Bringing High-Throughput Data Analysis to Biologists

by

Christopher S. Magnano

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

(Computer Sciences)

at the

UNIVERSITY OF WISCONSIN–MADISON

2022

Date of final oral examination: 03/07/2022

The dissertation is approved by the following members of the Final Oral Committee:
Sushmita Roy, Associate Professor, Biostatistics and Medical Informatics
Jignesh Patel, Professor, Computer Sciences
AnHai Doan, Professor, Computer Sciences
Rosemary Russ, Associate Professor, Curriculum and Instruction
Anthony Gitter, Associate Professor, Biostatistics and Medical Informatics

For my everyone whose support has nourished me,
for everyone who has tolerated the distance,
and at least a little bit for my cat.

ACKNOWLEDGMENTS

There are so many people who have helped me during my time at UW, but there are some I would like to thank for their specific contributions. I first want to thank my family and friends for all of their support (while noting that one's inclusion further down in these acknowledgments does not mean that I do not also consider you a friend). I am so lucky to have people around me who want me to succeed and help me along the way.

I want to especially thank Angela Thorp and Beth Bierman for all of their administrative help over the years. There is no way I, or really the entire CS and BMI departments, would be able to get anything done without you. I also want to thank coordinators and administrators for the CIBM program, especially Louise Pape, Mark Craven, and Sushmita Roy. My time with the CIBM program really helped me grow as a scientist.

I thank the members of my committee, Sushmita Roy, Jignesh Patel, AnHai Doan, Rosemary Russ, and Tony Gitter, for their feedback and for being so willing to help me get this dissertation and defense ready on shorter notice than is typical.

I would like to thank Eunju Park, Paul Ahlquist, James Bruce, and Mark Horswill for the HIV work we collaborated on that motivated my computational work. Anna Ritz, Tobias Rubel, and Pramesh Singh were invaluable for painting the big picture in network analysis.

The Delta program was a huge help in both my development as an educator and in setting the stage for my future growth in my teaching skills and ability to foster an inclusive community in my classroom. I especially want to thank Rosemary Russ, Devon Wixon, and Warren Scherer in this regard. Rosemary furthermore has also been a huge help in my educational research and my job hunt; her expertise and support were invaluable to both.

I want to thank everyone who attended pilot workshops and everyone I interviewed. I would not have been able to complete my research without you. I also want to thank others who worked on the ML4Bio workshop not mentioned yet, especially Debora Treu and Fangzhou Mu.

Members of my lab have been supportive and entertaining throughout my degree. Current lab members, Sam, David, Ben, Adam, Alyssa, and Daniel; and past members, Nafisah, Thevaa, Milica, Shengchao, Aaron, Atul, and Moayad, have both been great partners in collaboration and brainstorming, and made my experience so much more enjoyable than it would have otherwise been. I would especially like to thank Tony for all of his work and support over the years. I feel that I am especially lucky to have had an advisor I can always trust to advocate for and support me.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

Large-scale biological datasets, such as transcriptomic, proteomic, and other high-throughput assays, are routinely applied to characterize cellular states. While there is a proliferation of new computational methods for analyzing biological data, many of these methods never make it to actual use by biological researchers, or are not used to their full potential. In this thesis we explore methodological, computational, and educational roadblocks between computational methods and discovery in biological domains and create tools to help overcome them.

We specifically focus on biological network analyses, a popular class of methods in exploratory analyses. We propose a method, pathway parameter advising, which automates parameter selection in the pathway reconstruction task. This automation allows for biological researchers with less computational and methodological knowledge to take advantage of this class of methods more easily. We find that pathway parameter advising consistently selects parameters that lead to informative pathways with desirable topological qualities.

In addition to making methods more approachable, we also explore increasing the utility of biological network analyses by adding an additional layer of information to biological pathways, subcellular localization. We utilize public protein localization databases to predict the subcellular locations of protein interactions within the context of a specific biological pathway. This predictive task proves challenging, mainly due to surprising discrepancies between different sources of subcellular localization information. While the predictive problem remains unsolved, we are able to achieve moderate predictive performance in some cases and provide insight to the problem of contextualizing subcellular localization.

Finally, we present work involving the Machine Learning for Biologists (ML4Bio) workshop, a workshop for teaching machine learning concepts to biologists. The workshop focuses on practical machine learning skills for active biological researchers with minimal mathematical and computational background, emphasizing literature comprehension and experimental design. We conduct a study on 3 pilot workshops, finding that the workshop effectively introduces machine learning to biological researchers, especially increasing affective learning outcomes such as interest in and appreciation for machine learning. We also share lessons learned in designing educational materials for an active research audience.

Chapter 1

Introduction

The rapid increase in the scale of biological data available has affected nearly every field in biology (Reuter et al., 2015). This increase has been driven by high-throughput biological assays which can detect and measure wide varieties of biological entities. These methods tend to cast a wide net; they seek to capture a full picture of whatever biological entity they are measuring. Examples of this type of data include genomic data: direct readings of the genome, transcriptomic data: measuring the quantity of genes being converted into proteins, proteomic data: measuring the amount of proteins or examining the state of those proteins, or metabolomic data: measuring the amounts of other small molecules such as nutrients (Precone et al., 2015). These types of data are often referred to as "omic" datasets. There are a variety of technologies used to capture such data, such as mass spectrometry, RNA-seq, microarrays, ChIP-seq, and metabolic assays.

These datasets can be used to rapidly gain insight in biological processes. For instance, in Section 2.1 a motivating example is presented which shows how proteomic data gathered via mass spectrometry was used to investigate how HIV-1 infected cells respond to coming into contact with uninfected cells. The proteomic data allowed us to examine a snapshot of protein abundance and protein activation in infected cells when coming into contact with uninfected cells. While there are technological and data-specific considerations and data pre-processing which takes place, generally these "omic" datasets can be viewed as vectors of hundreds

to tens of thousands of real-valued measurements. However, this increase in data has also created an equal increase in the need for computational methods to analyze and interpret it. While the number of available bioinformatics tools has also rapidly increased, barriers remain impeding their adoption by and utility for biologists (Kulkarni and Frommolt, 2017).

One important facet of many experiments using "omic" datasets is that they are often not used to directly test specific biological hypotheses. Instead, these experiments are designed to find potential hypotheses for a specific biological mechanism, process, disease, or state. Computational methods which aid these exploratory analyses have different considerations than those that aid more targeted analyses, as the results of such methods are designed to aid discovery and interpretation rather than provide a correct answer. Therefore, interpretability and usability are even more important in these computational methods than in other methods in bioinformatics.

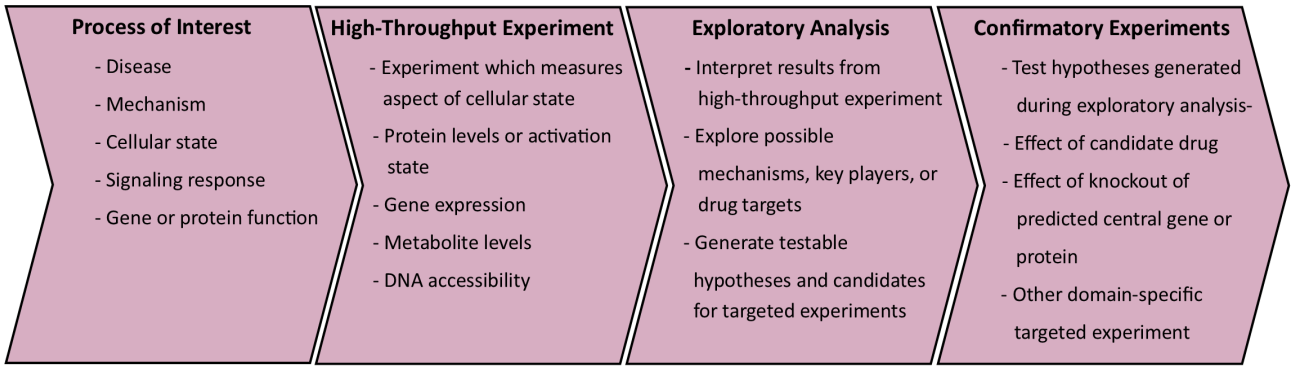

Figure 1.1: Overview of a typical workflow in using high-throughput experiments and exploratory analyses for biological discovery.

1.1 Network Analysis

One class of methods often used in exploratory biological analyses are network analyses. Network analysis can integrate and analyze large amounts of biological "omic" data from genomic, transcriptomic, proteomic, or metabolomic assays (Goh et al., 2012; Furlong, 2013). Placing omic data in a network context allows for the discovery of key members of a process that may be missed from a single data source and functional summarization for hypothesis

generation and other downstream analyses.

Biological networks which represent a single function or process are often referred to as pathways. Pathways can represent disease states, signaling pathways, cell metabolism, or cell cycle processes. A variety of databases exist containing pathways curated by biologists which represent current understanding of various biological processes, such as the Kyoto Encyclopedia of Genes and Genomes (KEGG) database (Kanehisa and Goto, 2000), Reactome (Fabregat et al., 2018), and NetPath (Kandasamy et al., 2010). An example of a biological pathway representing cell signaling in response to p53 activation from KEGG can be seen in Figure 1.2.

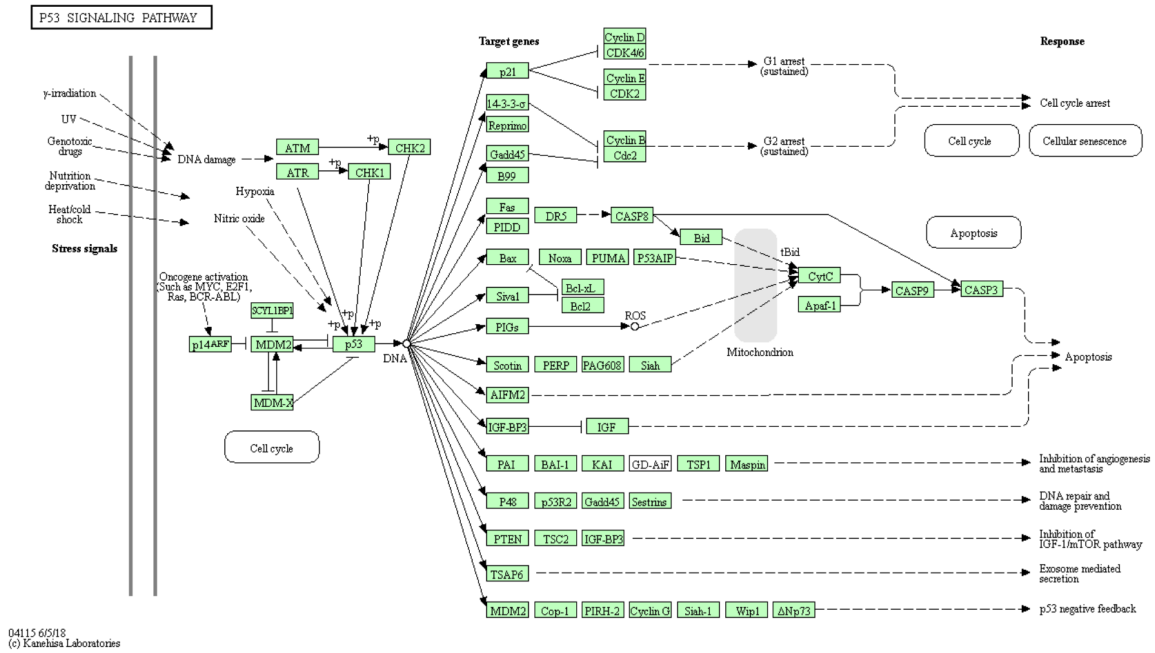

Figure 1.2: The p53 signaling pathway from the KEGG database. The green nodes in the network represent human proteins and genes, and the edges represent interactions. The double lines on the left of the pathway represent the cell membrane, giving partial information on the subcellular location of interactions in the pathway. Figure from KEGG database (Kanehisa and Goto, 2000).

While computational methods exist to compare the results of high-throughput experiments with pathways from databases to see if any pathway's members are over-represented (Reimand et al., 2019), pathways in curated databases are often incomplete and contain proteins or

genes that are not involved in a particular biological context (Köksal et al., 2018). Thus, it is often preferable to infer a customized subnetwork specific to an experimental dataset starting from all known protein interactions, referred to as the interactome. We refer to this problem as pathway reconstruction: using condition-specific input omic data to select nodes and edges from a generic background network that represent some process or cellular state. Pathway reconstruction differs from module detection (Choobdar et al., 2019), which divides a network into functional units or clusters. It is also distinct from network propagation (Cowen et al., 2017), which identifies relevant regions of a larger network but typically does not select specific edges within that region.

Formally, the pathway reconstruction problem takes the inputs:

- A network I consisting of a set of nodes I_N and a set of edges I_E. This will typically be a set of known interactions from a protein-protein interaction (PPI) database such as STRING (Szklarczyk et al., 2019).
- A set of nodes S, where $S \subseteq I_N$. From a biological perspective, this is a set of biological entities (genes, proteins, etc.) that are involved in the process of interest.

And produces as output:

- A network P where $P \subseteq I$. A subset of the set of known interaction and biological entities which create a network representing the process of interest.

Algorithms which solve this problem ideally seek to return a network P that maintains biological plausibility while most informatively representing the process of interest. However, algorithms typically actually maximize a topological criteria or perform some topological operation to find P. For instance, an extremely simple pathway construction method could return the minimum spanning tree of all nodes in S. However this tree would likely be biologically unrealistic, as it would include none of the community structure typically found in biological pathways (Ravasz et al., 2002). Pathway reconstruction algorithms approximate biological plausibility using a variety of computational methods, such as flow-based diffusion (Yeger-Lotem et al., 2009), statistical models (Cerami et al., 2010), or graph-theory

problems such as k-shortest paths (Ritz et al., 2016) or prize-collecting Steiner trees (Tuncbag et al., 2013).

1.2 Realizing the Potential of High-Throughput Biological Data Analyses

A number of barriers exist that prevent high-throughput biological data analyses from reaching their full potential. The accessibility and usability of these methods is often poor (Mangul et al., 2019; List et al., 2017), making it difficult to install and run them. This is especially true for researchers with less coding experience.

Additionally, computational workflows for interpreting high-throughput experiments require manual input and decision making (Kulkarni and Frommolt, 2017). While some level of careful decision making is necessary for determining the correct data preprocessing and algorithm choice, lack of guidance on these choices can be an additional barrier for researchers with less computational or statistical experience. Additionally, an overabundance of manual input in a computational pipeline can negatively impact reproducibility (Beaulieu-Jones and Greene, 2017). Manual steps add more room for error in both executing an experiment and accurately reporting on it. These steps are also places in a pipeline where bias can be introduced.

Improving the Usability of Exploratory Analysis Methods

Improving the usability and interpretability of exploratory analyses presents a number of unique computational challenges. Due to the open nature of exploratory analyses, there is rarely a clear overall objective to optimize. A classic example of this challenge is choosing the number of clusters in clustering methods such as k-means. There are a variety of methods dealing with the best way to choose the number of clusters in k-means, most of which involve some metric which measure the quality of the clusters being created. For instance, the popular elbow method involves examining the average distance between all points in all clusters as

the number of clusters increases, then manually choosing a perceived inflection point.

While there are properties of clusters which can be considered desirable and optimized, it is less clear what desirable properties more complex models like biological pathways should have. While a variety of heuristics have been used to guide pathway construction towards more desirable networks (Kedaigle and Fraenkel, 2018; Yeger-Lotem et al., 2009), there is a lack of a method for this task which clearly defines what a good pathway is independent of the method used to construct it.

This issue of the lack of a clear measure of goodness carries into the task of interpretation of biological pathways. Ideally, a method which aids the interpretation of a network would find areas of a network which are surprising or important. However, which areas meet these criteria are up to a biologist's interpretation and have no clear computational definition.

Adding Information to Exploratory Analyses

While improving network reconstruction methods is important, interpretation of exploratory pathway analyses remains difficult. Interpretation of a biological pathway can be split into two tasks: summarization and hypothesis generation. Summarization involves finding functional, physical, or other ontological patterns across the network. Methods which annotate pathways using GO terms (Bindea et al., 2009) or pathways from databases (Moriya et al., 2007) fall into this group and can aid biologists in understanding what is happening and what processes are involved in a pathway.

Hypothesis generation methods are typically looking to answer a more targeted question of a network, such as what is causing a certain cell state, what has changed due to a disease, or what is a possible drug target which would disrupt the pathway. Thus, as opposed to a broader functional summary, hypothesis generation looks for specific parts of the network which may be causal, vital, altered, or targetable. These parts of the network, often single nodes, are typically investigated in targeted followup experiments. Hypothesis generation can take a number of approaches to finding these areas of interest. Examples of hypothesis generation algorithms include using networks for active learning (Sverchkov and Craven,

2017), to find drug targets (Csermely et al., 2013), or to find driver mutations, mutations which likely caused a disease, in cancer (Horn et al., 2018).

One aspect of biological pathways which bridges both of these types of network interpretation is protein localization. Protein localization is the subcellular location of proteins during their interactions in a pathway, typically at the organelle level. Examples of the subcellular locations include the cell nucleus, the cell membrane, the cytoplasm, and the mitochondrion. Subcellular location of proteins or their interactions gives summary information about a pathway; it can help narrow the possible functional roles of the pathway and its members. For example, a protein which never enters the cell nucleus is highly unlikely to be directly involved in gene transcription, as that process is confined to the nucleus. In certain diseases, the location of some proteins can be altered. These include diseases such as Alzheimer's, ALS, Wilson disease, and various forms of cancer (Hung and Link, 2011). Therefore, knowing these non-standard localizations can give insight into how a disease is operating a cellular level, and which proteins may be key players.

Despite a number of methods that aim to generally predict protein localization, there has been little work in identifying localization of proteins within a pathway context. Prediction methods tend to treat the localization prediction problem as a multi-label classification, where proteins can have multiple locations across the different contexts in which they interact. Databases which store protein level localization information reflect this, listing multiple subcellular locations for proteins. It has been estimated that as many as 50% of proteins exist in multiple subcellular locations (Thul et al., 2017). This further highlights the need for a pathway-specific method for localization prediction, as it is unclear which of multiple localizations is relevant when using localization information to aid pathway interpretation.

Additionally, prediction methods focus on the possible unaltered localizations a protein can take, and thus cannot predict non-standard localizations of proteins in a diseased or otherwise abnormal state. When investigating diseases where abnormal localizations exist, prediction methods without pathway context and most database information become irrelevant. However considering the prediction within a pathway context will allow for the

consideration of abnormal localizations, as the abnormal localizations occur at a pathway level.

Making Data Analysis Education Practical

While it is important to improve tools used for bioinformatics analyses, these tools are not useful if biologists are unable to take advantage of them. Though experts in computational methods will likely always be needed to collaborate with biologists, increasing biologist understanding of computational methods will allow experimental biologists to perform some analyses without collaborator aid, increasing their accessibility. Additionally, while not all biologists will need to perform complex computational analyses, given that such analyses are increasing common it is important that all biologists are able to understand them and evaluate their quality. Biologists need to be able to assess the strength of computational evidence, and understand possible flaws in computational experimental design, in order to be full participants in modern biological research. Training in computational and bioinformatics methods has been found to have a long-standing positive impact on research in multiple areas including communication with colleagues, conducting better research, and validating results (Brazas and Ouellette, 2016).

One family of computational methods which has become increasingly popular is machine learning (Jones, 2019). Despite its popularity among computational researchers, machine learning remains elusive to experimental biologists, who form the majority of the life sciences research community, leaving powerful computational tools underappreciated and data generated in wet labs underexplored (Chicco, 2017). However, most machine learning courses and tutorials require substantial background knowledge in coding and mathematics, which many biologists may lack. On the other hand, bioinformatics workshops for biologists assume less coding experience, but participants are often taught to mechanically run through a software pipeline for certain tasks without learning the best practices in various stages of the workflow. Such an approach, though effective in the short term, can lead to error-prone data analysis, misinterpretation of results, and difficulty in adapting to other tasks in the long run of a

scientist's research effort. The community needs to explore novel educational frameworks in order to address these challenges in teaching machine learning to biologists.

We have created the ML4Bio (machine learning for biology) workshop to explore how to address this gap in computational training. Unlike traditional task-centric approaches, the educational objective of ML4Bio is to equip biologists with the proper mindset when it comes to applying machine learning in their research and the ability to critically analyze machine learning applications in their domain. Built around this core idea, the ML4Bio workshop prioritizes teaching machine learning literacy, that is, the right way to set up learning problems, how to reason about learning algorithms, and how to assess learned models.

1.3 Outline

This thesis is structured in the following manner. Chapter 2 introduces pathway parameter advising, a method for automating parameter selection in pathway reconstruction. This method helps remove the barriers of computational and methodological knowledge from utilizing pathway reconstruction analyses. In Chapter 3, we investigate the task of predicting subcellular localization within the context of a biological pathway. Ideally, creating a strong predictive model for predicting contextual subcellular localization would add an additional layer of information to pathway analyses, and allow inference of contextual localization when spatial proteomics experiments (Lundberg and Borner, 2019) are infeasible. Chapter 4 introduces the ML4Bio workshop, a workshop for teaching practical research skills in machine learning to biological researchers with minimal computational and mathematical background. The ML4Bio workshop focuses on approachability and practicality, teaching biologists the skills necessary for interpreting research that uses machine learning and finding opportunities for machine learning in their own research. We present results from a study on the workshop that give valuable insights into tailoring educational materials for an audience of active researchers. Finally, we conclude and propose possible future research directions in Chapter 5.

Chapter 2

Improving usability of biological network analyses

HIV-1 work presented in this chapter was performed in collaboration with members of the Ahlquist lab: Eunju Park, Paul Ahlquist, James Bruce, and Mark Horswill, mass spectrometry by members of the Coon lab: Alex Hebert, Gary Wilson, and Joshua Coon, and help and preliminary analyses by members of the Gitter lab: Nafisah Islam, Thevaa Chandereng, and Anthony Gitter. Pathway parameter advising work was performed in collaboration with Anthony Gitter (Magnano and Gitter, 2021).

2.1 Motivating work: HIV cell-cell contact

Background

As a motivating example of how exploratory analyses are used for biological discovery, and computational issues which can arise, I will present work done in exploring mechanisms of infection in human immunodeficiency virus type 1 (HIV-1). HIV-1 is a human pathogen which has claimed over 32[1] million lives. Increasing our understanding of HIV-1 can lead to new therapies and preventative measures, improving the lives of the over 35 million people

[1] https://www.who.int/en/news-room/fact-sheets/detail/hiv-aids

currently living with the disease. One aspect of HIV-1 infection where more study is needed is its multiple mechanisms of infecting cells. HIV-1 has two modes of infecting cells. It can either infect cells over long distances via cell-free virons (single viruses), or it can infect via direct cell-cell contact. It has been found *in vitro* HIV-1 can infect cells 100-1000 fold more efficiently using cell to cell contact than by cell-free transmission (Chen et al., 2007).

However, certain mechanisms of this cell to cell transmission are poorly understood. It is unclear how an infected cell responds to contact with an uninfected cell. The signaling pathways recruited in this response could be promising new drug targets.

With HIV-1 cell-cell contact as the process of interest, a high-throughput experiment was performed using mass spectrometry to investigate proteins active in HIV cells when coming into contact with uninfected cells. More specifically, mass spectrometry measures the overall abundance of proteins and their protein phosphorylation levels, which can be used to infer protein activation. An overview of the analysis is presented in Figure 2.1. These abundance levels, after filtering for changes deemed significant, were used as input into a pathway analysis.

Exploratory Network Analysis

Prize-Collecting Steiner Forest

We chose to perform network construction using the Prize-Collecting Steiner Forest (PCSF) algorithm. PCSF creates networks from sets of genes and a reference protein-protein interaction (PPI) network by finding connections in the PPI network between proteins of interest. In PCSF, proteins of interest are mapped to the reference PPI network and given prizes and all edges in the PPI network are given penalties. A set of nodes N is chosen, where each found connected component must contain at least one member of N. These can be thought of as the entry points of the found pathways. The forest $F = (V_F, E_F)$ with the highest score S is then found according to the following function:

$$\underset{F}{\operatorname{argmin}} \sum_{n \notin N_F} (\beta \cdot p(n) - \mu \cdot d(n)) + \sum_{e \in E_F} c(e) + \omega \cdot \kappa$$

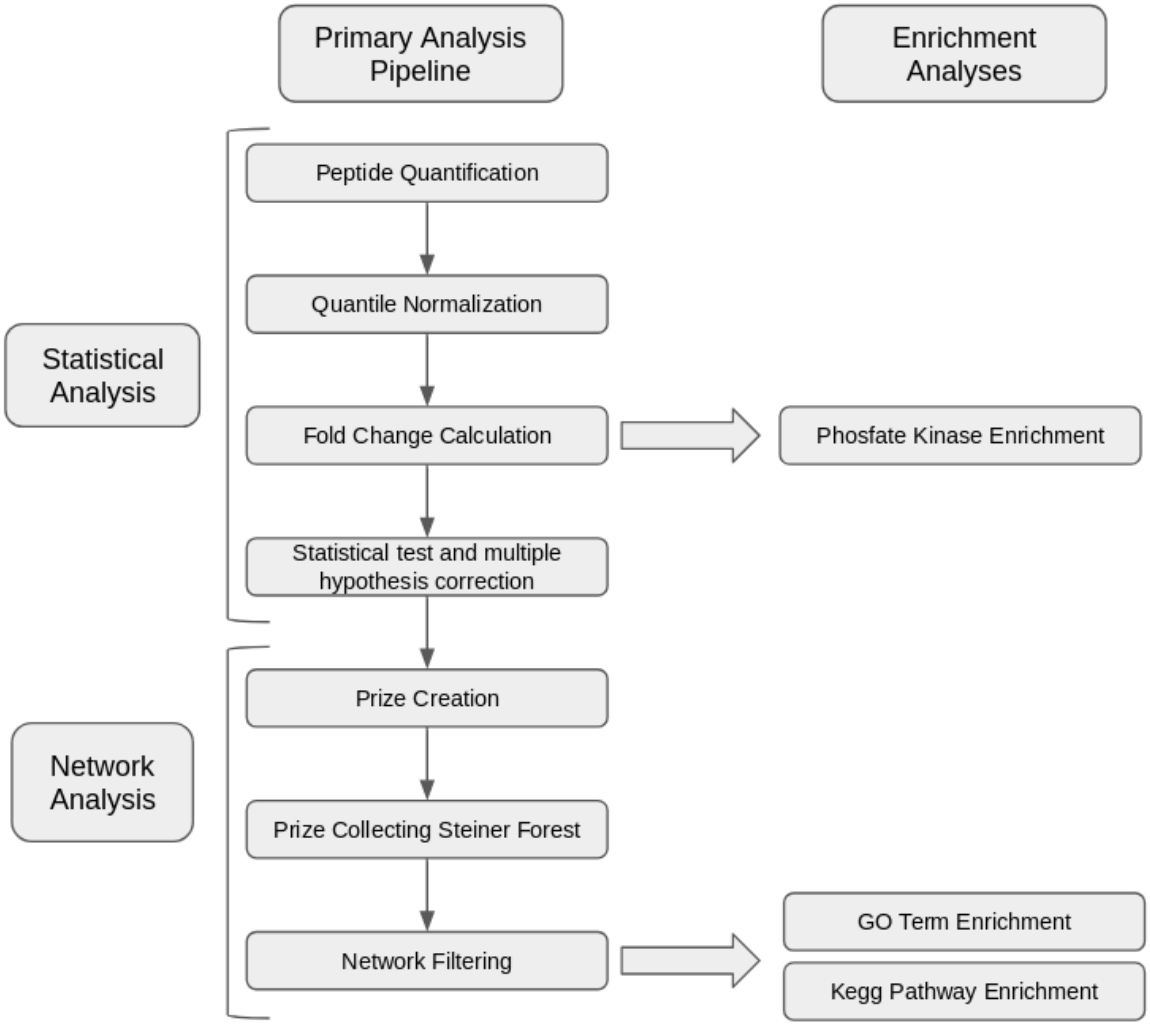

Figure 2.1: Overview of HIV data analysis pipeline.

where $p()$ is the positive prize for each node, $d()$ is a node's degree, $c()$ is the cost of each edge, and κ is the number of connected components in the pathway(the number of entry points used). The parameters β, μ, and ω are used to control the desired properties of the subnetwork. Choices of prizes, costs, and parameters are explained below. It can be seen that, as all edges are penalized, optimal solutions to PCSF will always be trees. We used the PCSF solver included in Omics Integrator (Tuncbag et al., 2016) which solves PCSF via a heuristic message passing algorithm (Bailly-Bechet et al., 2011). This implementation also includes a penalty for high degree nodes to lessen hub node bias and a penalty for the number of trees

in the final network.

Network Analysis Inputs

All significant proteins were used to generate protein prizes for PCSF. The magnitude of each protein's log transformed false discovery rate (FDR) was used directly as its prize. If a protein was found significant across both data types, the larger log transformed FDR was used. Sets of prizes were created for 0 to 5 minutes and for 0 to 60 minutes. This resulted in 1123 and 1050 prizes for the 0 to 5 minutes and for 0 to 60 minute networks, respectively.

Edge costs were computed from the iRefIndex PPI network (v13.0) (Razick et al., 2008) edge confidence scores. If multiple confidence scores were present for an interaction the highest score was used.

We created a set of entry proteins for PCSF from products of Env listed in the NCBI HIV database[2]. We selected all products which were lists as "binds" or "interacts with", totaling 462 proteins. These were chosen based on biologist input, as Env is proposed as the initiator of the cell-cell contact response.

Parameter Choice and Network Filtering

We ran PCSF over a sweep of different parameters to get a variety of networks. β was tested from 0 to 5 in increments of 0.5. κ was tested from 0 to 1.5 in increments of 0.1 and from 1.5 to 3 in increments of 0.5. μ was tested from 0 to 0.9 in increments of 0.015. This made for a total of 1463 networks for each timepoint. Calculations were run in parallel on HTCondor(Thain et al., 2005).

After performing a costly grid search over parameter values, there was no way to computationally determine which networks were viable for examination and further analysis. Therefore, we manually filtered out networks from the network cohort which we believed to be non-viable. This process was performed by looking through the network cohort with our biological collaborators, talking through which types of networks looked useful and which

[2] `https://www.ncbi.nlm.nih.gov/genome/viruses/retroviruses/hiv-1/interactions`

looked biologically implausible. We constructed a set of topological criteria specific to this set of networks, which would not be useful for other pathway analyses.

1. We removed any empty networks, leaving 1013 networks at 5 minutes and 287 at 60 minutes.

2. We removed networks which did not have a connected component with at least 25 vertices, leaving 954 networks at 5 minutes and 268 at 60 minutes.

3. We removed networks in which one vertex was connected to 10% or more of the network to eliminate networks dominated by a hub node, leaving 424 networks at 5 minutes and 210 at 60 minutes.

4. We removed networks which contained a branch of more than 8 vertices without any significantly changed proteins as neighbors as we did not consider these proteins sufficiently close to other significantly changed proteins in the network, leaving 223 networks at 5 minutes and 210 at 60 minutes.

An additional filter controlled for networks whose vertices were not at least half prize nodes, but this did not remove any of the networks. This resulted in 223 total networks in the 5 minute cohort and 210 networks in the 60 minute cohort. While these networks were all considered viable by our biological collaborators, and was done in as principled a manner as possible, it could still be the case that some bias was introduced to the network cohort by manually selecting these topological cut-offs. Furthermore, this time consuming process required both a computational expert and biological expert in that domain to determine which networks intuitively looked correct.

We then took the union of all filtered networks to create a final ensemble network for each time point. This network was both used for a GO-enrichment analysis and manually analyzed by our biological collaborators, which led to a hypothesis of 3 kinases playing key roles in the cell-cell contact response. More details and subsequent confirmatory experiments of these kinases can be found in Park (2018).

2.2 Parameter Tuning in Pathway Construction

Existing pathway reconstruction methods are based on diverse strategies such as combinatorial optimization problems (Tuncbag et al., 2016; Scott et al., 2006; Yosef et al., 2011), shortest paths (Ritz et al., 2016), enrichment analysis (Cerami et al., 2010), network flow (Basha et al., 2013; Goldberg and Tarjan, 1990; Komurov et al., 2010), and other graph theory algorithms. These methods also take in a variety of inputs. Some, such as the Prize-Collecting Steiner Forest (PCSF) algorithm (Tuncbag et al., 2016; Kedaigle and Fraenkel, 2018), accept scores for the biological entities of interest. Other methods, such as PathLinker (Ritz et al., 2016), require the inputs to be split into start-points (sources) and end-points (targets) for the pathway. Despite the different optimization strategies and inputs, pathway reconstruction algorithms almost always require the user to set parameters. Adjusting the parameters can produce pathways with drastically different topological properties and biological interpretations. For instance, in Figure 2.2 both pathways were created with the same PCSF algorithm and the same influenza host factor screen data (Section 2.5); they only differ in the parameters used. The pathway on the right is reasonably sized and can be interpreted and summarized for downstream analysis. The pathway on the left, however, includes over 7000 nodes and would impractical to interpret or analyze.

As highlighted by the HIV-1 cell-cell contact analysis, an open challenge is how to configure these critical pathway reconstruction parameters in a manner that is objective and applicable across diverse types of algorithms. Existing approaches tend to be informal and ad hoc, and most best practices in parameter tuning are not applicable to pathway reconstruction. The simplest way to choose a parameters would be to use default values. However, a single parameter setting cannot work for all datasets. The number of input proteins, genes, or metabolites varies based on the experiment, and the effects of input size can be unpredictable for different pathway reconstruction algorithms. For instance, if PathLinker is run with fixed parameters, increasing the number of source and target nodes will often result in a smaller final pathway, which is not necessarily what a user would intend. In addition, some methods are commonly run repeatedly and combined into an ensemble network (MacGilvray et al.,

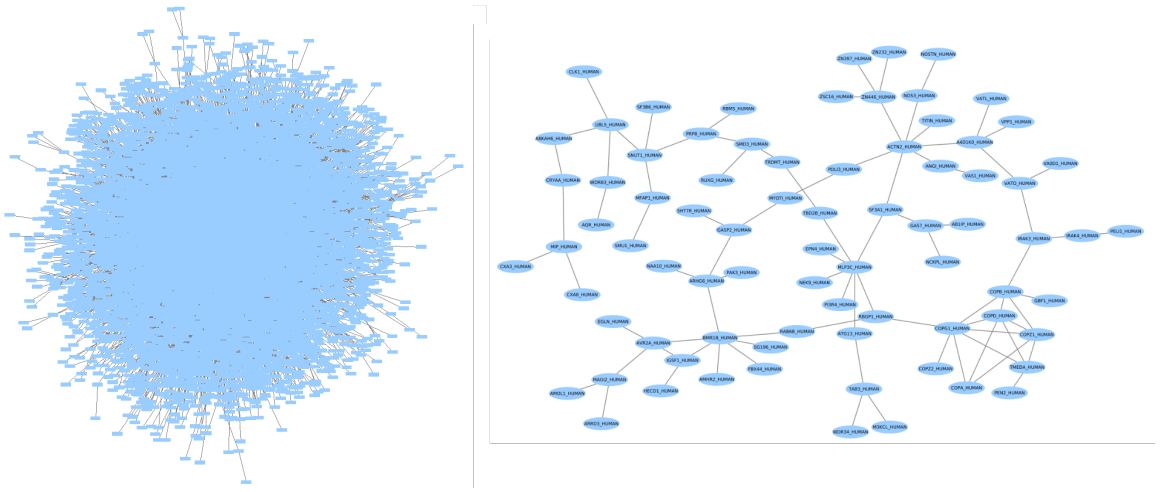

Figure 2.2: Influenza host factor pathways created using PCSF from RNA interference (RNAi) screens (Section 2.6), here showing the largest connected components from ensembling the bottom 100 ranked pathways (left) and the top 100 ranked pathways (right). The only difference in creating the networks was the range of PCSF parameter values.

2018; Budak et al., 2015; Khurana et al., 2017), which requires multiple parameter settings.

Parameter tuning methods from supervised learning are also a poor match. There is no ground truth for supervised parameter tuning, and unsupervised cross-validation is ineffective (Section 2.6). In addition, the objective functions of pathway reconstruction methods only approximate biologically meaningful graph topologies and typically have no probabilistic likelihood. Thus, their values cannot be compared between different parameter settings, and statistical model selection criteria such as the Akaike information criterion (Akaike, 1998) and the Bayesian information criterion (Schwarz, 1978) are not applicable. For instance, the PCSF objective function can be arbitrarily increased by changing the parameter values.

Given the lack of objective, quantitative methods for tuning, parameter settings are often chosen by manual inspection or informal heuristics. ResponseNet recommends choosing parameters that recover at least 30% of inputs while minimizing low confidence edges (Yeger-Lotem et al., 2009). PCSF recommends choosing pathways robust to small random input variation or matching the average degree of input nodes and non-input nodes (Kedaigle and Fraenkel, 2018). We show in Section 2.6 that these heuristics can perform poorly in practice.

Biologists often have intuition about which pathways are unrealistic or impractical for

downstream analysis, such as the 7000 node pathway in Figure 2.2 or subnetworks with unusual degree distributions. Pathway reconstruction is typically an exploratory analysis used to summarize the input data and generate hypotheses leading to further experiments. In this context, it is important to avoid implausible and uninterpretable pathway topologies. Therefore, parameter tuning should not focus on traditional notions of accuracy but instead formalize how *useful* generated pathways are to biologists. Our major contribution is providing a formal approach based on graph topology that quantifies this biological intuition and can be used to optimize pathway reconstruction in an objective manner.

One framework for finding optimal parameters in an uncertain setting is parameter advising (Kececioglu and DeBlasio, 2013; DeBlasio and Kececioglu, 2015, 2017), which was originally developed for multiple sequence alignment. Parameter advising can be used to adapt the parameter tuning framework in settings where no ground truth tuning set exists. However, parameter advising requires a means to estimate the accuracy of a model with a given set of parameters. Because pathway reconstruction is an exploratory analysis, there is no formal notion of accuracy. We overcome this limitation by leveraging background knowledge to create a ranking metric that prefers pathways in topological agreement with reference pathways. Our parameter tuning method, pathway parameter advising, uses the parameter advising framework in combination with a distance metric based on graphlet decomposition to measure similarity between generated pathways and pathways from curated databases. Pathway databases may be imperfect and incomplete, but they reflect models that the expert curators consider to be biologically plausible. Only measuring the topology of a generated network means that pathway parameter advising is also method agnostic. Pathway parameter advising can tune the parameters of any pathway reconstruction method.

2.3 The Pathway Parameter Advising Method

Pathway parameter advising is based on the parameter advising framework (DeBlasio and Kececioglu, 2015). A parameter advisor consists of two parts: a set of candidate parameter settings S and an accuracy estimator E. The parameter advisor evaluates each candidate

parameter setting in S using E to estimate the optimal parameter set. In order to adapt parameter advising to the pathway reconstruction task, we must choose a function E that can estimate the quality of a generated pathway. While we do not have a direct way to define what criteria an optimal solution satisfies, we do have access to pathways which match biologist intuition of what a biological pathway should look like. Curated pathway databases, such as the Kyoto Encyclopedia of Genes and Genomes (KEGG) (Kanehisa and Goto, 2000), Reactome (Fabregat et al., 2018), and NetPath (Kandasamy et al., 2010), contain pathways that have been compiled by biologists. Therefore, we can construct our estimator around these curated pathways. This leads to the key assumption of pathway parameter advising: *generated pathways more topologically similar to manually curated pathways are more useful to biologists*.

Our parameter tuning approach requires the inputs to the pathway generation algorithm, a set of candidate parameter settings, and a set of pathways from a curated pathway database. Pathway generation algorithms' input typically consists of an interactome, such as STRING (Szklarczyk et al., 2019), and a set of biological entities of interest, such as genes or proteins. We refer to the pathways created by the algorithm as *generated* pathways and the curated pathways as *reference* pathways. Pathway parameter advising uses a graphlet distance-based estimator E to score each generated pathway's similarity to the reference pathways. It uses these scores to return a ranking of the generated pathways (or their respective parameter settings).

Pathway parameter advising is designed to be method-agnostic. It can be run with any pathway reconstruction algorithm that generates pathways and has user-specified parameters. Currently, pathway parameter advising is designed to examine undirected graphs, and directed graphs are converted to be undirected while tuning.

Graphlet Decomposition

In order to topologically compare generated and reference pathways, we first decompose all pathways into their graphlet distributions. A graphlet is a subgraph of a particular size within a network. The concept of graphlets is similar to that of network motifs (Milo et al., 2002).

However, network motifs typically refer to graphlets that appear in a network significantly more often than expected by chance.

Original work with graphlets only considered connected graphlets to better capture local topology (Pržulj et al., 2004). However, we use both connected and disconnected subgraphs, thus allowing all possible combinations of nodes in a pathway to be considered a graphlet. This allows our parameter ranking to capture global topological properties such as pathway size in addition to local topology. One disadvantage of disconnected graphlet counts is that the counts of disconnected graphlets, such as the graphlet containing 4 unconnected nodes, grow at a much faster rate than those of connected graphlets in sparse networks. However, this disadvantage does not adversely affect our ranking metric (Section 2.6).

Pathway parameter advising uses the parallel graphlet decomposition library (Ahmed et al., 2015) to calculate counts of all graphlets up to size 4 in a pathway. This constitutes 17 possible graphlets (Figure 2.3). We convert these counts into frequencies and represent each pathway by a vector of 17 values between 0 and 1. This vector, referred to as the graphlet frequency distribution, summarizes the topological properties of a pathway, allowing us to quantify topological similarity.

Graphlets of up to size 4 were chosen to balance expressiveness and computational cost. Preliminary analyses found that Kegg pathways represented using an only size 3 graphlet decomposition clustered differently than Kegg pathways represented using an only size 4 graphlet decomposition. This suggested that the combination of multiple graphlet sizes would aid in the expressiveness of the final metric, as different sizes of graphlets create different topological embeddings. However, larger sized graphlets become computationally infeasible to calculate.

Distance Calculation

To calculate the topological distance between two pathways, we take the pairwise distance of their graphlet frequency distributions. For pathways G and H, we denote their frequencies of graphlet i as $F_i(G)$ and $F_i(H)$, scalars between 0 and 1. We then define the graphlet frequency

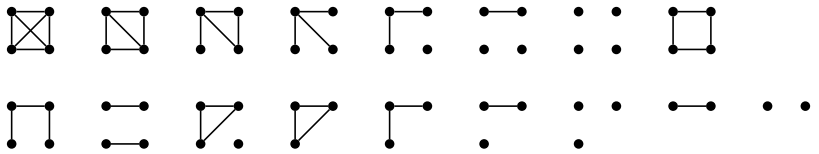

Figure 2.3: Pathways are decomposed into these 17 graphlets for graphlet frequency distance calculations.

distance $D(G, H)$ as

$$D(G,H) = \sum_{i=1}^{17} |F_i(G) - F_i(H)|$$

This differs from relative graphlet frequency distance, which log transforms and scales the raw graphlet counts (Pržulj et al., 2004). We considered other graphlet-based metrics such as a variation of relative graphlet frequency distance and graphlet correlation distance (Yaveroğlu et al., 2014) but found that they performed worse in our preliminary analyses (Figure 2.4).

Algorithm	Description	Parameters and Default Values	Range Tested
PathLinker (Ritz et al., 2016)	Connects receptors to transcriptional regulators via weighted k-shortest paths.	**k** (default=100) : Number of shortest paths.	1-1000
NetBox (Cerami et al., 2010)	Hierarchically constructs network using a hypergeometric test.	**p** (default=0.05) : p-value threshold for adding an edge	0-1
Prize-Collecting Steiner Forest (Tuncbag et al., 2013, 2016)	Assigns prizes to nodes and costs to edges; solves for highest scoring subnetwork with message passing algorithm.	β (default=1): relative weight of the node prizes versus edge costs	0-5
		ω (default=6): cost of adding an additional tree to the solution network	0-10
		μ (default=0): degree penalty	0-1
Minimum-Cost Flow (Goldberg and Tarjan, 1990)	Assigns edges costs and nodes as sources and targets of flow. Finds the network that satisfies flow constraints for the least cost.	**f** (default=10): amount of flow pushed through the network from sources to targets	1-50
		c (default=1): edge flow capacity	1-25

Table 2.1: The 4 pathway reconstruction methods and the parameters tuned for each.

Ranking Parameters

After calculating the graphlet frequency distribution for each generated and reference pathway, we can rank parameter settings by their mean graphlet frequency distance to the reference

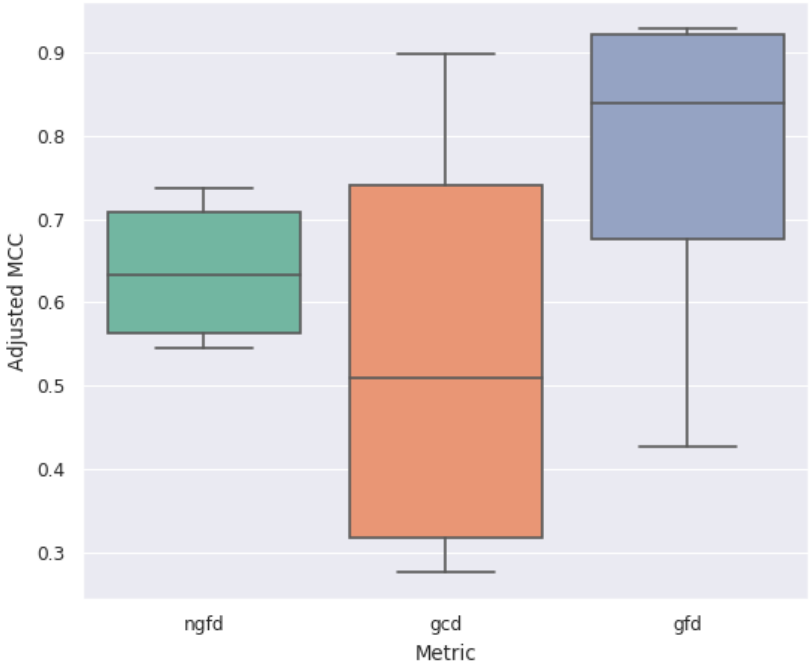

Figure 2.4: Examining the effect of different graphlet-based distance metrics on the adjusted MCC of pathway reconstruction. Reconstructions were performed on the validation pathways Wnt, TNF Alpha, and TGF Beta across the 4 pathway reconstruction algorithms. We examined 3 graphlet-based metrics for pathway parameter advising: normalized graphlet frequency distance (NGFD), graphlet correlation distance (GCD), and graphlet frequency distance (GFD). For NGFD, we wanted to explore a metric that takes advantage of all generated pathways being sub-networks of the same interactome. Thus, we normalized all graphlet frequencies by the corresponding graphlet's frequency in the interactome. We also explored GCD, which measures the correlation between connected graphlets in a pathway (Yaveroğlu et al., 2014). This creates a metric that is solely focused on local topology and has minimal information about pathway size or other global topological properties. Adjusted MCC was calculated the same way as in Section 2.6. GFD outperforms the other methods. One possible reason for GFD outperforming more complex methods like GCD is that GCD attempts to eliminate the signal of global topological properties such as size and give information on graphlets only. Some signal of global topology, however, likely aids in identifying which pathways are similar to reference pathways.

pathways to get E. When calculating this aggregate distance, we only consider the 20% closest reference pathways to the generated pathway. It is motivated by not requiring a generated pathway to be similar to every reference pathway, but instead similar to at least some reference

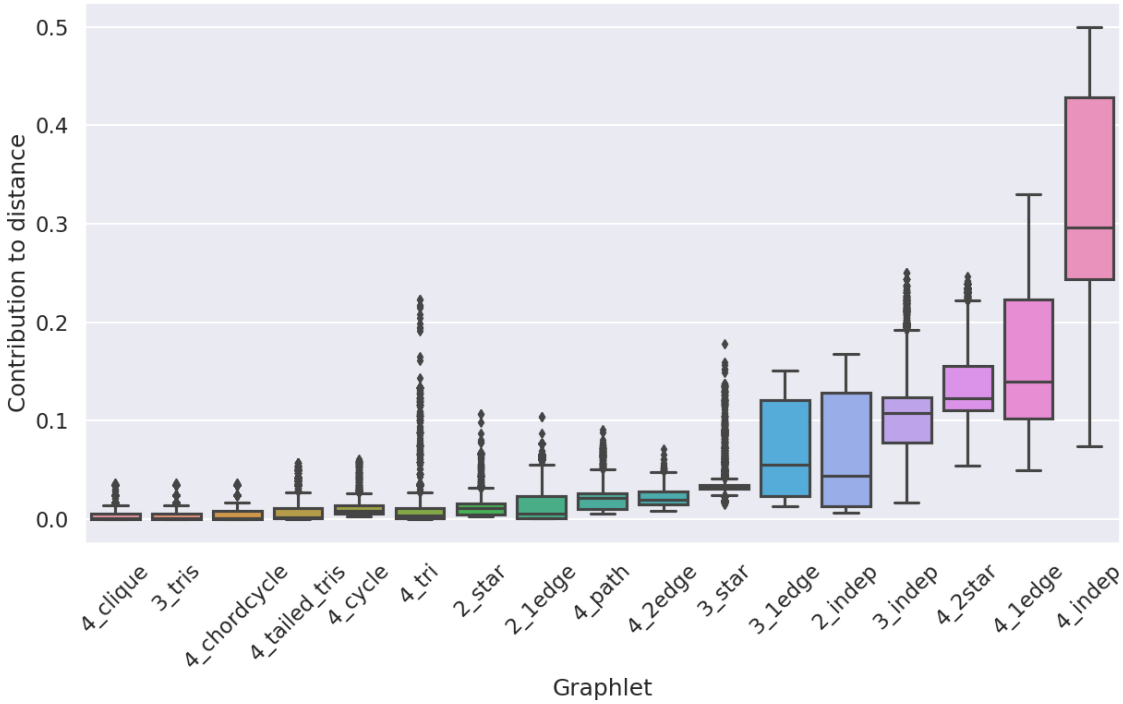

Figure 2.5: Contribution of each of the 17 graphlets to graphlet distance across the NetPath validation pathways Wnt, TNF alpha, and TGF beta and 4 pathway reconstruction algorithms. Graphlets are labeled as according to Ahmed et al. (2015). The 4 disconnected nodes graphlet, 4_indep, has a median contribution of about 30% of the GFD. Boxplots are filled in from the first to third quartiles with a line at the median, and whiskers representing 1.5 times the interquartile range.

pathways. Thus, a pathway G's score $E(G)$ is calculated as

$$E(G) = \sum_{r \in R_{top}} D(G, r)$$

where R_{top} is the set of the 20% closest reference pathways to G. The pathways, or equivalently the parameters used to generate those pathways, are sorted by $E(G)$ in descending order. Once the final ranking is created, the top generated pathway can be used for downstream analysis. Alternatively, for pathway reconstruction algorithms where it is common to create an aggregate network of multiple pathways, the top n pathways can be combined in an ensemble.

2.4 Pathway Reconstruction Methods

Pathway reconstruction algorithms were chosen to have a wide range of methodologies, from NetBox's statistical test to PathLinker's weighted shortest paths algorithm. We used the following 4 methods for our implausible pathway detection and reconstruction experiments. These methods and the parameters tested are summarized in Table 2.1.

PathLinker: PathLinker (Ritz et al., 2016) constructs pathways based on a weighted k shortest paths algorithm. It finds paths between sets of receptors and transcriptional regulators, similar to the source and target nodes in minimum-cost flow. It is controlled by the parameter k, which defines how many paths to return in the final network. We varied k from 1 to 1000 in increments of 1. We used PathLinker version 1.1 for all analyses.

NetBox: NetBox (Cerami et al., 2010) hierarchically constructs networks from a set of input nodes. At each iteration, it searches for nodes that connect two nodes in the current network. It then chooses to add these linker nodes to the network based on the results of a hypergeometric statistical test comparing the degree of the linker node to how many nodes in the pathway it connects. NetBox is controlled by the parameter p, a p-value cut-off, which sets the threshold for whether or not linker nodes should be included. We varied p from 0 to 1 on a log scale from 1×10^{-30} to 1 in increments of half an order of magnitude, giving a total of 60 steps. We used NetBox version 1.0 for all analyses.

Prize-Collecting Steiner Forest: In PCSF (Tuncbag et al., 2013, 2016; Kedaigle and Fraenkel, 2018), nodes are assigned prizes and edges are given costs. More details of PCSF are stated about in Section 2.1. We varied 3 PCSF parameters: β, which controls the relative weight of the node prizes versus edge costs was varied from 0 to 5 in increments of 0.5; μ, which affects the penalty for high-degree nodes was varied from 0 to 1 in increments of 0.1; and ω, which controls the cost of adding an additional tree to the solution network was varied from 0 to 10 in increments of 1. We used version 1.3 of the msgsteiner message-passing algorithm and version $0.3.1$ of OmicsIntegrator for all analyses.

Minimum-Cost Flow: The minimum cost flow problem assigns certain nodes in the network to be "sources" and others to be "targets". Edges, which transport the flow from node to

node, have a cost associated with using them and a capacity of how much flow they can hold. The solution is the network that satisfies the flow requirements of the source and target nodes while using lowest cost in edges (Goldberg and Tarjan, 1990). We implemented a version of min-cost flow using the solver provided in Google's OR-Tools[3], which solves the min-cost flow problem using the algorithm outlined in Bünnagel et al. (1998). This is a generic version of the algorithm used in ResponseNet (Basha et al., 2013). Two parameters control the min-cost flow solution: the total flow through the network, which we vary from 1 to 50 in increments of 1, and the edge flow capacity, which we vary from 1 to 25 in increments of 1. We used Google's OR-Tools version $7.1.6720$ for all analyses.

2.5 Experimental Setup

Parameter Selection Methods

We consider the following parameter selection strategies from the literature to evaluate our pathway parameter advising approach:

Cross-validation: Cross-validation (CV) involves splitting the input data into training and testing sets multiple times for each parameter setting. A method is then fit or trained on each training set and evaluated on each respective testing set. In this problem setting, we do not have external ground truth with which to evaluate the predictions on test set data. Instead, we performed 5-fold CV on subsets of the input data, choosing the parameter values that generate a pathway from the training set nodes that recover the highest proportion of the test set nodes.

ResponseNet recommendation: We also tested a parameter selection heuristic used by ResponseNet (Yeger-Lotem et al., 2009). The criterion is to select parameters that result in a pathway that includes at least 30% of the input set, while having the lowest proportion of low confidence edges. We extend this to rank the pathways that do include 30% of the inputs by their proportion of low confidence edges, followed by the pathways that include less than 30% of the inputs to form a full ranking.

[3] `https://developers.google.com/optimization/flow/mincostflow`

Randomization stability: As suggested by Kedaigle and Fraenkel (2018), for PCSF we can also rank pathways by their robustness, as measured by how often nodes appeared in multiple runs with small random perturbations to the scores on the input nodes. Ranking pathways this way was only available in PCSF. Although it could be adapted to other pathway reconstruction methods, we decided to use it only in the method for which it was directly implemented.

Datasets

Interactomes

For both PathLinker and NetBox, we used the interactome included as a part of their software packages. For PCSF and min-cost flow, we used an interactome from Köksal et al. (2018) that merged protein interactions from the iRefIndex database v13 (Razick et al., 2008) and kinase-substrate interactions from PhosphoSitePlus (Hornbeck et al., 2014). This resulted in a network with 161901 weighted edges.

Pathway databases

All parameter tuning was performed with Reactome as the set of reference pathways. Reactome (Fabregat et al., 2018) is a database of manually curated pathways, including 2287 human pathways. Reactome is open-source, where all contributions must provide literature evidence and are reviewed by an external domain expert before being added. Pathways smaller than 15 nodes were excluded as too small for meaningful interpretation.

The implausible pathway detection and reconstruction experiments were performed on pathways from the NetPath database. NetPath is a collection of 36 manually curated human signal transduction pathways (Kandasamy et al., 2010). We used 15 NetPath pathways that contain at least 1 receptor and transcriptional regulator and are sufficiently connected, as described by Ritz et al. (2016). We designated 3 of these NetPath pathways as validation pathways: Wnt, TGF Beta, and TNF Alpha. Validation pathways were used to guide the choice of distance measure, and the remaining 12 pathways were reserved for a quantitative

evaluation. We sampled the NetPath pathways in different ways for each pathway reconstruction algorithm to provide inputs in their expected formats. PCSF and NetBox do not require sources and targets, so we randomly sampled 30% of the pathway nodes as input. We also assigned each input a random prize sampled uniformly between 0 and 5 for PCSF. For PathLinker and min-cost flow, which require sources and targets, we selected all transcription factors and receptors for each pathway as outlined by Ritz et al. (2016).

Influenza host factors

Influenza host factor data was gathered from a meta-analysis of 8 RNAi studies (Tripathi et al., 2015). The meta-analysis used the raw RNAi screen data to calculate a consolidated Z score for a total of 1257 host factor genes.

Implausible Network Criteria

In order to examine the ability of pathway parameter advising to avoid parameter settings that lead to impractical pathways, we created topological criteria that we use to define pathways as plausible or implausible. We use these criteria as a heuristic to label pathways as positive (plausible) or negative (implausible). The labels enable us to evaluate pathway rankings as a classification problem, determining if a method can correctly classify pathways as plausible or implausible. These criteria are based on previous analyses of biological network, and are as follows:

Size: We allowed pathways that had between 10 and 1000 nodes. Pathways whose size was outside this range are not practical for hypothesis generation and downstream analysis.

Hub node dependence: A common issue with pathway reconstruction algorithm is an over-reliance on high-degree or hub nodes, which can create networks consisting almost entirely of a single node and its neighbors with few to no connections between those neighbors (Kedaigle and Fraenkel, 2018). We score this using the ratio of the degree of the highest degree node to the average node degree of the entire pathway. If the highest degree node has over 20 times more edges than the average node in the pathway, we consider that pathway implausible.

Clustering coefficient: Biological networks have been found to have clustering or community structure that is hierarchical (Ravasz et al., 2002); communities within the network exist at multiple scales and are often nested within each other. Thus, it would be reasonable to expect a plausible biological pathway to have at least a moderate level of community structure. We calculate the average clustering coefficient of all nodes in the pathway, a common metric for measuring community structure (Barabási and Oltvai, 2004). The clustering coefficient of a node is the proportion of its neighbors that are also neighbors of each other. This can be averaged over all nodes in the pathway as a measure of the overall level of clustering. We require pathways to have an average clustering coefficient of at least 0.05, as we expect at least some small level of clustering to exist. Because this requirement eliminated all PCSF pathways in 25% of tasks, when evaluating PCSF we excluded this metric in all cases.

Assortativity: A network's level of assortative mixing is defined as the tendency of high degree nodes to be connected to other high degree nodes. Biological networks have been found to be generally dissasortative, meaning that high degree nodes tend to be connected to low degree nodes (Newman, 2002; Albert et al., 2011). Assortativity is measured between -1 and 1, where assortative networks have positive values and dissasortative networks have negative values. This value can be viewed as the correlation between a node's degree and its neighbor's degrees within the pathway. We consider networks with assortativity between -1 and 0.1 plausible to allow for some leeway in pathways being slightly assortative.

We selected these criteria based on attributes it would be reasonable to expect a biological pathway to have, with values supported by the literature where possible. These thresholds were not influenced by the graph topologies in the reference pathway database in order to minimize circularity between the reference pathway-based rankings and the plausibility criteria used to evaluate those rankings.

While the criteria for defining a plausible network are useful for comparing networks created by the same method with different parameter settings, they should not be considered as a metric for comparing pathways across pathway reconstruction methods. Different pathway reconstruction methods are able to use different sources of information and have

complex strengths and weaknesses beyond the local topologies they return. For instance, NetBox, which had the highest proportion of plausible pathways, cannot take into account information such as edge confidence or scores on proteins of interest that other methods such as PCSF can. Although they are useful for evaluation, these properties are a binary way to define as a baseline if a pathway is reasonable or unreasonable. Thus, they could not be used to rank pathway reconstruction parameters themselves.

Criteria Grid Search

In order to make sure that our experimental results are not overly sensitive to the specific choice of cut-offs for pathway plausibility, we tested other cut-offs in a grid search. We varied the maximum network size cut-off from 200 to 2000, the hub node dependence measure from 5 to 50, the clustering coefficient cut-off from 0.0 to 0.1, and the assortativity cut-off from -0.5 to 0.5. Each range was divided into 10 intervals, for a total of 10000 sets of plausibililty cut-offs. Figures 2.9 and 2.6 evaluate the pathway reconstruction methods across these different thresholds.

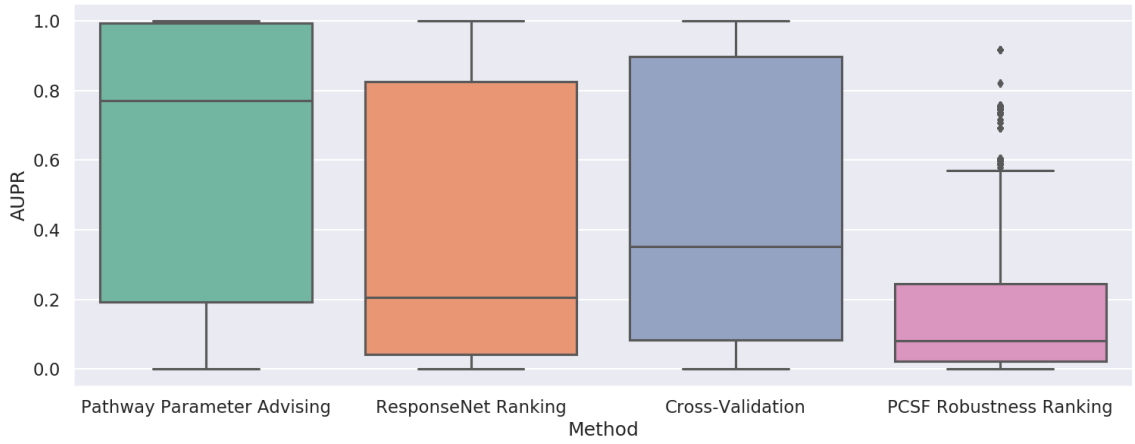

Figure 2.6: Performance of parameter selection methods on avoiding implausible networks aggregated for all considered 10000 plausibility criteria. AUPR is shown for all 12 NetPath pathways and 4 pathway reconstruction methods.

Matthew's Correlation Coefficient

In the NetPath evaluation, we used Matthew's Correlation Coefficient (MCC) to quantify the quality of generated pathways (Matthews, 1975). MCC is a metric that ranges between -1 and 1, where 1 indicates a perfect binary classification and -1 indicates a complete inverse classification. It can be viewed as the correlation between the predicted and true labels in a classification task. MCC has been shown to be well suited to evaluate classification in imbalanced settings (Boughorbel et al., 2017). When calculating performance, we consider all edges in a NetPath pathway as the positive set and all other edges as the negative net. MCC is defined as

$$MCC = \frac{TP \cdot TN - FP \cdot FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

where TP is the number of true positives, FP is the number of false positives, TN is the number of true negatives, and FN is the number of false negatives. When comparing MCC values of multiple pathways and methods, we normalized MCC values by the best possible MCC among all tested parameter values for that pathway and method. We refer to this value as the adjusted MCC.

Enrichment Analyses

GO (The Gene Ontology Consortium, 2018) and KEGG pathway (Kanehisa and Goto, 2000) enrichment was carried out with DAVID v6.7 (Huang et al., 2008). Enrichment was performed using GO biological process terms and all KEGG pathways. Thresholds for term inclusion were set to a count of 2 and an EASE score of 0.1.

Pathway parameter advising implementation

A Python implementation of pathway parameter advising is available at `https://github.com/gitter-lab/pathway-parameter-advising` under the MIT license. While the v0.1.0 release of the pathway parameter advising software supports Python v3.6, the original com-

putations for the results here used Python v2.7.16 and Anaconda version v2019.03. The following package versions were used: pandas v0.24.2, networkx v2.2, numpy v1.16.2, matplotlib v2.2.3, and seaborn v0.9.0. The Parallel Graphlet Decomposition library was pulled from GitHub on April 30, 2019.

2.6 Results

Because pathway reconstruction is an exploratory analysis and our goal is to maximize downstream biological utility, which cannot be directly quantified, we resort to multiple indirect approaches to evaluate pathway parameter advising. We first ensure that the graphlet distance ranking metric has desirable properties. It is not overly sensitive to the disconnected graphlets and shows that reference pathways are similar to one another. We then use the literature to define topological graph properties that make a candidate pathway biologically implausible and show that pathway parameter advising can optimize pathway reconstruction algorithms to avoid implausible pathways. Next, we demonstrate that we can improve the reconstruction of NetPath pathways by comparing predicted pathway edges with the NetPath ground truth. Finally, we show how pathway parameter advising can be applied in practice and generate an influenza host factor pathway from RNAi screens.

Evaluating the Ranking Metric

In order to determine the validity of our pathway ranking metric, we first examined the overall distribution of graphlet distances ($E(G)$ in Section 2.3) across all pathway reconstruction methods and Reactome pathways with added noise. We added noise to pathways by removing a percentage of all edges in the pathway, then adding back that many edges that did not appear in the original pathway. We also calculated the graphlet distance between each Reactome pathway and all of the other Reactome pathways. Figure 2.7 shows these distributions of graphlet distances. Reactome pathways had a lower mean distance than any set of reconstructed pathways, confirming that our metric ranks curated pathways over reconstructed pathways.

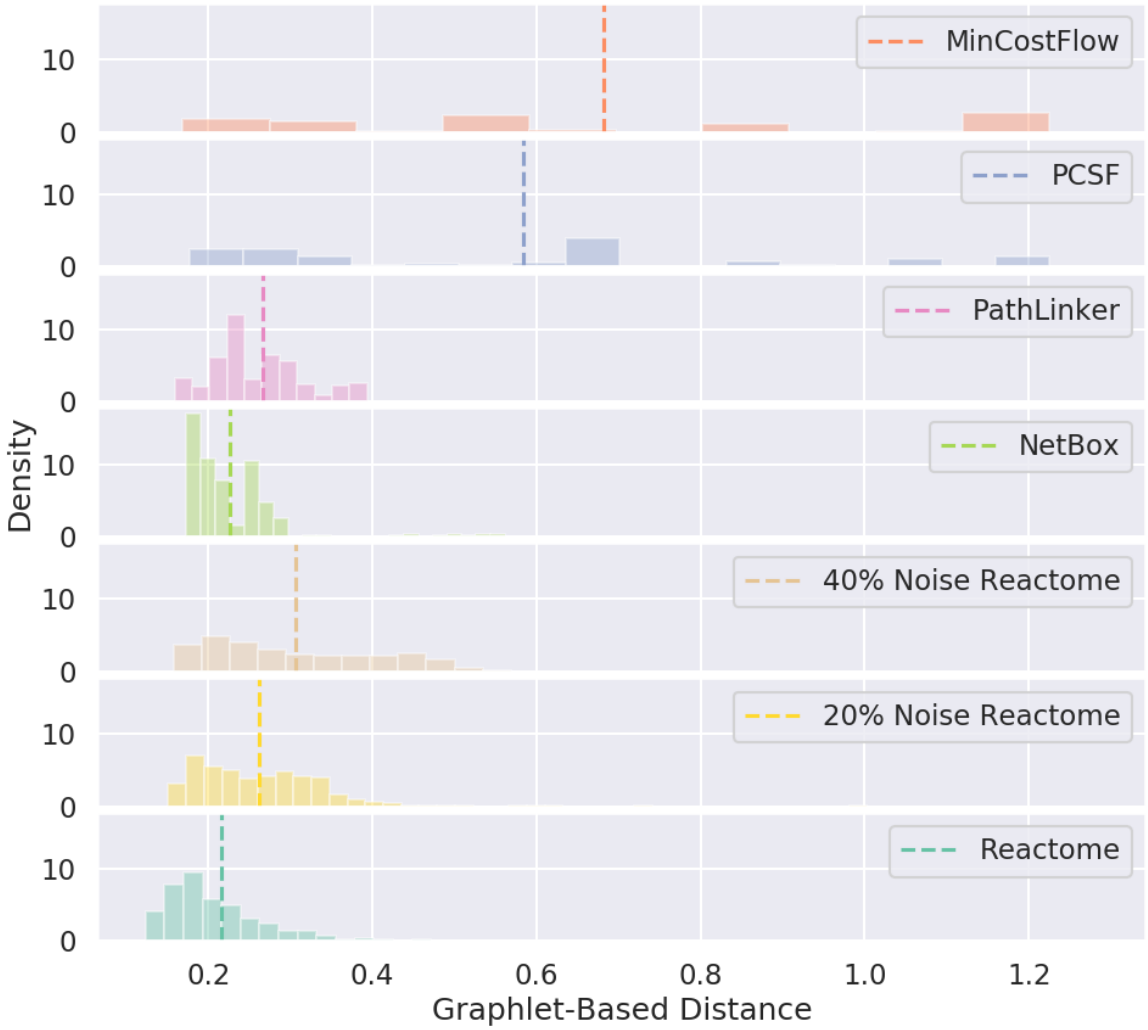

Figure 2.7: Distribution of graphlet-based distances ($E(G)$) for all reconstructed pathways, Reactome pathways, and Reactome pathways with added noise. Reactome pathways were excluded from their own distance calculation. Vertical dashed lines show the mean graphlet distance. Reactome pathways were found to have the lowest mean graphlet distance, confirming that our method ranks curated pathways over reconstructed pathways.

In order to examine the possibility that unconnected graphlets dominate the ranking metric, we first examined the breakdown of total graphlet distance by each of the individual graphlets in Figure 2.3. Figure 2.5 shows the contribution of each graphlet to the ranking metric. While a single graphlet, the 4 unconnected nodes graphlet, does have the largest contribution to graphlet distance, its median contribution to the ranking metric's value is only 30%. Because biological pathways tend to be sparse, this graphlet count scales with pathway

size. Thus, pathway size may contribute to approximately 30% of the ranking metric.

To confirm that this graphlet was not negatively impacting our ranking metric, we also examined its contribution to implausible pathway detection performance as described in Section 2.6. We used the NetPath pathways Wnt, TNF alpha, and TGF beta as validation pathways to evaluate different graphlet metrics and develop pathway parameter advising. These 3 validation pathways are excluded from all aggregate results. On these 3 held aside NetPath pathways, we found that using the 4 unconnected nodes graphlet alone resulted in poor performance and that its inclusion moderately boosts performance.

Implausible Pathway Detection

In order to evaluate pathway parameter advising, we considered its ability to avoid implausible networks. While it is difficult to define a single best pathway in the context of an exploratory analysis, some pathways are clearly biologically unrealistic, infeasible to analyze, or not useful for downstream analysis. Thus, pathway parameter advising should consistently rank parameter settings that lead to plausible networks above those that lead to implausible networks.

We applied the four pathway reconstruction methods to sampled NetPath pathways (Section 2.5) to reconstruct the 15 pathways, following the evaluation approach used by PathLinker (Ritz et al., 2016). This resulted in 60 parameter tuning tasks across the 15 pathways and 4 pathway reconstruction methods.

In these experiments pathways considered plausible, as defined in Section 2.5, are treated as the positive set. Pathways defined as implausible are the negative set. We then used precision-recall (PR) curves to evaluate how well pathway parameter advising and alternative parameter selection strategies distinguish plausible from implausible networks. Parameter selection methods that rank plausible networks above implausible networks will have a higher area under the PR curve (AUPR). Individual PR curves can be found at Magnano and Gitter (2021).

Figure 2.8 shows the distribution of AUPRs across the 4 pathway reconstruction methods.

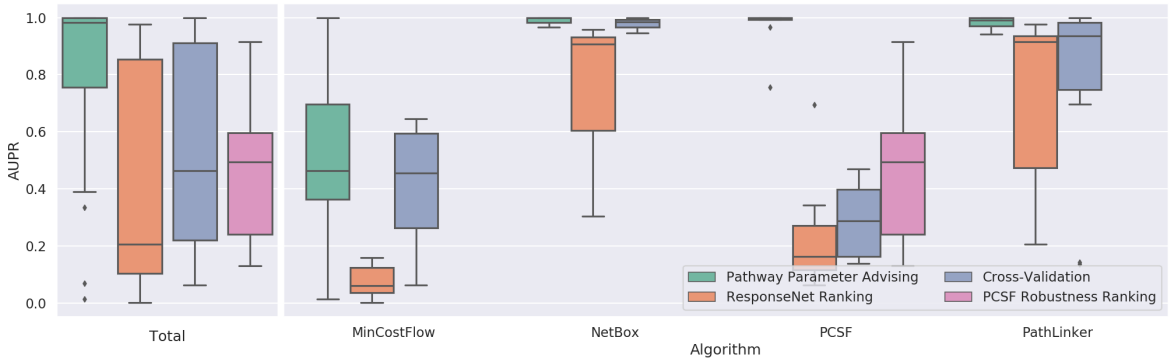

Figure 2.8: Performance of parameter selection methods on avoiding implausible networks. Boxplots represent the distributions of the AUPRs aggregated for 4 pathway reconstruction methods and 12 test pathway reconstructions from the NetPath database. Degenerate cases where all or no pathways met the plausibility criteria are excluded. Full results, including the 3 validation pathways and degenerate cases, can be found in Magnano and Gitter (2021)

Different methods also had varying proportions of networks identified as plausible, with min-cost flow having the lowest mean proportion at 11% and NetBox with the highest at 89%. Among the 12 test pathways, pathway parameter advising has the highest median AUPR for each pathway reconstruction method.

Of the 36 cases where AUPRs could be compared (both plausible and implausible pathways were present), pathway parameter advising had the highest AUPR in 30. Cross-validation had the highest AUPR in the other 6. The impact of the choice of parameter ranking strategy is most stark for PCSF, where graphlet frequency distance has perfect AUPR in almost all pathways and the other approaches struggle. Not only did pathway parameter advising have the highest median AUPR, but its performance was the most consistent; it had the lowest variance in AUPR across all tasks.

In order to make sure that performance was not overly influenced by our specific choice of criteria for plausible and implausible networks, we then varied the plausibility cut-off for each of the 4 topological properties we considered in Section 2.5 in a grid search, resulting in a total of 10000 configurations. Figure 2.9 shows the AUPR of each parameter ranking method as the plausibility cut-off for each topological feature is varied, and aggregate AUPR values are shown in Figure 2.6. Pathway parameter advising outperforms all other parameter ranking methods on average at each cut-off value.

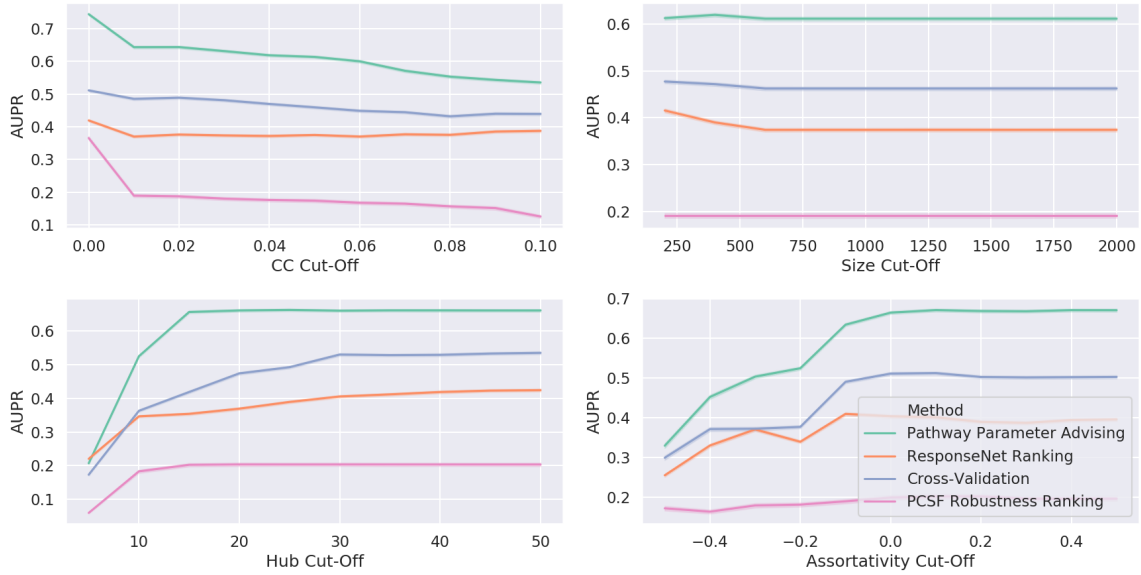

Figure 2.9: Performance of parameter selection methods on avoiding implausible networks as the cut-off for plausibility is varied across different topological features – clustering coefficient, pathway size, hub node dependence, and assortativity – as described in Section 2.5. Lines show mean AUPR over the varied cut-offs for the other 3 topological features for all 12 NetPath pathways and 4 pathway reconstruction methods.

NetPath Pathway Reconstruction

Having achieved our primarily goal of accurately prioritizing parameters that generate plausible pathways, we also evaluated the quality of the pathway reconstructions themselves. We compared pathway parameter advising to the alternative ranking methods and the default parameters. In these experiments we define all pathway edges in the NetPath pathway as a positive set and all other edges as a negative set. We then compared the ability of pathway parameter advising and other parameter ranking methods to promote reconstructed pathways that closely resemble their NetPath equivalent.

Figure 2.10 (left) shows the adjusted MCCs of all 48 pathway reconstruction tasks. While pathway parameter advising has the highest median adjusted MCC, the parameter selection method has less impact on MCC than it did on pathway plausibility (Figure 2.8). When stratified by pathway reconstruction algorithm, pathway parameter advising has the highest median adjusted MCC for PCSF and PathLinker. CV has the highest adjusted MCC in min-

cost flow and NetBox. Of the 48 reconstruction tasks, pathway parameter advising had the highest median adjusted MCC 21 times, while CV had 13, default parameters had 9, and the ResponseNet ranking had 8, including 3 cases where 2 methods tied.

Figure 2.10 (right) shows the distribution of best possible MCCs for all reconstruction tasks, including the 3 validation pathways. The best possible MCC was greater than 0.3 in only 4 of the 60 cases, and it was never greater than 0.4. Given these low unadjusted MCC values and the overall objective of our study, the implausible pathway detection experiment is a better indicator of method performance.

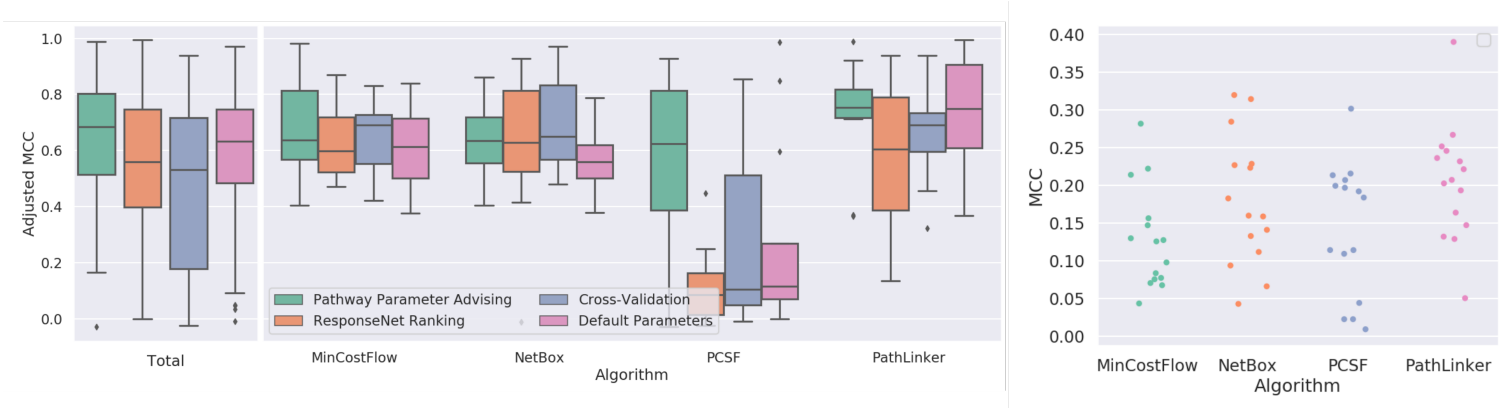

Figure 2.10: **Left:** Adjusted MCC of parameter selection methods on reconstructing 12 test pathways from the NetPath database across 4 pathway reconstruction methods. MCCs were normalized to the highest possible MCC within a given pathway reconstruction method and pathway. **Right:** The highest possible MCC of pathway reconstruction in 60 parameter sweeps across 4 pathway reconstruction methods and 15 NetPath pathways (validation and test). The MCC values are generally low, reflecting low overlap between the predicted and NetPath pathway edges.

Influenza Host Factor Pathway Reconstruction

To demonstrate how pathway parameter advising can guide the biological interpretation of omic data, we reconstructed an influenza host factor pathway. Our aim was to create a pathway that represents aspects of influenza's infectious activities and could lead to the discovery of new host factors or host factor regulators. We created an influenza host factor network using the 1257 host factors from a meta-analysis of 8 RNAi screens (Tripathi et al., 2015). These host factors were given as input to PCSF, using the same range of possible

parameter settings as in the other experiments. We used the magnitude of the consolidated Z scores given in the meta-analysis as node scores (see Section 2.5).

After creating the candidate host factor pathways, we ranked the parameter settings using pathway parameter advising. Figure 2.11 (left) shows the PR curve of different parameter ranking methods' ability to avoid implausible networks. Pathway parameter advising ranked the pathways almost perfectly, with an AUPR of 0.96, while other parameter selection methods had more difficulty separating implausible from plausible networks. CV and the ReponseNet rankings performed worse than the random baseline. This demonstrates that pathway parameter advising performs well not only on simulated data from NetPath but also on data aggregated from real high-throughput experiments.

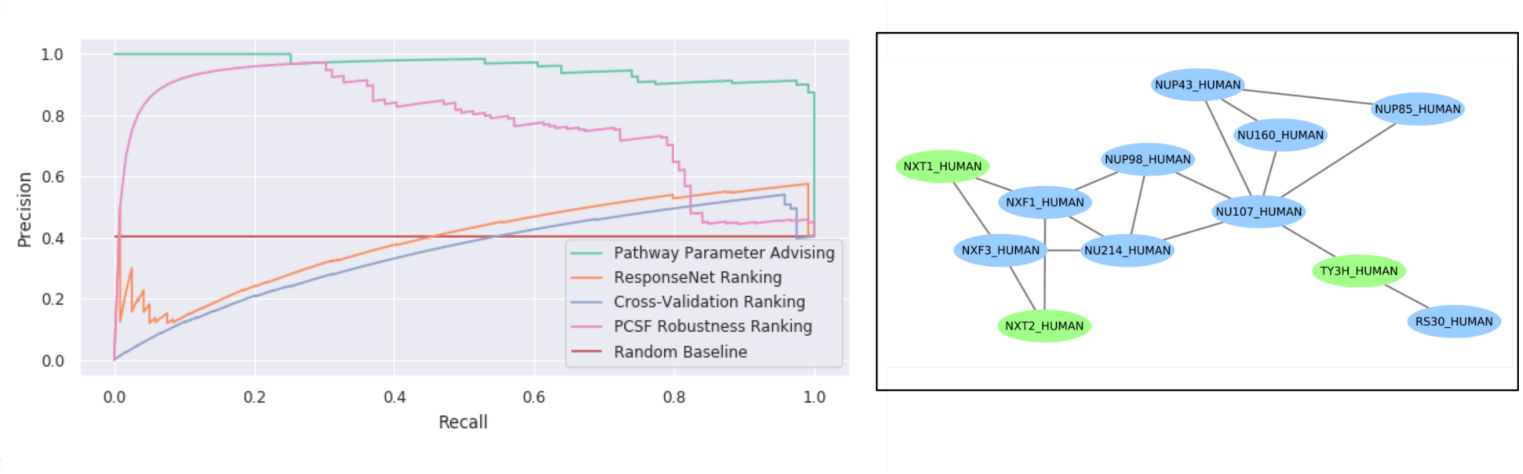

Figure 2.11: **Left**: Precision-recall curve for implausible networks in PCSF influenza host factor network construction. **Right**: A component of the influenza host factor ensemble pathway created from the top 50 PCSF parameter settings ranked by pathway parameter advising. This component represents 12 of the 86 total nodes in the pathway (Figure 2.12). Host factor nodes provided as input are shown in blue, while green nodes are "Steiner" nodes that PCSF predicts to connect the host factors.

We also created three ensemble networks from the resultant pathways from the top, middle, and bottom 50 parameter settings ranked by pathway parameter advising. As discussed in Section 2.2, ensembling networks is a common way to use PCSF. Thus, we expect the top 50 ensemble pathway to be best for downstream analysis and interpreting the input host factor data. Of the 3 constructed ensemble pathways, the pathway made from

the 50 highest ranked parameters contains 86 nodes. The middle ranked and low ranked ensemble pathways have 7337 and 15 nodes, respectively (Figure 2.12). The middle ranked pathway is too large to interpret and does not provide meaningful new insights into the relationships among host factors. The low ranked pathway is too small to illuminate new biological hypotheses. The top ranked pathway, however, is large enough for meaningful enrichment and downstream analyses while remaining small enough to be feasible.

We then performed a gene set enrichment analysis on the top ensemble pathway using DAVID (Huang et al., 2008) (Section 2.5). We tested both Gene Ontology (GO) biological process terms and KEGG pathway enrichment. Full enrichment results can be found in Magnano and Gitter (2021). The top 2 KEGG pathways enriched were RNA transport and influenza A. The influenza A pathway being enriched is a confirmation that our ensemble pathway is representing influenza processes well.

The RNA transport pathway enrichment captures one unique aspect of influenza A. It replicates within the nucleus, so it has complex processes for transporting viral RNA in and out of the nucleus (Dou et al., 2018). Similar concepts are observed in the top 2 enriched GO terms, mRNA and tRNA export from nucleus. We also see other general viral GO terms, which confirm the top ranked pathway's representation of influenza, such as viral transcription and intracellular transport of virus.

Figure 2.11 (right) shows one connected component representing 12 of the 86 nodes from the top ranked ensemble pathway. One node in particular, NXT2, was not among the original host factors but was identified as a possible host factor in a later genome-wide CRISPR/Cas9 screen (Han et al., 2018). This demonstrates how pathways chosen through pathway parameter advising could guide new discoveries.

The influenza study also illustrates the danger of running pathway reconstruction methods with default parameters alone. The PCSF network constructed using default parameters had 6676 nodes, which is too large to feasibly interpret. We then looked into the role of NXT2 in the default parameters network. Figure 2.13 shows the subnetworks of the default parameters network for all nodes reachable within 2 and 3 edges of NXT2. As opposed to a functionally

cohesive subnetwork, NXT2 only connects to a high-degree node. This node then connects to APP and SUMO2, which are the 2 highest degree nodes in the interactome. This results in 727 nodes being within 3 edges of NXT2, and is more likely an artifact of the PCSF algorithm run with improper parameters than a meaningful biological pathway structure. This hub based structure gives much less insight into the role of NXT2.

2.7 Conclusions and Future Work

Pathway parameter advising selects parameters that lead to useful, plausible networks for a variety of pathway reconstruction algorithms. This parameter tuning approach is algorithm-agnostic and uses background knowledge in the form of pathway databases to succeed in selecting reasonable pathways during pathway reconstruction.

Many of the networks down-ranked by pathway parameter advising, such as pathways with many thousands of nodes or a network consisting only of a single node and its neighbors, seem obvious to avoid. Typically, these types of generated pathways are ignored through a process of manual trial and error. Any manual step in the network analysis could lead to human error, may accidentally introduce bias into the final pathway model, and limits the number of parameter combinations that can be assessed. Therefore, automatically avoiding these poor pathways is important. The specific choice of criteria for defining implausible pathways was inconsequential. Pathway parameter advising excelled at implausible pathway detection compared to other parameter ranking methods for all definitions. Pathway parameter advising quantifies and deprioritizes implausible topologies without any human intervention except for the inclusion of background knowledge.

In addition to avoiding implausible pathways, pathway parameter advising narrowly performed best at reconstructing NetPath pathways. Although it was less clearly dominant than in the implausible pathway detection experiment, this further highlights its effectiveness. Much of the total performance is driven by how well pathway parameter advising performs in PCSF, though it is also the only one to be either the first or second highest performing for all 4 pathway reconstruction methods. However, raw MCC values in many test cases were so

low that differences in MCC were driven by only a few interactions, so this experiment alone does not provide enough evidence to draw strong conclusions.

We also found that no single graphlet dominated our ranking metric. There are likely two causes. In larger networks, such as complete protein interaction networks, disconnected graphlet counts can dominate other graphlets by orders of magnitude (Johansson et al., 2015). In contrast, the networks in biological pathway reconstruction tend to not be large enough for the portion of unconnected graphlets in sparse graphs to completely dominate other graphlets. In addition, our ranking metric only calculates distance from the closest 20% of reference pathways. Thus, the signal of pathway size from the 4 unconnected nodes graphlet guides the selection of the closest reference pathways to pathways of similar size. Within this 20%, other graphlets representing more local topology have a larger contribution.

Although we used different pathway databases in our experiments for reconstructing pathways and the set of reference pathways, it is possible that some pathways are similar across the NetPath and Reactome databases. Cross-database pathway similarity could cause a version of the reconstructed pathway to be used as a reference pathway. However, even if this is the case, the shared pathway would be 1 of the over 1000 Reactome pathways used as a reference set. Thus, its effect on the ranking metric would be negligible.

In both experiments, other parameter selection methods especially struggled choosing parameters for PCSF. Finding a good parameter setting for a method with multiple, complex parameters like PCSF can be especially difficult and important and is where pathway parameter advising is most useful. In contrast, a method like PathLinker contains a single parameter, which monotonically increases pathway size. Changing the parameter value has a relatively predictable effect.

There are some drawbacks to pathway parameter advising. It requires a parameter sweep as opposed to a single run with the default parameter setting. This greatly increases overall runtime for some pathway algorithms. Because pathway parameter advising is algorithm-agnostic, it makes no assumptions about the parameter space it is optimizing. Thus, pathway parameter advising has no way of knowing if the set of parameters considered is broad

enough to find the optimal pathway. However, it is worth noting that all other parameter selection methods tested except for the default parameters suffer from this drawback as well.

Another issue is that pathway parameter advising is dependent on a database of reference pathways. The popular pathway database Reactome works well in analyses here. However, if the optimal predicted pathway is reasonable but outside the range of topologies seen among the reference pathways, it would be overlooked.

Our distance metric focuses only on topology and does not include any information about the biological context. ResponseNet (Basha et al., 2019) and PathLinker (Youssef et al., 2018) extensions consider tissue-specificity and protein localization context, respectively. A possible extension of pathway parameter advising would be to account for this information, such as adding a penalty for interactions that occur in different tissues or cellular compartments. Similarly, we used the entire Reactome database as the reference pathways. Limiting the reference pathways to a certain process or function, such as signal transduction or disease, could allow pathway parameter advising to select pathways more similar to a domain of interest. In addition, instead of computing graphlet distance using 20% of all reference pathways, we could first cluster the reference pathways and consider the distances only to pathways in the most topologically similar cluster.

Another possible class of graph distance metrics are those based on graph matching (Xu et al., 2021), subgraph matching (Tian et al., 2007), or network alignment (Guzzi and Milenković, 2018; Ma and Liao, 2020). These methods seeks to find a 1:1 correspondence between 2 networks or between subgraphs within 2 networks, and the resultant matching or approximate matching can be used to compute a distance between networks such as edit distance (Wang et al., 2019). These matching-based distances could provide a distance that is more biologically informed, as pathway members could be directly compared between the constructed pathway and the reference pathway. However, these distance metrics might over-constrain constructed pathway topologies, not allowing for unexpected subgraph structures to be in the chosen pathway.

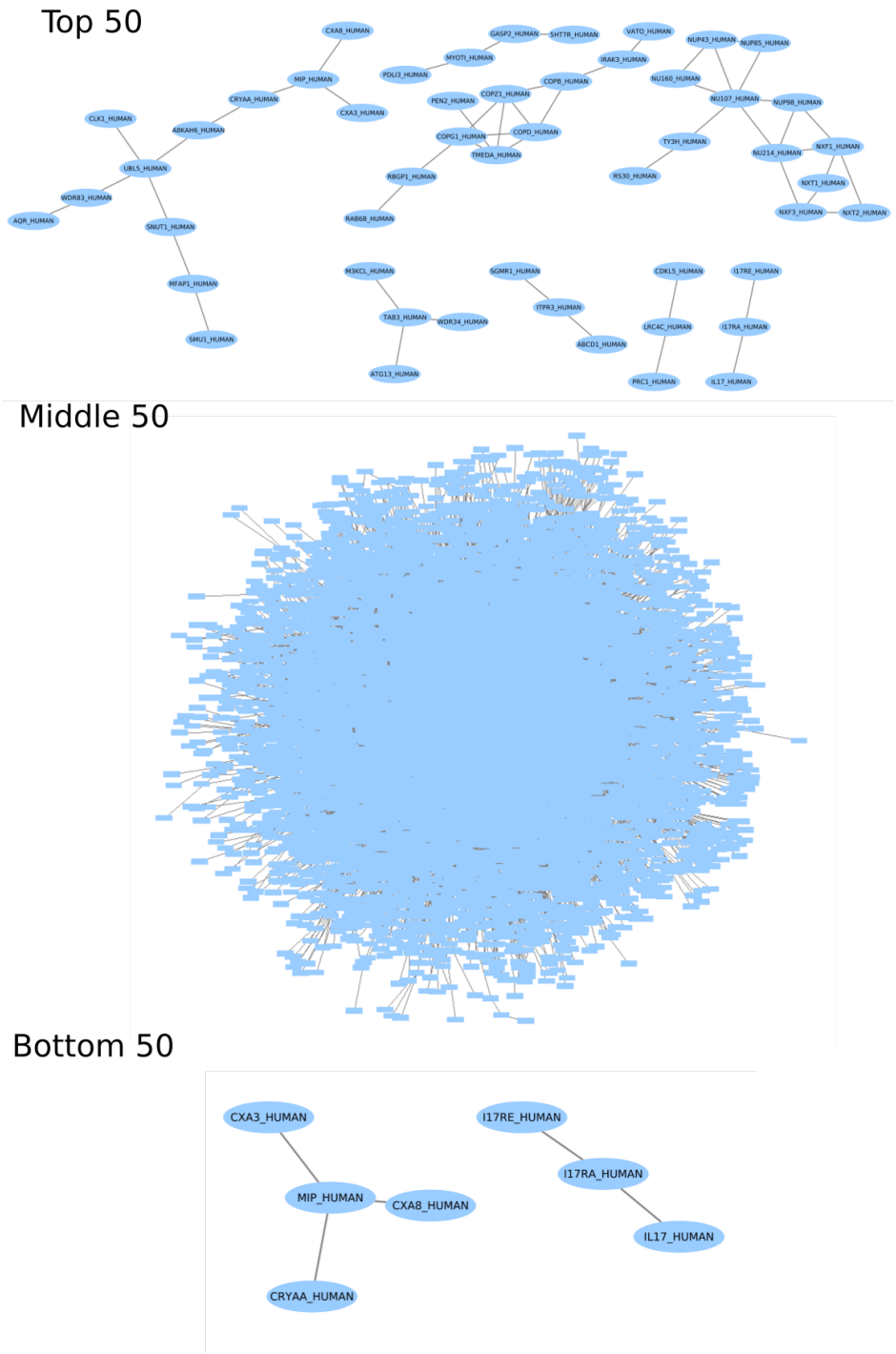

Figure 2.12: Influenza host factor networks created from ensembling PCSF runs. The resulting pathways from the top 50, middle 50, and bottom 50 parameter settings as ranked by pathway parameter advising. All connected components over 3 nodes are shown.

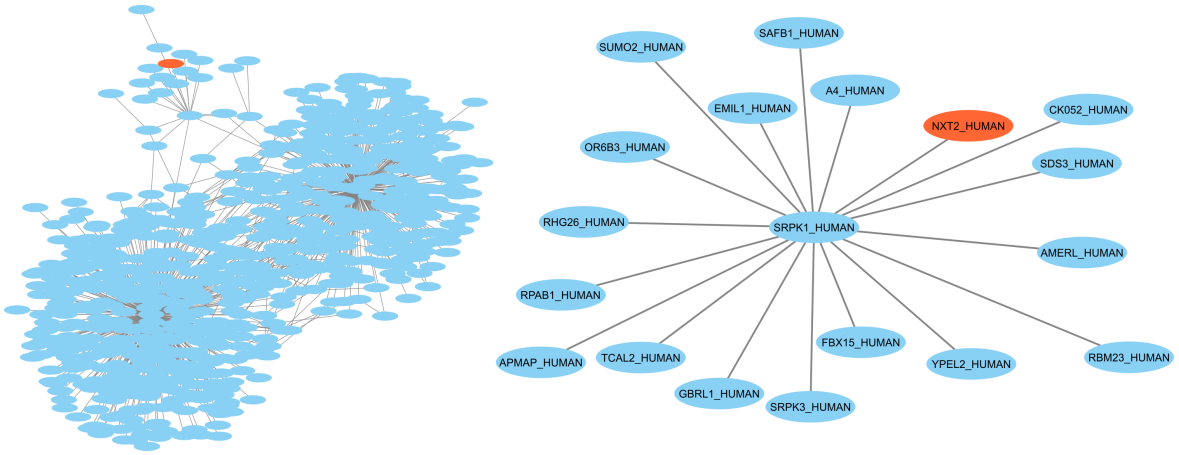

Figure 2.13: All nodes within distance 3 (**left**) and distance 2 (**right**) of NXT2, highlighted in orange, in the PCSF influenza host factor network constructed from default parameters. Using the default parameters alone resulted in a large hub-node focused network with little useful biological insight.

Chapter 3

Predicting localization within pathway context

3.1 Motivation and Related Work

Cellular state is dictated by a host of potential factors, from chromatin accessibility to protein abundance. Bringing additional sources of information to bear in biological analyses can elucidate these factors role and provide explanation for biological phenomena. One important factor in biological function is the physical location of proteins within the cell. Cells are compartmentalized into numerous subcellular locations that provide the chemical environment around proteins, informing their structure, and dictate which other proteins and biological entities are available to interact with. How different proteins localize to different subcellular compartments is a key layer of information for understanding cellular processes (Lundberg and Borner, 2019).

Protein localization not only dictates protein interactions in normal cellular processes, but also is an important factor that can contribute to abnormal cellular behavior. Diseases such as Alzheimer's, ALS, Wilson disease, and various forms of cancer all involve abnormal protein localizations (Hung and Link, 2011). Therefore, knowing these non-standard localization can give insight into how a disease is operating a cellular level, and which proteins may be

key players.

Ideally, this localization information could be combined with pathway reconstruction analyses. Thus a constructed pathway, which represents the cellular state being examined, could be labeled with protein localizations within the context of that particular cellular state. For example, during HIV-1 infection subcellular localization likely plays a role in regulating viral assembly, and disrupting mRNA subcellular localization has been proposed as a possible therapeutic strategy (Becker and Sherer, 2017). In the motivating HIV-1 work presented in Section 2.1, localization information added to the reconstructed pathway could provide additional information that leads to new theories or hypotheses about the process of HIV-1 infection. Therefore, localization information is needed in a form that can be contextualized to a particular biological pathway.

Computational prediction of protein subcellular localization has traditionally been focused on predicting the possible set of all subcellular localizations a protein belongs to in all biological processes it is a part of. There are a variety of features of protein structure and sequence that inform the chemical environments it likely operates in, and the localization processes it can participate in (Eisenhaber and Bork, 1998; Bauer et al., 2015). These predictions can help infer protein function and potential protein-protein interactions.

There has been a substantial amount of work on the computational problem of predicting the possible locations of a protein based on its sequence (Gardy and Brinkman, 2006; Imai and Nakai, 2010; Alaa et al., 2019). These methods have used a number of machine learning methods from logistic regression (Hua and Sun, 2001) to deep neural networks (Almagro Armenteros et al., 2017). Many methods also incorporate additional information, such as gene expression (Drawid and Gerstein, 2000) or GO annotations (Fyshe et al., 2008). Other methods incorporate network information (Ananda and Hu, 2010; Du and Wang, 2014; Garapati et al., 2020; Grover and Gatto, 2022), using the localizations of neighboring proteins to aid in localization prediction.

It is estimated that up to 50% of proteins localize to multiple cellular compartments (Thul et al., 2017; Zhang et al., 2008). Though predictive methods tend to consider protein localiza-

tion as static, localization in reality is a dynamic process mediated by a variety of physical factors and biological processes (Bauer et al., 2015). While most predictive methods account for this by treating localization prediction as a multi-label classification task, where proteins can be assigned multiple localizations, these methods still inherently treat localization as unchanging and without context. Contextual information such as tissue-type (Zhu et al., 2019) has been explored in a predictive context, but proteins can change subcellular compartment within the context of a single biological process in a cell. Single cells within same tissue have been found to have different localizations (Lundberg and Borner, 2019).

There are also a variety of databases which contain protein-specific localization data, such as MatrixDB (Chautard et al., 2010), Organelle DB (Wiwatwattana and Kumar, 2005), Compartments (Binder et al., 2014), and ComPPI (Veres et al., 2015). These databases range from primary, experimental sources of localization information to computationally predicted localizations and databases combining multiple sources of information. Many biological pathway databases also include localization information (Fabregat et al., 2018; Wishart et al., 2019). Pathway databases typically contain more specific information than protein-level databases; they provide localization information at the interaction level and include information about non-protein biological entities. Interaction-level localizations allow pathway databases to mostly circumvent the issue of multiple localizations at the protein level. Localization can be treated granularly when addressed within a certain biological pathway and at the point of a single interaction.

Many pathway databases, however, contain no localization information. For instance, of the 8 pathway databases included in Pathway Commons, 2 are fully labeled with localization information, 5 are partially labeled with localization information, and 1 contain no programmatically available localization information. Thus, these databases cannot be relied upon to provide localization information for all analyses. For visualization and exploratory or other analyses context-free protein level data alone cannot be directly plugged into pathways to give localization information. This is both because of the dynamic nature of subcellular localization, and because many members of pathways are not proteins. Localization informa-

tion may not be available for metabolites and other small molecules in biological pathways. Furthermore, the previously mentioned issue of pathway specificity in Chapter 2 also inhibits the usage of pathway databases in the context of localization. The cellular state of interest for a particular biological experiment may not match any available pathway in the database.

Localization of proteins to multiple compartments, and most sources of localization information presenting protein labels without specific context, both make the task of applying localization information to a reconstructed pathway difficult. Any attempt to directly apply information from protein-level databases to a reconstructed pathway would result in many proteins having multiple localizations with no obvious way to choose one. There would also be a significant proportion of missing information.

While there are experimental methods for determining localization (Lundberg and Borner, 2019) using mass spectrometry or cellular imaging, these methods can be expensive and technically involved. Ideally, localization information could be added to an existing analysis without the need for performing an additional experiment. However, current prediction methods and localization data sources do not provide this information in specific functional contexts or in a consistent, complete way.

Thus, there is a need for localization prediction at the pathway level. This context will allow for examination of where proteins or other biological entities are when they are performing a particular biological function, enabling the full utilization of localization data within biological exploratory analyses. Pathway-specific localization prediction could provide a valuable additional layer of information to biological analyses. We created a pipeline for prediction localization at the interaction level within the context of a biological pathway. We explored a variety of methods for performing prediction, and examined the methods' performance on interaction-level data from publicly available databases as well as in a case study involving human cytomegalovirus infection over time (Jean Beltran et al., 2016).

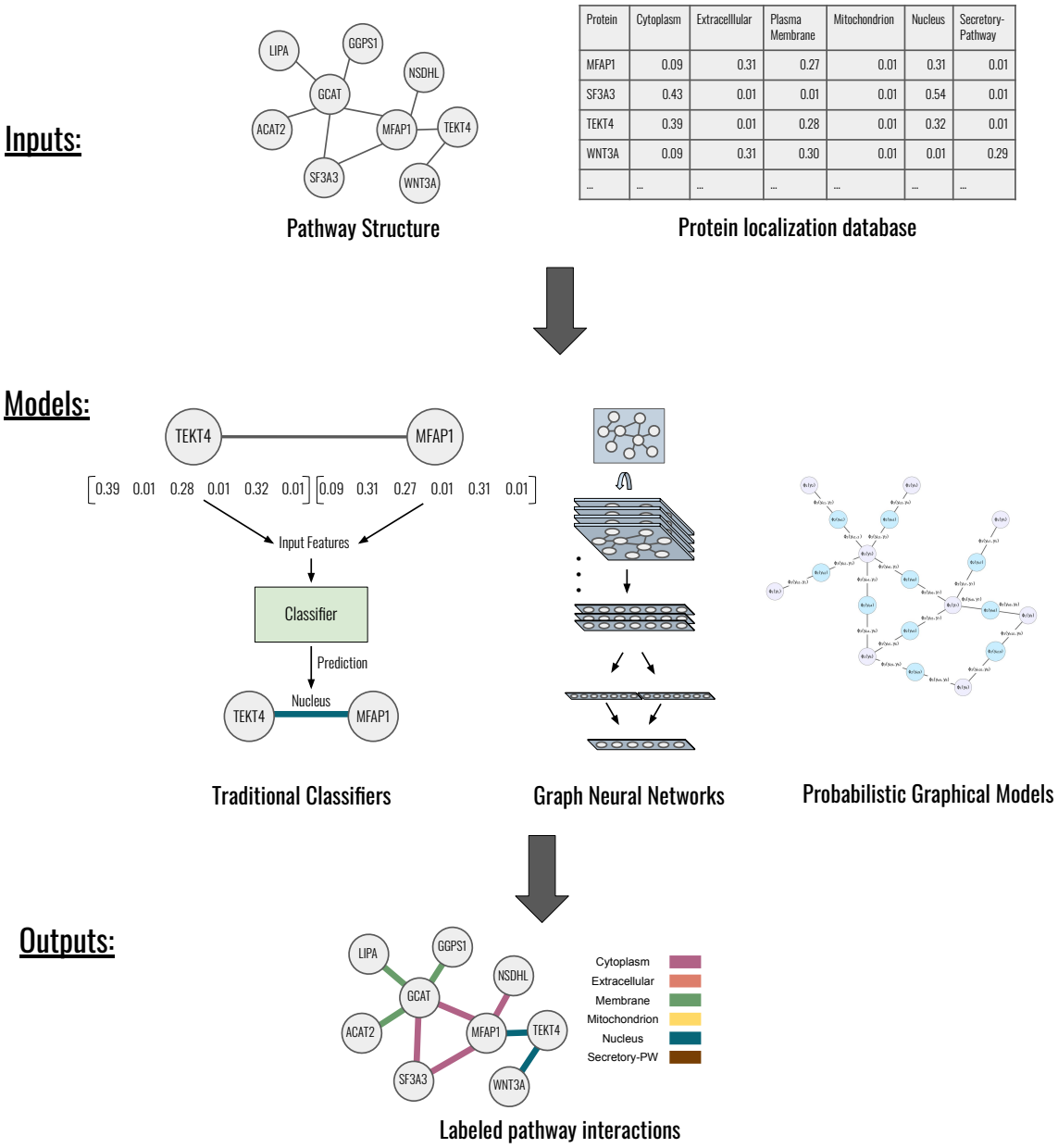

Figure 3.1: Overview of localization experimental workflow.

3.2 Interaction Localization Prediction

Given a biological pathway represented as a graph consisting of nodes and edges $G = (N, E)$, the goal is to predict one or more subcellular localizations for each edge $e \in E$. To perform a context-specific localization prediction, we consider subcellular location within a particular biological pathway. Furthermore, as opposed to predicting localization at the protein level we instead predict a localization for each edge in the pathway. Protein-level localization information is used as input to the prediction task.

Thus, the pathway-specific subcellular localization task can be represented as the following classification task:

Input: (1) A biological pathway represented as a graph consisting of nodes and edges $G = (N, E)$, and (2) protein level context-free localization information from a database for each protein in the pathway.

Output: A single localization assignment for each interaction $e \in E$.

We perform this prediction using three general categories of models: general classifiers, probabilistic graphical models, and graph convolutional neural networks. Each of these categories is explained in detail below.

Prediction at the interaction level has a number of advantages over protein-based localization prediction. First, predicting localizations for each interaction is a more biologically accurate depiction of subcellular localization. Proteins are not always in the same part of the cell; however, in the context of a specific biological function a protein's interaction with another protein can be considered to occur in only a single location with little loss in possible accuracy. Thus prediction localizations for each interaction better represents cellular processes than predictions at the protein level.

For example, in all Reactome and PathBank pathways, less than 5% of interactions occur multiple times within a single pathway in different subcellular locations. This is opposed to the estimations of the up to 50% percent of proteins which are estimated to localize to multiple cellular compartments across all of their biological functions Thul et al. (2017). Thus, in addition to being a more biologically accurate depiction of localization, interaction-based

localization prediction circumvents issues with multiple localizations.

Finally, interaction-based localization prediction better integrates with representations of biological pathways. Pathway databases such as Reactome and popular file formats for biological pathways such as BioPax Demir et al. (2010) only allow proteins to be in a single subcellular location. Multiple entities representing the same protein are used to represent a protein being in different subcellular locations in different contexts. Interaction-based prediction allows for direct use of these pathway models.

We used the biological pathway databases Reactome and PathBank as labeled data for training and evaluation, much like in Chapter 2. While models were trained with this pathway data, during inference the only available localization features are from protein-level localization databases. No interaction-level or pathway-level localization data is included at inference time as reconstructed pathways do not have any of this information.

3.3 Experimental Setup

Spatial Proteomics Case Study

MS Datasets: To investigate a possible use-case of pathway-based localization prediction, and validate its performance, we performed localization prediction on proteomic quantification of primary fibroblasts during human cytomegalovirus (HCMV) infection (Jean Beltran et al., 2016). Two mass spectromety quantification methods were used at 5 time points post infection: 24, 48, 72, 96, and 120 hours post infection (hpi). The first of these datasets provides label-free protein quantification at each timepoint. The other was quantified using isobaric labeling via tandem mass tags (TMTs). These two datasets will be referred to as the label-free and the TMT datasets, respectively.

Multi-organelle profiling was performed on the TMT dataset via gradient centrifugation to fractionate organelles. This process partially separates organelles into a set of subcellular fractions. While each of these fractions are not purely a single organelle, each organelle contains a unique signature in its quantification across the subcellular fractions (Lundberg

and Borner, 2019). A supervised learning model, in this case an ensemble of neural networks, is then trained on the marker proteins and used to infer localization labels for the non-marker proteins.

In the HCMV protein quantification a set of marker proteins were curated from Uniprot subcellular location annotations with experimental evidence; proteins that were annotated with multiple localizations were excluded from the marker set. Proteins that were not confidently assigned to a particular organelle were left as unlabeled. There were 2730 proteins in total, of which 1229 had localization labels at 24 hours post infection (hpi) and 1348 had labels at 120hpi. 574 of these proteins were marker proteins. More details of the protein labeling and quantification process can be found in (Jean Beltran et al., 2016).

Experimental Setup: We create the scenario where we wish to learn localization information about HCMV infection over time, but do not have access to spatial proteomic data over the course of infection. Instead, we attempted to use pathway localization prediction combined with a pathway reconstruction analysis to create context-specific pathways and localization predictions. We then compared these prediction to subcellular localizations found via spatial proteomics to evaluate our predictions.

To simulate the use of pathway localization prediction, we performed pathway reconstruction on the label-free data set at the first and last timepoints (24hpi and 120hpi). The best performing model from the pathway database experiments, the graph attention network with the Compartments database as features, was then trained on the set of marker proteins used in the original experiment and evaluated on all protein localization predictions from multi-organelle profiling.

Pathway reconstruction: Pathway reconstruction was performed using the OmicsIntegrator2 (Tuncbag et al., 2016) package. OmicsIntegrator2 performs pathway reconstruction via the prize-collecting Steiner forest problem, which was introduced in Chapter 2. Pathway reconstruction was setup using the SPRAS software package [1]. Protein fold-change was used as prizes for both time points. We used the same interactome as in Chapter 2 originally from

[1] `https://doi.org/10.6084/m9.figshare.14551476.v2`

(Köksal et al., 2018) that merged protein interactions from the iRefIndex database v13 (Razick et al., 2008) and kinase-substrate interactions from PhosphoSitePlus (Hornbeck et al., 2014). This resulted in a network with 161901 weighted edges. Pathway parameter advising Magnano and Gitter (2021) was used to select the top 50 pathways from 1000 candidate parameter combinations. ω was tested between 1 and 10, β was tested between 1 and 5, and μ was tested between 0.1 and 1. Each parameter was evaluated at 10 increments across its range. Pathways were constructed for the 24hpi, 48hpi, and 120hpi timepoints. The 48hpi pathways were used as a tuning set for parameter selection in the final model. Tuning was performed using Bayesian optimization as in the pathway database prediction experiment.

Data

Pathway Databases

Pathway datasets were constructed from the Reactome (Fabregat et al., 2018) and PathBank (Wishart et al., 2019) databases. Pathways were downloaded from Pathway Commons (Rodchenkov et al., 2019), and localization information was retrieved from BioPax pathway representations using the PyBioPax[2] package v0.1.0. Reactome contains localization information for all edges. PathBank, however, contains nodes and edges with missing localization information. These missing data were excluded from all analyses.

The original pathways in both Reactome and PathBank are represented as hypergraphs, where reaction edges can contain more than two nodes. PathwayCommons converts these hypergraphs to graphs using different rules for different types of reactions[3]. This hypergraph conversion can affect the resulting graph topology and class distribution. During preliminary experiments we found that the hypergraph conversion resulted in an over-representation of protein-complex edges. To represent a protein-complex that contains n proteins, the hypergraph conversions creates an edge between every possible pair of nodes, resulting in n^2 edges. For instance, in PathBank the pathway Protein Synthesis: Serine as a hypergraph has 4 hyper-edges. However, when converted to a graph the pathway contains 3318 edges, of

[2] `https://github.com/indralab/pybiopax`

[3] `http://www.pathwaycommons.org/pc2/formats`

which 3315 are of type "in-complex-with". In Reactome after hypergraph conversion over 75% of edges among all pathways are of type "in-complex-with". While this preserves protein information, for an edge classification task this conversion results in highly skewed data.

We collapsed protein complexes into single nodes where possible in all pathways. This was done by removing any node from a pathway if all of its edges were redundant with the protein-complex's edges. A single node for each complex was left in the pathway to represent that complex. This significantly reduced the size of some pathways. For instance, the Protein Synthesis: Serine pathway was reduced from 3318 to 14 edges. Though this loses some node information, collapsing protein complexes resulted in pathways that more more closely resembled the original hypergraph in both edge distribution and topology. Collapsing protein complexes coincidentally slightly decreased class imbalance in Reactome.

After this protein-complex collapsing step, all pathways with fewer than 4 nodes were excluded from the analysis. This resulted in 953 Reactome pathways and 467 PathBank pathways.

Both pathway databases contain a highly skewed distribution of localizations across all interactions. The rarest localization labels in both databases, secretory-pathway and nucleus in Reactome and PathBank, respectively, occurs in less than 0.5% of all edges. The most common localization, which is cytosol for both databases, consists of 38% of Reactome interactions and 52% of PathBank interactions.

Protein Localization Databases

Two protein localization databases were used throughout all experiments, Compartments (Binder et al., 2014) and ComPPI (Veres et al., 2015). ComPPI is a meta-database for protein subcellular localizations. It combines data from 8 subcellular localization databases. It does not include data from Compartments. Proteins are assigned scores for each of 6 subcellular locations: cytosol, plasma membrane, mitochondrion, extracellular, nucleus, and secretory-pathway. These 6 locations were used for all predictions; localizations for all other data sources were mapped to these 6. ComPPI combines weights for different types of evidence

across its data sources to give a probability of a protein to be found in a particular subcellular location. All human ComPPI data was retrieved on 2020-11-09.

Compartments is a protein subcellular localization database that combines data from 4 different types of data: database annotations, experimental screens, automated text mining, and predictive sequence-based models. Each data source is given a confidence score between 1 and 5 based on the level of evidence. Compartments assigns proteins to 1 of 11 subcellular locations. These 11 locations were mapped to the 6 localizations in the ComPPI database. All Compartments data was retrieved on 2021-09-29.

Uniprot Keyword Features

To explore the utility of additional protein data in predicting localization data, Uniprot keywords were collected for all human proteins. Uniprot keywords are a controlled hierarchical vocabulary that represent a variety of protein categories such as molecular function, disease participation, structural features, and biological processes. These keywords are manually assigned and include localization data. While Uniprot keywords provide a range of protein-level data, they consist of hundreds of terms, many of which are only used by a handful of proteins. Thus, they are impractical to use directly as features.

Keywords were converted into features through dimensionality reduction. Principal component analysis was performed on all keywords present in at least 5% of human proteins. Technical keywords such as "3D-Structure" and "Reference proteome" were excluded as not pertaining directly to the protein itself. Each protein was then represented by the first 6 components, chosen by a dropoff in explained variance after the first 6 components. The most important keywords for these components represented a variety of biological concepts; from functional categories such as "Tumor suppressor" and "Lipid biosynthesis" to structural features such as "ANK repeat" and "Voltage-gated channel". These components only accounted for 42% of variance. However, given the diversity of keywords it is unlikely a small number of features could fully represent them.

Models

Model and parameters selection were performed on a validation set of the 53 Reactome pathways categorized as belonging to the "developmental" functional area and a randomly chosen 10% of all PathBank pathways. Parameter selection was performed via Bayesian optimization (Balandat et al., 2020) using the Ax package[4] for neural network models and the Scikit-optimize package[5] for classifier models. Bayesian optimization was performed for 30 iterations for each model.

Model	Parameter	Description/Notes	Range
All Neural Networks	Learning Rate	Learning rate for training.	$10^{-5} - 0.01$
	Linear Depth	The number of linear layers.	$1 - 5$
	Convolutional Depth	The number of convolutional layers. Not used by fully connected network.	$1 - 10$
	Dim	The number of dimensions in hidden layers.	$24 - 128$
	Dropout	Whether or not to add dropout while training.	True, False
Graph Convolutional Network		No Unique Parameters	
Graph Attention Network	Heads	The number of attention heads	$1 - 5$
Graph Isomorphism Network		No Unique Parameters	
Fully connected Network	Activation	Activation function used	Tanh, ReLU
Random Forest	max_depth	Maximum tree depth.	$1 - 10$
	min_samples_split	Minimum samples to create branches.	$2 - 10$
	n_estimators	Number of trees.	$1 - 100$
	class_weight	Whether to balance class weights.	True, False
Logistic Regression	tol	Tolerance for training.	$10^{-6} - 0.1$
	penalty	Regularization penalty to use.	L2, None
	C	Regularization strength (lower is stronger).	$0.01 - 100$
	class_weight	Whether to balance class weights.	True, False

Table 3.1: Classifier and neural network models and parameters ranges searched for each. Chosen parameter values can be found in Table 3.2

Neural Networks

The maximum value in the output layer is used as the final classification. All neural network models were trained using cross-entropy loss. All neural networks were implemented using the PyTorch Geometric package (Fey and Lenssen, 2019).

[4] `https://ax.dev/`

[5] `https://scikit-optimize.github.io/stable/`

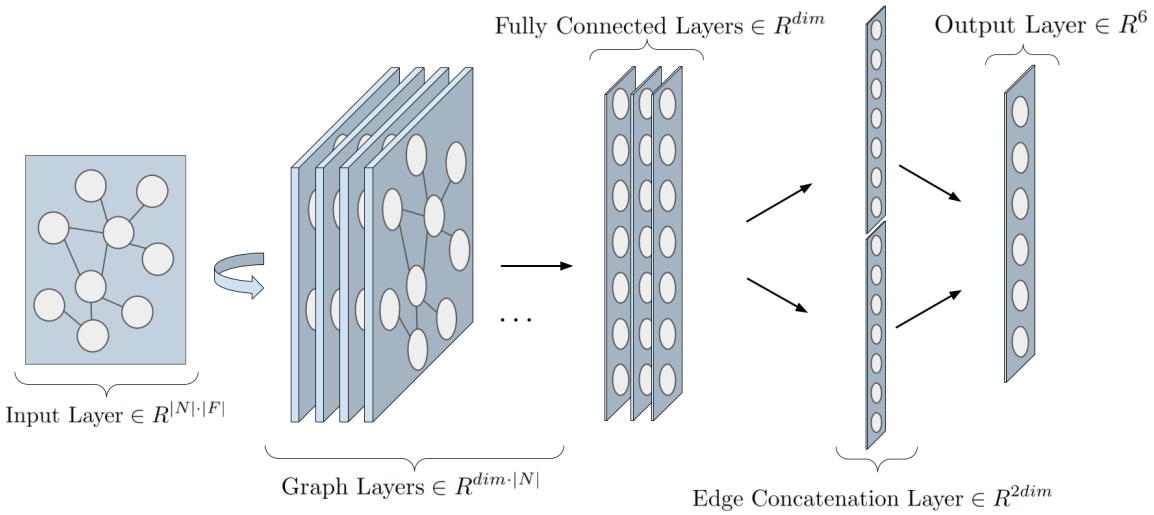

Figure 3.2: Overview of neural network architecture for graph neural networks. The number of graph layers is controlled by the parameters convolutional depth, and the number of fully connected layers is controlled by the parameter linear depth. $|N|$ represents the number of nodes in the input pathway and $|F|$ represents the number of input features for each node.

Fully connected neural network: In order to investigate the effects of graph topology on localization prediction, a topology free neural network was constructed. The fully connected neural network begins with an edge concatenation layer as shown in Figure 3.2, followed by fully connected layers of size dim up to linear depth.

Graph convolutional network: The graph convolutional network (Kipf and Welling, 2017) incorporated a set of message-passing convolutional layers before the final set of fully connected layers. These convolutional layers allow for information to be shared across the topology of the input network. The l^{th} convolutional layer $H^{(l)}$ are updated via the following rule:

$$H^{(l)} = ReLU(D^{-\frac{1}{2}}\widetilde{A}D^{\frac{1}{2}}H^{(l-1)}W^{(l-1)})$$

Where $\widetilde{A}$ is the adjacency matrix of the input pathway with added self-edges for all nodes, D is a degree matrix normalization factor where $D_{ii} = \sum_j \widetilde{A}_{ij}$, and $W^{(l)}$ is a set of weights for the l^{th} layer. This update rule provides a first-order approximation of spectral graph convolutions (Defferrard et al., 2016; Hammond et al., 2011) and is implemented in the

$GCNConv$ class in PyTorch Geometric.

Graph attention network: Graph attention networks extend graph convolutional networks by allowing each node choose which of its neighbors to pay attention to. As opposed to taking the average of its neighbors, each node computes a weighted average of its neighbors in graph convolutional layers (Veličković et al., 2018). Furthermore, many attention networks are multi-headed, where multiple attention weights are computed in parallel for each node. The number of heads to include is an input parameter, and generally increases accuracy at the cost of increased computational complexity. We used the $GATV2Conv$ class for graph layers, which is a more expressive implementation of graph attention networks that allows for more diversity in attention between nodes (Brody et al., 2021).

Graph isomorphism network: Graph isomorphism networks (Xu et al., 2019) take advantage of the similarity between neighbor aggregation in graph neural networks and the Weisfeiler-Lehman (WL) graph isomorphism test (Weisfeiler and Leman, 1968). The WL graph isomorphism test is a heuristic algorithm for determining graph isomorphisms. For two graphs, each iteration of the test every node aggregates its neighbors into a unique hash. These hashes are compared between the two graphs, and if they differ the graphs are known to be non-isomorphic. Iterations of the test are repeated until the user feels confident that the graphs are isomorphic; the algorithm cannot conclusively prove isomorphism.

The neighbor aggregation in each graph layer of a graph isomorphism network is formulated to be at least as powerful as the WL isomorphism test; the l^{th} layer is guaranteed to generate different embeddings of two graphs if those graphs would be found to be non-isomorphic via the WL isomorphism test in l iterations. The representation of each node in layer l of a graph isomorphism network, $h_n^{(l)}$, is computed as:

$$h_n^{(l)} = MLP^{(l)}((1 + \epsilon^{(l)}) \cdot h_n^{(l-1)} + \sum_{u \in Adj(n)} h_u^{(l-1)})$$

Where MLP is a multi-layer perceptron, ϵ is a learned parameter, and $Adj(n)$ is the set of nodes adjacent to n in the input pathway. We used the $GINConv$ class in PyTorch Geometric for graph isomorphism layers.

Probabilistic Graphical Models

Given the topological nature of the pathway level localization prediction algorithm, and that many localization databases contain uncertain or even probabilistic data, probabilistic graphical models are a natural choice for an initial approach to pathway level localization prediction. As moving between subcellular locations costs energy, it is unlikely to happen often within a single pathway. Therefore, we can make the assumption from a modeling perspective that the subcellular location of an interaction is dependent on the subcellular location of neighboring interactions within a pathway.

Probabilistic graphical models represent a set of N random variables $\mathbf{y}$ as nodes and dependencies between them as a set of edges E. We created two pairwise undirected probabilistic graphical models (Gewali and Monteiro, 2018), which we call NaivePGM and TrainedPGM, where each node $i \in N$ has a corresponding localization label y_i. These localization labels can take on any of the 6 localizations mentioned in Section 3.3. In these probabilistic graphical models the random variables obey a local Markov property, such that each random variable is conditionally independent of all others given its neighbors in the graph.

The NaivePGM is a Markov random field, where the joint probability of all localizations can be factorized as

$$P(\mathbf{y}) = \frac{1}{Z} \prod_{i \in N} \phi_i(y_i) \prod_{i,j \in E} \phi_{ij}(y_i, y_j)$$

Where Z is a normalizing function so that the combination of all possible configurations of $\mathbf{y}$ sum to 1 and $\phi_i y_i$ and $\phi_{ij} y_i, y_j$ are the unary and pairwise potential functions, respectively. The unary potential function defines the probabilities of a given node having each localization, while the pairwise potential functions define the joint probability of each pair of nodes that share an edge. For finding the task of finding the most likely configuration of $\mathbf{y}$, referred to as decoding, Z can be ignored. In the NaivePGM the input features are not used to parameterize the potential functions. Instead, the unary potential functions directly map the normalized features to class probabilities, and the joint probability tables directly map the normalized

features to join probability tables. This was chosen as both the ComPPI and Compartments scores represent confidences, with ComPPI scores directly representing probabilities for each localization.

The TrainedPGM is a conditional random field where the input features are treated as observations of additional variables. The probability of localization assignments $\mathbf{y}$ are then conditioned over the input features $\mathbf{x}$ as:

$$P(\mathbf{y}|\mathbf{x}) = \frac{1}{Z} \prod_{i \in N} \phi(y_i, x_i) \prod_{i,j \in E} \phi(x_i, x_j)$$

Here, the unary potential functions are now conditioned on observations of features x_i corresponding to each random variable y_i. Pairwise potentials are a function of observations, thus, each random variable y_i is conditionally independent of all other variables given its corresponding features/observations x_i.

The potential functions in conditional random fields are typically log-linear functions of the form $e^{w_i^T \phi_f(x_i, y_i)}$, parameterized via a weight vector $\mathbf{w}$ and ϕ_f simply represents features for each node. Additionally, typically the feature weight vectors are shared between nodes or sets of nodes. Thus, the entire model can then be represented as:

$$p(\mathbf{y}|\mathbf{x}, \mathbf{w}) = \frac{1}{Z} exp\Big(\sum_{i \in N} \mathbf{w}_f^T \phi_f(x_i, y_i) + \sum_{i,j \in E} \mathbf{w}_e^T \phi_e(x_i, x_j) \Big)$$

Where $\phi_f(x_i, y_i)$ is the single unary potential function that represents features for each node, and $\phi_e(x_i, x_j)$ is the single pairwise potential function that represents combinations of features. The weight vector $\mathbf{w_f}$ is a set of weights for each feature to each possible configuration of y_i, while the weight vector $\mathbf{w_e}$ is a set of weights for each feature and combination of configurations for y_i and y_j.

When represented with these potentials, the log likelihood of the model parameters $\mathbf{w}$ can be easily represented, and is differentiable, allowing for parameters to be learned by maximum likelihood estimation via gradient-based optimization (Gewali and Monteiro, 2018). Sets of nodes and edges can share the same set of model parameters, referred to as

parameter tying.

However, parameter learning for a conditional random field of this form did not converge when trained with stochastic gradient descent in preliminary experiments. This may be due to the underlying label distributions of different pathways being too different from each other.

Instead a model was chosen where potentials are represented by discriminative classifiers (Kosov, 2018), here random forests. This type of model is referred to as a discriminative random field (Kumar and Hebert, 2006). This was chosen over a more traditional log linear parameterization due to better performance on validation data during model selection.

These models only provide predictions on the nodes of the graph, while we are interested in localization labels on the edges. To convert the input pathway into an appropriate graphical model, each pathway is converted into a bipartite graph, where a node is added that represents each edge. First, all nodes from the original pathway are added to the graphical model. No edges from the original pathway are added. Instead, for each edge e_{ij} between nodes n_i and n_j, a node n_{eij} is added representing the interaction. Then 2 edges are added to the graphical model going from each original node to the new interaction node, $n_{ie_{ij}}$ and $n_{je_{ij}}$. An overview of this process can be found in Figure3.3. Each of the K features f_{ek} in n_{eij} are computed as the normalized product of features from n_i and n_j, here represented as f_{ik} and f_{jk}:

$$f_{ek} = \frac{f_{ik} f_{jk}}{\sum_{l=1}^{K} f_{el}}$$

This is equivalent to how interaction localization probabilities are calculated in ComPPI (Veres et al., 2015). Parameters are tied such that all original nodes are represented by one set of model parameters, and all interaction nodes are represented by another. This can be seen in panel C of Figure 3.3, where ϕ_1 is the set of model parameters that describes the relationship between input features for each protein and its localization, ϕ_2 describes the relationship between each interaction's combined features and its localization, and ϕ_3 describes the relationship between each protein and the interactions it participates in.

Final localization labels can be viewed as a maximum a posteriori (MAP) estimate of

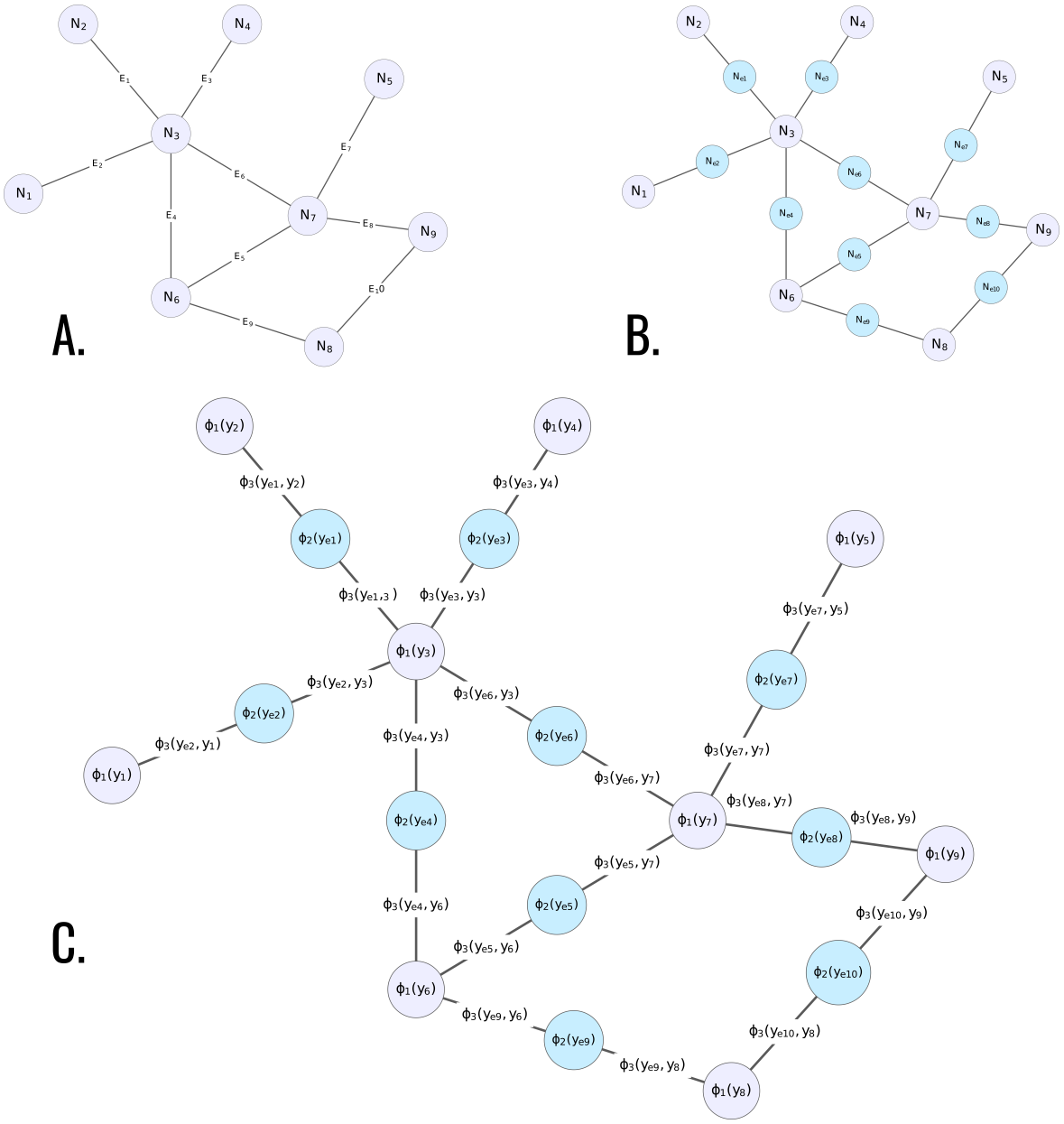

Figure 3.3: Overview of how pathways were represented as probabilistic graphical models for interaction classification. Panel A shows the original pathway structure. Panel B shows then interactions-nodes that are added to each pathway. Finally, Panel C shows how potential functions are used and tied. There are 2 sets of unary potentials, $\phi_1()$ and $\phi_2()$, which model the original nodes and the interaction nodes, respectively. $\phi_3()$ models how each interaction interacts with its adjacent nodes.

the configuration of all interaction node labels. Decoding was performed using loopy belief propagation, which approximates the MAP estimate via a message passing algorithm (Weiss, 2000). Loopy belief propagation was run for 10000 iterations in all cases. Both models were implemented in the DGM software library [6].

Other Classification Models

Two non-neural network classifiers were used to further examine the effect of incorporating topological information into localization prediction: logistic regression, referred to as Logit, and random forests, referred to as RF. These models were specifically included to examine the necessity of models that use topological information. It could be possible that information solely from the interacting proteins is sufficient to predict that interaction's localization. Interactions were represented by concatenating the features of the 2 nodes that make up that interaction, in alphanumeric order by protein identifier. Tested parameter ranges for these models are listed in Table 3.1. Both models were implemented in Python 3.9 using the Scikit-Learn package (Pedregosa et al., 2011) v1.0.2.

Metrics

We used balanced accuracy and F1 score to evaluate predictive performance. These metrics were chosen as popular metrics in multi-class classification that both give all classes equal importance under class imbalance.

Balanced accuracy can be viewed as the average recall across all classes:

$$\text{balanced accuracy} = \frac{1}{L}(\sum_{i=1}^{L} \frac{TP_i}{TP_i + FN_i})$$

where L is the number of classes, TP_i is the number of correctly predicted instances or true positives in class i, and FN_i is the number of incorrectly predicted instances or true negatives in class i.

[6] `https://research.project-10.de/dgmdoc/index.html`

The F1 score can be similarly defined as the average F1 score over all classes, treating each class as a one-against-all binary classification. The F1 score is the harmonic mean of the precision and recall. As the harmonic mean it will aggressively penalize predictions if either the precision or recall is low. It is calculated as:

$$F1 = \frac{1}{L}(\sum_{i=1}^{L} \frac{2P_i R_i}{P_i + R_i})$$

where P_i is the precision of class i, calculated as:

$$P_i = \frac{TP_i}{TP_i + FP_i}$$

and R_i is the recall of class i, calculated as:

$$R_i = \frac{TP_i}{TP_i + FN_i}$$

where FP_i is the number of instances incorrectly predicted to be class i or false positives, and TP_i and FN_i are defined as above. This is considered the 'macro' F1 score, a 'micro' F1 score can also be calculated which favors the majority class(Pillai et al., 2017).

3.4 Comparing Pathway and Localization Databases

To better understand the difficulty in predicting interaction localizations from protein-level localization data, we compared the localizations present in biological pathway databases versus those in protein localization databases. Figures 3.4 and 3.5 show proteins grouped by the localization of edges they belong to in Reactome. The scores shown are localization scores from two protein localization databases, ComPPI and Compartments (see Section 3.3 for more details).

Both pathway databases significantly disagree with both protein localization databases. Directly using data from protein localization databases would not be sufficient to accurately predict pathway level localization; in all combinations of protein localization databases

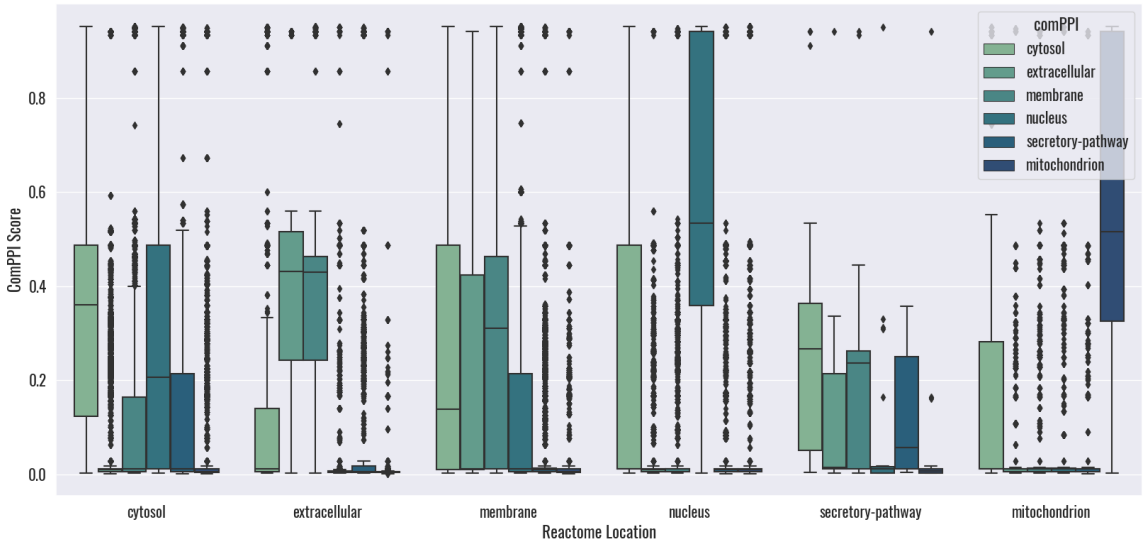

Figure 3.4: Distribution of ComPPI protein scores by the localization of Reactome edges they belong to. Scores the probability of a protein being in a given subcellular location retrieved from the ComPPI database.

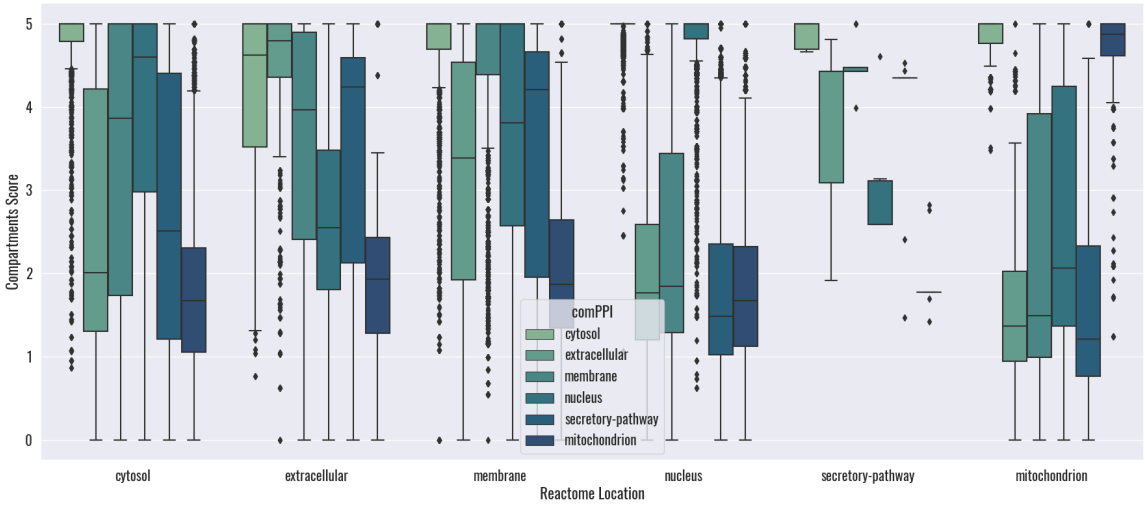

Figure 3.5: Distribution of Compartments protein scores by the localization of Reactome edges they belong to. Scores are confidence scores of a protein being in a given subcellular location, weighted by the type and amount of evidence available. Scores are retrieved from the Compartments database.

(Compartments and ComPPI) and pathway databases (Reactome and PathBank) there are a significant number of interactions whose participating proteins have identical features but localized to different parts of the cell. In addition many interaction localizations would be considered impossible when using a protein localization database alone. For example, almost 14% of interactions in Reactome are between proteins that have no localizations in common in ComPPI. This indicates that pathway topology or some other form of additional information is needed to correctly predict localization in context.

The Compartments database has more disagreement with pathway databases than ComPPI. While the range of ComPPI scores for all interaction localizations is wide, including a significant number of proteins existing in localizations where they have a score of 0, in all subcellular locations but secretory-pathways the median ComPPI score is highest for the corresponding Reactome localization. However, in Compartments the cytosol has the highest median score across the majority of all Reactome localizations.

There is also need for topological or at least some other source of information to effectively predict localizations in pathway databases from protein localization data. For 9.5% and 11.5% of total interactions in Reactome and PathBank, respectively, there exists another interaction between the same proteins in another pathway that has a different localization. When expanded to protein localization data, for over 40% of interactions in both Reactome and PathBank there exists at least 1 other interaction with identical information in ComPPI but a different localization. When using Compartments there are about half as many contradictory interactions by features.

3.5 Pathway Database Prediction

We investigated how well protein localization databases can be used to predict localizations in pathway databases, both to examine the feasibility of pathway-specific localization prediction and to further elucidate the relationship between protein localization databases and pathway databases. Details of the experimental setup can be found in Section 3.2 and are summarized in Figure 3.1.

Three general classes of models were used for localization prediction: graph neural networks, probabilistic graphical models, and traditional classifiers. Details of models and model implementation can be found in Section 3.3. 3 models, the fully connected neural network (FullyConnectedNN), random forest (RF), and logisitic regression (Logit) use no topological information. All 3 of these models instead concatenate the data of each interaction's interactors as its input. The other models, 3 graph neural network models (GAT, GIN, and GCN) and 2 probabilistic graphical models (NaivePGM and TrainedPGM) all use topological information from the pathway where localization is being predicted to encourage interactions near each other to have similar localizations.

All 8 models were tested on both protein localization databases and Uniprot keyword features (see Section 3.3) with the exception of the NaivePGM model, which could not use the Uniprot keyword features as it interprets input features directly as probabilities. The 2 pathway databases Reactome and PathBank were tested on, resulting in a total of 46 runs.

Data was split using 5-fold cross validation by pathway, so every individual pathway was completely in the training set or test set. As many protein-protein interactions occur across multiple pathways, this splitting strategy did result in the same interaction being included in the training set and test set at the same time. However, as high proportion of interactions had contradictory localizations across different pathway databases as mentioned above, especially in their feature representations, this was not considered a source of data leakage. The identical interaction in the test set and training set were likely to have different localizations because of the pathways they belonged to.

Figures 3.6–3.9 show predictive performance on PathBank and Reactome pathways, respectively. Models were overall able to achieve better performance on PathBank than Reactome, and generally models' perforance in predicting PathBank interaction localizations was more consistent accross pathways. However, on both datasets all models' F1 scores had high variance across pathways. Except for logistic regression in PathBank, all models got at least some pathways completely correct and some pathways completely wrong across all databases and feature sets. In PathBank the graph neural nework models, GCN, GAT, and GIN, were best

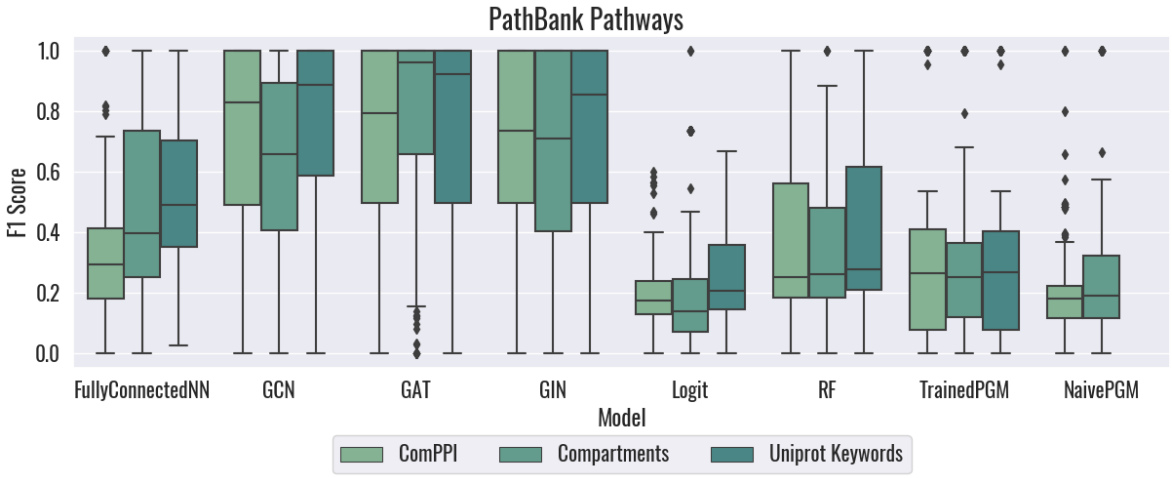

Figure 3.6: F1 score of predictive performance on PathBank localizations across all 427 used PathBank pathways. Scores are calculated per pathway; the distribution of scores is shown for each model.

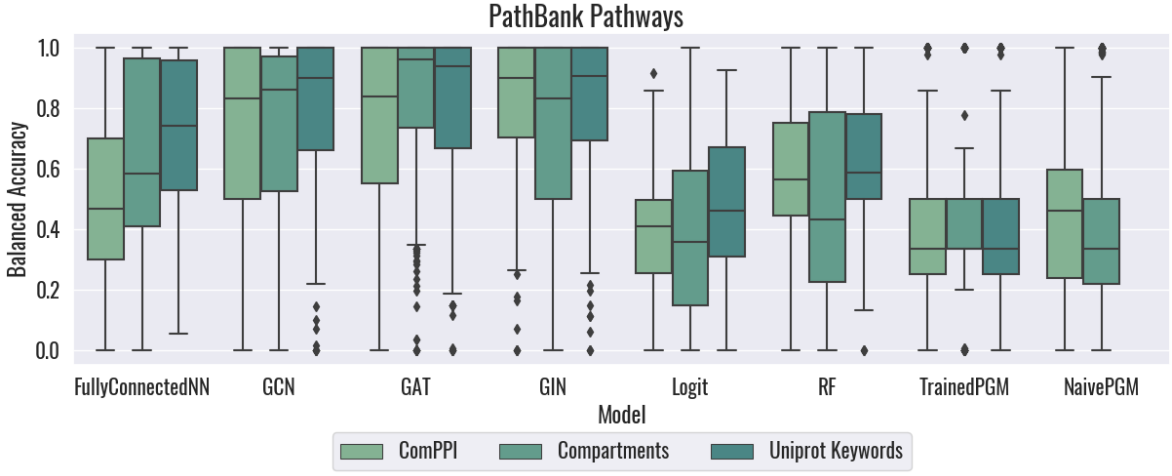

Figure 3.7: Balanced accuracy of predictive performance on PathBank localizations across all 427 used PathBank pathways. Scores are calculated per pathway; the distribution of scores is shown for each model.

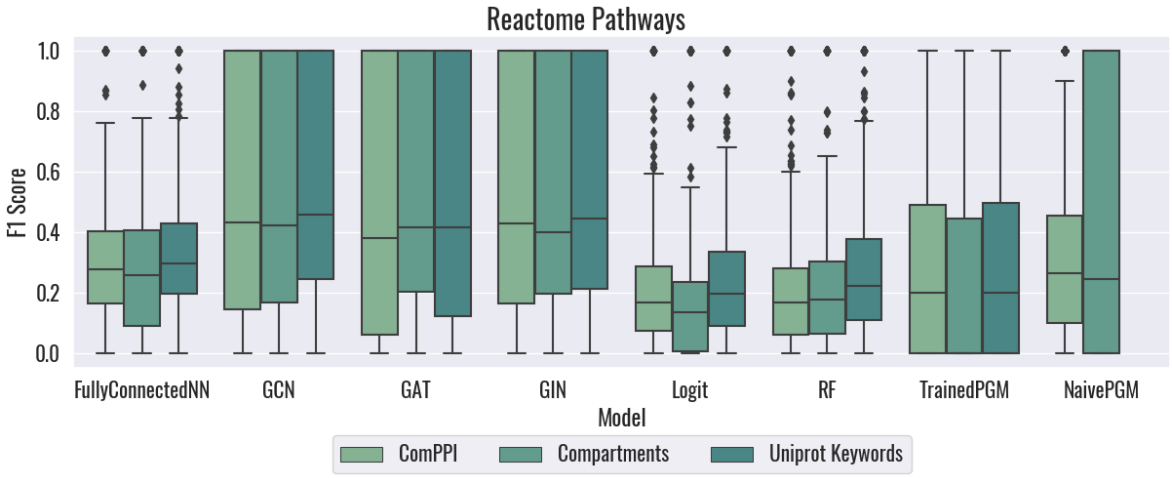

Figure 3.8: F1 scoreof predictive performance on Reactome localizations across all 918 used Reactome pathways. Scores are calculated per pathway; the distribution of scores is shown for each model.

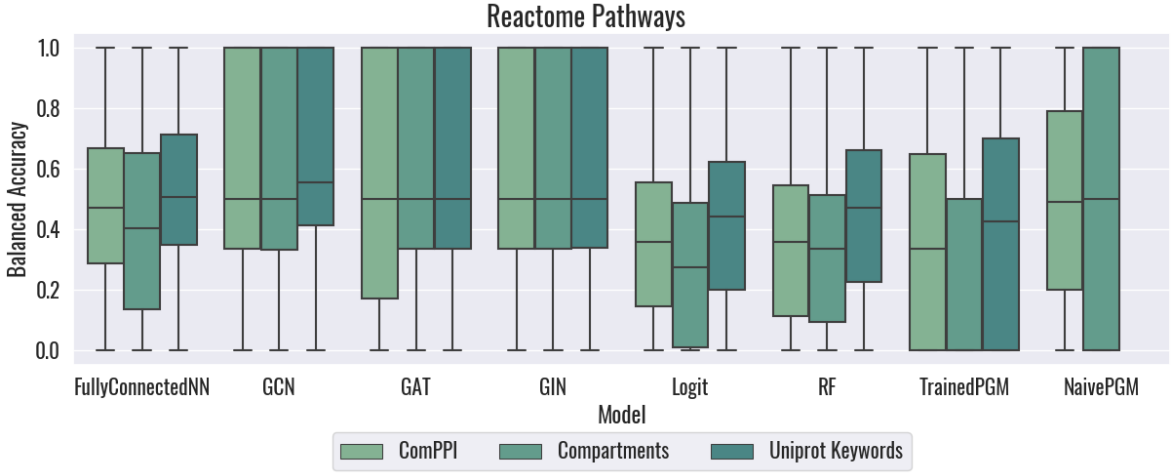

Figure 3.9: Balanced accuracy of predictive performance on PathBank localizations across all 427 used PathBank pathways. Scores are calculated per pathway; the distribution of scores is shown for each model.

performing across both metrics, but still had a high variance in their performance between different pathways. Performance in Reactome was generally low.

On both datasets graph neural networks outperformed all other models in F1 score. In PathBank the graph neural networks also performed best in balanced accuracy, but in Reactome while graph neural networks had the hightest balanced accuracy the FullyConnectedNN and NaivePGM had comparable performance. Generally, the FullyConnectedNN outperformed other models that did not use pathway topology and the probabilistic graphical models. Logistic regression and the trained probablistic graphical models were the worst performing. The worst-performing model overall was the TrainedPGM model when compared on pathway-stratified performance. It should be noted, however, that when calculating performance by pathway, the size of each pathway is not taken into account. This means that edges in very small pathways can have an outsized effect on total performance.

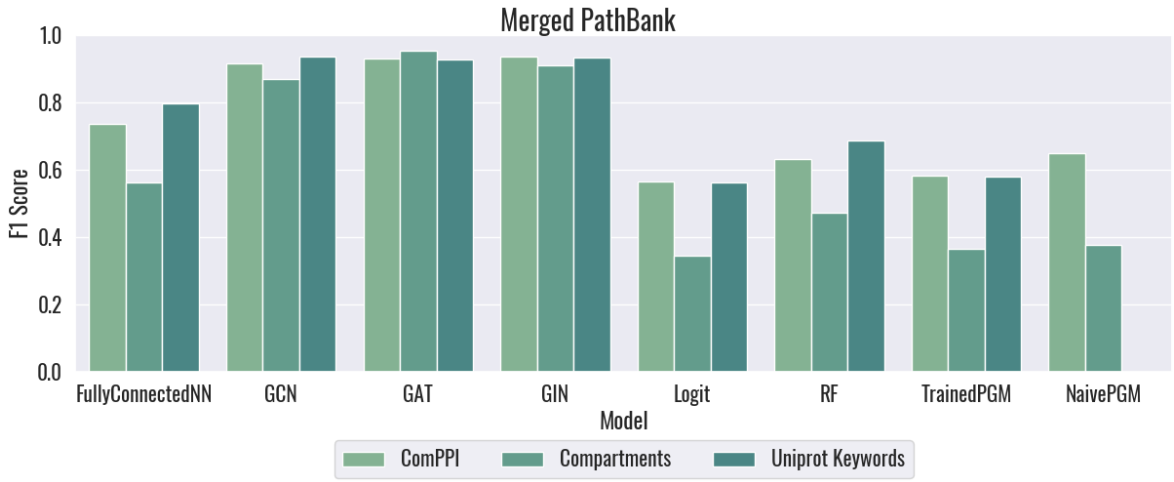

Figure 3.10: F1 score of predictive performance on PathBank localizations. All pathway edges are merged and measured together, resulting in 97792 edges total.

Figures 3.10–3.13 show F1 scores for each model aggregated from all pathways, where all edges across all pathways are used for a single performance calculation. When aggregated in this way all non-neural network models perform comparably. This suggests that the probabilistic graphical models, and the TrainedPGM model in particular, struggled with small pathways.

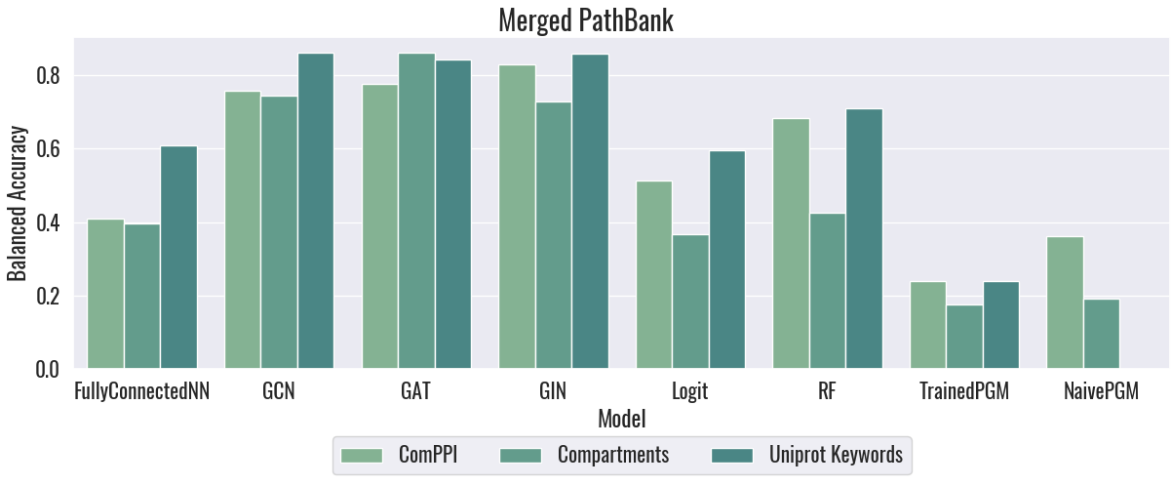

Figure 3.11: Balanced accuracy of predictive performance on PathBank localizations. All pathway edges are merged and measured together, resulting in 97792 edges total.

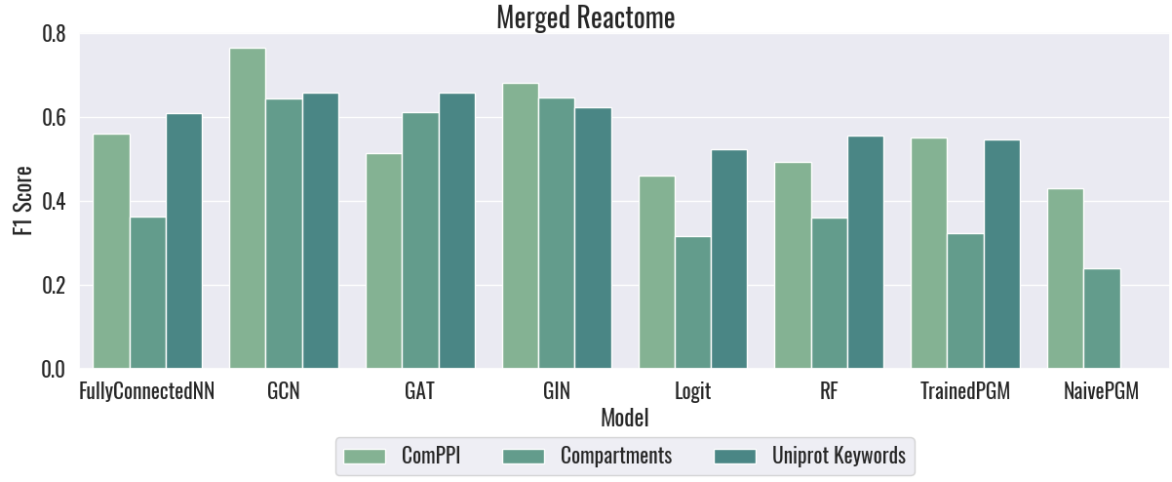

Figure 3.12: F1 score of predictive performance on Reactome localizations. All pathway edges are merged and measured together, resulting in 83855 edges total.

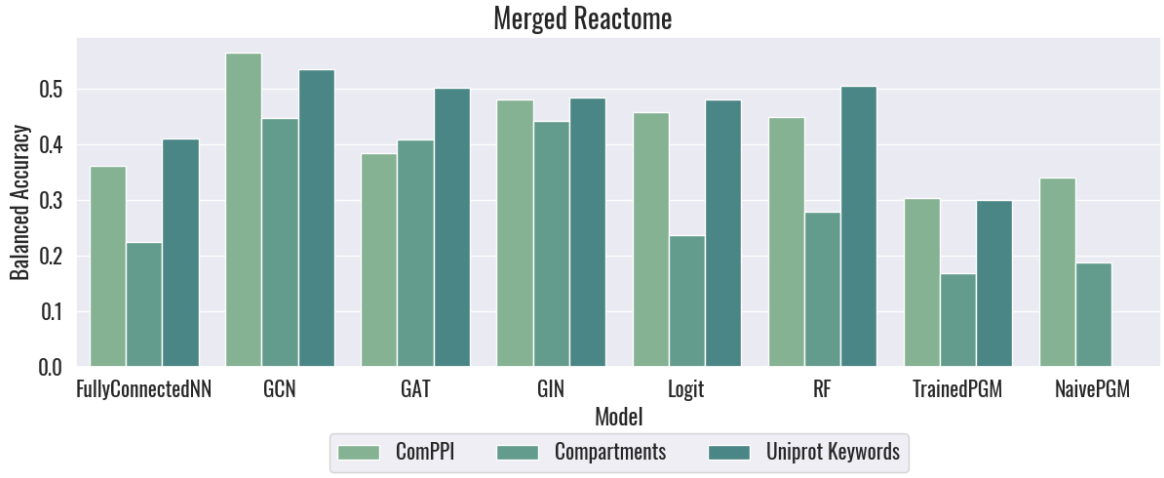

Figure 3.13: Balanced accuracy of predictive performance on Reactome localizations. All pathway edges are merged and measured together, resulting in 83855 edges total.

Pathway Smoothness

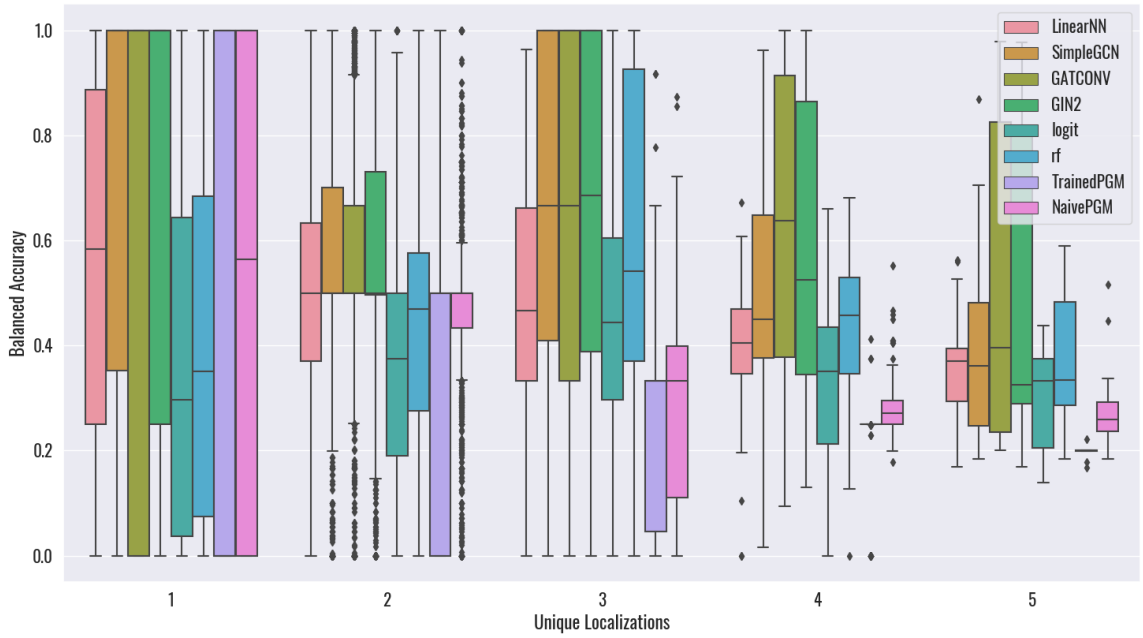

Figure 3.14: Balanced accuracy for each model by the number of unique locations in the true pathway.

The number of real and predicted unique localizations in each pathway also had a large effect on model performance. This can be thought of as the smoothness of the real or predicted

localizations in a pathway, or how strong the tendency is in a pathway for edges nearby to have the same localization. Ideally, a model would be able to detect that a pathway exists entirely in a single subcellular location and aggressively smooth its localization predictions over the pathway. When balanced accuracy is stratified by the number of unique localizations in each pathway, as shown in Figure 3.14, over and undersmoothing of different models' predictions can be seen more clearly.

Pathways with a single localization had the widest range of performance within each model. More extreme performances, at or nearly at 1.0 or 0.0 for these pathways, indicate that the model correctly predicted that the pathway had only a single localization. GAT and the TrainedPGM correctly guessed the number of localizations in all pathways with a single unique localization, though the TrainedPGM often incorrectly guessed which localization it was. The median balanced accuracy across all localizations was exactly 0.5 for all models except the random forest and logistic regression model among pathways with exactly 2 localizations. This is due to models predicting most pathways with exactly 2 localizations to have only 1 localization.

Overall, models without any method to transfer information across a pathway, the logistic regression, random forest, and fully connected neural network models, tended to undersmooth within each pathway. Figure 3.15 shows the distribution in the number of unique localizations in each pathway predicted by the different classes of models. The distributions for the RF, Logit, and NaivePGM models are right-skewed as compared to the true distribution, with a sizable proportion of pathways being predicted to have 5 or 6 different localizations. This is unsurprising, as these models contained no topological information with which to encourage proteins belonging to the same pathway to have the same localization. These models greatly underestimated the proportion of pathways with a single localization.

The TrainedPGM in particular tended to oversmooth. It predicted almost all pathways as having a single localization across both datasets. It also performs particularly poorly for pathways with 2 and 3 unique localizations. This is likely due to these pathways being more evenly split between multiple localizations than those with 4 or 5 localizations, resulting in a

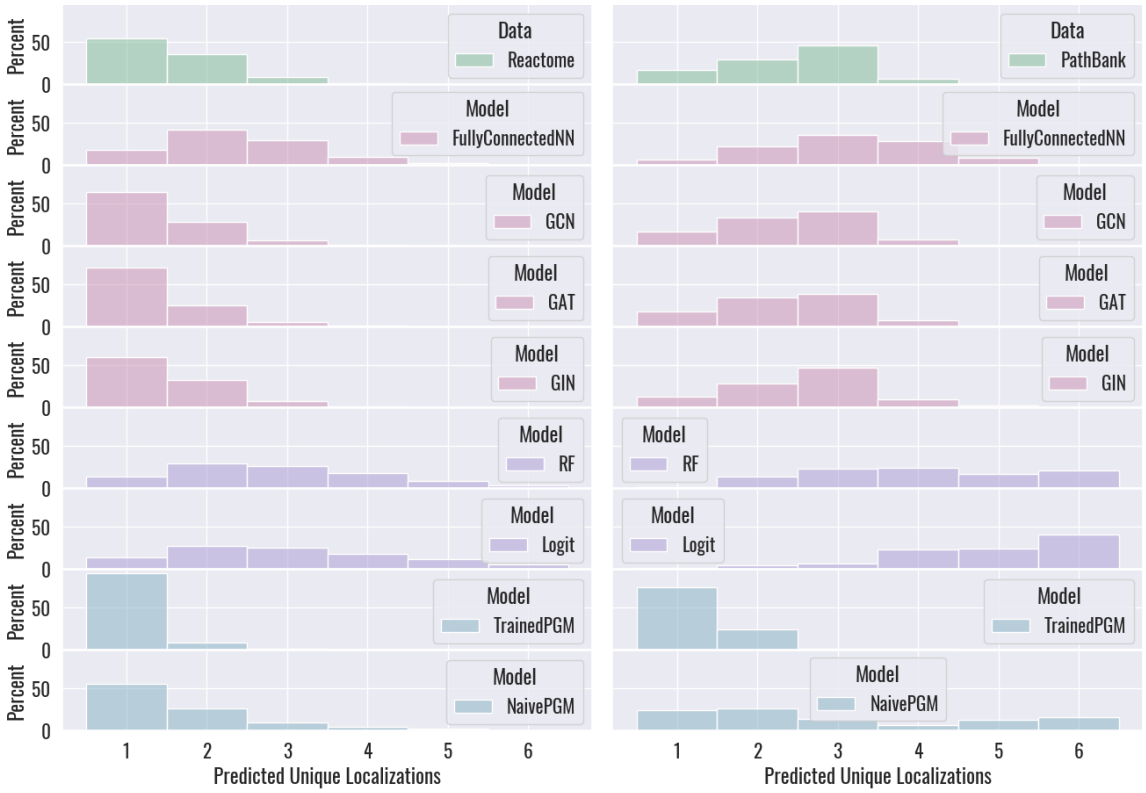

Figure 3.15: Distribution of the number of unique localizations in each pathway database, and as predicted by each model on each pathway database.

single localization prediction to perform poorly on these pathways.

Missing Data

Another feature of the models that use pathway topology is that they are also able to infer localizations for interactions where one or both nodes which make up the edge are missing information in the protein localization database. This can be due to the pathway members being non-protein biological entities such as metabolites, or due to the database not containing a particular protein.

Figure 3.16 shows the F1 score for all models stratified by the number of missing nodes for each interaction. Both the logistic regression and random forest models performed extremely poorly on missing data, correctly predicting the localization of almost no edges. In PathBank most models performed better on edges with more data, with the notable exception of the

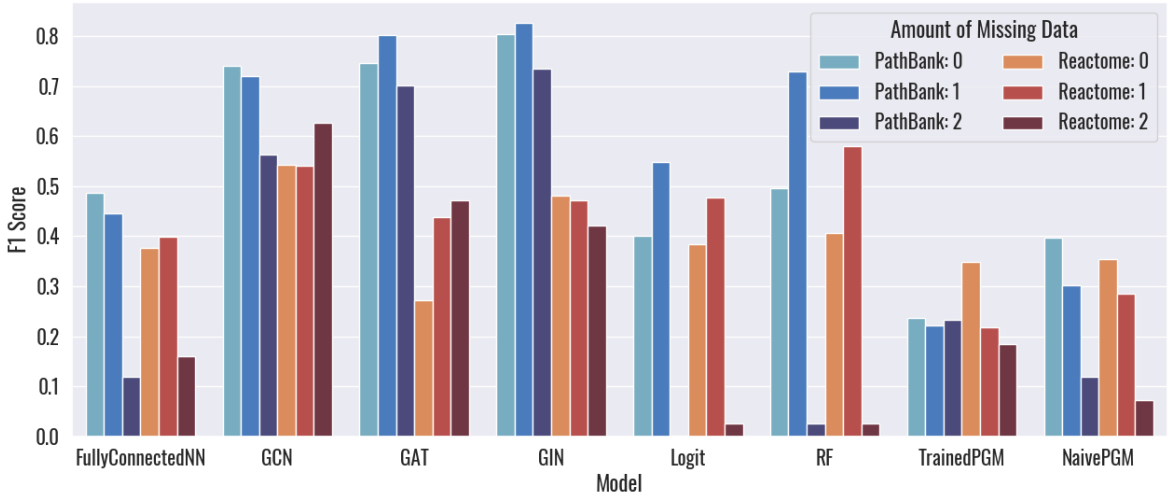

Figure 3.16: Predictive performance of models stratified by the amount of missing data on each edge, either no missing data (0), 1 node has missing data (1), or both nodes have missing data (2).

logistic regression and random forest models. These models performed significantly better when one of the two nodes was missing data.

In Reactome, most models performed better on interactions with missing data than on interactions without missing data. The graph neural networks especially performed best on interactions whose nodes were both missing data.

Figure 3.17 shows how interactions with missing data are distributed among each dataset and pathways with different number of unique localizations. Both PathBank and Reactome contain a significant proportion of interactions with missing data. In PathBank, interactions with different amounts of missing data are distributed across pathways with different numbers of unique localizations relatively identically.

However, in Reactome interactions with no missing data (missing node data $= 0$ in Figure 3.17) are over-represented in pathways with 2 or more unique localizations, while interactions with completely missing data (missing node data $= 2$ in Figure 3.17) are over-represented in pathways with a single unique localization. Thus, graph neural networks are likely performing better on interactions with more missing data due to their overall high performance on pathways with a single localization.

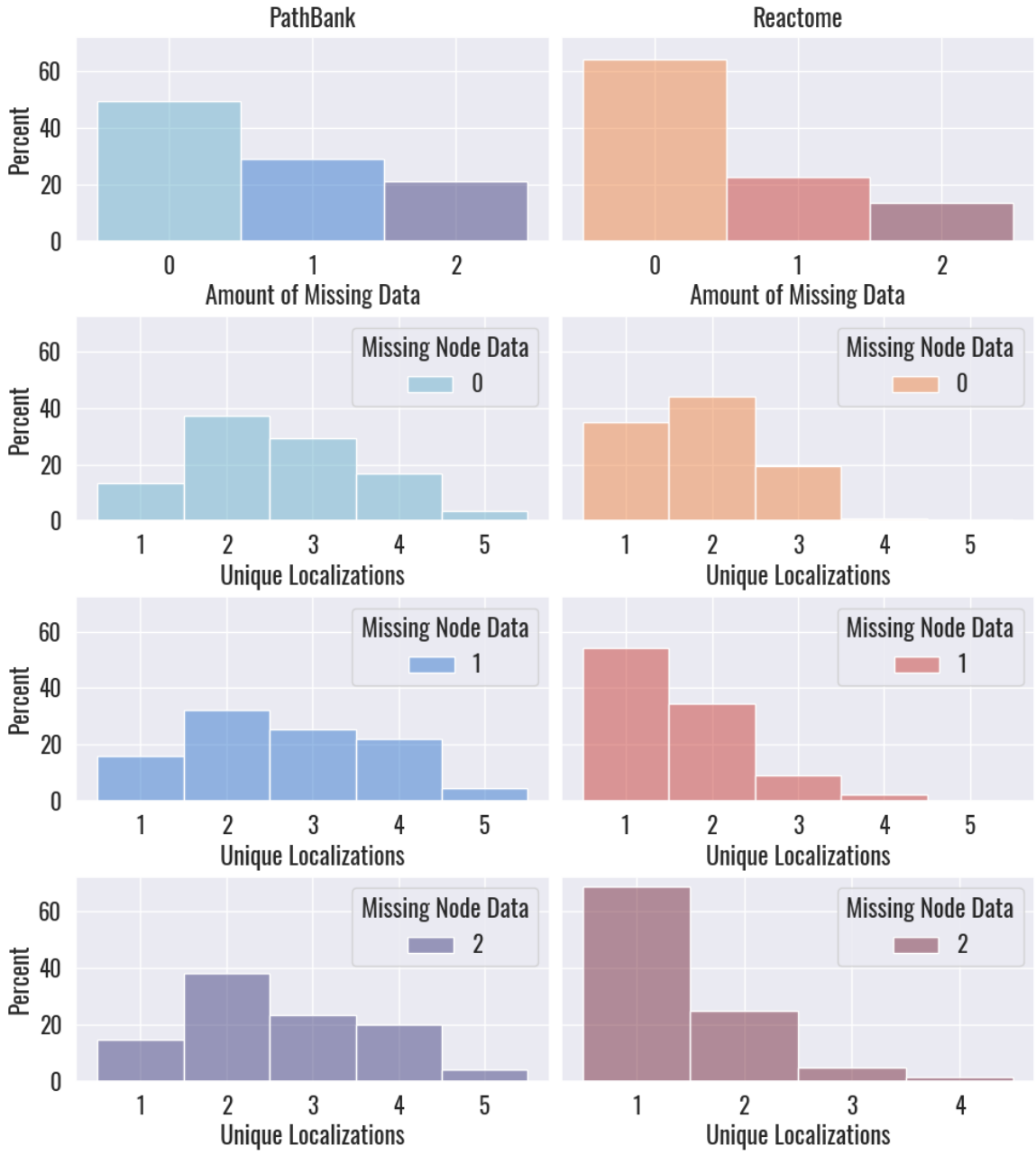

Figure 3.17: Distribution of missing data in Reactome and Pathbank, and how missing data is spread over pathway with different numbers of unique localizations. For edges whose nodes have either no missing data (0), 1 node has missing data (1), or both nodes have missing data (2) the distribution of pathways they belong to is shown by the number of unique localizations. The top histograms show the overall distribution of missing data in PathBank and Reactome.

3.6 Spatial Proteomics Case Study

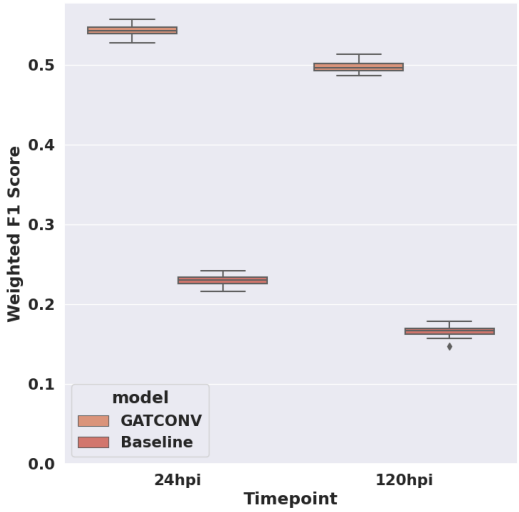

Figure 3.18: Predictive performance of graph attention network (GAT) on spatial mass spec data of viral infection at 24 and 120 hours post infection (hpi). The baseline model is the performance when always guessing the most common localization in the dataset. Models were trained on marker proteins and tested on non-marker proteins within pathways created with OmicsIntegrator2.

In order to examine how prediction localization at the pathway level could be used in a realistic biological setting, we performed a case study using spatial proteomic data on primary fibroblasts during human cytomegalovirus (HCMV) infection (Jean Beltran et al., 2016). We investigated the ability of a predictive model to infer localizations in the context of viral infection, potentially bypassing the need to collect spatial proteomic data. Here, we performed pathway reconstruction (Chapter 2) based on label-free mass spectrometry data, which measured protein abundance across the entire fibroblast at 24 hour and 120 hours after infection. We then trained one of the best performing models from the pathway database prediction task, GAT, on a set of publicly available marker proteins to attempt to predict the subcellular localizations derived from spatial mass spectrometry data.

The combined top pathways contained a total of 590 edges with localization information at 24hpi and 386 edges with localization information at 120hpi. There were 249 edges with

localization information in common between the two timepoints.

Figure 3.18 shows performance of the GAT model as compared to a baseline model that always predicts the most frequent localization. Overall predictive performance was low, but well above the baseline. Over half of interactions were predicted to be in the correct localization.

The GAT model was also able to moderately detect when an interaction changed localizations between the two timepoints. There were 35 interactions that changed localization between the two time points. Of these, GAT correctly predicted 14 that did change localization, a recall of 0.4. However, it had low precision, predicting 65 localization changes that did not occur.

3.7 Conclusions and Future Work

While there is some correspondence between protein localization databases and localization data in pathway databases, prediction of pathway localization remains difficult. Complex graph neural network models were required to achieve high predictive performance on PathBank pathways, and all models performed poorly in predicting Reactome localizations. Where high predictive performance is possible it requires a complex model with topological data. There are a number of possible reasons for this misalignment between localization information in pathway databases and protein localization databases.

While the best-performing models include topological information, implying that topology is needed to bring context to protein localization, it is possible that other types of data or contextual information are needed. Adding additional protein-level generic information in the form of Uniprot keywords appeared to only slightly improve performance, but this information may not provide needed context. Information such as tissue or cell-type specific localization may be an additional necessary layer to fully realize context-specific localization. However the possible necessity of contextual information such as tissue type, when most pathways in pathway databases are provided independent of tissue type, may be infeasible and further highlights the discrepancy between these databases.

The curated pathway diagrams that make up pathway databases are useful summarizations and abstractions of biological processes. However, these pathways often do not line up with specific results of biological experiments. For instance, protein phosphorylation has poor overlap with relevant pathways during EGFR signaling (Köksal et al., 2018), T cell receptor signaling (Cao et al., 2012), insulin signaling (Humphrey et al., 2015), and TGF-β signaling (D'Souza et al., 2014). As the protein localization databases used here rely more directly on experimental data than the curated pathway databases, this poor overlap could simply also manifest in subcellular localization. While this would not directly impact case study performance, as pathway database data was not used in training or pathway reconstruction, it could indirectly affect performance through model selection. The best model from pathway database prediction was used in the case study, and pathway parameter adivising, which was used to select pathway reconstruction parameters, attempts to find pathways whose topology most closely matches that of pathways in pathway databases.

The protein localization databases may also be too noisy and general for context-specific localization prediction. While some signal does exist, the wide range of distributions for ComPPI and Compartments scores across different pathway localizations highlights the noisiness of the prediction problem. It may be the case that the predictive task needs features that also include contextual information.

A possible explanation for the difference in performance between PathBank and Reactome is that PathBank is only partially labeled with subcellular localizations, while Reactome is fully labeled. While neither database makes their reasoning explicitly clear for their strategy to including subcellular localizations, it may the the case that, as Reactome requires all interactions to have a localization, Reactome's pathways are forced to include localization information even when an interactions subcellular location may not be known with high confidence. PathBank, however, can choose to omit low-confidence or unknown interaction localizations.

Despite the difference in score distribution and baseline performance between Compartments and ComPPI when used to predict subcellular localization, once used as inputs to

models both sets of features performed well, the highest performing models having almost identical performance between the two feature sets. One possible reason for the difference in baseline performance is that ComPPI is specifically built to be utilized in protein-protein interaction analyses, so the database may be better optimized predicting interaction-level information. However, classifiers were still able to find relationships between Comparments data and pathway database data.

The level of smoothing needed within each pathway was a major factor in each model's performance. Graph neural networks were much more likely to identify the correct number of unique localizations in each pathway than other models. Models without any topological information were unable to add any smoothing and thus tended to over-predict how many localizations were present in each pathway. While the naivePGM model generally undersmoothed, the trainedPGM model oversmoothed, predicting almost all pathways to only consist of a single localization. The parameters in the TrainedPGM conditional random field model are learned using pseudo-likelihood, which approximates maximum likelihood estimation (MLE). This approximate MLE does not account for class imbalance and will tend towards the majority class. Thus, this oversmoothing may be the result of the training method. Alternate probabilistic graphical model parameter learning methods that are better able to take class imbalance into account could improve model performance.

The extremely poor performance of random forests and logistic regression on edges with completely missing data may be due to how missing data is represented. Missing data was was represented by giving nodes a uniform distribution of scores for all localizations. While the neural network was able to differentiate and treat these uniform inputs differently, in the random forest and logistic regression models the classifier was unable to separate missing data from non-missing data. An alternate missing data representation would have likely resulted in improved performance for these models, such as making an additional binary feature indicating whether or not data was missing.

One surprising facet of predictive performance was how models behaved with missing node data. The RF and Logit models consistently had higher predictive performance when

one node was missing data than when both had data. It is unclear why this occurs, however, one possibility is that nodes in pathways with missing data are more likely to be non-protein biological entities such as metabolites. It could be the case that, even with less data, these interactions have more consistent localizations. More investigation is needed to fully understand the reason behind this trend.

Performance on predicting case-study localization data much more closely resembles performance in predicting Reactome localizations than in predicting PathBank localizations, though it is likely that both of these tasks are just very difficult. While overall performance was low, and likely too low to be useful in the context of a biological analysis, some predictive signal does exist in the problem. The graph neural network was able to correctly predict over half of total protein localizations from label-free mass spec data alone when combined with pathway reconstruction analysis. Additionally, while predicting the exact localizations associated with translocation events in proteins between timepoints was difficult, the model was able to have some success predicting which proteins had a translocation event between timepoints. Predicting these translocation events would be impossible only directly using data from a protein localization database, as there would be no difference between timepoints. This is especially impressive when it is considered that the only signal of the label-free mass spec data present in the model was embedded in constructed pathway topology.

The conversion of pathways from hypergraphs to graphs greatly impacted the class topology and class distribution of Reactome and PathBank pathways. Treatment of protein complexes can lead to orders of magnitude difference in the number of edges in the resultant pathways. Consideration of the desired analysis task is important when making this decision. For instance, the conversion we chose, creating protein complex nodes to represent complexes, removes node information but better preserves the edge structure and balance in the pathway. An analysis task focused specifically on nodes may want a conversion that better preserves node information at the possible cost of edge information. An important future work would be to consider these conversions in a more systemic way and quantify the hypergraph properties they alter or keep invariant.

An alternate direction in the treatment of hypergraphs would be moving prediction directly onto the hypergraph, removing any conversion step. While hypergraph neural networks exist (Feng et al., 2019; Yadati et al., 2019), they are much less mature than graph neural networks. However, this would preserve information lost from the hypergraph conversion.

While this work attempts to address pathway context in localization prediction, which could be considered the context of biological function, there are other contexts which would be valuable to explore with regards to protein localization. Single-cell spatial proteomics experiments have previously found proteins to vary by as much as 16% in either expression or spatial distribution between cells undergoing the same process in the same tissues (Thul et al., 2017). Using single-cell measurements to construct context-specific pictures of protein localization could provide an additional layer of information to single-cell analyses. Another contextual lens to consider is abnormal cellular function. This could include examining mutational effects or other diseases which cause abnormal localizations, such as cancer (Blise et al., 2021). Pathways or other contextual information could be a way to use available information to be able to explore atypical cellular states.

Table 3.2: All parameter values used.

Model	Dataset	Features	Parameter	Value
FullyConnectedNN	Reactome	ComPPI		
			lRate	0.002
			l_depth	2
			dropout	0.500
			dim	83
			activation	tanh
FullyConnectedNN	Reactome	Compartments		
			lRate	0.008

Continued on next page

Table 3.2: All parameter values used. (Continued)

Model	Dataset	Features	Parameter	Value
			l_depth	1
			dropout	0.500
			dim	82
			activation	tanh
FullyConnectedNN	Reactome	Uniprot KW		
			lRate	$8.38e-04$
			l_depth	3
			dropout	0.500
			dim	98
			activation	relu
FullyConnectedNN	PathBank	ComPPI		
			lRate	0.010
			l_depth	1
			dropout	0
			dim	41
			activation	tanh
FullyConnectedNN	PathBank	Compartments		
			lRate	0.009
			l_depth	2
			dropout	0

Continued on next page

Table 3.2: All parameter values used. (Continued)

Model	Dataset	Features	Parameter	Value
			dim	103
			activation	relu
FullyConnectedNN	PathBank	Uniprot KW		
			lRate	0.006
			l_depth	5
			dropout	0.500
			dim	81
			activation	tanh
GCN	Reactome	ComPPI		
			lRate	0.004
			l_depth	1
			dropout	0
			dim	81
			c_depth	6
GCN	Reactome	Compartments		
			lRate	$1.48e-04$
			l_depth	1
			dropout	0.500
			dim	108
			c_depth	2

Continued on next page

Table 3.2: All parameter values used. (Continued)

Model	Dataset	Features	Parameter	Value
GCN	Reactome	Uniprot KW		
			lRate	0.003
			l_depth	4
			dropout	0
			dim	87
			c_depth	3
GCN	PathBank	ComPPI		
			lRate	0.002
			l_depth	1
			dropout	0
			dim	44
			c_depth	8
GCN	PathBank	Compartments		
			lRate	0.006
			l_depth	2
			dropout	0
			dim	96
			c_depth	1
GCN	PathBank	Uniprot KW		
			lRate	0.003

Continued on next page

Table 3.2: All parameter values used. (Continued)

Model	Dataset	Features	Parameter	Value
			l_depth	3
			dropout	0
			dim	108
			c_depth	5
GAT	Reactome	ComPPI		
			lRate	0.003
			l_depth	2
			dropout	0
			dim	48
			c_depth	7
			num_heads	4
GAT	Reactome	Compartments		
			lRate	$2.78e-04$
			l_depth	4
			dropout	0.500
			dim	46
			c_depth	1
			num_heads	5
GAT	Reactome	Uniprot KW		
			lRate	0.010

Continued on next page

Table 3.2: All parameter values used. (Continued)

Model	Dataset	Features	Parameter	Value
			l_depth	2
			dropout	0
			dim	31
			c_depth	4
			num_heads	1
GAT	PathBank	ComPPI		
			lRate	0.003
			l_depth	4
			dropout	0.500
			dim	45
			c_depth	4
			num_heads	3
GAT	PathBank	Compartments		
			lRate	0.002
			l_depth	3
			dropout	0.500
			dim	43
			c_depth	3
			num_heads	4
GAT	PathBank	Uniprot KW		

Continued on next page

Table 3.2: All parameter values used. (Continued)

Model	Dataset	Features	Parameter	Value
			lRate	0.004
			l_depth	1
			dropout	0
			dim	39
			c_depth	2
			num_heads	4
GIN	Reactome	ComPPI		
			lRate	0.005
			l_depth	2
			dropout	0
			dim	95
			c_depth	3
GIN	Reactome	Compartments		
			lRate	$9.77e-04$
			l_depth	4
			dropout	0.500
			dim	24
			c_depth	1
GIN	Reactome	Uniprot KW		
			lRate	0.001

Continued on next page

Table 3.2: All parameter values used. (Continued)

Model	Dataset	Features	Parameter	Value
			l_depth	3
			dropout	0.500
			dim	60
			c_depth	2
GIN	PathBank	ComPPI		
			lRate	0.002
			l_depth	4
			dropout	0
			dim	68
			c_depth	1
GIN	PathBank	Compartments		
			lRate	0.001
			l_depth	3
			dropout	0.500
			dim	102
			c_depth	3
GIN	PathBank	Uniprot KW		
			lRate	$9.86e-04$
			l_depth	2
			dropout	0

Continued on next page

Table 3.2: All parameter values used. (Continued)

Model	Dataset	Features	Parameter	Value
			dim	80
			c_depth	1
Logit	Reactome	ComPPI		
			C	0.620
			class_weight	balanced
			penalty	l2
			tol	$1.00e-06$
Logit	Reactome	Compartments		
			C	91.893
			class_weight	balanced
			penalty	l2
			tol	0.062
Logit	Reactome	Uniprot KW		
			C	7.604
			class_weight	balanced
			penalty	l2
			tol	0.078
Logit	PathBank	ComPPI		
			C	0.044
			class_weight	balanced

Continued on next page

Table 3.2: All parameter values used. (Continued)

Model	Dataset	Features	Parameter	Value
			penalty	l2
			tol	$1.84e-05$
Logit	PathBank	Compartments		
			C	0.451
			class_weight	balanced
			penalty	l2
			tol	$4.71e-06$
Logit	PathBank	Uniprot KW		
			C	33.176
			class_weight	balanced
			penalty	l2
			tol	0.033
RF	Reactome	ComPPI		
			class_weight	balanced
			max_depth	3
			min_samples_split	2
			n_estimators	76
RF	Reactome	Compartments		
			class_weight	balanced
			max_depth	9

Continued on next page

Table 3.2: All parameter values used. (Continued)

Model	Dataset	Features	Parameter	Value
			min_samples_split	3
			n_estimators	81
RF	Reactome	Uniprot KW		
			class_weight	balanced
			max_depth	6
			min_samples_split	3
			n_estimators	100
RF	PathBank	ComPPI		
			class_weight	balanced
			max_depth	10
			min_samples_split	10
			n_estimators	72
RF	PathBank	Compartments		
			class_weight	balanced
			max_depth	10
			min_samples_split	10
			n_estimators	92
RF	PathBank	Uniprot KW		
			class_weight	balanced
			max_depth	10

Continued on next page

Table 3.2: All parameter values used. (Continued)

Model	Dataset	Features	Parameter	Value
			min_samples_split	10
			n_estimators	58

Chapter 4

Practical and approachable computational education for biologists.

Work on the ML4Bio workshop was was performed in collaboration with Fangzhou Mu, Rosemary S Russ, Milica Cvetkovic, Debora Treu, and Anthony Gitter (Magnano et al., 2022).

4.1 Motivation and Related Work

The increasing prevalence and importance of machine learning in biological research has created a need for machine learning training resources tailored towards biological researchers. However, existing resources are often inaccessible, infeasible, or inappropriate for biologists because they require significant computational and mathematical knowledge, demand an unrealistic time-investment, or teach skills primarily for computational researchers. Machine learning (ML) is a powerful tool for analyzing biological data and is increasingly popular in biological research. Biological publications using ML have increased exponentially over the past decades (Walsh et al., 2021). In 2017 almost 90% of 704 NSF principal investigators reported that they "are currently or will soon be analyzing large datasets (Barone et al., 2017)." However, the most commonly reported unmet needs were training-based. As of 2017, only about a quarter of life-sciences training programs taught necessary skills for data stewardship (Brazas et al., 2017). In the United States, the National Science Foundation and National Institutes of

Health have recognized the need for training at the intersection of ML and biology (National Institutes of Health, 2021b; National Science Foundation, 2020; National Institutes of Health, 2021a). The breadth of this gap means that biologists often lack the computational skills that are prerequisites for existing ML educational resources (Dinsdale et al., 2015). This gap can lead to missed insights from biological data (Chang, 2015) and contributes to the improper use of ML in biology (Jones, 2019; Walsh et al., 2021). Despite its popularity among computational researchers, machine learning remains elusive to experimental biologists, who form the majority of the life sciences research community, leaving powerful computational tools underappreciated and data generated in wet labs underexplored (Chicco, 2017).

Many resources have been created to help researchers acquire skills in ML. Comprehensive resources such as textbooks (Mitchell, 1997; Raschka et al., 2022; Murphy, 2022) and online courses require significant time investment, which may not be feasible for active researchers, and teach to a depth that is often unneeded for biological researchers. Other resources such as graphical research and education tools (Roushangar and Mias, 2018; Elia et al., 2021; Demšar et al., 2013; Gu et al., 2021), workshops (Teal et al., 2015), and written guides and reviews (Liu et al., 2019; Greener et al., 2022; Vercio et al., 2020),still often focus on coding, mathematics, or running ML. While these resources are important, not all biological researchers will necessarily need to code and run ML experiments independently (Mulder et al., 2018).

Thus, resources are needed that provide researchers with skills to productively navigate partnerships and collaborations around ML without necessarily directly executing ML research themselves. When designing these resources, it is essential to consider what skills are needed to interpret research that involves ML, communicate with collaborators about ML, and identify biological questions ML can solve. Resources centered around coding, the mathematical underpinnings of ML, or practical advice for using a certain technique do not necessarily fulfill this role.

We created the Machine Learning for Biologists (ML4Bio) workshop to introduce ML to biological researchers. The workshop aims to provide the skills biologists need to be active researchers in a landscape where ML is increasingly prevalent. It focuses on practical

research skills such as reading academic papers that use ML and drawing conclusions from ML experiments. We designed the ML4Bio workshop to be approachable and a reasonable time investment; it requires minimal mathematical and computational background and runs for 5 hours over 2 days. A key feature of the ML4Bio workshop is custom software based on the scikit-learn Pedregosa et al. (2011) library, which allows participants to explore and experiment with classification through a graphical interface without computational fluency.

The target audience for the ML4Bio workshop are biological researchers with no computational experience. While some of these researchers may wish to continue their education in machine learning to gain competencies similar to those that International Society for Computational Biology (ISCB) identifies as a "discovery scientist", others may take the ML4Bio workshop and realize that that machine learning does not suit their research needs. However, these researchers who do not continue their education will still gain valuable skills in interpreting and machine learning and identifying problems well-suited to machine learning in the future.

We used an iterative design process to refine the workshop over a series of 5 pilot studies from 2018 to 2021. The first 4 pilots were conducted in person, and the fifth was conducted online over Zoom. These iterations gave us insight into how to better align the workshop to our overall goals and address the needs of the workshop's audience. We then evaluated the effectiveness of the workshop over 3 additional sessions in 2021.

Participants were generally able to achieve the learning goals of the ML4Bio workshop and especially reported an increase in interest and confidence interacting with ML research. We feel that this success hinges on the workshop's approachability, careful design, and flexibility. The ML4Bio workshop effectively introduces ML to biological researchers, preparing them for future learning, collaboration, and comprehension of ML experiments in biological domains.

4.2 Workshop Design

Learning Goals

The ML4Bio workshop began with the intention to create a short, intensive workshop that empowers biological researchers to operate in fields where ML is increasingly common and identify where they might pursue ML collaborations in their own research. Rather than tackling the entire field of ML, we chose to focus on classification to limit the scope of the workshop to 1-2 days. The original learning goals of the workshop involved identifying problems in computational biology, understanding all parts of a typical ML workflow, being able to compare specific classifiers, performing model selection, evaluating a model on new data, and judging the use of ML in biological contexts. These learning goals were defined based on our professional experiences interacting with biological researchers around ML and through our observations of common challenges in published biological papers that use ML.

However, early iterations of the workshop revealed that some of these objectives require more extensive coding and mathematical background than is typical of our biological researcher audience. Additionally, early pilots showed that the scope of these learning goals was too large; we typically ran out of time before participants could learn the terminology and skills of the workshop. The workshop's prerequisites and scope needed to be realistic for the backgrounds and time constraints of biological researchers. Thus, we refocused on preparedness for ML research instead of fully equipping participants to perform ML research independently. Data from our initial workshops, coupled with our analysis of the strengths and weaknesses of existing ML resources, led us to embrace 3 key design principles (Brown and Campione, 1996) for our workshop. Specifically, we were committed to our workshop (a) focusing on preparedness over fluency or expertise, (b) necessitating minimal coding and mathematical background, and (c) requiring low time investment. We then used a backwards design paradigm (Wiggins and McTighe, 1998) to realign our learning goals with these design principles. The result is the following 4 learning goals whose justification and purpose we discuss in detail below. ML4Bio workshop participants should be able to:

1. Identify machine learning applications and differentiate aspects of a machine learning workflow.

2. Examine a machine learning problem for common factors that influence model selection and problem difficulty.

3. Discover major methodological flaws in a machine learning experiment presented in an academic paper.

4. Gain confidence in and appreciation for machine learning in biology.

Learning Goal 1: Characterizing common steps of ML workflows—data pre-processing, training and model selection, and testing and evaluation—gives participants a basis for understanding how ML works and provides a framework for dissecting and understanding unfamiliar ML concepts in the future. Thus, we consider characterizing a ML workflow as an important objective for preparing participants. Additionally, while familiarity with ML terminology is important for research comprehension and communication with collaborators, participants do not need to deeply know all ML terminology by the end of the workshop. As long as participants can generally identify parts of ML and a ML workflow, they are prepared to learn the terminology that is used by their collaborators and is most relevant to their research.

Learning Goal 2: Specific classifiers are another area of ML that required careful consideration. We originally chose a number of classifiers that we felt were a good introduction to the types of classifiers available and their limitations. General knowledge of what classifiers can and cannot do, and facets of problems such as linear separability that affect model selection, are required to evaluate problem difficulty, but detailed knowledge of specific classifiers is not. Ultimately, we felt that while the classifiers we had chosen do help demonstrate classifiers' range and limits, participants' general understanding of the factors that influence model selection will help them irrespective of which classifiers are popular in problems they are interested in.

Learning Goal 3: ML in biological applications often lacks proper validation or experimental design, especially when those who use it lack a technical background (Littmann et al., 2020; Whalen et al., 2021). Thus, we consider the ability to find major flaws in a ML experiment, as presented in a research paper, an important part of preparing participants. Since we focus on assessing experiments instead of performing them, we teach the types of evidence presented in a paper that indicate overfitting, data leakage, or improper evaluation metrics. However, subtle errors in ML workflow, such as those that result in indirect data leakage, are difficult to find. Researchers whose primary field is ML often miss indirect data leakage, and consistently detecting data leakage is considered an open challenge in ML(Whalen et al., 2021; Ashmore et al., 2021). Therefore, while participants learn the process of assessing a ML workflow, expecting them to be able to consistently find all subtle methodological errors is likely unrealistic.

Learning Goal 4: Finally, a major focus of the workshop is the affective objective of increasing confidence and appreciation for ML. Affective learning outcomes are those that, as opposed to skills or knowledge, relate to feelings or attitudes. Without any increased appreciation for and interest in ML, participants would likely not pursue ML as a possible way to solve a problem in the future or feel that ML could be valuable. While we do not expect participants to feel fully confident in ML and make it clear to participants that this workshop is an introduction and will not make them an expert, we hope participants will establish a baseline self-efficacy in interpreting research involving ML. We hope to motivate participants to pursue collaborations with ML experts when they identify a problem well-suited to ML.

The resulting ML4Bio workshop that addresses these 4 learning goals is an online 5-hour workshop divided evenly between 2 days. The workshop format is a mixture of software activities, active learning (Freeman et al., 2014) activities, and lectures. Participants use their personal computers to follow along with the online workshop materials and run the ml4bio software. The workshop introduces supervised ML workflows, evaluation metrics, a few common classifiers, and how ML experiments are presented in biological literature. It does not teach participants how to perform machine learning on their own. Figure 4.1

shows the various workshop activities and how they relate to the 4 learning goals. Below we explore 3 key features of the workshop design (software, active learning, and drawing on prior knowledge) that support participants in achieving the learning goals.

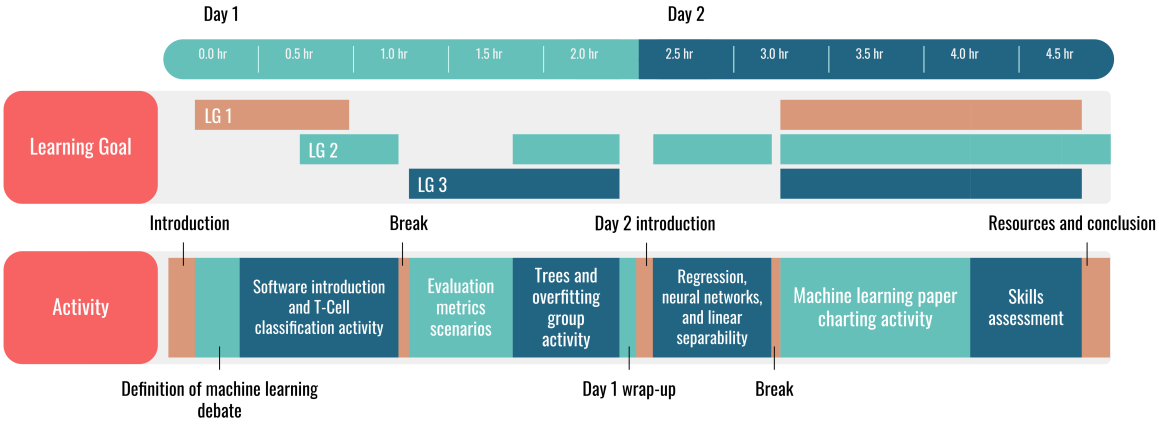

Figure 4.1: Timeline of the ML4Bio workshop. Activities are shown in addition to which non-affective learning goals (LGs) are addressed by that activity, as defined in section 4.2.

Software Design

The first—-and perhaps most fundamental—learning goal of the workshop involves understanding a ML workflow. In fact, without understanding that workflow, participants cannot successfully achieve the other objectives. As a result, we wanted to create structural supports in the workshop for this learning goal. Specifically, we wanted to scaffold participants' learning about the workflow in a way that did not rely on field-specific terminology or existing computational skills. To do so, we created the ml4bio software so biological researchers could visually explore the ML workflow (Figure 4.2).

The ml4bio software is written in Python and based on the popular ML library scikit-learn (Pedregosa et al., 2011). It uses PyQt5 (v5.15.4) for the graphical user interface. Participants are asked to download and install the Anaconda Python distribution and the ml4bio software before the workshop using step-by-step instructions provided on the workshop's website. We use Anaconda to create a conda Python environment for the ml4bio software via a script that installs and runs the software. In doing so, our software instantiates our

Figure 4.2: The layout of the ml4bio software, with colored panels showing its main sections. Users navigate a machine learning workflow in panel A, view summarized results in panel B, and view detailed information and data visualizations in panel C.

design principle of minimizing the need for extensive coding background. The ml4bio package is also available from GitHub (`https://github.com/gitter-lab/ml4bio`) or PyPI (`https://pypi.org/project/ml4bio/`).

Once installed, participants and instructors use the software throughout the workshop to walk through ML workflows, compare models and hyperparameters, and visualize decision boundaries and model performance. The software's user experience is optimized for education instead of other similar software that is designed to perform research-quality data analyses. Workshop participants are warned that the software is not meant to be used in research and should only be used as an educational tool. We purposefully limit certain user actions to encourage correct experimental setup and only show a subset of models and hyperparameters to avoid overwhelming users. These restrictions are consistent with our design focus on preparedness (rather than expertise) and low time investment.

The software's interface is laid out into the right and left halves of the screen (Figure 4.2). The left half lets the user navigate through the steps of a ML workflow: data selection, training,

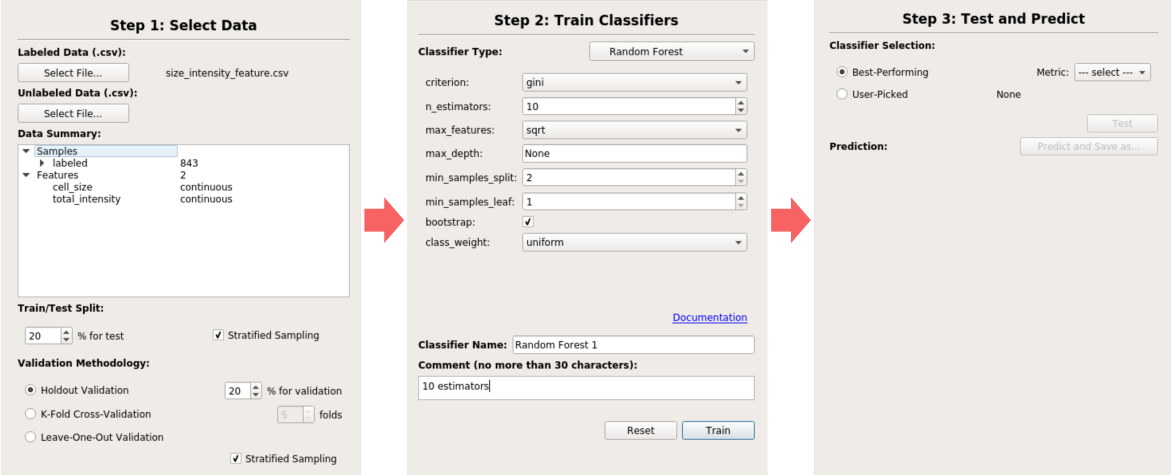

Figure 4.3: Different configurations of the left half of the software interface throughout a machine learning workflow.

and testing/predicting, thus directly supporting learning goal #1 (Figure 4.3). Laying out each of these steps is a key part of the software's design. At each step, the user is presented with reasonable choices for how to proceed to the next step of the workflow. The software allows users to move forward to the next step, but users generally cannot go back a step without fully resetting and choosing a new dataset. This prevents users from accidentally causing data leakage by performing additional model selection after viewing test set performance or choosing a different test set that might include data from a previous training set. Thus, the user can only perform a complete and standard ML workflow using the software, reinforcing the purpose and flow of each step.

When selecting data, users can view a summary of the data instances and features in a data summary window. The data is assumed to already be pre-processed. This mirrors our decision to keep detailed data pre-processing methods outside of the scope of the ML4Bio workshop, as pre-processing methods are often domain specific. Users can select a data splitting strategy for both a final test set and a validation set for model selection and whether to use stratified sampling.

In the training step, users can train and compare different classifier and hyperparameter performance on their training set and validation set. The software includes popular classifiers such as decision trees, random forests, support vector machines, neural networks, k-nearest

neighbors, and logistic regression. A subset of hyperparameters available in scikit-learn can be changed for each model, and each configuration can be given a name and comment.

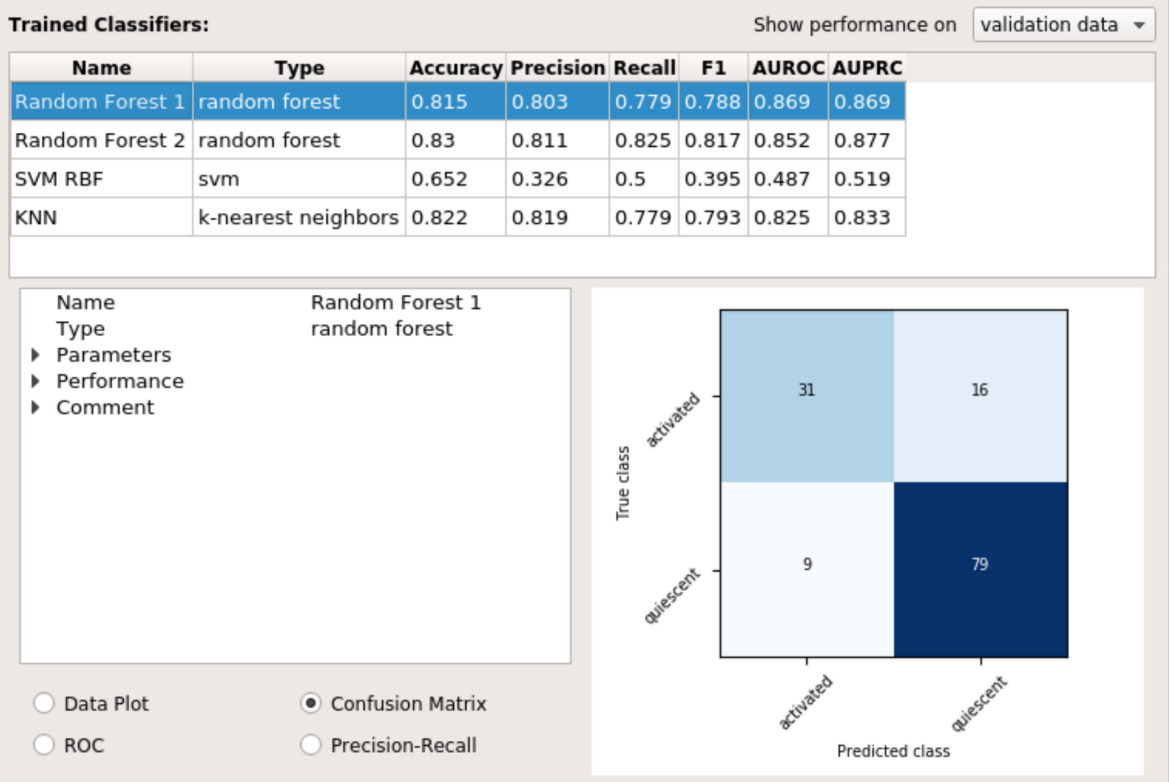

Figure 4.4: The right half of the ml4bio software interface. The top shows a summary of all classifiers created during model selection, and the bottom shows detailed information on the performance of the selected classifier. Note that multiple classifiers can only be viewed during model selection. The user must select a single model and can no longer see the performance of other models once the test data is examined.

As each model is trained, it is added to a table summarizing all trained models' performances (top half of Figure 4.4), where a number of classification performance metrics can be viewed for each model on either the validation set or the training set. Each model can be selected, where it is then shown in more detail (bottom half of Figure 4.4). Here, users can choose to view evaluation curves, a confusion matrix, or a plot of the data with the model's decision boundary for 2D datasets. Throughout the workshop we especially focus on the decision boundary visualization to show the differences between classifiers, the limits of different classifiers, how certain hyperparameters can affect how a classifier learns. This focus allows us to move away from the specific features of each classifier, which we removed based

on pilot data and refining our learning goals, and instead to focus on how different facets of classifiers and problems affect performance (Learning goal #2).

Finally, users can move to testing and predicting. In this step, users can select one of the currently trained models for final classification and evaluation. Users can select a model manually, or select the model which performed best on a certain metric. The software presents a warning that after the test set performance is shown, no more model selection can be performed. After acknowledging this, the right half of the software interface will show only the selected model, but training and validation set performance can still be viewed.

Active Learning

Although our early pilot workshops involved mostly lecture and a summative assessment activity, our use of the backward design paradigm to define learning goals also led us to redesign the workshop activities. Specifically, we transformed the workshop to involve more active learning opportunities for participants (Freeman et al., 2014).

The workshop now uses a variety of active learning strategies such as scenarios, polls, discussions, and problem solving. It is in an HTML format derived from The Carpentries (Wilson, 2016) lesson template and hosted on GitHub pages (`https://carpentries-incubator.github.io/ml4bio-workshop/`). This allows anyone to access, propose modifications to, or reuse any workshop materials through GitHub (`https://github.com/carpentries-incubator/ml4bio-workshop`). Git tags for each workshop date track the evolution of the workshop materials.

As an example of how active learning was added to the workshop, the introductory lesson was changed from a lecture to a debate activity. After sharing a textbook definition of ML (Mitchell, 1997), 3 scenarios are presented to participants. One such scenario is a person hand-writing a decision tree from personal knowledge. For each scenario, participants are asked to individually rate how much they do or do not think the scenario is an instance of ML and then justify their position in discussion. We end each scenario by showing how the instructors view each scenario.

We added additional polls and scenarios throughout the workshops. Participants use scenarios and software activities to learn how class balance affects different evaluation metrics. Scenario-based polls are also used as formative assessments at the end of some lessons. These include model selection scenarios, where participants are asked to choose an appropriate classifier based on factors such as the need for interpretability, dataset size, and expected linear separability. In addition, polls at the end of software activities help ensure that participants completed the activity successfully.

Our redesign of the workshop to support participant preparedness (Learning goal #3) led to the largest shift in the workshop towards active learning. Specifically, we added a ML paper charting activity, which is introduced in day 1 and occurs on day 2. At the end of the first day, we ask participants to choose a biological paper that uses ML from a list (Evans and Cushman, 2009; Chhatwal et al., 2009; Yip et al., 2012; Angermueller et al., 2017; Tian et al., 2018; Stokes et al., 2020; Albrecht et al., 2021; Magar et al., 2021) or a paper they brought, but we make sure that each paper is selected by 2 or more participants. We ask participants to skim this paper before the second day of the workshop.

On the second day of the workshop, after working through an example paper (Listgarten et al., 2004) as a group, participants attempt to fill out a chart cataloging the steps of the ML workflow (Learning goal #1), evaluating model performance (Learning goal #2), and critiquing the experimental design (Learning goal #3). Participants first spend some time individually, then with the others who chose the same paper. Finally, we rejoin as a group and discuss issues or interesting conversations that came up.

This activity gives participants guided experience in the interpretation of research involving ML. The transition from learning about ML more generally to having to interpret real ML experiments begins during the workshop. In doing so, we scaffold the participants in moving from lower-levels of Bloom's taxonomy (Crowe et al., 2008) (Identify) to higher-levels (Evaluate) , which have been noted to be difficult to teach in machine learning (Sulmont et al., 2019b) in the related the Biggs and Collis' Structure of the Observed Learning Outcome taxonomy (Biggs and Collis, 1982). Additionally, the choice of paper allows participants to

select a paper that interests them. Most of the papers do not cleanly fit into the standard ML workflow taught on the first day, as is expected given the huge variety of ways ML is used in biology. This "messiness" gives participants support in activities that look more like what they will experience in their professional lives.

We found that the online workshop format naturally supported these active learning activities. Zoom polls collected feedback on the scenarios. Screen annotations allowed participants to rate the introductory ML scenarios by drawing on a figure. Breakout rooms facilitated small group discussions of scenarios and the paper charting activity. Participants used the chat to ask questions during lessons that could be answered verbally by the instructor leading the lesson or via chat by a different instructor. Despite these advantages of the online workshops, we observed more extensive and dynamic discussions and research-related questions in our in-person pilot workshops.

Drawing on Prior Knowledge

Learners come to learning environments with prior knowledge which can both help or hinder their learning (Ambrose et al., 2010). The ML4Bio workshop is no exception: the workshop is intended for those who are involved in biological research, typically graduate students, postdocs, and staff scientists. These participants come to the workshop as trained researchers in a biological domain. Thus, when designing the workshop, we considered an andragogical approach; where *andra*gogy is an approach that specifically focuses on adult learners (Chan, 2010). Adult learners tend to be motivated by potential applications and learn through drawing on their own prior experiences. We designed workshop lessons to be task-oriented and use real biological applications of ML.

In the second lesson of the workshop, where participants are introduced to the ml4bio software and walk through the ML workflow, we use a motivating example of classifying T-cells as active or quiescent using imaging data (Wang et al., 2020). Throughout the workshop, we refer back to this dataset as well as synthetic datasets with the same features and classes that are designed to specifically show some facet of classifier behavior. Other real datasets are

included in the ML4Bio GitHub repository from the UCI Machine Learning Repository (Dua and Graff, 2017) and biological studies (Schwartz et al., 2015; Singh et al., 2002; Dagliyan et al., 2011; Lee et al., 2018). Using motivating biological problems leverages participants' prior knowledge to help them understand how classification works. Participants can more easily see what is reasonable or unreasonable in a familiar problem domain. Tailoring ML education to learners' primary domains has also been effective in undergraduate education (Sulmont et al., 2019a).

In the lesson where decision trees are introduced, earlier versions of the workshop spent time introducing the tree data structure. After feedback from a participant, in subsequent workshops we instead connected the tree data structure to phylogenetic trees which most participants were already familiar with. When teaching performance metrics we use biological scenarios where participants already have an intuition of what types of errors are more important, and so can more easily demonstrate why accuracy tends to be a poor metric when there is a large class imbalance.

We use participants' prior knowledge by centering the academic paper critique activity in the workshop. This activity draws on participants' existing abilities to read and analyze academic papers and merely has them extend those abilities to papers that include ML components. Introducing the skill of reading academic papers from the ground up would take much more than an hour or two to attain (Hubbard and Dunbar, 2017). We structured the ML paper charting activity to use this prior knowledge, as participants are encouraged to choose a paper to chart that they are interested in or comes from their research area.

While participants' prior knowledge generally enhances their learning, we also considered areas where prior knowledge could hinder it. Misconceptions can occur if participants incorrectly apply their prior knowledge and we do not catch and confront the misconception. We were especially cautious when designing the lesson on evaluation metrics. Many of the metrics used to evaluate classifier performance, such as precision and recall, have different meanings in laboratory settings. We directly address this and other possible overlapping terminology to participants.

4.3 Study Design

Participants: After we reformulated the workshop's learning goals and activities, we held 3 ML4Bio workshops online over Zoom on May 4^{th} and 6^{th}, August 2^{nd} and 5^{th}, and September 14^{th} and 16^{th} 2021. Participants were recruited to the workshops via email and could choose to participate in the study after registering for the workshop during the pre-workshop survey. 10, 18, and 19 workshop participants consented to participate in the study in the May, August, and September workshops, respectively. The study was approved by the Institutional Review Board of the University of Wisconsin-Madison (#2021-036), and we obtained electronic informed consent from all study participants. Participants could participate in the workshop and provide informal feedback on the workshop without participating in the study. Here we report only on those who consented to study participation. A breakdown of participant demographics is presented in the first part of the analysis.

Data Collection: To collect data on the workshop experience and its efficacy in achieving its learning goals, we designed 3 different data collection instruments: a pre-survey, a skills assessment, and a post-survey. The pre-survey was emailed to participants in the week before the workshop; the assessment was given during the workshop; and the post-survey was emailed to participants immediately after the workshop. No directly identifying participant information was collected.

The pre- and post-surveys were designed to collect participant demographic data, record their workshop experiences, and evaluate the workshop's affective learning goal (#4). In the pre-survey, we collected participant demographic data including current career stage; experience with statistics, ML, coding, and the command line; and overall goals and expectations for the workshop. In the post-survey, participants were asked about their experience in the workshop with regards to their expectations, pacing, time allocation, and general feedback.

The post-survey was also used to assess the workshop's affective goals. Specifically, we used a retrospective pre-post design for self-assessment of affective learning outcomes. Retrospective self-assessment has been shown to help prevent response-shift bias (Bhanji et al., 2012), where understanding of the question being asked can change between pre-

and post-assessments, while still identifying learning (Bhanji et al., 2012). For instance, a participant's increased understanding of ML could lead them to realize that they initially understood less than they thought they did, thus resulting in a decrease in self-assessed knowledge after learning. One paired pre-post question was included for verifying the retrospective questions.

In contrast to the surveys, the in-workshop assessment was designed to ascertain whether or not participants had achieved the content learning goals of the workshop (#1-3) and to verify participants' self-assessment of their knowledge and confidence in ML after the workshop. Participants were given a heavily modified excerpt from a paper that uses random forests to predict microRNA targets (Mendoza et al., 2013). The modifications included feature simplification and the changing model selection to be based on the test set, introducing data leakage into the workflow. Participants were asked to identify parts of the ML experiment such as the model, features, and data splitting strategy and to assess the experiment for overfitting, choice of performance metrics, and data leakage. This assessment allows direct measurement of participant's ability to understand and assess ML as presented in academic papers. Identifying parts of the ML experiment assesses achievement of learning goal #1, and evaluating the experimental design and model performance assesses achievement of learning goals #2 and #3. Note that without a pre-assessment of learning goals, this assessment strategy does not provide causal evidence that the workshop caused learning goal achievement. We decided that requiring participants to complete a pre-assessment would significantly lower interest in the workshop. Additionally, when paired with the retrospective self-assessment, we can draw conclusions about self-assessment of learning goal achievement and use the in-workshop assessment to verify the level of knowledge post-workshop.

Data Analysis: The first step in data analysis involved creating matched data sets for each consenting participant. 4 digit codes for each participant linked their pre-survey, in-workshop assessment, and post-survey.

The second step involved analyzing the self-reported survey data. Pre- and post-survey questions related to background, expectations, and experiences (rather than questions re-

lated to comfort and confidence) were analyzed using basic counts and descriptive statistics. However, both retrospective and paired pre-post questions were compared with two-sided Wilcoxon signed-rank tests. Tests were performed using the scipy.stats.wilcoxon method in SciPy (Virtanen et al., 2020) v1.7.1 with default parameters.

The in-workshop assessment was "graded" for correctness. Workshop designers (who are also ML researchers) determined correct answers for each question on the assessment. Author C.S.M. coded all answers given by participants. Additionally, authors C.S.M. and R.S.R. looked at the participants' explanations for their responses. From those explanations, we identified common themes in correct (and incorrect) answers. While many questions in the assessment have straightforward answers, later questions are less clear. The final 2 questions in particular, "How well did the model perform?" and "Do you trust the validity of these results?", do not have an obvious correct answer. We instead compare participants' responses to possible factors they were asked to identify the presence or absence of in other questions: data leakage, improper performance metrics, and overfitting. How the presence or absence of these factors, and the degree to which they occur, affects participants' trust in the presented results provides insight into how the participant will engage with ML research.

4.4 Study Results

Attendees' Backgrounds and Expectations

Over the 3 workshop sessions there were 47 participants in total who completed the pre-survey. These participants consisted of 2 undergraduate students, 21 graduate students, 16 post-doctoral researchers, 6 staff scientists, 1 principal investigator, and 1 technician (Table 4.1). 15 participants only completed the pre-survey, 6 completed only the pre-survey and assessment, and 26 completed all instruments. The 21 incomplete responses include participants who did not return for the second day and participants who completed the workshop but did not fill out the post-survey.

Of the 47 participants, 46 had never taken a ML course, and 6 had never taken a calculus

Date	Undergraduate	Graduate	Post-doctoral	Staff Scientist	PI	Other	Total
May	0	2	5	2	0	1	10
August	2	10	5	0	1	0	18
September	0	9	6	4	0	0	19
Total	2	21	16	6	1	1	47

Table 4.1: Participants of three workshops. The single "Other" response was noted as "Research Specialist/Technician". The table includes participants who completed the pre-survey but did not complete the assessment or post-survey.

or statistics course. Before the workshop, 13 self-reported as knowing nothing about ML, 27 as knowing a little, and 7 as knowing a moderate amount. Only 3 participants reported having more than a little research experience with ML. Half of participants had at least a moderate amount of coding experience and experience with the command line interface. These data align with our experiences in the pilot workshops and provide strong support for our design choice to minimize the need for coding and mathematical background knowledge.

Participants' expectations generally aligned with the workshop's learning goals. 33 participants were interested in generally learning about ML with responses such as *"basic overview of machine learning"* and *"understanding how machine learning works."* 24 participants specifically mentioned wanting to learn questions they could answer in their own research using ML or how to apply ML to their research. These expectations align with the current learning goals of the workshop and are consistent with our focus on "preparedness" rather than ML expertise.

In-Workshop Assessment

Assessment results are summarized in Table 4.2. In-workshop assessment results show that learning goals #1-#3 were generally achieved, though identifying subtle instances of data leakage proved challenging for many participants. Most participants submitted complete evaluations; of the 30 responses collected only 4 were incomplete. When presented with an altered excerpt from an academic paper, almost all responses correctly identified the target variable, number of instances, model, data splitting strategy, and performance metrics.

Data Leakage: Data leakage was added to the excerpt by choosing the features used in

Target ML concept represented in different assessment questions	Coded Response Types	No. Responses
Experimental Design	Correctly identified class label	31
	Correctly identified number of instances	28
	Correctly identified model	31
	Correctly identified evaluation metrics	30
	Correctly identified data split	31
	Incorrectly identified class label	1
	Incorrectly identified number of instances	4
	Incorrectly identified model	1
	Incorrectly identified evaluation metrics	2
	Incorrectly identified data split	1
Data Leakage	Correct Identification	14
	Incorrect Identification	11
	Unsure	4
Overfitting	Provided justification by noticing gap in training and test sets	10
	Other correct justification	5
	Incorrect justification	2
	Provided no justification	11
	Unsure	3
Performance Metrics	Correctly identified metrics as appropriate (no justification)	24
	Correctly identified metrics as appropriate and mentioned additional appropriate metric	3
	Correctly identified metrics as appropriate and mentioned most important confusion matrix quadrants	2
Trust	Results are valid	14
	Results are invalid (no explanation)	7
	Results are invalid because of data splitting	5
	Results are invalid because of performance gap in training and test sets	2
	Unsure	1

Table 4.2: In-workshop assessment results. The numbers presented in this table represent the number of open-ended responses to assessment questions that were coded along each of the dimensions listed. Total number of responses per concept differ because not all respondents answers all of the questions.

a final model based on test set performance. 14 out of 29 responses correctly identified the presence of data leakage in the experiment, while 11 responses claimed that there was no data leakage, and 4 expressed that they were unsure. 2 of the responses that claimed there

was no data leakage appear to have confused the concepts of data leakage and class balance, as they provide the balance of negative and positive examples in the dataset as evidence for a lack of data leakage. Almost all of the responses that provided an explanation for the presence of data leakage correctly cited the lack of validation set or the choice of the final model based on test set performance as evidence of data leakage.

Overfitting and Performance Metrics: When asked about evidence of overfitting, 10 responses explicitly considered the 0.13 gap in accuracy between the training and test sets in their response. Some participants considered this gap evidence of overfitting, while others said that the gap was not large enough to consider overfitting to be a problem. We consider both of these subjective interpretations as acceptable. Other factors mentioned in responses were that it is harder find evidence of overfitting without a validation set, random forests are not prone to overfitting, and the low ratio of the number of features to the size of the dataset. 11 responses did not justify their answer, and 3 responses were uncertain about the presence of overfitting. Participants' critique of metric choice also showed an understanding of the correct factors to consider. For instance, some responses mentioned how, given the likelihood of further experimental validation, false negatives were more important than false positives, so sensitivity was an important metric.

Trust: Finally, when asked whether or not they trusted the validity of the results, participants had split opinions. 14 responses felt the results were trustworthy and did not provide any explanation. Of the 14 who said no, 5 brought up the data splitting strategy, 2 mentioned the performance gap between the training and testing sets, and 7 provided no explanation. We are not sure how to interpret participants' lack of trust in the paper, and it suggests opportunities for further learning may be necessary to differentiate the severity of different problems with ML workflows.

Affective Outcomes

In addition to achieving the content-based learning goals, the data indicates that the workshop was also successful in helping participants achieve the affective learning goal (#4).

Participants' comfort in training classifiers for a research project generally increased after the workshop ($p = 2.2 \times 10^{-4}$, $n = 26$, Wilcoxon signed-rank test) as shown in Figure 4.5. Before the workshop, over half of participants reported being not at all or a little comfortable, whereas after, among those who responded the majority were either a little or somewhat comfortable. While some participants reported as being very comfortable training classifiers after the workshop, this response might show an overestimation of ML skills. We do not expect participants to be able to use ML in their own research without assistance after the workshop. One participant who noted that they were very comfortable before the workshop lowered their reported comfort level after the workshop to somewhat comfortable. This is likely due to response-shift bias (Bhanji et al., 2012), where after learning more about ML this participant felt that they had overestimated their abilities before the workshop.

Figure 4.5: Sankey diagram of participants' responses pertaining to comfort with machine learning before and after the workshop across all 3 sessions. Note that a significant proportion of those who completed a pre-survey and not a post-survey did not attend the workshop at all. 47 completed the pre-survey, 30 completed the post-survey, and 26 completed both.

There was an almost universal increase in self-reported knowledge and affective learning outcomes from before to after the workshop (Figure 4.6). We consider this self-reported increase in knowledge evidence of increased confidence and self-efficacy. Participants reported a marked increase in their confidence in identifying a problem that is well-suited to ML in

their research ($p = 5.3 \times 10^{-6}$, $n = 26$, Wilcoxon signed-rank test) and an increase in their likelihood to pursue ML for that problem ($p = 9.4 \times 10^{-6}$, $n = 26$, Wilcoxon signed-rank test). A majority of participants reported that they had little or no interest in pursuing and confidence in identifying ML before the workshop, while a majority reported that they were at least very interested and very confident after the workshop. Participants reported a significant but lesser increase in general knowledge of ML ($p = 6.4 \times 10^{-6}$, $n = 26$, Wilcoxon signed-rank test) and confidence in reviewing a paper that uses ML ($p = 1.3 \times 10^{-5}$, $n = 26$, Wilcoxon signed-rank test), with the majority of participants reporting that they were only somewhat confident or knowledgeable after the workshop. Participants were not overconfident and did not claim that they were fully ready to review papers involving ML after the workshop.

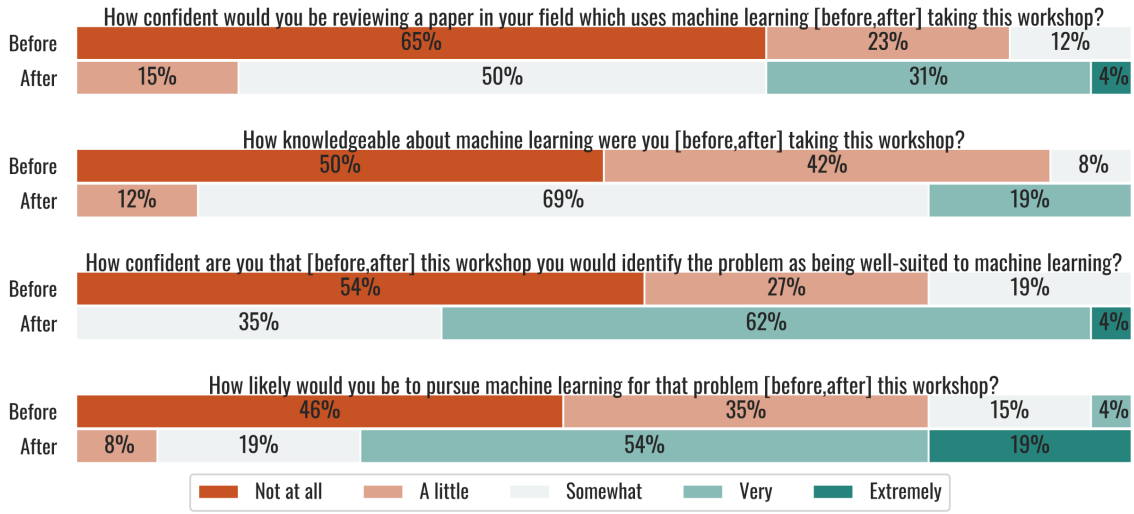

Figure 4.6: Participant responses to self-reported knowledge, confidence, and interest in machine learning before and after the workshop. Note that these questions used a retrospective design, meaning that participants were asked about both before and after the workshop in the post-survey.

Workshop Experiences and Expectations

Not only did the workshop meet our learning goals, participants also reported that it met their expectations. Satisfaction with the workshop was high. On a 5-point Likert scale, 10 participants responded that the workshop exceeded expectations, 11 that the workshop met

all expectations, 6 that the workshop met most expectations, and none that the workshop met some or did not meet expectations. Of those who responded that the workshop met most expectations, there was no clear consensus on which expectations were not met. More real-world examples, coding, clustering, and how to use ML were mentioned as unmet expectations.

When asked which workshop topics were most valuable, almost every part of the workshop was named by at least some participants. The two most commonly named parts of the workshop were lessons that involved classifiers and the paper charting activity, with 12 and 7 participants naming those parts, respectively. Participants generally valued learning the variety of classifiers available. Typical responses were focused on *"different classifiers and how they compare"* or *"discussing the different classifiers."* Three participants specifically mentioned that being able to visualize data or how classifiers work was particularly helpful.

A number of the responses mentioning the paper charting activity noted that it was particularly valuable because of its applicability and realism:

> *"Working through the topics with the ML4bio sample where we could see all the different graphs and assessment statistics to understand how they relate. Also discussing papers where not everything is laid out the same way or fully documented and learning to recognize that." "I thought the research paper exercise was helpful. Specifically because we could bring in articles that interest us."*

Two participants said that the in-workshop assessment was the most valuable part of the workshop for them. Their responses mention that performing an assessment, then checking it as a group, gave immediate feedback on their learning. Other participants noted that scenarios and polls used throughout the workshop also helped them check their own learning:

> *"I found all of it really valuable. I especially liked the knowledge check at the end through literature evaluations. This course solidified a lot of the ideas behind ML for me."*

Participants were asked which workshop topics they would find difficult to explain at a high level. Details of classifiers were reported as the most common topic participants would

have the most trouble explaining. 13 responses mention classifiers in some form in their response. Most responses name specific classifiers and focus on more detail: *"The details of how each algorithm works"*, *"Logistic regression vs neural networks"*, and *"Explaining how a neural network works."* Other common areas of confusion were model selection and data leakage, with 5 and 3 responses mentioning them, respectively. Responses mentioning model selection often included choosing a classifier for a certain task and optimizing hyperparameters. The number of responses mentioning data leakage may be influenced by the in-workshop assessment, as participants were aware of which parts of the in-workshop assessment gave them trouble while taking the post-survey. Despite these continued areas of confusion, participants highlight that the workshop met their goals and provided them with valuable skills and knowledge around ML.

4.5 Lessons Learned

The ML4Bio workshop effectively achieves its goals. Participants left the workshop having attained basic knowledge of ML, increased interest in ML, and preparedness for continued learning. While participants struggled with some of the more subtle aspects of designing ML experiments, namely detecting data leakage, fully preparing participants to perform ML research on their own is not the goal of the workshop. However, improving the workshop such that participants are better able to evaluate ML experiments, and especially detect data leakage, is an important future goal.

Provide Flexibility for Participants to Bring Their Own Goals

When asked in the pre-survey what they were hoping to learn from the workshop, most participants named a specific research question they were interested in exploring with ML. This aligns with previous literature on adult learners; adult learners are typically more motivated than non-adult learners by real problems (Chan, 2010). Thus, we found it especially important to ground workshop activities in real or at least realistic data wherever possible.

While having a problem in mind for the workshop can motivate participants, it also complicates meeting participant expectations. The research questions participants bring likely require tools and knowledge beyond a general introduction to ML. Therefore, in workshops geared towards active researchers it is more likely that the workshop's learning goals may not perfectly align with a participant's needs.

We mitigate this possible misalignment in a number of ways. We provide a variety of applications throughout the workshop, so that participants are likely to see at least one problem that is similar to their research area. However, we do not cover application-specific pre-processing and feature generation. This is evidenced by multiple participants mentioning that they would want more information on image analysis techniques with ML, even though the first presented dataset is an image classification problem.

Additionally, we provide resources for participants to continue learning about ML. These resources allow participants, even if their goals were not fully met during the workshop, to have an accessible next step for learning about their specific application. We present these resources to participants in the final workshop lesson. The resources include online textbooks, a Jupyter notebook demonstrating a ML workflow, ML-focused code tutorials, and Carpentries workshops for participants interested in learning more technical skills. We plan to continue to grow this list of resources as we are presented with new participant interests.

Finally, during the workshop we give participants space to explore what they find to be interesting. This exploration is clearest during the literature charting activity where participants can choose from a variety of papers to investigate or bring their own. Multiple participants found this activity to be the most valuable part of the workshop. Participants can explore a specific application of ML with support from fellow participants and workshop instructors.

Participants bring their own goals to the workshop. Accommodating these goals can be seen as counter to the backwards-design paradigm, where learning goals are chosen in advance. Incorporating application breadth, resources for participants to continue to learn on their own, and flexibility in the workshop structure are effective tools for creating room

for these goals while still conducting a learning goal-driven workshop.

Assessments are Worth the Time

Participants found both formative and summative assessments valuable throughout the workshop. Despite assessment being a well-known tool in school-based learning environments (Black and Wiliam, 2009), we were initially hesitant to include assessments because we felt they might lower interest from participants. However, multiple participants named the in-workshop assessment the most valuable part of the workshop, though this assessment was originally designed for the study and not directly as a learning tool. Reviewing the assessment afterwards allowed participants to catch misconceptions they otherwise would have taken away from the workshop.

Participants mentioned that other in-workshop assessments, polls, and scenarios throughout each lesson were valuable checks of their knowledge. These quick assessments also allowed the workshop instructors to notice and spend extra time on areas participants were especially confused about.

An additional possible positive effect of the assessments was providing a mastery experience: a challenge that is successfully completed, demonstrating improvement. Mastery experiences lead to increased efficacy and confidence (Bandura, 1977). In the final assessment, almost all participants were able to correctly identify parts of the presented ML experiment. This may have helped demonstrate to participants their new knowledge of ML and helped lead to the marked increases in self-rated confidence and knowledge of ML participants expressed.

Set Expectations as Clearly as Possible

Every iteration of the workshop included more information about what participants should expect. However, while the number of participants who noted they had misaligned expectations decreased over subsequent iterations of the workshop, every workshop still had some participants who expressed that there was something they were hoping or expecting to learn

from the workshop that was not covered.

The nature of the workshop's audience may have exacerbated this issue. Participants may have come to the workshop having already heard of methods that are popular in their field and expected to learn about them during the workshop. Beyond more general expectations, it may have helped set expectations if we had noted specific, popular concepts that would not be covered in the workshop. For example, we originally noted that the workshop would only present supervised learning and not unsupervised learning. However, some participants had heard of clustering and may not have realized that clustering is a part of unsupervised learning, thus still expecting to learn it. Explicitly stating popular terms and buzzwords can help communicate with participants and manage expectations.

Participants were also uncertain about how deeply they should understand some of the workshop content. This confusion was especially apparent during the lesson on logistic regression and neural networks. Multiple participants expressed that they felt they did not understand all the details of how logistic regression and neural networks work in their post-survey. However, we did not expect participants to understand the mathematics behind these models. Better delimiting what we expected participants to learn about these models and what was out of scope may have reduced this confusion.

4.6 Future Directions

We plan to continue to refine and expand the ML4Bio workshop. One area of improvement is to further pare down the knowledge participants need to achieve the workshop's learning goals. While we have already adapted the scope of some lessons, for instance, by removing the mathematical details of how logistic regression works, it would be productive to approach this refinement more systematically. This streamlining could free time to explore more nuanced examples of experimental design flaws like data leakage.

While there was little evidence of participants overestimating their skills in ML, further clarifying the limitations of the workshop and ML in general would aid participants in choosing their next steps. This could include adding additional cautionary language to the

concluding lesson to make it clear that participants should not expect to be able to run research quality ML analyses independently after this workshop. Emphasizing the importance of data cleaning and pre-processing, which often require domain-specific strategies, could help participants leave with a correct understanding of their current skills in ML.

While we are able to determine the achievement of learning goals through the current set of surveys and assessment, some of the questions on the assessment do not encourage useful feedback. Questions on whether or not the results are valid or if the metrics are appropriate do not have a definitive answer, but a large proportion of participants answered them with a one-word "yes" or "no". Ideally these questions would encourage participants to rephrase their answer or to change the question so that it has a clear correct answer. In the current form of these questions, a large proportion of participants' answers were not useful for analysis, though they did still appear to provide a valuable learning experience for participants.

One possible future direction for the workshop would be to increase its ability to accommodate different participants' goals. This increased flexibility could be provided by having branching activities or multiple versions of the workshop. These multiple versions could be the same activities with all datasets focused on a single area of application, or even have different activities tailored towards, for instance, imaging applications. We could also keep a single version of the workshop but provide branching activities. For instance, for the last half-hour of the workshop participants could choose whether to be introduced to unsupervised learning, deep neural networks, or creating ML workflows in Python. This strategy, however, would require multiple instructors.

A tool to export workflows in the ml4bio software to Jupyter notebooks would provide a powerful link to technical skills. Participants interested in coding could see how workflows they perform in the ml4bio software are expressed as code, giving a smoother transition to coding ML workflows. Currently, we provide a Jupyter notebook with Python code demonstrating an example ML workflow similar to those implemented in the ml4bio software. Participants can run this notebook in a web browser with Binder(Project Jupyter et al., 2018).

A few participants in each workshop consistently struggled with software installation both

before and during the workshop. Some issues arose from installing and configuring Anaconda. Others were due to the scripts we provided to create the required conda environment or the environment itself. Including more details about expected behavior and installation screenshots in our setup instructions partially alleviated but did not eliminate these issues. A cross-platform installer that provides the ml4bio software and required datasets would make the workshop more accessible and reduce the amount of command line troubleshooting required. We are exploring the conda constructor tool to create this type of installer. In the long term, rewriting the ml4bio software to run in a web browser without any local installation would minimize the technical requirements and help scale the workshop to larger audiences. However, this would require substantial software development. There are also numerous ways to improve our existing ml4bio software such as support for more classifiers, visualization of datasets with more than 2 features, saving models and settings, more interface tooltips, and better text scaling.

We plan to expand our current instructor notes to the point that we could provide the ML4Bio workshop as a full lesson plan others could teach. While all activities in the workshop are laid out in the online materials and current instructor notes, they are not detailed enough for someone to teach the workshop without first observing it. The workshop could also be scaled up to larger sessions if additional helpers were present to lead group activities and troubleshoot software issues. The ML4Bio workshop has joined The Carpentries Incubator to gain additional support and feedback and to expand the audience and possible future instructors. Our workshop design and instruction have already benefited from the principles taught in The Carpentries Instructor Training and specific suggestions from Carpentries instructors. We hope to continue to expand and improve the ML4Bio workshop so that it continues to be an effective tool for helping biologists participate in an increasingly computational research world.

Chapter 5

Conclusions and Future Work

The complexity behind the process of modern scientific discovery inevitably requires increased specialization, and with it increased levels of collaboration between scientists. This can be seen in the biological sciences as the volume of data and number of researchers per published article has increased over time (Cordero et al., 2016). This is not inherently a negative trend; however, this increase can create barriers to biological discovery as more specialists and data need to be gathered. The careful development of computational tools, publicly available data, and predictive algorithms can partially alleviate this burden. Data that can be accurately inferred does not have to be measured, and well-designed tools can reduce the specialization required for a particular analysis or pipeline.

In this thesis, we have approached improving the state of high-throughput biological analysis from 3 angles. We have examined automating parameter selection to reduce the computational and methodological burden required to perform pathway reconstruction analyses, and we have investigated the inference of localization data withing the context of biological pathways to provide context-specific localization information that is currently typically gathered through investigation experiments. We have also approached this improvement from the direction of education, creating and evaluating a practical introduction to machine learning for active biological researchers.

5.1 Contributions

The contributions in each chapter are summarized as follows:

- Chapter 2, Improving the usability of biological network analyses:
  - We showed that current methods for selecting parameters in pathway reconstruction have trouble consistently selecting good parameters.
  - We implemented a graphlet-based measure of topological similarity to a set of reference pathways which separates plausible and implausible reconstructed pathways.
  - We created pathway parameter advising, a tool to perform automatic parameter selection in pathway reconstruction.
  - We defined a framework for evaluating the effectiveness of parameter selection based on a set of literature-derived topological constrains.
  - We demonstrated that pathway parameter advising can generally select realistic pathway topologies, and be used to guide effective pathway reconstruction in biological analyses.
- Chapter 3, Predicting localization within pathway context:
  - We showed that there exists a large amount of disagreement between protein localizations in protein-level databases and localizations in pathway databases. The cause of this disagreement is unclear but it is pervasive across multiple pathway databases and protein-level localization databases.
  - We showed that different pathway databases also have significant disagreement in localization, including the general distribution of localizations across pathways.
  - We demonstrated that predicting pathway-level localization is a difficult task. Models that incorporate pathway topology can improve performance over those that do not.

  - We demonstrated that the topology of reconstructed pathways from mass spectrometry data contains information about how protein localizations changed over time.

- Chapter 4, Practical and approachable computational education for biologists:
  - We created the ML4Bio workshop, an open resource for effectively introducing machine learning to biologists, focused on practical research skills and increasing confidence and interest in machine learning.
  - We demonstrated how guiding principles and iterative design were used to improve and design the workshop over multiple pilots.
  - We demonstrated how prior knowledge, especially high-level research skills, can be utilized in workshops to jump-start learning about literature comprehension in a new area.
  - We found that workshop participants typically attend with a specific goal or application in mind. Providing flexibility for participants to fulfill their own goal is important to participants.
  - We demonstrated how careful software design can bypass technical and quantitative skills typically needed to engage with computational methods.

5.2 Conclusions

In Chapter 2 we were able to create a method, pathway parameter advising, that effectively automates parameter selection in pathway reconstruction. Pathway parameter advising uses a topological distance metric derived from the graph theory concept of graphlet decomposition, which allows a graph to be represented as a distribution of small subgraphs. Pathway parameter advising uses this metric to tune parameters such that networks created with network methods look similar to those from publicly available databases of biological networks. When used to choose optimal parameter settings for reconstructing pathways from curated

databases, pathway parameter advising generally chose better parameter settings than other parameter selection methods. We also showed that pathway parameter advising consistently chooses parameter settings that create pathways with more desirable topological properties than other parameter selection methods. Finally, when used to create an influenza host factor network pathway parameter advising led to an interpretable pathway that effectively represented the biological process being examined.

Following this, in Chapter 3 we investigated predicting subcellular localization within the context of a biological pathway. This predictive task is still an unsolved problem, as performance has high variance between different individual pathways and pathway databases in even the highest-performing models, but meaningful insights were gained into the nature of the problem and what is needed to solve it. Pathway topology was found to improve model performance, as graph neural networks outperformed other models that did not have any topological information. Additionally, the ability of models to correctly estimate pathway-wide localization smoothing was a major factor in their performance. Predictive performance varied greatly between different biological pathway databases. It is unclear what the exact cause of this is, though it may be related to Reactome labeling the localization of every interaction while PathBank provides only partial localization labels. While some predictive signal exists for predicting localization in contextual spatial proteomics data, and constructed pathway topology was able to provide some signal of translocation events during viral infection events, more work needs to be done to investigate how this data differs from data in both pathway databases and protein localization databases.

Finally, in Chapter 4 we created a workshop, ML4Bio, for introducing machine learning to biological researchers. The ML4Bio workshop focuses on classification and was designed around 3 principles: focusing on preparedness over fluency or expertise, necessitating minimal coding and mathematical background, and requiring low time investment. It incorporates active learning methods and custom open source software that allows participants to explore machine learning workflows. After 5 initial pilots to improve workshop design, we performed a study on 3 workshop sessions. Despite some confusion around identifying

subtle methodological flaws in machine learning workflows, participants generally reported that the workshop met their goals, provided them with valuable skills and knowledge, and greatly increased their interest in machine learning. ML4Bio is an educational tool for biological researchers, and its creation and evaluation provides valuable insight into tailoring educational resources for active researchers in different domains.

A fascinating and challenging part of the network-based exploratory analyses explored in Chapters 2 and 3 is the disagreement both among different pathway databases and between pathway databases and other sources of biological data. This tension can be seen in both chapters. In Chapter 2, when creating networks from influenza host factors, the ensemble network created from the top 50 ranked pathways by pathway parameter advising only contained 86 nodes from an input of 1257 host factors. While this network fulfilled the goals of pathway parameter advising, in that the resulting constructed pathway was functionally relevant and easily interpretable, it could be argued that this interpretability and topological familiarity was reached at the expense of including more biologically relevant host factors.

The case study in Chapter 3, and the results of attempting to predict localizations from pathway databases, highlight this tension as well. The two pathway databases examined, PathBank and Reactome, varied significantly in their distribution of localizations and in the predictive performance that was achieved on them. Furthermore, in the case study of human cytomegalovirus localization prediction, the graph attention network got less than 60% of interaction localizations correct. This is despite the same model using the same Compartments-based features achieving high predictive performance on PathBank localizations. However, during training it was able to achieve higher training performance on the interactions involving marker proteins used in Jean Beltran et al. (2016) than in the pathway database prediction experiment. This is unsurprising, as these marker proteins were derived from Uniprot, which integrates data from protein localization databases to provide localization labels. Even though in a general sense the protein localization database labels, pathway database labels, and spatial mass spectrometry data all represent the subcellular location of biological entities, each of these sources of data disagrees with the other two.

The work presented in this dissertation takes the approach that representations that have worked well before for biologists will continue to work well in the future, in a sense siding with hand curated data in pathway databases over data that is more directly experimentally derived as being more useful for exploratory analysis. In pathway parameter advising, this assumption manifests in using topological distance to a pathway database to score constructed pathways. While these pathways will look similar to ones biologists have created in the past, and thus will likely be useful in hypothesis generation, there is a longer-term issue of steering biological discovery away from the expected. It is possible that pathway parameter advising could steer a biologist away from a highly unusual topology because it does not match any in pathway databases. However it is results that differ most from the current consensus of the biological community, which are encoded in pathway databases, that could lead to the greatest biological discoveries.

5.3 Future Work

Making usability explicit

Both pathway parameter advising and pathway specific localization prediction seek to improve network-based exploratory analyses. In this thesis, these methods are evaluated on both publicly available curated biological pathway data, and in case studies approximating realistic experimental use cases for these methods. The utility of these methods is then considered to be their performance and the biological insights that could be drawn from using them. An alternate approach could be to explicitly measure the utility of exploratory analysis methods. Data could be gathered via interviews and use case studies with biological researchers. Measuring the usefulness of these methods directly would give a clearer picture of their utility, and likely elucidate where biologists succeed and fail with them. Creating realistic use cases for biologists to evaluate would have to be done carefully; it would be a challenging task to imitate the act of hypothesis generation and biological discovery.

Speeding up Pathway Parameter Advising

As discussed in Chapter 2, a drawback of pathway parameter advising is that it requires the pathway reconstruction algorithm to run for all candidate parameter combinations. In algorithms with multiple parameters, this can quickly grow to thousands of algorithm runs. However, it is often clear during computation that certain ranges of parameters will produce low-quality pathways. For instance, when using PCSF, a parameter combination may weight edge costs to the point where the algorithm produces an empty network. Additional exploration of the prize-cost tradeoff beyond this point is likely not needed.

Bayesian optimization (Balandat et al., 2020) is a framework that uses Gaussian processes to estimate model performance in its parameter space. In each iteration, Bayesian optimization combines its performance estimate with an additional estimate of its uncertainty. It then chooses which parameter combination to execute next based on the overall value of observing that point in the parameter space based on a explore/exploit trade-off. Bayesian optimization often reduces the number of iterations needed to find optimal or close to optimal parameter combinations by orders of magnitude compared to a comprehensive grid search. Furthermore, it makes no explicit assumptions about the model or parameter space.

Incorporating Bayesian optimization into pathway parameter advising would likely greatly speed up its computation, especially for more complex models. Additionally, Bayesian optimization was effective when used for parameter selection in prediction pathway subcellular localizations (Chapter 3). However, the parameter space based on graphlet frequency distance used by pathway parameter advising is likely extremely unsmooth for most models. It is possible that the Gaussian process used to estimate model performance would perform poorly in this application.

Streamlining Pathway Reconstruction with Software

Another barrier to using pathway reconstruction methods is that methods are often difficult to install and run (Mangul et al., 2019; List et al., 2017). They can require complex computational setups to perform parameter sweeps or investigations of multiple data sources, significant data

cleaning, and complex environment setups to run effectively. To help overcome these issues I am a part of the team creating the signaling pathway reconstruction analysis streamliner (SPRAS), a computational framework for running and evaluating pathway reconstruction. The goal of SPRAS is to make it easier to use pathway reconstruction methods, and compare multiple pathway reconstruction methods, by combining these methods and all steps of a network analysis pipeline into a single installation, setup, and data format.

SPRAS uses the python workflow manager Snakemake and the Docker platform to perform network analysis workflows. Its goal is to create an easy to use tool which can pre-process data, use pathway parameter advising to automate parameter selection, run a variety of network analysis methods, and then evaluate and package results. This allows biologists to easily run network analyses and compare multiple methods or other modeling decisions.

Further Investigation in Localization Prediction

Predicting pathway specific localizations is far from a solved problem. Predictive performance, shown in Chapter 3 varied between datasets but performed poorly in a case study. Spatial proteomics data gives a view into what localization looks like within a specific context. Databases and methods for protein localization and localization have typically aggregated this data; losing this contextual information. A deeper investigation using spatial proteomics data from a variety of different contexts could help reveal how contextual localization relates to aggregated data, and give insight into how to improve predictive performance.

The disagreement in localization between pathway databases and protein localization databases also warrants further investigation. Stratifying protein localization databases by data source could help reveal which types of biological data most and least agree with pathway databases. The biological context of pathway databases could also be further investigated. Matching a subset of pathways that better fit the biological contexts that protein localization databases' sources are built on could improve agreement between these types of databases.

Finally, alternate data sources as features could improve predictive performance. This could involve other context free sources of protein level data, or protein data that has context.

One example would be to incorporate tissue or cell type into localization prediction. It may be the case that there are factors affecting localization that the models presented in this thesis have no means to account for.

Refining the Goals of ML4Bio

The learning goals of the ML4Bio workshop were created by carefully considering how our set of guiding principals for the workshop intersected with the workshop's audience. These principals were based on preliminary pilots of the workshop and identification of gaps in the current offerings for machine learning education for active biological researchers. While participants of the workshop felt it was valuable, due to negative selection bias there are likely unmet needs in machine learning training from biological researchers who were uninterested in the workshop from the recruitment emails. A more systematic examination of the current state of machine learning in biological fields, the state of biological researchers' skills and training, and which parts of machine learning are most important for biological researchers would further refine the workshop's focus and could help identify these unmet needs.

Ideally, data would be collected from researchers with a variety of computational skill levels, biological research interests, and career stages. It would be important to consider data on both the dimensions of mathematical, technical, and methodological skills. While there has already been significant research of developing a core curriculum for bioinformatics and computational skills for biological researchers in general (Welch et al., 2014; Mulder et al., 2018; McClatchy et al., 2020), much of this work focuses on the long term goals of developing a comprehensive curriculum for future researchers. Narrowing the focus to practical skills for machine learning as was done for the ML4Bio workshop could yield more specific information on what needs to be done now to improve the current state of computational methods in biological research.

We have done preliminary work on designing and piloting an interview for principal investigators (PIs) in biological research areas specifically about machine learning education for biological researchers. Currently, the 40 minute interview seeks to collect information

on the machine learning skills PIs believe are most valuable, how PIs from computational and non-computational backgrounds view the current state of machine learning in biology, and which parts of machine learning they believe are most important. Initial results have emphasized proper evaluation of models and experimental setup, agreeing with the goals of the ML4Bio workshop. However, they have also found technical skills in programming to be important, highlighting the need for additional educational tools for these skills.

A ML4BIO WORKSHOP PRE-SURVEY

Informed Consent

University of Wisconsin - Madison
Research Participant Information and Consent Form

Study Title: Learning Machine Learning for Biologists: A study of a workshop for bioscience researchers
Principal Investigator: Anthony Gitter (Phone: (608) 316-4442; Email: gitter@biostat.wisc.edu)
Student Researcher: Chris Magnano (Email: cmagnano@wisc.edu)

Description of the research
You are invited to participate in a research study about machine learning education for biological researchers. We are inviting you because of your academic background and interest in learning about machine learning. The purpose of this research study is to learn how to train biological researchers in machine learning and what parts of machine learning are most important for biological researchers. This study will include about 50 total participants. This research will be conducted at University of Wisconsin-Madison (UW-Madison).

What will my participation involve?
If you decide to participate in this research study, we will use the pre- and post-workshop surveys and an assessment you will complete during the workshop for this research study. If you do not participate, we still ask that you complete these surveys and assessment but will not use your responses for research. However, completion of the surveys and assessment is not required for participation in the workshop. You may skip any question on the surveys or assessment that you do not wish to answer.

The surveys will take approximately 20 minutes to complete outside of the workshop. The assessment will be completed during the workshop and will also take about 20 minutes to complete.

Are there any risks to me?
Your name in all responses to the surveys and assessment will be replaced with a pseudonym. However, you may share information about your research area, which could uniquely identify you (for example, a research problem you are working on) that would make your responses impossible to fully de-identify. There is a risk that

How will my confidentiality be protected?
This study is confidential. Neither your name nor any other identifiable information will be published. Your name in all data collected will be replaced with a pseudonym. However, your responses could contain information which can identify you. We will limit access to your responses and store this information securely. We may publish and present what we learn from this study, including directly quoting your responses, but none of this information will identify you directly without your permission.

Whom should I contact if I have questions?
You may ask any questions about the research at any time. If you have questions about the research you should contact the Principal Investigator Anthony Gitter at gitter@biostat.wisc.edu.

If you are not satisfied with response of research team, have more questions, or want to talk with someone about your rights as a research participant, you should contact the Education and Social/Behavioral Science IRB Office at (608) 265-4312.

If you decide not to participate or to withdraw from the study, you may do so without penalty.

Your acceptance indicates that you have read this consent information, had an opportunity to ask any questions about your participation in this research, and voluntarily consent to participate.

- ○ I consent, begin the study
- ○ I do not consent, I do not wish to participate

Please enter the last two digits of your phone number followed by the two digit day of the month you were born. For instance, if your phone number is (608)-933-8057 and

your birthday is October 3, then your code would be 5703. This code will be used to link your pre- and post-workshop surveys, and will be destroyed after use.

Block 1

This first set of questions asks for some background information about your experience with machine learning.

Have you ever taken...

	Yes	No
...a machine learning course?	○	○
...a calculus course?	○	○
...a probability or statistics course?	○	○

How much...

	None at all	A little	A moderate amount	A lot	A great deal
...do you know about machine learning?	○	○	○	○	○
...coding experience do you have?	○	○	○	○	○
...research experience do you have with machine learning?	○	○	○	○	○
...experience with the command line interface on your computer do you have?	○	○	○	○	○

Which of the following **best** describes how you would like to split your research effort in the future?

- ○ 100% bench work, no computation
- ○ 75% bench work, 25% computation
- ○ 50% bench work, 50% computation
- ○ 25% bench work, 75% computation
- ○ No bench work, 100% computation

How comfortable would you be training classifiers for a research project on your own?

- ○ Not at all comfortable
- ○ A little comfortable
- ○ Somewhat comfortable
- ○ Very comfortable
- ○ Extremely uncomfortable

Which of the following best describes your current career stage?

- ○ Undergraduate student
- ○ Graduate student
- ○ Post-doctoral
- ○ Scientist
- ○ Principal investigator
- ○ [ ] Not listed, please tell us:

Block 2

This second set of questions asks about your current research and expectations for the workshop.

Do you encounter large datasets in your research (too large to navigate in a spreadsheet)?

- ○ Yes
- ○ No

What research questions do you want to answer with the data you collect or analyze?

(1-2 sentences)

What do you expect from the workshop? (1-2 sentences)

B ML4BIO WORKSHOP IN-WORKSHOP ASSESSMENT

ID gathering

University of Wisconsin - Madison
Research Participant Information and Consent Form

Study Title: Learning Machine Learning for Biologists: A study of a workshop for bioscience researchers
Principal Investigator: Anthony Gitter (Phone: (608) 316-4442; Email: gitter@biostat.wisc.edu)
Student Researcher: Chris Magnano (Email: cmagnano@wisc.edu)

Please enter the last two digits of your phone number followed by the two digit day of the month you were born. For instance, if your phone number is (608)-933-8057 and your birthday is October 3, then your code would be 5703. This code will be used to link your pre- and post-workshop surveys, and will be destroyed after use.

Block 1

Instructions: Please read these adapted excerpts from the paper, "RFMirTarget: predicting human microRNA target genes with a random forest classifier", fill in the chart on page 2, and answer the questions on page 3. Please note that the excerpts have been adapted for this activity, including significantly altering the experimental procedure and possibly introducing errors. Formulas for performance metrics can be found in lesson 03

In this paper we discuss and explore the predictive power of RFMirTarget, a ML approach for predicting human miRNAs target genes based on the random forests algorithm.

RFMirTarget is trained with a set of positive and negative examples of miRNA-target pairs that is pre-processed by the software miRanda in order to identify the actual interacting sites between each miRNA-mRNA pair and prepare the data set for feature extraction. The alignments provided by miRanda are the source for extracting features, which in turn are used to train the random forest classifier.

We train RFMirTarget with experimentally verified examples of human miRNA-target pairs. The data set is composed of 289 biologically validated positive examples extracted from miRecords database and 289 systematically identified tissue-specific negative examples. These examples were split into a training set consisting of 80% of instances and a testing set consisting of 20% of the data. Training and testing sets were balanced to contain equal proportions of positive and negative examples.

The set of descriptive features used to train RFMirTarget is divided into three categories: alignment features, thermodynamics features, and structural features.

1. Alignment features: Score and length of the miRNA-target alignment as evaluated by miRanda.
2. Thermodynamics features: Evaluation of the minimum free energy (MFE) of the complete miRNA-target alignment computed by RNAduplex.
3. Position-based features: Evaluation of each base pair from the 5′-most position of the miRNA up to the 30th position of the alignment, assigning nominal values to designate the kind of base pairing in each position: a G:C match, an A:U match, a G:U wobble pair, a gap and a mismatch.

To train this RF model we adopt the standard number of trees suggested by the randomForest R package, namely 500 trees. In order to determine the number of features each tree in the random forest should have we experimented giving trees between 1 and 35 features. We used 12 features for the final model as it had the highest specificity on the testing set.

The performance of RFMirTarget is assessed by computing the total prediction accuracy (ACC), specificity (SPE), sensitivity (SEN). Training set performance was ACC: 92.21, SEN: 93.73, SPE: 91.11. The classification results drawn from our experimental procedure was ACC: 78.98, SEN: 80.04, and SPE: 77.48.

Adapted from: Mendoza, Mariana R., et al. "RFMirTarget: predicting human microRNA target genes with a random forest classifier." PloS one 8.7 (2013): e70153.

In this excerpt, what is...

...the class label?

...the number of instances?

...the model?

...the evaluation metric(s) used?

How was the data split between training, testing, and validation?

Is there any evidence of data leakage?

Is there any evidence of overfitting?

Was the evaluation metric(s) used appropriate?

How did the model perform?

Do you trust the validity of these results?

C ML4BIO WORKSHOP POST-SURVEY

Informed Consent

University of Wisconsin - Madison

Study Title: Learning Machine Learning for Biologists: A study of a workshop for bioscience researchers
Principal Investigator: Anthony Gitter (Phone: (608) 316-4442; Email: gitter@biostat.wisc.edu)
Student Researcher: Chris Magnano (Email: cmagnano@wisc.edu)

Thank you for participating the the ML4Bio workshop. This survey will ask about your experiences during the workshop and any feedback you have. Reading and adjusting based on your feedback is the main method we have of improving the workshop.

Please enter the last two digits of your phone number followed by the two digit day of the month you were born. For instance, if your phone number is (608)-933-8057 and your birthday is October 3, then your code would be 5703. This code will be used to link your pre- and post-workshop surveys, and will be destroyed after use.

Block 1

Which of the following **best** describes how the workshop met your expectations?

- ○ Did not meet expectations
- ○ Met some expectations
- ○ Met most expectations
- ○ Met all expectations
- ○ Exceeded expectations

Which of your workshop expectations were not met?

How comfortable would you be training classifiers for a research project on your own?

- ○ Not at all comfortable
- ○ A little comfortable
- ○ Somewhat comfortable
- ○ Very comfortable
- ○ Extremely uncomfortable

For the next questions, please think about your knowledge before the workshop compared to your knowledge now.

	Not at all	A little	Somewhat	Very	Extremely
How confident would you be reviewing a paper in your field which uses machine learning **before** taking this workshop?	○	○	○	○	○
How confident would you be reviewing a paper in your field which uses machine learning **after** taking this workshop?	○	○	○	○	○
How knowledgeable about machine learning were you **before** taking this workshop?	○	○	○	○	○
How knowledgeable about machine learning are you **after** taking this workshop?	○	○	○	○	○

Imagine that in your research, you encountered a problem well-suited to machine learning.

	Not at all	A little	Somewhat	Very	Extremely
How confident are you that **before** this workshop you would identify the problem as being well-suited to machine learning?	○	○	○	○	○
How confident are you that **after** this workshop you would identify the problem as being well-suited to machine learning?	○	○	○	○	○
How likely would you be to pursue machine learning for that problem **before** this workshop?	○	○	○	○	○
How likely would you be to pursue machine learning for that problem **after** this workshop?	○	○	○	○	○

Which workshop topics were most valuable for you?

Which workshop topics would be difficult for you to explain to a friend at a high level?

Overall, how would you rate the pace of the workshop?

- ○ Slower than I would have liked
- ○ About the right pace
- ○ Faster than I would have liked

How would you rate the amount of time spent on each of the following topics?

	Not enough time spent	About the right amount of time spent	Too much time spent
Using the software	○	○	○
Viewing the website	○	○	○
Interacting with the workshop organizers	○	○	○
In breakout rooms	○	○	○
Discussing math and computational background	○	○	○
Presenting biological motivation and examples	○	○	○
Presenting the machine learning workflow	○	○	○
Presenting specific classifiers	○	○	○
Discussing scenarios	○	○	○
Creating charts of academic papers	○	○	○

What questions about machine learning do you have that were not addressed during the workshop?

Please share any other suggestions for improving the workshop, ml4bio software, website, guides, or other materials.

Was there any information you wish you had before the workshop?

REFERENCES

Ahmed, N. K., J. Neville, R. A. Rossi, and N. Duffield. 2015. Efficient graphlet counting for large networks. In *2015 ieee international conference on data mining*, 1–10.

Akaike, Hirotugu. 1998. A New Look at the Statistical Model Identification. In *Selected Papers of Hirotugu Akaike*, ed. Emanuel Parzen, Kunio Tanabe, and Genshiro Kitagawa, 215–222. Springer Series in Statistics, New York, NY: Springer New York.

Alaa, Asem, Ayman M. Eldeib, and Ahmed A. Metwally. 2019. Protein Subcellular Localization Prediction Based on Internal Micro-similarities of Markov Chains. In *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, 1355–1358.

Albert, Réka, Bhaskar DasGupta, Rashmi Hegde, Gowri Sangeetha Sivanathan, Anthony Gitter, Gamze Gürsoy, Pradyut Paul, and Eduardo Sontag. 2011. Computationally efficient measure of topological redundancy of biological and social networks. *Physical Review. E, Statistical, Nonlinear, and Soft Matter Physics* 84(3 Pt 2):036117.

Albrecht, Steffen, Maximilian Sprang, Miguel A. Andrade-Navarro, and Jean-Fred Fontaine. 2021. seqQscorer: automated quality control of next-generation sequencing data using machine learning. *Genome Biology* 22(1):75.

Almagro Armenteros, José Juan, Casper Kaae Sønderby, Søren Kaae Sønderby, Henrik Nielsen, and Ole Winther. 2017. DeepLoc: prediction of protein subcellular localization using deep learning. *Bioinformatics* 33(21):3387–3395.

Ambrose, Susan A., Michael W. Bridges, Michele DiPietro, Marsha C. Lovett, and Marie K. Norman. 2010. *How learning works: Seven research-based principles for smart teaching*. How Learning Works: Seven Research-Based Principles for Smart Teaching, San Francisco, CA, US: Jossey-Bass.

Ananda, M. Mondal, and Jianjun Hu. 2010. NetLoc: Network based protein localization prediction using protein-protein interaction and co-expression networks. In *2010 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 142–148. Journal Abbreviation: 2010 IEEE International Conference on Bioinformatics and Biomedicine (BIBM).

Angermueller, Christof, Heather J. Lee, Wolf Reik, and Oliver Stegle. 2017. DeepCpG: accurate prediction of single-cell DNA methylation states using deep learning. *Genome Biology* 18(1):67.

Ashmore, Rob, Radu Calinescu, and Colin Paterson. 2021. Assuring the Machine Learning Lifecycle: Desiderata, Methods, and Challenges. *ACM Computing Surveys* 54(5):111:1–111:39.

Bailly-Bechet, M., C. Borgs, A. Braunstein, J. Chayes, A. Dagkessamanskaia, J.-M. François, and R. Zecchina. 2011. Finding undetected protein associations in cell signaling by belief propagation. *Proceedings of the National Academy of Sciences* 108(2):882–887.

Balandat, Maximilian, Brian Karrer, Daniel Jiang, Samuel Daulton, Ben Letham, Andrew G Wilson, and Eytan Bakshy. 2020. Botorch: A framework for efficient monte-carlo bayesian optimization. In *Advances in neural information processing systems*, ed. H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, vol. 33, 21524–21538. Curran Associates, Inc.

Bandura, A. 1977. Self-efficacy: toward a unifying theory of behavioral change. *Psychological Review* 84(2):191–215.

Barabási, Albert-László, and Zoltán N. Oltvai. 2004. Network biology: understanding the cell's functional organization. *Nature Reviews Genetics* 5(2):101–113.

Barone, Lindsay, Jason Williams, and David Micklos. 2017. Unmet needs for analyzing biological big data: A survey of 704 NSF principal investigators. *PLOS Computational Biology* 13(10):e1005755.

Basha, Omer, Omry Mauer, Eyal Simonovsky, Rotem Shpringer, and Esti Yeger-Lotem. 2019. ResponseNet v.3: revealing signaling and regulatory pathways connecting your proteins and genes across human tissues. *Nucleic Acids Research* 47(W1):W242–W247.

Basha, Omer, Shoval Tirman, Amir Eluk, and Esti Yeger-Lotem. 2013. ResponseNet2.0: revealing signaling and regulatory pathways connecting your proteins and genes–now with human data. *Nucleic Acids Research* 41(W1):W198–W203.

Bauer, Nicholas C., Paul W. Doetsch, and Anita H. Corbett. 2015. Mechanisms regulating protein localization. *Traffic* 16(10):1039–1061. `https://onlinelibrary.wiley.com/doi/pdf/10.1111/tra.12310`.

Beaulieu-Jones, Brett K, and Casey S Greene. 2017. Reproducibility of computational workflows is automated using continuous analysis. *Nature Biotechnology* 35(4):342–346.

Becker, Jordan T, and Nathan M Sherer. 2017. Subcellular Localization of HIV-1 gag-pol mRNAs Regulates Sites of Virion Assembly. *Journal of virology* 91(6):e02315–16.

Bhanji, Farhan, Ronald Gottesman, Willem de Grave, Yvonne Steinert, and Laura R. Winer. 2012. The retrospective pre–post: A practical method to evaluate learning from an educational program. *Academic Emergency Medicine* 19(2):189–194.

Biggs, John B, and Kevin F Collis. 1982. *Evaluating the quality of learning: The solo taxonomy (structure of the observed learning outcome)*. Academic Press.

Bindea, Gabriela, Bernhard Mlecnik, Hubert Hackl, Pornpimol Charoentong, Marie Tosolini, Amos Kirilovsky, Wolf-Herman Fridman, Franck Pagès, Zlatko Trajanoski, and Jérôme Galon. 2009. ClueGO: a Cytoscape plug-in to decipher functionally grouped gene ontology and pathway annotation networks. *Bioinformatics* 25(8):1091–1093.

Binder, Janos X., Sune Pletscher-Frankild, Kalliopi Tsafou, Christian Stolte, Seán I. O'Donoghue, Reinhard Schneider, and Lars Juhl Jensen. 2014. COMPARTMENTS: unification and visualization of protein subcellular localization evidence. *Database* 2014(bau012). Bau012, `https://academic.oup.com/database/article-pdf/doi/10.1093/database/bau012/8244417/bau012.pdf`.

Black, Paul, and Dylan Wiliam. 2009. Developing the theory of formative assessment. *Educational Assessment, Evaluation and Accountability (formerly: Journal of Personnel Evaluation in Education)* 21(1):5.

Blise, Katie E., Shamilene Sivagnanam, Grace L. Banik, Lisa M. Coussens, and Jeremy Goecks. 2021. Single-Cell Spatial Proteomics Analyses of Head and Neck Squamous Cell Carcinoma Reveal Tumor Heterogeneity and Immune Architectures Associated with Clinical Outcome.

Boughorbel, Sabri, Fethi Jarray, and Mohammed El-Anbari. 2017. Optimal classifier for imbalanced data using Matthews Correlation Coefficient metric. *PLOS ONE* 12(6):e0177678.

Brazas, Michelle D., Sarah Blackford, and Teresa K. Attwood. 2017. Plug gap in essential bioinformatics skills. *Nature* 544(7649):161–161.

Brazas, Michelle D, and B F Francis Ouellette. 2016. Continuing Education Workshops in Bioinformatics Positively Impact Research and Careers. *PLoS computational biology* 12(6): e1004916–e1004916. Publisher: Public Library of Science.

Brody, Shaked, Uri Alon, and Eran Yahav. 2021. How attentive are graph attention networks? *arXiv preprint arXiv:2105.14491*.

Brown, Ann L., and Joseph C. Campione. 1996. Psychological theory and the design of innovative learning environments: On procedures, principles, and systems. In *Innovations in learning: New environments for education*, 289–325. Hillsdale, NJ, US: Lawrence Erlbaum Associates, Inc.

Budak, Gungor, Oyku Eren Ozsoy, Yesim Aydin Son, Tolga Can, and Nurcan Tuncbag. 2015. Reconstruction of the temporal signaling network in Salmonella-infected human cells. *Frontiers in Microbiology* 6:730.

Bünnagel, Ursula, Bernhard Korte, and Jens Vygen. 1998. Efficient implementation of the Goldberg-Tarjan minimum-cost flow algorithm. *Optimization Methods and Software* 10(2): 157–174.

Cao, Lulu, Yiyuan Ding, Norris Hung, Kebing Yu, Anna Ritz, Benjamin J. Raphael, and Arthur R. Salomon. 2012. Quantitative Phosphoproteomics Reveals SLP-76 Dependent Regulation of PAG and Src Family Kinases in T Cells. *PLOS ONE* 7(10):e46725.

Cerami, Ethan, Emek Demir, Nikolaus Schultz, Barry S. Taylor, and Chris Sander. 2010. Automated Network Analysis Identifies Core Pathways in Glioblastoma. *PLOS ONE* 5(2): e8918.

Chan, Sang. 2010. Applications of Andragogy in Multi-Disciplined Teaching and Learning. *Journal of Adult Education* 39(2):25–35.

Chang, Jeffrey. 2015. Core services: Reward bioinformaticians. *Nature* 520(7546):151–152.

Chautard, Emilie, Marie Fatoux-Ardore, Lionel Ballut, Nicolas Thierry-Mieg, and Sylvie Ricard-Blum. 2010. MatrixDB, the extracellular matrix interaction database. *Nucleic Acids Research* 39(suppl_1):D235–D240.

Chen, Ping, Wolfgang Hübner, Matthew A. Spinelli, and Benjamin K. Chen. 2007. Predominant mode of human immunodeficiency virus transfer between t cells is mediated by sustained env-dependent neutralization-resistant virological synapses. *Journal of Virology* 81(22):12582–12595. `https://jvi.asm.org/content/81/22/12582.full.pdf`.

Chhatwal, Jagpreet, Oguzhan Alagoz, Mary J. Lindstrom, Charles E. Kahn, Katherine A. Shaffer, and Elizabeth S. Burnside. 2009. A Logistic Regression Model Based on the National Mammography Database Format to Aid Breast Cancer Diagnosis. *American Journal of Roentgenology* 192(4):1117–1127.

Chicco, Davide. 2017. Ten quick tips for machine learning in computational biology. *BioData Mining* 10(1):35.

Choobdar, Sarvenaz, Mehmet E. Ahsen, Jake Crawford, Mattia Tomasoni, Tao Fang, David Lamparter, Junyuan Lin, Benjamin Hescott, Xiaozhe Hu, Johnathan Mercer, Ted Natoli, Rajiv Narayan, The DREAM Module Identification Challenge Consortium, Aravind Subramanian, Gustavo Stolovitzky, Zoltán Kutalik, Kasper Lage, Donna K. Slonim, Julio Saez-Rodriguez, Lenore J. Cowen, Sven Bergmann, and Daniel Marbach. 2019. Assessment of network module identification across complex diseases. *Nature Methods* 16(9):843–852.

Cordero, Radames J. B., Carlos M. de León-Rodriguez, John K. Alvarado-Torres, Ana R. Rodriguez, and Arturo Casadevall. 2016. Life science's average publishable unit (apu) has increased over the past two decades. *PLOS ONE* 11(6):1–14.

Cowen, Lenore, Trey Ideker, Benjamin J. Raphael, and Roded Sharan. 2017. Network propagation: a universal amplifier of genetic associations. *Nature Reviews Genetics* 18(9):551–562.

Crowe, Alison, Clarissa Dirks, and Mary Pat Wenderoth. 2008. Biology in Bloom: Implementing Bloom's Taxonomy to Enhance Student Learning in Biology. *CBE-Life Sciences Education* 7(4):368–381.

Csermely, Peter, Tamás Korcsmáros, Huba J.M. Kiss, Gábor London, and Ruth Nussinov. 2013. Structure and dynamics of molecular networks: A novel paradigm of drug discovery: A comprehensive review. *Pharmacology & Therapeutics* 138(3):333–408.

Dagliyan, Onur, Fadime Uney-Yuksektepe, I. Halil Kavakli, and Metin Turkay. 2011. Optimization Based Tumor Classification from Microarray Gene Expression Data. *PLOS ONE* 6(2):e14579.

DeBlasio, Dan, and John Kececioglu. 2015. Parameter advising for multiple sequence alignment. *BMC Bioinformatics* 16(2):A3.

———. 2017. Learning Parameter-Advising Sets for Multiple Sequence Alignment. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 14(5):1028–1041.

Defferrard, Michaël, Xavier Bresson, and Pierre Vandergheynst. 2016. Convolutional neural networks on graphs with fast localized spectral filtering. In *Advances in neural information processing systems*, ed. D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, vol. 29. Curran Associates, Inc.

Demir, Emek, Michael P Cary, Suzanne Paley, Ken Fukuda, Christian Lemer, Imre Vastrik, Guanming Wu, Peter D'Eustachio, Carl Schaefer, Joanne Luciano, Frank Schacherer, Irma Martinez-Flores, Zhenjun Hu, Veronica Jimenez-Jacinto, Geeta Joshi-Tope, Kumaran Kandasamy, Alejandra C Lopez-Fuentes, Huaiyu Mi, Elgar Pichler, Igor Rodchenkov, Andrea Splendiani, Sasha Tkachev, Jeremy Zucker, Gopal Gopinath, Harsha Rajasimha, Ranjani Ramakrishnan, Imran Shah, Mustafa Syed, Nadia Anwar, Özgün Babur, Michael Blinov, Erik Brauner, Dan Corwin, Sylva Donaldson, Frank Gibbons, Robert Goldberg, Peter Hornbeck, Augustin Luna, Peter Murray-Rust, Eric Neumann, Oliver Ruebenacker, Matthias Samwald, Martijn van Iersel, Sarala Wimalaratne, Keith Allen, Burk Braun, Michelle Whirl-Carrillo, Kei-Hoi Cheung, Kam Dahlquist, Andrew Finney, Marc Gillespie, Elizabeth Glass, Li Gong, Robin Haw, Michael Honig, Olivier Hubaut, David Kane, Shiva Krupa, Martina Kutmon, Julie Leonard, Debbie Marks, David Merberg, Victoria Petri, Alex Pico, Dean Ravenscroft, Liya Ren, Nigam Shah, Margot Sunshine, Rebecca Tang, Ryan Whaley, Stan Letovksy, Kenneth H Buetow, Andrey Rzhetsky, Vincent Schachter, Bruno S Sobral, Ugur Dogrusoz, Shannon McWeeney, Mirit Aladjem, Ewan Birney, Julio Collado-Vides, Susumu Goto, Michael Hucka, Nicolas Le Novère, Natalia Maltsev, Akhilesh Pandey, Paul Thomas, Edgar Wingender, Peter D Karp, Chris Sander, and Gary D Bader. 2010. The BioPAX community standard for pathway data sharing. *Nature Biotechnology* 28(9):935–942.

Demšar, Janez, Tomaž Curk, Aleš Erjavec, Črt Gorup, Tomaž Hočevar, Mitar Milutinovič, Martin Možina, Matija Polajnar, Marko Toplak, Anže Starič, Miha Štajdohar, Lan Umek, Lan Žagar, Jure Žbontar, Marinka Žitnik, and Blaž Zupan. 2013. Orange: Data mining toolbox in Python. *The Journal of Machine Learning Research* 14(1):2349–2353.

Dinsdale, Elizabeth, Sarah C. R. Elgin, Neal Grandgenett, William Morgan, Anne Rosenwald, William Tapprich, Eric W. Triplett, and Mark A. Pauley. 2015. NIBLSE: A network for integrating bioinformatics into life sciences education. *CBE-Life Sciences Education* 14(4):le3.

Dou, Dan, Rebecca Revol, Henrik Östbye, Hao Wang, and Robert Daniels. 2018. Influenza a virus cell entry, replication, virion assembly and movement. *Frontiers in Immunology* 9: 1581–1581.

Drawid, Amar, and Mark Gerstein. 2000. A Bayesian system integrating expression data with sequence patterns for localizing proteins: comprehensive application to the yeast genome11Edited by F. Cohen. *Journal of Molecular Biology* 301(4):1059–1075.

Du, Pufeng, and Lusheng Wang. 2014. Predicting Human Protein Subcellular Locations by the Ensemble of Multiple Predictors via Protein-Protein Interaction Network with Edge Clustering Coefficients. *PLOS ONE* 9(1):e86879.

Dua, Dheeru, and Casey Graff. 2017. UCI machine learning repository.

D'Souza, Rochelle C. J., Anna M. Knittle, Nagarjuna Nagaraj, Maarten van Dinther, Chunaram Choudhary, Peter ten Dijke, Matthias Mann, and Kirti Sharma. 2014. Time-resolved dissection of early phosphoproteome and ensuing proteome changes in response to tgf-β. *Science Signaling* 7(335):rs5–rs5. `https://www.science.org/doi/pdf/10.1126/scisignal.2004856`.

Eisenhaber, F., and P. Bork. 1998. Wanted: Subcellular localization of proteins based on sequence. *Trends in Cell Biology* 8(4):169–170.

Elia, Miriam, Carola Gajek, Alexander Schiendorfer, and Wolfgang Reif. 2021. An interactive web application for decision tree learning. In *Proceedings of the first teaching machine learning and artificial intelligence workshop*, ed. Bernd Bischl, Oliver Guhr, Heidi Seibold, and Peter Steinbach, vol. 141 of *Proceedings of Machine Learning Research*, 11–16. PMLR.

Evans, Jeffrey S., and Samuel A. Cushman. 2009. Gradient modeling of conifer species using random forests. *Landscape Ecology* 24(5):673–683.

Fabregat, Antonio, Steven Jupe, Lisa Matthews, Konstantinos Sidiropoulos, Marc Gillespie, Phani Garapati, Robin Haw, Bijay Jassal, Florian Korninger, Bruce May, Marija Milacic, Corina Duenas Roca, Karen Rothfels, Cristoffer Sevilla, Veronica Shamovsky, Solomon Shorser, Thawfeek Varusai, Guilherme Viteri, Joel Weiser, Guanming Wu, Lincoln Stein, Henning Hermjakob, and Peter D'Eustachio. 2018. The Reactome Pathway Knowledgebase. *Nucleic Acids Research* 46(D1):D649–D655.

Feng, Yifan, Haoxuan You, Zizhao Zhang, Rongrong Ji, and Yue Gao. 2019. Hypergraph neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence* 33(01):3558–3565.

Fey, Matthias, and Jan E. Lenssen. 2019. Fast graph representation learning with PyTorch Geometric. In *Iclr workshop on representation learning on graphs and manifolds*.

Freeman, Scott, Sarah L. Eddy, Miles McDonough, Michelle K. Smith, Nnadozie Okoroafor, Hannah Jordt, and Mary Pat Wenderoth. 2014. Active learning increases student performance in science, engineering, and mathematics. *Proceedings of the National Academy of Sciences* 111(23):8410–8415.

Furlong, Laura I. 2013. Human diseases through the lens of network biology. *Trends in Genetics* 29(3):150–159.

Fyshe, Alona, Yifeng Liu, Duane Szafron, Russ Greiner, and Paul Lu. 2008. Improving subcellular localization prediction using text classification and the gene ontology. *Bioinformatics* 24(21):2512–2517.

Garapati, Hita Sony, Gurranna Male, and Krishnaveni Mishra. 2020. Predicting subcellular localization of proteins using protein-protein interaction data. *Genomics* 112(3):2361–2368.

Gardy, Jennifer L., and Fiona S. L. Brinkman. 2006. Methods for predicting bacterial protein subcellular localization. *Nature Reviews Microbiology* 4(10):741–751.

Gewali, Utsav B., and Sildomar T. Monteiro. 2018. A tutorial on modelling and inference in undirected graphical models for hyperspectral image analysis. *International Journal of Remote Sensing* 39(20):7104–7143. `https://doi.org/10.1080/01431161.2018.1465614`.

Goh, Wilson W. B., Yie H. Lee, Maxey Chung, and Limsoon Wong. 2012. How advancement in biological network analysis methods empowers proteomics. *Proteomics* 12(4-5):550–563.

Goldberg, Andrew V., and Robert E. Tarjan. 1990. Finding Minimum-Cost Circulations by Successive Approximation. *Mathematics of Operations Research* 15(3):430–466.

Greener, Joe G., Shaun M. Kandathil, Lewis Moffat, and David T. Jones. 2022. A guide to machine learning for biologists. *Nature Reviews Molecular Cell Biology* 23(1):40–55.

Grover, Aayush, and Laurent Gatto. 2022. Protfinder: finding subcellular locations of proteins using protein interaction networks. *bioRxiv*. `https://www.biorxiv.org/content/early/2022/01/14/2022.01.11.475836.full.pdf`.

Gu, Qiang, Anup Kumar, Simon Bray, Allison Creason, Alireza Khanteymoori, Vahid Jalili, Björn Grüning, and Jeremy Goecks. 2021. Galaxy-ML: An accessible, reproducible, and scalable machine learning toolkit for biomedicine. *PLOS Computational Biology* 17(6):e1009014.

Guzzi, Pietro Hiram, and Tijana Milenković. 2018. Survey of local and global biological network alignment: The need to reconcile the two sides of the same coin. *Briefings in Bioinformatics* 19(3):472–481.

Hammond, David K., Pierre Vandergheynst, and Rémi Gribonval. 2011. Wavelets on graphs via spectral graph theory. *Applied and Computational Harmonic Analysis* 30(2):129–150.

Han, Julianna, Jasmine T. Perez, Cindy Chen, Yan Li, Asiel Benitez, Matheswaran Kandasamy, Yoontae Lee, Jorge Andrade, Benjamin tenOever, and Balaji Manicassamy. 2018. Genome-wide CRISPR/Cas9 screen identifies host factors essential for influenza virus replication. *Cell Reports* 23(2):596–607.

Horn, Heiko, Michael S Lawrence, Candace R Chouinard, Yashaswi Shrestha, Jessica Xin Hu, Elizabeth Worstell, Emily Shea, Nina Ilic, Eejung Kim, Atanas Kamburov, Alireza Kashani, William C Hahn, Joshua D Campbell, Jesse S Boehm, Gad Getz, and Kasper Lage. 2018. NetSig: network-based discovery from cancer genomes. *Nature Methods* 15(1):61–66.

Hornbeck, Peter V., Bin Zhang, Beth Murray, Jon M. Kornhauser, Vaughan Latham, and Elzbieta Skrzypek. 2014. PhosphoSitePlus, 2014: mutations, PTMs and recalibrations. *Nucleic Acids Research* 43(D1):D512–D520. `http://oup.prod.sis.lan/nar/article-pdf/43/D1/D512/17437800/gku1267.pdf`.

Hua, Sujun, and Zhirong Sun. 2001. Support vector machine approach for protein subcellular localization prediction. *Bioinformatics* 17(8):721–728.

Huang, Da Wei, Brad T Sherman, and Richard A Lempicki. 2008. Systematic and integrative analysis of large gene lists using DAVID bioinformatics resources. *Nature Protocols* 4:44.

Hubbard, Katharine E., and Sonja D. Dunbar. 2017. Perceptions of scientific research literature and strategies for reading papers depend on academic career stage. *PLOS ONE* 12(12):1–16.

Humphrey, Sean J, S Babak Azimifar, and Matthias Mann. 2015. High-throughput phosphoproteomics reveals in vivo insulin signaling dynamics. *Nature Biotechnology* 33(9):990–995.

Hung, Mien-Chie, and Wolfgang Link. 2011. Protein localization in disease and therapy. *Journal of Cell Science* 124(20):3381.

Imai, Kenichiro, and Kenta Nakai. 2010. Prediction of subcellular locations of proteins: Where to proceed? *PROTEOMICS* 10(22):3970–3983.

Jean Beltran, Pierre M., Rommel A. Mathias, and Ileana M. Cristea. 2016. A Portrait of the Human Organelle Proteome In Space and Time during Cytomegalovirus Infection. *Cell Systems* 3(4):361–373.e6.

Johansson, Fredrik D., Otto Frost, Carl Retzner, and Devdatt Dubhashi. 2015. Classifying large graphs with differential privacy. In *Modeling decisions for artificial intelligence*, ed. Vicenc Torra and Torra Narukawa, 3–17. Cham: Springer International Publishing.

Jones, David T. 2019. Setting the standards for machine learning in biology. *Nature Reviews Molecular Cell Biology* 20(11):659–660.

Kandasamy, Kumaran, S. Sujatha Mohan, Rajesh Raju, Shivakumar Keerthikumar, Ghantasala S. Sameer Kumar, Abhilash K. Venugopal, Deepthi Telikicherla, J. Daniel Navarro, Suresh Mathivanan, Christian Pecquet, Sashi Kanth Gollapudi, Sudhir Gopal Tattikota, Shyam Mohan, Hariprasad Padhukasahasram, Yashwanth Subbannayya, Renu Goel, Harrys K. C. Jacob, Jun Zhong, Raja Sekhar, Vishalakshi Nanjappa, Lavanya Balakrishnan, Roopashree Subbaiah, Y. L. Ramachandra, B. Abdul Rahiman, T. S. Keshava Prasad, Jian-Xin Lin, Jon C. D. Houtman, Stephen Desiderio, Jean-Christophe Renauld, Stefan N. Constantinescu, Osamu Ohara, Toshio Hirano, Masato Kubo, Sujay Singh, Purvesh Khatri, Sorin Draghici, Gary D. Bader, Chris Sander, Warren J. Leonard, and Akhilesh Pandey. 2010. NetPath: a public resource of curated signal transduction pathways. *Genome Biology* 11(1): R3.

Kanehisa, M., and S. Goto. 2000. KEGG: kyoto encyclopedia of genes and genomes. *Nucleic Acids Research* 28(1):27–30.

Kececioglu, John, and Dan DeBlasio. 2013. Accuracy Estimation and Parameter Advising for Protein Multiple Sequence Alignment. *Journal of Computational Biology* 20(4):259–279.

Kedaigle, Amanda J., and Ernest Fraenkel. 2018. Discovering altered regulation and signaling through network-based integration of transcriptomic, epigenomic and proteomic tumor data. *Methods in Molecular Biology (Clifton, N.J.)* 1711:13–26.

Khurana, Vikram, Jian Peng, Chee Yeun Chung, Pavan K. Auluck, Saranna Fanning, Daniel F. Tardiff, Theresa Bartels, Martina Koeva, Stephen W. Eichhorn, Hadar Benyamini, Yali Lou, Andy Nutter-Upham, Valeriya Baru, Yelena Freyzon, Nurcan Tuncbag, Michael Costanzo, Bryan-Joseph San Luis, David C. Schöndorf, M. Inmaculada Barrasa, Sepehr Ehsani, Neville Sanjana, Quan Zhong, Thomas Gasser, David P. Bartel, Marc Vidal, Michela Deleidi, Charles Boone, Ernest Fraenkel, Bonnie Berger, and Susan Lindquist. 2017. Genome-Scale Networks Link Neurodegenerative Disease Genes to α-Synuclein through Specific Molecular Pathways. *Cell Systems* 4(2):157–170.e14.

Kipf, Thomas N., and Max Welling. 2017. Semi-supervised classification with graph convolutional networks. In *International conference on learning representations (iclr)*.

Köksal, Ali Sinan, Kirsten Beck, Dylan R. Cronin, Aaron McKenna, Nathan D. Camp, Saurabh Srivastava, Matthew E. MacGilvray, Rastislav Bodík, Alejandro Wolf-Yadlin, Ernest Fraenkel, Jasmin Fisher, and Anthony Gitter. 2018. Synthesizing Signaling Pathways from Temporal Phosphoproteomic Data. *Cell Reports* 24(13):3607–3618.

Komurov, Kakajan, Michael A. White, and Prahlad T. Ram. 2010. Use of Data-Biased Random Walks on Graphs for the Retrieval of Context-Specific Networks from Genomic Data. *PLOS Computational Biology* 6(8):e1000889.

Kosov, Sergey. 2018. Multi-layer conditional random fields for revealing unobserved entities. Ph.D. thesis, Universität Siegen.

Kulkarni, Pranav, and Peter Frommolt. 2017. Challenges in the Setup of Large-scale Next-Generation Sequencing Analysis Workflows. *Computational and structural biotechnology journal* 15:471–477. Publisher: Research Network of Computational and Structural Biotechnology.

Kumar, Sanjiv, and Martial Hebert. 2006. Discriminative random fields. *International Journal of Computer Vision* 68(2):179–201. Sanjiv Kumar is currently with Google Research, Pittsburgh, PA, USA. His contact email is: sanjivk@google.com.

Lee, Michael, Erdahl T Teber, Oliver Holmes, Katia Nones, Ann-Marie Patch, Rebecca A Dagg, Loretta M S Lau, Joyce H Lee, Christine E Napier, Jonathan W Arthur, Sean M Grimmond, Nicholas K Hayward, Peter A Johansson, Graham J Mann, Richard A Scolyer, James S Wilmott, Roger R Reddel, John V Pearson, Nicola Waddell, and Hilda A Pickett. 2018. Telomere sequence content can be used to determine ALT activity in tumours. *Nucleic Acids Research* 46(10):4903–4918.

List, Markus, Peter Ebert, and Felipe Albrecht. 2017. Ten Simple Rules for Developing Usable Software in Computational Biology. *PLOS Computational Biology* 13(1):e1005265. Publisher: Public Library of Science.

Listgarten, Jennifer, Sambasivarao Damaraju, Brett Poulin, Lillian Cook, Jennifer Dufour, Adrian Driga, John Mackey, David Wishart, Russ Greiner, and Brent Zanke. 2004. Predictive models for breast cancer susceptibility from multiple single nucleotide polymorphisms. *Clinical Cancer Research* 10(8):2725–2737.

Littmann, Maria, Katharina Selig, Liel Cohen-Lavi, Yotam Frank, Peter Hönigschmid, Evans Kataka, Anja Mösch, Kun Qian, Avihai Ron, Sebastian Schmid, Adam Sorbie, Liran Szlak, Ayana Dagan-Wiener, Nir Ben-Tal, Masha Y. Niv, Daniel Razansky, Björn W. Schuller, Donna Ankerst, Tomer Hertz, and Burkhard Rost. 2020. Validity of machine learning in biology and medicine increased through collaborations across fields of expertise. *Nature Machine Intelligence* 2(1):18–24.

Liu, Yun, Po-Hsuan Cameron Chen, Jonathan Krause, and Lily Peng. 2019. How to Read Articles That Use Machine Learning: Users' Guides to the Medical Literature. *JAMA* 322(18): 1806–1816.

Lundberg, Emma, and Georg H. H. Borner. 2019. Spatial proteomics: A powerful discovery tool for cell biology. *Nature Reviews Molecular Cell Biology* 20(5):285–302.

Ma, Cheng-Yu, and Chung-Shou Liao. 2020. A review of protein–protein interaction network alignment: From pathway comparison to global alignment. *Computational and Structural Biotechnology Journal* 18:2647–2656.

MacGilvray, Matthew E., Evgenia Shishkova, Deborah Chasman, Michael Place, Anthony Gitter, Joshua J. Coon, and Audrey P. Gasch. 2018. Network inference reveals novel connections in pathways regulating growth and defense in the yeast salt response. *PLOS Computational Biology* 13(5):1–28.

Magar, Rishikesh, Prakarsh Yadav, and Amir Barati Farimani. 2021. Potential neutralizing antibodies discovered for novel corona virus using machine learning. *Scientific Reports* 11(1): 5261.

Magnano, Chris S, and Anthony Gitter. 2021. Automating parameter selection to avoid implausible biological pathway models. *NPJ systems biology and applications* 7(1):1–12.

Magnano, Chris S, Fangzhou Mu, Rosemary S Russ, Milica Cvetkovic, Debora Treu, and Anthony Gitter. 2022. An approachable, flexible, and practical machine learning workshop for biologists. *bioRxiv*. `https://www.biorxiv.org/content/early/2022/02/04/2022.02.03.479008.full.pdf`.

Mangul, Serghei, Thiago Mosqueiro, Richard J. Abdill, Dat Duong, Keith Mitchell, Varuni Sarwal, Brian Hill, Jaqueline Brito, Russell Jared Littman, Benjamin Statz, Angela Ka-Mei Lam, Gargi Dayama, Laura Grieneisen, Lana S. Martin, Jonathan Flint, Eleazar Eskin, and Ran Blekhman. 2019. Challenges and recommendations to improve the installability and archival stability of omics computational tools. *PLOS Biology* 17(6):e3000333. Publisher: Public Library of Science.

Matthews, B.W. 1975. Comparison of the predicted and observed secondary structure of T4 phage lysozyme. *Biochimica et Biophysica Acta (BBA) - Protein Structure* 405(2):442–451.

McClatchy, Susan, Kristin M. Bass, Daniel M. Gatti, Adam Moylan, and Gary Churchill. 2020. Nine quick tips for efficient bioinformatics curriculum development and training. *PLOS Computational Biology* 16(7):1–10.

Mendoza, Mariana R., Guilherme C. da Fonseca, Guilherme Loss-Morais, Ronnie Alves, Rogerio Margis, and Ana L. C. Bazzan. 2013. RFMirTarget: Predicting human microRNA target genes with a random forest classifier. *PLOS ONE* 8(7):1–18.

Milo, R., S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, and U. Alon. 2002. Network Motifs: Simple Building Blocks of Complex Networks. *Science* 298(5594):824–827.

Mitchell, Tom. 1997. *Machine learning*. McGraw Hill.

Moriya, Yuki, Masumi Itoh, Shujiro Okuda, Akiyasu C. Yoshizawa, and Minoru Kanehisa. 2007. KAAS: an automatic genome annotation and pathway reconstruction server. *Nucleic Acids Research* 35(suppl_2):W182–W185.

Mulder, Nicola, Russell Schwartz, Michelle D. Brazas, Cath Brooksbank, Bruno Gaeta, Sarah L. Morgan, Mark A. Pauley, Anne Rosenwald, Gabriella Rustici, Michael Sierk, Tandy Warnow, and Lonnie Welch. 2018. The development and application of bioinformatics core competencies to improve bioinformatics training and education. *PLOS Computational Biology* 14(2):1–14.

Murphy, Kevin P. 2022. *Probabilistic machine learning: An introduction*. MIT Press.

National Institutes of Health. 2021a. Data generation projects for the NIH bridge to artificial intelligence (Bridge2AI) program (OT2). https://commonfund.nih.gov/sites/default/files/OT2-Data-Generation-Projects-B2AI-051321-508.pdf.

———. 2021b. Notice of special interest (NOSI): Administrative supplements for workforce development at the interface of information sciences, artificial intelligence and machine learning (AI/ML), and biomedical sciences. https://grants.nih.gov/grants/guide/notice-files/NOT-OD-21-079.html.

National Science Foundation. 2020. National artificial intelligence (AI) research institutes. https://www.nsf.gov/pubs/2020/nsf20604/nsf20604.htm.

Newman, M. E. J. 2002. Assortative mixing in networks. *Physical Review Letters* 89:208701.

Park, Eunju. 2018. *Analyzing contributions of host cellular proteins and signaling pathways to hiv-1 replication and cell-cell spread*. University of Wisconsin-Madison.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12:2825–2830.

Pillai, Ignazio, Giorgio Fumera, and Fabio Roli. 2017. Designing multi-label classifiers that maximize f measures: State of the art. *Pattern Recognition* 61:394–404.

Precone, Vincenza, Valentina Del Monaco, Maria Valeria Esposito, Fatima Domenica Elisa De Palma, Anna Ruocco, Francesco Salvatore, and Valeria D'Argenio. 2015. Cracking the Code of Human Diseases Using Next-Generation Sequencing: Applications, Challenges, and Perspectives. *BioMed research international* 2015:161648–161648. Edition: 2015/11/19 Publisher: Hindawi Publishing Corporation.

Project Jupyter, Matthias Bussonnier, Jessica Forde, Jeremy Freeman, Brian Granger, Tim Head, Chris Holdgraf, Kyle Kelley, Gladys Nalvarte, Andrew Osheroff, M. Pacer, Yuvi Panda, Fernando Perez, Benjamin Ragan-Kelley, and Carol Willing. 2018. Binder 2.0 - Reproducible, interactive, sharable environments for science at scale. *Proceedings of the 17th Python in Science Conference* 113–120.

Pržulj, N., D. G. Corneil, and I. Jurisica. 2004. Modeling interactome: scale-free or geometric? *Bioinformatics* 20(18):3508–3515. http://oup.prod.sis.lan/bioinformatics/article-pdf/20/18/3508/521683/bth436.pdf.

Raschka, Sebastian, Yuxi Liu, and Vahid Mirjalili. 2022. *Machine Learning with PyTorch and Scikit-Learn*. Packt Publishing.

Ravasz, E., A. L. Somera, D. A. Mongru, Z. N. Oltvai, and A.-L. Barabási. 2002. Hierarchical Organization of Modularity in Metabolic Networks. *Science* 297(5586):1551–1555.

Razick, Sabry, George Magklaras, and Ian M. Donaldson. 2008. iRefIndex: A consolidated protein interaction database with provenance. *BMC Bioinformatics* 9(1):405.

Reimand, Jüri, Ruth Isserlin, Veronique Voisin, Mike Kucera, Christian Tannus-Lopes, Asha Rostamianfar, Lina Wadi, Mona Meyer, Jeff Wong, Changjiang Xu, Daniele Merico, and Gary D. Bader. 2019. Pathway enrichment analysis and visualization of omics data using g:Profiler, GSEA, Cytoscape and EnrichmentMap. *Nature Protocols* 14(2):482–517.

Reuter, Jason A, Damek V Spacek, and Michael P Snyder. 2015. High-throughput sequencing technologies. *Molecular cell* 58(4):586–597.

Ritz, Anna, Christopher L. Poirel, Allison N. Tegge, Nicholas Sharp, Kelsey Simmons, Allison Powell, Shiv D. Kale, and T. M. Murali. 2016. Pathways on demand: automated reconstruction of human signaling networks. *npj Systems Biology and Applications* 2(1):1–9.

Rodchenkov, Igor, Ozgun Babur, Augustin Luna, Bulent Arman Aksoy, Jeffrey V Wong, Dylan Fong, Max Franz, Metin Can Siper, Manfred Cheung, Michael Wrana, Harsh Mistry, Logan Mosier, Jonah Dlin, Qizhi Wen, Caitlin O'Callaghan, Wanxin Li, Geoffrey Elder, Peter T Smith, Christian Dallago, Ethan Cerami, Benjamin Gross, Ugur Dogrusoz, Emek Demir, Gary D Bader, and Chris Sander. 2019. Pathway Commons 2019 Update: integration, analysis and exploration of pathway data. *Nucleic Acids Research* 48(D1):D489–D497. `https://academic.oup.com/nar/article-pdf/48/D1/D489/31697764/gkz946.pdf`.

Roushangar, Raeuf, and George I. Mias. 2018. ClassificaIO: machine learning for classification graphical user interface. *bioRxiv* 240184.

Schwartz, Michael P., Zhonggang Hou, Nicholas E. Propson, Jue Zhang, Collin J. Engstrom, Vitor Santos Costa, Peng Jiang, Bao Kim Nguyen, Jennifer M. Bolin, William Daly, Yu Wang, Ron Stewart, C. David Page, William L. Murphy, and James A. Thomson. 2015. Human pluripotent stem cell-derived neural constructs for predicting neural toxicity. *Proceedings of the National Academy of Sciences* 112(40):12516–12521.

Schwarz, Gideon. 1978. Estimating the Dimension of a Model. *The Annals of Statistics* 6(2): 461–464.

Scott, Jacob, Trey Ideker, Richard M. Karp, and Roded Sharan. 2006. Efficient Algorithms for Detecting Signaling Pathways in Protein Interaction Networks. *Journal of Computational Biology* 13(2):133–144.

Singh, Dinesh, Phillip G. Febbo, Kenneth Ross, Donald G. Jackson, Judith Manola, Christine Ladd, Pablo Tamayo, Andrew A. Renshaw, Anthony V. D'Amico, Jerome P. Richie, Eric S. Lander, Massimo Loda, Philip W. Kantoff, Todd R. Golub, and William R. Sellers. 2002. Gene expression correlates of clinical prostate cancer behavior. *Cancer Cell* 1(2):203–209.

Stokes, Jonathan M., Kevin Yang, Kyle Swanson, Wengong Jin, Andres Cubillos-Ruiz, Nina M. Donghia, Craig R. MacNair, Shawn French, Lindsey A. Carfrae, Zohar Bloom-Ackermann, Victoria M. Tran, Anush Chiappino-Pepe, Ahmed H. Badran, Ian W. Andrews, Emma J. Chory, George M. Church, Eric D. Brown, Tommi S. Jaakkola, Regina Barzilay, and James J. Collins. 2020. A Deep Learning Approach to Antibiotic Discovery. *Cell* 180(4):688–702.e13.

Sulmont, Elisabeth, Elizabeth Patitsas, and Jeremy R. Cooperstock. 2019a. Can You Teach Me To Machine Learn? In *Proceedings of the 50th ACM Technical Symposium on Computer Science Education*, 948–954. SIGCSE '19, New York, NY, USA: Association for Computing Machinery.

———. 2019b. What Is Hard about Teaching Machine Learning to Non-Majors? Insights from Classifying Instructors' Learning Goals. *ACM Transactions on Computing Education* 19(4):33:1–33:16.

Sverchkov, Yuriy, and Mark Craven. 2017. A review of active learning approaches to experimental design for uncovering biological networks. *PLOS Computational Biology* 13(6): 1–26.

Szklarczyk, Damian, Annika L. Gable, David Lyon, Alexander Junge, Stefan Wyder, Jaime Huerta-Cepas, Milan Simonovic, Nadezhda T. Doncheva, John H. Morris, Peer Bork, Lars J. Jensen, and Christian von Mering. 2019. STRING v11: protein-protein association networks with increased coverage, supporting functional discovery in genome-wide experimental datasets. *Nucleic Acids Research* 47(D1):D607–D613.

Teal, Tracy K., Karen A. Cranston, Hilmar Lapp, Ethan White, Greg Wilson, Karthik Ram, and Aleksandra Pawlik. 2015. Data Carpentry: Workshops to Increase Data Literacy for Researchers. *International Journal of Digital Curation* 10(1):135–143.

Thain, Douglas, Todd Tannenbaum, and Miron Livny. 2005. Distributed computing in practice: the condor experience. *Concurrency - Practice and Experience* 17(2-4):323–356.

The Gene Ontology Consortium. 2018. The Gene Ontology Resource: 20 years and still GOing strong. *Nucleic Acids Research* 47(D1):D330–D338. `http://oup.prod.sis.lan/nar/article-pdf/47/D1/D330/27437640/gky1055.pdf`.

Thul, Peter J., Lovisa Åkesson, Mikaela Wiking, Diana Mahdessian, Aikaterini Geladaki, Hammou Ait Blal, Tove Alm, Anna Asplund, Lars Björk, Lisa M. Breckels, Anna Bäckström, Frida Danielsson, Linn Fagerberg, Jenny Fall, Laurent Gatto, Christian Gnann, Sophia Hober, Martin Hjelmare, Fredric Johansson, Sunjae Lee, Cecilia Lindskog, Jan Mulder, Claire M. Mulvey, Peter Nilsson, Per Oksvold, Johan Rockberg, Rutger Schutten, Jochen M. Schwenk, Åsa Sivertsson, Evelina Sjöstedt, Marie Skogs, Charlotte Stadler, Devin P. Sullivan, Hanna Tegel, Casper Winsnes, Cheng Zhang, Martin Zwahlen, Adil Mardinoglu, Fredrik Pontén, Kalle von Feilitzen, Kathryn S. Lilley, Mathias Uhlén, and Emma Lundberg. 2017. A subcellular map of the human proteome. *Science* 356(6340):eaal3321.

Tian, David, Stephanie Wenlock, Mitra Kabir, George Tzotzos, Andrew J. Doig, and Kathryn E. Hentges. 2018. Identifying mouse developmental essential genes using machine learning. *Disease Models & Mechanisms* 11(12):dmm034546.

Tian, Yuanyuan, Richard C. McEachin, Carlos Santos, David J. States, and Jignesh M. Patel. 2007. SAGA: A subgraph matching tool for biological graphs. *Bioinformatics* 23(2):232–239.

Tripathi, Shashank, Marie O. Pohl, Yingyao Zhou, Ariel Rodriguez-Frandsen, Guojun Wang, David A. Stein, Hong M. Moulton, Paul DeJesus, Jianwei Che, Lubbertus C.F. Mulder, Emilio Yángüez, Dario Andenmatten, Lars Pache, Balaji Manicassamy, Randy A. Albrecht, Maria G. Gonzalez, Quy Nguyen, Abraham Brass, Stephen Elledge, Michael White, Sagi Shapira, Nir Hacohen, Alexander Karlas, Thomas F. Meyer, Michael Shales, Andre Gatorano, Jeffrey R. Johnson, Gwen Jang, Tasha Johnson, Erik Verschueren, Doug Sanders, Nevan Krogan, Megan Shaw, Renate König, Silke Stertz, Adolfo García-Sastre, and Sumit K. Chanda. 2015.

Meta- and orthogonal integration of influenza "omics" data defines a role for UBR4 in virus budding. *Cell Host & Microbe* 18(6):723–735.

Tuncbag, Nurcan, Alfredo Braunstein, Andrea Pagnani, Shao-Shan Carol Huang, Jennifer Chayes, Christian Borgs, Riccardo Zecchina, and Ernest Fraenkel. 2013. Simultaneous Reconstruction of Multiple Signaling Pathways via the Prize-Collecting Steiner Forest Problem. *Journal of Computational Biology* 20(2):124–136.

Tuncbag, Nurcan, Sara J. C. Gosline, Amanda Kedaigle, Anthony R. Soltis, Anthony Gitter, and Ernest Fraenkel. 2016. Network-Based Interpretation of Diverse High-Throughput Datasets through the Omics Integrator Software Package. *PLOS Computational Biology* 12(4): e1004879.

Veličković, Petar, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. *International Conference on Learning Representations*. Accepted as poster.

Vercio, Lucas Lo, Kimberly Amador, Jordan J. Bannister, Sebastian Crites, Alejandro Gutierrez, M. Ethan MacDonald, Jasmine Moore, Pauline Mouches, Deepthi Rajashekar, Serena Schimert, Nagesh Subbanna, Anup Tuladhar, Nanjia Wang, Matthias Wilms, Anthony Winder, and Nils D. Forkert. 2020. Supervised machine learning tools: A tutorial for clinicians. *Journal of Neural Engineering* 17(6):062001.

Veres, Daniel V, Dávid M Gyurkó, Benedek Thaler, Kristóf Z Szalay, Dávid Fazekas, Tamás Korcsmáros, and Peter Csermely. 2015. ComPPI: a cellular compartment-specific database for protein-protein interaction network analysis. *Nucleic acids research* 43(Database issue): D485–D493. Edition: 2014/10/27 Publisher: Oxford University Press.

Virtanen, Pauli, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* 17:261–272.

Walsh, Ian, Dmytro Fishman, Dario Garcia-Gasulla, Tiina Titma, Gianluca Pollastri, Jennifer Harrow, Fotis E. Psomopoulos, and Silvio C. E. Tosatto. 2021. DOME: Recommendations for supervised machine learning validation in biology. *Nature Methods* 18(10):1122–1127.

Wang, Ran, Yixiang Fang, and Xing Feng. 2019. Efficient Parallel Computing of Graph Edit Distance. In *2019 IEEE 35th International Conference on Data Engineering Workshops (ICDEW)*, 233–240.

Wang, Zijie J., Alex J. Walsh, Melissa C. Skala, and Anthony Gitter. 2020. Classifying T cell activity in autofluorescence intensity images with convolutional neural networks. *Journal of Biophotonics* 13(3):e201960050.

Weisfeiler, Boris, and Andrei Leman. 1968. The reduction of a graph to canonical form and the algebra which appears therein. *Nauchno-Technicheskaya Informatsia* (9).

Weiss, Yair. 2000. Correctness of local probability propagation in graphical models with loops. *Neural Computation* 12(1):1–41.

Welch, Lonnie, Fran Lewitter, Russell Schwartz, Cath Brooksbank, Predrag Radivojac, Bruno Gaeta, and Maria Victoria Schneider. 2014. Bioinformatics curriculum guidelines: Toward a definition of core competencies. *PLOS Computational Biology* 10(3):1–10.

Whalen, Sean, Jacob Schreiber, William S. Noble, and Katherine S. Pollard. 2021. Navigating the pitfalls of applying machine learning in genomics. *Nature Reviews Genetics* 1–13.

Wiggins, Grant, and Jay McTighe. 1998. What is backward design. *Understanding by design* 1: 7–19.

Wilson, Greg. 2016. Software Carpentry: lessons learned. *F1000Research* (3:62).

Wishart, David S, Carin Li, Ana Marcu, Hasan Badran, Allison Pon, Zachary Budinski, Jonas Patron, Debra Lipton, Xuan Cao, Eponine Oler, Krissa Li, Maïlys Paccoud, Chelsea Hong, An C Guo, Christopher Chan, William Wei, and Miguel Ramirez-Gaona. 2019. PathBank: a comprehensive pathway database for model organisms. *Nucleic Acids Research* 48(D1):D470–D478. `https://academic.oup.com/nar/article-pdf/48/D1/D470/31697631/gkz861.pdf`.

Wiwatwattana, Nuwee, and Anuj Kumar. 2005. Organelle DB: a cross-species database of protein localization and function. *Nucleic Acids Research* 33(suppl_1):D598–D604.

Xu, Haoyan, Ziheng Duan, Yueyang Wang, Jie Feng, Runjian Chen, Qianru Zhang, and Zhongbin Xu. 2021. Graph partitioning and graph neural network based hierarchical graph matching for graph similarity computation. *Neurocomputing* 439:348–362.

Xu, Keyulu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. 2019. How powerful are graph neural networks? In *International conference on learning representations*.

Yadati, Naganand, Madhav Nimishakavi, Prateek Yadav, Vikram Nitin, Anand Louis, and Partha Talukdar. 2019. Hypergcn: A new method for training graph convolutional networks on hypergraphs. In *Advances in neural information processing systems*, ed. H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, vol. 32. Curran Associates, Inc.

Yaveroğlu, Ömer Nebil, Noël Malod-Dognin, Darren Davis, Zoran Levnajic, Vuk Janjic, Rasa Karapandza, Aleksandar Stojmirovic, and Nataša Pržulj. 2014. Revealing the Hidden Language of Complex Networks. *Scientific Reports* 4:4547.

Yeger-Lotem, Esti, Laura Riva, Linhui Julie Su, Aaron D Gitler, Anil G Cashikar, Oliver D King, Pavan K Auluck, Melissa L Geddie, Julie S Valastyan, David R Karger, Susan Lindquist, and Ernest Fraenkel. 2009. Bridging high-throughput genetic and transcriptional data reveals cellular responses to alpha-synuclein toxicity. *Nature Genetics* 41(3):316–323.

Yip, Kevin Y., Chao Cheng, Nitin Bhardwaj, James B. Brown, Jing Leng, Anshul Kundaje, Joel Rozowsky, Ewan Birney, Peter Bickel, Michael Snyder, and Mark Gerstein. 2012. Classification of human genomic regions based on experimentally determined binding sites of more than 100 transcription-related factors. *Genome Biology* 13(9):R48.

Yosef, Nir, Einat Zalckvar, Assaf D. Rubinstein, Max Homilius, Nir Atias, Liram Vardi, Igor Berman, Hadas Zur, Adi Kimchi, Eytan Ruppin, and Roded Sharan. 2011. ANAT: A Tool for Constructing and Analyzing Functional Protein Networks. *Science Signaling* 4(196):pl1.

Youssef, Ibrahim, Jeffrey Law, and Anna Ritz. 2018. Integrating Protein Localization with Automated Signaling Pathway Reconstruction. In *2018 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*, 163–168.

Zhang, Song, Xuefeng Xia, Jincheng Shen, Yun Zhou, and Zhirong Sun. 2008. DBMLoc: A Database of proteins with multiple subcellular localizations. *BMC Bioinformatics* 9(1):127.

Zhu, Lu, Ralf Hofestädt, and Martin Ester. 2019. Tissue-Specific Subcellular Localization Prediction Using Multi-Label Markov Random Fields. *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 16(5):1471–1482.