Weighting Protein-Protein Interaction Networks

Gavin Gray

Master of Science by Research

Neuroinformatics

DTC in Neuroinformatics and Computational Neuroscience

School of Informatics

University of Edinburgh

2014

Abstract

This project focuses on the synaptic active zone network defined through experiments performed as part of the SYNSYS collaboration. Diverse data sources were used to appropriately weight the edges using a combination of a supervised binary classifier and a manually adjusted Naive Bayes model. A spectral modularity community detection algorithm was used to produce two sets of communities from the weighted and unweighted cases. The resulting communities were then compared using Normalised Mutual Information (NMI) and in terms of disease association. Communities detected and disease associations within these differed between the weighted and unweighted cases.

Acknowledgements

Firstly, I would like to acknowledge Colin Mclean as the main supervisor to this project and provided extensive input to both the implementation, planning and execution of all parts of the project. Finlay Maguire provided Biological expertise during this project, evaluating the theoretical basis behind many features and checking some of the code. As with the nature of this project, there were many databases and public resources that were used. These are described throughout the report and referenced appropriately. The project also made use of the Scikit-learn(Pedregosa et al., 2011) package throughout. Finally, this report was written using code developed by Chris Brown, Alex Shearn and Danilo Orlando.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Gavin Gray*)

Contents

List of Figures

Chapter 1

Introduction

To improve the insight gained on disease through the use of PPI networks we aim to incorporate as much of the available information on the existence of protein interactions as possible. This provides scope for data mining from a vast array of biological databases and requires the application of machine learning techniques to deal with very large datasets.

1.1 Motivation

It is estimated that disorders of the brain cost Europe €798 billion Euros in 2010(Olesen et al., 2012). Specifically, depression is estimated to cost Europe €91.9 billion in 2004 and schizophrenia along with associated psychotic disorders is estimated at €93.9 billion. There are many diseases with limited treatment options in the brain, and a deeper understanding of the brain is required to treat them.

Synaptic connections composed of the proteins of the cell are likely to be involved in these diseases(Chua et al., 2010; SynSys Project, 2014). However, the exact proteins or genes involved, and their organisation, in these diseases remain an important research topic. If it is possible to even gain slightly more information about associated proteins at the synapse it may be possible to develop new treatments(Li, Klemmer and Smit, 2010).

The aim of this project is to gather more information using various computational tools. This includes work from various areas such as Bioinformatics, Machine Learning and Software Development, thus making it a good learning exercise. It is a worthwhile investment in the fundamentals of constructing PPI

networks.

1.2 Outline

This project involved combining both direct and indirect data sources to create weighted edges in a PPI graph to affect the communities detected by a Community Detection algorithm. A flow chart describing this process and the elements involved is shown in figure 1.1. Pre-synaptic Immunoprecipitation, or pull-down, experiments performed as part of the SYNSYS project(SynSys Project, 2014) provide empirical validation of a set of interacting proteins for this project shown in figure 2.1 and described in table 2.1. Direct and indirect data refers to the interactions databases and other databases, respectively, as described in table 2.2.

The resulting weighted and unweighted PPI graphs were then separated into communities using a spectral based Community Detection algorithm as described in section 3.4.1. Comparison between these sets of communities was performed using NMI(Lancichinetti, Fortunato and Radicchi, 2008).

Using information on disease association of proteins it was possible to estimate the involvement of a community in a particular disease using a hypergeometric test as described in section 3.4.3. Specifically, this project focused on the two diseases schizophrenia and Alzheimer's. Our hypothesis was that the weighted communities would find more likely communities involved in these diseases than in the unweighted case. If the weighted case represents a real cluster of proteins this should be the case as seen in Soler-López et al. (2011).

The body of this report is split into three chapters. Chapter 2 describes the presynaptic dataset and data sources used to construct weighted and unweighted networks. The methods used to generate the weighted network and details on the community detection are described in chapter 3. Chapter 4 highlights candidate communities found in each case and analysis of these results. In addition, the execution of all commands required to reproduce the project were recorded in notebooks described in section B. This project was stored in a git repository and is publicly available in a repository described in Appendix A.

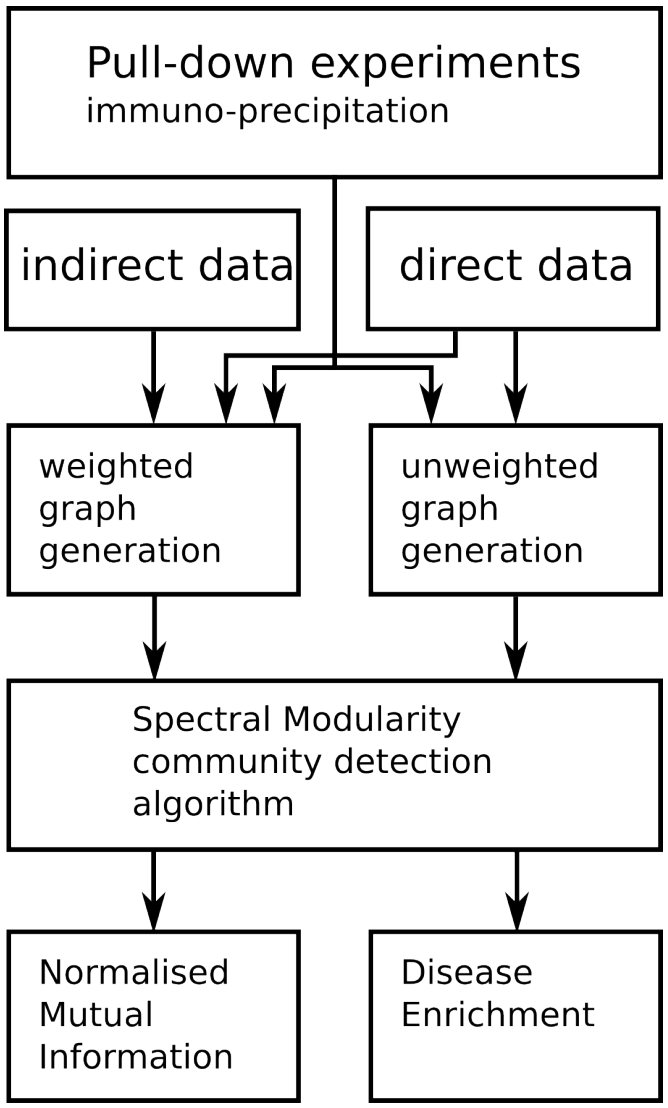


Figure 1.1: A flow chart describing how the elements of the project as a whole, as shown in the project proposal.

Conclusion

The following report covers the background information necessary to understand the methods of the project and the results which were obtained. It was approached as an opportunity to combine a variety of new tools in disease research on PPI networks.

Chapter 2

Background

This project involved the use of protein interaction prediction to build weighted PPI networks improve the performance of Community Detection on a PPI network for disease research. The following chapter describes the relationship of the disease to the proteins of the synapse. PPI networks, and their application to disease research is then described.

2.1 The synapse and protein interaction

Proteins consist of approximately 20% of cell mass in a typical eukaryote(Lodish et al., 2000). Each of these proteins are molecules which fit into machinery of a cell within the human body.Functions of these cells include almost all cellular functions; there are proteins capable of pumping ions, reshaping DNA and fluorescing(Alberts et al., 2008). A crude model of the cell is to map the interactions between these molecular machines to try to guess about the functioning of the cell. These models are PPI networks and can be useful for targeting proteins in disease research(B. Chen et al., 2013).

Synapses are the contacts between nerve cells where the vast majority of communication between nerve cells occurs, the only exceptions being through signalling molecules that can cross the cell membrane as shown in figure 2.1. There are two types of synapses in the nervous system, electrical and chemical(Kandel, Schwartz and Jessell, 2000). Electrical synapses form a simple electrical connection through an ionic substrate between two neurons. Chemical synapses are involved in a much more complex system of neurotransmitter release and reception. Synapses are therefore important to the functioning of the nervous system.

A problem with synapse function will likely cause large problems to the nervous system, so diseases of the nervous system are likely to involve problems with synapse function. Investigating the functioning of these proteins will help to explain the functioning of the synapse and hopefully provide insight into the diseases of the synapse(SynSys Project, 2014).

The proteins at the synapse drive synaptic communication, which in turn defines the functioning of the brain. As these proteins define the functioning of the brain any disorders which affect the brain are very likely to involve these proteins. Disorders which affect the brain are also very common and poorly understood, affecting one in three people in the developed world. Curing these diseases therefore may be possible through a greater understanding of the interactions of proteins at the synaptic level(SynSys Project, 2014; Chua et al., 2010).

Physical interaction between proteins can be inferred from a range of different experiments. Typical contemporary protein interaction networks rely on databases of confirmed interactions from a variety of experiments, for example in Kenley and Cho (2011) several well-known interaction databases were used, such as BioGRID(Stark et al., 2006). By forming a network from these individual interactions as edges and clustering this network Kenley and Cho (2011) were able to predict complexes and functional associations. As with functional association, through associating community members with disease it is possible to associate communities with diseases, as will be discussed in chapter 3.

Two papers, Ito et al. (2001) and Uetz et al. (2000), were able to leverage large volumes of recent interaction data and build interaction networks. These papers were able to make interesting discoveries about the network of interactions in yeast simply by investigating subnetworks in the network that was produced.

The aim of this project is to extend work in the field of protein interaction prediction (Qi, Bar-Joseph and Klein-Seetharaman, 2006; McDowall, Scott and Barton, 2009; Rodgers-Melnick, Culp and DiFazio, 2013; von Mering et al., 2005) to weighting protein interactions with a posterior probability through the use of varied data sources. Specifically, the interactions we are considering are those of the active zone network illustrated in figure 2.1 found as part of the SYNSYS project(SynSys Project, 2014). This data forms a set of proteins and a prepared unweighted list of protein interactions summarised in table 2.1. These proteins and their interactions were found through immuno-precipitation, or pull-down,

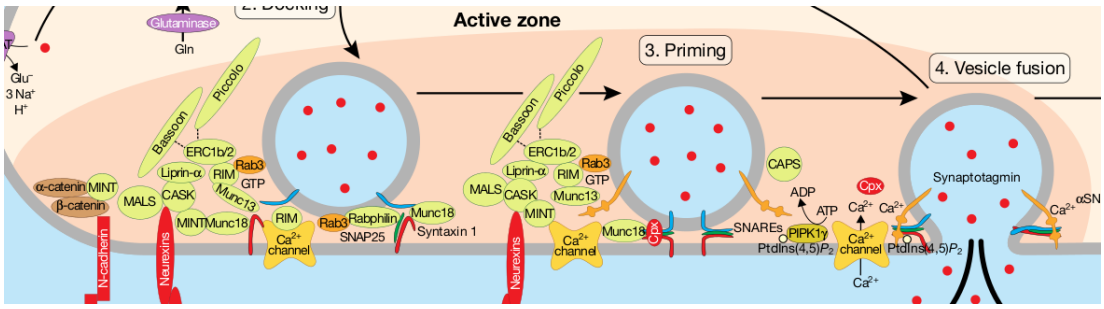


Figure 2.1: An illustration of the proteins identified to be involved in the active zone network(Chua et al., 2010).

Number of		
baits	preys	interactions
24	1548	9372

Table 2.1: A table summarising the results of the pull-down experiments performed as part of the SYNSYS project(SynSys Project, 2014), and the active zone network defined using them, used in this project.

experiments in the mouse hippocampus focusing on the pre-synapse. In these experiments a set of bait proteins are selected and used to attract a set of prey proteins, the interaction between bait and prey being the interactions detected. The exact set of interactions used in the unweighted network used was prepared prior to this project using additional resources: HIPPIE(Schaefer et al., 2012), InterologWalk(Gallone et al., 2011), BiGRID(Stark et al., 2006), CCSB(Yu et al., 2008), HPRD(Baolin and Bo, 2007), IntAct(Hermjakob et al., 2004) and MDC(Futschik, Chaurasia and Herzel, 2007).

2.2 Protein complexes and community detection

As mentioned in the previous section it is possible to analyse PPI networks to detect protein complexes and functional groups. This has recently been achieved through use of Community Detection(B. Chen et al., 2013; Wang et al., 2010), which uses various methods to find community structure in graphs.

Community structure is described as a characteristic of graphs which have many connections within sub-groups but few connections outside that group(Newman, 2012). Unfortunately, this description is not specific on exact measures for a

graph to have community structure. Community detection algorithms are simply tested on graphs that are agreed to exhibit community structure with the aim of finding the pre-defined communities.

Two important approaches to the problem of Community Detection are traditional hierarchical methods and more recent optimization based methods(Newman, 2012). Hierarchical methods were developed in the field of sociology and involves grading nodes by how highly connected they are in the network and then using this value to group nodes into communities. Optimization based methods involves, such as spectral modularity, involves grading edges and removing them iteratively to reveal the community structure.

2.3 Protein-protein interaction prediction

Protein interaction prediction was developed to solve the problem of incomplete and unreliable interaction data by combining both direct and indirect information(Qi, 2008). Direct information are the result of experiments, such as yeast two-hybrid, intended to directly find protein-protein interactions. Indirect information includes biological data that was not gathered directly to find interactions, such as gene expression data.

To predict a protein interaction we need to have a value or sequence of values from which to make our guess as to the existence of an interaction. For each interaction this set of values are known as features. The bulk of this project, described in chapter 3, involved obtaining these values for every feature necessary to train the classifier and classify the interactions of the synaptic network.

The classifier, or model, is a machine learning algorithm that can learn from a labelled training set how to sort these vectors of features into the appropriate category. However, these algorithms cannot make predictions unless the training data is informative. Also, the training data must be an accurate representation of the case the algorithm is planned to be applied to.

Completing the interactome of a given organism from incomplete data is a major goal for some works in the protein interaction field, such as Rodgers-Melnick, Culp and DiFazio (2013). The goal in this project is to appropriately weight interactions in a PPI network to improve the performance of a Community Detection algorithm.

Weakly interacting proteins will have a lower confidence in their interacting

Data source type	Examples
Primary interaction databases	Database of Interacting Proteins(Xenarios et al., 2002) (DIP)(Xenarios et al., 2002) Human Integrated Protein-Protein Interaction rEference(Schaefer et al., 2012) (HIPPIE)(Schaefer et al., 2012) BioGRID(Stark et al., 2006) iRefIndex(Razick, Magklaras and Donaldson, 2008)
Pull-down experiment results	Described in table 2.1
Associated features	Features derived from Gene Ontology(Ashburner et al., 2000), described in section 4.1.1 Those used in Rodgers-Melnick, Culp and DiFazio (2013), described in section 4.1.2
Other PPI prediction resources	Search Tool for the Retrieval of Interacting Genes/Proteins (STRING)(von Mering et al., 2005) InterologWalk(Gallone et al., 2011)

Table 2.2: A table summarising the different sources of data used in the course of the project.

at all, as it will have been observed less frequently. Therefore, by weighting the interactions in a PPI network according to our confidence we can also make the PPI network reflect more closely the true interactions existing in vivo.

2.4 Data sources and networks

Many different data sources were considered for inclusion in this project. These different data sources fall into categories described in table 2.2, while the results of the SYNSYS pulldown experiments can be found in table 2.1.

The indirect sources of data were chosen based on usage in the literature, such as in the case of Gene Ontology(Qi, Bar-Joseph and Klein-Seetharaman, 2006).

Many of those considered were found to difficult to use, due to poor accesibility, such as in the case of gene expression, or differences in naming conventions. Direct data sources were listed by investigating all of the available databases which could be of use and choosing from these.

Conclusion

PPI networks are an important tool in the investigation of cell and synaptic function. Protein interaction prediction and various interaction databases provide a way to construct these networks. The next chapter discusses the construction of weighted networks making use of the interaction network described in table 2.1 and the data ssources described in table 2.2.

Chapter 3

Methodology

This chapter describes the use of two probabilistic models to construct a weighted PPI network. The first of these is a supervised binary classifier, implemented from a selection of either logistic regression, Support Vector Machine (SVM) or random forest. As the project progressed this method was unsuccessful and a second conditionally independent probabilistic model was used to update on prior knowledge and produce the final weightings. Feature extraction was the predominant component in this project and involved turning the various data sources involved into usable features in a machine learning framework. The chosen Community Detection method, the fast and efficient Spectral Modularity algorithm, is also described.

3.1 Feature vectors

The set of biologically relevant information associated with a given protein interaction is referred to as a feature vector and the constituent elements features. The process of retrieving these from biological databases is referred to as feature extraction. This is a task which commonly involved mapping between non-standardised protein identifiers and indexing large tables of biological data automatically.

In supervised learning problems we wish to learn a mapping between input variables and output variables given a training set. Defining this training set rigorously, it consists of input variables $\boldsymbol{x}$, which are typically vectors of values known as features. The output variables in a classification problems are a set of labels(Murphy, 2012, p. 2) which are denoted as y. In the case of binary classific-

ation these are simply either 0 or 1. Given N training vectors, $\boldsymbol{x}_i$, corresponding to N interactions in the network and training labels y_i we can define our training set $\mathcal{D}$ for N data points as:

$$\mathcal{D} = ((\boldsymbol{x}_i, y_i))_{i=1}^{N} \tag{3.1}$$

Our problem involves taking various types of biological data, such as entries from biological databases indicating that proteins are involved in the same part of a cell and using these as features. The training labels are either an interaction (a one) or a non-interaction (a zero). Interactions are taken to be any interactions in the iRefIndex(Razick, Magklaras and Donaldson, 2008) consolidated database. This database was chosen for reasons described in section 3.2. Non-interactions are random binary combinations of Entrez protein IDs, which is a method applied in other works(Qi, Bar-Joseph and Klein-Seetharaman, 2006) to create negative training examples. These are also checked against the iRefIndex database to ensure they are not accidentally known interactions. The full training set contained 188,833 true interactions and 997,760 non-interactions, but only a sub-sample of this was used for training to ensure a ratio of 600 non-interactions per true interaction.

What we would like to estimate is the posterior probability of an interaction existing given a new feature vector after training our classifier. For any model $\mathcal{H}$ and a new feature vector $\boldsymbol{x}^*$ we can express this using Bayes theorem:

$$p(y^* = 1|\boldsymbol{x}^*, \mathcal{D}, \mathcal{H}) = \frac{p(\boldsymbol{x}^*|y^* = 1, \mathcal{D}, \mathcal{H})p(y^* = 1|\mathcal{D}, \mathcal{H})}{\sum y^* p(\boldsymbol{x}^*|y^*, \mathcal{D}, \mathcal{H})} \tag{3.2}$$

Where in equation 3.2 posterior probability of an interaction $y^* = 1$ conditioned on a novel feature vector $\boldsymbol{x}^*$ is $p(y^* = 1|\boldsymbol{x}^*, \mathcal{D}, \mathcal{H})$ and the prior is $p(y^* = 1|\mathcal{D}, \mathcal{H})$.

We explicitly apply a prior to the probability of interaction based on the expected ratio of interactions to non-interactions stated in (Qi, Bar-Joseph and Klein-Seetharaman, 2006). This is described in more detail in section 3.3.

3.1.1 Protein identifier mapping

Mapping from one protein identifier to another became a significant problem in this project. Unfortunately, most Biological databases maintain their own

indexing method to identify different genes and proteins. New data sources being integrated into this project would often be using a different identification scheme to the NCBI Entrez GeneID originally chosen to use in the PPI network.

Genes are uniquely defined by their amino acid sequence, which is a long series of letters. However, for the sake of posterity databases containing information about genes typically apply an identifier for each gene that is much shorter than the sequence and can encode other information about the gene, typically referred to as an accession. The Entrez GeneID identifier is relatively simple, just consisting of a number generated when the gene was added to the database(Maglott et al., 2007).

Other popular schemes include the Ensembl identifier from the Ensembl database(Hubbard et al., 2002), Uniprot identifiers from the Uniprot database(The UniProt Consortium, 2008) and even those used only for specific databases such as DIP identifiers(Xenarios et al., 2002). Mapping between these different identifiers is difficult as each identifier may map to none or many in another database. The reason this happens is due to isoforms of different proteins; different amino acid sequences can code for a protein with the same name. This can be due to, for example, alternative splicing(Black, 2003) or simple submission errors(Zeeberg et al., 2004).

Various tools exist to map from one protein identifier to another: Ensembl's BioMart(Smedley et al., 2009) is a versatile web-based tool, for example. In this project, simple conversion tables from NCBI's Gene(Maglott et al., 2007) ftp server were primarily used. Another tool used was the Uniprot(The UniProt Consortium, 2008) online service web service.

Unfortunately, using any of these services there will be a number of identifiers which cannot be converted and many IDs mapping to the same Entrez identifier as different protein isoforms are picked up. One way to avoid this problem is to only refer to a single canonical form of any given protein and find this protein in other databases through its amino acid sequence. This ensures that when referring to an interaction between two identifiers the interaction is always simply between two proteins.

Otherwise, as in this project, the interaction is detected between two Entrez identifiers; which corresponds to an interaction between genes - possibly only a single interaction between combinations of the isoforms of each gene. Unfortunately, this means that this project is only concerned with gene interaction

prediction until the Entrez identifiers have been carefully canonicalized. This is not really a problem, as we are only aiming to provide a weighting to a graph, rather than provide an accurate prediction of interaction between proteins.

A solution to this problem is provided by the iRefIndex(Razick, Magklaras and Donaldson, 2008) database, which combines many databases and stores canonicalized entries. Using this database, it would be possible to ensure that the proteins used in a future project would be reliable canonical proteins. Additionally, each protein of interest should ideally be stored with reference to its sequence in, for example, FASTA format.

3.2 Supervised model

To begin supervised learning, the model required a set of accepted true interactions. The database of interacting proteins (DIP) is a database of interactions proven by small-scale experiments(Xenarios et al., 2002). Each interaction added is hand curated so it was expected as a reliable training set. This database was also used as a training set in Qi, Bar-Joseph and Klein-Seetharaman (2006), who used the same supervised training method.

Unfortunately, problems were found with DIP as a training set. Features derived from interaction other databases, such as STRING, would win out in importance versus all indirect features as shown in figure 3.1. HIPPIE was also tested as a training set, but this lead to the same problem.

It was decided that only indirect features, as described in section 2.4 should be used in the trained supervised classifier and direct evidence integrated into the final weightings in an explicit Bayesian method described in section 3.3; the results of which are described in section 4.3.5. Once these databases were removed the performance of the classifier was drastically lower. However, all of the available features had more closely distributed importances in the final classifier, as shown in section 4.3.4.

To train the final classifier the iRefIndex(Razick, Magklaras and Donaldson, 2008) database was used to find real interactions due to its canonicalisation of identifiers and the number of databases it includes. Again, non-interactions were generated as random combinations of Entrez Gene IDs. These were also checked against the positive interactions to ensure known interactions were not present.

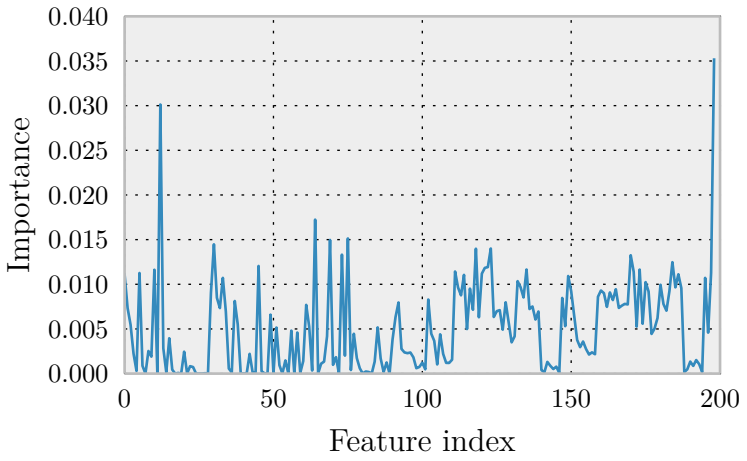


Figure 3.1: An example of an unbalanced set of feature importances plotted after fitting a Random Forest classifier to a dataset containing interaction database derived features. Feature indexes are not as described in table 3.1. Importances are dimensionless and show that only a small number of features are important to the classifier.

3.2.1 Labeling feature vectors

PPI prediction features are a set of values for each interaction considered, if it is a real or non-interaction. This arrangement is illustrated in figure 3.2. The features used are described by index in table 3.1.

To keep track of the various different data sources and assemble the features into vectors to be used in the classifiers a dedicated piece of code was required. The code developed is written as a python module called ocbio.extract. Usage and development notes for this program are referenced in Appendix B.3.

3.2.2 Model Selection

The classification problem we are solving is different in that we are really trying to obtain a realistic weighting of interactions for use in a PPI network. Classification is normally concerned about picking a decision threshold to classify examples into categories. However, in this case the output of our process is the posterior probability of the model $\mathcal{H}$ as in equation 3.2. Where $\mathcal{H}$ is any of the following classification models, such as logistic regression.

The posterior probability was constructed using the Python package Scikit-learn(Pedregosa et al., 2011). Each classifier implemented in this package has a

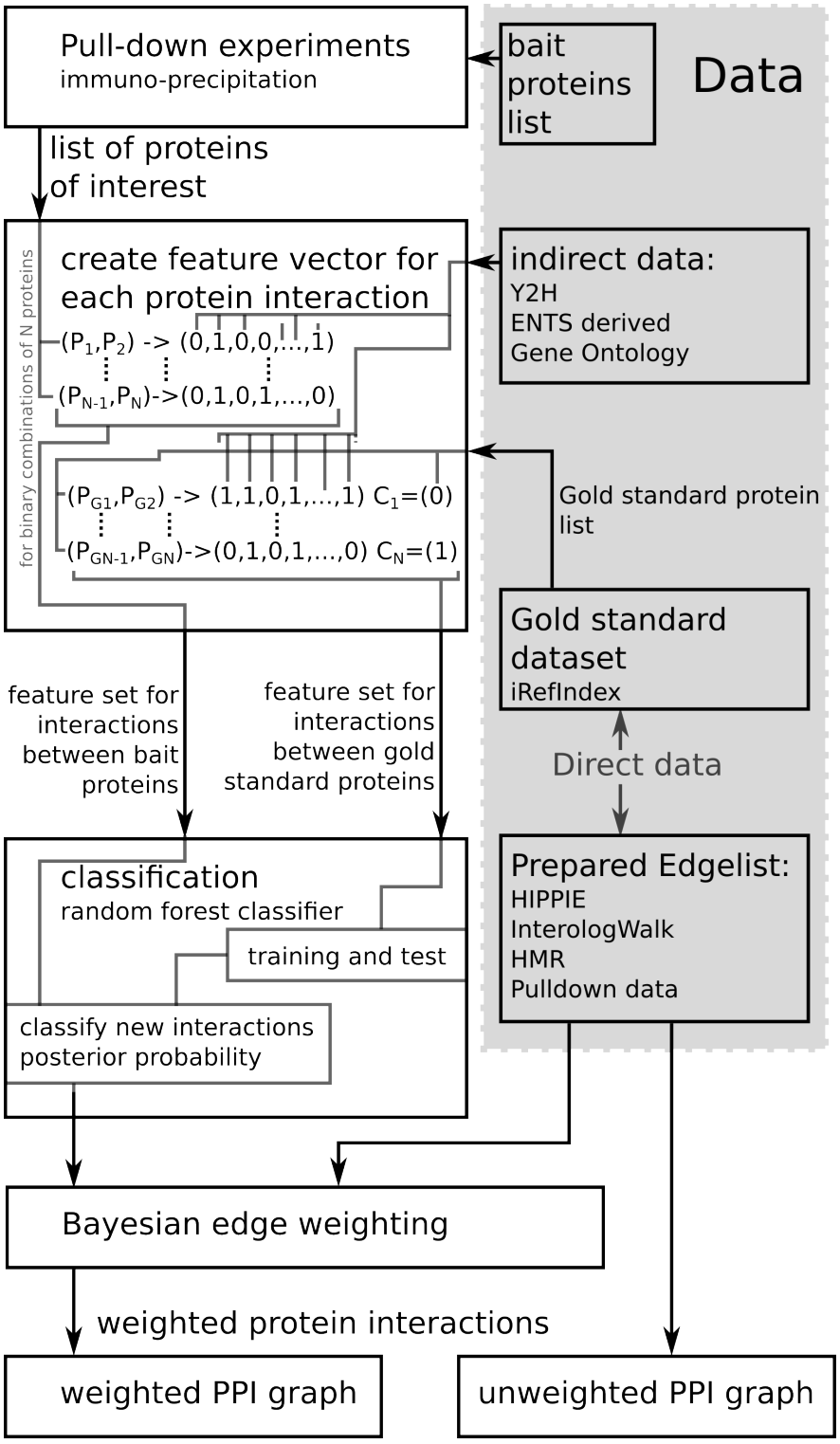


Figure 3.2: A diagram showing the structure of feature vectors and their relationship to the project as a whole. This figure is based on a similar flow chart shown in the project proposal.

Feature	Indices	Description
Gene Ontology	1-90	Described in section 4.1.1, with individual indices described in Appendix C.1.
Y2H	91	Y2H experimentally derived feature
ENTS derived	92-198	Features used by the ENTS classifier, described by index in supplementary table for Rodgers-Melnick, Culp and DiFazio (2013).
ENTS summary	199	Prediction result of the classifier in Rodgers-Melnick, Culp and DiFazio (2013).

Table 3.1: Each feature used in the final classifier is described by index.

similar interface allowing modular code to be written. In addition, this package is actively developed with all the required classifiers having efficient implementations.

There were three classifiers chosen: a logistic regression model, a random forest model and a SVM.

Each classifier used requires tuning of hyper-parameters which will affect its overall performance. These hyper-parameters are described in table 3.2 for each of the models used in the project. The optimal values for these found can be found in table 4.2.

Logistic Regression

Logistic regression is a linear model being used for binary classification. It is equivalent to a linear regression model transformed through a sigmoid function(Murphy, 2012, p. 376), denoted here by σ:

$$p(c = 1|\boldsymbol{x}) = \sigma(b + \boldsymbol{x}^T * \boldsymbol{w}) \tag{3.3}$$

Where in equation 3.3 $\boldsymbol{x}$ is the vector of features and c is the class label - using 1 for a real interaction and 0 for a non-interaction. The weights and biases are the parameters of this model, expressed in the above equation as b and $\boldsymbol{w}$, respectively.

Classifier	Hyper-parameters	Description
Logistic Regression	C	Inverse of regularisation strength, the α parameter in section 3.2.2.
Support Vector Machine	kernel	The kernel used. Three options: linear, Radial Basis Function (RBF) and polynomial.
	Gamma(γ)	Kernel coefficient.
	C	As with Logistic Regression, a penalty parameter.
Random Forest	N estimators	Number of trees used in the forest.
	Max features	Number of features considered when looking for splits, part of the problem of finding the best decision at each node described in section 3.2.2.
Extremely Randomized Trees	N estimators	As with Random Forest.
	Max features	As with Random Forest.

Table 3.2: Summary of the hyper-parameters described in the Scikit-learn (Pedregosa et al., 2011) documentation to be tuned for each of the models considered.

This divides the points, feature vectors, in the dataset by a hyperplane, classifying the points on each side of the hyperplane into different classes. For data that is linearly separable, this produces a classifier that will make no mistakes on the test data. Unfortunately, the data we are working with is not linearly separable as shown by visualization in section 4.2.

To find the parameters in this model the log likelihood must be maximised; corresponding to the maximum likelihood solution. The log likelihood, for N feature vectors, is given by:

$$L(\boldsymbol{w}, b) = \sum_{n=1}^{N} c^n \log \sigma(b + \boldsymbol{w}^T \boldsymbol{x}^n) + (1 - c^n) \log(1 - \sigma(b + \boldsymbol{w}^T \boldsymbol{x}^n)) \tag{3.4}$$

Regularisation of the weights represents a prior belief that the weights should not increase without bound. In a case where the data is linearly separable and where regularisation is not applied the weights will increase without bound to produce extremely confident classifications(Barber, 2012, p. 381). To stop this from happening we apply a penalty term, $\frac{1}{C}$, to the size of the weights:

$$L'(\boldsymbol{w}, b) = L(\boldsymbol{w}, b) - \frac{1}{C}\boldsymbol{w}^T\boldsymbol{w} \tag{3.5}$$

Tuning this hyper-parameter is the goal of a grid search, which is discussed in section 3.2.3, and the optimal value found in this case was 0.0215 as shown in table 4.2.

Support Vector Machines

Logistic regression can be generalised to apply kernel functions to the input features to obtain better classifications. Support Vector Machines exploit this while also applying a different objective function intended to avoid overfitting(Murphy, 2012, p. 383). The objective in placing the hyperplane for a SVM is "maximum margin" in that it attempts to maintain the same distance from the closest opposing class points.

These are often successful classifiers in practice. Applications include text categorisation, hand-written character recognition, image classification and biosequences analysis(Cristianini and Shawe-Taylor, 2000).

The hyper-parameters for a SVM control the kernels, along with the regularisation parameter as described for logistic regression in section 3.2.2. Two of the three hyper-parameters which can be tuned during a grid search operation are the degree of polynomial kernels if chosen and the gamma coefficient of the kernels. In table 4.2 it can be seen that the best performance was achieved with a RBF kernel, C of 10.0 and gamma of 10.0.

Random Forest

A random forest is a another of these models. It operates as a combination of many decision trees. Decision trees are intuitively simple in that it consists of a series of comparisons arranged in a tree as shown in figure 3.3. Each feature is tested at a node to come to a final conclusion about the state of the interaction.

In this way decision trees are both simple and tractable methods for using a feature vector to classify interactions as real or false and there are many automated ways to generate effective decision trees. The problem in the design of a decision tree is which comparison to choose at each node, which is beyond the scope of this report, a full description can be found in Murphy (2012, p. 544).

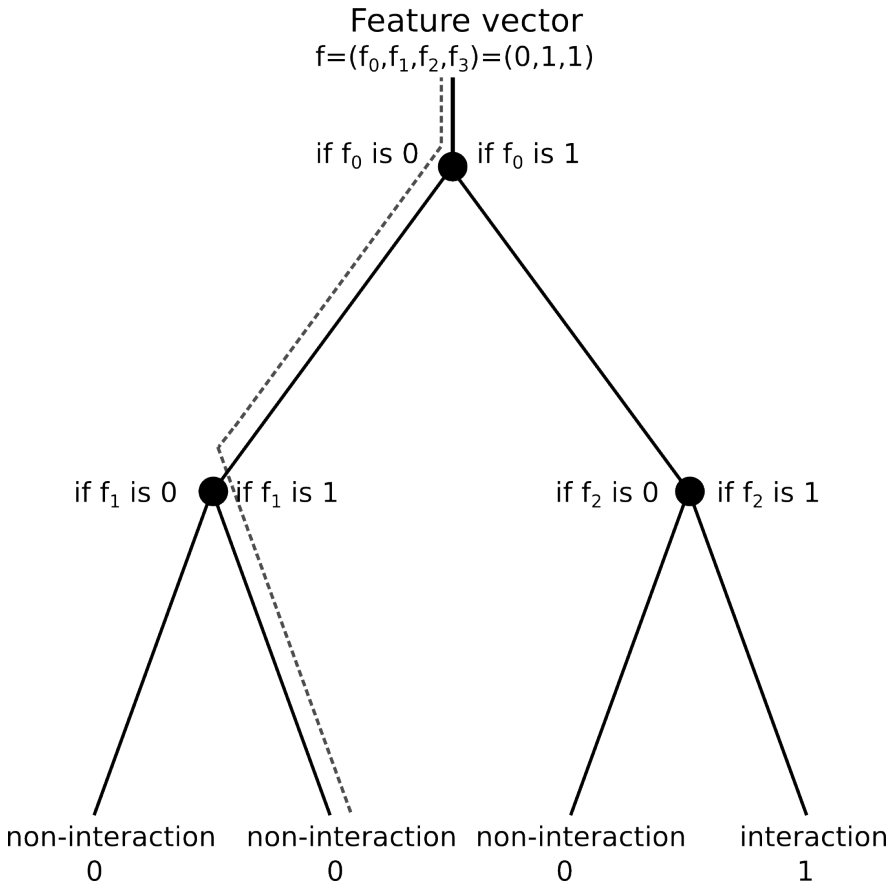


Figure 3.3: A simple example decision tree, illustrating the process of sequential, dependent decisions. The example feature vector f follows a path through the tree indicated by a dashed line. Based on an image obtained through Wikimedia(Wikimedia Commons, 2013).

Other advantages of decision trees include the ability to handle a mixture of discrete and continuous inputs, automatic variable selection, scaling to large datasets readily and the ability to handle missing inputs, with modification.

Unfortunately, despite the strengths of decision trees there are still problems with stability: small changes in the input data can produce large changes in the output(Murphy, 2012, p. 550). The random forest algorithm addresses this problem by providing redundancy; multiple trees are grown and their results averaged. For M trees trained on different subsets of the data(Murphy, 2012, p. 551) we obtain:

$$f(x) = \sum_{m=1}^{M} \frac{1}{M} f_m(x) \tag{3.6}$$

Where $f_m(x)$ is mth tree. This is simply averaging the results of the trees and is known as bagging.

With the ability to work on large datasets and mixing continuous and discrete data these types of classifiers would appear to already be well suited to this problem. This is what has been observed in the literature, with these classifiers achieving the best performance despite different types of biological data being used(Qi, Bar-Joseph and Klein-Seetharaman, 2006; Rodgers-Melnick, Culp and DiFazio, 2013). Due to these reports in the literature it appeared that this classifier would be the best choice for our protein interaction prediction task.

The two hyper-parameters varied in our search were the number of trees, or estimators and the max features described in table 3.2. In table 4.2 it can be seen that 44 estimators and a max feature value of 25 were found to be optimal.

Other options

Other options considered for our classification problem, but not included in the project due to time constraints included Feedforward neural networks, Naive Bayes and Beta regression. Naive Bayes in particular would have required modifying the code from Scikit-learn to deal with data from multiple different distributions or implementing Weka's solution of kernel density estimated distributions for each different feature(John and Langley, 1995) due to current Python implementations being limited to either the Gaussian or Bernoulli case.

3.2.3 Model testing

Various measures, such as accuracy, ROC and precision-recall, were applied to each model to estimate their performance in different ways. These tests included simple accuracy measures applied over learning curve and grid searches along with plotting ROC and precision-recall curves. Grid searches of parameter values were used to find optimal hyper-parameters in all three models.

Cross-validation is the practice of repetitively splitting the data into training and test sets and was applied to tests during hyper-parameter searches(Witten, Frank and Hall, 2011, p. 152). It was also applied when plotting learning curves to get a statistical estimate of the reliability of the metrics applied to the classifier. As the training set is approximately 10^6 samples, these were applied as random sub-samples of the full data set, but stratified to maintain the same proportion of non-interactions to interactions as in the full training set. This is important as it must reflect the expected ratio for real protein interactions to non-interactions. The use of pipelines and learning curves in this process is described in Appendix B.

A grid search is a cross-validated test measuring accuracy testing the classifier with a variety of different hyper-parameters. Classifiers, such as a Logistic Regression model, have some number of hyper-parameters. A Logistic Regression model, for example, has a single hyper-parameter so that the grid search for this model simply involves varying this parameter to obtain the optimum performance on the test set.

Before classification could be performed the missing data in the feature vectors had to be imputed. This was performed by filling the missing values with the mean value of that feature. This is a common technique that is applied if it is likely the data is missing at random. In this case the data that is missing is due to mismatches in protein identifier mapping dictionaries, which is likely random and independent of the interaction prediction task.

Accuracy

The accuracy value plotted in the learning curve and used in grid searches of parameters values is simply the proportional of correctly classified instances in a training or validation set. Typically, the protein interaction prediction problem is sparse, in that there are very few interactions for the large number of non-

interactions. This is reflected in the accuracy in that a classifier that simply always predicts zero can still achieve a very high accuracy. Using this accuracy value is therefore problematic and this requires the other measures employed, such as ROC and precision-recall AUC values.

ROC curve

A Receiver Operating Characteristic, or ROC, curve plots the variation in true positive to false positive rate as the threshold of classification is varied. This makes it useful as an illustration of the tradeoff possible with this particular classifier. The Area Under Curve, or AUC, value is the area under this line. A higher AUC value corresponds to a better classifier, although there is some controversy surrounding this(Hanczar et al., 2010). These concerns center on the problems of small sample sizes, which are not the case in this project.

Precision-recall curve

A precision-recall curve plots the precision of a classifier against its recall as the threshold of classification is varied. The precision is defined as the number of true positive results over the number of total positive results. Recall is defined as the number of true positive results against the number of available positive examples The area under the curve is used to gauge the classifier's effectiveness.

3.3 Conditionally independent probabilistic model

Weighting interactions using the posterior distribution of a probabilistic model requires that the model accurately represents beliefs about the system being modelled. Supervised classification requires that true interactions are known in order to find the parameters of the model. Unfortunately, in this problem we cannot know for certain whether an interaction is real or false. Therefore, when fitting a supervised model we only obtain, at best, an accurate predictor for the interaction database used to fit the model as shown in figure 3.1.

As the PPI network edges we plan to weight have already been defined through combining different interaction databases we already have a strong prior on the existence of these interactions.

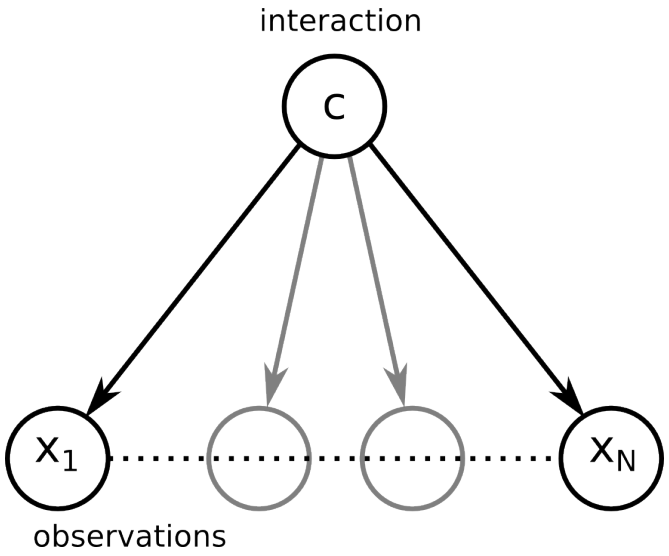


Figure 3.4: A belief network illustrating the Naive Bayes model, equivalent to that used for inference when weighting the interactions of the PPI network.

Using the classifier on its own can then, at best, reproduce one of the databases used to create this network and many of the interactions will be incorrectly weighted much lower than expected. The solution is to treat both the result of the classifier and the edgelist inclusion as observable events dependent on the latent "interaction" variable. Along with these variables, we can also include other protein interaction prediction resources, such as STRING(von Mering et al., 2005).

The model we have chosen to update our prior belief using assumes conditional independence of the observable variables given the hidden interaction variable. This equates to a Naive Bayes model with a belief network as shown in figure 3.4. An advantage of this model is the simplicity of its analysis. The disadvantage is that our variables could be dependent, as the classifier is trained on some of the same interaction databases that the HIPPIE database is composed of.

The class label, interaction, is not observed in this model so we cannot solve to find the parameters. To define the model, the only way to proceed is to manually define them by conservatively estimating the conditional true positive and false positive rates of the Bernoulli distributions. Continuous distributions, such as the classifier predictions or prediction databases, were estimated in a supervised way using a sample of the training set used to train the supervised classification model.

3.4 Comparison of weighted and unweighted PPI networks

It is hoped that a weighted network will provide new insight into the interactions of proteins in the active zone described in table 2.1. For this reason, comparing the unweighted and weighted cases of the graph produced is a major goal of this project. The following sections describe this process and the measures applied to both networks.

3.4.1 Community detection

Three algorithms were identified for use in this project for community detection, the first two are described in Newman and Girvan (2004). Geodesic edge betweenness and random edge betweenness are based on betweenness measures to partition the graph, making them optimisation approaches. The third, spectral modularity, was the chosen algorithm in this project.

The original advantages of spectral modularity techniques were compatible results on generated test data in less time than competing methods in Newman (2006). In a recent paper, this method was compared favourably to other methods in terms of CPU time(Mclean et al., 2014).

3.4.2 Normalised Mutual Information

Mutual information intuitively is the reduction in uncertainty about one random variable by observing another Defined in terms of entropy it is(Mackay, 2003):

$$I(X;Y) = H(X) - H(X|Y) \tag{3.7}$$

Where $H(X)$ is the entropy of the random variable X and $H(X|Y)$ is the conditional entropy of X given Y.

In the case of the function we are using to perform this from Scikit-learn the mutual information is normalised by $\sqrt{H(X) \times H(Y)}$(Pedregosa et al., 2011). This produces a value between zero and one which reflects the redundancy of the distributions - 1.0 being exactly the same and 0.0 being independent.

3.4.3 Disease Enrichment

By linking the proteins in a cluster to known disease annotations it is possible to estimate the likelihood that a given community is involved in a particular disease. Using a list of known genes involved in the disease of interest the test finds if the number of these genes in the communities detected is significant. The code being used produces uses a hypergeometric test to find this likelihood and returns a p-value.

In the case of this project, due to the aims of the SYNSYS(SynSys Project, 2014) project and for simplicity, only two diseases were investigated: Schizophrenia and Alzheimer's. The genes associated with schizophrenia and Alzheimer's used are part of the internal code used.

Conclusion

This chapter described how the weighted network was constructed using both a supervised probabilistic model and another conditionally independent model. After the failings of the supervised method were apparent, this combined method was the best solution found to the prediction problem. The next chapter will discuss the results of both of these techniques and the analysis of the weighted and unweighted networks produced.

Chapter 4

Results

As the project progressed the intended approach was found to be flawed and some changes had to be made. The first of these was the change from a DIP-based training set to HIPPIE-based training set. After the classification had been performed the use of a supervised classifier with this training set was found to produce poor results for weighting interactions. Using all of the data available it was possible to run a simple Bayesian alternative to continue weighting the interactions.

4.1 PPI feature vectors

27 features were identified for extraction and these are described in Appendix C. Of these, only 8 were successfully processed into a usable form.

Only 3 were used in the final supervised classifier, which neglected features directly derived from interaction databases. The features used to train the final classifier are shown in table 4.1. Yeast two-hybrid is an internal database of Yeast two-hybrid results used directly. A brief description of the remaining features is given below.

4.1.1 Gene Ontology

The Gene Ontology(Ashburner et al., 2000) is a resource of annotations for genes to indicate various characteristics in a hierarchical manner, such as cellular localisation or function. This resource has been used in past papers(Qi, Bar-Joseph and Klein-Seetharaman, 2006) and in databases such as STRING(von Mering et al., 2005) to predict protein interactions. Intuitively, it can be used to detect

Feature	Size	Type	Coverage on training set	Coverage on active zone network
Gene Ontology	90	Binary categorical	100.0%	100.0%
Yeast two-hybrid	1	Numerical	100.0%	100.0%
ENTS derived	107	Numerical	38.39%	42.74%

Table 4.1: A table summarising the components of the feature vectors used in the final classifier.

when, for example, two proteins are localised in the same area of the cell - as this would increase the probability that these two proteins interact. Details on exactly how this feature was generated can be found in the notebook reference in Appendix C.1.

4.1.2 Features derived from ENTS

107 features were obtained through analysis and modification of the bundled code and data downloaded from the website of Rodgers-Melnick, Culp and DiFazio (2013). In turn, most of these features were generated through the Multiloc2 program of Blum, Briesemeister and Kohlbacher (2009). The remaining features are pairwise combinations of conserved protein domains, which are conserved "modules" of proteins described in Janin and Chothia (1985).

4.2 Data visualisation

The feature vectors corresponding to interactions or non-interactions are plotted in the following section. Two techniques are used to plot these. The first of these is to reduce the dimensionality of the vector down to a size that can be plotted. The second was to plot all dimensions simultaneously using a parallel-coordinate plot.

For all graphs, two cases were investigated: in proportion and out of proportion. In proportion refers to the case where the proportion of interactions to non-interactions is correct; specifically, 1 interaction to 600 non-interactions. Out of proportion maintains the same number of interactions to non-interactions. This is important as it is often easier to separated the data when the classes are equally split due to the sheer number of non-interactions.

Reducing dimensionality

If the data were two dimensional we would like to plot it as a scatter plot and see if there was a clear grouping of the points. This would indicate that a classification algorithm would be able classify the data. As the data has in excess of one hundred dimensions it is necessary to reduce the dimensionality before plotting. Two methods were tested to reduce the dimensionality of the data to two dimensions so that it could be easily plotted. The first of these is PCA, which is relatively simple with a fast implementation, and the second is t-SNE, which is more complicated but has achieved better performance in recent works. Both of these methods are described below in more detail.

In PCA the method to express a high-dimensional point $\boldsymbol{x}$ as a low dimensional point relies on finding an approximation for this point according to(Barber, 2012, p. 330):

$$\boldsymbol{x}^n \approx \boldsymbol{c} + \sum_{j=1}^{M} y_j^n \boldsymbol{b}^j \equiv \tilde{\boldsymbol{x}}^n \tag{4.1}$$

Where the low dimensional points are $\boldsymbol{y}^n$, $\boldsymbol{c}$ is a constant and $\boldsymbol{b}_j$ are the principal components. The algorithm to find these principal components is described in Barber (2012, p. 333).

The resulting plots produced using this technique are shown in figures 4.1 and 4.2 for the in and out of proportion cases, respectively.

In the case of the data being out of proportion it appears that the groups can be partially separated. However, this is not the case for the in proportion case, and it is very difficult to discern any interactions.

A more complex method to reduce the dimensionality of the data is t-distributed Stochastic Neighbourhood Embedding (t-SNE)(Van der Maaten and Hinton, 2008). This technique won the Merck Visualization Challenge. It differs from PCA in the objective function in that it aims to maintain a similarity measure between points between the high and low-dimensional cases.

In addition to using this technique it was also recommended to use Truncated Singular Value Decomposition(SVD) to reduce the dimensionality of the vector to 50 beforehand. This was for implementation reasons in Scikit-learn.

Unfortunately, this was no more successful than PCA on this dataset, as shown in figures 4.3 and 4.4. It should also be noted that the number of points

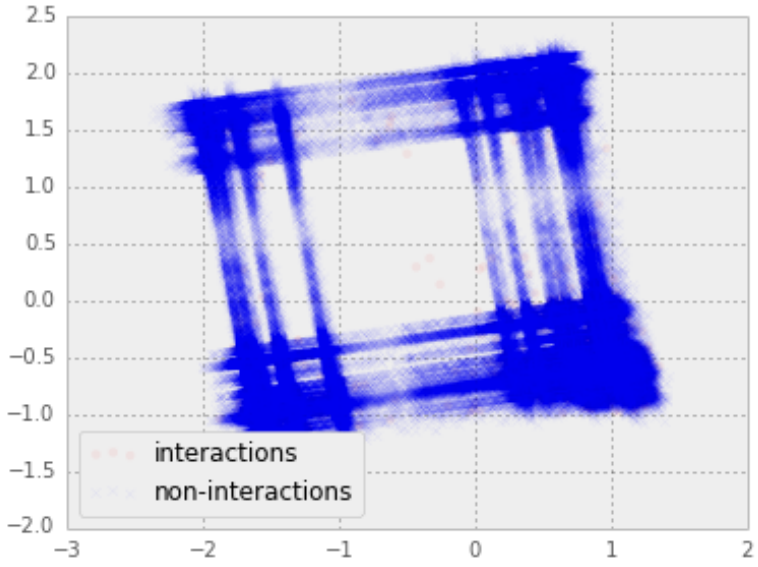


Figure 4.1: In proportion plot of the data reduced to two dimensions using PCA.

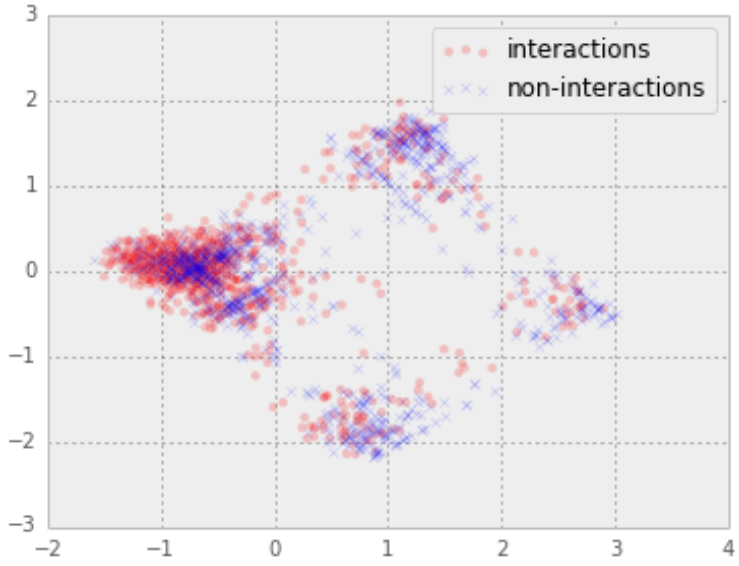


Figure 4.2: Out of proportion plot of the data reduced to two dimensions using PCA.

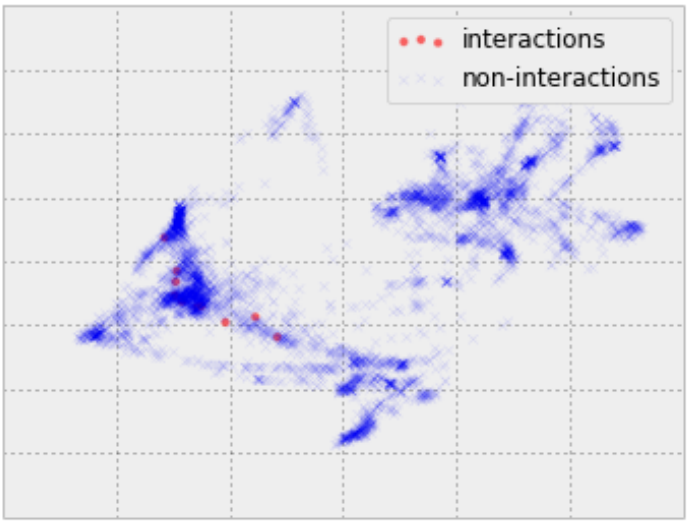


Figure 4.3: In proportion plot of the data reduced to two dimensions using t-SNE. Axes are not labeled as they are meaningless after the transformation.

plotted in these graphs is much lower than in the case of PCA as it is much more computationally intensive.

Both of the methods above suggest that this classification problem is difficult, as the points are not significantly separated in any of the plots.

4.2.1 High dimensional plots

Neglecting reducing the dimensionality of the data there are some plots which are able to illustrate a very large number of dimensions. Two which have been applied in this project are parallel lines plots and Andrew's curves(Andrews, 1972). Parallel lines plots are relatively simple in that each feature is simply scaled and plotted along the x axis at its index in the vector, producing a number of overlapping lines. Andrew's curves are more complex, and the method is described below along with the results obtained. The parallel lines plots are shown in proportion in figure 4.5 and out of proportion in figure 4.6. In either case, it is not clear whether the data easily separates into two classes. The in proportion case in particular shown in figure 4.5 is very difficult to separate into interactions and non-interactions.

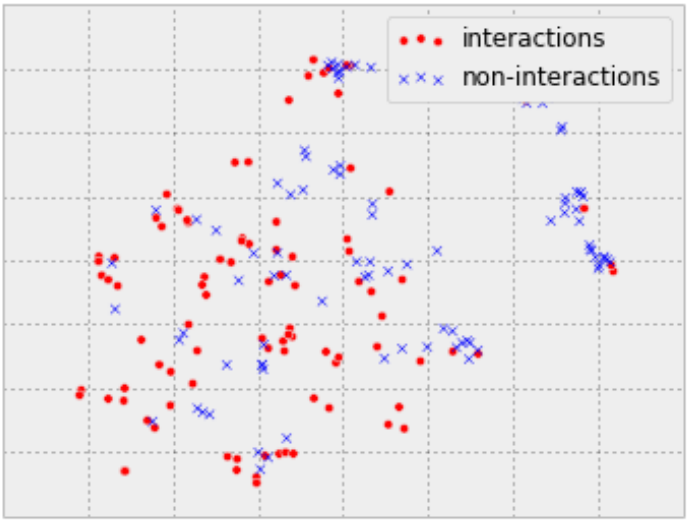


Figure 4.4: Out of proportion plot of the data reduced to two dimensions using t-SNE. Axes are not labeled as they are meaningless after the transformation.

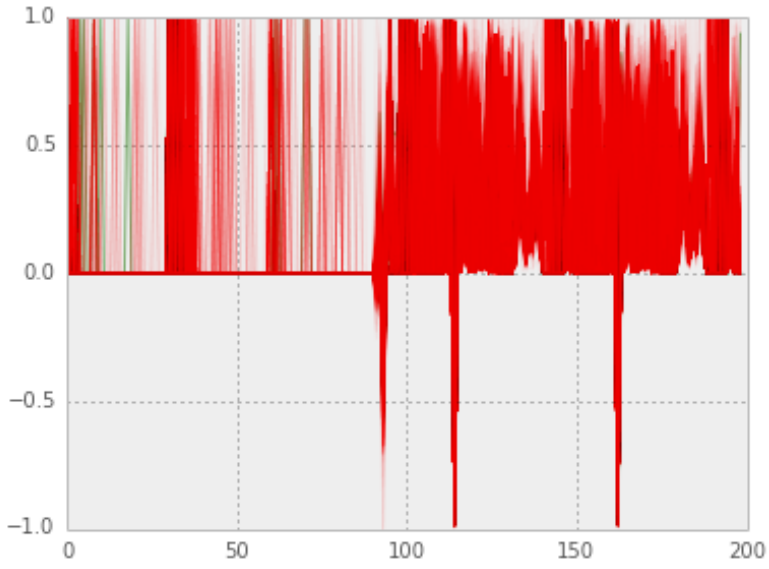


Figure 4.5: In proportion parallel line plot of the data used to train the final classifier. Each feature vector is scaled into the graph interval and plotted, overlapping.

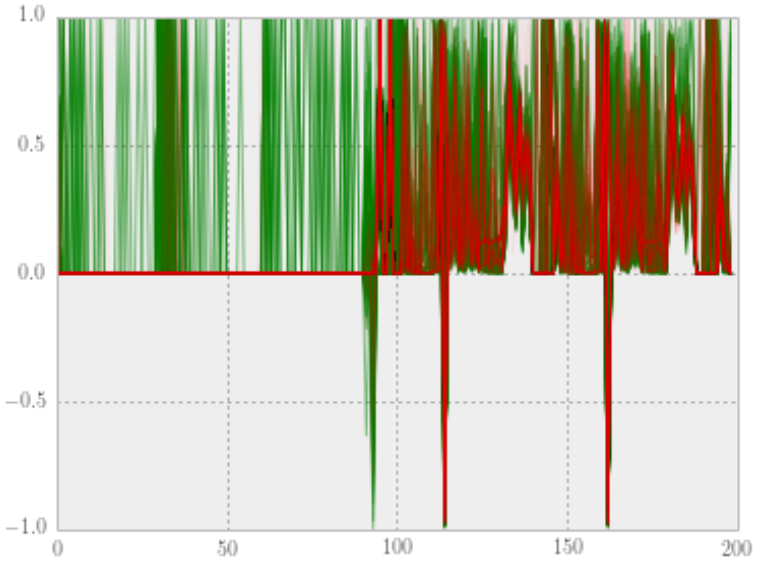


Figure 4.6: Out of proportion parallel line plot of the data used to train the final classifier. Each feature vector is scaled into the graph interval and plotted, overlapping.

4.3 Model selection results

In a classification task there are two common components: a training set that is labeled and a set of data which the classifier is to be applied to, without labels. This task is similar in that we have created a training set with labels and the active zone network forms a set of feature vectors that have no labels. However, this does not accurately encode our prior knowledge of the active zone network interactions, which were picked for their likelihood to be true interactions. In this application supervised classification alone is not sufficient to solve the problem of creating accurate weights. This drove the development of the technique described in section 3.3.

This section describes both the results of training the classifier as planned on the labeled training set and the results of the Bayesian method to weight the interactions of the active zone network.

Classifier	Hyper-parameter values	Test set accuracy	Training set accuracy
Logistic Regression	C : 0.0215	$0.9984 \pm 3 \times 10^{-5}$	$0.99847 \pm 2 \times 10^{-5}$
Support Vector Machine	kernel: RBF Gamma: 10.0 C: 10.0	0.9985± N/A	$0.99933 \pm 1.2 \times 10^{-4}$
Random Forest	N estimators: 44 Max features: 25	$0.99837 \pm 3 \times 10^{-5}$	$0.99921 \pm 3 \times 10^{-5}$
Extremely Randomized Trees	N estimators: 94 Max features: 25	$0.99837 \pm 3 \times 10^{-5}$	$0.99921 \pm 3 \times 10^{-5}$

Table 4.2: Summary of the hyper-parameter combinations found by grid search and associated statistical accuracy measures.

4.3.1 Classifier accuracy and best hyper-parameters

Grid searches of hyper-parameter values were used to find the optimal combination. The results of this search are shown in table 4.2.

Although the results of the Support Vector Machine appear to be good, it was trained on a relatively small dataset and its performance in later tests, such as ROC, were poor. For this reason, it was not chosen.

The random forest can be seen to be performing worse than the logistic regression model in this test, with a slightly lower accuracy, but within the standard error boundaries. This could be due to random variation in the accuracy score. A modified accuracy score to deal with unbalanced classes would have been preferable.

4.3.2 ROC curves

As described in section 3.2.3 ROC curves were applied to all classifiers to characterise the performance of each. The ROC curve produced by the logistic regression model is shown in figure 4.7. The positive AUC value is desirable and the curve of the ROC curve suggests this model can achieve a relatively good true positive rate at low false positive rate. To avoid false positive protein interactions being classified it would likely be used at this threshold.

However, the ROC curve shown in figure 4.8 is very similar to that created by the logistic regression model except it has slightly better performance at low false positive rates. Neither show very good performance and it is clear that only at very high false positive rates will the classifiers reliably find the true interactions.

Unfortunately, it was not possible to train the support vector machine on a large enough training set to plot a realistic ROC curve due to time constraints, so it is not shown here. The notebook referenced in appendix B.2 provides the full

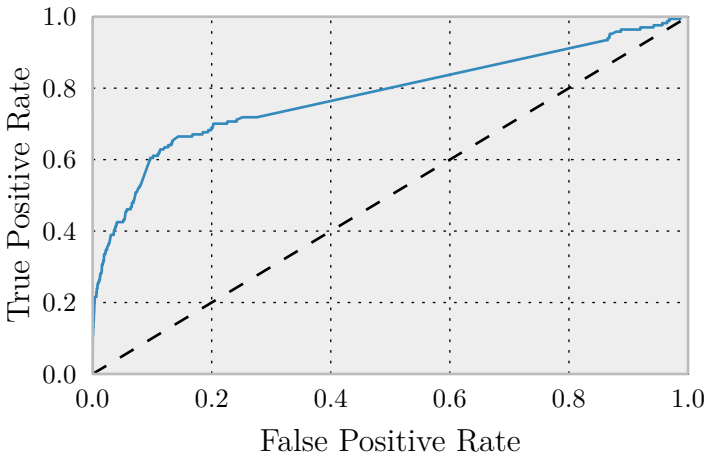


Figure 4.7: The ROC curve produced by a logistic regression model on the test data set using the best parameters shown in table 4.2 with an AUC of 0.781.

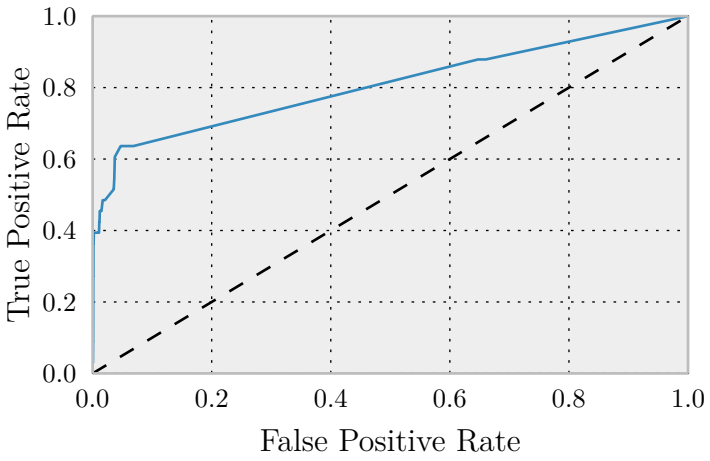


Figure 4.8: The ROC curve produced by a random forest model on the test data set using the best parameters shown in table 4.2 with an AUC of 0.806.

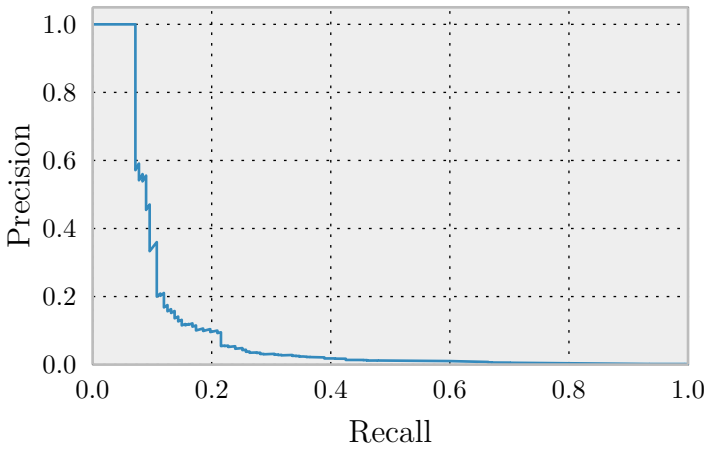


Figure 4.9: The precision-recall curve produced by a logistic regression model on the test data set using the best parameters shown in table 4.2 with an AUC of 0.11.

results obtained. This is also true of the precision-recall graph in the following section.

4.3.3 Precision-recall curves

As described in section 3.2.3 the classifiers were also characterised through plotting a precision-recall curve. This is shown for the random forest and logistic regression models in figures 4.10 and 4.9, respectively. Although the random forest model achieves a better AUC value, the curve demonstrates it is unable to recall as many samples with perfect precision as the logistic regression model.

As with the ROC curve, neither classifier shows particularly good performance.

4.3.4 Feature importances

Both logistic regression models and random forests can quickly return feature importances. In the case of logistic regression, this is as simple as looking at the weights applied to each input feature. Random forests store the importances of each feature internally using Scikit-learn and this can be easily queried. These importances measures are plotted in figures 4.11 and 4.12. Both show a relatively even distribution of weights over the features, between those from Gene Ontology or derived from features in Rodgers-Melnick, Culp and DiFazio (2013).

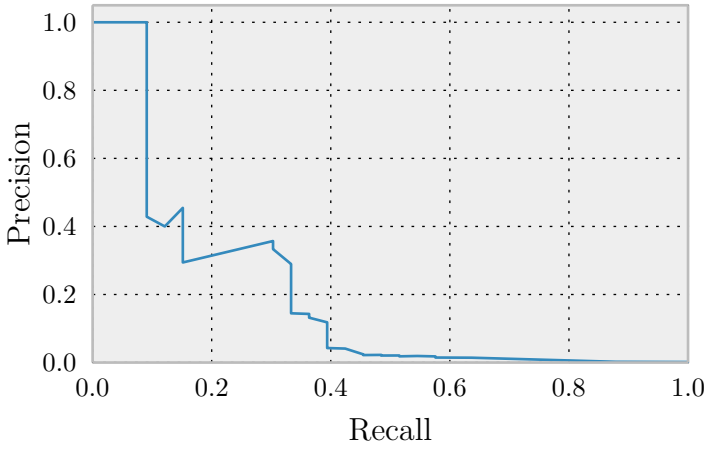


Figure 4.10: The precision-recall curve produced by a random forest model on the test data set using the best parameters shown in table 4.2 with an AUC of 0.12. This graph shows some artefacts due to the extremely small number of positive interactions even in relatively large sample sizes.

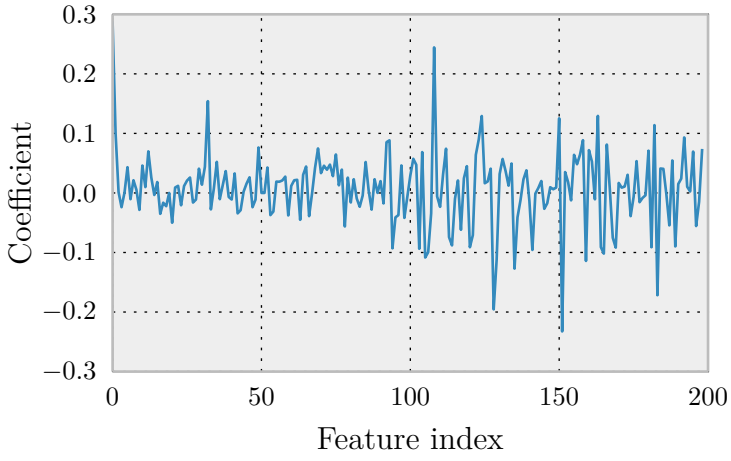


Figure 4.11: The internal weighting of features in a logistic regression model trained using the parameters described in table 4.2. The coefficients plotted on the y axis have no unit. Feature indices are as described in table 3.1.

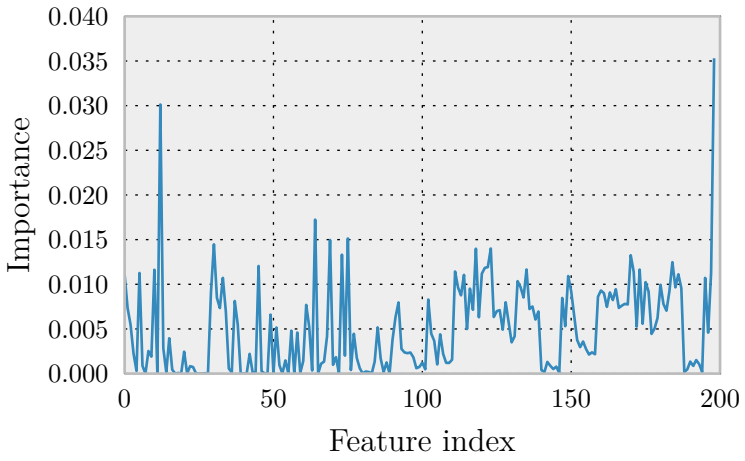


Figure 4.12: The importances of each input feature as reported by the random forest model trained using the parameters described in table 4.2. The importances plotted on the y axis have no unit. Feature indices are as described in table 3.1.

Parameter	Value
Edgelist TPR	0.9
Edgelist TNR	0.5
iRefIndex TPR	0.9
iRefIndex TNR	0.5

Table 4.3: A table summarising the parameters chosen in the probabilistic model used two weight the interactions.

4.3.5 Bayesian weighting of interactions

Confidence values were chosen to be conservative but otherwise were arbitrary to the task and could be more extensively assessed in future work. Estimating the error rates of the relevant databases was beyond the scope of this project. The values chosen are shown in table 4.3. As the Edgelist has been reviewed and because iRefIndex represents a combination of many databases a True Positive Rate (TPR) of 0.9 seemed like a fair estimate of the accuracy of positive predictions. Negative predictions were more difficult to estimate. Both TNRs were set to be 0.5, which is very likely to be an underestimate as most interactions are false.

Once the Bayesian updating of the weights was completed it was necessary to check the distribution of the weights produced. This distribution is shown in figure 4.13 The interactions not found in the iRefIndex database will have a lower

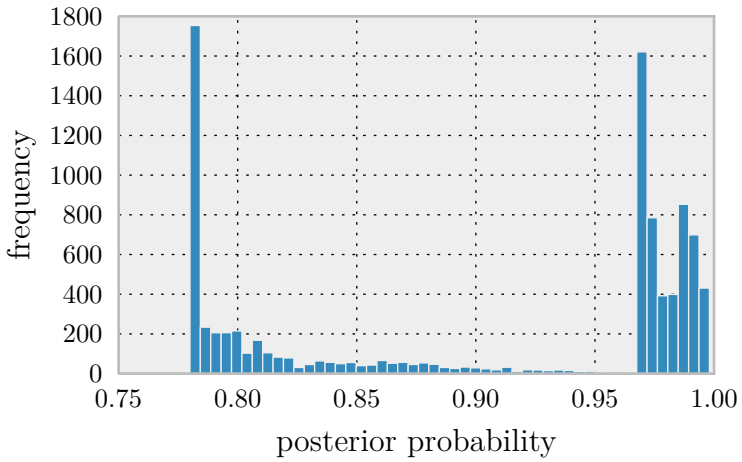


Figure 4.13: A histogram of the weights produced using the method of Bayesian weight generation described in section 3.3.

posterior than those that are, causing the grouping seen in figure 4.13.

4.4 Comparison of weighted and unweighted PPI networks

Once the weighted edges had been generated the communities in both the weighted and unweighted cases were found using a Spectral Modularity algorithm. The resulting graphs of the active zone network, divided into communities, are shown in figures 4.14 and 4.15 for the unweighted and weighted cases, respectively. By inspection, it is easy to see that the clusters detected differ.

This observation is mirrored in the results of the NMI test. Both NMI test implementations returned a value of 0.604, indicating that the clustering is quite similar.

4.4.1 Disease enrichment

The communities detected to be involved in disease differ by number between weighted and unweighted. As these numbers are arbitrary indexes, it is necessary to compare these communities between the two graphs to see if both graphs contain similar communities involved in disease. The significant disease communities discovered in the unweighted and weighted cases can be seen in tables 4.4 and 4.5, respectively.

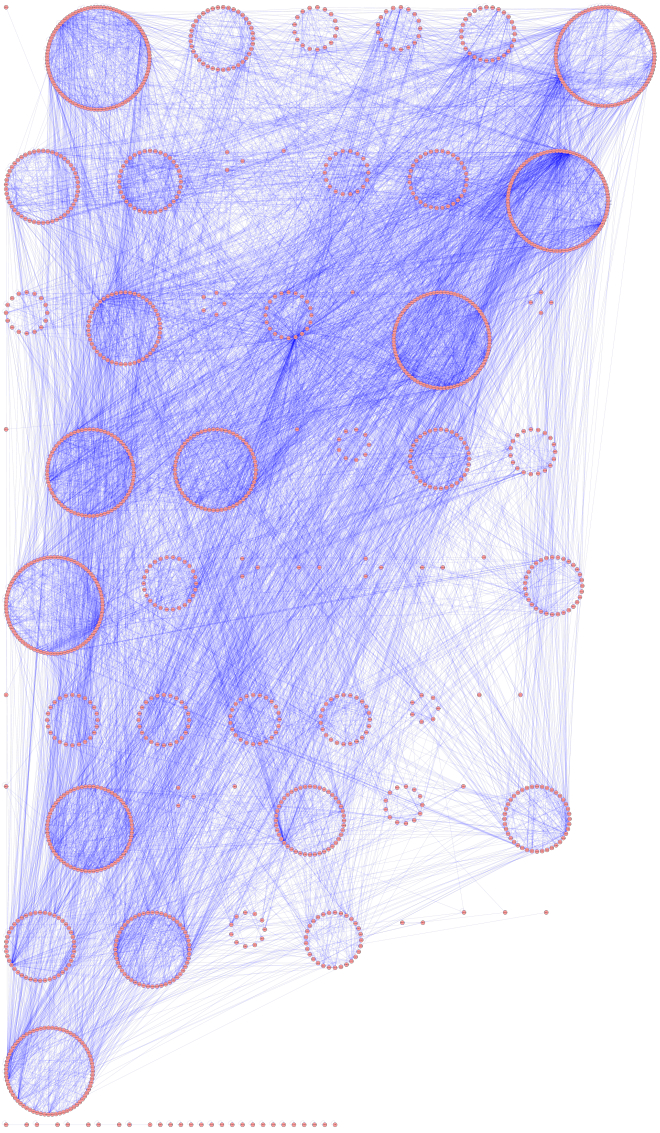


Figure 4.14: The unweighted graph of the active zone network divided into communities.

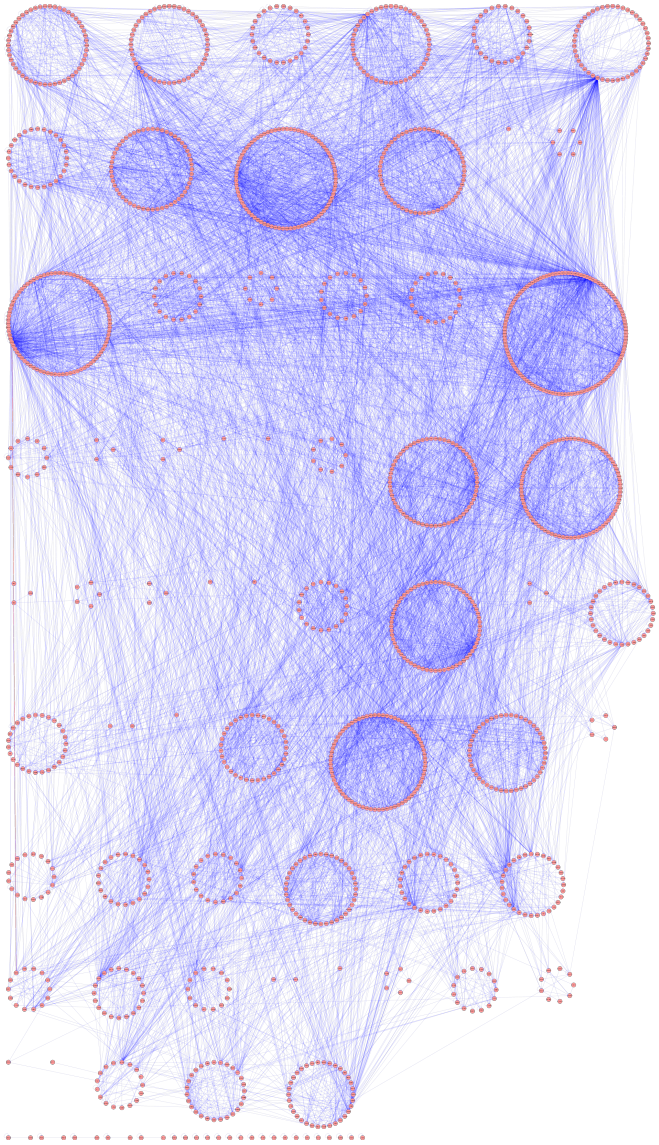


Figure 4.15: The weighted graph of the active zone network divided into communities.

Community	Size	disease	p-value	p_lower (%)
44	35	schizophrenia	3.45×10^{-3}	0.6
45	73	schizophrenia	3.90×10^{-3}	0.4
48	17	Alzheimer's_disease	7.57×10^{-3}	0.2
33	64	Alzheimer's_disease	8.39×10^{-3}	0.7
33	64	schizophrenia	2.33×10^{-2}	2.2
14	18	schizophrenia	4.19×10^{-2}	3.2
2	48	schizophrenia	4.67×10^{-2}	4.9
15	8	Alzheimer's_disease	4.98×10^{-2}	5.2

Table 4.4: Disease enrichment results for the weighted communities detected in the active zone network.

Community	Size	disease	p-value	p_lower (%)
29	88	schizophrenia	1.69×10^{-3}	0.2
2	99	schizophrenia	4.32×10^{-3}	0.5
30	26	Alzheimer's_disease	5.37×10^{-3}	0.7
61	24	schizophrenia	1.96×10^{-2}	1.5
63	23	Alzheimer's_disease	3.50×10^{-2}	4.3
61	24	Alzheimer's_disease	4.25×10^{-2}	4.5
63	23	schizophrenia	4.62×10^{-2}	4.3

Table 4.5: Disease enrichment results for the unweighted communities detected in the active zone network.

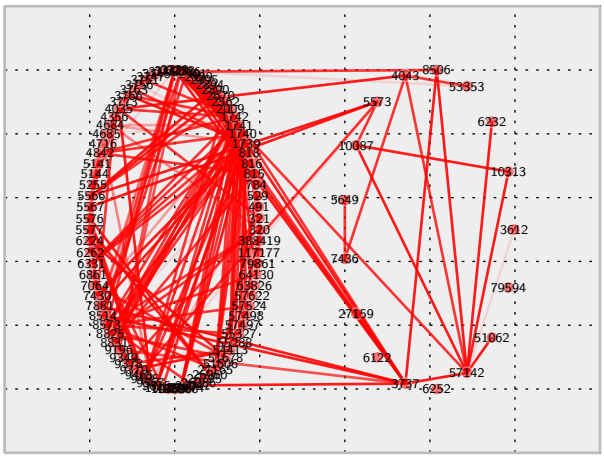
Looking at the most likely community for Schizophrenia in the unweighted graph, it is possible to compare the members of this community to the members of all the communities in the weighted graph to find the closest match. Community 29 in the unweighted graph was found to be the most likely to be involved in Schizophrenia. The membership test used equates to the number of members shared between two communities over the number of members in either community. For this community it was found that community 33 in the weighted graph most closely matched 29 with a value of 0.45.

These graphs were then plotted to investigate the reason for this difference as shown in figure 4.16. It can be seen that there are many more points found in the unweighted graph that are also in the weighted graph than vice versa. In this graph it is not clear why many of these proteins have been separated, there are no clear weak connections holding together the unweighted case. However, in figure 4.17 we can see that the weighted graph contains most of the proteins of the unweighted community minus a set that are very weakly connected.

Conclusion

Although it is easy to demonstrate that weighting the graph has had an effect on the communities detected, it is much more difficult to discern what this effect has been or how the weighting has brought it about. However, many of the results presented here are positive, showing that the various techniques involved can be used to produce a result.

Unweighted community 29 interactions



Weighted community 33 interactions

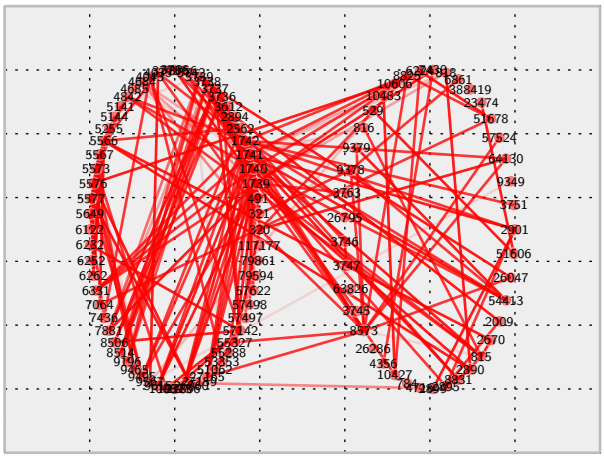
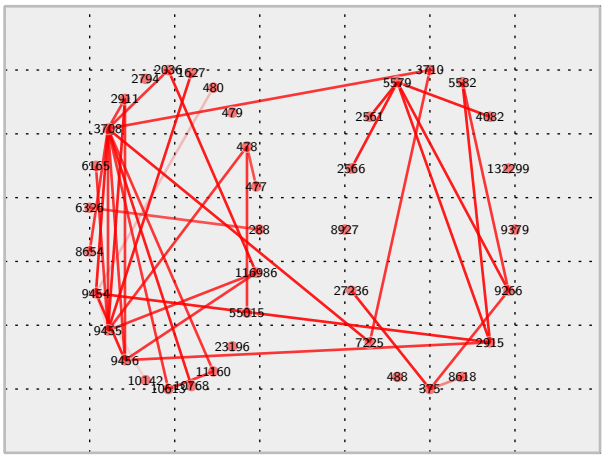


Figure 4.16: The communities 29 and 33 from the unweighted and weighted graphs, respectively are plotted. In each plot the nodes not shared in each community are plotted separated from the main community on the right.

Unweighted community 64 interactions



Weighted community 44 interactions

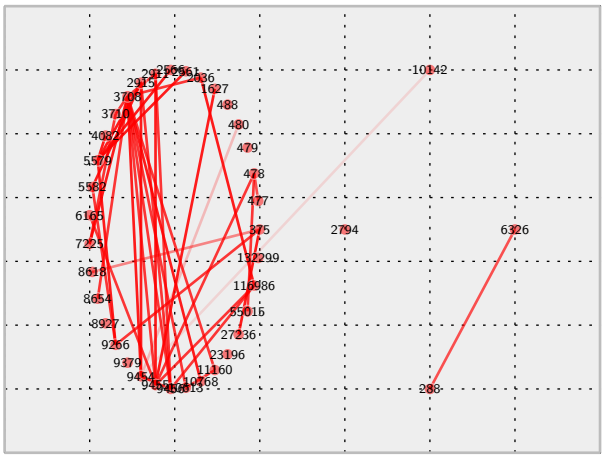


Figure 4.17: The communities 64 and 44 from the unweighted and weighted graphs, respectively are plotted. In each plot the nodes not shared in each community are plotted separated from the main community on the right.

Chapter 5

Conclusions

This project involved an array of different tools and data. Despite many difficulties and mistakes during the planning of the project, all of the aims of the project as stated in the proposal have been met. Despite lacking conclusive results this work represents a step towards a system for unifying diverse interaction data sources.

5.1 Deliverables

During the course of the project a large number of data sources were tested and extracted into a machine learning workflow. These formed large feature vector files which could be used for classification. Three different classifiers were then trained on this data and compared. Of these, the best was used to provide predictions to weight edges. A Bayesian method was used to combine this with other data sources and prior knowledge to generate the edge weights. These edge weights were then used to create a weighted PPI graph which was then compared to its unweighted counterpart.

5.2 Future work

As a basis for future work, this work illustrates many difficulties in working with varied publicly available data sources. However, it also provides insight into the correct method for weighting interaction edges. These edges should be weighted directly as an estimate of interaction strength. The premise of this project was that the posterior probability of the interaction existing would correlate well with

the strength of the interaction as interactions which are strong will be observed more often.

However, if a full probabilistic model was to be designed the latent variable - which was interaction in our model - could be a continuous variable in the unit interval defined at a Beta distribution. The problem then becomes one of estimating interaction strength, which is difficult to observe in order to obtain the training set required to create a probabilistic model. Using the array of biological databases available it would be possible to link different observations based on strong biological prior knowledge.

Conclusion

Despite the results of this project, the resources and insight into better solutions generated made it worthwhile. In the future it, building on the results of this project it will be possible to create a PPI network that accurately summarises our knowledge of interactions and interaction strength using all of the available data.

Appendix A

Repository

Git is version control software and was used extensively in this project in combination with Github(GitHub, Inc, 2008) and git-annex(Hess, 2014). The project is built from two repositories, the first inside the second: a large git-annex based repository containing all the data and a smaller git repository stored on github containing all the code and documentation for the report(Gray and Maguire, 2014b). History for all the code and documentation, every previous version, is available publicly online but the data cannot be made publicly available.

Hosting the code online was intended to aid remote collaborators. It is also useful in order to maintain an accurate log of the work involved in the project. The repository also includes a wiki(Gray and Maguire, 2014a) providing documentation on some of the code available in the repository and weekly reports covering the entire project.

The repository consists of several directories:

- notebooks:
  - Contains IPython notebooks covering code executed during the project.
  - Each notebook includes inline documentation on what is being done, and why.
- ocbio:
  - This contains the Python module of code developed during the project.
  - The major component of this is the extract.py file, which deals with writing feature vectors for use in classification.

- proposal:
  - Only contains the original proposal for the project.
- report:
  - Contains this report and all the required files to compile it.
  - Based on a repository for Masters project templates(Chris Brown, 2011) with modifications by Danilo Orlando.
- scripts:
  - Contains scripts, but these were not used during the bulk of the project.

The code in this repository may be useful to a future project, but could also be substantially improved upon. It is more likely that the notes on how to go about a protein interaction prediction task, and this report, will be more useful to future work.

Weekly reports were kept as a summary of the work completed every week for supervisors and to maintain a log of the project. These can be found on the project wiki(Gray and Maguire, 2014a).

A.0.1 Parallel processing with IPython.parallel

To take maximum advantage of the available computing facilities and because the sample sizes in the project exceed one million the decision was made to prepare the code for parallel processing on a remote server.

Particularly, grid search operations to optimize performance of the classifier were considered to be processor intensive and vital to the success of the prediction task. The easiest way to set up these interactive parallel processing operations was the parallel processing model in IPython(The IPython Development Team, 2014).

The notebooks using parallel processing are the notebooks on classifier training, which are described in Appendix B.2. This usage depended on code from a parallel processing tutorial(Grisel, 2013) to distribute memory to the cores using Numpy's memmap methods. The code to do this has been integrated into the ocbio module in the project repository and can be used as shown in the notebooks.

Potentially, and as described in the tutorial, this code could be used to run the classifier training on cloud services using Starcluster.

Appendix B

Notebooks

The job of quickly running arbitrary processing on a variety of different data sources, each of which are being encountered for the first time was approached using IPython notebooks. Quick interactive programming was useful as unexpected problems could be quickly solved. Also, a detailed log, with inline comments, could be kept to track exactly what was done.

It would be possible for anyone with access to the data to run this code again to verify it. The code was run with Python version 2.7.7 and Scikit-learn 0.15.0. The notebooks can be found at the following locations:

- The root directory for these notebooks can be found here: `https://github.com/ggray1729/opencast-bio/tree/master/notebooks`
- These can be viewed here: `http://nbviewer.ipython.org/github/ggray1729/opencast-bio/tree/master/notebooks/`

B.1 Feature Extraction Notebooks

Of the feature extraction notebooks in the repository, not all of them were successful. Only those that were successful are listed here.

B.1.1 Gene Ontology

In total 90 features were extracted from the Gene Ontology as binary values. The following notebook describes how the features applied were generated:

- The notebook in the opencast-bio repository can be found here: `https://github.com/ggray1729/opencast-bio/blob/master/notebooks/Extracting%20Gene%20Ontology%20features%200.2.ipynb`
- This notebook can be viewed here: `http://nbviewer.ipython.org/github/ggray1729/opencast-bio/blob/master/notebooks/Extracting%20Gene%20Ontology%20features%200.2.ipynb`

B.1.2 Features derived from ENTS

107 features were generated derived from the work in Rodgers-Melnick, Culp and DiFazio (2013). These were found by analysing the code provided on the papers web page as described in the following notebook:

- The notebook in the opencast-bio repository can be found here: `https://github.com/ggray1729/opencast-bio/blob/master/notebooks/Inspecting%20ENTS%20code.ipynb`
- This notebook can be viewed here: `http://nbviewer.ipython.org/github/ggray1729/opencast-bio/blob/master/notebooks/Inspecting%20ENTS%20code.ipynb`

B.2 Classifier Training

This notebook contains all the code that was run to train and test the classifier used in this project. It involves model selection, grid search of parameters and various plots describing the performance of different classifiers, such as ROC curves and Precision Recall curves. Links are provided to the code and an online service to view the notebook:

- The notebook in the opencast-bio repository can be found here: `https://github.com/ggray1729/opencast-bio/blob/master/notebooks/Classifier%20Training.ipynb`
- This notebook can be viewed here: `http://nbviewer.ipython.org/github/ggray1729/opencast-bio/blob/master/notebooks/Classifier%20Training.ipynb`

In this notebook both learning curves and pipelines are used in the training of the classifiers. These are described in the following sections:

Pipeline

A pipeline is a combination of algorithms intended to run on the data in sequence after the data has been split into training and test. In the case of this project the pipeline involved three components: a mean value filling imputer, a standard scaler and the classifier itself. The imputer simply replaced missing values in the training data with the corresponding mean value for that column. Scikit-learn's standard scaler centers the data at zero mean and unit variance.

Learning Curves

Learning curves, as in the notebook referenced in Appendix B.2, were used in this project to ensure that the number of samples used in a grid search was sufficient. The learning curve plots the accuracy of a classifier after it has been trained using cross-validation on a varying number of samples.

B.3 ocbio.extract Usage

This notebook describes how to use the code developed in the project to build feature vectors from the various data sources. It can be found in the following locations:

- The notebook in the opencast-bio repository can be found here: `https://github.com/ggray1729/opencast-bio/blob/master/notebooks/ocbio.extract%20usage%20notes.ipynb`
- This notebook can be viewed here: `http://nbviewer.ipython.org/github/ggray1729/opencast-bio/blob/master/notebooks/ocbio.extract%20usage%20notes.ipynb`

Appendix C

Data sources

These data sources were gathered on the project wiki, which can be found here: `https://github.com/ggray1729/opencast-bio/wiki/Feature-extraction`. The vast majority could not be included in the project due to time constraints but the list may be useful for future work.

C.1 Gene Ontology Features

Each feature used in the Gene Ontology feature represents an event where both proteins in the interaction have the same Gene Ontology term active. If both are active the feature is 1, otherwise it is zero. To select effective features, commonly occurring features in the active zone network were selected from each Gene Ontology domain. These are summarised by domain and in the order used in the feature vectors in table C.1 for the first 10 of each domain. The remaining terms are listed in table C.2.

Domain	Term	Occurrences in active zone	Index
Molecular Function	protein binding	960	1
	poly(A) RNA binding	308	2
	ATP binding	232	3
	metal ion binding	137	4
	identical protein binding	100	5
	RNA binding	99	6
	protein homodimerization activity	97	7
	calcium ion binding	88	8
	zinc ion binding	79	9
	GTP binding	78	10
Cellular Component	cytoplasm	614	31
	cytosol	575	32
	nucleus	554	33
	extracellular vesicular exosome	550	34
	plasma membrane	449	35
	mitochondrion	265	36
	integral component of membrane	252	37
	nucleolus	235	38
	nucleoplasm	143	39
	membrane	130	40
Biological Process	small molecule metabolic process	266	61
	gene expression	184	62
	viral process	144	63
	signal transduction	130	64
	cellular protein metabolic process	125	65
	synaptic transmission	114	66
	RNA metabolic process	104	67
	mRNA metabolic process	99	68
	transmembrane transport	98	69
	blood coagulation	97	70

Table C.1: The features used in the Gene Ontology feature, by domain, and in the order used in the feature vector. The observed frequency in the active zone network is also shown. These are only listed for the first 10 of each domain. The remaining features, in order, are shown in table C.2.

Domain		
Molecular Function	Cellular Component	Biological Process
Indices		
1-30	31-60	61-90
protein binding	cytoplasm	small molecule metabolic process
poly(A) RNA binding	cytosol	gene expression
ATP binding	nucleus	viral process
metal ion binding	extracellular vesicular exosome	signal transduction
identical protein binding	plasma membrane	cellular protein metabolic process
RNA binding	mitochondrion	synaptic transmission
protein homodimerization activity	integral component of membrane	RNA metabolic process
calcium ion binding	nucleolus	mRNA metabolic process
zinc ion binding	nucleoplasm	transmembrane transport
GTP binding	membrane	blood coagulation
nucleotide binding	cell junction	apoptotic process
protein kinase binding	perinuclear region of cytoplasm	innate immune response
DNA binding	Golgi apparatus	axon guidance
structural constituent of ribosome	endoplasmic reticulum	translation
GTPase activity	integral component of plasma membrane	translational initiation
actin binding	extracellular space	transcription, DNA-templated
enzyme binding	mitochondrial inner membrane	RNA splicing
molecular_function	mitochondrial matrix	mitotic cell cycle
protein complex binding	endoplasmic reticulum membrane	mRNA splicing, via spliceosome
calmodulin binding	extracellular region	nuclear-transcribed mRNA catabolic process, nonsense-mediated decay
protein serine/threonine kinase activity	dendrite	oxidation-reduction process
receptor binding	intracellular membrane-bounded organelle	translational elongation
protein domain specific binding	cytoskeleton	GTP catabolic process
structural molecule activity	microtubule	carbohydrate metabolic process
protein heterodimerization activity	Golgi membrane	viral life cycle
protein C-terminus binding	neuronal cell body	protein phosphorylation
SH3 domain binding	protein complex	negative regulation of apoptotic process
structural constituent of cytoskeleton	ribonucleoprotein complex	ATP catabolic process
transporter activity	centrosome	SRP-dependent cotranslational protein targeting to membrane
signal transducer activity	actin cytoskeleton	cell death

Table C.2: The full list of GO terms used in the feature vector, by index, is shown.

Bibliography

Alberts, B., Johnson, A., Lewis, J., Raff, M., Roberts, K. and Walter, P. (2008) *Molecular Biology of the Cell*, 5th, Garland Science, ISBN: 0-8153-3218-1; 0-8153-4072-9.

Andrews, D.F. (1972) 'Plots of high-dimensional data', *Biometrics*, pp. 125–136, URL: http://www.jstor.org/stable/2528964 (visited on 08/08/2014).

Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T., Harris, M.A., Hill, D.P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J.C., Richardson, J.E., Ringwald, M., Rubin, G.M. and Sherlock, G. (2000) 'Gene Ontology: tool for the unification of biology', en, *Nature Genetics* 25 (1), pp. 25–29, ISSN: 1061-4036, DOI: 10.1038/75556, URL: http://www.nature.com/ng/journal/v25/n1/abs/ng0500_25.html (visited on 22/04/2014).

Baolin, L. and Bo, H. (2007) 'HPRD: a high performance RDF database', *Network and Parallel Computing*, Springer, pp. 364–374, URL: http://link.springer.com/chapter/10.1007/978-3-540-74784-0_37 (visited on 14/08/2014).

Barber, D. (2012) *Bayesian Reasoning and Machine Learning*, en, Cambridge University Press, ISBN: 9780521518147.

Black, D.L. (2003) 'Mechanisms of Alternative Pre-Messenger Rna Splicing', *Annual Review of Biochemistry* 72 (1), pp. 291–336, DOI: 10.1146/annurev.biochem.72.121801.161720, URL: http://dx.doi.org/10.1146/annurev.biochem.72.121801.161720 (visited on 14/08/2014).

Blum, T., Briesemeister, S. and Kohlbacher, O. (2009) 'MultiLoc2: integrating phylogeny and Gene Ontology terms improves subcellular protein localiza-

tion prediction', *BMC bioinformatics* 10 (1), p. 274, URL: http://www.biomedcentral.com/1471-2105/10/274/ (visited on 07/08/2014).

Chen, B., Fan, W., Liu, J. and Wu, F.-X. (2013) 'Identifying protein complexes and functional modules—from static PPI networks to dynamic PPI networks', en, *Briefings in Bioinformatics*, bbt039, ISSN: 1467-5463, 1477-4054, DOI: 10.1093/bib/bbt039, URL: http://bib.oxfordjournals.org/content/early/2013/06/18/bib.bbt039 (visited on 20/03/2014).

Chris Brown, A.S. (2011) *UG4 Project Report Template*, URL: https://github.com/proa/ug4-report-template (visited on 13/08/2014).

Chua, J.J.E., Kindler, S., Boyken, J. and Jahn, R. (2010) 'The architecture of an excitatory synapse', en, *Journal of Cell Science* 123 (6), pp. 819–823, ISSN: 0021-9533, 1477-9137, DOI: 10.1242/jcs.052696, URL: http://jcs.biologists.org/content/123/6/819 (visited on 23/04/2014).

Cristianini, N. and Shawe-Taylor, J. (2000) *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*, en, Cambridge University Press, ISBN: 9780521780193.

Futschik, M.E., Chaurasia, G. and Herzel, H. (2007) 'Comparison of human protein–protein interaction maps', en, *Bioinformatics* 23 (5), pp. 605–611, ISSN: 1367-4803, 1460-2059, DOI: 10.1093/bioinformatics/btl683, URL: http://bioinformatics.oxfordjournals.org/content/23/5/605 (visited on 14/08/2014).

Gallone, G., Simpson, T.I., Armstrong, J.D. and Jarman, A.P. (2011) 'Bio::Homology::InterologW - A Perl module to build putative protein-protein interaction networks through interolog mapping', en, *BMC Bioinformatics* 12 (1), p. 289, ISSN: 1471-2105, DOI: 10.1186/1471-2105-12-289, URL: http://www.biomedcentral.com/1471-2105/12/289/abstract (visited on 09/04/2014).

GitHub, Inc (2008) *GitHub*, URL: https://github.com/ (visited on 13/08/2014).

Gray, G. and Maguire, F. (2014a) *opencast-bio project wiki*, URL: https://github.com/ggray1729/opencast-bio/wiki (visited on 13/08/2014).

Gray, G. and Maguire, F. (2014b) *opencast-bio repository*, URL: https://github.com/ggray1729/opencast-bio (visited on 13/08/2014).

Grisel, O. (2013) *Parallel Machine Learning with scikit-learn and IPython*, URL: `https://github.com/ogrisel/parallel_ml_tutorial` (visited on 13/08/2014).

Hanczar, B., Hua, J., Sima, C., Weinstein, J., Bittner, M. and Dougherty, E.R. (2010) 'Small-sample precision of ROC-related estimates', *Bioinformatics* 26 (6), pp. 822–830, URL: `http://bioinformatics.oxfordjournals.org/content/26/6/822.short` (visited on 05/08/2014).

Hermjakob, H., Montecchi-Palazzi, L., Lewington, C., Mudali, S., Kerrien, S., Orchard, S., Vingron, M., Roechert, B., Roepstorff, P., Valencia, A. et al. (2004) 'IntAct: an open source molecular interaction database', *Nucleic acids research* 32 (suppl 1), pp. D452–D455, URL: `http://nar.oxfordjournals.org/content/32/suppl_1/D452.short` (visited on 14/08/2014).

Hess, J. (2014) *git-annex*, URL: `https://git-annex.branchable.com/` (visited on 13/08/2014).

Hubbard, T., Barker, D., Birney, E., Cameron, G., Chen, Y., Clark, L., Cox, T., Cuff, J., Curwen, V., Down, T. et al. (2002) 'The Ensembl genome database project', *Nucleic acids research* 30 (1), pp. 38–41, URL: `http://nar.oxfordjournals.org/content/30/1/38.short` (visited on 13/08/2014).

Ito, T., Chiba, T., Ozawa, R., Yoshida, M., Hattori, M. and Sakaki, Y. (2001) 'A comprehensive two-hybrid analysis to explore the yeast protein interactome', en, *Proceedings of the National Academy of Sciences* 98 (8), pp. 4569–4574, ISSN: 0027-8424, 1091-6490, DOI: `10.1073/pnas.061034498`, URL: `http://www.pnas.org/content/98/8/4569` (visited on 25/07/2014).

Janin, J. and Chothia, C. (1985) 'Domains in proteins: definitions, location, and structural principles', *Methods in enzymology* 115, pp. 420–430, URL: `http://www.sciencedirect.com/science/article/pii/0076687985150305/pdf?md5=e4ec0621f9047761ac85b3c3fa8808e2&pid=1-s2.0-0076687985150305-main.pdf&_valck=1` (visited on 07/08/2014).

John, G.H. and Langley, P. (1995) 'Estimating continuous distributions in Bayesian classifiers', *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, Morgan Kaufmann Publishers Inc., pp. 338–345, URL: `http://dl.acm.org/citation.cfm?id=2074196` (visited on 29/07/2014).

Kandel, E., Schwartz, J. and Jessell, T. (2000) *Principles of Neural Science*, English, 4 edition, New York: McGraw-Hill Medical, ISBN: 9780838577011.

Kenley, E.C. and Cho, Y.-R. (2011) 'Detecting protein complexes and functional modules from protein interaction networks: A graph entropy approach', en, *PROTEOMICS* 11 (19), pp. 3835–3844, ISSN: 1615-9861, DOI: 10.1002/pmic.201100193, URL: http://onlinelibrary.wiley.com/doi/10.1002/pmic.201100193/abstract (visited on 17/03/2014).

Lancichinetti, A., Fortunato, S. and Radicchi, F. (2008) 'Benchmark graphs for testing community detection algorithms', *Physical Review E* 78 (4), p. 046110, URL: http://journals.aps.org/pre/abstract/10.1103/PhysRevE.78.046110 (visited on 13/08/2014).

Li, K.W., Klemmer, P. and Smit, A.B. (2010) 'Interaction proteomics of synapse protein complexes', en, *Analytical and Bioanalytical Chemistry* 397 (8), pp. 3195–3202, ISSN: 1618-2642, 1618-2650, DOI: 10.1007/s00216-010-3658-z, URL: http://link.springer.com/article/10.1007/s00216-010-3658-z (visited on 20/03/2014).

Lodish, H., Berk, A., Zipursky, S.L., Matsudaira, P., Baltimore, D. and Darnell, J. (2000) *Molecular Cell Biology*, 4th, W. H. Freeman, ISBN: 0-7167-3136-3.

Mackay, D.J.C. (2003) *Information Theory, Inference, and Learning Algorithms*, ISBN: 0521642981.

Maglott, D., Ostell, J., Pruitt, K.D. and Tatusova, T. (2007) 'Entrez Gene: gene-centered information at NCBI', en, *Nucleic Acids Research* 35 (suppl 1), pp. D26–D31, ISSN: 0305-1048, 1362-4962, DOI: 10.1093/nar/gkl993, URL: http://nar.oxfordjournals.org/content/35/suppl_1/D26 (visited on 29/07/2014).

McDowall, M.D., Scott, M.S. and Barton, G.J. (2009) 'PIPs: human protein–protein interaction prediction database', en, *Nucleic Acids Research* 37 (suppl 1), pp. D651–D656, ISSN: 0305-1048, 1362-4962, DOI: 10.1093/nar/gkn870, URL: http://nar.oxfordjournals.org/content/37/suppl_1/D651 (visited on 25/06/2014).

Mclean, C., Kotsos, T., He, X., Simpson, I. and Armstrong, D. (2014) 'Modularity Based Clustering Suite'.

Murphy, K.P. (2012) *Machine Learning: A Probabilistic Perspective*, London: The MIT Press, ISBN: 978-0-262-01802-0, URL: http://dl.acm.org/citation.cfm?id=2380985.

Newman, M.E.J. (2006) 'Modularity and community structure in networks', en, *Proceedings of the National Academy of Sciences* 103 (23), pp. 8577–8582, ISSN: 0027-8424, 1091-6490, DOI: 10.1073/pnas.0601602103, URL: http://www.pnas.org/content/103/23/8577 (visited on 17/03/2014).

Newman, M.E.J. (2012) 'Communities, modules and large-scale structure in networks', en, *Nature Physics* 8 (1), pp. 25–31, ISSN: 1745-2473, DOI: 10.1038/nphys2162, URL: http://www.nature.com/nphys/journal/v8/n1/abs/nphys2162.html (visited on 17/03/2014).

Newman, M.E.J. and Girvan, M. (2004) 'Finding and evaluating community structure in networks', *Physical Review E* 69 (2), p. 026113, DOI: 10.1103/PhysRevE.69.026113, URL: http://link.aps.org/doi/10.1103/PhysRevE.69.026113 (visited on 17/03/2014).

Olesen, J., Gustavsson, A., Svensson, M., Wittchen, H.-U., Jönsson, B., CDBE2010 study group and European Brain Council (2012) 'The economic cost of brain disorders in Europe', eng, *European Journal of Neurology: The Official Journal of the European Federation of Neurological Societies* 19 (1), pp. 155–162, ISSN: 1468-1331, DOI: 10.1111/j.1468-1331.2011.03590.x.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. et al. (2011) 'Scikit-learn: Machine learning in Python', *The Journal of Machine Learning Research* 12, pp. 2825–2830, URL: http://dl.acm.org/citation.cfm?id=2078195 (visited on 29/07/2014).

Qi, Y. (2008) 'Learning of protein interaction networks', PhD thesis, Universitat Pompeu Fabra, Spain, URL: http://www-2.cs.cmu.edu/~qyj/paper_CMU/qyj-disertation-v17-final.pdf (visited on 09/04/2014).

Qi, Y., Bar-Joseph, Z. and Klein-Seetharaman, J. (2006) 'Evaluation of different biological data and computational classification methods for use in protein interaction prediction', en, *Proteins: Structure, Function, and Bioinformatics* 63 (3), pp. 490–500, ISSN: 1097-0134, DOI: 10.1002/prot.20865, URL: http:

//onlinelibrary.wiley.com/doi/10.1002/prot.20865/abstract (visited on 25/03/2014).

Razick, S., Magklaras, G. and Donaldson, I.M. (2008) 'iRefIndex: A consolidated protein interaction database with provenance', en, *BMC Bioinformatics* 9 (1), p. 405, ISSN: 1471-2105, DOI: 10.1186/1471-2105-9-405, URL: http://www.biomedcentral.com/1471-2105/9/405/abstract (visited on 29/07/2014).

Rodgers-Melnick, E., Culp, M. and DiFazio, S.P. (2013) 'Predicting whole genome protein interaction networks from primary sequence data in model and non-model organisms using ENTS', en, *BMC Genomics* 14 (1), p. 608, ISSN: 1471-2164, DOI: 10.1186/1471-2164-14-608, URL: http://www.biomedcentral.com/1471-2164/14/608/abstract (visited on 24/06/2014).

Schaefer, M.H., Fontaine, J.-F., Vinayagam, A., Porras, P., Wanker, E.E. and Andrade-Navarro, M.A. (2012) 'HIPPIE: Integrating Protein Interaction Networks with Experiment Based Quality Scores', *PLoS ONE* 7 (2), ISSN: 1932-6203, DOI: 10.1371/journal.pone.0031826, URL: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC3279424/ (visited on 08/04/2014).

Smedley, D., Haider, S., Ballester, B., Holland, R., London, D., Thorisson, G. and Kasprzyk, A. (2009) 'BioMart – biological queries made easy', en, *BMC Genomics* 10 (1), p. 22, ISSN: 1471-2164, DOI: 10.1186/1471-2164-10-22, URL: http://www.biomedcentral.com/1471-2164/10/22/abstract (visited on 29/07/2014).

Soler-López, M., Zanzoni, A., Lluís, R., Stelzl, U. and Aloy, P. (2011) 'Interactome mapping suggests new mechanistic details underlying Alzheimer's disease', en, *Genome Research* 21 (3), pp. 364–376, ISSN: 1088-9051, 1549-5469, DOI: 10.1101/gr.114280.110, URL: http://genome.cshlp.org/content/21/3/364 (visited on 14/08/2014).

Stark, C., Breitkreutz, B.-J., Reguly, T., Boucher, L., Breitkreutz, A. and Tyers, M. (2006) 'BioGRID: a general repository for interaction datasets', *Nucleic acids research* 34 (suppl 1), pp. D535–D539, URL: http://nar.oxfordjournals.org/content/34/suppl_1/D535.short (visited on 02/08/2014).

SynSys Project (2014) *synsys - A European expertise Network on building the synapse*, URL: http://www.synsys.eu/ (visited on 24/04/2014).

The IPython Development Team (2014) *Using IPython for parallel computing*, URL: http://ipython.org/ipython-doc/2/parallel/parallel_intro.html (visited on 13/08/2014).

The UniProt Consortium (2008) 'The Universal Protein Resource (UniProt)', en, *Nucleic Acids Research* 36 (suppl 1), pp. D190–D195, ISSN: 0305-1048, 1362-4962, DOI: 10.1093/nar/gkm895, URL: http://nar.oxfordjournals.org/content/36/suppl_1/D190 (visited on 29/07/2014).

Uetz, P., Giot, L., Cagney, G., Mansfield, T.A., Judson, R.S., Knight, J.R., Lockshon, D., Narayan, V., Srinivasan, M., Pochart, P., Qureshi-Emili, A., Li, Y., Godwin, B., Conover, D., Kalbfleisch, T., Vijayadamodar, G., Yang, M., Johnston, M., Fields, S. and Rothberg, J.M. (2000) 'A comprehensive analysis of protein–protein interactions in Saccharomyces cerevisiae', en, *Nature* 403 (6770), pp. 623–627, ISSN: 0028-0836, DOI: 10.1038/35001009, URL: http://www.nature.com/nature/journal/v403/n6770/abs/403623a0.html (visited on 25/07/2014).

Van der Maaten, L. and Hinton, G. (2008) 'Visualizing data using t-SNE', *Journal of Machine Learning Research* 9 (2579-2605), p. 85, URL: http://siplab.tudelft.nl/sites/default/files/vandermaaten08a.pdf (visited on 08/08/2014).

Von Mering, C., Jensen, L.J., Snel, B., Hooper, S.D., Krupp, M., Foglierini, M., Jouffre, N., Huynen, M.A. and Bork, P. (2005) 'STRING: known and predicted protein-protein associations, integrated and transferred across organisms', *Nucleic Acids Research* 33 (Database issue), pp. D433–D437, ISSN: 0305-1048, DOI: 10.1093/nar/gki005, URL: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC539959/ (visited on 10/04/2014).

Wang, J., Li, M., Deng, Y. and Pan, Y. (2010) 'Recent advances in clustering methods for protein interaction networks', en, *BMC Genomics* 11 (Suppl 3), S10, ISSN: 1471-2164, DOI: 10.1186/1471-2164-11-S3-S10, URL: http://www.biomedcentral.com/1471-2164/11/S3/S10/abstract (visited on 20/03/2014).

Wikimedia Commons (2013) *File:Simple decision tree.svg*, URL: http://commons.wikimedia.org/wiki/File:Simple_decision_tree.svg.

Witten, I.H., Frank, E. and Hall, M.A. (2011) *Data Mining: Practical Machine Learning Tools and Techniques: Practical Machine Learning Tools and Techniques*, en, Elsevier, ISBN: 9780080890364.

Xenarios, I., Salwinski, L., Duan, X.J., Higney, P., Kim, S.-M. and Eisenberg, D. (2002) 'DIP, the Database of Interacting Proteins: a research tool for studying cellular networks of protein interactions', *Nucleic Acids Research* 30 (1), pp. 303–305, ISSN: 0305-1048, URL: http://www.ncbi.nlm.nih.gov/pmc/articles/PMC99070/ (visited on 10/04/2014).

Yu, H., Braun, P., Yıldırım, M.A., Lemmens, I., Venkatesan, K., Sahalie, J., Hirozane-Kishikawa, T., Gebreab, F., Li, N., Simonis, N., Hao, T., Rual, J.-F., Dricot, A., Vazquez, A., Murray, R.R., Simon, C., Tardivo, L., Tam, S., Svrzikapa, N., Fan, C., Smet, A.-S.d., Motyl, A., Hudson, M.E., Park, J., Xin, X., Cusick, M.E., Moore, T., Boone, C., Snyder, M., Roth, F.P., Barabási, A.-L., Tavernier, J., Hill, D.E. and Vidal, M. (2008) 'High-Quality Binary Protein Interaction Map of the Yeast Interactome Network', en, *Science* 322 (5898), pp. 104–110, ISSN: 0036-8075, 1095-9203, DOI: 10.1126/science.1158684, URL: http://www.sciencemag.org/content/322/5898/104 (visited on 14/08/2014).

Zeeberg, B.R., Riss, J., Kane, D.W., Bussey, K.J., Uchio, E., Linehan, W.M., Barrett, J.C. and Weinstein, J.N. (2004) 'Mistaken Identifiers: Gene name errors can be introduced inadvertently when using Excel in bioinformatics', en, *BMC Bioinformatics* 5 (1), p. 80, ISSN: 1471-2105, DOI: 10.1186/1471-2105-5-80, URL: http://www.biomedcentral.com/1471-2105/5/80/abstract (visited on 14/08/2014).