer, T.M. Murali, and B.W. Sobral. The landscape of human proteins interacting with viruses and other pathogens. *PLOS Pathogens*, 4(2):e32, 2008.

M.D. Dyer et al. The human-bacterial pathogen protein interaction networks of bacillus anthracis, francisella tularensis, and yersinia pestis. *PLOS One*, 5(8), 2010.

M.D. Dyer et al. Supervised learning and prediction of physical interactions between human and hiv proteins. *Infect., Genetics and Evol.*, 11:917–923, 2011.

T. Evgeniou and M. Pontil. Regularized multi-task learning. *ACM SIGKDD*, 2004.

R.E. Fan et al. Liblinear: A library for large linear classification. *JMLR*, 9, 2008. Software available at `http://www.csie.ntu.edu.tw/~cjlin/liblinear`.

Wei Fan, Ian Davidson, Bianca Zadrozny, and Philip S Yu. An improved categorization of classifier's sensitivity on sample selection bias. In *Data Mining, Fifth IEEE International Conference on*, pages 4–pp. IEEE, 2005.

Parvin Sadat Feizabadi and Sebastian Padó. Combining seemingly incompatible corpora for implicit semantic role labeling. In *Proc. of *SEM*, pages 40–50, Denver, Colorado, USA, June 2015.

Charles J. Fillmore and Collin Baker. A frames approach to semantic analysis. In Bernd Heine and Heiko Narrog, editors, *The Oxford Handbook of Linguistic Analysis*, pages 791–816. Oxford University Press, Oxford, UK, December 2009.

R.D. Finn et al. ipfam: visualization of protein–protein interactions in pdb at domain and amino acid resolutions. *Bioinf.*, 21(3):410–2, 2005.

R.D. Finn et al. The pfam protein families database. *Nucl. Acids Res.*, 38:D211–22, 2010.

Michael Fleischman, Namhee Kwon, and Eduard Hovy. Maximum entropy models for FrameNet classification. In Michael Collins and Mark Steedman, editors, *Proc. of EMNLP*, pages 49–56, 2003.

J. Garcia, E. Guney, et al. Biana: a software framework for compiling biological interactions and analyzing networks. *BMC Bioinformatics*, 11:56, 2010.

Javier Garcia-Garcia, Sylvia Schleker, Judith Klein-Seetharaman, and Baldo Oliva. Bips: Biana interolog prediction server. a tool for protein–protein interaction inference. *Nucleic acids research*, 40(W1):W147–W151, 2012.

Daniel Gildea and Daniel Jurafsky. Automatic labeling of semantic roles. *Computational Linguistics*, 28(3):245–288, 2002.

Iryna Gurevych, Judith Eckle-Kohler, Silvana Hartmann, Michael Matuschek, Christian M. Meyer, and Christian Wirth. UBY - a large-scale unified lexical-semantic resource based on LMF. In *Proc. of EACL*, pages 580–590, Avignon, France, April 2012.

Karl Moritz Hermann, Dipanjan Das, Jason Weston, and Kuzman Ganchev. Semantic frame identification with distributed word representations. In *Proc. of ACL*, pages 1448–1458, Baltimore, Maryland, USA, June 2014.

C. Hernandez-Reyes and A. Schikora. Salmonella, a cross-kingdom pathogen infecting humans and plants. *FEMS Microbiology Letters*, 343:1–7, 2013.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. OntoNotes: the 90% solution. In *Proc. of HLT-NAACL*, pages 57–60, New York City, USA, June 2006.

J. Huang, A. Smola, A. Gretton, K.M. Borgwardt, and B. Scholkopf. Correcting sample selection bias by unlabeled data. *NIPS*, 2007.

A.L. Iniguez, Y.M. Dong, H.D. Carter, B.M.M. Ahmer, J.M. Stone, and E.W. Triplett. Regulation of enteric endophytic bacterial colonization by plant defenses. *Molecular Plant-Microbe Interactions*, 18:169–178, 2005.

Laurent Jacob, Jean-philippe Vert, and Francis R Bach. Clustered multi-task learning: A convex formulation. In *Advances in neural information processing systems (NIPS)*, pages 745–752, 2009.

Ali Jalali, Sujay Sanghavi, Chao Ruan, and Pradeep K Ravikumar. A dirty model for multi-task learning. *Advances in Neural Information Processing Systems*, pages 964–972, 2010.

Nathalie Japkowicz and Shaju Stephen. The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5):429–449, 2002.

Shuiwang Ji and Jieping Ye. An accelerated gradient method for trace norm minimization. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 457–464. ACM, 2009.

Jing Jiang and ChengXiang Zhai. Instance weighting for domain adaptation in nlp. In *ACL*, volume 7, pages 264–271, 2007.

Rong Jin, Luo Si, and ChengXiang Zhai. Preference-based graphic models for collaborative filtering. In *UAI*, pages 329–336, 2002.

Thorsten Joachims. Transductive inference for text classification using support vector machines. *ICML*, 99:200–209, 1999.

Thorsten Joachims. Svm^{light}, 2008. `http://svmlight.joachims.org/`.

Richard Johansson. Non-atomic classification to improve a semantic role labeler for a low-resource language. In *Proceedings of the Sixth International Workshop on Semantic Evaluation*, pages 95–99. Association for Computational Linguistics (ACL), 2012.

Richard Johansson. Training parsers on incompatible treebanks. In *Proc. of NAACL-HLT*, pages 127–137, Atlanta, Georgia, USA, June 2013.

Gregory Jubelin, Frederic Taieb, et al. Pathogenic bacteria target nedd8-conjugated cullins to hijack host-cell signaling pathways. *PLOS Pathogens*, 2010.

J.G. Kim, W. Stork, and M.B. Mudgett. Xanthomonas type iii effector xopd desumoylates tomato transcription factor slerf4 to suppress ethylene responses and promote pathogen growth. *Cell Host Microbe*, 13(2):143–54, 2013.

M. W. B. Kirzinger, G. Nadarasah, and J. Stavrinides. Insights into cross-kingdom plant pathogenic bacteria. *Genes*, 2:980–997, 2011.

Lingpeng Kong, Nathan Schneider, Swabha Swayamdipta, Archna Bhatia, Chris Dyer, and Noah A. Smith. A dependency parser for tweets. In *Proc. of EMNLP*, pages 1001–1012, Doha, Qatar, October 2014.

Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. *Computer*, (8):30–37, 2009.

M. Kshirsagar, J. G. Carbonell, and J. Klein-Seetharaman. Techniques to cope with missing data in host-pathogen protein interaction prediction. *Bioinformatics*, 2012.

M. Kshirsagar, J. G. Carbonell, and J. Klein-Seetharaman. Multi-task learning for host-pathogen protein interactions. *Bioinformatics*, 2013.

M. Kshirsagar, S. Schleker, J. Carbonell, and J. Klein-Seetharaman. Techniques for transferring host-pathogen protein interactions knowledge to new tasks. *Frontiers in Microbiology*, 6(36), 2015a.

M. Kshirsagar, S. Thomson, N. Schneider, J. Carbonell, N. Smith, and C. Dyer. Frame-semantic role labeling with heterogeneous annotations. *Assoc. for Computational Linguistics (ACL)*, 2015b.

Abhishek Kumar and Hal Daume III. Learning task grouping and overlap in multi-task learning. 2012.

R. Kumar and B. Nanduri. Hpidb–a unified resource for host-pathogen interactions. *BMC Bioinf.*, 2010.

T. Kuznetsova, H. Mischak, W. Mullen, and J.A. Staessen. Urinary proteome analysis in hypertensive patients with left ventricular diastolic dysfunction. *European Heart Journal*, 2012.

Namhee Kwon, Michael Fleischman, and Eduard Hovy. FrameNet-based semantic parsing using maximum entropy models. In *Proc. of Coling*, pages 1233–1239, Geneva, Switzerland, August 2004.

Philippe Lamesch, Tanya Z Berardini, Donghui Li, David Swarbreck, Christopher Wilks, Rajkumar Sasidharan, Robert Muller, Kate Dreher, Debbie L Alexander, Margarita Garcia-Hernandez, et al. The arabidopsis information resource (tair): improved gene annotation and new tools. *Nucleic acids research*, 40(D1):D1202–D1210, 2012.

Bin Li. Cross-domain collaborative filtering: A brief survey. In *Tools with Artificial Intelligence (ICTAI), 2011 23rd IEEE International Conference on*, pages 1085–1086. IEEE, 2011.

Bin Li, Qiang Yang, and Xiangyang Xue. Transfer learning for collaborative filtering via a rating-matrix generative model. In *International Conference on Machine Learning*, pages 617–624. ACM, 2009.

Yan Li, QingQing Zhang, Jiangguang Zhang, Liang Wu, Yijun Qi, and Jian-Min Zhou. Identification of micrornas involved in pathogen-associated molecular pattern-triggered plant innate immunity. *Plant physiology*, 152(4):2222–2231, 2010.

Ken Litkowski. SENSEVAL-3 task: Automatic labeling of semantic roles. In Rada Mihalcea and Phil Edmonds, editors, *Proc. of SENSEVAL-3, the Third International Workshop on the Evaluation of Systems for the Semantic Analysis of Text*, pages 9–12, Barcelona, Spain, July 2004.

Dong C. Liu and Jorge Nocedal. On the Limited Memory BFGS Method for Large Scale Optimization. *Math. Program.*, 45(3):503–528, December 1989.

Jinyi Liu, J Hollis Rice, Nana Chen, Thomas J Baum, and Tarek Hewezi. Synchronization of developmental processes and defense signaling by growth regulating transcription factors. *PloS one*, 9(5):e98477, 2014.

Jun Liu, Shuiwang Ji, and Jieping Ye. Multi-task feature learning via efficient $l_{2,1}$-norm minimization. In *Proceedings of the twenty-fifth conference on uncertainty in artificial intelligence (UAI)*, pages 339–348, 2009.

Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Multiple source adaptation and the rényi divergence. In *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence*, pages 367–374. AUAI Press, 2009.

André F. T. Martins, Dipanjan Das, Noah A. Smith, and Eric P. Xing. Stacking dependency parsers. In *Proc. of EMNLP*, pages 157–166, Honolulu, Hawaii, October 2008.

Yuichiroh Matsubayashi, Naoaki Okazaki, and Jun'ichi Tsujii. A comparative study on generalization of semantic roles in FrameNet. In *Proc. of ACL-IJCNLP*, pages 19–27, Suntec, Singapore, August 2009.

L. Matthews, G. Gopinath, M. Gillespie, et al. Reactome knowledgebase of biological pathways and processes. *Nucleic Acids Res.*, 2008.

Andreas Maurer. Bounds for linear multi-task learning. *The Journal of Machine Learning Research*, 7:117–139, 2006.

Andreas Maurer, Massi Pontil, and Bernardino Romera-Paredes. Sparse coding for multitask and transfer learning. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, pages 343–351, 2013.

A. K. Menon and C. Elkan. Link prediction via matrix factorization. *ECML*, 2011.

Tom M Mitchell. *The need for biases in learning generalizations*. Department of Computer Science, Laboratory for Computer Science Research, Rutgers Univ., 1980.

M. Shahid Mukhtar, Anne-Ruxandra Carvunis, M. Dreze, et al. Independently evolved virulence effectors converge onto hubs in a plant immune system network. *Science*, 333(6042):596–601, 2011a.

M Shahid Mukhtar, Anne-Ruxandra Carvunis, Matija Dreze, Petra Epple, Jens Steinbrenner, Jonathan Moore, Murat Tasan, Mary Galli, Tong Hao, Marc T Nishimura, et al. Independently evolved virulence effectors converge onto hubs in a plant immune system network. *science*, 333(6042):596–601, 2011b.

Asuka Nanbo, Masaki Imai, Shinji Watanabe, et al. Ebolavirus is internalized into host cells via macropinocytosis in a viral glycoprotein-dependent manner. *PLoS pathogens*, 6(9):e1001121, 2010.

Nagarajan Natarajan and Inderjit S. Dhillon. Inductive matrix completion for predicting genedisease associations. *Bioinformatics*, 2014.

Joakim Nivre and Ryan McDonald. Integrating graph-based and transition-based dependency parsers. In *Proc. of ACL-HLT*, pages 950–958, Columbus, Ohio, USA, June 2008.

G. Ostlund, T. Schmitt, K. Forslund, T. Kostler, D.N. Messina, S. Roopra, O. Frings, and ELL. Sonnhammer. Inparanoid 7: new algorithms and tools for eukaryotic orthology analysis. *Nucleic Acids Res.*, 38:D196–D203, 2010.

Martha Palmer, Daniel Gildea, and Paul Kingsbury. The Proposition Bank: an annotated corpus of semantic roles. *Computational Linguistics*, 31(1):71–106, March 2005.

Ellie Pavlick, Travis Wolfe, Pushpendre Rastogi, Chris Callison-Burch, Mark Drezde, and Benjamin Van Durme. FrameNet+: Fast paraphrastic tripling of FrameNet. In *Proc. of ACL-IJCNLP*, Beijing, China, July 2015.

Anastasia Pentina and Christoph H Lampert. A pac-bayesian bound for lifelong learning. *ICML*, 2014.

Dinh Q Phung, Svetha Venkatesh, et al. Ordinal boltzmann machines for collaborative filtering. In *UAI*, pages 548–556. AUAI Press, 2009.

Joan Planas-Iglesias, Manuel A Marin-Lopez, Jaume Bonet, Javier Garcia-Garcia, and Baldo Oliva. iloops: a protein–protein interaction prediction server based on structural features. *Bioinformatics*, 29(18):2360–2362, 2013.

T.S.K. Prasad et al. Human protein reference database - 2009 update. *Nucl. Acids Res.*, 3(Database issue):D767–72, 2009.

Vasin Punyakanok, Dan Roth, and Wen-tau Yih. The importance of syntactic parsing and inference in semantic role labeling. *Computational Linguistics*, 34(2): 257–287, 2008. URL `http://cogcomp.cs.illinois.edu/page/software_view/SRL`.

Y. Qi, H.K. Dhiman, Z. Bar-Joseph, et al. Systematic prediction of human membrane receptor interactions. *Proteomics*, 23(9):5243–55, 2009.

Y. Qi, O. Tastan, J. G. Carbonell, J. Klein-Seetharaman, and J. Weston. Semi-supervised multi-task learning for predicting interactions between hiv-1 and human proteins. *Bioinformatics*, 2010.

Y. Qi et al. Evaluation of different biological data and computational classification methods for use in protein interaction prediction. *Proteins*, 63(3):490–500, 2006.

Rajat Raina, Andrew Y Ng, and Daphne Koller. Constructing informative priors using transfer learning. In *Proceedings of the 23rd international conference on Machine learning*, pages 713–720. ACM, 2006.

Josef Ruppenhofer, Michael Ellsworth, Miriam R. L. Petruck, Christopher R. Johnson, and Jan Scheffczyk. FrameNet II: extended theory and practice, September 2010. URL `https://framenet2.icsi.berkeley.edu/docs/r1.5/book.pdf`.

Badrul Sarwar, George Karypis, Joseph Konstan, and John Riedl. Item-based collaborative filtering recommendation algorithms. In *WWW*, pages 285–295. ACM, 2001.

C. F. Schaefer, K. Anthony, S. Krupa, et al. Pid: The pathway interaction database. *Nucleic Acids Res.*, 2009.

A. Schikora, I. Virlogeux-Payant, E. Bueso, A.V. Garcia, T. Nilau, A. Charrier, S. Pelletier, P. Menanteau, M. Baccarini, P. Velge, and H. Hirt. Conservation of salmonella infection mechanisms in plants and animals. *PLoS One*, 6(e24112), 2011.

Adam Schikora, Alessandro Carreri, Emmanuelle Charpentier, and Heribert Hirt. The dark side of the salad: Salmonella typhimurium overcomes the innate immune response of arabidopsis thaliana and shows an endopathogenic lifestyle. *PLoS One*, 3(5):e2279, 2008.

S. Schleker, J. Sun, B. Raghavan, et al. The current salmonella-host interactome. *Proteomics Clin Appl.*, 6(1-2):117–33, 2012.

S. Schleker, M. Kshirsagar, and J. Klein-Seetharaman. Comparing human-salmonella with plant-salmonella protein-protein interaction predictions. *Frontiers in Microbiology*, 6(45), 2015.

Anton Schwaighofer, Volker Tresp, and Kai Yu. Learning gaussian process kernels via hierarchical bayes. In *Advances in Neural Information Processing Systems*, pages 1209–1216, 2004.

R. Sharan, S. Suthram, R.M. Kelley, T. Kuhn, S. McCuine, P. Uetz, T. Sittler, R.M. Karp, and T. Ideker. Conserved patterns of protein interaction in multiple species. *Proc. Natl. Academy Sciences (PNAS)*, 2005.

Juwen Shen, Jian Zhang, Xiaomin Luo, Weiliang Zhu, Kunqian Yu, Kaixian Chen, Yixue Li, and Hualiang Jiang. Predicting protein–protein interactions based only on sequences information. *Proceedings of the National Academy of Sciences*, 104(11): 4337–4341, 2007.

Natali Shirron and Sima Yaron. Active suppression of early immune response in tobacco by the human pathogen salmonella typhimurium. *PLoS One*, 6(4):e18855, 2011.

A. P. Singh and G. J. Gordon. Relational learning via collective matrix factorization. *KDD*, 2008.

R. Singh, J. Xu, and B. Berger. Struct2net: Integrating structure into protein-protein interaction prediction. *Pacific Symposium on Biocomputing*, 2006.

Sören Sonnenburg, Gunnar Rätsch, Sebastian Henschel, Christian Widmer, Jonas Behr, Alexander Zien, Fabio de Bona, Alexander Binder, Christian Gehl, and Vojtěch Franc. The shogun machine learning toolbox. *The Journal of Machine Learning Research*, 11:1799–1802, 2010. `http://www.shogun-toolbox.org`.

A. Stein et al. 3did: identification & classification of domain-based interactions of known 3d structure. *Nuc. Acids Res.*, 39:D718–23, 2011.

Michael R Stratton, Peter J Campbell, and P Andrew Futreal. The cancer genome. *Nature*, 458(7239):719–724, 2009.

M. Sugiyama, S. Nakajima, H. Kashima, P.V. Buenau, and M. Kawanabe. Direct importance estimation with model selection and its application to covariate shift adaptation. *NIPS*, 2008.

Oscar Täckström, Kuzman Ganchev, and Dipanjan Das. Efficient inference and structured learning for semantic role labeling. *Transactions of the Association for Computational Linguistics*, 3:29–41, January 2015.

O. Tastan et al. Prediction of interactions between hiv-1 and human proteins by information integration. *Pac. Symp. Biocomput.*, (14):516–527, 2009.

S. D. Tekir, Ali S., Tunahan C., and Kutlu O. U. Infection strategies of bacterial and viral pathogens through pathogen-host protein protein interactions. *Frontiers in Microbial Immunology*, 2012.

Cynthia A. Thompson, Roger Levy, and Christopher D. Manning. A generative model for semantic role labeling. In *Machine Learning: ECML 2003*, pages 397–408, 2003.

Sebastian Thrun. Is learning the n-th thing any easier than learning the first? *Advances in neural information processing systems*, pages 640–646, 1996.

Sebastian Thrun and Lorien Pratt. Learning to learn: Introduction and overview. In *Learning to learn*, pages 3–17. Springer, 1998.

P. Uetz, Y. A. Dong, et al. Herpesviral protein networks and their interaction with the human proteome. *Science*, (311):239–242, 2006.

Selen Uguroglu and Jaime Carbonell. Feature selection for transfer learning. In *Machine Learning and Knowledge Discovery in Databases (ECML-PKDD)*, pages 430–442. Springer, 2011.

UniProt Consortium. Ongoing and future developments at the universal protein resource. *Nucl. Acids Res.*, 39:D214–D219, 2011.

Vladimir Naumovich Vapnik. *Statistical learning theory*, volume 1. Wiley New York, 1998.

Randi Vita, James A Overton, Jason A Greenbaum, et al. The immune epitope database (iedb) 3.0. *Nucleic acids research*, 43(D1):D405–D412, 2015.

R.-S. Wang, Y. Wang, et al. Analysis on multi-domain cooperation for predicting protein-protein interactions. *BMC Bioinformatics*, 2007.

Xuezhi Wang and Jeff Schneider. Flexible transfer learning under support and model shift. In *Advances in Neural Information Processing Systems (NIPS)*, pages 1898–1906, 2014.

C. Widmer, J. Leiva, Y. Altun, and G. Ratsch. Leveraging sequence classification by taxonomy-based multitask learning. *RECOMB*, 2010.

R. Winnenburg, M. Urban, A. Beacham, et al. Phi-base update: additions to the pathogen host interaction database. *Nucleic Acids Res*, 2008.

Pengcheng Wu and Thomas G Dietterich. Improving svm accuracy by training on auxiliary data sources. In *Proceedings of the twenty-first international conference on Machine learning*, page 110. ACM, 2004.

X. Wu, L. Zhu, J. Guo, et al. Prediction of yeast protein-protein interaction network: insights from the gene ontology and annotations. *Nucleic Acids Res.*, 34(7):2137–50, 2006.

Q. Xu, E. W. Xiang, and Q. Yang. Protein-protein interaction prediction via collective matrix factorization. *International Conference on Bioinformatics and Biomedicine*, 2010.

Qian Xu and Qiang Yang. A survey of transfer and multitask learning in bioinformatics. *Journal of Computing Science and Engineering*, 5(3):257–268, 2011.

Zhao Xu, Kristian Kersting, and Volker Tresp. Multi-relational learning with gaussian processes. *IJCAI*, 2009.

John Yu and T. Joachims. Learning structural svms with latent variables. *ICML*, 2009.

Yang Yu and Zhi-Hua Zhou. A framework for modeling positive class expansion with single snapshot. In *Advances in Knowledge Discovery and Data Mining (KDD)*, pages 429–440. Springer, 2008.

Xiao-Tong Yuan, Xiaobai Liu, and Shuicheng Yan. Visual classification with multitask joint sparse representation. *Image Processing, IEEE Transactions on*, 21(10):4349–4360, 2012.

A. Yuille and A. Rangarajan. The concave-convex procedure. *Neural Computation*, 2003.

Matthew Zeiler. ADADELTA: An Adaptive Learning Rate Method. *CoRR*, abs/1212.5701, 2012. URL `http://dblp.uni-trier.de/rec/bibtex/journals/corr/abs-1212-5701`.

Kun Zhang, Krikamol Muandet, Zhikun Wang, et al. Domain adaptation under target and conditional shift. In *Proceedings of the 30th International Conference on Machine Learning (ICML)*, pages 819–827, 2013.

Yi Zhang and Jeff G Schneider. Learning multiple tasks with a sparse matrix-normal penalty. In *Advances in Neural Information Processing Systems (NIPS)*, pages 2550–2558, 2010.

Yu Zhang, Bin Cao, and Dit-Yan Yeung. Multi-domain collaborative filtering. *arXiv preprint arXiv:1203.3535*, 2012.