EPiC Series in Computing

Volume 70, 2020, Pages 129–138

Proceedings of the 12th International Conference on Bioinformatics and Computational Biology

Mining approximate frequent dense modules from multiple gene expression datasets

San Ha Seo and Saeed Salem

Department of Computer Science,
North Dakota State University,
Fargo, North Dakota, U.S.A
sanha.seo@ndsu.edu ; saeed.salem@ndsu.edu

Abstract

Large amount of gene expression data has been collected for various environmental and biological conditions. Extracting co-expression networks that are recurrent in multiple co-expression networks has been shown promising in functional gene annotation and biomarkers discovery. Frequent subgraph mining reports a large number of subnetworks. In this work, we propose to mine approximate dense frequent subgraphs. Our proposed approach reports representative frequent subgraphs that are also dense. Our experiments on real gene coexpression networks show that frequent subgraphs are biologically interesting as evidenced by the large percentage of biologically enriched frequent dense subgraphs.

1 Introduction

Adavances in DNA microarray technology have enabled the collection and analysis of huge amount of gene expression data. Gene expression datasets can be clustered by genes, samples, or both genes and samples simultaneously. In gene-based clustering, each cluster of genes may correspond to co-functional and/or co-regulated genes. In sample-based clustering, each cluster may correspond to disease or cancer types. In general, only a subset of genes and a subset of samples are related to a particular biological process [10]. Therefore, in biclustering, genes and samples are clustered simultaneously. In this work, we focus on gene-based clustering.

1.1 Mining Single Gene Expression Dataset

Clustering coexpressed genes have proven useful in understanding gene function and gene regulation. Coexpressed genes are likely to have similar biological functions, and clustering coexpressed genes can help predict previously unknown gene functions based on the genes with known functions in the same cluster [3]. Coexpressed genes are also likely to be co-regulated. Clustering coexpressed genes can help identify regulatory motifs by searching for common DNA sequences at the promoter regions of the genes in the same cluster [2]. Various conventional clustering methods have been employed for finding groups in gene expression data, including k-means [6] and hierarchical [3] approaches. While these general-purpose algorithms have

Q. Ding, O. Eulenstein and H. Al-Mubaid (eds.), BICOB 2020 (EPiC Series in Computing, vol. 70), pp. 129–138

proven useful in some applications, they have not been effective in many applications [10]. Several algorithms designed specifically for gene expression data have been proposed [9, 19]. [19] proposed a graph-theoretic approach to clustering gene expression data. Dataset is first represented as a coexpression network, which is a weighted graph where each node corresponds to a gene and each edge corresponds to a 'coexpression link' (highly correlated pair of genes). The coexpression network is split recursively based on the minimum cut. It has been shown that these algorithms perform better than the conventional approaches on some datasets.

1.2 Integrating Multiple Gene Expression Datasets

For experimental reasons, many coexpression links are of questionable biological relevance. Due to the complex procedure of microarray experiments, gene expression data often contains a lot of noise, leading to a significant number of spurious coexpression links [7]. Additionally, some coexpression may be caused by the simultaneous perturbation of multiple biological pathways in the particular experiment, rather than biological relevance [8]. These spurious coexpression link may lead to the discovery of false modules.

To overcome the problem of spurious links, recent studies have focused on integrating multiple gene expression datasets to discover gene clusters that appear across multiple datasets, based on the expectation that biological modules are active across multiple datasets. Graph theoretic approaches have been used in many of these studies. Each gene expression dataset is modelled as a gene coexpression network, which is a graph where each node corresponds to a gene and each edge corresponds to a coexpression link between two genes. In [13], the authors combined frequent coexpression links in multiple coexpression networks to build a summary graph, and applied hierarchical clustering and the MCODE [1] algorithms on the summary graph to discover highly connected modules. It was shown that coexpression links present in many datasets are more likely to represent known functional relevance. However, directly clustering an aggregated graph may result in the discovery of false positive modules, which are not dense in the original set of networks. The edges in these modules may be scattered across the networks such that they are highly connected in the aggregated graph, but neither frequent nor highly connected in the original networks [7]. Several algorithms have been proposed to address this problem [7, 8, 17].

The CODENSE [7] algorithm efficiently mines coherent dense subgraphs across large number of graphs. A coherent subgraph is a subgraph whose edges show highly correlated occurrences across the whole graph set. The algorithm first builds a summary graph and mines the dense subgraphs in the summary graph. Then for each dense summary subgraph, it constructs the second-order graph, which illustrates the co-occurrence of edges across the original graph set. Finally, dense subgraphs in the second-order graph are extracted as the final results. The algorithm is able to overcome the false positive module problem by exploiting the property that a coherent subgraph's second-order graph must be dense. Another notable feature of CODENSE is its ability to mine overlapping modules. It is important because, in general, one gene may be involved in multiple biological processes.

In the approach proposed by Huang et al. [8], the coexpression graphs are represented as a binary edge occurrence matrix where each row corresponds to an edge, each column corresponds to a graph, and each entry indicates the presence of the edge in the graph. Frequent itemset mining technique is employed to mine frequent edgesets from the edge occurrence matrix. These frequent edgesets serve as seeds for a biclustering algorithm, which uses simulated annealing to maximize an objective function such that the discovered biclusters tend to be large and have high density of ones. Connected components in the biclusters are returned as the final output

modules. The discovered modules are frequent but not necessarily highly connected.

The MFMS [17] algorithm mines maximal frequent collections of k-cliques and percolated k-cliques across many coexpression networks. The coexpression networks are first represented as the binary edge occurrence matrix. Maximal itemset mining algorithm (GenMax [5]) is then used on the edge occurrence matrix to mine maximal frequent edgesets. Cliques and percolated cliques are extracted from the subgraph induced by each maximal frequent edgeset. Mining maximal frequent edgesets is equivalent to mining exact biclusters, that is, biclusters whose entries are all ones. This means the discovered modules cannot contain noise. This may be too restrictive and may fail to identify biologically relevant modules that contain some noise due to the noisy nature of gene expression data. Salem et. al. [18, 16] proposed an approach that constructs a weighted network whose nodes corresponds to the original edges in the coexpression networks. The weight between two edges is calculated as a combined score based on the topological similarity between the edges and the occurrence similarity.

In this work, we develop an algorithm to mine approximate frequent dense modules, which are frequent dense modules that may contain some noise. First, we construct the edge occurrence matrix from the set of coexpression networks. Then biclusters with high density are mined from the edge occurrence matrix. Finally, we extract dense modules from the subgraph induced by each bicluster.

2 Methods

Gene coexpression networks have a property that each gene occurs only once in the network. This type of network can be modelled as a relation graph, where each node has a unique label. A relation graph set is a set of graphs that share a common set of nodes.

Relation Graph Set: A relation graph set is a set of n graphs $\mathcal{G} = \{G_1, G_2, \ldots, G_n\}$ where $G_i = (V, E_i)$ and $E_i \subseteq V \times V$. A common set of nodes V is shared by all graphs.

To set a common framework for the discussion of the different methods, we represent the n graphs as a summary graph $G(V, E)$ and an associated binary edge occurrence matrix, $\mathcal{B}$. Each row of the matrix is a binary vector whose entries represent the presence/absence of the edge in the corresponding graph.

Summary Graph and Edge Occurrence Matrix: Given a relation graph set $\mathcal{G} = \{G_1, G_2, \ldots, G_n\}$ where $G_i = (V, E_i)$, let $E = \{e_1, e_2, \ldots, e_m\} = \bigcup_{i=1}^{n} E_i$. The edge occurrence matrix $\mathcal{B}$ is an $m \times n$ binary matrix where $\mathcal{B}_{ij} = 1$ if $e_i \in E_j$; 0 otherwise. The relation graph set can be represented as $\mathcal{G} = (V, E, \mathcal{B})$.

Edge-Induced Subgraph: Given a graph $G(V, E)$ and an edgeset $E' \subseteq E$, the edge-induced subgraph $G'(V', E')$ of G (induced by edgeset E') is a graph whose edgeset is E' and the node set is $V' = \bigcup V(e)$ for all $e \in E'$ where $V(e)$ denotes the endpoints of e.

Figure 1(b) shows the summary graph and the occurrence matrix of the graph set in (a). Also, in (d) the edge-induced subgraphs for the edgesets in (c). A set of edges that induce a connected subgraph is called a frequent connected edgeset if the edges appear in at least a minimum number of graphs (*minsup*). Note that a frequent edgeset is not necessarily connected. The

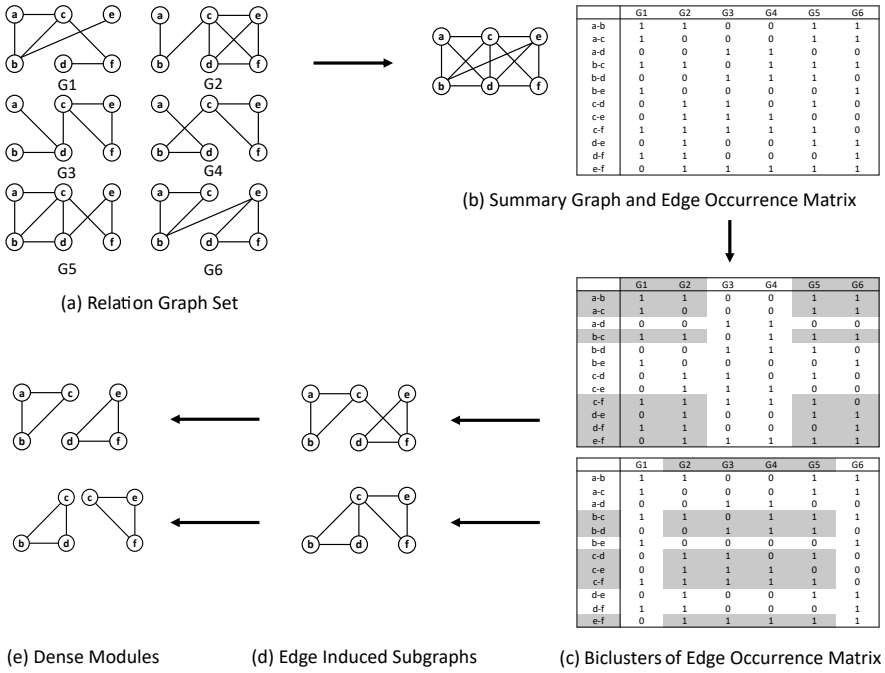


Figure 1: Steps in Mining Approximate Frequent Dense Subgraphs: (a) Graph set; (b) Summary graph and edge occurrence matrix; (c) Mining approximate frequent subgraphs; (d-e) Extracting dense modules

graph induced by a frequent edgeset can have multiple connected components in each of the supporting graphs.

In our work, however, we want to allow noise in the discovered subgraphs. Therefore, we define approximate frequent subgraph, which is a relaxed version of frequent subgraph. This definition allows the discovery of subgraphs that are 'close enough' to the exact frequent subgraphs.

Approximate Frequent Subgraph Given a relation graph set $\mathcal{G} = \{G_1, G_2, \ldots, G_n\}$ and a noise ratio r, a subgraph $G'(V', E')$ is an approximate frequent subgraph if and only if there exists a graph set $D \subseteq \mathcal{G}$ such that $|D| \geq minsup$ and for every edge $e \in E'$, e occurs in at least $ceiling(|D| * (1 - r))$ graphs in D.

The noise ratio r is a real number between 0 and 1, and represents the maximum noise allowed. An r value of zero means no noise is allowed. Typically we want to keep this value close to zero. In our definition of approximate frequent subgraph, an edge e need not be present in all graphs in D, as long as it occurs in most of them. In our work, we are interested in discovering frequent subgraphs that are dense. Both the density and the recurrence of the subgraph increase the likelihood that the subgraph is biologically meaningful.

Graph Density: The density of a graph G is $2m/(n(n-1))$ where m is the number of edges and n is the number of nodes in G. G is dense if its density is greater than or equal to a minimum density threshold.

Our problem is formulated as follows: Given a relation graph set $\mathcal{G} = \{G_1, G_2, \ldots, G_n\}$, minimum support, noise, and density thresholds, find subgraphs that are both approximate frequent and dense. We follow a two-steps approach to mine these patterns. The first step is to extract edgesets that have similar edges' occurrence, and the second step is to extract dense subgraphs from the induced subgraph of each edgeset. For the second step, we use the Dense Module Enumeration method [4]. We first discuss approaches to mine frequent edgesets from the binary edge occurrence matrix.

2.1 Frequent Itemset Mining

Frequent itemset mining is one of the most commonly used techniques in pattern mining. Its goal is to find sets of objects that frequently occur together. Frequent itemset mining has many applications in various areas including marketing, social media, and bioinformatics, among others. Here, we briefly explain frequent itemset mining, and then will explain how frequent edgesets are mined. Let $E = \{e_1, e_2, \ldots, e_{|E|}\}$ be the set of the union of all the edges in the graphs. The graph set is represented as a set of transactions defined over the set of edges in the graphs. Each graph is essentially a subset of edges from the entire edgesets, i.e., $E_i \subseteq E$. Given a minimum support threshold *minsup*, a set of edges $E' \subseteq E$ is called a frequent edge set if it appear in at least *minsup* graphs in the relation graph set. Let $S(E')$ denote the set of graph ids of the graphs that contain E'. The support of an edgeset E' is $|S(E')|$.

Because of the anti-monotone property of the support of an edgeset, all subsets of a frequent edgeset are frequent. The frequent edgesets would have large overlap between the patterns and the number of these frequent patterns will be large. To overcome this problem, only the maximal frequent patterns are mined. An edgeset is a maximal frequent edgeset if it is frequent and none of its superset is maximal. We employ the GenMax algorithm for mining all maximal frequent edgesets from the graphs [5]. The set of maximal frequent edgesets is defined as follow:

$$M = \{M_1, M_2, \cdots, M_{|M|}\}$$

where each M_i is a maximal frequent edgeset.

2.2 Biclustering

The effectiveness of the standard clustering methods is limited because, in general, a group of data objects may exhibit similar patterns under only a subset of attributes. The same limitation exists when clustering based on attributes. In gene expression data, for instance, a subset of genes is generally coexpressed under only a subset of samples. This limitation has led to the development of biclustering algorithms. A biclustering algorithm clusters rows and columns simultaneously, and thus is able to extract local patterns that are not discovered by the standard clustering methods. In matrix representation, a bicluster corresponds to a submatrix with high row (or column) similarities. Different approaches are employed for extracting biclusters from real datasets and binary datasets. Some biclustering algorithms for real datasets are described in [11, 20]. We will not describe these algorithms here, since we are primarily interested in biclustering binary data in this work.

Several biclustering algorithms for binary datasets have been developed [12, 21, 15]. In [12] and [21], biclusters are defined as submatrices with high density of ones. The common theme of these algorithms is to define an objective function such that a bicluster with a high score tends to be large in size and have high density of ones. In an iterative process, a bicluster that

Algorithm 1 Modified BiBit Algorithm

Input:
$\mathcal{B}$: $m \times n$binary matrix; m edges $\times$ n graphs
mnr: minimum number of rows
mnc: minimum number of columns
r: maximum noise
Output:
$\mathcal{X}$: list of final biclusters

```
1.   for every edge pair (i, j) do
2.     S(i, j) = S(i) ∩ S(j)
3.     if S(i, j) is new and |S(i, j)| ≥ mnc then
4.       (I, S(i, j)) = ({i, j}, S(i, j))
5.       for Every remainder edge, q ∈ E \ I  do
6.         if  |S(q) ∩ S(i, j)|/|S(i, j)| ≥ 1 − r. then
7.           I = I ∪ q
8.         end if
9.       end for
10.      if |I| ≥ mnr then
11.        X = X ∪ (I, S(i, j))
12.      end if
13.    end if
14.  end for
```

maximizes the objective function is found. These algorithms can only find one bicluster in each run. Therefore, the discovered bicluster is masked from the input matrix in the subsequent runs. As the result, these algorithms are unable to discover overlapping biclusters.

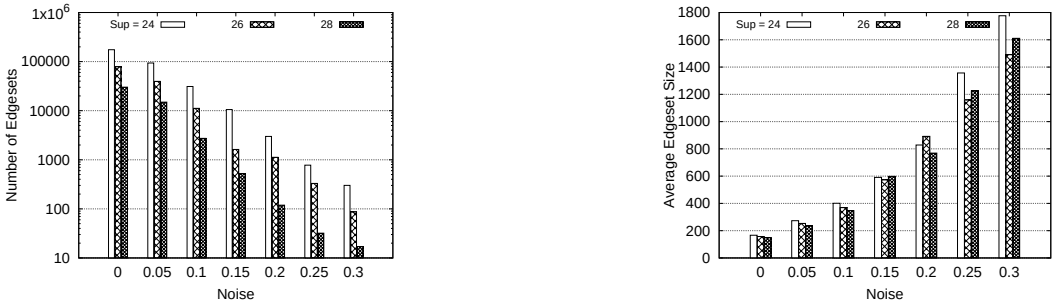
BiBit [15] is another biclustering algorithm for binary data. BiBit algorithm is capable of discovering overlapping biclusters and is easily extended to handle noisy data. A modified version of BiBit is used in this work. The BiBit algorithm is explained in the following subsection.

2.2.1 BiBit (Bit-Pattern Biclustering)

BiBit [15] is a biclustering algorithm for binary datasets. It is relatively efficient and robust to density and size of input data.

Given an $m \times n$ binary matrix $\mathcal{B}$ in which rows correspond to edges and columns correspond to graphs, let $S(i)$ denote the set of column (graph) indices j such that $\mathcal{B}_{ij} = 1$, i.e., $S(i) = \{j \mid \mathcal{B}_{ij} = 1\}$, and let $S(i,j) = S(i) \cap S(j)$. The Bibit algorithm selects a pair of rows i and j and generates the bit-pattern $(\{i,j\}, S(i,j))$. A bit-pattern is a tuple of the set of rows and its supporting columns. The bit-pattern $(\{i,j\}, S(i,j))$ is used as a seed pattern for a bicluster and $S(i,j)$ represents the column set for the bicluster. Each remaining row q is added to the bicluster if $S(q) \cap S(i,j) = S(i,j)$, that is, if the set of columns in which object q appears is a superset of the set of columns in which both objects i and j appear. The result is a bicluster consisting of all ones in the columns of the seed pair. This process is repeated for every pair of rows as a seed. Only bicluster with at least mnc columns and mnr rows are returned.

While the BiBit algorithm is efficient, it is not without limits. One problem is that it may not discover every bicluster that satisfies the given condition. This happens because when the size of a seed pattern is much larger than mnc, it will miss rows that satisfy the mnc requirement but do not match the seed pattern. For example, if the seed patterns appear in 90% of the graphs, this seed can only be extended with edges that appear in the same set of graphs, and



(a) Number of Edgesets

(b) Average Size of Edgeset

Figure 2: Topological Properties of Edgesets

thus limiting the number of possible extensions available for the seed pattern. This is a minor problem if the number of columns in the input binary matrix is small, but it becomes more apparent if the number of columns is large. Another limitation of the BiBit algorithm is that it is only able to discover exact biclusters, which are biclusters consisting of all ones. The BiBit algorithm can be modified to discover approximate biclusters, that is, biclusters consisting of some noise by relaxing the condition for extending a bicluster.

2.2.2 Modified BiBit Algorithm

Our algorithm extends the the BiBit algorithm [15] to discover approximate biclusters (biclusters with noise). We first explain the modified BiBit algorithm. Although the BiBit algorithm is relatively efficient, it is only able to mine exact biclusters. We extend the algorithm to allow the discovery of approximate biclusters, that is, biclusters that may contain noise. The main modification is that when considering rows to add to a bicluster, we allow addition of rows that do not necessarily match the exact pattern being considered, as long as they are 'close enough' to the pattern. The maximum allowed number of violating columns in each row is determined by the input noise ratio and the size of the particular pattern. As the result, the modified BiBit algorithm discovers biclusters whose noise to size ratio is less than or equal to r.
Figure 1(c) shows two biclusters for a noise threshold of 0.25. The first bicluster is extended from the seed (a-b,b-c) that appear in $\{G_1, G_2, G_5, G_6\}$, and all the remaining edges in the bicluster appear in at least three of these graphs.

The algorithm is illustrated in Algorithm 1. In line 1, the algorithm starts the extension process for each pair of rows (edges) in the binary matrix. In line 3, the pattern $(\{i,j\}, S(i,j))$ is considered visited (not new) if $S(i,j)$ is similar (Jaccard similarity) to the column set of an already visited pattern. In line 6, $S(q) \cap S(i,j)$ is considered approximately equal to $S(i,j)$ if it appears in 'most' of the columns of the seed pattern. For a maximum allowed noise r, row q is a valid extension if $|S(q) \cap S(i,j)|/|S(i,j)| \geq 1 - r$. Note that because the seed pattern $(\{i,j\}, S(i,j))$ must consist of all ones in the columns of the pattern, the algorithm does not discover all biclusters of a given noise ratio. A notable feature of the algorithm is that it requires noise to be distributed across the rows. This is a desirable feature because it prevents the discovery of biclusters in which all noise is concentrated in few rows, which may lead to the discovery of subgraphs with false edges.

		BiBit			with 0.1 noise			with 0.2 noise			with 0.3 noise		
α	θ	M'	$\overline{DM}$	$\overline{V'}$	M'	$\overline{DM}$	$\overline{V'}$	M'	$\overline{DM}$	$\overline{V'}$	M'	$\overline{DM}$	$\overline{V'}$
28	0.5	30171	23.5	3.5	2726	92.7	3.7	119	406.8	4.0	17	1548.1	4.2
	0.6	30171	26.9	3.0	2726	102.8	3.0	119	369	3.1	17	1105.2	3.2
	0.7	30171	1.0	3.0	2726	1.1	3.0	119	7.7	3.1	17	44.6	3.4
29	0.5	16812	22.3	3.4	869	93.4	3.7	50	406.7	4.0	5	1389.8	4.2
	0.6	16812	25.7	3.0	869	102.9	3.0	50	365.3	3.1	5	999.6	3.2
	0.7	16812	1.0	3.0	869	1.2	3.0	50	8.2	2.1	5	40.2	3.4
30	0.5	8476	21.1	3.4	377	134.3	3.8	20	518.7	4.0	1	1726	4.2
	0.6	8476	24.6	3.0	377	145.8	3.0	20	442.3	3.1	1	1207	3.2
	0.7	8476	1.0	3.0	377	1.6	3.0	20	11.6	3.1	1	49	3.5

Table 1: Topological properties of the frequent dense modules

3 Experiments

To evaluate the effectiveness of the proposed approach, we mined approximate frequent dense patterns from 35 tissue gene coexpression networks constructed by the Genetic Network Analysis Tool [14]. The coexpression networks were inferred from Genotype-Tissue Expression (GTEx) data [1]. Each coexpression network is constructed from the gene expression of non-diseased tissue samples. On average there are $14,415$ links in each network among $9,998$ genes. In total, there are $1,548,622$ unique links that appear in at least one network, $4,127$ edges that appear in at least 20 networks, and on average each link appears in 3.28 networks. Figure 2 shows how the number and the average size of approximate frequent edgesets vary for different minimum support and density thresholds. Figure 2(a) illustrates that for lower support, we get more frequent edgesets and in (b) we get noticeably smaller frequent edgesets for smaller noise thresholds.

The topological properties of the frequent dense modules is shown in Table 1. M' denotes the number of approximate frequent edgesets that have at least one dense module for the specified density threshold, $\overline{DM}$ denotes the average number of dense modules in edge-induced subgraph of each edgeset and $\overline{V}$ denotes the average size of the modules. As the minimum support is increased, the number of approximate frequent edgesets decreases, and the average number of dense modules also decreases. For larger density thresholds, the average size of the dense modules decreases. As the noise ratio is increased we get larger dense modules. For the topological analysis, only modules with at least three nodes were considered.

Biological enrichment analysis of the reported dense frequent modules shows that the modules are enriched with KEGG pathways and molecular functions. A dense frequent module is enriched if it overlaps with the geneset of a known molecular signature. We used the over-representation of genes with a specific annotation in a gene set using the hybergeometric test (with $pvalue = 0.01$). For biological terms, we used the KEGG pathway database (186 sets covering $5,241$ genes) and the GO Molecular Function Ontology ($1,645$ sets covering $15,599$ genes). Table 2 shows the percentage of frequent dense modules that are biologically enriched. As shown in the results, frequent dense modules with smaller noise ratios have higher enrichment. Enrichment with GO molecular functions is higher than KEGG pathways since there are much more molecular functions and they cover more genes. Table 3 shows the top biological

[1] https://www.gtexportal.org/

Sup	Density	BiBit		with 0.1 noise		with 0.2 noise		with 0.3 noise	
α	θ	E_{MF}	E_{KEGG}	E_{MF}	E_{KEGG}	E_{MF}	E_{KEGG}	E_{MF}	E_{KEGG}
	0.5	100	53.3	100	57.0	97.9	59.1	84.9	56.1
29	0.6	100	51.2	100	51.0	100.0	47.6	93.8	38.7
	0.5	100	52.9	100	58.9	97.3	60.4	81.6	55.3
30	0.6	100	51.0	100	50.9	100.0	47.2	92.2	38.5

Table 2: GO term enrichment analysis for frequent dense modules

KEGG pathway	Count	GO Molecular Function	Count
RIBOSOME	21256	CONSTITUENT OF RIBOSOME	21256
OXIDATIVE PHOSPHORYLATION	5760	ANTIGEN BINDING	8790
HUNTINGTONS DISEASE	5743	IMMUNOGLOBULIN BINDING	6798
ALZHEIMERS DISEASE	5741	ELECTRON TRANSFER	5447
PARKINSONS DISEASE	5741	OXIDOREDUCTASE	5006
PRION DISEASES	4028	NADH DEHYDROGENASE	5006

Table 3: Top enriched biological signatures: bibit with 0.1 noise, minsup 29, density 0.6

signatures that were enriched the most in the reported patterns for $sup = 29$, $noise = 0.1$, and $density = 0.6$.

4 Conclusion

Mining frequent dense modules from multiple gene coexpression networks has applications in functional gene annotation and biomarker discovery. We have proposed a biclustering-based approach for mining such modules. We first mine biclusters with high density, and dense modules are then extracted from the edgesets of these biclusters. A key feature of the proposed approach is that it reports representative patterns from the set of frequent patterns that has a high overlap and very computationally intensive to mine. Since the proposed approach only explore a small part of the frequent patterns search space, the running time is extremely fast and takes about a minute for support 24 and four seconds for support 30. Experiments on real gene coexpression networks show that the reported frequent dense modules are biologically enriched with known KEGG pathways and molecular functions.

Acknowledgement

This material is based upon work supported by the National Science Foundation under Grant No. RII Track-2 FEC 1826834.

References

[1] Gary D. Bader and Christopher W.V. Hogu. An automated method for finding molecular complexes in large protein interaction networks. *BMC Bioinformatics*, 4(2), 2003.

[2] Alvis Brazma and Jaak Vilo. Gene expression data analysis. *FEBS Letters*, 480(1):17–24, 2000.

[3] Michael B. Eisen, Paul T. Spellman, Patrick O. Brown, and David Botstein. Cluster analysis and display of genome-wide expression patterns. *Proceedings of the National Academy of Sciences*, 95(25):14863–14868, 1998.

[4] Elisabeth Georgii, Sabine Dietmann, Takeaki Uno, Philipp Pagel, and Koji Tsuda. Enumeration of condition-dependent dense modules in protein interaction networks. *Bioinformatics*, 25(7):933–940, 2009.

[5] Karam Gouda and Mohammed J. Zaki. GenMax: An efficient algorithm for mining maximal frequent itemsets. *Data Mining and Knowledge Discovery: An International Journal*, 11(3):223–242, Nov 2005.

[6] Laurie J. Heyer, Semyon Kruglyak, and Shibu Yooseph. Exploring expression data: identification and analysis of coexpressed genes. *Genome research*, 9 11:1106–15, 1999.

[7] Haiyan Hu, Xifeng Yan, Yu Huang, and Xianghong Jasmine Zhou. Mining coherent dense subgraphs across massive biological networks for functional discovery. *Bioinformatics*, 21 Suppl 1:i213–i221, 2005.

[8] Yu Huang, Haifeng Li, Haiyan Hu, Xifeng Yan, Michael S. Waterman, Haiyan Huang, and Xianghong Jasmine Zhou. Systematic discovery of functional modules and context-specific functional annotation of human genome. *Bioinformatics*, 23(13):i222–i229, 2007.

[9] Daxin Jiang, Jian Pei, and Aidong Zhang. Dhc: A density-based hierarchical clustering method for time series gene expression data. In *Proceedings of the 3rd IEEE Symposium on BioInformatics and BioEngineering*, BIBE '03, pages 393–, Washington, DC, USA, 2003. IEEE Computer Society.

[10] Daxin Jiang, Chun Tang, and Aidong Zhang. Cluster analysis for gene expression data: A survey. *IEEE Trans. on Knowl. and Data Eng.*, 16(11):1370–1386, November 2004.

[11] Yuval Kluger, Ronen Basri, Joseph Chang, and Mark Gerstein. Spectral biclustering of microarray data: Coclustering genes and conditions. *Genome research*, 13:703–16, 05 2003.

[12] Mehmet Koyutürk, Wojciech Szpankowski, and Ananth Grama. Biclustering gene-feature matrices for statistically significant dense patterns. pages 480 – 484, 09 2004.

[13] Homin K. Lee, Amy K. Hsu, Jon Sajdak, Jie Qin, and Paul Pavlidis. Coexpression analysis of human genes across many microarray data sets. *Genome Res.*, 14(6):1085–1094, 2004.

[14] Emma Pierson, the GTEx Consortium, Daphne Koller, Alexis Battle, and Sara Mostafavi. Sharing and specificity of co-expression networks across 35 human tissues. *PLOS Computational Biology*, 11(5):1–19, 05 2015.

[15] Domingo S. Rodriguez-Baena, Antonio J. Perez-Pulido, and Jesus S. Aguilar-Ruiz. A biclustering algorithm for extracting bit-patterns from binary datasets. *Bioinformatics*, 27(19):2738–2745, October 2011.

[16] Saeed Salem. Template edge similarity graph clustering for mining multiple gene expression datasets. *International Journal of Data Mining and Bioinformatics*, 18(1):28–39, 2017.

[17] Saeed Salem and Cagri Ozcaglar. MFMS: Maximal frequent module set mining from multiple human gene expression data sets. In *Proceedings of the 12th International Workshop on Data Mining in Bioinformatics*, BioKDD '13, pages 51–57, New York, NY, USA, 2013. ACM.

[18] Saeed Salem and Cagri Ozcaglar. Hybrid coexpression link similarity graph clustering for mining biological modules from multiple gene expression datasets. *BioData Mining*, 7(1):16, 2014.

[19] Roded Sharan and Ron Shamir. Center click: A clustering algorithm with applications to gene expression analysis. In *Proceedings of the Eighth International Conference on Intelligent Systems for Molecular Biology*, pages 307–316. AAAI Press, 2000.

[20] Heather L. Turner, Trevor C. Bailey, and Wojtek J. Krzanowski. Improved biclustering of microarray data demonstrated through systematic performance tests. *Computational Statistics Data Analysis*, 48:235–254, 2005.

[21] Miranda Uitert, Wouter Meuleman, and Lodewyk Wessels. Biclustering sparse binary genomic data. *Journal of computational biology : a journal of computational molecular cell biology*, 15:1329–45, 01 2009.