Boosting Gene Expression Clustering with System-Wide Biological Information and Deep Learning

by

Hongzhu Cui

A Thesis

Submitted to the Faculty

of the

WORCESTER POLYTECHNIC INSTITUTE

In partial fulfillment of the requirements for the

Degree of Master of Science

in

Computer Science

by

____

May 2019

APPROVED:

____

Professor Dmitry Korkin, Major Thesis Advisor

____

Professor Carolina Ruiz, Second Thesis Reader

Abstract

Gene expression analysis provides genome-wide insights into the transcriptional activity of a cell. One of the first computational steps in exploration and analysis of the gene expression data is clustering. With a number of standard clustering methods routinely used, most of the methods do not take prior biological information into account. Here, we propose a new approach for gene expression clustering analysis. The approach benefits from a new deep learning architecture, Robust Autoencoder, which provides a more accurate high-level representation of the feature sets, and from incorporating prior system-wide biological information into the clustering process. We tested our approach on two gene expression datasets and compared the performance with two widely used clustering methods, hierarchical clustering and k-means, and with a recent deep learning clustering approach. Our approach outperformed all other clustering methods on the labeled yeast gene expression dataset. Furthermore, we showed that it is better in identifying the functionally common clusters than k-means on the unlabeled human gene expression dataset. The results demonstrate that our new deep learning architecture can generalize well the specific properties of gene expression profiles. Furthermore, the results confirm our hypothesis that the prior biological network knowledge is helpful in the gene expression clustering.

Acknowledgements

I would like to extend my sincere gratitude to my thesis advisor Professor Dmitry Korkin. He is a respectful and resourceful scholar. He has also exemplified to me what entails to do academic research: hard work, open-mindedness, scientific attitude, and perseverance. These have become invaluable assets in my life.

I owe my gratitude to my thesis reader, Professor Carolina Ruiz, for her help, patience and encouragement, which support me keeping improving this thesis.

A special thanks to my fellow Korkin lab members over the years, including but not limited to: Nan Zhao, Andi Dhroso, Pavel Terentiev, Nathan Johnson, Suhas Srinivasan, Oleksandr Narykov, Ana Leshchyk, Nan Hu. You all have been an invaluable resource and good friends.

Finally, I would like to thank my parents. I thank them for their unconditional love and support, and all the precious values they taught me that have become part of my life.

Contents

1. **Introduction** 6

2. **Background and Related Work** 9

   2.1 Problem Formulation 9
   2.2 Conventional Methods for Gene Expression Clustering Analysis 10
   2.3 PCA and Robust PCA 11
   2.4 Stacked Denoising Auto-encoder 13
   2.5 Module Detection in Biological Network 14

3. **Proposed Computational Solution** 16

   3.1 Proposed Deep Architecture to Regenerate Gene Expression Profile 18
   3.2 Community Detection Algorithm 20
   3.3 Network Based Similarity Measure and Proper Weighting Strategy 21
   3.4 Improved Agglomerative Clustering as Last Step 22

4. **Experimental Protocol** 23

   4.1 Gene Expression Datasets 23
   4.2 Construction of the Protein-protein Interaction Networks 23
   4.3 Selecting Baseline Methods Compared against Our Protocol 24
   4.4 Evaluation Criteria for Yeast Gene Expression Datasets 25
   4.5 GO Enrichment Analysis for Human Gene Expression Dataset 26

5. **Results** 28

   5.1 Two interactomes and their corresponding community structures 28
   5.2 Incorporating prior biological network information and weighting strategy ... 30
   5.3 Evaluation of our protocol on the Yeast gene expression dataset 33
   5.4 Evaluation of our protocol on the human gene expression dataset 34

6. **Conclusion and Future Work** 37

6.1 Final Conclusion 37
6.2 Future Work 38

Chapter 1

Introduction

Gene expression quantification and analysis using DNA microarrays, RNA sequencing (RNA-Seq), and other methods [1-3] have been proved to be an exceptionally powerful tool to quantitatively study the relationships among sets of genes. Global gene expression analysis provides quantitative information about the protein and mRNA abundance across the whole organism and in the individual tissues and cells [4], allowing to explore a wide range of biological processes [5]. Capturing the gene expression patterns can help studying molecular mechanisms implicated in diseases and cellular responses to drug treatment, thus facilitating drug discovery and development [4]. Global analysis of the gene expression data has been carried out by a number of supervised and unsupervised machine learning methods [6, 7]. An intuitive approach to analysis of the massive volumes of expression data is to first group the genes into smaller subsets based on common expression patterns they share, and without any preliminary knowledge of what each of these groups should include. Unsupervised learning, or clustering, methods are well-suited to address this problem [8].

Until recent, clustering of the genes expression data has been commonly carried out using the classical unsupervised learning methods, such as k-means or Expectation Maximization (EM) algorithms [9, 10]. At the same time, deep learning has made great strides in advancing both supervised and unsupervised learning, becoming routine methods in image recognition [11], natural language processing [12], and most recently in bioinformatics and genomics [13, 14]. Autoencoder is one of the commonly used deep architectures, and it has been proven successful to learn low-dimensional representations of biological data [15]. However, an autoencoder is sensitive to the outliers, which are widely present in the gene expression data. As a result, this may affect the generalization patterns uncovered by such architecture. Furthermore, most of the current clustering methods do not take into account the prior biological information that could guide the clustering procedure.

In the past decade, substantial improvements have been made in utilizing high-throughput "-omics" to map most components of cellular networks [16, 17]. Among them, human protein interactome and its edgotyping studies have attracted major attention [18]. Network properties of the interactome have provided insights into the system-wide biological properties and the interactome evolution [19, 20]. Of special interest is a property that is also found in many real-world networks, the community structure [21], in which the network nodes are joined together in tightly knit groups, while the groups themselves are only loosely connected with each other. One of the key ideas behind our work is incorporating the gene community information for the tested gene sets into the clustering process; we expect that such information would improve the clustering accuracy.

Here, we propose a novel protocol, which combines a new deep architecture with the prior biological knowledge for gene expression clustering analysis. Our protocol could be divided into two main stages. First, we use a deep network to learn important characteristics of the gene expression profiles. We leverage a new autoencoder method, Robust autoencoder [22]. The approach is designed to extract more robust features from the input data. Once the network is trained, the low dimensional representation of the gene expression profile is used for the clustering task. In the second stage, we define a network-based metric which allows introducing the community information of each gene in the network into our clustering process. The hypothesis behind this idea is that, if two genes are in the same network community, then they are more likely to communicate with each other and share the same expression pattern. Our new clustering protocol is based on the Eisen clustering [23].

We evaluated our method on two distinct gene expression datasets, one with external labels and the other one unlabeled. Specifically, we compared the performance of our method for gene expression clustering with two traditional clustering methods that are commonly used for the gene expression analysis, k-means and hierarchical clustering. We found that our method outperformed the traditional clustering methods on both labeled and unlabeled datasets. Furthermore, the proposed approach was more accurate than a deep learning autoencoder method. The results demonstrate that the new deep architecture could capture the high-level features from the gene expression profiles. Furthermore, the results confirm our hypothesis that the prior biological network knowledge could be utilized for optimizing the gene expression clustering task.

Chapter 2

Background and Related Work

2.1 Problem Formulation

With the tremendous advancement of RNA-Seq technology[2], as well microarray technology[24], we are able to gather genome-wide expression data during important biological processes and across collections of related samples. Exploring the gene expression patterns can give us more in-depth insights and an enhanced understanding of biological processes and disease pathogenesis. However, due to the huge number of genes and the high dimension of the gene expression profile, it is very hard for human brains to comprehend the dataset and extract insights. This is where clustering come into play. Researchers and scientists applied various clustering techniques to address this challenge. The use of clustering techniques can help reveal natural structures and identify interesting patterns underlying the data[25].

The clustering tasks in our approach are carried-out using unsupervised learning methods. For a given similarity measure defined in an unsupervised learning method, the objects belonging to the same cluster are more similar to each other than to those ones from other clusters. In the case of gene expression data clustering, a cluster may contain a number of genes or samples with similar expression patterns. After the preprocessing stage, the data are presented as a matrix $X = \{ x_{ij} \}$. Each cell x_{ij} in the matrix stands for an expression level of gene i from sample j at a specific time point or in a specific condition

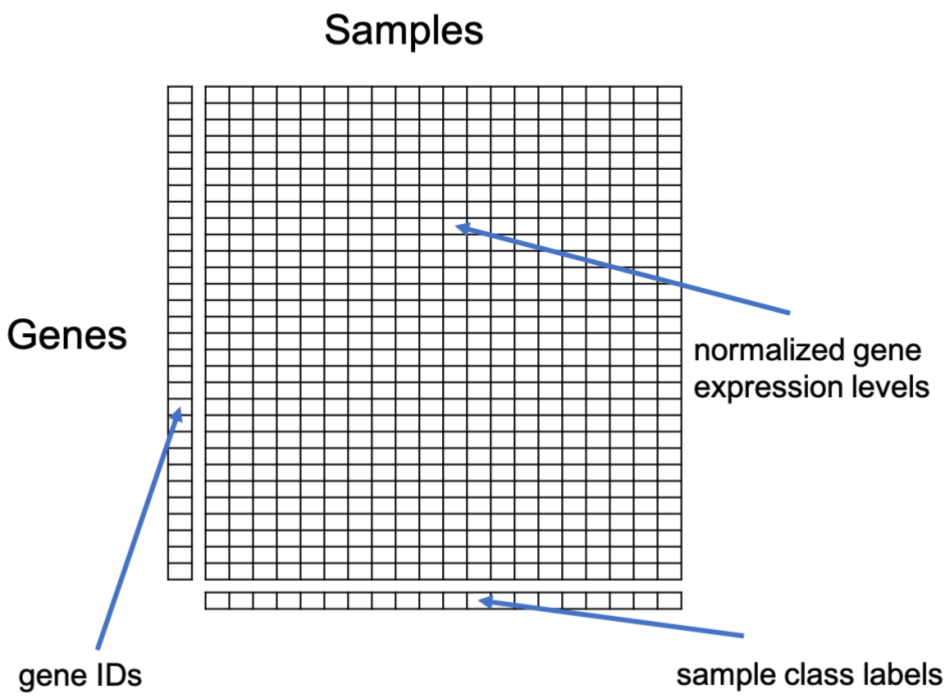


Figure 2.1: Illustration of gene expression clustering analysis. After pre-processing, the gene expression data is represented as a matrix. The goal of clustering is to subdivide a set of items in such a way that similar items fall into the same cluster, whereas dissimilar items fall in different clusters

(See Fig 2.1). The clustering of gene expression data can be divided into two main categories: gene-based clustering and sample-based clustering [26]. In this work, we focus on the gene-based clustering. The goal is to group genes with similar expression patterns (co-expressed genes). The expression patterns, in turn, will be used to help in our understanding of gene function, gene regulation, and cellular processes.

2.2 Conventional Methods for Gene Expression Clustering Analysis

There are two most important classes of clustering methods for gene expression analysis: partitioning-based methods and hierarchical clustering methods[8]. Partitioning methods divide the data into a predetermined number of clusters. A partitioning method constructs several partitions from the given data, with each partition representing a cluster. The k-means algorithm[9] is a typical partition-based clustering method. When we run k-means algorithms, we need to predefine a number k, which refers to the number of centroids in the dataset. A centroid is the imaginary location representing the center of the cluster. After that, each data point will be assigned to the closest centroid. After all data points are assigned with a cluster label, the positions of the k centroids are recalculated. This process will be repeated until the k centroids remain the same.

Hierarchical clustering methods[27] are of different philosophy compared against partition-based clustering. As suggested from the name, they produce a hierarchy of clusters. In hierarchical clustering, each cluster is subdivided into smaller clusters, forming a tree-shaped data structure or dendrogram. Hierarchical clustering methods generally fall into two types: agglomerative clustering and divisive clustering. In the context of gene expression clustering analysis, agglomerative hierarchical clustering starts with the single-gene clusters and successively joins the closest clusters until all genes have been joined into the supercluster[28]. Divisive clustering methods operate in the opposite way; all genes start in one cluster, and they are recursively spited into smaller clusters, as one moves down the hierarchy. A popular hierarchical clustering method was applied to analyze the first yeast gene expression data by Eisen et al[23]; hence it is often referred as 'Eisen clustering'.

2.3 PCA and Robust PCA

Because of the huge number of genes measured at the same time point and the complexity of biological processes, there is an urgent need to develop analytical methodology to reduce the dimension of gene expression data and make the analysis more manageable. Some classical techniques, such as principal component analysis (PCA), have been applied to analyze gene expression data. PCA[29] is a classic orthogonal linear transformation. It transforms the data to a new coordinate system such that the greatest variance will be reserved. Linear PCA projects data onto a linear manifold in high dimensional space. However, this classic linear transformation is not ideal for discovering non-linear representations. The complexity and variability of many real-world problems naturally require non-linear methods. In many real-world problems, non-linearity and outliers exist at the same time. Typically, PCA does not work well when these outliers exist[30]. The linear manifold of PCA will shift to offset the huge errors of those faraway outliers. This shifting will harm the information preservation for those normal observations. This distracted manifold has a large reconstruction error for all other observations. Eliminating the influence of those outliers is needed.

One shortcoming of PCA is its sensitivity to significant corruptions and outlying observations. Robust Principal Component Analysis (RPCA)[31] splits a raw input matrix X into a low-rank matrix L_o and a sparse matrix S_o:

$$X = L_0 + S_0$$

The low-rank matrix L_o contains our interested pattern and the sparse matrix S_o consists of element-wise outlying parts which cannot be captured by low-rank pattern L_o. We constrain the rank of matrix L_o as low as possible and the sparse matrix S_o element-wisely as sparse as possible. The L_o could be represented by a linear manifold, while the S_o is a filter that peels the faraway part from the linear manifold. (See Fig2.2) RPCA allows for the careful removal of sparse outliers, so that the remaining low-rank approximation is faithful to the true low-rank subspace describing the raw data. In short, Robust principal component analysis (RPCA) refines PCA by making PCA robust to outliers.

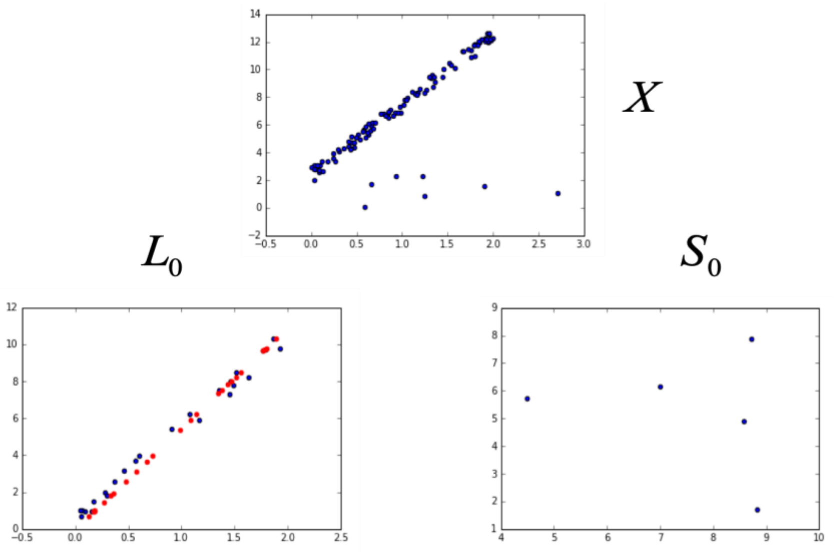


Figure 2.2: Illustration of RPCA. The input data can be decomposed into two parts. L_o is the low rank matrix and S_o is the sparse matrix. RPCA allows for the careful teasing apart of sparse outliers so that the remaining low-rank approximation is faithful to the true low-rank subspace describing the raw data.

2.4 Stacked Denoising Auto-encoder

Recent years have witnessed the power of deep learning on a wide range of application[13, 32]. One advantage about deep learning is that it could learn a hierarchical representation of the data through multiple layers of abstraction. Autoencoder is one of the most widely used deep architectures. Specifically, an autoencoder is a feed forward multi-layer neural network in which the output target is the input itself. An auto-encoder is trained to copy an input to its output. This process seems trivial, but the meaningful part is the dimension-reduced hidden layers learned to reproduce the input and thus these low dimensional hidden layers are trained to be lowest loss representations of the input. From the perspective of dimensionality deduction, auto-encoder is a generalized framework for non-linear dimension reduction process by applying non-linear activating function in encoder and decoder. In other words, auto-coder could project the original data in the high dimensional space to non-linear manifold in lower dimensional space.

The denoising autoencoder model[33] is a popular deep learning architecture and can be viewed as a stochastic version of the autoencoder. It randomly corrupts the input data and trains the parameters to recover the uncorrupted data from the corrupted one. Denoising autoencoders can be stacked to form a deep network, i.e. stacked denoising autoencoder[34]. (See Fig 2.3) The denoising autoencoder's goal is to learn the mapping from the corrupted data to the original uncorrupted data. One of the method's caveats is that it still needs the information about the original uncorrupted data for the training. Since the original, uncorrupted, data present the crucial prior knowledge for denoising autoencoder, the quality of the original data will influence the denoising autoencoder's map building and the quality of discovered features. If the original input contains outliers, denoising autoencoder's training will still learn to recover these outlying parts and the quality of discovered features could be misled by these outlying parts.

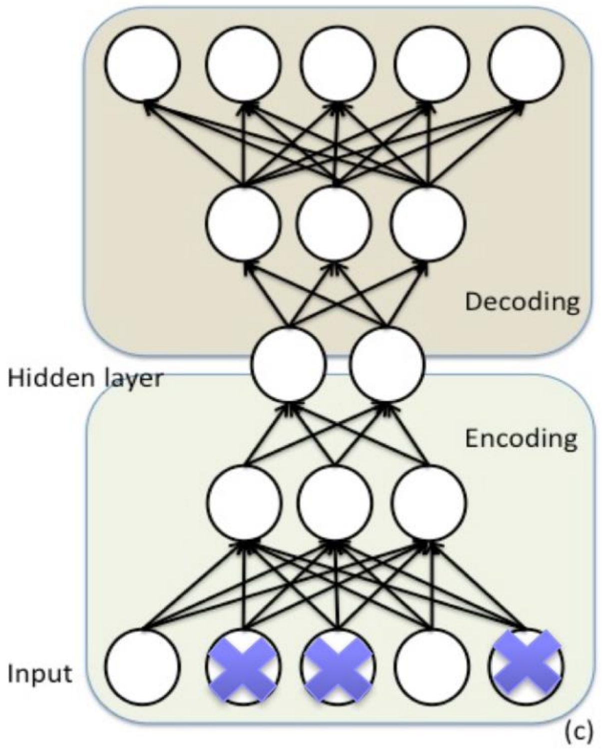


Figure 2.3: Architecture of Stacked Denoising Autoencoder. Denoising autoencoder randomly corrupts the input data and trains the parameters to recover the uncorrupted data from the corrupted one. Denoising autoencoders can be stacked to form a deep network.

2.5 Module Detection in Biological Network

Community structure could be viewed as a subnetwork of nodes that are more densely connected compared to the parts of the network[35]. It is a common characteristic in many physical networks, including the Internet and World Wide Web[36], social networks[37], and different kinds of biomolecular networks [21]. (See Fig2.4) Physiological and disease processes are typically not driven by a single gene, but a group of genes that interact within molecular modules or pathways in the context of complex biological network. Identification of such modules in gene or protein networks is at the core of many current analysis methods in biomedical research. Nowadays, it is generally accepted that biological networks are not randomly connected but follow certain structural patterns[38-40]. Among these structural patterns, modularity is one of the most important features of biological networks. By modularity, we mean that nodes are

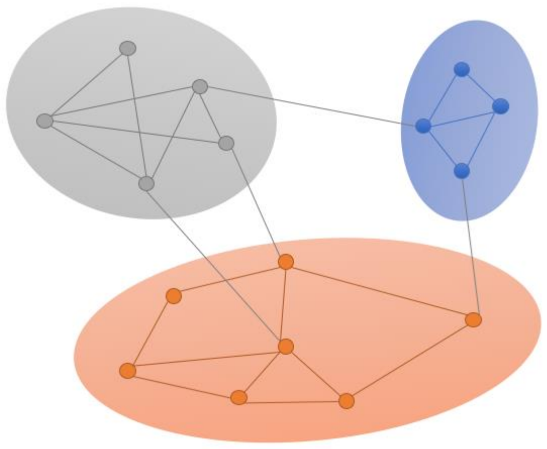


Figure 2.4 Community structure in the network. A network community is a set of network nodes, which are densely connected internally.

tightly connected with each other as a community, while having less connections with outside world. The general problem of identifying the functional modules in a biological network by relying exclusively on the network's topology is a challenging one due to the lack of information about specific genes/proteins contributing to the topological features of the network.

Chapter 3

Proposed Computational Solution

In this work, we propose a novel protocol, which combines deep architectures and prior biological knowledge for gene expression clustering analysis. Our protocol could be divided into two main stages. First, we use a deep network to learn important characteristics of the gene expression profiles. We leverage a new autoencoder method, Robust autoencoder[41]. The approach is designed to extract more robust features from the input data. Once the network has been trained, we use the low dimensional representation of gene expression profile for later clustering task. In the second stage, we introduce a network-based metric, which could signal the community information of each gene in the network, into our clustering process. (See Fig 3.1)

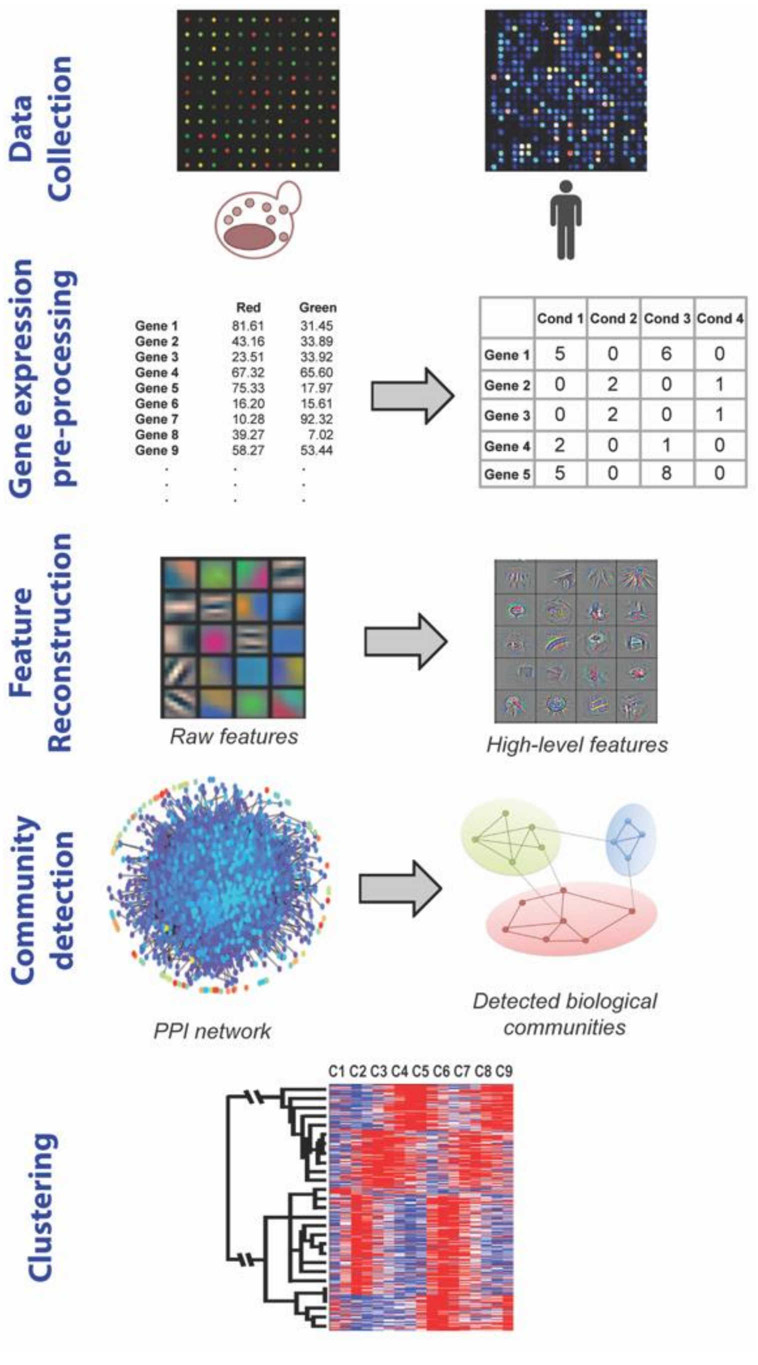


Figure 3.1: General workflow of our protocol. After acquiring the raw gene expression data. Our method consists of four basic steps: input data pre-processing, feature reconstruction using deep architectures, detecting community structure from the network, and incorporating gene network community information into clustering. The two datasets used in this study are gene expression dataset for the yeast cell cycle and human gene expression data from the genomics of drug sensitivity in cancer study.

3.1 Proposed Deep Architecture to Regenerate Gene Expression Profile

Our deep learning approach to gene expression clustering is driven by its ability to learn a hierarchical representation of the data through multiple layers of abstraction. In this work, we propose to apply our newly developed Robust autoencoder method[41]. The method improves the basic deep learning autoencoder model by building an outlier filter on top of a standard autoencoder, an idea that was inspired by the Robust Principal Component Analysis (RPCA) [42].

To simultaneously address the problems of outliers and non-linearity, we integrate the basic ideas of Robust PCA into the autoencoder model. In the Robust autoencoder approach, we introduce a filter layer before a normal autoencoder (Fig. 3.2). The filter layer culls out the outlying parts that are difficult to reconstruct by the autoencoder. Thus, the outlier filter introduces robustness, while the autoencoder provides nonlinearity. The low dimensional representation learned by the autoencoder is defined by the compressed features that reflect the trend of the observation majority. Similar to Robust PCA, we decompose our input data X into two parts: $X = L_D + S$, where L_D is a matrix that can be represented by a non-linear manifold, and S represents the outliers which will corrupt and skew the non-linear manifold. By peeling off the outliers from X into S, the autoencoder could perfectly recover the remaining L_D. Our loss function for a given input X is defined as:

$$argmin_{W,b,S} \left\| L_D - D_{W,b}\left(E_{W,b}(L_D)\right)\right\|_2 + \lambda \|S\|_1, \ such\ that \ \ X - L_D - S = 0,$$

where $E_{W,b}$ denotes an encoder function, $D_{W,b}$ denotes a decoder function, W is a projection matrix, b is the bias term, and λ is a balancing parameter to tuning the power of sparsity. We feed L_D as the input data to a standard deep autoencoder to learn the low-dimensional representations. The autoencoder is trained through minimizing the reconstruction error $\left\| L_D - D_{W,b}\left(E_{W,b}(L_D)\right)\right\|$. The minimized reconstruction error indicates that L_D can be projected to a low-dimensional nonlinear manifold without significant information loss. S contains all outlying observations, which have high reconstruction errors and cannot be interpreted by the majority observations. We require S to be sparse because we want the autoencoder to capture the trend of the majority of observations, while the outliers are expected to be rare. When minimizing the first term, we want the input of the autoencoder L_D to be perfectly reconstructed. Thus, we need to move more observations to S. Similarly, when minimizing the second term, S will contain the increasingly smaller number of the non-zero elements. Sparsifying the outlier filter S leaves more errors to L_D, and the reconstruction task of autoencoder becomes harder. In this optimization, L_D and S are mutually influenced by the constraint $X - L_D - S = 0$. The λ is the tuning parameter, which balances the impact of two optimizers. After training the whole model, the matrix S contains point-wise outliers, and L_D should retain the majority of information about X inside the hidden layer.

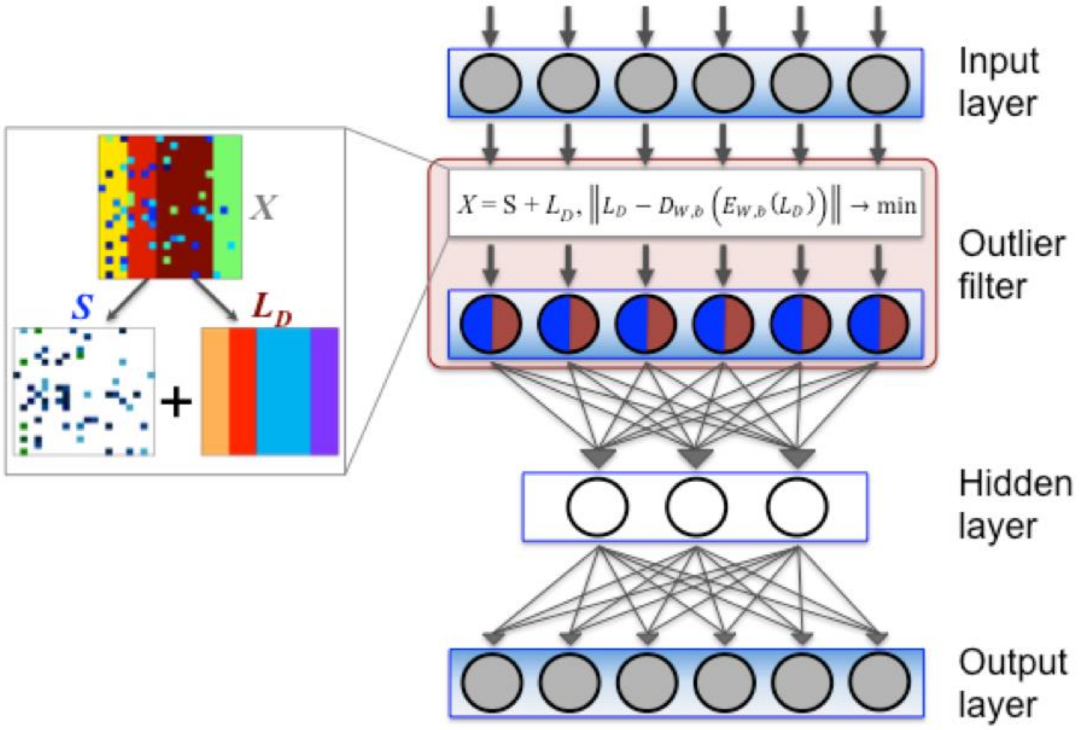


Figure 3.2: Architecture of robust autoencoder. In Robust autoencoder approach, an outlier filter layer before a normal autoencoder is introduced, providing robustness, while the autoencoder provides nonlinearity. We decompose the input data X into two parts: L_D, a matrix representing by a non-linear manifold, and S, a matrix representing the outliers which will corrupt and skew the non-linear manifold. The goal is to filter out the outliers from X, thus recovering L_D.

We solve the minimization problem of Robust autoencoder using an approach similar to [22]. While individual optimization techniques exist for training an autoencoder or Robust PCA (*e.g.*, alternating direction method of multipliers, ADMM algorithm [43]), to the best of our knowledge no methods previously existed that could simultaneously optimize both. In [22] , the authors train the autoencoder using back-propagation and the outlier filter using the shrinkage function. Back-propagation is an essential element of the deep autoencoder training, but it requires the objective function to be smooth to take advantage of chain rule of differentiation. This is not the case in our problem, since the second term in our objective function, $\|S\|_1$, is not smooth or differentiable. However, in [22] they solved this problem using a refined method is based on the basic idea of ADMM algorithm. The original objective function is broken into two smaller pieces, each of which is then easier to handle, where (1) a back-propagation algorithm is used to minimize the reconstruction cost of an autoencoder $\left\|L_D - D_{W,b}\left(E_{W,b}(L_D)\right)\right\|$, and (2) a shrinkage function on $\|S\|_1$ is used to sparsify S with the fixed L_D. Then [22] borrow an idea from the alternating projection forcing both optimizers to obey the constraint.

3.2 Community Detection Algorithm

Determining these community structures in a network can provide insight into the structural and functional organization of the network and can be useful in improving graph algorithms, such as spectral clustering [21]. In a basic community detection setting, a network node is defined as belonging to at most one community. The majority of community detection methods adopt such simplification. In this paper, we resort to a widely used methods for community detection based on modularity maximization, the Louvain method [44]. Modularity, Q, measures the quality of a partition of the network into communities and is defined as:

$$Q = \sum_{s=1}^{m} \left[\frac{l_s}{|E|} - \left(\frac{d_s}{2|E|} \right)^2 \right]$$

for the overall network with $|E|$ edges that is partitioned into m communities, where l_s is the number of edges between the nodes belonging to the s-th community and d_s is the sum of the degrees of the nodes in the *s-th* community. The modularity maximization method detects communities by finding the network partitions that have particularly high modularity. Since the exhaustive search over all possible partitions is usually intractable, the Louvain Method leverages an approximate greedy optimization approach. Specifically, it iteratively optimizes local communities until the global modularity can no longer be improved, given perturbations to the current community state [44].

3.3 Network Based Similarity Measure and Proper Weighting Strategy

To identify genes that share similar patterns, a similarity (or dissimilarity) measure is required. However, most of the commonly used similarity/dissimilarity measures, such as Pearson correlation coefficient or Euclidean distance, do not take the prior biological information into account. In this work, we propose that such prior information on the biological network communities could be used to adjust the distance between the two gene expression profiles, thus improving the clustering performance. The weighting idea is based on a hypothesis that if two genes are a part of the same community, they are more likely to be joined via a direct or indirect interaction and hence share the same expression pattern. To achieve that, we introduce a new metric that is weighted by the community information of a gene pair in the PPI network. Specifically, for any two genes we check if these two genes are in the same community using the results of the above community detection algorithm. If they are in the same community, their original distance will be assigned a small weight with the effect of shortening the distance. Otherwise, their distance will be assigned a large weight, with the effect of elongating the distance.

The specific strategy of assigning a weight to the distance between a pair of genes is of critical importance. To derive this strategy, we take advantage on the yeast expression dataset (See subsection4.1 below for more details), whose external labels correspond to the 5 phases of cell cycles. By comparing the Adjusted Rand Indices (See subsection4.3 below for more details), one can systematically evaluate a spectrum of strategies with various magnitudes of the weights. Here, we evaluate 5×5=25 combinations of the following pairs of weights (w_k, w_m). The distance between a pair of genes is assigned a weight with one of the five values, w_k=0.6, 0.7, 0.8, 0.9, or 1.0, if the genes are in the same community, and a weight with one of the five values, w_m=1.0, 1.1, 1.2, 1.3, or 1.4, if the genes are not in the same community. The best performing weight combination will be integrated into our clustering approach.

3.4 Improved Agglomerative Clustering as Last Step

The gene expression data is first pre-processed using the standard data cleaning and normalization methods [45]. Then, our new approach is introduced in two main steps. First, we use Robust autoencoder to initialize deep architectures. Once Robust autoencoders generalize specific properties of the gene expression profiles, the intermediate representation serves as an input for the clustering task. For clustering, instead of applying traditional similarity measures, we adopt the biological network based measure defined above. The measure is based on the Pearson correlation coefficient, which could detect both positive and negative correlations and is scale invariant on centered data. The similarity measure is then implemented for the agglomerative hierarchical clustering [23]. The linkage criterion for the merge strategy in the agglomerative clustering procedure is the average linkage, which minimizes the average of the distances between all pairs of clusters.

Chapter 4

Experimental Protocol

4.1 Gene Expression Datasets

To test our approach on the real-world data, we used two distinct large-scale datasets. The first dataset includes gene expression for the yeast cell cycle [46]. It is organized in 17 time stamps for a set of 420 genes in yeast. Based on the gene functional categories, each gene was assigned to one or more "phases". We removed the gene expression profiles for the genes that were assigned to more than one phase, resulting in a subset of 384 genes that were partitioned into 5 phases of cell cycle. The yeast dataset is widely used in practice to assess the clustering quality using the five phases assignment as an external criterion [26, 47]. The second dataset is obtained from the Genomics of Drug Sensitivity in Cancer (GDSC) study [48]. The dataset captures the gene expression profiles of different human cancer cell lines in response to drug compounds. It consists of 17,419 genes expressed in 83 cell lines. Overall, these two datasets differ in several principal aspects. First, the datasets are of substantially different sizes. In addition, the first dataset is time series data, while the second dataset is from different cancer cell lines. Finally, the datasets come from two different species.

4.2 Construction of the Protein-protein Interaction Networks

To extract the community information for the gene set and link it to the expression data, we studied the protein products of these genes in the context of the physical protein-protein interaction (PPI) network. To this end, two PPI networks are used: HINT yeast network [49] and the human interactome project network (HI-II-14) [18]. HINT network is organized as a database of high-quality protein-protein interactions collected from several databases manually as well as using an automated protocol. The comprehensive coverage of the interactome makes it possible to fully understand the network properties of the yeast genes. The human interactome HI-II-14 is another recently released source of PPI data. It is constructed through mapping binary PPIs obtained by systematically interrogating all pairwise combinations of human proteins using yeast two-hybrid high-throughput experiments. For each network, we run the community detection algorithm and apply the extracted community information during clustering.

4.3 Selecting Baseline Methods Compared against Our Protocol

The gene expression data is first pre-processed using the standard data cleaning and normalization methods [45]. Then, our new approach is introduced in two main steps. First, we use Robust autoencoder to initialize deep architectures. Once Robust autoencoders generalize specific properties of the gene expression profiles, the intermediate representation serves as an input for the clustering task. For clustering, instead of applying traditional similarity measures, we adopt the biological network based measure defined above. The measure is based on the Pearson correlation coefficient, which could detect both positive and negative correlations and is scale invariant on centered data. The similarity measure is then implemented for the agglomerative hierarchical clustering [23]. The linkage criterion for the merge strategy in the agglomerative clustering procedure is the average linkage, which minimizes the average of the distances between all pairs of clusters.

In the past decade, hundreds of new clustering algorithms have been developed and applied to the gene expression data. However, the performance of each clustering algorithm relies on specific properties of the input dataset and their underlying assumptions. There is no agreement on the best performing clustering algorithm for all datasets [50]. Therefore, for the baseline methods, we only implement two most widely used clustering methods: Eisen clustering and k-means. In addition, we compared our new approach to a basic autoencoder based clustering similar to the one that have been recently used for clustering the microarray gene expression data [47]. By comparing the performance of our approach to these methods we test how much of improvement over the traditional clustering algorithms, if any, can an advanced clustering method achieve, and whether including prior biological information into the gene expression clustering analysis can further improve the clustering accuracy.

4.4 Evaluation Criteria for Yeast Gene Expression Datasets

First, we evaluate the clustering results against the reference partition for the yeast dataset, since the external labels for each gene are provided. Specifically, we use the Adjusted Rand Index (ARI) [51], a frequently used measure for cluster validation [51]. ARI quantifies the degree of agreement between two partitions: one given by the clustering algorithm and the other labeled by external criteria. For a partition U generated by the clustering algorithm and a reference partition V, ARI is calculated as:

$$\mathrm{ARI} = \frac{\binom{n}{2}(a+d) - [(a+b)(a+c)+(c+d)(b+d)]}{\binom{n}{2}^2 - [(a+b)(a+c)+(c+d)(b+d)]}$$

Here, n is the total number of samples; a is the number of gene pairs in the same cluster for both sets U and V; b is the number of gene pairs in the same cluster in U, but in different clusters in V; c is the number of gene pairs in the same cluster in V and in different clusters in U; and d is the number of gene pairs that are placed in different clusters for both, U and V. The value of ARI is defined to lie between 0 and 1, and a high score represents a good agreement between the clustering result and the reference partition. We computed the ARI scores for the clustering results using our protocol, and compared them with ARI scores obtained using the two baseline clustering methods and the basic autoencoder based clustering.

4.5 GO Enrichment Analysis for Human Gene Expression Dataset

In contrast to the yeast set, no external labels are given for the GDSC sets, and the ARI metric cannot be used. In this case, a different evaluation procedure is required. Thus, we evaluate the clustering results based on their agreement with the available biological knowledge, such as Gene Ontology [52]. Here, we apply the following evaluation protocol. First, for the GDSC dataset, we set the number of clusters to be 100. Next, since the baseline hierarchical clustering can result in many singleton clusters, we select 10 most populated clusters for the analysis. For each cluster, we perform gene enrichment analysis and obtain the corresponding list of enriched GO terms. In the GO enrichment analysis, we use the third level of the GO hierarchy and kept the GO terms with P-value ≤ 0.01. The third level represents a trade-off between having too general, but well-populated GO terms from the second level (*e.g.*, GO:0050789 regulation of biological process) and more specific but not well-populated terms from the fourth level, which cannot be used for the enrichment analysis. We compared our results for the two baseline methods. More specifically, we compared the p-values of the enriched GO terms existing for Robust autoencoder and at least one baseline method results. We expect that, for most of the significant GO terms, our protocol would output smaller p-values compared to either of the two baseline methods. These results would suggest that our protocol could identify more coherent clusters. The GO enrichment was performed using DAVID [53], and multiple testing correction was done via false discovery rate estimation.

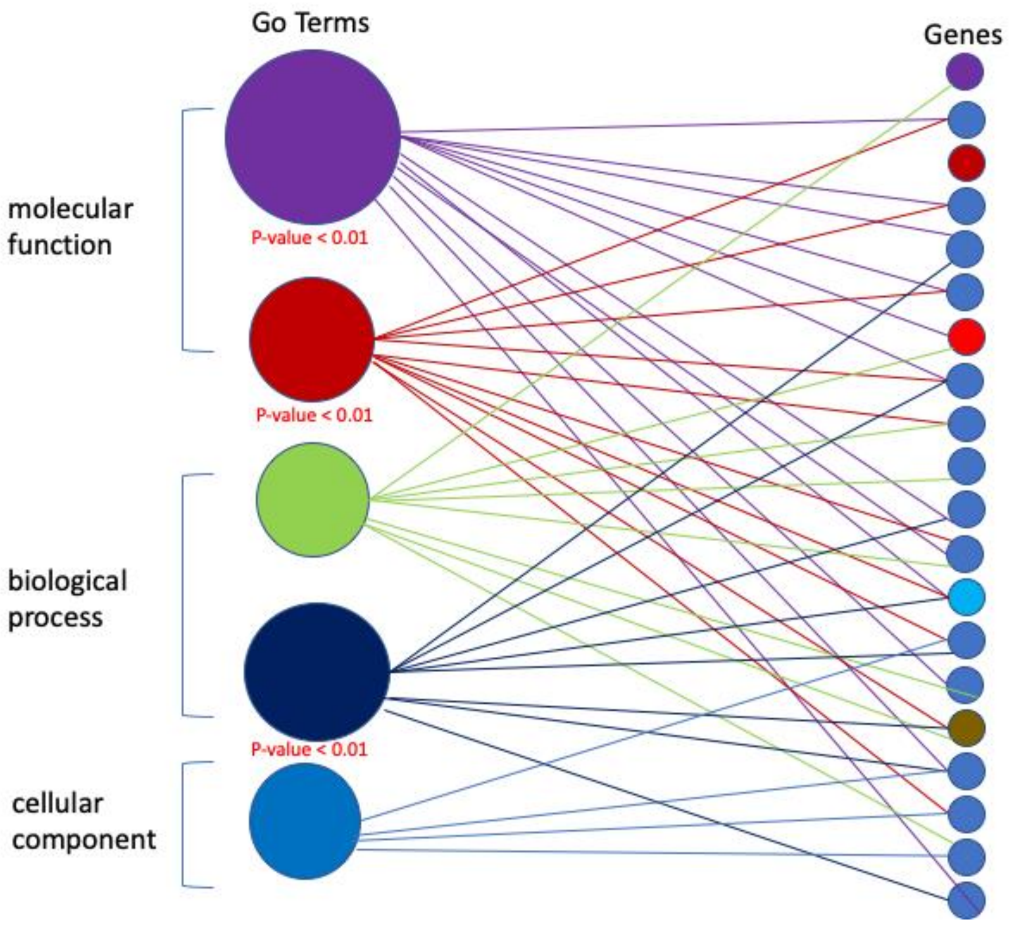


Figure 4.1: Basic idea of GO enrichment analysis. Gene Ontology system assign genes a set of predefined labels depending on their functional characteristics. GO term enrichment analysis is done by testing the input gene set against the background information to see which GO term is enriched for the input genes

Chapter 5

Results

5.1 Two interactomes and their corresponding community structures

Two PPI networks were extracted and analyzed, the yeast and human interactomes. For the yeast gene sets, we collected the PPI data from HINT database [49]. For the human interactome, we used the recently published interactome (referred to as HI-II-14 network [18]). Overall, HINT yeast network consisted of 5,687 proteins and 21,528 corresponding PPIs, while HI-II-14 network consisted of 11,787 genes and 32,465 corresponding PPIs (Table 1). A major giant component [54] existed in both interactomes, with several isolated sets of interactions on the periphery. Both interactomes shared the scale-free property [54], which means that most nodes in the network had only a few interactions and a few highly connected nodes (hubs) held the whole network together (Fig. 5.1, Figs. S1, S2 in Supplementary Data).

Table 1. The basic statistics between the two PPI network used in the evaluation protocol

	N of genes	*N of PPIs*	*N of communities*
HINT	5,687	21,528	81
HI-II-14	11,786	32,465	143

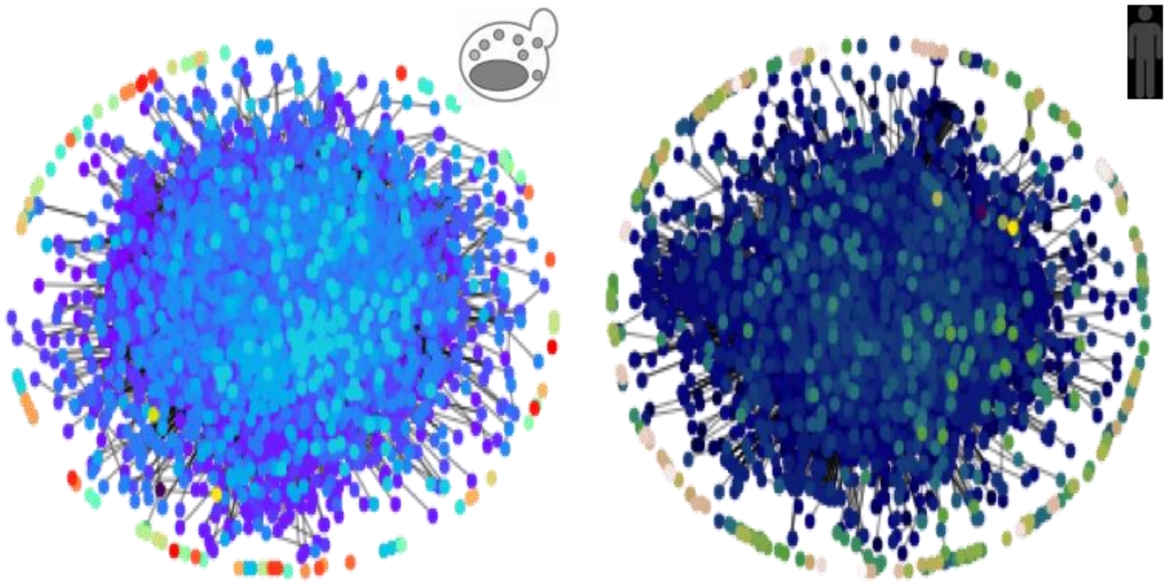


Figure 5.1: Visualization of two protein-protein interaction (PPI) networks. Two PPI networks used in this works are HINT yeast network (left) and the human interactome project network (right).

The detection of community structure played a critical role in our protocol. Once the interactome was constructed, we mapped the gene set to the interactome, determined which community they belonged to, and later used this information to weight the distance between any pair of gene expression profiles. We ran the Louvain method [44] on the two interactomes separately. After running the community detection algorithm on both networks, we obtained 81 and 143 communities from the yeast and human interactomes, correspondingly (Table 1, Fig. 5.2). The largest community in the yeast interactome was composed of 764 genes.

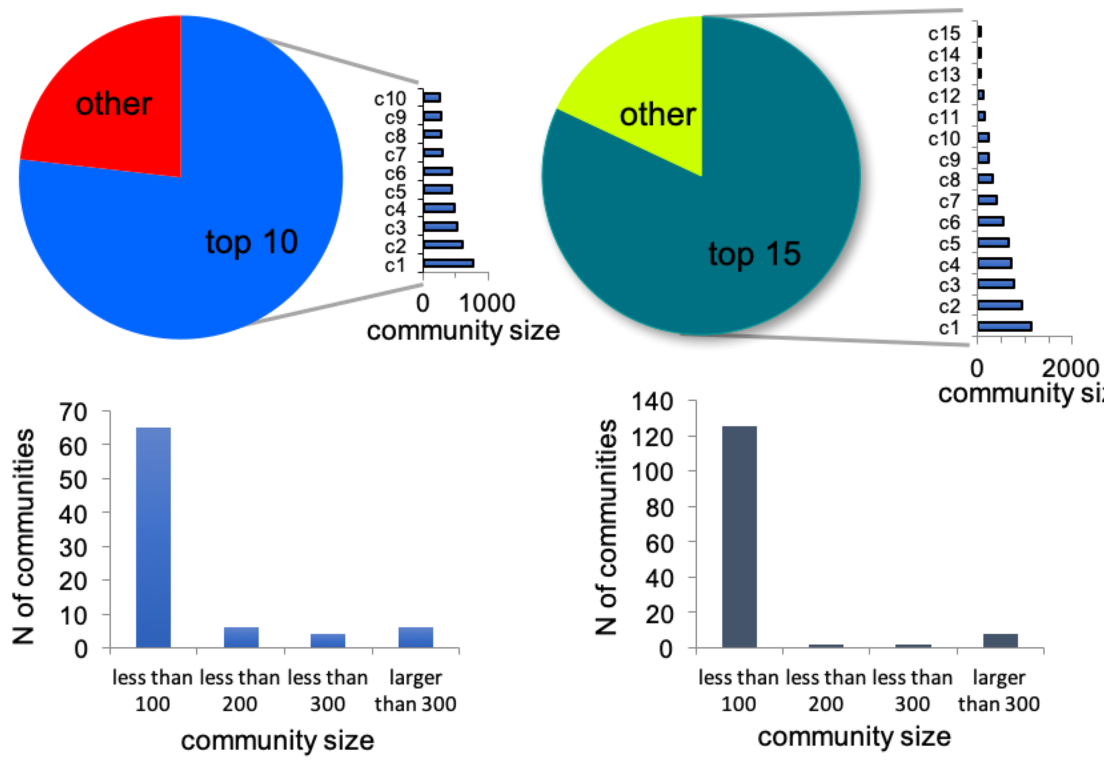


Figure 5.2: Community information from protein-protein interaction (PPI) networks. The networks share similarities in the size distribution of the largest communities (top 10 largest communities in yeast PPI network and top 15 communities in human PPI network, respectively, shown in the two pie charts). Furthermore, in both networks, communities with small numbers of nodes (<100) are predominant ones

The top 10 largest communities covered 77% of the total proteins in the network. The other communities were all composed of only few nodes. Similarly, to the yeast network, the first 15 communities accounted for 82% proteins in the human interactome, while the largest community contained 1,129 proteins (9.6%) (Fig 5.2).

5.2 Incorporating prior biological network information and weighting strategy

After the community detection stage, we examined every gene pair from the gene expression list to determine if they were in the same community. Then, we utilized this information to weight the distance between each pair of gene expression profiles. We compared the weighted clustering results with the baseline clustering results to demonstrate the effectiveness of incorporating network community information. For the baseline clustering methods, we implemented two most widely used approaches, k-means and hierarchical clustering. The two baseline methods were considered as the "un-weighted" clustering approaches. We then determined the optimized combinations of weights using a basic grid search on the hierarchical clustering method. Specifically, the search explored the weights from the range 0.6 to 1 (with a step of 0.1) for each pair of genes that were in the same community, and from the range 1.0 to 1.4 (with the same step) if the genes were not in the same community. The best performing combination was selected for our protocol.

The effectiveness of including the biological information was assessed on the labeled yeast gene expression dataset, since one could accurately evaluate the clustering performance only when the external labels were available. For each of the two baseline methods, we set the number of generated clusters to be five (matching the total number of different labels in the yeast dataset). Hierarchical clustering method performed with ARI of 0.448 on the yeast dataset, while k-means performed with ARI of 0.420. The ARI values after applying different weighting strategy ranged from 0.444 to 0.488 (Table 2). Overall, the accuracy after applying the weighting strategy was better compared to the un-weighted baseline methods. These results demonstrated that the biological network community information could be utilized to improve the traditional clustering. The results also supported the hypothesis that gene pairs in the same community of the PPI network are more likely to share the same expression pattern. Also, we note that the weight combination 0.9 and 1.3 yielded the most accurate results. Therefore, we adopted this weighting strategy for our protocol.

The denoising autoencoder model [33] is another popular deep learning architecture. Denoising autoencoder is mostly viewed as a randomized version of the autoencoder. Autoencoders are commonly used for feature selection and extraction. However, autoencoder could risk learning the so-called "Identity Function", which means that the output equals the input. Denoising Autoencoders solve this problem by randomly turning some of the input values to zero. In other words, it randomly corrupts the input data and trains the parameters to recover the uncorrupted data from the corrupted one. The denoising autoencoder's goal is to learn the mapping from the corrupted data to the original uncorrupted data. One of the method's caveats is that it still needs the information about the original uncorrupted data for the training. If the original input contains outliers, denoising autoencoder's training will still learn to recover these outlying parts and the quality of discovered features could be misled by these outlying parts.

In contrast, Robust autoencoder distinguishes the outliers from corrupted data without the knowledge of uncorrupted data. To illustrate that Robust autoencoder is a better choice than the denoising autoencoder for regenerating the gene expression profile, we applied both methods on the yeast expression dataset. We considered the individual effects of deep architecture on the clustering results, *i.e.*, without applying the community information to weight the distance in the protocol. For Robust autoencoder, the best ARI obtained across different hidden layer sizes was 0.5, whereas the highest ARI obtained for the denoising autoencoder was 0.48 (Fig. 5.3). Thus, our deep architecture performed better, although not significantly. We also noted that Robust autoencoder suffered from the greater variation of ARI values compared to denoising autoencoder.

Table 2. Comparison of results from different weighting strategies obtained when including the network community information.

		Weight for genes in the different community			
		1.1	**1.2**	**1.3**	**1.4**
Weight for genes in the same community	**0.6**	0.470	0.469	0.455	0.462
	0.7	0.480	0.473	0.475	0.448
	0.8	0.461	0.441	0.485	0.471
	0.9	0.444	0.474	**0.488**	0.474

5.3 Evaluation of our protocol on the Yeast gene expression dataset

In our protocol, instead of taking as an input for clustering the raw expression data, we reconstructed the features via Robust autoencoder and used this intermediate feature representation for clustering, so the best performing weight combination was not directly assigned to the raw dataset. To compare the results of our protocol with the baseline methods on the yeast dataset, the same ARI measure was calculated. The results showed that our protocol, which incorporates the prior biological information on the regenerated data from the deep architecture, outperformed the baseline methods applied to the raw data (Fig. 5.3, Fig. S4 in Supplementary Data). Furthermore, the results of our protocol outperform the baseline method with the used community information for the pairs of genes. This behavior is perhaps due to the ability of the architecture to learn important properties in the underlying input distribution. Also, we note that, compared against the results without incorporating biological information, the former clustering results had smaller variation of ARI values, suggesting that incorporating the prior biological information could stabilize the clustering process. Finally, we found that deep architecture does not guarantee that it will always perform better than the basic clustering methods. For instance, our deep architecture with hidden size of 5, the performance is comparable to the baseline methods. This implies that tuning parameters of deep architecture is a critical but not a simple step for these methods.

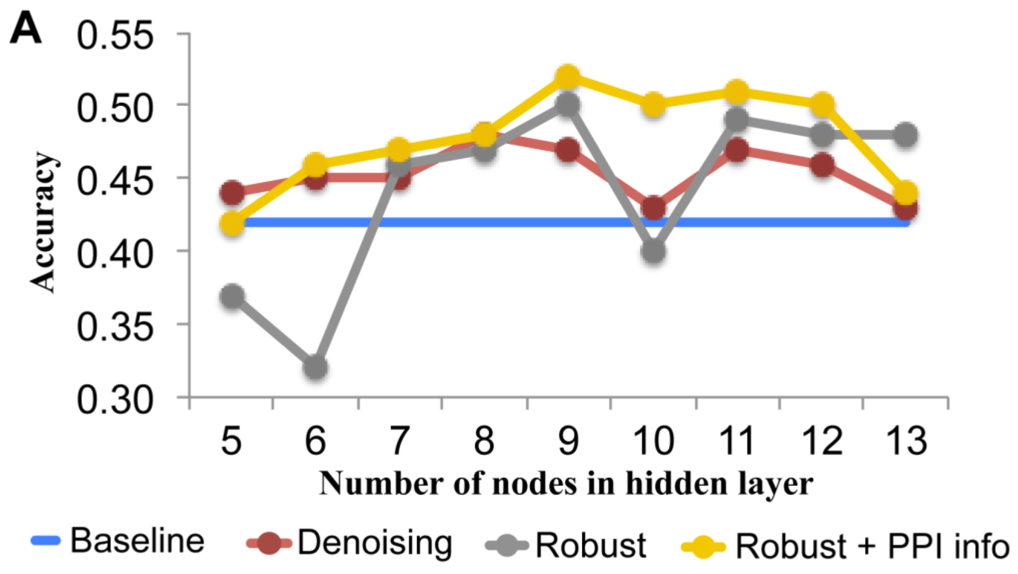


Figure 5.3: Evaluation of the new clustering approach. Comparison of the performance of two deep architectures against baseline methods performed on previously labeled yeast gene expression dataset. The accuracy measure used here is Adjusted Rand Index (ARI). Shown is the comparison of our approach that combines the Robust autoencoder architecture with the PPI network community information (yellow) against the base line K-means clustering method (blue), standard denoising autoencoder (red), and Robust autoencoder without additional biological information (grey).

5.4 Evaluation of our protocol on the human gene expression dataset

When implementing our protocol on the GDSC dataset, we used the results got from the Yeast dataset to guide the construction of the deep architecture. Specifically, we used a comparable percentage of the input layer size as in the best performing deep structure for the Yeast dataset to build the hidden layer. This led to a hidden layer with 55 nodes.

The human gene expression dataset consisted of 17,419 genes expressed in 83 cell lines. We independently applied our protocol as well as the k-means and hierarchical clustering methods on this gene set, while setting the cluster number in each case to be 100. Out of 100 clusters, we focused on the top 10 largest clusters and performed the GO enrichment analysis on these clusters. We only selected the third level GO terms in the GO hierarchy tree and compared the results against k-means and hierarchical clustering (Fig. 5.3, Fig. S3 and Tables S1, S2 in Supplementary Data). Comparing against k-means, 22 GO terms from the third level were enriched in at least one cluster in both cases, and most of the GO terms identified by our protocol had smaller *P*-values. This indicated that our protocol could group a more coherent and meaningful set of genes into a cluster. Compared against hierarchical clustering, we obtained 114 GO terms enriched in at least one cluster. In this case, the number of GO terms obtained in our approach (N_1=61) with smaller *P*-value was slightly larger than the number obtained in hierarchical clustering (N_1=53). This did not indicate that our protocol could significantly improve the traditional hierarchical clustering in terms of generating more coherent clusters. However, we noted another interesting observation. One main problem about hierarchical clustering is that it groups too many genes into a very large, giant, cluster. In this case, the largest cluster resulted from hierarchical clustering consisted of 11,043 genes, and its size was almost comparable to the first three largest clusters found by our protocol. This suggests that our protocol could compensate the inability of hierarchical clustering to further separate the clusters.

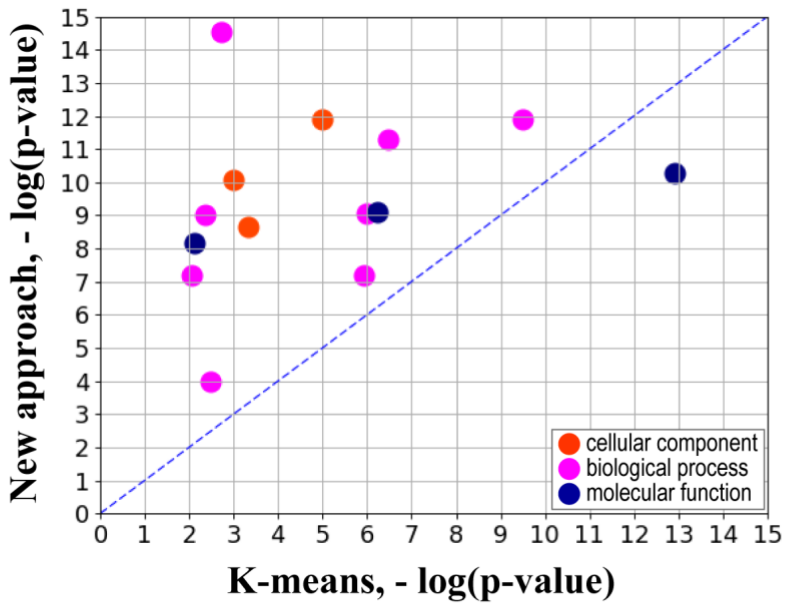


Figure 5.4: Comparison of enriched Gene Ontology terms between our approach and K-means for the human gene expression dataset. The values are converted using negative log of p-value function. A smaller p-value reflects a larger proportion of the cluster members sharing the same GO term.

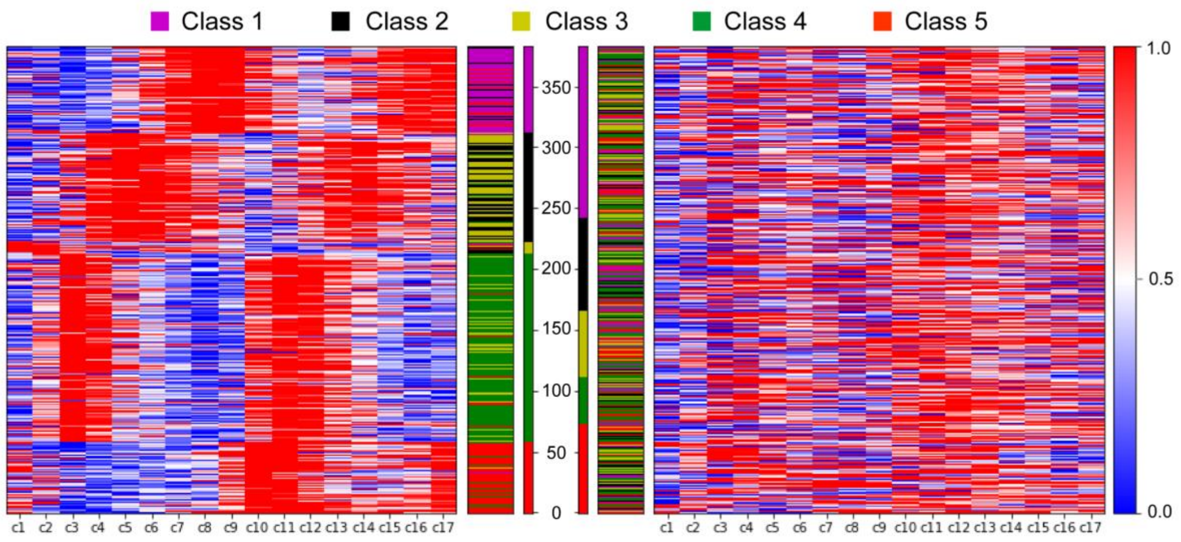


Figure 5.5: Performance of our approach against the base line K-means clustering (right) on the yeast gene expression dataset. The comparison of our approach (left) against the base line K-means clustering (right) provides a visibly better clustering into 5 previously labeled gene classes across 17 different time stamps (c1-c17).

Chapter 6

Conclusion and Future Work

6.1 Final Conclusion

In his paper, we present a proof-of-principle study where we integrate system-wide biological knowledge into the microarray-based gene expression clustering task by leveraging a novel deep learning architecture. We trained a Robust autoencoder to learn general patterns of the gene expression profiles. The obtained low dimensional representations of gene expression profiles were then used for the clustering task. To increase the clustering accuracy, the clustering algorithm employed a knowledge-based molecular network similarity measure. We compared the performance of our clustering approach with two widely used clustering methods, k-means and agglomerative hierarchical clustering. We selected these methods because of several reasons. First, these two methods have been arguably the most widely used in the gene expression analysis to date, with a wide range of applications and are considered the golden standard [55-59]. Furthermore, k-means has been consistently among the top performing clustering methods for gene expression data in recent comparative evaluation studies [55, 56]. Other methods for clustering gene expression data have been also recently introduced [60]. Having shown the superior performance of the deep learning paradigm over these traditional clustering approaches, our next step is to carry out a more comprehensive assessment of our approach by including other clustering methods and protocols.

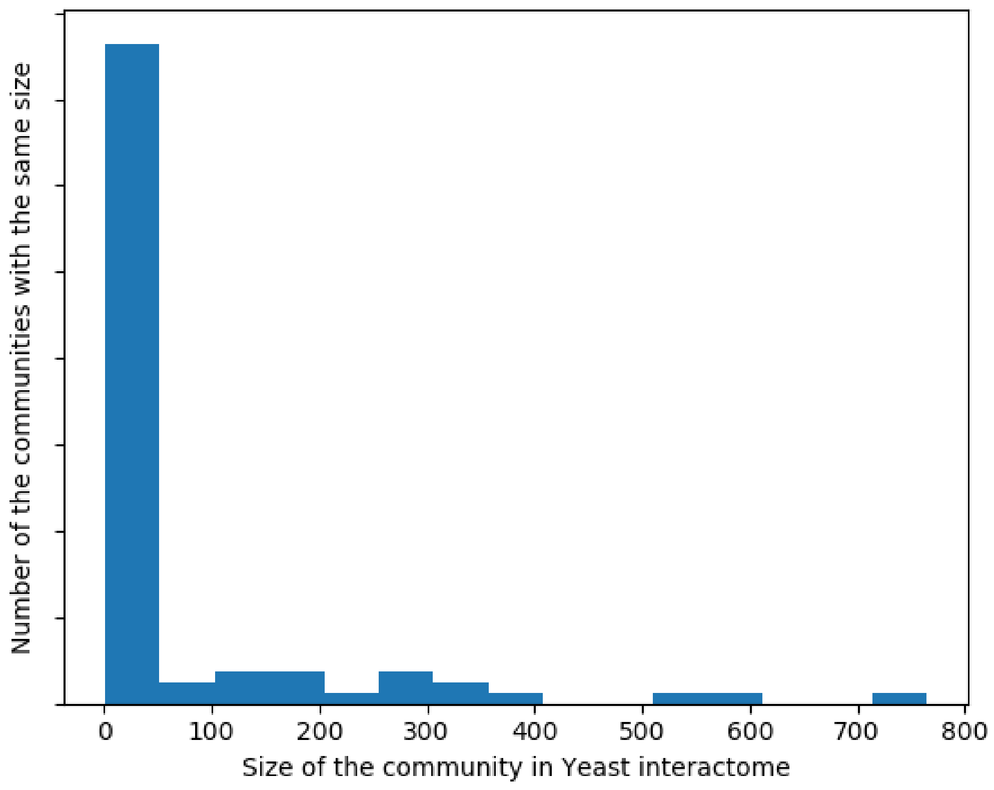
Another important aspect for the performance assessment is exploring multiple experimentally validated gene expression datasets to determine the tasks for which our approach will be most useful. In this work, we have explored two large-scale datasets from different species, each carrying different expression patterns: gene co-expression in different stages of yeast cell cycle and common response to cancer drug compounds. Many other interesting datasets also focus on the cancer-related data or cell cycle data [56], while others include [55, 61]. One limitation of our approach is in its requirement of the large-scale interactomics data, which is currently available only for a handful of species [18, 62, 63]. The human interactome is arguably the most well-studied protein-protein interaction system, making our approach applicable to a large number of disease-associated expression dataset.

Our results demonstrate the effectiveness of using (i) deep networks and (ii) prior biological information for the gene expression clustering analysis. Several other conclusions have been made from this work. First, we used a fairly simple deep learning architecture because of the long computation time. In future work, we plan to adopt a much deeper architecture. An autoencoder with a single encoder and decoder is usually considered as a shallow model. The way of extending shallow autoencoder to deep autoencoders is to add more encoding and decoding phases. A typical implementation of this idea is the stacked autoencoders [33]. The same idea could be applied to the Robust autoencoder model presented here. To address the problem of computational overhead, one can resort to the GPU computing algorithms.

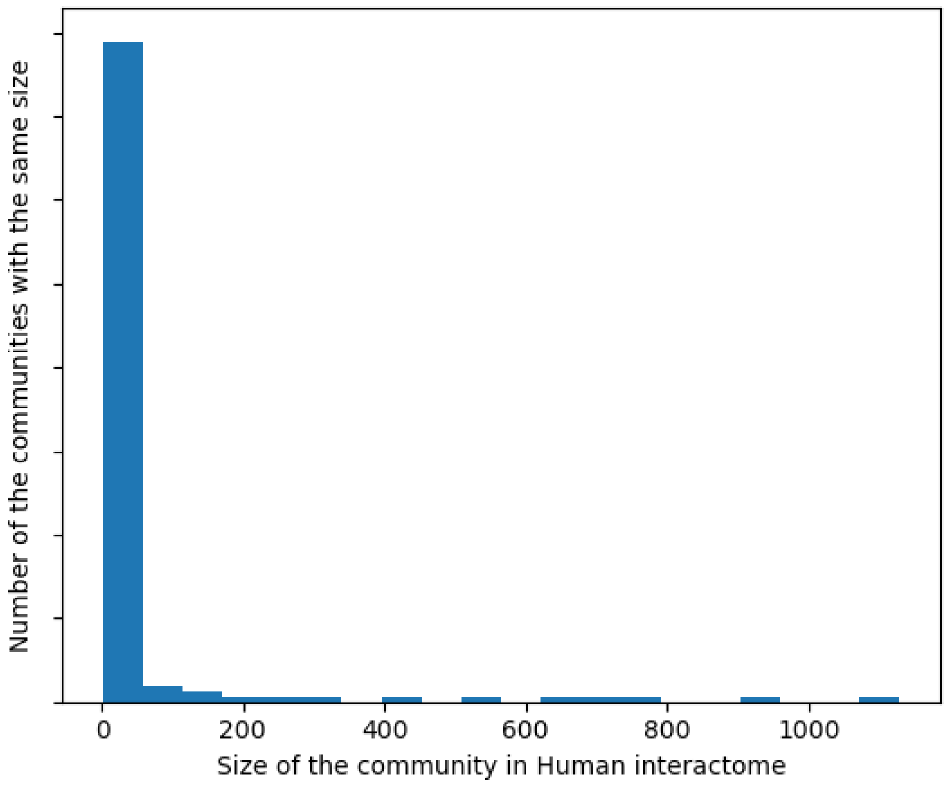
6.2 Future Work

As to the future direction, in spite of the improved accuracy over the standard clustering methods as well as over the basic autoencoder, our clustering protocol could be further optimized in several ways. For example, one can explore other distance metrics that have been previously shown to perform well in the clustering with homogenous features [5]. Alternatively, we plan to investigate if the clustering performance can be improved by supplying the complementary biological information. For example, instead of the gene community information used in this work, the shortest path between two nodes in the network can be considered, since the former sometimes provides more accurate information than the latter.

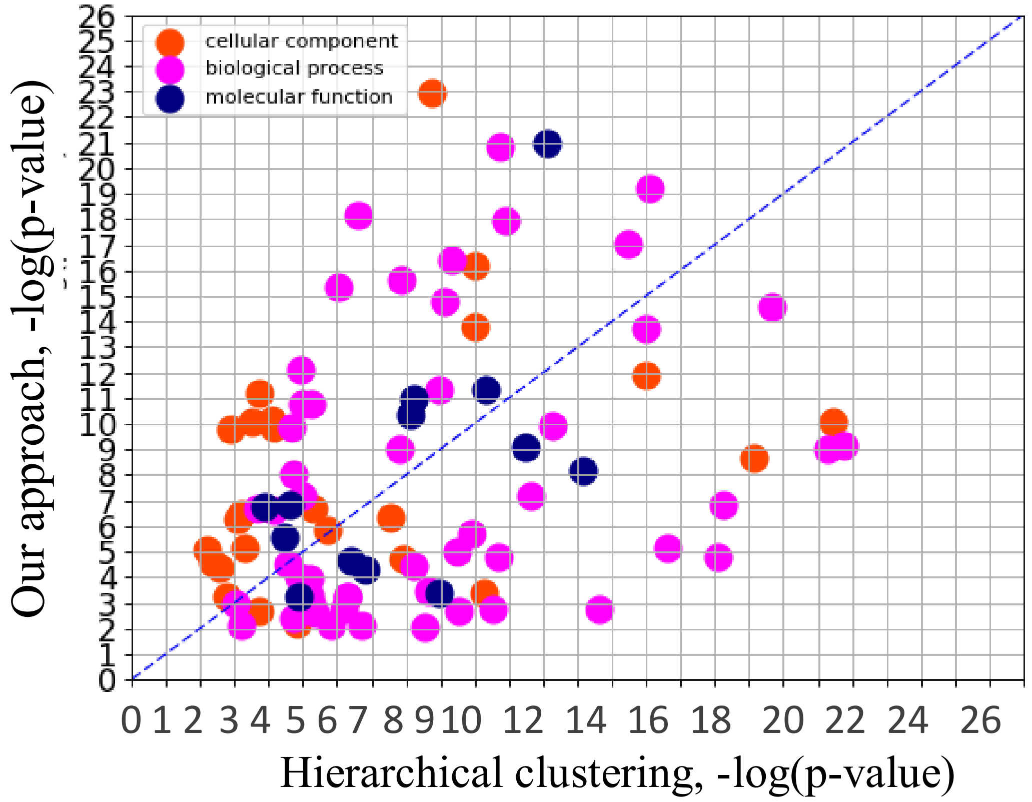
Appendix



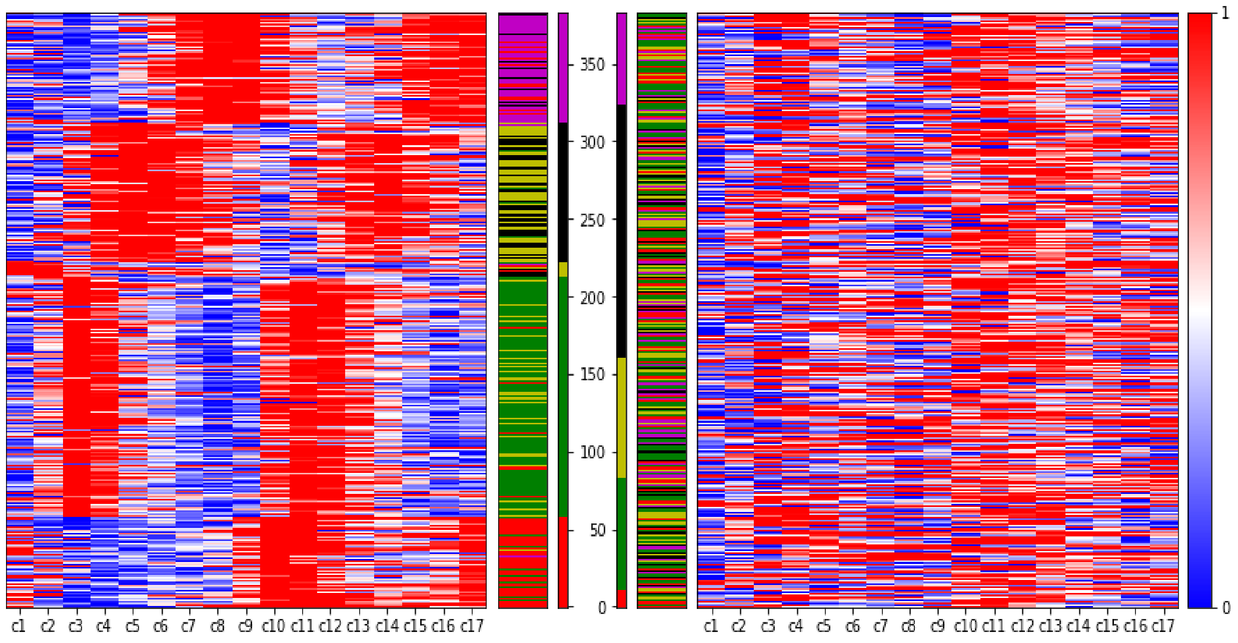
Supplementary Figure S1. Distribution of communities with different sizes in the yeast interactome.



Supplementary Figure S2. Distribution of communities with different sizes in the human interactome



Supplementary Figure S3. Comparison of enriched Gene Ontology (GO) terms between our protocol and traditional hierarchical clustering for the human gene expression dataset. The values are converted using negative log of p-value function. A smaller p-value reflects a larger proportion of the cluster members sharing the same GO term.



Supplementary Figure S4. Performance of our approach (left) against the base line hierarchical clustering (right) on the yeast gene expression dataset provides a visibly better clustering into 5 previously labeled gene classes across 17 different time stamps (c1-c17).

Supplementary Table 1 Comparison of the result of GO enrichment analysis based on the protocol and k-means clustering.

GO term	new approach p-value	kmeans p-value
GO:0044459~plasma membrane part	1.07E-21	1.49E-05
GO:0005102~receptor binding	7.97E-10	5.78E-07
GO:0003013~circulatory system process	1.06E-04	3.27E-03
GO:0048513~organ development	6.77E-24	2.66E-08
GO:0007165~signal transduction	6.61E-08	1.16E-06
GO:0031224~intrinsic to membrane	1.25E-12	1.02E-05
GO:0009653~anatomical structure morphogenesis	5.27E-12	3.49E-07
GO:0048731~system development	3.93E-35	8.74E-10
GO:0005615~extracellular space	2.89E-27	1.52E-07
GO:0016020~membrane	8.10E-11	1.04E-03
GO:0031012~extracellular matrix	1.20E-23	1.55E-12
GO:0030154~cell differentiation	1.41E-16	1.11E-03
GO:0004872~receptor activity	5.33E-11	1.22E-13
GO:0050877~neurological system process	1.24E-12	3.24E-10
GO:0007267~cell-cell signaling	2.87E-15	1.95E-03
GO:0009888~tissue development	9.72E-10	4.47E-03
GO:0009611~response to wounding	6.30E-08	8.71E-03
GO:0005578~proteinaceous extracellular matrix	9.83E-25	5.37E-14
GO:0044425~membrane part	2.32E-09	4.84E-04
GO:0016337~cell-cell adhesion	9.12E-10	1.02E-06
GO:0022803~passive transmembrane transporter activity	6.55E-09	7.91E-03
GO:0005886~plasma membrane	3.76E-18	3.60E-06

Supplementary Table 2 Comparison of the result of GO enrichment analysis based on the protocol and hierarchical clustering.

GO term	new approach p-value	hierarchical p-value
GO:0044459~plasma membrane part	1.07E-21	4.94E-43
GO:0006935~chemotaxis	5.79E-04	5.14E-07
GO:0051239~regulation of multicellular organismal process	7.32E-10	1.78E-21
GO:0070727~cellular macromolecule localization	1.94E-07	2.07E-04
GO:0006952~defense response	4.89E-04	6.60E-06
GO:0005102~receptor binding	7.97E-10	3.41E-12
GO:0003013~circulatory system process	1.06E-04	6.56E-06
GO:0032553~ribonucleotide binding	4.32E-11	7.77E-09
GO:0030198~extracellular matrix organization	2.34E-03	4.63E-06
GO:0044424~intracellular part	1.60E-109	5.52E-75
GO:0016817~hydrolase activity, acting on acid anhydrides	2.82E-06	3.43E-05
GO:0048519~negative regulation of biological process	3.68E-03	1.94E-05
GO:0009986~cell surface	4.60E-07	2.90E-08
GO:0030247~polysaccharide binding	4.00E-04	1.10E-09
GO:0009057~macromolecule catabolic process	9.60E-18	3.36E-15
GO:0042330~taxis	5.79E-04	5.14E-07
GO:0016324~apical plasma membrane	6.69E-06	5.28E-04
GO:0031090~organelle membrane	7.62E-11	8.47E-05
GO:0043233~organelle lumen	9.15E-80	1.79E-55
GO:0044057~regulation of system process	7.01E-03	1.56E-06
GO:0045177~apical part of cell	4.32E-05	2.66E-03
GO:0070013~intracellular organelle lumen	1.52E-84	8.91E-62

GO:0048513~organ development	6.77E-24	5.69E-46
GO:0051240~positive regulation of multicellular organismal process	8.31E-03	2.97E-09
GO:0015031~protein transport	1.66E-11	5.70E-06
GO:0051082~unfolded protein binding	1.60E-07	1.39E-04
GO:0045184~establishment of protein localization	1.76E-11	9.90E-06
GO:0043229~intracellular organelle	4.35E-93	5.55E-75
GO:0051276~chromosome organization	9.57E-09	1.91E-05
GO:0022403~cell cycle phase	4.57E-16	8.93E-07
GO:0006811~ion transport	1.85E-06	1.28E-10
GO:0006974~response to DNA damage stimulus	5.95E-20	8.31E-16
GO:0006928~cell motion	1.82E-03	2.29E-14
GO:0048193~Golgi vesicle transport	1.08E-03	8.91E-04
GO:0042995~cell projection	1.37E-06	1.96E-06
GO:0022603~regulation of anatomical structure morphogenesis	3.34E-05	6.06E-09
GO:0009725~response to hormone stimulus	7.78E-03	2.01E-07
GO:0046930~pore complex	3.33E-07	6.31E-04
GO:0048468~cell development	7.66E-06	2.50E-16
GO:0008285~negative regulation of cell proliferation	7.16E-03	6.45E-04
GO:0050793~regulation of developmental process	1.51E-05	8.42E-18
GO:0019866~organelle inner membrane	1.47E-10	6.90E-05
GO:0005626~insoluble fraction	6.22E-03	1.43E-05
GO:0007165~signal transduction	6.61E-08	1.09E-05
GO:0019538~protein metabolic process	2.19E-16	1.42E-08
GO:0042127~regulation of cell proliferation	1.55E-05	2.10E-11
GO:0048518~positive regulation of biological process	1.92E-03	2.93E-10

GO:0044260~cellular macromolecule metabolic process	2.61E-46	5.10E-41
GO:0031224~intrinsic to membrane	1.25E-12	1.03E-15
GO:0010817~regulation of hormone levels	1.83E-03	6.42E-07
GO:0051094~positive regulation of developmental process	3.32E-04	2.36E-09
GO:0044444~cytoplasmic part	4.20E-44	1.48E-19
GO:0009653~anatomical structure morphogenesis	5.27E-12	3.04E-29
GO:0051649~establishment of localization in cell	2.35E-07	7.40E-05
GO:0009887~organ morphogenesis	1.43E-07	5.72E-18
GO:0008104~protein localization	1.48E-10	2.11E-05
GO:0031966~mitochondrial membrane	8.72E-11	3.18E-04
GO:0044428~nuclear part	1.88E-88	1.36E-62
GO:0005740~mitochondrial envelope	6.05E-12	1.93E-04
GO:0048731~system development	3.93E-35	8.76E-71
GO:0003713~transcription coactivator activity	5.27E-04	1.35E-05
GO:0016879~ligase activity, forming carbon-nitrogen bonds	5.05E-05	1.51E-07
GO:0005615~extracellular space	2.89E-27	2.57E-28
GO:0034641~cellular nitrogen compound metabolic process	4.82E-28	1.05E-28
GO:0005643~nuclear pore	5.47E-07	7.94E-04
GO:0016020~membrane	8.10E-11	3.55E-21
GO:0044429~mitochondrial part	1.10E-27	6.15E-15
GO:0031012~extracellular matrix	1.20E-23	2.31E-35
GO:0034702~ion channel complex	1.87E-07	5.02E-06
GO:0046907~intracellular transport	4.45E-12	1.14E-09
GO:0030154~cell differentiation	1.41E-16	9.34E-41
GO:0060348~bone development	1.11E-03	6.05E-06
GO:0043232~intracellular non-membrane-bounded organelle	7.94E-31	1.11E-20
GO:0010467~gene expression	1.99E-14	1.02E-15

GO:0009059~macromolecule biosynthetic process	9.33E-06	3.08E-10
GO:0003723~RNA binding	1.13E-21	7.55E-13
GO:0007267~cell-cell signaling	2.87E-15	2.27E-19
GO:0008284~positive regulation of cell proliferation	9.14E-05	1.40E-05
GO:0045595~regulation of cell differentiation	1.73E-03	2.88E-11
GO:0005622~intracellular	7.56E-112	7.98E-77
GO:0030529~ribonucleoprotein complex	2.11E-32	3.32E-19
GO:0009888~tissue development	9.72E-10	5.04E-21
GO:0009611~response to wounding	6.30E-08	2.33E-12
GO:0008134~transcription factor binding	1.49E-07	2.35E-05
GO:0044420~extracellular matrix part	4.02E-04	5.54E-11
GO:0031967~organelle envelope	1.06E-23	1.71E-09
GO:0044248~cellular catabolic process	4.26E-17	4.84E-10
GO:0044249~cellular biosynthetic process	1.24E-10	5.44E-13
GO:0022891~substrate-specific transmembrane transporter activity	2.37E-05	4.27E-07
GO:0033554~cellular response to stress	1.04E-18	1.26E-11
GO:0009897~external side of plasma membrane	1.81E-05	1.27E-08
GO:0022402~cell cycle process	7.00E-19	2.46E-07
GO:0048285~organelle fission	1.75E-15	7.66E-10
GO:0000776~kinetochore	8.37E-06	6.06E-03
GO:0043005~neuron projection	5.89E-04	1.59E-03
GO:0017076~purine nucleotide binding	1.06E-11	6.06E-09
GO:0005737~cytoplasm	1.39E-54	3.90E-22
GO:0044427~chromosomal part	1.59E-14	9.29E-11
GO:0043231~intracellular membrane-bounded organelle	1.64E-101	4.98E-81
GO:0044446~intracellular organelle part	1.29E-101	4.31E-67

GO:0001883~purine nucleoside binding	4.92E-12	4.59E-11
GO:0005578~proteinaceous extracellular matrix	9.83E-25	3.88E-35
GO:0006139~nucleobase, nucleoside, nucleotide and nucleic acid metabolic process	5.31E-29	5.24E-31
GO:0005635~nuclear envelope	1.52E-10	1.24E-03
GO:0007565~female pregnancy	3.01E-05	2.90E-05
GO:0044425~membrane part	2.32E-09	7.32E-19
GO:0042254~ribosome biogenesis	7.11E-13	1.11E-05
GO:0016337~cell-cell adhesion	9.12E-10	1.57E-08
GO:0000151~ubiquitin ligase complex	2.53E-05	4.16E-03
GO:0005604~basement membrane	2.00E-03	1.89E-04
GO:0000278~mitotic cell cycle	1.38E-21	1.84E-11
GO:0022803~passive transmembrane transporter activity	6.55E-09	7.33E-14
GO:0005886~plasma membrane	3.76E-18	5.72E-35
GO:0005681~spliceosome	6.51E-17	9.93E-11

Bibliography

1. Lockhart, D.J. and E.A. Winzeler, *Genomics, gene expression and DNA arrays.* Nature, 2000. **405**(6788): p. 827-836.
2. Wang, Z., M. Gerstein, and M. Snyder, *RNA-Seq: a revolutionary tool for transcriptomics.* Nature reviews genetics, 2009. **10**(1): p. 57.
3. Edfors, F., et al., *Gene-specific correlation of RNA and protein levels in human cells and tissues.* Molecular Systems Biology, 2016. **12**(10): p. 883.
4. Lovén, J., et al., *Revisiting global gene expression analysis.* Cell, 2012. **151**(3): p. 476-482.
5. Belacel, N., Q. Wang, and M. Cuperlovic-Culf, *Clustering methods for microarray gene expression data.* Omics: a journal of integrative biology, 2006. **10**(4): p. 507-531.
6. Lyons-Weiler, J., S. Patel, and S. Bhattacharya, *A classification-based machine learning approach for the analysis of genome-wide expression data.* Genome research, 2003. **13**(3): p. 503-512.
7. Kuo, W.P., et al., *A primer on gene expression and microarrays for machine learning researchers.* Journal of Biomedical Informatics, 2004. **37**(4): p. 293-303.
8. D'Haeseleer, P., *How does gene expression clustering work?* Nature biotechnology, 2005. **23**(12): p. 1499.
9. Hartigan, J.A. and M.A. Wong, *Algorithm AS 136: A k-means clustering algorithm.* Journal of the Royal Statistical Society. Series C (Applied Statistics), 1979. **28**(1): p. 100-108.
10. Moon, T.K., *The expectation-maximization algorithm.* IEEE Signal processing magazine, 1996. **13**(6): p. 47-60.
11. Ciregan, D., U. Meier, and J. Schmidhuber. *Multi-column deep neural networks for image classification*. in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. 2012. IEEE.
12. Collobert, R. and J. Weston. *A unified architecture for natural language processing: Deep neural networks with multitask learning*. in *Proceedings of the 25th international conference on Machine learning*. 2008. ACM.
13. LeCun, Y., Y. Bengio, and G. Hinton, *Deep learning.* Nature, 2015. **521**(7553): p. 436-444.
14. Chen, Y., et al., *Gene expression inference with deep learning.* Bioinformatics, 2016. **32**(12): p. 1832-1839.
15. Chen, L., et al., *Learning a hierarchical representation of the yeast transcriptomic machinery using an autoencoder model.* BMC bioinformatics, 2016. **17**(1): p. S9.
16. Barabasi, A.-L. and Z.N. Oltvai, *Network biology: understanding the cell's functional organization.* Nature reviews genetics, 2004. **5**(2): p. 101-113.
17. Cui, H., et al., *The variation game: Cracking complex genetic disorders with NGS*

and omics data. Methods, 2015. **79**: p. 18-31.
18. Rolland, T., et al., *A proteome-scale map of the human interactome network.* Cell, 2014. **159**(5): p. 1212-1226.
19. Alhindi, T., et al., *Protein interaction evolution from promiscuity to specificity with reduced flexibility in an increasingly complex network.* Scientific Reports, 2017. **7**: p. 44948.
20. Han, J.-D.J., et al., *Evidence for dynamically organized modularity in the yeast protein–protein interaction network.* Nature, 2004. **430**(6995): p. 88-93.
21. Leskovec, J., et al. *Statistical properties of community structure in large social and information networks*. in *Proceedings of the 17th international conference on World Wide Web*. 2008. ACM.
22. Zhou, C.P., Randy *Anomaly Detection with Robust Deep Auto-encoders*. in *Proceedings of the 23th ACM SIGKDD international conference on Knowledge discovery and data mining*. 2017. Halifax, Nova Scotia - Canada: ACM.
23. Eisen, M.B., et al., *Cluster analysis and display of genome-wide expression patterns.* Proceedings of the National Academy of Sciences, 1998. **95**(25): p. 14863-14868.
24. Schena, M., et al., *Quantitative monitoring of gene expression patterns with a complementary DNA microarray.* Science, 1995. **270**(5235): p. 467-470.
25. Jiang, D., C. Tang, and A. Zhang, *Cluster analysis for gene expression data: A survey.* IEEE Transactions on Knowledge & Data Engineering, 2004(11): p. 1370-1386.
26. Jiang, D., C. Tang, and A. Zhang, *Cluster analysis for gene expression data: a survey.* IEEE Transactions on knowledge and data engineering, 2004. **16**(11): p. 1370-1386.
27. Johnson, S.C., *Hierarchical clustering schemes.* Psychometrika, 1967. **32**(3): p. 241-254.
28. Sherlock, G., *Analysis of large-scale gene expression data.* Current opinion in immunology, 2000. **12**(2): p. 201-205.
29. Wold, S., K. Esbensen, and P. Geladi, *Principal component analysis.* Chemometrics and intelligent laboratory systems, 1987. **2**(1-3): p. 37-52.
30. De la Torre, F. and M.J. Black. *Robust principal component analysis for computer vision*. in *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*. 2001. IEEE.
31. Hubert, M. and S. Engelen, *Robust PCA and classification in biosciences.* Bioinformatics, 2004. **20**(11): p. 1728-1736.
32. Schmidhuber, J., *Deep learning in neural networks: An overview.* Neural networks, 2015. **61**: p. 85-117.
33. Vincent, P., et al. *Extracting and composing robust features with denoising autoencoders*. in *Proceedings of the 25th international conference on Machine learning*. 2008. ACM.
34. Vincent, P., et al., *Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion.* Journal of machine learning research, 2010. **11**(Dec): p. 3371-3408.
35. Newman, M.E., *Modularity and community structure in networks.* Proceedings of the national academy of sciences, 2006. **103**(23): p. 8577-8582.
36. Palla, G., et al., *Uncovering the overlapping community structure of complex networks in nature and society.* nature, 2005. **435**(7043): p. 814.
37. Girvan, M. and M.E. Newman, *Community structure in social and biological networks.* Proceedings of the national academy of sciences, 2002. **99**(12): p. 7821-

7826.
38. Barabasi, A.-L. and Z.N. Oltvai, *Network biology: understanding the cell's functional organization.* Nature reviews genetics, 2004. **5**(2): p. 101.
39. Barabási, A.-L., N. Gulbahce, and J. Loscalzo, *Network medicine: a network-based approach to human disease.* Nature reviews genetics, 2011. **12**(1): p. 56.
40. Costa, L.d.F., et al., *Characterization of complex networks: A survey of measurements.* Advances in physics, 2007. **56**(1): p. 167-242.
41. Zhou, C. and R.C. Paffenroth. *Anomaly detection with robust deep autoencoders*. in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. 2017. ACM.
42. Wright, J., et al. *Robust principal component analysis: Exact recovery of corrupted low-rank matrices via convex optimization*. in *Advances in neural information processing systems*. 2009.
43. Boyd, S., et al., *Distributed optimization and statistical learning via the alternating direction method of multipliers.* Foundations and Trends® in Machine Learning, 2011. **3**(1): p. 1-122.
44. De Meo, P., et al. *Generalized louvain method for community detection in large networks*. in *Intelligent Systems Design and Applications (ISDA), 2011 11th International Conference on*. 2011. IEEE.
45. Herrero, J., R. Díaz-Uriarte, and J. Dopazo, *Gene expression data preprocessing.* Bioinformatics, 2003. **19**(5): p. 655-656.
46. Yeung, K.Y. and W.L. Ruzzo, *Principal component analysis for clustering gene expression data.* Bioinformatics, 2001. **17**(9): p. 763-774.
47. Gupta, A., H. Wang, and M. Ganapathiraju. *Learning structure in gene expression data using deep architectures, with an application to gene clustering*. in *Bioinformatics and Biomedicine (BIBM), 2015 IEEE International Conference on*. 2015. IEEE.
48. Yang, W., et al., *Genomics of Drug Sensitivity in Cancer (GDSC): a resource for therapeutic biomarker discovery in cancer cells.* Nucleic acids research, 2013. **41**(D1): p. D955-D961.
49. Das, J. and H. Yu, *HINT: High-quality protein interactomes and their applications in understanding human disease.* BMC systems biology, 2012. **6**(1): p. 92.
50. Quackenbush, J., *Computational analysis of microarray data.* Nature reviews genetics, 2001. **2**(6): p. 418-427.
51. Yeung, K.Y. and W.L. Ruzzo, *Details of the adjusted rand index and clustering algorithms, supplement to the paper an empirical study on principal component analysis for clustering gene expression data.* Bioinformatics, 2001. **17**(9): p. 763-774.
52. Ashburner, M., et al., *Gene Ontology: tool for the unification of biology.* Nature genetics, 2000. **25**(1): p. 25-29.
53. Huang, D.W., B.T. Sherman, and R.A. Lempicki, *Bioinformatics enrichment tools: paths toward the comprehensive functional analysis of large gene lists.* Nucleic acids research, 2009. **37**(1): p. 1-13.
54. Bollobás, B. *The Evolution of Random Graphs—the Giant Component*. in *Random Graphs*. 2001.
55. de Souto, M.C., et al., *Clustering cancer gene expression data: a comparative study.* BMC bioinformatics, 2008. **9**(1): p. 497.
56. Freyhult, E., et al., *Challenges in microarray class discovery: a comprehensive examination of normalization, gene selection and clustering.* BMC bioinformatics, 2010. **11**(1): p. 503.

57. Spencer, W.C., et al., *A spatial and temporal map of C. elegans gene expression.* Genome research, 2011. **21**(2): p. 325-341.
58. Zang, S., et al., *Identification of differentially-expressed genes in intestinal gastric cancer by microarray analysis.* Genomics, proteomics & bioinformatics, 2014. **12**(6): p. 276-283.
59. Thomou, T., et al., *Adipose-derived circulating miRNAs regulate gene expression in other tissues.* Nature, 2017. **542**(7642): p. 450.
60. Andreopoulos, B., et al., *A roadmap of clustering algorithms: finding a match for a biomedical application.* Briefings in Bioinformatics, 2009. **10**(3): p. 297-314.
61. Manfield, I.W., et al., *Arabidopsis Co-expression Tool (ACT): web server tools for microarray-based gene expression analysis.* Nucleic acids research, 2006. **34**(suppl_2): p. W504-W509.
62. Consortium, A.I.M., *Evidence for network evolution in an Arabidopsis interactome map.* Science, 2011. **333**(6042): p. 601-607.
63. Vo, T.V., et al., *A proteome-wide fission yeast interactome reveals network evolution principles from yeasts to human.* Cell, 2016. **164**(1): p. 310-323.